**MINI-PROJECT REPORT**

**<Car Details GUI Connected with MySQL Database>**

**BY**

**DANIEL ADHIHTYA (2262213)**

B. Tech in Computer Science (Data Science)

1. **Introduction**

The Car Details Management GUI is a Java application connected with a MySQL database. It allows users to input and manage details of cars, such as type, seating capacity, fuel type, transmission, and price range. This report outlines the design, implementation, and functionality of the system.

1. **Design**

The GUI is designed using Java Swing components to provide a user-friendly interface. It consists of text fields for entering car details and buttons for adding cars to the database and displaying existing car information. The layout follows a simple and intuitive design, making it easy for users to interact with the system.

1. **Implementation (Code)**

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit this template

\*/

package carrentalgui;

/\*\*

\*

\* @author ksbas

\*/

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.\*;

public class CarRentalGUI extends JFrame implements ActionListener {

private Connection connection;

private JTextField carTypeField, seaterField, fuelTypeField, transmissionField, priceRangeField;

private JButton addButton, displayButton;

private JTextArea displayArea;

public CarRentalGUI() {

try {

// Establishing database connection

connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/car\_rental", "root", "21200000");

} catch (SQLException e) {

e.printStackTrace();

JOptionPane.showMessageDialog(null, "Failed to connect to database.");

}

// Setting up the GUI components

setTitle("Car Rental Management System");

setSize(500, 400);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JPanel inputPanel = new JPanel(new GridLayout(6, 2));

inputPanel.add(new JLabel("Car Type:"));

carTypeField = new JTextField();

inputPanel.add(carTypeField);

inputPanel.add(new JLabel("Seater:"));

seaterField = new JTextField();

inputPanel.add(seaterField);

inputPanel.add(new JLabel("Fuel Type:"));

fuelTypeField = new JTextField();

inputPanel.add(fuelTypeField);

inputPanel.add(new JLabel("Transmission:"));

transmissionField = new JTextField();

inputPanel.add(transmissionField);

inputPanel.add(new JLabel("Price Range:"));

priceRangeField = new JTextField();

inputPanel.add(priceRangeField);

addButton = new JButton("Add Car");

addButton.addActionListener(this);

displayButton = new JButton("Display Cars");

displayButton.addActionListener(this);

JPanel buttonPanel = new JPanel();

buttonPanel.add(addButton);

buttonPanel.add(displayButton);

displayArea = new JTextArea(10, 40);

displayArea.setEditable(false);

JScrollPane scrollPane = new JScrollPane(displayArea);

getContentPane().add(inputPanel, BorderLayout.NORTH);

getContentPane().add(buttonPanel, BorderLayout.CENTER);

getContentPane().add(scrollPane, BorderLayout.SOUTH);

}

@Override

public void actionPerformed(ActionEvent e) {

if (e.getSource() == addButton) {

addCar();

} else if (e.getSource() == displayButton) {

displayCars();

}

}

private void addCar() {

try {

String carType = carTypeField.getText();

String seater = seaterField.getText();

String fuelType = fuelTypeField.getText();

String transmission = transmissionField.getText();

String priceRange = priceRangeField.getText();

PreparedStatement statement = connection.prepareStatement("INSERT INTO cars1 (car\_type, seater, fuel\_type, transmission, price\_range) VALUES (?, ?, ?, ?, ?)");

statement.setString(1, carType);

statement.setString(2, seater);

statement.setString(3, fuelType);

statement.setString(4, transmission);

statement.setString(5, priceRange);

int rowsAffected = statement.executeUpdate();

if (rowsAffected > 0) {

JOptionPane.showMessageDialog(null, "Car added successfully.");

} else {

JOptionPane.showMessageDialog(null, "Failed to add car.");

}

} catch (SQLException ex) {

ex.printStackTrace();

JOptionPane.showMessageDialog(null, "Error: " + ex.getMessage());

}

}

private void displayCars() {

try {

Statement statement = connection.createStatement();

ResultSet resultSet = statement.executeQuery("SELECT \* FROM cars1");

StringBuilder displayText = new StringBuilder();

while (resultSet.next()) {

displayText.append("Car Type: ").append(resultSet.getString("car\_type")).append(", ");

displayText.append("Seater: ").append(resultSet.getString("seater")).append(", ");

displayText.append("Fuel Type: ").append(resultSet.getString("fuel\_type")).append(", ");

displayText.append("Transmission: ").append(resultSet.getString("transmission")).append(", ");

displayText.append("Price Range: ").append(resultSet.getString("price\_range")).append("\n");

}

displayArea.setText(displayText.toString());

} catch (SQLException e) {

e.printStackTrace();

JOptionPane.showMessageDialog(null, "Error: " + e.getMessage());

}

}

public static void main(String[] args) {

SwingUtilities.invokeLater(() -> new CarRentalGUI().setVisible(true));

}

}

1. **Results**

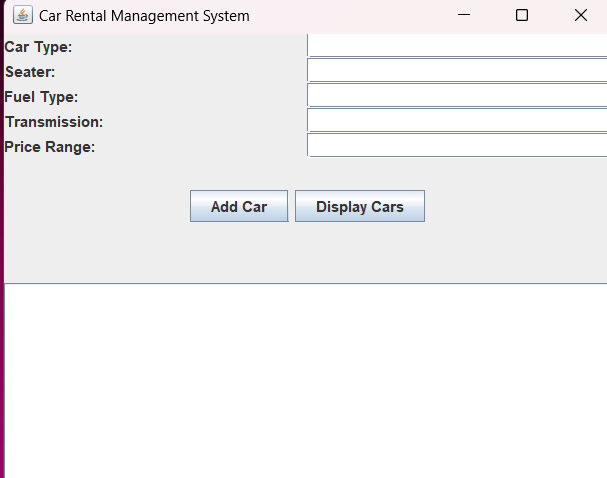
****

Fig1:Graphical User Interface

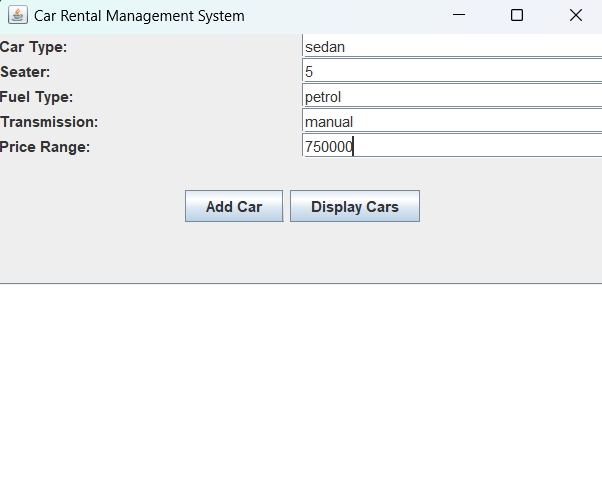
****

Fig2:Entering Car Details

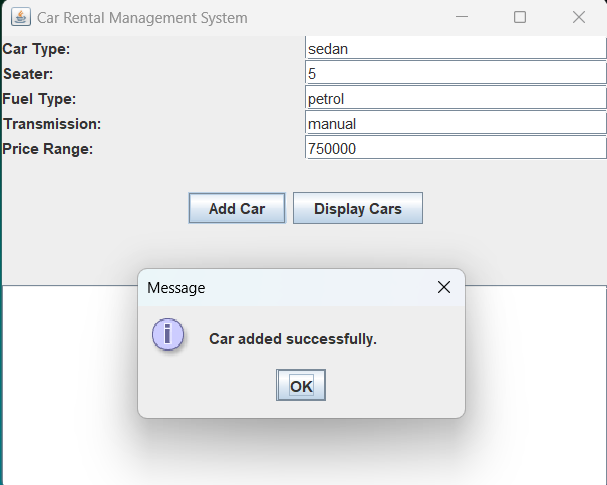
****

Fig3:Adding Car Details By Using ‘Add Car’ Button

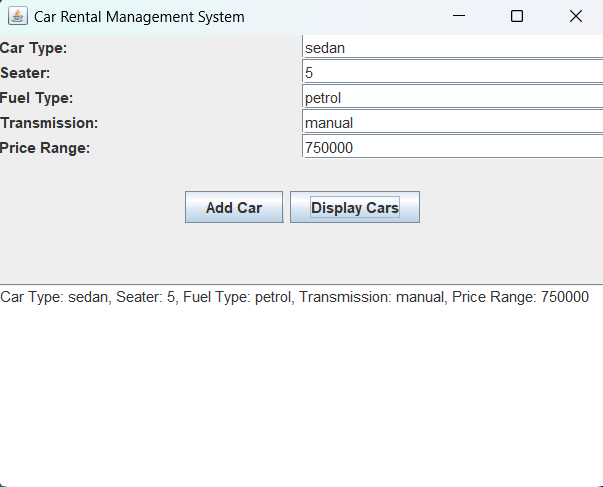
****

Fig4:Displaying Car Details By Using ‘Display Cars’Button

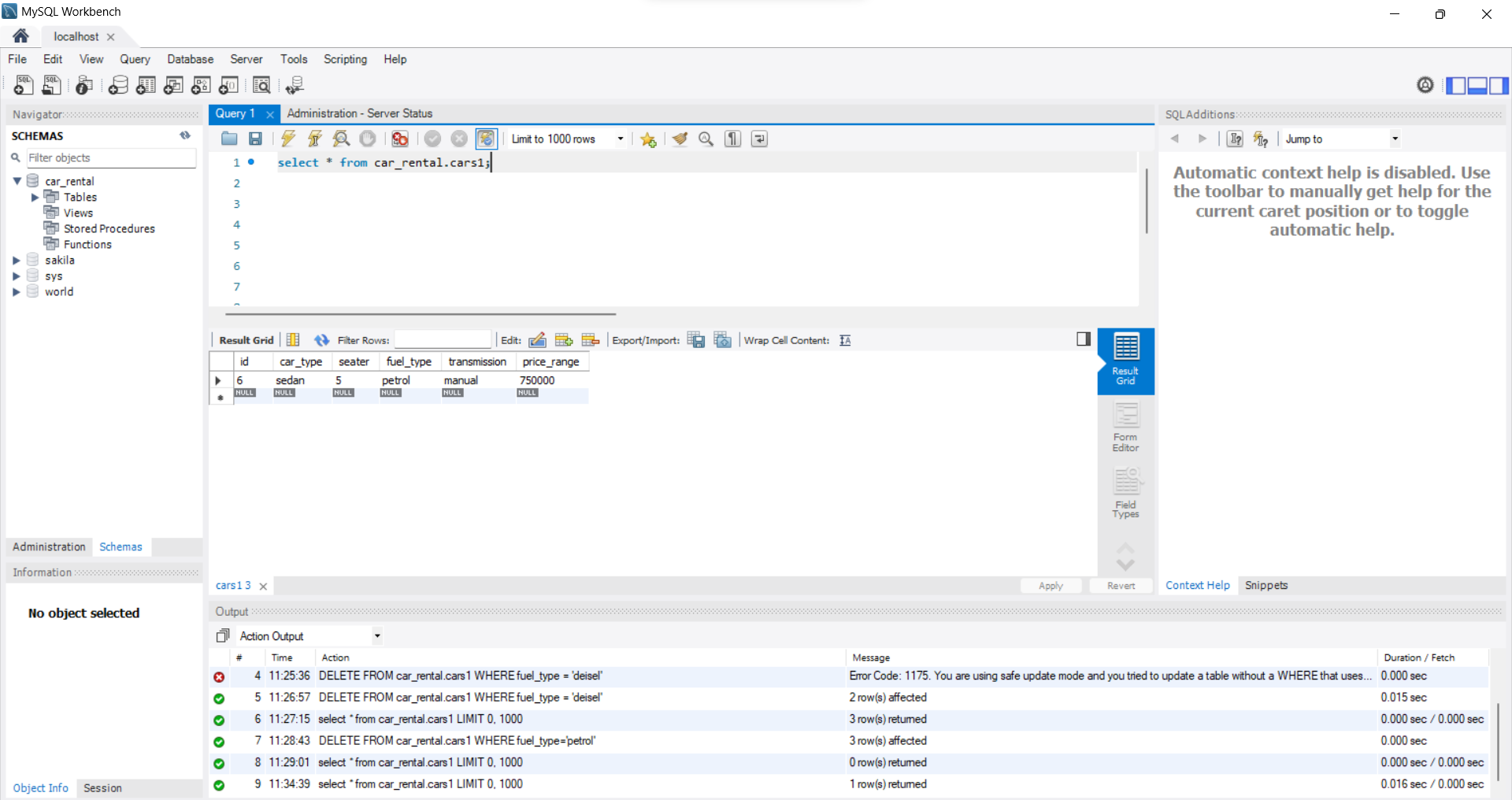
****

Fig5:MySQL WorkBench where Data Stored in MySQL DataBase

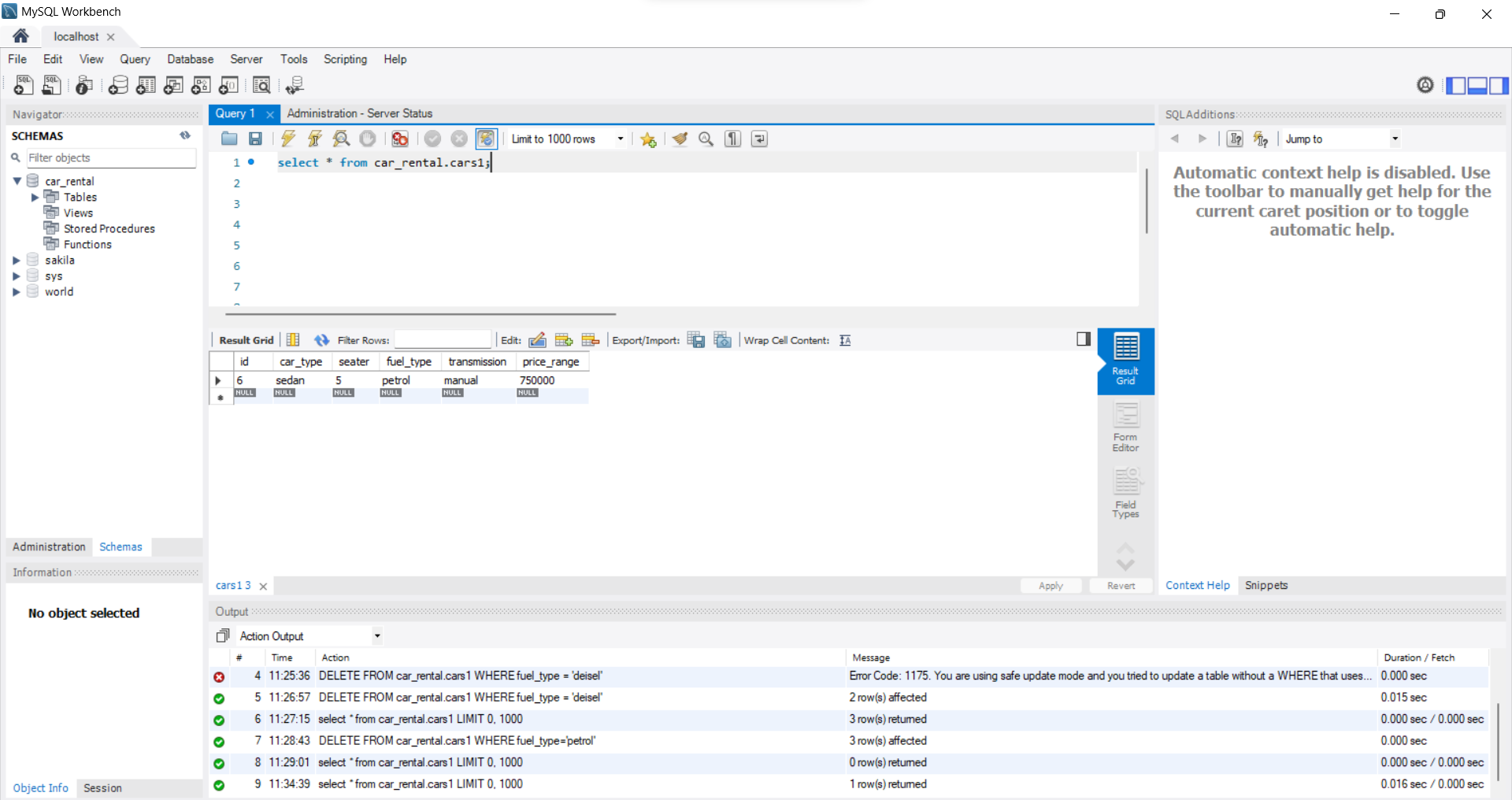
****

Fig6:Using SQL Queries to display the content of the database

1. **Conclusion**

The Car Details Management GUI effectively integrates Java Swing components with a MySQL database to facilitate the management of car information. Users can add new cars to the database and retrieve existing car details with ease. The system demonstrates the seamless interaction between the frontend GUI and backend database, providing a robust solution for car details management.