Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Лабораторна робота №5

Інфраструктура програмного забезпечення WEB-застосувань

Тема: Робота з файлами (на прикладі зображень)

Виконали
студенти групи ІП-11:
Головня О. Р.,
Панченко С. В.,
Сідак К. І.

Перевірив:
Орленко С. П.

Київ 2024

ЗМІСТ

1 МЕТА

Дослідити процес роботи із зображеннями у існуючому веб-застосунку, використовуючи два підходи до збереження файлів: у бази даних та у файлову систему, при цьому визначивши переваги та недоліки обох підходів.

2 ЗАВДАННЯ

Лабораторна робота присвячена розширенню функціоналу існуючого веб-застосунку та полягає у додаванні можливості завантажувати, переглядати, видаляти та замінювати зображення для сутностей у застосунку. Оскільки варіанти попередньої роботи були довільними, то прикладами роботи з зображеннями можуть слугувати, наприклад:

- фото співробітника (1 зображення на об'єкт);
- фотографії товару (кілька зображень на об'єкт – не є обов’язковою вимогою для цієї роботи).

3 ВИКОНАННЯ

3.1 Вступ та огляд проекту

Представлена система управління студентами є комплексним програмним рішенням, розробленим з використанням сучасних технологій та методологій розробки програмного забезпечення. Проект втілює передові практики веб-розробки та демонструє ефективне використання мікросервісної архітектури для створення масштабованого та надійного додатку.

Основною метою системи є забезпечення ефективного управління даними студентів та академічних груп у навчальному закладі. Система надає можливість зберігати та управляти інформацією про студентів, включаючи їхні особисті дані та фотографії, а також підтримує гнучке управління академічними групами. Особлива увага в проекті приділена забезпеченню високої продуктивності, надійності та безпеці даних.

3.2 Технологічний стек та обґрунтування вибору технологій

Серверна частина системи реалізована на мові програмування Rust, що є сучасною системною мовою програмування з акцентом на безпеку, паралельність та практичну продуктивність. Вибір Rust обумовлений декількома ключовими факторами. По-перше, система керування пам'яттю Rust забезпечує безпеку пам'яті на рівні компіляції, що виключає цілий клас потенційних помилок та вразливостей. По-друге, вбудована підтримка асинхронного програмування дозволяє ефективно обробляти велику кількість паралельних запитів. По-третє, висока продуктивність Rust робить його ідеальним вибором для серверних додатків з інтенсивною обробкою даних.

Для реалізації веб-серверу використовується фреймворк Actix-web, який є одним з найшвидших веб-фреймворків не лише в екосистемі Rust, а й серед усіх доступних веб-фреймворків. Actix-web надає потужний набір інструментів для створення веб-додатків, включаючи вбудовану підтримку WebSocket, гнучку

систему маршрутизації та ефективну обробку статичних файлів. Фреймворк також забезпечує відмінну інтеграцію з асинхронною екосистемою Rust та надає зручні абстракції для роботи з HTTP-запитами та відповідями.

3.3 Архітектура бази даних та управління даними

Система використовує гібридний підхід до зберігання даних, поєднуючи реляційну базу даних PostgreSQL та документоорієнтовану базу даних MongoDB. Таке рішення дозволяє оптимально використовувати сильні сторони кожної системи керування базами даних.

PostgreSQL використовується для зберігання структурованих даних про студентів. Схема бази даних включає детально продумані таблиці з відповідними обмеженнями та зв'язками. Основна таблиця студентів містить поля для зберігання особистої інформації, включаючи ім'я, прізвище та приналежність до групи. Особлива увага приділена зберіганню фотографій студентів, для чого реалізовано два альтернативні підходи.

Перший підхід передбачає зберігання фотографій безпосередньо в базі даних у форматі BLOB. Це забезпечує атомарність операцій та спрощує резервне копіювання, оскільки всі дані зберігаються в одному місці. Для оптимізації продуктивності при роботі з BLOB-даними використовуються спеціальні індекси та механізми кешування.

Другий підхід полягає у зберіганні фотографій у файловій системі з записом шляхів до файлів у базі даних. Цей метод забезпечує кращу продуктивність при роботі з великими файлами та дозволяє легко масштабувати систему зберігання. Для забезпечення цілісності даних використовується транзакційний підхід при збереженні файлів та оновленні записів у базі даних.

MongoDB використовується для зберігання інформації про академічні групи. Вибір документоорієнтованої бази даних для цієї задачі обумовлений необхідністю гнучкого масштабування та можливістю легко модифікувати структуру даних про групи без необхідності змін схеми бази даних. Документи

в MongoDB містять інформацію про назву групи, унікальний ідентифікатор та додаткові метадані.

Центральним елементом серверної архітектури є система маршрутизації, реалізована за допомогою Actix-web. Всі HTTP-запити проходять через систему middleware, яка забезпечує логування, обробку помилок та перевірку автентифікації. Кожен endpoint має власний обробник, який інкапсулює бізнес-логіку та взаємодію з базами даних.

Система включає механізм валідації вхідних даних, який перевіряє формат файлів, що завантажуються, а також забезпечує санітизацію текстових даних. Всі операції з файлами виконуються асинхронно, що дозволяє ефективно обробляти велику кількість паралельних запитів.

Клієнтська частина системи реалізована як Single Page Application з використанням чистого JavaScript без залежності від важких фреймворків.

Використання контейнеризації за допомогою Docker дозволяє легко розгортати додаткові екземпляри сервісів при збільшенні навантаження. Кожен компонент системи може бути масштабований незалежно, що забезпечує ефективне використання ресурсів.

Реалізовано детальне логування всіх важливих подій та помилок. Логи структуровані та містять всю необхідну інформацію для діагностики проблем.

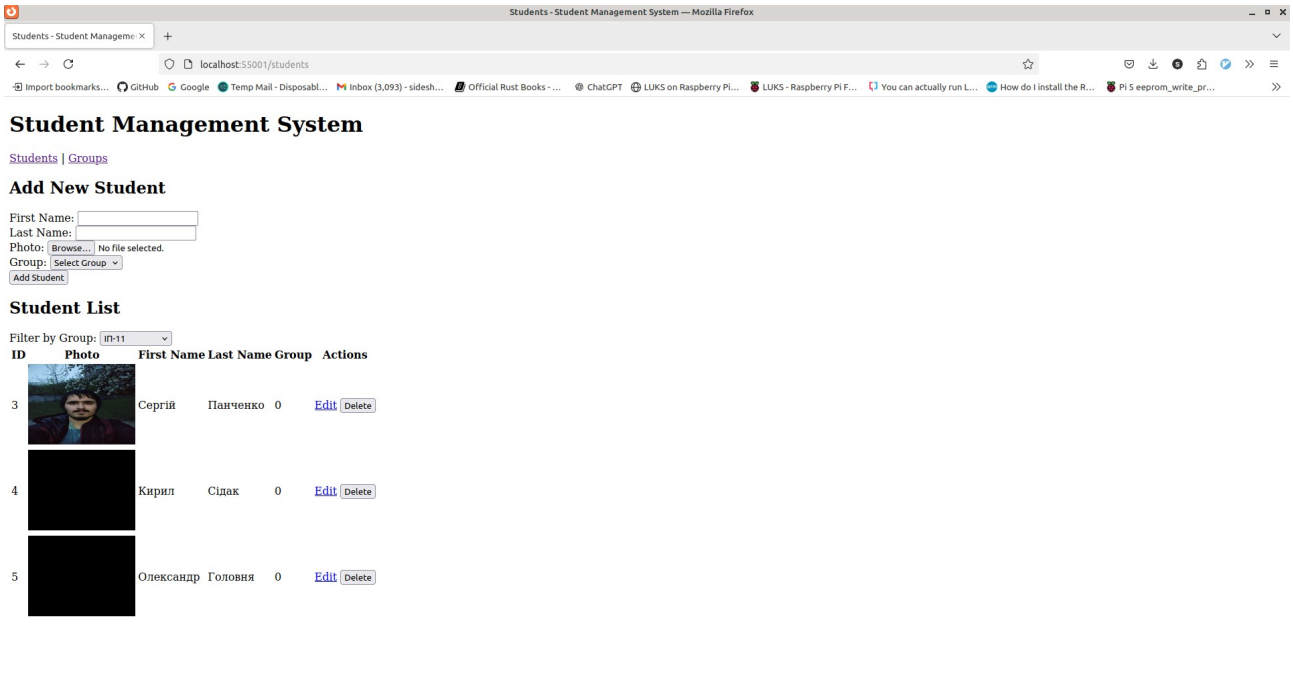

Рисунок 3.1 — Сторінка студентів

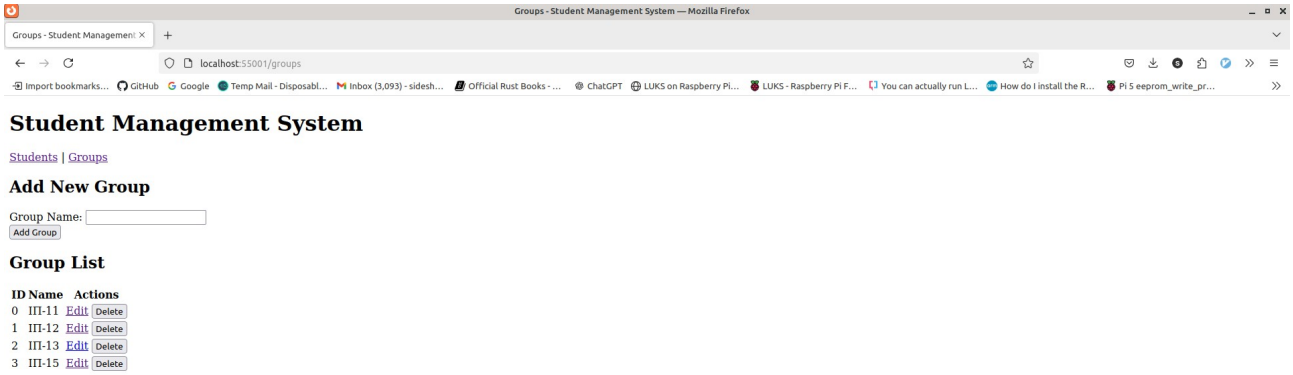

Рисунок 3.2 — Сторінка груп

ВИСНОВКИ

Розроблена система демонструє ефективне застосування сучасних технологій та підходів до розробки веб-додатків. Використання Rust в поєднанні з сучасними веб-технологіями забезпечує високу продуктивність та надійність системи. Архітектурні рішення, прийняті під час розробки, забезпечують гнучкість системи та можливість її подальшого розвитку. Використання контейнеризації спрощує розгортання та масштабування системи.