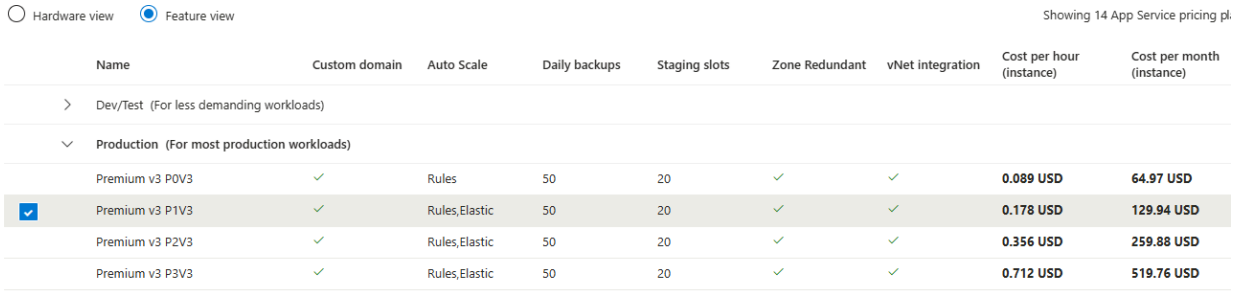
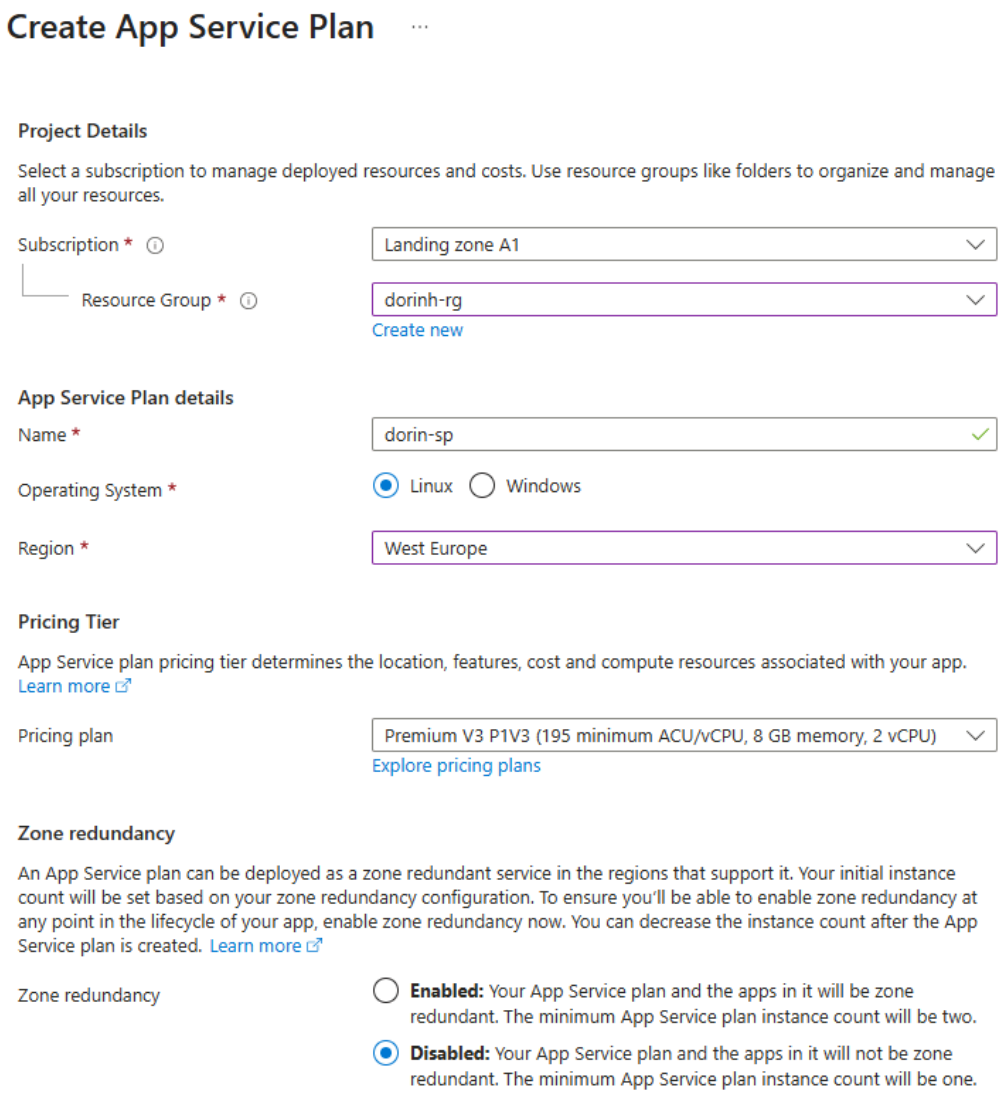
Create an App Service plan

1. Sign in to the Azure Portal
2. In the left menu, select Create a resource → search for App Service plan → click Create.
3. Project Details
    a. Subscription → Select your subscription.
    b. Resource Group → Choose an existing resource group or click Create new.
4. App Service Plan Details
    a. Name → Enter a unique name for your App Service Plan.
    b. Operating System → Select Linux or Windows, depending on your workload.
    c. Region → Choose the Azure region closest to your users (e.g., West Europe, East US).
5. Pricing Tier
    a. Pricing Plan → Select a tier that fits your needs (e.g., Free, Basic B1, Premium V3 P1V3).



	Name	Custom domain	Auto Scale	Daily backups	Staging slots	Zone Redundant	vNet integration	Cost per hour (instance)	Cost per month (instance)
	Dev/Test (For less demanding workloads)								
	Production (For most production workloads)								
	Premium v3 P0V3	✓	Rules	50	20	✓	✓	**0.089 USD**	**64.97 USD**
☑	Premium v3 P1V3	✓	Rules,Elastic	50	20	✓	✓	**0.178 USD**	**129.94 USD**
	Premium v3 P2V3	✓	Rules,Elastic	50	20	✓	✓	**0.356 USD**	**259.88 USD**
	Premium v3 P3V3	✓	Rules,Elastic	50	20	✓	✓	**0.712 USD**	**519.76 USD**

6. Zone Redundancy
    a. Enabled → Your App Service Plan will run across availability zones (if supported).
    b. Disabled → Your App Service Plan will not be zone redundant.
7. Click Review + Create.
8. Once validation passes, click Create.
9. After deployment, your App Service Plan will be available in the chosen resource group.

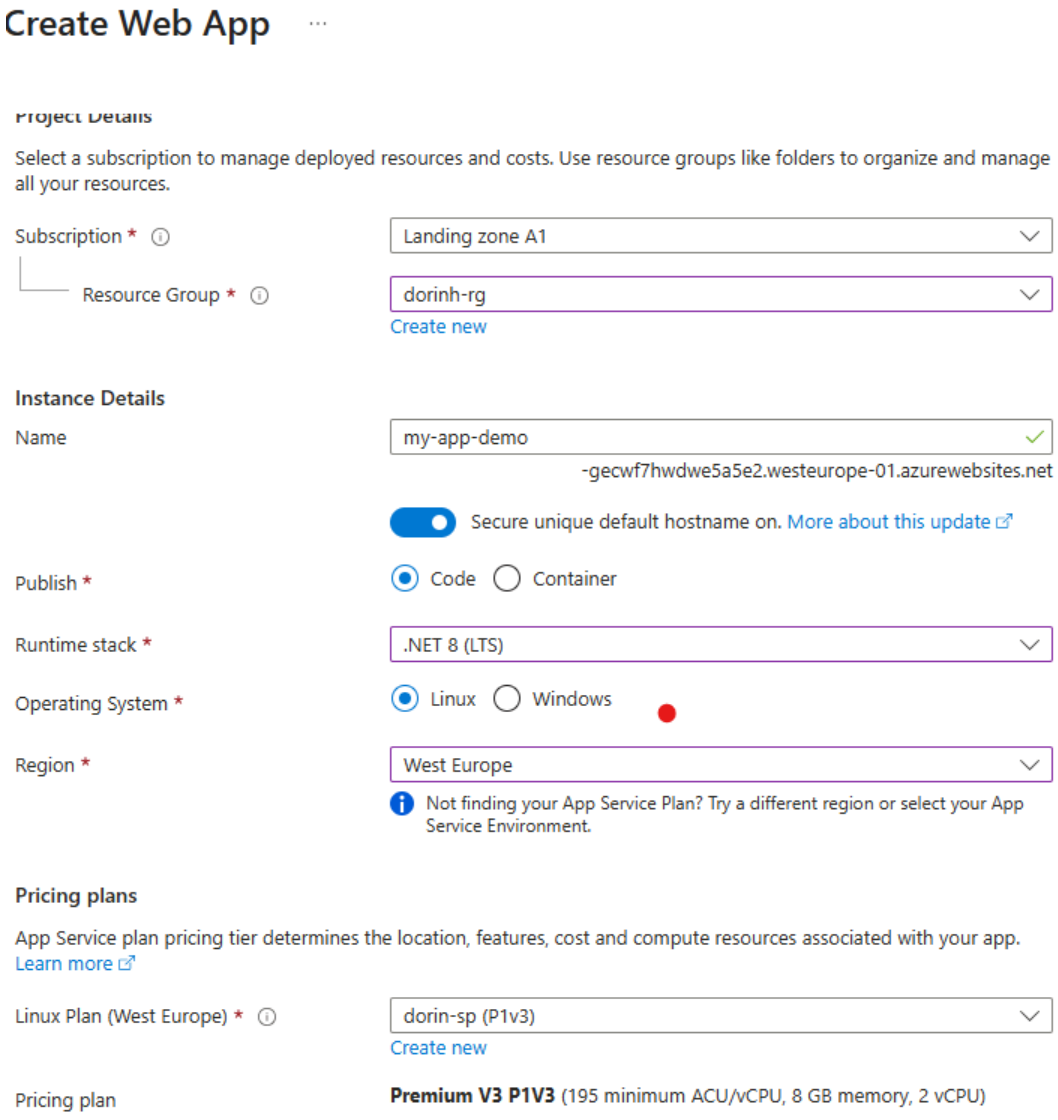


Create a web app

1. Navigate to App Services
    a. In the Azure portal search for App Services in the top search bar.
    b. Select Create > Web App.
2. Select Basics
    a. **Subscription**: Choose your Azure subscription.
    b. **Resource Group**: Enter a new name or select an existing resource group.
    c. **Name**: Enter a globally unique name for your web app.
    d. **Publish**: Choose Code.
    e. **Runtime stack**: Select your desired runtime: .Net 8
    f. **Operating System**: Choose Linux.

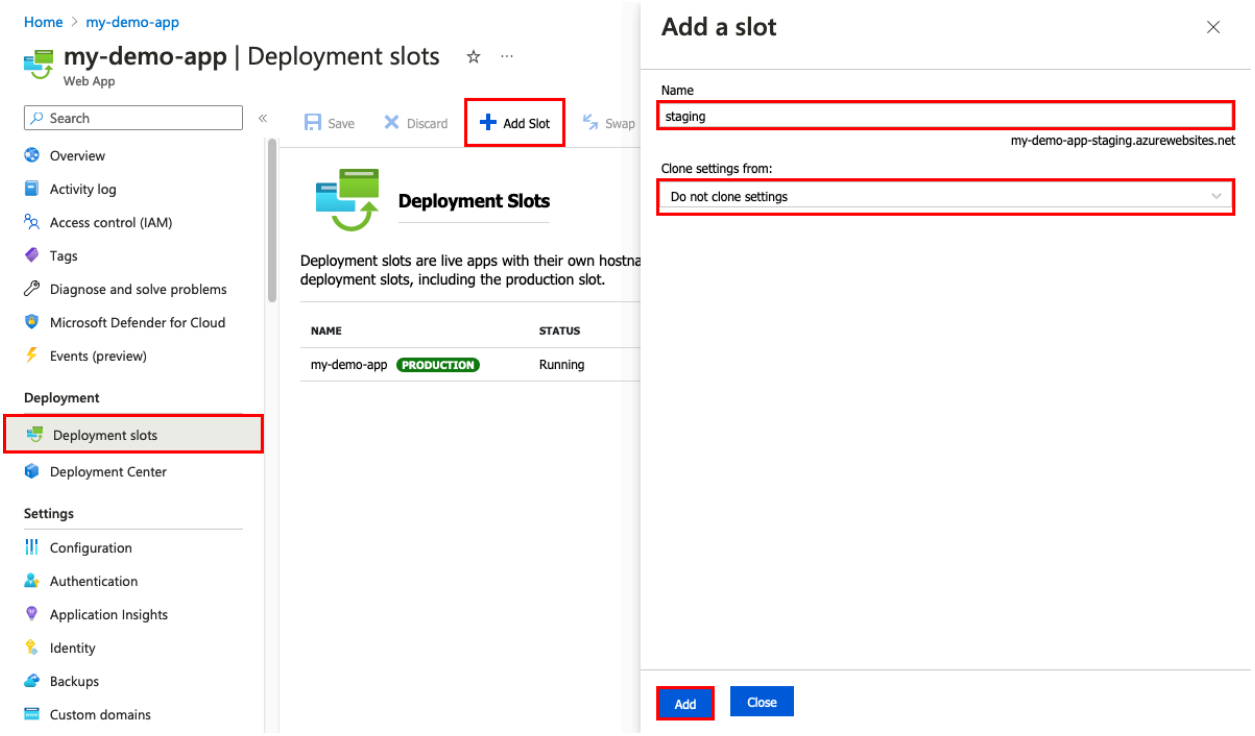
g. **Region**: Select the Azure region where you want to host the app.

3. App Service Plan
    a. Select an existing App Service plan or create a new one.
    b. If creating a new plan:
        i. Enter the plan name.
        ii. Select the SKU
        iii. Select the size/tier that meets your requirements.
4. Select Next: Tags and add any resource tags if needed.
5. Review + Create
6. Select Review + create.
7. Verify all details.
8. Select Create to deploy the web app.



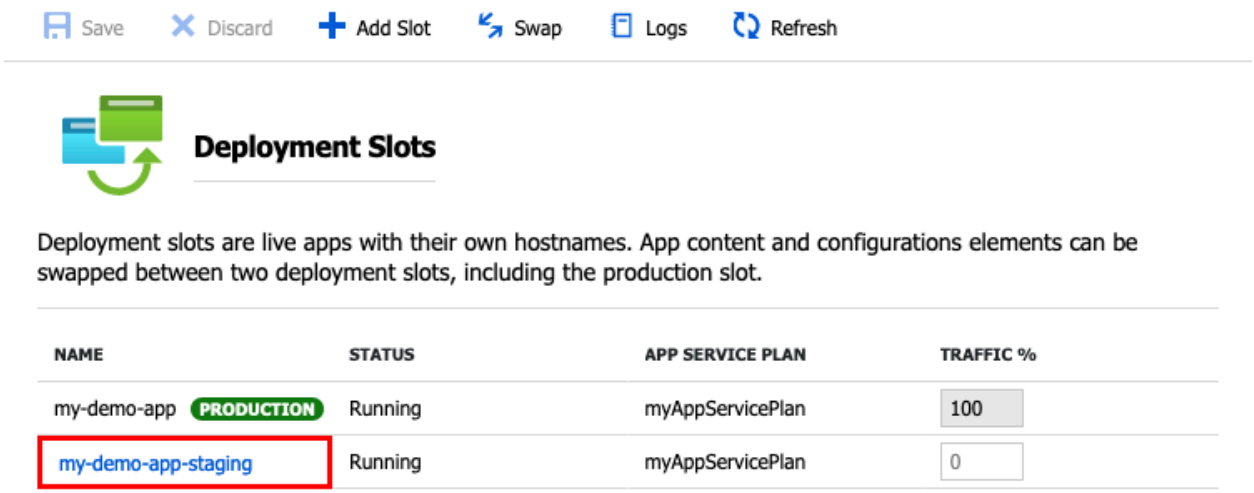
Add a Staging Slot

1. In the Azure portal, navigate to your app's management page.
2. In the left pane, select Deployment slots > Add Slot.
   Note: If the app isn't already in the Standard, Premium, or Isolated tier, select Upgrade and go to the Scale tab of your app before continuing.
3. In the Add a slot dialog box, give the slot a name, and select whether to clone an app configuration from another deployment slot. Select Add to continue.



You can clone a configuration from any existing slot. Settings that can be cloned include app settings, connection strings, language framework versions, web sockets, HTTP version, and platform bitness.

4. After the slot is added, select Close to close the dialog box. The new slot is now shown on the Deployment slots page. By default, Traffic % is set to 0 for the new slot, with all customer traffic routed to the production slot.
5. Select the new deployment slot to open that slot's resource page.

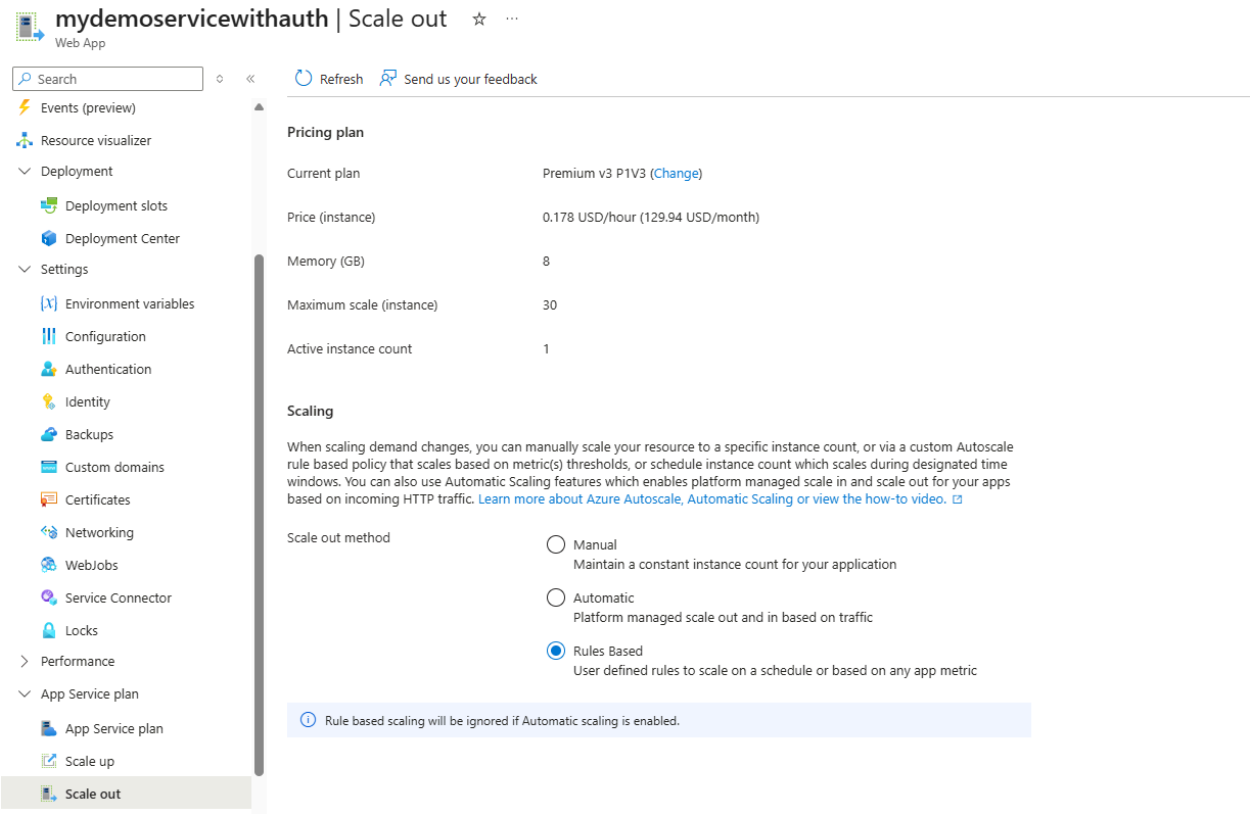


The staging slot has a management page just like any other App Service app. You can change the slot's configuration. To remind you that you're viewing the deployment slot, the app name is shown as <app-name>/<slot-name>, and the app type is App Service (Slot). You can also see the slot as a separate app in your resource group, with the same designations.

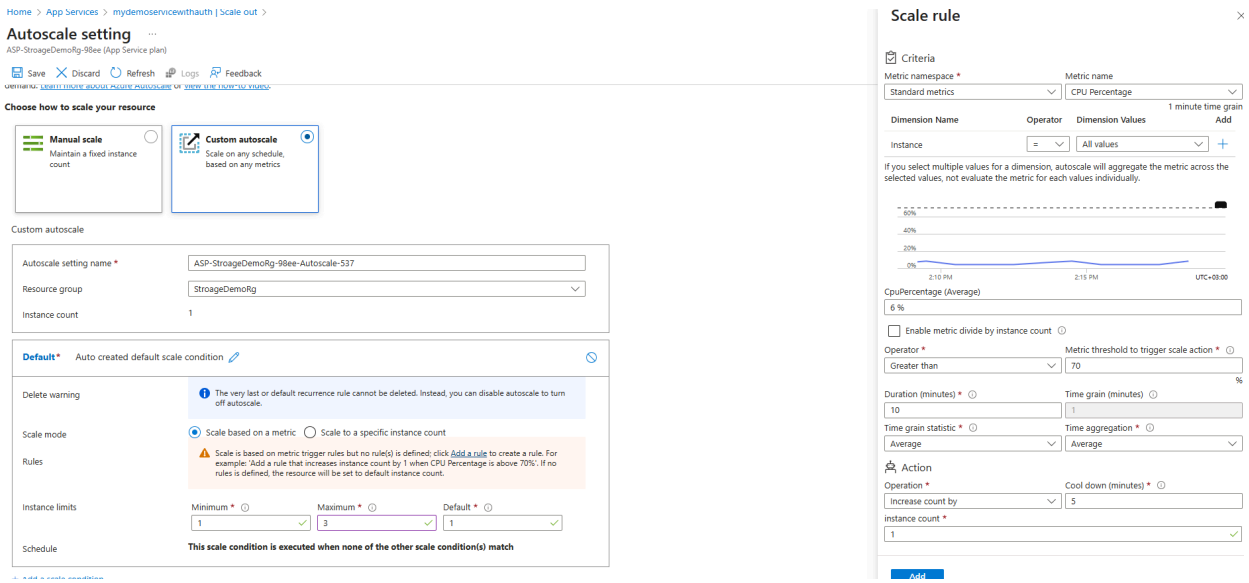
6. Select the app URL on the slot's resource page. The deployment slot has its own host name and is also a live app. To limit public access to the deployment slot, see Azure App Service IP restrictions.

Create autoscale setting

1. Open the Azure portal
2. Go to https://portal.azure.com
3. Navigate to your Web App
    a. In the search bar, type App Services.
    b. Select your Web App from the list.



4. Open Scale-out (App Service plan)
    a. In the left menu of the Web App, under Settings, select Scale out (App Service plan).
    b. This takes you to the autoscale settings for the App Service plan that hosts your web app.
5. Choose Scaling Option
    a. By default, you will see two options:
        i. Manual scale – fixed number of instances.
        ii. Custom autoscale – scale based on rules.
    b. Select Custom autoscale.
6. Define Autoscale Settings
    a. Autoscale setting name: Enter a descriptive name (e.g., WebApp-Autoscale).
    b. Resource group: Select the resource group for the autoscale setting.
    c. Target resource: Confirm it is pointing to your App Service plan.
    d. Scale mode: Choose Scale based on a metric.



7. Configure Scaling Rules
    a. Click Add a rule.
    b. Choose a metric (e.g., CPU Percentage, Memory Usage, HTTP Queue Length).
    c. Set the condition (e.g., CPU > 70% for 10 minutes).
    d. Define the scale action (e.g., Increase count by 1).
    e. Optionally, add a scale-in rule (e.g., CPU < 30% for 10 minutes → Decrease count by 1).
8. Set Instance Limits
    a. Define minimum, maximum, and default instance counts.
    b. Example: Min = 1, Max = 10, Default = 2.
9. (Optional) Add a Schedule
    a. You can create profiles with different scaling rules based on time of day or day of week (e.g., higher capacity during business hours).
10. Review and Create
    a. Review your autoscale configuration.
    b. Select Save (or Create if starting from scratch).
11. Verify
    a. Your web app will now automatically scale in or out based on the defined rules.

Deploy a web app

Install

- The Azure Tools extension in VS Code
- The latest https://dotnet.microsoft.com/en-us/download/dotnet/8.0

Create an ASP.NET web app

1. Open a terminal window on your machine to a working directory. Create a new .NET web app using the dotnet new webapp command, and then change directories into the newly created app.

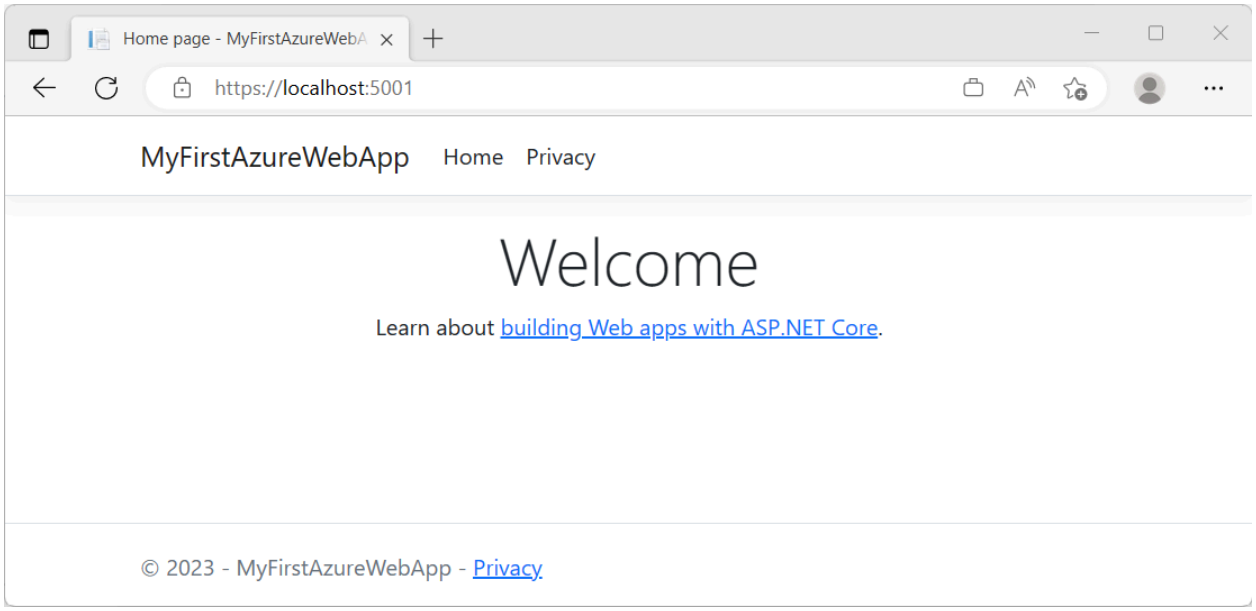
```
dotnet new webapp -n MyFirstAzureWebApp --framework net8.0
cd MyFirstAzureWebApp
```

2. From the same terminal session, run the application locally using the dotnet run command.

```
dotnet run --urls=https://localhost:5001/
```

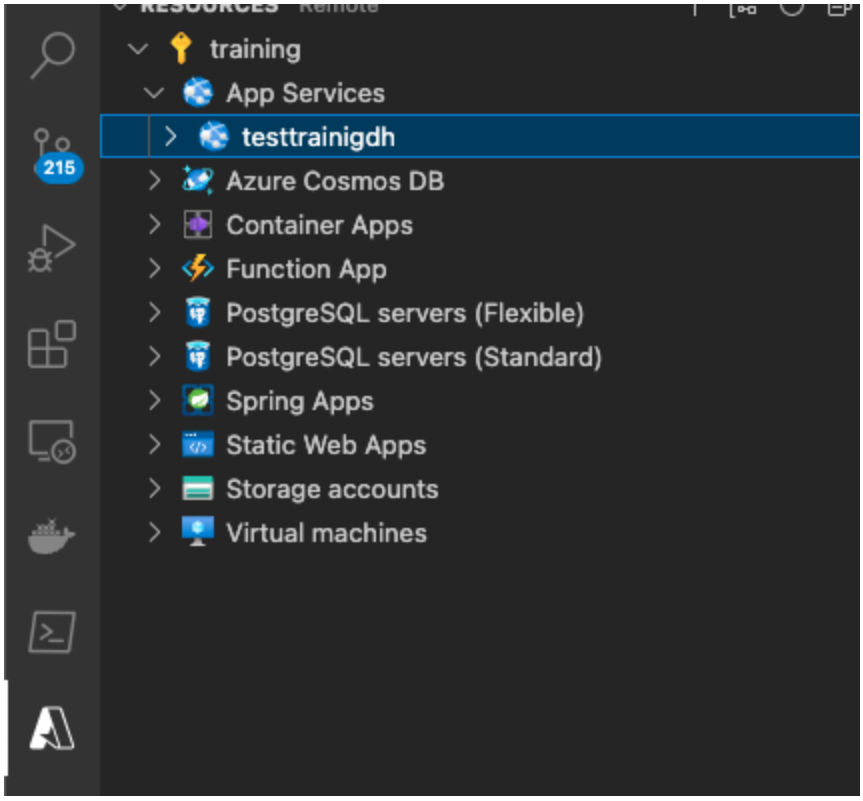
3. Open a web browser, and navigate to the app at https://localhost:5001.

   You see the template ASP.NET Core 8.0 web app displayed on the page.



Publish your web app

1. Click on the Azure extension
2. Log in with your account
3. Select training subscription
4. Find your app service



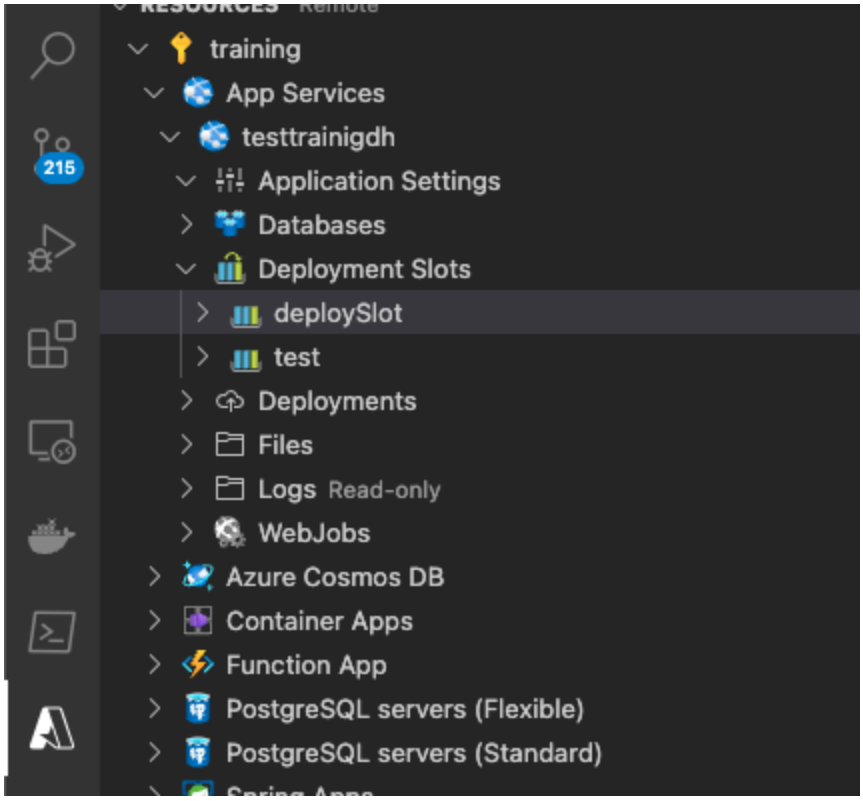
5. Right-click on the app service icon
6. Click on deploy to the web app
7. Open your app

Publish your web app to a slot

1. Open **Pages/Index.cshtml.**
2. Replace the first element with the following code:

```
<div class="jumbotron">
    <h1>.NET 💜 Azure</h1>
    <p class="lead">Example .NET app to Azure App Service.</p>
</div>
```

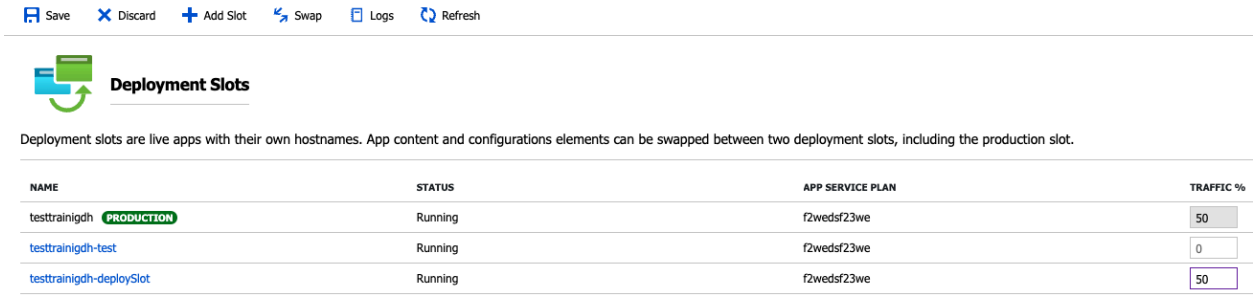
3. Save your changes.
4. In Visual Studio Code
5. Expand on the app service icon
6. Select a deploy slot
7. Right-click on the slot and select Deploy to slot



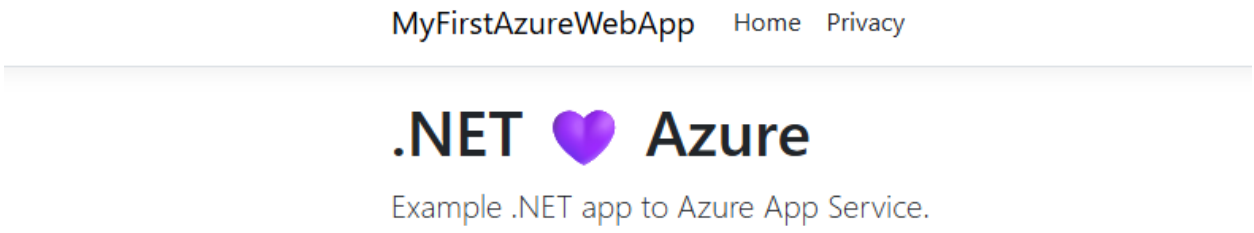
8. Open your slot app

Configure incoming traffic to the web app

1. Open the portal
2. Open your web app
3. Select deployment slots
4. Modify the traffic percentage and save



5. Make sure you open pages on an incognito tab, as deployment slots have session affinity



Note: If the deployment fails set the following variables to the app service slot:

- ENABLE_ORYX_BUILD : false
- SCM_DO_BUILD_DURING_DEPLOYMENT : false