

LAPORAN PRAKTIKUM

APLIKASI WEB

OLEH: MUHAMMAD EFFLIN RIZQALLAH LIMBONG (22537144007)

MODUL 3

TOPIK:

MODULE 3 (Progress Bar, Widget, Caching, Multi Pages, and Themes)

MODULE 4 (App Model and Development)

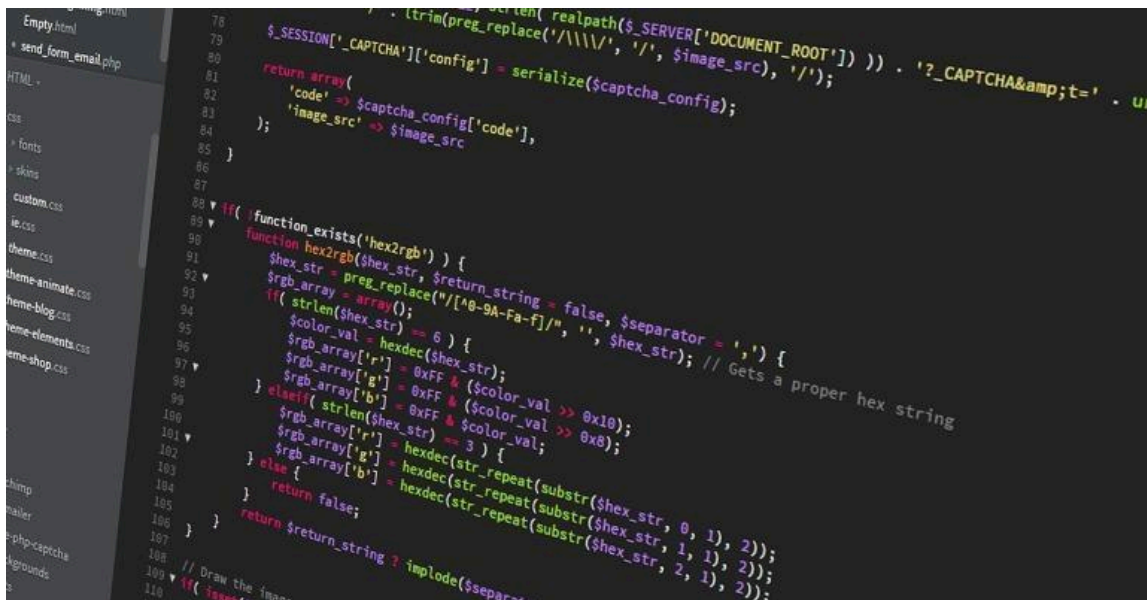


TABLE OF CONTENTS

Week #4	1
A. Penjelasan Tugas Praktikum	3
B. Langkah-langkah dan Screenshot	4
C. Kendala yang Dialami	22
D. Kesimpulan	22

A. Penjelasan Tugas Praktikum

Streamlit merupakan library yang dimiliki Python yang bersifat open source. Streamlit sendiri dikeluarkan pada bulan Oktober tahun 2019. Ada beberapa Package yang terkandung pada Streamlit, diantaranya adalah Flask dan Django. Streamlit biasanya digunakan untuk membuat aplikasi web (Web Apps). Selain dari itu, Streamlit bersifat open source sehingga mudah untuk dibagikan ke pengguna-pengguna lain. Streamlit dapat memudahkan pengguna untuk mengubah data script menjadi aplikasi berbasis web yang interaktif.

Ketika Streamlit dieksekusi atau dijalankan, streamlit akan membuat server lokal. Aplikasi yang dibuat akan tampil di tab browser secara default. Tab browser tersebut adalah tempat dimana pengguna dapat membuat chart, text, widget, table, dan lain-lain.

Tugas praktikum pada pertemuan ini adalah mengeksplorasi dan mempraktikkan beberapa fitur Streamlit, diantaranya adalah Progress Bar Widget, Caching, Multi Pages, dan Session & Callback. Selain dari itu, tugas berikutnya adalah untuk mengeksplorasi dan mempraktikkan

B. Langkah-langkah dan Screenshot

MODULE 3

A. STATUS WIDGET:

1. st.progress():

Pada saat menambahkan komputasi yang berjalan secara lama, dapat ditambahkan 'st.progress()' untuk menampilkan status secara real time. Untuk itu, perlu diimport 'time' agar dapat menggunakan method 'time.sleep()' untuk mensimulasikan komputasi yang berjalan lama.

Kode Program:

```
import streamlit as st

import time

# Streamlit Progress Bar -----

'Memulai Komputasi yang Lama'

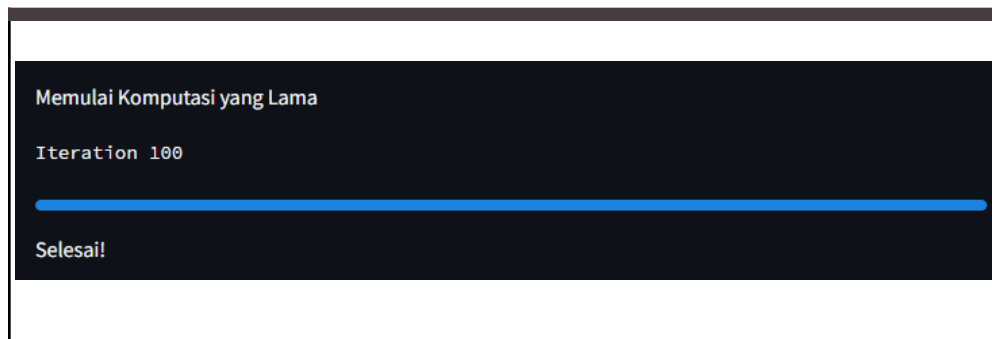
# Menambahkan Placeholder

newest_Iteration = st.empty()

progBar = st.progress(0)
```

```
for i in range(100):  
  
    # Mengupdate Progress Bar untuk setiap Iterasi  
  
    newest_Iteration.text(f'Iteration {i+1}')  
    progBar.progress(i + 1)  
  
    time.sleep(0.1)  
  
'Selesai!'
```

Screenshot:



2. st.spinner():

st.spinner() akan menampilkan pesan untuk sementara sembari mengeksekusi block kode.

Kode Program:

```
import streamlit as st

import time

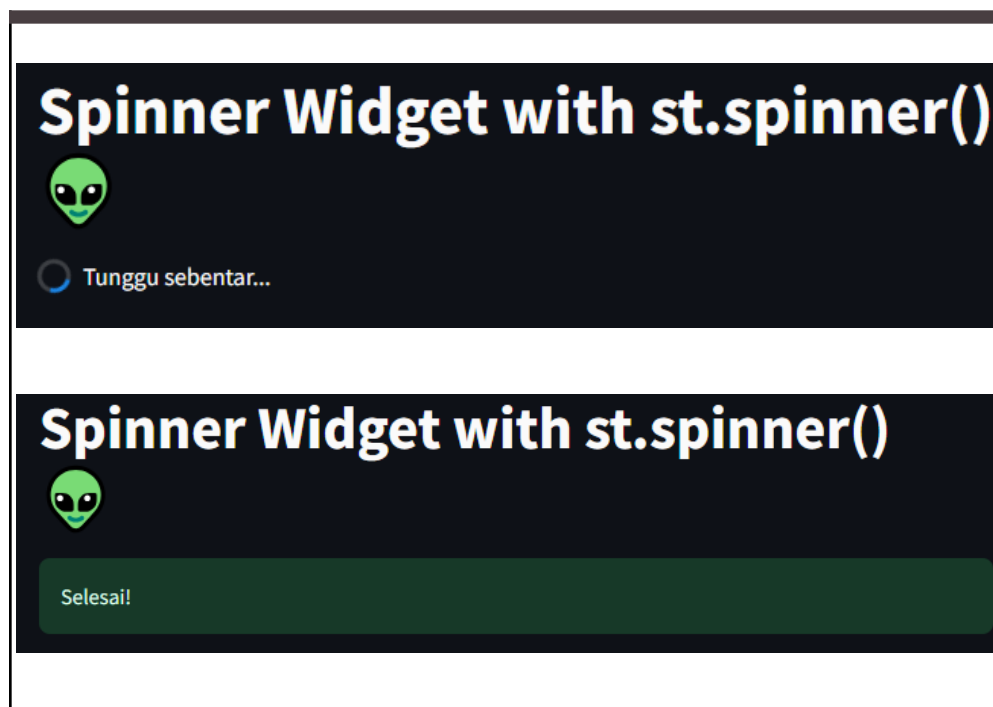
# Streamlit st.spinner()

with st.spinner('Tunggu sebentar...'):

    time.sleep(5)

st.success("Selesai!")
```

Screenshot:



3. `st.balloons()`:

`st.balloons()` akan menampilkan selebrasi dengan balon-balon.

Kode Program:

```
import streamlit as st

# Streamlit st.balloons()

st.balloons()
```

Screenshot:



4. `st.snowflake()`:

`st.snowflake()` akan menampilkan selebrasi dengan salju (Snowflakes).

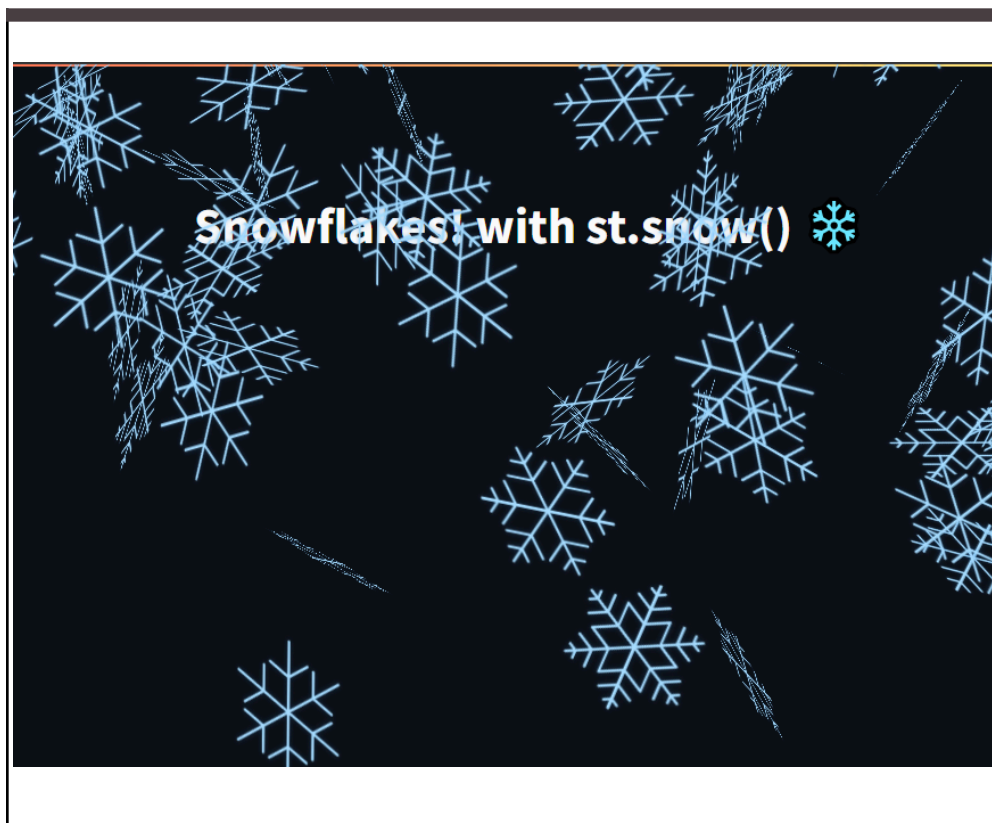
Kode Program:

```
import streamlit as st

# Streamlit st.balloons()

st.snow()
```

Screenshot:



5. `st.error()`:

`st.error()` akan menampilkan pesan error pada aplikasi.

Kode Program:

```
import streamlit as st

# Streamlit st.error()

st.error('Error Telah Terjadi!')
```

Screenshot:



6. `st.warning()`:

`st.warning()` digunakan untuk menampilkan pesan warning dalam kotak kuning pada aplikasi.

Kode Program:

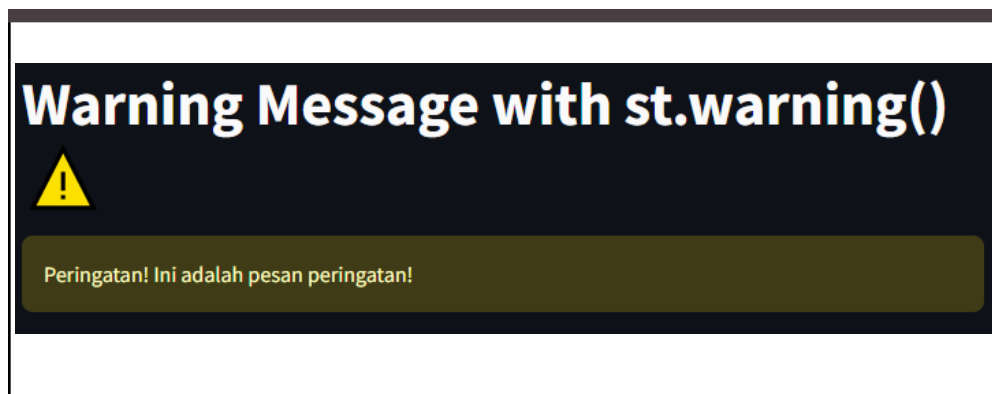
```
# st.text()
```

```
import streamlit as st

# Streamlit st.error()

st.warning('Peringatan! Ini adalah pesan
peringatan!')
```

Screenshot:



B. CACHING:

Streamlit cache memungkinkan aplikasi untuk berjalan lebih cepat walaupun sedang memuat data dari web, memanipulasi dataset-dataset besar, atau melakukan komputasi yang besar (mahal).

Untuk menggunakan cache, bungkus fungsi dengan decorator '@st.cache_data'. Opsi lain adalah dengan menggunakan:

```
st.cache(func=None, persist=False, allow_output_mutation=False,
show_spinner=True, suppress_st_warning=False, hash_funcs=None,
max_entries=None, ttl=None)
```

Kode Program:

```
import streamlit as st

import pandas as pd

@st.cache_data # Penambahan decorator caching

def load_data(url):

    df = pd.read_csv(url)


    return df

df = load_data("https://github.com/plotly/datasets/raw/master/uber-rides-data1.csv")

st.dataframe(df)

st.button("Rerun")
```

Screenshot:

 RUNNING... Stop Deploy

Running `load_data(...)`.

	Date/Time	Lat	Lon
0	2014-04-01 00:11:00	40.769	-73.9549
1	2014-04-01 00:17:00	40.7267	-74.0345
2	2014-04-01 00:21:00	40.7316	-73.9873
3	2014-04-01 00:28:00	40.7588	-73.9776
4	2014-04-01 00:33:00	40.7594	-73.9722
5	2014-04-01 00:33:00	40.7383	-74.0403
6	2014-04-01 00:39:00	40.7223	-73.9887
7	2014-04-01 00:45:00	40.762	-73.979
8	2014-04-01 00:55:00	40.7524	-73.996
9	2014-04-01 01:01:00	40.7575	-73.9846

Rerun

C. MULTI PAGES:

Aplikasi Multi pages sama mudahnya dengan membuat aplikasi satu halaman. Cukup menambahkan lebih banyak halaman ke aplikasi yang sudah ada.

Di folder yang berisi kode script utama, buat folder baru bernama "pages". Tambahkan file .py baru di folder tersebut untuk menambahkan lebih banyak halaman ke aplikasi. Jalankan streamlit dan jalankan kode script utama seperti biasa.

Kode Program Halaman Pertama:

```
import streamlit as st

st.markdown("# THE FIRST PAGE :one:")

st.sidebar.markdown("Halaman Pertama :one:")

st.write("Selamat Datang di Halaman Utama!")
```

Kode Program Halaman Kedua:

```
import streamlit as st

st.markdown("# THE SECOND PAGE :two:")

st.sidebar.markdown("Halaman Kedua :two:")

st.write("Selamat Datang di Halaman Kedua!")
```

Kode Program Halaman Ketiga:

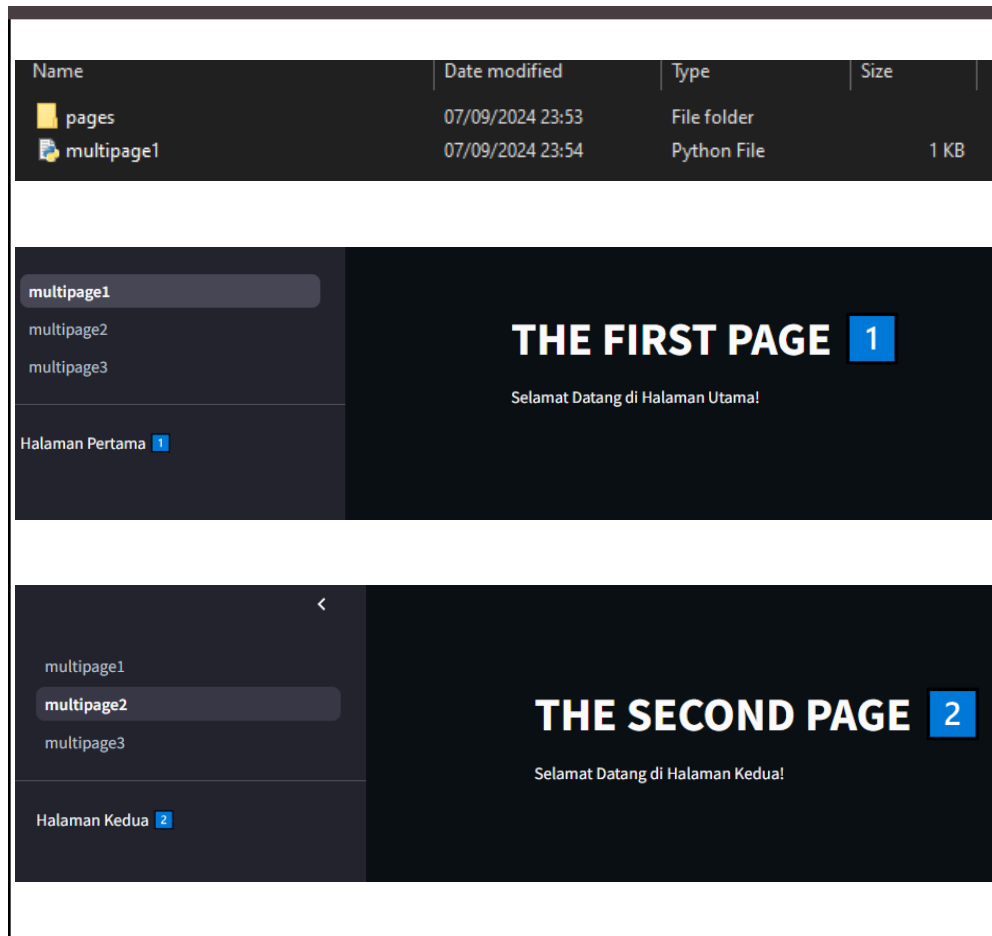
```
import streamlit as st

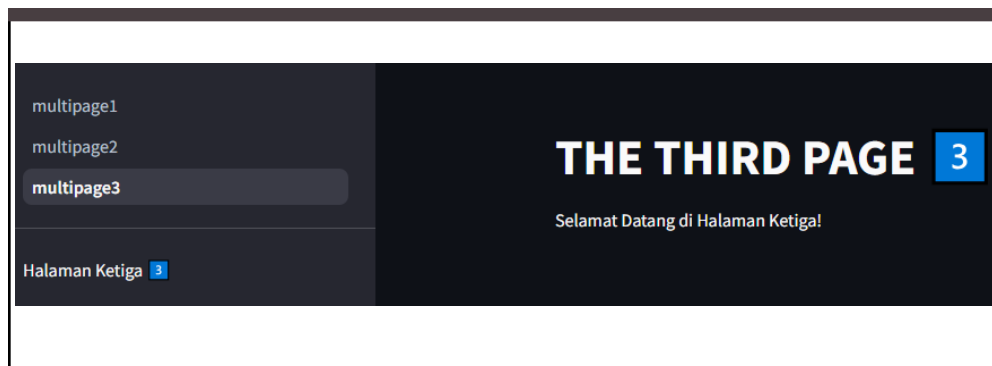
st.markdown("# THE THIRD PAGE :three:")

st.sidebar.markdown("Halaman Ketiga :three:")

st.write("Selamat Datang di Halaman Ketiga!")
```

Screenshot:





D. SESSION DAN CALLBACK:

Kode Program Halaman Pertama:

```
import streamlit as st

#without any arguments (args,kwargs)

def form_callback():

    st.write(st.session_state.my_slider)

    st.write(st.session_state.my_checkbox)

with st.form(key='my_form'):

    slider_input = st.slider('My slider', 0, 10, 5,
key='my_slider')

    checkbox_input = st.checkbox('Iya atau Tidak',
key='my_checkbox')
```

```
submit_button =  
st.form_submit_button(label='Submit',  
on_click=form_callback)
```

Screenshot:

The image displays two screenshots of a web application interface. Both screenshots show a dark-themed UI with a slider and a checkbox.

Top Screenshot: The slider is labeled "My slider" and has a range from 0 to 10. The slider handle is positioned at 5. Below the slider is a checkbox labeled "Iya atau Tidak" which is unchecked. A "Submit" button is located below the checkbox.

Bottom Screenshot: This screenshot is identical to the top one, but it includes two small green boxes at the top left. The first box contains the number "5", and the second box contains the word "False".

The image shows a Streamlit web application interface. At the top, there is a green status bar with the number '5' and the word 'True'. Below this, there is a section titled 'Myslider' containing a horizontal slider. The slider has a red track and a red handle positioned at the value '5'. The range of the slider is from 0 to 10. Below the slider, there is a checkbox labeled 'Iya atau Tidak' which is checked. At the bottom of this section, there is a red 'Submit' button.

Module 4 (App Model and Development)

Kode Program:

```
import streamlit as st

import pandas as pd

import numpy as np

# Atur Judul dan Jalankan

st.title("UBER PICKUPS DI NYC")

# Atur Constant

DATE_COLUMN = 'date/time'
```

```

DATA_URL =
'https://s3-us-west-2.amazonaws.com/streamlit-demo-
data/uber-raw-data-sep14.csv.gz'

# Membuat Fungsi Data Loader (ditambahkan dekorator
cache st.cache_data)

@st.cache_data

def load_data(nrows):

    data = pd.read_csv(DATA_URL, nrows=nrows)

    # Mentransformasi nama kolom menjadi lowercase

    lowercase = lambda x: str(x).lower()

    data.rename(lowercase, axis='columns',
inplace=True)

    # Mentransformasi data kolom ke dalam format
datetime

    data[DATE_COLUMN] =
pd.to_datetime(data[DATE_COLUMN])

    return data

# Menampilkan Data ke Streamlit

```

```

# Mengambil Data. Jalankan

data_load_state = st.text('Memuat data...')

data = load_data(10000)

data_load_state.text("Selesai! (menggunakan
st.cache_data)")

# Memperlihatkan dan Menyembunyikan DataFrame.
Jalankan

if st.checkbox('Menampilkan Data Mentah'):

    st.subheader('Data Mentah')

    st.write(data)

# Menambahkan Histogram. Jalankan

st.subheader('Jumlah Pickup per Jam')

hist_values =
np.histogram(data[DATE_COLUMN].dt.hour, bins=24,
range=(0,24))[0]

st.bar_chart(hist_values)

# Menambahkan Maps

```

```
# Filter-Filter Map Berdasarkan Filter On The Hour

# Nomor pada rentang 0-23

hour_to_filter = st.slider('hour', 0, 23, 17)

filtered_data = data[data[DATE_COLUMN].dt.hour ==
hour_to_filter]

# Perlihatkan Map. Jalankan

st.subheader('Map of all pickups at %s:00' %
hour_to_filter)

st.map(filtered_data)
```

Screenshot:

UBER PICKUPS DI NYC

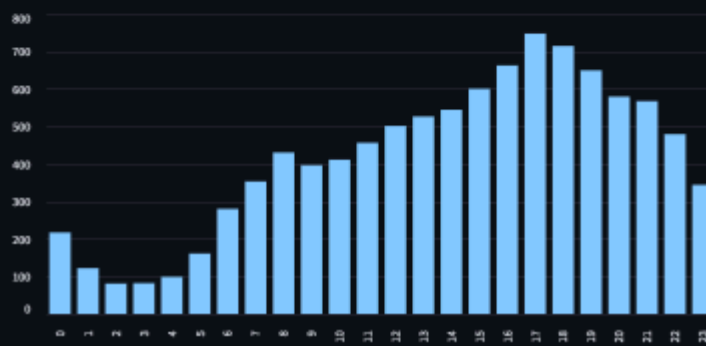
Selesai! (menggunakan st.cache_data)

☒ Menampilkan Data Mentah

Data Mentah

	date/time	lat	lon	base
0	2014-09-01 00:01:00	40.2201	-74.0021	B02512
1	2014-09-01 00:01:00	40.75	-74.0027	B02512
2	2014-09-01 00:03:00	40.7559	-73.9864	B02512
3	2014-09-01 00:06:00	40.745	-73.9889	B02512
4	2014-09-01 00:11:00	40.8145	-73.9444	B02512
5	2014-09-01 00:12:00	40.6735	-73.9918	B02512
6	2014-09-01 00:15:00	40.7471	-73.6472	B02512
7	2014-09-01 00:16:00	40.6613	-74.2691	B02512
8	2014-09-01 00:32:00	40.3745	-73.9999	B02512
9	2014-09-01 00:33:00	40.7633	-73.9773	B02512

Jumlah Pickup per Jam





C. Kendala yang Dialami

Untuk praktikum ini, kami selaku peserta praktikum mengalami kendala pada modul 3, terkhususnya bagian Session dan Callback

D. Kesimpulan

Melalui praktikum ini, peserta praktikum telah memahami konsep dan mempraktikkan sendiri beberapa fitur Streamlit, diantaranya adalah Widget yang terdiri dari; Progress Bar (`st.progress()`), Spinner (`st.spinner()`), Balloons (`st.balloons()`), Snowflakes (`st.snow()`), Error Box (`st.error()`), dan Warning Box (`st.warning()`).

Selanjutnya, peserta praktikum juga telah memahami dan mempraktikkan fitur Caching pada Streamlit dengan menggunakan decorator `@st.cache_data`. Kemudian juga mengenai konsep Multipages, Session, serta Callback.

Selain itu, peserta praktikum telah memahami dan mempraktikkan sendiri tentang bagaimana konsep dari sebuah model aplikasi dan pengembangannya menggunakan Streamlit dengan library Numpy dan Pandas.