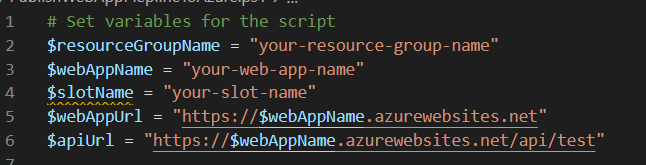
This script is written in PowerShell, a scripting language used for automating administrative tasks on Windows machines. The purpose of this script is to publish a web application to Azure and test the API connectivity.

Text

Description automatically generated

Here are the step-by-step explanations for each part of the script:



Set Variables:

The first step of the script is to set variables for the different parameters that will be used throughout the script. The variables are assigned values that correspond to the specific names of the Azure resources that are being used. For example, the variable $resourceGroupName is set to the name of the resource group that the web app is in.

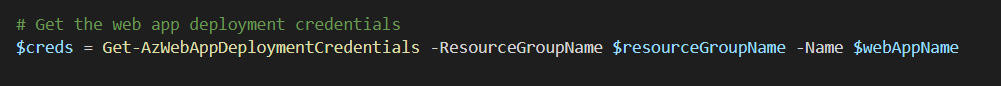
Connect to Azure:

Text

Description automatically generated

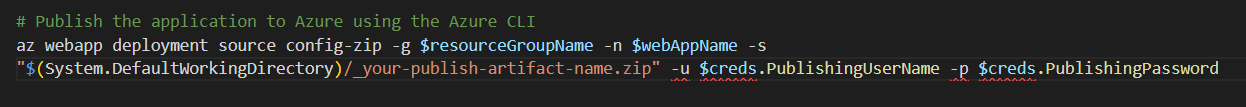
The script uses the Connect-AzAccount command to connect to Azure using the Azure PowerShell module. This command prompts the user to enter their Azure credentials, which are then used to authenticate the script to access Azure resources.

Get Web App Deployment Credentials:



The next step is to retrieve the deployment credentials for the web app. This is done using the Get-AzWebAppDeploymentCredentials command, which retrieves the publishing credentials for the specified web app. The credentials are stored in the $creds variable for later use.

Publish the Application to Azure:



The script uses the az webapp deployment source config-zip command to publish the application to Azure. This command uploads the application package in the form of a .zip file to the specified web app. The command takes several parameters, including the resource group name, web app name, and deployment credentials, which are passed as arguments using the variables set earlier.

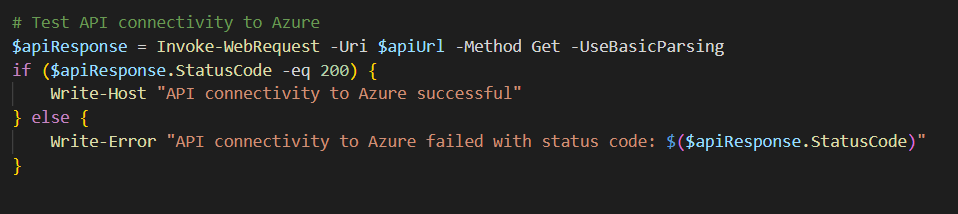
Test Publishing of the Application in Azure:

Text

Description automatically generated

The script then tests the publishing of the application in Azure by sending a GET request to the web app URL using the Invoke-WebRequest command. The response status code is then checked to ensure that the application has been successfully published to Azure. If the status code is 200, which indicates success, a message is displayed saying that the application has been successfully published to Azure. If the status code is anything other than 200, an error message is displayed.

Test API Connectivity to Azure:



Finally, the script tests the API connectivity to Azure by sending a GET request to the API URL using the Invoke-WebRequest command. The response status code is checked to ensure that the API connectivity is successful. If the status code is 200, which indicates success, a message is displayed saying that the API connectivity to Azure is successful. If the status code is anything other than 200, an error message is displayed.

In summary, this script connects to Azure, retrieves deployment credentials, publishes a web application to Azure, and tests both the publishing of the application and the API connectivity to Azure.