# loading the saved model

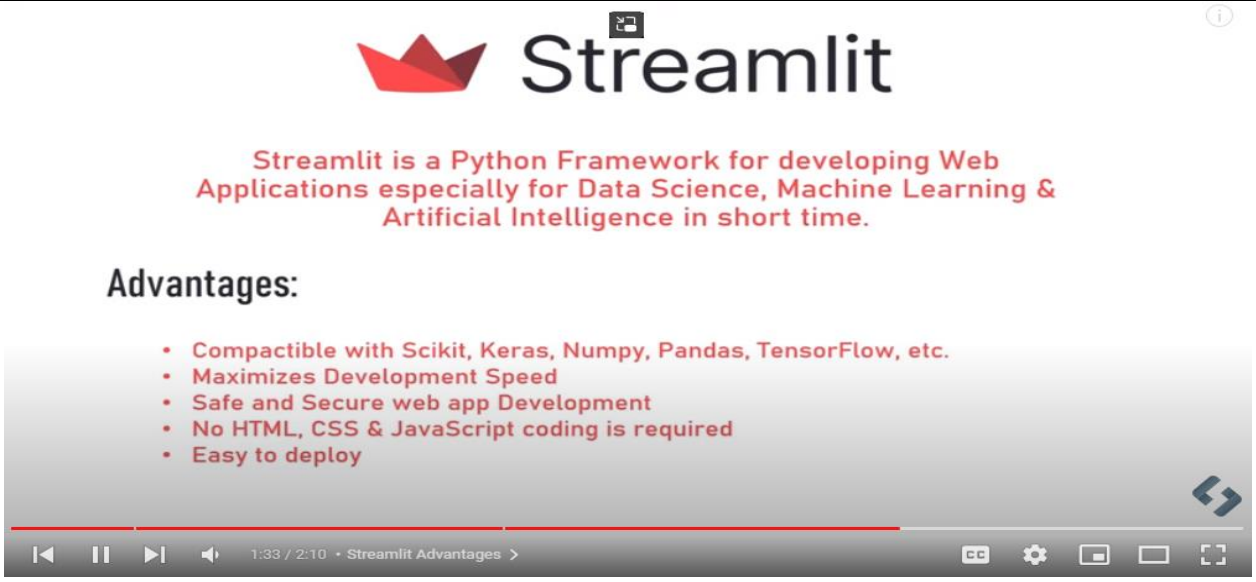
```
loaded_model = pickle.load(open('C:/Users/Afroz/Desktop/Sidharthan program/Complete-
Machine-Learning-Course-Part-5-main/5.1. Deploy ML Model using
Streamlit/trained_model.sav', 'rb'))
```

Image upload

```
# Display Images
from PIL import Image
img = Image.open("C:/Users/Afroz/Desktop/Streamlit Test/giraffe.png")
st.image(img, width=200)
```

Video upload

```
# Video Upload
video_file = open('C:/Users/Afroz/Desktop/Streamlit Test/streamlit.mp4', 'rb')
video_bytes = video_file.read()
st.video(video_bytes)
```



Streamlit

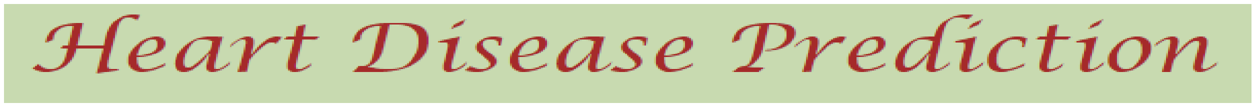
Streamlit is a Python Framework for developing Web Applications especially for Data Science, Machine Learning & Artificial Intelligence in short time.

Advantages:

- Compactible with Scikit, Keras, Numpy, Pandas, TensorFlow, etc.
- Maximizes Development Speed
- Safe and Secure web app Development
- No HTML, CSS & JavaScript coding is required
- Easy to deploy

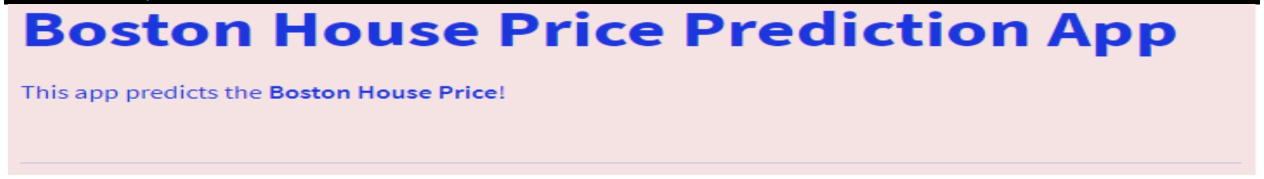
# Color Test

```
title = '<p style="font-family:Lucida Calligraphy; color:Brown; font-size:
45px;">Heart Disease Prediction</p>'
st.markdown(title, unsafe_allow_html=True)
```


Heart Disease Prediction

# st.write

```
st.write("""
# Boston House Price Prediction App

This app predicts the **Boston House Price**!
""")
st.write('---')
```


Boston House Price Prediction App

This app predicts the **Boston House Price**!

# st.title

```
st.title("My final revision Afroz")
```

My final revision Afroz

st.header

```
st.header("How are you friend")
```

How are you friend

st.subheader

```
st.subheader("I am Afroz")
```

I am Afroz

st.caption

```
st.caption('This is a string that explains something above.')
```

This is a string that explains something above.

st.text

```
st.text("Hello i am Afroz")
```

```
Hello i am Afroz
```

st.markdown

```
st.markdown("### This is a markdown")
```

This is a markdown

st.markdown

```
st.markdown("# ***This is a markdown***" )
```

This is a markdown

st.markdown

```
st.markdown("## ***This is a markdown***" )
```

This is a markdown

st.markdown

```
st.markdown("### ***This is a markdown***" )
```

This is a markdown

st.success

```
st.success("Success")
```

Success

st.info

```
st.info("Information")
```

Information

st.warning

```
st.warning("Warning")
```

Warning

st.error("Error")

```
st.error("Error")
```

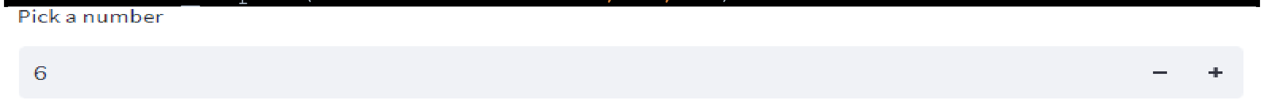
Error

# st.code

```
x=2021
```

# st.number_input

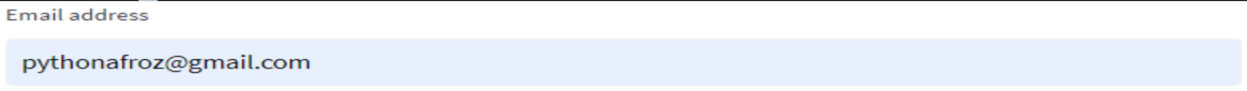
```
st.number_input('Pick a number', 0,10)
```


Pick a number

6

# st.text_input

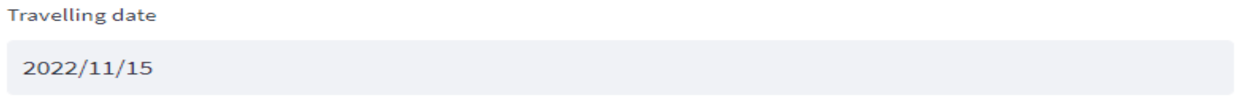
```
st.text_input('Email address')
```


Email address

pythonafroz@gmail.com

# st.date_input

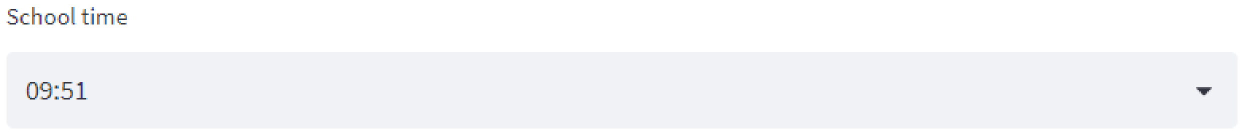
```
st.date_input('Travelling date')
```


Travelling date

2022/11/15

# st.time_input

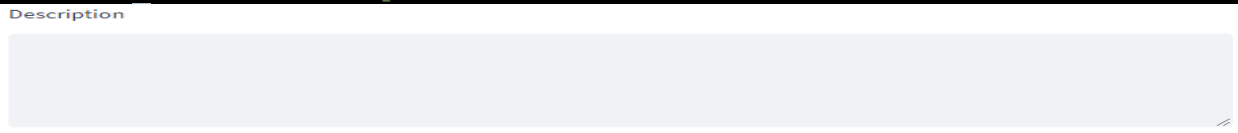
```
st.time_input('School time')
```


School time

09:51

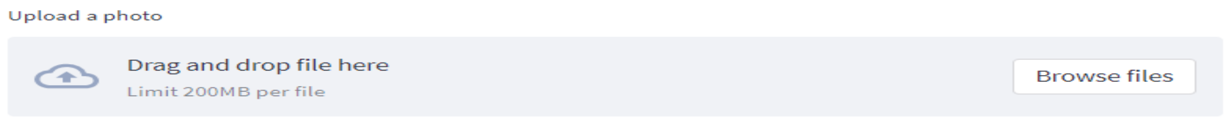
# st.text_area

```
st.text_area('Description')
```


Description

# st.file_upload

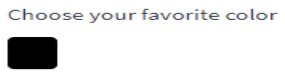
```
st.file_uploader('Upload a photo')
```


Upload a photo

Drag and drop file here
Limit 200MB per file

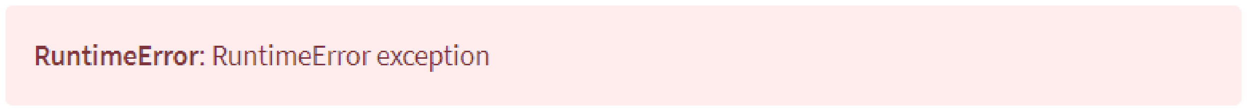
# st.color_picker

```
st.file_uploader('Uploa
st.color_picker('Choose your favorite color')
```


Choose your favorite color

# st.exception

```
st.exception(RuntimeError("RuntimeError exception"))
```


RuntimeError: RuntimeError exception

st.write

```
st.write("Text with write")
```

Text with write

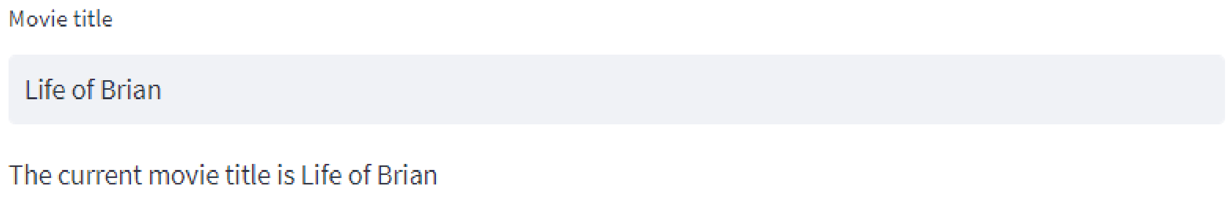
st.write

```
st.write(range(10))
```

```
range(0, 10)
```

# st.text input

```
title = st.text_input('Movie title', 'Life of Brian')
st.write('The current movie title is', title)
```



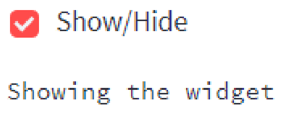
Movie title

Life of Brian

The current movie title is Life of Brian

# st.checkbox

```
if st.checkbox("Show/Hide"):
   # display the text if the checkbox returns True value
   st.text("Showing the widget")
```

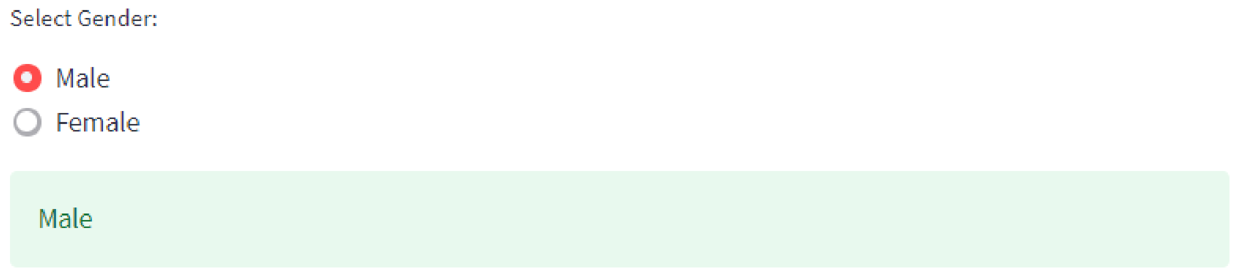


✓ Show/Hide

```
Showing the widget
```

# st.radio

```
status = st.radio("Select Gender: ", ('Male', 'Female'))
if (status == 'Male'):
   st.success("Male")
else:
   st.success("Female")
```



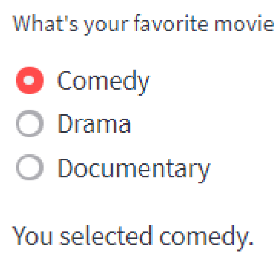
Select Gender:

● Male
○ Female

Male

# st.radio

```python
genre = st.radio(
    "What's your favorite movie genre",
    ('Comedy', 'Drama', 'Documentary'))

if genre == 'Comedy':
    st.write('You selected comedy.')
else:
    st.write("You didn't select comedy.")
```



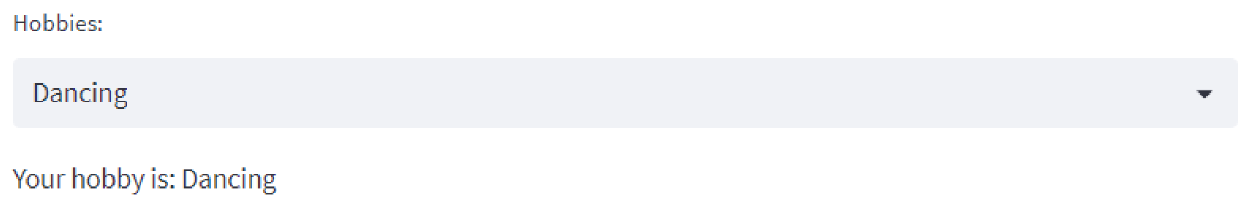
What's your favorite movie genre

● Comedy
○ Drama
○ Documentary

You selected comedy.

# st.selectbox

```python
hobby = st.selectbox("Hobbies: ",['Dancing', 'Reading', 'Sports'])
st.write("Your hobby is: ", hobby)
```



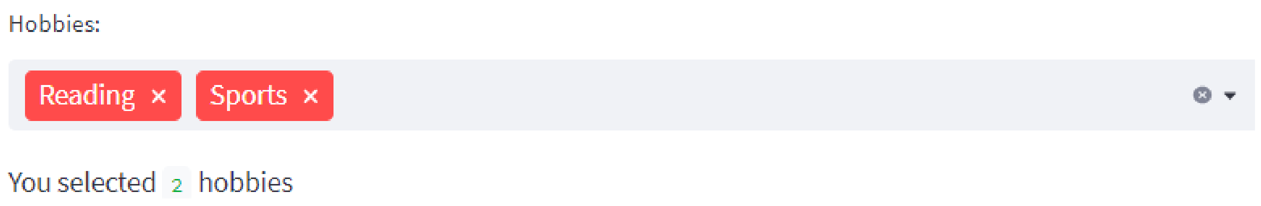
Hobbies:

Dancing

Your hobby is: Dancing

# st.multiselect

```python
# Multi select box
hobbies = st.multiselect("Hobbies: ",['Dancing', 'Reading', 'Sports'])
st.write("You selected", len(hobbies), 'hobbies')
```



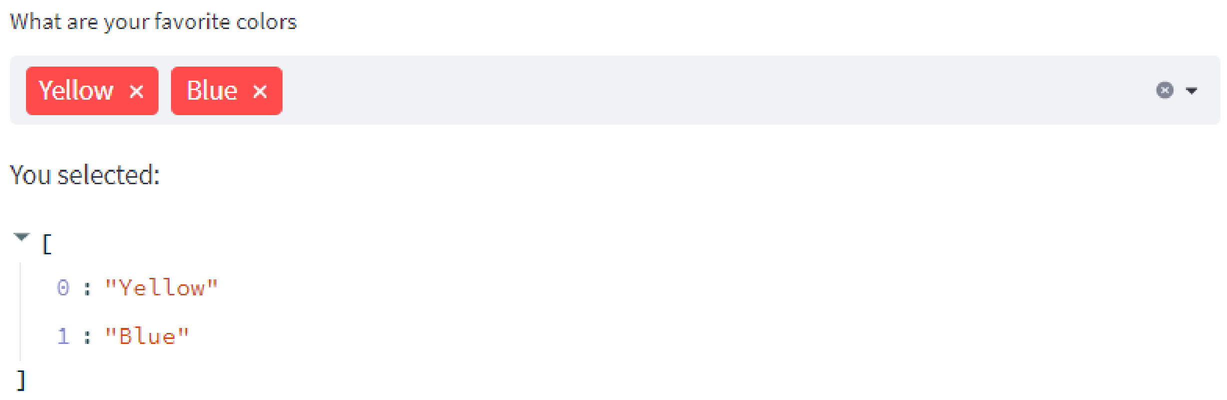
Hobbies:

Reading × Sports ×

You selected 2 hobbies

# st.multiselect

```python
# st.multiselect
options = st.multiselect(
    'What are your favorite colors',
    ['Green', 'Yellow', 'Red', 'Blue'],
    ['Yellow', 'Red'])

st.write('You selected:', options)
```

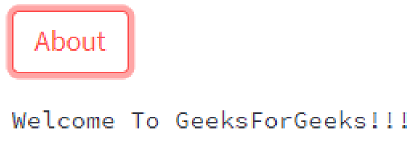

What are your favorite colors

You selected:

```
[
  0 : "Yellow"
  1 : "Blue"
]
```

# st.button

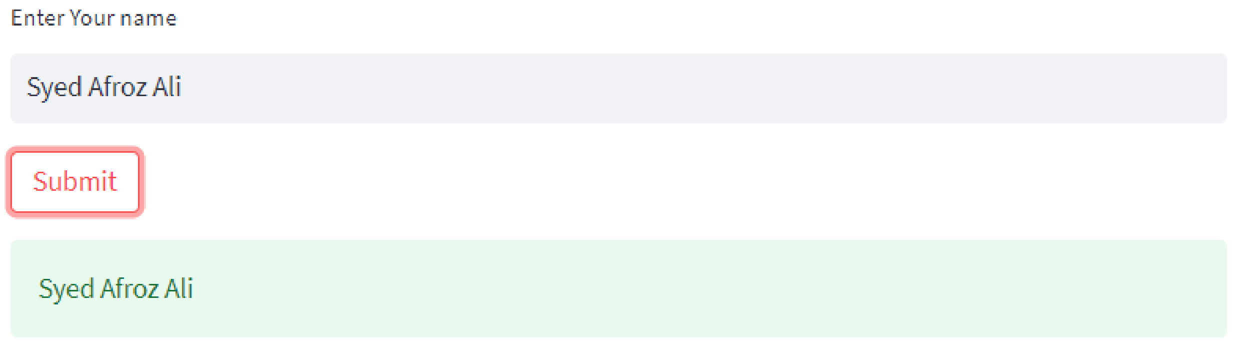
```
# Create a button, that when clicked, shows a text
if(st.button("About")):
    st.text("Welcome To GeeksForGeeks!!!")
```



```
Welcome To GeeksForGeeks!!!
```

# st.text_input

```
name = st.text_input("Enter Your name", "Type Here ...")

if(st.button('Submit')):
   result = name.title()
   st.success(result)
```

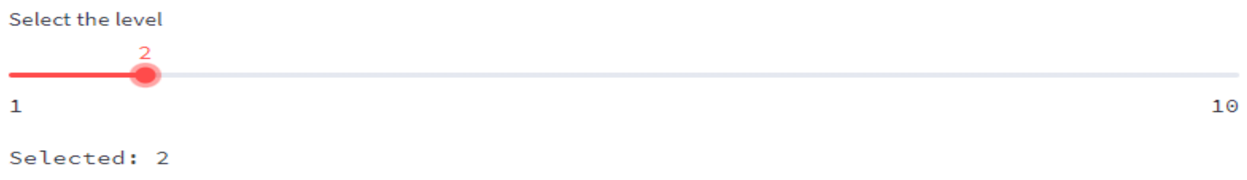

Enter Your name

Syed Afroz Ali

Syed Afroz Ali

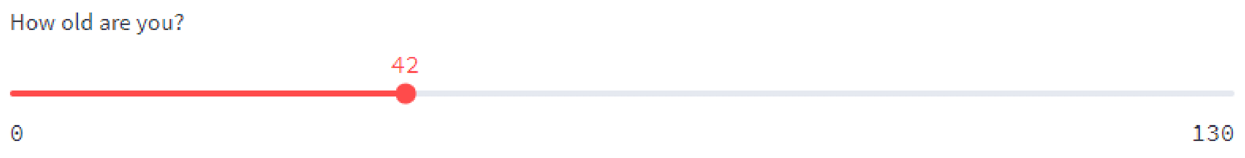
# st.slider

```
level = st.slider("Select the level", 1, 10,)
st.text('Selected: {}'.format(level))
```


Select the level

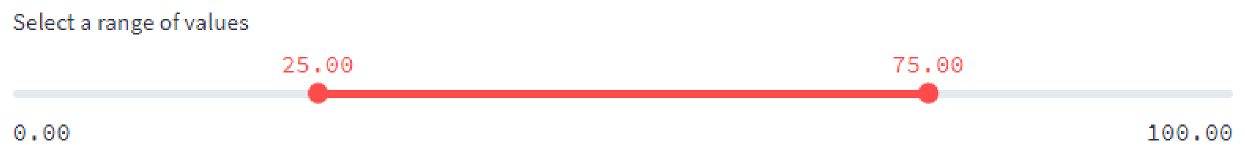
```
Selected: 2
```

```
age = st.slider('How old are you?', 0, 130, 25)
st.write("I'm ", age, 'years old')
```



I'm `42` years old

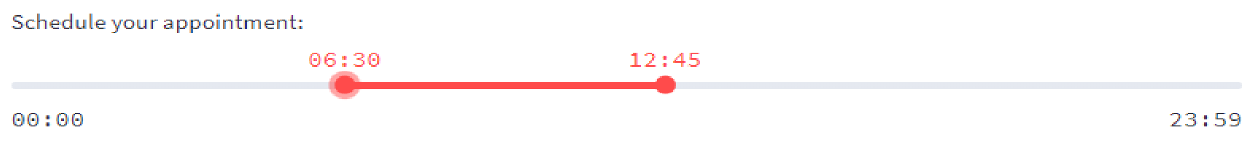
```
values = st.slider(
    'Select a range of values',
    0.0, 100.0, (25.0, 75.0))
st.write('Values:', values)
```



Values: `(25.0, 75.0)`

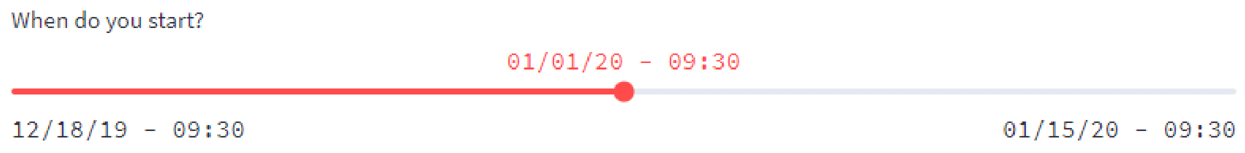
```
from datetime import time
appointment = st.slider(
    "Schedule your appointment:",
    value=(time(11, 30), time(12, 45)))
st.write("You're scheduled for:", appointment)
```

You selected wavelengths between red and blue



You're scheduled for: `(datetime.time(6, 30), datetime.time(12, 45))`

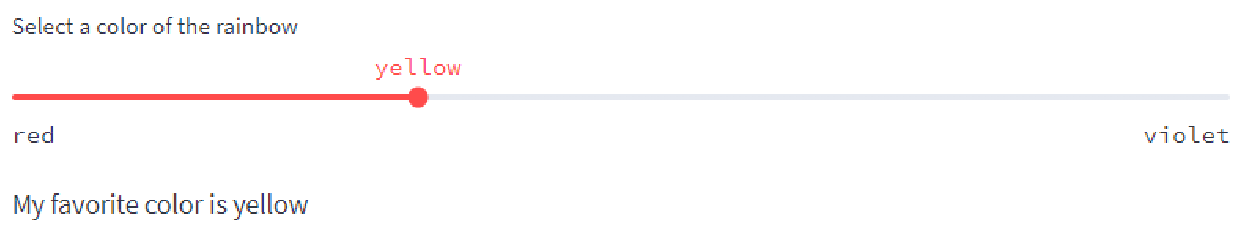
```
from datetime import datetime
start_time = st.slider(
    "When do you start?",
    value=datetime(2020, 1, 1, 9, 30),
    format="MM/DD/YY - hh:mm")
st.write("Start time:", start_time)
```



Start time: `2020-01-01 09:30:00`

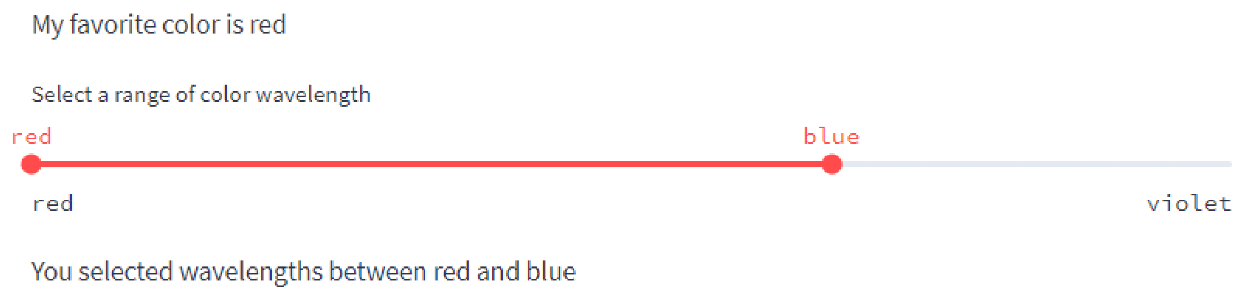
# st.select_slider

```
color = st.select_slider(
    'Select a color of the rainbow',
    options=['red', 'orange', 'yellow', 'green', 'blue', 'indigo', 'violet'])
st.write('My favorite color is', color)
```


Select a color of the rainbow

My favorite color is yellow

```
start_color, end_color = st.select_slider(
    'Select a range of color wavelength',
    options=['red', 'orange', 'yellow', 'green', 'blue', 'indigo', 'violet'],
    value=('red', 'blue'))
st.write('You selected wavelengths between', start_color, 'and', end_color)
```

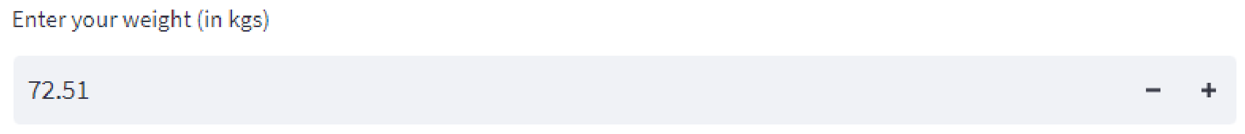

My favorite color is red

Select a range of color wavelength

You selected wavelengths between red and blue

# st.number_input

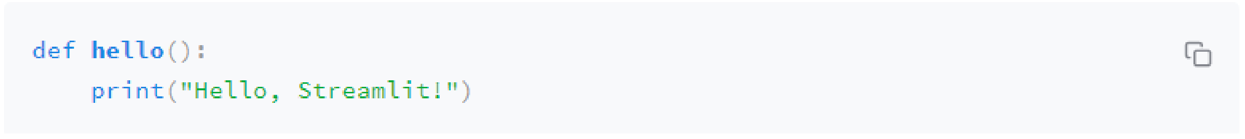
```
# st.number_input
weight = st.number_input("Enter your weight (in kgs)")
```


Enter your weight (in kgs)

72.51

# Code display

```
code = '''def hello():
    print("Hello, Streamlit!")'''
st.code(code, language='python')
```


```
def hello():
    print("Hello, Streamlit!")
```

# st.latex

```
# Formula Display
st.text('This is some text.')
st.latex(r'''
    a + ar + a r^2 + a r^3 + \cdots + a r^{n-1} =
    \sum_{k=0}^{n-1} ar^k =
    a \left(\frac{1-r^{n}}{1-r}\right)
    ''')
```

```
This is some text.
```

$$a + ar + a r^2 + a r^3 + \cdots + a r^{n-1} = \sum_{k=0}^{n-1} ar^k = a \left(\frac{1-r^{n}}{1-r}\right)$$

```
st.latex(r''' a+a r^1+a r^2+a r^3 ''')
```

$$a+a r^1+a r^2+a r^3$$

# st.metric

```
# Color Display
st.metric(label="Temperature", value="70 °F", delta="1.2 °F")
```

Temperature

70 °F

↑ 1.2 °F

```
col1, col2, col3 = st.columns(3)
col1.metric("Temperature", "70 °F", "1.2 °F")
col2.metric("Wind", "9 mph", "-8%")
col3.metric("Humidity", "86%", "4%")
```

Temperature	Wind	Humidity
70 °F	9 mph	86%
↑ 1.2 °F	↓ -8%	↑ 4%

```
st.metric(label="Gas price", value=4, delta=-0.5,
    delta_color="inverse")
```

Gas price

4

↓ -0.5

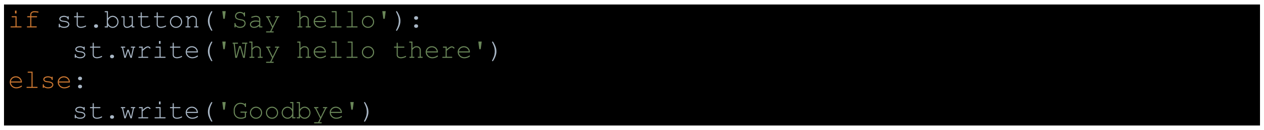
```
st.metric(label="Active developers", value=123, delta=123,
    delta_color="off")
```

Active developers

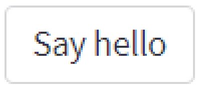
123

↑ 123

# st.button

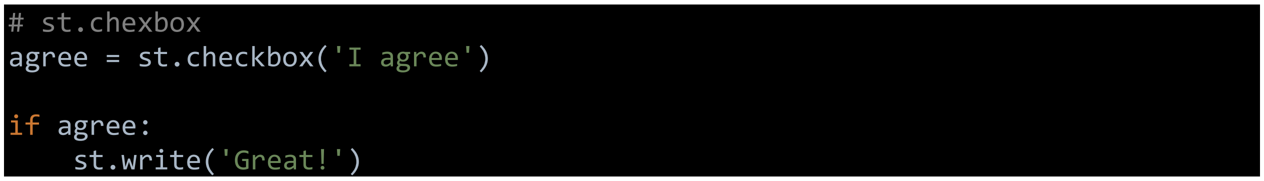


```python
if st.button('Say hello'):
    st.write('Why hello there')
else:
    st.write('Goodbye')
```



Goodbye

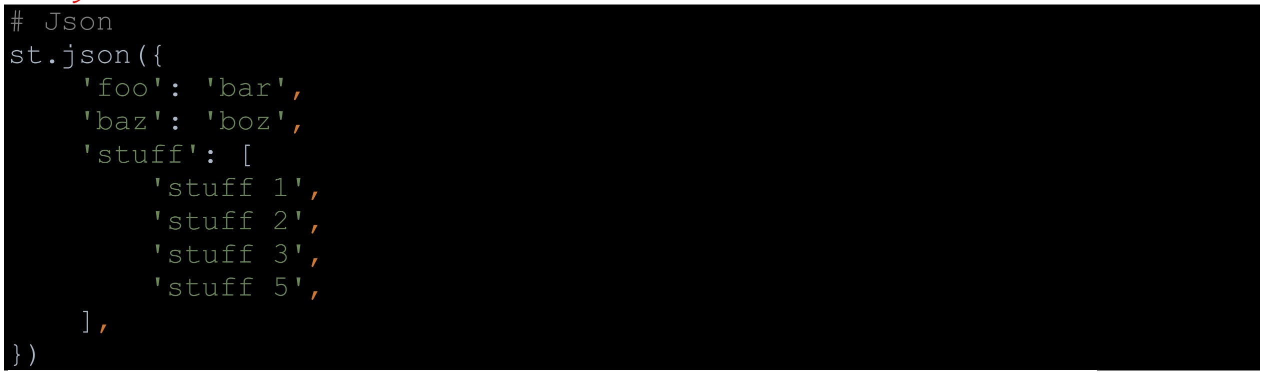
# st.checkbox



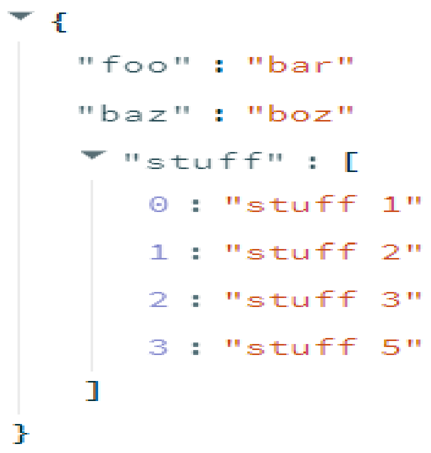
```python
# st.chexbox
agree = st.checkbox('I agree')

if agree:
    st.write('Great!')
```

Great!

# st.jason



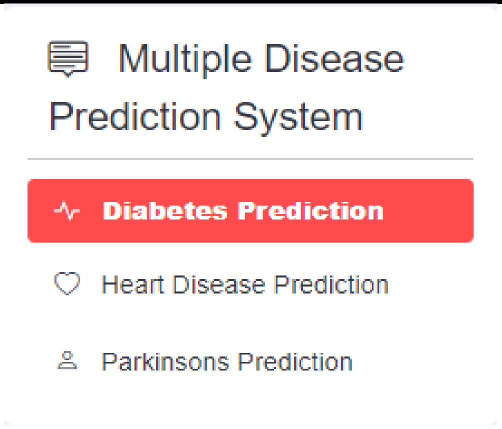
```python
# Json
st.json({
    'foo': 'bar',
    'baz': 'boz',
    'stuff': [
        'stuff 1',
        'stuff 2',
        'stuff 3',
        'stuff 5',
    ],
})
```



```
{
   "foo" : "bar"
   "baz" : "boz"
   "stuff" : [
      0 : "stuff 1"
      1 : "stuff 2"
      2 : "stuff 3"
      3 : "stuff 5"
   ]
}
```

# st.sidebar

```python
# sidebar for navigation
from streamlit_option_menu import option_menu
with st.sidebar:
    
    selected = option_menu('Multiple Disease Prediction System',
                          
                          ['Diabetes Prediction',
                           'Heart Disease Prediction',
                           'Parkinsons Prediction'],
                          icons=['activity', 'heart', 'person'],
                          default_index=0)


# Diabetes Prediction Page
if (selected == 'Diabetes Prediction'):
    # page title
    st.title('Diabetes Prediction using ML')

# Heart Disease Prediction Page
if (selected == 'Heart Disease Prediction'):
    # page title
    st.title('Heart Disease Prediction using ML')

# Parkinson's Prediction Page
if (selected == "Parkinsons Prediction"):
    # page title
    st.title("Parkinson's Disease Prediction using ML")
```

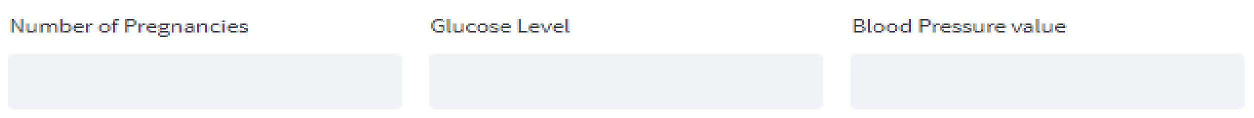


Icons:

https://icons.getbootstrap.com/

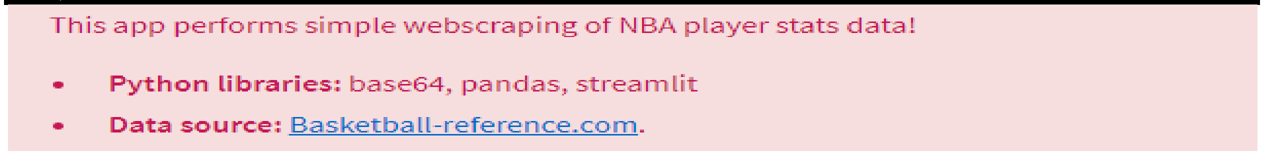
# Multiple Column

```python
# getting the input data from the user
col1, col2, col3 = st.columns(3)

with col1:
    Pregnancies = st.text_input('Number of Pregnancies')

with col2:
    Glucose = st.text_input('Glucose Level')

with col3:
    BloodPressure = st.text_input('Blood Pressure value')
```



# St.markdown

```python
st.markdown("""
This app performs simple webscraping of NBA player stats data!
* **Python libraries:** base64, pandas, streamlit
* **Data source:** [Basketball-reference.com](https://www.basketball-reference.com/).
""")
```

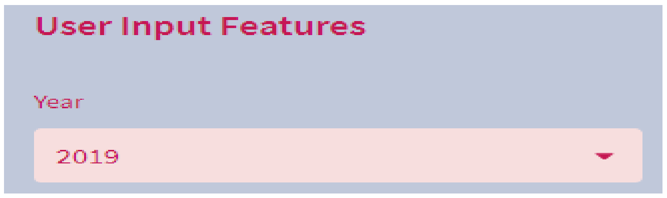


This app performs simple webscraping of NBA player stats data!

- **Python libraries:** base64, pandas, streamlit
- **Data source:** Basketball-reference.com.

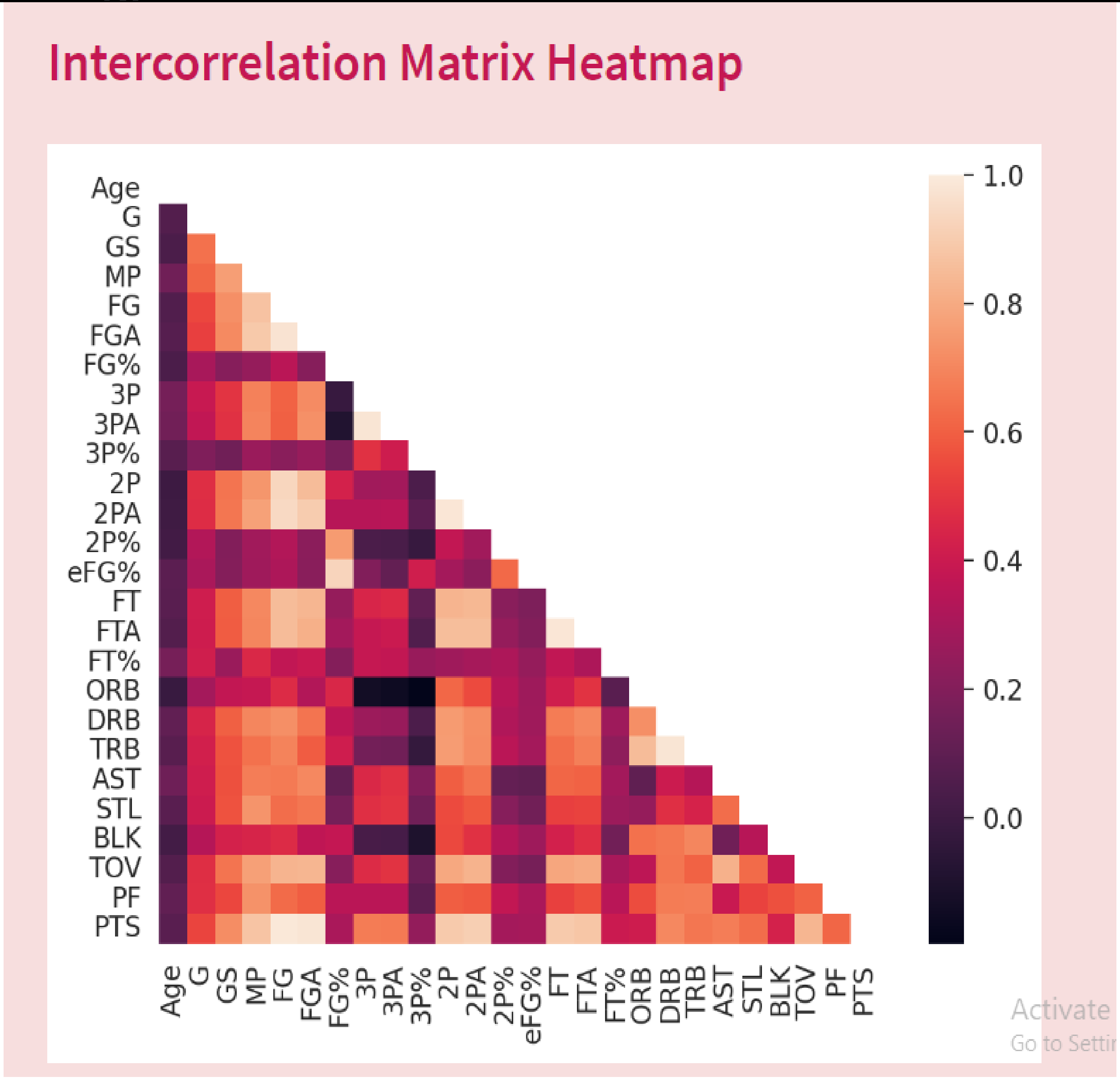
# st.sidebar.header

```python
st.sidebar.header('User Input Features')
selected_year = st.sidebar.selectbox('Year',
list(reversed(range(1950,2020))))
```



# Heatmap

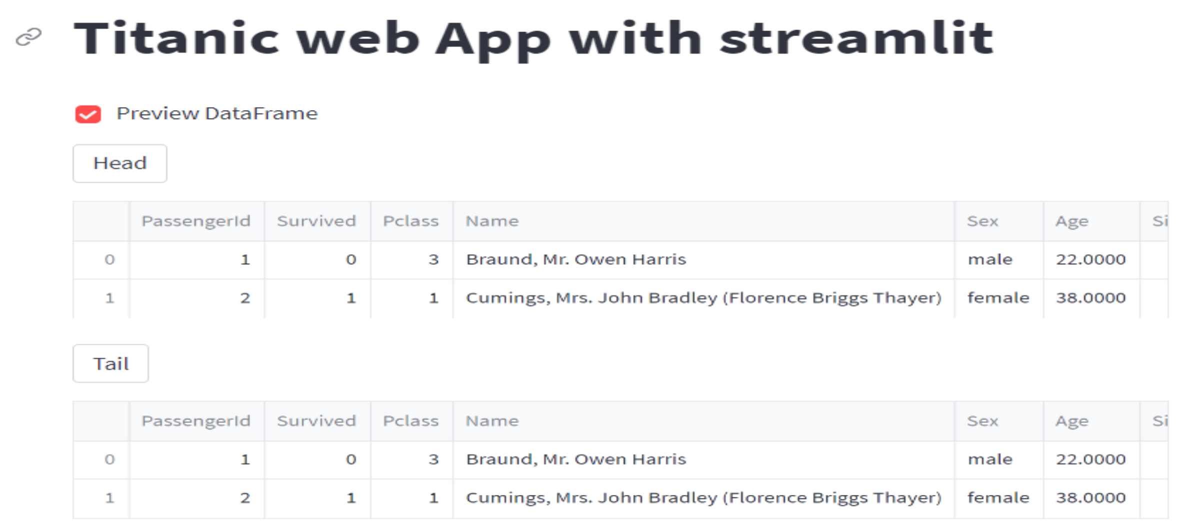
```
if st.button('Intercorrelation Heatmap'):
    st.header('Intercorrelation Matrix Heatmap')
    df_selected_team.to_csv('output.csv',index=False)
    df = pd.read_csv('output.csv')

    corr = df.corr()
    mask = np.zeros_like(corr)
    mask[np.triu_indices_from(mask)] = True
    with sns.axes_style("white"):
        f, ax = plt.subplots(figsize=(7, 5))
        ax = sns.heatmap(corr, mask=mask, vmax=1, square=True)
    st.pyplot()
```



Intercorrelation Matrix Heatmap

# Data Loading

```
# Plotting Pkgs
import matplotlib.pyplot as plt
import seaborn as sns
from PIL import Image, ImageFilter, ImageEnhance

st.set_option('deprecation.showPyplotGlobalUse', False)

def main():
    """Simple EDA App"""
    st.title("Titanic web App with streamlit")

    # Our Dataset
    my_dataset = "titanic.csv"
    # To Improve speed and cache data
    @st.cache(persist=True)
    def explore_data(dataset):
        df = pd.read_csv(os.path.join(dataset))
        return df

    # Load Our Dataset

    data = explore_data(my_dataset)
    # Show Dataset
    if st.checkbox("Preview DataFrame"):
        if st.button("Head"):
            st.write(data.head(2))
        if st.button("Tail"):
            st.write(data.tail(2))
        else:
            st.write(data.head(2))

if __name__ == '__main__':
    main()
```



Titanic web App with streamlit

✓ Preview DataFrame

Head

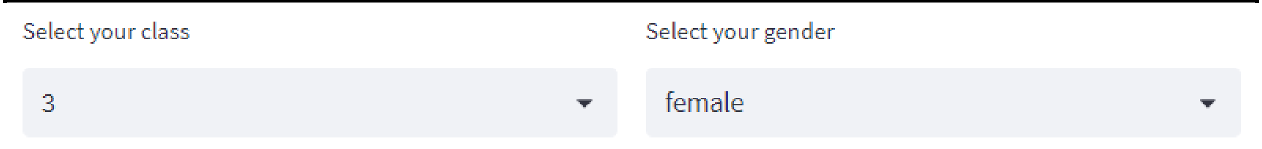
	PassengerId	Survived	Pclass	Name	Sex	Age	Si
0	1	0	3	Braund, Mr. Owen Harris	male	22.0000	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38.0000	

Tail

	PassengerId	Survived	Pclass	Name	Sex	Age	Si
0	1	0	3	Braund, Mr. Owen Harris	male	22.0000	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38.0000	

# Data Input Column

```python
import streamlit as st
import pandas as pd
import os

# components
def main():
        # selecting pclass
    col1, col2 = st.columns(2)

    with col1:
        pclass = st.selectbox("Select your class", (1, 2, 3))
        # msg = 'You selected the {} class'
        # if pclass == 1:
        #    st.write(msg.format('First'))
        # elif pclass == 2:
        #    st.write(msg.format('middle'))
        # else:
        #    st.write(msg.format('Third'))

    # Select your sex and title:
    with col2:
        sex = st.selectbox("Select your gender", ('male', 'female'))
        if sex == 'female':
            title = 'Miss'
            # st.write('You are a Miss')
        else:
            title = 'Mr'
            # st.write('You are a Mr')


if __name__ == '__main__':
    main()
```


Select your class

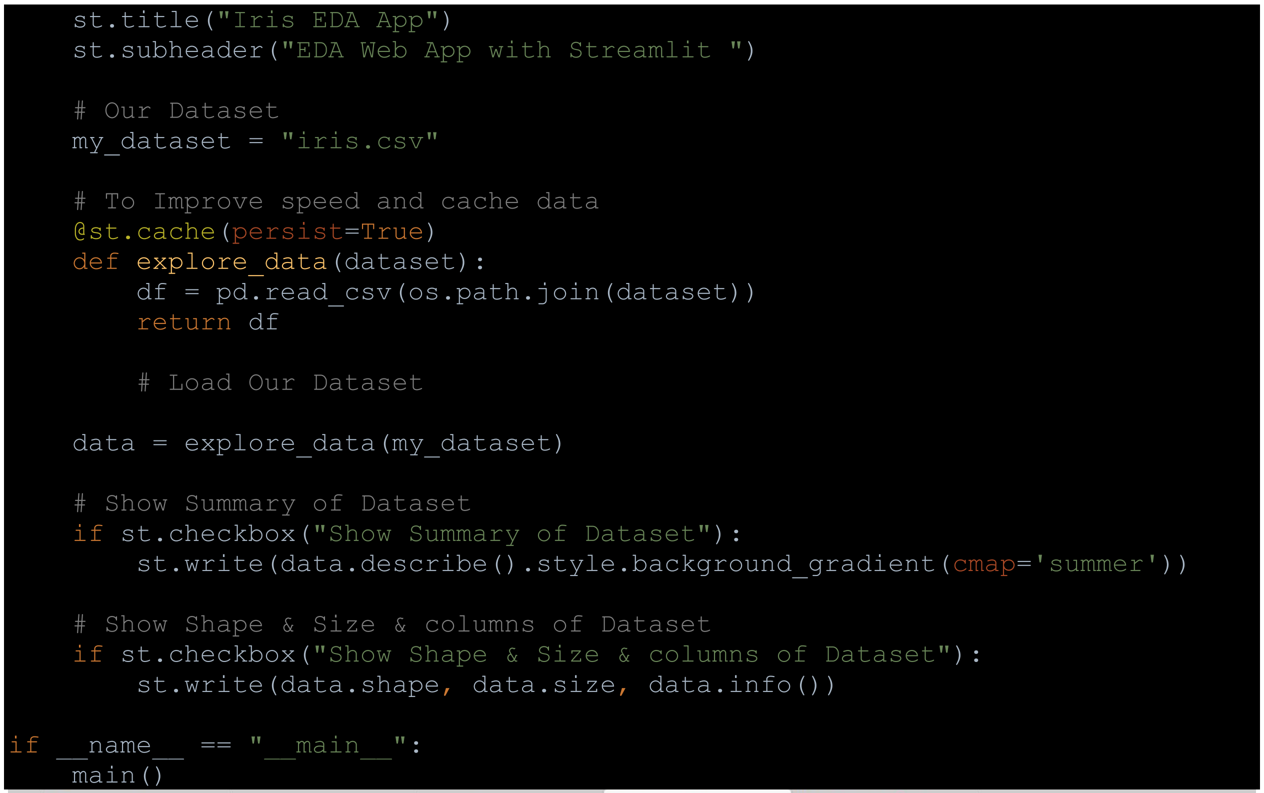
3

Select your gender

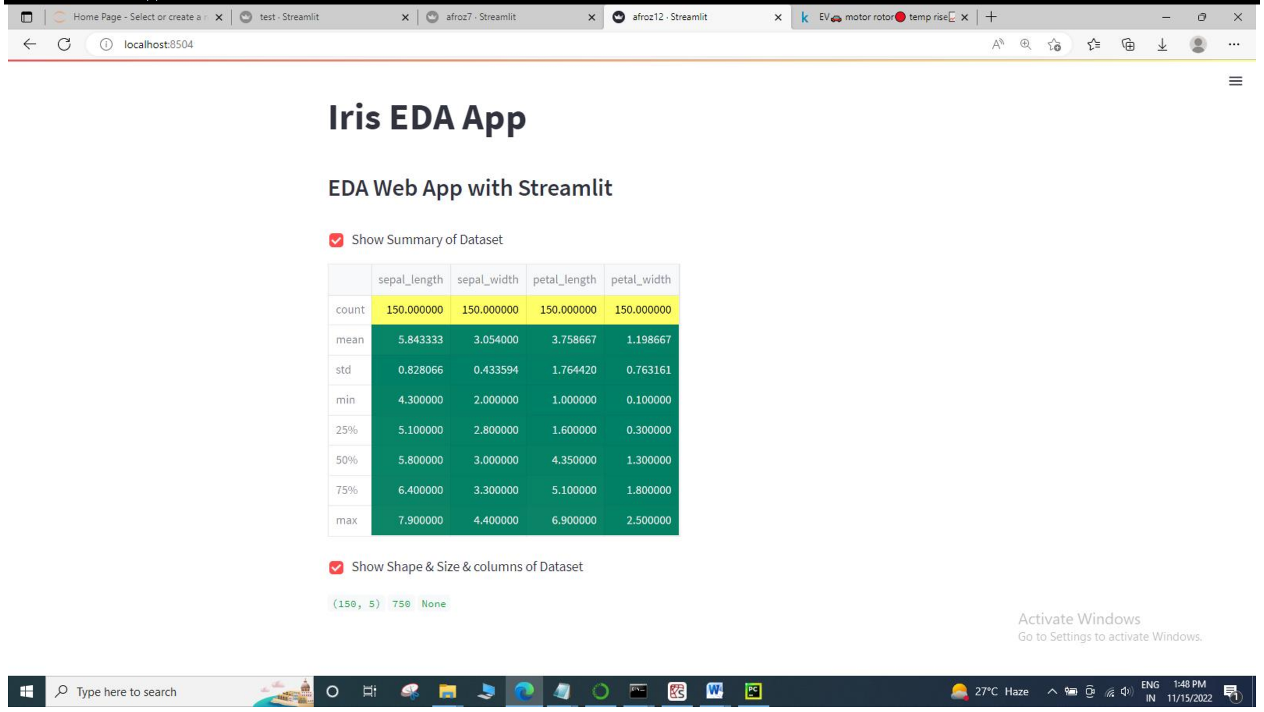
female

# Data Description

```python
import streamlit as st

# EDA Pkgs
import pandas as pd
import numpy as np
import os

# Plotting Pkgs
import matplotlib.pyplot as plt
import seaborn as sns
from PIL import Image, ImageFilter, ImageEnhance
def main():
```



```
    st.title("Iris EDA App")
    st.subheader("EDA Web App with Streamlit ")

    # Our Dataset
    my_dataset = "iris.csv"

    # To Improve speed and cache data
    @st.cache(persist=True)
    def explore_data(dataset):
        df = pd.read_csv(os.path.join(dataset))
        return df

        # Load Our Dataset

    data = explore_data(my_dataset)

    # Show Summary of Dataset
    if st.checkbox("Show Summary of Dataset"):
        st.write(data.describe().style.background_gradient(cmap='summer'))

    # Show Shape & Size & columns of Dataset
    if st.checkbox("Show Shape & Size & columns of Dataset"):
        st.write(data.shape, data.size, data.info())

if __name__ == "__main__":
    main()
```



Iris EDA App

EDA Web App with Streamlit

☑ Show Summary of Dataset

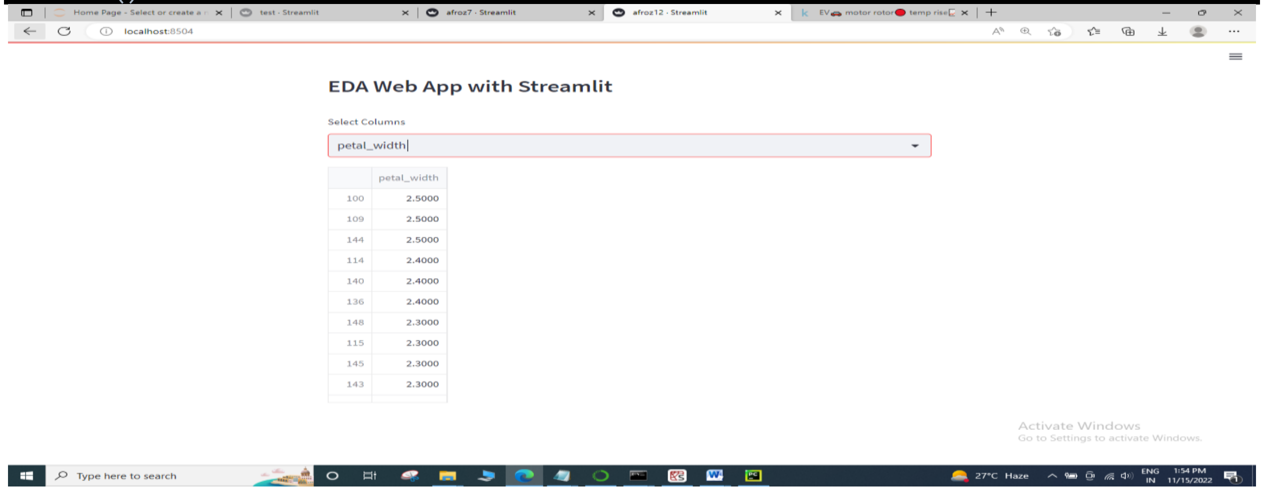
	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

☑ Show Shape & Size & columns of Dataset

```
(150, 5)  750  None
```

# Data Summary

```python
import streamlit as st
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
import seaborn as sns
from PIL import Image, ImageFilter, ImageEnhance

def main():
    st.title("Iris EDA App")
    st.subheader("EDA Web App with Streamlit ")

    # Our Dataset
    my_dataset = "iris.csv"
    # To Improve speed and cache data
    @st.cache(persist=True)
    def explore_data(dataset):
        df = pd.read_csv(os.path.join(dataset))
        return df

    # Load Our Dataset
    data = explore_data(my_dataset)
    # Selection of Columns
    species_option = st.selectbox('Select Columns',
                                  ('sepal_length', 'sepal_width', 'petal_length',
'petal_width', 'species'))
    if species_option == 'sepal_length':
        st.write(data['sepal_length'].sort_values(ascending=False))
    elif species_option == 'sepal_width':
        st.write(data['sepal_width'].sort_values(ascending=False))
    elif species_option == 'petal_length':
        st.write(data['petal_length'].sort_values(ascending=False))
    elif species_option == 'petal_width':
        st.write(data['petal_width'].sort_values(ascending=False))
    elif species_option == 'species':
        st.write(data['species'].value_counts(ascending=False))
    else:
        st.write("Select A Column")

if __name__ == "__main__":
    main()
```



EDA Web App with Streamlit

Select Columns

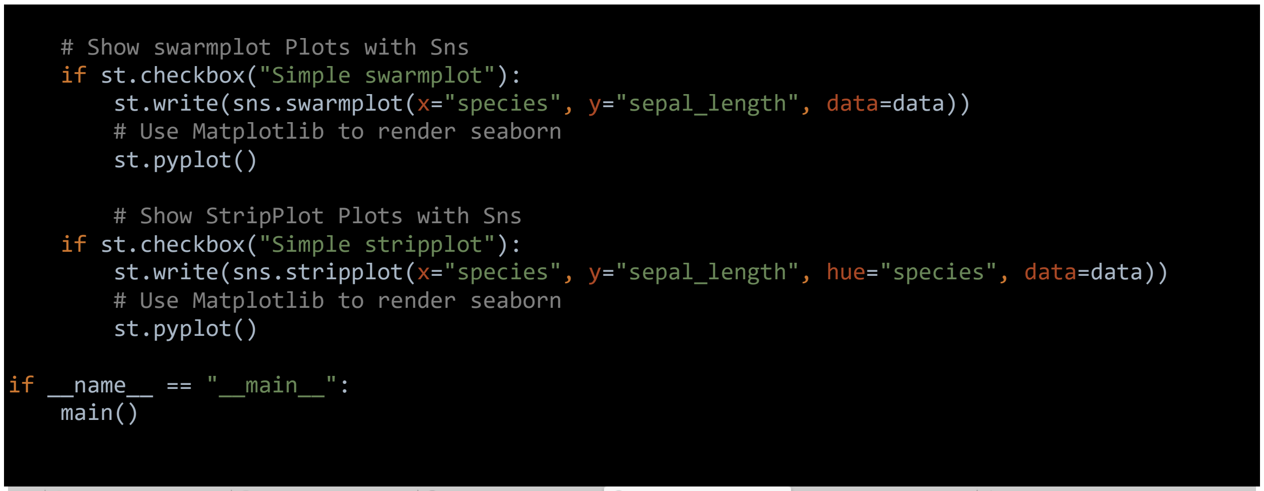
petal_width

	petal_width
100	2.5000
109	2.5000
144	2.5000
114	2.4000
140	2.4000
136	2.4000
148	2.3000
115	2.3000
145	2.3000
143	2.3000

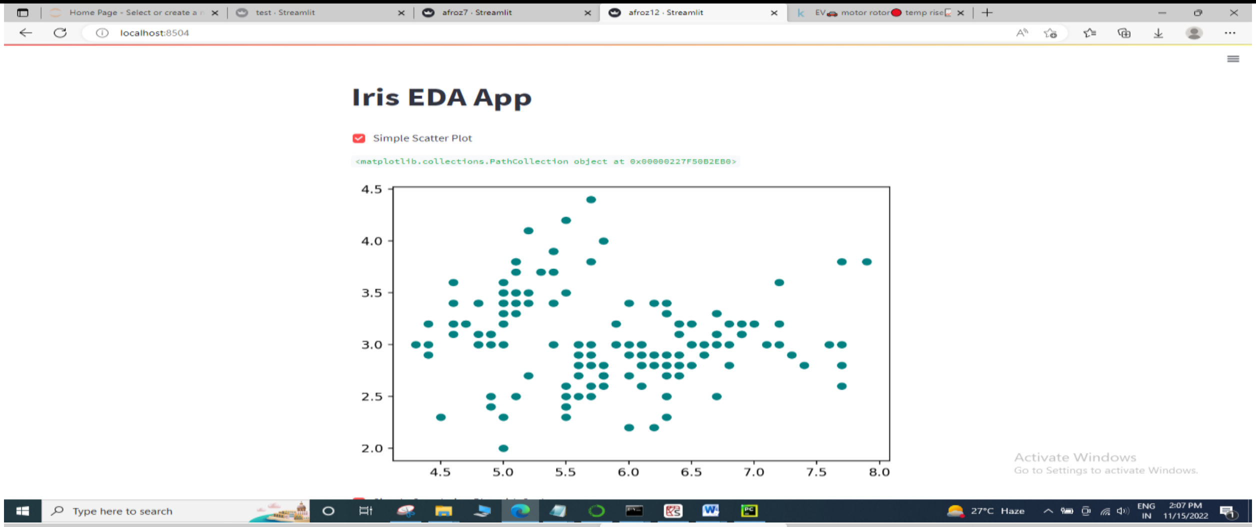
# Plots

```
import streamlit as st
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
import seaborn as sns
from PIL import Image, ImageFilter, ImageEnhance

def main():
    st.title("Iris EDA App")

    # Our Dataset
    my_dataset = "iris.csv"

    # To Improve speed and cache data
    @st.cache(persist=True)
    def explore_data(dataset):
        df = pd.read_csv(os.path.join(dataset))
        return df

    # Load Our Dataset
    data = explore_data(my_dataset)

    # Show histplot Plots with Sns
    if st.checkbox("Simple Scatter Plot"):
        st.write(plt.scatter(data['sepal_length'], data['sepal_width'], color='teal'))
        # Use Matplotlib to render seaborn
        st.pyplot()

    # Show Correlation Plots with Sns
    if st.checkbox("Simple Correlation Plot with Seaborn "):
        st.write(sns.heatmap(data.corr(), annot=True, cmap='winter'))
        # Use Matplotlib to render seaborn
        st.pyplot()

    # Show Pairplot Plots with Sns
    if st.checkbox("Simple PairPlot with Seaborn "):
        st.write(sns.pairplot(data, hue=('species'), corner=True))
        # Use Matplotlib to render seaborn
        st.pyplot()

    # Show CountPlot Plots
    if st.checkbox("Simple CountPlot with Seaborn "):
        st.write(sns.countplot(data['species']))
        # Use Matplotlib to render seaborn
        st.pyplot()

    # Show histplot Plots with Sns
    if st.checkbox("Simple histplot"):
        st.write(sns.histplot(data['sepal_length'], kde=True))
        # Use Matplotlib to render seaborn
        st.pyplot()

    # Show Kde Plots with Sns
    if st.checkbox("Simple Kdeplot"):
        st.write(sns.kdeplot(data=data['sepal_length'], shade=True).set_title("Iris Sepal
Length Kde Plot"))
        # Use Matplotlib to render seaborn
```

```python
        st.pyplot()

    # Show Catplot Plots with Sns
    if st.checkbox("Simple Catplot Plot"):
        st.write(sns.catplot(x="species", y="petal_length", data=data))
        # Use Matplotlib to render seaborn
        st.pyplot()

    # Show Boxplot Plots with Sns
    if st.checkbox("Simple Boxplot Plot"):
        st.write(sns.boxplot(x="species", y="sepal_length", data=data))
        # Use Matplotlib to render seaborn
        st.pyplot()

    # Show Boxenplot Plots with Sns
    if st.checkbox("Simple Boxenplot Plot"):
        st.write(sns.boxenplot(y="sepal_length", x="species", data=data))
        # Use Matplotlib to render seaborn
        st.pyplot()

    # Show Barplot Plots with Sns
    if st.checkbox("Simple Barplot Plot"):
        st.write(sns.barplot(y='sepal_width', x='species', data=data))
        # Use Matplotlib to render seaborn
        st.pyplot()

    # Show lineplot Plots with Sns
    if st.checkbox("Simple lineplot Plot"):
        st.write(sns.lineplot(data=data, color='r'))
        # Use Matplotlib to render seaborn
        st.pyplot()

    # Show displot Plots with Sns
    if st.checkbox("Simple displot Plot"):
        st.write(sns.displot(data=data, x="sepal_length"))
        # Use Matplotlib to render seaborn
        st.pyplot()

    # Show JointGrid Plots with Sns
    if st.checkbox("Simple JointGrid Plot"):
        st.write(sns.jointplot(data=data, x="sepal_length", y="sepal_width", hue="species"))
        # Use Matplotlib to render seaborn
        st.pyplot()

    # Show kdeplot Plots with Sns
    if st.checkbox("Simple kdeplot"):
        st.write(sns.kdeplot(data=data))
        # Use Matplotlib to render seaborn
        st.pyplot()

    # Show rugplot Plots with Sns
    if st.checkbox("Simple rugplot"):
        st.write(sns.rugplot(data=data, x="sepal_length", y="sepal_width", hue="species"))
        # Use Matplotlib to render seaborn
        st.pyplot()

    # Show violinplot Plots with Sns
    if st.checkbox("Simple violinplot"):
        st.write(sns.violinplot(y="sepal_length", x="species", data=data))
        # Use Matplotlib to render seaborn
        st.pyplot()
```



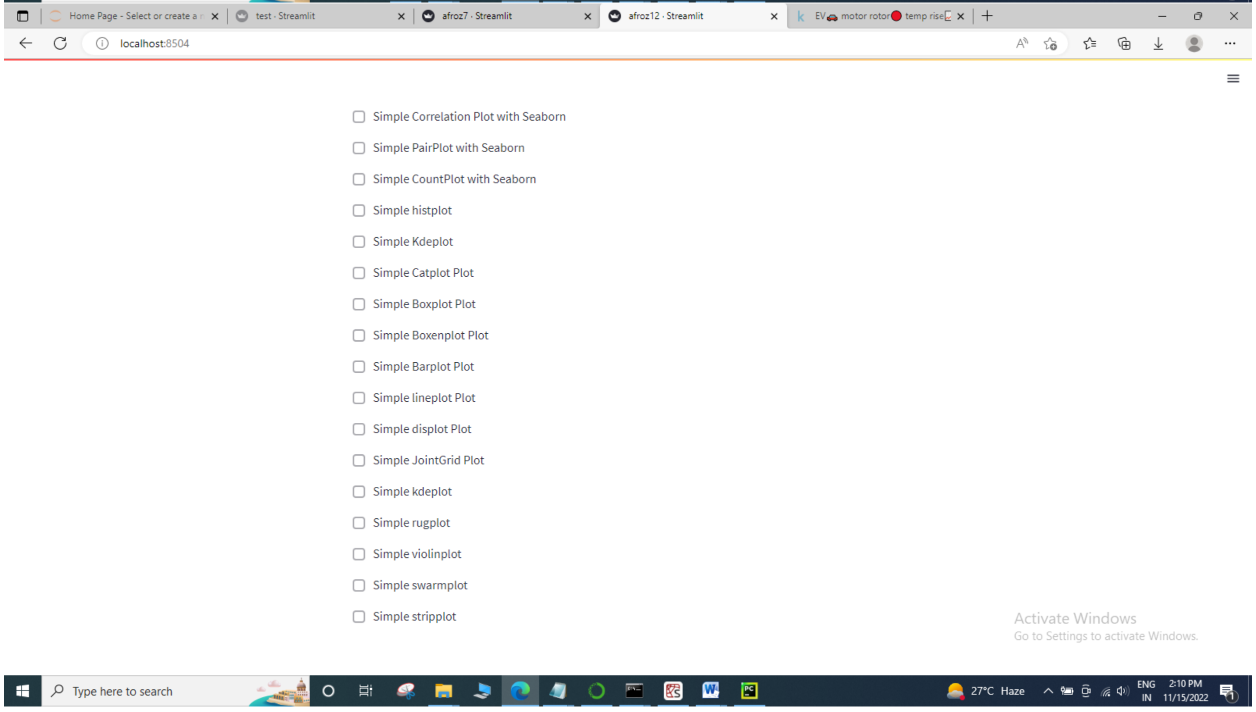
```
    # Show swarmplot Plots with Sns
    if st.checkbox("Simple swarmplot"):
        st.write(sns.swarmplot(x="species", y="sepal_length", data=data))
        # Use Matplotlib to render seaborn
        st.pyplot()

        # Show StripPlot Plots with Sns
    if st.checkbox("Simple stripplot"):
        st.write(sns.stripplot(x="species", y="sepal_length", hue="species", data=data))
        # Use Matplotlib to render seaborn
        st.pyplot()

if __name__ == "__main__":
    main()
```



Iris EDA App

☑ Simple Scatter Plot

```
<matplotlib.collections.PathCollection object at 0x00000227F50B2EB0>
```



☐ Simple Correlation Plot with Seaborn
☐ Simple PairPlot with Seaborn
☐ Simple CountPlot with Seaborn
☐ Simple histplot
☐ Simple Kdeplot
☐ Simple Catplot Plot
☐ Simple Boxplot Plot
☐ Simple Boxenplot Plot
☐ Simple Barplot Plot
☐ Simple lineplot Plot
☐ Simple displot Plot
☐ Simple JointGrid Plot
☐ Simple kdeplot
☐ Simple rugplot
☐ Simple violinplot
☐ Simple swarmplot
☐ Simple stripplot