DATABASE MANAGEMENT SYSTEM
PROJECT REPORT

Semester IV, B.Tech (IT)

VEHICO
A VEHICLE MANAGEMENT PORTAL

GROUP-22

Submitted to : **Dr. Anjali Gautam**

MEMBERS	ROLL NUMBER
YASHPREET SINGH	IIT2019124
ARPIT AGARWAL	IIT2019139
AYUSH TIWARI	IIT2019151
MRINAL BHAVE	IIT2019152

CONTENTS

INTRODUCTION

Vehicle Management System - Vehico is a desktop application which can be used for car booking and rental purposes. This application allows users to book rides by providing them with different options to travel. It also keeps track of all the rides being taken by the passengers, vehicle and fuel expenses, repair history and vehicles assigned to each driver.

Day to day travel made easy and seamless with the use of Vehico, the vehicle management system. Users can easily book a ride on the click of a button on affordable prices without having to hassle. The application is user friendly and easy to use. Drivers can update their information on the app when they're ready to pick up a passenger, and as soon as a passenger books a ride the driver nearest to them is assigned to the passenger. This allows for efficient fuel consumption and lesser pick up time, which makes the user experience better. The passenger can see their real time location in the application throughout the journey, which assures safety.

The repair history of vehicles and other details are maintained in a database accessible to admin, which makes the system efficient and easy to use.

DEVELOPMENT TOOLS

- **Java Swing GUI** : Swing in Java is a Graphical User Interface (GUI) toolkit that includes the GUI components. It provides a rich set of widgets and packages to make sophisticated GUI components for Java applications. Swing is a part of Java Foundation Classes(JFC), which is an API for Java programs that provide GUI.
- **JDBC for MySql (PHPMyAdmin) connectivity** : JDBC drivers are Java library files with the extension .jar used by all Java applications to connect to the database.
- **SQL (queries)**
- **JavaFX-SDK-16** : The JavaFX Software Development Kit (SDK) provides the command-line tools and technologies to develop expressive content for applications deployed to browsers, desktops, and mobile devices.

SYSTEM REQUIREMENTS

- JDK: Latest version
- Operating System: Any
- IDE: Java Netbeans / IntelliJ / Eclipse
- Active Internet connection for accessing google maps.

FEATURES

There are mainly 3 modules in this project.

1. **Admin Management**
    - Login
    - Add Vehicle
    - View Vehicle Details
    - Delete Vehicle
    - Ban Driver
    - View Rides History
    - Vehicle Repair History
    - Manage Fuel Price

2. **Driver Management**
    - Login
    - Register
    - Upon signing up, the driver selects the kind of vehicle he/she drives and is allotted a vehicle (if vacant).
    - Option for turning ride status online and start picking rides.
    - View Previous Rides
    - Check for Current Ride and update ride status (AtPickup / Running / Completed).
    - On completion of the ride, distance is noted and Bill is calculated.

3. **Passenger Management**
    - ❏ Passengers need to sign up, and login thereafter.
    - ❏ Upon logging in, they can Book Car / Rickshaw / Bus Rides.
    - ❏ From the Google Maps (integrated), they can enter From and To locations and begin the ride.
    - ❏ Directions are available via Google Maps for various vehicle routes.
    - ❏ Car & Driver Details can be viewed anytime.
    - ❏ Passengers Ride History

SCHEMA

1. **Admin Table**

   Fields:

    - AdminID varchar(8) *(PRIMARY KEY)*
    - AdminP varchar(30)

2. **Ban Driver Table**

   Fields:

    - ID varchar(50) *(PRIMARY KEY)*
    - CarId varchar(50)
    - Password varchar(50)
    - Name varchar(50)
    - FatherName varchar(50)
    - Gender varchar(10)
    - SecurityQuestion varchar(50)
    - Type varchar(8)
    - aadhar varchar(15)
    - ContactNo varchar(20)

- DOB varchar(20)

3. **Bus Table**

   Fields:

   - BusId int *(PRIMARY KEY)*,
   - Model varchar(50)
   - Version varchar(50)
   - Color varchar(50)
   - PlateNo varchar(50)
   - RegistrationDate varchar(50)
   - ExpirationDate varchar(50)
   - BusName varchar(50)
   - Assigned tinyint
   - Fromm varchar(50)
   - Too varchar(50)

4. **Car table**

   Fields:

   - CarId int *(PRIMARY KEY)*
   - Model varchar(50)
   - Version varchar(50)
   - Color varchar(50)
   - PlateNo varchar(20)
   - RegistrationDate varchar(50)
   - ExpirationDate varchar(50)
   - CarName varchar(50)
   - Assigned tinyint

5. **Driver table**
   Fields:

   - ID varchar(50) *(PRIMARY KEY)*
   - Password varchar(50)
   - Name varchar(50)
   - FatherName varchar(50)
   - Gender varchar(10)
   - SecurityQuestion varchar(50)
   - Type varchar(8)
   - aadhar varchar(15)
   - ContactNo varchar(20)
   - DOB varchar(20)

6. **Driver Bus ride table**
   Fields:

   - ID int *(PRIMARY KEY)*
   - Username varchar(10) *(FOREIGN KEY)*
   - BusId int *(FOREIGN KEY)*
   - DriverAvail tinyint
   - Fromm varchar(50)
   - Too varchar(50)
   - Datee varchar(50)
   - RideStatus varchar(50)
   - StartTime varchar(50)
   - EndTime varchar(50)
   - BillStatus varchar(50)
   - Bill double
   - NoOfPassengers tinyint

7. **Driver Bus Table**
   Fields:

   - SNo int *(PRIMARY KEY)*
   - DriverUsername varchar(20) *(FOREIGN KEY)*

- BusID int *(FOREIGN KEY)*
- Fromm varchar(50)
- Too varchar(50)

8. **Driver Car Ride Table**
   Fields:

   - ID tinyint *(PRIMARY KEY)*
   - Username varchar(50) *(FOREIGN KEY)*
   - CarId int *(FOREIGN KEY)*
   - DriverAvail tinyint
   - PUsername varchar(50) *(FOREIGN KEY)*
   - Fromm varchar(50)
   - Too varchar(50)
   - Datee varchar(50)
   - RideStatus varchar(50)
   - StartTime varchar(20)
   - EndTime varchar(20)
   - BillStatus varchar(50)
   - Bill double

9. **Driver Car Table**
   Fields:

   - SNo int *(PRIMARY KEY)*
   - DriverUsername varchar(50) *(FOREIGN KEY)*
   - CarID int *(FOREIGN KEY)*

10. **Driver Rickshaw Ride Table**
    Fields:

    - ID tinyint *(PRIMARY KEY)*
    - Username varchar(50) *(FOREIGN KEY)*
    - RickshawId int *(FOREIGN KEY)*
    - DriverAvail tinyint
    - PUsername varchar(50) *(FOREIGN KEY)*

- Fromm varchar(50)
- Too varchar(50)
- Datee varchar(20)
- RideStatus varchar(50)
- StartTime varchar(50)
- EndTime varchar(50)
- BillStatus varchar(50)
- Bill float

11. <u>Driver Rickshaw Table</u>

Fields:

- SNo int *(PRIMARY KEY)*
- DriverUsername varchar(50) *(FOREIGN KEY)*
- RickshawID int *(FOREIGN KEY)*

12. <u>Fuel Table</u>

Fields:

- id int *(PRIMARY KEY)*
- price double

13. <u>Passenger Bus Rides Table</u>

Fields:

- SNo int *(PRIMARY KEY)*
- iD int *(FOREIGN KEY)*
- Username varchar(50) *(FOREIGN KEY)*
- VehicleId int *(FOREIGN KEY)*
- PUsername varchar(50) *(FOREIGN KEY)*

14. <u>Passenger Table</u>

Fields:

- PID varchar(50) *(PRIMARY KEY)*
- PPass varchar(50)
- PName varchar(50)
- PFName varchar(50)

- PGender varchar(50)
- PSec varchar(50)
- aadhar varchar(50)
- ContactNo varchar(50)
- DOB varchar(50)

15. <u>Repair History Table</u>

Fields:

- date varchar(50) *(UNIQUE)*
- driver_id varchar(50) *(FOREIGN KEY)*
- carID tinyint
- request varchar(180)

16. <u>Rickshaw Table</u>

Fields:

- RickshawId int *(PRIMARY KEY)*
- Model varchar(50)
- Version varchar(50)
- Color varchar(50)
- PlateNo varchar(20)
- RegistrationDate varchar(50)
- ExpirationDate varchar(50)
- RickshawName varchar(50)
- Assigned tinyint

17. <u>Ride realtime table</u>

Fields:

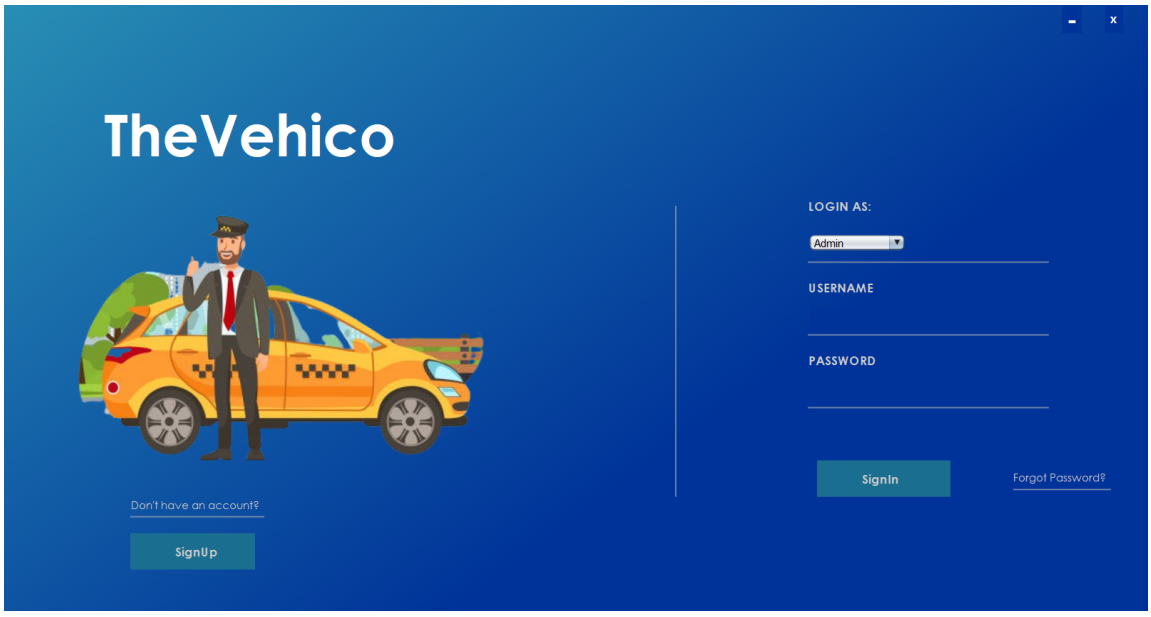
- iD varchar(50) *(PRIMARY KEY)*
- Username varchar(50) *(FOREIGN KEY)*
- VehicleId int
- PUsername varchar(50) *(FOREIGN KEY)*
- Fromm varchar(50)
- Too varchar(50)

- Datee varchar(50)
- RideStatus varchar(50)
- StartTime varchar(50)
- EndTime varchar(50)
- BillStatus varchar(50)
- Bill double
- NoOfPassengers tinyint

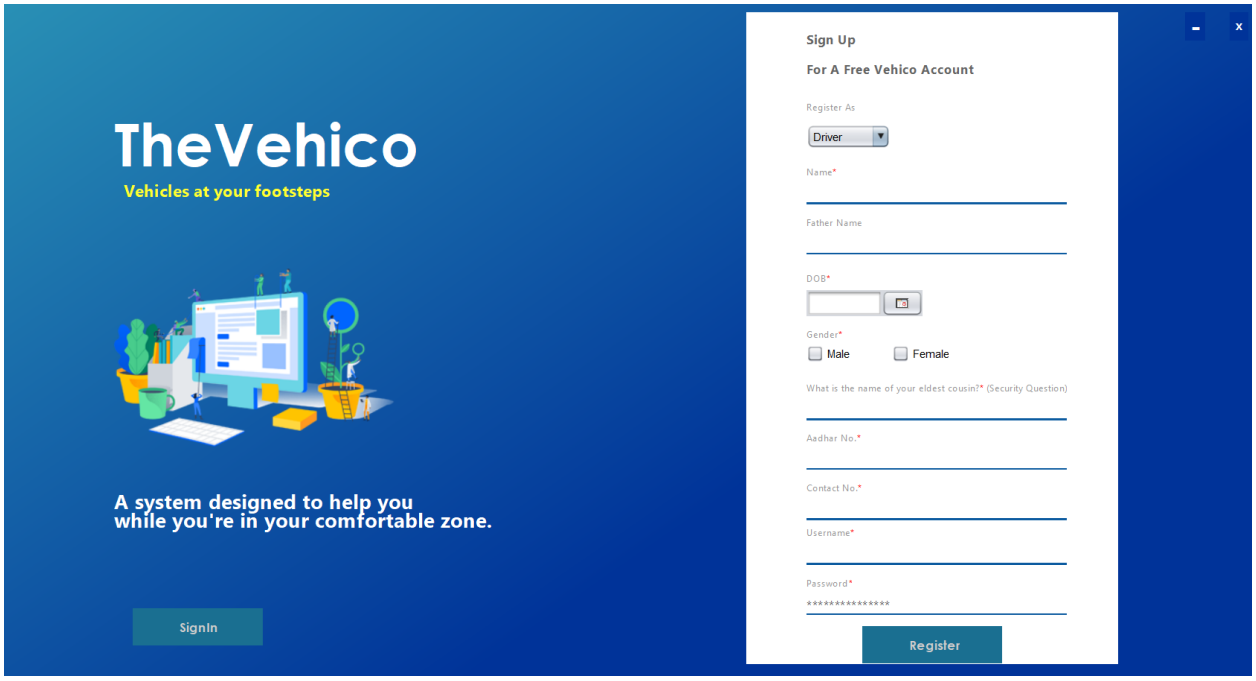
IMPLEMENTATION

The first window which is displayed in the application is the login window.

Login Page: Users can select their role from admin, driver or passenger. On entering username and password, if the entered credentials are correct then the user will be directed to their respective dashboard. If the credentials are incorrect, an error message will be displayed.

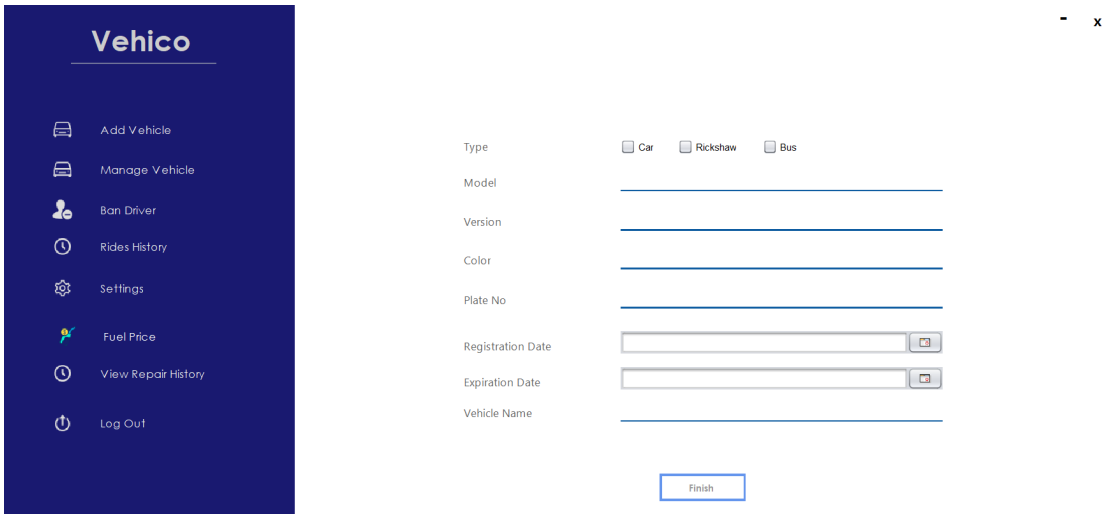


Register Page: On this page the user will be able to register himself driver or passenger .

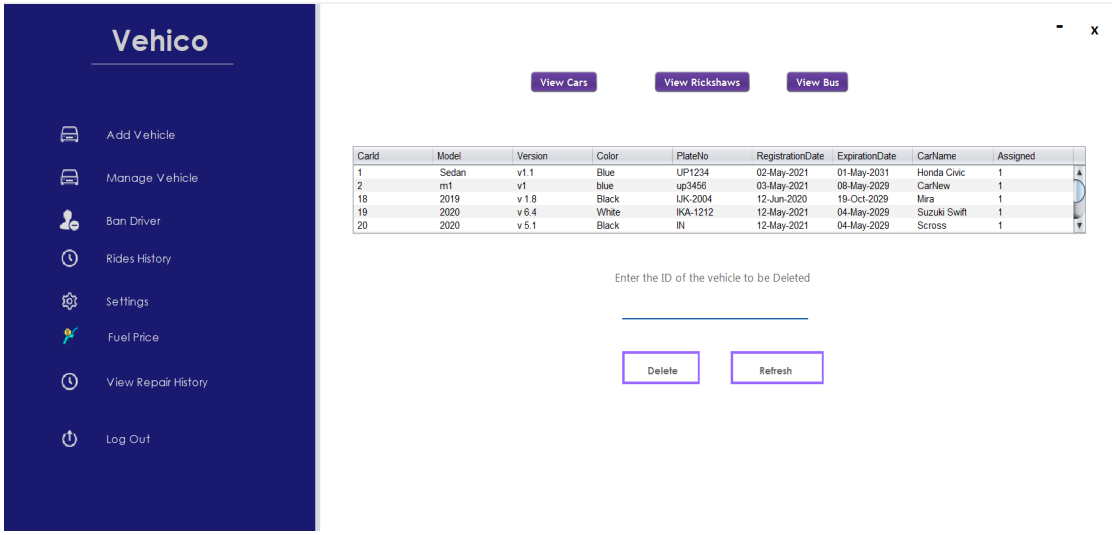


Admin

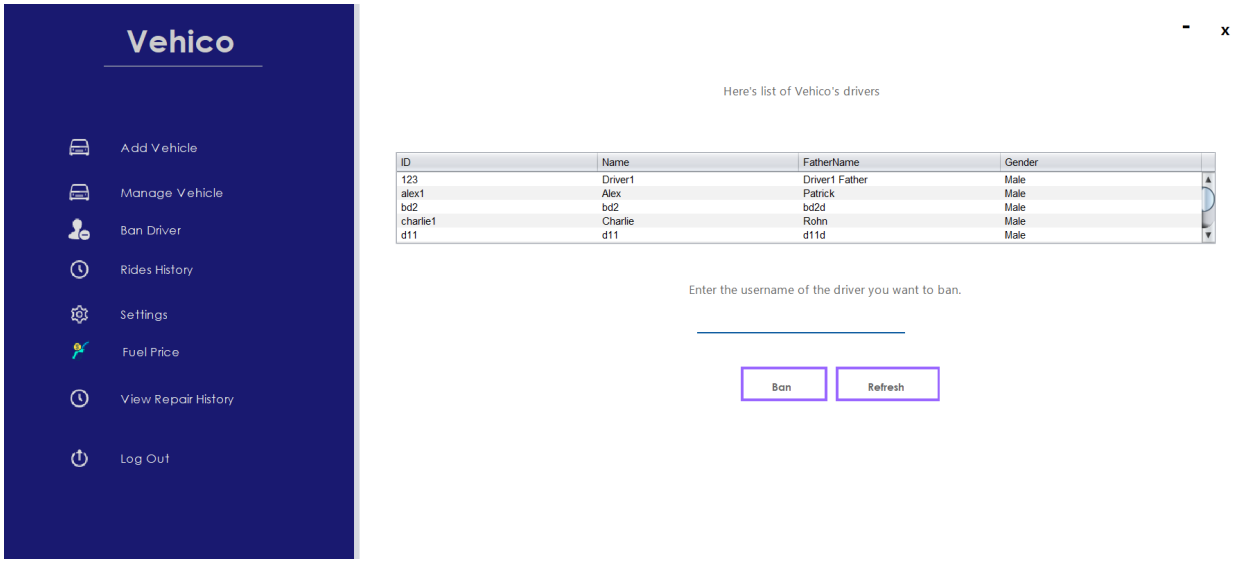
- **Add Vehicle:** On this page admin can add vehicle i.e. car, rickshaw or bus.



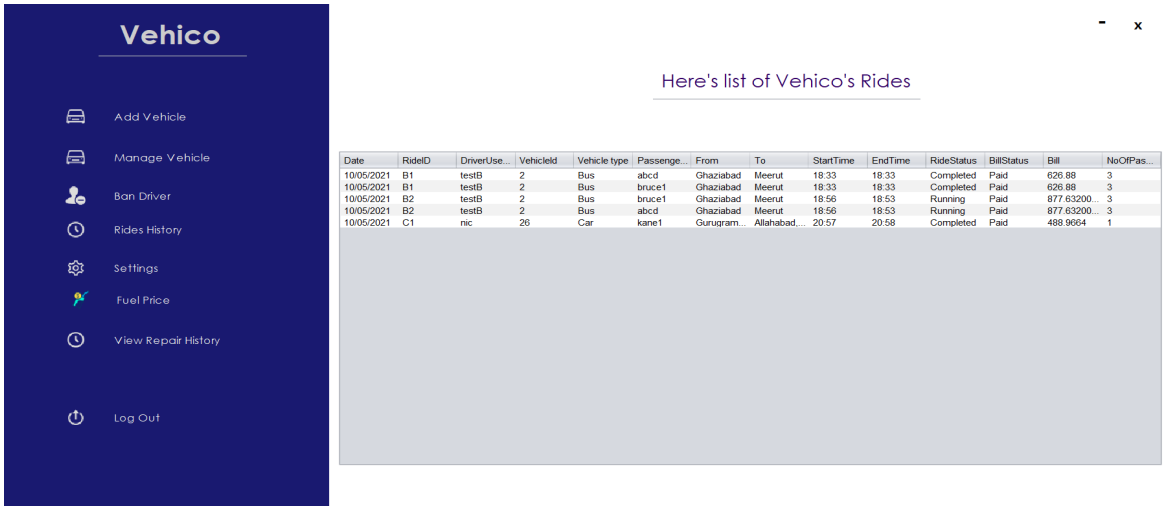
- **Remove Vehicle:** On this page admin can view/delete vehicles from the database. The vehicle could be either a car or a rickshaw or a bus.



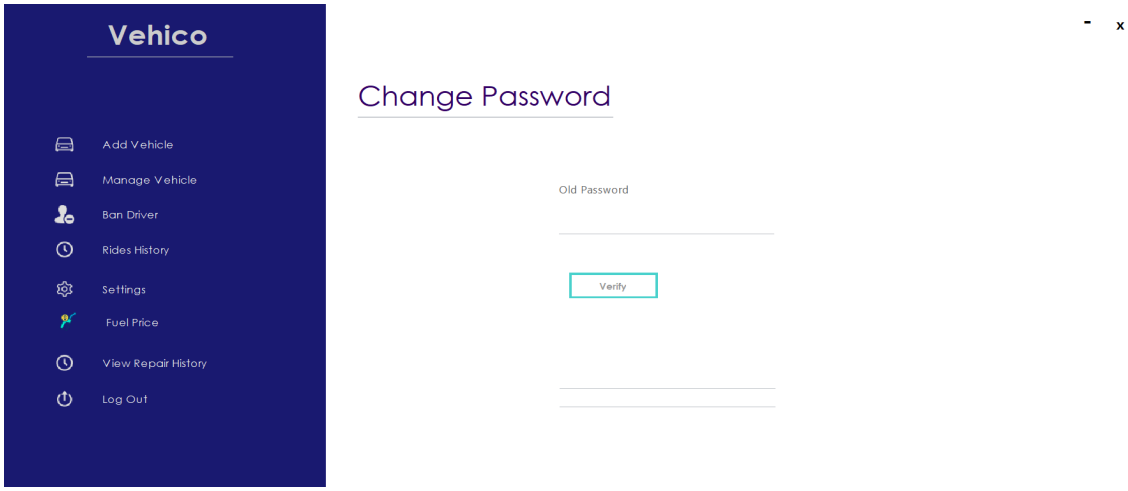
- **Ban Driver:** On this page admin can view all the drivers and can also ban one.



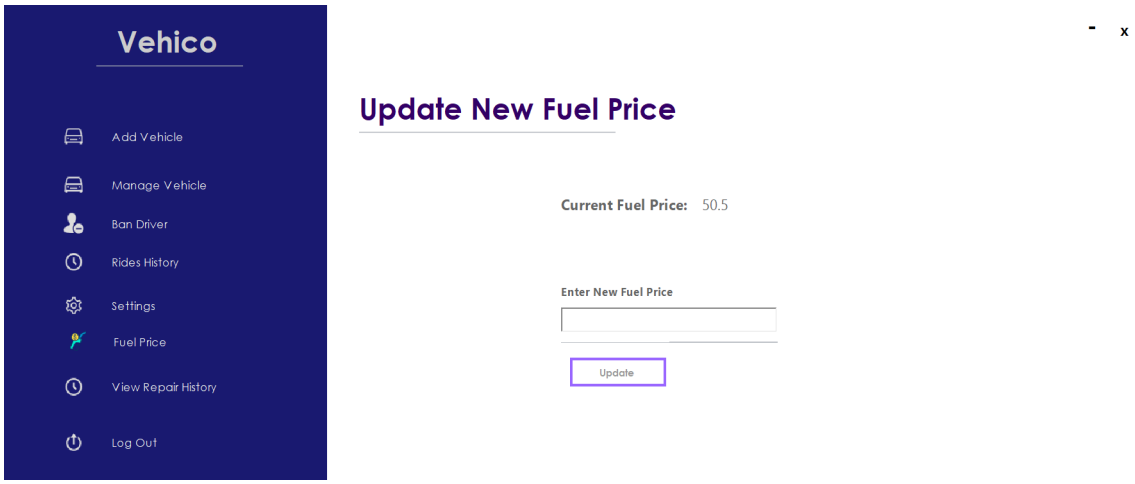
❖ **View Rides:** On this page admin can view all the previous rides.



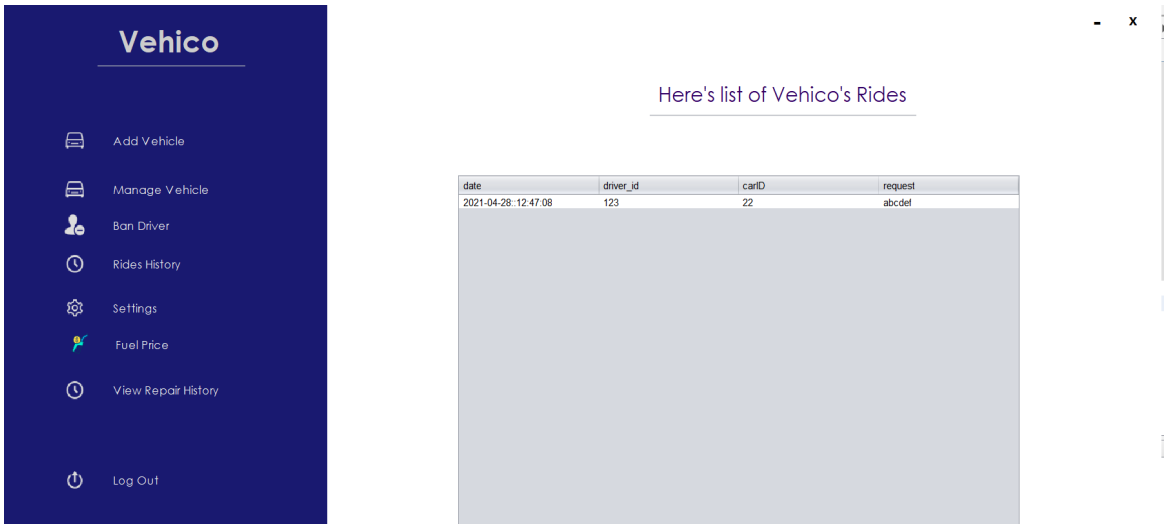
❖ **Settings:** On this page admin can change his password.



❖ **Fuel Price:** On this page admin can change the fuel price

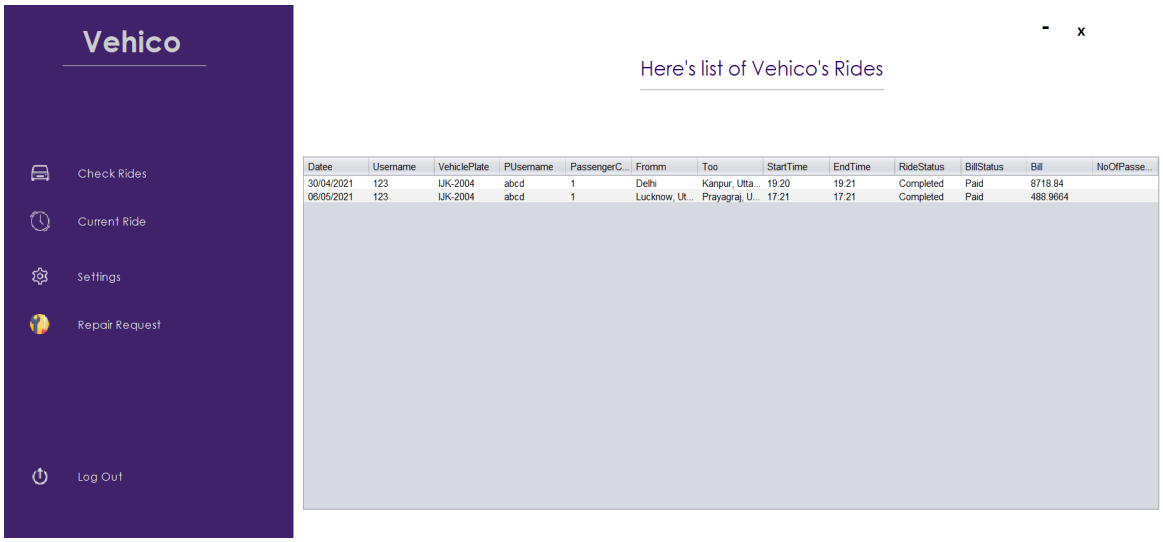


- **View Repair History:** On this page admin can view the repair history.

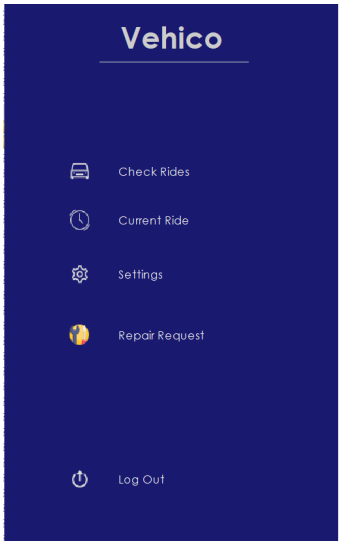


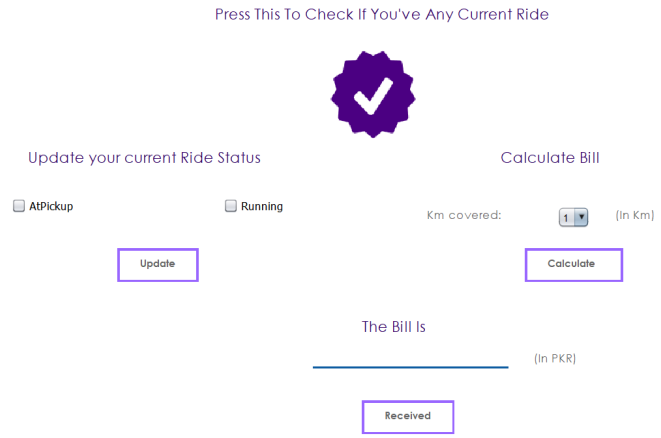
Driver

- **Check Rides:** On this page the driver can view his previous rides.

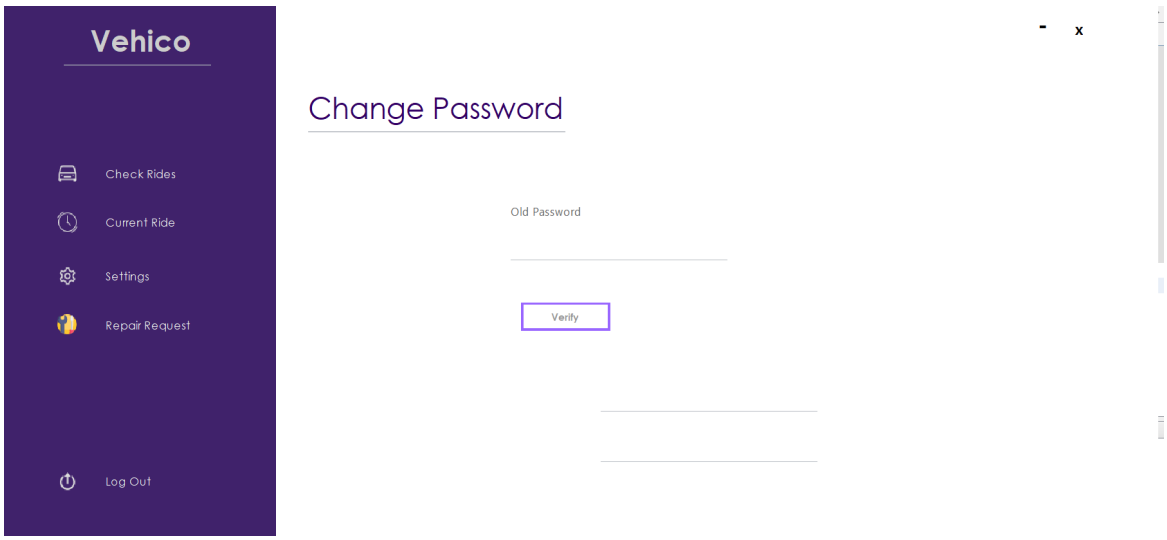


- **Current Ride**: On this page the driver can check if he has any ongoing current ride.If there is an ongoing ride then he can update the status of the ride.After completion of ride he can enter the total distance covered and can generate a bill. On bill payment he can update the status of payment.

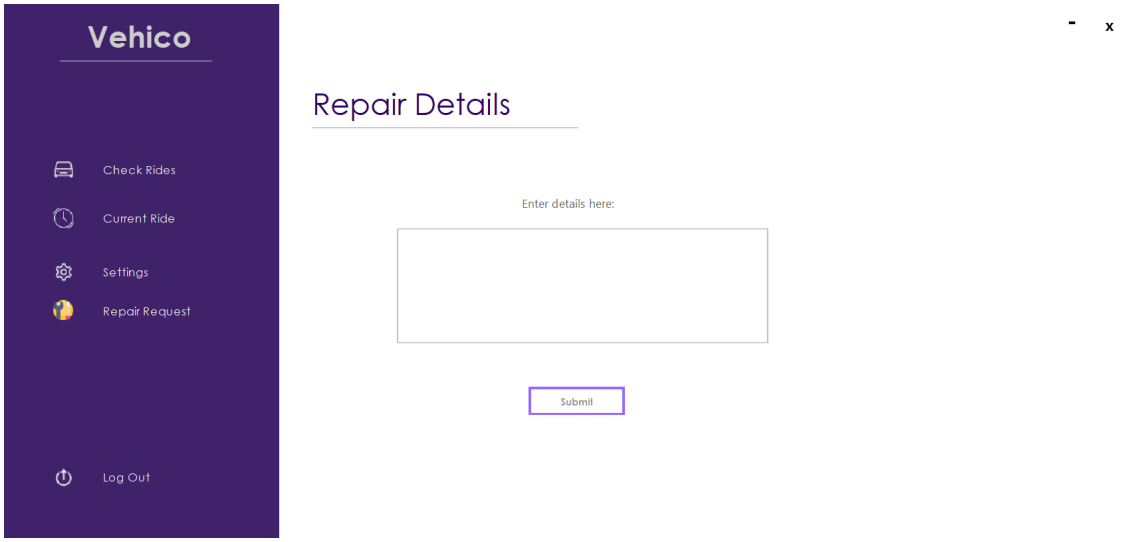




- **Settings:** On this page the driver can change his password.

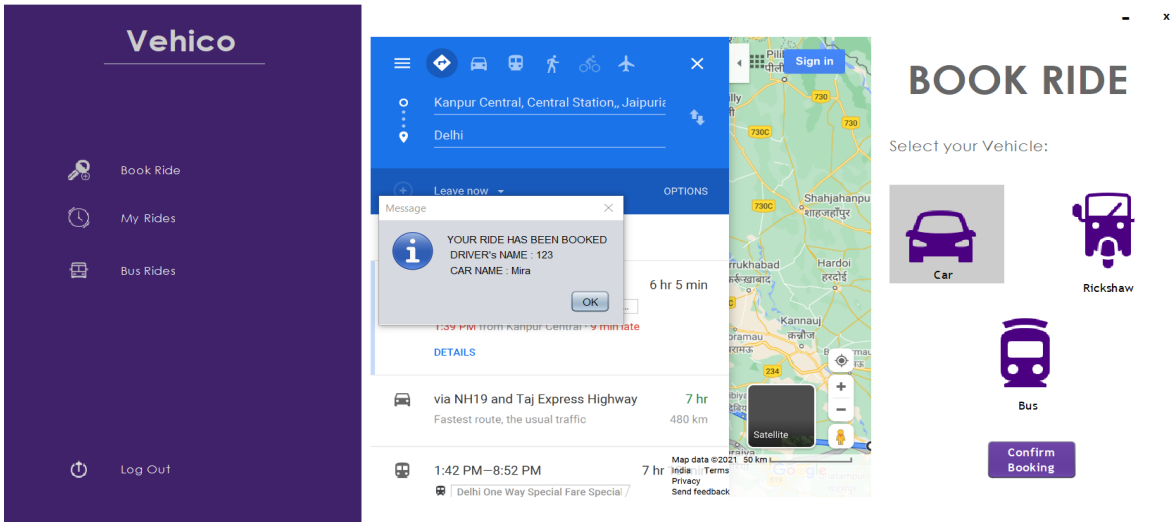


❖ **Repair Request:** On this page the driver can apply for a repair request.

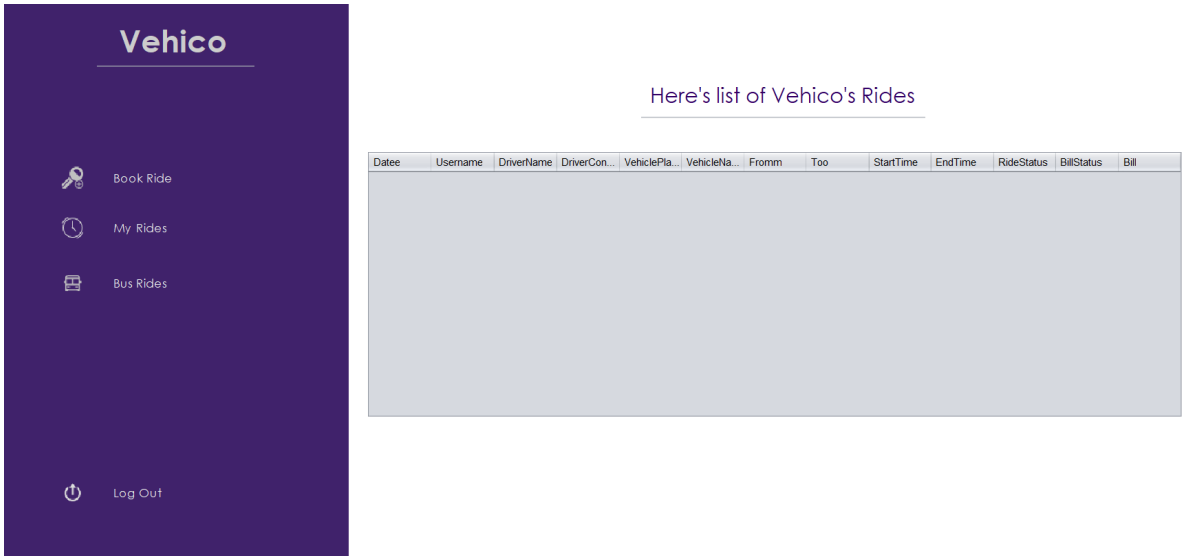


Passenger

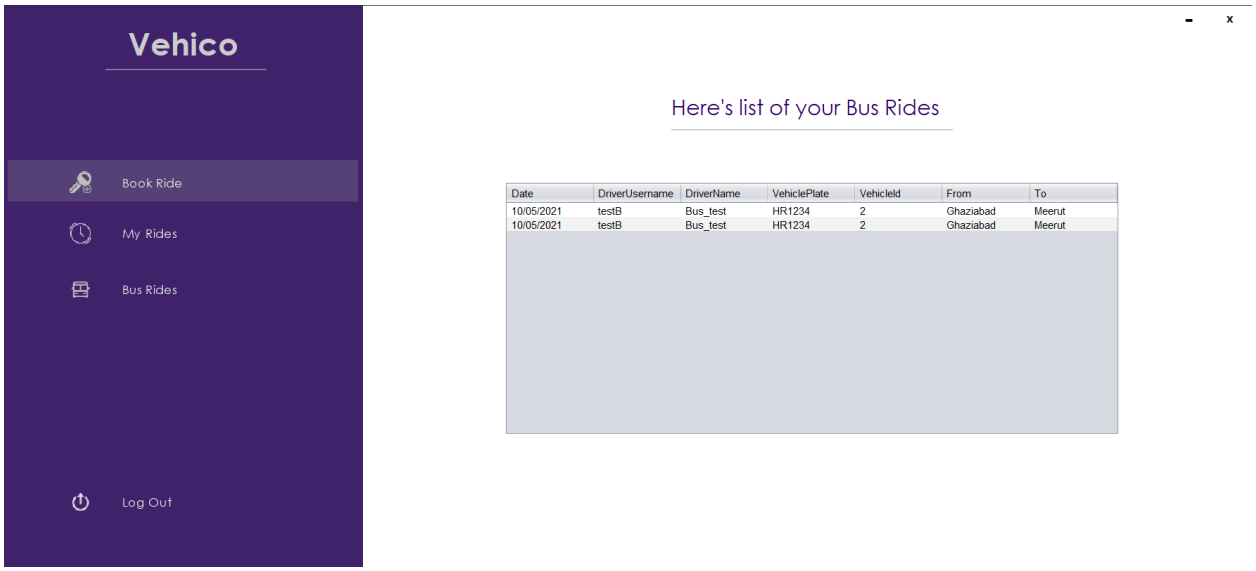
❖ **Book Ride:** On this page the passenger can book a ride by choosing type of vehicle , starting location and final destination.



❖ **My Rides:** On this page Passenger can view all of his previous rides.



❖ **Bus Rides:** On this page the passenger can view all of his previous bus rides.



CONVERTING TABLES IN 3rd NORMAL FORM :

- Since there are no multi-valued attributes in any of the defined tables, *all tables are already in 1NF form*.
- Also, since there is no table where candidate key is composite, *there wouldn't be cases for failing 2NF form*.
- We have a table CarT where we have the information regarding the Car which is added by the Admin. Also, when the driver registers, he/she will only be able to register if the corresponding type of vehicle that they drive has a vacant availability. Now, post successful registration, the details for the driver are stored in the Driver table.
  We also have a table DriverCarT where the driver and the corresponding car details are stored. Initially we had stored CarId, CarName, CarPlateNo (out of which Name & PlateNo can be derived from Id) and from Driver table, DriverUsername & DriverName (name can be derived from username).
  So, DriverUsername & CarId act as Foreign Keys.
  But since DriverUsername can derive all other attributes, it actually becomes a Primary Key.

  DriverUsername -> CarId
  CarId -> CarPlateNo
  DriverUsername -> CarPlateNo

  We can clearly see that there is a case of Transitive Dependency, hence making the table fail in satisfying the conditions for 3rd Normal Form.

  In order to rectify the anomaly, the best option is to keep only the DriverUsername & CarId and retrieve attributes DriverName (from Driver) and CarName, CarPlateNo (from CarT by specifying CarId).

  Hence we now have a table which satisfies the conditions for 3NF.

  Similarly, the same thing was done for the Rickshaw and Bus tables (RickshawT, DriverRickshawT, BusT, DriverBusT,) in the database.

- Let us now look at the other huge tables, one of them being DriverCarRT.

  Similar to the anomalies present in the above stated tables, in DriverCarRT too, we have multiple attribute pairs which can be derived from each other apart from the primary key.
  Earlier in this table we had attributes DriverUsername along with DriverName and similarly CarId along with CarName and CarPlateNo.
  Now, since the candidate key is ID, it will derive all other attributes.

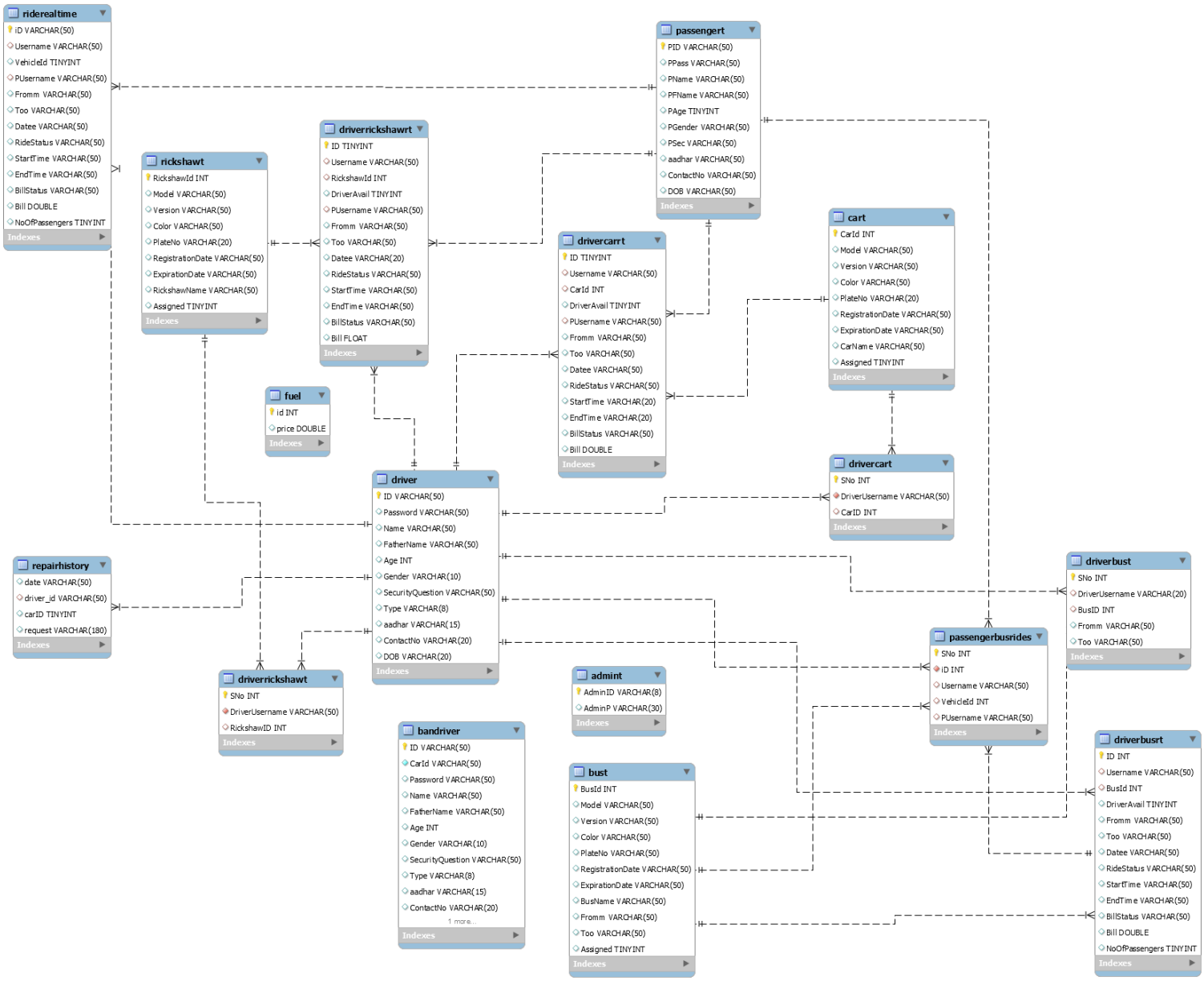
  But,

  ID -> DriverUsername
  DriverUsername -> DriverName
  CarId -> CarPlateNo
  CarId -> CarName

  Therefore in order to remove anomalies, we remove all columns that could be derived from DriverUsername or CarId or any other attribute resulting in failing of 3NF form.
  Similarly, the same is done for the tables DriverBusRT, DriverRickshawRT, RideRealtime, PassengerBusRides.

  Now, it is ensured that there is *no Transitive dependency and all tables are in 3NF*.

E-R DIAGRAM


riderealtime
ID VARCHAR(50)
Username VARCHAR(50)
VehicleId TINYINT
PUsername VARCHAR(50)
Fromm VARCHAR(50)
Too VARCHAR(50)
Datee VARCHAR(50)
RideStatus VARCHAR(50)
StartTime VARCHAR(50)
EndTime VARCHAR(50)
BillStatus VARCHAR(50)
Bill DOUBLE
NoOfPassengers TINYINT
Indexes
passengert
PID VARCHAR(50)
PPass VARCHAR(50)
PName VARCHAR(50)
PFName VARCHAR(50)
PAge TINYINT
PGender VARCHAR(50)
PSec VARCHAR(50)
aadhar VARCHAR(50)
ContactNo VARCHAR(50)
DOB VARCHAR(50)
Indexes
rickshawt
RickshawId INT
Model VARCHAR(50)
Version VARCHAR(50)
Color VARCHAR(50)
PlateNo VARCHAR(20)
RegistrationDate VARCHAR(50)
ExpirationDate VARCHAR(50)
RickshawName VARCHAR(50)
Assigned TINYINT
Indexes
driverrickshawrt
ID TINYINT
Username VARCHAR(50)
RickshawId INT
DriverAvail TINYINT
PUsername VARCHAR(50)
Fromm VARCHAR(50)
Too VARCHAR(50)
Datee VARCHAR(20)
RideStatus VARCHAR(50)
StartTime VARCHAR(50)
EndTime VARCHAR(50)
BillStatus VARCHAR(50)
Bill FLOAT
Indexes
drivercarrt
ID TINYINT
Username VARCHAR(50)
CarId INT
DriverAvail TINYINT
PUsername VARCHAR(50)
Fromm VARCHAR(50)
Too VARCHAR(50)
Datee VARCHAR(50)
RideStatus VARCHAR(50)
StartTime VARCHAR(20)
EndTime VARCHAR(20)
BillStatus VARCHAR(50)
Bill DOUBLE
Indexes
cart
CarId INT
Model VARCHAR(50)
Version VARCHAR(50)
Color VARCHAR(50)
PlateNo VARCHAR(20)
RegistrationDate VARCHAR(50)
ExpirationDate VARCHAR(50)
CarName VARCHAR(50)
Assigned TINYINT
Indexes
fuel
id INT
price DOUBLE
Indexes
drivercart
SNo INT
DriverUsername VARCHAR(50)
CarID INT
Indexes
driver
ID VARCHAR(50)
Password VARCHAR(50)
Name VARCHAR(50)
FatherName VARCHAR(50)
Age INT
Gender VARCHAR(10)
SecurityQuestion VARCHAR(50)
Type VARCHAR(8)
aadhar VARCHAR(15)
ContactNo VARCHAR(20)
DOB VARCHAR(20)
Indexes
repairhistory
date VARCHAR(50)
driver_id VARCHAR(50)
carID TINYINT
request VARCHAR(180)
Indexes
driverbust
SNo INT
DriverUsername VARCHAR(20)
BusID INT
Fromm VARCHAR(50)
Too VARCHAR(50)
Indexes
passengerbusrides
SNo INT
ID INT
Username VARCHAR(50)
VehicleId INT
PUsername VARCHAR(50)
Indexes
driverrickshawt
SNo INT
DriverUsername VARCHAR(50)
RickshawID INT
Indexes
admint
AdminID VARCHAR(8)
AdminP VARCHAR(30)
Indexes
bandriver
ID VARCHAR(50)
CarId VARCHAR(50)
Password VARCHAR(50)
Name VARCHAR(50)
FatherName VARCHAR(50)
Age INT
Gender VARCHAR(10)
SecurityQuestion VARCHAR(50)
Type VARCHAR(8)
aadhar VARCHAR(15)
ContactNo VARCHAR(20)
1 more...
Indexes
bust
BusId INT
Model VARCHAR(50)
Version VARCHAR(50)
Color VARCHAR(50)
PlateNo VARCHAR(50)
RegistrationDate VARCHAR(50)
ExpirationDate VARCHAR(50)
BusName VARCHAR(50)
Fromm VARCHAR(50)
Too VARCHAR(50)
Assigned TINYINT
Indexes
driverbusrt
ID INT
Username VARCHAR(50)
BusId INT
DriverAvail TINYINT
Fromm VARCHAR(50)
Too VARCHAR(50)
Datee VARCHAR(50)
RideStatus VARCHAR(50)
StartTime VARCHAR(50)
EndTime VARCHAR(50)
BillStatus VARCHAR(50)
Bill DOUBLE
NoOfPassengers TINYINT
Indexes

PROCEDURE TO RUN

- ❏ Install Netbeans & import this project.
- ❏ Install JavaFX-SDK-16
- ❏ Add all libraries situated in the link.
- ❏ Add Libraries from the lib folder (inside JavaFX folder) after installing JavaFX-SDK-16.
- ❏ Add *--module-path "**PATH to lib folder in JavaFX-SDK-16**" --add-modules=javafx.controls,javafx.fxml,javafx.web,javafx.media,javafx.swing* in VM options.
- ❏ Open the file **DbConnection.java**, alter URL depending upon your MySQL Server, Username and Password.
- ❏ Database (SQL file) is present in the **/Database** folder.
- ❏ Admin account username: **1** & Admin account password : **y**
- ❏ For Driver & Passenger, register and proceed.

FUTURE SCOPE

The application can be further modified to accommodate other types of vehicles and increased functionality for the users. It can be used to maintain and help with the functioning of larger public transport systems like buses and cabs for interstate travel. The use of this application will allow the work to be digital and rely on less paperwork which would mean more efficient, less time consuming and error free work.

REFERENCES

- ❏ Designing a SWING GUI in Netbeans
- ❏ Connect To MySQL Database
- ❏ Stack OverFlow
- ❏ JavaFX-SDK-16 Webview/JavaFXPanel