1. What is Streamlit and what are it's main features?

- Streamlit is an open-source Python library designed for creating and sharing custom web applications for machine learning and data science projects
- Main Features :
    - Ease of Use
    - Data Visualization
    - Layout and Styling
    - Easy Deployment
    - Extensibility

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

- Streamlit Vs Flask
    - Streamlit is primarily used for quick building web apps for data science and machine learning, while micro web framework designed for building web app and API's.
    - Flask is requires relatively more setup compared to Streamlit.
    - Streamlit comes with inbuilt widgets while Flask Doesnt.

- Streamlit Vs Django
    - Streamlit focuses on creating web apps for data science and machine learning while Django is a full stack framework designed for building more complex websites
    - Django requires more setup and config as compared to Streamlit
    - In Streamlit visualization is better than Django.

3. What are some typical use cases for Streamlit?

- Data Exploration and Visulization
- Machine Learning Model Prototyping
- Interactive reports and presentation
- Data analysis tools
- NLP

4. How do you create a simple Streamlit app?

- Step-1 : Install Streamlit
    - pip install streamlit

- Step-2 : Create a Py Script

- Step-3 : Run the Streamlit app
    - streamlit run app.py

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

- A basic Streamlit script consist of : Imports, App title, Texts, Widgets, Data, VIsualization, Layout etc

- Imports :
    - Import streamlit as st
    - Import pandas as pd

- App Title :
    - st.title(“My Streamlit App”)

- Text :
    - st.text(“This is an Streamlit App”)

- WIdgets :
    - Number = st.slider(“select a number”, 1, 25, 60)

- Data :
    - Data = pd.DataFrame({ data)}

- Visulization :
    - st.line_chart(data)

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

- Adding Sliders :
    - import streamlit as st
    - slider_value = st.slider("Select a number", min_value=0, max_value=200, value=50, step=1)
    - st.write("Slider value:", slider_value)

- Adding Buttons :
    - if st.button("Click me"):
    - st.write("Hello World!!")
    - else:
    - st.write(“You Didnt Click the button.")

- Adding Text Inputs :
    - Text_input = st.text_input(“Enter Some Text”)
    - st.write(Text_input)

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

- Streamlit handles user interaction by using widgets and reactive programming. It provides a variety of widgets that are easy to understand and implement and it also uses reactive programming model i.e user can interact with scripts to rerun them.

- While it handles state management by Session State i.e Streamlit allows to preserve info across reruns and user can initialize session state variables with default values and update them based on user interaction.

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

- Use Project directory to separate different components and resources
- Split code into modules to improve readability and reusability
- Use Session state
- Provide Documentation
- Have a good Error handling
- Use Version Control

9. How would you deploy a Streamlit app locally?

- Step-1 : Install Python
- Step-2 : Install Streamlit
- Step-3 : Create Streamlit App
    - import streamlit as st
    - import pandas as pd
    - import numpy as np
    -
    - st.title("Streamlit App")
    - st.header("Welcome to my Streamlit app")
    -
    - slider_value = st.slider("Select a number", 0, 100, 50)
    - st.write("Slider value:", slider_value)
- Step-4 : Run it locally
    - streamlit run app.py

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

- Step-1 : Install Python
- Step-2 : Install Streamlit
- Step-3 : Create Streamlit App
    - import streamlit as st
    - import pandas as pd
    - import numpy as np
    -
    - st.title("Streamlit App")
    - st.header("Welcome to my Streamlit app")
    -
    - slider_value = st.slider("Select a number", 0, 100, 50)
    - st.write("Slider value:", slider_value)
- Step-4 : Run it locally
    - streamlit run app.py

- Now you can deploy it to a hosting service such as Streamlit Community Cloud, Heroku etc

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

- The requirements.txt files serves a crucial role in the deployment of apps created with streamlit.
- It's main purpose includes Dependency Manangement and Reproducibility.
- It also helps in Deployment automation.