C:\Users\Messom\Desktop\Monash.jpg

FIT5192 Assignment One - Design Report

Spring Car Sales System

Hanlin Huang:2\*\*\*\*\*\*4

**Contents Page**

1. Overview…………………………………………………………………………… 3
2. Functional diagram…………………………………………………………………. 3
3. Core program functionality……………………………………………………….... 4
4. Usability Design Review…………………………………………………………… 7
5. Checklist of site functionality………………………………………………………. 8
6. User stories…………………………………………………………………………. 9
7. Data dictionary………………………………………………………………...…… 20
8. **Overview**

The spring car sales system will provide the user with two Java EE client: application client and web client.

Through the application client the user can search for car through the combination of features like Manufacturer, Model Name, Model Number or Type. And view the search result in a tabular format which enable the choosing of one record and show the detailed information about this car.

The web client will provide user with more services. It will allow the registration of users and according to the different type of user, it will provide user with different services.

For a customer who have registered into this system, he or she can choose one salesperson to serve him or her and then he or she can search the available car through the combination of Manufacturer, Model Name, Model Number and type and choose to view a detailed information of a selected car. In the car detailed information page, user can decide whether to buy the car. Once decide to buy the car, the information of this order will be shown to the user.

For a sales person who have access to this system, he or she has the ability to search all the car, edit the repository which is to delete, add and update the car information in the repository. Besides, he or she can search the customer based on a combination of user id, user lastname, firstname, type andemail information. At last the sales information can be retrieved by the salesperson thorough this system.

1. **Functional diagram**

1. **Core program functionality**

* **Entity build**: the entity is the building block of the database’s table. And is the object manipulated throughout the program. Here display car entity.

@Entity

@NamedQueries({@NamedQuery(name = "getAllCar", query = "SELECT c from Car c")})

public class Car implements Serializable{

@Id

private String VIN;

private String modelNo;

private String modelName;

private String manufacturer;

private String type;

private String thumbnail;

private String description;

private String carStatus;

private Date producedDate;

public Car(){

}

Constructors and getter setter methods ….

* **Multi-conditions query**: there are many places apply multi-conditions query to retrieve the result from the database, for instance searching car based on different key words, searching user based on different elements and searching sales records. The following function demonstrates the customer’s query of car.

public List<Car> customerCarSearch(String manufacturer, String modelName, String modelNo, String type) throws Exception {

StringBuilder queryStr = new StringBuilder();

queryStr.append("SELECT c FROM Car c WHERE 1 = 1 ");

if (modelName != null && !modelName.isEmpty()) {

queryStr.append("AND c.modelName = :modelName ");

}

if (modelNo != null && !modelNo.isEmpty()) {

queryStr.append("AND c.modelNo = :modelNo ");

}

if (manufacturer != null && !manufacturer.isEmpty()) {

queryStr.append("AND c.manufacturer = :manufacturer ");

}

if (type != null && !type.isEmpty()) {

queryStr.append("AND c.type = :type ");

}

queryStr.append("AND c.carStatus = :carStatus");

Query query = entityManager.createQuery(queryStr.toString());

if (modelName != null && !modelName.isEmpty()) {

query.setParameter("modelName", modelName);

}

if (modelNo != null && !modelNo.isEmpty()) {

query.setParameter("modelNo", modelNo);

}

if (manufacturer != null && !manufacturer.isEmpty()) {

query.setParameter("manufacturer", manufacturer);

}

if (type != null && !type.isEmpty()) {

query.setParameter("type", type);

}

query.setParameter("carStatus", "Accessible");

return query.getResultList();

}

* **Add car**: the function code below demonstrate the process of adding a new car to the repository once the function is triggered. The similar process can also be applied to the registration when add a new user.This function is written in the managed bean of editing repository.

public void addCar() throws Exception{

manipulateCar.addCar(car.getVIN(),car.getModelName(),car.getModelNo(),car.getManufacturer(),car.getType(),

car.getCarStatus(),car.getDescription(),car.getThumbnail());

}

The following function’s code is implemented in EJB which create a car object and persist this object to the database through entity manager.

public void addCar(String VIN, String modelName, String modelNumber, String manufacturer, String type,

String status, String description, String picAddress) throws Exception {

Car car = new Car(VIN, modelNumber, modelName, manufacturer, type, picAddress, description, status);

entityManager.persist(car);

}

* **Add Records**: when customer buy a car, there will be a sales record insert into the Sales table. According to the action customer choose, he or she may pay directly or just put the car into shopping cart, the sale status can be different. The following codes display insertion of a record and it’s implemented in EJB.

public void addRecord(Integer customerId, String carVIN, Integer salesPersonId, String date, String statues) {

java.sql.Date tradeDate = java.sql.Date.valueOf(date);

Users customer = entityManager.find(Users.class, customerId);

Users salesPerson = entityManager.find(Users.class, salesPersonId);

Car carBought = entityManager.find(Car.class, carVIN);

if (statues.equals("Paid")) {

carBought.setCarStatus("Sold");

} else if (statues.equals("Unpaid")) {

carBought.setCarStatus("In Process");

}

Sale saleRecord = new Sale(customer, tradeDate, salesPerson, carBought, statues);

entityManager.persist(saleRecord);

}

* **JAAS security mechanism:** the JAAS security mechanism is applied to the system. The following code display the login function used to identify the current login user’s validation and direct them to different page it is implemented in the managed bean responsible for login..

public void login() {

try {

//message = "";

HttpServletRequest request = (HttpServletRequest) FacesContext.getCurrentInstance().getExternalContext().getRequest();

request.login(this.email,this.password);

this.currentUser = searchUser.searchUser(email, PasswordEncryption.Encrypt(password,"SHA-256"));

FacesContext facesContext = FacesContext.getCurrentInstance();

HttpServletResponse response = (HttpServletResponse) facesContext.getExternalContext().getResponse();

if (request.isUserInRole("S")) {

response.sendRedirect("salesPerson/salesPersonMainPage.xhtml");

facesContext.responseComplete();

} else if (request.isUserInRole("C")) {

response.sendRedirect("customer/customerMainPage.xhtml");

facesContext.responseComplete();

} else {

FacesContext.getCurrentInstance().getExternalContext().redirect("/index.xhtml");

}

} catch (Exception e) {

e.printStackTrace();

}

}

1. **Usability Design Review**

* **User registration:** user can register to the system by providing their information like firstname, lastname, email address, phone number, address, etc., and select the role of himself that will decide the service he or she can have.
* **Car search**: the car search function is available to both the customer and salesperson. But the searching results provided to the customer are only available cars while the salesperson can search out the car of different status: sold and available.
* **View car detail**: after the car is searched out based on some key words and the result is displayed in a table, user can choose to see the detail of specific car. And these information are retrieved from the database.
* **Buy car**: after the customer view the detail of car, he or she can buy the car right now and an order record of this sale will display to the user, and the status of the car has been changed from “available” to “sold” and can no longer be searched by the customer.
* **Repository edit**: the salesperson is endowed the ability to manipulate the car in the car repository. He or she will firstly view a list of all the cars in the repository. Then, he or she can edit the target car or delete a car record. Or, the salesperson can add a car record to the repository.
* **User search**: the sales person can search the customer based on the firstname, lastname, userId, email, etc. The searching results are displayed in a table.
* **Record search**: the record of sales can be searched based on different query requirement provided.

1. **User stories**

**When the user run the web application, the login page is like Figure 1.**

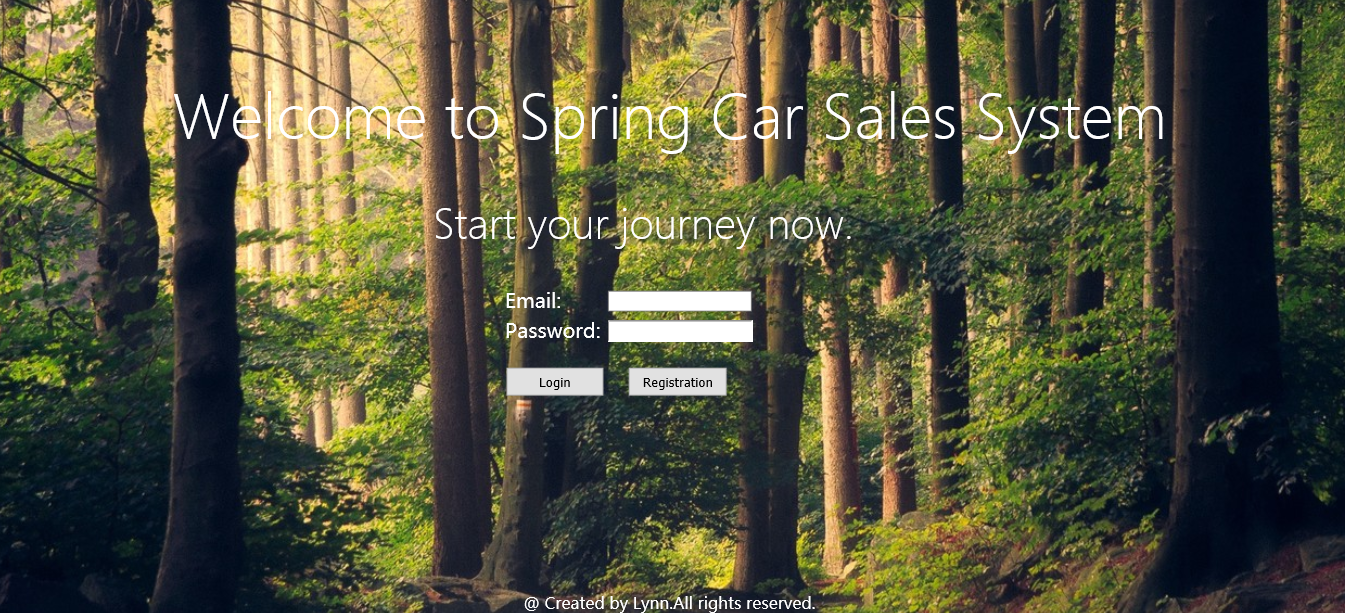


Figure 1

If the person is the new user to this system, he or she has to register first, click the register button and jump to the register page like Figure 2.

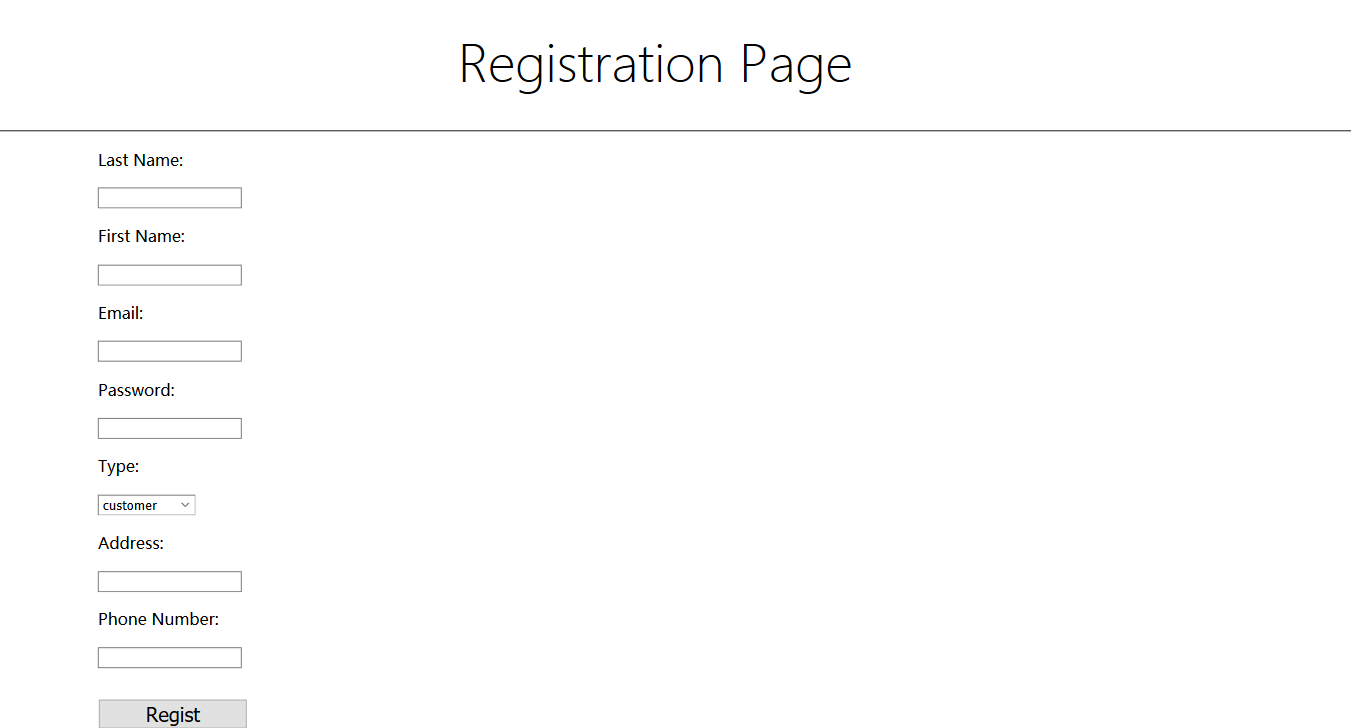


Figure 2

The new user need to complete the form, if there is something wrong with the format of input message, a warning will be shown like Figure 3.

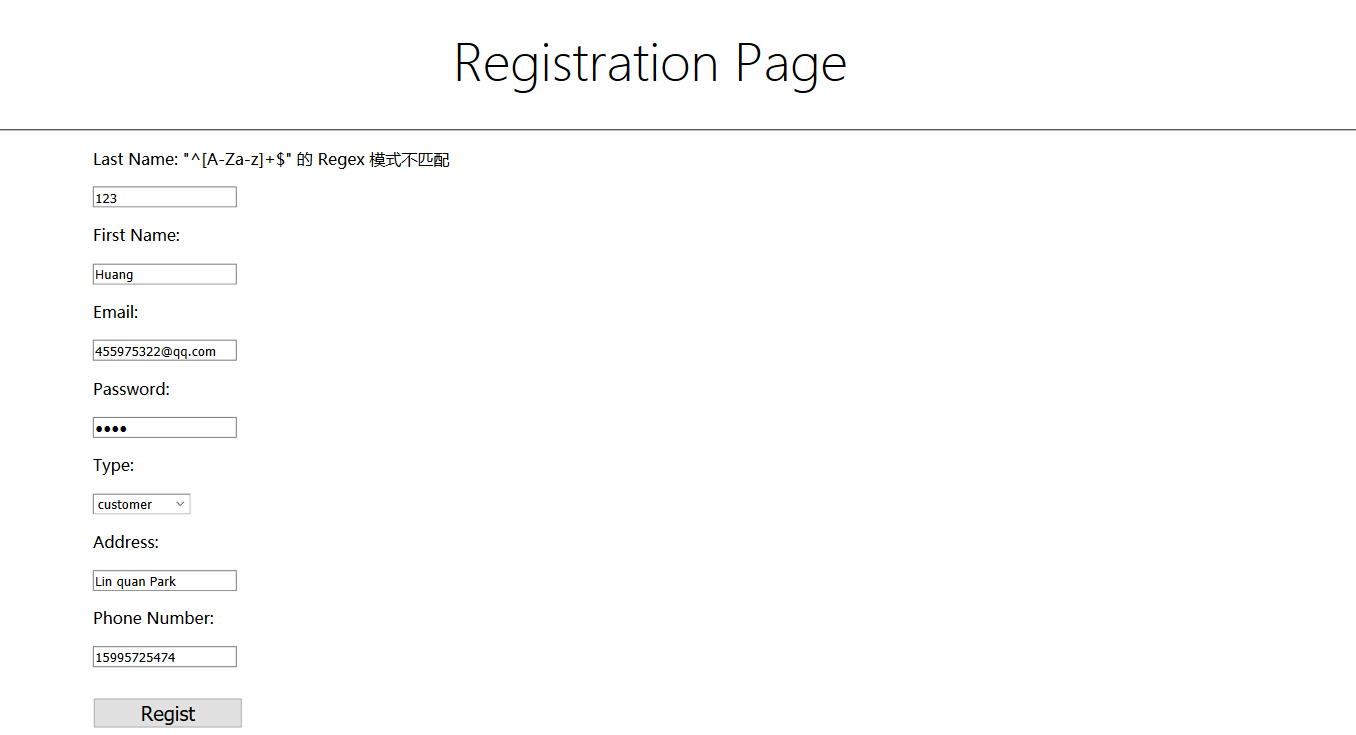


Figure 3

When the information is correctly filled and click the regist button, the page will jump to the login page. Use the account information to login, the customer main page is like Figure 4.The user name will be shown on the page. On the main page choose one server to provide service in the following period.

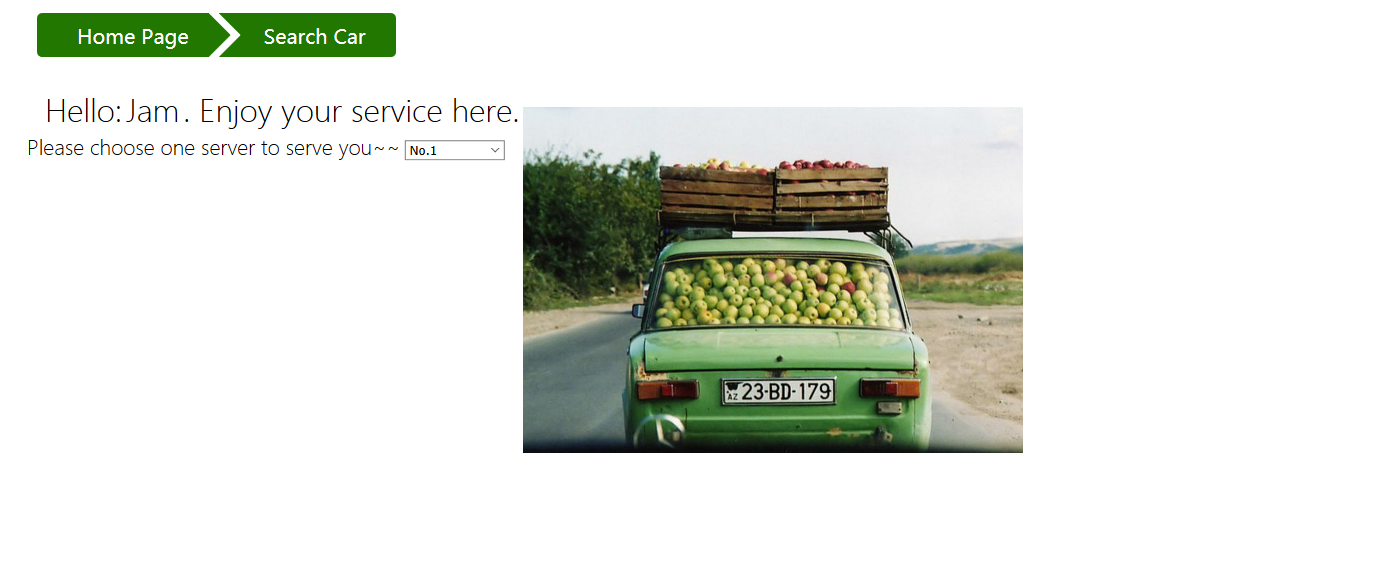


Figure 4

Click search car on the navigation bar, the page will jump to the car search page like Figure 5.

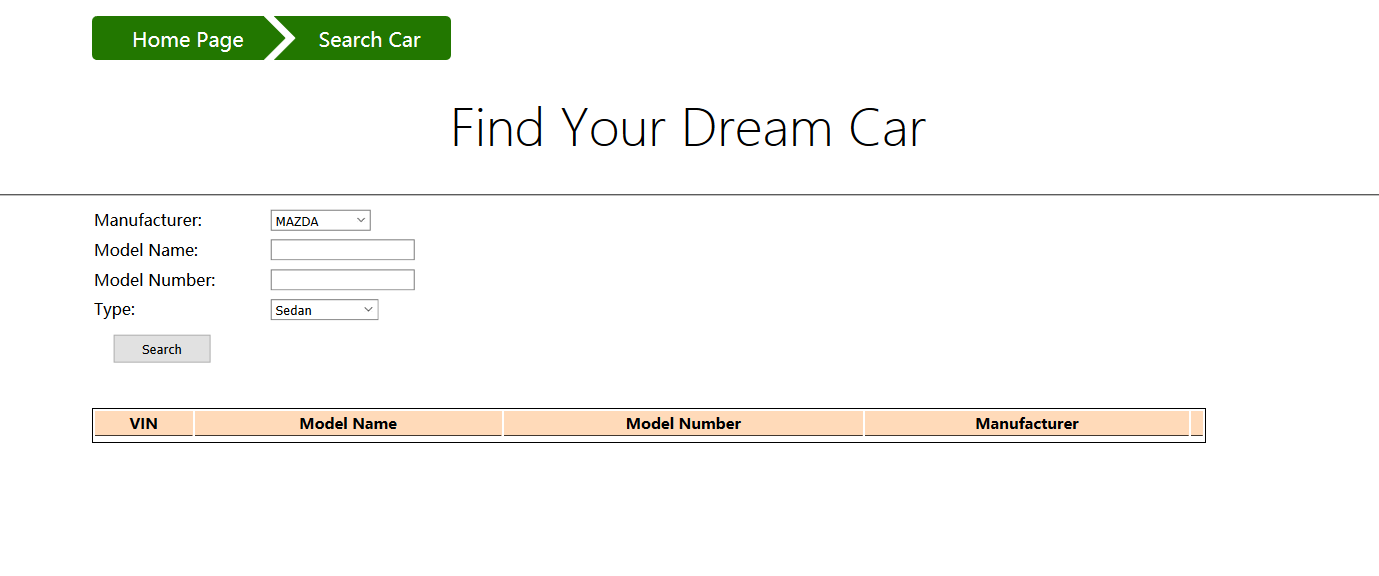


Figure 5

Choose the selection requirements and click search button, the Figure 6 will display.

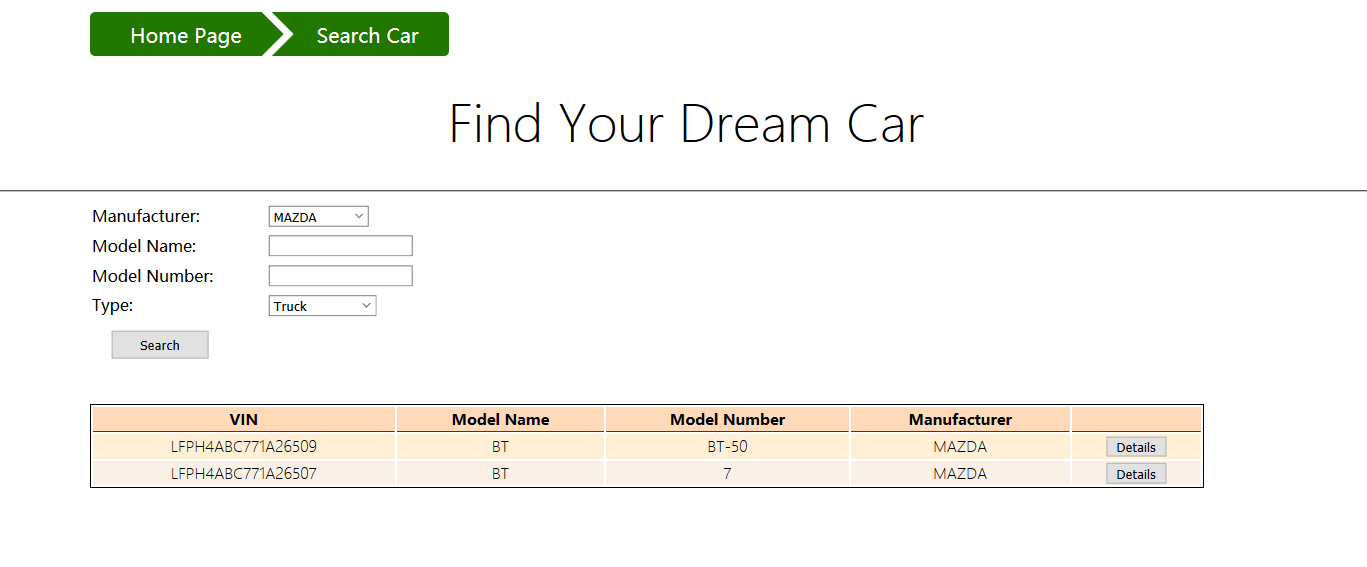


Figure 6

Click Detail button on one row and will display the detail information about the selected car, like Figure 7.

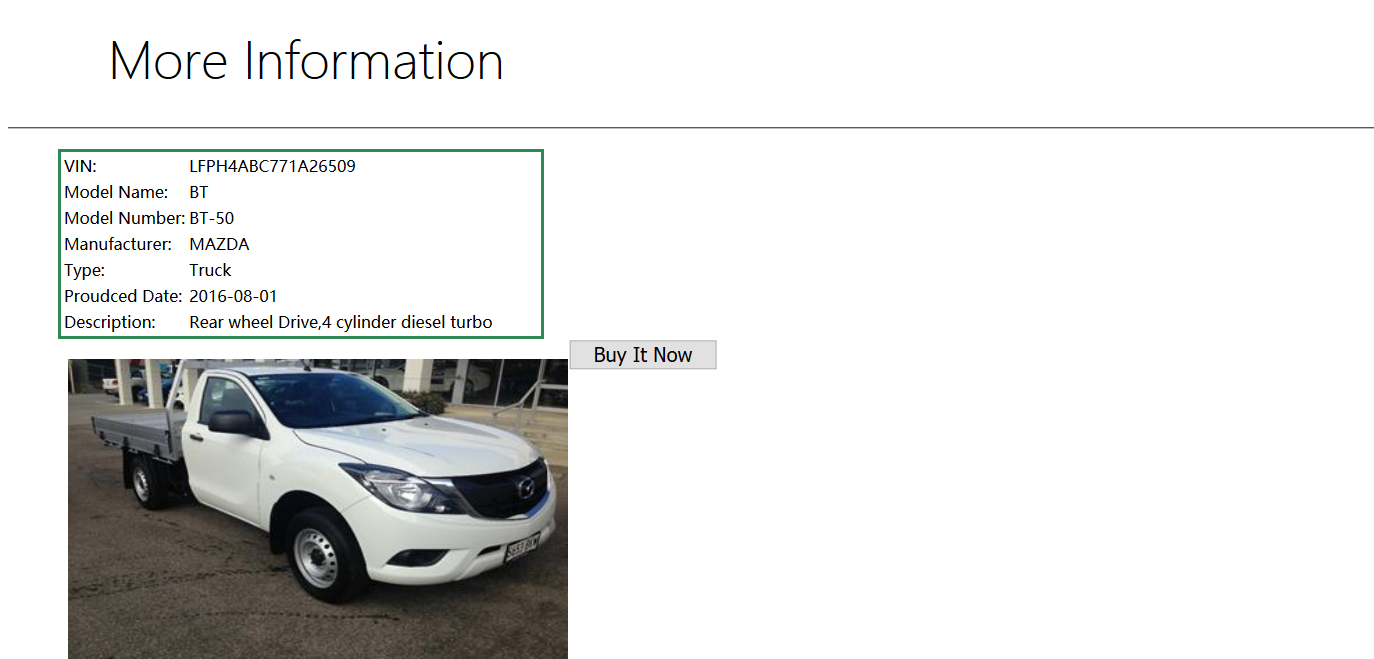


Figure 7

If the user want to buy this car, then after click the buy it now button, a page shows the order will appear like Figure 8.

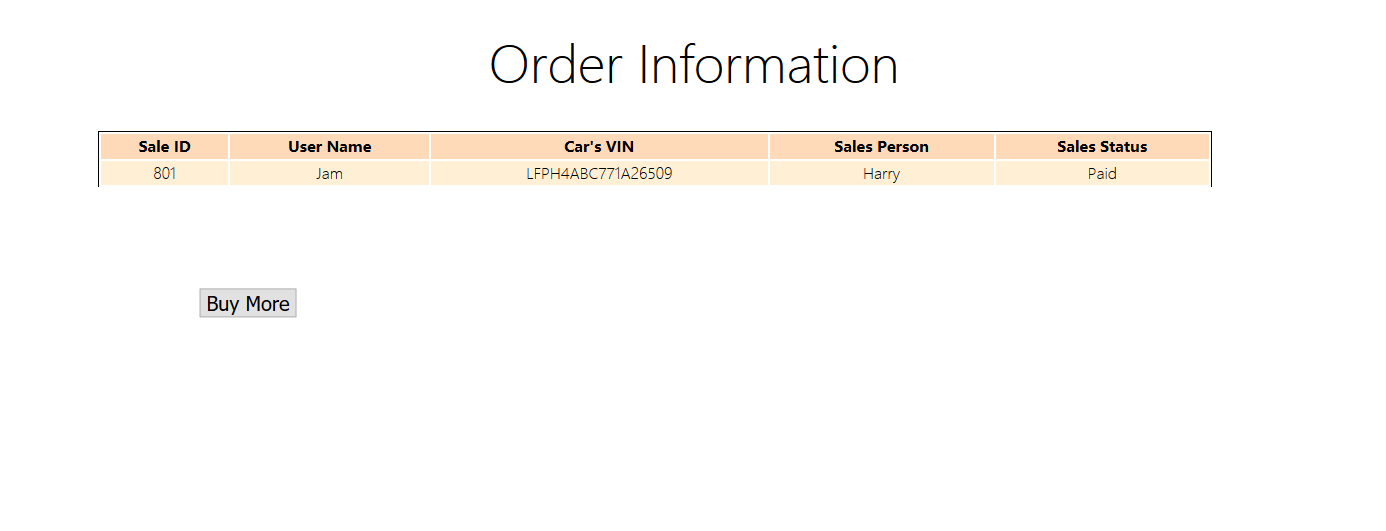


Figure 8

Click buy more button, we will find the record of this car can’t be retrieved by the customer, like Figure 9.Customer can continue to buy the car.

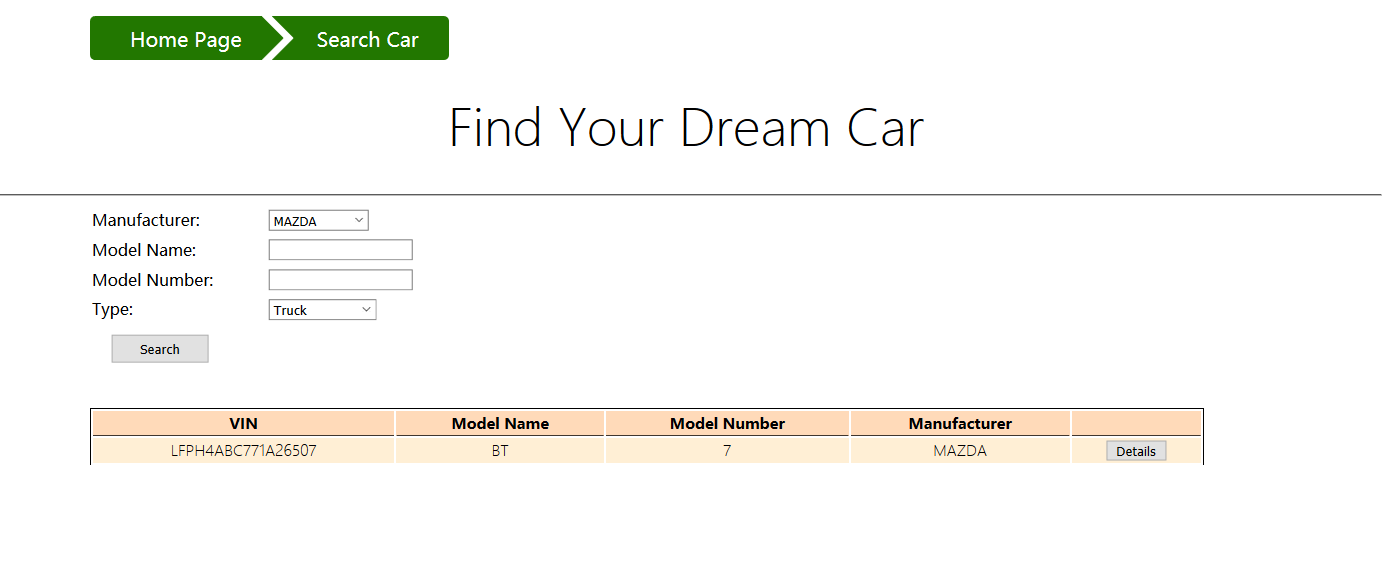


Figure 9

When a salesperson login the system, the salesperson main page is like Figure 10.

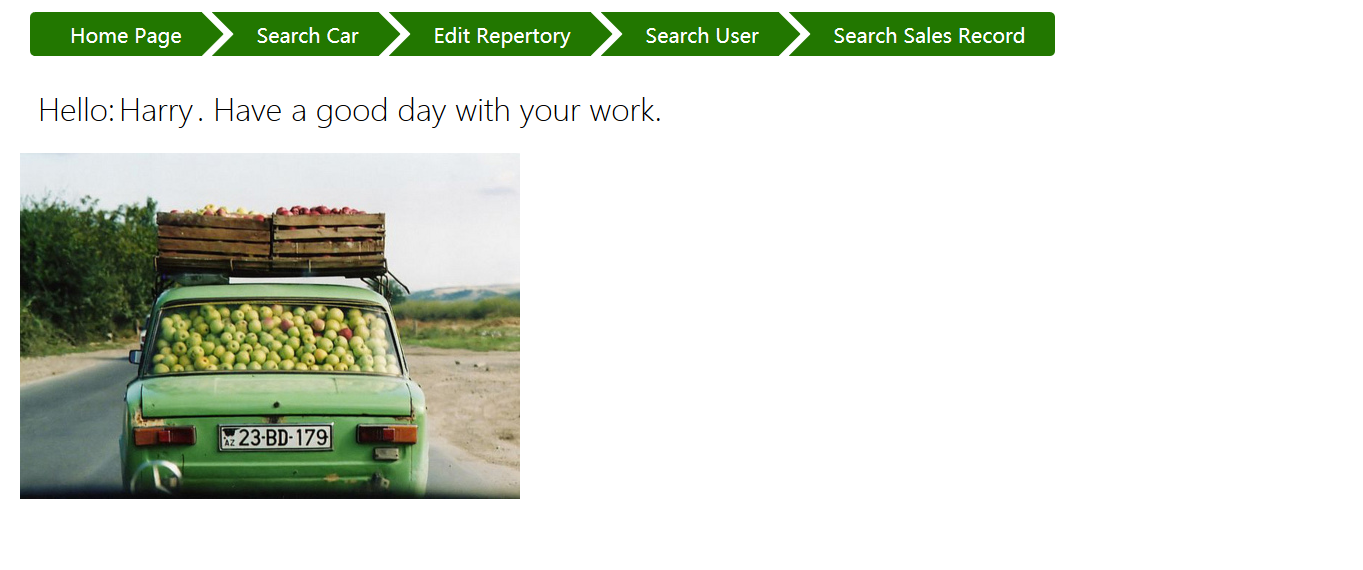


Figure 10

For salesperson it can retrieve all the car of specific query condition no matter the status of car. The search page is like Figure 11.

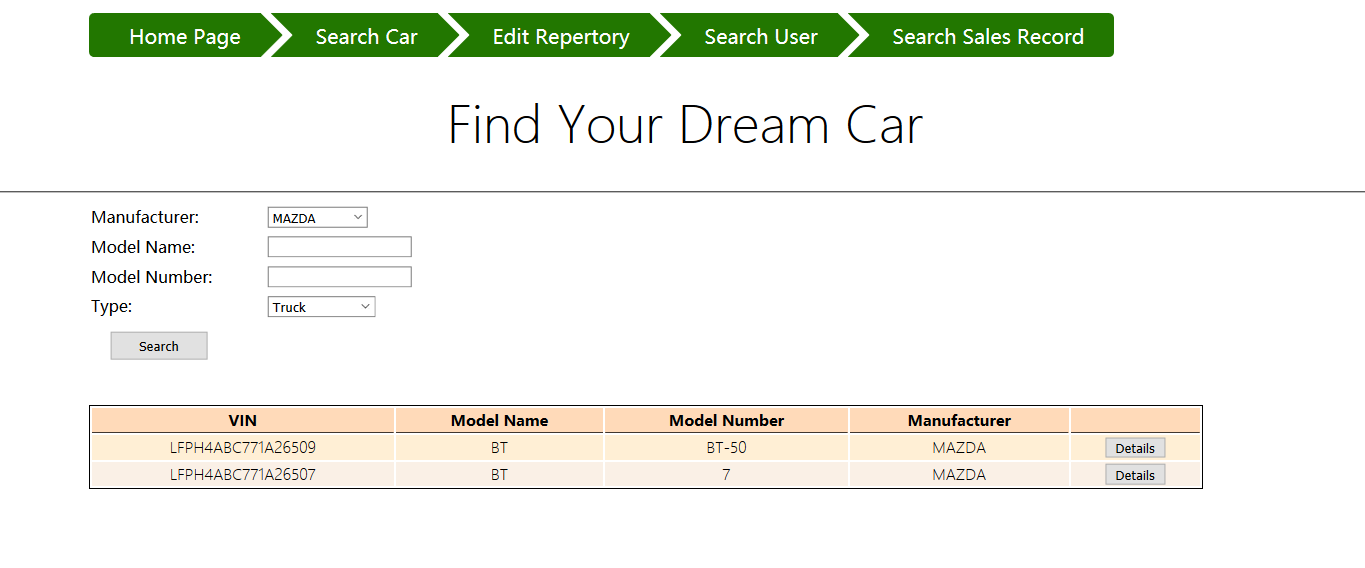


Figure 11

Click the detail button, the detail of the car will be shown, but doesn’t provide a buy car option. Figure 12 show the situation.

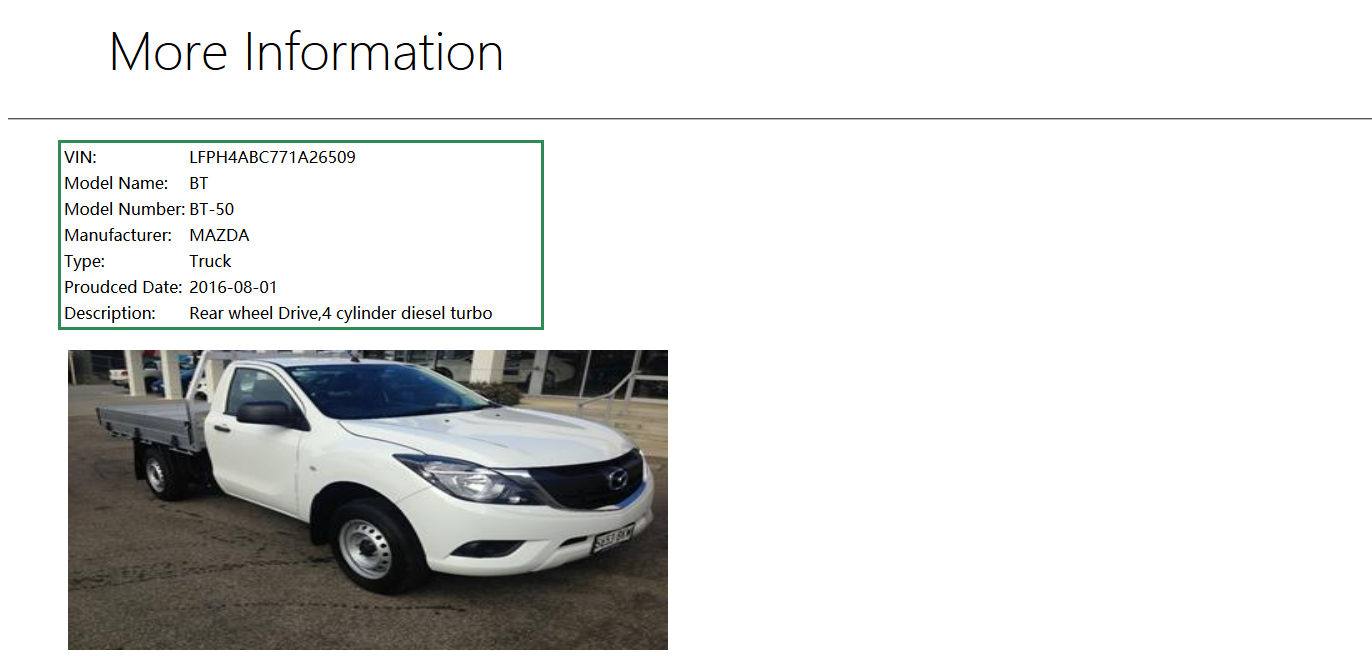


Figure 12

Back to the search car page and click the edit repository on the navigation bar. The edit repository page is like Figure 13, all the car in the repository displayed in the table.

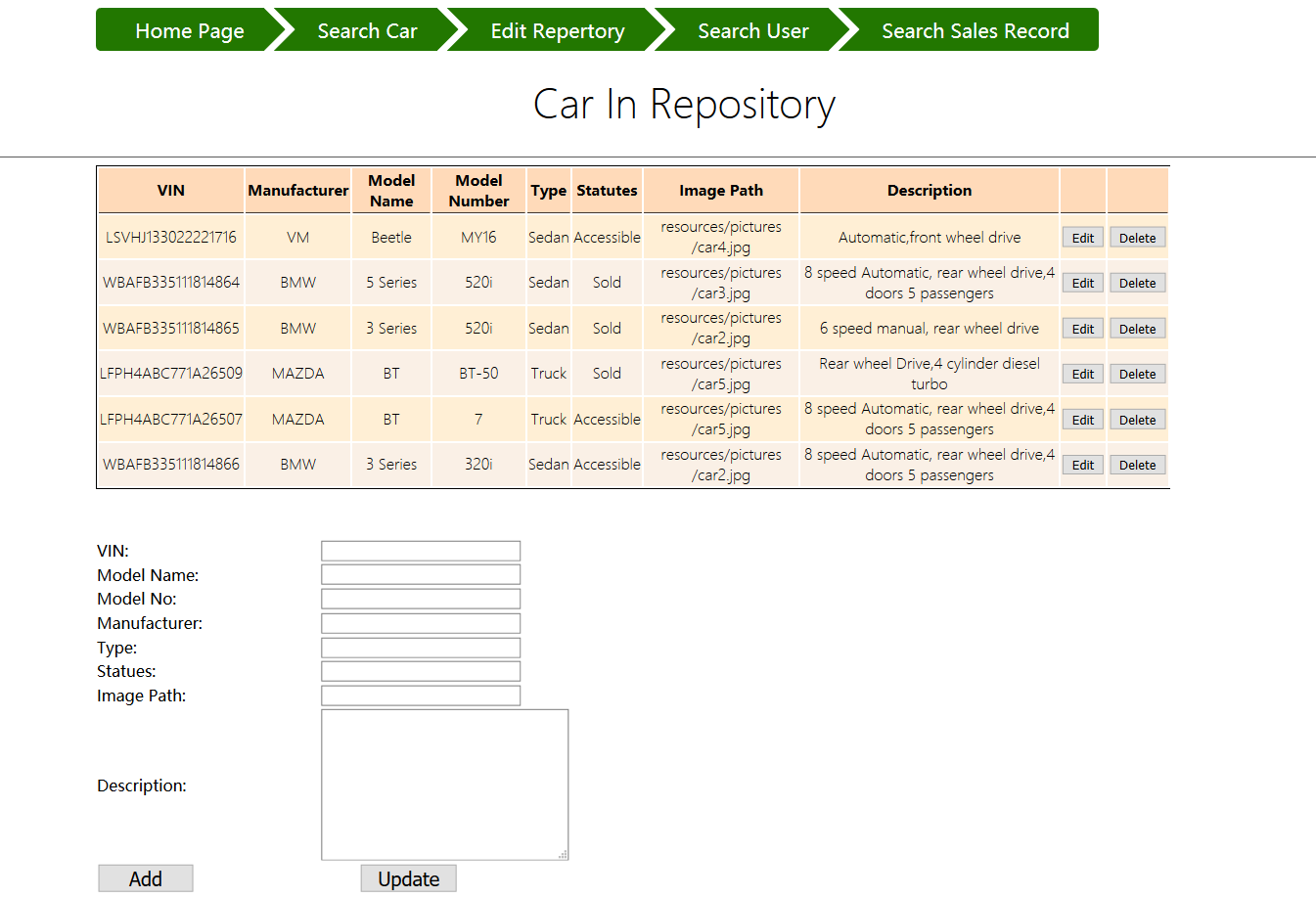


Figure 13

Click the edit button on the last row, the information is displayed into the below form. Shown in Figure 14.

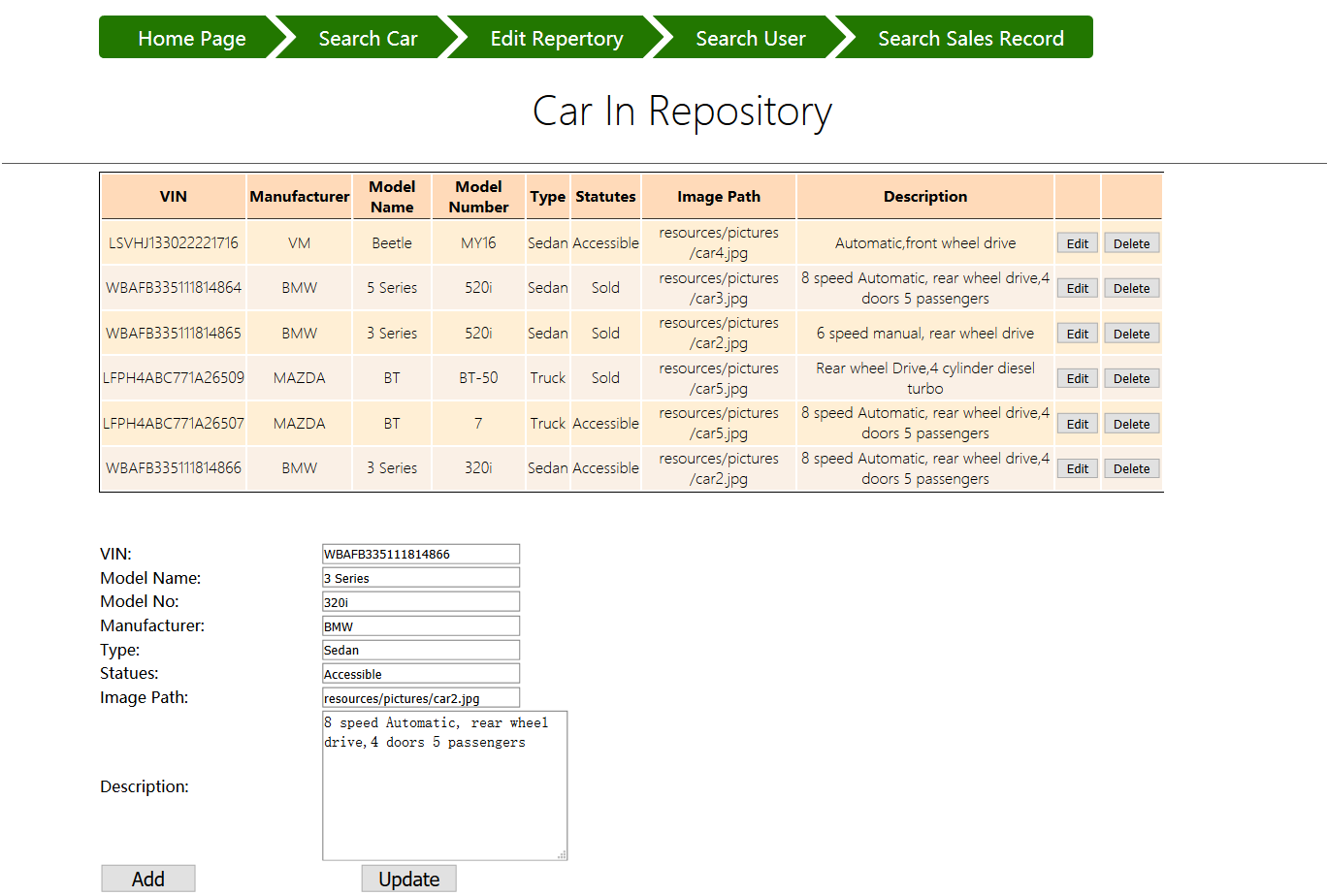
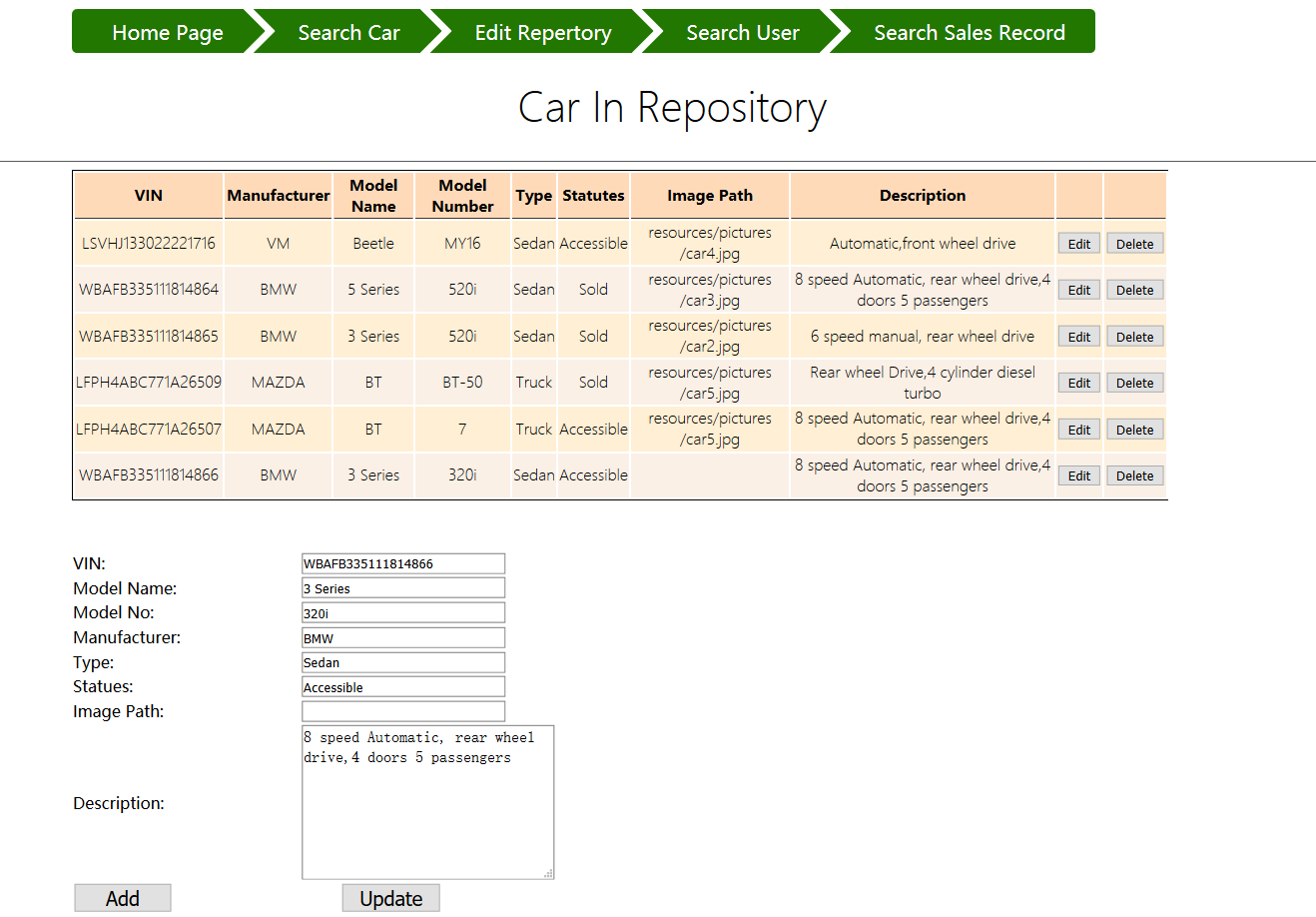


Figure 14

Delete the image path and click update button, the modified records is shown in the table. Shown in figure 15.

Figure 15

When click the delete button on one of the row, here for example last row, this car’s information will be removed. Shown in Figure 16.

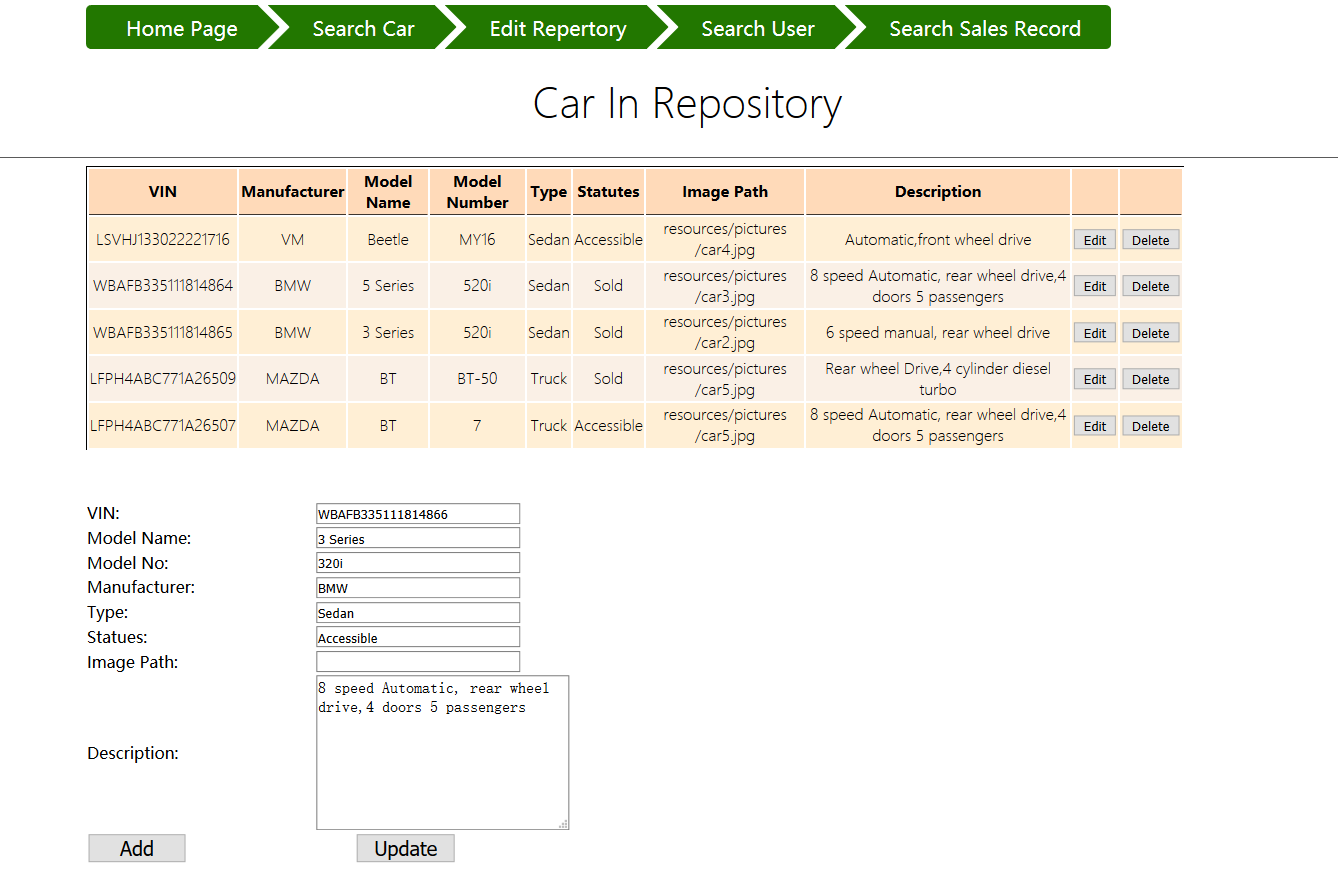


Figure 16

When click add button and input the information in the form, a new record will appear on the table. Shown in Figure 17.

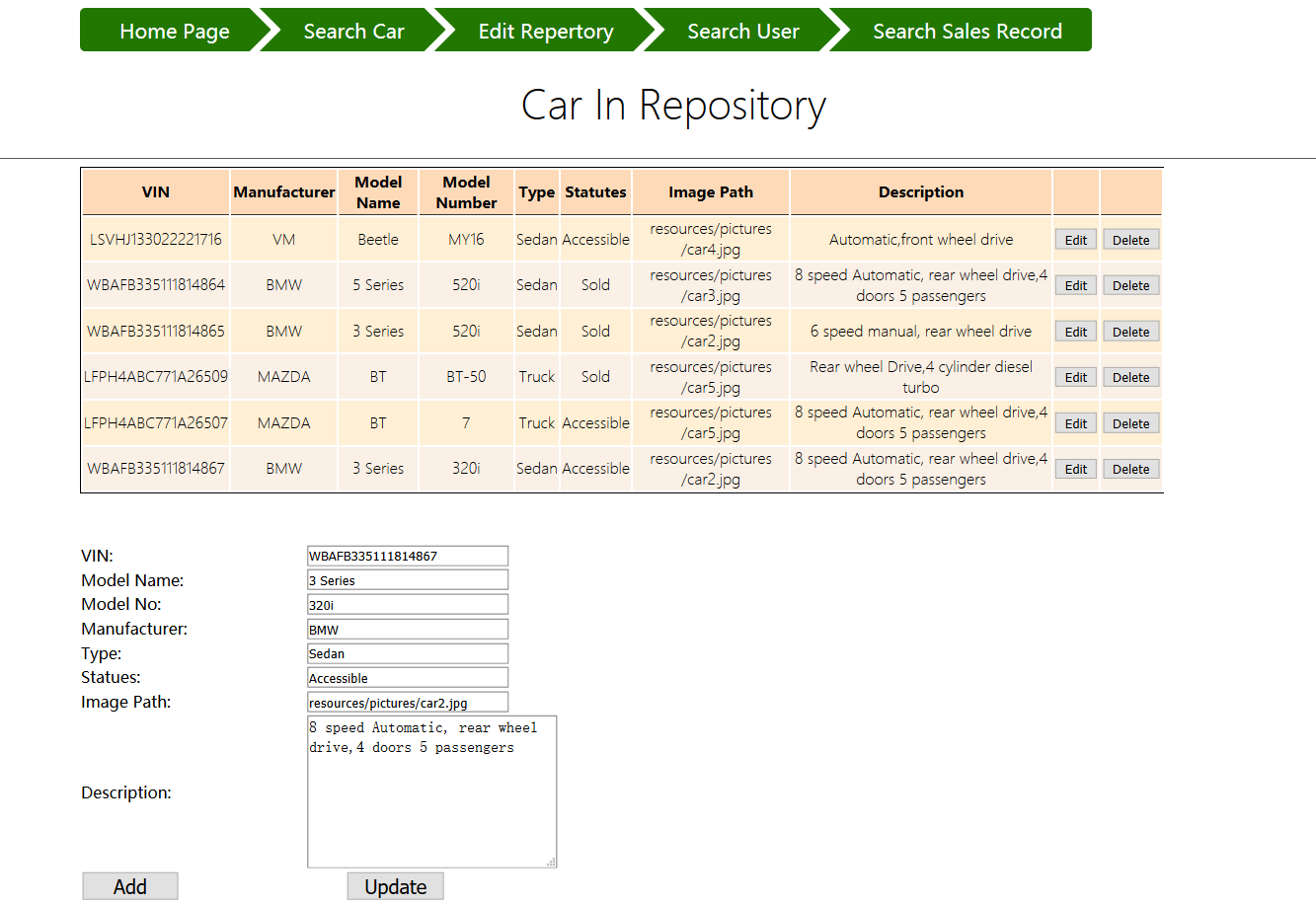


Figure 17

Click the search user on navigation bar, the user search page is like Figure 18.

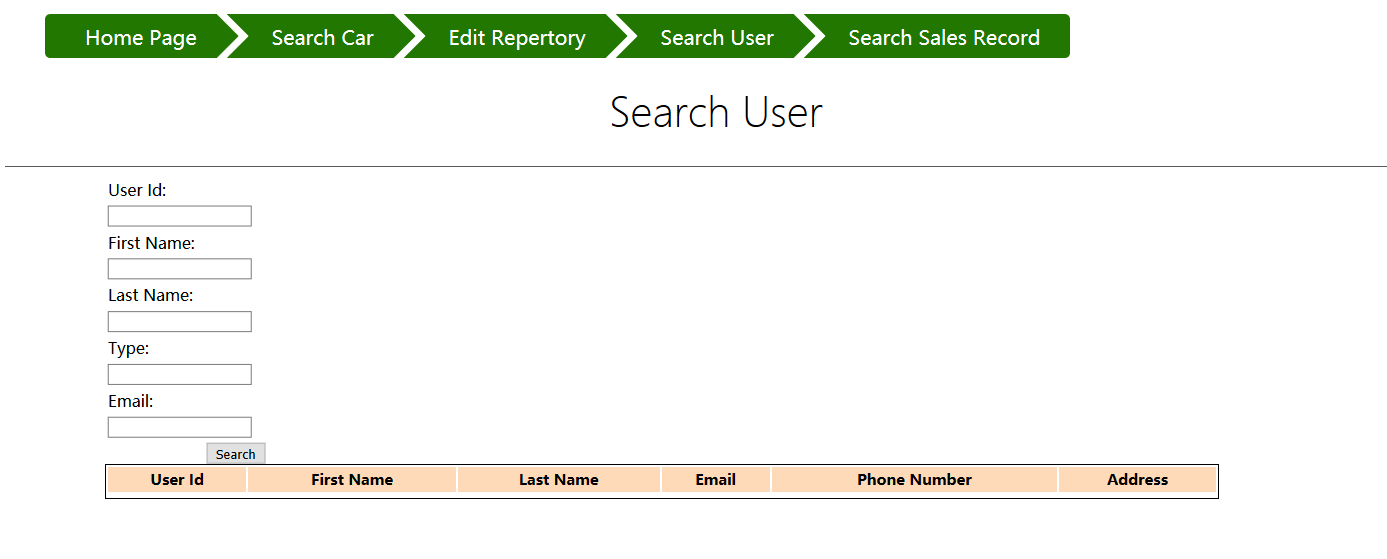


Figure 18

Input the search requirement and click search button, the page is like Figure 19.

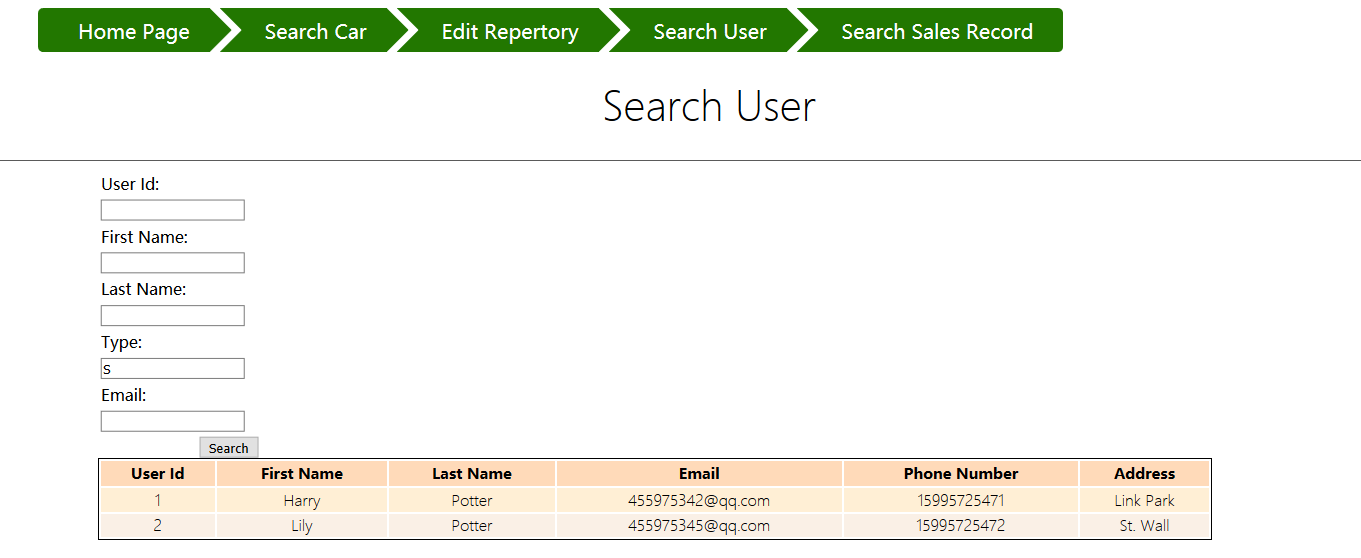


Figure19

Click the search sales record on the navigation bar, the search page is like Figure 20.

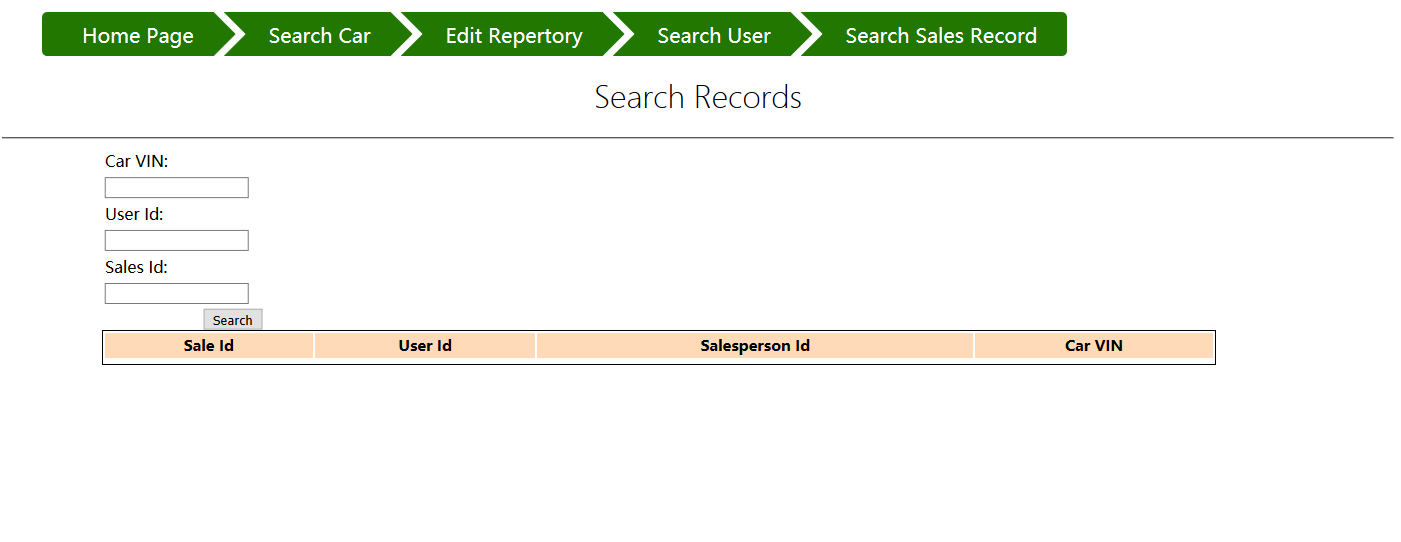


Figure 20

After input the query requirement and click search button the page is like Figure21.

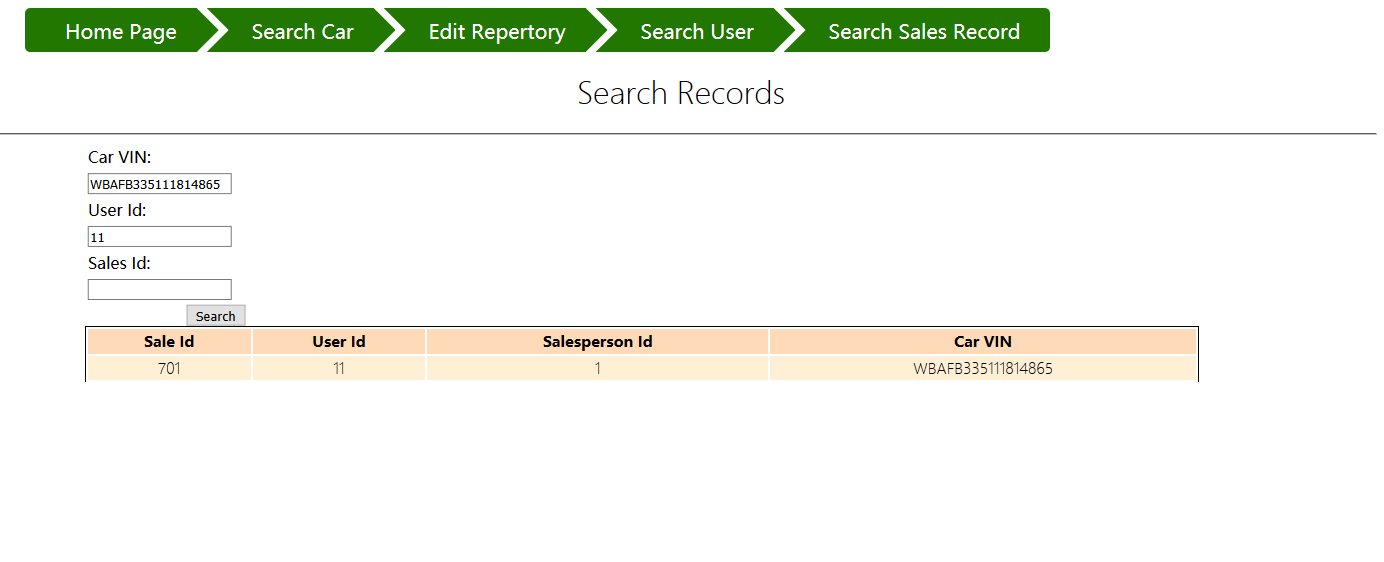


Figure21

1. **Data dictionary**

There are three entities in this system: Users, Car, Sale. So there exists 3 data table in the database. They are USERS table, CAR table and SALE table.

The structure of USERS table in database is shown below:

ID INTEGER DEFAULT GENERATED\_BY\_DEFAULT NOT NULL GENERATED ALWAYS AS IDENTITY,

ADDRESS VARCHAR(255),

EMAIL VARCHAR(255),

FIRSTNAME VARCHAR(255),

LASTNAME VARCHAR(255),

PASSWORD VARCHAR(255),

PHONENUM VARCHAR(255),

"TYPE" VARCHAR(255)

The ID field serves as primary key and. The type filed is used to distinguish the customer and salesperson.

The structure of the CAR table in database is shown below:

VIN VARCHAR(255) NOT NULL,

CARSTATUS VARCHAR(255),

DESCRIPTION VARCHAR(255),

MANUFACTURER VARCHAR(255),

MODELNAME VARCHAR(255),

MODELNO VARCHAR(255),

PRODUCEDDATE DATE, THUMBNAIL VARCHAR(255),

"TYPE" VARCHAR(255)

The VIN filed is serves as primary key and the CARSTATUS is used to store the status of the car such as “available” or “Sold” which will used as constraints when different roles( there is customer and salesperson) to retrieve cars.

The structure of SALE table is shown below:

SALEID BIGINT NOT NULL,

SALEDATE DATE,

SALESSTATUS VARCHAR(255),

CAR\_VIN VARCHAR(255),

CUSTOMER\_ID INTEGER,

SALESPERSON\_ID INTEGER

The SALEID serves as primary key, the filed of SALESSTATUS record the status of this sales record. And the CAR\_VIN, CUSTOMER\_ID and SALESPERSON\_ID are foreign keys. CAR\_VIN referenced the CAR table. And other two reference the USERS table.