Level 5 Data Engineer

Hackathon

Developing a Cloud-Based Data Dashboard Using Open Data and Tools

---

Scenario

Your task is to design and deploy a simple cloud-based data dashboard that provides insights from an open dataset. You will use readily available datasets from platforms like Kaggle, leverage open-source code from GitHub, and use official Azure online tutorials to guide your implementation.

NB: You may not be able to complete all of the steps, only complete the steps that you can and avoid getting stuck on any particular step for too long (best idea just to move on to the next step). Document your learning and challenges as you go along.

Objective

Create a Python-based dashboard that visualises data from an open dataset, deploying it to the cloud, and ensuring it aligns with best practices in cost monitoring, auto-scaling, and continuous monitoring.

Available Resources

- **Open Datasets:** Choose a dataset from Kaggle that interests your team. Examples include datasets on global temperatures, COVID-19 statistics, or stock market data.
- **Open-Source Code:** Utilise existing code repositories on GitHub that provide templates or components for data dashboards. Frameworks like **Streamlit** or **Dash** can help you build interactive dashboards quickly. You can use any other dashboarding solution if you want (it does not have to be open-source, but document why you chose not to follow the open-source route if necessary).
- **Azure Tutorials:** Follow official Azure tutorials to assist with deployment:
    - Quickstart: Deploy a Python web app to Azure
    - Tutorial: Monitor a web app using Azure Monitor

Hackathon Tasks

1. **Select and Prepare a Dataset**
    - **Choose a Dataset:**
        - Browse Kaggle for a dataset that is manageable and relevant.
        - Ensure the dataset is not too large to process within the time frame.

- **Data Preparation:**
  - Download and inspect the dataset.
  - Perform necessary data cleaning using Python libraries like `pandas`.
  - Handle missing values, data types, and formatting.

2. **Develop the Dashboard**
   - **Set Up the Environment:**
     - Create your environment and install required packages (`streamlit`, `pandas`, `matplotlib`, etc.).
   - **Build the Dashboard:**
     - Use for example **Streamlit** or **Dash** to create an interactive dashboard.
     - Incorporate at least two visualisations (e.g., line chart, bar graph).
     - Add interactive elements like sliders or filters to enhance user experience.
   - **Code Reuse:**
     - Leverage open-source code from GitHub repositories to accelerate development.
     - Example repositories:
       - Streamlit Examples
       - Dash Sample Apps
3. **Deploy the Dashboard to Azure**
   - **Follow Azure Tutorials:**
     - Use the Quickstart guide to deploy your app using Azure App Service.
   - **Deployment Steps:**
     - Create an Azure account (use free student credits if available).
     - Use **Git** for version control and integrate with Azure deployment.
     - Configure deployment settings and environment variables as needed.
   - **Infrastructure as Code (Optional):**
     - If time permits, use Azure Resource Manager templates or **Azure CLI** scripts to define your infrastructure.
4. **Monitor Costs and Implement Auto-Scaling**
   - **Cost Monitoring:**
     - Access the Azure Cost Management tool to monitor resource usage.
     - Ensure you are using free tiers or minimal resources to keep costs low.
   - **Auto-Scaling Considerations:**
     - Discuss how auto-scaling could be implemented to handle increased load.
     - Understand Azure's auto-scaling features, even if not fully implemented during the hackathon.
5. **Set Up Continuous Monitoring**
   - **Enable Monitoring:**
     - Use Azure Monitor to track application performance.
   - **Configure Basic Alerts:**
     - Set up alerts for critical metrics like application downtime or high CPU usage.
   - **Dashboard Visualisation:**

        - Create a basic monitoring dashboard within Azure to visualise these metrics.
6. **Documentation and Presentation**
    - **Prepare Documentation:**
        - Summarise the steps taken, challenges faced, and solutions implemented.
        - Include code snippets or links to your GitHub repository.
    - **Presentation:**
        - Create a brief presentation or demo to showcase your dashboard.
        - Highlight key features, insights gained from the data, and how you addressed cost monitoring and sustainability considerations.

Example Project Outline

- **Dataset:** Global Temperature Time Series
- **Dashboard Framework:** Streamlit
- **Visualisations:**
    - Line chart showing temperature changes over time.
    - Map displaying average temperatures by country.
- **Deployment:**
    - Deployed on Azure App Service using the free tier.
    - Followed the Azure Quickstart guide for deployment.
- **Monitoring:**
    - Enabled Azure Monitor to track app availability and performance.
    - Set up an alert for app downtime.

Additional Tips

- **Time Management:**
    - Allocate time effectively: 30 minutes for dataset selection and preparation, 1.5 hours for development, and 1 hour for deployment and testing.
- **Collaboration:**
    - Divide tasks among team members based on skills:
        - Data preparation and visualisation.
        - Dashboard development.
        - Deployment and monitoring setup.
- **Use Simple Solutions:**
    - Focus on core functionality rather than advanced features.
    - Use pre-built components and templates to save time.
- **Testing:**
    - Test the dashboard locally before deploying.

- Ensure all dependencies are included in requirements files.