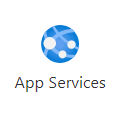
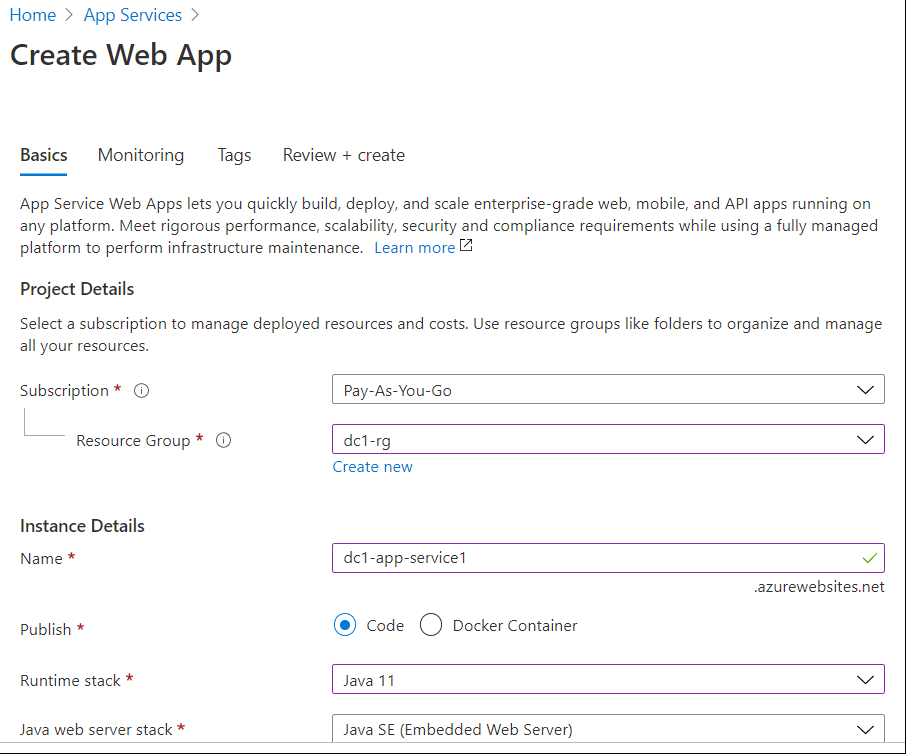
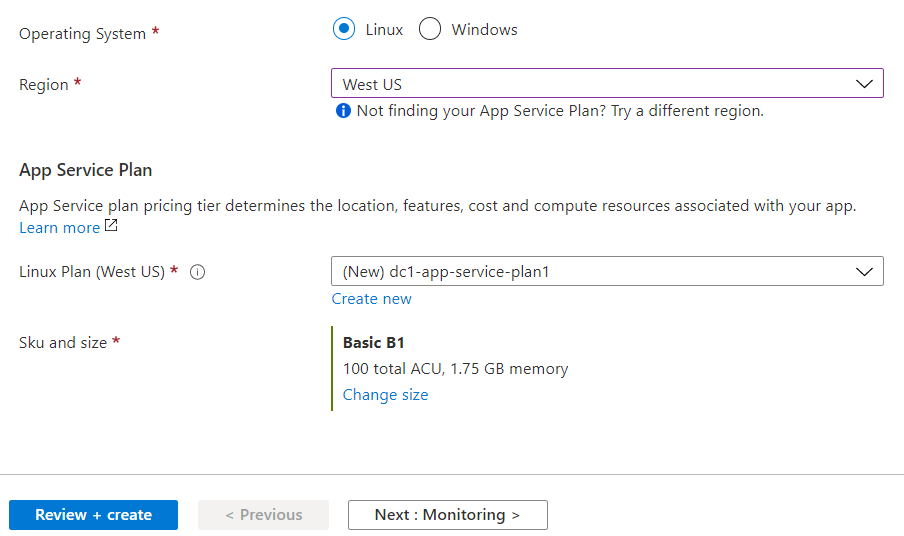
Access Azure Blob Storage from Azure App Service using System Assigned Managed Identity

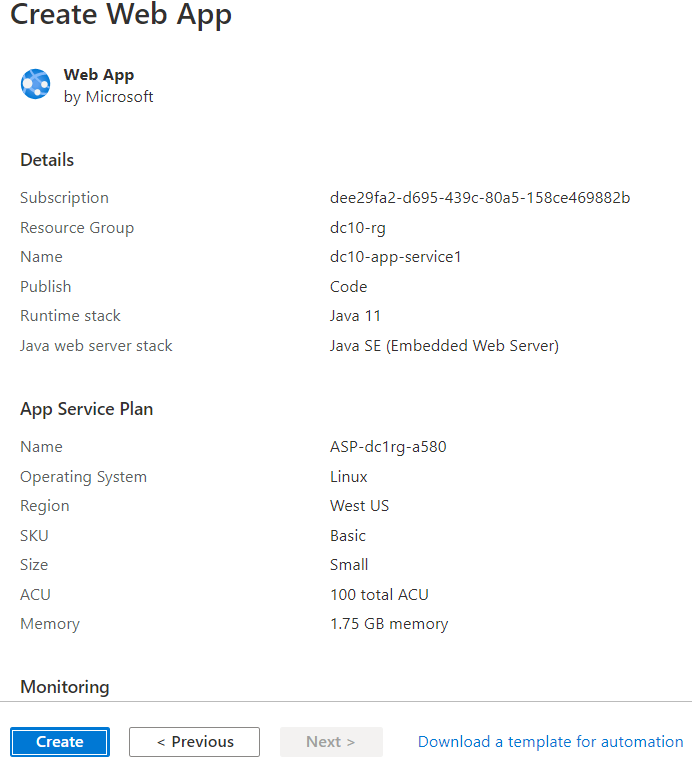
Reference sample code: <https://github.com/milindvb/mbJavaSDKBlobStorageManagedIdentity>

* Create an App Service



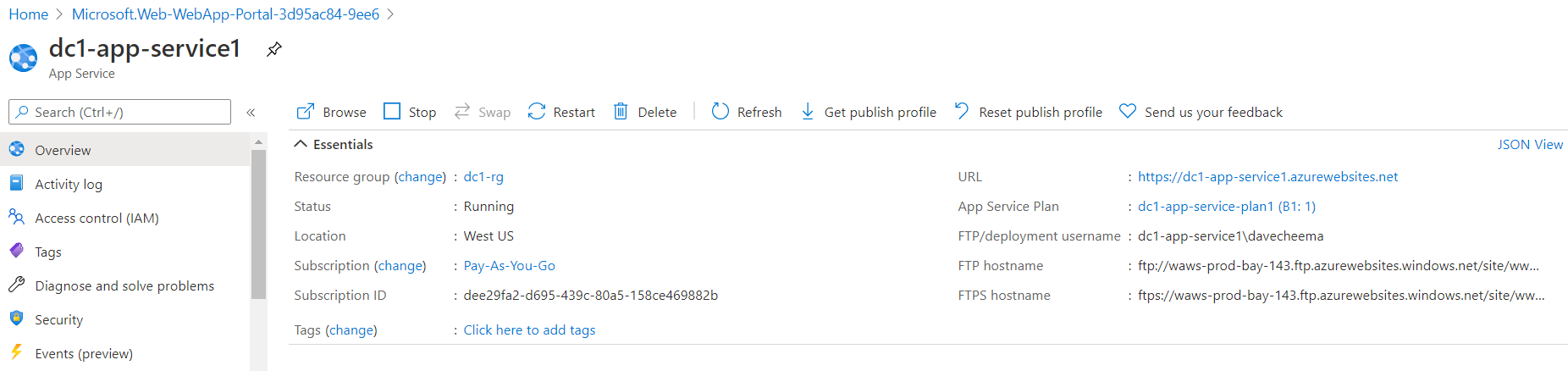
 

* Fill out all fields, click on **Review + create**

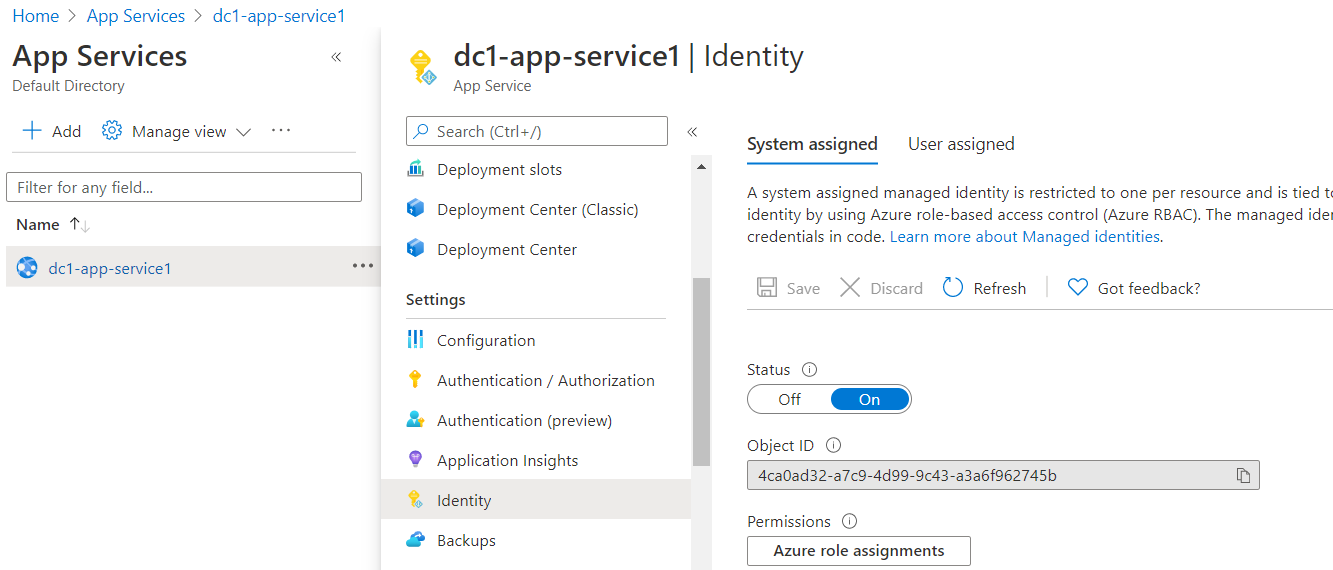


* Click on Create

Note down app service details on the top right side of the page



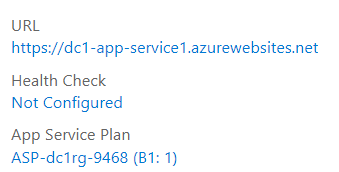
* Go to App Service details
* Click on Identity on the left side



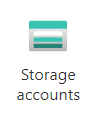
* Select System assigned on the right side
* Slide the Status to **On**

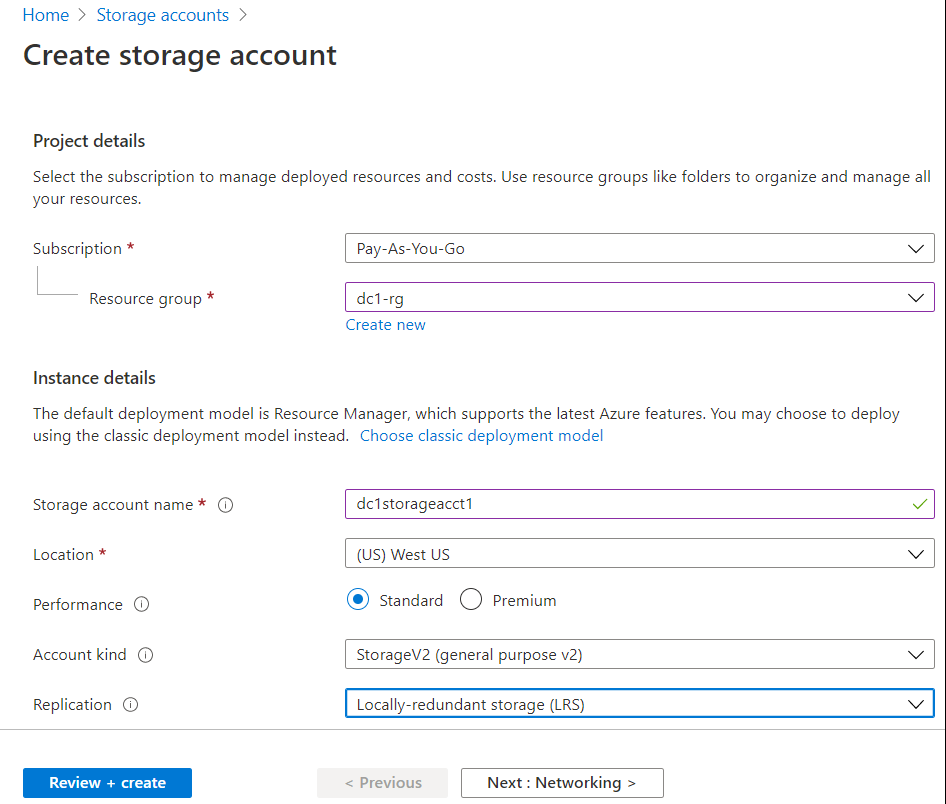
It will generate the Object ID and enable the Save button at the top

* Click on **Save**
* Go to App Service detail screen and copy the following items and save them to a known location:

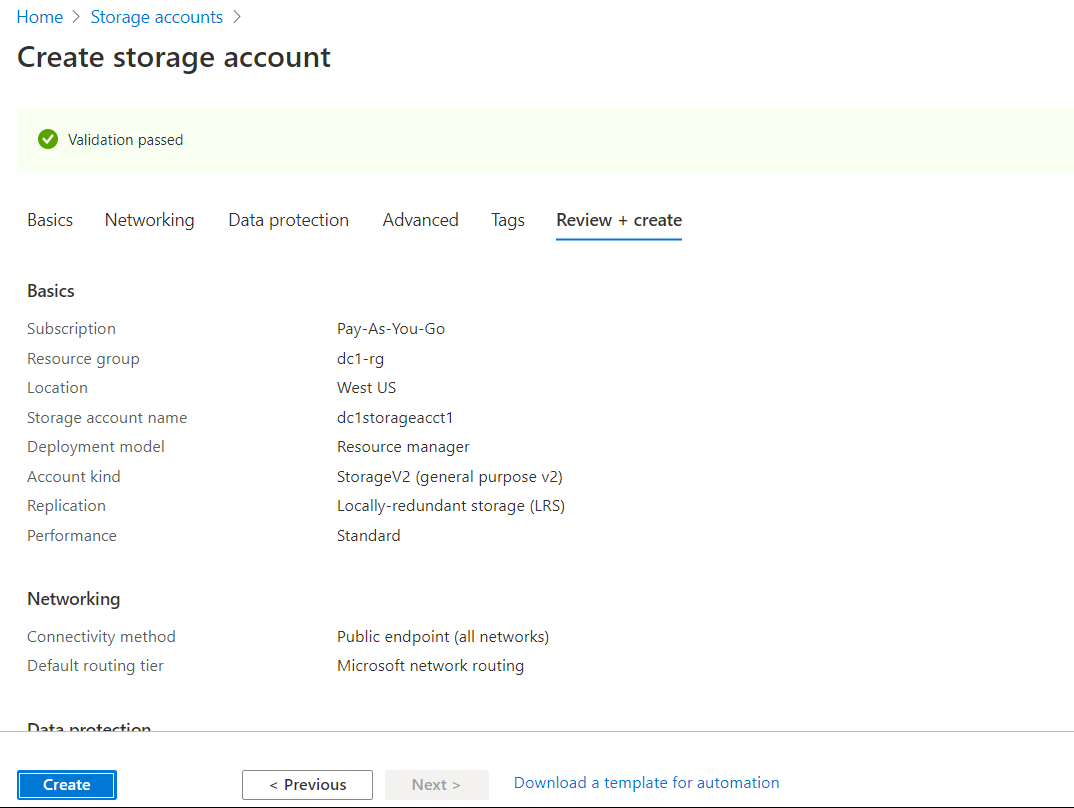


* Make a note of the URL and App Service Plan of the Azure App Service instance
* Create a storage account

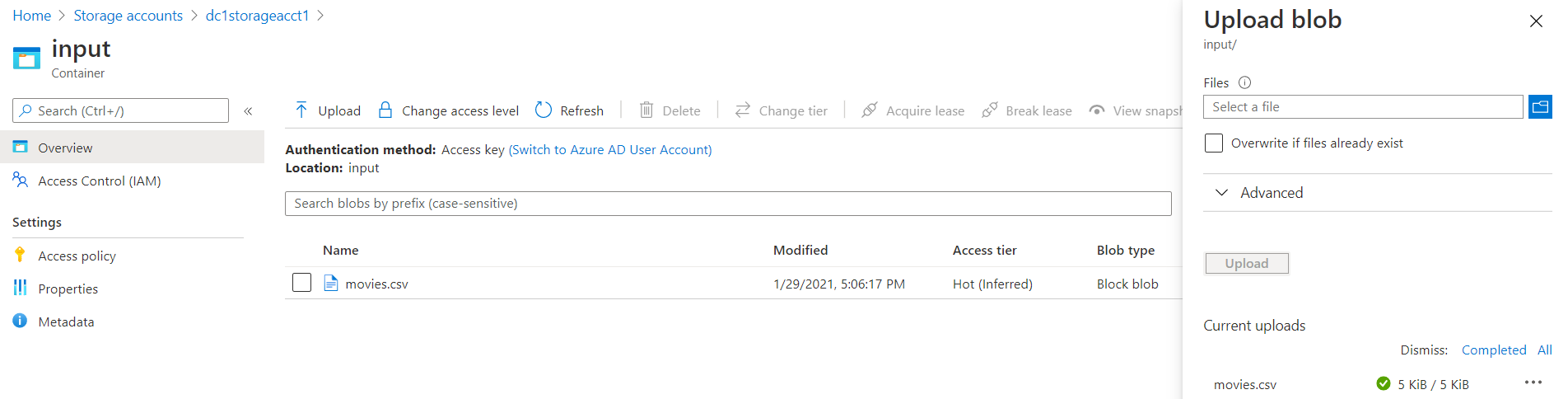




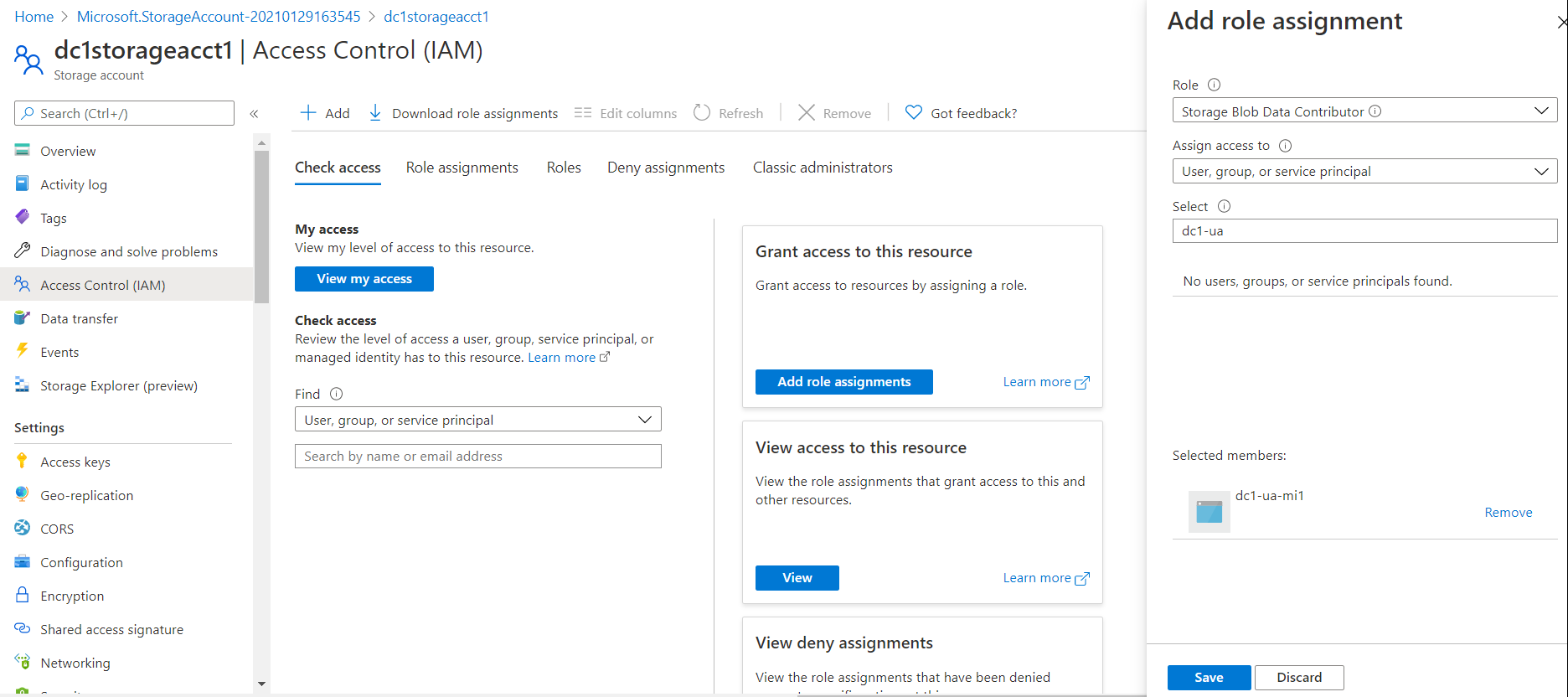
* Enter all required information, click on **Review + create**



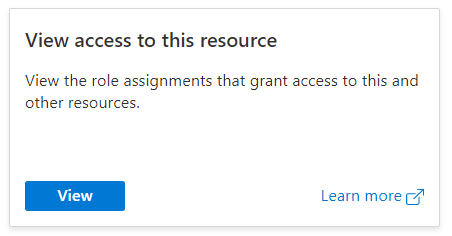
* Create on **Create**
* Create a container and upload a test file as shown below:



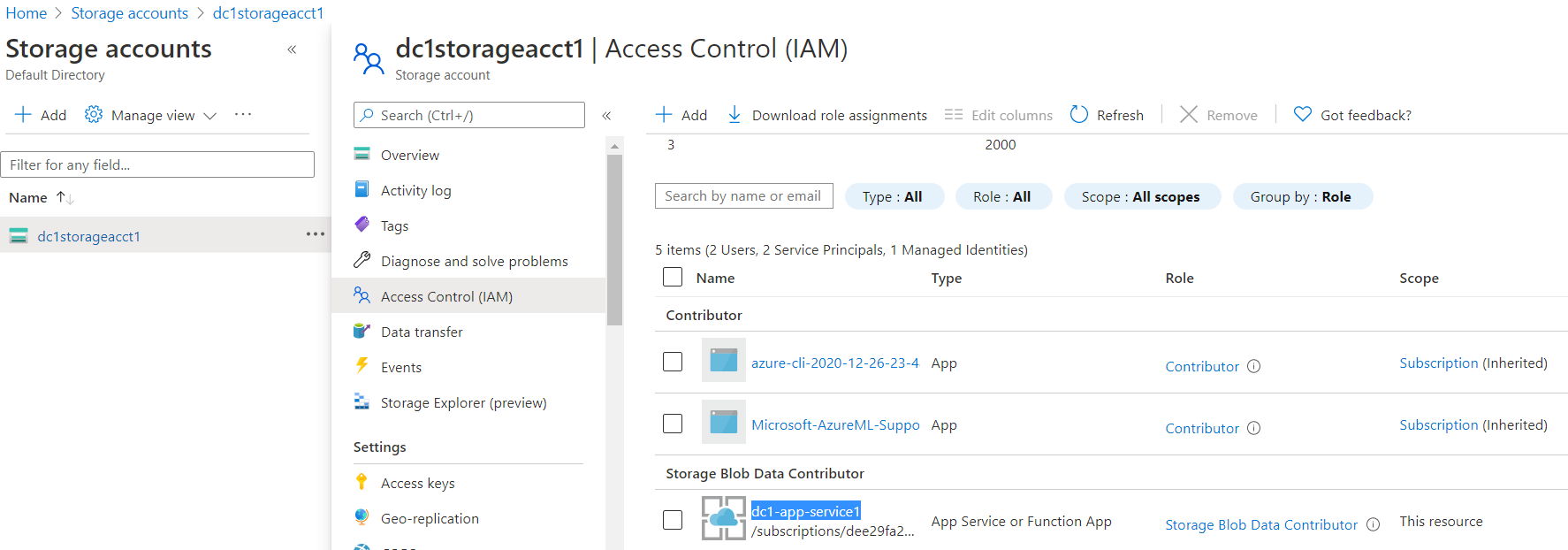
* Go to Storage Account detail



* Click on Access Control (IAM), Click on Add at the top on the right side
* Select **Storage Blob Data Contributor**
* On the far-right side, search for and select **dc1-app-service1** and click on **Save**
* Optionally, you can verify the permissions have added are saved

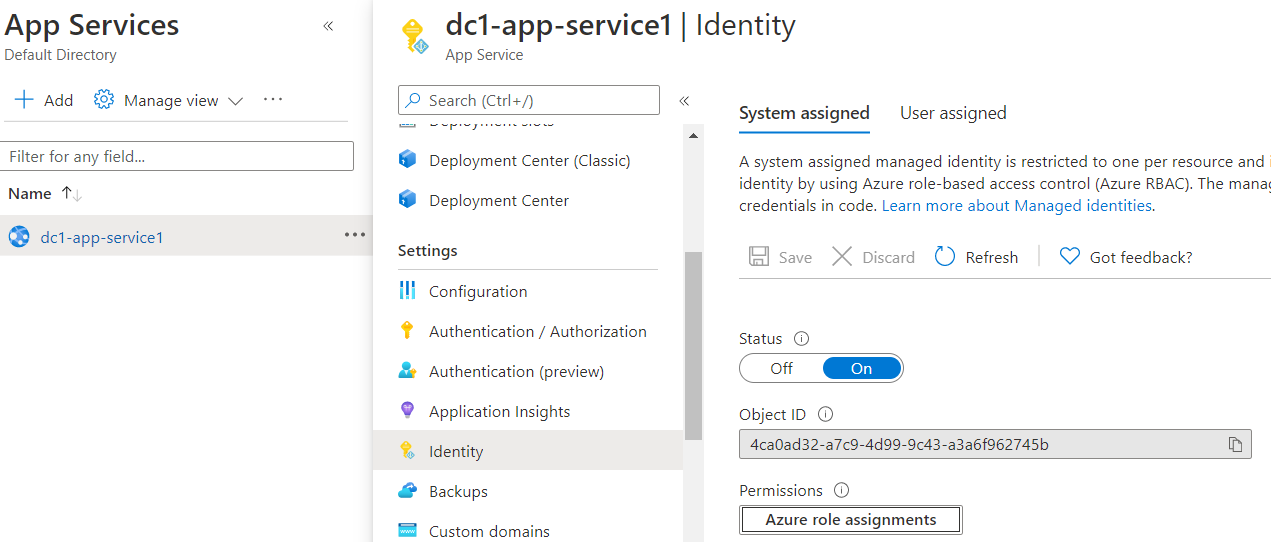


You will see something similar to:

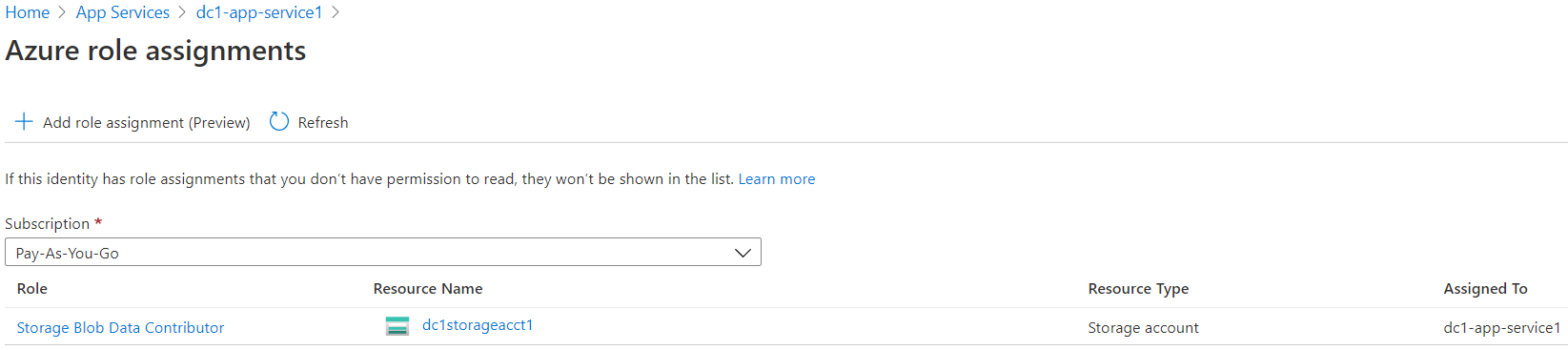


You should be able to see Storage Blob Datta Contributor, dc1-app-service1

* To ensure that the App Service has the proper permissions to the storage, click on **Azure role assignment**:



You should see similar to what is shown below:

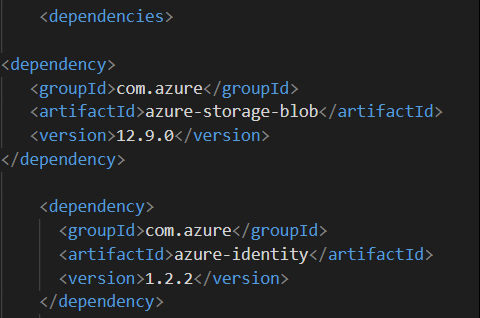


* Go to your **development environment**, e.g., Visual Studio Code, NetBeans, etc., create a web app.

**Note:** the code for this example used here is downloaded from <https://github.com/milindvb/mbJavaSDKBlobStorageManagedIdentity> and **Visual Studio Code** was used to develop and compile the code

* Open the project in Visual Studio Code
* Open **pom.xml** file

1. Go to the **<dependencies>** section and add/ensure the following elements:

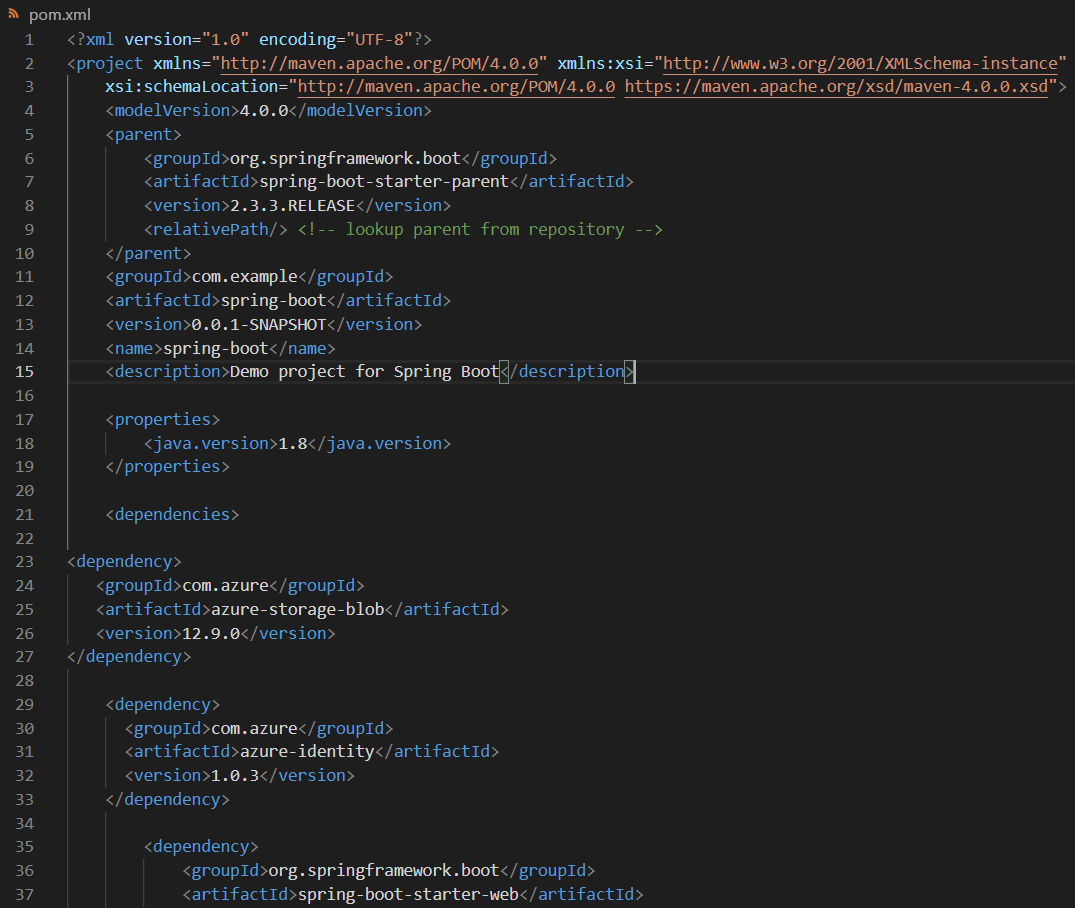
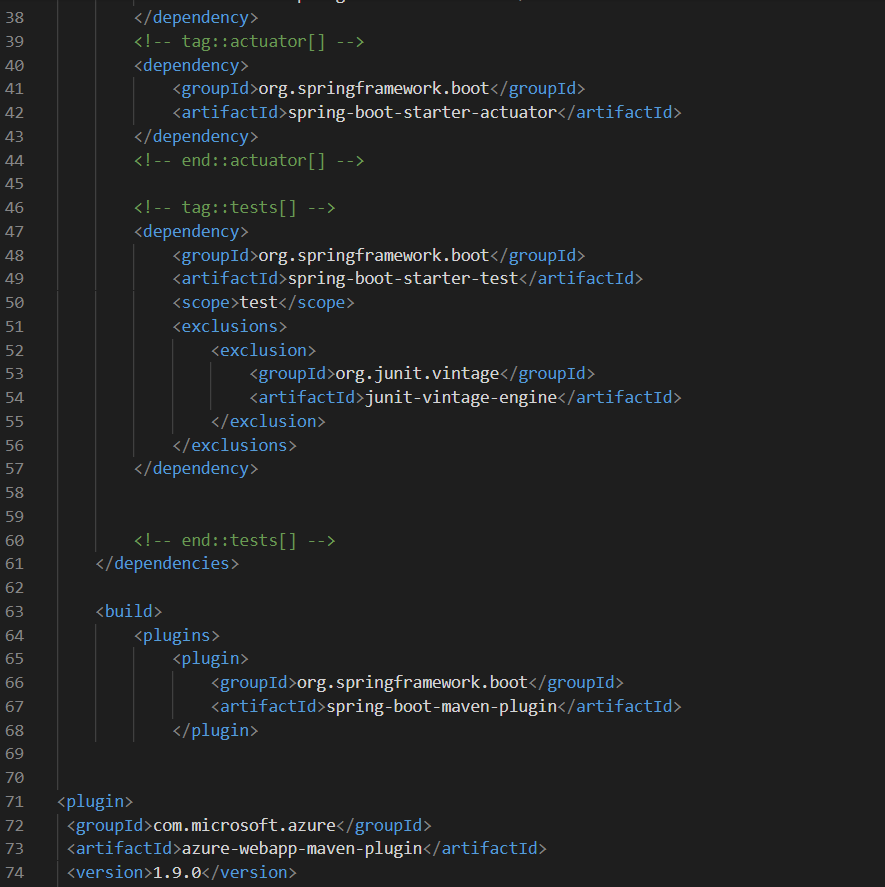


1. Add/update the following elements:

|  |  |  |
| --- | --- | --- |
| Machine generated alternative text: Web App information  <resourceGroup>changeThis-ResourceGrp</resourceGroup>  <appName>changeThis-Appname</appName>  centralus</region>  <reglon>  Java Runtime Stack for Web App on Linux-->  <1inuxRuntime>jre8</1inuxRuntime> | =====> |  |

Note: It must be just **above the <linuxRuntime>** element

The completed pom.xml file should look similar to the one shown below:

* **Save** pom.xml
* **Open** src --> main\java\com\example\springboot --> **HelloController.java** file
* Make the following changes:

Line 18: replace with your storage account name



Line 24: replace with your container name

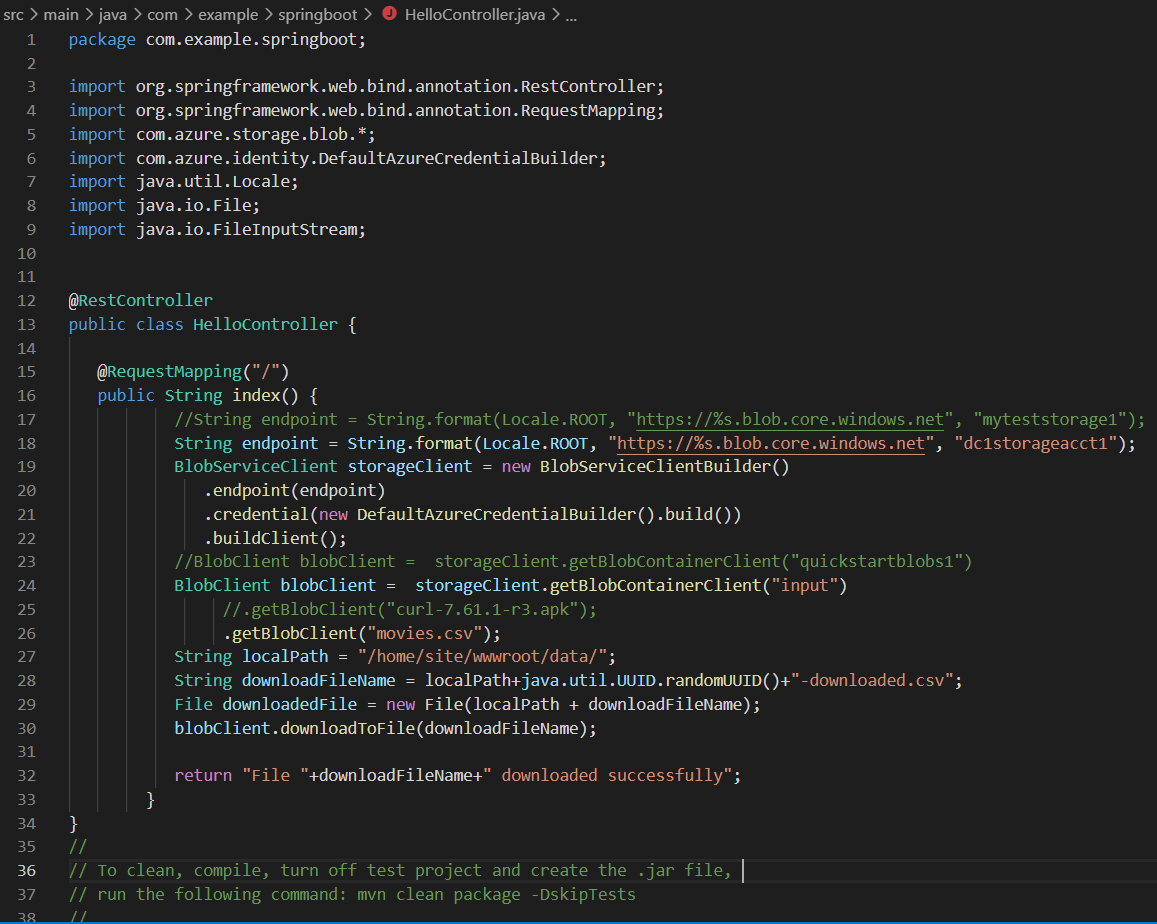


Line 26: replace with your blob name



Note: your line numbers may vary.

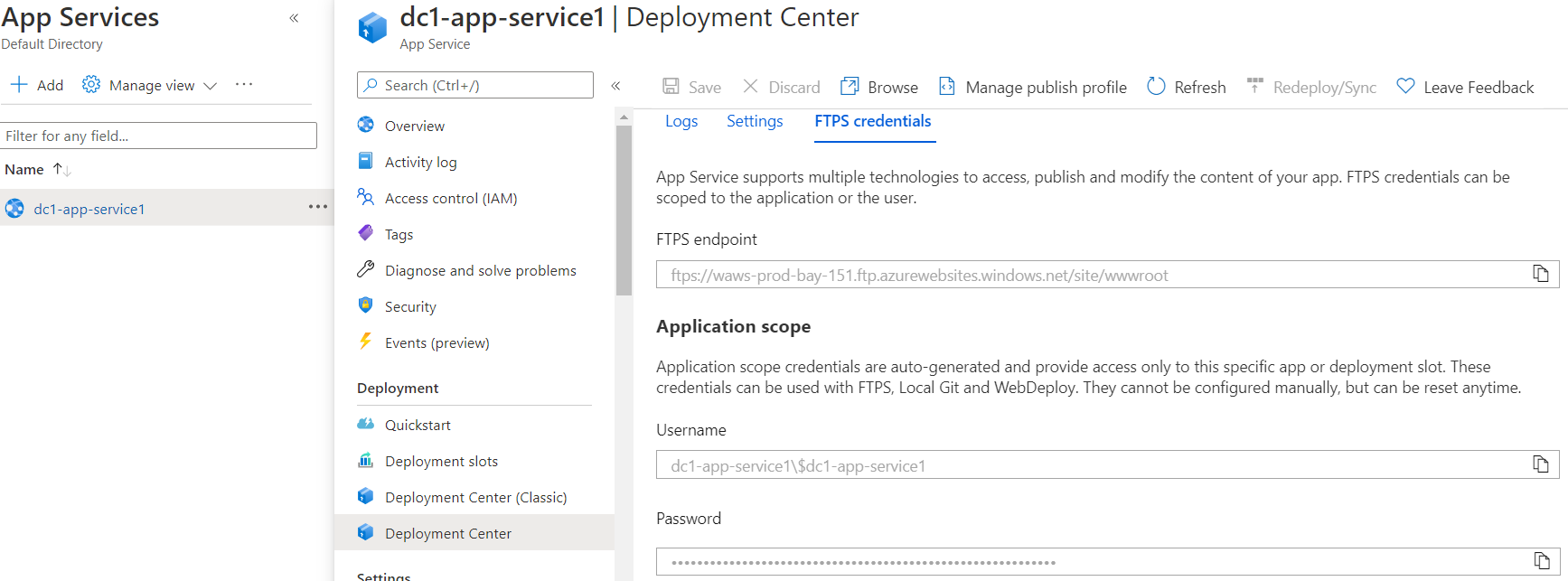
Your source code file should look similar to the one shown below:



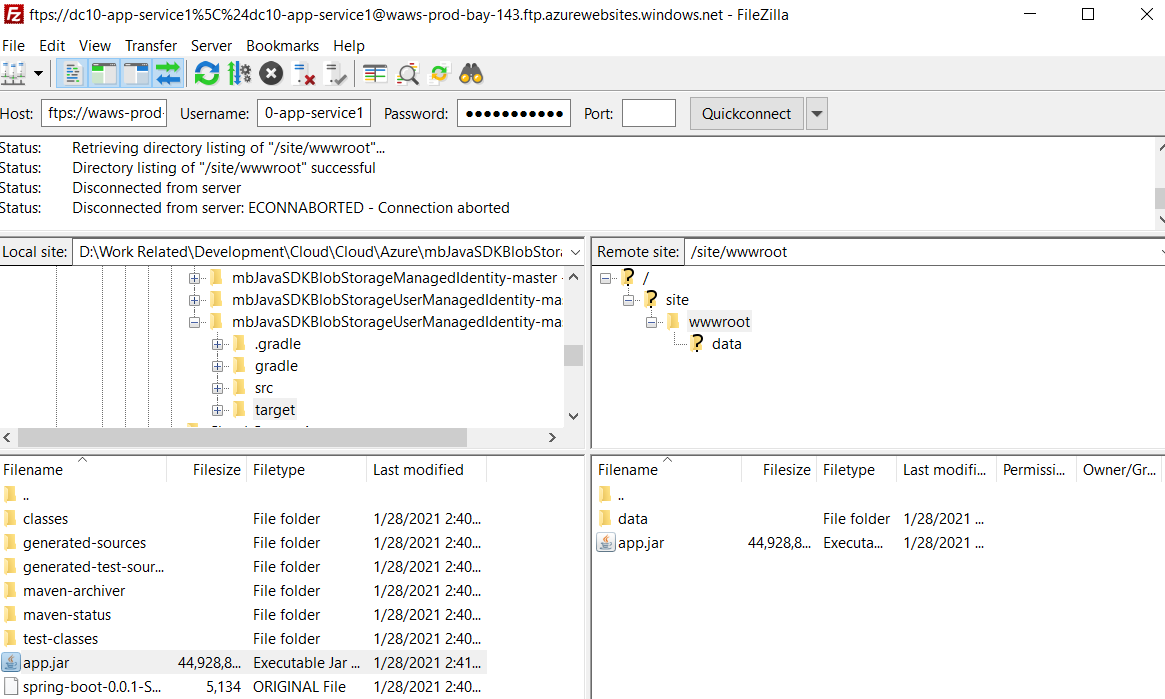
* You can clean the workspace, compile code, turn off the test project and create .jar file using the following command in a Terminal window: **mvn clean package -DskipTes**ts
* Rename your .jar file

|  |  |  |
| --- | --- | --- |
| Machine generated alternative text: v main \ java \ com\example\springboot  O Applicationjava  O HelloController.java  > test  v target  >  >  >  >  >  >  classes  generated-sources  generated-test-sources  maven-archiver  maven-status  test-classes  spring-boot-O.O. 1 -SNAPSHOT.jar | Rename  .jar file to  App.jar | Machine generated alternative text: v main \ java \ com \example\springboot  O Applicationjava  O HelloController.java  > test  v target  >  >  >  >  >  >  classes  generated-sources  generated-test-sources  maven-archiver  maven-status  test-classes  app.Jar |

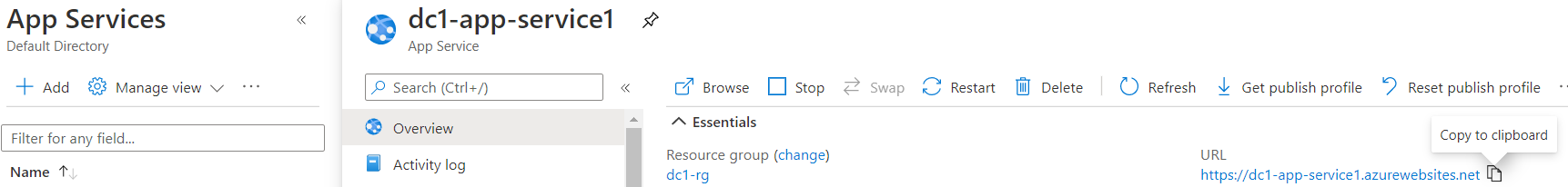
* Go to the target section on the left side and rename **spring-boot-0.0.1-SNAPSHOT.jar** 🡪 **app.jar**
* Go to the App Service in the Azure portal, e.g., dc1-app-service1 🡪 click on **Deployment Center** 🡪 click on **FTP credentials**



* Open up an SFTP (e.g., Filezilla) session



* Copy **FTPS endpoint**, **Username**, **Password** settings of the App Service and enter them into the FileZilla credentials. Hit Enter
* Upload the app.jar file into the Azure App Service workspace (wwwroot)
* Create a folder named **data** under wwwroot folder. *Note: your use case may be different. This folder is created to validate the PoC*
* Go back to the Overview section of the App Service in the Azure portal, **Stop and Start** the service.



* Copy the URL of the App Service, e.g., https://dc1-app-service1.azurewebsites.net
* Wait for a couple of minutes
* Open a new tab in your browser and paste the app service address in the Browser’s Address Bar

If all goes well, you should a message similar to what is shown below:

