PaaS – Deployment of C# WCF web service with Azure PaaS

**Objective:** Get practical experience in C# WCF Web services development and deployment on Cloud PaaS with Microsoft Azure App Services.

**Tasks:**

1. Create and manage Microsoft Azure App Service environment (PaaS)
2. Deploy a C# WCF Web Service on Cloud PaaS with Microsoft Azure App Services
3. Test Azure PaaS Web Service using SoapUI tool
4. Test Azure PaaS Web Service using C# client
5. Develop and deploy your own C# WCF Web Service and Client

**Lab environment**:

* Visual Studio Enterprise 2007 with the following workloads:
  + ASP.NET and web development
  + Microsoft Azure SDK for .NET
* SoapUI tool

*Contents*

[Step 0: Register for Microsoft Azure Free Student Account](#_heading=h.1fob9te) 3

[Step 1: Create a new App Service Environment (PaaS) on Windows Azure](#_heading=h.qfufu0u0v30f) 4

[Create an App Service](#_heading=h.g2fru06pva2m) 4

[Manage the Azure web app](#_heading=h.tyjcwt) 7

[Step 2: Deploy C# WCF service on Windows Azure](#_heading=h.3dy6vkm) 11

[Step 3: Test Azure WCF Service using SoapUI](#_heading=h.1t3h5sf) 15

[Step 4: Test Azure WCF Service using desktop client](#_heading=h.4d34og8) 15

[Step 5. Clean up resources](#_heading=h.2s8eyo1) 16

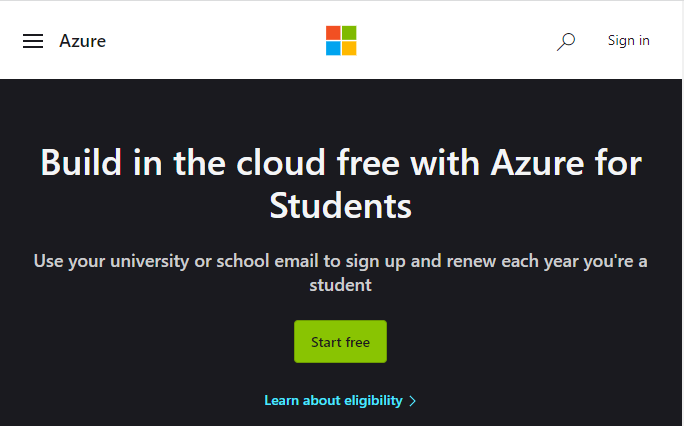
[Advanced assignment](#_heading=h.17dp8vu) 16

[References](#_heading=h.3rdcrjn) **16**

# Step 0: Register for Microsoft Azure Free Student Account

If you have not done so, register for Microsoft Azure Free Student Account at <https://azure.microsoft.com/en-gb/free/students/>. Use your UNIVERSITY EMAIL ACCOUNT.

**P.S. Do not forget to delete all created resources after you complete the labsheet!   
Otherwise, you can run out of your free budget!**

****

# 

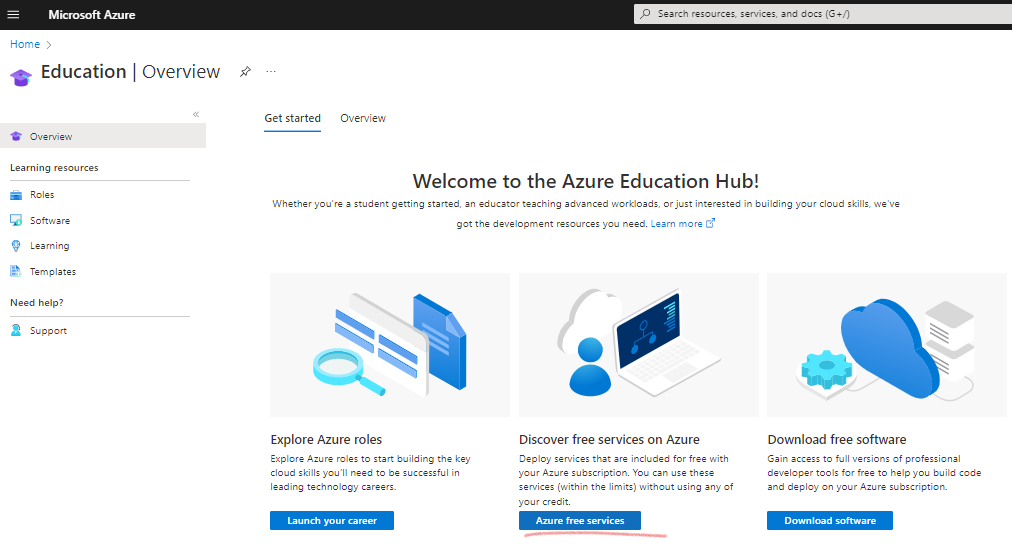
# 

# Step 1: Create a new App Service Environment (PaaS) on Windows Azure

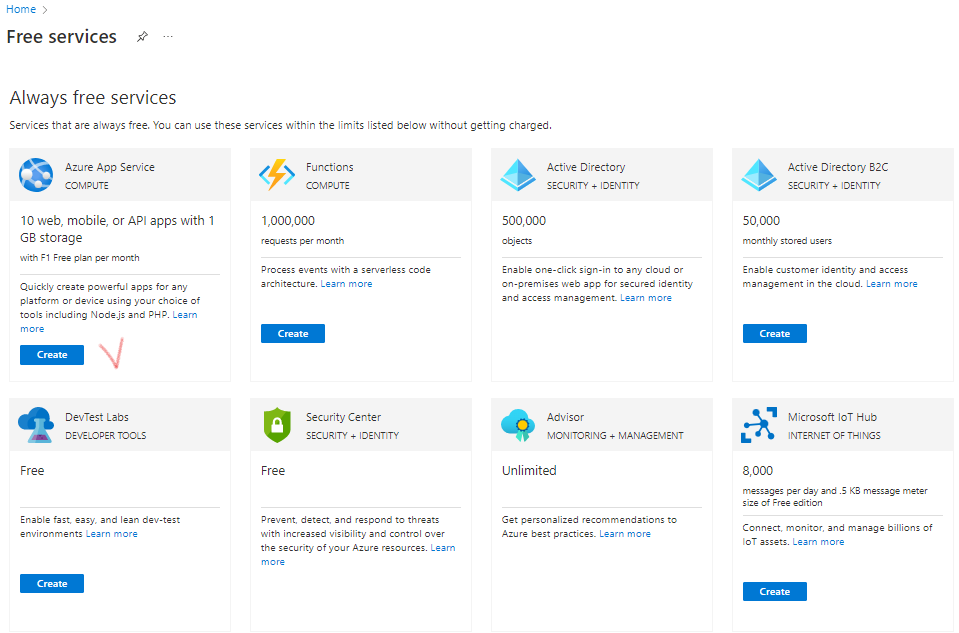
**Tips:** Please note that the actual appearance may differ slightly from the screenshots below due to frequent changes to the Microsoft Azure GUI.

## **Create an App Service**

Open a web browser and navigate to Microsoft Azure Portal via <https://azure.microsoft.com/en-gb/free/students/>. Select ‘Get Started’ and navigate to Azure Free Services.



Scroll down and click on ‘Azure App Services -> Create’.



Select ‘Azure for Students’ subscription.

Specify the name of your instance, e.g. **WcfServiceCalc**. Azure will automatically check availability of the DNS name. This name should be unique (you can use your student ID as a part of the name).

Azure will also automatically create a new **Resource group** for you based of the instance name.

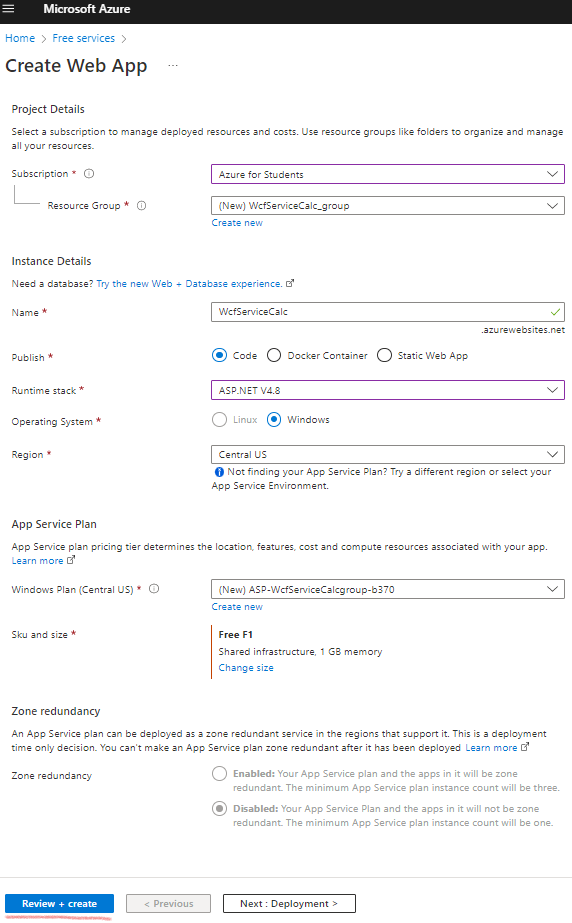
A [Resource group](https://docs.microsoft.com/en-us/azure/azure-resource-manager/resource-group-overview#terminology) is a logical container into which Azure resources like web apps, databases, and storage accounts are deployed and managed.

Select **ASP.NET 4.8** as the **runtime stack**.

You can select the **region** of your choice, or use the default.

Next, you can change the **App Service Plan** which defines the ‘size’ of a virtual instance (number of CPU cores, RAM, Disk space) depending on your goal and price.

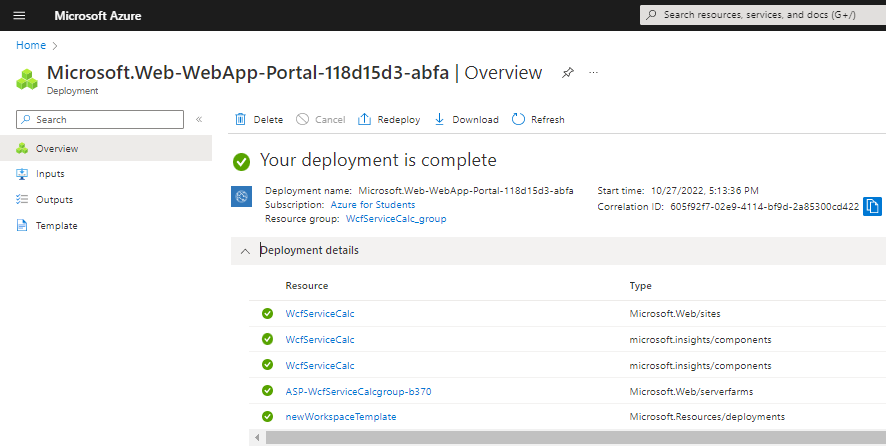
The only one App Service Plan is available for the student subscription for free. If you use another subscription, e.g. pay-as-you-go, you can see more options available.



Click on ‘**Review and Create**’

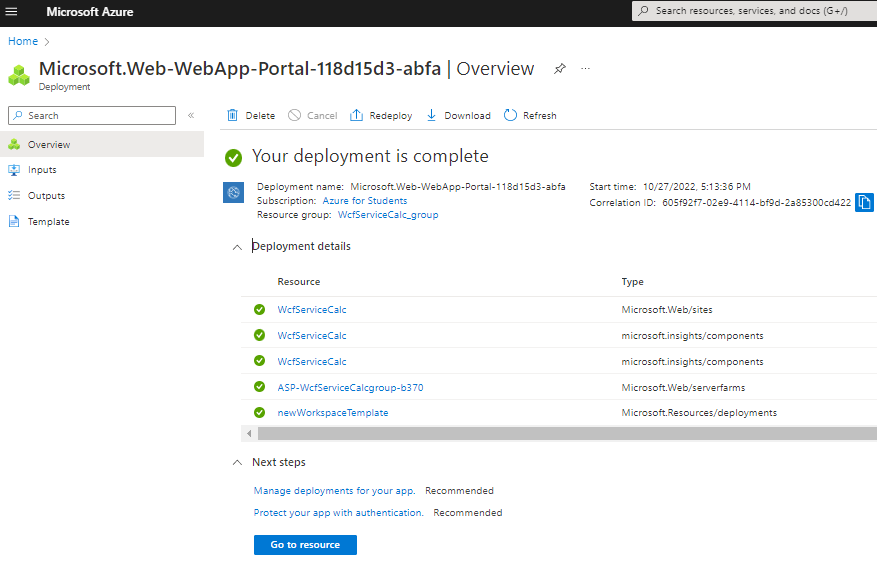
If you need, you can set up advanced deployment, networking and monitoring options by clicking on ‘**Next: Deployment**’. For example, at the deployment tab you can Enable **GitHub Actions** to continuously deploy your app. GitHub Actions is an automation framework that can build, test, and deploy your app whenever a new commit is made in your repository. If your code is in GitHub, choose your repository here and we will add a workflow file to automatically deploy your app to App Service. If your code is not in GitHub, go to the Deployment Center once the web app is created to set up your deployment.

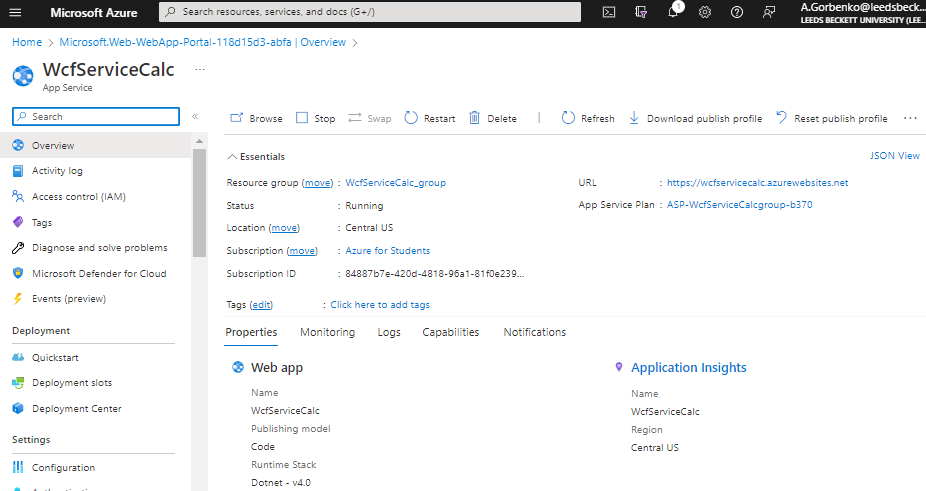
Wait until the application environment is deployed.



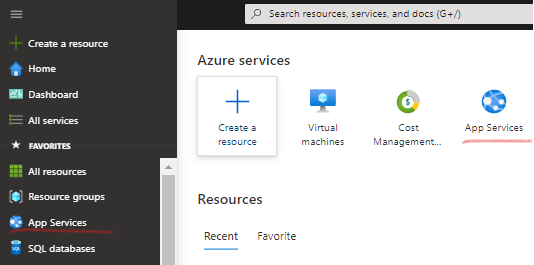
## **Manage the Azure web app**

Click on ‘Go to resource’.

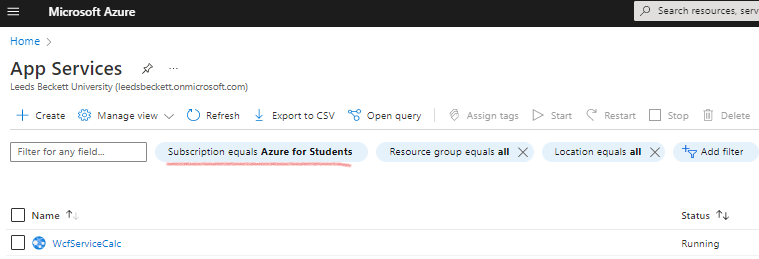




Alternatively, you can find your created service environment through the App Services option from the Home menu.



If you cannot see your app service, make sure you select the right subscription plan:

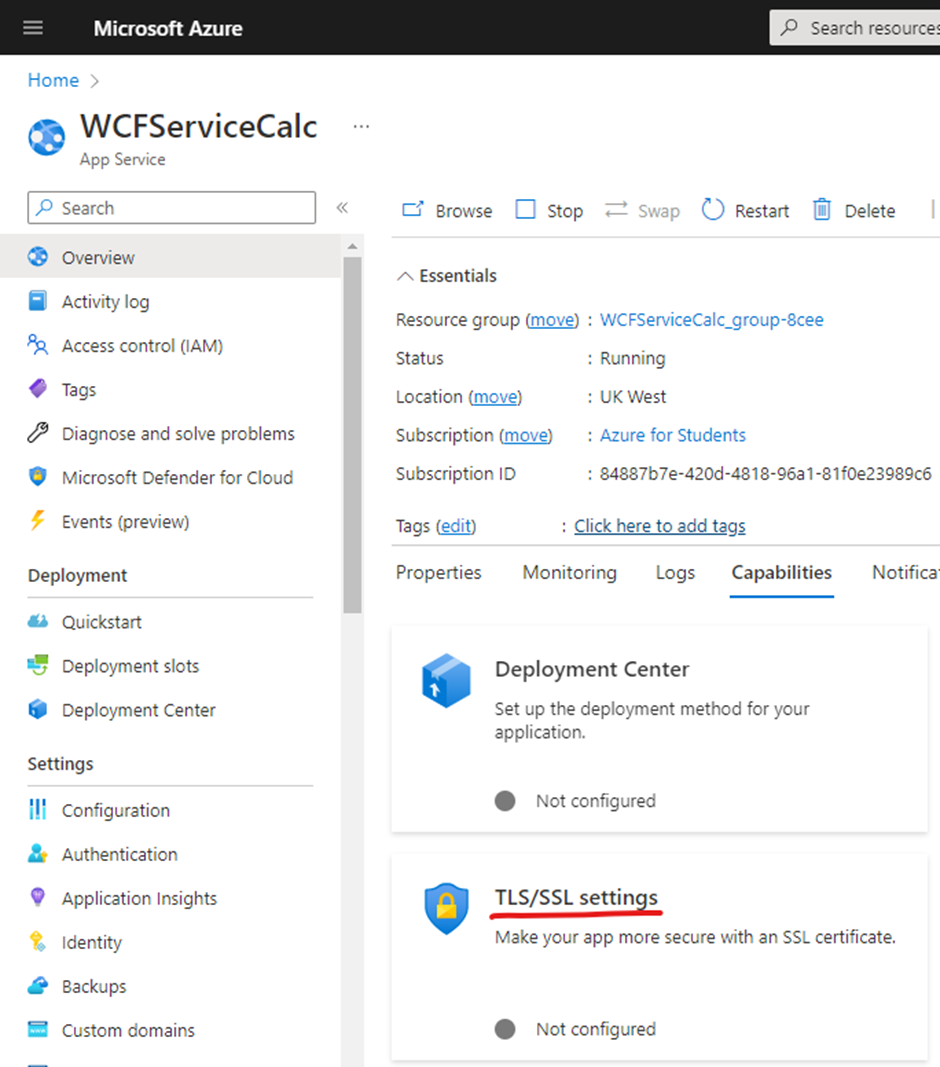


From here you can perform basic management tasks like browse, stop, start, restart, and delete the App Service environment or the deployed application. The left menu provides different pages for configuring your app.

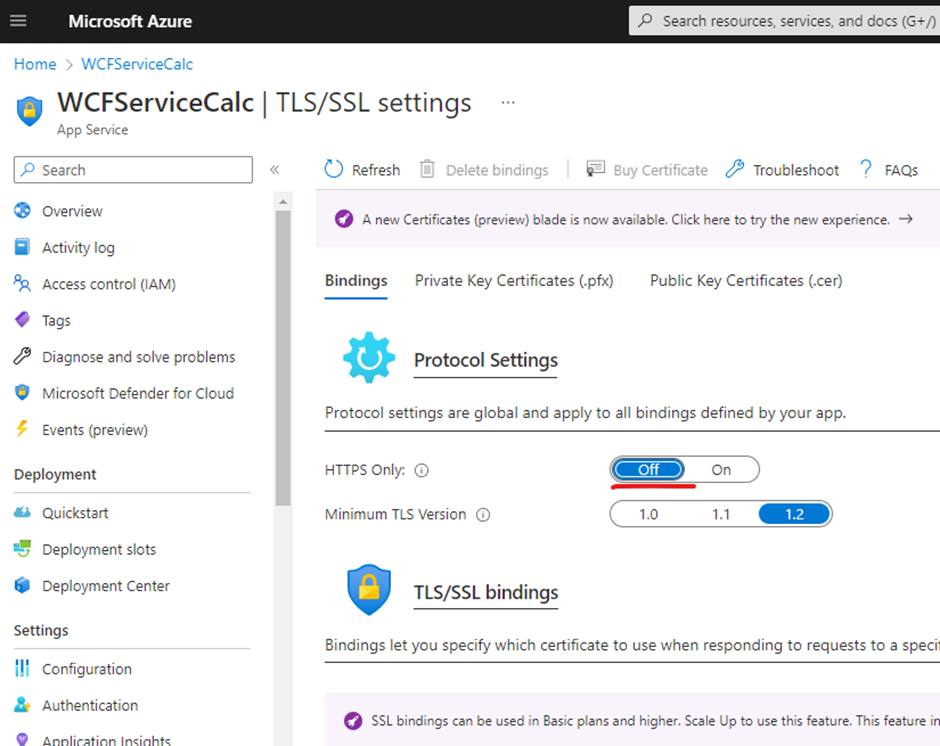
## **Configure HTTP/HTTPS access**

We are going to make our WcfServiceCalc application to be accessible via HTTP protocol. By default, Microsoft Azure converts all http requests to https and forwards them from port 80 to port 443. To change this and enable unencrypted access to our service we need to set ‘HTTPS only’ parameter to ‘Off’.

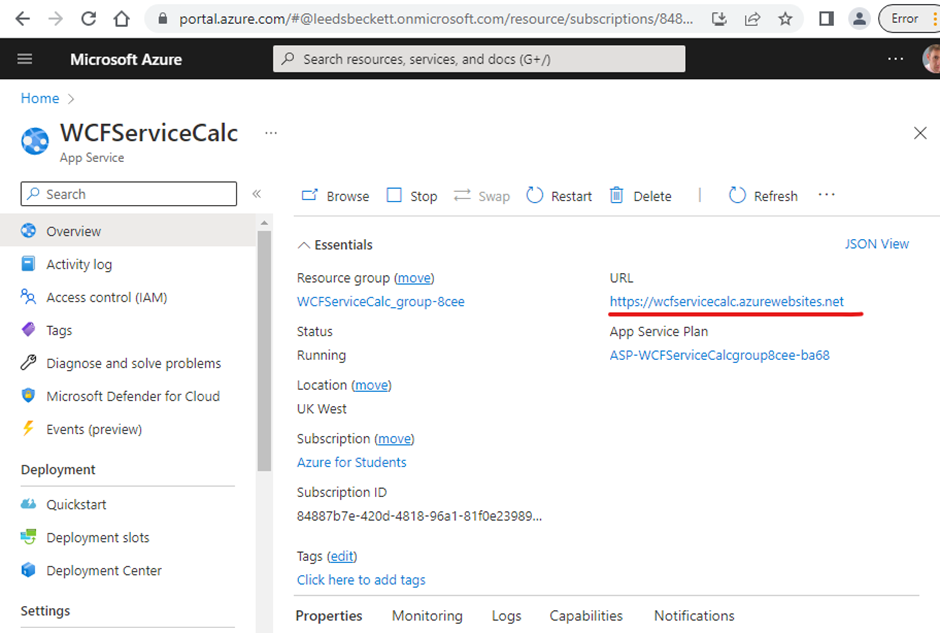
To do this, open the ’Capabilities’ tab of the created ‘WcfServiceCalc’ App Service and choose ’TL/SSL settings’.

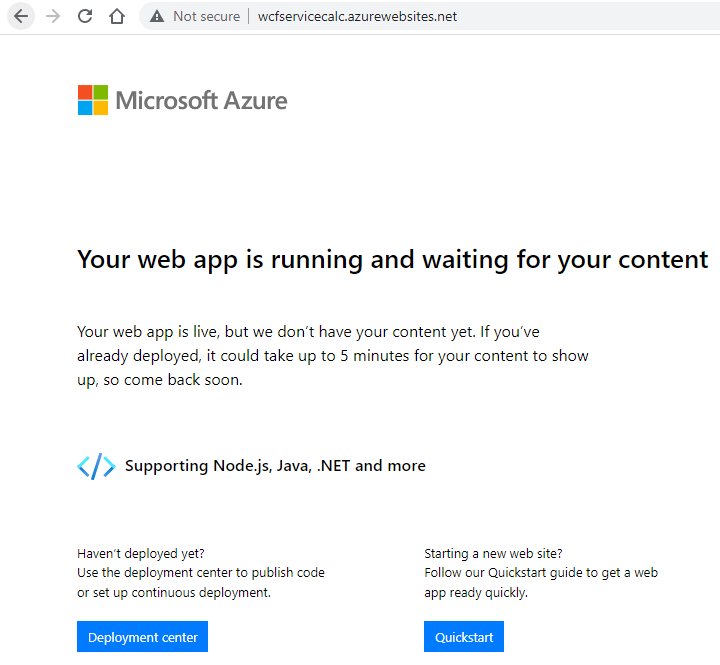


There you can switch off the ‘HTTPS only’ option:



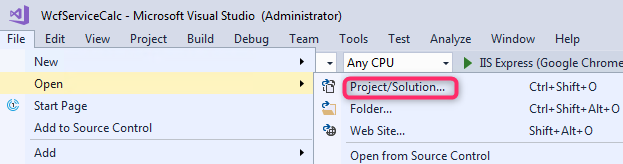
Now visit your app service using the provided URL to make sure it is running and waiting for your content. Try both https:// and http:// prefixes to check that you successfully enabled HTTP protocol.

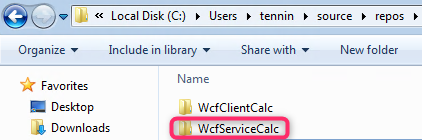




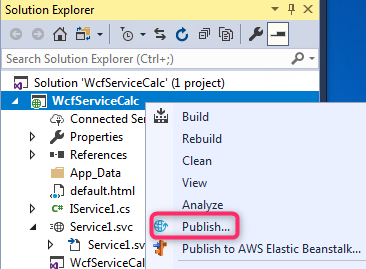
# Step 2: Deploy C# WCF service on Windows Azure

Run Visual Studio and open the **WcfServiceCalc** project from the source code repository, which was discussed in one of the previous labs.



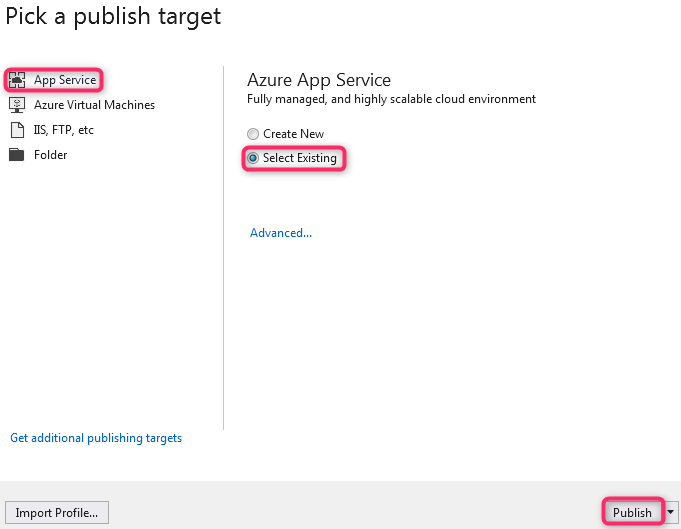


Right click on the WcfServiceCalc project name in the Solution Explorer and choose **Publish…**

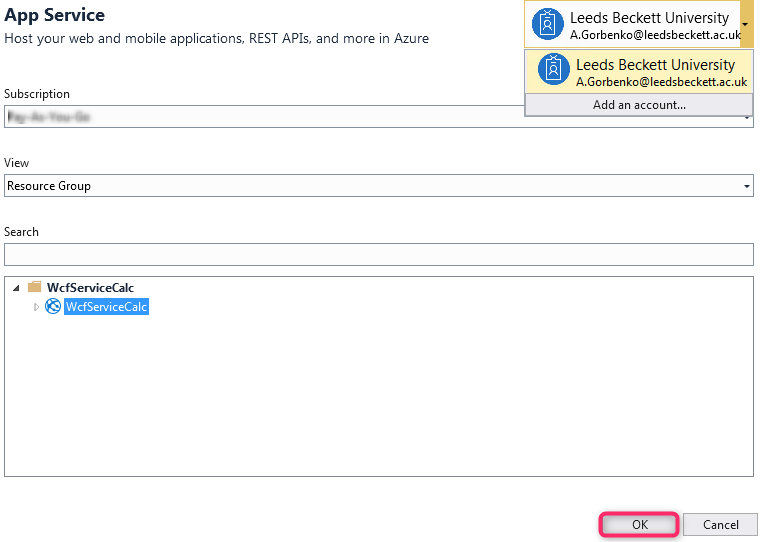


This opens the **Create App Service** dialog, which helps you create all the necessary Azure resources to run the ASP.NET web app in Azure.

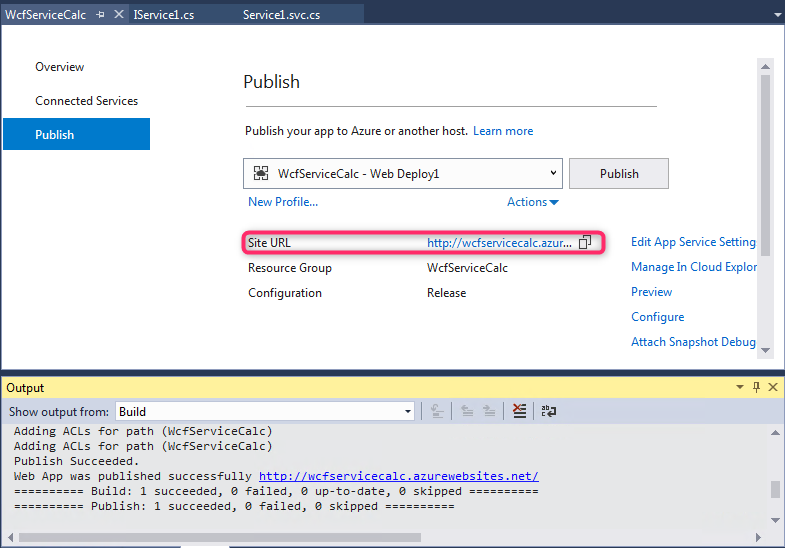
First, pick the **App Service** as a publish target and **Select existing**.



Add Azure account if required, select your free Subscription and browse to the App service you created earlier on Azure portal.



Wait until your project is successfully published to Microsoft Azure.



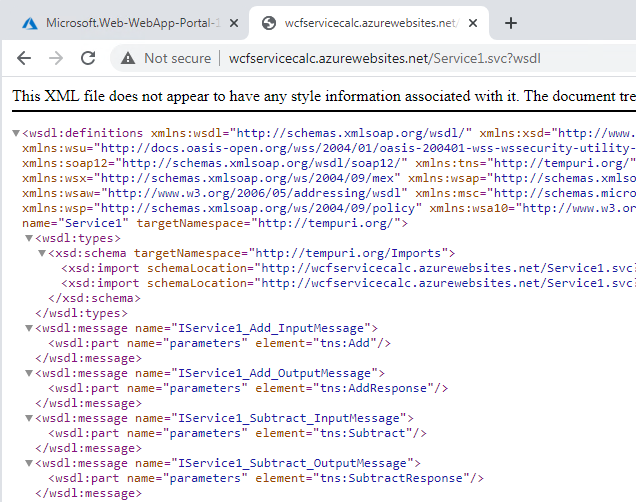
Copy the web service URL and open it in a web browser.

The web app name is used as the URL prefix in the format http://<app\_name>.azurewebsites.net.

Congratulations, your web service is running live in Azure App Service.

Access the web service at [http://wcfservicecalc.azurewebsites.net**/Service1.svc**](http://wcfservicecalc.azurewebsites.net/Service1.svc)

View WSDL description at [http://wcfservicecalc.azurewebsites.net**/Service1.svc?wsdl**](http://wcfservicecalc.azurewebsites.net/Service1.svc?wsdl)

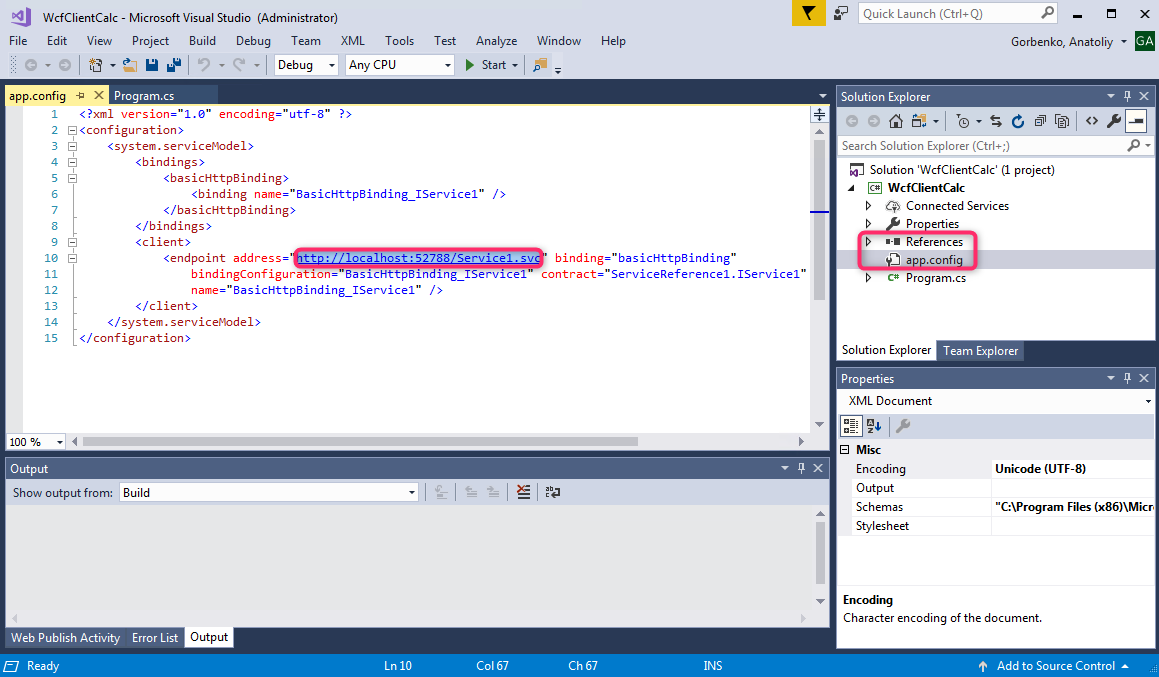


# Step 3: Test Azure WCF Service using SoapUI

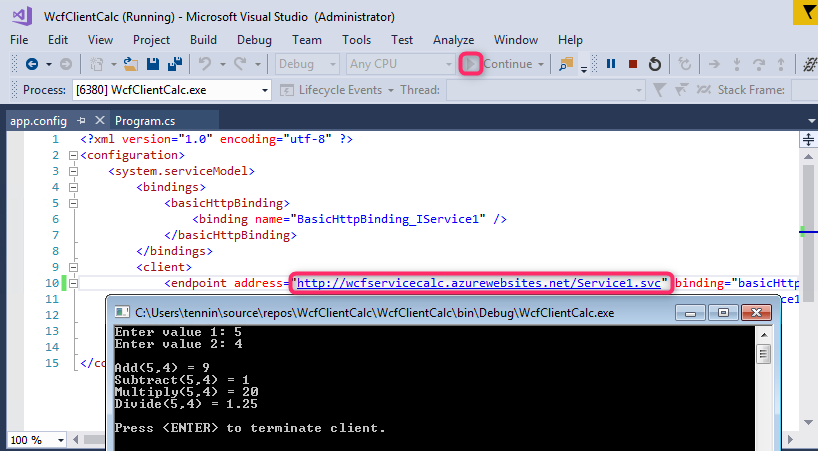
Test the Azure WCF Service using SoapUI (Create SOAP request by providing a ref to the service WSDL: e.g. <http://wcfservicecalc.azurewebsites.net/Service1.svc?wsdl> )

# Step 4: Test Azure WCF Service using desktop client

In Visual Studio open the WcfClientCalc console application project from the source code repository.   
Open app.config file in the Solution Explorer and update the endpoint address with the URL of a web service deployed on Microsoft Azure.



Run the program.



# Step 5. Clean up resources

In the preceding steps, you created Azure resources in a resource group. If you don't expect to need these resources in the future, you can delete them by deleting the resource group on Microsoft Azure portal.

# Advanced assignment

1. Create your own WCF service and deploy it to Microsoft Azure.
2. Develop a desktop.Net client application to communicate with your WCF service hosted on Azure PaaS Cloud.
3. On your own you can also try to can create and run C# RESTful web service by following the tutorial’s instructions <https://docs.microsoft.com/en-us/azure/app-service/app-service-web-tutorial-rest-api>

# References

Azure Web Services

<https://blogs.msdn.microsoft.com/nishasingh/2012/12/04/creating-and-deploying-a-wcf-service-on-windows-azure-and-consuming-it-in-windows-8-store-app/>

<https://docs.microsoft.com/en-us/azure/app-service/app-service-web-tutorial-rest-api>

Azure WCF Clients

<https://docs.microsoft.com/en-us/dotnet/framework/wcf/how-to-create-a-wcf-client>

<https://docs.microsoft.com/en-us/dotnet/framework/wcf/how-to-configure-a-basic-wcf-client>

<https://docs.microsoft.com/en-us/dotnet/framework/wcf/how-to-use-a-wcf-client>

Azure Web Applications

<https://docs.microsoft.com/en-us/azure/app-service/app-service-web-get-started-dotnet>

<https://docs.microsoft.com/en-us/azure/app-service/app-service-web-tutorial-dotnet-sqldatabase>