Sri Lanka Institute of Information Technology

Vehicle Repair Management System

Project Report

Information Technology Project 2021

Project ID: ITP2021_S1_02_G7

Submitted by:

1. IT19968216 -Balasooriya D.P.K.D
2. IT19961590-Dilmika B.G.N
3. IT1966618-Siriwardana H.D.T.H
4. IT19966922-Wimukthi H.R.Y
5. IT19958620-Wickramasinghe T.L
6. IT19960364-Madushanka G.T
7. IT19962580-Sansala M.G.N
8. IT19970332-Priyal N.C.I

Submitted Date :

30.05.20

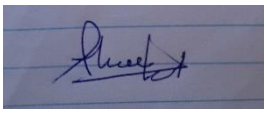
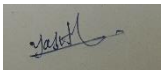
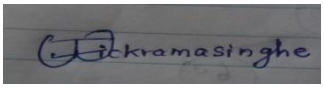
Declaration

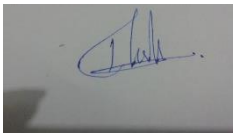
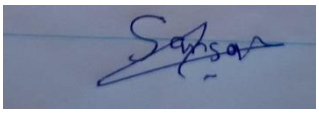
We declare that this project report or part of it was not a copy of a document done by any organization, university any other institute or a previous student project group at SLIIT and was not copied from the Internet or other sources.

Project Details

Project Title	RASA vehicle repair service management system
Project ID	ITP2021_S1_02_G7

Group Members

Reg. No	Name	Signature
IT19968216	Balasooriya D.P.K.D	
IT19961590	Dilmika B.G.N	
IT19966618	Siriwardhane H.D.T.H	
IT19966922	Wimukthi H.R.Y	
IT19958620	Wickramasinghe T.L	

IT19960364	Madushanka G.T	
IT19962580	Sansala M.G.N	
It19970332	Priyal N.C.I	

Abstract

The vehicle repair company "RASA" is based in Malabe, Sri Lanka. The company's job is to repair vehicles that come in to repair with or without an accident. The company does it for the customer when a vehicle arrives. The company runs and maintains its network manually and holds records to document the specifics of its inventory, service details, client data, and account details and also employee details, they still have no computerized program as it poses several problems such as data incoherence, data unreliability, and so on. The approach is a repetitive and inefficient way and could have resulted in a serious loss of data. As a client requirement, we proposed a vehicle repair management system for "RASA" company. Our team chose a set of features to build for this system after discussing it with the client and agreed to create a java-based web application as a response to the existing problems confronting the company. Our team has agreed to use Java as their programming languages and MySQL as a database tool for backend and HTML, CSS, JS, Sementic UI as frontend tools.

Acknowledgement

We express our sincere thanks firstly to our lecturers who stood as pillars behind our team and guide us to success. Our team acknowledge the indebtedness to our teaching staff who encourage us with their valuable guidance and kind supervision throughout every task of this project.

Special thanks for Ms. Janani Tharmaseelan, Ms. Archchana, Ms. Uthpala Samarakoonfor given their useful time to explain our mistakes and educate about those errors.

We pay our deep sense of gratitude to the company that offered us this opportunity to build thisproject. Our team express our heartiest thanks to the company staff who supported us with providing relevant information that are crucial to build this process.

Lastly, we are thankful to all those who helped us in any way to make this project a successful outcome.

Table of Contents

Declaration ..... **ii**

Abstract ..... **iii**

Table of Contents ..... **v**

List of Figures ..... **vi**

List of Tables ..... **ix**

List of Acronyms and Abbreviations ..... **x**

1. Introduction **1**

1.1 Problem Statement 1

1.2 Product Scope 2

1.3 Project Report Structure 2

2. Methodology **3**

2.1 Requirements and Analysis 3

2.2 Design 19

2.3 Implementations 42

2.4 Testing 70

3. Conclusion **67**

4. References **88**

[1] Lecture 5 - Final Report. **88**

[2] IEEE Citation Guidelines. **88**

List of Figures

Figure 1:Logo-RASA 1

Figure 2: UD – employee details management 4

Figure 3: UD – employee payments management 5

Figure 4: UD –customer details management 6

Figure 5: UD –work progress management 7

Figure 6: UD –financial management 8

Figure 7: UD –car records management 9

Figure 8: UD –car rental management 10

Figure 9: AD – employee details management 11

Figure 10: AD – Employee payment management (attendance management) 12

Figure 11: AD – Employee payment management (salary management) 13

Figure 12: AD – Employee payment management (advance management) 14

Figure 13: AD--customer details management 15

Figure 14: AD-- work progress management 16

Figure 15: AD financial management (payment management) 17

Figure 16: AD financial management (Budget management) 18

Figure 17: AD car records management 19

Figure 18: AD car rental management 20

Figure 19: High level architecture diagram 21

Figure 20: Class Diagram 22

Figure 21: Entity Relationship diagram 23

Figure 22: customer registration 24

Figure 23: vehicle registration 24

Figure 24: add service 25

Figure 25: search service entry 25

Figure 26: service entry page 26

Figure 27: work progress management System 27

Figure 28: add new service 27

Figure 29: assign employee 28

Figure 30: add repair components 28

Figure 31: estimate report overview 29
Figure 32: financial management 29
Figure 33: Latest payment 30
Figure 34: all payment 31
Figure 35: Budget UI 31
Figure 36: car record retrieve 32
Figure 37: car record register 33
Figure 38,39,40:car record edit 34
Figure 41:available cars 35
Figure 42:Rent form 36
Figure 43:Rent Details 37
Figure 44:update rental details 38
Figure 45:employee registration 39
Figure 46:employee details 40
Figure 47:employee update form 41
Figure 48: employee profile 42
Figure 49: employee salary and attendance management 43
Figure 50: latest advance record management 43
Figure 51:Add Advance 44
Figure 52: customer class 45
Figure 53: Employee class 45
Figure 54 Employee advance Class: 46
Figure 55:repair class 46
Figure 56: work progress class 47
Figure 57:Vehcile class 47
Figure 58 payment class: 48
Figure 59:Budget class 48
Figure 60 :client service 49
Figure 61:employee service 49
Figure 62: Employee Advance service 50
Figure 63:workprogress service 50
Figure 64:Vehicle service 51
Figure 65:service entry service 51
Figure 66:budget service 52
Figure 67:payment service 52

Figure 68 add customer: 53

Figure 69:payement servlet 53

Figure 70:add service entry 54

Figure 71:create work progress 54

Figure 72:upload vehicle 55

Figure 73: get service details 55

Figure 74 Get budget amount : 56

Figure 75: Update payment 56

Figure 76:add payment 57

Figure 77:display latest payment 57

Figure 78:car record model 58

Figure 79:car record service 58

Figure 80:car record delete servlet 59

Figure 81:car record insert servlet 60

Figure 82:car record retrieve servlet 60

Figure 83:car record update edit servlet 61

Figure 84:car record update servlet 62

Figure 85:car record delete servlet 62

List of Tables

Table 1: Testing- Add service 65

List of Acronyms and Abbreviations

ER_____Entity Relationship
CD_____Class diagram

AD_____Activiti Diagram
UI_____User Interface

1. Introduction

1.1 Problem Statement

RASA is a car repair company that has been well known in the Kaduwela area for many years. It does all kinds of repair work. As a company, RASA wants to manage its employees' information and payments, inventory information such as vehicles coming into repair and their owners, distribution information, and auto parts. Also, you need to manage the registrations and payments related to the rental car service provided by the company. Finally, the profit should be compared with the income and expenses of the company and records should be kept for all the above. After discussing all these issues, our team came up with solutions to the problems faced by the company and a system that can provide them.

Figure 1:Logo

1.2 Product Scope

Web base application was developed by us for event planning company to deal with vehicle repairs, manage inner company works. This scope covers the accompanying capacities through online application. Such as Finance Management System, Customer Management System, Employee Management System, Service Management System, Inventory Management system, Work Progress Management System, Rent Management System. This scope gives an easy to use system andeffective information preparing.

1.3 Project Report Structure

First Section:

As the first section it mainly focused on client's problems which they have while using manual system and client's requirements. Therefore, in this part gave the solutions for that problem andinclude the overall project plan and the scope of the project by us.

Second Section:

In this section focused on methodology part which contain the clarification of Requirement Analysis, Designs, Implementation, and the Testing. Use case Diagrams and the Activity Diagrams are used for describing Requirement Analysis part. The Design part is classified utilizing the guide of the ER diagram, Class Diagram, High-Level Architecture Diagram and thesnapshots of user Interfaces. For the Implementation part there are Snapshots of the Implementations and finally there are test cases for the test areas.

Third Section:

This section indicates the information of evaluation and accomplishment of the whole project and the new knowledge which are learned through this project.

Forth Section:

This section indicates all the references which used to full fill this project.

Methodology

1.4 Requirements and Analysis

The Objective of the process is to convert a manual vehicle repair processes into an automated system. After a thorough analysis these manual process was divide into eight main functions. Which included Customer Details management, Work progress management, Financial management, Employee Details Management, Employee Payment Management, Car Records Management, Car Rental Management and Inventory Management. In previously this company use paper based system to handle documents. In this company documents are really important to handle. So, with the implementation of the system, the documents are electronically stored and can be downloaded as PDF. Furthermore, previously all the calculations were done by manually. But from this newly created system payment calculation and estimate amount calculations also automated. The aims of this system are to reduce the time that consume for paper works, increase accuracy of the calculations and handle files perfectly.

Monthly analyzed reports can be downloaded from the system which can be used in the decision-making process of the system. Which gives a chance to the company to increase profit and performance of it.

1.1 Functional Requirement

Reliability - Consisted of good performance and able to trust the content. The system is developed involving the latest technologies.

Efficiency - The highest or the peak level of performance when using the least amount of inputs in order to generate the highest output.

The system will handle all the input data without any problems. So that every task that done using this system is very fast. So user can gain huge efficiency from this system.

Accuracy - The correctness or the precision of the stored or produced data in the system. Accuracy of the data is very important in this system because this company always deals with sensitive data of the customer and also with lots of calculations. So inputs data need to be correct. If it is not all the calculations become wrong. Because of that in this system all the input data will be validating in entry level as well as database level.

Ease of use - Easy to handle and understand the concepts of a process. All the process of this system is break down to small processes and implement. So then any user can interact with this system very easily. Every part of this system can be handle by any user who has basic knowledge of handling a computer.

Usability - How easy it is to use and handle the system for the client can be defined as usability. In the implemented system a side navigational bar consists of easy keywords with images to access any page. All the details are displayed and described in a simple form which will be easy to understand to any user.

1.1.1 Use Case Diagrams

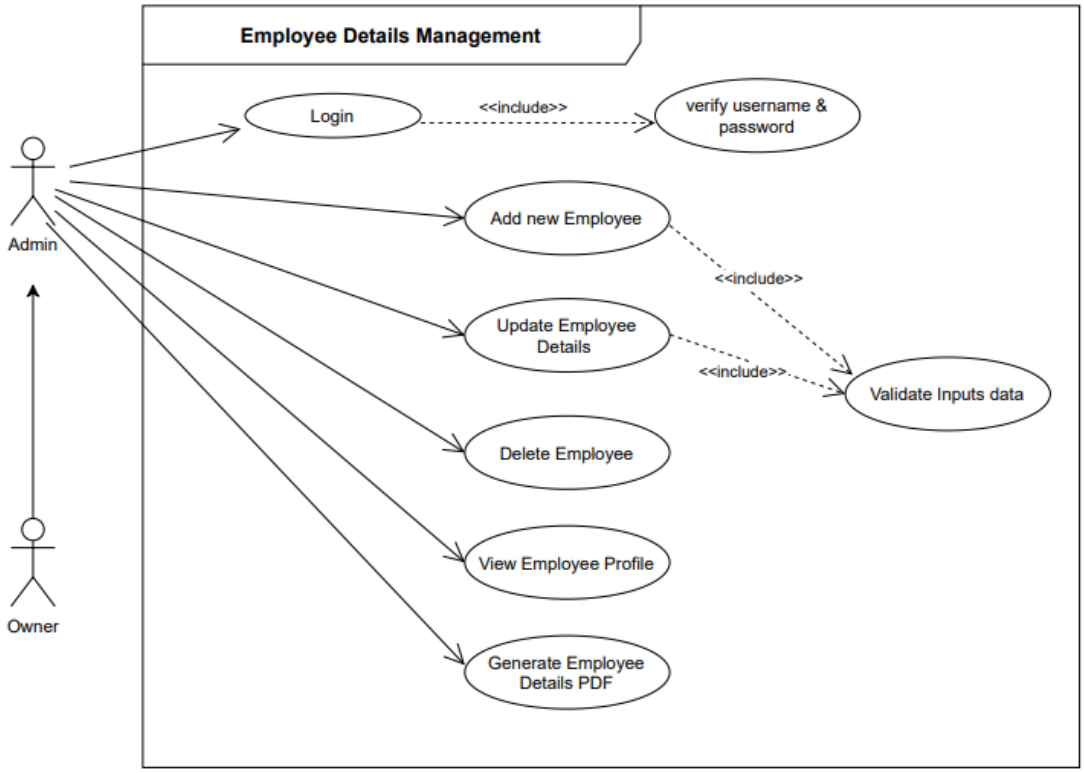


Figure 2: UD – employee details management

\

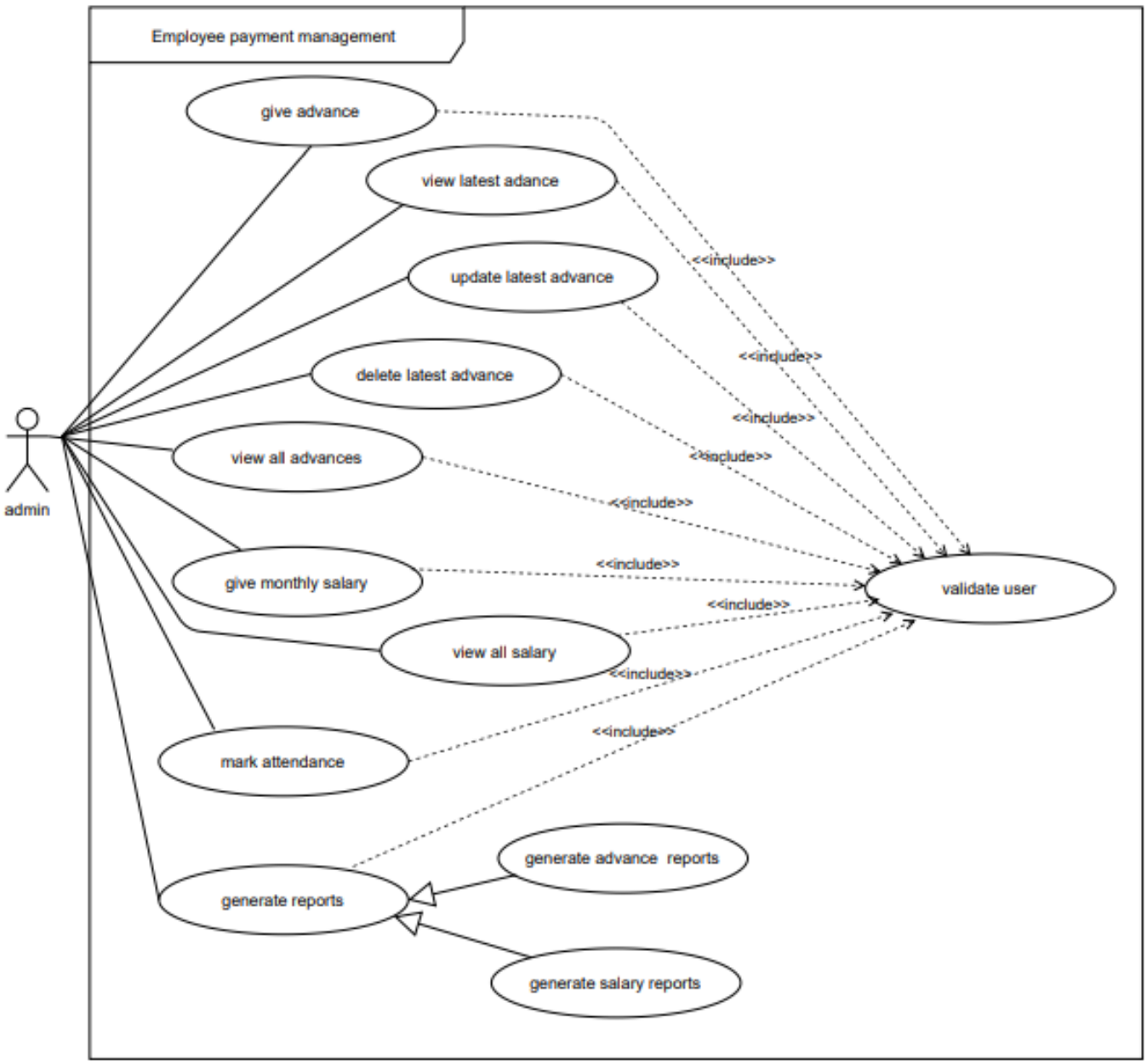


Figure 3: UD – employee payments management

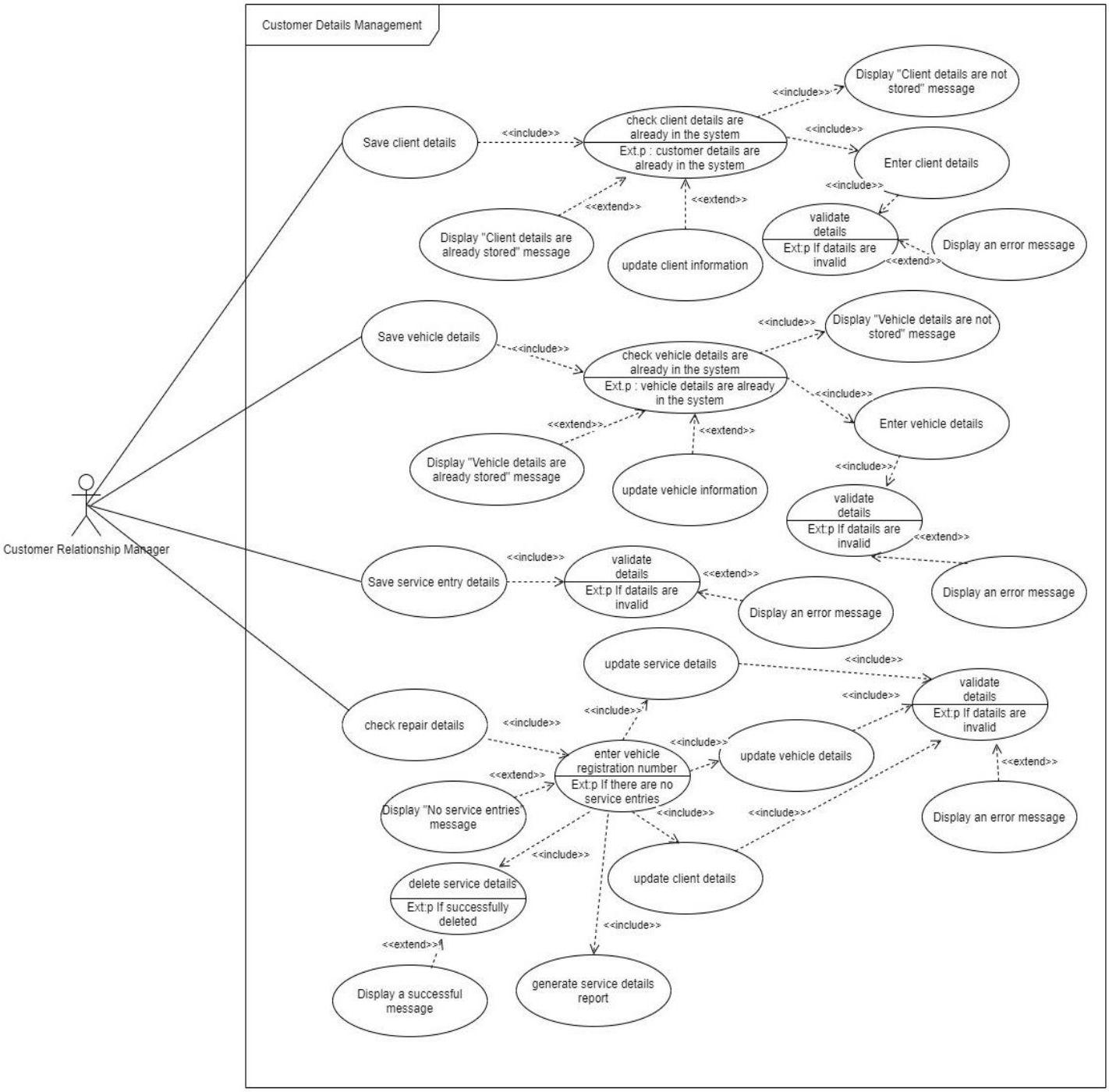


Figure 4: UD –customer details management

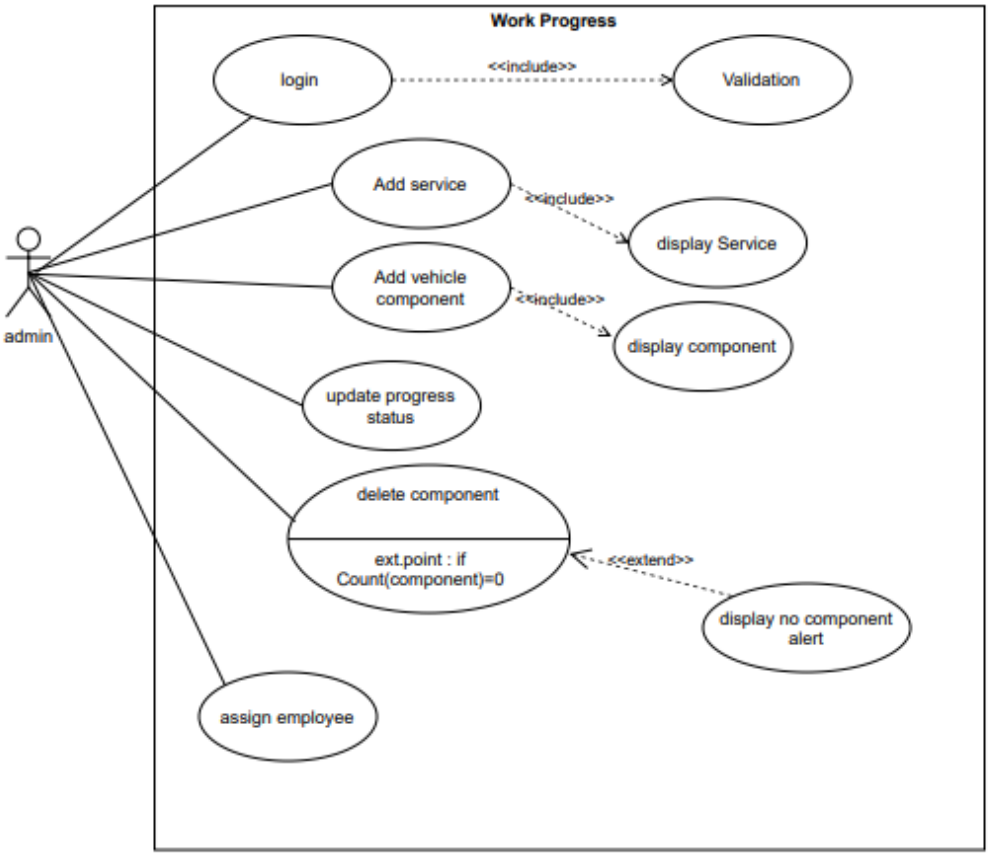


Figure 5: UD –work progress management

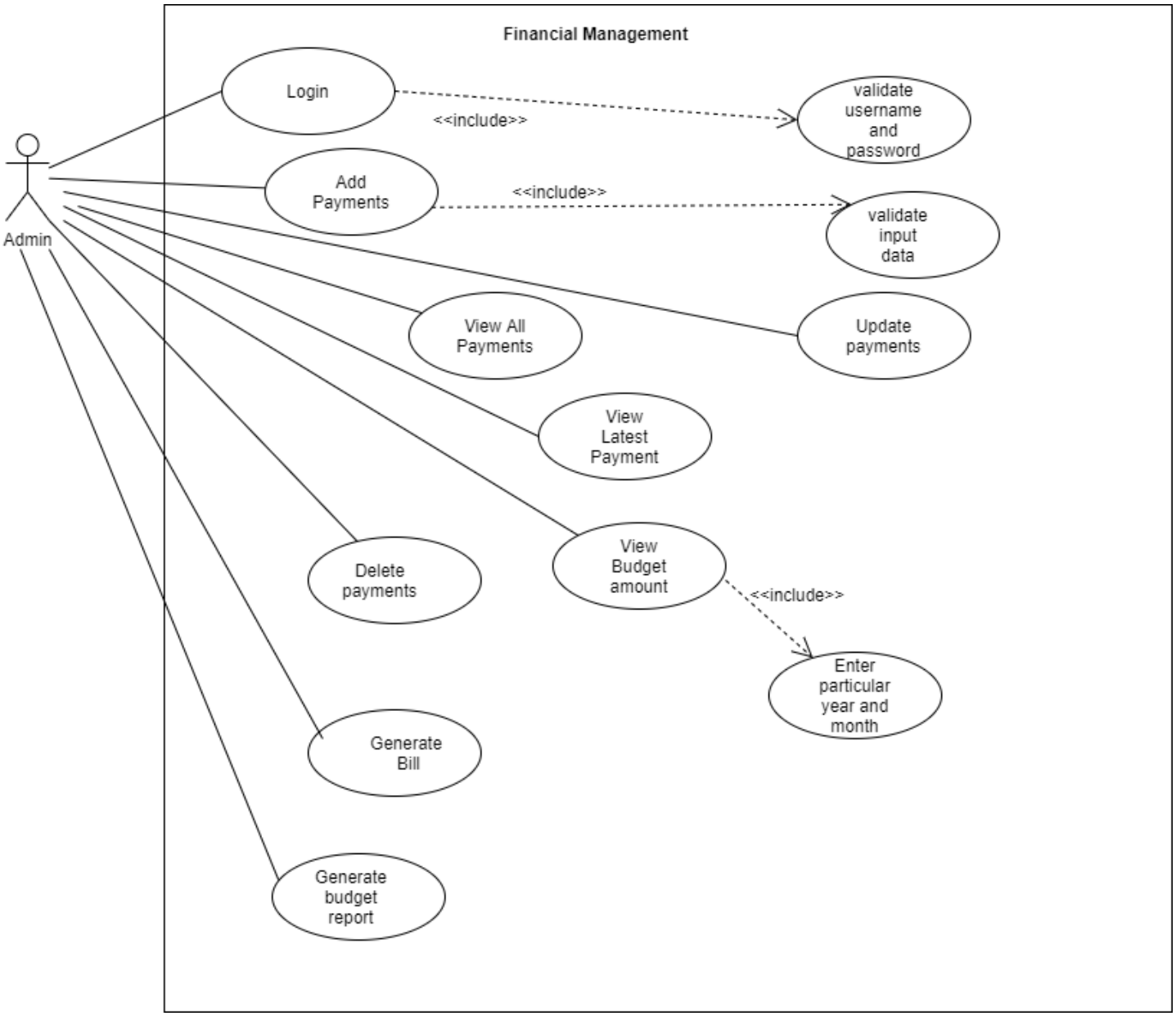


Figure 6: UD –financial management

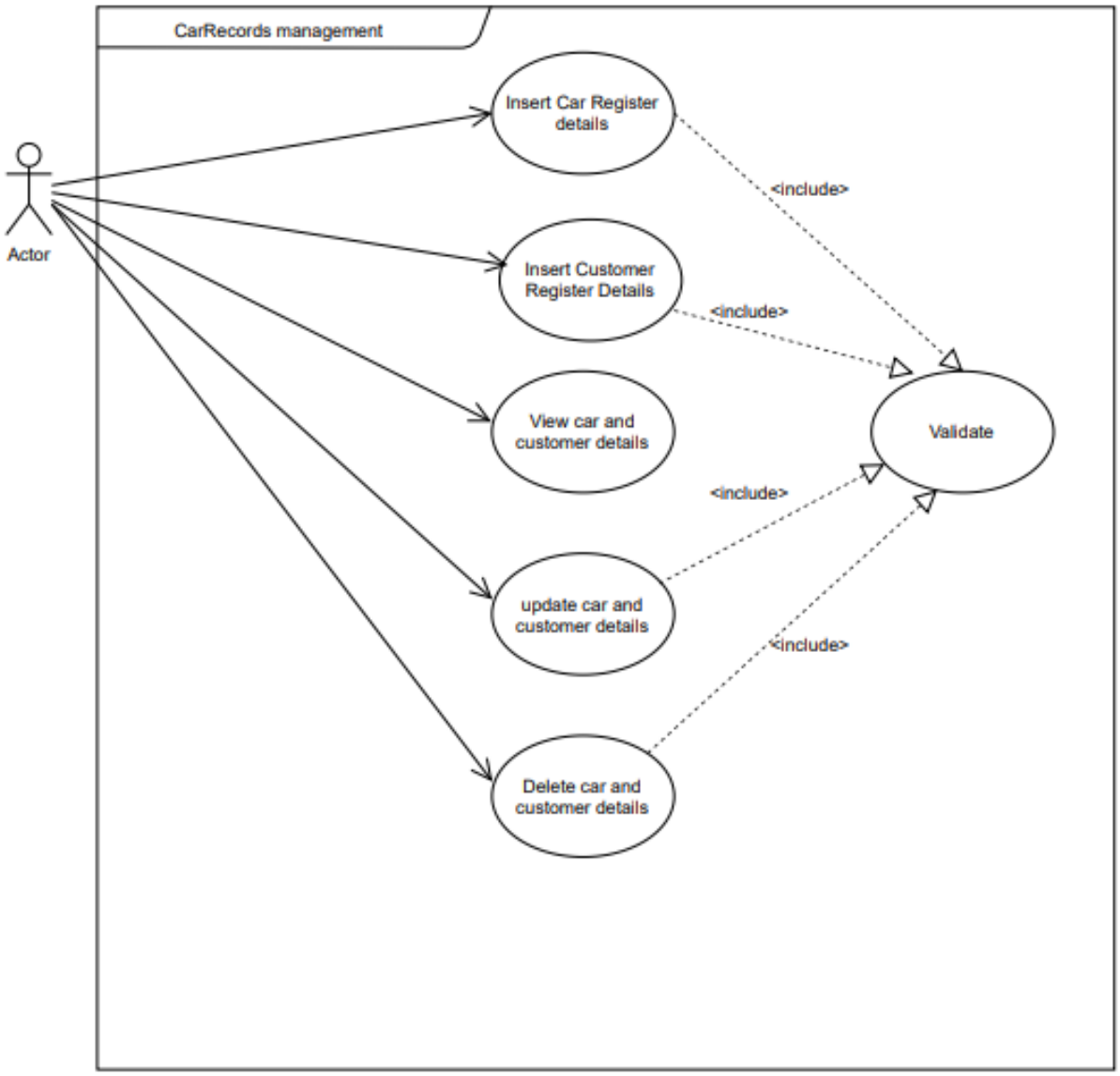


Figure 7: UD –car records management

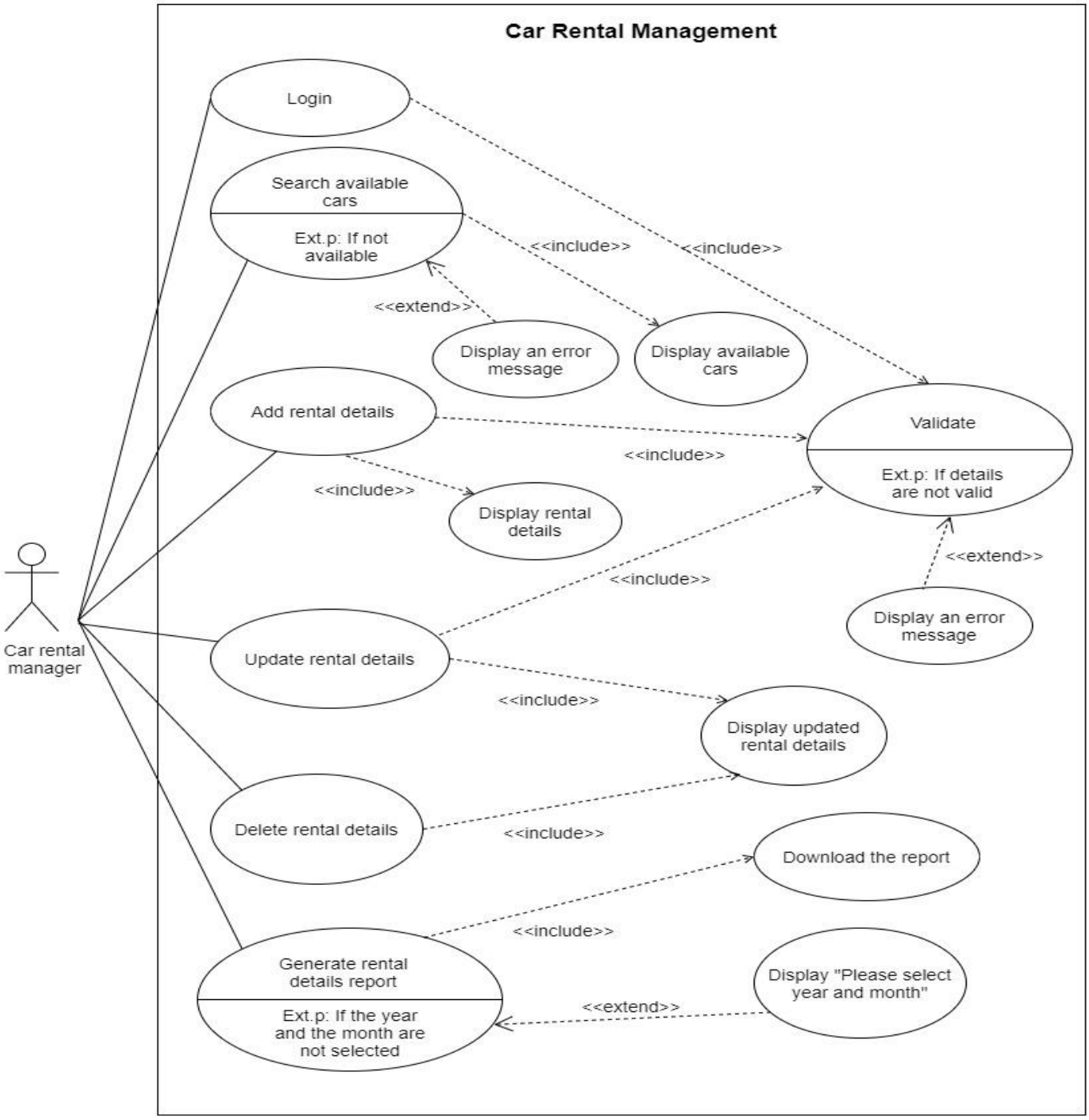
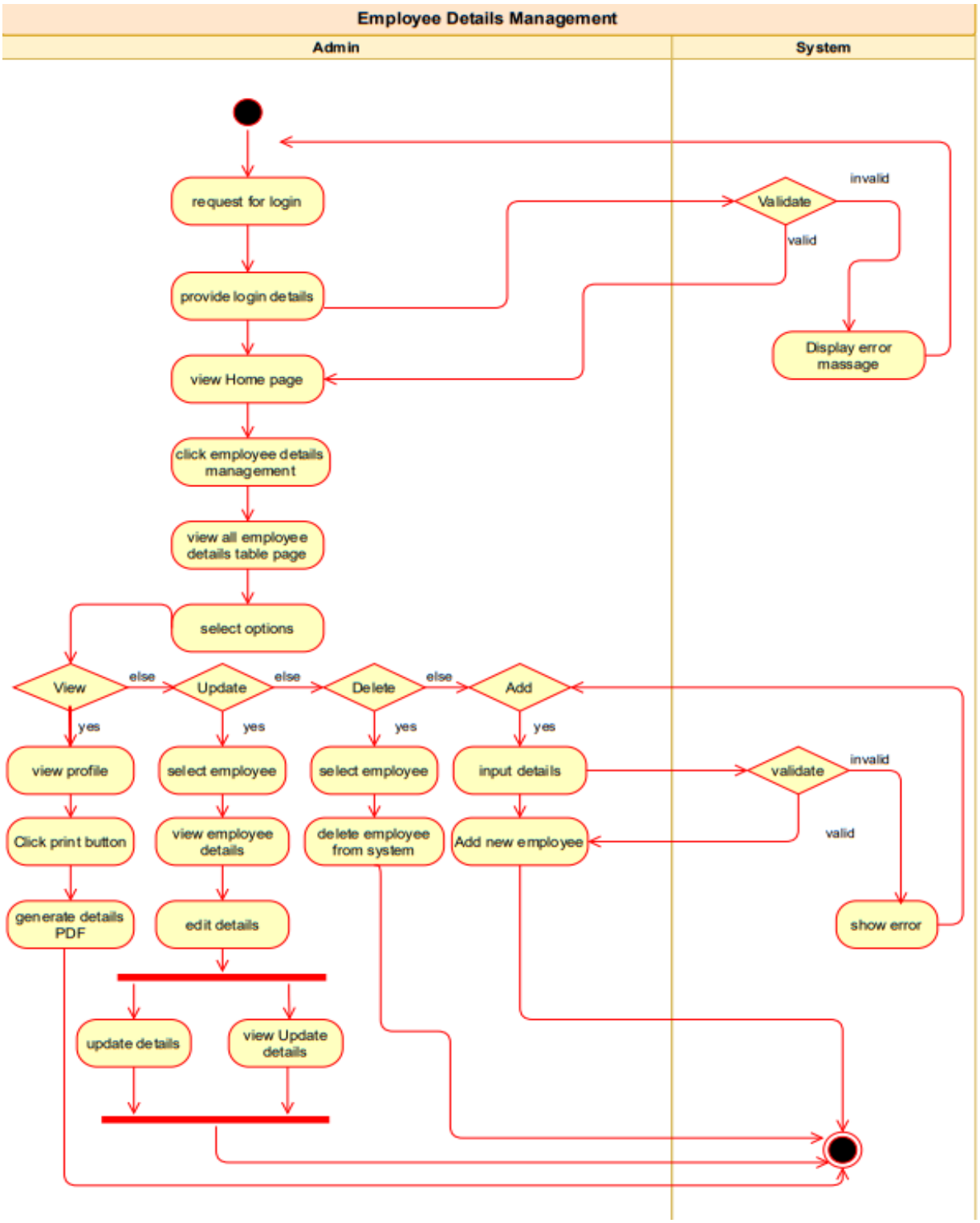


Figure 8: UD –car rental management

1.1.2 Activity Diagrams

Figure 9: AD – employee details management



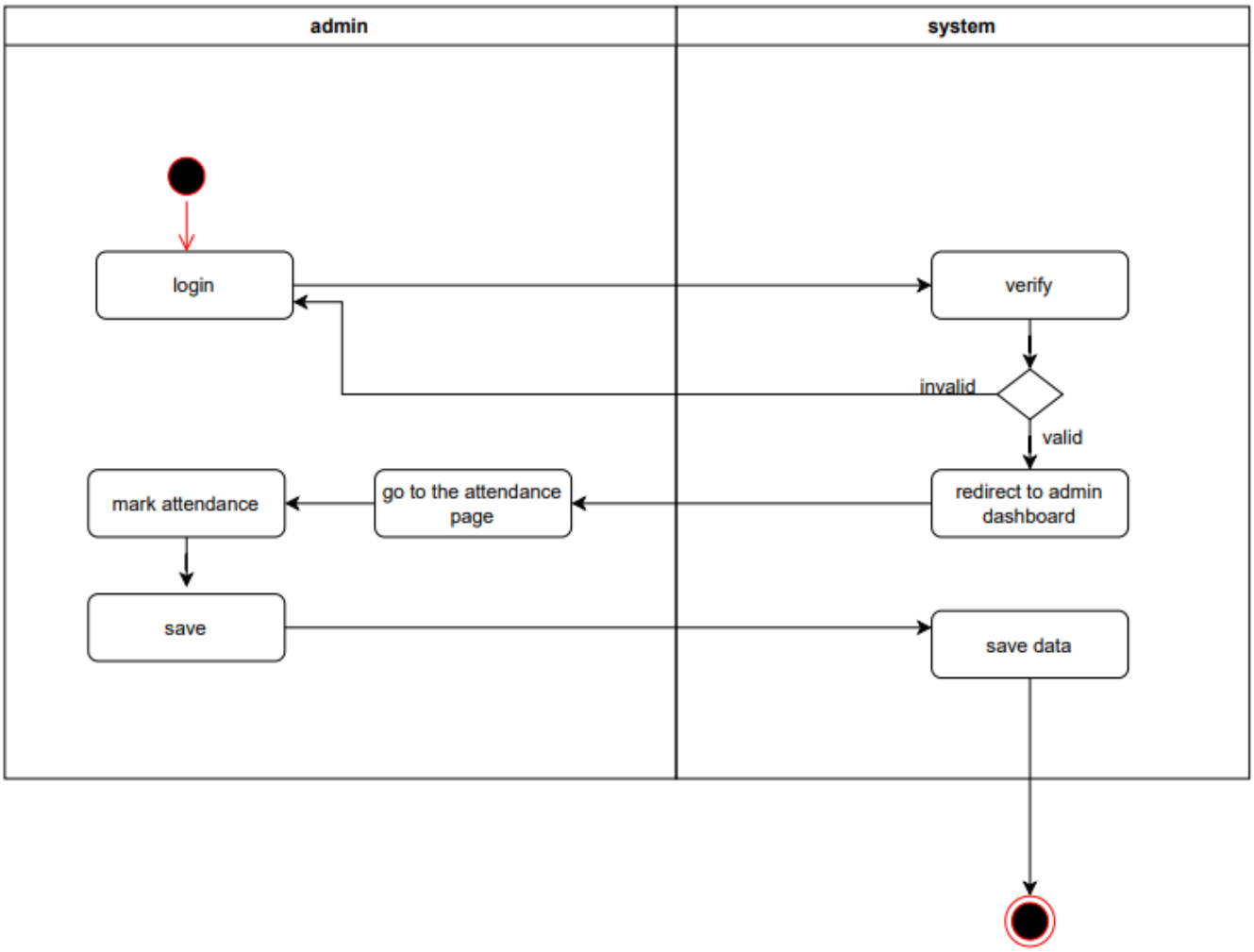


Figure 10: AD – Employee payment management (attendance management)

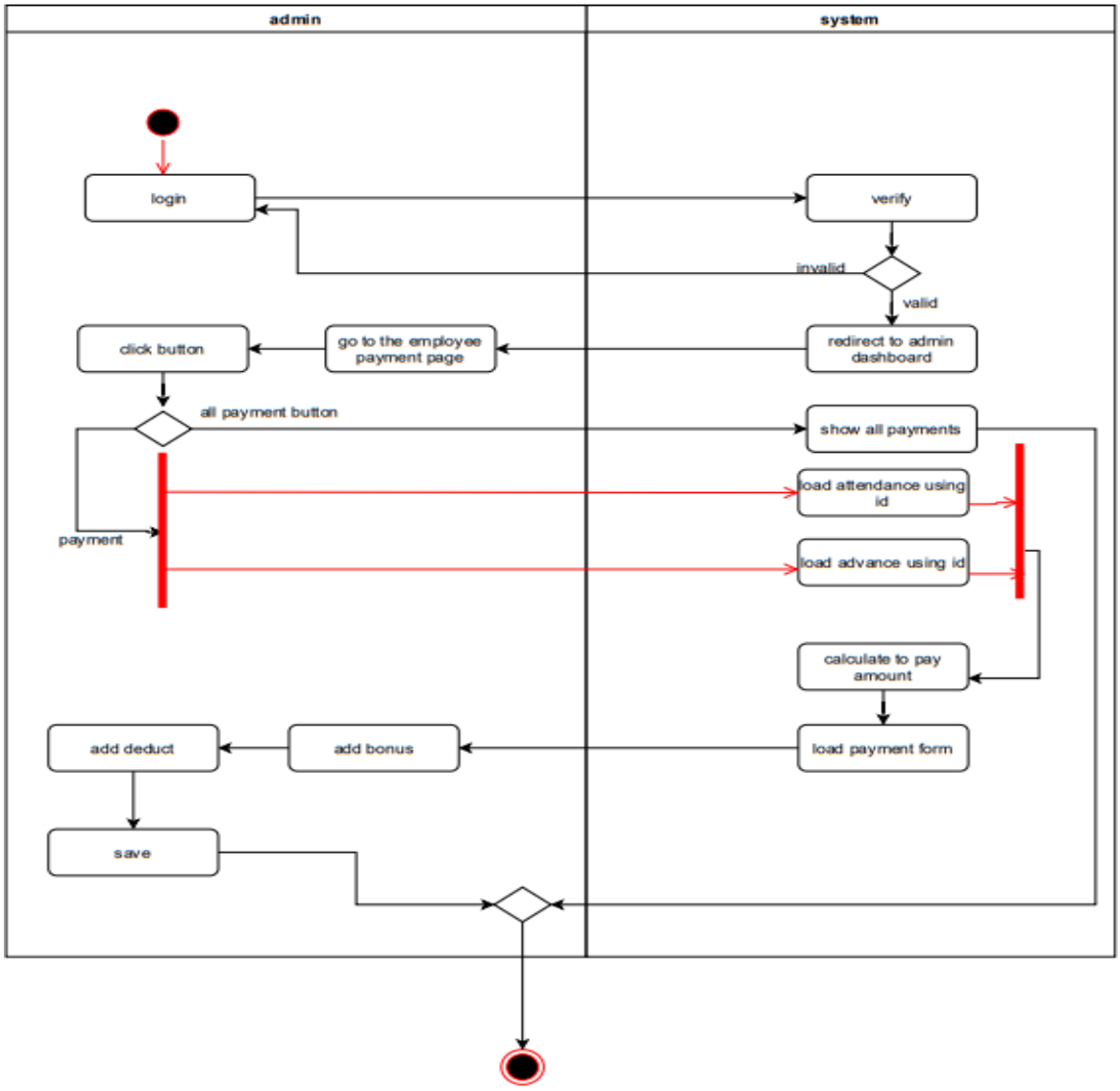


Figure 11: AD – Employee payment management (salary management)

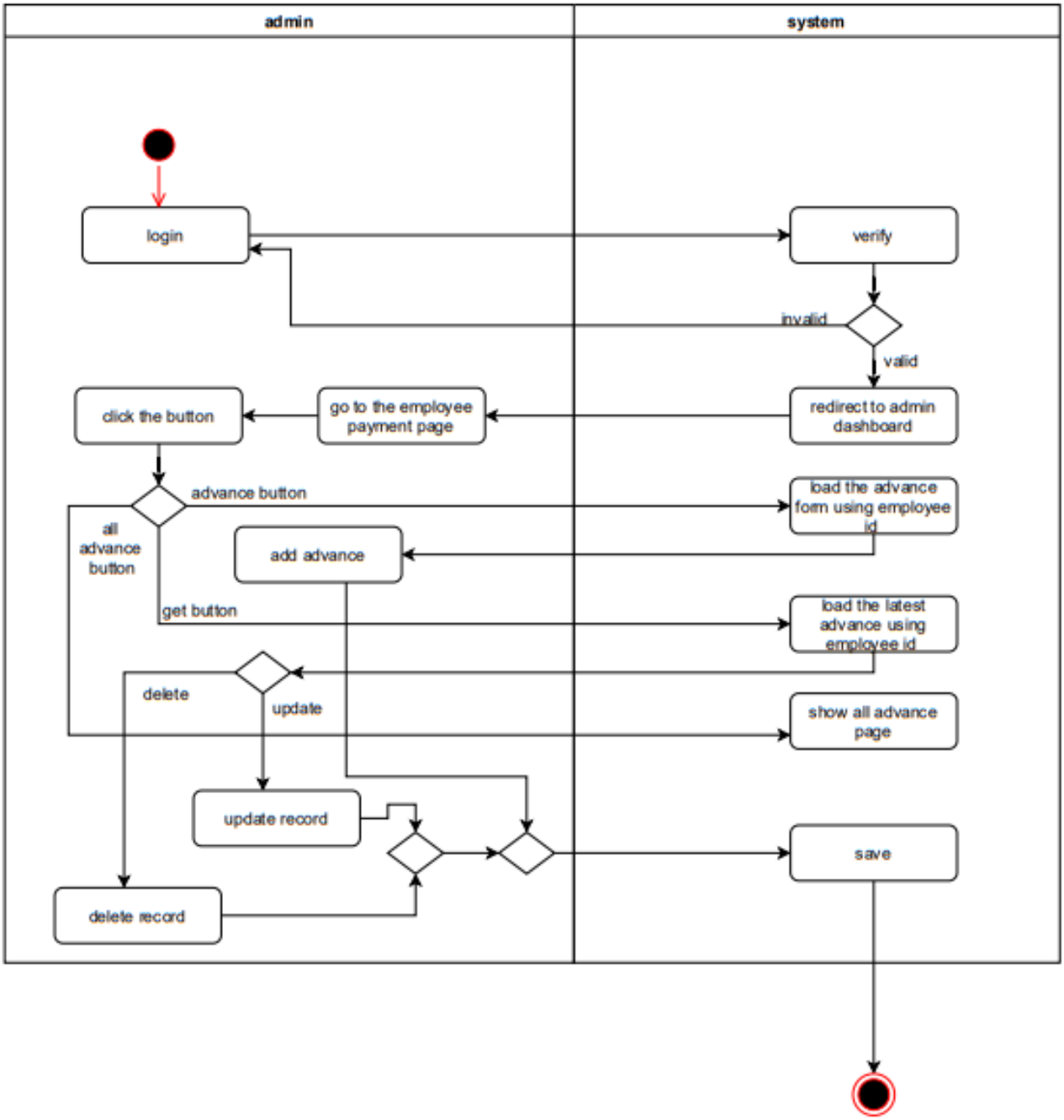


Figure 12: AD – Employee payment management (advance management)

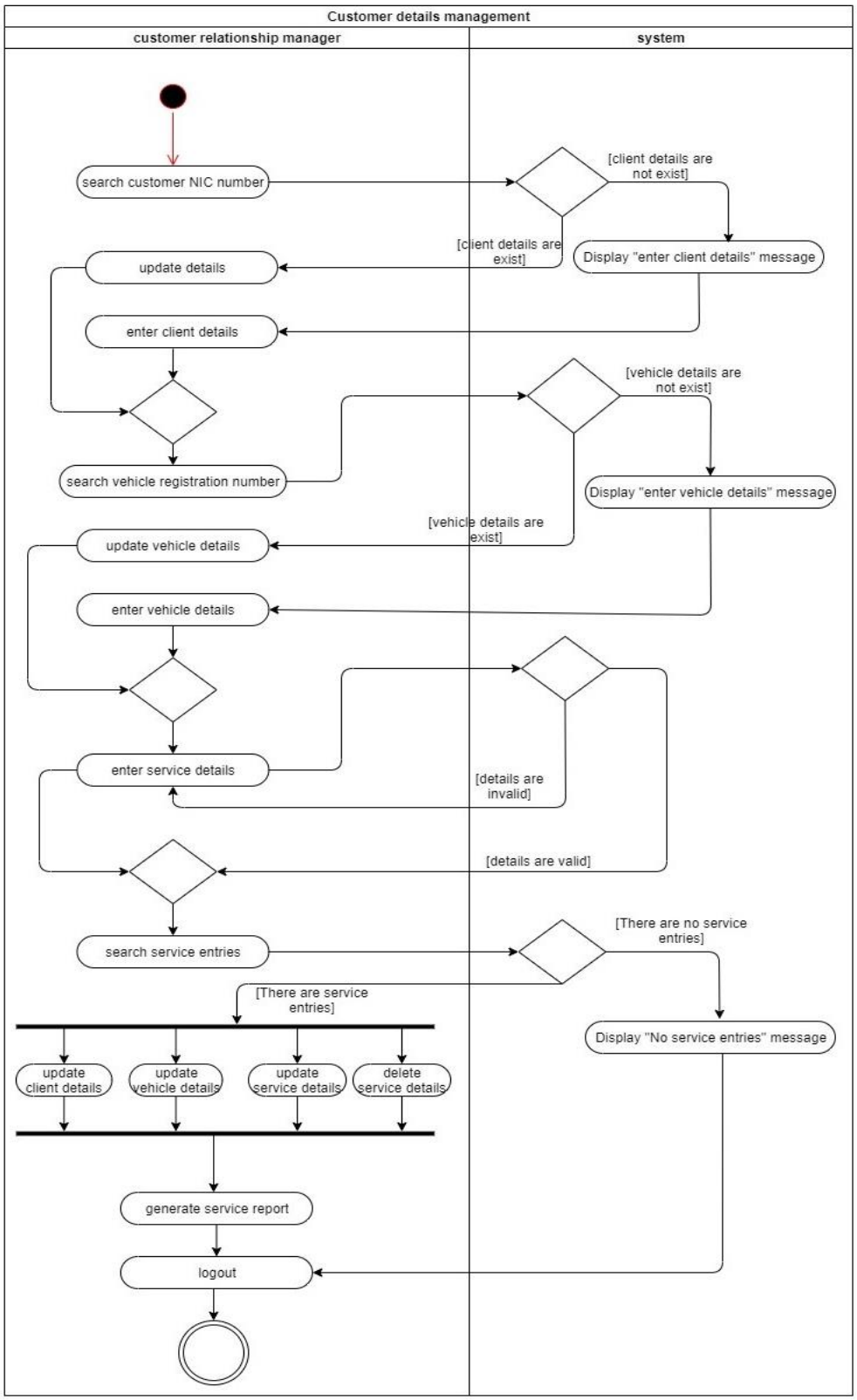


Figure 13: AD-customer details management

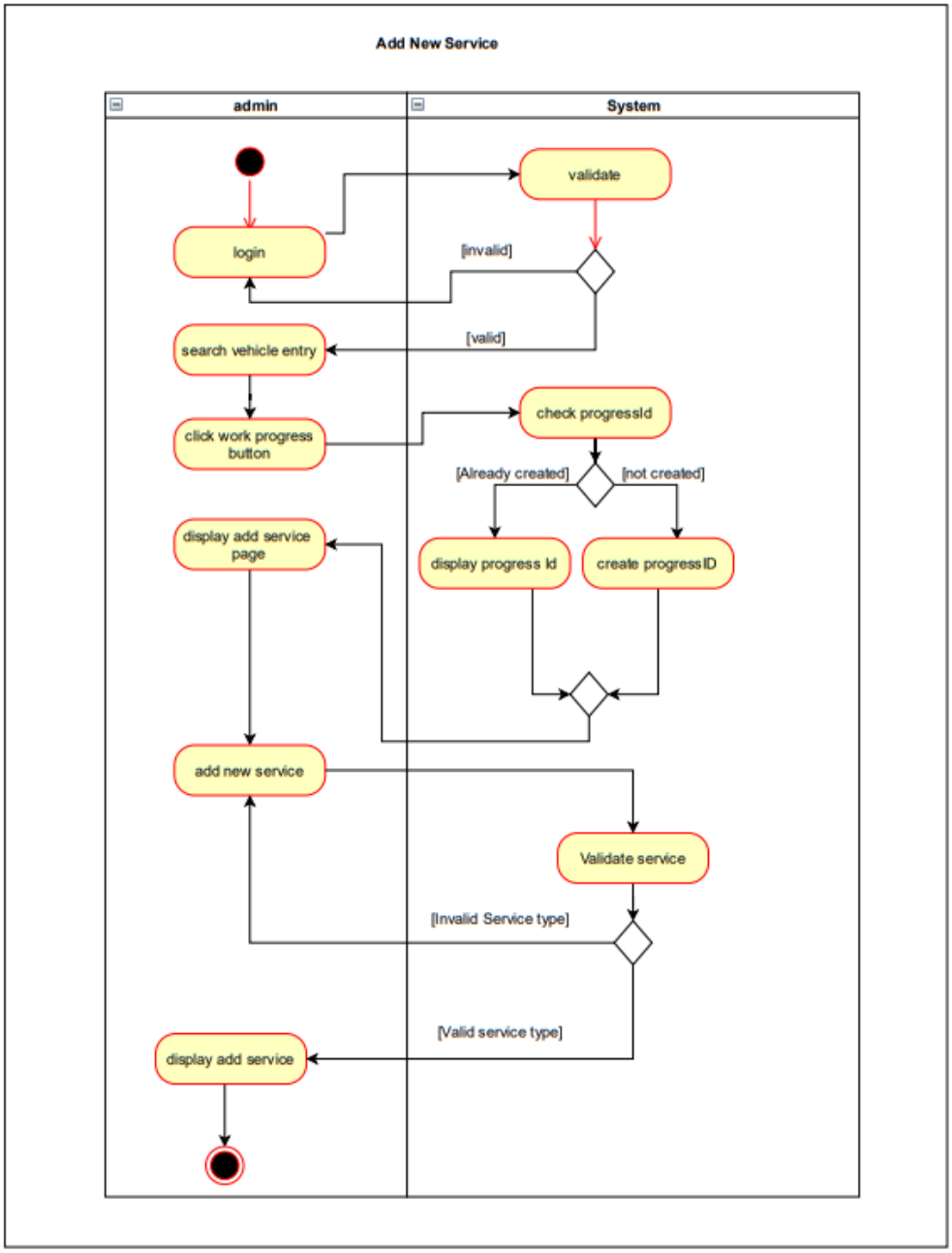


Figure 14:AD work progress management

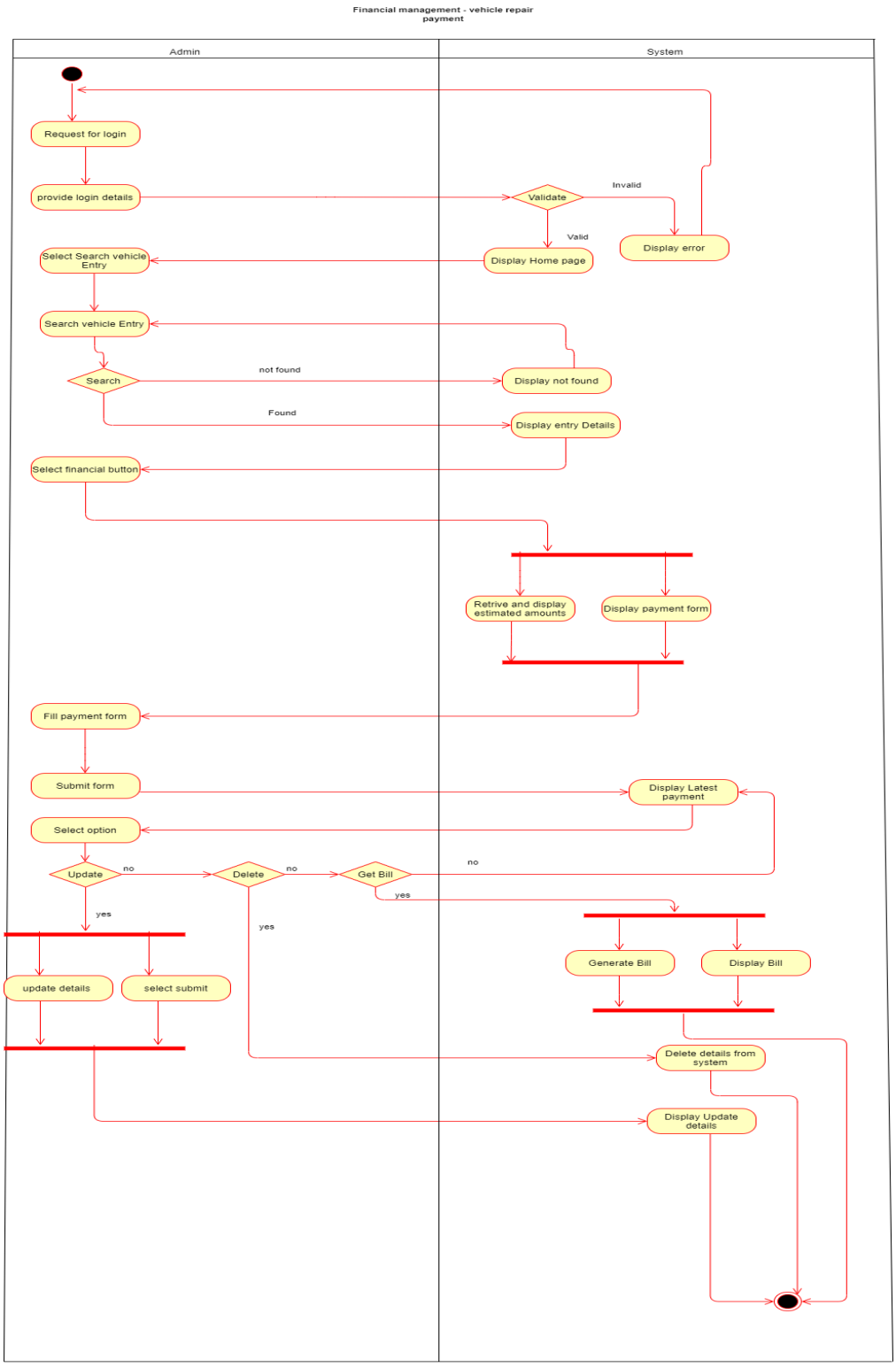


Figure 15: AD financial management (payment management)

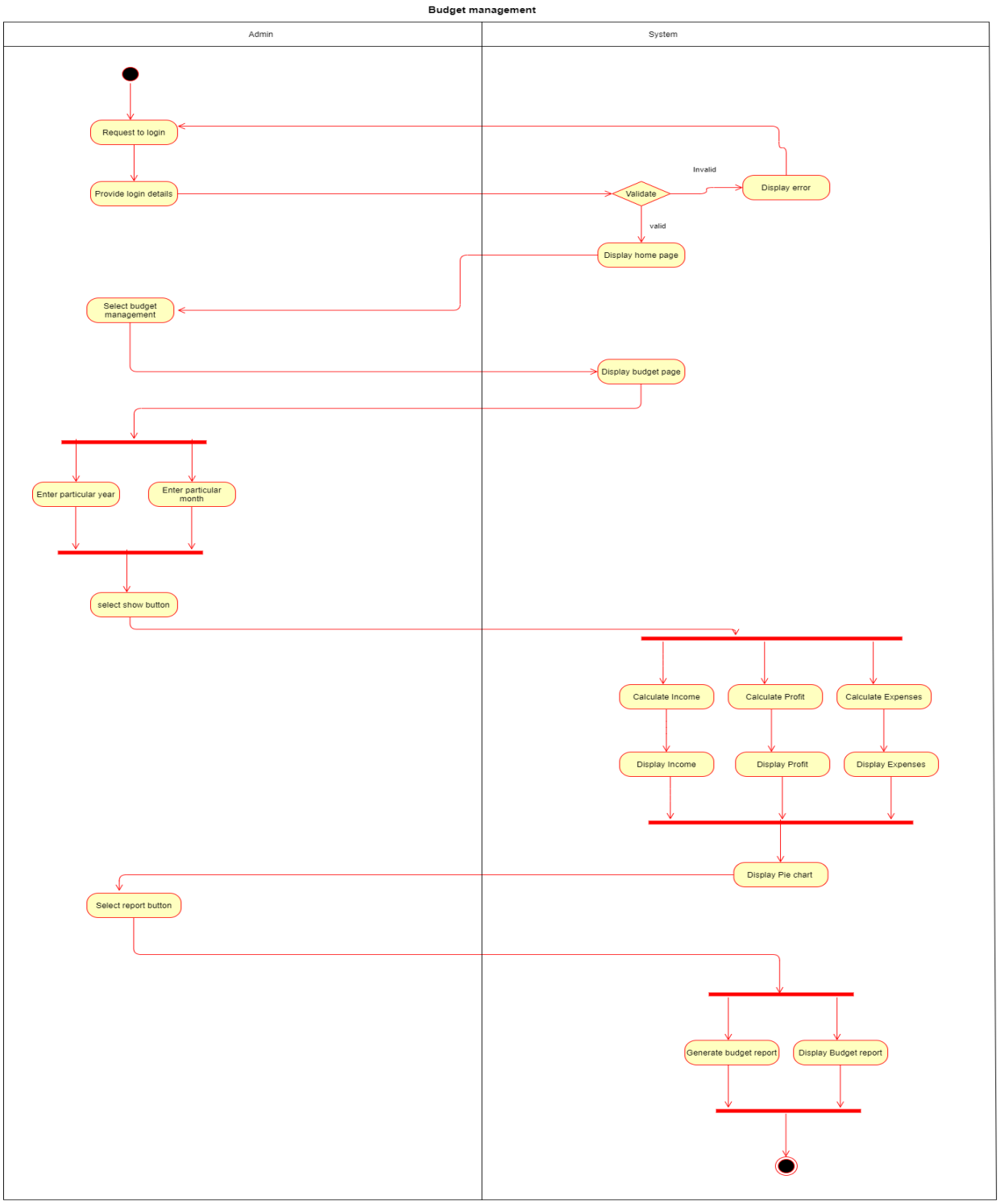


Figure 16:AD financial management (Budget management)

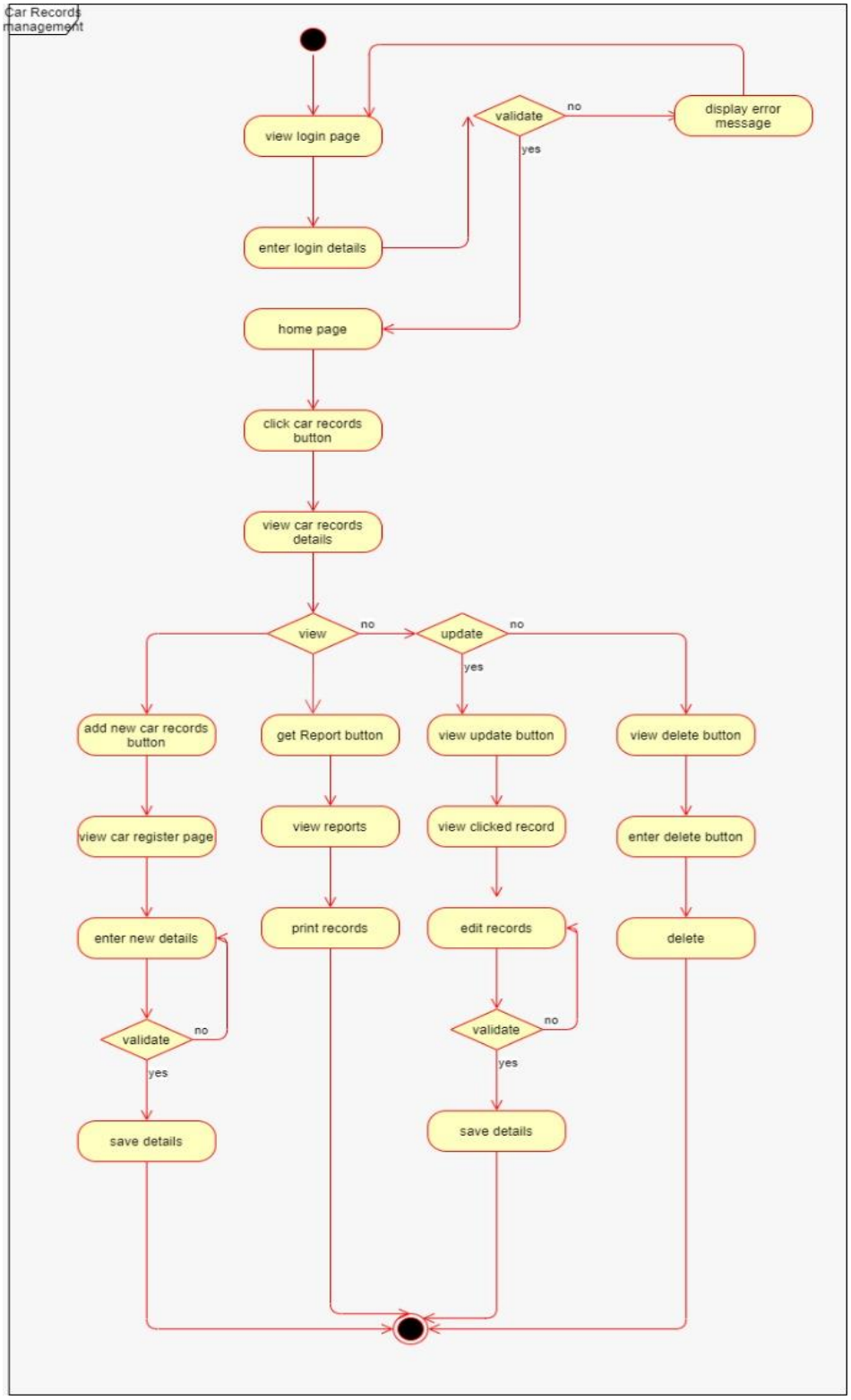
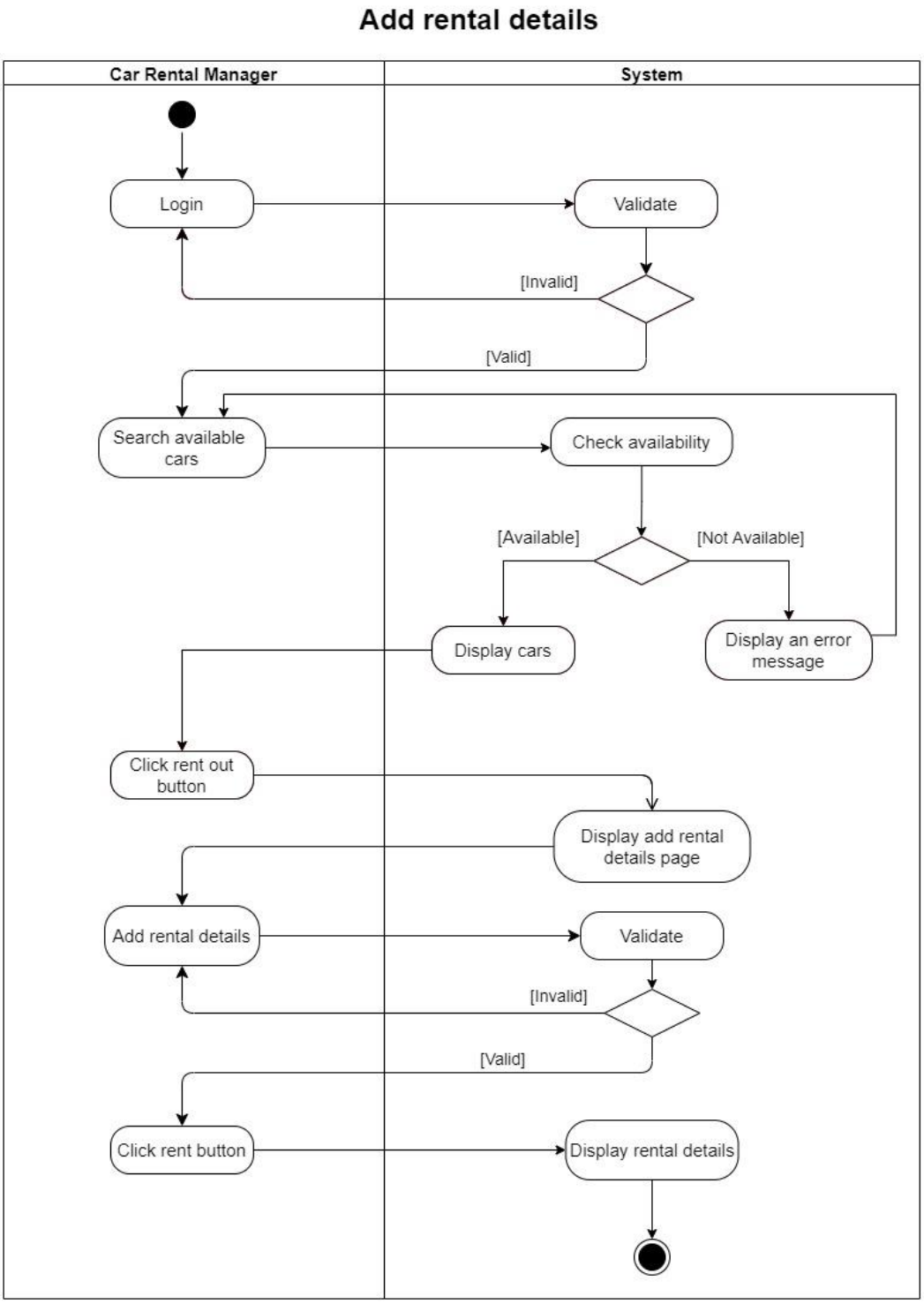


Figure 17:AD car records management

Figure 18:AD car rental management



1.2 Design

1.2.1 High Level Architecture Diagram

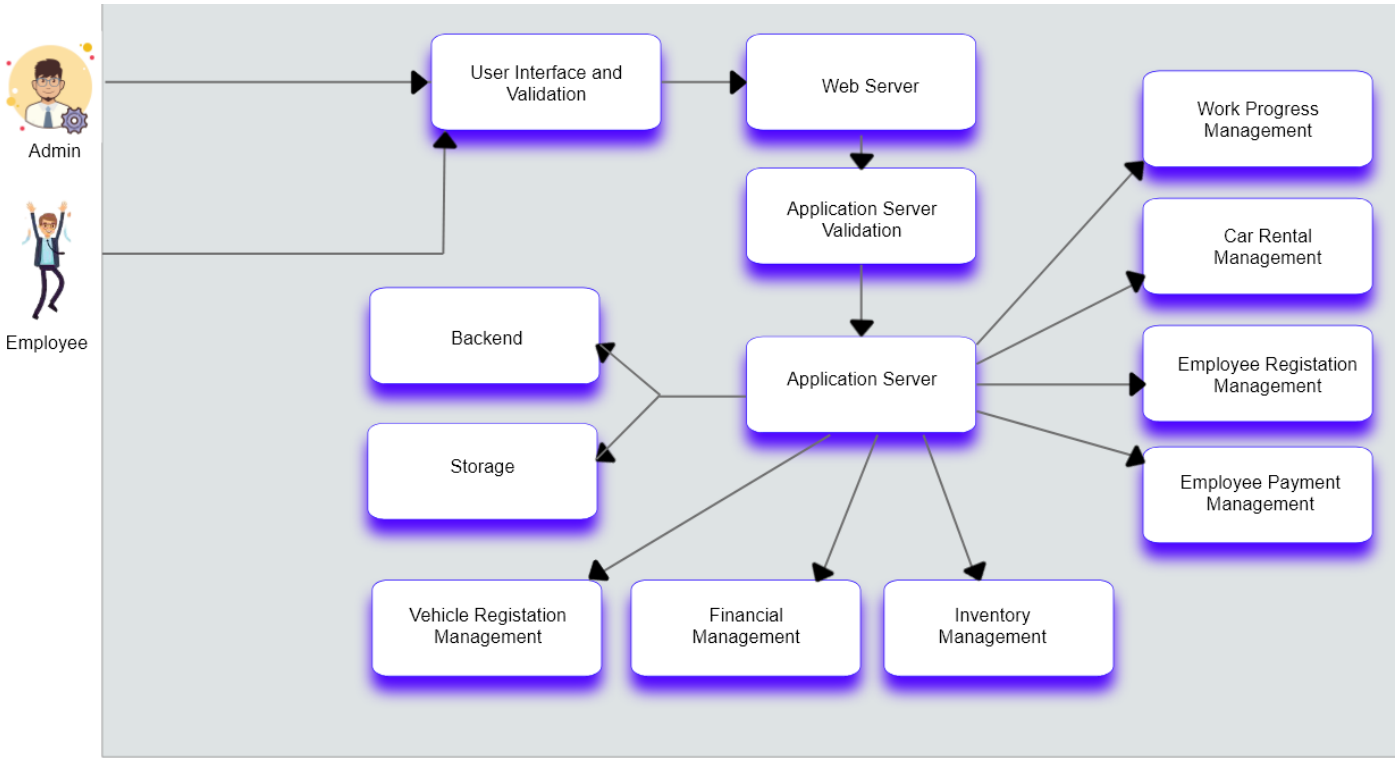
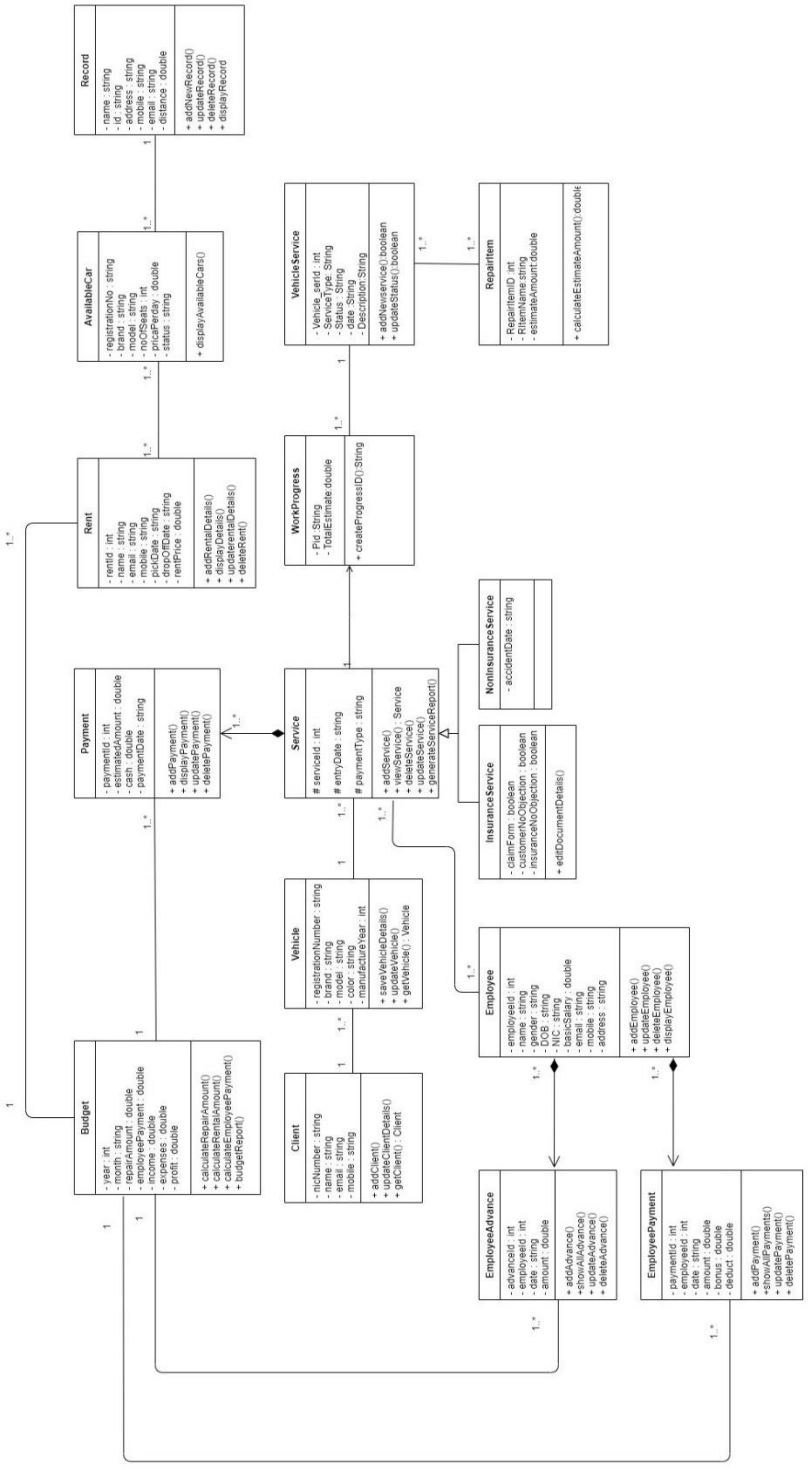
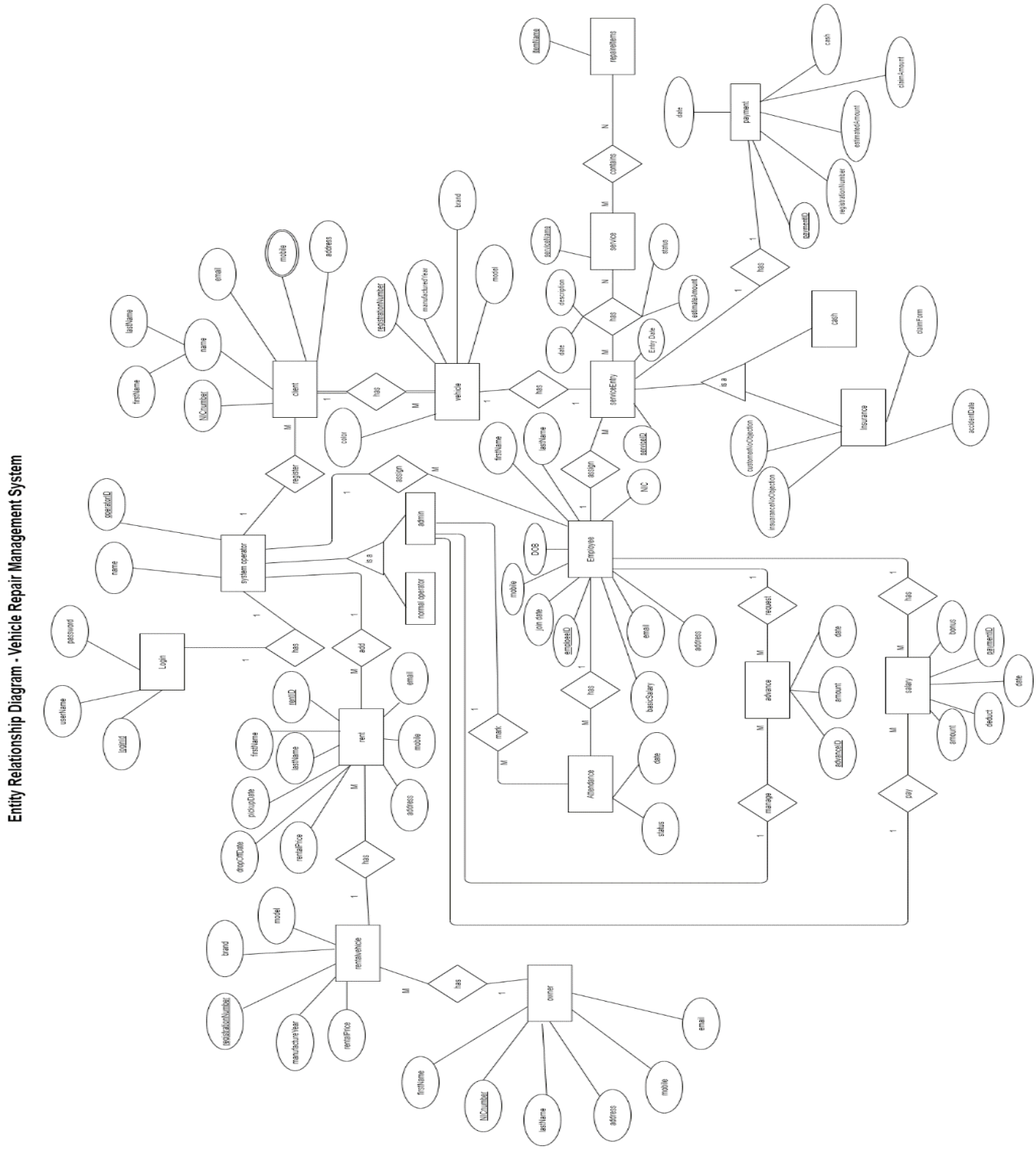


Figure 19: High level architecture diagram

1.2.2 Class Diagram



1.2.3 Entity Relationship Diagram



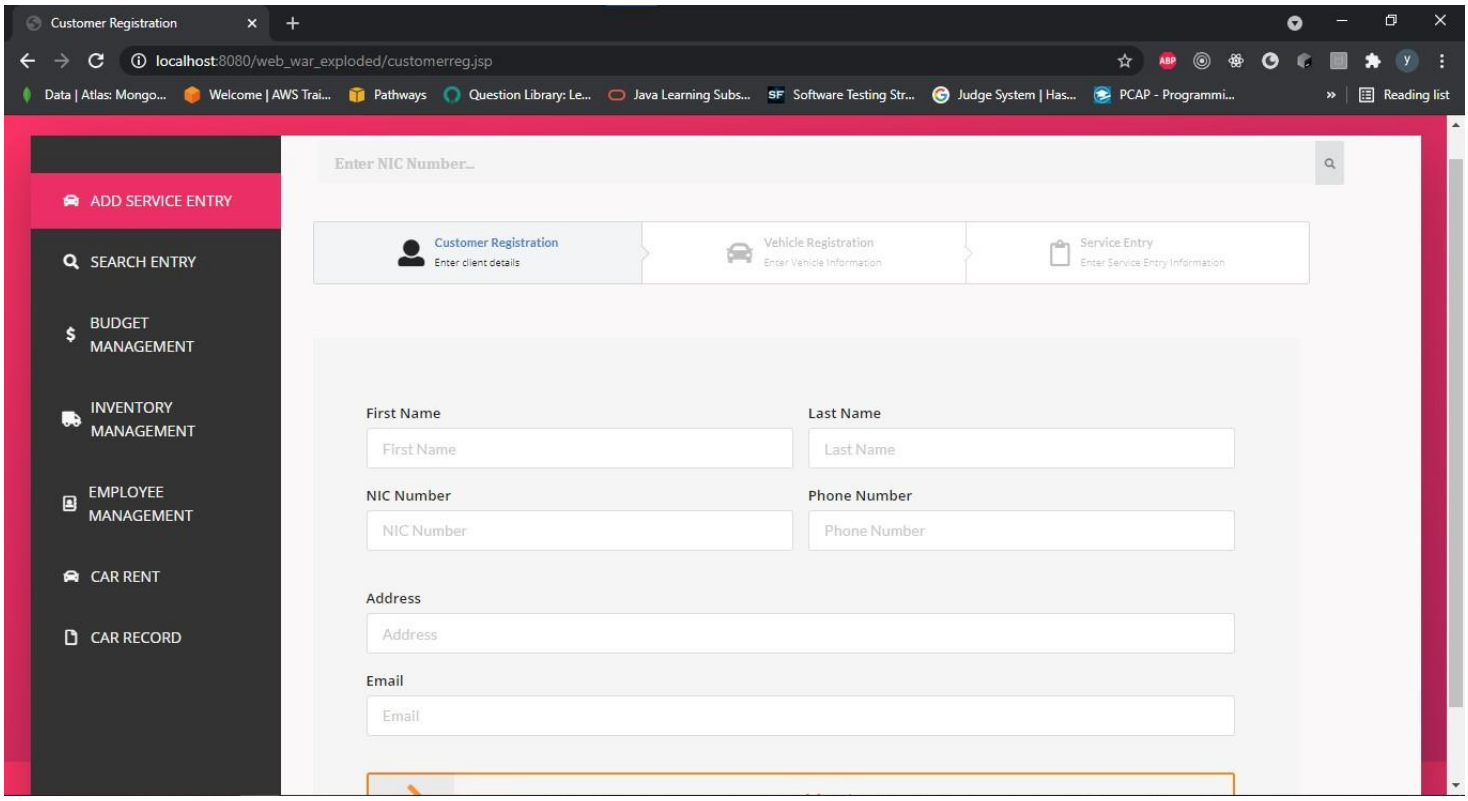


Figure 22: customer registration

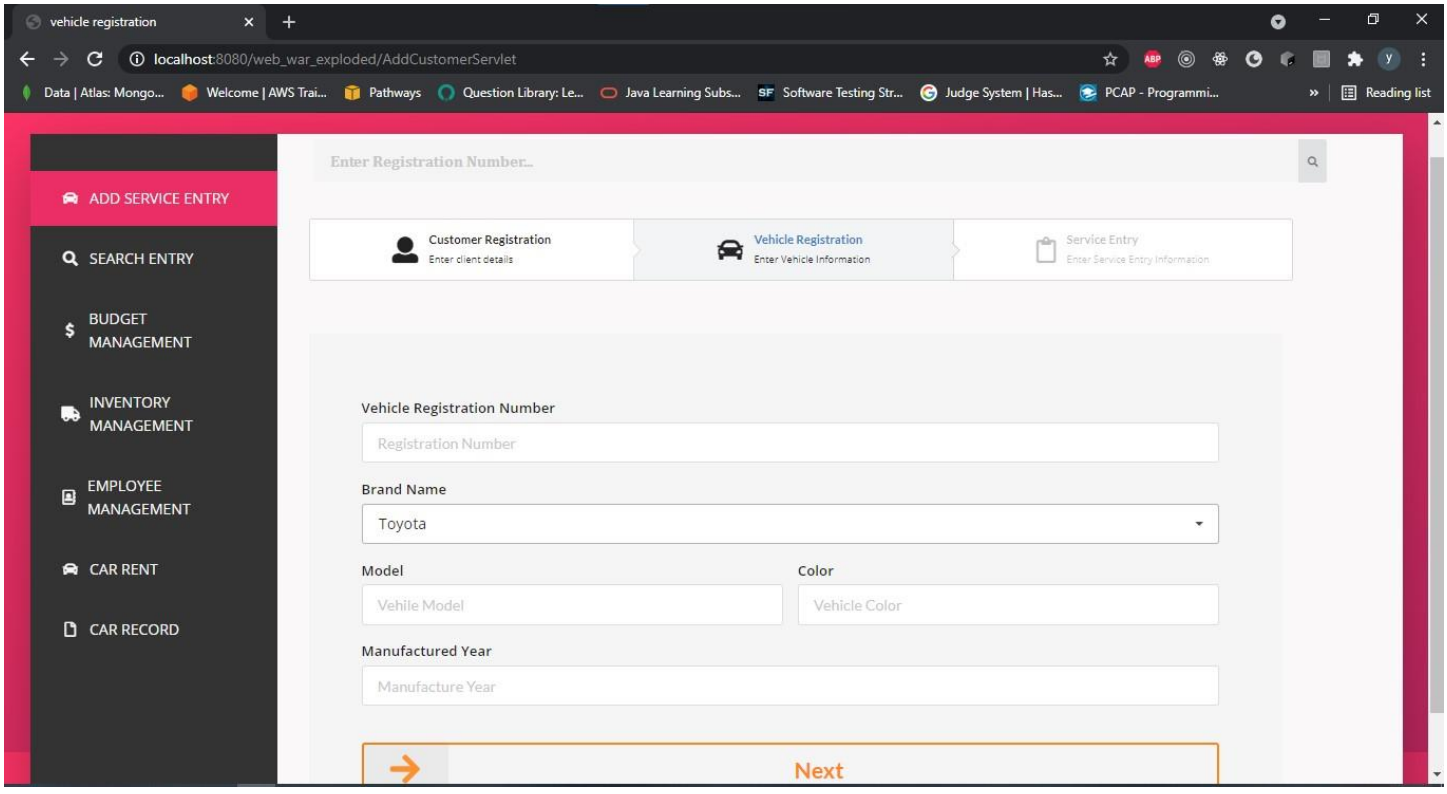


Figure 23: vehicle registraion

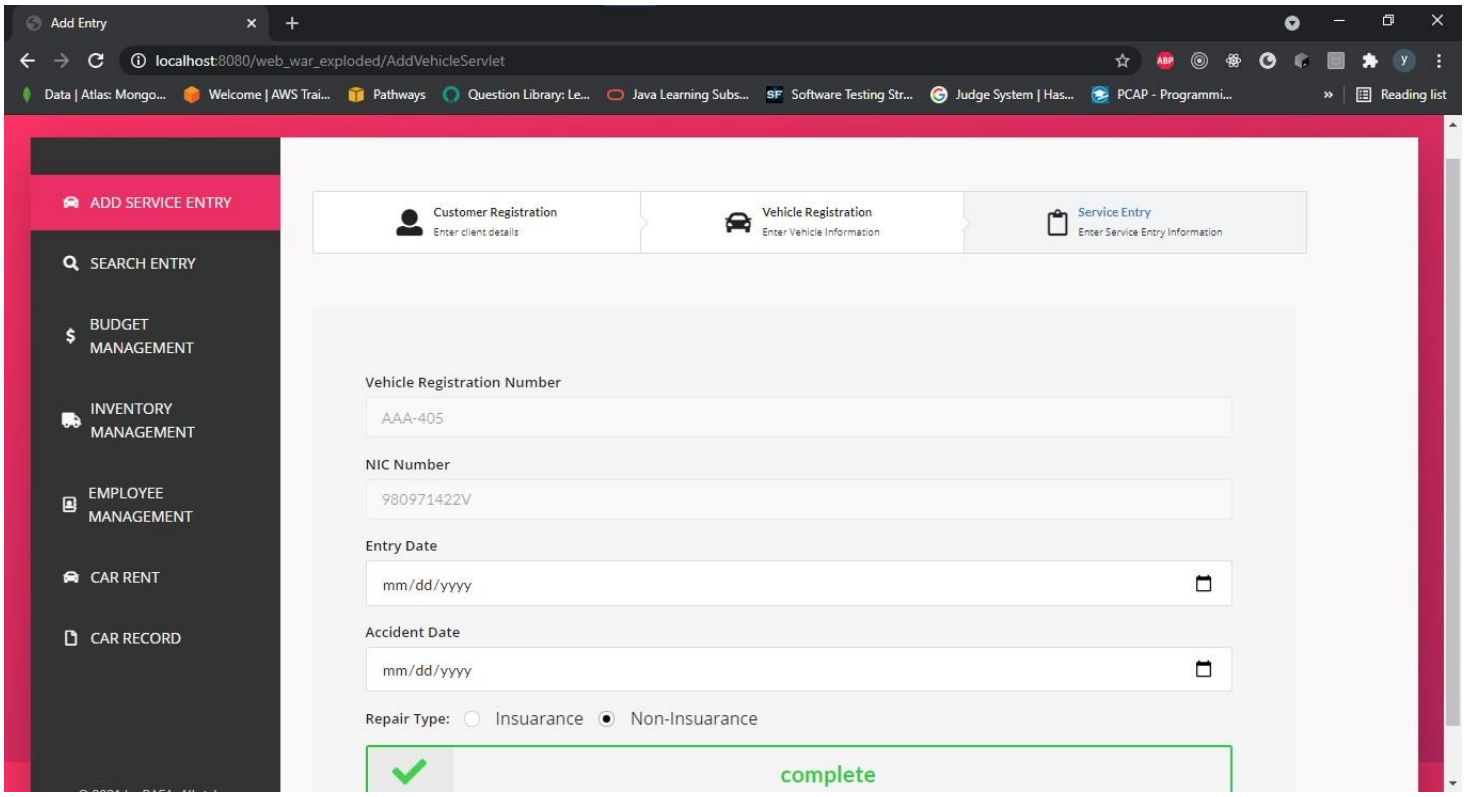


Figure 24: add service

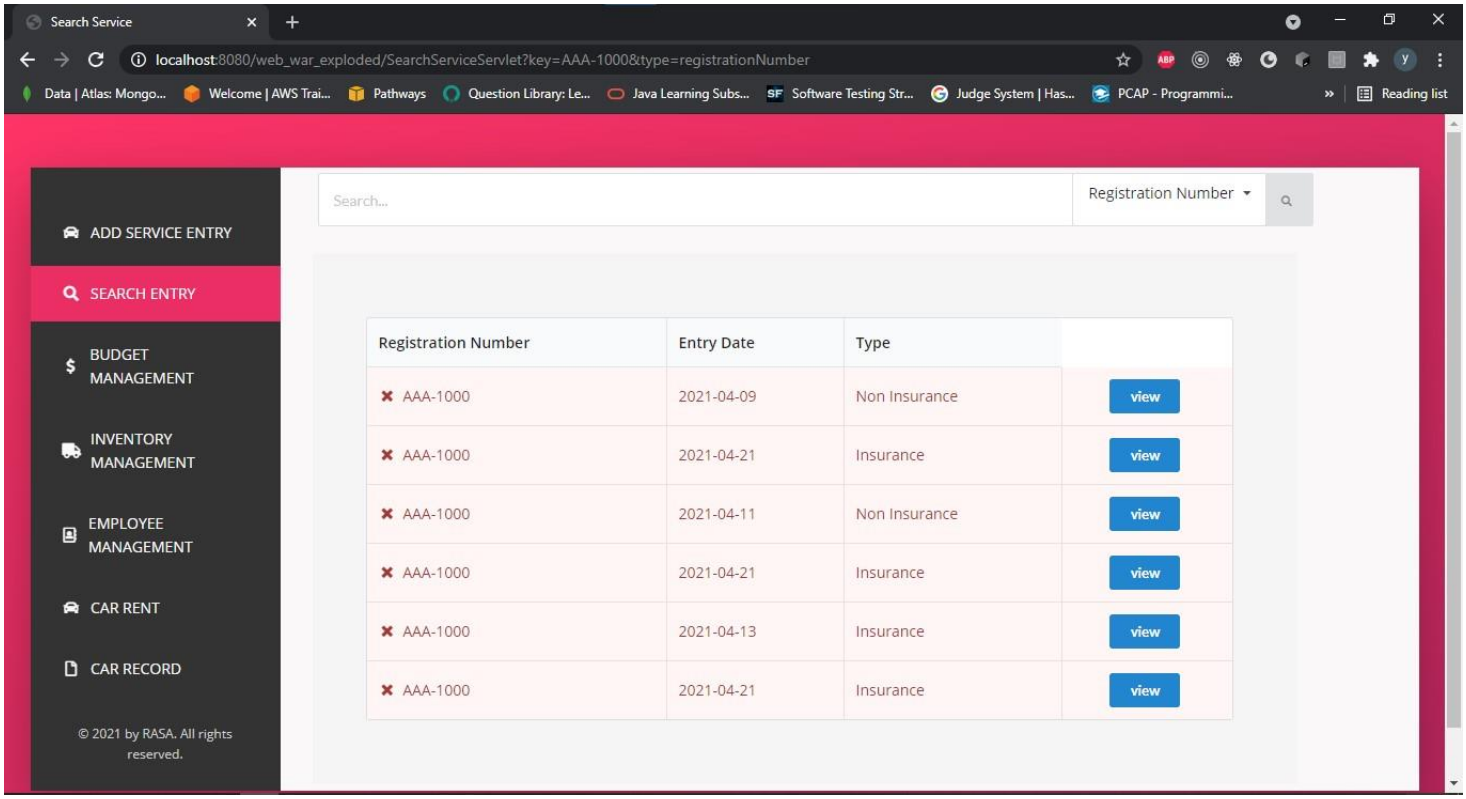


Figure 25: search service entry

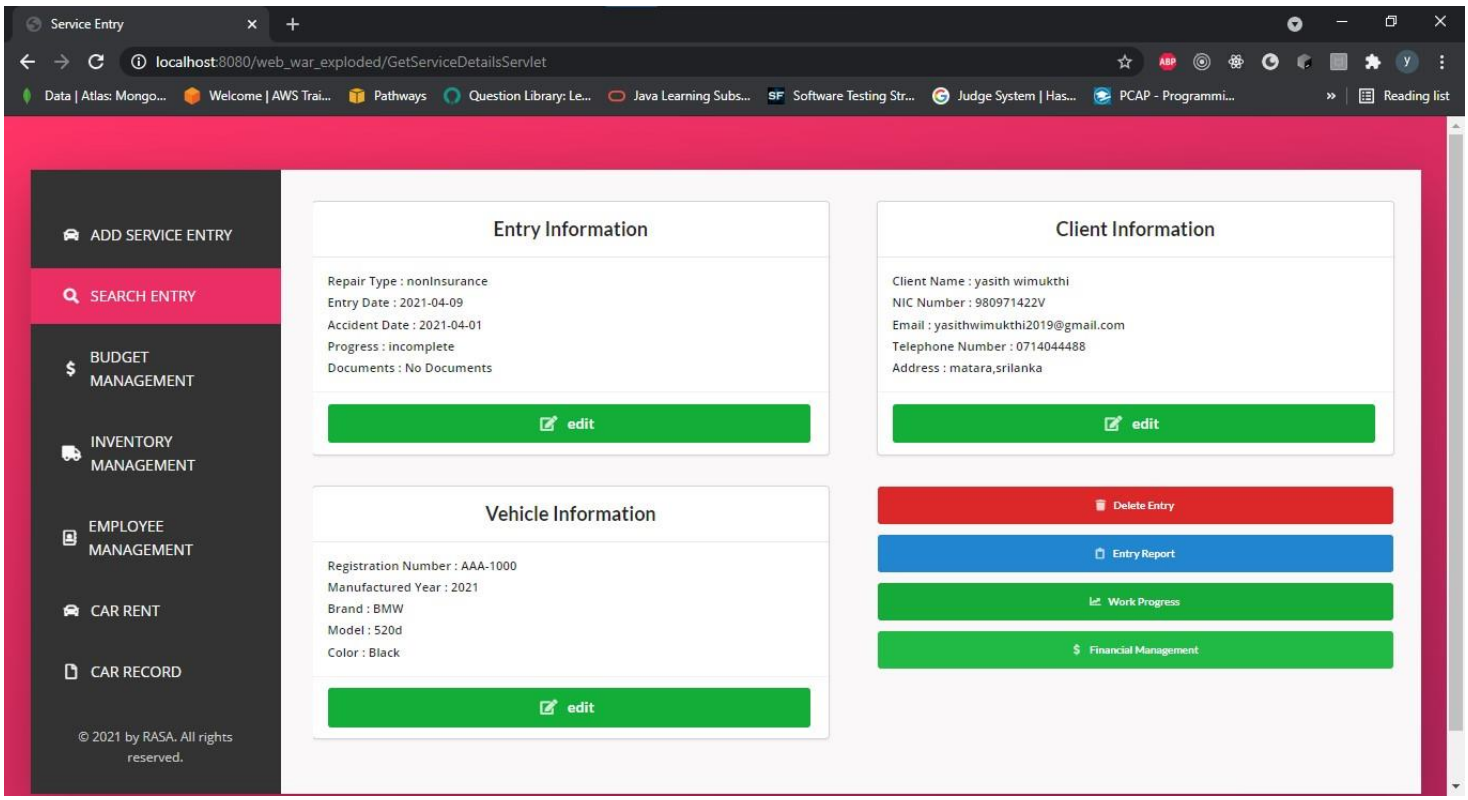


Figure 26: service entry page

Figure 27: ***Work progress management***

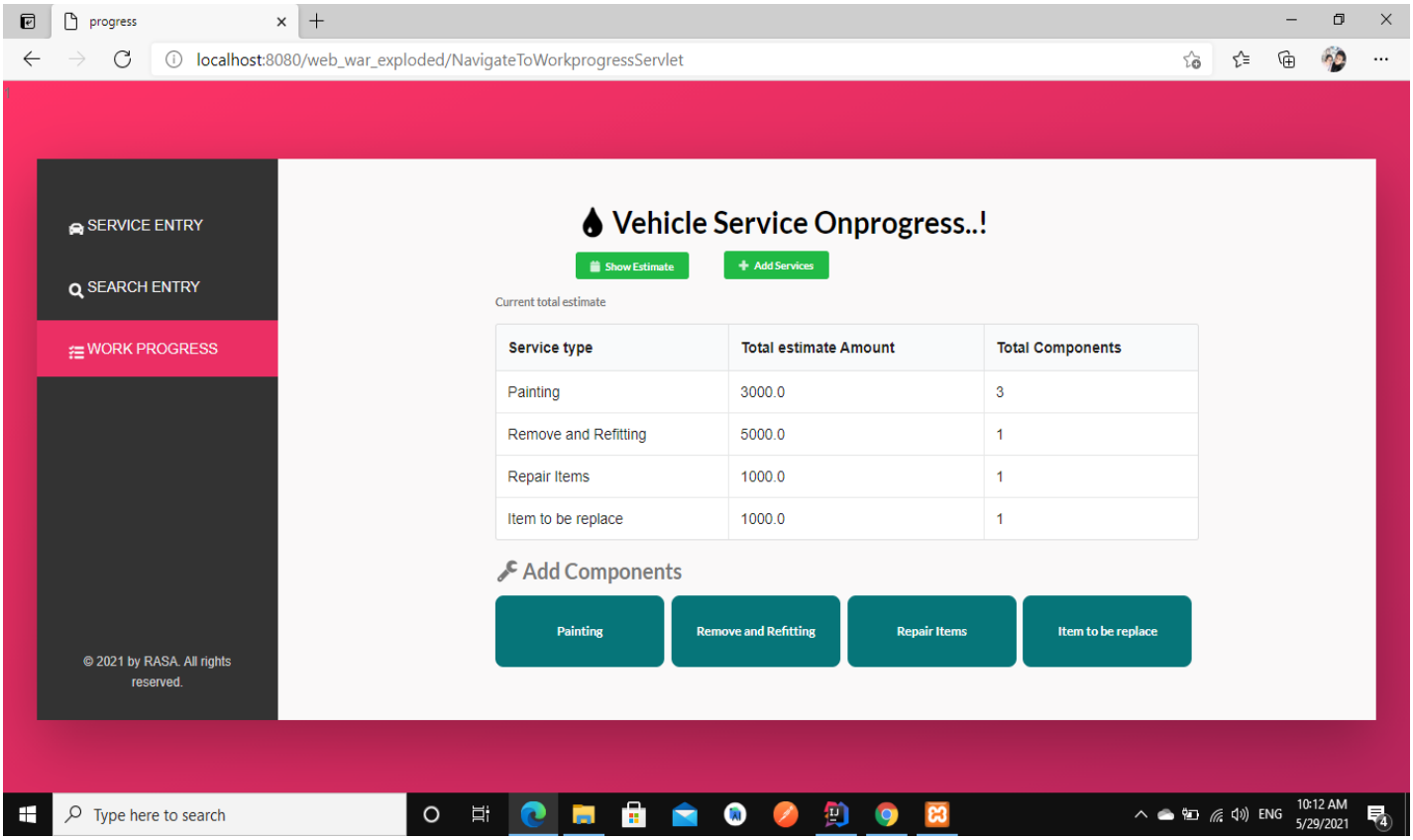


Figure 28: ***Add new Service***

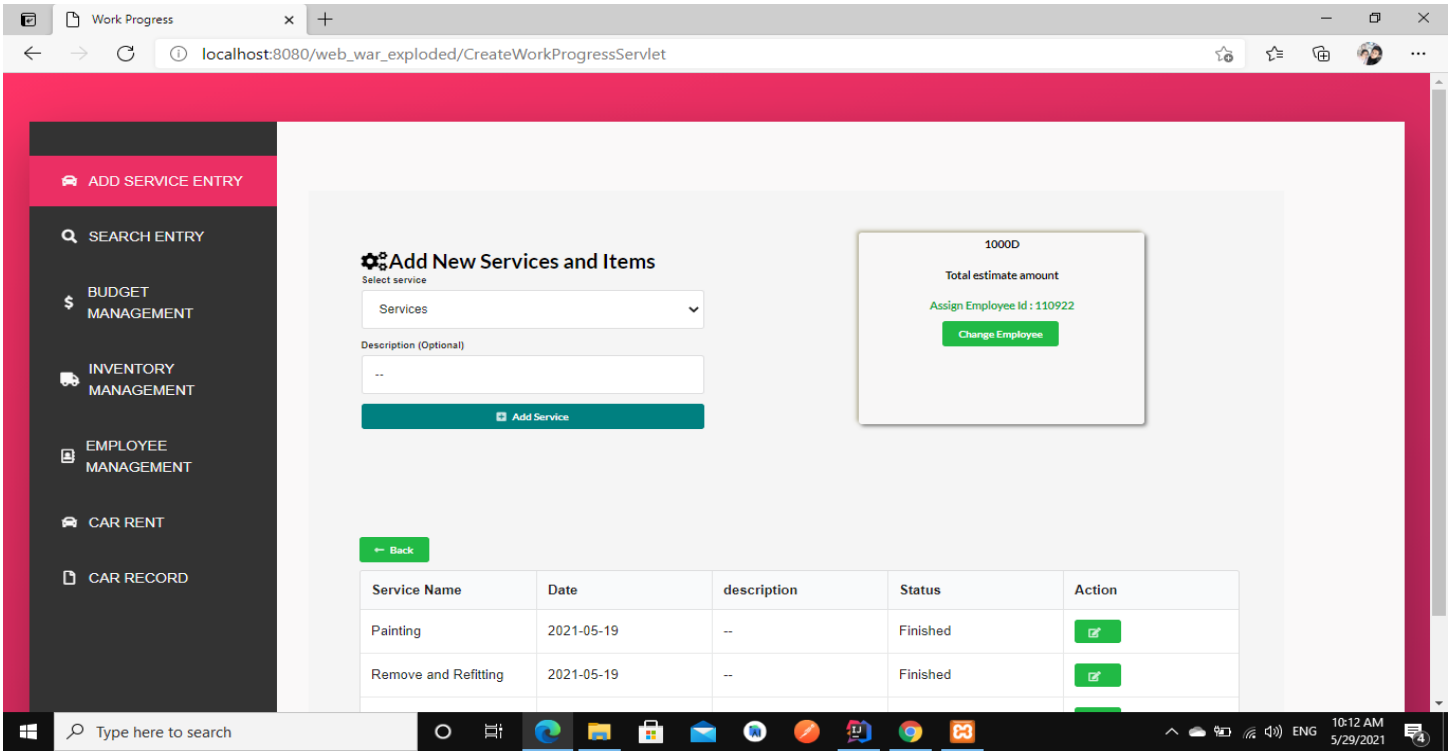


Figure 29: ***Assign employee***

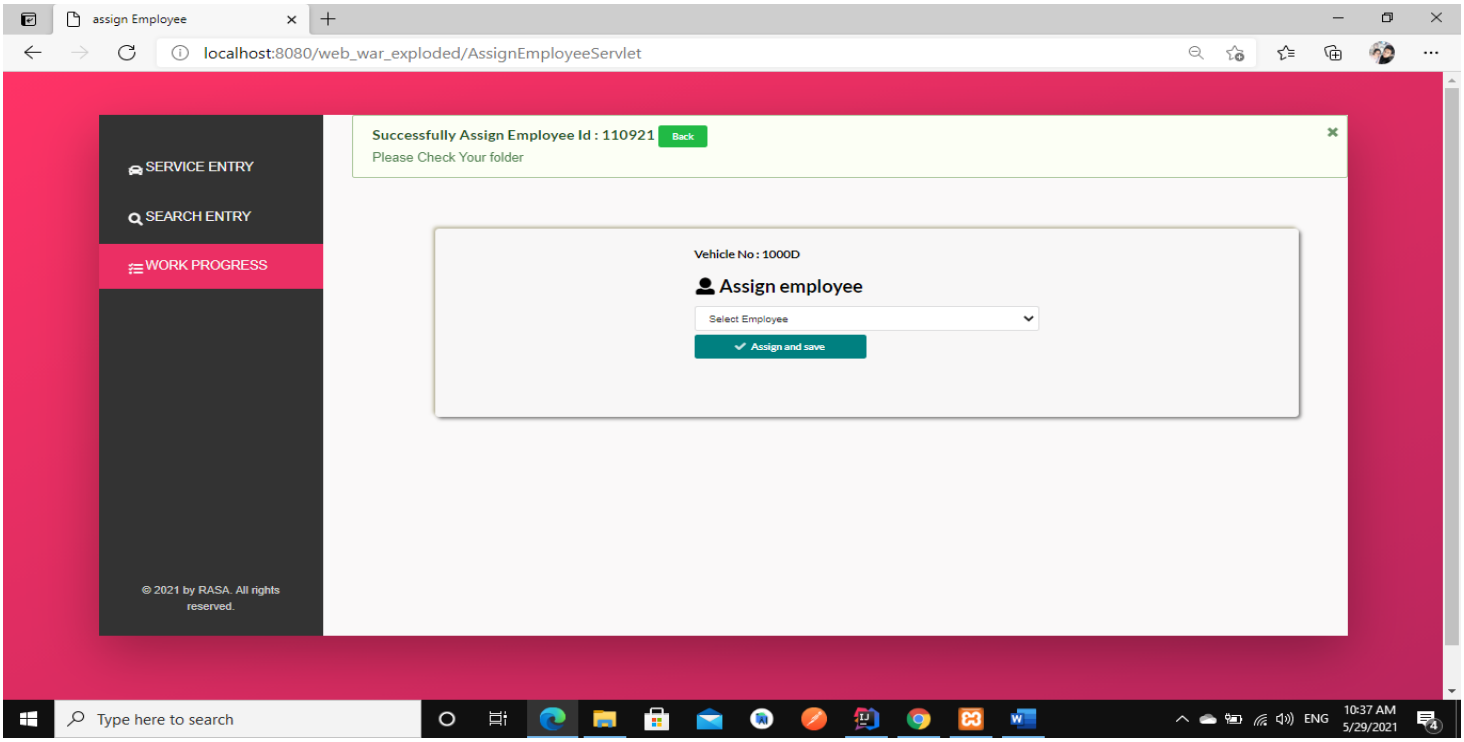


Figure 30: ***Add repair components***

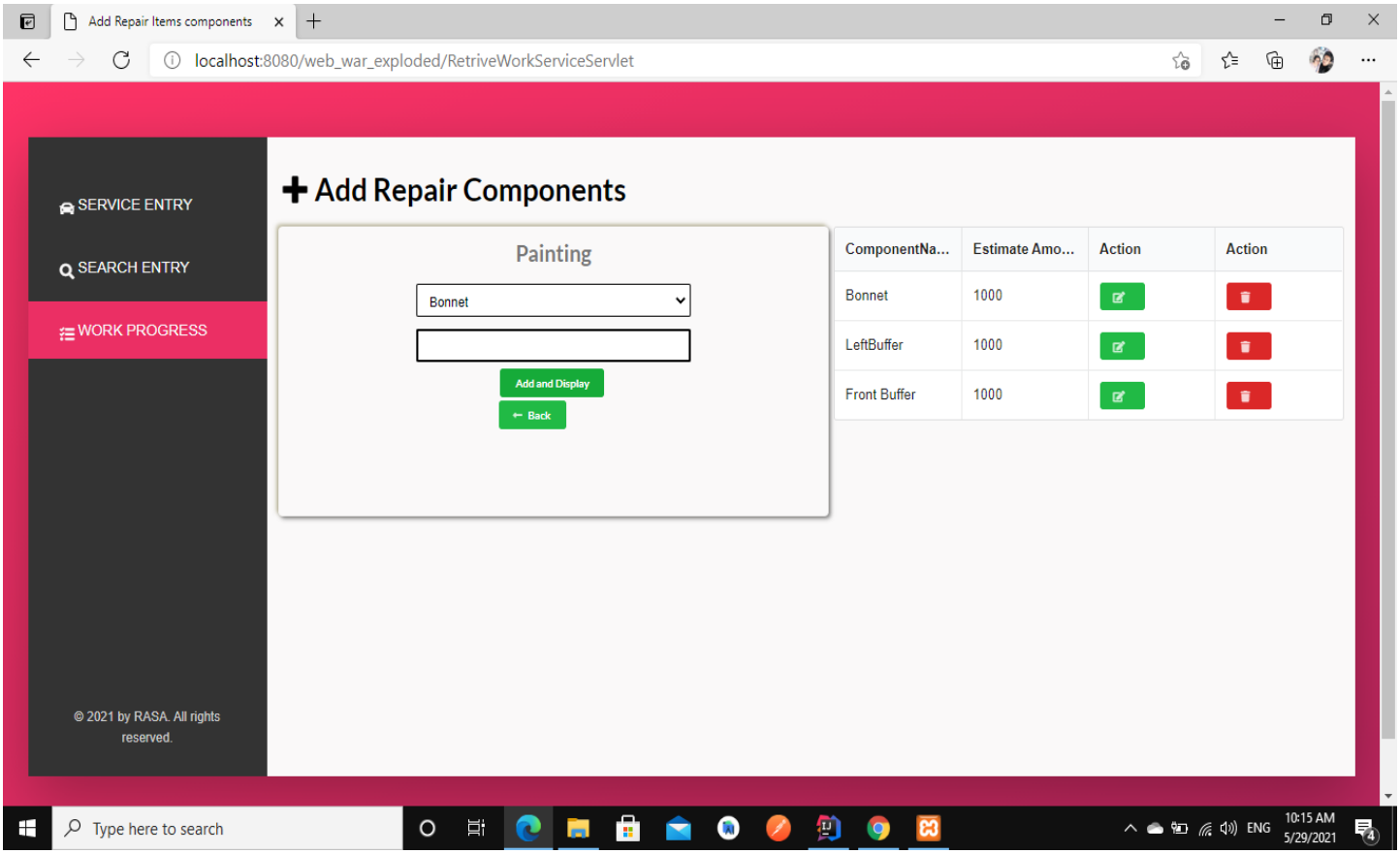


Figure 31: ***Estimate report overview***

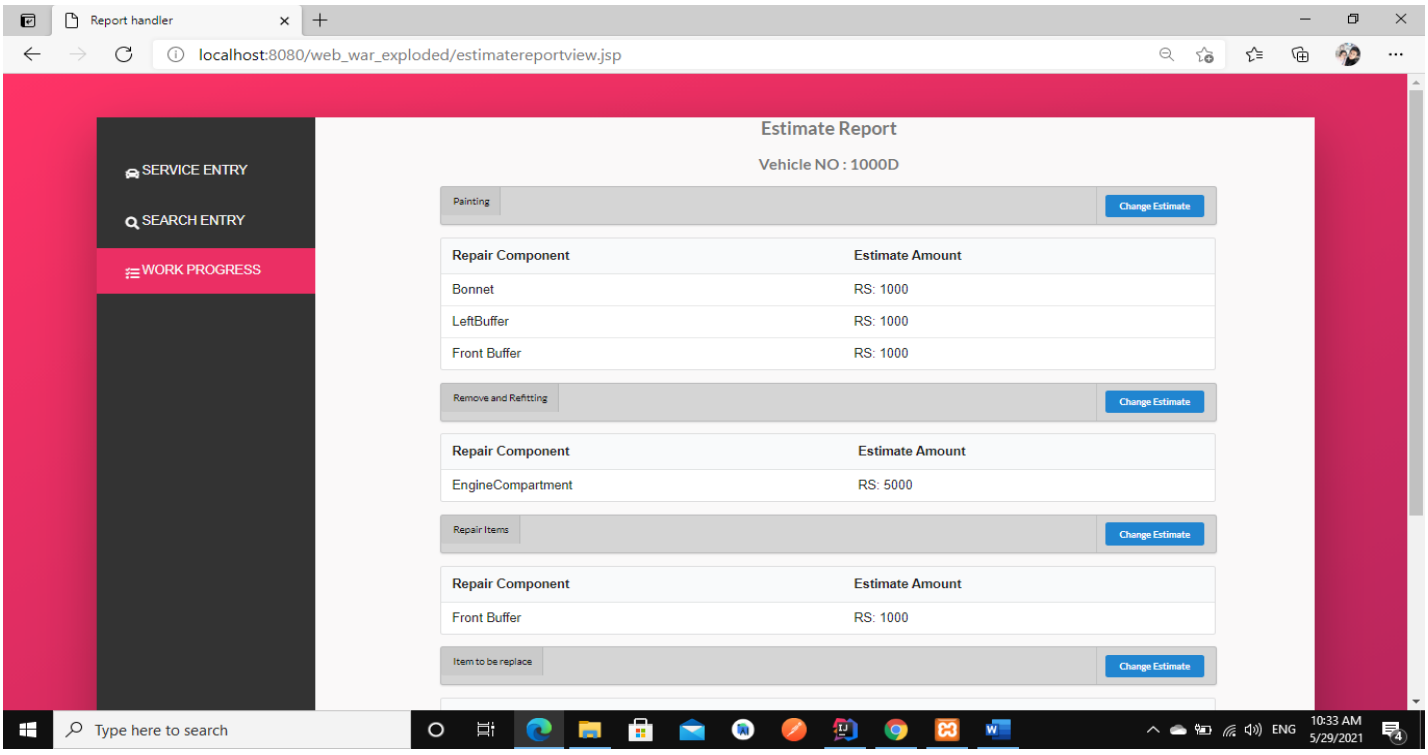
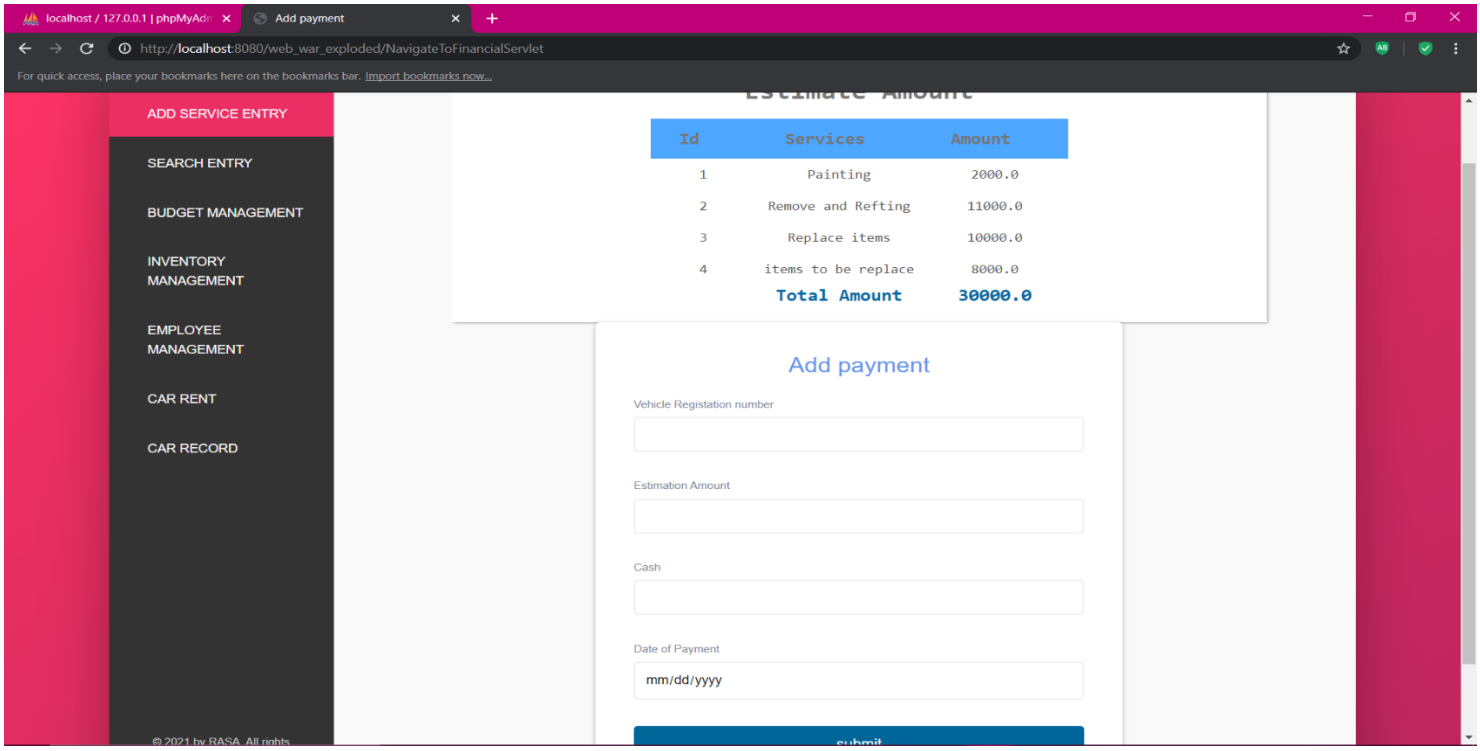


Figure 32: ***Financial Management***



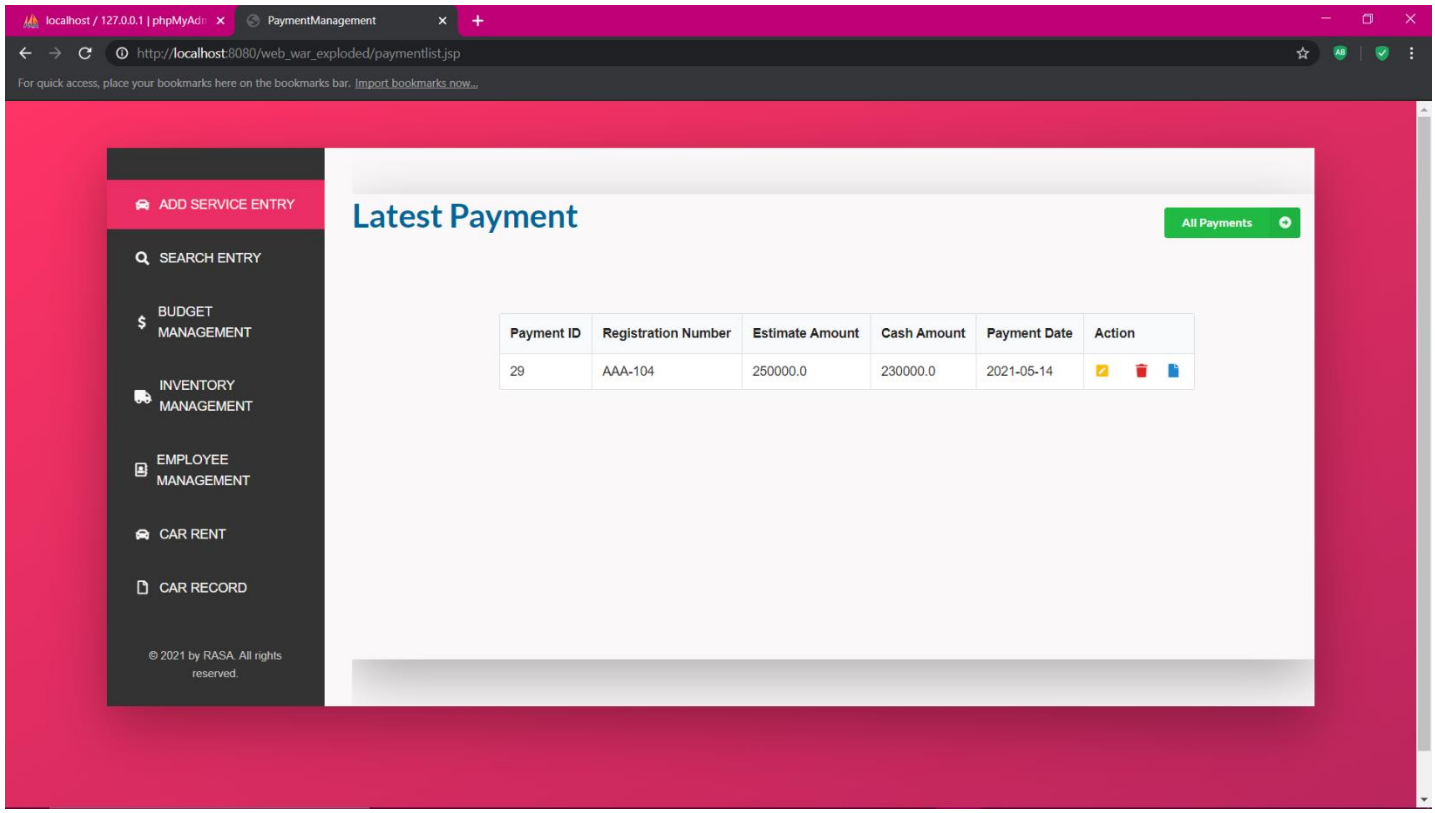


Figure 33: Latest Payments

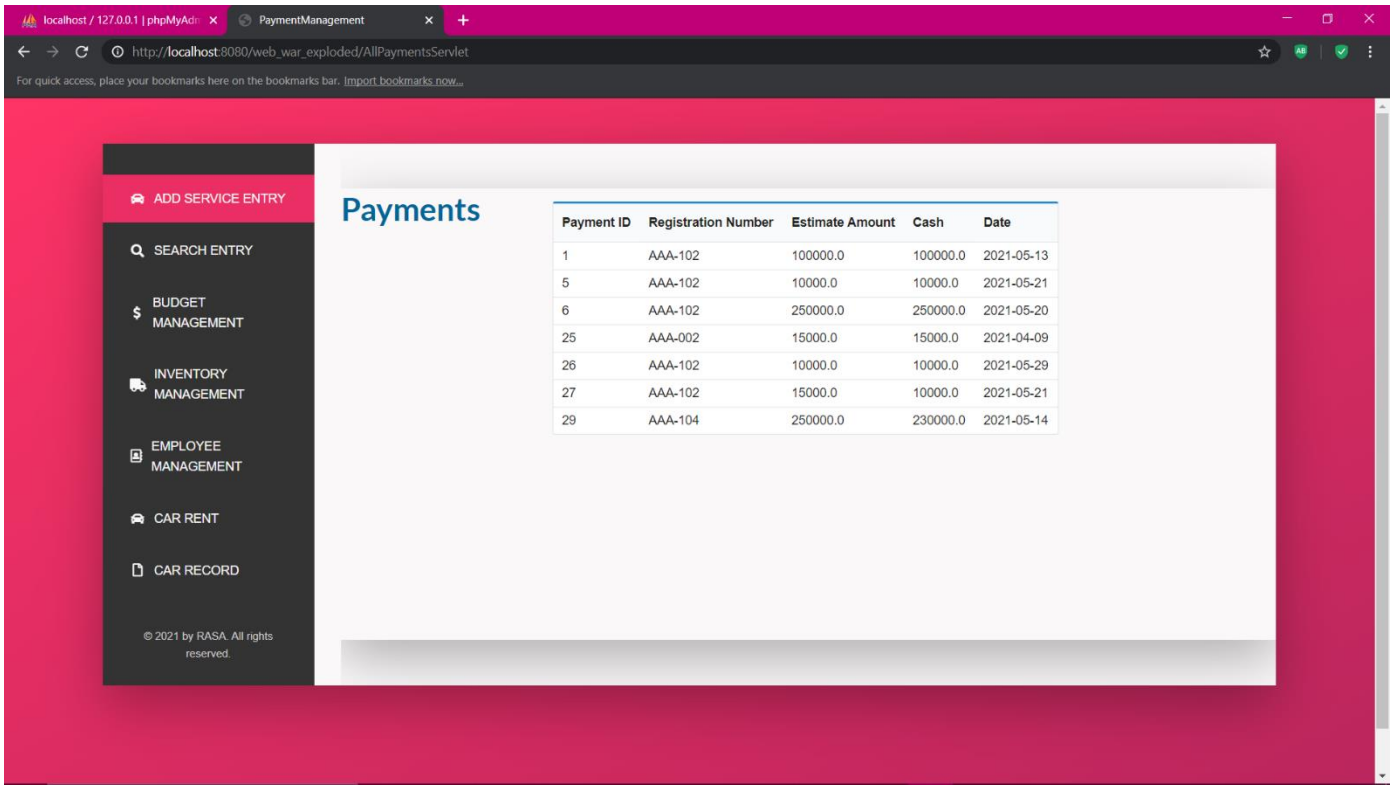


Figure 34: All payments

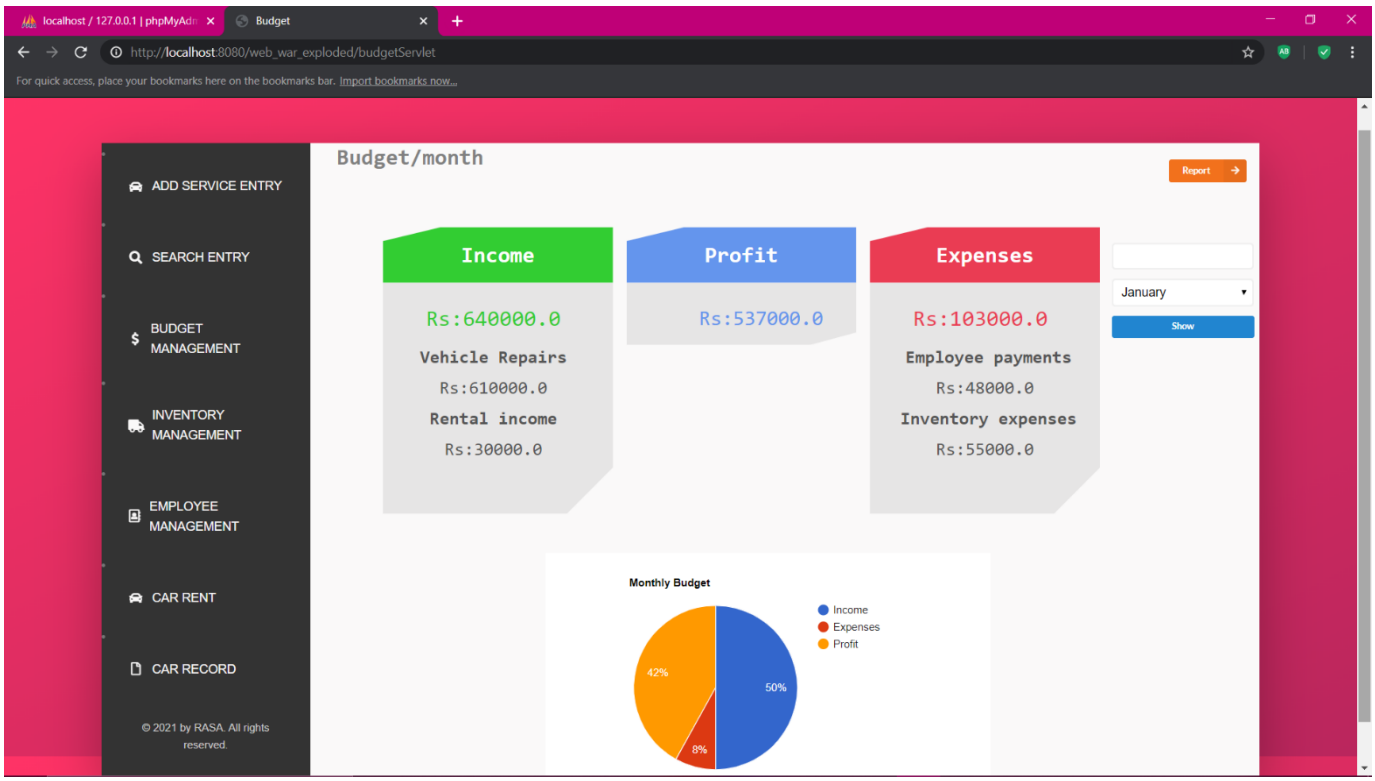


Figure 35: Budget Ui

Figure 36: ***Car Records retrieve***

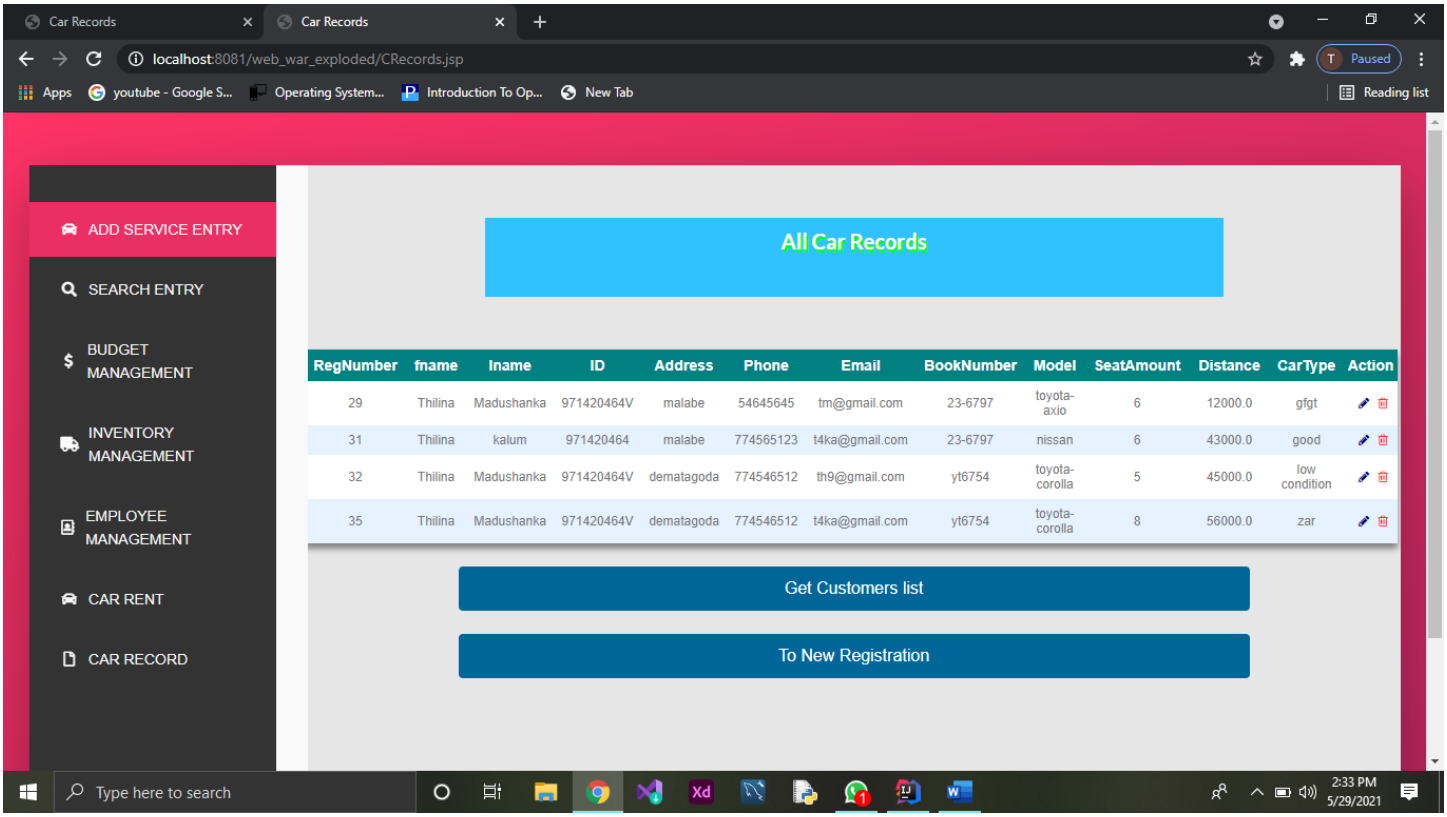
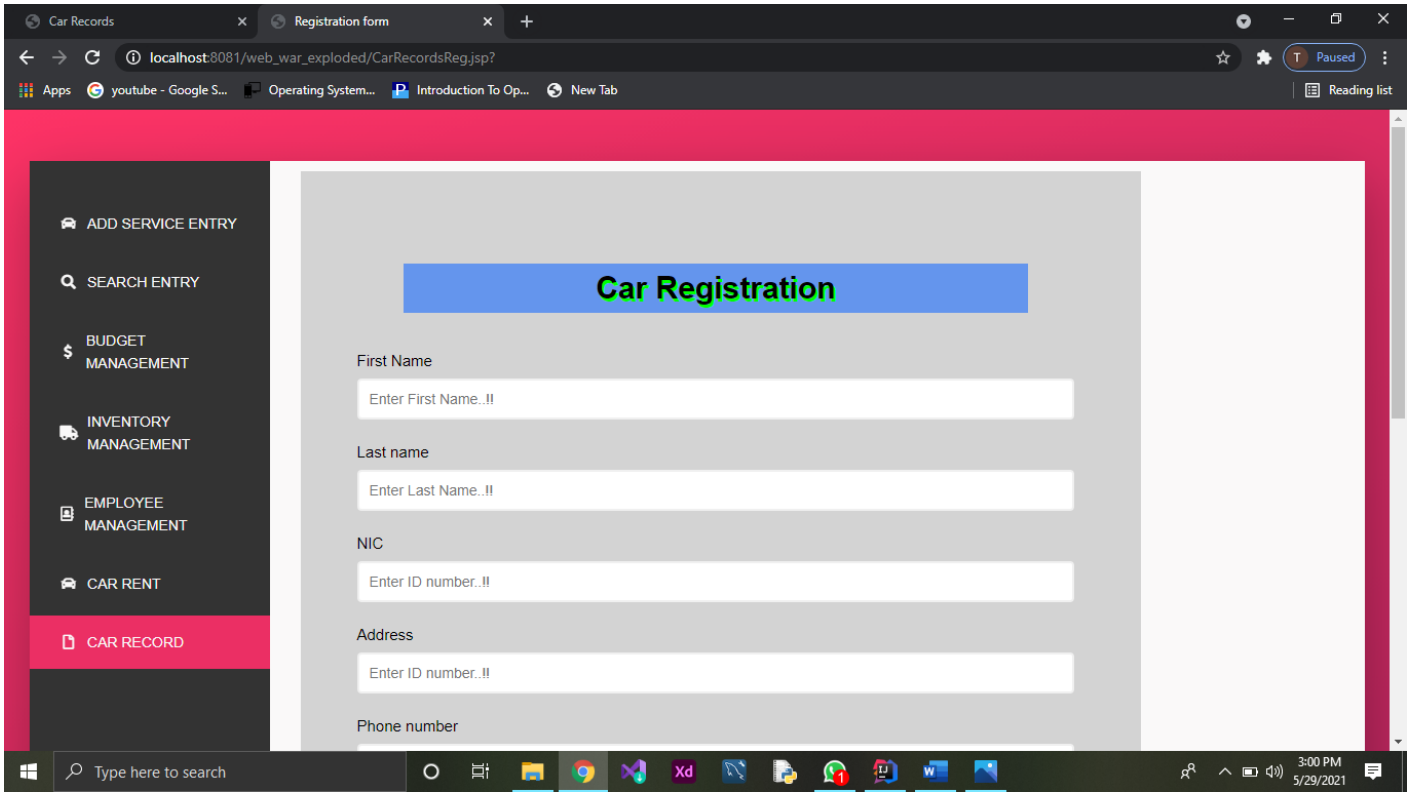
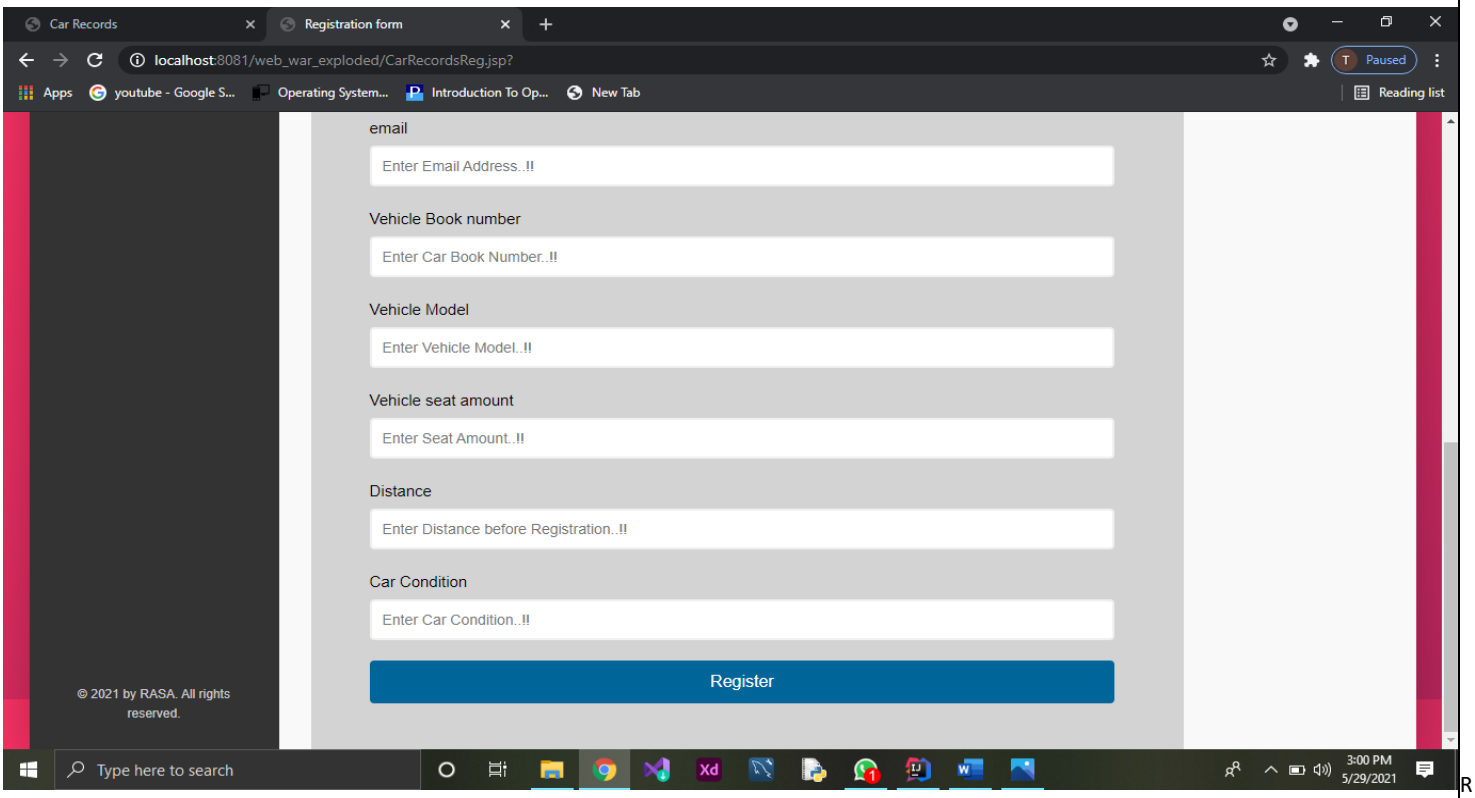


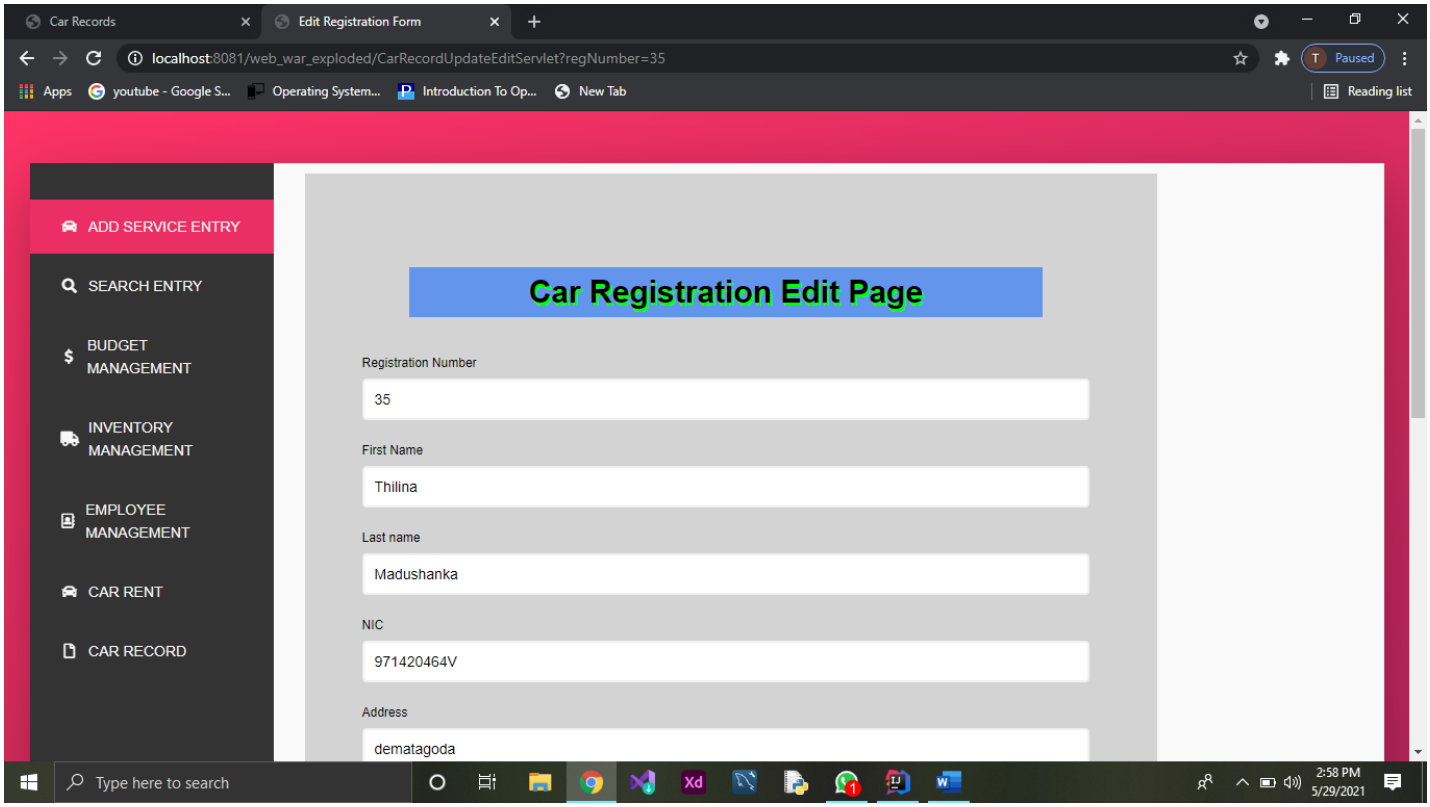
Figure 37: ***Car Records Register***


Car Records
Registration form
localhost:8081/web_war_exploded/CarRecordsReg.jsp?
Paused
Apps
youtube - Google S...
Operating System...
Introduction To Op...
New Tab
Reading list
ADD SERVICE ENTRY
SEARCH ENTRY
BUDGET MANAGEMENT
INVENTORY MANAGEMENT
EMPLOYEE MANAGEMENT
CAR RENT
CAR RECORD
Car Registration
First Name
Enter First Name..!!
Last name
Enter Last Name..!!
NIC
Enter ID number..!!
Address
Enter ID number..!!
Phone number
Type here to search
3:00 PM
5/29/2021


Car Records
Registration form
localhost:8081/web_war_exploded/CarRecordsReg.jsp?
Paused
Apps
youtube - Google S...
Operating System...
Introduction To Op...
New Tab
Reading list
email
Enter Email Address..!!
Vehicle Book number
Enter Car Book Number..!!
Vehicle Model
Enter Vehicle Model..!!
Vehicle seat amount
Enter Seat Amount..!!
Distance
Enter Distance before Registration..!!
Car Condition
Enter Car Condition..!!
Register
© 2021 by RASA. All rights reserved.
Type here to search
3:00 PM
5/29/2021

R

Figure 39,40: Car Records Edit Jsp



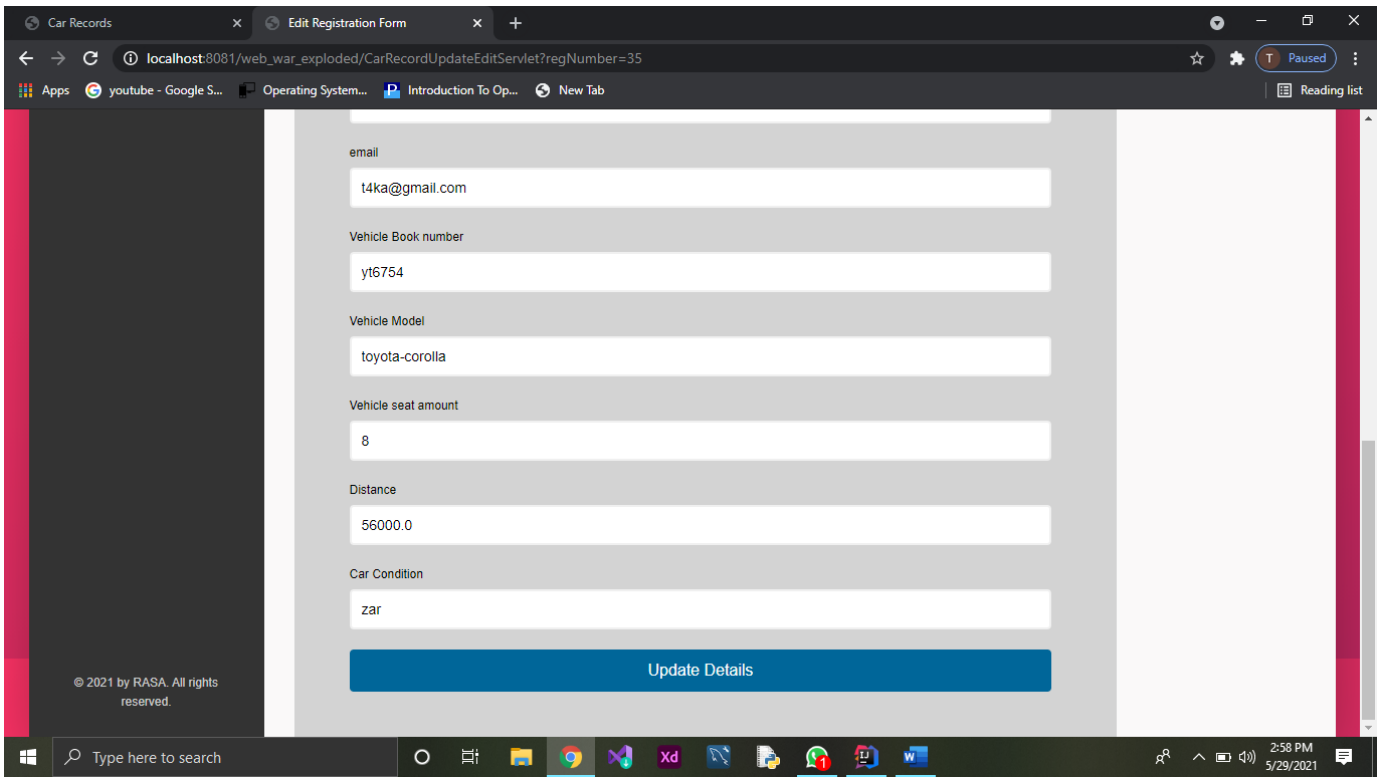


Figure 41: Available Cars

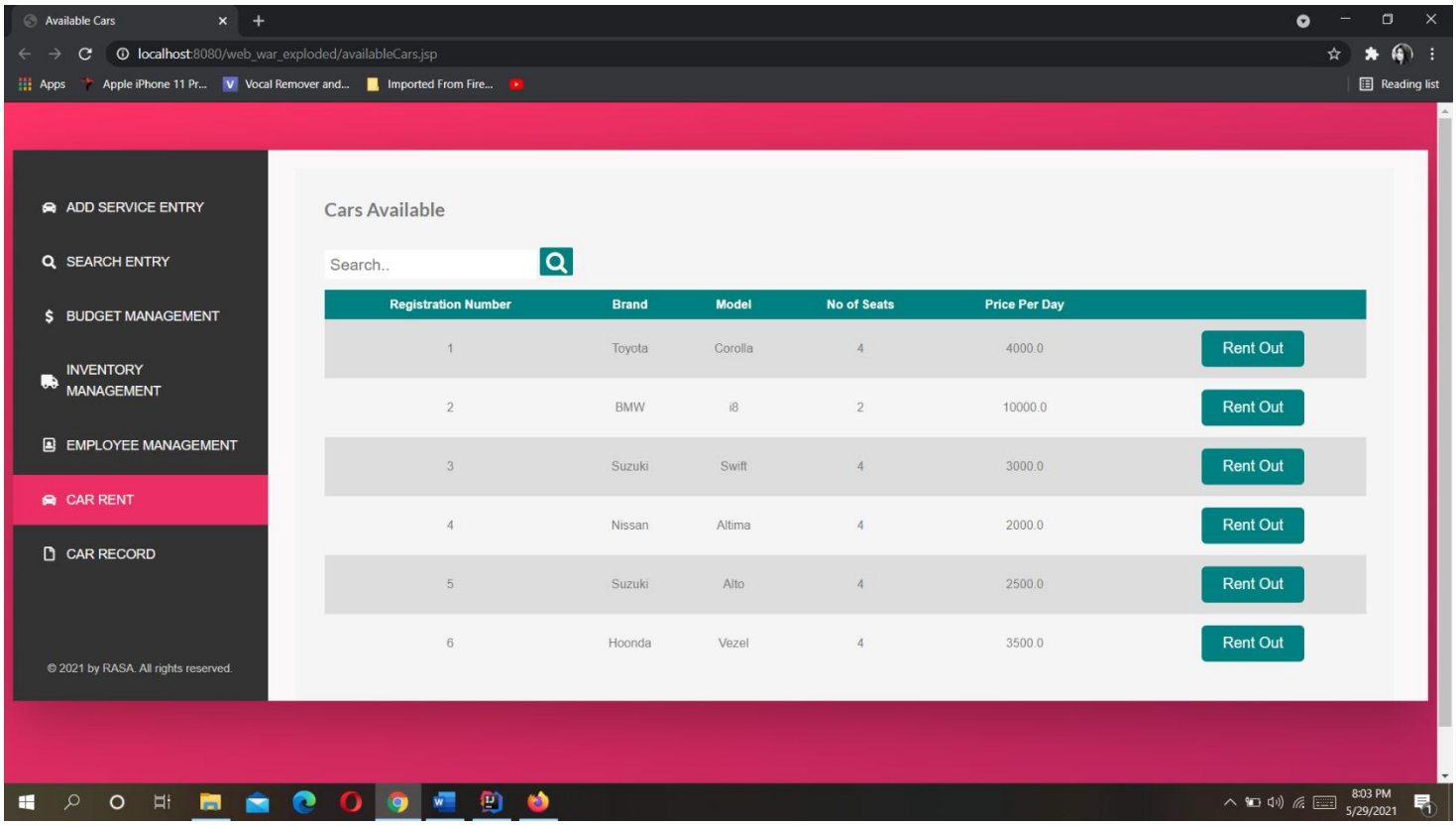


Figure 42: Rent Form

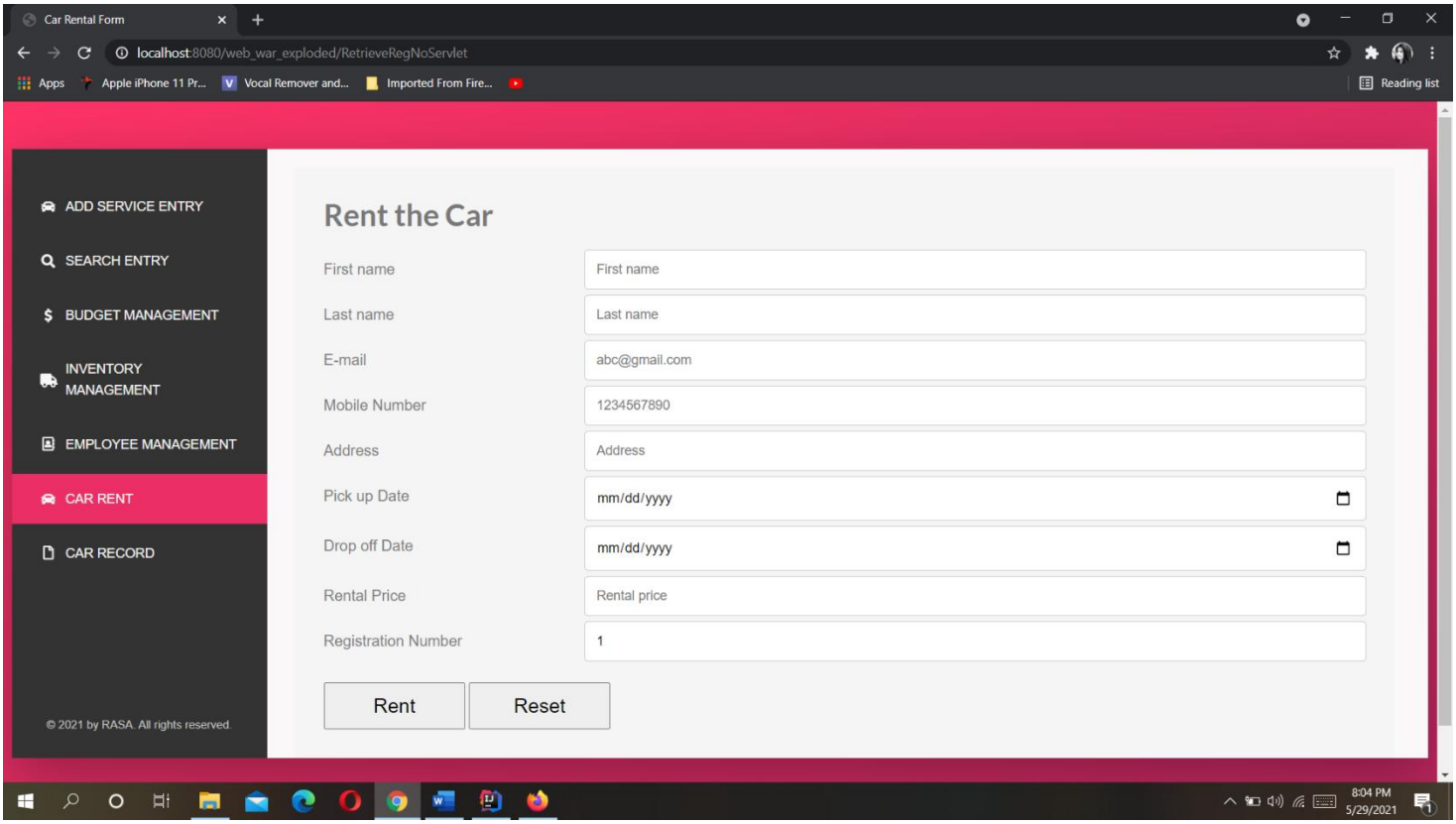


Figure 43: Rental Details

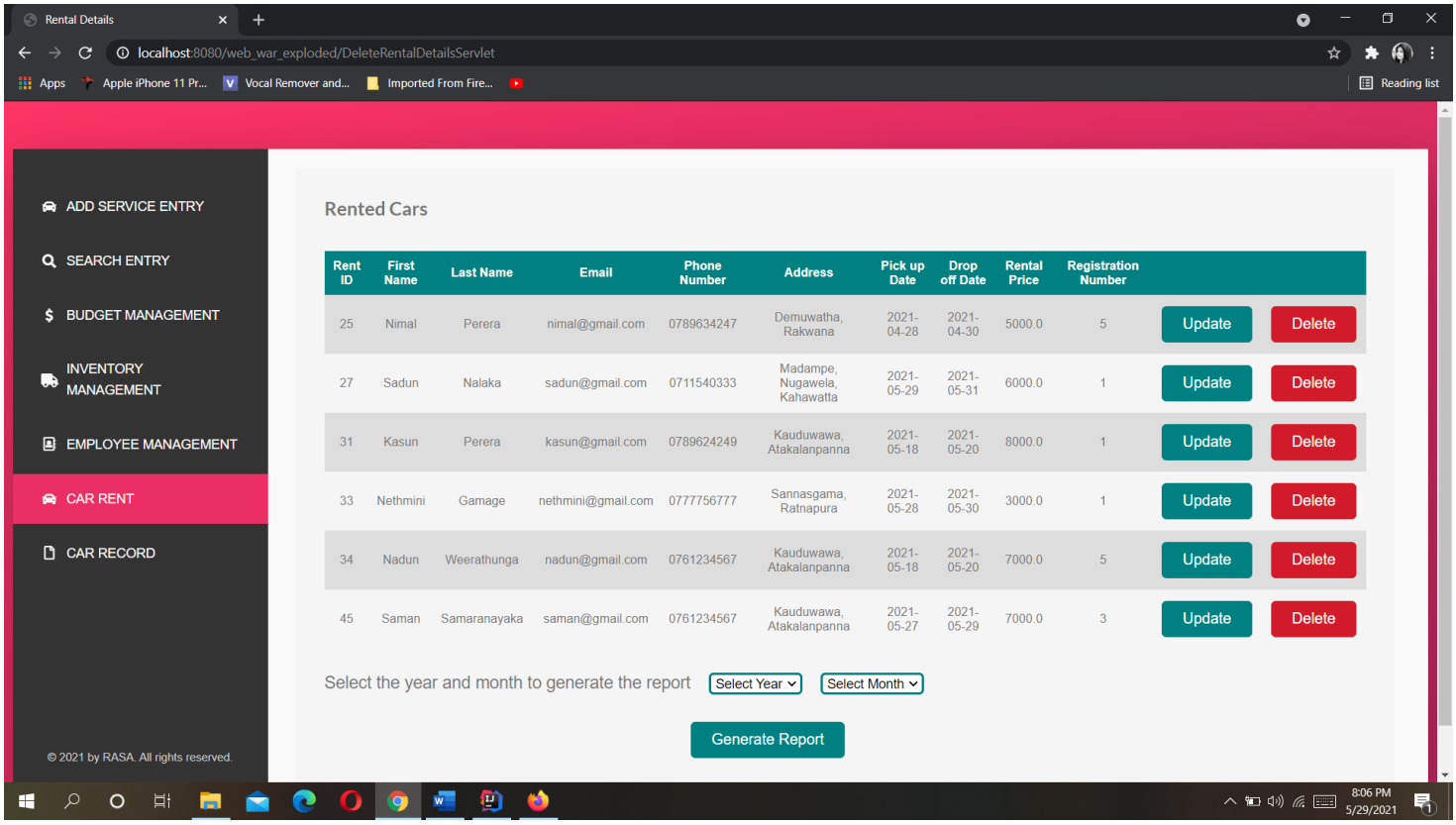


Figure 44: Update Rental Details

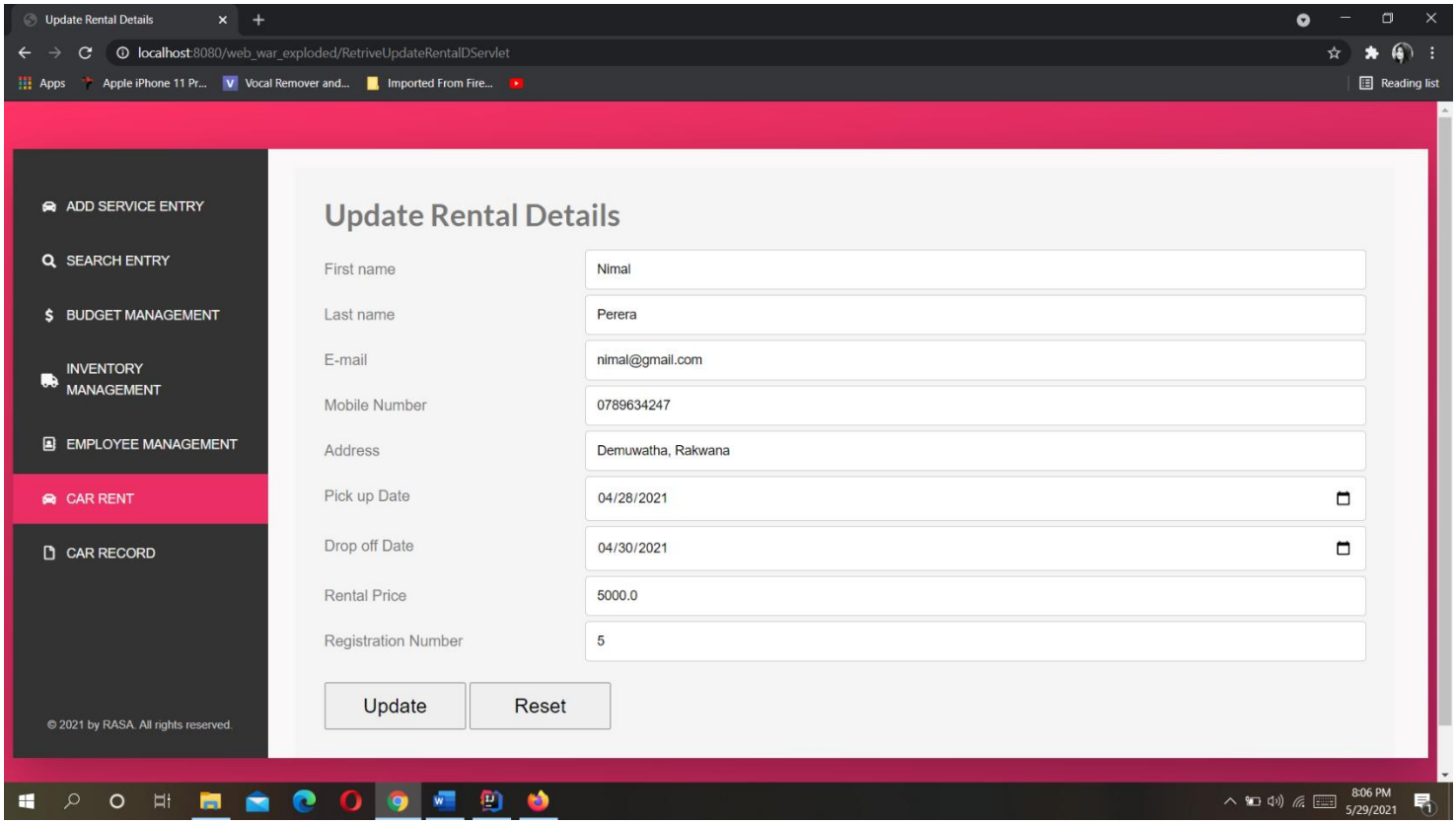


Figure 45: employee registration

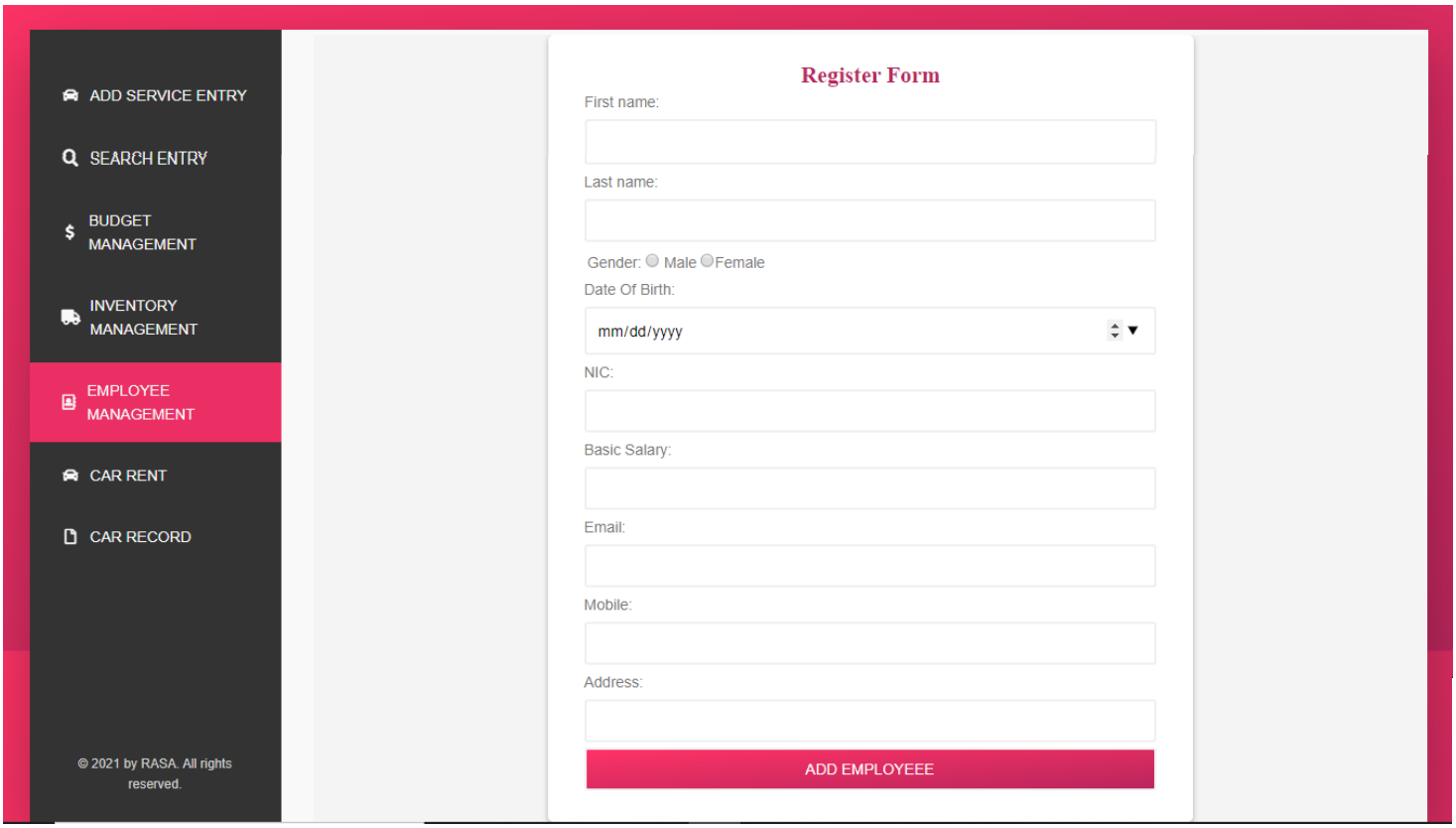


Figure 46: Employee details

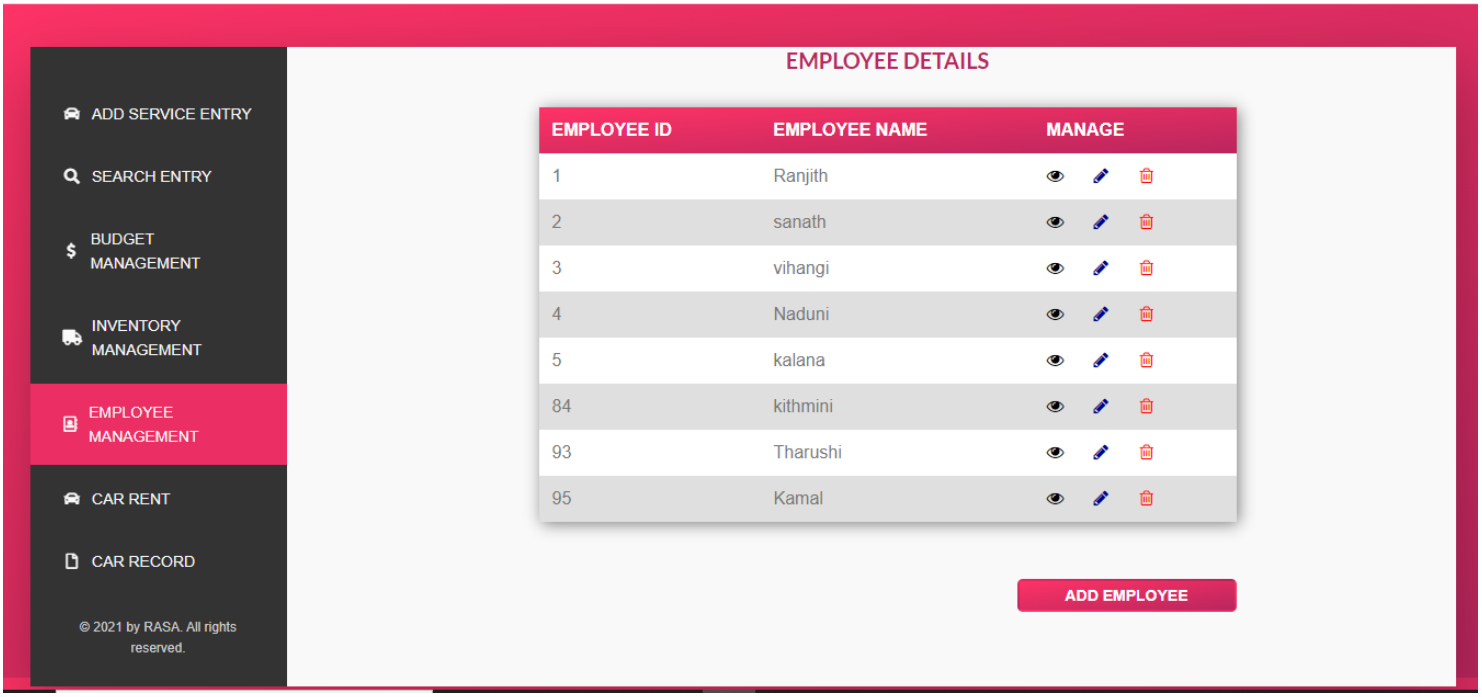


Figure 47: Employee update form

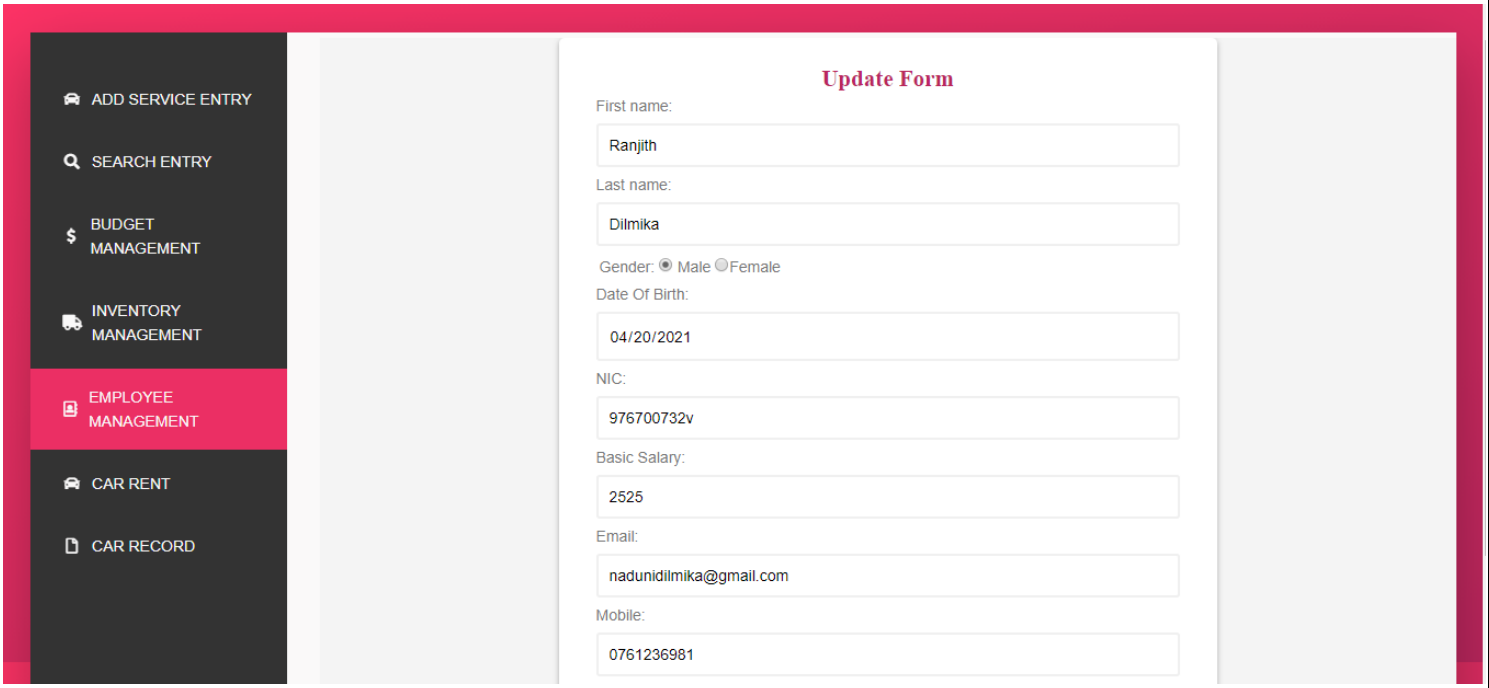


Figure 48: employee profile

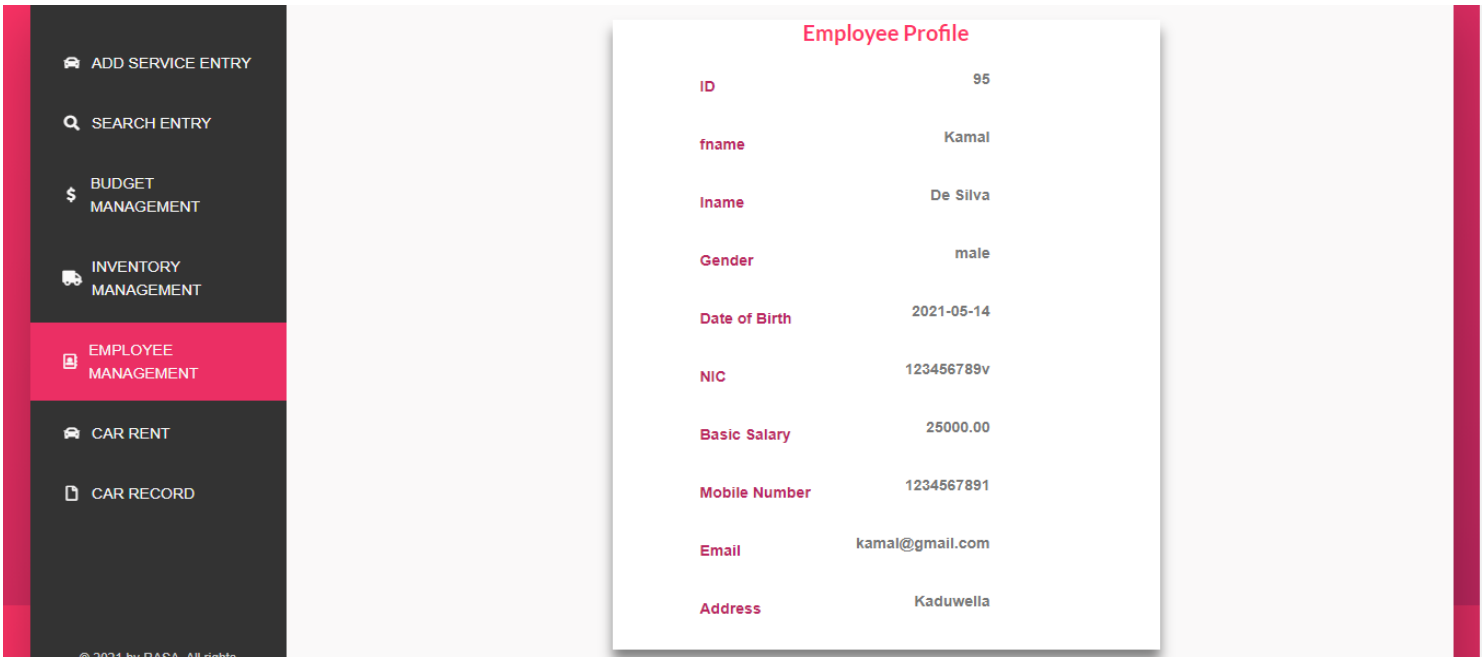


Figure 49: employee salary and attendance management

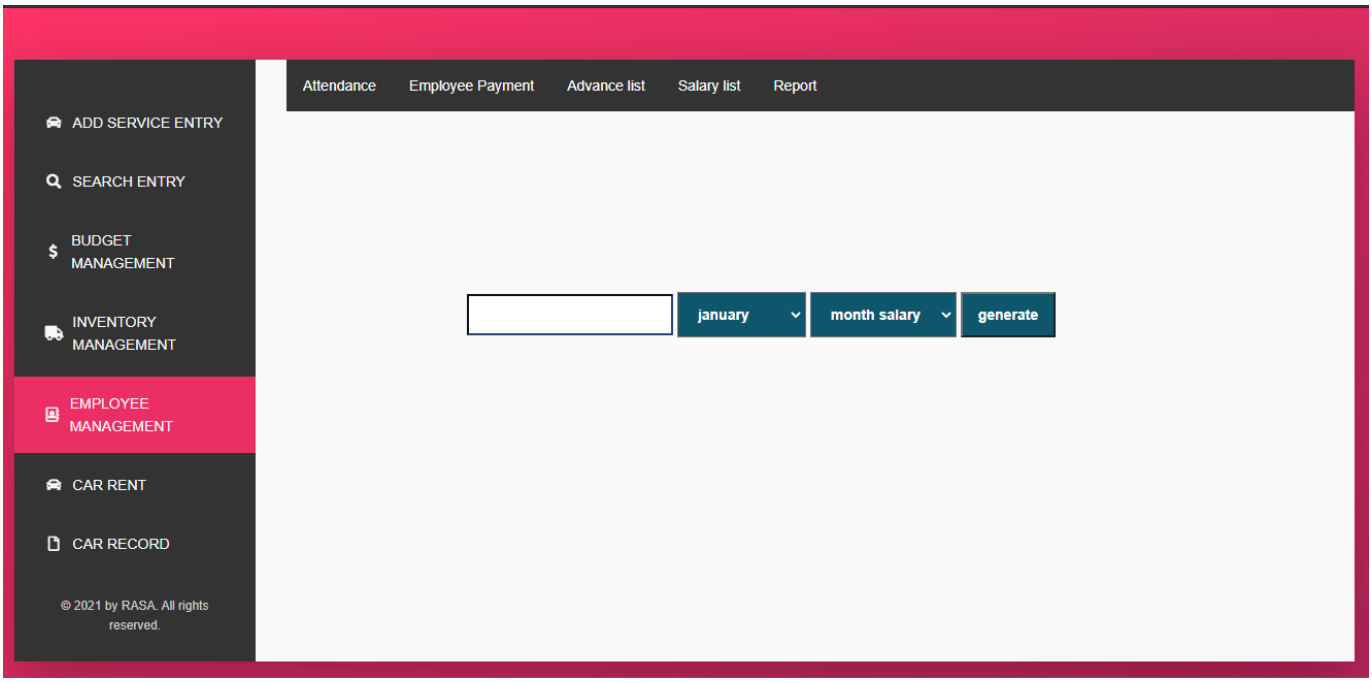


Figure 50: latest advance record

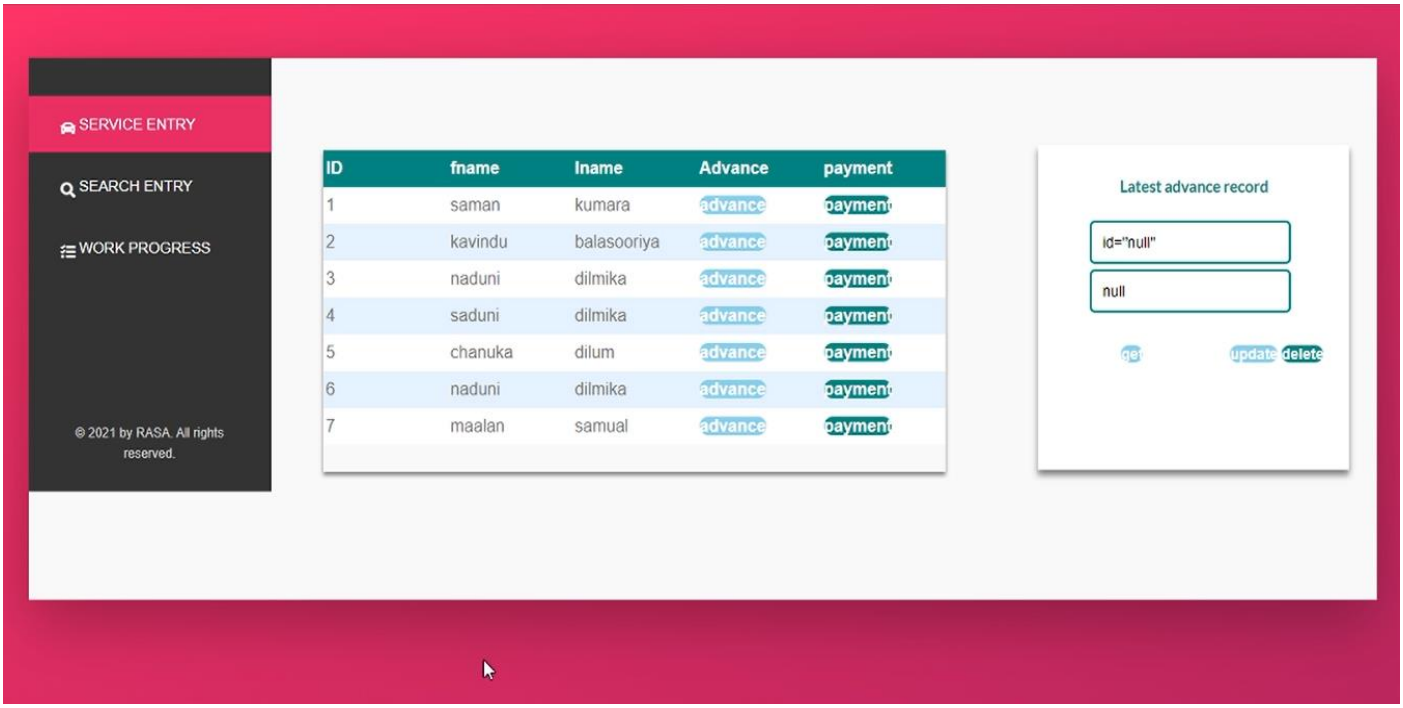
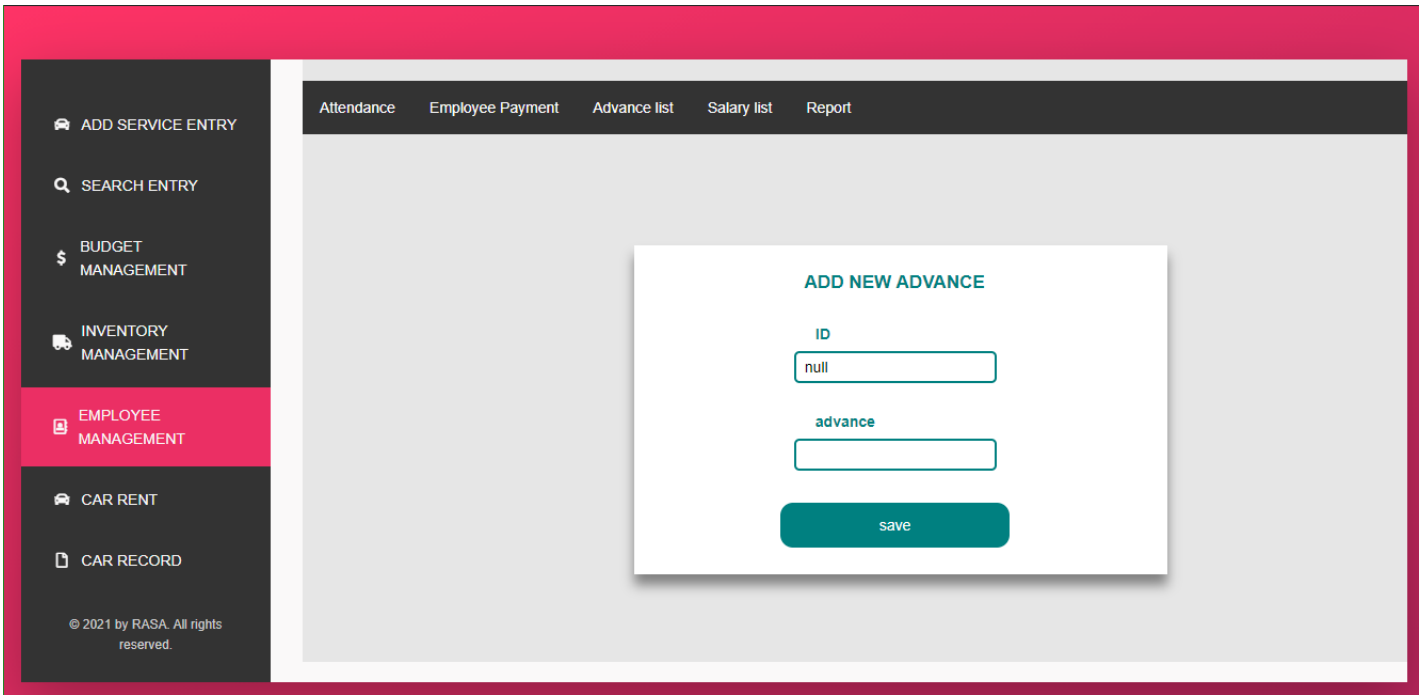
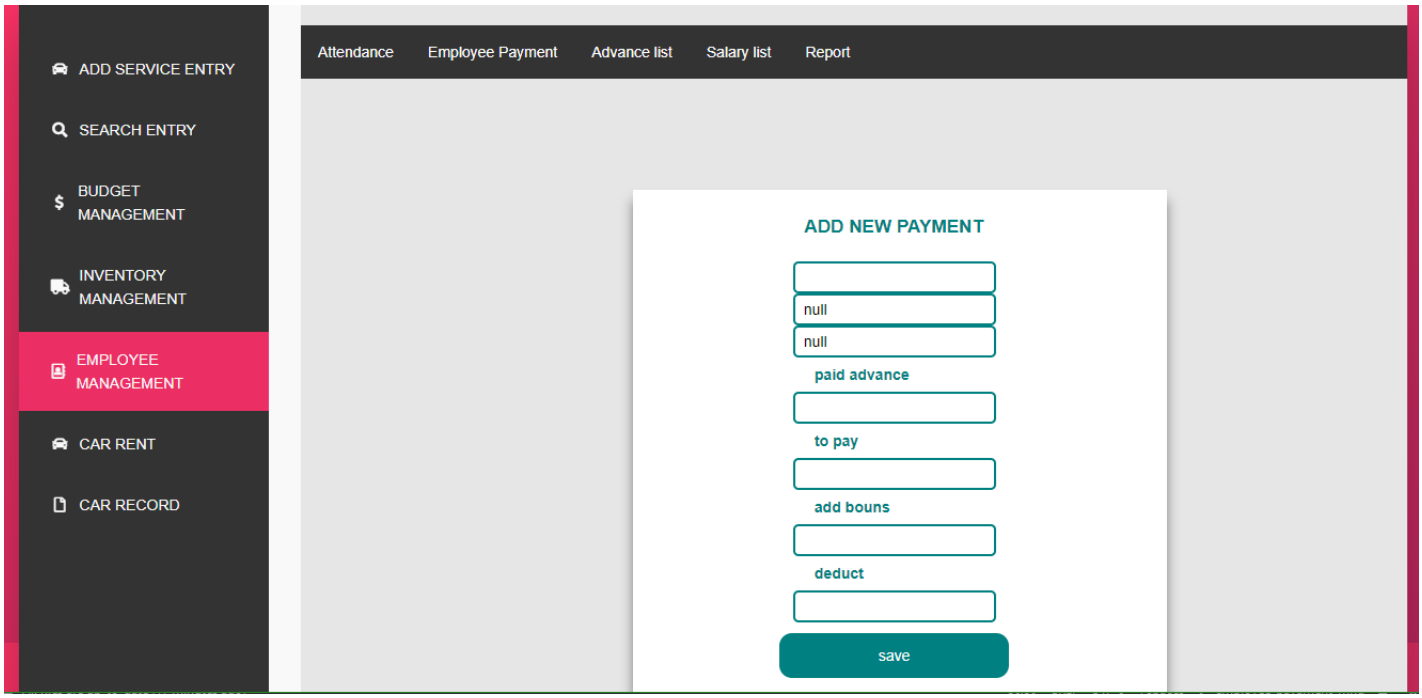


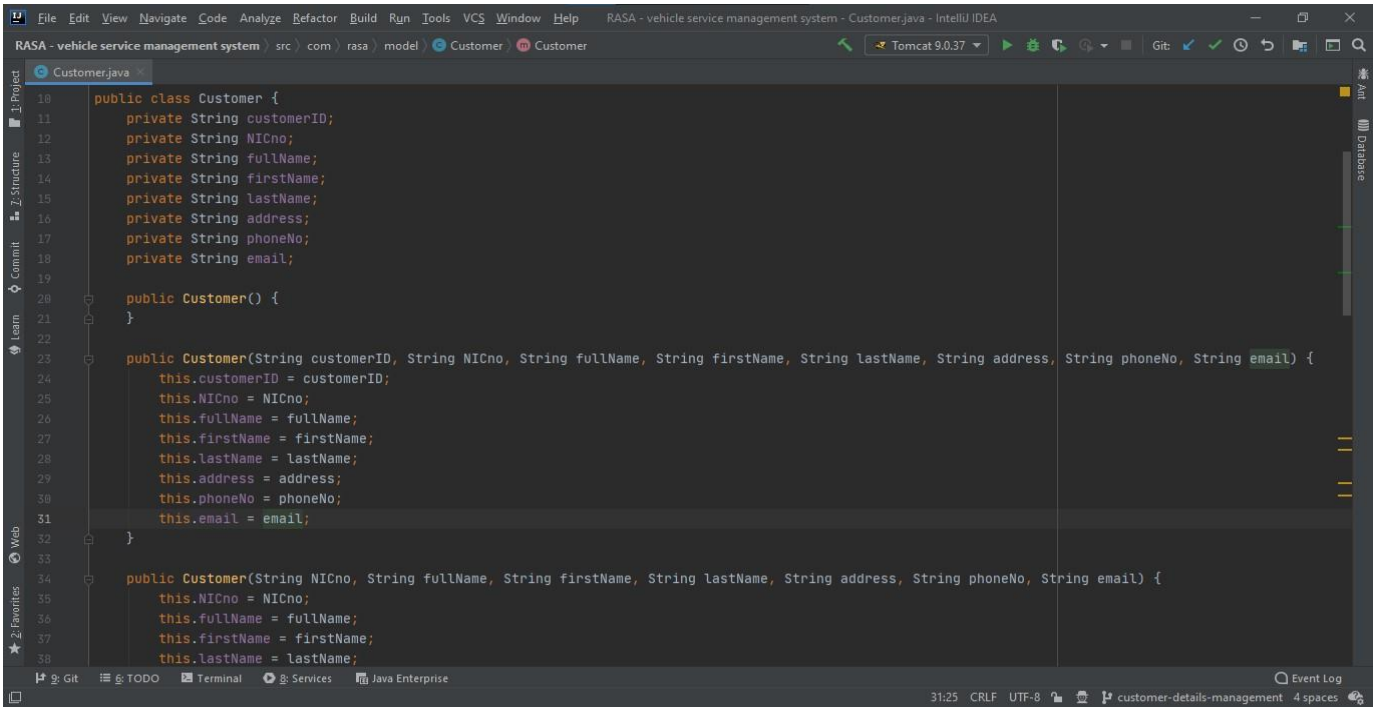
Figure 51: add Advance





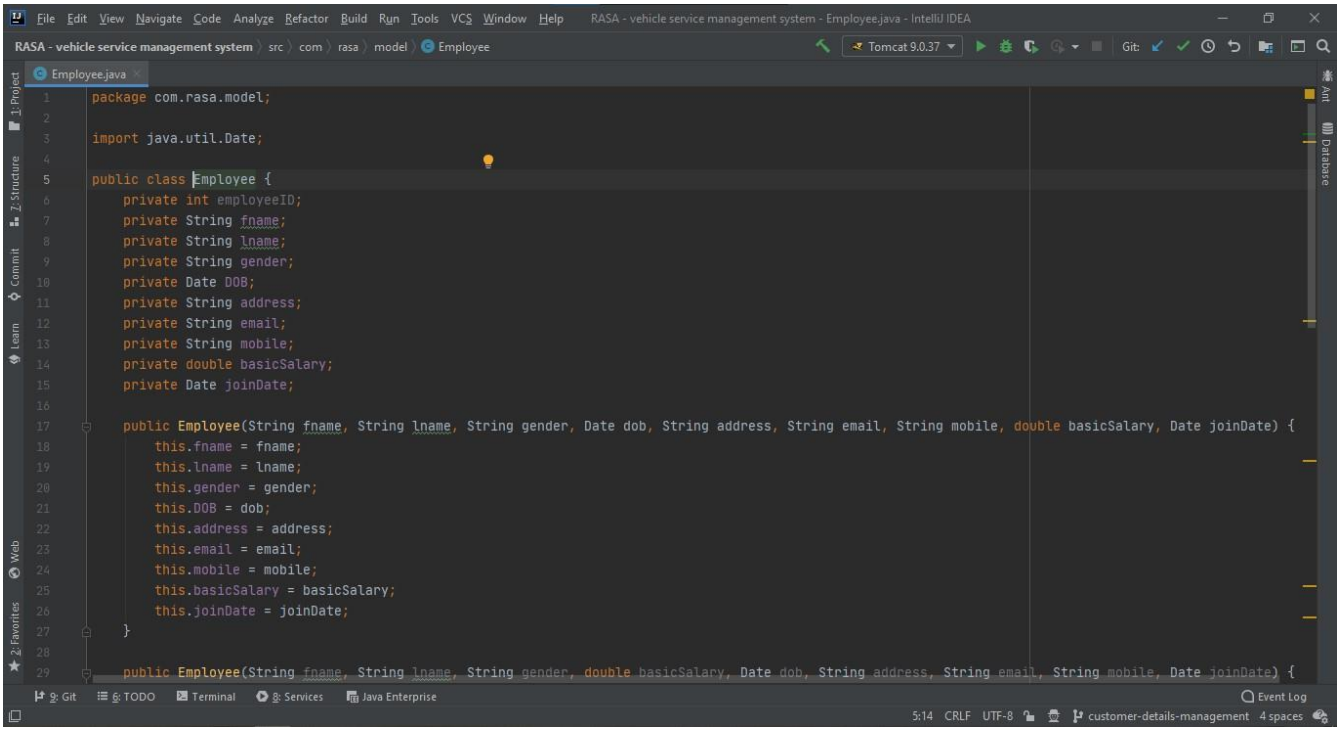
Back- End Development

Model Classes



```java
public class Customer {
    private String customerID;
    private String NICno;
    private String fullName;
    private String firstName;
    private String lastName;
    private String address;
    private String phoneNo;
    private String email;

    public Customer() {
    }

    public Customer(String customerID, String NICno, String fullName, String firstName, String lastName, String address, String phoneNo, String email) {
        this.customerID = customerID;
        this.NICno = NICno;
        this.fullName = fullName;
        this.firstName = firstName;
        this.lastName = lastName;
        this.address = address;
        this.phoneNo = phoneNo;
        this.email = email;
    }

    public Customer(String NICno, String fullName, String firstName, String lastName, String address, String phoneNo, String email) {
        this.NICno = NICno;
        this.fullName = fullName;
        this.firstName = firstName;
        this.lastName = lastName;
```

Figure 52: customer class



```java
package com.rasa.model;

import java.util.Date;

public class Employee {
    private int employeeID;
    private String fname;
    private String lname;
    private String gender;
    private Date DOB;
    private String address;
    private String email;
    private String mobile;
    private double basicSalary;
    private Date joinDate;

    public Employee(String fname, String lname, String gender, Date dob, String address, String email, String mobile, double basicSalary, Date joinDate) {
        this.fname = fname;
        this.lname = lname;
        this.gender = gender;
        this.DOB = dob;
        this.address = address;
        this.email = email;
        this.mobile = mobile;
        this.basicSalary = basicSalary;
        this.joinDate = joinDate;
    }

    public Employee(String fname, String lname, String gender, double basicSalary, Date dob, String address, String email, String mobile, Date joinDate) {
```

Figure 53: Employee class

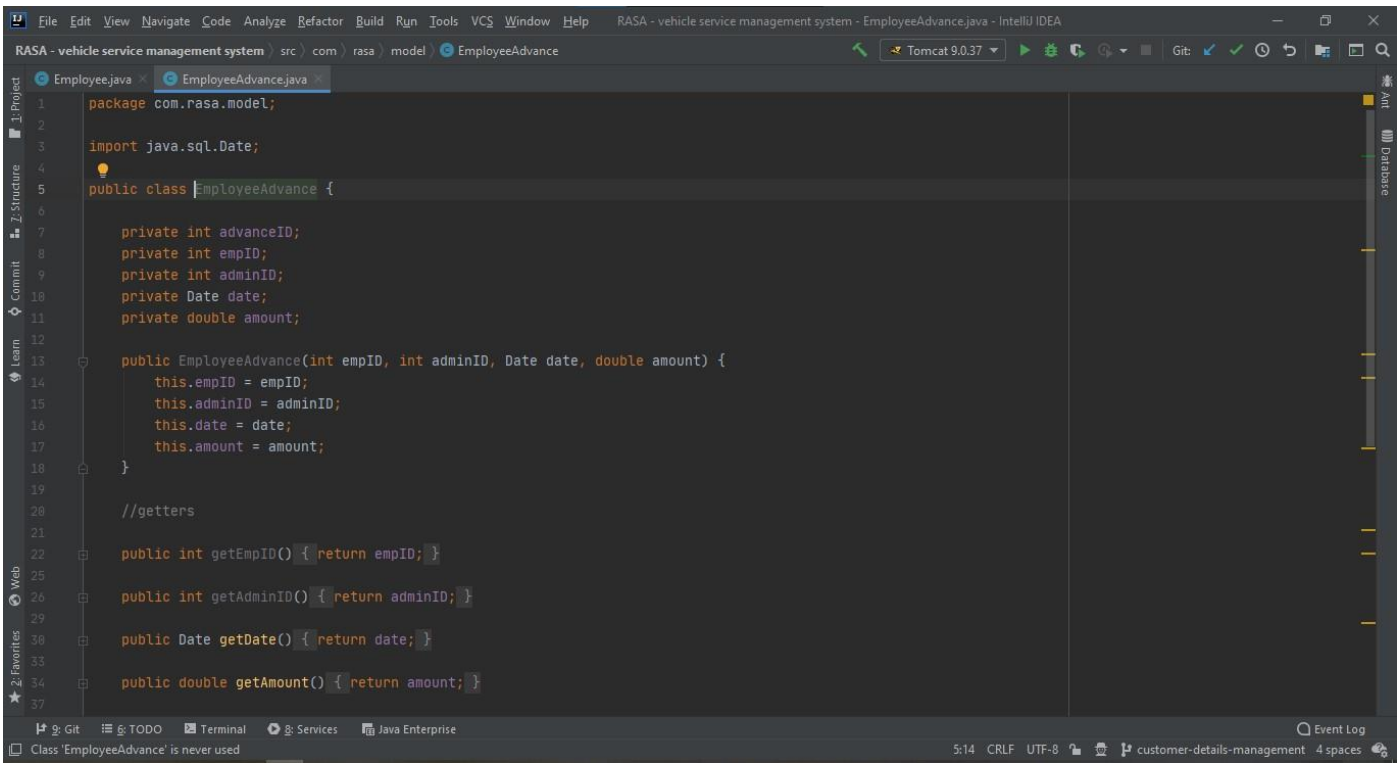


Figure 54: Employee Advance class

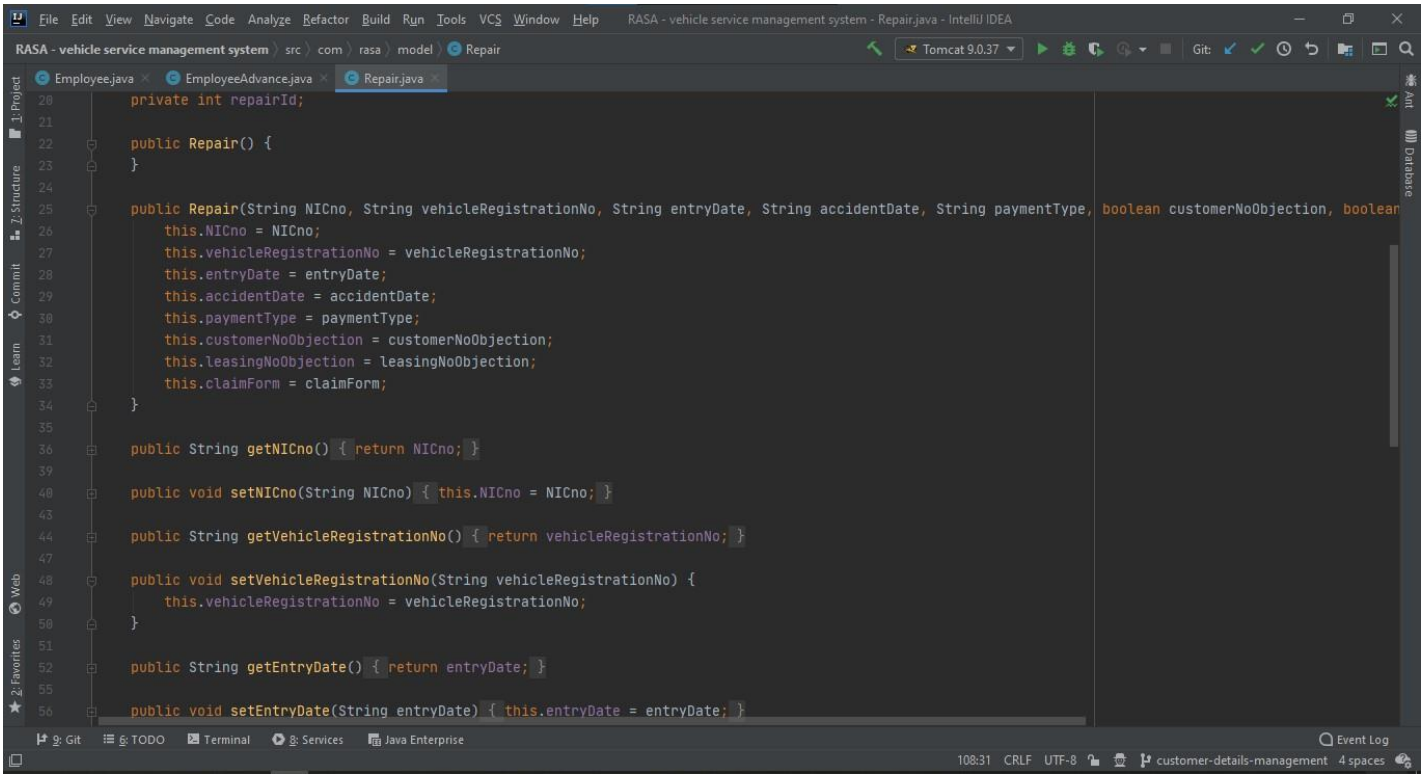
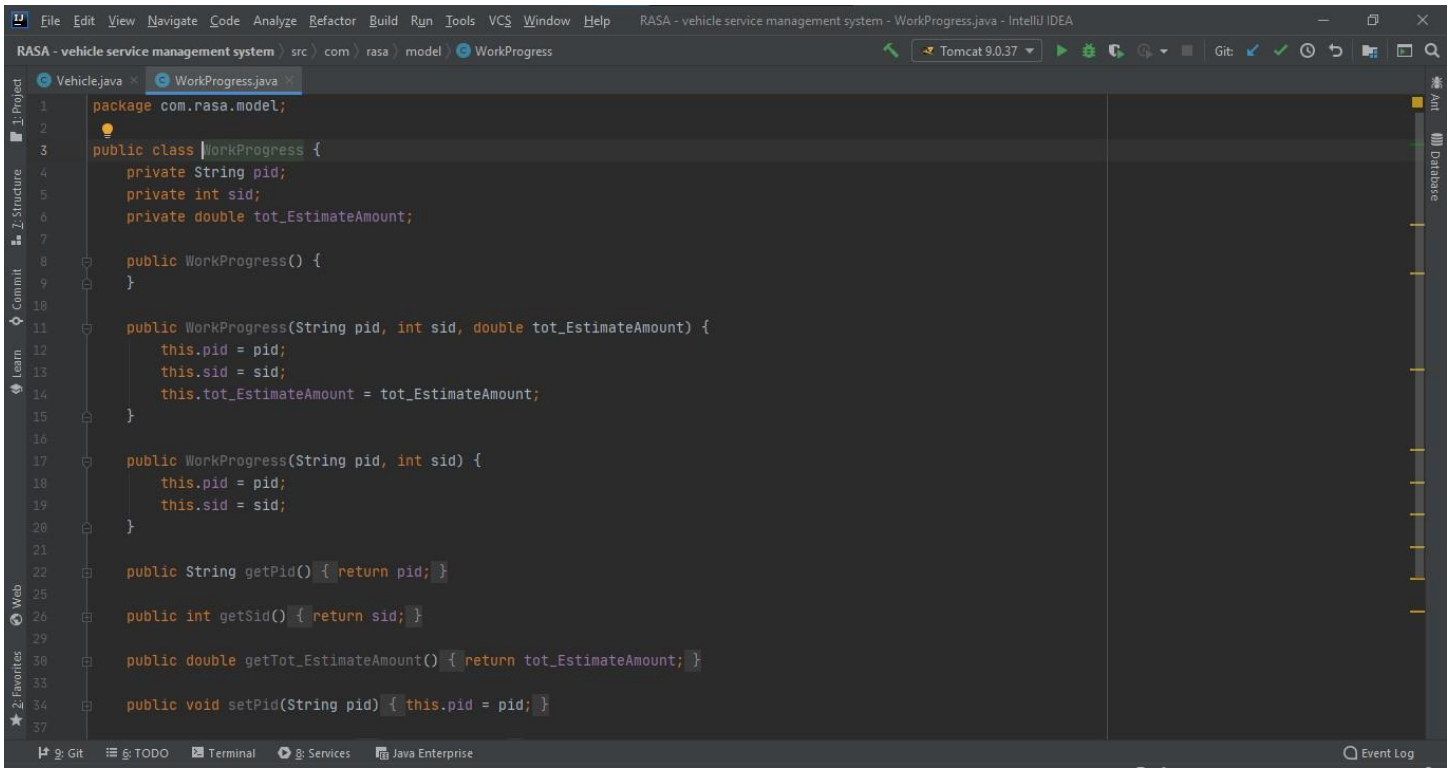
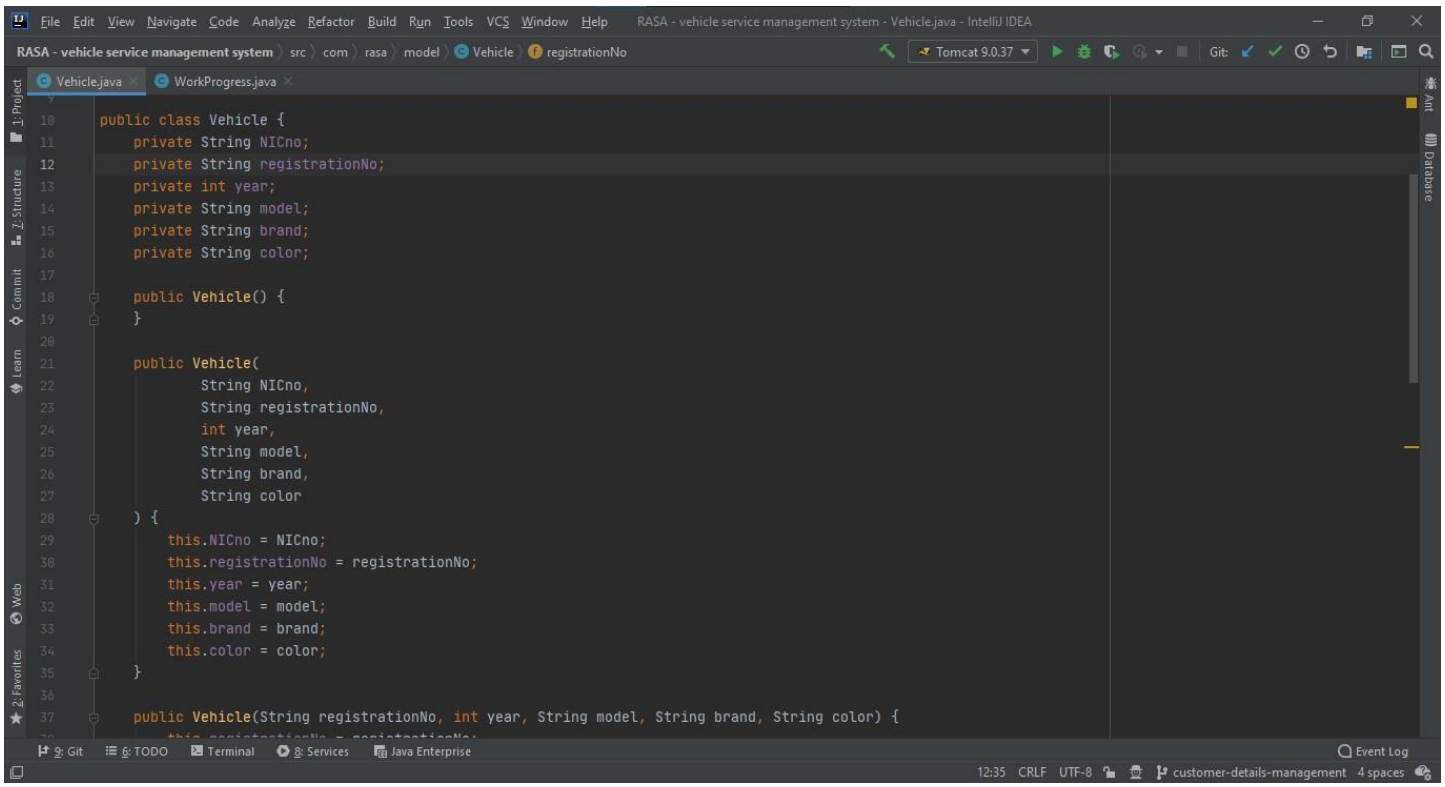


Figure 55: Repair class



```
package com.rasa.model;

public class WorkProgress {
    private String pid;
    private int sid;
    private double tot_EstimateAmount;

    public WorkProgress() {
    }

    public WorkProgress(String pid, int sid, double tot_EstimateAmount) {
        this.pid = pid;
        this.sid = sid;
        this.tot_EstimateAmount = tot_EstimateAmount;
    }

    public WorkProgress(String pid, int sid) {
        this.pid = pid;
        this.sid = sid;
    }

    public String getPid() { return pid; }

    public int getSid() { return sid; }

    public double getTot_EstimateAmount() { return tot_EstimateAmount; }

    public void setPid(String pid) { this.pid = pid; }
```

Figure 56: work progress class



```
public class Vehicle {
    private String NICno;
    private String registrationNo;
    private int year;
    private String model;
    private String brand;
    private String color;

    public Vehicle() {
    }

    public Vehicle(
            String NICno,
            String registrationNo,
            int year,
            String model,
            String brand,
            String color
    ) {
        this.NICno = NICno;
        this.registrationNo = registrationNo;
        this.year = year;
        this.model = model;
        this.brand = brand;
        this.color = color;
    }

    public Vehicle(String registrationNo, int year, String model, String brand, String color) {
```

Figure 57: Vehicle class

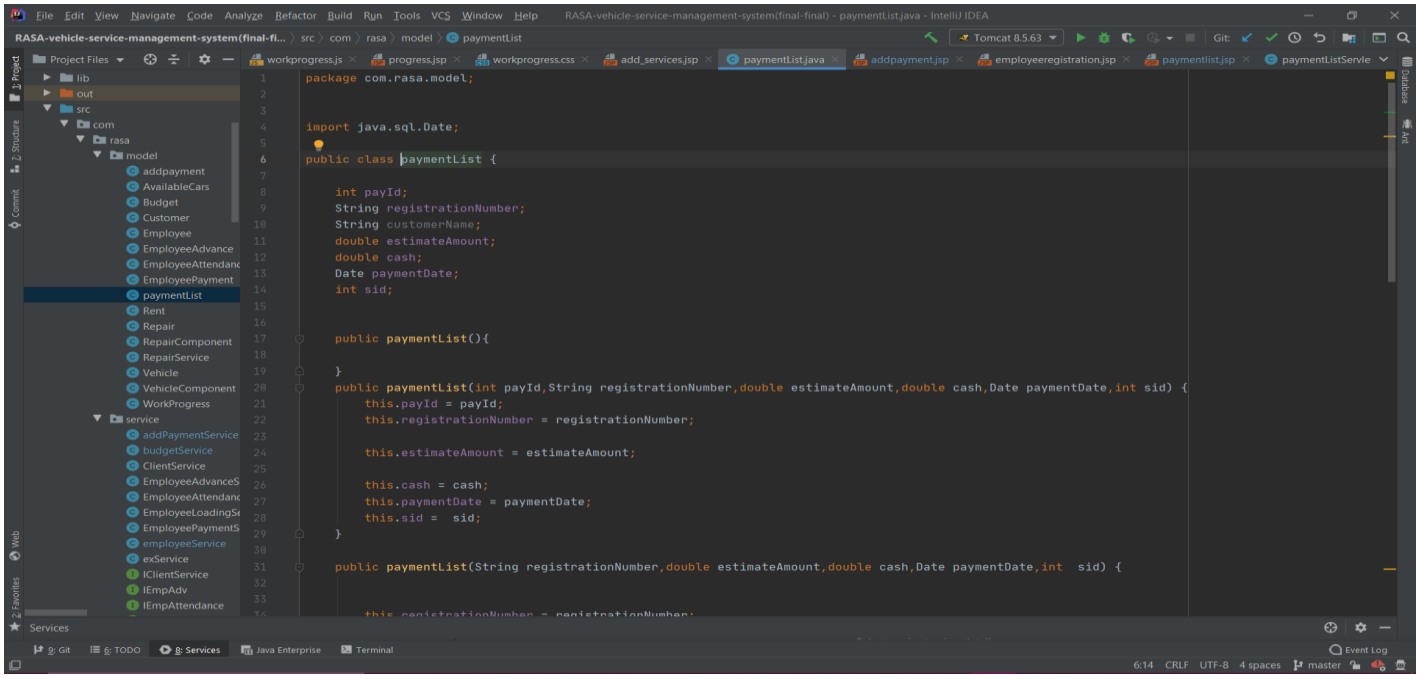


Figure 58: payment class

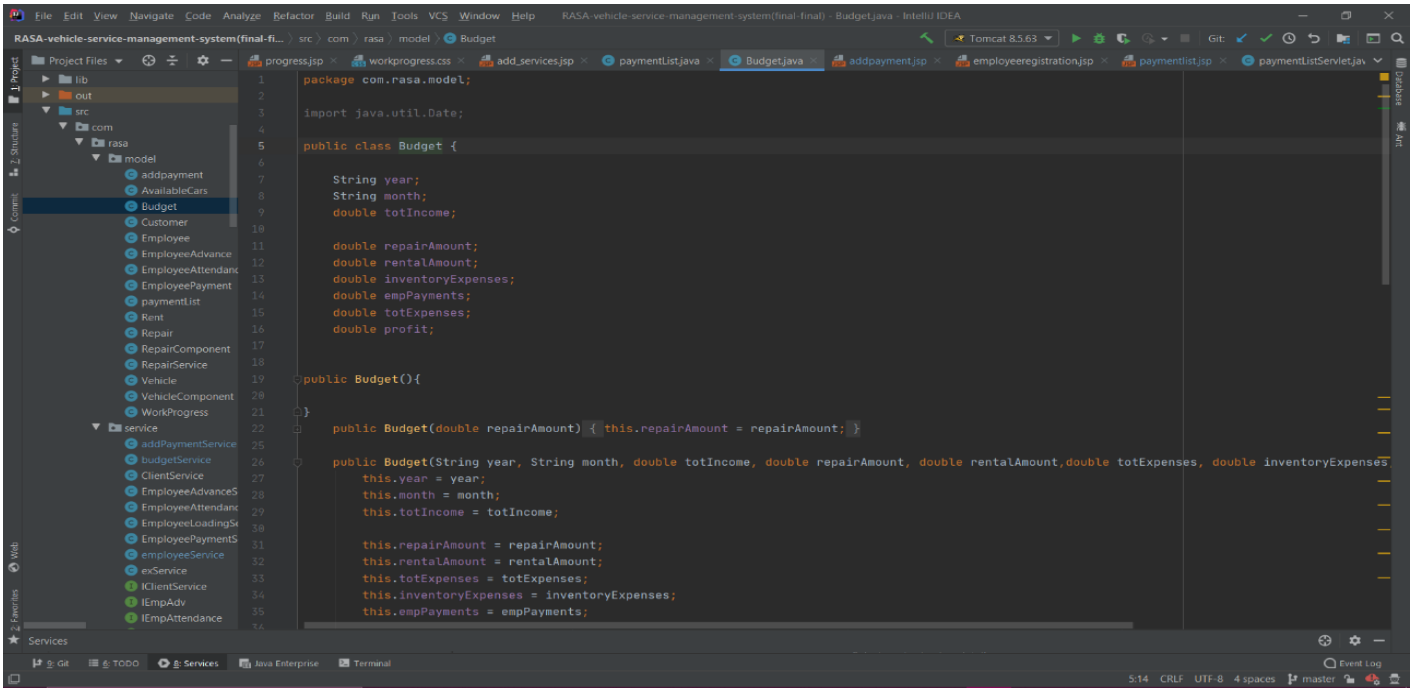
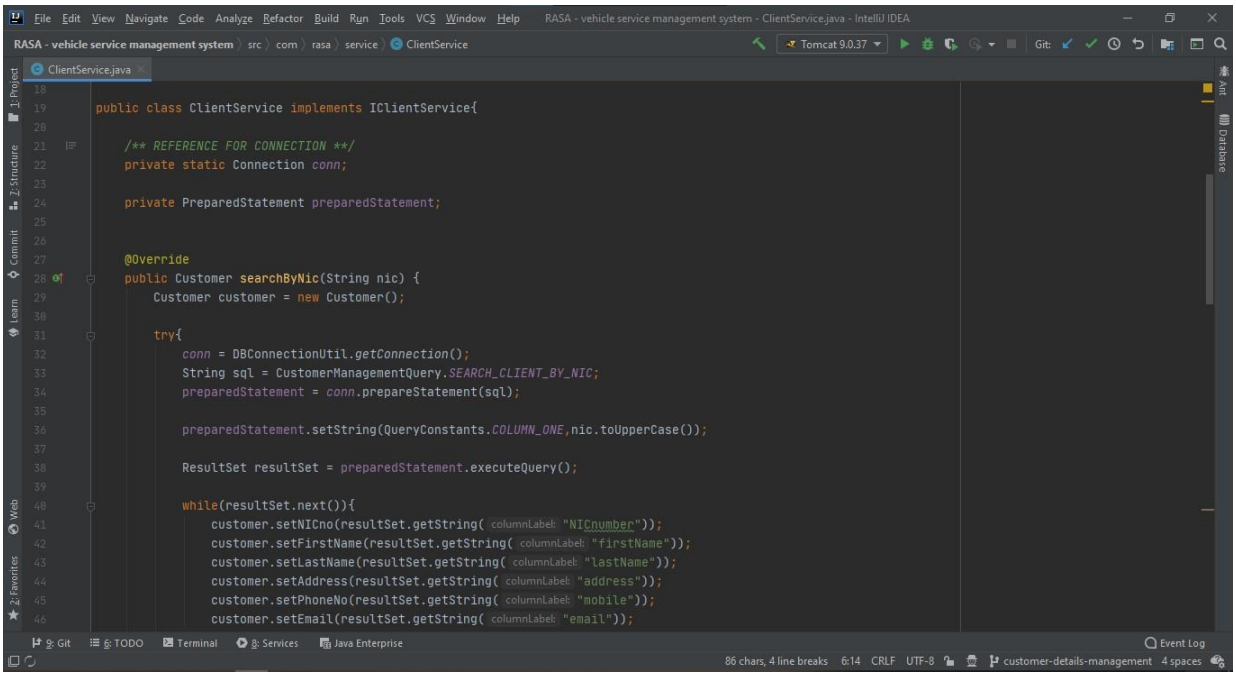


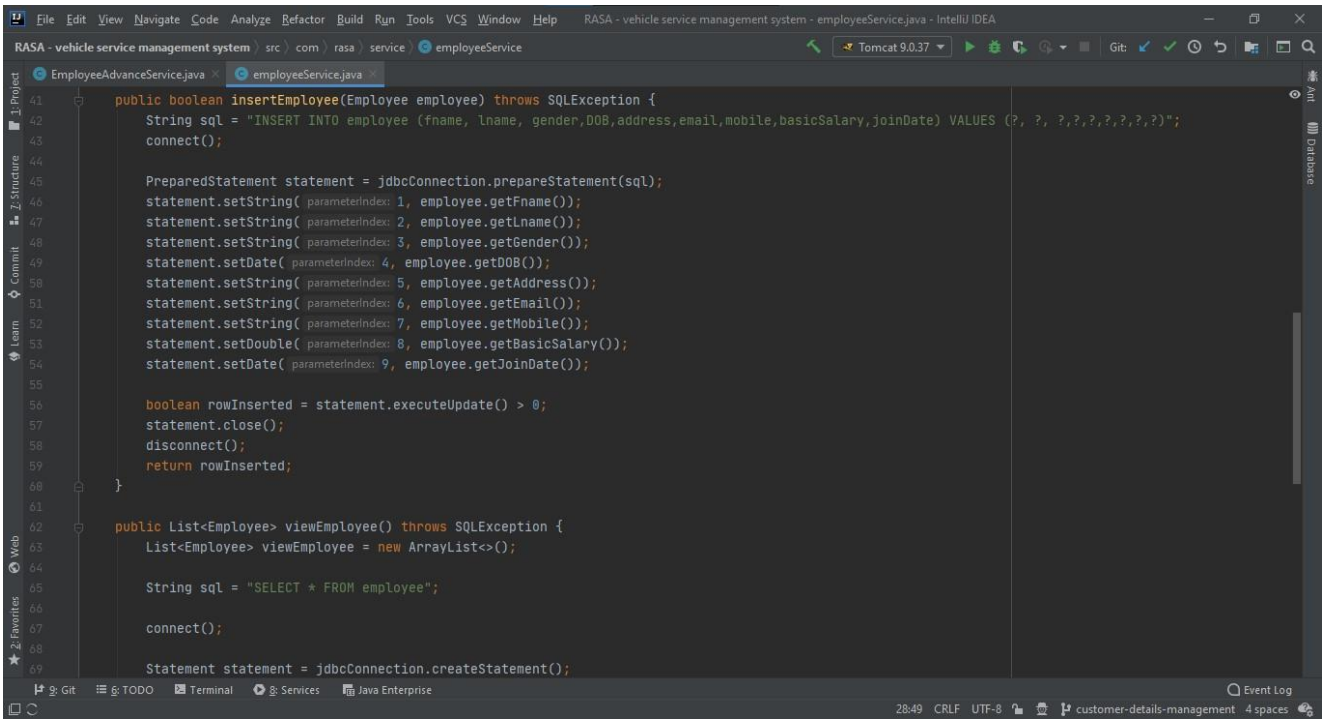
Figure 59: Budget class

Service classes



```
public class ClientService implements IClientService{

    /** REFERENCE FOR CONNECTION **/
    private static Connection conn;

    private PreparedStatement preparedStatement;


    @Override
    public Customer searchByNic(String nic) {
        Customer customer = new Customer();

        try{
            conn = DBConnectionUtil.getConnection();
            String sql = CustomerManagementQuery.SEARCH_CLIENT_BY_NIC;
            preparedStatement = conn.prepareStatement(sql);

            preparedStatement.setString(QueryConstants.COLUMN_ONE,nic.toUpperCase());

            ResultSet resultSet = preparedStatement.executeQuery();

            while(resultSet.next()){
                customer.setNICno(resultSet.getString( columnLabel: "NICnumber"));
                customer.setFirstName(resultSet.getString( columnLabel: "firstName"));
                customer.setLastName(resultSet.getString( columnLabel: "lastName"));
                customer.setAddress(resultSet.getString( columnLabel: "address"));
                customer.setPhoneNo(resultSet.getString( columnLabel: "mobile"));
                customer.setEmail(resultSet.getString( columnLabel: "email"));
```

Figure 60: Client Service



```
    public boolean insertEmployee(Employee employee) throws SQLException {
        String sql = "INSERT INTO employee (fname, lname, gender,DOB,address,email,mobile,basicSalary,joinDate) VALUES (?, ?, ?,?,?,?,?,?,?)";
        connect();

        PreparedStatement statement = jdbcConnection.prepareStatement(sql);
        statement.setString( parameterIndex: 1, employee.getFname());
        statement.setString( parameterIndex: 2, employee.getLname());
        statement.setString( parameterIndex: 3, employee.getGender());
        statement.setDate( parameterIndex: 4, employee.getDOB());
        statement.setString( parameterIndex: 5, employee.getAddress());
        statement.setString( parameterIndex: 6, employee.getEmail());
        statement.setString( parameterIndex: 7, employee.getMobile());
        statement.setDouble( parameterIndex: 8, employee.getBasicSalary());
        statement.setDate( parameterIndex: 9, employee.getJoinDate());

        boolean rowInserted = statement.executeUpdate() > 0;
        statement.close();
        disconnect();
        return rowInserted;
    }

    public List<Employee> viewEmployee() throws SQLException {
        List<Employee> viewEmployee = new ArrayList<>();

        String sql = "SELECT * FROM employee";

        connect();

        Statement statement = jdbcConnection.createStatement();
```

Figure 61: employee service

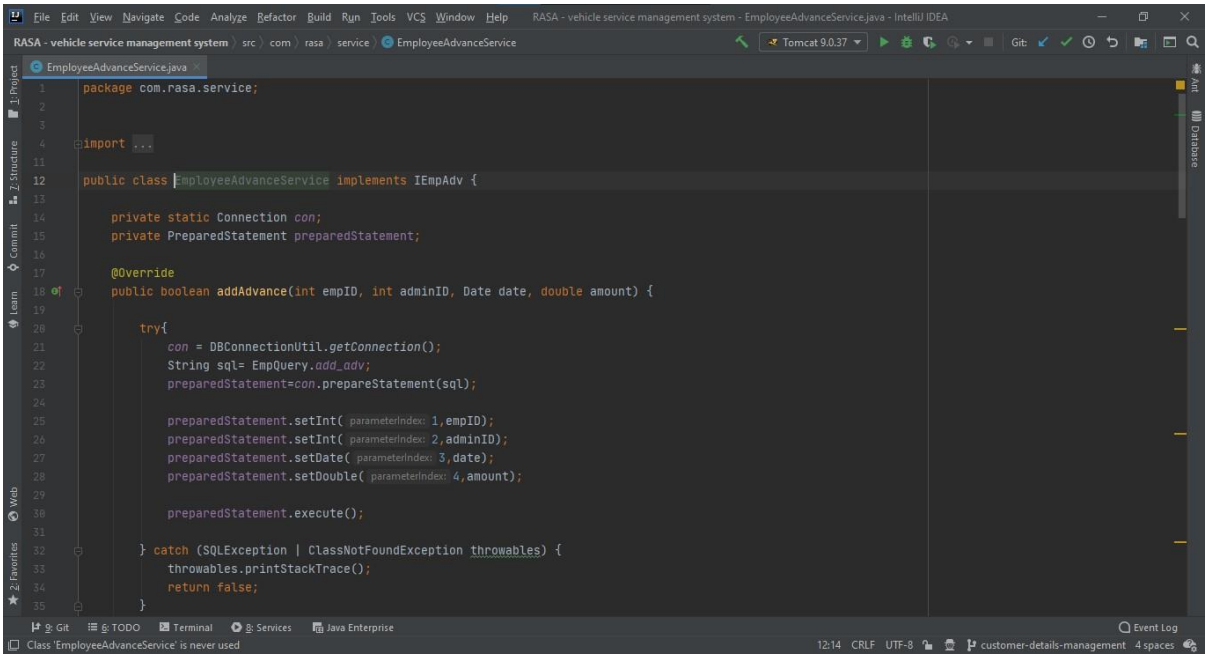


Figure 62: Employee Advance service

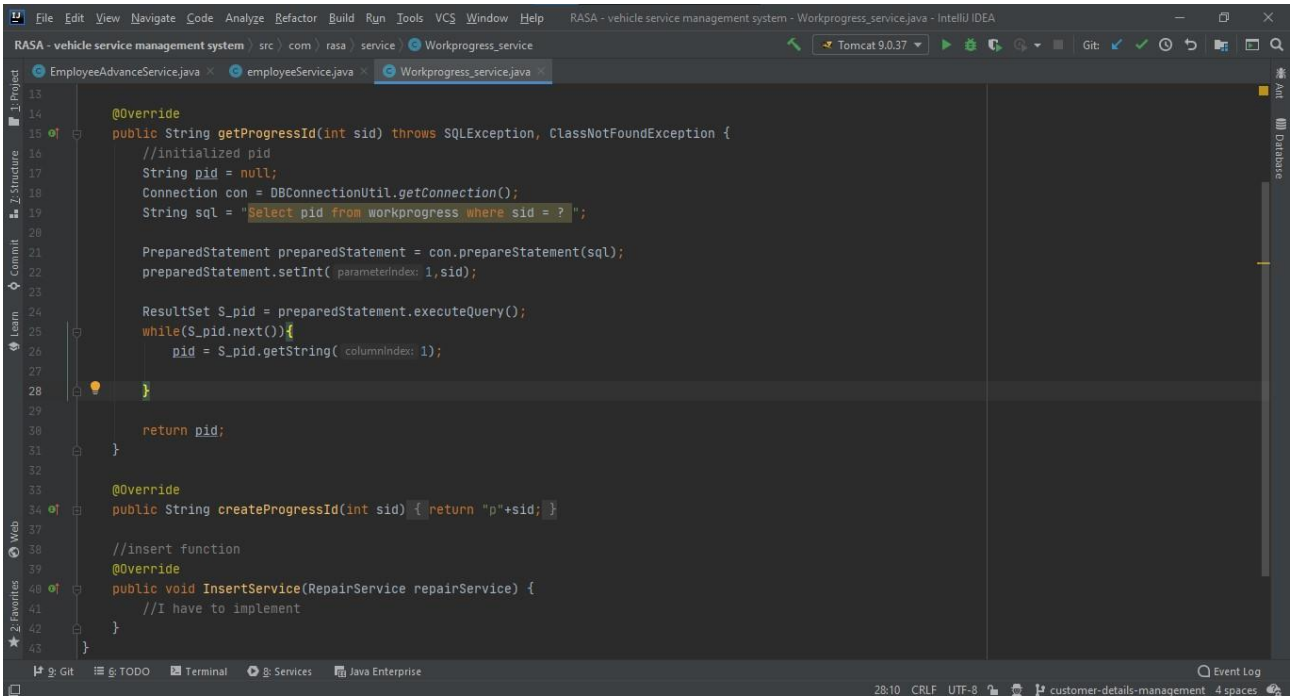
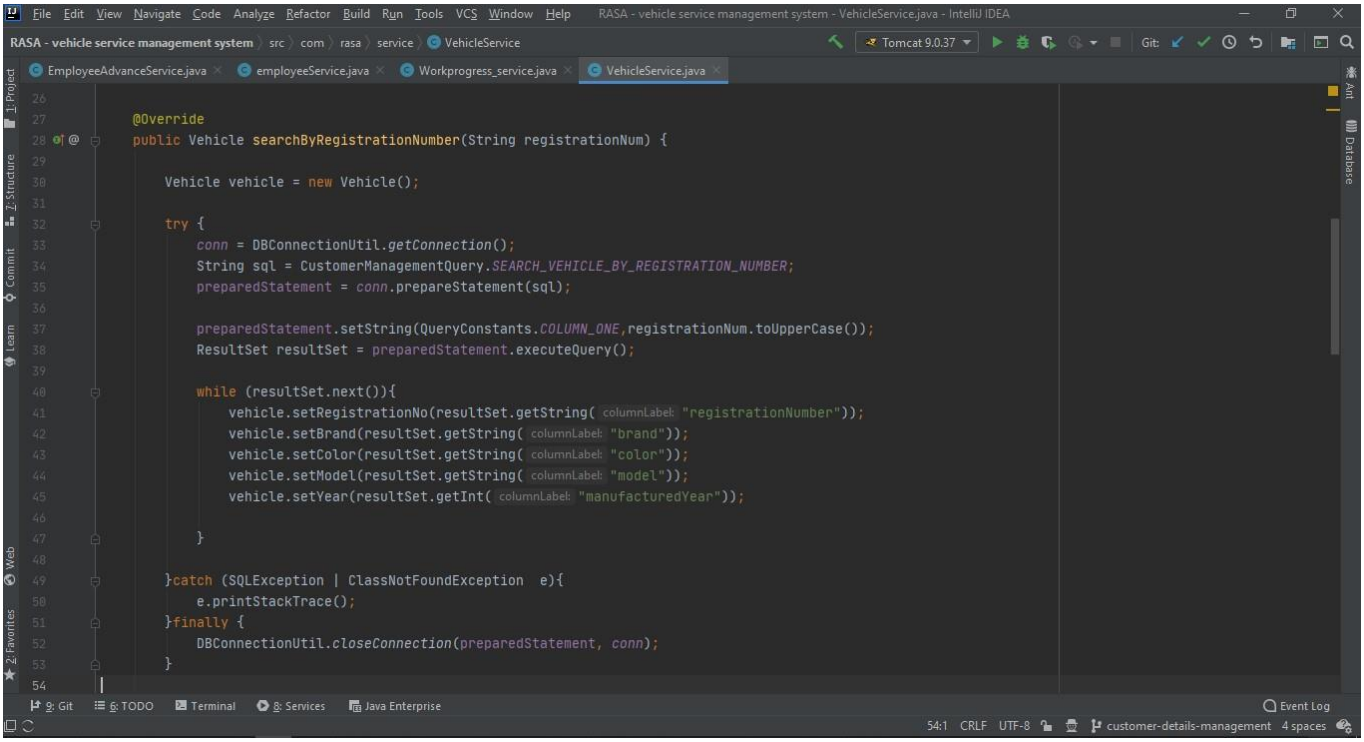
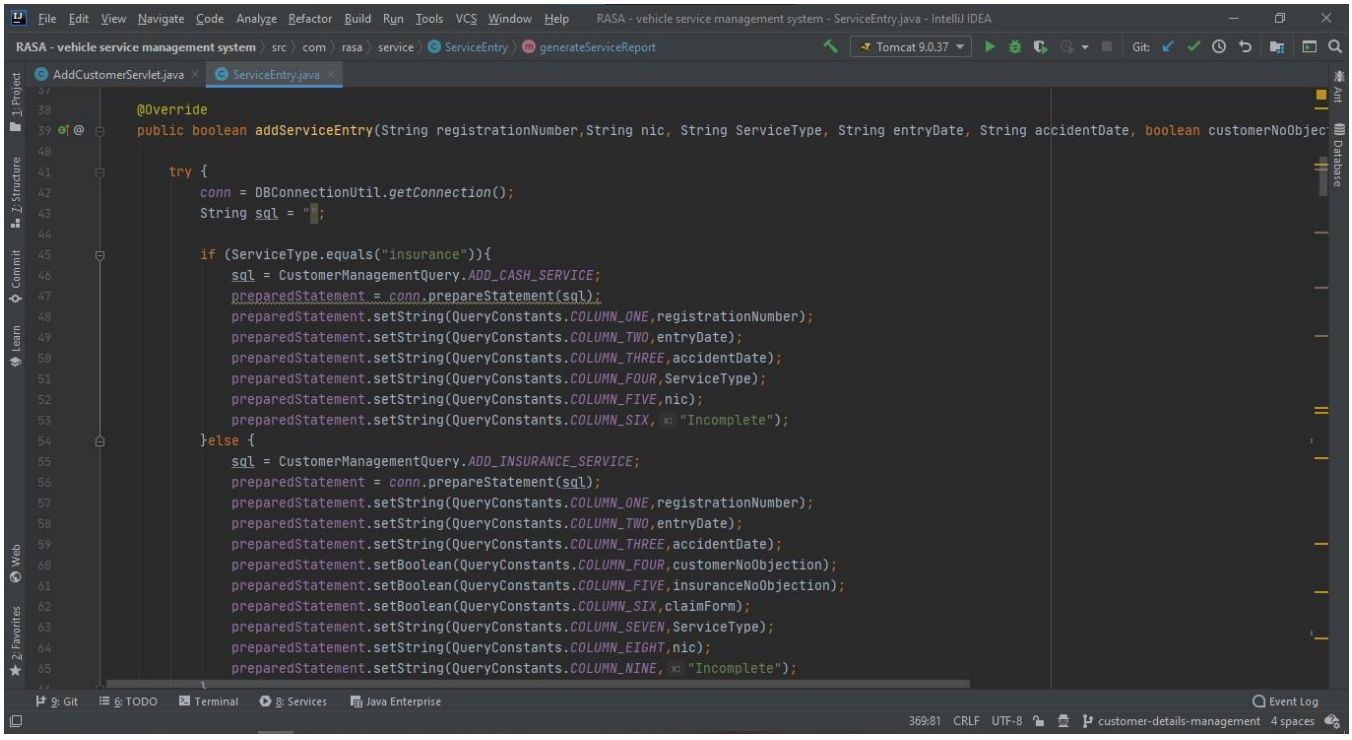


Figure 63:work progress service



```
@Override
public Vehicle searchByRegistrationNumber(String registrationNum) {

    Vehicle vehicle = new Vehicle();

    try {
        conn = DBConnectionUtil.getConnection();
        String sql = CustomerManagementQuery.SEARCH_VEHICLE_BY_REGISTRATION_NUMBER;
        preparedStatement = conn.prepareStatement(sql);

        preparedStatement.setString(QueryConstants.COLUMN_ONE,registrationNum.toUpperCase());
        ResultSet resultSet = preparedStatement.executeQuery();

        while (resultSet.next()){
            vehicle.setRegistrationNo(resultSet.getString( columnLabel: "registrationNumber"));
            vehicle.setBrand(resultSet.getString( columnLabel: "brand"));
            vehicle.setColor(resultSet.getString( columnLabel: "color"));
            vehicle.setModel(resultSet.getString( columnLabel: "model"));
            vehicle.setYear(resultSet.getInt( columnLabel: "manufacturedYear"));

        }

    }catch (SQLException | ClassNotFoundException  e){
        e.printStackTrace();
    }finally {
        DBConnectionUtil.closeConnection(preparedStatement, conn);
    }
```

Figure 64: vehicle service



```
@Override
public boolean addServiceEntry(String registrationNumber,String nic, String ServiceType, String entryDate, String accidentDate, boolean customerNoObjec

    try {
        conn = DBConnectionUtil.getConnection();
        String sql = "";

        if (ServiceType.equals("insurance")){
            sql = CustomerManagementQuery.ADD_CASH_SERVICE;
            preparedStatement = conn.prepareStatement(sql);
            preparedStatement.setString(QueryConstants.COLUMN_ONE,registrationNumber);
            preparedStatement.setString(QueryConstants.COLUMN_TWO,entryDate);
            preparedStatement.setString(QueryConstants.COLUMN_THREE,accidentDate);
            preparedStatement.setString(QueryConstants.COLUMN_FOUR,ServiceType);
            preparedStatement.setString(QueryConstants.COLUMN_FIVE,nic);
            preparedStatement.setString(QueryConstants.COLUMN_SIX, x: "Incomplete");
        }else {
            sql = CustomerManagementQuery.ADD_INSURANCE_SERVICE;
            preparedStatement = conn.prepareStatement(sql);
            preparedStatement.setString(QueryConstants.COLUMN_ONE,registrationNumber);
            preparedStatement.setString(QueryConstants.COLUMN_TWO,entryDate);
            preparedStatement.setString(QueryConstants.COLUMN_THREE,accidentDate);
            preparedStatement.setBoolean(QueryConstants.COLUMN_FOUR,customerNoObjection);
            preparedStatement.setBoolean(QueryConstants.COLUMN_FIVE,insuranceNoObjection);
            preparedStatement.setBoolean(QueryConstants.COLUMN_SIX,claimForm);
            preparedStatement.setString(QueryConstants.COLUMN_SEVEN,ServiceType);
            preparedStatement.setString(QueryConstants.COLUMN_EIGHT,nic);
            preparedStatement.setString(QueryConstants.COLUMN_NINE, x: "Incomplete");
```

Figure 65: service entry service

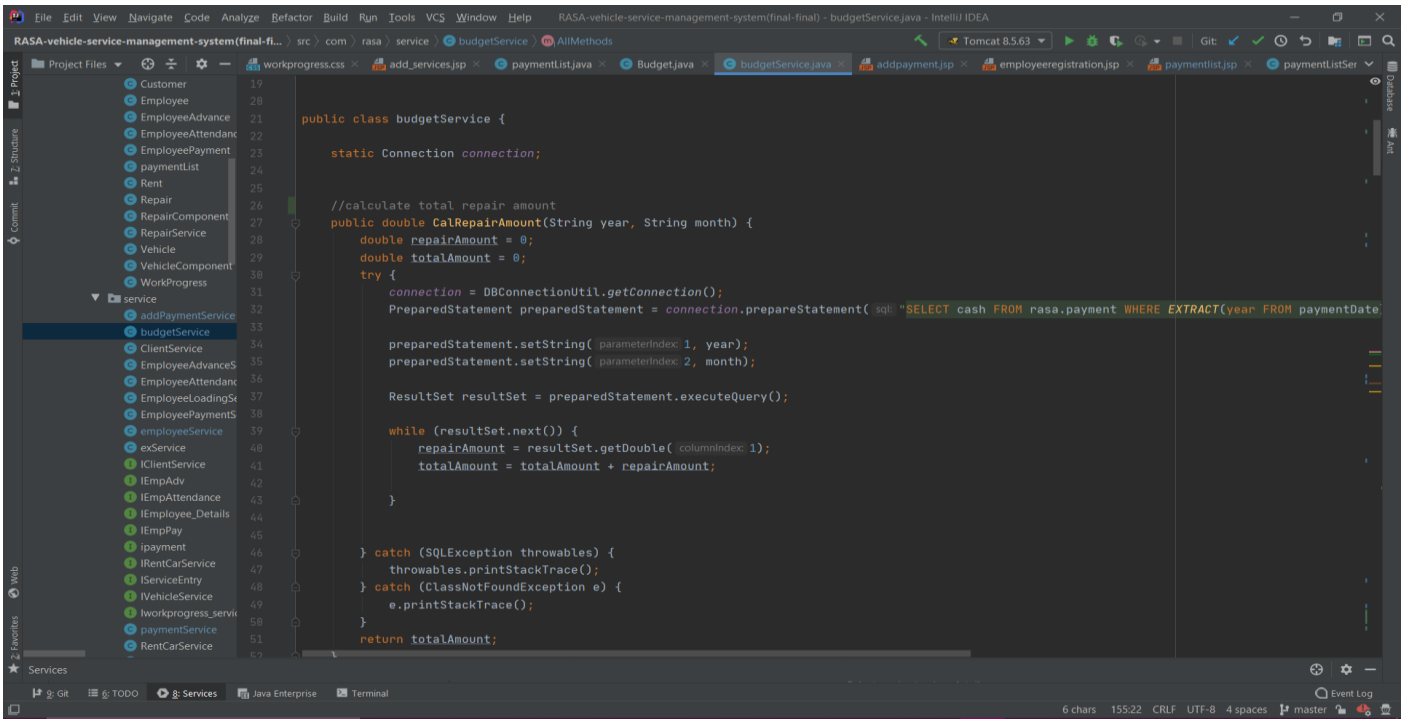


Figure 66: budget service

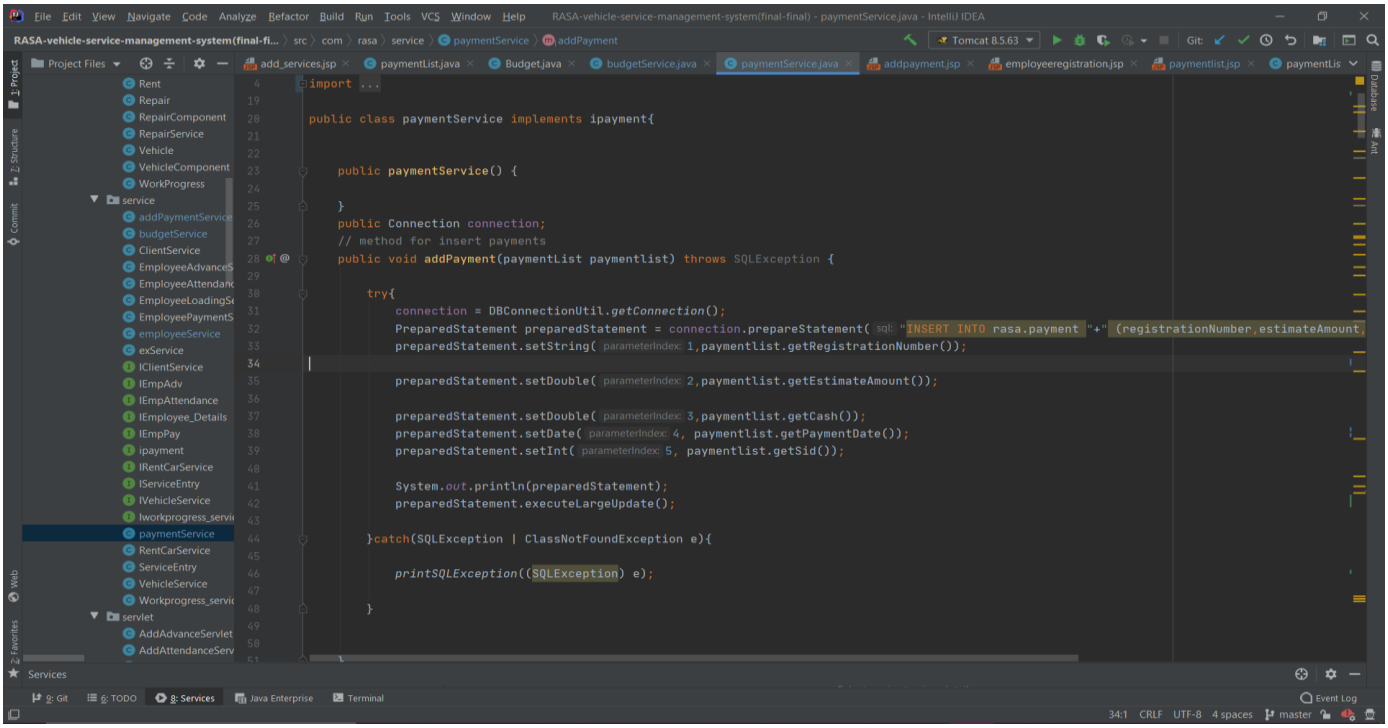
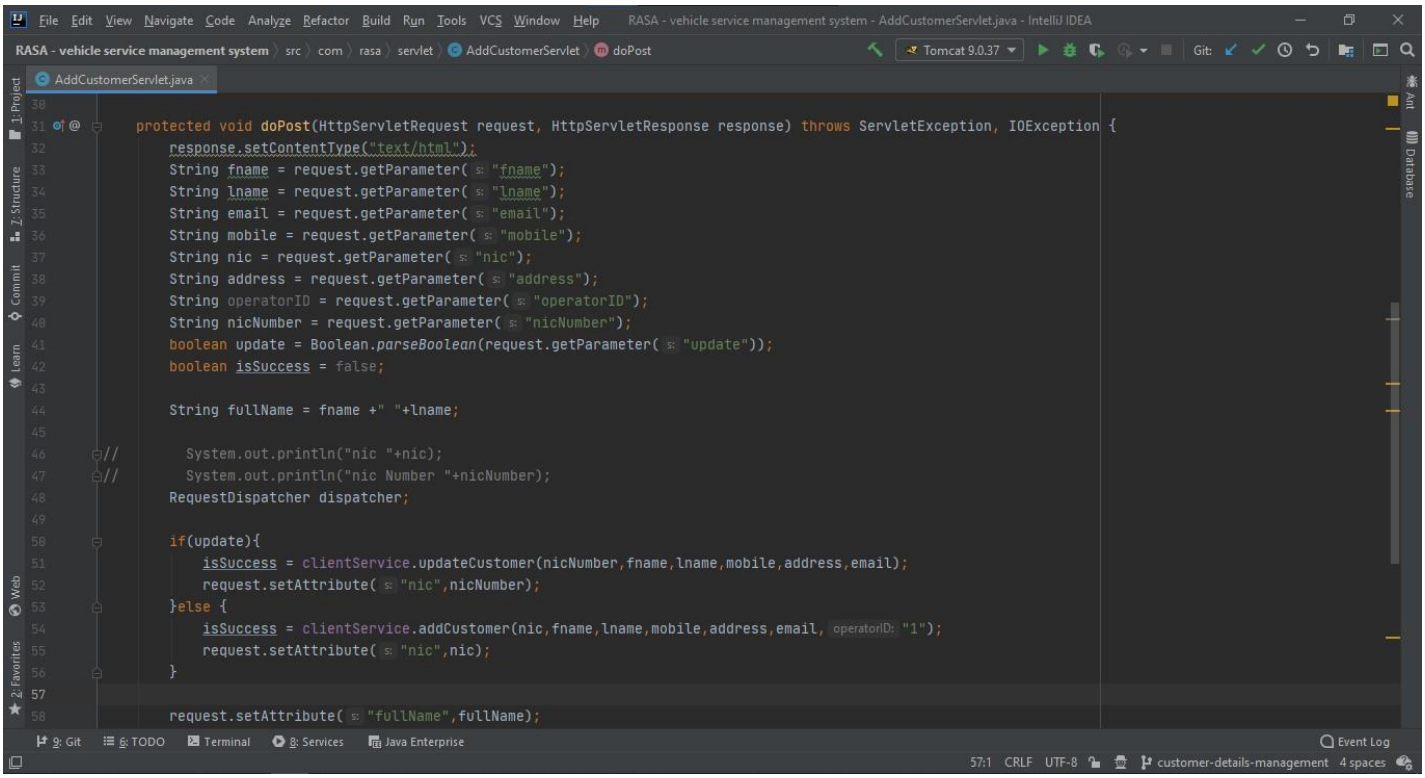


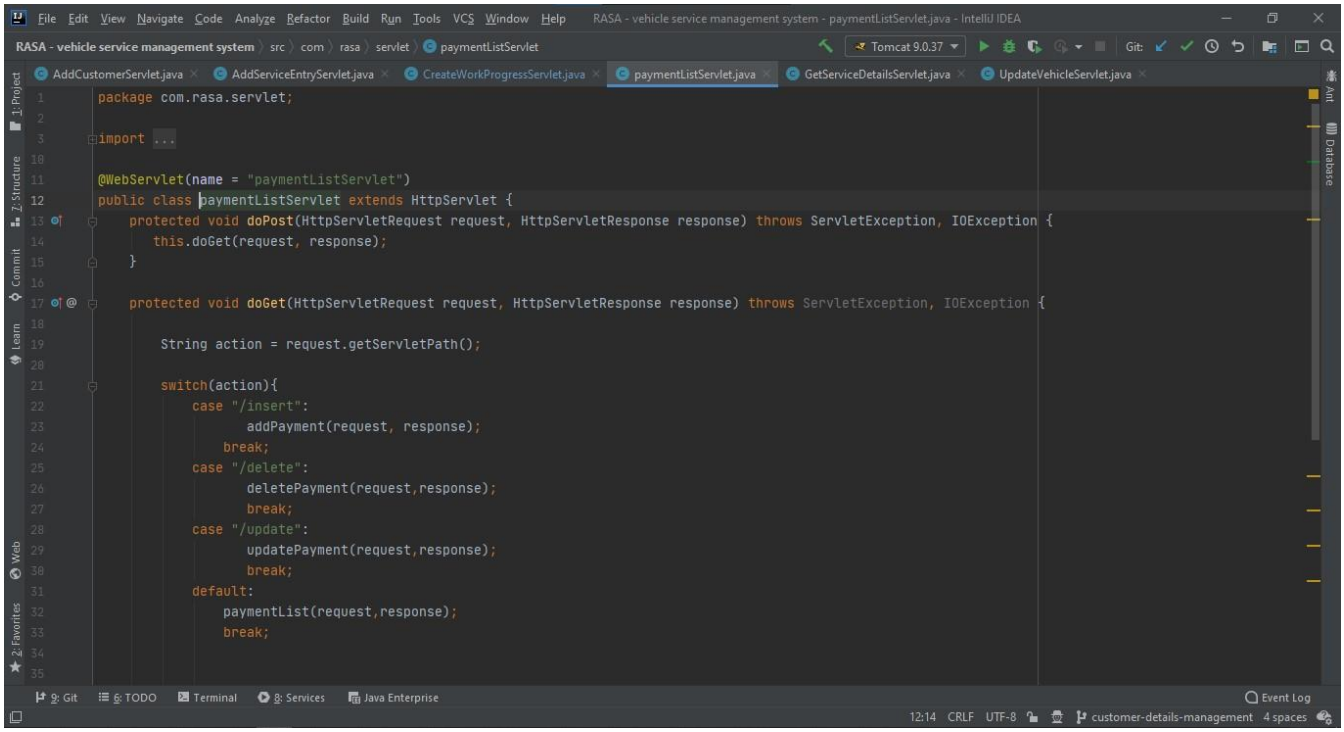
Figure 67: payment service

Servlet classes



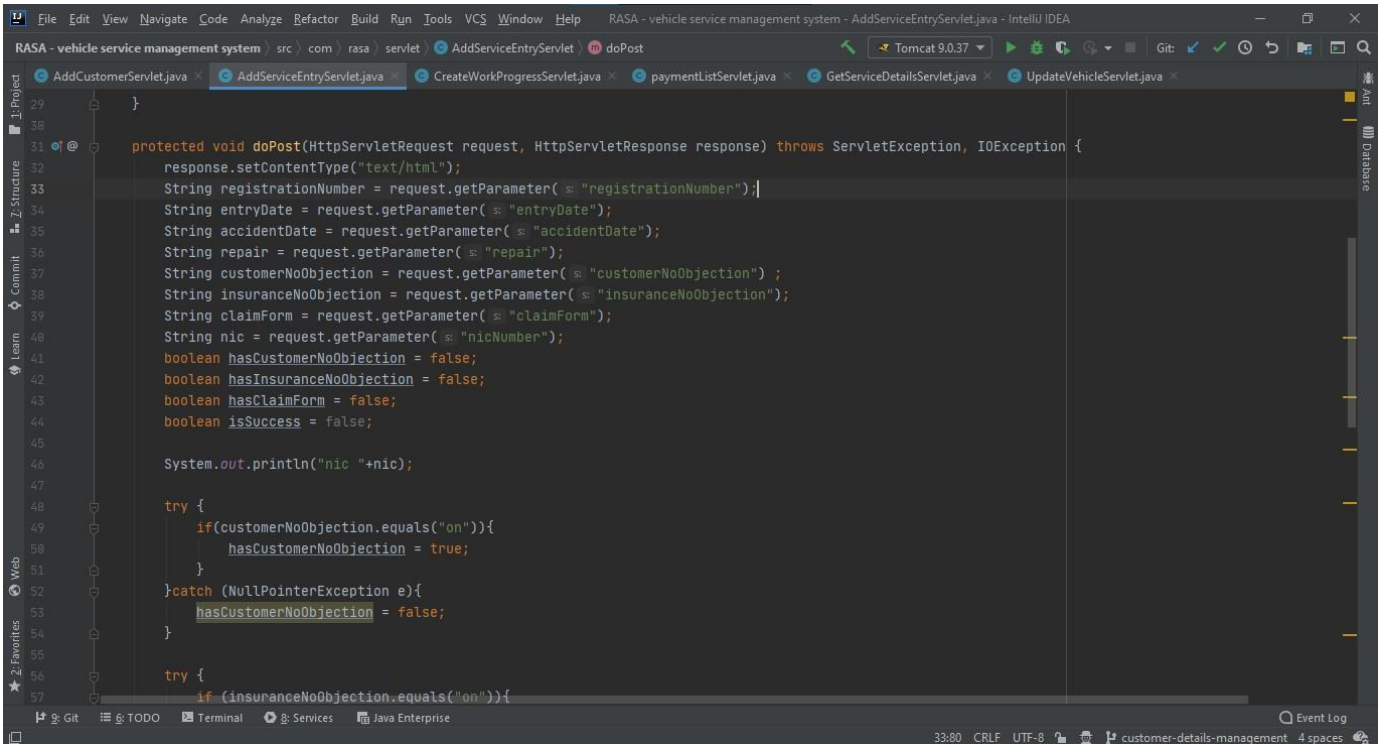
```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    response.setContentType("text/html");
    String fname = request.getParameter( s: "fname");
    String lname = request.getParameter( s: "lname");
    String email = request.getParameter( s: "email");
    String mobile = request.getParameter( s: "mobile");
    String nic = request.getParameter( s: "nic");
    String address = request.getParameter( s: "address");
    String operatorID = request.getParameter( s: "operatorID");
    String nicNumber = request.getParameter( s: "nicNumber");
    boolean update = Boolean.parseBoolean(request.getParameter( s: "update"));
    boolean isSuccess = false;

    String fullName = fname +" "+lname;

//      System.out.println("nic "+nic);
//      System.out.println("nic Number "+nicNumber);
    RequestDispatcher dispatcher;

    if(update){
        isSuccess = clientService.updateCustomer(nicNumber,fname,lname,mobile,address,email);
        request.setAttribute( s: "nic",nicNumber);
    }else {
        isSuccess = clientService.addCustomer(nic,fname,lname,mobile,address,email, operatorID: "1");
        request.setAttribute( s: "nic",nic);
    }

    request.setAttribute( s: "fullName",fullName);
```

Figure 68: Add Customer



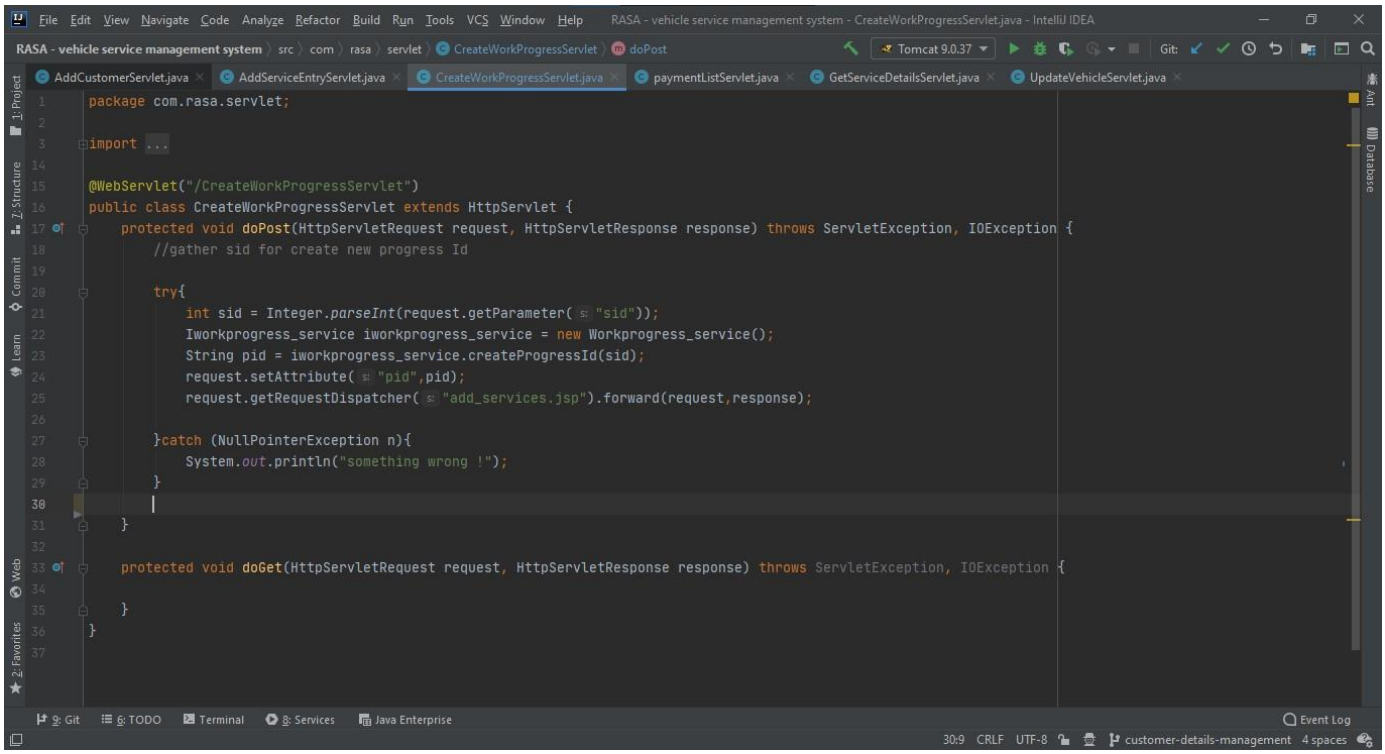
```
package com.rasa.servlet;

import ...

@WebServlet(name = "paymentListServlet")
public class paymentListServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        this.doGet(request, response);
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

        String action = request.getServletPath();

        switch(action){
            case "/insert":
                  addPayment(request, response);
                break;
            case "/delete":
                  deletePayment(request,response);
                  break;
            case "/update":
                  updatePayment(request,response);
                  break;
            default:
                paymentList(request,response);
                break;
```

Figure 69: Payment Servlet



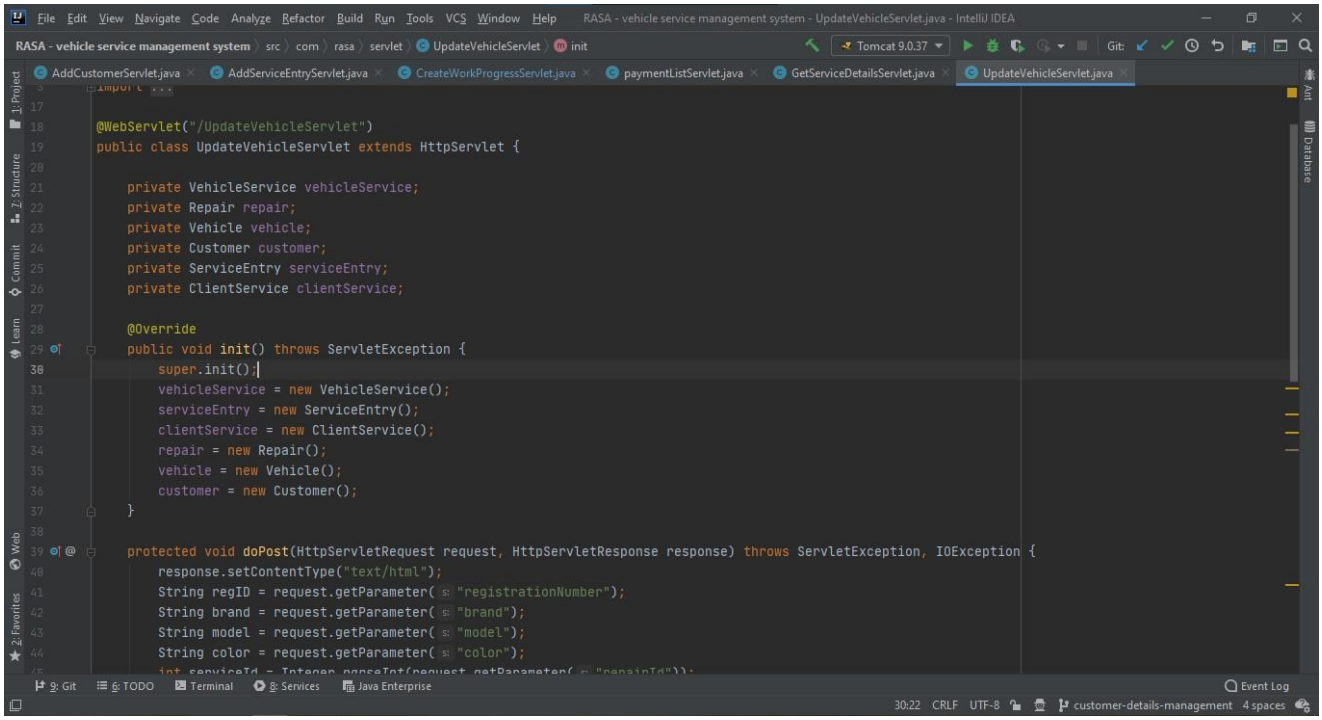
```
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        response.setContentType("text/html");
        String registrationNumber = request.getParameter( s: "registrationNumber");
        String entryDate = request.getParameter( s: "entryDate");
        String accidentDate = request.getParameter( s: "accidentDate");
        String repair = request.getParameter( s: "repair");
        String customerNoObjection = request.getParameter( s: "customerNoObjection") ;
        String insuranceNoObjection = request.getParameter( s: "insuranceNoObjection");
        String claimForm = request.getParameter( s: "claimForm");
        String nic = request.getParameter( s: "nicNumber");
        boolean hasCustomerNoObjection = false;
        boolean hasInsuranceNoObjection = false;
        boolean hasClaimForm = false;
        boolean isSuccess = false;

        System.out.println("nic "+nic);

        try {
            if(customerNoObjection.equals("on")){
                hasCustomerNoObjection = true;
            }
        }catch (NullPointerException e){
            hasCustomerNoObjection = false;
        }

        try {
            if (insuranceNoObjection.equals("on")){
```

Figure 70: Add service Entry



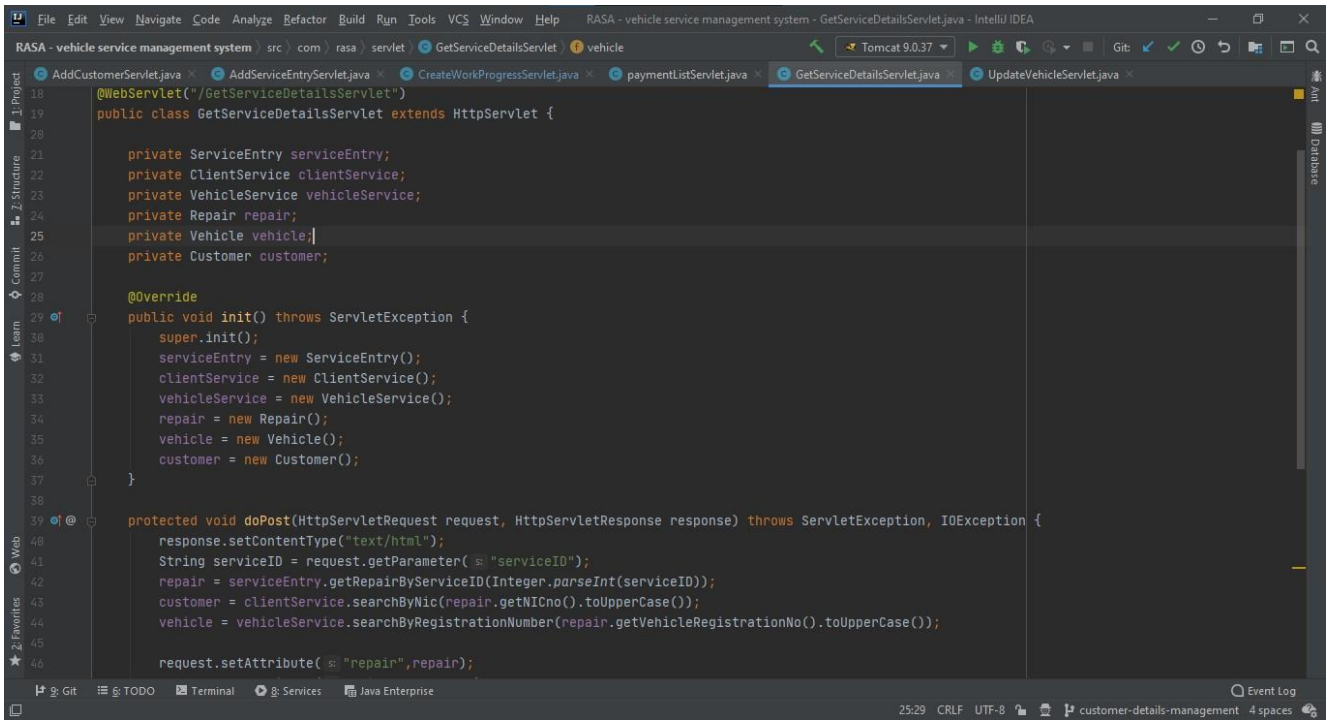
```
package com.rasa.servlet;

import ...

@WebServlet("/CreateWorkProgressServlet")
public class CreateWorkProgressServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        //gather sid for create new progress Id

        try{
            int sid = Integer.parseInt(request.getParameter( s: "sid"));
            Iworkprogress_service iworkprogress_service = new Workprogress_service();
            String pid = iworkprogress_service.createProgressId(sid);
            request.setAttribute( s: "pid",pid);
            request.getRequestDispatcher( s: "add_services.jsp").forward(request,response);

        }catch (NullPointerException n){
            System.out.println("something wrong !");
        }

    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

    }
}
```

Figure 71: create work progress



```
@WebServlet("/UpdateVehicleServlet")
public class UpdateVehicleServlet extends HttpServlet {

    private VehicleService vehicleService;
    private Repair repair;
    private Vehicle vehicle;
    private Customer customer;
    private ServiceEntry serviceEntry;
    private ClientService clientService;

    @Override
    public void init() throws ServletException {
        super.init();
        vehicleService = new VehicleService();
        serviceEntry = new ServiceEntry();
        clientService = new ClientService();
        repair = new Repair();
        vehicle = new Vehicle();
        customer = new Customer();
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        response.setContentType("text/html");
        String regID = request.getParameter( s: "registrationNumber");
        String brand = request.getParameter( s: "brand");
        String model = request.getParameter( s: "model");
        String color = request.getParameter( s: "color");
```

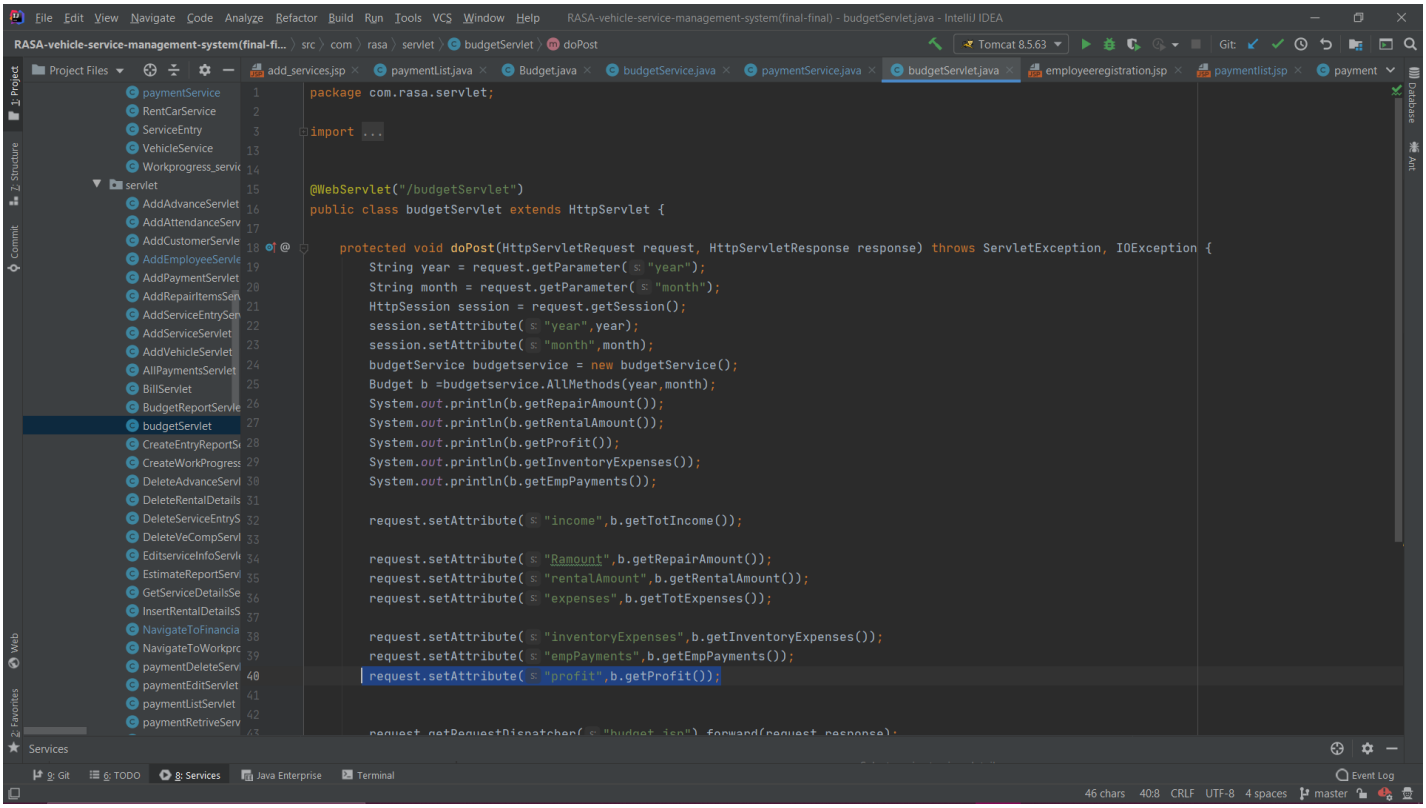
Figure 72: Upload vehicle



```
@WebServlet("/GetServiceDetailsServlet")
public class GetServiceDetailsServlet extends HttpServlet {

    private ServiceEntry serviceEntry;
    private ClientService clientService;
    private VehicleService vehicleService;
    private Repair repair;
    private Vehicle vehicle;
    private Customer customer;

    @Override
    public void init() throws ServletException {
        super.init();
        serviceEntry = new ServiceEntry();
        clientService = new ClientService();
        vehicleService = new VehicleService();
        repair = new Repair();
        vehicle = new Vehicle();
        customer = new Customer();
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        response.setContentType("text/html");
        String serviceID = request.getParameter( s: "serviceID");
        repair = serviceEntry.getRepairByServiceID(Integer.parseInt(serviceID));
        customer = clientService.searchByNic(repair.getNICno().toUpperCase());
        vehicle = vehicleService.searchByRegistrationNumber(repair.getVehicleRegistrationNo().toUpperCase());

        request.setAttribute( s: "repair",repair);
```

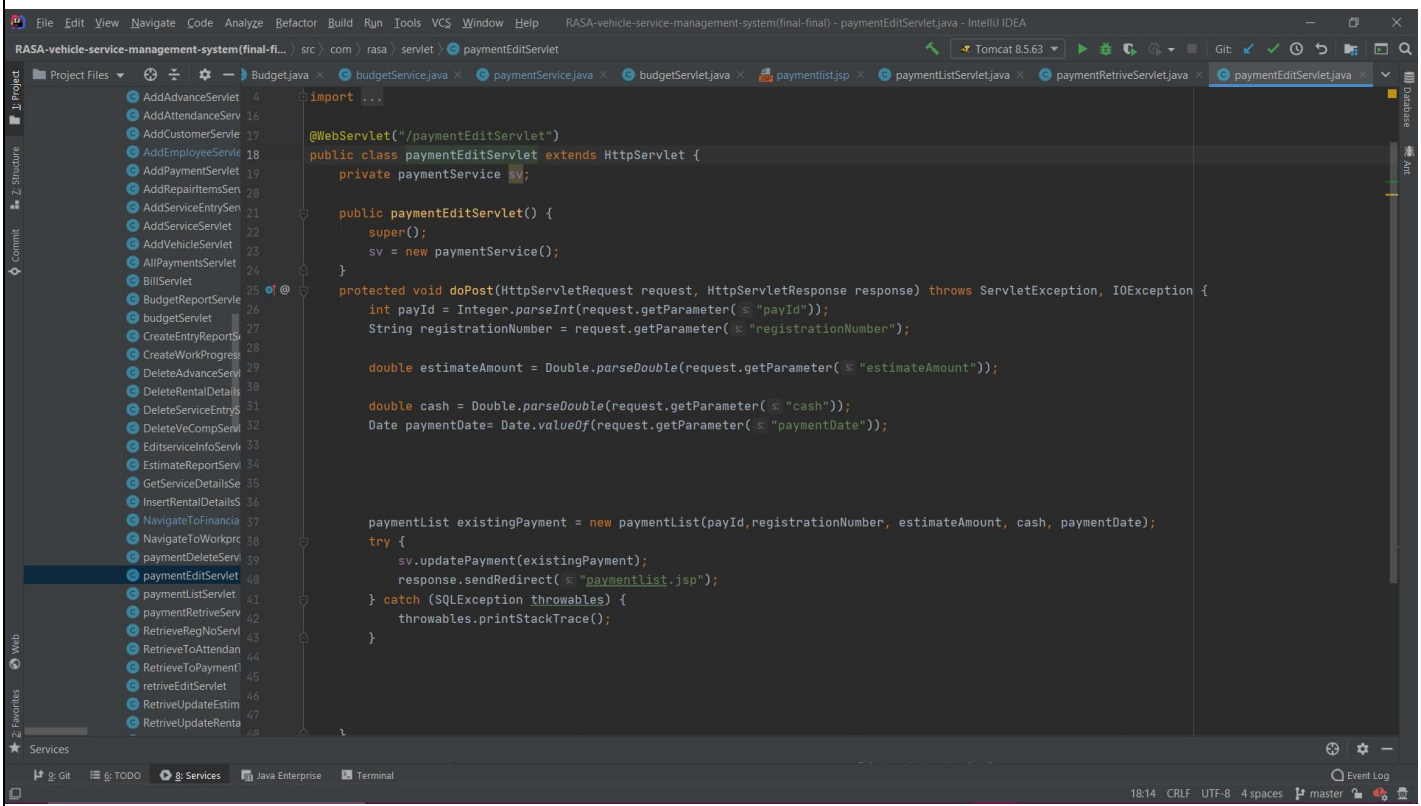
Figure 73: Get Service Details

Figure 74: Get budget amount



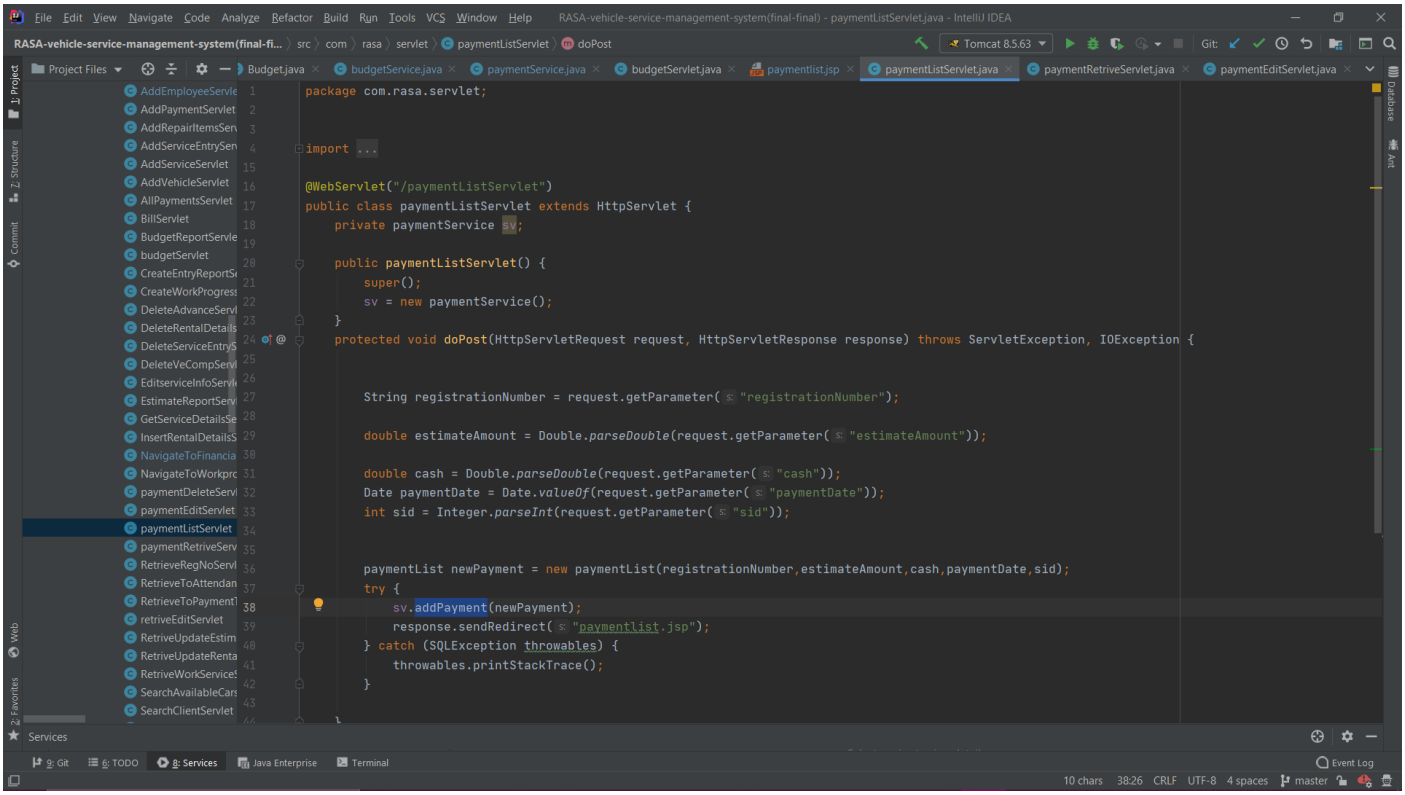
```
package com.rasa.servlet;

import ...


@WebServlet("/budgetServlet")
public class budgetServlet extends HttpServlet {

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        String year = request.getParameter( s: "year");
        String month = request.getParameter( s: "month");
        HttpSession session = request.getSession();
        session.setAttribute( s: "year",year);
        session.setAttribute( s: "month",month);
        budgetService budgetservice = new budgetService();
        Budget b =budgetservice.AllMethods(year,month);
        System.out.println(b.getRepairAmount());
        System.out.println(b.getRentalAmount());
        System.out.println(b.getProfit());
        System.out.println(b.getInventoryExpenses());
        System.out.println(b.getEmpPayments());

        request.setAttribute( s: "income",b.getTotIncome());

        request.setAttribute( s: "Ramount",b.getRepairAmount());
        request.setAttribute( s: "rentalAmount",b.getRentalAmount());
        request.setAttribute( s: "expenses",b.getTotExpenses());

        request.setAttribute( s: "inventoryExpenses",b.getInventoryExpenses());
        request.setAttribute( s: "empPayments",b.getEmpPayments());
        request.setAttribute( s: "profit",b.getProfit());
```

Figure 75: Update payment



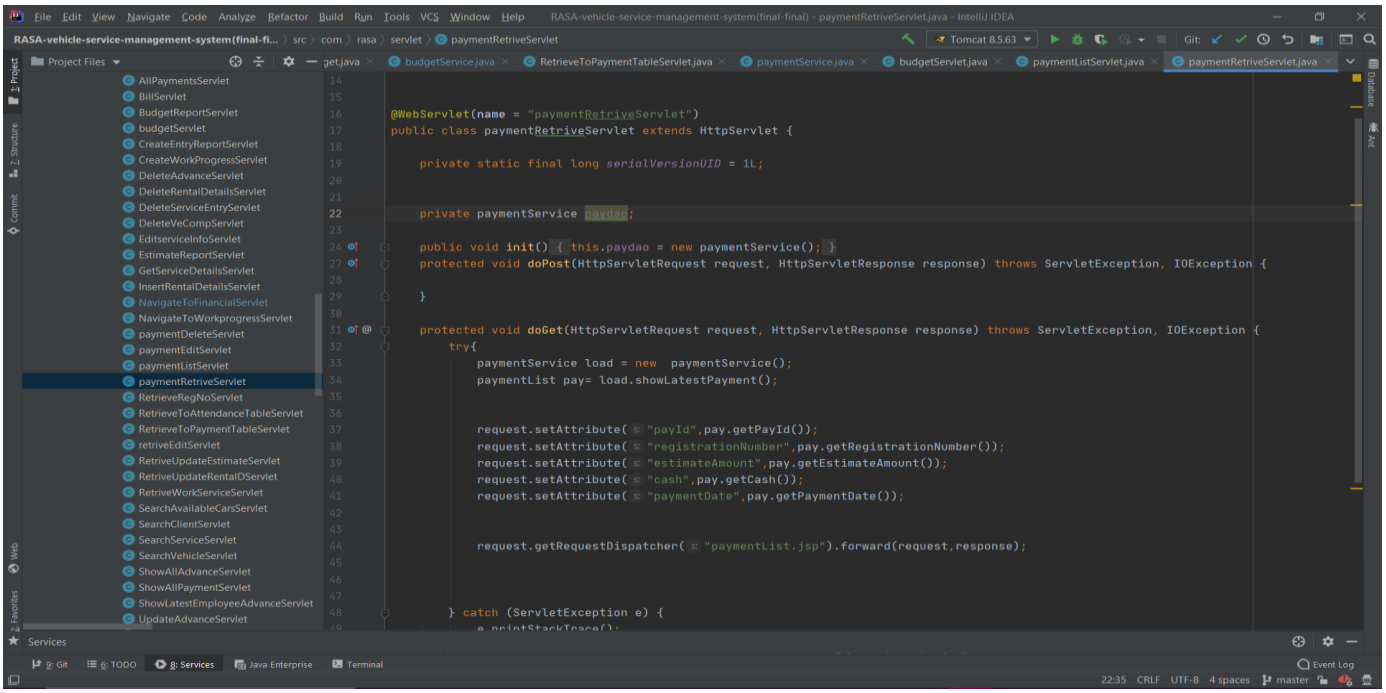
```
import ...

@WebServlet("/paymentEditServlet")
public class paymentEditServlet extends HttpServlet {
    private paymentService sv;

    public paymentEditServlet() {
        super();
        sv = new paymentService();
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        int payId = Integer.parseInt(request.getParameter( s: "payId"));
        String registrationNumber = request.getParameter( s: "registrationNumber");

        double estimateAmount = Double.parseDouble(request.getParameter( s: "estimateAmount"));

        double cash = Double.parseDouble(request.getParameter( s: "cash"));
        Date paymentDate= Date.valueOf(request.getParameter( s: "paymentDate"));



        paymentList existingPayment = new paymentList(payId,registrationNumber, estimateAmount, cash, paymentDate);
        try {
            sv.updatePayment(existingPayment);
            response.sendRedirect( s: "paymentlist.jsp");
        } catch (SQLException throwables) {
            throwables.printStackTrace();
        }
```

Figure 76:Add payment



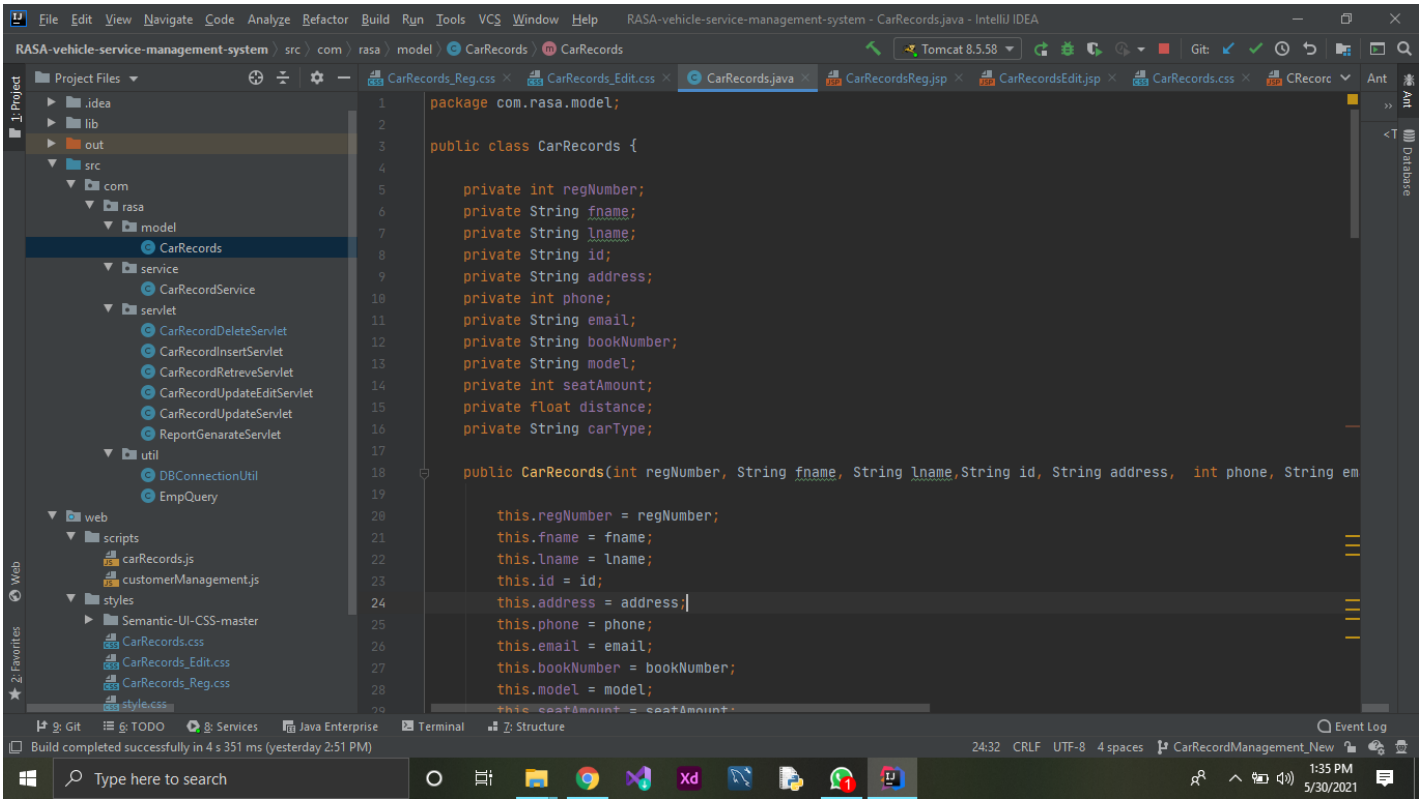
```
package com.rasa.servlet;


import ...

@WebServlet("/paymentListServlet")
public class paymentListServlet extends HttpServlet {
    private paymentService sv;

    public paymentListServlet() {
        super();
        sv = new paymentService();
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {


        String registrationNumber = request.getParameter( s: "registrationNumber");

        double estimateAmount = Double.parseDouble(request.getParameter( s: "estimateAmount"));

        double cash = Double.parseDouble(request.getParameter( s: "cash"));
        Date paymentDate = Date.valueOf(request.getParameter( s: "paymentDate"));
        int sid = Integer.parseInt(request.getParameter( s: "sid"));


        paymentList newPayment = new paymentList(registrationNumber,estimateAmount,cash,paymentDate,sid);
        try {
            sv.addPayment(newPayment);
            response.sendRedirect( s: "paymentlist.jsp");
        } catch (SQLException throwables) {
            throwables.printStackTrace();
        }
```

Figure 77: Display latest payment

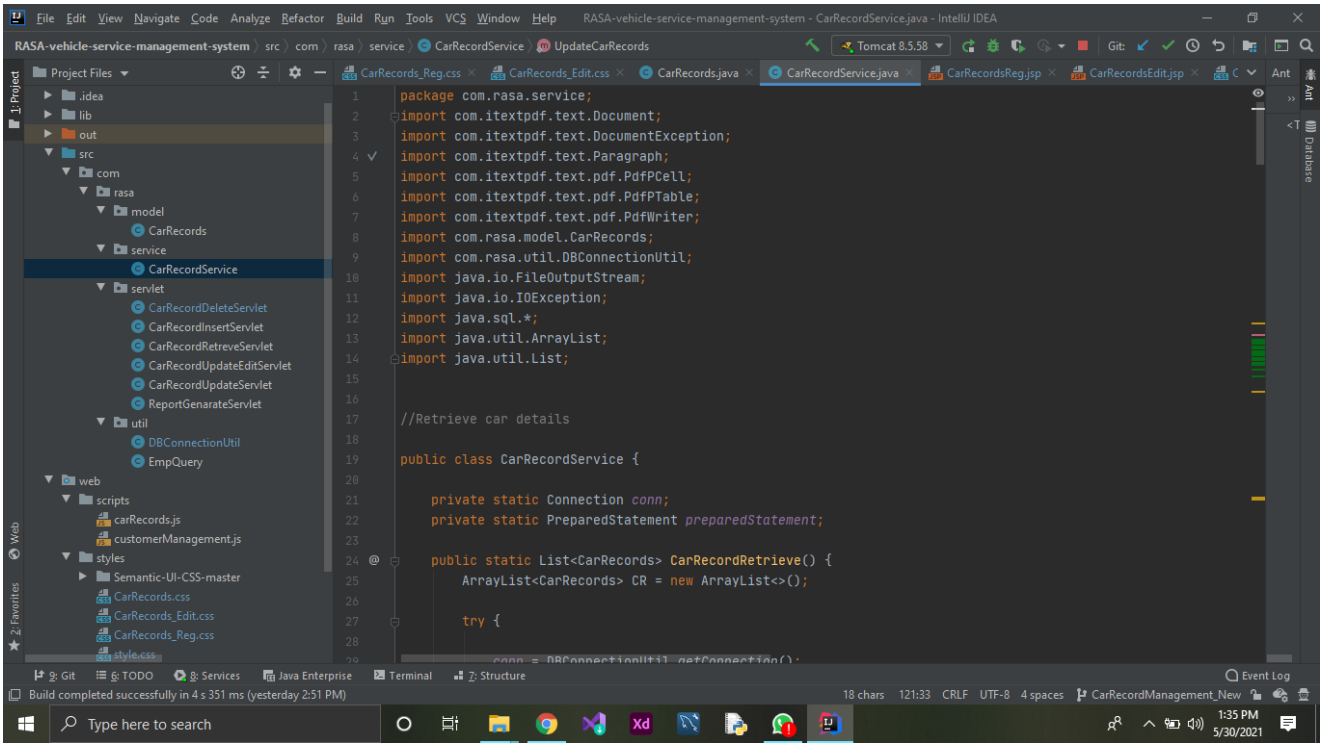


```
@WebServlet(name = "paymentRetriveServlet")
public class paymentRetriveServlet extends HttpServlet {

    private static final long serialVersionUID = 1L;


    private paymentService paydao;

    public void init() { this.paydao = new paymentService(); }
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        try{
            paymentService load = new  paymentService();
            paymentList pay= load.showLatestPayment();


            request.setAttribute( s: "payId",pay.getPayId());
            request.setAttribute( s: "registrationNumber",pay.getRegistrationNumber());
            request.setAttribute( s: "estimateAmount",pay.getEstimateAmount());
            request.setAttribute( s: "cash",pay.getCash());
            request.setAttribute( s: "paymentDate",pay.getPaymentDate());


            request.getRequestDispatcher( s: "paymentList.jsp").forward(request,response);



        } catch (ServletException e) {
```

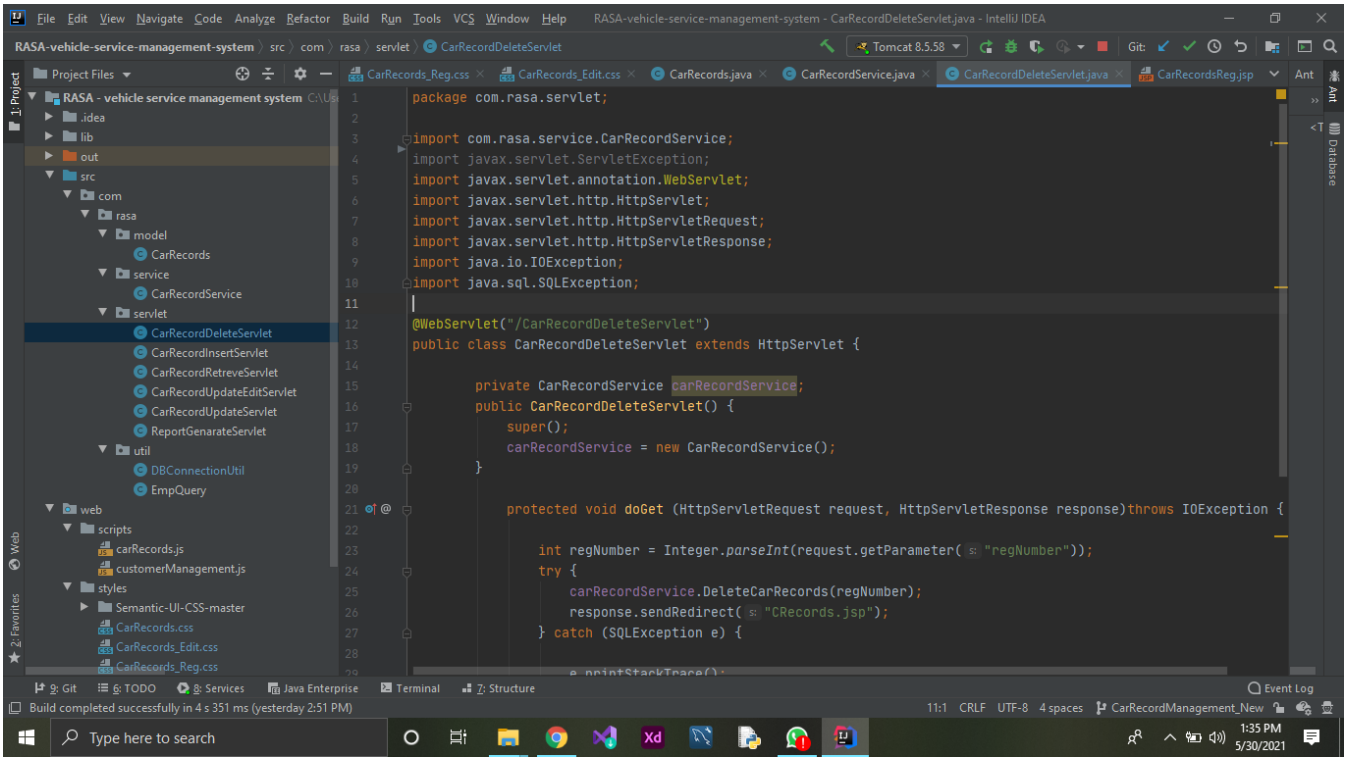
Car records model class *Figure 78*



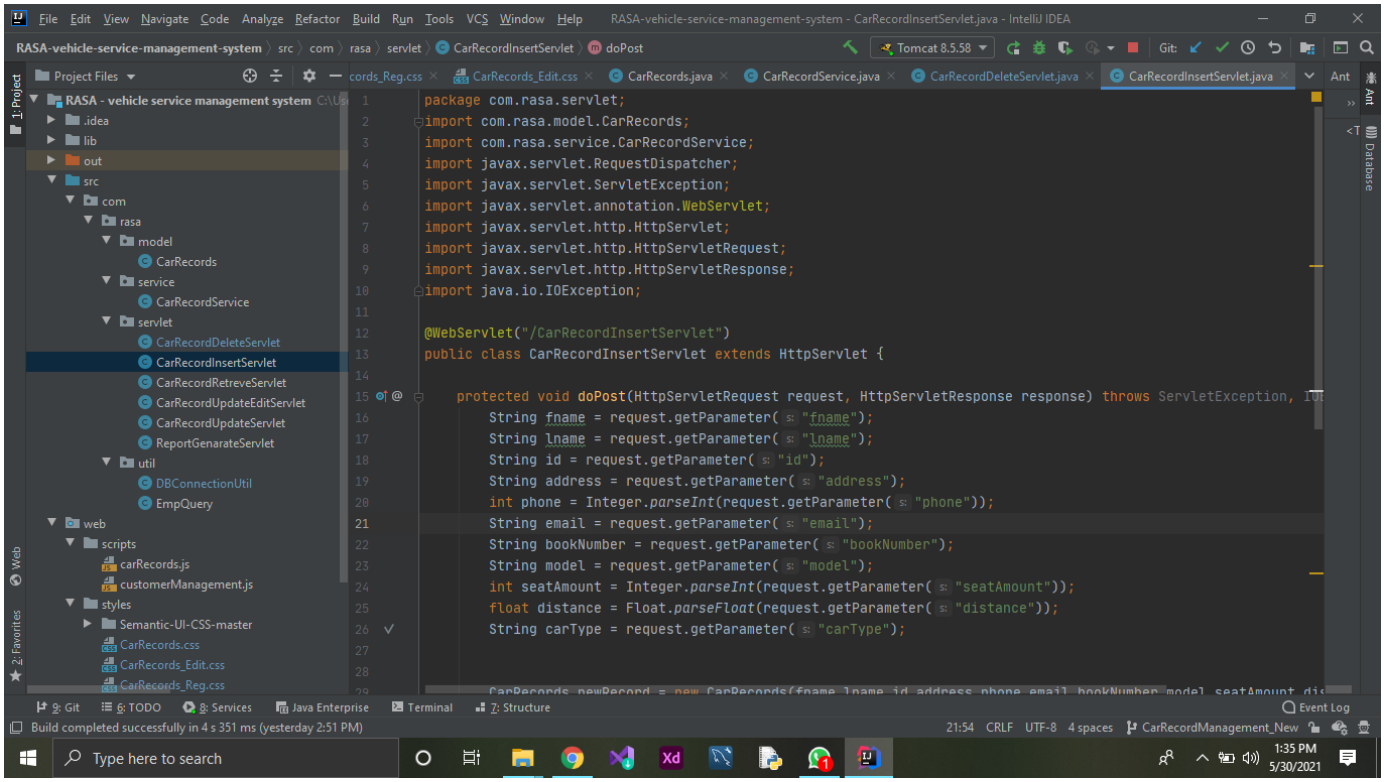
Car records service *Figure 79*



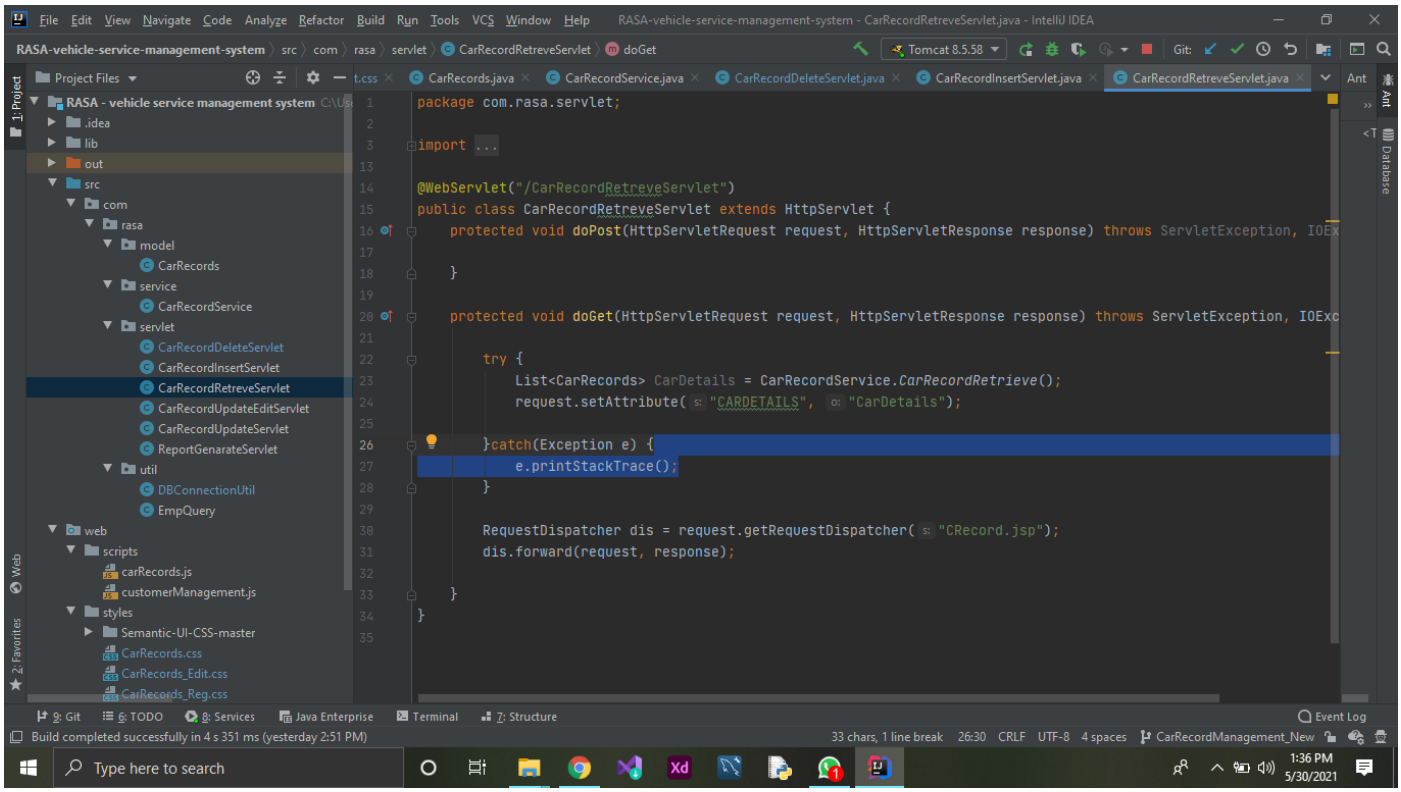
Car Records delete servlet *Figure 80*



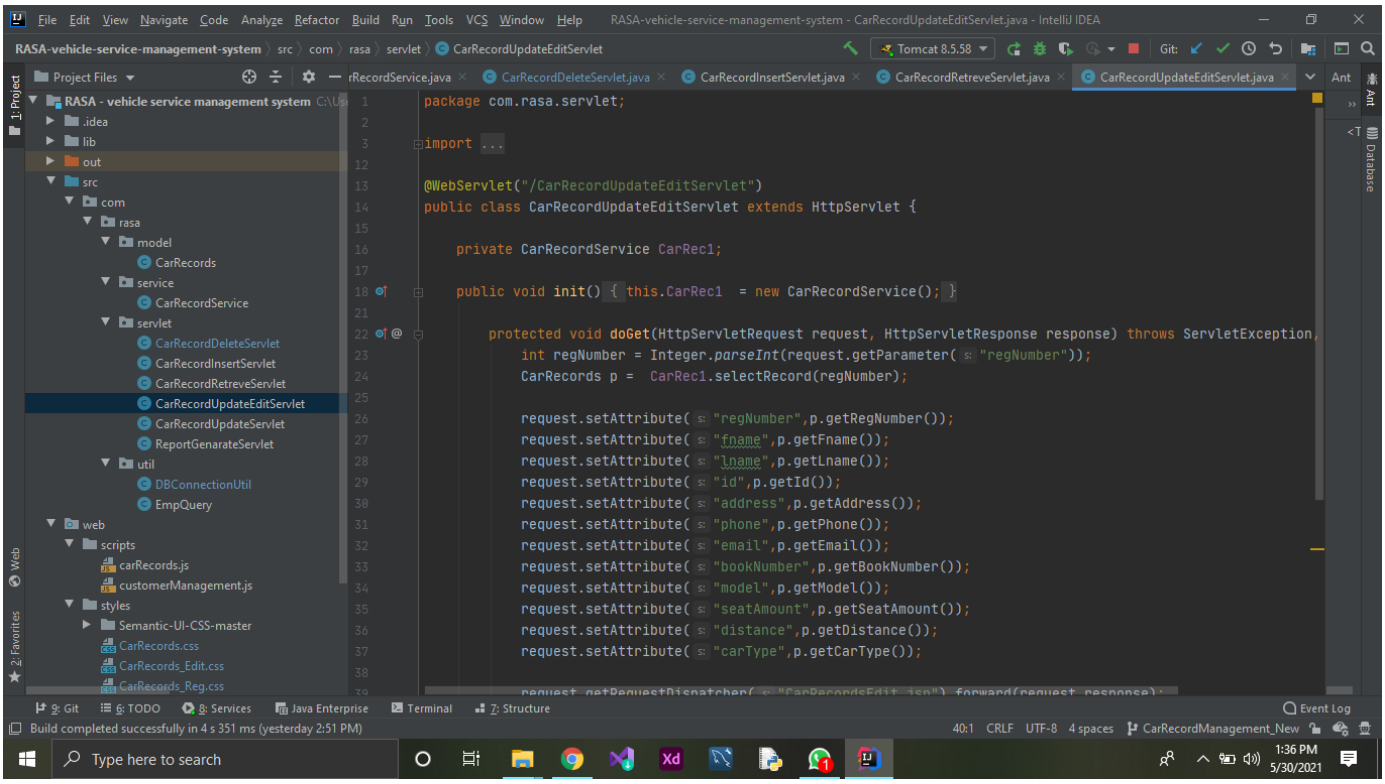
Car records Insert servlet *Figure 81*



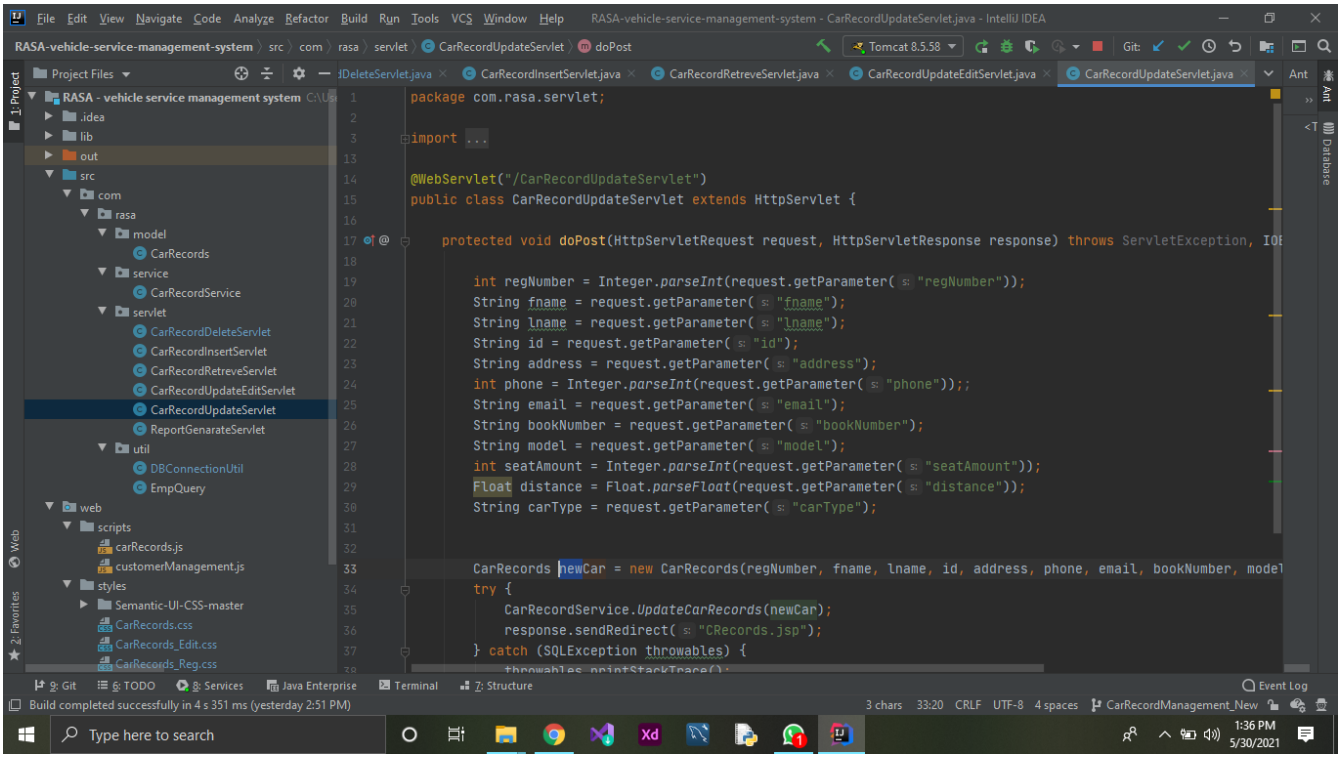
Car records retrieve servlet *Figure 82*



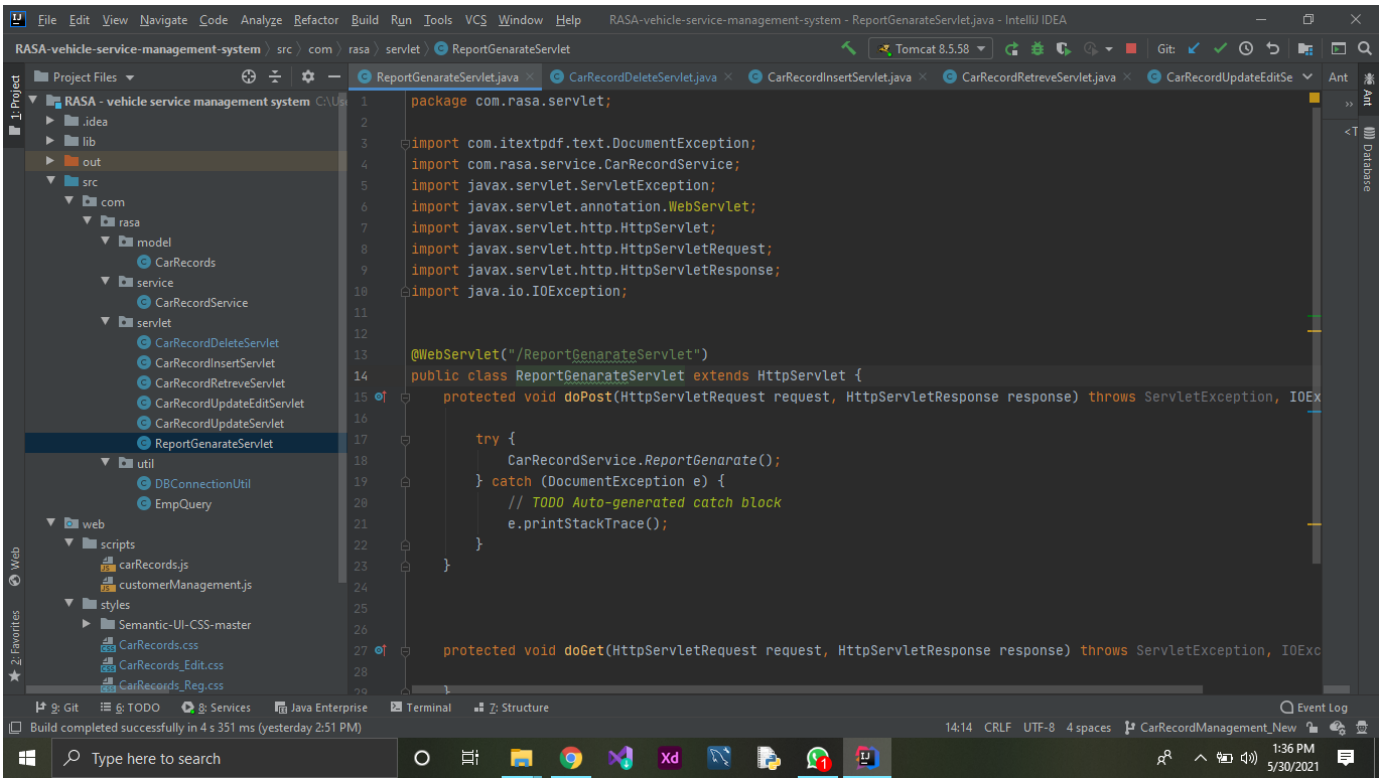
Car record update edit servlet *Figure 83*



Car record update servlet *Figure 84*



Car record delete servlet *Figure 85*



Testing

Test ID	Test Inputs	Expected Outputs	Actual Outputs	Result (Pass/ Fail)	Comments
1	Input present as the attendance status	Save the attendance and employee disappear from the attendance table	Expected Output	Pass	Add attendance works 100% accurately.
2	Real amount as Advance	Save the Advance and redirect to the employee payment page	Expected Output	Pass	Add Advance function works 100% accurately
3	Real values as bonus and deduct	Save the salary in database and redirect to the Employee Payment page	Expected output	Pass	Adding new Salary function worked 100% accurately

4	Assign employee name, Gender, Date of Birth, Basic Salary,NIC, Email, mobile, Address Employee ID is automatically generated by the system	A new employee should beadded andmake an employeeprofile.	A new employee is added, and in the database all the employee's information is kept. In the view employee tab, current employee information and the newly added can be viewed	Pass	Adding of a new employeewas executed successfully.
5	Update employee name, Gender, Date of Birth, NIC, Basic Salary, Email, contact number, Address	Relevant employee detailsare updated	Details of workers are updated and maintained in the database. In the view employee tab, current employee information and the newly updated employee details canbe viewed.	Pass	Updating employee details functions wasexecuted successfully.
6-Work Progress Management	Service name = remove and refitting status = on progress	Data display successfully	Expected output	Pass	Service details added successfully and validated successfully
7-Work Progress Management	Component name = bonnet Estimate amount = 1000.00 Component name = front buffer	Total estimate amount = 3000.00 Total component = 2	Expected output	Pass	Calculation works 100% accurately

	Estimate amount = 2000.00				
8-Car Record Management	Assign rental details first name, last name, email, mobile, address, pick up date, drop off date, rental price.	A new rental details record should be added	Expected output	Pass	Rental details added 100% accurately
9- Car Rental Management	Update first name, last name, email, mobile, address, pick up date, drop off date, rental price.	Relevant rental details are updated	Expected output	Pass	Rental details updated successfully
10 – Financial Management	Year = 2021 Month = “May”	Profit = Rs:5000	Profit = Rs:5000	Pass	Profit is correctly calculated according to given year and month

11-Financial Management	Registration number= CBD123 Estimate amount = 20000 Cash = 20000 Payment date = 2021/05/20	Data display successfully	Expected output	pass	Payment details added and display correctly and validated succesfully

2 Conclusion

The web-based system of Vehicle Repair Management System was implemented as a replacement to the previously practiced manual process. The new system was implemented by reducing the complexity of the manual process while breaking into eight functions. The functions Customer Details management, Work progress management, Financial management, Employee Details Management,

Employee Payment Management, Car Records Management, Car Rental Management and Inventory Management. Through the system, reports can be downloaded as an electronic document which can be used to analyzed data.

In this system main objective is to handle documents of the system because this company handle lots of documents per day. From this system documents are handle more effiently.In Customer Details Management Function manage all details of the customer in accuracy. In Work progress management Function deals with services that use to repair a vehicle and estimated amount as well as with employees who are assign to for each service. In Financial Management Function manage vehicle repair payments and show monthly profit to the user with using diagrams and amounts.

From Employee Details Management will handle all the details of the employees of the company and from using Employee Payment Management function help to user to take employee attendance and store them and calculate employee's payments.

Furthermore, from Car Records Management function user can keep details of the vehicle and vehicle owner's data who given their vehicle to rent. And also Car Rental Management Function helps to handle all rental processes and check availability of the vehicles. Moreover, Inventory Management Function use to manage inventory section of the company storing supplier's data, prices of the items and quantity.