

Міністерство освіти і науки України
Львівський національний університет імені Івана Франка
Факультет електроніки та комп'ютерних технологій
Кафедра системного проектування

Курсова робота
“Розробка інформаційного веб-сайту гуртожитку”

Виконала:
студентка групи ФeП-21
спеціальності 121 – Інженерія програмного забезпечення
_____ Ящук С. М.
Науковий керівник:
_____ Завідувач кафедри доц. Шувар Р. Я.
«____» _____ 2024 р.

Львів 2024

АНОТАЦІЯ

У цій курсовій роботі представлено розробку “Веб-сайту гуртожитку”. Основний акцент зроблено на проєктуванні бази даних, яка забезпечує збереження інформації про студентів, їхні дані проживання, а також обробку реєстраційних форм та запитань від мешканців. Робота виконана за допомогою мови програмування Python у середовищі PyCharm з використанням фреймворку Flask. Для збереження даних використовувала MySQL.

ABSTRACT

This coursework presents the development of a “Dorm Website”. The main emphasis is on designing a database that stores information about students, their residence data, as well as processing registration forms and questions from residents. The work was done using the Python programming language in the PyCharm environment using the Flask framework. MySQL was used to store data.

Зміст

АНОТАЦІЯ.....	2
ABSTRACT.....	2
ВСТУП.....	5
РОЗДІЛ 1. ТЕОРЕТИЧНІ ВІДОМОСТІ.....	6
1.1. Загальні відомості про бази даних	6
1.2. Інтеграція баз даних із веб-технологіями	9
1.3. Мова програмування Python	10
1.4. Flask: мікрофреймворк для веб-додатків	10
1.5. Розробка інтерфейсу: HTML, CSS та JavaScript.....	11
1.6. Використання Jinja2 для шаблонізації	12
РОЗДІЛ 2. АНАЛІТИЧНИЙ ОГЛЯД	14
2.1. Актуальність веб-сайту для студентів гуртожитку	14
2.2. Аналіз існуючих веб-сайтів гуртожитків	15
2.3. Вибір технологій для реалізації проекту	15
РОЗДІЛ 3. ПОСТАНОВКА ЗАВДАННЯ	18
3.1. Вхідні дані для виконання проекту	18
3.2. Архітектура рішення: серверна частина, база даних, інтерфейс	19
РОЗДІЛ 4. РЕАЛІЗАЦІЯ ПРОЄКТУ	21
4.1. Проектування та реалізація бази даних.....	21
4.2. Розробка серверної частини на Flask	24
4.3. Створення функціональних сторінок	27
4.4. Інтеграція фронтенду з бекендом	32
4.5. Приклад роботи додатку	35

ВИСНОВОК.....40

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....41

ВСТУП

У сучасних умовах, коли ефективне управління житлом та забезпечення комфортичних умов для студентів стає важливим аспектом, організація правильної роботи з даними про мешканців гуртожитку є необхідною для підтримки порядку та зручності. Метою моєї курсової роботи було створення веб-сайту для управління інформацією про студентів, що проживають у гуртожитку, з використанням сучасних інструментів веб-розробки та управління базами даних.

Додаток дозволяє користувачам реєструватися, заповнювати особисті дані, переглядати новини, знаходити потрібну інформацію про умови проживання, а також ставити запитання до адміністрації гуртожитку. Користувачі можуть переглядати важливі оголошення, надавати зворотний зв'язок і звертатися за допомогою через форму запитань.

Цей додаток стане корисним інструментом для студентів, які проживають у гуртожитку, а також для адміністраторів, яким він дозволяє ефективно організувати облік студентів та взаємодію з ними.

У курсовій роботі детально описано процес розробки цього додатку: від аналізу потреб користувачів до створення бази даних для зберігання інформації про студентів, новини та запитання. Особливу увагу приділено проєктуванню та реалізації функціоналу, що сприяє ефективному управлінню даними студентів і забезпеченням зручного інтерфейсу для всіх користувачів.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ВІДОМОСТІ

1.1. Загальні відомості про бази даних

Бази даних: визначення та основні поняття

Бази даних (БД) представляють собою впорядковані колекції інформації, які використовуються для зберігання, організації та зручного управління даними. Вони є ключовим елементом багатьох сучасних інформаційних систем, забезпечуючи доступ, обробку та оновлення великих обсягів даних.

У сфері інформаційних технологій бази даних слугують для зберігання й роботи з інформацією, яка може стосуватися різноманітних сфер діяльності: від персональних даних користувачів до інформації про фінансові транзакції.

Класифікація баз даних

Бази даних можуть бути класифіковані за їхньою структурою та способом організації інформації. Основними типами є:

1. Реляційні бази даних: найбільш поширений тип, де дані зберігаються у форматі таблиць. Ці системи дозволяють ефективно працювати зі зв'язками між даними. Приклади: MySQL, PostgreSQL, Oracle Database.
2. NoSQL бази даних: нереляційні системи, які не дотримуються традиційної моделі таблиць. Вони створені для роботи з великими обсягами даних та підтримують різні структури, такі як документи, графи або формат ключ-значення. Приклади: MongoDB, Redis.

Системи керування базами даних (СКБД)

СКБД - це програмні засоби, що дозволяють створювати, управляти та отримувати доступ до баз даних. Вони забезпечують інструменти для роботи з інформацією, підтримуючи її безпеку та цілісність.

Реляційна модель баз даних та основи роботи з MySQL

Реляційна модель баз даних — це логічна структура, що використовує табличний формат для представлення даних. У цій моделі спосіб подання інформації не залежить від її фізичного розташування. Основою реляційної моделі є математичне поняття відношення, яке відповідає структурі таблиці.

Зв'язки між таблицями

У реляційній моделі даних зв'язки між таблицями створюються через ключові поля. Головним механізмом для цього є співставлення значень полів у різних таблицях. Зазвичай зв'язок встановлюється між первинним ключем основної таблиці та зовнішнім ключем залежної таблиці.

Типи міжтаблиців зв'язків

1. **Один-до-одного (1:1)** — кожен запис у одній таблиці відповідає лише одному запису в іншій. Наприклад, таблиця з інформацією про користувачів може бути пов'язана з таблицею адрес, якщо кожен користувач має тільки одну адресу.
2. **Один-до-багатьох (1:N)** — одному запису в першій таблиці може відповідати кілька записів у другій. Наприклад, один проект може включати кількох учасників.
3. **Багато-до-багатьох (M:N)** — записи з першої таблиці можуть бути пов'язані з декількома записами другої, і навпаки. Для реалізації таких зв'язків зазвичай використовується проміжна таблиця. Наприклад, таблиці користувачів і проектів можуть бути з'єднані через таблицю, яка відображає участь у проектах.

Однією з найпоширеніших систем керування базами даних (СКБД) є **MySQL**.

MySQL — це реляційна система керування базами даних з відкритим кодом, яка забезпечує створення, оновлення, видалення і аналіз даних.

Основні компоненти:

1. **Таблиця** — головна структура для зберігання даних, організованих у вигляді рядків і стовпців.
2. **Рядок (запис)** — це окремий елемент даних, наприклад, інформація про одного студента чи проект.
3. **Стовпець (атрибут)** — це характеристика, яка визначає властивості даних, наприклад, ім'я студента, назва проекту або його категорія.
4. **Первинний ключ** — унікальний ідентифікатор для кожного запису в таблиці.

Для роботи з MySQL застосовується мова SQL (Structured Query Language), яка дозволяє ефективно маніпулювати даними, виконуючи запити на їх створення, зміну або аналіз.

Нормалізація даних

Нормалізація — це процес впорядкування даних у базі для запобігання їх дублюванню та забезпечення цілісності інформації. Цей підхід передбачає оптимальне структурування таблиць, щоб дані зберігалися найбільш ефективно. Існує декілька рівнів нормалізації:

1. **Перша нормальна форма (1NF):** кожен стовпець у таблиці містить тільки одне унікальне значення, а повторювані групи даних відсутні.
2. **Друга нормальна форма (2NF):** усі атрибути, які не є частиною первинного ключа, повинні повністю залежати від нього.
3. **Третя нормальна форма (3NF):** всі поля в таблиці мають бути незалежними один від одного, а залежності, що створюють транзитивність, мають бути усунені.

4. **Нормальна форма Бойса-Кодда (BCNF):** уdosконалена версія 3NF, де кожна детермінанта є кандидатом на ключ.

5. **Четверта нормальна форма (4NF):** усуває багатозначні залежності між атрибутами таблиці.

6. **П'ята нормальна форма (5NF):** розділяє таблиці на мінімальні, які можна з'єднати без втрати даних, забезпечуючи цілісність проекції-з'єднання (PJ/NF).

Ці рівні нормалізації допомагають оптимізувати структуру бази даних і зробити її більш ефективною для зберігання та управління інформацією.

1.2. Інтеграція баз даних із веб-технологіями

Інтеграція баз даних із веб-технологіями — це процес, який забезпечує взаємодію між веб-додатками та базами даних, дозволяючи ефективно зберігати, отримувати й обробляти інформацію. У сучасних веб-додатах бази даних відіграють ключову роль, зберігаючи важливі дані, такі як профілі користувачів, контент сайтів та інші ресурси. Завдяки цій інтеграції веб-додатки можуть обмінюватися інформацією з користувачами в режимі реального часу.

Користувач здійснює запит через веб-браузер, який передається серверу для обробки. Сервер формує SQL-запит і надсилає його до бази даних. Після виконання запиту база даних повертає результати серверу, який конвертує їх у зручний для користувача формат і відображає через інтерфейс веб-додатку.

Методи інтеграції залежать від архітектури конкретного проекту. У процесі важливу роль відіграють відповідні технології та інструменти. Наприклад, у проекті з базою даних гуртожитку використовується фреймворк Flask для обробки запитів, формування SQL-запитів до бази MySQL та динамічного створення веб-сторінок за допомогою шаблонів Jinja2.

Завдяки такій інтеграції веб-додатки можуть забезпечувати стабільну й безпечну роботу, обробляти запити користувачів, надавати доступ до актуальних даних та зберігати результати їхньої діяльності.

1.3. Мова програмування Python

Python є однією з найбільш популярних мов програмування у світі, яка знаходить застосування в багатьох галузях, таких як веб-розробка, аналіз даних, машинне навчання, автоматизація та багато інших.

Головні переваги Python:

1. Зручний синтаксис. Python розроблений для простоти та легкості у використанні, що робить його чудовим вибором для початківців.
2. Розширенна стандартна бібліотека. Мова включає великий набір вбудованих інструментів і модулів для вирішення різноманітних завдань: робота з файлами, мережами, базами даних тощо.
3. Різноманіття підходів до програмування. Python підтримує процедурний, об'єктно-орієнтований і функціональний стилі написання коду.
4. Кросплатформеність. Python-програми сумісні з багатьма операційними системами, зокрема Windows, macOS і Linux.

Python у веб-розробці:

Python надає багатий вибір інструментів і фреймворків для створення веб-додатків будь-якої складності. Найвідомішими з них є Django та Flask, які забезпечують ефективний процес розробки та широкий функціонал.

1.4. Flask: мікрофреймворк для веб-додатків

Flask — це популярний мікрофреймворк для Python, призначений для створення веб-додатків із максимальною свободою у виборі архітектури. Він

дозволяє розробникам розпочати проект із нуля, забезпечуючи повний контроль над його структурою.

Що пропонує Flask:

Flask спрощує створення серверної частини веб-додатків, приймаючи запити від браузера, обробляючи їх і повертаючи результати у вигляді HTML-сторінок або даних у форматі JSON.

Основні особливості Flask:

1. Легкість і мінімалізм: Flask не вимагає сувороого дотримання певної структури проекту, дозволяючи розробнику самостійно вибирати необхідні інструменти та підходи.
2. Гнучкість і інтеграція: Фреймворк легко працює з іншими бібліотеками, наприклад, SQLAlchemy для роботи з базами даних або Flask-WTF для обробки форм.
3. Вбудований шаблонізатор: Flask використовує Jinja2, що дозволяє динамічно створювати HTML-сторінки, підвищуючи зручність та ефективність роботи з інтерфейсом.

1.5. Розробка інтерфейсу: HTML, CSS та JavaScript

Інтерфейс веб-додатку розробляється за допомогою таких технологій:

1. **HTML** (HyperText Markup Language) — це основа для створення структури веб-сторінок. Завдяки HTML визначаються елементи сторінки, такі як блоки, форми, кнопки та інші компоненти.
2. **CSS** (Cascading Style Sheets) — використовується для оформлення веб-сторінок. За допомогою CSS можна змінювати кольори, шрифти, макет елементів та їх адаптивність до різних пристрій.

3. **JavaScript** — мова програмування, що дозволяє додавати інтерактивність на стороні клієнта. Наприклад, вона використовується для перевірки правильності введених даних у формах перед їх відправленням.

Набір цих технологій дає змогу створювати зручні та естетично привабливі інтерфейси, що значно покращує користувальський досвід.

1.6. Використання **Jinja2** для шаблонізації

Jinja2 — це шаблонізатор, який інтегрується з фреймворком Flask, полегшуючи створення динамічних веб-сторінок. Завдяки **Jinja2** розробники можуть вставляти серверні змінні та дані безпосередньо в HTML-код, що дозволяє зробити сторінки інтерактивними та зручними для користувачів.

Як працює **Jinja2**?

Коли користувач відправляє запит на веб-сторінку, сервер обробляє цей запит, отримує необхідні дані з бази та через **Jinja2** вбудовує їх в HTML-шаблон. У результаті виходить сторінка, яка виглядає як звичайний статичний сайт, але містить динамічно згенеровану інформацію.

Основні функції **Jinja2**:

- Виведення змінних: дозволяє вставляти змінні прямо в HTML-код, наприклад, для відображення імені користувача, який увійшов в акаунт.
- Цикли та умови: **Jinja2** дозволяє створювати динамічний HTML, наприклад, для відображення таблиць чи списків. Також можна використовувати умовні конструкції для зміни вигляду сторінки залежно від даних.
- Наслідування шаблонів: замість дублювання однакових елементів (наприклад, шапки сайту чи меню), можна створювати базовий шаблон і "успадковувати" його на інших сторінках. Це спрощує підтримку сайту та зміни в його структурі.

Чому це зручно?

Jinja2 дозволяє автоматизувати створення веб-сторінок та зробити код більш зрозумілим. Наприклад, замість того, щоб вручну повторювати один і той самий HTML для кожного проекту чи користувача, можна динамічно генерувати сторінки залежно від серверних даних.

РОЗДІЛ 2. АНАЛІТИЧНИЙ ОГЛЯД

2.1. Актуальність веб-сайту для студентів гуртожитку

Гуртожитки є важливими частинами студентського життя, надаючи умови для комфорtnого проживання та навчання. Однак комунікація між студентами та адміністрацією часто є складною та трудомісткою. Веб-сайт, який забезпечує швидкий доступ до інформації, зручний спосіб реєстрації на поселення, а також можливість звернення до адміністрації з питаннями чи пропозиціями, стає важливим інструментом для оптимізації процесів у гуртожитку.

Часто студенти мають питання або потреби, які необхідно терміново вирішити, але не завжди є можливість звернутися до адміністрації особисто. В такому випадку веб-сайт стає незамінним, оскільки дозволяє легко отримати відповідь на запитання, ознайомитися з новинами чи зареєструватися на поселення без необхідності відвідувати адміністрацію.

Значення веб-сайту: Веб-сайт Гуртожитку дозволяє студентам здійснювати реєстрацію на поселення, переглядати новини, отримувати актуальну інформацію, а також звертатися до адміністрації з питаннями чи пропозиціями в зручний час. Це зменшує навантаження на адміністрацію та підвищує ефективність взаємодії між студентами та керівництвом.

Сучасні тенденції: З кожним роком зростає попит на онлайн-інструменти, які спрощують комунікацію та організацію різних процесів. Веб-сайти для студентів гуртожитків стають все більш актуальними, оскільки вони забезпечують швидкий доступ до важливої інформації, зменшують адміністративне навантаження та сприяють покращенню умов для навчання та проживання. Очікується, що в майбутньому подібні веб-сайти будуть ставати невід'ємною частиною інфраструктури навчальних закладів.

2.2. Аналіз існуючих веб-сайтів гуртожитків

Переглянувши ряд веб-сайтів гуртожитків, я помітила, що багато з них не мають деяких важливих компонентів, які могли б значно полегшити процес комунікації між студентами та адміністрацією. Зокрема, відсутність реєстраційної форми на поселення є однією з основних проблем, оскільки студенти змушені звертатися до адміністрації особисто або через інші канали для оформлення поселення. Це створює зайві труднощі і потребує додаткового часу, як для студентів, так і для адміністраторів.

Ще одним недоліком є відсутність функції для надсилання запитань адміністрації, що є важливим елементом для зручної комунікації. Без такої можливості студентам доводиться шукати інші способи зв'язку, що може бути не так ефективно, як через спеціальну форму на сайті.

Враховуючи ці проблеми, я вирішила створити сайт для гуртожитку, який об'єднає всі необхідні функції для зручного користування. Мій веб-сайт включатиме реєстраційну форму на поселення онлайн, форму для надсилання запитань адміністрації, а також інші інструменти для швидкого отримання новин та комунікації. Це дозволить студентам легко реєструватися на поселення, звертатися з питаннями до адміністрації та бути в курсі важливої інформації, що значно покращить їхнє перебування в гуртожитку.

2.3. Вибір технологій для реалізації проєкту

Для розробки веб-сайту гуртожитку було обрано низку технологій, які дозволяють ефективно реалізувати необхідний функціонал, забезпечуючи зручність для користувачів і ефективну роботу з базою даних. Ось основні з них:

Flask (Python)

Причини вибору: Flask є мінімалістичним і зручним у використанні фреймворком, що дозволяє швидко створювати веб-додатки. Він легкий у вивченні та має високу гнучкість. Крім того, Python є популярною мовою програмування, що надає безліч можливостей для подального розвитку проєкту.

- Роль у проєкті: Flask обраний для створення серверної частини сайту, забезпечуючи налаштування веб-сервера та обробку запитів користувачів.
- Застосування: Flask використовується для обробки маршрутів і взаємодії з базою даних для таких сторінок, як реєстрація на поселення, запитання до адміністрації, перегляд новин тощо.
- Приклад: Коли студент реєструється на поселення, дані з реєстраційної форми передаються на сервер через Flask і зберігаються в базі даних MySQL.

MySQL

Причини вибору: MySQL є простою в установці та використанні реляційною базою даних з високою швидкістю обробки запитів і здатністю працювати з великими обсягами інформації.

- Роль у проєкті: MySQL використовується для зберігання даних про студентів, реєстрації на поселення, запитів до адміністрації та новин. Реляційна структура дозволяє ефективно управлюти взаємозв'язками між таблицями і зберігати дані в організованому вигляді.
- Застосування: Всі дані про студентів, поселення, запитання і новини зберігаються в MySQL. Наприклад, при реєстрації на поселення, вся інформація з форми зберігається в базі даних.
- Приклад: Коли студент подає заявку на поселення, її дані (ім'я, кімната, термін перебування) зберігаються в відповідних таблицях MySQL.

JavaScript

JavaScript використовується для створення динамічного інтерфейсу на стороні клієнта, забезпечуючи інтерактивність веб-сайту.

Jinja2

Причини вибору: Jinja2 інтегрується з Flask для динамічного створення сторінок і дозволяє зручно передавати змінні у HTML-шаблони.

- Роль у проєкті: Jinja2 генерує динамічний HTML-код для створення веб-сторінок на сервері. Застосування: Наприклад, коли студент реєструється для поселення або надсилає запит до адміністрації, Jinja2 генерує необхідну сторінку для відображення результатів на основі отриманих даних.

HTML та CSS

- Роль у проєкті: HTML і CSS використовуються для створення структури та дизайну веб-сторінок.
- Застосування: HTML визначає структуру сторінок, включаючи форми для реєстрації, запитів та новин, а CSS забезпечує їх стильове оформлення, покращуючи користувацький досвід.

Для управління базою даних я використовую MySQL Workbench, що забезпечує зручність у налаштуванні та адмініструванні бази даних.

РОЗДІЛ 3. ПОСТАНОВКА ЗАВДАННЯ

3.1. Вхідні дані для виконання проєкту

Для успішної реалізації веб-сайту гуртожитку основними вхідними даними є:

Форма для реєстрації на поселення:

- Параметри для реєстрації: ПІБ, група, факультет, курс, номер телефону, електронна пошта.
- Контактні дані для зв'язку з адміністрацією, підтвердження реєстрації.

Форма для питань:

- Можливість надсилати питання адміністрації гуртожитку.
- Поле для детального опису питання чи коментаря.

Пошук студентів за ПІБ:

- Функція для пошуку студентів за їхніми прізвищем та іменем для зручності контактів між ними.

Сторінка новин:

- Розділ для публікації новин та актуальної інформації щодо життя в гуртожитку, оголошень та змін у правилах.

Посилання на чат Telegram:

- Вбудоване посилання на чат Telegram для швидкої комунікації з адміністрацією або іншими студентами гуртожитку.

Інформаційна сторінка про дирекцію:

- Сторінка з контактною інформацією та даними про дирекцію гуртожитку.

Ці дані допомагають налаштувати веб-сайт для зручного використання студентами та адміністрацією гуртожитку, забезпечуючи ефективну комунікацію, реєстрацію на поселення та доступ до актуальної інформації.

3.2. Архітектура рішення: серверна частина, база даних, інтерфейс

Серверна частина

Серверна частина реалізована за допомогою Flask, який виконує роль основного веб-сервера, обробляючи HTTP-запити та керуючи маршрутизацією між клієнтом і базою даних. Всі операції, такі як реєстрація на поселення, надсилання запитань адміністрації, перегляд новин, обробка автентифікації та авторизації користувачів, виконуються на сервері.

Основні елементи серверної частини:

Flask обробляє маршрути (routes), які відповідають за:

- Реєстрацію студентів на поселення.
- Надсилання запитань адміністрації.
- Перегляд новин.
- Відображення інформаційної сторінки дирекції.
- Генерацію посилання на Telegram-чат.

Сервер взаємодіє з клієнтською частиною, передаючи дані у форматі HTML (за допомогою шаблонів Jinja2).

База даних

Всі дані користувачів, запитань, новин та іншої інформації зберігаються у реляційній базі даних MySQL (з використанням MySQL Workbench). Інформація організована в таблиці, такі як:

- **Користувачі:** дані про студентів і адміністрацію.

- **Реєстрація на поселення:** дані про заявки студентів на поселення.
- **Запитання:** інформація про повідомлення, надіслані адміністрації.
Інтерфейс користувача (UI)

Інтерфейс користувача створений за допомогою HTML, CSS та JavaScript. Для організації динамічного контенту на сторінках використовується Jinja2 — шаблонізатор для Python, що дозволяє впроваджувати змінні та логіку безпосередньо в HTML-розділку.

Основні елементи інтерфейсу:

- **Форма реєстрації на поселення:** дозволяє студентам надсилюти заявки онлайн.
- **Форма для запитань:** дає змогу студентам комунікувати з адміністрацією безпосередньо через сайт.
- **Сторінка новин:** відображає актуальні оголошення та новини.
- **Інформаційна сторінка дирекції:** містить контакти та інформацію дирекції.
- **Посилання на Telegram-чат:** вбудована кнопка для швидкого переходу до чату.

Ця архітектура забезпечує ефективну роботу веб-сайту гуртожитку, надаючи користувачам зручний доступ до важливих функцій та інформації.

РОЗДІЛ 4. РЕАЛІЗАЦІЯ ПРОЄКТУ

4.1. Проектування та реалізація бази даних

Для розробки бази даних використовувалася система керування базами даних MySQL із середовищем розробки MySQL Workbench.

Таблиці

У структурі бази даних реалізовано чотири основних таблиць: student, question, requests, news. (Рис. 4.1.1)

Tables_in_гуртожиток	
▶	news
	question
	requests
	student

Рис.4.1.1. Структура бази даних гуртожитка

Структура таблиць

1. **Таблиця student:** містить інформацію про студентів, що проживають в гуртожитку.

```
class Student(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    full_name = db.Column(db.String(100), nullable=False)
    course = db.Column(db.Integer, nullable=False)
    faculty = db.Column(db.String(50), nullable=False)
    floor = db.Column(db.Integer, nullable=False)
    room_number = db.Column(db.Integer, nullable=False)
    phone_number = db.Column(db.String(15), nullable=False)
```

2. Таблиця question: зберігає надісланні запитання користувачів.

```
class Question(db.Model):  
    id = db.Column(db.Integer, primary_key=True)  
    name = db.Column(db.String(100), nullable=False)  
    email = db.Column(db.String(100), nullable=False)  
    message = db.Column(db.Text, nullable=False)
```

3. Таблиця requests: зберігає данні користувачів, які подають заяву на гуртожиток.

```
class Requests(db.Model):  
    id = db.Column(db.Integer, primary_key=True)  
    name = db.Column(db.String(100), nullable=False)  
    group = db.Column(db.String(100), nullable=False)  
    faculty = db.Column(db.String(100), nullable=False)  
    course = db.Column(db.Integer, nullable=False)  
    phone = db.Column(db.String(20), nullable=False)  
    email = db.Column(db.String(100), nullable=False)
```

4. Таблиця news: містить актуальні новини.

```
class News(db.Model):  
    id = db.Column(db.Integer, primary_key=True)  
    title = db.Column(db.String(200), nullable=False)  
    content = db.Column(db.Text, nullable=False)
```

Реалізація баз даних:

- Всі таблиці було створено за допомогою моделей та ORM(SQLAlchemy).
- Дані таблиць були заповнені вручну через SQL-запити для демонстрації можливостей системи.

Висновки

База даних спроєктована таким чином, щоб забезпечити:

- Масштабованість: можливість додавати нові таблиці чи поля.
- Продуктивність: оптимізація структури та використання відповідних типів даних.

Опис таблиць:

1. Student

Таблиця **Student** зберігає основну інформацію про студентів, які проживають у гуртожитку. Вона містить такі поля, як повне ім'я, курс навчання, факультет, поверх та номер кімнати. Окрім того, додається контактний номер телефону студента для оперативного зв'язку. Поле *id* є унікальним ідентифікатором кожного запису, що дозволяє швидко знаходити інформацію. Ця таблиця використовується для пошуку та перегляду даних студентів адміністрацією або іншими користувачами.

2. Question

Таблиця **Question** зберігає запити або запитання, які студенти надсилають через форму на сайті. Вона включає поля для імені відправника, його електронної пошти та тексту повідомлення. Поле *id* ідентифікує кожен окремий запит у базі даних. Використовується для відстеження звернень студентів і їх подальшої обробки адміністрацією.

Таблиця допомагає налагодити ефективний зворотний зв'язок між мешканцями гуртожитку та адміністрацією.

3. News

Таблиця **News** зберігає актуальні новини, пов'язані з життям гуртожитку. Поле **title** використовується для зберігання заголовка новини, що коротко описує її суть. Поле **content** містить детальний текст новини, який пояснює подію чи оновлення. Кожна новина ідентифікується унікальним **id**, що дозволяє легко отримувати або змінювати записи. Ця таблиця використовується для відображення інформації на головній сторінці сайту.

4. Requests

Таблиця **Requests** зберігає дані заявок студентів на поселення в гуртожиток. Вона містить поля для імені заявника, його групи, факультету, курсу, контактного телефону та email. Поле **id** унікально ідентифікує кожну заявку, що дозволяє ефективно керувати даними. Дані з таблиці використовуються адміністрацією для розгляду та затвердження заявок. Ця таблиця забезпечує зручну систему обробки реєстрації студентів онлайн.

4.2. Розробка серверної частини на Flask

Розробка серверної частини веб-сайту гуртожитку базується на мікрофреймворку **Flask**, який забезпечує гнучкий та простий спосіб створення. Основними завданнями серверної частини є обробка HTTP-запитів, взаємодія з базою даних MySQL та генерація динамічного HTML-контенту.

Для початку роботи з серверною частиною в додатку PyCharm створюю новий файл app.py в загальній папці проекту (рис. 4.2.1).

Цей файл створює веб-сайт на Flask, який використовує SQLAlchemy для роботи з базою даних та Flask-WTF для форм. Веб-сайт дозволяє користувачам шукати студентів за ПІБ, переглядати новини, надсилювати запитання дирекції гуртожитку, а також реєструватися для поселення. Крім того, він підтримує прості функції, такі як відправка даних через форми, перевірка заповнених полів і збереження інформації в базу даних.

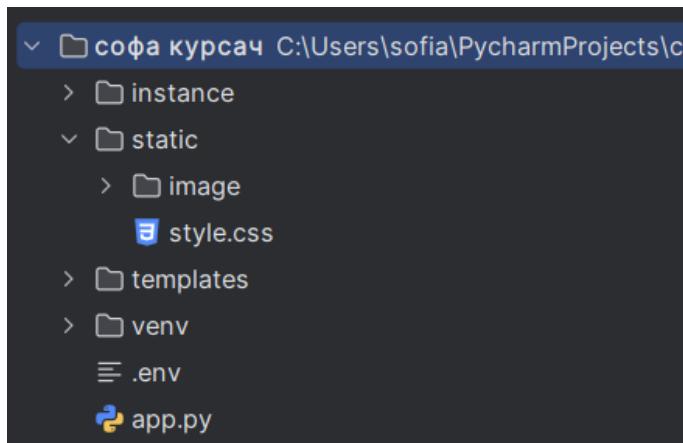


Рис. 4.2.1. Структура папки з фалом app.py

Основні компоненти серверної частини Конфігурація додатку:

- Flask додаток налаштовано для роботи з базою даних за допомогою бібліотеки Flask_SQLAlchemy.
- Параметри підключення до бази даних (хост, користувач, пароль, назва бази) зберігаються у файлі .env для безпечноного доступу.

```
1 from flask import Flask, render_template, r
2 from flask_sqlalchemy import SQLAlchemy
3 from flask_wtf import FlaskForm
```

Рис. 4.2.2. Підключення Flask-SQLAlchemy

Маршрутизація

Серверна частина веб-додатку реалізує маршрутизацію, що дозволяє обробляти запити користувачів та відповідати на них динамічним контентом. Вона забезпечує доступ до різних сторінок, форм і функціональностей сайту. Нижче наведено опис основних маршрутів:

- Головна сторінка (/)**

Відображає основну сторінку сайту гуртожитку, на якій представлені новини. Цей маршрут використовує шаблон index.html для динамічного відображення вмісту, отриманого з таблиці новин у базі даних.

- Сторінка студентів (/students)**

Забезпечує функціонал пошуку студентів за повним ім'ям. Використовує форму для введення запиту, обробляє пошукові запити та відображає результати з таблиці студентів у базі даних.

- Деталі новин (/news/<int:news_id>)**

Цей маршрут відображає детальний вміст конкретної новини, выбраної користувачем. Дані отримуються з таблиці новин на основі id, а шаблон news_detail.html використовується для представлення інформації.

- Запитання (/questions)**

Дозволяє користувачам надсилати запитання через спеціальну форму. Дані запитань зберігаються в базі даних, а після успішного надсилання користувач отримує повідомлення про підтвердження.

- **Реєстрація (/registration)**

Забезпечує можливість онлайн-реєстрації студентів для поселення. Форма приймає інформацію про студента (ім'я, групу, факультет, курс тощо) і зберігає її в таблиці заявок (Requests). У разі успіху користувач отримує відповідь про успішну реєстрацію.

- **Сторінка управління (/management)**

Представляє інтерфейс для адміністративного управління функціями сайту. Цей маршрут призначений для використання адміністрацією гуртожитку.

- **Чат (/chat)**

Перенаправляє користувача на зовнішнє посилання чату гуртожитку, наприклад, у Telegram. Це дозволяє мешканцям швидко взаємодіяти один з одним.

Маршрути дозволяють реалізувати основні функції веб-сайту гуртожитку, забезпечуючи зручний інтерфейс і динамічний обмін даними між користувачами та сервером.

4.3. Створення функціональних сторінок

В головній папці проекту, на рівні з app.py, створюю папку Templates, в якій створюватиму всі html-файли (рис.4.3.1)

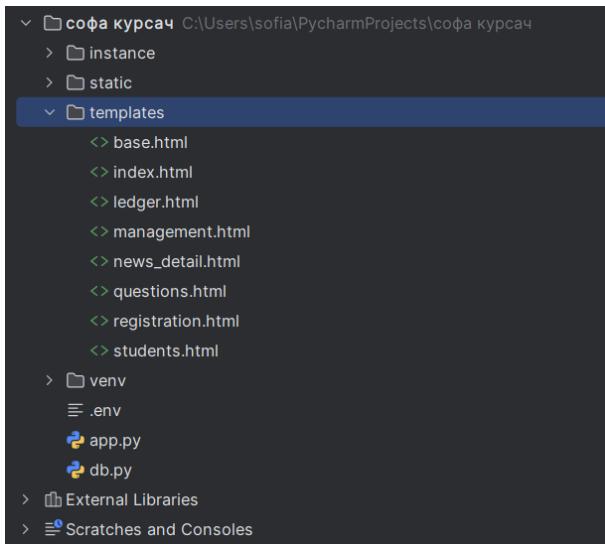


Рис. 4.3.1. Структура папки Templates.

Всі сторінки створені за допомогою HTML, CSS та JS (для динамічних елементів).

Короткий опис файлів:

- Base.html:

Задає базову структуру веб-сторінки з заголовком, меню навігації, місцем для відображення повідомлень та блоком для динамічного вмісту, що буде визначений у дочірніх шаблонах. Файл використовується як базовий шаблон.

- Index.html:

Шаблон головної сторінки сайту гуртожитку, де відображаються останні новини. У коді реалізовані стилізація новинних блоків, можливість відображення деталей новини за допомогою JavaScript, а також додано зображення гуртожитку у бічній панелі.

- Management.html:

Шаблон сторінки, яка відображає інформацію про дирекцію гуртожитку. Вміст сторінки включає заголовок, контактні дані директора, телефон та email, розташовані у вигляді списку.

- News_detail.html:

Цей шаблон створює сторінку для відображення окремої новини з її заголовком (news_item.title) та текстом (news_item.content). Також додається посилання для повернення на головну сторінку з усіма новинами.

- Question.html:

Цей шаблон створює сторінку для надсилання запитань користувачами через форму. У формі є поля для введення імені, електронної пошти та самого запитання, які обробляються через об'єкт question_form. Після відправлення користувачеві відображаються повідомлення про успішність або помилки за допомогою Flash-повідомень.

- Registration.html:

Цей код створює сторінку реєстрації для поселення в гуртожиток з формою, що включає поля для введення ПІБ, група, факультет, курс, телефон та email. Після заповнення форми дані відправляються методом POST, а JavaScript обробляє дію, показуючи повідомлення про успішну реєстрацію.

- Students.html:

Цей код створює форму для пошуку студентів за ПІБ, де користувач може ввести запит і отримати список результатів, якщо знайдено відповідних студентів. Якщо знайдені студенти, відображаються їхні дані, включаючи ПІБ, факультет, курс, кімнату, поверх і телефон.

Реалізовані сторінки:

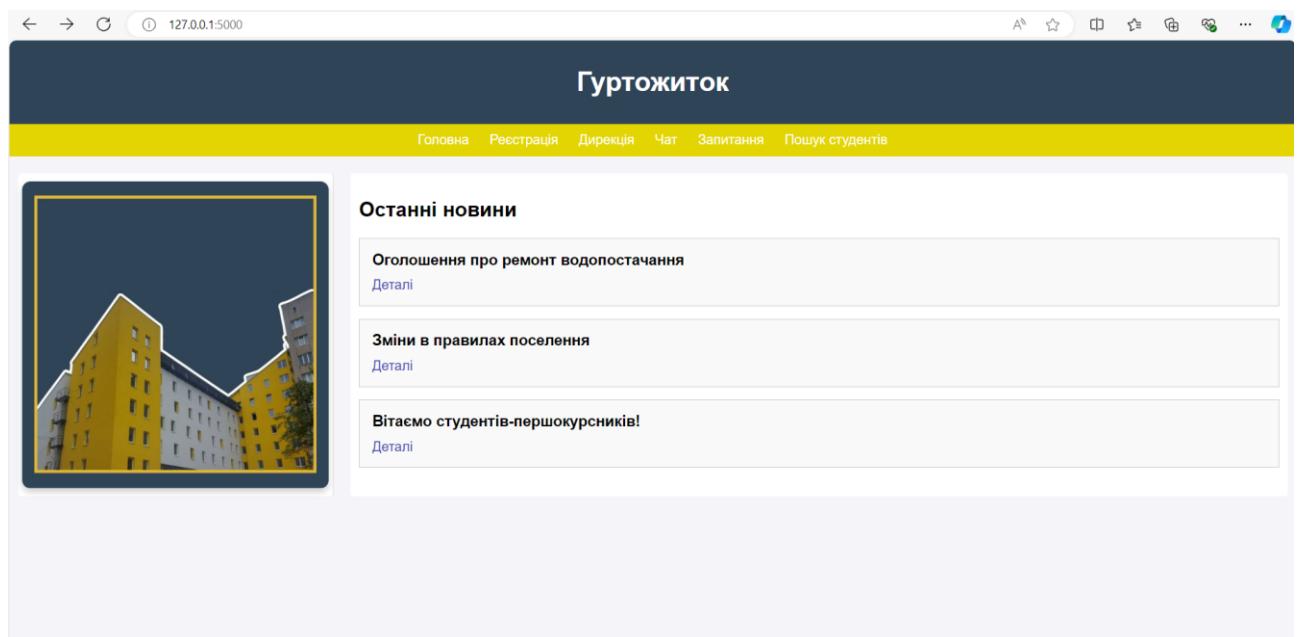


Рис. 4.3.2. Головна сторінка

На головній сторінці доступні кнопки:

- Реєстрація – перенаправляє нас на реєстраційну форму для поселення.
- Дирекція – сторінка з контактною інформацією про дирекцію гуртожитку
- Чат – телеграм-чат гуртожитку, де можна зв'язатись з дирекцією та іншими студентами, що проживають в гуртожитку.
- Запитання – форма завдяки якій можна написати пропозицію чи запитання дирекції гуртожитку.
- Пошук студентів – де при введенні ПІБ можна дізнатись контактну інформацію про студента, що проживає в гуртожитку.

The screenshot shows a registration form titled "Реєстрація для поселення". The form is intended for students to fill out their personal information. It includes fields for PIB (ПІБ), group (Група), faculty (Факультет), course (Курс), phone number (Номер телефону), and email (Електронна адреса). A green button at the bottom right is labeled "Зареєструватися" (Register).

Рис. 4.3.3. Сторінка реєстрації для поселення

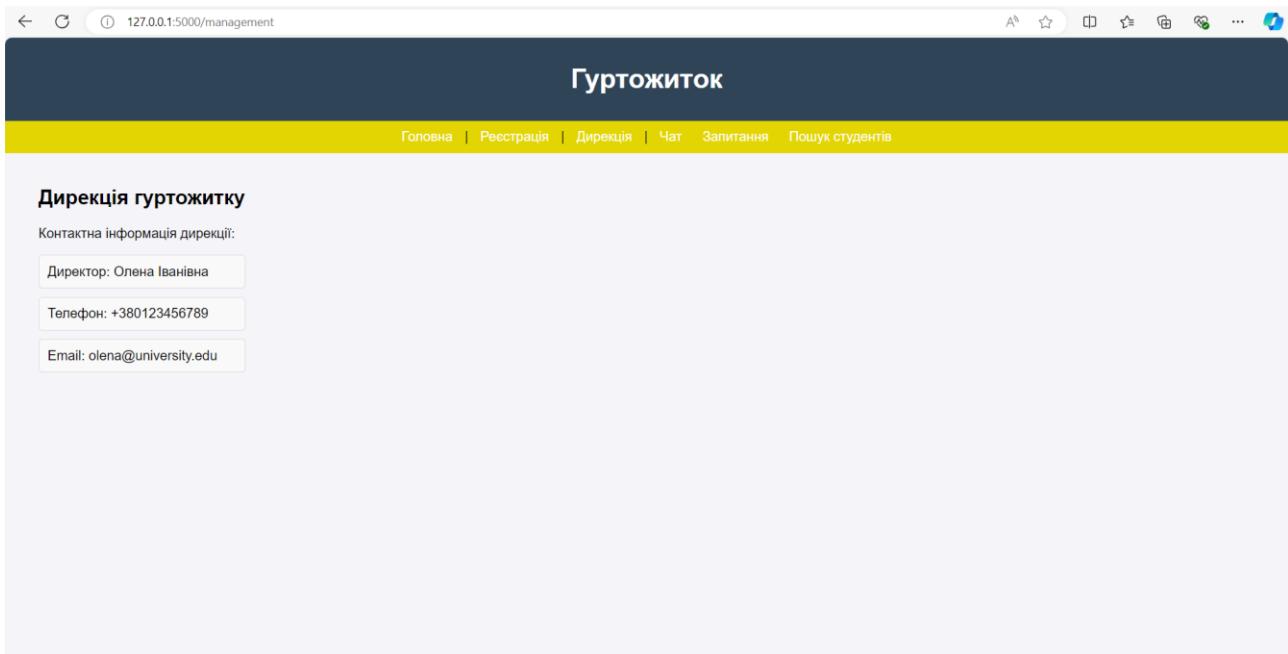


Рис. 4.3.4. Сторінка контактної інформації дирекції

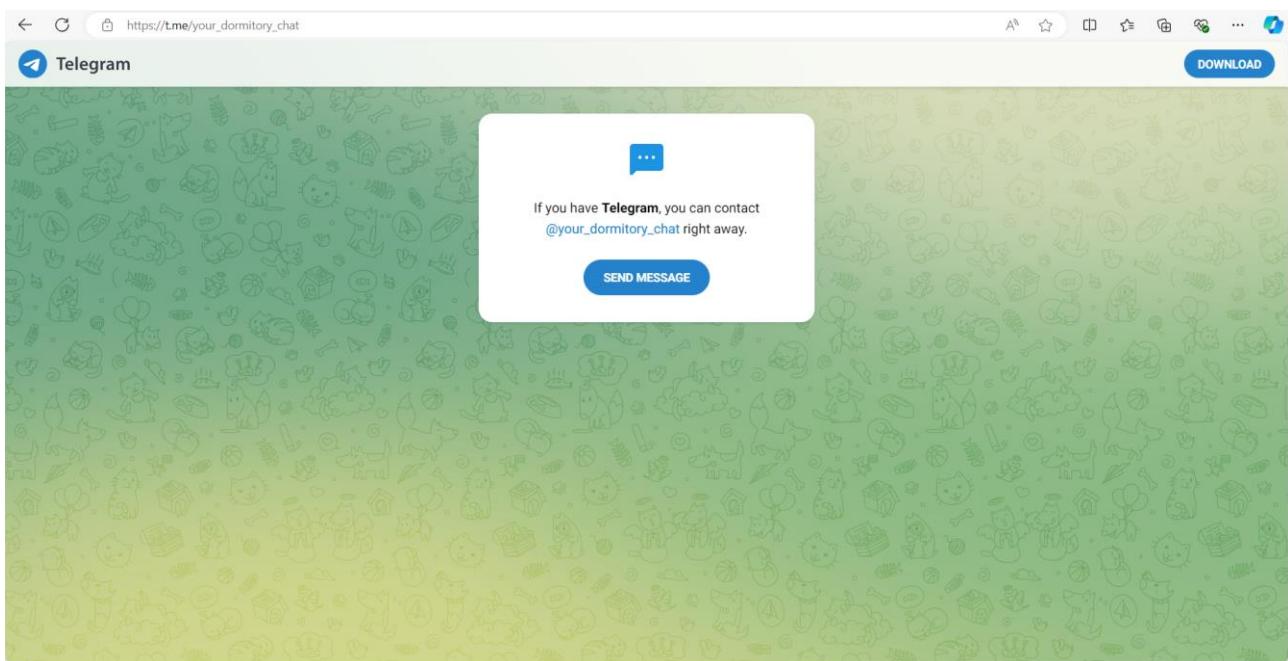


Рис. 4.3.5. Сторінка, що перенаправляє нас на телеграм-чат

127.0.0.1:5000/questions

Надіслати запитання

Головна Реєстрація Дирекція Чат Запитання Пошук студентів

Запитайте у нас!

Ваше ім'я:

Ваш Email:

Ваше запитання:

Рис. 4.2.6. Сторінка для надсилання запитань та пропозицій

127.0.0.1:5000/students

Гуртожиток

Головна | Реєстрація | Дирекція | Чат Запитання Пошук студентів

Пошук студентів

Введіть ПІБ:

Рис. 4.3.7. Сторінка пошуку студентів за ПІБ

4.4. Інтеграція фронтенду з бекендом

Інтеграція фронтенду з бекендом є важливим етапом створення веб-сайту, оскільки саме вона забезпечує взаємодію між користувачем і серверною частиною. У проекті веб-сайт гуртожитку цей процес здійснюється за допомогою фреймворку Flask, який забезпечує динамічну взаємодію між користувачським інтерфейсом та сервером.

Передача даних від фронтенду до бекенду

На сторінках, що передбачають введення даних, таких як реєстрація, пошук студентів або надсилання запитань, використовуються HTML-форми з методом POST. Flask обробляє ці дані за допомогою об'єкта request, отримуючи введену користувачем інформацію. Далі ці дані перевіряються, обробляються та, за необхідності, зберігаються у базі даних MySQL через ORM SQLAlchemy, що інтегрована у проект.

```
71 @app.route('/students', methods=['GET', 'POST'])
72 def students():
73     search_form = SearchForm()
74     students = []
75     if search_form.validate_on_submit() and search_form.query.data:
76         query = search_form.query.data.strip()
77         print(query)
78         students = Student.query.filter(Student.full_name.like(f"%{query}%")).all()
79         print(students)
80     return render_template('template_name_or_list: 'students.html', search_form=search_form, students=students)
```

Рис.4.4.1. Введення даних для пошуку студентів

```
88 @app.route('/questions', methods=['GET', 'POST'])
89 def questions():
90     question_form = QuestionForm()
91
92     if question_form.validate_on_submit():
93         new_question = Question(
94             name=question_form.name.data,
95             email=question_form.email.data,
96             message=question_form.message.data
97         )
98         db.session.add(new_question)
99         db.session.commit()
100        flash(message: 'Ваше запитання успішно надіслано!', category: 'success')
101        return redirect(url_for('questions'))
102
103    return render_template('template_name_or_list: 'questions.html', question_form=question_form)
```

Рис.4.4.2. Форма для надсилання запитань та пропозицій

```

105 @app.route('/registration', methods=['GET', 'POST'])
106 def registration():
107     if request.method == 'POST':
108         # Отримуємо дані з форми
109         name = request.form['name']
110         group = request.form['group']
111         faculty = request.form['faculty']
112         course = request.form['course']
113         phone = request.form['phone']
114         email = request.form['email']
115         print(f"name = {name}, group = {group}, faculty = {faculty}, course = {course}, phone = {phone}, email = {email}")
116
117         # Перевірка на наявність всіх полів
118         if not all([name, group, faculty, course, phone, email]):
119             return jsonify({"error": "Всі поля повинні бути заповнені!"}), 400
120
121         # Створення нового запису в базі даних
122         new_request = Requests(name=name, group=group, faculty=faculty,
123                               course=course, phone=phone, email=email)

```

Рис.4.4.3. Форма реєстрації для поселення

Використання шаблонів Jinja2

Flask забезпечує інтеграцію фронтенду через шаблони Jinja2, які надають можливість динамічно відображати дані на HTML-сторінках.

Наприклад, для відображення даних у списку проектів використовується такий підхід:

```

21      </nav>
22      <div class="container">
23          <h2>Запитайте у нас!</h2>
24          <form method="POST" action="{{ url_for('questions') }}">
25              {{ question_form.hidden_tag() }}
26              <div>
27                  {{ question_form.name.label }}:
28                  {{ question_form.name(size=32) }}
29              </div>
30              <div>

```

Рис.4.4.4. Приклад коду

Підсумок

Інтеграція фронтенду з бекеном забезпечила повну функціональність веб-додатку, дозволяючи обробляти користувачькі дані, зберігати їх у базі даних та динамічно відображати на веб-сторінках.

4.5. Приклад роботи додатку

Запускаю проект командою python app.py (Рис. 4.4.5)

```
PS C:\Users\sofia\PycharmProjects\софа курсач> python app.py
 * Serving Flask app 'app'
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 959-620-588
```

Рис.4.4.5. Приклад запуску проєкту

- 1) На головній сторінці можемо переглянути актуальні новини, натиснувши кнопку «Деталі».

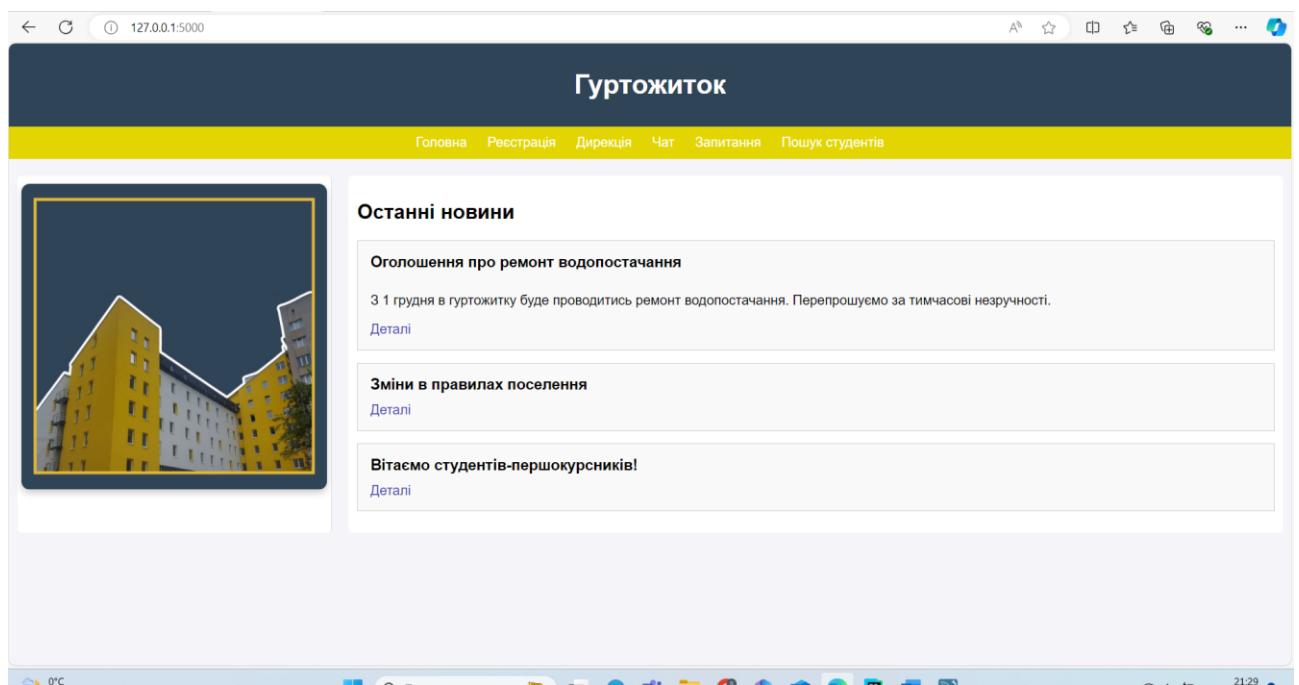


Рис. 4.5.1. Перегляд новин

2) Переходимо на сторінку реєстрації Рис. 4.5.2 та реєструємось Рис. 4.5.3.

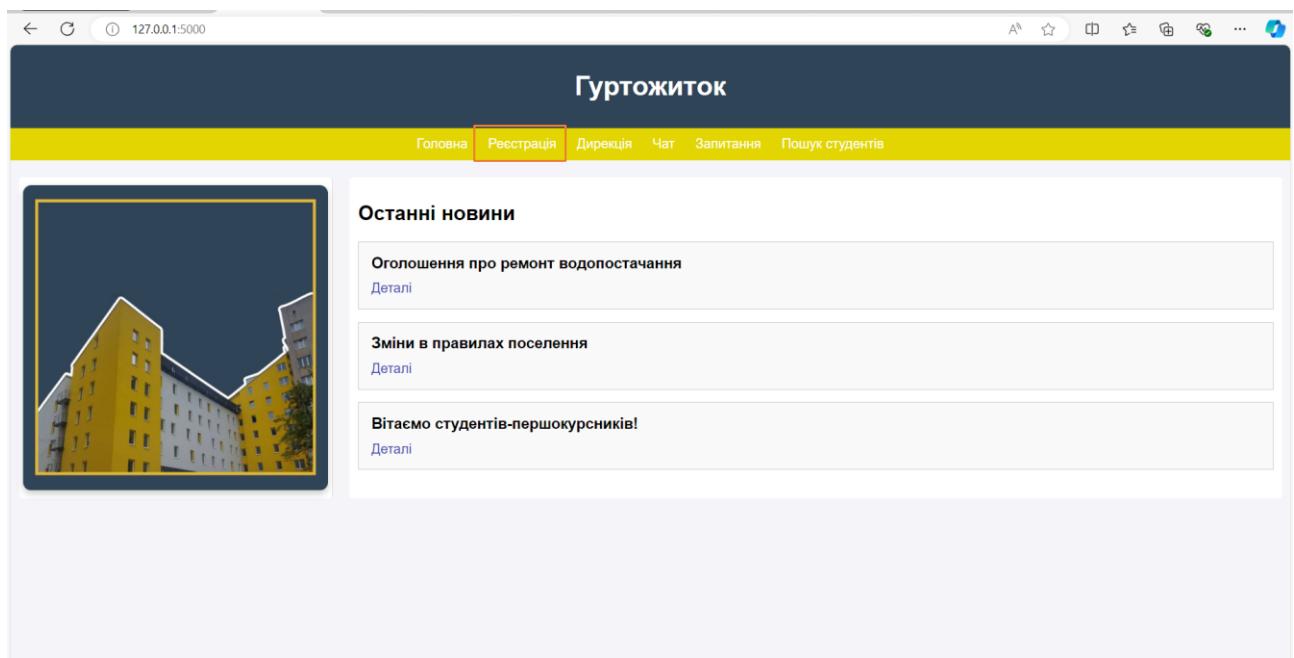


Рис. 4.5.2.

A screenshot of a web browser showing a registration form titled 'Реєстрація для поселення'. The form asks for personal information: 'ПІБ:' (Name) with the value 'Ящук Софія Миколаївна', 'Група (наприклад, ФЕП-21):' (Group) with the value 'ФЕП-21', 'Факультет:' (Faculty) with the value 'Електроніки', 'Курс:' (Course) with the value '2', 'Номер телефону:' (Phone number) with the value '1234567890', 'Електронна адреса:' (Email address) with the value 'sofa@gmail.com', and a green 'Зареєструватися' (Register) button at the bottom.

Рис. 4.5.3. Вносимо дані в реєстраційну форму

3) Перевіряємо наявність нового користувача в таблиці requests Рис. 4.5.4.

Result Grid							
	id	name	group	faculty	course	phone	email
▶*	1	Ящук Софія Миколаївна	ФЕП-21	Електроніки	2	1234567890	sofa@gmail.com
	NUL	NUL	NUL	NUL	NUL	NUL	NUL

Рис. 4.5.4. Вивід таблиці requests

- 4) Переходимо на сторінку «Запитання» Рис. 4.5.5. та надсилаємо запитання чи пропозицію Рис. 4.5.6.

The screenshot shows a web interface for a student dormitory management system. The main title is 'Гуртожиток'. The navigation bar includes links for 'Головна', 'Реєстрація', 'Дирекція', 'Чат', 'Запитання' (which is highlighted in red), and 'Пошук студентів'. On the left, there's a sidebar with the heading 'Останні новини' (Recent news) and three news items:

- Оголошення про ремонт водопостачання (Details)
- Зміни в правилах поселення (Details)
- Вітаємо студентів-першокурсників! (Details)

Рис. 4.5.5.

Рис. 4.5.6. Надсилаємо запитання або пропозицію

Рис. 4.5.7. Отримуємо повідомлення про успішне надсилання

	id	name	email	message
	1	Софія	sofa3@gmail.com	Vitao! Pidkajitez koliv bude zimova khvila poselenja? Dzakuo.
	NULL	NULL	NULL	NULL

Рис. 4.5.8. Вивід таблиці question

- 5) Переходимо на сторінку «Пошук студента» Рис. 4.5.9. та вводимо дані для пошуку Рис. 4.5.10.

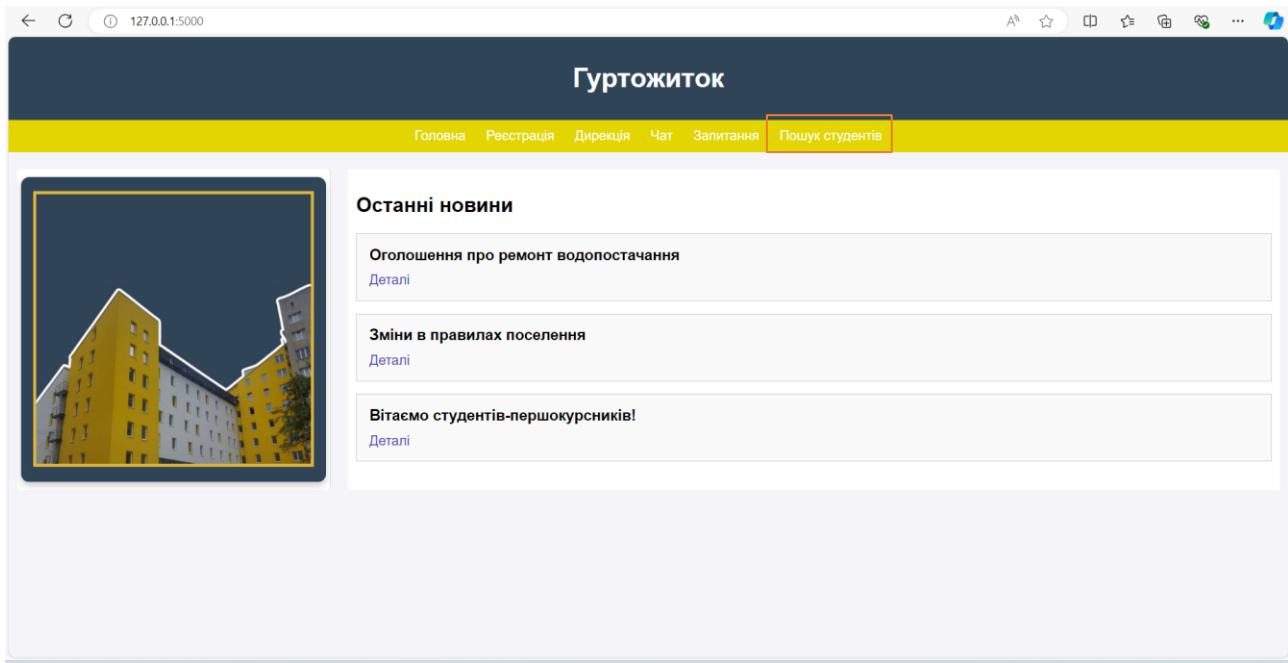


Рис. 4.5.9.

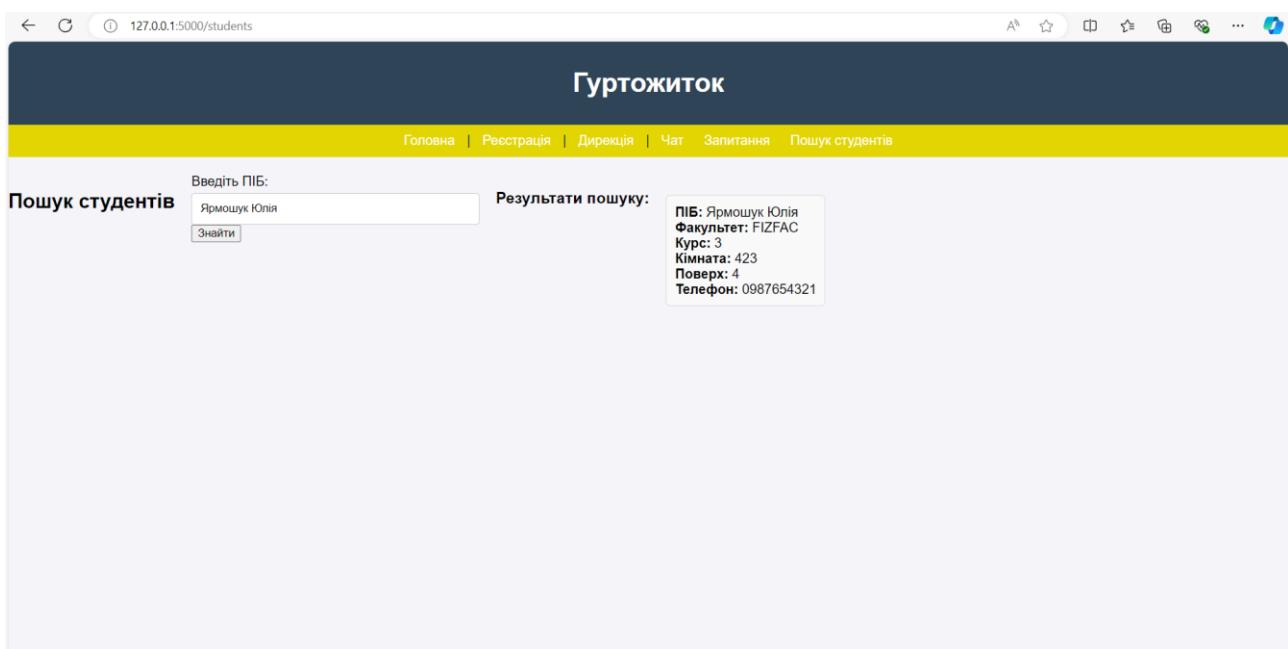


Рис. 4.5.10. Знаходимо студента за ПІБ

ВИСНОВОК

У процесі розробки цього проекту було створено основу для веб-сайту гуртожитку, який має полегшити взаємодію студентів з адміністрацією та надавати необхідну інформацію про проживання. Сайт дозволяє студентам реєструватися для поселення, ставити запитання адміністрації та отримувати актуальні новини.

Було розроблено структуру бази даних, що включає таблиці для зберігання даних про студентів, запити на поселення, запитання до адміністрації та новини. Архітектура враховує інтеграцію з фронтендом для динамічного відображення інформації.

У майбутньому планується додати наступні можливості:

- Особисті акаунти студентів із вкладками, як-от «Мої запити» або «Мої кімнати».
- Розширений пошук студентів із фільтрами за курсом, факультетом тощо.
- Система багатомовності для зручності іноземних студентів.
- Можливість збереження важливих новин, контактів або інших корисних матеріалів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

Documentation MySQL: <https://dev.mysql.com/doc/>

Documentation Flask: <https://flask.palletsprojects.com/>

Documentation SQLAlchemy: [SQLAlchemy Documentation — SQLAlchemy 2.0 Documentation](https://www.sqlalchemy.org/)

Documentation JavaScript: <https://developer.mozilla.org/en-US/docs/Web/JavaScript>

Documentation CSS: <https://developer.mozilla.org/en-US/docs/Web/CSS>

Documentation Jinja2: <https://jinja.palletsprojects.com/en/stable/>

Documentation Git: <https://git-scm.com/doc>

Додатки

Додаток А

Посилання на репозиторій з проектом: git@github.com:hvirix/Kursova_robota.git

Додаток Б

Інструкція по встановленню та запуску моого проекту:

1. Встановити на пристрій Python, Git
2. Виконати команду `git clone git@github.com:hvirix/Kursova_robota.git`
3. Перейти у відповідну папку за допомогою команди `cd`
4. Встановити Flask та відповідні бібліотеки(SQLAlchemy, PYMYSQL, dotenv etc) використовуючи менеджер пакетів `pip` та команду `pip install <назва пакету>`.
5. Налаштuvati середовище(використати змінні з файлу `.env`)
6. Запустити проект за допомогою команди `python app.py`