# Publishing from Visual Studio

Deploying C# Web API to Azure

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

In future this plan can always be scaled up if needs be. Click OK

Graphical user interface, application

Description automatically generated

App service instance has been created and notice by default the new instance is already selected meaning this is where the API app to this app service instance.

Graphical user interface, text, application, email

Description automatically generated

Don’t need to worry about API management for this demo so skip this step.

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Note may not have time to click check box

So now have an app server for Visual Studio to deploy to. If the API app has any dependencies (such as SQL Server) then they will have been located by the wizard and an option to configure them will be made available.

If do need SQL Server:

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Need to create an instance of SQL Server so click on the “New” option

Graphical user interface, application

Description automatically generated

Need to remove the dot in the server name, specify a location, admin name and password.

Graphical user interface, text, application, email

Description automatically generated

SQL Server instance and database are both being created in Azure

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Now we have a SQL Server instance but we need to deploy our database to it. So we need to tell Visual Studio to execute the database migrations when the app is deployed.

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Deployment should now take place, it may take a while.

Graphical user interface, text, application

Description automatically generated

404 error because at the root url there is nothing to display. To invoke the API’s functionality simply extend the url to include a valid endpoint:

# Publishing From Visual Studio Code

See [Publish an ASP.NET Core app to Azure with Visual Studio Code | Microsoft Learn](https://learn.microsoft.com/en-us/aspnet/core/tutorials/publish-to-azure-webapp-using-vscode?view=aspnetcore-6.0) for step by step guide

Graphical user interface, text, application, email

Description automatically generated

In the integrated terminal, use the following command to generate a Release package in a folder located at bin/Publish:

dotnet publish -c Release -o ./bin/Publish

**Publish to Azure App Service**

Leveraging the Azure App Service extension for Visual Studio Code, follow the steps below to publish the website directly to the Azure App Service.

**Create a new Azure Web App resource**

If you don't have an existing Azure Web App resource to publish to, you must create one.

1. In the Azure extension tab, in the **RESOURCES** pane, expand the subscription you wish to use.  
   Graphical user interface, text, application, chat or text message

   Description automatically generated  
   Text

   Description automatically generated  
   A screenshot of a computer

   Description automatically generated with medium confidence
2. Right-click **App Services** and select **Create New Web App...**.
3. Follow the prompts:
   1. Enter a unique name for the web app.
   2. Select the most recent stable .NET runtime (such as .NET 6 (LTS)). Do not select the ASP.NET runtime, which is for .NET Framework apps.
   3. Select your pricing tier. Free (F1) is acceptable for this tutorial.

**Publish to Azure**

1. Right click the bin\Publish folder, SCROLL DOWN TO and select Deploy to Web App... and follow the prompts.
   1. Select the subscription where the Azure Web App resource is located.
   2. Select the Azure Web App resource to which you will publish.
   3. Select **Deploy** when prompted with a confirmation dialog.
2. Once the deployment is finished, click Browse Website to validate the deployment.

Graphical user interface, text

Description automatically generated

If Above window disappears you can right click on the App Service and select “Browse Website”

A screenshot of a computer

Description automatically generated with medium confidence

# Monitoring Apps

## Enable application logging (Windows)

To enable application logging for Windows apps in the [Azure portal](https://portal.azure.com/), navigate to your app and select **App Service logs**.

Graphical user interface, application

Description automatically generated

Select **On** for either **Application Logging (Filesystem)** or **Application Logging (Blob)**, or both.

The **Filesystem** option is for temporary debugging purposes, and turns itself off in 12 hours. The **Blob** option is for long-term logging, and needs a blob storage container to write logs to. The **Blob** option also includes additional information in the log messages, such as the ID of the origin VM instance of the log message (InstanceId), thread ID (Tid), and a more granular timestamp ([EventTickCount](https://learn.microsoft.com/en-us/dotnet/api/system.datetime.ticks)).

# Logging exceptions in the code can be monitored and logged via monitoring logs or via AppInsights.

Add line of code that throws an exception in one of the API methods

Graphical user interface, text

Description automatically generated with medium confidence

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, chat or text message

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

A screenshot of a computer

Description automatically generated

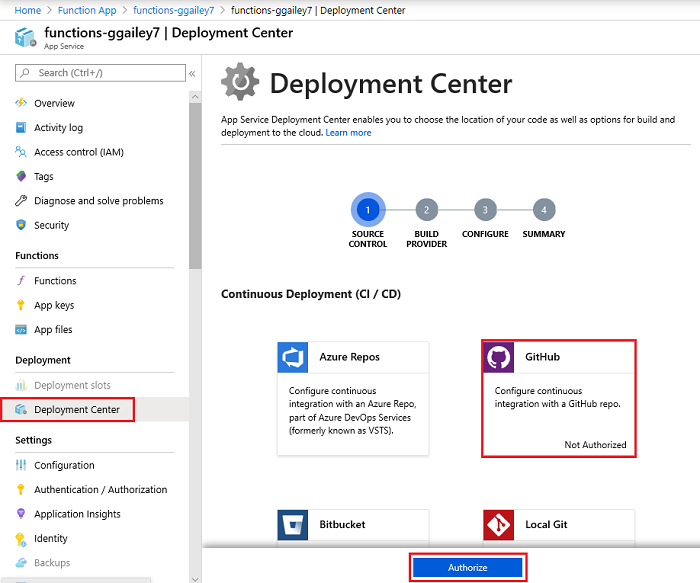
Graphical user interface, application, Word

Description automatically generated

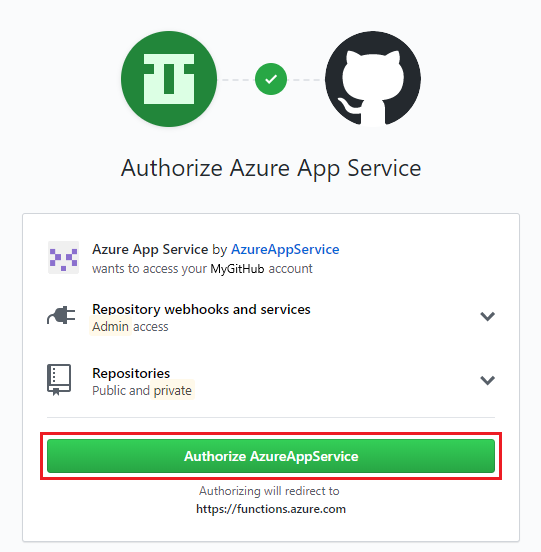
# CI/CD GitHub to Azure

See: [Continuous deployment for Azure Functions | Microsoft Learn](https://learn.microsoft.com/en-us/azure/azure-functions/functions-continuous-deployment#continuous-deployment-requirements)

1. In your function app in the [Azure portal](https://portal.azure.com/), select **Deployment Center**, select **GitHub**, and then select **Authorize**. If you've already authorized GitHub, select **Continue** and skip the next step.



1. In GitHub, select **Authorize AzureAppService**.

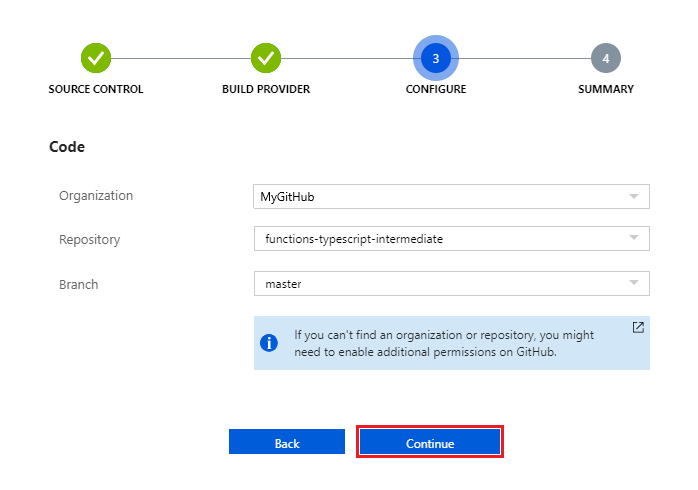


Enter your GitHub password and then select **Continue**.

1. Select one of the following build providers:
   * **~~Git Hub Actions~~ This is the default but doesn’t seem to work**
   * **App Service build service**: Best when you don't need a build or if you need a generic build. **SELECT THIS OPTION!**
   * **~~Azure Pipelines~~** : Best when you need more control over the build. This provider currently is in preview.

Select **Continue**.

1. Configure information specific to the source control option you specified. For GitHub, you must enter or select values for **Organization**, **Repository**, and **Branch**. The values are based on the location of your code. Then, select **Continue**.



1. Review all details, and then select **Finish** to complete your deployment configuration.