

## IIB43203 ASSESSMENT COVERSHEET

Attach this coversheet as the cover of your submission. All sections must be completed.

### Section A: Submission Details

<b>Programme</b>	:	BACHELOR IN INFORMATION TECHNOLOGY (HONS) (INTERNET OF THINGS)
<b>Course Code &amp; Name</b>	:	IIB43203 – CLOUD COMPUTING
<b>Assessment</b>	:	<b>GROUP PROJECT ASSIGNMENT</b>
<b>Dateline</b>	:	<b>3/6/2025</b>
<b>Penalties</b>	:	<ul style="list-style-type: none"><li>• <b>10%</b> will be deducted for late submission.</li><li>• Plagiarised work is an Academic Offence in University Rules &amp; Regulations and will be penalised accordingly.</li><li>• <b>AI content</b> will affect your mark</li></ul>

### Section B: Academic Integrity

Tick (✓) each box below if you agree:

- I have read and understood the UniKL's policy on Plagiarism in University Rules & Regulations.  
This submission is my own, unless indicated with proper referencing.  
This submission has not been previously submitted or published.  
This submission follows the requirements stated in the course.

### Section C: Submission Detail

#### **Office Receipt of Submission**

No	Student Name(s)	Student ID(s)
1.	NUR AISHAH KAMALIAH BINTI MEZLAN	52224123034
2.	SHARIFAH NUR QISTINA NABILA BINTI SYED ABD RANI	52224123537
3.	SOLIHAH NAFISATUL 'ILMI BINTI MUHAMMAD NAZRI	52224123588
4.	NUR HUDA BATRISYIA BINTI HARMIZI	52224123502

---

#### **Student Receipt of Submission**

The submission receipt of the student task will be based on the **VLE submission date and time**.

**Jobscope**

Name	Lab 30A	Lab 30B	Lab 31	Lab 32	Report	Video
NUR AISHAH KAMALIAH BINTI MEZLAN	/	/			/	
SHARIFAH NUR QISTINA NABILA BINTI SYED ABD RANI			/	/	/	
SOLIHAH NAFISATUL ‘ILMI BINTI MUHAMMAD NAZRI	/				/	/
NUR HUDA BATRISYIA BINTI HARMIZI			/		/	/

## Lab 30A: Create Deployment Slots

A deployment slot serves as a separate environment for a web application, allowing new versions to be tested before being released to the live site. By swapping slots, updates can be deployed seamlessly without causing downtime. For instance, if an organization chooses to use Azure App Service deployment slots to streamline the update process for its social media web application, the next step would be to configure the application in Azure and set up the necessary deployment slots.

### Create a web app

1. On the Azure portal menu or from the Home page, select Create a resource.

The screenshot shows the Microsoft Azure Home page. At the top, there's a search bar and several navigation icons. Below the search bar, there's a section titled 'Azure services' with a 'Create a resource' button highlighted with a red box. Underneath, there's a 'Resources' section showing a single item: 'Azure subscription 1' (Type: Subscription, Last Viewed: a week ago). At the bottom, there are sections for 'Navigate' and 'Tools'.

2. From the left-hand menu, click on Web, then search for and choose Web App.

The screenshot shows the 'Create a resource' page. On the left, there's a sidebar with 'Get Started' and 'Recently created'. Below that is a 'Categories' section with various service options like AI + Machine Learning, Analytics, Blockchain, Compute, Containers, Databases, Developer Tools, DevOps, Identity, Integration, Internet of Things, IT & Management Tools, and Media. In the center, there's a search bar and a 'Popular Azure services' section with links to Function App, Azure AI services, Web App (which is highlighted with a red box), Azure OpenAI, Virtual network, SQL Database, and Key Vault. To the right, there's a 'Popular Marketplace products' section with links to Windows Server 2022 Datacenter, Windows 10 Pro, Ubuntu Server 22.04 LTS, Free SQL Server License, Red Hat Enterprise Linux, Debian 12, Visual Studio 2022 Pro, and Microsoft 365 Apps. At the bottom right, there's a 'Give feedback' link.

3. On the Basics tab, enter the following values for each setting.

	Setting	Value
Project Details	Subscription	Select the subscription you'd like to use to complete the exercise
	Resource Group	Create a new resource group named mslearn-slots
Instance Details	Name	Enter a unique name
	Publish	Code
	Runtime stack	ASP.NET V4.8
Operating System Windows	Region	Select a region near you
App Service Plan	Windows Plan	Accept default
	Pricing Plan	Accept default

The screenshot shows the 'Create Web App' wizard in the Microsoft Azure portal. The 'Basics' step is selected. The form fields are as follows:

- Subscription \***: Azure subscription 1
- Resource Group \***: (New) mslearn-slots
- Instance Details**:
  - Name**: hungryrose
  - Publish**: Code
  - Runtime stack**: ASP.NET V4.8
  - Operating System \***: Windows
  - Region**: Loading...
- Pricing plans**:
  - Windows Plan**: (New) ASP-mslearn-slots-84f7
  - Pricing plan**: Standard S1 (100 total ACU, 1.75 GB memory, 1 vCPU)
- Zone redundancy**:
  - Enabled**: Your App Service plan and the apps in it will be zone redundant. The minimum App Service plan instance count will be two.
  - Disabled**: Your App Service plan and the apps in it will not be zone redundant. The minimum App Service plan instance count will be one.

At the bottom, there are navigation buttons: 'Review + create' (highlighted in blue), '< Previous', and 'Next: Database >'.

4. Select Next: Deployment and select Next: Networking. Do not change anything.

5. Select Next: Monitoring, and enter the following value for the setting.

	Setting	Value
Application Insights	Enable Application Insights	No

The screenshot shows the 'Create Web App' wizard in the Microsoft Azure portal. The current step is 'Monitor + secure'. Under 'Application Insights', the 'Enable Application Insights' checkbox is unchecked. Under 'Microsoft Defender for Cloud', the 'Enable Defender for App Service' checkbox is also unchecked. At the bottom, there are buttons for 'Review + create' and 'Next : Tags >'. The URL in the address bar is [https://portal.azure.com/#create/Microsoft.Web/applications](#).

6. Select Review + Create and wait for it to be validated.

The screenshot shows the 'Review + create' step of the 'Create Web App' wizard. The validation status is 'Validating'. The summary section shows a 'Web App' by Microsoft with a 'Standard (S1) sku'. A note states: 'Basic authentication for this app is currently disabled and may impact deployments. Click to learn more.' At the bottom, there are buttons for 'Create', '< Previous', 'Next >', and 'Download a template for automation'. The URL in the address bar is [https://portal.azure.com/#create/Microsoft.Web/applications](#).

- Once the information is validated, click **Create** and wait for Azure to finish setting up the web app.

The screenshot shows the Microsoft Azure Overview page for a deployment named "Microsoft.Web-WebApp-Portal-33f21b7f-82e9". The deployment status is "Deployment is in progress". The resource listed is "ASP-mslearnslots-84f7", which is a Microsoft.Web/serverfarms type and currently "OK". The deployment started at 5/29/2025, 7:15:48 AM. A sidebar on the right provides links to Microsoft Defender for Cloud, Microsoft tutorials, and expert support.

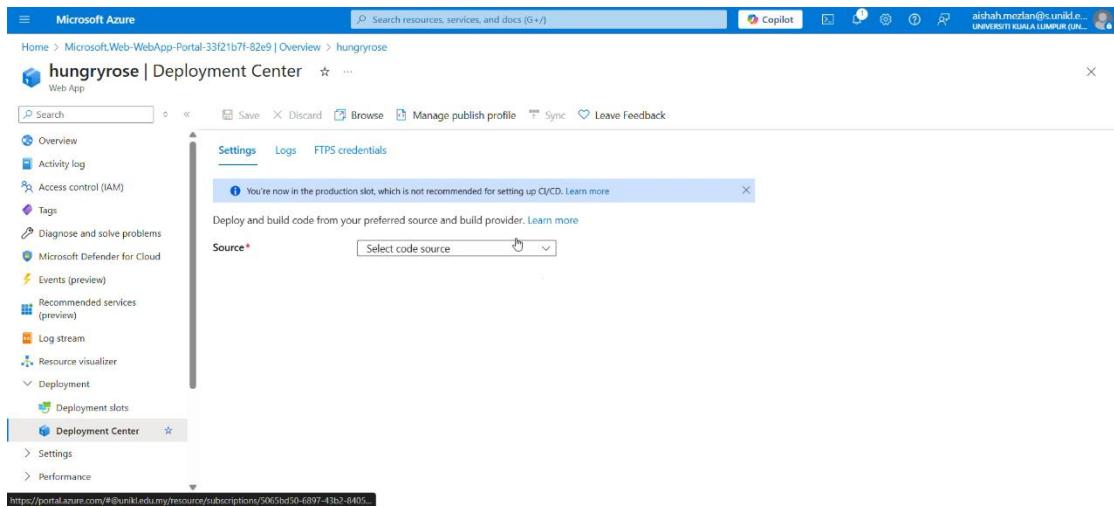
- When deployment successfully completes, select Go to resource. The App Service pane appears.

The screenshot shows the Microsoft Azure App Service Overview page for a web app named "hungryrose". The "Essentials" section displays basic details: Resource group (move) is "mslearn-slots", Status is "Running", Location (move) is "Canada Central", Subscription (move) is "Azure subscription 1", and Subscription ID is "5065bd50-6897-43b2-8405-323c9cadff0". The "Properties" tab is selected, showing the Web app settings: Name is "hungryrose", Publishing model is "Code", and Domains include "hungryrose.azurewebsites.net" (Default domain) and "Add custom domain". The "Deployment Center" shows deployment logs and last deployment status. The "Networking" section lists the Virtual IP address as "20.48.204.12" and Outbound IP addresses as "130.107.180.169, 130.107.180.207".

## Configure git deployment

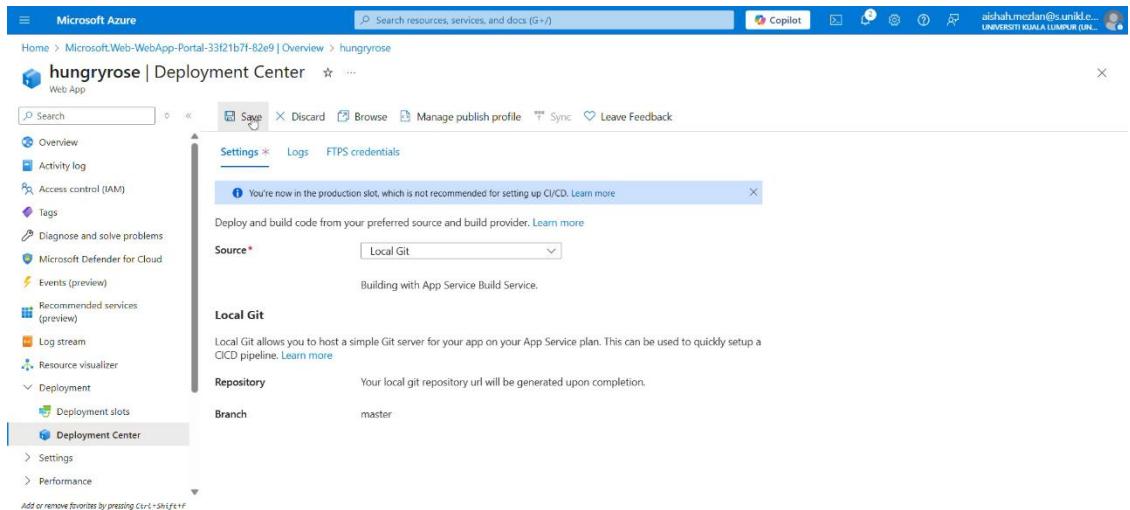
Any standard deployment tools can be used to deploy a web application and manage its deployment slots. In this section, the web application will be configured to utilize a local Git repository for deployment purposes. The following steps outline the setup process.

1. On your web app App Service page, in the left menu, under Deployment, select Deployment Center.



The screenshot shows the Microsoft Azure Deployment Center settings page for a web app named "hungryrose". The left sidebar shows various service links like Overview, Activity log, and Deployment slots. The main pane is titled "Settings" and displays a message: "You're now in the production slot, which is not recommended for setting up CI/CD. Learn more". Below this, there's a "Source\*" dropdown menu with "Select code source" selected. The URL at the bottom is <https://portal.azure.com/#@unikl.edu.my/resource/subscriptions/600bd0-6897-4fb7-8405...>.

2. On the Settings tab, for Source, select Local Git. Make sure to enable your SCM Basic Authentication. On the top menu bar, select Save.



The screenshot shows the Microsoft Azure Deployment Center settings page for the same web app "hungryrose". The "Source\*" dropdown now has "Local Git" selected. The "Local Git" section is expanded, showing a note: "Local Git allows you to host a simple Git server for your app on your App Service plan. This can be used to quickly setup a CI/CD pipeline. Learn more". It also shows a "Repository" field with a placeholder "Your local git repository url will be generated upon completion." and a "Branch" field set to "master". The top menu bar includes a "Save" button, which is highlighted in red in the screenshot. The URL at the bottom is <https://portal.azure.com/#@unikl.edu.my/resource/subscriptions/600bd0-6897-4fb7-8405...>.

3. On the resulting Deployment Center pane, select the Local Git/FTPS credentials tab.

The screenshot shows the Microsoft Azure Deployment Center for a Web App named "hungryrose". The "Local Git/FTPS credentials" tab is active. The "Application scope" section is visible, containing fields for "FTPS endpoint" (set to `https://waws-prod-xt1-091.ftp.azurewebsites.windows.net/site/wwwroot`), "Git Clone Uri" (set to `https://hungryrose.scm.azurewebsites.net:443/hungryrose.git`), "FTPS Username" (set to `hungryrose\$hungryrose`), "Local Git Username" (set to `\$hungryrose`), and "Password" (a masked password). Below these, the "User scope" section is shown, which is currently empty.

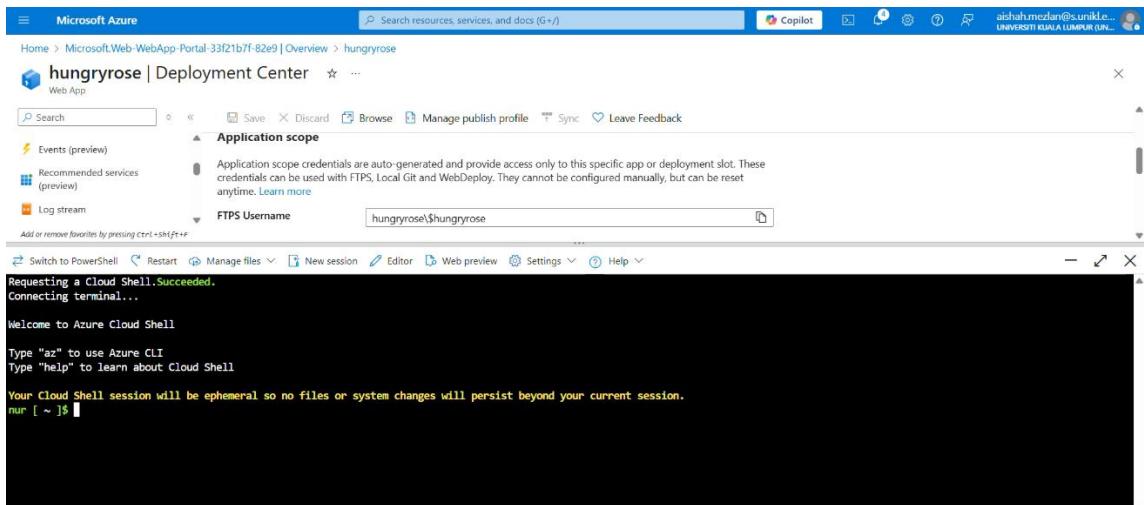
4. In the User Scope section, enter a username and password of your choice. Then, click Save in the top menu bar. Be sure to note down the username and password for future use.

The screenshot shows the Microsoft Azure Deployment Center for a Web App named "hungryrose". The "User scope" section is now populated with credentials. The "Username" field contains "aishahmezan", the "Password" field contains a masked password, and the "Confirm Password" field also contains a masked password. The rest of the interface remains the same as the previous screenshot.

## Configure the git client and clone the web app source code

Next, the Git client will be configured within the Cloud Shell environment and used to clone a sample web application. The steps outlined below will guide this process.

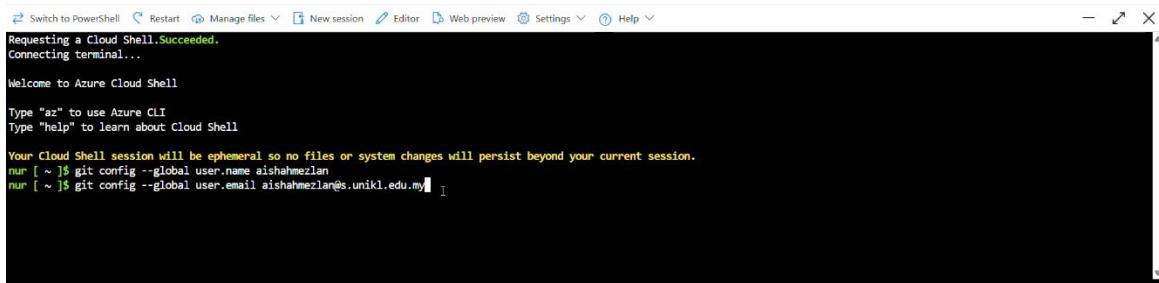
1. In the Azure portal, open Azure Cloud Shell by selecting its icon on the top toolbar.



2. In the Cloud Shell toolbar, ensure that Bash is selected. Then, copy the code provided below into a Notepad file and replace the quoted values with a preferred username and email address. These configuration details are not associated with any specific Azure account or subscription, so any chosen values may be used.

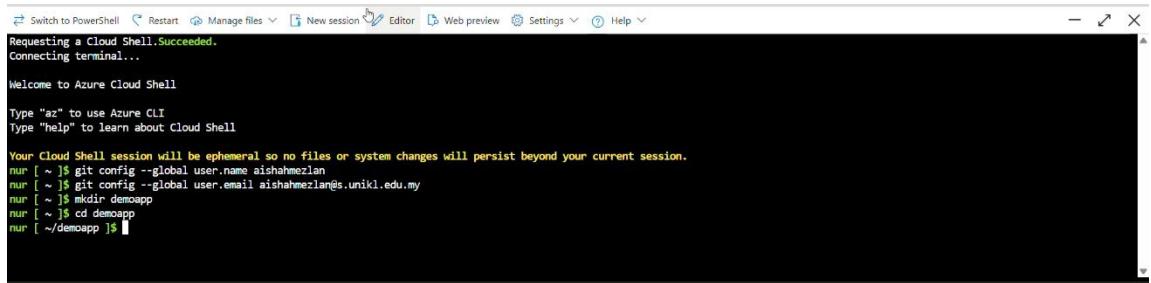
```
git config --global user.name "your-username"  
git config --global user.email "your-email-address"
```

3. Copy and paste your edited code into the Cloud Shell and run it.



4. Create a folder for the source code. Run the following commands.

```
mkdir demoapp  
cd demoapp
```

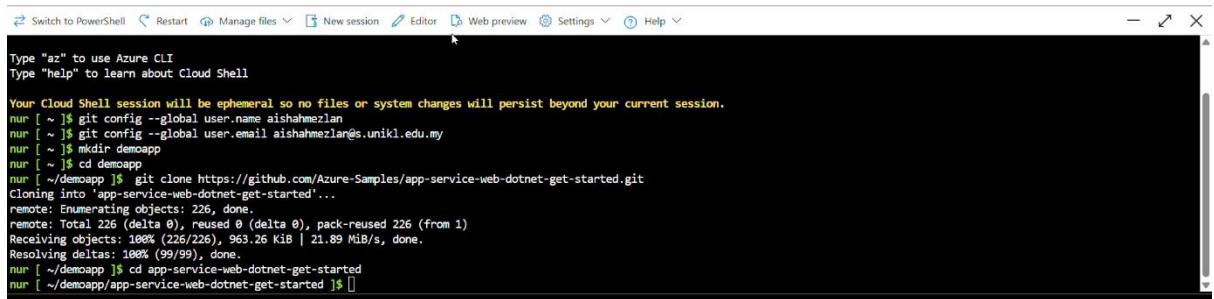


The screenshot shows the Azure Cloud Shell interface. At the top, there are navigation links: 'Switch to PowerShell', 'Restart', 'Manage files', 'New session', 'Editor', 'Web preview', 'Settings', and 'Help'. Below this is a status bar with the message 'Requesting a Cloud Shell. Succeeded.' and 'Connecting terminal...'. The main area is a dark terminal window. It displays the following command history:

```
Welcome to Azure Cloud Shell  
Type "az" to use Azure CLI  
Type "help" to learn about Cloud Shell  
Your Cloud Shell session will be ephemeral so no files or system changes will persist beyond your current session.  
nur [ ~ ]$ git config --global user.name aishahmezlan  
nur [ ~ ]$ git config --global user.email aishahmezlan@s.unikl.edu.my  
nur [ ~ ]$ mkdir demoapp  
nur [ ~ ]$ cd demoapp  
nur [ ~/demoapp ]$
```

5. Clone the source for the web app. Run the following commands.

```
git clone https://github.com/Azure-Samples/app-service-web-dotnet-get-started.git  
cd app-service-web-dotnet-get-started
```



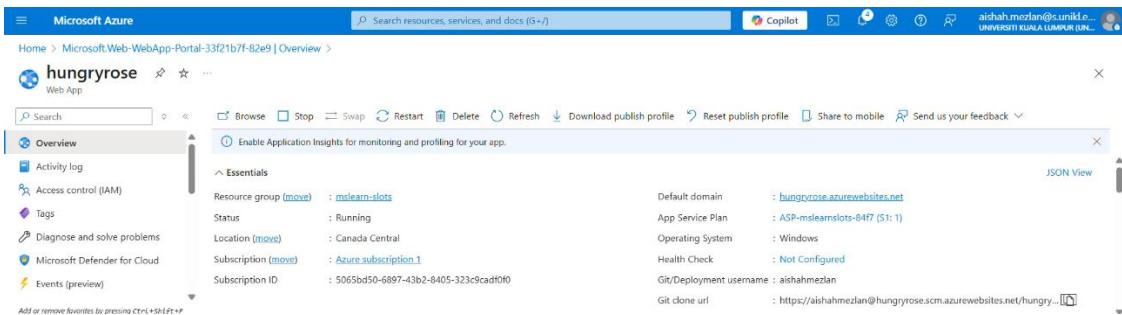
The screenshot shows the Azure Cloud Shell interface. At the top, there are navigation links: 'Switch to PowerShell', 'Restart', 'Manage files', 'New session', 'Editor', 'Web preview', 'Settings', and 'Help'. Below this is a status bar with the message 'Type "az" to use Azure CLI' and 'Type "help" to learn about Cloud Shell'. The main area is a dark terminal window. It displays the following command history and output of the 'git clone' command:

```
Type "az" to use Azure CLI  
Type "help" to learn about Cloud Shell  
Your Cloud Shell session will be ephemeral so no files or system changes will persist beyond your current session.  
nur [ ~ ]$ git config --global user.name aishahmezlan  
nur [ ~ ]$ git config --global user.email aishahmezlan@s.unikl.edu.my  
nur [ ~ ]$ mkdir demoapp  
nur [ ~ ]$ cd demoapp  
nur [ ~/demoapp ]$ git clone https://github.com/Azure-Samples/app-service-web-dotnet-get-started.git  
Cloning into 'app-service-web-dotnet-get-started'...  
remote: Enumerating objects: 226, done.  
remote: Total 226 (delta 0), reused 0 (delta 0), pack-reused 226 (from 1)  
Receiving objects: 100% (226/226), 963.26 KiB | 21.89 MiB/s, done.  
Resolving deltas: 100% (99/99), done.  
nur [ ~/demoapp ]$ cd app-service-web-dotnet-get-started  
nur [ ~/demoapp/app-service-web-dotnet-get-started ]$
```

## Configure a git remote to deploy the app to production.

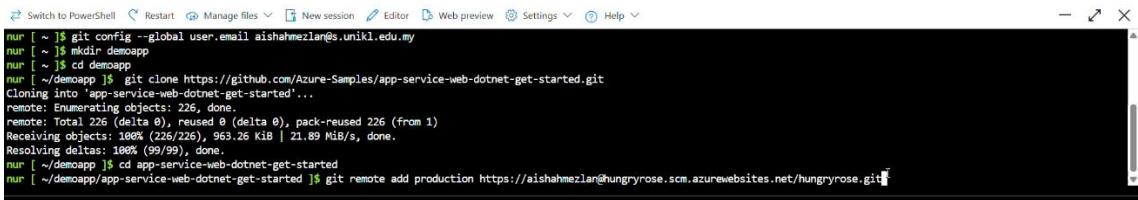
To deploy the source code to the production slot of the web app using Git, it is necessary to configure the app's Git URL as a remote repository. The steps below outline how to complete this setup.

1. In the Azure portal, make sure your web app is running. From the left-hand menu, click on Overview.
2. In the Overview pane, locate the Essentials section. Here, you'll find the Default domain and the Git clone URL. Hover over the Default domain and click the Copy to clipboard icon. This domain includes the deployment name of your web app.



3. Hover over the Git clone URL and click the Copy to clipboard icon. Keep in mind that this URL also includes your deployment username. (Refresh if it does not appear).
4. In Cloud Shell, run the following command to add the Git remote named production. Replace `git-clone-url` with the URL you copied in the previous step.

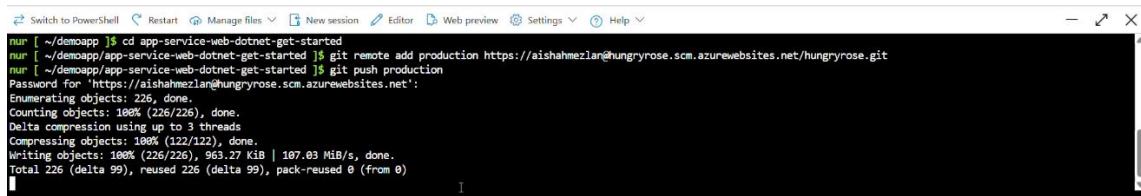
```
cd app-service-web-dotnet-get-started  
git remote add production <git-clone-url>
```



```
[~] $ git config --global user.email aishahmezlan@s.unikl.edu.my  
[~] $ mkdir demapp  
[~] $ cd demapp  
[~/demapp] $ git clone https://github.com/Azure-Samples/app-service-web-dotnet-get-started.git  
Cloning into 'app-service-web-dotnet-get-started'...  
remote: Enumerating objects: 226, done.  
remote: Total 226 (delta 0), reused 0 (delta 0), pack-reused 226 (from 1)  
Receiving objects: 100% (226/226), 963.26 KiB | 21.89 MiB/s, done.  
Resolving deltas: 100% (99/99), done.  
[~/demapp] $ cd app-service-web-dotnet-get-started  
[~/demapp/app-service-web-dotnet-get-started] $ git remote add production https://aishahmezlan@hungryrose.scm.azurewebsites.net/hungryrose.git
```

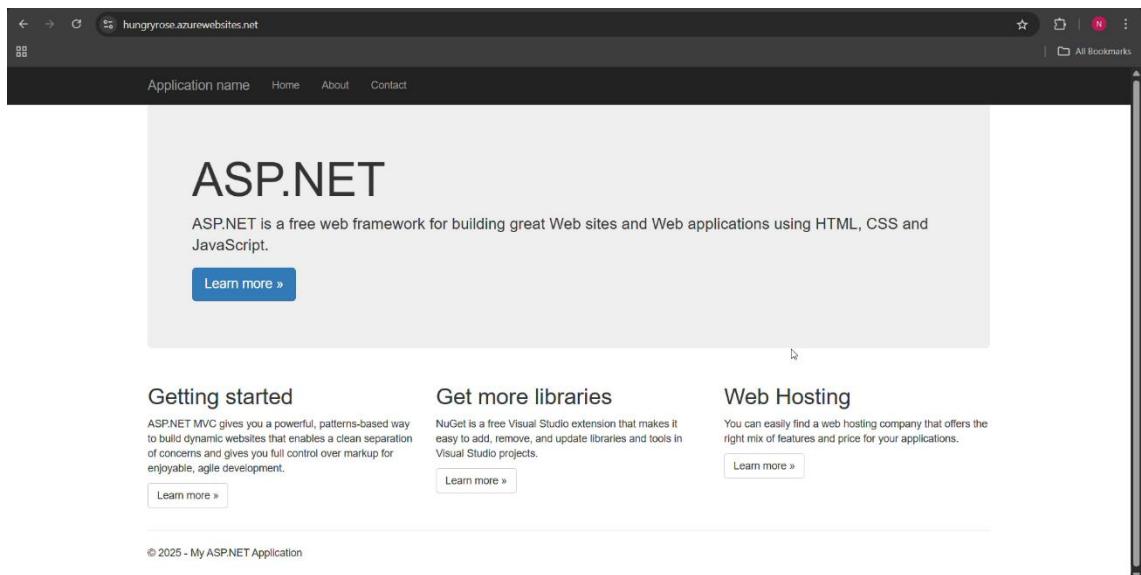
- To deploy the web app to the production slot, run the following command. When you're prompted for the password, enter your deployment password you created previously.

```
git push production
```



```
Switch to PowerShell ⌘ Restart Manage files New session Editor Web preview Settings Help
run [ ~/demapp ]$ cd app-service-web-dotnet-get-started
run [ ~/demapp/app-service-web-dotnet-get-started ]$ git remote add production https://aishahmezlan@hungryrose.scm.azurewebsites.net/hungryrose.git
run [ ~/demapp/app-service-web-dotnet-get-started ]$ git push production
Password for 'https://aishahmezlan@hungryrose.scm.azurewebsites.net':
Enumerating objects: 226, done.
Counting objects: 100% (226/226), done.
Delta compression using up to 3 threads
Compressing objects: 100% (122/122), done.
Writing objects: 100% (226/226), 963.27 kB | 107.03 MiB/s, done.
Total 226 (delta 99), reused 226 (delta 99), pack-reused 0 (from 0)
```

- When the deployment finishes, in the Azure portal, go to the web app's Overview page, and then select Default domain. You can paste it into a browser or double-click to open the URL in a new tab.



- Close the browser tab that displays the web app.

## Create a new staging slot

The newly created web app currently includes only a single slot, which is the production slot where the initial source code has been deployed. The next step involves creating an additional deployment slot to stage and test new versions of the web app before pushing them to production.

1. On the Azure portal menu or from the Home page, select All resources.

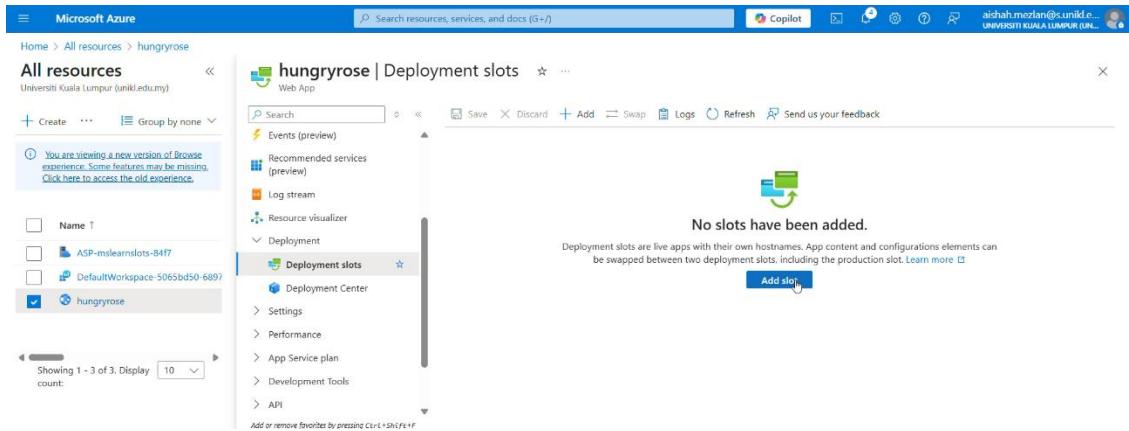
The screenshot shows the Microsoft Azure portal's 'All resources' blade. At the top, there are navigation links like 'Home', 'Create', 'Manage view', 'Refresh', 'Export to CSV', 'Open query', 'Assign tags', and 'Delete'. Below these are filter options: 'Subscription equals all', 'Resource Group equals all', 'Type equals all', 'Location equals all', and 'Add filter'. The main table lists resources with columns for Name, Type, Resource Group, Location, and Subscription. The 'hungryrose' web app is selected, indicated by a checked checkbox in the first column. The table shows three rows of data.

Name	Type	Resource Group	Location	Subscription
ASP-mslearnslots-84f7	App Service plan	mslearn-slots	Canada Central	Azure subscription 1
DefaultWorkspace-5065bd50-6897-43b2-8405-323c9cadff00-C...	Log Analytics workspace	DefaultResourceGroup-CCAN	Canada Central	Azure subscription 1
<b>hungryrose</b>	App Service	mslearn-slots	Canada Central	Azure subscription 1

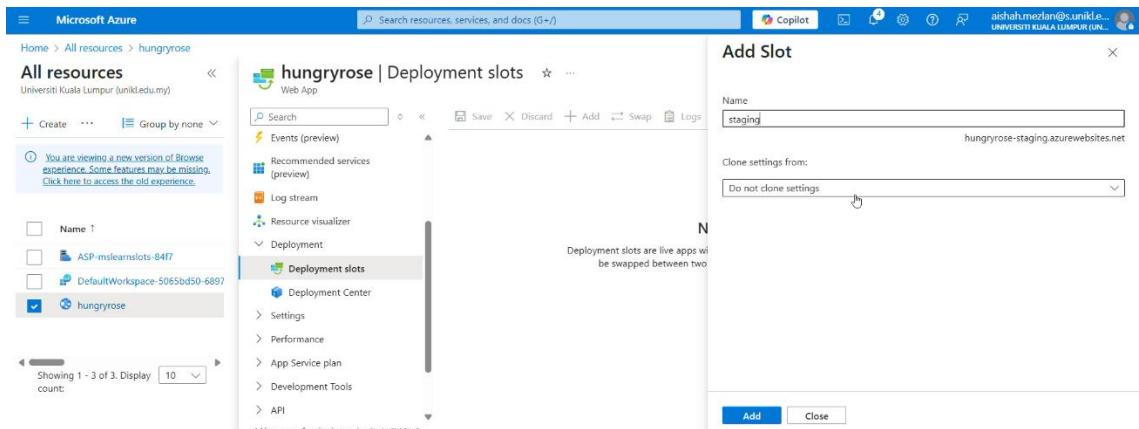
2. Select your web app. The web app App Service pane appears.

The screenshot shows the Microsoft Azure portal's 'hungryrose' App Service blade. The left sidebar lists 'Events (preview)', 'Recommended services (preview)', 'Log stream', 'Resource visualizer', 'Deployment', 'Deployment slots', 'Deployment Center', 'Settings', 'Performance', 'App Service plan', 'Development Tools', and 'API'. The 'Settings' item is currently selected. The right pane displays the 'Essentials' section with details such as Resource group (mslearn-slots), Status (Running), Location (Canada Central), Subscription (Azure subscription 1), Subscription ID (5065bd50-6897-43b2-8405-323c9cadff00), Default domain (hungryrose.azurewebsites.net), App Service Plan (ASP-mslearnslots-84f7 (S1: 1)), Operating System (Windows), Health Check (Not Configured), Git/Deployment username (aishahmezlan), and Git clone url (https://aishahmezlan@hungryrose.scm.azurewebsites.net/hungry...).

- In the menu pane, under Deployment, select Deployment slots. The Deployment slots pane appears.



- Select Add Slot, the Add a slot pane appears.
- In the Name field, enter staging, accept the default for Clone settings from, and then select Add.

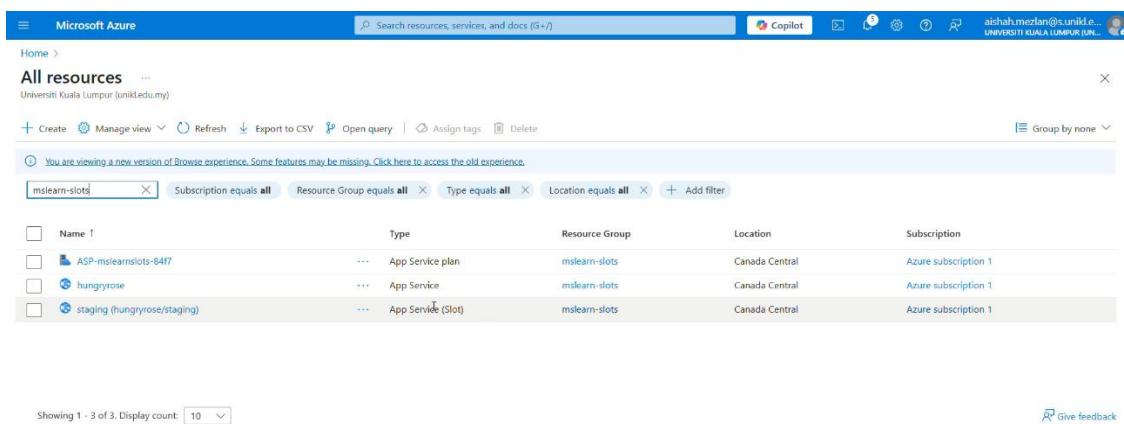


- After the deployment slot is successfully created, select Close.

## Set up git deployment for the staging slot

Configure the newly created deployment slot to use Git for deploying source code, similar to how the production slot was set up. Follow the steps below to complete the Git deployment configuration for this slot.

1. In the Azure portal menu or on the Home page, select All resources. In the list of resources, filter by Resource group set to mslearn-slots. You will see two App Service entries—deployment slots appear as separate apps in the portal. Click on the entry for the staging slot to open its Overview pane.

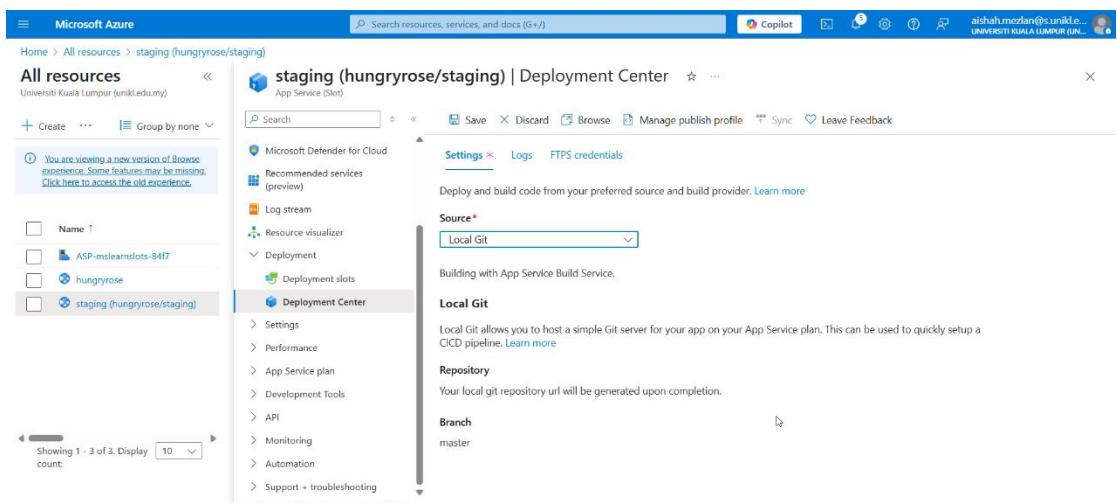


The screenshot shows the Microsoft Azure 'All resources' page. A search bar at the top contains 'mslearn-slots'. Below it, a table lists three resources:

Name	Type	Resource Group	Location	Subscription
ASP-mslearnslots-8417	App Service plan	mslearn-slots	Canada Central	Azure subscription 1
hungryrose	App Service	mslearn-slots	Canada Central	Azure subscription 1
staging (hungryrose/staging)	App Service (Slot)	mslearn-slots	Canada Central	Azure subscription 1

At the bottom of the table, it says 'Showing 1 - 3 of 3. Display count: 10'.

2. On the Overview pane, in the left menu, under Deployment, select Deployment Center.
3. On the Settings tab, for Source, select Local Git. In the top menu bar, select Save.



The screenshot shows the 'Deployment Center' settings for the 'staging (hungryrose/staging)' slot. The 'Source' dropdown is set to 'Local Git'. Other options shown include 'Microsoft Defender for Cloud', 'Recommended services (preview)', 'Log stream', 'Resource visualizer', 'Deployment', and 'Deployment slots'. The 'Save' button is visible in the top right corner.

4. On the resulting Deployment Center pane, select the Local Git/FTPS credentials tab.
5. Under User scope, enter a new username and password of your choice, and from the top menu bar, select Save. Make a note of the username and password for later.

The screenshot shows the Microsoft Azure Deployment Center pane for the 'staging (hungryrose/staging)' App Service Slot. The 'Local Git Username' field contains '\$hungryrose\_staging'. In the 'User scope' section, the 'Username' field is set to 'kamallahmezan' and the 'Password' field contains a masked password. The 'Confirm Password' field also contains a masked password. A tooltip for 'User scope' explains that user scope credentials are defined by the user and can be used with all apps. The 'Deployment Center' tab is selected in the sidebar.

## Set up git to deploy the app to the staging slot

To deploy the source code to the new deployment slot using the Git client, an extra remote must be added to the existing Git configuration. Follow the instructions below to complete this setup.

1. In the Azure portal, on the staging web app's Overview page, in the Essentials section, copy the Git clone url. Note that the URL contains your deployment username. (Refresh if it does not appear)

The screenshot shows the Azure portal's Overview page for a specific app service slot. The 'Essentials' section provides key deployment information:

- Resource group: mslearn-slots
- Status: Running
- Location: Canada Central
- Subscription: Azure subscription 1
- Subscription ID: 5065bd50-6897-43b2-8405-323c9cadff0f
- Default domain: hungryrose-staging.azurewebsites.net
- App Service Plan: ASP-mslearn-slots-8417 (\$1: 1)
- Operating System: Windows
- Health Check: Not Configured
- Git/Deployment username: kamaliahmezan
- Git clone url: https://kamaliahmezan@hungryrose-staging.scm.azurewebsites...

2. To add the remote for the staging slot, run the following commands in Cloud Shell. Replace git-clone-uri with the URI from the previous step.

```
cd demoapp
```

```
cd app-service-web-dotnet-get-started
```

```
git remote add staging git-clone-uri
```

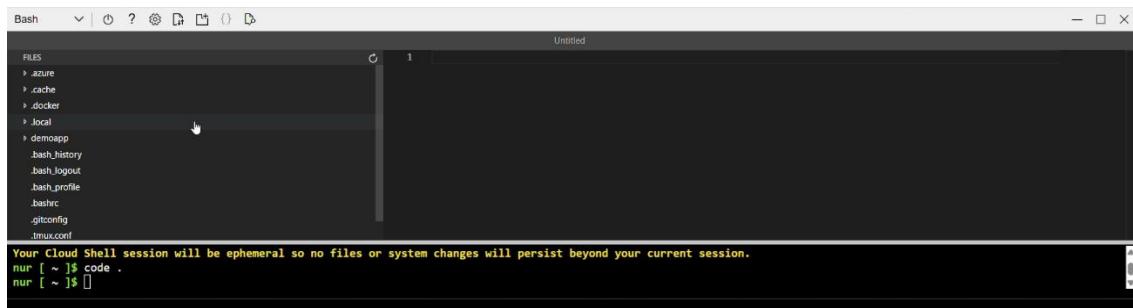
```
Requesting a Cloud Shell.Succeeded.
Connecting terminal...
Your Cloud Shell session will be ephemeral so no files or system changes will persist beyond your current session.
num [ ~ ]$ git remote add staging https://kamaliahmezan@hungryrose-staging.scm.azurewebsites.net/hungryrose.git
fatal: not a git repository (or any of the parent directories): .git
num [ ~ ]$ cd demoapp
num [ ~/demoapp ]$ cd app-service-web-dotnet-get-started
num [ ~/demoapp/app-service-web-dotnet-get-started ]$ git remote add staging https://kamaliahmezan@hungryrose-staging.scm.azurewebsites.net/hungryrose.git
```

## Modify the app source and deploy the app to the staging slot

Next, make a small change to the web app, and then use git to deploy the new version to the staging slot:

1. In Cloud Shell, run the following command.

```
code .
```

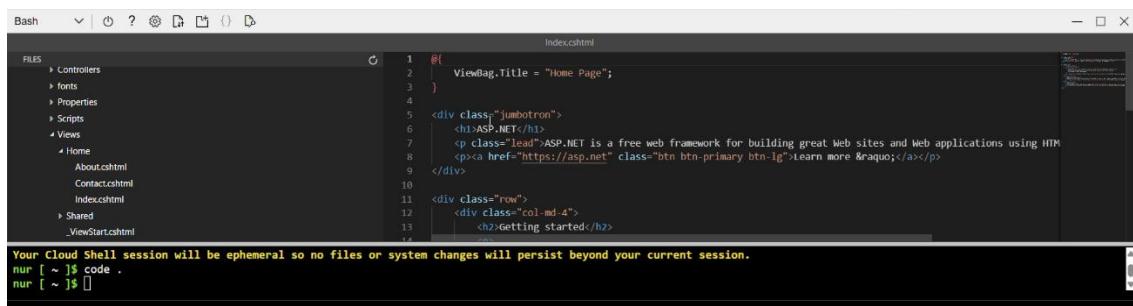


The screenshot shows a Bash terminal window titled "Untitled". The left sidebar displays a file tree with the following structure:

```
FILES
> azure
> .cache
> .docker
> local
> demoapp
  bash.history
  bash.logout
  bash.profile
  bashrc
  .gitconfig
  .tmux.conf
```

The main pane shows a blank terminal session with the message: "Your Cloud Shell session will be ephemeral so no files or system changes will persist beyond your current session." At the bottom, there are two command-line prompts: "nur [ ~ \$ code ." and "nur [ ~ \$ ]".

2. In the list of Files, expand demoapp > app-service-web-dotnet-get-started > aspnet-get-started > Views > Home.
3. Select Index.cshtml.



The screenshot shows the same Bash terminal window. The main pane now displays the content of the "Index.cshtml" file:

```
Index.cshtml
1  @{
2      ViewBag.Title = "Home Page";
3  }
4
5  <div class="jumbotron">
6      <h1>ASP.NET</h1>
7      <p>ASP.NET is a free web framework for building great Web sites and Web applications using HTML<br/><a href="https://asp.net" class="btn btn-primary btn-lg">Learn more &gt;</a></p>
8  </div>
9
10 <div class="row">
11     <div class="col-md-4">
12         <h2>Getting started</h2>
```

At the bottom, the message "Your Cloud Shell session will be ephemeral so no files or system changes will persist beyond your current session." is visible, along with the command-line prompts: "nur [ ~ \$ code ." and "nur [ ~ \$ ]".

4. Locate the following code.

```
<h1>ASP.NET</h1>
```

5. Replace that code with this code.

# Web App Version 2



The screenshot shows a Bash terminal window with the title "Bash". The terminal displays the code for the "Index.cshtml" file. The code includes a ViewBag assignment, a jumbotron div containing a h1 header and a lead paragraph with a link to ASP.NET, and a row div with a col-md-4 column containing a h2 header. The terminal also shows a message about Cloud Shell session being ephemeral and a few command-line inputs.

```
1 @{
2     ViewBag.Title = "Home Page";
3 }
4
5 <div class="jumbotron">
6     <h1>Web App Version 2</h1>
7     <p class="lead">ASP.NET is a free web framework for building great Web sites and Web applications using HTML<br/><a href="https://asp.net" class="btn btn-primary btn-lg">Learn more &gt;</a></p>
8 </div>
9
10 <div class="row">
11     <div class="col-md-4">
12         <h2>Getting started</h2>
13     </div>
14 </div>
```

Your Cloud Shell session will be ephemeral so no files or system changes will persist beyond your current session.  
nur [ ~ \\$ code .  
nur [ ~ \\$ ]

6. To save your changes, press Ctrl+S, and then press Ctrl+Q to close the editor.

In Cloud Shell, execute the following commands to commit the updated version of your app to Git and deploy it to the staging slot. When asked, enter your deployment password.

```
cd /demoapp/app-service-web-dotnet-get-started
```

```
git add .
```

```
git commit -m "New version of web app."
```

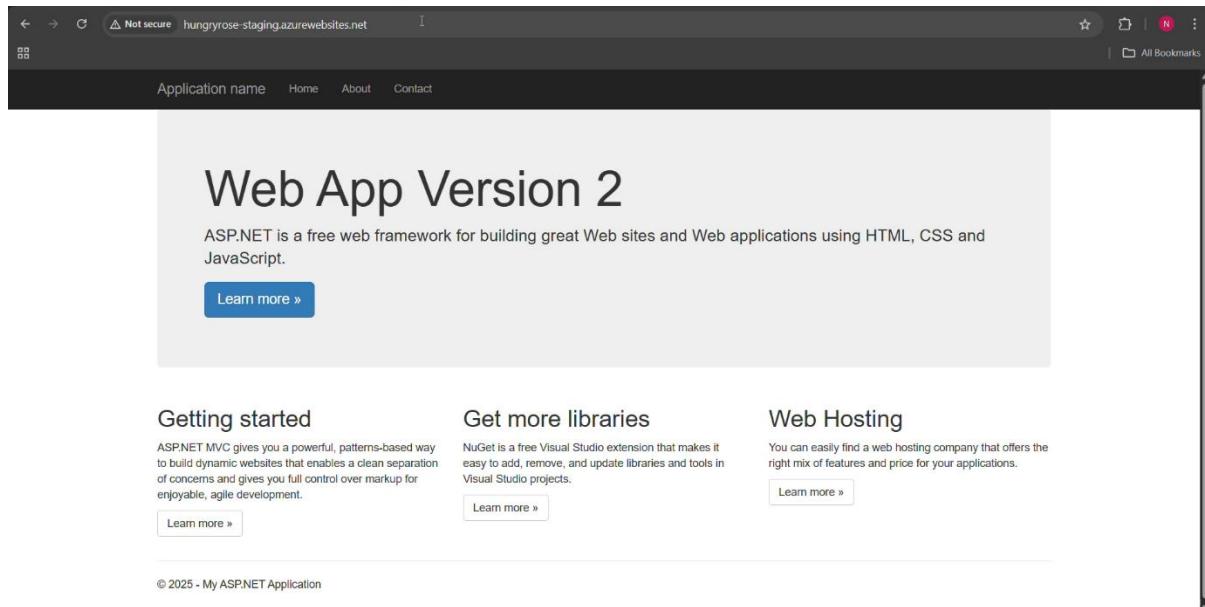
```
git push staging main:master
```

```
nur [ ~/demopp/app-service-web-dotnet-get-started ]$ git push staging main:master
Password for 'https://kamalashmezan@hungryrose-staging.scm.azurewebsites.net':
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Updating branch 'master'.
remote: Updating submodules.
remote: Preparing deployment for commit id '9d04bec707'.
remote: Generating deployment script.
remote: Project file path: .\aspnet-get-started\aspnet-get-started.csproj
remote: Solution file path: .\aspnet-get-started.sln
remote: Generating deployment script for .NET Web Application
remote: Generated deployment script files
remote: Running deployment command...
remote: Handling .NET Web Application deployment.
remote: MSBuild auto-detection: using msbuild version '14.0.23107.0' built by: D14REL' from 'C:\Program Files (x86)\MSBuild\14.0\Bin'.
remote: Restoring NuGet package Antlr.3.5.0.2.
remote: GET https://api.nuget.org/v3-flatcontainer/antlr/3.5.0.2/antlr.3.5.0.2.nupkg
remote: OK https://api.nuget.org/v3-flatcontainer/antlr/3.5.0.2/antlr.3.5.0.2.nupkg 27ms
```

## Browse the staging slot

Now you can see the new version of the web app by visiting the URL of the staging deployment slot.

In the Azure portal, navigate to the Overview page for the staging slot, then click Browse in the top menu bar. The updated web app will open in a new browser tab.



At this stage, the staging slot contains the new version of your code, allowing you to test it. Keep in mind that the production slot still runs the previous version, so users have not seen your new updates yet.

## Lab 30B: Deploy a web app by using deployment slots

When it's time to swap two deployment slots, it's important to confirm that each slot has the appropriate configuration after the swap. For instance, after completing tests on version 2 of a social media web application, the next step would be to deploy it to the production environment. To streamline future deployments, automatic slot swapping can also be configured. This section explains how to carry out both manual and automatic slot swaps.

### Configure a slot setting

Before deploying version 2 of the web app, a slot setting should be configured. While these settings will not affect the demo app, this exercise is designed to demonstrate how configurations behave during slot swaps.

To configure slot settings:

1. Select All resources in the Azure portal, and navigate the Overview page of the production slot of the web app (the main web app).

The screenshot shows the Azure portal interface for managing a web application named 'hungryrose'. On the left, the 'All resources' blade is open, displaying a list of resources including 'ASP-mslearnslots-8417', 'DefaultWorkspace-5065bd50-6897', 'hungryrose', and 'staging (hungryrose/staging)'. The main content area is focused on the 'hungryrose' web app's overview. The 'Overview' tab is selected, providing a summary of the app's configuration. Key details shown include the resource group 'mslearn-slots', which is being moved; the app is currently running in the Canada Central location; and it is associated with the 'Azure subscription 1' and has a subscription ID of '5065bd50-6897-43b2-8405-323c9cadf0f0'. The 'Properties' tab is active, showing the 'Web app' properties where the name is set to 'hungryrose' and the publishing model is 'Code'. The top navigation bar includes search, copilot, and user account information.

## 2. Under Settings, select Environment Variables

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes 'Microsoft Azure', a search bar, and user information 'aishah.mezlan@s.unikl.edu.my UNIVERSITY KUALA LUMPUR (UNI...)'.

The main area displays the 'hungryrose | Environment variables' page for a Web App. On the left, a sidebar lists 'Resource visualizer', 'Deployment', 'Deployment slots', 'Deployment Center', and 'Settings'. Under 'Settings', 'Environment variables' is selected, highlighted with a blue border. Other options include 'Configuration', 'Authentication', 'Identity', 'Backups', 'Custom domains', 'Certificates', 'Networking', 'Scale up (App Service plan)', and 'Scale out (App Service plan)'. Below the sidebar are 'Apply' and 'Discard' buttons.

The central area is titled 'App settings' and contains a table for managing environment variables. The table has columns for 'Name', 'Value', 'Deployment slot setting', 'Source', and 'Delete'. A search bar and 'Add', 'Refresh', 'Show values', 'Advanced edit', and 'Full reference values' buttons are located above the table. The table currently contains one row with the name 'Name' and value 'Value'.

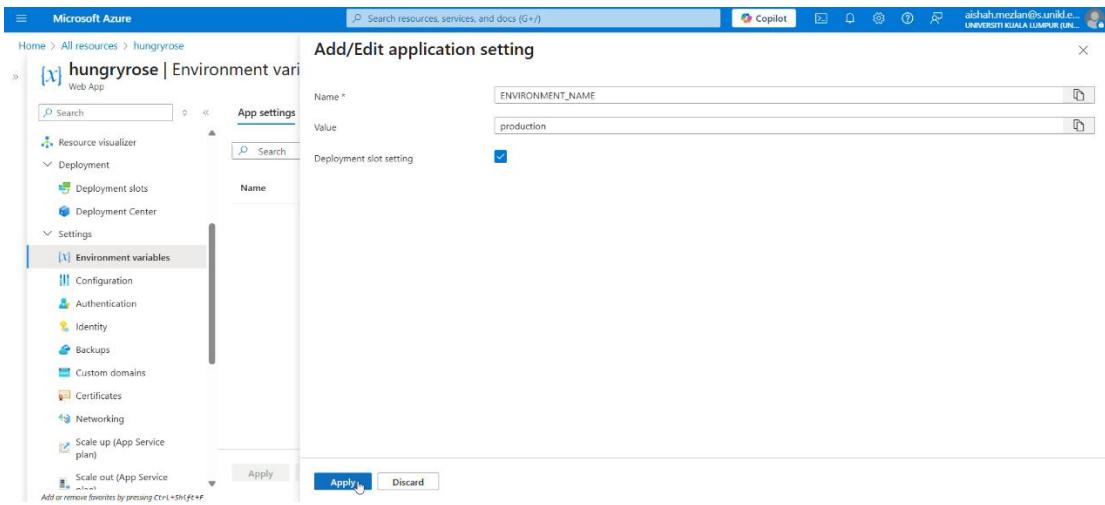
## 3. In the App settings section, click Add. Add/Edit application setting pane appears.

The screenshot shows the Microsoft Azure portal interface, similar to the previous one, but with a different focus. The 'Add/Edit application setting' pane is open in the center.

The pane has fields for 'Name \*' and 'Value'. Below these fields is a 'Deployment slot setting' checkbox. At the bottom of the pane are 'Apply' and 'Discard' buttons.

The background sidebar and overall layout are identical to the first screenshot, showing the 'hungryrose | Environment variables' page.

4. Create a new setting named ENVIRONMENT\_NAME with the value production. Make sure to check the deployment slot setting box to designate it as a slot-specific setting.

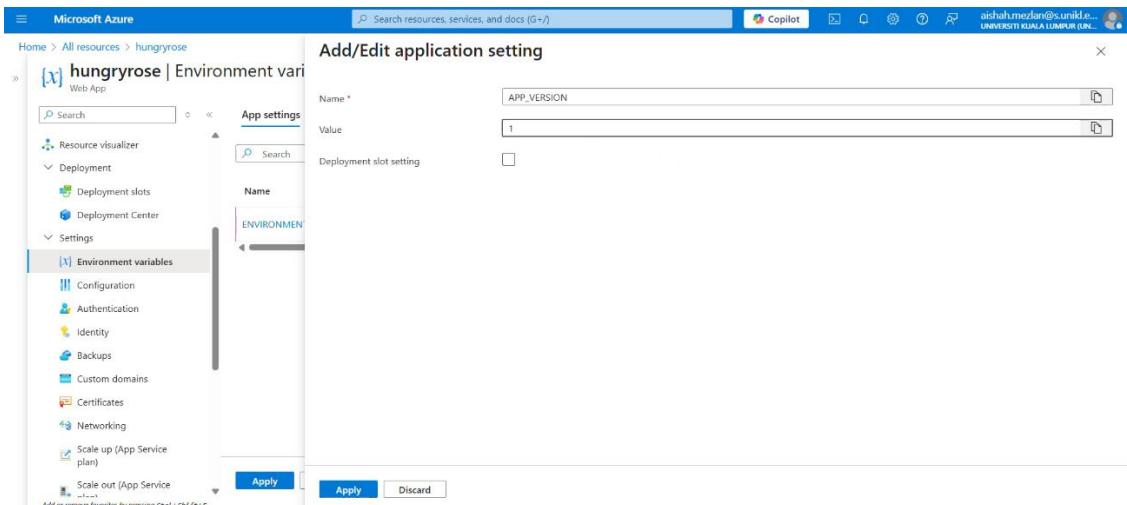


The screenshot shows the Microsoft Azure portal interface for managing an environment variable. On the left, the navigation pane is open with 'hungryrose' selected. Under 'App settings', there is a sub-section for 'Environment variables'. A new setting is being created with the following details:

- Name**: ENVIRONMENT\_NAME
- Value**: production
- Deployment slot setting**:

At the bottom right of the dialog are 'Apply' and 'Discard' buttons.

5. Add another setting called APP\_VERSION, and enter the value 1. Don't make this a slot setting.



The screenshot shows the Microsoft Azure portal interface for managing an environment variable. On the left, the navigation pane is open with 'hungryrose' selected. Under 'App settings', there is a sub-section for 'Environment variables'. A new setting is being created with the following details:

- Name**: APP\_VERSION
- Value**: 1
- Deployment slot setting**:

At the bottom right of the dialog are 'Apply' and 'Discard' buttons.

6. After both are done, click Apply to save changes.

The screenshot shows the Microsoft Azure portal interface. The URL in the address bar is <https://portal.azure.com/#@unik.edu.my/resource/subscriptions/5065bd50-6897-43b2-8405-323c9cadff0/resourceGroups/mslearn-slots/providers/Microsoft.Web/sites/hungryrose/environmen...>. The page title is "hungryrose | Environment variables". The left sidebar shows "All resources > hungryrose" and "Web App". The main content area is titled "App settings" under "Environment variables". It lists two environment variables: "APP\_VERSION" with value "2" and "ENVIRONMENT\_NAME" with value "staging". Both have "Show value" checkboxes checked. The "Source" column indicates "App Service" for both. At the bottom are "Apply" and "Discard" buttons.

7. Repeat the steps on the Staging slot, but use the following values

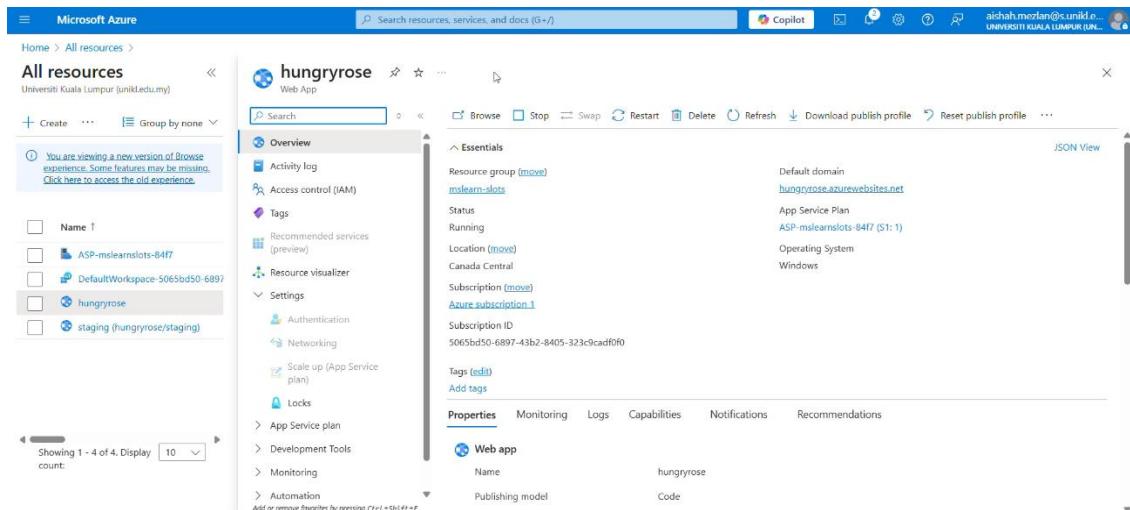
Name	Value	Deployment slot setting
ENVIRONMENT_NAME	staging	Yes
APP_VERSION	2	No

The screenshot shows the Microsoft Azure portal interface. The URL in the address bar is <https://portal.azure.com/#@unik.edu.my/resource/subscriptions/5065bd50-6897-43b2-8405-323c9cadff0/resourceGroups/mslearn-slots/providers/Microsoft.Web/sites/hungryrose/staging/environmen...>. The page title is "staging (hungryrose/staging) | Environment variables". The left sidebar shows "All resources > staging (hungryrose/staging)" and "App Service (Slot)". The main content area is titled "App settings" under "Environment variables". It lists three environment variables: "APP\_VERSION" with value "2", "ENVIRONMENT\_NAME" with value "staging", and "WEBSITE\_NODE\_DEFAULT\_VER..." with value "10". All three have "Show value" checkboxes checked. The "Source" column indicates "App Service" for all. At the bottom are "Apply" and "Discard" buttons.

## Swap the slots

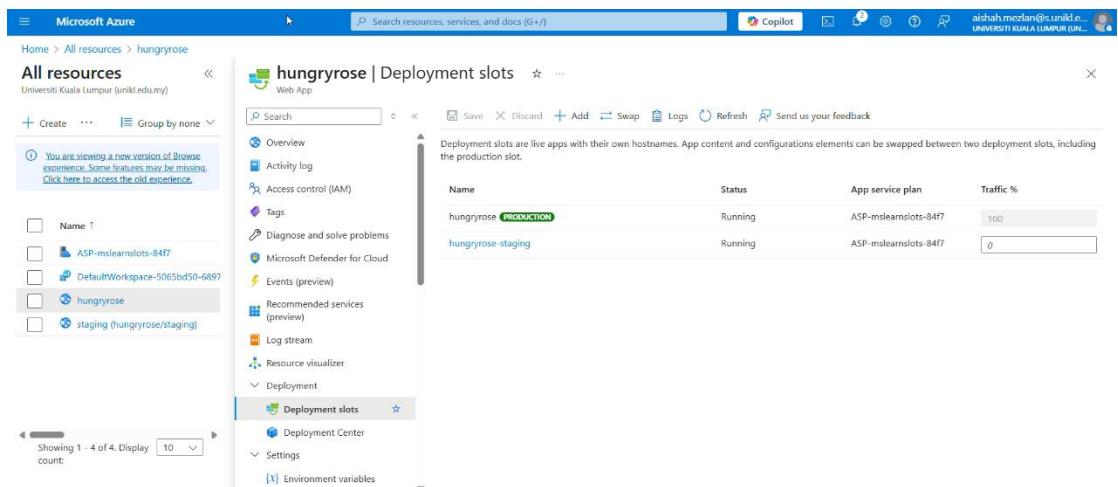
After testing version 2 of the web app in the staging slot, it can be deployed by swapping the slots. Follow the steps below to proceed:

1. To ensure you are configuring the production slot, go to All resources and then select the production slot of your web app.



The screenshot shows the Microsoft Azure portal's 'All resources' view. On the left, there's a search bar and a sidebar with options like 'Create', 'Group by none', and a note about viewing a new version of the browser experience. The main pane displays a list of resources, including 'hungryrose' (selected), 'ASP-mslearnslots-84f7', 'DefaultWorkspace-5065bd50-6897', and 'staging (hungryrose/staging)'. On the right, the 'hungryrose' resource details page is open. It shows the 'Essentials' section with information like Resource group (mslearn-slots), Status (Running), Location (Canada Central), and App Service Plan (ASP-mslearnslots-84f7). Below this is the 'Properties' section, which includes the 'Web app' properties: Name (hungryrose) and Publishing model (Code). At the top of the page, there are buttons for 'Search', 'Copilot', and other management actions, along with a 'JSON View' link.

2. In the left menu pane, under Deployment, select Deployment slots. On the top, click Swap



The screenshot shows the Microsoft Azure portal's 'All resources' view again, but this time the 'Deployment slots' section is selected in the left sidebar. The main pane displays the 'Deployment slots' table for the 'hungryrose' app. It lists two slots: 'hungryrose PRODUCTION' (Status: Running, App service plan: ASP-mslearnslots-84f7, Traffic %: 100) and 'hungryrose-staging' (Status: Running, App service plan: ASP-mslearnslots-84f7, Traffic %: 0). Above the table, there are buttons for 'Save', 'Discard', 'Add', 'Swap', 'Logs', 'Refresh', and 'Send us your feedback'. The table has columns for Name, Status, App service plan, and Traffic %. The 'hungryrose PRODUCTION' row is highlighted with a blue background.

3. Confirm that you are swapping the staging and production slots. Notice how the swap will impact settings: the APP\_VERSION setting value will be exchanged between the slots, but the ENVIRONMENT\_NAME slot setting will remain unchanged. Then, click Start Swap.

Setting	Type	Old Value	New Value
PhpVersion	General	5.6	
APP_VERSION	AppSetting	2	1
WEBSITE_NODE_DEFAULT...	AppSetting	6.9.1	Not set

4. Once the swap is finished, navigate to the Overview page of the production slot's web app and click **Browse**. The web app will open in a new browser tab, showing that version 2 is now live in production.

**Getting started**  
ASP.NET MVC gives you a powerful, patterns-based way to build dynamic websites that enables a clean separation of concerns and gives you full control over markup for enjoyable, agile development.  
[Learn more »](#)

**Get more libraries**  
NuGet is a free Visual Studio extension that makes it easy to add, remove, and update libraries and tools in Visual Studio projects.  
[Learn more »](#)

**Web Hosting**  
You can easily find a web hosting company that offers the right mix of features and price for your applications.  
[Learn more »](#)

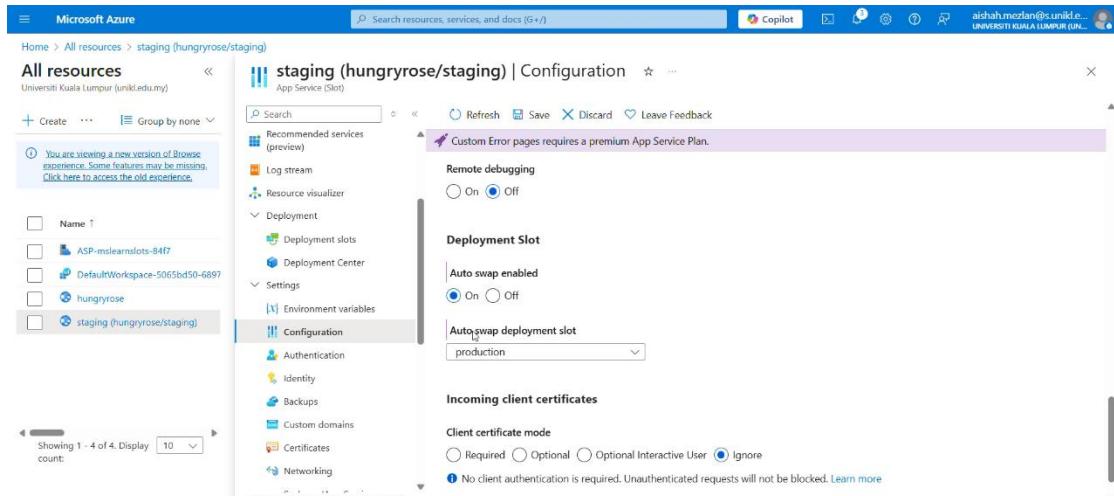
© 2025 - My ASP.NET Application

5. Close the browser tab

## Configure auto swap for the staging slot

With deployment slots in use, enabling continuous deployment can be achieved by turning on the auto swap feature for the web app. When auto swap is activated, deploying new code to the staging slot prompts Azure to automatically warm it up and then swap the staging and production slots, deploying the update to production. To configure auto swap, follow these steps:

1. Go to the Configuration page of the staging slot's web app, and navigate to the General settings tab.
2. Set Auto swap enabled to On. Then, select production in the Auto swap deployment list, and then select Save.



## Deploy new code and auto swap it into production

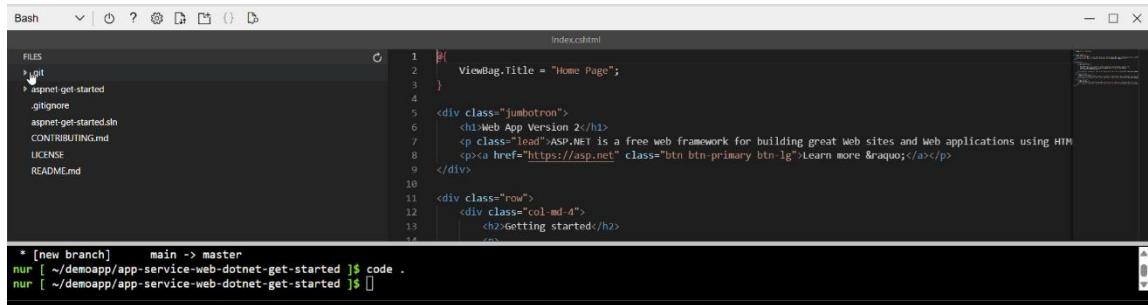
Next, the code will be updated to create version 3 of the web app. Deploying this version to the staging slot will allow observation of the auto swap feature in action. Follow these steps:

1. Open Cloud Shell and execute the following commands.

```
cd ~/demoapp/app-service-web-dotnet-get-started/
```

```
code .
```

2. In the code editor, in the File list on the left, expand aspnet-get-started > Views > Home, and then select Index.cshtml.



```
index.cshtml
1  @{
2     ViewBag.Title = "Home Page";
3 }
4
5 <div class="jumbotron">
6     <h1>Web App Version 2</h1>
7     <p>ASP.NET is a free web framework for building great Web sites and Web applications using HTML<br/><a href="https://asp.net" class="btn btn-primary btn-lg">Learn more &gt;</a></p>
8
9 </div>
10
11 <div class="row">
12     <div class="col-md-4">
13         <h2>Getting started</h2>
14     </div>
15 </div>
```

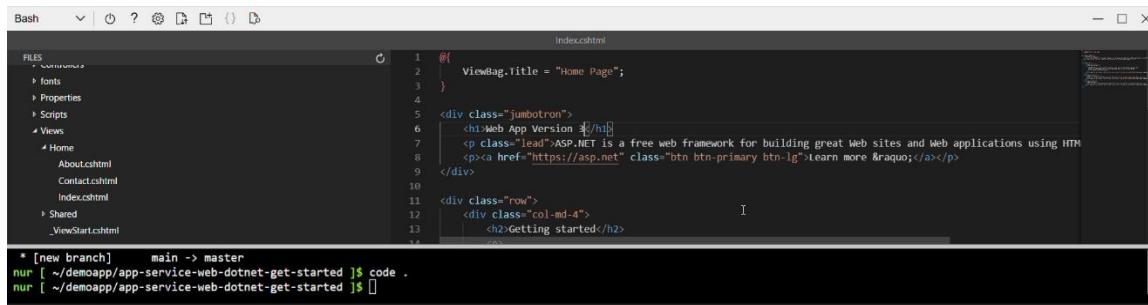
\* [new branch] main -> master  
nur [ ~/demoapp/app-service-web-dotnet-get-started ]\$ code .  
nur [ ~/demoapp/app-service-web-dotnet-get-started ]\$ ]

3. Locate the following code.

```
<h1>Web App Version 2</h1>
```

4. Replace that code with this code.

```
<h1>Web App Version 3</h1>
```



```
index.cshtml
1  @{
2     ViewBag.Title = "Home Page";
3 }
4
5 <div class="jumbotron">
6     <h1>Web App Version 3</h1>
7     <p>ASP.NET is a free web framework for building great Web sites and Web applications using HTML<br/><a href="https://asp.net" class="btn btn-primary btn-lg">Learn more &gt;</a></p>
8
9 </div>
10
11 <div class="row">
12     <div class="col-md-4">
13         <h2>Getting started</h2>
14     </div>
15 </div>
```

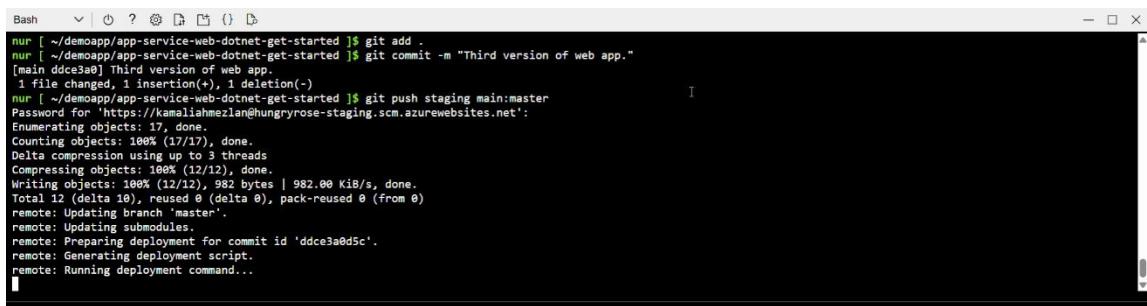
\* [new branch] main -> master  
nur [ ~/demoapp/app-service-web-dotnet-get-started ]\$ code .  
nur [ ~/demoapp/app-service-web-dotnet-get-started ]\$ ]

5. To save your changes, press Ctrl+S
6. In Cloud Shell, enter the following commands. Enter your deployment password when you're prompted.

```
git add .
```

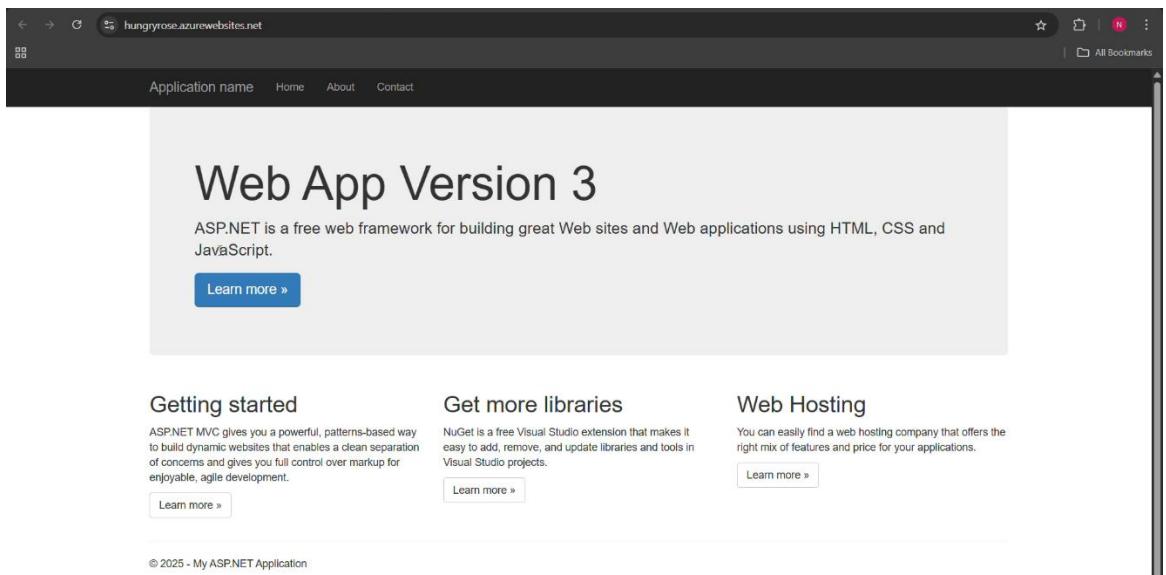
```
git commit -m "Third version of tweb app."
```

```
git push staging main:master
```



```
Bash ~ ~/demoapp/app-service-web-dotnet-get-started $ git add .
nur [ ~/demoapp/app-service-web-dotnet-get-started $ git commit -m "Third version of web app."
[main ddce3a0] Third version of web app.
 1 file changed, 1 insertion(+), 1 deletion(-)
nur [ ~/demoapp/app-service-web-dotnet-get-started $ git push staging main:master
Password for 'https://Kameliahmeilan@hungryrose-staging.scm.azurewebsites.net':
Enumerating objects: 17, done.
Counting objects: 100% (17/17), done.
Delta compression using up to 3 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (12/12), 982 bytes | 982.00 KiB/s, done.
Total 12 (delta 10), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Updating branch 'master'.
remote: Updating submodules.
remote: Preparing deployment for commit id 'ddce3a0d5c'.
remote: Generating deployment script.
remote: Running deployment command...
```

7. Wait for the deployment to complete. Near the end of the output, you'll see a message indicating that an auto swap to the production slot has been requested.
8. In the Azure portal, go to the Overview page of the production slot's web app and click Browse. Version 3 of the web app should open in a new browser tab. If you still see the previous version, wait a moment and refresh the page. Although the swap is atomic and happens instantly, App Service may take a few moments to prepare and execute the operation.



Application name Home About Contact

# Web App Version 3

ASP.NET is a free web framework for building great Web sites and Web applications using HTML, CSS and JavaScript.

[Learn more »](#)

**Getting started**  
ASP.NET MVC gives you a powerful, patterns-based way to build dynamic websites that enables a clean separation of concerns and gives you full control over markup for enjoyable, agile development.

[Learn more »](#)

**Get more libraries**  
NuGet is a free Visual Studio extension that makes it easy to add, remove, and update libraries and tools in Visual Studio projects.

[Learn more »](#)

**Web Hosting**  
You can easily find a web hosting company that offers the right mix of features and price for your applications.

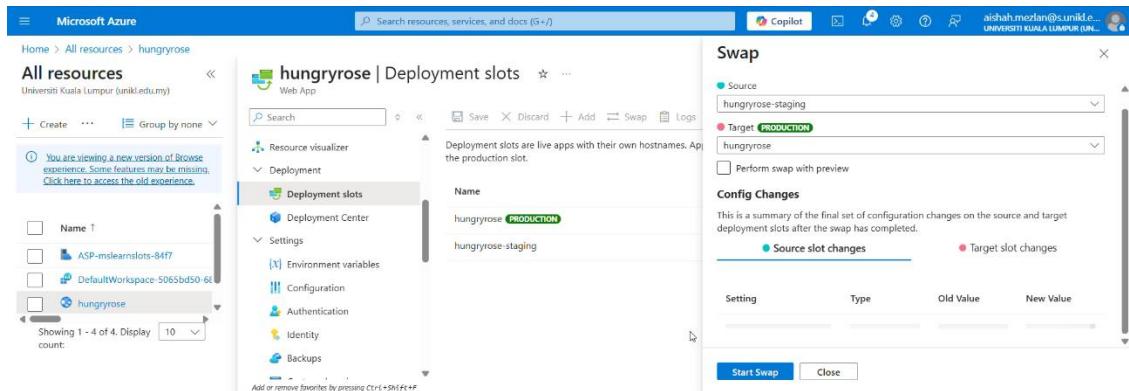
[Learn more »](#)

© 2025 - My ASP.NET Application

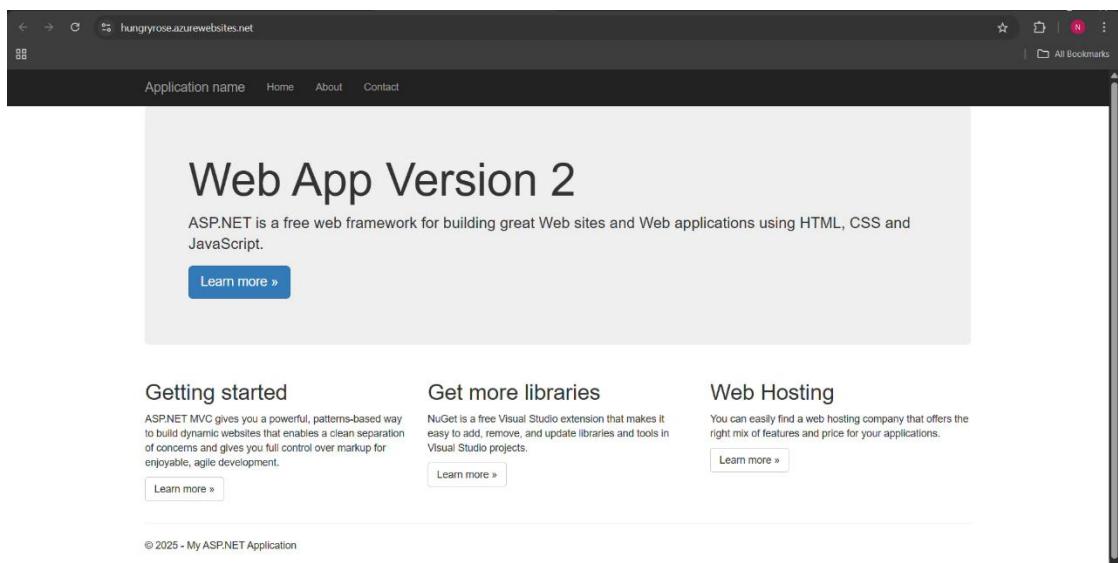
## Roll back the new version

If deploying version 3 of the app to production causes an unexpected issue, it can be quickly resolved by rolling back to the previous version through swapping the deployment slots again.

1. In the Azure portal, go to the Deployment slots page of the production slot's web app.
2. Swap the staging and production slots.



3. Once the swap is complete, go to the Overview page and select Browse to view the app. You'll see that version 2 has been successfully redeployed to production.



## **Lab 31: Scale a web app manually**

A system should be scaled out when an increase in traffic is expected or when there is a decline in performance.

In the case of a hotel reservation system, the number of web app instances may be increased to accommodate extra traffic caused by special events, promotional offers, or seasonal fluctuations. Once demand decreases, the system can be scaled back accordingly.

In this section, an App Service plan will be created, and a web app will be deployed using that plan. The performance of the web app will be monitored under load. Afterward, the app will be scaled out, and the improved performance will be verified.

This section uses a sample web app that implements a web API, which includes HTTP POST and GET operations to create and retrieve customer bookings for a hotel reservation website. These bookings are not saved—GET operations return dummy data.

A client app will also be run to simulate multiple users sending POST and GET requests simultaneously. This provides the workload needed to test the web app's performance before and after scaling.

### **Create an App Service plan and web app**

An individual Azure subscription is required to complete this section, and there may be associated charges. If an Azure subscription is not already available, a free account should be created before beginning.

1. In the Azure portal, from the menu or the Home page, select Create a resource. The Create a resource pane will appear.
2. In the menu under Categories, select Web, then search for and select Web App, and click Create. The Create Web App pane will open.
3. On the Basics tab, provide the following values for each setting as required.

	Setting	Value
Project Details	Subscription	Select the Azure subscription you'd like to use for this exercise.
	Resource Group	Create a new resource group called mslearn-scale.
Instance Details	Name	Enter a unique name that you can remember for later in this exercise.
	Publish	Code
	Runtime Stack	.NET Core 3.1 (LTS)
	Operating System	Windows
	Region	Leave default
App Service Plan	App Service plan	Leave default
	Location	Leave default

4. Select Review + Create, and when validation completes, select Create. Wait for the web app to be created and deployed.

## Build and deploy the web app

1. Open Cloud Shell and run the following command to download the source code for the hotel reservation system.

```
git clone https://github.com/MicrosoftDocs/mslearn-hotel-reservation-system.git
```

2. Go to mslearn-hotel-reservation-system/src folder.

```
cd mslearn-hotel-reservation-system/src
```

```
Bash [ ~ ]$ git clone https://github.com/MicrosoftDocs/mslearn-hotel-reservation-system.git
Type "az" to use Azure CLI
Type "help" to learn about Cloud Shell
Warning: Failed to mount the Azure file share. Your cloud drive won't be available.
sharifah [ ~ ]$ Cloning into 'mslearn-hotel-reservation-system'...
remote: Enumerating objects: 33, done.
remote: Counting objects: 100% (4/4), done.
remote: Total 33 (delta 3), reused 3 (delta 3), pack-reused 29 (from 1)
Receiving objects: 100% (33/33), 17.36 KB | 3.47 MB/s, done.
Resolving deltas: 100% (3/3), done.
sharifah [ ~ ]$ cd mslearn-hotel-reservation-system/src
```

3. Build the applications for the hotel reservation system. There are two components:

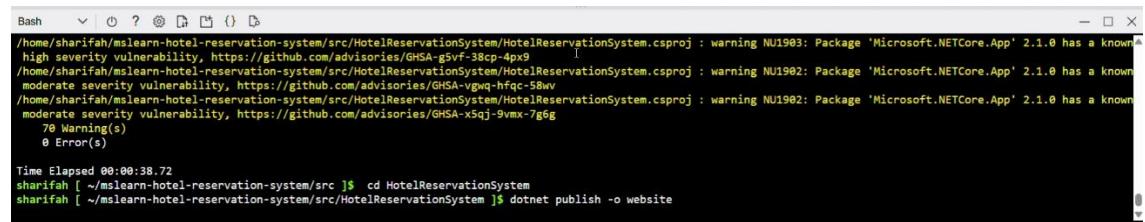
- A **web app** that implements the web API for the system.
- A **client app** used to perform load testing on the web app.

```
dotnet build
```

4. Prepare the HotelReservationSystem web app for publishing.

```
cd HotelReservationSystem
```

```
dotnet publish -o website
```



```
Bash └─? @ ⓘ () 
/home/sharifah/mslearn-hotel-reservation-system/src/HotelReservationSystem/HotelReservationSystem.csproj : warning NU1903: Package 'Microsoft.NETCore.App' 2.1.0 has a known high severity vulnerability, https://github.com/advisories/GHSA-g5vf-38cp-4px9
/home/sharifah/mslearn-hotel-reservation-system/src/HotelReservationSystem/HotelReservationSystem.csproj : warning NU1902: Package 'Microsoft.NETCore.App' 2.1.0 has a known moderate severity vulnerability, https://github.com/advisories/GHSA-vgqg-hfc-58wv
/home/sharifah/mslearn-hotel-reservation-system/src/HotelReservationSystem/HotelReservationSystem.csproj : warning NU1902: Package 'Microsoft.NETCore.App' 2.1.0 has a known moderate severity vulnerability, https://github.com/advisories/GHSA-x5qj-9vmx-7g6g
 70 Warning(s)
 0 Error(s)

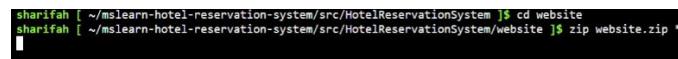
Time Elapsed 00:00:38.72
sharifah [ ~/mslearn-hotel-reservation-system/src ]$ cd HotelReservationSystem
sharifah [ ~/mslearn-hotel-reservation-system/src/HotelReservationSystem ]$ dotnet publish -o website
```

5. Go to the folder that holds the published website files. Compress these files into a ZIP archive. Then, upload the ZIP file to the web app you set up earlier. Copy the code below into Notepad, replace <your-webapp-name> with your actual web app name, and then paste the updated code into Cloud Shell to run it.

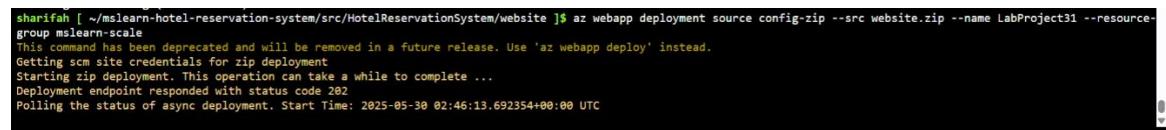
```
cd website
```

```
zip website.zip *
```

```
az webapp deployment source config-zip --src website.zip --name YOUR-WEBAPP-NAME --resource-group mslearn-scale
```

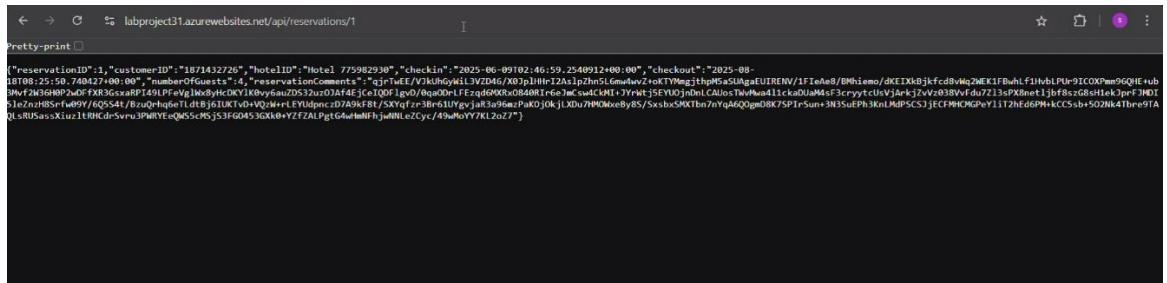


```
sharifah [ ~/mslearn-hotel-reservation-system/src/HotelReservationSystem ]$ cd website
sharifah [ ~/mslearn-hotel-reservation-system/src/HotelReservationSystem/website ]$ zip website.zip *
```



```
sharifah [ ~/mslearn-hotel-reservation-system/src/HotelReservationSystem/website ]$ az webapp deployment source config-zip --src website.zip --name LabProject31 --resource-group mslearn-scale
This command has been deprecated and will be removed in a future release. Use 'az webapp deploy' instead.
Getting scm site credentials for zip deployment
Starting zip deployment. This operation can take a while to complete ...
Deployment endpoint responded with status code 202
Polling the status of async deployment. Start Time: 2025-05-30 02:46:13.692354+00:00 UTC
```

6. Open your web browser and navigate to `http://<your-webapp-name>.azurewebsites.net/api/reservations/1`. You should see a JSON document displaying the information for reservation number 1.



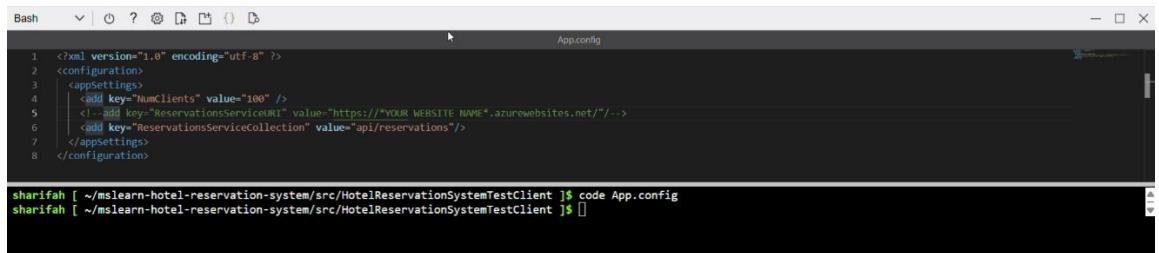
**Monitor the performance of the web app before scaling out**

1. Return to the Cloud Shell and go to the `~/mslearn-hotel-reservation-system/src/HotelReservationSystemTestClient` folder.

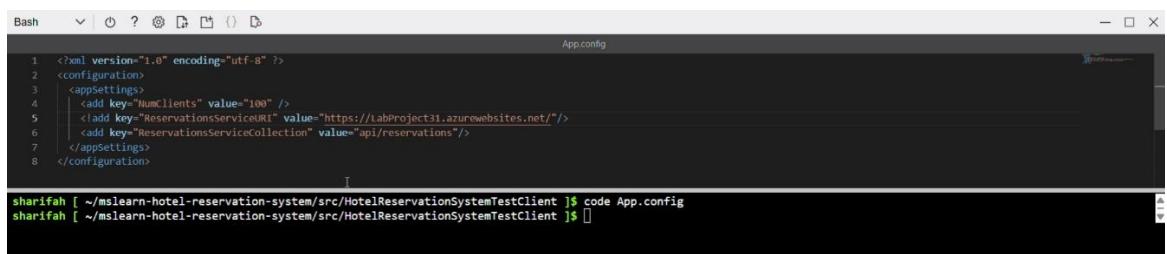
```
cd ~/mslearn-hotel-reservation-system/src/HotelReservationSystemTestClient
```

2. Edit the App.config file in this folder by using the code editor.

## code App.config



3. Uncomment the line that specifies the ReservationsServiceURI and replace the value *YOUR WEBSITE NAME* with the name of your web app.



4. Save the file (Ctrl+S) and close the code editor (Ctrl+Q)
  5. Rebuild the test client app with the new configuration

dotnet build

6. Edit the HotelReservationSystemTestClient.csproj file to use the correct framework.

code HotelReservationSystemTestClient.csproj

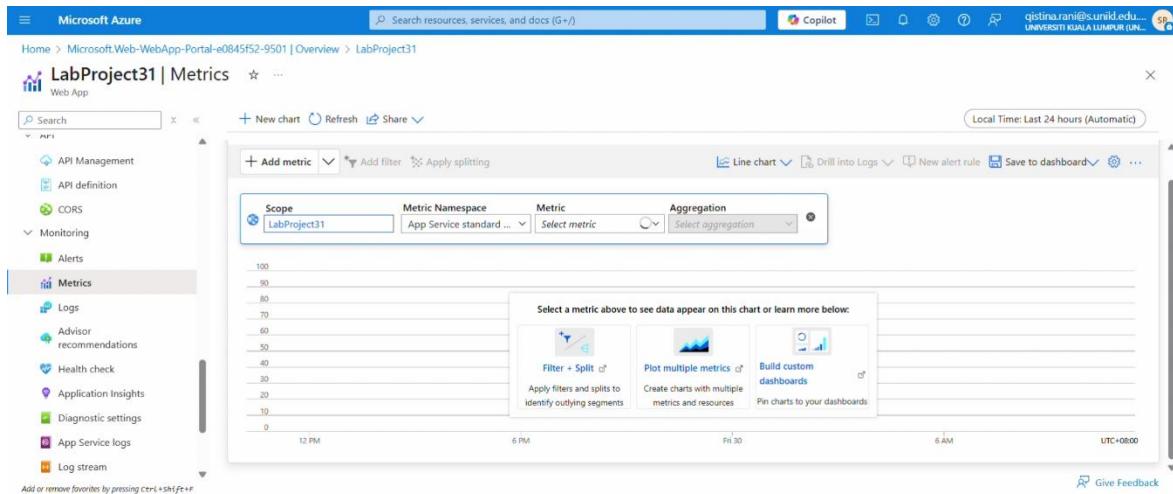
In this case, net8.0 is being used instead of netcore2.1.

7. Save the file (Ctrl+S) and close the code editor (Ctrl+Q)
  8. Run the client app

dotnet run

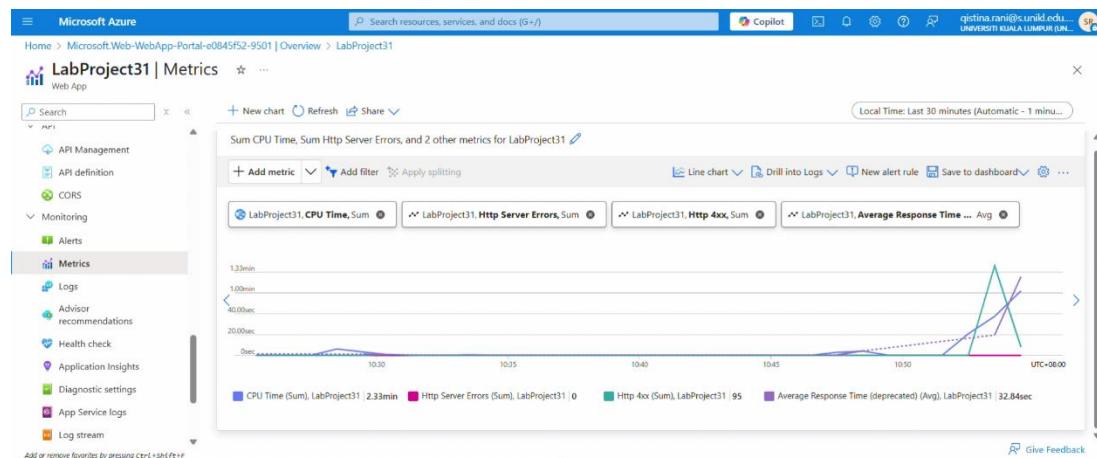
Several messages will appear as the clients begin running, making reservations, and sending queries. Let the system run for a few minutes. You'll notice that the responses become slow, and the client requests will eventually start failing with HTTP 408 (Timeout) errors.

9. In the Azure portal, click Go to resource to open your web app.
10. In the left-hand menu, scroll down to the Monitoring section and select Metrics.
11. In the top menu bar of the Metrics page, click the time range option that says Local Time: Last 24 hours (Automatic). Change it to Last 30 minutes, then click Apply.

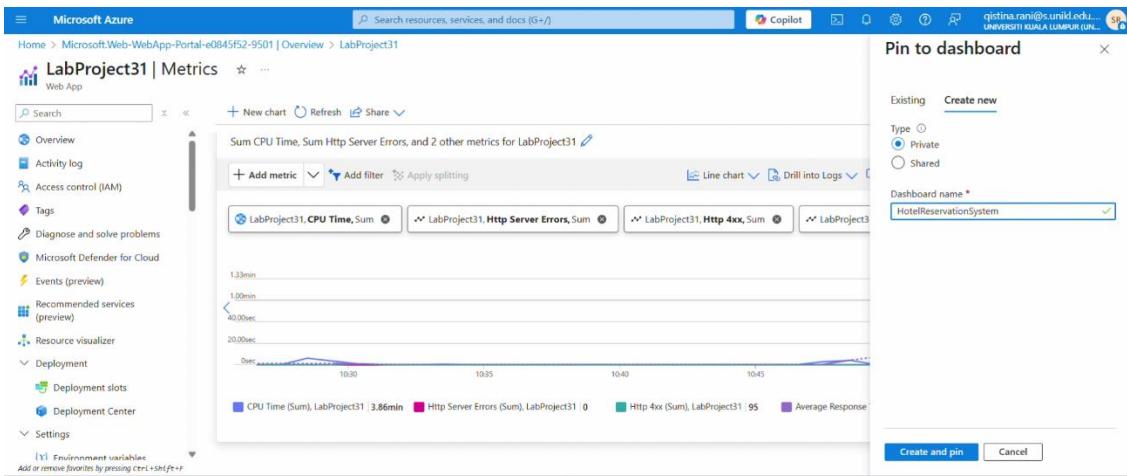


12. In the menu under Chart Title, add the following metrics to the chart.

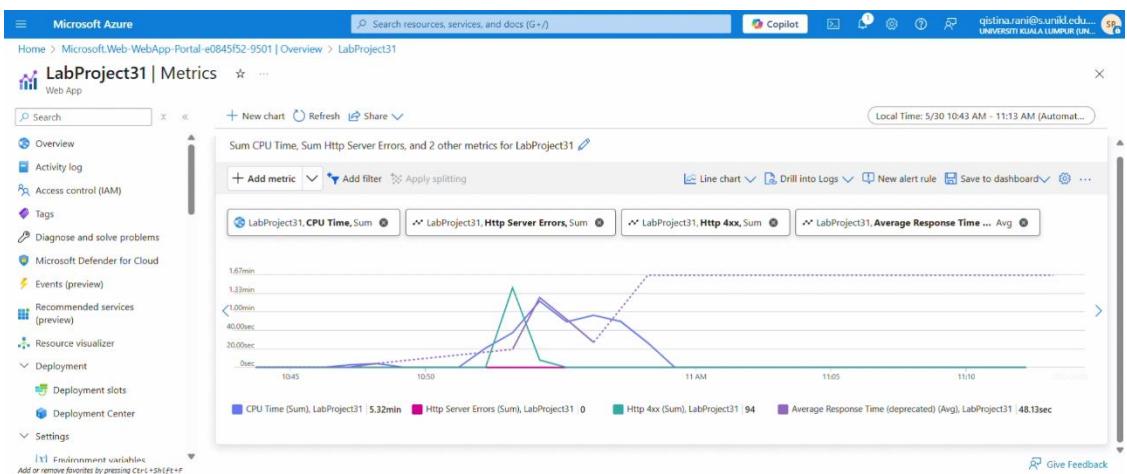
- Select Add metric CPU Time. Select the Sum aggregation.
- Select Add metric Http Server Errors. Select the Sum aggregation.
- Select Add metric Http 4xx. Select the Sum aggregation.
- Select Add metric Average Response Time. Select the Sum aggregation.



13. In the menu under Chart Title, select Pin to dashboard, then click Pin. Create dashboard first if you don't have any.



14. Let the system run for about five minutes to stabilize. After that, take note of the CPU Time, the number of HTTP 4xx errors, and the average response time. You should observe a high number of HTTP 4xx errors—specifically, HTTP 408 Timeout errors—and an average response time of several seconds. Depending on how well the web server is handling the load, you might also see occasional HTTP server errors.



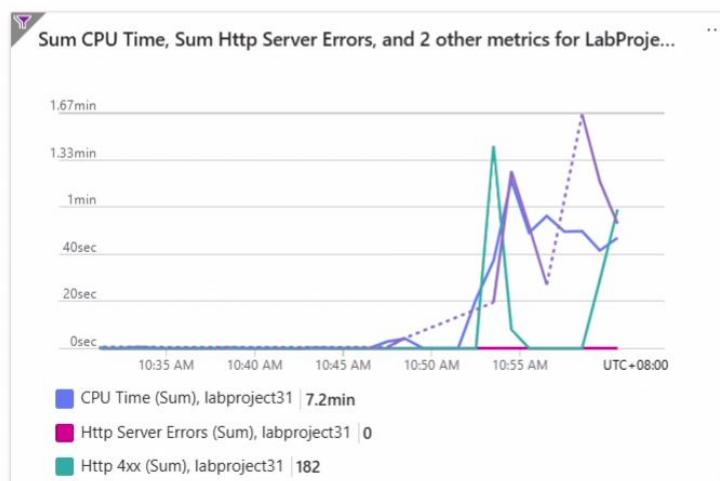
15. Leave the client app running while you perform the next task.

## Scale out the web app and verify the performance improvement

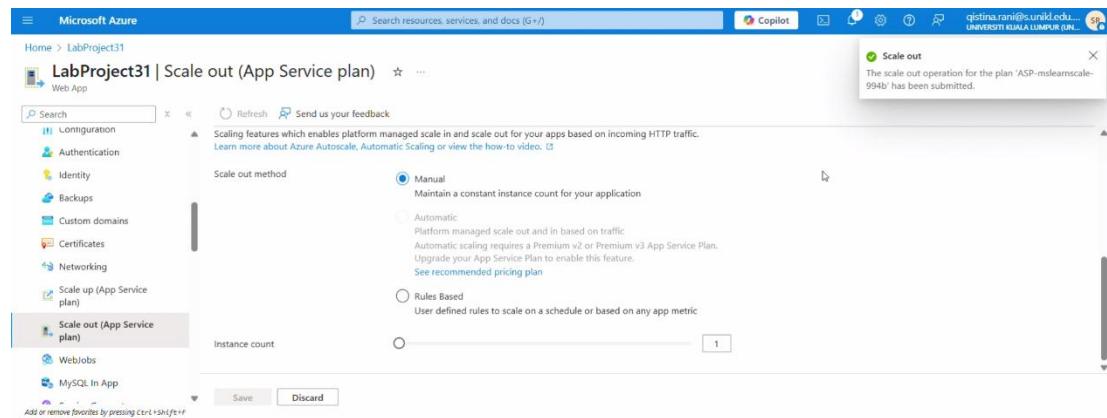
1. In the Azure portal, click on your web app name under App Services. In the menu, under Settings, select Scale-out (App Service Plan).
2. Change the Instance count to 5, then click Save.

The screenshot shows the Azure portal interface for managing an App Service plan. The left sidebar lists various settings like Authentication, Backups, Custom domains, Certificates, Networking, Scale up (App Service plan), and Scale out (App Service plan). The 'Scale out (App Service plan)' option is selected. On the right, there's a section titled 'Scale out method' with two options: 'Manual' (selected) and 'Automatic'. Under 'Manual', it says 'Maintain a constant instance count for your application' and shows a slider set to '5'. Below the slider is a link to 'See recommended pricing plan'. At the bottom are 'Save' and 'Discard' buttons.

3. Open Cloud Shell. You should notice a decrease in the number of requests failing with errors, although some requests may still time out.
4. Let the app run for another five minutes. Then, go to the dashboard in the Azure portal and view the chart displaying the app's metrics. You should observe a significant increase in CPU time, reflecting the added processing power from scaling out to five instances. The average response time should have decreased, and the number of HTTP 4xx errors should have gone down as well. The chart will typically indicate the point at which the scale-out took place.



5. If you'd like to explore further, try increasing the instance count of your App Service plan to 10, which is the maximum supported by the S1 pricing tier. You should see an additional rise in CPU time, along with a reduction in response time and fewer HTTP 4xx errors.
6. When you're done, return to the Cloud Shell running the client app and press Enter to stop it.
7. Then, in the Azure portal, reset the instance count to 1:
  - Select your web app.
  - In the menu, choose Scale out (App Service plan).
  - On the Configure tab, set the instance count to 1.
  - Click Save in the menu bar at the top of the App Service pane.



## Lab 32: Scale up a web app

Scaling up provides more powerful resources for running a web app and also increases the number of instances available for scaling out. In the hotel reservation system, scaling out is necessary to accommodate a growing number of visitors. Scaling up enables further scaling out and is likely required to support any new functionality added to the web app. In this section, the hotel reservation system web app that was previously deployed will be scaled up. The same test client application used earlier will be run again to simulate user activity, and the web app's performance will be monitored.

### Examine the current pricing tier for the web app

An individual Azure subscription is required to complete this section, and there may be associated charges. If an Azure subscription is not already available, a free account should be created before beginning.

1. In the Azure portal, from the menu or the Home page, select All resources, then navigate to your App Service plan.

Name	Type	Resource Group	Location	Subscription
Application Insights Smart Detection	Action group	mslearn-scale	Global	Azure subscription 1
ASP-mslearn-scale-994b	App Service plan	mslearn-scale	Canada Central	Azure subscription 1
cs110032002f5a992ff	Storage account	cloud-shell-storage-southeastasia	Southeast Asia	Azure subscription 1
DefaultWorkspace-9a4a509fe515-40eb-a569-2509dd0522f2-C...	Log Analytics workspace	DefaultResourceGroup-CCAN	Canada Central	Azure subscription 1
LabProject31	App Service	mslearn-scale	Canada Central	Azure subscription 1
LabProject31	Application Insights	mslearn-scale	Canada Central	Azure subscription 1

- Under Settings, select Scale up (App Service plan). This will display the details of the current pricing tier. The plan is set to S1, which offers 100 Azure Compute Units and 1.75 GB of memory, running on an A-Series virtual machine.

Premium v3 P3mvs	195*	8	64	250	30	1.446 USD	1055.872 USD
Premium v3 P4mvs	195*	16	128	250	30	2.893 USD	2111.744 USD
Premium v3 P5mvs	195*	32	256	250	30	5.786 USD	4223.488 USD
<b>Legacy</b>							
Standard S1	100	1	1.75	50	10	0.11 USD	80.30 USD
<b>Standard S1</b>	<b>100</b>	<b>1</b>	<b>1.75</b>	<b>50</b>	<b>10</b>	<b>0.11 USD</b>	<b>80.30 USD</b>
Standard S2	100	2	3.5	50	10	0.22 USD	160.60 USD
Standard S3	100	4	7	50	10	0.44 USD	321.20 USD
Premium P1	100	1	1.75	250	20	0.33 USD	240.90 USD
Premium P2	100	2	3.5	250	20	0.66 USD	481.80 USD
Premium P3	100	4	7	250	20	1.32 USD	963.60 USD
Premium v2 P1V2	210	1	3.5	250	30	0.22 USD	160.60 USD
Premium v2 P2V2	210	2	7	250	30	0.44 USD	321.20 USD
Premium v2 P3V2	210	4	14	250	30	0.88 USD	642.40 USD

## Run the test client app

- In the Cloud Shell, go to the ~/mslearn-hotel-reservation-system/src/HotelReservationSystemTestClient folder:

```
cd ~/mslearn-hotel-reservation-system/src/HotelReservationSystemTestClient
```

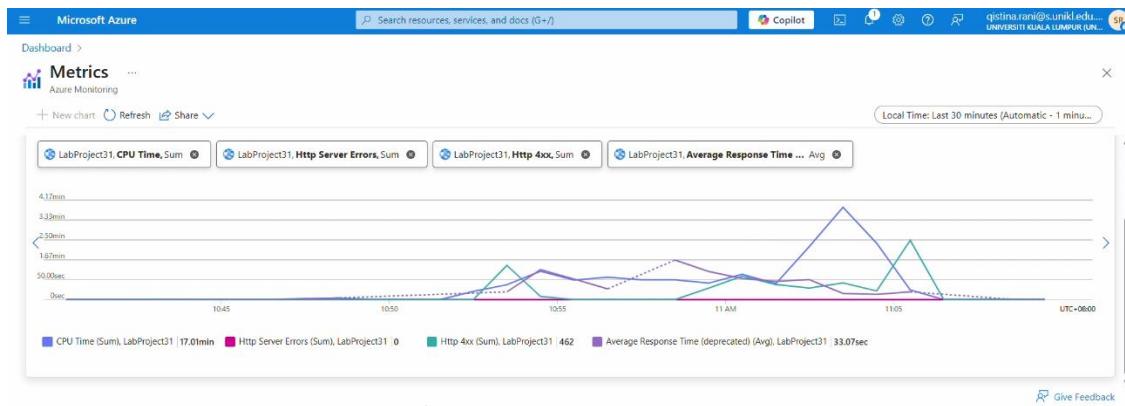
- Run the client app:

```
dotnet run
```

```
Bash └─ sharifah [ ~/mslearn-hotel-reservation-system/src/HotelReservationSystemTestClient ]$ dotnet run
Client ClientB7 querying reservation: Reservation 809792374, CustomerID 529429868, HotelID Hotel 596701246, Checkin 6/9/2025, Checkout 8/16/2025, Guests 2
Comments: WjB8MM4Qb+3H/gVPugXv3Uz+x+w101R+j1Zp15/+xLdSuXIdAraN9SCGUfW1Tdwh+AmfTRECoCMr+felgOmPFXH61Bng8G7Hyunbjw2u7nfdgiUxXhI41PndGxhp+IuevyafKqotzxehZN4uc0Tm3yV16XqdInR
m#7fDmPKf1Cs+h/Cf1pRM10raAyJHClaOR8q61BVtPooD+f4/1NXEK1lMDfOsQv6qPt3zYrTM431YTLPJeiw/ne61z3eHxNg1RcQkbe6z2mWaOxtY+G+J5B6fdFjPxP/ph62uYAT7XugunpEzIGnjSSAmRPj0V+2NTVmrxQk
B/36/yP8VHzTDjh1Hm4m5BqTKjv25m17c7Td1/Zs1n/Opgrm85qbVeB614+ppdrHQKAxHs5hBwPuVgts6f/c/VR7hv62vdMoAZLHCpFfirDgcrSE9CmQKVfoD1CD+E++3+u5xH1jhs1Ps/OYJo@vFFqtptTA+@V4Lgg@Vs
v1sYalVBmAQINbYShmfqvPtdNshJwozmc7wld8qg3zQHYtkrixmlciquduKwMkFmkj2yA5ExhgGAYCe28zy/NUSKhe12lMbXR1iLUa+H9XH34Nq3Xu10Hye5rnAbhuQfG5bc68my1PQRFTxKnZvB0W9L3VcurvwA3
ltkZ3GdqKrkfzfb5C4YNnaXaOpmgZU3V91gvSGU96+NNQhNk4qKpdFbhoyNpbmwX7PU89xRpecmulg9kthSl+8p/hZsdeywku8+VfoXKZgerFgsx6oQhDd1BykUbhyT7ju6zGu8VCLf1C2em9vMBEh74FBio12vsl
4jRC1571C+088CzCpT9AxauRbcChc1ry2DtP+KLs9d9FH8gYFYM51hRbsuixnsuVG81YNUxU2zTwOI=
Client ClientB7 making reservation 1524268947
```

Let the system run for a few minutes. As in the beginning of the previous exercise, the responses will be slow, and the client requests will soon begin to fail with HTTP 408 (Timeout) errors.

- Continue running the app and wait for another five minutes. Then, navigate to the chart displaying the web app metrics on the Azure portal dashboard. As observed in the previous section, the statistics should show relatively slow response times and a high number of HTTP 4xx errors.



## Scale up the web app and monitor the results

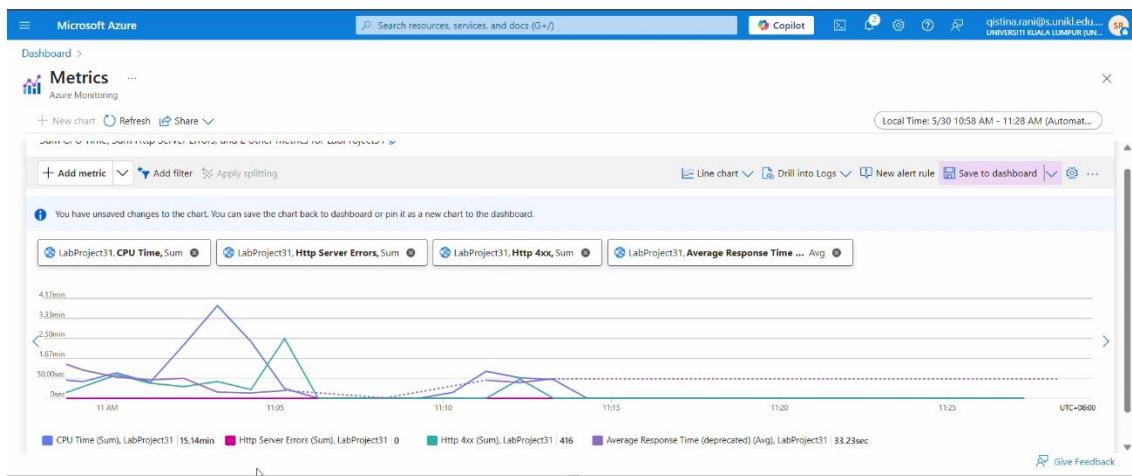
- In the Azure portal, go back to the page for the App Service plan.
- Under Settings, select Scale up (App Service plan).
- Choose the P2V2 pricing tier and click Apply. This tier provides 420 Azure Compute Units (ACU)—more than four times the power of the S1 tier—and 7 GB of memory, running on a Dv2-Series virtual machine. However, it costs approximately four times more than the S1 pricing tier.

The figure is a screenshot of the Microsoft Azure 'Scale up (App Service plan)' configuration page for the web app 'LabProject31'. The left sidebar shows navigation options like Authentication, Identity, Backups, Custom domains, Certificates, Networking, Scale up (App Service plan), Scale out (App Service plan), WebJobs, MySQL In App, Service Connector, Properties, and Locks. The main area displays a table of Azure Compute Unit (ACU) tiers and their prices:

Premium v3 P2mv3	195*	4	32	250	30	<b>0.723 USD</b>	<b>\$27.936 USD</b>	
Premium v3 P3mv3	195*	8	64	250	30	<b>1.446 USD</b>	<b>1055.872 USD</b>	
Premium v3 P4mv3	195*	16	128	250	30	<b>2.893 USD</b>	<b>2111.744 USD</b>	
Premium v3 P5mv3	195*	32	256	250	30	<b>5.786 USD</b>	<b>4223.488 USD</b>	
Standard S1	100	1	1.75	50	10	<b>0.11 USD</b>	<b>80.30 USD</b>	
Standard S2	100	2	3.5	50	10	<b>0.22 USD</b>	<b>160.60 USD</b>	
Standard S3	100	4	7	50	10	<b>0.44 USD</b>	<b>321.20 USD</b>	
Premium P1	100	1	1.75	250	20	<b>0.33 USD</b>	<b>240.90 USD</b>	
Premium P2	100	2	3.5	250	20	<b>0.66 USD</b>	<b>481.80 USD</b>	
Premium P3	100	4	7	250	20	<b>1.32 USD</b>	<b>963.60 USD</b>	
Premium v2 P1V2	210	1	3.5	250	30	<b>0.22 USD</b>	<b>160.60 USD</b>	
Premium v2 P2V2	210	2	7	250	30	<b>0.44 USD</b>	<b>321.20 USD</b>	
Premium v2 P3V2	210	4	14	250	30	<b>0.88 USD</b>	<b>642.40 USD</b>	

\*ACU/vCPU is an approximation of the SKU's relative performance. Add or remove favorites by pressing Ctrl+Shift+F. Learn more about App Service pricing.

4. After scaling up, wait for another five minutes, then check the performance chart on the dashboard in the Azure portal.



5. Return to the Cloud Shell that's running the client app. Select Enter to stop the app.

Link to our YouTube video: <https://youtu.be/Fxcf24ehqeA?si=ZW5AbQzCrx6L3v7w>