Streamlit – Deployment Module

Subjective test

Varnika Mulay

1. What is Streamlit and what are its main features?
   Ans: Streamlit is an open-source framework designed for data scientists and analysts who want to create and deploy apps without using HTML, JavaScript, or web development languages. It enables developers to build attractive user interfaces for their Python projects.
   The main Features are:
   - Web app development may be done quickly with this simple and easy-to-use API.
   - The program can automatically update itself when the source code is updated or modified.
   - It can be easily deployed using cloud or local platforms.
   - Offers a variety of interactive widgets for creating interactive user interfaces, such as text inputs, buttons, sliders, etc.
   - It also supports integration with various Python visualization libraries, such as Matplotlib, Seaborn, and Plotly, for better data representation.

2. How does Streamlit differ from other web application frameworks like Flask or Django?
   Ans:
   - Streamlit is mainly designed for specific Machine Learning and Data Science applications, whereas Flask and Django are designed for general web development projects.
   - Streamlit is also more user-friendly than Flask and Django as it is easier for developers to create and deploy their apps.
   - The main focus of Streamlit is to simplify the process of app development for data scientists and developers hence, it mainly focuses on creating interactive dashboards and visualizations to represent the data analysis.

- Another main difference is that Streamlit has a feature that enables it to automatically update/ modify its UI when the developer makes changes in the code.

3. What are some typical use cases for Streamlit?
   Ans:
   - Machine Learning: Developing and deploying various ML models with user-interactive capabilities.
   - Data analysis: Creating interactive visualizations and dashboards to represent the various patterns and relationships in the dataset.
   - Reporting: Creating interactive reports that help the users understand the data and its features.

4. How do you create a simple Streamlit app?
   Ans: To create a simple streamlit app we need to follow the following steps:
   - Install streamlit using the ‘pip install streamlit’ command.
   - Create a new python file say “hello.py”.
   - Import the necessary library using ‘import streamlit as st’ command.
   - Write ‘st.write(‘Hello world!’)’ in the file and save it.
   - Run the file using ‘streamlit run hello.py’ command and it will run on localhost.

5. Can you explain the basic structure of a Streamlit script?
   Ans:
   - A basic Streamlit script will contain the necessary libraries like streamlit library and other libraries according to the user’s need.
   - It could also contain a write command (st.write()) to display the value on the UI.
   - If the user has any specific dataset to analyze, we will have to use pandas library and create a dataframe for further analysis.
   - It could contain various plots like bar graphs and line plots to display the data representations or trends.
   - If we want to make an interactive app, we’ll have to include widgets like buttons, switches, etc as well.

6. How do you add widgets like sliders, buttons, and text inputs to a Streamlit app?
   Ans: To add widgets like sliders, buttons, etc. streamlit provides various functions like:
   - Slider: st.slider(label, min_value, max_value, value)
   - Button: st.button(label)
   - Text Input: st.text_input(label)
   - Selectbox: st.selectbox(label, options)

7. How does Streamlit handle user interaction and state management?
   Ans: Streamlit handles user interactions using its interactive widgets through which the user can input data. With every new entry, the script will rerun the entire code to modify other dependent values (like graphs) from top to bottom. Streamlit also provides a function called 'st.session_state' which helps the user in state management.

8. What are some best practices for organizing and structuring a Streamlit project?
   Ans: The best practices for organizing and structuring a Streamlit project are:
   - Divide the script into various functions and classes according to their tasks so that the code is readable and arranged properly.
   - Use comments wherever possible for better understanding of the code.
   - Use the Streamlit functionalities like columns or pages which can arrange the app in a better way.
   - Use error-handling methods like try-except blocks to handle errors (if any) in a more readable way to the users.

9. How would you deploy a Streamlit app locally?
   Ans: We can deploy a Streamlit app locally using the 'streamlit run {filename}.py' command. This command will automatically redirect the app to the default web browser and will run on localhost. If it does not redirect, the user can enter the path provided by the terminal on the web browser which will run the app.

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

Ans: To deploy a Streamlit app on a free cloud service we can follow the following steps:

- Upload the main code on GitHub.
- On the UI click on the ‘Deploy’ option and enter the repository details, GitHub url, and the necessary details.
- Click deploy and the app will be deployed on Streamlit Community cloud.
- If you want to deploy it on some other platform like AWS, choose that option and configure it. After successful configuration, push your code in it and deploy it.

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

    Ans: The requirement.txt file is a file containing all the libraries and packages used to run your Streamlit script. When the user deploys the app, the deployment platform will read the file and install all the necessary libraries and packages required for the app to execute successfully.