# PROJECT PART 2

**COURSE: CS-GY 6083-B, Fall 2020**

**SUBMISSION DATE: 2020/12/20**

**Guangqian Leng (net ID: gl1898)**

**Zulong Ye (net ID: zy2010)**

# Contents

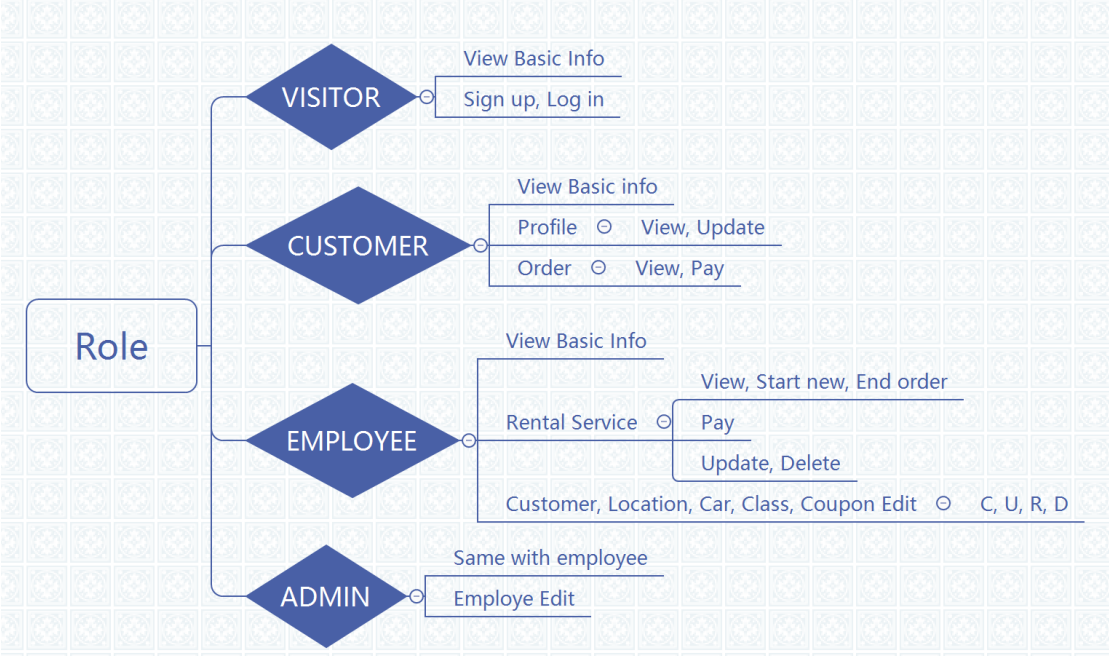
# Execute summary

This is WOW's rental information management system. We refer to mature car rental service websites on the market, design the functions of the system, and modify it according to specific business requirements.

According to the business rule, we design tables in the database like class, corporate, corp_discount, customer, individual, indi_coupon, invoice, location, payment, services, user_password, vehicle. All information will split into several parts stored in these.

By using our system, users can better understand information. For instance, they may know the nearest office affording their favorite cars quickly, and raise propriate requirements to the company. Employees can better manage business information. For Instance, they can give a good recommendation to customers according to information in the system such as order history, vehicle status and so on. The company can improve business and provide better service. For instance, according to the statistics of all customers' order, the company may know which office is the most favorite location for pick up, then the company can put more cars here and expand scale of this office to improve business performance.

There are 4 roles (Visitor, Customer, Employee, Administrator) in our system. Visitors can only view some general information such as the offices in different locations and the cars we have. After signup and login, visitor will change to Customer. Customers can see and edit their profile as well as view their order history. Employee is a core role of information management. He or she can read, update, create and delete the information in the system. When they serve the customers through the system, it may check whether data match the database, compute the price automatically, and generate an order. Admin can not only manage the information as Employee do, but also administer the account of all the employees.



# 1. Brief details

Software

- XAMPP, Chrome, Safari
- Editor: Sublime Text, Visual Studio Code
- Version control: git, Github

Language

- Front end: HTML, CSS, JavaScript
- Back end: PHP

Database

- MySQL

# 2. Table Schema

## 2.1 List of Tables

| Table | Action | Rows | Type | Collation | Size | Overhead |
|---|---|---|---|---|---|---|
| class | Browse Structure Search Insert Empty Drop | 3 | InnoDB | utf8mb4_general_ci | 16.0 KiB | - |
| corporate | Browse Structure Search Insert Empty Drop | 1 | InnoDB | utf8mb4_general_ci | 32.0 KiB | - |
| corp_discount | Browse Structure Search Insert Empty Drop | 2 | InnoDB | utf8mb4_general_ci | 16.0 KiB | - |
| customer | Browse Structure Search Insert Empty Drop | 3 | InnoDB | utf8mb4_general_ci | 16.0 KiB | - |
| individual | Browse Structure Search Insert Empty Drop | 2 | InnoDB | utf8mb4_general_ci | 32.0 KiB | - |
| indi_coupon | Browse Structure Search Insert Empty Drop | 3 | InnoDB | utf8mb4_general_ci | 16.0 KiB | - |
| invoice | Browse Structure Search Insert Empty Drop | 3 | InnoDB | utf8mb4_general_ci | 48.0 KiB | - |
| location | Browse Structure Search Insert Empty Drop | 4 | InnoDB | utf8mb4_general_ci | 16.0 KiB | - |
| payment | Browse Structure Search Insert Empty Drop | 5 | InnoDB | utf8mb4_general_ci | 32.0 KiB | - |
| services | Browse Structure Search Insert Empty Drop | 5 | InnoDB | utf8mb4_general_ci | 64.0 KiB | - |
| user_password | Browse Structure Search Insert Empty Drop | 5 | InnoDB | utf8mb4_general_ci | 16.0 KiB | - |
| vehicle | Browse Structure Search Insert Empty Drop | 10 | InnoDB | utf8mb4_general_ci | 48.0 KiB | - |
| 12 tables | Sum | 46 | InnoDB | utf8mb4_general_ci | 352.0 KiB | 0 B |

## 2.2 Records

| class_id<br>the ID of a vehicle class | class_name<br>the name of a vehicle class | rental_rate<br>the rantal rate of a vehicle class | over_fee<br>the over mailage fee of a vehicle class |
|---|---|---|---|
| 1 | small car | 10.00 | 1.00 |
| 2 | mid-size car | 20.00 | 2.00 |
| 3 | SUV | 30.00 | 3.00 |

**Class**

| cust_id<br>the unique ID of a customer | emp_id<br>the Employee ID of the customer who rents the car ... | corp_reg_no |
|---|---|---|
| 111 | 1 | 1 |

**Corporate**

| corp_reg_no<br>the register no of a corporation | corp_name<br>the name of a corporation | corp_dis<br>the corporation coupon discount |
|---|---|---|
| 1 | asd | 0.80 |
| 2 | apple | 0.50 |

## Corporation Discount

| cust_id<br>the unique ID of a customer | fname<br>first name of customer | lname<br>last name of customer | cust_street<br>the street of a customer | cust_city<br>the city of a customer | cust_state<br>the state of a customer | cust_zipcode<br>the zipcode of a customer | email<br>the email of a customer | cust_phone_no<br>the phone number of a customer | cust_type<br>the customer type |
|---|---|---|---|---|---|---|---|---|---|
| 106 | qw | ert | <script> | new york | NEW | 11201 | qwert@qwert.com | 1234567890 | I |
| 111 | as | dfg | 5 metrotech center | new york | NEW YORK | 11200 | asdfg@asdfg.com | 1234567890 | C |
| 113 | zxc | vb | 1, SW | Beijing | AL | 10055 | zxcvb@zxcvb.com | 1234567899 | I |

## Customer

| cust_id<br>the unique ID of a customer | driver_lno<br>the driver license number of an individual | insur_cop_name<br>the insurance company name of an individual | insur_pol_no<br>the insurance policy number of an individual | coupon_id |
|---|---|---|---|---|
| 106 | 111 | as | 11111 | 9999 |
| 113 | 1 | 1 | 1 | 9999 |

## Individual

| coupon_id<br>the unique ID of a kind of coupons | indi_dis<br>the discount of an individual | start_date<br>the start date of a individual coupon | end_date<br>the end date of a individual coupon |
|---|---|---|---|
| 1 | 0.80 | 2020-01-01 00:00:00 | 2021-12-31 00:00:00 |
| 2 | 0.50 | 2020-01-01 00:00:00 | 2021-12-03 00:00:00 |
| 9999 | 1.00 | 2020-12-01 17:42:36 | 2099-12-31 00:00:00 |

## Indi-coupon

| invoice_id<br>the invoice ID | invoice_date<br>the invoice date | invoice_amount<br>the invoice amount | cust_id | service_id | status | real_amount | remain_amount |
|---|---|---|---|---|---|---|---|
| 1 | 2020-12-31 00:00:00 | 300.00 | 106 | 1 | paid | 300.00 | 0.00 |
| 2 | 2999-01-01 00:00:00 | 999999.00 | 106 | 2 | unfinished | 0.00 | 0.00 |
| 6 | 2020-12-11 00:00:00 | 100.00 | 106 | 6 | unpaid | 100.00 | 70.00 |

## Invoice

| location_id<br>the unique ID of a location | loc_street<br>the street of a location | loc_city<br>the city of a location | loc_state<br>the state of a location | loc_zipcode<br>the zipcode of a location | loc_phone_no<br>the phone number of a location |
|---|---|---|---|---|---|
| 1 | 25 Street | New York | New York | 54321 | 123456 |
| 2 | 76 Street | Buffalo | New York | 12345 | 123456 |
| 3 | 12 Street | Pittsburgh | Pennsylvania | 12345 | 123456 |
| 9999 | test | test | test | 99999 | 9999999999 |

## Location

| pay_id<br>the payment ID | pay_date<br>the payment date | pay_method<br>the payment method | pay_card_no<br>the card number for a payment | invoice_id | pay_amount |
|---|---|---|---|---|---|
| 1607832220 | 2020-12-13 04:03:40 | C | 123 | 1 | 100.00 |
| 1607832275 | 2020-12-13 04:04:35 | D | 1234 | 1 | 200.00 |
| 1607872545 | 2020-12-13 15:15:45 | D | 12 | 6 | 10.00 |
| 1607917750 | 2020-12-14 03:49:10 | C | 111 | 6 | 10.00 |
| 1607921453 | 2020-12-14 04:50:53 | C | 123 | 6 | 10.00 |

## Payment

| service_id<br>the service ID | pick_date<br>the pick up date | drop_date<br>the drop off date | start_odometer<br>the start odometer | end_odometer<br>the end odometer | d_limit<br>the Daily Odometer Limit for the rental service (o… | vin | pick_location_id | drop_location_id |
|---|---|---|---|---|---|---|---|---|
| 1 | 2020-12-01 00:00:00 | 2020-12-31 00:00:00 | 10.00 | 1000.00 | 200.00 | 1 | 1 | 2 |
| 2 | 2020-11-01 00:00:00 | 2999-01-01 00:00:00 | 100.00 | 9999.00 | 10.00 | 2 | 1 | 9999 |
| 3 | 2020-12-01 00:00:00 | 2021-01-11 00:00:00 | 100.00 | 1000.00 | 50.00 | 3 | 2 | 1 |
| 4 | 2020-12-01 00:00:00 | 2020-12-10 00:00:00 | 100.00 | 1000.00 | 10.00 | 4 | 2 | 3 |
| 6 | 2020-12-01 00:00:00 | 2020-12-11 00:00:00 | 10.00 | 1000.00 | 200.00 | 4 | 2 | 2 |

**Services**

| u_id | username | password | u_type |
|---|---|---|---|
| 1 | admin@wow.com | 21232f297a57a5a743894a0e4a801fc3 | ADMIN |
| 106 | qwert@qwert.com | a384b6463fc216a5f8ecb6670f86456a | CUSTOMER |
| 107 | emp1@wow.com | 86ea3363ba65c10f3f1ef299b126de29 | EMPLOYEE |
| 111 | asdfg@asdfg.com | 040b7cf4a55014e185813e0644502ea9 | CUSTOMER |
| 113 | zxcvb@zxcvb.com | eb89f40da6a539dd1b1776e522459763 | CUSTOMER |

**User Password**

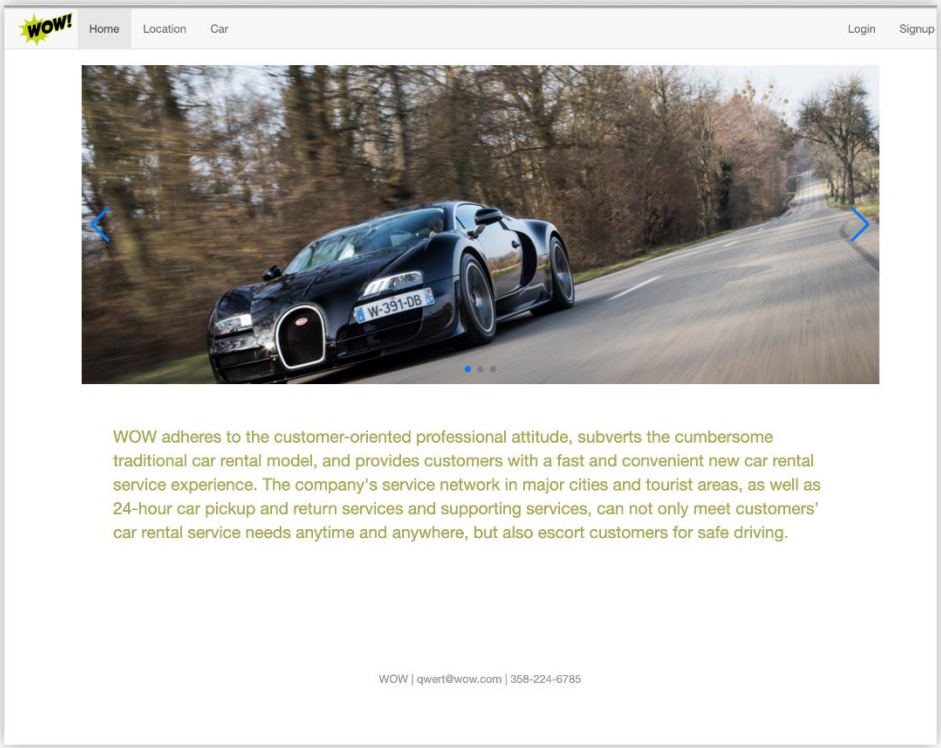
| make<br>he make of a vehicle | model<br>the model of a vehicle | year<br>the produce date of a vehicle | vin<br>the VIN of a vehicle | lpn<br>the License Plate number of a vehicle | class_id | location_id |
|---|---|---|---|---|---|---|
| Benz | BenzSC | 2010-09-01 22:11:13 | 1 | 1111 | 1 | 2 |
| Audi | AudiSW | 2005-01-01 00:00:00 | 10 | 00000 | 2 | 3 |
| Benz | BenzMC | 2000-01-01 00:00:00 | 2 | 22222 | 2 | 1 |
| Benz | BenzSUV | 2000-01-01 00:00:00 | 3 | 33333 | 3 | 1 |
| BMW | BMWSC | 2000-01-01 00:00:00 | 4 | 44444 | 1 | 2 |
| BMW | BMWLC | 2000-01-01 00:00:00 | 5 | 55555 | 3 | 3 |
| BMW | BMWSUV | 2005-01-01 00:00:00 | 6 | 66666 | 3 | 3 |
| Audi | AudiSC | 2005-01-01 00:00:00 | 7 | 77777 | 1 | 2 |
| Audi | AudiPSUV | 2005-01-01 00:00:00 | 8 | 88888 | 3 | 1 |
| Audi | AudiSV | 2005-01-01 00:00:00 | 9 | 99999 | 3 | 2 |

**Vehicle**

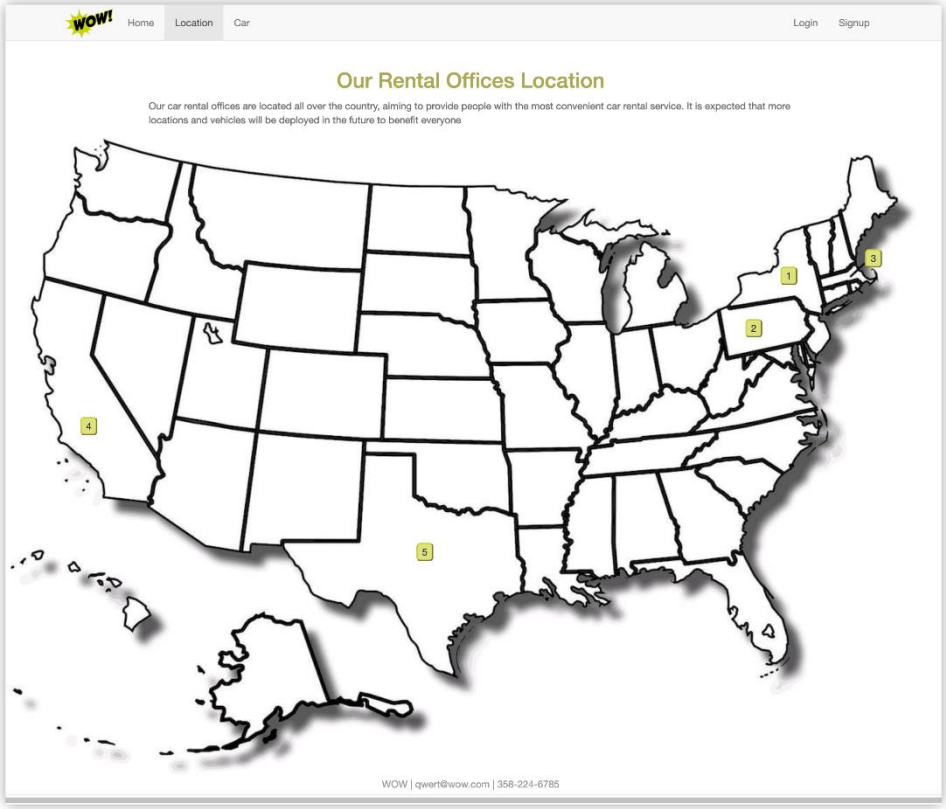
# 3. Web Application

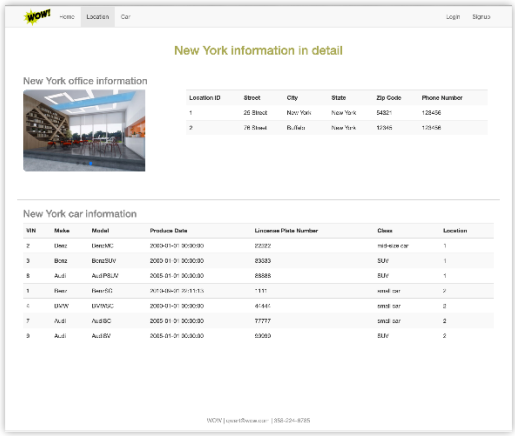
## 3.1 Screen Shots

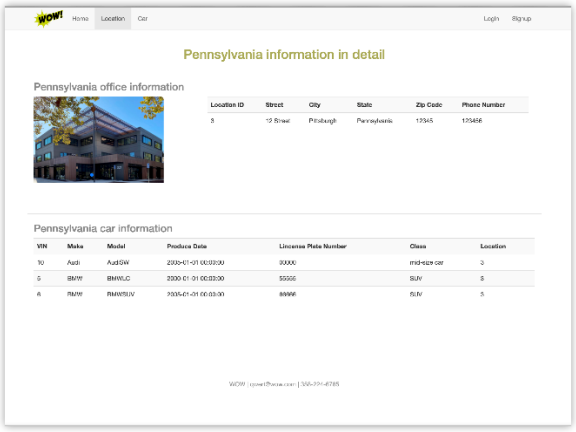
### 3.1.1 Homepage




### 3.1.2 Location

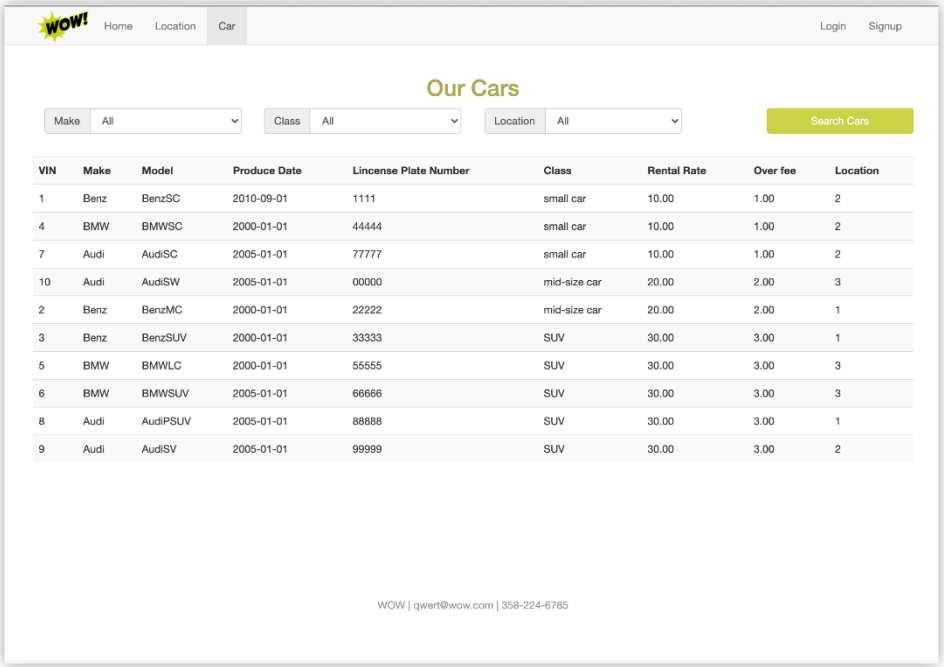


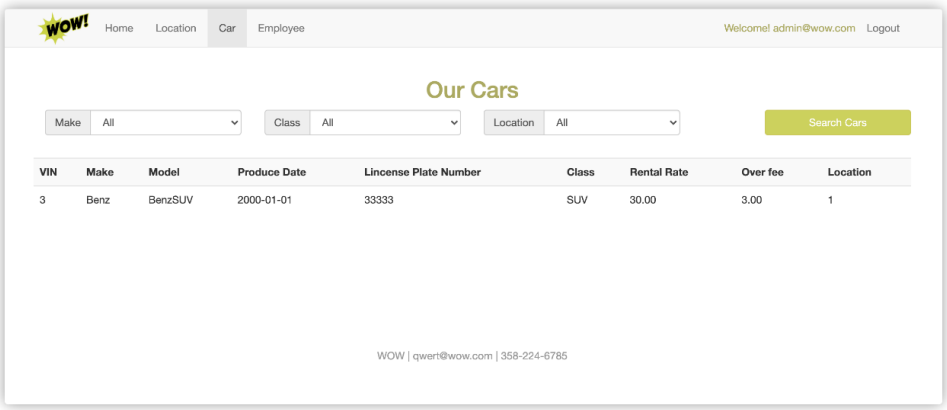



When we click specific locations, we can get location information.

### 3.1.3 Car

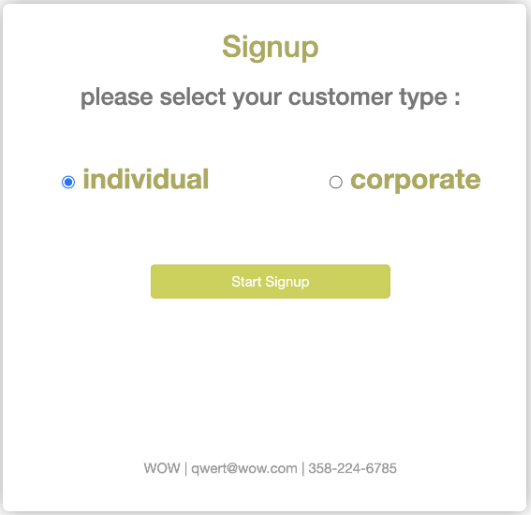



When we filter to search Make (Benz) and Class (SUV), we can get the specific result from all cars.

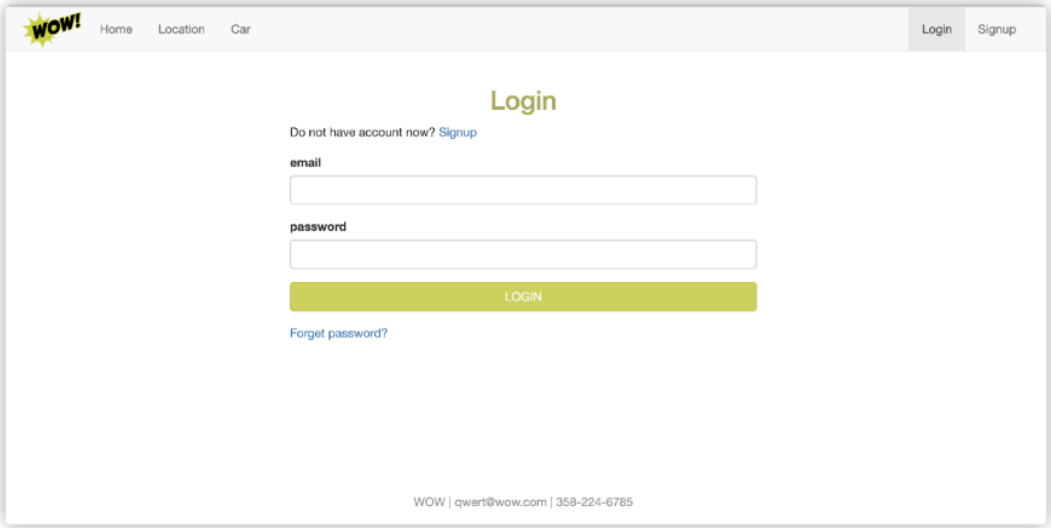
### 3.1.4 Customer Operation

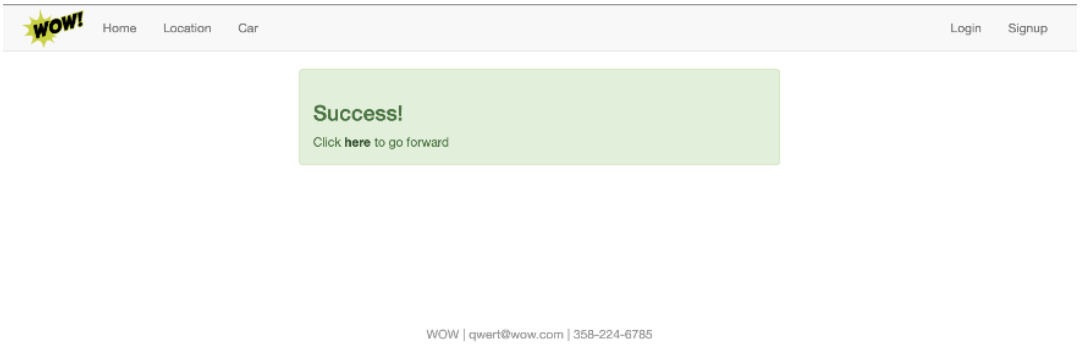
## Signup



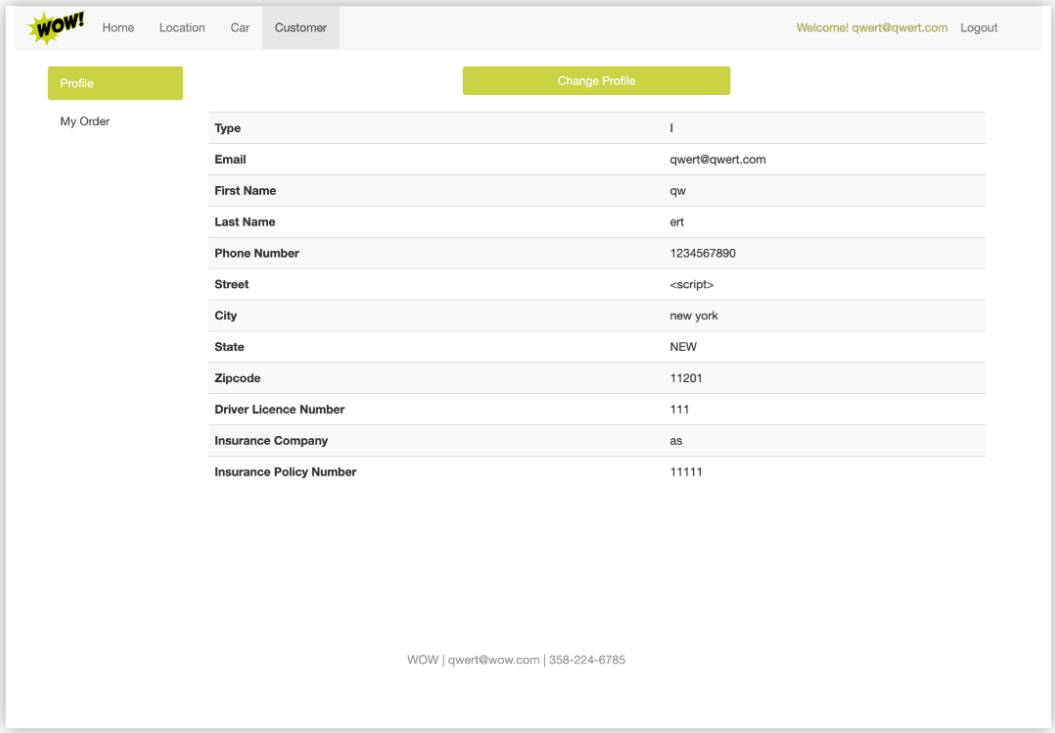


## Login

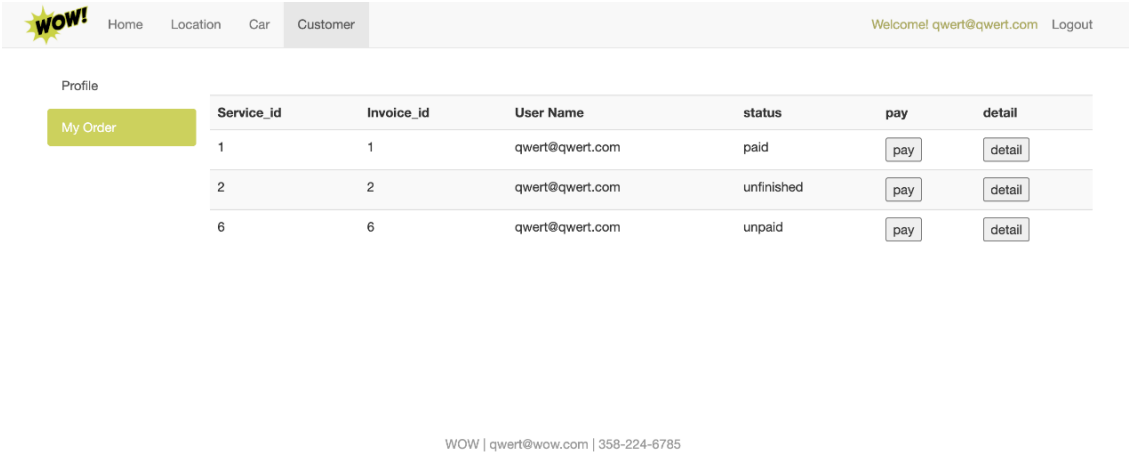


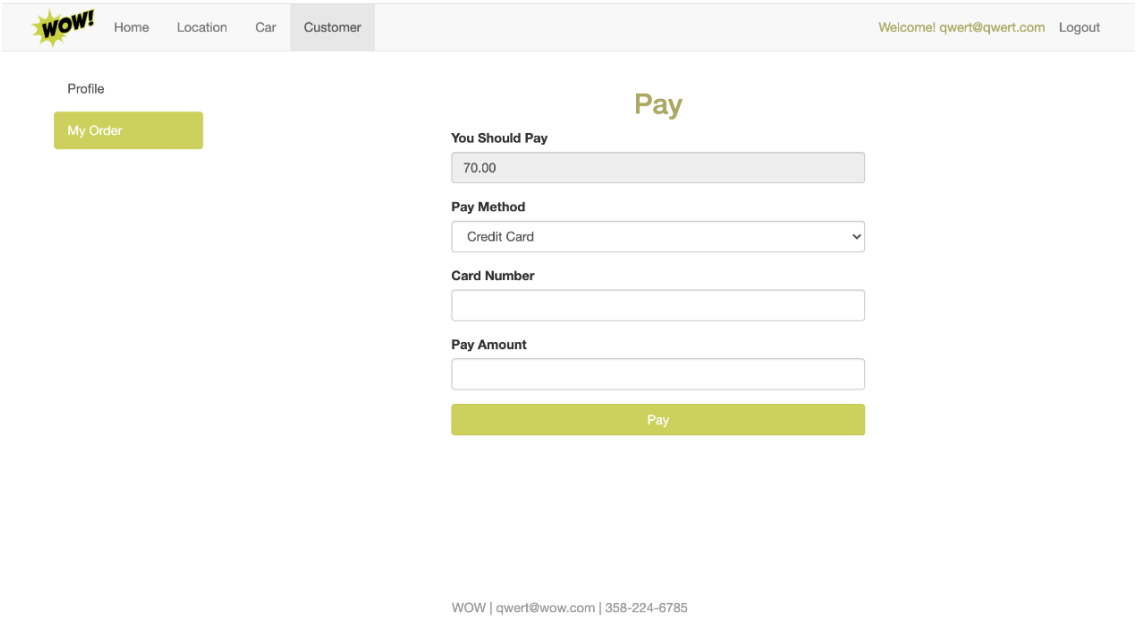

## Customer Information



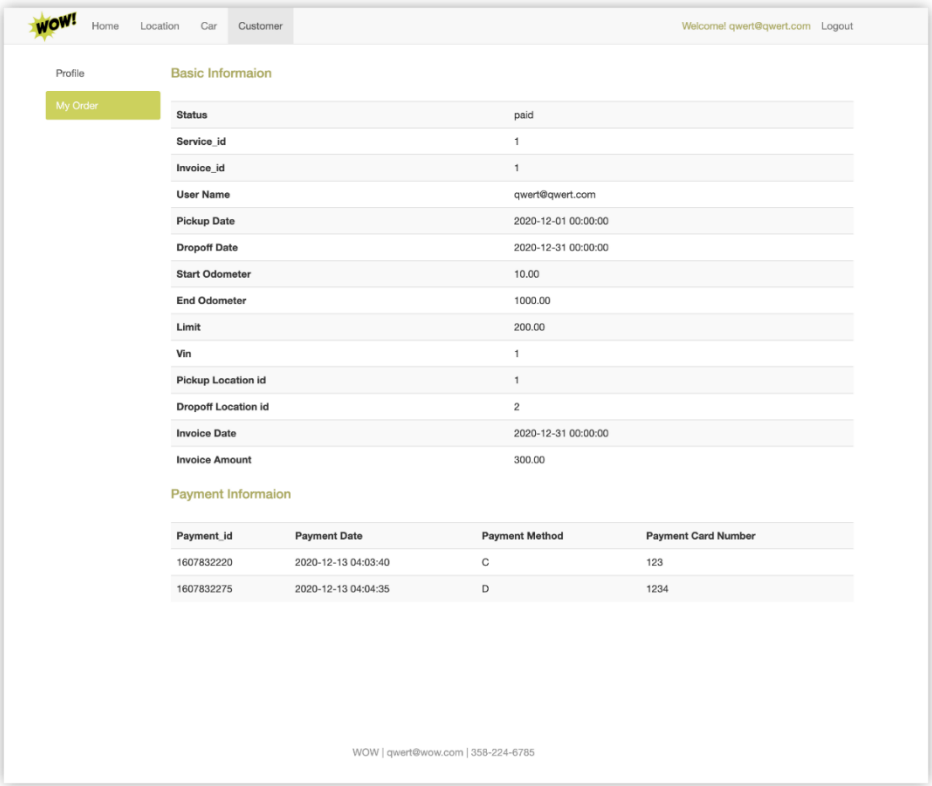
## Orders of the customer




## Payment

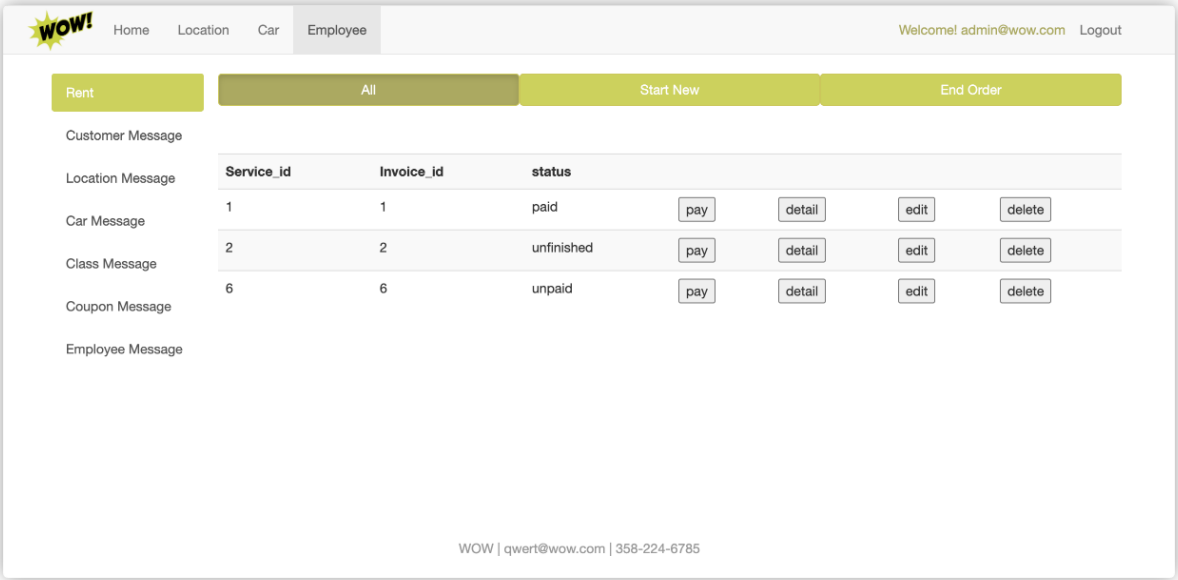



## Order Details

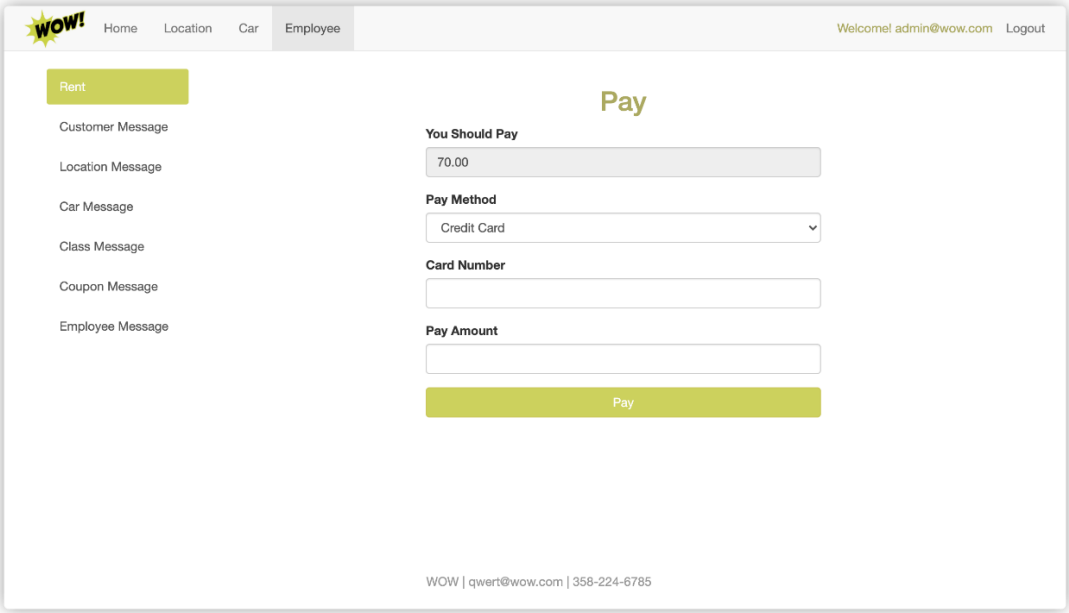



### 3.1.5 Employee Operation

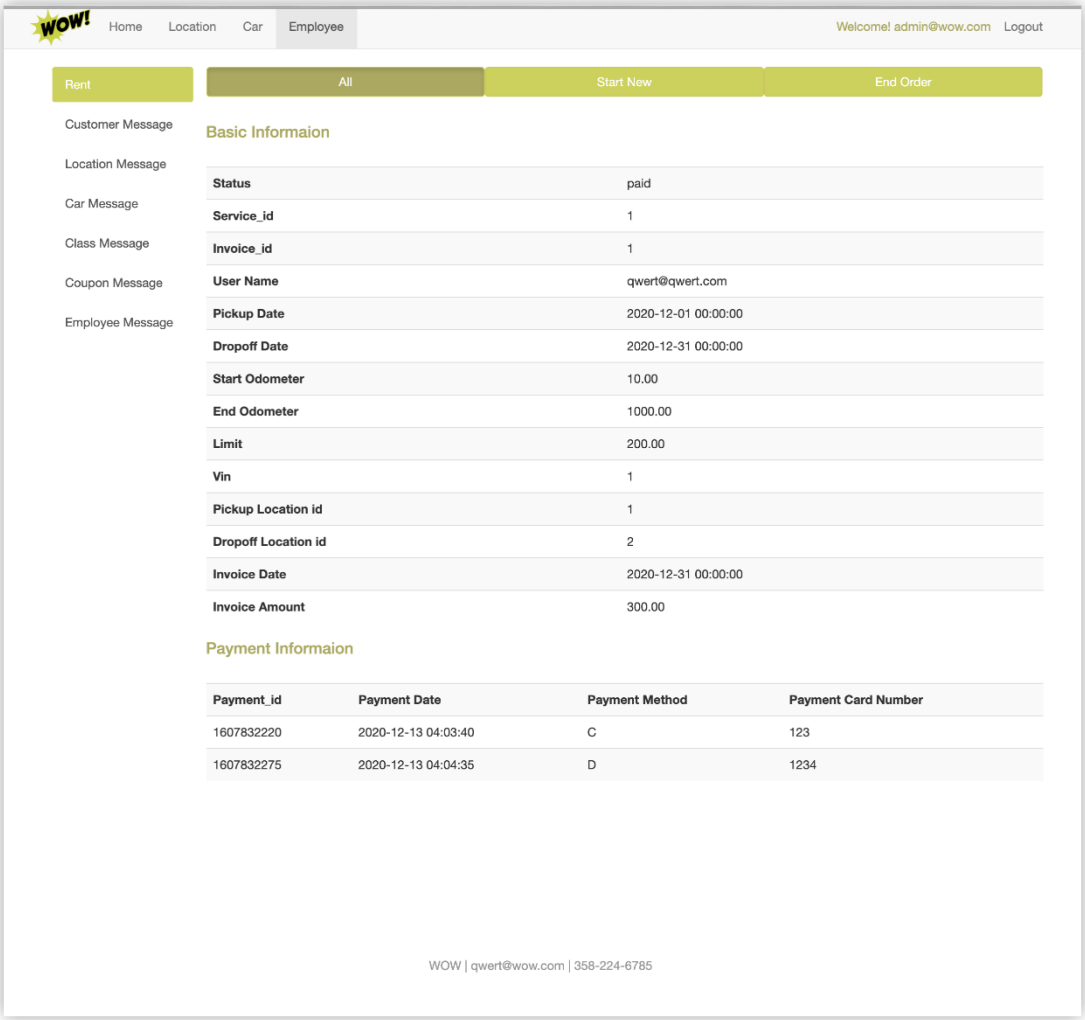
## All rental order



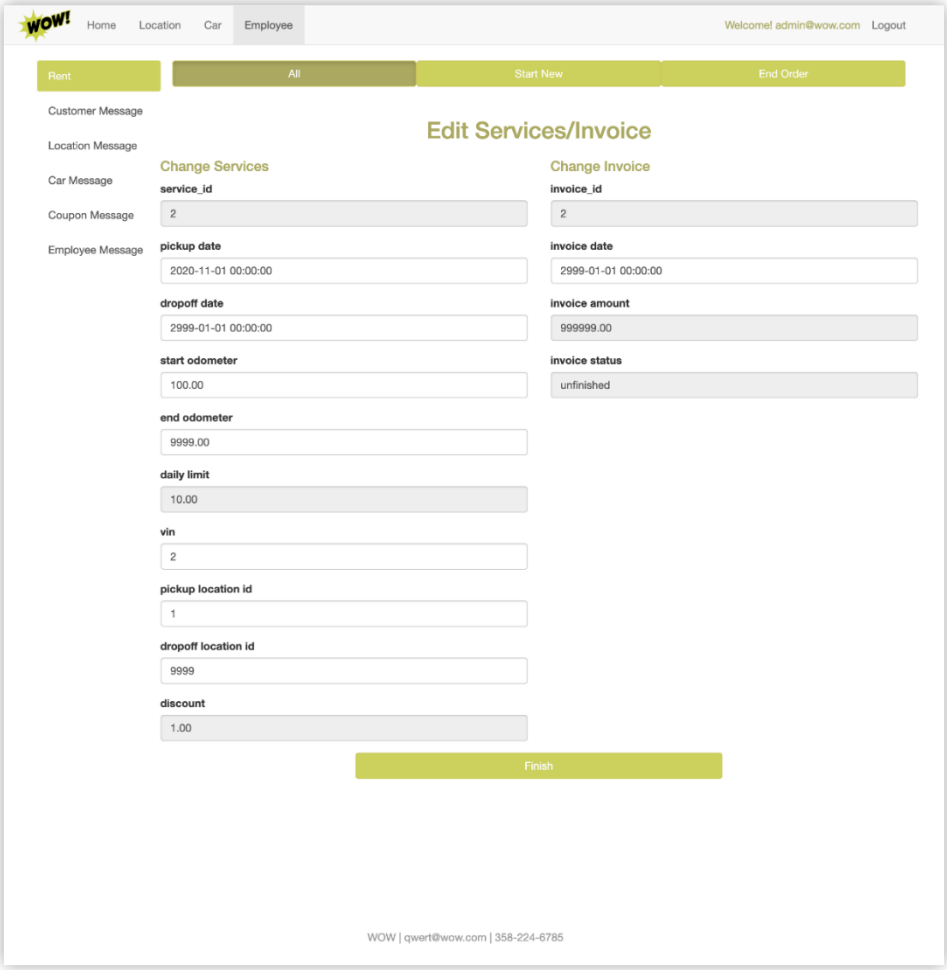
## Pay for customer



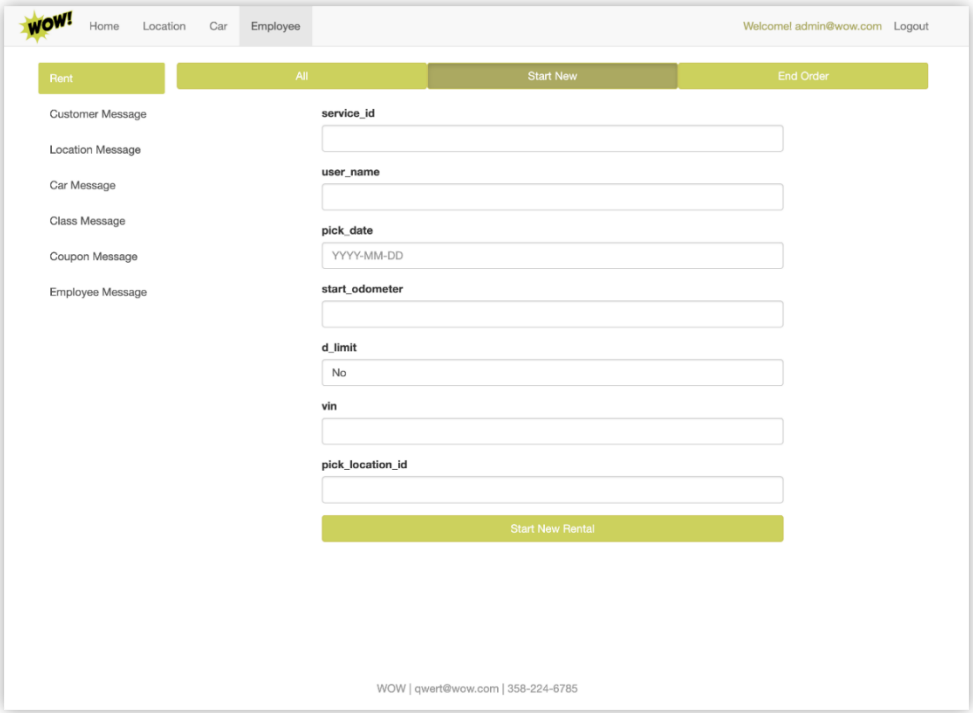

## Order detail of a customer



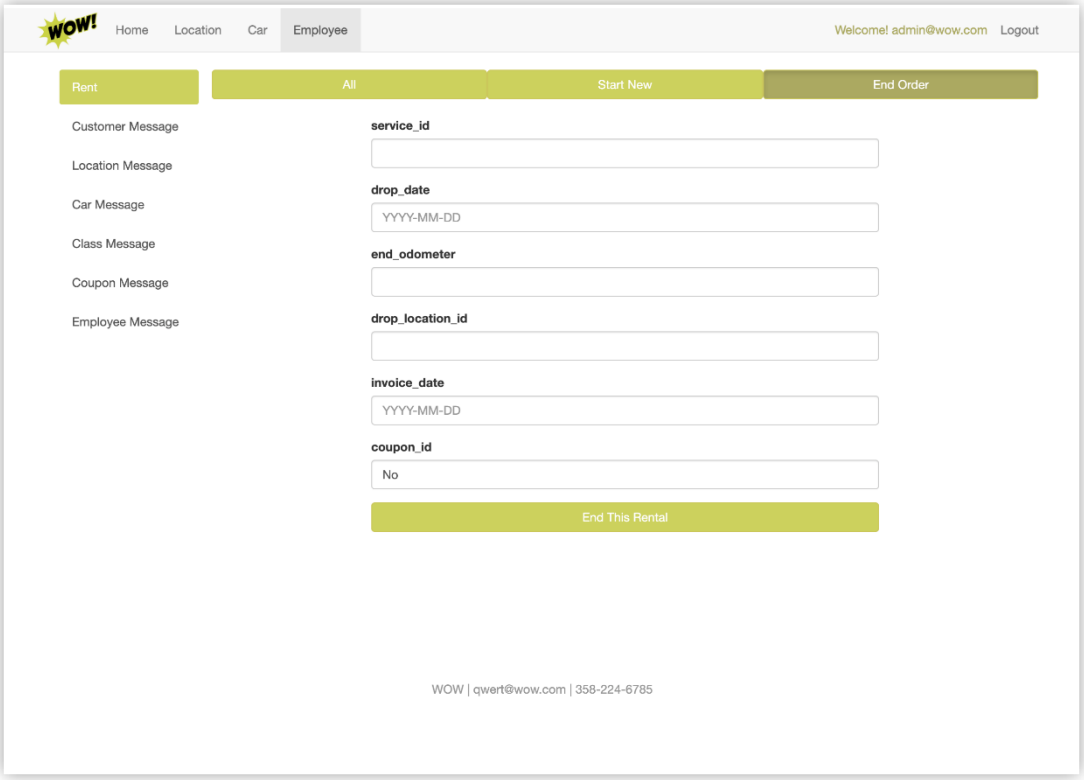
## Edit an order information



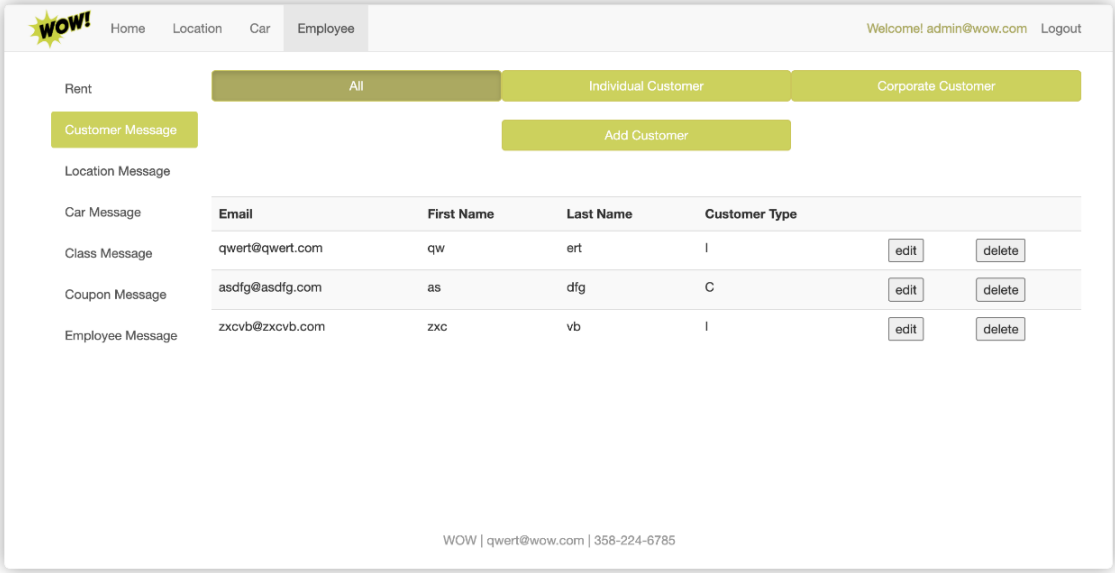

## Start a new order for a customer

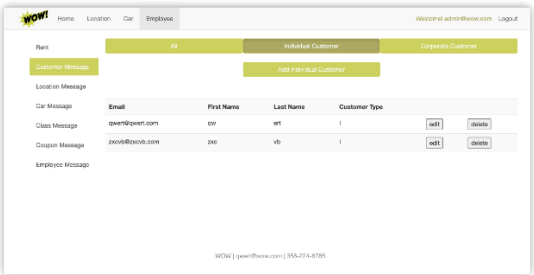


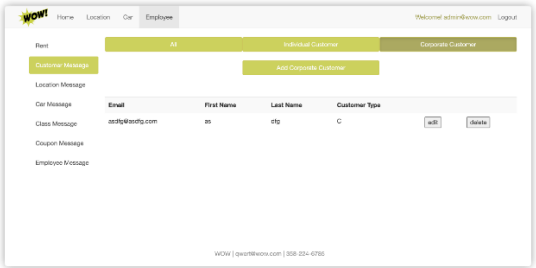
## End an order after the vehicle is returned



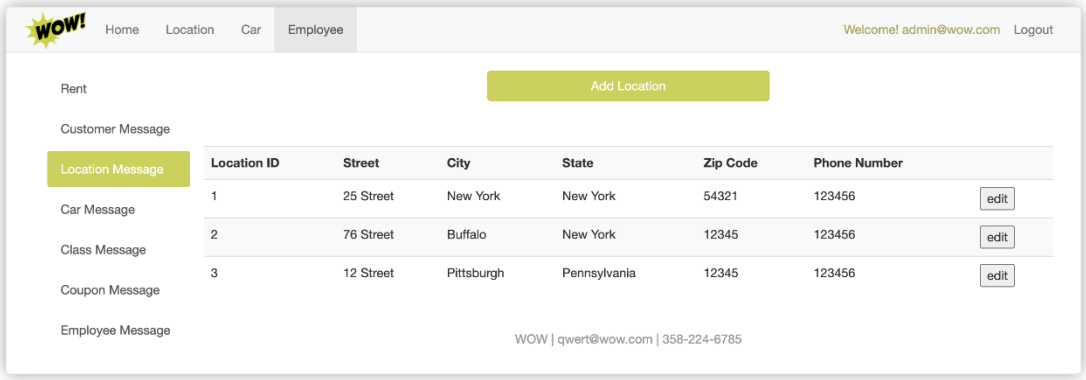

## Customer Management

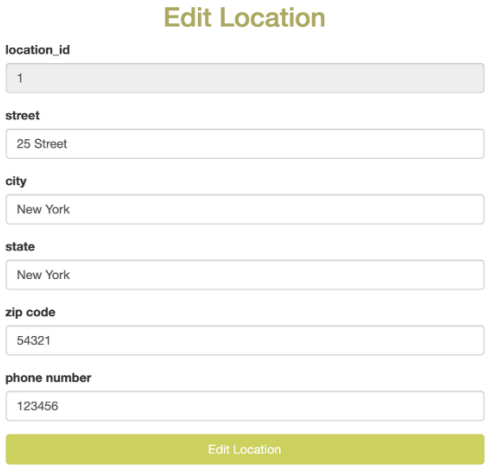


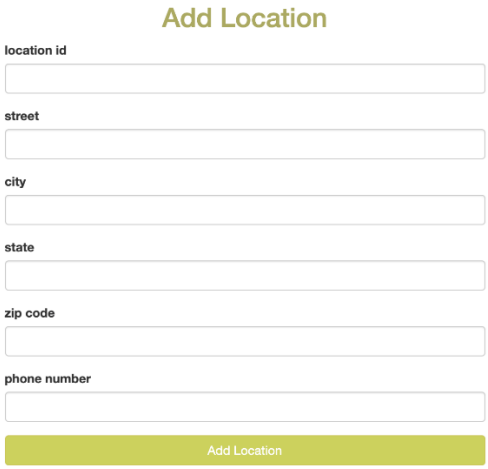





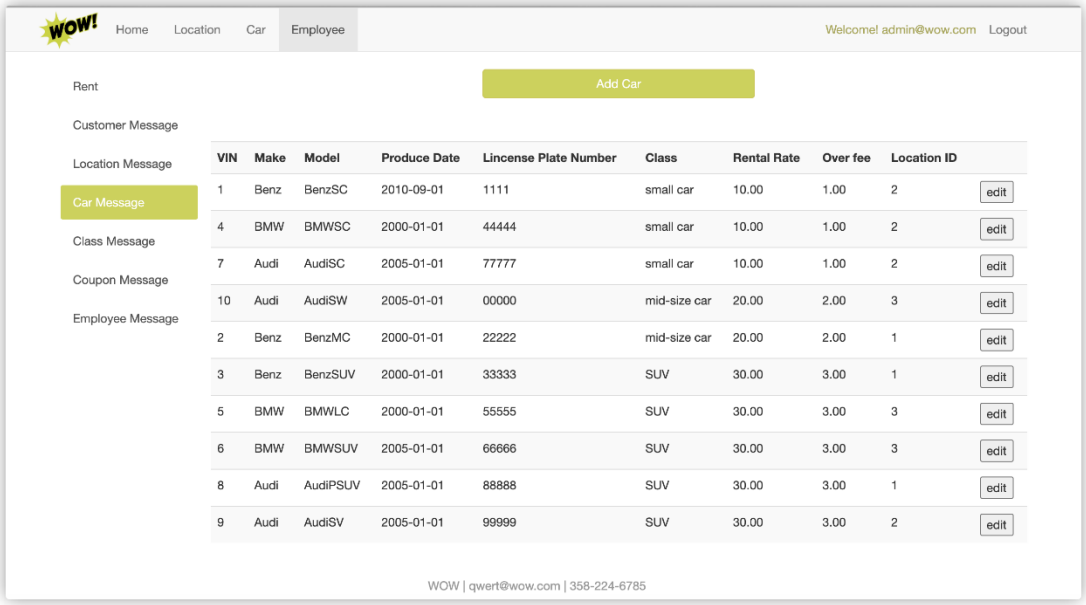
## Location Management

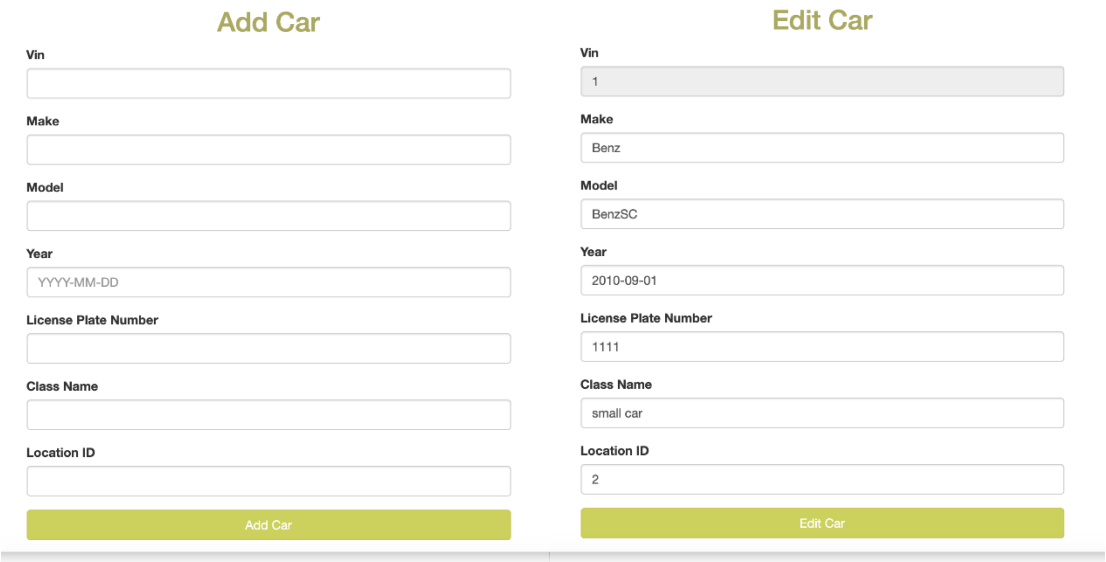


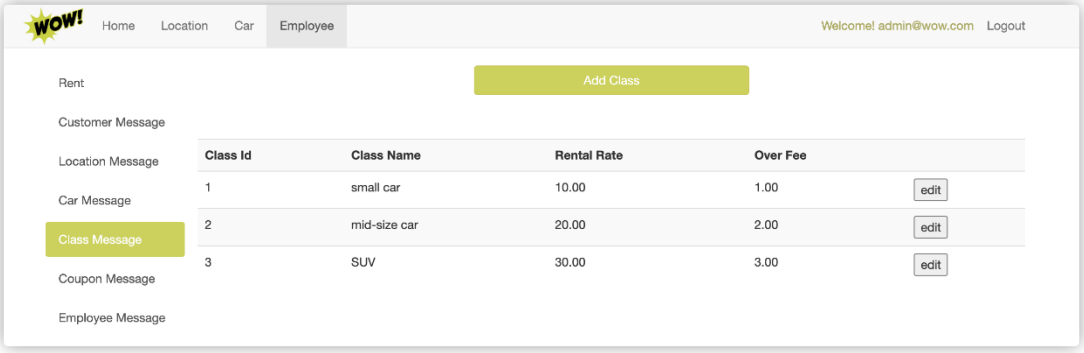



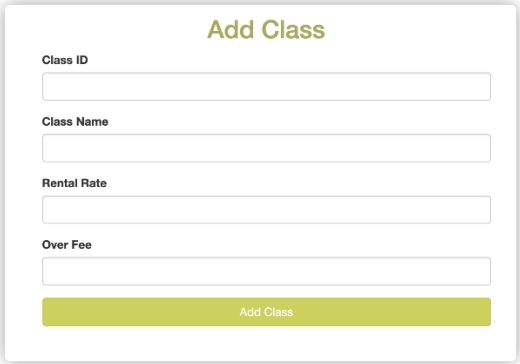

## Vehicle Management

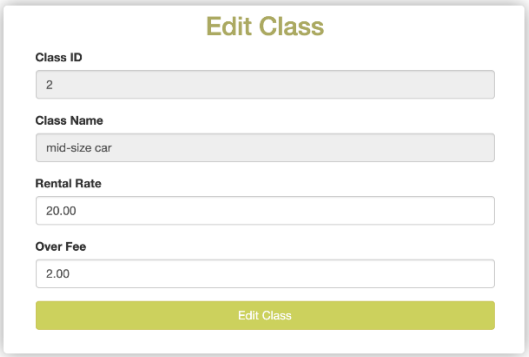




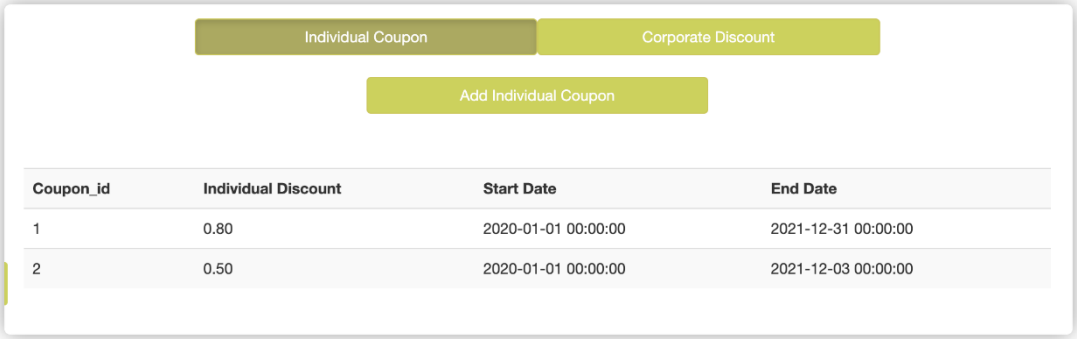

## Class Management

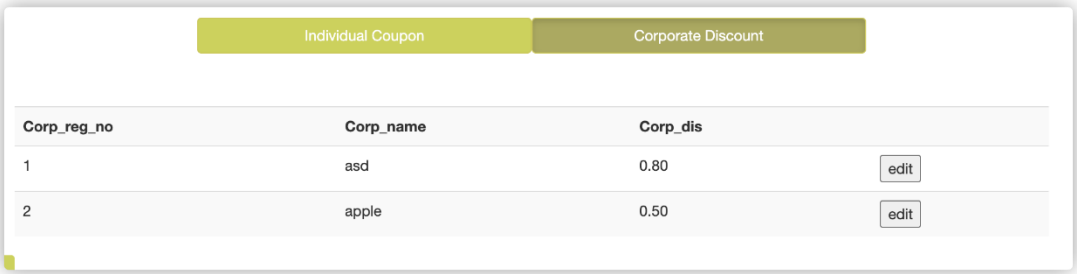


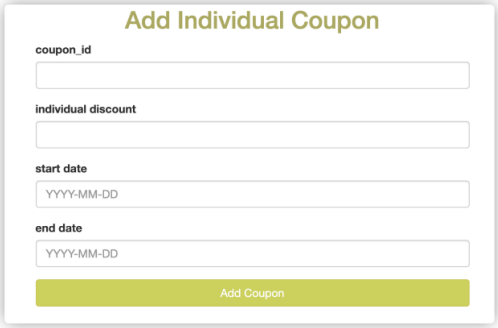



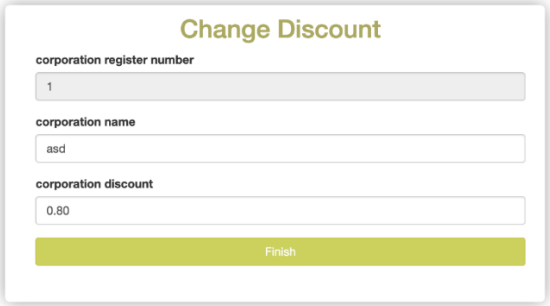




## Coupon Management

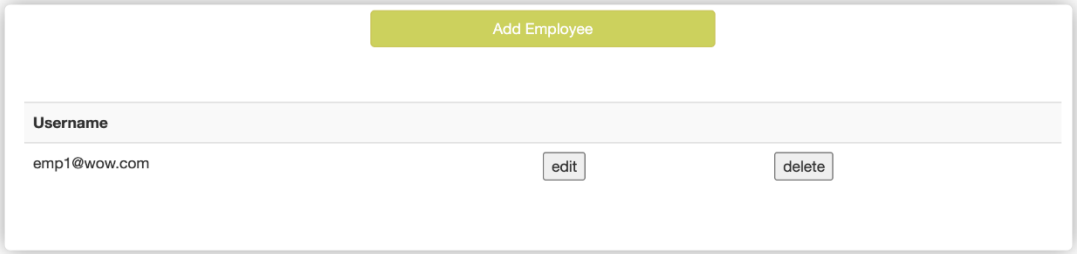


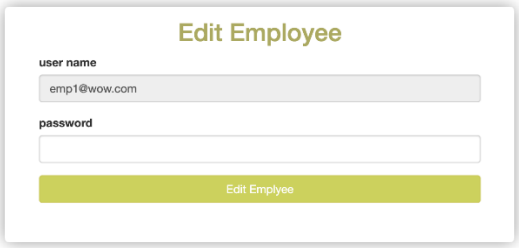



**Employee Management**





## 3.2 Security Feature

### 3.2.1 Cross Site Script attack

In order to make protection against Cross Site Script attack, we take methods as followed.

- When using setcookie(), we set httponly=true. Then, cookies can only be accessed through the HTTP protocol and cannot be accessed through scripting languages like JavaScript.
- Add htmlspecialchars() or htmlentities() to escape characters to each output in the page and user input texts.
- Restrict special characters in input text, and use max data length check to

restrict unvalid long script.

### 3.2.2 SQL injections

In order to make protection against SQL injections, we used the "prepared statement" through php code in each text which user input, to ensure every input word is valid. Furthermore, we may use escaping function to preprocess SQL input parameters.

### 3.2.3 Password Protection

We use the hash and encryption methods to protect the password. We plan to let the super administrators enter a random data as a master key. Then, users' input password will be hash to a 32 digits length data via MD5. Then we will use the standard triple DES algorithm with master key to encrypt the input password data.

### 3.2.4 Transaction

We use Transaction concept to implement atomicity, durability, isolation, and consistency. For instance, when we update an individual customer information, we need to update user_password table, customer table, individual table. All these tables should be updated in once operation, which means that all tables should not be updated once either of them failed.

## 4. Reflections on project

Through working on the project, we learned large amount of knowledge. For instance, we lack specific experience in front-end interface, back-end connection, network communication knowledge such as cookies and cookies, security operation online such as protection against SQL injection, cross site script attack and so on. With the progress going on, we learned related courses and search for the Internet to get specific information to solve the problems we meet. It helps us improve our project.

Through the practice and test base on business rule, we encountered, analyzed, and solved problems which will not appear in the basic design at first. For instance, the time attribute includes year, month, day as well as hour, minute, seconds. If we ignore the hour, minute, and seconds, we will get wrong comparison results. When front end gets values from back end from database, we need to pay attention to the datatype because data transmission or display will have other data types. If we want to do some value comparison, we need change these to float type. We get these experiences only after debugging.

During the specific implementation, we had to make some changes to the structure of the database considering the realization of the function. For example, when designing 'Invoice', we only designed the 'total amount' at first. But when making a specific payment, we need to consider the status of the order, the user's discounted amount, and the remaining amount after each payment, so we added a few more attributes.

The difference between SQL statements in different databases is also a big problem. We used Oracle's Data-Modeler in the design process, however when loading into MySQL, there were some syntax differences, such as the rule of writing trigger.

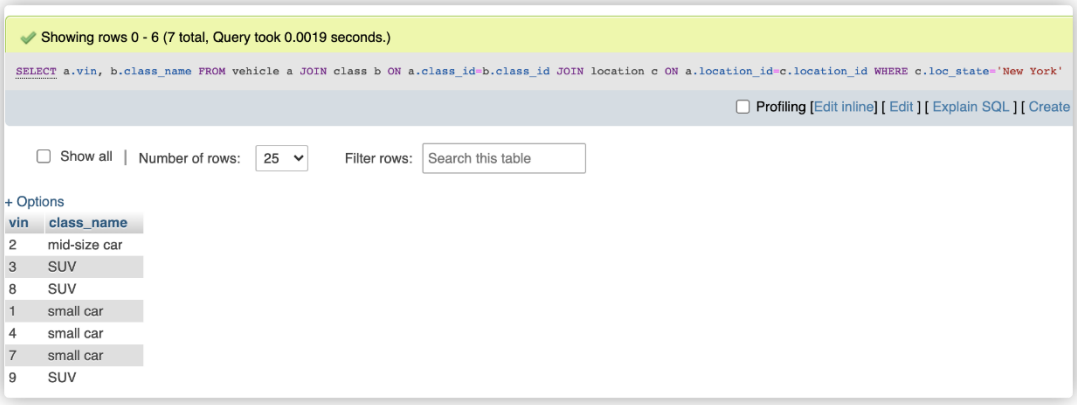
Through learning from class material and Internet, we solved these problems well.

We encounter time management problems at first because of the large load during the final period and lack of related experience. When we two teammates were in trouble, we reviewed the System Development Life Cycle and learned from it. We made the propriate plan and analysis, finished the designing and implementation according to our timeline. We often communicate with each other, therefore can find problems, and solve them quickly. Owning to the version control tool, git helps us cooperate better and have a good development experience.

# 5. Business Analysis with SQLs

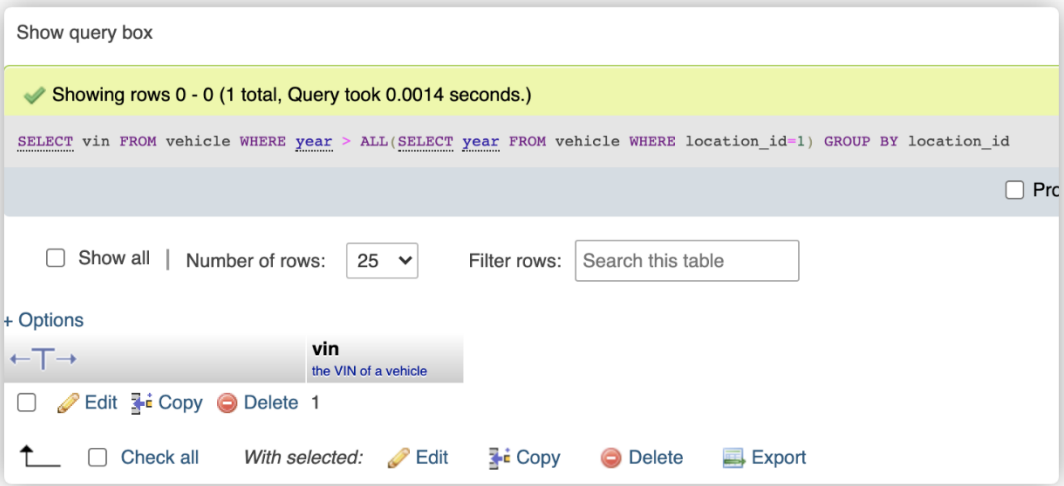
## 5.1 Join

Find all vehicle VIN and CLASS name in all New York Offices.




## 5.2 Sub Query

Find the vehicle's VIN, whose product year is later than all the vehicle's product year in location id 1.



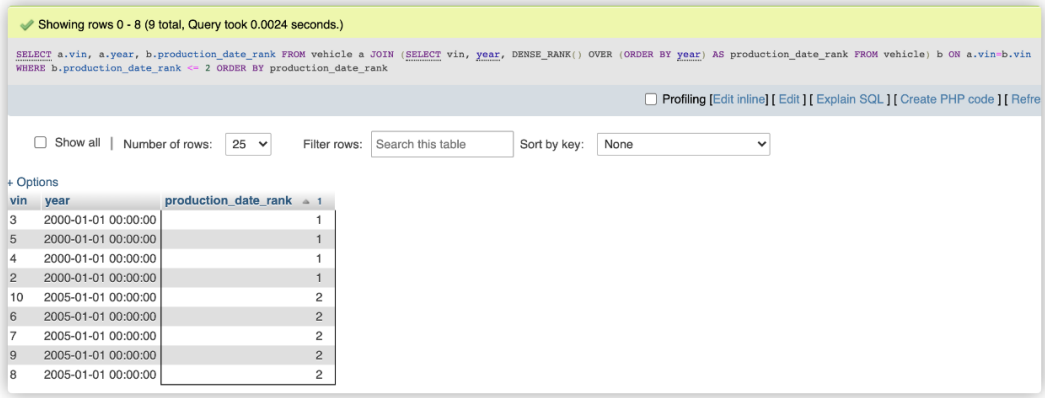
## 5.3 With

Firstly, find all vehicles in New York Offices. Then, find the VIN and year whose product year are later than all vehicles in New York Offices.

There is a product year is 2019, we cannot find more closest date vehicle record.


## 5.4 Top-N queries

Find VIN, year and ranking by product year. Choosing vehicles which have the oldest product year and second oldest product year.




# 6. Extra Feature

In addition to implement basic functions to the project, we practice more and add some extra features. We list some features as followed.