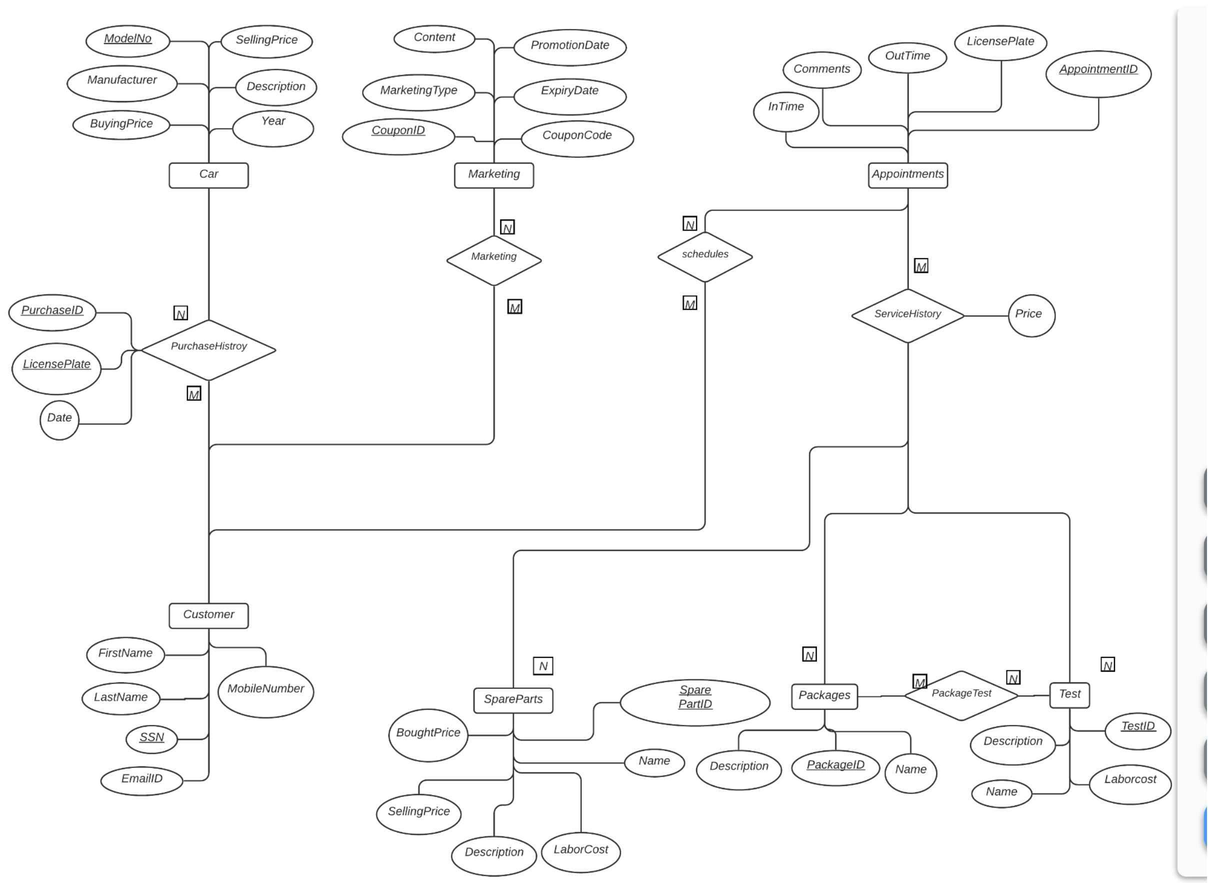
PROJECT REPORT

Computer Science 631
Database Management System Design
Project members: Nikita Ramesh Gaikwad, Rajendra Prasad Patil, Sukriti Sibal

1. Summary of System Requirements

Operating System : Windows/Linux
Environment: OpenJDK8, Tomcat 7.*.*
RAM Specifications: 1GB RAM
Storage Specifications: 50 GB HardDisk

2. ER Diagram



3 - Relational Schema Design

Entities-

Car

Marketing

Appointment

Service

Customer

Spare Part

Packages

Test

Relationships-

PurchaseHistory (Between Customer and Car)

Marketing (Between Customer and Marketing)

Schedules (Between Customer and Appointment)

ServiceHistory (Between Appointment, Package, SpareParts and Test)

PackageTest (Between Package and Test)

Step 1 - Strong Entities: Mapping all strong entities into a table.

Car (**Model No**, Buying Price, Selling Price, Manufacturer, Description, Year)

Marketing (**Coupon ID**, Marketing Type, Content, PromotionDate, ExpiryDate, CouponCode)

Appointments (**Appointment ID**, In Time, Out Time, Comment, License Plate)

Customer (**SSN**, First Name, Last Name, Email, Mobile Number)

Spare Parts (**Spare Part ID**, Bought Price, Selling Price, Description, Labor Cost, Name)

Packages (**Package ID,** Name, Description)

Test (**Test ID**, Labor Cost, Description, Name)

Step 2 - M: N Relationship: All the M: N relationships are mapped into tables.

PurchaseHistory:

PurchaseHistory (Purchase ID, Date, License Plate, Model No, SSN)

ServiceHistory:

ServiceHistory (price, Appointment ID, Spare Part ID, Package ID, Test ID)

PackageTest:

PackageTest(Package ID, Test ID, Appointment ID)

Marketing *:*

We are not handling this as of now.

Marketing(Coupon ID, SSN)

Schedules *:*

This is between customer and appointment, SSN is added to the appointments to maintain the relationship.

Schedules(Appointment ID, SSN)

Step 3: Final Schema

Car (**Model No**, Buying Price, Selling Price, Manufacturer, Description, Year)

Marketing (**Coupon ID**, Marketing Type, Content, PromotionDate, ExpiryDate, CouponCode)

Appointments (**Appointment ID**, In Time, Out Time, Comment, License Plate)

Customer (**SSN**, First Name, Last Name, Email, Mobile Number)

Spare Parts (**Spare Part ID**, Bought Price, Selling Price, Description, Labor Cost, Name)

Packages (**Package ID,** Name, Description)

Test (**Test ID**, Labor Cost, Description, Name)

PurchaseHistory (Purchase ID, Date, License Plate, Model No, SSN)

ServiceHistory (price, Appointment ID, Spare Part ID, Package ID, Test ID)

PackageTest(Package ID, Test ID, Appointment ID)

Marketing(Coupon ID, SSN)

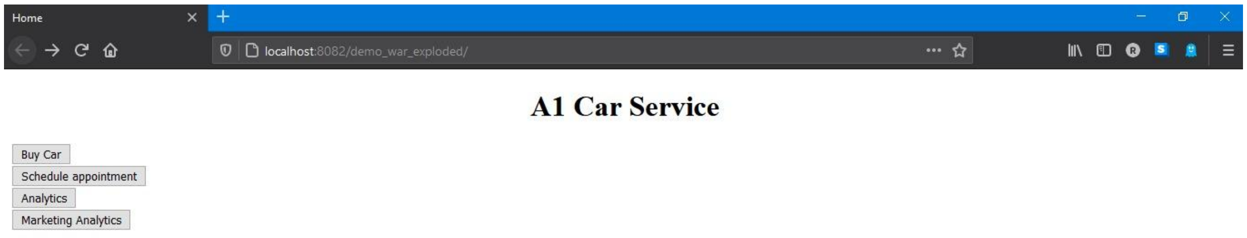
Schedules(Appointment ID, SSN)

4. Application Program Design

5. User Manual

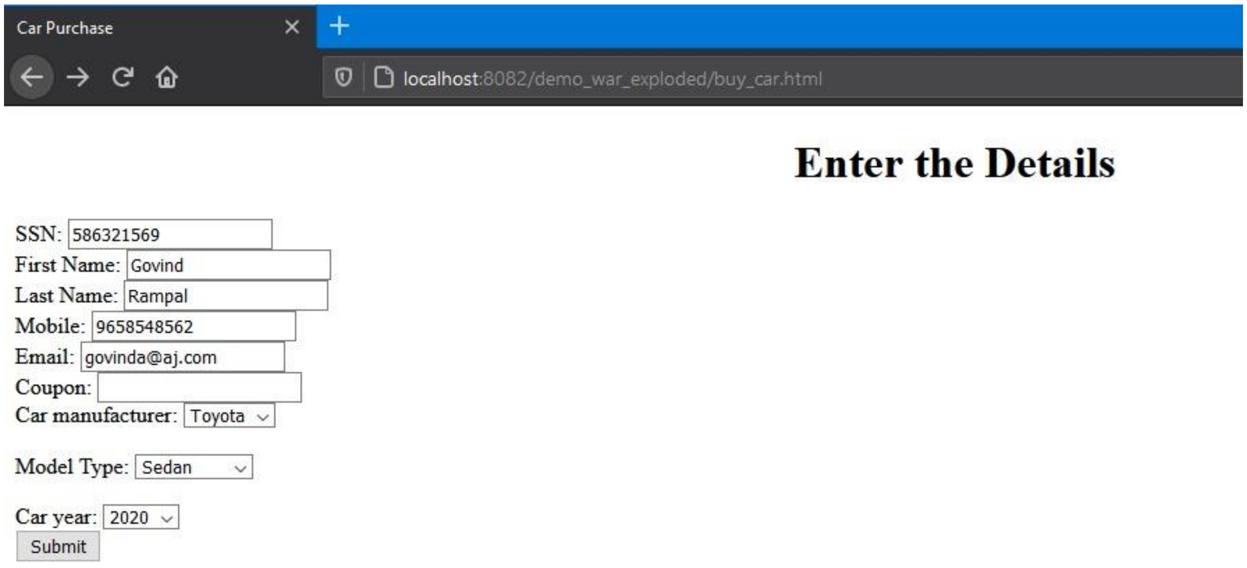
Home Screen

This is main home screen(page) which includes “Buy car”, “Schedule appointment”, “Analytics”, and “Marketing Analytics”



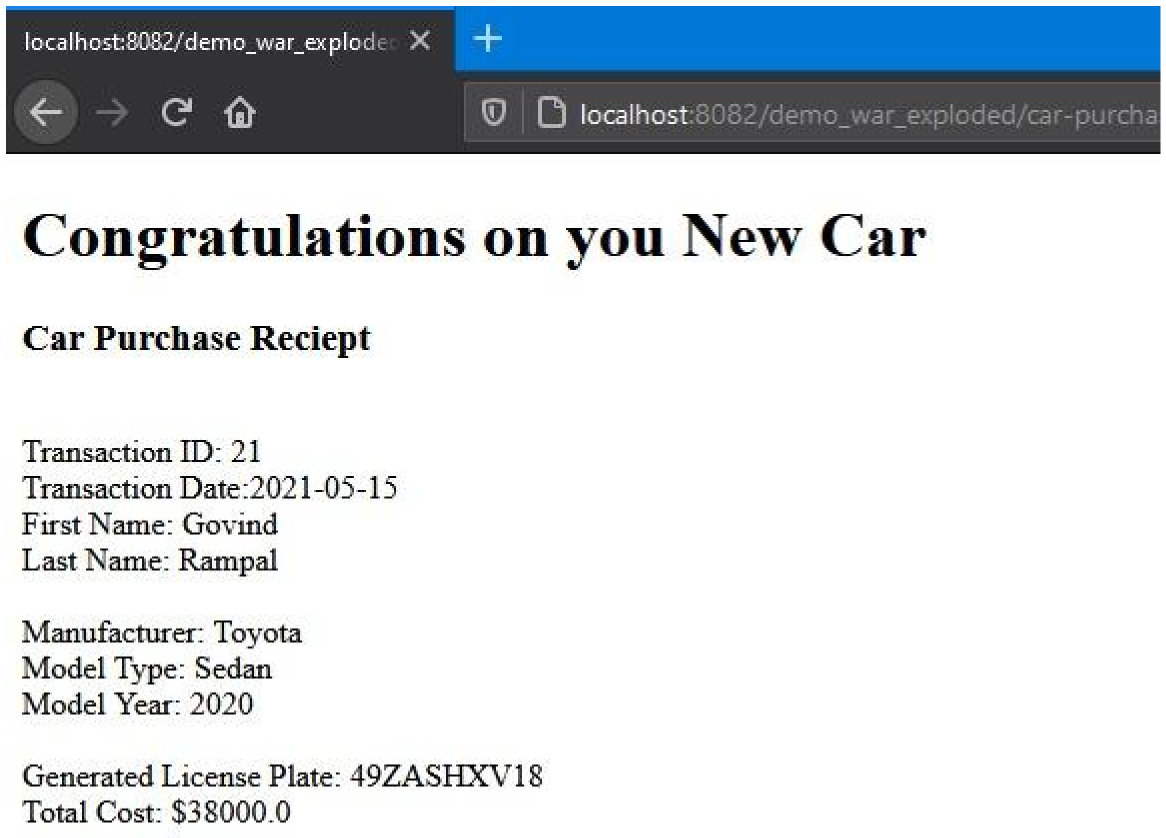
Buy Car Page

Whenever a customer buys a car, the following details are displayed in the database which are as follows “SSN”, “First Name”, “Last Name”, “Mobile”, “Email”, “Coupon”, “Car Manufacturer”, “Car Model” and “Car Year”.



SSN: 586321569
First Name: Govind
Last Name: Rampal
Mobile: 9658548562
Email: govinda@aj.com
Coupon:
Car manufacturer: Toyota
Model Type: Sedan
Car year: 2020

Receipt generated after buying a car



localhost:8082/demo_war_explodec

localhost:8082/demo_war_exploded/car-purcha

Congratulations on you New Car

Car Purchase Reciept

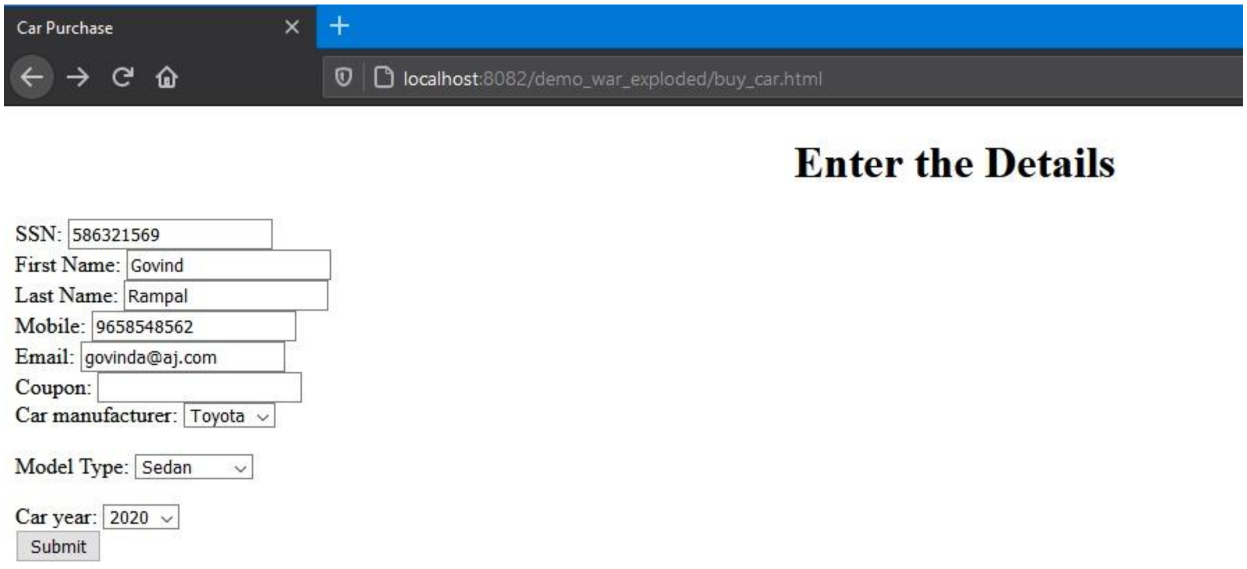
Transaction ID: 21
Transaction Date:2021-05-15
First Name: Govind
Last Name: Rampal

Manufacturer: Toyota
Model Type: Sedan
Model Year: 2020

Generated License Plate: 49ZASHXV18
Total Cost: $38000.0

Service Booking Page

After the customer books an appointment following details as entered in the database such as “SSN”, “First Name”, “Last Name”, “Mobile”, “Email”, “Plate Number”, “In Time”, “Out Time”, “Coupon”, “Package”, “Tests”, “Spare Parts to be replaced”

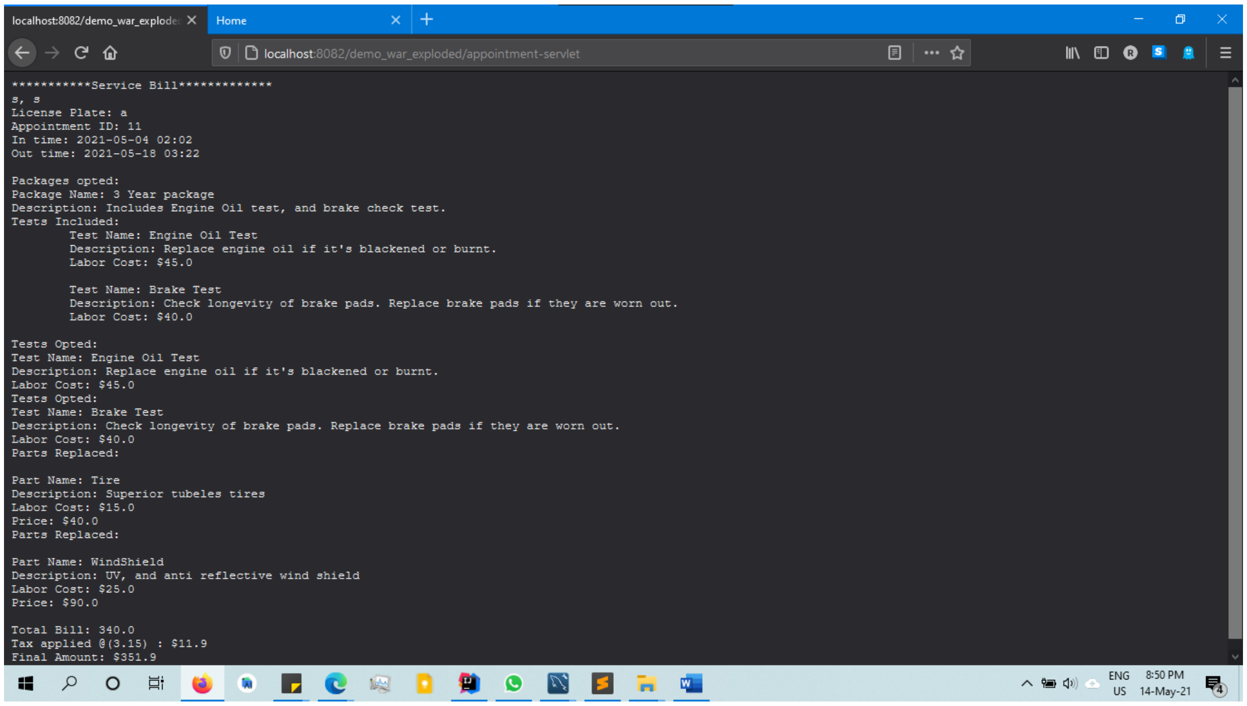


Enter the Details

SSN:	586321569
First Name:	Govind
Last Name:	Rampal
Mobile:	9658548562
Email:	govinda@aj.com
Coupon:	
Car manufacturer:	Toyota
Model Type:	Sedan
Car year:	2020

Service Booking Confirmation Page

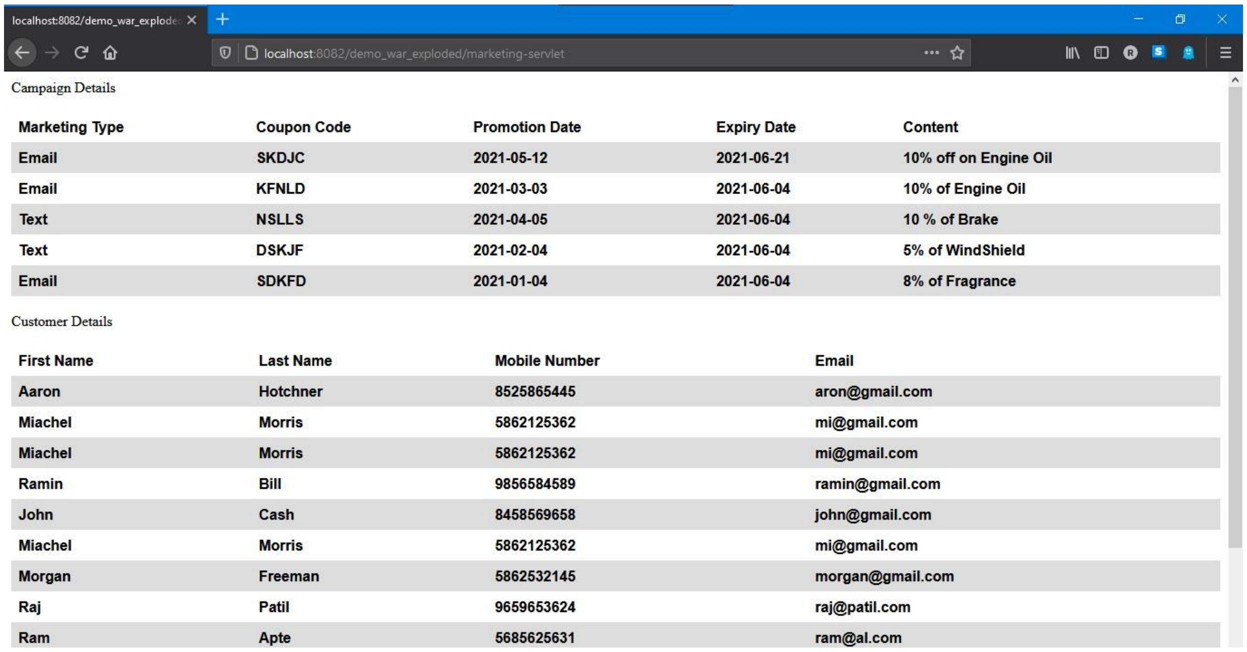
Once the customer confirms the service that has been taken, it generates the car details and service that has been given and further the bill is generated.


localhost:8082/demo_war_exploded
Home
localhost:8082/demo_war_exploded/appointment-servlet

```
***********Service Bill*************
s, s
License Plate: a
Appointment ID: 11
In time: 2021-05-04 02:02
Out time: 2021-05-18 03:22

Packages opted:
Package Name: 3 Year package
Description: Includes Engine Oil test, and brake check test.
Tests Included:
        Test Name: Engine Oil Test
        Description: Replace engine oil if it's blackened or burnt.
        Labor Cost: $45.0

        Test Name: Brake Test
        Description: Check longevity of brake pads. Replace brake pads if they are worn out.
        Labor Cost: $40.0

Tests Opted:
Test Name: Engine Oil Test
Description: Replace engine oil if it's blackened or burnt.
Labor Cost: $45.0
Tests Opted:
Test Name: Brake Test
Description: Check longevity of brake pads. Replace brake pads if they are worn out.
Labor Cost: $40.0
Parts Replaced:

Part Name: Tire
Description: Superior tubeles tires
Labor Cost: $15.0
Price: $40.0
Parts Replaced:

Part Name: WindShield
Description: UV, and anti reflective wind shield
Labor Cost: $25.0
Price: $90.0

Total Bill: 340.0
Tax applied @(3.15) : $11.9
Final Amount: $351.9
```

Marketing Campaign Page:

It displays the different marketing campaigns and various attributes associated with it, such as “Coupon Code‘, “Coupon start date”, “Coupon expiry date”, and “Coupon description”.



localhost:8082/demo_war_exploded/marketing-servlet

Campaign Details

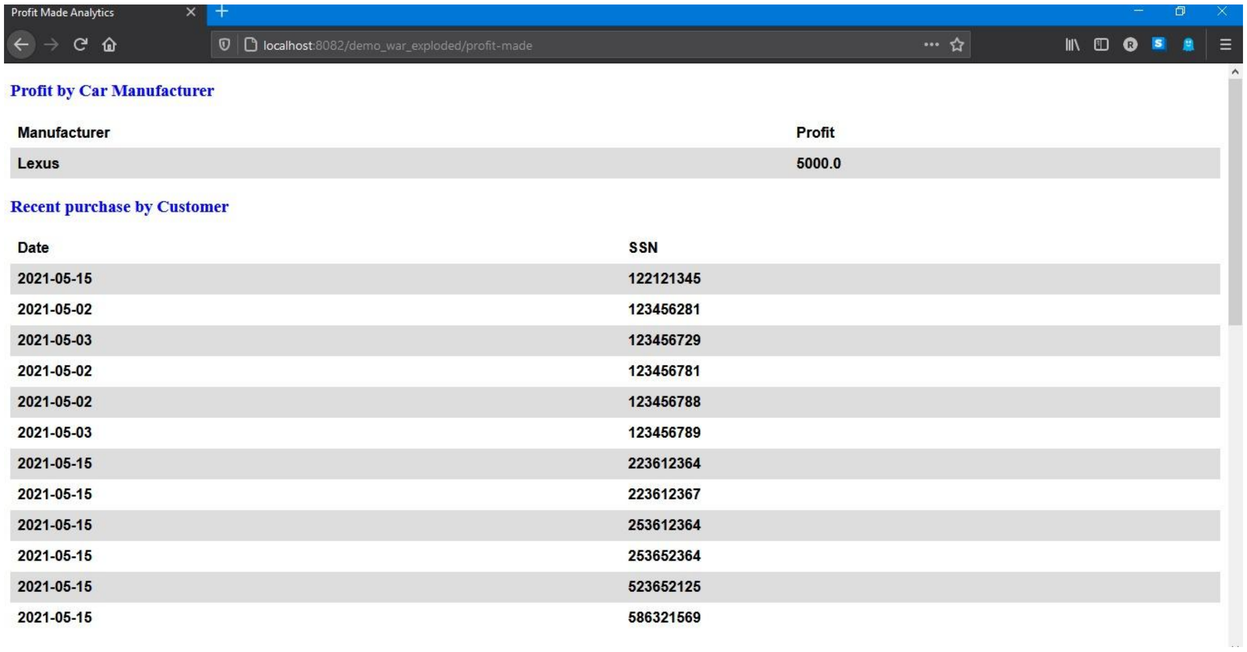
Marketing Type	Coupon Code	Promotion Date	Expiry Date	Content
Email	SKDJC	2021-05-12	2021-06-21	10% off on Engine Oil
Email	KFNLD	2021-03-03	2021-06-04	10% of Engine Oil
Text	NSLLS	2021-04-05	2021-06-04	10 % of Brake
Text	DSKJF	2021-02-04	2021-06-04	5% of WindShield
Email	SDKFD	2021-01-04	2021-06-04	8% of Fragrance

Customer Details

First Name	Last Name	Mobile Number	Email
Aaron	Hotchner	8525865445	aron@gmail.com
Miachel	Morris	5862125362	mi@gmail.com
Miachel	Morris	5862125362	mi@gmail.com
Ramin	Bill	9856584589	ramin@gmail.com
John	Cash	8458569658	john@gmail.com
Miachel	Morris	5862125362	mi@gmail.com
Morgan	Freeman	5862532145	morgan@gmail.com
Raj	Patil	9659653624	raj@patil.com
Ram	Apte	5685625631	ram@al.com

Profit Analytics

Profit given by Car Manufacturer, Recent purchase by customer, Cars purchased by each customer and amount spent by each customer.



Profit Made Analytics

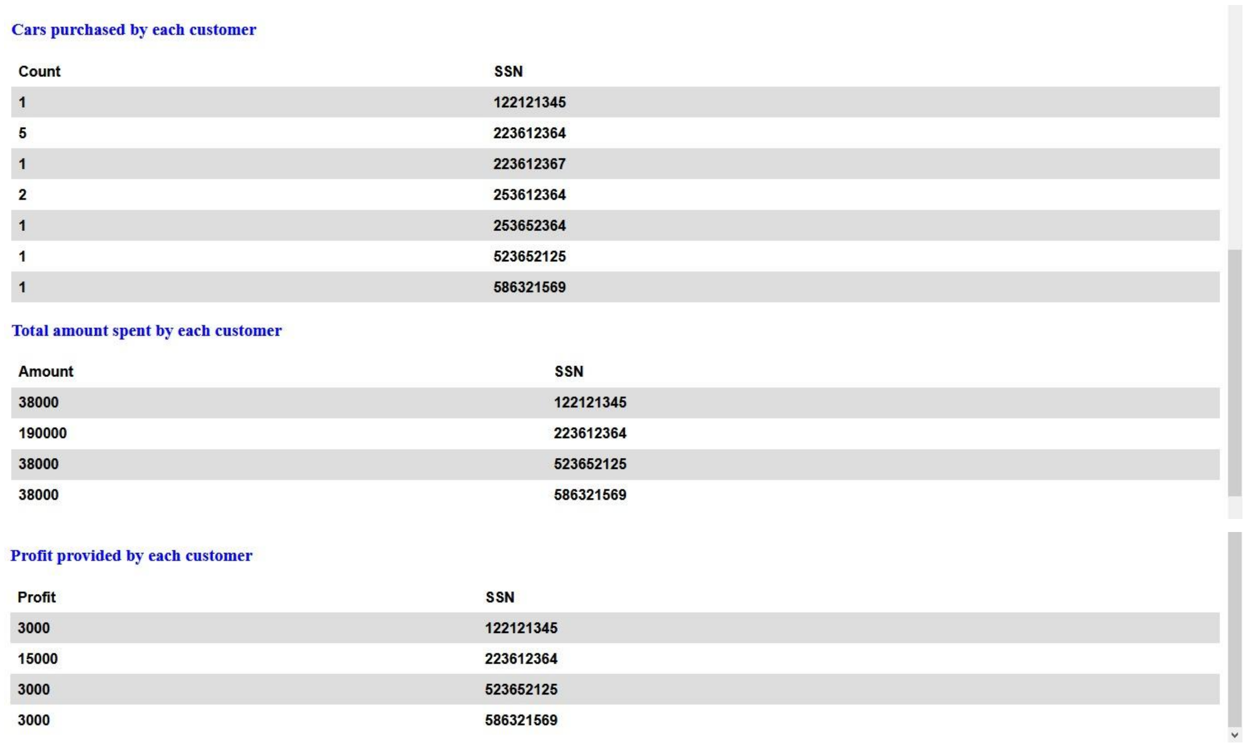
localhost:8082/demo_war_exploded/profit-made

Profit by Car Manufacturer

Manufacturer	Profit
Lexus	5000.0

Recent purchase by Customer

Date	SSN
2021-05-15	122121345
2021-05-02	123456281
2021-05-03	123456729
2021-05-02	123456781
2021-05-02	123456788
2021-05-03	123456789
2021-05-15	223612364
2021-05-15	223612367
2021-05-15	253612364
2021-05-15	253652364
2021-05-15	523652125
2021-05-15	586321569



Cars purchased by each customer

Count	SSN
1	122121345
5	223612364
1	223612367
2	253612364
1	253652364
1	523652125
1	586321569

Total amount spent by each customer

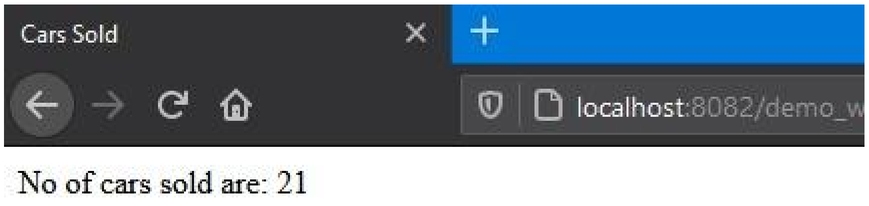
Amount	SSN
38000	122121345
190000	223612364
38000	523652125
38000	586321569

Profit provided by each customer

Profit	SSN
3000	122121345
15000	223612364
3000	523652125
3000	586321569

Car Analytics

It depicts the number of car sold

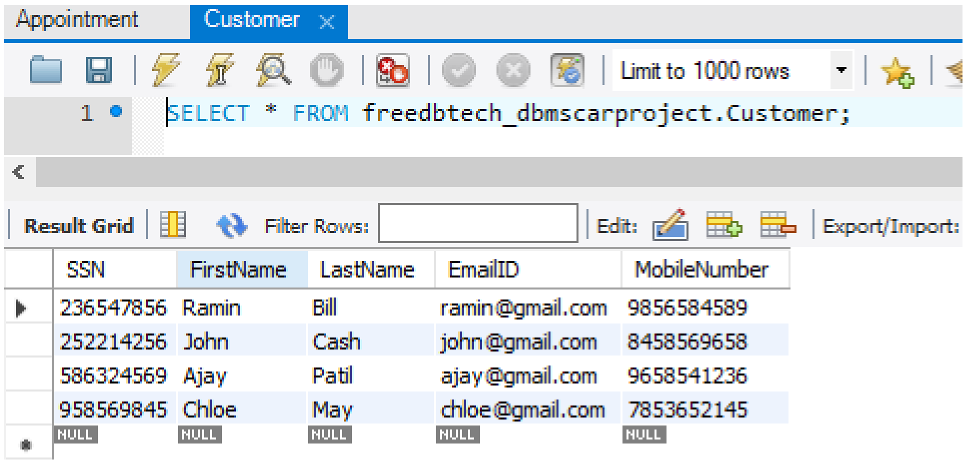


No of cars sold are: 21

6. Appendix

Database

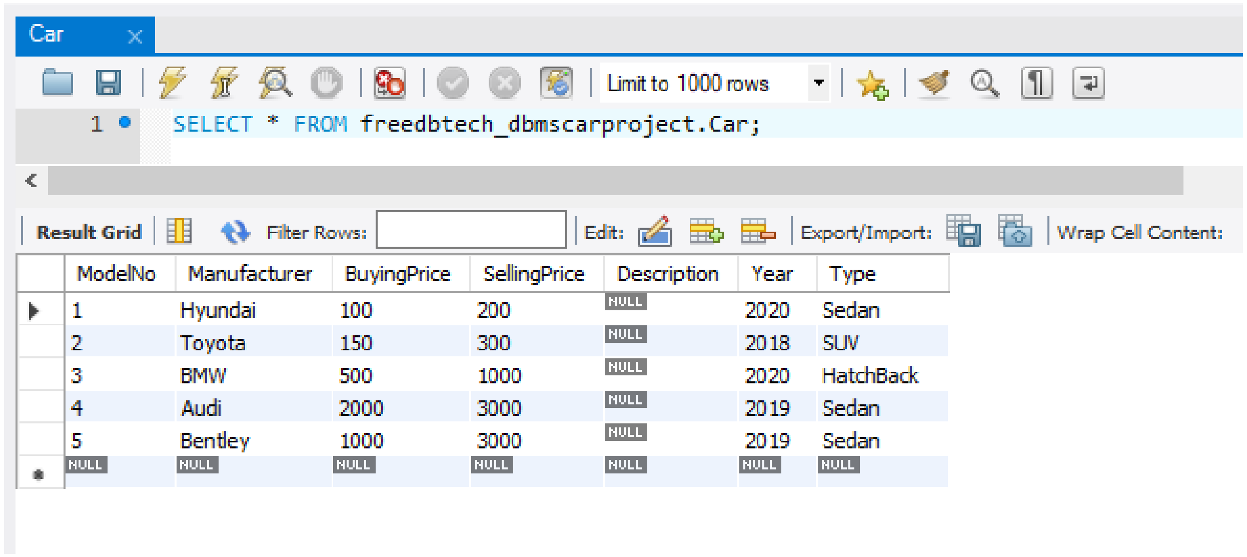
Customer



```
SELECT * FROM freedbtech_dbmscarproject.Customer;
```

SSN	FirstName	LastName	EmailID	MobileNumber
236547856	Ramin	Bill	ramin@gmail.com	9856584589
252214256	John	Cash	john@gmail.com	8458569658
586324569	Ajay	Patil	ajay@gmail.com	9658541236
958569845	Chloe	May	chloe@gmail.com	7853652145
NULL	NULL	NULL	NULL	NULL

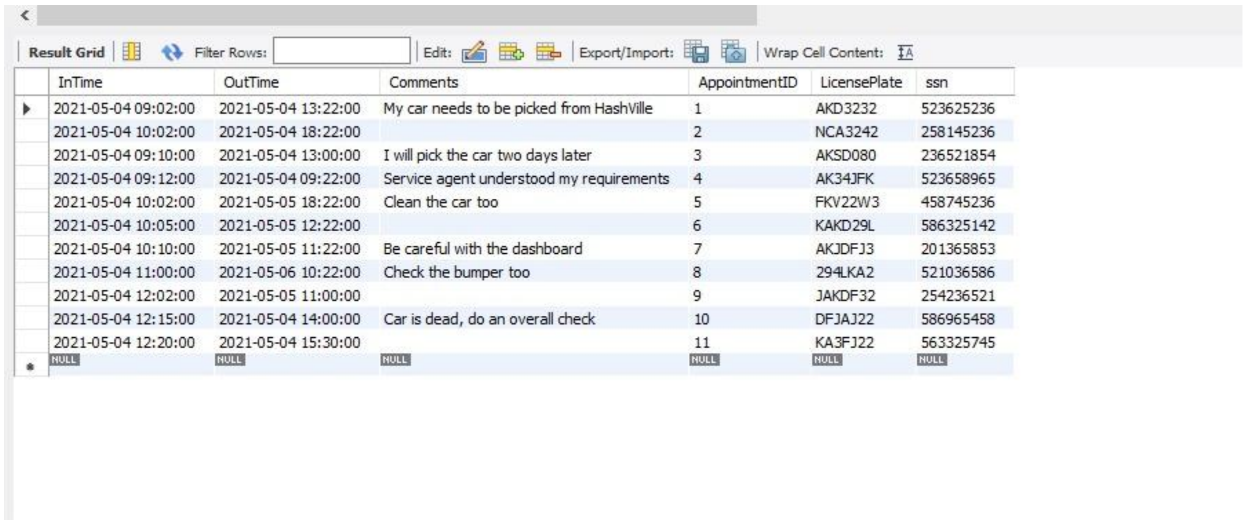
Car



```
SELECT * FROM freedbtech_dbmscarproject.Car;
```

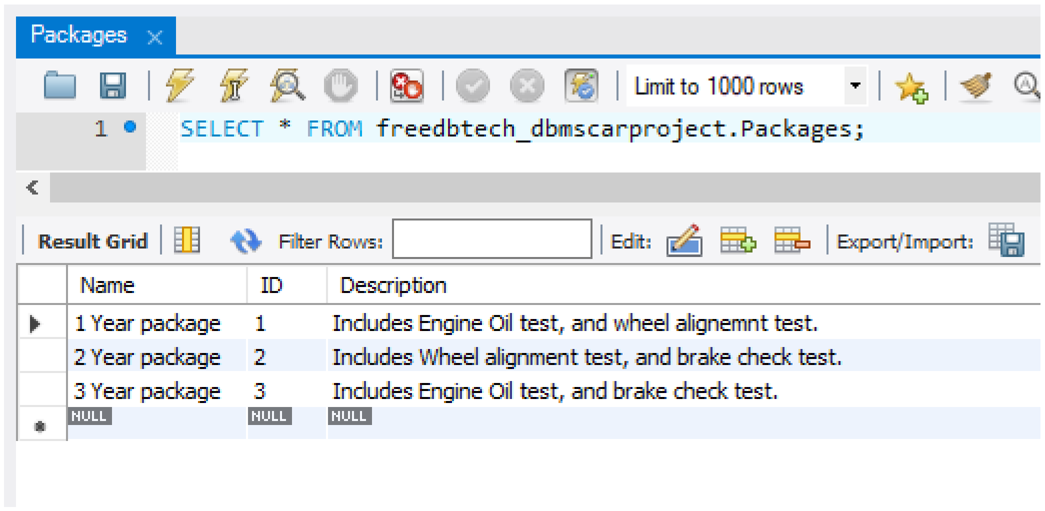
ModelNo	Manufacturer	BuyingPrice	SellingPrice	Description	Year	Type
1	Hyundai	100	200	NULL	2020	Sedan
2	Toyota	150	300	NULL	2018	SUV
3	BMW	500	1000	NULL	2020	HatchBack
4	Audi	2000	3000	NULL	2019	Sedan
5	Bentley	1000	3000	NULL	2019	Sedan
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Appointment



InTime	OutTime	Comments	AppointmentID	LicensePlate	ssn
2021-05-04 09:02:00	2021-05-04 13:22:00	My car needs to be picked from HashVille	1	AKD3232	523625236
2021-05-04 10:02:00	2021-05-04 18:22:00		2	NCA3242	258145236
2021-05-04 09:10:00	2021-05-04 13:00:00	I will pick the car two days later	3	AKSD080	236521854
2021-05-04 09:12:00	2021-05-04 09:22:00	Service agent understood my requirements	4	AK34JFK	523658965
2021-05-04 10:02:00	2021-05-05 18:22:00	Clean the car too	5	FKV22W3	458745236
2021-05-04 10:05:00	2021-05-05 12:22:00		6	KAKD29L	586325142
2021-05-04 10:10:00	2021-05-05 11:22:00	Be careful with the dashboard	7	AKJDFJ3	201365853
2021-05-04 11:00:00	2021-05-06 10:22:00	Check the bumper too	8	294LKA2	521036586
2021-05-04 12:02:00	2021-05-05 11:00:00		9	JAKDF32	254236521
2021-05-04 12:15:00	2021-05-04 14:00:00	Car is dead, do an overall check	10	DFJAJ22	586965458
2021-05-04 12:20:00	2021-05-04 15:30:00		11	KA3FJ22	563325745
NULL	NULL	NULL	NULL	NULL	NULL

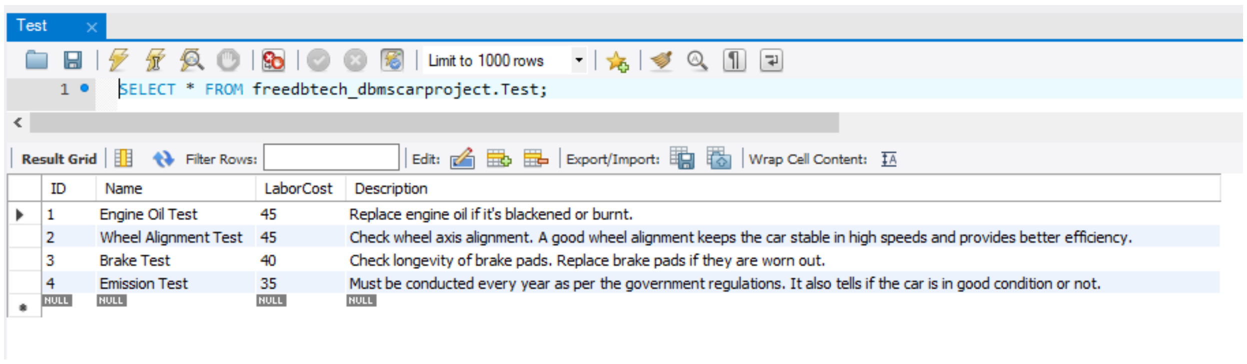
Packages



```
SELECT * FROM freedbtech_dbmscarproject.Packages;
```

Name	ID	Description
1 Year package	1	Includes Engine Oil test, and wheel alignemnt test.
2 Year package	2	Includes Wheel alignment test, and brake check test.
3 Year package	3	Includes Engine Oil test, and brake check test.
NULL	NULL	NULL

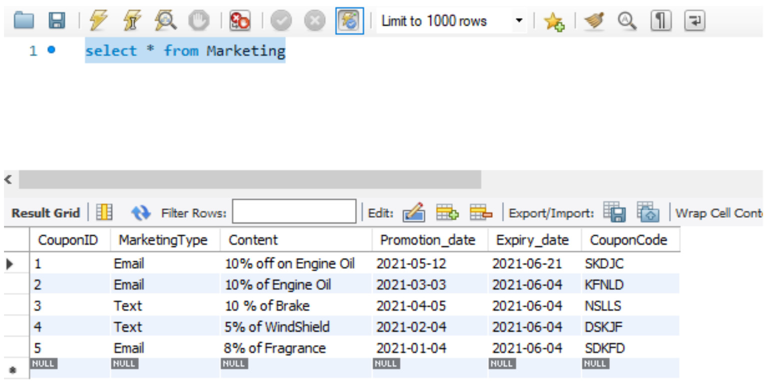
Test



```
SELECT * FROM freedbtech_dbmscarproject.Test;
```

ID	Name	LaborCost	Description
1	Engine Oil Test	45	Replace engine oil if it's blackened or burnt.
2	Wheel Alignment Test	45	Check wheel axis alignment. A good wheel alignment keeps the car stable in high speeds and provides better efficiency.
3	Brake Test	40	Check longevity of brake pads. Replace brake pads if they are worn out.
4	Emission Test	35	Must be conducted every year as per the government regulations. It also tells if the car is in good condition or not.
NULL	NULL	NULL	NULL

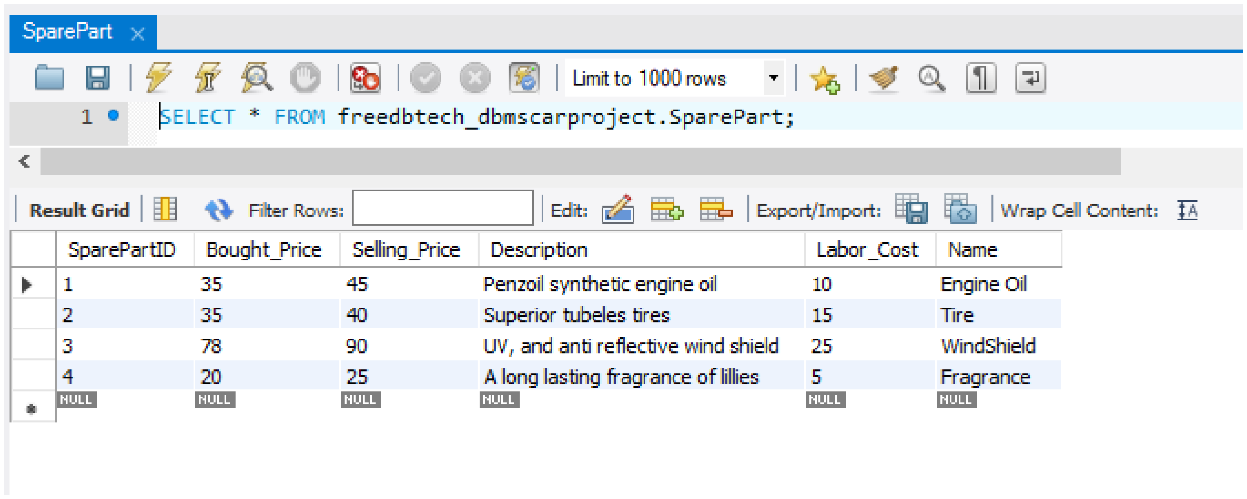
Marketing



```
select * from Marketing
```

CouponID	MarketingType	Content	Promotion_date	Expiry_date	CouponCode
1	Email	10% off on Engine Oil	2021-05-12	2021-06-21	SKDJC
2	Email	10% of Engine Oil	2021-03-03	2021-06-04	KFNLD
3	Text	10 % of Brake	2021-04-05	2021-06-04	NSLLS
4	Text	5% of WindShield	2021-02-04	2021-06-04	DSKJF
5	Email	8% of Fragrance	2021-01-04	2021-06-04	SDKFD
NULL	NULL	NULL	NULL	NULL	NULL

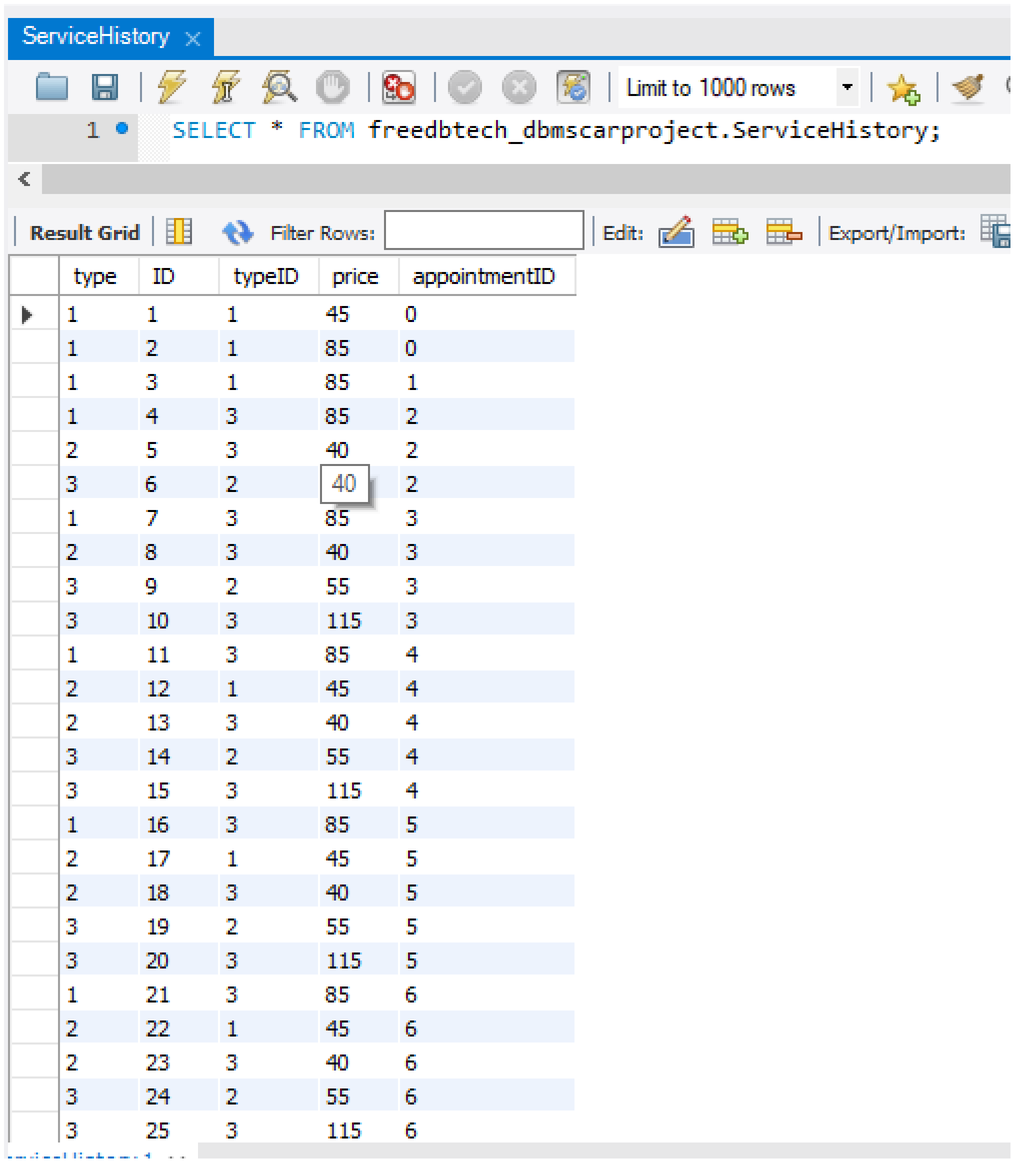
Spare Part



```
SELECT * FROM freedbtech_dbmscarproject.SparePart;
```

SparePartID	Bought_Price	Selling_Price	Description	Labor_Cost	Name
1	35	45	Penzoil synthetic engine oil	10	Engine Oil
2	35	40	Superior tubeles tires	15	Tire
3	78	90	UV, and anti reflective wind shield	25	WindShield
4	20	25	A long lasting fragrance of lillies	5	Fragrance
NULL	NULL	NULL	NULL	NULL	NULL

Service History

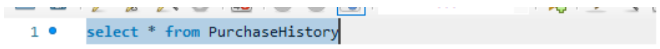


```
SELECT * FROM freedbtech_dbmscarproject.ServiceHistory;
```

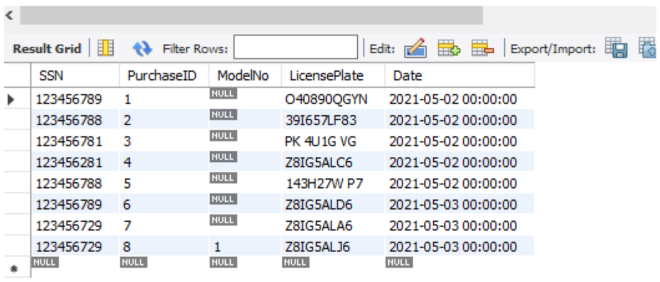
type	ID	typeID	price	appointmentID
1	1	1	45	0
1	2	1	85	0
1	3	1	85	1
1	4	3	85	2
2	5	3	40	2
3	6	2	40	2
1	7	3	85	3
2	8	3	40	3
3	9	2	55	3
3	10	3	115	3
1	11	3	85	4
2	12	1	45	4
2	13	3	40	4
3	14	2	55	4
3	15	3	115	4
1	16	3	85	5
2	17	1	45	5
2	18	3	40	5
3	19	2	55	5
3	20	3	115	5
1	21	3	85	6
2	22	1	45	6
2	23	3	40	6
3	24	2	55	6
3	25	3	115	6

Purchase History

Insert



```
select * from PurchaseHistory
```



SSN	PurchaseID	ModelNo	LicensePlate	Date
123456789	1	NULL	O40890QGYN	2021-05-02 00:00:00
123456788	2	NULL	39I657LF83	2021-05-02 00:00:00
123456781	3	NULL	PK 4U1G VG	2021-05-02 00:00:00
123456281	4	NULL	Z8IG5ALC6	2021-05-02 00:00:00
123456788	5	NULL	143H27W P7	2021-05-02 00:00:00
123456789	6	NULL	Z8IG5ALD6	2021-05-03 00:00:00
123456729	7	NULL	Z8IG5ALA6	2021-05-03 00:00:00
123456729	8	1	Z8IG5ALJ6	2021-05-03 00:00:00
NULL	NULL	NULL	NULL	NULL

Insert Statements-

Appointment Table-

```
CREATE TABLE `Appointment` (
  `InTime` datetime NOT NULL,
  `OutTime` datetime NOT NULL,
  `Comments` mediumtext,
  `AppointmentID` int(11) NOT NULL AUTO_INCREMENT,
  `LicensePlate` varchar(10) NOT NULL,
  `ssn` int(9) NOT NULL,
  PRIMARY KEY (`AppointmentID`),
  UNIQUE KEY `AppointmentID_UNIQUE` (`AppointmentID`)
) ENGINE=InnoDB AUTO_INCREMENT=12 DEFAULT CHARSET=latin1;
```

Car Table

```
CREATE TABLE `Car` (
  `ModelNo` varchar(25) NOT NULL,
  `Manufacturer` varchar(45) DEFAULT NULL,
  `BuyingPrice` float DEFAULT NULL,
  `SellingPrice` float DEFAULT NULL,
  `Description` mediumtext,
  `Year` int(4) NOT NULL,
  `Type` varchar(45) DEFAULT NULL,
  PRIMARY KEY (`ModelNo`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Customer Table

```
CREATE TABLE `Customer` (
  `SSN` int(9) NOT NULL,
  `FirstName` varchar(45) DEFAULT NULL,
  `LastName` varchar(45) DEFAULT NULL,
  `EmailID` varchar(45) DEFAULT NULL,
  `MobileNumber` varchar(10) DEFAULT NULL,
  PRIMARY KEY (`SSN`),
  UNIQUE KEY `SSN_UNIQUE` (`SSN`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Packages Table

```
CREATE TABLE `Packages` (
  `Name` varchar(45) NOT NULL,
  `ID` int(11) NOT NULL,
  `Description` varchar(145) DEFAULT NULL,
  PRIMARY KEY (`ID`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Marketing Table-

```
CREATE TABLE `Marketing` (
  `CouponID` int(11) NOT NULL,
  `MarketingType` varchar(45) DEFAULT NULL,
  `Content` varchar(45) DEFAULT NULL,
  `Promotion_date` date DEFAULT NULL,
  `Expiry_date` date DEFAULT NULL,
  `CouponCode` varchar(45) DEFAULT NULL,
  PRIMARY KEY (`CouponID`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Service History

```
CREATE TABLE `ServiceHistory` (
  `type` int(1) NOT NULL,
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `typeID` int(11) NOT NULL,
  `price` float NOT NULL,
  `appointmentID` int(11) NOT NULL,
  PRIMARY KEY (`ID`)
) ENGINE=InnoDB AUTO_INCREMENT=51 DEFAULT CHARSET=latin1;
```

Spare Part

```
CREATE TABLE `SparePart` (
  `SparePartID` int(11) NOT NULL,
  `Bought_Price` double NOT NULL,
  `Selling_Price` double NOT NULL,
  `Description` varchar(45) NOT NULL,
  `Labor_Cost` double NOT NULL,
  `Name` varchar(45) NOT NULL,
  PRIMARY KEY (`SparePartID`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Test Table

```
CREATE TABLE `Test` (
  `ID` int(11) NOT NULL,
  `Name` varchar(30) NOT NULL,
  `LaborCost` float NOT NULL,
  `Description` varchar(145) DEFAULT NULL,
  PRIMARY KEY (`ID`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Package Test

```
CREATE TABLE `PackageTest` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `PackageID` int(11) DEFAULT NULL,
  `TestID` int(11) DEFAULT NULL,
  PRIMARY KEY (`ID`)
) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=latin1;
```

Purchase History

```
CREATE TABLE `PurchaseHistory` (
  `SSN` int(9) DEFAULT NULL,
  `PurchaseID` int(11) NOT NULL AUTO_INCREMENT,
  `ModelNo` varchar(8) DEFAULT NULL,
  `LicensePlate` varchar(10) NOT NULL,
  `Date` datetime DEFAULT NULL,
  PRIMARY KEY (`PurchaseID`,`LicensePlate`),
  UNIQUE KEY `LicensePlate_UNIQUE` (`LicensePlate`),
  UNIQUE KEY `PurchaseID_UNIQUE` (`PurchaseID`)
) ENGINE=InnoDB AUTO_INCREMENT=9 DEFAULT CHARSET=latin1;
```

Source Code

Servlets

Marketing Servlet

```
package com.nikrajsuk.crm.servlets;

import com.nikrajsuk.crm.qm.MarketingQm;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;

@WebServlet(name = "marketingServlet", value = "/marketing-servlet")
public class MarketingSvlt extends HttpServlet {

   @Override
   protected void doPost(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
       String message = getCampaignDetails();
       sendOutput(resp, message);
   }

   public String getCampaignDetails() {
       return new MarketingQm().fetchCampaignDetails();
   }

   public void sendOutput(HttpServletResponse resp, String outputString)
throws IOException {
       PrintWriter out = resp.getWriter();
       String ot =
"<!DOCTYPEhtml><html><head><style>table{font-family:arial,sans-serif;border-c
ollapse:collapse;width:100%;}td,th{border:1pxsolid#dddddd;text-align:left;pad
ding:8px;}tr:nth-child(even){background-color:#dddddd;}</style></head>"
               + outputString + "</body></html>\n";

       out.print(ot);

   }

}
```

Appointment Servlet

```
//
// Source code recreated from a .class file by IntelliJ IDEA
// (powered by FernFlower decompiler)
```

```
//

package com.nikrajsuk.crm.servlets;

import com.nikrajsuk.crm.qm.AppointmentQm;
import com.nikrajsuk.crm.qm.CustomerQm;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet(
  name = "appointmentServlet",
  value = {"/appointment-servlet"}
)
public class AppointmentSvlt extends HttpServlet {
  public AppointmentSvlt() {
  }

  protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, IOException {
      String ssn = req.getParameter("SSN");
      String fname = req.getParameter("fname");
      String lname = req.getParameter("lname");
      String mobileNum = req.getParameter("mobile");
      String emailID = req.getParameter("Email");
      String licensePlate = req.getParameter("Plate Number");
      String intime = req.getParameter("in_time_date");
      String intimetime = req.getParameter("in_time_time");
      String outtime = req.getParameter("out_time_date");
      String outtimetime = req.getParameter("out_time_time");
      ArrayList<String> pkgs = new ArrayList();
      pkgs.add(req.getParameter("pkg1"));
      pkgs.add(req.getParameter("pkg2"));
      pkgs.add(req.getParameter("pkg3"));
      ArrayList<String> tests = new ArrayList();
      tests.add(req.getParameter("test1"));
      tests.add(req.getParameter("test2"));
      tests.add(req.getParameter("test3"));
      tests.add(req.getParameter("test4"));
      ArrayList<String> parts = new ArrayList();
      parts.add(req.getParameter("part1"));
      parts.add(req.getParameter("part2"));
```

```
        parts.add(req.getParameter("part3"));
        parts.add(req.getParameter("part4"));
        String message = "";
        ArrayList<String> b = (new CustomerQm()).fetchBySSN(Integer.valueOf(ssn));
        if (b.size() == 0) {
            (new CustomerQm()).insertNewSSN(Integer.valueOf(ssn), fname, lname, emailID,
Integer.valueOf(mobileNum));
        }

        Integer appID = (new AppointmentQm()).createAppointment(Integer.valueOf(ssn), intime
+ " " + intimetime, outtime + " " + outtimetime, licensePlate, "", pkgs, tests, parts);
        message = (new AppointmentQm()).generateBill(appID, Integer.valueOf(ssn),
licensePlate, intime + " " + intimetime, outtime + " " + outtimetime, pkgs, tests, parts);
        this.sendOutput(resp, message);
    }

    public void sendOutput(HttpServletResponse resp, String outputString) throws IOException
{
        PrintWriter out = resp.getWriter();
        resp.setCharacterEncoding("UTF-8");
        out.print(outputString);
        out.flush();
    }
}
```

Customer Servlet

```
package com.nikrajsuk.crm.servlets;

import com.google.gson.Gson;
import com.google.gson.JsonObject;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;

import com.nikrajsuk.crm.objs.CustomerObj;
import com.nikrajsuk.crm.qm.*;

@WebServlet(name = "customerServlet", value = "/customer-servlet")
```

```
public class CustomerSvlt extends HttpServlet {

   @Override
   protected void doPost(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
       String reqType = req.getParameter("reqType");
       String params = req.getParameter("params");
       String message = "";
       message = processParams(reqType, params);
       sendOutput(resp, message);
   }

   public void sendOutput(HttpServletResponse resp, String outputString)
throws IOException {
       PrintWriter out = resp.getWriter();
       resp.setContentType("application/json");
       resp.setCharacterEncoding("UTF-8");
       out.print(outputString);
       out.flush();
   }

   public String processParams(String reqType, String params) {
       if (reqType.equalsIgnoreCase("getBySSN")) {
           JsonObject convertedObject = new Gson().fromJson(params,
JsonObject.class);
           String SSN = convertedObject.get("SSN").toString();
           return new CustomerQm().fetchBySSN(Integer.valueOf(SSN));
       } else {
           // new customer data is entered
           CustomerObj convertedObject = new Gson().fromJson(params,
CustomerObj.class);
           if (new CustomerQm().insertNewSSN(convertedObject) == 1) {
               JsonObject obj = new JsonObject();
               obj.addProperty("success", true);
               return obj.toString();
           } else {
               JsonObject obj = new JsonObject();
               obj.addProperty("success", false);
               return obj.toString();
           }
       }

   }
```

Purchase Servlet

```
package com.nikrajsuk.crm.servlets;
```

```java
import com.google.gson.Gson;
import com.google.gson.JsonObject;
import com.nikrajsuk.crm.objs.CustomerObj;
import com.nikrajsuk.crm.qm.CustomerQm;
import com.nikrajsuk.crm.qm.PurchasesQm;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;

@WebServlet(name = "carPurchaseServlet", value = "/car-purchase")
public class PurchaseSvlt extends HttpServlet {

   @Override
   protected void doPost(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
       String ssn = req.getParameter("ssn");
       String fname = req.getParameter("fname");
       String lname = req.getParameter("lname");
       String mobile = req.getParameter("mobile");
       String email = req.getParameter("email");
       String carManufacturer = req.getParameter("car-manufacturer");
       String carModel = req.getParameter("car-model");
       String carYear = req.getParameter("car-year");

       String message = createPurchaseEntry(ssn, fname, lname, mobile, email,
carModel);
       sendOutput(resp, message);
   }

   public void sendOutput(HttpServletResponse resp, String outputString)
throws IOException {
       String message = "<!DOCTYPE html><html><body><h1>Congratulations on
you New Car</h1><h3>Car Purchase Reciept</h3><br>Transaction ID:
$trnxId</br><br>Transaction Date: $trnxdt</br><br>SSN: $ssn</br><br>First
Name: $fname</br><br>Last Name: $lname</br><br>Email ID:
$emailid</br><br>Mobile Number: $mnum</br><br>Manufacturer: $mfr</br><br>Car
Model: $cm</br><br>Year: $yr</br></body></html> ";
       PrintWriter out = resp.getWriter();
       resp.setContentType("application/html");
       resp.setCharacterEncoding("UTF-8");
       out.print(message);
       out.flush();
   }

   public String createPurchaseEntry(String ssn, String fName, String lName,
String mobile, String email,
                                     String carModel){

       new CustomerQm().insertNewSSN(Integer.valueOf(ssn), fName, lName,
email, Integer.valueOf(mobile));
       new PurchasesQm().createPurchase(Integer.valueOf(ssn), carModel);
       return "";
   }

}
```

Cars Sold Analytics Servlet-

```
//
// Source code recreated from a .class file by IntelliJ IDEA
// (powered by FernFlower decompiler)
//

package com.nikrajsuk.crm.servlets;

import com.nikrajsuk.crm.qm.AnalyticsQm;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Date;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet(
   name = "carsSoldServlet",
   value = {"/cars-sold-analytics"}
)
public class CarsSoldAnalyticsSvlt extends HttpServlet {
   public CarsSoldAnalyticsSvlt() {
   }

   protected void doPost(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
       String from = req.getParameter("Start Date");
       String to = req.getParameter("End Date");
       String message = this.process(Date.valueOf(from), Date.valueOf(to));
       this.sendOutput(resp, message);
   }

   public void sendOutput(HttpServletResponse resp, String outputString)
throws IOException {
       outputString = "<!DOCTYPE html>\n<html lang=\"en\">\n<head>\n    <meta
charset=\"UTF-8\">\n    <title> Cars Sold</title>\n</head>\n<body>" +
outputString + "</body>\n</html>";
       PrintWriter out = resp.getWriter();
       resp.setCharacterEncoding("UTF-8");
       out.print(outputString);
       out.flush();
   }

   public String process(Date from, Date to) {
       int count = (new AnalyticsQm()).carSold(from, to);
       return "No of cars sold are: " + String.valueOf(count);
   }
}
```

SSN Servlet

```
package com.nikrajsuk.crm.servlets;

import com.nikrajsuk.crm.qm.CustomerQm;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;
import java.util.List;

@WebServlet(name = "ssnServlet", value = "/ssn-servlet")
public class SSNSvlt extends HttpServlet {

   @Override
   protected void doPost(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
       String ssn = req.getParameter("ssn");
       String message = processParams(ssn);
       sendOutput(resp, message);
   }

   public void sendOutput(HttpServletResponse resp, String outputString)
throws IOException {
       PrintWriter out = resp.getWriter();
       resp.setCharacterEncoding("UTF-8");
       resp.setContentType("application/json");
       out.print(outputString);
       out.flush();
   }

   public String processParams(String SSN){
       ArrayList<String> b =  new
CustomerQm().fetchBySSN(Integer.valueOf(SSN)  );
       String result = "";
```

```
        for(String item : b){
            result = item + "xxxx";
        }
        if (result.length() > 5)
            result = result.substring(0, result.length() - 4);
        return result + "ddd";

    }

}
```

Profit Made Servlet

```
package com.nikrajsuk.crm.servlets;

import com.nikrajsuk.crm.qm.AnalyticsQm;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Date;

@WebServlet(name = "profitMadeServlet", value = "/profit-made")
public class ProfitMadeSvlt extends HttpServlet {

   @Override
   protected void doPost(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
        String from = req.getParameter("Start Date");
        String to = req.getParameter("End Date");
        String message = process(from, to);
        sendOutput(resp, message);
   }

   public void sendOutput(HttpServletResponse resp, String outputString)
throws IOException {
        outputString = "<!DOCTYPE html>\n" +
                "<html lang=\"en\">\n" +
                "<head>\n" +
                "    <meta charset=\"UTF-8\">\n" +
                "    <title>Profit Analytics</title>\n" +
                "</head>\n" +
                "<body>" + outputString +
                "</body>\n" +
                "</html>";
        PrintWriter out = resp.getWriter();
        resp.setCharacterEncoding("UTF-8");
        out.print(outputString);
        out.flush();
    }

    public String process(String from, String to) {
        return new AnalyticsQm().profit_manufacturer(from, to);
    }


}
```

Car Sold Analytics Servlet

```
package com.nikrajsuk.crm.servlets;

import com.nikrajsuk.crm.qm.AnalyticsQm;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Date;

@WebServlet(name = "carsSoldServlet", value = "/cars-sold-analytics")
public class CarsSoldAnalyticsSvlt extends HttpServlet {

    @Override
    protected void doPost(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {

        String from = req.getParameter("Start Date");
        String to = req.getParameter("End Date");
        String message = process(Date.valueOf(from), Date.valueOf(to));
        sendOutput(resp, message);
    }

    public void sendOutput(HttpServletResponse resp, String outputString)
throws IOException {
        outputString = "<!DOCTYPE html>\n" +
                "<html lang=\"en\">\n" +
                "<head>\n" +
                "    <meta charset=\"UTF-8\">\n" +
                "    <title> Cars Sold</title>\n" +
                "</head>\n" +
                "<body>" + outputString +
                "</body>\n" +
                "</html>" ;

        PrintWriter out = resp.getWriter();
        resp.setCharacterEncoding("UTF-8");
        out.print(outputString);
        out.flush();
    }

    public String process(Date from, Date to){
        int count = new AnalyticsQm().carSold(from, to);
        return "No of cars sold are: "+String.valueOf(count);
    }

}
```

Query Manager

Analytics Query Manager

```
package com.nikrajsuk.crm.qm;

import java.sql.*;
import java.util.Random;

public class AnalyticsQm {

    public static void main(String[] args) {
        System.out.println("Hello World");

    }

    public int carSold(Date startDate, Date endDate) {
        Integer result = -1;
        new ConnectionManager().createClass();
        java.util.Date javaDate = new java.util.Date();
        java.sql.Date mySQLDate = new java.sql.Date(javaDate.getTime());

        try {
            Connection con = new ConnectionManager().connect();
            String query = "Select count(PurchaseID) as cnt from
PurchaseHistory where Date between ? and ? ";
            PreparedStatement ps = con.prepareStatement(query);
            ps.setDate(1, startDate);
            ps.setDate(2, endDate);

            ResultSet rs = ps.executeQuery();
            if (rs.next()) {
                result = rs.getInt("cnt");
            }

            ps.close();
            con.close();

        } catch (SQLException exc) {
            exc.printStackTrace();
        }
        return result;

    }

    public String profit_manufacturer(String startDate, String endDate) {
        String output = "*******Profit by Car Manufacturer*******\n\n";
        output += "Manufacturer\tModel No\tYear\tProfit\n";

        new ConnectionManager().createClass();
        java.util.Date javaDate = new java.util.Date();
        java.sql.Date mySQLDate = new java.sql.Date(javaDate.getTime());

        try {
            Connection con = new ConnectionManager().connect();
            String query = "select Manufacturer, C.ModelNo as Modnum, Year, "
+
                    "sum(SellingPrice) -sum(BuyingPrice) as profit " +
                    "from Car C, PurchaseHistory H where H.ModelNo=C.ModelNo
and " +
                    "Date between ? and ? group by Manufacturer";
            PreparedStatement ps = con.prepareStatement(query);
            ps.setString(1, startDate);
            ps.setString(2, endDate);

            ResultSet rs = ps.executeQuery();

            if (rs.next()) {
                output += rs.getFloat("Manufacturer") + "\t";
                output += rs.getFloat("Modnum") + "\t";
                output += rs.getFloat("Year") + "\t";
                output += rs.getFloat("profit") + "\n";
            }
            ps.close();
            con.close();
            rs.close();
        } catch (SQLException exc) {
            exc.printStackTrace();
        }
        return output;
    }

    public int profit_date(Date startDate, Date endDate) {
        Integer result2 = -1;
        new ConnectionManager().createClass();
        java.util.Date javaDate = new java.util.Date();
        java.sql.Date mySQLDate = new java.sql.Date(javaDate.getTime());

        try {
            Connection con = new ConnectionManager().connect();
            String query = "select date, sum(SellingPrice) -sum(BuyingPrice)
as profit from Car C, PurchaseHistory H where H.ModelNo=C.ModelNo and Date
between ? and ? group by Date";
            PreparedStatement ps = con.prepareStatement(query);
            ps.setDate(1, startDate);
            ps.setDate(2, endDate);

            ResultSet rs = ps.executeQuery();

            if (rs.next()) {
                result2 = rs.getInt("profit");
            }

            ps.close();
            con.close();

        } catch (SQLException exc) {
            exc.printStackTrace();
        }
        return result2;
    }
}
```

Appointment Query Manager

```
package com.nikrajsuk.crm.qm;

import com.google.gson.Gson;
import com.nikrajsuk.crm.objs.CustomerObj;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Date;
import java.util.ArrayList;

public class AppointmentQm {

   public final float tax = (float) 3.15;
   public static void main(String[] args) {
       System.out.println("Hello World");
   }

   public Integer createAppointment(Integer SSN, String inTime, String
outTime, String licensePlate, String comments,
                                    ArrayList<String> pkgs, ArrayList<String>
tests, ArrayList<String> parts) {
       Integer appointmentID = 0;
       new ConnectionManager().createClass();
       try {

           Connection con = new ConnectionManager().connect();
           String query = "INSERT INTO Appointment (`InTime`, `OutTime`,
`LicensePlate`, `comments`, `ssn`) " +
                   "VALUES ( ?, ?, ?, ?, ?)";
           PreparedStatement ps = con.prepareStatement(query);

           ps.setString(1, inTime);
           ps.setString(2, outTime);
           ps.setString(3, licensePlate);
           ps.setString(4, comments);
           ps.setInt(5, SSN);

           ResultSet rs = null;
           ps.executeUpdate();

           query = "SELECT AppointmentID FROM Appointment WHERE InTime = ?
AND OutTime = ? " +
                   "AND LicensePlate = ? AND ssn = ?";
           ps = con.prepareStatement(query);
           ps.setString(1, inTime);
           ps.setString(2, outTime);
           ps.setString(3, licensePlate);
           ps.setInt(4, SSN);

           rs = ps.executeQuery();
           while (rs.next()){
               appointmentID = rs.getInt("AppointmentID");
           }
           for(String pkgID: pkgs){
               if (pkgID == null) {
                   continue;
               }
               query = "INSERT INTO ServiceHistory (appointmentID, type,
typeID, price) values (?, ?, ?, " +
                       "(select sum(LaborCost) from Test as t where t.ID IN
(select testID from PackageTest as p " +
                       "where p.PackageID = ?)) );";
               ps = con.prepareStatement(query);
               ps.setInt(1, appointmentID);
               // 1 for package, 2 for test, 3 for spare part
               ps.setInt(2, 1);
               ps.setInt(3, Integer.valueOf(pkgID));
               ps.setInt(4, Integer.valueOf(pkgID));
```

```
                ps.executeUpdate();
            }

            for(String testID: tests){
                if (testID == null) {
                    continue;
                }
                query = "INSERT INTO ServiceHistory (appointmentID, type, 
typeID, price) values (?, ?, ?, " +
                        "(select LaborCost from Test as t where t.ID = ?)) ;";
                ps = con.prepareStatement(query);
                ps.setInt(1, appointmentID);
                // 1 for package, 2 for test, 3 for spare part
                ps.setInt(2, 2);
                ps.setInt(3, Integer.valueOf(testID));
                ps.setInt(4, Integer.valueOf(testID));
                ps.executeUpdate();
            }

            for(String partID: parts){
                if (partID == null) {
                    continue;
                }
                query = "INSERT INTO ServiceHistory (appointmentID, type, 
typeID, price) values (?, ?, ?, " +
                        "(select Selling_Price+Labor_Cost from SparePart as s 
where s.SparePartID = ?)) ;";
                ps = con.prepareStatement(query);
                ps.setInt(1, appointmentID);
                // 1 for package, 2 for test, 3 for spare part
                ps.setInt(2, 3);
                ps.setInt(3, Integer.valueOf(partID));
                ps.setInt(4, Integer.valueOf(partID));
                ps.executeUpdate();
            }



            ps.close();
            con.close();
            rs.close();

        } catch (SQLException exc) {
            exc.printStackTrace();
        }
        return appointmentID;

    }

    public String generateBill(Integer appID, Integer ssn, String lic, String 
intime, String outime, ArrayList<String> pkgs, ArrayList<String> tests,
                              ArrayList<String> parts){
        String bill = "";
        ArrayList<String> userDetails = new CustomerQm().fetchBySSN(ssn);
        bill += "**********Service Bill************\n";
        bill += userDetails.get(2) + ", " + userDetails.get(1) + "\n";
        bill += "License Plate: " + lic + "\n";
        bill += "Appointment ID: " + appID + "\n";
        bill += "In time: " + intime + "\n";
        bill += "Out time: " + outime + "\n\n";

        bill += "Packages opted:\n";
        String query = "";
        PreparedStatement ps = null;
        ResultSet rs = null;

        new ConnectionManager().createClass();
        try {
            Connection con = new ConnectionManager().connect();
            for(String pkgid: pkgs){
                if(pkgid==null){
                    continue;
                }
                query = "Select Name, Description from Packages where ID = ?";
                ps = con.prepareStatement(query);
                ps.setInt(1, Integer.parseInt(pkgid));
                rs = ps.executeQuery();
                while (rs.next()){
                    bill += "Package Name: " + rs.getString("Name") + "\n";
                    bill += "Description: " + rs.getString("Description")+
"\n";
                }
                bill += "Tests Included: \n";
                query = "Select Name, LaborCost, Description from Test where
ID IN (\n" +
                        "SELECT TestID FROM PackageTest where PackageID = ?
)";
                ps = con.prepareStatement(query);
                ps.setInt(1, Integer.parseInt(pkgid));
                rs = ps.executeQuery();
                while (rs.next()){
                    bill += "\tTest Name: " + rs.getString("Name") + "\n";
                    bill += "\tDescription: " + rs.getString("Description")+
"\n";
                    bill += "\tLabor Cost: $" + rs.getString("LaborCost")+
"\n\n";

                }
            }
            for(String test: tests){
                if(test==null){
                    continue;
                }
                bill += "Tests Opted: \n";

                query = "Select Name, Description, LaborCost from Test where
ID = ?";
                ps = con.prepareStatement(query);
```

```
               ps.setInt(1, Integer.parseInt(test));
                rs = ps.executeQuery();
               while (rs.next()){
                   bill += "Test Name: " + rs.getString("Name") + "\n";
                   bill += "Description: " + rs.getString("Description")+
"\n";
                   bill += "Labor Cost: $" + rs.getString("LaborCost")+
"\n\n";
               }

           }

           for(String part: parts){
               if(part==null){
                   continue;
               }
               bill += "Parts Replaced: \n\n";

                query = "Select Name, Description, Labor_Cost, Selling_Price
from SparePart where SparePartID = ?";
                ps = con.prepareStatement(query);
               ps.setInt(1, Integer.parseInt(part));
                rs = ps.executeQuery();
               while (rs.next()){
                   bill += "Part Name: " + rs.getString("Name") + "\n";
                   bill += "Description: " + rs.getString("Description")+
"\n";
                   bill += "Labor Cost: $" + rs.getString("Labor_Cost")+
"\n";
                   bill += "Price: $" + rs.getString("Selling_Price")+ "\n";
               }

           }

           query = "select sum(price) as p from ServiceHistory where
appointmentID = ?";
           ps = con.prepareStatement(query);
           ps.setInt(1, appID);
           rs = ps.executeQuery();
           while (rs.next()){
               Float totalBill = rs.getFloat("p");
               bill += "\nTotal Bill: \t\t" + totalBill + "\n";
               bill += "Tax applied @(" + tax +"):\t\t $" + (totalBill * 3.5
/ 100)+ "\n";
               bill += "Final Amount:\t\t $" + (totalBill + (totalBill * 3.5
/ 100))+ "\n";
           }


           ps.close();
           con.close();
           rs.close();
       } catch (SQLException exc) {
           exc.printStackTrace();
```

```
        }

      return bill;
   }

   private float getPackagePrice() {
       return 0;
   }

   public Integer createHistory(Integer appointmentID, String type, Integer
ID, Float price) {
       new ConnectionManager().createClass();
       try {

           Connection con = new ConnectionManager().connect();
           String query = "INSERT INTO ServiceHistory (`ID`, `type`,
`typeID`, `price`) " +
                   "VALUES ( ?, ?, ?, ?)";
           PreparedStatement ps = con.prepareStatement(query);
           ps.setInt(1, appointmentID);
           ps.setString(2, type);
           ps.setInt(3, ID);
           ps.setFloat(4, price);

           ResultSet rs = ps.executeQuery();            while (rs.next()){
               appointmentID = 1;
           }

           ps.close();
           con.close();

       } catch (SQLException exc) {
           exc.printStackTrace();
       }
       return appointmentID;

   }

}
```

Car Purchase Query Manager-

```
//
// Source code recreated from a .class file by IntelliJ IDEA
// (powered by FernFlower decompiler)
//

package com.nikrajsuk.crm.qm;

import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class CarPurchase {
   public CarPurchase() {
   }

   public void main(String[] args) {
       this.fetchResults();
   }

   public String fetchResults() {
       String res = "";
       (new ConnectionManager()).createClass();

       try {
           Connection con = (new ConnectionManager()).connect();
           Statement stat = con.createStatement();
           String query = "SELECT * from user; ";
           ResultSet rs = stat.executeQuery(query);
           rs.next();
           res = String.valueOf(rs.getInt("id"));
           stat.close();
           con.close();
           System.out.println(res);
       } catch (SQLException var6) {
           var6.printStackTrace();
       }

       return res;
   }
}
```

Customer Query Manager

```
package com.nikrajsuk.crm.qm;

import java.sql.*;
import java.util.ArrayList;
import com.nikrajsuk.crm.objs.CustomerObj;

public class CustomerQm {

   public static void main(String[] args) {
       System.out.println("Hello World");
       ArrayList<String> res = new CustomerQm().fetchBySSN(1234567880);
       System.out.println(res);
```

```
    }

    public ArrayList<String> fetchBySSN(Integer SSN) {
        ArrayList<String> details = new ArrayList<String>();
        new ConnectionManager().createClass();
        CustomerObj cus = new CustomerObj();
        try {
            Connection con = new ConnectionManager().connect();
            String query = "SELECT * FROM Customer where SSN = ?";
            PreparedStatement ps = con.prepareStatement(query);
            ps.setInt(1, SSN);

            ResultSet rs = ps.executeQuery();

            while (rs.next()){
                details.add(String.valueOf(rs.getInt("SSN")));
                details.add(String.valueOf(rs.getString("FirstName")));
                details.add(String.valueOf(rs.getString("LastName")));
                details.add(String.valueOf(rs.getInt("MobileNumber")));
                details.add(String.valueOf(rs.getInt("EmailID")));
            }

            ps.close();
            con.close();
            return details;

        } catch (SQLException exc) {
            exc.printStackTrace();
        }

        return details;
    }


    public int insertNewSSN(Integer SSN, String firstName, String lastName,
String emailID, Integer mobileNum){
        Integer result = 0;

        new ConnectionManager().createClass();
        try {

            Connection con = new ConnectionManager().connect();
            String query = "INSERT INTO Customer (`SSN`, `FirstName`,
`LastName`, `EmailID`, `MobileNumber`) " +
                    "VALUES ( ?, ?, ?, ?, ?)";
            PreparedStatement ps = con.prepareStatement(query);
            ps.setInt(1, SSN);
            ps.setString(2, firstName);
            ps.setString(3, lastName);
            ps.setString(4, emailID);
            ps.setString(5, String.valueOf(mobileNum));

            result  = ps.executeUpdate();
            ps.close();
            con.close();

        } catch (SQLException exc) {
            exc.printStackTrace();
        }
        return result;

    }

}
```

Service Query Manager

```
package com.nikrajsuk.crm.qm;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Random;

public class ServiceQm {

    public static void main(String[] args) {
        System.out.println("Hello World");

    }

    public double getPackagePrice(String pName) {
        double result = 0;
        new ConnectionManager().createClass();
        java.util.Date javaDate = new java.util.Date();
        java.sql.Date mySQLDate = new java.sql.Date(javaDate.getTime());

        try {
            Connection con = new ConnectionManager().connect();
            String query = "Select SUM(LaborCost) as cost FROM Test Where ID
IN ALL" +
                    "(Select TestID FROM PackageTest where PackageID IN (" +
                    "Select ID from Packages where Name = ?))";
            PreparedStatement ps = con.prepareStatement(query);
            ps.setString(1, pName);

            ResultSet rs = ps.executeQuery();
            while (rs.next()) {
                result = Double.valueOf(rs.getFloat("cost"));
            }
            ps.close();
            con.close();
        } catch (SQLException exc) {
            exc.printStackTrace();
        }
        return result;

    }

    public double getTestPrice(String tName) {
        double result = 0.0;
        new ConnectionManager().createClass();
        java.util.Date javaDate = new java.util.Date();
        java.sql.Date mySQLDate = new java.sql.Date(javaDate.getTime());

        try {
            Connection con = new ConnectionManager().connect();
            String query = "Select LaborCost FROM Test where Name  = ?";
            PreparedStatement ps = con.prepareStatement(query);
            ps.setString(1, tName);

            ResultSet rs = ps.executeQuery();
            while (rs.next()) {
                result = Double.valueOf(rs.getFloat("LaborCost"));
            }
            ps.close();
            con.close();

        } catch (SQLException exc) {
            exc.printStackTrace();
        }
        return result;

    }

    public double getSparePartWhat(String sparePart, String what) {
        double result = 0.0;
        new ConnectionManager().createClass();
        java.util.Date javaDate = new java.util.Date();
        java.sql.Date mySQLDate = new java.sql.Date(javaDate.getTime());

        try {
            Connection con = new ConnectionManager().connect();
            String query = "Select ? FROM Test where Name  = ?";
            PreparedStatement ps = con.prepareStatement(query);
            ps.setString(1, what);
            ps.setString(2, sparePart);

            ResultSet rs = ps.executeQuery();
            while (rs.next()) {
                result = rs.getFloat(what);
            }
            ps.close();
            con.close();

        } catch (SQLException exc) {
            exc.printStackTrace();
        }
        return result;

    }
}
```

Marketing Query Manager

```
//
// Source code recreated from a .class file by IntelliJ IDEA
// (powered by FernFlower decompiler)
//

package com.nikrajsuk.crm.qm;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class MarketingQm {
   public MarketingQm() {
   }

   public static void main(String[] args) {
        System.out.println("Hello World");
   }

   public String fetchCampaignDetails() {
        String result = "Campaign Details<br><br>";
        result = result + "<table>";
        result = result + "<tr><th>Marketing Type</th><th>Coupon
Code</th><th>Promotion Date</th><th>Expiry Date</th><th>Content</th></tr>";
        (new ConnectionManager()).createClass();

        try {
            Connection con = (new ConnectionManager()).connect();
            String query = "Select * from Marketing";
            PreparedStatement ps = con.prepareStatement(query);

            ResultSet rs;
            for(rs = ps.executeQuery(); rs.next(); result = result +
"</tr>") {
                result = result + "<tr>";
                result = result + "<th>" + rs.getString("MarketingType") +
"</th>";
                result = result + "<th>" + rs.getString("CouponCode") +
"</th>";
                result = result + "<th>" + rs.getString("Promotion_date") +
"</th>";
```

```
                result = result + "<th>" + rs.getString("Expiry_date") +
"</th>";
                result = result + "<th>" + rs.getString("Content") +
"</th>";
            }

            result = result + "</table>";
            result = result + "<table>";
            result = result + "<br>Customer Details<br><br>";
            result = result + "<tr><th>First Name</th><th>Last
Name</th><th>Mobile Number</th><th>Email</th></tr>";
            query = "Select * from Customer";
            ps = con.prepareStatement(query);

            for(rs = ps.executeQuery(); rs.next(); result = result +
"</tr>") {
                result = result + "<tr>";
                result = result + "<th>" + rs.getString("FirstName") +
"</th>";
                result = result + "<th>" + rs.getString("LastName") +
"</th>";
                result = result + "<th>" + rs.getString("MobileNumber") +
"</th>";
                result = result + "<th>" + rs.getString("EmailID") +
"</th>";
            }

            result = result + "<table>";
            ps.close();
            con.close();
            rs.close();
        } catch (SQLException var6) {
            var6.printStackTrace();
        }

        return result;
    }
}
```

Purchase Query Manager

```
package com.nikrajsuk.crm.qm;

import com.google.gson.Gson;
import com.nikrajsuk.crm.objs.CustomerObj;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Random;
```

```
public class PurchasesQm {

   public static void main(String[] args) {
       System.out.println("Hello World");

   }

   public int createPurchase(Integer SSN, String modelNo) {
       Integer result = 0;
       new ConnectionManager().createClass();
       java.util.Date javaDate = new java.util.Date();
       java.sql.Date mySQLDate = new java.sql.Date(javaDate.getTime());

       try {
           Connection con = new ConnectionManager().connect();
           String query = "INSERT INTO PurchaseHistory (`SSN`, `ModelNo`,
`LicensePlate`, `Date`) " +
                   "VALUES ( ?, ?, ?, ?)";
           PreparedStatement ps = con.prepareStatement(query);
           ps.setInt(1, SSN);
           ps.setString(2, modelNo);
           ps.setString(3, generateLicensePlate().trim());
           ps.setDate(4, mySQLDate);

           result = ps.executeUpdate();

           ps.close();
           con.close();

       } catch (SQLException exc) {
           exc.printStackTrace();
       }
       return result;

   }

   public String generateLicensePlate(){
       Random randNum = new Random();
       String rdm = "ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890";
       String licensePlate = "";
       for(int i=0; i< 10; i++){
           licensePlate +=
String.valueOf(rdm.toCharArray()[randNum.nextInt(rdm.toCharArray().length)]);
       }
       return licensePlate;
   }

}
```

Car Query Manager

```
package com.nikrajsuk.crm.qm;

import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class CarQm {

   public void main(String[] args) {
       // do nothing

   }

   public ArrayList<String> fetchCarModelsList(String manufacturer) {
       ArrayList<String> res = new ArrayList<String>();
       new ConnectionManager().createClass();

       try {
           Connection con = new ConnectionManager().connect();
           Statement stat = con.createStatement();
           String query = "SELECT ModelNo FROM Car WHERE `Manufacturer` = ?";
           PreparedStatement ps = con.prepareStatement(query);
           ps.setString(1, manufacturer);
           ResultSet rs = ps.executeQuery();
           while (rs.next()) {
               res.add(rs.getString("ModelNo"));
           }

           stat.close();
           con.close();
       } catch (SQLException throwables) {
           throwables.printStackTrace();
       }

       return res;
   }

   public String fetchWhat(String manufacturer, String modelNo, String what)
{
       String res = "";
       new ConnectionManager().createClass();

       try {
           Connection con = new ConnectionManager().connect();
           Statement stat = con.createStatement();
           String query = "SELECT ? FROM Car WHERE `Manufacturer` = ? AND
`ModelNo` = ?";
           PreparedStatement ps = con.prepareStatement(query);

           ps.setString(1, what);
           ps.setString(2, manufacturer);
           ps.setString(3, modelNo);

           ResultSet rs = ps.executeQuery();
```

```
            while (rs.next()) {
                res = rs.getString(what);
            }

            stat.close();
            con.close();
        } catch (SQLException throwables) {
            throwables.printStackTrace();
        }

        return res;
    }

    public ArrayList<String> fetchManufacturerList() {
        ArrayList<String> res = new ArrayList<String>();

        new ConnectionManager().createClass();

        try {
            Connection con = new ConnectionManager().connect();
            Statement stat = con.createStatement();
            String query = "SELECT DISTINCT Manufacturer FROM Car ";
            ResultSet rs = stat.executeQuery(query);
            while (rs.next()) {
                res.add(rs.getString("Manufacturer"));
            }

            stat.close();
            con.close();
        } catch (SQLException throwables) {
            throwables.printStackTrace();
        }

        return res;
    }

}
```

Connection Query Manager

```
package com.nikrajsuk.crm.qm;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;


public class ConnectionManager {
   String URL = "jdbc:mysql://freedb.tech:3306/freedbtech_dbmscarproject";
   String userName = "freedbtech_nikrajsuk";
   String password = "g0r!llaDance";

   public Connection connect() throws SQLException {
       return DriverManager.getConnection(URL, userName, password);
   }

   public void createClass() {
       try {
           Class.forName("com.mysql.cj.jdbc.Driver");
       } catch (ClassNotFoundException e) {
           e.printStackTrace();
       }
   }

}
```

HTML Files

index.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
       "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
   <title>Home</title>
</head>
<body>
<h1 align="center">A1 Car Service</h1>


<button type="button"
onclick="location.href='http://localhost:8082/demo_war_exploded/buy_car.html'
;"
       value="Buy car">Buy Car
</button>
<br>
<button align="center" type="button" Schedule appointment

onclick="location.href='http://localhost:8082/demo_war_exploded/appointment.h
tml';"> Schedule appointment
</button>
<br>
<button align="center" type="button"
onclick="location.href='http://localhost:8082/demo_war_exploded/analytics.htm
l';"
       value="Analytics">Analytics
</button>
<br>

<form action="marketing-servlet" method="post">
   <input type="submit" value="Marketing Analytics">
```

```
</form>

</body>
</html>
```

buy_car.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
       "http://www.w3.org/TR/html4/loose.dtd">
<html lang="en">
<head>
   <title>Buy Car</title>
</head>

<body>

<form action="car-purchase" method="post">
   SSN: <input type="text" id="ssn" name="ssn"><br>
   First Name: <input type="text" id="fname" name="fname"><br>
   Last Name: <input type="text" id="lname" name="lname"><br>
   Mobile: <input type="text" id="mobile" name="mobile"><br>
   Email: <input type="text" id="email" name="email"><br>
   Coupon: <input type="text" id="coupon" name="coupon"><br>

   Car manufacturer:
   <select name="Car manufacturer" id="car-manufacturer">
       <option value="Toyota">Toyota</option>
       <option value="Lexus">Lexus</option>
   </select><br><br>

   Car model:
   <select name="Car model" id="car-model">
       <option value="Toyota">Camry</option>
       <option value="Lexus">City</option>
   </select><br><br>

   <label for="car-year">Car year:</label>
   <select name="Car year" id="car-year">
       <option value="2020">2020</option>
       <option value="2021">2021</option>
   </select>
   <input type="submit" value="Submit"> <br/>
</form>

</body>
</html>
```

analytics.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
       "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
   <title>Home</title>
</head>
<body>
<h1>Please Select One</h1>

<button type="button"
onclick="location.href='http://localhost:8082/demo_war_exploded/cars_sold.htm
l';"
       value="Cars Sold">Cars Sold</button>
<button type="button"
onclick="location.href='http://localhost:8082/demo_war_exploded/profit_analyt
ics.html';"
value="Profit Analytics">Profit Analytics</button>

</body>
</html>
```

Appointment.html-

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
       "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
   <title>Appointment</title>
</head>
<body>
<form action="appointment-servlet" method="post">
   SSN: <input type="text" id="ssn" name="SSN" ><br>
   First name: <input type="text" id="fname" name="fname" ><br>
   Last name: <input type="text" id="lname" name="lname"><br>
   Mobile: <input type="text" id="mobile" name="mobile"><br>
   Email: <input type="text" id="Email" name="Email"><br>
   Plate Number: <input type="text" id="plate no" name="Plate Number"><br>
   In Time: <input type="date" id="in_time_date" name="in_time_date"> <input
type="time" id="in_time_time" name="in_time_time"><br>
   Out Time: <input type="date" id="out_time_date" name="out_time_date">
<input type="time" id="out_time_time" name="out_time_time"><br>
   Coupon: <input type="text" id="coupon" name="Coupon"><br><br>

   Choose a package:<br>
   <input type="checkbox" id="pkg1" name="pkg1" value=1> 1 Year package <br>
   <input type="checkbox" id="pkg2" name="pkg2" value=2> 2 Year package <br>
   <input type="checkbox" id="pkg3" name="pkg3" value=3> 3 Year package <br>
   <br><br>
   Choose Tests:<br>
   <input type="checkbox" id="test1" name="test1" value=1> Engine Oil Test
<br>
```

```
   <input type="checkbox" id="test2" name="test2" value=2> Wheel Alignment
Test <br>
   <input type="checkbox" id="test3" name="test3" value=3> Brake Test <br>
   <input type="checkbox" id="test4" name="test4" value=4> Emission Test <br>
   <br><br>
   Choose Spare Parts to be replaced:<br>
   <input type="checkbox" id="part2" name="part2" value=1> Engine Oil <br>
   <input type="checkbox" id="part1" name="part1" value=2> Tires <br>
   <input type="checkbox" id="part3" name="part3" value=3> WindShield <br>
   <input type="checkbox" id="part4" name="part4" value=4> Fragrance
   <br><br>
   <input type="submit" value="Submit">
</form>
</body>
</html>
```

cars_sold.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title> Cars Sold</title>
</head>
<body>
<form name = "carSoldAnalyticsServlet" action="cars-sold-analytics"
method="post" >
   <h4>Cars Sold Analytics</h4>
   <h4> Select Date Range:</h4>
   Start day: <input type="date" id="startdate" name="Start Date"><br><br>
   End Date: <input type="date" id="enddate" name="End Date"><br><br>

   <input type="submit">
</form>
</body>
</html>
```

SSN_check.html-

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
       "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
   <title>Buy Car</title>
   <script>
       function testVariable() {
           var SSN = document.getElementById("ssn").value;
           document.getElementById('spanResult').textContent = SSN + "1";
       }
   </script>
</head>

<body>
SSN: <input type="text" id="ssn" name="ssn"><br>
<button  onclick="testVariable()">Submit</button> <br/>

</body>
</html>
```

profit_analytics.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>B</title>
</head>
<body>
<form action="profit-made" method="post">
   <label for="startdate">Start day:</label>
   <input type="date" id="startdate" name="Start Date"><br><br>
   <label for="enddate">End Date:</label>
   <input type="date" id="enddate" name="End Date"><br><br>
   <input type="submit">

</form>
</body>
</html>
```

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd"
        version="4.0">

</web-app>
```

pom.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
   <modelVersion>4.0.0</modelVersion>

   <groupId>com.example</groupId>
   <artifactId>demo</artifactId>
   <version>1.0-SNAPSHOT</version>
   <name>demo</name>
   <packaging>war</packaging>

   <properties>
       <maven.compiler.target>1.8</maven.compiler.target>
       <maven.compiler.source>1.8</maven.compiler.source>
       <junit.version>5.7.0</junit.version>
   </properties>

   <dependencies>
       <dependency>
           <groupId>javax.servlet</groupId>
           <artifactId>javax.servlet-api</artifactId>
           <version>4.0.1</version>
           <scope>provided</scope>
       </dependency>
       <dependency>
           <groupId>org.junit.jupiter</groupId>
           <artifactId>junit-jupiter-api</artifactId>
           <version>${junit.version}</version>
           <scope>test</scope>
       </dependency>
       <dependency>
           <groupId>org.junit.jupiter</groupId>
           <artifactId>junit-jupiter-engine</artifactId>
           <version>${junit.version}</version>
           <scope>test</scope>
       </dependency>
       <!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java
-->
       <dependency>
           <groupId>mysql</groupId>
           <artifactId>mysql-connector-java</artifactId>
           <version>8.0.23</version>
       </dependency>
       <!-- https://mvnrepository.com/artifact/com.google.code.gson/gson
-->
       <dependency>
           <groupId>com.google.code.gson</groupId>
           <artifactId>gson</artifactId>
           <version>2.8.6</version>
       </dependency>
```

```
      <dependency>
          <groupId>org.projectlombok</groupId>
          <artifactId>lombok</artifactId>
          <version>1.16.20</version>
      </dependency>
  </dependencies>

  <build>
      <plugins>
          <plugin>
              <groupId>org.apache.maven.plugins</groupId>
              <artifactId>maven-war-plugin</artifactId>
              <version>3.3.0</version>
          </plugin>
      </plugins>
  </build>
</project>
```