**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 3](#_heading=h.30j0zll)

[Step 1: Running Visual Studio 2007 3](#_heading=h.1fob9te)

[Step 2: Exploring C# WCF Web Service 4](#_heading=h.3znysh7)

[Exploring the Service1.svc.cs 4](#_heading=h.2et92p0)

[Exploring the IService1.cs 5](#_heading=h.tyjcwt)

[Launching and testing the WCF Web Service 5](#_heading=h.3dy6vkm)

[Step 3: Exploring and executing the WCF Web Service Client 8](#_heading=h.1t3h5sf)

[Step 4: Launching a Web Service on AWS Elastic Beanstalk PaaS 9](#_heading=h.4d34og8)

[Launching a Web Service on the AWS Elastic Beanstalk PaaS 9](#_heading=h.2s8eyo1)

[Test the WCF Web Service deployed at AWS Elastic Beanstalk 15](#_heading=h.17dp8vu)

[Step 4: Deleting AWS Elastic Beanstalk resources after use 17](#_heading=h.3rdcrjn)

# 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...

Graphical user interface, application

Description automatically generated  
Graphical user interface, text, application

Description automatically generated

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

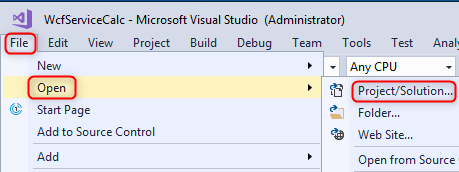
Graphical user interface, application, Word

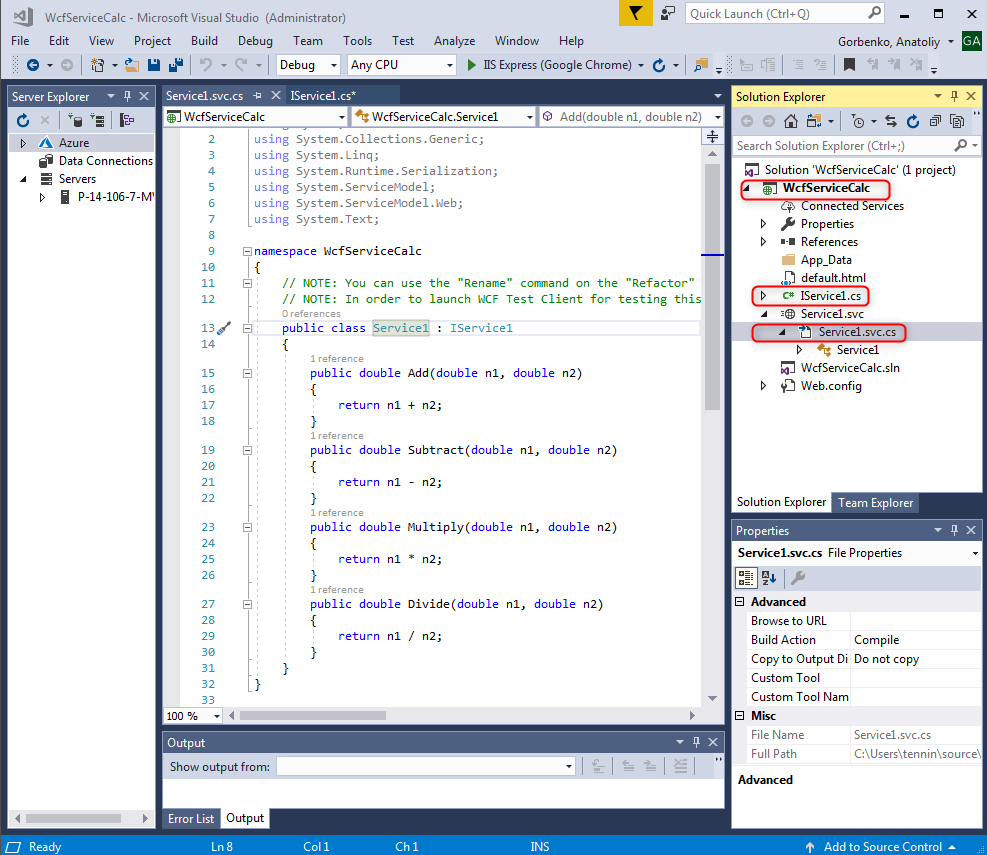
Description automatically generated

# 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*.





# Step 2: Exploring C# WCF Web Service

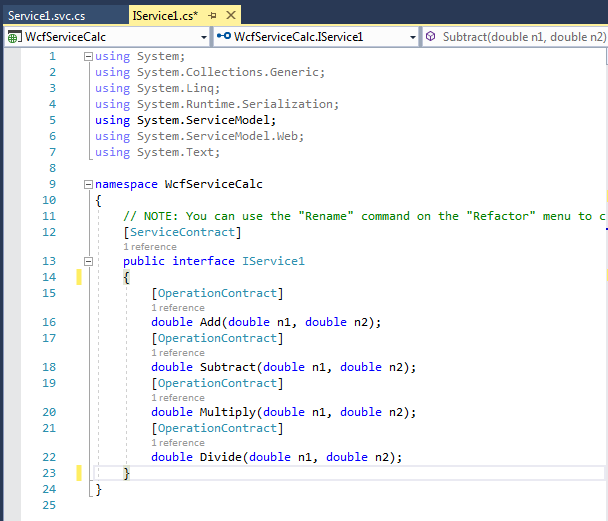
**Windows Communication Foundation (WCF)** is a framework for building service-oriented applications. Using WCF, you can send data as asynchronous messages from one service endpoint to another. A service endpoint can be part of a continuously available service hosted by IIS (Internet Information Server), 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.



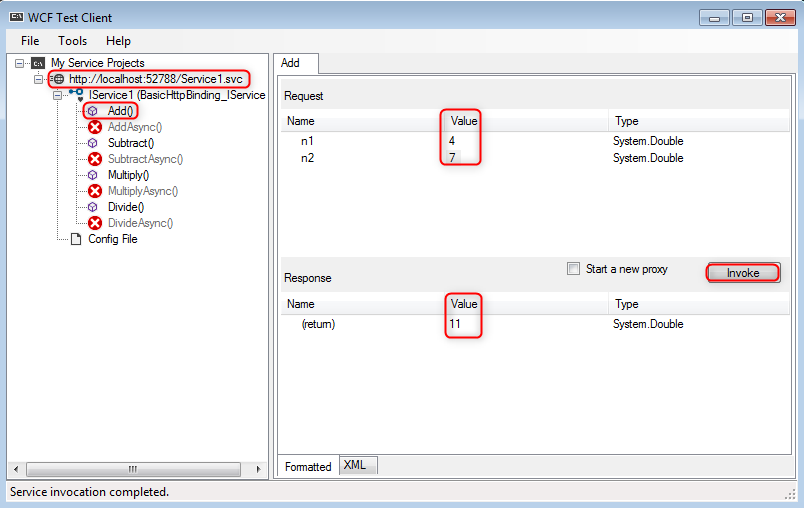
## Launching and testing the WCF Web Service

1. In order to launch WCF Web Service for testing this service, please select *Service1.svc.cs* at the Solution Explorer and start debugging.

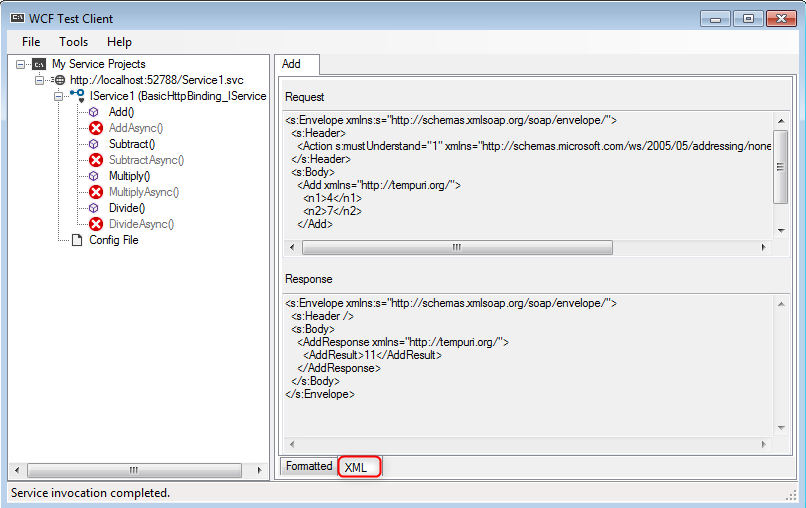
Visual Studio includes IIS Express, a lightweight, self-contained version of IIS optimized for developers. It makes it easy to develop and test web project.

1. 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.



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

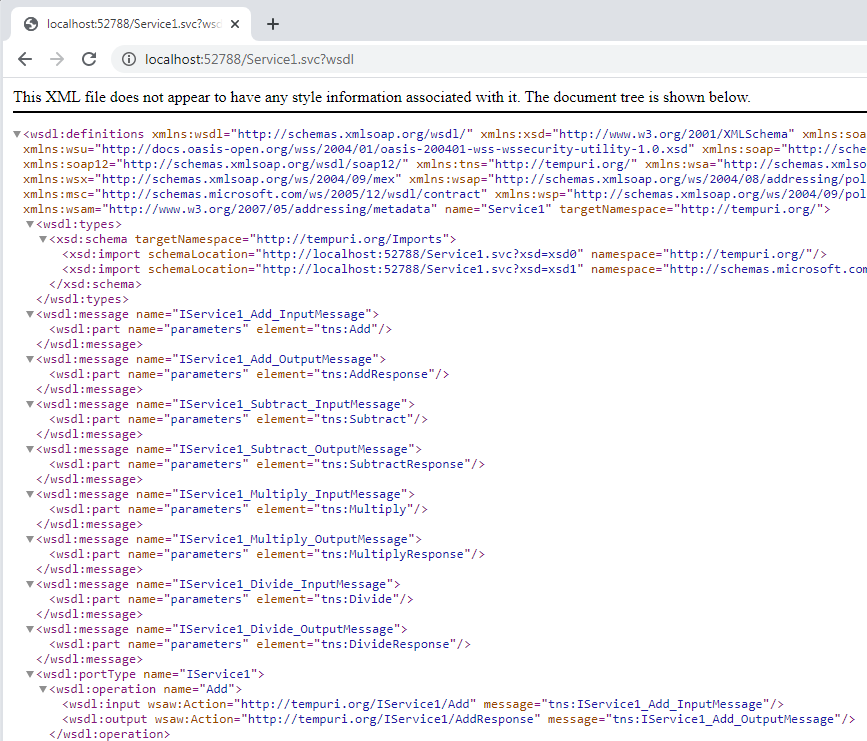


1. Run a web browser and check WSDL description of the web service at: <http://localhost:52788/Service1.svc?wsdl> (the port number in your case can be different from 52788 as in the above example; please, notice the right port number from the *WCF Test Client* window as shown below:   
   Graphical user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

   Description automatically generated

Graphical user interface, text, application

Description automatically generated

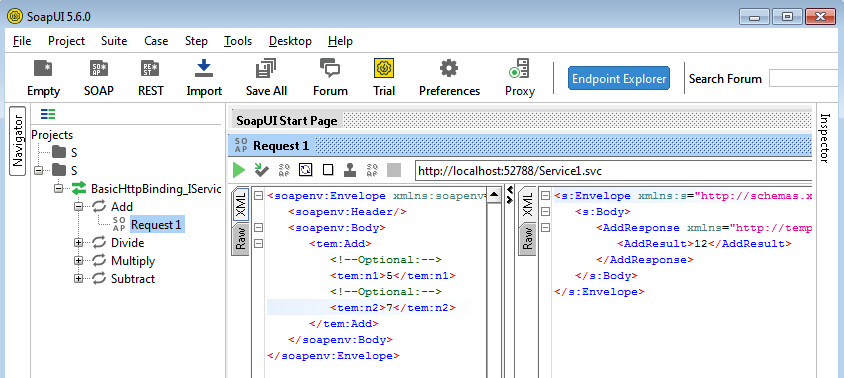


1. You can also test the web service using SoapUI tool. Run the *SoapUI* on your PC and create a new SOAP Project. Provide the URL to the wsdl file, e.g. <http://localhost:52788/Service1.svc?wsdl>:

Graphical user interface, text, application, email

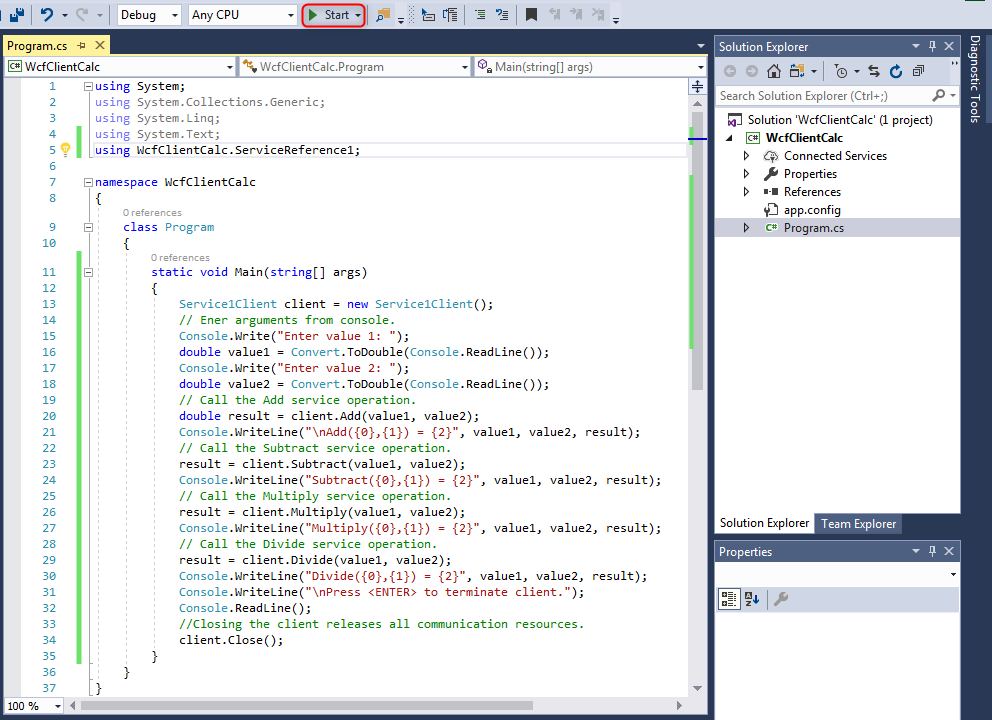
Description automatically generated

1. 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.

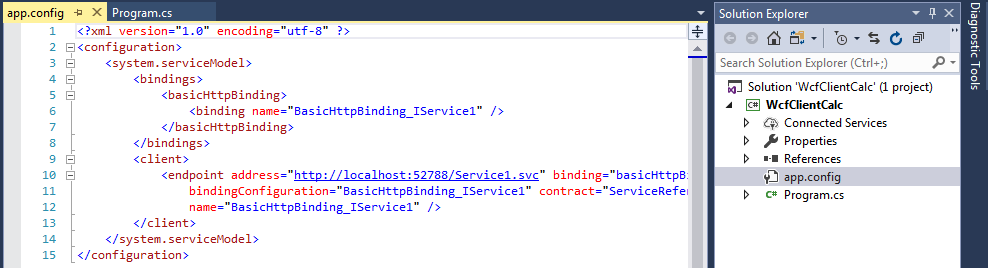


# Step 3: Exploring and executing the WCF Web Service Client

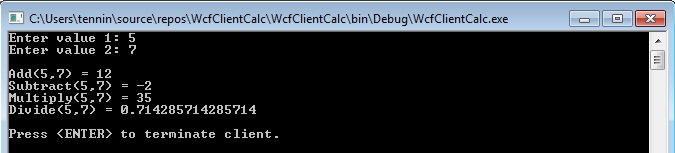
1. Open a new copy of Visual Studio (make sure the first Visual Studio still executes the WcfServerCalc project).
2. Open the **WcfClientCalc** project in the solution Explorer panel (open from C:\Users\tennin\source\repos) and open the *Program.cs* file. The **WcfClientCalc** is a desktop application which invokes different operations of the **WcfServerCalc** service.



1. Explore the client’s program code in **Program.cs** and notice the way a web service is invoked
2. Check the **app.config** which specifies the target Web Service. Notice the URL address of the service. Change it if needed.



1. Execute (run) the client.



1. 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.

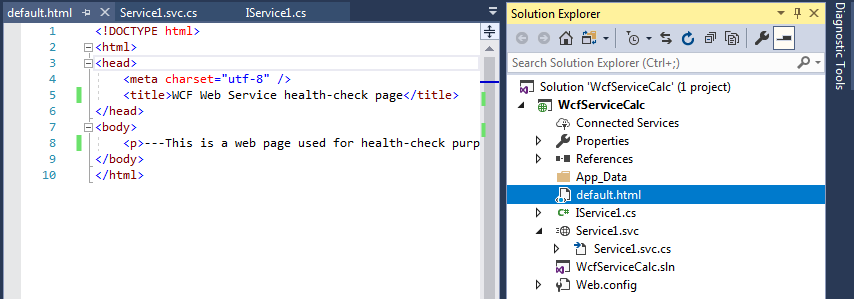
You can simply upload your code and Elastic Beanstalk automatically handles the deployment, from capacity provisioning, load balancing, auto-scaling to application health monitoring. At the same time, you retain full control over the AWS resources powering your application and can access the underlying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



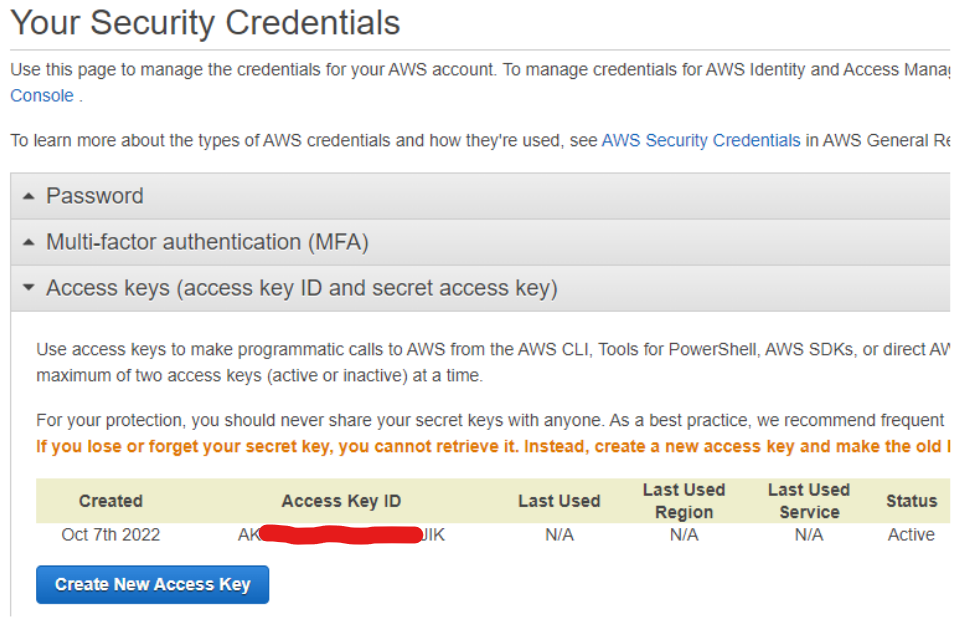
1. Configure the AWS default profile on Visual Studio to get access to your AWS account
   1. Log in to your AWS account <https://aws.amazon.com/> and select/search for IAM (Identity and Access Management) centre:

Graphical user interface, application, website

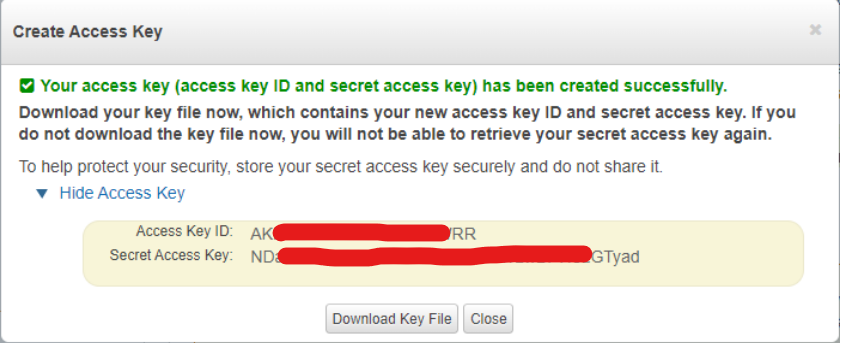
Description automatically generated

Click on ‘Manage access keys’ and create a New Access Key   
A screenshot of a computer

Description automatically generated



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.



|  |  |
| --- | --- |

* 1. Copy and paste your credentials into USER\_HOME/.aws/credentials and save the file

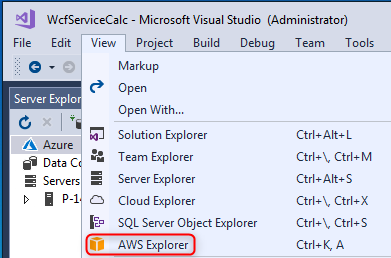
1. Open the AWS Toolkit on Visual Studio and copy/paste your Access Key and Secret Key:

Graphical user interface, text, application, Word

Description automatically generated  
Graphical user interface, text, application, email

Description automatically generated

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

Description automatically generated

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:

Graphical user interface, text, application, email

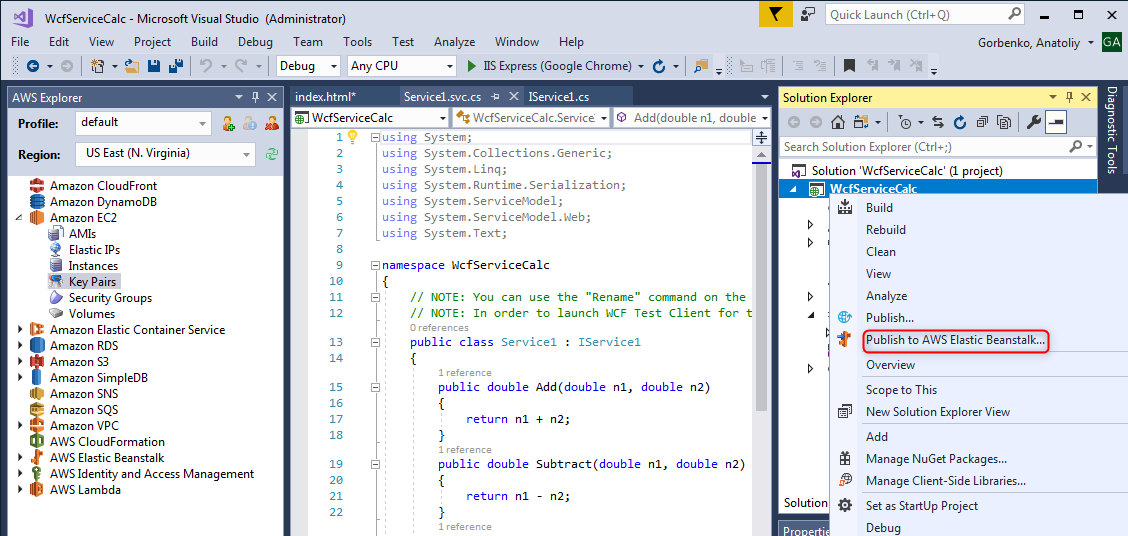
Description automatically generated  
Graphical user interface, text, application, email

Description automatically generated  
You can filter images by the type of operating system:

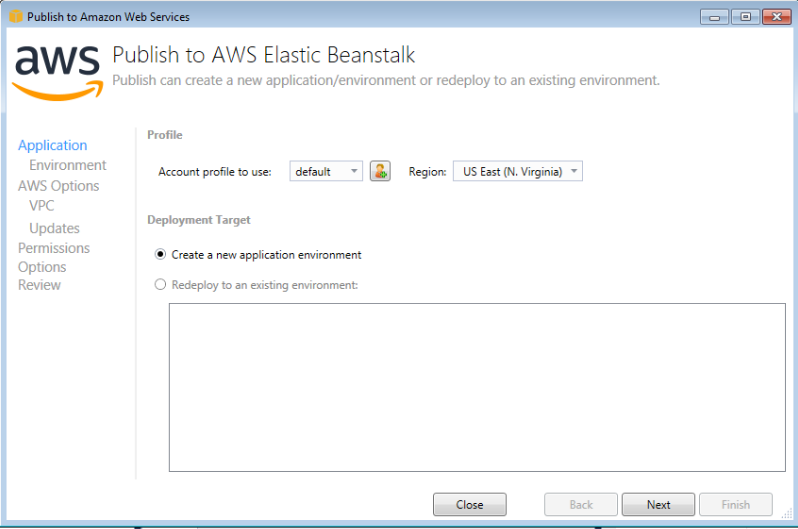
Graphical user interface, text

Description automatically generated

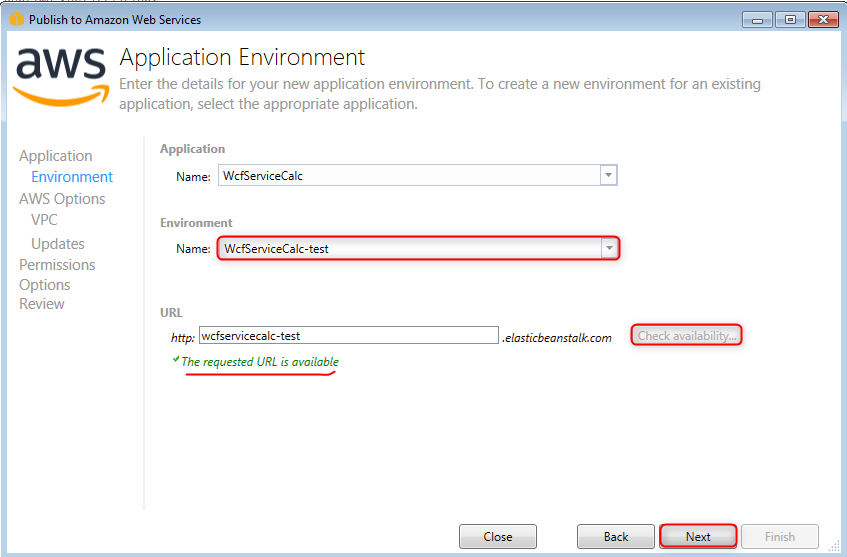
1. Now. Let us deploy your Calculator Service to AWS Elastic Beanstalk Paas. Right Click on the **WcfServiceCalc** project and select **Publish to AWS Elastic Beanstalk**



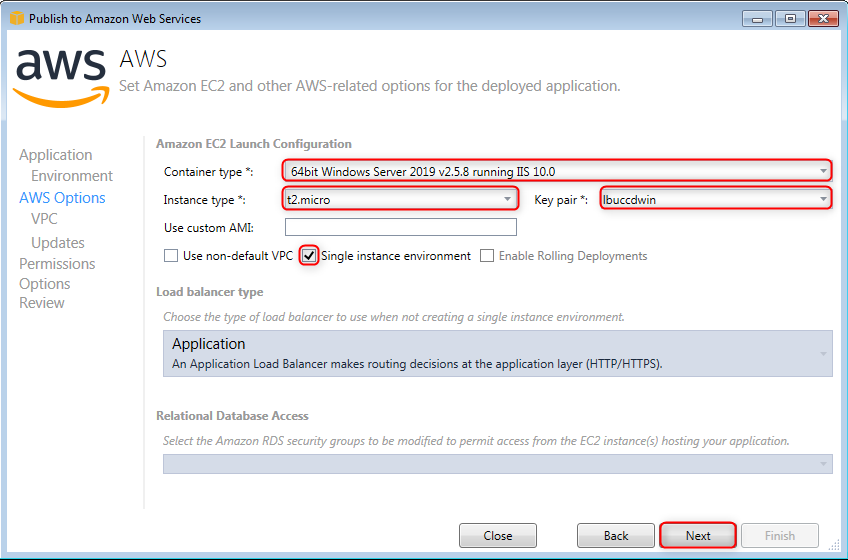
1. Now you will be guided through the AWS Elastic Beanstalk publish process
   1. Create a new application environment using the **default** account profile. You can change the region if you wish.



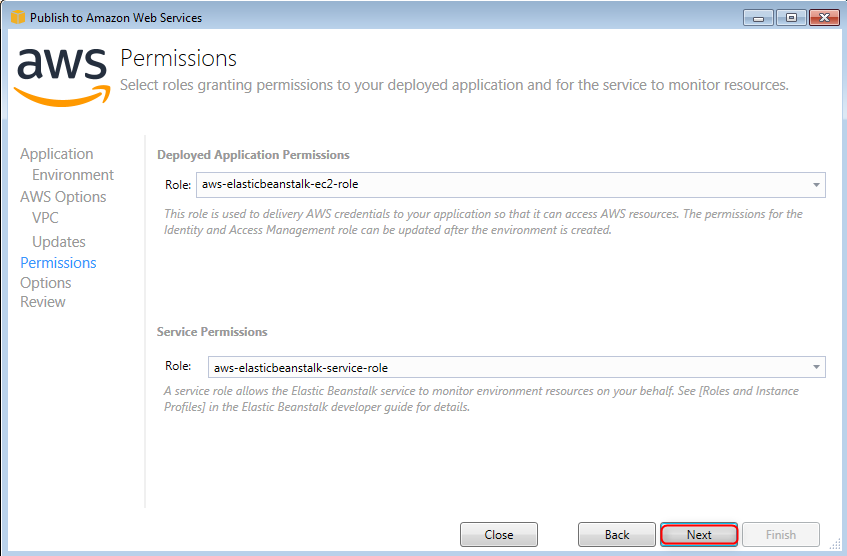
* 1. Specify the environment name, URL and check its availability. **Note:** the default environment name can be already taken by your classmates. So, include your student id as a part of the name.



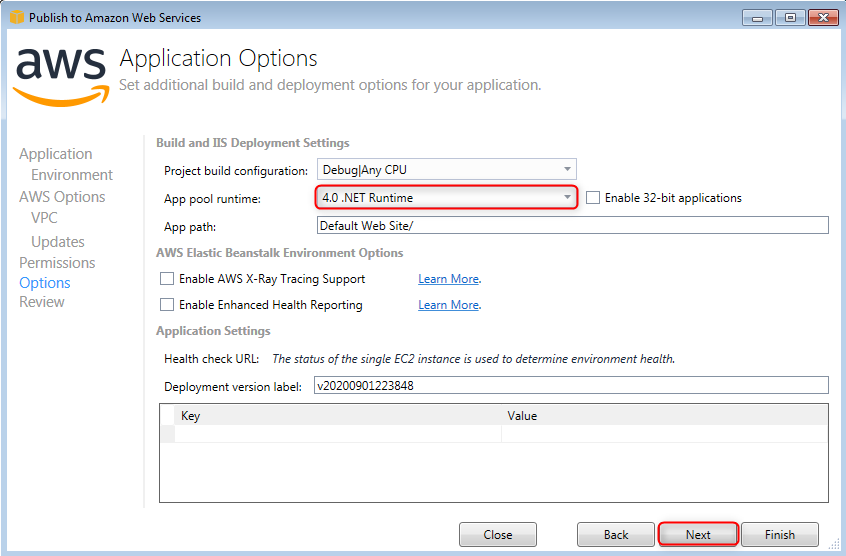
* 1. Configure the AWS options
     1. On the AWS Options page, in Amazon EC2 Launch Configuration, from the Container type drop-down list, choose an Amazon Machine Image (AMI) type that will be used for your application;
     2. In the Instance type drop-down list, specify an Amazon EC2 instance type to use (micro instance will minimize the cost associated with running the instance);
     3. 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;
     4. select the Single instance environment box to deploy your application on a single Amazon EC2 instance



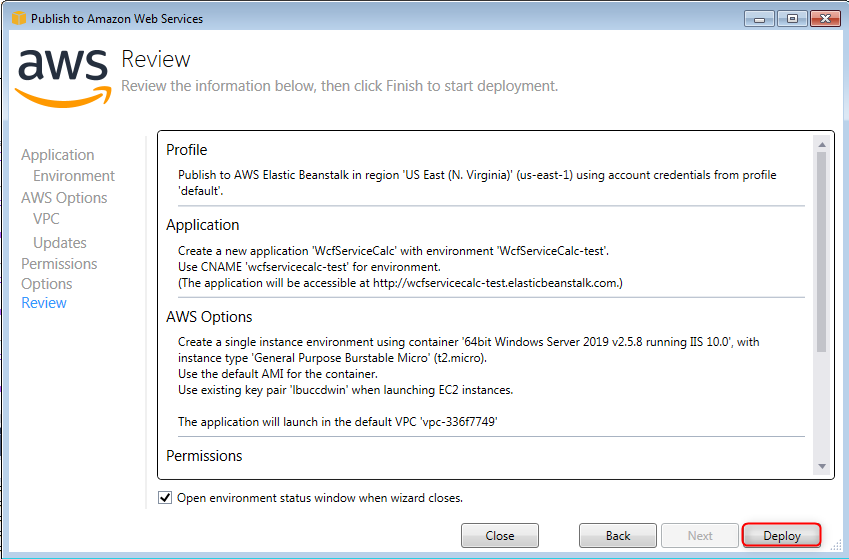
* 1. Leave default permissions for your instance; you can configure them latter on if you wish



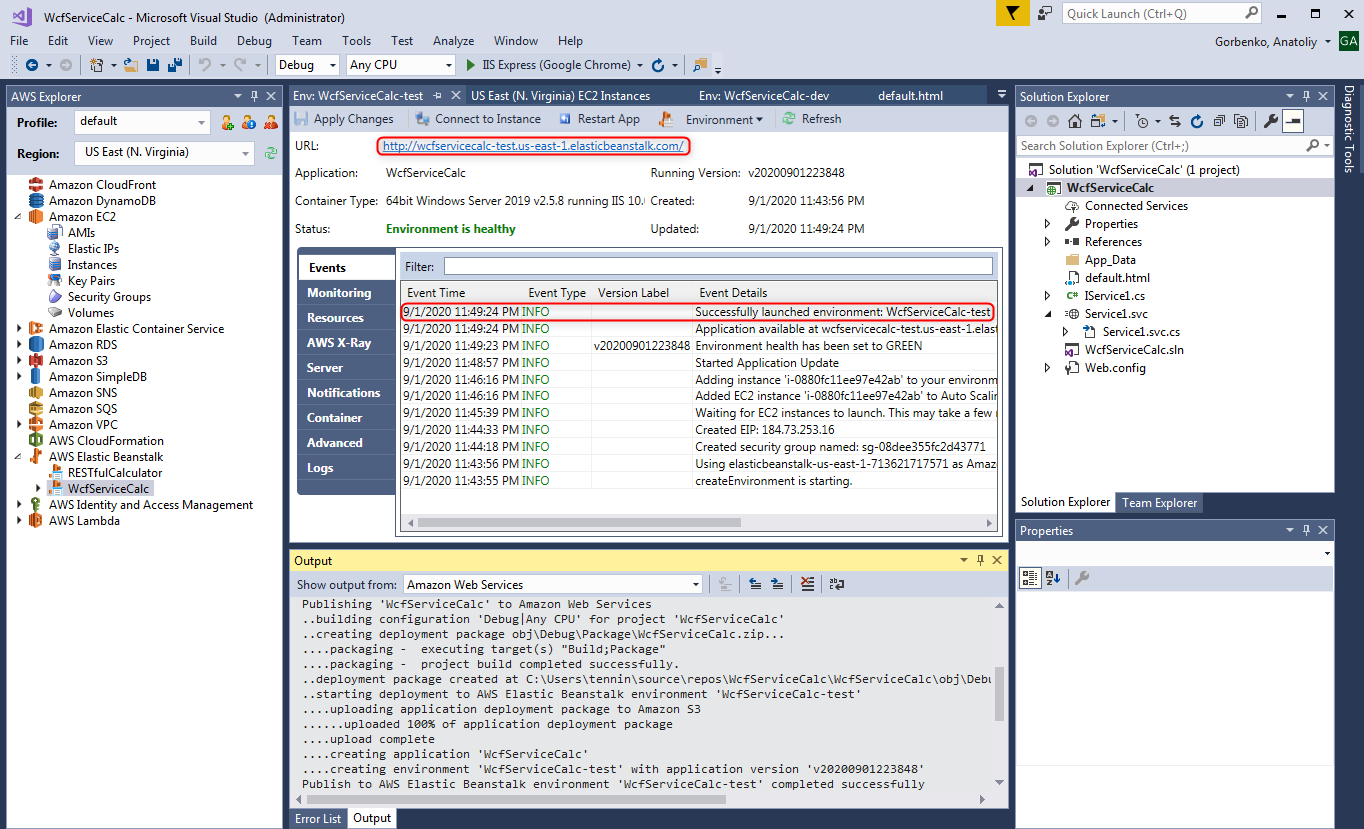
* 1. Make sure you use the same version of the .NET Runtime environment



* 1. Deploy your project



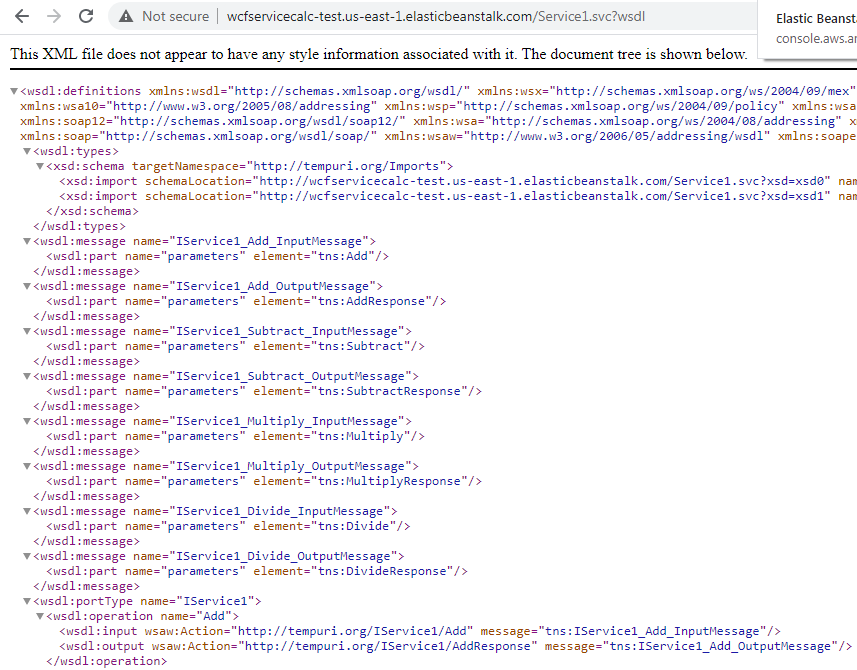
* 1. Wait for the instance being launched and the project being deployed and open the web service URL



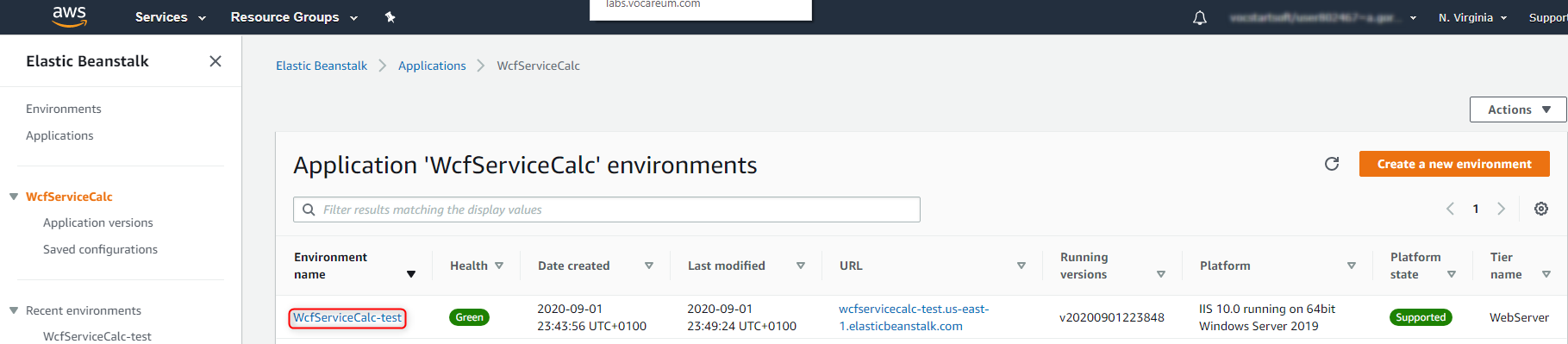


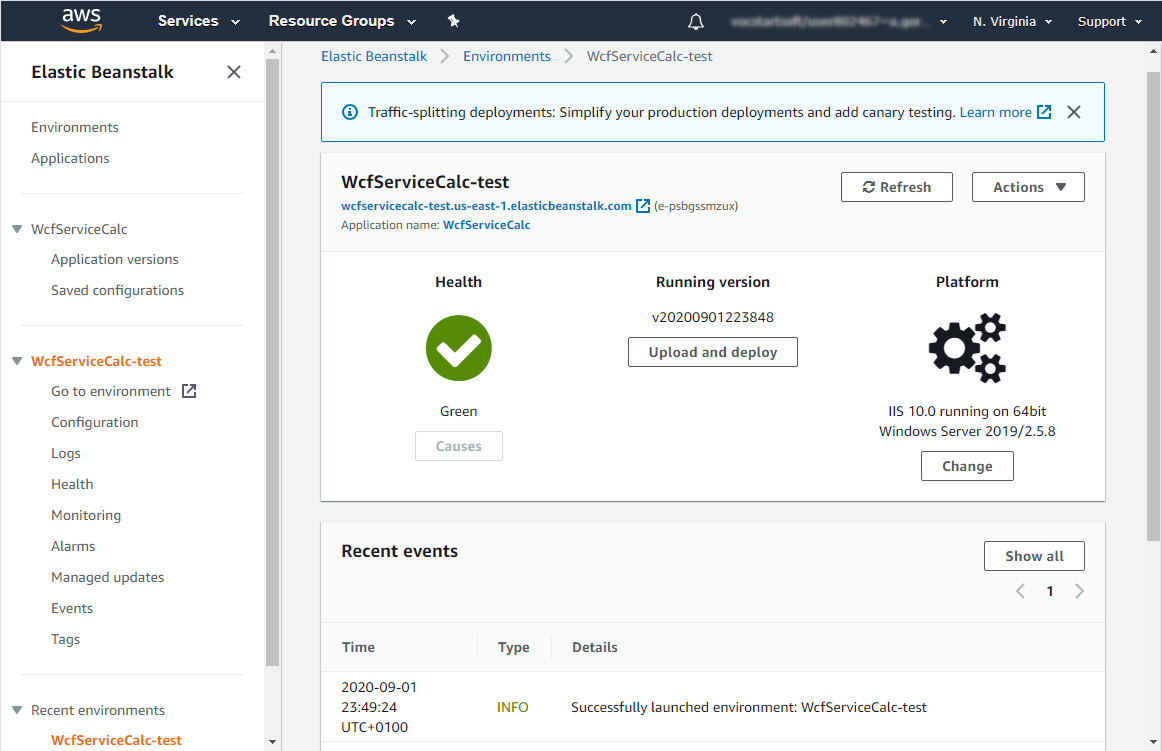
## Test the WCF Web Service deployed at AWS Elastic Beanstalk

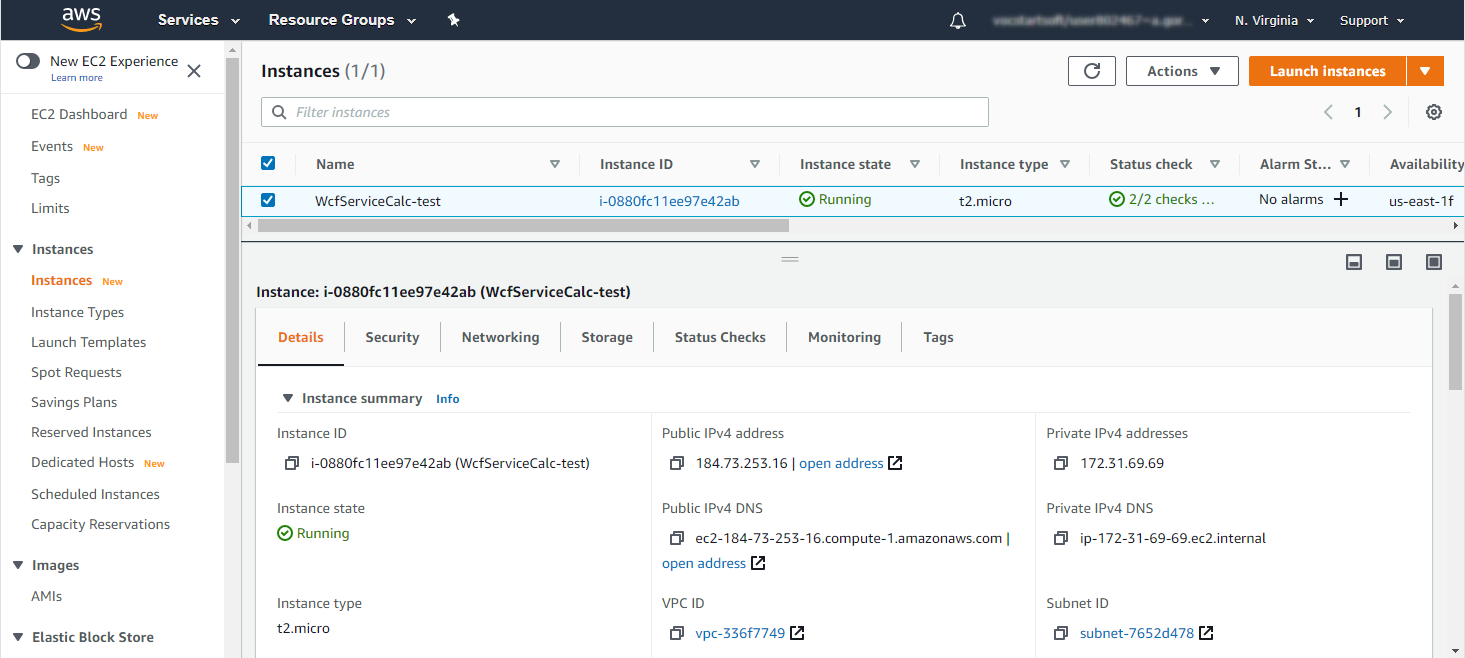
1. Check the WSDL description of a Web Service



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







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.