Project Title: Azure-Based Data Ingestion, Processing, and Web App Deployment Pipeline

📁 SCENARIO 1: Create Linux VM and Store Daily Logs

🔷 Objective:

Log daily user activity and store it as a log file.

💡 Architecture:

User Activity → Linux VM → Cron Job → /var/logs/auth.log-<date>.log

🛠️ Implementation:

- Provision a **Linux VM** from Azure.
- Create a logUserActivity.sh script that logs user activity (e.g., who, uptime, etc.).
- Use crontab to run it daily:

0 2,14 * * * sudo cp -f /var/log/auth.log /home/azureuser/logfile.log (Executes twice in a day at 2 am and 2 pm)

📁 SCENARIO 2: Upload Logs to MySQL Table VMLogs

🔷 Objective:

Send daily log data to a MySQL database table.

💡 Architecture:

Linux VM → Cron Job → Python Script → MySQL Table (VMLogs)

🛠️ Implementation:

- Install MySQL client and Python connector on the VM.
- Python script reads from log file and inserts into table VMLogs.
- Cron job entry:

0 18 * * * /usr/bin/python3 /home/azureuser/scripts/upload_logs.py (execute at 6 pm daily)

📁 SCENARIO 3: Copy Blob Storage Data to MySQL using ADF

🔷 **Objective:**

Move CSV file from Blob Storage to MySQL Products table using Azure Data Factory.

💡 **Architecture:**

Azure Blob (sample.csv)

↓

Azure Data Factory Pipeline

↓

Azure SQL/MySQL Table (Products)

🛠️ **Steps:**

1. **Blob Storage Setup**
    - Create a **Storage Account** in Azure
    - Create a **Container** named raw
    - Upload sample.csv
2. **Azure SQL/MySQL Setup**
    - Create table Products:
3. **Azure Data Factory**
    - Create linked services:
        - Azure Blob Storage
        - Azure SQL/MySQL
    - Create dataset for source (sample.csv)
    - Create dataset for sink (MySQL table)
    - Build and trigger a **Copy Data pipeline**

📁 **SCENARIO 4: Web App Deployment with GitHub and Azure App Services**

🔷 **Objective:**

Deploy a web app that reads from ProcessedData table and displays it.

💡 **Architecture:**

User Browser

↓

Azure Web App (Python + Flask)

↓

MySQL Database (ProcessedData)

↓

GitHub (CI/CD deployment)

🛠️ **Implementation Steps:**

1. **Web App Code (Flask + MySQL)**
2. **Push Code to GitHub**
    - Create a new GitHub repo.
3. **Azure Web App Setup**
    - Create an App Service (Linux, Python runtime)
    - In Deployment Center, link to your GitHub repo and enable **Auto-Deployment**
4. **Configure App Settings**
    - Add environment variables for DB credentials in App Service → Configuration
5. **Run the App**
    - App service will fetch data from ProcessedData and render it.