PaaS – Exploration and Deployment of WCF C# web services with AWS Elastic Beanstalk

Objective: Get practical experience in C# WCF Web services development and deployment on Cloud PaaS with AWS Elastic Beanstalk as well as in creating and running web service clients.

Tasks:

1. Explore a C# WCF Web Service
2. Test a Web Service using SoapUI tool
3. Explore a WCF Web Service C# client
4. Deploy a C# WCF Web Service on AWS Elastic Beanstalk PaaS service

Lab environment:

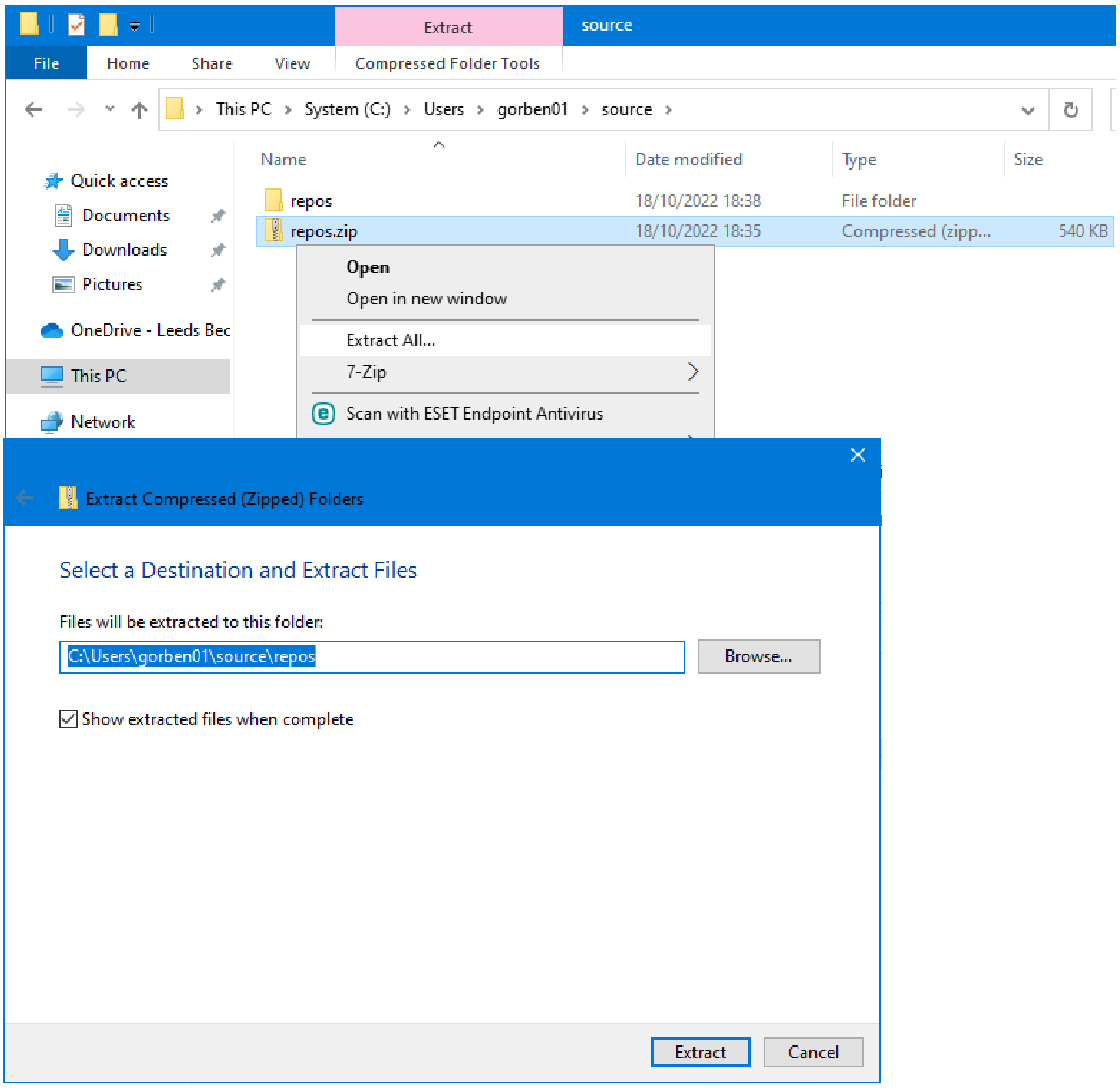
- Visual Studio IDE with the following workloads:
    - ASP.NET and web development
    - Azure development
    - AWS Toolkit for Visual Studio (https://aws.amazon.com/visualstudio/)
- SoapUI tool

Contents

Step 0: Downloading the Project

From the VLE download the *repos.zip* file and save it to *C:\Users\c1234567\source\* folder on the University (bottom) PC, where c1234567 – is your user id.

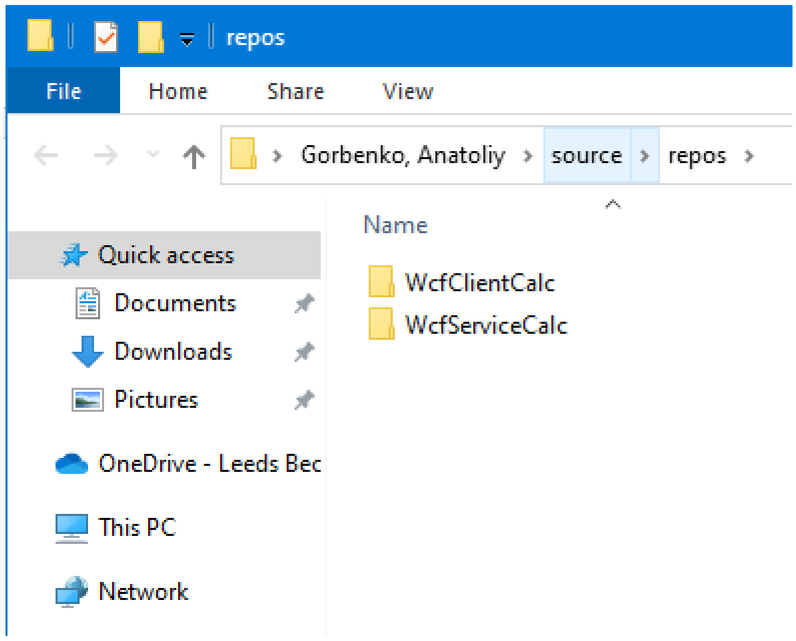
Extract All...



Files will be extracted to this folder:
C:\Users\gorben01\source\repos

☑ Show extracted files when complete

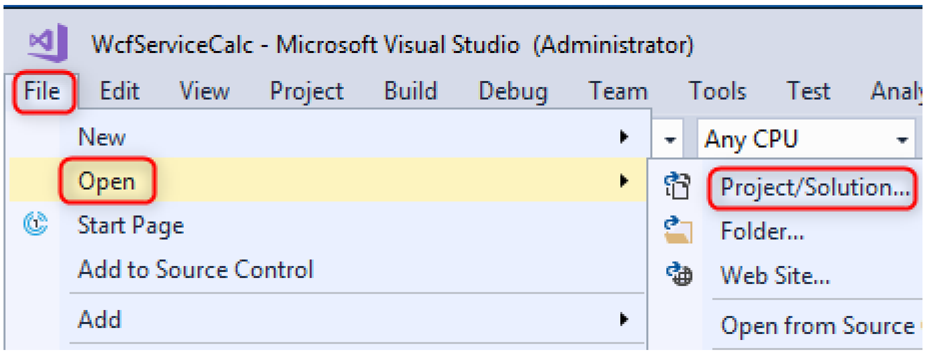
Make sure you have *WcfClientCalc* and *WcfServiceCalc* folders exactly in *C:\Users\c1234567\source\repos* location:

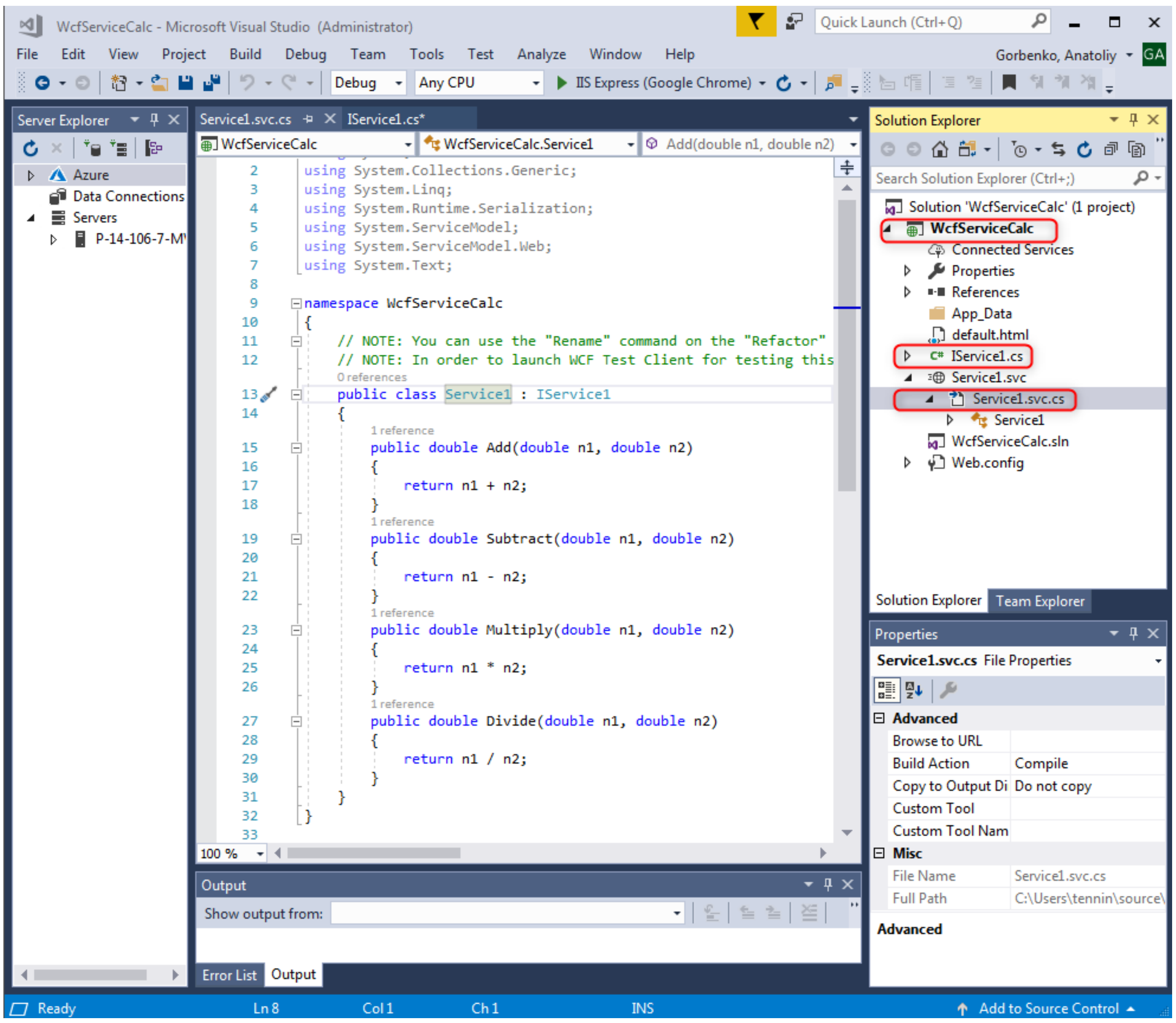


Step 1: Running Visual Studio

Run the Visual Studio on the University's (bottom) PC.

Open the **WcfServiceCalc** project in the solution Explorer panel (open *C:\Users\c1234567\source\repos*) and open two .cs source files: *Service1.svc.cs* and *IService1.cs*.





```
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;
using System.Text;

namespace WcfServiceCalc
{
    // NOTE: You can use the "Rename" command on the "Refactor"
    // NOTE: In order to launch WCF Test Client for testing this
    0 references
    public class Service1 : IService1
    {
        1 reference
        public double Add(double n1, double n2)
        {
            return n1 + n2;
        }
        1 reference
        public double Subtract(double n1, double n2)
        {
            return n1 - n2;
        }
        1 reference
        public double Multiply(double n1, double n2)
        {
            return n1 * n2;
        }
        1 reference
        public double Divide(double n1, double n2)
        {
            return n1 / n2;
        }
    }
}
```

Properties

Service1.svc.cs File Properties

Advanced	
Browse to URL	
Build Action	Compile
Copy to Output Di	Do not copy
Custom Tool	
Custom Tool Nam	
Misc	
File Name	Service1.svc.cs
Full Path	C:\Users\tennin\source\

Advanced

Step 2: Exploring C# WCF Web Service

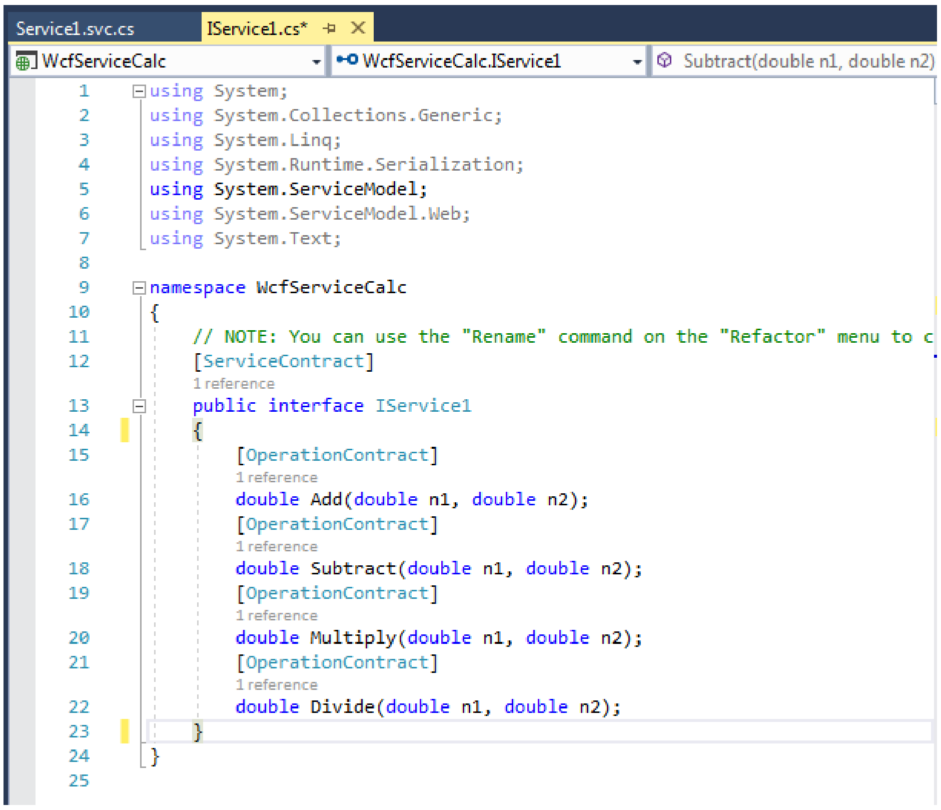
Windows Communication Foundation (WCF) is a framework for building service-oriented applications. Using WCF, you can send data as asynchronous messages from one service endpoint to another. A service endpoint can be part of a continuously available service hosted by IIS (Internet Information Server), or it can be a service hosted in an application. An endpoint can be a client of a service that requests data from a service endpoint. The messages can be as simple as a single character or word sent as XML, or as complex as a stream of binary data.

Exploring the Service1.svc.cs

The C# class Service1 implements IService1 interface. It implements four methods: Add, Subtract, Multiply and Divide. Each method accepts two operands and returns the result of a corresponding arithmetic operation.

Exploring the IService1.cs

IService.cs describes a Web Service interface by specifying service's operations and their input and output parameters.



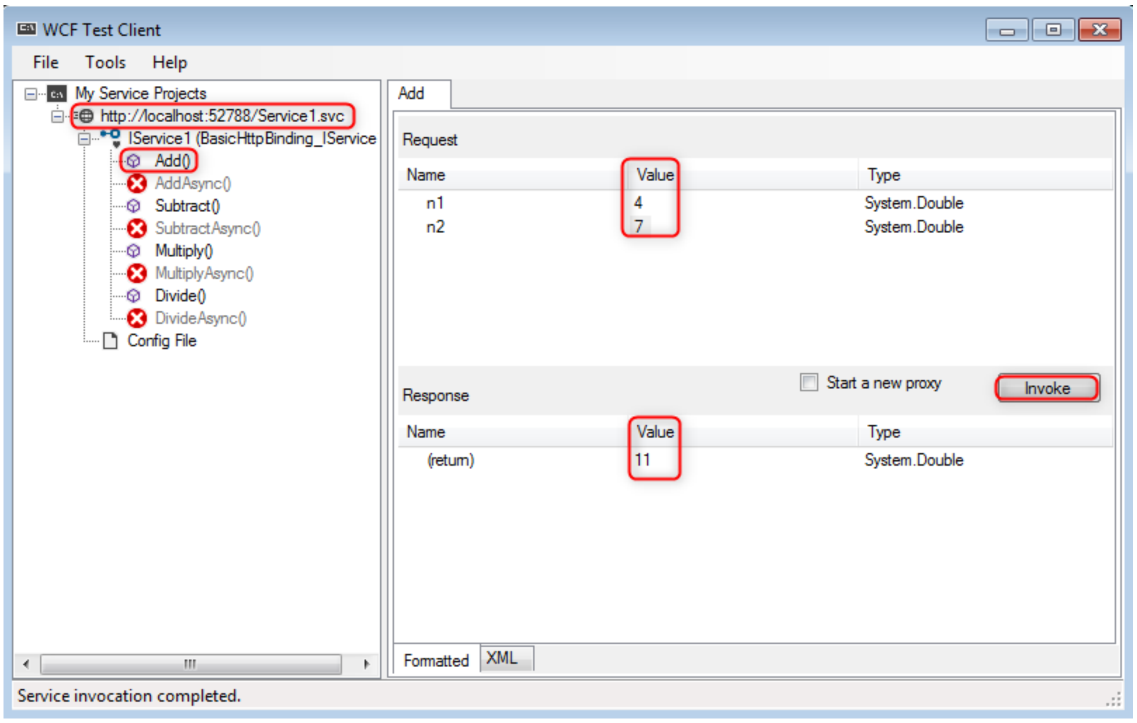
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;
using System.Text;

namespace WcfServiceCalc
{
    // NOTE: You can use the "Rename" command on the "Refactor" menu to c
    [ServiceContract]
    1 reference
    public interface IService1
    {
        [OperationContract]
        1 reference
        double Add(double n1, double n2);
        [OperationContract]
        1 reference
        double Subtract(double n1, double n2);
        [OperationContract]
        1 reference
        double Multiply(double n1, double n2);
        [OperationContract]
        1 reference
        double Divide(double n1, double n2);
    }
}
```

Launching and testing the WCF Web Service

a. In order to launch WCF Web Service for testing this service, please select *Service1.svc.cs* at the Solution Explorer and start debugging.
Visual Studio includes IIS Express, a lightweight, self-contained version of IIS optimized for developers. It makes it easy to develop and test web project.

b. After Visual the web service starts, the *WCF Test Client* launches automatically. It will detect Web Service interface description (WSDL) and generate stubs for all web service operations. You can change values of the input parameters, invoke the web service and check the result returned in service's response.

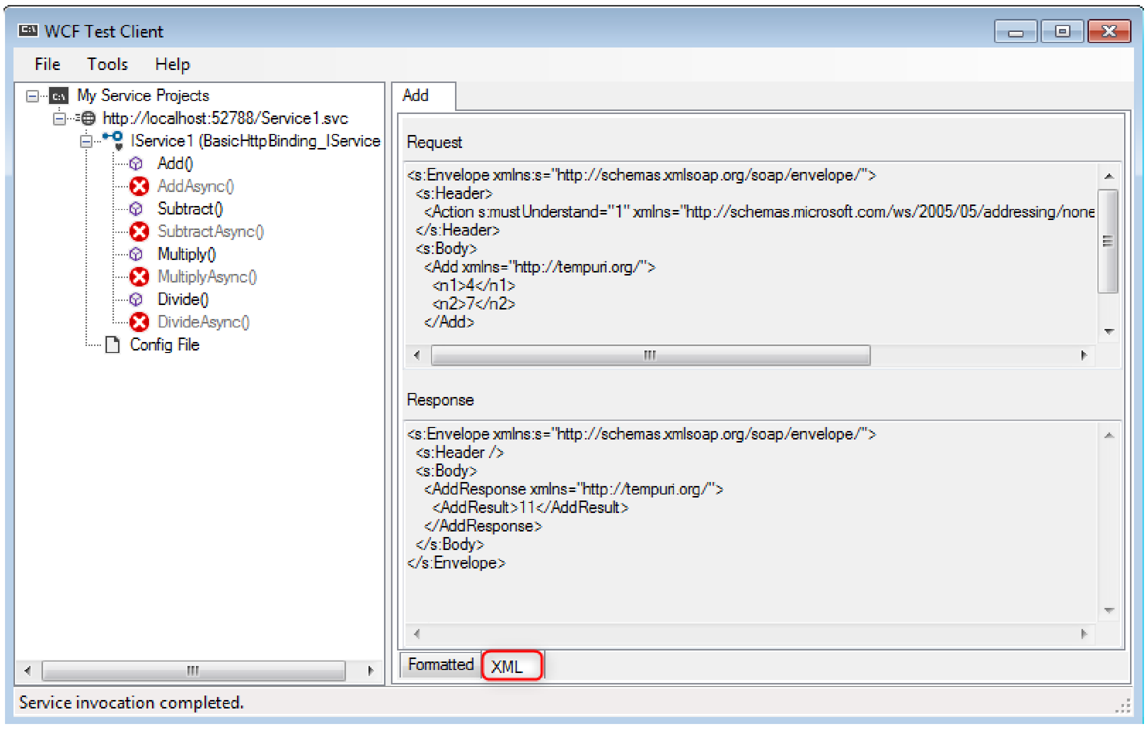
WCF Test Client (WcfTestClient.exe) is a GUI tool (it is a part of Visual Studio) that allows you to input parameters of arbitrary types, submit that input to the service, and view the response the service sends back. It provides a seamless service testing experience when combined with WCF Service Auto Host.



Name	Value	Type
n1	4	System.Double
n2	7	System.Double

Name	Value	Type
(return)	11	System.Double

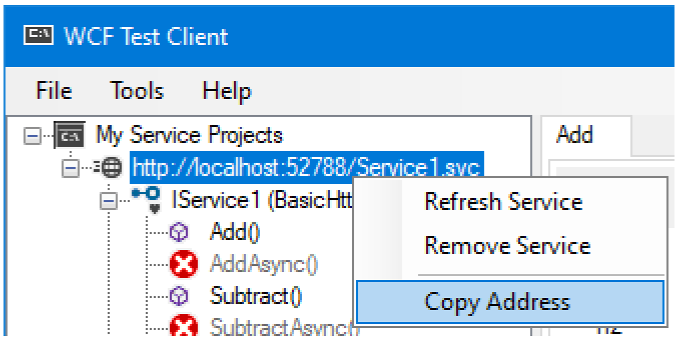
c. You can also see the raw XML request and response sent to and received from the web service.

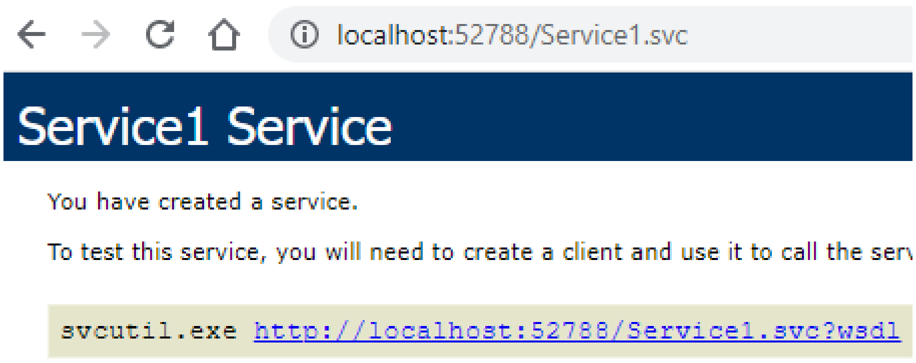


```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <Action s:mustUnderstand="1" xmlns="http://schemas.microsoft.com/ws/2005/05/addressing/none
  </s:Header>
  <s:Body>
    <Add xmlns="http://tempuri.org/">
      <n1>4</n1>
      <n2>7</n2>
    </Add>
```

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header />
  <s:Body>
    <AddResponse xmlns="http://tempuri.org/">
      <AddResult>11</AddResult>
    </AddResponse>
  </s:Body>
</s:Envelope>
```

d. Run a web browser and check WSDL description of the web service at: http://localhost:52788/Service1.svc?wsdl (the port number in your case can be different from 52788 as in the above example; please, notice the right port number from the *WCF Test Client* window as shown below:



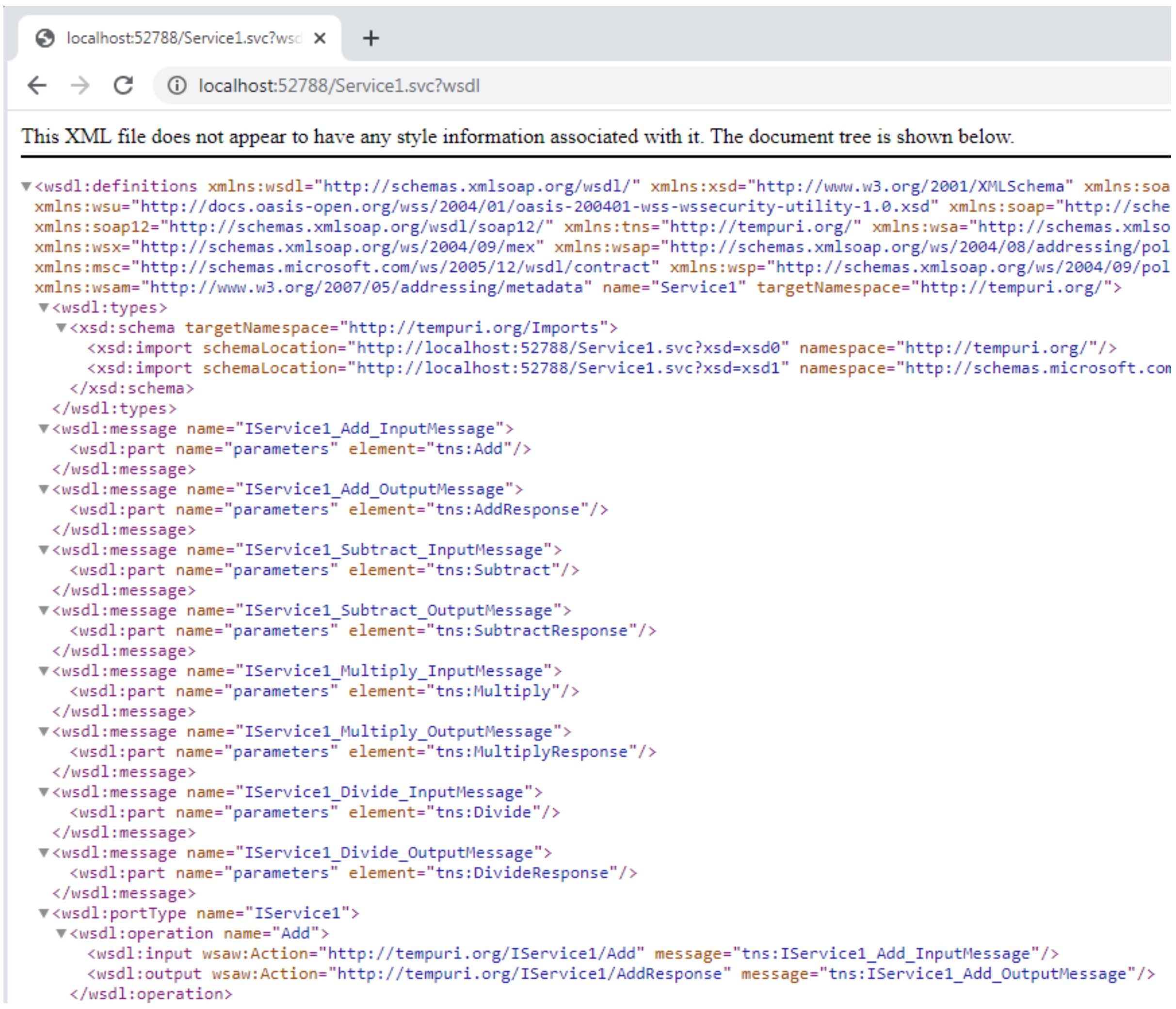


Service1 Service

You have created a service.

To test this service, you will need to create a client and use it to call the serv

```
svcutil.exe http://localhost:52788/Service1.svc?wsdl
```

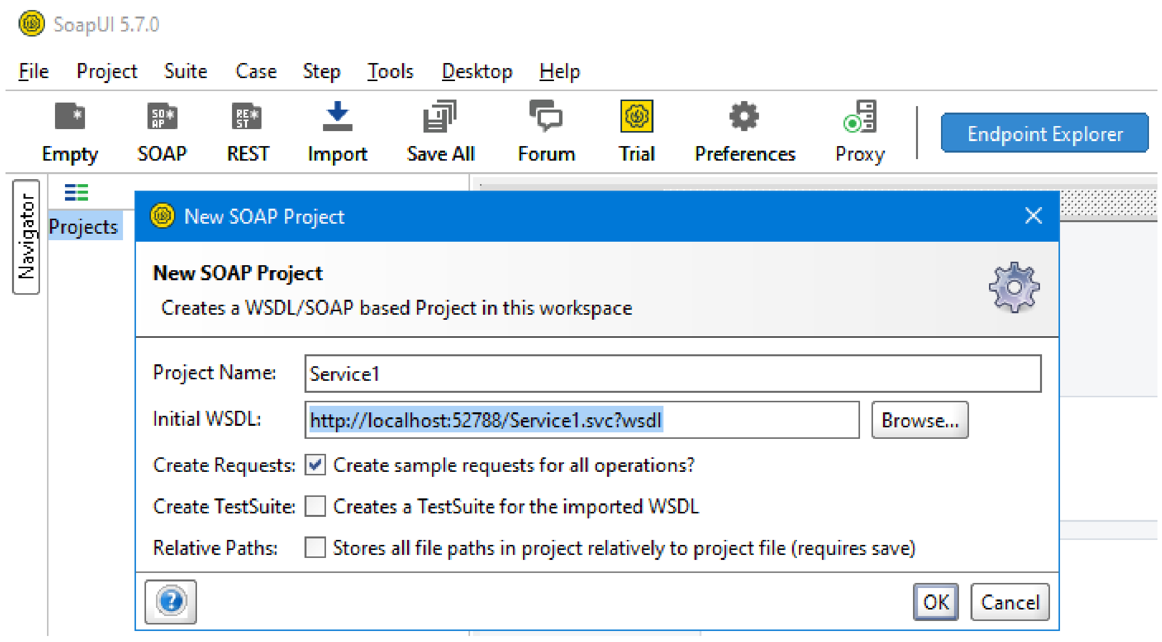


localhost:52788/Service1.svc?wsc
localhost:52788/Service1.svc?wsdl

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

```
▼<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soa
 xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd" xmlns:soap="http://sche
 xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/" xmlns:tns="http://tempuri.org/" xmlns:wsa="http://schemas.xmlso
 xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex" xmlns:wsap="http://schemas.xmlsoap.org/ws/2004/08/addressing/pol
 xmlns:msc="http://schemas.microsoft.com/ws/2005/12/wsdl/contract" xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/pol
 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="http://localhost:52788/Service1.svc?xsd=xsd0" namespace="http://tempuri.org/"/>
       <xsd:import schemaLocation="http://localhost:52788/Service1.svc?xsd=xsd1" namespace="http://schemas.microsoft.com
     </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:MultiplyResponse"/>
   </wsdl:message>
  ▼<wsdl:message name="IService1_Divide_InputMessage">
     <wsdl:part name="parameters" element="tns:Divide"/>
   </wsdl:message>
  ▼<wsdl:message name="IService1_Divide_OutputMessage">
     <wsdl:part name="parameters" element="tns:DivideResponse"/>
   </wsdl:message>
  ▼<wsdl:portType name="IService1">
    ▼<wsdl:operation name="Add">
       <wsdl:input wsaw:Action="http://tempuri.org/IService1/Add" message="tns:IService1_Add_InputMessage"/>
       <wsdl:output wsaw:Action="http://tempuri.org/IService1/AddResponse" message="tns:IService1_Add_OutputMessage"/>
     </wsdl:operation>
```

e. You can also test the web service using SoapUI tool. Run the *SoapUI* on your PC and create a new SOAP Project. Provide the URL to the wsdl file, e.g. http://localhost:52788/Service1.svc?wsdl:

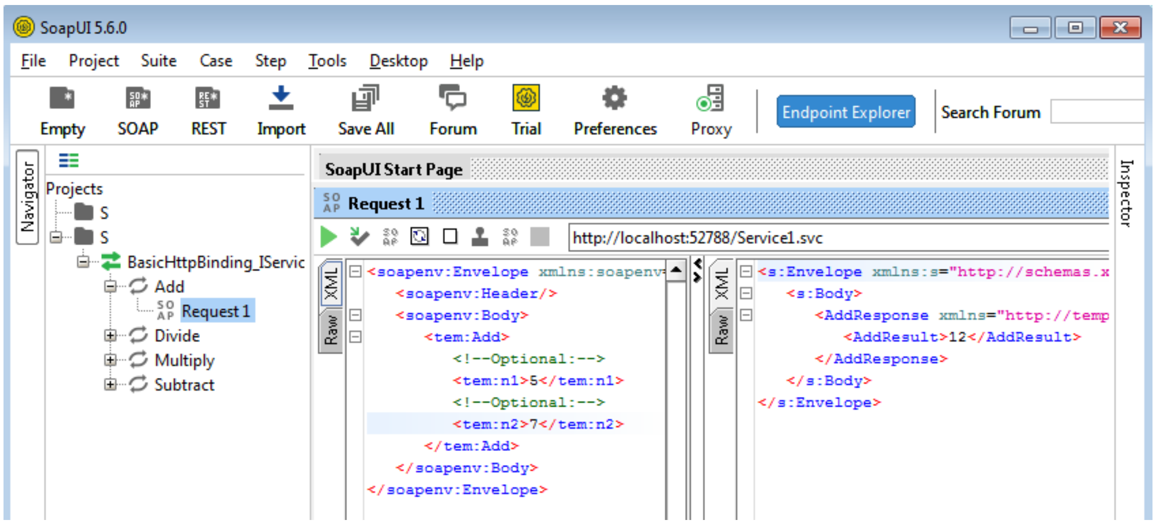


SoapUI 5.7.0
File Project Suite Case Step Tools Desktop Help
Empty SOAP REST Import Save All Forum Trial Preferences Proxy Endpoint Explorer
Navigator
Projects
New SOAP Project

New SOAP Project
Creates a WSDL/SOAP based Project in this workspace

Project Name:	Service1
Initial WSDL:	http://localhost:52788/Service1.svc?wsdl
Create Requests:	☑ Create sample requests for all operations?
Create TestSuite:	☐ Creates a TestSuite for the imported WSDL
Relative Paths:	☐ Stores all file paths in project relatively to project file (requires save)

f. Select one of the methods, e.g. Add; set values of n1 and n2 operands in the request template and invoke the service (press on the green triangle button). Notice the response (switch between Xml and Raw). Experiment with different service's methods and values.



http://localhost:52788/Service1.svc

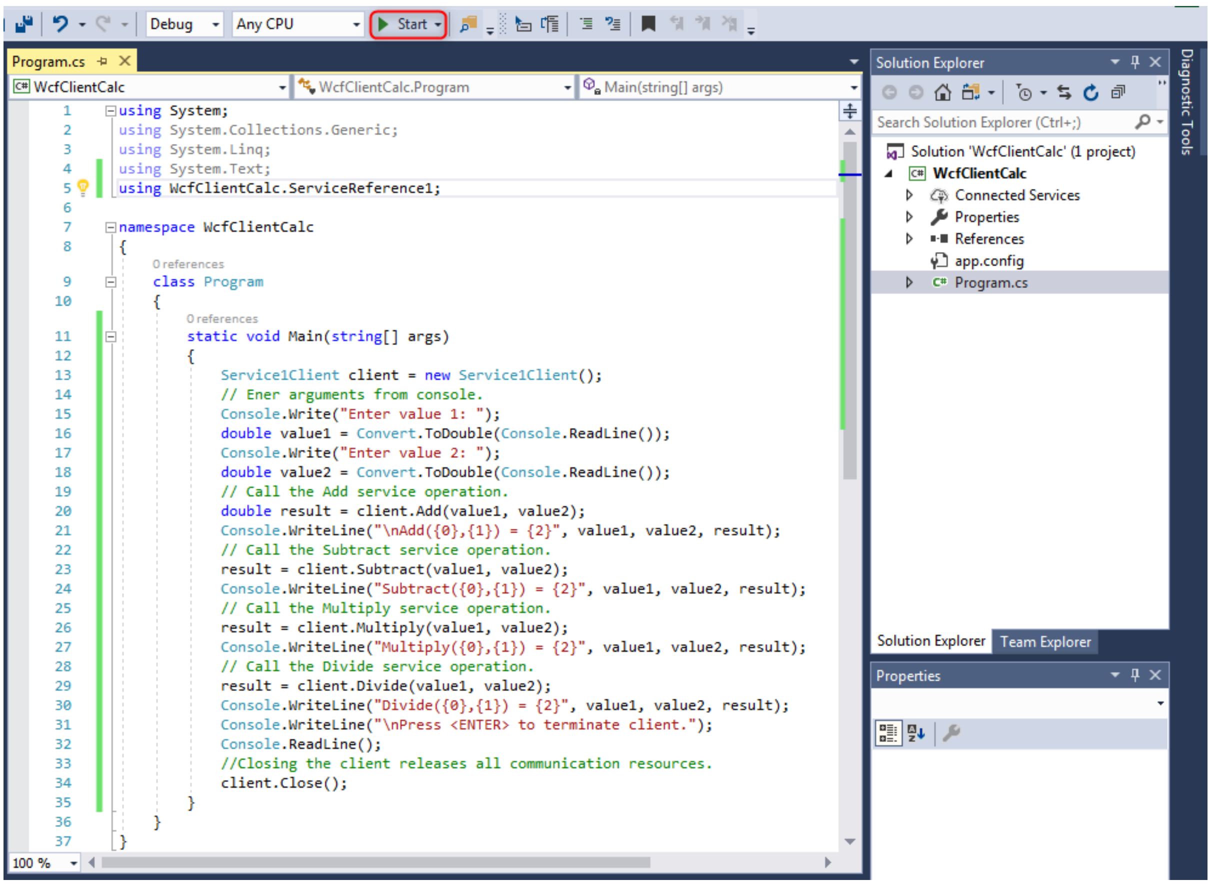
```
<soapenv:Envelope xmlns:soapenv=
   <soapenv:Header/>
   <soapenv:Body>
      <tem:Add>
         <!--Optional:-->
         <tem:n1>5</tem:n1>
         <!--Optional:-->
         <tem:n2>7</tem:n2>
      </tem:Add>
   </soapenv:Body>
</soapenv:Envelope>
```

```
<s:Envelope xmlns:s="http://schemas.x
   <s:Body>
      <AddResponse xmlns="http://temp
         <AddResult>12</AddResult>
      </AddResponse>
   </s:Body>
</s:Envelope>
```

Step 3: Exploring and executing the WCF Web Service Client

a. Open a new copy of Visual Studio (make sure the first Visual Studio still executes the WcfServerCalc project).

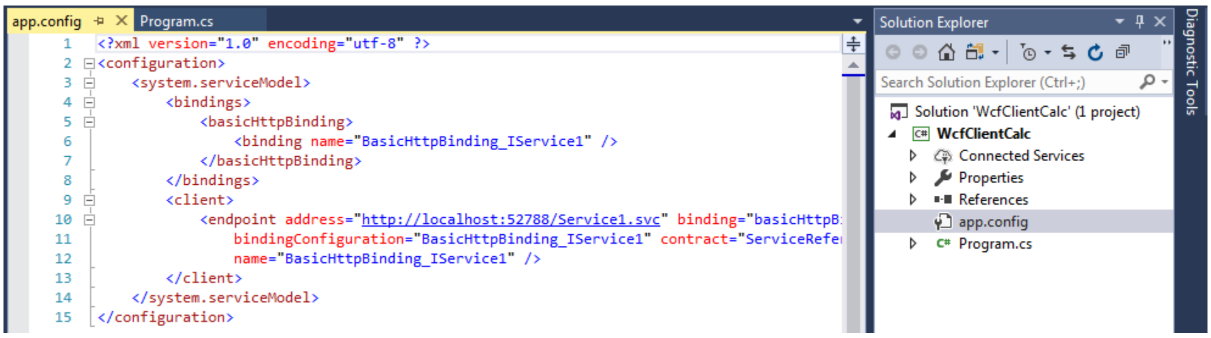
b. Open the **WcfClientCalc** project in the solution Explorer panel (open from C:\Users\tennin\source\repos) and open the *Program.cs* file. The **WcfClientCalc** is a desktop application which invokes different operations of the **WcfServerCalc** service.



```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using WcfClientCalc.ServiceReference1;

namespace WcfClientCalc
{
    class Program
    {
        static void Main(string[] args)
        {
            Service1Client client = new Service1Client();
            // Ener arguments from console.
            Console.Write("Enter value 1: ");
            double value1 = Convert.ToDouble(Console.ReadLine());
            Console.Write("Enter value 2: ");
            double value2 = Convert.ToDouble(Console.ReadLine());
            // Call the Add service operation.
            double result = client.Add(value1, value2);
            Console.WriteLine("\nAdd({0},{1}) = {2}", value1, value2, result);
            // Call the Subtract service operation.
            result = client.Subtract(value1, value2);
            Console.WriteLine("Subtract({0},{1}) = {2}", value1, value2, result);
            // Call the Multiply service operation.
            result = client.Multiply(value1, value2);
            Console.WriteLine("Multiply({0},{1}) = {2}", value1, value2, result);
            // Call the Divide service operation.
            result = client.Divide(value1, value2);
            Console.WriteLine("Divide({0},{1}) = {2}", value1, value2, result);
            Console.WriteLine("\nPress <ENTER> to terminate client.");
            Console.ReadLine();
            //Closing the client releases all communication resources.
            client.Close();
        }
    }
}
```

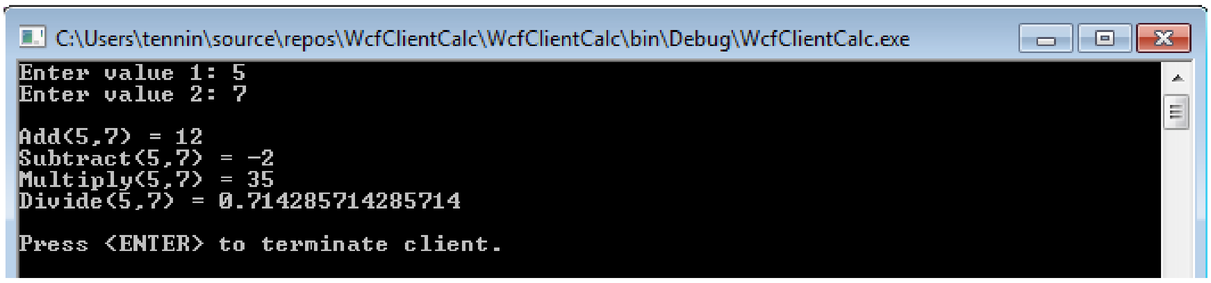
c. Explore the client's program code in **Program.cs** and notice the way a web service is invoked

d. Check the **app.config** which specifies the target Web Service. Notice the URL address of the service. Change it if needed.



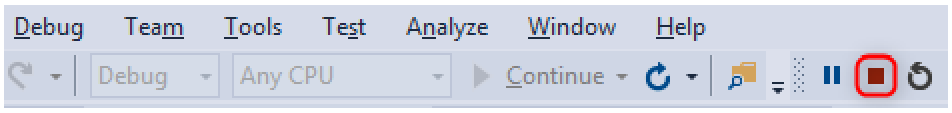
```xml
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
    <system.serviceModel>
        <bindings>
            <basicHttpBinding>
                <binding name="BasicHttpBinding_IService1" />
            </basicHttpBinding>
        </bindings>
        <client>
            <endpoint address="http://localhost:52788/Service1.svc" binding="basicHttpB
                bindingConfiguration="BasicHttpBinding_IService1" contract="ServiceRefe
                name="BasicHttpBinding_IService1" />
        </client>
    </system.serviceModel>
</configuration>
```

e. Execute (run) the client.


C:\Users\tennin\source\repos\WcfClientCalc\WcfClientCalc\bin\Debug\WcfClientCalc.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.
```

a. Close the WcfClientCalc and stop debugging of the WcfServerCalc.



Step 4: Launching a Web Service on AWS Elastic Beanstalk PaaS

AWS Elastic Beanstalk is an easy-to-use PaaS service for deploying and scaling web applications and services developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker on familiar servers such as Apache, Nginx, Passenger, and IIS.

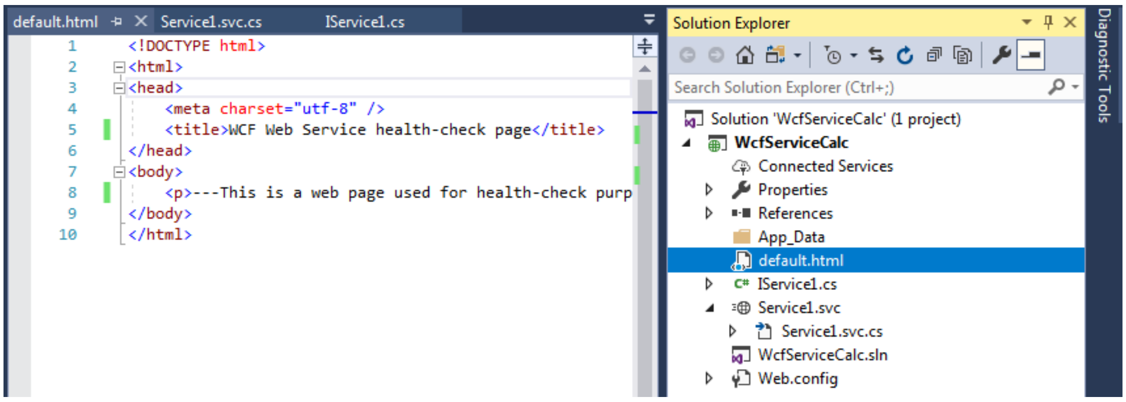
You can simply upload your code and Elastic Beanstalk automatically handles the deployment, from capacity provisioning, load balancing, auto-scaling to application health monitoring. At the same time, you retain full control over the AWS resources powering your application and can access the underlying resources at any time.

You need to pay for the AWS resources needed to store and run your applications. Usually your code runs on one or several dedicated VMs.

Launching a Web Service on the AWS Elastic Beanstalk PaaS

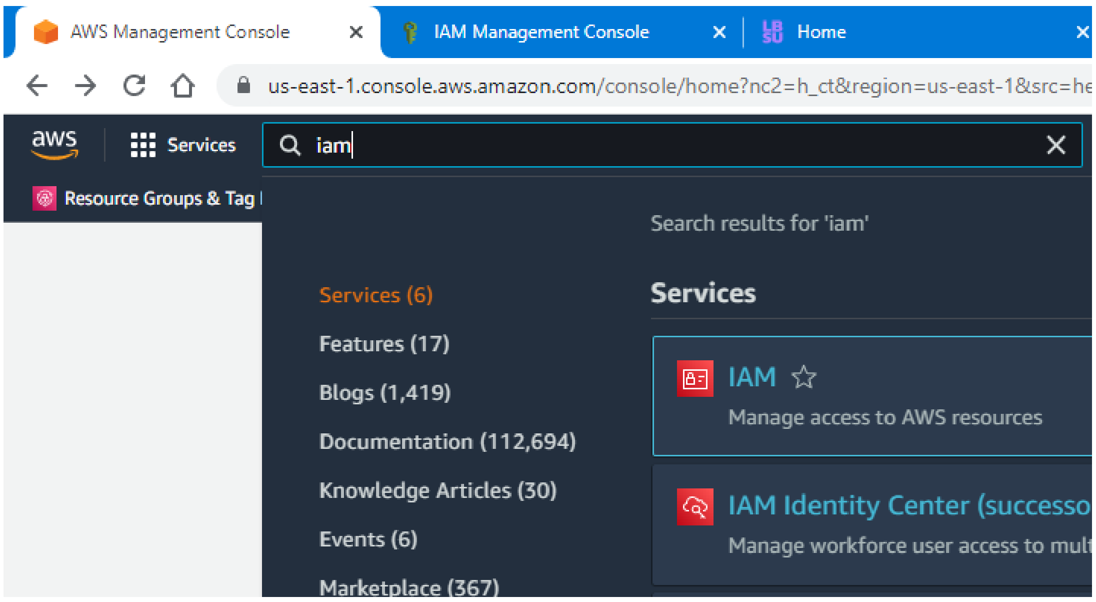
1. Notice the static html page **default.html** that was added into the WcfServerCalc project.

AWS Load Balancer periodically (every 10 seconds) sends requests to the registered instances to test their status by sending HTTP GET requests. These tests are called health checks. An instance is considered healthy if it returns a 200 response code within the health check interval. By default, the load balancer is configured to open a TCP connection on port 80. If the instance acknowledges the connection, it is considered healthy. Thus, you need to have some web page in the root of your Web service to pass the health check when the instance is deployed. Later on, you will be able to override this setting by specifying another path, e.g. a URL to the WSDL description of your web service.

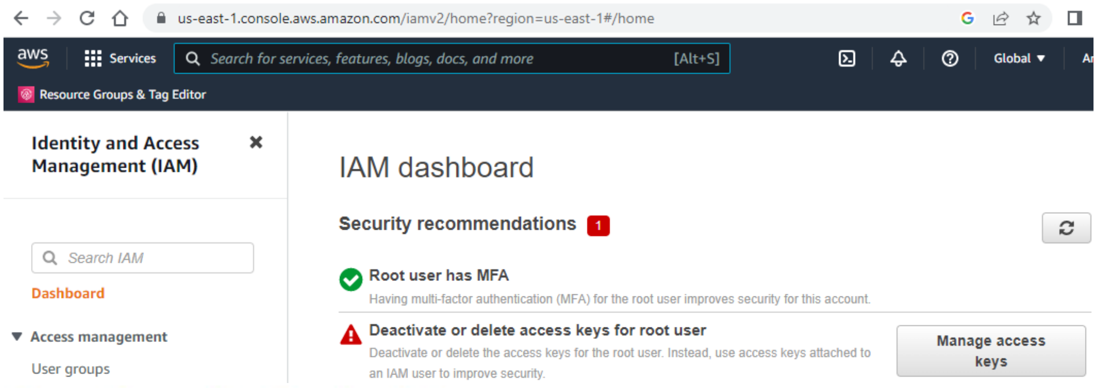


```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <title>WCF Web Service health-check page</title>
</head>
<body>
    <p>---This is a web page used for health-check purp
</body>
</html>
```

2. Configure the AWS default profile on Visual Studio to get access to your AWS account
   a. Log in to your AWS account https://aws.amazon.com/ and select/search for IAM (Identity and Access Management) centre:

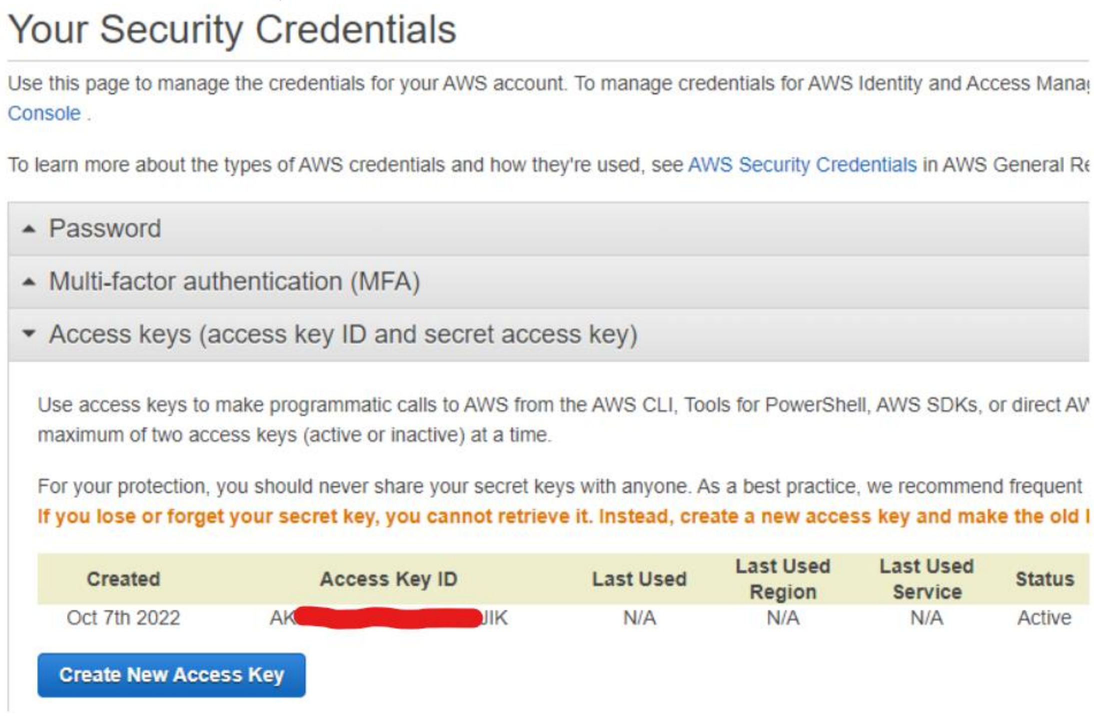


Click on 'Manage access keys' and create a New Access Key


us-east-1.console.aws.amazon.com/iamv2/home?region=us-east-1#/home
aws
Services
Search for services, features, blogs, docs, and more
[Alt+S]
Global
Resource Groups & Tag Editor
Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
IAM dashboard
Security recommendations 1
Manage access keys

Root user has MFA
Having multi-factor authentication (MFA) for the root user improves security for this account.

Deactivate or delete access keys for root user
Deactivate or delete the access keys for the root user. Instead, use access keys attached to an IAM user to improve security.


Your Security Credentials
Password
Multi-factor authentication (MFA)
Access keys (access key ID and secret access key)
Create New Access Key

Use this page to manage the credentials for your AWS account. To manage credentials for AWS Identity and Access Mana Console .

To learn more about the types of AWS credentials and how they're used, see AWS Security Credentials in AWS General Re

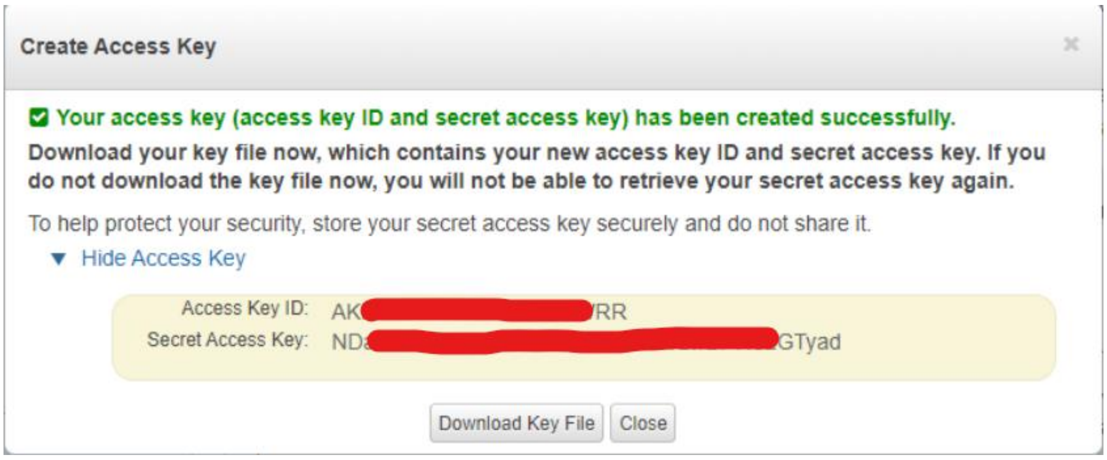
Use access keys to make programmatic calls to AWS from the AWS CLI, Tools for PowerShell, AWS SDKs, or direct AV maximum of two access keys (active or inactive) at a time.

For your protection, you should never share your secret keys with anyone. As a best practice, we recommend frequent

If you lose or forget your secret key, you cannot retrieve it. Instead, create a new access key and make the old I

Created	Access Key ID	Last Used	Last Used Region	Last Used Service	Status
Oct 7th 2022	AK[redacted]JIK	N/A	N/A	N/A	Active

You will be provided a pair of Access Key ID and Secret Access Key. Download a key file with these keys and save it for the future use. **Note**, that the Access Key file is different from the Key pair file which you used for the passwordless access to VMs.


Create Access Key

Your access key (access key ID and secret access key) has been created successfully.

Download your key file now, which contains your new access key ID and secret access key. If you do not download the key file now, you will not be able to retrieve your secret access key again.

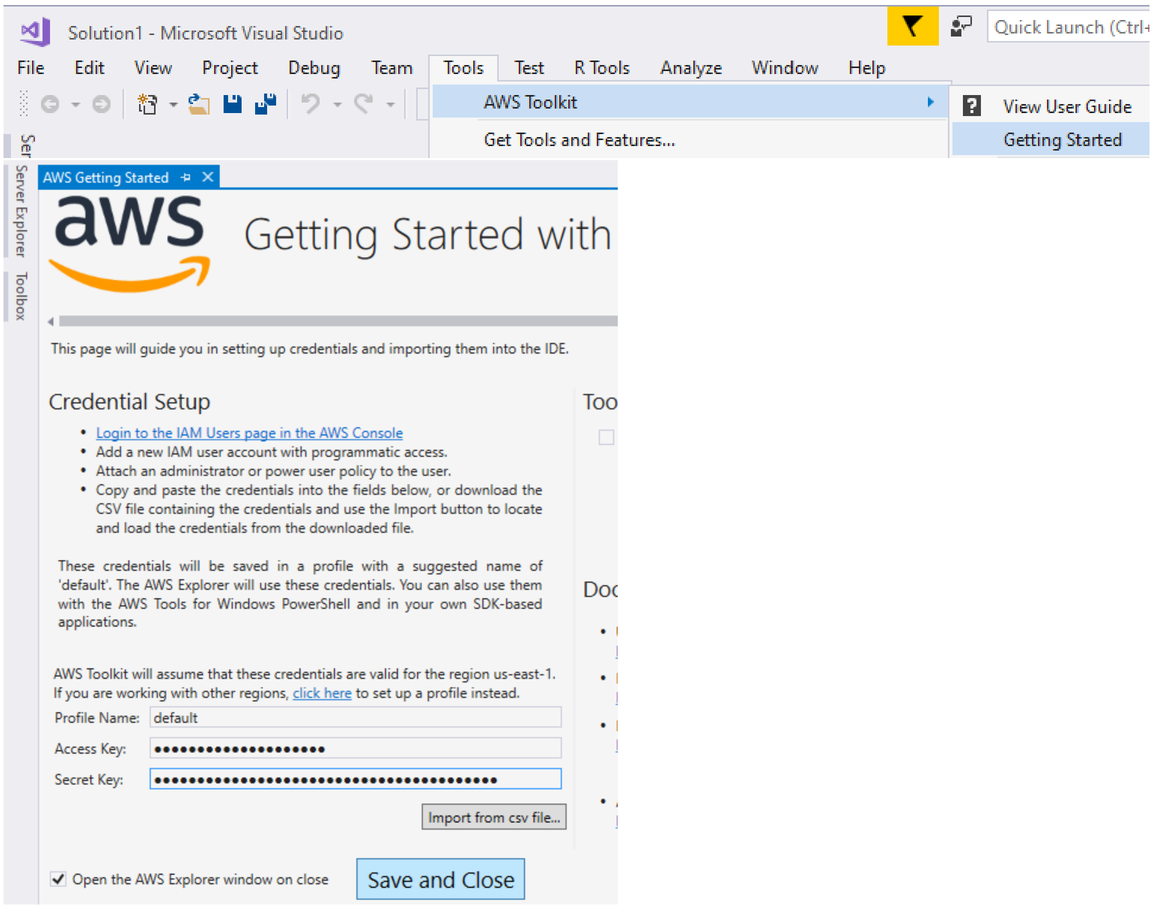
To help protect your security, store your secret access key securely and do not share it.

▼ Hide Access Key

Access Key ID:	AK[redacted]RR
Secret Access Key:	ND[redacted]GTyad

b. Copy and paste your credentials into USER_HOME/.aws/credentials and save the file

3. Open the AWS Toolkit on Visual Studio and copy/paste your Access Key and Secret Key:



This page will guide you in setting up credentials and importing them into the IDE.

Credential Setup

- Login to the IAM Users page in the AWS Console
- Add a new IAM user account with programmatic access.
- Attach an administrator or power user policy to the user.
- Copy and paste the credentials into the fields below, or download the CSV file containing the credentials and use the Import button to locate and load the credentials from the downloaded file.

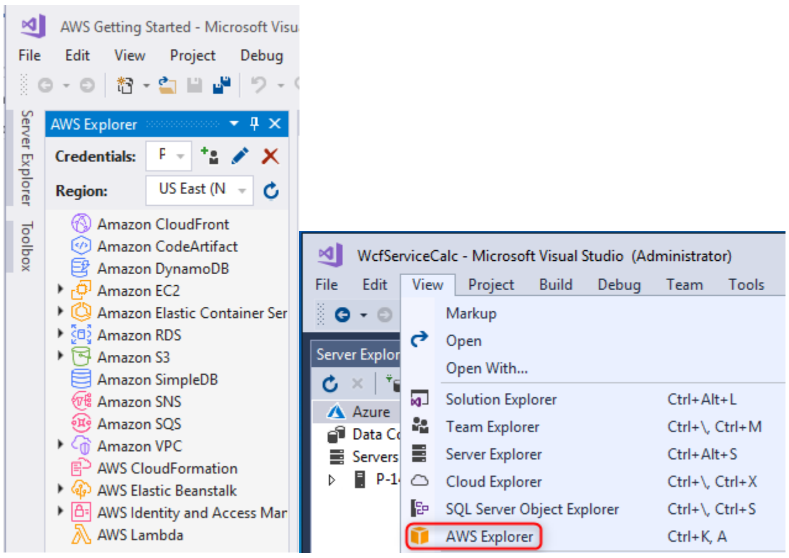
These credentials will be saved in a profile with a suggested name of 'default'. The AWS Explorer will use these credentials. You can also use them with the AWS Tools for Windows PowerShell and in your own SDK-based applications.

AWS Toolkit will assume that these credentials are valid for the region us-east-1. If you are working with other regions, click here to set up a profile instead.

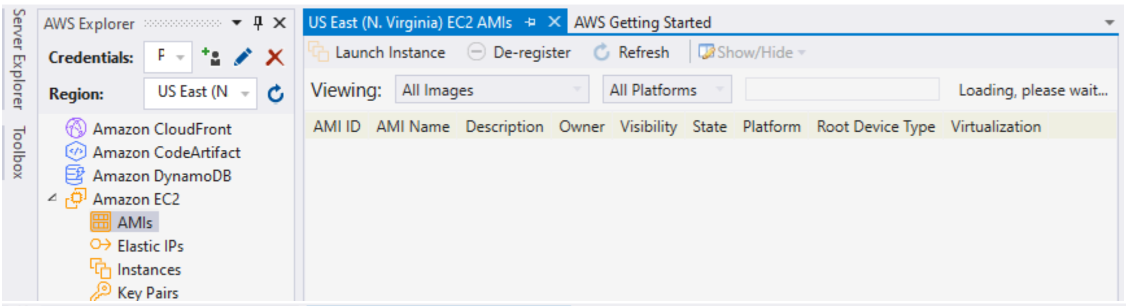
Profile Name:	default
Access Key:	••••••••••••••••••••
Secret Key:	••

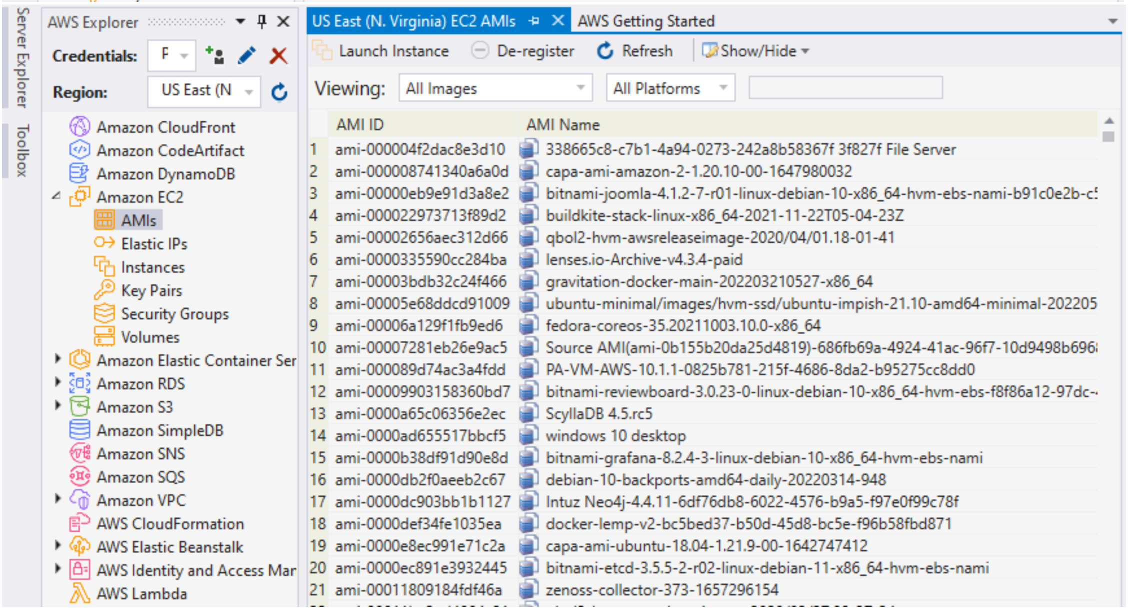
✓ Open the AWS Explorer window on close

Now you can see the AWS Explorer panel on the left (if not, Select ‘View’ -> ‘AWS Explorer’)



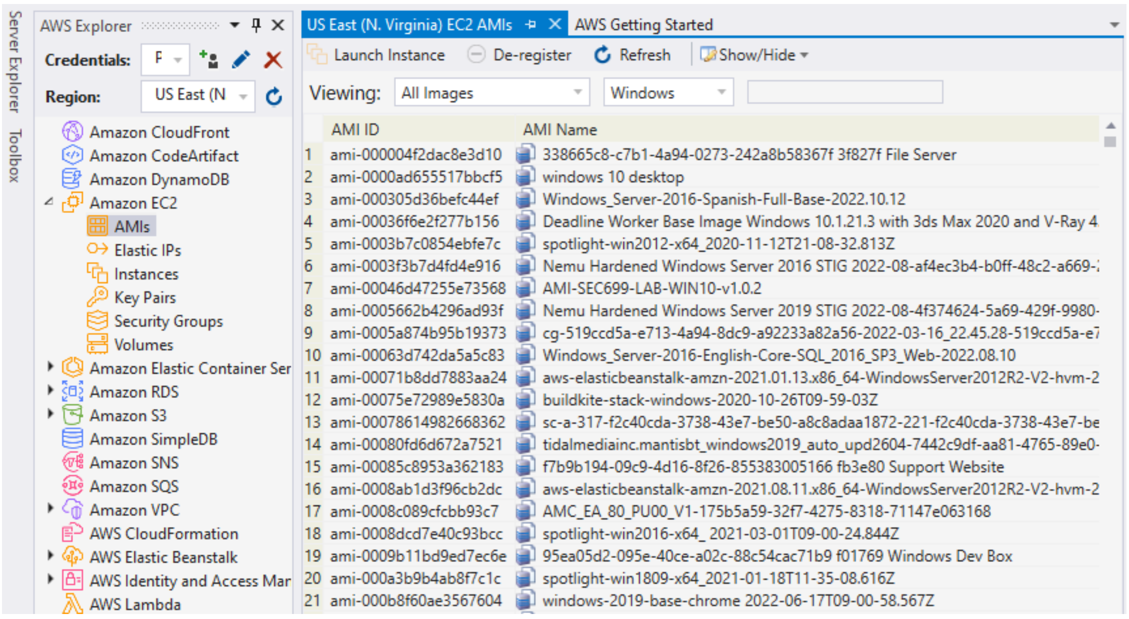
You can use AWS Explorer to view and manage your resources on AWS. For example, let us list available Virtual Images (AMIs) available. With this purpose, select ‘Amazon EC2’ and click on ’AMIs’. Wait for the AWS Toolkit to retrieve the list of available VM images:





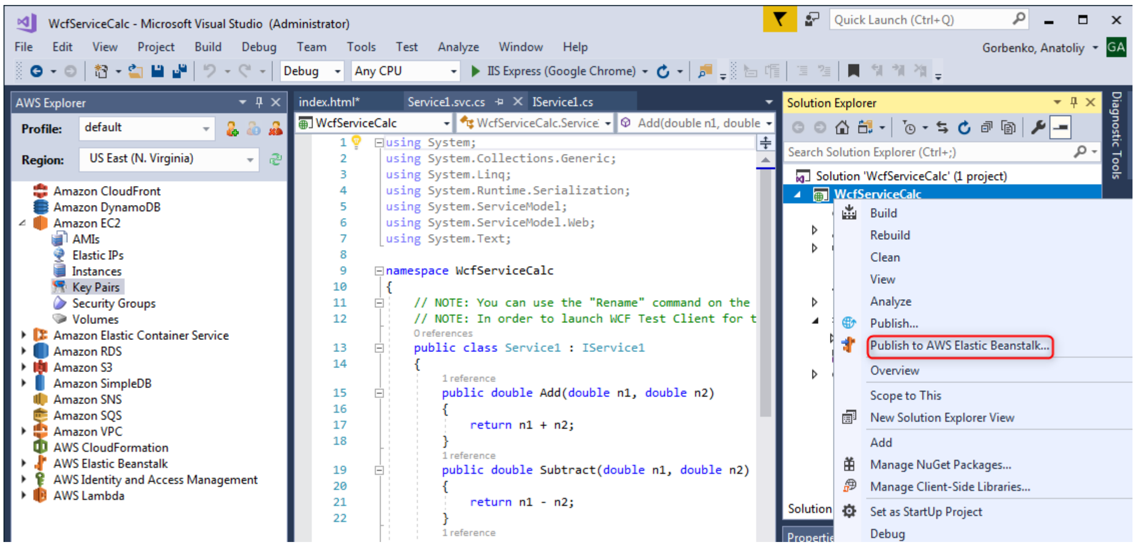
	AMI ID	AMI Name
1	ami-000004f2dac8e3d10	338665c8-c7b1-4a94-0273-242a8b58367f 3f827f File Server
2	ami-000008741340a6a0d	capa-ami-amazon-2-1.20.10-00-1647980032
3	ami-00000eb9e91d3a8e2	bitnami-joomla-4.1.2-7-r01-linux-debian-10-x86_64-hvm-ebs-nami-b91c0e2b-c
4	ami-000022973713f89d2	buildkite-stack-linux-x86_64-2021-11-22T05-04-23Z
5	ami-00002656aec312d66	qbol2-hvm-awsreleaseimage-2020/04/01.18-01-41
6	ami-0000335590cc284ba	lenses.io-Archive-v4.3.4-paid
7	ami-00003bdb32c24f466	gravitation-docker-main-202203210527-x86_64
8	ami-00005e68ddcd91009	ubuntu-minimal/images/hvm-ssd/ubuntu-impish-21.10-amd64-minimal-202205
9	ami-00006a129f1fb9ed6	fedora-coreos-35.20211003.10.0-x86_64
10	ami-00007281eb26e9ac5	Source AMI(ami-0b155b20da25d4819)-686fb69a-4924-41ac-96f7-10d9498b696
11	ami-000089d74ac3a4fdd	PA-VM-AWS-10.1.1-0825b781-215f-4686-8da2-b95275cc8dd0
12	ami-00009903158360bd7	bitnami-reviewboard-3.0.23-0-linux-debian-10-x86_64-hvm-ebs-f8f86a12-97dc-
13	ami-0000a65c06356e2ec	ScyllaDB 4.5.rc5
14	ami-0000ad655517bbcf5	windows 10 desktop
15	ami-0000b38df91d90e8d	bitnami-grafana-8.2.4-3-linux-debian-10-x86_64-hvm-ebs-nami
16	ami-0000db2f0aeeb2c67	debian-10-backports-amd64-daily-20220314-948
17	ami-0000dc903bb1b1127	Intuz Neo4j-4.4.11-6df76db8-6022-4576-b9a5-f97e0f99c78f
18	ami-0000def34fe1035ea	docker-lemp-v2-bc5bed37-b50d-45d8-bc5e-f96b58fbd871
19	ami-0000e8ec991e71c2a	capa-ami-ubuntu-18.04-1.21.9-00-1642747412
20	ami-0000ec891e3932445	bitnami-etcd-3.5.5-2-r02-linux-debian-11-x86_64-hvm-ebs-nami
21	ami-00011809184fdf46a	zenoss-collector-373-1657296154

You can filter images by the type of operating system:



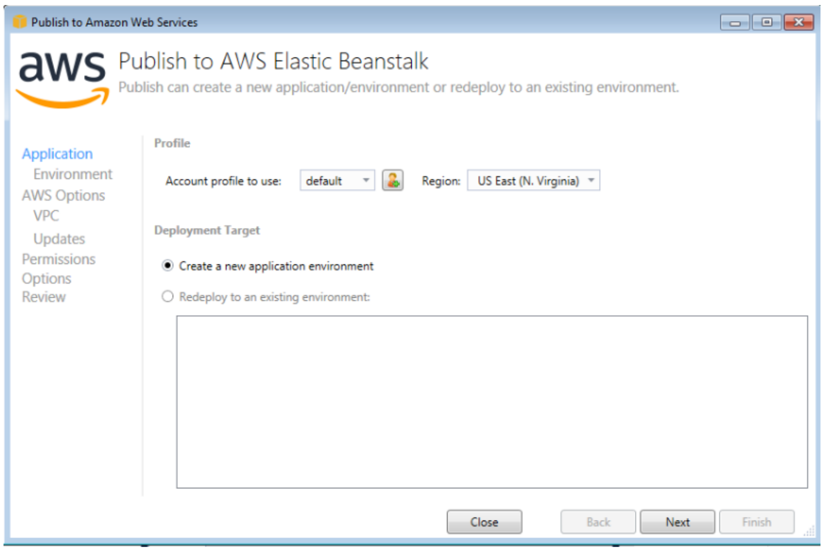
	AMI ID	AMI Name
1	ami-000004f2dac8e3d10	338665c8-c7b1-4a94-0273-242a8b58367f 3f827f File Server
2	ami-0000ad655517bbcf5	windows 10 desktop
3	ami-000305d36befc44ef	Windows_Server-2016-Spanish-Full-Base-2022.10.12
4	ami-00036f6e2f277b156	Deadline Worker Base Image Windows 10.1.21.3 with 3ds Max 2020 and V-Ray 4
5	ami-0003b7c0854ebfe7c	spotlight-win2012-x64_2020-11-12T21-08-32.813Z
6	ami-0003f3b7d4fd4e916	Nemu Hardened Windows Server 2016 STIG 2022-08-af4ec3b4-b0ff-48c2-a669-:
7	ami-00046d47255e73568	AMI-SEC699-LAB-WIN10-v1.0.2
8	ami-0005662b4296ad93f	Nemu Hardened Windows Server 2019 STIG 2022-08-4f374624-5a69-429f-9980-
9	ami-0005a874b95b19373	cg-519ccd5a-e713-4a94-8dc9-a92233a82a56-2022-03-16_22.45.28-519ccd5a-e7
10	ami-00063d742da5a5c83	Windows_Server-2016-English-Core-SQL_2016_SP3_Web-2022.08.10
11	ami-00071b8dd7883aa24	aws-elasticbeanstalk-amzn-2021.01.13.x86_64-WindowsServer2012R2-V2-hvm-2
12	ami-00075e72989e5830a	buildkite-stack-windows-2020-10-26T09-59-03Z
13	ami-00078614982668362	sc-a-317-f2c40cda-3738-43e7-be50-a8c8adaa1872-221-f2c40cda-3738-43e7-be
14	ami-00080fd6d672a7521	tidalmediainc.mantisbt_windows2019_auto_upd2604-7442c9df-aa81-4765-89e0-
15	ami-00085c8953a362183	f7b9b194-09c9-4d16-8f26-855383005166 fb3e80 Support Website
16	ami-0008ab1d3f96cb2dc	aws-elasticbeanstalk-amzn-2021.08.11.x86_64-WindowsServer2012R2-V2-hvm-2
17	ami-0008c089cfcbb93c7	AMC_EA_80_PU00_V1-175b5a59-32f7-4275-8318-71147e063168
18	ami-0008dcd7e40c93bcc	spotlight-win2016-x64_ 2021-03-01T09-00-24.844Z
19	ami-0009b11bd9ed7ec6e	95ea05d2-095e-40ce-a02c-88c54cac71b9 f01769 Windows Dev Box
20	ami-000a3b9b4ab8f7c1c	spotlight-win1809-x64_2021-01-18T11-35-08.616Z
21	ami-000b8f60ae3567604	windows-2019-base-chrome 2022-06-17T09-00-58.567Z

4. Now. Let us deploy your Calculator Service to AWS Elastic Beanstalk Paas. Right Click on the **WcfServiceCalc** project and select **Publish to AWS Elastic Beanstalk**



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;
using System.Text;

namespace WcfServiceCalc
{
    // NOTE: You can use the "Rename" command on the
    // NOTE: In order to launch WCF Test Client for t
    0 references
    public class Service1 : IService1
    {
        1 reference
        public double Add(double n1, double n2)
        {
            return n1 + n2;
        }
        1 reference
        public double Subtract(double n1, double n2)
        {
            return n1 - n2;
        }
        1 reference
```

5. Now you will be guided through the AWS Elastic Beanstalk publish process
   a. Create a new application environment using the **default** account profile. You can change the region if you wish.



Publish to AWS Elastic Beanstalk

Publish can create a new application/environment or redeploy to an existing environment.

Profile

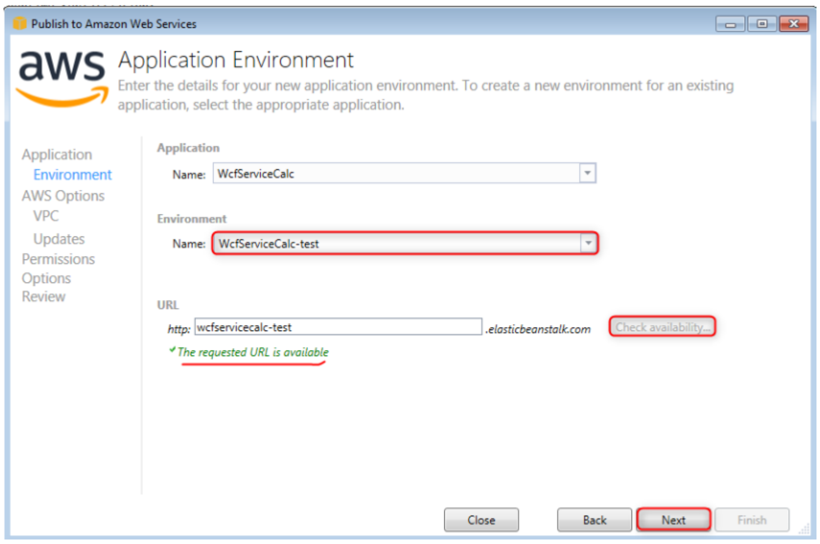
Account profile to use: default Region: US East (N. Virginia)

Deployment Target

● Create a new application environment

○ Redeploy to an existing environment:

b. Specify the environment name, URL and check its availability. **Note:** the default environment name can be already taken by your classmates. So, include your student id as a part of the name.



Application Environment

Enter the details for your new application environment. To create a new environment for an existing application, select the appropriate application.

Application

Name: WcfServiceCalc

Environment

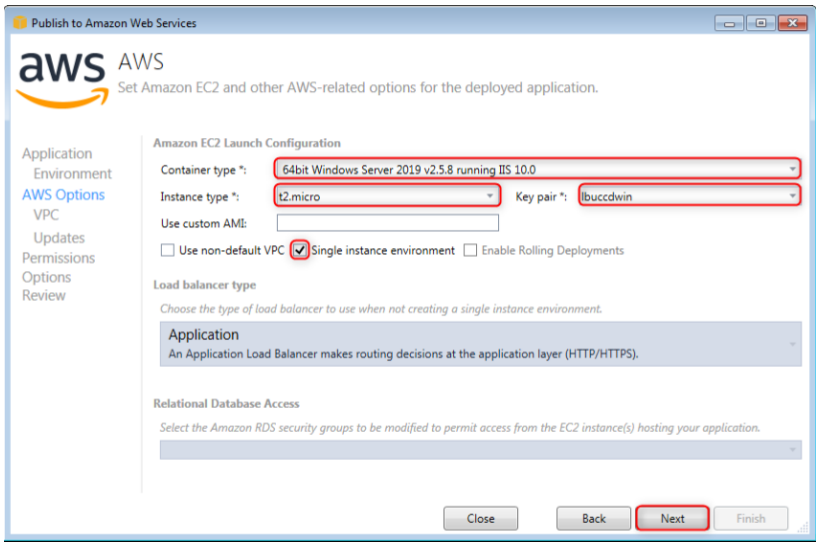
Name: WcfServiceCalc-test

URL

http: wcfservicecalc-test .elasticbeanstalk.com

✓ The requested URL is available

c. Configure the AWS options
   i. On the AWS Options page, in Amazon EC2 Launch Configuration, from the Container type drop-down list, choose an Amazon Machine Image (AMI) type that will be used for your application;
   ii. In the Instance type drop-down list, specify an Amazon EC2 instance type to use (micro instance will minimize the cost associated with running the instance);
   iii. In the Key pair drop-down list, you can choose an Amazon EC2 instance key pair (if you have) to use to sign in to the instances for management purpose;
   iv. select the Single instance environment box to deploy your application on a single Amazon EC2 instance


Publish to Amazon Web Services

AWS

Set Amazon EC2 and other AWS-related options for the deployed application.

Application
Environment
AWS Options
VPC
Updates
Permissions
Options
Review

Amazon EC2 Launch Configuration

Container type *: 64bit Windows Server 2019 v2.5.8 running IIS 10.0

Instance type *: t2.micro Key pair *: lbuccdwin

Use custom AMI:

☐ Use non-default VPC ☑ Single instance environment ☐ Enable Rolling Deployments

Load balancer type

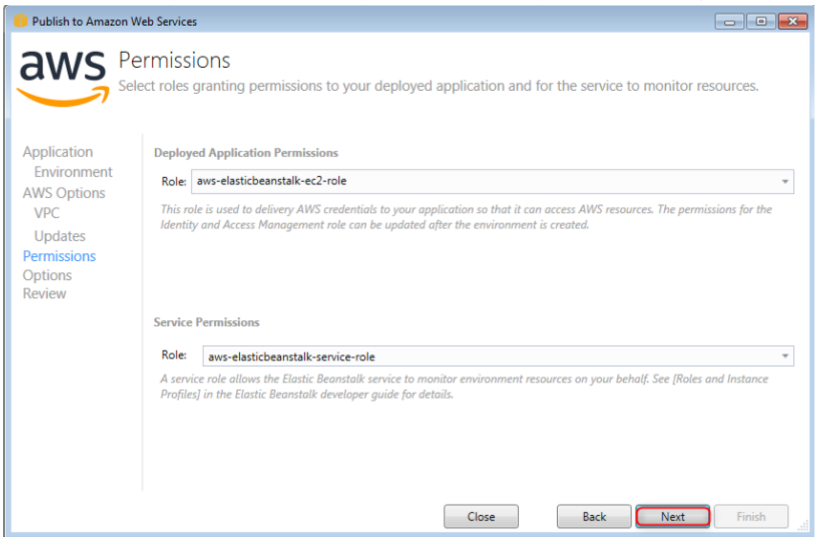
Choose the type of load balancer to use when not creating a single instance environment.

Application
An Application Load Balancer makes routing decisions at the application layer (HTTP/HTTPS).

Relational Database Access

Select the Amazon RDS security groups to be modified to permit access from the EC2 instance(s) hosting your application.

d. Leave default permissions for your instance; you can configure them latter on if you wish


Publish to Amazon Web Services

Permissions

Select roles granting permissions to your deployed application and for the service to monitor resources.

Application
Environment
AWS Options
VPC
Updates
Permissions
Options
Review

Deployed Application Permissions

Role: aws-elasticbeanstalk-ec2-role

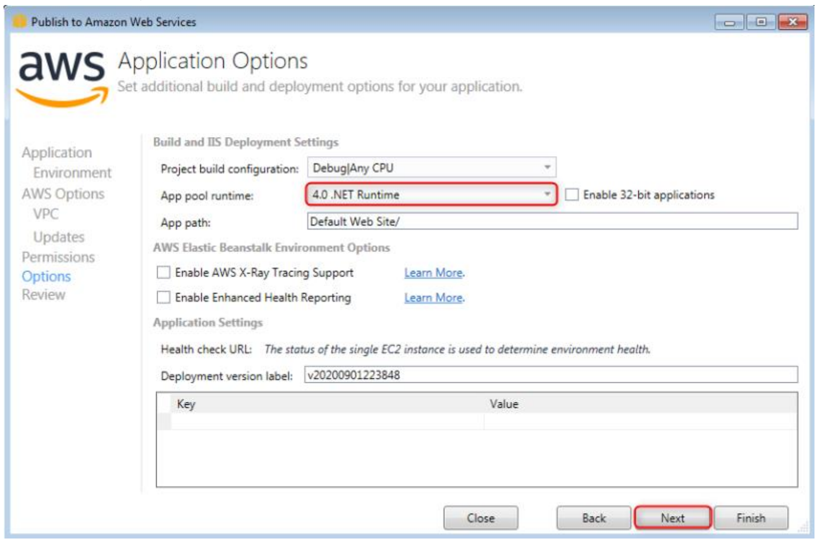
This role is used to delivery AWS credentials to your application so that it can access AWS resources. The permissions for the Identity and Access Management role can be updated after the environment is created.

Service Permissions

Role: aws-elasticbeanstalk-service-role

A service role allows the Elastic Beanstalk service to monitor environment resources on your behalf. See [Roles and Instance Profiles] in the Elastic Beanstalk developer guide for details.

e. Make sure you use the same version of the .NET Runtime environment


Publish to Amazon Web Services

Application Options

Set additional build and deployment options for your application.

Application
Environment
AWS Options
VPC
Updates
Permissions
Options
Review

Build and IIS Deployment Settings

Project build configuration: Debug|Any CPU

App pool runtime: 4.0 .NET Runtime ☐ Enable 32-bit applications

App path: Default Web Site/

AWS Elastic Beanstalk Environment Options

☐ Enable AWS X-Ray Tracing Support Learn More.

☐ Enable Enhanced Health Reporting Learn More.

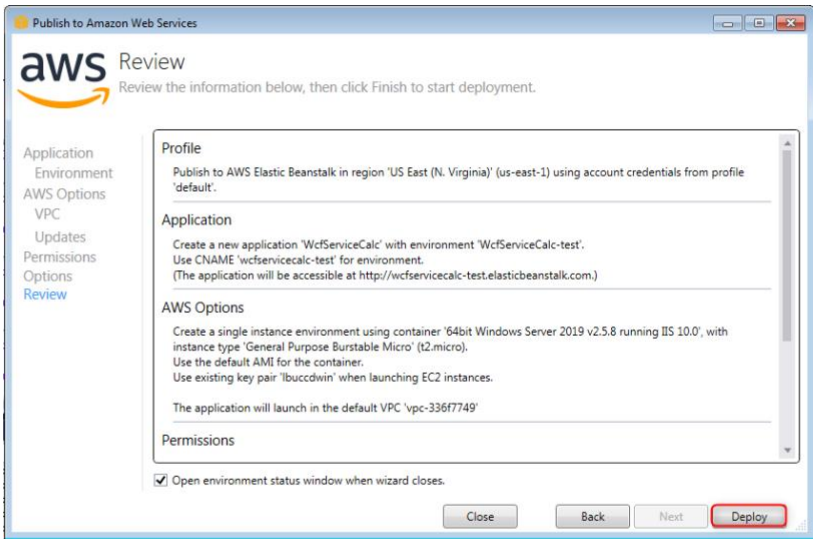
Application Settings

Health check URL: *The status of the single EC2 instance is used to determine environment health.*

Deployment version label: v20200901223848

Key	Value

f. Deploy your project



Review

Review the information below, then click Finish to start deployment.

Profile

Publish to AWS Elastic Beanstalk in region 'US East (N. Virginia)' (us-east-1) using account credentials from profile 'default'.

Application

Create a new application 'WcfServiceCalc' with environment 'WcfServiceCalc-test'.
Use CNAME 'wcfservicecalc-test' for environment.
(The application will be accessible at http://wcfservicecalc-test.elasticbeanstalk.com.)

AWS Options

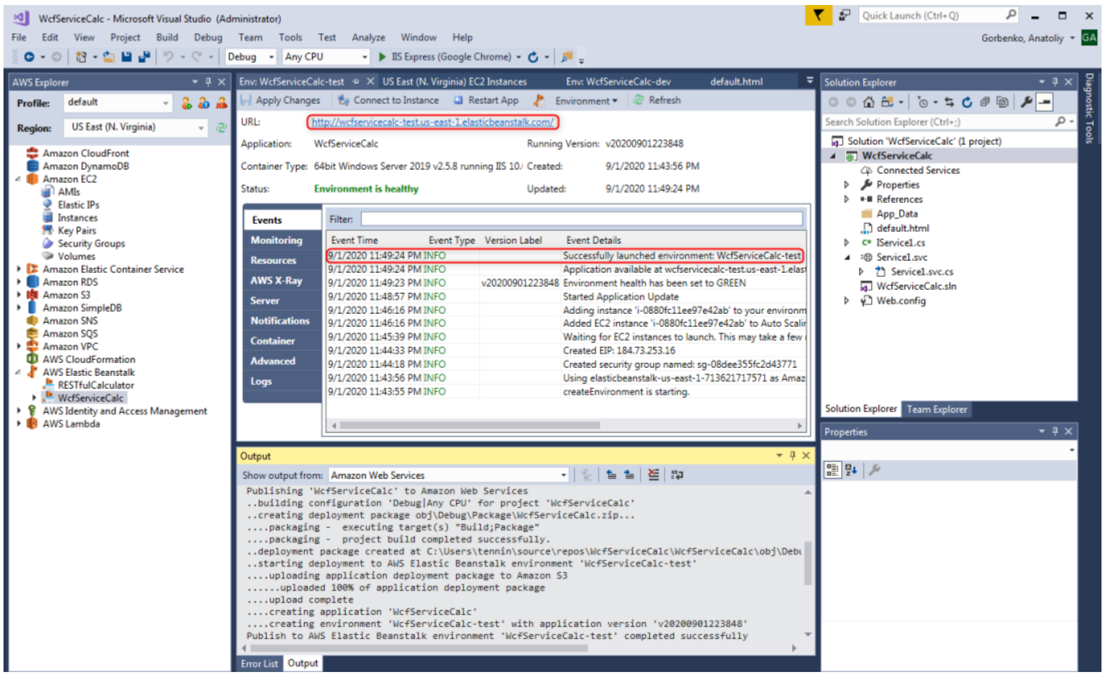
Create a single instance environment using container '64bit Windows Server 2019 v2.5.8 running IIS 10.0', with instance type 'General Purpose Burstable Micro' (t2.micro).
Use the default AMI for the container.
Use existing key pair 'lbuccdwin' when launching EC2 instances.

The application will launch in the default VPC 'vpc-336f7749'

Permissions

☑ Open environment status window when wizard closes.

g. Wait for the instance being launched and the project being deployed and open the web service URL



URL: http://wcfservicecalc-test.us-east-1.elasticbeanstalk.com/

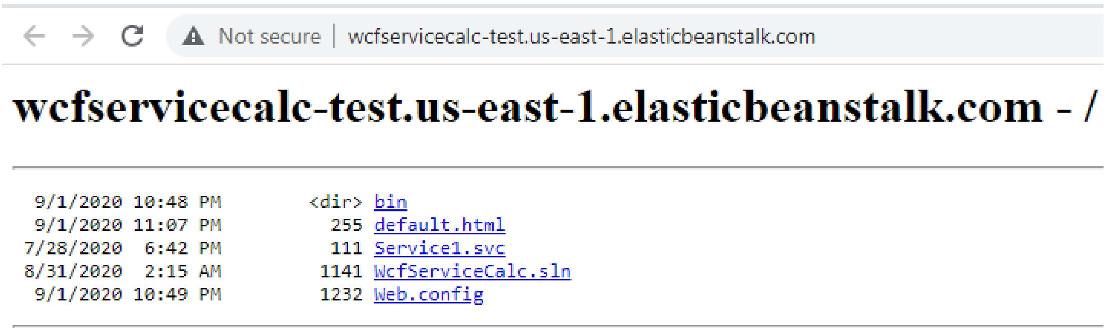
Application: WcfServiceCalc   Running Version: v20200901223848

Container Type: 64bit Windows Server 2019 v2.5.8 running IIS 10.   Created: 9/1/2020 11:43:56 PM

Status: Environment is healthy   Updated: 9/1/2020 11:49:24 PM

Event Time	Event Type	Version Label	Event Details
9/1/2020 11:49:24 PM	INFO		Successfully launched environment: WcfServiceCalc-test
9/1/2020 11:49:24 PM	INFO		Application available at wcfservicecalc-test.us-east-1.elas
9/1/2020 11:49:23 PM	INFO	v20200901223848	Environment health has been set to GREEN
9/1/2020 11:48:57 PM	INFO		Started Application Update
9/1/2020 11:46:16 PM	INFO		Adding instance 'i-0880fc11ee97e42ab' to your environm
9/1/2020 11:46:16 PM	INFO		Added EC2 instance 'i-0880fc11ee97e42ab' to Auto Scalin
9/1/2020 11:45:39 PM	INFO		Waiting for EC2 instances to launch. This may take a few
9/1/2020 11:44:33 PM	INFO		Created EIP: 184.73.253.16
9/1/2020 11:44:18 PM	INFO		Created security group named: sg-08dee355fc2d43771
9/1/2020 11:43:56 PM	INFO		Using elasticbeanstalk-us-east-1-713621717571 as Amaz
9/1/2020 11:43:55 PM	INFO		createEnvironment is starting.

```
Publishing 'WcfServiceCalc' to Amazon Web Services
..building configuration 'Debug|Any CPU' for project 'WcfServiceCalc'
..creating deployment package obj\Debug\Package\WcfServiceCalc.zip...
....packaging -  executing target(s) "Build;Package"
....packaging -  project build completed successfully.
..deployment package created at C:\Users\tennin\source\repos\WcfServiceCalc\WcfServiceCalc\obj\Debu
..starting deployment to AWS Elastic Beanstalk environment 'WcfServiceCalc-test'
....uploading application deployment package to Amazon S3
......uploaded 100% of application deployment package
....upload complete
....creating application 'WcfServiceCalc'
....creating environment 'WcfServiceCalc-test' with application version 'v20200901223848'
Publish to AWS Elastic Beanstalk environment 'WcfServiceCalc-test' completed successfully
```

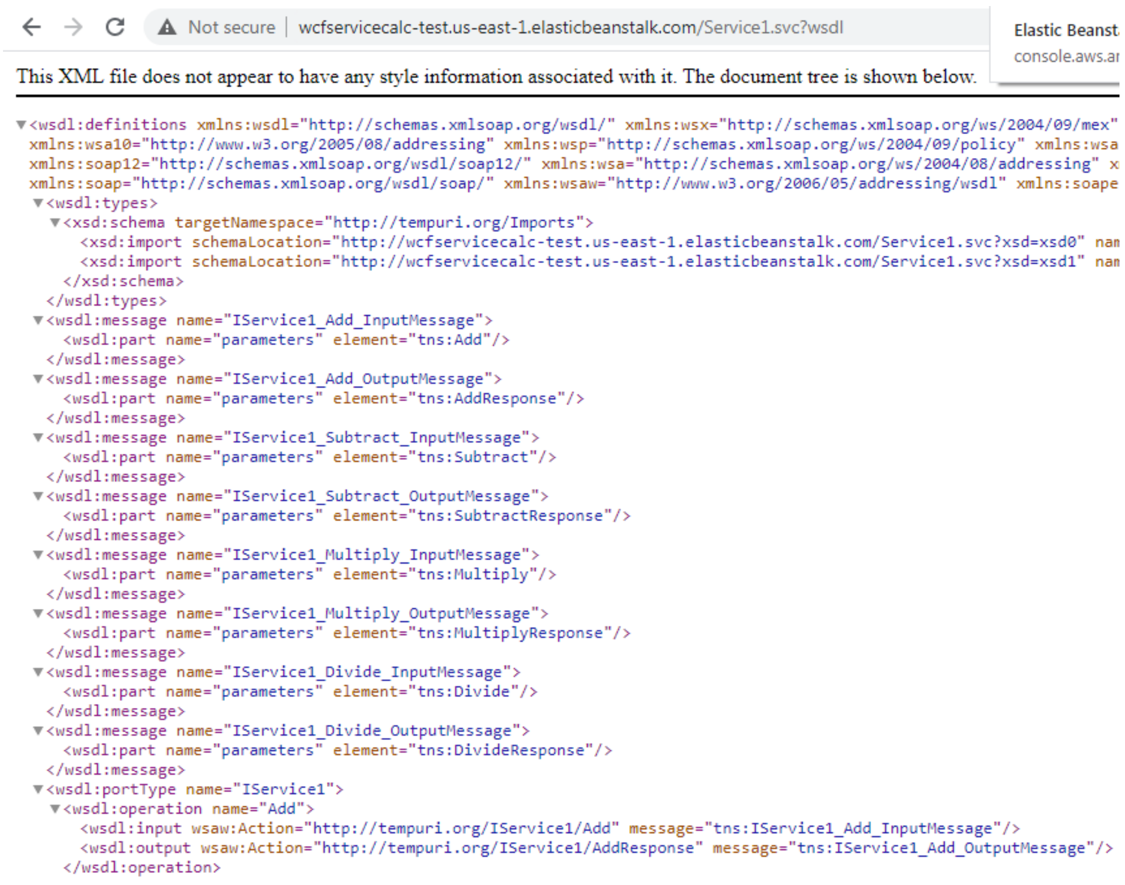


wcfservicecalc-test.us-east-1.elasticbeanstalk.com - /

```
 9/1/2020 10:48 PM        <dir> bin
 9/1/2020 11:07 PM          255 default.html
7/28/2020  6:42 PM          111 Service1.svc
8/31/2020  2:15 AM         1141 WcfServiceCalc.sln
 9/1/2020 10:49 PM         1232 Web.config
```

Test the WCF Web Service deployed at AWS Elastic Beanstalk

1. Check the WSDL description of a Web Service



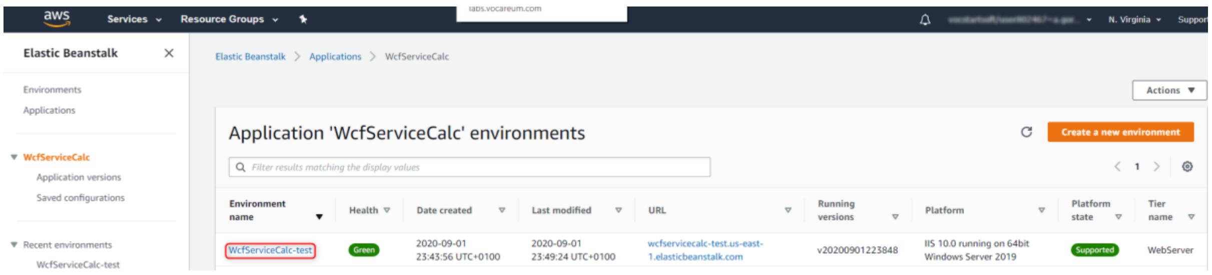
Not secure | wcfservicecalc-test.us-east-1.elasticbeanstalk.com/Service1.svc?wsdl

Elastic Beanst
console.aws.ar

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

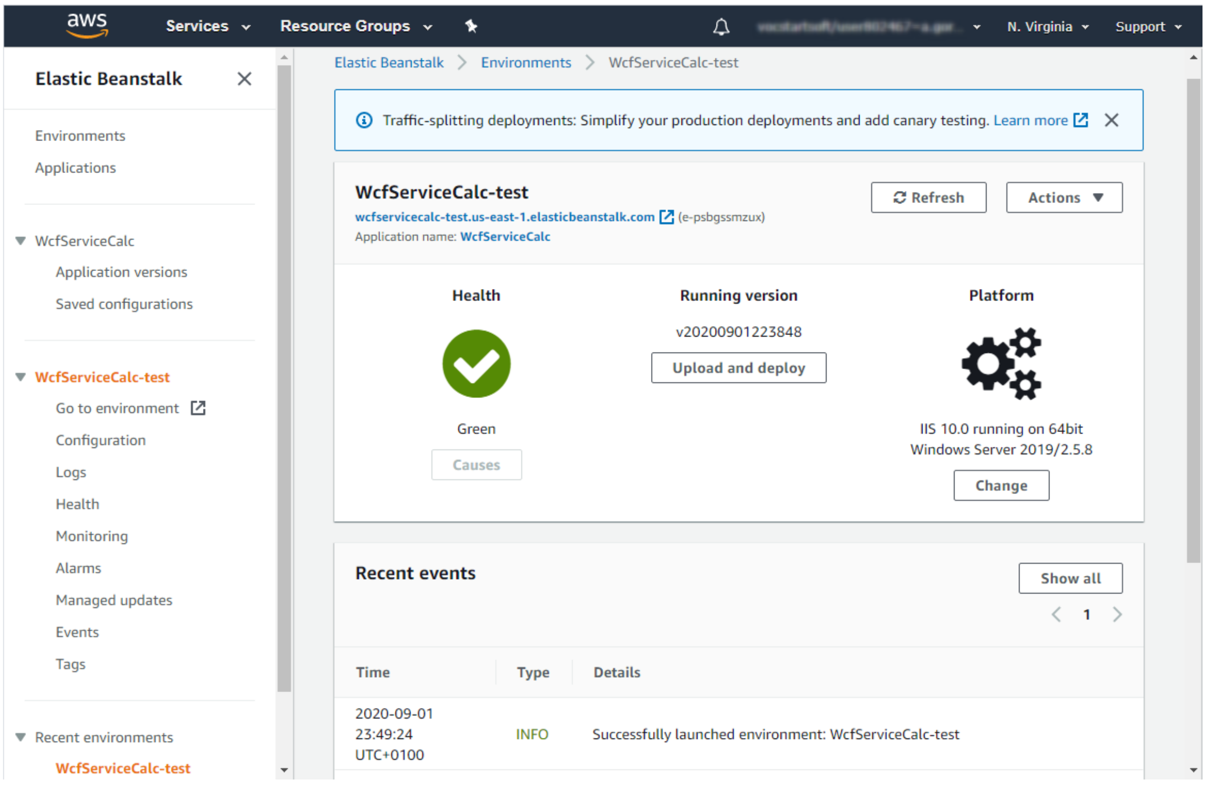
```
▼<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex"
 xmlns:wsa10="http://www.w3.org/2005/08/addressing" xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy" xmlns:wsa
 xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/" xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" x
 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsdl" xmlns:soape
  ▼<wsdl:types>
    ▼<xsd:schema targetNamespace="http://tempuri.org/Imports">
       <xsd:import schemaLocation="http://wcfservicecalc-test.us-east-1.elasticbeanstalk.com/Service1.svc?xsd=xsd0" nam
       <xsd:import schemaLocation="http://wcfservicecalc-test.us-east-1.elasticbeanstalk.com/Service1.svc?xsd=xsd1" nam
     </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:MultiplyResponse"/>
   </wsdl:message>
  ▼<wsdl:message name="IService1_Divide_InputMessage">
     <wsdl:part name="parameters" element="tns:Divide"/>
   </wsdl:message>
  ▼<wsdl:message name="IService1_Divide_OutputMessage">
     <wsdl:part name="parameters" element="tns:DivideResponse"/>
   </wsdl:message>
  ▼<wsdl:portType name="IService1">
    ▼<wsdl:operation name="Add">
       <wsdl:input wsaw:Action="http://tempuri.org/IService1/Add" message="tns:IService1_Add_InputMessage"/>
       <wsdl:output wsaw:Action="http://tempuri.org/IService1/AddResponse" message="tns:IService1_Add_OutputMessage"/>
     </wsdl:operation>
```

2. Use the SoapUI to test your WCF Web Service deployed at AWS Elastic Beanstalk
3. Update the **WcfClientCalc** client application to invoke the WCF Web Service deployed at AWS Elastic Beanstalk
4. Logging into AWS management console and explore how the deployed service can be viewed and managed in the Elastic Beanstalk and EC2 services.



Application 'WcfServiceCalc' environments

Environment name	Health	Date created	Last modified	URL	Running versions	Platform	Platform state	Tier name
WcfServiceCalc-test	Green	2020-09-01 23:43:56 UTC+0100	2020-09-01 23:49:24 UTC+0100	wcfservicecalc-test.us-east-1.elasticbeanstalk.com	v20200901223848	IIS 10.0 running on 64bit Windows Server 2019	Supported	WebServer



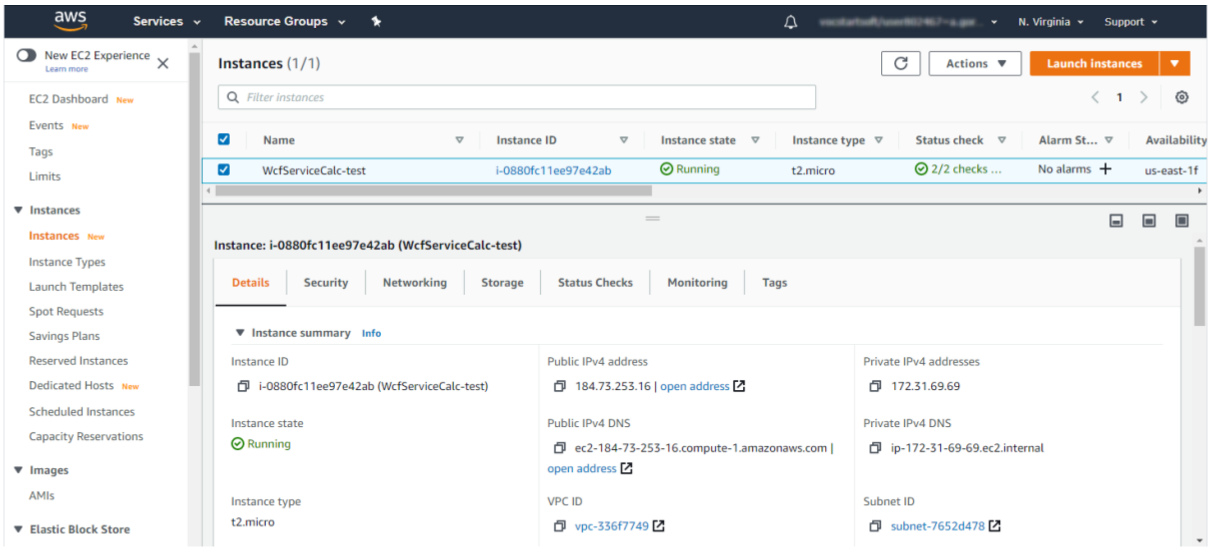
Traffic-splitting deployments: Simplify your production deployments and add canary testing. Learn more

WcfServiceCalc-test
wcfservicecalc-test.us-east-1.elasticbeanstalk.com (e-psbgssmzux)
Application name: WcfServiceCalc

Health	Running version	Platform
Green	v20200901223848	IIS 10.0 running on 64bit Windows Server 2019/2.5.8

Recent events

Time	Type	Details
2020-09-01 23:49:24 UTC+0100	INFO	Successfully launched environment: WcfServiceCalc-test



Instances (1/1)

Name	Instance ID	Instance state	Instance type	Status check	Alarm St...	Availability
WcfServiceCalc-test	i-0880fc11ee97e42ab	Running	t2.micro	2/2 checks ...	No alarms	us-east-1f

Instance: i-0880fc11ee97e42ab (WcfServiceCalc-test)

Instance summary Info

Instance ID	Public IPv4 address	Private IPv4 addresses
i-0880fc11ee97e42ab (WcfServiceCalc-test)	184.73.253.16 | open address	172.31.69.69
Instance state	**Public IPv4 DNS**	**Private IPv4 DNS**
Running	ec2-184-73-253-16.compute-1.amazonaws.com | open address	ip-172-31-69-69.ec2.internal
Instance type	**VPC ID**	**Subnet ID**
t2.micro	vpc-336f7749	subnet-7652d478

If you wish, you can connect to the Windows virtual instance running your WCF web service via the RDP.

Step 4: Deleting AWS Elastic Beanstalk resources after use

Log into AWS management console and delete created AWS Elastic Beanstalk environments and applications. Alternatively, you can delete it using AWS Explore in Visual Studio.