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**
з предмету «Технології розроблення програмного забезпечення»
на тему «**Діаграма розгортання. Діаграма компонентів. Діаграма взаємодій та послідовностей**»

Виконав
студент групи ІА-23:
Воронюк Є. В.

Перевірив:
Мягкий М. Ю.

Київ 2024

**Зміст**

**Мета:** розробка та аналіз концептуальної моделі для обраного варіанту з використанням UML-діаграм. Зокрема створення діаграми розгортання, компонентів, взаємодій та послідовностей.

Завдання 1: ознайомитися з короткими теоретичними відомостями.
Завдання 2: розробити діаграму розгортання для проектованої системи.
Завдання 3: розробити діаграму компонентів для проектованої системи.
Завдання 4: розробити діаграму послідовностей для проектованої системи.
Завдання 5: скласти звіт про виконану роботу.

**Тема для виконання:**

**22** - **FTP-server (state, builder, memento, template method, visitor, client- server)**

FTP-сервер повинен вміти коректно обробляти і відправляти відповіді по протоколу FTP, з можливістю створення користувачів (з паролями) і доступних їм папок, розподілу прав за стандартною схемою (rwe), ведення статистики з'єднань, обмеження максимальної кількості підключень і максимальної швидкості поширення глобально і окремо для кожного облікового запису.

**Короткі теоретичні відомості:**

1. Діаграма розгортання (Deployment Diagram)

Діаграми розгортання є важливим інструментом для візуалізації фізичного розміщення програмних компонентів у системі. Вони показують, як програмне забезпечення розгорнуте на апаратному забезпеченні, і які зв'язки існують між різними частинами системи. Основні елементи діаграми включають:

**Вузли (Nodes):** Це фізичні або віртуальні пристрої, які можуть містити програмне забезпечення. Вузли поділяються на:

**Пристрої (Devices):** Фізичні елементи, такі як сервери, комп'ютери, маршрутизатори. Вони представляють апаратне забезпечення, на якому запускається програмне забезпечення.

**Середовища виконання (Execution Environments):** Програмні платформи, такі як операційні системи або сервери додатків, які можуть містити інші програмні компоненти. Вони забезпечують середовище для виконання програмного забезпечення.

**Зв'язки (Connections):** Визначають, як вузли взаємодіють між собою, зазвичай через мережеві протоколи або інші технології зв'язку. Зв'язки можуть мати атрибути, такі як назва протоколу (наприклад, HTTP, IPC) або технології (наприклад, .NET Remoting, WCF).

**Артефакти (Artifacts):** Файли або інші фізичні прояви програмного забезпечення, такі як виконувані файли, бібліотеки, конфігураційні файли. Вони представляють програмне забезпечення, яке розгортається на вузлах.

Діаграми розгортання можуть бути описовими, без конкретних деталей про обладнання, або екземплярними, з конкретними деталями про апаратне забезпечення та програмне забезпечення. Описові діаграми корисні на ранніх етапах проектування, тоді як екземплярні діаграми використовуються на завершальних стадіях розробки.

2. Діаграма компонентів (Component Diagram)

Діаграми компонентів описують структуру системи через її модулі або компоненти. Вони допомагають зрозуміти, як різні частини системи взаємодіють одна з одною. Основні види діаграм компонентів:

**Логічні:** Відображають систему як набір автономних модулів, які взаємодіють між собою. Це допомагає візуалізувати архітектуру системи на концептуальному рівні. Кожен компонент може бути взаємозамінним і не обов'язково знаходиться в межах одного фізичного пристрою.

**Фізичні:** Показують, як компоненти розподілені між різними фізичними вузлами системи. Цей підхід застарів і зазвичай замінюється діаграмами розгортання.

**Виконувані:** Кожен компонент представляє собою файл або набір файлів, які можуть бути виконані, такі як .exe, бібліотеки або HTML-сторінки. Це дозволяє візуалізувати систему на рівні виконуваних файлів або процесів.

Діаграми компонентів допомагають візуалізувати загальну структуру коду, специфікувати виконувані варіанти системи та забезпечити повторне використання коду. Вони також можуть включати інтерфейси та схеми баз даних для більш детального уявлення про систему.

3. Діаграма взаємодій та послідовностей (Interaction and Sequence Diagrams)

Діаграми послідовностей використовуються для моделювання динамічної поведінки системи, показуючи, як об'єкти взаємодіють один з одним у певній послідовності. Вони допомагають зрозуміти, як різні частини системи взаємодіють у часі. Основні елементи включають:

**Стан дії (Action State):** Відображає виконання окремих дій або кроків алгоритму. Кожна дія має вхідний і вихідний перехід. Стан дії зазвичай моделює один крок виконання алгоритму або потоку управління.

**Переходи:** Нетриггерні переходи, які виконуються після завершення дії. Вони можуть бути умовними, з використанням сторожових умов. Переходи дозволяють моделювати розгалуження та паралельні процеси.

**Дорожки (Swimlanes):** Використовуються для розподілу дій між різними підрозділами організації, що дозволяє моделювати бізнес-процеси. Кожна дорожка представляє окремий підрозділ або роль, відповідальну за виконання певних дій.

Діаграми взаємодій та послідовностей допомагають візуалізувати алгоритми виконання, потоки управління та бізнес-процеси, фокусуючи увагу на результатах і змінах стану системи. Вони є важливим інструментом для розуміння динамічної поведінки системи та взаємодії між її компонентами.

**Хід роботи::**

**Завдання 2:**

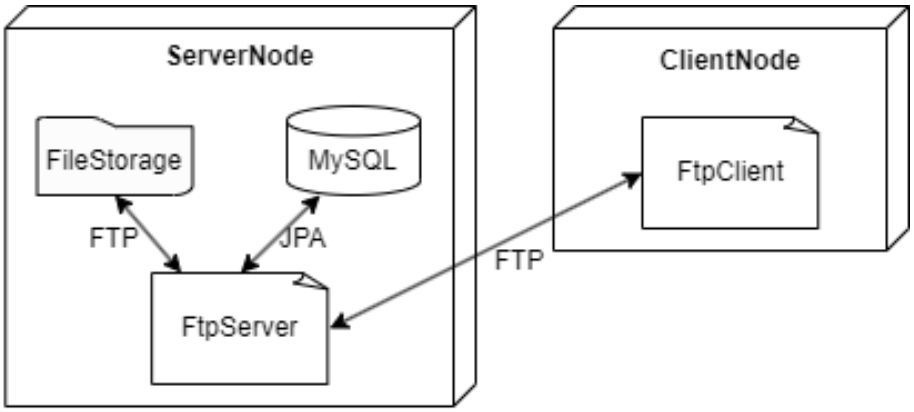


Рисунок 1 – діаграма розгортання

Короткий опис діаграми розгортання:
Діаграма ілюструє архітектурну конфігурацію та взаємодію між компонентами у розгорнутій системі.

**ServerNode**: цей вузол представляє серверну сторону системи, де розгорнуто ftp-сервер та пов'язані компоненти. ftpServer – це артефакт, що представляє сам ftp-сервер, який дозволяє здійснювати ftp-з'єднання та виконувати операції над файлами на сервері. MySql – артефакт бази даних, який використовується для зберігання даних сервера. Взаємодія між ftp-сервером та базою даних відбувається за допомогою технології JPA (Java Persistence API). FilesStorage – це артефакт, що представляє сховище файлів на сервері; ftp-сервер взаємодіє із цим сховищем для збереження та доступу до файлів.

**ClientNode**: цей вузол представляє клієнтську сторону системи, де розгорнуто ftp-клієнт. FtpClient – це артефакт, що представляє ftp-клієнта, який взаємодіє із ftp-сервером за допомогою ftp-протоколу.

**Завдання 3:**

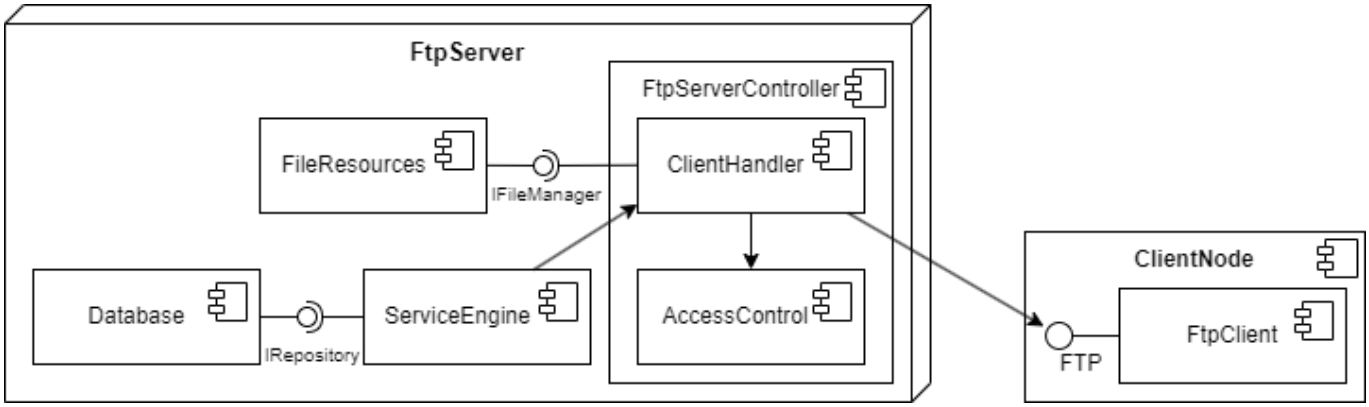


Рисунок 2 – діаграма компонентів

Короткий опис діаграми компонентів:

**ClientNode** – виступає в ролі клієнтського вузла, що включає компонент: FtpClient – взаємодіє із FtpServer через інтерфейс FTP, надсилаючи запити на підключення, передачу даних та аутентифікацію.

**FtpServer** – головний компонент серверної частини, який включає кілька підсистем для обробки запитів клієнтів та управління ресурсами.

**FtpServerCotroller** – відповідає за управління процесами на сервері та взаємодію з іншими компонентами серверної частини. Складається із компонентів ClientHandler та AccessControl. ClientHandler – обробляє з'єднання з клієнтами, приймає їхні запити та направляє їх на відповідну обробку. AccessControl – відповідає за аутентифікацію та авторизацію користувачів, забезпечуючи контроль доступу до ресурсів сервера.

**ServiceEngine** – службовий компонент для виконання операцій, пов'язаних із запитами користувачів. Використовує інтерфейс iRepository для взаємодії з базою даних, та керує запитами на доступ до даних користувача і ресурсів.

**Database** – компонент для зберігання та отримання даних користувача. Працює через інтерфейси та забезпечує зберігання інформації про користувачів, сесії та інші необхідні дані для роботи сервера.

**FileResources** – компонент, який містить файли та ресурси, доступні для користувачів FTP. Використовує інтерфейс IFileManager для обробки запитів на збереження, отримання та передачу файлів через FtpServerController.

Взаємодії між компонентами:

ClientNode (через FtpClient) підключається до FtpServer, надсилаючи запити на доступ до файлів і автентифікацію.

FtpServerController (через ClientHandler) приймає ці запити та використовує AccessControl для перевірки прав доступу.

AccessControl звертається до ServiceEngine для автентифікації користувача, яка вимагає отримання інформації з Database.

**Завдання 4:**

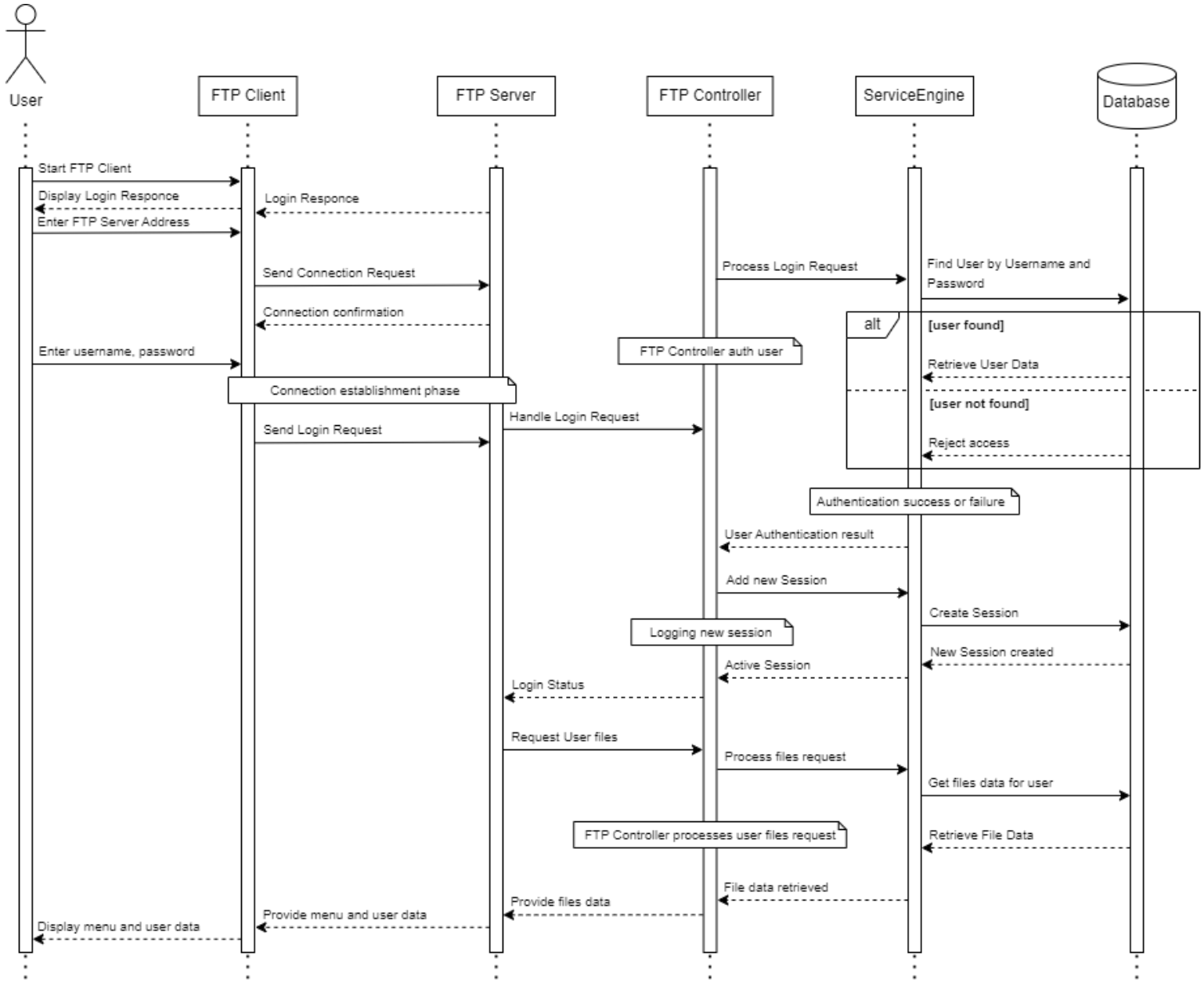


Рисунок 3 – діаграма послідовностей для входу користувача

Короткий опис діаграми послідовностей:

Дана діаграма послідовностей ілюструє процес входу користувача на FTP-сервер і взаємодію між різними компонентами системи.

**User** – актор, що починає взаємодію із системою, запускаючи FTP-клієнт.

**FTP Client** – комп’ютерна програма або термінал для спрощення доступу до ftp-сервера.

**FTP Server** – це сервер, що використовує протокол FTP, отримує запити від користувачів та оброблює їх.

**FTP Controller** – обробник команд FTP, який взаємодіє з сервером FTP та іншими компонентами для обробки запитів користувача.

**ServiceEngine** - компонент, який відповідає за обробку запитів та автентифікацію користувачів.

**Database** - база даних, де зберігаються дані користувачів та інформація про файли.

Детальний опис кроків процесу входу користувача:

- Користувач починає взаємодію з системою, запускаючи FTP-клієнт. Користувач отримує екран для введення логіна.
- Користувач вводить FTP-адресу сервера. FTP-клієнт відправляє запит на підключення до FTP-сервера.
- FTP-сервер відповідає на запит підключення, надсилаючи відповідь, що з’єднання підтверджене. FTP-клієнт відображає відповідь користувачеві.
- Користувач вводить свої облікові дані (ім'я користувача та пароль) у FTP-клієнті.
- Відбувається етап встановлення з'єднання: FTP-клієнт надсилає запит на вхід до FTP-сервера з обліковими даними користувача.
- FTP-сервер отримує та обробляє запит на вхід, передаючи його до FTP-контролера для перевірки облікових даних.
- FTP-контролер виконує автентифікацію користувача, передаючи облікові дані до ServiceEngine.
- ServiceEngine перевіряє облікові дані користувача в базі даних: якщо користувач знайдений, база даних надсилає дані користувача до ServiceEngine. У разі [user not found], доступ відхиляється.
- Після перевірки облікових даних: ServiceEngine повертає результат автентифікації до FTP-контролера (успіх або помилка).
- Якщо автентифікація успішна, FTP-контролер додає нову сесію для користувача та надсилає статус активної сесії назад до FTP-сервера.
- FTP-сервер відправляє користувачеві статус входу та активної сесії через FTP-клієнт.
- Після успішного входу FTP-клієнт надсилає запит на отримання файлів користувача через FTP-сервер до FTP-контролера.

- FTP-контролер обробляє запит на файли користувача: FTP-контролер передає запит на отримання даних файлів до ServiceEngine.
- ServiceEngine звертається до Бази даних для отримання даних про файли користувача. База даних надсилає дані файлів назад до ServiceEngine.
- ServiceEngine передає отримані дані файлів назад до FTP-контролера.
- FTP-контролер надсилає дані файлів користувача до FTP-сервера, який передає їх до FTP-клієнта.
- FTP-клієнт відображає меню з даними користувача та списком файлів, завершуючи процес входу.

**Вихідний код системи:**

Вихідний код системи розміщено на GitHub: [посилання]

**Висновок:** під час виконання даної лабораторної роботи було розроблено та проаналізовано концептуальну модель системи FTP-серверу з використанням UML-діаграм. Зокрема створення діаграми розгортання, компонентів, та послідовностей.