Subjective Test

Q1. What is Streamlit and what are its main features?

Ans: Streamlit is a popular open-source app framework used to create and share data apps. It's particularly well-suited for machine learning and data science applications, allowing users to build and deploy interactive web applications with minimal effort.

Its main features include:

1. **Ease of Use:** Write apps in pure Python with minimal effort.
2. **Interactive Widgets:** Integrate sliders, buttons, and other widgets easily.
3. **Auto-Refresh**: Automatically updates the app when the source code changes.
4. **Data Visualization**: Integrates with popular data visualization libraries like Matplotlib, Plotly, and Altair.
5. **Deployment Options:** Simple deployment via Streamlit Sharing, Heroku, AWS, and more.

Q2. How does Streamlit differ from other web application frameworks like Flask or Django?

Ans: **Purpose:** Streamlit is designed specifically for data apps, while Flask and Django are general-purpose web frameworks.

Simplicity: Streamlit allows rapid development of interactive data visualizations without needing HTML, CSS, or JavaScript, unlike Flask and Django.

Real-Time Updates: Streamlit apps automatically update with code changes, which is not a built-in feature of Flask or Django.

Q3. What are some typical use cases for Streamlit?

Ans: **Data Exploration:** Quickly visualize and interact with datasets.

Prototyping ML Models: Develop and share machine learning models with interactive interfaces.

Dashboards: Create real-time dashboards for monitoring data and metrics.

Educational Tools: Build interactive tutorials and demonstrations for data science concepts.

Q4. How do you create a simple Streamlit app?

Ans: **Step1**: Install Streamlit using pip:

```
pip install streamlit
```

Step2: Create a Python script (app.py):

```
import streamlit as st

st.title("Hello, Streamlit!")

st.write("This is a simple Streamlit app.")
```

Step3: Run the app:

```
streamlit run app.py
```

Q5. Can you explain the basic structure of a Streamlit script?

Ans: A basic Streamlit script includes:

Importing Streamlit: import streamlit as st

Defining the App Layout: Using functions like st.title(), st.write(), st.dataframe(), etc.

Adding Widgets: Interactive elements like sliders and buttons using st.slider(), st.button(), etc.

Q6. How do you add widgets like sliders, buttons, and text inputs to a Streamlit app?

Ans:

```
import streamlit as st
slider_value = st.slider('Select a range of values', 0, 110, (10, 64))
button_clicked = st.button('Click me')
text_input = st.text_input('Enter your name')
st.write(f"Slider value: {slider_value}, Button clicked: {button_clicked}, Name: {text_input}")
```

Q7. How does Streamlit handle user interaction and state management?

Ans: Streamlit uses session state to manage user interaction. Widgets automatically maintain their state across reruns. We can use the st.session_state API for more complex state management.

Q8. What are some best practices for organizing and structuring a Streamlit project?

Ans: **Modularize Code**: Split code into functions and modules.

Use a Config File: Store configurations in a separate file.

Version Control: Use Git for version control.

Documentation: Add comments and documentation to the code.

Q9. How would you deploy a Streamlit app locally?

Ans: Run the app using the command:

streamlit run your_script.py

We must ensure the necessary dependencies are installed in our local environment.

Q10. Can you describe the steps to deploy a Streamlit app?

Ans: Streamlit provides several options for deployment, including streamlit sharing, Heroku, AWS and others.

Streamlit Sharing is the easiest way to deploy streamlit apps. It requires a GITHUB account.

1. Push your streamlit app to a github repository.
2. Go to streamlit sharing and sign-in with your github account.
3. Click on ‘New App’ & select the repository nad branch where your app is located.
4. Click deploy.

Q11. What is the purpose of the requirements.txt file in the context of Streamlit deployment?

Ans: The requirements.txt file lists all the dependencies required by our Streamlit app. This ensures that the deployment environment has all necessary packages installed, ensuring consistent behaviour across different setups.