МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кваліфікаційна наукова
робота на правах рукопису

ТЕРНОВИЙ ІВАН МИХАЙЛОВИЧ

УДК 004.942 : 37.091.8

ДИСЕРТАЦІЯ
ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ПРОЕКТУВАННЯ АВТОМАТИЗОВАНОЇ СИСТЕМИ ЗАБЕЗПЕЧЕННЯ ДІЯЛЬНОСТІ СТУДЕНТСЬКОГО МІСТЕЧКА

Подається на здобуття наукового ступеня доктора філософії в галузі інформаційних технологій зі спеціальності 126 – Інформаційні системи та технології.

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ І. М. Терновий

Науковий керівник – Хамула Орест Григорович, кандидат технічних наук, професор

Львів – 2025

АНОТАЦІЯ

Терновий І. М. Інформаційна технологія проектування автоматизованої системи забезпечення діяльності студентського містечка. — Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 126 «Інформаційні системи та технології». – Національний університет «Львівська Політехніка», Львів, 2025.

У дисертації вирішено нагальну науково-прикладну задачу створення інформаційної технології автоматизованої системи забезпечення діяльності студентського містечка, зосереджуючись на виявленні та дослідженні чинників, що впливають на зазначені процеси, і застосуванні інструментів прогностичної оцінки якості на основі нечіткої логіки. Результатом виконаного дослідження є розробка структурно-функціональної моделі інформаційної технології для даного процесу.

Дисертаційна робота складається зі вступу, чотирьох розділів, висновків, переліку використаних джерел та додатків.

У **вступі** підкреслено актуальність обраної теми, що торкається автоматизації управління студентським містечком за допомогою інформаційних систем (ІС). Відзначено зростаючий інтерес до цифрових інструментів в освіті, виклики впровадження ІС в освітній та управлінській сферах, а також потребу в гнучких системах, здатних реагувати на індивідуальні потреби кожного учасника освітнього середовища. Показано зв'язок цього дослідження з науковою діяльністю кафедри мультимедійних технологій, метою якої є вдосконалення інформаційних технологій для автоматизації освітніх процесів. Сформульовано мету дослідження: розробка інформаційної технології для автоматизованого управління студентським містечком, заснованої на нечіткій логіці та системному аналізі, а також визначено основні завдання. Обґрунтовано наукову новизну, яка полягає в інтеграції багаторівневої моделі факторів впливу на функціонування ІС та

використанні нечіткої логіки для оптимізації процесів. Підкреслено практичну цінність роботи, що полягає в перспективі впровадження запропонованої технології у закладах вищої освіти для підвищення ефективності управління студентськими містечками. Описано особистий внесок автора, який включає розробку моделей і методик, а також участь у спільних публікаціях з науковим керівником. Результати дослідження представлено на низці науково-практичних конференцій. Підсумовано структуру та обсяг дисертаційної роботи.

У **першому розділі** зосереджено увагу на актуальному стані та викликах, з якими стикаються при використанні інтелектуальних систем (ІС) в освітньому середовищі. Здійснено аналіз цілей, класифікації та елементів ІС, а також особливостей їхнього впровадження в українському цифровому освітньому ландшафті. Підкреслено ключові бар'єри, такі як інтеграція різноманітних систем, забезпечення безпеки даних та адаптація до індивідуальних освітніх потреб. Визначено недосліджені області застосування нечіткої логіки та інфографіки для автоматизації управління, що стало основою для формування дослідницьких завдань. Проаналізовано сучасні платформи, включаючи Moodle, та їхні можливості для автоматизації навчального процесу і адміністрування в умовах студентського містечка.

У **другому розділі** зосереджено увагу на дослідженні чинників, що визначають впровадження та практичне використання інформаційної технології для автоматизованої системи обслуговування студентського містечка. Ідентифіковано основні компоненти, що впливають на функціонування ІС, зокрема, доступність ресурсів, ефективність взаємодії та ступінь автоматизації процесів. Розроблено семантичну мережу для формалізації взаємозв'язків між цими факторами, що стала фундаментом для побудови багатошарової моделі їх впливу, використовуючи математичне моделювання ієрархій.

Модель оптимізовано, обравши ваги факторів на основі матриці попарного зіставлення та шкали відносної значущості. Згенеровано альтернативні сценарії функціонування ІС, використовуючи функції корисності

та методи лінійного зростання, що дало змогу виділити оптимальний варіант згідно з критерієм максимальної ефективності. Застосовано методи нечіткої логіки для врахування суб’єктивних складових, таких як рівень задоволеності користувачів, що підвищило адаптивність системи.

У **третьому розділі** було презентовано процес проєктування та втілення системи відповідно до визначених вимог. Спершу було розгорнуто обґрунтовано поділ інформаційної технології на складові: підсистема для взаємодії з користувачами, модуль новин, підсистема для обміну повідомленнями та модуль бронювання кімнат. Відповідно до цих модулів були створені діаграми варіантів використання для відображення доступного функціоналу згідно з типом аутентифікованого користувача, а також діаграми класів для візуалізації сутностей та взаємозв'язків між ними. Окрім того, було розроблено структурно-функціональну модель інформаційної технології автоматизації діяльності студентського містечка. Перед безпосереднім втіленням системи було обґрунтовано вибір мови Ruby та фреймворку Ruby on Rails, що підтримує шаблон MVC, який сприяє розподілу відповідальностей, спрощує супровід коду та забезпечує гнучкість системи. Далі було подано опис розробки підсистеми користувачів, включаючи: застосування бібліотеки devise, приклад міграцій бази даних, візуалізацію можливостей реєстрації, входу в систему, відновлення паролю, функцію пошуку студента за ключовими словами та функціонал зручного завантаження фотографії в особистий профіль. Варто зазначити, що для спрощення додавання даних про наявних студентів було розроблено механізм автоматичного створення записів на основі зчитування .csv файлу з попередньо визначеними атрибутами. Також, у цьому розділі було описано втілення підсистеми новин, призначеної переважно для створення та поширення інформації серед мешканців студмістечка користувачами з роллю менеджера. Крім того, було детально представлено реалізацію функціоналу для листування, включаючи можливість відправлення персоналізованих або загальних повідомлень на електронну пошту. Важливо згадати про підсистему бронювання кімнат, імплементацію якої було чітко

описано з прикладами програмного забезпечення для відповідних класів MVC шаблону, включно з прикладами ергономічного графічного представлення. Для оптимізації швидкодії системи було інтегровано фонові задачі на основі бібліотеки sidekiq, що підтримує пріоритетні черги та паралелізм. Щоб зробити інформаційну технологію зручною для іноземних мешканців, було реалізовано підтримку англійської мови. Слід згадати про впровадження онлайн чату для користувачів, реалізованого за допомогою протоколу WebSocket. Важливим етапом було налаштування: статичного аналізатора коду, CI/CD, прикладу автоматичного розгортання за допомогою Github Workflow та Heroku, інтеграції із SendGrid API для надсилання листів через протокол SMTP.

У **четвертому розділі** розглядаються критичні заходи кібербезпеки, ключові загрози та способи протидії їм з використанням Ruby та фреймворку Ruby on Rails. Насамперед, було розтлумачено небезпеку SQL-ін'єкцій, їхній механізм виникнення, негативні наслідки ігнорування та методи захисту. Далі було приділено увагу Cross-Site Request Forgery. На конкретних прикладах було показано, як ця загроза впливає на систему загалом, але з допомогою правильного використання методу protect_from_forgery у Ruby on Rails можна уникнути потенційної шкоди. Окремо було розглянуто наслідки використання вразливих бібліотек в інформаційній технології. Підтримка залежностей на належному рівні є критичною умовою безпеки, адже навіть додаток, розроблений з урахуванням найвищих стандартів безпеки, може бути вразливим через проблеми у бібліотеках, які використовуються. Також були описані методи протидії Cross-Site Scripting, що може виникнути внаслідок виконання шкідливого коду JavaScript в системі. Тому було підкреслено важливість використання ERB-темплейтів та санітизації вхідних даних. Додатково було згадано про використання Strong parameters у фреймворку Ruby on Rails, щоб унеможливити зміну даних, які користувач не мав на меті змінювати. Далі було підкреслено необхідність коректного налаштування Content Security Policy, щоб система мала попередньо визначені безпечні джерела для завантаження шрифтів, стилів та графічного контенту. Також було

розглянуто загрозу, пов'язану з контролем доступу, адже при відсутності належної перевірки на боці сервера існує ризик надання користувачеві надлишкового рівня доступу, що, в свою чергу, може спричинити серйозні наслідки. Слід зазначити, що існує безліч різних безпекових загроз, тому в кінці розділу було представлено перелік загальних принципів безпеки, що мінімізують ймовірність появи вразливих місць у системі.

У висновках подані загальні отримані результати та пропозиції.

На основі проведеної роботи було досягнуто наступних вагомих результатів:

вперше:

- створено класифікаційну модель факторів якості процесу проектування автоматизованої системи забезпечення діяльності студентського містечка, на підставі якої здійснено формалізоване відображення та опис зв’язків між факторами за допомогою семантичних мереж і логіки предикатів, що забезпечило виконання подальших досліджень з використанням теорії ієрархій та нечіткої логіки;
- синтезовано та оптимізовано багаторівневі моделі пріоритетності впливу виокремлених факторів на якість процесу проектування автоматизованої системи забезпечення діяльності студентського містечка на основі розрахунку і упорядкування їх вагових значень за методами ранжування та аналізу ієрархій, що уможливило проектування альтернативних і розрахунок оптимальних варіантів якісної реалізації розглянутих процесів;
- побудовано багаторівневу модель, яка відтворює алгоритм формування якості процесу проектування автоматизованої системи забезпечення діяльності студентського містечка, що забезпечило отримання прогнозованих інтегральних показників якості даних процесів на основі заданих терм-множин значень, які відповідають означеним лінгвістичним термам лінгвістичних змінних;

- розроблено структурно-функціональну модель інформаційної технології формування якості процесу проектування автоматизованої системи забезпечення діяльності студентського містечка;

удосконалено:

- процес визначення пріоритетних факторів впливу на вибір інформаційної технології формування якості процесу проектування автоматизованої системи забезпечення діяльності студентського містечка;

отримано подальший розвиток:

- метод ранжування факторів в аспекті зіставлення та коригування наслідків його використання з інформацією, здобутою за допомогою методу аналізу ієрархій, що забезпечило відповідність визначених вагових значень факторів мірі їхнього впливу на досліджувані процеси.

Практична цінність даної праці визначається можливістю застосування розробленої технології задля автоматизації менеджменту в закладах вищої освіти. Це сприятиме оптимізації наявних ресурсів, покращенню прозорості робочих процесів, а також кращій адаптації до вимог кінцевих споживачів послуг.

Результати дисертаційної роботи впроваджені в Українській академії друкарства, на кафедрі інформаційних мультимедійних технологій м. Львів, у лекційних та практичних курсах дисциплін, зокрема: «Організація веборієнтованих баз даних», «Інформаційна безпека електронних ресурсів», «Системи управління електронним документообігом», «Технологія електронних мультимедійних видань» та «Управління мультимедійним проєктом». Запропонована інформаційна технологіябула апробована та частково впровадження у Львівському торговельно-економічному університеті**,** м. Львів та відповідає науково-дослідній темі Львівського торговельно-економічного університету «Моделювання та застосування хаотичних динамічних систем для оптимізації алгоритмів штучного інтелекту, машинного навчання та веб-технологій (державний реєстраційний номер 0124U004447). Дані про впровадження підтверджено відповідними документами.

Ключові слова: інформаційна система, клієнт-серверна модель, нечітка логіка, модель, фактор, математичне моделювання, метод ранжування, візуалізація, метод попарного порівняння, кібербезпека, сканування вразливостей, СУБД, сервер, оптимізація, бази даних.

ABSTRACT

Ternovyy I. M. Information technology of designing an automated system for ensuring the activities of a student campus. – Manuscript.

Dissertation for the degree of Doctor of Philosophy in specialty 126 “Information Systems and Technologies”. – Lviv Polytechnic National University, – Lviv, 2025.

The dissertation solves the current scientific and applied problem of developing information technology of designing an automated system for providing the activities of a student campus through the isolation and study of factors influencing the specified processes and the use of means of predictive quality assessment based on fuzzy logic. The implementation of the research is the construction of a structural and functional model of the information technology of the considered process.

The dissertation consists of an introduction, four sections, conclusions, a list of sources used and appendices.

The **introduction** emphasizes the importance of the chosen topic, which concerns the automation of student campus management through the use of information systems (IS). The focus is on the increasing interest in digital solutions in the educational sphere, the difficulties of implementing IS in educational and management processes, as well as the need for adaptive systems capable of responding to the unique requests of each member of the educational space. The relationship of the study with the scientific work of the Department of Multimedia Technologies, the purpose of which is to improve information technologies for the automation of educational processes, is demonstrated. The goal of the study is formulated - the creation of information technology for automated management of a student campus, based on fuzzy logic and system analysis, and key tasks are identified. The scientific novelty is substantiated, which consists in combining a multi-level model of factors influencing the functioning of IS and the application of fuzzy logic to improve processes. The practical value of the work lies in the prospect of implementing the proposed technology in higher education institutions to increase the efficiency of campus management. The author's personal contribution, which

includes the development of models and methods, as well as joint publications with the scientific supervisor, is described. The results of the study were presented at a number of scientific and practical conferences. The structure and scope of the dissertation work are summarized.

The first section examines the current state of affairs and the difficulties that arise when using intelligent systems (IS) in the field of education. An overview of the goals of IS, their classification and components, as well as the specifics of their implementation in the Ukrainian digital educational space is provided. The focus is on the main obstacles, including the integration of various systems, information protection and customization for individual needs. Unexplored aspects of the use of fuzzy logic and infographics for management automation are identified, which became the starting point for setting research objectives. Modern platforms, in particular Moodle, and their capabilities for automating learning and administration on a student campus are considered.

The second section analyzes the factors that influence the implementation and use of information technology of an automated campus support system. Key elements that influence the operation of IS are highlighted, including the availability of resources, the quality of interaction and the level of automation. A semantic network was created for the formal representation of the relationships between factors, which formed the basis for the development of a multilevel model of their influence using the method of mathematical modeling of hierarchies. The model was optimized by determining the weights of factors based on the pairwise comparison matrix and the scale of relative importance. Alternative scenarios of the IS operation were created using utility functions and linear growth methods, which made it possible to determine the optimal option using the criterion of maximum efficiency. Fuzzy logic methods were used to take into account subjective aspects, for example, the level of user satisfaction, which increased the adaptability of the system.

The scientific novelty of the presented research lies in the creation of information technology for automating campus management. It combines fuzzy logic

and a multilevel model of influential factors, and also involves the formation of a methodology for choosing the best ways of IS functioning.

The practical value of this work is determined by the possibility of applying the developed technology to automate management in higher education institutions. This will contribute to the optimization of available resources, improved transparency of work processes, as well as better adaptation to the requirements of end users of services.

The third section demonstrated the process of designing and implementing the system according to the tasks set. First, the division of information technology into subsystems was described in detail and argued, namely: a subsystem for users, a subsystem for news, a subsystem for correspondence and a subsystem for booking rooms. According to the identified subsystems, use case diagrams were developed to present the available functionality according to the type of authenticated user and class diagrams to clearly present the entities and associations between them. Additionally, a structural and functional model of information technology for automating the campus activity system was developed. Before starting the implementation of the system, the main programming language Ruby and the framework Ruby on Rails were chosen with justification, which supports the MVC pattern, which promotes the separation of responsibilities, simplifies code support and makes the system more flexible. The following is a description of the development of the subsystem for users, namely: use of the devise library, an example of database migrations, a graphical representation of the possibility of registration, login, password recovery, the ability to search for a student by keywords, functionality for convenient uploading of a photo to a personal profile. It is worth noting that to facilitate adding data about existing students, a mechanism was developed for automatically creating student records in the database based on reading a .csv file, which should contain predefined attributes. In addition, this section described the implementation of the subsystem for news, which should mainly be used by users with the role of manager to create and distribute information among the residents of the campus. A detailed description of the implementation of the

functionality for correspondence was also presented, which includes the ability to send personalized or general messages to an email address. It is important to mention the subsystem for booking rooms, because the implementation of this subsystem was clearly described, adding software examples for the corresponding classes of the MVC template, including examples of ergonomic graphical representation. To improve the system's performance, background tasks were integrated based on the sidekiq library, which supports priority queues and parallelism. In addition, to make information technology convenient for foreign residents of the campus, support for English and Ukrainian was added. It is worth mentioning the implementation of an online chat for information technology users, which was built using the WebSocket protocol. An important stage was the configuration of the following tools: static code analyzer, CI/CD, an example of configuring automatic deployment using Github Workflow and Heroku, integration with SendGrid API for sending emails using the SMTP protocol.

The **fourth section** describes important cybersecurity measures, popular threats, and methods for neutralizing them using the Ruby programming language and the Ruby on Rails framework. First, the danger of SQL injection was explained, the principle of its formation, the consequences of ignoring it, and prevention methods. Then, attention was paid to Cross-Site Request Forgery, the demonstrated examples show how this threat affects the system as a whole, but with the correct use of the protect_from_forgery method from the Ruby on Rails framework, it is possible to prevent potential damage. The consequences of using vulnerable libraries in information technology are separately considered. It is worth noting that maintaining dependencies at the proper level is a necessary security condition, because even an application developed with high security standards can be vulnerable due to problems in the libraries used. Additionally, methods for combating Cross-Site Scripting are described, which can occur in the event of execution of malicious JavaScript code in the system. That is why the importance of using ERB templates and sanitizing input data was emphasized. In addition, it was necessary to mention the use of Strong parameters in the Ruby on Rails framework to prevent changes to data that the user

did not plan to modify. Then the importance of correctly configuring the Content Security Policy was emphasized so that the system had predefined safe sources for downloading fonts, styles, and graphic content in general. The threat associated with access control was also mentioned, because with improper verification on the server side, there is a possibility of providing excessive level of access to the user, which in turn can provoke serious consequences. It is worth noting that there are a large number of different types of security threats, which is why a list of general security principles was given at the end of the section that minimize the possibility of vulnerabilities in the system.

The conclusions present the general results and proposals.

Based on the work carried out, the following significant results were achieved:

for the first time

- a classification model of quality factors of the process of designing an automated system for ensuring the activities of a student campus was created, on the basis of which a formalized mapping and description of the relationships between factors was carried out using semantic networks and predicate logic, which ensured the implementation of further research using the theory of hierarchies and fuzzy logic;

- multilevel models of the priority of the influence of the identified factors on the quality of the process of designing an automated system for ensuring the activities of a student campus were synthesized and optimized based on the calculation and ordering of their weight values using the methods of ranking and analysis of hierarchies, which made it possible to design alternative and calculate optimal options for the high-quality implementation of the processes under consideration;

- a multi-level model was built that reproduces the algorithm for forming the quality of the design process of an automated system for ensuring the activities of a student campus, which ensured the receipt of predicted integral indicators of the quality of these processes based on the specified term sets of values that correspond to the specified linguistic terms of linguistic variables;

- a structural and functional model of information technology for forming the quality of the design process of an automated system for ensuring the activities of a student campus was developed;

improved:

- the process of determining priority factors influencing the choice of information technology for shaping the quality of the process of designing an automated system for supporting the activities of the student campus;

further development was obtained

- a method for ranking factors in terms of comparing and adjusting the consequences of its use with information obtained using the method of hierarchy analysis, which ensured the correspondence of the determined weight values of factors to the degree of their influence on the processes under study.

The results of the dissertation work have been implemented at the Ukrainian Academy of Printing, at the Department of Information Multimedia Technologies in Lviv, in lecture and practical courses of disciplines, in particular: "Organization of web-oriented databases", "Information security of electronic resources", "Electronic document management systems", "Technology of electronic multimedia publications" and "Multimedia project management". The proposed information technology was tested and partially implemented at the Lviv University of Trade and Economics, Lviv and corresponds to the research topic of the Lviv University of Trade and Economics "Modeling and application of chaotic dynamic systems for optimization of artificial intelligence algorithms, machine learning and web technologies (state registration number 0124U004447). Implementation data is confirmed by relevant documents.

Keywords: information system, client-server model, fuzzy logic, model, factor, mathematical modeling, ranking method, visualization, method of pairwise comparisons, cybersecurity, vulnerability scanning, DBMS, server, optimization, databases.

ПЕРЕЛІК ПУБЛІКАЦІЙ ЗДОБУВАЧА

Публікації в іноземних виданнях

1. Ternovyy I. Khamula O. FEATURES OF CREATING NEWS IN THE CAMPUS INFORMATION SYSTEM. / *Ricerche scientifiche e metodi della loro realizzazione: esperienza mondiale e realtà domestiche: Raccolta di articoli scientifici «ΛΌГОΣ» con gli atti della VI Conferenza scientifica e pratica internazionale, Bologna, 15 Novembre, 2024. Bologna-Vinnytsia: Associazione Italiana di Storia Urbana & UKRLOGOS Group LLC*, 2024. P. 179-181. URL: https://doi:1036074/logos-15.11.2024.040.

Публікації у виданнях, що входять до міжнародних наукометричних баз даних (Index Copernicus) та є науковими фаховими виданнями України:

2. Терновий І. М., Хамула О. Г. Теоретичні засади проєктування інформаційної системи студмістечка. / *Комп'ютерні технології друкарства*. № 1 (51). 2024. С. 78-89. URL: https://doi:10.32403/2411-9210-2024-1-51-78-89.

3. Терновий І. М., Хамула О. Г. Аналіз та визначення пріоритетності факторів впливу на процес проектування інформаційної системи для студмістечка. /*Поліграфія і видавнича справа*. № 2 (88). 2024. С. 83-94. URL: https://doi:10.32403/0554-4866-2024-2-88-83-94.

4. Терновий І. М., Хамула О. Г. Оптимізація ієрархічної моделі факторів впливу на проектування інформаційної системи для студмістечка /*Наукові записки УАД*. 2024. № 2 (69). 2024. с. 130-139. URL: https://doi:10.32403/1998-6912-2024-2-69-130-139.

5. Ternovyy I. Khamula O.DETERMINATION AND ANALYSIS OF PARAMETERS THAT INFLUENCE THE DESIGN OF THE STUDENT CAMPUS MANAGEMENT INFORMATION SYSTEM /*Комп'ютерні технології друкарства*. 2024. Vol. 2. no. 52. P. 178-193. URL: https://doi:10.32403/2411-9210-2024-2-52-178-193.

Публікації, які засвідчують апробацію матеріалів дисертації:

6. Терновий І. М., Хамула О. Г. / Розробка веб сервісу для студентського містечка. / *Тези доповідей наук.-техн конференції професорсько-викладацького складу, наукових працівників і аспірантів* (07.02.-11.02.2022р.). Львів: УАД. 2022. С.147.

7. Терновий І. М., Хамула О. Г. / Моніторинг систем веб сервісу за допомогою ELASTIC OBSERVABILITY / *Тези доповідей наук.-техн. конференції професорсько-викладацького складу, наукових працівників і аспірантів* (06.02.-10.02.2023р.). Львів: УАД. 2023. С.69.

8. Терновий І. М., Хамула О. Г. / Методи комунікації між сервісами в мікросервісній архітектурі / *Поліграфічні, мультимедійні та web-технології: тези доп. VIII Міжнародна. наук.-техн. конф*. (16-20 травня 2023, м. Харків) Харків: ТОВ «Друкарня Мадрид», 2023. Т1. С. 94-95.

9. Терновий І. М., Хамула О. Г. / Керування конфіденційними змінними за допомогою Hashicorp Vault / *Тези доповідей наук.-техн конференції професорсько-викладацького складу, наукових працівників і аспірантів* (05.02.-09.02.2024 р.). Львів: УАД. 2024. С. 211.

10. Ternovyy I. Khamula O.Nevidomska B.General Security Principles in Web Service Design. *The VII International scientific and practical conference «Present and future: priority areas of research in scientific and educational activities»,* February 17-19, 2025, Prague, Czech Republic. P. 201-206. URL:https://eu-conf.com/wp-content/uploads/2024/12/PRESENT-AND-FUTURE-PRIORITY-AREAS-OF-RESEARCH-IN-SCIENTIFIC-AND-EDUCATIONAL-ACTIVITIES.pdf.

ЗМІСТ

Вступ ……...… 20
Розділ 1. Сучасний стан та проблематика використання інформаційних технологій в освітньому процесі ……. 30
1.1. Призначення та використання інформаційних технологій …… 30
1.2. Класифікація інформаційних технологій ……. 36
1.3. Інформаційні технології для цифрового освітнього середовища в Україні ……. 50
1.4. Компоненти систем інформаційних технологій…… 59
1.5. Формування завдань дослідження …….. 67
Висновки до розділу 1 …… 71
Розділ 2. Дослідження факторів впливу на процес реалізації та використання інформаційної технології автоматизованої системи забезпечення діяльності студентського містечка …… 73
2.1. Виокремлення факторів, які впливають на функціонування інформаційної технології автоматизованої системи забезпечення діяльності студентського містечка …… 73
2.2. Синтез моделі факторів впливу на функціонування інформаційної технології ……. 93
2.3. Оптимізація моделі факторів впливу на функціонування інформаційної технології ……..…….. 102
2.4. Формування альтернативних варіантів процесу функціонування інформаційної технології автоматизованої системи забезпечення діяльності студентського містечка …….. 104
2.5. Проектування інформаційної технології автоматизованої системи забезпечення діяльності студентського містечка з використанням засобів нечіткої логіки …… 111
Висновки до розділу 2 …… 119

Розділ 3. Реалізація інформаційної технології автоматизованої системи забезпечення діяльності студентського містечка ……... 122
3.1. Проектування та опис етапів реалізації інформаційної технології …… 122
3.2. Імплементація автоматизованої системи інформаційної технології …. 129
3.2.1. Реалізація підсистеми для користувачів в інформаційній технології . 132
3.2.2. Реалізація підсистеми для новин в інформаційній технології ……. 138
3.2.3. Реалізація підсистеми для листування в інформаційній технології … 142
3.2.4. Реалізація підсистеми бронювання кімнат в інформаційній технології …….…… 144
3.2.5. Інтеграція фонових задач ……. 151
3.2.6. Локалізація сайту інформаційної технології забезпечення діяльності студентського містечка …….. 153
3.2.7. Фільтрація ……. 155
3.2.8. Розробка чату …… 156
3.2.9. Налаштування статичного аналізатора коду ……. 160
3.3. Розгортання автоматизованої системи інформаційної технології…….…… 161
3.3.1. Налаштування CI/CD ……...… 161
3.3.2. Розгортання засобами Heroku ……. 163
3.3.3. Налаштування автоматичного розгортання ……...……... 167
3.3.4. Sendgrid ……. 168
Висновок до розділу 3 ……...…… 171
Розділ 4. Формування безпеки автоматизованої системи підтримки функціонування студентського містечка: типи загроз та методи їх нейтралізації …… 173
4.1. SQL ін’єкції …….. 173
4.2. Cross Site Request Forgery ……...…… 174
4.3. Методи сканування безпеки системи на вразливості …….…. 177
4.4. Захист від XSS ……. 180

4.5. Mass Assignment (Strong Parameters) ……. 182
4.6. Content Security Policy …… 185
4.7. Broken Access Control ……. 189
4.8. Загальні безпекові принципи функціонування автоматизованої системи ……. 193
Висновок до розділу 4 …… 197
Висновки …….. 200
Список використаних джерел …… 202
Додатки …… 223
Додаток А. Перелік публікацій здобувача ……... 224
Додаток Б. Відомості про апробацію результатів дисертації …… 226
Додаток В. Довідка про використання в навчальному процесі результатів дисертаційної роботи …… 227
Додаток Д. Довідка про впровадження результатів дисертаційного дослідження …… 229

ВСТУП

Актуальність теми. Актуальність запропонованого дисертаційного дослідження значною мірою обумовлена викликами, що постають перед студентськими містечками у процесі організації їхньої повсякденної роботи. На сьогоднішній день, майже всі вищі навчальні заклади (ВНЗ) як в Україні, так і за кордоном, здійснюють облік різноманітних «об'єктів», а також процесів, які з ними пов’язані (ведення електронних баз даних, контроль за виконанням навчальних планів, поселення мешканців студентського містечка, їх виселення після закінчення терміну дії угоди про проживання або за порушення правил, визначених у договорі), використовуючи інформаційні системи, що потребують належного управління. Натомість, найчастіше застосовуються паперові реєстри, журнали та інші матеріальні носії інформації. Найбільш поширеним елементом комп'ютеризації в цій галузі є переведення паперових документів в електронний вигляд з використанням пакету програм Microsoft Excel. Але такий локальний підхід не дає змоги встановити зв’язок та співвіднести інформацію з іншими сферами університетського управління.

Також, майже всі освітні установи тепер володіють власними інтернет-сторінками, де можна знайти ключові новини, розклад занять, дані про академічні успіхи, інформацію про викладацький склад та інший потрібний матеріал. Незважаючи на активне впровадження комп'ютерних технологій у сфері освіти, багато аспектів ще потребують вдосконалення. Щоб впевнитися в цьому, варто лише подивитися на роботу вищих навчальних закладів.

Окрему увагу у цьому контексті, слід зосередити на взаємодії студента з університетським простором. Згідно з нашими спостереженнями та дослідженнями опублікованих матеріалів, цей аспект найменше охоплений автоматизацією та потребує глибшого аналізу. При вступі до вишу, а згодом і поселенні в гуртожиток, майбутні мешканці зіштовхуються з операціями, що підлягають оптимізації. До таких відноситься, наприклад, оплата за

проживання, отримання важливих сповіщень та дії у випадку втрати або відсутності перепустки при спробі потрапити до гуртожитку. Для сплати за проживання студент повинен спочатку дізнатись номер рахунку, потім відвідати найближче відділення банку, внести необхідну суму, отримати платіжний документ та передати його коменданту. Що стосується інформування студентів, на даний час комендант роздає паперові оголошення, які часто важко помітити, особливо, коли студенти поспішають. Основною перешкодою при вході до гуртожитку стає відсутність перепустки. Відповідно, навіть за наявності сплати за проживання, потрапити до помешкання буде неможливо. Існують також й інші проблеми, але вищезазначених прикладів має бути достатньо.

Після аналізу наявних публікацій, що досліджують цю проблему та способи її розв'язання, стає очевидним – їх кількість обмежена.

Процедура поселення студентів до гуртожитків – важлива складова інформаційної системи менеджменту університету. Значну кількість публікацій присвячено саме процесам поселення, їх автоматизації та комп’ютерній підтримці для оптимізації управлінських рішень. Щоб здобути цілісне уявлення про бізнес-процеси та організаційну структуру майбутньої системи, у цьому дослідженні аналізуються відповідні університетські положення. Наприклад, «Порядок поселення студентів у студентські гуртожитки» НТУУ «КПІ» [1], «Положення про студентське містечко Київського національного економічного університету імені Вадима Гетьмана» [2], «Положення про порядок поселення та проживання в гуртожитках студмістечка НПУ імені М. П. Драгоманова» [3], «Положення про студентський гуртожиток Центральноукраїнського національного технічного університету» [4], а також деякі інші нормативні документи. Хоча нормативно-правові акти Кабінету Міністрів та Міністерства освіти та науки України [5] здебільшого окреслюють загальні правила функціонування інтернатних закладів та норми проживання, вони також можуть відігравати ключову роль при розподілі функцій та визначенні правил у проектній системі. Наприклад, відповідно до Положення про студентські

гуртожитки: «Житлова площа в студентських гуртожитках не надається особам, які забезпечені житлом у тому ж населеному пункті, в якому зареєстровано гуртожиток» [2]. Положення конкретного університету детально розписує процес поселення, виселення та проживання, визначаючи, зокрема, що для отримання ордера необхідна довідка про сплату внесків [1].

Також було проаналізовано практичні публікації, зокрема проект створення веб-інформаційної системи «Гуртожиток» у Київському національному економічному університеті імені Вадима Гетьмана [6]. Метою створення цієї системи було не лише автоматизувати процеси, а й забезпечити відкритий доступ через мережу Інтернет, що унеможливлює несанкціоноване заселення та сприяє дистанційному контролю за умовами проживання в університетських гуртожитках. Деякі з функцій пов’язані з основними системами управління студентськими містечками, наприклад, декан гуртожитку перевіряє комплект документів для постійного мешканця та змінює його статус з «претендента на постійне проживання/вакантне місце» на «постійного мешканця/вакантне місце». Проте, не було враховано участь у процесі поселення представників деяких функціональних підрозділів, таких як деканат і директор студмістечка, які видають направлення та розподіляють місця між факультетами/відділеннями. Недоліком цієї системи є відкритий доступ до рішень. Оскільки цей ресурс містить особисту інформацію мешканців гуртожитку, доступ до нього повинен бути суворо обмежений відповідно до категорій користувачів. Аналіз свідчить, що більшість розробників систем використовують загальновідомий методологічний підхід для розв'язання цього завдання в суто бухгалтерському ключі. Іншими словами, основна увага приділяється автоматизації ведення відповідних баз даних, швидкому та своєчасному коригуванню, а також формуванню необхідних звітів для бухгалтерії та керівництва університету. Водночас, важливим аспектом студентського гуртожитку є врахування низки якісних факторів, які можуть впливати на психологічну атмосферу в кімнаті гуртожитку та, як наслідок, опосередковано впливати на успішність студентів.

Під час розробки автоматизованої системи розподілу місць у студентських гуртожитках планується застосувати компоненти штучного інтелекту, здатні брати до уваги низку якісних показників та характеристик під час прийняття рішень щодо поселення. За фундамент взято систему нечіткого логічного виведення. Сукупність правил, їхня відповідність, обґрунтованість та масштаб цільової предметної області визначають ступінь відповідності моделі реальному процесу обліку мешканців студентського містечка. Дослідження дає підстави вважати, що модель процесу управління студентським містечком є уніфікованою для різних навчальних закладів, оскільки вирішує аналогічні задачі для всіх закладів.

Вагомі здобутки у вивченні питання застосування інформаційних технологій у освітньому просторі ґрунтуються на методологічних напрацюваннях С. Архангельського, Ю. Бабанського, С. Гончаренка, М. Махмутова, Є. Полота, В. Сагарди, Л. Виготського, П. Гальперіна, Г. Костюка, О. Матюшкіна, Н. Тализіної та інших.

Неабиякий внесок у теоретичне підґрунтя розробки та застосування інформаційних технологій у виробництві та освітній сфері здійснили дослідники Б. Дурняк, В. Сеньківський, Л. Сікора, І. Піх, Б. Ковальський, О. Тимченко, М. Луцків, Т. Нерода, О. Хамула, В. Сабат, чиї наукові роботи присвячені розробці моделей та інформаційних технологій формування та прогнозування якості технологічних процесів.

Узагальнивши викладені зверху результати та труднощі, стає очевидною логічна потреба у розробці та впровадженні автоматизованої системи. Вона об'єднає функції забезпечення студентського містечка в цілісний механізм, що позитивно вплине на ефективність та комфорт використання, як для студентів, так і для керівництва. Автоматизація цих процесів не тільки покращить управління студентським містечком, але й забезпечить захист інформації, розширить доступ до сервісів, а також допоможе раціонально використовувати ресурси. Зважаючи на ці виклики, спостерігається значний інтерес до створення

інноваційної інформаційної технології автоматизованої системи для підтримки діяльності студентського містечка.

Таким чином, розроблення моделей інформаційної технології формування якості реалізації автоматизованої системи забезпечення діяльності студентського містечка основі методів системного аналізу і теорії нечітких множин є актуальним науковим завданням.

Мета і завдання дослідження. Метою дисертаційної роботи є проектування та імплементація інформаційної технології автоматизованої системи забезпечення діяльності студентського містечка на підставі виокремлення та аналізу факторів впливу на досліджуваний процес та прогностичного оцінювання його якості засобами нечіткої логіки.. Для того, щоб досягти поставлену мету дослідження, було визначено на виконання наступний список **завдань**:

- проаналізувати вже наявні інформаційні технології, що використовуються в освітньому просторі;
- ідентифікувати та систематизувати чинники, що забезпечують якість процесу проєктування автоматизованої системи обслуговування студентського містечка, застосувавши метод експертного опитування та створити семантичні мережі взаємозв'язків між ними;
- здійснити аналіз потенційного функціоналу запропонованої інформаційної технології відповідно до актуальних потреб, що існують в рамках студентського містечка, зокрема визначити ключові з них;
- спроектувати і обчислити альтернативні та виокремити оптимальні варіанти проектування автоматизованої системи обслуговування студентського містечка на основі лінійного згортання критеріїв та нечіткого відношення переваг;
- обчислити нормовані значення функцій приналежності лінгвістичних процесів на базі обробки складених матриць попарних порівнянь та використання введених лінгвістичних термінів та нечітких логічних рівнянь;

- спроектувати та описати відповідні підсистеми веб-сервісу, а задля ергономічного представлення спроєктованих підсистем застосувати діаграми прецедентів та діаграми класів;
- розробити структурно-функціональну модель інформаційної технології формування якості процесу проектування автоматизованої системи забезпечення діяльності студентського містечка;
- розробити онлайн чат для можливості комунікації між собою користувачами веб-сервісу;
- покроково описати автоматизований процес розгортання веб-сервісу, зокрема налаштувати CI/CD відповідно.

Об’єктом дослідження є процеси реалізації інформаційної технології проектування автоматизованої системи забезпечення діяльності студентського містечка.

Предметом дослідження є фактори, моделі, методи та засоби формування критеріїв якості процесів реалізації автоматизованої системи забезпечення діяльності студентського містечка.

Методи дослідження. У дисертації застосовано: експертний метод опитування – для виявлення та підтвердження процесів, процедур та чинників реалізації автоматизованої системи забезпечення діяльності студентського містечка; теорія графів та семантичних мереж, елементи логіки предикатів – для забезпечення компактності та візуальної наочності подання взаємозв’язків між факторами впливу, впорядкування відношення між ними та можливості застосування лінгвістичних методів опису знань та використання математичного апарату для розрахунку вагових значень чинників і визначення їх рангів; метод ранжування факторів – для створення багаторівневих моделей вагових значень факторів якості досліджуваних процесів; метод аналізу ієрархій – для побудови моделей пріоритетного впливу факторів на якість; метод попарних порівнянь – для виконання оптимізації моделей чинників забезпечення якості; методи багатокритеріальної оптимізації (теорію ухвалення рішень) – функцію корисності та нечітке відношення переваги – для пошуку

оптимальних варіантів забезпечення якості та оптимальних альтернатив досліджуваних процесів; методи та засоби нечіткої логіки – для оцінювання формування якості реалізації автоматизованої системи забезпечення діяльності студентського містечка.

Наукова новизна одержаних результатів. У дисертації розв'язано науково-прикладне завдання проектування та імплементації інформаційної технології автоматизованої системи забезпечення діяльності студентського містечка. На основі проведеної роботи було досягнуто наступних вагомих результатів:

вперше:

- створено класифікаційну модель факторів якості процесу проектування автоматизованої системи забезпечення діяльності студентського містечка, на підставі якої здійснено формалізоване відображення та опис зв'язків між факторами за допомогою семантичних мереж і логіки предикатів, що забезпечило виконання подальших досліджень з використанням теорії ієрархій та нечіткої логіки;
- синтезовано та оптимізовано багаторівневі моделі пріоритетності впливу виокремлених факторів на якість процесу проектування автоматизованої системи забезпечення діяльності студентського містечка на основі розрахунку і упорядкування їх вагових значень за методами ранжування та аналізу ієрархій, що уможливило проектування альтернативних і розрахунок оптимальних варіантів якісної реалізації розглянутих процесів;
- побудовано багаторівневу модель, яка відтворює алгоритм формування якості процесу проектування автоматизованої системи забезпечення діяльності студентського містечка, що забезпечило отримання прогнозованих інтегральних показників якості даних процесів на основі заданих терм-множин значень, які відповідають означеним лінгвістичним термам лінгвістичних змінних;

- розроблено структурно-функціональну модель інформаційної технології формування якості процесу проектування автоматизованої системи забезпечення діяльності студентського містечка;

удосконалено:

- процес визначення пріоритетних факторів впливу на вибір інформаційної технології формування якості процесу проектування автоматизованої системи забезпечення діяльності студентського містечка;

отримано подальший розвиток

- метод ранжування факторів в аспекті зіставлення та коригування наслідків його використання з інформацією, здобутою за допомогою методу аналізу ієрархій, що забезпечило відповідність визначених вагових значень факторів мірі їхнього впливу на досліджувані процеси.

Практичне значення одержаних результатів. У результаті виконання дисертаційної роботи було досягнуто значних результатів, що мають наступне практичне застосування:

- імплементовано програмний продукт інформаційної технології автоматизованої системи забезпечення діяльності студентського містечка, що надає можливість керівництву студентських містечок впроваджувати дану систему для оптимізації внутрішніх процесів, зокрема тих, які пов’язані з документообігом існуючих студентів;
- розроблено підсистеми для новин та листування, які істотно спрощують поширення різного роду новин, адже відпадає необхідність в паперових оголошеннях, які студенти досить часто можуть не побачити. Це в свою чергу економить зусилля та ресурси працівників студентського містечка для поширення паперових оголошень;
- спроектовано та розроблено функціонал бронювання вільних кімнат студентами, якщо в них є бажання змінити поточну кімнату проживання. В такому випадку менеджер студентського містечка буде чітко бачити заявки студентів, які бажають проживати в конкретній кімнаті, і вже після вивчення отриманих заявок є можливість зробити зважене рішення

на рахунок даних заяв. Такий підхід спрощує комунікацію студентів з адміністрацією студентського містечка, а також відкидає необхідність в фізичній присутності;

- програмне забезпечення представлено українською та англійською мовами, що в свою чергу дозволяє іноземним студентам користуватись інформаційною технологією на рівні з українськими студентами.

Результати дисертаційної роботи використано у:

– навчальному процесі Української академії друкарства під час викладання дисциплін «Організація веборієнтованих баз даних», «Інформаційна безпека електронних ресурсів», «Системи управління електронним документообігом», «Технологія електронних мультимедійних видань» та «Управляння мультимедійним проєктом»;

– апробовано та впровадження у Львівському торговельно-економічному університеті**,** м. Львів, та відповідає науково-дослідній темі Львівського торговельно-економічного університету «Моделювання та застосування хаотичних динамічних систем для оптимізації алгоритмів штучного інтелекту, машинного навчання та веб-технологій (державний реєстраційний номер 0124U004447).

Дані про впровадження підтверджено відповідними документами.

Особистий внесок здобувача. Основні дослідження (постановка завдань, виокремлення, класифікація та ранжування факторів впливу на якість процесу проектування автоматизованої системи забезпечення діяльності студентського містечка, побудова семантичних мереж формалізованих зв’язків між факторами, проектування і розрахунок альтернативних та оптимальних варіантів реалізації процесів, синтезування моделей, проектування нечітких логічних рівнянь, розрахунок функцій належності) та подання їх результатів у вигляді публікацій і доповідей на конференціях здійснено здобувачем особисто. У спільних публікаціях автору належить провідна роль: [76, 150, 151] – дослідження платформ провадження інформаційних технологій; [60, 74] – виокремлення факторів забезпечення якості процесу проектування

автоматизованої системи забезпечення діяльності студентського містечка; [55] – використання теорії графів для визначення пріоритетності факторів між собою; [67, 75] – реалізація на практиці основ використання структурно-функціональної моделі інформаційної технології проектування автоматизованої системи забезпечення діяльності студентського містечка; [161, 197] – визначення вразливостей системи та її безпека.

Апробація результатів дисертації. Головні результати дисертації доповідались та обговорювались на: VIII Міжнародних науково-практичних конференціях «Поліграфічні, мультимедійні та WEB-технології» 16-20 травня 2023 р. (Харків, 2023); звітних науково-технічних конференціях професорсько-викладацького складу, наукових працівників і аспірантів Української академії друкарства (Львів, 2022-2024); VI Міжнародній науково-практичних конференції«Conferenza scientifica e pratica internazionale» (Bologna, 15 Novembre, 2024); VII International scientific and practical conference «Present and future: priority areas of research in scientific and educational activities» (February 17-19, 2025, Prague, Czech Republic).

Публікації. Результати проведених досліджень викладено у 10 наукових працях, серед яких: одна публікація в іноземному виданні; 4 з яких – у фахових виданнях України, що входять до міжнародної бази даних Index Copernicus; 5 публікацій у матеріалах наукових конференцій.

Структура та обсяг дисертації. Дисертація складається з анотацій, вступу, чотирьох розділів, висновків, списку використаних джерел із 224 найменувань та 4 додатків. Загальний обсяг роботи – 229 сторінок, 180 з яких займає основний текст. У дисертації 45 рисунків та 11 таблиць.

РОЗДІЛ 1.
СУЧАСНИЙ СТАН ТА ПРОБЛЕМАТИКА ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ В ОСВІТНЬОМУ ПРОЦЕСІ

1.1. Призначення та використання інформаційних технологій

Сьогодні в царині комп’ютерних технологій щодня з'являються нові технології. Доступ до Інтернету та його розвиток надали людству змогу використовувати новітні технології в більшості галузей промисловості. На даний момент дуже важливо є забезпечити впровадження новітніх комп'ютерних інструментів у різні традиційні галузі, зокрема й у систему освіти.

Розвиток та формування інформаційних систем можливо охарактеризувати в декілька етапів становлення. Історичний розвиток інформаційних систем розпочався в середині XX століття з появою перших електронних обчислювальних машин. У 1940-х роках такі машини, як ENIAC (1945), використовувалися для наукових і військових обчислень, закладаючи основу для автоматизації обробки даних. Ці ранні системи були громіздкими, орієнтованими на виконання окремих завдань і не мали інтегрованого підходу до управління інформацією. У 1950-х роках з’явилися перші комерційні комп’ютери, такі як IBM 701 (1952), які почали застосовуватися в бізнесі для обліку та планування.

У 1970-х роках розвиток транзисторної технології та поява мейнфреймів (наприклад, IBM System/360) сприяли створенню централізованих інформаційних систем. Ці системи використовувалися для управління великими обсягами даних в банках, урядових установах і великих університетах. У цей період з’явилися перші системи управління базами даних (СУБД), такі як IMS від IBM (1968) і реляційна модель Кодда (1970), які дозволили структурувати дані та забезпечити швидкий доступ до них.

В освіті ІС почали застосовуватися для автоматизації адміністративних процесів, таких як облік студентів, планування розкладів та обробка оцінок.

Наприклад, у США системи на кшталт Student Information Systems (SIS) стали основою для управління даними в університетах. У цей період також зародилися перші експерименти з комп’ютерним навчанням (Computer-Based Training, CBT), які передбачали використання ІС для подання навчального матеріалу, як у концепції адаптивного навчання.

Запровадження персональних комп'ютерів (ПК) у 1980-х, зокрема IBM PC (1981) та Apple Macintosh (1984), відчинило двері до обчислювальних технологій для широкого кола користувачів. Розвиток локальних мереж (LAN) і протоколів TCP/IP заклав основу для мережевих інформаційних систем. В цей період виникли клієнт-серверні архітектури, що дозволили ділити обчислювальні ресурси між користувачами.

В освітньому процесі ПК почали використовувати для створення електронних навчальних матеріалів та автоматизації бібліотечних систем. У 1990-х роках поява Всесвітньої павутини (World Wide Web, 1991) та браузерів, таких як Netscape, відкрила нові обрії для дистанційного навчання. Як зазначають Яремчук Н. А., Сікоза О. М. та Кельберг С. М. [7], системи дистанційного навчання з контрольними блоками та індивідуальними траєкторіями почали формуватись саме тоді, застосовуючи мережеві технології для доступу до навчальних ресурсів через Інтернет.

У 2000-х роках стрімкий розвиток Інтернету та широкосмугового зв'язку сприяв появі веб-орієнтованих інформаційних систем. Платформи дистанційного навчання, на зразок Moodle (2002), набули популярності у вищих навчальних закладах завдяки своїй доступності та гнучкості. У дослідженні Колос К. Р. [8] підкреслено роль Moodle у розвитку компетентностей через дистанційну освіту, що стало підґрунтям для автоматизації навчальних процесів у студентських містечках.

Цей період також характеризується появою хмарних технологій та програмного забезпечення як послуги (SaaS), що надало університетам можливість використовувати ІС без значних фінансових вкладень у власну інфраструктуру. Наприклад, системи управління навчанням (Learning

Management Systems, LMS) та студентські інформаційні системи (SIS) почали інтегруватися з веб-порталами, забезпечуючи студентам доступ до розкладів, оцінок і навчальних матеріалів.

Починаючи з 2010-х років, ІС перейшли до хмарних архітектур, які забезпечують масштабованість, доступність та економію ресурсів. Технології Інтернету речей (IoT), штучного інтелекту (ШІ) та великих даних (Big Data) відкрили нові горизонти для автоматизації. У студентських містечках IoT використовується для моніторингу інфраструктури (аудиторії, гуртожитки), а ШІ – для персоналізації навчання, зокрема в адаптивних системах.

Методи нечіткої логіки стали важливим інструментом для керування складними системами з невизначеністю, наприклад, для оцінки пріоритетності запитів чи рівня задоволеності студентів. Інфографіка, як підкреслено в дисертаційній роботі Конюхова А. Д. [9], почала застосовуватися для візуалізації даних, що полегшує прийняття рішень адміністрацією. Дослідження Рашевської Н. В. [10] демонструє, що мобільні технології та змішане навчання стали ключовими для сучасних ІС, дозволяючи студентам отримувати доступ до ресурсів через смартфони.

Якщо розглядати крізь призму етапів розвитку, картина виглядає так. Зародження інформаційних систем датується 1950-ми роками, відомими тоді як "автоматизовані системи управління" (надалі – АСУ), розроблені для автоматизації керування підприємствами з використанням великих електронно-обчислювальних машин (надалі – ЕОМ) та централізованої обробки даних. Їх діяльність обмежувалася розв'язанням певних функціональних управлінських задач, наприклад, задач бухгалтерського обліку. Для кожної такої задачі готувалися дані, створювалася математична модель і розроблялося програмне забезпечення. Окрім процедур безпосереднього розв'язання задачі, до програм вносилися процедури формування та ведення необхідного інформаційного фонду [11].

Другий етап еволюції інформаційних систем припадає на 1980-ті роки. Основою таких систем стала концепція централізовано керованої бази даних,

яка за допомогою спеціального програмного продукту обслуговує прикладні програми. Тобто забезпечувалося колективне використання даних. Відповідно до цього, системи другого покоління називали, зокрема, "Управлінські (адміністративні) ІС" (Management Information System).

Для інформаційних систем третього покоління притаманним є втілення ідеї єдиної централізованої бази моделей – обчислювальних блоків, які спільно використовуються багатьма прикладними програмами. Ці системи отримали найменування «Системи підтримки прийняття рішень» (Decision Support System, далі – СППР). СППР – це інтерактивна комп'ютерна система, розроблена для забезпечення різних видів діяльності в процесі ухвалення рішень стосовно слабо структурованих або неструктурованих питань.

Інформаційні системи четвертого покоління – «Система планування ресурсів з синхронізацією з користувачем» (Customer Synchronized Resource Planning). Сформовані комп'ютеризацією суспільства, переходом до інформаційного суспільства та поступом суспільства знань.

Однак, варто відзначити, що кожен з наступних етапів формування та впровадження інформаційних систем не обмежував використання та розвиток попередніх, а лише розширював можливості та сферу їх застосування.

Система (від дав.-гр. σύστημα – «сполучення») – порядок, зумовлений правильним, планомірним розташуванням та взаємним зв'язком частин будь-яких елементів [12].

Отже, з формальної лінгвістичної перспективи, інформаційна система – це взаємозалежний набір даних та/або відомостей; або, кажучи інакше, впорядковані дані.

Поняття інформаційної системи в українському правовому полі не визначено, натомість в Законі України «Про захист інформації в інформаційно-телекомунікаційних системах» зустрічаємо кілька похідних понять, класифікованих за типами систем, зокрема:

- інформаційна (автоматизована) система – організаційно-технічна система, в якій впроваджується технологія обробки інформації з використанням технічних та програмних інструментів;
- інформаційно-телекомунікаційна система – сукупність інформаційних та телекомунікаційних систем, що працюють як єдине ціле в процесі обробки інформації.

Тобто, інформаційна система розглядається як середовище, в якому впроваджено автоматизовану обробку інформації. Без певного рівня автоматизації інформаційна система, з нормативної точки зору вітчизняного законодавця, фактично не існує. При цьому, визначення автоматичних інформаційних систем також відсутнє.

Але ж автоматизовані інформаційні системи становлять лише один з різновидів інформаційних систем, якщо брати до уваги критерій автоматизації. Оскільки інформаційні системи збирають будь-яку наявну інформацію, то і її обробка може здійснюватися як людським ресурсом, так і з допомогою комп'ютерів чи інших технічних пристроїв. Таким чином, існує об'єктивна потреба розділяти типи інформаційних систем, що дозволить сформувати та застосувати правильніший підхід до їх використання, а за потреби — й регулювання.

Серед варіантів інтеграції комп'ютерних технологій в освітній процес — впровадження спеціальних інформаційних систем, що можуть автоматизувати багато важливих процесів у навчальних закладах. Створення такого типу систем вимагає враховувати ряд факторів, що відносно специфічні, порівнюючи з потребами впровадження комп'ютерних технологій в багатьох інших сферах та бізнес-проектах зокрема.

Дані, які опрацьовуються сучасним комп'ютерним устаткуванням та технологіями, мусять мати чітку структуру, у межах якої відбувається процес їхнього оброблення. Аналогічно, під час аналізу окремих бізнес-ініціатив, що передбачають застосування передових технологій, положення стосовно обробки інформації, її функції та структури в комп'ютерній техніці мають бути

пов'язані з певною концепцією або абстракцією. З огляду на цю необхідність організованості даних у всіх сучасних організаціях, бізнес-додатках, різноманітних сферах виробництва тощо, формується ще одне визначення - інформаційні системи.

Різні фахівці по-різному визначають термін «інформаційна система». Деякі відомі визначення включають:

- Інформаційна система технічно може бути визначена як набір взаємопов'язаних компонентів, які збирають, обробляють, зберігають і поширюють інформацію для підтримки ухвалення рішень і контролю в організації.

- Інформаційна система зазвичай являє собою комбінацію апаратних засобів, програмного забезпечення та комунікаційних мереж, які люди створюють і використовують для збору, створення та поширення корисних даних в організаційному середовищі.

Як видно з цих визначень, вони фокусуються на двох способах опису інформаційних систем: компонентах, з яких складається інформаційна система, і ролі, яку ці компоненти відіграють в організації. Розглянемо кожен спосіб опису терміна окремо [13, 14].

Інформаційна система (ІС) - це сукупність взаємопов'язаних компонентів: інструментів, методик та персоналу, які застосовуються для створення [9], обробки, збереження та забезпечення доступу до даних з метою підтримки певних процесів або видів діяльності. Відповідно до Державного стандарту ДСТУ 2392-94 [15], інформаційну систему визначають як систему комунікацій, що здійснює збір, пошук, обробку та розповсюдження інформації для користувачів. У широкому розумінні, інформаційні системи - це організаційно-технічні комплекси, які інтегрують комп'ютерну техніку та програмне забезпечення з організаційними регламентами та людськими ресурсами для ефективного опрацювання інформації [16].

Метою інформаційних систем є покращення ефективності функціонування організацій та окремих процесів через автоматизацію роботи з

інформацією. ІС покликані гарантувати швидке отримання перевірених даних для прийняття рішень, керування ресурсами та документування операцій. В освітній сфері призначення ІС – підтримувати навчальний процес, адміністративне управління закладом, відстеження успішності, доступ до освітніх матеріалів та інше, задля підвищення якості та зручності освітніх послуг. Сучасні інформаційні системи фактично стали фундаментом цифровізації освітнього простору, дозволяючи збирати, зберігати, аналізувати та передавати освітню інформацію між усіма учасниками процесу (студентами, викладачами, адміністрацією, батьками) в електронній формі.

Отже, мета інформаційних систем зумовлює їхню організацію та розподіл. Інформаційні системи бувають ручні або автоматизовані, розміщені локально або розподілені по мережі, маленькі (для конкретних завдань) або інтегровані, корпоративного рівня. В освітньому процесі це проявляється у різноманітті ІС - від елементарних баз даних для фіксації результатів навчання до складних освітніх платформ, автономних або хмарних. В наступних розділах ми більш детально проаналізуємо основні типи інформаційних систем та їх складові, а також особливості сучасних інформаційних систем, що формують цифрове освітнє середовище.

Інформаційні системи є наріжним каменем цифрової трансформації всього суспільства. Вони не тільки дають змогу організаціям працювати значно ефективніше, а й народжують нові способи управління, надання послуг та взаємодії між людьми. У найближчому майбутньому роль ІС буде лише збільшуватися, адже вони об'єднуватимуться з найновішими цифровими технологіями, як-от штучний інтелект, великі масиви даних та хмарні обчислення.

1.2. Класифікація інформаційних технологій

Як було зазначено вище, інформаційна система (ІС) – це сукупність організаційних і технічних засобів для зберігання та обробки інформації з метою задоволення інформаційних потреб користувачів. Різноманітність

інформаційних систем зумовлена широким спектром завдань, які вони вирішують, та рівнів управління, які вони підтримують. Інформаційні системи можуть істотно відрізнятися за типом керованих об'єктів, характером і масштабом розв'язуваних завдань і багатьма іншими характеристиками [18]:

1. За рівнем або масштабом діяльності - національні, регіональні (місцеві), галузеві, асоціації, підприємства або установи, технічні процеси.

Національна інформаційна система має на меті вирішення критично важливих економічних питань держави. Використовуючи потужні комп'ютерні системи та економіко-математичні підходи, вона створює довгострокові та оперативні плани розвитку країни, здійснює облік досягнутих результатів, координує діяльність різних сфер народного господарства, формує державний бюджет та контролює його реалізацію.

Територіальна (регіональна) ІС служить для управління адміністративно-територіальними утвореннями. Сюди зараховують обласні, міські та районні ІС. Ці системи опрацьовують інформацію, потрібну для реалізації функцій територіального управління, плюс формують звіти та оперативні дані для регіонів і керівних органів держави й економіки.

2. За рівнем автоматизації процесів управління - інформаційно-пошукові.

Галузеві інформаційні системи керування розроблені для управління підпорядкованими підприємствами та організаціями. Галузеві інформаційні системи керування використовуються в промисловості, сільському господарстві, будівництві та на транспорті. Галузеві ІС забезпечують інформаційну підтримку галузевих міністерств та підпорядкованих їм управлінських підрозділів.

Інформаційні системи управління підприємством (ІСУП), або інформаційні системи виробничого призначення (ІСВП) - це системи, що використовують сучасні засоби автоматизованої обробки даних, економічні, математичні та інші методи для регулярного розв'язання завдань управління виробничо-господарською діяльністю підприємств.

Інформаційні системи керування технологічними процесами (ІСУ ТП) відповідають за стан виробничих процесів. Головна відмінність цих систем від розглянутих раніше криється передусім у специфіці різноманітних машин, пристроїв та обладнання, що підлягають контролю. Ще одна відмінність полягає у способі передавання інформації. В ІСУ технологічних установок основною формою передавання інформації є сигнали, на відміну від інших ІСУ, де переважають документи.

3. За ступенем централізації оброблення інформації - централізовані ІС, децентралізовані ІС, інформаційні системи колективного користування.

В залежності від цілей використання та поставлених задач, що стоять перед ІС на стадіях збору та обробки даних, ІС класифікуються на наступні види: інформаційно-пошукові; інформаційно-довідкові; інформаційно-керуючі (менеджмент); інтелектуальні інформаційні системи та системи підтримки прийняття рішень.

Інформаційно-пошукові системи (ІПС) зосереджені на вирішенні задач пошуку інформації. В цих системах не відбувається смислового опрацювання інформації.

В інформаційно-довідкових системах (ІДС) результати пошуку використовуються для обчислення значення арифметичної функції.

4. За ступенем інтеграції функцій - багаторівневі ІС, інтегровані за рівнями управління (підприємство - асоціація, асоціація - галузь тощо) та багаторівневі ІС, інтегровані за рівнями планування і т. ін.

Інформаційно-керуючі системи, або ж системи керування, – це організаційно-технічні комплекси, розроблені для прийняття рішень, ґрунтуючись на автоматизації інформаційних потоків у галузі управління. Відповідно, ці системи створюються для автоматичного вирішення різноманітних управлінських питань.

Сучасне покоління інформаційних систем включає в себе системи підтримки ухвалення рішень (СПУР) та інформаційні системи, що використовують штучний інтелект (інтелектуальні ІС).

СПУР - це інтерактивні комп'ютерні системи. Зацікавленість у СПУР безперервно збільшується як у багатообіцяючому руслі розвитку комп'ютерних технологій та засобів покращення продуктивності праці в галузі економічного управління.

Штучний інтелект (ШІ) - це сконструйована людиною штучна система, що використовує комп'ютерні технології для відтворення здатності людини вирішувати комплексні та творчі завдання. Розвиток логічних та лінгвістичних моделей в теорії штучного інтелекту суттєво вплинув на процес створення інтелектуальних інформаційних систем. Ці моделі забезпечили можливість формалізації специфічних знань про об'єкти та процеси, які підлягають управлінню. До логіко-лінгвістичних моделей відносять семантичні мережі, фрейми та продукційні системи, які часом об'єднуються загальним терміном "програмне та апаратне забезпечення систем штучного інтелекту".

Існує три типи інтелектуальних ІС:

1. Інтелектуальні інформаційно-пошукові системи (системи запитань та відповідей) дозволяють кінцевим користувачам, тобто тим, хто не є програмістами, спілкуватися з базами даних та сховищами знань, використовуючи спеціалізовану мову, яка наближена до природної. Це відбувається в процесі інтерактивного діалогу.

2. Обчислювальні та логічні структури, що відкривають можливість для простих користувачів, котрі не мають спеціальності програміста або фахівця з прикладної математики, розв'язувати задачі, спілкуючись з комп'ютером через обчислювальні методи та необхідне програмне забезпечення.

3. Експертні системи, які дозволяють представити знання у спеціалізованому описовому форматі, але разом з тим ускладнюють використання математичного апарату, чи ж експертні системи, що дозволяють ефективно комп'ютеризувати галузі, де це раніше було нездійсненним.

Щодо інформаційних систем в організації, фахівці зазвичай визначають три головні види, які використовуються в компанії. Ці типи часто представляють у вигляді піраміди. В ній різні типи інформаційних систем

розташовуються з огляду на важливість структурованого ухвалення рішень користувачами в конкретному бізнесі. Цю схему представлено на рисунку 1.1.

Рисунок 1.1: Структура типів інформаційних систем в організації [19]

Розгляньмо кожен з рівнів організації інформаційних систем по черзі. Варто зауважити, що, як ми переконаємось згодом, певні функції інформаційних систем корелюють з типами систем на конкретних рівнях цієї схеми.

Рівень операційної підтримки.

Рівень операційної підтримки напряму стосується щоденних бізнес-процесів, здебільшого зосереджуючись на обробці транзакцій. Ключовою рисою цієї ієрархії є те, що користувачі цих систем приймають рішення, які мають чітку структуру. Це означає, що для ухвалення рішень користувачі повинні застосовувати наперед визначені правила та базу знань. Серед найпоширеніших типів систем, які розглядаються на цьому рівні, виділяють системи управління ланцюгами поставок (SCM), системи управління взаємовідносинами з клієнтами (CRM) та системи обробки транзакцій.

Перший різновид системи підтримки бізнесу, знаний як SCM, опікується керуванням потоками товарів, інформації, ресурсів та даних через увесь ланцюжок постачання. Він охоплює все: від постачальників сировини та матеріалів до компаній, що займаються обробкою даних, дистриб'юторів та роздрібних продавців. Цей тип системи керує обміном інформацією між різними учасниками процесу та контролює доставку послуг безпосередньо до кінцевого споживача. Користувачеві системи не потрібно витрачати зусилля на управління самою системою або ухвалювати рішення на основі деяких результатів обробки даних, отриманих на попередньому етапі.

Прикладом функціонування такої системи може бути втілення торгової платформи в супермаркеті. Коли касир оформлює чек на покупку, система автоматично списує відповідні позиції для певних товарів, наявних на складі, а через деякий проміжок часу просто інформує постачальника про необхідність замовити нові товари, разом з цим автоматично надсилаючи запит на формування нової поставки.

Другий різновид системи – управління взаєминами з клієнтурою (CRM). Цей різновид систем застосовується для діяльності з окремими клієнтами компанії у царині маркетингу, продажу, сервісу та створення нових товарів. Застосування таких систем полегшує процес взаємодії бізнесу з окремими клієнтами й таким чином дає можливість підтримувати з ними міцні та позитивні відносини.

Останній різновид систем щільно взаємодіє з операціями обробки інформації всередині конкретної компанії. Цей тип систем об'єднує різні підрозділи організації та керує процесом реалізації кінцевого продукту відповідно до операцій кожного відділу, таких як продажі та маркетинг, виробництво, фінанси та управління персоналом. У більших організаціях обробка транзакцій інтегрована в окремі системи планування ресурсів підприємства (ERP), що є ключовим фундаментом майже всіх компаній. Ці системи підтримують ланцюжок створення вартості продукту, тобто

послідовність дій та процесів, завдяки яким компанія збільшує вартість своєї продукції.

Принципи функціонування подібних систем перегукуються з підходами, що використовуються в системах SCM, однак управління виробництвом зосереджене всередині однієї компанії. Відповідні підрозділи автоматично оновлюють дані про продажі, як тільки новий продукт стає доступним для клієнтів компанії. З огляду на це, можна стверджувати, що ERP-системи можуть сприяти досягненню позитивних результатів для окремих компаній, забезпечуючи тісну взаємодію з іншими інформаційними системами, як це й було зазначено у вищеописаних ролях.

Рівень підтримки використання інформації

Наступним етапом розповсюдження інформаційних систем в організації є забезпечення роботи з інформацією. Цей етап присвячений використанню інформаційних систем для вирішення задач, що вимагають роботи з абстрактною інформацією та знаннями, а не задач безпосередньої обробки, розробки та надання готових даних згідно з відомими процесами. До таких задач відноситься напівструктуроване прийняття рішень. Це зумовлено тим, що, хоча дані в цьому процесі можуть ґрунтуватися на наборі рекомендацій або правил, остаточне рішення має приймати користувач системи. З цієї причини цей етап також часто називають етапом підтримки тактичного управління.

Серед категорій інформаційних систем, що відносяться до цього рівня, виділяють системи професійної підтримки, системи для спільної діяльності та системи управління знаннями.

Системи професійної підтримки знайшли своє застосування на підприємствах для виконання конкретних завдань, властивих певній професії. Наприклад, розробники програмного забезпечення в організаціях, що спеціалізуються на інформаційних технологіях, активно використовують інтегровані середовища розробки (IDE) для написання коду продукту.

Біохіміки вдаються до спеціалізованих програм для 3D-моделювання, аби візуалізувати молекулярні структури та протестувати дію новітніх ліків ще до клінічних випробувань.

Системи спільної роботи застосовуються для полегшення комунікації та командної взаємодії між працівниками організацій. Як вже згадувалося, існує кілька різновидів таких систем. Один з них - це система управління загальним доступом, що використовується для керування певним проектом, документом чи процесом, контролюючи усю комп'ютерну систему. Інший тип систем спільної роботи - відеоконференції або обмін миттєвими повідомленнями. Ці платформи надають можливість проводити зустрічі з віддаленими співробітниками або обмінюватися даними через Інтернет за допомогою корпоративного месенджера.

Системи управління знаннями застосовуються для збору, обробки та зберігання цінної інформації та досвіду, накопиченого під час діяльності компанії. Шляхом акумуляції необхідних відомостей про конкретні процеси та настанови, ці системи сприяють оптимізації дій, необхідних для вирішення схожих задач у майбутньому. Ця інформація може включати тексти, зображення, поєднані з документами, методики проєктування, приклади найкращих практик та інше. Прикладом реалізації подібної системи може бути використання корпоративної wiki-системи, яка дає змогу обмінюватися передовим досвідом між різними співробітниками організації. Варто також підкреслити, що оскільки організаційний досвід часто має радше неформальний, ніж формальний характер, такі системи повинні скеровувати користувачів до тих працівників організації, які володіють специфічними знаннями.

Рівень управлінської підтримки.

Найвищий щабель інформаційних систем управління призначений для забезпечення організаційного контролю та аналізу даних, зібраних внаслідок функціонування систем нижчого рівня, наприклад, систем обробки транзакцій. Така потреба виникає через специфіку управління цими системами, яка полягає

у необхідності ухвалення неструктурованих рішень, що ґрунтуються як на внутрішній інформації, що надходить з інформаційних систем, так і на зовнішніх даних, отриманих від ділових партнерів, постачальників та клієнтів. Цей рівень також може позначатися як рівень стратегічного управління. Системи, які використовуються на цьому рівні, включають системи управлінської звітності, системи підтримки прийняття рішень та інформаційні системи управління.

Системи управлінської звітності розроблені для щоденного забезпечення деталізованими та масштабними інформаційними звітами, що враховують специфіку кожної зони відповідальності керівництва. Зазвичай, такі системи є знаряддям праці для менеджерів одного рівня, наприклад, керівників та супервайзерів конкретних підрозділів, скажімо, відділу збуту або ж відділу обслуговування клієнтів. Менеджери використовують ці системи для аналізу даних та результатів діяльності команд підрозділів за конкретний період часу, минулий чи поточний, і значно рідше – для прогнозування майбутніх результатів або як один з факторів визначення нових стратегічних цілей організації чи конкретного виробничого відділу.

Системи підтримки прийняття рішень (СППР) створені для професійного опрацювання великих масивів даних з метою отримання висновків, що скеровують стратегічне планування організації. Ці системи часто класифікують як додатки бізнес-аналітики. Існує дві основні категорії СППР: системи, що базуються на моделях, та системи, що базуються на даних. Перший тип СППР функціонує на основі обмеженого набору даних та заздалегідь розробленої моделі. Приміром, коли компанія мусить визначити вартість нового виробу, торговий менеджер здатний застосувати маркетингову систему прийняття рішень. Ця система містить модель, що охоплює ряд змінних, на кшталт ціни виробу, його собівартості та витрат на ЗМІ/рекламу, для передбачення обсягів продажів упродовж перших п’яти років на ринку. Маніпулюючи компонентами самої моделі, менеджери мають можливість співставляти результати прогнозування та обирати найбільш вигідну ціну продажу.

Системи, що спираються на дані, послуговуються алгоритмами для аналізу великих масивів інформації, накопиченої за певний період у сховищах. Ці системи прагнуть передбачити певні статистичні моделі, ґрунтуючись на вже наявних даних, а управлінці можуть використовувати отримані дані як ключові показники ефективності під час роботи з системою. Доволі часто для розробки інформаційних систем цього типу застосовуються методи штучного інтелекту, бо навчені нейронні мережі здатні виокремлювати необхідну для прогнозування результатів інформацію з великих обсягів неструктурованих даних.

Інформаційні системи для керівництва - це платформи, що демонструють великі масиви ключової інформації у стислому та зрозумілому форматі, переважно через графічні панелі. Ці системи використовуються управлінським апаратом компанії, зокрема генеральним директором, вищим керівництвом та радою директорів, задля моніторингу діяльності, оцінки ринкових умов та визначення стратегічного вектору розвитку компанії. Інформація, яку керівництво використовує в цих системах, містить звіти з усіх внутрішніх інформаційних систем малого та середнього бізнесу (МСБ), результати аналізу ринку конкурентами та загальні економічні тенденції на ринку. Застосування таких систем здатне полегшити процес планування майбутнього розвитку підприємства, базуючись виключно на актуальній узагальненій інформації [20, 21].

У класичній теорії управління інформаційними системами виокремлюють кілька основних типів ІС, які відповідають різним рівням організації та прийняття рішень [22]. До них зараховують:

- **Системи обробки транзакцій (Transaction Processing Systems, TPS).** Це основні інформаційні системи для автоматизації щоденних операцій і транзакцій в організації. В освітньому закладі роль TPS можуть виконувати, приміром, електронні журнали та реєстри, що фіксують операції (відвідуваність, оцінки, розклад, нарахування заробітної плати та інше). TPS забезпечують збір детальних даних та їх збереження у базах

даних, підтримують поточну діяльність на операційному рівні. Характерна риса таких систем – великий обсяг введення та зберігання інформації, регулярні оновлення і висока стабільність. У школі прикладом TPS є система електронного класного журналу, де кожна оцінка або запис про відвідування – це транзакція, що зберігається в системі. TPS є основою для отримання інформації на вищих рівнях – управління та аналітики.

- **Офісні інформаційні системи (Office Automation Systems, OAS).** Це комплекси, спроектовані для сприяння щоденній офісній діяльності, управлінню документами та взаємодії. До їх числа входять електронна пошта, системи електронного документообігу, текстові редактори, таблиці, платформи для спільної роботи. В освітніх установах офісні ІС надають адміністрації та персоналу інструменти для підготовки документів (наказів, звітів), планування нарад, обміну інформацією між педагогами та батьками (наприклад, електронна розсилка). Такі системи збільшують ефективність праці співробітників навчального закладу, автоматизуючи типові офісні операції та сприяючи переходу на безпаперовий документообіг. Наприклад, в Україні наказом МОН № 707 від 08.08.2022 було дозволено електронне ведення шкільної документації (класних журналів) без дублювання на паперових носіях [23], що стало вагомим кроком до повної автоматизації офісних процесів у школах.
- **Системи управління знаннями та контентом (Knowledge Work Systems (KWS); Content Management Systems (CMS)).** До цієї групи входять інформаційні системи, що сприяють продукуванню нового знання та керуванню інформаційними надбаннями. В університетах це можуть бути наукові бібліотеки, сховища наукових робіт, платформи підтримки наукових досліджень. В освітніх закладах KWS представлені електронними бібліотеками, базами даних навчальних матеріалів, системами управління навчальним контентом (наприклад, системи, які дають змогу викладачам розробляти та структурувати цифрові навчальні

курси, презентації, тестові завдання). До цієї категорії належать також системи електронного освітнього контенту та платформи масових відкритих онлайн-курсів (MOOC). Ключова функція таких систем – накопичення знань та забезпечення простого доступу до них для учнів, студентів, викладачів і науковців.

- **Менеджмент-інформаційні системи (Management Information Systems, MIS).** Це системи, що функціонують на рівні менеджменту організації, забезпечуючи керівників середньої ланки необхідною інформацією для планування, контролю та прийняття обґрунтованих рішень. MIS збирає дані з операційних систем (TPS), об'єднує їх та формує звіти, показники, аналітику щодо функціонування установи. У школі прикладом MIS може бути комплексна ІС управління навчальним закладом (англ. EMIS – Educational Management Information System), що включає модулі обліку успішності, кадрового складу, фінансів, матеріальних ресурсів, та генерує аналітичні звіти для директора та заступників. Такі системи дають змогу відслідковувати успішність окремих учнів та класів, аналізувати якість освіти, керувати розкладом, навантаженням вчителів тощо. Впровадження MIS в освіті сприяє прозорості та обґрунтованості управлінських рішень. Дослідження свідчать, що сучасні заклади вищої освіти дедалі активніше використовують MIS та подібні системи для оптимізації своєї діяльності [24]. Різновидами MIS є, наприклад, системи класу ERP для освіти (Enterprise Resource Planning – планування ресурсів закладу), які інтегрують різні аспекти діяльності ВНЗ або великої школи – від зарахування студентів до випуску та адміністративно-господарських процесів.
- **Системи підтримки прийняття рішень (Decision Support Systems, DSS).** DSS розроблені для топ-менеджерів та аналітиків, сприяють прийняттю нетипових, малоструктурованих рішень, базуючись на даних. В освітній сфері DSS можуть бути застосовані керівництвом освітніх органів або університетів для аналізу статистичних даних, створення

моделей розвитку, прогнозування набору студентів та оцінки ефективності нововведень. Ці системи, як правило, інтегрують дані внутрішніх інформаційних систем з зовнішніми джерелами (наприклад, демографічною інформацією, результатами досліджень) та надають інструменти моделювання, аналізу "що-якщо", а також візуалізації. Прикладом DSS державного масштабу є аналітичні панелі (дашборди) Міністерства освіти, які агрегують дані з багатьох шкіл для виявлення тенденцій та проблем в освітній системі. DSS в навчальному закладі можуть допомогти, наприклад, оптимізувати розклад та використання аудиторій, або розрахувати рейтинг викладачів за певними критеріями.

- **Виконавчі інформаційні системи (Executive Information Systems, EIS) або системи вищого рівня підтримки рішень.** Це один із різновидів систем підтримки прийняття рішень (DSS), розроблений з урахуванням специфічних потреб вищого керівництва (директора установи, голів освітніх управлінь). EIS представляють собою узагальнену, стратегічно значущу інформацію в зручній для оперативного аналізу формі (графіки, ключові показники, індикатори ефективності). Для ректора великого вишу EIS може відображати, наприклад, загальну динаміку кількості студентів, фінансовий стан, рейтинги факультетів, показники якості освіти, результати акредитації та інше. Такі системи мають інтуїтивно зрозумілий інтерфейс і часто використовують інформаційні панелі (dashboard) для візуалізації ключових показників. В умовах шкіл EIS можуть бути не дуже поширеними як окремі рішення, проте елементи виконавчої інформаційної аналітики інтегруються в комплексні системи управління освітою. Наприклад, державні інформаційно-аналітичні системи, такі як Єдина державна електронна база з питань освіти (ЄДЕБО), надають керівникам освітніх органів агреговані дані національного масштабу, що може розглядатись як реалізація принципів EIS.

Слід підкреслити, що представлені види інформаційних систем не є строго розділеними – сучасне програмне забезпечення часто поєднує в собі можливості різних класів. Це особливо помітно в освітніх платформах: наприклад, стандартний комплекс для управління університетом може включати модулі транзакційного рівня (запис студентів на курси – TPS), звітності для керівництва факультетів (MIS), аналітичні блоки (частини DSS) та інші. Певні системи (наприклад, платформи дистанційного навчання) можливо розглядати одночасно і як TPS (виконання навчальних завдань – транзакції), і як MIS (звіти про успішність), і частково як KWS (управління навчальним контентом).

Інформаційні системи в освіті. Найчастіше виділяють в окремий клас – Educational Management Information Systems (EMIS), або просто освітні ІС. Вони охоплюють широкий спектр систем, від шкільного до державного рівня, включаючи електронні щоденники, системи керування навчанням, системи обліку контингенту та навчальних досягнень, бібліотечні системи, системи тестування та інші. Наприклад, науковці зазначають, що основні типи ІС в освітніх закладах містять TPS, OAS, KWS, MIS, DSS та EIS, кожна з яких функціонує на відповідному рівні [25]. Такий багатошаровий підхід дає змогу освітнім установам забезпечити як операційну діяльність (ведення інформації про учнів, оцінки, розклад), так і тактичне та стратегічне управління (аналіз успішності, прийняття рішень щодо розвитку закладу).

Отже, в установі, як ми розуміємо, існує багато різновидів інформаційних систем, що класифікуються відповідно до їх функцій та характеру інформації, яку вони опрацьовують, задля досягнення результатів, котрі відповідають потребам користувачів. Окрім цих видів та функцій інформаційних платформ, також критично важливо враховувати, в яких організаціях та спеціалізаціях впроваджена певна система. Виходячи з інформації, необхідної для використання інформаційної системи в конкретній галузі, визначається її головна функція та основні складові, з яких вона повинна складатися.

1.3. Інформаційні технології для цифрового освітнього середовища в Україні

В українському освітньому просторі зараз активно застосовуються різноманітні програмні рішення, які втілюють у життя згадані типи інформаційних систем. Наприклад, для забезпечення навчального процесу та управління використовуються електронні журнали (TPS/OAS), платформи керування навчальним контентом, такі як Moodle (здатний виконувати функції CMS та частково MIS для навчального процесу), інтегровані системи управління закладами (наприклад, Automated University Management Systems для ЗВО), а також інформаційно-аналітичні системи загальнодержавного рівня, що збирають освітню статистику (як складова DSS/EIS для Міністерства).

Перехід від традиційних, паперових підходів до навчання до цифрового формату визначає одну з ключових тенденцій у сучасній освіті. Цифрове освітнє середовище (ЦОС) – це комплексна система електронних інструментів, ресурсів та платформ, що забезпечують організацію навчального процесу та управління освітою в цифровій формі. В Україні створення та розвиток ЦОС є одним із ключових напрямів державної політики, що закріплено в Концепції цифрової трансформації освіти і науки. Ця концепція спрямована на формування єдиної екосистеми цифрових інструментів в освіті, що включає безпечне електронне освітнє середовище, розвиток інфраструктури, підвищення рівня цифрової грамотності та автоматизацію процесів з використанням даних [26]. Практичними кроками стали національні програми (наприклад, з 2020 року діє програма «Дія. Цифрова освіта», що має на меті навчити цифрової грамотності, та проекти забезпечення шкіл комп’ютерами і інтернетом, а також нормативні зміни, які узаконили використання електронних документів в освіті.

Інформаційні системи, які використовуються в українському освітньому процесі, можна розділити на два основні типи: (1) системи, призначені для підтримки безпосередньо навчання (цифрові навчальні матеріали, електронні журнали, платформи для змішаної та дистанційної освіти) – саме вони

формують електронне освітнє середовище закладу; (2) системи, орієнтовані на управління та адміністрування освіти (облік статистичних даних, адміністрування закладів, аналітичні інструменти) – вони функціонують як на рівні окремого закладу, так і на рівні регіону чи навіть всієї країни. У цьому розділі ми зосередимося на першій категорії, оскільки вона має прямий вплив на щоденну навчальну діяльність учнів та вчителів, проте також розглянемо важливі управлінські системи, що забезпечують загальний цифровий простір освіти.

Натепер в українських школах та вишах застосовують різноманітні типові платформи: як вітчизняні розробки, так і міжнародні продукти. З-поміж найпоширеніших можна вирізнити:

Електронні журнали та щоденники успішності. Це інформаційні системи, що замінюють паперові класні журнали й учнівські щоденники, пропонуючи зручні засоби для фіксації оцінок, відвідування, домашніх завдань, а також доступ батьків і учнів до цих даних онлайн. В Україні впровадження електронних журналів на рівні шкіл набуло особливої ваги в час пандемії 2020–2021 років, коли дистанційне навчання вимагало електронного обліку, а далі – в умовах воєнного часу, коли паперова документація стає менш надійною. Міністерство освіти і науки (МОН) дало школам дозвіл вести журнал виключно в електронній формі (без обов'язкового паперового дублювання) [27], а також започаткувало розробку державної безкоштовної системи електронних журналів (E-Journal). У той же час, навчальні заклади мають право вибирати комерційні чи відкриті платформи за власним бажанням. З-поміж найпоширеніших варіантів: платформа «Моя Школа» і система «Нові Знання (NZ.ua)».

– **«Моя Школа»** – Українська електронна система, що інтегрує в собі можливості класного журналу, щоденника та платформи для дистанційного навчання. Цю платформу було рекомендовано Міністерством освіти і науки України для використання у освітніх закладах [28]. Система «Моя Школа» надає вчителям можливість

виставляти оцінки, складати плани уроків, підтримувати зв'язок з учнями та їхніми батьками; учням – переглядати свої оцінки, розклад занять, отримувати завдання; батькам – відслідковувати успішність своїх дітей та отримувати сповіщення від школи. З архітектурної точки зору, «Моя Школа» - це веб-орієнтована хмарна система (доступна через браузер та мобільний додаток), до якої школи підключаються за моделлю SaaS. Це звільняє заклади освіти від необхідності утримувати власні сервери – достатньо лише інтернет-з'єднання. Постачальник послуг бере на себе технічну підтримку та оновлення програмного забезпечення. Такий підхід є вигідним для шкіл, зокрема у громадах з обмеженим штатом ІТ-фахівців. Проте використання хмарних сервісів висуває певні вимоги до безпеки даних – критично важливо, щоб провайдер забезпечував захист особистої інформації школярів. «Моя Школа» декларує відповідність вимогам захисту даних та наявність підтримки від Держспецзв’язку (мабуть, через проходження атестації комплексної системи захисту).

– «Нові Знання» (NZ.ua) – Ще одна розробка вітчизняних програмістів - платформа електронних журналів та щоденників, що доступна українським школам безкоштовно. Сервіс презентує себе як інструмент для електронних журналів і щоденників з функціями для дистанційного навчання [29]. За функціональністю "Нові Знання" дуже близька до "Моєї Школи": вчителі можуть створювати уроки в електронному журналі, оцінювати роботи учнів, заповнювати плани уроків; учні та батьки – переглядати оцінки, домашні завдання, слідкувати за успішністю, передбачено модуль обміну повідомленнями, формування звітів для адміністрації закладу освіти тощо [30]. Платформа інтегрована з ІСУО (Інформаційною системою управління освітою) для підтвердження даних шкіл, що підвищує довіру до її використання. Акцент на безпеці даних є важливою особливістю "Нових Знань": портал має позитивний висновок Держспецзв’язку стосовно комплексної

системи захисту інформації (КСЗІ), термін дії якого продовжено до кінця воєнного стану. Це означає, що "Нові Знання" відповідають державним стандартам захисту даних (зашифровані канали передачі даних, контроль доступу, резервне копіювання та інше), що є вкрай важливим при роботі з персональними даними дітей. Безкоштовність системи, ймовірно, забезпечується державною підтримкою чи грантами, що дозволяє навіть невеликим громадам використовувати електронний журнал без фінансових обмежень. За даними МОН, вже налічуються тисячі школярів та батьків, які використовують NZ.ua [31]. Перевагою є наявність мобільних додатків (Android, iOS). Водночас, викликом для таких масштабних систем є масштабованість: у випадку, коли платформою одночасно користуються сотні тисяч користувачів (особливо під час вечірнього "піку" перегляду оцінок батьками), система повинна витримувати велике навантаження на сервери та канали зв’язку.

Деякі великі міські центри мали власні розробки. Наприклад, у Києві функціонував проєкт "Єдина школа", що використовувався для ведення електронного щоденника. Проте, станом на 2023 рік, спостерігається загальнонаціональна тенденція переходу або на єдину державну платформу eJournal, або на використання рекомендованих Міністерством освіти та науки платформ, згідно з вибором школи. Державну систему eJournal розроблено Інститутом освітньої аналітики спільно з Міністерством. Наприкінці 2022 – на початку 2023 року система проходила апробацію: близько третини шкіл взяли участь у випробуванні [32]. Державний е-журнал гарантує безоплатність та захист, а також автоматизований збір даних для освітньої статистики. Є підстави сподіватися, що після успішного тестування система буде розгорнута повсюдно.

- Системи керування навчанням і контентом (Learning Management Systems, LMS). Якщо електронні журнали головно автоматизують облік та адміністративні аспекти навчання, то LMS зосереджуються на самому процесі навчання, надаючи цифрові інструменти для створення й

проведення занять, спілкування між викладачем та учнями, оцінювання знань. Найбільш поширеним представником LMS в українській освіті є Moodle – відкрита система управління навчальним контентом. Moodle активно використовується у закладах вищої освіти України вже багато років, а з початком пандемії її почали впроваджувати і деякі школи та коледжі. Moodle дозволяє викладачу розмістити курс (лекційні матеріали, завдання, тести), керувати доступом студентів, спілкуватися на форумі, оцінювати виконані роботи. Вона підтримує модуль журналу оцінок, генерацію звітів, тому може частково слугувати й електронним журналом. Переваги Moodle: безкоштовність, гнучкість (велика кількість плагінів), можливість розгортання як на власному сервері, так і використання у хмарі. Багато університетів інтегрували Moodle зі своїми інформаційними системами (наприклад, з системами деканату) для автоматичного перенесення списків студентів, оцінок тощо. У школах впровадження Moodle відбувалося повільніше, адже для його налаштування і підтримки потрібні технічні навички. У воєнний час Moodle стала основою для проекту «Всеукраїнська школа онлайн» (ВШО) – національної платформи дистанційного навчання для учнів 5–11 класів. ВШО функціонує на базі LMS Moodle, наповненої готовим контентом (відеоуроки, тести) згідно зі шкільною програмою [33]. Ця платформа стала доступною для всіх українських школярів та вчителів, її можна використовувати як допоміжний ресурс або як основу навчання, якщо школа змушена працювати дистанційно (актуально під час вимушеного дистанційного навчання, наприклад, у період пандемії чи воєнного стану). ВШО забезпечує учнів доступом до якісного перевіреного контенту, а вчителям – методичну підтримку та можливість відстежувати успішність учнів. Окрім Moodle, деякі заклади застосовують інші LMS: Google Classroom (як частину Google Workspace), Canvas, ClassDojo (для молодшої школи) тощо.

- **Хмарні освітні платформи (Google Workspace for Education, Microsoft Teams тощо).** Широкого поширення зазнали комплекси сервісів від глобальних IT-компаній, адаптовані для освітніх потреб. Google Workspace for Education (раніше відомий як G Suite for Education) – це набір хмарних застосунків від Google, що безкоштовно надається закладам освіти. Він включає електронну пошту Gmail на домені школи, Google Classroom (спрощена LMS для розподілу та збору завдань), Google Диск для зберігання матеріалів, Google Docs/Sheets/Slides для спільної роботи з документами, а також Google Meet для проведення відеоуроків. Багато українських шкіл перейшли на Google Workspace під час карантину, оскільки цей інструментарій не потребував власних серверів і міг бути швидко розгорнутий. Схожий комплект пропонує Microsoft – Office 365 Education (тепер часто інтегрований у платформу Microsoft Teams для освіти), який включає аналоги сервісів (OneDrive, Outlook, SharePoint, Teams для проведення конференцій та занять). Переваги хмарних пакетів: висока надійність і масштабованість (інфраструктура Google/Microsoft витримує велику кількість користувачів), багатофункціональність, постійні оновлення і підтримка. Недоліки: залежність від інтернет-з’єднання та зовнішніх серверів, питання безпеки даних (особисті дані учнів зберігаються за кордоном). Проте, компанії декларують відповідність вимогам GDPR і загалом забезпечують високий рівень безпеки. Для України важливо, що доступ до таких платформ має бути безперервним – у 2022 році виникали ризики, пов’язані з кібератаками або можливим відключенням українських користувачів, але натомість спостерігалася значна підтримка: Google і Microsoft розширили для України можливості своїх освітніх програм, беручи до уваги воєнні умови. Таким чином, хмарні сервіси стали невід’ємною частиною цифрового освітнього простору, доповнюючи або замінюючи локальні інформаційні системи.

Проблемні аспекти впровадження інформаційних систем в освіті. Попри безперечні плюси (швидкий доступ до інформації, зрозумілість, інтерактивне навчання, легкість управління), інтеграція інформаційних систем у освітній процес наштовхується на низку труднощів, особливо технічних:

- *Технічне забезпечення та інфраструктура.* Не всі освітні установи мають належне комп'ютерне обладнання. До сьогодні існують школи, зокрема в селах, які відчувають нестачу комп'ютерів чи швидкісного інтернет-з'єднання. Незважаючи на державні інвестиції у забезпечення шкіл інтернетом, виникають перешкоди, зокрема у воєнний час - пошкодження інфраструктури та періодичні відключення електроенергії, що негативно впливає на стабільне функціонування інформаційних систем. Крім того, не всюди є ІТ-фахівці: невеликі школи можуть не мати системного адміністратора для налаштування, наприклад, Moodle. Ці обставини гальмують процес цифровізації та потребують або спрощених хмарних сервісів, або централізованої підтримки.
- *Інтеграція різних систем.* Освіта - це складна система, що охоплює безліч аспектів, таких як навчальні матеріали, оцінювання знань, управління персоналом, бібліотечні ресурси, фінансування та багато іншого. Існує небезпека розділення, коли кожна задача вирішується окремою інформаційною системою, що не взаємодіє з іншими. Це може призводити до дублювання інформації та плутанини для користувачів, які змушені працювати з багатьма різними інтерфейсами. В Україні питання інтеграції є надзвичайно актуальним: скажімо, електронний щоденник школи сам по собі, платформа дистанційного навчання окремо, державні реєстри окремо. Зараз здійснюються спроби об'єднання даних. Державна е-платформа, такі проекти як «Всеукраїнська школа онлайн», «Єдина освітня інформаційна система» мають на меті забезпечити спільні стандарти даних та обмін інформацією. Проте, інтеграція вимагає узгоджених форматів та API. На рівні закладу, рішенням може бути впровадження комплексної системи (наприклад, єдиний портал, що

включає в себе і щоденник, і Moodle, і бібліотеку). Фактично, університети часто замовляють інтегроване рішення (як згаданий в дослідженні комплекс «Politec-Soft» для управління освітнім процесом). В школах частіше спираються на готові платформи, які поступово теж інтегруються з іншими – так, «Нові Знання» інтегровані з державною базою ІСУО, а Google Classroom можна об'єднати з Moodle та іншими.

- *Безпека та захист даних.* Інформаційні системи освіти оперують чутливими особистими даними неповнолітніх, тому питання кібербезпеки є критично важливим. Загрози включають несанкціонований доступ (хакерські атаки, витік даних учнів, оцінок), навмисне псування або знищення даних, а також ризики шахрайства. В умовах гібридної війни українські ресурси піддаються кібератакам, і освітні сайти не виняток. Наприклад, траплялися масовані DDoS-атаки на платформи дистанційного навчання. Держава через Держспецзв’язку вимагає від інформаційних систем, що використовуються в державних установах, сертифікації КСЗІ – як ми згадували, «Нові Знання» та деякі інші платформи її отримали. Це забезпечує базовий рівень довіри. Проте не всі комерційні продукти мають такі сертифікати. Школи часом легковажно підходять до вибору ІС, користуючись безкоштовними веб-сервісами без гарантій захисту. Це проблема, яку варто вирішувати просвітницькою діяльністю та регулюванням. Ще один важливий момент безпеки - забезпечення безперебійної роботи. Необхідно турбуватися про резервне копіювання журналів та баз даних: у випадку збою системи чи знищення сервера (що в реаліях війни є більш ніж вірогідною загрозою), школа повинна мати резерв своїх даних. Хмарні рішення частково вирішують це питання, оскільки дані зберігаються у віддалених дата-центрах з резервуванням, однак тут виникає питання довіри до провайдера. Відтак, безпека ІС в освіті – це комплекс заходів: від технічних аспектів (шифрування, резервування) до організаційних дій (навчання користувачів кібергігієни, сертифікації систем).

- *Масштабованість і продуктивність.* Коли одночасно функціонують сотні або тисячі юзерів (наприклад, під час онлайн-занять або при масовому виставленні семестрових оцінок), інформаційна система мусить мати відповідну продуктивність. На початку пандемії деякі платформи не витримували навантаження: відомі випадки, коли всеукраїнські онлайн-уроки на одному сервері збирали забагато відвідувачів, і ресурс ставав недоступним. Тому масштабованість – ключовий параметр. Хмарні глобальні сервіси (Google, Microsoft) практично не мають проблем з цим, але локальні або державні системи мають бути спроєктовані з розрахунком на пікові навантаження. Використання хмарних технологій і контейнеризації – один із шляхів вирішення: державний е-журнал, імовірно, розгорнуто в державному приватному хмарному середовищі, яке можна масштабувати. Якщо ж школа розгортає систему на власному сервері, її потужностей може не вистачити в довгостроковій перспективі. Отож, більшість закладів віддають перевагу SaaS-рішенням. Продуктивність також страждає через старе обладнання – й досі в деяких школах комп'ютери слабкі, і навіть веб-інтерфейси на них функціонують повільно. Вихід – поступове оновлення парку техніки, що теж закладено в плани МОН.
- *Людський фактор і прийняття технологій.*Технічна реалізація системи – це лише перший крок, важливо, щоб її активно застосовували. Перешкодами часто виступають консервативність та надмірне навантаження вчителів паперовою тяганиною. Поки електронний журнал дублював паперовий, більшість викладачів не відчували потреби заповнювати обидва, і ставились до електронного формально. З ліквідацією паперового дублювання ситуація має покращитись. Проте навчання персоналу, надання методичних рекомендацій – вкрай важливі. Окремі дослідження вказують на дефіцит часу у працівників освіти для опанування нових ІКТ – суттєва проблема. Це особливо актуально в умовах війни, коли навантаження зростає, а обставини ускладнюються.

Тут на допомогу приходять інтуїтивно зрозумілі інтерфейси та спрощення систем: чим простіше використовувати, тим швидше відбудеться адаптація.

Підсумовуючи, поточний стан впровадження інформаційних систем в освіті України визначається активним розширенням цифрової інфраструктури, виникненням як державних, так і приватних платформ, призначених для потреб освіти. Держава приділяє значну увагу нормативному регулюванню та підтримці, що виражається в рекомендаціях МОН, програмах оснащення та розробці національних платформ. Водночас постають технічні й організаційні труднощі: оснащення кожної школи потрібним устаткуванням та підключенням до мережі Інтернет, поєднання різних інформаційних систем в єдину платформу, захист даних учасників навчального процесу, розгортання рішень по всій державі, підготовка кваліфікованих спеціалістів. Незважаючи на ці складнощі, поступ є помітним – за останні роки українські школи масово запровадили електронні журнали, дистанційне та змішане навчання стали реальністю завдяки LMS і хмарним сервісам, розроблено централізовані сховища освітніх матеріалів (ВШО) та інше. Отже, інформаційні системи перетворилися на невід’ємний елемент сучасного освітнього процесу, а їх подальше удосконалення та розвиток (враховуючи розв'язання наявних проблем) є ключем до покращення якості та доступності освіти в Україні в цифрову епоху.

1.4. Компоненти систем інформаційних технологій

Інформаційні системи, зазвичай, включають п’ять ключових елементів: комп’ютерне обладнання, програмне забезпечення, дані, люди та процеси. Структуру компонентів інформаційної системи можна зобразити на діаграмі, як показано на рисунку 1.2. В останні роки суттєвим компонентом стала також комунікаційна мережа. Перші три основні компоненти визначаються як інформаційні технології, або IT [34]. Не випадково останні два компоненти, тобто люди та процеси, є важливими складовими інформаційних систем,

оскільки вони дозволяють відокремити концепцію інформаційних систем від більш технічних аспектів, наприклад, від інформаційних технологій в інформатиці.

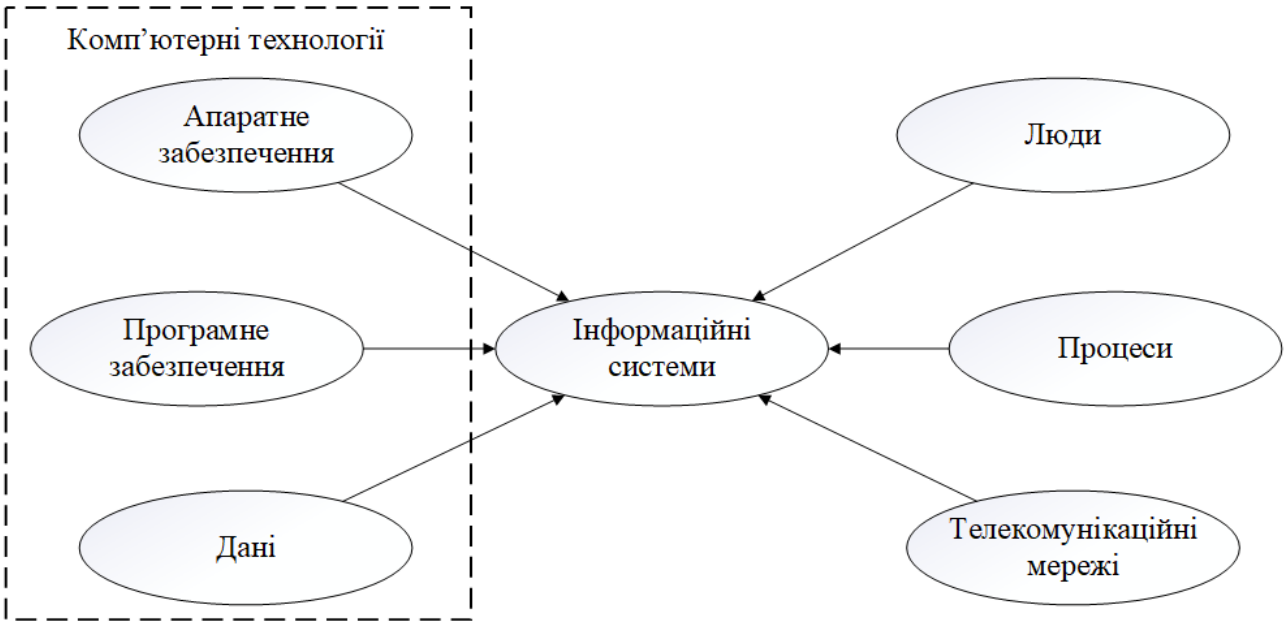


Рисунок 1.2. Схематична діаграма компонентів інформаційної системи

Розглянемо кожну складову частину окремо:

Комп'ютерне обладнання

Апаратне забезпечення в інформаційних системах – це фізичні складові, до яких можна доторкнутися. Їхнє призначення - обробка інформації на апаратному рівні всередині системи. Ці фізичні компоненти включають все, що використовують користувачі для досягнення результатів взаємодії з системою. Це комп'ютери, ноутбуки, клавіатури, миші, віддалені сервери, дискові накопичувачі, флеш-накопичувачі та інші пристрої. Разом, всі ці елементи формують комп'ютерне обладнання інформаційної системи конкретного підприємства.

Програмне забезпечення комп'ютера

Програмне забезпечення в інформаційних системах — це складова, котра функціонує як сукупність важливих команд, що визначають дії апаратних засобів. Його ключовою особливістю є нематеріальність, тобто відсутність

фізичної форми. Зазвичай, цей компонент поділяють на два основні типи: операційні системи (системне ПЗ) та прикладне програмне забезпечення.

Перший клас, як правило, відіграє роль посередника між обладнанням та прикладним програмним забезпеченням. Операційні системи забезпечують користувачеві інструменти для коректного управління апаратурою, організації даних та програмних файлів, а також для "розумного" контролю над комп'ютером.

Прикладне програмне забезпечення – це інструмент для вирішення конкретних задач користувачів. Основними компонентами цього різновиду є призначені для користувача програми та додатки, які функціонують під контролем операційної системи, створеної саме для них.

Дані

Дані можливо подати як сукупність взаємопов’язаних фактів. Скажімо, домашня адреса конкретної особи (держава, населений пункт, вулиця, номер будинку), особиста інформація у певній соціальній мережі, опис конкретної події, що нещодавно відбулася десь на планеті, – все це компоненти певних даних. Зазвичай нерелевантні фрагменти даних не є надто корисними для інформаційних систем. Проте, якщо дані згруповані, відфільтровані та структуровані в кілька баз даних, вони перетворюються на основний інструмент для ведення бізнесу в межах конкретної інформаційної системи.

Аналізування й управління такими блоками інформації здатно допомогти організації відслідковувати перебіг прийняття рішень, видавати специфічні результати обробки конкретним користувачам системи та в цілому контролювати ефективність бізнесу.

Інформаційні блоки - це ключові елементи інформаційної системи бізнесу, застосовувані для аналізу й оперування даними. Розглядаючи ці три окремі складники разом, їхню структурну модель можна зобразити так, як показано на рисунку 1.3.

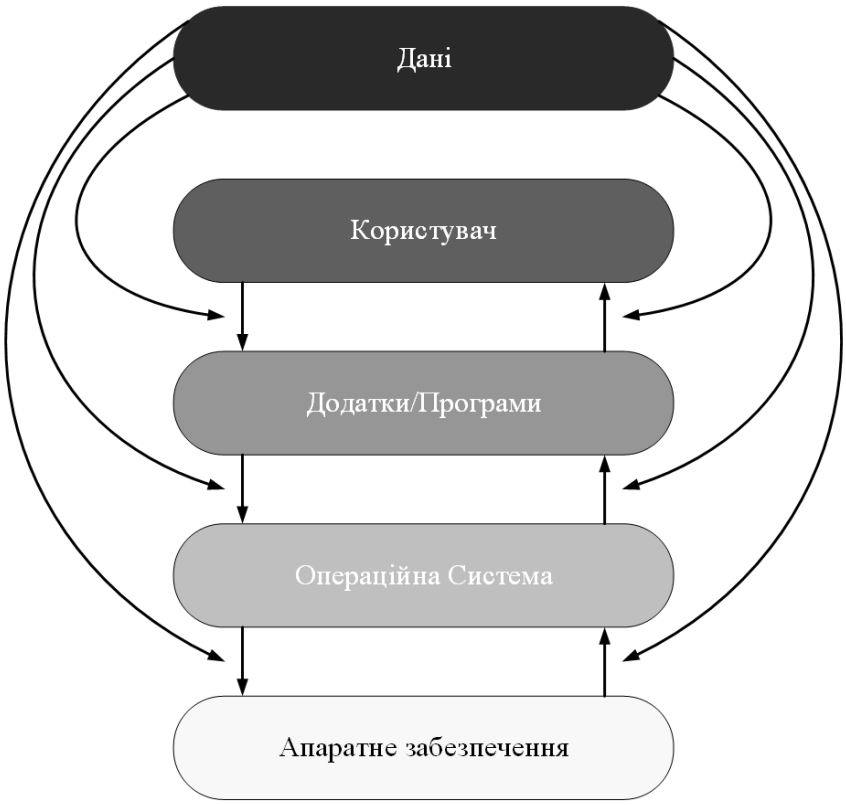


Рисунок 1.3. Взаємодія користувача з трьома компонентами технології інформаційних систем [35]

Згідно з представленим графіком, дані слугують ключовим компонентом, що взаємодіє на кожному етапі обробки інформації, від «заліза» до програмного забезпечення. Їхнє використання є критично важливим під час розробки будь-якої інформаційної системи.

Телекомунікаційна мережа

Мережа - це спосіб об’єднання та взаємодії між окремими комп'ютерами, різними мобільними пристроями або, ширше кажучи, між різними апаратними компонентами для обміну даними. З’єднання може бути встановлено як з використанням кабелів, так і без них. Наприклад, взаємодія між комп’ютерами в офісі або між окремими мережами в центрі обробки даних, відбувається через дротові з'єднання. Бездротові з'єднання застосовуються для підключення портативних пристроїв, смартфонів, вбудованих систем, використовуючи мікрохвилі або радіохвилі.

Окрім зв'язності, конкретні мережі класифікуються відповідно до застосовуваних технологій і протоколів з’єднання. Відповідно, окремі

користувачі отримують безлімітний доступ до отримання та обробки певних даних з інформаційних систем, незалежно від місця їх перебування. З вищесказаного випливає, що телекомунікаційні мережі – це комбінація апаратних і програмних засобів, що гарантують безперервне з’єднання для обробки даних у розподілених комп'ютерних системах, які зараз застосовуються майже в усіх відомих організаціях.

Люди

Окрім суто технічних аспектів, про які вже йшлося, вирішальним компонентом сучасних інформаційних систем є люди, що беруть у них участь. Саме люди відповідають за функціонування структурних частин кожної інформаційної системи, впровадження додаткових процесів та загальну оцінку результатів обробки даних. До них належать: розробники, менеджери з експлуатації, керівники та директори організаційних підрозділів, працівники служби підтримки користувачів та інші фахівці, без яких неможлива ефективна робота інформаційних систем.

Процес

І наостанок, але не менш важливо: процес - це послідовність дій, необхідна інформаційній системі для досягнення визначеного результату або поставленої задачі. За суттю, процеси підлягають обговоренню та документуванню під час розробки конкретної інформаційної системи, щоб результати обробки даних, отримані за її допомогою, відповідали вимогам бізнесу конкретної організації. Проте, в різних організаціях ключовою метою впровадження сучасних процесів при їх інтеграції в інформаційну систему є досягнення певного рівня автоматизації окремих задач для отримання бажаного результату опрацювання конкретних даних [36].

Автоматизація життєвих процесів студентського містечка - це багатогранне завдання, що вимагає розробки єдиного цифрового простору для керування навчальними, управлінськими, громадськими та інфраструктурними аспектами. Інформаційні системи (ІС) для студентського містечка сприяють налагодженню зв'язків між студентами, викладачами, адміністрацією та

технічними підрозділами, забезпечуючи раціональне використання ресурсів та підвищення загальної продуктивності. З огляду на сучасні підходи до дистанційного навчання й автоматизації, зокрема праці Носенка Ю. Г. та інших [37], Колос К. Р. [38], Яремчук Н. А. та інших [39], а також беручи до уваги актуальність нечіткої логіки та інфографіки, можна детальніше розглянути основні складові таких систем. Кожен з компонентів виконує специфічні задачі, взаємодіє з іншими модулями та сприяє створенню інформаційного середовища, що адаптується.

Блок баз знань та даних слугує фундаментом інформаційної системи, гарантуючи централізоване зберігання, обробку та доступ до інформаційних ресурсів студентського містечка. Цей компонент вміщує реляційні бази даних, які містять інформацію про учасників освітнього процесу, інфраструктуру, розклади занять, нормативні документи та аналітичні дані. У дослідженні Яремчук Н. А., Сікози О. М. та Кельберга С. М. [39] акцентується важливість блоку бази знань у системах дистанційного навчання для організованого зберігання навчальних матеріалів та регламентів процесу. У студентському містечку блок реалізовує наступні функції: зберігання даних про учасників освітнього процесу (Персональні дані студентів (академічна успішність, місце проживання), викладачів (графіки роботи, навчальні курси) та співробітників, забезпечуючи персоналізацію, як в адаптивній моделі); керування інфраструктурними даними (відомості про стан аудиторій, гуртожитків, бібліотек, включно з їх доступністю та технічним обслуговуванням); нормативна база (електронні копії правил, положень і регламентів, доступні через веб-портал); інтеграція з платформами (синхронізація з системами дистанційного навчання, наприклад Moodle, для обміну інформацією про курси та оцінки, що відображено у роботі Колос К. Р. [38]).

Технічно блок базується на системі управління базами даних (СУБД) MySQL, PostgreSQL з API для взаємодії в реальному часі. Скажімо, студент має змогу перевірити наявність книг у бібліотеці, а адміністрація — згенерувати звіт про використання аудиторій.

Блок вхідного контролю та моніторингу автоматично збирає та обробляє дані про актуальний стан справ у студентському містечку. Цей модуль відслідковує ступінь використання ресурсів, рівень задоволеності студентів і їхню навчальну активність. У праці Колос К. Р. [38] представлено застосування Moodle для оцінки навичок, що може бути використано для моніторингу. До складу блоку входять: модуль збору відгуків (опитування щодо якості інфраструктури чи послуг, що підвищує мотивацію, згідно з Герасименком І. В. [40]; модуль моніторингу ресурсів (відстеження зайнятості аудиторій і гуртожитків, ймовірно, з використанням IoT-сенсорів); модуль моніторингу успішності (аналіз відвідуваності та успішності, подібно до адаптивної системи Носенка Ю. Г. та інших [37]; модуль адміністративного контролю (оцінка оперативності обробки запитів і ефективності роботи персоналу). Зібрана інформація передається до блоку аналізу. Реалізація передбачає використання веб-форм, інформаційних панелей (дашбордів) та систем ведення журналів (логування).

Блок побудови особистих траєкторій адаптує сервіси до запитів користувачів, надаючи їм індивідуальний підхід. Яремчук Н. А. та співавтори [39] акцентують увагу на значущості індивідуальних траєкторій у навчанні. У студентському містечку цей блок реалізує: персоналізацію навчання (рекомендації курсів або матеріалів на основі успішності, що сприяє самостійності, як у Дубового З. С. [41]); автоматизацію розкладів (особисті графіки та сповіщення через мобільні додатки); адміністративну підтримку (автоматизовані заявки на гуртожиток або бронювання приміщень); соціальну інтеграцію (рекомендації заходів відповідно до профілю студента).

Блок аналізу та оптимізації призначений для опрацювання інформації з метою вимірювання продуктивності та вдосконалення робочих потоків. Ефективність змішаного навчання, яку підтримує Рашевська Н. В. [42], дозволяє адаптувати підходи до аналізу даних. Структура блоку передбачає наявність: аналітичного модуля (оцінює KPI (час відгуку, навантаження на обладнання)); прогностичного модуля (передбачає потребу у ресурсах на основі

аналізу часових рядів); модуля оптимізації (видає рекомендації щодо перерозподілу аудиторій чи процедур); модуля нечіткої логіки (враховує особисті фактори (зручність, комунікація)). Реалізація даного блоку може використовувати Python (pandas, scikit-learn) і BI-платформи (Power BI). Блок взаємодіє з базою знань і візуалізацією.

Блок візуалізації даних та інфографіки подає інформацію у легкій для сприйняття формі. У працях Конюхова А. Д. [43] наголошується на можливостях інфографіки. Блок містить такі елементи: інтерактивні інформаційні панелі (дашборди), що відображають статистику, наприклад, завантаженість або успішність; аналітичні звіти (графічні зображення), призначені для адміністрації, що показують ефективність різних процесів; персоналізовані інформаційні сторінки (розклади занять та оцінки для студентів, представлені графічно); інфографічні довідники (схеми кампусу або інструкції у вигляді графіків). Для реалізації використовуються Matplotlib, D3.js, Tableau. Блок працює спільно з аналітичним розділом та базою знань.

Блок безпеки та ідентифікації стоїть на варті даних, управляючи доступом. Згідно з дослідженнями Богуша В. М., Кривуци В. Г. та колег [41], використовуються унікальні ідентифікатори та шифрування для захисту. Структура блоку включає: аутентифікацію (індивідуальні ID для входу через мобільний застосунок); шифрування (безпека інформації з AES, RSA алгоритмами); рольовий доступ (обмеження прав користувачів); журналювання (реєстрація дій для перевірки). Для реалізації застосовується HTTPS, OAuth, Keycloak. Взаємодія здійснюється з базою знань та системою моніторингу.

Блок комунікації служить для організації взаємодії між користувачами системи. Про чати і форуми можна дізнатися з праці [44]. Даний модуль містить у собі: повідомлення (сповіщення через email, SMS, push-повідомлення); чат та форум (для дискусій на академічні теми та питань студентів); служба підтримки (чат-боти та тікет-системи для обробки звернень); відеоконференції (інтеграція з Zoom для проведення консультацій). В реалізації

використовуються Firebase, Discourse, Rocket.Chat. Блок комунікацій інтегрується з блоками бази знань та безпеки.

З викладеного, інформаційна система для автоматизації студентського містечка, в основному, складається із семи ключових блоків: база даних, управління та спостереження, персональні траєкторії, аналіз та оптимізація, візуалізація, захист, комунікація. Кожен блок реалізує визначені функції, від зберігання інформації до взаємодії, формуючи гнучке середовище. Застосування кожного з цих блоків збільшує продуктивність.

1.5. Формування завдань дослідження

У сфері інформаційних технологій освітнього простору, враховуючи дистанційне віртуальне навчання, спостерігається стрімкий розвиток, що зумовило появу тисяч програмних продуктів. Незважаючи на швидкий поступ, кількість ключових розробників цих продуктів обмежена; виділяються лише кілька десятків компаній, які зосереджені на перспективних розробках.

Сучасний поступ інформаційних технологій відкриває перед нами нові горизонти автоматизації управлінських процесів у закладах вищої освіти, особливо у контексті організації студентських містечок. Студмістечко як складна система охоплює управління освітніми, адміністративними, соціальними та інфраструктурними процесами, що зумовлює необхідність інтеграції різноманітних інформаційних ресурсів та технологій. В умовах стрімкого розвитку дистанційного навчання, детально висвітленого в працях Єрмоленка О. В., Ковальова В. І., Лісного А. І., Сєркова О. А., особливу увагу приділено адаптивним системам, здатним враховувати індивідуальні потреби користувачів. Схожий підхід може бути реалізований при автоматизації управління студентським містечком, де адаптація інформаційних технологій до вимог студентів, викладачів та адміністрації сприяє підвищенню загальної ефективності управління.

Автоматизація керування студентським містечком має на меті вдосконалити процеси планування, відстеження та координування різних видів

діяльності. Це охоплює розподіл ресурсів, організацію навчального процесу, підтримку інфраструктури та забезпечення зв'язку між усіма учасниками освітнього процесу. Відповідно до досліджень Герасименка І. В., застосування інформаційно-комунікаційних технологій (ІКТ) у дистанційному навчанні підвищує зацікавленість студентів через ефект новизни та сприяє самостійному навчанню. Стосовно студентського містечка, ІКТ можуть створити інтегроване інформаційне середовище, що підтримує освітні та адміністративні функції, наприклад, автоматизацію складання розкладів, керування гуртожитками, організацію подій та моніторинг успішності.

Дослідження Рашевської Н. В. акцентують увагу на продуктивності комбінованого навчання з використанням мобільних технологій, що здатне знайти застосування в управлінні студентським містечком через розробку мобільних додатків для перегляду розкладів, отримання сповіщень, резервування ресурсів та взаємодії з адміністрацією.

Водночас, як зазначає Дубовий З. С., формування самостійності студентів у дистанційному середовищі є ключовим аспектом, який може бути реалізований через автоматизовані системи, що надають студентам персоналізовані рекомендації та доступ до ресурсів відповідно до їхніх індивідуальних потреб.

Одним із перспективних шляхів автоматизації управління студентським містечком виступає застосування методів нечіткої логіки. Вони дають можливість моделювати надскладні системи, беручи до уваги невизначеність та суб'єктивні аспекти. Відповідно до аналізу цілей дослідження, нечітка логіка – це дієвий інструмент для формування якості процесів, особливо в умовах організації дистанційного навчання та інформатизації освітнього процесу загалом. У контексті студентського містечка нечітка логіка може використовуватися для оцінювання та визначення пріоритетності факторів, які впливають на якість управління, серед яких – рівень задоволеності студентів, ефективність використання наявних ресурсів, оперативність реагування на запити та адаптивність до змін у навчальному процесі.

Наприклад, адаптивна модель, що запропонована Єрмоленком О. В. та командою, використовує структуроване представлення даних, беручи до уваги рейтинг освоєння матеріалу. Подібним чином, в управлінні студентським містечком можливо створити модель, яка визначає ефективність управління, спираючись на такі аспекти, як: доступність інфраструктури, якість комунікації та ступінь автоматизації процесів. Застосування нечіткої логіки дозволяє конструювати ієрархічні моделі, які враховують вагові коефіцієнти факторів, подібно до підходу, викладеному у роботах Мочалова О. О. та Гайші О. О., де запропоновані унікальні ідентифікатори для персоналізації доступу до ресурсів.

Для розробки інформаційної технології автоматизації керування студентським містечком пропонується синтез ієрархічної моделі. Вона інтегрує компоненти навчального, адміністративного та соціального управління в єдину систему. З огляду на дослідження Колос К. Р. [38] про застосування системи Moodle для підвищення кваліфікації вчителів, існує можливість адаптувати подібну Moodle-орієнтовану модель для потреб керування студентським містечком.

Така модель може містити: блок бази даних (зібрані відомості про розклад занять, доступні ресурси, статути та вимоги студмістечка); модуль вхідного контролю (проводить аналіз актуального стану системи, наприклад, ступінь заповненості аудиторій, кількість поданих запитів до адміністрації); модуль формування персональних траєкторій (формулює індивідуальні рекомендації студентам та викладачам з урахуванням їх потреб та пріоритетів); блок аналізу та оптимізації (використовує принципи нечіткої логіки для оцінювання ефективності управління та виявлення ключових напрямів для покращення).

Така модель відповідає принципам, описаним Яремчук Н. А., Сікозою О. М. та Кельбергом С. М., де підкреслюється необхідність індивідуальних траєкторій та автоматизованого контролю в системах дистанційного навчання.

Ефективне управління кампусом потребує ясності в даних, щоб приймати виважені рішення. Як вказано у науковому дослідженні, використання інфографіки для представлення інформації є багатообіцяючим підходом, який ще недостатньо вивчено в умовах дистанційного навчання. У студентському містечку інфографіка може бути використана для: відображення статистики користування ресурсами (аудиторії, гуртожитки, бібліотеки); візуалізації результатів навчання студентів; представлення аналітичних звітів для керівництва стосовно ефективності різних процесів.

На базі методів, що були представлені Фомічовим С. К. та його колегами, інфографіку можна вбудовувати в автоматизовану систему управління за допомогою баз даних, які оновлюються в режимі реального часу й візуалізують результати роботи студентського містечка. Приміром, інформаційні панелі з інфографікою здатні демонструвати ступінь завантаженості інфраструктури або рівень задоволеності студентів, що сприяє прийняттю рішень управлінського характеру.

Аналіз наявних досліджень, зокрема праць авторки Шиліної Г. А., виявляє проблемні аспекти впровадження інформаційних технологій, що стосуються контенту, методичних підходів та організаційних структур. У розрізі студентського містечка, на нашу думку, першочергове значення мають такі виклики: об'єднання різнорідних систем (потреба інтегрувати навчальні, адміністративні й соціальні платформи в єдине інформаційне поле); пристосування до індивідуальних запитів (врахування різноманітних вимог студентства, викладацького складу та адміністративного персоналу); забезпечення захисту інформації (як відзначають Мочалов О. О. та Гайша О. О., застосування унікальних ідентифікаторів і шифрування є ключовим елементом для забезпечення безпеки даних).

Перспективи впровадження ІТ охоплюють розбудову адаптивного, взаємопов'язаного та захищеного простору, що сприятиме автоматизації керування університетським містечком. Застосування нечіткої логіки, відповідно до пропозицій у науковій роботі, надає можливість вдосконалити

процес прийняття рішень, беручи до уваги неоднозначність та особисті аспекти, що набуває особливого значення у складних системах, як-от університетське містечко.

Розробка інформаційної технології автоматизації управління студентським містечком становить важливу наукову задачу, що ґрунтується на поєднанні методів нечіткої логіки, системного аналізу та сучасних інформаційних технологій. Об'єднання ієрархічної моделі управління, застосування інфографіки для візуалізації даних та врахування індивідуальних потреб учасників освітнього процесу відкриває можливості для збільшення ефективності управління. Наступні дослідження слід зосередити на експериментальній перевірці запропонованої моделі та створенні методичних рекомендацій для її впровадження у закладах вищої освіти.

Висновки до розділу 1

Розділ зосереджується на дослідженні актуального стану справ та викликів, пов'язаних із використанням інформаційних систем (ІС) в освіті, з акцентом на їхнє застосування для автоматизації управління університетським містечком. Здійснений аналіз дав можливість сформулювати важливі висновки, що об'єднують теоретичні та практичні аспекти інтеграції ІС в освітнє середовище.

1. Функція та застосування ІС в освіті увиразнюють їхню ключову роль в автоматизації навчальних, управлінських, інфраструктурних та соціальних процедур. Інформаційні системи забезпечують індивідуалізацію навчання, раціоналізацію управління ресурсами, підтримання комунікації та аналітичну обробку даних, що сприяє покращенню ефективності освітнього процесу. Практичне використання ІС включає інтеграцію платформ, наприклад, Moodle, автоматизацію запитів та розробку адаптивних навчальних середовищ. Водночас, такі виклики, як інтеграція систем та навчання користувачів, вимагають подальшого дослідження.

2. Класифікація інформаційних систем демонструє розмаїття цих систем відповідно до функцій, архітектури та ступеня автоматизації. У сфері освіти актуальними є системи управління навчанням (LMS), студентські інформаційні системи (SIS) та платформи для адміністративних потреб. Цей розподіл дозволяє точно з'ясувати, які інформаційні системи найкраще відповідають вимогам студентського містечка, зокрема для підтримки індивідуалізованих освітніх траєкторій.

3. Дослідження інформаційних систем (ІС) в українському цифровому освітньому середовищі демонструють інтенсивне використання цифрових інструментів у вітчизняних вишах. Це виражається, зокрема, у застосуванні платформ для віддаленого навчання та мобільних додатків. Аналіз підкреслює ключову роль комбінованого формату навчання та мобільних рішень у забезпеченні більшого доступу до освітніх можливостей. В той же час, виклики, такі як нерівномірний доступ до необхідного обладнання та недостатньо повна інтеграція інформаційних систем, зберігають свою актуальність.

4. Компоненти ІС охоплюють блоки бази знань, перевірки вхідних даних, особистих траєкторій, аналізу, візуалізації, безпеки та зв'язку. Кожен із них виконує важливу функцію у створенні цілісного середовища. Застосування нечіткої логіки та інфографіки дає змогу ефективно керувати складними процесами та покращує наочність даних.

5. Визначення цілей дослідження виявило недоліки у сучасних наукових розробках, зокрема, у використанні нечіткої логіки та інфографіки для вдосконалення якості автоматизації освітнього процесу. Огляд наукових праць підкреслює потребу у створенні нових підходів до розробки адаптивних ІС, які враховують персональні потреби та сприяють оптимізації управління університетським містечком.

РОЗДІЛ 2.
ДОСЛІДЖЕННЯ ФАКТОРІВ ВПЛИВУ НА ПРОЦЕС РЕАЛІЗАЦІЇ ТА ВИКОРИСТАННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ АВТОМАТИЗОВАНОЇ СИСТЕМИ ЗАБЕЗПЕЧЕННЯ ДІЯЛЬНОСТІ СТУДЕНТСЬКОГО МІСТЕЧКА

2.1. Виокремлення факторів, які впливають на функціонування інформаційної технології автоматизованої системи забезпечення діяльності студентського містечка

Сучасне виробництво та надання послуг, випуск товарів, потребують інформаційних технологій, які задовольняють інформаційні потреби менеджменту, виробництва, постачання, маркетингу, збуту та інших відділів. Інформаційні технології дають змогу ефективно управляти всіма видами ресурсів підприємства. Своєчасна та актуальна інформація дозволяє зосереджувати ресурси там, де це потрібно, для реалізації основних, пріоритетних задач у потрібний час.

Окрім того, інформаційні технології відкривають ширші обрії для професійного зростання фахівців, даючи їм змогу вести господарську діяльність підприємств значно ефективніше, з чітко окресленими цілями, економно та з максимальним результатом. Стратегічне значення інформаційних технологій в сучасному світі незаперечне: вони суттєво впливають на економіку, оперативно адаптуються до ринкових змін, зберігають і посилюють конкурентні переваги для отримання максимального прибутку. Застосування інформаційних систем кардинально змінює підходи до управління та значно покращує продуктивність компанії [45].

Ось ключові вектори вдосконалення організаційного менеджменту: розгортання інформаційних систем (ІС), базованих на передовому апаратному та програмному забезпеченні, інтеграція інформаційних технологій, реалізація

розподіленої обробки даних у мережах, застосування економіко-математичних методів та моделей, а також використання систем підтримки прийняття рішень.

Мета створення ІС полягає у детальному описі об'єкта дослідження, його умов функціонування та взаємодій, що виражені через набір конкретних показників. ІС служить для своєчасного забезпечення управлінського апарату необхідною та достатньою інформацією, критичною для ухвалення обґрунтованих рішень, а також сприяє підвищенню ефективності діяльності керівництва та підпорядкованих підрозділів [46].

Не оминула змін і освітня сфера. Застосування інформаційних технологій в освітньому процесі стало особливо помітним під час введення карантину та воєнного стану в Україні.

Проаналізувавши новітні дослідження та публікації, можна відзначити, що вони здебільшого присвячені впровадженню та вивченню інформаційних технологій для дистанційного навчання. Наприклад, у роботі [47] автор розглядає питання порівняння використання інформаційних технологій з традиційними методами управління закладом вищої освіти, вказуючи на суттєве покращення сприйняття навчального матеріалу завдяки візуалізації інформації. Інформаційні технології, як свідчать дослідження, також сприяють підвищенню мотивації до навчання.

У праці [47] підкреслюється значення використання сучасних цифрових освітніх технологій, які базуються на особистісно-орієнтованому підході та компетентнісному навчанні, у процесі організації професійної підготовки студентів для здобуття вищої освіти.

Питання якості навчання студентів в умовах цифрового освітнього простору та дистанційного навчання було предметом дослідження науковців [48-51]. У своїх роботах дослідники звертають увагу на необхідність вдосконалення цифрового освітнього середовища для сучасного етапу професійної підготовки здобувачів вищої освіти. Це, зокрема, повинно сприяти ефективному формуванню цифрових та дослідницьких компетентностей майбутніх інженерів-програмістів, фахівців з комп’ютерних наук, а також

інших спеціальностей. Одним з ключових елементів програмно-методичного забезпечення сучасного університетського простору є розробка та впровадження інформаційних систем, спрямованих на професійну підготовку студентів.

Стосовно друкованих праць, котрі б висвітлювали розробку та експлуатацію інформаційних систем, пов'язаних з навчальним процесом в університетах, включаючи проєктування та застосування інформаційної системи студентського містечка, нам не вдалося відшукати. На нашу переконання, саме інтеграція всіх інформаційних систем освітнього процесу відкриває широкі перспективи для усіх, хто залучений до нього: сприяє їхній кращій адаптації, отриманню максимально релевантної інформації та якісних послуг.

Для здійснення нашого дослідження оптимальним є проведення опитування серед учасників освітнього процесу з метою з'ясування їхньої думки щодо ключових факторів, які впливають на процес проєктування інформаційної системи управління студентським містечком. На основі цього опитування нами було ідентифіковано основні аспекти, що впливають на вибір та впровадження інформаційних технологій для автоматизації освітнього процесу, зокрема, функціонування студентського містечка, після чого проведемо їх детальний аналіз.

Параметр «користувач» – це фундаментальний елемент при розробці інформаційних систем для студентського містечка. Визначення та характеристика користувачів відіграють важливу роль у створенні продуктивних інтерфейсів, плануванні функціоналу та визначенні прав доступу.

Головними категоріями користувачів виступають студенти, адміністративний апарат студмістечка, співробітники охорони, технічний персонал та батьки студентів. Для студентів передбачено широкий спектр послуг: доступ до особистої інформації, перегляд розкладу занять, подача заяв на поселення, перевірка наявності вільних місць, повідомлення про проблеми в

кімнатах, участь у соціальних подіях. Користуючись інформаційною системою, студенти потребують швидкого доступу до інформації про статус заявок на проживання, а також можливості отримувати сповіщення про суттєві зміни.

Для адміністрації студмістечка, інформаційна система повинна охоплювати керування заявками на поселення, відстеження стану кімнат, розподіл ресурсів (місць в гуртожитках), генерування звітів та організацію заходів для студентів. Вимоги до системи для адміністраторів можна зобразити як наступне: наявність зручної інформаційної панелі, можливість перегляду статистики житлової площі, контроль за дотриманням правил, аналіз даних про студентів.

Щодо працівників служби безпеки, їхній інформаційний ресурс має забезпечувати контроль доступу до гуртожитків, реєстрацію відвідувачів, моніторинг безпеки та фіксацію порушень. Для них система має надавати швидкий доступ до бази даних студентів; до систем реєстрації подій та інцидентів; та до систем сповіщень про порушення.

Наступною категорією користувачів інформаційної системи є технічний персонал. Для цієї групи користувачів передбачені такі функції: керування технічними заявками (ремонт, обслуговування); проведення інвентаризації обладнання у студентських гуртожитках. Інформаційна система для цієї категорії користувачів повинна забезпечувати інтуїтивно зрозумілий інтерфейс для оформлення заявок на ремонт та відстеження їхнього статусу; управління технічним обслуговуванням будівлі.

Стосовно батьків студентів, варто зазначити, що не всі інформаційні системи передбачають цю категорію користувачів, іноді такі послуги відсутні. Але якщо система надає доступ для батьків, то вони повинні мати можливість переглядати інформацію про успішність своєї дитини; отримувати доступ до даних про місце проживання та стан здоров'я дитини. Ця інформація повинна бути актуальною і містити дані про умови проживання, безпеку та наявність гуртожитків.

Для гарантування безпеки персональних даних та ефективного функціонування системи необхідно чітко розмежувати права доступу для різних категорій користувачів. Розглянемо детальніше розподіл цих прав: студенти матимуть доступ виключно до власних даних, а також до сервісів, що стосуються їхнього навчання та проживання. Адміністративний персонал, зрозуміло, отримає повний доступ до всієї інформації та функціональності, що необхідна для адміністрування системи в цілому. Співробітники служби безпеки матимуть право переглядати інформацію про відвідувачів та відповідати за безпеку, однак вони не зможуть вносити зміни до особистих даних студентів. Технічні працівники матимуть доступ до інформації про технічне обслуговування та запитів на ремонт. Батькам буде надано обмежений доступ до інформації, наприклад, до відомостей про успішність та статус проживання їхньої дитини.

Аби кожен користувач міг користуватися системою, необхідно створити обліковий запис, що дозволить увійти до системи з ім'ям користувача та паролем, що забезпечує контроль доступу. Щодо інтерфейсу системи, на нашу думку, він має бути простим, зрозумілим інтуїтивно та пристосованим до вимог кожної категорії користувачів. Система також повинна передбачати для адміністраторів можливість формування різноманітних звітів (стан гуртожитку, технічні проблеми, події, статистика).

Підсумовуючи все сказане раніше, можливо зробити висновок, що параметр "користувач" має надзвичайне значення під час розробки інформаційних систем для студентських містечок. Це пояснюється тим, що він диктує належну організацію доступу, сприяє ефективному використанню наявних ресурсів, забезпечує зручність для студентів та гарантує безпеку в гуртожитках. Ретельне врахування різних категорій користувачів та визначення їх прав є ключем до створення системи, яка буде максимально корисною для кожного учасника.

Наступним суттєвим параметром при розробці інформаційної системи для студентського містечка визначено її «функціональність». Цей параметр

включає в себе сукупність функцій та інструментів, необхідних для надання користувачам можливості ефективно керувати та обслуговувати студентів у гуртожитках, а також забезпечувати зручність та безпеку на території. Система має дозволяти всім користувачам реєструватися, створюючи облікові записи та проходити аутентифікацію для отримання доступу, підтримуючи різні рівні прав. Студенти повинні мати можливість подавати онлайн-заявки на проживання, вказуючи дату заїзду, тип кімнати, особливі побажання та інше. Адміністратори, у свою чергу, зможуть розглядати та обробляти ці заявки, розподіляти місця та змінювати статуси заявок (схвалено/відхилено).

Ще однією функцією, яку має реалізувати система, є автоматизований розподіл місць у гуртожитках. В контексті цієї функціональності система повинна забезпечувати автоматичне управління розподілом кімнат, беручи до уваги індивідуальні потреби студентів (наприклад, потреби іноземних студентів, студентів з обмеженими можливостями). Система також повинна мати можливості для відображення та управління розкладами. Студенти мають можливість зручного доступу до свого розкладу занять та заходів, що стосуються проживання у гуртожитку, а також перегляду інформації про тимчасові обмеження або зміни в розкладі (наприклад, щодо водопостачання чи ремонтних робіт).

Однією з ключових можливостей системи є керування зверненнями щодо сервісу та ремонту. Студенти повинні мати змогу подавати запити на усунення несправностей та технічне обслуговування обладнання у своїх кімнатах (скажімо, проблеми з сантехнікою чи електрикою). Технічний персонал, своєю чергою, матиме змогу отримувати сповіщення про нові запити, опрацьовувати їх та змінювати статус виконання. Крім того, важливою є можливість формування звітів про виконані роботи та стан інвентарю.

Наступною функцією, яку має забезпечувати система, є механізми комунікації (оповіщення та сповіщення). Ця частина функціональності системи має першочергове значення. Система повинна дозволяти обмінюватися сповіщеннями, використовуючи SMS або електронну пошту, про критичні

ситуації (наприклад, порушення правил безпеки або проведення аварійних робіт у гуртожитках).

Ключовою характеристикою роботи системи є управління безпекою. З цією метою система має забезпечувати: реєстрацію відвідувачів, контроль доступу до гуртожитків і перевірку документів; відеоспостереження (за умови наявності такої можливості); керівництво та служби безпеки мусять мати доступ до інформації про інциденти та порушення і реагувати на них у режимі реального часу.

Ще однією ключовою рисою функціонування системи є забезпечення можливостей оплати та керування фінансами. Система повинна передбачати можливість для студентів здійснювати оплату за проживання, комунальні послуги та інші необхідні платежі безпосередньо через систему, формувати фінансові звіти за рахунками та керувати всіма платіжними операціями. Також важливою функцією є відстеження заборгованості та автоматичне надсилання нагадувань про несплачені рахунки.

Крім того, система має передбачати можливість інтеграції з іншими університетськими сервісами. Студенти повинні мати змогу авторизуватися в різних системах університету, використовуючи єдиний набір облікових даних, забезпечуючи зручність та ефективність доступу до необхідних ресурсів.

Отже, функціонал інформаційних систем студентських містечок має бути всеохопним, інтуїтивно зрозумілим та різнобічним задля дієвого керування умовами проживання, розселенням та навчанням студентів у межах кампусу. Система повинна забезпечувати високий рівень автоматизації процесів, заощаджувати час адміністраторів і студентів, гарантувати безпеку та комфорт перебування в гуртожитку.

Наступним виділеним ключовим параметром є «безпека». Безпека – один з найважливіших аспектів розробки інформаційної системи студентського містечка. Це зумовлено тим, що системи обробляють чутливу інформацію, таку як особисті дані студентів, фінансові операції, інформацію про проживання та відвідувачів. Безпека забезпечує захист цієї інформації від несанкціонованого

доступу, втрати, пошкодження та модифікації. Система мусить захищати дані. База даних зобов'язана містити політики безпеки, включаючи регулярні оновлення та виправлення, брандмауери та системи виявлення вторгнень (IDS). Для шифрування даних потрібно використовувати найсучасніші алгоритми шифрування (наприклад, AES-256 для зберігання, TLS для передачі). Також система повинна передбачати створення резервних копій. Необхідно налаштувати регулярне автоматичне резервне копіювання даних для уникнення їх втрати внаслідок технічних збоїв чи атак. Система мусить підтримувати чітку ролеву модель, де кожен користувач має доступ лише до тих даних і функцій, які йому потрібні. Студенти матимуть доступ лише до своїх даних, технічний персонал – лише до технічних запитів, а адміністратори – до всіх даних, включаючи звіти та статистику. Також, варто проводити документування та моніторинг усіх спроб доступу до системи, які повинні реєструватися, щоб мати змогу їх проаналізувати надалі. Це дозволяє виявляти підозрілу активність та запобігати спробам несанкціонованого доступу.

Система має бути міцно захищена від типових загроз для веб-застосунків, як-от SQL-ін'єкції, міжсайтовий скриптинг (XSS) та фальсифікація міжсайтових запитів (CSRF). Це передбачає застосування перевірених технік роботи з інформацією, що надходить. Необхідне використання надійних способів обробки вхідних даних, наприклад, параметризованих SQL-запитів, фільтрації та верифікації. На рівні мережі також слід подбати про безпеку, використовуючи захищені протоколи передачі даних (наприклад, HTTPS) та VPN для доступу до систем з віддалених точок. Запобігання DDoS-атакам потребує відповідних заходів, таких як захист на рівні інфраструктури та сервіси фільтрації трафіку для захисту від розподілених атак на відмову в обслуговуванні (DDoS).

Для гарантування належної безпеки, необхідно також подбати про фізичний захист серверів та інших критичних складових інфраструктури від несанкціонованого доступу. Це передбачає використання спеціалізованих приміщень з контролем доступу, систем відеоспостереження та обмеження

доступу виключно уповноваженим особам. Щоб мінімізувати ризики, пов'язані з вразливостями, якими можуть скористатися хакери, слід регулярно оновлювати операційні системи, серверне програмне забезпечення, бібліотеки та інші компоненти.

Система повинна бути сумісною з вимогами законодавства щодо захисту персональних даних, зокрема з положеннями Загального регламенту про захист даних Європейського Союзу (GDPR). Відповідно до цих вимог, слід також забезпечити належне зберігання персональних даних. Інформація про студентів повинна зберігатися лише протягом часу, необхідного для виконання функцій системи, і повинна бути зашифрована на всіх етапах зберігання та передавання.

На наш погляд, щоб гарантувати конфіденційність даних і загалом підвищити рівень безпеки, надзвичайно важливо проводити навчання користувачів та співробітників щодо роботи в цій системі. Усі користувачі, включаючи студентів, адміністративний персонал та технічний персонал, повинні пройти спеціальну підготовку з питань безпеки, яка включає базові знання з кібергігієни (наприклад, правильне створення паролів, уникнення фішингових атак).

Таким чином, безпека стає критичним фактором під час розробки інформаційних систем для студентських містечок, оскільки ці системи працюють з особистою інформацією студентів, фінансовою документацією, технічними специфікаціями та питаннями безпеки проживання. Ретельне планування та впровадження механізмів захисту на кожному етапі (від фізичної безпеки до захисту даних і мережі) може запобігти зловживанням, забезпечити конфіденційність та гарантувати захист від як зовнішніх, так і внутрішніх загроз.

Наступний аспект, на який слід звернути увагу під час планування інформаційної системи, – це «інтеграція». В контексті розробки інформаційних систем для студентських містечок інтеграція означає можливість системи результативно взаємодіяти з іншими інформаційними системами, платформами та технологіями, які використовуються як в межах, так і за межами

студентського містечка. Завдяки інтеграції забезпечується збір, обробка та обмін даними між різними системами, що сприяє оптимізації управління та покращенню досвіду користувачів.

Інтеграція дозволяє уникнути дублювання даних, підвищує рівень автоматизації процесів, спрощує адміністрування, мінімізує ризик помилок та робить використання системи зручнішим для студентів та співробітників.

Ось ключові вектори інтеграції в інформаційних системах студмістечок: взаємодія з університетськими інформаційними платформами; взаємодія з фінансовими та платіжними структурами; взаємодія з системами контролю доступу та безпеки; взаємодія з системами зв'язку та оповіщення; взаємодія з бібліотечними, культурними та іншими системами.

Інтеграція з платформами дистанційного навчання, як, наприклад, Moodle, відкриває для студентів зручний доступ до розкладів занять, звітів успішності та іншої академічної інформації, без потреби переходити між різними системами. Студенти отримують змогу автоматично реєструватися на курси, тренінги, культурні та спортивні заходи, організовані університетом чи адміністрацією студмістечка. Інтеграція з цією системою забезпечує синхронізацію даних між різними платформами.

Інтегрування з платіжними платформами (такими як Приват24, PayPal, банківські картки) дає змогу студентам здійснювати оплату за проживання, комунальні послуги, штрафи та інші виплати напряму з інтерфейсу системи студмістечка. Система повинна мати інтеграцію з бухгалтерською системою університету для автоматичного формування платіжних документів, нарахування оплати за проживання, отримання та керування фінансовими транзакціями.

Щодо систем контролю доступу та безпеки, то ця система має інтегруватися з системами контролю доступу в будівлі та кімнати гуртожитків, дозволяючи студентам отримувати доступ до приміщень, та реєструвати відвідувачів за допомогою електронних студентських квитків або карток.

Наявність студентського квитка відкриває перед студентом двері до численних можливостей. Його можна використовувати як перепустку до гуртожитку, бібліотеки та їдальні, а також для здійснення платежів у студентському містечку.

Електронна пошта відіграє ключову роль, автоматично надсилаючи важливі повідомлення: підтвердження поселення, нагадування про оплату, інформація про наявність місць в гуртожитку та інші.

Інтеграція з бібліотечною системою університету дає змогу студентам користуватися книгами й електронними ресурсами, використовуючи свій обліковий запис. Приміром, функція автоматичного блокування доступу до гуртожитків, якщо студент має заборгованість перед бібліотекою.

Ця система має забезпечувати інтеграцію з платформою для реєстрації на культурні, спортивні або академічні події, що дозволить студентам реєструватися на заходи, організовані університетом або студмістечком, та автоматично отримувати повідомлення про зміни або нові події.

З наведеного вище розуміємо, що інтеграція — ключовий елемент при проектуванні інформаційної системи студентського містечка. Вона гарантує злагоджену та продуктивну взаємодію різноманітних платформ. Інтеграція університетських систем із зовнішніми сервісами веде до автоматизації процесів, знижує дублювання інформації та зменшує обсяг ручної роботи, що в підсумку підвищує ефективність та комфорт для студентів і адміністрації.

Фактор «мобільності» в розробці інформаційних систем для студентських містечок полягає в гарантуванні зручного доступу до функціоналу системи та даних з мобільних пристроїв – смартфонів, планшетів та подібних. Це передбачає адаптивний дизайн, розробку мобільних додатків та оптимізацію процесів для забезпечення доступу до системи для студентів, адміністраторів та інших користувачів будь-коли та з будь-якого місця.

Отже, мобільність визначено як ключовий фактор. Це зумовлено тим, що студенти й персонал здебільшого пересуваються та потребують оперативного доступу до даних, зокрема зі смартфонів. Завдяки цьому параметру системи,

студенти матимуть змогу подавати заявки на проживання в гуртожитках, переглядати наявні кімнати, здійснювати оплату проживання та звертатися по обслуговування, не контактуючи з адміністрацією, використовуючи мобільний додаток чи адаптивну веб-версію. Вони зможуть отримувати доступ до персональної інформації, відстежувати стан заявки на поселення, перевіряти розклад занять, залишки платежів і відгуки про сервіси гуртожитку. Крім того, вони матимуть змогу подавати заявки на ремонт у кімнаті чи повідомляти про неполадки (наприклад, проблеми з сантехнікою чи електрикою).

Створення кастомних мобільних додатків під iOS та Android платформи дасть змогу користувачам пересилати сповіщення, записуватися на заходи, переглядати розклад і користуватися фінансовими відомостями. Якщо розробка мобільного додатку не передбачена, можлива розробка адаптивного веб-сайту, котрий правильно показуватиме контент на різноманітних пристроях (смартфони, планшети). Веб-ресурс повинен бути зручним для користування з мобільних пристроїв та вміщати всі ключові функції системи. За допомогою зазначених програмних рішень, система зможе надсилати сповіщення на мобільні телефони або електронну пошту, повідомляючи користувачів про оплати, оновлення правил проживання, дедлайни реєстрації тощо. Критично важливо забезпечити безпеку доступу до мобільних додатків та веб-версії системи, застосовуючи багатофакторну аутентифікацію задля запобігання несанкціонованому доступу.

Для підвищення комфорту використання мобільні додатки можна поєднати з рештою університетських сервісів. Йдеться про електронні журнали успішності, освітні онлайн-платформи, а також системи електронного доступу до книгозбірень, їдалень тощо. Таким чином, студенти зможуть користуватися всіма ресурсами університету через один мобільний додаток.

Мобільні додатки варто розробляти з орієнтацією на зручність та інтуїтивне розуміння інтерфейсу користувачами на різноманітних пристроях. Необхідно адаптувати дизайн під маленькі дисплеї смартфонів та планшетних комп'ютерів.

Насамкінець щодо цього показника системи, варто відзначити його ключову роль у процесі розробки інформаційних систем студентського містечка. Це зумовлено тим, що мобільність гарантує студентам, співробітникам адміністрації та іншим працівникам простий та комфортний доступ до всіх сервісів і функціоналу, незалежно від часу та розташування – чи то територія студмістечка, чи будь-яке інше місце. Зручний мобільний доступ сприяє підвищенню адміністративної ефективності, покращує взаємодію між студентами та адміністрацією, а також дозволяє зменшити витрати часу на виконання щоденних операцій. Мобільні сервіси також забезпечують більшу доступність і сучасність системи. Це має велике значення для студентів, для яких цифрові технології стали невід'ємною частиною повсякденного життя.

«Аналіз даних» при проектуванні інформаційної системи студентського містечка — це критично важливий елемент, який передбачає збирання, опрацювання, тлумачення та впорядкування даних для розробки продуктивної та комфортної для використання системи. Цей елемент містить кілька ключових складових: акумулювання даних; дослідження потреб користувачів; обробка та розподіл даних на частини; аналіз та візуалізація даних; виявлення закономірностей та передбачення; безпека та захист інформації.

Отже, аналіз даних — це необхідний складник у процесі проєктування, що забезпечує розробку систем, які відповідають дійсним потребам користувачів та сприяють ефективному управлінню студентським містечком.

Параметр "технічна підтримка" під час проектування інформаційних систем для студентського містечка являє собою набір дій та процедур, котрі скеровані на підтримання безперебійної працездатності системи та негайне усунення технічних неполадок, з якими можна зіткнутися в процесі експлуатації. Значення технічної підтримки полягає у гарантуванні користувачам (студентам, персоналу та адміністрації) постійного доступу до системи та надання своєчасної допомоги при виникненні будь-яких проблем.

Цей параметр передбачає формування команди технічної підтримки, котра буде опікуватися працездатністю системи, або ж залучення сторонніх

спеціалістів. Це включає операторів підтримки, що реагують на звернення користувачів, системних адміністраторів та розробників, котрі розв'язують складні технічні задачі. Щоб система функціонувала належним чином, слід підготувати інструкції для користувачів, відповіді на типові питання та перелік найпоширеніших запитань. Це дасть можливість користувачам самостійно розв'язувати прості проблеми, не залучаючи підтримку. Таким чином, буде знижено навантаження на технічну підтримку та підвищено зручність використання системи.

Необхідно також гарантувати регулярне оновлення програмного забезпечення, виправлення помилок та оптимізацію системи задля підвищення стійкості та продуктивності роботи системи. До цього належать також оновлення безпеки для захисту ваших даних від потенційних загроз.

Для виявлення типових проблем та загального вдосконалення роботи системи, потрібно збирати відгуки користувачів та аналізувати звернення до технічної підтримки. Це може допомогти створити більш ефективні алгоритми підтримки та зробити взаємодію з системою максимально зрозумілою та зручною для користувачів. Варто також розробити план дій для відновлення функціональності системи у разі серйозного збою або можливої втрати даних. Це передбачає регулярне резервне копіювання інформації та проведення тестувань системи відновлення.

Належне технічне забезпечення гарантує, що інформаційні системи студентського містечка працюють продуктивно, оперативно вирішують проблеми та забезпечують позитивний користувацький досвід.

Наступний параметр, визнаний критичним, - «гнучкість та масштабованість». Система має легко піддаватися змінам та адаптуватися до нових потреб, уникаючи масштабної реорганізації. Це передбачає, що адміністратори та користувачі зможуть додавати або модифікувати функціонал, наприклад, інтегрувати нові модулі для управління послугами, скажімо, онлайн-бронювання житла або інтеграцію з навчальними платформами університету. За умови модульної архітектури, функції системи розбиваються

на незалежні блоки (модулі), які можна додавати або вилучати окремо. Приміром, система може включати модулі для управління розміщенням, платіжних операцій, контролю доступу та інше, функціонуючи як самостійні компоненти. Це значно спрощує додавання нових функцій, оновлення та тестування системи.

Масштабованість інфраструктури означає здатність збільшувати систему, не втрачаючи продуктивності, шляхом додавання серверного обладнання або розширення бази користувачів. Наприклад, система має бути здатною ефективно функціонувати навіть при збільшенні кількості користувачів та обсягів даних, як-от при збільшенні числа студентів у студентському містечку чи його розширенні.

Система має передбачати створення профілів користувачів з різними рівнями допуску, скажімо, для адміністраторів, студентів і працівників безпеки. Це забезпечить легке налаштування та обмеження доступу до різних розділів системи з урахуванням функцій користувача. Розробникам цієї системи слід передбачити можливість автоматичного розширення функціоналу та додавання нових сервісів, спираючись на наявну інфраструктуру. Наприклад, система зможе автоматично розширюватися у відповідь на збільшення навантаження або автоматично оновлювати модулі, не створюючи простоїв для користувачів.

Дана система мусить мати адаптивний інтерфейс, що означає підтримку різних платформ, зокрема мобільних пристроїв та стаціонарних комп'ютерів. Система повинна бездоганно працювати як на мобільних додатках, так і у веб-інтерфейсах, даючи користувачам змогу отримати доступ до критично важливої інформації у будь-який момент.

Забезпечення гнучкості й масштабованості інформаційних систем для студмістечок гарантує бездоганну працездатність систем навіть при збільшенні кількості користувачів, а також їхню легку адаптацію до нових потреб, що з’являються з плином часу. Це робить інвестиції у подібні системи більш вигідними та зручними в управлінні на довгострокову перспективу.

«Технічна підтримка» у процесі проектування інформаційних систем студмістечка є критично важливим компонентом для безперебійної роботи системи й оперативного розв’язання технічних проблем. Вона сприяє збереженню стабільності системи, легкості використання та позитивного досвіду для всіх користувачів, зокрема студентів, співробітників та адміністрації. Для забезпечення належного функціонування запроектованої системи, потрібно створити або залучити команду кваліфікованих фахівців, відповідальних за обслуговування та розвиток системи. До них можуть належати оператори, які приймають запити від користувачів, системні адміністратори, що підтримують інфраструктуру, та розробники, які усувають технічні проблеми чи додають нові функції.

Для коректної функціональності системи, вона повинна містити розділ з корисною інформацією, що допомагає користувачам (FAQ, інструкції, керівництва, відеоматеріали). Це дозволить їм вирішувати типові труднощі самостійно, без потреби звертатися до служби підтримки. Такий підхід знижує навантаження на техпідтримку, а також прискорює отримання відповідей на поширені питання користувачами.

Система повинна бути зорієнтована на отримання зворотного зв'язку й постійне вдосконалення. Для цього важливо організувати збір коментарів та оцінок від користувачів щодо якості підтримки, а також аналізувати найбільш поширені звернення. Це сприятиме покращенню системи, вирішенню проблем, що повторюються, та загалом позитивно впливатиме на взаємодію з користувачами.

Крім того, служба технічної підтримки відповідає за гарантування безпеки особистих даних користувачів, що включає в себе суворе дотримання відповідних стандартів. Це передбачає використання захищених каналів передачі інформації, дотримання політики конфіденційності та систематичне оновлення систем безпеки.

Загалом, технічна підтримка визначає стабільність функціонування інформаційної системи студмістечка. Вона гарантує безперебійну доступність

сервісів, оперативне вирішення проблем та надійний захист для всіх, хто користується системою.

"Зворотний зв'язок" – ключовий елемент у розробці інформаційних систем для студмістечка. Відгуки користувачів, включаючи студентів, адміністрацію і технічний персонал, є надзвичайно важливими для удосконалення функцій системи, виявлення наявних недоліків та потреб. Якісний зворотний зв'язок сприяє створенню корисних та ефективних систем.

В систему потрібно інтегрувати інструменти для отримання відгуків, наприклад, форми, анкети та вікна для коментарів. Вони можуть бути складовою самої системи, частинами веб-сайту або окремими мобільними додатками для зворотного зв'язку. Варто також розглянути можливість надання різних шляхів для комунікації з керівництвом та технічною підтримкою, включаючи електронну пошту, онлайн-чат, телефонні лінії або спеціалізовані онлайн-портали. Забезпечення різноманітності каналів спростить процес отримання зворотного зв'язку від користувачів з різними потребами та технічними знаннями.

Для найефективнішої роботи системи зворотного зв'язку необхідно систематично проводити опитування для оцінки різних аспектів: задоволеності системою загалом, її функціональністю, швидкістю реакції та якістю технічної підтримки. Це дасть змогу виявити сильні сторони та сфери, які вимагають покращення. Не менш важливо правильно організувати та класифікувати отримані відгуки, наприклад, групуючи їх за типом проблеми (технічні збої, обмеження функціональності, зручність інтерфейсу) та пріоритетністю. Такий підхід дозволить виділити найбільш актуальні проблеми та визначити пріоритетні напрями для роботи. Швидке реагування на відгуки сприяє підтримці довіри користувачів до системи та забезпечує позитивне враження від взаємодії з нею. Зворотний зв'язок, навіть якщо він анонімний, має особливе значення, адже він дозволяє зібрати критичні зауваження. Гарантування конфіденційності дає можливість користувачам вільно висловлювати свої думки та надавати реалістичні пропозиції щодо вдосконалення.

Зворотний зв'язок під час розробки систем для студентського містечка – це ключовий інструмент для формування комфортного для користувачів простору, гарантуючи відповідність системи їхнім потребам та очікуванням. Це не просто покращує якість надаваних послуг, а й зміцнює довіру користувачів, що веде до ефективної функціональності студмістечка.

Останній з обраних параметрів, але точно не останній за значенням - це «доступність». Доступність виступає ключовим аспектом при розробці інформаційних систем для студентського містечка, прагнучи зробити їх максимально зручними та зрозумілими для всіх користувачів, незалежно від їхніх особливостей, пристроїв чи умов використання. Високий рівень доступності сприяє ефективній взаємодії між студентами, викладачами та адміністрацією, забезпечуючи рівний доступ до інформації та послуг.

Проєктування цієї інформаційної системи зобов'язує до розробки інтерфейсів, що максимально враховують потреби користувачів з обмеженими можливостями. Йдеться про інтеграцію програмних компонентів, що полегшують взаємодію з системою для людей з порушеннями зору та слуху. Це, зокрема, стосується реалізації підтримки екранних зчитувачів, текстових редакторів та налаштувань субтитрів для відеоконтенту. При цьому, система має відповідати міжнародним нормам доступності, а саме – вимогам Керівних принципів доступності веб-контенту (WCAG). Необхідно передбачити коректне використання кольорової гами, забезпечення текстових альтернатив для зображень, чітку структуру контенту та зручну клавіатурну навігацію.

Оскільки університети все частіше інтегрують міжнародну мобільність, ця система потребує багатомовної підтримки, щоб зробити її доступною для іноземних студентів та співробітників. Не слід забувати, що у майбутньому до системи матимуть доступ з різних пристроїв, тому важливо забезпечити адаптивний дизайн. Це забезпечить зручність використання системи незалежно від типу пристрою, яким користується людина.

Система має бути розроблена так, щоб функціонувати без збоїв у випадках технічних проблем або великих навантажень. Це передбачає

використання резервних серверів, регулярне резервне копіювання даних та швидке відновлення після збоїв. Також необхідно забезпечити цілодобову доступність системи, 7 днів на тиждень, щоб ключові функції, наприклад, бронювання кімнат, перегляд розкладу та оплата послуг, були завжди доступні. Це особливо актуально в університетському середовищі, де робочі графіки користувачів можуть бути різними.

Забезпечення доступності інформаційних систем кампусу є ключем до забезпечення зручності використання для всіх користувачів, незалежно від їхніх фізичних можливостей або технічних потреб. Це створює рівні можливості для всіх студентів та співробітників, покращує загальне враження від користування та сприяє ефективній роботі системи загалом.

Інформація у сучасному світі набула статусу одного з найцінніших ресурсів, а інформаційна система (ІС) перетворилася на незамінний інструмент практично в усіх сферах діяльності. Різноманіття задач, що вирішуються за допомогою ІС, зумовило появу великої кількості різних типів систем, що відрізняються принципами побудови та закладеними правилами обробки інформації [60].

Зростання інформаційного наповнення, комплексність питань, що стоять перед керівниками підприємств та організацій, потреба у врахуванні численних взаємозалежних аспектів, здатність адаптуватися до динамічних змін – все це ставить гостру необхідність у формуванні відповідного інформаційного простору [52].

Сучасні компанії та установи функціонують в середовищі постійного інформаційного потоку, вимагаючи швидкого аналізу та ухвалення рішень на його основі. Обчислювальна техніка та інформаційні технології (ІТ) демонструють стрімкий розвиток. На сьогодні успіх та прибутковість підприємств визначаються, серед іншого, рівнем розвитку інформаційних технологій, швидкістю та якістю обробки інформації, а також адекватністю та раціональністю прийняття рішень.

Для узгодження всіх зусиль у стратегічному розвитку компанії та її інформаційних систем, поширеною та суттєвою помилкою, якої припускаються не лише ІТ-спеціалісти, але й керівництво найвищого рівня, є підхід менеджерів, які, впровадивши колись інформаційні системи (ІС), припиняють подальшу роботу. Через це процес проектування ІС стає обов'язковим елементом. Якщо компанія вдається до цього процесу не вперше, то термін "проектування" фактично рівнозначний поняттю "розробка ІС". Це пояснює бурхливий прогрес у технологіях проектування ІС за останні роки. Перш за все, це стосується розвитку CASE-технологій, які істотно скорочують час проектування ІС, сприяють організації спільної роботи команди, дозволяють оперативно вносити зміни та швидко реагувати на мінливі обставини [53, 54].

Окрему нішу утворюють сучасні ІТ – автоматизація процесів життєдіяльності освітніх установ, взаємодія з учнями та керівництвом вишів. У цьому контексті проєктування та розробка ІС потребують знання основоположних методологій та програмних інструментів, що дають змогу керувати цими освітніми процесами максимально швидко та без похибок.

Рівень професійного навчання фахівців вимагає удосконалення цифрового освітнього простору, який має можливість забезпечити ефективне формування цифрових і дослідницьких навичок майбутніх професіоналів. Одним з ключових елементів програмно-методичного забезпечення середовища сучасних університетів є створення та застосування інформаційних систем, націлених на професійну підготовку фахівців.

З аналізу наявної літератури, що вище, можна виокремити ключовий висновок: дослідження не містять достатньо інформації про специфіку конструювання інформаційних систем, зокрема, про параметри, що впливають на їх проектування. Вирішення цієї проблеми, особливо в епоху постійного зростання обсягів інформації та збільшення вимог, має надзвичайне значення при розробці інформаційних систем. Це здатне суттєво прискорити процес реалізації проектів, заощаджуючи час розробників.

2.2. Синтез моделі факторів впливу на функціонування інформаційної технології

Для синтезу ієрархічної моделі акторів проектування інформаційної системи для студмістечка застосовано апарат теорії графів. Ця методика дозволяє моделювати взаємозв'язки та взаємовпливи між об'єктами, що робить її придатною для вирішення поставленої задачі. Обраний метод добре себе зарекомендував та використовується для розрахунків у випадках, коли параметри системи не можуть бути виражені математично.

Першим кроком є побудова графа зв'язків. Графова структура дає змогу наочно представити досліджувані системи, що містять велику кількість взаємодій та взаємозалежностей. Вузли графа відповідають об'єктам (параметрам), а ребра – відображають вплив та залежність між ними. Для практичного здійснення розрахунків нами було організовано опитування, в результаті якого визначено ряд ключових параметрів, що впливають на процес проектування інформаційної системи управління освітнім процесом. Експертним шляхом також встановлено вплив кожного параметра на інші, що дозволяє сформувати граф зв'язків цих параметрів.

g_1 – Користувачі (К);

g_2 – Функціональність (Ф);

g_3 – Безпека (Б);

g_4 – Інтеграція (І);

g_5 – Мобільність (М);

g_6 – Аналіз даних (АД);

g_7 – Технічна підтримка (ТП);

g_8 – Гнучкість і масштабованість (ГМ);

g_9 – Зворотний зв'язок (ЗЗ);

g_{10} – Забезпечення доступності (ЗД).

Досліджувана множина параметрів проектування інформаційної системи для студмістечка приймемо$i=\{1, 2, 3…n\}$ представлена у вигляді орієнтованого графу (рис.2.1).

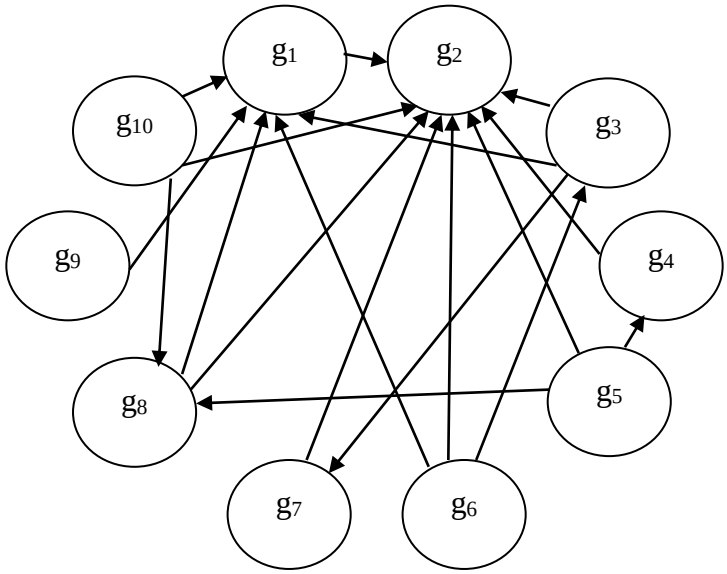


Рис. 2.1. Вихідний граф зв’язків між факторами проектування інформаційної системи для студмістечка

Наступним етапом в теорії графів є побудова бінарної матриці залежності ***B***, яка використовується для відображення всіх залежностей між факторами. Кожен фактор може приймати значення 0, коли відсутня залежність між двома об’єктами або 1, коли присутня залежність між двома досліджуваними факторами [56]:

$$b_{ij} = \begin{cases} 0, \text{ якщо } i \text{ незалежить від } j \\ 1, \text{ якщо } i \text{ залежить від } j \end{cases} \quad (1)$$

Результат побудови подано у таблиці 2.1.

Таблиця 2.1

Бінарна матриця залежності

		1	2	3	4	5	6	7	8	9	10
		К	**Ф**	**Б**	**І**	**М**	**АД**	**ТП**	**ГМ**	**ЗЗ**	**ЗД**
1	**К**	0	1	0	0	0	0	0	0	0	0
2	**Ф**	0	0	0	0	0	0	0	0	0	0
3	**Б**	1	1	0	0	0	0	1	0	0	0
4	**І**	0	1	0	0	0	0	0	0	0	0
5	**М**	0	1	0	1	0	0	0	1	0	0

6	**АД**	1	1	1	0	0	0	0	0	0	0
7	**ТП**	0	1	0	0	0	0	0	0	0	0
8	**ГМ**	1	1	0	0	0	0	0	0	0	0
9	**ЗЗ**	1	0	0	0	0	0	0	0	0	0
10	**ЗД**	1	1	0	0	0	0	0	1	0	0

Наступний етап визначає формування матриці досяжності ***D***, яка показує, чи існує шлях між будь-якими двома об’єктами. Її побудова зводиться до розв’язання даного рівняння [57]:

$$(I + B)^{n-1} \leq (I + B)^n = (I + B)^{n+1}, \quad (2)$$

де ***I*** – одинична матриця, яку піднесено до ступеня n, щоб виконувалась умова:

$$d_{ij} = \begin{cases} 1, \text{ якщо з вершини } i \text{ можна попасти в } j \\ 0, \text{ в іншому випадку} \end{cases}. \quad (3)$$

Матриця досяжності набуде наступного вигляду (таблиця 2.2):

Таблиця 2.2

Матриця досяжності

		1	2	3	4	5	6	7	8	9	10
		К	**Ф**	**Б**	**І**	**М**	**АД**	**ТП**	**ГМ**	**ЗЗ**	**ЗД**
1	**К**	1	1	0	0	0	0	0	0	0	0
2	**Ф**	0	1	0	0	0	0	0	0	0	0
3	**Б**	1	1	1	0	0	0	1	0	0	0
4	**І**	0	1	0	1	0	0	0	0	0	0
5	**М**	1	1	0	1	1	0	0	1	0	0
6	**АД**	1	1	1	0	0	1	1	0	0	0
7	**ТП**	0	1	0	0	0	0	1	0	0	0
8	**ГМ**	1	1	0	0	0	0	0	1	0	0
9	**ЗЗ**	1	1	0	0	0	0	0	0	1	0
10	**ЗД**	1	1	0	0	0	0	0	1	0	1

Дана методологія показує, що вершина орієнтованого графу *j* досягається з вершини *i* якщо існує шлях від вершини *i* до *j*. Тому, ця вершина називається досягнутою. Їх підмножину $R(g_i)$ визначають елементи рядків матриці. Аналогічно, вершина *i* є попередницею вершини *j*, якщо вона досягається з цієї вершини. Цю підмножину $B(g_i)$ вершин попередниць визначають стовпці матриці досяжності. На перетині множин вершин $B(g_i)=R(g_i)\cap B(g_i)$ буде отримано певний рівень ієрархії пріоритетної дії факторів [58].

Отже, практична побудова ітераційних циклів досліджуваної системи зводиться до формування таблиці 2.3, яка містить:

- підмножину $R(g_i)$ – елементи *i*-го рядка матриці досяжності;
- підмножину $B(g_i)$ – елементи *i*-го стовпця матриці досяжності;
- підмножину $R(g_i)\cap B(g_i)$, яка формується як логічний перетин елементів підмножин $R(g_i)$ і $B(g_i)$.

Таблиця 2. 3

Визначення першої ітерації

k_i	$R(g_i)$	$B(g_i)$	$R(g_i)\cap B(g_i)$
1	1, 2	1, 3, 5, 6, 8, 9 , 10	1
2	2	1, 2, 3, 4, 5, 6, 7, 8, 9, 10	2
3	1, 2, 3, 7	3, 6	3
4	2, 4	4, 5	4
5	1, 2, 4, 5, 8	5	5
6	1, 2, 3, 6, 7	6	6
7	2, 7	3, 6, 7	7
8	1, 2, 8	5, 8, 10	8
9	1,2, 9	9	9
10	1, 2, 8, 10	10	10

В першій ітерації рівність $B(g_i)=R(g_i)\cap B(g_i)$ в таблиці 3 виконується для елементів 5, 6, 9 і 10. Ними є фактори: мобільність (М); аналіз даних (АД);

зворотний зв'язок (ЗЗ); забезпечення доступності (ЗД). Вони визначають перший (найнижчий) рівень в ієрархічній моделі факторів проектування інформаційної системи для студмістечка.

Друга ітерація формується наступним чином: викидаємо з таблиці 3 рядки з номерами 5, 6, 9 та 10, а в другому стовпці викреслюємо цифри 5, 6, 9 та 10 (табл. 2.4).

Таблиця 2.4

Визначення другої ітерації

ki	***R(g_i)***	***B(g_i)***	***R(g_i)∩B(g_i)***
1	1, 2	1, 3, 8	1
2	2	1, 2, 3, 4, 7, 8	2
3	1, 2, 3, 7	3	3
4	2, 4	4	4
7	2, 7	3, 7	7
8	1, 2, 8	8	8

У таблиці 2.4 рівність $B(g_i)=R(g_i)\cap B(g_i)$ виконується для елементів під номерами 3, 4 та 8. Це: безпека (Б); інтеграція (І); гнучкість і масштабованість (ГМ), які займають другий рівень в ієрархії. Згідно описаної вище методології формуємо наступні ітерації (табл. 2.5) [59].

Таблиця 2.5

Визначення третьої та четвертої ітерації

ki	***R(g_i)***	***B(g_i)***	***R(g_i)∩B(g_i)***
1	1, 2	1	1
2	2	1, 2, 7	2
7	2, 7	7	7

Отже, отримані ітераційні рівні представляють групу параметрів, які підпорядковані або пов'язані з вищим рівнем. Досліджувані об’єкти згідно

теорії графів уможливили побудову ієрархічної моделі параметрів проектування інформаційної системи для студмістечка. Вона наочно показує структуру, в якій об’єкти розташовані за рівнями. Відповідно, кожен формується залежно від важливості, складності або взаємозв'язків між досліджуваними параметрами (рис. 2.2) [60].

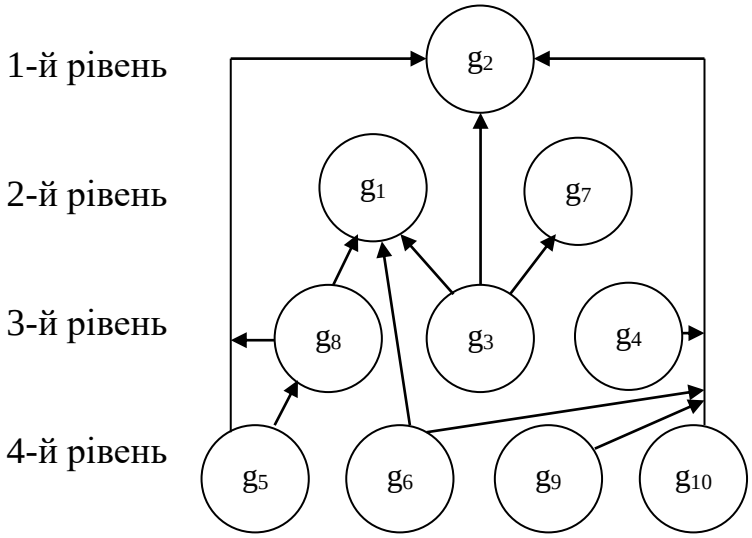


Рис. 2.2. Ієрархічна модель параметрів впливу на процес проектування інформаційної системи для студмістечка

Розробка інформаційної системи для студмістечка є надзвичайно актуальною з кількох причин. Насамперед, це зростання чисельності студентів. Оскільки зі збільшенням кількості студентів у навчальних закладах виникає потреба в ефективному управлінні даними та ресурсами. Наступне завдання яке повинна виконувати дана система це покращення комунікації між усіма учасниками освітнього процесу. Автоматизація рутинних завдань, таких як реєстрація на курси, управління гуртожитками та облік оцінок, дозволяє зекономити час і зусилля. Важливим завданням системи є також доступність до інформації. Завдяки інформаційній системі студенти зможуть легко отримувати актуальну інформацію про розклад, оцінки, події та інші важливі аспекти життя студмістечка. Відповідно, збір та аналіз даних дозволить адміністрації приймати обґрунтовані рішення, поліпшувати якість освіти та управління. Дана система повинна забезпечити та легше адаптуватись до нових викликів. У

сучасному світі, зокрема в умовах пандемії, дистанційне навчання та гібридні моделі потребують нових підходів у організації навчального процесу. Запропонована система повинна допомогти в управлінні безпекою, відслідковуючи доступ до територій, записуючи події та забезпечуючи моніторинг. Інформаційні системи зможуть сприяти розвитку студентських ініціатив, волонтерства та участі в заходах, що покращує загальну атмосферу в студмістечку [55].

Даний граф (рис. 2.1) дозволяє наочно візуалізувати досліджувані системи. Для цього дослідження буде використано метод попарного порівняння [55]. Його методологія полягає у порівнянні досліджуваних елементів в парах, тобто визначення переваги одного критерію над іншим.

На першому етапі, при використанні даної методики, створюється квадратна матриця попарних порівнянь G=(g_{ij}). Вона формується за результатами оцінок експертів, тобто кожен експерт приймає рішення щодо множини факторів G={g_1, g_2, g_3…, g_n}. Важливою умовою при цьому є: врахувати наступне, коли фактор g_i має однакову відносну важливість з g_j, тодіg_{ij}=1, g_{ji}=1, а матриця Gотримає вигляд [55]:

$$G=\begin{bmatrix} 1 & g_{12} & \dots & g_{1n} \\ 1/g_{12} & 1 & \dots & g_{2n} \\ \dots & \dots & \dots & \dots \\ 1/g_{1n} & 1/g_{2n} & \dots & 1 \end{bmatrix} \qquad (1)$$

Кожному експерту, який здійснює оцінювання факторів запропоновано шкалу відносної важливості критеріїв за Сааті [57]. В її основі лежить шкала значень від 1 до 9:

1 – об’єкти рівноцінні;

3 – один об’єкт дещо переважає інший;

5 – один об’єкт переважає інший;

7 – один об’єкт значно переважає інший;

9 – один об’єкт абсолютно переважає інший;

2, 4, 6, 8 – компромісні проміжні значення.

На основі даної методики отримана квадратна матриця G попарних порівнянь (таблиця 2.6):

Таблиця 2.6

Матриця попарних порівнянь ієрархічної моделі факторів проектування інформаційної системи для студмістечка

	К	Ф	Б	І	М	АД	ТП	ГМ	ЗЗ	ЗД
К	1	1/3	3	3	5	5	2	3	5	5
Ф	3	1	5	5	9	9	3	5	9	9
Б	1/3	1/5	1	2	3	3	1/3	2	3	3
І	1/3	1/5	1/2	1	3	3	1/3	2	3	3
М	1/5	1/9	1/3	1/3	1	2	1/7	1/3	2	1/2
АД	1/5	1/9	1/3	1/3	1/2	1	1/7	1/3	2	2
ТП	1/2	1/3	3	3	7	7	1	3	5	5
ГМ	1/3	1/5	1/2	1/2	3	3	1/3	1	3	3
ЗЗ	1/5	1/9	1/3	1/3	1/2	1/2	1/5	1/3	1	2
ЗД	1/5	1/9	1/3	1/3	2	1/2	1/5	1/3	1/2	1

Наступний етап оптимізації ієрархічної моделі включає нормалізацію матриці та обчислення ваг факторів, тобто визначення вектору пріоритетів. Для цього, присвоєно множині факторів $G=\{g_1, g_2, g_3\ldots, g_n\}$числову вагу $W=\{w_1, w_2, w_3\ldots, w_n\}$. Вона відповідає експертним порівнянням факторів. Таким чином, всі елементи матриці $G=(g_{ij})$ будутьобчислюватись за правилом:

$$G=\begin{bmatrix} w_{1/w_1} w_{1/w_2} \ldots w_{1/w_n} \\ w_{2/w_1} w_{2/w_2} \ldots w_{2/wn} \\ \ldots \ldots \ldots \ldots \\ w_{n/w_1} w_{n/w_2} \ldots w_{n/w_n} \end{bmatrix} \quad (2)$$

На практиці, всі отримані значення в матриці нормалізуються діленням кожного фактору на суму значень для кожного стовпця матриці. Відносну вагу кожного елементу матриці визначає середнє значення кожного рядка нормалізованої матриці:

$$V_i = \frac{\sqrt[n]{\left(\Pi_{j=1}^{n} g_{ij}\right)}}{\sum_{i=1}^{n}\left(\sqrt[n]{\left(\Pi_{j=1}^{n} g_{ij}\right)}\right)} \quad (3)$$

Отримано значення вектору пріоритетів:

V_n = (0,173; 0,332; 0,083; 0,072; 0,029; 0,029; 0,161; 0,063; 0,026; 0,026).

Для кращого візуального представлення компоненти вектору помножимо на коефіцієнт k=1000.

Отримано вектор: V_n = (173; 332; 83; 72; 29; 29; 161; 63; 26; 26)

Для обчислення узгодженості в експертних рішеннях вагових значень факторів виконано множення вектору пріоритетів (V_n)на матрицю попарних порівнянь:

V_{n1}= (1,829; 3,449; 0,871; 0,757; 0,323; 0,319; 1,698; 0,657; 0,296; 0,291).

Наступним кроком є перевірка узгодженості експертних рішень.

Для визначення коректності та згоди в оцінках експертами факторів доцільно використовувати відхилення величини максимального значення λ_{max} від порядку матриці n.

Приблизне значення λ_{max} обчислюємо:

$$\lambda_{max} = \sum_{j=1}^{n} G_j V_j, \quad (4)$$

де $G_j = \sum_{i=1}^{n} g_{ij}$ - сума елементів i-стовпця матриці; V_j – вектор пріоритетів.

Значення λ_{max}= 10,6. V_{n2} = (10,52; 10,37; 10,45; 10,43; 10,9; 10,76; 10,49; 10,41; 10,74; 10,9). Індекс узгодженості (IU) розраховуємо:

$$IU = \frac{\lambda_{max} - n}{n-1} \quad (5)$$

Отримане значення IU=0,06 порівнюється із середнім еталонним значенням (таблиця 2.7).

Таблиця 2.7

Шкала еталонних значень випадкових матриць різного розміру

Кількість об'єктів	3	4	5	6	7	8	9	10	11	12	13	14	15
Еталонне значення індексу	0,58	0,90	1,12	1,24	1,32	1,41	1,45	1,49	1,51	1,54	1,56	1,57	1,59

Отже, 0,06<0,1x1,49, тому варто вважати про належну узгодженість в експертних рішеннях, щодо попарного порівняння факторів.

2.3. Оптимізація моделі факторів впливу на функціонування інформаційної технології

Для виконання оптимізації ієрархічної моделі факторів проектування інформаційної системи для студмістечка отриманої в попередньому дослідженні встановлено числовий ряд ваг факторів [61]:

К(g_1) – 50; Ф (g_2) – 70; Б (g_3) – 30; І (g_4) – 30; М (g_5) – 20; АД (g_6) – 20; ТП (g_7) – 50, ГМ (g_8) – 30, ЗЗ (g_9) – 20, ЗД (g_{10}) – 20. Вони складають визначають вихідні значення рівнів в ієрархії [18]. На основі ваг сформовано вихідний вектор: $V_{вих}$(50; 70; 30; 30; 20; 20; 50; 30; 20; 20).

Рівень збіжності вагових значень компонент вихідного ($V_{вих}$) та нормалізованого (V_n) векторів підтверджує гістограма (рис. 2.3).

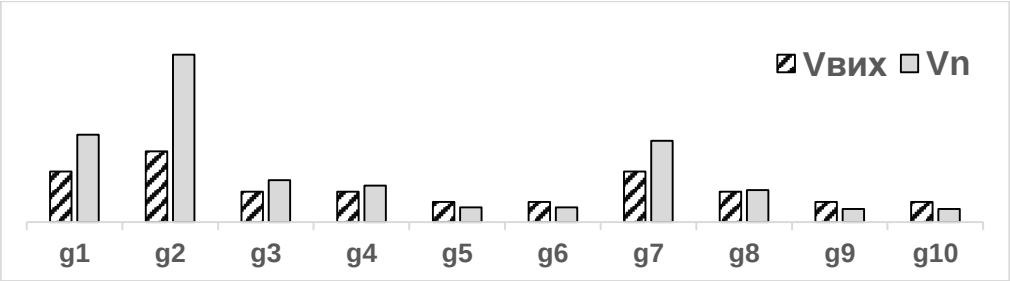


Рис. 2.3. Гістограма вагових значень компонент вихідного ($V_{вих}$) та нормалізованого (V_n) векторів

Отримані величини компонент нормалізованого вектору є оптимізованими величинами. Їх використано для побудови моделі зображеної на рис. 2.4. Метод попарного порівняння дає можливість структурувати досліджувані елементи, врахувати їх ваги та оцінити узгодженість експертних оцінок [63].

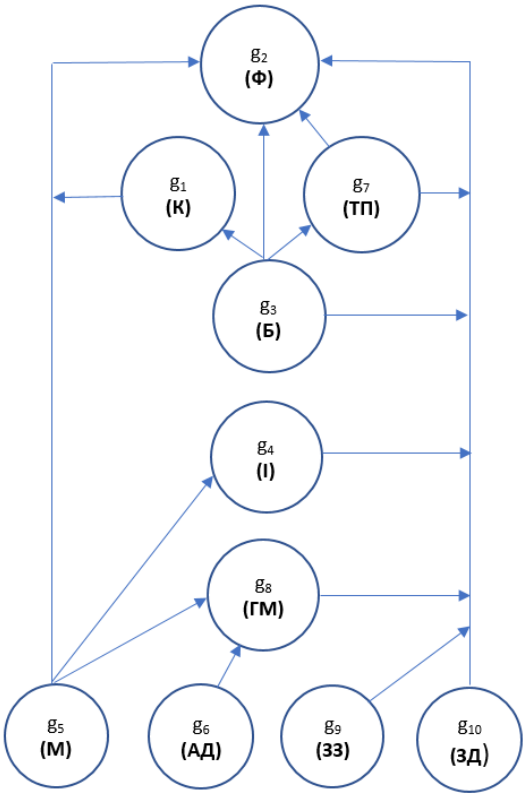


Рис.2.4. Оптимізована ієрархічна модель факторів проектування інформаційної системи для студмістечка

Створення інформаційних систем для закладів вищої освіти є важливим кроком на шляху до модернізації освітнього процесу та управління навчальними закладами. Такі системи дозволяють автоматизувати такі рутинні процеси, як облік успішності здобувачів освіти, управління розкладом занять, комунікація між учасниками освітнього процесу та доступ до навчальних матеріалів. Це допомагає підвищити ефективність роботи персоналу, покращити якість освітніх послуг та забезпечити прозорість управління [55].

Інформаційні системи також допомагають закладам вищої освіти адаптуватися до сучасних викликів, таких як дистанційна освіта, інтегруючись з іншими платформами та забезпечуючи легкий доступ до даних з будь-якого місця. Як наслідок, впровадження таких рішень позитивно впливає як на якість освіти, так і на задоволеність усіх учасників навчального процесу. Таким чином, розробка та впровадження інформаційних систем є необхідним кроком

для навчальних закладів, які прагнуть бути інноваційними, конкурентоспроможними та реагувати на потреби сучасного суспільства.

2.4. Формування альтернативних варіантів процесу функціонування інформаційної технології автоматизованої системи забезпечення діяльності студентського містечка

Щоб провести багатокритеріальне прийняття рішень на основі факторів проектування інформаційної системи (g_1–g_{10}), застосовано структурований підхід із методологією, яка враховує кілька критеріїв (факторів) для оцінки та вибору найкращого варіанту системи.

Метод парного порівняння – це техніка багатокритеріального аналізу для прийняття рішень, яка визначає відносну важливість ряду факторів шляхом їх попарного порівняння. Під час кожного порівняння визначається, який із факторів є важливішим, і йому присвоюється відповідна кількість балів. Подаємо досліджувані фактори проектування:

g_1 – Користувачі (К)

g_2 – Функціональність (Ф)

g_3 – Безпека (Б)

g_4 – Інтеграція (І)

g_5 – Мобільність (М)

g_6 – Аналіз даних (АД)

g_7 – Технічна підтримка (ТП)

g_8 – Гнучкість і масштабованість (ГМ)

g_9 – Зворотний зв’язок (ЗЗ)

g_{10} – Забезпечення доступності (ЗД)

Результатом є ранжування факторів за їх значимістю, що допомагає приймати обґрунтовані рішення щодо пріоритетів під час проектування системи.

Припустимо, що пріоритети базуються на таких припущеннях:

- Безпека (g_3) – найкритичніший фактор, оскільки захист даних студентів, викладачів та адміністрації має найвищий пріоритет.
- Користувачі (g_1) – друга за важливістю категорія, оскільки система має бути зручною для всіх груп користувачів (студенти, викладачі, адміністрація).
- Функціональність (g_2) – визначає, наскільки добре система відповідає потребам студмістечка (наприклад: планування розкладу, відвідування, бронювання приміщень).
- Доступність (g_{10}) – забезпечує можливість використання системи всіма, включаючи людей з обмеженими можливостями.
- Інтеграція (g_4) – важлива для взаємодії з іншими системами (наприклад, платежі, академічні записи).
- Мобільність (g_5) – надає студентам доступ через мобільні пристрої.
- Гнучкість та масштабованість (g_8) – необхідні для адаптації системи до майбутніх змін.
- Аналіз даних (g_6) – допомагає в прийнятті рішень (наприклад, аналіз відвідуваності чи успішності).
- Технічна підтримка (g_7) – забезпечує стабільну роботу системи.
- Зворотний зв’язок (g_9) – сприяє вдосконаленню системи, але має нижчий пріоритет.

На першому етапі дослідження створюється квадратна матриця попарних порівнянь $G=(g_{ij})$. Вона формується за результатами оцінок експертів, тобто кожен експерт приймає рішення щодо множини факторів $G=\{g_1, g_2, g_3\ldots, g_n\}$, з дотриманням умови $g_{ij} + g_{ji}=1$. Якщо один критерій визначено пріоритетнішим за інший, це автоматично означає, що другий критерій поступається йому за важливістю. Наприклад, Безпека (G_3) важливіша за Функціональність (G_2), відповідно, Функціональність (G_2) менш важлива за Безпеку (G_3). Матриця $G=(g_{ij})$ отримає вигляд:

Приклад матриці для факторів проектування студмістечка

	g_1	g_2	g_3	g_4	g_5	g_6	g_7	g_8	g_9	g_{10}
g_1	0	1	0	1	1	1	1	1	1	1
g_2	0	0	0	1	1	1	1	1	1	1
g_3	1	1	0	1	1	1	1	1	1	1
g_4	0	0	0	0	1	1	1	1	1	0
g_5	0	0	0	0	0	1	1	1	1	0
g_6	0	0	0	0	0	0	1	0	1	0
g_7	0	0	0	0	0	0	0	0	1	0
g_8	0	0	0	0	0	1	1	0	1	0
g_9	0	0	0	0	0	0	0	0	0	0
g_{10}	0	0	0	1	1	1	1	1	1	0

Наступний етап включає ранжування факторів. Для кожного фактору обчислюється сума балів у відповідному рядку матриці:

$$s_i = \sum_{j=1}^{n} G_{ij},$$

де s_i– сума рядка матриці, G_{ij}– елементи рядка *i*.

Максимальний бал, який може отримати фактор, дорівнює n−1 = 10−1 = 9. Це значення досягається, якщо фактор домінує над усіма іншими.

Обчислюємо суми балів (s_i) для кожного фактору проектування студмістечка (табл. 2.8):

Таблиця 2.8

Ранжування факторів

Фактор	**Розрахунок**	**Сума (s_i)**
g_1	0 + 1 + 0 + 1 + 1 + 1 + 1 + 1 + 1 + 1	**8**
g_2	0 + 0 + 0 + 1 + 1 + 1 + 1 + 1 + 1 + 1	**7**
g_3	1 + 1 + 0 + 1 + 1 + 1 + 1 + 1 + 1 + 1	**9**
g_4	0 + 0 + 0 + 0 + 1 + 1 + 1 + 1 + 1 + 0	**5**
g_5	0 + 0 + 0 + 0 + 0 + 1 + 1 + 1 + 1 + 0	**4**
g_6	0 + 0 + 0 + 0 + 0 + 0 + 1 + 0 + 1 + 0	**2**
g_7	0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 1 + 0	**1**
g_8	0 + 0 + 0 + 0 + 0 + 1 + 1 + 0 + 1 + 0	**3**
g_9	0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0	**0**
g_{10}	0 + 0 + 0 + 1 + 1 + 1 + 1 + 1 + 1 + 0	**6**

Впорядкуємо фактори за важливістю:

g_3 (Безпека) – 9 балів (найвищий пріоритет)

g_1 (Користувачі) – 8 балів

g_2 (Функціональність) – 7 балів

g_{10} (Доступність) – 6 балів

g_4 (Інтеграція) – 5 балів

g_5 (Мобільність) – 4 балів

g_8 (Гнучкість і масштабованість) – 3 бали

g_6 (Аналіз даних) – 2 бали

g_7 (Технічна підтримка) – 1 бал

g_9 (Зворотний зв'язок) – 0 балів

Отже, фактор «Безпека» визначено найкритичнішим фактором, оскільки він переважає всі інші. «Користувачі» та «Функціональність» також мають високий пріоритет. А, «Зворотній зв'язок» не отримав жодного балу, що свідчить про його найменшу важливість у даному аналізі.

Для визначення альтернативних варіантів проектування інформаційної системи для студмістечка виконаємо нормалізацію факторів.

$w_i = \frac{s_i}{\sum_{i=1}^{n} s_i}$, де w_i – нормалізована вага фактору G_i

Сума всіх балів $\sum s_i = 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1 + 0 = 45$

Виконано обчислення нормалізованих ваг факторів (табл. 2.9):

Таблиця 2.9

Результати обчислення ваг факторів

Фактор	**Вага (w_i)**	**Пояснення**
(g_3) Безпека	0,200 (20,0%)	Найвищий пріоритет – захист даних користувачів.
(g_1) Користувачі	0,178 (17,8%)	Зручність і адаптація для всіх груп користувачів.
(g_2) Функціональність	0,156 (15,6%)	Відповідність основним потребам студмістечка (напр., розклад, бронювання).
(g_{10}) Доступність	0,133 (13,3%)	Інклюзивність, включаючи людей з обмеженими можливостями.

(g_4) Інтеграція	0,111 (11,1%)	Взаємодія з іншими системами (платежі, академічні записи).
(g_5) Мобільність	0,089 (8,9%)	Доступ через мобільні пристрої.
(g_8) Гнучкість і масштабованість	0,067 (6,7%)	Адаптація до майбутніх змін.
(g_6) Аналіз даних	0,044 (4,4%)	Підтримка прийняття рішень (напр., аналіз успішності).
(g_7) Технічна підтримка	0,022 (2,2%)	Стабільність і надійність системи.
(g_9) Зворотний зв'язок	0,000 (0,0%)	Не впливає на поточні пріоритети.

Отже, найсильніший пріоритет отримали фактори (сумарно 53,4%): «Безпека» (20%), «Користувачі» (17,8%), «Функціональність» (15,6%). Система має насамперед забезпечувати захист даних, зручність і базові функції. Середній пріоритет (сумарно 33,3%): «Доступність», «Інтеграція», «Мобільність». Ці фактори є важливими, але менш критичними, ніж безпека чи функціональність.

Низький пріоритет (сумарно 13,3%): «Технічна підтримка» та «Аналіз даних» мають обмежений вплив, а «Зворотній зв'язок» взагалі не враховано.

На основі ранжування факторів фокусна група експертів виконала порівняння трьох гіпотетичних систем, де визначила ключові критерії реалізації інформаційної системи для студмістечка (табл. 2.10).

Таблиця 2.10

Визначення домінуючих факторів для кожної інформаційної системи

Фактор	**Вага (w_i)**	**Система A**	**Система B**	**Система C**
(g_3) Безпека	0,200	**Висока** (домінує)	Середня	**Низька** (ризик)
(g_1) Користувачі	0,178	**Дружній інтерфейс**	Складний інтерфейс	Середня зручність
(g_2) Функціональність	0,156	Обмежена	**Повний функціонал**	Частковий функціонал
(g_{10}) Доступність	0,133	Середня	Середня	**Висока**
(g_4) Інтеграція	0,111	**Слабка** (недолік)	Середня	Середня
(g_5) Мобільність	0,089	Базовий доступ	**Оптимізована для мобільних**	Обмежена мобільність
(g_8) Гнучкість	0,067	Середня	Низька	**Висока**
(g_6) Аналіз даних	0,044	Відсутній	Базовий	**Розширений аналіз**
(g_7) Технічна підтримка	0,022	Надійна	**Відсутня**	Обмежена

Наступний етап включає, формування матриці значень для кожного варіанту інформаційної системи (наприклад, Система А, Система В, Система С) за факторами проектування студмістечка. Ця матриця відображає, наскільки добре кожна система задовольняє кожен фактор і служить основою для порівняння альтернатив. Вона показує, як системи оцінюються за факторами, такими як безпека (g_3), користувачі (g_1), функціональність (g_2) тощо. Формування виконуємо за таким правилом:

1. Рядки відповідають факторам оцінки ($g_1 – g_{10}$).
2. Стовпці відповідають варіантам систем (альтернативам, Система А, Система В, Система С).
3. Елементи (x_{ij}) – числові оцінки, що відображають, наскільки альтернатива j відповідає критерію *i*. (табл.2.11).

Таблиця 2.11

Матриця оцінок

Фактор	**Система А**	**Система В**	**Система С**
g_1 – Користувачі (К)	8	6	7
g_2 – Функціональність (Ф)	7	9	6
g_3 – Безпека (Б)	9	5	4
g_4 – Інтеграція (І)	4	7	6
g_5 – Мобільність (М)	6	8	5
g_6 – Аналіз даних (АД)	5	4	8
g_7 – Технічна підтримка (ТП)	6	3	5
g_8 – Гнучкість і масштабованість (ГМ)	7	6	7
g_9 – Зворотний зв’язок (ЗЗ)	5	5	5
g_{10} – Забезпечення доступності (ЗД)	6	7	9

Розрахунок інтегрального рейтингу систем А, В та С.

Використовуємо формулу лінійної згортки:

$$\text{Рейтинг системи} = \sum_{j=1}^{10} w_i \, x_i$$

де: w_i – вага *i*-го фактору, x_i- оцінка системи за цим фактором (див. табл. 4).

Розрахунок інтегрального рейтингу альтернативних варіантів інформаційної системи для студмістечка:

Система А

Рейтинг системи = (8×0,178) + (7×0,156) + (9×0,200) + (4×0,111) + (6×0,089) + (5×0,044) + (6×0,022) + (7×0,067) + (6×0,133) **= 1,424 + 1,092 + 1,800 + 0,444 + 0,534 + 0,220 + 0,132 + 0,469 + 0,798** = **6,913**

Система В

Рейтинг системи = (6×0,178) + (9×0,156) + (5×0,200) + (7×0,111) + (8×0,089) + (4×0,044) + (3×0,022) + (6×0,067) + (7×0,133) = **1,068 + 1,404 + 1,000 + 0,777 + 0,712 + 0,176 + 0,066 + 0,402 + 0,931** = **6,536**

Система С

Рейтинг системи = (7×0,178) + (6×0,156) + (4×0,200) + (6×0,111) + (5×0,089) + (8×0,044) + (5×0,022) + (7×0,067) + (9×0,133) = **1,246 + 0,936 + 0,800 + 0,666 + 0,445 + 0,352 + 0,110 + 0,469 + 1,197** = **6,221**

Результати ранжування систем:

Система А – **6,91** (Найкраща безпека та зручність)

Система В – **6,54** (Найкраща функціональність і мобільність)

Система С – **6,22** (Найкраща аналітика та доступність)

Система **А** є беззаперечним лідером завдяки двом ключовим перевагам. По-перше, вона забезпечує найнадійніший захист даних – цей показник є критично важливим для організацій, які працюють із конфіденційною інформацією. По-друге, інтерфейс користувача цієї системи розроблено з орієнтацією на зручність, що робить роботу з нею комфортною та інтуїтивно зрозумілою. Хоча можливості інтеграції дещо обмежені, цей недолік можна компенсувати додатковими зусиллями з налаштування.

Система **B** виділяється своєю функціональністю – вона пропонує найширший набір інструментів для вирішення складних завдань. Особливо варто відзначити мобільну версію, яка ідеально підходить для користувачів, що постійно перебувають у русі. Однак перед впровадженням варто врахувати дві важливі деталі: необхідність додаткових інвестицій у систему безпеки та потребу в організації якісної технічної підтримки.

Система **C** знайде застосування в певних сценаріях. Вона демонструє вражаючі показники доступності, що робить її ідеальним вибором для забезпечення інклюзивності. Також варто відзначити розширені аналітичні функції. Проте низький рівень безпеки значно обмежує сферу застосування – система підходить лише для внутрішніх процесів, де конфіденційність даних не є критичним фактором.

При виборі оптимального рішення рекомендується орієнтуватися на специфічні потреби організації. У більшості випадків, особливо коли йдеться про роботу з конфіденційною інформацією, система **A** є найкращим вибором. Якщо на першому плані стоять функціональність і мобільність, а технічні ризики можна мінімізувати, варто розглянути систему **B**. Система **C** підходить для нішевих проектів, де пріоритетами є доступність і аналітика.

2.5. Проектування інформаційної технології автоматизованої системи забезпечення діяльності студентського містечка з використанням засобів нечіткої логіки

Для поділу факторів на дві групи за допомогою нечіткої логіки можна використати критерії, які відображають їхню природу: одні фактори більше стосуються користувацького досвіду та взаємодії, інші — технічних аспектів і функціональності системи. Ось пропозиція поділу:

$$Q = F_Q\,(M,\ T)$$

Група 1: Користувацько-орієнтовані фактори

- Користувачі - m_1
- Мобільність – m_2
- Зворотний зв'язок – m_3
- Забезпечення доступності – m_4

Назва: **«Користувацький досвід»**. Ця група охоплює фактори, пов’язані з взаємодією користувачів із системою, їхньою зручністю, доступом і відгуками.

$$M = F_M\,(m_1, m_2, m_3, m_4)$$

Група 2: Технічно-функціональні фактори

- Функціональність – t_1
- Безпека – t_2
- Інтеграція – t_3
- Аналіз даних – t_4
- Технічна підтримка – t_5
- Гнучкість і масштабованість – t_6

Назва: **«Технічна ефективність»**. Ця група включає фактори, що стосуються технічних характеристик, безпеки, інтеграційних можливостей і масштабованості системи.

$$T = F_T\,(t_1, t_2, t_3, t_4, t_5, t_6)$$

Такий поділ дозволяє чітко розмежувати аспекти, пов’язані з користувачами, та технічні компоненти, що є логічним для аналізу з використанням нечіткої логіки. Будуємо багаторівневу модель формування інтегрального показника якості порівняльного процесу дослідження процесу реалізації інформаційної системи автоматизації роботи студмістечка (рис. 2.5):

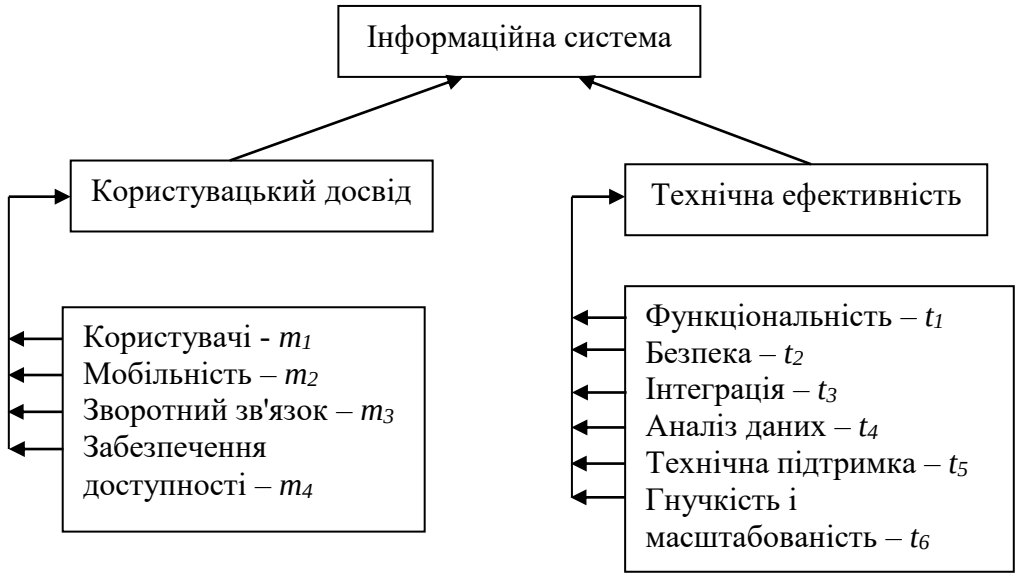


Рис. 2. 5. Багаторівнева модель формування інтегрального показника якості порівняльного процесу дослідження процесу реалізації інформаційної системи автоматизації роботи студмістечка

Терм-множини значень лінгвістичних змінних

Змінна	Лінгвістичне трактування змінної	Універсальна база значень (терм-множина U)	Лінгвістичні терми (множина H)
m_1	Користувачі	(20-60) роки	досвідчений, юзер, початківець
m_2	Мобільність	(0-100) %	повністю мобільна, обмежено мобільна, не мобільна
m_3	Зворотний зв'язок	(1-5) у.о.	зручно, посередньо, складно
m_4	Забезпечення доступності	(20-100) %	висока, середня, низька
t_1	Функціональність	(0-100) %	функціональна, обмежено функціональна, не функціональна
t_2	Безпека	(0-100) %	безпечна, частково безпечна, небезпечна
t_3	Інтеграція	(0-100) %	повна сумісність, часткова сумісність, не сумісність
t_4	Аналіз даних	(0-100) %	високоточний, середньої точності, не точний
t_5	Технічна підтримка	(20-100) %	швидка, помірна, повільна
t_6	Гнучкість і масштабованість	(1-5) у.о.	високоадаптивна, обмежено масштабована, не маштабована

Представимо терми лінгвістичної змінної m_1нечіткими множинами:

$$\text{Досвідчений} = \left(\frac{1}{20}; \frac{0,85}{30}; \frac{0,6}{40}; \frac{0,25}{50}; \frac{0,11}{60} \right), \text{ роки;}$$

$$\text{Юзер} = \left(\frac{0,11}{20};\frac{0,6}{30};\frac{1}{40};\frac{0,6}{50};\frac{0,11}{60}\right), \text{роки};$$

$$\text{Початківець} = \left(\frac{0,11}{20};\frac{0,25}{30};\frac{0,6}{40};\frac{0,85}{50};\frac{1}{60}\right), \text{ роки.}$$

Представимо терми лінгвістичної змінної m_2нечіткими множинами:

$$\text{Повністю мобільна} = \left(\frac{0,11}{0};\frac{0,25}{25};\frac{0,5}{50};\frac{0,75}{75};\frac{1}{100}\right), \%;$$

$$\text{Обмежено мобільна} = \left(\frac{0,11}{0};\frac{0,5}{25};\frac{1}{50};\frac{0,5}{75};\frac{0,11}{100}\right), \%;$$

$$\text{Немобільна} = \left(\frac{1}{0};\frac{0,75}{25};\frac{0,5}{50};\frac{0,25}{75};\frac{0,11}{100}\right), \%.$$

Представимо терми лінгвістичної змінної m_3нечіткими множинами:

$$\text{Зручно} = \left(\frac{0,11}{1};\frac{0,33}{2};\frac{0,77}{3};\frac{0,88}{4};\frac{1}{5}\right), \text{ у.о.};$$

$$\text{Посередньо} = \left(\frac{0,11}{1};\frac{0,77}{2};\frac{1}{3};\frac{0,77}{4};\frac{0,11}{5}\right), \text{ у.о.};$$

$$\text{Складно} = \left(\frac{1}{1};\frac{0,88}{2};\frac{0,77}{3};\frac{0,33}{4};\frac{0,11}{5}\right), \text{ у.о.}$$

Представимо терми лінгвістичної змінної m_4нечіткими множинами:

$$\text{Висока} = \left(\frac{0,11}{20};\frac{0,25}{40};\frac{0,6}{60};\frac{0,85}{80};\frac{1}{100}\right), \%;$$

$$\text{Середня} = \left(\frac{0,11}{20};\frac{0,6}{40};\frac{1}{60};\frac{0,6}{80};\frac{0,11}{100}\right), \%;$$

$$\text{Низька} = \left(\frac{1}{20};\frac{0,85}{40};\frac{0,6}{60};\frac{0,25}{80};\frac{0,11}{100}\right), \%.$$

Матриця знань для лінгвістичної змінної *M* «Користувацько-орієнтована ефективність»

Якщо				*Тоді*
m_1	m_2	m_3	m_4	Користувацько-орієнтована ефективність
юзер	немобільний	складно	низька	низька
початківець	обмежено мобільна	посередньо	низька	
юзер	повністю мобільна	посередньо	середня	середня
юзер	обмежено мобільна	посередньо	висока	
досвідчений	повністю мобільна	зручно	висока	висока
юзер	повністю мобільна	зручно	середня	

Представимо терми лінгвістичної змінної t_1нечіткими множинами:

$$\text{Функціональна} = \left(\frac{0,11}{0};\frac{0,22}{25};\frac{0,78}{50};\frac{0,89}{75};\frac{1}{100}\right),\ \%;$$

$$\text{Обмежено функціональна} = \left(\frac{0,11}{0};\frac{0,78}{25};\frac{1}{50};\frac{0,78}{75};\frac{0,11}{100}\right),\%;$$

$$\text{Не функціональна} = \left(\frac{1}{0};\frac{0,89}{25};\frac{0,78}{50};\frac{0,22}{75};\frac{0,11}{100}\right),\%.$$

Представимо терми лінгвістичної змінної t_2нечіткими множинами:

$$\text{Безпечна} = \left(\frac{0,11}{0};\frac{0,33}{25};\frac{0,77}{50};\frac{0,88}{75};\frac{1}{100}\right),\ \%;$$

$$\text{Частково безпечена} = \left(\frac{0,11}{0};\frac{0,77}{25};\frac{1}{50};\frac{0,77}{75};\frac{0,11}{100}\right),\ \%;$$

$$\text{Небезпечна} = \left(\frac{1}{0};\frac{0,88}{25};\frac{0,55}{50};\frac{0,33}{75};\frac{0,11}{100}\right),\ \%.$$

Представимо терми лінгвістичної змінної t_3нечіткими множинами:

$$\text{Повна сумісність} = \left(\frac{0,11}{0};\frac{0,33}{25};\frac{0,77}{50};\frac{0,88}{75};\frac{1}{100}\right),\ \%;$$

Часткова сумісність = $\left(\frac{0,11}{0};\frac{0,88}{25};\frac{1}{50};\frac{0,88}{75};\frac{0,11}{100}\right)$, %;

Несумісність= $\left(\frac{1}{0};\frac{0,88}{25};\frac{0,55}{50};\frac{0,33}{75};\frac{0,11}{100}\right)$, %.

Представимо терми лінгвістичної змінної t_4нечіткими множинами:

Високоточний = $\left(\frac{0,11}{0};\frac{0,22}{25};\frac{0,78}{50};\frac{0,89}{75};\frac{1}{100}\right)$,%;

Середньої точності = $\left(\frac{0,11}{0};\frac{0,78}{25};\frac{1}{50};\frac{0,78}{75};\frac{0,11}{100}\right)$, %.

Неточний = $\left(\frac{1}{0};\frac{0,89}{25};\frac{0,78}{50};\frac{0,22}{75};\frac{0,11}{100}\right)$,%

Представимо терми лінгвістичної змінної t_5нечіткими множинами:

Швидка = $\left(\frac{0,11}{20};\frac{0,25}{40};\frac{0,5}{60};\frac{0,75}{80};\frac{1}{100}\right)$, %;

Помірна = $\left(\frac{0,11}{20};\frac{0,5}{40};\frac{1}{60};\frac{0,5}{80};\frac{0,11}{100}\right)$, %.

Повільна =$\left(\frac{1}{0};\frac{0,75}{40};\frac{0,5}{60};\frac{0,25}{80};\frac{0,11}{100}\right)$, %

Представимо терми лінгвістичної змінної t_6нечіткими множинами:

Високоадаптивна=$\left(\frac{0,11}{1};\frac{0,33}{2};\frac{0,77}{3};\frac{0,88}{4};\frac{1}{5}\right)$, у.о.;

Обмежено маштабована = $\left(\frac{0,11}{1};\frac{0,77}{2};\frac{1}{3};\frac{0,77}{4};\frac{0,11}{5}\right)$, у.о.;

Немаштабована = $\left(\frac{1}{1};\frac{0,88}{2};\frac{0,77}{3};\frac{0,33}{4};\frac{0,11}{5}\right)$, у.о.

На основі матриці знань (табл.) будуємо нечіткі логічні рівняння:

- Для терму «низька»

$$\mu^{низька} = \mu^{юзер}(m_1) \wedge \mu^{немоб}(m_2) \wedge \mu^{скл}(m_3) \wedge \mu^{низька}(m_4) \vee$$
$$\mu^{початк}(m_1) \wedge \mu^{об.моб}(m_2) \wedge \mu^{посеред}(m_3) \wedge \mu^{низька}(m_4);$$

- Для терму «середня»

$$\mu^{середн} = \mu^{юзер}(m_1) \wedge \mu^{повн.м}(m_2) \wedge \mu^{посердн}(m_3) \wedge \mu^{середн}(m_4) \vee$$
$$\mu^{юзер}(m_1) \wedge \mu^{обм.моб}(m_2) \wedge \mu^{посередн}(m_3) \wedge \mu^{вис}(m_4);$$

- Для терму «висока»

$$\mu^{вис} = \mu^{досв}(m_1) \wedge \mu^{повн.м}(m_2) \wedge \mu^{зручно}(m_3) \wedge \mu^{вис}(m_4) \vee$$
$$\mu^{досв}(m_1) \wedge \mu^{повн.моб}(m_2) \wedge \mu^{зручно}(m_3) \wedge \mu^{сер}(m_4);$$

Підставимо значення функцій належності у нечіткі логічні рівняння:

$$\mu^{низька} = 0,6 \wedge 0,25 \wedge 0,33 \wedge 0,25 \vee 0,25 \wedge 0,5 \wedge 0,77 \wedge 0,25 = 0,6$$
$$\mu^{середн} = 0,6 \wedge 0,75 \wedge 0,77 \wedge 0,6 \vee 0,6 \wedge 0,5 \wedge 0,77 \wedge 0,85 = 0,77$$
$$\mu^{вис} = 0,85 \wedge 0,75 \wedge 0,88 \wedge 0,85 \vee 0,85 \wedge 0,75 \wedge 0,88 \wedge 0,6 = 0,88$$

Матриця знань для лінгвістичної змінної *T* «Технічно-функціональна ефективність»

Якщо						*Тоді*
t_1	t_2	t_3	t_4	t_5	t_6	Технічно-функціональна ефективність
нефункціональна	небезпечн	несумісність	неточний	помірна	не маштабована	низька
нефункціональна	частково безпечна	несумісність	сер. точності	повільна	обмежено маштабована	
обмежено функціональна	частково безпечна	повна сумісність	сер. точності	помірна	високоадаптивна	середня
функціональна	безпечна	часткова сумісність	сер. точності	швидка	Обмежено маштабована	
функціональна	безпечна	повна сумісність	високоточний	швидка	високоадаптивна	висока
функціональна	безпечна	часткова сумісність	сер. точності	помірна	високоадаптивна	

На основі матриці знань (табл.) будуємо нечіткі логічні рівняння:

- Для терму «низька»

$$\mu^{низк} = \mu^{нефунк}(t_1) \wedge \mu^{небезп}(t_2) \wedge \mu^{несум}(t_3) \wedge \mu^{неточн}(t_4) \wedge \mu^{помір}(t_5) \wedge \mu^{немашт}(t_6) \vee$$
$$\mu^{нефунк}(t_1) \wedge \mu^{част.безп}(t_2) \wedge \mu^{несум}(t_3) \wedge \mu^{помір}(t_4) \wedge \mu^{повіл}(t_5) \wedge \mu^{обм.машт}(t_6);$$

- Для терму «середня»

$$\mu^{сер} = \mu^{обм.функ}(t_1) \wedge \mu^{част.безп}(t_2) \wedge \mu^{п.сум}(t_3) \wedge \mu^{помірн}(t_4) \wedge \mu^{швид}(t_5) \wedge \mu^{вис.адап}(t_6)$$
$$\vee \mu^{функ}(t_1) \wedge \mu^{безп}(t_2) \wedge \mu^{частк.сум}(t_3) \wedge \mu^{помір}(t_4) \wedge \mu^{швидк}(t_5) \wedge \mu^{обм.машт}(t_6);$$

- Для терму «висока»

$$\mu^{вис} = \mu^{функ}(t_1) \wedge \mu^{безп}(t_2) \wedge \mu^{п.сум}(t_3) \wedge \mu^{в.точн}(t_4) \wedge \mu^{швидк}(t_5) \wedge \mu^{в.адапт}(t_6) \vee$$
$$\mu^{функ}(t_1) \wedge \mu^{безп}(t_2) \wedge \mu^{ч.сум}(t_3) \wedge \mu^{помір}(t_4) \wedge \mu^{помір}(t_5) \wedge \mu^{в.адапт}(t_6);$$

Підставимо значення функцій належності факторів $t_1 - t_6$ у нечіткі логічні рівняння:

$$\mu^{низьк} = 0,22 \wedge 0,33 \wedge 0,33 \wedge 0,22 \wedge 0,5 \wedge 0,33 \vee$$
$$0,22 \wedge 0,77 \wedge 0,33 \wedge 0,78 \wedge 0,25 \wedge 0,33 = 0,5$$
$$\mu^{середн} = 0,78 \wedge 0,77 \wedge 0,88 \wedge 0,78 \wedge 0,75 \wedge 0,77 \vee$$
$$0,89 \wedge 0,88 \wedge 0,88 \wedge 0,78 \wedge 0,75 \wedge 0,33 = 0,88$$
$$\mu^{вис} = 0,89 \wedge 0,88 \wedge 0,88 \wedge 0,89 \wedge 0,75 \wedge 0,88 \vee$$
$$0,89 \wedge 0,88 \wedge 0,88 \wedge 0,88 \wedge 0,78 \wedge 0,5 = 0,89$$

Для найвищого рівня Q функції належності матимуть вид:

— для терму «низька»

$$\mu_Q^{низьк} = 0,6 \wedge 0,5 \vee 0,77 \wedge 0,33 = 0,6$$

— для терму «середня»

$$\mu_Q^{сер} = 0,77 \wedge 0,89 \vee 0,77 \wedge 0,88 = 0,88$$

— для терму «висока»

$$\mu_Q^{вис} = 0,88 \wedge 0,89 \vee 0,77 \wedge 0,89 = 0,89$$

Операцію дефазифікації проведем за формулою:

$$Q = \frac{\sum_{i=1}^{m} u_i \cdot \mu(u_i)}{\sum_{i=1}^{m} \mu(u_i)}$$

Приймемо умовні межі для змінної Q =1…100 од. Розрахунок проведемо за трьома точками поділу: 1 од, 50 од, 100 од. У результаті розрахунку отримаємо числове значення ефективності інформаційної системи:

$$Q = \frac{1 \cdot 0,6 + 50 \cdot 0,88 + 100 \cdot 0,89}{0,6 + 0,88 + 0,89} = 56,37 од.$$

Для прикладу, на рис.2.6 представлено графічну візуалізацію функцій належності лінгвістичної змінної t_1 „Функціональність”.

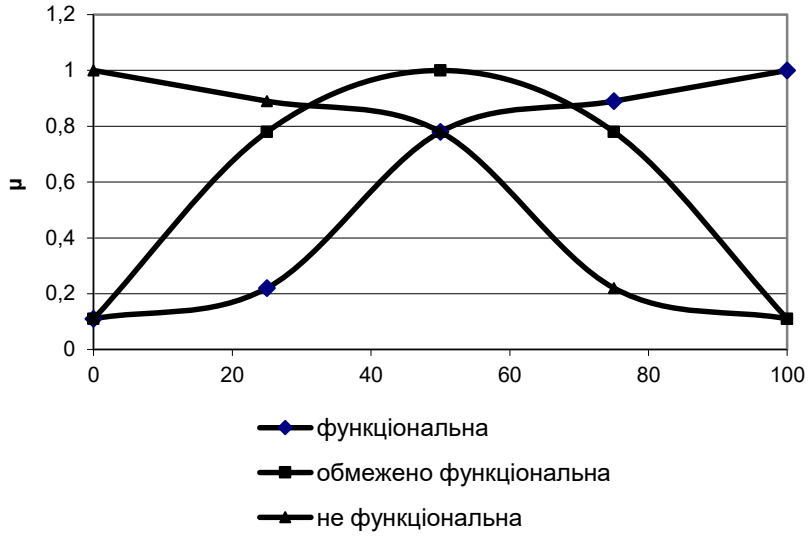


Рис. 2.6. Функції належності лінгвістичної змінної t_1 «Функціональність»

Висновки до розділу 2.

1. Використання інформаційних систем для студентських містечок має багато переваг, які сприяють ефективному управлінню житловими, адміністративними та навчальними процесами. Це може покращити досвід як студентів, так і адміністративного персоналу, автоматизуючи завдання, полегшуючи комунікацію та забезпечуючи постійний доступ до інформації. Основні переваги такої системи включають ряд функцій існування студмістечка. Дані системи дозволяють легко управляти інформацією про заповнюваність, розташування та доступність кімнат, а також автоматизує процес бронювання та реєстрації. Це в свою чергу зменшує навантаження на адміністраторів і дозволяє студентам переглядати та обирати доступні для них варіанти. Користувачі зможуть здійснювати платежі онлайн, отримувати рахунки та нагадування, а також відстежувати фінансові операції, зменшуючи навантаження на бухгалтерію та підвищуючи прозорість платежів.

2. Завдяки інформаційним системам студмістечка зможуть краще управляти використанням енергоресурсів і зменшити непотрібне споживання. Наприклад, аналізуючи споживання електроенергії та води, можна оптимізувати використання ресурсів і зменшити вплив на навколишнє

середовище. Автоматизація багатьох рутинних процесів, в свою чергу, зменшує ручну роботу в адміністративному відділі. Це не лише пришвидшує надання послуг, але й зменшує ризик людських помилок, покращує якість обслуговування та впорядковує роботу. Загалом, інформаційні системи студмістечок забезпечують високий рівень комфорту для студентів, підвищують адміністративну ефективність, сприяють безпеці та інтеграції для всіх мешканців та співробітників. Це інструмент, який може створити сучасне, інтегроване та безпечне середовище. І це ще не всі переваги використання інформаційною системою студмістечок.

3. Індекс узгодженості проектованої інформаційної системи IU склав 0,06, що варто вважати про належну узгодженість в експертних рішеннях, щодо попарного порівняння факторів.

4. Проведено оптимізацію ієрархічної моделі факторів проектування інформаційної системи для студмістечка, яка дала можливість більш детально поділити фактори за важливістю, а саме з чотирьох рівнів ми отримали шість рівнів.

5. Для оцінки та вибору найкращого варіанту системи було використано метод парного порівняння – це техніка багатокритеріального аналізу для прийняття рішень, яка визначає відносну важливість ряду факторів шляхом їх попарного порівняння. Для цього було систему розбито на три підсистеми. Як показали розрахунки, система А є беззаперечним лідером завдяки двом ключовим перевагам. По-перше, вона забезпечує найнадійніший захист даних – цей показник є критично важливим для організацій, які працюють із конфіденційною інформацією. По-друге, інтерфейс користувача цієї системи розроблено з орієнтацією на зручність, що робить роботу з нею комфортною та інтуїтивно зрозумілою.

6. Для вирішення проблемних моментів та неточностей при розрахунках в умовах невизначеності та присутності істини, нами було запропоновано використати нечітку логіку. Для цього було побудовано багаторівневу модель формування інтегрального показника якості порівняльного процесу

дослідження процесу реалізації інформаційної системи автоматизації роботи студмістечка, а також описані терм-множини значень лінгвістичних змінних. Як показали результати, отримано числове значення ефективності інформаційної системи 56,37 од, що більше за 50 та менше за 100 од. Дана операція дає можливість побудувати прогностичну модель, що відкриває можливості прогнозування якості розглянутого процесу.

РОЗДІЛ 3
РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ АВТОМАТИЗОВАНОЇ СИСТЕМИ ЗАБЕЗПЕЧЕННЯ ДІЯЛЬНОСТІ СТУДЕНТСЬКОГО МІСТЕЧКА

3.1. Проектування та опис етапів реалізації інформаційної технології

Перед безпосередньою розробкою системи важливим етапом є детальне проектування, яке бере до уваги наперед поставлені завдання, які стосуються функціоналу. Згідно загальних завдань даної роботи, можна виділити декілька явних кандидатів на роль підсистем, а саме:

- *Підсистема для користувачів*. Щоб ефективно організувати підсистему для користувачів, для початку необхідно провести розподіл функціоналу веб-сервісу відповідно до типу зареєстрованого користувача. При поставлених завданнях перед системою має зміст використати наступні ролі: студент, менеджер та адміністратор. Розподіл ролей може відбуватись наступним чином: роль студента відповідає всім особам, які навчаються в закладі освіти, менеджер - це працівники адміністрації студентського містечка, адміністратор - це завідувач організації студентського містечка.
- *Підсистема для новин.* Дана підсистема ставить собі за мету вирішити незручності пов’язані із поширенням важливої інформації серед мешканців студентського містечка. При відсутності відповідного програмного забезпечення, очевидним рішення можуть слугувати настінні паперові плакати, які мають явний недолік: студенти, які поспішають по своїх справах можуть несвідомо їх проігнорувати, що призведе до браку інформації, яку адміністрація студентського містечка намагалась поширити. Саме для того варто спроектувати підсистему для ефективного створення та поширення новин серед користувачів веб-сервісу. Необхідним подальшим етапом може слугувати розробка веб-сторінки із представленням усіх новини, а також варто представити

можливість створення важливих новин, які будуть мати пріоритет в графічному відображенні.

- *Підсистема для листування.* Існують випадки, які потребують негайного поширення новин, без очікування на те, коли користувач, який має роль студента, зможе зайти на веб-сервіс, щоб прочитати їх. Вирішенням даної проблеми може слугувати надсилання персоналізованих листів, які проінформують користувача про важливу інформацію, яку адміністрація студентського містечка хотіла поширити.
- *Підсистема бронювання кімнат.* Варто зазначити, що в студентському містечку існують різні кімнати для проживання студентів, вони можуть відрізнятись загальним станом, поверхом, зручностями, місткістю і т.д. Саме тому мешканцям студентського містечка важливо розуміти, які кімнати є зайняті, а в які потенційно є шанс поселитися, щоб якісно покращити власні умови проживання. Іноді кімнати можуть стати вільними у зв’язку з тим, що деякі мешканці можуть змінити місце проживання через особисті обставини. Саме тому необхідно продумати алгоритм зміни кімнати проживання студентом, щоб оптимізувати відповідні трудомісткі процеси.

Для того, щоб краще зрозуміти зони відповідальності кожної із запропонованих підсистем, варто скористатись різного роду діаграмами. Діаграма прецедентів [79] (англ. use-case diagram) - це тип діаграми, яка відображає співвідношення акторів та прецедентів в аплікації. Цінність даної діаграми полягає у графічному представленні сутностей чи акторів, що взаємодіють з веб-сервісом за допомогою варіантів використання. Варіант використання (англ. usecase) використовують для описання функціоналу, який веб-сервіс надає актору. Для кращого представлення функціоналу було розроблено діаграми прецедентів окремо для кожної ролі, а саме: студент, менеджер, адміністратор, а також для звичайного незареєстрованого користувача. Приклад діаграми прецедентів [79] для незареєстрованого користувача (рис. 3.1):

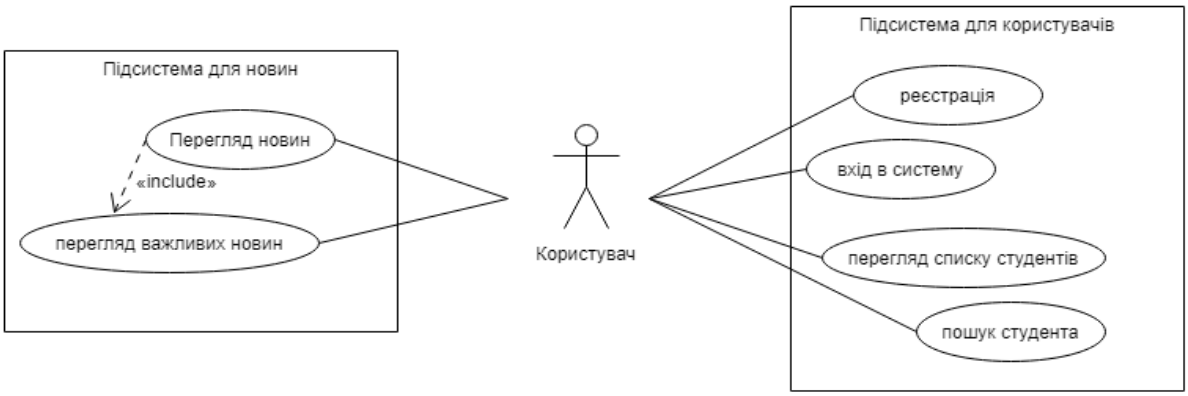


Рис. 3.1. Діаграма прецедентів для незареєстрованого користувача

Діаграма прецедентів [79] для ролі студента (рис.3.2.):

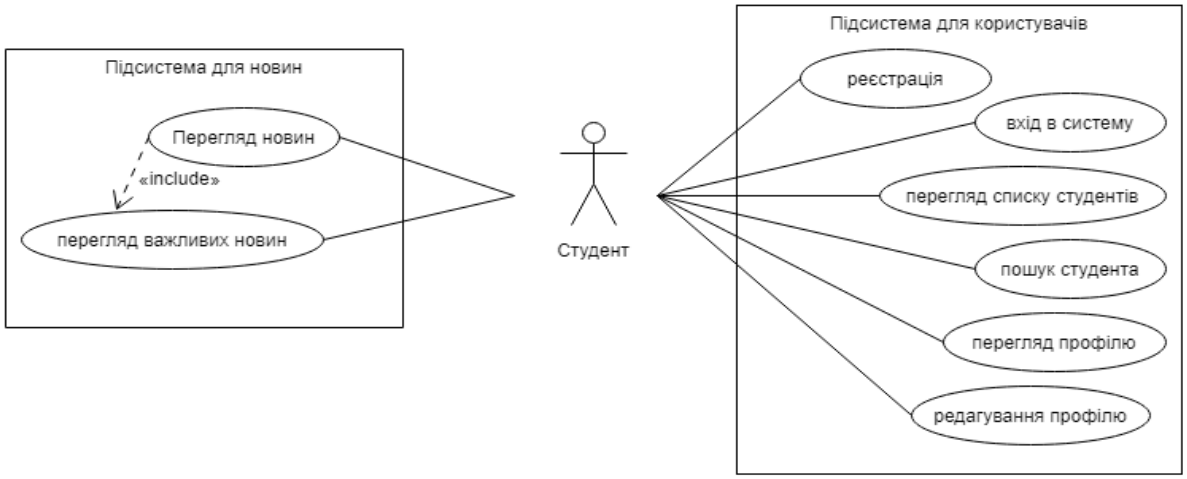


Рис. 3.2. Діаграма прецедентів для ролі студента

Якщо окремо розглядати користувача, який має роль менеджера студентського містечка, то діаграма прецедентів [79], яка описує список доступного функціоналу для даної ролі може бути представлена наступним чином (рис. 3.3):

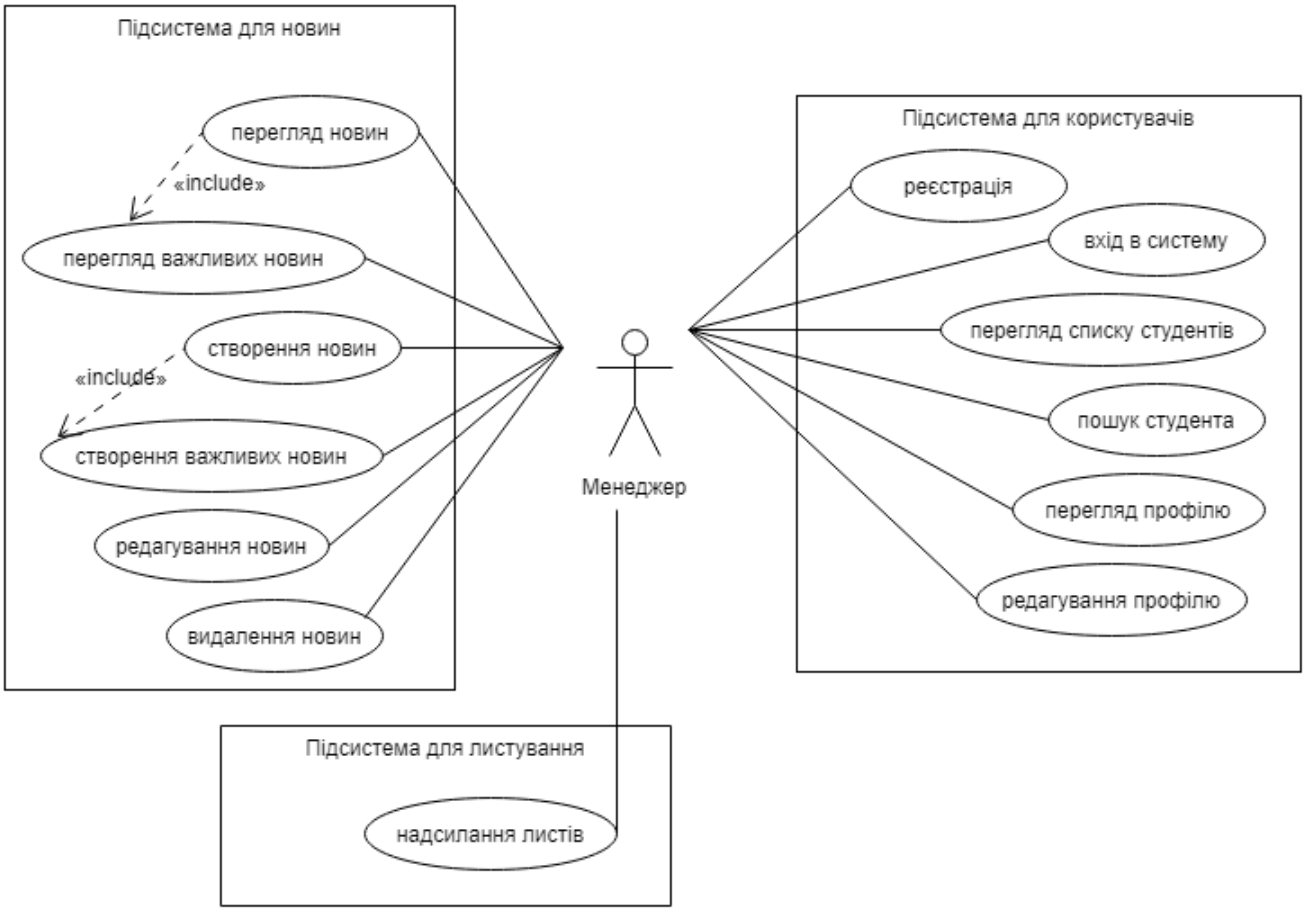


Рис. 3.3. Діаграма прецедентів для ролі менеджера

Варто також розробити діаграму прецедентів [79] для ролі з найбільшим рівнем доступу, а саме для адміністратора студентського містечка. Беручи до уваги необхідний для веб-аплікації функціонал, було запропоновано наступну діаграму з описом можливостей для даної ролі (рис. 3.4):

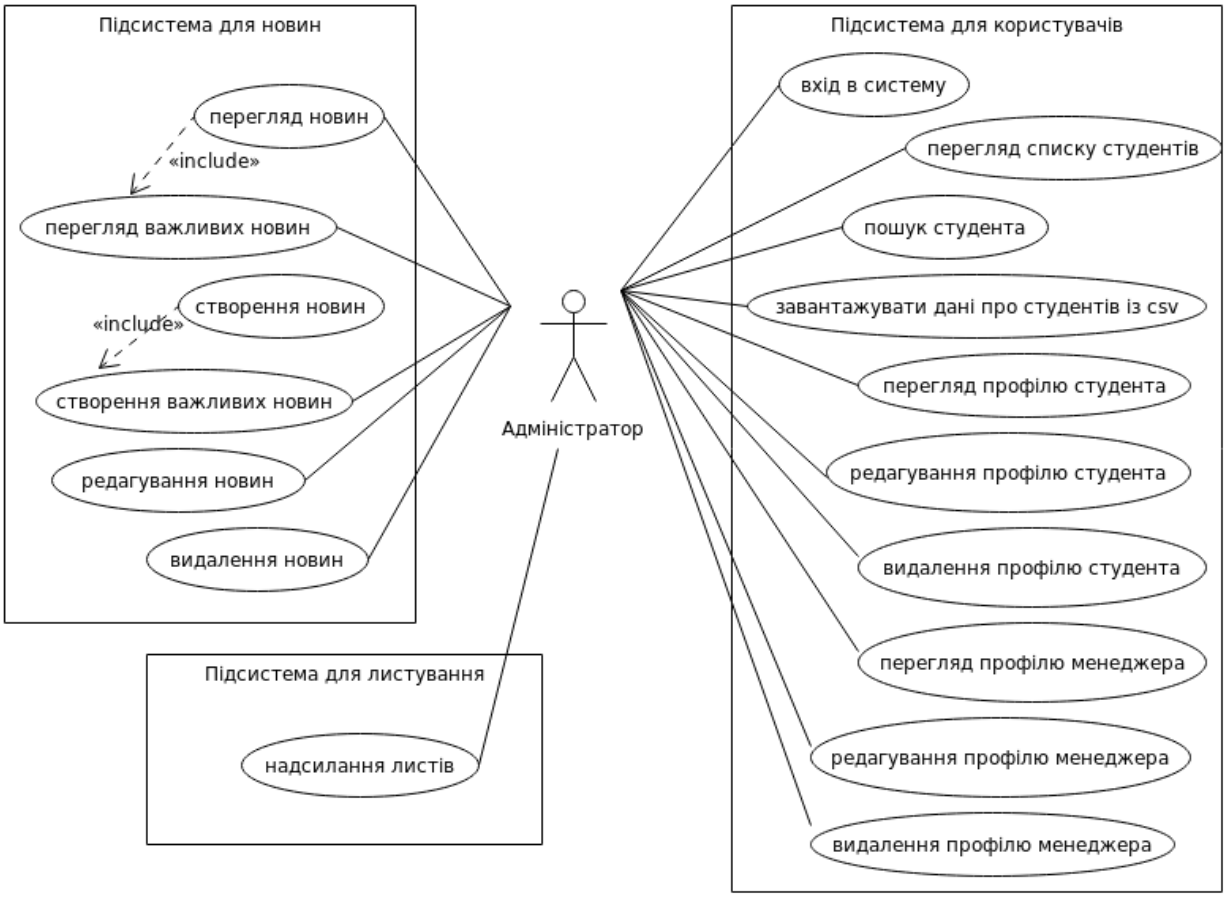


Рис. 3.4. Діаграма прецедентів для ролі адміністратора студентського містечка

Варто зазначити, що діаграма прецедентів [79] використовується для опису лише певної частини програмного забезпечення з точки зору проектування веб-сервісу. Для того, щоб розглянути веб-сервіс з альтернативної сторони, потрібно скористатись діаграмою класів, яка використовується для представлення статичної структури моделі системи в межах об’єктно-орієнтованого програмування. Такого роду діаграма відображає класи, об’єкти, а також їхні асоціації. Для прикладу може розглянути базову діаграму класів [71] для підсистеми для користувачів разом із підсистемою новин для веб-сервісу (рис. 3.5):

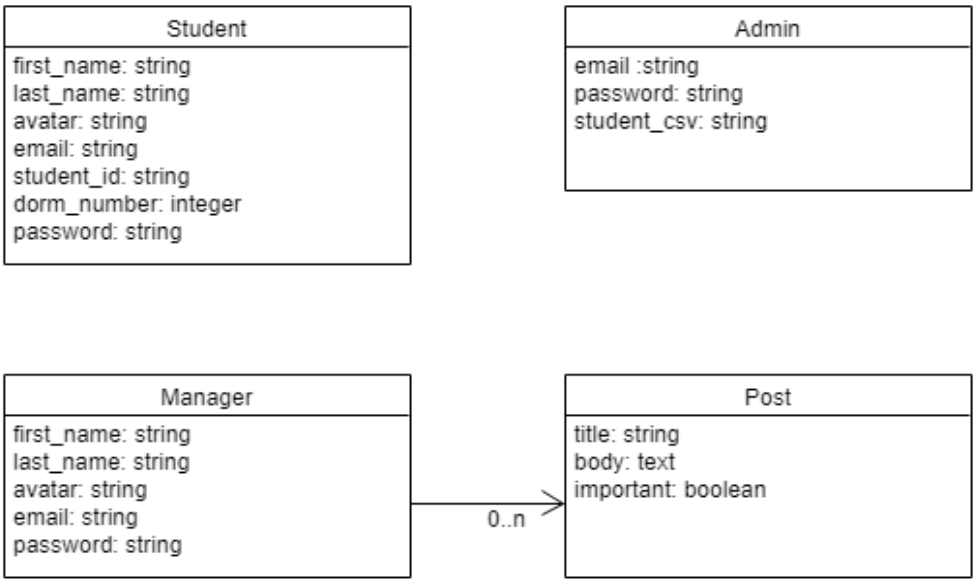


Рис. 3.5. Приклад базової діаграми класів для підсистеми для користувачів разом із підсистемою новин для веб-сервісу [71]

Діаграма класів [71] для підсистеми бронювання кімнат можна описати наступним представленням (рис. 3.6.):

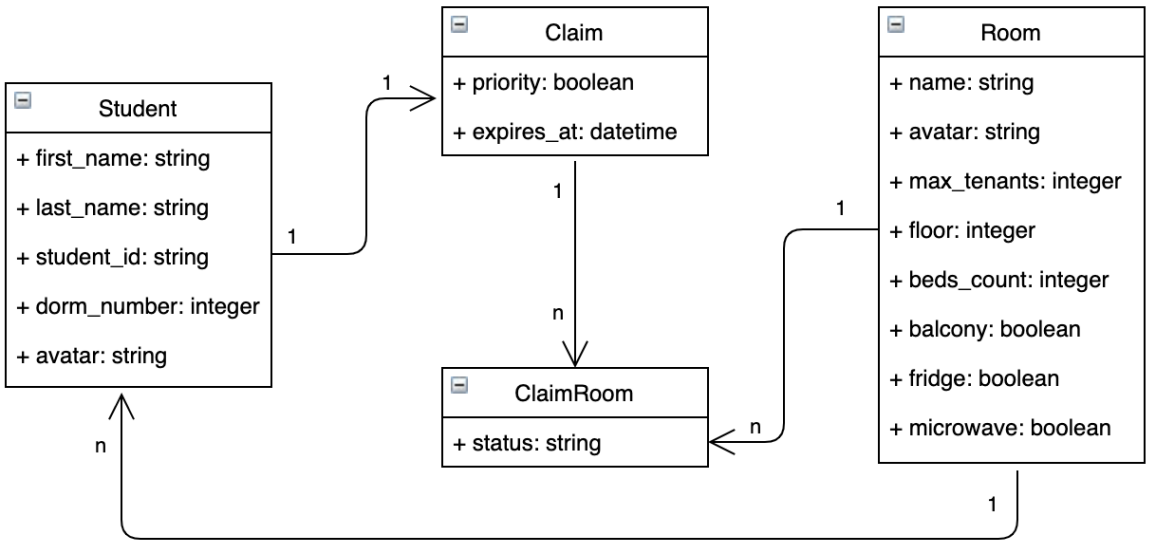
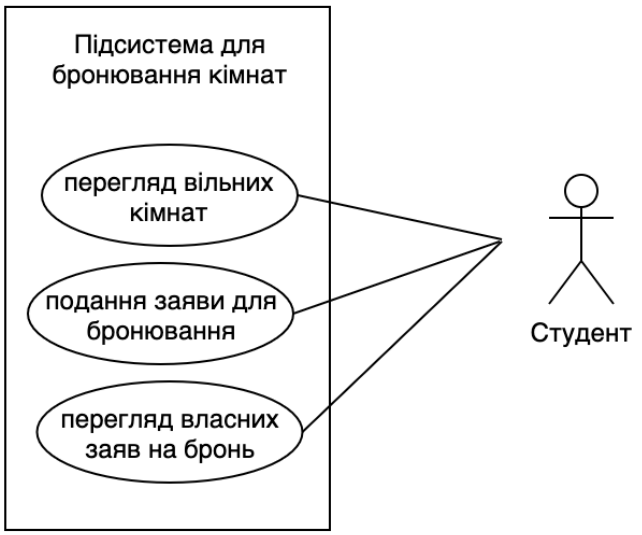
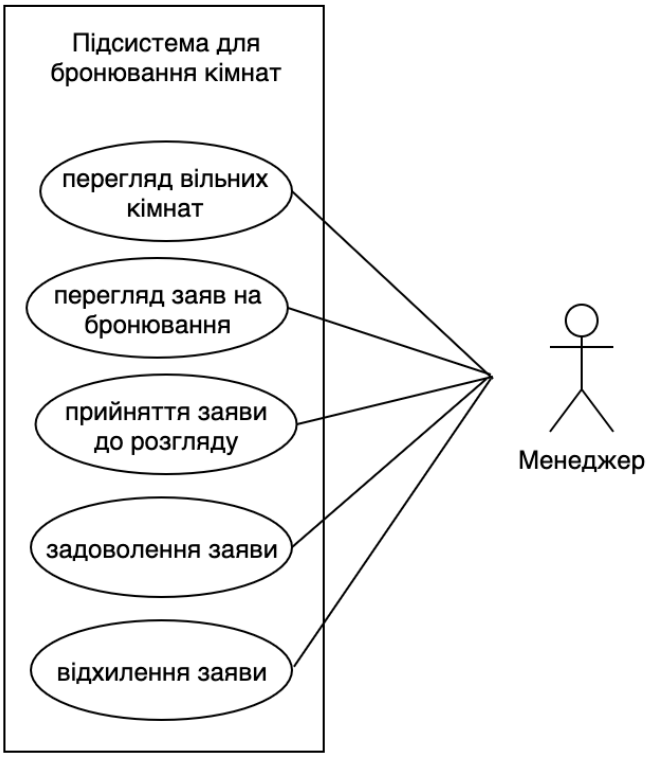


Рис. 3.6. Діаграма класів для підсистеми бронювання кімнат [71]

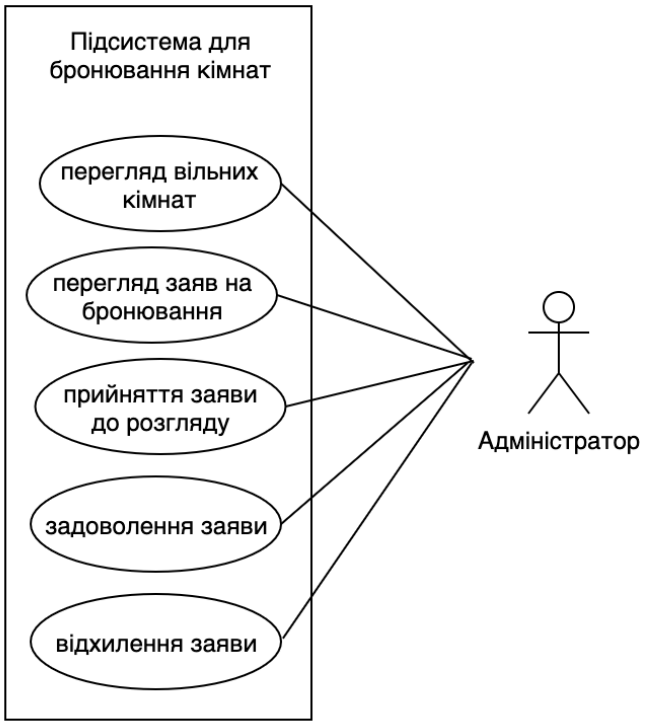
Додатково було розроблено діаграми прецедентів [79] окремо для кожної ролі в межах підсистеми бронювання кімнат, а саме: студента, менеджера та адміністратора (рис. 3.7).


Підсистема для бронювання кімнат
перегляд вільних кімнат
подання заяви для бронювання
перегляд власних заяв на бронь
Студент

а)


Підсистема для бронювання кімнат
перегляд вільних кімнат
перегляд заяв на бронювання
прийняття заяви до розгляду
задоволення заяви
відхилення заяви
Менеджер

б)



в)

Рис. 3.7. Діаграми прецедентів окремо для кожної ролі в межах підсистеми бронювання кімнат, а саме: студента (а), менеджера (б) та адміністратора (в)

3.2. Імплементація автоматизованої системи інформаційної технології

Для того, щоб розпочати розробку веб-сервісу для автономної системи забезпечення діяльності студентського містечка потрібно визначити мови програмування, які корисно було б використати. Під час аналізу різного роду підходів, один з найбільш підходящих під задані вимоги є варіант побудови системи на базі MVC [72] архітектури. MVC (Model-View-Controller) [72] - це архітектурний шаблон, що розділяє веб-систему на три логічні частини: Model(модель) управляє даними та бізнес-логікою, View (представлення) відповідає за відображення інформації користувачу, а Controller (контролер) обробляє запити, змінює модель і визначає, яке представлення показати при конкретному запиті. Це сприяє розділенню відповідальностей, спрощує підтримку коду та робить систему більш гнучкою.

Серед багатьох популярних та гнучких мов програмування, які вже мають написані фреймворки, що підтримують MVC [72] шаблон, було

прийнято рішення про використання мови програмування Ruby [82] та її зручного фреймворку для розробки веб-сервісів RubyonRails(RoR) [81]. Рішення було прийнято через вагомі переваги даного програмного забезпечення, а саме:

- Значна Ruby [82] спільнота;
- Зручність у використанні додаткових бібліотек;
- Наявність чітких стандартів;
- Динамічна типізація;
- Швидкість написання функціоналу мовою програмування Ruby [82];
- Загальна популярність фреймворку RubyonRails [81].

На базі попередніх розрахунків, та використовуючи отримані результати, ми можемо реалізувати структурно-функціональну модель інформаційної технології автоматизації системи діяльності студмістечка (рис. 3.8):

Серед багатьох популярних та гнучких мов програмування, які вже мають написані фреймворки, що підтримують MVC [72] шаблон, було прийнято рішення про використання мови програмування Ruby [82] та її зручного фреймворку для розробки веб-сервісів RubyonRails(RoR) [81]. Рішення було прийнято через вагомі переваги даного програмного забезпечення, а саме:

- Велика та активна Ruby [82] спільнота;
- Велика кількість допоміжних бібліотек та ресурсів;
- Серйозне дотримання стандартів;
- Ruby [82] - динамічно типізована мова;
- Гнучкість мови програмування Ruby [82];
- Фреймворк Ruby on Rails [81] популярний для написання стартапів.

Крок 1.Формування основ по визначенню стану та можливостей реалізації автоматизації управлінських рішень для студентського містечка	аналіз сучасного стану використання інформаційних технологій в Україні та за рубежем
	дослідження інформаційних систем, що використовуються в освітньому середовищі
	з'ясування передумов та вимог, щодо реалізації інформаційної технології автоматизації роботи студентського містечка

Крок 2. Створення моделі та вибір факторів впливу на процес реалізації інформаційної технології автоматизації роботи студентського містечка	аналіз та порівняння інформаційних систем, що використовуються в освітньому середовищі
	виокремлення факторів впливу на процес реалізації інформаційної системи автоматизації діяльності студентського містечка
	побудова моделі факторів впливу на процес реалізації інформаційної системи
	визначення вагових значень виокремлених факторів методом ієрархій
	аналіз отриманих результатів використання методу ієрархій
	проведення оптимізації отриманої моделі факторів впливу на процес реалізації інформаційної системи автоматизації діяльності студмістечка

Крок 3.Проектування, імплементація та безпека інформаційної технології проектування автоматизованої системи забезпечення діяльності студентського містечка	проектування та опис підсистеми
	імплементація системи
	розгортання системи
	визначення можливих вразливостей системи
	створення захисту даних
	Опис загальних безпекових принципів

Нечітка система прогностичного оцінювання якості розглянутого процесу	**Крок 4. Прогнозування якості реалізації інформаційної технології автоматизованої системи діяльності студмістечка**	
	ФАЗИФІКАЦІЯ	
		основи нечіткої логіки для формування якості процесу
		синтез моделей формування інтегрального показника якості процесу
		визначення терм-множин значень лінгвістичних змінних
		розрахунок числових значень функцій належності для відповідного терму
		виконання нормування функцій належності для термів
	ДЕФАЗИФІКАЦІЯ	
		розрахунок значень функцій належності лінгвістичних змінних нищого та вищого рівня
		розрахунок кількісних показників рівня прогнозованої якості реалізації процесу

Рис. 3.8. Структурно-функціональна модель інформаційної технології автоматизації системи діяльності студмістечка

3.2.1. Реалізація підсистеми для користувачів в інформаційній технології

Беручи до уваги той факт, що для веб-сервісу необхідно забезпечити функціонал пов’язаний із організацією роботи користувачів, було використано популярну бібліотеку devise [84], яка є доступною для фреймворку RubyonRails [81]. Вона надає якісну та надійну реалізацію авторизації та аутентифікації користувачів веб-сервісу. Згідно поставлених завдань, необхідно провести розподіл користувачів на ролі, а саме: студент, менеджер та адміністратор. Одним із шляхів, щоб досягти розбиття даної підсистеми на вищевказані ролі, є створення моделі користувача(user) та моделі роль(role), яка в свою чергу потребуватиме асоціації між моделлю user та role, але в такому випадку модель user буде мати багато властивостей, які в певних об’єктах будуть надлишкові. Прикладом може слугувати роль студента: дана сутність повинна мати інформацію про студентський квиток, але тоді необхідно щоб модель user зберігала це у власних атрибутах, проте з іншого боку студентський квиток відсутній у менеджера чи адміністратора, тому в такому випадку цей атрибут буде нерелевантним для інших ролей. Іншим демотивуючим фактором може слугувати ймовірність тривалих опрацювань запитів до таблиці users, через те, що всі користувачі будуть зберігатись в одній таблиці бази даних, а як відомо кількість записів в таблиці прямо впливає на швидкодію запитів. Щоб уникнути вище наведених проблем було використано інший підхід до реалізації ролей, а саме: було створено окремо модель для студента, менеджера та адміністратора. Для цього було використано наступний генератор модель: *railsgeneratedevise `назва моделі`*. Дана команда додатково генерує файли міграцій, які мають інформацію про запропоновані зміни до бази даних, і щоб виконати міграції потрібно скористатись функціоналом фреймворку RubyonRails [81] для виконань міграцій: *railsdb:migrate*. Результатом цієї команди буде створення нових таблиць в базі даних[73] та оновлений файл *db/schema.rb*, який містить дані про схему бази даних. Приклад цього файлу:

```ruby
ActiveRecord::Schema.define(version: 20190517144502) do

  # These are extensions that must be enabled in order to support this database
  enable_extension "plpgsql"

  create_table "admins", force: :cascade do |t|
    t.string   "email",              default: "", null: false
    t.string   "encrypted_password",  default: "", null: false
    t.datetime "remember_created_at"
    t.datetime "created_at",                      null: false
    t.datetime "updated_at",                      null: false
    t.string   "students_csv"
    t.index ["email"], name: "index_admins_on_email", unique: true, using: :btree
  end

  create_table "managers", force: :cascade do |t|
    t.string   "email",                 default: "", null: false
    t.string   "encrypted_password",    default: "", null: false
    t.string   "reset_password_token"
    t.datetime "reset_password_sent_at"
    t.datetime "remember_created_at"
    t.datetime "created_at",                         null: false
    t.datetime "updated_at",                         null: false
    t.string   "avatar"
    t.string   "first_name"
    t.string   "last_name"
    t.index ["email"], name: "index_managers_on_email", unique: true, using: :btree
    t.index ["reset_password_token"], name: "index_managers_on_reset_password_token", unique: true, using: :btree
  end
```

Варто зауважити, що вищезгадані міграції не містять усіх потрібних властивостей для сутностей, тому потрібно згенерувати нові міграції, які міститимуть необхідні властивості. Прикладом однієї із них може слугувати розробка first_name та last_name для студента. Для цього було створено міграцію за допомогою генератора наступною командою: *railsgeneratemigrationAddNameFieldsToStudents*.

Результатом цієї команди буде новостворений файл в директорії *db/migrate*. Після безпосереднього додавання необхідних властивостей, файл міграцій набуде наступного вигляду:

```ruby
class AddNameFieldsToStudents < ActiveRecord::Migration[5.0]
  def change
    add_column :students, :first_name, :string
    add_column :students, :last_name, :string
  end
end
```

Після виконання міграції командою *rails db:migrate* буде згенеровано оновлену схему бази даних:

```
create_table "students", force: :cascade do |t|
  t.string   "email",                  default: "", null: false
  t.string   "encrypted_password",     default: "", null: false
  t.string   "reset_password_token"
  t.datetime "reset_password_sent_at"
  t.datetime "remember_created_at"
  t.datetime "created_at",                          null: false
  t.datetime "updated_at",                          null: false
  t.string   "first_name"
  t.string   "last_name"
  t.string   "student_id"
  t.string   "avatar"
  t.integer  "dorm_number"
  t.index ["email"], name: "index_students_on_email", unique: true, using: :btree
  t.index ["reset_password_token"], name: "index_students_on_reset_password_token", unique: true, using: :btree
end
```

Для розробки сутностей менеджера та адміністратора було виконано аналогічні дії, також було запропоновано наступний графічний інтерфейс для можливості користувачам зареєструватись та увійти у веб-сервіс (рис. 3.9):

Рис. 3.9. Графічний інтерфейс для можливості користувачам зареєструватись та увійти у веб-сервіс

Посилаючись на діаграми прецедентів [81], можна бачити функціонал пов’язаний із фільтрацією студентів, саме тому розроблено наступний модуль, який використовується в моделі студента.

```ruby
module Filterable
  extend ActiveSupport::Concern

  included do
    default_scope { order(:created_at) }
    scope :by_keyword, ->(keyword) { by_first_name(keyword).or(by_last_name(keyword)).or(by_email(keyword)) }
    scope :by_first_name, ->(first_name) { where('first_name LIKE ?', "%#{first_name}%") }
    scope :by_last_name, ->(last_name) { where('last_name LIKE ?', "%#{last_name}%") }
    scope :by_email, ->(email) { where('email LIKE ?', "%#{email}%") }
  end
end
```

Можливість пошуку студента за допомогою ключових слів є представлена на головній сторінці, прикладом може слугувати наступний рисунок (рис. 3.10):

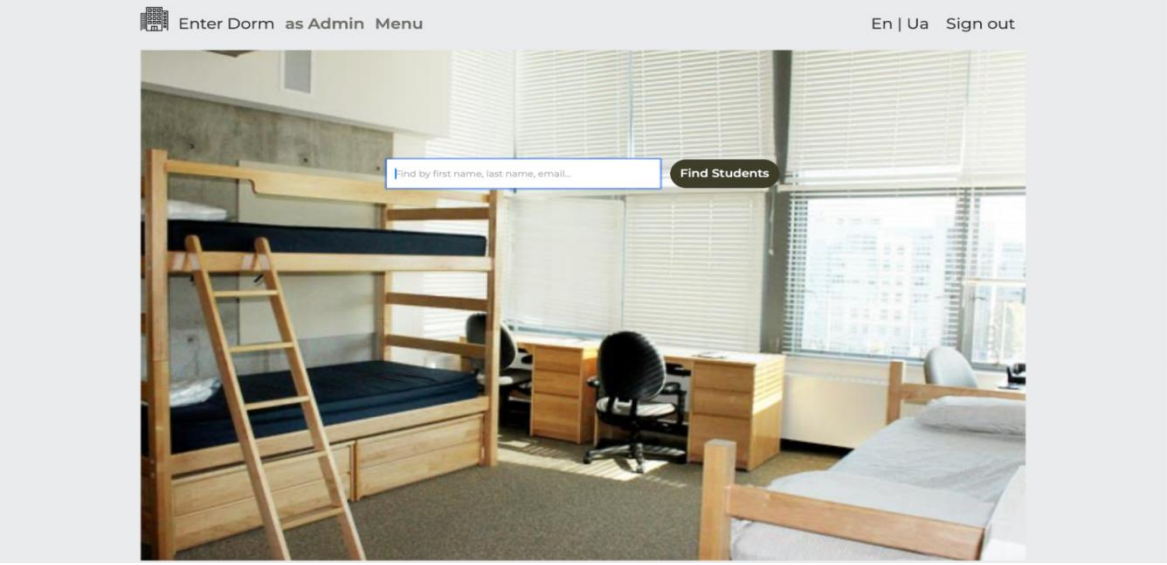


Рис. 3.10. Можливість пошуку студента за допомогою ключових слів є представлена на головній сторінці

Важливим атрибутом сутності студента є власне фото, яке може бути використане для візуальної ідентифікації особи, тому для розробки даного функціоналу було використав так званий uploader(функціонал, що надає фреймворк Ruby on Rails [82]) і окрему бібліотеку під назвою carrierwave [85].

Згідно документації бібліотеки carrierwave [85] було створено сутність, яка несе відповідальність за завантаження та збереження фото студента для профілю, окремо за допомогою методу *extension_whitelist* було обмежено

формати файлів, які можуть бути завантажені як фото профілю користувачів, щоб унеможливити завантаження потенційно небезпечних файлів у систему.

```
class AvatarUploader < CarrierWave::Uploader::Base
  include CarrierWave::RMagick

  storage :file

  def store_dir
    "uploads/#{model.class.to_s.underscore}/#{mounted_as}/#{model.id}"
  end

  def extension_whitelist
    %w(jpg jpeg png)
  end
end
```

Для асоціації конкретного фото і кореспондента, необхідно створити відповідний атрибут *avatar* для сутностей користувачів, а також використати *AvatarUploader* для їхнього коректного завантаження (рис. 3.11).

```
class Student < ApplicationRecord
  include Filterable

  devise :database_authenticatable, :registerable,
         :recoverable, :rememberable, :validatable

  mount_uploader :avatar, AvatarUploader

  STUDENTS_PER_PAGE = 15.freeze

  self.per_page = STUDENTS_PER_PAGE
end
```

Важливо вказати, що синтаксис *mount_uploader :avatar, AvatarUploader* чітко демонструє використання вищезгаданого класу *AvatarUploader* для обробки зображення.

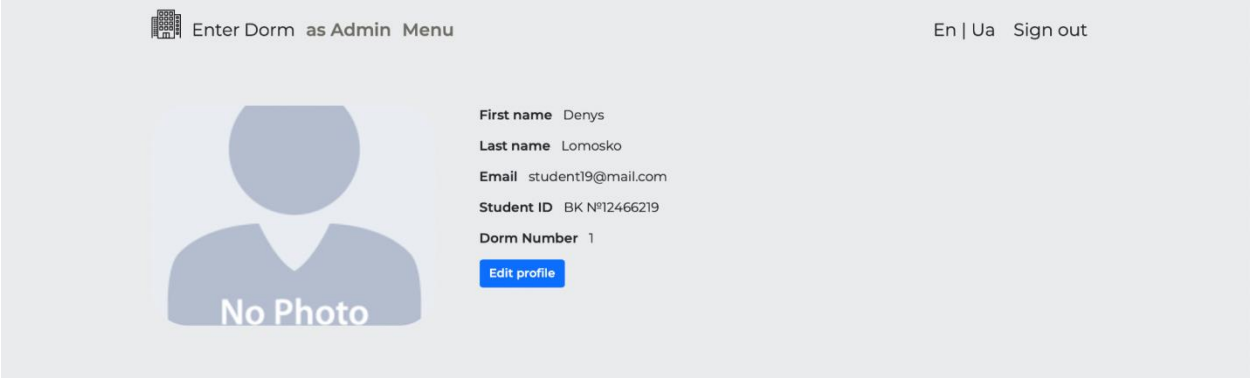


Рис. 3.11. Графічний інтерфейс для асоціації конкретного фото і кореспондента

Додатковим і не менш важливим функціоналом є можливість завантажувати у базу даних сайту дані про багатьох студентів, за задумом дана інформація повинна бути у форматі .csv (comma separated value) (рис. 3.12). Для реалізації даного функціоналу було представлено окремий сервіс, який правильно зберігає дані про студентів використовуючи атрибути із вхідного файлу.

```
require 'csv'

class StudentsCsvService
  PUBLIC_FOLDER = '/public'.freeze
  attr_reader :admin

  def initialize(admin)
    @admin = admin
  end

  def call
    CSV.foreach(file_url, headers: true) do |row|
      Student.create(row.to_hash)
    end
  end

  private

  def file_url
    Rails.root.to_s + PUBLIC_FOLDER + admin.students_csv_url
  end
end
```

Необхідно зазначити, що лише адміністратори веб-сервісу мають змогу це виконувати.

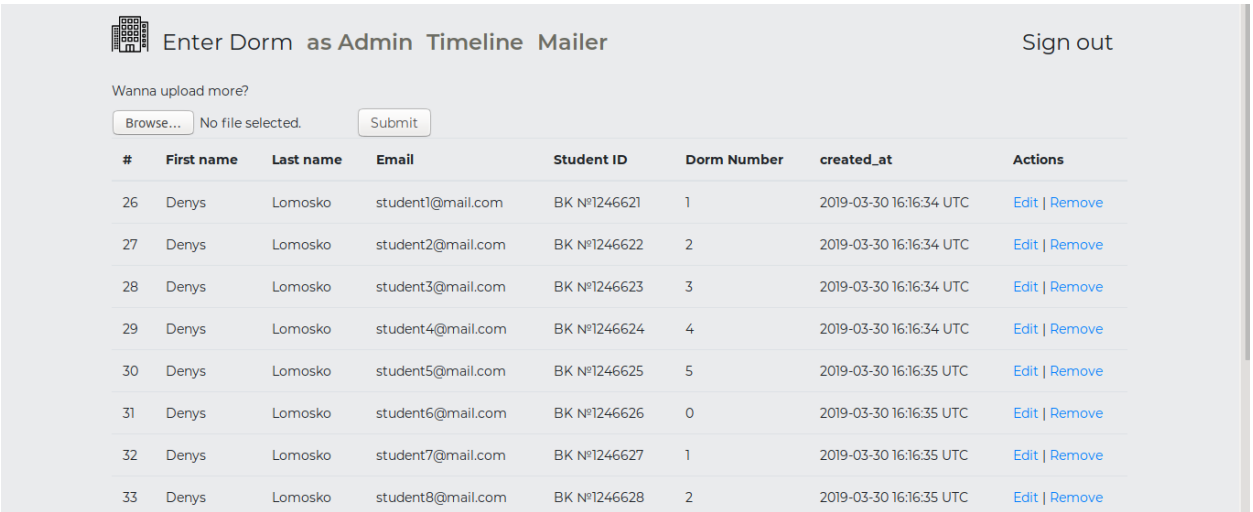


Рис. 3.12. Вигляд бази даних сайту де представлені дані про студентів у форматі .csv

3.2.2. Реалізація підсистеми для новин в інформаційній технології

Важко переоцінити значимість даної підсистеми для новин, адже основною її перевагою слугує якісне поширення інформації серед студентів. Дана підсистема ставить собі за мету скоротити використання та розповсюдження паперових оголошень, які мають обмежену видимість, тобто студенти можуть випадково не прочитати оголошення і тим самим не бути проінформованими про якусь важливу новину. Для того, щоб імплементувати дану підсистему, було створено сутність Post та додано необхідну міграцію для створення таблиці в базі даних [73]. Варто вказати наступні властивості в міграції: заголовок новини, тіло новини, інформація про те, чи новина є особливо важливою та додатково необхідно додати ідентифікатор менеджера, який створив новину, щоб мати можливість отримати список новин асоційованих з певним менеджером.

```
class CreatePosts < ActiveRecord::Migration[5.0]
  def change
    create_table :posts do |t|
      t.string :title
      t.text :body
      t.boolean :important
      t.integer :manager_id

      t.timestamps
    end
  end
end
```

Посилаючись на існуючу діаграму прецедентів [79] для ролі менеджера, потрібно реалізувати можливість створення новин менеджером. Важливою є асоціація між сутністю Manager та Post таким чином, що менеджер може писати багато новин і конкретна новина може бути асоційованою лише з відповідним менеджером. Для цього було використано наступні синтаксичні конструкції фреймворку Ruby on Rails [80]: *has_many :posts* до моделі Manager та *belongs_to :manager* до сутності Post. Конструкція *belongs_to :manager* означає, що модель Post має зовнішній ключ на сутність Manager, в даному випадку це *manager_id*, що вказано на рисунку вище. Окрім того, було додано базові валідації для властивостей *title* та *body* в моделі Post, щоб гарантувати присутність даних атрибутів. Враховуючи вищенаведені зміни, модель Post набула наступного вигляду.

```ruby
class Post < ApplicationRecord
  validates :title, presence: true
  validates :body, presence: true

  belongs_to :manager

  default_scope { order(:created_at) }
  scope :important, -> { where(important: true) }

  POSTS_PER_PAGE = 15.freeze

  self.per_page = POSTS_PER_PAGE
end
```

Додатково було імплементовано графічне представлення сторінки менеджера, яка містить інформацію про користувача, можливість перейти на сторінку редагування профілю, посилання на сторінку написання новини, а також список наявних новин, які були створені в минулому, з можливістю їх модерації або видалення (рис. 3.13).

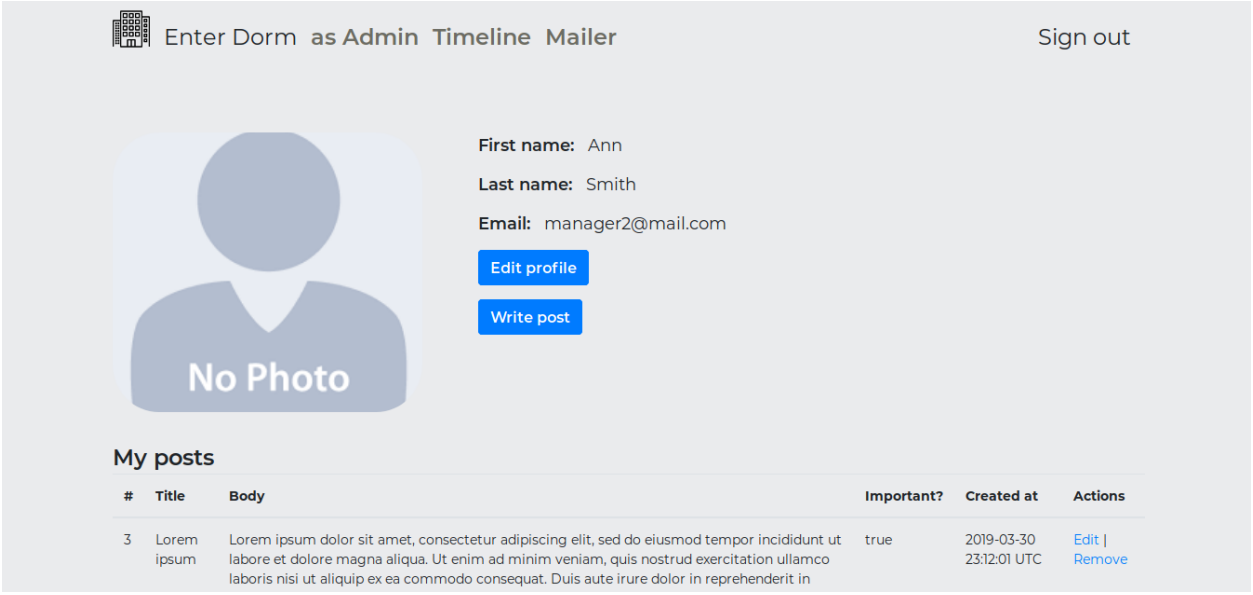


Рис. 3.13. Графічне представлення сторінки менеджера, яка містить інформацію про користувача

Згідно архітектурного шаблону MVC [72], було розроблено контролер для обробки запитів, що стосуються опрацювання новин, а прикладом такого контролера може слугувати наступний код програми:

```
class PostsController < ApplicationController
  before_action :find_post, except: %i[index new create]

  def index
    @posts = Post.all
  end

  def new
    @post = Post.new
  end

  def edit
  end

  def create
    @post = Post.new(post_params.merge(manager_id: params[:manager_id]))
    if @post.save
      redirect_to manager_path(params[:manager_id])
    else
      render :new
    end
  end

  def update
    @post.assign_attributes(post_params)
    if @post.save
      redirect_to manager_path(params[:manager_id])
    else
```

```
      render :edit
    end
  end

  def destroy
    @post.destroy
    redirect_back(fallback_location: root_path)
  end

  private

  def find_post
    @post ||= Post.find(params[:id])
  end

  def post_params
    params.require(:post).permit(:title, :body, :important)
  end
end
```

3.2.3. Реалізація підсистеми для листування в інформаційній технології

Важливо згадати, що частина студентів може нехтувати користуванням даного веб-сервісу, тому варто розробити механізм поширення новин для такої неактивної множини користувачів. Логічним варіантом слугує надсилання персоналізованих листів на електронну скриньку, адже під час реєстрації студенти повинні вказати електронну скриньку, тому при потребі з бази даних можна отримати контакти усіх студентів. Окремо розглянуто можливість надсилання листів групі студентів, для того, щоб оптимізувати швидкість процесу. Фреймворк Ruby on Rails [81] надає бібліотеку ActionMailer [83], яка істотно полегшує роботу з надсиланням електронних листів програмними засобами.

```ruby
class NotificationsMailer < ApplicationMailer
  def notify(email, subject, body)
    @student = Student.find_by(email: email)
    @body = body
    @subject = subject
    mail(to: @student.email, subject: @subject)
  end
end
```

Бібліотека ActionMailer [83] потребує коректних налаштувань в залежності від робочого середовища, прикладами таких файлів можуть слугувати:
config/environment/staging.rb, config/environment/production.rb.

Очевидно, що при формуванні листа потрібно додати адресу відправника, а за замовчування вона налаштовується в класі ApplicationMailer. Варто зазначити, що новостворений клас NotificationsMailer наслідується від ApplicationMailer [83], тому адреса відправника буде видима і в дочірньому класі.

```ruby
class ApplicationMailer < ActionMailer::Base
  default from: 'info@enterdorm.com'
  layout 'mailer'
end
```

Зрозуміло, що кожен лист повинен мати якесь змістове наповнення, саме тому було створено окремий файл, який буде містити коректні вихідні дані та відповідне графічне представлення, прикладом даного файлу може слугувати app/view/notifications_mailer/notify.html.haml. Для того, щоб даний файл автоматично був врахованим під час роботи програми, необхідно надати коректну назву: метод класу мейлера має співпадати з назвою представлення, тобто view. В такому випадку фреймворк Ruby on Rails [82] коректно буде працювати, використовуючи сутність NotificationsMailer.

Окремо було розроблено графічний інтерфейс для форми надсилання листів з можливістю вказати тему листа, електронну скриньку студента, а також бажаний вміст листа (рис. 3.14).

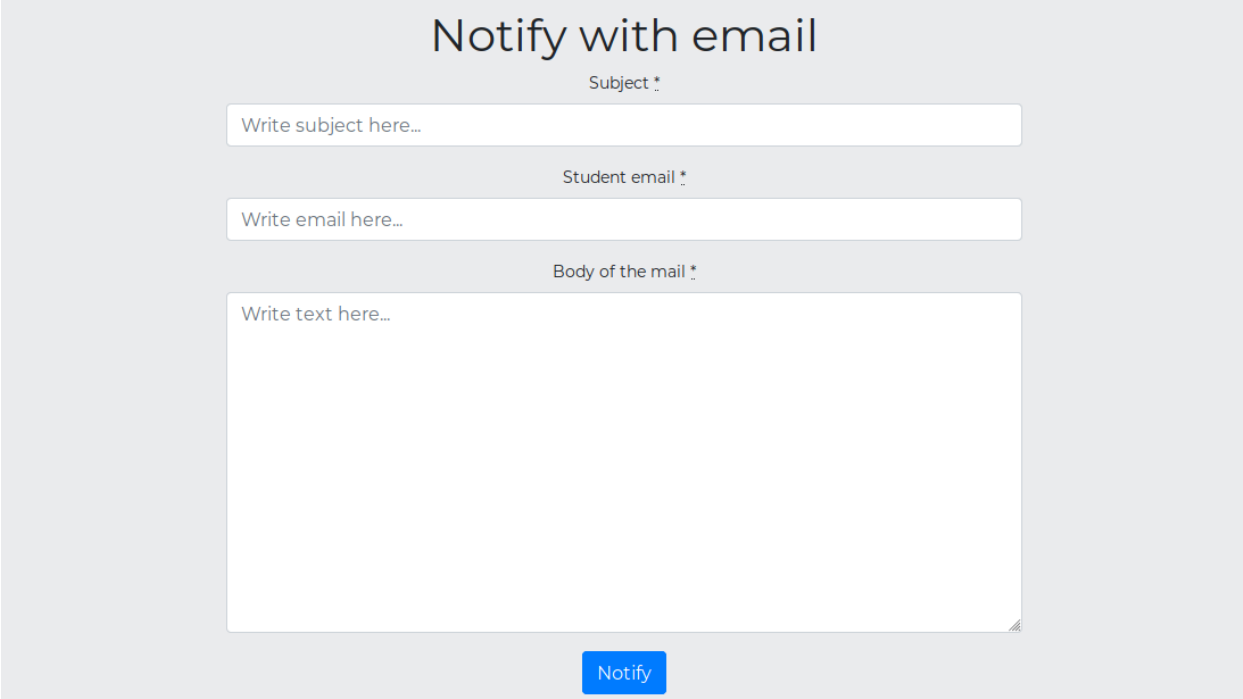


Рис. 3.14. Графічний інтерфейс для форми надсилання листів

3.2.4. Реалізація підсистеми бронювання кімнат в інформаційній технології

Згідно із заданих завдань дисертаційної роботи, було розпочато імплементацію підсистеми для бронювання кімнат мешканцями студентського містечка. Дана підсистема вимагає створення та налаштування відповідних сутностей: Room, Claim та ClaimRoom. Варто згадати, що відповідні атрибути та асоціації між сутностями були описані в діаграмі класів [71]. Важливо, що студент не може проживати в декількох кімнатах одночасно, саме тому сутність Student має асоціацію на сутність Room типу один до багатьох. З іншого боку більшість кімнат є багатомісними, тому необхідно врахувати можливість отримання списку студентів базуючись на конкретній кімнаті. Окремо було створено модель Claim для створення безпосередньої заявки, додатково дана модель асоційована із сутністю Room таким чином, щоб була змога одночасно надсилати заявки на декілька кімнат, що збільшує шанси зміни кімнати. Такого роду зв’язок потребує таблицю ClaimRoom, яка буде зберігати ідентифікатори конкретної заявки та відповідної кімнати.

Додатково було імплементовано валідацію атрибутів відповідних сутностей. Прикладами перевірки вхідних даних в сутності Room можуть слугувати наступні дані: поверх(floor) повинен бути додатнім числом, назва кімнати(name) обов'язково повинна бути вказаною, максимальна кількість мешканців (max_tenants) не має бути від'ємною, кількість ліжок(beds_count) має бути додатнім число і т.д. Сутність Claim має асоціацію із сутністю Student типу один до одного. Окремо було визначено коректні статуси заявок для зміни кімнати: нова заявка (new), в розгляді(in_progress), затверджено(approved), відхилено(rejected). Додатково було розроблено перевірку ідентичних заяв, тобто якщо заява конкретного студента для конкретної кімнати вже існує.

```
class Claim < ApplicationRecord
  belongs_to :student
  has_many :claim_rooms
  has_many :rooms, through: :claim_rooms

  validate :belongs_to_one_student, on: :create

  private

  def belongs_to_one_student
    return if Claim.where(student_id: student_id).none?

    errors.add(:student_id, 'Already exist')
  end
end
```

Для детального розуміння варто розглянути приклад розробки моделі ClaimRoom, яка надає наступні можливості:

- вказувати поточні статуси поданої заявки (ALLOWED_STATUES);
- заборони використання сторонніх статусів;
- фільтрувати заявки згідно заданого статусу;
- змінювати статус заявки відповідно.

```
class ClaimRoom < ApplicationRecord
  ALLOWED_STATUSES = %w[new in_progress approved rejected].freeze

  belongs_to :claim
  belongs_to :room

  validates :status, inclusion: { in: ALLOWED_STATUSES }
  validates :room_id, uniqueness: { scope: :claim_id }

  scope :new_claims, -> { where(status: 'new') }
  scope :in_progress, -> { where(status: 'in_progress') }
  scope :approved, -> { where(status: 'approved') }
  scope :rejected, -> { where(status: 'rejected') }

  POSTS_PER_PAGE = 15.freeze

  self.per_page = POSTS_PER_PAGE

  def mark_in_progress!
    update(status: 'in_progress')
  end

  def approve!
    update(status: 'approved')

    claim.student.update(room_id: room.id)
  end

  def reject!
    update(status: 'rejected')
  end
end
```

Додатково в моделі Room було розроблено запит до бази даних [73], який повертає кімнати з кількістю студентів меншою за заявлену, тобто кімнати, які потенційно можуть бути місцем проживання для студентів.

```ruby
class Room < ApplicationRecord
  mount_uploader :avatar, AvatarUploader

  has_many :claim_rooms
  has_many :claims, through: :claim_rooms
  has_many :students

  validates :name, presence: true
  validates :max_tenants, numericality: { greater_than: 0 }, presence: true
  validates :beds_count, numericality: { greater_than: 0 }, presence: true
  validates :floor, numericality: { greater_than: 0 }, presence: true

  scope :available, -> {
    joins('LEFT OUTER JOIN students ON students.room_id = rooms.id')
      .group(:id, :room_id)
      .having('COUNT(students.room_id) < rooms.max_tenants')
      .order(created_at: :desc)
  }

  ROOMS_PER_PAGE = 15.freeze

  self.per_page = ROOMS_PER_PAGE

  def students_living_count
    students.size
  end

  def is_available_for_tenants?
    students.size < max_tenants
  end
end
```

Прикладом графічного інтерфейсу для ергономічного відображення з можливістю фільтрування існуючих кімнат студентського містечка може слугувати наступне представлення (рис. 3.15):

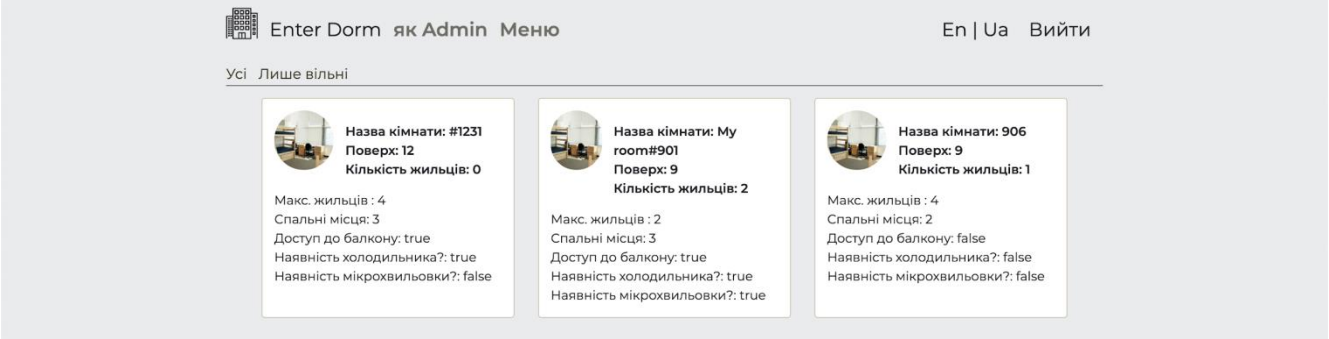


Рис. 3.15. Графічний інтерфейс для ергономічного відображення з можливістю фільтрування існуючих кімнат студентського містечка

В ході імплементації системи, також було запропоновано візуальне представлення конкретної кімнати, яке включає інформацію про мешканців,

функціонал, залежний від типу користувача, а також важливий для проживання опис кімнати (рис. 3.16).

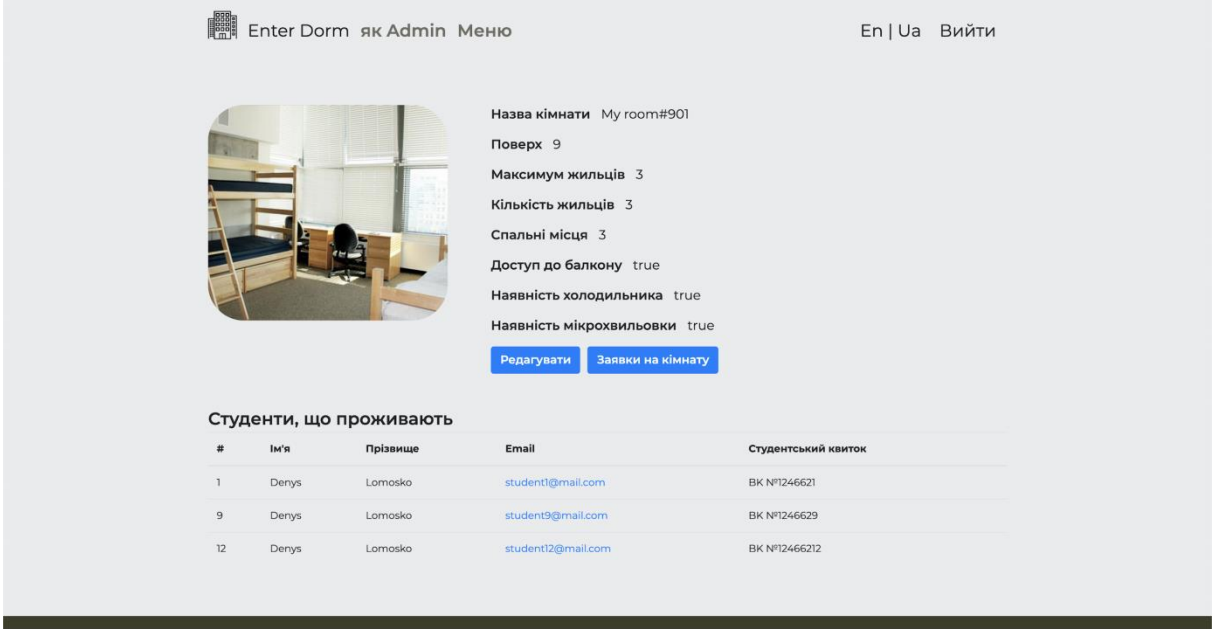


Рис. 3.16. Візуальне представлення конкретної кімнати

Варто зазначити, що графічне представлення значною мірою залежить від присвоєної ролі користувача, тому якщо на сторінку кімнати увійти як студент, то замість дій «Редагувати» та «Заявки на кімнату» буде доступна кнопка «Подати заявку», якщо виконуються умови доступності її до заселення і це не є поточна кімната студента. Для ролей менеджера та адміністратора студентського містечка, було надано можливість переглядати існуючі заявки для конкретної кімнати, а саме графічне представлення було запропоноване в такому вигляді (рис. 3.17):

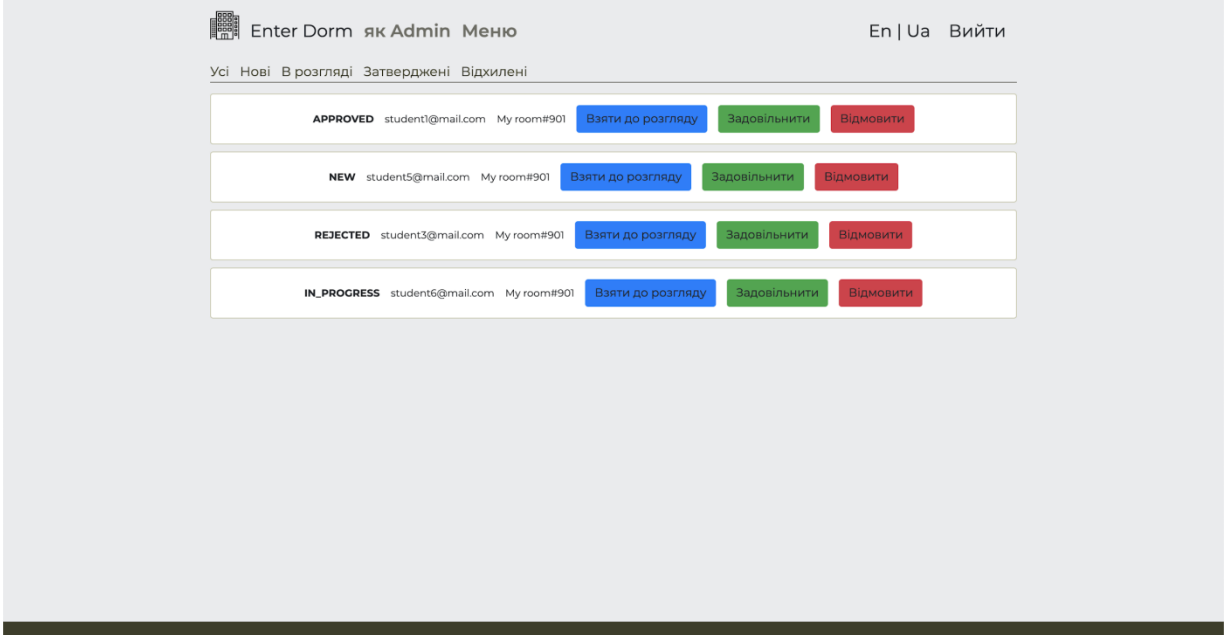


Рис. 3.17. Можливість переглядати менеджерами та адміністраторами існуючі заявки для конкретної кімнати

Для ергономічності було розроблено можливість фільтрувати заявки за існуючими статусами, а для самої реалізації підсистеми бронювання кімнат було розроблено кілька контролерів, ознайомитись із прикладом одного з них можна на наступному рисунку.

```
class ClaimsController < ApplicationController
  before_action :authenticate_student!, only: %i[create]
  before_action :authenticate_manager, only: %i[index approve reject mark_in_progress]
  before_action :find_room
  before_action :find_claim_room, only: %i[approve reject mark_in_progress]

  # GET (/:locale)/rooms/:room_id/claims
  def index
    @claim_rooms = filter_claim_rooms(params[:filter]).page(params[:page])
  end

  # POST (/:locale)/rooms/:room_id/claims
  def create
    if current_student.claim_for_room(@room.id)
      flash[:success] = t('flash.claimed')
    else
      flash[:notice] = t('flash.already_claimed')
    end

    redirect_to room_path(@room)
  end
```

```ruby
# POST (/:locale)/rooms/:room_id/claims/:id/mark_in_progress
def mark_in_progress
  @claim_room.mark_in_progress!

  redirect_to room_claims_path(@room)
end

# POST (/:locale)/rooms/:room_id/claims/:id/accept
def approve
  @claim_room.approve!

  redirect_to room_claims_path(@room)
end

# POST (/:locale)/rooms/:room_id/claims/:id/reject
def reject
  @claim_room.reject!

  redirect_to room_claims_path(@room)
end
```

Згідно із документацією фреймворку RubyonRails [82], публічні методи контролера називаються діями(actions), а доступ до них за допомогою URL-адрес [90] налаштовується у файлі routes.rb. Приклад фільтрування заявок на зміну кімнат відповідно до статусу:

```ruby
private

def filter_claim_rooms(filter_param)
  @claim_rooms = @room.claim_rooms

  case filter_param
  when 'new'
    @claim_rooms.new_claims
  when 'in_progress'
    @claim_rooms.in_progress
  when 'approved'
    @claim_rooms.approved
  when 'rejected'
    @claim_rooms.rejected
  else
    @claim_rooms
  end
end
```

Окремої уваги заслуговує метод ‘before_action’, який згідно документації виконується перед кожною дією(action). Зручність даного методу полягає в тому, що такий підхід забирає дублювання коду у випадку, якщо потрібно виконати

якусь схожу функцію для кількох дій (actions). Прикладом може слугувати наступний виклик:

```
before_action :find_claim_room, only: %i[approve reject mark_in_progress]
```

В такому випадку функція ‘find_claim_room’ буде викликатись перед наступними діями (actions), як-от: ‘approve’, ‘reject’, ‘mark_in_progress’, а сам метод ‘find_claim_room’ виглядає таким чином:

```
  def find_room
    @room ||= Room.find(params[:room_id])
  end

  def find_claim_room
    @claim_room ||= ClaimRoom.find_by(claim_id: params[:id], room_id: params[:room_id])
  end
end
```

3.2.5. Інтеграція фонових задач

Значною проблемою в роботі сервера може слугувати велика нагрузка, яка зумовлена виконанням великої кількості завдань одночасно. Саме тому було використано бібліотеку sidekiq [88], яка працює на базі Redis [82]. Дана бібліотека надає можливість використання фонових задач (backgroundprocessing) [90], які свою чергу можуть виконуватись паралельно в декількох потоках, що істотно покращує загальну швидкодію веб-сервісу. Крім того, для зручного управління та аналізу фонових задач, sidekiq [87] надає ергономічну панель (рис. 3.18).

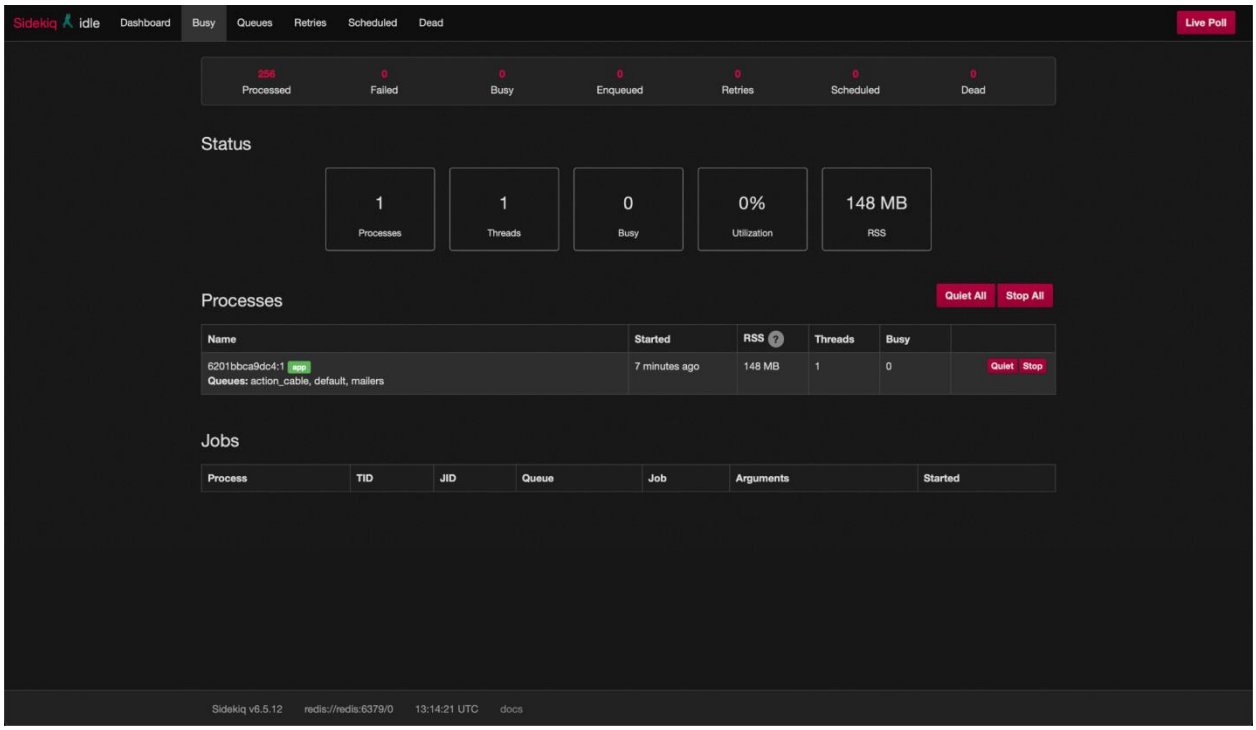


Рис. 3.18. Ергономічна панель управління та аналізу фонових задач

Згідно документації бібліотеки sidekiq [87], фонова задача може мати різні статуси: скасована, запланована, в повторі(невдале виконання), в черзі, в процесі, завершено невдало, завершено вдало. Важливим фактором є надання пріоритету певних задач над іншим, саме тому за допомогою sidekiq [87] можна налаштувати пріоритетну чергу у конфігураційному файлі ‘sidekiq.yml’:

```
:concurrency: 1
:pidfile: tmp/pids/sidekiq.pid
:logfile: log/sidekiq.log
staging:
  :concurrency: 2
production:
  :concurrency: 10
:queues:
  - action_cable
  - default
  - mailers
```

Варто розглянути властивість queues, яка вказує пріоритетність черг у випадку, якщо черг є декілька. Згідно даного файлу налаштувань, можна бачити, що черга action_cable має найвищий пріоритет і буде виконуватись в першу чергу, а листи будуть надсилатись з найменшим пріоритетом в черзі mailers.

3.2.6. Локалізація сайту інформаційної технології забезпечення діяльності студентського містечка

Налаштування декількох доступних мов інтерфейсу у фреймворку Ruby on Rails [82] може бути реалізовано за допомогою бібліотеки I18n [88]. Варто зазначити що тексти з перекладом для зручності містяться в одній директорії, а саме: config/locales/. Згідно наперед визначених завдань, необхідно передбачити доступ до веб-сервісу англійською та українською мовами, тому було створено дві окремі директорії en/ та ua/ з відповідними файлами перекладу. Файл ініціалізації translation.rb буде виконуватись під час старту веб-сервісу і буде завантажувати шляхи до файлів з перекладами, доступні локалі, вказана локаль за замовчуванням.

```
# frozen_string_literal: true

# All translations from config/locales/*.yml are auto loaded.
Rails.application.config.i18n.load_path += Dir[Rails.root.join('config', 'locales', '**', '*.yml')]

Rails.application.config.i18n.available_locales = %i[en ua]

Rails.application.config.i18n.default_locale = :en

Rails.application.config.locale_codes_mapping = { en: 'en', ua: 'ua' }
```

Порівняємо config/locales/ua/common.yml та config/locales/en/common.yml:

```
ua:
  common:
    not_specified: Не вказано
    update: Оновити
    edit: Редагувати
    remove: Видалити
    actions: Дії
    next: Наступна сторінка
    previous: Попередня сторінка
    submit: Зберегти
```

```
en:
  common:
    not_specified: Not specified
    update: Update
    edit: Edit
    remove: Remove
    actions: Actions
    next: Next page
    previous: Previous page
    submit: Submit
```

Згідно вищевказаного файлу, текст ‘‘Видалити’’ буде доступний через функцію I18n.t (‘common.remove’), де t – translate (перекласти), причому ключ ‘en’ чи ‘ua’ явно не вказується, щоб динамічно вибирати потрібний переклад базуючись мові інтерфейсу користувача. Для прикладу розглянемо даний файл, який містить переклади:

```
- if @posts.present?
  %table.table
    %thead
      %tr
        %th{ scope: 'col' } #
        %th{ scope: 'col' }= t('admins.posts.owner')
        %th{ scope: 'col' }= t('admins.posts.title')
        %th{ scope: 'col' }= t('admins.posts.body')
        %th{ scope: 'col' }= t('admins.posts.important')
        %th{ scope: 'col' }= t('admins.posts.created_at')
        %th{ scope: 'col' }= t('admins.common.actions')
    %tbody
      - @posts.each do |post|
        %tr
          %td{ scope: 'row' }= post.id
          %td{ scope: 'row' }= link_to post.manager.email, manager_path(post.manager)
          %td{ scope: 'row' }= post.title
          %td{ scope: 'row' }= post.body
          %td{ scope: 'row' }= post.important?
          %td{ scope: 'row' }= post.created_at
          %td{ scope: 'row' }
            = link_to t('admins.common.edit'), edit_manager_post_path(post.manager, post)
            |
            = link_to t('admins.common.remove'),
                      manager_post_path(post.manager, post),
                      method: :delete,
                      data: { confirm: t('admins.common.sure') }
  = will_paginate @posts, previous_label: t('common.previous'), next_label: t('common.next')
- else
  %h1.mt-5.text-center= t('admins.posts.no_posts')
```

Для зручності є опція вибору мови інтерфейсу в шапці веб-сервісу, це досягається елементом ‘En | Ua’, де вибір En чи Ua перекладає графічний інтерфейс відповідно (рис. 3.19).

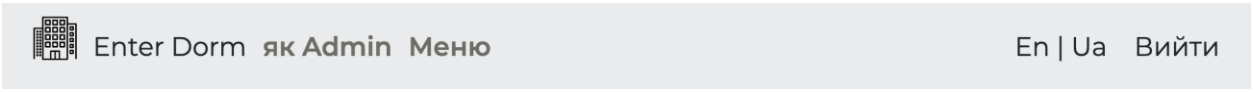


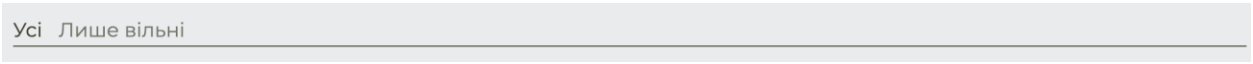
Рис. 3.19. Опція вибору мови інтерфейсу

З технічної точки зору переклад веб-сервісу досягається зміною визначених шляхів за допомогою відповідного префіксу локалі. Для англійської мови стрічка запиту буде містити ‘en’ і ‘ua’ для української відповідно: ‘/ua/managers/1/posts/new’. Даний підхід було реалізовано таким чином:

```
scope "(:locale)", :locale => /en|ua/ do
    …
end
```

3.2.7. Фільтрація

Для ергономічно відображення відповідних сутностей було розроблено підтримку фільтрації, що дозволяє переглядати лише відібрані за певною властивістю елементи. Прикладом може слугувати веб-сторінка для відображення існуючих кімнат студентського містечка, адже до неї було додано важливий фільтр, який повертає лише вільні для заселення студентами кімнати. Приклад графічного представлення фільтра для вибору кімнат:



Наступний запит до бази даних реалізує отримання вільних для заселення кімнат:

```
scope :available, -> {
  joins('LEFT OUTER JOIN students ON students.room_id = rooms.id')
    .group(:id, :room_id)
    .having('COUNT(students.room_id) < rooms.max_tenants')
    .order(created_at: :desc)
}
```

Також було додано фільтрацію і для графічного представлення заявок на бронювання вільних кімнат (рис. 3.20). Прикладом графічного представлення, де зверху є панель із доступними фільтрами, може слугувати наступний рисунок.

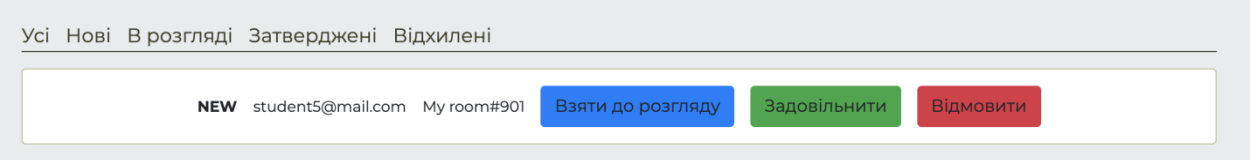


Рис. 3.20. Фільтрація для графічного представлення заявок на бронювання вільних кімнат

Окремо було додано наступні запити до бази даних, щоб відфільтрувати заявки для бронювання згідно заданого статусу:

```
scope :new_claims, -> { where(status: 'new') }
scope :in_progress, -> { where(status: 'in_progress') }
scope :approved, -> { where(status: 'approved') }
scope :rejected, -> { where(status: 'rejected') }
```

Додатково реалізовано функціонал пошуку студента за ключовим словом, яке виконує пошук за іменем, прізвищем та електронною поштою студента. Даний функціонал винесено в окремий модуль, який згодом використано в сутності студента за допомогою наступної синтаксичної конструкції ‘includeFilterable’.

```
module Filterable
  extend ActiveSupport::Concern

  included do
    default_scope { order(:created_at) }
    scope :by_keyword, ->(keyword) { by_first_name(keyword).or(by_last_name(keyword)).or(by_email(keyword)) }
    scope :by_first_name, ->(first_name) { where('first_name LIKE ?', "%#{first_name}%") }
    scope :by_last_name, ->(last_name) { where('last_name LIKE ?', "%#{last_name}%") }
    scope :by_email, ->(email) { where('email LIKE ?', "%#{email}%") }
  end
end
```

3.2.8. Розробка чату

Для того, щоб користувачі даного веб-сервісу мали можливість комунікувати між собою та обмінюватись повідомленнями, імплементовано онлайн-чат, чий функціонал буде працювати за допомогою протоколу WebSocket (рис. 3.21) [96]. Даний протокол забезпечує відкритий двосторонній зв’язок між клієнтом та сервером, який підтримує комунікацію в реальному часі, а фреймворк Ruby on Rails [82] має вбудовану бібліотеку ActionCable [97], яка підтримує вищевказаний тип комунікації.

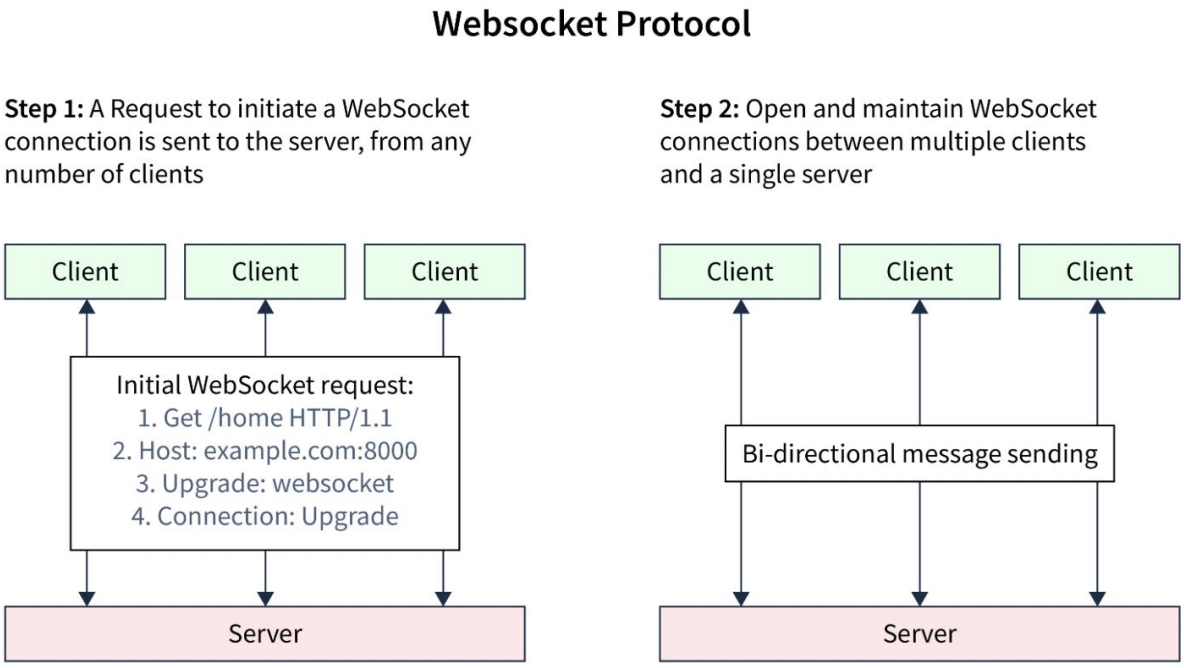


Рис. 3.21. Двосторонній зв’язок між клієнтом та сервером за допомогою WebSocketProtocol

Спочатку було налаштовано клієнтську частину чату, яка буде відповідати за встановлення початкового зв’язку та комунікацію з сервером, а також відображення даних на сторінці чату.

```
App.chat = App.cable.subscriptions.create({ channel: 'ChatChannel' }, {
  connected() {
    // Called when the subscription is ready for use on the server
  },

  disconnected() {
    // Called when the subscription has been terminated by the server
  },

  received(data) {
    document.getElementById('messages').insertAdjacentHTML('beforeend', data['content']);
  },

  message(content) {
    this.perform('message', { content: content });
  }
})
```

На рахунок безпосереднього опрацювання відправлень повідомлень в чат користувачами веб-сервісу, було розроблено окрему компоненту ‘App.ChatPage’. Можна бачити, що при натиску клавіші ‘Enter’ виконується функція, яка надсилає введені користувачем дані через протокол WebSocket [95] за допомогою функції ‘App.chat’, яка була визначена рисунком вище.

```
class App.ChatPage extends App.UIComponent
  defaultContainerID: document

  _initVariables: ->
    @$messagesContainer = $('#messages')

  _initUI: =>
    @scrollToMessagesBottom()

  _initEventListeners: ->
    $('body').on 'keypress', '[data-behavior~=chat_speaker]', @onMessageSend

  onMessageSend: (event) =>
    # return if not ENTER or received an empty message
    return if (event.keyCode != 13) || (event.target.value.length < 1)

    App.chat.message(event.target.value)
    event.target.value = ''
    event.preventDefault()

    @scrollToMessagesBottom()

  scrollToMessagesBottom: ->
    # scroll to bottom of the messages container
    @$messagesContainer.animate({
      scrollTop: @$messagesContainer.prop('scrollHeight')
    }, 500);
```

Слідвідмітити, що для коректної роботи чату потрібно розробити на стороні сервера необхідний функіонал, як-от: можливість підписки на канал комунікації, надсилання повідомлень в потрібний канал комунікації.

```ruby
class ChatChannel < ApplicationCable::Channel
  def subscribed
    stream_from 'chat_channel'
  end

  def unsubscribed
    # Any cleanup needed when channel is unsubscribed
  end

  def message(data)
    Message.create!(content: data['content'])
  end
end
```

Один з варіантів графічного представлення чату для користувачів студентського містечка може слугувати наступний приклад (рис. 3.22).

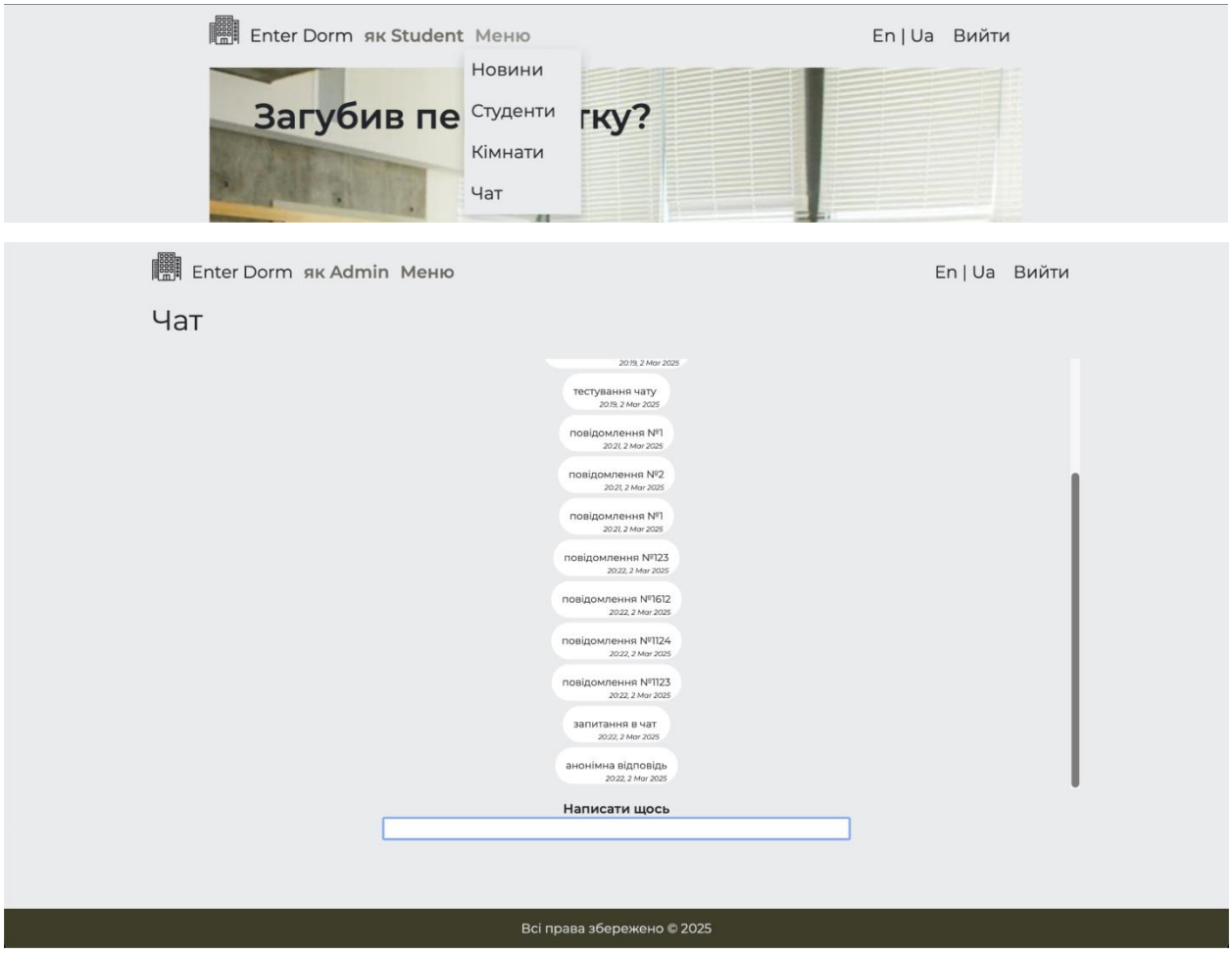


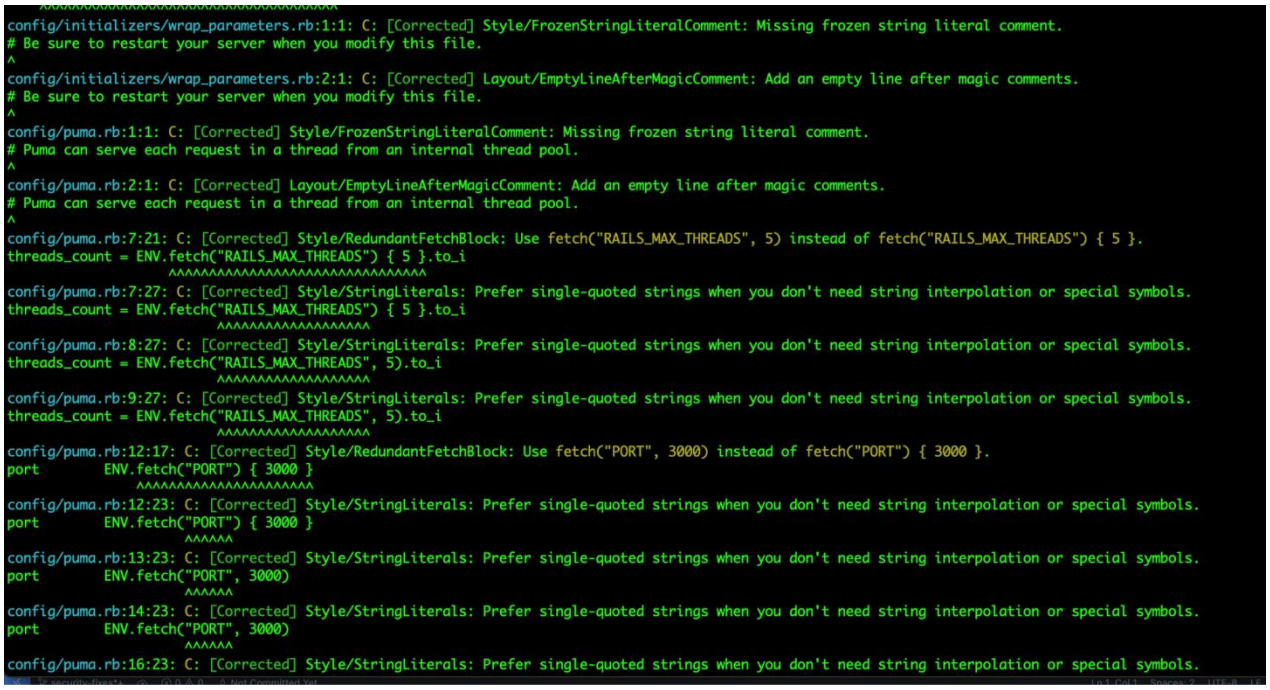
Рис. 3.22. Графічне представлення чату для користувачів студентського містечка

3.2.9. Налаштування статичного аналізатора коду

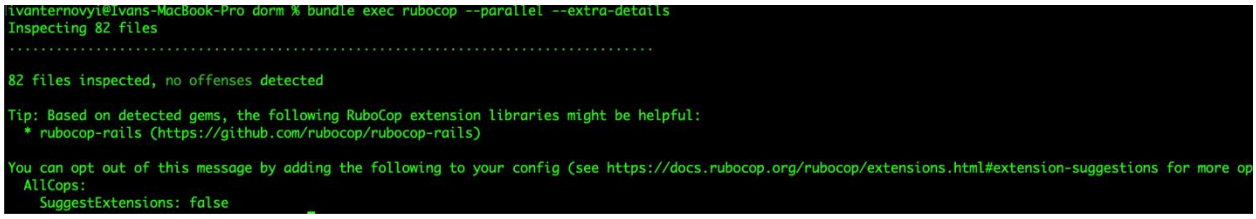
Під час розробки програмного забезпечення важливо підтримувати аналогічну структуру коду, форматування та стиль загалом, саме тому було прийнято рішення про налаштування Rubocop [98] - статичного аналізатора коду, який надає відповідні CLI [99] команди. Однієї з важливих команд може слугувати “bundleexecrubocop .”, яка повертає рекомендації щодо покращення загального форматування та стилю програмного коду.

```
82 files inspected, 214 offenses detected, 196 offenses auto-correctable
```

Слід, для прикладу, розглянути кілька зауважень, щоб краще розуміти їх суть.



Після виправлення зауважень пов’язаних з форматуванням та стилістикою коду, виконано команду “bundleexecrubocop .” знову і на даному етапі отримано інформацію про відсутність явних зауважень. Приклад позитивного результату можна бачити на наступному зображенні:



3.3. Розгортання автоматизованої системи інформаційної технології

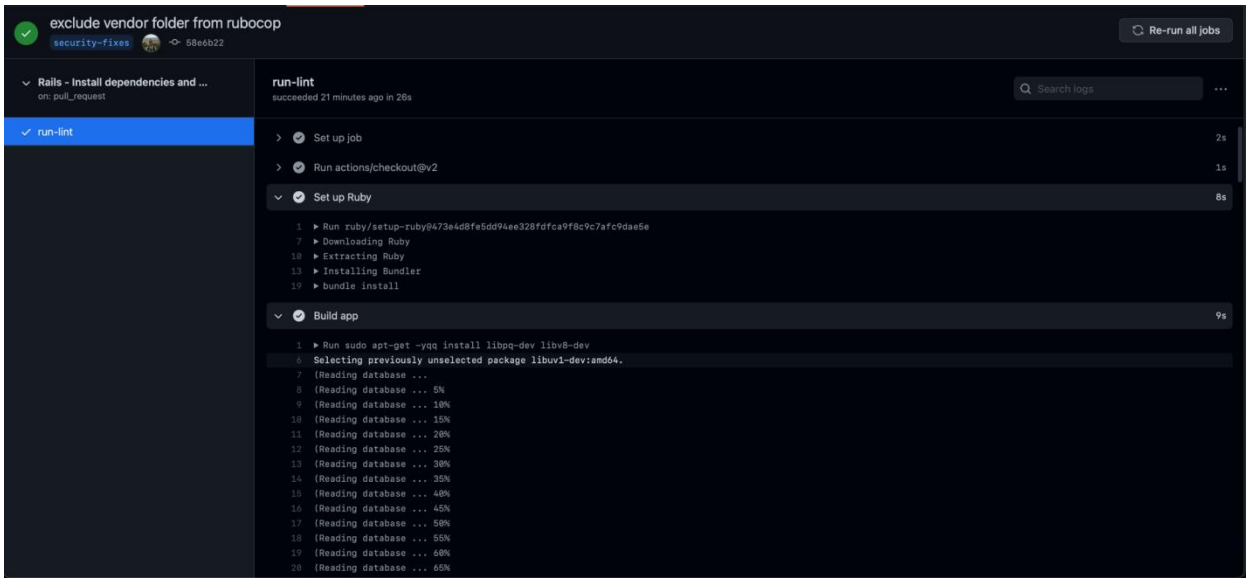
Для того, щоб веб-сервіс був доступний для реальних користувачів, необхідно провести процес розгортання. В даному випадку пропонується розглянути приклад успішного процесу розгортання засобами CI/CD [106], GithubWorkflow [110] та Heroku [106].

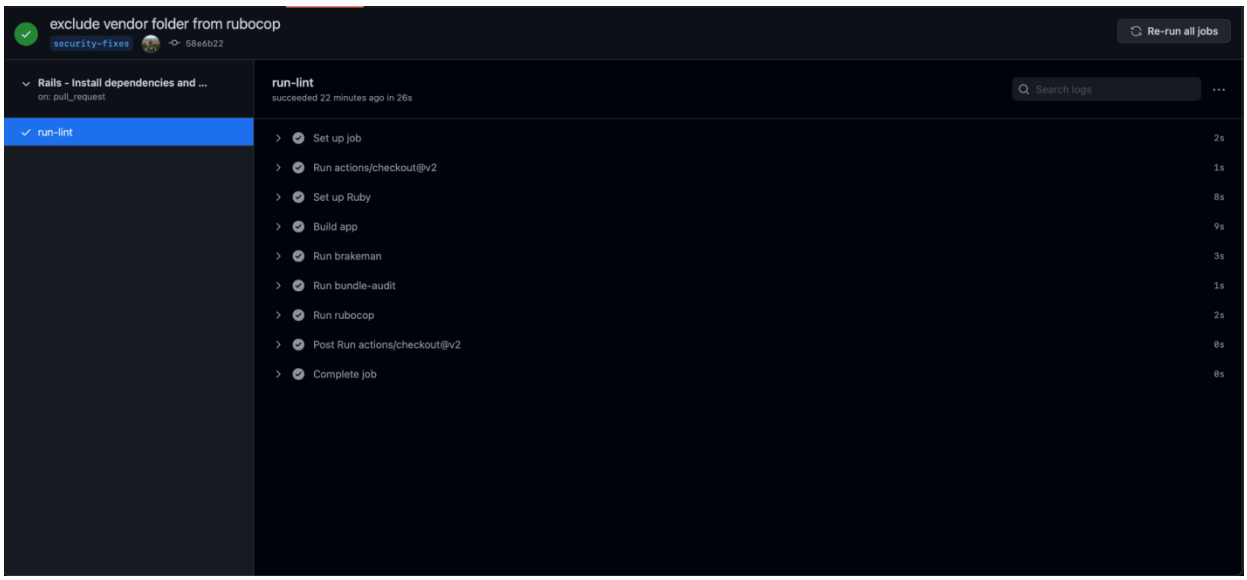
3.3.1. Налаштування CI/CD

Важливо зазначити, що перед розгортанням системи було б зручно налаштувати CI/CD [108], яке містило б кроки пов’язані із статичними аналізаторами коду, засобами перевірки на загальний стан безпеки веб-сервісу, а саме: Rubocop [98], Brakeman [106], bundler-audit [104]. На даний момент ті засоби можуть бути використані лише в межах CLI [100], тому їхнє автоматичне використання істотно покращить якість та швидкість розробки програмного забезпечення. Саме тому було розглянуто GithubWorkflow [110], що ставить собі за мету виконувати вищевказані перевірки після кожної зміни(коміту) веб-сервісу на github.com [109], де вихідний код веб-сервісу безпосередньо зберігається. Ознайомившись із документацією GithubWorkflow [110], для ініціалізації створено директорію “.workflows/” в корені репозиторію і додано файл з конфігураціями всередину. Прикладом конфігураційного файлу GithubWorkflow [110] може слугувати наступний рисунок.

```yaml
name: Install dependencies, run linters and security checks

on:
  push:
    branches: [master]
  pull_request:
    branches: [master]
jobs:
  run-lint:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
      - name: Set up Ruby
        uses: ruby/setup-ruby@v1
        with:
          ruby-version: 2.6.8
          bundler-cache: true

      - name: Build app
        run: |
          sudo apt-get -yqq install libpq-dev libv8-dev
          gem install bundler:1.17.3
          bundle install --jobs 4 --retry 3

      - name: Run brakeman
        run: |
          bundle exec brakeman -q -w2

      - name: Run bundle-audit
        run: |
          bundle exec bundle-audit check --update

      - name: Run rubocop
        run: |
          bundle exec rubocop --parallel
```

Ключ “jobs” містить всі команди, які повинні виконуватись для того, щоб розуміти, що код відповідає наперед заданим стандартам якості. Згідно вказаних налаштувань, CI/CD [108] процес буде запускатись на операційній системі ubuntu з налаштованою мовою програмування Ruby [83] відповідної версії. Додатково додано опцію “bundler-cache: true” , яка буде керувати [56] встановлені залежності, що при повторному запуску CI/CD [108] буде мати позитивний вплив на швидкодію. Приклад виконання Brakeman [106], bundler-audit [104] та Rubocop [98]:

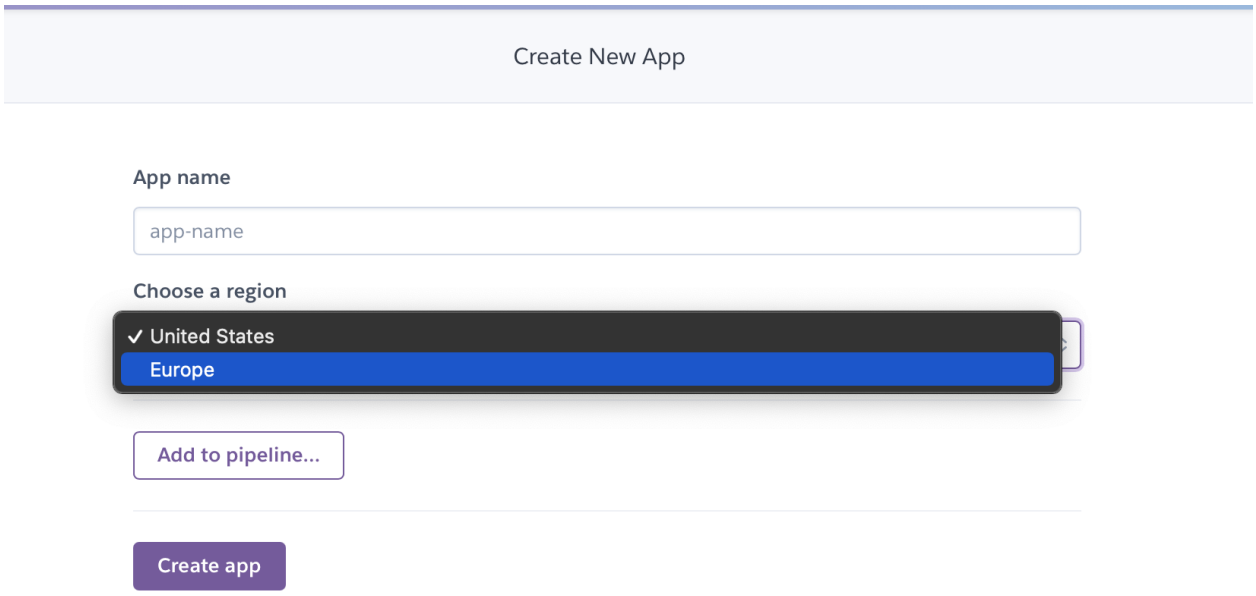




3.3.2. Розгортання засобами Heroku

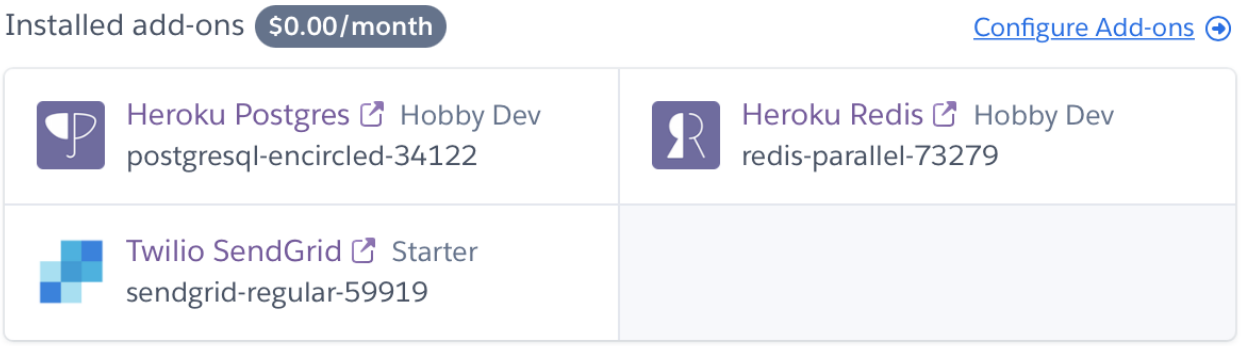
Після налаштування CI/CD [108] варто розглянути можливість розгортання веб-сервісу засобами Heroku [105] - зручною платформою, яка ставить собі за мету робити розгортання надійним, зрозумілим та відносно швидким.

Далі ініційовано окремий сервіс на Heroku [106] в європейському регіоні, щоб покращити швидкість завантаження, бо сервери будуть знаходитись географічно ближче до України.



Додатково необхідно встановити herokuCLI [112] клієнт, який відповідає за взаємодію із віддаленим сервером за допомогою локального терміналу користувацького комп'ютера.

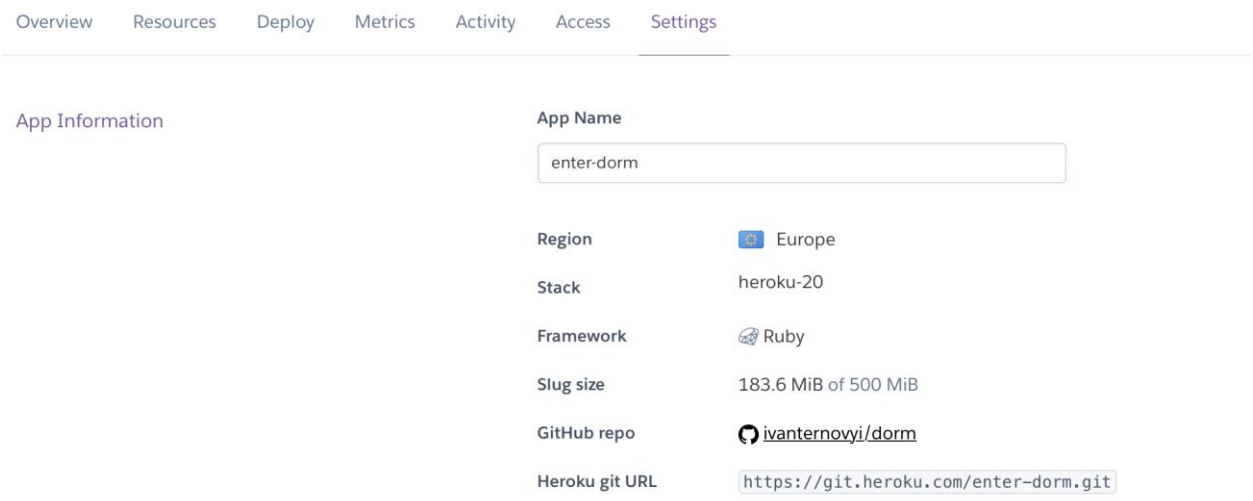
Необхідно зазначити, що веб-сервіс потребує базу даних PostgreSQL [93], сховище для фонових задач Redis [35] і засіб для розсилки листів Sendgrid [113], тому ці засоби були налаштовані за допомогою Heroku [98].



Для коректної роботи бібліотеки Sidekiq [87], яка базується на фонових задачах, необхідно налаштувати Sidekiq Worker, який варто створити в спеціальному файлі – Procfile [114], окремо додано команду запуску: "worker: bundle exec sidekiq -c 2".

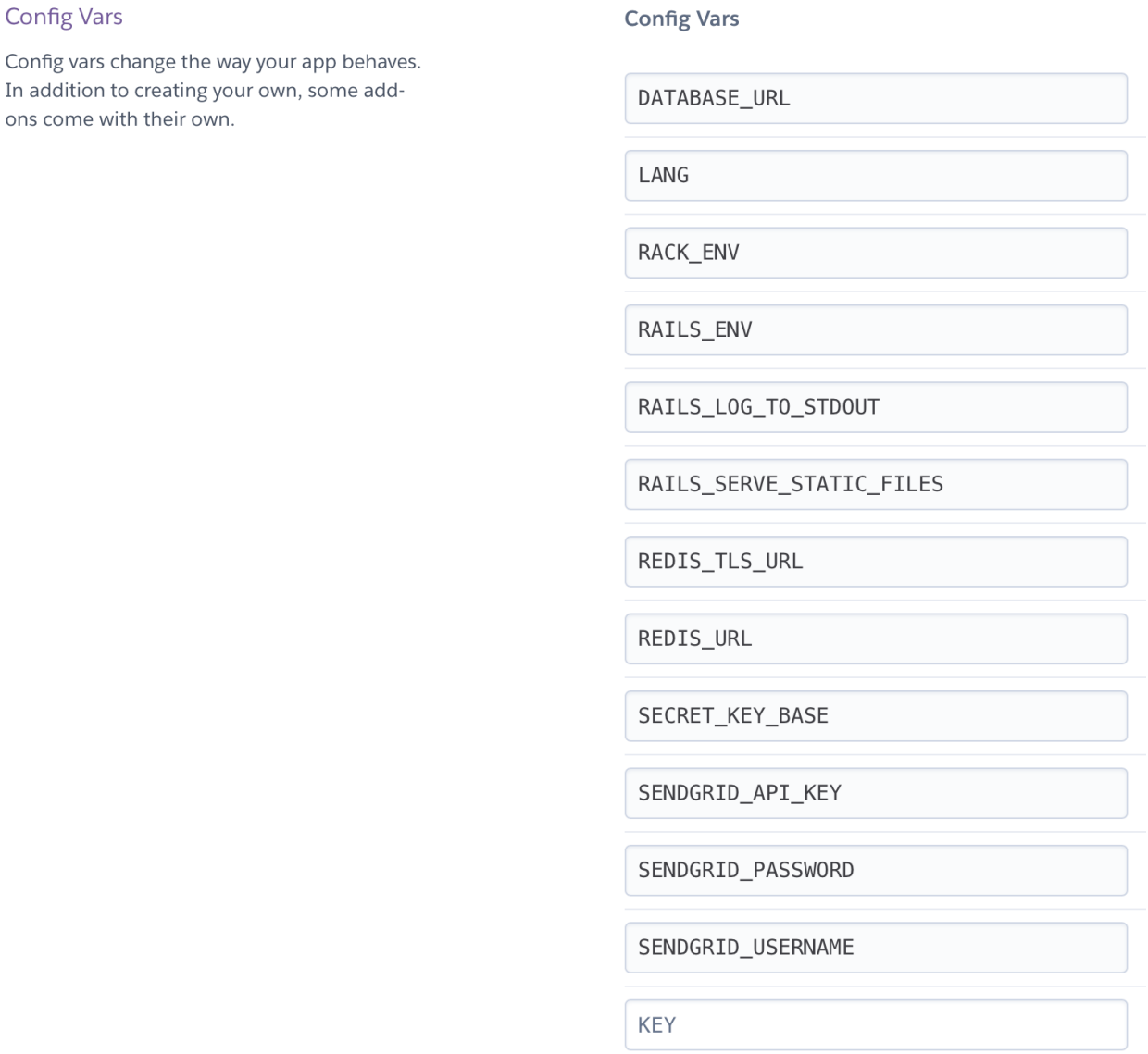
Після вищевказаних дій, можемо бачити список запущених процесів, які містять процес веб-сервісу і процес пов’язаний із виконанням фонових задач.

Також Heroku надає можливість перегляду різного роду налаштувань та конфігурацій, які були використані під час запуску веб-сервісу.

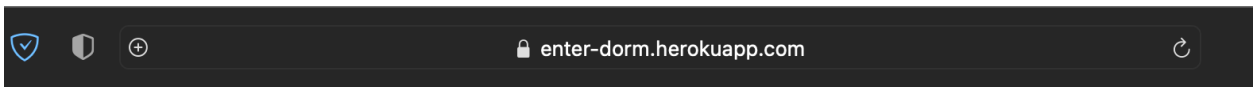


Для успішної роботи веб-сервісу необхідною умовою є використання змінних, які передаються із графічного інтерфейсу Heroku [107]. Прикладом таких змінних може бути інформація про базу даних [73], ключі доступу та інші приватні дані. За замовчування такого роду дані є прихованими на графічному представленні, щоб лише при явній потребі мати доступ до них з міркувань безпеки.

Функціонал “Reveal Config Vars” надає можливість перегляду конфіденційних змінних, а також допускає їх зміну при потребі.

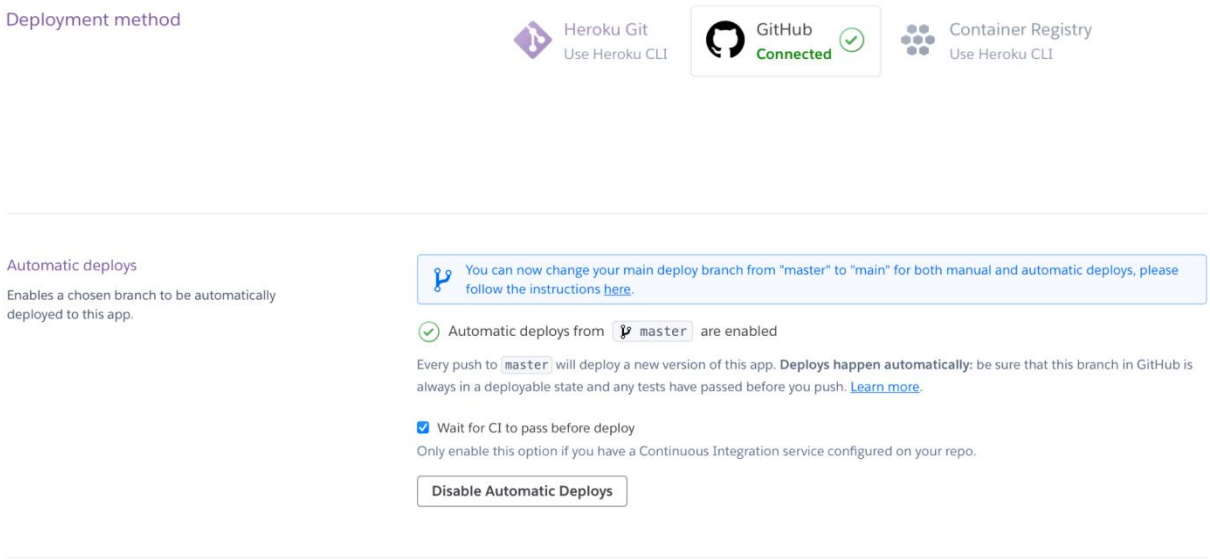


Після виконання вищевказаних кроків, прикладом доступності веб-сервісу може слугувати наступне зображення:

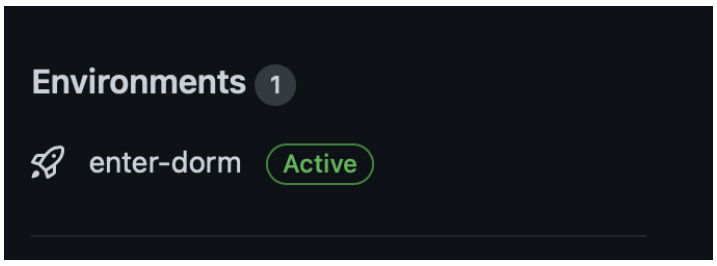


3.3.3. Налаштування автоматичного розгортання

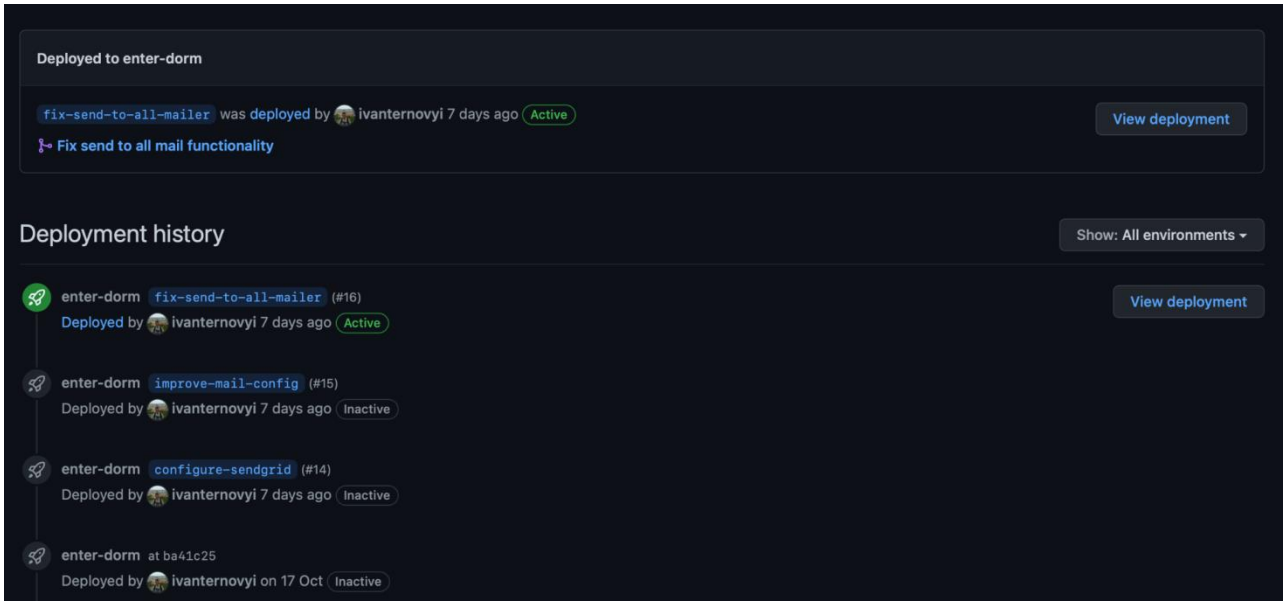
Логічним варіантом оптимізації процесу подальшої розробки веб-сервісу є налаштування автоматичного розгортання, яке варто ініціювати при кожній успішній зміні доданій до гілки master. Якщо репозиторій веб-сервісу знаходиться на Github [108], то Heroku [106] надає можливість відповідної безпосередньої інтеграції.



Варто зазначити, що після вищевказаних маніпуляцій, появилось нове середовище(Environment) у репозиторії на Github [109]. Це дає змогу бачити посилання за яким веб-сервіс є розгорнутим, а також сам процес розгортання в загальному.

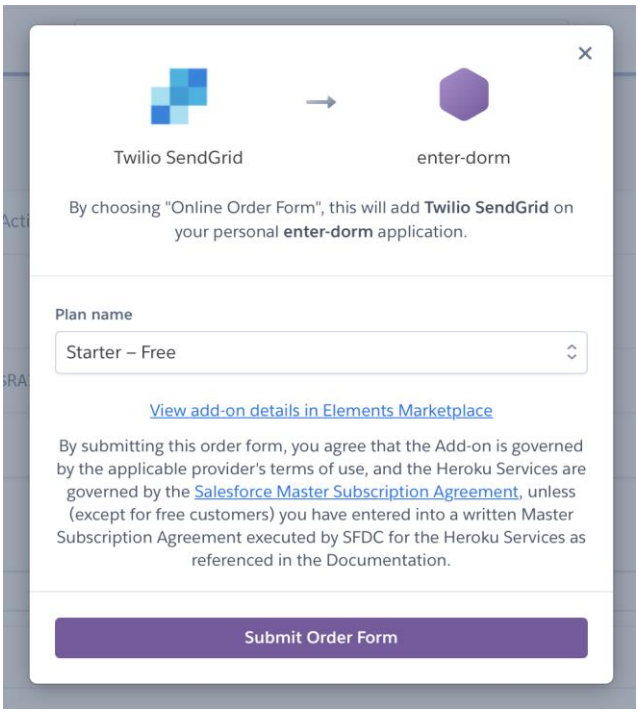


Додатково існує можливість перегляду історії змін, пов’язаних із розгортанням, а за допомогою посилання “View deployment” можна перейти на актуальну версію розгорнутого веб-сервісу.

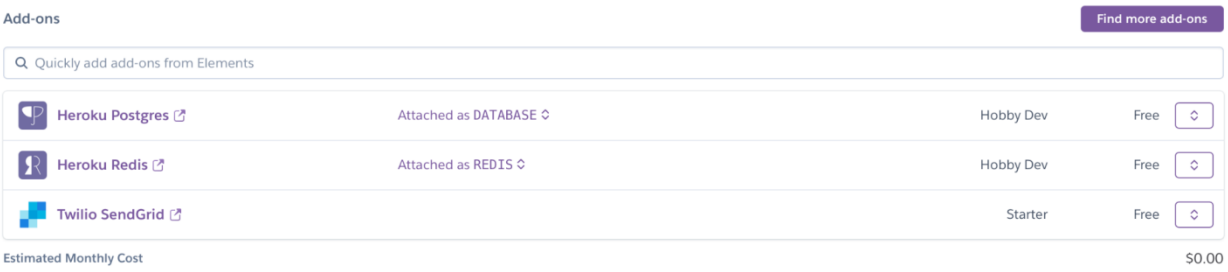


3.3.4. Sendgrid

Варто згадати, що веб-сервіс містить функціонал пов’язаний із надсиланням листів, тому потрібно коректно налаштувати SMTP сервер [115], щоб веб-сервіс міг опрацьовувати такого роду запити. Слід зауважити, що у випадку розгортання веб-сервісу на платформі Heroku [106], потрібно обрати сервіс для зручного керування листами серед доступних додатків на Heroku і таким сервісом може слугувати Sendgrid [113]. Для початку роботи достатньо вибрати безкоштовний план використання Sendgrid на Heroku, який дозволяє надсилати до сто тисяч листів в місяць.



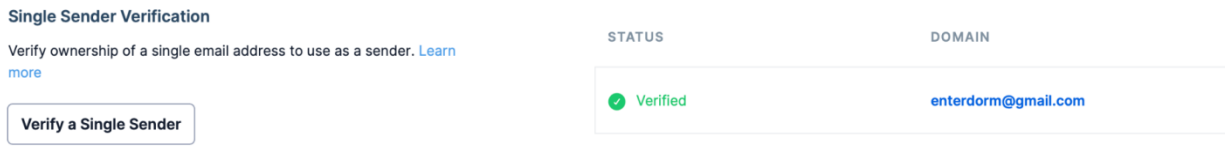
Після налаштування Sendgrid [113], можна бачити його серед існуючих додатків, які є у використанні для коректного функціонування веб-сервісу:



Після додавання Sendgrid на Heroku, необхідно коректно налаштувати ActionMailer [88], щоб фреймворк Ruby on Rails [83] використовував Sendgrid [113] як основний SMTP сервер [115] для надсилання електронних листів. Прикладом для SMTP конфігурацій в ActionMailer може слугувати наступне:

```ruby
# frozen_string_literal: true

ActionMailer::Base.smtp_settings = {
  user_name: 'apikey',
  password: ENV['SENDGRID_API_KEY'],
  domain: 'enter-dorm.herokuapp.com',
  address: 'smtp.sendgrid.net',
  port: 587,
  authentication: :plain,
  enable_starttls_auto: true
}

ActionMailer::Base.default from: 'EnterDorm <enterdorm@gmail.com>'
ActionMailer::Base.perform_deliveries = true
ActionMailer::Base.raise_delivery_errors = true

ActionMailer::Base.delivery_method = if Rails.env.test?
                                       :test
                                     elsif Rails.env.development?
                                       :letter_opener
                                     else
                                       :smtp
                                     end

ActionMailer::Base.default_url_options = if Rails.env.test? || Rails.env.development?
                                           { host: 'localhost', port: 3000 }
                                         elsif Rails.env.production?
                                           { host: 'enter-dorm.herokuapp.com' }
                                         end

Rails.application.routes.default_url_options = ActionMailer::Base.default_url_options
```

Крім того, Sendgrid вимагає налаштування відправника, який має існуючу пошту та представляє певну компанію чи організацію. Для прикладу було ініційовано електронну скриньку “enterdorm@gmail.com” і додано її для використання в Sendgrid [113].



Додатково Sendgrid надає зручне графічне представлення для перегляду інформації про статус надісланих листів (рис. 3.23). Крім того, є можливість аналізу різного роду причин пов’язаних із помилками відправлень листів, якщо такі існують.

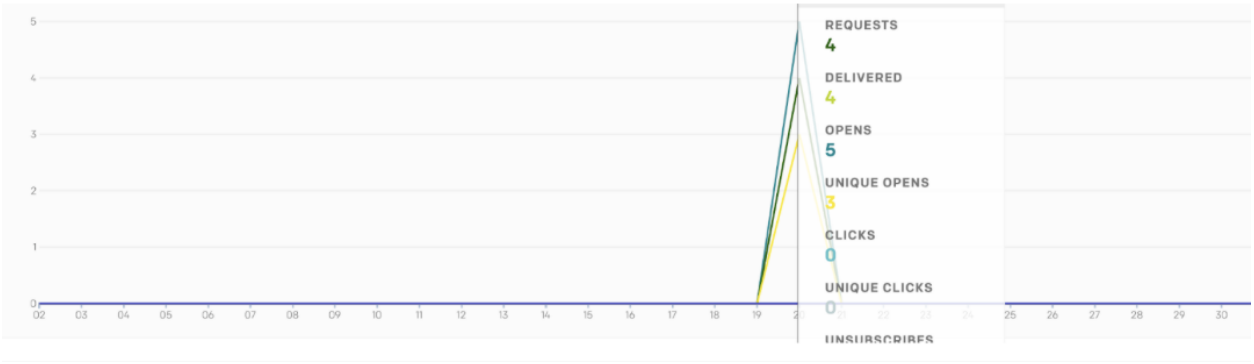
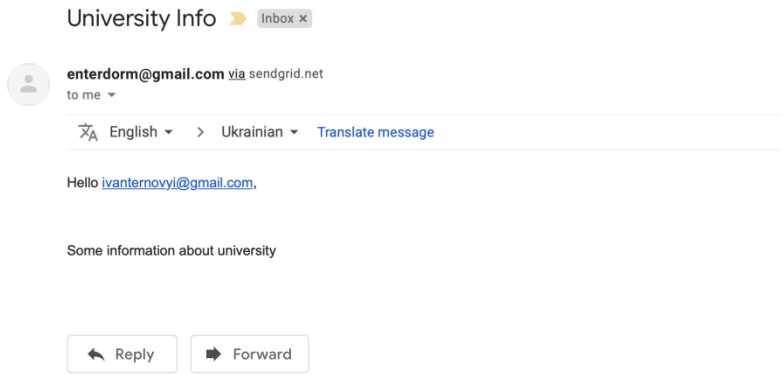


Рис. 3.23. Графічне представлення інформації про статус надісланих листів

Окрім того, щоб впевнитись в коректності розсилки листів, було протестовано даний функціонал за допомогою персоналізованого листа на власну скриньку і як видно з наступного рисунку, все пройшло успішно:



3.4. Висновок до розділу 3

У цьому розділі було детально розглянуто процес проектування за допомогою діаграм прецедентівта діаграм класів, поетапної імплементації та розгортання системи на платформі Heroku. Було описано архітектуру підсистем, їхню взаємодію, а також реалізацію ключових функціональних компонентів, що забезпечують роботу користувачів різних ролей, новин, листування, бронювання кімнат. Особливу увагу приділено інтеграції фонових задач, локалізації та фільтрації контенту, що підвищує ефективність та загальну ергономіку системи.

Варто зауважити, що імплементація кожної підсистеми виконана з урахуванням сучасних підходів до розробки, що забезпечує гнучкість, масштабованість та безпеку веб-сервісу. Використання статичного аналізатора коду сприяє підтримці хорошої якості коду та зниженню ризиків появи різного роду вразливостей. Зокрема, налаштування статичного аналізатора дозволяє виявляти потенційні проблеми на ранніх етапах розробки, що значно підвищує стабільність та безпеку системи загалом.

Приклад розгортання системи було реалізовано через автоматизовані процеси CI/CD на базі GithubWorkflow, що дозволяє забезпечити стабільність оновлень та ефективне розгортання на платформі Heroku. Завдяки налаштуванню автоматичного розгортання процес оновлення системи відбувається безперервно, що в свою чергу знижує ризик виникнення помилок. Додатково було інтегровано SendGrid, який ставить собі за мету опрацьовувати та відправляти електронні листи, для забезпечення коректної роботи підсистеми листування.

Важливим аспектом імплементації є забезпечення зручності використання веб-сервісу для користувачів різних ролей. Реалізовані підсистеми дають змогу студентам бути в курсі останніх новин, комунікувати між собою за допомогою онлайн чату реалізованого за допомогою WebSocket протоколу, здійснювати бронювання вільних кімнат при потребі, а також загалом використовувати веб-сервіс із зручністю. Варто згадати, що впровадження сучасних технологій дозволило досягти високої продуктивності та ефективності роботи системи. Таким чином, виконані кроки дозволяють системі відповідати сучасним вимогам щодо продуктивності, безпеки та зручності використання. Подальший розвиток системи може включати розширення її функціональності, оптимізацію роботи підсистем, а також удосконалення заходів безпеки для забезпечення ще більшої стійкості до можливих загроз.

РОЗДІЛ 4
ФОРМУВАННЯ БЕЗПЕКИ АВТОМАТИЗОВАНОЇ СИСТЕМИ ПІДТРИМКИ ФУНКЦІОНУВАННЯ СТУДЕНТСЬКОГО МІСТЕЧКА: ТИПИ ЗАГРОЗ ТА МЕТОДИ ЇХ НЕЙТРАЛІЗАЦІЇ

У сьогоденні цифровий світ розгортається стрімко, але безпека залишається наріжним каменем будь-якого онлайн-ресурсу. Важливо виділяти час для гарантування безпеки персональної інформації користувачів та забезпечення її надійного захисту. Спробуємо розглянути кілька потенційних кіберзагроз для конкретного веб-сервісу, а також можливі стратегії протидії.

4.1. SQL ін’єкції

SQL-ін’єкція [176] – це загроза, котра дозволяє зловмисникові дістатися до відомостей з бази даних, змінювати їх або псувати. На цю загрозу слід звернути пильну увагу, адже цей веб-сервіс має власну базу даних, де, можливо, зберігаються персональні дані користувачів. Варто розглянути кілька різновидів SQL-ін’єкцій як приклад.

Популярним є такий тип SQL ін’єкцій, за допомогою якого до запиту до бази даних додається твердження “OR 1=1”, а саме: “SELECTId, Name, PasswordFROMUsersWHEREId = 5 or 1=1;”. За рахунок того, що твердження “OR 1=1” завжди повертає правдиве значення, то даний запит буде успішним і буде повернуто “Id”, “Name” та “Password” усіх записів із таблиці “Users”.

Також необхідно розглянути SQL-ін'єкцію, що здатна спричинити втрату даних. Цей прийом потребує підтримки з боку бази даних групування запитів, інакше кажучи, кількох запитів, що відокремлені крапкою з комою. Розглянемо приклад такої групи запитів:

“SELECT * FROM Users; DELETE FROM Users WHERE Id = 5;”.

У разі, якщо база даних має підтримку подібних позначень, зловмисники здатні використовувати це у власних цілях. Припустімо, на графічному

інтерфейсі передбачено введення певного значення, котре буде залучене у фільтрації користувачів, а сам запит матиме такий вигляд:

userName = getInputString();

query = “SELECT * FROM Users WHERE Name =” + userName

У такому разі, замість імені, зловмисник може вписати додаткові відомості, що здатні скомпрометувати запит до бази даних. Цього можна досягти, ввівши подібний рядок:

“‘name’; DROPTABLEUsers”. Тим самим це значення буде взяте до уваги і буде передане як частина запиту до бази даних, а цілий запит буде мати наступний вигляд:

“SELECT * FROMUsersWHERName =’ name’; DROPTABLEUsers”.

У такому разі спершу ми витягнемо необхідний запис з таблиці, після чого видалимо всі записи з цієї таблиці. Як можна переконатися, такий вид SQL ін’єкцій є вкрай ризикованим, адже існує загроза втрати інформації.

Щоб уникнути SQL ін’єкцій [23], варто застосовувати SQL параметри. Завдяки такому функціоналу SQL буде здійснювати перевірку й контроль коректності значень, які передаються безпосередньо до вихідного запиту. Враховуючи той факт, що цей веб-сервіс використовує PostgreSQL як СУБД та ActiveRecord [45] як ORM, вищезгадані операції слід реалізовувати на боці ActiveRecord. Для початку варто переглянути всі запити до бази даних та трансформувати їх таким чином, щоб вони використовували SQL параметри наступним чином:

```
scope :by_first_name, ->(first_name) { where('first_name LIKE ?', "%#{first_name}%") }
scope :by_last_name, ->(last_name) { where('last_name LIKE ?', "%#{last_name}%") }
scope :by_email, ->(email) { where('email LIKE ?', "%#{email}%") }
```

4.2. Cross Site Request Forgery

CSRF [177] – це різновид загрози, яка змушує кінцевого користувача виконувати небажані операції в межах веб-сервісу, де користувач уже пройшов аутентифікацію. Застосовуючи розсилку посилань електронною поштою або

через чати, зловмисник може підмінити наміри користувача на власні, шахрайські. Вдала CSRF [177] атака здатна призвести до втрати коштів користувачем, зміни адреси електронної пошти тощо. Більше того, якщо жертва – адміністратор системи, під загрозою опиняється весь веб-сервіс. Спробуємо розібратися в принципах роботи цієї техніки, щоб мати чітке уявлення.

Існує безліч способів, які використовують зловмисники, аби спонукати користувача до небажаних дій. Уявімо Алісу, котра планує перевести гроші Бобу через веб-додаток bank.com, вразливий до CSRF [177]. Також є зловмисниця – Марія, що прагне ошукати Алісу і таким чином заволодіти грошима. Щоб це стало можливим, Марії потрібно створити шахрайське посилання та переконати Алісу його натиснути.

Розглянемо випадок, коли відбувається GET запит. Можна спрогнозувати, що посилання буде мати наступний вигляд: `GET http://bank.com/transfer.do?acct=BOB&amount=100 HTTP/1.1`

В такому випадку Марія може змінити суму переказу на 100 000 і змінити отримувача на свої дані, тобто генерує посилання схоже на наступне: `http://bank.com/transfer.do?acct=MARIA&amount=100000`

Тепер, наступним викликом для Марії буде заохотити Алісу натиснути на надане вище посилання. Цього можна домогтися, застосовуючи його, скажімо, для перегляду світлин, наприклад:

```
<a href="http://bank.com/transfer.do?acct=MARIA&amount=100000">View my Pictures!
</a>
```

ну або використати несправжнє зображення розміром 0x0 пікселів:

```
<img src="http://bank.com/transfer.do?
acct=MARIA&amount=100000" width="0" height="0" border="0">
```

Якщо це зображення потрапить до електронного листа, Аліса не матиме можливості нічого вгадати, але браузер виконає запит, що спричинить списання коштів.

Розгляньмо також сценарій з POST запитом. Єдина відмінність в атаці між GET та POST запитами полягає у способах здійснення атаки з боку жертви. Припустімо, запит має таку структуру:

```
POST http://bank.com/transfer.do HTTP/1.1

acct=BOB&amount=100
```

Даний запит не може бути виконаним за допомогою використання зображення, як у минулому варіанті, але тут можна використати FORM тег:

```
<form action="http://bank.com/transfer.do" method="POST">

<input type="hidden" name="acct" value="MARIA"/>
<input type="hidden" name="amount" value="100000"/>
<input type="submit" value="View my pictures"/>

</form>
```

Сама форма вимагає, щоб користувач надіслав її, натиснувши клавішу, але цей крок можна обійти за допомогою JavaScript [15]:

```
<body onload="document.forms[0].submit()">

<form...
```

Аби убезпечитися від ін'єкції цього типу, розробники фреймворку Ruby on Rails радять застосовувати вбудований механізм захисту від CSRF [177], названий “protect_from_forgery” [168]. Цей метод щоразу при новому запиті створює унікальний маркер, відомий тільки нашому веб-сервісу, і при кожному зверненні до сервера Ruby on Rails [156] здійснює перевірку присутності та відповідності цього маркера. Якщо маркер коректний, тобто відповідає дійсності, все гаразд, і такому запиту можна довіряти; у випадку відсутності маркера або його невірності, маємо підозрілу активність, яка може свідчити про наявність CSRF [176] ін'єкції. У такому випадку, коли маркер невірний, сервер реагуватиме поверненням помилки.

Сам токен може міститись в хедері з ключем “X-CSRF-Token”.

Розглянемо як це реалізовано у даному веб-сервісі:

```
class ApplicationController < ActionController::Base
  include Errorable

  protect_from_forgery with: :exception

  before_action :set_locale
```

Розглянемо як саме CSRF [24] токен використовується при наявності FORM тегу (рис. 4.1):

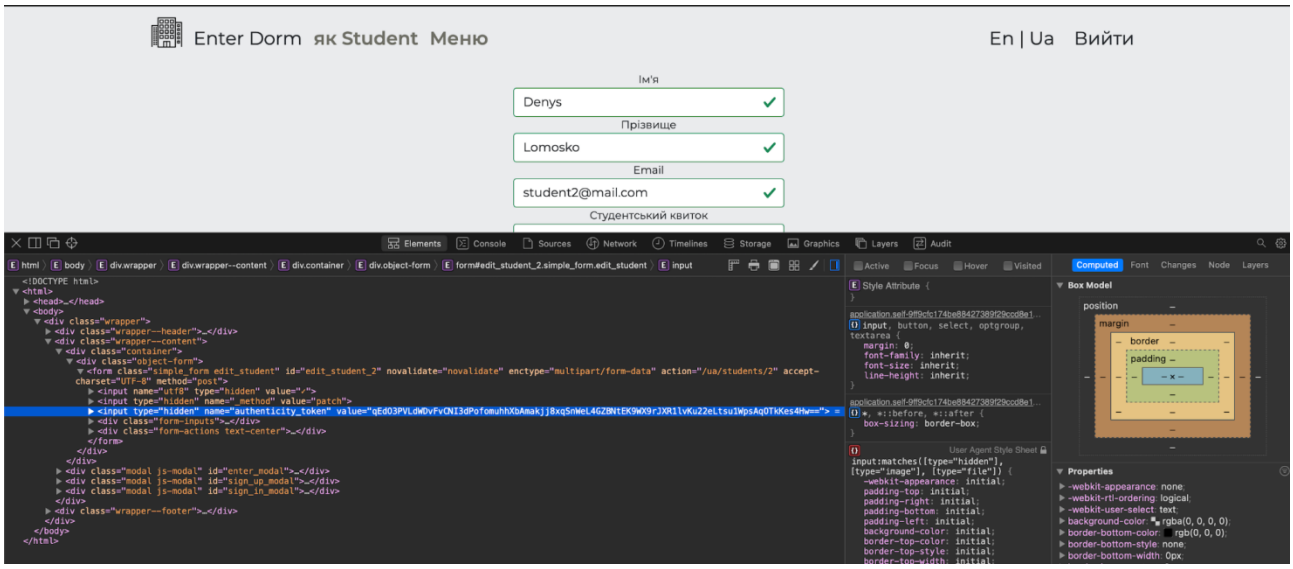


Рис. 4.1. Використання CSRF токену при наявності FORM тегу

4.3. Методи сканування безпеки системи на вразливості

При розробці веб-сервісу було використано чимало сторонніх бібліотек, що значно полегшують та прискорюють загальну імплементацію. Логічно, що ці бібліотеки, як і будь-який код, піддаються певним загрозам, таким як: SQL-ін'єкції [176], CSRF [177], XSS [178] та інші види ін'єкцій. Аби переконатися у безпечності використання сторонніх бібліотек, можна скористатися відомою бібліотекою, яка сканує наявні залежності на вразливості. Ця бібліотека має доступ до бази даних вразливостей, виявлених іншими розробниками в певних сторонніх бібліотеках. Дана бібліотека називається “bundler-audit” [180], а її запуск здійснюється за допомогою CLI [175] наступною командою: “bundle exec bundle-audit check --update”. Після виконання цієї команди було отримано

досить великий перелік вразливих бібліотек та запропоновано варіанти відповідних контрзаходів:

```
Name: carrierwave
Version: 1.3.1
Advisory: CVE-2021-21305
Criticality: High
URL: https://github.com/carrierwaveuploader/carrierwave/security/advisories/GHSA-cf3w-g86h-35x4
Title: Code Injection vulnerability in CarrierWave::RMagick
Solution: upgrade to ~> 1.3.2, >= 2.1.1

Name: jquery-rails
Version: 4.3.5
Advisory: CVE-2020-11023
Criticality: Medium
URL: https://blog.jquery.com/2020/04/10/jquery-3-5-0-released
Title: Potential XSS vulnerability in jQuery
Solution: upgrade to >= 4.4.0

Name: puma
Version: 3.12.2
Advisory: CVE-2021-29509
Criticality: High
URL: https://github.com/puma/puma/security/advisories/GHSA-q28m-8xjw-8vr5
Title: Keepalive Connections Causing Denial Of Service in puma
Solution: upgrade to ~> 4.3.8, >= 5.3.1

Name: puma
Version: 3.12.2
Advisory: CVE-2020-5249
Criticality: Medium
URL: https://github.com/puma/puma/security/advisories/GHSA-33vf-4xgg-9r58
Title: HTTP Response Splitting (Early Hints) in Puma
Solution: upgrade to ~> 3.12.4, >= 4.3.3
```

```
Name: puma
Version: 3.12.2
Advisory: CVE-2020-11077
Criticality: Medium
URL: https://github.com/puma/puma/security/advisories/GHSA-w64w-qqph-5gxm
Title: HTTP Smuggling via Transfer-Encoding Header in Puma
Solution: upgrade to ~> 3.12.6, >= 4.3.5

Name: puma
Version: 3.12.2
Advisory: CVE-2020-5247
Criticality: Medium
URL: https://github.com/puma/puma/security/advisories/GHSA-84j7-475p-hp8v
Title: HTTP Response Splitting vulnerability in puma
Solution: upgrade to ~> 3.12.4, >= 4.3.3

Name: puma
Version: 3.12.2
Advisory: CVE-2021-41136
Criticality: Low
URL: https://github.com/puma/puma/security/advisories/GHSA-48w2-rm65-62xx
Title: Inconsistent Interpretation of HTTP Requests ('HTTP Request Smuggling') in puma
Solution: upgrade to ~> 4.3.9, >= 5.5.1

Name: sidekiq
Version: 6.0.6
Advisory: CVE-2021-30151
Criticality: Medium
URL: https://github.com/advisories/GHSA-grh7-935j-hg6w
Title: Cross-site Scripting in Sidekiq
Solution: upgrade to ~> 5.2.0, >= 6.2.1
```

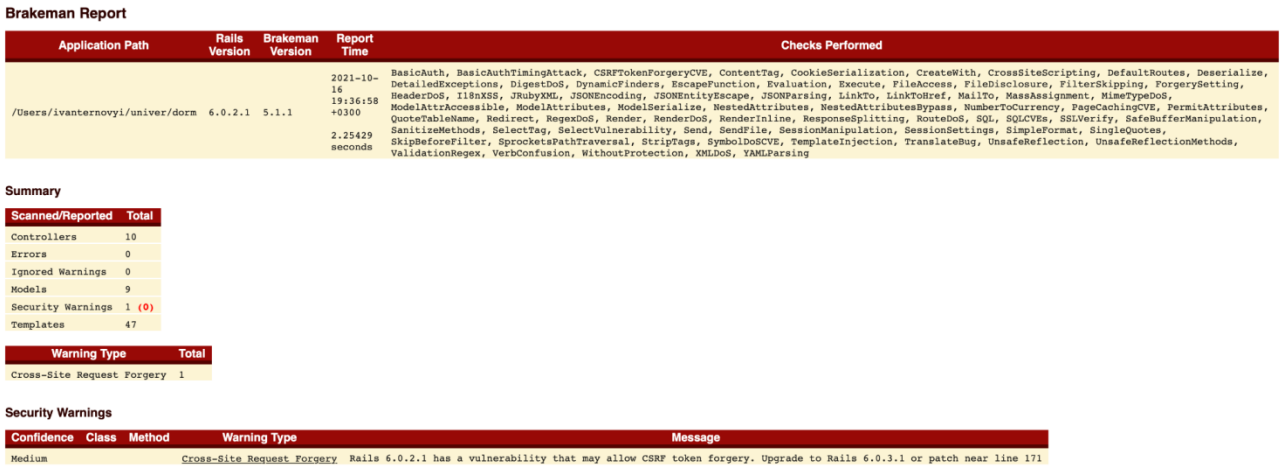
Далі було розпочато процес оновлення сторонніх залежностей, щоб зробити веб-сервіс безпечнішим. Для прикладу було оновлено “puma” [181]:

```
Using pry 0.12.2
Fetching puma 5.5.1 (was 3.12.2)
Installing puma 5.5.1 (was 3.12.2) with native extensions
Using rack-protection 2.0.8.1
```

Після тривалого оновлення та підбору версій сторонніх бібліотек, можна бачити, що веб-сервіс не має залежностей на вразливі сторонні бібліотеки:

```
ivanternovyi@Ivans-MacBook-Pro dorm % bundle exec bundle-audit check --update
Updating ruby-advisory-db ...
From https://github.com/rubysec/ruby-advisory-db
 * branch            master     -> FETCH_HEAD
Already up to date.
Updated ruby-advisory-db
ruby-advisory-db: 522 advisories
No vulnerabilities found
```

Також додатково було використано бібліотеку “brakeman” [182], яка також перевіряє систему на вразливості. Після запуску команди “bundleexecbrakeman -q -w2” генерується HTML [166] сторінка зі звітом по безпеці (рис. 4.2):



Brakeman Report

Application Path	Rails Version	Brakeman Version	Report Time	Checks Performed
/Users/ivanternovyi/univer/dorm	6.0.2.1	5.1.1	2021-10-16 19:36:58 +0300 2.25429 seconds	BasicAuth, BasicAuthTimingAttack, CSRFTokenForgeryCVE, ContentTag, CookieSerialization, CreateWith, CrossSiteScripting, DefaultRoutes, Deserialize, DetailedExceptions, DigestDoS, DynamicFinders, EscapeFunction, Evaluation, Execute, FileAccess, FileDisclosure, FilterSkipping, ForgerySetting, HeaderDoS, I18nXSS, JRubyXML, JSONEncoding, JSONEntityEscape, JSONParsing, LinkTo, LinkToHref, MailTo, MassAssignment, MimeTypeDoS, ModelAttrAccessible, ModelAttributes, ModelSerialize, NestedAttributes, NestedAttributesBypass, NumberToCurrency, PageCachingCVE, PermitAttributes, QuoteTableName, Redirect, RegexDoS, Render, RenderDoS, RenderInline, ResponseSplitting, RouteDoS, SQL, SQLCVEs, SSLVerify, SafeBufferManipulation, SanitizeMethods, SelectTag, SelectVulnerability, Send, SendFile, SessionManipulation, SessionSettings, SimpleFormat, SingleQuotes, SkipBeforeFilter, SprocketsPathTraversal, StripTags, SymbolDoSCVE, TemplateInjection, TranslateBug, UnsafeReflection, UnsafeReflectionMethods, ValidationRegex, VerbConfusion, WithoutProtection, XMLDoS, YAMLParsing

Summary

Scanned/Reported	Total
Controllers	10
Errors	0
Ignored Warnings	0
Models	9
Security Warnings	1 (0)
Templates	47

Warning Type	Total
Cross-Site Request Forgery	1

Security Warnings

Confidence	Class	Method	Warning Type	Message
Medium			Cross-Site Request Forgery	Rails 6.0.2.1 has a vulnerability that may allow CSRF token forgery. Upgrade to Rails 6.0.3.1 or patch near line 171

Рис. 4.2. Генерування HTML сторінки зі звітом по безпеці після запуску команди “bundle exec brakeman -q -w2”

4.4. Захист від XSS

Cross-Site Scripting (XSS) [162] — це різновид загрози, яка дозволяє зловмисникам впроваджувати згубний код (зазвичай JavaScript [168]) у веб-застосунок, що виконується напряму у браузері. Кібератака подібного типу може спричинити викрадення інформації сесій, облікових записів юзерів, здійснення небажаних операцій від імені користувачів чи навіть повне руйнування безпеки веб-сервісу. Розглянемо деякі вмонтовані способи оборони фреймворку Ruby on Rails [156] від XSS [179], що зменшують вірогідність втілення таких атак.

В Ruby on Rails [156], для шаблонів, створених через ERB [157], екранування даних функціонує за замовчуванням. Тобто, кожен текст, вставлений у шаблон, автоматично модифікується таким чином, щоб спеціальні символи (<, >, &) не інтерпретувались як HTML [164] або JavaScript [168] код.

```
%tr
  %td{ scope: 'row' }= post.id
  %td{ scope: 'row' }= post.title
  %td{ scope: 'row' }= post.body
  %td{ scope: 'row' }= post.important?
  %td{ scope: 'row' }= post.created_at
```

Навіть якщо зміст post, взятого з прикладу, матиме шкідливі дані, скажімо: <script>alert('XSS')</script>, що мало би призвести до появи модального вікна на сторінці, то фреймворк гарантовано обробить небезпечний вміст, і код не буде активований, а буде просто відображено <script>alert('XSS')</script>.

Не слід забувати про захист від потенційно небезпечних посилань, бо такі лінки нерідко є першоджерелом компрометації даних користувачів на веб-платформах. Під час створення посилань вкрай важливо уникати введення зловмисних URL-адрес [148], зокрема тих, що містять Javascript [168] як префікс, адже існує загроза невиправданого запуску певного коду.

```
.objects-list
  - @claim_rooms.each do |claim_room|
    .objects-list--item.-wide
      .-status-text= claim_room.status
      - student = claim_room.claim.student
      = link_to student.email, student_path(student)
      - room = claim_room.room
      = link_to room.name, room_path(room)
      = link_to t('claims.index.mark_in_progress'),
        mark_in_progress_room_claim_path(claim_room.room, claim_room.claim),
        method: :post,
        class: 'btn btn-primary'
      = link_to t('claims.index.approve'),
        approve_room_claim_path(claim_room.room, claim_room.claim),
        method: :post,
        class: 'btn btn-success'
      = link_to t('claims.index.reject'),
        reject_room_claim_path(claim_room.room, claim_room.claim),
        method: :post,
        class: 'btn btn-danger'
```

У цьому прикладі наочно демонструється пряме застосування методу link_to [202], що є вбудованою функцією фреймворку та відповідає за захищеність посилань. Спроба вставити javascript:alert(‘XSS’) як посилання, застосовуючи метод link_to, призведе до блокування цього коду з боку Ruby on Rails [170], що вказує на загальну безпеку веб-додатку.

4.5. Mass Assignment (Strong Parameters)

Mass Assignment[192] – це тип небезпеки, який визначається як масова зміна інформації, що виникає через надсилання клієнтом неочікуваних даних під час обробки різноманітних запитів, на кшталт: POST, PUT, PATCH та інших.

Фреймворк Ruby on Rails [156] застосовує механізм Strong Parameters [39], який гарантує безпечну фільтрацію вхідних даних, забезпечуючи захист веб-сервісу від потенційних атак, пов'язаних із масовим присвоєнням (Mass Assignment). Цей спосіб передбачає обов'язкове вказування дозволених атрибутів для кожного окремого запиту. За допомогою методу require [203] задається необхідний кореневий ключ, а через функцію permit [203] визначається перелік лише тих параметрів, які є допустимими до використання та не спричинять негативних наслідків для функціонування програми. Застосування Strong Parameters [192] запобігає коригуванню критичних або небажаних полів, підвищуючи безпеку та забезпечуючи чіткий контроль над вхідною інформацією. Варто навести приклад:

```ruby
class RoomsController < ApplicationController
  before_action :find_room, except: %i[index new create]
  before_action :authenticate_manager, except: %i[index show]

  # ...

  def create
    @room = Room.new(room_params)
    if @room.save
      redirect_to room_path(@room)
    else
      render :new
    end
  end

  def update
    @room.assign_attributes(room_params)
    if @room.save
      redirect_to room_path(@room)
    else
      render :edit
    end
  end

  private

  # ...

  def room_params
    params.require(:room).permit(
      :name, :max_tenants, :floor, :beds_count,
      :balcony, :fridge, :microwave, :avatar
    )
  end
end
```

Ми бачимо, що метод `room_params` застосовує згадані вище способи контролю передачі даних, а саме: `require` і `permit`. Для коректного зчитування даних, параметри вхідного запиту повинні відповідати наступній структурі:

```
{
  room: {
    name: "test value",
    max_tenants: "test value",
    floor: "test value",
    beds_count: "test value",
    balcony: "test value",
    fridge: "test value",
    microwave: "test value",
    avatar: "test value"
  }
}
```

У разі виникнення потреби, параметри Strong params дають змогу обмежити перелік полів, що зазнають змін у разі успішного виконання запиту. Наприклад, маємо змогу дозволити зміну лише назви кімнати, додавши permit(:name) [203], після чого, не має значення, які додаткові поля буде передано запитом від клієнта, бо в контролері братиметься до уваги чітко тільки значення поля name.

```
def room_params
  params.require(:room).permit(:name)
end
```

У ймовірному майбутньому розширених моделей Room слід враховувати необхідність коригування методу room_params, аби нововведені дані були коректно збережені в базі даних.

В цілому застосування Strong Parameters [192] в фреймворку Ruby on Rails [156] є обов'язковим, оскільки впроваджує безпеку за замовчуванням: параметри, які явно не авторизовані для змін, ігноруються, тим самим оберігаючи веб-сервіс від неконтрольованого і небажаного внесення змін до бази даних.

4.6. Content Security Policy

Під час користування веб-застосунком, браузер мусить завантажити та відобразити вміст і код веб-сторінки, що може охоплювати значний обсяг інформації. Цей процес є типовим і зазвичай не викликає побоювань, адже більшість сучасних веб-сайтів мають складну структуру, складену з великої кількості коду, написаного мовами HTML [166], CSS [165] та JavaScript [168], а також інших складових, як-от: зображення та файли, що використовуються в коді. Код, який браузер завантажує, може походити як з того самого джерела (домену, з якого надійшов запит), так і з іншого, хоча браузери за замовчуванням цього не розрізняють.

Після завантаження всіх ресурсів для веб-застосунку, браузер повинен обробити й запустити їх, одночасно відфільтровуючи потенційно небезпечний вміст або небажаний доступ до інформації, що не стосується поточного веб-сервісу. Для мінімізації ризиків, сервер веб-застосунку повинен передавати інформацію про Content Security Policy [196] у заголовку відповіді на запит, забезпечуючи коректну обробку браузером тільки перевірених ресурсів. Такий механізм захисту допомагає уникнути атак типу Cross Site Scripting (XSS) [196]. Розглянемо приклад заголовку відповіді на запит з CSP [196]:

```
"Content-Security-Policy: default-src 'self';
                          script-src 'self';
                          style-src 'self';
                          font-src 'self';
                          img-src 'self';
                          frame-src 'self';"
```

Слід підкреслити, що кожна частина цього заголовка відповідає за окремий ресурс. До того ж, директива "self" за умовчанням гарантує використання ресурсів виключно з того ж самого джерела, що й веб-сервіс. У разі потреби є змога визначити домени, з яких будуть завантажені ресурси,

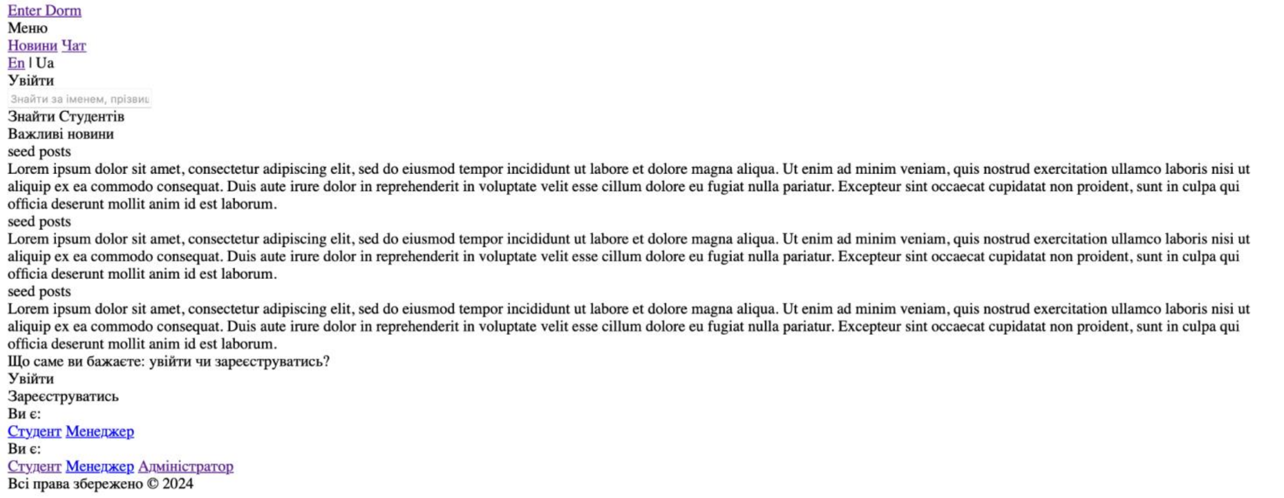
вказавши їхні URL-адреси [47] разом з "self". Існує також можливість дозволити всі домени для отримання ресурсів, шляхом використання “*” замість “self”, але цей підхід рекомендується застосовувати лише в екстремальних ситуаціях.

У фреймворку Ruby on Rails версії 5.2 [156] та новіших версіях підтримка CSP [196] реалізована за замовчуванням. Щоб налаштувати Content Security Policy [197] для вашої веб-аплікації, необхідно створити файл під назвою “config/initializers/content_security_policy.rb”. У цьому файлі буде розміщено список правил безпечного управління джерелами контенту. Наступний приклад може слугувати ілюстрацією:

config > initializers > content_security_policy.rb

```ruby
# frozen_string_literal: true

Rails.application.config.content_security_policy do |policy|
  policy.default_src "example.com", :https
  policy.font_src "example.com", :https, :data
  policy.img_src "example.com", :https, :data
  policy.object_src :none
  policy.script_src "example.com", :https
  policy.style_src "example.com", :https, :unsafe_inline
  policy.report_uri "/your-csp-violation-endpoint"
end

```

У цьому прикладі було вказано "example.com" як джерело всіх ресурсів для веб-застосунку, включаючи програмний код, зображення, шрифти та описи стилів. Слід підкреслити, що це було зроблено навмисно, оскільки ця адреса не є дійсною, і таким чином ми можемо перевірити, що станеться з веб-сторінкою у такому разі.



Можна бачити, що всі стилі, шрифти та інші ресурси не були автоматично завантажені, адже веб-аплікація очікує ресурси з “example.com”, хоча на сервері є вся необхідна інформація.

Під час інспекції веб-сторінки дійсно можна спостерігати наявність ресурсів на сервері, проте встановлена нами політика завантаження цих ресурсів забороняє їхнє завантаження, таким чином, блокуючи їх. Цей приклад демонструє, як саме CSP [196] блокує ресурси на практиці, та як це може бути корисним для забезпечення загальної безпеки веб-сервісу.

Ключовим етапом вважається безпосередній процес створення веб-застосунку, адже під час реальної розробки програмного продукту недоцільно звертатися до ресурсів, які застосовуються для візуалізації клієнтської версії. В цілому, коректне CSP передбачає значну кількість модифікацій та тестування. У такому випадку доречним буде використання вмонтованої функції, яка відображає помилки CSP [196], але при цьому не перекриває доступ до контенту.

```
# frozen_string_literal: true

Rails.application.config.content_security_policy_report_only = true
```

Вказавши дану опцію, ми можемо бачити функціональний сайт (рис. 4.3):

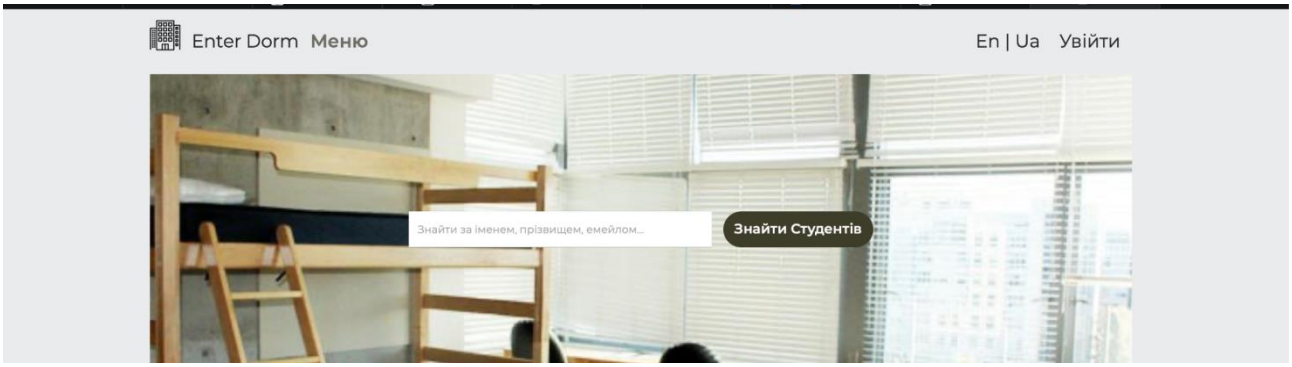


Рис. 4.3. Зовнішній вигляд стартової сторінки сайту

Хоча помилки будуть видимі для подальшого їх виправлення і пошуку причини їх появлення:

Важливим до розгляду випадком може слугувати вбудований код в HTML [166], хоча є декілька варіантів вирішення такого роду порушень.

- Перемістити весь вбудований код і вбудовані стилі до файлу. Даний підхід дозволяє покращити безпеку веб-сервісу та підтримувати його узгодженість.
- Перемістити код у тег і отримати його хеш-ключ, який генерується хешуванням коду всередині тегу за допомогою алгоритму SHA256 [151]. Досить зручно, що такий підхід надає можливість додавати конкретний хеш-ключ безпосередньо до директиви, повідомляючи браузеру, що код у цьому тегу дозволений.
- Використовувати атрибут тегу «nonce» [205] і додати його до відповідного тегу. Недоліком даного рішення можна вважати необхідність його оновлення для кожного запиту на стороні сервера.

Політика безпеки контенту (CSP) [196] у фреймворку Ruby on Rails [156] слугує захистом веб-сервісу від загроз, зокрема, XSS (міжсайтове скриптування) [179]. Це досягається шляхом керування завантаженням та обробкою конкретних ресурсів. Активований CSP [196] дозволяє вказувати, які джерела вважаються дозволеними для скриптів, стилів, зображень та інших типів вмісту. Такий метод зменшує ймовірність виконання зловмисного коду, навіть якщо атакуючий зуміє впровадити його безпосередньо у веб-застосунок. Варто відзначити, що Ruby on Rails [156] має вбудовану функціональність для CSP [196], що перетворює налаштування політики на швидкий та ефективний метод покращення безпеки.

4.7. Broken Access Control

Однією з найважливіших проблем безпеки, на яку варто звертати особливу увагу та докладати необхідних зусиль для її усунення, є BrokenAccessControl[194]. Небезпека полягає в тому, що зловмисник має

можливість отримати надмірний рівень доступу до застосунку, що, у свою чергу, може призвести до несанкціонованого витоку або спотворення даних.

Сторінка входу в веб-застосунок або інший ресурс може служити прикладом контролю доступу. Контроль доступу, згідно з назвою, — це сукупність правил і механізмів, які застосунок впроваджує для регулювання та обмеження доступу. Цей процес широко відомий як авторизація [52]. Зазвичай, після того, як сервер ідентифікує користувача за допомогою механізму входу або аутентифікації [206], він дозволяє або блокує доступ до певних функцій та ресурсів у системі. Також контроль доступу є основою для відстеження дій користувачів та моніторингу ресурсів.

Для ефективного впровадження надійної та безпечної системи контролю доступу, потрібні доволі значні зусилля. Контроль доступу до веб-додатків тісно переплітається з архітектурою та функціями, які він повинен реалізовувати. Крім того, користувачі на більшості платформ часто виконують кілька ролей, що збільшує складність обмежень.

Розробка належного контролю доступу в додатках може бути викликом. В залежності від масштабу та складності системи, відповідним рішенням може бути впровадження простого рішення автентифікації [206], використання бібліотеки третьої сторони з відкритим кодом, або поєднання обох. У Ruby on Rails [156] існують такі популярні та надійні рішення, як Devise [159], Cancancan [207] та AuthLogic [208].

Варто розглянути те, як зловмисники намагаються використати вразливі місця, щоб отримати доступ до аплікації.

Порушення контролю доступу, по суті, об'єднує низку експлойтів [209] або вразливостей, які є відомими та можуть нести загрозу для контролю доступу веб-сервісу. Чимало недоліків є досить простими у виправленні, проте якщо їх не помітити, зловмисники можуть легко ними скористатися. Крім того, наслідки порушення контролю доступу можуть бути фатальними, адже зловмисники мають змогу читати, змінювати або навіть видаляти контент і заволодівати застосунком. З огляду на це, відчувається нагальна потреба у

вирішенні таких проблем. Для ілюстрації візьмемо до уваги незахищені ідентифікатори [211], обхід шляху, дозволи на файли [212] та клієнтське кешування [210].

Щоб веб-застосунок зберігав інформацію про користувачів, потрібна база даних, а це передбачає використання ідентифікаторів для розпізнавання об'єктів. Більшість сучасних веб-сайтів застосовують певний формат ідентифікатора чи індексу для швидкого та ефективного опрацювання даних. Як згадувалось раніше, доступ до певних ресурсів не завжди є загальнодоступним. Тому, якщо веб-сервіс дозволяє вводити ідентифікатор у параметрах URL, і його просто передбачити, виникає потенційна загроза. Наприклад, за посиланням https://www.website.com/profile?id=12345 [201] формується профіль користувача. Якщо вручну змінити номер ідентифікатора у рядку запиту, а на сервері відсутній контроль доступу для перевірки запитувача, зловмисник може дістати доступ до будь-яких профілів на сайті.

Обхід шляху передбачає навігацію деревом каталогів файлової системи [59]. Загалом це означає, що системи, які дозволяють користувачеві отримати доступ до ресурсів файлової системи сервера, є ненадійними. Ця ситуація особливо критична, якщо шлях не перевіряється сервером. Прикладом може слугувати наступне URL-посилання [201]: https://www.website.com/photos?name=avatar.jpg. У цьому випадку зловмисник може змінити avatar.jpg навіть на ../../etc/passwd і таким чином спробувати отримати доступ до конфіденційних даних.

Окреслюючи попередню проблему, проблема доступу до файлів залежить від механізму дозволів [212] файлової системи сервера. Серверна частина застосунку зберігає багато важливих конфігураційних файлів [214] та ресурсів, що не підлягають змінам і, зазвичай, повинні бути недоступними для читання та виконання для користувачів. Проте, якщо конфігурація дозволів не налаштована належним чином, зловмисник може спробувати отримати доступ до цих файлів, повністю скомпрометувавши застосунок. Приклад спроби

отримати доступ до файлу, який повинен бути недоступним для читання: https://www.website.com/photos?name=../../Rakefile.

Цілком вірогідно, що частина потенційних користувачів веб-застосунку може використовувати один комп'ютер спільно з іншими особами. Цей аспект розробники не в змозі контролювати, і саме він зумовлює вразливість клієнтського кешу [210]. Вирішення проблеми з кешуванням на боці клієнта є складним завданням, адже зловмисник може скористатися сесійними даними [215] або кешованими сторінками чи даними, що вже наявні в веб-браузері автентифікованого [206] користувача. Тобто, зловмисник повинен отримати фізичний доступ до комп'ютера жертви. У такому випадку він матиме змогу завдати користувачу шкоди.

Дієвим методом захисту від подібних загроз може бути застосування бібліотеки "devise" [159], яку можливо інтегрувати у фреймворк Ruby on Rails [156].

```
# Devise is a flexible authentication solution for Rails based on Warden
gem 'devise', '~> 4.8'
```

Прикладом виправлення вразливостей пов’язаних із незахищеними ідентифікаторами може бути додаткова перевірка чи поточний ідентифікатор є справді тим, що і був в самому запиті.

```
# frozen_string_literal: true

class UserPolicy < ApplicationPolicy
  def deny_view_student?(admin)
    @user&.id != @record.id && !admin
  end
  ...
end
```

```ruby
# frozen_string_literal: true

class StudentsController < ApplicationController
  before_action :authenticate_student, except: :index
  before_action :find_student, except: %i[index new]

  def show
    if policy.deny_view_student?(current_admin)
      flash[:error] = t('flash.no_access_student')

      redirect_back(fallback_location: students_path)
    end
  end
  ...
end
```

Ми можемо помітити, як у методі відбувається порівняння ідентифікатора студента з тим, що зараз розглядається.

Для ліквідації ризиків доступу до файлів необхідні ґрунтовні знання конфігурації та адміністрування сервера. Проте, якщо відсутня потреба в власних конфігураціях, то жодні зміни не потрібні. Як і раніше, фреймворк Ruby on Rails [156] бере на себе значну частину роботи.

З точки зору кешування на клієнтській стороні, найбільш оптимальним рішенням є найпростішим. Не слід зберігати конфіденційну інформацію користувача безпосередньо на його пристрої. У випадку, якщо така потреба дійсно виникає, можна вдатися до використання коректних HTML мета-тегів [166] та асинхронної перевірки для підтвердження необхідних дозволів.

4.8. Загальні безпекові принципи функціонування автоматизованої системи

Детально розглянувши список безпекових загроз, може виникнути логічне запитання: як саме проектувати та розробляти веб-сервіс, щоб мінімізувати ризик виникнення різного роду атак, а саме тому варто розглянути наступні безпекові принципи:

1. Принцип найменших привілеїв та розподілу обов’язків.

Цей принцип безпеки [222] зосереджується на наданні користувачам тільки мінімально необхідного доступу для виконання їхніх обов'язків. Отже, кінцеві користувачі мають мати доступ лише до тих ресурсів, що потрібні для їхньої роботи, не більше. Такий підхід сприяє зменшенню ймовірності надмірного доступу до конфіденційної інформації чи систем, бо користувачі мають доступ тільки до того, що їм безпосередньо потрібно. "Найменший привілей" – ключовий принцип безпеки, якого слід дотримуватися для забезпечення безпеки даних і систем застосунку.

Щодо розподілу обов'язків, то це можна вважати основним принципом внутрішнього контролю в організаціях. Даний принцип можна охарактеризувати як систему стримувань, яка в свою чергу гарантує, що жодна особа не може мати контролю над усіма процесами певної транзакції і це досягається шляхом призначення різних завдань різним суб’єктам, щоб ніхто не міг контролювати весь процес. Такий підхід допомагає зменшити ризик шахрайства та помилок.

2. Принцип глибокої оборони.

Принцип ешелонованої оборони [223] передбачає багаторівневий підхід до забезпечення безпеки конкретного застосунку. В основі лежить концепція, згідно з якою у випадку компрометації одного рівня безпеки, решта рівнів повинні зіграти вирішальну роль у захисті сервісу. Серед прикладів таких рівнів - фізична безпека, безпека мережі [216], безпека програмного забезпечення та безпека даних. Мета ешелонованої оборони – створити безпечне середовище, здатне протистояти потенційним атакам, а також оперативно виявляти будь-які інциденти безпеки та адекватно реагувати на них. Застосування кількох рівнів захисту може зменшити ймовірність атаки, а у разі її успіху – мінімізувати збитки.

3. Принцип нульової довіри.

Нульова довіра – це концепція безпеки, згідно з якою кожен користувач, пристрій та мережа автоматично вважаються небезпечними, і їхню ідентичність необхідно верифікувати перед будь-яким наданням прав доступу. Цей підхід

базується на переконанні, що система не повинна сліпо довіряти жодному учаснику: ні користувачеві, ні пристрою, ні мережі. Замість цього, кожний запит на доступ повинен пройти аутентифікацію та авторизацію [205] перш ніж отримати відповідні дозволи. Крім того, модель нульової довіри передбачає безперервний моніторинг та аналіз дій користувачів, щоб гарантувати надання доступу виключно тим, хто його потребує. Її мета – знизити вірогідність витоків чутливої інформації та інших інцидентів безпеки, шляхом забезпечення доступу до даних лише уповноваженим користувачам.

4. Принцип безпеки відкритого типу.

Security-in-the-Open – це концепція, котра підкреслює вагомість захисту у розробці програмного забезпечення з відкритим кодом. Цей підхід акцентує увагу на потребі для розробників розуміти безпекові наслідки власного коду та застосовувати відповідні заходи для забезпечення захищеності: використання безпечних технік програмування, тестування на вразливості та застосування безпечних інструментів розробки.

Варто також звернути увагу на деякі ключові аспекти написання безпечного програмного забезпечення:

Перевірка вхідних даних.

Справді важливим фактором може бути безпосередня перевірка всіх вхідних даних на очікуваний тип, формат і довжину перед їх обробкою. В такий спосіб можна запобігти таким загрозам, як SQL-ін’єкція [180] і переповнення буфера [218].

1. Обробка помилок

У разі виникнення помилок варто обробляти їх та винятки максимально безпечним способом і при цьому в жодному разі не розкривати конфіденційну інформацію про аплікації зловмиснику.

2. Автентифікація та авторизація

Загальним правилом буде запровадження надійних механізмів автентифікації та авторизації [206], щоб забезпечити доступ до конфіденційних даних і ресурсів лише авторизованим користувачам.

3. Криптографія

Варто використовувати криптографічні функції та протоколи для захисту даних під час їх передачі, як-от: HTTPS [219] і шифрування [218]. Варто зазначити, що рекомендовано визначити сталий період існування криптографічних змінних і по їхньому завершенню змінювати приватні ключі [166], щоб тим самим тримати безпеку.

4. Найменші привілеї

Рекомендовано використовувати принцип найменших привілеїв [169] під час написання програмного забезпечення, щоб програмний код і сама система, на якій він працює, отримували мінімальні права доступу, необхідні для виконання своїх функцій.

5. Безпечне керування пам’яттю

В багатьох випадках при розробці аплікацій краще використовувати мови високого рівня або належним чином керувати пам’яттю, щоб запобігти атакам, пов’язаним з пам’яттю, таким як переповнення буфера [217] та використання після звільнення [221].

6. Безпечне зберігання конфіденційних змінних

У коді слід уникати жорстко закодованих конфіденційних змінних, таких як: паролі, ключі шифрування, ключі доступу, а зберігати їх у надійному сховищі.

7. Тестування безпеки

Важливим кроком під час розробки та перед безпосереднім розгортанням систему є перевірка програмного забезпечення на наявність вразливостей.

8. Перевірка коду

Корисним також буде регулярна перевірка коду на наявність вразливостей у безпеці, наприклад за допомогою автоматизованих інструментів. Для фреймворку RubyonRails [156] такими інструментами можуть слугувати наступні бібліотеки: bundler-audit [180] (пошук вразливостей у вже встановлених бібліотеках), brakeman [182] (статичний аналіз коду), SourceClear від Veracode [195] (сторонній сервіс для виявлення вразливостей в аплікації).

9. Актуальність

Слід підтримувати програмне забезпечення в актуальному стані з найновішими версіями фреймворків та бібліотек, найкращими методами безпеки та виправленнями вразливостей, щоб забезпечити максимальну безпеку програмного забезпечення.

Висновок до розділу 4

Безпека веб-аплікацій є одним із найважливіших аспектів сучасної розробки програмного забезпечення, особливо у випадку з фреймворком RubyonRails, який широко використовується для створення динамічних та масштабованих систем. Практично будь-який веб-сервіс, що працює в мережі, завжди піддається потенційному ризику різного роду атак, спрямованих на компрометацію конфіденційності, цілісності та доступності даних. У цьому розділі було розглянуто одні з найбільш поширених типів загроз, які притаманні сервісам розробленим за допомогою фреймворку RubyonRails.

Зокрема, CrossSiteScripting (XSS) є однією з найпоширеніших атак, що дозволяє зловмисникам виконувати шкідливий код, написаний на мові програмування JavaScript, у браузерах користувачів. Дана загроза може призвести до викрадення сесій, крадіжки особистих даних або модифікації вмісту сторінки на той, який вказав зловмисник. Захист від XSS у фреймворку RubyonRails реалізується завдяки автоматичному екрануванню HTML через відповідні методи, що перешкоджає впровадженню небезпечних скриптів у сторінку.

Іншою поширеною загрозою є міжсайтові підробки запитів (CSRF), які дозволяють зловмисникам виконувати несанкціоновані дії від імені користувачів веб-аплікації. У фреймворку RubyonRails реалізований вбудований механізм CSRF-захисту у вигляді токенів, які додаються до кожного запиту і перевіряються сервером, що унеможливлює проведення подібних атак без доступу до відповідного токена.

Порушення контролю доступу (BrokenAccessControl) є серйозною вразливістю, оскільки дозволяє неавторизованим користувачам отримати доступ до ресурсів, які вони не повинні бачити або модифікувати взагалі. Основною проблемою є недостатня перевірка прав доступу на рівні програмного коду. Щоб вирішити вищезгадану проблему, фреймворк RubyonRails надає ефективні механізми контролю доступу для перевірки авторизації користувачів, а також можливість використання спеціалізованих бібліотек, наприклад, devise або CanCanCan, що дозволяють гнучко керувати правами доступу.

Ще однією не менш важливою, а часом і критичною загрозою, є SQL-ін'єкції (SQLInjection), які виникають через небезпечну обробку вхідних даних у запитах до бази даних. Потенційні зловмисники можуть вставити власний шкідливий SQL-код у стрічку запиту у веб-браузеру, що може спровокувати витік даних, їх модифікацію або навіть повне видалення бази даних, про що було описано вище. Варто згадати, що фреймворк RubyonRails автоматично надає можливість захисту від SQL-ін'єкцій завдяки використанню параметризованих запитів через ActiveRecord, однак розробникам потрібно зважати на використання прямих SQL-запитів без належного екранування даних, адже це може стати потенційним вразливим місцем.

Окрему загрозу становлять вразливості залежностей (DependencyVulnerabilities), які виникають через використання сторонніх бібліотек, що можуть містити вразливості, які напряму впливають на безпеку цілої веб-аплікації. Оскільки RubyonRails сильно залежить від екосистеми сторонніх бібліотек, то важливо регулярно оновлювати залежності до останніх версій, використовувати такі інструменти моніторингу вразливостей, як bundler-audit або Brakeman, а також ретельно перевіряти підтримку зовнішніх бібліотек перед їх використанням у власному програмному забезпеченні.

Таким чином, безпека системи веб-аплікації написаної на RubyonRails має бути побудована на комплексному підході, що включає використання вбудованих механізмів захисту, дотримання принципів безпечного кодування,

регулярний аудит безпеки та оновлення залежностей. Кожен тип загрози може спровокувати досить серйозні наслідки, починаючи від компрометації окремих акаунтів до повного порушення роботи системи. Саме тому під час безпосередньої розробки веб-аплікації необхідно слідувати безпековим принципам, інтегруючи безпеку на всіх етапах розробки.

Застосування таких практик, як LeastPrivilege (мінімально необхідні привілеї), DefenseinDepth (багаторівневий захист) та SecuritybyDefault (безпека за замовчуванням), дозволяє значно знизити ризики та створити стійку до атак веб-аплікацію. З огляду на динамічний розвиток кіберзагроз, важливо не лише впроваджувати базові заходи безпеки, але й постійно стежити за новими трендами у сфері кібербезпеки, адаптуючи систему до сучасних викликів.

У підсумку, безпека веб-аплікації написаної за допомогою фреймворку RubyonRails є безперервним процесом, що вимагає системного підходу та відповідальності. Лише поєднання якісного написання програмного забезпечення, використання перевірених інструментів та регулярного аналізу безпеки може гарантувати, що веб-сервіс залишатиметься захищеним від атак та відповідатиме сучасним стандартам кібербезпеки.

ВИСНОВКИ

У дисертаційній роботі розв’язано актуальне науково-прикладне завдання розроблення моделі інформаційної технології проектування автоматизованої системи забезпечення діяльності студентського містечка на підставі виокремлення і дослідження факторів впливу на вказаний процес та прогностичного оцінювання якості засобами нечіткої логіки.

В дисертації отримано такі результати:

1. Наведено порівняльну інформацію про розвиток інформаційних систем та технологій як за кордоном, так і в Україні.

2. Описано та охарактеризовано найбільш поширені інформаційні технології, які використовуються в освітньому процесі.

3. Здійснено ранжування факторів впливу та розроблено багаторівневі моделі вагових значень виокремлених факторів та побудовано моделі пріоритетного впливу на якість досліджуваних процесів.

4. Виконано оптимізацію моделі факторів впливу на процес проектування інформаційної технології автоматизованої системи забезпечення діяльності студентського містечка з використанням методу попарних порівнянь за допустимими межами критеріїв – максимальне значення відхилення (λmax = 10,06) та індексу узгодження (IU = 0,06). Проведено порівняльний аналіз результатів представлених методів ранжування та аналізу ієрархій та визначення кращого.

5. Запроектовано та розраховано альтернативні варіанти реалізації досліджуваного процесу проектування автоматизованої системи забезпечення діяльності студентського містечка. Найбільшу перевагу отримав проект *A*, який був орієнтований на безпека та зручність та відповідно має A = **6,913** порівняно з іншими варіантами *B* = **6,536** (орієнтований на функціональність і мобільність) та C = **6,221** (орієнтований виключно на аналітика та доступність).

6. Побудовано матриці попарних порівнянь рангів факторів для термів у точках поділу універсальних терм-множин, згідно з якими отримано нормовані

значення функцій належності лінгвістичних змінних процесів. Здійснено графічне відображення розрахованих функцій належності.

7. Запроектовано нечіткі бази знань та сформовано нечіткі матриці знань для відповідних лінгвістичних змінних. Побудовано системи нечітких логічних рівнянь, які визначають процедури отримання функцій належності для множини лінгвістичних термів.

8. Використовуючи метод центру мас, виконано дефазифікацію нечітких множин та розраховано кількісні показники рівня прогнозованої якості розглянутих процесів. Отримано наступні числові значення рівнів якості реалізації проектування автоматизованої системи забезпечення діяльності студентського містечка — Q = 56,37%.

9. Розроблено структурно-функціональну модель інформаційної технології формування якості реалізаціїпроцесупроектування автоматизованої системи забезпечення діяльності студентського містечка.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Порядок поселення студентів у студентські гуртожитки» НТУУ «КПІ». URL: https://studprofkom.kpi.ua/polozhennya-pro-poselennya-ta-prozhyvannya/.

2. Положення про студентське містечко Київського національного економічного університету імені Вадима Гетьмана. URL:https://drive.google.com/file/d/1q30i64MhptqxylfSEcVusa-rAy6oU9Ny/view.

3. Положення про порядок поселення та проживання в гуртожитках студмістечка НПУ імені М. П. Драгоманова. URL:https://npu.edu.ua/images/Profkom_document/document_NPU/%D0%9F%D0%BE%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%BD%D1%8F_%D0%BF%D0%BE%D1%81%D0%B5%D0%BB%D0%B5%D0%BD%D0%BD%D1%8F.pdf

4. Положення про студентський гуртожиток Центральноукраїнського національного технічного університету. URL:https://kntu.kr.ua/file/content/410/provision-on-student-hostel.pdf.

5. Державні санітарні правила і норми влаштування, утримання загальноосвітніх навчальних закладів. URL:https://mon.gov.ua/static-objects/mon/sites/1/npa/5a1fe801a0e83.pdf.

6. Веб-інформаційна система «Гуртожиток» Київського національного економічного університету імені Вадима Гетьмана. URL:https://fisit.kneu.edu.ua/ua/University_en/map/hurt1_5/.

7. Яремчук Н. А., Сікоза О. М., Кельберг С. М. Система дистанційного навчання // Патент на корисну модель № 42514. Дата публікації відомостей про видачу патенту 10.07.2009.

8. Колос К. Р. Система Moodle як засіб розвитку предметних компетентностей учителів інформатики в умовах дистанційної післядипломної освіти : дис. робота на здобуття наукового ступеня канд. пед. наук. — Київ, 2011.

9. Конюхов А. Д. Моделі інформаційної технології у дистанційному навчанні з використанням інфографіки. дис. робота на здобуття наукового ступеня канд. тех. наук. — Львів, 2021.

10. Рашевська Н. В. Мобільні інформаційно-комунікаційні технології навчання вищої математики студентів вищих технічних навчальних закладів : дис. робота на здобуття наукового ступеня канд. пед. наук. — Київ, 2011.

11. Юдкова К.В., Чернишина Г.Г. Класифікація інформаційних систем. "*Інформація і право*" № 3 (15)/2015. С. 92-98.

12. Академічний тлумачний словник української мови. – URL:http://sum.in.ua/s/systema

13. Bourgeois D. T., Smith J. L., Wang S., Mortati J. What is an Information system? [Електронний ресурс]. – pp. 3-8 / *Information Systems for Business and Beyond* // Open Textbooks. – Saylor Academy, 2019 – p. 323. – URL:https://digitalcommons.biola.edu/open-textbooks/1/

14. Zwass, V. Information system. [Електронний ресурс]. // Encyclopedia Britannica. – URL:https://www.britannica.com/topic/information-system.

15. Цифровізація української освіти: реалізація, проблеми і перспективи. URL:https://oplatforma.com.ua/article/16004-tsifrovizatsiya-ukrainskoi-osviti-realizatsiya-problemi-i-perspektivi

16. Державним стандартом ДСТУ 2392-94. URL:https://dbn.at.ua/_ld/11/1166_DSTU2392-94.pdf

17. КишЛ.М., КлочкоО.В., ПотаповаН.А. Інформаційнісистемиітехнологіїуправлінняорганізацією: Навч. посіб. – Вінниця: ВНАУ, 2014. – 212 с.

18. Класифікація інформаційних систем. URL: https://stboinf.wordpress.com/2011/05/28/Класифікація інформаційних систем/.

19. Прийняття управлінських рішень [Текст] : навчальний посібник / [Петруня Ю. Є., Говоруха В. Б., Літовченко Б. В. та ін.] ; за ред. Ю. Є. Петруні. – Дніпропетровськ : АМСУ, 2015. – 189 с.

20. Bourgeois D. T., Smith J. L., Wang S., Mortati J. What is an Information system? [Електронний ресурс]. – pp. 3-8 / *Information Systems for Business and Beyond* // Open Textbooks. – Saylor Academy, 2019 – p. 323. – URL: https://digitalcommons.biola.edu/open-textbooks/1/.

21. Types of Information System: TPS, DSS & Pyramid Diagram [Електронний ресурс] // Guru99, 2021. – URL:https://www.guru99.com/mis-types-information-system.html.

22. Класифікація інформаційних систем. URL: https://pidru4niki.com/13340203/menedzhment/klasifikatsiya_informatsiynih_sistem.

23. Діловодство у школі: дозволено вести класний журнал в електронній формі без потреби дублювання його на папері | Міністерство освіти і науки України. URL: https://mon.gov.ua/news/dilovodstvo-u-shkoli-dozvoleno-vesti-klasniy-zhurnal-v-elektronniy-formi-bez-potrebi-dublyuvannya-yogo-na-paperi.

24. Сучасні технології навчання у вищій школі : модульний посібник для слухачів авторських курсів підвищення кваліфікації викладачів МІПК ПУЕТ / В. Ю. Стрельніков, І. Г. Брітченко. – Полтава : ПУЕТ, 2013. – 309 с.

25. Ефективність інформаційних систем : навч. посібник / В.А. Нецветаєв, Є.В. Кочура, Л.А. Манелюк. МОН України, Нац. гірн. унів. Д. : НГУ. 2014. - 190с.

26. Цифровізація української освіти: реалізація, проблеми і перспективи / URL:https://oplatforma.com.ua/article/16004-tsifrovizatsiya-ukrainskoi-osviti-realizatsiya-problemi-i-perspektivi#:~:text=Цифровізація освіти – один із, автоматизацією збору і аналізу даних.

27. Діловодство у школі: дозволено вести класний журнал в електронній формі без потреби дублювання його на папері. URL: https://www.google.com.ua/search?q=https%3A%2F%2Fmon.gov.ua%2Fnews%2Fdilovodstvo-u-shkoli-dozvoleno-vesti-klasniy-zhurnal-v-elektronniy-formi-bez-potrebi-dublyuvannya-yogo-na-paperi%23%3A%7E%3Atext%3D%25D0%25B9%25D0%25BE%25D0%25B3%25D0%25BE%2520%25D0%25BD%25D0%25B0%2520%25D0%25BF%25D0%25B

0%25D0%25BF%25D0%25B5%25D1%2580%25D1%2596%2C%25D0%25B0%25
D0%25BF%25D0%25B0%25D1%2580%25D0%25B0%25D1%2582%25D0%25B
D%25D0%25BE%25D0%25B3%25D0%25BE%2520%25D0%25BA%25D0%25BE
%25D0%25BC%25D0%25BF%25D0%25BB%25D0%25B5%25D0%25BA%25D1
%2581%25D1%2583&sca_esv=6983e842059c9cf0&sxsrf=AHTn8zpCn3e7yY79zjx
oWW4wmAA9n6bUmQ%3A1745605764015&source=hp&ei=g9QLaKWIO7iswPA
PmJHmkQo&iflsig=ACkRmUkAAAAAaAvilGAUcawclqucKTqQ00hOkNcc0rfT&
ved=0ahUKEwjl1eqb6POMAxU4FhAIHZiIOaIQ4dUDCBc&uact=5&oq=https%3A
%2F%2Fmon.gov.ua%2Fnews%2Fdilovodstvo-u-shkoli-dozvoleno-vesti-klasniy-
zhurnal-v-elektronniy-formi-bez-potrebi-dublyuvannya-yogo-na-
paperi%23%3A%7E%3Atext%3D%25D0%25B9%25D0%25BE%25D0%25B3%25
D0%25BE%2520%25D0%25BD%25D0%25B0%2520%25D0%25BF%25D0%25B
0%25D0%25BF%25D0%25B5%25D1%2580%25D1%2596%2C%25D0%25B0%25
D0%25BF%25D0%25B0%25D1%2580%25D0%25B0%25D1%2582%25D0%25B
D%25D0%25BE%25D0%25B3%25D0%25BE%2520%25D0%25BA%25D0%25BE
%25D0%25BC%25D0%25BF%25D0%25BB%25D0%25B5%25D0%25BA%25D1
%2581%25D1%2583&gs_lp=Egdnd3Mtd2l6ItUCaHR0cHM6Ly9tb24uZ292LnVhL
25ld3MvZGlsb3ZvZHN0dm8tdS1zaGtvbGktZG96dm9sZW5vLXZlc3RpLWtsYXN
uaXktemh1cm5hbC12LWVsZWt0cm9ubml5LWZvcm1pLWJlei1wb3RyZWJpLWR
1Ymx5dXZhbm55YS15b2dvLW5hLXBhcGVyaSM6fjp0ZXh0PSVEMCVCOSVE
MCVCRSVEMCVCMyVEMCVCRSUyMCVEMCVCRCVEMCVCMCUyMCVEMCVCMCVE
MCVCRiVEMCVCMCVEMCVCRiVEMCVCNSVEMSU4MCVEMSU5NiwlRDA
lQjAlRDAlQkYlRDAlQjAlRDElODAlRDAlQjAlRDElODIlRDAlQkQlRDAlQkUl
RDAlQjMlRDAlQkUlMjAlRDAlQkElRDAlQkUlRDAlQkMlRDAlQkYlRDAlQkIl
RDAlQjUlRDAlQkElRDElODElRDElODMyBxAuGCcY6gIyBxAjGCcY6gIyBxAj
GCcY6gIyBxAjGCcY6gIyBxAjGCcY6gIyBxAjGCcY6gIyBxAjGCcY6gIyBxAjGC
cY6gIyBxAjGCcY6gIyBxAjGCcY6gJI0i1QsxNYsxNwAXgAkAEAmAEAoAEAqg
EAuAEDyAEA-AEC-
AEBmAIBoAKCC6gCCpgDggvxBZLfoEiUhc3pkgcDNy0xoAcAsgcAuAcA&sclie
nt=gws-wizï.

28. Лист МОН України №1/11-3169 від 06.05.2020. URL: htttp//moiashkola.ua.

29. Безкоштовні електронні класні журнали та щоденники з можливостями дистанційного навчання.URL: https://moiashkola.ua/.

30. Безкоштовні електронні класні журнали та щоденники з можливостями дистанційного навчання. URL: https://nz.ua/.

31. NZ.UA - Електронні журнали. URL: https://play.google.com/store/apps/details?id=ua.nz.NZClient&hl=uk&pli=1.

32. Державні безкоштовні електронні щоденники та журнали для закладів загальної середньої освіти. URL: https://e-journal.iea.gov.ua/.

33. Цифровізація української освіти: реалізація, проблеми і перспективи. URL: https://oplatforma.com.ua/article/16004-tsifrovizatsiya-ukrainskoi-osviti-realizatsiya-problemi-i-perspektivi.

34. Соколов В. Ю. Інформаційні системи і технології : Навч. посіб. - К. : ДУІКТ, 2010. - 138 с.

35. David T. Bourgeois, James L. Smith, Shouhong Wang & Joseph Mortati, Biola University via Saylor Foundation. Що таке інформаційна система? URL: https://ukrayinska.libretexts.org/%D0%A0%D0%BE%D0%B1%D0%BE%D1%87%D0%B0_%D1%81%D0%B8%D0%BB%D0%B0/%D0%9A%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D1%96_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%B8_%D1%82%D0%B0_%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D1%96_%D1%82%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D1%96%D1%97/%D0%86%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D1%96_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B8/%D0%86%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D1%96_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B8_%D0%B4%D0%BB%D1%8F_%D0%B1%D1%96%D0%B7%D0%BD%D0%B5%D1%81%D1%83_%D1%82%D0%B0_%D0%B7%D0%B0_%D0

%B9%D0%BE%D0%B3%D0%BE_%D0%BC%D0%B5%D0%B6%D0%B0%D0%BC%D0%B8_(Bourgeois)_(%D0%B2%D0%B8%D0%B4%D0%B0%D0%BD%D0%BD%D1%8F_2019)/01%3A_%D0%A9%D0%BE_%D1%82%D0%B0%D0%BA%D0%B5_%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0%3F/100%3A_%D0%A9%D0%BE_%D1%82%D0%B0%D0%BA%D0%B5_%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0%3F.

36. R. Awati. What is an information system (IS)? URL: https://www.techtarget.com/whatis/definition/IS-information-system-or-information-services.

37. Носенко Ю.Г. Адаптивні системи навчання: сутність, характеристика, стан використання у вітчизняних закладах педагогічної освіти. *Науковий журнал «Фізико-математична освіта*». випуск 3(17), 2018. С. 73-78.

38. Колос К. Р. Система Moodle як засіб розвитку предметних компетентностей учителів інформатики в умовах дистанційної післядипломної освіти : дис. робота на здобуття наукового ступеня канд. пед. наук. — Київ, 2011.

39. Яремчук Н. А., Сікоза О. М., Кельберг С. М. Система дистанційного навчання // Патент на корисну модель № 42514. Дата публікації відомостей про видачу патенту 10.07.2009.

40. Герасименко І. В. Психолого-педагогічні технології дистанційного навчання в процесі ЗВО // URL: https://lib.iitta.gov.ua/id/eprint/705731/1/%D0%93%D0%B5%D1%80%D0%B0%D1%81%D0%B8%D0%BC%D0%B5%D0%BD%D0%BA%D0%BE_%D0%93%D0%B0%D0%BB%D0%B0%D1%81%D1%83%D0%BD_%D0%93%D0%BB%D1%83%D1%89%D0%B5%D0%BD%D0%BA%D0%BE.pdf

41. Методика формування самостійності майбутніх учителів музики у процесі дистанційного навчання / З. С. Дубовий // Інноваційна педагогіка. -

2019. - Вип. 11(1). - С. 131-136. - URL: http://nbuv.gov.ua/UJRN/innped_2019_11%281%29__30.

42. Рашевська Н. В. Мобільні інформаційно-комунікаційні технології навчання вищої математики студентів вищих технічних навчальних закладів : дис. робота на здобуття наукового ступеня канд. пед. наук. — Київ, 2011.

43. Конюхов А. Д. Моделі інформаційної технології у дистанційному навчанні з використанням інфографіки. дис. робота на здобуття наукового ступеня канд. тех. наук. — Львів, 2021.

44. Форуми та чати. URL: https://informatikahgj6.blogspot.com/2016/03/blog-post_31.html.

45. Зелінська О.В., Потапова Н.А., Волонтир Л.О. Інформаційні системи та технології в галузі. Навчальний посібник. / О.В. Зелінська, Н.А. Потапова, Л.О. Волонтир, - Вінниця: ВНАУ, 2020. – 263 с.

46. Бутенко Т. А., Сирий В. М. Інформаційні системи та технології : навчальний посібник. Харків: ХНАУ ім. В.В. Докучаєва, 2020. 207 с.

47. Шерман М. І., Самчинська Я. Б., Корнієнко Ю. М. Розробка інформаційної системи професійної підготовки здобувачів вищої освіти в умовах цифрового освітнього середовища. Open educationale-environment of modern University, № 11 (2021). С. 184-200.

48. Самчинська Я.Б., Шерман М.І., Сікелінда М.О. Імлементація теми з розробки чатботів в університетський курс «Офісні комп'ютерні технології». Електронне наукове фахове видання «Відкрите освітнє е-середовище сучасного університету». 2020. №9. С.121-133. URL: https://openedu.kubg.edu.ua/journal/index.php/openedu/article/view/311//

49. ШишкінаМ.П. Методологічні засади проектування хмаро-орієнтованого освітньо-наукового середовища закладу вищої освіти. Інформаційні технології в освіті. 2019. №4 (41). С.21-33.

50. Карпенко А. В. Розвиток інтелектуальних активів людського потенціалу: теорія та практика: монографія. Запоріжжя: ФОП В.В. Мокшанов, 2018. 510 с. URL:https://jmonographs.donnu.edu.ua/issue/view/217.

51. Savchenko N., Sherman M., Arystova L., Tymkiv L., Revenko N., & Mordovtseva N. Psychological and pedagogical aspects of management of activation of cognitive activity of applicants for higher education. Laplage in Journal. 2021. 7(3). pp. 607-615. URL:https://doi.org/10.24115/S2446-62202021731348p.607-615.

52. Бутенко Т. А., Сирий В. М.Інформаційні системи та технології : навчальний посібник. Харків: ХНАУ ім. В.В. Докучаєва, 2020. 207 с.

53. Проектування інформаційних систем: Загальні питання теорії проектування ІС (конспект лекцій): навч. посіб. для студ. спеціальності 122 «Комп'ютерні науки» / КПІ ім. Ігоря Сікорського; уклад.: О. С. Коваленко, Л. М. Добровська.– Київ : КПІ ім. Ігоря Сікорського, 2020. – 192с.

54. Зінов'єва О.Г., Шаров С.В. Проектування інформаційних систем:конспект лекцій. Запоріжжя, 2024. – 148 с.

55. Терновий І. М., Хамула О. Г. Оптимізація ієрархічної моделі факторів впливу на проектування інформаційної системи для студмістечка /*Наукові записки УАД*. 2024. № 2 (69). 2024. с. 130-139. URL:doi:10.32403/1998-6912-2024-2-69-130-139.

56. Сеньківський В. М. Модель ієрархії критеріїв якості книжкових видань. / *Наукові записки УАД*. Львів, 2007. Вип. 11. С.73–80.

57. Лямець В. І. Системний аналіз. Вступний курс / В. І. Лямець, А. Д. Тевяшев. 2-е вид., переробл. та допов. Харків : ХНУРЕ, 2004. 448 с.

58. Хамула О.Г., Конюхов А.Д. Виокремлення факторів впливу композиційного оформлення WEB-додатку для дистанційної освіти. *ІІ Міжнародна науково-технічна конференціія «Поліграфічні, мультимедійні та WEB-технології»*: тези доповідей, Харків, 2017. С.235-236.

59. Хамула О.Г.,.Конюхов А.Д. Розробка моделі факторів впливу на експлуатаційні показники пластикових карток. / *Техніка і технологія друкарства.* Київ. 2016. №4(54). С.21-28.

60. Терновий І. М., Хамула О. Г. Аналіз та визначення пріоритетності факторів впливу на процес проектування інформаційної системи для

студмістечка./ *Поліграфія і видавнича справа*. № 2 (88). 2024. С. 83-94. URL:doi:10.32403/0554-4866-2024-2-88-83-94.

61. Сорока К. О. Основи теорії систем і системного аналізу: навч. посіб. 2-ге видання, перероблене і виправлене. Х. : Тимченко А. М., 2005. 286 с.

62. VasiutaS., KhamulaO.. Синтез моделі факторів композиційного оформлення інфографіки. / *Поліграфія і видавнича справа*. № 2 (74) Львів: УАД, 2017. С. 59-65.

63. VasiutaS., KhamulaO., TymchenkoO.. Optimization of the Mathematical Model of Factors of Composite Design of Infographic. *The materials at the 2018 IEEE 13th International scientific and technical conference Computer Science and Information Technologies CSIT 2018* in Lviv, Ukraine, 11-14 September, Volumes 2. pp. 58-61. ISBN^ 978-1-5386-6463-6.

64. Бутенко Т. А., Сирий В. М. Інформаційні системи та технології : навчальний посібник. Харків: ХНАУ ім. В.В. Докучаєва, 2020. 207 с.

65. Проектування інформаційних систем: Загальні питання теорії проектування ІС (конспект лекцій): навч. посіб. для студ. спеціальності 122 «Комп'ютерні науки» / КПІ ім. Ігоря Сікорського; уклад.: О. С. Коваленко, Л. М. Добровська. – Київ : КПІ ім. Ігоря Сікорського, 2020. – 192с.

66. Зінов'єва О.Г., Шаров С.В. Проектування інформаційних систем: конспект лекцій. Запоріжжя, 2024. – 148 с.

67. Терновий І. М., Хамула О. Г. Теоретичні засади проєктування інформаційної системи студмістечка. /*Комп'ютерні технології друкарства*. № 1 (51). 2024. С. 78-89. URL:doi. 10.32403/2411-9210-2024-1-51-78-89.

68. Ternovyy I., Khamula O. Features of creating news in the campus information system. *Ricerche scientifiche e metodi della loro realizzazione: esperienza mondiale e realtà domestiche: збірник наукових праць «ΛΌГОΣ» з матеріалами VI Міжнародної науково-практичної конференції*, м. Болонья, 15 листопада 2024 р. – Вінниця-Болонья: ТОВ «УКРЛОГОС Груп», Associazione Italiana di Storia Urbana, 2024. pp. 179-181.

69. Gehlawat, M. School Management Information System: An Effective Tool for Augmenting the School Practices // *New Frontiers hn Education: International Journal of Education & Research*. – №47. – pp. 57-64. – URL: https://www.researchgate.net/publication/315380267_School_Management_Information_System_An_Effective_Tool_for_Augumenting_the_School_Practices.

70. Lewy J, Management Information Systems in Education // 2020. – URL: https://www.iris.co.uk/blog/management-information-systems-in-education/.

71. Львов М. С., Співаковський О. В., Щедролосьєв Д. Є. Інформаційна система управління вищим навчальним закладом як платформа реалізації управління навчальним процесом. *Вісник Харківського національного університету*, 2005, с. 1-21.

72. Шерман М. І., Самчинська Я. Б., Корнієнко Ю. М. Розробка інформаційної системи професійної підготовки здобувачів вищої освіти в умовах цифрового освітнього середовища. *Open educationale-environment of modern University*, № 11 (2021). С. 184-200. URL: https//doi: 10.28925/2414-0325.2021.1116.

73. Бушуєв С. Д., Цюцюра М. І. Методологія розробки та принципи функціонування інформаційної технології гармонізації змісту освіти. *Інформаційні технології і засоби навчання*, 2018, Том 63, №1. С. 12-21.

74. Ternovyy I. Khamula O. DETERMINATION AND ANALYSIS OF PARAMETERS THAT INFLUENCE THE DESIGN OF THE STUDENT CAMPUS MANAGEMENT INFORMATION SYSTEM /*Комп'ютерні технології друкарства*. 2024. Vol. 2. no. 52. P. 178-193. URL: https//doi:10.32403/2411-9210-2024-2-52-178-193.

75. Терновий І. М., Хамула О. Г. / Розробка веб сервісу для студентського містечка. / *Тези доповідей наук.-техн конференції професорсько-викладацького складу, наукових працівників і аспірантів* (07.02.-11.02.2022р.). Львів: УАД. 2022. С.147.

76. Терновий І. М., Хамула О. Г. / Моніторинг систем веб сервісу за допомогою ELASTIC OBSERVABILITY / *Тези доповідей наук.-техн.*

конференції професорсько-викладацького складу, наукових працівників і аспірантів (06.02.-10.02.2023р.). Львів: УАД. 2023. С.69.

77. Dave Thomas, Chad Fowler, Andy Hunt, 2013.- 886c. - (Programming Ruby 1.9 & 2.0 (4th edition))

78. Sam Ruby; Pragmatic Bookshelf, 2016. - 488c. - (Agile Web Development with Rails 5).

79. Use case diagram [Electronic resource]. - URL: https://www.geeksforgeeks.org/system-design/use-case-diagram/.

80. Терновий І. Магістерська кваліфікаційна робота. ЛНУ ім. Ів. Франка. 2022р.

81. Ruby on Rails framework [Electronic resource]. - URL:https://github.com/rails/rails

82. Ruby programming language [Electronic resource]. - URL:https://github.com/ruby/ruby.

83. ActionMailer::Base [Electronic resource]. - URL:https://api.rubyonrails.org/v5.2.3/classes/ActionMailer/Base.html.

84. Devise gem [Electronic resource]. - URL:https://github.com/plataformatec/devise.

85. Carrierwave gem [Electronic resource]. - URL:https://github.com/carrierwaveuploader/carrierwave.

86. Background process [Electronic resource]. - URL:https://learn.microsoft.com/en-us/azure/architecture/best-practices/background-jobs.

87. Sidekiq [Electronic resource]. URL:https://sidekiq.org/.

88. Rails I18n [Electronic resource]. - URL:https://guides.rubyonrails.org/i18n.html.

89. Scss [Electronic resource]. - URL:https://sass-lang.com.

90. HTML [Electronic resource]. - URL:https://developer.mozilla.org/uk/docs/Web/HTML.

91. CoffeeScript [Electronic resource]. - URL:https://coffeescript.org/.

92. JavaScript [Electronic resource]. - URL:https://developer.mozilla.org/en-US/docs/Web/JavaScript.

93. PostgreSQL [Electronic resource]. - URL:https://www.postgresql.org/.

94.Ruby on Rails guides [Electronic resource]. - URL:https://guides.rubyonrails.org/.

95. Pat Shaughnessy, 2013.- 336c. - (Ruby Under a Microscope: An Illustrated Guide to Ruby Internals).

96. WebSocket [Electronic resource]. - URL:https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API.

97. ActionCable [Electronic resource]. - URL:https://api.rubyonrails.org/classes/ActionCable.html.

98. Rubocop [Electronic resource]. - URL:https://github.com/rubocop/rubocop

99.CLI [Electronic resource]. - URL:https://www.w3schools.com/whatis/whatis_cli.asp.

100. SQL Injection [Electronic resource]. - URL:https://www.w3schools.com/sql/sql_injection.asp.

101. CSRF [Electronic resource]. - URL:https://owasp.org/www-community/attacks/csrf.

102. Securing Rails Applications [Electronic resource]. - URL:https://guides.rubyonrails.org/security.html.

103. XSS [Electronic resource]. - URL:https://owasp.org/www-community/attacks/xss/.

104. bundler-audit gem [Electronic resource]. - URL:https://github.com/rubysec/bundler-audit

105. Puma gem [Electronic resource]. - URL:https://github.com/puma/puma

106. Brakeman gem [Electronic resource]. - URL:https://github.com/presidentbeef/brakeman

107. Heroku [Electronic resource]. - URL:https://heroku.com/

108. CI/CD [Electronic resource]. - URL:https://www.redhat.com/en/topics/devops/what-is-ci-cd

109. Github [Electronic resource]. - URL:https://github.com/

110. Github Workflows [Electronic resource]. - URL:https://docs.github.com/en/actions/learn-github-actions/workflow-syntax-for-github-actions

111. Heroku CLI [Electronic resource]. - URL:https://devcenter.heroku.com/articles/heroku-cli

112. Redis [Electronic resource]. - URL:https://redis.io

113. Sendgrid [Electronic resource]. - URL:https://sendgrid.com

114. Procfile [Electronic resource]. - URL:https://devcenter.heroku.com/articles/procfile

115. SMTP server [Electronic resource]. - URL:https://sendgrid.com/blog/what-is-an-smtp-server/

116.Strong Parameters [Electronic resource]. - URL:https://api.rubyonrails.org/classes/ActionController/StrongParameters.html

117. Security Vulnerability [Electronic resource]. - URL:https://owasp.org/www-community/vulnerabilities/

118. Broken Access Control [Electronic resource]. - URL:https://owasp.org/Top10/A01_2021-Broken_Access_Control/

119. Veracode SourceClear [Electronic resource]. - URL:https://docs.veracode.com/r/t_sc_cli_agent

120. Content Security Policy. [Electronic resource]. - URL:https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP

121. CSP usage in Ruby on Rails. [Electronic resource]. - URL:https://api.rubyonrails.org/classes/ActionDispatch/ContentSecurityPolicy.html

122. ActiveRecord. [Electronic resource]. - URL:https://guides.rubyonrails.org/active_record_basics.html

123. ERB. [Electronic resource]. - URL:https://ruby-doc.org/stdlib-2.5.3/libdoc/erb/rdoc/ERB.html

124. URL. [Electronic resource]. - URL:https://developer.mozilla.org/en-US/docs/Web/API/URL

125. link_to. [Electronic resource]. - URL:https://apidock.com/rails/ActionView/Helpers/UrlHelper/link_to

126. ActionController::Parameters. [Electronic resource]. - URL:https://api.rubyonrails.org/classes/ActionController/Parameters.html

127. OpenSSL SHA256. [Electronic resource]. - URL:https://docs.openssl.org/3.1/man3/SHA256_Init/

128. “nonce” tag. [Electronic resource]. - URL:https://developer.mozilla.org/en-US/docs/Web/HTML/Global_attributes/nonce

129. Authentication vs. Authorization. [Electronic resource]. - URL:https://auth0.com/docs/get-started/identity-fundamentals/authentication-and-authorization

130. Cancancan gem. [Electronic resource]. - URL:https://github.com/CanCanCommunity/cancancan

131. AuthLogic gem. [Electronic resource]. - URL:https://github.com/binarylogic/authlogic

132. Exploit. [Electronic resource]. - URL:https://www.cisco.com/c/en/us/products/security/advanced-malware-protection/what-is-exploit.html

133. Server-side and client-side caching. [Electronic resource]. - URL:from: https://www.geeksforgeeks.org/server-side-caching-and-client-side-caching/

134. Insecure Direct Object Reference. [Electronic resource]. - URL:https://cheatsheetseries.owasp.org/cheatsheets/Insecure_Direct_Object_Reference_Prevention_Cheat_Sheet.html

135. File permissions in Linux. [Electronic resource]. - URL:https://www.redhat.com/en/blog/linux-file-permissions-explained

136. File system structure in Linux. [Electronic resource]. - URL:https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/6/html/storage_administration_guide/ch-filesystem#s1-filesystem-fhs

137. Configuration file. [Electronic resource]. - URL:https://www.redhat.com/en/topics/linux/what-configuration-file

138. Sessions. [Electronic resource]. - URL:https://auth0.com/docs/manage-users/sessions

139. Network security. [Electronic resource]. - URL:https://www.geeksforgeeks.org/network-security/

140. Buffer overflow. [Electronic resource]. - URL:https://www.cloudflare.com/en-gb/learning/security/threats/buffer-overflow/

141. Encryption. [Electronic resource]. - URL:https://cloud.google.com/learn/what-is-encryption?hl=en#

142. HTTPS protocol. [Electronic resource]. - URL:https://www.cloudflare.com/en-gb/learning/ssl/what-is-https/

143. Private key encryption. [Electronic resource]. - URL:https://www.sciencedirect.com/topics/computer-science/private-key-encryption

144. Use-after-free vulnerability. [Electronic resource]. - URL:https://learn.microsoft.com/en-gb/cpp/sanitizers/error-heap-use-after-free

145. Least privilege principle. [Electronic resource]. - URL:https://learn.microsoft.com/en-us/entra/identity-platform/secure-least-privileged-access

146. Defense in depth. [Electronic resource]. - URL:https://www.cloudflare.com/en-gb/learning/security/glossary/what-is-defense-in-depth/

147. Secure by Default. [Electronic resource]. - URL:https://blog.cloudflare.com/secure-by-default-understanding-new-cisa-guide/

148. Class diagram. [Electronic resource]. - URL:https://www.geeksforgeeks.org/unified-modeling-language-uml-class-diagrams/

149. MVC pattern. [Electronic resource]. - URL:https://developer.mozilla.org/en-US/docs/Glossary/MVC

150. Database. [Electronic resource]. - URL:https://aws.amazon.com/what-is/database/

151. Терновий І. М., Хамула О. Г. / Методи комунікації між сервісами в мікросервісній архітектурі / *Поліграфічні, мультимедійні та web-технології:*

тези доп. VIII Міжнародна. наук.-техн. конф. (16-20 травня 2023, м. Харків) Харків: ТОВ «Друкарня Мадрид», 2023. Т1. С. 94-95.

152. Ternovyy I. Khamula O. FEATURES OF CREATING NEWS IN THE CAMPUS INFORMATION SYSTEM. / *Ricerche scientifiche e metodi della loro realizzazione: esperienza mondiale e realtà domestiche: Raccolta di articoli scientifici «ΛΌГОΣ» con gli atti della VI Conferenza scientifica e pratica internazionale*, Bologna, 15 Novembre, 2024. Bologna-Vinnytsia: Associazione Italiana di Storia Urbana& UKRLOGOS Group LLC, 2024. pp. 179-181. doi:1036074/logos-15.11.2024.040.

153. Dave Thomas, Chad Fowler, Andy Hunt, 2013.- 886с. - (Programming Ruby 1.9 & 2.0 (4th edition)).

155. Sam Ruby; Pragmatic Bookshelf, 2016. - 488с. - (Agile Web Development with Rails 5).

154. Use case diagram [Electronic resource]. - URL:https://www.geeksforgeeks.org/system-design/use-case-diagram/.

156.Ruby on Rails framework [Electronic resource]. - URL:https://github.com/rails/rails

157. Ruby programming language [Electronic resource]. - URL:https://github.com/ruby/ruby

158. ActionMailer::Base [Electronic resource]. - URL:https://api.rubyonrails.org/v5.2.3/classes/ActionMailer/Base.html

159. Devise gem [Electronic resource]. - URL:https://github.com/plataformatec/devise

160. Carrierwave gem [Electronic resource]. - URL:https://github.com/carrierwaveuploader/carrierwave

161. Background process [Electronic resource]. - URL:https://learn.microsoft.com/en-us/azure/architecture/best-practices/background-jobs

162. Ternovyy I. Khamula O.Nevidomska B.General Security Principles in Web Service Design. *The VII International scientific and practical conference*

«Present and future: priority areas of research in scientific and educational activities», February 17-19, 2025, Prague, Czech Republic. pp. 201-206. URL:https://eu-conf.com/wp-content/uploads/2024/12/PRESENT-AND-FUTURE-PRIORITY-AREAS-OF-RESEARCH-IN-SCIENTIFIC-AND-EDUCATIONAL-ACTIVITIES.pdf.

163. 10. Sidekiq [Electronic resource]. - URL:https://sidekiq.org/

164.Rails I18n [Electronic resource]. - URL: https://guides.rubyonrails.org/i18n.html

165. Scss [Electronic resource]. - URL:https://sass-lang.com

166. HTML [Electronic resource]. - URL: https://developer.mozilla.org/uk/docs/Web/HTML

167. CoffeeScript [Electronic resource]. - URL:https://coffeescript.org/

168. JavaScript [Electronic resource]. - URL:https://developer.mozilla.org/en-US/docs/Web/JavaScript

169. PostgreSQL [Electronic resource]. - URL:https://www.postgresql.org/

170. Ruby on Rails guides [Electronic resource]. - URL: https://guides.rubyonrails.org/

171. Pat Shaughnessy, 2013.- 336c. - (Ruby Under a Microscope: An Illustrated Guide to Ruby Internals)

172. WebSocket [Electronic resource]. - URL:https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API.

173. ActionCable [Electronic resource]. - URL: https://api.rubyonrails.org/classes/ActionCable.html

174. Rubocop [Electronic resource]. - URL: https://github.com/rubocop/rubocop

175. CLI [Electronic resource]. - URL: https://www.w3schools.com/whatis/whatis_cli.asp

176. SQL Injection [Electronic resource]. - URL: https://www.w3schools.com/sql/sql_injection.asp

177. CSRF [Electronic resource]. - URL:https://owasp.org/www-community/attacks/csrf

178.Securing Rails Applications [Electronic resource]. - URL:https://guides.rubyonrails.org/security.html

179. XSS [Electronic resource]. - URL:https://owasp.org/www-community/attacks/xss/

180. bundler-audit gem [Electronic resource]. - URL: https://github.com/rubysec/bundler-audit

181. Puma gem [Electronic resource]. - URL:https://github.com/puma/puma

182. Brakeman gem [Electronic resource]. – URL :https://github.com/presidentbeef/brakeman

183. Heroku [Electronic resource]. - URL:https://heroku.com/

184. CI/CD [Electronic resource]. - URL: https://www.redhat.com/en/topics/devops/what-is-ci-cd

185. Github [Electronic resource]. - URL:: https://github.com/

186. Github Workflows [Electronic resource]. - URL: https://docs.github.com/en/actions/learn-github-actions/workflow-syntax-for-github-actions

187. Heroku CLI [Electronic resource]. - URL: https://devcenter.heroku.com/articles/heroku-cli

188. Redis [Electronic resource]. - URL:https://redis.io

189. Sendgrid [Electronic resource]. - URL:https://sendgrid.com

190. Procfile [Electronic resource]. - URL: https://devcenter.heroku.com/articles/procfile

191.SMTP server [Electronic resource]. - URL:https://sendgrid.com/blog/what-is-an-smtp-server/

192. Strong Parameters [Electronic resource]. - URL: https://api.rubyonrails.org/classes/ActionController/StrongParameters.html

193. Security Vulnerability [Electronic resource]. - URL: https://owasp.org/www-community/vulnerabilities/

194. Broken Access Control [Electronic resource]. - URL: https://owasp.org/Top10/A01_2021-Broken_Access_Control/

195. Veracode SourceClear [Electronic resource]. - URL: https://docs.veracode.com/r/t_sc_cli_agent

196. Content Security Policy. [Electronic resource]. - URL: https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP

197. CSP usage in Ruby on Rails. [Electronic resource]. - URL: https://api.rubyonrails.org/classes/ActionDispatch/ContentSecurityPolicy.html

198. Терновий І. М., Хамула О. Г. / Керування конфіденційними змінними за допомогою Hashicorp Vault / *Тези доповідей наук.-техн конференції професорсько-викладацького складу, наукових працівників і аспірантів* (05.02.-09.02.2024 р.). Львів: УАД. 2024. С. 211.

199. ActiveRecord. [Electronic resource]. - URL: https://guides.rubyonrails.org/active_record_basics.html

200. ERB. [Electronic resource]. - URL:https://ruby-doc.org/stdlib-2.5.3/libdoc/erb/rdoc/ERB.html

201. URL. [Electronic resource]. - URL:https://developer.mozilla.org/en-US/docs/Web/API/URL

202. link_to. [Electronic resource]. - URL: https://apidock.com/rails/ActionView/Helpers/UrlHelper/link_to

203. ActionController::Parameters. [Electronic resource]. - URL: https://api.rubyonrails.org/classes/ActionController/Parameters.html

204. OpenSSL SHA256. [Electronic resource]. - URL: https://docs.openssl.org/3.1/man3/SHA256_Init/

205. “nonce” tag. [Electronic resource]. - URL: https://developer.mozilla.org/en-US/docs/Web/HTML/Global_attributes/nonce

206. Authentication vs. Authorization. [Electronic resource]. - URL: https://auth0.com/docs/get-started/identity-fundamentals/authentication-and-authorization

207. Cancancan gem. [Electronic resource]. – URL: https://github.com/CanCanCommunity/cancancan

208. AuthLogic gem. [Electronic resource]. - URL: https://github.com/binarylogic/authlogic

209. Exploit. [Electronic resource]. - URL: https://www.cisco.com/c/en/us/products/security/advanced-malware-protection/what-is-exploit.html

210. Client-side caching. [Electronic resource]. - URL: https://www.geeksforgeeks.org/server-side-caching-and-client-side-caching/

211. Insecure Direct Object Reference. [Electronic resource]. - URL: https://cheatsheetseries.owasp.org/cheatsheets/Insecure_Direct_Object_Reference_Prevention_Cheat_Sheet.html

212. File permissions in Linux. [Electronic resource]. - URL: https://www.redhat.com/en/blog/linux-file-permissions-explained

213. File system structure in Linux. [Electronic resource]. - URL: https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/6/html/storage_administration_guide/ch-filesystem#s1-filesystem-fhs

214. Configuration file. [Electronic resource]. - URL: https://www.redhat.com/en/topics/linux/what-configuration-file

215. Sessions. [Electronic resource]. - URL:https://auth0.com/docs/manage-users/sessions

216. Network security. [Electronic resource]. - URL: https://www.geeksforgeeks.org/network-security/

217. Buffer overflow. [Electronic resource]. - URL: https://www.cloudflare.com/en-gb/learning/security/threats/buffer-overflow/

218. Encryption. [Electronic resource]. - URL: https://cloud.google.com/learn/what-is-encryption?hl=en#

219. HTTPS protocol. [Electronic resource]. - URL: https://www.cloudflare.com/en-gb/learning/ssl/what-is-https/

220. Private key encryption. [Electronic resource]. - URL: https://www.sciencedirect.com/topics/computer-science/private-key-encryption

221. Use-after-free vulnerability. [Electronic resource]. - URL: https://learn.microsoft.com/en-gb/cpp/sanitizers/error-heap-use-after-free

222. Least privilege principle. [Electronic resource]. - URL: https://learn.microsoft.com/en-us/entra/identity-platform/secure-least-privileged-access

223. Defense in depth. [Electronic resource]. - URL: https://www.cloudflare.com/en-gb/learning/security/glossary/what-is-defense-in-depth/

224. Secure by Default. [Electronic resource]. - URL: https://blog.cloudflare.com/secure-by-default-understanding-new-cisa-guide/

ДОДАТКИ

Додаток А

ПЕРЕЛІК ПУБЛІКАЦІЙ ЗДОБУВАЧА

Публікації в іноземних виданнях

1. Ternovyy I. Khamula O. FEATURES OF CREATING NEWS IN THE CAMPUS INFORMATION SYSTEM. / Ricerche scientifiche e metodi della loro realizzazione: esperienza mondiale e realtà domestiche: Raccolta di articoli scientifici «ΛΌΓΟΣ» con gli atti della VI Conferenza scientifica e pratica internazionale, Bologna, 15 Novembre, 2024. Bologna-Vinnytsia: Associazione Italiana di Storia Urbana& UKRLOGOS Group LLC, 2024. P. 179-181. URL:https://doi:1036074/logos-15.11.2024.040.

Публікації у виданнях, що входять до міжнародних наукометричних баз даних (Index Copernicus) та є науковими фаховими виданнями України:

2. Терновий І. М., Хамула О. Г. Теоретичні засади проєктування інформаційної системи студмістечка. / *Комп'ютерні технології друкарства*. № 1 (51). 2024. С. 78-89. URL:https://doi:10.32403/2411-9210-2024-1-51-78-89.

3. Терновий І. М., Хамула О. Г. Аналіз та визначення пріоритетності факторів впливу на процес проектування інформаційної системи для студмістечка. /*Поліграфія і видавнича справа*. № 2 (88). 2024. С. 83-94. URL:https://doi:10.32403/0554-4866-2024-2-88-83-94.

4. Терновий І. М., Хамула О. Г. Оптимізація ієрархічної моделі факторів впливу на проектування інформаційної системи для студмістечка /*Наукові записки УАД*. 2024. № 2 (69). 2024. с. 130-139. URL:https://doi:10.32403/1998-6912-2024-2-69-130-139.

5. Ternovyy I. Khamula O.DETERMINATION AND ANALYSIS OF PARAMETERS THAT INFLUENCE THE DESIGN OF THE STUDENT CAMPUS MANAGEMENT INFORMATION SYSTEM /*Комп'ютерні технології друкарства*. 2024. Vol. 2. no. 52. P. 178-193. URL:https://doi:10.32403/2411-9210-2024-2-52-178-193.

Публікації, які засвідчують апробацію матеріалів дисертації:

6. Терновий І. М., Хамула О. Г. / Розробка веб сервісу для студентського містечка. / *Тези доповідей наук.-техн конференції професорсько-викладацького складу, наукових працівників і аспірантів* (07.02.-11.02.2022р.). Львів: УАД. 2022. С.147.

7. Терновий І. М., Хамула О. Г. / Моніторинг систем веб сервісу за допомогою ELASTIC OBSERVABILITY / *Тези доповідей наук.-техн. конференції професорсько-викладацького складу, наукових працівників і аспірантів* (06.02.-10.02.2023р.). Львів: УАД. 2023. С.69.

8. Терновий І. М., Хамула О. Г. / Методи комунікації між сервісами в мікросервісній архітектурі / *Поліграфічні, мультимедійні та web-технології: тези доп. VIII Міжнародна. наук.-техн. конф*. (16-20 травня 2023, м. Харків) Харків: ТОВ «Друкарня Мадрид», 2023. Т1. С. 94-95.

9. Терновий І. М., Хамула О. Г. / Керування конфіденційними змінними за допомогою Hashicorp Vault / *Тези доповідей наук.-техн конференції професорсько-викладацького складу, наукових працівників і аспірантів* (05.02.-09.02.2024 р.). Львів: УАД. 2024. С. 211.

10. Ternovyy I. Khamula O.Nevidomska B.General Security Principles in Web Service Design. *The VII International scientific and practical conference «Present and future: priority areas of research in scientific and educational activities»,* February 17-19, 2025, Prague, Czech Republic. pp. 201-206. URL: https://eu-conf.com/wp-content/uploads/2024/12/PRESENT-AND-FUTURE-PRIORITY-AREAS-OF-RESEARCH-IN-SCIENTIFIC-AND-EDUCATIONAL-ACTIVITIES.pdf.

Додаток Б.

ВІДОМОСТІ ПРО АПРОБАЦІЮ РЕЗУЛЬТАТІВ ДИСЕРТАЦІЇ

− звітних науково-технічних конференціях професорсько-викладацького складу, наукових працівників і аспірантів Української академії друкарства (Львів, 2022–2024);

−VIII Міжнародній наук.-техн. конф. Поліграфічні, мультимедійні та web-технології(16-20 травня 2023, м. Харків);

−VI Conferenza scientifica e pratica internazionale. Ricerche scientifiche e metodi della loro realizzazione: esperienza mondiale e realtà domestiche: Raccolta di articoli scientifici «ΛΌΓΟΣ» (Bologna, 15 Novembre, 2024. Bologna-Vinnytsia);

−VII International scientific and practical conference «Present and future: priority areas of research in scientific and educational activities» (February 17-19, 2025, Prague, Czech Republic).

Додаток В.

ДОВІДКА ПРО ВИКОРИСТАННЯ В НАВЧАЛЬНОМУ ПРОЦЕСІ РЕЗУЛЬТАТІВ ДИСЕРТАЦІЙНОЇ РОБОТИ

Проректор з науково-педагогічної роботи
Української академії друкарства

_____ Угрин Я.М.

"24" 06 _____ 2024 р.

Д О В І Д К А

про використання матеріалів дисертації Тернового І. М.

«Інформаційна технологія проектування автоматизованої системи забезпечення діяльності студентського містечка»

в освітньому процесі Української академії друкарства

Підготовка спеціалістів в Українській академії друкарства орієнтована на сучасний стан поліграфії та видавничої справи і запровадження прогресивних інформаційних технологій. В дисциплінах комп'ютерного циклу використовуються видавничі системи для опрацювання і верстання тексту та ілюстрацій, пакети керування базами даних, засоби організації електронних мультимедійних видань.

Одні з основних дисциплін навчального плану передбачають комп'ютеризацію та моделювання процесів комп'ютерного опрацювання текстової та графічної інформацій для випуску електронних мультимедійних видань, в тому числі побудову, дослідження та удосконалення моделей інформаційної технології проектування електронних мультимедійних видань під певні типи призначень. Це знайшло відображення в робочих навчальних програмах і методичних посібниках для лекційних та практичних курсах студентів освітнього ступеня «бакалавр» і «магістр» спеціальності «Видавництво

та поліграфія» освітньо-професійної програми «Технологія електронних мультимедійних видань», а саме такі дисципліни як «Організація веборієнтованих баз даних», «Електронний документообіг», «Технологія електронних мультимедійних видань», «Управління мультимедійним проєктом» на кафедрі інформаційних мультимедійних технологій Української академії друкарства.

Під час вивчення перерахованих вище дисциплін використовуються алгоритми, моделі та методи, побудова, дослідження і реалізація яких здійснені у дисертаційній роботі Тернового І. М. «Інформаційна технологія проектування автоматизованої системи забезпечення діяльності студентського містечка». Основні серед них: ієрархічно структуровані, упорядковані за ваговими коефіцієнтами та оптимізовані графічні моделі вагових значень та пріоритетного впливу факторів на якість створення та запровадження автоматизованої системи забезпечення діяльності роботи студмістечка, які є основою створення інформаційної системи; системи поселення студентів; ведення ділової переписки; підтримки інформаційного забезпечення; схоронності персональних даних користувачів системи.

Матеріали дисертації використовуються під час визначення тем та змісту кваліфікаційних робіт студентів технологічних спеціальностей.

Завідувач кафедри інформаційних
мультимедійних технологій
д.т.н., професор  В. І. Сабат

Додаток Д.

ДОВІДКА ПРО ВПРОВАДЖЕННЯ РЕЗУЛЬТАТІВ ДИСЕРТАЦІЙНОГО ДОСЛІДЖЕННЯ

ЦЕНТРАЛЬНА СПІЛКА СПОЖИВЧИХ ТОВАРИСТВ УКРАЇНИ
ЛЬВІВСЬКИЙ ТОРГОВЕЛЬНО-ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
вул. Туган-Барановського, 10,
Львів, Україна, 79005
тел.: +38 (032) 275 65 50; 295 81 02
факс: +38 (032) 295 81 02
e-mail: lute@lute.lviv.ua

THE ASSOCIATION OF CONSUMER SOCIETIES OF UKRAINE
LVIV UNIVERSITY OF TRADE AND ECONOMICS
10, Tuhan-Baranovskoho street,
Lviv, Ukraine, 79005
tel.: +38 (032) 275 65 50; 295 81 02
fax: +38 (032) 295 81 02
e-mail: lute@lute.lviv.ua

04.03.2025р. № 81/01-1.08

на № _____

ДОВІДКА
про впровадження результатів дисертаційного дослідження здобувача ступеня доктора філософії за спеціальністю 126 "Інформаційні системи та технології" Тернового Івана Михайловича за темою "Інформаційна технологія проектування автоматизованої системи забезпечення діяльності студентського містечка"

Дослідження аспіранта Тернового І. М. відповідає науково-дослідній темі Львівського торговельно-економічного університету "Моделювання та застосування хаотичних динамічних систем для оптимізації алгоритмів штучного інтелекту, машинного навчання та веб-технологій" (державний реєстраційний номер 0124U004447).

У рамках виконаної дисертаційної роботи розроблено інформаційні технології проектування автоматизованих систем, що дають змогу забезпечувати ефективне управління діяльністю студмістечка. Заслуговує на увагу "алгоритм оптимізації розподілу ресурсів та поселення студентів".

Дослідженнями забезпечено автоматизацію ключових процесів управління студмістечком; підвищення якості обслуговування студентів; підвищення точності та ефективності опрацювання даних; поліпшення комунікації між адміністрацією та студентами та інше.

На підставі виконаної роботи та наукових результатів можна зробити висновок, що система управління студмістечком успішно впроваджена та функціонує відповідно до вимог. Рекомендується продовжити моніторинг результативності системи в майбутньому для подальшого вдосконалення та адаптації до нових вимог. Результати даної роботи мають практичну цінність і можуть бути рекомендовані для подальшого використання в освітніх установах.

Проректор з наукової роботи **Богдан СЕМАК**