**Redbus Data Scraping with Selenium & Dynamic Filtering using Streamlit**

Objective:

To scrape the Redbus data for multiple routes and store in SQL database. Build a Streamlit application to analyze the stored data.

Step 1: Data scraping

* Data scraping for the Redbus website is done using Selenium
* Python code for the scraping is Redbus\_scrape.py
* Below is the algorithm used for the web scraping
  + **#website to scrape -** Tag to find the code where scraping is started
  + Use driver.get and launch the website for scraping
  + Identify the elements for the bus routes available and extract the bus route names and corresponding href links
  + Using for loop process bus routes extracted and get into each bus route
  + **Get\_Bus\_details()** – Function to extract all the bus details available under each route
    - **##expand hidden buses** – used to expand few bus that are hidden and later extract those details
    - Initialize multiple lists for getting busname, bus type, departure time,duration, arrival time,rating,bus fare,seat availability
    - **##Scroll till end and get all the division with bus details -** Identify the multiple division where bus details are listed and extract by scrolling till the end of the page
    - **#Extract every bus details**
    - With the extracted details using for loop process each bus division and identify each element like name, time, price etc.
    - Using try and except concept to catch any exception i.e. if any specific element is missing for that bus null value is assigned
    - **# dictionary of lists**
    - After extracting all the elements form a dictionary with all the lists
    - Create a pandas dataframe with the dictionary
    - **#format date and time**
    - **format\_date(date1)** – function to convert the extracted date from website in Date format
    - **format\_time(time)** - function to convert the extracted date from website in Date format
    - Adding new column to Dataframe with Datetime format values combining the date and time fields
    - Return the complete dataframe with all the bus details for the particular route
  + Process all the bus routes the same way and get each elements
  + Concatenate all the dataframe with bus details from every route
  + Write the combined dataframe into csv file

Step 2: Data Storage

* Data storage for the Redbus website is done using MySQL
* Python code for the data processing is Redbus\_data\_storage.py
  + **#Query to create table for storing bus data** – MySQL query to create table to store all the bus details
  + **# Read the csv file with bus details and process** – read the file created with redbus details and process each row and make it compatible to store in the database
  + **#Insert data into table** – Query to insert the data into the table and commit

Step 3: Streamlit Application

* An application to display and filter the scraped data is done using streamlit
* Python code for the streamlit is Red\_bus\_streamlit.py
  + **#Application header** – Add the application name as header
  + **#design columns and add drop down in each column** – Below dropdown is added based on the available data in the table
    - **#Bus route dropdown**
    - **#Bus type dropdown**
    - **#A/C type dropdown**
    - **#rating dropdown**
    - **#Bus fare dropdown**
    - **#Timing dropdown**
  + **####Selecting data from table based on the route selection made –** Select the data from the table for the selected route and display
  + **#==================filtering data based on dropdown selection**
    - For each dropdown filtering of data is done and displayed the resulting data. Filtering of data is done under below tags
    - **#seat option**
    - **#Bus type**
    - **#rating option**
    - **#pricing option**
    - **#Timing option**
  + **#Display filtered data**