REDBUS PROJECT DOCUMENTATION

1. OVERVIEW

This documentation describes how to use Selenium to scrape data of minimum 10 Government State Bus Transport from the RedBus website and store it in a MySQL database. This process involves setting up the environment, writing the web scraping code, managing data storage, data analysis using SQL and data visualization.

2. PREREQUISITES

Before you start, ensure you have the following:

- **Python**: Download and install from python.org.
- **Selenium**: Install via pip.
- **MySQL**: Install MySQL server and MySQL Workbench (for managing the database).
- **MySQL Connector for Python**: Install via pip.
- **WebDriver**: Download the appropriate WebDriver for your browser (e.g., ChromeDriver for Google Chrome).

2.1 Install Required Libraries

pip install selenium: **For web scraping**
pip install mysql-connector-python: **To connect Python with MySQL**
pip install pandas: **For data manipulation and analysis**
pip install sqlalchemy: **SQL toolkit for flexible queries**
pip install streamlit: **Framework for interactive web app in Python**

3. SETUP AND CONFIGURATION

3.1. MySQL Database Setup

1. **Start MySQL Server**: Ensure the MySQL server is running

2. **Create a Database**:

```
CREATE DATABASE IF NOT EXISTS redbus
```

3. **Create a Table:**

```
CREATE TABLE IF NOT EXISTS redbus.bus_route
        (id int primary key auto_increment,
        state_transport_name text,
        route_name text,
        route_link text,
        bus_name text,
        bus_type text,
        departing_time time,
        duration text,
        arrival_time time,
        star_rating float,
        fare_price decimal(10,2),
        seats_available int,
        seat_type text,
        created_on datetime default current_timestamp)
```

4. CODE EXPLANTION

4.1. Web Scraping with Selenium

a) Minimum 10 Government Buses defined in a generalservice.py file as array of dictionary type

```
dict_bus_links=[
        {'route': 'APSRTC', 'route_link': 'https://www.redbus.in/online-booking/apsrtc'},
        {'route': 'KERALA RTC', 'route_link': 'https://www.redbus.in/online-booking/ksrtc-
kerala'},
        {'route': 'TSRTC', 'route_link': 'https://www.redbus.in/online-booking/tsrtc'},
        {'route': 'KTCL', 'route_link': 'https://www.redbus.in/online-booking/ktcl'},
        {'route': 'RSRTC', 'route_link': 'https://www.redbus.in/online-booking/rsrtc'},
        {'route': 'SBSTC', 'route_link': 'https://www.redbus.in/online-booking/south-bengal-
state-transport-corporation-sbstc'},

        {'route': 'HRTC', 'route_link': 'https://www.redbus.in/online-booking/hrtc'},

        {'route': 'ASTC', 'route_link': 'https://www.redbus.in/online-booking/astc'},

        {'route': 'UPSRTC', 'route_link': 'https://www.redbus.in/online-booking/uttar-
pradesh-state-road-transport-corporation-upsrtc'},

        {'route': 'WBTC', 'route_link': 'https://www.redbus.in/online-booking/wbtc-
ctc'}

     ]
```

b) Web Scraping Code in web_scraping_service.py file

```
from selenium import webdriver

from selenium.webdriver.common.by import By

from selenium.common.exceptions import NoSuchElementException

import time
```

# Global Variables

```
   list_pages = []
```

# Setting up the web driver

```
driver = webdriver.Chrome()
```

# Open RedBus Government Bus link

```
driver.get(state_transport_link)
```

# Wait for page to load

```
time.sleep(5)
```

# To maximize the window

```
driver.maximize_window()
```

# To scroll the page by 1500px vertically down

```
driver.execute_script("window.scrollBy(0, 1500);", "")
```

# To get number of pages and store in list_pages

```
page_element = driver.find_element(By.XPATH,"//div[@class='DC_117_paginationTable']")

list_pages.append(page_element.text)
```

```
list_pages = list_pages[0].split('\n')
 # To get last page number
last_page = int(list_pages[-1])
# To start web scraping and scrap data for each pages
    for i in range(1, last_page+1):
        print(str(i))
        if(i > 1):
            # Function to go to route
             go_to_route(driver, state_transport_link)
            # Pause the program for 5 seconds
            time.sleep(5)
            page_navigation(str(i), driver)
            # Pause the program for 5 seconds
            time.sleep(5)
            # Function to Start Web Scraping
            start_webscrapping(state_transport, driver)
        else:
            # Function to Start Web Scraping
            start_webscrapping(state_transport, driver)
# To close the web driver
    driver.close()
```

c) Function to Start Web Scraping

```
def start_webscrapping(state_transport, driver):

    # Pause the program for 5 seconds
    time.sleep(5)
    # Function to get all bus routes in the state transport
    route_buses = get_bus_route(driver)
    for route_bus in route_buses:
        active_bus_route = route_bus['route']
        active_bus_routelink = route_bus['routelink']
        driver.get(active_bus_routelink)
        # Pause the program for 5 seconds
        time.sleep(5)
        # Function to scroll the page down
        if(scroll_down(driver)):
            # Function to Click on View Buses Button
            if(click_view_page(driver)):
                # Function to extract bus details
                list_bus_data = extract_bus_details(state_transport, active_bus_route,
active_bus_routelink, driver)
                # Convert list to dataframe
                df_bus_data = pd.DataFrame(list_bus_data)
                # Insert data into redbus MySQL database
                dbservice.insert_data(df_bus_data)
                # Back to previous page
                driver.back()
            else:
                # Back to previous page
                driver.back()
        else:
            # Back to previous page
            driver.back()
    return

# Function to go to routes
def go_to_route(driver, state_transport_link):
    return driver.get(state_transport_link)
# Function to get all bus routes in the state transport
def get_bus_route(driver):
    list_route_buses = []
    route_buses = driver.find_elements(By.CSS_SELECTOR,"a[class='route']")
    for route_bus in route_buses:
        list_route_buses.append({"route": route_bus.text, "routelink":
route_bus.get_attribute('href')})
    return list_route_buses
```

# Function to Scroll Page Down

```
def scroll_down(driver):
```

# Get scroll height

```
    last_height = driver.execute_script("return document.body.scrollHeight")
    while True:
```

# Scroll down to the bottom

```
        driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
```

# Pause the program execution for 2 seconds

```
        time.sleep(2)
```

# Calculate new scroll height and compare with last scroll height

```
        new_height = driver.execute_script("return document.body.scrollHeight")
        if new_height == last_height:
            return True
        last_height = new_height
```

# Function to Click on View Buses Button

```
def click_view_page(driver):
    driver.execute_script("window.scrollTo(0, document.body.scrollTop);")
    time.sleep(2)


    try:
        buttons = driver.find_elements(By.XPATH, "//div[@class='button' and text()='View
Buses']")
        for m in range(0,len(buttons)):
            driver.execute_script("arguments[0].scrollIntoView(true);", buttons[m])
            driver.execute_script("arguments[0].click()", buttons[m])
        driver.execute_script("window.scrollTo(0, document.body.scrollTop);")
    except:
        print("No View Buses Button Found")
    return True

# Function to Extract Bus Details
def extract_bus_details(state_transport, bus_route, bus_link, driver):
    bus_data_arr = []
    busesdiv = driver.find_elements(By.XPATH,"//div[@class='clearfix row-one']")
    for bus in busesdiv:
        try:
            busname = bus.find_element(By.XPATH,".//div[@class='column-two p-right-10 w-30
fl']//div[@class='travels lh-24 f-bold d-color']").text
        except NoSuchElementException:
            busname = ""
        try:
            bustype = bus.find_element(By.XPATH,".//div[@class='column-two p-right-10 w-30
fl']//div[@class='bus-type f-12 m-top-16 l-color evBus']").text
        except NoSuchElementException:
            bustype = ""
        try:
            busdeparturetime = bus.find_element(By.XPATH,".//div[@class='column-three p-
right-10 w-10 fl']//div[@class='dp-time f-19 d-color f-bold']").text
        except NoSuchElementException:
            busdeparturetime = ""
        try:
            busduration = bus.find_element(By.XPATH,".//div[@class='column-four p-right-10 w-
10 fl']//div[@class='dur l-color lh-24']").text
        except NoSuchElementException:
```

```
    busduration = ""
try:
    busarraivaltime = bus.find_element(By.XPATH,".//div[@class='column-five p-right-10 w-10 fl']//div[@class='bp-time f-19 d-color disp-Inline']").text
except NoSuchElementException:
    busarraivaltime = ""
try:
    busrating = bus.find_element(By.XPATH,".//div[@class='column-six p-right-10 w-10 fl']//div[@class='rating-sec lh-24']").text
except NoSuchElementException:
    busrating = 0.0
try:
    busprice = bus.find_element(By.XPATH,".//div[@class='column-seven p-right-10 w-15 fl']//div[@class='seat-fare ']//div[@class='fare d-block']//span[@class='f-19 f-bold' or @class='f-bold f-19']").text
except NoSuchElementException:
    busprice = "0"
try:
    busseats = bus.find_element(By.XPATH,".//div[@class='column-eight w-15 fl']").text
except NoSuchElementException:
    busseats = ""
if busseats == "":
    busseats = 0
    busseattype = ""
else:
    if "\n" in busseats:
        try:
            busseatsplit = busseats.split("\n")
            # Seats Available
            seatsavailable = busseatsplit[0]
            try:
                seatsavailablesplit = seatsavailable.split(" ")
                busseats = int(seatsavailablesplit[0])
            except:
                busseats = 0
            # Seat Type
            busseattype = busseatsplit[1]
        except:
            busseatsplit = busseats
            try:
                seatsavailablesplit = seatsavailable.split(" ")
                busseats = int(seatsavailablesplit[0])
            except:
                busseats = 0
            # Seat Type
            busseattype = ""
    else:
        # Seats Available
        seatsavailable = busseats
        try:
            seatsavailablesplit = seatsavailable.split(" ")
            busseats = int(seatsavailablesplit[0])
        except:
            busseats = 0
        # Seat Type
        busseattype = ""
bus_data = dict(
    state_transport_name = state_transport,
    route_name = bus_route,
    route_link = bus_link,
            bus_name = busname,
            bus_type = bustype,
            departing_time = busdeparturetime,
            duration = busduration,
            arrival_time = busarraivaltime,
            star_rating = float(busrating),
            fare_price = busprice,
            seats_available = busseats,
            seat_type = busseattype
        )
        bus_data_arr.append(bus_data)
    return bus_data_arr
```

# Function to Navigating Page

```
def page_navigation(page_number, driver):
    driver.execute_script("window.scrollBy(0, 1500);", "")
    time.sleep(5)
    active_page = driver.find_element(By.XPATH,f"//div[@class='DC_117_pageTabs ' and
text()='{page_number}']")
    active_page.click()
```

4.2. Storing Data in MySQL

```
def insert_data(df_data):
    data = tuple(df_data.to_numpy().tolist())
    try:
        db = db_connection()
        cursor = db.cursor()
        cursor.executemany('''INSERT INTO redbus.bus_route
                (state_transport_name, route_name, route_link, bus_name, bus_type,
departing_time, duration, arrival_time, star_rating, fare_price, seats_available, seat_type)
```

```
                VALUES (%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)''', data)
        db.commit()
    except Exception as e:
        print(e)
    finally:
        cursor.close()
        db.close()
```

# Function for MySQL DB Connection

```
def db_connection():
    return
mysql.connector.connect(host=DB_HOST,user=DB_USERNAME,passwd=DB_PASSWORD)
```

5. APPLICATION USAGE

5.1. Running the Scraper

1. **Run the Streamlit**: Execute the Streamlit to start the scraping process.

   Streamlit run main.py

5.2. Verifying Data Storage

1. **Open MySQL Workbench**
2. **Check the database**:

   USE redbus;
   SELECT * FROM bus_route;

   This will display the data stored in the bus_route table

6. ERROR HANDLING AND DEBUGGING

1. **Element Not Found:** Ensure the HTML structure hasn’t changed. Update the locators accordingly.

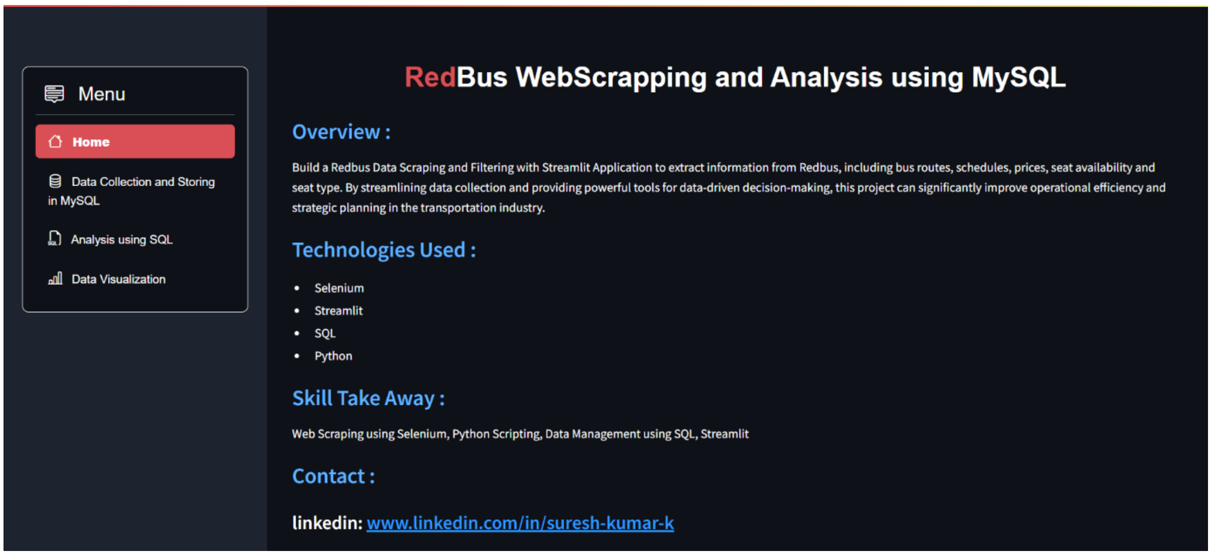
2. **Timeouts:** Increase time.sleep() durations if pages take longer to load.

3. **Connection Errors:** Verify MySQL server is running and credentials are correct.

7. CONCLUSION

This document outlines the steps for scraping data of minimum 10 Government State Transport Buses from RedBus using Selenium and storing it in MySQL. The code samples provided cover essential operations for extracting data and interacting with the database along with the application usage.

SCREEN SHOTS



RedBus WebScrapping and Analysis using MySQL

Overview :

Build a Redbus Data Scraping and Filtering with Streamlit Application to extract information from Redbus, including bus routes, schedules, prices, seat availability and seat type. By streamlining data collection and providing powerful tools for data-driven decision-making, this project can significantly improve operational efficiency and strategic planning in the transportation industry.

Technologies Used :

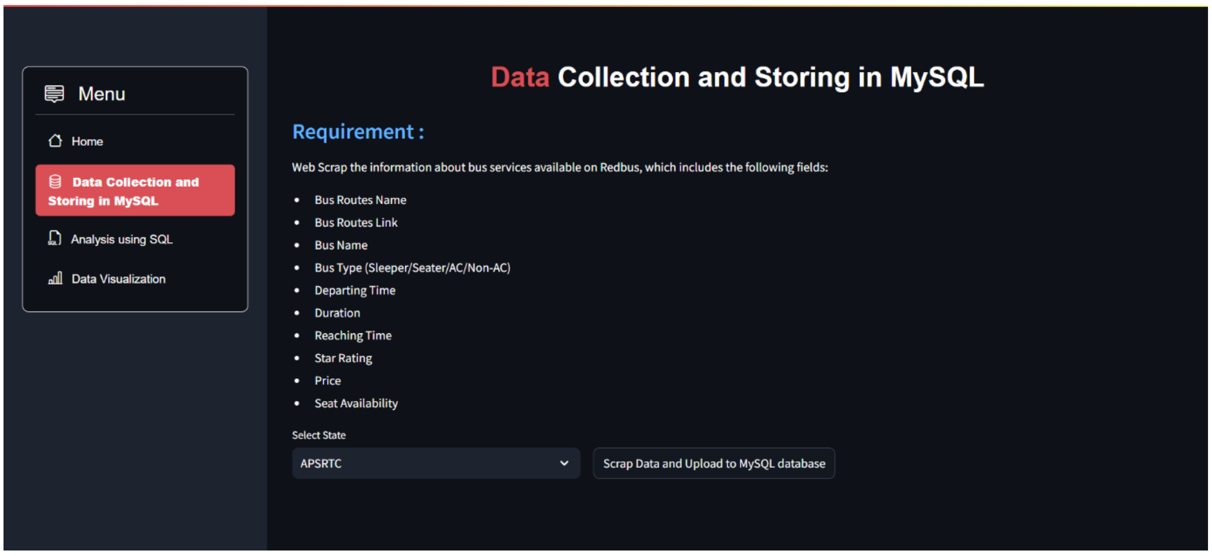
- Selenium
- Streamlit
- SQL
- Python

Skill Take Away :

Web Scraping using Selenium, Python Scripting, Data Management using SQL, Streamlit

Contact :

linkedin: www.linkedin.com/in/suresh-kumar-k


Menu
Home
Data Collection and Storing in MySQL
Analysis using SQL
Data Visualization

Data Collection and Storing in MySQL

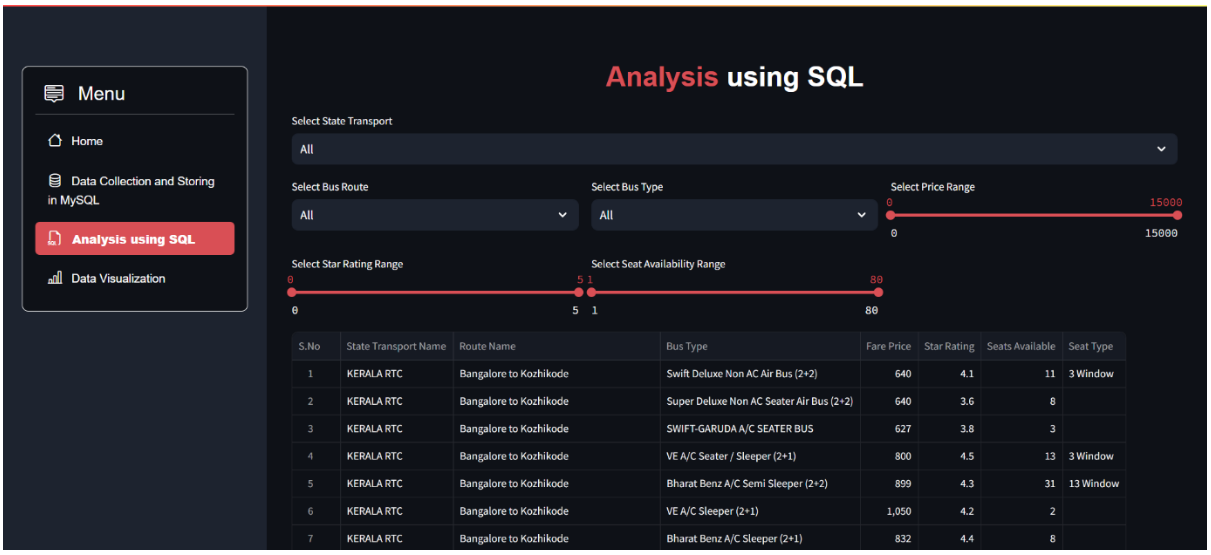
Requirement :

Web Scrap the information about bus services available on Redbus, which includes the following fields:

- Bus Routes Name
- Bus Routes Link
- Bus Name
- Bus Type (Sleeper/Seater/AC/Non-AC)
- Departing Time
- Duration
- Reaching Time
- Star Rating
- Price
- Seat Availability

Select State: APSRTC

Scrap Data and Upload to MySQL database



Analysis using SQL

Select State Transport: All

Select Bus Route: All

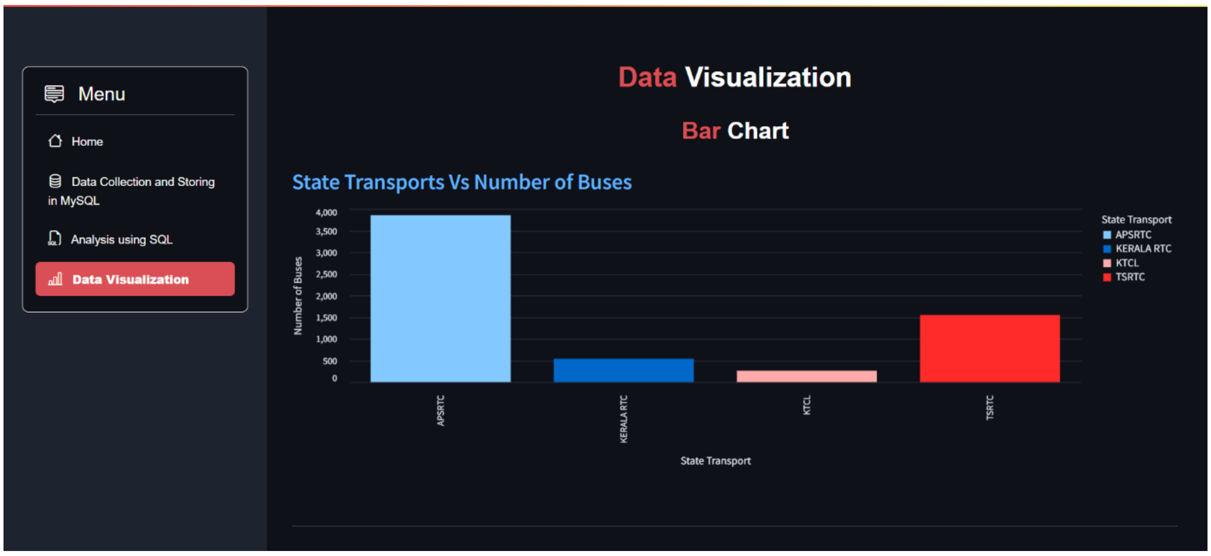
Select Bus Type: All

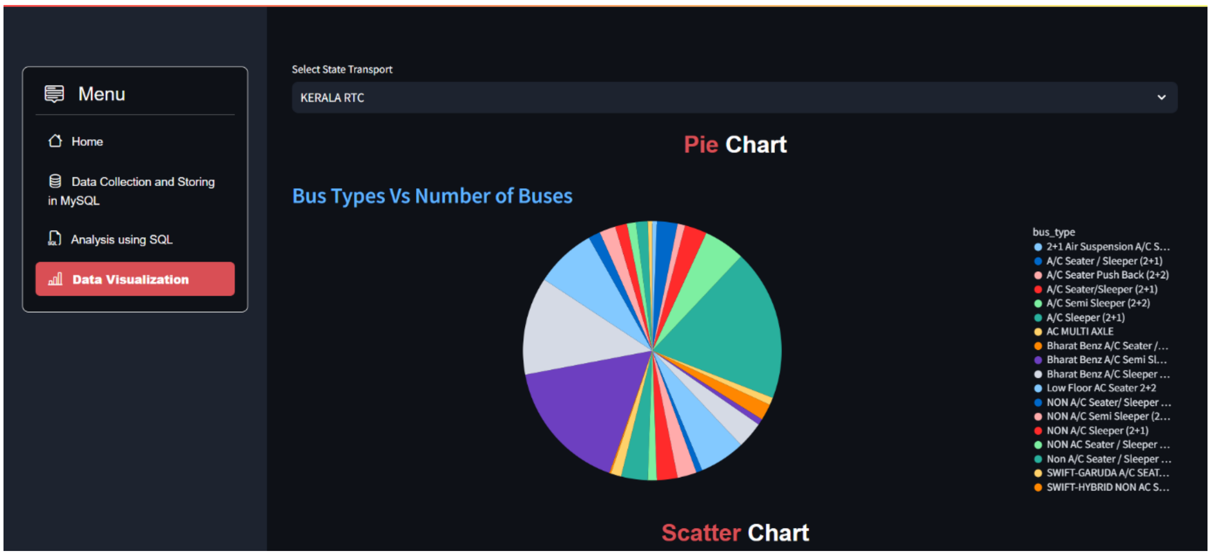
Select Price Range: 0 – 15000

Select Star Rating Range: 0 – 5

Select Seat Availability Range: 1 – 80

S.No	State Transport Name	Route Name	Bus Type	Fare Price	Star Rating	Seats Available	Seat Type
1	KERALA RTC	Bangalore to Kozhikode	Swift Deluxe Non AC Air Bus (2+2)	640	4.1	11	3 Window
2	KERALA RTC	Bangalore to Kozhikode	Super Deluxe Non AC Seater Air Bus (2+2)	640	3.6	8	
3	KERALA RTC	Bangalore to Kozhikode	SWIFT-GARUDA A/C SEATER BUS	627	3.8	3	
4	KERALA RTC	Bangalore to Kozhikode	VE A/C Seater / Sleeper (2+1)	800	4.5	13	3 Window
5	KERALA RTC	Bangalore to Kozhikode	Bharat Benz A/C Semi Sleeper (2+2)	899	4.3	31	13 Window
6	KERALA RTC	Bangalore to Kozhikode	VE A/C Sleeper (2+1)	1,050	4.2	2	
7	KERALA RTC	Bangalore to Kozhikode	Bharat Benz A/C Sleeper (2+1)	832	4.4	8	




Menu
Home
Data Collection and Storing in MySQL
Analysis using SQL
Data Visualization
Select State Transport
KERALA RTC
Pie Chart
Bus Types Vs Number of Buses
bus_type
2+1 Air Suspension A/C S...
A/C Seater / Sleeper (2+1)
A/C Seater Push Back (2+2)
A/C Seater/Sleeper (2+1)
A/C Semi Sleeper (2+2)
A/C Sleeper (2+1)
AC MULTI AXLE
Bharat Benz A/C Seater /...
Bharat Benz A/C Semi Sl...
Bharat Benz A/C Sleeper ...
Low Floor AC Seater 2+2
NON A/C Seater/ Sleeper ...
NON A/C Semi Sleeper (2...
NON A/C Sleeper (2+1)
NON AC Seater / Sleeper ...
Non A/C Seater / Sleeper ...
SWIFT-GARUDA A/C SEAT...
SWIFT-HYBRID NON AC S...
Scatter Chart

`