НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Навчально-науковий інститут атомної та теплової енергетики
Кафедра цифрових технологій в енергетиці

"На правах рукопису"
УДК_____

«До захисту допущено»
Завідувач кафедри
_____ Наталія АУШЕВА
"___"_____2022р.

Магістерська дисертація

зі спеціальності - 122 Комп'ютерні науки
за освітньо-професійною програмою магістерської підготовки - Комп'ютерний моніторинг та геометричне моделювання процесів і систем

на тему «Інтеграція ERP систем на платформі Microsoft Dynamics 365 Business Central за допомогою веб сервісів»

Виконав: студент 2 курсу, групи ТР-11мп
Соболєв Павло Олександрович _____
(прізвище, ім'я, по батькові) (підпис)

Науковий керівник доцент, к. т. н. Лабжинський В.А. _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Консультант _____ _____ _____
(назва розділу) (вчені ступінь та звання, прізвище, ініціали) (підпис)

Рецензент_____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ - 2022

Національний технічний університет України
“ Київський політехнічний інститут ім. Ігоря Сікорського”

Навчально-науковий інститут атомної та теплової енергетики
Кафедра цифрових технологій в енергетиці
Рівень вищої освіти другий, магістерський
За освітньою програмою "Комп’ютерний моніторинг та геометричне моделювання процесів і систем"
Спеціальності 122 Комп’ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Наталія АУШЕВА
(підпис)
«____» _____ 2022р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ ДИСЕРТАЦІЮ СТУДЕНТУ

Соболєв Павло Олександрович
(прізвище, ім’я, по батькові)

1. Тема дисертації «Інтеграція ERP систем на платформі Microsoft Dynamics 365 Business Central за допомогою веб сервісів»

Науковий керівник Лабжинський Володимир Анатолійович, к. т. н., доцент
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від “07” листопада 2022 року № 4067-с

2. Строк подання студентом дисертації 15 грудня 2022 року

3. Вихідні дані до роботи мова програмування JavaScript, бібліотека для реалізації користувальницького інтерфейсу React.js, середовище виконання JavaScript Node.js, середовище розробки JetBrains WebStorm.

4. Перелік питань, які потрібно розробити Дослідити переваги використання ERP систем. Обрати веб сервіси для інтеграції ERP системи. Проаналізувати процеси керування студмістечком. Побудувати ієрархічні діаграми користувачів системи. Реалізувати каркас архітектури системи керування студмістечком. Розробити модулі користувачів системи. Реалізувати моделі, презентери та представлення для системи керування студмістечком. Під’єднати розроблений клієнтський застосунок до ERP системи. Протестувати систему на тестових даних.

5. Орієнтований перелік ілюстративного матеріалу актуальність; мета, об’єкт та предмет дослідження; завдання роботи; умовні схеми роботи засобів для розробки системи; схеми архітектури системи; схема модулів системи; схема компонентів модуля; діаграми прецедентів; інструкція розгортання застосунку; сторінка авторизації; персональні кабінети користувачів системи; висновки.

6. Орієнтований перелік публікацій Акт впровадження в ТОВ “Ікспенд Україна”

7. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

8. Дата видачі завдання «1» вересня 2022р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання магістерської дисертації	Строки виконання етапів магістерської дисертації	Примітка
1	Затвердження теми роботи	до 05.09.2022р.	
2	Вивчення та аналіз задачі	до 10.09.2022р.	
3	Проєктування системи	до 25.09.2022р.	
4	Програмна реалізація клієнтського застосунку	до 10.10.2022р.	
5	Інтеграція ERP системи у клієнтський застосунок	до 15.10.2022р.	
6	Тестування системи	до 17.10.2022р.	
7	Написання дисертації	до 25.10.2022р.	
8	Перевірка дисертації науковим керівником	до 26.10.2022р.	
9	Захист програмного продукту	26.10.2022р.	
10	Внесення виправлень у дисертацію	до 27.11.2022р.	
11	Передзахист дисертації	22.11.2022р.	
12	Захист дисертації	21.12.2022р.	

Студент _____ Соболєв П.О.
(підпис) (прізвище та ініціали)

Науковий керівник _____ Лабжинський В.А.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Магістерська дисертація за темою “Інтеграція ERP систем на платформі Microsoft Dynamics 365 Business Central за допомогою веб сервісів” виконана студентом кафедри цифрових технологій в енергетиці ННІ ІАТЕ Соболєвим Павлом Олександровичем зі спеціальності 122 “Комп’ютерні науки” за освітньо-професійною програмою “Цифрові технології в енергетиці” та складається зі: вступу; 6 розділів (“Постановка задачі інтеграції ERP систем на платформі Microsoft Dynamics 365 Business Central за допомогою веб сервісів”, “Автоматизація процесу керування студмістечком”, “Засоби реалізації користувальницького інтерфейсу системи керування студмістечком”, “Опис програмної реалізації користувальницького інтерфейсу системи керування студмістечком”, “Методика роботи з користувальницьким інтерфейсом системи керування студмістечком”, “Розроблення стартап-проєкту”), висновків до кожного з цих розділів; загальних висновків; списку використаних джерел, який налічує 15 джерел та 1 додаток. Загальний обсяг роботи – 92 сторінки.

Актуальність теми. Існує проблема підвищення продуктивності, яка набирає більшого значення, коли фахівцям потрібно виконувати багато однакової роботи, яка циклічно повторюється. Такі системи потребують автоматизації. Для реалізації автоматизованих систем використовуються клієнт-серверні архітектури. Така архітектура вимагає кваліфікованого персоналу для її підтримки та адміністрування. Тому було вирішено використовувати ERP систему як серверний застосунок, яка не вимагає поглиблених знань для взаємодії з нею. Таким чином, після інтеграції ERP системи у клієнтський додаток, персонал з мінімальними навичками програмування зможе адмініструвати та підтримувати працездатність автоматизованої системи. У результаті вибору предметної області було вирішено розробити автоматизовану систему керування студмістечком. Така система забезпечить хороший рівень комунікації між студентом та адміністрацією.

Метою роботи є створення користувальницького інтерфейсу для ERP системи керування студмістечком та проведення інтеграції ERP системи у розроблений клієнтський застосунок за допомогою веб сервісів.

Завдання дослідження:

1. Проаналізувати процеси керування студмістечком та принципи роботи зі студентами
2. Здійснити порівняльний аналіз наявних застосунків для керування студмістечком
3. Проаналізувати та обрати технології, які найкраще підходять для реаліз системи
4. Реалізувати клієнтський застосунок системи керування студмістечком
5. Інтегрувати ERP систему керування студмістечком у розроблений клієнтський застосунок

Об’єкт дослідження. Процес керування студмістечком за допомогою ERP системи на платформі Microsoft Dynamics 365 Business Central.

Предмет дослідження. Веб сервіси для інтеграції ERP системи керування студмістечком на платформі Microsoft Dynamics 365 Business Central.

Практичне значення отриманих результатів. У розроблений клієнтський застосунок системи керування студмістечком буде інтегровано ERP систему керування студмістечком на платформі Mircosoft Dynamics 365 Business Central, що дозволить розв’язати проблему неефективного керування студмістечком.

Апробація результатів дисертації. Результати дисертації було впроваджено в ТОВ “Ікспенд Україна”.

Ключові слова. Автоматизація процесів, інтеграція, ERP система, клієнтський застосунок, користувальницький інтерфейс, веб сервіс, система керування студмістечком, платформа.

ABSTRACT

The master's thesis on the topic "Integration of ERP systems on the Microsoft Dynamics 365 Business Central platform using web services" was completed by Soboliev Pavlo Oleksandrovych, a student of the Department of Digital Technologies in Energy Sciences of the National Academy of Sciences of the Russian Academy of Sciences, from the specialty 122 "Computer Science" under the educational and professional program "Digital Technologies in energy" and consists of: introduction; 6 sections ("Setting the task of integrating ERP systems on the Microsoft Dynamics 365 Business Central platform using web services", "Automation of the campus management process", "Means of implementing the user interface of the campus management system", "Description of the software implementation of the user interface of the campus management system", "Methodology working with the user interface of the campus management system", "Development of a startup project"), conclusions to each of these sections; general conclusions; the list of used sources, which includes 15 sources and 1 application. The total volume of work is 92 pages.

Relevance of the topic. There is a problem of increasing productivity, which becomes more important when specialists need to perform a lot of the same work, which is cyclically repeated. Such systems need automation. Client-server architectures are used to implement automated systems. Such architecture requires qualified personnel to support and administer it. Therefore, it was decided to use the ERP system as a server application that does not require in-depth knowledge to interact with it. Thus, after integrating the ERP system into the client application, staff with minimal programming skills will be able to administer and maintain the automated system. As a result of the choice of the subject area, it was decided to develop an automated campus management system. Such a system will provide a good level of communication between the student and the administration.

The aim of the work is to create a user interface for the ERP campus management system and to integrate the ERP system into the developed client application using web services.

Objectives of the study:

1. To analyze the campus management processes and principles of working with students
2. To make a comparative analysis of existing applications for campus management
3. Analyze and select the technologies that are best suited for the implementatio the system
4. Implement the client application of the campus management system
5. Integrate the ERP campus management system into the developed client application

Object of research. The process of campus management using an ERP system on the Microsoft Dynamics 365 Business Central platform.

Subject of research. Web services for the integration of ERP campus management system on the Microsoft Dynamics 365 Business Central platform.

Practical significance of the results. The ERP campus management system on the Microsoft Dynamics 365 Business Central platform will be integrated into the developed client application of the campus management system, which will solve the problem of inefficient campus management.

Approbation of the results of the dissertation. The results of the dissertation were implemented in “Xpand Ukraine” LLC.

Keywords. Process automation, integration, ERP system, client application, user interface, web service, campus management system, platform.

ЗМІСТ

СПИСОК ТЕРМІНІВ, СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ

API (англ. Application Programming Interface) – прикладний програмний інтерфейс

CRM (англ. Customer Relationship Management) – інформаційна система управління відносинами з клієнтами

CRUD (англ. Create, Read, Update, Delete) – операції для сутностей: створити, переглянути, редагувати, видалити відповідно

CSS (англ. Cascading Style Sheets) – це спеціальна мова стилю сторінок, що використовується для опису їхнього зовнішнього вигляду

DOM (англ. Document Object Model) – специфікація прикладного програмного інтерфейсу для роботи зі структурованими документами

ERP (англ. Enterprise Resource Planning) – корпоративна інформаційна система, призначена для автоматизації обліку й керування

HTML (англ. HyperText Markup Language) – мова тегів, якою пишуться гіпертекстові документи для мережі Інтернет

IDE (англ. Integrated Development Environment) – інтегроване середовище розробки

NPM (англ. Node Package Manager) – менеджер пакетів

REST (англ. Representational State Transfer) – підхід до архітектури мережевих протоколів, які забезпечують доступ до інформаційних ресурсів

SASS (англ. Syntactically Awesome Stylesheets) – скриптова метамова, яка інтерпретується в каскадні таблиці стилів

SPA (англ. Single-Page Application) – веб застосунок чи веб сайт, який вміщується на одній сторінці з метою забезпечити користувачу досвід близький до користування настільною програмою

VCS (англ. Version Control System) – програмний інструмент для керування версіями одиниці інформації

ВСТУП

Автоматизація процесів – це використання цифрових технологій для автоматичного виконання завдань. Мета автоматизації полягає в тому, щоб усунути ручну роботу, яку можна виконувати цифровим способом, змусивши комп'ютер розв'язувати задачі за вас. Щоб створити найкращу систему автоматизації для своєї галузі, потрібно розуміти, що у сфері автоматизації існують різні типи систем. Кілька таких типів автоматизації часто поєднуються для отримання максимально якісних результатів.

Програмне забезпечення для автоматизації процесів може включати роботизовану автоматизацію процесів або автоматизацію бізнес-процесів, але саме завдяки поєднанню декількох типів автоматизації у єдину систему, фахівці зможуть отримати значно кращі результати. Кожна система автоматизує свої конкретні завдання. Автоматизована система бізнес-процесів буде супроводжувати весь процес від початку до кінця, виконуючи різні завдання, поки його виконання не буде завершено. Окремі завдання, як, наприклад, надсилання електронних листів клієнтам, може виконувати роботизована автоматизація.

Як правило, такі автоматизовані системи використовують клієнт-серверну архітектуру. Така архітектура розподіляє роботу з обробки даних між сервером та клієнтами, якими зазвичай є персональні комп'ютери. Комп'ютери мають значну обчислювальну потужність і тому здатні отримувати необроблені дані, які повертає сервер, і форматувати результат для виведення. Прикладні програми та процесори запитів можуть зберігатися та виконуватися на персональних комп'ютерах. Мережевий трафік зводиться до запитів на маніпулювання даними, що надсилаються з персональних комп'ютерів на сервер бази даних, і необроблені дані повертаються в результаті цього запиту.

Одним із головних недоліків клієнт-серверної архітектури є потреба у кваліфікованому персоналі для підтримки та адміністрування серверної частини. Якщо серверний застосунок був реалізований одним розробником, а потім система перейшла під управління до іншого розробника – цей фахівець повинен розуміти

та знати як додавати новий функціонал, редагувати уже наявний, розв’язувати проблеми, які виникають із серверним додатком.

Щоб уникнути таких проблем, потрібно використовувати ERP систему. Це система планування ресурсів підприємства, яка містить інтегровані програмні додатки, що допомагають керувати повсякденними алгоритмами, бізнес-процесами та операціями в сфері фінансів, людських ресурсів, розподілу та інших галузей. ERP системи є важливими програмами для більшості організацій, оскільки вони об’єднують усі процеси, необхідні для ведення діяльності, в єдину систему, яка також полегшує планування ресурсів. Системи ERP зазвичай працюють на інтегрованій програмній платформі, використовуючи загальні визначення даних, що працюють в одній базі даних.

Розроблена ERP система, яка використовується як серверний застосунок, у заданій предметній області надає фактично користувальницький інтерфейс для редагування функціональних можливостей серверного додатка. Тому розробнику, який надає реалізовану ERP систему, не потрібно буде постійно підтримувати її. А це своєю чергою розв’язує проблему потреби у кваліфікованому персоналі для адміністрування серверної частини системи.

В результаті вибору предметної області, було вирішено автоматизовувати керування студмістечком. У студентських гуртожитках існує проблема неоптимізованих процесів: живі неконтрольовані черги абітурієнтів на поселення; формування ордерів на поселення та інших документів вручну; інвентаризація у паперовому вигляді, - і це тільки початок. Тому розробити автоматизовану систему керування студмістечком є актуальною задачею.

Метою роботи є створення користувальницького інтерфейсу для ERP системи керування студмістечком та проведення інтеграції ERP системи у розроблений клієнтський застосунок за допомогою веб сервісів.

Для досягнення мети потрібно розв’язати наступні задачі: проаналізувати процеси керування студмістечком та принципи роботи зі студентами; здійснити порівняльний аналіз наявних застосунків для керування студмістечком; проаналізувати та обрати технології, які найкраще підходять для реалізації

системи; реалізувати клієнтський застосунок системи керування студмістечком; інтегрувати ERP систему керування студмістечком у розроблений клієнтський застосунок.

Об’єктом дослідження є процес керування студмістечком за допомогою ERP системи на платформі Microsoft Dynamics 365 Business Central.

Предметом дослідження є веб сервіси для інтеграції ERP системи керування студмістечком на платформі Microsoft Dynamics 365 Business Central.

У розроблений клієнтський застосунок системи керування студмістечком буде інтегровано ERP систему керування студмістечком на платформі Mircosoft Dynamics 365 Business Central, що дозволить розв’язати проблему неефективного керування студмістечком.

Робота складається зі вступу, 6 розділів, висновків до розділів, загальних висновків, списку використаних джерел (15 джерел), 1 додатку на 2 сторінках. Загальний обсяг дисертації – 92 сторінки. Основний зміст викладено на 78 сторінках. Роботу проілюстровано 22 таблицями та 32 рисунками.

1 ПОСТАНОВКА ЗАДАЧІ ІНТЕГРАЦІЇ ERP СИСТЕМ НА ПЛАТФОРМІ MICROSOFT DYNAMICS 365 BUSINESS CENTRAL ЗА ДОПОМОГОЮ ВЕБ СЕРВІСІВ

У даному розділі викладено постановку задачі інтеграції ERP систем на платформі Microsoft Dynamics 365 Business Central за допомогою веб сервісів.

У першому підрозділі сформовано мету роботи та завдання, які потрібно розв’язати для досягнення мети.

У другому підрозділі сформовано вимоги до функціональних можливостей ERP системи керування студмістечком на платформі Microsoft Dynamics 365 Business Central.

У третьому підрозділі сформовано вимоги до функціональних можливостей користувальницького інтерфейсу системи керування студмістечком.

Вкінці надано висновки до розділу.

1.1 Мета інтеграції ERP систем на платформі Microsoft Dynamics 365 Business Central за допомогою веб сервісів

Метою роботи є створення користувальницького інтерфейсу для ERP системи керування студмістечком та проведення інтеграції ERP системи у розроблений клієнтський застосунок за допомогою веб сервісів.

Завдання, які потрібно розв’язати для досягнення мети:

1. Проаналізувати процеси керування студмістечком та принципи роботи зі студентами
2. Здійснити порівняльний аналіз наявних застосунків для керування студмістечком
3. Проаналізувати та обрати технології, які найкраще підходять для реалізації системи

4. Реалізувати клієнтський застосунок системи керування студмістечком
5. Інтегрувати ERP систему керування студмістечком у розроблений клієнтський застосунок

1.2 Вимоги до функціональних можливостей ERP системи керування студмістечком на платформі Microsoft Dynamics 365 Business Central

ERP система керування студмістечком на платформі Microsoft Dynamics 365 Business Central повинна містити п’ять типів користувачів: “Адміністратор”, “Завідувач гуртожитку”, “Комендант”, “Персонал” та “Студент”.

ERP система керування студмістечком на платформі Microsoft Dynamics 365 Business Central повинна містити вісім сутностей: “Новина”, “Поселення”, “Оплата”, “Відпрацювання”, “Послуга”, “Контакт”, “Кімната” та “Документ”.

ERP система керування студмістечком на платформі Microsoft Dynamics 365 Business Central при CRUD операціях над користувачами та сутностями повинна одразу оновлювати дані у базі даних.

ERP система керування студмістечком на платформі Microsoft Dynamics 365 Business Central повинна надавати користувачу “Адміністратор” наступні функціональні можливості:

1. Авторизація у систему
2. Переглядати, додавати, редагувати, видаляти користувачів: “Адміністратор”, “Завідувач гуртожитку”, “Комендант”, “Персонал” та “Студент”
3. Переглядати, додавати, редагувати, видаляти сутності: “Новина”, “Поселення”, “Оплата”, “Відпрацювання”, “Послуга”, “Контакт”, “Кімната” та “Документ”

ERP система керування студмістечком на платформі Microsoft Dynamics 365 Business Central повинна надавати користувачу “Завідувач гуртожитку” наступні функціональні можливості:

1. Авторизація у систему
2. Можливість редагувати свої персональні дані
3. Переглядати, додавати, редагувати, видаляти користувачів: “Комендант”, “Персонал” та “Студент”
4. Переглядати, додавати, редагувати, видаляти сутності: “Новина”, “Поселення”, “Оплата”, “Відпрацювання”, “Послуга”, “Контакт”, “Кімната” та “Документ”

ERP система керування студмістечком на платформі Microsoft Dynamics 365 Business Central повинна надавати користувачу “Комендант” наступні функціональні можливості:

1. Авторизація у систему
2. Можливість редагувати свої персональні дані
3. Переглядати, додавати, редагувати, видаляти користувачів: “Персонал” та “Студент”
4. Переглядати, додавати, редагувати, видаляти сутності: “Новина”, “Поселення”, “Оплата”, “Відпрацювання”, “Послуга”, “Контакт”, “Кімната” та “Документ”

ERP система керування студмістечком на платформі Microsoft Dynamics 365 Business Central повинна надавати користувачу “Персонал” наступні функціональні можливості:

1. Авторизація у систему
2. Можливість редагувати свої персональні дані
3. Переглядати, додавати, редагувати, видаляти користувачів “Студент”
4. Переглядати, додавати, редагувати, видаляти сутності: “Новина”, “Поселення”, “Оплата”, “Відпрацювання”, “Послуга”, “Контакт”, “Кімната” та “Документ”

ERP система керування студмістечком на платформі Microsoft Dynamics 365 Business Central повинна надавати користувачу “Студент” наступні функціональні можливості:

1. Авторизація у систему
2. Можливість редагувати свої персональні дані
3. Переглядати користувачів “Студент”
4. Переглядати сутності: “Новина”, “Поселення”, “Оплата”, “Відпрацювання”, “Послуга”, “Контакт”, “Кімната” та “Документ”

ERP система керування студмістечком на платформі Microsoft Dynamics 365 Business Central повинна надавати власні веб сервіси для інтеграції або документацію щодо самостійної реалізації веб сервісів для інтеграції у клієнтський застосунок.

1.3 Вимоги до функціональних можливостей користувальницького інтерфейсу системи керування студмістечком

Розроблений клієнтський застосунок системи керування студмістечком повинен надавати користувальницький інтерфейс для взаємодії з ERP системою керування студмістечком на платформі Microsoft Dynamics 365 Business Central.

Розроблений клієнтський застосунок системи керування студмістечком повинен надавати користувальницький інтерфейс для взаємодії з обліковим записом користувача “Адміністратор” та містити наступні елементи:

1. Сторінка авторизації
2. Сторінка зміни тимчасового пароля
3. Сторінка профілю, де можна редагувати свої дані
4. Сторінки керування користувачами: “Адміністратор”, “Завідувач гуртожитку”, “Комендант”, “Персонал” та “Студент”
5. Сторінка керування новинами, де можна редагувати новини

6. Сторінка керування поселенням студентів, де можна надавати інформацію про поселення, прогрес поточного поселення, дозволяти завантажити ордер на поселення та договір
7. Сторінка керування оплатою студентів, де можна редагувати помісячну оплату
8. Сторінка керування відпрацюванням студентів, де можна редагувати інформацію про щомісячне відпрацювання
9. Сторінка керування послугами, де можна редагувати послуги для студентів, створювати та надавати доступ до різних видів документів
10. Сторінка керування контактами, де можна редагувати контакти

Розроблений клієнтський застосунок системи керування студмістечком повинен надавати користувальницький інтерфейс для взаємодії з обліковим записом користувача “Завідувач гуртожитку” та містити наступні елементи:

1. Сторінка авторизації
2. Сторінка зміни тимчасового пароля
3. Сторінка профілю, де можна редагувати свої дані
4. Сторінки керування користувачами: “Комендант”, “Персонал” та “Студент”
5. Сторінка керування новинами, де можна редагувати новини
6. Сторінка керування поселенням студентів, де можна надавати інформацію про поселення, прогрес поточного поселення, дозволяти завантажити ордер на поселення та договір
7. Сторінка керування оплатою студентів, де можна редагувати помісячну оплату
8. Сторінка керування відпрацюванням студентів, де можна редагувати інформацію про щомісячне відпрацювання
9. Сторінка керування послугами, де можна редагувати послуги для студентів, створювати та надавати доступ до різних видів документів
10. Сторінка керування контактами, де можна редагувати контакти

Розроблений клієнтський застосунок системи керування студмістечком повинен надавати користувальницький інтерфейс для взаємодії з обліковим записом користувача “Комендант” та містити наступні елементи:

1. Сторінка авторизації
2. Сторінка зміни тимчасового пароля
3. Сторінка профілю, де можна редагувати свої дані
4. Сторінки керування користувачами: “Персонал” та “Студент”
5. Сторінка керування новинами, де можна редагувати новини
6. Сторінка керування поселенням студентів, де можна надавати інформацію про поселення, прогрес поточного поселення, дозволяти завантажити ордер на поселення та договір
7. Сторінка керування оплатою студентів, де можна редагувати помісячну оплату
8. Сторінка керування відпрацюванням студентів, де можна редагувати інформацію про щомісячне відпрацювання
9. Сторінка керування послугами, де можна редагувати послуги для студентів, створювати та надавати доступ до різних видів документів
10. Сторінка керування контактами, де можна редагувати контакти

Розроблений клієнтський застосунок системи керування студмістечком повинен надавати користувальницький інтерфейс для взаємодії з обліковим записом користувача “Персонал” та містити наступні елементи:

1. Сторінка авторизації
2. Сторінка зміни тимчасового пароля
3. Сторінка профілю, де можна редагувати свої дані
4. Сторінки керування користувачами “Студент”
5. Сторінка керування новинами, де можна редагувати новини
6. Сторінка керування поселенням студентів, де можна надавати інформацію про поселення, прогрес поточного поселення, дозволяти завантажити ордер на поселення та договір

7. Сторінка керування оплатою студентів, де можна редагувати помісячну оплату
8. Сторінка керування відпрацюванням студентів, де можна редагувати інформацію про щомісячне відпрацювання
9. Сторінка керування послугами, де можна редагувати послуги для студентів, створювати та надавати доступ до різних видів документів
10. Сторінка керування контактами, де можна редагувати контакти

Розроблений клієнтський застосунок системи керування студмістечком повинен надавати користувальницький інтерфейс для взаємодії з обліковим записом користувача “Студент” та містити наступні елементи:

1. Сторінка авторизації
2. Сторінка зміни тимчасового пароля
3. Сторінка профілю, де можна редагувати свої дані
4. Сторінка новин, де можна переглядати новини
5. Сторінка поселенням, де можна переглядати інформацію про поселення, прогрес поточного поселення, завантажити ордер на поселення та договір
6. Сторінка оплати, де можна здійснити помісячну оплату
7. Сторінка відпрацювання, де можна переглядати інформацію про щомісячне відпрацювання
8. Сторінка послуг, де можна переглядати послуги, запросити надати послугу, створити та завантажити різні види документів
9. Сторінка контактів, де можна переглядати контакти

Розроблений клієнтський застосунок системи керування студмістечком повинен містити веб сервіси, сумісні з ERP системою керування студмістечком на платформі Microsoft Dynamics 365 Business Central, або використовувати реалізовані веб сервіси ERP системи керування студмістечком для її інтеграції

1.4 Висновки до розділу

Інтеграція ERP систем на платформі Microsoft Dynamics 365 Business Central за допомогою веб сервісів полягає у створенні користувальницького інтерфейсу для ERP системи керування студмістечком на платформі Microsoft Dynamics 365 Business Central та під’єднанні ERP системи до розробленого клієнтського застосунку.

ERP система керування студмістечком на платформі Microsoft Dynamics 365 Business Central повинна задовольняти вимоги до функціональних можливостей для її інтеграції у розроблений клієнтський застосунок. Для цього ERP системі достатньо містити вісім сутностей, п’ять типів користувачів та надавати можливість здійснювати CRUD операції над ними.

Розроблений клієнтський застосунок системи керування студмістечком повинен надавати користувальницький інтерфейс у формі персональних кабінетів для взаємодії з ERP системою керування студмістечком на платформі Microsoft Dynamics 365 Business Central та відповідати її функціональним вимогам.

2 АВТОМАТИЗАЦІЯ ПРОЦЕСУ КЕРУВАННЯ СТУДМІСТЕЧКОМ

У даному розділі викладено характеристику процесу керування студмістечком, причини автоматизації та методи її виконання.

У першому підрозділі представлено опис наявного процесу керування студмістечком.

У другому підрозділі представлено метод автоматизації процесу керування студмістечком за допомогою ERP системи на платформі Microsoft Dynamics 365 Business Central.

У третьому підрозділі представлено основний сценарій роботи з автоматизованою системою керування студмістечком.

Вкінці надано висновки до розділу.

2.1 Аналіз наявного процесу керування студмістечком

Наявні процеси керування студмістечком є неоптимізованими та неефективними. Щоб краще зрозуміти проблему керування студмістечком, розглянемо основний сценарій роботи зі студентами.

Спочатку абітурієнт, який пройшов конкурсний відбір у вищий навчальний заклад, отримує лист на пошту із запрошенням на подачу оригіналів документів до приймальної комісії. Як правило, у цьому листі вказують у який гуртожиток буде заселено студента. В деяких випадках, коли є проблеми із розрахунком вільних місць у гуртожитках для вступників, інформацію про поселення першокурсник отримує після подачі оригіналів документів та його зачислення на факультет.

Після отримання мінімальної інформації про поселення, студент направляється у вказаний гуртожиток та розпочинає процес поселення. На цьому

етапі бувають випадки, коли вступнику повідомляють, що сталася помилка, і йому потрібно поселятись у зовсім інший гуртожиток.

Тепер, коли студент потрапив у правильний гуртожиток, місць вистачає – потрібно написати декілька заяв на поселення. Вступник повинен звернутись до волонтерів, які зазвичай допомагають гуртожитку із процесом поселення, та взяти у них зразки заяв. Тут присутня неефективна жива черга, тому доведеться почекати, поки хтось із волонтерів звільниться та допоможе новоприбулому студенту. Під час заповнення документів, студенти часто спілкуються із волонтерами, які надають інформацію щодо їх подальших кроків.

Після заповнення заяв, волонтери показують кімнату, в яку було направлено вступника для проживання. В кімнату можна не потрапити з першого разу, адже сусіди можуть бути відсутні на момент поселення, а дублікату ключів у завідувача гуртожитку може не бути. В такій ситуації студента можуть тимчасово помістити у кімнату-ізолятор, яку використовують у екстренних випадках, якщо та, звичайно, вільна.

Потім, вступнику у паспортиста видають ордер на поселення. Тепер йому потрібно пройти медичний огляд у студентській поліклініці, та, якщо це хлопець – встати на військовий облік у військово-мобілізаційний відділ та районний військовий комісаріат. Часто трапляється так, що цей етап може розтягнутись на декілька днів, адже поселення може відбуватись у неприймальні дні військового комісаріату, або лікарі, які проводять медичний огляд можуть бути відсутні.

Після того, як студент встав на військовий облік та пройшов медичний огляд, йому потрібно звернутись до завідувача гуртожитку з ордером на поселення, довідкою від лікаря та відміткою районного військового комісаріату для продовження процесу поселення. На цьому етапі він також зустрічає живу чергу, адже в період поселення важко знайти момент, коли завідувач гуртожитку вільний. Коли студенту все ж вдається потрапити до завідувача гуртожитку, він складає з ним угоду на проживання, оплачує проживання мінімум на два місяця та подає заяву на отримання перепустки. Відділ перепусток в період поселення також перезавантажений, тому на перепустку прийдеться почекати до одного дня.

Так виглядає опис процесу поселення студентів першого курсу у гуртожиток (рисунок 2.1). У даному випадку можна виділити десять основних кроків, які потрібно пройти.

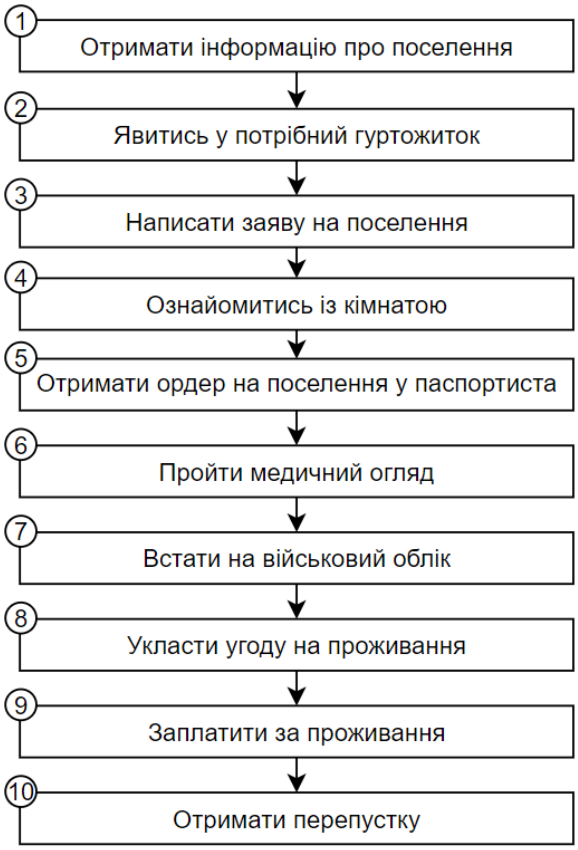

Рисунок 2.1 – Умовна схема процесу поселення студента першого курсу

Якщо говорити про студентів другого курсу і вище, для них процес повторного поселення спрощується.

Вони повинні прибути до гуртожитка у період поселення старших курсів, який, як правило, проходить за тиждень до поселення студентів першого курсу. Не зважаючи на те, що у цей період немає першокурсників, які створювали живі неефективні черги, процес поселення все одно повільний, адже тепер ці неефективні живі черги формуються із студентів старших курсів.

Далі студенти отримують ордер, проходять медичний огляд, укладають угоду на проживання, сплачують проживання та відновлюють перепустку.

Для студентів старших курсів процес поселення (рисунок 2.2) можна розділити на вісім кроків.

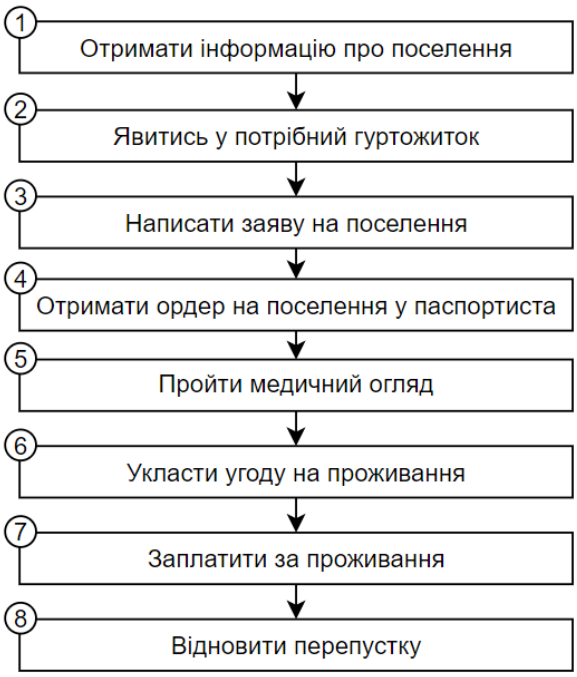

Рисунок 2.2 – Умовна схема процесу поселення студента старшого курсу

У студмістечку існує багато інших процесів, які є також неефективними та потребують автоматизації. Студенти під час проживання у гуртожитку повинні відвідувати щомісячні відпрацювання, а інформацію про такі відпрацювання вони можуть дізнатись лише у завідувача гуртожитку. Часто адміністрації необхідно повідомити якусь важливу новину, і зазвичай це відбувається у месенджері, де дуже легко не побачити, загубити повідомлення. Якщо студенту необхідно запросити нові меблі у кімнату – потрібно йти до завідувача гуртожитку, ставити питання чи меблі є в наявності, писати заяву.

Тому виконати якісну автоматизацію процесів керування студмістечком є актуальною задачею.

2.2 Автоматизація процесу керування студмістечком за допомогою ERP системи на платформі Microsoft Dynamics 365 Business Central

На даний момент для ведення обліку та автоматизації базових процесів керування студмістечком у гуртожитках використовують примітивні системи управління взаємовідносинами з клієнтами, тобто CRM системи.

Системи управління відносинами з клієнтами дозволяють персоналу керувати відносинами зі своїми поточними клієнтами та потенційними клієнтами. В таких системах зібрані інструменти та модулі, які допомагають компаніям або підприємствам залучати нових клієнтів і розвивати відносити з існуючими.

Основна мета CRM систем полягає в тому, щоб будь-яка взаємодія підприємства з клієнтом здійснювалася максимально швидко та ефективно. Окрім того, що системи надають підприємствам безперебійну взаємодію з клієнтами, CRM також можна налаштувати для автоматизації бізнес-процесів, які можуть бути складними для ручного виконання.

У студмістечку роль підприємства грає гуртожиток, а роль клієнта – студент. Проте, у гуртожитках використовуються набагато примітивніші CRM системи, де в основному ведеться лише облік. Тобто фактично такі системи являються графічним інтерфейсом для зручного керування базою даних

Щоб підвищити якість та оптимізувати процеси керування студмістечком, потрібно використовувати систему планування ресурсів підприємства, тобто ERP систему.

ERP система спрямована на організацію робочих процесів та операцій всередині компанії. З її допомогою здійснюється управління трудовими ресурсами, активами, ланцюжком ієрархічних процесів та фінансами [4]. Це більш серйозне програмне забезпечення. Воно призначене для використання насамперед у галузях зі складною та багаторівневою структурою процесів, якою і являється керування студмістечком.

Використовуючи якісну ERP систему, адміністрація гуртожитку зможе не тільки вести облік студентів, але й виконувати інвентаризацію обладнання, вести контроль своєчасності оплати та загальних витрат, автоматично генерувати необхідні документи.

В результаті вибору ERP системи було вирішено використовувати Microsoft Dynamics 365 Business Central (рисунок 2.3). Дана ERP система надає усі функціональні можливості CRM системи, яку використовують у гуртожитках, а також програмний інтерфейс для розширення та створення нових функціональних можливостей.

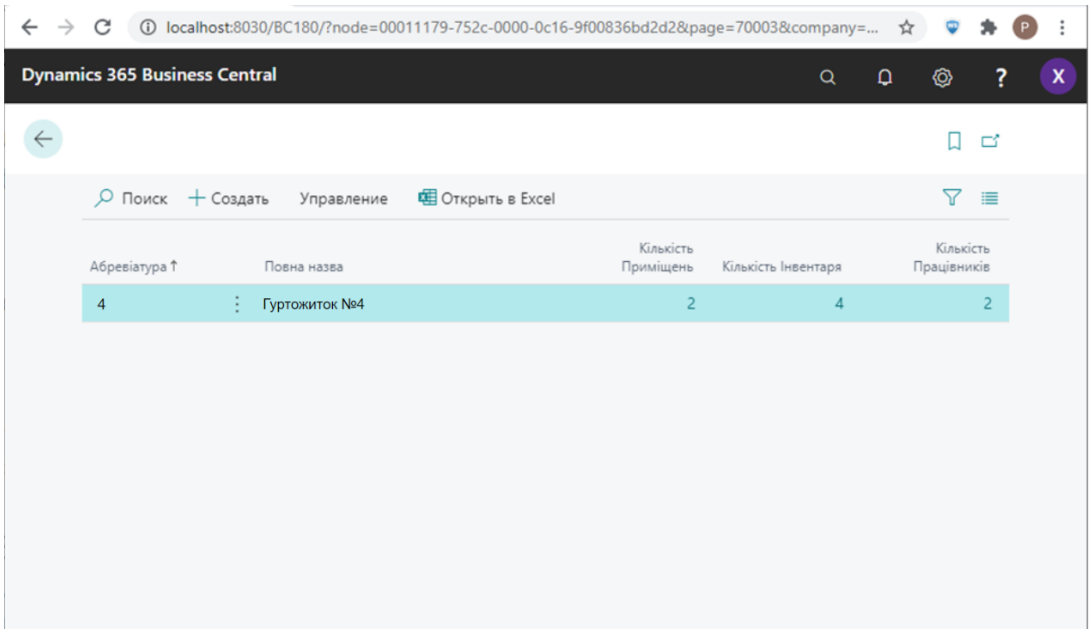

Рисунок 2.3 – Вкладка інвентаризації гуртожитку у ERP системі Microsoft Dynamics 365 Business Central

Для якісної автоматизації процесів керування гуртожитком потрібна власна ERP система керування студмістечком на платформі Microsoft Dynamics 365 Business Central, яка буде містити функціональні можливості платформи, та додаткові модулі для генерації документів та відправлення запитів від одного користувача системи до іншого. За рахунок розширених функціональних можливостей, така система дозволить автоматизувати більшу кількість процесів

керування студмістечком та надасть можливість відмовитись від використання примітивних CRM систем.

2.3 Аналіз основного сценарію роботи автоматизованої системи керування студмістечком

Щоб автоматизувати усі процеси керування гуртожитком потрібна ERP система керування студмістечком на платформі Microsoft Dynamics 365 Business Central. А для того, щоб якісно користуватися цією автоматизацією, необхідно реалізувати користувальницький інтерфейс для ERP системи та провести її інтеграцію у розроблений клієнтський застосунок за допомогою веб сервісів. У результаті інтеграції ми отримаємо повноцінну автоматизовану систему керування студмістечком.

При вступі до вищого навчального закладу абітурієнт надає свої контактні дані, а саме номер телефону та адресу електронної пошти. Тому спочатку в автоматизовану систему керування студмістечком буде завантажено список студентів, які поступили та потребують місце у гуртожитку. Згодом, вони будуть розподілені по гуртожитках, і кожному такому студенту буде відправлено SMS-повідомлення та лист на електронну пошту з даними для входу у систему керування студмістечком у вигляді логіну, яким буде адреса електронної пошти, або номер телефону, та тимчасовим паролю. Якщо студент спробує авторизуватись, йому одразу буде запропоновано змінити тимчасовий пароль.

Після успішної авторизації вступник потрапляє у персональний кабінет користувача “Студент”. Йому надається статус “В процесі поселення”. Студент одразу отримує необхідну інформацію про поселення, де буде вказано, у який гуртожиток його було направлено на проживання, рекомендовану дату для поселення та терміни, до якого часу необхідно завершити поселення. Також надається форма заяви на поселення, номер кімнати, інформацію про кімнату, інформацію про поверх, де знаходиться ця кімната, інформацію про сусідів, які

проживають у кімнаті, ордер на поселення, інформацію про медичний огляд, угоду на проживання та загальний прогрес процесу поселення. Виконавши усі необхідні кроки, студенту стає доступна сторінка оплати, яка перенаправляє його на наявний сервіс. Після успішної оплати вступнику надається статус “Проживає”.

Тепер студент може користуватись автоматизованою системою керування студмістечком. Йому доступна сторінка новин гуртожитку, відпрацювання, де відображено місячну норму робіт, сторінка послуг, де студент може запросити нові меблі, поселення гостя, поселення на літо, тимчасове виселення та сторінка контактів для зворотного зв’язку з адміністрацією.

Вкінці навчального року, якщо студент не планує поселятись на літо, йому потрібно оформити тимчасове виселення із гуртожитку. У цьому випадку студенту надається статус “Тимчасово виселений”. А на початку нового навчального року студенту знову присвоюється статус “В процесі поселення”. Таким чином у студента завжди будуть чергуватись три статуси: “В процесі поселення”, “Проживає” та “Тимчасово виселений”. Якщо студента відраховують з вищого навчального закладу, або розривають угоду на проживання, тобто виганяють із гуртожитку, студенту присвоюють статус “Виселений”, та після завершення процедури виселення – видаляють його обліковий запис.

Для адміністрування запитів студента існують користувачі “Адміністратор”, “Завідувач гуртожитку”, “Комендант” та “Персонал”, яким надається власні персональні кабінети із правами адміністрування. Такі користувачі публікують новини, інформацію про поселення, надають доступ до документів, редагують списки відпрацювань, контролюють своєчасність оплати за проживання та надають студентам послуги.

Отже, автоматизована система керування студмістечком допомагає розв’зати проблему неоптимізованих процесів у гуртожитку та полегшує взаємодію студента та адміністрації, робить її ефективнішою.

2.4 Висновки до розділу

Наявні процеси керування студмістечком є неефективними та неоптимізованими. Внаслідок аналізу цих процесів було виявлено ряд проблем, які допомогли визначити вимоги до основних функціональних можливостей системи керування студмістечком.

На даний момент у гуртожитках використовують примітивні CRM системи, які надають можливість вести облік студентів та інформації, пов’язаної з ними. Фактично такі системи являються графічним інтерфейсом для зручного керування базою даних. Аналіз такого способу керування студмістечком дав зрозуміти, що ефективніше буде використовувати ERP систему на платформі Microsoft Dynamics 365 Business Central з розширеними функціональними можливостями, що дозволить автоматизувати більшу кількість процесів керування гуртожитком.

Щоб якісно користуватись усіма можливостями ERP системи на платформі Microsoft Dynamics 365 Business Central, необхідно розробити користувальницький інтерфейс та інтегрувати ERP систему у розроблений клієнтський застосунок, що і утворить повноцінну автоматизовану систему керування студмістечком. Аналіз основного сценарію роботи такої системи дав зрозуміти, що сформованих вимог до функціональних можливостей системи достатньо для автоматизації необхідних процесів керування студмістечком.

3 ЗАСОБИ РЕАЛІЗАЦІЇ КОРИСТУВАЛЬНИЦЬКОГО ІНТЕРФЕЙСУ СИСТЕМИ КЕРУВАННЯ СТУДМІСТЕЧКОМ

У даному розділі представлено аналіз та обґрунтування вибору програмних засобів реалізації користувальницького інтерфейсу для ERP системи керування студмістечком на платформі Microsoft Dynamics 365 Business Central та її інтеграції у розроблений клієнтський застосунок.

Назви восьми підрозділів чітко вказують, який програмний засіб реалізації користувальницького інтерфейсу буде описано у підрозділі.

Вкінці надано висновки до розділу.

3.1 Обґрунтування вибору мови програмування JavaScript

JavaScript – це скриптова мова, мова сценаріїв на стороні клієнта. Це означає, що текстовий файл ASCII обробляється браузером клієнта, а не на онлайн-сервері. Це дозволяє завантажити веб сторінку без зв’язку з головним сервером за допомогою JavaScript. Наприклад, функція JavaScript може перевірити інтернет форму перед її надсиланням, щоб переконатися, чи всі вказані поля заповнені. Код JavaScript може видати повідомлення про помилку до того, як будь-яка інформація дійсно буде передана на сервер [9].

JavaScript є дуже простою у використанні мовою програмування. Крім того, вона має дуже простий синтаксис, який можуть легко зрозуміти як початківці, так і досвідчені програмісти, тому очевидно, що це робить Javascript дуже зручною мовою програмування для початківців.

JavaScript має величезну підтримку спільноти розробників. Простими словами, підтримка спільноти розробників допомагає вирішувати помилки, з якими може зіткнутись розробник. Програміст може легко знайти рішення на

StackOverflow для більшості помилок та проблем. JavaScript знаходиться на першому місці у багатьох рейтингах, заснованих на великій підтримці спільноти розробників.

JavaScript – це мова програмування, яка постійно розвивається, а це означає, що вона розробляється та підтримується відповідно до вимог сучасного світу.

У JavaScript є багато потужних фреймворків, такі як Vue.js та Angular.js, бібліотек, такі як React.js, та середовищ виконання, такі як Node.js. Усі вони можуть використовуватись одночасно при розробці додатків в залежності від вимог до функціональних можливостей. На сьогоднішній день React.js є однією з найпопулярніших бібліотек JavaScript, яка активно використовується в розробці інтерфейсів веб сайтів.

Відлагодження в JavaScript є дуже простим завдяки його підтримці в браузері. Тобто процес відлагодження можна виконувати безпосередньо в браузері. Це допомагає шукати та виправляти помилки швидко та ефективно.

Найважливішим моментом у JavaScript є те, що майже все в JavaScript є об'єктом, навіть функції. Це робить JavaScript дуже потужною мовою програмування. Об'єкти можуть мати властивості і їх можна присвоювати будь-якій змінній. Отже, функції, які є об'єктами в JavaScript, також можуть мати властивості і їх також можна присвоювати змінним. Функції в JavaScript є настільки ефективними, що їх можна навіть передавати в якості аргументу в іншу функцію. У будь-якому випадку, JavaScript – це не класова об'єктно-орієнтована мова програмування, а об'єктно-орієнтована мова програмування, заснована на прототипах.

Спочатку JavaScript використовувався тільки для динамічних веб-сторінок, але з роками він значно вдосконалився і розширив свій горизонт до різних областей, таких як розробка серверних застосунків, використовуючи Node.js, та розробка користувальницьких інтерфейсів, використовуючи React.js або Angular.js. JavaScript широко використовується для створення повноцінних веб-додатків з використанням різних технологічних стеків, таких як стек MERN (MongoDB, Express, React, Node) та стек MEAN (MongoDB, Express, Angular, Node).

Моя мета – розробити користувальницький інтерфейс для ERP системи керування студмістечком на платформі Microsoft Dynamics 365 Business Central та інтегрувати ERP систему у розроблений клієнтський застосунок. Для досягнення цієї мети JavaScript – це ідеальний вибір мови програмування.

3.2 Обґрунтування вибору середовища виконання Node.js

Node.js – це середовище виконання JavaScript поза браузером з складною архітектурою (рисунок 3.1), яке дозволяє писати серверний код для веб-сторінок, веб-додатків і програм командного рядка [7].

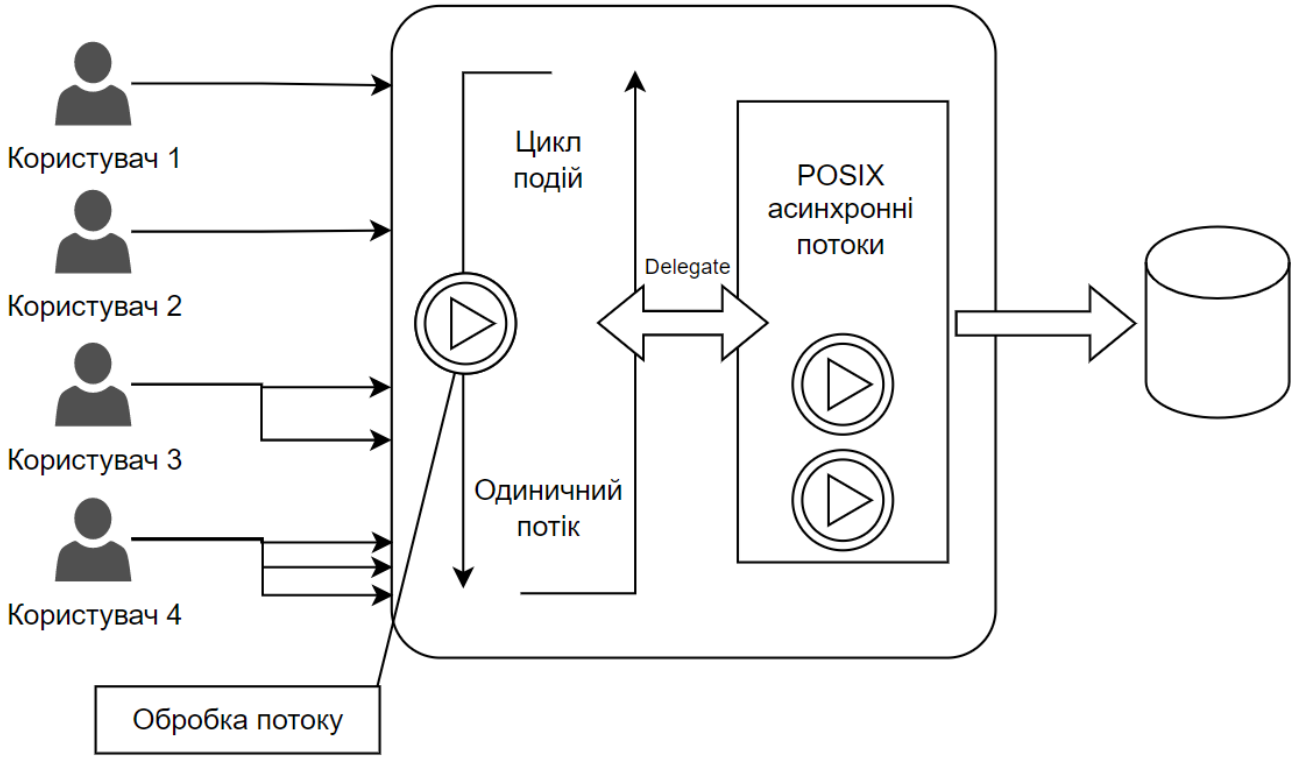

Рисунок 3.1 – Схема архітектури середовища виконання Node.js

Node.js – це незалежне середовище виконання на основі Chrome V8, двигуна JavaScript. Це означає, що це середовище виконання може самостійно виконувати код програми, незалежно від типу операційної системи [12].

Кількість готових модулів і зовнішніх бібліотек постійно зростає, чому сприяє використання пакетного менеджера NPM (Node Package Manager). У NPM опубліковано тисячі бібліотек і інструментів для розробки JavaScript. Завдяки постійній підтримці спільноти NPM зосереджується на заохоченні користувачів додавати нові пакети, щоб мати багато готових рішень для конкретної проблеми.

Потоки – це набори даних, і обробка цих даних вимагає першокласних методів обробки даних введення-виведення. Node.js добре справляється з таким типом введення-виведення, дозволяючи користувачам перекодувати медіафайли під час їх одночасного завантаження. Для обробки даних потрібно менше часу порівняно з іншими методами обробки даних. Його потоки роблять завдання введення-виведення набагато легшими.

У 2015 році деякі компанії, включаючи IBM, Microsoft, PayPal, Fidelity і SAP, організували Node.js Foundation. Це незалежна спільнота, метою якої є сприяння розвитку основних інструментів Node.js. Фонд був створений для прискорення розробки Node.js і забезпечення його широкого впровадження. Зараз кількість організацій, які використовують це середовище виконання в розробці, постійно зростає. До нього входять майже три сотні відомих компаній, таких як Medium, Uber та інші.

Node.js ідеально підходить для створення програм для соціальних мереж, потокових служб, інструментів для співпраці, інтерактивних бізнес-сайтів, програм обміну миттєвими повідомленнями тощо. Усі ці програми мають одну спільну рису – вони можуть передавати живий текст, аудіо чи відеодані.

Завдяки структурі, керованій подіями, і WebSockets, Node.js може ефективно обробляти потоки в реальному часі з високим трафіком і підтримувати потокове передавання високого рівня з потоками даних читання/запису. У результаті користувачі не відчувають затримок в обміні даними або оновленнях.

Отже, середовище Node.js є хорошим вибором для виконання JavaScript, що допоможе досягнути мети магістерської дисертації.

3.3 Обґрунтування вибору менеджера пакетів NPM

Менеджер пакетів – це інструмент для керування програмними пакетами. Він автоматизує процес інсталяції, оновлення, налаштування та видалення програмного забезпечення послідовним чином [13]. Це полегшує розробникам установку та керування складними програмними системами без необхідності вручну відстежувати та встановлювати залежності.

NPM (Node Package Manager) є потужним інструментом, який дозволяє розробникам легко ділитися та повторно використовувати код. Це менеджер пакетів за замовчуванням для середовища виконання JavaScript Node.js, який використовується мільйонами розробників у всьому світі для спільного використання та повторного використання коду. За допомогою NPM програміст може легко завантажувати та використовувати попередньо написані пакети коду JavaScript, що полегшує створення складних програм за допомогою лише кількох рядків коду.

Найбільша перевага NPM полягає в тому, що він готовий до роботи одразу після того, як розробник встановить Node.js. Це був перший із менеджерів пакетів Node і тому використовується скрізь. NPM є надійним вибором для розробки користувальницького інтерфейсу системи керування студмістечком.

3.4 Обґрунтування вибору системи контролю версій Git

Git – це система керування версіями, яка стала світовим стандартом контролю версій. Система Git являєтсья розподіленою. Це означає, що локальний клон проєкту є повним репозиторієм контролю версій [1]. Такі повнофункціональні локальні репозиторії спрощують роботу офлайн або віддалено. Розробники фіксують свою роботу локально, а потім синхронізують свою копію репозиторію з копією на сервері. Механіка роботи такого типу відрізняється від централізованого

керування версіями, де клієнти повинні синхронізувати код із сервером перед створенням нових версій коду.

Гнучкість і популярність Git роблять його чудовим вибором для будь-якої команди. Багато студентів навіть першого курсу вже знають, як користуватися Git. Спільнота користувачів Git створила ресурси для навчання розробників, а підтримка спільноти Git дозволяє легко отримати допомогу за потреби. Майже кожне середовище розробки має підтримку Git, а інструменти командного рядка Git реалізовані в кожній основній операційній системі.

Щоразу, коли робота зберігається, Git створює Commit. Commit – це моментальний знімок усіх файлів на певний момент часу. Якщо файл не змінився від одного Commit до наступного, Git використовує попередньо збережений файл. Така механіка роботи відрізняється від інших систем, які зберігають початкову версію файлу та ведуть облік дельт з часом.

Commit створює посилання на інший Commit, утворюючи графік історії розробки. Можна повернути код до попереднього commit, перевірити, як файли змінювалися від одного Commit до іншого і переглянути інформацію, наприклад, де і коли були внесені зміни. Commit ідентифікується у Git унікальним криптографічним хешем вмісту commit. Оскільки все хешується, неможливо внести зміни, втратити інформацію або пошкодити файли, не виявивши цього у Git.

Кожен розробник зберігає зміни у власному локальному сховищі коду. Як наслідок, може бути багато різних змін на основі одного Commit. Git надає інструменти для ізоляції змін і подальшого їх об’єднання. Гілки, які є легкими вказівниками на незавершену роботу, керують цим поділом. Після завершення роботи, створеної у гілці, її можна об’єднати назад у головну Master гілку команди рисунок 3.2).

Такі функціональні можливості необхідні для досягнення мети магістерської дисертації, адже при додаванні нових змін можна наткнутись на конфліктування модулів та інші проблеми, які можна буде швидко вирішити за допомогою Git.

3.5 Обґрунтування вибору хмарної платформи Heroku

Heroku – це хмарна платформа яка дозволяє веб розробникам і компаніям створювати, запускати та контролювати веб та мобільні застосунки. Крім того, це платформа, яка дозволяє веб розробникам розгортати, масштабувати та керувати онлайн-додатками за допомогою різних мов програмування [8].

Платформа містить багато вбудованих компонентів, починаючи від системи оповіщення про помилки, аналітичних інструментів розробки та закінчуючи службами безпеки для моніторингу, кешування, розсилки та мережевих додатків.

Основною перевагою платформи Heroku є вбудоване швидке розгортання проєктів. Служби для розгортання доступні одразу після реєстрації у платформі. Крім того, розробнику не доведеться турбуватися про інфраструктуру, оскільки програмне забезпечення подбає про це.

Хмарна платформа Heroku має вбудовані модулі для взаємодії з Node.js та Git, що робить її хорошим вибором для інтеграції ERP системи керування студмістечком у розроблений клієнтський застосунок.

3.6 Обґрунтування вибору бібліотеки React.js

ReactJS – це компонентна бібліотека JavaScript, створена Facebook. React полегшує створення інтерактивного інтерфейсу користувача за допомогою вбудованих компонентів та забезпечує ефективне керування станами цих компонентів [2]. Існує можливість об’єднувати різну кількість компонентів, щоб створювати складні програми, не втрачаючи їхнього стану в DOM (Document Object Model).

Незважаючи на те, що тут я аналізую React як інструмент для реалізації користувальницького інтерфейсу, його також можна використовувати для розробки мобільних застосунків за допомогою React Native, потужної рідної бібліотеки з відкритим кодом для мобільних додатків.

React вважається найпопулярнішою бібліотекою JavaScript. Її можна використовувати для створення всього, що користувач може бачите в інтернеті: будь це проста, але інтерактивна програма, як Instagram, складна програма для потокового передавання з великою базою користувачів і підтримкою кількох мов і регіонів, як Netflix, або програма, як Facebook, з дуже великим набором даних і великим навантаженням, здатною обробляти запити понад мільярда користувачів паралельно.

Завдяки компонентній архітектурі React дозволяє створювати високомасштабований односторінковий додаток SPA (Single Page Application), у якому вміст сторінки динамічно завантажується під час взаємодії користувача без перезавантаження всієї сторінки. Однак це створює ряд проблем. Існують такі ситуації, коли потрібно оновлювати DOM для кожної зміни, викликаної взаємодією користувача на веб сторінці. Будь-яка дія може змусити DOM, який є структурою дерева, оновитися. І, якщо веб сторінка клієнта складна, наявність кількох компонентів інтерфейсу користувача може призвести до зниження швидкодії системи.

Щоб розв’язати цю проблему, React використовує концепцію віртуального DOM, який можна розглядати як копію реального DOM (рисунок 3.2).

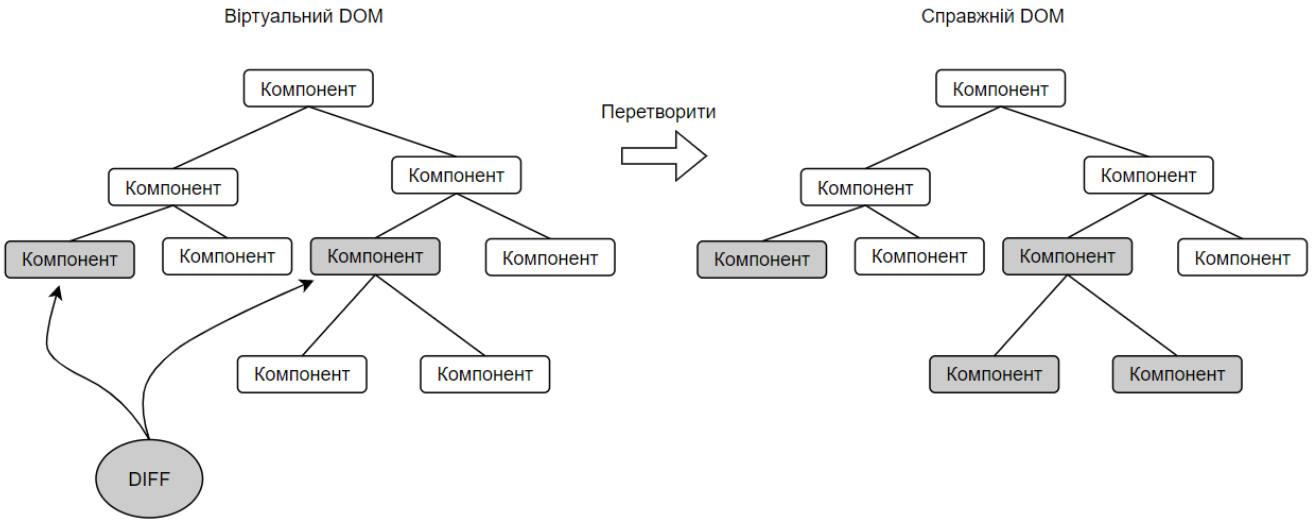

Рисунок 3.2 – Схема взаємодії справжнього DOM та віртуального DOM

Тепер усі зміни, спричинені взаємодією користувача чи іншими подіями, спочатку обробляються віртуальним DOM, а справжній DOM оновиться тільки тоді, коли ядро React буде вважати за потрібним це зробити. Така механіка позбавляє користувача від масового повторного створення дерева DOM для кожної тривіальної зміни, і, як результат, програма стає швидкою та ефективнішою.

Найбільша перевага React з точки зору розробника, яка також є причиною того, що React став найпопулярнішою бібліотекою JavaScript, полягає в тому, що ця бібліотека є ідеальною для програмістів початківців за рахунок своєї простоти та хорошої документації. Кожен, хто має базове розуміння HTML і JavaScript, може швидко почати роботу з React. Незважаючи на те, що він використовує спеціальне розширення синтаксису JSX (JavaScript XML) для змішування JavaScript і HTML, що являється нетрадиційним для новачків, це не завадило React стати найулюбленішою бібліотекою JavaScript серед громади розробників.

На відміну від інших звичайних фреймворків, React не обмежує програміста в плані ставлення до структури свого коду. Це дає розробникам React свободу виражати власні архітектурні стилі для створення програм. Тим, хто любить працювати з чистим JavaScript, подобається ця гнучкість, оскільки вони не звикли до ідеї фреймворку, який контролює структуру коду в програмі.

Враховуючи усі переваги, React.js є підходящою бібліотекою для реалізації користувальницького інтерфейсу системи керування студмістечком.

3.7 Обґрунтування вибору бібліотеки контролю станів Redux

Redux – це бібліотека JavaScript для керування станами компонентів застосунку та їх зберігання [3]. Кожен компонент системи може отримати доступ до будь-якого стану, який йому потрібен, залежно від обставин. Redux створює процес для взаємодії зі сховищем станів так, що компоненти не просто оновлюють чи зчитують сховище випадковим чином. Redux повністю автономно керує станами та надає доступ до них усім компонентам по чітко визначеній структурі.

У Redux можна виділити три основних компоненти: Action, Store та Reducer (рисунок 3.3).

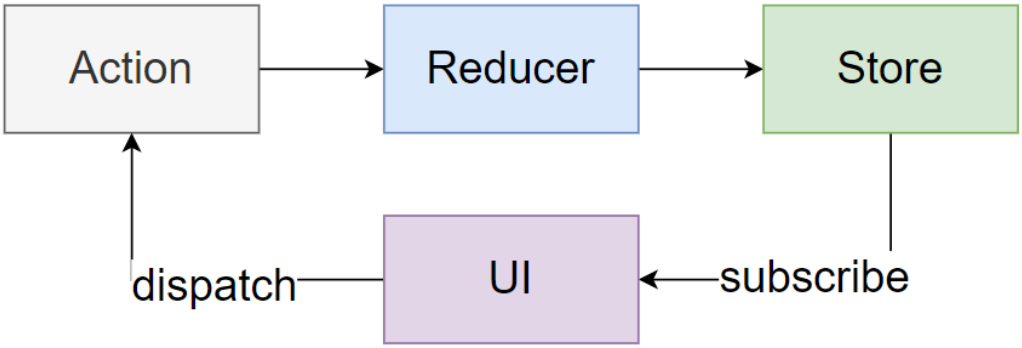

Рисунок 3.3 – Схема компонентів бібліотеки керування станами Redux

Action – це подія, задача якої – надсилати дані із програми до Store в результаті взаємодії клієнта з користувальницьким інтерфейсом. Store, або сховище, зберігає стани компонентів або програми. Reducer – це функція, яка бере поточний стан компонента або програми, виконує дію та повертає новий стан.

У системі керування студмістечком потрібно реалізувати стани користувача “Студент”. Окрім цього, у сутностей системи також може бути декілька станів. Тому використання бібліотеки керування станами Redux є необхідністю для досягнення мети магістерської дисертації.

3.8 Обґрунтування вибору середовища розробки JetBrains WebStorm

WebStrom – це середовище розробки для програмування на JavaScript та TypeScript [10]. Це оптимальний вибір як для приватного використання, так і для розробки в команді. Він дійсно підходить всім завдяки різноманітності запропонованих шаблонів для налаштування проєкту та тестування. У WebStorm

реалізовано зручне керування версіями, швидкий аналіз коду, який допомагає відстежувати проблеми без зупинки процесу розробки.

WebStorm популярний серед спільноти розробників завдяки своїй модульній структурі та зручності для користувача. У IDE є можливість налаштування проєкту за допомогою плагінів, можливість створення версій для приватних та командних проєктів, а також є сховище із широким набором шаблонів проєктів з уже налаштованими фреймворками. WebStorm сприяє розробці високоякісного структурованого коду за допомогою аналізу коду у фоновому режимі, який підтримує структуру коду та проєкту завдяки авторському завершенню рядків коду, які починає друкувати розробник, та інтелектуальному коментуванню.

Оскільки усі обрані програмні засоби реалізації зв’язані з мовою JavaScript, WebStorm – це хороший вибір середовища розробки для реалізації користувальницького інтерфейсу системи керування студмістечком.

3.9 Висновки до розділу

Вибір мови програмування JavaScript робить розробку клієнтського інтерфейсу для системи керування студмістечком зручною завдяки процесу відлагодження в браузері та підвищує швидкодію застосунку завдяки виконанню на стороні клієнта.

Використання середовища виконання Node.js, завдяки самостійного виконання коду JavaScript та його потоковій структурі, підвищує продуктивність усієї системи.

Менеджер пакетів NPM самостійно встановлює пакети та налаштовує необхідні залежності, що сильно спрощує розробку застосунку.

Завдяки використанню системи контролю версій Git можна керувати та відстежувати поетапні зміни у проєкті, що також сильно допомагає у процесі розробки додатка.

Вибір хмарної платформи Heroku обґрунтовується наявністю вбудованих компонентів, які допомагають швидко розгорнути та налаштувати систему керування студмістечком.

Бібліотека React.js надає потужні сервіси для реалізації користувальницького інтерфейсу, завдяки яким спрощується розробка клієнтської частини застосунку без обмежень в плані структури проєкту.

Використання бібліотеки контролю станів Redux.js, завдяки своїм основним компонентам, які керують станами користувача “Студент” та інших сутностей, допомагає реалізувати необхідні функціональні можливості системи керування студмістечком.

Середовище розробки JetBrains WebStorm розроблене спеціально для програмування на мові JavaScript. Його вибір обґрунтовується наявністю вбудованих функцій, які спрощують процес розробки та роблять його ефективнішим.

4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ КОРИСТУВАЛЬНИЦЬКОГО ІНТЕРФЕЙСУ СИСТЕМИ КЕРУВАННЯ СТУДМІСТЕЧКОМ

У даному розділі викладено характеристику програмної реалізації користувальницького інтерфейсу для ERP системи керування студмістечком на платформі Microsoft Dynamics 365 Business Central.

У першому підрозділі представлено опис архітектури автоматизованої системи керування студмістечком та користувальницького інтерфейсу.

У другому підрозділі представлено опис функціональних можливостей системи керування студмістечком згідно зі сформульованими вимогами.

У третьому підрозділі представлено ієрархічну модель системи та опис компонентів користувальницького інтерфейсу, які допомагають реалізувати функціональні можливості системи керування студмістечком.

Вкінці надано висновки до розділу.

4.1 Опис архітектури системи керування студмістечком

Архітектура автоматизованої системи керування студмістечком складається з інфраструктури ERP системи на платформі Microsoft Dynamics 365 Business Central, користувальницького інтерфейсу та представляється клієнт-серверною архітектурою.

Архітектуру клієнт-сервер можна порівняти з моделлю споживач-виробник, де клієнт грає роль споживача, тобто запитує послуги, а сервер є виробником, тобто постачальником послуг.

Клієнт-серверна архітектура, яку також називають моделлю клієнт-сервер, – це мережева програма, яка розподіляє завдання та навантаження між клієнтами та сервером, які знаходяться в одній системі або об’єднані мережею [11].

Взаємодія між елементами клієнт-серверної архітектури здійснюється за допомогою створення протоколу транспортного рівня. Спочатку клієнт надсилає свій запит через мережевий пристрій. Потім мережевий сервер приймає та обробляє запит користувача. Після цього сервер доставляє відповідь клієнту.

Створюючи протокол транспортного рівня та використовуючи модель клієнт-сервер, користувач повинен дотримуватись чітко визначених вимог. Протокол транспортного рівня повинен відповідати наступним критеріям: клієнт-серверне з'єднання, клієнт-серверна взаємодія, клієнт-серверна автентифікація та інші способи забезпечення цілісності даних після того, як пакети будуть доставлені та оброблені.

Як правило, клієнт-серверна архітектура складається з сервера бази даних, клієнтського застосунку та серверного застосунку (рисунок 4.1).

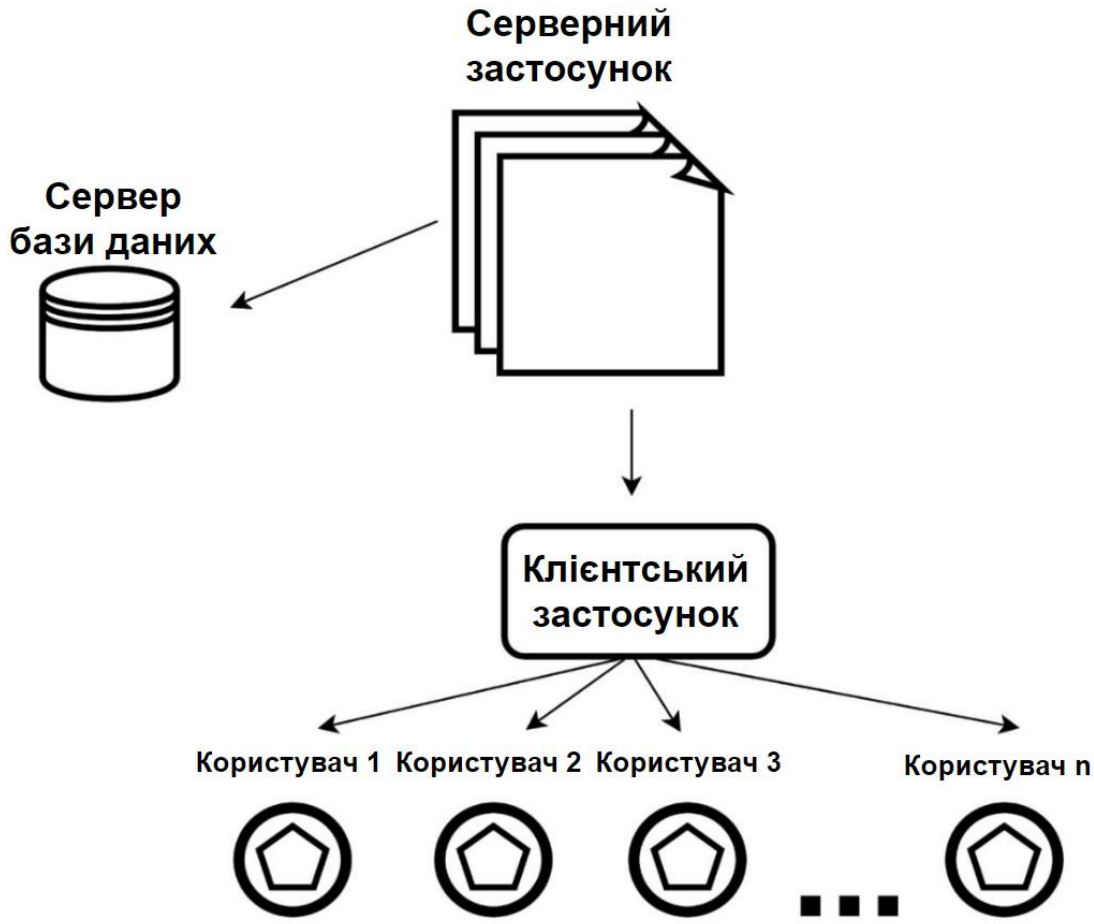

Рисунок 4.1 – Схема клієнт-серверної архітектури

Тепер розглянемо архітектуру автоматизованої системи керування студмістечком (рисунок 4.2).

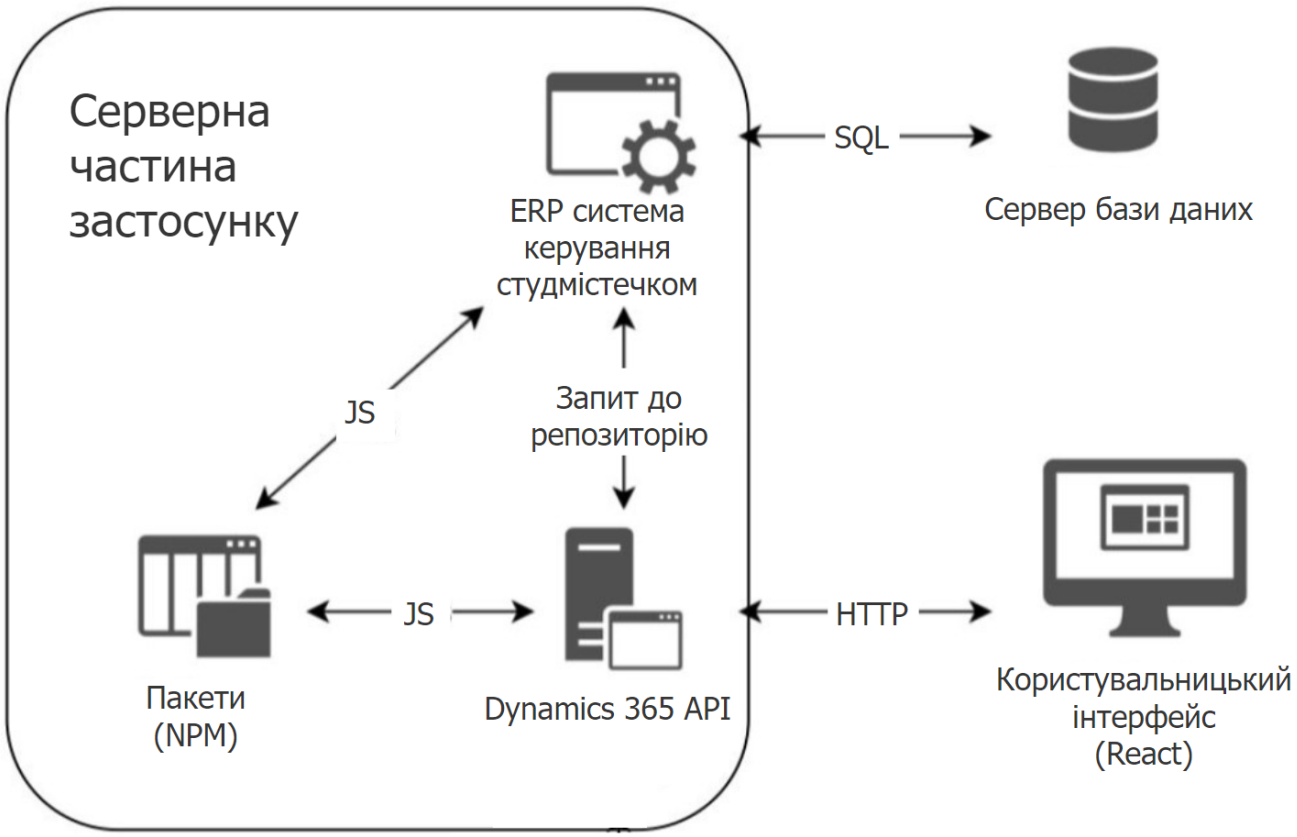

Рисунок 4.2 – Схема архітектури системи керування студмістечком

На серверній частині застосунку відбувається обробка запитів за допомогою ERP системи керування студмістечком на платформі Microsoft Dynamics 365 Business Central. За керування запитами відповідає Microsoft Dynamics 365 API. Взаємодія між клієнтською та серверною частинами здійснюється за допомогою HTTP протоколу.

Мета моєї магістерської дисертації – реалізація клієнтської частини автоматизованої системи керування студмістечком. Клієнтська частина – це користувальницький інтерфейс для взаємодії з ERP системою керування студмістечком на платформі Microsoft Dynamics 365 Business Central. Тому розглянемо архітектуру клієнтського застосунку системи (рисунок 4.3).

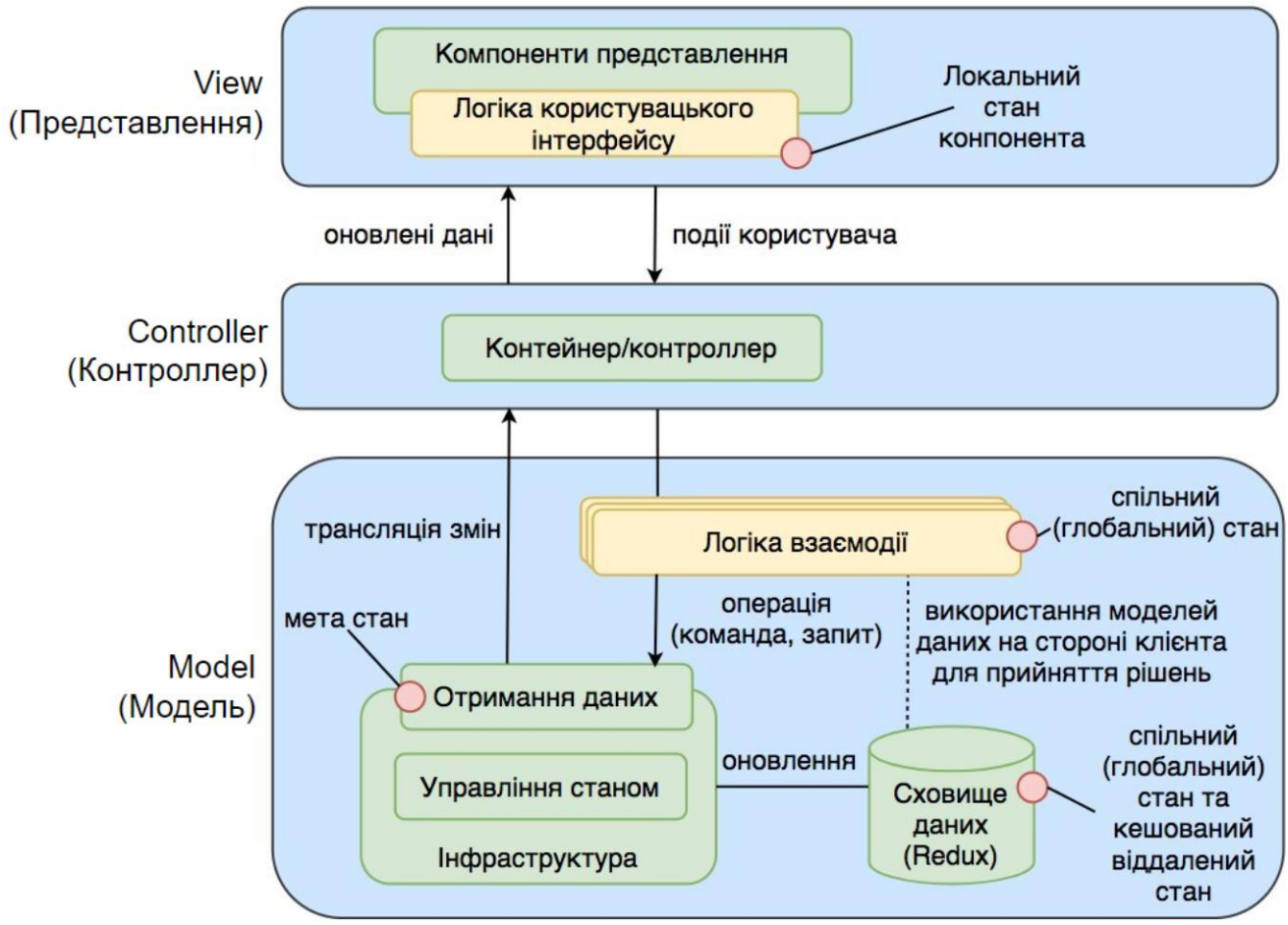

Рисунок 4.3 – Схема архітектури клієнтського застосунку системи керування студмісткечком

Архітектура клієнтського застосунку представлена шаблоном MVC (Model, View, Controller) та розділена на три основні логічні компоненти відповідно. Кожен із цих компонентів призначений для керування конкретними елементами розробки додатка. MVC є однією з найбільш поширених структур веб розробки для створення гнучких і масштабованих проєктів.

Використовуючи шаблон MVC, розробники можуть розділяти внутрішній та зовнішній частини коду на окремі розділи. Це значно спрощує керування та налаштування будь-якої сторони, не впливаючи на іншу.

Уся логіка користувача, пов’язана з даними, представлена компонентом Model. Це можуть бути будь-які додаткові дані, пов’язані з бізнес-логікою, або дані, які передаються між компонентами View та Controller. Наприклад, об’єкт

«Студент» може отримувати дані клієнта з бази даних, редагувати їх, а потім оновлювати дані в базі даних або використовувати їх для відображення у View.

Усі функціональні можливості користувальницького інтерфейсу програми реалізовано в частині архітектури View. Наприклад, View користувача “Студент” включатиме кожен елемент користувальницького інтерфейсу, з яким взаємодіє користувач: вікно з інформацією про поселення, меню вкладок та інші елементи персонального кабінету.

Компонент Controller служить інтерфейсом між компонентами Model і View. Якщо компонент Model користувача “Студент” реалізує CRUD операції з оновленням бази даних, Controller користувача “Студент” оброблятиме всі взаємодії користувача з елементами View для виконання заданих CRUD операцій.

4.2 Опис функціональних можливостей користувальницького інтерфейсу системи керування студмістечком

Характеристика основних функціональних можливостей користувальницького інтерфейсу системи керування студмістечком представлена у вигляді UML діаграм прецедентів. Для ефективнішого проєктування діаграм потрібно використовувати скорочене представлення складних прецедентів.

Керування сутністю в інтерфейсі користувача буде здійснюватись на вкладці з назвою сутності за допомогою таблиці редагування цих сутностей, де можна переглядати сутності, створювати нову, редагувати та видаляти наявні. Опис керування сутністю системи повинен містити наступні прецеденти: “Створити нову сутність”, “Редагувати наявну сутність”, “Видалити наявну сутність” та “Переглядати список сутностей”. Такі прецеденти описують CRUD операції та будуть представлені у вигляді загального прецеденту “CRUD операції над сутністю” (рисунок 4.4).

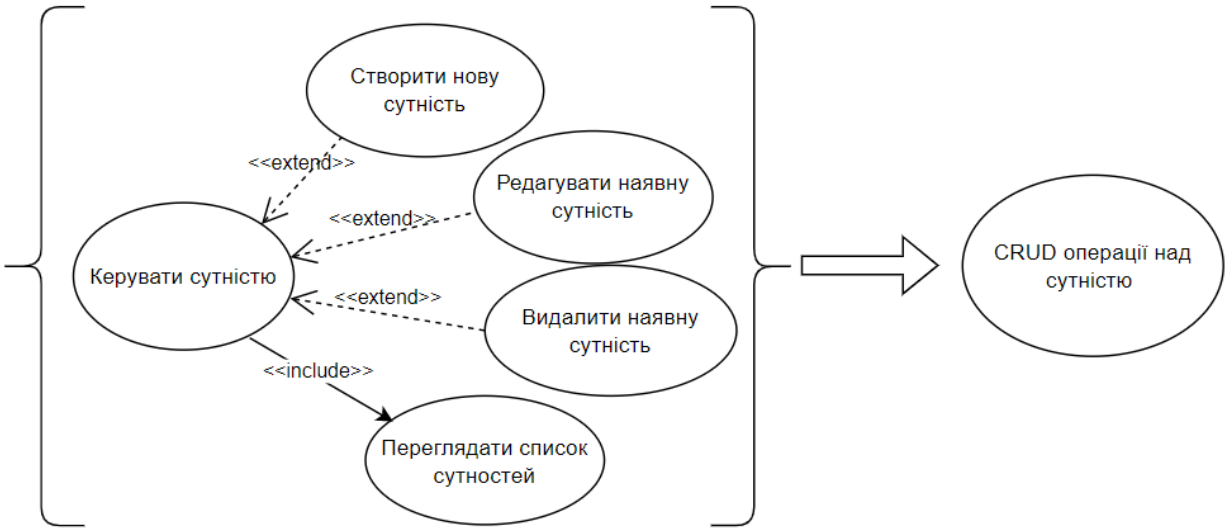

Рисунок 4.4 – Діаграма скороченого представлення прецеденту “Керувати сутністю”

Скорочене представлення складних прецедентів буде застосовано до прецедентів керування сутностями, наприклад керування “Новинами” (рисунок 4.5), та користувачами, наприклад керування користувачем “Завідувач гуртожитку” (рисунок 4.6).

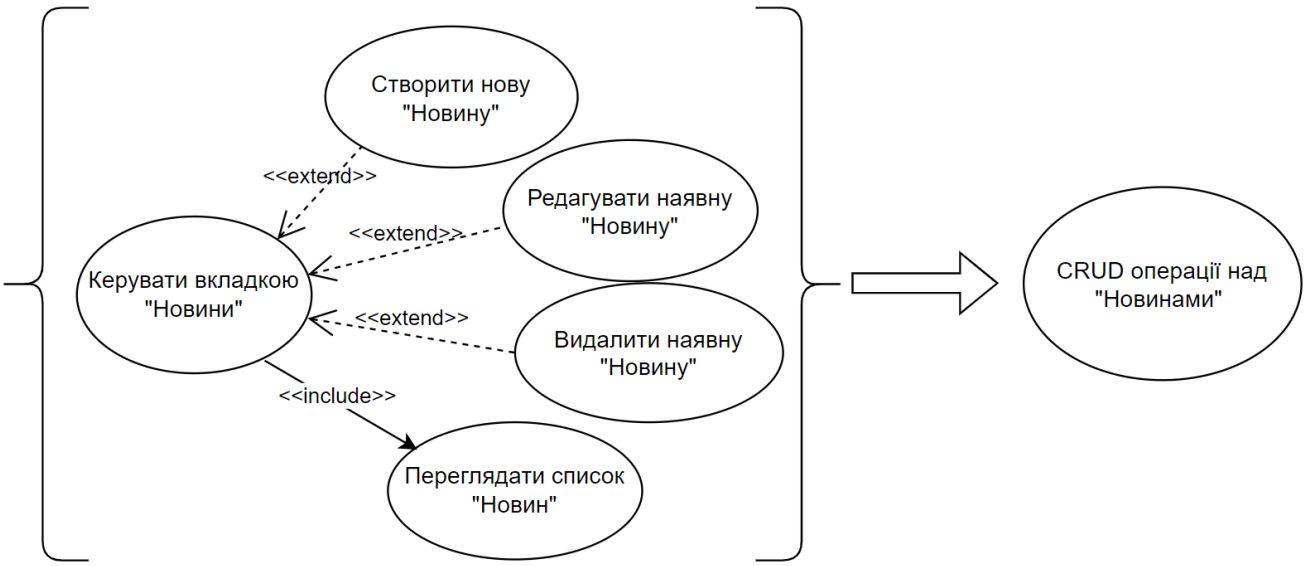

Рисунок 4.5 – Діаграма скороченого представлення прецеденту керування “Новинами”

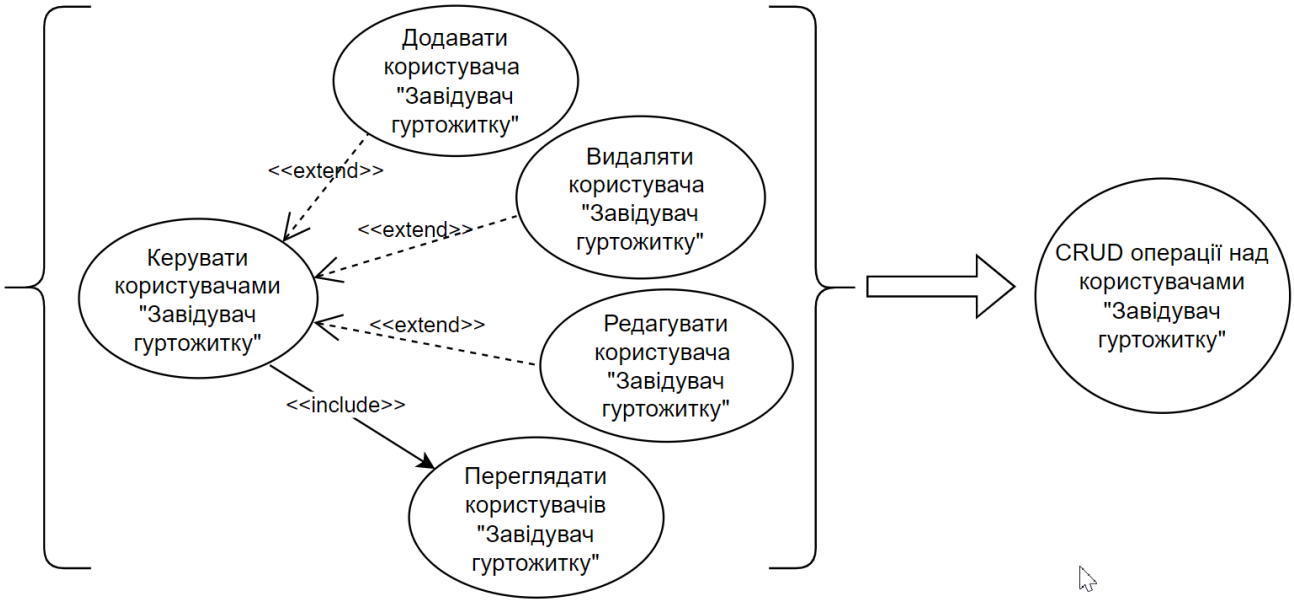

Рисунок 4.6 – Діаграма скороченого представлення прецеденту керування користувачем “Завідувач гуртожитку”

Користувач “Адміністратор” – це ключовий обліковий запис адміністрування системи керування студмістечком, який призначений для використання фахівцем, який буде підтримувати систему у майбутньому.

На сторінці власного профілю він може редагувати свої персональні дані: змінити ім’я, логін та пароль.

Вкладка “Користувачі” представлена у вигляді таблиці, де користувач “Адміністратор” може додати нового користувача, редагувати дані наявного користувача та видаляти їх облікові записи. Користувач “Адміністратор” може керувати усіма типами користувачів: “Адміністратор”, “Завідувач гуртожитку”, “Комендант”, “Персонал” та “Студент”.

Вкладка “Інформація про поселення” представлена у вигляді таблиці, де користувач “Адміністратор” може надавати, редагувати або видаляти різні види інформації про поселення для усіх або конкретних студентів. Тут адміністратор може вивантажити ордер на поселення та угоду на проживання для конкретного студента.

Вкладка “Послуги” представлена у вигляді таблиці, де користувач

“Адміністратор” може керувати послугами, які надає гуртожиток, оброблювати запити студентів на отримання послуг та вивантажувати необхідний документ або заяву для конкретного студента.

Вкладка “Новини” представлена у вигляді таблиці, де користувач “Адміністратор” може публікувати нову новину, редагувати та видаляти наявні новини для усіх студентів.

Вкладка “Оплата” представлена у вигляді таблиці, де користувач “Адміністратор” може надавати, редагувати та видаляти помісячні рахунки для конкретного студента.

Вкладка “Відпрацювання” представлена у вигляді таблиці, де користувач “Адміністратор” може надавати, редагувати та видаляти інформацію про помісячну норму відпрацювань.

Вкладка “Контакти” представлена у вигляді таблиці, де користувач “Адміністратор” може додавати, редагувати та видаляти контакти адміністрації гуртожитку для зворотного зв’язку.

При використанні системи керування студмістечком на практиці, користувач “Адміністратор” може не брати активну участь у процесах керування. Наприклад, навряд чи фахівець, якому буде видано обліковий запис адміністратора буде відмічати у таблиці “Відпрацювання”, скільки студент відпрацював годин. Такі функціональні можливості повинні бути в облікових записах “Завідувач гуртожитку”, “Комендант” та “Персонал”, адже ведення процесу керування гуртожитком їх основна мета.

Причина додавання таких функціональних можливостей користувачу “Адміністратор” полягає в ієрархії прав доступу. У системі керування студмістечком повинен бути супер користувач зі всіма правами інших користувачів, яким і являється користувач “Адміністратор”. Його основна відмінність від інших типів користувачів полягає у можливості створювати нові облікові записи користувачів “Адміністратор” та “Завідувач гуртожитку”. Спроєктуємо діаграму прецедентів користувача “Адміністратор” (рисунок 4.7).

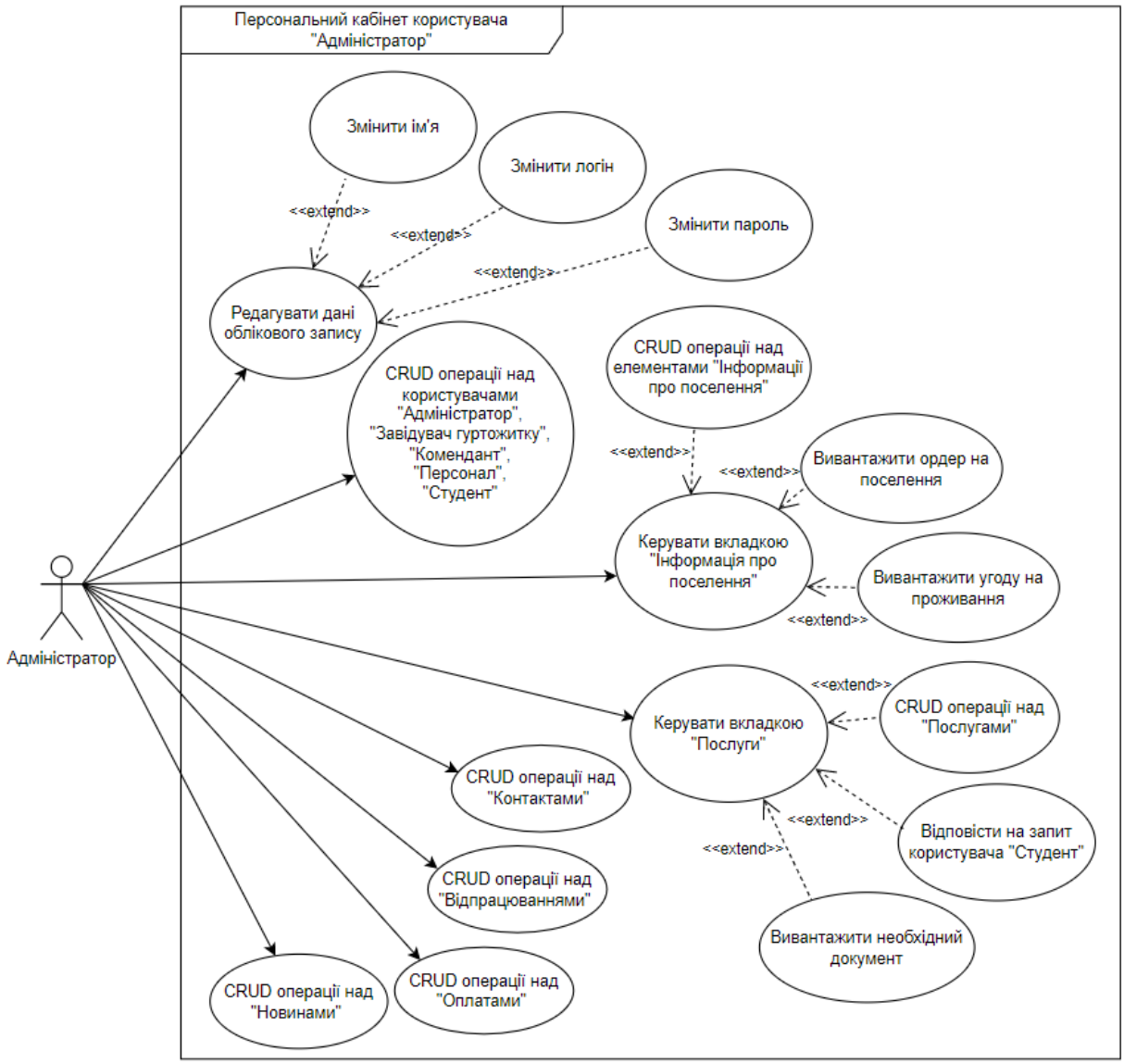

Рисунок 4.7 – Діаграма прецедентів користувача “Адміністратор”

Користувач “Завідувач гуртожитку” – це обліковий запис адміністрування системи керування студмістечком, який призначений для використання фахівцем на посаді завідувача гуртожитку. Саме завідувач гуртожитку найбільше взаємодіє зі студентами та контролює усі процеси керування гуртожитком.

Функціональні можливості користувача “Завідувач гуртожитку” повністю повторюють можливості облікового запису адміністратора, окрім керування користувачами “Адміністратор” та “Завідувач гуртожитку”. Спроєктуємо діаграму прецедентів користувача “Завідувач гуртожитку” (рисунок 4.8).

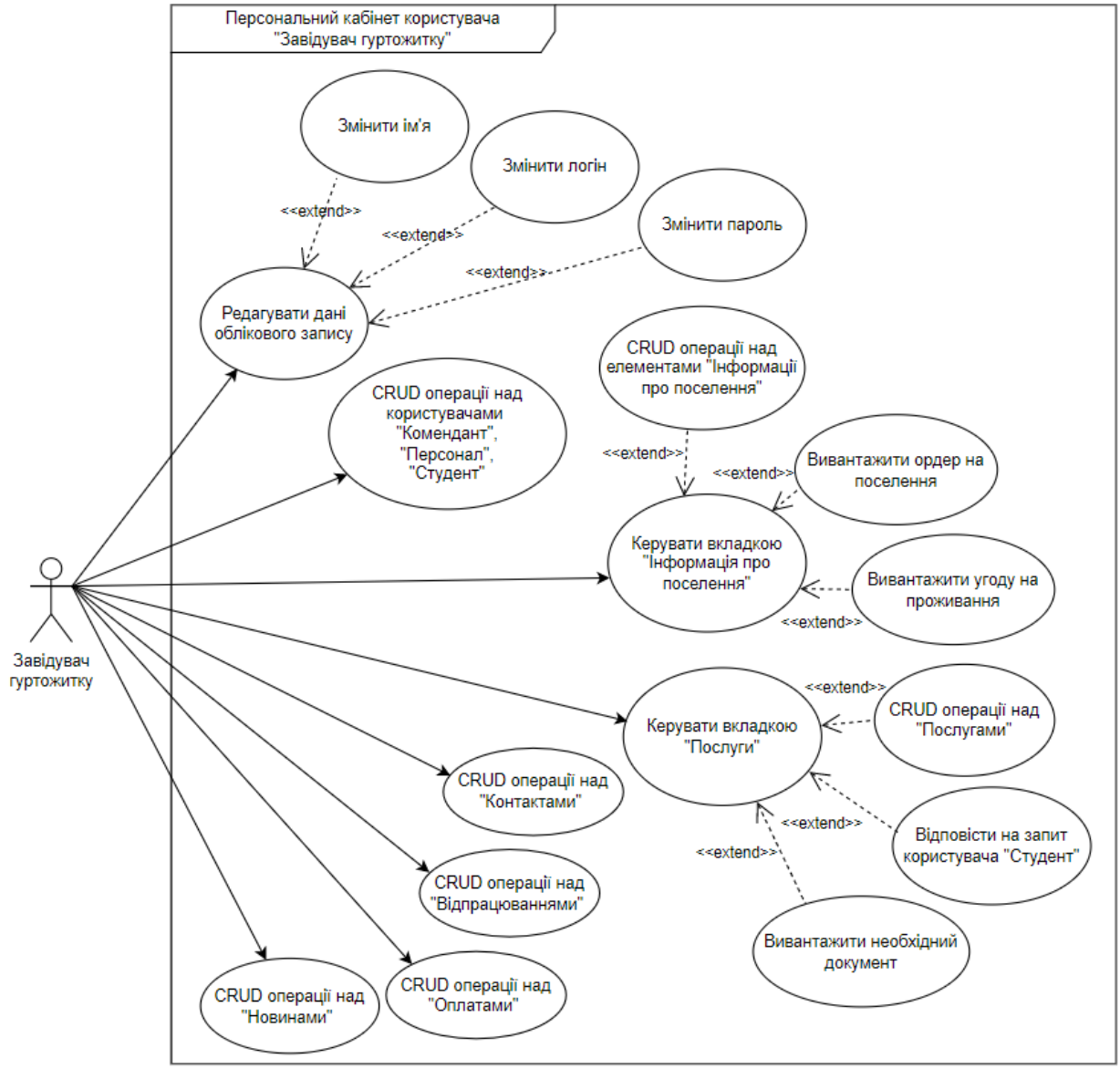

Рисунок 4.8 – Діаграма прецедентів користувача “Завідувач гуртожитку”

Користувач “Комендант” – це обліковий запис адміністрування системи керування студмістечком, який призначений для використання фахівцем на посаді коменданта гуртожитку. Комендант вважається заступником завідувача, тому він також бере активну участь у контролюванні процесів керування гуртожитком.

Функціональні можливості користувача “Комендант” повністю повторюють можливості облікового запису завідувача, окрім керування користувачами “Комендант”. Спроєктуємо діаграму прецедентів користувача “Комендант” (рисунок 4.9).

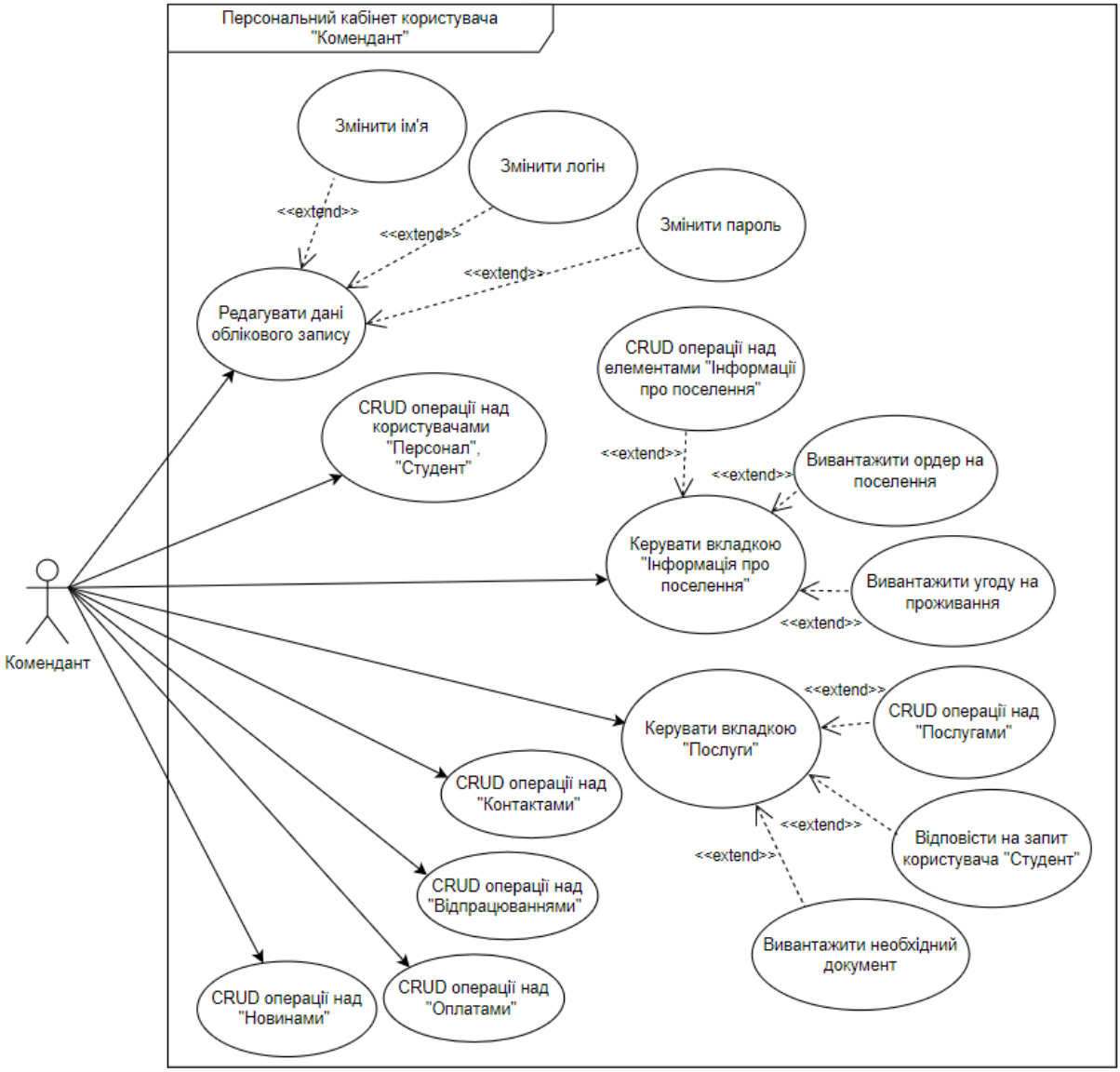

Рисунок 4.9 – Діаграма прецедентів користувача “Комендант”

Користувач “Персонал” – це обліковий запис адміністрування системи керування студмістечком, який призначений для використання фахівцем спеціально для керування автоматизованою системою. Саме персонал повинен контролювати своєчасність оплати студентів, норми їх відпрацювань, відповідати на запити з надання послуг та повідомляти коменданта чи завідувача про процеси у системі за необхідністю.

Користувач “Персонал” може керувати лише обліковими записами студентів. Спроєктуємо діаграму прецедентів користувача “Персонал” (рисунок 4.10).

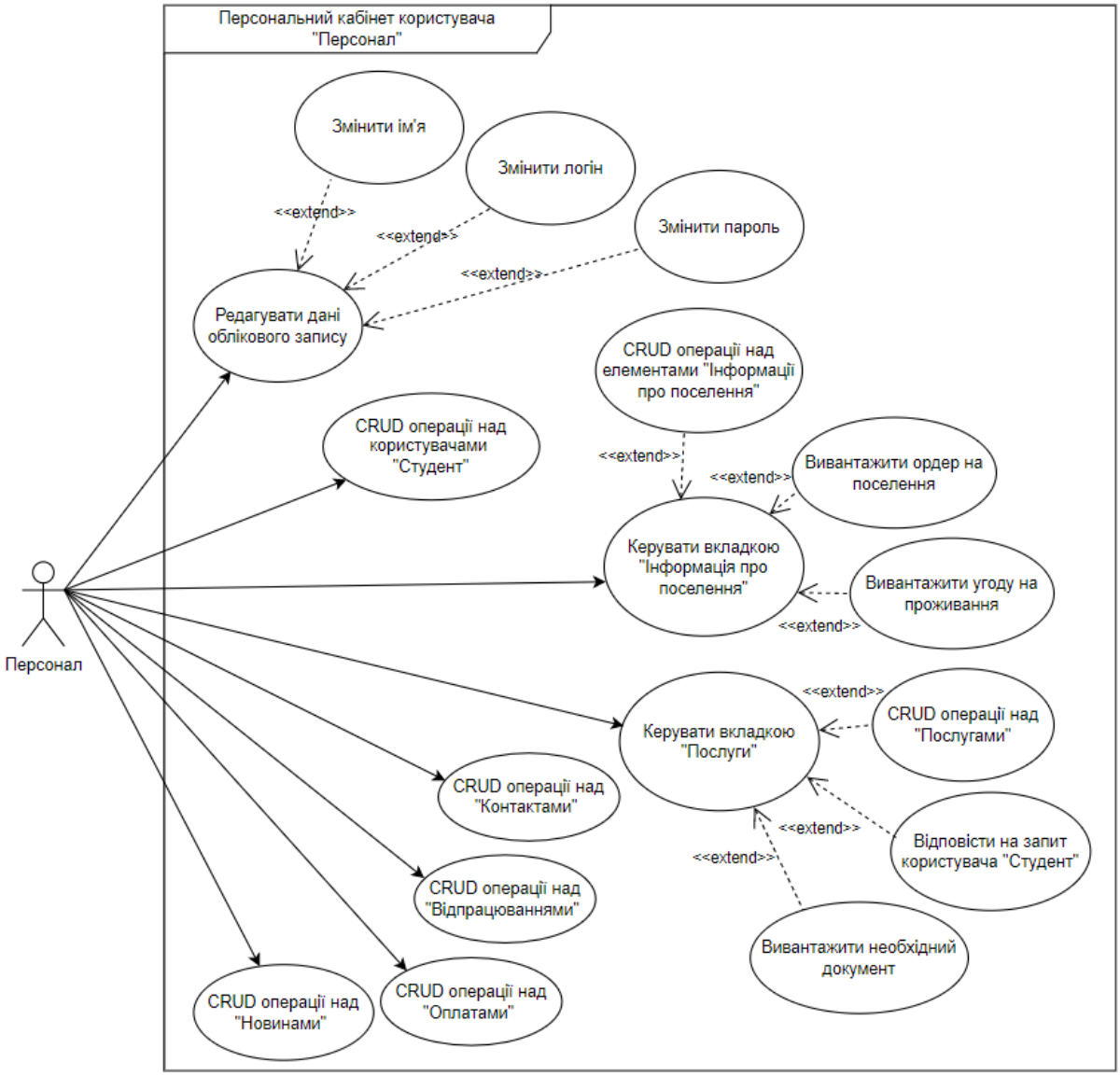

Рисунок 4.10 – Діаграма прецедентів користувача “Персонал”

Користувач “Студент” – це обліковий запис, призначений для використання студентами, які проживають або збираються проживати у гуртожитку. Студент може редагувати дані для входу в обліковий запис, переглядати стрічку новин, переглядати рахунки, виставлені адміністрацією гуртожитку та здійснювати по ним оплату, переглядати графік власних відпрацювань, переглядати контакти адміністрації гуртожитку для зворотного зв’язку, переглядати інформацію про поселення та завантажувати необхідні документи для поселення, переглядати послуги, які надає гуртожиток, та надсилати по ним запит адміністрації.

Спроєктуємо діаграму прецедентів користувача “Студент” (рисунок 4.11).

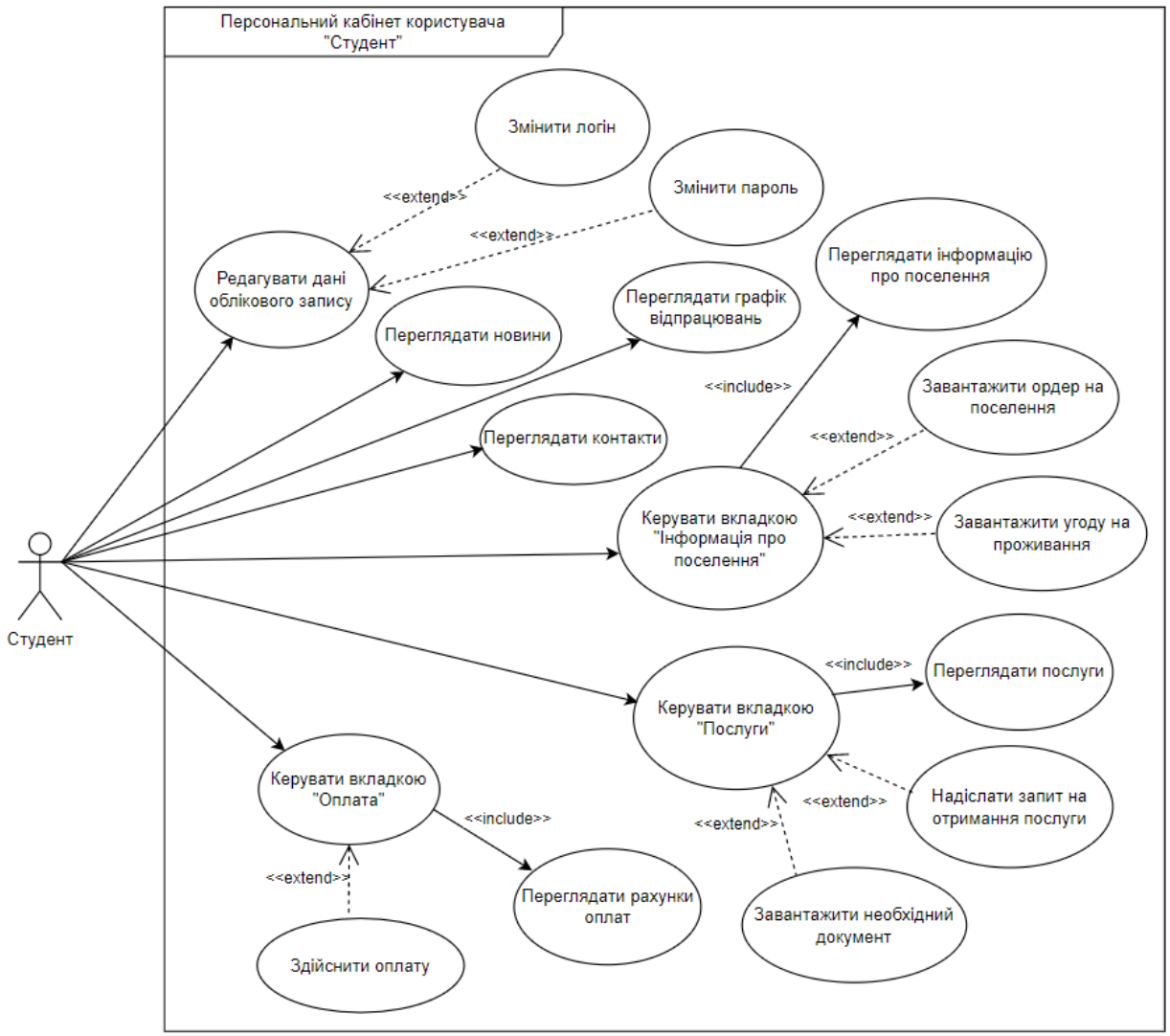

Рисунок 4.11 – Діаграма прецедентів користувача “Студент”

Характерною особливість користувача “Студент” є те, що він не може редагувати персональні дані облікового запису, а лише дані для авторизації. Це пов’язано з тим, що студенти не можуть вільно реєструватись у системі керування студмістечком. Їх облікові записи створюють користувачі-адміністратори та надсилають їм дані для входу.

4.3 Опис компонентів користувальницького інтерфейсу системи керування студмістечком

У системі керування студмістечком група із п’яти користувачів утворює ієрархічну модель ролей системи (рисунок 4.12).

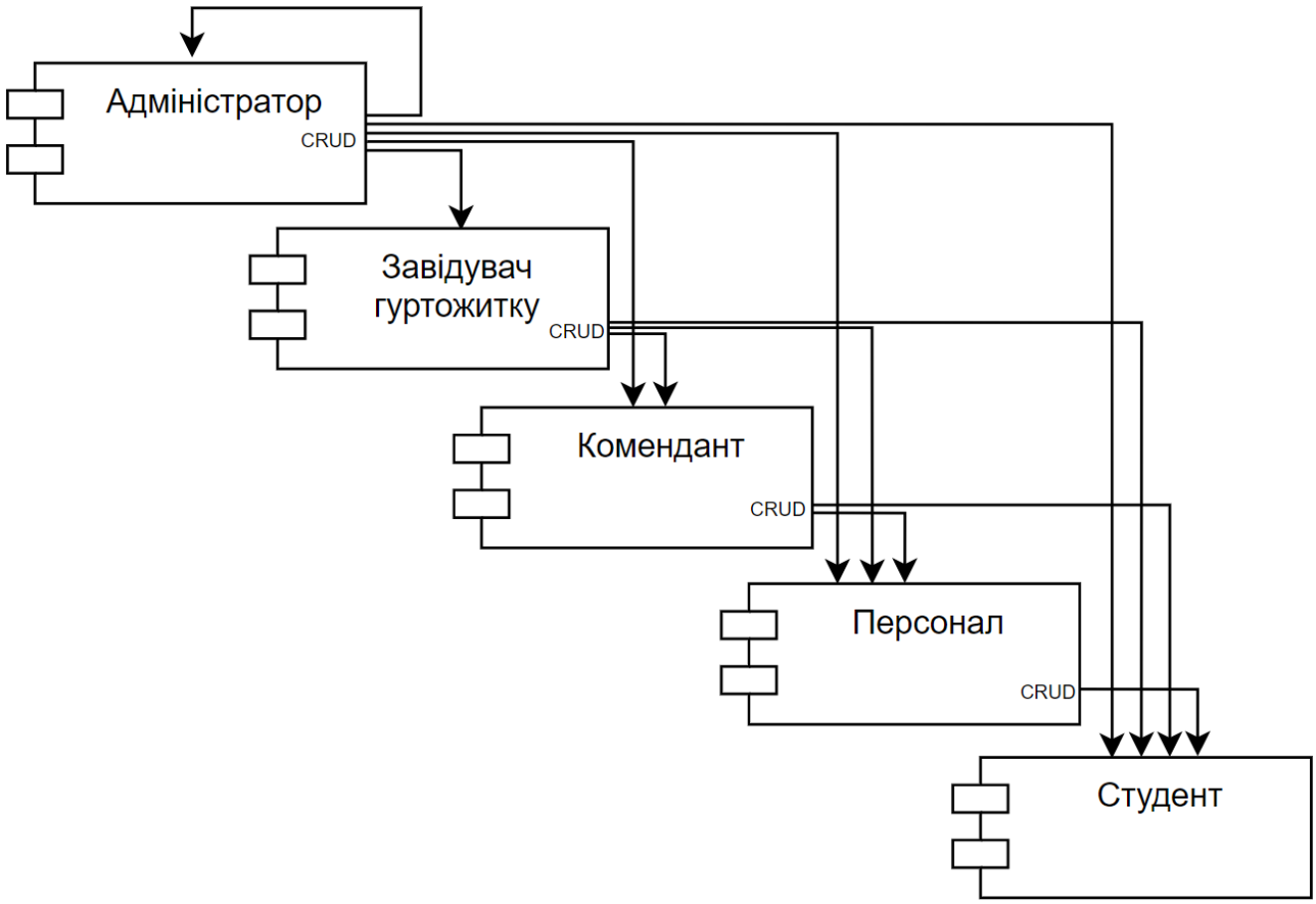

Рисунок 4.12 – Ієрархічна модель ролей системи

Роль – це функціональні можливості адміністрування системи. Головним користувачем системи керування студмістечком є користувач “Адміністратор”. У нього є права редагування усіх типів користувачів. Користувач “Завідувач гуртожитку” може редагувати лише типи користувачів, які в ієрархічній моделі ролей знаходяться після нього: користувач “Комендант”, користувач “Персонал”

та користувач “Студент”. Аналогічно, користувач “Комендант” може керувати лише персоналом та студентами, а персонал – тільки студентами.

Модуль кожного користувача складається з компонентів. Спроєктуємо схему компонентів користувача “Студент” (рисунок 4.13).

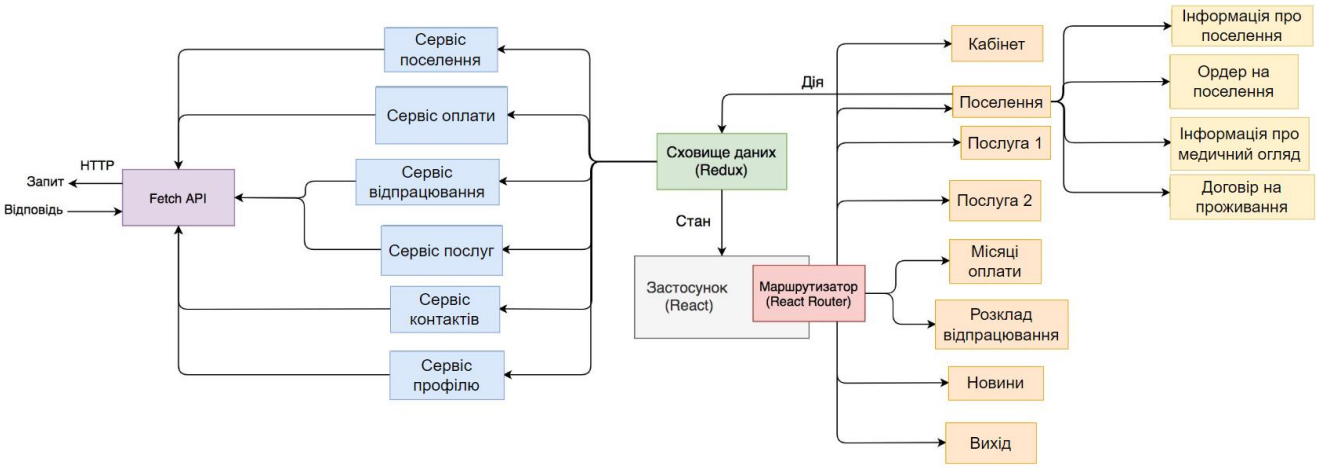

Рисунок 4.13 – Схема компонентів користувача “Студент”

В правій частині схеми знаходяться компоненти користувача “Студент”. Кожен компонент – це структурна одиниця інтерфейсу, з яким взаємодіє користувач. У персональному кабінеті користувача “Студент” є вкладки “Профіль”, “Поселення”, “Послуги” та інші. Усі ці елементи є окремими компонентами. Взаємодія користувача з елементом інтерфейсу потрапляє у сховище Redux, та після обробки дії передає стан у React застосунок.

Для кожного компоненту створений відповідний сервіс, який буде оброблювати взаємодію користувача з елементом інтерфейсу. Сервіси представлені у лівій частині схеми. Результат обробки сервісу передається у веб сервіс, який реалізує інтерфейс Fetch API та взаємодіє з Dynamics 365 API ERP системи керування студмістечком.

4.4 Висновки до розділу

Архітектура системи керування студмістечком представлена клієнт-серверною архітектурою, що дозволяє розподілити завдання та навантаження між клієнтами та серверами, які знаходяться в одній системі або об’єднані комп’ютерною мережею.

Архітектура користувальницького інтерфейсу системи керування студмістечком представлена шаблоном MVC (Model, View, Controller), де Model реалізує бізнес-логіку компонента, View відображає результат обробки у вигляді інтерфейсу користувача, а Controller обробляє взаємодію користувача з інтерфейсом. Це дозволяє структурувати програмний код на чітко визначені компоненти, кожен з яких виконує свої конкретні функції.

Спроєктовані діаграми прецедентів та схема компонентів, які відповідають вимогам до функціональних можливостей користувальницького інтерфейсу, допомогли визначити порядок реалізації функціональних можливостей, що мінімізувало кількість помилок при розробці.

5 МЕТОДИКА РОБОТИ З КОРИСТУВАЛЬНИЦЬКИМ ІНТЕРФЕЙСОМ СИСТЕМИ КЕРУВАННЯ СТУДМІСТЕЧКОМ

У даному розділі викладено характеристику взаємодії та особливості роботи з користувальницьким інтерфейсом системи керування студмістечком.

У першому підрозділі представлено мінімальні та рекомендовані системні вимоги для користування автоматизованою системою керування студмістечком.

У другому підрозділі представлено алгоритм розгортання автоматизованої системи керування студмістечком за допомогою хмарного сервісу Heroku.

У третьому підрозділі представлено основний сценарій взаємодії з користувальницьким інтерфейсом системи керування студмістечком.

Вкінці надано висновки до розділу.

5.1 Системні вимоги для користування автоматизованою системою керування студмістечком

Системними вимогами для користування автоматизованою системою керування студмістечком є системні вимоги для користування інтернет браузером.

Мінімальні системні вимоги для використання інтернет браузера з правильним відображенням JavaScript: операційні системи Windows 7, macOS 10.13, Ubuntu 10.05, Debian 6, процесор Intel Pentium 4 або аналог, 400 Мб вільної пам’яті на диску та 1 Гб оперативної пам’яті.

Рекомендовані системні вимоги для використання інтернет браузера з правильним відображенням JavaScript: операційні системи Windows 10, macOS 10.15, процесор Intel Core i3 5 покоління або аналог, 1 Гб вільної пам’яті на диску та 2 Гб оперативної пам’яті.

Рекомендується використовувати браузер Google Chrome.

5.2 Алгоритм розгортання системи керування студмістечком за допомогою хмарного сервісу Heroku

Спочатку необхідно здійснити розгортання застосунку за допомогою хмарного сервісу Heroku. Створюємо обліковий запис у сервісі (рисунок 5.1).

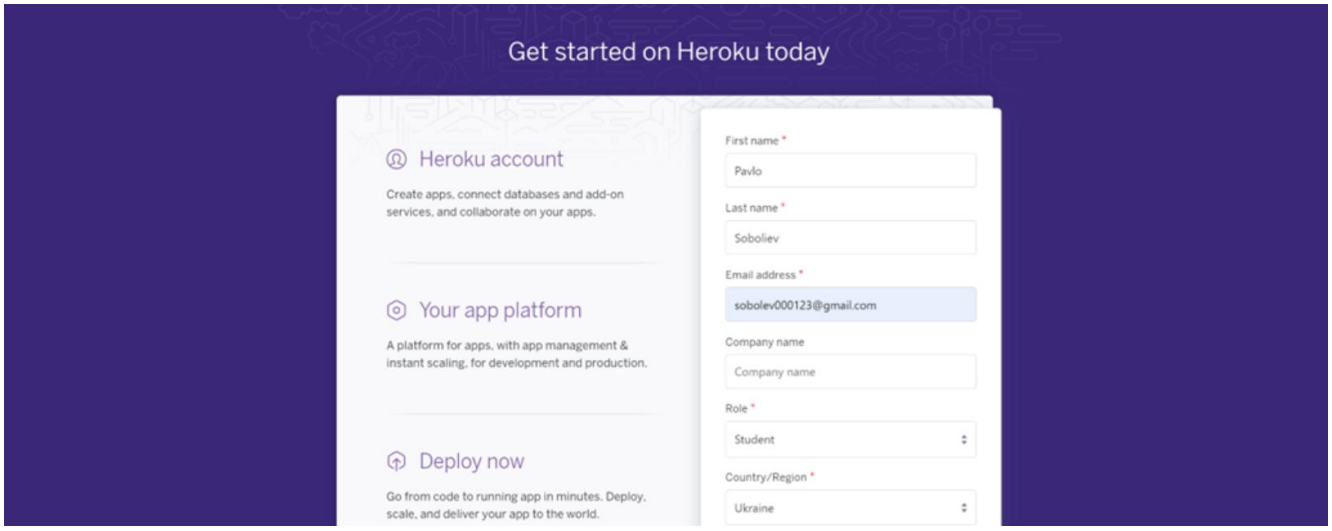

Рисунок 5.1 – Створення облікового запису в хмарному сервісі Heroku

Задаємо ім’я нового додатка та створюємо застосунок (рисунок 5.2).

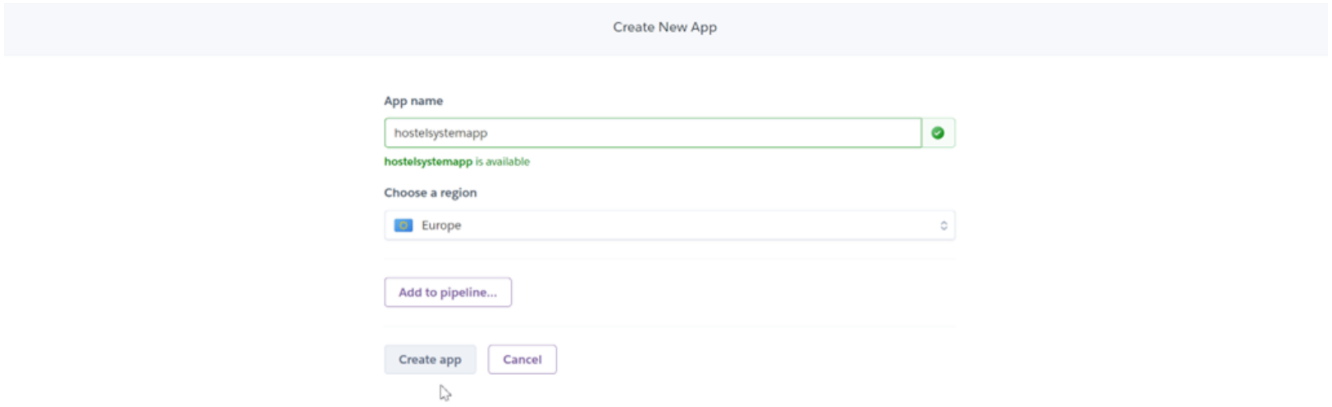

Рисунок 5.2 – Створення нового застосунку в хмарному сервісі Heroku

Після створення застосунку у хмарному сервісі Heroku нам відкривається форма налаштування додатку. Необхідно вибрати метод розгортання застосунку, прив’язати сховище Git з системою керування студмістечком та здійснити з’єднання з ним (рисунок 5.3).

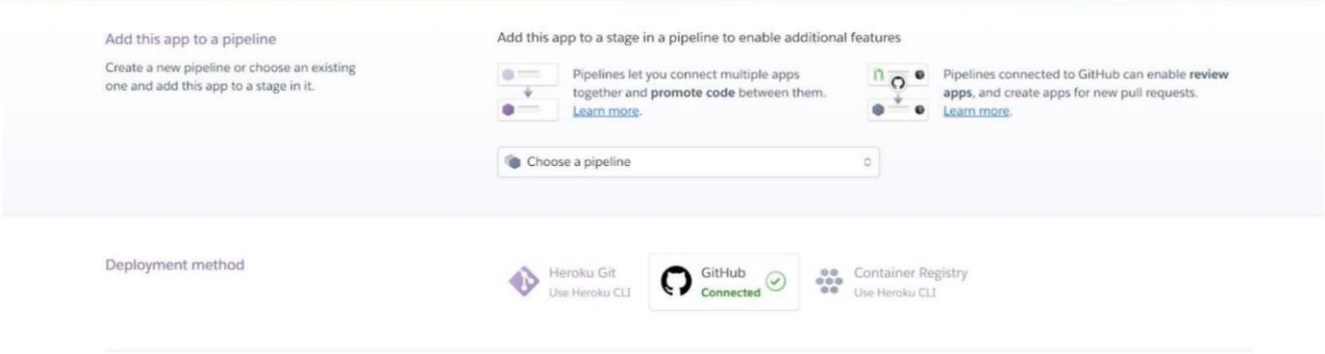

Рисунок 5.3 – З’єднання сховища Git з хмарним сервісом Heroku

Heroku також надає можливість автоматичного розгортання застосунку після внесення змін у сховище системи контролю версій Git. Обираємо основну гілку master та розгортаємо застосунок (рисунок 5.4).

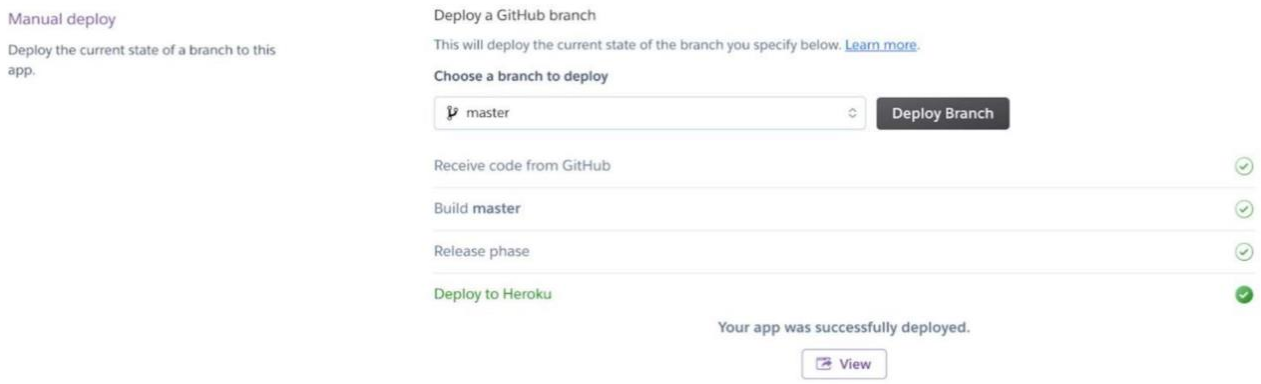

Рисунок 5.4 – Розгортання застосунку в основній гілці master

Тепер користувач може розпочати взаємодію з користувальницьким інтерфейсом системи керування студмістечком.

5.3 Взаємодія з користувальницьким інтерфейсом системи керування студмістечком

Користувачу “Студент” надається логін та тимчасовий пароль для авторизації у системі керування студмістечком (рисунок 5.5).

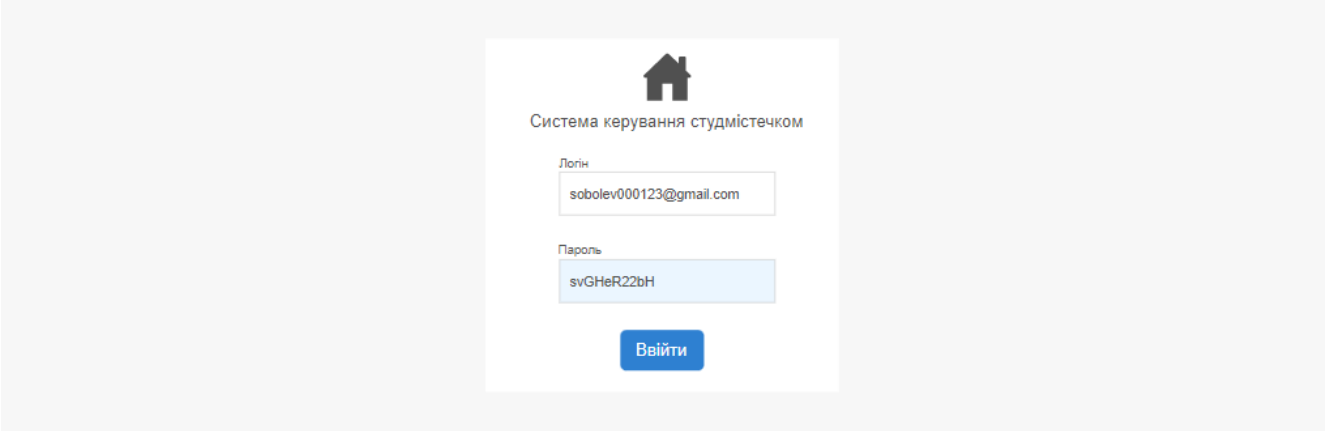

Рисунок 5.5 – Авторизація в обліковий запис користувача “Студент”

Тимчасовий пароль згенеровано адміністрацією гуртожитку автоматично, тому при першій спробі авторизації студенту необхідно змінити тимчасовий пароль на його власний (рисунок 5.6).

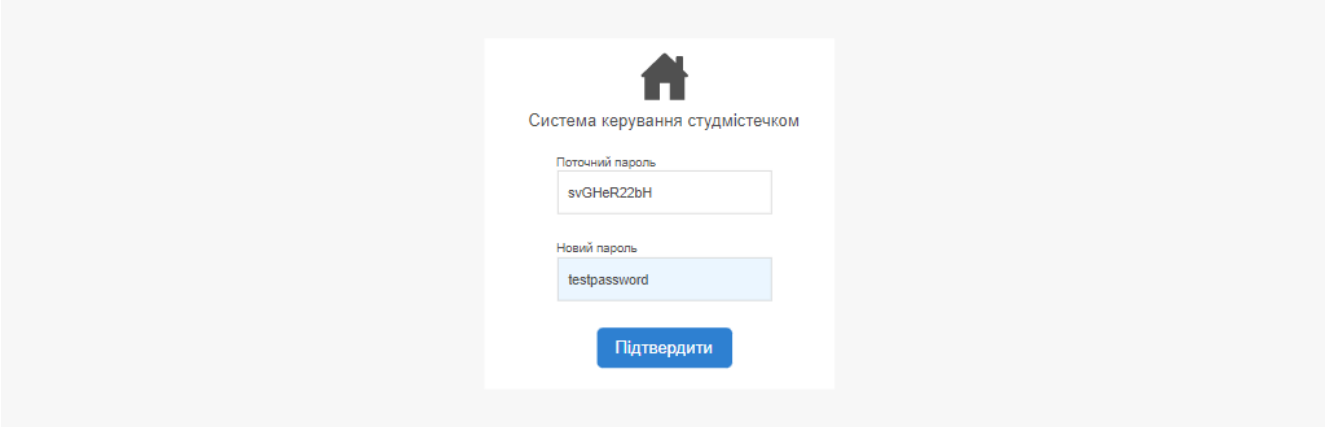

Рисунок 5.6 – Зміна тимчасового пароля

Після успішної авторизації, студент потрапляє у свій персональний кабінет. Спочатку йому надається статус “В процесі поселення” та доступ до трьох основних вкладок: “Поселення”, “Оплата” та “Контакти” (рисунок 5.7).

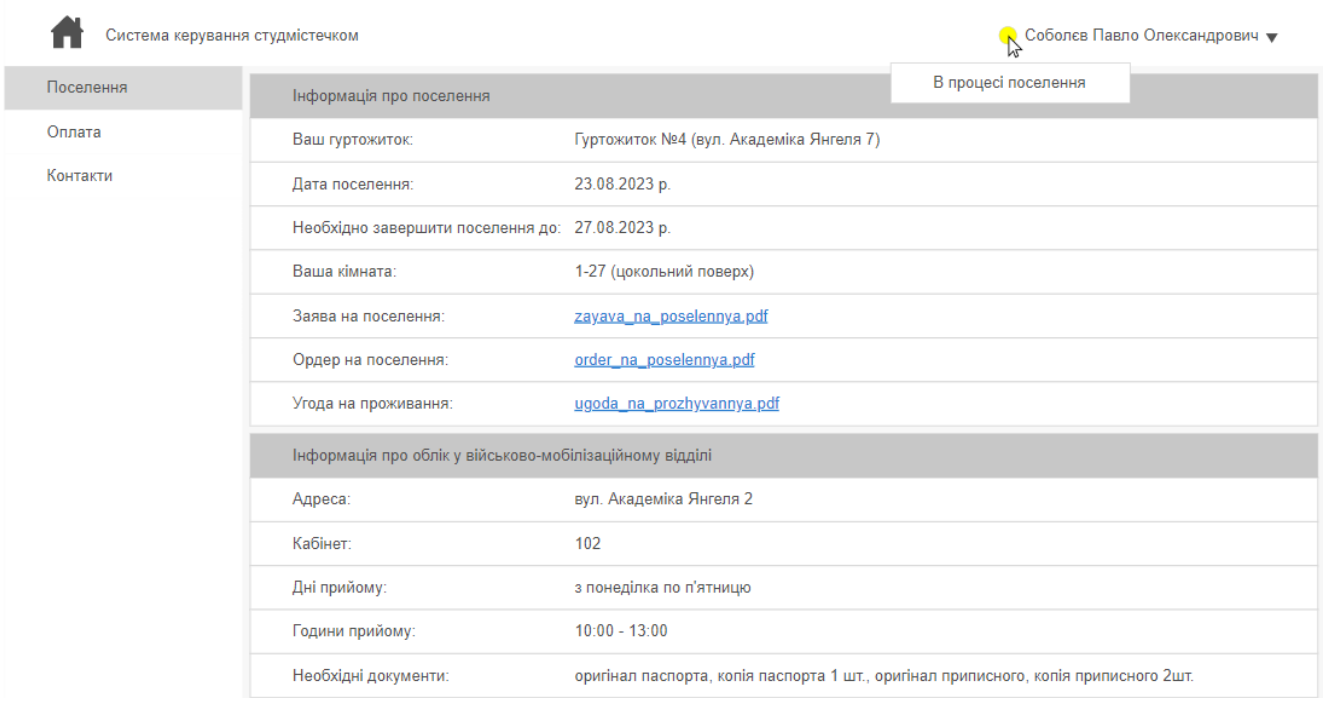

Рисунок 5.7 – Персональний кабінет користувача “Студент” зі статусом “В процесі поселення”

На вкладці “Поселення” знаходиться уся необхідна інформація про поселення: інформація про гуртожиток, дата початку та кінця процесу поселення, інформація про кімнату, заява на поселення, ордер на поселення, угода на проживання, інформація про те, як встати на військовий облік та пройти медичний огляд.

Згідно з порядком поселення студентів, щоб завершити процес поселення, вступнику необхідно встати на військовий облік та пройти медичний огляд згідно з інформацією на вкладці “Поселення”. Після того, як студент виконає усі кроки, йому потрібно перейти на вкладку “Оплата” та здійснити оплату мінімум за два місяці проживання (рисунок 5.8).

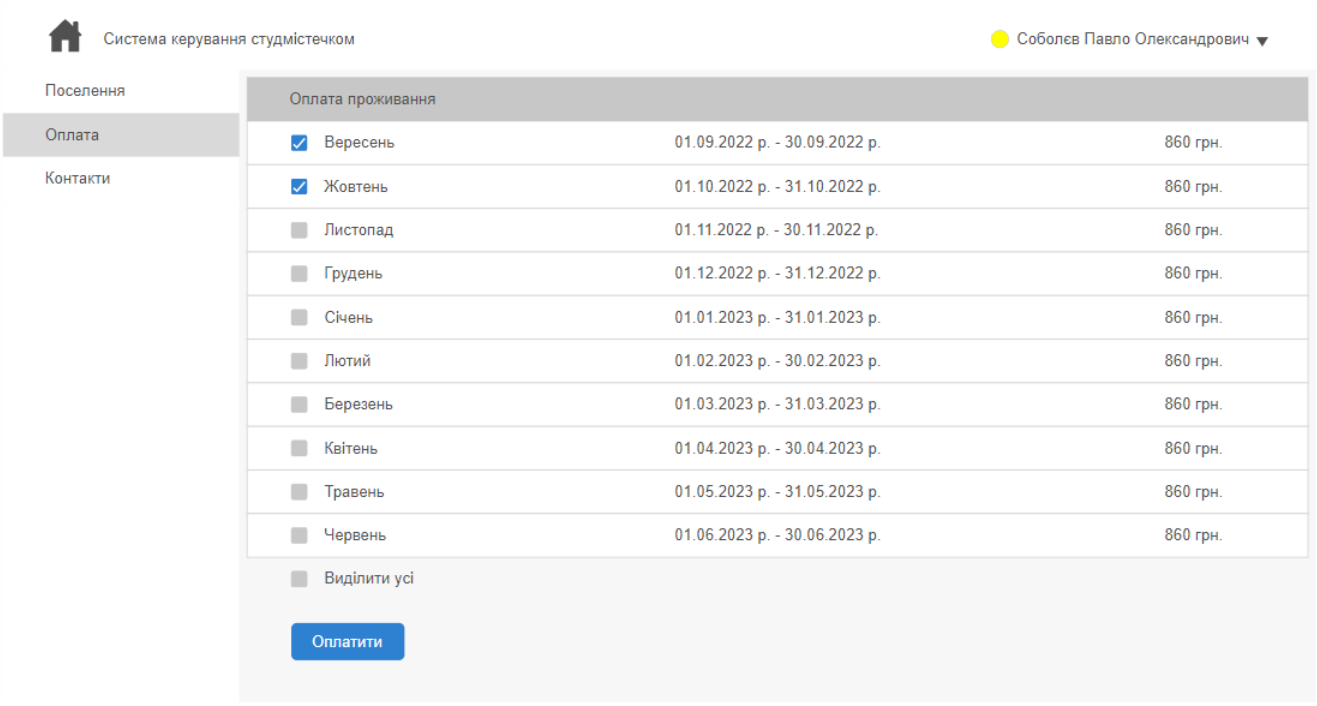

Рисунок 5.8 – Вкладка оплати проживання

Після завершення процесу поселення студенту надається статус “Проживає” та доступ до вкладок “Новини”, “Відпрацювання” та “Послуги” (рисунок 5.9).

Рисунок 5.9 – Персональний кабінет користувача “Студент” зі статусом “Проживає”

На вкладці “Відпрацювання” представлено таблицю з інформацією про години відпрацювання студента за кожен місяць (рисунок 5.10).

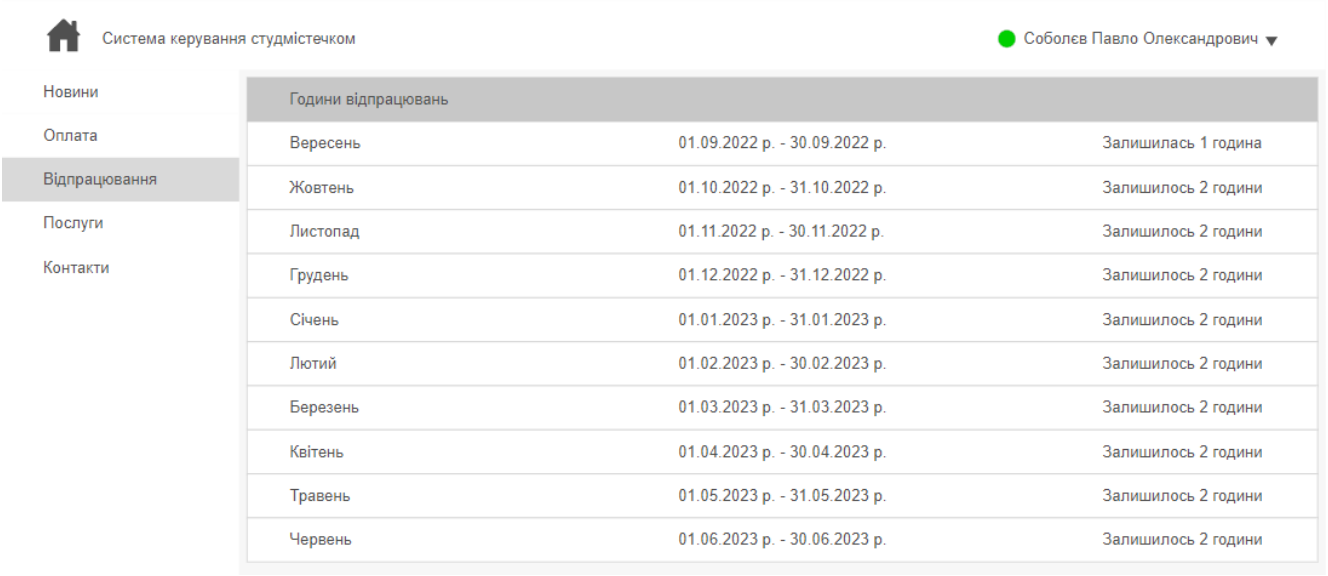

Рисунок 5.10 – Вкладка “Відпрацювання”

На вкладці “Послуги” представлено таблицю послуг, які надає гуртожиток (рисунок 5.11).

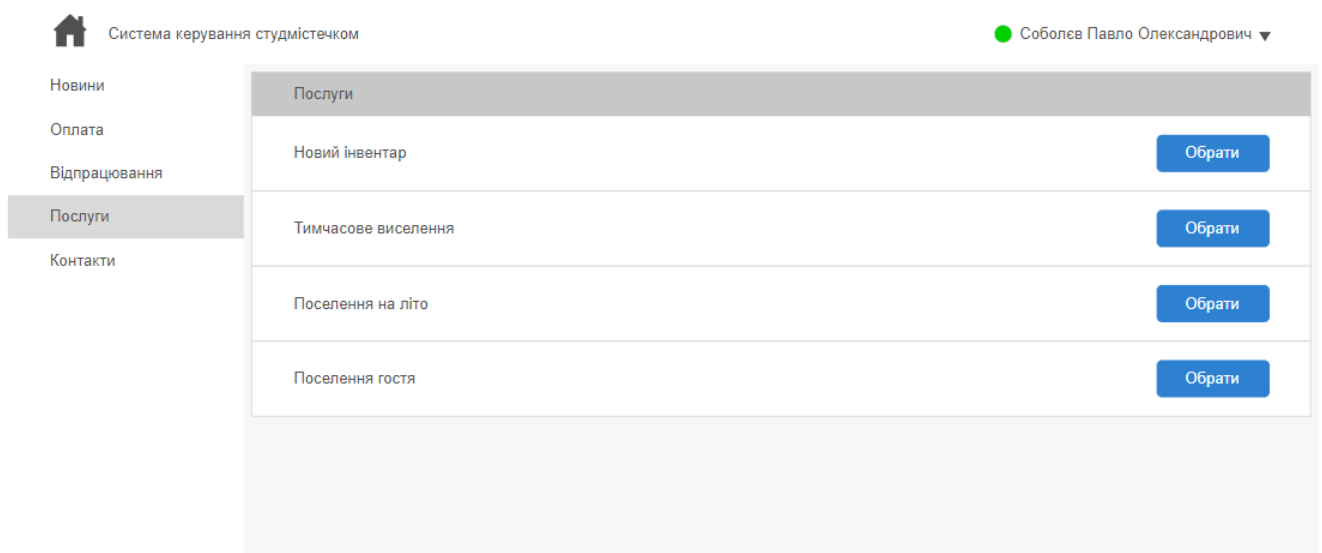

Рисунок 5.11 – Вкладка “Послуги”

Користувач “Студент” може надіслати запит адміністрації гуртожитку на отримання послуги. Обираємо послугу “Новий інвентар” (рисунок 5.12).

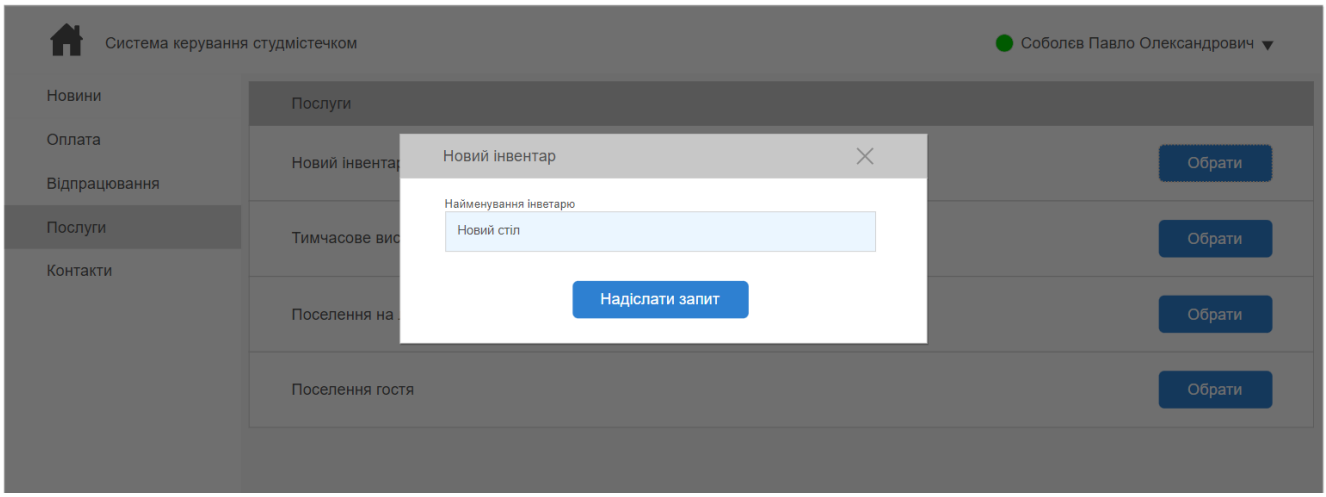

Рисунок 5.12 – Модальне вікно створення запиту на отримання послуги “Новий інвентар”

Створені запити будуть представлені у вигляді таблиці, яка буде з’являтись перед таблицею з послугами (рисунок 5.13).

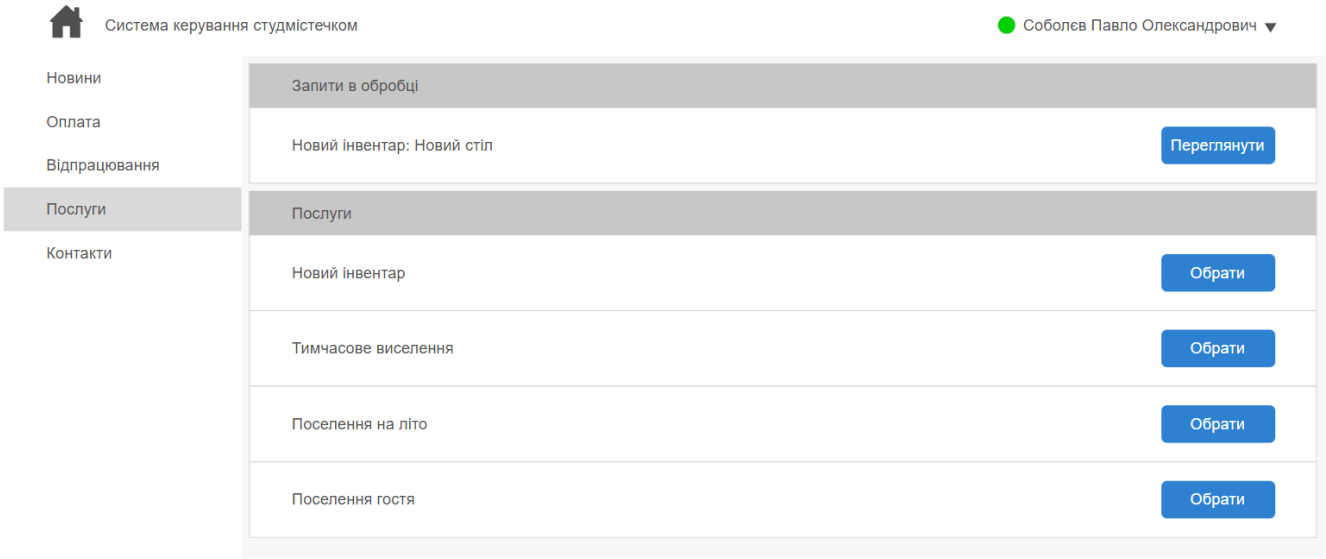

Рисунок 5.13 – Вкладка “Послуги” після створення запиту на отримання послуги “Новий інвентар”

5.4 Висновки до розділу

Визначені системні вимоги для користування системою керування студмістечком являються системними вимогами для використання інтернет браузера, тому у типового користувача не повинно бути проблем зі швидкодією застосунку.

Розроблена інструкція розгортання додатка з використанням хмарного сервісу Heroku допоможе користувачу швидко запустити систему керування студмістечком на хмарному сервері та розпочати роботу з нею.

Опис взаємодії з користувальницьким інтерфейсом системи керування студмістечком демонструє, що застосунок відповідає вимогам до функціональних можливостей системи, показує, що було досягнуто мету магістерської дисертації, та допомагає користувачу працювати з клієнтським застосунком.

6 РОЗРОБЛЕННЯ СТАРТАП-ПРОЄКТУ

У даному розділі викладено характеристику процесу розроблення стартап-проєкту для впровадження системи керування студмістечком.

У першому підрозділі представлено опис ідеї стартап-проєкту для системи керування студмістечком, визначення сильних та слабких сторін ідеї проєкту.

У другому підрозділі представлено характеристику технологічного аудиту ідеї проєкту.

У третьому підрозділі представлено аналіз ринкових можливостей запуску стартап-проєкту, характеристику потенційного ринку та потенційних споживачів.

У четвертому підрозділі представлено опис розроблення ринкової стратегії стартап-проєкту, вибір цільових груп потенційних споживачів, визначення стратегії розвитку проєкту.

У п’ятому підрозділі представлено опис розроблення маркетингової стратегії, визначення ключових переваг концепції потенційного товару, характеристику трирівневої маркетингової моделі.

Вкінці надано висновки до розділу.

6.1 Опис ідеї проєкту

Для опису ідеї стартапу необхідно проаналізувати зміст ідеї, можливі напрямки застосування продукту-результату реалізації ідеї, основні вигоди, що може отримати користувач продукту (за кожним напрямком застосування), відмінності від існуючих на ринку аналогів та замінників.

Аналіз цих чинників зручно проводити у табличному вигляді. Зміст ідеї, можливі напрямки застосування, основні переваги, які зможе отримати користувач, представлено у таблиці 6.1.

Таблиця 6.1.

Опис ідеї стартап-проєкту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Веб орієнтоване програмне забезпечення для керування студмістечком	Вирішення задач керування студмістечком: автоматизація поселення студентів першого курсу, автоматизація поселення студентів старших курсів, автоматизація процесів керування гуртожитком	Дослідження наявних процесів керування студмістечком та гуртожитком
	Навчальний процес	Демонстрація та ознайомлення студентів, що вивчають методи автоматизації складних процесів та систем. Приклади вирішення певних задач
	Науково-дослідний процес	Збереження та аналіз отриманих результатів для покращення наявних чи розробки принципово нових методів автоматизації процесів керування студмістечка.

Моя мета – це реалізація користувальницького інтерфейсу системи керування студмістечком, але для аналізу ідеї стартап-проєкту доцільніше надавати характеристику усієї системи керування студмістечком.

Визначення технічно-економічних характеристик та особливостей ідеї, їх аналіз порівняно з програмними продуктами-конкурентами наведено у таблиці 6.2.

Таблиця 6.2.

Визначення сильних, слабких та нейтральних характеристик ідеї проєкту

№	Техніко-економічні характеристики	(потенційні) товари/концепції конкурентів		
		Мій проєкт	Конкурент 1	Конкурент 2
1	Швидкість розрахунків	Нижча відносно конкурентів швидкість розрахунків через веб орієнтованість системи.	Краща швидкість розрахунків через використання низькорівневої мови програмування та відсутність графічного інтерфейсу	Краща швидкість розрахунків через використання низькорівневої мови програмування та відсутність графічного інтерфейсу
2	Точність отриманих результатів	Краща точність отриманих результатів	Середня точність отриманих результатів	Середня точність отриманих результатів
3	Доступ до програмного забезпечення	Не потребує встановлення, доступний у веб браузері	Потребує встановлення на машину	Потребує встановлення на машину
4	Зручність у використанні	Має інтуїтивно зрозумілий графічний інтерфейс	Не має графічного інтерфейсу	Не має графічного інтерфейсу
5	Ціна за підписку	Низька	Безплатний, але потребує платного ПЗ для перегляду результатів	Безплатний, але потребує платного ПЗ для перегляду результатів

Сильними сторонами системи відносно конкурентів є точність отриманих результатів, наявність графічного користувальницького інтерфейсу, гнучкість в доступі до системи. До слабких сторін можна віднести швидкість розрахунків.

6.2 Технологічний аудит ідеї проєкту

Технологічний аудит проєкту проводиться для огляду та аналізу засобів та технологій, за допомогою яких можна реалізувати ідею проєкту. Зокрема перед реалізацією ідеї необхідно визначити, які технології використовувати при реалізації, чи вони наявні та чи доступні розробникам для реалізації ідеї. Аналіз технологічної здійсненності проєкту наведено в таблиці 6.3.

Таблиця 6.3.

Технологічна здійсненність ідеї проєкту

№ п/п	Ідея проєкту	Технології реалізації	Наявність технологій	Доступність технологій
1	Веб інтерфейс користувача	HTML + CSS + JS	Наявна	Доступна, безплатна
2	Веб сервер	Heroku	Наявна	Доступна, платна
3	Реалізація розрахункових алгоритмів	JS	Наявна	Доступна, безплатна
4	Серверна частина	Node.js	Наявна	Доступна, безплатна

Отже, проєкт реалізувати можливо. Обрана технологія реалізації проєкту: HTML+CSS+JS для веб інтерфейсу; Heroku в якості веб серверу; JS для реалізації розрахункових алгоритмів; Node.js для побудови серверної частини застосунку.

6.3 Аналіз ринкових можливостей запуску стартап-проєкту

Важливою складовою розроблення стартап-проєкту є аналіз ринку та можливості запуску проєкту на ринку. Необхідно визначити та спланувати напрямки розвитку проєкту з урахуванням поточного стану ринку, запитів та потреби потенційних клієнтів і пропозицій виробників товарів конкурентів.

У таблиці 6.4 представлено аналіз наявності конкурентів, динаміки розвитку ринку, наявності попиту, обсягу продаж, наявність специфічних вимог для виходу на ринок.

Таблиця 6.4.

Попередня характеристика потенційного ринку стартап-проєкту

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од.	5
2	Загальний обсяг продаж, грн/ум. од.	270 грн/місяць
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Відсутні
5	Специфічні вимоги до стандартизації та сертифікації	Відсутні
6	Середня норма рентабельності в галузі (або по ринку), %	10%

Ринок є привабливий для входу, адже має невелику кількість основних гравців і має потребу в оновлених програмних засобах з графічним інтерфейсом для автоматизації процесів керування студмістечком.

У таблиці 6.5 проаналізовано потенційні групи клієнтів-користувачів програмного продукту з характеристикою та попередній список вимог до необхідних характеристик проєкту для кожної групи.

Таблиця 6.5.

Характеристика потенційних споживачів стартап-проєкту

Потреба, що формує ринок	Цільова аудиторія	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
Автоматизація процесів керування студмістечком	Дослідники, науковці	Використання для автоматизації процесів керування гуртожитком	Точність та швидкість розрахунків, можливість зберігати та аналізувати отримані результати, можливість завантаження даних з файлу
	Студенти	Використання для вивчення способів автоматизації, принципів проєктування	Зручність у роботі з додатком
	Викладачі	Викладання матеріалу для курсу автоматизації процесів, принципи проєктування	Точність результатів, зручність у роботі з додатком, можливість завантаження даних з файлу

Для розуміння можливості запуску проєкту на ринку слід проаналізувати фактори, що перешкоджають впровадженню проєкту (таблиця 6.6).

Таблиця 6.6.

Фактори загроз впровадження проєкту

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Збільшення витрат на розробку та впровадження	Збільшення витрат на розробку та впровадження програмного забезпечення, підтримку клієнтів	Збільшення вартості підписки, перехід на технології, що пришвидшать розробку
2	Поява нових конкурентних продуктів	Поява нових програмних продуктів на ринку, що можуть створювати конкуренцію і переманювати клієнтів	Удосконалення продукту, покращення маркетингу, програми лояльності

Серед зазначених загроз немає таких, які могли б унеможливити запуск продукту та його розвиток із залученням нових споживачів.

Для розуміння можливості запуску проєкту на ринку слід також проаналізувати фактори, що допоможуть впровадити проєкт (таблиця 6.7).

Таблиця 6.7.

Фактори можливостей впровадження проєкту

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії

Продовження таблиці 6.7

1	Збільшення попиту на програмний продукт	Можливість використовувати сучасне програмне забезпечення для автоматизації процесів керування студмістечком породжує попит на програмний продукт	Розширення системи, удосконалення існуючих функціональних можливостей та додавання нових.
2	Вихід конкурентів з ринку	Припинення постачання/оновлення програмного забезпечення конкурентами	Збільшення клієнтської бази шляхом реклами, впровадженням безкоштовного тестового періоду

Потенційно на ринку можуть виникнути можливості для розвитку програмного продукту, заохочення нових клієнтів і як наслідок збільшення прибутку.

Загальні риси конкуренції на ринку представлені в таблиці 6.8.

Таблиця 6.8.

Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (потенційні дії компанії, щоб залишатись конкурентною)
1.Тип конкуренції – олігополія	Домінує невелика кількість конкуруючих компаній	Якісний маркетинговий підхід, розробка кращого за всіма параметрами продукту ніж в конкурентів

Продовження таблиці 6.8

2. За рівнем конкурентної боротьби: міжнародне конкурентне середовище	Боротьба за клієнтів ведеться в межах багатьох країн	Локалізація інтерфейсу користувача, якісний маркетинговий підхід
3. Внутрішньогалузева – за галузевою ознакою	Продукт використовується в сфері проживання в гуртожитках	Розробка унікальних для ринку функціональних можливостей
4. Конкуренція за видами товарів: – товарно-родова	Конкуренція між різними системами для автоматизації процесів керування студмістечком	Розробка унікальних для ринку функціональних можливостей
5. За характером конкурентних переваг – не цінова	Конкурентні переваги продукту ґрунтуються на його функціональних можливостях	Впровадження нових та покращення існуючих функціональних можливостей
6. За інтенсивністю – не марочна	Споживачі звертають увагу не на бренд, а на функціональні можливості	Створення продукту, що задовольняє потреби споживачів

Для більш детального аналізу конкуренції в галузі і вироблення стратегії розвитку бізнесу можна використовувати модель Майкла Портера, яку ще називають «Аналізом п'яти сил». До п'яти сил Портера належать наступні компоненти: прямі конкуренти в галузі, потенційні конкуренти, постачальники (п-ки), клієнти та товари-замінники.

Аналіз конкуренції в галузі за М. Портером наведено в таблиці 6.9.

Таблиця 6.9.

Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	П-ки	Клієнти	Товар-замінник
	Конкуренти можуть покращувати власні продукти та проводити рекламну кампанію	Наявні	—	Продукт орієнтований на бажання та вимоги клієнтів.	—
Висновки	Наразі конкуренти не оновлюють продукти	Присутні. Вихід на ринок можливий	П-ки відсутні	Умови ринку диктуються клієнтами, їх вимогами до продукту	Товари-замінники не обмежують роботу на ринку

Отже, робота на ринку можлива, товари-замінники відсутні, прямі конкуренти не вдосконалюють власні продукти, клієнти диктують умови та орієнтуються на функціональні можливості продукту.

На основі аналізу конкуренції, а також із урахуванням характеристик ідеї проєкту, вимог споживачів до товару та факторів маркетингового середовища визначено та обґрунтовано фактори конкурентоспроможності, які наведені у таблиці 6.10.

Таблиця 6.10.

Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проєктів значущим)
1	Швидкість розрахунків	Швидкість роботи є досить значним фактором, проте не вирішальним. Користувач не відчує різницю в секунду при вирішенні певної задачі
2	Зручність у використанні	Користувацький інтерфейс та зручність у використанні є важливим фактором в клієнт-орієнтованих програмних продуктах
3	Точність отриманих результатів	Вона відображається в ефективності автоматизації процесів керування студмістечком
4	Повторне використання даних	Повторне використання даних дозволяє виконувати моделювання певної задачі оминаючи процес заповнення форми вхідними даними
5	Засоби для візуалізації отриманих результатів	Представлення результатів дослідження не тільки у текстовому вигляді, а й у формі користувальницького інтерфейсу робить аналіз результатів ефективнішим

За визначеними факторами конкурентоспроможності проведений аналіз сильних та слабких сторін стартап-проєкту, який представлений в таблиці 6.11, де П – стартап-проєкт, а К – конкурент.

Таблиця 6.11

Порівняльний аналіз сильних та слабких сторін проєкту

№ п/п	Фактор конкурентоспроможності	Бал	Рейтинг товарів-конкурентів						
			-3	-2	-1	0	+1	+2	+3
1	Швидкість аналізу	20				П	К2	К1	
2	Зручність у використанні	19			К1	К2			П
3	Точність отриманих результатів	18				К1	К2	П	
4	Повторне використання даних	17				К1	К2		П
5	Засоби для візуалізації отриманих результатів	15			К2	К1			П

Одним із методів стратегічного планування є SWOT-аналіз (таблиця 6.12), який дозволяє провести детальне дослідження зовнішнього та внутрішнього середовища і передбачає розділення чинників і явищ на чотири категорії: сильні сторони проєкту (Strengths), слабкі сторони проєкту (Weaknesses), можливості при реалізації проєкту (Opportunities), загрози при здійсненні проєкту (Threats).

Таблиця 6.12.

SWOT- аналіз стартап-проєкту

Сильні сторони: точність отриманих розрахунків, доступ до системи, повторне використання даних, засоби для аналізу отриманих результатів	Слабкі сторони: швидкість розрахунків
Можливості: збільшення попиту на товар, розробка нових модулів	Загрози: нові конкуренти на ринку, збільшення ціни на розробку

За результатами SWOT-аналізу розроблено альтернативні варіанти ринкової поведінки (таблиця 6.13) для запуску стартап-проєкту на ринку та приблизний оптимальний час їх ринкової реалізації з огляду на потенційно можливі нові продукти конкурентів, що можуть бути виведені на ринок.

Таблиця 6.13.

Альтернативи ринкового впровадження стартап-проєкту

№ п/п	Альтернатива ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Розробка версії продукту з реалізацією запланованих функціональних можливостей. Продукт буде повністю готовим до використання	висока	До 2 місяців
2	Розробка версії продукту з підтримкою нових функціональних можливостей	середня	До 5 місяців
3	Створення мобільної версії програмного забезпечення	низька	До 3 місяців
4	Побудова комплексної системи для керування студмістечком	середня	До 12 місяців

Обрана альтернатива під номером один, оскільки її реалізація є найбільш вірогідна та потребує найменше часу на розробку

6.4 Розроблення ринкової стратегії проєкту

Ринкова стратегія – це вибір цільових ринків і розрахунок сценаріїв виходу і присутності на цих ринках. На основі ринкової стратегії розробляються фінансова, маркетингова, інформаційна стратегії, стратегія щодо розвитку персоналу і стратегія виробництва, якщо таке є [15]. Для поточної ідеї стартап-проєкту не потрібно продумувати стратегію по розвитку персоналу та стратегію виробництва.

Для розроблення ринкової стратегії просування продукту на ринку необхідно перш за все визначити цільові групи споживачів, проаналізувати потенційний попит кожної з обраних групи споживачів. Вибір цільових груп потенційних споживачів наведено в таблиці 6.14.

Таблиця 6.14.

Вибір цільових груп потенційних споживачів

Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
Вищі навчальні заклади, науково-дослідні інститути та лабораторії, що займаються автоматизацією фізичних процесів	Споживачі готові сприйняти продукт	Середній попит в межах цільового сегмента	Конкуренція з товарами-конкурентами	Продукт легко впровадити на ринок, адже клієнти потребують його

У зв’язку з тим, що існує лише один цільовий сегмент, в якості основної стратегії обирається стратегія концентрованого маркетингу.

Визначення базової стратегії розвитку наведено в таблиці 6.15.

Таблиця 6.15.

Визначення базової стратегії розвитку

Обрана альтернатива розвитку проєкту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до альтернативи	Базова стратегія розвитку
Розробка базової версії продукту з подальшим розширенням та додаванням нових модулів	Стратегія розширення первинного попиту	Поточні конкуренти не займаються оновленням та розширенням своїх програмних продуктів	Стратегія диференціації

Визначення базової стратегії конкурентної поведінки представлено в таблиці 6.16.

Таблиця 6.16.

Визначення базової стратегії конкурентної поведінки

Чи є проєкт «першопроходцем» на ринку?	Чи буде компанія шукати нових споживачів або забирати наявних у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента і які?	Стратегія конкурентної поведінки
Ні	Шукати нових та переманювати клієнтів у конкурентів шляхом реклами та надання пробного періоду	Не буде	Стратегія лідера

На основі вимог споживачів з обраних сегментів до постачальника та до продукту, а також в залежності від обраної базової стратегії розвитку та стратегії конкурентної поведінки розроблена стратегія позиціювання та представлена в таблиці 6.17.

Таблиця 6.17.

Визначення стратегії позиціювання

Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проєкту	Вибір асоціацій, які мають сформувати комплексну позицію власного проєкту (три ключових)
Точність розрахунків, зручність у використанні та доступ до ПЗ	Стратегія диференціації	Точність, зручність у використанні	Точність, зручність, сучасність

6.5 Розроблення маркетингової програми стартап-проєкту

Ключові переваги концепції потенційного товару описано в таблиці 6.18.

Таблиця 6.18.

Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)

Продовження таблиці 6.18

1	Швидкодія	Достатньо швидкий процес обчислення	Необхідно пришвидшити процес обчислень за рахунок паралельних обчислень
2	Точність	Точність за рахунок використання сучасних засобів розробки та удосконалених алгоритмів	Система дає більш точні результати у порівнянні з конкурентами
3	Зручність у використанні	Зручний, інтуїтивно зрозумілий інтерфейс користувача	Конкуренти не мають графічного інтерфейсу та потребують встановлення платних програм для перегляду результатів
4	Доступ до системи	Система реалізована у вигляді веб застосунку, доступ з веб браузеру майже будь-якого пристрою	Програмні продукти конкурентів вимагають встановлення та мають досить складний процес розгортання

Трирівнева маркетингова модель товару представлена в таблиці 6.19.

Таблиця 6.19.

Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові
I. Товар за задумом	Програмне забезпечення зі зручним інтерфейсом користувача для керування студмістечком Достатня швидкість аналізу даних Зручність у використанні Простота доступу до продукту

Продовження таблиці 6.19

II. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх /Тл/Е/Ор
	1. Швидкодія	М	Тх/Тл
	2. Висока точність результатів	М	Тх/Тл
	3. Легкий доступ до продукту	М	Тх/Тл
	4. Зручність у використанні	М	Вр/Тл
III. Товар у перспективі	Якість: відповідає нормам розробки програмного забезпечення.		
	Пакування: продукт представлений у вигляді веб застосунку.На веб сайті назва продукту, інформація про розробника, опис функціональних можливостей продукту, контакти для підтримки, інструкція по використанню програмного забезпечення, ціна підписки, план розширення та доповнення продукту		

Проєкт буде захищений як інтелектуальна власність. Наступним кроком є визначення цінових меж, якими необхідно керуватись при встановленні ціни на потенційний товар. Результат представлений в таблиці 6.20.

Таблиця 6.20.

Визначення меж встановлення ціни

Рівень цін на товари-конкуренти	Рівень цін на товари-замінники	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар

Продовження таблиці 6.20

Безкоштовний, проте необхідне платне програмне забезпечення для повноцінного функціонування програмного продукту	—	600 тис. грн	150-300 грн/місяць (підписка). Наявність безкоштовного пробного періоду

Формування системи збуту (специфіка закупівельної поведінки цільових клієнтів, функція, глибина каналу та систему збуту) наведено в таблиці 6.21.

Таблиця 6.21.

Формування системи збуту

Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
Реєстрація у системі, щомісячна оплата підписки за доступ до системи	Надання доступу до системи після оплати	Виробник - споживач	Збут через купівлю підписки на веб сайті системи

Останньою складової маркетингової програми є розроблення концепції маркетингових комунікацій, що спирається на попередньо обрану основу для позиціювання, визначену специфіку поведінки клієнтів. Результат представлений в таблиці 6.22.

Таблиця 6.22.

Концепція маркетингових комунікацій

Специфіка поведінки цільових клієнтів	Канали комунікацій цільових клієнтів	Ключові позиції, обрані для позиціювання	Завдання рекламного повідомлення	Концепція рекламного звернення
Відповідально обирають систему	Телефон, електронна пошта, месенджери	Підтримка клієнтів, оновлення та доповнення продукту	Показати та донести до клієнта переваги програмного продукту	Автоматизація процесів керування студмістечком

6.6 Висновки до розділу

Аналіз ринкових можливостей запуску стартап-проєкту демонструє, що ринкова комерціалізація проєкту з системою керування студмістечком є можливою та доцільною.

Розроблення ринкової стратегії проєкту дало зрозуміти, що існують перспективи впровадження системи керування студмістечком завдяки високій конкурентоспроможності застосунку, низькому стану конкуренції та наявності потенційних груп клієнтів.

Подальша імплементація проєкту з системою керування студмістечком є доцільною. Можна виділити основні шляхи розвитку проєкту: розширення функціональних можливостей системи, створення мобільного додатку та реалізація системи керування готельною структурою на основі системи керування студмістечком.

ВИСНОВКИ

Наявні процеси керування студмістечком є неефективними та неоптимізованими. Внаслідок аналізу цих процесів було виявлено ряд проблем, які допомогли визначити вимоги до основних функціональних можливостей системи керування студмістечком.

На даний момент у гуртожитках використовують примітивні CRM системи, які надають можливість вести облік студентів та інформації, пов’язаної з ними. Аналіз такого способу керування студмістечком дав зрозуміти, що ефективніше буде використовувати ERP систему на платформі Microsoft Dynamics 365 Business Central з розширеними функціональними можливостями, що дозволить автоматизувати більшу кількість процесів керування гуртожитком. Щоб якісно користуватись усіма можливостями ERP системи необхідно розробити користувальницький інтерфейс та інтегрувати ERP систему у розроблений клієнтський застосунок, що і утворить повноцінну автоматизовану систему керування студмістечком.

Кожен обраний програмний засіб реалізації по-своєму допомагає, як у розробці, так і в користуванні. Бібліотеки React.js, Redux.js на мові JavaScript та середовище розробки для JavaScript JetBrains WebStorm надають потужні сервіси для створення користувальницького інтерфейсу, які допомагають реалізувати необхідні функціональні можливості системи керування студмістечком та спростити процес розробки застосунку. Середовище Node.js самостійно виконує JavaScript код та за допомогою своєї потокової структури підвищує продуктивність системи. Менеджер пакетів NPM самостійно встановлює та налаштовує необхідні компоненти, а система контролю версій Git дозволяє відстежувати поетапні зміни у проєкті, що спрощує процес розробки. За допомогою хмарного сервісу Heroku процес розгортання застосунку є простим та швидким. Усі обрані програмні засоби реалізації добре взаємодіють між собою, що робить процес розробки зручним, а користувальницький досвід позитивним.

Архітектура системи керування студмістечком представлена клієнт-серверною архітектурою, що дозволяє розподілити завдання та навантаження між клієнтами та сервером. Архітектура користувальницького інтерфейсу системи представлена шаблоном MVC (Model, View, Controller), що дозволяє структурувати програмний код на чітко визначені компоненти, кожен з яких виконує свої конкретні функції. Спроєктовані діаграми прецедентів та схема компонентів, які відповідають вимогам до функціональних можливостей користувальницького інтерфейсу, допомогли визначити порядок реалізації функціональних можливостей, що мінімізувало кількість помилок при розробці.

Визначені системні вимоги для користування системою керування студмістечком являються системними вимогами для використання інтернет браузера, тому у типового користувача не повинно бути проблем зі швидкодією застосунку. Розроблена інструкція розгортання додатка з використанням хмарного сервісу Heroku допоможе користувачу швидко запустити систему керування студмістечком на хмарному сервері та розпочати з нею роботу. Опис взаємодії з користувальницьким інтерфейсом системи демонструє, що застосунок відповідає вимогам до функціональних можливостей системи, показує, що було досягнуто мету магістерської дисертації, та допомагає користувачу працювати з клієнтським застосунком.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Abildskov J. Practical Git: confident git through practice. New York : Apress, 2020. 200 p.
2. Banks A. Learning React: modern patterns for developing React apps. Sebastopol : O'Reilly Media, 2020. 310 p.
3. Bugl D. Learning Redux: write maintainable, consistent, and easy-to-test web applications. Birmingham : Packt Publishing, 2017. 374 p.
4. Demiliani S. Mastering Microsoft Dynamics 365 Business Central: discover extension development best practices, build advanced ERP integrations, and use DevOps tools. Birmingham : Packt Publishing, 2019. 770 p.
5. Gomaa H. Software modeling and design: UML, Use Cases, patterns, and software architectures. Cambridge : Cambridge University Press, 2011. 776 p.
6. Horton S. A web for everyone: designing accessible user experiences. New York : Rosenfeld Media, 2014. 288 p.
7. Ihrig C. J. Pro Node.js for developers. New York : Apress, 2013. 310 p.
8. Kemp C., Gyger B. Professional Heroku programming. New York : Wiley & Sons, 2013. 528 p.
9. Negrino T. JavaScript. Berkeley : Peachpit Press, 2012. 518 p.
10. Rosca S., Patin D. The WebStorm essentials. Birmingham : Packt Publishing, 2015. 184 p.
11. Saternos C. Client-server web apps with JavaScript and Java: rich, scalable, and RESTful. Sebastopol : O'Reilly Media, 2014. 260 p.
12. Syed B. Beginning Node.js. New York : Apress, 2014. 308 p.
13. Wexler J. Get programming with Node.js. New York : Manning, 2019. 480 p.
14. Wieruch R. The road to learn React: your journey to master plain yet pragmatic React.js. Columbia : CreateSpace Independent Publishing Platform, 2018. 198 p.
15. Верлока В.С., Коноваленко М.К., Сиволовська О.В. Стратегічний маркетинг: навчальний посібник. Харків : УкрДАЗТ, 2007. 289 с.

ДОДАТОК А

Інтеграція ERP систем на платформі Microsoft Dynamics 365 Business Central за допомогою веб сервісів

Акт впровадження

УКР.НТУУ”КПІ”_ІАТЕ_ЦТЕ_ТР11МП_22М

Аркушів 1

Київ - 2022

АКТ ВПРОВАДЖЕННЯ

ЗАТВЕРДЖУЮ
Керівник підприємства

_____ І. А. Черкашин
(підпис)

АКТ ВПРОВАДЖЕННЯ

результатів магістерської дисертації Соболєва П.О. на тему «Інтеграція ERP систем на платформі Microsoft Dynamics 365 Business Central за допомогою веб сервісів», яка виконана в Національному технічному університеті України «Київський політехнічний інститут ім. Ігоря Сікорського» («КПІ ім. Ігоря Сікорського») ”5” грудня 2022 р.

Нами, представниками кафедри цифрових технологій в енергетиці «КПІ ім. Ігоря Сікорського» та ТОВ «Ікспенд Україна», даний акт складено про те, що для використання в розробках спеціалізованого програмного забезпечення ТОВ «Ікспенд Україна» прийняті результати магістерської дисертації Соболєва П. О., а саме програмний засіб – користувальницького інтерфейсу для ERP системи керування студмістечком на платформі Microsoft Dynamics 365 Business Central у вигляді запису на CD, а також документацію програмного супроводу.

Представник кафедри ЦТЕ «КПІ ім. Ігоря Сікорського»
Керівник дипломної роботи Лабжинський Володимир Анатолійович _____
(прізвище, ім’я, по батькові) (підпис)

Представник ТОВ «Ікспенд Україна»
Керівник підприємства Черкашин Іван Андрійович _____
(прізвище, ім’я, по батькові) (підпис)