

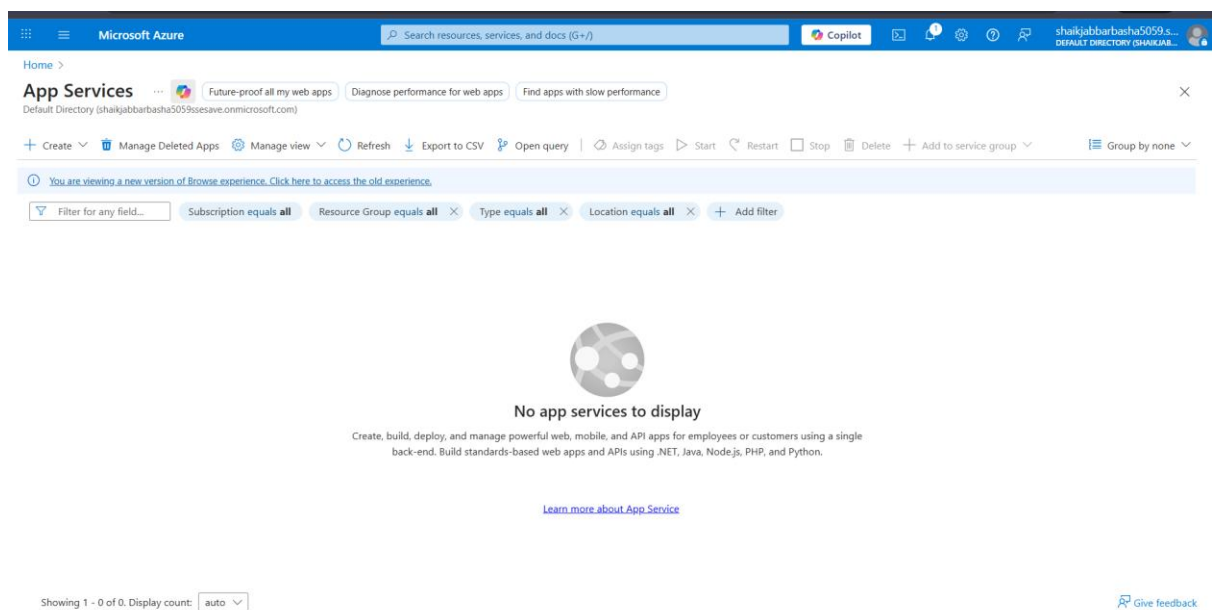
Aim

To create a simple web application using **Python (Flask)** and deploy it on a **Public Cloud Platform (AWS Elastic Beanstalk)** to demonstrate the concept of **Platform as a Service (PaaS)**.

Procedure

Step 1: Create a Simple Web Application

1. Install Python and Flask on the local system.
2. Create a project folder named `simple_web_app`.
3. Create a file named `app.py`.
4. Write a basic Flask application that displays a welcome message.
5. Test the application locally using a web browser.



Step 2: Prepare Application for Deployment

1. Create a `requirements.txt` file containing the Flask dependency.
2. Ensure the application listens on the default port provided by the cloud environment.
3. Verify that the application runs successfully in the local environment.

Microsoft Azure

Search resources, services, and docs (G+)

[Home](#) > [App Services](#) >

Create Web App

Basics

Database

Deployment

Networking

Monitor + secure

Tags

Review + create

Try Managed Instances (preview) on Azure App Service: a new option that delivers the platform benefits you rely on today, plus added features and flexibility to help you modernize applications seamlessly [Learn More](#)

App Service Web Apps lets you quickly build, deploy, and scale enterprise-grade web, mobile, and API apps running on any platform. Meet rigorous performance, scalability, security and compliance requirements while using a fully managed platform to perform infrastructure maintenance. [Learn more](#)

Project Details

Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * ⓘ

Azure for Students

Resource Group * ⓘ

(New) Resource group

[Create new](#)

Instance Details

Name

Web App name

.azurewebsites.net

☒

Secure unique default hostname on. [More about this update](#)

Publish *

☒ Code ☐ Container

Review + create

< Previous

Next : Database >

Step 3: Create an AWS Account

1. Sign up or log in to the **AWS Management Console**.
2. Navigate to **Elastic Beanstalk**, which is a PaaS service.
3. Choose **Create Application**.

Step 4: Deploy the Application on AWS Elastic Beanstalk

1. Select **Java** as the platform.

2. Upload the application source code (ZIP file).
3. Configure basic settings such as application name and environment.
4. Launch the environment.

Microsoft Azure

Search resources, services, and docs (G+)

[Home](#) > [App Services](#) >

Create Web App ...

Subscription ^{*} ⓘ

Azure for Students

Resource Group ^{*} ⓘ

resource_group

Create new

Instance Details

Name

resourcesapp

.azurewebsites.net

☒

Secure unique default hostname on. [More about this update](#) ⓘ

Publish ^{*}

☒ Code

☐ Container

Runtime stack ^{*}

Java 11

Java web server stack ^{*}

Java SE (Embedded Web Server)

Operating System ^{*}

☒ Linux

☐ Windows

Region ^{*}

Canada Central

ⓘ

Not finding your App Service Plan? Try a different region or select your App Service Environment.

Pricing plans

App Service plan pricing tier determines the location, features, cost and compute resources associated with your app.

[Learn more](#) ⓘ

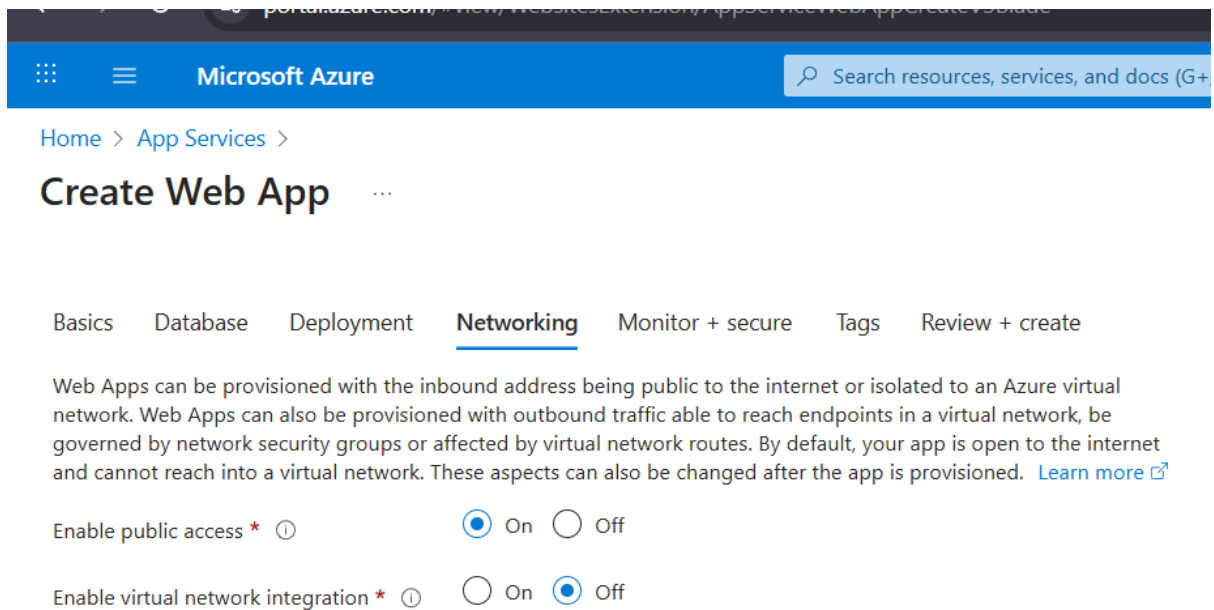
Review + create

< Previous

Next : Database >

Step 5: Access the Web Application

1. Wait for AWS Elastic Beanstalk to provision resources automatically.
2. Copy the generated **public URL**.
3. Open the URL in a web browser.
4. Verify that the web application runs successfully.



Step 6: Demonstrate PaaS Features

1. Observe that AWS manages:
 - Server provisioning
 - Operating system
 - Scaling
 - Load balancing
2. The developer focuses only on application code, demonstrating **Platform as a Service (PaaS)**.

Result

A simple Python-based web application was successfully created and hosted on **AWS Elastic Beanstalk**, demonstrating the working of **Platform as a Service (PaaS)**.

Create Web App

Details

Subscription	c552dc0f-ec3d-4ed4-9b72-b669503ca0d1
Resource Group	resource_group
Name	resourcesapp
Secure unique default hostname	Enabled
Publish	Code
Runtime stack	Java 11
Java web server stack	Java SE (Embedded Web Server)

App Service Plan (New)

Name	ASP-resourcegroup-942c
Operating System	Linux
Region	Canada Central
SKU	Basic
Size	Small
ACU	100 total ACU
Memory	1.75 GB memory

Monitor + secure (New)

Application Insights	Enabled
Name	resourcesapp
Region	Canada Central

[Create](#)[< Previous](#)[Next >](#)[Download a template for automation](#)