Project Report

ID: N15568403

Net ID: jz3766

Name: Jianfei Zhao

Course: CSGY-6083-B

Date of Submission: 12/19/2020

1. Description

My entire design is based on the business rules made by WOW company and some reasonable assumptions listed below. In project part 1, I designed the database schema that define the relational and logical model based on the business case. In project part 2, I first modify and improve the database model that designed in part 1. Then, I use Python Flask web developing framework with SQLite, a small and self-contained database engine, to design a practical and compact website which implements the business cases made by WOW company.

My website has two interfaces, one for the customers, one for the employees(administrator) in WOW company. Each customer can have two different types of account, individual account and corporate account. For corporate account, user must set their company name and employee ID, and only those companies that have relationship with WOW company can have their employees create the corporate account, since each affiliated company has fixed discount that provided to their employees. User must choose their account type when they register in the website.

For customers that don't have an account, or don't want to register for the moment. They can only go through the website, searching for vehicles and rental office locations. If they want to start a reservation, they need to login first, otherwise the system will redirect to the login page.

For the customer interface, user can search for vehicles and rental office locations, and most importantly, reserve vehicles. User need to choose the pick-up location, return location, pick-up date and return date when they start a new reservation. Then, they should select a vehicle in the pick-up location and enter the start odometer value and the daily odometer limit. After they return the car back to the return location, they need to end the order and enter the end odometer value. So the system could compute the total amount the user need to pay and generate the invoice for that order. Finally, user could pay for the order and system will record all the data and write into the database.

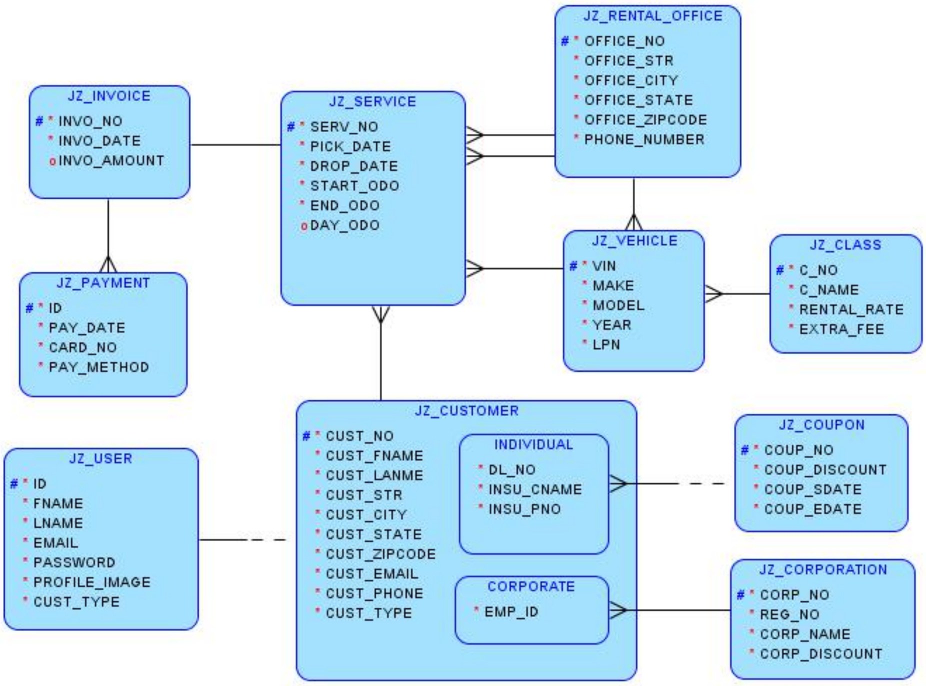
For the employee interface, the administrator could create, update, delete all the information about the vehicles, classes of vehicles, rental offices, affiliated companies, coupons and all the users registered in the website. I assume that all the administrator have email in the format of adminXXX@wow.com and they must login the system using this email.

Assumptions:

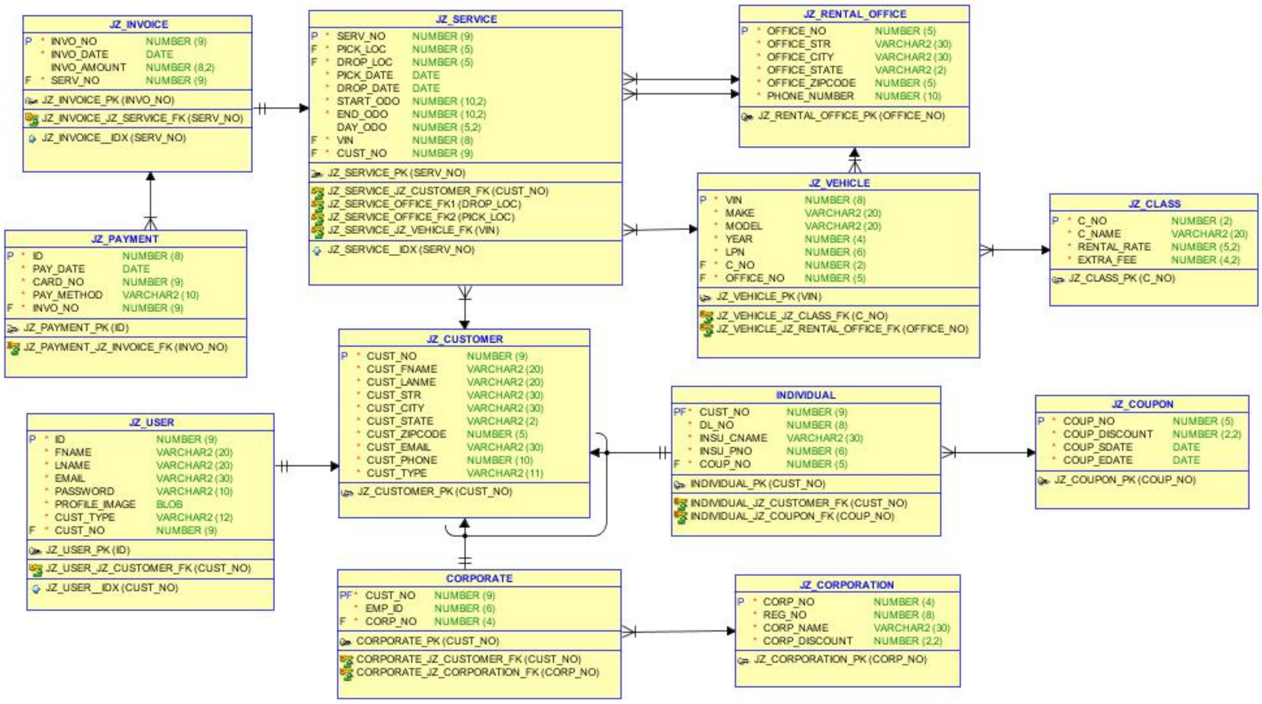
- A corporation may have many employees who use the corporate account with same corporate discount to rent cars.
- Same coupon can belong to multiple individual customers but one individual customer may not have or can only have one discount coupon.
- The WOW company will mail coupon to some customers with the coupon number on it, so when the customer want to use that coupon, they could enter the coupon number in their account page.
- Customer can only pick up or drop off a car at the location of the rental office.
- The Daily Odometer Limit attribute is optional when user start a new order.

- The system administrators know the rental office number of each location, so they can directly entering the number of the rental office when updating the information of the vehicles in that office location.

2. Logical model



3. Relational model



4. Software and Database

For this project, I use Python Flask web developing framework with SQLite as the back-end database.

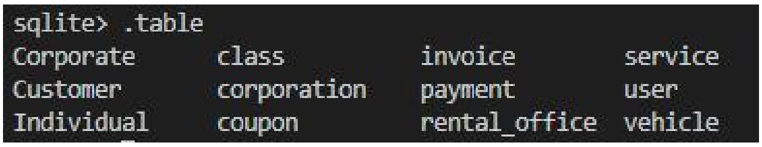
Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications. It began as a simple wrapper around Werkzeug and Jinja and has become one of the most popular Python web application frameworks. I also use SQLAlchemy, an open-source SQL toolkit and object-relational mapper (ORM) for the Python, to build the database schema, insert or update data and write the database query.

SQLite is an in-process library that implements a self-contained, zero-configuration, transactional SQL database engine. The code for SQLite is in the public domain and is thus free for use for any purpose, commercial or private. SQLite is the most widely deployed database in the world with more applications than we can count, including several high-profile projects.

SQLite is an embedded SQL database engine. Unlike most other SQL databases, SQLite does not have a separate server process. SQLite reads and writes directly to ordinary disk files. A complete SQL database with multiple tables, indices, triggers, and views, is contained in a single disk file.

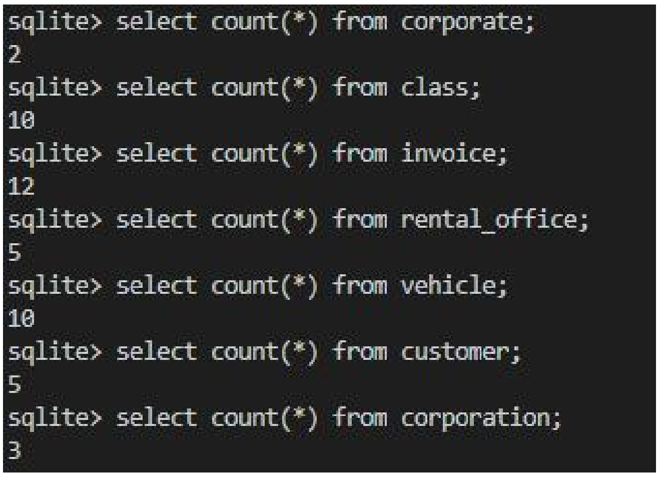
5. Tables and data

The list of tables in my database:

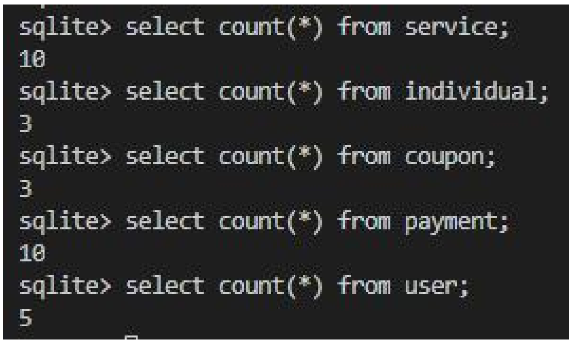


```
sqlite> .table
Corporate      class          invoice        service
Customer       corporation    payment        user
Individual     coupon         rental_office  vehicle
```

Total number of records in each table:



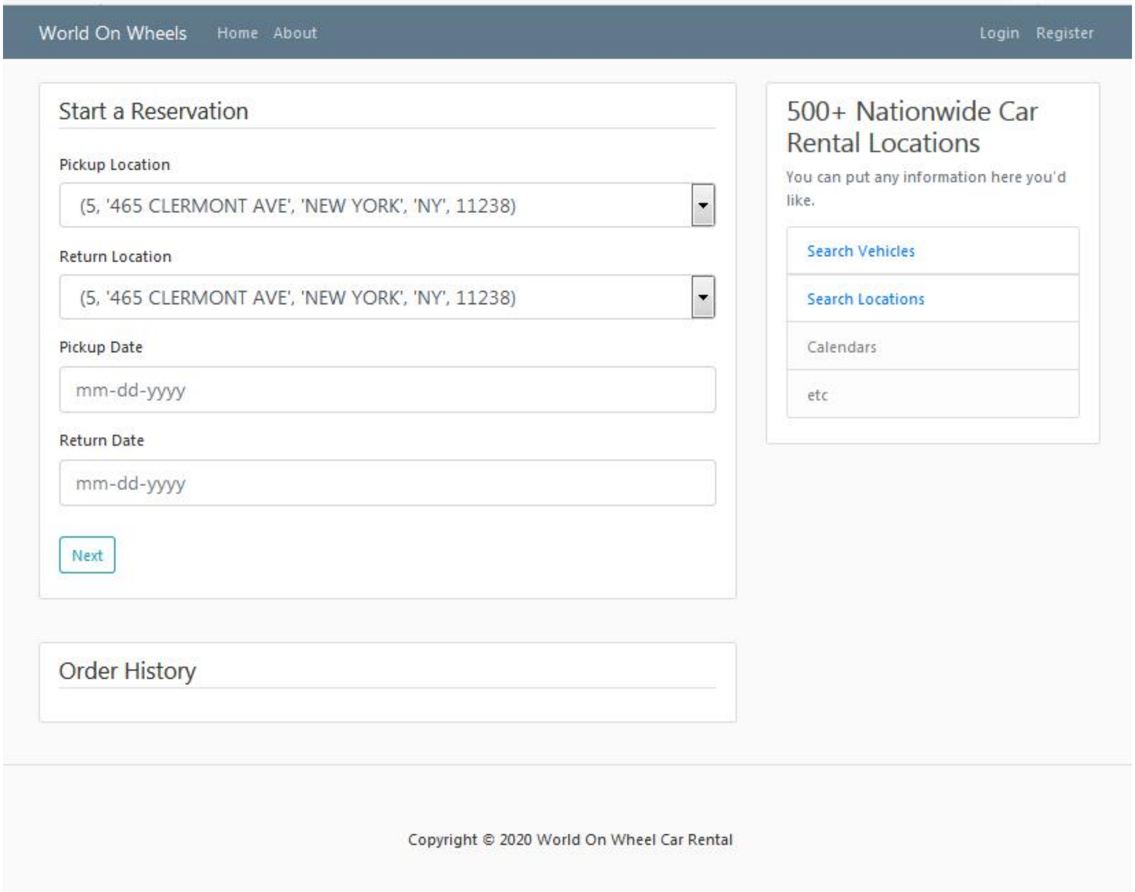
```
sqlite> select count(*) from corporate;
2
sqlite> select count(*) from class;
10
sqlite> select count(*) from invoice;
12
sqlite> select count(*) from rental_office;
5
sqlite> select count(*) from vehicle;
10
sqlite> select count(*) from customer;
5
sqlite> select count(*) from corporation;
3
```



```
sqlite> select count(*) from service;
10
sqlite> select count(*) from individual;
3
sqlite> select count(*) from coupon;
3
sqlite> select count(*) from payment;
10
sqlite> select count(*) from user;
5
```

6. Screenshots of My Web Application

1) Home page without login



Start a Reservation

Pickup Location

(5, '465 CLERMONT AVE', 'NEW YORK', 'NY', 11238)

Return Location

(5, '465 CLERMONT AVE', 'NEW YORK', 'NY', 11238)

Pickup Date

mm-dd-yyyy

Return Date

mm-dd-yyyy

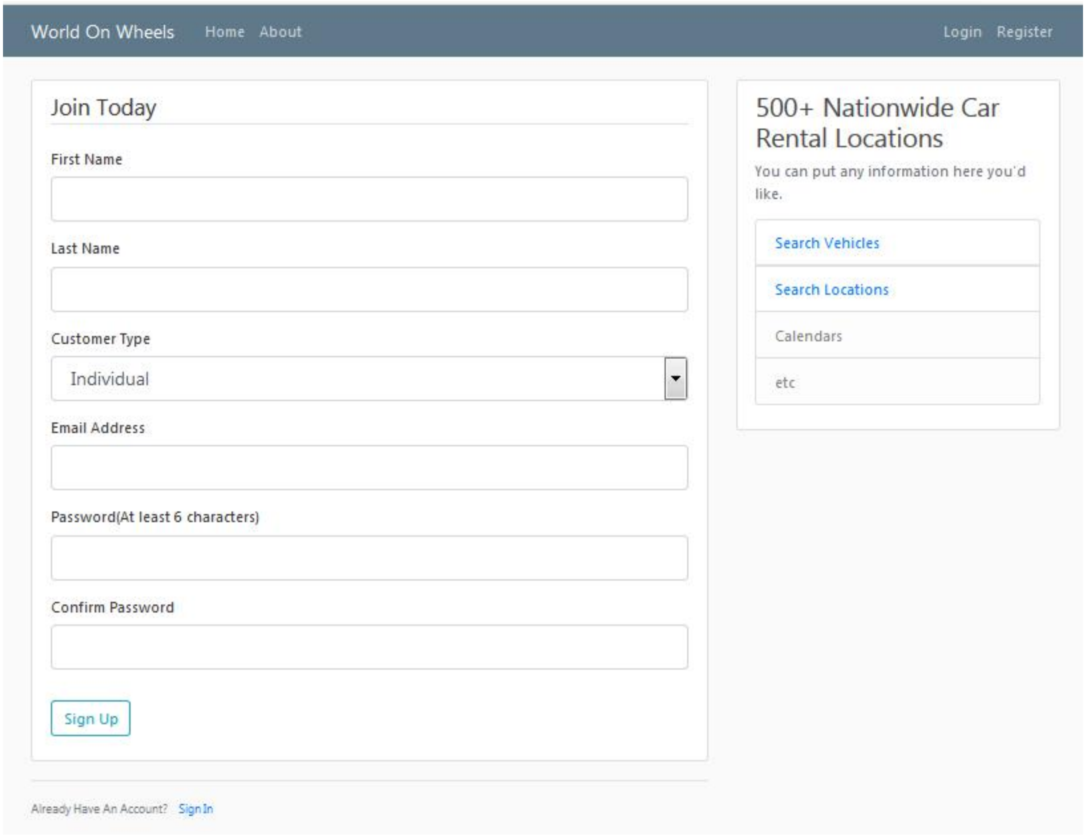
Order History

500+ Nationwide Car Rental Locations

You can put any information here you'd like.

Copyright © 2020 World On Wheel Car Rental

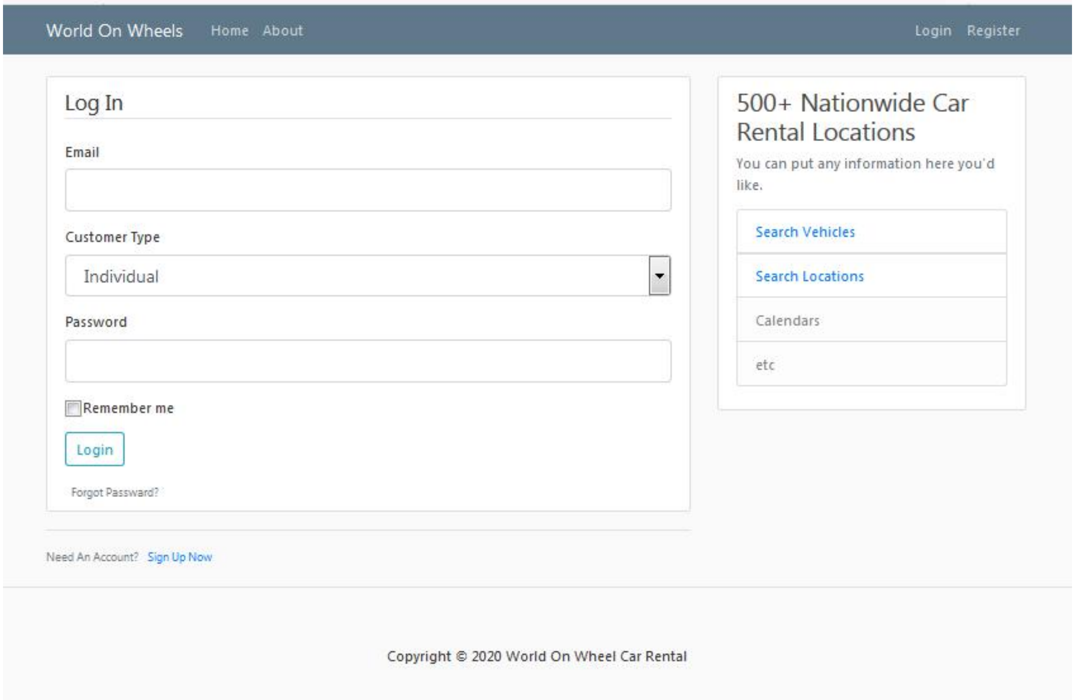
2) Register page



You can put any information here you'd like.

Already Have An Account? Sign In

3) Login page

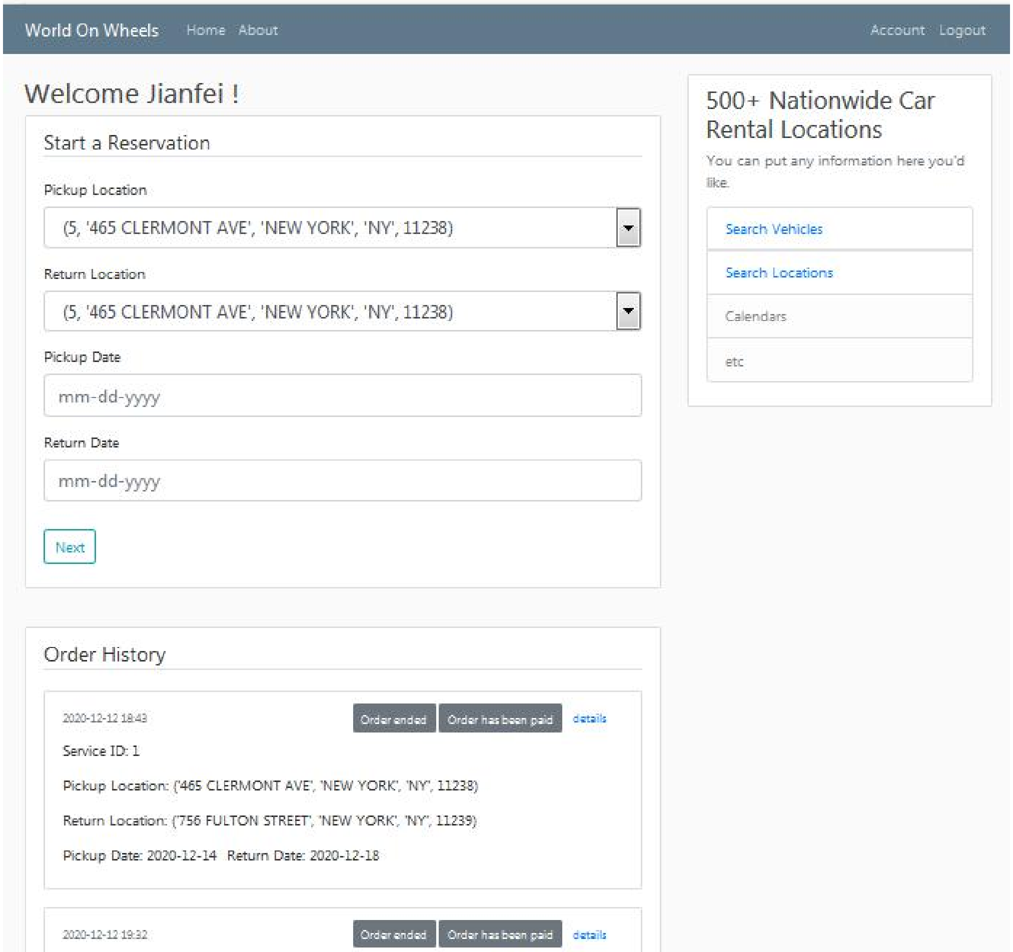


You can put any information here you'd like.

Need An Account? Sign Up Now

Copyright © 2020 World On Wheel Car Rental

4) Home page after login



Welcome Jianfei !

Start a Reservation

Pickup Location: (5, '465 CLERMONT AVE', 'NEW YORK', 'NY', 11238)

Return Location: (5, '465 CLERMONT AVE', 'NEW YORK', 'NY', 11238)

Pickup Date: mm-dd-yyyy

Return Date: mm-dd-yyyy

500+ Nationwide Car Rental Locations

You can put any information here you'd like.

Order History

2020-12-12 18:43

Service ID: 1

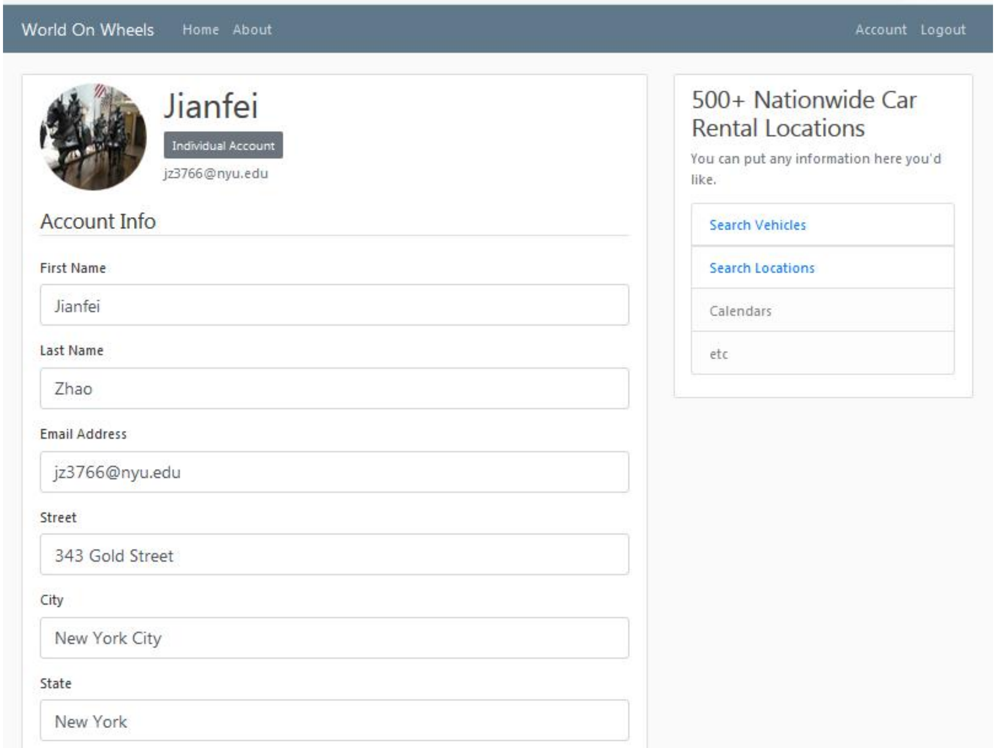
Pickup Location: ('465 CLERMONT AVE', 'NEW YORK', 'NY', 11238)

Return Location: ('756 FULTON STREET', 'NEW YORK', 'NY', 11239)

Pickup Date: 2020-12-14 Return Date: 2020-12-18

2020-12-12 19:32

5) Account page for individual customer



Jianfei

jz3766@nyu.edu

Account Info

First Name: Jianfei

Last Name: Zhao

Email Address: jz3766@nyu.edu

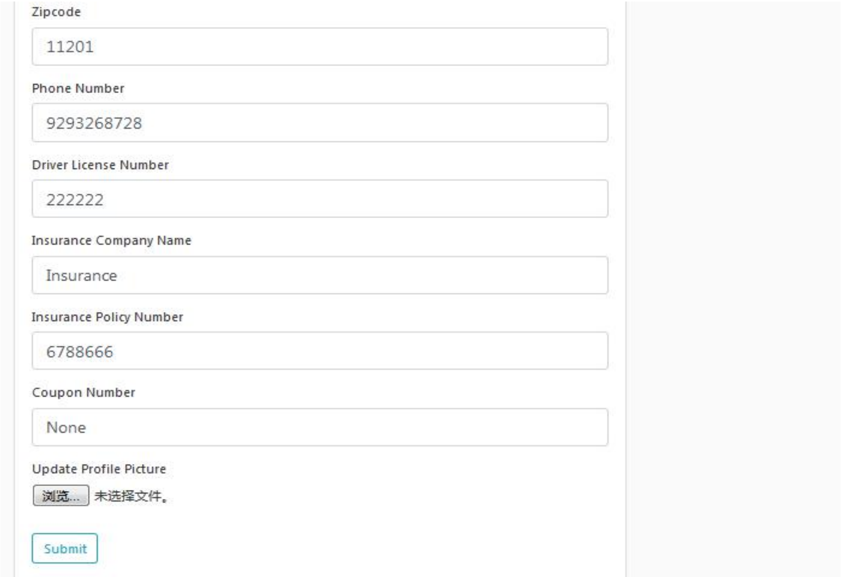
Street: 343 Gold Street

City: New York City

State: New York

500+ Nationwide Car Rental Locations

You can put any information here you'd like.



Zipcode

11201

Phone Number

9293268728

Driver License Number

222222

Insurance Company Name

Insurance

Insurance Policy Number

6788666

Coupon Number

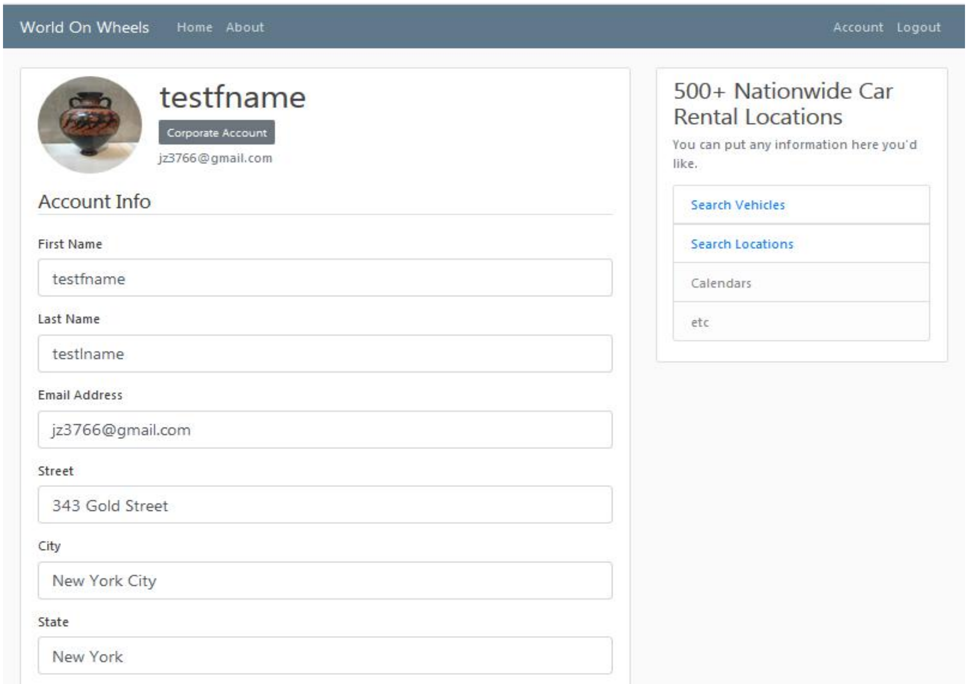
None

Update Profile Picture

浏览... 未选择文件。

Submit

6) Account page for corporate customer



testfname

Corporate Account

jz3766@gmail.com

Account Info

First Name

testfname

Last Name

testlname

Email Address

jz3766@gmail.com

Street

343 Gold Street

City

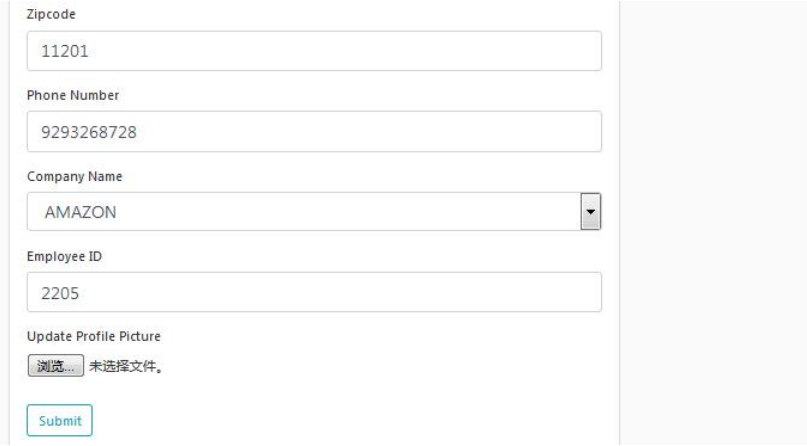
New York City

State

New York

500+ Nationwide Car Rental Locations

You can put any information here you'd like.



Zipcode

11201

Phone Number

9293268728

Company Name

AMAZON

Employee ID

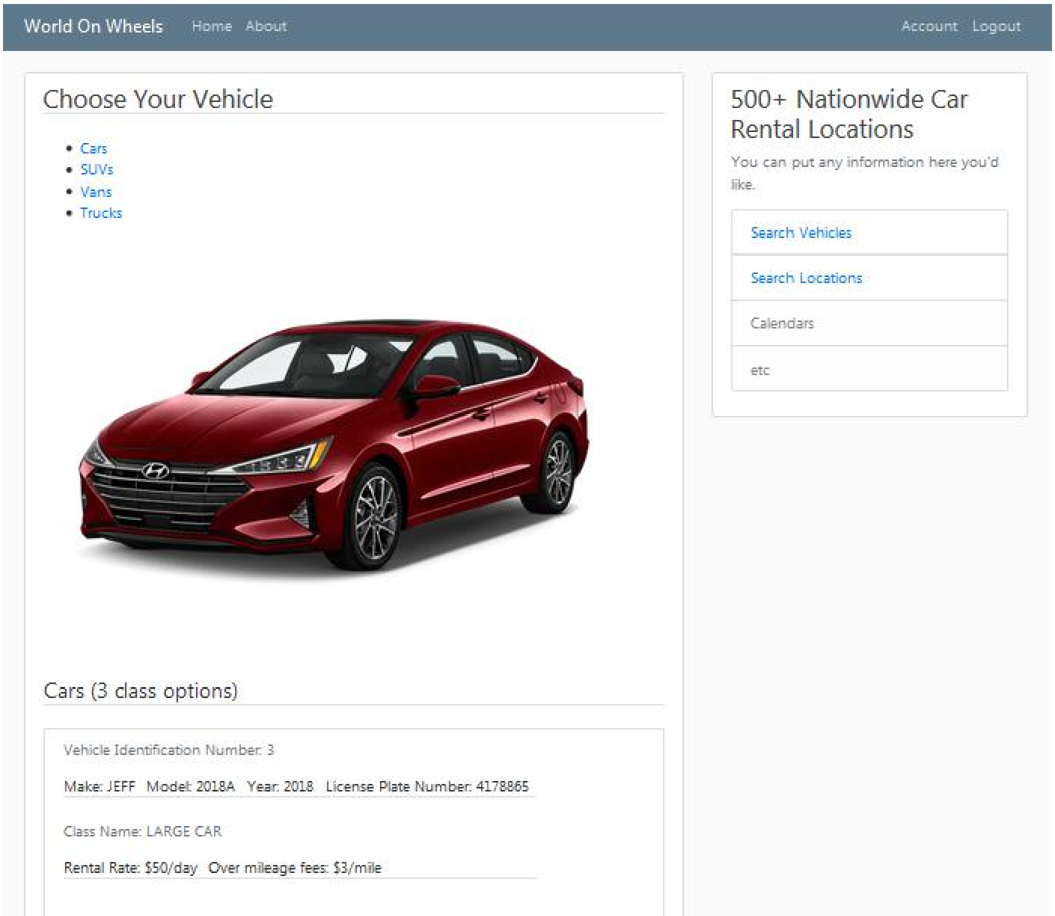
2205

Update Profile Picture

浏览... 未选择文件。

Submit

7) Search vehicles



Choose Your Vehicle

- Cars
- SUVs
- Vans
- Trucks

Cars (3 class options)

Vehicle Identification Number: 3

Make: JEFF Model: 2018A Year: 2018 License Plate Number: 4178865

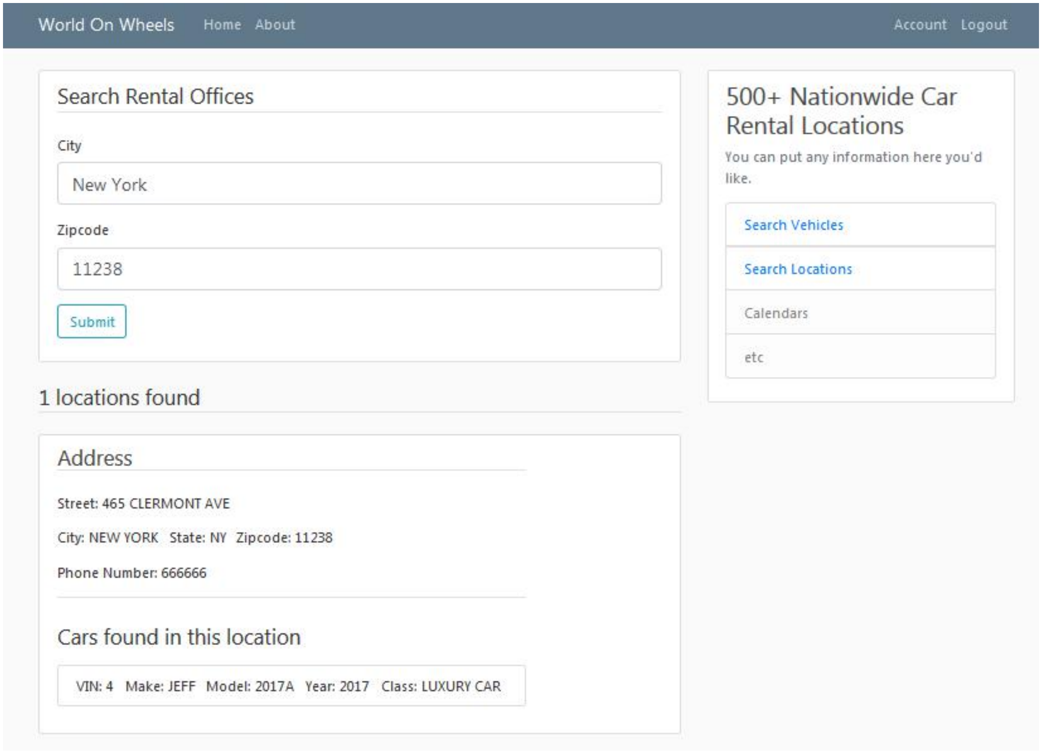
Class Name: LARGE CAR

Rental Rate: $50/day Over mileage fees: $3/mile

500+ Nationwide Car Rental Locations

You can put any information here you'd like.

8) Search locations



Search Rental Offices

City

New York

Zipcode

11238

1 locations found

Address

Street: 465 CLERMONT AVE

City: NEW YORK State: NY Zipcode: 11238

Phone Number: 666666

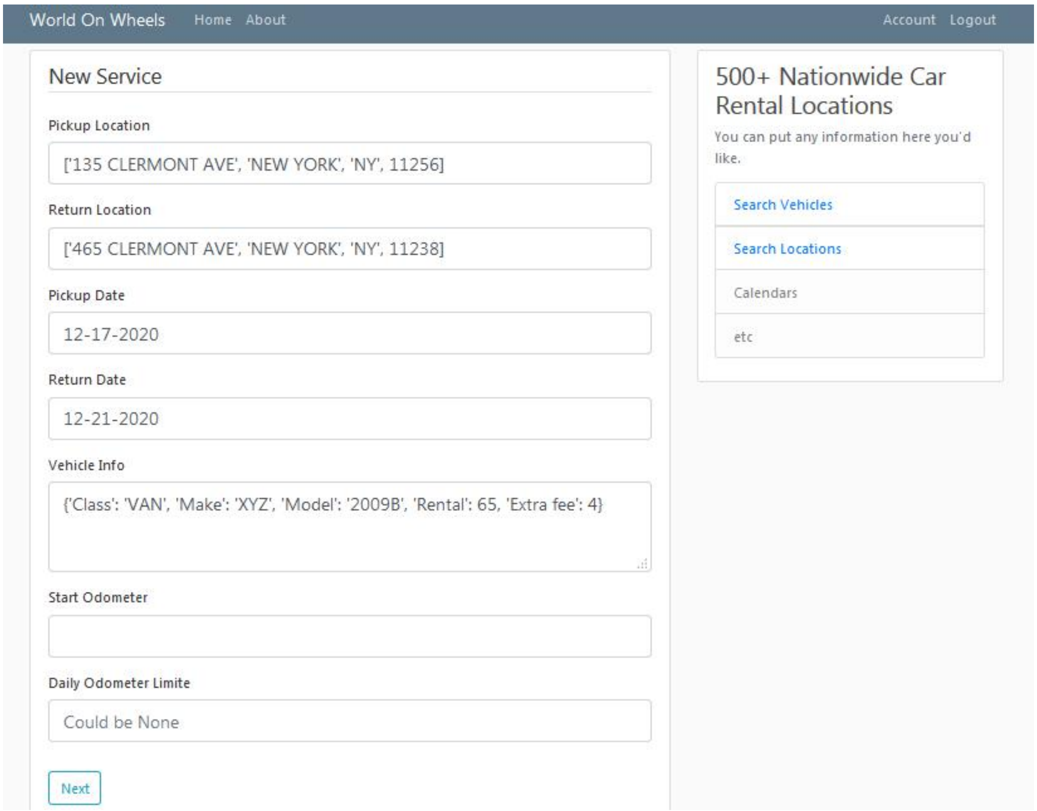
Cars found in this location

VIN: 4 Make: JEFF Model: 2017A Year: 2017 Class: LUXURY CAR

500+ Nationwide Car Rental Locations

You can put any information here you'd like.

9) New service page after choosing vehicles and locations



New Service

Pickup Location: ['135 CLERMONT AVE', 'NEW YORK', 'NY', 11256]

Return Location: ['465 CLERMONT AVE', 'NEW YORK', 'NY', 11238]

Pickup Date: 12-17-2020

Return Date: 12-21-2020

Vehicle Info: {'Class': 'VAN', 'Make': 'XYZ', 'Model': '2009B', 'Rental': 65, 'Extra fee': 4}

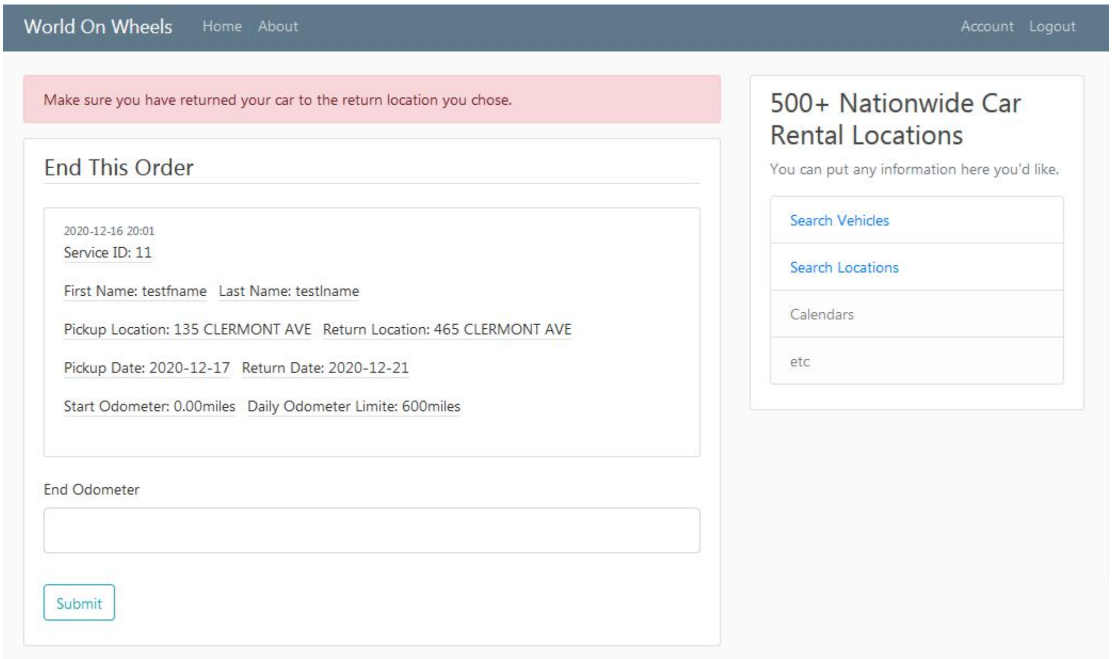
Start Odometer:

Daily Odometer Limite: Could be None

500+ Nationwide Car Rental Locations

You can put any information here you'd like.

10) End order session



Make sure you have returned your car to the return location you chose.

End This Order

2020-12-16 20:01

Service ID: 11

First Name: testfname Last Name: testlname

Pickup Location: 135 CLERMONT AVE Return Location: 465 CLERMONT AVE

Pickup Date: 2020-12-17 Return Date: 2020-12-21

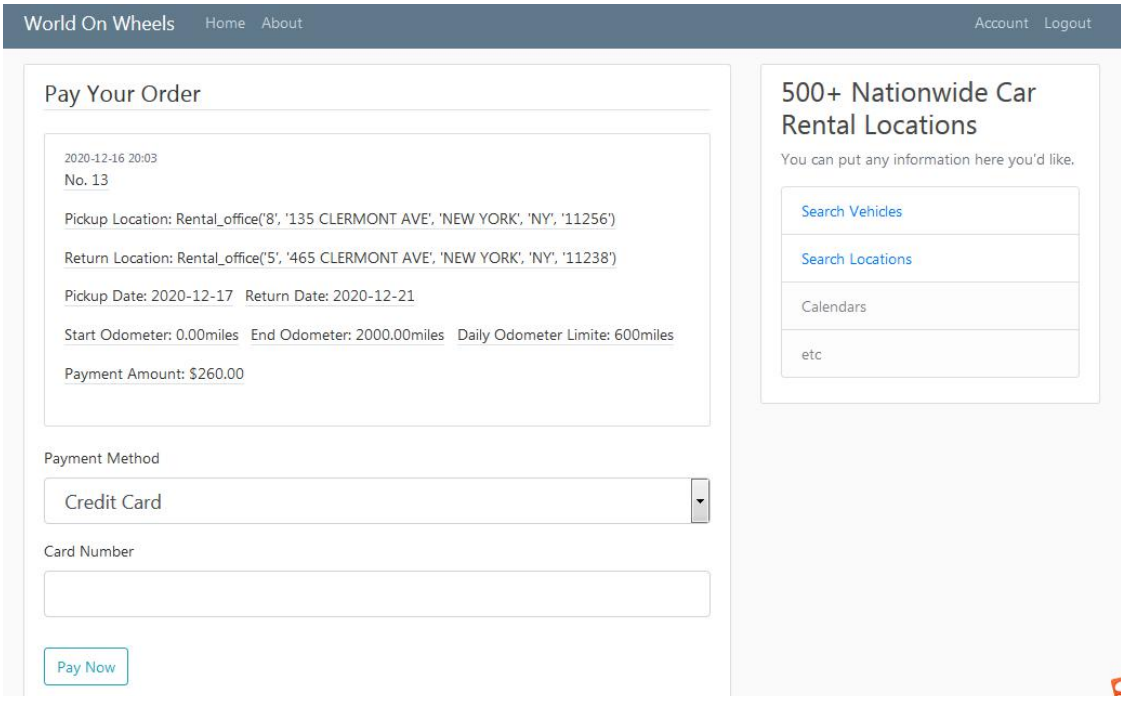
Start Odometer: 0.00miles Daily Odometer Limite: 600miles

End Odometer:

500+ Nationwide Car Rental Locations

You can put any information here you'd like.

11) Payment session



Pay Your Order

2020-12-16 20:03

No. 13

Pickup Location: Rental_office('8', '135 CLERMONT AVE', 'NEW YORK', 'NY', '11256')

Return Location: Rental_office('5', '465 CLERMONT AVE', 'NEW YORK', 'NY', '11238')

Pickup Date: 2020-12-17 Return Date: 2020-12-21

Start Odometer: 0.00miles End Odometer: 2000.00miles Daily Odometer Limite: 600miles

Payment Amount: $260.00

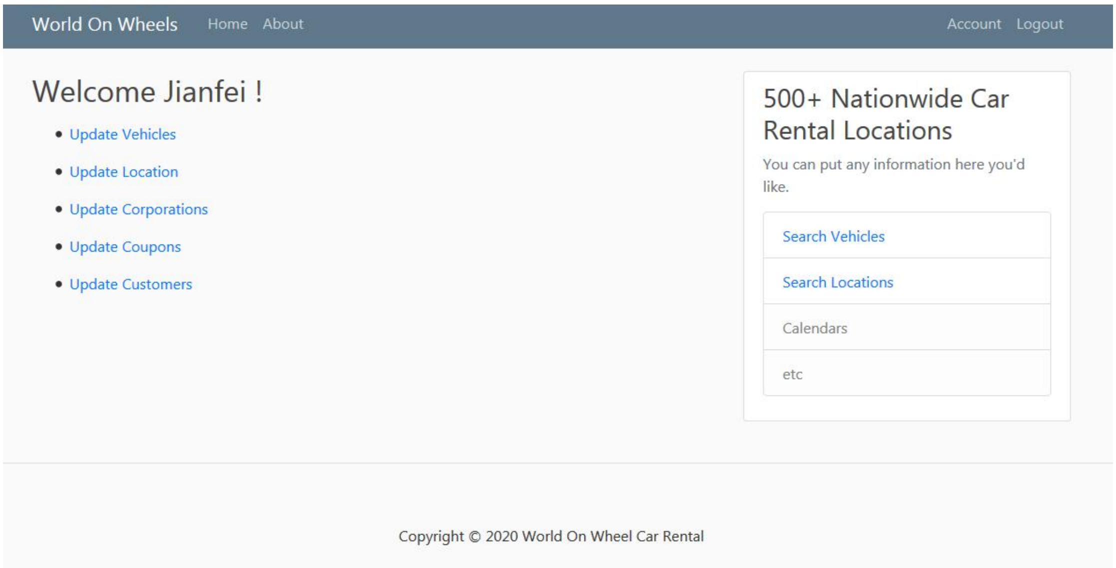
Payment Method: Credit Card

Card Number:

500+ Nationwide Car Rental Locations

You can put any information here you'd like.

12) Employee interface home page



Welcome Jianfei !

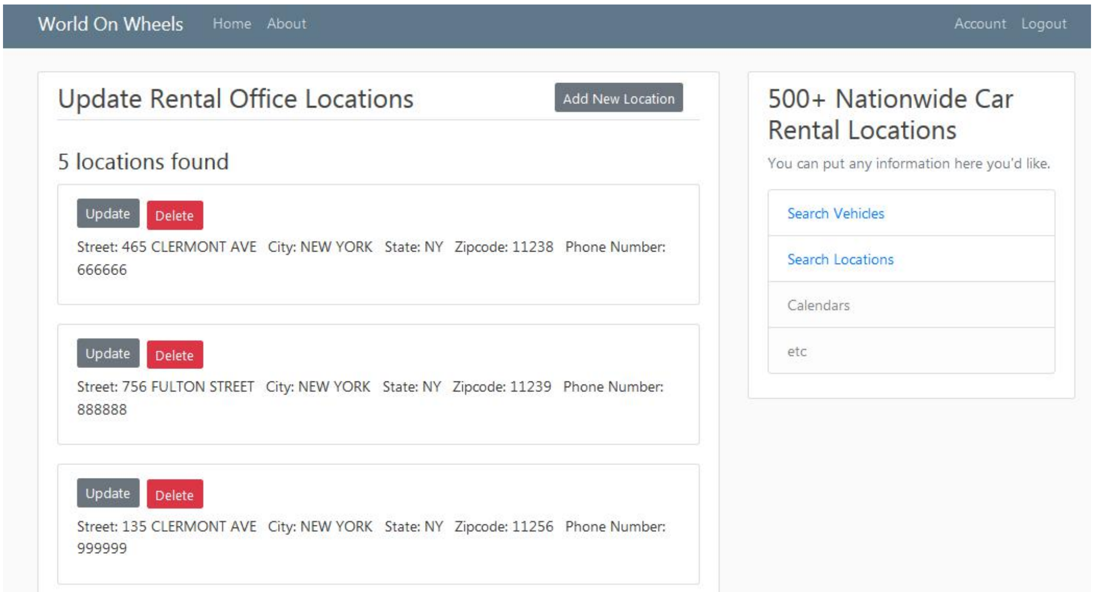
- Update Vehicles
- Update Location
- Update Corporations
- Update Coupons
- Update Customers

500+ Nationwide Car Rental Locations

You can put any information here you'd like.

Copyright © 2020 World On Wheel Car Rental

13) Update locations session



Update Rental Office Locations

5 locations found

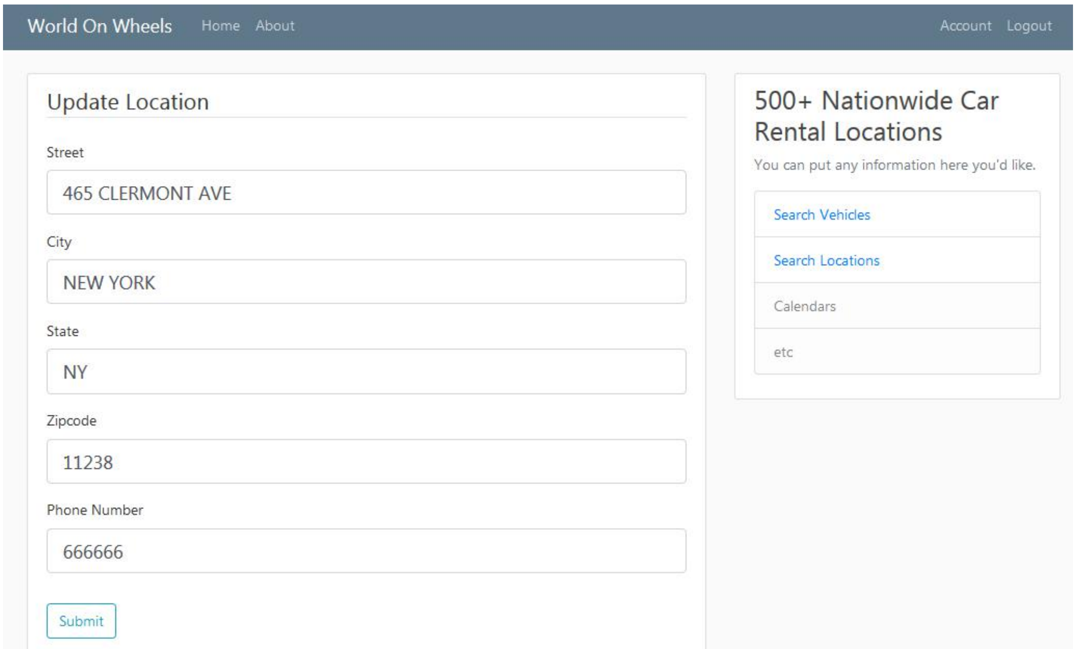
Street: 465 CLERMONT AVE City: NEW YORK State: NY Zipcode: 11238 Phone Number: 666666

Street: 756 FULTON STREET City: NEW YORK State: NY Zipcode: 11239 Phone Number: 888888

Street: 135 CLERMONT AVE City: NEW YORK State: NY Zipcode: 11256 Phone Number: 999999

500+ Nationwide Car Rental Locations

You can put any information here you'd like.



Update Location

Street: 465 CLERMONT AVE

City: NEW YORK

State: NY

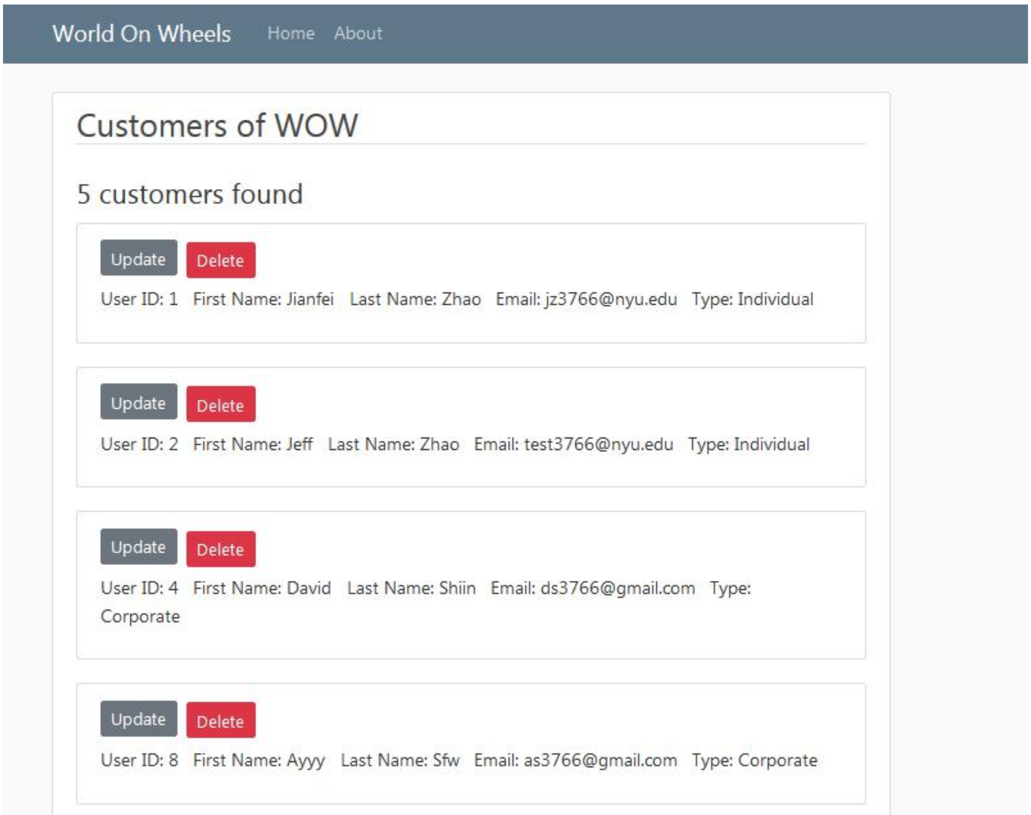
Zipcode: 11238

Phone Number: 666666

500+ Nationwide Car Rental Locations

You can put any information here you'd like.

14) Update customers session



Customers of WOW

5 customers found

User ID: 1 First Name: Jianfei Last Name: Zhao Email: jz3766@nyu.edu Type: Individual

User ID: 2 First Name: Jeff Last Name: Zhao Email: test3766@nyu.edu Type: Individual

User ID: 4 First Name: David Last Name: Shiin Email: ds3766@gmail.com Type: Corporate

User ID: 8 First Name: Ayyy Last Name: Sfw Email: as3766@gmail.com Type: Corporate

The other sessions for employee interface such as update vehicles, corporations, coupons, classes of vehicles are similar to the two sessions above.

7. Security features

1) Password protection

I use a library in Python called Bcrypt to generate a hexadecimal hashed password in the register route when user enter their password, and use the same library in login route to decrypt the password that user enter in. The system only stores the encrypt(hashed) password for every user, so nobody else can know the real password of a user.

2) Commit() & rollback()

I use try-except logic in Python for updating data in database, so each time when we update the record, if there are some conflicts or interrupts happens, the session will not commit our changes and rollback the operations.

3) XSS Attack prevention

Jinja is Flask's default template engine and it provides a protected framework for automation of testing programs, whose behaviour is unknown and must be investigated. Jinja also has a powerful automatic HTML Escaping, which helps preventing Cross-site Scripting (XSS Attack). Jinja is also easy to debug.

8. Reflections about project

Doing this project on my own is quit challenging for me since this is my first time to develop a web application, but it also makes me familiar with the whole process, from back-end to front-end. Each time I encountered a problem, either for designing the web page, or for modifying bugs in my code, I need to search on the Internet by myself and I really learned a lot through this process, and it also makes me familiar with Python Flask and writing HTML codes, as well as how to transfer data between database and website. This project is really a practical exercise to examine what we have learned in class.

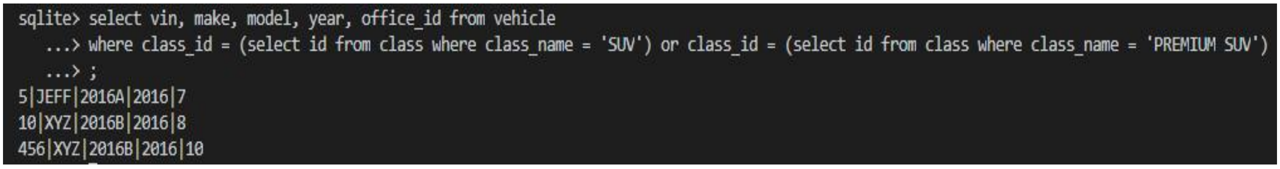
The most important lesson I get from this project is to design the database model properly at the beginning, and spend more time to check the logic and relations between each tables. In this case when we start to design the web page, we will not going to modify our design for database many times.

Finally, there is still much space for me to improve my web application, and I think if there are more time, I will not need to simplify some functions in the web, and to add those functions is not difficult. However, the total time management for me is not bad at least for my thought, and this project is very helpful for my job-hunting in the next year.

9. Business analysis

1) Sub-query

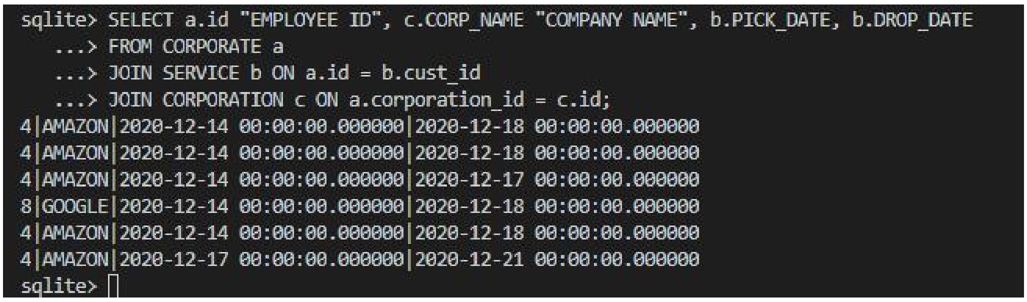
For all the SUVs, select vehicle information and the number of office they belongs to.



```
sqlite> select vin, make, model, year, office_id from vehicle
   ...> where class_id = (select id from class where class_name = 'SUV') or class_id = (select id from class where class_name = 'PREMIUM SUV')
   ...> ;
5|JEFF|2016A|2016|7
10|XYZ|2016B|2016|8
456|XYZ|2016B|2016|10
```

2) Joins

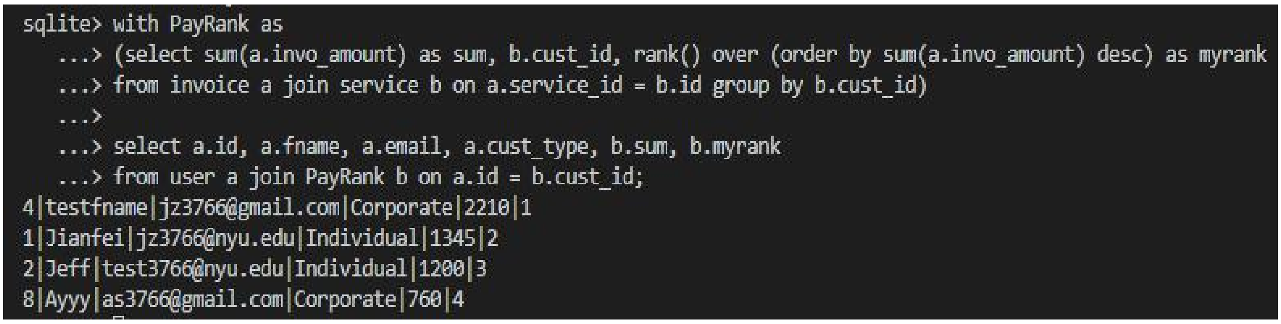
For each rental service with corporate account, show employee ID, corporation name, pick up date and drop off date correspond to those service.



```
sqlite> SELECT a.id "EMPLOYEE ID", c.CORP_NAME "COMPANY NAME", b.PICK_DATE, b.DROP_DATE
   ...> FROM CORPORATE a
   ...> JOIN SERVICE b ON a.id = b.cust_id
   ...> JOIN CORPORATION c ON a.corporation_id = c.id;
4|AMAZON|2020-12-14 00:00:00.000000|2020-12-18 00:00:00.000000
4|AMAZON|2020-12-14 00:00:00.000000|2020-12-18 00:00:00.000000
4|AMAZON|2020-12-14 00:00:00.000000|2020-12-17 00:00:00.000000
8|GOOGLE|2020-12-14 00:00:00.000000|2020-12-18 00:00:00.000000
4|AMAZON|2020-12-14 00:00:00.000000|2020-12-18 00:00:00.000000
4|AMAZON|2020-12-17 00:00:00.000000|2020-12-21 00:00:00.000000
sqlite> 
```

3) RANK function for TOP-N queries

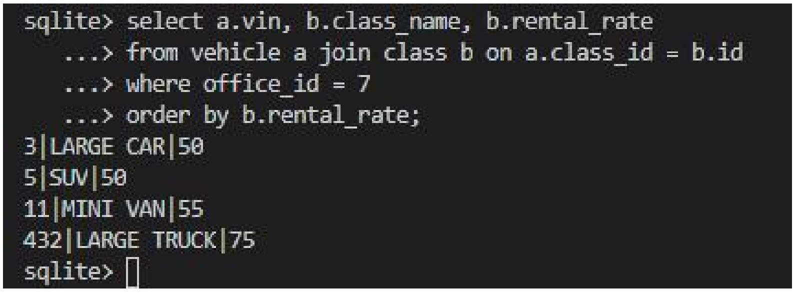
Select user ID, first name, email, customer type, sum of total payment amount, and the rank of total payment amount through all the users.



```
sqlite> with PayRank as
   ...> (select sum(a.invo_amount) as sum, b.cust_id, rank() over (order by sum(a.invo_amount) desc) as myrank
   ...> from invoice a join service b on a.service_id = b.id group by b.cust_id)
   ...>
   ...> select a.id, a.fname, a.email, a.cust_type, b.sum, b.myrank
   ...> from user a join PayRank b on a.id = b.cust_id;
4|testfname|jz3766@gmail.com|Corporate|2210|1
1|Jianfei|jz3766@nyu.edu|Individual|1345|2
2|Jeff|test3766@nyu.edu|Individual|1200|3
8|Ayyy|as3766@gmail.com|Corporate|760|4
```

4) Join

For all the vehicles in office location 7, select VIN, class that the vehicle belongs to, rental rate each day, and order by rental rate.



```
sqlite> select a.vin, b.class_name, b.rental_rate
   ...> from vehicle a join class b on a.class_id = b.id
   ...> where office_id = 7
   ...> order by b.rental_rate;
3|LARGE CAR|50
5|SUV|50
11|MINI VAN|55
432|LARGE TRUCK|75
sqlite> 
```