Azure Cloud Exercises

Exercise 1: Create and Configure a Virtual Machine

Objective: Create and configure Ubuntu and Windows Virtual Machines on Azure Portal.

1. Create an Ubuntu VM:

o Log in to the Azure Portal.

o Navigate to Virtual Machines > Create.

o Choose Ubuntu Server 20.04 LTS.

o Configure:

▪ Size: Standard_B1s (or similar)

▪ Authentication Type: SSH (generate a key pair if not available).

▪ Inbound Port: Allow SSH (port 22).

o Deploy and connect using SSH.

2. Create a Windows VM:

o Follow similar steps, selecting Windows Server 2022.

o Configure:

▪ Size: Standard_B1s (or similar)

▪ Authentication Type: Username and Password.

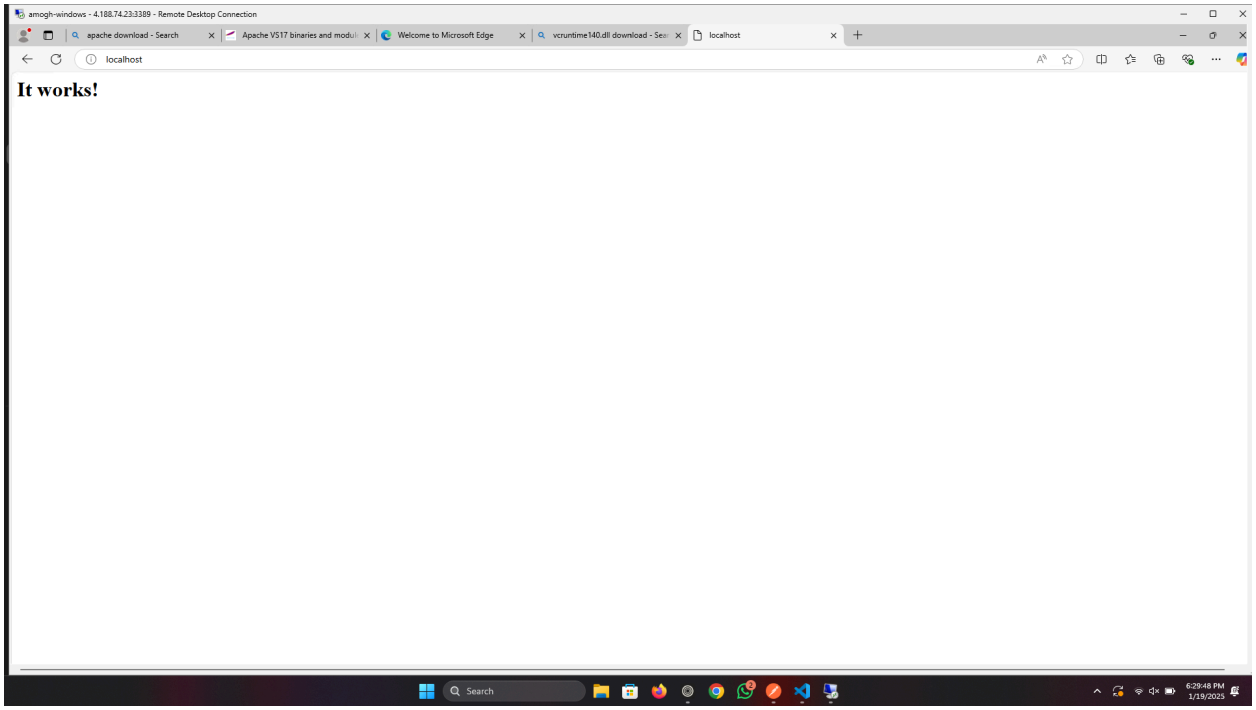
▪ Inbound Port: Allow RDP (port 3389).

o Deploy and connect using RDP.

3. Task:

o Install Apache or IIS on the respective VMs.

o Verify by accessing the default web page from your local browser.



Exercise 2: Deploy a Static Web Application

Objective: Host a static website using Azure App Service.

1. Navigate to App Services > Create.
2. Choose:
o Runtime Stack: Python 3.10 (or latest).
o Operating System: Linux.
o Region: Closest to your location.
3. Deploy the application.
4. Upload a simple static website (e.g., index.html and CSS files) using FTP or the Kudu console.
5. Task:
o Verify the deployment by accessing the site via its public URL.
o Modify the HTML to include a message like: "Welcome to Azure Static Web Apps!"

Welcome to Azure Static Web Apps!

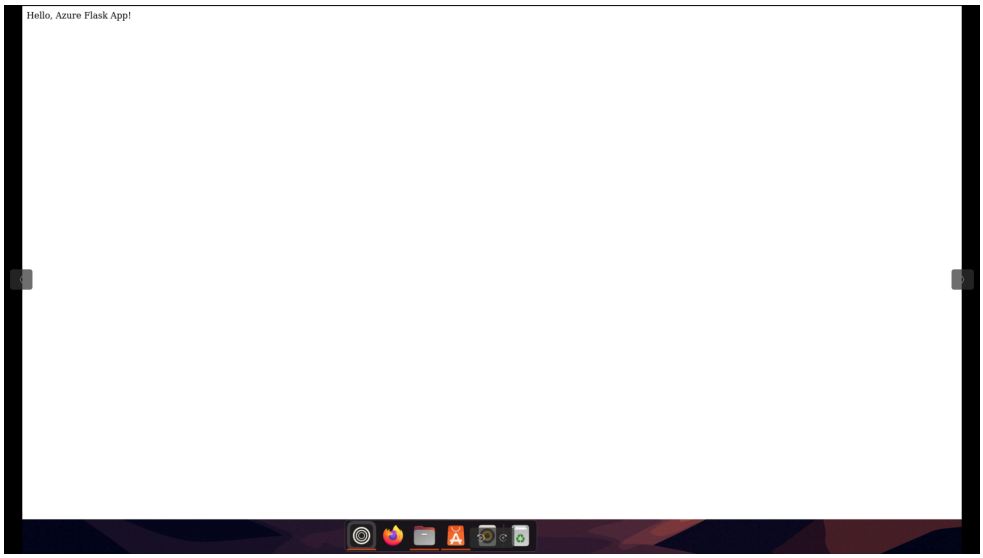
Exercise 3: Deploy a Flask Application (Dynamic Web App)

Objective: Deploy a Python Flask application using Azure App Service.

1. Create a Flask app:

```
from flask import Flask
app = Flask(__name__)
@app.route('/')
def home():
 return "Hello, Azure Flask App!"
if __name__ == '__main__':
 app.run(debug=True)
```

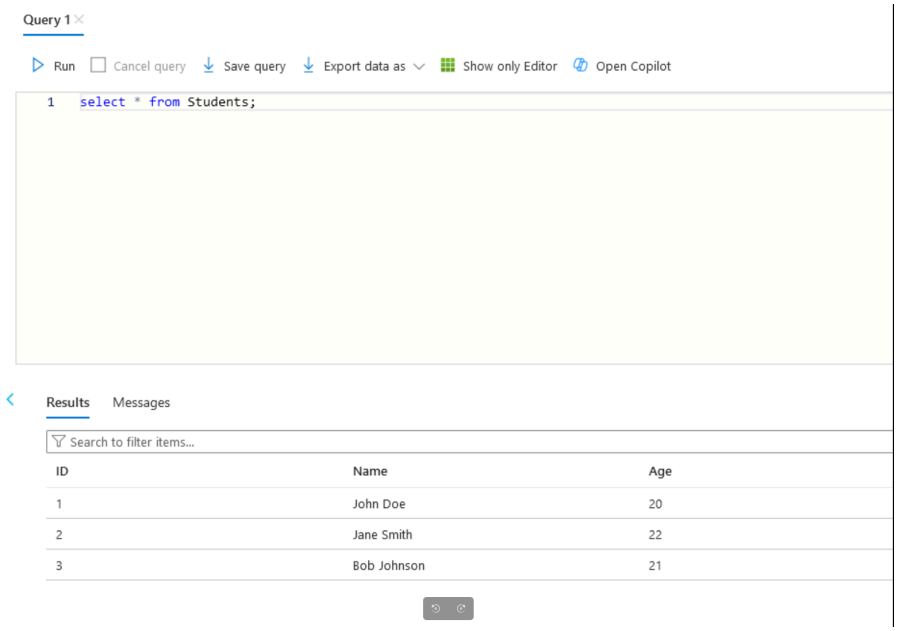
2. Push the code to a GitHub repository.
3. In the Azure Portal, navigate to App Services > Create.
4. Configure:
o Runtime Stack: Python 3.10 (or latest).
o Deployment Source: Connect your GitHub repository.
5. Deploy the Flask app and verify it by accessing the public URL.



Exercise 4: Set Up and Use an Azure SQL Database

Objective: Create an Azure SQL Database and connect to it from your local machine.

1. Navigate to SQL Databases > Create.
2. Configure:
o Database Name: StudentDB.
o Server: Create a new server with username and password.
o Compute + Storage: Use the free tier.
3. Deploy the database.
4. Connect using Azure Data Studio or SQL Server Management Studio (SSMS).
5. Task:
o Create a table Students with columns ID, Name, and Age.
o Insert sample data and query it.



Exercise 5: Integrate Flask App with Azure SQL Database

Objective: Connect a Flask app to Azure SQL Database and perform CRUD operations.

1. Use the Flask app from Exercise 3.
2. Install required libraries:

pip install flask pyodbc

3. Modify the app to connect to the SQL Database:

```
import pyodbc
conn = pyodbc.connect(
 'DRIVER={ODBC Driver 17 for SQL Server};'
 'SERVER=<your_server>.database.windows.net;'
 'DATABASE=StudentDB;'
 'UID=<your_username>;'
 'PWD=<your_password>'
)
cursor = conn.cursor()
```

4. Add a route to fetch and display data from the Students table.
5. Deploy the updated app to Azure App Service.
6. Task:

o Verify CRUD functionality by interacting with the app via its public URL.