Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Технології розроблення програмного забезпечення

Лабораторна робота №3

«ДІАГРАМА РОЗГОРТАННЯ. ДІАГРАМА КОМПОНЕНТІВ. ДІАГРАМА ВЗАЄМОДІЙ ТА ПОСЛІДОВНОСТЕЙ.»

Виконав:
студент групи ІА–22:
Клочков І. Ф.

Перевірив:
Мягкий Михайло Юрійович

Київ 2024

Зміст

Тема: Діаграма розгортання. Діаграма компонентів. Діаграма взаємодій та послідовностей.

Мета: Проаналізувати тему, розробити діаграму розгортання, діаграму компонентів, діаграму взаємодій та послідовностей.

Хід роботи

..18 Shell (total commander) (state, prototype, factory method, template method, interpreter, client-server)

Оболонка повинна вміти виконувати основні дії в системі - перегляд файлів папок в файлової системі, перемикання між дисками, копіювання, видалення, переміщення об'єктів, пошук.

1. Короткі теоретичні відомості

Діаграма розгортання (Deployment Diagram)

Діаграми розгортання (Deployment Diagram) відображають фізичне розташування компонент системи та показують, на якому обладнанні запускається програмне забезпечення.

Основними елементами є вузли, що можуть бути:

- Пристроями (фізичне обладнання, наприклад, комп’ютери),
- Середовищами виконання (програмне забезпечення, яке запускає інші компоненти, наприклад, операційна система).

Вузли можуть бути з'єднані шляхами, що показують взаємодію між ними, з використанням різних протоколів чи технологій. Усередині вузлів можуть бути артефакти — файли, які реалізують компоненти системи (виконувані файли, дані тощо).

Діаграма компонентів (Component Diagram)

Діаграма компонентів відображає компоненти, залежності та зв'язки між ними. Компонент може бути представлений у вигляді бібліотеки, модуля або іншого незалежного елемента програмного коду.

Основні елементи діаграми компонентів:

- Компоненти – модулі або частини системи, які виконують певну функцію (наприклад, обробка даних, управління з'єднаннями).
- Інтерфейси – методи взаємодії між компонентами.
- Зв'язки – асоціації між компонентами, що показують, які саме компоненти використовують інші.

Діаграма послідовностей (Sequence Diagram)

Діаграма послідовностей використовується для моделювання процесів системи з погляду часової послідовності виконання дій. Вона відображає, як різні елементи системи обмінюються повідомленнями для виконання певної функції.

Основні елементи діаграми послідовностей:

- Актори – користувачі або зовнішні системи, що ініціюють певні дії.
- Об’єкти – компоненти або класи системи, що беруть участь у процесі.
- Повідомлення – передача даних між об'єктами.
- Часова шкала – вертикальна лінія, що показує часову послідовність подій.

2. Реалізація частини функціональності системи

Під час реалізації частини функціональності системи, описаної діаграмами з попередньої лабораторної роботи, було створено програмні класи сутностей: File, Directory, Drive; інтерфейси та класи репозиторіїв: IRepository IFileRepository, IDirectoryRepository, IDriveRepository, FileRepository, DirectoryRepository, DriveRepository для взаємодії з базою даних. Також були створені інтерфейси та класи сервісів: IFileService, IDirectoryService, IDriveService, FileService, DirectoryService, DriveService. Ці сервіси використовуються для взаємодії з користувачем через реалізовані контролери: FileController, DirectoryController, DriveController.
Структура проекту після реалізації частини функціональності системи зображена на рисунку 1.

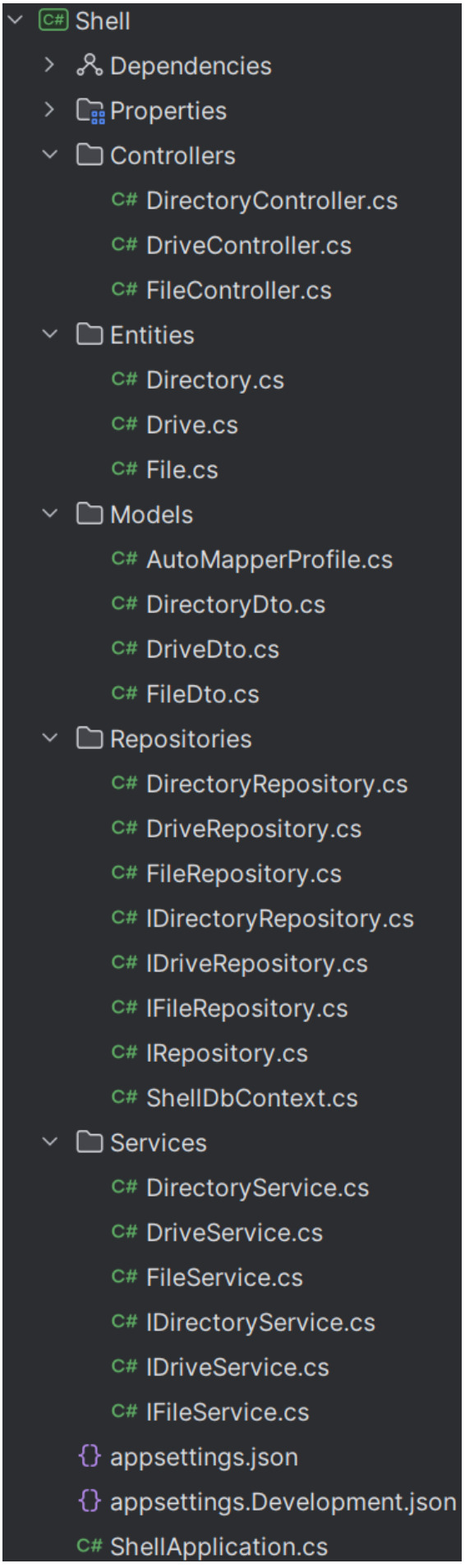

Рисунок 1 – реалізація частини функціональності системи

3. Діаграма послідовностей

Діаграма послідовностей для процесу “Створення та отримання файлу” зображена на рисунку 2.

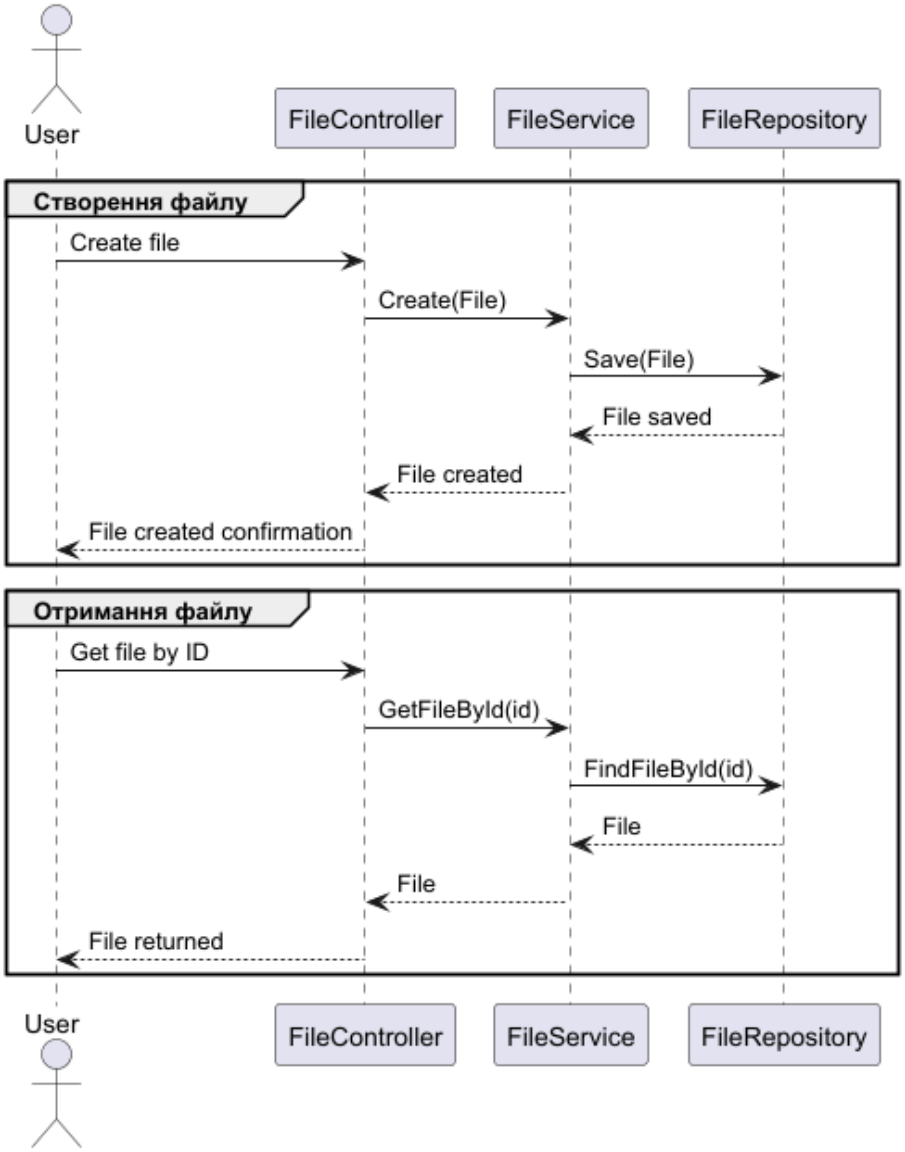

Рисунок 2 – діаграма послідовностей

Сценарій створення файлу:

1. Користувач відправляє запит на створення файлу до FileController.
2. FileController передає запит до FileService, викликаючи метод Create(File).
3. FileService викликає метод Save(File) у FileRepository для збереження файлу в базі даних.
4. FileRepository підтверджує FileService про успішне збереження файлу.
5. FileService інформує FileController про успішне створення файлу.
6. FileController надсилає користувачу підтвердження про створення файлу.

Сценарій отримання файлу:

1. Користувач відправляє запит на отримання файлу за ID до FileController.
2. FileController передає запит до FileService, викликаючи метод GetFileById(id).
3. FileService запитує файл у FileRepository, викликаючи метод FindFileById(id).
4. FileRepository повертає файл до FileService.
5. FileService передає файл до FileController.
6. FileController повертає файл користувачу.

4. Діаграма розгортання

Діаграма розгортання системи зображена на рисунку 3.

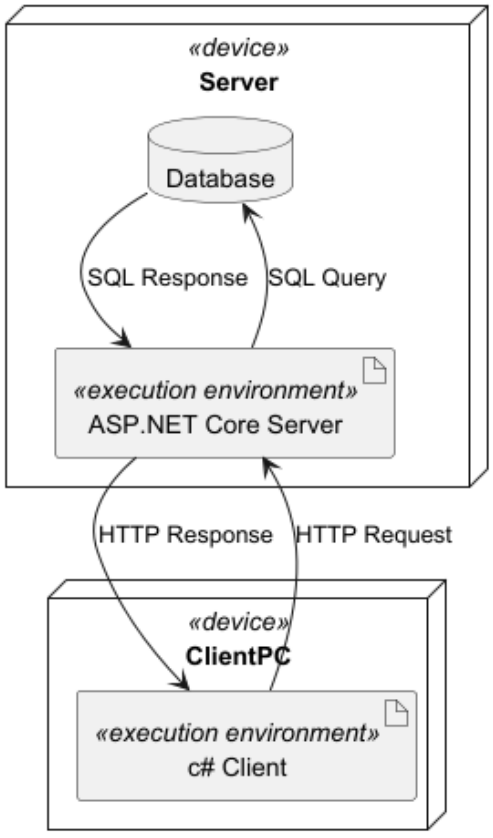

Рисунок 3 – діаграма розгортання

Дана діаграма відображає архітектуру файлової оболонки, яка складається з серверної та клієнтської частин. Сервер представлений як пристрій Server і має артефакт ASP.NET Server, який є виконуваним файлом серверної частини програми, а також базу даних Database. Клієнт теж представлений як пристрій

ClientPC, що має артефакт c# Client, який позначає виконуваний файл клієнтської частини програми. Взаємодія між компонентами відбувається за допомогою HTTP-запитів від клієнта до сервера і навпаки, та виконання SQL-запитів сервером до бази даних і отриманням результатів виконання запитів від бази даних до сервера.

На цій діаграмі зображено, як клієнтська частина програми взаємодіє з сервером і сервер взаємодіє з клієнтом та базою даних, використовуючи архітектурні елементи та їх зв’язки в системі.

5. Діаграма компонентів

Діаграма компонентів розроблюваної системи зображена на рисунку 4.

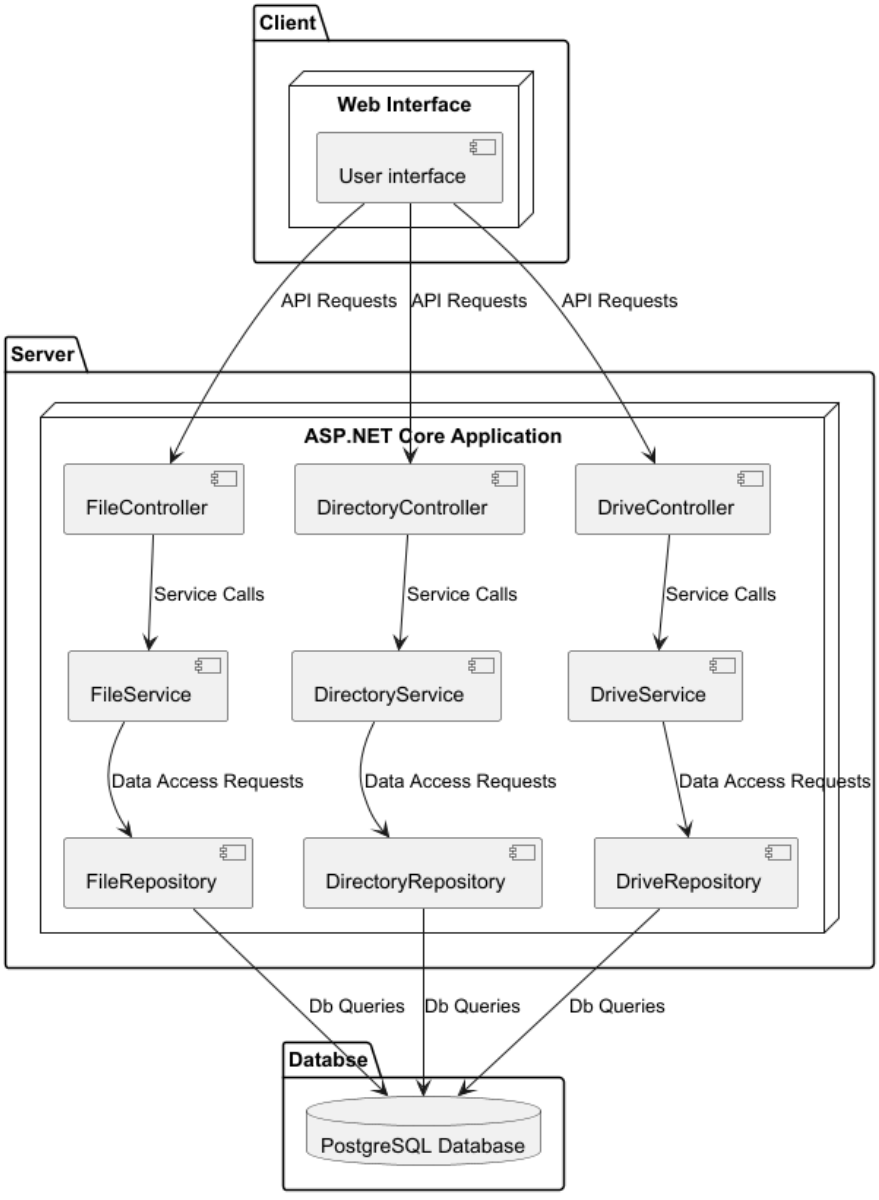

Рисунок 4 – діаграма компонентів

Діаграма компонентів для файлової складається з трьох основних частин: клієнт, сервер і база даних. На сервері розташована програма ASP.NET Core, що включає контролери (FileController, DirectoryController, DriveController), сервіси (FileService, DirectoryService, DriveService) та репозиторії (FileRepository, DirectoryRepository, DriveRepository). База даних PostgreSQL зберігає дані про файли, папки та диски. Клієнтський веб інтерфейс взаємодіє з контролерами, надсилаючи запити на сервер. Контролери обробляють ці запити, викликаючи відповідні сервіси, які в свою чергу взаємодіють із репозиторіями для збереження або отримання даних з бази даних. Ця структура демонструє, як різні компоненти системи працюють разом для виконання запитів користувача.

Посилання на код: https://github.com/ivanchikk/trpz

Висновок: виконавши дану лабораторну роботу, я проаналізував тему, розробив діаграму розгортання, діаграму компонентів, діаграму взаємодій та послідовностей.