Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня

бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *Зембіцька Анжела Олександрівна*
(ПІБ)

академічної групи *122-18ск-2*
(шифр)

спеціальності *122 Комп'ютерні науки*
(код і назва спеціальності)

освітньої програми *Комп'ютерні науки*
(назва освітньої програми)

на тему: *Розробка клієнтської частини для обліку процесу розселення студентів у гуртожитку НТУ «Дніпровська політехніка»*

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи	*доц. Реута О.В.*			
розділів:				
Спеціальний	*доц. Реута О.В.*			
економічний	*проф. Вагонова О.Г.*			
Рецензент	*проф. Гнатушенко В.В.*			
Нормоконтролер	*доц. Гуліна І.Г.*			

Дніпро
2021

Міністерство освіти і науки України
НТУ «Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
завідувач кафедри
програмного забезпечення комп'ютерних систем
(повна назва)

_____ І.М. Удовик
(підпис) (прізвище, ініціали)

« » _____ 2021 року

ЗАВДАННЯ
на кваліфікаційну роботу

бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *122-18ск-2* *Зембіцької Анжели Олександрівни*
(група) (прізвище та ініціали)

тема кваліфікаційної роботи *Розробка клієнтської частини для обліку процесу розселення студентів у гуртожитку НТУ «Дніпровська політехніка»*

затверджена наказом ректора НТУ «ДП» від *07.06.2021* № *317-с*

Розділ	Зміст виконання	Термін виконання
Спеціальний	***На основі матеріалів виробничої практики та інших науково-технічних джерел провести аналіз стану рішення проблеми та постановку задачі.*** ***Обґрунтувати вибір та здійснити реалізацію методів вирішення проблеми***	***13.05.2021 р.***
Економічний	***Провести розрахунок трудомісткості розробки програмного забезпечення, витрат на створення ПЗ й тривалості його розробки***	***27.05.2021 р.***

Завдання видав _____ *доц. Реута О.В*
(підпис) (посада, прізвище, ініціали)

Завдання прийняв до виконання _____ *Зембіцька А.О.*
(підпис) (прізвище, ініціали)

Дата видачі завдання: *14.01.2021 р.*

Термін подання кваліфікаційної роботи до ЕК: *11.06.2021 р.*

РЕФЕРАТ

Пояснювальна записка: 87 с., 38 рис., 3 дод., 20 джерел.

Об'єкт розробки: інформаційна система гуртожитку для ведення обліку та зберігання даних.

Мета кваліфікаційної роботи: створення системи підтримки операцій адміністрації гуртожитку для підвищення продуктивності та скорочення затрат часу на занесення даних та ведення обліку реалізований за допомогою автоматизації процесу ведення бази даних системи та виконання запитів.

У вступі розглядається аналіз та сучасний стан проблеми, конкретизується мета кваліфікаційної роботи та галузь її застосування, наведено обґрунтування актуальності теми та уточнюється постановка завдання.

У першому розділі проаналізовано предметну галузь, визначено актуальність завдання та призначення розробки, сформульовано постановку завдання, зазначено вимоги до програмної реалізації, технологій та програмних засобів.

У другому розділі проаналізовані наявні рішення, обрано платформи для розробки, виконано проектування і розробка програми, описана робота програми, алгоритм і структура її функціонування, а також виклик та завантаження програми, визначено вхідні і вихідні дані, охарактеризовано склад параметрів технічних засобів.

В економічному розділі визначено трудомісткість розробленої інформаційної системи, проведений підрахунок вартості роботи по створенню програми та розраховано час на його створення.

Практичне значення полягає у створенні додатка, що надає можливість електронного зберігання даних про гуртожиток та студентів, ведення бази даних та облік поселень та виселень.

Актуальність інформаційної системи визначається попитом на подібні розробки, що оптимізують та спрощують дії щодо ведення бази даних

гуртожитку, скорочують час на введення даних на заселення; підвищують ефективність та швидкість роботи адміністрації шляхом електронного ведення обліку, та унеможливлюють помилки при поселенні студентів у кімнати.

Список ключових слів: ІНФОРМАЦІЙНА СИСТЕМА, КОМП’ЮТЕР, ОБЛІК, БАЗА ДАНИХ, АВТОМАТИЗАЦІЯ, ПРОЕКТУВАННЯ, ДОДАТОК, МЕНЮ, ВКЛАДКА.

ABSTRACT

Explanatory note: 87 pages, 38 figures, 3 appendix, 20 sources.

Object of development: dormitory information system for accounting and data storage.

The purpose of the qualification work: to create a system to support the operations of the dormitory administration to increase productivity and reduce the time spent on data entry and accounting implemented through the automation of the process of maintaining the system database and queries.

The introduction considers the analysis and the current state of the problem, specifies the purpose of the qualification work and the scope of its application, provides a justification for the relevance of the topic and clarifies the problem.

In the first section the subject branch is analyzed, the urgency of the task and the purpose of development are defined, the statement of the task is formulated, the requirements to software realization, technologies and software are specified.

The second section analyzes the available solutions, selected platforms for development, designed and developed the program, describes the program, algorithm and structure of its operation, as well as calling and loading the program, determines the input and output data, describes the parameters of hardware.

The economic section determines the complexity of the developed information system, calculates the cost of work to create a program and calculates the time for its creation.

The practical significance lies in the creation of an application that provides the ability to electronically store data on dormitories and students, maintain a database and record settlements and evictions.

The relevance of the information system is determined by the demand for such developments that optimize and simplify the maintenance of the dormitory database, reduce the time to enter data for settlement; increase the efficiency and speed of the administration through electronic accounting, and prevent mistakes when settling students in rooms.

Keywords: INFORMATION SYSTEM, COMPUTER, ACCOUNTING, DATABASE, AUTOMATION, DESIGN, APPENDIX, MENU, TAB.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД – база даних;

ІС – інформаційна система;

ОС – операційна система;

ПЗ – програмне забезпечення;

СКБД – система керування базами даних;

ER – Entity Relationship;

SQL - Structured Query Language;

UML – Unified Modeling Language.

ЗМІСТ

ВСТУП

Розроблена система призначена для автоматизації роботи адміністрації гуртожитку ВНЗу.

Студентський гуртожиток –місце тимчасового проживання студентів під час навчання, яке належить ВУЗам. Зазвичай гуртожитки мають від п’яти поверхів, з близько 20 кімнатами на поверсі, де можуть проживати від двох людей, один ВНЗ може мати від одного гуртожитку такого типу.

Ведення обліку в Excel для такої кількості даних може призвести до виникнення помилок у пік заселення студентів – на початку навчального року.

Створення автоматизованої системи для гуртожитку обумовлене потребою маніпулювати великими масивами даних, автоматизації та прискорення роботи адміністрації, та уникнення можливих помилок та накладання даних при заселенні.

Метою кваліфікаційної роботи є розробка веб-додатку для автоматизації роботи адміністрації гуртожитку.

Для досягнення поставленої мети необхідно вирішити такі основні задачі

– провести аналіз організації роботи адміністрації та основні виконувані задачі;

– провести аналіз наявних аналогів на нашому та зарубіжному ринку;

– на основі результатів порівнянь висунути вимогу до системи та виконуваних процесів;

– спроектувати алгоритм, базу даних, та створити його програмну реалізацію;

– здійснити тестування й відлагодження створеного програмного забезпечення.

У відповідності до проведеного аналізу поставлені основні функціональні задачі перед системою:

– автоматизація ведення даних;

– зниження проценту помилок при заселенні;
– прискорення роботи адміністратора.

Практичне значення розробленого проекту полягає в автоматизації процесів ведення даних, можливість студентів занесення своїх даних, та подавання заявок на заселення, полегшення роботи адміністрації гуртожитку, можливість швидкого оновлення даних.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1.Загальні відомості з предметної галузі

1.1.1 Поняття та актуальність теми

Студентський гуртожиток – це місце тимчасового проживання студентів під час навчання, яке належить ВУЗам.

Актуальність розробки проекту очевидна. З кожним роком збільшується кількість бажаючих поступити в вищий навчальний заклад. У зв'язку зі економічними труднощами, не кожен іногородній студент має можливість знімати квартиру, або жити у готелях. Кожен рік у гуртожитки заселяється приблизно 30% від першокурсників, не враховуючи студентів старших курсів, які вже проживають у гуртожитках. Це обумовлюється досить низькою вартістю проживання, зручним розташуванням по відношенню до навчальних корпусів університету, і не поганими житловими умовами.

Заселення це критичний процес, тобто зсув його в часі може призвести до різних надзвичайних ситуацій. Наприклад, зрив графіка заселення, неминуче призведе до неможливості навчального процесу студентів з інших міст.

При заселені студентів у гуртожитки, необхідна автоматизація та своєчасне оновлення інформації, щоб через недостовірність даних про наявність вільних місць та великої кількості числа студентів, бажаючих заселитися, в гуртожитках не склалася ситуація, коли в двомісну кімнату записують троє, а то й четверо студентів, що природно неприпустимо по санітарним нормам (6 $м^2$ на людину).

1.1.2 Огляд наявних програмних аналогів

У відкритому доступі в інтернеті не було знайдено жодного аналога з подібними функціями.

Зі знайдених систем є додаток TSU.Helper, який належить Томському ВНЗ.

Через нього можна подати заявку на виправлення побутових проблем в гуртожитках. Студенти можуть повідомити про поломку в своїй кімнаті або загальнодоступних місцях в гуртожитку. Вхід в TSU.Helper здійснюється через сервіс ТГУ.Аккаунти, при цьому аккаунт повинен бути підтвердженим. У додатку можна залишити заявку, яка буде перевірена і передана на виконання електрику, теслі, сантехніку та іншим. Список заявок переглядає комендант гуртожитку. Він ставить завдання працівникам і контролює процес виконання. Коли заявка буде виконана, студент отримає повідомлення.

1.2.Призначення розробки та галузь застосування

Кваліфікаційна робота призначена для створення додатка, що надає можливість автоматизації роботи системи гуртожитку, тобто прискорення та полегшення роботи адміністрації.

Система дасть можливість студентам заносити свої дані, та обирати кімнати для проживання. А адміністрації гуртожитку залишиться лише підтвердити дані студента та заявку на заселення.

Головна цінність розробленої системи полягає в тому, що вона буде зберігати актуальні дані про стан заселення, вільні кімнати і т.д., де можна через функції пошуку знаходити необхідну інформацію, що прискорить процес заселення, та не буде створювати проблем накладки даних.

1.3.Підстава для розробки

Підставою для розробки кваліфікаційної роботи на тему «Розробка клієнтської частини для обліку процесу розселення студентів у гуртожитку НТУ «Дніпровська політехніка» є наказ по Національному технічному університету «Дніпровська політехніка» від 07.06. 2021р. № 317-с.

1.4.Постановка завдання

Задачею кваліфікаційної роботи є розробка серверної та клієнтської частини додатку «Гуртожиток №1». Додаток має мати наступні функції:

– Виведення інформації про гуртожиток та поверхи;

– Виведення розгорнутої інформації про кімнати – кількість місць, кількість вільних місць, опис кімнати;

– Реєстрація/авторизація користувачів;

– Вибір кімнати та заповнення заявок на заселення до кімнати;

– Редагування адміністратором даних;

Додаток має бути зі зрозумілим для користувача інтерфейсом, мати перевірку валідності даних та шифрування паролів. Також дані повинні зберігатися на сервері у базі даних та бути логічно структурованими.

За результатами тестування програми, необхідно зробити висновки з працеспроможності розробленого додатку та його функцій.

1.5.Вимоги до програми або програмного виробу

1.5.1. Вимоги до функціональних характеристик

Програмна реалізація проекту має складатися з таких частин, як сторінки входу/реєстрації, головної сторінки, інформації про кімнати, форми для заповнення заявки, сторінки для обробки даних комендантом або адміністратором.

Дані повинні зберігатися на сервері у базі даних, та мати шифрування паролів і cookie.

1.5.2. Вимоги до інформаційної безпеки

Основні вимоги до інформаційної безпеки:

– цілісність даних;

– захист від неавторизованого редагування;

– конфіденційність інформації;

– доступність інформації для всіх авторизованих користувачів.

Для надійної роботи програмного забезпечення необхідно дотримуватися таких факторів:

− використання ліцензійного ПЗ;

− захист від зловмисних програм;

− шифрування даних на сервері;

− встановлення блоків безперебійного живлення.

1.5.3. Вимоги до складу та параметрів технічних засобів

Для програмного додатку потрібен комп’ютер чи ноутбук з рекомендованими параметрами:

– підтримка 64-розрядного процесору та операційної системи;

– операційна система Windows 10 64bit;

– процесор Intel Core i5 3470 3.2 GHz/AMD X8 FX-8350 4 GHz;

– оперативна пам'ять 8GB ОЗУ;

– відео-карта nVidia GTX 650 з 1 Гб відео-пам’яті, AMD HD7860 з 1 Гб Відео-пам’яті;

– DirectX версії 12;

– жорсткий диск мінімум в 72 GB.

1.5.4. Вимоги до інформаційної та програмної сумісності

Програма має бути написана на об’єктно-орієнтованій мові програмування Java для серверної частини, та HTML, CSS та JavaScript для клієнтської частини веб-додатку.

Функціонування розробленої системи має бути забезпечено на наступних операційних системах:

– Windows 98, 2000 і вище;
– Windows NT 4.0, XP.

Та у всіх браузерах, по типу Chrome, Internet Explorer, Mozilla Firefox та ін.

РОЗДІЛ 2
ПРОЄКТУВАННЯ ТА РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1. Функціональне призначення системи

Робота забезпечує можливість автоматизації роботи системи гуртожитку, тобто прискорення та полегшення роботи адміністрації. Розроблена система дає можливість студентам самостійно заносити свої дані, обирати кімнати та заповнювати заявки на заселення. Адміністратору необхідно буде лише перевірити дані та розглянути заявку, а саме відхилити або підтвердити.

2.2. Опис застосованих математичних методів

Розроблена система не потребує застосування математичних методів, окрім простих операцій, що використовуються під час роботи алгоритмів та аналізу даних, та не потребують окремого опису.

2.3. Опис використаних технологій та мов програмування

Для розробки клієнтської частини інформаційної системи була використана мова гіпертекстової розмітки HTML, каскадна таблиця стилів CSS та фреймворк Bootstrap. Для серверної частини використана мова Java, фреймворки Maven та Spring та об’єктно-реляційна система керування базами даних PostgreSQL. Проект був розроблений у середовищі IntelliJ IDEA Ultimate 2020.3.3

HTML (від англ. HyperText Markup Language - «мова гіпертекстової розмітки») - стандартизована мова розмітки документів у Всесвітній павутині. Більшість веб-сторінок містять опис розмітки на мові HTML. Мова HTML інтерпретується браузерами; отриманий в результаті інтерпретації

форматований текст відображається на екрані монітора комп'ютера або мобільного пристрою.

Мова HTML до 5-ї версії визначалась як додаток SGML (стандартної узагальненої мови розмітки за стандартом ISO 8879). Специфікації HTML5 формулюються в термінах DOM (об'єктній моделі документа).

У всесвітній павутині HTML-сторінки, як правило, передаються браузерам від сервера по протоколах HTTP або HTTPS, у вигляді простого тексту або з використанням шифрування.

В HTML можна вбудувати програмний код на мові програмування JavaScript, для управління поведінкою і змістом веб-сторінок. Також включення CSS в HTML описує зовнішній вигляд і макет сторінки.

HTML - тегова мова розмітки документів. Будь-який документ на мові HTML являє собою набір елементів, причому початок і кінець кожного елемента позначається спеціальними позначками - тегами. Елементи можуть бути порожніми, тобто не містять ніякого тексту та інших даних. В цьому випадку зазвичай не вказується тег що закриває (наприклад, тег розриву рядків
 - одиночний і закривати його не потрібно). Крім того, елементи можуть мати атрибути, що визначають будь-які їх властивості (наприклад, атрибут href = "у посилання). Атрибути вказуються в відкриваючому тезі. Ось приклади фрагментів HTML-документа:

 Текст між двома тегами - відкриває і закриває. </ strong>

 Тут елемент містить атрибут href, тобто гіперпосилання.

А ось приклад порожнього елемента:

Регістр, в якому набране ім'я елемента і імена атрибутів, в HTML значення не має (на відміну від XHTML). Елементи можуть бути вкладеними.

Крім елементів, в HTML-документах є і сутності - «спеціальні символи». Суті починаються з символу амперсанда і мають вигляд & ім'я; або & # NNNN ;, де NNNN - код символу в Юнікоді в десятковій системі числення.

Наприклад, & copy; - знак авторського права (©). Як правило, сутності використовуються для представлення символів, відсутніх в кодуванні документа, або ж для виведення «спеціальних» символів: & amp; - амперсанда (&), & lt; - символу «менше» (<) і & gt; - символу «більше» (>), які некоректно записувати «звичайним» чином, через їх особливе значення в HTML.

CSS - (англ. Cascading Style Sheets «каскадні таблиці стилів») - формальна мова опису зовнішнього вигляду документа (веб-сторінки), написаного з використанням мови розмітки (найчастіше HTML або XHTML). Також може застосовуватися до будь-яких XML-документах, наприклад, до SVG або XUL.

CSS використовується розробниками веб-сторінок для завдання кольорів, шрифтів, стилів, розташування окремих блоків і інших аспектів представлення зовнішнього вигляду цих веб-сторінок. Основною метою розробки CSS було відділення опису логічної структури веб-сторінки від опису зовнішнього вигляду цієї веб-сторінки (яке тепер проводиться за допомогою формального мови CSS). Такий поділ може збільшити доступність документа, надати велику гнучкість і можливість управління його поданням, а також зменшити складність і повторюваність в структурному вмісті.

Крім того, CSS дозволяє представляти один і той же документ в різних стилях або методах виведення, таких як екранне уявлення, друковане подання, читання голосом (спеціальним голосовим браузером або програмою читання з екрану), або при виведенні пристроями, що використовують шрифт Брайля

Правила CSS можуть розташовуватися як в самому веб-документі, зовнішній вигляд якого вони описують, так і в зовнішніх файлах, що мають розширення .css. Формат CSS - це текстовий файл, в якому міститься перелік правил CSS і коментарів до них.

Стилі CSS можуть бути підключені або впроваджені в описуваний ними веб-документ чотирма способами:

– коли опис стилів знаходиться в окремому файлі, воно може бути підключено до документа за допомогою елемента <link>, включеного в елемент <head>:

<link rel="stylesheet" type="text/css" href="style.css">

– коли файл стилів розміщується окремо від батьківського документа, він може бути підключений до документа інструкцією @import в елементі <style>:

<style media="all"> @import url(style.css); </style>

– коли стилі описані всередині документа, вони можуть бути включені в елемент <style>, який, включається в елемент <head>:

<style>

body { color: red; }

</style>

– коли стилі описані в тілі документа, вони можуть розташовуватися в атрибутах окремого елемента:

У перших двох випадках до документа застосовані зовнішні стилі, а в наступних двох - внутрішні стилі.

Bootstrap (також відомий як Twitter Bootstrap) - вільний набір інструментів для створення сайтів і веб-додатків. Включає в себе HTML- і CSS-шаблони оформлення для типографіки, веб-форм, кнопок, міток, блоків навігації та інших компонентів веб-інтерфейсу, включаючи JavaScript-розширення.

Bootstrap використовує сучасні напрацювання в області CSS і HTML, тому необхідна бібліотека почала розроблятися як внутрішня бібліотека компанії Twitter під назвою Twitter Blueprint. Після кількох місяців розробки він був відкритий під назвою Bootstrap 19 серпня 2011.

Основними нововведеннями другої версії, що з'явилася 31 січня 2012 року, стали 12-колоночная сітка і підтримка адаптивності.

Третя версія випущена 19 серпня 2013 року. У ній адаптивність отримала подальший розвиток, був здійснений перехід до концепції mobile first, оптимізації перш за все під мобільні пристрої. Дизайн за замовчуванням став плоским.

Робота над четвертою версією розпочато 29 жовтня 2014 року. Альфа-версія вийшла 19 серпня 2015 року. Перша бета-версія випущена 10 серпня 2017. Друга бета-версія випущена 19 жовтня 2017. 18 січня 2018 випущена перша стабільна версія Bootstrap 4.

Основні інструменти Bootstrap:

– сітки - заздалегідь задані розміри колонок, які можна відразу ж використовувати, наприклад, ширина колонки 140 px відноситься до класу .span2 (.col-md-2 в третій версії фреймворка), який можна використовувати в CSS-описі документа;

– шаблони - фіксований або гумовий шаблон документа;

– типографіка - опису шрифтів, визначення деяких класів для шрифтів, таких як код, цитати;

– медіа - надає деяке упорядкування зображень та відео;

– таблиці - кошти оформлення таблиць, аж до додавання функціональності сортування;

– форми - класи для оформлення форм і деяких подій, що відбуваються з ними;

– навігація - класи оформлення для панелей, вкладок, переходу по сторінках, меню і панелі інструментів;

– алерти - оформлення діалогових вікон, підказок і спливаючих вікон.

Java- строго типізована об'єктно-орієнтована мова програмування загального призначення, розроблений компанією Sun Microsystems (в подальшому придбаної компанією Oracle). Розробка ведеться співтовариством, організованим через Java Community Process; мова і основні його технології поширюються за ліцензією GPL. Права на торговельну марку належать корпорації Oracle.

Програми Java зазвичай транслюються в спеціальний байт-код, тому вони можуть працювати на будь-якої комп'ютерної архітектурі, для якої існує реалізація віртуальної Java-машини. Дата офіційного випуску - 23 травня 1995

року. Займає високі місця в рейтингах популярності мов програмування (2-е місце в рейтингах IEEE Spectrum (2020) і TIOBE (2021)).

Програми на Java транслюються в байт-код Java, який виконується віртуальною машиною Java (JVM) - програмою, яка обробляє байтовий код і інструкції яка передається обладнанню як інтерпретатор.

Перевагою подібного способу виконання програм є повна незалежність байт-коду від операційної системи і устаткування, що дозволяє виконувати Java-додатки на будь-якому пристрої, для якого існує відповідна віртуальна машина. Іншою важливою особливістю технології Java є гнучка система безпеки, в рамках якої виконання програми повністю контролюється віртуальною машиною. Будь-які операції, які перевищують встановлені повноваження програми (наприклад, спроба несанкціонованого доступу до даних або з'єднання з іншим комп'ютером), викликають негайне переривання.

Часто до недоліків концепції віртуальної машини відносять зниження продуктивності. Ряд удосконалень кілька збільшив швидкість виконання програм на Java:

– застосування технології трансляції байт-коду в машинний код безпосередньо під час роботи програми (JIT-технологія) з можливістю збереження версій класу в машинному коді;

– широке застосування переносних орієнтованого коду (native-код) в стандартних бібліотеках;

– апаратні засоби, що забезпечують прискорену обробку байт-коду (наприклад, технологія Jazelle, підтримувана деякими процесорами архітектури ARM).

За даними сайту shootout.alioth.debian.org, для семи різних завдань час виконання на Java становить в середньому в півтора-два рази більше, ніж для C / C ++, в деяких випадках Java швидше, а в окремих випадках в 7 разів повільніше. З іншого боку, для більшості з них споживання пам'яті Java-машиною було в 10-30 разів більше, ніж програмою на C / C ++. Також примітне дослідження, проведене компанією Google, згідно з яким

відзначається істотно нижча продуктивність і більше споживання пам'яті в тестових прикладах на Java в порівнянні з аналогічними програмами на C ++.

Ідеї, закладені в концепцію і різні реалізації середовища віртуальної машини Java, надихнули безліч ентузіастів на розширення переліку мов, які могли б бути використані для створення програм, що виконуються на віртуальній машині. Ці ідеї знайшли також вираз в специфікації загальномовна інфраструктури CLI, закладеної в основу платформи .NET компанією Microsoft.

Apache Maven - фреймворк для автоматизації збирання проектів на основі опису їх структури в файлах на мові POM (англ. Project Object Model), що є підмножиною XML. Проект Maven видається співтовариством Apache Software Foundation, де формально є частиною Jakarta Project.

Назва системи є словом з мови ідиш, сенс якого можна приблизно висловити як «збирач знання».

Maven забезпечує декларативну, а не імперативну (на відміну від засобу автоматизації збирання Apache Ant) збірку проекту. У файлах опису проекту міститься його специфікація, а не окремі команди виконання. Всі завдання по обробці файлів, описані в специфікації, Maven виконує за допомогою їх обробки послідовністю вбудованих і зовнішніх плагінів.

Maven використовується для побудови і управління проектами, написаними на Java, C #, Ruby, Scala, та іншими мовами.

Серед примітних альтернатив - система автоматичного складання Gradle, побудована на принципах Apache Ant і Maven, але використовує спеціалізований DSL на Groovy замість POM-конфігурації.

Spring Framework (або коротко Spring) - універсальний фреймворк з відкритим вихідним кодом для Java-платформи. Також існує форк для платформи .NET Framework, названий Spring.NET.

Перша версія була написана Родом Джонсоном, який вперше опублікував її разом з виданням своєї книги «Expert One-on-One Java EE Design and Development» (Wrox Press, жовтень 2002 року).

Фреймворк був вперше випущений під ліцензією Apache 2.0 license в червні 2003 року. Перша стабільна версія 1.0 була випущена в березні 2004. Spring 2.0 був випущений в жовтні 2006, Spring 2.5 - в листопаді 2007, Spring 3.0 в грудні 2009, і Spring 3.1 в грудні 2011. Поточна версія - 5.3.x.

Незважаючи на те, що Spring не забезпечував якусь конкретну модель програмування, він став широко поширеним в Java-співтоваристві головним чином як альтернатива і заміна моделі Enterprise JavaBeans. Spring надає велику свободу Java-розробникам в проектуванні; крім того, він надає добре документовані і легкі у використанні засоби вирішення проблем, що виникають при створенні додатків корпоративного масштабу.

Тим часом, особливості ядра Spring застосовні в будь-якому Java-додатку, і існує безліч розширень і удосконалень для побудови веб-додатків на Java Enterprise платформі. З цих причин Spring набув великої популярності і визнається розробниками як стратегічно важливий фреймворк.

PostgreSQL - вільна об'єктно-реляційна система управління базами даних (СКБД).

Існує в реалізаціях для безлічі UNIX-подібних платформ, включаючи AIX, різні BSD-системи, HP-UX, IRIX, Linux, macOS, Solaris / OpenSolaris, Tru64, QNX, а також для Microsoft Windows.

Сильними сторонами PostgreSQL вважаються:

– високопродуктивні і надійні механізми транзакцій і реплікації;

– розширювана система вбудованих мов програмування: в стандартному постачанні підтримуються PL / pgSQL, PL / Perl, PL / Python і PL / Tcl; додатково можна використовувати PL / Java, PL / PHP, PL / Py, PL / R, PL / Ruby, PL / Scheme, PL / sh і PL / V8, а також є підтримка завантаження модулів розширення на мові C;

– спадкування;

– можливість індексування геометричних (зокрема, географічних) об'єктів і наявність базується на ній розширення PostGIS;

– вбудована підтримка слабоструктурованих даних в форматі JSON з можливістю їх індексації;

– розширюваність (можливість створювати нові типи даних, типи індексів, мови програмування, модулі розширення, підключати будь-які зовнішні джерела даних).

PostgreSQL створена на основі некомерційної СУБД Postgres, розробленої як open-source проект в Каліфорнійському університеті в Берклі. До розробки Postgres, що почалася в 1986 році, мав безпосереднє відношення Майкл Стоунбрейкер, керівник більш раннього проекту Ingres, на той момент вже придбаного компанією Computer Associates. Назва розшифровується як «Post Ingres», і при створенні Postgres були застосовані багато ранніх напрацювання.

Стоунбрейкер і його студенти розробляли нову СУБД протягом восьми років з 1986 по 1994 рік. За цей період в синтаксис були введені процедури, правила, призначені для користувача типи і інші компоненти. У 1995 році розробка знову розділилася: Стоунбрейкер використовував отриманий досвід у створенні комерційної СУБД Illustra, яку просуває його власної однойменної компанією (придбаної згодом компанією Informix), а його студенти розробили нову версію Postgres - Postgres95, в якій мова запитів POSTQUEL - спадщина Ingres - був замінений на SQL.

Розробка Postgres95 була виведена за межі університету і передана команді ентузіастів. Нова СУБД отримала ім'я, під яким вона відома і розвивається в поточний момент - PostgreSQL.

IntelliJ IDEA - інтегроване середовище розробки програмного забезпечення для багатьох мов програмування, зокрема Java, JavaScript, Python, розроблена компанією JetBrains.

Перша версія з'явилася в січні 2001 року і швидко набула популярності як перша середа для Java з широким набором інтегрованих інструментів для рефакторінга, які дозволяли програмістам швидко реорганізовувати вихідні тексти програм. Дизайн середовища орієнтований на продуктивність роботи

програмістів, дозволяючи сконцентруватися на функціональних завданнях, в той час як IntelliJ IDEA бере на себе виконання рутинних операцій.

Починаючи з шостої версії продукту IntelliJ IDEA надає інтегрований інструментарій для розробки графічного інтерфейсу користувача. Серед інших можливостей, середа добре сумісна з багатьма популярними вільними інструментами розробників, такими як CVS, Subversion, Apache Ant, Maven і JUnit. У лютому 2007 року розробники IntelliJ анонсували ранню версію плагіна для підтримки програмування на мові Ruby.

Починаючи з версії 9.0, середа доступна в двох редакціях: Community Edition і Ultimate Edition. Community Edition є повністю вільною версією, доступною під ліцензією Apache 2.0, в ній реалізована повна підтримка Java SE, Kotlin, Groovy, Scala, а також інтеграція з найбільш популярними системами управління версіями. В редакції Ultimate Edition, доступною під комерційною ліцензією, реалізована підтримка Java EE, UML-діаграм, підрахунок покриття коду, а також підтримка інших систем управління версіями, мов та фреймворків.

Створення нового проекту в IntelliJ IDEA. Програма пропонує вибір мов програмування.

Мови:

- Java;
- JavaScript;
- HTML / XHTML / HAML;
- CSS / SASS / LESS;
- XML / XSL / XPath;
- ActionScript / MXML;
- Python;
- Ruby;
- Groovy;
- Scala;

– SQL;

– PHP;

– Kotlin;

– Clojure;

– Ci;

– C ++;

– Go;

– Rust (за допомогою офіційного плагіна).

Ряд мов підтримується за допомогою плагінів сторонніх розробників, зокрема, так реалізована підтримка OCaml, GLSL, Erlang, Fantom, Haskell, Lua, Mathematica, Perl5, Object Pascal.

2.4. Опис структури програми та алгоритмів її функціонування

2.4.1 Опис алгоритмів програми

Програма має 5 рівнів доступу прецедентів до програми що зображено на Use-case діаграмі (рис.2.1), а саме:

Перший рівень – Незареєстрований користувач, до його можливостей відносимо:

– Встановити додаток/ зайти на сайт.

– Переглянути інформацію про кімнати.

Він може побачити, які кімнати вільні або зайняті, кількість мешканців у кімнатах, та наявність вільних місць.

Другий рівень – Гість, він успадковує можливості Незареєстрованого користувача, але не є підтвердженим студентом, тому він має особистий кабінет без можливості заповнення заявок на заселення.

Третій рівень – Студент, він успадковує можливості Незареєстрованого користувача, та має додатково такі можливості:

– Зареєструватися/Ввійти.

– Обрати кімнату.

– Заповнити заявку на заселення.

Четвертий рівень – Комендант, він успадковує можливості Студента, та наступні властивості:

– Розглянути заявку на заселення.

– Прийняти або відхилити заявку.

– Додати інформацію про нового мешканця у БД;

– Виселити мешканця, що проживає.

– Видалити інформацію про мешканця, який виселився/якого виселили.

П’ятий рівень – Адміністратор, він має можливості, що і Комендант. Також має додаткові властивості:

– Реєструвати нових користувачів сайту/додатку, надавати права доступу;

– Змінювати інформацію про кімнати – вільні чи заселені.

– Змінювати інформацію про мешканців.

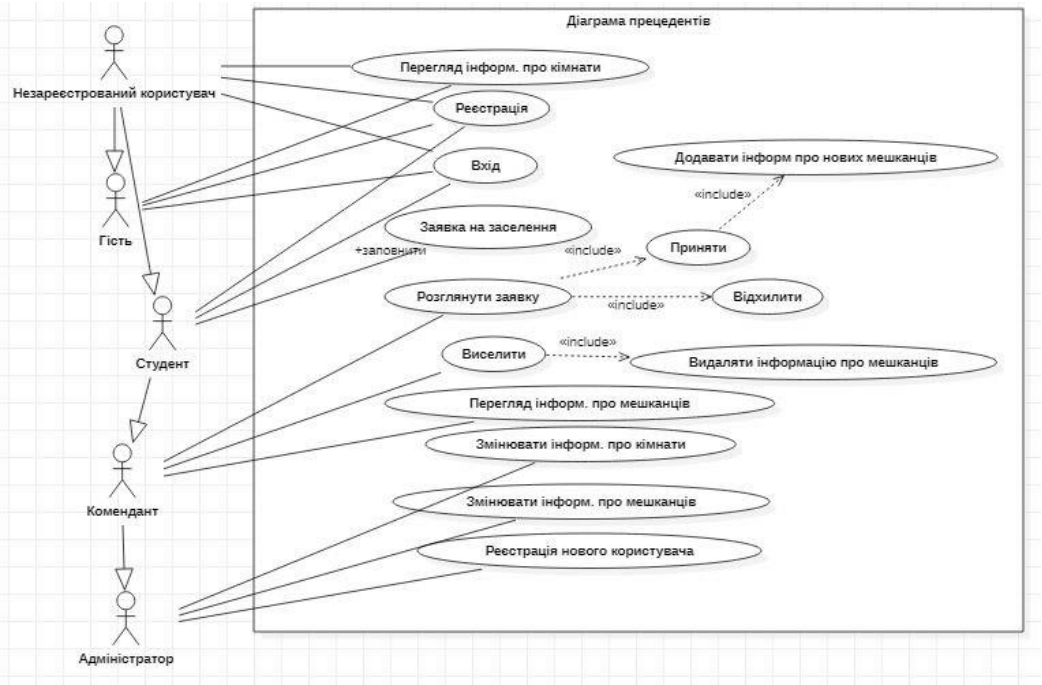


Рис. 2.1. Use-case діаграма.

2.4.2 Структура роботи програми

Інтерфейс розробленої програми має наступну структуру:

– реєстрація, під час якої студент створює свій обліковий запис та заносить свої дані;

– вхід, де студент вказує свої логін та пароль, та переходить до розширеного списку можливостей;

– перегляд інформації, де студент бачить інформацію про кімнати, їй опис та наявність місць;

– заповнення заявки на заселення.

Після підтвердження заявки адміністратором, це відобразиться в особистому кабінеті студента.

На рисунку 2.2 зображена діаграма послідовності, яка повністю описує цикл взаємодії студента з програмою.

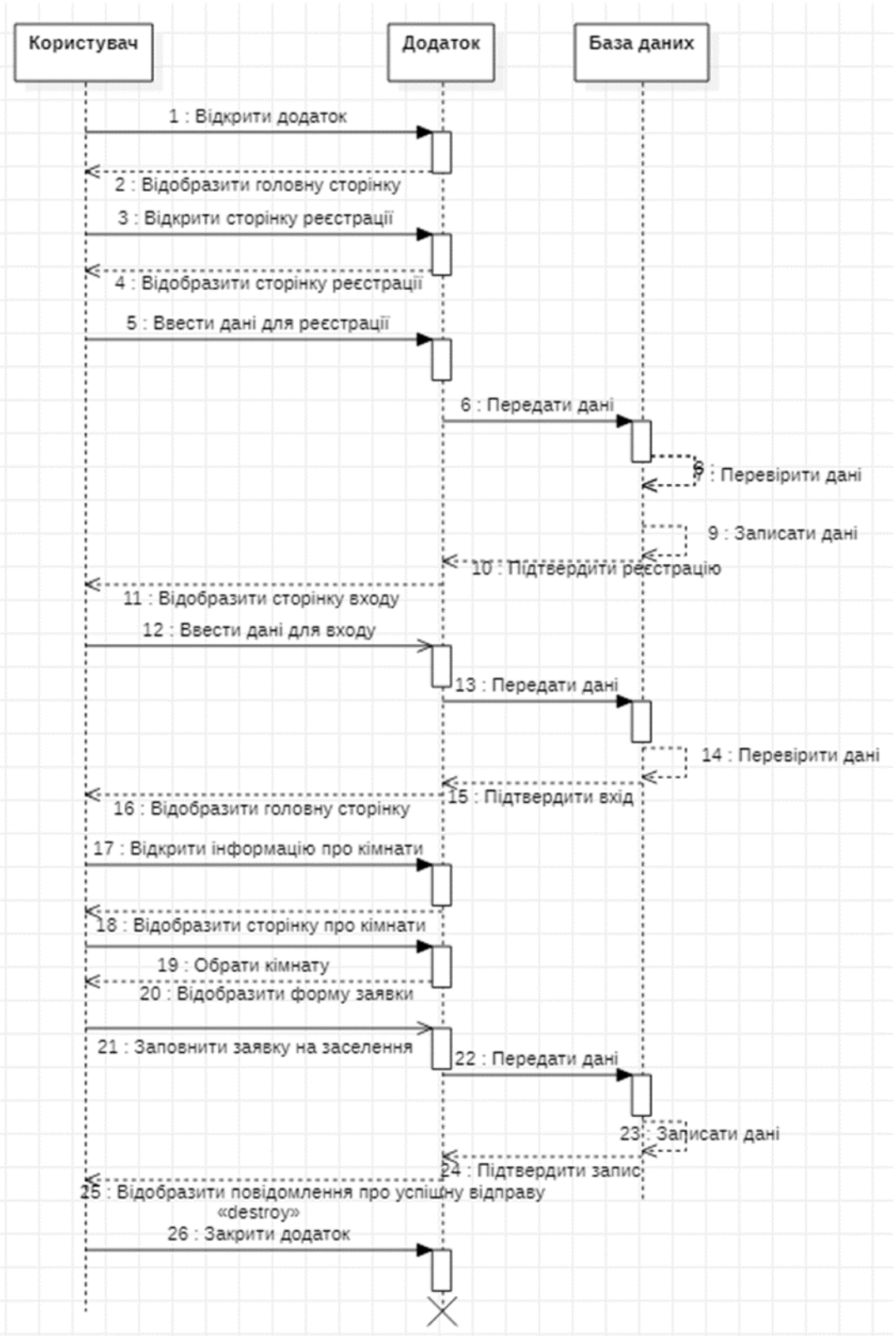


Рис. 2.2 Діаграма послідовностей

2.5. Обґрунтування та організація вхідних та вихідних даних програми

До вхідних даних проекту належать дані, які передаються до бази даних через форми, а саме:

1. Дані, що задає студент під час реєстрації;

Під час реєстрації студент задає такі дані:

– логін – текстове поле, необхідні дані формату [a-z], та розміром від 5 до 20 символів;

– пароль – текстове поле з типом «password», де необхідно ввести дані формату [0-9a-zA-Z], та розміром мінімум 8 символів;

– ПІБ – текстові поля, з даними формату [А-Яа-яґҐЁёІіЇїЄє'’\s-], де враховується, що може бути подвійне ім’я або прізвище;

– факультет – скорочена назва факультету, наприклад ФІТ – Факультет Інформаційних Технологій;

– група – навчальна група студента;

– номер телефону ;

– місце реєстрації – прописка студента;

2. Дані у формі входу– логін та пароль;

3. Дані що задає адміністратор про поверхи:

– номер поверху;

– кількість кімнат;

4. Дані що задає адміністратор про кімнати:

– номер поверху;

– номер кімнати;

– кількість мешканців;

– опис.

До вихідних даних належать всі дані, що виводяться з бази даних під час роботи проекту:

– дані, занесені студентом та адміністратором;
– відомості про користувачів та студентів;
– відомості про всі заявки;
– відомості про гуртожиток.

2.6. Опис розробленої системи

2.6.1. Використані технічні засоби

Для розробки ІС використовувався комп’ютер з наступними параметрами:
– операційна система Windows 10;
– процесор Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz;
– оперативна пам'ять 8GB ОЗП;
– відео-карта nVidia GeForce 8600/9600GT, ATI/AMD;
– Radeon HD2600/3600;
– DirectX версії 9.0c;
– жорсткий диск мінімум в 15 GB.

2.6.2. Використані програмні засоби

Запропонована програма написана на строго типізованій об’єктно-орієнтовані мові програмування Java в середовищі програмування IntelliJ IDEA Ultimate 2020.3.3

2.6.3. Виклик та завантаження програми

Для запуску проекту необхідно запустити pgAdmin – для роботи локального серверу (рис.2.3), та в браузері ввести «localhost:8080/» і натиснути «Enter», після чого запуститься головна сторінка сайту (рис.2.4).

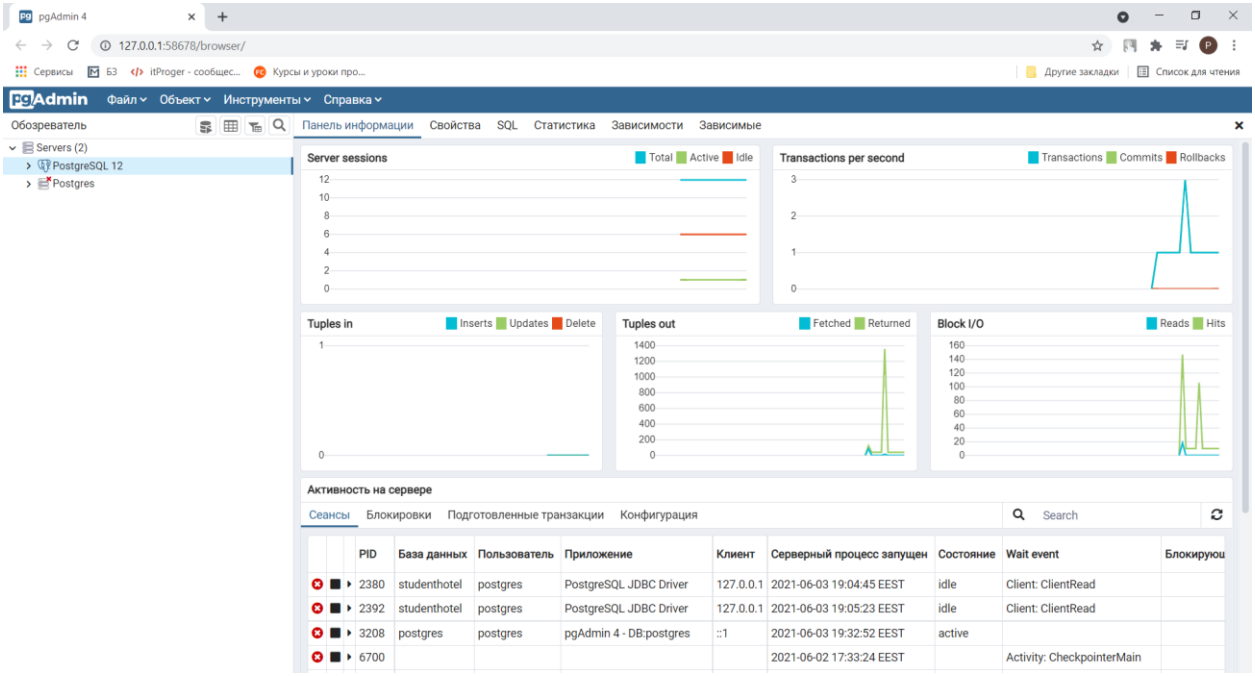
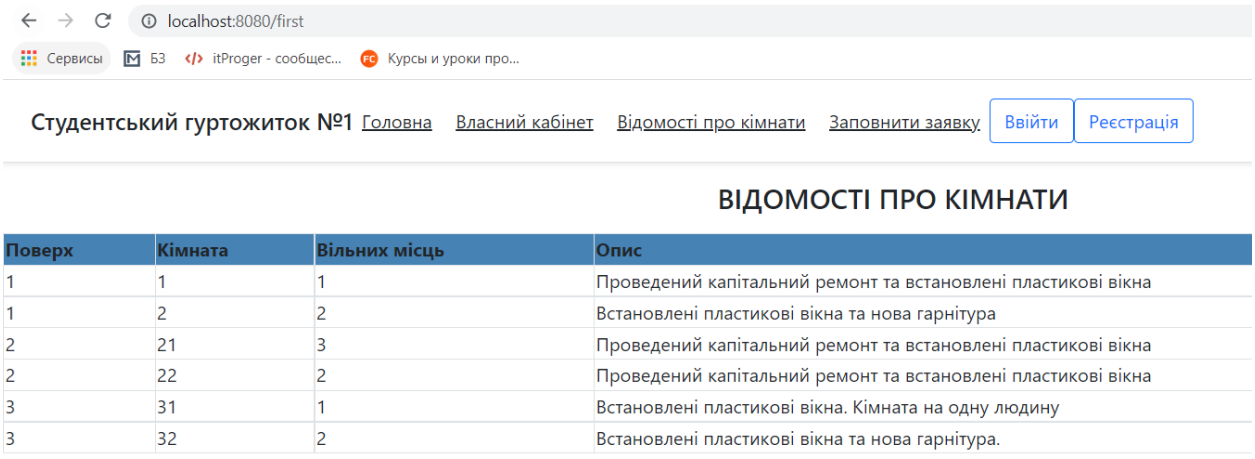


Рис. 2.3. Локальний сервер.



Поверх	Кімната	Вільних місць	Опис
1	1	1	Проведений капітальний ремонт та встановлені пластикові вікна
1	2	2	Встановлені пластикові вікна та нова гарнітура
2	21	3	Проведений капітальний ремонт та встановлені пластикові вікна
2	22	2	Проведений капітальний ремонт та встановлені пластикові вікна
3	31	1	Встановлені пластикові вікна. Кімната на одну людину
3	32	2	Встановлені пластикові вікна та нова гарнітура.

Рис. 2.4. Головна сторінка сайту.

2.6.4. Опис інтерфейсу користувача

Програма складається з таких вікон:

1. Головна сторінка (рис.2.5) – перша сторінка, що відображається при запуску. У верхній частині є «шапка», де відображаються ссилки на інші

сторінки та кнопки «Реєстрація» та «Вхід». В нижній частині виводиться «футер», де відображається права та правила.

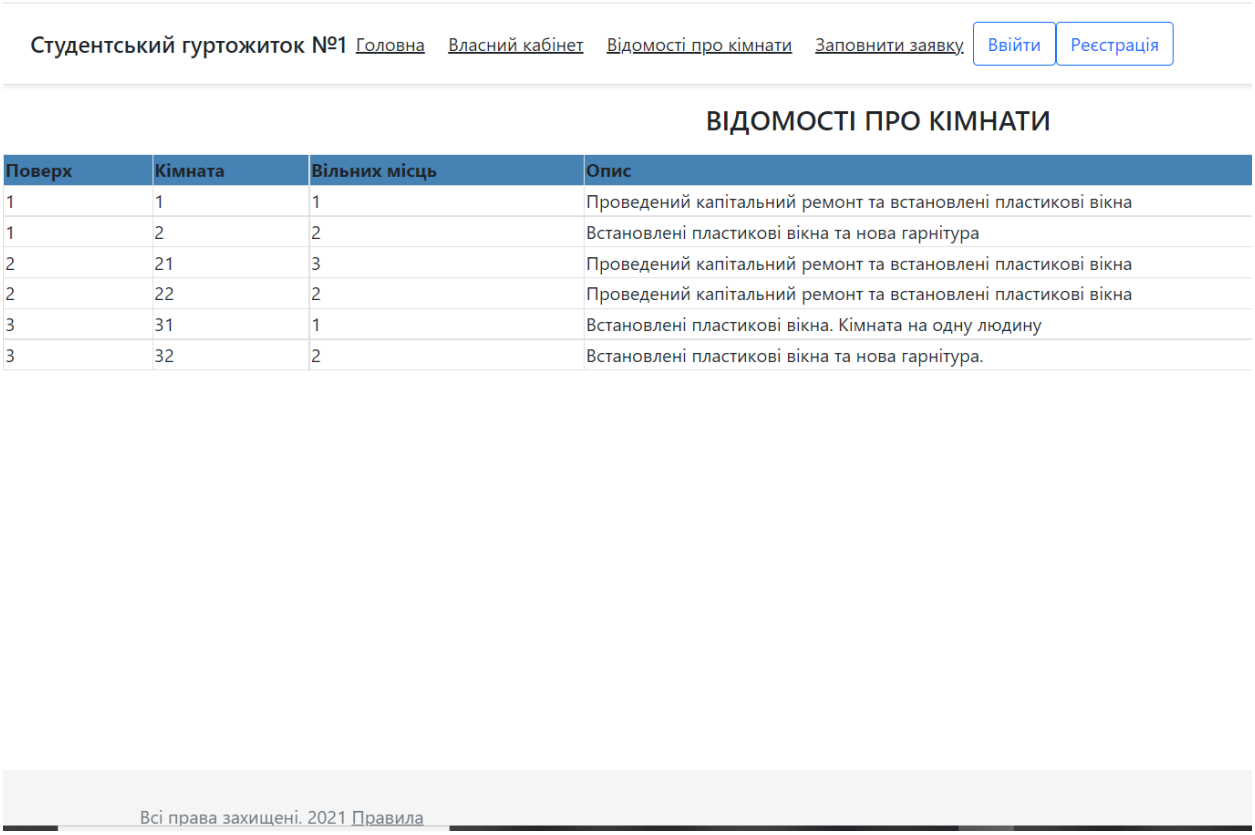


Рис. 2.5. Головна сторінка.

2. Сторінка реєстрації (рис.2.6) – необхідно задати свої дані, які у подальшому будуть використовуватися для входу та інших діях. Також, необхідно щоб логін був англійськими літерами у нижньому реєстрі. Пароль складався з літер та цифр. А інші дані були написані на українській мові.

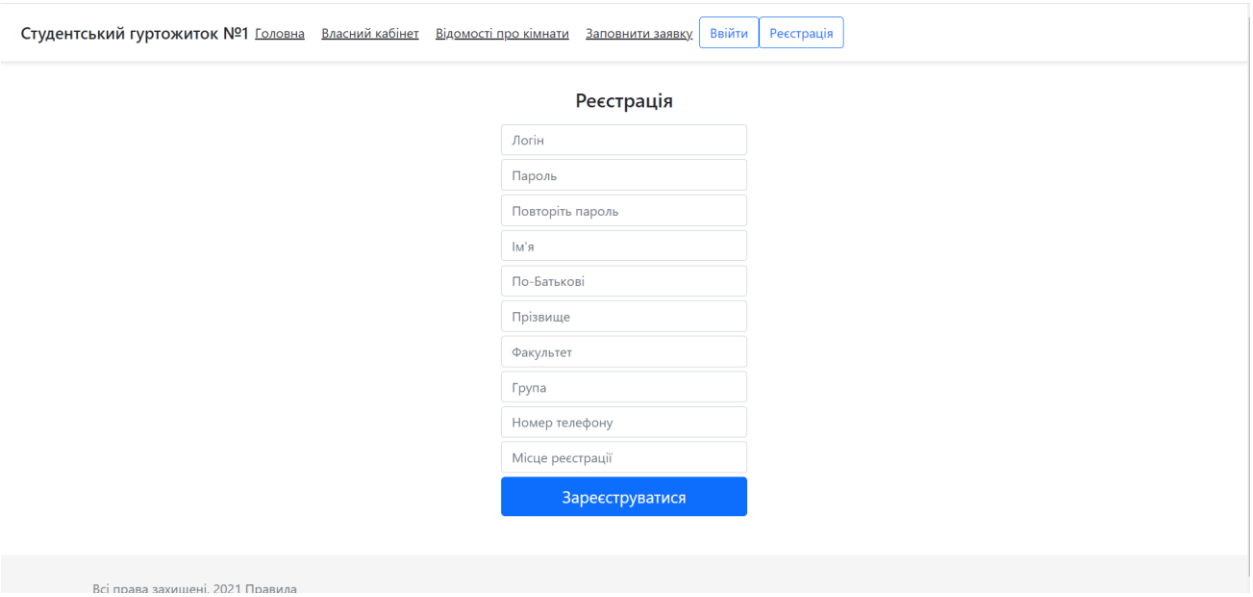


Рис. 2.6. Сторінка реєстрації.

3. Сторінка входу (рис.2.7) – сторінка, де необхідно ввести власні дані для подальшої роботи з сайтом.

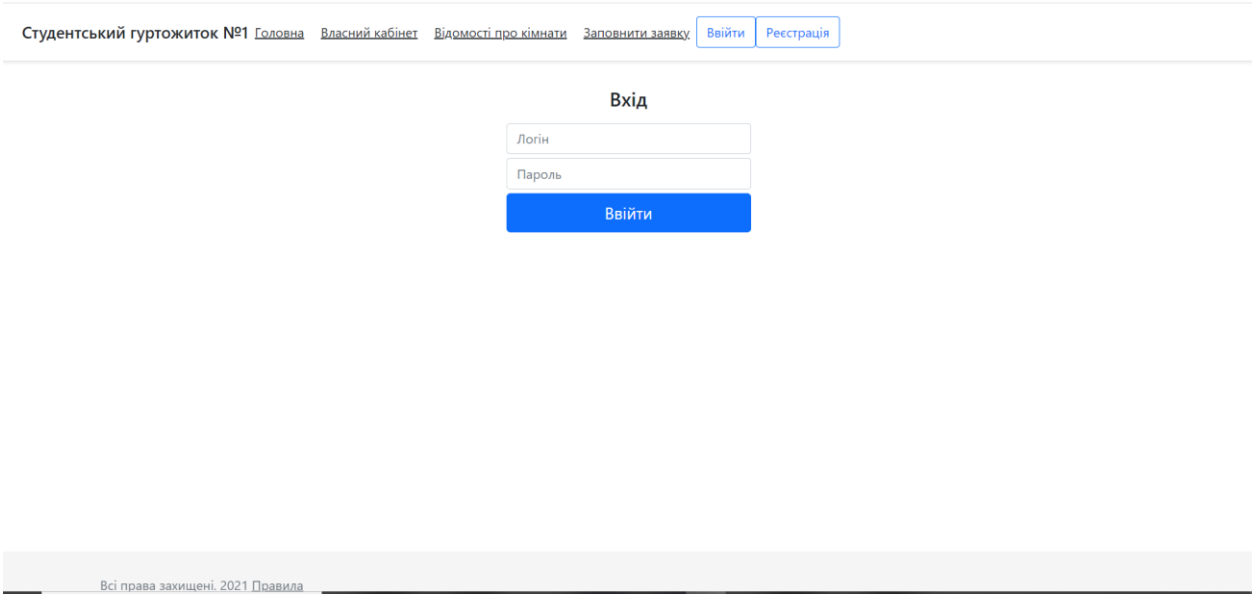


Рис. 2.7. Сторінка входу.

4. Власний кабінет (рис.2.8) – сторінка, де відображається інформація про користувача, його роль, заявки та їх статус –прийнято або відхилено.

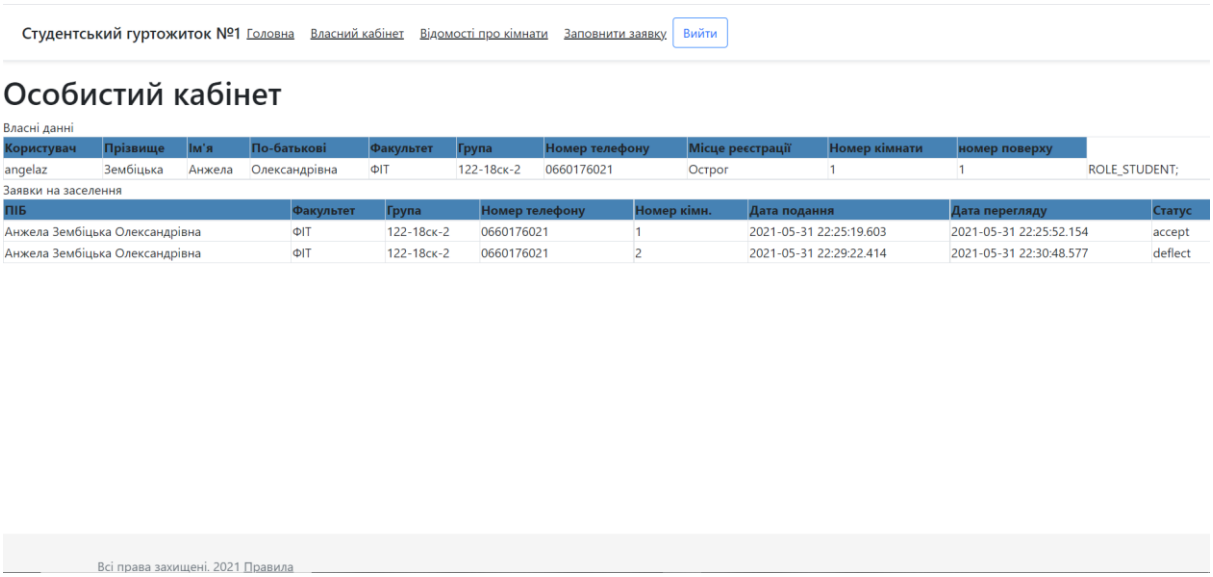


Рис. 2.8. Власний кабінет.

5. Відомості про кімнати (рис.2.9)- сторінка, де виводиться актуальна інформація про кімнати та їх стан.

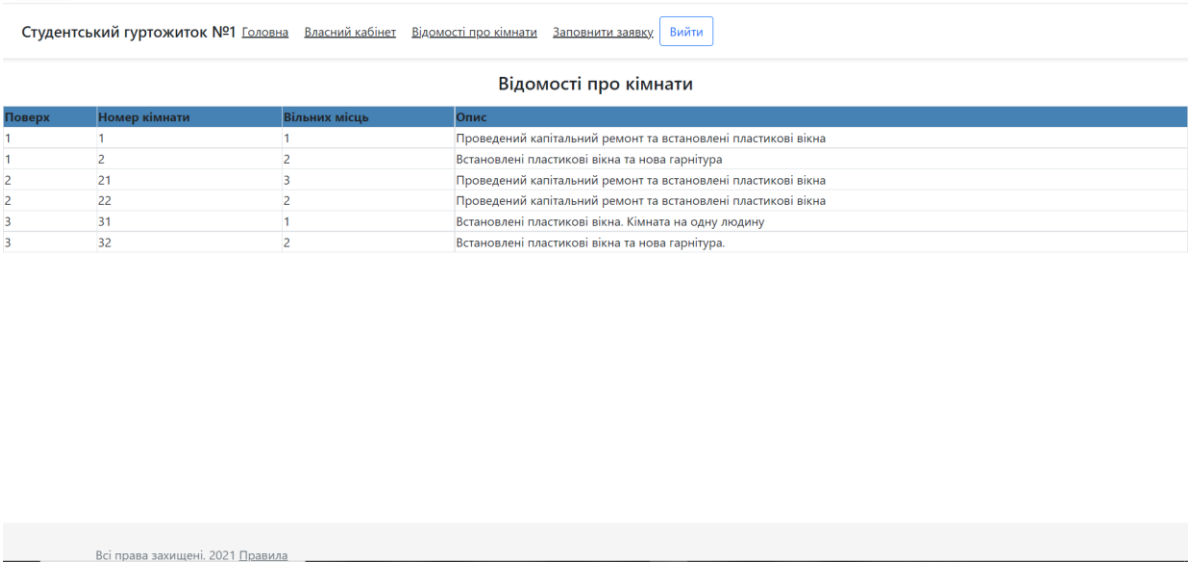


Рис. 2.9. Відомості про кімнати.

6. Заповнити заявку (рис.2.10) – сторінка, де студент може заповнити заявку на заселення. Необхідно ввести свої дані у заданому форматі та обрати номер поверху та кімнату.

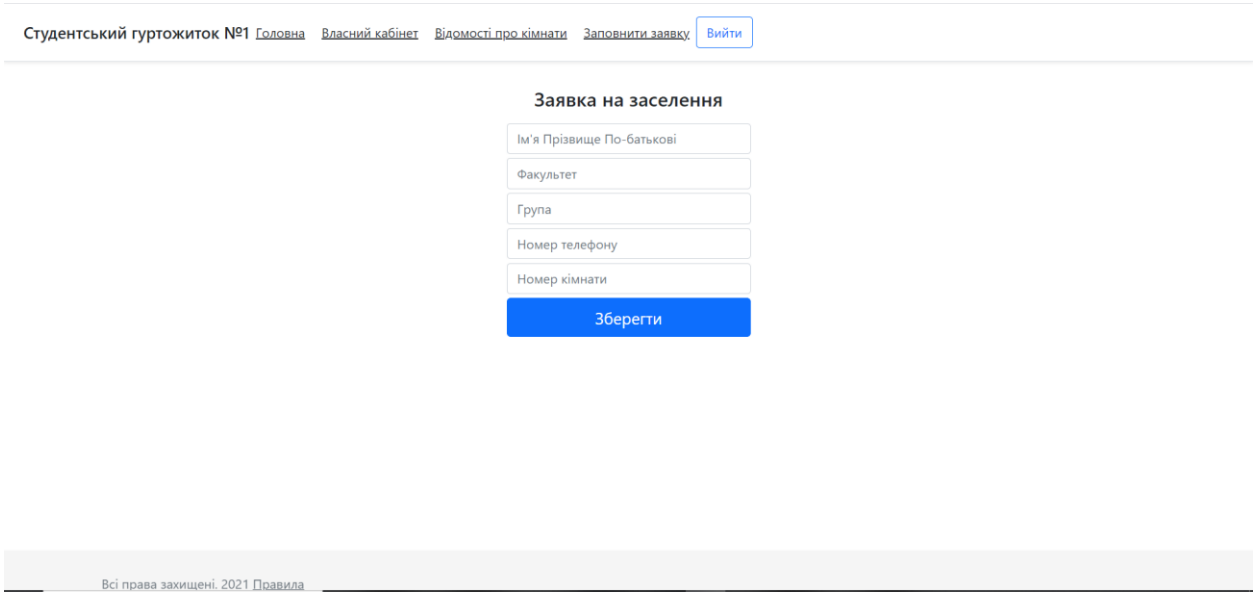


Рис. 2.10. Заявка на заселення.

7. Користувачі (рис.2.11) – сторінка для адміністратора, де виводяться дані про користувачів, реєстрації яких необхідно підтвердити, та дані про підтверджених студентів. Також є функції пошуку за параметрами – за факультетом, групою та прізвищем.

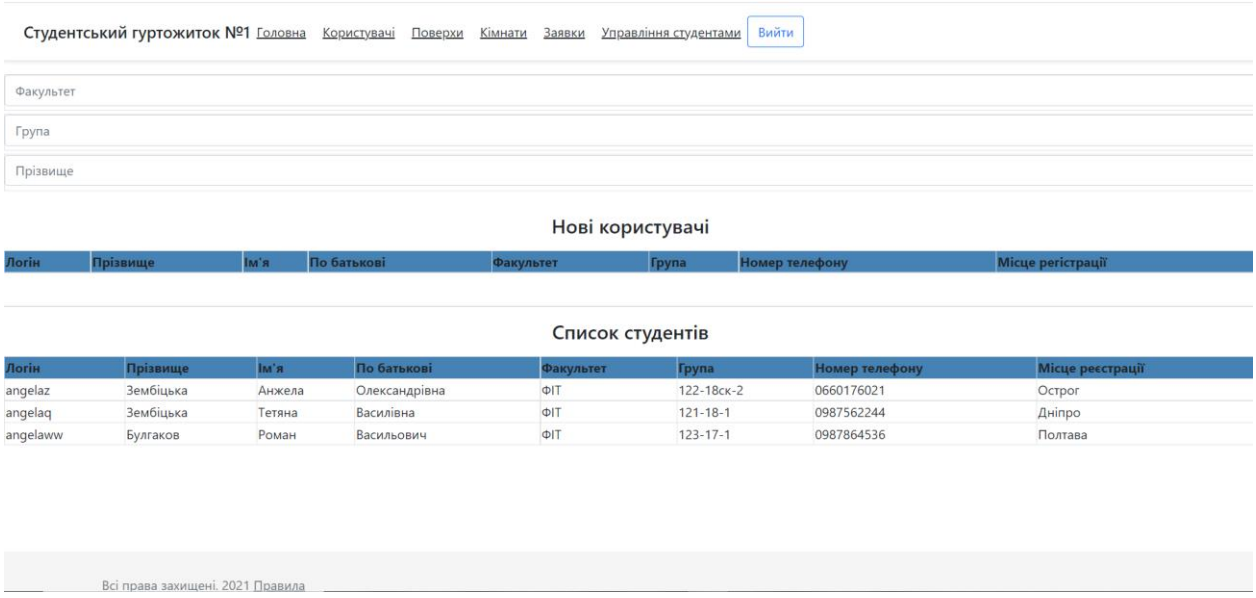


Рис. 2.11. Користувачі.

8. Поверхи (рис.2.12) – сторінка для адміністратора, де можна додати дані про поверхи, або видалити інформацію про наявні поверхи.

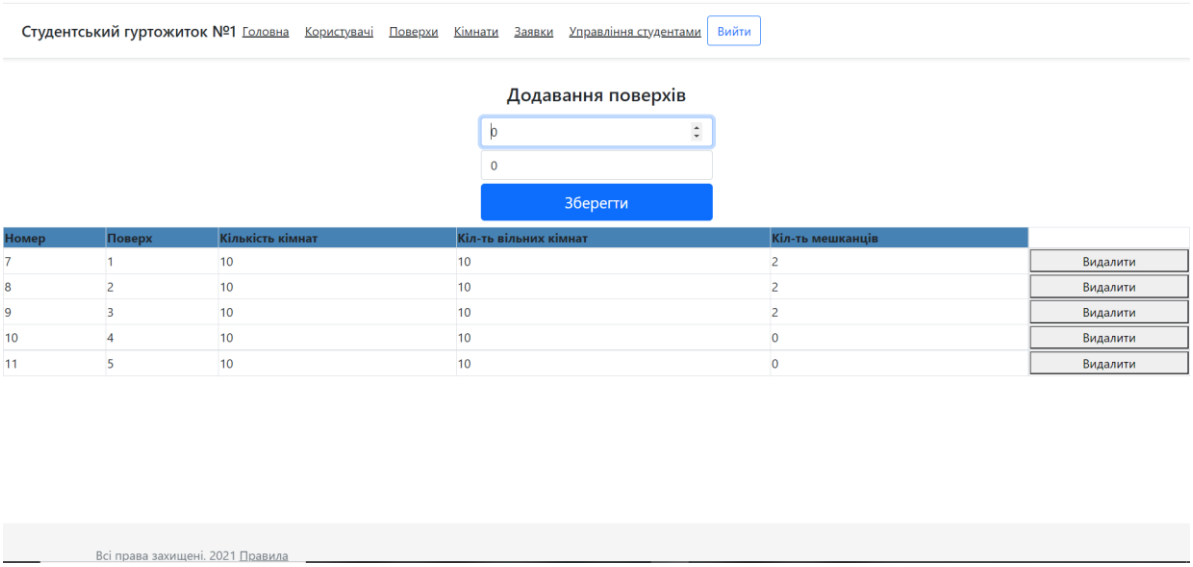


Рис. 2.12. Поверхи.

9. Кімнати (рис.2.13) – сторінка для адміністратора, де можна додати інформацію про кімнати, їх опис, або переглянути інформацію про наявні кімнати та змінити дані.

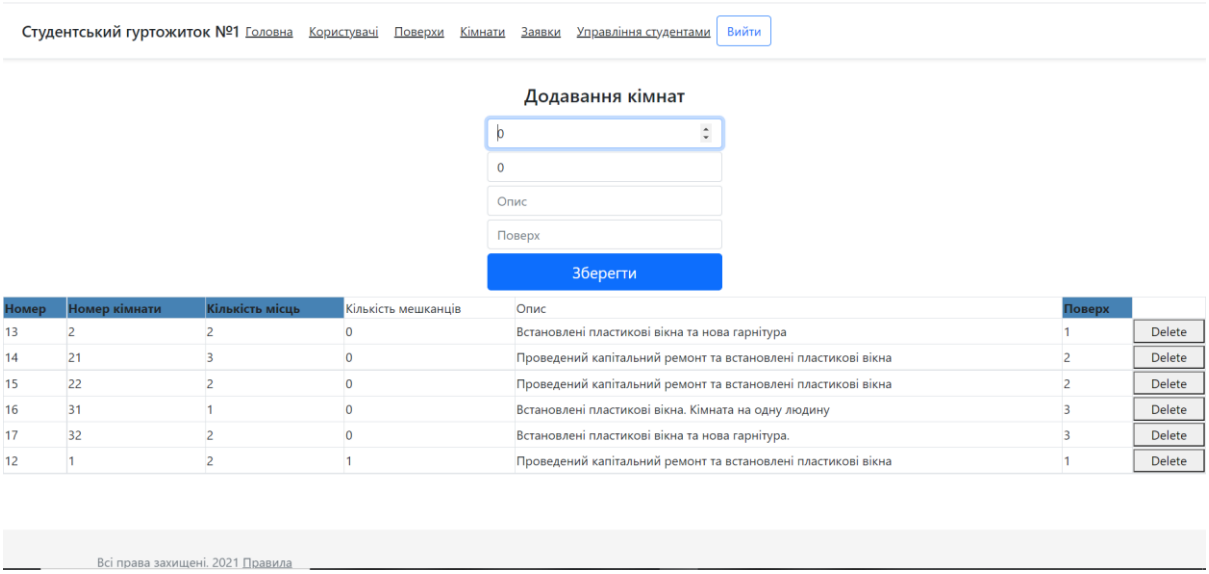


Рис. 2.13. Кімнати.

10. Заявки (рис.2.14) сторінка для адміністратора. Виводяться дані про подані заявки, де можна їх або підтвердити або відхилити.

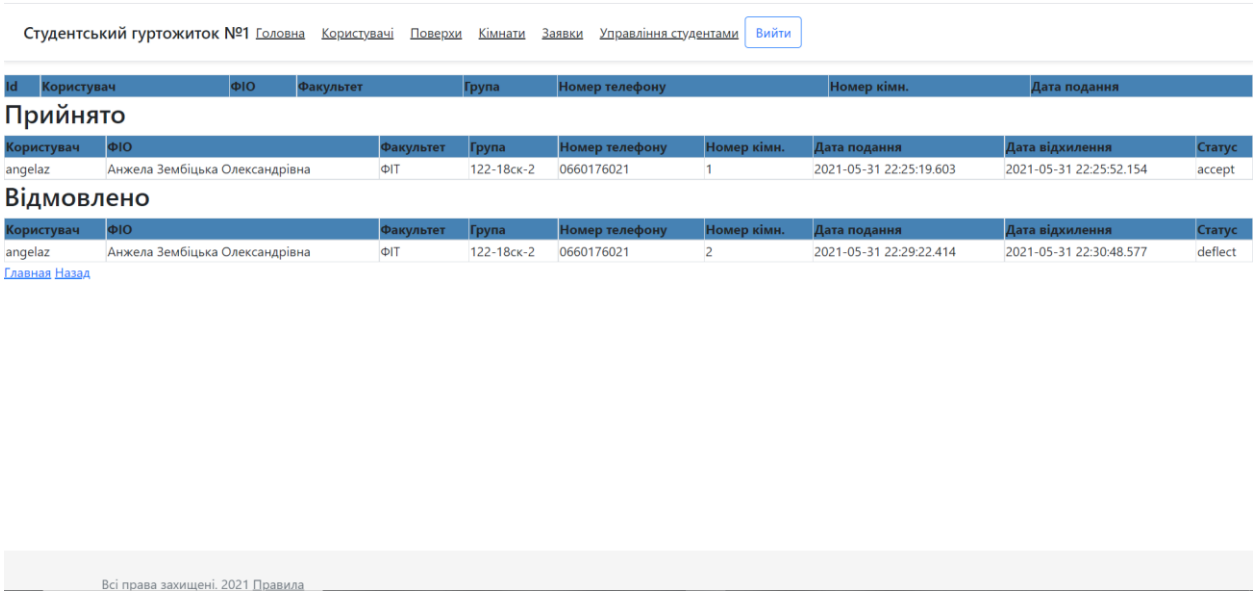


Рис. 2.14. Заявки.

11. Управління студентами (рис.2.15) – сторінка для адміністратора, де ми можемо заселяти або виселяти студентів, та додавати нові заселення.

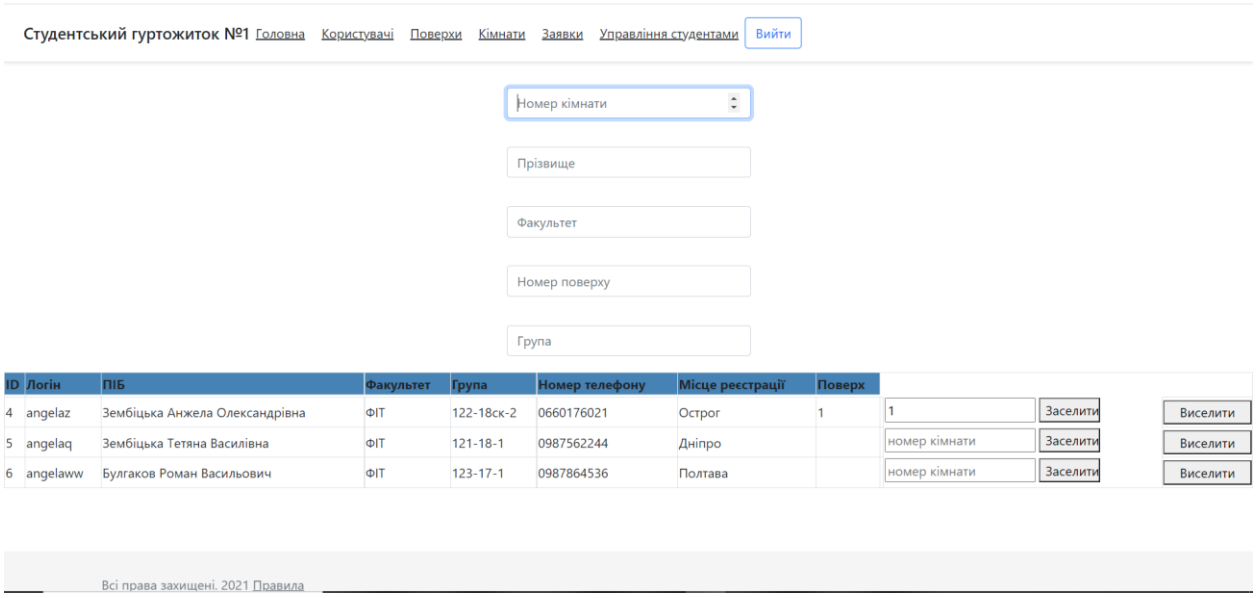


Рис. 2.15. Управління студентами.

2.6.5. Порядок роботи з розробленою системою

2.6.5.1 Порядок роботи для користувача

Для студента першою дією буде відкрити сайт, та побачити головну сторінку (рис. 2.16).

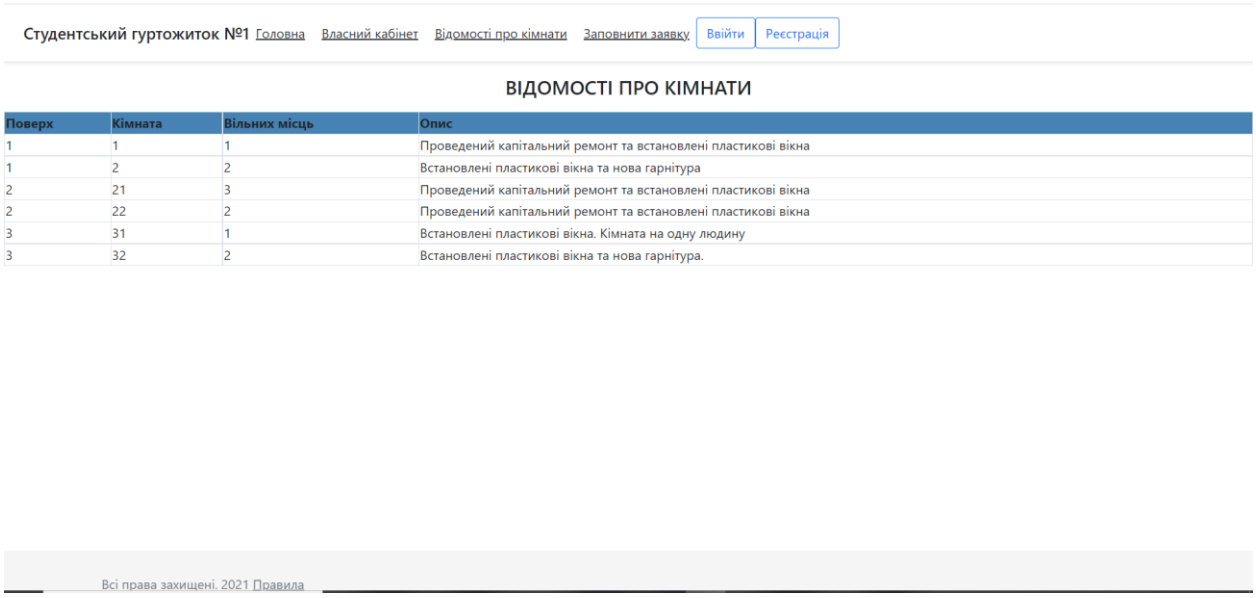


Рис. 2.16. Головна сторінка.

Після чого, якщо студент не був зареєстрованим – виконати реєстрацію (рис. 2.17), де необхідно вказати валідні дані, у іншому випадку буде виведена помилка вводу даних (рис. 2.18).

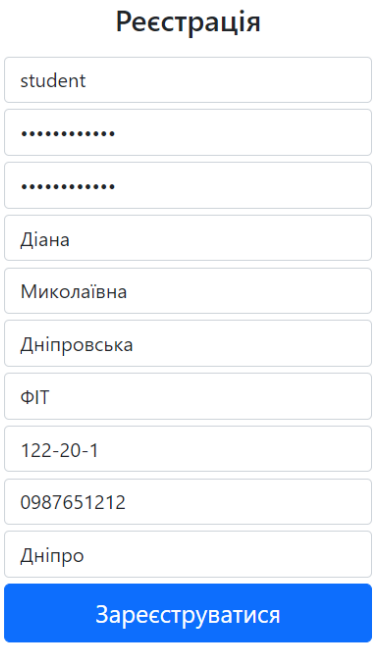


Рис. 2.17. Реєстрація.

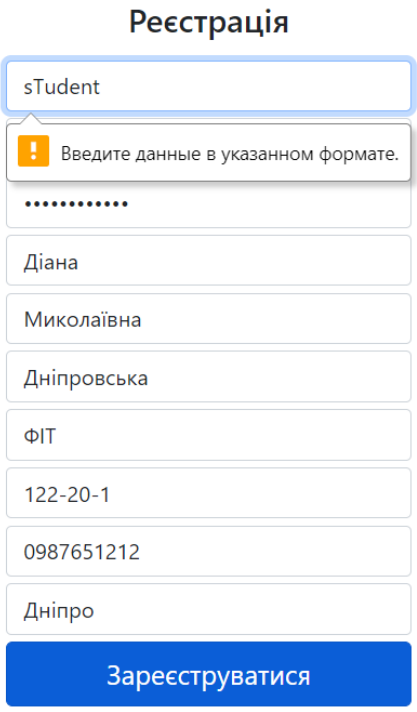


Рис. 2.18. Помилка при реєстрації.

Після вдалої реєстрації, необхідно виконати вхід до аккаунту (рис. 2.19).

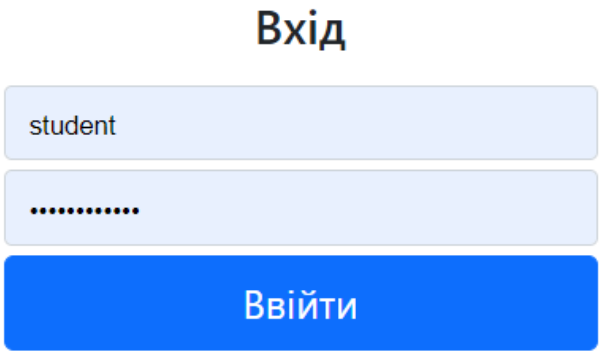


Рис. 2.19. Форма входу.

В особистому кабінеті буде відображена інформація про користувача та його статус (рис.2.20), після підтвердження адміністратором, його статус зміниться на студента (рис. 2.21).

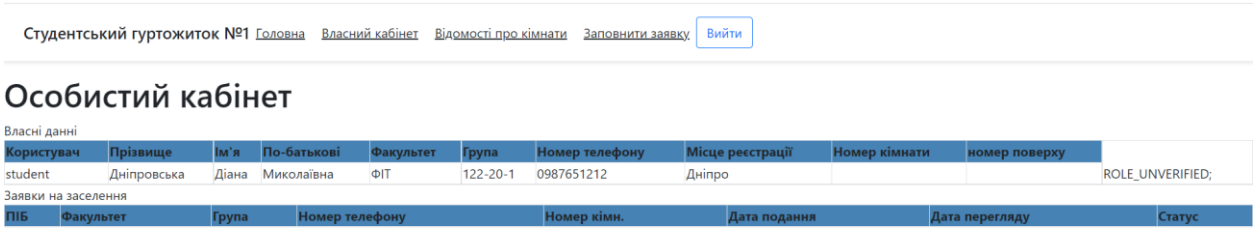


Рис. 2.20. Дані до підтвердження.

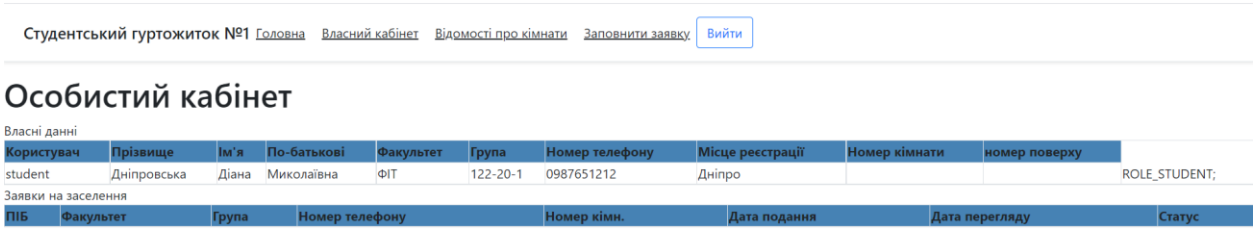
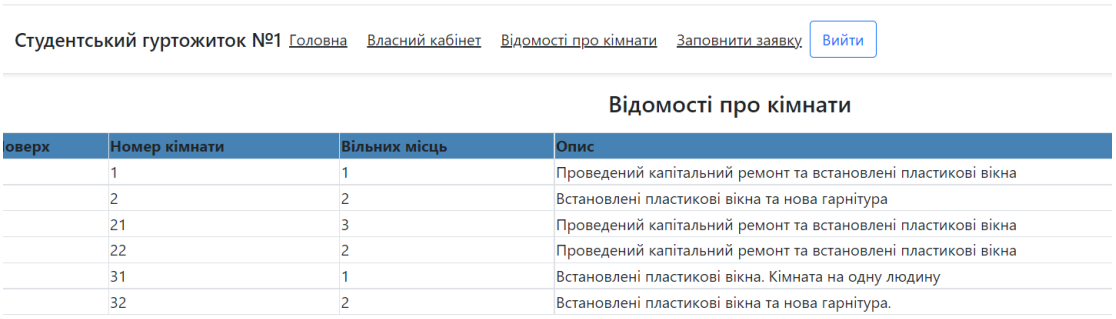


Рис. 2.21. Дані після підтвердження.

Студент може переглянути актуальну інформацію про кімнати та обрати кімнату, до якої б хотів заселитися (рис. 2.22).



оверх	Номер кімнати	Вільних місць	Опис
	1	1	Проведений капітальний ремонт та встановлені пластикові вікна
	2	2	Встановлені пластикові вікна та нова гарнітура
	21	3	Проведений капітальний ремонт та встановлені пластикові вікна
	22	2	Проведений капітальний ремонт та встановлені пластикові вікна
	31	1	Встановлені пластикові вікна. Кімната на одну людину
	32	2	Встановлені пластикові вікна та нова гарнітура.

Рис. 2.22. Інформація про кімнати.

Для цього студенту необхідно заповнити заявку на заселення, вказавши свої дані, та номер кімнати (рис.2.23). Після чого, створена заявка буде відображатися в особистому кабінеті (рис. 2.24), після підтвердження заявки, її статус зміниться (рис.2.25).

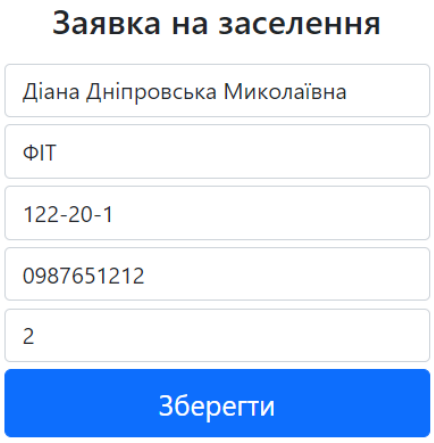
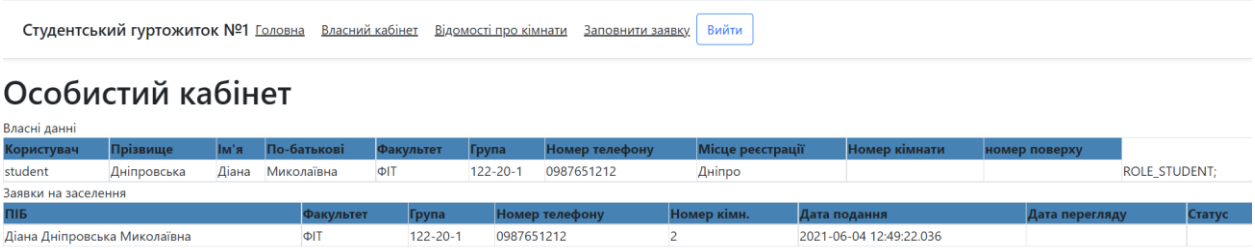


Рис. 2.23. Заявка на заселення.



Користувач	Прізвище	Ім'я	По-батькові	Факультет	Група	Номер телефону	Місце реєстрації	Номер кімнати	номер поверху	
student	Дніпровська	Діана	Миколаївна	ФІТ	122-20-1	0987651212	Дніпро			ROLE_STUDENT;

Заявки на заселення

ПІБ	Факультет	Група	Номер телефону	Номер кімн.	Дата подання	Дата перегляду	Статус
Діана Дніпровська Миколаївна	ФІТ	122-20-1	0987651212	2	2021-06-04 12:49:22.036		

Рис. 2.24. Особистий кабінет.

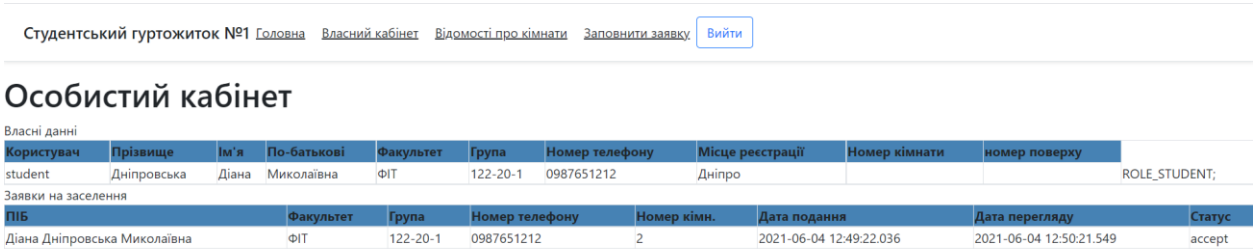


Рис. 2.25. Підтверджена заявка.

2.6.5.2 Порядок роботи для адміністратора

Для роботи адміністратора, необхідно виконати вхід до облікового запису адміністратора, після чого він зможе працювати з інформаційною системою.

З основних задач адміністратора є підтвердження зареєстрованих користувачів (рис.2.26-2.27). На цій вкладці також можна виконувати пошук за параметрами.

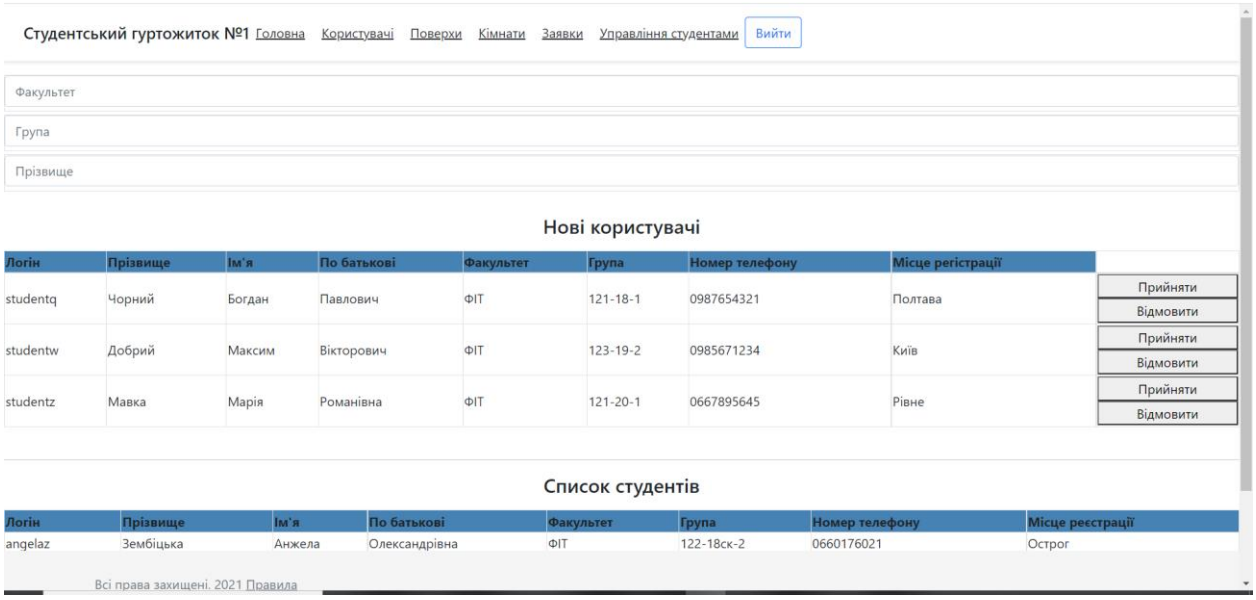


Рис. 2.26. Користувачі до підтвердження.

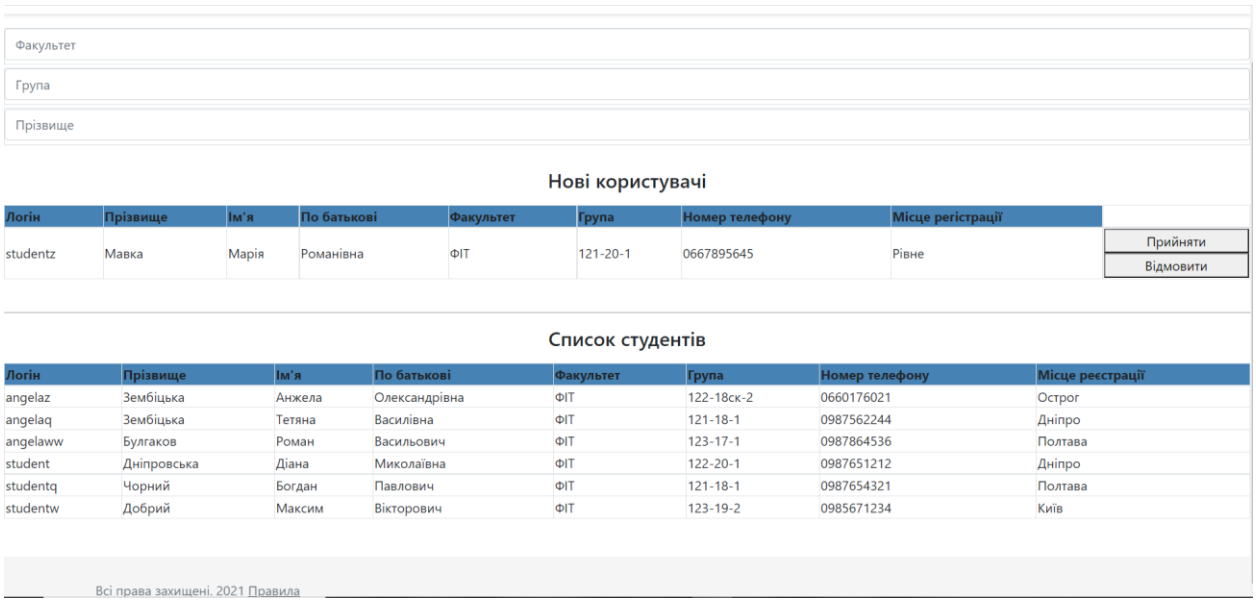


Рис. 2.27. Користувачі після підтвердження реєстрації.

Наступна задача, це робота з вкладкою «Поверхи», де за необхідності можна додавати дані про нові поверхи або видаляти застарілу інформацію (рис.2.28-2.30).

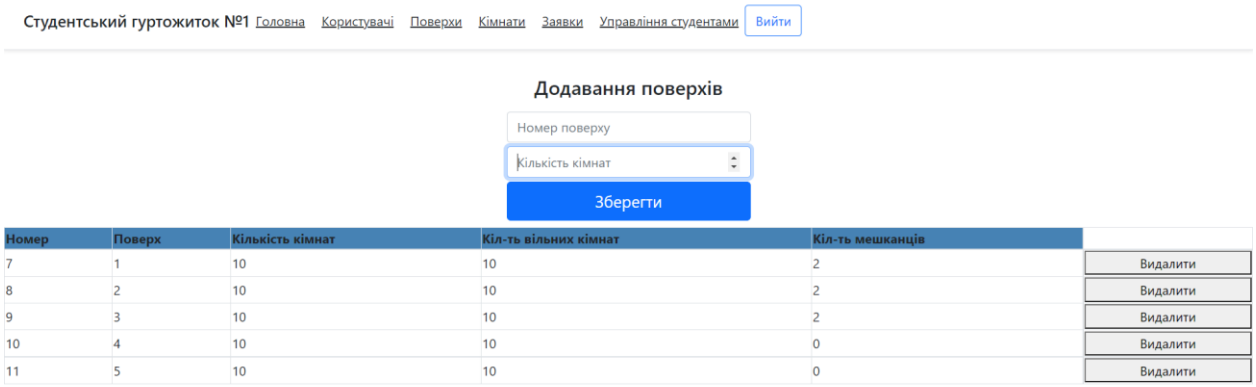


Рис. 2.28. Інформація про поверхи.

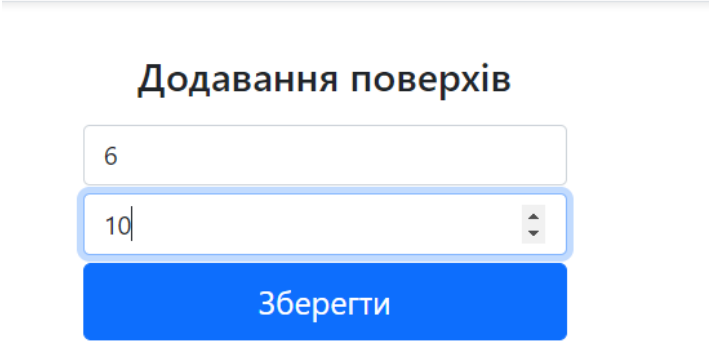


Рис. 2.29. Додавання нового поверху.

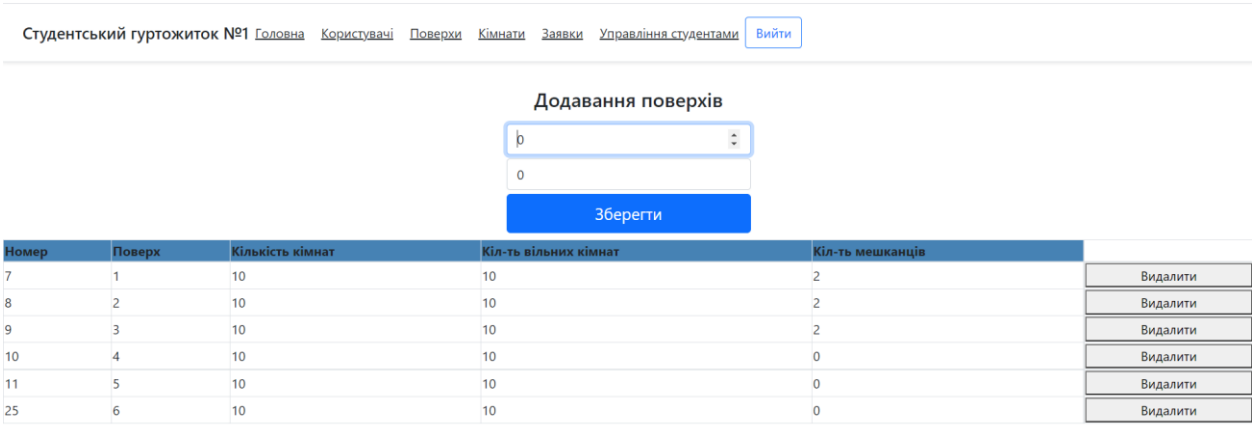


Рис. 2.30. Інформація про поверхи з новими даними.

Також, адміністратор може працювати з даними про кімнати. Переглядати, додавати нові кімнати та видаляти застарілу інформацію (рис.2.31-2.33).

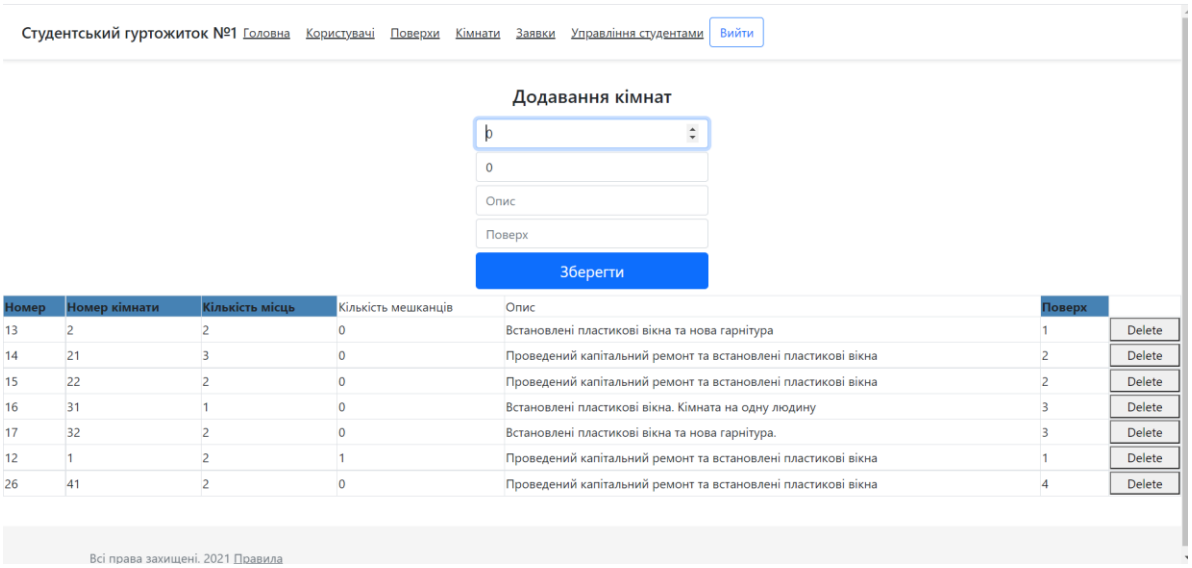


Рис. 2.31. Інформація про кімнати.

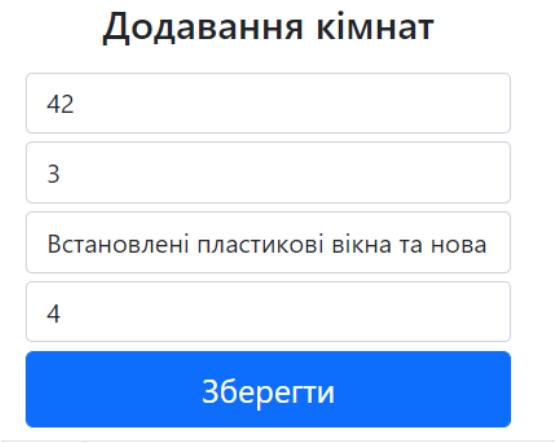


Рис. 2.32. Додавання нової кімнати.

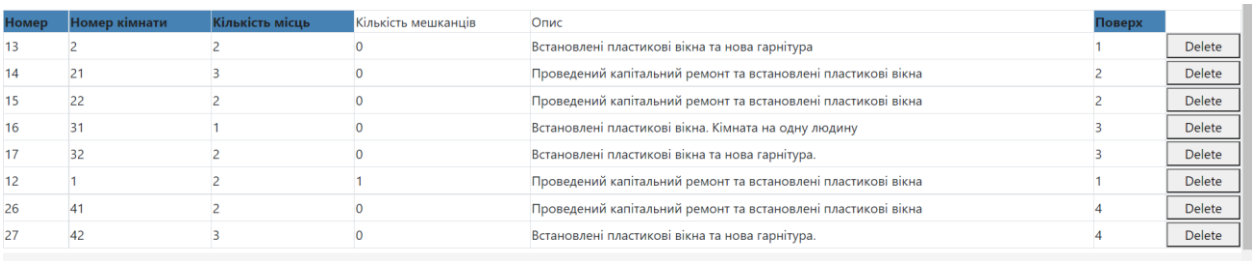


Рис. 2.33. Додана нова кімната.

На вкладці «Заявки», адміністратор може переглянути подані заявки (рис.2.34), та підтвердити або відхилити їх (рис.2.35).

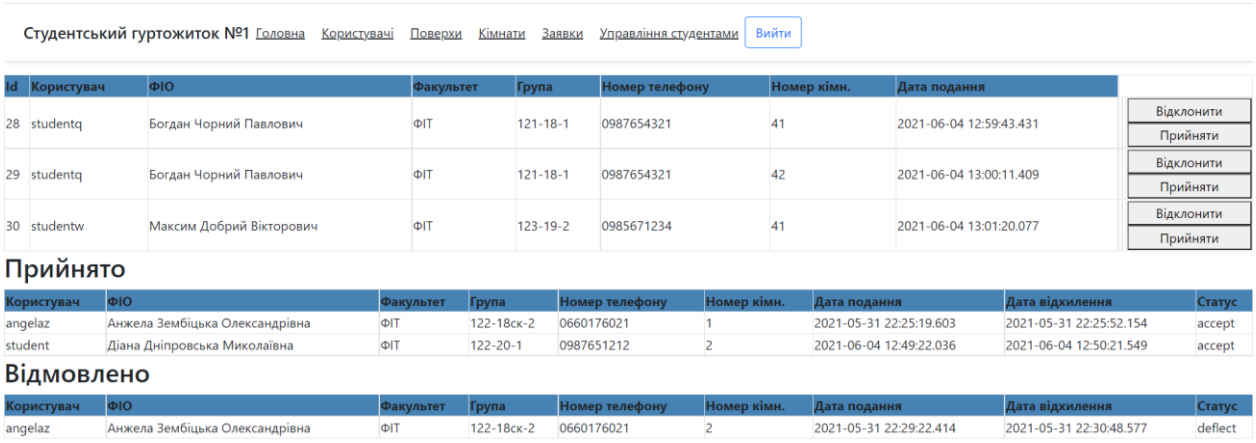


Рис. 2.34. Заявки на заселення.

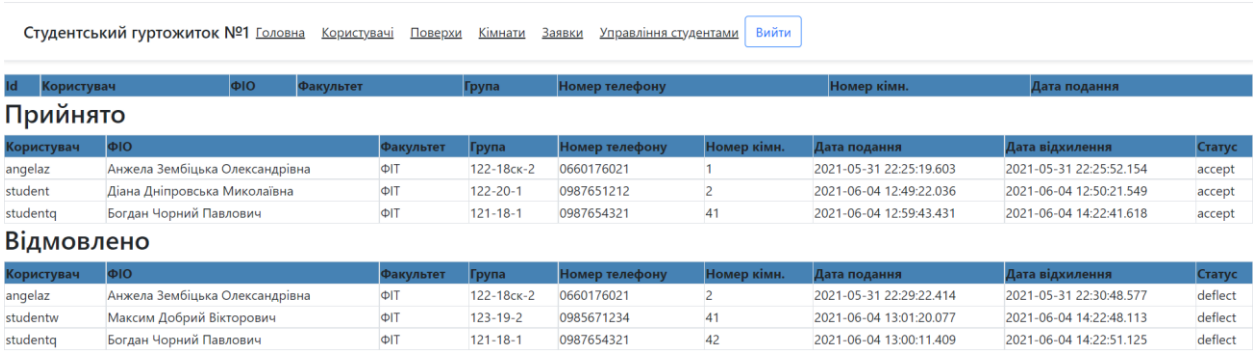


Рис. 2.35. Заявки на заселення після розгляду.

На вкладці «Управління студентами», адміністратор може заселяти студентів, вказавши їм номер кімнати, або виселяти їх (рис.2.36-2.37). Або виконувати пошук за параметрами (рис.2.38).

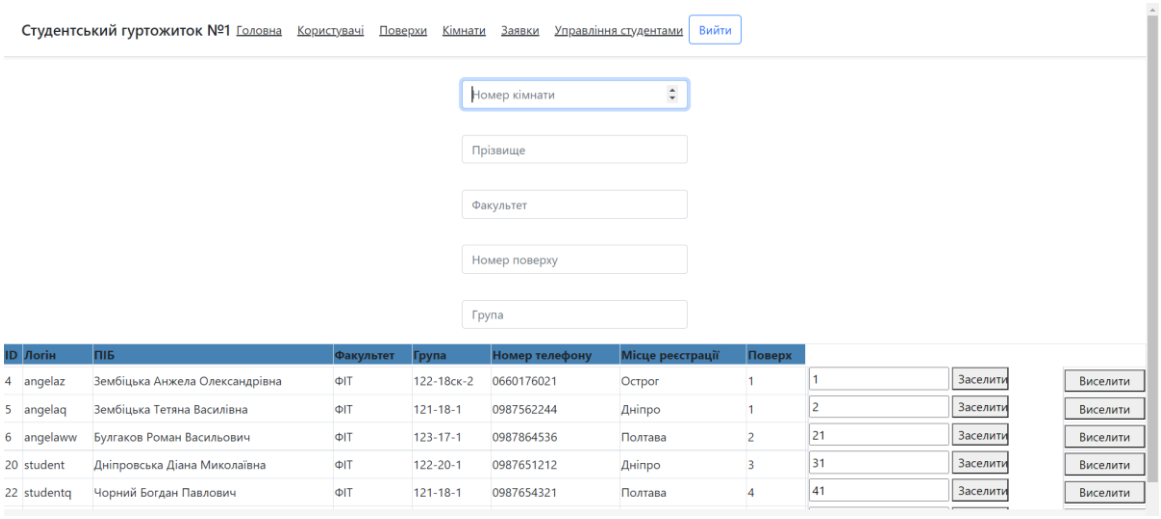


Рис. 2.36. Управляння студентами.

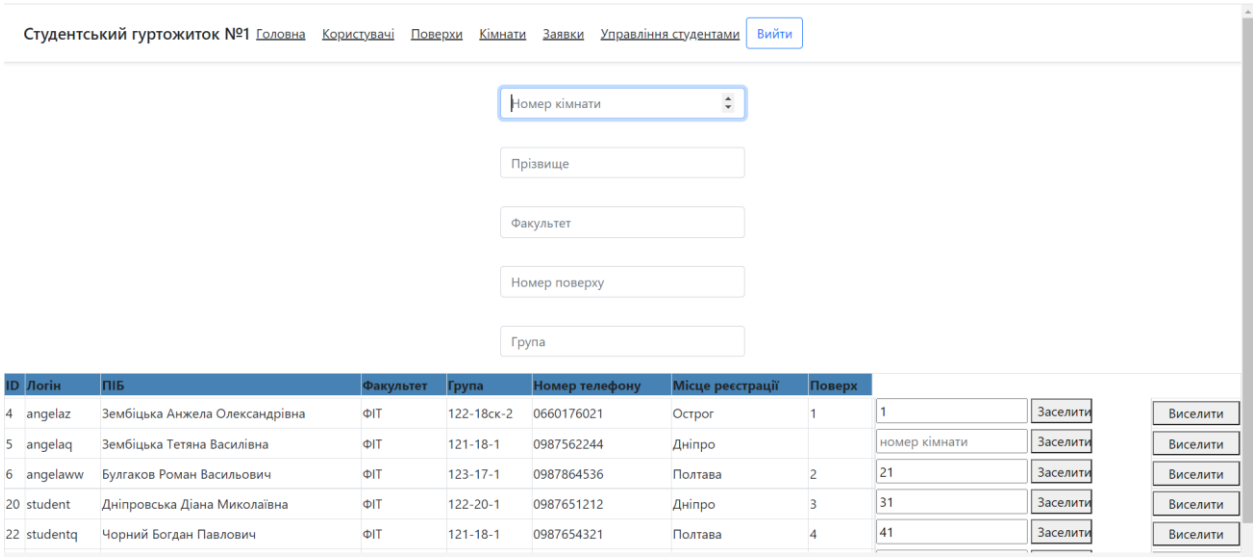


Рис. 2.37. Виселення студента під номером 5.

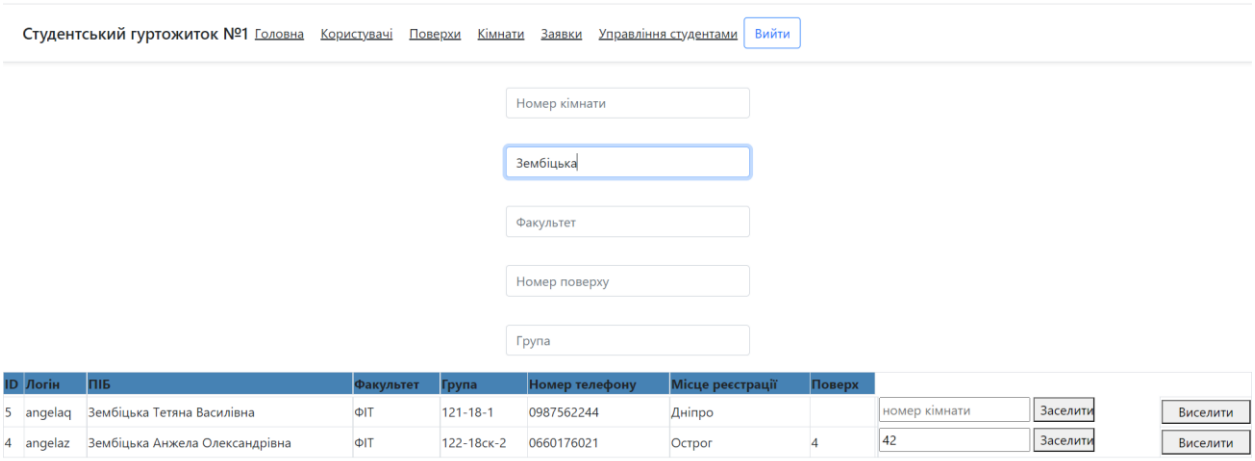


Рис. 2.38. Пошук за прізвищем «Зембіцька».

Після завершення роботи, адміністратор може натиснути «Вихід», або закрити вкладку сайту.

РОЗДІЛ 3
ЕКОНОМІЧНИЙ РОЗДІЛ

3.1 Розрахунок трудомісткості та вартості розробки програмного продукту

Вхідні дані:

– Передбачуване число операторів – 1060;

– Коефіцієнт складності програми – 1,5;

– Коефіцієнт корекції програми в ході її розробки – 0,05;

– Годинна заробітна плата програміста, грн/год – 150.

Нормування праці в процесі створення ПЗ істотно ускладнено в силу творчого характеру праці програміста. Тому трудомісткість розробки ПЗ може

бути розрахована на основі системи моделей з різною точністю оцінки.

Трудомісткість розробки ПЗ можна розрахувати за формулою:

$$t = t_u + t_a + t_n + t_{o\ тл} + t_д, \text{ людино-годин,} \quad (3.1)$$

де:

to - витрати праці на підготовку й опис поставленої задачі (приймається 50);

ti - витрати праці на дослідження алгоритму рішення задачі;

ta - витрати праці на розробку блок-схеми алгоритму;

tп - витрати праці на програмування по готовій блок-схемі;

totл - витрати праці на налагодження програми на ЕОМ;

tд - витрати праці на підготовку документації.

Складові витрати праці визначаються через умовне число операторів у ПЗ, яке розробляється.

Умовне число операторів (підпрограм):

$$Q = q \times C \times (1+ p) \text{ , людино-годин,} \quad (3.2)$$

де q - передбачуване число операторів;

C - коефіцієнт складності програми;

p - коефіцієнт кореляції програми в ході її розробки.

$$Q = 1060 \cdot 1,5 \cdot (1+ 0,05) = 1670 \text{ людино-годин.} \quad (3.3)$$

Витрати праці на вивчення опису задачі ti визначається з урахуванням уточнення опису і кваліфікації програміста:

$$t_u = \frac{QB}{(75..85)K}, \text{ людино-годин,} \quad (3.4)$$

де B - коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі; B=1.2 ... 1.5;

k - коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності.

Витрати праці на розробку алгоритму рішення задачі:

$$t_u = \frac{1670*1,2}{80*0,8} = 31, \text{ людино-годин.} \quad (3.5)$$

Витрати на складання програми по готовій блок-схемі:

$$t_\alpha = \frac{Q}{(20..25)K} \text{ людино-годин.} \quad (3.6)$$

$$t_a = \frac{1670}{20*0,8} = 104 \text{ людино-годин.} \quad (3.7)$$

Витрати на складання програми по готовій блок-схемі:

$$t_n = \frac{Q}{(20..25)K} \text{ людино-годин.} \quad (3.8)$$

$$t_n = \frac{1670}{25*0,8} = 53 \text{людино-годин.} \quad (3.9)$$

Витрати праці на налагодження програми на ЕОМ:

$$t_{\text{отл}} = \frac{Q}{(4..5)K} \text{ людино-годин.} \quad (3.10)$$

$$t_{\text{отл}} = \frac{1670}{4*0,8} = 521 \text{ людино-годин.} \quad (3.11)$$

за умови комплексного налагодження завдання:

$$t^k_{\text{отл}} = 1,2 * t_{\text{отл}}; \quad (3.12)$$

$$t_{\text{отл}} = 521*1,2 = 625,2 \quad (3.13)$$

Витрати праці на підготовку документації:

$$t_д = t_{др}+t_{до}\text{ , людино-годин,} \quad (3.14)$$

де tдр - трудомісткість підготовки матеріалів і рукопису.

$$t_{др} = \frac{Q}{(15..20)K}\text{ , людино-годин.} \quad (3.15)$$

$$t_{др} = \frac{1670}{15*0,8} = 139 \text{ людино-годин,} \quad (3.16)$$

tдо - трудомісткість редагування, печатки й оформлення документації

$$t_{до} = 0{,}75 \cdot t_{др}\text{ , людино-годин.} \quad (3.17)$$

$$t_{do} = 0{,}75 \cdot 139 = 104 \quad (3.18)$$

$$t_{д} = 139+104 = 243 \text{ людино-годин.} \quad (3.19)$$

Отримуємо трудомісткість розробки програмного забезпечення:

$$t = 31+104+53+625+243 = 1056 \text{ людино-годин.} \quad (3.20)$$

3.2. Витрати на створення програмного забезпечення

Витрати на створення ПЗ Кпо включають витрати на заробітну плату виконавця програми Зз/п і витрат машинного часу, необхідного на налагодження програми на ЕОМ:

$$K_{ПО} = З_{ЗП} + З_{МВ}\text{ , грн,} \quad (3.21)$$

Заробітна плата виконавців визначається за формулою:

$$З_{ЗП} = t \cdot C_{ПР}\text{ , грн,} \quad (3.22)$$

де:

t - загальна трудомісткість, людино-годин;

Спр - середня годинна заробітна плата програміста, грн/година

$$З_{ЗП} = 1056 \cdot 150 = 158\,400 \text{грн.} \quad (3.23)$$

Вартість машинного часу, необхідного для налагодження програми на ЕОМ:

$$З_{МВ} = тотл \times С_М \text{ , грн,} \quad (3.24)$$

де:

тотл - трудомісткість налагодження програми на ЕОМ, год.

См - вартість машино-години ЕОМ, 7 грн/год.

Визначені в такий спосіб витрати на створення програмного забезпечення є частиною одноразових капітальних витрат на створення інформаційної системи.

$$З_{МВ} = 625 \times 7 = 4375 \text{ грн.} \quad (3.25)$$

$$Кпо = 158\ 400 + 4375 = 162775 \text{ грн.} \quad (3.26)$$

Очікуваний період створення ПЗ:

$$T = \frac{t}{B_k * F_p} \text{міс.} \quad (3.27)$$

де:

Bk - число виконавців;

Fp - місячний фонд робочого часу (при 40 годинному робочому тижні Fp=176 годин).

$$T = \frac{1056}{1*176} = 6 \text{міс.} \quad (3.28)$$

Висновки Визначено трудомісткість розробленої інформаційної системи (1056 люд-год), проведений підрахунок вартості роботи по створенню програми (162775 грн.) та розраховано час на його створення (6 міс).

ВИСНОВКИ

Метою кваліфікаційної роботи було створення клієнтської частини системи обліку процесу розселення студентів у гуртожитку, що може зацікавити вищі навчальні заклади, як ресурс прискорення та полегшення процесу розселення студентів.

Основна задача розроблюваного проекту полягає в автоматизації роботи системи гуртожитку, тобто прискорення та полегшення роботи адміністрації.

Система дає можливість студентам заносити свої дані, та обирати кімнати для проживання. А адміністрації гуртожитку залишиться лише підтвердити дані студента та заявку на заселення.

Головна цінність розробленої системи полягає в тому, що вона зберігає актуальні дані про стан заселення, вільні кімнати, де можна через функції пошуку знаходити необхідну інформацію, що прискорить процес заселення, та не буде створювати проблем накладки даних.

Клієнтська частина системи була розроблена за допомогою таких ресурсів, як HTML, CSS та Bootstrap.

Під час виконання даної дипломної роботи були виконані наступні задачі:

1. Ознайомлення з теорією правил розселення студентів.
2. Розглянуті наявні у відкритому доступі аналоги.
3. Описані алгоритми функціонування та послідовність роботи.
4. Створена клієнтська частина додатку зі зручним інтерфейсом.

Створена інформаційна система має наступні функції:

– виведення інформації про гуртожиток та поверхи;

– виведення розгорнутої інформації про кімнати – кількість місць, кількість вільних місць, опис кімнати;

– реєстрація/авторизація користувачів;

– вибір кімнати та заповнення заявок на заселення до кімнати;

– редагування адміністратором даних;

Цінність програми для вищих навчальних закладів є в прискорені процесу розселення студентів на початку навчального року, можливості уникнути помилок при розподіленні кімнат, та унеможливлює накладки даних.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. ДСТУ 2394-94 Інформація та документація. Комплектування фонду, бібліографічний опис, аналіз документів. Терміни та визначення. – Чинний від 01.01.1995. — Київ : Держстандарт України, 1994. – 88 с.

2. Методичні вказівки з виконання економічного розділу в дипломних проектах студентів спеціальності “Комп’ютерні системи ” / Укладачі О.Г. Вагонова, Нікітіна О.Б. Н.Н. Романюк – Дніпропетровськ: Національний гірничий університет. – 2013. – 23с.

3. Методичні рекомендації до виконання кваліфікаційних робіт бакалаврів напряму підготовки 122 «Комп’ютерні науки» галузі знань 12 Інформаційні технології/, Л.М. Коротенко , О.С. Шевцова; Нац. гірн. ун-т. – Д : ДВНЗ НГУ, 2021. – 50 с.

4. Білл Кеннеді, Чак Муссіано - "HTML і XHTML. Детальний керівництво (HTML & HXTML. The Definitive Guide)"

5. Ерік Мейер - "CSS-каскадні таблиці стилів. Детальний керівництво (Cascading Style Sheets: The Definitive Guide)"

6. К. Шмітт - "CSS. Рецепти програмування (CSS: Cookbook)"

7. Б. Хенік - "HTML і CSS. Шлях до досконалості (HTML і CSS: The Good Parts)"

8. Документація з Bootstrap 4 на російській мові / URL - https://bootstrap-4.ru/

9. Ознайомлення з Bootstrap 4. Встановлення та початок роботи / URL - https://twbs.docs.org.ua/getting-started/

10. Справочник по HTML / URL - http://htmlbook.ru/html

11. Справочник по CSS / URL - http://htmlbook.ru/css

12. Опис аналогу додатку TSU.Helper / URL - http://csi.tsu.ru/ru/content/tsuhelper

13. Іван Панченко. PostgreSQL: вчора, сьогодні, завтра. Відкриті системи. СУБД, № 03, 2015 – 238с.

14. Глушаков С. Програмування на Java 2. Харків: Фоліо, 2003. - 536 с.

15. Вайнман Л, Вайнман В. Креативний Web-дизайн на HTML 4. Видавництво «ДиаСофт», 2003. – 562с.

16. Перрі, Б. У. Java сервлети і JSP. Збірник рецептів. М.: Кудіц-прес, 2009. 768 с.

17. Николайчук Я. М. Проектування спеціалізованих комп'ютерних систем навч. посібник / Я. М. Николайчук, Н. Я. Возна, І. Р. Пітух. – Тернопіль ТзОВ "Терно-граф", 2010. - 392 с.

18. Maven – Introduction / URL - http://maven.apache.org/what-is-maven.html

19. Spring Framework – Основи / URL - https://javarush.ru/groups/posts/spring-framework-java-1.

20. Spring Data / URL - https://spring.io/projects/spring-data.

ДОДАТОК А

КОД ПРОГРАМИ

First.jsp

```
<%@ taglib prefix="sec" uri="http://www.springframework.org/security/tags" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ page language="java" contentType="text/html; charset=utf-8" pageEncoding="utf-8"%>

<!DOCTYPE HTML>
<html>
<head>
    <title>Главная</title>
    <jsp:include page="../blocks/head.jsp"/>
</head>
<body>
<jsp:include page="../blocks/header.jsp"/>

    <h1 class="h4 mb-3 font-weight-center text-center">ВІДОМОСТІ ПРО КІМНАТИ</h1>
    <table class="border justify-content-center table-striped">
        <thead>
        <th class="border">Поверх</th>
        <th class="border">Кімната</th>
        <th class="border">Вільних місць</th>
        <th class="border">Опис</th>
        </thead>
        <c:forEach items="${freeRoomForAll}" var="rooms">
            <tr class="table-striped">
                <td class="border">${rooms.floors.numberFloor}</td>
                <td class="border">${rooms.numberRoom}</td>
                <td class="border">${rooms.numberOfFreePlacesInTheRoom}</td>
                <td class="border">${rooms.description}</td>
            </tr>
        </c:forEach>
```

```
    </table>
</div>
<jsp:include page="../blocks/footer.jsp"/>
</body>
</html>
```

Admin.jsp

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>
<%@ taglib prefix="th" uri="http://www.springframework.org/tags/form" %>
<%@ page language="java" contentType="text/html; charset=utf-8" pageEncoding="utf-8"%>

<!DOCTYPE html>
<html>
<head>
    <jsp:include page="../blocks/head.jsp"/>
    <jsp:include page="../blocks/header.jsp"/>
    <meta charset="utf-8">

    <title>Log in with your account</title>
<%-- <link rel="stylesheet" type="text/css" href="${contextPath}/resources/css/style.css">--%>
</head>

<body>
<form th:action="@{controlResidents}" method="get" class="mb-1 border form-inline">
    <td class="tha"><!--/*@thymesVar id="faculty" type="actions"*/-->
        <input type="text" name="facultyForFilter" th:value="${faculty}" class="form-control ml-1" placeholder="Факультет"></td>
</form>
<form th:action="@{controlResidents}" method="get" class="mb-1 border form-inline">
    <td class="tha"><!--/*@thymesVar id="groupIn" type="actions"*/-->
```

```
        <input   type="text"   name="groupInForFilter"   th:value="${groupIn}"
class="form-control" placeholder="Група"></td></form>

<form th:action="@{controlResidents}" method="get" class="mb-1 border form-
inline">

    <td class="tha"><!--/*@thymesVar id="surname" type="actions"*/-->

        <input   type="text"   name="surnameForFilter"   th:value="${surname}"
class="form-control" placeholder="Прізвище"></td>

</form>

<br>

<h1 class="h4 mb-3 font-weight-center text-center">Нові користувачі</h1>


    <table class="border justify-content-center table-striped">

        <thead>

        <th  class="border">Логін</th>

        <th  class="border">Прізвище</th>

        <th  class="border">Ім'я</th>

        <th  class="border">По батькові</th>

        <th  class="border">Факультет</th>

        <th  class="border">Група</th>

        <th  class="border">Номер телефону</th>

        <th  class="border">Місце регістрації</th>

        </thead>

        <c:forEach items="${unverifiedUsers}" var="users">

            <tr>

                <td class="border">${users.username}</td>

                <td class="border">${users.residents.surname}</td>

                <td class="border">${users.residents.name}</td>

                <td class="border">${users.residents.lastname}</td>

                <td class="border">${users.residents.faculty}</td>

                <td class="border">${users.residents.groupIn}</td>

                <td class="border">${users.residents.phoneNumber}</td>

                <td class="border">${users.residents.registration}</td>

                <td class="border">

                    <form   action="${pageContext.request.contextPath}/admin"
method="post">

                        <form:form>
```

```
                        <input          type="hidden"          name="userId"
value="${users.id}"/>
                        <input type="hidden" name="action" value="accept"/>
                        <button type="submit">Прийняти</button></form:form>
                        <form:form>
                        <input          type="hidden"          name="userId"
value="${users.id}"/>
                        <input type="hidden" name="action" value="reject"/>
                        <button type="submit">Відмовити</button>
                        </form:form>
                    </form>
                </td>
            </tr>
        </c:forEach>
    </table>
<br>
<hr>
<h1 class="h4 mb-3 font-weight-center text-center">Список студентів</h1>
<table  class="border justify-content-center table-striped">
<thead>
<th class="border">Логін</th>
<th class="border">Прізвище</th>
<th class="border">Ім'я</th>
<th class="border">По батькові</th>
<th class="border">Факультет</th>
<th class="border">Група</th>
<th class="border">Номер телефону</th>
<th class="border">Місце реєстрації</th>
</thead>
<c:forEach items="${studentUsers}" var="users">
    <tr>
    <td class="border">${users.username}</td>
    <td class="border">${users.residents.surname}</td>
    <td class="border">${users.residents.name}</td>
    <td class="border">${users.residents.lastname}</td>
```

```
    <td class="border">${users.residents.faculty}</td>
    <td class="border">${users.residents.groupIn}</td>
    <td class="border">${users.residents.phoneNumber}</td>
    <td class="border">${users.residents.registration}</td>
    </tr>
    </c:forEach>
</table>

</div>
<jsp:include page="../blocks/footer.jsp"/>
</body>
</html>
```

Applications.jsp

```
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="input" uri="http://www.springframework.org/tags/form" %>
<%@    page    language="java"    contentType="text/html;    charset=utf-8"
pageEncoding="utf-8"%>

<!DOCTYPE html>
<html>
<jsp:include page="../blocks/head.jsp"/>

<body>
<jsp:include page="../blocks/header.jsp"/>
<div>
    </form>

        <%--    Форма для создания комнаты--%>

        <%--       вывод таблицы комнат     --%>
        <div>
            <table class="border justify-content-center table-striped">
```

```
                <thead>
                <th class="border">Id</th>
                <th class="border">Користувач</th>
                <th class="border">ФІО</th>
                <th class="border">Факультет</th>
                <th class="border">Група</th>
                <th class="border">Номер телефону</th>
                <th class="border">Номер кімн.</th>
                <th class="border">Дата подання</th>

                </thead>
                ${forAdminMessage}
                <c:forEach                    items="${unverifiedApplications}"
var="applications">
                    <tr>
                        <td class="border">${applications.id}</td>
                        <td
class="border">${applications.users.username}</td>
                        <td
class="border">${applications.nameSurnameLastname}</td>
                        <td class="border">${applications.faculty}</td>
                        <td class="border">${applications.groupIn}</td>
                        <td class="border">${applications.phoneNumber}</td>
                        <td
class="border">${applications.rooms.numberRoom}</td>
                        <td class="border">${applications.creationDate}</td>
                            <%--                 Исправить    --%>
                        <td
class="border">${rooms.applicationsForAccommodation.nameSurnameLastname}</td>
                            <%--                 Исправить    --%>
                        <td class="border">${rooms.floors.numberFloor}</td>
                        <td class="border">

                                <%--                          Форма для Комнаты
по текущему Id в строке--%>
                            <form
action="${pageContext.request.contextPath}/applications" method="post">
```

```
                                <form:form>
                                <input   type="hidden"   name="applicationId"
value="${applications.id}"/>
                                <input       type="hidden"       name="action"
value="deflect"/>
                                <button type="submit">Відклонити</button>
                                </form:form>
                                <form:form>
                                    <input type="hidden" name="applicationId"
value="${applications.id}"/>
                                    <input     type="hidden"     name="action"
value="accept"/>
                                    <button type="submit">Прийняти</button>
                                </form:form>
                            </form>
                        </td>
                    </tr>
                </c:forEach>
            </table>

<table class="border justify-content-center table-striped">
            <h2>Прийнято</h2>
            <thead>
            <th class="border">Користувач</th>
            <th class="border">ФІО</th>
            <th class="border">Факультет</th>
            <th class="border">Група</th>
            <th class="border">Номер телефону</th>
            <th class="border">Номер кімн.</th>
            <th class="border">Дата подання</th>
            <th class="border">Дата відхилення</th>
            <th class="border">Статус</th>
            </thead>
            ${forAdminMessage}
            <c:forEach items="${acceptApplications}" var="applications">
                <tr>
```

```
                    <td class="border">${applications.users.username}</td>
                    <td
class="border">${applications.nameSurnameLastname}</td>
                    <td class="border">${applications.faculty}</td>
                    <td class="border">${applications.groupIn}</td>
                    <td class="border">${applications.phoneNumber}</td>
                    <td class="border">${applications.rooms.numberRoom}</td>
                    <td class="border">${applications.creationDate}</td>
                    <td class="border">${applications.dateOfChange}</td>
                    <td class="border">${applications.status}</td>

                </tr>
            </c:forEach>
            </table>



<table class="border justify-content-center table-striped">
    <h2>Відмовлено</h2>
<thead>
<th class="border">Користувач</th>
<th class="border">ФІО</th>
<th class="border">Факультет</th>
<th class="border">Група</th>
<th class="border">Номер телефону</th>
<th class="border">Номер кімн.</th>
<th class="border">Дата подання</th>
<th class="border">Дата відхилення</th>
<th class="border">Статус</th>
</thead>
${forAdminMessage}
<c:forEach items="${deflectApplications}" var="applications">
    <tr>
    <td class="border">${applications.users.username}</td>
    <td class="border">${applications.nameSurnameLastname}</td>
```

```
    <td class="border">${applications.faculty}</td>
    <td class="border">${applications.groupIn}</td>
    <td class="border">${applications.phoneNumber}</td>
    <td class="border">${applications.rooms.numberRoom}</td>
    <td class="border">${applications.creationDate}</td>
    <td class="border">${applications.dateOfChange}</td>
    <td class="border">${applications.status}</td>

    </tr>
    </c:forEach>
</table>

</div>
<jsp:include page="../blocks/footer.jsp"/>
</body>
</html>
```

applicationsForAccommodation.jsp

```
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ page language="java" contentType="text/html; charset=utf-8"
pageEncoding="utf-8"%>

<!DOCTYPE html>
<html>
<jsp:include page="../blocks/head.jsp"/>
<body>
<jsp:include page="../blocks/header.jsp"/>
<div>
    <h1>${numberOfApplicationsExceeded}</h1>
    ${applicationError}
    <form:form method="post"
modelAttribute="applicationsForAccommodationForm" class="form-signin pb-2">
        <h1 class="h4 mb-3 font-weight-center text-center">Заявка на
заселення</h1>
```

```
        <div>

            <form:input type="text" path="nameSurnameLastname" class="form-
control"   required=""   pattern="[А-Яа-яґҐЁёІіЇїЄє'’’\s-]{2,20}{2,20}{2,20}"
placeholder="Ім'я Прізвище По-батькові"></form:input>

<%--            <form:errors path="nameSurnameLastname"></form:errors>--%>

<%--                ${nameError}--%>

        </div>

        <div>

            <form:input type="text" path="faculty" placeholder="Факультет"
class="form-control" required=""></form:input>

        </div>

        <div>

            <form:input   type="text"   path="groupIn"   placeholder="Група"
class="form-control" required=""></form:input>

        </div>


        <div>

            <form:input type="text" path="phoneNumber" placeholder="Номер
телефону" pattern="[0-9]{10}" class="form-control" required=""></form:input>

        </div>

        ${roomError}

        <div>

            <input type="number" name="numberRoom" placeholder="Номер кімнати"
class="form-control" required="">

        </div>

        <button   type="submit"   class="btn   btn-lg   btn-primary   btn-
block">Зберегти</button>

    </form:form>

</div>

<jsp:include page="../blocks/footer.jsp"/>

</body>

</html>
```

ControlResidents.jsp

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>
```

```
<%@ taglib prefix="th" uri="http://www.springframework.org/tags/form" %>

<%@ page language="java" contentType="text/html; charset=utf-8"
pageEncoding="utf-8"%>


<!DOCTYPE html>

<html>

<head>

    <jsp:include page="../blocks/head.jsp"/>

</head>


<body>

<jsp:include page="../blocks/header.jsp"/>

<%--<% Long numberRoomForFilter = request.getParameterNames();  %>--%>

<%--action="/filter"--%>

<%--action=""--%>

<%--action="controlResidents?numberRoomForFilter=${numberRoomForFilter}--%>

<%--<form method="get">--%>

<%--<td><form:form>--%>

<%--      <input type="number" name="numberRoomForFilter" placeholder="номер
кімнати" >--%>

<%--              <%   Object   numberRoomForFilter   =<input   type="number"
name="numberRoomForFilter" placeholder="номер кімнати" >;--%>

<%--        pageContext.setAttribute("allStudent", numberRoomForFilter);%>--%>

<%--    <button type="submit">Пошук</button>--%>

<%--</form:form></td>--%>

<%--</form>--%>

<%--<div>--%>

<form th:action="@{controlResidents}" method="get" class="form-signin">

    <!--/*@thymesVar id="numberRoom" type="actions"*/-->

    <th><input            type="number"            name="numberRoomForFilter"
th:value="${numberRoom}" class="form-control ml-1" placeholder="Номер кімнати"
autofocus="true" class="form-control" required=""></th>

</form>

<form th:action="@{controlResidents}" method="get" class="form-signin">

        <th><!--/*@thymesVar id="surname" type="actions"*/-->

        <input   type="text"   name="surnameForFilter"   th:value="${surname}"
class="form-control ml-1" placeholder="Прізвище"></th>
```

```
</form>
<form th:action="@{controlResidents}" method="get" class="form-signin">
    <th><!--/*@thymesVar id="faculty" type="actions"*/-->
        <input type="text" name="facultyForFilter" th:value="${faculty}"
class="form-control ml-1" placeholder="Факультет"></th></form>


<form th:action="@{controlResidents}" method="get" class="form-signin">
    <th><!--/*@thymesVar id="numberFloor" type="actions"*/-->
        <input type="number" name="numberFloorForFilter"
th:value="${numberFloor}" class="form-control ml-1" placeholder="Номер
поверху"></th>
</form>
<form th:action="@{controlResidents}" method="get" class="form-signin">
    <th><!--/*@thymesVar id="groupIn" type="actions"*/-->
        <input type="text" name="groupInForFilter" th:value="${groupIn}"
class="form-control ml-1" placeholder="Група"></th></form>
<div>
    <table class="border justify-content-center table-striped">
        <thead>
        <th class="border">ID</th>
        <th class="border">Логін</th>
        <th class="border">ПІБ</th>
<%--        <th>Ім'я</th>--%>
<%--        <th>По батькові</th>--%>
        <th class="border">Факультет</th>
        <th class="border">Група</th>
        <th class="border">Номер телефону</th>
        <th class="border">Місце реєстрації</th>
        <th class="border">Поверх</th>
        </thead>
        <td class="border">${roomFulError}</td>
        <td class="border">${roomError}</td>
        <c:forEach items="${allStudent}" var="residents">
            <tr>
                <td class="border">${residents.id}</td>
                <td class="border">${residents.users.username}</td>
```

```
                <td   class="border">${residents.surname}   ${residents.name}
${residents.lastname}</td>
<%--                <td></td>--%>
<%--                <td></td>--%>
                <td class="border">${residents.faculty}</td>
                <td class="border">${residents.groupIn}</td>
                <td class="border">${residents.phoneNumber}</td>
                <td class="border">${residents.registration}</td>
                <td class="border">${residents.rooms.floors.numberFloor}</td>
                <td class="border">
                    <form
action="${pageContext.request.contextPath}/controlResidents" method="post">
                        <td><form:form>
                        <input         type="number"         name="numberRoom"
value="${residents.rooms.numberRoom}" placeholder="номер кімнати" >
                        <input         type="hidden"         name="residentId"
value="${residents.id}"/>
                        <input type="hidden" name="action" value="moveTo"/>
                        <button type="submit" class="b">Заселити</button>
                        </form:form></td>
                        <td class="border"><form:form>
                            <input       type="hidden"       name="residentId"
value="${residents.id}"/>
                            <input         type="hidden"         name="action"
value="moveOut"/>
                            <button type="submit">Виселити</button>
                        </form:form></td>
                    </form>


                </td>
            </tr>
        </c:forEach>
    </table>
</div>
<jsp:include page="../blocks/footer.jsp"/>
</body>
</html>
```

Floors.jsp

```
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ page language="java" contentType="text/html; charset=utf-8" pageEncoding="utf-8"%>

<!DOCTYPE html>
<html>
<head>
    <jsp:include page="../blocks/head.jsp"/>
</head>

<body>
<jsp:include page="../blocks/header.jsp"/>
<div>
    ${forAdmin}
<form:form method="post" modelAttribute="floorAdd" class="form-signin pb-2">
    <h1 class="h4 mb-3 font-weight-center text-center">Додавання поверхів</h1>
    <div>
    <form:input type="number" min="1" path="numberFloor" placeholder="Номер поверху" autofocus="true" class="form-control" required=""></form:input>
    <form:errors path="numberFloor"></form:errors>
            ${floorError}
    </div>
    <div>
    <form:input type="number" min="1" path="numberOfRoomsPerFloor" placeholder="Кількість кімнат" autofocus="true" class="form-control" required=""></form:input>
    </div>
<%--    <div>--%>
<%--     <form:input type="number" min="1" path="numberOfFreeRoomsOnTheFloor" placeholder="Кол. свободных комнат"></form:input>--%>
<%--    </div>--%>
    <button type="submit" value="save" class="btn btn-lg btn-primary btn-block">Зберегти</button>
    </form:form>
```

```
<%--    <form:form method="post" modelAttribute="allFloorsForm">--%>
        <div>
            <table class="border justify-content-center table-striped">
                <thead>
                <th class="border">Номер</th>
                <th class="border">Поверх</th>
                <th class="border">Кількість кімнат</th>
                <th class="border">Кіл-ть вільних кімнат</th>
                <th class="border">Кіл-ть мешканців</th>
                </thead>
                <c:forEach items="${allFloors}" var="floors">
                    <tr>
                        <td class="border">${floors.id}</td>
                        <td class="border">${floors.numberFloor}</td>
                        <td
class="border">${floors.numberOfRoomsPerFloor}</td>
                        <td
class="border">${floors.numberOfFreeRoomsOnTheFloor}</td>
                        <td class="border">${floors.rooms.size()}</td>
                        <td class="border">
                            <form
action="${pageContext.request.contextPath}/floorDelete" method="post">
                                <input      type="hidden"      name="floorId"
value="${floors.id}"/>
                                <input       type="hidden"       name="action"
value="delete"/>
                                <button type="submit">Видалити</button>
                            </form>
                        </td>
                    </tr>
                </c:forEach>
            </table>
            </div>

</div>
<jsp:include page="../blocks/footer.jsp"/>
```

```
</body>
</html>
```

Login.jsp

```
<%@ taglib prefix="sec" uri="http://www.springframework.org/security/tags" %>
<%--<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>--%>
<%@ page language="java" contentType="text/html; charset=utf-8"
pageEncoding="utf-8"%>

<!DOCTYPE html>
<html>
<head>
    <jsp:include page="../blocks/head.jsp"/>
</head>

<body>
<jsp:include page="../blocks/header.jsp"/>
<sec:authorize access="isAuthenticated()">
    <% response.sendRedirect("/"); %>
</sec:authorize>
<div>
    <form method="POST" action="/login" class="form-signin pb-2">
        <h1 class="h4 mb-3 font-weight-center text-center">Вхід</h1>
            <input name="username" type="text" placeholder="Логін "
                   autofocus="true" class="form-control" required=""/>
            <input name="password" type="password" placeholder="Пароль"
class="form-control" required=""/>
            <button type="submit" class="btn btn-lg btn-primary btn-
block">Ввійти</button>

    </form>
</div>
<jsp:include page="../blocks/footer.jsp"/>
</body>
</html>
```

News.jsp

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ page language="java" contentType="text/html; charset=utf-8" pageEncoding="utf-8"%>

<!DOCTYPE html>
<html>
<head>
    <jsp:include page="../blocks/head.jsp"/>
</head>
<body>
<jsp:include page="../blocks/header.jsp"/>
<div>
    <h1 class="h4 mb-3 font-weight-center text-center">Відомості про кімнати</h1>
    <table class="border justify-content-center table-striped">
        <thead>
        <th class="border">Поверх</th>
        <th class="border">Номер кімнати</th>
        <th class="border">Вільних місць</th>
        <th class="border">Опис</th>
        </thead>
        <c:forEach items="${freeRoomForAuthorized}" var="rooms">
            <tr>
                <td class="border">${rooms.floors.numberFloor}</td>
                <td class="border">${rooms.numberRoom}</td>
                <td class="border">${rooms.numberOfFreePlacesInTheRoom}</td>
                <td class="border">${rooms.description}</td>
            </tr>
        </c:forEach>
    </table>
</div>
<jsp:include page="../blocks/footer.jsp"/>
</body>
```

```
</html>
```

personalArea.jsp

```
<%@ taglib prefix="sec" uri="http://www.springframework.org/security/tags" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ page language="java" contentType="text/html; charset=utf-8"
pageEncoding="utf-8"%>

<!DOCTYPE HTML>
<html>
<jsp:include page="../blocks/head.jsp"/>
<body>
<jsp:include page="../blocks/header.jsp"/>
<div>
    <h1>Особистий кабінет</h1>
    <table class="border justify-content-center table-striped">
        <h>Власні данні</h>
        <thead>
        <th class="border">Користувач</th>
        <th class="border">Прізвище</th>
        <th class="border">Ім'я</th>
        <th class="border">По-батькові</th>
        <th class="border">Факультет</th>
        <th class="border">Група</th>
        <th class="border">Номер телефону</th>
        <th class="border">Місце реєстрації</th>
        <th class="border">Номер кімнати</th>
        <th class="border">номер поверху</th>

        </thead>
        <c:forEach items="${userInfo}" var="resident">
            <tr>
                <td class="border">${resident.users.username}</td>
                <td class="border">${resident.surname}</td>
```

```
                <td class="border">${resident.name}</td>
                <td class="border">${resident.lastname}</td>
                <td class="border">${resident.faculty}</td>
                <td class="border">${resident.groupIn}</td>
                <td class="border">${resident.phoneNumber}</td>
                <td class="border">${resident.registration}</td>
                <td class="border">${resident.rooms.numberRoom}</td>
                <td class="border">${resident.rooms.floors.numberFloor}</td>
                <td>
                    <c:forEach                items="${resident.users.roles}"
var="role">${role.name}; </c:forEach>
                </td>
            </tr>
        </c:forEach>
    </table>

    <table class="border justify-content-center table-striped">
        <h>Заявки на заселення</h>
        <thead>
        <th class="border">ПІБ</th>
        <th class="border">Факультет</th>
        <th class="border">Група</th>
        <th class="border">Номер телефону</th>
        <th class="border">Номер кімн.</th>
        <th class="border">Дата подання</th>
        <th class="border">Дата перегляду</th>
        <th class="border">Статус</th>
        </thead>
        <c:forEach items="${userApplicatonsList}" var="applications">
            <tr>
                <td class="border">${applications.nameSurnameLastname}</td>
                <td class="border">${applications.faculty}</td>
                <td class="border">${applications.groupIn}</td>
                <td class="border">${applications.phoneNumber}</td>
                <td class="border">${applications.rooms.numberRoom}</td>
```

```
                <td class="border">${applications.creationDate}</td>
                <td class="border">${applications.dateOfChange}</td>
                <td class="border">${applications.status}</td>
            </tr>
        </c:forEach>
    </table>
</div>
<jsp:include page="../blocks/footer.jsp"/>
</body>
</html>
```

Registration.jsp

```
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ page language="java" contentType="text/html; charset=utf-8"
pageEncoding="utf-8"%>

<!DOCTYPE html>
<html>
<head>
    <jsp:include page="../blocks/head.jsp"/>
</head>
<body>
<jsp:include page="../blocks/header.jsp"/>
<div>

        <form:form method="post" modelAttribute="userForm" class="form-signin
pb-2">
            <h1 class="h4 mb-3 font-weight-center text-center">Реєстрація</h1>
        <div>
            <form:input type="text" path="username" placeholder="Логін"
                        autofocus="true"   class="form-control"   required=""
pattern="[a-z]{5,20}"></form:input>
            <form:errors path="username"></form:errors>
                ${usernameError}
```

```
        </div>
        <div>
            <form:input type="password" path="password" placeholder="Пароль"
class="form-control" required="" autofocus="" minlength="8" pattern="[0-9a-zA-
Z]{8,}"></form:input>
        </div>
        <div>
            <form:input type="password" path="passwordConfirm"
                        placeholder="Повторіть пароль" class="form-control"
required="" autofocus="" minlength="8" pattern="[0-9a-zA-Z]{8,}"></form:input>
            <form:errors path="password"></form:errors>
                ${passwordError}
        </div>
            <div>
                <input type="text" name="name" placeholder="Ім'я" class="form-
control" required="" autofocus="" pattern="[А-Яа-яґҐЁёІіЇїЄє'’\s-]{2,20}">
            </div>
            <div>
                <input type="text" name="lastname" placeholder="По-Батькові"
class="form-control" required="" autofocus="" pattern="[А-Яа-яґҐЁёІіЇїЄє'’\s-
]{2,20}">
            </div>
            <div>
                <input type="text" name="surname" placeholder="Прізвище"
class="form-control" required="" autofocus="" pattern="[А-Яа-яґҐЁёІіЇїЄє'’\s-
]{2,20}">
            </div>
            <div>
                <input type="text" name="faculty" placeholder="Факультет"
class="form-control" required="" autofocus="">
            </div>
            <div>
                <input type="text" name="groupIn" placeholder="Група"
class="form-control" required="" autofocus="">
            </div>
            <div>
                <input type="text" name="phoneNumber" placeholder="Номер
телефону" class="form-control" required="" autofocus="" pattern="[0-9]{10}">
            </div>
```

```
            <div>
                <input type="text" name="registration" placeholder="Місце реєстрації" class="form-control" required="" autofocus="" pattern="[А-Яа-яґҐЁёІіЇїЄє'’’\s-]{2,20}">
            </div>

            <button type="submit" class="btn btn-lg btn-primary btn-block">Зареєструватися</button>
    </form:form>
</div>
<jsp:include page="../blocks/footer.jsp"/>
</body>
</html>
```

Room.jsp

```
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="input" uri="http://www.springframework.org/tags/form" %>
<%@ page language="java" contentType="text/html; charset=utf-8" pageEncoding="utf-8"%>

<!DOCTYPE html>
<html>
<head>
    <jsp:include page="../blocks/head.jsp"/>
    <title>Кімнати</title>
</head>

<body>
<jsp:include page="../blocks/header.jsp"/>
<div>

<%--    Форма для создания комнаты--%>
    <form method="post" action="addRoom" class="form-signin pb-2">
        <form:form modelAttribute="roomAdd">
```

```
            <h1  class="h4  mb-3  font-weight-center  text-center">Додавання кімнат</h1>
        <div>
            <form:input  type="number"  path="numberRoom"  placeholder="Номер кімнати" autofocus="true" class="form-control" required=""></form:input>
            <form:errors path="numberRoom"></form:errors>
                ${roomError}
                ${floorError}
        </div>
        <div>
            <form:input      type="number"      path="numberOfSeatsInTheRoom" placeholder="Кількість    місць"    autofocus="true"    class="form-control" required=""></form:input>
        </div>
<%--        <div>--%>
<%--             <form:input type="number" path="numberOfFreePlacesInTheRoom" placeholder="Кол. свободных мест"></form:input>--%>
<%--        </div>--%>
            <div>
                <form:input type="text" path="description" placeholder="Опис" autofocus="true" class="form-control" required=""></form:input>
            </div>
            <div>
                <input type="number" name="numberFloor" placeholder="Поверх" autofocus="true" class="form-control" required="">
            </div>
            <button type="submit" value="save" class="btn btn-lg btn-primary btn-block">Зберегти</button>
        </form:form>
    </form>
    <div>
        <table class="border justify-content-center table-striped">
            <thead>
            <th class="border">Номер</th>
            <th class="border">Номер кімнати</th>
            <th class="border">Кількість  місць</th>
            <td class="border">Кількість мешканців</td>
            <td class="border">Опис</td>
```

```
            <!--<td class="border">Заявки</td>-->
            <th class="border">Поверх</th>

            </thead>
            ${forAdminMessage}
            <c:forEach items="${allRoom}" var="rooms">
                <tr>

                    <td class="border">${rooms.id}</td>
                    <td class="border">${rooms.numberRoom}</td>
                    <td class="border">${rooms.numberOfSeatsInTheRoom}</td>
                    <td class="border">${rooms.residents.size()}</td>
                    <td class="border">${rooms.description}</td>
                    <%--<c:forEach
items="${rooms.applicationsForAccommodation}" var="app">
                        <td class="border">${app.nameSurnameLastname}</td>
                    </c:forEach>--%>
                    <td class="border">
<%--            вывод таблицы комнат            --%>
<%--                        Форма для удаления Комнаты по текущему Id в строке-
-%>
                        <form
action="${pageContext.request.contextPath}/room" method="post">
                            <input         type="hidden"         name="roomId"
value="${rooms.id}"/>
                            <input         type="hidden"         name="action"
value="delete"/>
                            <button type="submit">Delete</button>
                        </form>
            </c:forEach>
        </table>
    </div>
</div>
<jsp:include page="../blocks/footer.jsp"/>
</body>
```

```
</html>
```

Main.css

```
.form-signin {
    width: 100%;
    max-width: 330px;
    padding: 15px;
    margin: auto;
}
.footer{
    position: fixed;

    bottom: 0;
    width: 100%;
    height: 50px;
    line-height: 50px;
    background-color: #f5f5f5;
}
#errorBlock{
    display: none;
}

table{
    border: 1px;
    width: 100%;
}
th{
    background:#4682B4;
}
button{
    width: 100%;
}
.b{
    width:75px;
```

```
}
input{
    margin-bottom: 5px;
}
.tha{
    width: 33%;
}
```

ДОДАТОК Б

ВІДГУК КЕРІВНИКА ЕКОНОМІЧНОГО РОЗДІЛУ

ДОДАТОК В

ПЕРЕЛІК ФАЙЛІВ НА ДИСКУ

Ім’я файлу	**Опис**
Пояснювальні документи	
Диплом_Зембіцька.doc.	Пояснювальна записка до кваліфікаційної роботи. Документ Word.
Диплом_Зембіцька.pdf.	Пояснювальна записка до кваліфікаційної роботи в форматі PDF.
Програма	
StudentHotel.rar	Архів. Містить коди програми і откомпільовану програму.
Презентація	
Презентація_Зембіцька.ppt	Презентація кваліфікаційної роботи.