**Overview**

This project consists of three main components: data scraping, data storage in a PostgreSQL database, and a Streamlit application for viewing and filtering the data.

**Data Scraping**

The data scraping script collects bus route and bus details from multiple websites and stores the data in a CSV file. It uses Selenium for web scraping and Pandas for data manipulation.

**Data Storage in PostgreSQL**

The data collected through web scraping is saved into a PostgreSQL database. This involves creating a database and a table, then inserting the data from the CSV files into the table.

**Streamlit Application**

A Streamlit application is used to fetch the data from the PostgreSQL database and provide an interface for users to filter and view the bus details.

**Detailed Documentation**

**Data Scraping Script**

The data scraping script performs the following steps:

1. **Import Required Libraries**:
   * psycopg2: PostgreSQL database adapter for Python.
   * pandas: Library for data manipulation and analysis.
   * sqlalchemy: SQLAlchemy engine for database interaction.
   * selenium: Web scraping tool.
   * Other libraries for handling time, exceptions, and datetime.
2. **Database Connection Details**:

db\_user = 'postgres'

db\_password = 'sivan23'

db\_host = 'localhost'

db\_port = '5432'

db\_name = 'bus\_details'

1. **Function to Connect to PostgreSQL Database**:

def connect\_db():

return psycopg2.connect(

dbname=db\_name,

user=db\_user,

password=db\_password,

host=db\_host,

port=db\_port

)

1. **Initialize WebDriver**:

driver = webdriver.Chrome()

1. **List of Websites to Scrape**:

websites = [...]

1. **Scraping Route Data**:
   * Loop through each website.
   * Load the website and wait for the page to load.
   * Scrape the route names and links.
   * Handle pagination to get all route details.
2. **Store Route Data in DataFrame and CSV**:

df\_route = pd.DataFrame({'route\_name': route\_names, 'route\_link': route\_links})

df\_route.to\_csv('route\_data.csv', index=False)

1. **Scrape Bus Details**:
   * Loop through each route link.
   * Load the link and wait for the page to load.
   * Extract bus details such as bus name, type, departing time, duration, reaching time, star rating, price, and seat availability.
2. **Store Bus Data in DataFrame and CSV**:

df\_bus = pd.DataFrame({

'id': ids,

'route\_name': bus\_route\_names,

'route\_link': bus\_route\_links,

'bus\_name': bus\_names,

'bus\_type': bus\_types,

'departing\_time': departing\_times,

'duration': durations,

'reaching\_time': reaching\_times,

'star\_rating': star\_ratings,

'price': prices,

'seat\_availability': seat\_availabilities

})

df\_bus.to\_csv('bus\_data.csv', index=False)

1. **Close WebDriver**:

driver.quit()

**Storing Data in PostgreSQL**

1. **Create Database**:

conn = psycopg2.connect(...)

conn.autocommit = True

cur = conn.cursor()

cur.execute(sql.SQL("CREATE DATABASE {}").format(sql.Identifier(db\_name)))

cur.close()

conn.close()

1. **Connect to the New Database and Create Table**:

conn = connect\_db()

cur = conn.cursor()

create\_bus\_table = """

CREATE TABLE IF NOT EXISTS bus\_data (

id SERIAL PRIMARY KEY,

route\_name TEXT,

route\_link TEXT,

busname TEXT,

bustype TEXT,

departing\_time TIME,

duration TEXT,

reaching\_time TIME,

star\_rating FLOAT,

price DECIMAL,

seats\_available INT

);

"""

cur.execute(create\_bus\_table)

conn.commit()

cur.close()

conn.close()

1. **Insert Data into PostgreSQL Table**:

connection\_string = f'postgresql+psycopg2://{db\_user}:{db\_password}@{db\_host}:{db\_port}/{db\_name}'

engine = create\_engine(connection\_string)

df\_bus = pd.read\_csv('bus\_data.csv')

df\_bus.to\_sql('bus\_data', engine, index=False, if\_exists='replace')

**Streamlit Application**

1. **Import Required Libraries**:

import streamlit as st

import pandas as pd

from sqlalchemy import create\_engine

1. **Database Connection Details**:

db\_user = 'postgres'

db\_password = 'sivan23'

db\_host = 'localhost'

db\_port = '5432'

db\_name = 'bus\_details'

connection\_string = f'postgresql+psycopg2://{db\_user}:{db\_password}@{db\_host}:{db\_port}/{db\_name}'

1. **Function to Get Data from PostgreSQL**:

def get\_data\_from\_postgres():

try:

engine = create\_engine(connection\_string)

query = "SELECT \* FROM bus\_data"

df = pd.read\_sql(query, engine)

return df

except Exception as e:

st.error(f"Error fetching data: {e}")

return None

1. **Streamlit Application Code**:

def main():

st.title("Bus Details")

df = get\_data\_from\_postgres()

if df is not None and not df.empty:

st.write("Current List", df)

st.sidebar.header("Filter Options")

if 'route\_name' in df.columns:

unique\_route\_names = df['route\_name'].unique()

selected\_route\_name = st.sidebar.selectbox("Route Name", ["All"] + list(unique\_route\_names))

if selected\_route\_name != "All":

df = df[df['route\_name'] == selected\_route\_name]

if 'bus\_type' in df.columns:

unique\_bus\_types = df['bus\_type'].unique()

selected\_bus\_type = st.sidebar.selectbox("Bus Type", ["All"] + list(unique\_bus\_types))

if selected\_bus\_type != "All":

df = df[df['bus\_type'] == selected\_bus\_type]

if 'star\_rating' in df.columns:

star\_ratings = df['star\_rating'].dropna().unique()

star\_ratings = sorted(set(star\_ratings))

star\_ratings = [round(rating, 1) for rating in star\_ratings]

selected\_star\_rating = st.sidebar.selectbox("Star Rating", ["All"] + star\_ratings)

if selected\_star\_rating != "All":

df = df[df['star\_rating'] == selected\_star\_rating]

if 'price' in df.columns:

df['price'] = df['price'].replace('[\₹,]', '', regex=True)

df['price'] = pd.to\_numeric(df['price'], errors='coerce')

df = df.dropna(subset=['price'])

price\_options = sorted(df['price'].unique())

selected\_price = st.sidebar.selectbox("Price", ["All"] + price\_options)

if selected\_price != "All":

df = df[df['price'] == selected\_price]

st.write("Filtered Details", df)

else:

st.write("No data found or an error occurred.")

if \_\_name\_\_ == "\_\_main\_\_":

main()