**JAVA AWT BASED- REGISTRATION FORM- SQL CONNECTIVITY USING JDBC**

*A*

*Report*

*Submitted in partial fulfilment of the Requirements for the award of the Degree of* **BACHELOR OF ENGINEERING**

IN

## INFORMATION TECHNOLOGY

By

B. Manindra

 1602-19-737-081

**Department of Information Technology**

**Vasavi College of Engineering(Autonomous**

**(Affiliated to Osmania University)**

**Ibrahimbagh, Hyderabad-31**

**BONAFIDE CERTIFICATE**

This is to certify that this project report titled “Automatic Fuel Tracker In Vehicle ” is the bonafide mini project work of Mr.B.Manindra bearing hall ticket number 1602-1-737-081 under the guidance of B. Leelavathy during 4th sem B.E for the academic year 2020-2021.

External Examiner Internal Examiner

B.LEELAVATHY Assistant Professor Department Of Information Technology

**Automatic Fuel Tracker In Vehicles.**

**Abstract:**

Now a days many people are facing problems related to fuel management in vehicles.They are being struck on road due to unknown amount of fuel and getting stopped.Now, here is a solution for that which tracks the fuel in vehicles and gives the notifications.This application tracks the fuel present in vehicle and gives notification to respective user.It also says how much distance will the vehicle move from current location.This gives notifications to the user mobile, time to time and reduces the risk of lacking of fuel.To get access and benefits of this application user must provide information such as login id,password,vehicle type,vehicle number,fuel type and the fixed mileage of the vehicle.By using this application user will be knowing the status offuel and saves more time instead of taking risks.

**Introduction:**

Requirements:

Tables:

* USERS
* VEHICLE
* FUEL\_DETECTOR

Relations:

* USERS\_VEHICLE
* DETECTOR\_VEHICLE

List of attributes with domain types:

Users:

USER\_ID NOT NULL NUMBER(5)

USER\_NAME VARCHAR2(20)

USER\_MOBILE\_NO VARCHAR2(10)

USER\_EMAIL VARCHAR2(20)

**Vehicle:**

VEHICLE\_ID NOT NULL VARCHAR2(10)

MIELAGE NUMBER(5)

VEHICLE\_NAME VARCHAR2(10)

VEHICLE\_TYPE VARCHAR2(10)

**Fuel\_Detector:**

CURRENT\_FUEL NUMBER(5)

CAPACITY NUMBER(5)

TANK\_ID NOT NULL VARCHAR2(10)

FUEL\_TYPE VARCHAR2(10)

**Users\_Vehicle:**

USER\_ID NUMBER(5)

VEHICLE\_ID VARCHAR2(10)

DAY DATE

**detector\_Vehicle:**

TANK\_ID VARCHAR2(10)

VEHICLE\_ID VARCHAR2(10)

**SOFTWARE USED:**

Java Eclipse, Oracle 11g Database, Java SE version 7, SQL\*Plus.

**Eclipse:** It is an integrated development environment (IDE) used in computer programming. It contains a base workspace and an extensible plug in system for customizing the environment. The Eclipse software development kit (SDK), which include java development tools is meant for java developers.

**SQL \*plus:** SQL \*plus is a command line tool proprietary to oracle. You can send SQL Queries to the server using the tool. It can also help you format the result of query. SQL is the query language that is used to communicate with the oracle server to access and modify data.

**Java AWT:**

Java AWT (Abstract Window Toolkit) is  an API to develop GUI or window-based

applications in java.

Java AWT components are platform-dependent i.e. components are displayed according

to the view of operating system. AWT is heavyweight i.e. its components are using the

resources of OS.

The java.awt package provides classes for AWT API such as TextField, Label, TextArea,

RadioButton, CheckBox, Choice, List etc.

**Java Swing:**

Java Swing is a part of Java Foundation Classes (JFC) which was designed for enabling large-scale enterprise development of Java applications. Java Swing is a set of APIs that provides graphical user interface (GUI) for Java programs. Java Swing is also known as Java GUI widget toolkit.

Java Swing or Swing was developed based on earlier APIs called Abstract Windows Toolkit (AWT).  Swing provides richer and more sophisticated GUI components than AWT. The GUI components are ranging from a simple label to complex tree and table. Besides emulating look and feel of various platforms, Swing also provides the pluggable look and feel to allow look and feel of Java programs independent from the underlying platform.

**SQL:**

Structure Query Language(SQL) is a database query language used for storing and

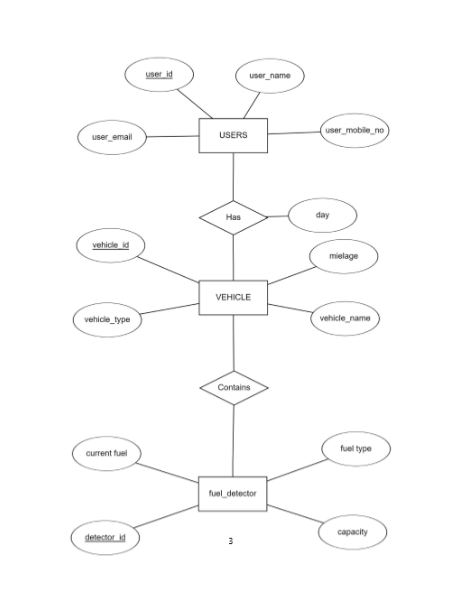
managing data in Relational DBMS. SQL was the first commercial language introduced

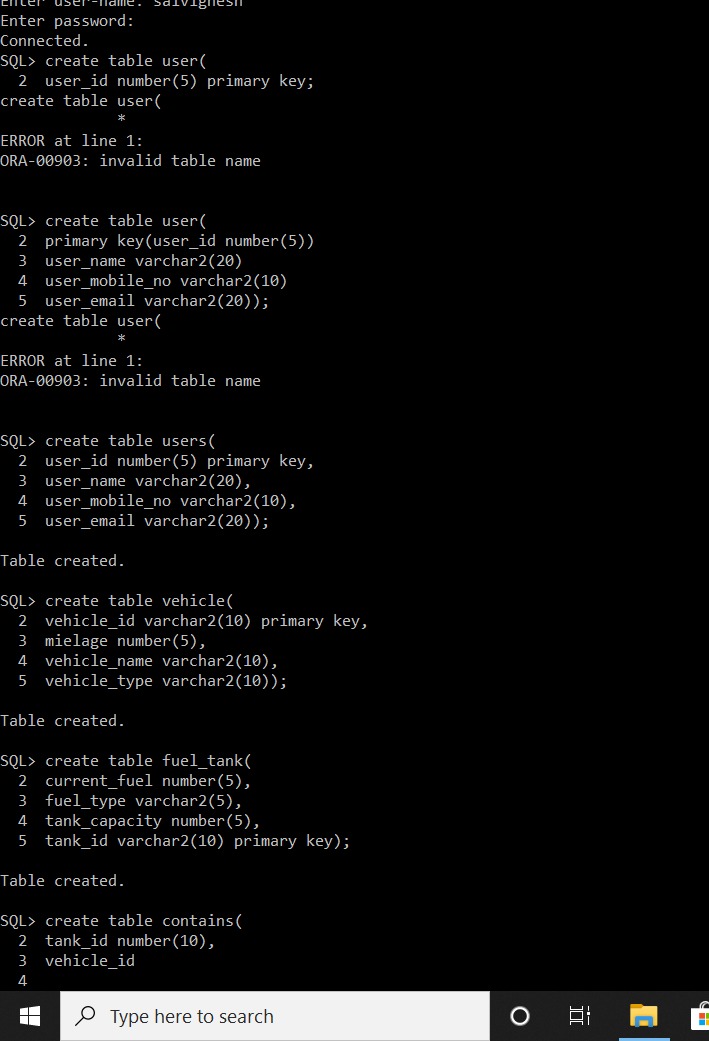
for E.F Codd&#39;s Relational model of database. Today almost all RDBMS (MySql, Oracle,

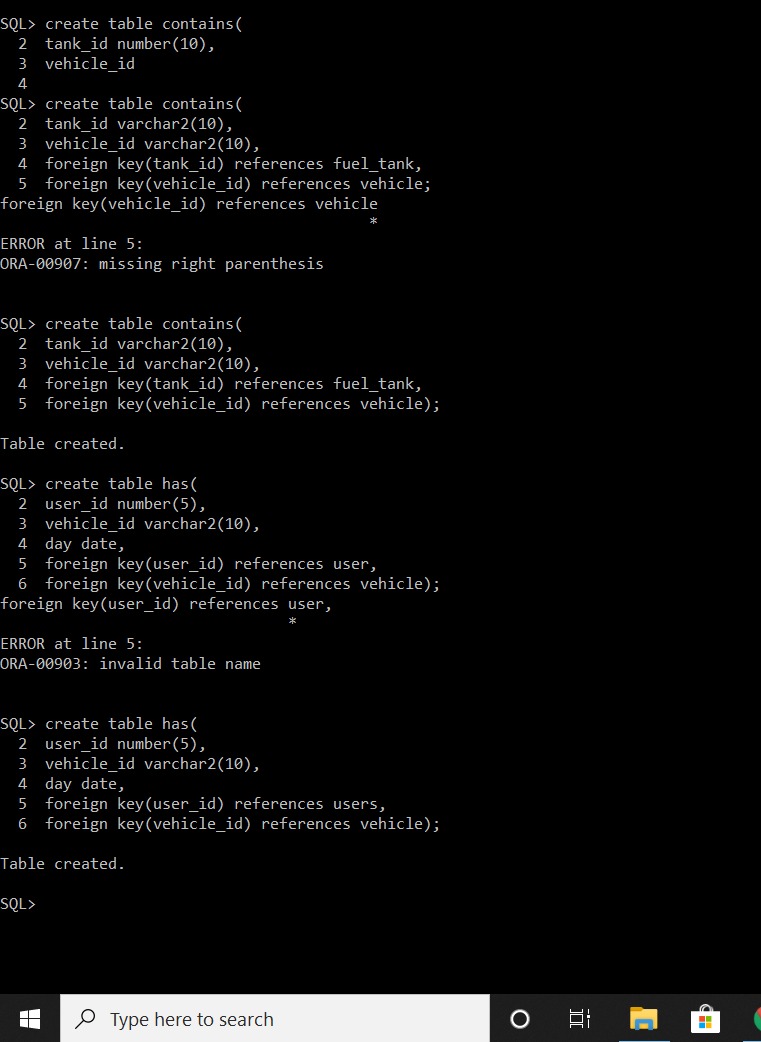
Infomix, Sybase, MS Access) use SQL as the standard database query language. SQL is

used to perform all types of data operations in RDBMS**.**

**ER Diagram:**

****

**DDL COMMANDS: **

****

**Java-SQL Connectivity using JDBC:**

Java Database Connectivity (JDBC) is an application programming interface (API) for

the programming language Java, which defines how a client may access a database. It is a

Java-based data access technology used for Java database connectivity. It is part of

the Java Standard Edition platform, from Oracle Corporation. It provides methods to

query and update data in a database and is oriented towards relational databases.

The connection to the database can be performed using Java programming (JDBC API)

as:

private void connToDb()

{

try

{

Class.forName("oracle.jdbc.driver.OracleDriver");

connection=DriverManger.getConnection("jdbc:oracle:thin:@localhost:1522:xe","manindra","1270");

statement=connection.createStatement();

}

catch(SQLExceptionconnectException)

{

System.out.println(connectException.getMessage());

System.out.println(connectException.getSQLState());

System.out.println(connectException.getErrorCode());

System.exit(1);

}

catch(Exception e)

{

System.err.println("Unable to find and load driver");

System.exit(1);

}

}

Thus, the connection from Java to Oracle database is performed and therefore, can be

used for updating tables in the database directly.

**Program For Main GUI:**

**GUI For Insert In User Table:**

import java.awt.\*;

import java.awt.event.\*;

import java.sql.\*;

public class InsertUser extends Panel

{

Button insertUserButton;

TextFielduser\_idText, user\_nameText, user\_emailText, user\_mobile\_noText;

TextAreaerrorText;

Connection connection;

Statement statement;

public InsertUser()

{

try

{

Class.forName("oracle.jdbc.driver.OracleDriver");

}

catch (Exception e)

{

System.err.println("Unable to find and load driver");

System.exit(1);

}

connectToDB();

}

public void connectToDB()

{

try

{

connection = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","manindra","1270");

statement = connection.createStatement();

}

catch (SQLExceptionconnectException)

{

System.out.println(connectException.getMessage());

System.out.println(connectException.getSQLState());

System.out.println(connectException.getErrorCode());

System.exit(1);

}

}

public void buildGUI()

{

//Handle Insert Account Button

insertUserButton = new Button("Insert User");

insertUserButton.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent e)

{

try

{

Statement statement = connection.createStatement();

//String query = "INSERT INTO sailors (SID,SNAME, RATING, AGE) VALUES (2,'Divya',7,20)";

String query= "INSERT INTO users VALUES(" + user\_idText.getText() + ", " + "'" + user\_nameText.getText() + "'," +user\_emailText.getText() + "," + user\_mobile\_noText.getText() + ")";

int i = statement.executeUpdate(query);

errorText.append("\nInserted " + i + " rows successfully");

}

catch (SQLExceptioninsertException)

{

displaySQLErrors(insertException);

}

}

});

user\_idText = new TextField(15);

user\_nameText = new TextField(15);

user\_emailText = new TextField(25);

user\_mobile\_noText = new TextField(15);

errorText = new TextArea(10, 40);

errorText.setEditable(false);

Panel first = new Panel();

first.setLayout(new GridLayout(4, 2));

first.add(new Label("User ID:"));

first.add(user\_idText);

first.add(new Label("User Name:"));

first.add(user\_nameText);

first.add(new Label("User Email:"));

first.add(user\_emailText);

first.add(new Label("User Mobileno:"));

first.add(user\_mobile\_noText);

first.setBounds(125,90,200,100);

Panel second = new Panel(new GridLayout(4, 1));

second.add(insertUserButton);

second.setBounds(125,220,150,100);

Panel third = new Panel();

third.add(errorText);

third.setBounds(125,320,300,200);

setLayout(null);

add(first);

add(second);

add(third);

setSize(500, 600);

setVisible(true);

}

private void displaySQLErrors(SQLException e)

{

errorText.append("\nSQLException: " + e.getMessage() + "\n");

errorText.append("SQLState: " + e.getSQLState() + "\n");

errorText.append("VendorError: " + e.getErrorCode() + "\n");

}

public static void main(String[] args)

{

InsertUser us = new InsertUser();

us.buildGUI();

}

}

**GUI For Update In Users Table**

import java.awt.\*;

import java.awt.event.\*;

import java.sql.\*;

public class UpdateUser extends Panel

{

Button updateUserButton;

List userIDList;

TextFielduser\_idText, user\_nameText, user\_emailText, user\_mobile\_noText;

TextAreaerrorText;

Connection connection;

Statement statement;

ResultSetrs;

public UpdateUser()

{

try

{

Class.forName("oracle.jdbc.driver.OracleDriver");

}

catch (Exception e)

{

System.err.println("Unable to find and load driver");

System.exit(1);

}

connectToDB();

}

public void connectToDB()

{

try

{

connection = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","manindra”,”1270");

statement = connection.createStatement();

}

catch (SQLExceptionconnectException)

{

System.out.println(connectException.getMessage());

System.out.println(connectException.getSQLState());

System.out.println(connectException.getErrorCode());

System.exit(1);

}

}

private void loadUsers()

{

try

{

rs = statement.executeQuery("SELECT USER\_ID FROM users");

while (rs.next())

{

userIDList.add(rs.getString("USER\_ID"));

}

}

catch (SQLException e)

{

displaySQLErrors(e);

}

}

public void buildGUI()

{

userIDList = new List(10);

loadUsers();

add(userIDList);

//When a list item is selected populate the text fields

userIDList.addItemListener(new ItemListener()

{

public void itemStateChanged(ItemEvent e)

{

try

{

rs = statement.executeQuery("SELECT \* FROM users where USER\_ID ="+userIDList.getSelectedItem());

rs.next();

user\_idText.setText(rs.getString("USER\_ID"));

user\_nameText.setText(rs.getString("USER\_NAME"));

user\_emailText.setText(rs.getString("USER\_EMAIL"));

user\_mobile\_noText.setText(rs.getString("USER\_MOBILE\_NO"));

}

catch (SQLExceptionselectException)

{

displaySQLErrors(selectException);

}

}

});

//Handle Update Sailor Button

updateUserButton = new Button("Update User");

updateUserButton.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent e)

{

try

{

Statement statement = connection.createStatement();

int i = statement.executeUpdate("UPDATE users "

+ "SET user\_name='" + user\_nameText.getText() + "', "

+ "user\_email=" + user\_emailText.getText() + ", "

+ "user\_mobile\_no ="+ user\_mobile\_noText.getText() + " WHERE user\_id = "

+ userIDList.getSelectedItem());

errorText.append("\nUpdated " + i + " rows successfully");

userIDList.removeAll();

loadUsers();

}

catch (SQLExceptioninsertException)

{

displaySQLErrors(insertException);

}

}

});

user\_idText = new TextField(15);

user\_idText.setEditable(false);

user\_nameText = new TextField(15);

user\_emailText = new TextField(15);

user\_mobile\_noText = new TextField(15);

errorText = new TextArea(10, 40);

errorText.setEditable(false);

Panel first = new Panel();

first.setLayout(new GridLayout(4, 2));

first.add(new Label("User ID:"));

first.add(user\_idText);

first.add(new Label("User Name:"));

first.add(user\_nameText);

first.add(new Label("User Email:"));

first.add(user\_emailText);

first.add(new Label("User Mobileno:"));

first.add(user\_mobile\_noText);

Panel second = new Panel(new GridLayout(4, 1));

second.add(updateUserButton);

Panel third = new Panel();

third.add(errorText);

add(first);

add(second);

add(third);

setSize(500, 600);

setLayout(new FlowLayout());

setVisible(true);

}

private void displaySQLErrors(SQLException e)

{

errorText.append("\nSQLException: " + e.getMessage() + "\n");

errorText.append("SQLState: " + e.getSQLState() + "\n");

errorText.append("VendorError: " + e.getErrorCode() + "\n");

}

public static void main(String[] args)

{

UpdateUserupu = new UpdateUser();

upu.buildGUI();

}

}

**GUI For Delete in Users Table:**

import java.awt.\*;

import java.awt.event.\*;

import java.sql.\*;

public class DeleteUser extends Panel

{

Button deleteUserButton;

List userIDList;

TextFielduser\_idText, user\_nameText, user\_emailText, user\_mobile\_noText;

TextAreaerrorText;

Connection connection;

Statement statement;

ResultSetrs;

public DeleteUser()

{

try

{

Class.forName("oracle.jdbc.driver.OracleDriver");

}

catch (Exception e)

{

System.err.println("Unable to find and load driver");

System.exit(1);

}

connectToDB();

}

public void connectToDB()

{

try

{

connection = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","saivignesh","kanamatareddy");

statement = connection.createStatement();

}

catch (SQLExceptionconnectException)

{

System.out.println(connectException.getMessage());

System.out.println(connectException.getSQLState());

System.out.println(connectException.getErrorCode());

System.exit(1);

}

}

private void loadUsers()

{

try

{

rs = statement.executeQuery("SELECT \* FROM users");

while (rs.next())

{

userIDList.add(rs.getString("USER\_ID"));

}

}

catch (SQLException e)

{

displaySQLErrors(e);

}

}

public void buildGUI()

{

userIDList = new List(10);

loadUsers();

add(userIDList);

//When a list item is selected populate the text fields

userIDList.addItemListener(new ItemListener()

{

public void itemStateChanged(ItemEvent e)

{

try

{

rs = statement.executeQuery("SELECT \* FROM users");

while (rs.next())

{

if (rs.getString("USER\_ID").equals(userIDList.getSelectedItem()))

break;

}

if (!rs.isAfterLast())

{

user\_idText.setText(rs.getString("USER\_ID"));

user\_nameText.setText(rs.getString("USER\_NAME"));

user\_emailText.setText(rs.getString("USER\_EMAIL"));

user\_mobile\_noText.setText(rs.getString("USER\_MOBILE\_NO"));

}

}

catch (SQLExceptionselectException)

{

displaySQLErrors(selectException);

}

}

});

//Handle Delete Sailor Button

deleteUserButton = new Button("Delete User");

deleteUserButton.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent e)

{

try

{

Statement statement = connection.createStatement();

int i = statement.executeUpdate("DELETE FROM users WHERE USER\_ID = "

+ userIDList.getSelectedItem());

errorText.append("\nDeleted " + i + " rows successfully");

user\_idText.setText(null);

user\_nameText.setText(null);

user\_emailText.setText(null);

user\_mobile\_noText.setText(null);

userIDList.removeAll();

loadUsers();

}

catch (SQLExceptioninsertException)

{

displaySQLErrors(insertException);

}

}

});

user\_idText = new TextField(15);

user\_nameText = new TextField(15);

user\_emailText = new TextField(15);

user\_mobile\_noText = new TextField(15);

errorText = new TextArea(10, 40);

errorText.setEditable(false);

Panel first = new Panel();

first.setLayout(new GridLayout(4, 2));

first.add(new Label("User ID:"));

first.add(user\_idText);

first.add(new Label("User Name:"));

first.add(user\_nameText);

first.add(new Label("User Email:"));

first.add(user\_emailText);

first.add(new Label("User Mobileno:"));

first.add(user\_mobile\_noText);

Panel second = new Panel(new GridLayout(4, 1));

second.add(deleteUserButton);

Panel third = new Panel();

third.add(errorText);

add(first);

add(second);

add(third);

setSize(450, 600);

setLayout(new FlowLayout());

setVisible(true);

}

private void displaySQLErrors(SQLException e)

{

errorText.append("\nSQLException: " + e.getMessage() + "\n");

errorText.append("SQLState: " + e.getSQLState() + "\n");

errorText.append("VendorError: " + e.getErrorCode() + "\n");

}

public static void main(String[] args)

{

DeleteUserdelu = new DeleteUser();

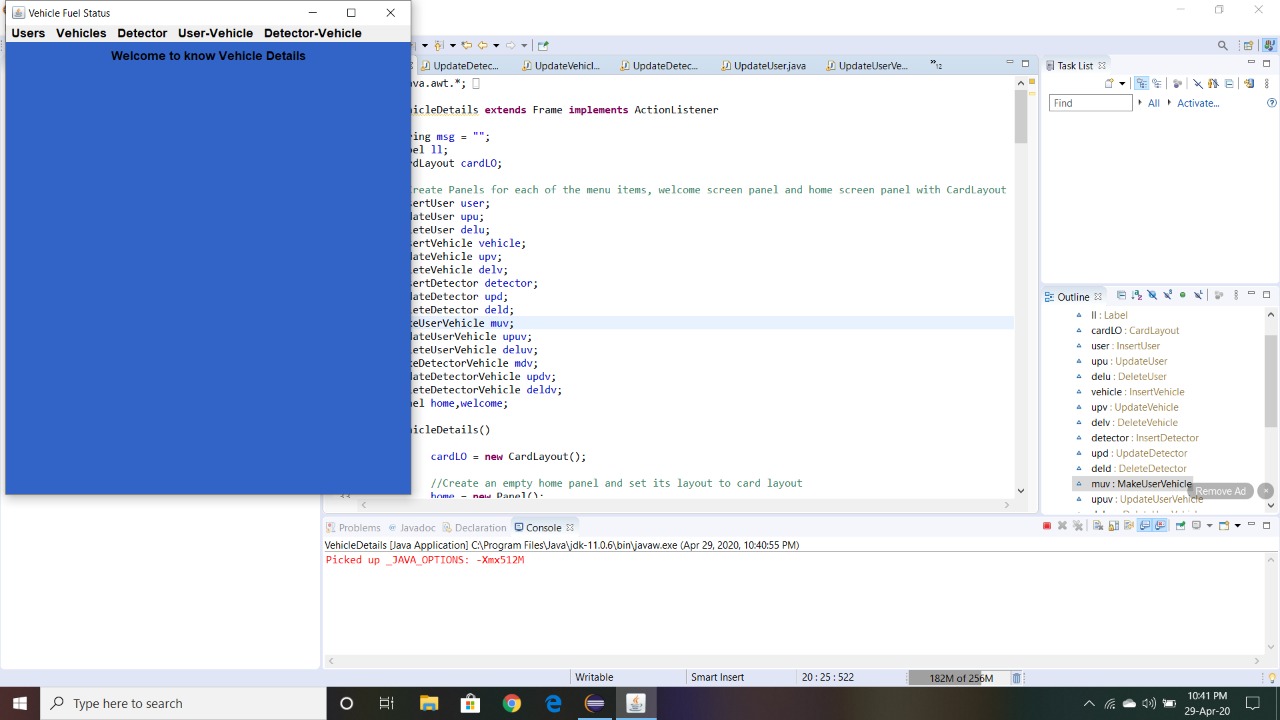
delu.buildGUI();

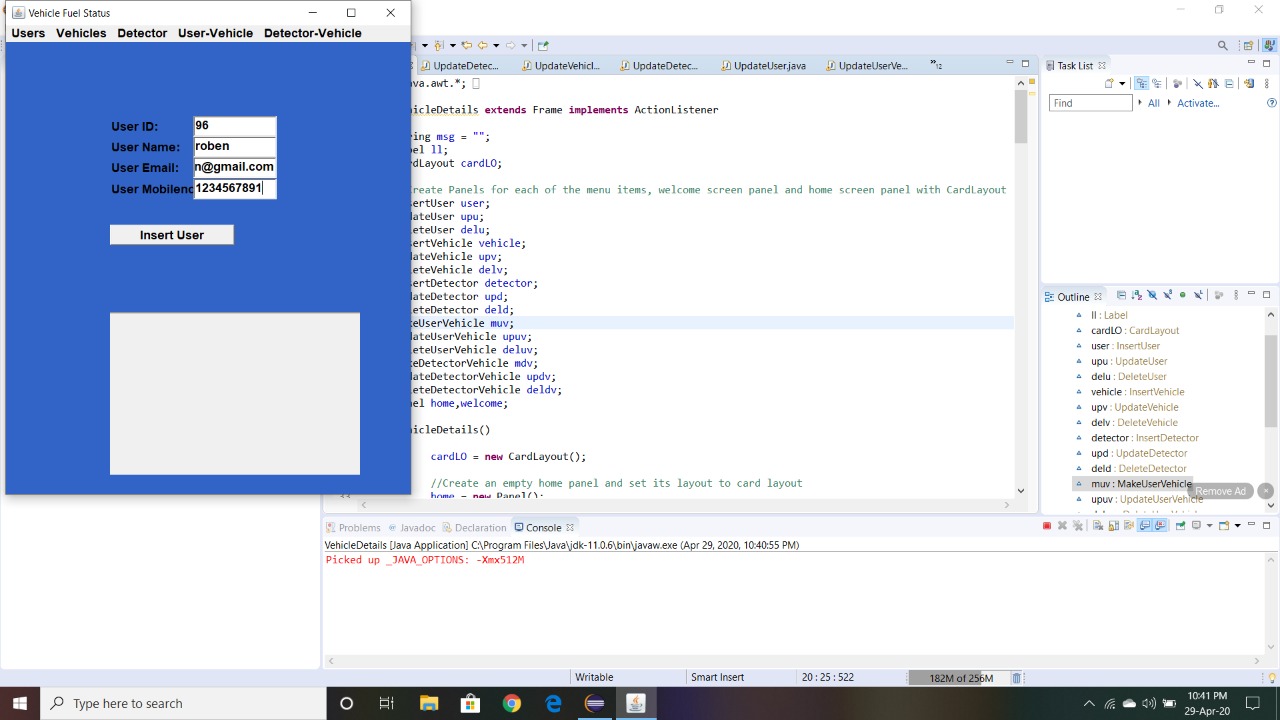
}

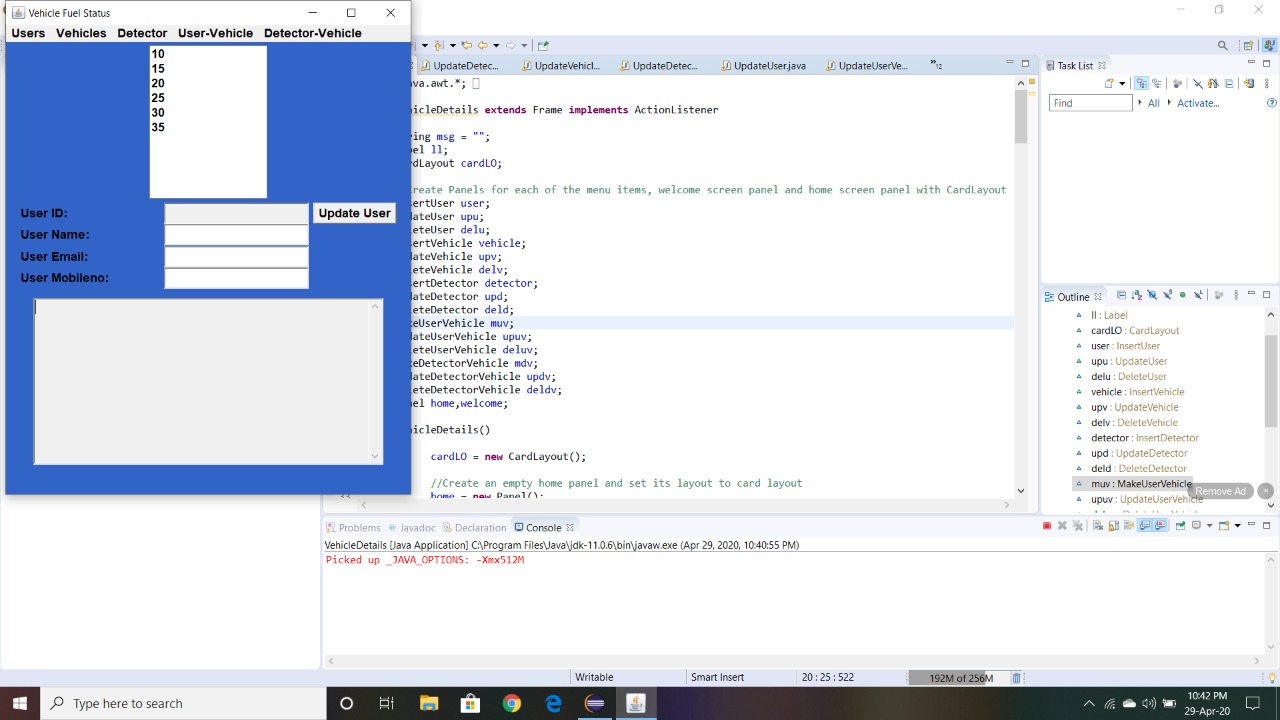
}

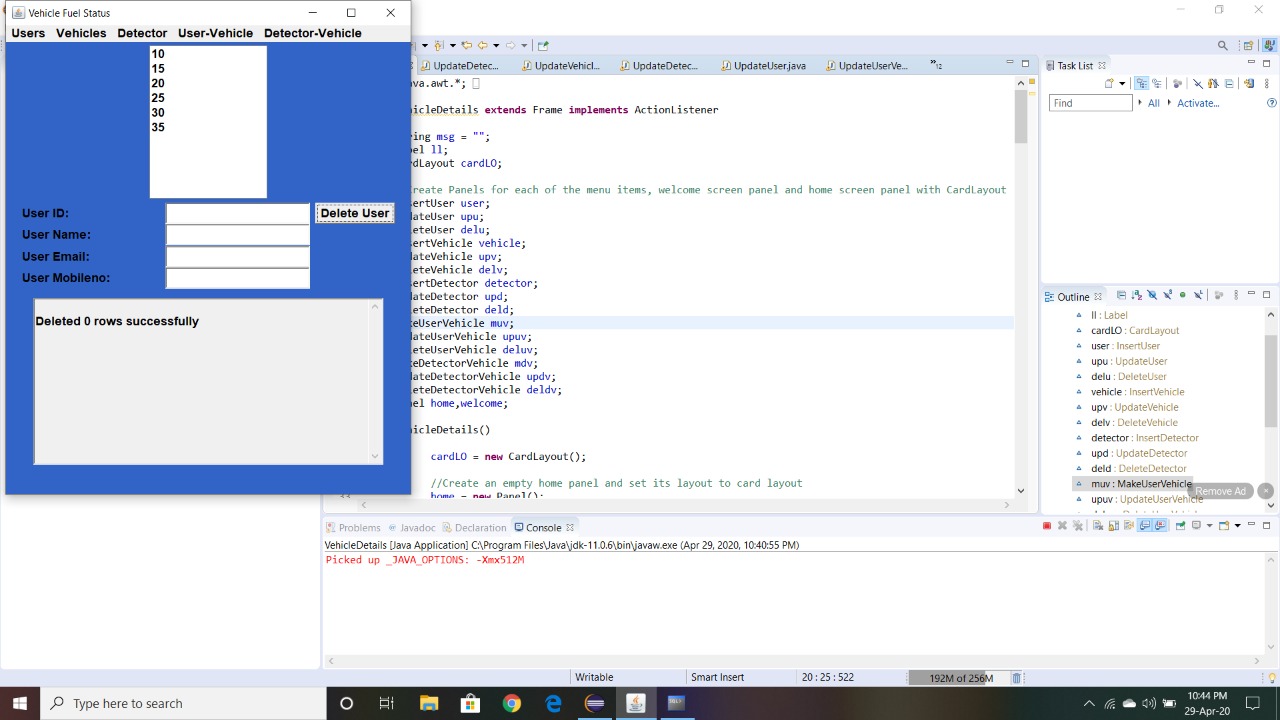
**OUTPUT :**

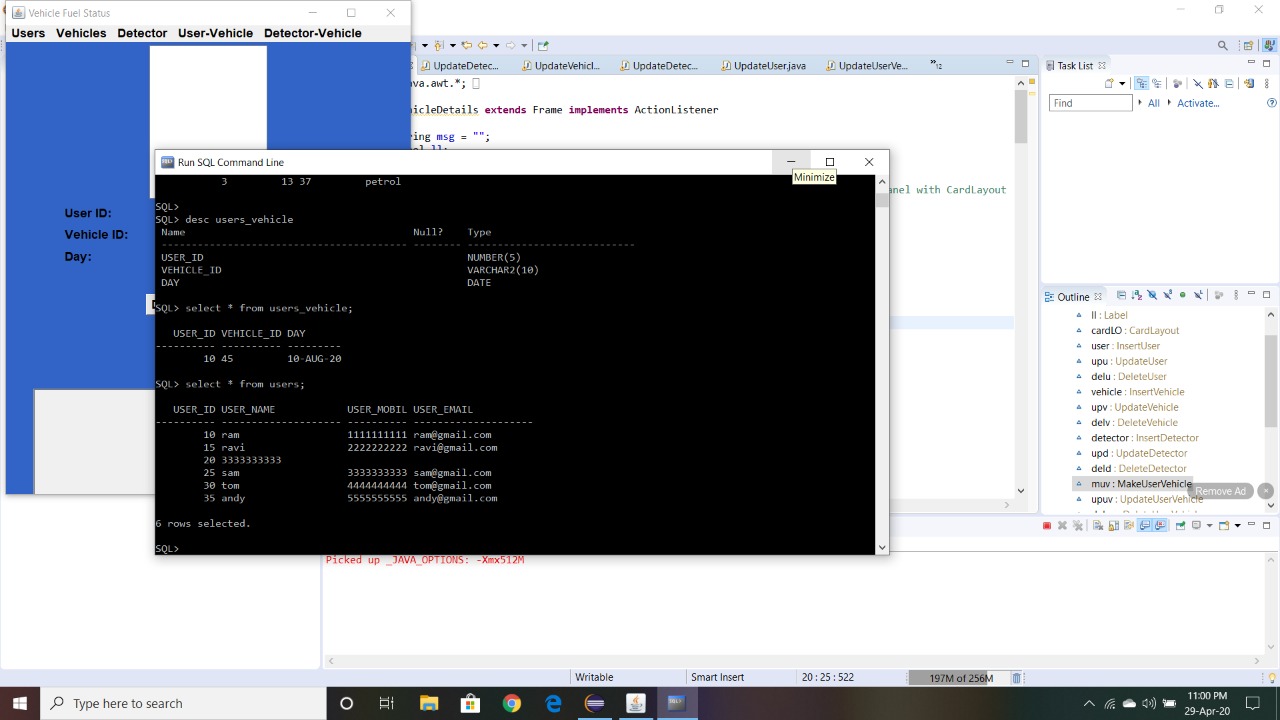
**GUI For Main Frame:**

****

****

****

****

****

**Discussion and Future work**

While doing this project I got new ideas I understood how to work on projects. Now to further extend this project I want to create a android app by which I can control my project on my hand and connect to it. this project efficiently stores the data in tables and we can manipulate it easily by friendly userinterface.

**References:**

<https://www.decodejava.com/what-is-jdbc.htm>

<https://docs.oracle.com/javase/8/docs/api/>

<https://www.tutorialspoint.com/swing/index.htm>

**GITHUB LINK :**

**https://github.com/TECHBUDDY-1270/1602-19-737-081-DBMS-Report**