

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота № 9

з дисципліни «Технології розроблення
програмного забезпечення»

Тема: «Взаємодія компонентів системи.»

Виконав:
студент групи ІА-32
Задорожний Костянтин
Леонідович

Перевірив:
Мякий Михайло
Юрійович

Київ - 2025

Тема: Взаємодія компонентів системи.

Мета: Вивчити види взаємодії додатків (Client-Server, Peer-to-Peer, Serviceoriented Architecture), та реалізувати в проєктованій системі одну із архітектур.

Завдання

- Ознайомитись з короткими теоретичними відомостями.
- Реалізувати частину функціоналу робочої програми у вигляді класів та їхньої взаємодії для досягнення конкретних функціональних можливостей.
- Реалізувати функціонал для роботи в розподіленому оточенні відповідно до обраної теми.
- Реалізувати взаємодію розподілених частин:
 - Для клієнт-серверних варіантів: реалізація клієнтської і серверної частини додатків, а також загальної частини (middleware); зв'язок клієнтської і серверної частин за допомогою WCF, TcpClient, .NETRemoting на розсуд виконавця.
 - Для однорангових мереж: реалізація взаємодії клієнтських додатків за допомогою WCF Peer to peer channel.
 - Для SOA додатків: реалізація сервісу, що надає послуги клієнтським застосуванням; викладання сервісу в хмару або підняття у вигляді Web Service на локальній машині; використання токенів для передачі даних про автентифікації, двостороннє шифрування.
- Підготувати звіт щодо виконання лабораторної роботи. Поданий звіт повинен містити: діаграму класів, яка представляє спроектовану архітектуру. Навести фрагменти програмного коду, які є суттєвими для відображення реалізованої архітектури.

Предметна область:

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

FTP-сервер повинен вміти коректно обробляти і відправляти відповіді п 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.

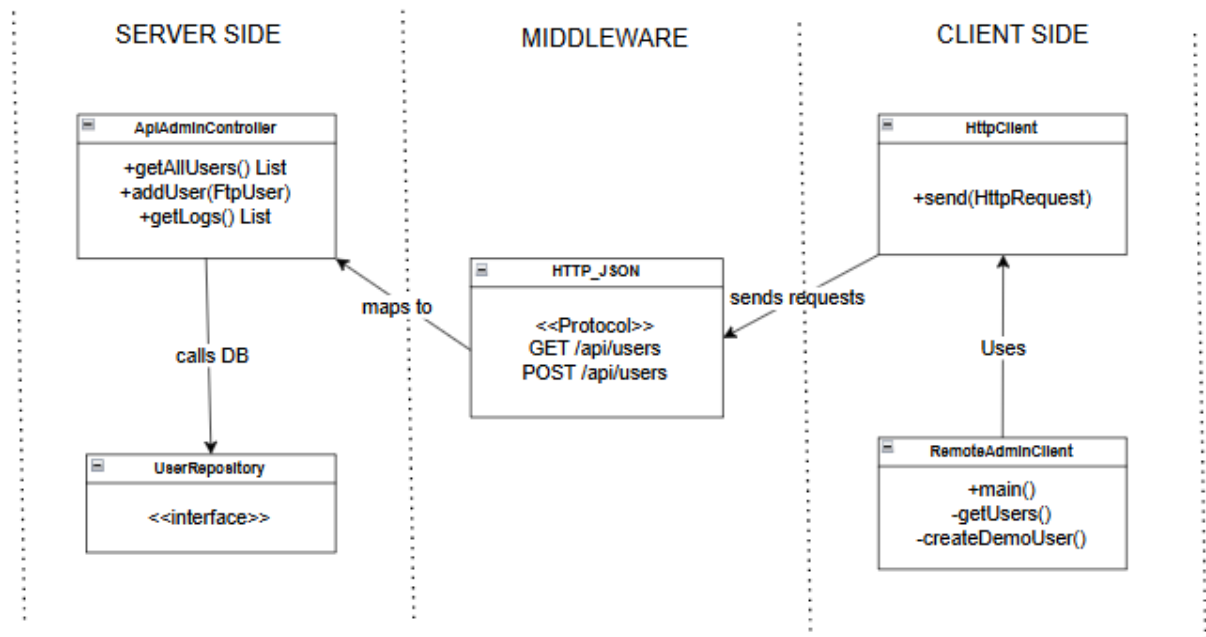
Хід роботи

Для виконання роботи я реалізував архітектуру Клієнт-Сервер (Client-Server). Оскільки проєкт написаний на Java (Spring Boot), використовувати технології .NET (WCF, Remoting) недоцільно. Я використав стандартний для Java підхід: **REST API** поверх HTTP.

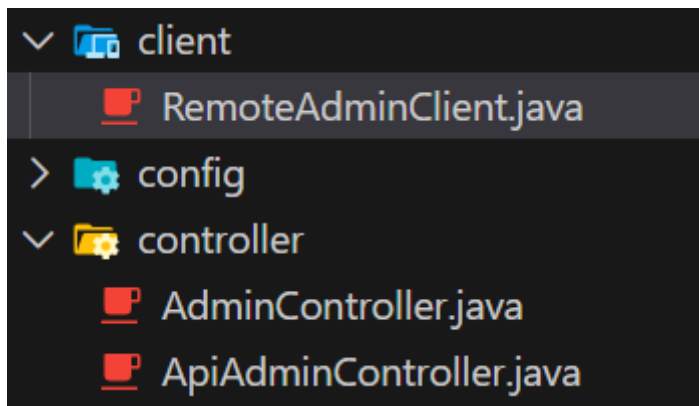
Суть реалізації:

1. **Серверна частина (Server):** Вже існуючий FTP-менеджер. Додам до нього REST контролер (ApiAdminController), який буде віддавати дані у форматі JSON, а не HTML.
2. **Клієнтська частина (Client):** Окремий консольний Java-додаток, який буде з'єднуватися з сервером через мережу, отримувати список користувачів та логів і виводити їх на екран.
3. **Middleware:** У ролі middleware виступає протокол HTTP та формат JSON, що забезпечують передачу даних між різними частинами системи.

Діаграма класів Client-Server для FTP server:



Реалізація в коді:



ApiAdminController.java відкриває доступ до системи через HTTP:

```
14 // SERVER PART: REST API Controller
15 @RestController
16 @RequestMapping("/api")
17 public class ApiAdminController {
18
19     @Autowired
20     private UserRepository userRepository;
21
22     @Autowired
23     private ConnectionLogRepository logRepository;
24
25     // 1. Отримати список користувачів (JSON)
26     @GetMapping("/users")
27     public List<FtpUser> getAllUsers() {
28         return userRepository.findAll();
29     }
30
31     // 2. Додати користувача (JSON input)
32     @PostMapping("/users")
33     public ResponseEntity<String> addUser(@RequestBody FtpUser user) {
34         // Створення папки
35         File home = new File(user.getHomeDirectory());
36         if (!home.exists()) {
37             home.mkdirs();
38         }
39
40         userRepository.save(user);
41         return ResponseEntity.ok("User saved successfully with ID: " + user.getId());
42     }
43
44     @GetMapping("/logs")
45     public List<ConnectionLog> getLogs() {
46         return logRepository.findAll();
47     }
48 }
```

RemoteAdminClient.java це окрема консольна програма. Вона знаходиться в тому ж проєкті для зручності, але працює незалежно.

```
Run | Debug
15 public static void main(String[] args) {
16
17     try (Scanner scanner = new Scanner(System.in)) {
18
19         System.out.println(x: "=====");
20         System.out.println(x: "    REMOTE FTP ADMIN CLIENT (LAB 9)    ");
21         System.out.println(x: "=====");
22
23         while (true) {
24             System.out.println(x: "\nОберіть дію:");
25             System.out.println(x: "1. Отримати список користувачів (JSON)");
26             System.out.println(x: "2. Переглянути логи з'єднань");
27             System.out.println(x: "3. Створити демо-користувача ('remote_user')");
28             System.out.println(x: "0. Вихід");
29             System.out.print(s: "Ваш вибір > ");
30
31             if (!scanner.hasNextLine()) {
32                 break;
33             }
34
35             String choice = scanner.nextLine();
36
37             try {
38                 switch (choice) {
39                     case "1":
40                         getUsers();
41                         break;
42                     case "2":
43                         getLogs();
44                         break;
45                     case "3":
46                         createDemoUser();
47                         break;
48                     case "0":
49                         System.out.println(x: "Завершення роботи клієнта...");
50                         return;
51                     default:
52                         System.out.println(x: "Невірний вибір. Спробуйте ще раз.");
53                 }
54             } catch (Exception e) {
55
56
57
58
59
60
61
62 private static void getUsers() throws IOException, InterruptedException {
63     HttpRequest request = HttpRequest.newBuilder()
64         .uri(URI.create(SERVER_URL + "/users"))
65         .GET()
66         .build();
67
68     HttpResponse<String> response = client.send(request, HttpResponse.BodyHandlers.ofString());
69
70     System.out.println(x: "\n--- ВІДПОВІДЬ СЕРВЕРА (Користувачі) ---");
71     System.out.println(response.body());
72 }
73
74 private static void getLogs() throws IOException, InterruptedException {
75     HttpRequest request = HttpRequest.newBuilder()
76         .uri(URI.create(SERVER_URL + "/logs"))
77         .GET()
78         .build();
79
80     HttpResponse<String> response = client.send(request, HttpResponse.BodyHandlers.ofString());
81
82     System.out.println(x: "\n--- ВІДПОВІДЬ СЕРВЕРА (Логи) ---");
83     System.out.println(response.body());
84 }
85
86 private static void createDemoUser() throws IOException, InterruptedException {
87     String jsonBody = "{"
88         + "\"username\": \"remote_user\", \"
89         + "\"password\": \"12345\", \"
90         + "\"homeDirectory\": \"/C:/ftp/remote_user\", \"
91         + "\"canWrite\": true, \"
92         + "\"maxUploadRate\": 50000, \"
93         + "\"maxDownloadRate\": 0 \"
94         + \"}";
95
96     HttpRequest request = HttpRequest.newBuilder()
97         .uri(URI.create(SERVER_URL + "/users"))
98         .header(name: "Content-Type", value: "application/json")
99         .POST(HttpRequest.BodyPublishers.ofString(jsonBody))
100         .build();

```

Результати:

Список користувачів:

```
=====
REMOTE FTP ADMIN CLIENT (LAB 9)
=====

Оберіть дію:
1. Отримати список користувачів (JSON)
2. Переглянути логи з'єднань
3. Створити демо-користувача ('remote_user')
0. Вихід
Ваш вибір > 1

--- ВІДПОВІДЬ СЕРВЕРА (Користувачі) ---
[{"id":1,"username":"admin","password":"12345","homeDirectory":"C:/ftp/admin","canRead":true,"canWrite":true,"maxUploadRate":0,"maxDownloadRate":0,"maxIdleTime":300}, {"id":2,"username":"Leonid","password":"12345","homeDirectory":"C:/ftp/user8","canRead":true,"canWrite":true,"maxUploadRate":10000,"maxDownloadRate":3,"maxIdleTime":300}, {"id":3,"username":"ia-32fiot-23-010","password":"12345","homeDirectory":"C:/ftp/user8","canRead":true,"canWrite":true,"maxUploadRate":3,"maxDownloadRate":4,"maxIdleTime":300}, {"id":4,"username":"admin_test","password":"12345","homeDirectory":"C:/ftp/admin1","canRead":true,"canWrite":true,"maxUploadRate":6,"maxDownloadRate":5,"maxIdleTime":300}, {"id":7,"username":"adminn","password":"12345","homeDirectory":"C:/ftp/admin","canRead":true,"canWrite":true,"maxUploadRate":0,"maxDownloadRate":4,"maxIdleTime":300}, {"id":9,"username":"user1","password":"12345","homeDirectory":"C:/ftp/user3","canRead":true,"canWrite":true,"maxUploadRate":5000,"maxDownloadRate":0,"maxIdleTime":300}]
```

Логи з'єднань:

```
Оберіть дію:
1. Отримати список користувачів (JSON)
2. Переглянути логи з'єднань
3. Створити демо-користувача ('remote_user')
0. Вихід
Ваш вибір > 2

--- ВІДПОВІДЬ СЕРВЕРА (Логи) ---
[{"id":1,"username":"admin","clientId":"/[0:0:0:0:0:0:1]:55730","action":"LOGIN_SUCCESS","timestamp":"2025-12-02T22:52:46.883096"}, {"id":2,"username":"admin","clientId":"/[0:0:0:0:0:0:1]:55730","action":"DELETE: test.txt","timestamp":"2025-12-02T22:52:46.946199"}, {"id":3,"username":"admin","clientId":"/[0:0:0:0:0:0:1]:55731","action":"LOGIN_SUCCESS","timestamp":"2025-12-02T22:52:50.29079"}, {"id":4,"username":"admin","clientId":"/[0:0:0:0:0:0:1]:55736","action":"LOGIN_SUCCESS","timestamp":"2025-12-02T22:52:54.198473"}, {"id":5,"username":"admin","clientId":"/[0:0:0:0:0:0:1]:55736","action":"UPLOAD: test.txt","timestamp":"2025-12-02T22:52:54.224773"}, {"id":6,"username":"admin","clientId":"/[0:0:0:0:0:0:1]:63641","action":"LOGIN_SUCCESS","timestamp":"2025-12-02T23:06:44.051408"}, {"id":7,"username":"admin","clientId":"/[0:0:0:0:0:0:1]:63644","action":"LOGIN_SUCCESS","timestamp":"2025-12-02T23:06:50.778849"}, {"id":8,"username":"admin","clientId":"/[0:0:0:0:0:0:1]:63644","action":"UPLOAD: Знімок екрана 2025-12-01 085133.png","timestamp":"2025-12-02T23:06:50.802827"}, {"id":9,"username":"admin","clientId":"/[0:0:0:0:0:0:1]:63644","action":"UPLOAD: Знімок екрана 2025-12-01 085210.png","timestamp":"2025-12-02T23:06:54.330872"}]
```

Створення демо користувача:

```
Ваш вибір > 3

--- РЕЗУЛЬТАТ СТВОРЕННЯ ---
Status Code: 200
Body: User saved successfully with ID: 10
```

Вихід з клієнтської частини:

```
Оберіть дію:
1. Отримати список користувачів (JSON)
1. Отримати список користувачів (JSON)
2. Переглянути логи з'єднань
3. Створити демо-користувача ('remote_user')
0. Вихід
Ваш вибір > 0
Завершення роботи клієнта...
```

Висновки:

Я вивчив види взаємодії додатків (Client-Server, Peer-to-Peer, Serviceoriented Architecture) та реалізував в проєктованій системі архітектуру Client-Server.