How to deploy **.NET Core Web API** (AlphabetCheckWebApp) to **Azure Cloud** using **Azure App Service**.

**1. Deploying the Web API to Azure App Service (Direct Deployment)**

**Prerequisites:**

* **Azure account**.
* Install **Azure CLI**: [Install Azure CLI](https://learn.microsoft.com/en-us/cli/azure/install-azure-cli).
* Install **Visual Studio** or **Visual Studio Code** for the development and deployment.

**Step-by-Step Guide:**

**Step 1: Create an App Service on Azure**

1. **Log in to Azure CLI**:

az login

1. **Create a Resource Group** (if you don’t have one already):

az group create --name AlphabetCheckResourceGroup --location eastus

1. **Create an App Service Plan**: This defines the region and pricing tier for your app.

az appservice plan create --name AlphabetCheckAppServicePlan --resource-group AlphabetCheckResourceGroup --sku B1 --is-linux

1. **Create a Web App**: Now, create the web app that will host your API.

az webapp create --name AlphabetCheckWebApi --resource-group AlphabetCheckResourceGroup --plan AlphabetCheckAppServicePlan --runtime "DOTNET|8.0"

**Step 2: Deploy the Application**

1. **Publish the application**: You need to publish the app to a folder before deploying it to Azure. In **Visual Studio**:
   * Right-click on the project in **Solution Explorer**.
   * Select **Publish**.
   * Choose **Azure** as the target and select your **Azure App Service** created earlier.
   * Click **Publish**.

Alternatively, we can publish from the command line, use the following:

**dotnet publish --configuration Release --output ./publish**

1. **Access the Web API**: Once deployed, you can visit the app using the URL:

[https://AlphabetCheckWebApi.azurewebsites.net/api/alphabetcheck/{inputString}](https://AlphabetCheckWebApi.azurewebsites.net/api/alphabetcheck/%7binputString%7d)

Using CI/CD pipeline

**Deploying a .NET Web API to Azure App Service Using Azure DevOps**

**Prerequisites**

1. **Azure Account**
2. **Azure App Service**: Create an App Service for hosting .NET Web API.
3. **Azure DevOps Account**
4. **Code Repository**:
5. **Azure Service Connection**: Set up a service connection in Azure DevOps to allow the pipeline to deploy to Azure App Service.

**Step 2: Create a Service Connection in Azure DevOps**

**Step 3: Create a CI Pipeline for Build and Test**

1. In **Azure DevOps**, **Pipelines** > **Create Pipeline**.
2. Select repository (Azure Repos Git or GitHub).
3. Choose **YAML** for pipeline configuration.
4. Create or edit a azure-pipelines.yml file to define the pipeline.

**Example azure-pipelines.yml for Building the .NET Web API:**

trigger:

branches:

include:

- main # Trigger pipeline on push to the main branch

pool:

vmImage: 'windows-latest' # Use the latest Windows build agent

variables:

buildConfiguration: 'Release'

steps:

- task: UseDotNet@2

inputs:

packageType: 'sdk'

version: '6.x' # Specify the version of .NET SDK to use

- task: DotNetCoreCLI@2

inputs:

command: 'restore'

projects: '\*\*/\*.csproj' # Restore NuGet packages

- task: DotNetCoreCLI@2

inputs:

command: 'build'

projects: '\*\*/\*.csproj'

arguments: '--configuration $(buildConfiguration)' # Build the project

- task: DotNetCoreCLI@2

inputs:

command: 'test'

projects: '\*\*/\*Tests.csproj'

arguments: '--configuration $(buildConfiguration)' # Run tests

- task: DotNetCoreCLI@2

inputs:

command: 'publish'

projects: '\*\*/\*.csproj'

arguments: '--configuration $(buildConfiguration) --output $(Build.ArtifactStagingDirectory)' # Publish the app

zipAfterPublish: true

**Explanation:**

1. **UseDotNet** task installs the necessary .NET SDK.
2. **DotNetCoreCLI restore** restores NuGet packages.
3. **DotNetCoreCLI build** builds the project.
4. **DotNetCoreCLI test** runs unit tests.
5. **DotNetCoreCLI publish** publishes the application into an output folder, which will be used in the deployment pipeline.

**Step 4: Create a CD Pipeline to Deploy to Azure App Service**

After CI pipeline set up, create a new **Release Pipeline** to deploy application.

**Example Release Pipeline Steps for Deployment:**

1. In Azure DevOps, **Pipelines** > **Releases**.
2. Create a new **Release Pipeline**.
3. Add an **Artifact** by selecting **Build Pipeline** .
4. Add a **Stage** (e.g., Dev or Production).
5. In the **Stage**, add a new **Deploy to Azure App Service** task.

**Example Deploy Task Configuration:**

1. In the **Deploy** stage, click **Add** > **Azure App Service Deploy**.
2. Set up the following:
   * **Azure Subscription**: Choose the service connection you created earlier.
   * **App Service Name**: Select the **App Service** you created in Step 1.
   * **Package or Folder**: Select $(System.ArtifactsDirectory)/drop (the published output from the build).
   * **Deploy to Slot**
3. Save and run the pipeline.