

1. Assignment(hands on): My First Streamlit app

i. Tools / Libraries Used

- **Python** – Core programming language
- **Streamlit** – Web app framework for building interactive applications
- **Pandas** – Data manipulation and DataFrame handling

ii. Project Description

The **Smart Interactive Streamlit Dashboard** is a multi-feature web application built using Streamlit.

This application demonstrates various interactive components such as:

- Sidebar course selection
- Profile display section
- User input handling (text input & number input)
- Buttons and checkbox interactions
- Programming language selection
- Session-based counter functionality
- DataFrame display using Pandas
- CSV file upload and preview
- Image display from URL

The project showcases how to build an **interactive, dynamic, and user-friendly web application** using Streamlit with minimal code. It integrates multiple UI elements and state management concepts, making it ideal for beginners learning web app development with Python.

iii. Project Code

```
import streamlit as st  
import pandas as pd  
  
# TITLE  
st.title("Integrated Streamlit Application")
```

```
# SIDEBAR MENU  
st.sidebar.title("Courses Menu")  
course = st.sidebar.selectbox(
```

```
"Select Course",
["Data Science", "Full Stack Java", "Full Stack Python", "Dot Net"]

)

st.sidebar.success(f"You selected {course}")

# PROFILE SECTION
st.header("My Profile")
st.write("Name: Kavyashree N")
st.write("Role: Data Science Intern")
st.write("Skills: Python, SQL, Machine Learning")

# - USER INPUT SECTION
st.header("User Input Section")

name = st.text_input("Enter your name")
age = st.number_input("Enter your age", min_value=0, max_value=100)

if st.button("Submit"):
    st.success("Button Clicked Successfully!")
    st.write(f"Hello {name}, you are {age} years old.")

# CHECKBOX SHOW/HIDE

if st.checkbox("Show Secret Message"):
    st.write("Welcome to Streamlit ")

# SELECTBOX
st.header("Programming Language Selection")

language = st.selectbox(
    "Choose Programming Language",
```

```
        ["Python", "Java", "C++", "JavaScript"]  
    )  
  
    st.write(f"You selected: {language}")  
  
# COUNTER  
st.header("Simple Counter")  
  
if "count" not in st.session_state:  
    st.session_state.count = 0  
  
if st.button("Increase Counter"):  
    st.session_state.count += 1  
  
st.write("Counter Value:", st.session_state.count)  
  
# DATAFRAME DISPLAY  
st.header("Display Sample DataFrame")  
  
data = {  
    "Name": ["Alice", "Bob", "Charlie"],  
    "Salary": [50000, 60000, 70000]  
}  
  
df = pd.DataFrame(data)  
st.dataframe(df)  
  
# CSV FILE UPLOAD  
st.header("Upload CSV File")  
  
file = st.file_uploader("Upload a CSV file", type=["csv"])
```

```

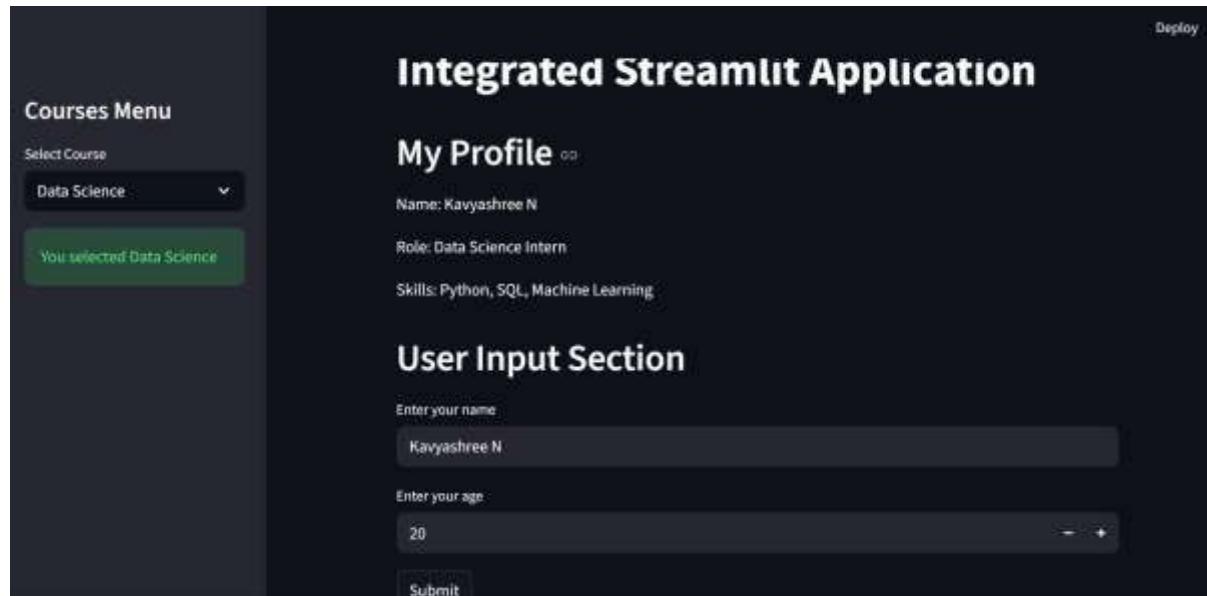
if file is not None:
    uploaded_df = pd.read_csv(file)
    st.write("Uploaded Data:")
    st.dataframe(uploaded_df)

# IMAGE DISPLAY
st.header("Display Image")

st.image(
    "https://images.unsplash.com/photo-1519389950473-47ba0277781c",
    caption="Sample Image",
    use_container_width=True
)

```

iv. OUTPUT:



Courses Menu

Select Course

Data Science

You selected Data Science

User Input Section

Enter your name

Kavyashree.N

Enter your age

20

Submit

Button Clicked Successfully!

Hello Kavyashree N, you are 20 years old.

Show Secret Message

Welcome to Streamlit

Deploy

This screenshot shows a Streamlit application interface. On the left, there's a sidebar titled 'Courses Menu' with a dropdown menu set to 'Data Science'. A green box below it says 'You selected Data Science'. The main area is titled 'User Input Section'. It contains two text input fields: one for 'Enter your name' with 'Kavyashree.N' typed in, and another for 'Enter your age' with '20' typed in. Below these is a 'Submit' button. A green box at the bottom displays the message 'Button Clicked Successfully!'. Underneath the input fields, there's a message 'Hello Kavyashree N, you are 20 years old.' followed by a checkbox labeled 'Show Secret Message' which is checked. At the bottom, a general welcome message 'Welcome to Streamlit' is shown. In the top right corner, there are 'Deploy' and 'Edit' buttons.

Courses Menu

Select Course

Full Stack Python

You selected Full Stack Python

Choose Programming Language

Python

You selected: Python

Simple Counter

Increase Counter

Counter Value: 3

Display Sample DataFrame

	Name	Salary
0	Alice	50000
1	Bob	60000
2	Charlie	70000

Deploy

This screenshot shows a Streamlit application interface. On the left, there's a sidebar titled 'Courses Menu' with a dropdown menu set to 'Full Stack Python'. A green box below it says 'You selected Full Stack Python'. The main area has a dropdown menu titled 'Choose Programming Language' set to 'Python', with a green box below it saying 'You selected: Python'. The title 'Simple Counter' is displayed, along with a 'Increase Counter' button and a 'Counter Value: 3' label. Below this, the title 'Display Sample DataFrame' is shown, followed by a table with three rows of data. The table has columns for index ('0', '1', '2'), name ('Alice', 'Bob', 'Charlie'), and salary ('50000', '60000', '70000'). In the top right corner, there are 'Deploy' and 'Edit' buttons.

Courses Menu

Select Course

Full Stack Python

You selected Full Stack Python

Upload CSV File

Upload a CSV file

Drag and drop file here
Limit 200MB per file + CSV

Browse files

Employee_Salary_Dataset.csv 0.0KB

Uploaded Data:

ID	Experience_Years	Age	Gender	Salary
0	1	35	Female	250000
1	2	31	Male	50000
2	3	33	Female	170000
3	4	32	Male	25000
4	5	37	Male	100000
5	6	35	Male	500000

Courses Menu

Select Course

Full Stack Python

You selected Full Stack Python

Display Image



2. Project Title : Role-Based Employee Salary Filter App

i. Tools / Libraries Used

- **Python** – Core programming language
- **Streamlit** – For building interactive web applications
- **Pandas** – For reading, filtering, and displaying CSV data

ii. Project Description

The **Role-Based Employee Salary Filter App** is an interactive Streamlit web application that allows users to:

1. **Select their role** (HR, Manager, Employee) from the sidebar.
2. View different sidebar messages based on their selected role.
3. Upload an employee CSV file.
4. Automatically filter and display employees whose salary is greater than ₹50,000.
5. Display the total number of employees matching the filter criteria.

Key Features:

- Role-based dynamic content display
- CSV file upload functionality
- Salary-based filtering using Pandas
- Conditional column validation
- Clean and interactive UI

This project demonstrates:

- Conditional rendering
- Data filtering
- File handling
- Role-based access logic
- Real-time data visualization

iii. Code Implementation:

```
import streamlit as st
```

```
import pandas as pd
```

```
st.title("Employee Salary Filter App")

# ROLE SELECTION
st.sidebar.title("User Role")
role = st.sidebar.selectbox(
    "Select Your Role",
    ["HR", "Manager", "Employee"]
)

# Display content based on role
if role == "HR":
    st.sidebar.success("Welcome HR ")
elif role == "Manager":
    st.sidebar.info("Manager Dashboard ")
elif role == "Employee":
    st.sidebar.warning("Employee View ")

st.write(f"## Logged in as: {role}")

# FILE UPLOAD
file = st.file_uploader("Upload Employee CSV File", type=["csv"])

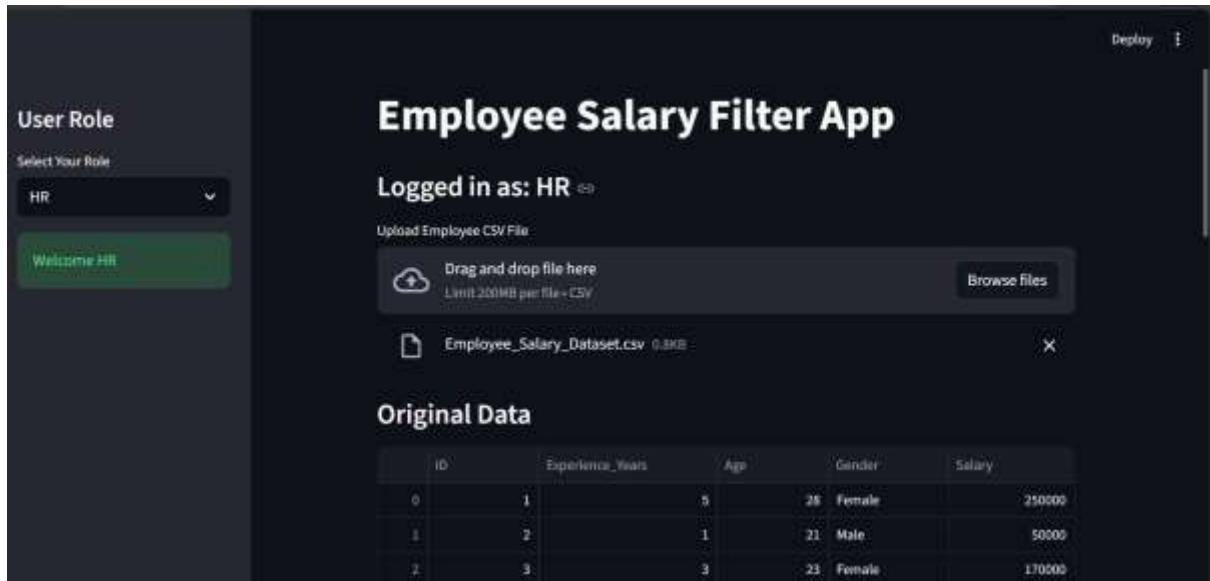
if file is not None:
    df = pd.read_csv(file)

    st.subheader("Original Data")
    st.dataframe(df)

# SALARY FILTER
if "Salary" in df.columns:
    filtered_df = df[df["Salary"] > 50000]
```

```
st.subheader("Employees with Salary > 50,000")  
st.dataframe(filtered_df)  
  
st.success(f'{len(filtered_df)} employees found with salary > 50,000')  
else:  
    st.error("Salary column not found in CSV file.")
```

iv. Output:



The screenshot shows a user interface for an "Employee Salary Filter App". On the left, there's a sidebar titled "User Role" with a dropdown menu set to "HR" and a green button labeled "Welcome HR". The main area has a title "Employee Salary Filter App" and a header "Logged in as: HR". Below this is a "Upload Employee CSV File" section with a cloud icon, a "Drag and drop file here" input field, and a "Limit 200MB per file - CSV" note. A file named "Employee_Salary_Dataset.csv" (0.9KB) is listed with a delete "X" button. At the bottom, there's a section titled "Original Data" showing a table with columns: ID, Experience_Years, Age, Gender, and Salary. The data rows are:

ID	Experience_Years	Age	Gender	Salary
0	1	5	Female	250000
1	2	1	Male	50000
2	3	3	Female	170000

localhost:8081

User Role

Select Your Role

HR

Welcome HR

Original Data

ID	Experience_Years	Age	Gender	Salary
0	1	5	28 Female	250000
1	2	1	21 Male	50000
2	3	3	23 Female	170000
3	4	2	22 Male	25000
4	5	1	17 Male	10000
5	6	25	62 Male	5001000
6	7	19	54 Female	800000
7	8	2	21 Female	9000
8	9	10	36 Female	61500
9	10	15	54 Female	650000

localhost:8081

User Role

Select Your Role

HR

Welcome HR

Employees with Salary > 50,000

ID	Experience_Years	Age	Gender	Salary
0	3	5	28 Female	250000
2	3	3	23 Female	170000
5	6	25	62 Male	5001000
6	7	19	54 Female	800000
8	9	10	36 Female	61500
9	10	15	54 Female	650000
10	11	4	26 Female	250000
11	12	6	29 Male	1400000
12	13	14	39 Male	6000050
13	14	11	40 Male	220100

23 employees found with salary > 50,000