**Лабораторна робота № 6**

**Тема:** Створення WEB-сервісу на базі технології WCF

**Студент:** Дигодюк Володимир Володимирович

**Група:** Комп'ютерні науки, 5 курс

Дата: 08.04.2024

**Конкретизована мета роботи**

Метою лабораторної роботи є навчитися створювати WEB-сервіси, що взаємодіють через мережу за допомогою технології Windows Communication Foundation (WCF), та навчитися робити висновки на основі проведеного дослідження.

**Короткий перелік досліджуваного матеріалу**

* Розробка WCF сервісу
* Використання бібліотеки класів для створення інтерфейсів
* Конфігурація WCF сервісу в **App.config**
* Тестування сервісу з використанням клієнтського застосунку

**Постановка задачі**

Варіант 3. Завданням лабораторної роботи було розробити WCF сервіс, який дозволяє користувачу отримувати список директорій на машині в залежності від переданого шляху.

**Порядок виконання роботи**

1. Встановлення та налаштування середовища розробки Visual Studio.
2. Створення проектів **Lab6ModernProgrammingTechnologies**, **Lab6Client** та **Lab6Interfaces**.
3. Визначення інтерфейсу **IDirectoryService** в проекті **Lab6Interfaces**.
4. Реалізація сервісу **DirectoryService** у проекті **Lab6ModernProgrammingTechnologies**.
5. Налаштування конфігурації WCF сервісу в **App.config**.
6. Створення клієнта WCF у **Lab6Client**, який використовує **ChannelFactory** для підключення до сервісу.
7. Тестування взаємодії між клієнтом та сервером.
8. Виконання запитів на отримання списку директорій та виведення результатів в консоль.
9. Додавання можливості взаємодії для користувача та логування дій сервера.

**Порядок виконання роботи, тесаме що вище, лиш з кращим**

1. **Встановлення та налаштування середовища розробки Visual Studio.**
   * Встановлено Visual Studio з необхідними пакетами для розробки .NET додатків.
   * Сконфігуровано середовище для підтримки розробки WCF сервісів.
2. **Створення проектів Lab6ModernProgrammingTechnologies, Lab6Client та Lab6Interfaces.**
   * Створено основний проект **Lab6ModernProgrammingTechnologies** для сервера.
   * Додано клієнтський проект **Lab6Client** за допомогою шаблону Console App.
   * Створено бібліотеку інтерфейсів **Lab6Interfaces** з використанням шаблону Class Library.
3. **Визначення інтерфейсу IDirectoryService в проекті Lab6Interfaces.**
   * Визначено інтерфейс **IDirectoryService** з методом **GetDirectories**, який приймає шлях до директорії як параметр і повертає список директорій.

csharpCopy code

[ServiceContract] public interface IDirectoryService { [OperationContract] List<string> GetDirectories(string path); }

1. **Реалізація сервісу DirectoryService у проекті Lab6ModernProgrammingTechnologies.**
   * Реалізовано клас **DirectoryService**, який імплементує **IDirectoryService**.
   * Використано **System.IO.Directory.GetDirectories(path)** для отримання списку директорій.

public class DirectoryService : IDirectoryService { public List<string> GetDirectories(string path) { // Логіка методу } }

1. **Налаштування конфігурації WCF сервісу в App.config.**
   * Налаштовано базову адресу та ендпоінти для **ServiceHost** у файлі **App.config**.
2. **Створення клієнта WCF у Lab6Client, який використовує ChannelFactory для підключення до сервісу.**
   * Ініціалізовано **ChannelFactory** і створено проксі для звернення до сервера.

ChannelFactory<IDirectoryService> channelFactory = new ChannelFactory<IDirectoryService>(binding, endpointAddress); IDirectoryService serviceClient = channelFactory.CreateChannel();

1. **Тестування взаємодії між клієнтом та сервером.**
   * Протестовано, що сервер приймає запити від клієнта та повертає відповідні дані.
   * При виникненні проблеми з доступом до URL (AddressAccessDeniedException) використано **netsh http add urlacl url=http://+:8000/ user=Everyone** для надання дозволів.
2. **Виконання запитів на отримання списку директорій та виведення результатів в консоль.**
   * Клієнтський додаток отримує шлях до директорії від користувача і відправляє запит серверу.

Console.WriteLine("Enter the path:"); string path = Console.ReadLine(); var directories = serviceClient.GetDirectories(path); directories.ForEach(Console.WriteLine);

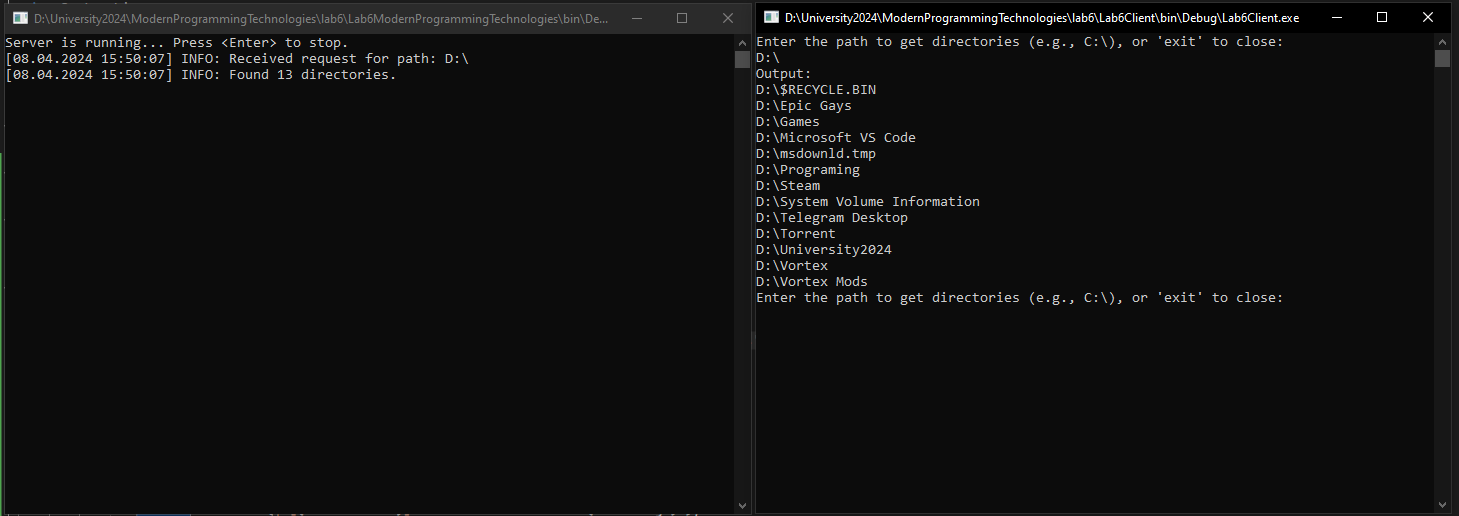
1. **Додавання можливості взаємодії для користувача та логування дій сервера.**
   * Клієнтський додаток тепер дозволяє користувачу вводити декілька шляхів поспіль, покращуючи взаємодію.
   * Сервер логує запити та помилки, надаючи деталізовану інформацію для налагодження.

Console.WriteLine($"[{DateTime.Now}] ERROR: Access denied.");

Console.WriteLine($"[{DateTime.Now}] INFO: Found {directories.Count} directories.");

…

Фінальні результати роботи, а саме скріни виконання додаю нижче:



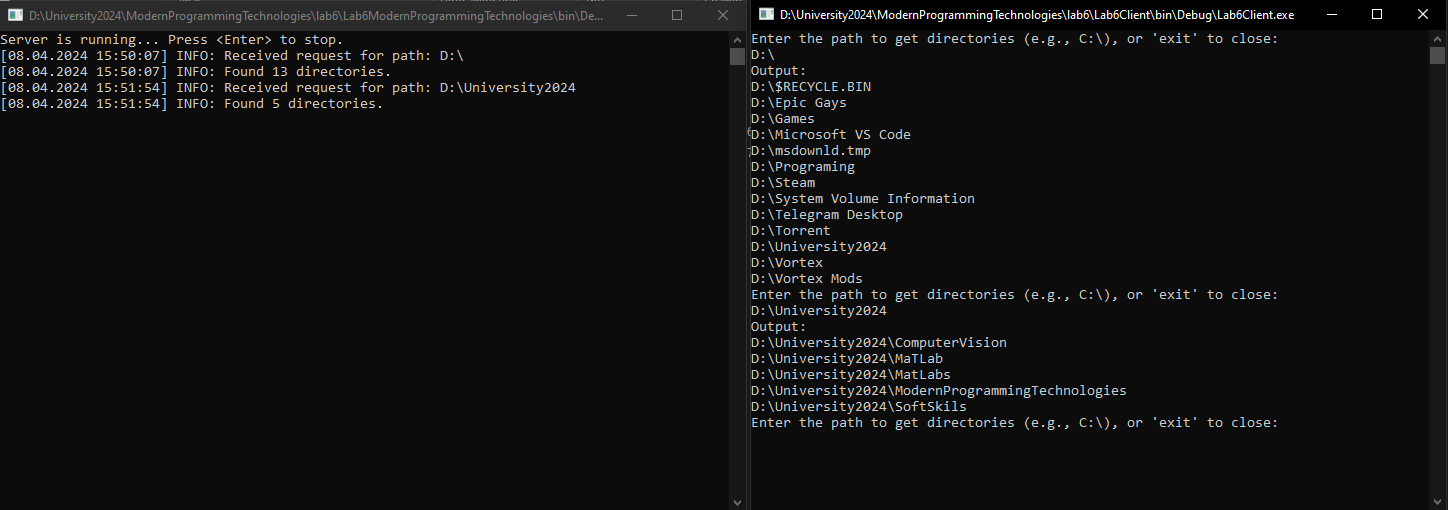


Рисунок 1: Вивід каталогів, що розташовані по певному шляху і повторний вивід по інакшому шляху

**Отримані результати**

* Було створено три проекти, які налаштовані для взаємодії за допомогою WCF.
* Сервіс успішно обробляє запити від клієнта та повертає відповідний список директорій.
* Інтерфейс користувача дозволяє вводити шляхи для отримання списку директорій.
* Сервер виконує логування дій та помилок, забезпечуючи важливу інформацію для моніторингу та виправлення помилок.

**Висновки**

У ході лабораторної роботи було розроблено практичні навички роботи з WCF та розуміння принципів побудови сервіс-орієнтованих застосунків. Було отримано досвід в налаштуванні взаємодії між клієнтом і сервером, реалізації інтерфейсу користувача та логуванні серверних процесів. Під час виконання роботи виникли певні труднощі, зокрема, у конфігурації WCF сервісу та реєстрації URL, але ці проблеми були успішно вирішені.

Контрольні питання

1. **Що таке WEB-сервіс?**

WEB-сервіс — це програмний сервіс, що розроблений для підтримки взаємодії між різними машинами через мережу. Зазвичай, він надає набір веб-орієнтованих API, які можуть бути викликані через HTTP. WEB-сервіси дозволяють різним застосункам взаємодіяти між собою безпосередньо, не залежно від платформ та мов програмування.

1. **Що таке сервіс орієнтована архітектура? Назвіть основні принципи сервіс орієнтованої архітектури.**

Сервіс орієнтована архітектура (SOA) — це стиль архітектури програмного забезпечення, де функціональність розподілена на окремі сервіси, кожен з яких може бути викликаний через стандартизовані інтерфейси, зазвичай через мережу. Основні принципи SOA включають:

* + Взаємодія через мережу: Сервіси взаємодіють через стандартизовані протоколи.
  + Слабке зв'язування: Сервіси залежать один від одного якомога менше.
  + Автономність: Кожен сервіс є автономним і відповідає за свою власну функціональність.
  + Відкритість інтерфейсів: Інтерфейси сервісів відкриті та доступні для використання іншими сервісами.
  + Складаність за допомогою композиції: Складні додатки можуть бути побудовані шляхом композиції менших сервісів.
  + Стандартизація: Взаємодія між сервісами базується на стандартах, що дозволяє легко інтегрувати сервіси, розроблені різними виробниками.

1. **Що таке SOAP?**

SOAP (Simple Object Access Protocol) — це протокол обміну структурованими повідомленнями, який використовується для взаємодії в розподілених обчислювальних системах. Він базується на XML та зазвичай передає дані через HTTP або інші протоколи мережевої передачі. SOAP використовується для обміну повідомленнями між сервісами, що дозволяє реалізовувати віддалений виклик процедур (RPC).

1. **Для чого використовується WCF?**

WCF (Windows Communication Foundation) — це фреймворк від Microsoft для побудови сервіс-орієнтованих застосунків. Він надає багатий набір функціональностей для підтримки комунікації між сервісами, включаючи засоби для роботи з повідомленнями, безпеку, транзакції та багато іншого. WCF дозволяє створювати сервіси, які можуть спілкуватися через різні протоколи і формати повідомлень, забезпечуючи гнучкість та інтероперабельність.

1. **Що таке ServiceContract, DataContract, OperationContract?**
   * **ServiceContract**: Вказує, що інтерфейс або клас визначає контракт сервісу у WCF. Це абстракція, яка визначає набір операцій, які сервіс буде надавати.
   * **DataContract**: Використовується для визначення типів, які передаються між клієнтом та сервісом. Дозволяє вказати, як дані будуть серіалізовані для передачі по мережі.
   * **OperationContract**: Вказує, що метод у сервісному контракті є операцією, яка доступна клієнтам через мережу. Він визначає, що ця частина інтерфейсу реалізує логіку бізнес-операцій, доступну для виклику ззовні.