

Lab - PaaS - Розгортання веб-сервісів WCF C Sharp за допомогою Microsoft Azure App Service

Мета: Отримати практичний досвід розробки веб-сервісів C# WCF та їх розгортання на хмарній PaaS-платформі Microsoft Azure App Service, а також створення та запуск клієнтів веб-сервісів.

Завдання:

1. Дослідити веб-сервіс C# WCF.
2. Протестувати веб-сервіс локально (WCF Test Client / SoapUI).
3. Дослідити клієнт C# для веб-сервісу.
4. Розгорнути веб-сервіс WCF в Azure App Service.
5. Налаштuvати клієнт на роботу з хмарним сервісом.

Лабораторне середовище:

- Visual Studio (workload "Azure development").
- Microsoft Azure Subscription (Free Tier / Student).

Крок 0: Завантаження проекту

Використовуйте той самий архів `repos.zip`, що був наданий до оригінальної лабораторної роботи. Розпакуйте його у зручну папку (наприклад, `C:\Users\ВашеІм'я\source\repos`).

Крок 1: Запуск та дослідження локально

Оскільки проект розділений на два окремих рішення (Server і Client), нам потрібно запустити їх паралельно у двох різних вікнах Visual Studio.

1.1. Запуск Сервера (WCF Service)

1. Зайдіть у папку `WcfServiceCalc`.
2. Відкрийте файл `WcfServiceCalc.sln`.
3. Якщо Visual Studio запропонує оновити версію .NET Framework — погодьтеся.
4. Натисніть **F5** (або кнопку "Start"), щоб запустити сервіс у режимі налагодження.

5. Відкриється браузер або **WCF Test Client**.
6. **ВАЖЛИВО:** Не закривайте це вікно Visual Studio і не зупиняйте налагодження. Сервер має працювати, щоб клієнт міг до нього підключитися.

localhost - /

28.07.2020	17:42	<dir> App_Data
24.11.2025	20:07	<dir> bin
01.09.2020	22:07	255 default.html
31.08.2020	18:49	744 IService1.cs
30.08.2020	23:19	<dir> obj
30.08.2020	23:19	<dir> Properties
28.07.2020	17:42	111 Service1.svc
28.07.2020	17:46	1002 Service1.svc.cs
01.09.2020	22:34	5315 WcfServiceCalc.csproj
01.09.2020	22:34	1457 WcfServiceCalc.csproj.user
31.08.2020	1:15	1141 WcfServiceCalc.sln
28.07.2020	17:42	1156 Web.config
28.07.2020	17:42	1300 Web.Debug.config
28.07.2020	17:42	1361 Web.Release.config

Mісце для скріншота: Запущений WCF Service (браузер або Test Client)

1.2. Запуск Клієнта (WCF Client)

1. Відкрийте **нове, друге вікно** Visual Studio (не закриваючи перше).
2. Через меню *File -> Open -> Project/Solution* знайдіть папку **WcfClientCalc**.
3. Відкрийте файл **WcfClientCalc.sln**.
4. Так само погодьтеся на оновлення .NET Framework, якщо запропонують.
5. Натисніть **F5** для запуску клієнта.

1.3. Перевірка зв'язку

1. Після запуску клієнта відкриється **консольне вікно**.
2. Введіть два числа за запитом програми (наприклад, 5 і 7).
3. Програма надішле запит на ваш локальний сервер (який ви запустили у пункті 1.1) і виведе результати операцій (Add , Subtract тощо).
4. Якщо ви бачите розраховані результати — локальне середовище налаштовано вірно.

```
C:\WINDOWS\system32\cmd.exe
Enter value 1: 5
Enter value 2: 7
Add(5,7) = 12
Subtract(5,7) = -2
Multiply(5,7) = 35
Divide(5,7) = 0,714285714285714
Press <ENTER> to terminate client.
```

Місце для скріншота: Консоль клієнта з результатами обчислень

Крок 2: Публікація веб-сервісу в Azure App Service

Ми будемо використовувати **Azure App Service (Windows)**, оскільки WCF нативно працює на Windows.

0. Налаштування Web.config для Azure (HTTPS). Оскільки Azure використовує захищене з'єднання (HTTPS), нам потрібно налаштувати WCF сервіс для роботи з ним.

1. У проекті `WcfServiceCalc` відкрийте файл `Web.config`.
2. Знайдіть секцію `<system.serviceModel>`.
3. Замініть **весь** **вміст** цієї секції на наступний код:

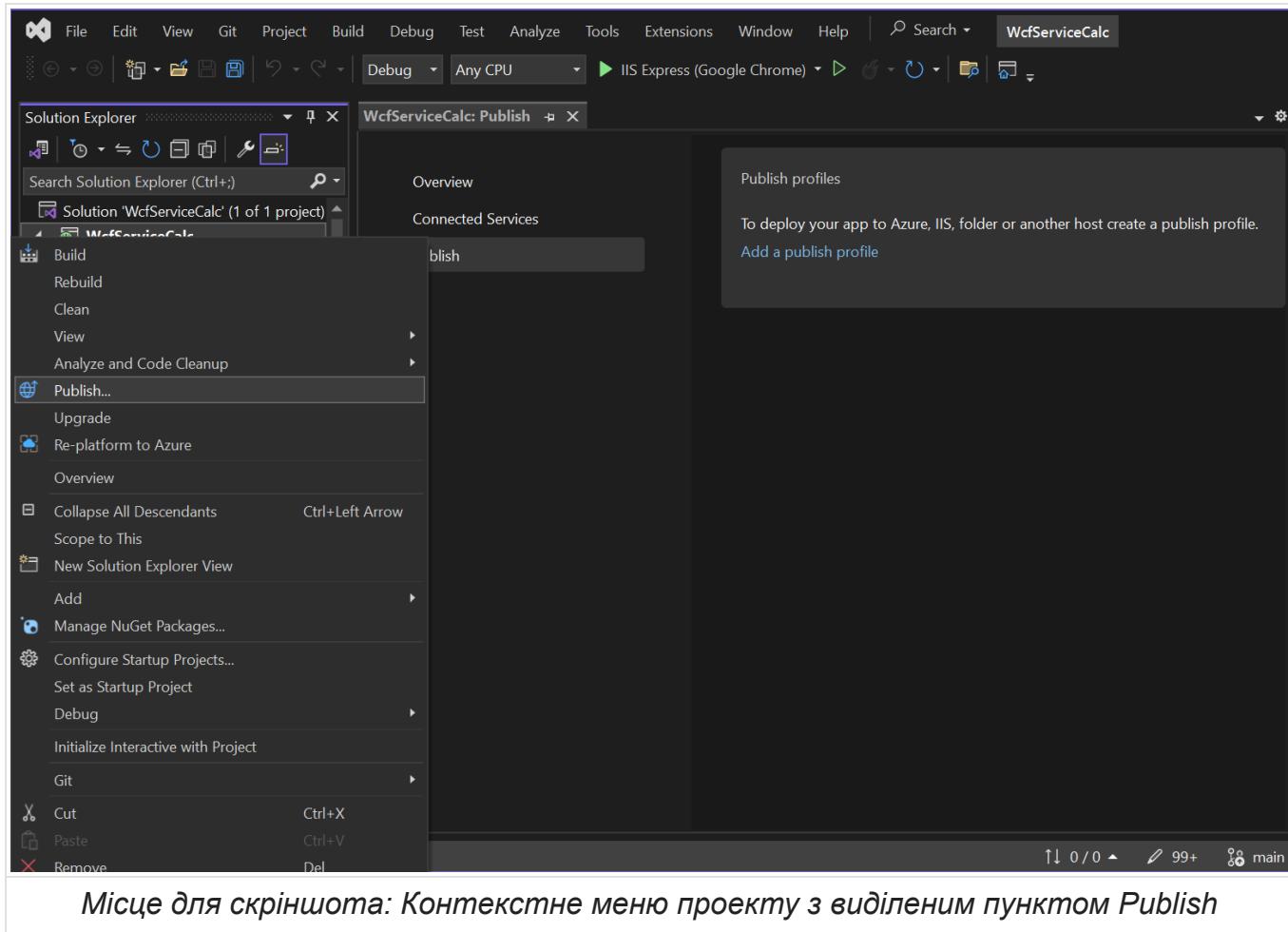
```
<system.serviceModel>
  <bindings>
    <basicHttpBinding>
      <binding name="SecureBinding">
        <security mode="Transport">
          <transport clientCredentialType="None" />
        </security>
      </binding>
    </basicHttpBinding>
  </bindings>
```

```

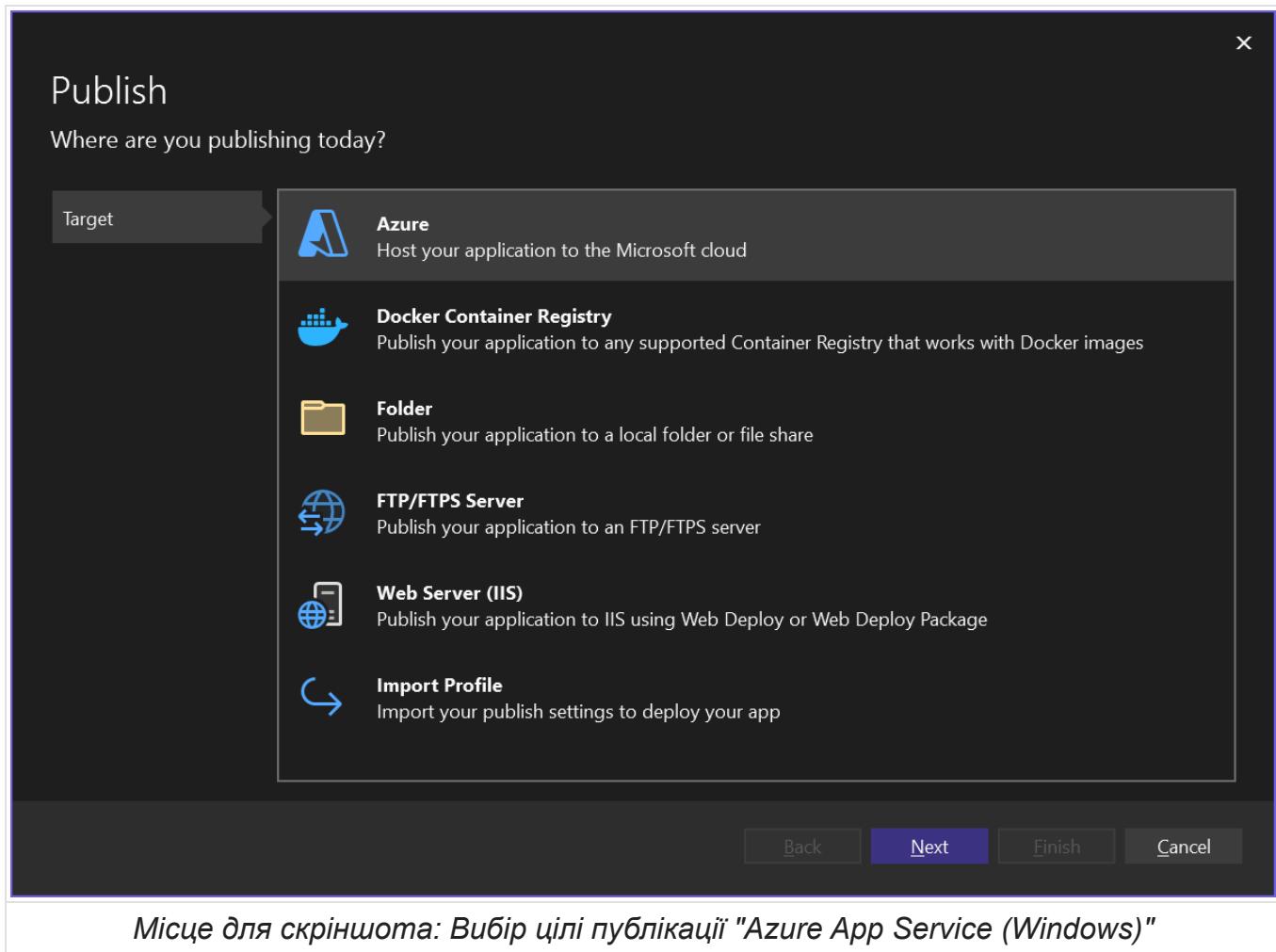
<protocolMapping>
    <add binding="basicHttpBinding" scheme="https"
bindingConfiguration="SecureBinding" />
</protocolMapping>
<behaviors>
    <serviceBehaviors>
        <behavior>
            <serviceMetadata httpGetEnabled="true" httpsGetEnabled="true"/>
            <serviceDebug includeExceptionDetailInFaults="false"/>
        </behavior>
    </serviceBehaviors>
</behaviors>
<serviceHostingEnvironment multipleSiteBindingsEnabled="true" />
</system.serviceModel>

```

- У Solution Explorer натисніть правою кнопкою миші на проект **WcfServiceCalc** (Сервер) і оберіть "**Publish...**" (Опублікувати).

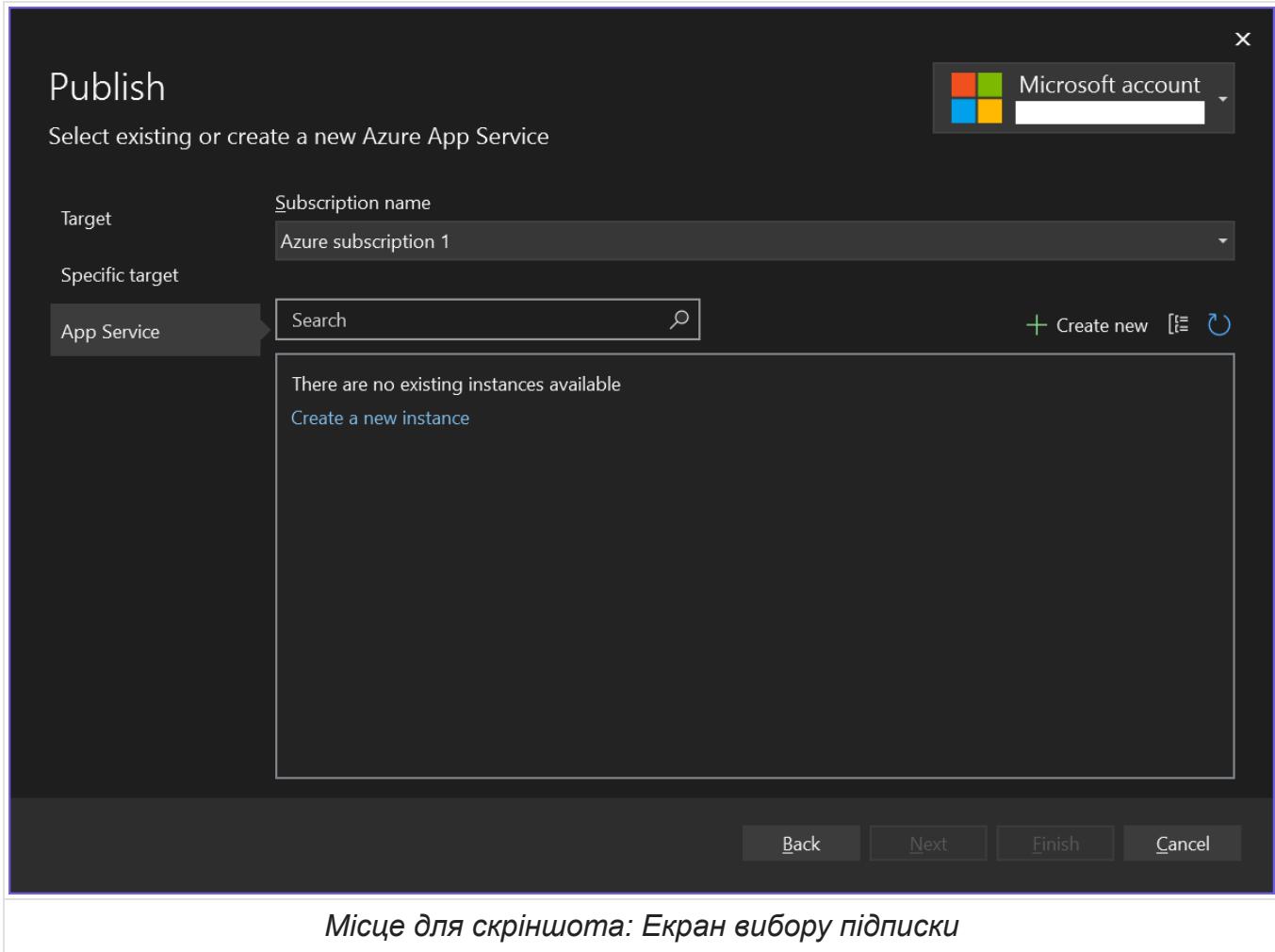


- У вікні майстра публікації оберіть **Azure** і натисніть "Next".
- Оберіть **Azure App Service (Windows)** і натисніть "Next".



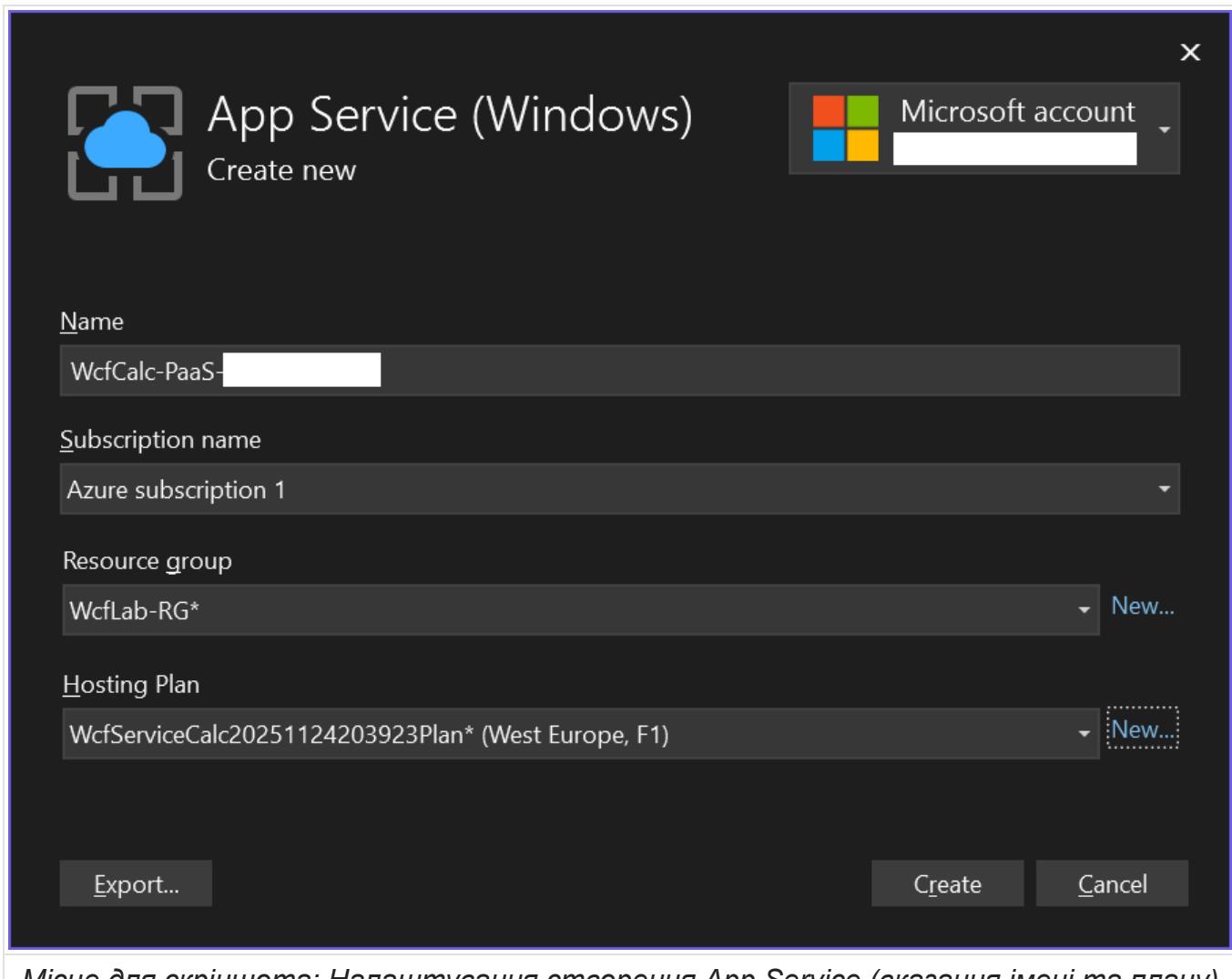
Місце для скріншота: Вибір цілі публікації "Azure App Service (Windows)"

4. У наступному вікні натисніть **Specific target -> App Service (Windows)**, **Next** та **Finish**, щоб створити новий ресурс.
5. Якщо у вас не було завантажене необхідне доповнення, вискочить **VS Installer** та запропонує довстановити його. Погодьтеся (та за необхідності повторіть кроки 1-3).
6. Увійдіть в акаунт **Microsoft**, у якому ви активували підписку.
* **Size:** Оберіть **Free (F1)** або **Shared (D1)**, щоб уникнути витрат.



Місце для скріншота: Екран вибору підписки

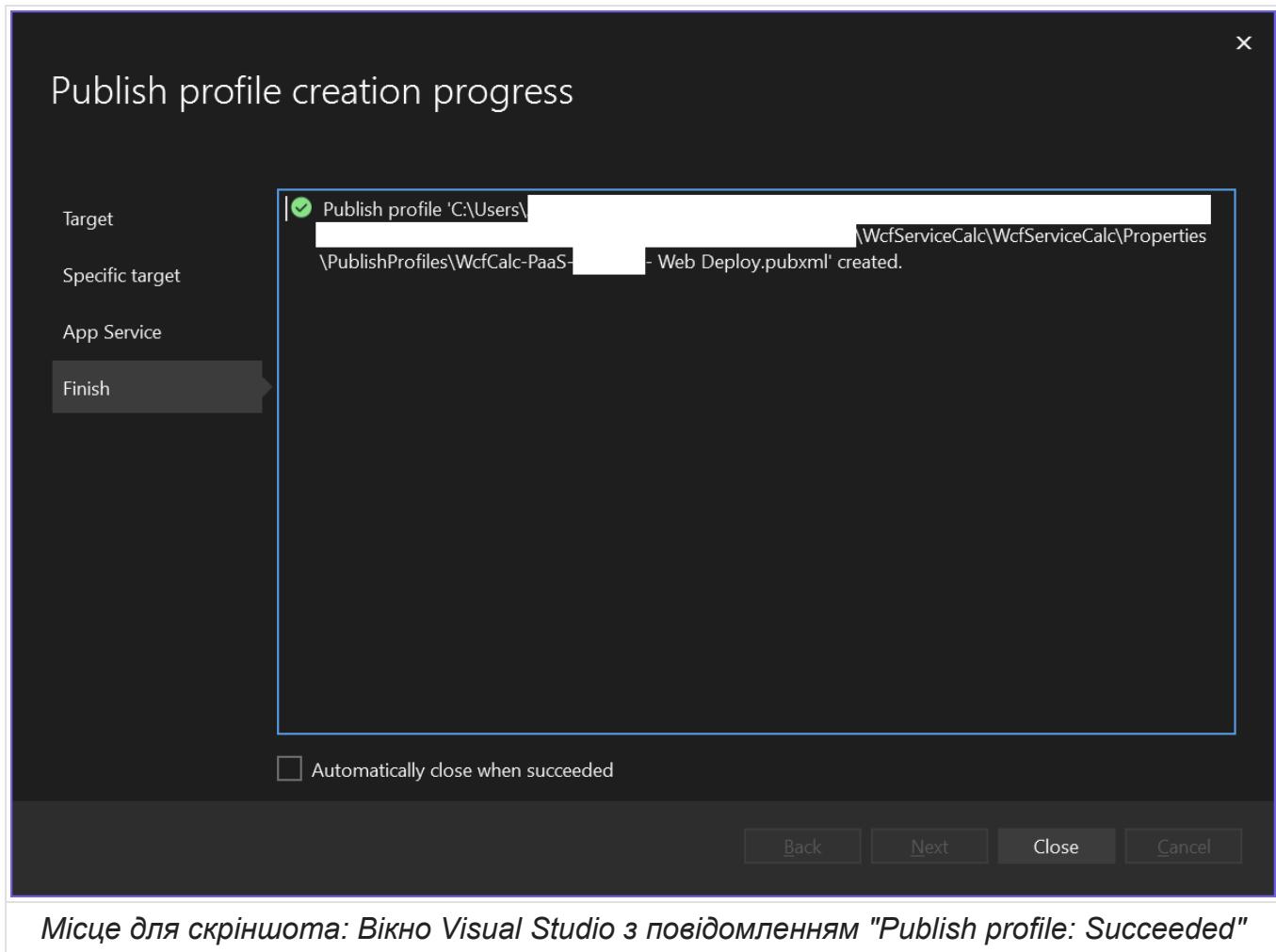
8. Натисніть кнопку **Create new +**.
9. Заповніть поля діалогового вікна "App Service (Windows)":
 - **Name:** Введіть унікальне ім'я (наприклад, `WcfCalc-PaaS-[ВашеПрізвище]`). Це ім'я стане частиною URL-адреси (`.azurewebsites.net`).
 - **Subscription:** Оберіть вашу активну підписку.
 - **Resource Group:** Натисніть "New" і створіть групу, наприклад, `WcfLab-RG` .
 - **Hosting Plan:** Створіть новий план.
 - **Location:** Оберіть найближчий регіон (наприклад, `West Europe`).
 - **Size:** Оберіть **Free (F1)** або **Shared (D1)**, щоб уникнути витрат.



Місце для скріншота: Налаштування створення App Service (вказання імені та плану)

9. Натисніть **Create**. Зачекайте, поки ресурси створяться в Azure.
10. Після створення натисніть **Finish**.

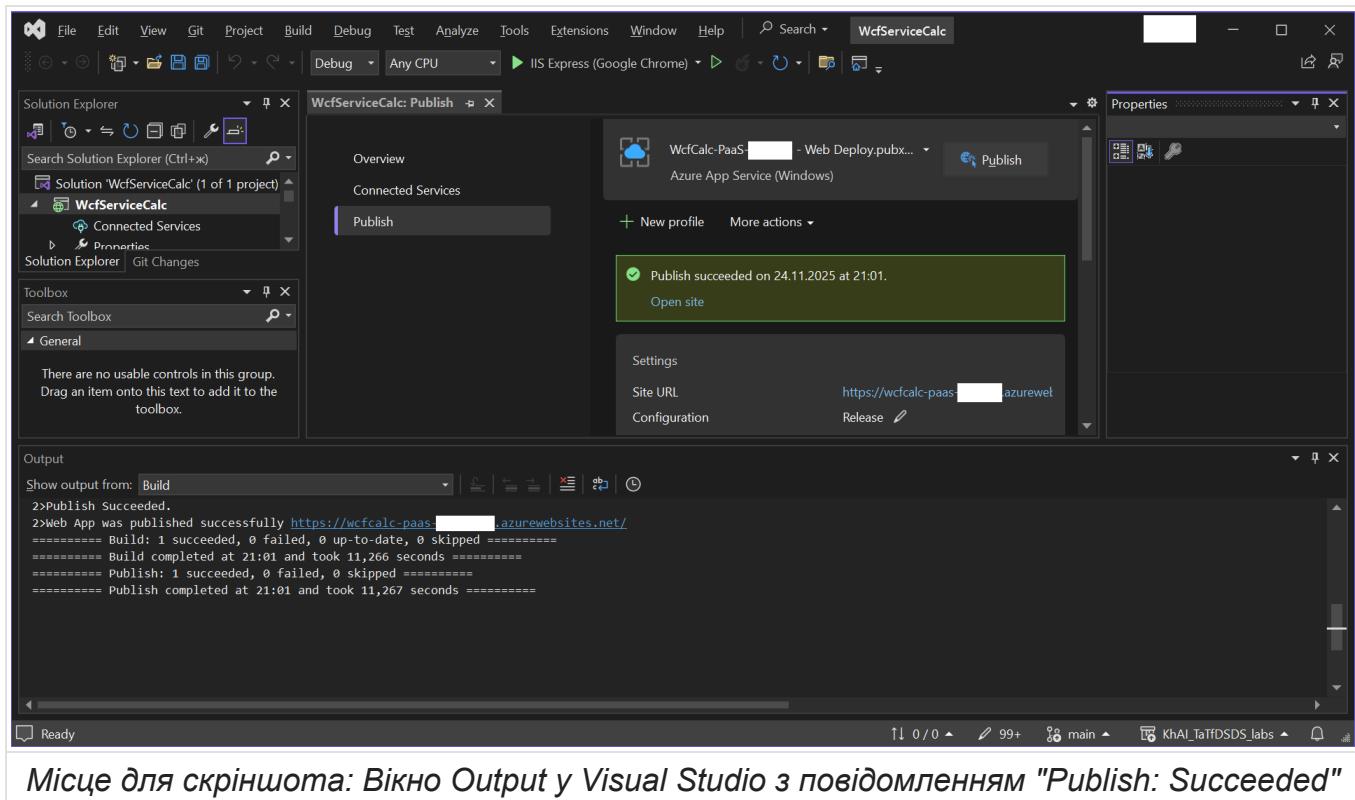
* *Примітка:* Якщо ви побачите помилку 403/404 або "Directory Listing Denied" – це нормально, бо за кореневою адресою `/` нічого немає. Головне, що сервіс працює за своєю адресою.



Місце для скріншота: Вікно Visual Studio з повідомленням "Publish profile: Succeeded"

11. Ви повернетесь на головний екран публікації. Натисніть кнопку **Publish** у верхньому правому куті.
12. Слідкуйте за вікном **Output**. Коли публікація завершиться, Visual Studio автоматично відкриє ваш браузер.

* *Примітка:* Якщо ви побачите помилку 403/404 або "Directory Listing Denied" – це нормально, бо за кореневою адресою `/` нічого немає. Головне, що сервіс працює за своєю адресою.



Місце для скріншота: Вікно Output у Visual Studio з повідомленням "Publish: Succeeded"

Крок 3: Перевірка розгорнутого сервісу

1. У браузері, що відкрився, допишіть до адреси: /Service1.svc .
 - Повна адреса має виглядати так: [https://wcfcalc-paas-\[ВашеПрізвище\].azurewebsites.net/Service1.svc](https://wcfcalc-paas-[ВашеПрізвище].azurewebsites.net/Service1.svc)
2. Ви повинні побачити службову сторінку "Service1 Service" з повідомленням "You have created a service".

Service1 Service

You have created a service.

To test this service, you will need to create a client and use it to call the service. You can do this using the svcutil.exe tool from the command line with the following syntax:

```
svcutil.exe http://wcfcalc-paas-siroklyn.azurewebsites.net/Service1.svc?wsdl
```

You can also access the service description as a single file:

```
http://wcfcalc-paas-siroklyn.azurewebsites.net/Service1.svc?singleWSDL
```

This will generate a configuration file and a code file that contains the client class. Add the two files to your client application and use the generated client class to call the Service. For example:

C#

```
class Test
{
    static void Main()
    {
        Service1Client client = new Service1Client();

        // Use the 'client' variable to call operations on the service.

        // Always close the client.
        client.Close();
    }
}
```



Visual Basic

Mісце для скріншота: Браузер із відкритою сторінкою .svc в Azure

3. Допишіть ?wsdl в кінці URL (<https://.../Service1.svc?wsdl>), щоб переконатися, що опис сервісу доступний. Скопіюйте це повне посилання на WSDL (для кроку 4). повідомленням "You have created a service".

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
    xmlns:soap="http://schemas.xmlsoap.org/soap/" xmlns:wsdl_soap12="http://schemas.xmlsoap.org/wsdl/soap12/" xmlns:tns="http://tempuri.org/"
    xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex"
    xmlns:wsaw="http://schemas.xmlsoap.org/ws/2004/08/addressing/policy" xmlns:wsaw1="http://www.w3.org/2006/05/addressing/wsdl"
    xmlns:msc="http://schemas.microsoft.com/ws/2005/12/wsdl/contract" xmlns:wsps="http://schemas.xmlsoap.org/ws/2004/09/policy"
    xmlns:wsa10="http://www.w3.org/2005/08/addressing" xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata" name="Service1"
    targetNamespace="http://tempuri.org/">
    <wsdl:types>
        <xsd:schema targetNamespace="http://tempuri.org/Imports">
            <xsd:import schemaLocation="https://wcfcalc-paas-XXXXXXXXXX.azurewebsites.net/Service1.svc?xsd=xsd0" namespace="http://tempuri.org/"/>
            <xsd:import schemaLocation="https://wcfcalc-paas-XXXXXXXXXX.azurewebsites.net/Service1.svc?xsd=xsd1"
                namespace="http://schemas.microsoft.com/2003/10/Serialization/"/>
        </xsd:schema>
    </wsdl:types>
    <wsdl:message name="IService1_Add_InputMessage">
        <wsdl:part name="parameters" element="tns:Add"/>
    </wsdl:message>
    <wsdl:message name="IService1_Add_OutputMessage">
        <wsdl:part name="parameters" element="tns:AddResponse"/>
    </wsdl:message>
    <wsdl:message name="IService1_Subtract_InputMessage">
        <wsdl:part name="parameters" element="tns:Subtract"/>
    </wsdl:message>
    <wsdl:message name="IService1_Subtract_OutputMessage">
        <wsdl:part name="parameters" element="tns:SubtractResponse"/>
    </wsdl:message>
    <wsdl:message name="IService1_Multiply_InputMessage">
        <wsdl:part name="parameters" element="tns:Multiply"/>
    </wsdl:message>
    <wsdl:message name="IService1_Multiply_OutputMessage">
        <wsdl:part name="parameters" element="tns:MulitplyResponse"/>
    </wsdl:message>

```

Mісце для скріншота: Браузер із відкритою сторінкою .svc?wsdl в Azure

4. Якщо у вас, замість .xml файлу, видає той самий результат, що в пункті 3.2, виконайте наступні кроки:

Azure працює через захищений протокол HTTPS, але за замовчуванням старі проекти WCF дозволяють віддавати метадані тільки через HTTP. Щоб виправити це:

1. Поверніться у **Visual Studio**.

2. У **Solution Explorer** розгорніть проект **WcfServiceCalc** і відкрийте файл **Web.config**.

3. Знайдіть секцію **<serviceBehaviors>** -> **<behavior>**.

4. Знайдіть рядок:

```
<serviceMetadata httpGetEnabled="true"/>
```

5. Змініть його, додавши дозвіл для HTTPS (**httpsGetEnabled="true"**):

```
<serviceMetadata httpGetEnabled="true" httpsGetEnabled="true"/>
```

6. Збережіть файл (**Ctrl+S**).

7. Знову опублікуйте проект в Azure: натисніть правою кнопкою миші на проект **WcfServiceCalc** -> **Publish...** -> кнопка **Publish**.

8. Після успішної публікації спробуйте знову відкрити посилання з **?wsdl** (наприклад, <https://.../Service1.svc?wsdl>). Тепер ви маєте побачити XML-код.

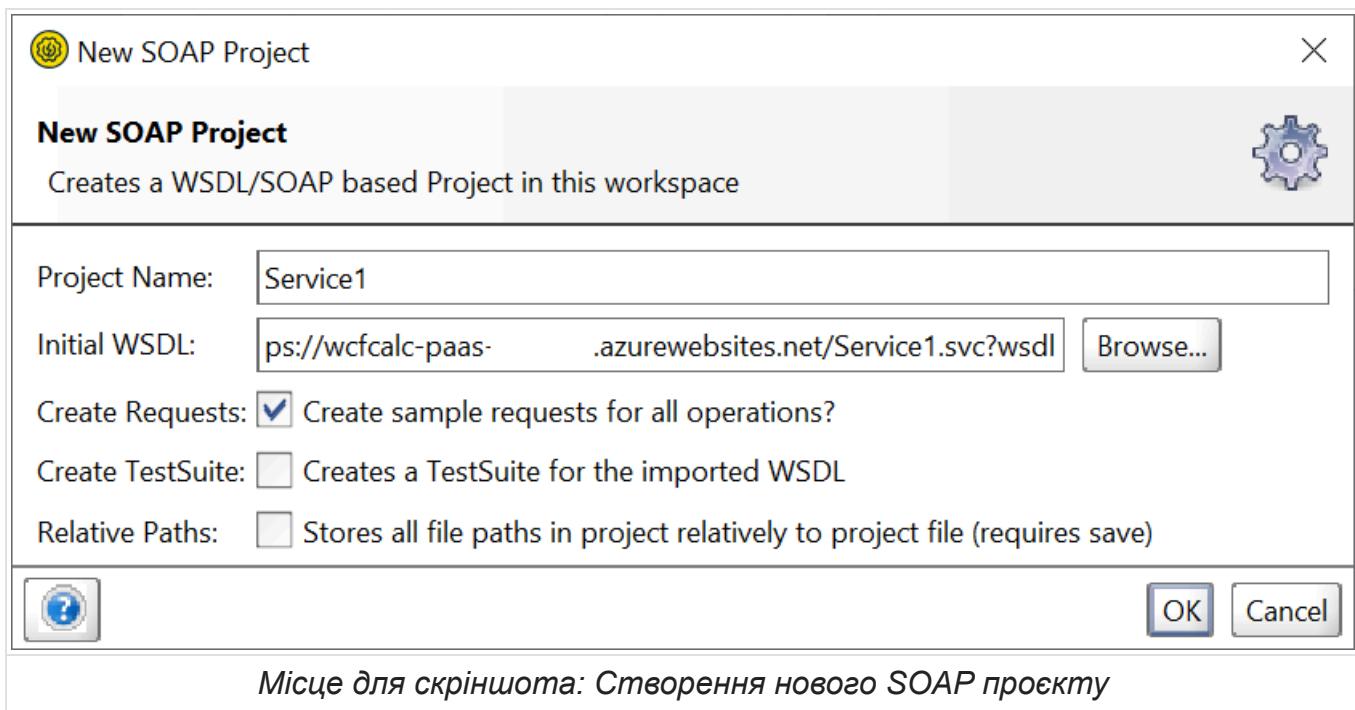
Крок 4: Тестування за допомогою SoapUI (Опціонально)

1. Відкрийте **SoapUI**.

2. Натисніть "SOAP" (або File -> New SOAP Project).

3. У полі **Initial WSDL** вставте URL-адресу вашого WSDL з Azure (з Кроку 3).

4. Натисніть OK. SoapUI завантажить визначення сервісу.
результатом додавання.



5. Розгорніть операцію Add , відкрийте Request 1 , введіть значення для n1 та n2 і натисніть зелену стрілку (Play).
6. Переконайтесь, що у правій частині вікна прийшла відповідь з результатом додавання.

The screenshot shows the SoapUI interface with the following details:

- Toolbar:** File, Project, Suite, Case, Step, Tools, Desktop, Help.
- Navigator:** Projects, Service1, BasicHttpBinding, Add, Divide, Multiply, Subtract, BasicHttpBinding, Add, Divide, Multiply, Subtract.
- Request 1:** URL: https://wcfcalc-paas-siroklyn.azurewebsites.net/Service1.svc. The request XML is shown in the left pane, and the response XML is shown in the right pane.
- Properties Panel:** Request Properties table with columns Property and Value.
- Bottom Status Bar:** response time: 66ms (174 bytes), SoapUI log, http log, jetty log, error log, wsrm log, memory log.

Mісце для скріншота: Успішний запит у SoapUI до хмарного сервісу

Крок 5: Налаштування C# Клієнта

Тепер налаштуємо консольну програму WcfClientCalc так, щоб вона зверталася до вашого сервісу в хмарі, а не локально.

1. У Solution Explorer розгорніть проект WcfClientCalc .
2. Відкрийте файл app.config .
3. Знайдіть секцію <system.serviceModel> .
4. Замініть її на цей код, не забудьте вставити вашу URL-адресу:

Було:

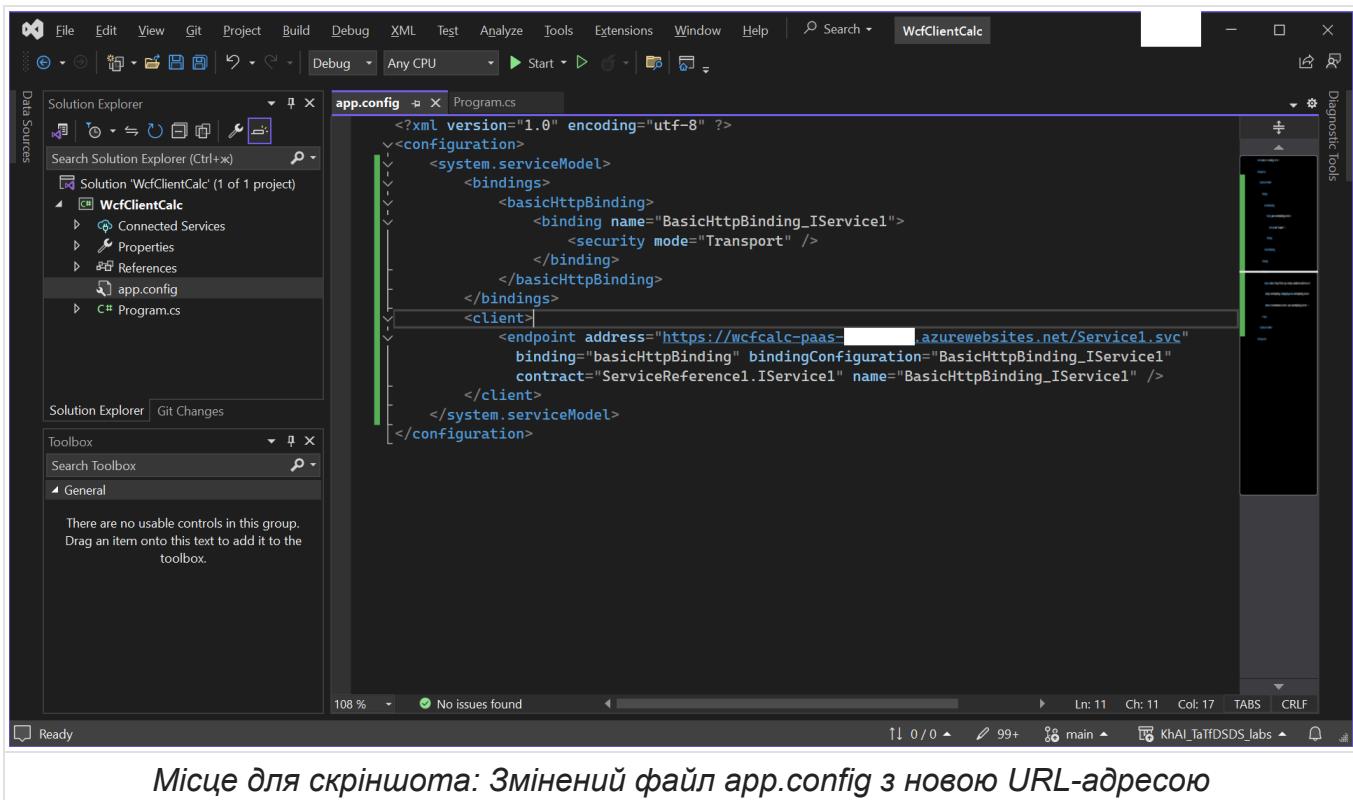
```
<system.serviceModel>
  <bindings>
    <basicHttpBinding>
      <binding name="BasicHttpBinding_IService1" />
    </basicHttpBinding>
```

```
</bindings>
<client>
    <endpoint address="http://localhost:52788/Service1.svc"
binding="basicHttpBinding"
        bindingConfiguration="BasicHttpBinding_IService1"
contract="ServiceReference1.IService1"
        name="BasicHttpBinding_IService1" />
</client>
</system.serviceModel>
```

Має стати (приклад):

```
<system.serviceModel>
    <bindings>
        <basicHttpBinding>
            <binding name="BasicHttpBinding_IService1">
                <security mode="Transport" />
            </binding>
        </basicHttpBinding>
    </bindings>
    <client>
        <endpoint address="https://wcfcalc-paas-
[ВашеПрізвище].azurewebsites.net/Service1.svc" binding="basicHttpBinding"
bindingConfiguration="BasicHttpBinding_IService1"
contract="ServiceReference1.IService1" name="BasicHttpBinding_IService1" />
    </client>
</system.serviceModel>
```

5. Збережіть файл (Ctrl+S).



Місце для скріншота: Змінений файл app.config з новою URL-адресою

Старі версії .NET не підтримують TLS 1.2 за замовчуванням, що викликає помилку з'єднання з Azure.

6. Відкрийте файл `Program.cs` у проекті `WcfClientCalc`.
7. Знайдіть метод `static void Main(string[] args)`.
8. Додайте цей рядок на самий початок методу:

```
// ПРИМУСОВЕ ВКЛЮЧЕННЯ TLS 1.2 для сумісності з Azure
System.Net.ServicePointManager.SecurityProtocol =
(System.Net.SecurityProtocolType)3072;
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using WcfClientCalc.ServiceReference1;
6
7  namespace WcfClientCalc
8  {
9      class Program
10     {
11         static void Main(string[] args)
12         {
13             System.Net.ServicePointManager.SecurityProtocol = (System.Net.SecurityProtocolType)30;
14             ServiceClient client = new ServiceClient();
15             // Enter arguments from console.
16             Console.WriteLine("Enter value 1: ");
17             double value1 = Convert.ToDouble(Console.ReadLine());
18             Console.WriteLine("Enter value 2: ");
19             double value2 = Convert.ToDouble(Console.ReadLine());
20             // Call the Add service operation.
21             double result = client.Add(value1, value2);
22             Console.WriteLine("\nAdd({0},{1}) = {2}", value1, value2, result);
23             // Call the Subtract service operation.
24             result = client.Subtract(value1, value2);
25             Console.WriteLine("Subtract({0},{1}) = {2}", value1, value2, result);
26             // Call the Multiply service operation.
27             result = client.Multiply(value1, value2);
28             Console.WriteLine("Multiply({0},{1}) = {2}", value1, value2, result);
29             // Call the Divide service operation.
30             result = client.Divide(value1, value2);
31             Console.WriteLine("Divide({0},{1}) = {2}", value1, value2, result);
32         }
33     }
34 }
```

Місце для скріншота: Змінений файл Program.cs з новим рядком

9. Збережіть файл (**Ctrl+S**).
10. Натисніть **F5** для запуску клієнта.
11. Введіть числа у консолі. Якщо програма видає коректний результат, це означає, що ваш локальний клієнт успішно з'єднався з віддаленим сервером в Azure.

```
C:\WINDOWS\system32\cmd.exe
Enter value 1: 5
Enter value 2: 7

Add(5,7) = 12
Subtract(5,7) = -2
Multiply(5,7) = 35
Divide(5,7) = 0,714285714285714

Press <ENTER> to terminate client.
```

Місце для скріншота: Робота консольного клієнта, що отримує дані з Azure

Крок 6: Очищення ресурсів

ВАЖЛИВО: Після завершення роботи видаліть створені ресурси, щоб уникнути списання коштів з вашого рахунку Azure.

1. Зайдіть на портал portal.azure.com.
2. Перейдіть у розділ "**Resource groups**" (Групи ресурсів).
3. Знайдіть групу WcfLab-RG (яку ви створили на кроці 2).
4. Натисніть "**Delete resource group**".
5. Введіть назву групи для підтвердження та натисніть "**Delete**". Це видалить App Service та App Service Plan.

The screenshot shows the Microsoft Azure portal interface. On the left, there's a navigation sidebar with various options like Activity log, Access control (IAM), Tags, Resource visualizer, Events, Settings, Cost Management, Monitoring, Automation, and Help. The main area is titled "Delete a resource group" and displays the following information:

- Resource group to be deleted:** WcfLab-RG
- 2 resources being deleted across 1 subscriptions, 1 resource groups:**

Name	Resource type	Subscription	Resource Group
WcfCalc-PaaS	App Service	Azure subscription 1	WcfLab-RG
WcfServiceCalc20251124203923Plan	App Service plan	Azure subscription 1	WcfLab-RG
- Enter resource group name to confirm deletion ***: The input field contains "WcfLab-RG".
- Buttons:** Delete (highlighted in red) and Cancel.

At the bottom of the screenshot, there is a caption in Russian: "Місце для скріншота: Підтвердження видалення групи ресурсів" (Place for screenshot: Confirmation of resource group deletion).