УДК 004.42

О. В. ВОЙЦЕХОВСЬКА, Н. П. БАБЮК, В. В. МАЛІЦЬКИЙ

ІНФОРМАЦІЙНА СИСТЕМА КЕРУВАННЯ ОБЛІКОМ СТУДЕНТІВ У ГУРТОЖИТКАХ

Вінницький національний технічний університет,
21021, Хмельницьке шосе, 95, м. Вінниця, Україна, e-mail: vojcexovska.o.v@vntu.edu.ua

Анотація. Розроблено інформаційну систему керування обліком студентів у гуртожитках, яка призначена для автоматизації процесів управління студентськими гуртожитками, зокрема обліку, рейтингування та контролю проживання студентів. Проведений порівняльний аналіз існуючих систем показав відсутність в них можливостей ведення та автоматизації процесів рейтингування і фіксації порушень. Визначено функціональні та нефункціональні вимоги до розробленої інформаційної системи. На основі визначених вимог систему спроектовано за клієнт-серверною архітектурою, розроблено структуру клієнтської та серверної частин. Взаємодія між рівнями клієнт-серверної архітектури інформаційної системи побудована через API. В системі керування обліком студентів реалізовано функції авторизації користувачів, перегляду та редагування даних студентів, додавання нових студентів до бази даних та автоматизованого створення звітності та рейтингування. Для забезпечення контролю доступу до системи користувачам присвоюється певна роль із відповідними правами. Розроблено вебдодаток з використанням сучасних технологій, зокрема для клієнтської частини використано React, я для серверної – Node.js. Реалізована інтеграція з Google Drive API для завантаження та перегляду файлів з фіксацією порушень. Сувора автентифікація користувачів на рівні API відбувається засобами платформи Firebase Authentication. Виконане оцінювання витрат ресурсів показало, що на фазі розробки та розгортання системи необхідна підвищена увага до системно-інженерної діяльності розробників.

Ключові слова: інформаційна система, облік студентів, рейтингування, гуртожиток, React, MySQL, Firebase, Node.js.

Abstract. An information system for student accounts management in dormitories has been developed to automate the processes of student dormitory administration, including record-keeping, ranking, and monitoring of student residency. A comparative analysis of existing systems revealed that they lack capabilities for managing and automating ranking processes and violation tracking. The functional and non-functional requirements for the developed information system were identified. Based on these requirements, the system was designed using a client-server architecture, and the structures of both the client and server components were created. Interaction between the layers of the client-server architecture is implemented through an API.

The student accounts management system includes user authorization, viewing and editing student data, adding new students to the database, and automated report generation and ranking. To ensure access control, each user is assigned a specific role with corresponding permissions. A web application was developed using modern technologies: React for the client side and Node.js for the server side. Integration with the Google Drive API was implemented to enable file uploading and viewing for violation tracking. Strict API-level user authentication is handled through Firebase Authentication. Resource cost estimation showed that during the development and deployment phases, increased attention to the systems-engineering activities of the developers is required.

Keywords: information system, student registration, rating, dormitory, React, MySQL, Firebase, Node.js.

DOI: 10.31649/1681-7893-2025-50-2-30-39

ВСТУП

Невід’ємною частиною освітнього процесу у сучасних вищих навчальних закладах є проживання студентів у гуртожитку, а управління проживанням в них студентів є однією з ключових сфер, яка потребує систематизації та оптимізації процесів.

Більшість гуртожитків досі використовують застарілі методи обліку мешканців, ведення документації та контролю дисциплінарних порушень, зокрема паперові журнали або локальні електронні таблиці для ведення обліку мешканців, проведення рейтингування студентів та фіксації порушень правил користування гуртожитком. Це призводить до низької швидкості оновлення інформації, затримок в її обробці та зниження прозорості. Саме тому розробка інформаційної онлайн-системи, яка б дозволяла централізовано керувати всіма аспектами проживання студентів, сприяла автоматизації цих процесів, підвищенню прозорості й ефективності управління гуртожитками, а також спонукала до дотримання студентами внутрішніх правил проживання, є актуальною.

Мета роботи полягає в розширенні функціональних можливостей наявних систем обліку студентів для гуртожитків шляхом створення інформаційної онлайн системи для обліку, рейтингування та контролю проживання, яка забезпечить можливість проведення поселення та виселення, формування рейтингу, контролю дисциплінарних заходів, ведення звітності за окремим студентом та збереження супровідних документів.

ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

На сьогодні широко використовуються такі способи управління освітнім процесом, як автоматизовані системи управління навчальними закладами або ведення документообігу у електронних таблицях. При цьому спеціалізованих рішень, які повністю охоплюють управління гуртожитками, облік поселення, рейтингування студентів, контроль порушень та забезпечують інтуїтивну візуалізацію розміщення, практично немає.

Автоматизовані системи управління (АСУ) навчальним закладом зазвичай включають модулі для обліку студентів, реєстрації, відвідуваності, виставлення оцінок та формування звітності.

Відома система «Студмістечко», яка є частиною АСУ «ВНЗ», що автоматизує основні процеси життєдіяльності студентського містечка: від розподілу місць у гуртожитках до контролю оплати та управління пропускною системою [1].

Основними функціями системи «Студмістечко» є поселення студентів на основі даних із іншої частини АСУ «ВНЗ» – АС «Деканат», автоматизація діяльності дирекції студентського містечка навчального закладу, спрощення та вдосконалення документообігу, підвищення оперативності, узгодженості та достовірності даних. Вона забезпечує зручне формування і ведення документації, включаючи договори, заяви та звіти. Проте вона не підтримує візуалізацію розміщення студентів у гуртожитку, має недостатню гнучкість для створення рейтингів чи фіксації порушень.

Ще однією відомою системою керування освітнім процесом є JetIQ, яка являє собою єдину інтегровану клієнт-серверну навчальну систему, де реалізовано функції дистанційного та змішаного навчання і управління закладом вищої освіти. Вона впроваджена і активно використовується у Вінницькому національному технічному університеті. JetIQ призначена для об’єднання всіх ключових процесів в межах університету: від для управління освітнім процесом, моніторингу результатів навчання до фіксації оплати за навчання та проживання у гуртожитках [2].

Один із модулів, пов’язаний з обліком проживання студентів у гуртожитках, – модуль «Гуртожитки та оплата», в якому наявна можливість подання заяв на поселення від студентів та подальше підтвердження заяв адміністрацією університету. Також є функції аналізу стану та візуального представлення заповненості кімнат. При наведенні на потрібну кімнату, відображається список студентів, які там проживають. Наявна функція перевірки оплати за гуртожиток, яка показує сальдо на поточний момент.

Крім того, багато освітніх закладів користуються простими електронними таблицями, такими як Google Sheets чи Microsoft Excel, для ведення списків студентів, реєстрації поселення та фіксації даних студентів. Це найпростіший і найшвидший спосіб організації обліку. Перевагами такого підходу є безкоштовність, простоту використання, гнучкість у створенні формул та розрахунків, а також можливість спільного редагування даних при використанні Google Sheets. Проте, цей метод має відсутність ролевого контролю доступу, непридатність для реалізації задач автоматизованого обліку актів порушень, рейтингування студентів або створення інтерактивних мап гуртожитків.

Отже, проведений аналіз існуючих сервісів показав обмежені функціональні можливості для вирішення завдань, пов’язаних із житловим обліком. Жоден з існуючих аналогів не надає повного набору функціональних можливостей для візуалізації розміщення студентів у гуртожитках, рейтингування мешканців, формування актів порушень та взаємодії між відповідальними особами та деканатом.

Тому поставлена задача розробити інформаційну онлайн систему керування обліком студентів у гуртожитках, яка розширює функціональні можливості існуючих систем, шляхом надання деканату закладу вищої освіти та адміністрації гуртожитків сервісу для рейтингування студентів, візуалізації їх

розміщення у житлових приміщеннях, формування актів порушень та організації взаємодії між відповідальними особами.

ПРОЕКТУВАННЯ АРХІТЕКТУРИ СИСТЕМИ КЕРУВАННЯ ОБЛІКОМ СТУДЕНТІВ У ГУРТОЖИТКАХ

Розроблена інформаційна система повинна задовольняти такі функціональні вимоги: забезпечення авторизації користувачів із отриманням JWT-токена; можливість введення, перегляду та редагування інформації про студентів; відображення інтерактивної мапи гуртожитку; можливість архівування даних; ведення обігу інформації про поселення, перепоселення та виселення студентів; створення звітності про наявність та кількість порушень; складання рейтингів студентів.

Для цієї інформаційної системи можна виділити такі нефункціональні вимоги, як використання сучасного стеку технологій, можливість масштабування, захист автентифікованого трафіку, забезпечення завадостійкості

Інформаційна система, побудована за клієнт-серверною архітектурою, має чітке розділення на три основні рівні: клієнтський інтерфейс, серверну частину та базу даних (рис. 1).

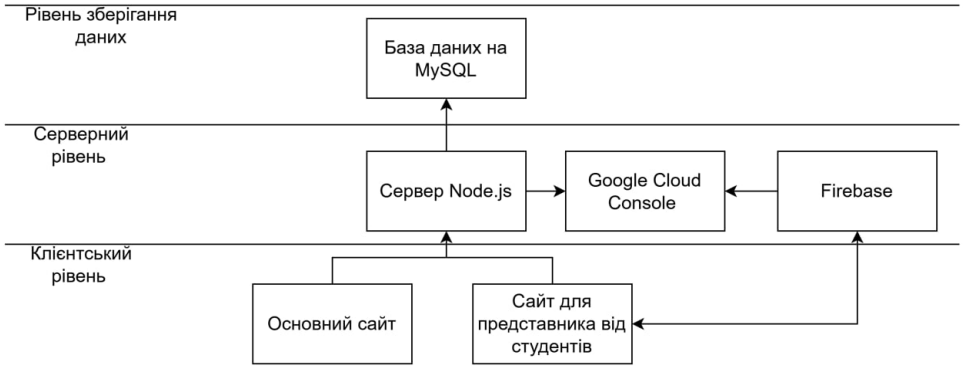

Рисунок 1 – Структурна схема архітектурних рівнів системи

Для реалізації клієнтської частини (рис. 2) використано відкриту бібліотеку React.js, призначену для розробки користувацьких інтерфейсів, яка зменшує час оновлення окремих частин системи, що є корисним при створенні багатосторінкових вебзастосунків [3].

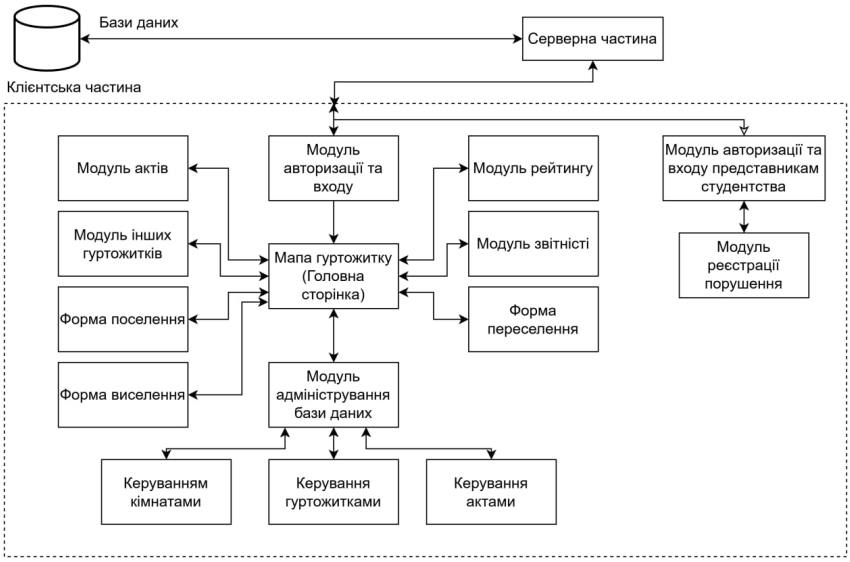

Рисунок 2 – Структурна схема клієнтської частини

Реалізовано інтерактивний інтерфейс користувача, який працює у браузері та забезпечує зручну взаємодію системи з користувачем та відправляє запити до сервера через REST API. Клієнт надсилає HTTP-запити до API-сервера (GET, POST, PUT, DELETE) і отримує відповіді у форматі JSON.

Для реалізації серверної частини (рис. 3) обрано технологію Node.js – платформу з відкритим кодом, призначену для створення високопродуктивних мережних застосунків мовою JavaScript [4].

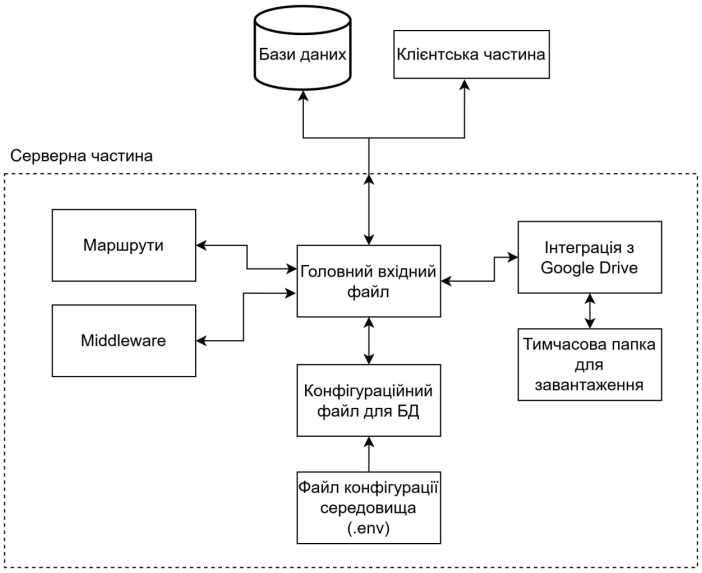

Рисунок 3 – Структурна схема серверної частини

Серверна частина містить такі компоненти як Маршрути, Middleware, головний вхідний файл, файл конфігурації для бази даних, файл конфігурації середовища .env, модуль інтеграції з Google Drive та тимчасова папка для завантаження. Крім обробки запитів і взаємодії з MySQL, серверна частина включає підтримку зовнішніх сервісів, зокрема Google Cloud Console та Firebase Authentication, які використовуються для автентифікації представників студентського самоврядування.

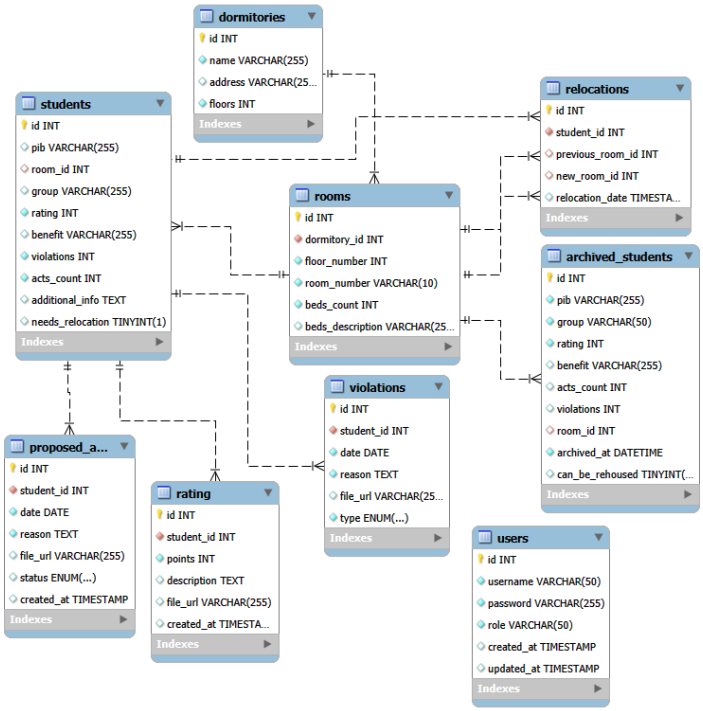

Рисунок 4 – ER-діаграма бази даних

Для забезпечення централізованого зберігання та обробки інформації про студентів, гуртожитки, кімнати, акти порушень і рейтинги у системі використовується реляційна база даних, реалізована за

допомогою СУБД MySQL. База даних містить такі таблиці як user, students, archived_students, dormitories, rooms, violations, ratings, relocations, proposed_acts. ER_модель БД подано на рис. 4.

Контроль доступу до системи побудований за принципом розподілу ролей, де користувачам присвоюється певна роль із відповідними правами, що визначають рівень його доступу до ресурсів. Такий механізм забезпечує ефективне управління доступом, дозволяє гнучко налаштовувати відповідні дозволи, а також спрощує адміністрування і підвищує безпеку розробленої інформаційної системи. Виокремлено три основні ролі користувачів: «Адміністратор системи», «Адміністратор бази даних» та «Представник студентів», кожна з яких має відповідні права і доступ до визначених ресурсів. На рис. 5 показано ролі користувачів і їх можлива взаємодія із системою.

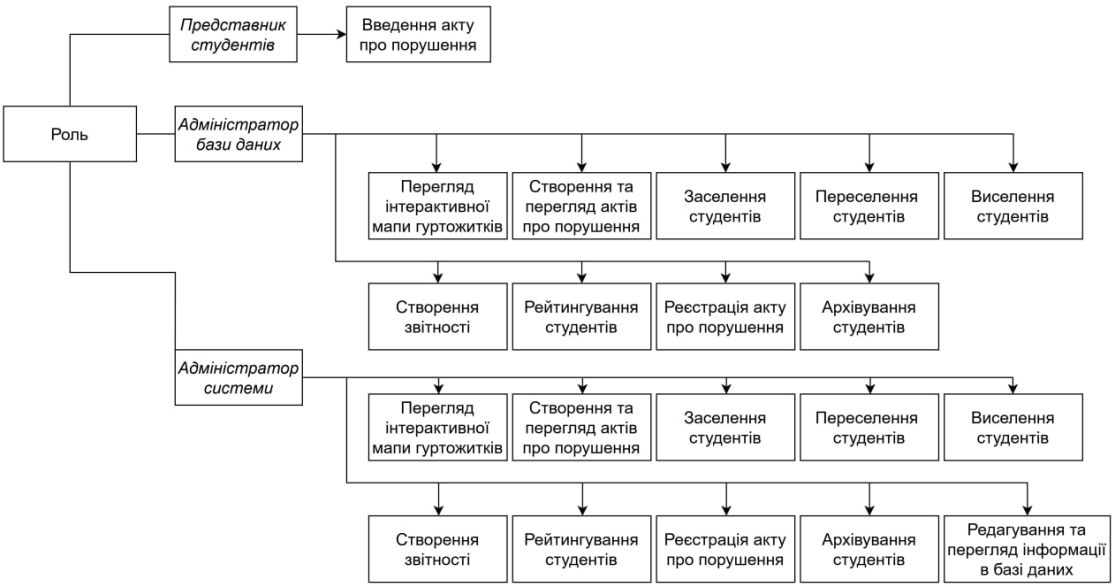

Рисунок 5 – Ролі користувачів в системі обліку та доступні їм функції

Взаємодія між рівнями клієнт-серверної архітектури розробленої системи організована через API, а передача даних здійснюється у форматі JSON. Схему взаємодії користувача з компонентами системи наведено на рис. 6.

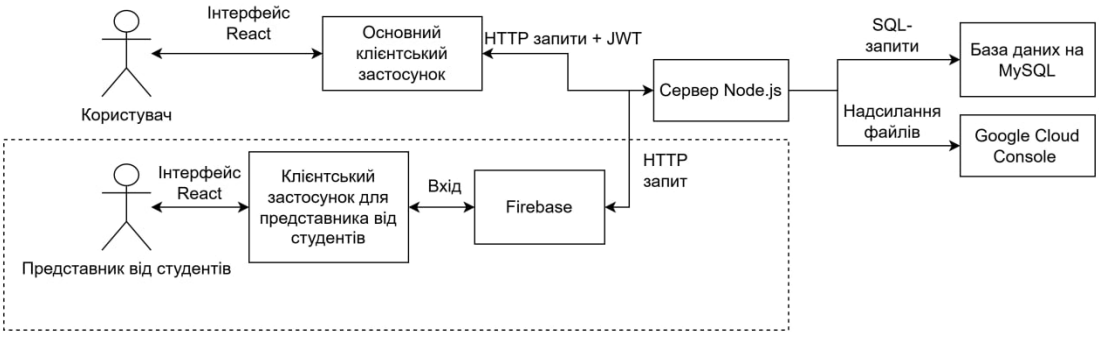

Рисунок 6 – Схема взаємодії користувача з компонентами системи

Для організації маршрутизації, обробки HTTP-запитів і створення RESTful API використано Express.js, що забезпечило модульну структуру та спрощену інтеграцію з базою даних, системами авторизації та сторонніми сервісами.

Для реалізації безпечної автентифікації користувачів використано платформу розробки Firebase від компанії Google, зокрема модуль Firebase Authentication, який підтримує різні методи входу, наприклад через email і пароль, акаунт Google та одноразові коди, що забезпечує гнучкість і надійність у керуванні доступом [5]. Модуль інтегрується із клієнтською частиною та Google акаунтами. Користувачі проходять автентифікацію через Google (OAuth 2.0), в результаті чого отримують безпечний токен доступу, який надсилається до серверу, де проходить валідацію через Firebase SDK.

ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ КЕРУВАННЯ ОБЛІКОМ СТУДЕНТІВ У ГУРТОЖИТКАХ

За спроектованою архітектурою інформаційної системи розроблено вебдодаток, в якому реалізовано облік проживання студентів в гуртожитках, контролювання порушень правил проживання, а також автоматизоване рейтингування студентів, передбачає можливість масштабування, зокрема додавання різних модулів [6]. Спроектована онлайн система дозволяє реалізувати реєстрацію та редагування особистих даних поселених студентів, облік їх проживання в кімнатах гуртожитку, виконувати перепоселення і виселення студентів, вводити рейтингові бали, створювати звіти про наявність та кількість порушень та про рейтингування. Доступ та керування даними в системі здійснюється за допомогою розробленої адміністративної панелі.

При відкритті вебдодатку користувач вводить логін та пароль для доступу до системи. Після успішного входу відображається головне меню з доступними сторінками.

Якщо користувач має роль адміністратора системи, він отримує доступ до головної сторінки, вигляд якої подано на рис. 7. На цій сторінці подано схему кімнат гуртожитку з можливістю вибору поверху та кімнати. Також користувач має можливість переходити на інші сторінки системи.

Рисунок 7 – Вигляд головної сторінки системи керування обліком студентів у гуртожитках

Сторінка «Інші гуртожитки» надає можливість перегляду даних студентів, які проживають в інших гуртожитках.

Сторінка «Акти» використовується для реєстрації і перегляду порушень правил проживання у гуртожитку.

За допомогою сторінок «Поселення», «Перепоселення» та «Виселення» можна керувати розміщенням студентів по кімнатах гуртожитків.

Сторінки «Звітність» та «Звітність (рейтинг)» забезпечують автоматизоване формування звітів щодо порушень та рейтингування мешканців гуртожитку.

Сторінка «Інструкція» містить довідкову інформацію про систему та її використання.

На сторінці «Рейтинг» відображається і вводиться інформація щодо рейтингових оцінок студентів. Користувачі з роллю адміністратора бази даних мають доступ до всіх функцій адміністратора системи та додаткового меню «Адміністрування», яке включає сторінки адміністрування бази даних.

Через сторінку «Керування актами» адміністратор може опрацьовувати порушення в базі даних, а сторінка «Керування кімнатами» дозволяє додавати кімнати і редагувати їхній опис у вибраному гуртожитку.

Сторінка «Керування гуртожитками» надає можливість додавати гуртожитки, створювати схематичні плани поверхів і розташування блоків.

Для користувачів, які мають роль «Представник студентів», надається доступ лише до сторінки «Запропонувати акт», де вони можуть подавати інформацію про порушення правил для розгляду адміністратором системи.

Також для них створено окремий програмний модуль реєстрації порушень правил проживання студентів у гуртожитку [7], який дозволяє проходити автентифікацію з допомогою сервісів Google (Firebase Authentication), вводити та, за потреби, реєструвати акт порушення правил проживання студентів у гуртожитку. Користувач модуля повинен мати обліковий запис Google, оскільки без входу через сервіси Google доступ до програмного модуля неможливий. Дані про вхід зберігаються в сервісі Firebase, який забезпечує безпечне зберігання та управління обліковими записами користувачів. Після успішної автентифікації користувач може вводити особисті дані студента, дату, опис порушення та прикріплювати файли. Усі внесені дані передаються на сервер, де обробляються і зберігаються в базі даних. У випадку успішного внесення інформації система надсилає відповідне повідомлення.

Для забезпечення безпеки даних, сумісності та стабільності роботи розробленої системи проведено аналіз вразливостей з допомогою інструменту командного рядка npm audit, що надається npm (Node Package Manager), який допомагає виявляти та виправляти вразливості безпеки в npm-пакетах, що використовуються в проекті Node.js [8]. Він зв'язується з базою даних команди безпеки npm, щоб отримати найновішу інформацію про вразливості для пакетів, перелічених у файлі package.json розробленого проєкту. Відбувається порівняння отриманої інформації з версіями пакетів, встановлених у проєкті, для визначення наявності будь-яких вразливостей. Після завершення аналізу, npm audit генерує звіт із детальним описом знайдених вразливостей, їх рівнів серйозності та рекомендацій щодо виправлення.

В результаті проведеного аналізу системи виявлено 18 вразливостей, з яких 1 критична та 8 високих. Більшість з них пов’язані із використаним технологічним стеком, тому проведено оновлення пакетів та сторонніх бібліотек, що були використані в проєкті. Для виправлення вразливостей npm додав 1232 пакети, змінив 3 та виявив 1404 пакетів для аудиту, в результаті чого отримано 9 вразливостей (6 високих, 3 середніх).

Для безпечного усунення всіх вразливостей виконано команду npm audit fix, результати виконання якої наведено на рис. 8.

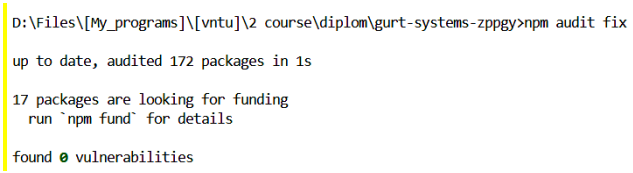

Рисунок 8 – Результат виконання команди npm audit fix

Після виправлень усі критичні вразливості усунуто. Подібна ситуація є типовою для каталогу node_modules і означає, що не всі вразливості фактично небезпечні. Часто це транзитивні залежності, які не потрапляють у production-збірку.

Також для підвищення рівня захисту даних в розробленій системі було введено сувору автентифікацію на рівні API, що дає можливість скоротити час, на який зловмисник може отримати доступ до системи. Кожен запит перевіряється на наявність і дійсність JWT-токена, створеного за допомогою стійкого криптографічного секретного ключа. Крім того, для токенів введено обмеження часу їх дії і додано механізм refresh-токенів для безпечного оновлення користувацьких сесій. В результаті користувач не зможе ввійти до системи чи отримати дані із неї, маючи застарілий токен доступу.

ОЦІНЮВАННЯ РЕСУРСІВ ДЛЯ РОЗРОБКИ СИСТЕМИ

Важливим фактором в процесі розробки інформаційних систем є обчислення зусиль та вартості з урахуванням функціонального розміру системи та коригування отриманих значень на основі низки факторів навколишнього середовища, пов'язаних із системною інженерією. Для цього в роботі використано модель витрат ресурсів на конструктивну системну інженерію (COSYSMO – The Constructive Systems Engineering Cost Model) [9].

COSYSMO виконує оцінювання системи за 13-ма факторами витрачених ресурсів, що поділені на 5 основних груп: розуміння системи, її складність, операційні фактори розробки, вплив персоналу та середовища проєкту на кінцевий результат. Фактори витрачених ресурсів є мультиплікативними факторами, які визначають зусилля, необхідні для завершення етапів життєвого циклу в інженерії розробленої інформаційної системи [9].

Наприклад, якщо команда є продуктивною та вчасно завершує організаційні фази проекту, можна встановити фактор витрат, пов’язаний з персоналом/командою (PCAP), як “Дуже високий”. Цей рейтинг

відповідає множнику зусиль 0,66, що означає, що проект вимагав лише 66% зусиль системної інженерії, на противагу до типового проекту.

Модель COSYSMO оцінює необхідні зусилля в людино-годинах – PH, головним чином на основі оцінки розміру розробленої системи, що вимірюється з урахуванням еквівалентних факторів витрачених ресурсів:

Трудовитрати PH, оцінені у людино-годинах, визначаються за формулою:

$$PH = A \cdot AdjSize^{E} \cdot \prod EM_{j},$$

де: A – калібрувальний параметр;

$AdjSize^{E}$ – скоригований розмір проєкту;

E – експонента (враховує масштабні фактори);

EM_{j} – фактори витрат ресурсів.

Коефіцієнт коригування зусиль у рівнянні зусиль – це добуток множників зусиль, що відповідають кожному з факторів витрат для проекту.

Формула скоригованого розміру AdjSize має вигляд:

$$AdjSize = \sum_{i=1}^{N} e\,\mathrm{Re}\,q_{i}\left(Type(SD), Difficulty(SD)\right) \times PartialDevFactor(AL_{Start}(SD)\,AL_{End}\,RType(SD)),$$

де: $e\,\mathrm{Re}\,q_{i}$ – еквіваленти вимог до системи;

$Type(SD)$ – тип елементу (вимоги, сценарій, інтерфейс, алгоритм);

Difficulty(SD) – складність елементу згідно еталонної шкали Low/Nominal/High/Very High;

$PartialDevFactor_{i}$ – коефіцієнт часткової розробки (0–100%);

AL_Start і AL_End – рівні готовності (Assessment Level), які відображають, на якій стадії життєвого циклу вимога була отримана і на якій стадії виконана.

Отже, кожен фактор витрат (вимоги, інтерфейси, алгоритми, сценарії) перетворюється в кількість еквівалентних вимог, що коригується згідно типу, складності, обсягу нової або повторної розробки. Потім усі ці значення складаються, що дає Adjusted Size – фактичний об'єм роботи системного інженера.

Для розрахунку масштабованих факторів розробки експонента розраховується за формулою:

$$E = E_{Base} + SF_{Risk} + SF_{\mathrm{Pr}\,ocess} + SF_{\mathrm{Re}\,qVolatility},$$

де: E_{Base} — базове значення експоненти;

SF — масштабовані фактори: ризики/можливості, здатність виконання процесів, готовність до зміни вимог.

Відповідно до оцінювання витрат ресурсів для розробленої системи, було здійснено зважування показників та факторів, що впливають на розробку програмних систем (рис. 9).

У розрахунку потрібно вказувати точні значення *eReq*, які калібрують під кожен окремий проєкт, дані для коефіцієнту часткової розробки для усіх комбінацій готовності вимог, а також відображати частково модифіковані елементи факторів витрат ресурсів.

Розрахунки COSYSMO для розробленої системи базувалися на оцінюванні розміру проекту, виміряного за чотирма факторами:

1. Вимірювання функціональних та нефункціональних вимог до розробленої системи.
2. Кількість інтерфейсів розробленої системи.
3. Складність і реалізованість алгоритмів.
4. Кількість виконуваних сценаріїв розробленої системи.

Загальний розмір інформаційної системи визначається на основі показників, які надавались для кожного фактору оцінювання. Для розробленої системи керування обліком студентів розглядалась 21 функціональна та нефункціональна вимога, з яких 10 рівня Low, 8 рівня Nominal, 3 – High, 9 системних інтерфейсів, 3 критичні алгоритми, 17 операційних сценаріїв. Загальний розмір ресурсів виражається через еквівалентні вимоги та є основним вихідним показником для визначення зусиль, витрачених для створення системи. В результаті обчислень для розробленої системи було отримане значення 324,7 еквівалентних номінальних вимог, на основі якого було розраховано трудомісткість, час та вартість через масштабні коефіцієнти та продуктивність розробки системи (рисунок 9).

Total Size = 342.7 Equivalent Nominal Requirements

Phase / Activity	Conceptualize	Develop	Operational Test and Evaluation	Transition to Operation
Acquisition and Supply	1.0	84.4	21.5	13.2
Technical Management	88.4	152.7	100.4	60.3
System Design	241.1	283.6	120.5	63.8
Product Realization	46.1	106.4	113.4	88.6
Product Evaluation	131.9	197.8	293.1	109.9

Рисунок 9 – Результати оцінювання витрат на розроблену систему керування обліком

Кожне значення комірок таблиці на рисунку 9 – це кількість еквівалентних номінальних вимог, які потрібно опрацювати на певній фазі проєкту для певного виду системно-інженерної діяльності. На різних фазах проєкту було визначено, що під час концептуалізації здійснюється аналіз і оцінювання задач з невеликою кількістю документування, на час фази розробки припадає пік навантаження, на фазі тестування та випробування виконується значний обсяг валідації, а під час фази розгортання системи на сервері виконується менше завдань, які мають високу вагу затрат ресурсів на їх виконання.

ВИСНОВКИ

Впровадження інформаційної системи керуванням обліком студентів у гуртожитках є актуальним і необхідним кроком для сучасних закладів вищої освіти.

Реалізована інформаційна система поєднує функції реєстрації студентів, поселення, переселення, фіксації порушень і досягнень, автоматизованого формування рейтингу та звітності з інтерактивною мапою гуртожитку та централізованим зберіганням документів в Google Drive.

Розроблена система підвищує якість обліку студентів, зменшує використання паперової документації, а також автоматизує формування звітності, що сприяє оперативності та точності управлінських процесів. Завдяки впровадженню чіткої ролі користувачів із налаштованими правами доступу й розгалуженій функціональності, включаючи інтерактивні модулі для керування гуртожитками, поселенням, порушеннями та рейтингуванням, підвищено рівень захищеності даних, покращено UI/UX системи.

Виконано оцінювання витрат ресурсів для розробки системи на різних фазах виконання проєкту. Визначено, що на фазі розробки та розгортання системи необхідна підвищена увага до системно-інженерної діяльності розробників.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. АС «Студмістечко». Available at: https://vuz.osvita.net/as-studmistechko/ (accessed 25.11.2025).
2. Електронна система управління закладом вищої освіти (ЗВО) "JetIQ". Available at: https://wiki.vntu.edu.ua/uk/Електронна_система_управління_закладом_вищої_освіти_(ЗВО)_"JetIQ" (accessed 25.11.2025).
3. React. The library for web and native user interfaces. Available at: https://react.dev/ (accessed 28.11.2025).
4. Node.js – Run JavaScript Everywhere. Available at: https://nodejs.org/en (accessed 28.11.2025).
5. Firebase Authentication – Google. Available at: https://firebase.google.com/docs/auth (accessed 01.12.2025).
6. Кирилащук С.А., Войцеховська О.В., Бабюк Н.П. Комп'ютерна програма «Онлайн система обліку, рейтингування та контролю проживання студентів у гуртожитку». Свідоцтво про реєстрацію авторського права №135785, дата реєстрації 06.05.2025, опубл. 30.06.2025, бюл. №90.
7. Бабюк Н.П., Войцеховська О.В., Маліцький В.В. Комп'ютерна програма «Програмний модуль реєстрації порушень правил проживання студентів у гуртожитку». Свідоцтво про реєстрацію авторського права №135686, дата реєстрації 05.05.2025, опубл. 30.06.2025, бюл. №90.

8. What is npm audit? Available at: https://www.geeksforgeeks.org/node-js/what-is-npm-audit/ (accessed 03.12.2025).
9. Alstad J.P. Development of COSYSMO 3.0: An Extended, Unified Cost Estimating Model for Systems Engineering. Procedia Computer Science, 153 (2019) 55–62. https://doi.org/10.1016/j.procs.2019.05.055
10. Constructive Systems Engineering Cost Model (COSYSMO). Available at: https://softwarecost.org/tools/COSYSMO/ (accessed 03.12.2025).

REFERENCES

1. AS "Studmistechko". Available at: https://vuz.osvita.net/as-studmistechko/ (accessed 25.11.2025).
2. Electronic management system of a higher education institution (HEI) "JetIQ". Available at: https://wiki.vntu.edu.ua/uk/Електронна_система_управління_закладом_вищої_освіти_(ЗВО)_"JetIQ" (accessed 25.11.2025).
3. React. The library for web and native user interfaces. Available at: https://react.dev/ (accessed 28.11.2025).
4. Node.js – Run JavaScript Everywhere. Available at: https://nodejs.org/en (accessed 28.11.2025).
5. Firebase Authentication – Google. Available at: https://firebase.google.com/docs/auth (accessed 01.12.2025).
6. Kyrylashchuk S.A., Voitsekhovska O.V., Babiuk N.P. Computer program “Online system for accounting, rating and control of student accommodation in a dormitory”. Certificate of registration of copyright No. 135785, date of registration 06.05.2025, publ. 30.06.2025, bull. No. 90.
7. Babiuk N.P., Voitsekhovska O.V., Malitsky V.V. Computer program “Software module for registering violations of the rules of student accommodation in a dormitory”. Certificate of registration of copyright No. 135686, date of registration 05.05.2025, publ. 30.06.2025, bull. No. 90.Кирилащук С.А.,
8. What is npm audit? Available at: https://www.geeksforgeeks.org/node-js/what-is-npm-audit/ (accessed 03.12.2025).
9. Alstad J.P. Development of COSYSMO 3.0: An Extended, Unified Cost Estimating Model for Systems Engineering. Procedia Computer Science, 153 (2019) 55–62. https://doi.org/10.1016/j.procs.2019.05.055
10. Constructive Systems Engineering Cost Model (COSYSMO). Available at: https://softwarecost.org/tools/COSYSMO/ (accessed 03.12.2025).

Надійшла до редакції: 15.11.2025

ВОЙЦЕХОВСЬКА ОЛЕНА ВАЛЕРІЇВНА – кандидат техн. наук, доцент кафедри обчислювальної техніки, Вінницький національний технічний університет***, e-mail:*** ***vojcexovska.o.v@vntu.edu.ua***

БАБЮК НАТАЛЯ ПЕТРІВНА – кандидат техн. наук, доцент кафедри програмного забезпечення, Вінницький національний технічний університет, ***e-mail:*** ***babiuk@vntu.edu.ua***

МАЛІЦЬКИЙ ВЛАДИСЛАВ ВІТАЛІЙОВИЧ – студент групи 1КІ-25м факультету інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет,
e-mail: ***vindener12@gmail.com.***

O. V. VOITSEKHOVSKA, N. P. BABIUK, V. V. MALITSKYI
INFORMATION SYSTEM FOR STUDENT ACCOUNTS MANAGEMENT IN DORMITORIES
Vinnytsia National Technical University