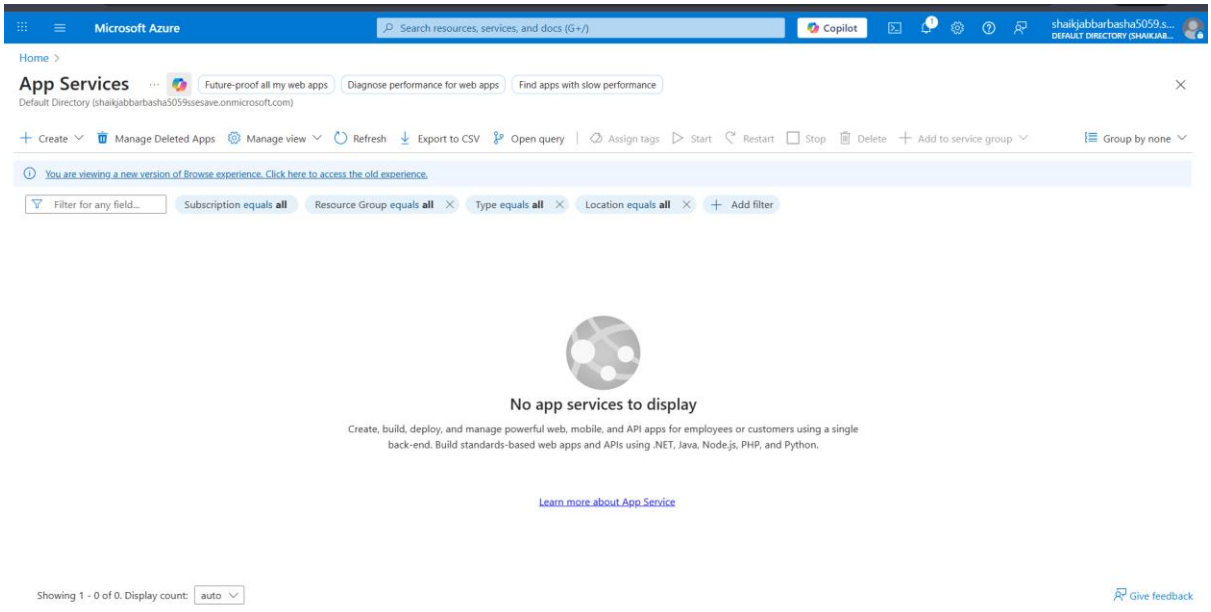
Aim

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

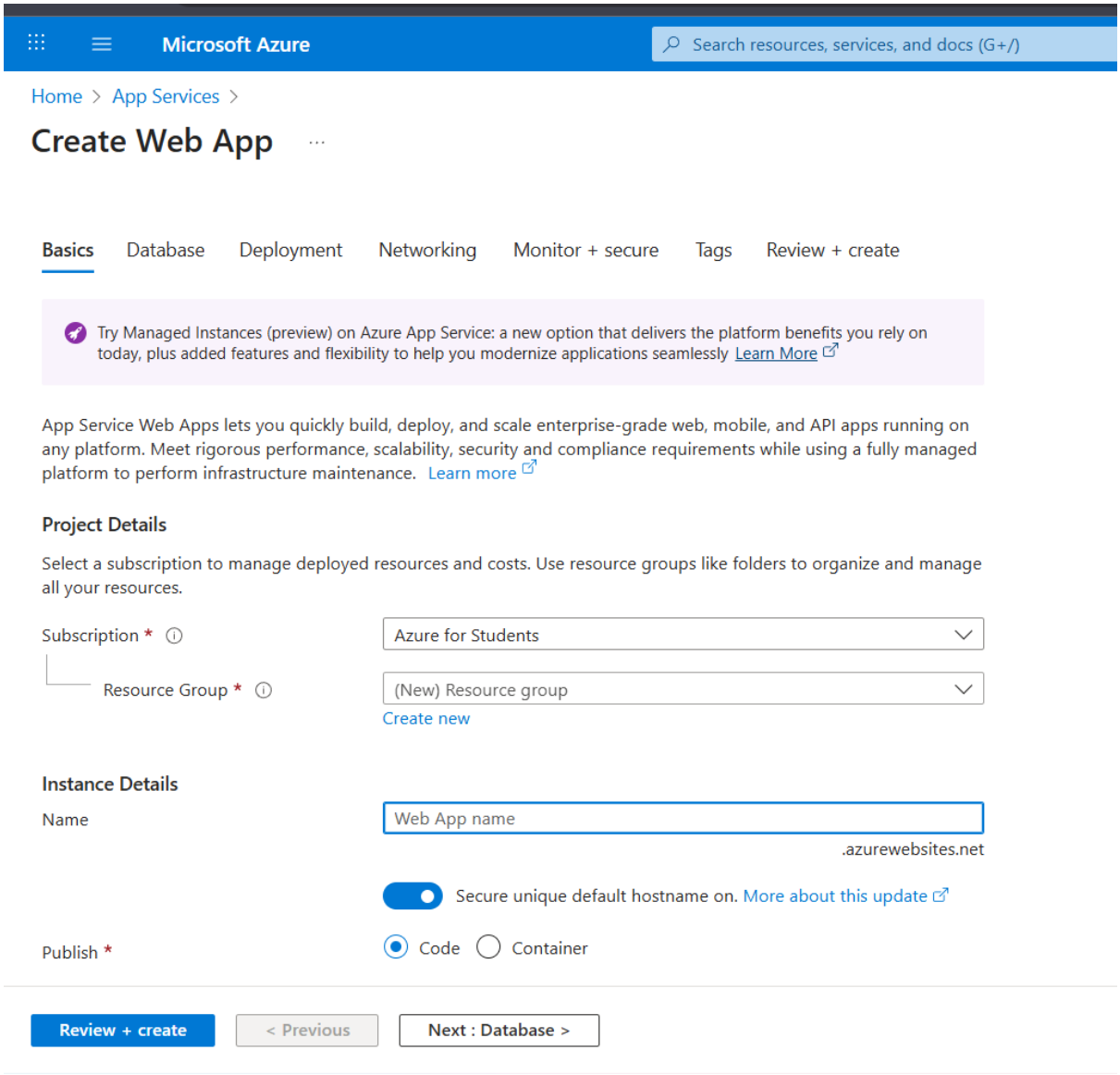
Step 1: Create a Simple Web Application

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.



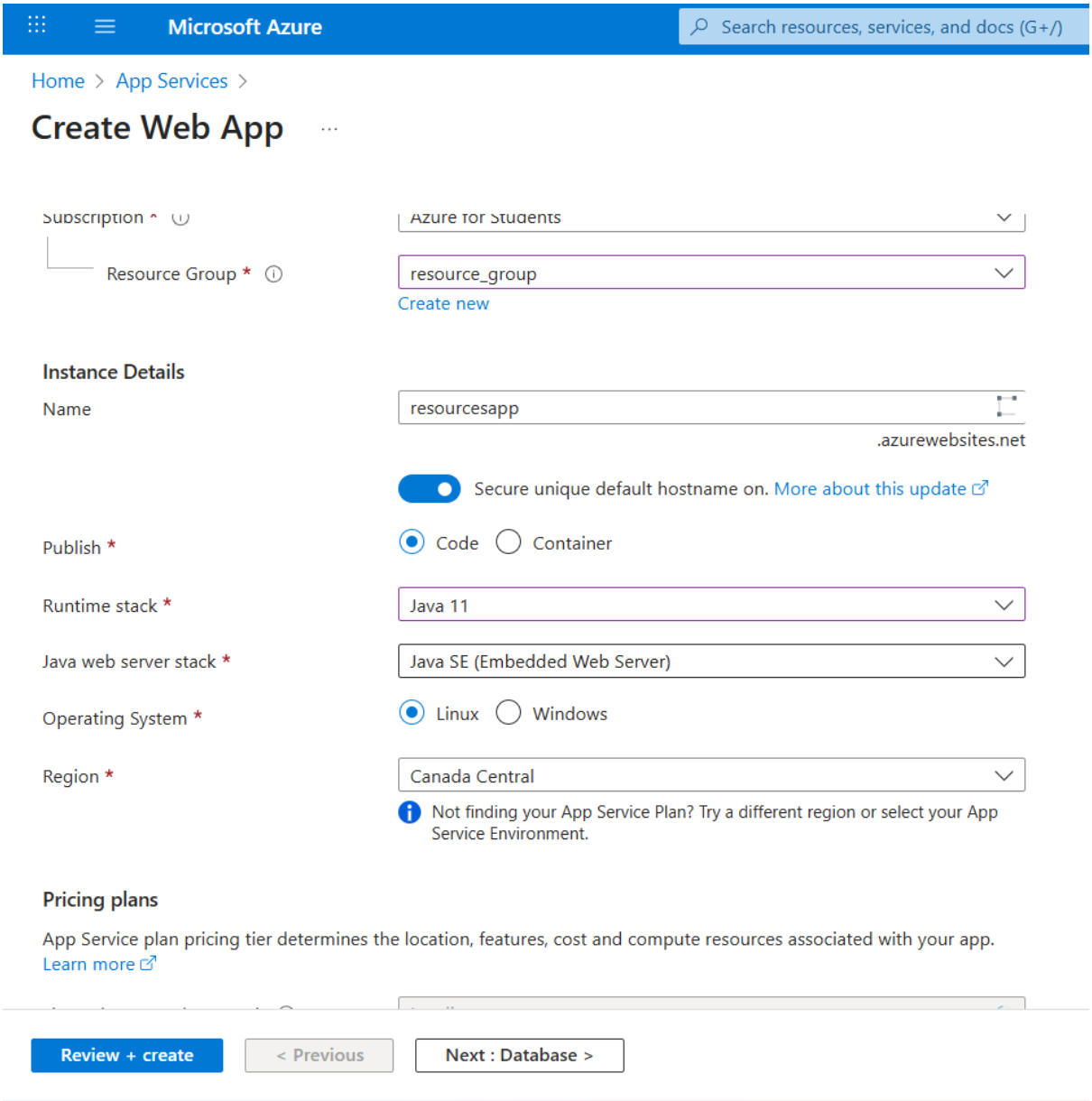
Step 3: Create an AWS Account

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

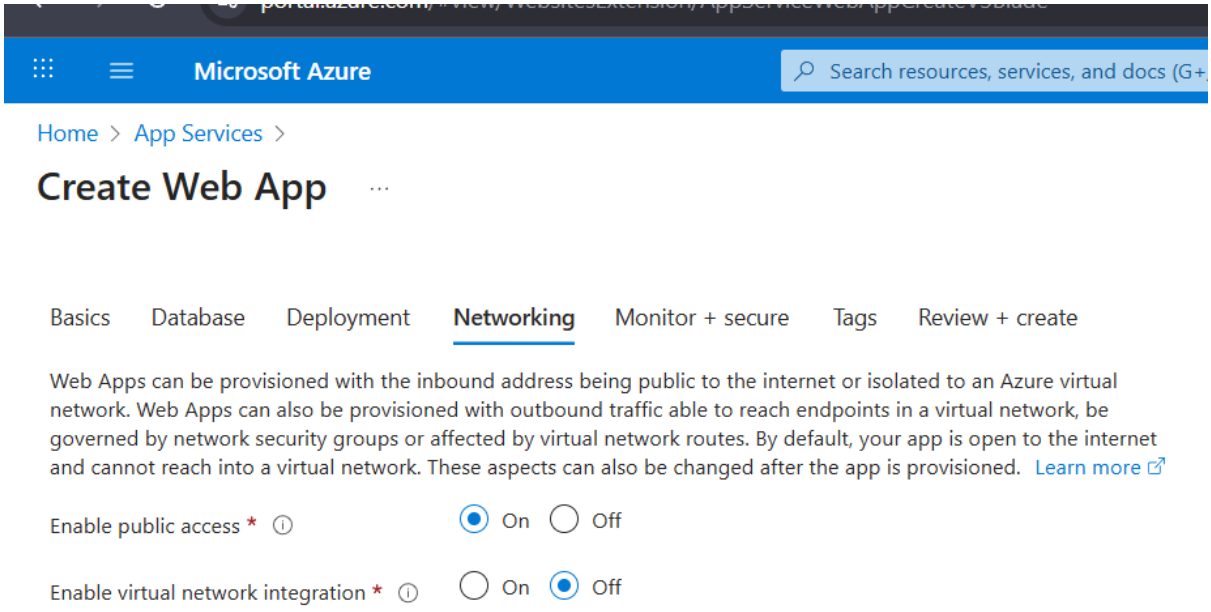
1. Select **Java** as the platform.

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



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)**.