**Main Function Code:**

The main function code includes the scraping of bus route and bus details from the redBus website using Selenium and stores the data into a database. It effectively navigates, extracts, and stores large-scale data from a complex website using automation and database integration.

**Functions used**

**1.extract\_route\_details(driver, state\_name, last\_page\_number)**

* Navigates through all pages for a state until the last page and collects route links and names using the route() function.

**2.get\_bus\_details(driver, bus\_routes\_list)**

* Iterates over the route links in bus\_routes\_list.
* For each route:
  + Navigates to the route page.
  + Clears any filters using **click\_clear\_all\_filters\_button().**
  + Scrolls through the page using **scroll()** to ensure all buses are visible.
  + Extracts bus details like name, type, times, and price.
  + Appends the details to bus\_details\_list.

**3.scroll(driver)**

* Scrolls to the bottom of the page until the entire page is loaded.

**4.click\_view\_buses(driver)**

* Clicks on "**View Buses**" buttons on the page to load bus details.

**5.click\_clear\_all\_filters\_button(driver)**

* Clicks the "**Clear All Filters**" button if available.

**6.route(driver)**

* Finds route elements on the page and extracts:
* Route link (href attribute).
* Route name (title attribute).
* Appends these to bus\_routes\_list.

**Error Handling**

Used try-except Blocks to catch exceptions at various levels of the scraping process to ensure the program continues executing even if some errors occur.

**Database Integration**

**insert\_data\_as\_dataframe(bus\_details\_list):**

This function Converts bus\_details\_list into a pandas DataFrame and Inserts the DataFrame into a database table.

**Db\_operations.py**

This code provides a complete solution for interacting with a MySQL database to manage and query data about bus routes and their details. Here's an explanation of each section:

**Future enhancements:**

Here are some future enhancements for this project to improve its functionality, scalability, and user-friendliness:

**Data Validation and Cleaning**

Implement data validation to ensure that only valid data is inserted into the database.

Check for duplicate records before inserting data to avoid redundancy.

**Advanced Filtering and Search**

Add support for more complex filtering options, such as:

Multiple bus routes or bus types in a single query.

Date and time range for filtering departure times.

**User Authentication and Role Management**

Add user authentication to restrict access to the database.

**Data Visualization and Insights**

Provide dashboards with visualizations like graphs and charts for key metrics (e.g., average price, ratings).

**Streamlit.py**

It is designed to allow users to filter, sort, and explore bus route details based on various criteria like price, rating, seat availability, and departure times.