This is viknesh SR I have created a red bus project using Selenium, SQL,streamlit

Python is the main language used here.

1)selenium(for data extraction)

in this project, selenium is used for data extraction from the Red Bus website for 10 states. For all the 10 states we have extracted all the bus routes and for all the bus routes we have extracted all the bus details which included bus name, bus type, star rating, price, bus timing, duration and available seats.

**Web Scraping Bus Routes and Details using Selenium**

**Overview**: This Python script automates the process of collecting bus route and bus details from multiple state transport corporation pages on the Redbus website. The script uses Selenium for browser automation, scraping data from different pages and saving it to CSV files.

**Key Libraries**:

* selenium: For browser automation and web scraping.
* pandas: For data manipulation and saving scraped data to CSV.
* time: For managing time intervals between actions during scraping

**Navigating Pages**:(for page\_number in range(1, 10):)

* This block of code handles pagination, scraping multiple pages (up to 10) for each state:

**Scrolling Function**:(def scroll():)

* Automates scrolling to the bottom of the page to load additional bus routes.

**Scraping Bus Routes and Bus Details**:(def scrape\_route\_page():)

* Collects bus routes from the current page and stores route names and links in a list.

**Extracting Bus Details**:(for route in route\_data:)

* Extracts data for each bus on the current route, such as bus name, type, departure time, duration, rating, price, and seat availability

**Saving Data to CSV**:

* Each state's bus details are saved as a separate CSV file.

2)**Combining CSV Data and Inserting into PostgreSQL**

**Overview**: This script combines multiple CSV files (one for each state) into a single dataset and inserts it into a PostgreSQL database. It also extracts seat availability as numerical data.

**Key Libraries**:

* pandas: For loading, merging, and cleaning data from CSV files.
* psycopg2: For interacting with a PostgreSQL database.
* re: For extracting numerical values from seat availability text.

**Combining CSV Files**:

* Loads CSV files for each state, concatenates them, and stores the combined data in a new CSV (bus\_route.csv).

**Extracting Seat Availability**:

* Uses regex to extract numbers from the seat availability column, converting text like "5 seats left" into integers

**Database Connection and Table Creation**:

* Connects to the PostgreSQL database and creates a bus\_routes table if it doesn't exist.

**Inserting Data into the Database**:

* Iterates over the rows of the combined dataset and inserts each row into the bus\_routes table in PostgreSQL.

**3)Streamlit Dashboard for Bus Tickets Booking**

**Overview**: This Streamlit application connects to a PostgreSQL database and provides a user-friendly interface to view and filter bus route information. Users can filter data based on route names, price, star ratings, and bus types. The data is fetched from a PostgreSQL table bus\_routes and displayed as a dynamic table on the web interface.

**Key Libraries**:

* streamlit: To build the web application.
* psycopg2: For interacting with the PostgreSQL database.
* pandas: For querying and manipulating data.

1. **Database Connection**

* This function establishes a connection to the PostgreSQL database Redbus hosted on localhost using the credentials provided (user='postgres' and password='1234').

2. **Fetching Route Names**

Queries the database for distinct route names from the bus\_routes table.

* The route names are returned as a list and used to populate the route selection dropdown in the sidebar.

3. **Fetching Filtered Bus Data**

Fetches bus data filtered by route name, star ratings, and bus types.

* Sorts the results based on price in either ascending (Low to High) or descending (High to Low) order.
* The SQL query dynamically adjusts to include optional filters for star ratings and bus types, and the params list is used to safely insert these values into the query.

4. **Main Function to Run the Streamlit Application**

**UI Setup**:

* The application title is set to 'Bus Tickets Booking Dashboard'.
* The sidebar contains filter options such as route name and price sorting. The filters are populated dynamically based on the data fetched from the database.

**Data Handling**:

* If a route is selected, the corresponding bus data is fetched and displayed in a table. Filters for star ratings and bus types are also applied, based on user selections.
* The data is displayed using st.dataframe, allowing users to view and interact with the filtered bus data in the main app area.

**Conclusion**

This overall model provides the automatization for data extraction from the red bus storing the acquired data to the database and using that data and building a web application using streamlit   
 **Screenshots**

Are attached below

