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

Streamlit is a Python library that aims to make it easy to build interactive web applications for data analysis and machine learning. Main Features

- API which is easy to utilize

- Immediate updates

- Widgets that allow for interaction (sliders, buttons...)

- Charting libraries (like Matplotlib, Plotly)

- Simplified deployment methods

- Configurable design

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

- Objective : Streamlit suits for data science applications; Flask and Django suit for general-purpose websites.

- User Friendliness : To achieve interactivity in Streamlit, you need very little code.

- Combination : Data science libraries blend well with Streamlit.

- Rate of Progress : With Streamlit, prototyping becomes faster.

- Control of The Situation : Unlike Flask/Django, state management is simpler in Streamlit

3. What are some typical use cases for Streamlit?

Machine learning demonstrations: Creating demonstrations for machine learning models that are interactive.

Data visualization: Constructing dashboards to help visualize complex data sets.

Exploratory data analysis (EDA): Building applications that allow for interactive exploration and analysis of datasets.

Prototyping: Creating quick models of new ideas that use data and sharing them with people who need to know about it before investing any time or money into it.

Experiment tracking: A visual display showing the results of various experiments side by side.

Educational tools: A syllabus and a demonstration are included in it

4. How do you create a simple Streamlit app?

1. Install Streamlit: pip install streamlit

2. Create a Python Script: For instance, app.py.

3. Write the app’s code:

import streamlit as st

st.title('Hello Streamlit!')

st.write('This is a simple Streamlit app.')

4. To run the app, you should execute `streamlit run app.py` in the terminal.

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

Imports : Import Streamlit and the rest of the libraries.

- Title and Headers : `st.title()`, `st.header()`

- Text and Markdown : `st.write()`, `st.markdown()`

- Widgets : `st.button()`, `st.slider()`

- Visualizations : `st.pyplot()`, `st.plotly\_chart()`

- Layout : `st.sidebar()`, `st.columns()`

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

- Slider :

age = st.slider('Select your age', 0, 100, 25)

st.write('Age:', age)

- Button :

if st.button('Click Me'):

st.write('Button clicked!')

- Text Input :

name = st.text\_input('Enter your name')

st.write('Name:', name)

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

- Across different runs it is important that Widgets maintain their states so that they can be used again in subsequent ones. - `st.session\_state` should be used so as not to lose any information stored about them during our interactions with them or while fetching new content from another page of our website;

- When we need some data at different points of execution where same function is required multiple times (i.e., for example getting values needed by more than one operation), then it may become quite slow because, being expensive computations, they require much time. In this case we will cache these values

- If we want certain computations pre-computed instead of being recalculated at every event update but still preserve our statefulness even after page reloads(issue number), one might use the following: <code>`st.cache` and `st.cache\_data`.

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

- Split up your code into modules and functions.

-Separate your parameters into configuration files.

-Make use of Git for version control.

-Whenever you write some code, always remember to write some comments or documentation so that someone else who looks at it could easily understand what you were trying to achieve with that piece of work.

-Write tests as necessary whenever possible as this will help you avoid making any mistakes while coding.

-Always work within virtual environment while handling any project tasks involved in installation on multiple setups

9. How would you deploy a Streamlit app locally?

1. Install Dependencies : `pip install -r requirements.txt`

2. Run the App : `streamlit run app.py`

3. Access Locally : Open `http://localhost:8501`

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

1. Prepare the Application: Verify that it is working on your machine.

2. List all dependencies in 'requirements.txt'.

3. Specify the Platform: Streamlit Cloud, Heroku, etc.

4. To release: - For Streamlit Cloud: Connect your repository in GitHub. - On Heroku: Create a file named Procfile; then add web: streamlit run app.py and push it to Heroku.

5. Should you need them, provide Environmental Variables.

6. Verify online that it was deployed well

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

Streamlit app could not have been made successfully across various platforms without setting it up in the right way because you would want it to run without hitches and this is only possible when using a platform where everything required has been installed as indicated in the corresponding file