



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

Лабораторна робота №3

Технології розроблення програмного забезпечення

Тема проєкту ‘Shell (total commander)’

Виконала: студентка групи ІА-33

Маляревич Анжела Валентинівна

Перевірив:

Мягкий Михайло Юрійович

Київ - 2025

Зміст

Теоретичні відомості.....	3
Хід роботи.....	5
Діаграма компонентів.....	7
Діаграма розгортання для проєктованої системи.....	9
Сценарій 1: Копіювання вибраних файлів/папок з розв’язанням конфліктів.....	11
Сценарій 2: Пошук файлів/папок і перехід до розташування.....	12
Сценарій 3: Створення нової папки.....	14
Застосунок:.....	16
Контрольні питання.....	19

Тема: Основи проектування розгортання.

Мета: Навчитися проектувати діаграми розгортання та компонентів для системи, що проектується, а також розробляти діаграми взаємодії, а саме діаграми послідовностей, на основі сценаріїв зроблених в попередній лабораторній роботі.

Тема 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.

Теоретичні відомості

Діаграми розгортання

Показують, на якому обладнанні або середовищі виконання працюють частини системи.

- Складаються з вузлів: **пристрої** (комп'ютери, сервери) та **середовища виконання** (ОС, вебсервери).
- Усередині вузлів знаходяться **артефакти** – файли програм, бібліотеки, сценарії, конфігурації.
- Між вузлами будуються зв'язки, що відображають спосіб комунікації (HTTP, IPC тощо).

Бувають два типи:

- **описові** – для загального бачення архітектури,

- **екземплярні** – для конкретних пристроїв і ПЗ на етапі впровадження.

Діаграми компонентів

Відображають систему як набір модулів та їхні залежності.

Типи діаграм:

- **логічні** – показують незалежні програмні модулі та їхню взаємодію,
- **фізичні** – демонструють виконувані файли (.exe, .dll) і залежності,
- **виконувані** – зосереджуються на процесах, базах даних, сторінках.

Використовуються для:

- розуміння структури вихідного коду,
- створення інсталяційних пакетів,
- забезпечення повторного використання компонентів.

Діаграми послідовностей

Відображають взаємодію між об'єктами у часі в межах певного сценарію.

Містять:

- **акторів** – користувачів чи зовнішні системи,
- **об'єкти** – з життєвим циклом (лінією життя),
- **повідомлення** – виклики методів, обмін даними,
- **активності** – відрізки часу виконання дій.

Дозволяють:

- показати логіку роботи системи крок за кроком,

- уточнити бізнес-процеси,
- виявити потенційні проблеми ще під час проєктування.

Хід роботи

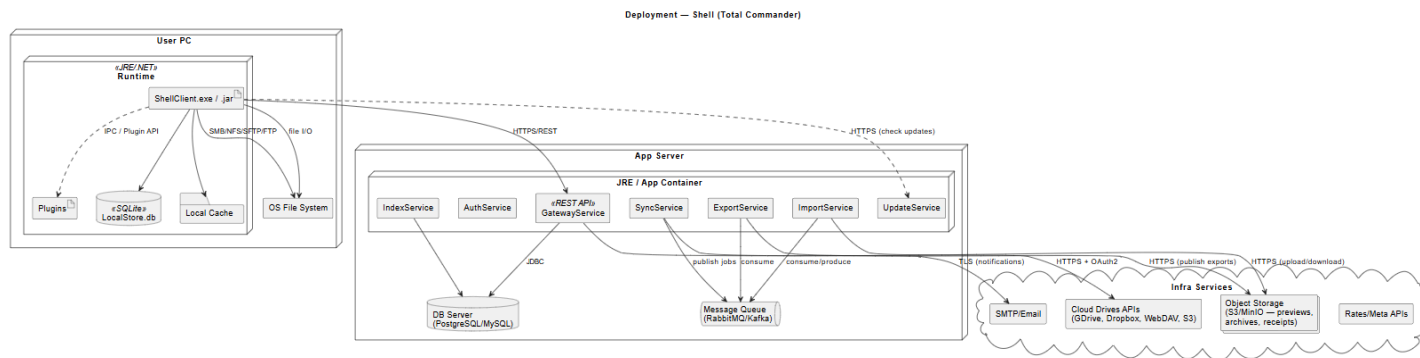


Рисунок 1 - Діаграма розгортання

Дана діаграма відображає архітектуру розгортання програмного комплексу “Shell (Total Commander)”, побудованого за принципом клієнт–серверної взаємодії з використанням сервісно-орієнтованої архітектури (SOA). На комп’ютері користувача у середовищі JRE/.NET запускається клієнтський застосунок ShellClient у вигляді виконуваного файлу або JAR-архіву. Він забезпечує користувачу доступ до файлової системи, підтримує роботу з плагінами (архіви, протоколи SFTP/FTP, інтеграція з хмарами, попередній перегляд) та використовує локальну базу SQLite для збереження історії, налаштувань та кешу мініатюр і попередніх переглядів. Клієнт напряду взаємодіє з файловою системою операційної системи, асоціаціями відкриття файлів та допоміжними утилітами (наприклад, антивірусним сканером). Для мережевих ресурсів клієнт може працювати через протоколи SMB/NFS/SFTP/FTP, а для взаємодії з серверною частиною використовується HTTPS/REST API.

На стороні застосункового сервера у контейнері JRE розгорнуті окремі сервіси, що відповідають за ключові функції системи:

- GatewayService — центральний REST-вхід для клієнтів;
- AuthService — автентифікація та управління доступом користувачів;
- IndexService — глобальний пошук по файловій системі;
- SyncService — синхронізація з хмарними сховищами та зовнішніми джерелами;
- ImportService та ExportService — імпорт/експорт великих даних та інтеграція зі сторонніми системами;
- UpdateService — перевірка та оновлення клієнтського застосунку й плагінів.

Усі сервіси взаємодіють між собою через REST API та використовують JDBC для доступу до реляційної бази даних, яка розгорнута на сервері з підтримкою PostgreSQL або MySQL. База даних зберігає інформацію про об'єкти файлової системи, користувачів, історію операцій та параметри роботи. Система також інтегрується із зовнішніми сервісами: Cloud Drives API (Google Drive, Dropbox, WebDAV, S3) для роботи з хмарними сховищами, SMTP-сервером для надсилення сповіщень і повідомлень, а також із зовнішніми API (наприклад, для отримання метаданих або довідкової інформації). Таким чином, уся система забезпечує повний набір функцій сучасного файлового менеджера: роботу з локальними й мережевими файловими системами, синхронізацію з хмарними сервісами, централізоване збереження даних, безпечну автентифікацію користувачів та зручну інтеграцію з плагінами й зовнішніми програмами.

Діаграма компонентів

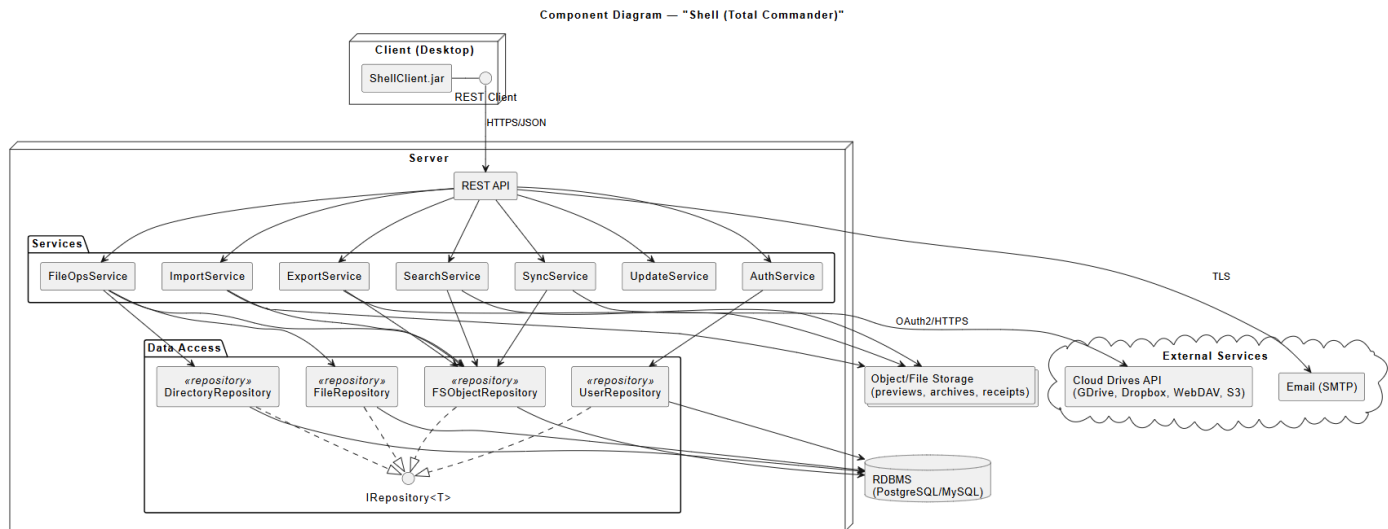


Рисунок 2 –Діаграма компонентів

Ця діаграма відображає компонентну структуру системи “Shell (Total Commander)”, організованої за сервісно-орієнтованими принципами (SOA). Клієнтська частина представлена застосунком **ShellClient.jar**, який запускається на робочому столі користувача та взаємодіє з сервером через **REST API**. Клієнт відповідає за інтерфейс користувача, роботу з локальною файловою системою та інтеграцію з плагінами.

Серверна частина включає набір сервісів:

- **FileOpsService** — виконує основні файлові операції (копіювання, переміщення, видалення, перейменування);
- **ImportService** та **ExportService** — відповідають за імпорт і експорт даних;
- **SearchService** — здійснює пошук та індексацію об’єктів;

- SyncService — синхронізує дані з хмарними сховищами та мережевими ресурсами;
- UpdateService — перевіряє й встановлює оновлення клієнтської частини та плагінів;
- AuthService — відповідає за автентифікацію користувачів і контроль доступу.

Зберігання даних реалізовано через шар доступу до даних (Data Access), який містить кілька репозиторіїв: UserRepository, FSObjectRepository, DirectoryRepository, FileRepository. Вони працюють через спільний інтерфейс IRepository<T> та взаємодіють із реляційною СУБД (PostgreSQL/MySQL). Великі об'єкти, такі як архіви, попередні перегляди або вкладення, зберігаються окремо в Object/File Storage. Система також підключається до зовнішніх сервісів: Cloud Drives API (Google Drive, Dropbox, WebDAV, S3) — для інтеграції з хмарними сховищами, та Email (SMTP) — для надсилення повідомлень і сповіщень. Таким чином, діаграма демонструє модульну архітектуру системи: клієнт працює з сервером через REST-шар, сервер складається з окремих сервісів і спільного шару доступу до даних, а інтеграція із зовнішніми сервісами розширює функціональні можливості файлового менеджера.

Діаграма розгортання для проєктованої системи

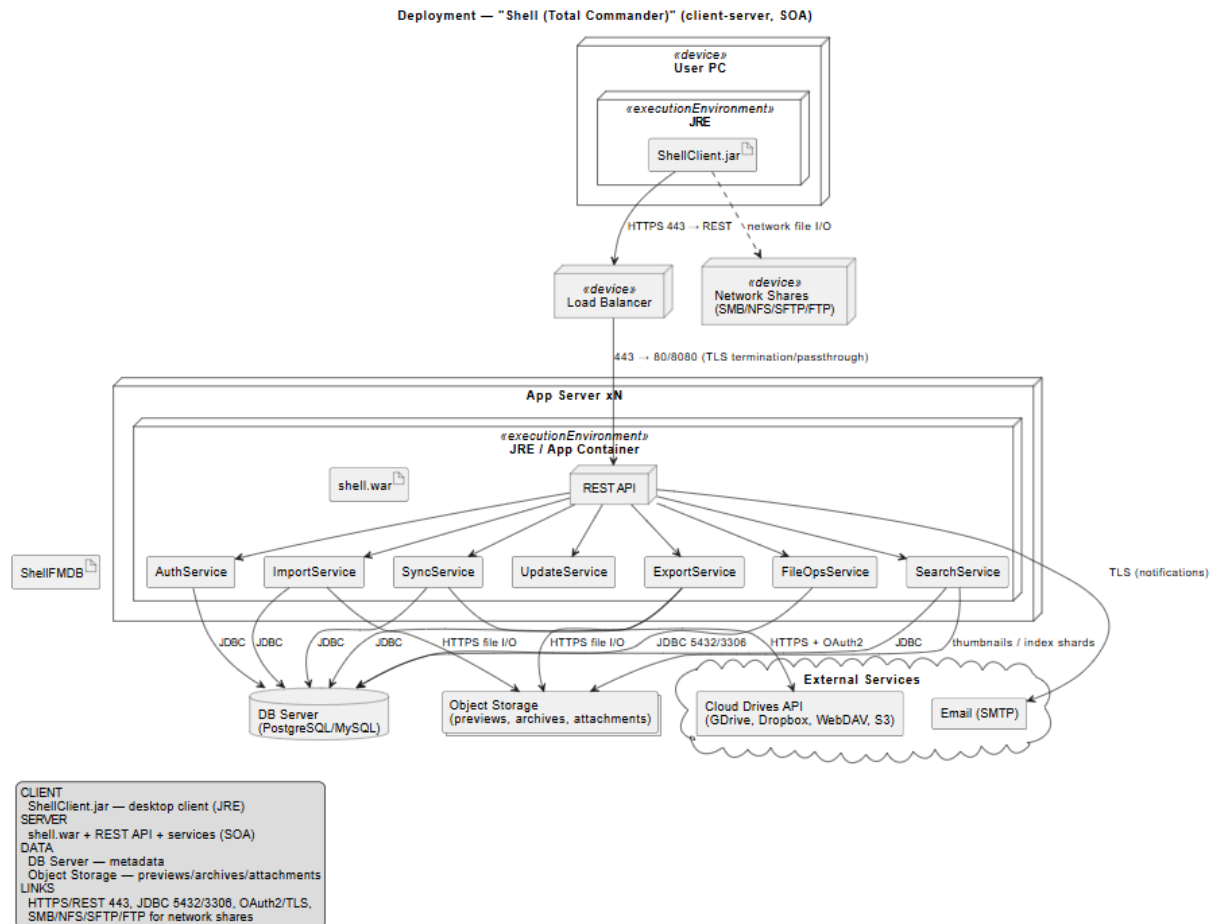


Рисунок - 3 - Діаграма розгортання для проєктованої системи

Це діаграма розгортання для проєктованої системи “Shell (Total Commander)”. Вона відображає фізичне розташування компонентів у клієнт–серверній архітектурі з використанням сервісно-орієнтованих принципів (SOA). Клієнтська частина представлена застосунком ShellClient.jar, що запускається на комп’ютері користувача у середовищі JRE. Клієнт взаємодіє із сервером через HTTPS/REST API та забезпечує користувачу доступ до файлової системи, мережових ресурсів і плагінів. Для

масштабування та відмовостійкості використовується Load Balancer, який розподіляє запити між декількома екземплярами застосункового сервера.

На стороні сервера у контейнері JRE розгортається веб-застосунок `shell.war`, що включає основні сервіси:

- FileOpsService — базові файлові операції (копіювання, переміщення, видалення, перейменування);
- SearchService — пошук та індексація об'єктів;
- SyncService — синхронізація з хмарними сховищами та мережевими дисками;
- ImportService та ExportService — обмін файлами з іншими системами;
- UpdateService — оновлення клієнта та плагінів;
- AuthService — автентифікація та управління доступом.

Для збереження метаданих використовується DB Server (PostgreSQL/MySQL) із базою ShellFMDB, а великі об'єкти, такі як архіви, прев'ю та вкладення, зберігаються в Object Storage. Система також взаємодіє із зовнішніми сервісами: Cloud Drives API (Google Drive, Dropbox, WebDAV, S3) через захищений канал HTTPS із використанням OAuth2 та Email-сервісом (SMTP) для відправки повідомлень і сповіщень через TLS. Крім того, клієнт може напряду працювати з мережевими ресурсами через SMB/NFS/SFTP/FTP. Таким чином, діаграма демонструє, як клієнт і серверні компоненти взаємодіють між собою та із зовнішніми сервісами, забезпечуючи повний набір функцій сучасного файлового менеджера.

Сценарій 1: Копіювання вибраних файлів/папок з розв'язанням конфліктів

Передумови:

- Користувач автентифікований (за потреби) і має доступ до вихідної та цільової директорій.
- У виборі (selection) є хоча б один файл/папка.

Постумови:

- Вибрані об'єкти скопійовано у цільову директорію згідно з обраною політикою розв'язання конфліктів.
- Панель оновлена: показано нові копії та підсумок операції.

Взаємодіючі сторони: Користувач, Система.

Короткий опис:

Користувач копіює один або кілька об'єктів у вказану ціль. Якщо в ціль вже існують об'єкти з такими іменами, система запитує політику: перезаписати / пропустити / перейменувати / застосувати до всіх.

Основний потік подій:

1. Користувач обирає «Копіювати», вказує цільову директорію.
2. Система перевіряє доступність цілі, за потреби створює її.
3. Для кожного об'єкта система перевіряє конфлікт імені.
4. У разі конфлікту система показує діалог вибору політики; застосовує її.
5. Система копіює об'єкт(и), оновлює вміст цільової панелі.
6. Показується підсумок: скільки успішно, пропущено, з помилкою.

Винятки:

- Недостатньо прав/місця → повідомлення та пропозиція змінити ціль.
- Ім'я занадто довге/заборонені символи → пропозиція перейменувати.
- Об'єкт зайнятий іншим процесом → пропустити або повторити пізніше.
- Переривання користувачем → частковий результат із підсумком.

Примітки:

- Для великих папок може виконуватися попередня оцінка обсягу/кількості.

- Політику «застосувати до всіх» можна скасувати впродовж операції.

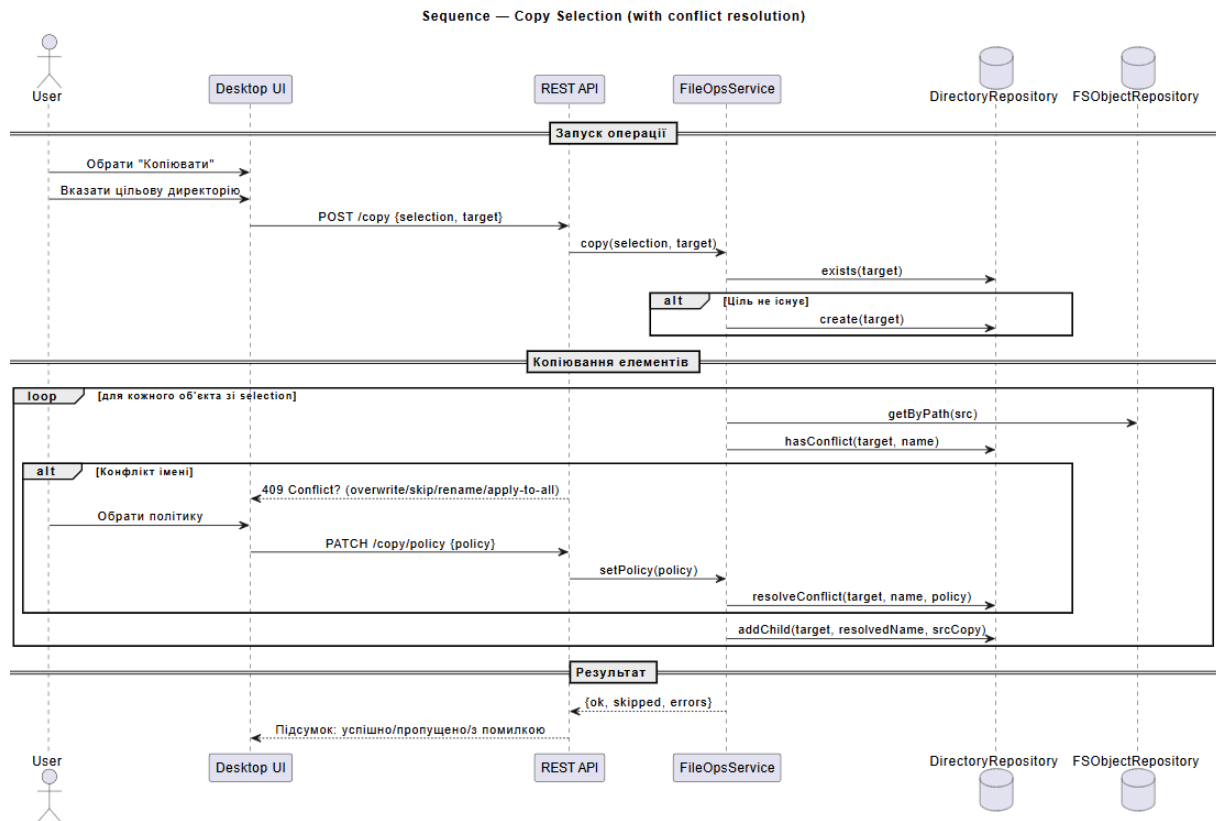


Рисунок 4 - Діаграма послідовностей для сценарію 1

Сценарій 2: Пошук файлів/папок і перехід до розташування

Передумови:

- Запущено клієнт, відкрито панель (диск/каталог) або корінь.
- Користувач знає критерії пошуку (маска, розширення, дата, розмір тощо).

Постумови:

- Відображено результати пошуку, виконано перехід до вибраного об'єкта або виконано дію над ним.

Взаємодіючі сторони: Користувач, Система.

Короткий опис:

Користувач задає критерії, система виконує рекурсивний пошук/по індексу, показує результати зі шляхами та метаданими; користувач може відкрити каталог знайденого елемента або виконати з ним дію.

Основний потік подій:

1. Користувач обирає «Пошук», вводить критерії і підтверджує.
2. Система виконує пошук (рекурсивно або за індексом), показує прогрес.
3. Повертається список результатів із базовими метаданими.
4. Користувач обирає елемент → «Перейти до каталогу» або іншу дію.
5. Панель відкривається на батьківській директорії з фокусом на елементі.

Винятки:

- Занадто загальні критерії → попередження, пропозиція звужити пошук.
- Недоступні підкаталоги → позначка в результатах, частковий список.
- Порожні критерії → підказка заповнити хоча б одну умову.

Примітки:

- Результати можна відфільтрувати/відсортувати (ім'я, дата, розмір).
- Пошук підтримує шаблони та збережені профілі.

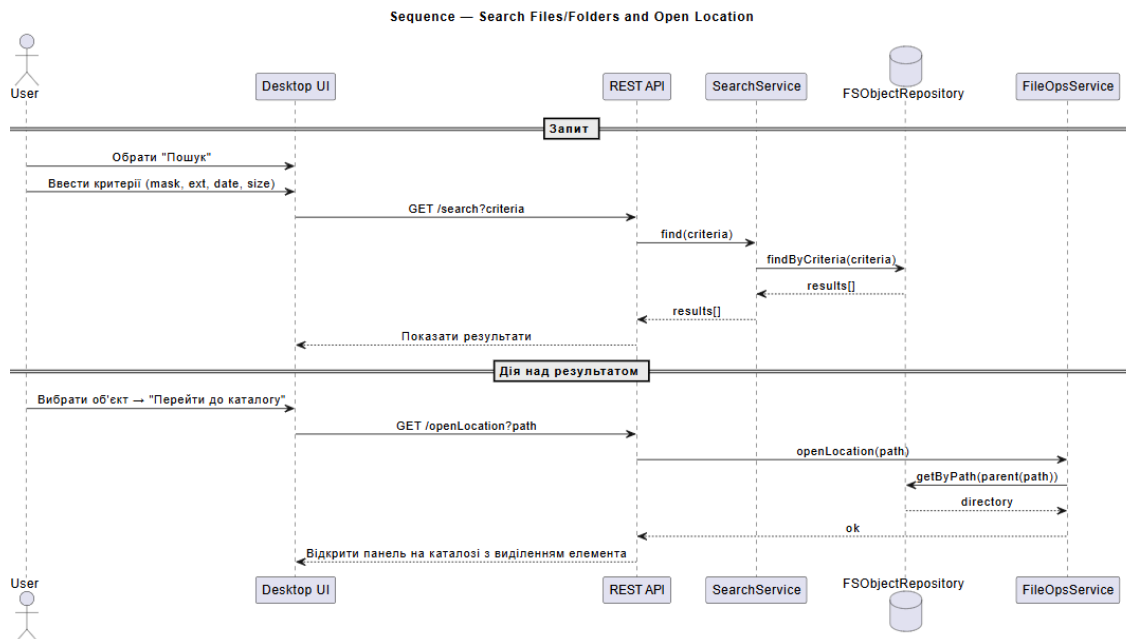


Рисунок 5 - Діаграма послідовностей для сценарію 2

Сценарій 3: Створення нової папки

Передумови:

- Користувач має права на створення в поточній директорії.
- Відкрита панель у цільовому каталозі.

Постумови:

- У поточній директорії з'являється нова папка з валідною назвою.
- Папка виділена в панелі й доступна для подальших дій.

Взаємодіючі сторони: Користувач, Система.

Короткий опис:

Користувач вводить назву нової папки; система перевіряє валідність і відсутність дублікату, створює директорію та оновлює панель.

Основний потік подій:

1. Користувач обирає «Створити папку».
2. Система запитує назву, користувач вводить і підтверджує.
3. Система перевіряє: існування батьківського каталогу, права доступу, валідність імені, відсутність дублікату.
4. Створюється директорія; панель оновлюється з виділенням нової папки.

Винятки:

- Папка з такою назвою вже існує → система пропонує іншу назву.
- Недостатньо прав/некоректне ім'я/помилка ФС → повідомлення та повторне введення/скасування.

Примітки:

- За замовчуванням може підставлятися «New Folder» з миттєвим редагуванням.
- Можлива автоперевірка недопустимих символів і максимальної довжини імені.

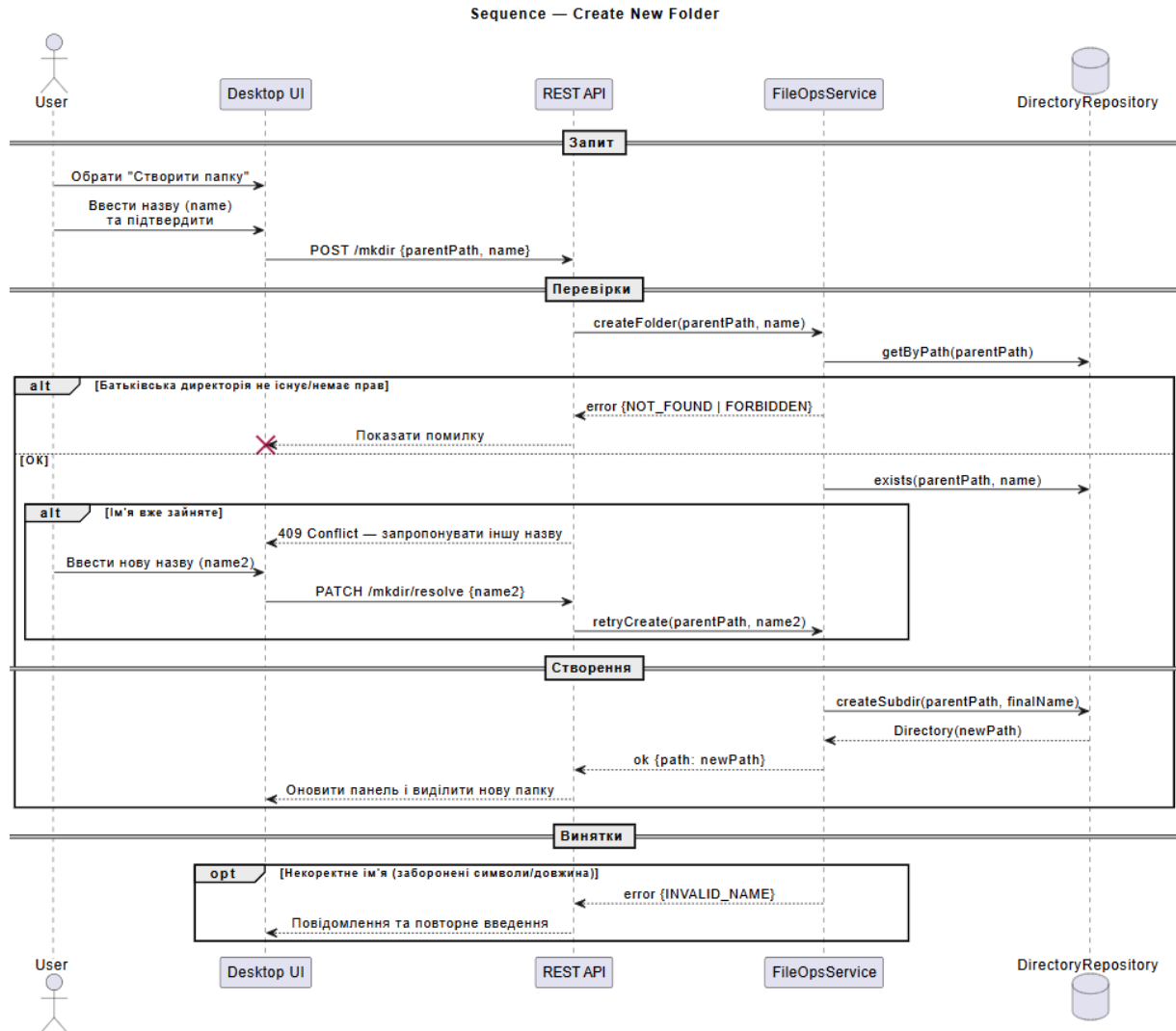
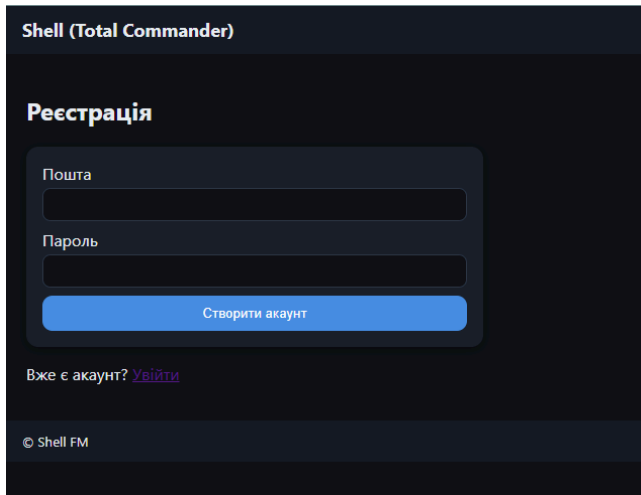


Рисунок 6 - Діаграма послідовностей для сценарію 3

Застосунок:

Сторінка реєстрації нового користувача (</register>)

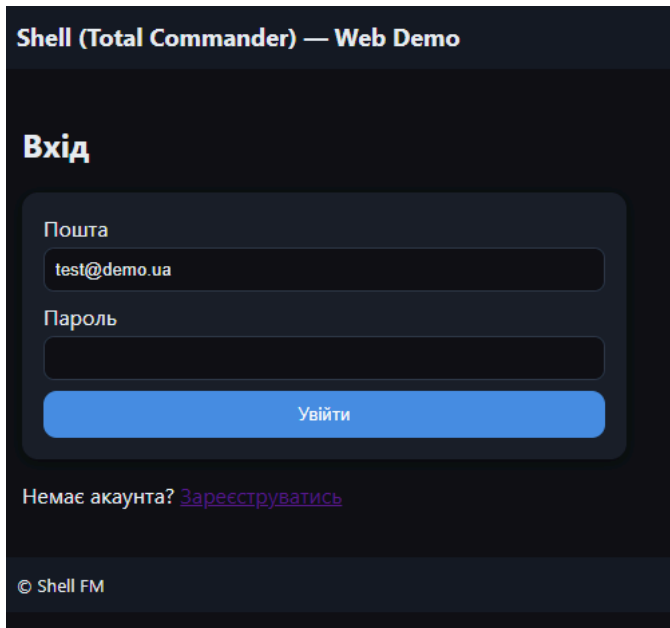
Форма містить поля для введення пошти та пароля. Після натискання кнопки «Створити акаунт» створюється новий об'єкт користувача (User), який зберігається в базі даних через UserRepository.



The screenshot shows a web browser window titled "Shell (Total Commander)". The page has a dark theme and is titled "Реєстрація" (Registration). It contains two input fields: "Пошта" (Email) and "Пароль" (Password). Below these fields is a blue button labeled "Створити акаунт" (Create account). At the bottom of the form, there is a link "Вже є акаунт? [Увійти](#)" (Already have an account? [Login](#)). The footer of the page displays "© Shell FM".

Сторінка входу в систему (</login>)

Після натискання кнопки «Увійти» дані перевіряються через сервіс [AuthService](#), який звертається до [UserRepository](#) та звіряє пароль



The screenshot shows a web browser window titled "Shell (Total Commander) — Web Demo". The page has a dark theme and is titled "Вхід" (Login). It contains two input fields: "Пошта" (Email) with the value "test@demo.ua" and "Пароль" (Password). Below these fields is a blue button labeled "Увійти" (Login). At the bottom of the form, there is a link "Немає акаунта? [Зареєструватись](#)" (Don't have an account? [Register](#)). The footer of the page displays "© Shell FM".

Сторінка пошуку файлів (/search)

Це основна робоча форма системи.

Користувач може задати параметри пошуку:

- маску імені файлів (наприклад *report*),
- розширення файлів (наприклад pdf).

Після натискання кнопки «Шукати» створюється об'єкт SearchQuery, який зберігається в таблиці SEARCH_QUERY.

Сервіс SearchService виконує пошук у файловій системі (у межах домашньої директорії користувача) і зберігає знайдені результати у таблиці SEARCH_RESULT.

Shell (Total Commander) — Web Demo

Пошук файлів (демо)

Маска імені (* дозволено):

Розширення (без крапки):

Шукати

Мої запити

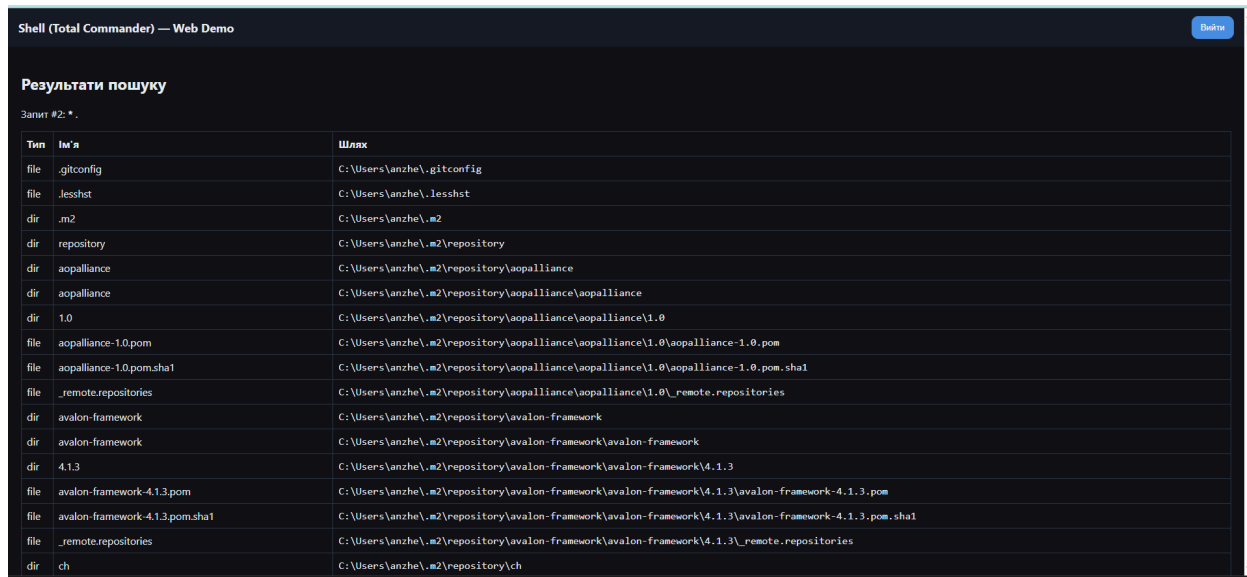
[Запит #2 — *](#), створено 2025-10-23 15:50

[Запит #1 — *](#), створено 2025-10-23 15:46

© Shell FM

Сторінка результатів пошуку (/results/{id})

Після виконання запиту користувача автоматично перенаправляє на цю сторінку. Вона відображає таблицю з результатами пошуку, зчитаними з бази даних. Для кожного запису показано тип об'єкта (файл або каталог), його назву та повний шлях. Реалізована логіка вибірки даних із таблиць SEARCH_QUERY і SEARCH_RESULT через репозиторій SearchResultRepository.



Тип	Ім'я	Шлях
file	.gitconfig	C:\Users\anzhe\.gitconfig
file	.lessht	C:\Users\anzhe\.lessht
dir	.m2	C:\Users\anzhe\.m2
dir	repository	C:\Users\anzhe\.m2\repository
dir	aopalliance	C:\Users\anzhe\.m2\repository\aopalliance
dir	aopalliance	C:\Users\anzhe\.m2\repository\aopalliance\aopalliance
dir	1.0	C:\Users\anzhe\.m2\repository\aopalliance\aopalliance\1.0
file	aopalliance-1.0.pom	C:\Users\anzhe\.m2\repository\aopalliance\aopalliance\1.0\aopalliance-1.0.pom
file	aopalliance-1.0.pom.sha1	C:\Users\anzhe\.m2\repository\aopalliance\aopalliance\1.0\aopalliance-1.0.pom.sha1
file	_remote.repositories	C:\Users\anzhe\.m2\repository\aopalliance\aopalliance\1.0_remote.repositories
dir	avalon-framework	C:\Users\anzhe\.m2\repository\avalon-framework
dir	avalon-framework	C:\Users\anzhe\.m2\repository\avalon-framework\avalon-framework
dir	4.1.3	C:\Users\anzhe\.m2\repository\avalon-framework\avalon-framework\4.1.3
file	avalon-framework-4.1.3.pom	C:\Users\anzhe\.m2\repository\avalon-framework\avalon-framework\4.1.3\avalon-framework-4.1.3.pom
file	avalon-framework-4.1.3.pom.sha1	C:\Users\anzhe\.m2\repository\avalon-framework\avalon-framework\4.1.3\avalon-framework-4.1.3.pom.sha1
file	_remote.repositories	C:\Users\anzhe\.m2\repository\avalon-framework\avalon-framework\4.1.3_remote.repositories
dir	ch	C:\Users\anzhe\.m2\repository\ch

Висновок: У ході виконання лабораторної роботи я доопрацювала програмну частину системи «Shell (Total Commander)» відповідно до розроблених діаграм розгортання та компонентів. Реалізовано повноцінну клієнт–серверну архітектуру, що забезпечує повний цикл роботи з даними — від введення на формі до збереження інформації в базі даних і подальшого відображення результатів на інтерфейсі користувача. У проєкті створено основні візуальні форми: сторінку авторизації та реєстрації користувачів і сторінку пошуку файлів з відображенням результатів. Дані користувачів, пошукових запитів і знайдених файлів зберігаються в реляційній базі даних H2 або PostgreSQL за допомогою шаблону Repository, що забезпечує розмежування між рівнями застосунку. Система підтримує автентифікацію користувачів, пошук файлів за маскою й розширенням, збереження історії пошуків і перегляд результатів після перезапуску.

Контрольні питання

1. Що собою становить діаграма розгортання?

Діаграма розгортання UML показує фізичну структуру системи — які програмні компоненти розміщені на апаратних або програмних вузлах і як вони взаємодіють у середовищі виконання.

2. Які бувають види вузлів на діаграмі розгортання?

Існують апаратні вузли (сервери, комп'ютери, мобільні пристрої) і програмні вузли (контейнери, середовища виконання, віртуальні машини). Вони можуть бути вкладеними одне в одне.

3. Які бувають зв'язки на діаграмі розгортання?

Використовуються асоціації (фізичні зв'язки між вузлами), комунікаційні лінії (мережеві з'єднання) і залежності (використання одного елемента іншим).

4. Які елементи присутні на діаграмі компонентів?

На діаграмі компонентів зображають компоненти системи, їхні інтерфейси, зв'язки залежності, артефакти, що реалізують ці компоненти, і підсистеми для групування.

5. Що становлять собою зв'язки на діаграмі компонентів?

Зв'язки відображають відносини залежності між компонентами — один компонент може використовувати інтерфейс іншого або бути реалізованим через певний артефакт.

6. Які бувають види діаграм взаємодії?

Основні види діаграм взаємодії: діаграма послідовностей, діаграма комунікації, діаграма часу (timing) і діаграма огляду взаємодій. Вони описують динамічні процеси системи.

7. Для чого призначена діаграма послідовностей?

Діаграма послідовностей використовується для показу порядку обміну повідомленнями між об'єктами під час виконання сценарію, що дозволяє зрозуміти логіку роботи системи.

8. Які ключові елементи можуть бути на діаграмі послідовностей?

До ключових елементів діаграми послідовностей належать актори, об'єкти або класи, життєві лінії (lifelines), повідомлення, фрагменти

типу alt чи loop та умови виконання.

9. Як діаграми послідовностей пов'язані з діаграмами варіантів використання?

Діаграма варіантів використання описує функції системи з позиції користувача, а діаграма послідовностей деталізує конкретний сценарій, показуючи взаємодію між об'єктами.

10. Як діаграми послідовностей пов'язані з діаграмами класів?

Діаграма класів описує структуру системи, а діаграма послідовностей — поведінку об'єктів, створених на основі цих класів, демонструючи, як вони співпрацюють у певному процесі.