EECS3311 Final Project Report

Student name: Yixi Zhao

Student Number: 214936298

EECS Account: frankzyx

Student email: frankzyx@my.yorku.ca

A. Design Patter
   1. Visitor Pattern: I applied visitor patter for customer, officer and admin. Because this part is for three users shared the same data structure. Because visitor pattern used for concrete Object structure, and rarely changing.
   2. Iterator pattern: I used ArrayList for saveing generic type data.
   3. Singleton pattern: I applied singleton pattern for customer officer and admin because this will help saving memory and avoiding bugs.
B. Instruction using
   1. Changing absolute Path: I used absolute path for csv file. Users need to change absolute path before running the application.

      customer path in maintainCustomer and maintainOfficer

```
public class maintainCustomer implements maintain {


    private ArrayList<customer> customerDB = new ArrayList<customer>();
    private final String customerPath = "D:\\EECS3311\\Winter\\Project\\customer.csv";
```

```
public class maintainOfficer implements maintain{
    private ArrayList<customer> customerDB = new ArrayList<customer>();
    private ArrayList<officer> officerDB = new ArrayList<officer>();
    private final String officerPath = "D:\\EECS3311\\Winter\\Project\\officer.csv";
    private final String customerPath = "D:\\EECS3311\\Winter\\Project\\customer.csv";
    private officer currentOfficer = new officer();
```

      officer path in maintainOfficer

```
public class maintainOfficer implements maintain{
    private ArrayList<customer> customerDB = new ArrayList<customer>();
    private ArrayList<officer> officerDB = new ArrayList<officer>();
    private final String officerPath = "D:\\EECS3311\\Winter\\Project\\officer.csv";
    private final String customerPath = "D:\\EECS3311\\Winter\\Project\\customer.csv";
    private officer currentOfficer = new officer();
```

      order path in maintainOrder

```
public class maintainOrder {
    private final String orderPath = "D:\\EECS3311\\Winter\\Project\\bookinginfo.csv";
    private maintainPlot plot_manage;
```

parkingPlot path in maintainOrder

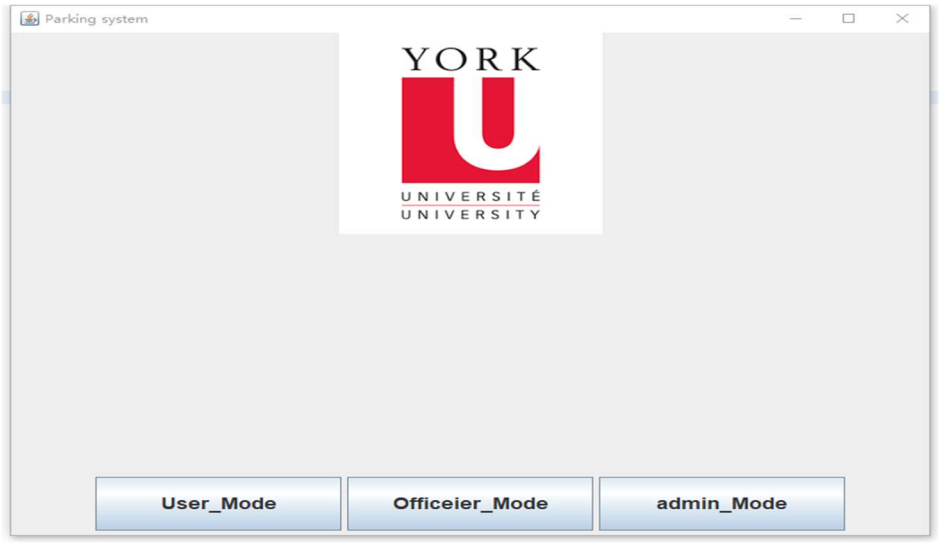
```
public class maintainPlot {
    private ArrayList<parkingPlot> parking = new ArrayList<parkingPlot>();
    private final String path = "D:\\EECS3311\\Winter\\Project\\parkingPlot.csv";

    public maintainPlot() throws Exception {
```

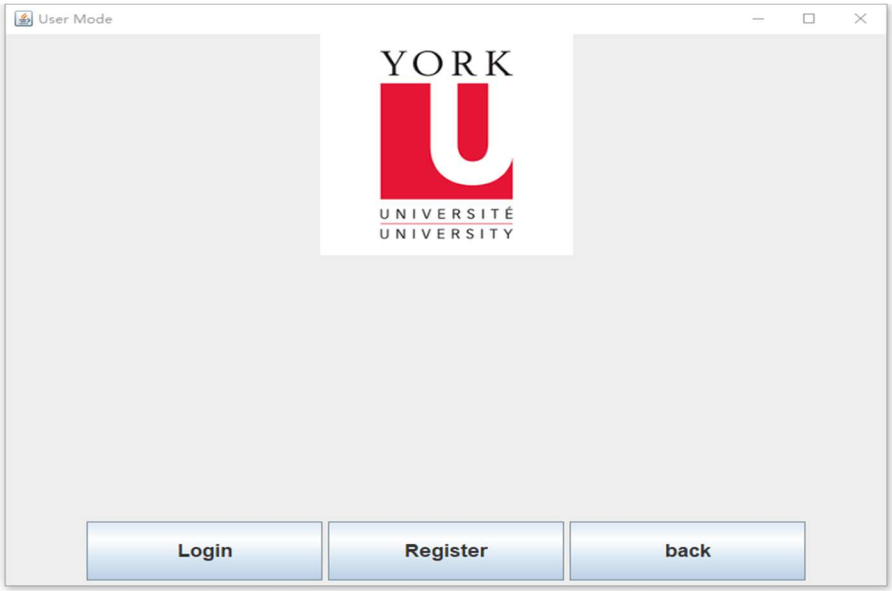
2. To pay for orders, cardNum must be 16digits, or it won't be valid.
3. Registering a officer, system will save it info both in customer and officer database
4. Admin can only be added from backend.
5. User run application form mainView.java
6. Please check the CSV Jar file, because I add it from Java build path as external jar file.

C. GUI guidance

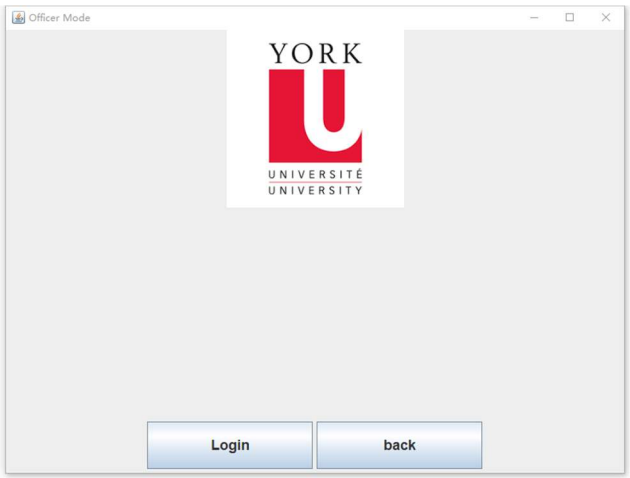
1. Main view



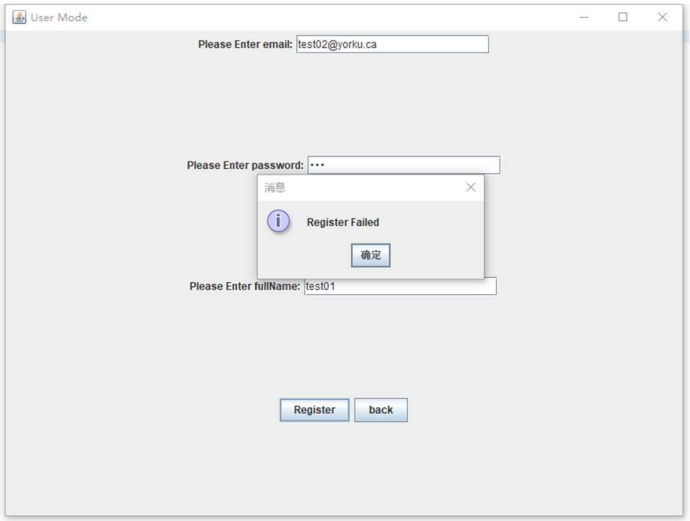
2. User_mode



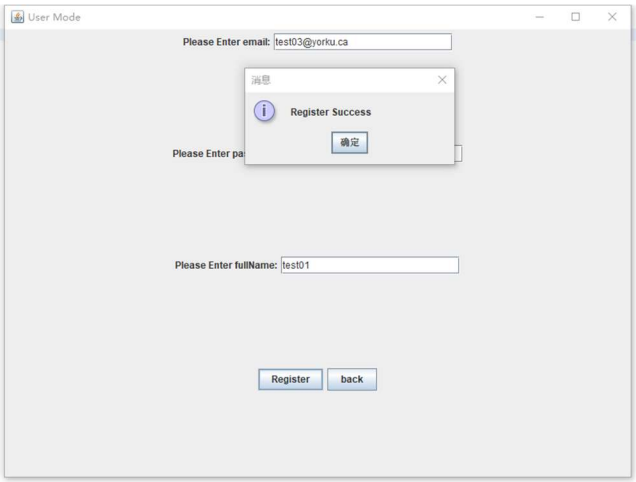
3. Officer_Mode



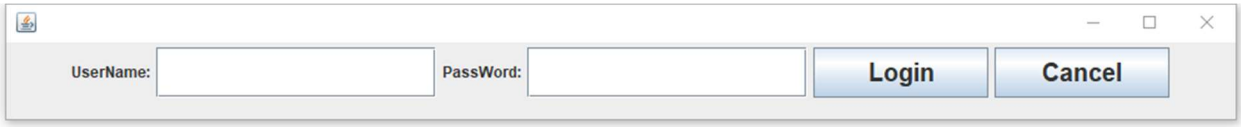
4. Customer register with existing email



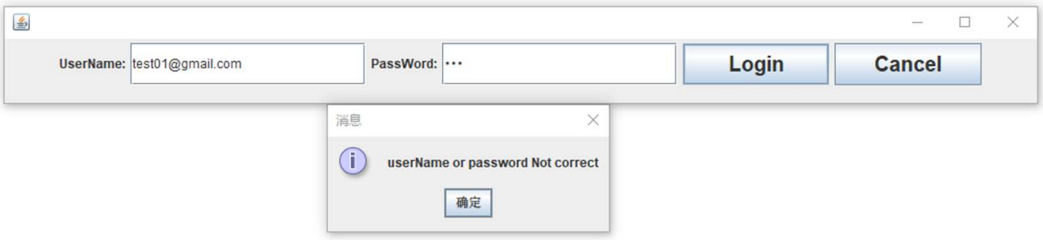
5. Customer register successfully



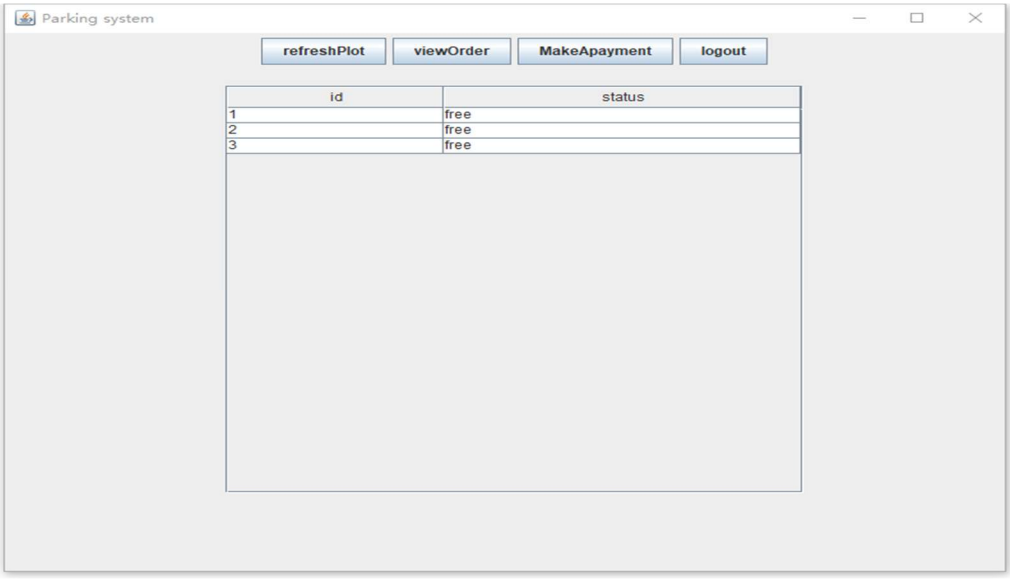
6. Customer login



7. Login failed.

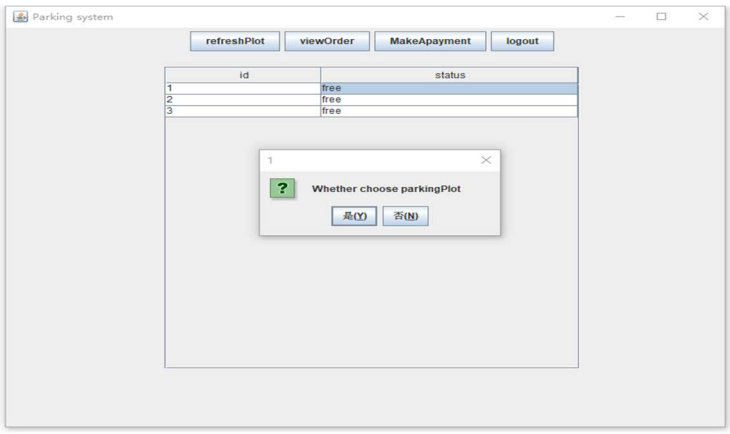


8. Login success

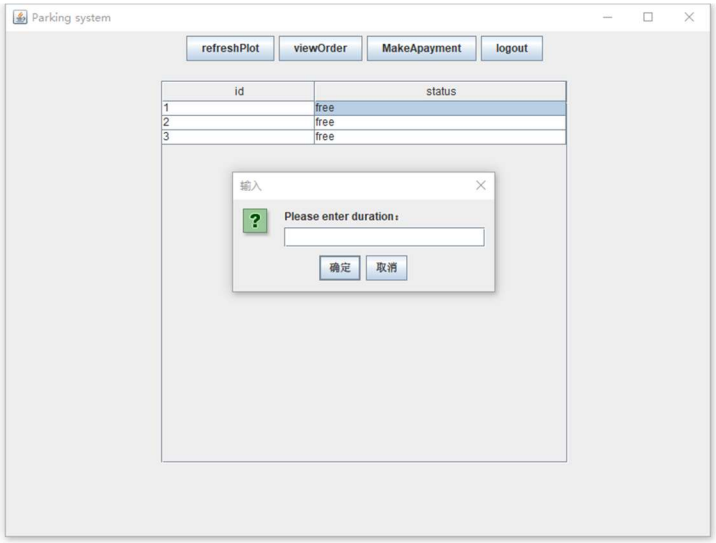


Refresh button for refresh parking lot status, customer can type one of parking plot to book.

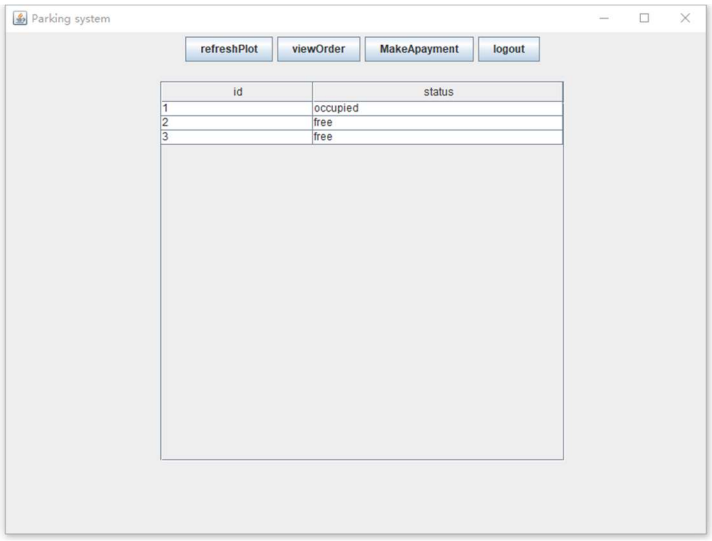
9. Customers tap a plot to book.



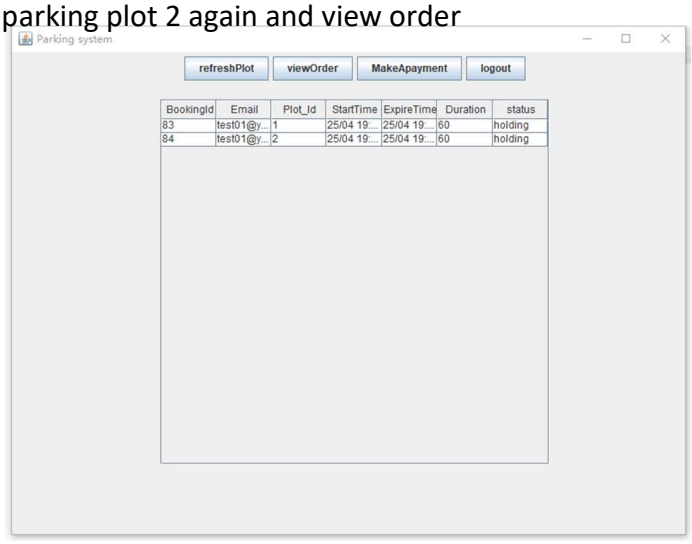
10. Customer needs to enter duration to book.



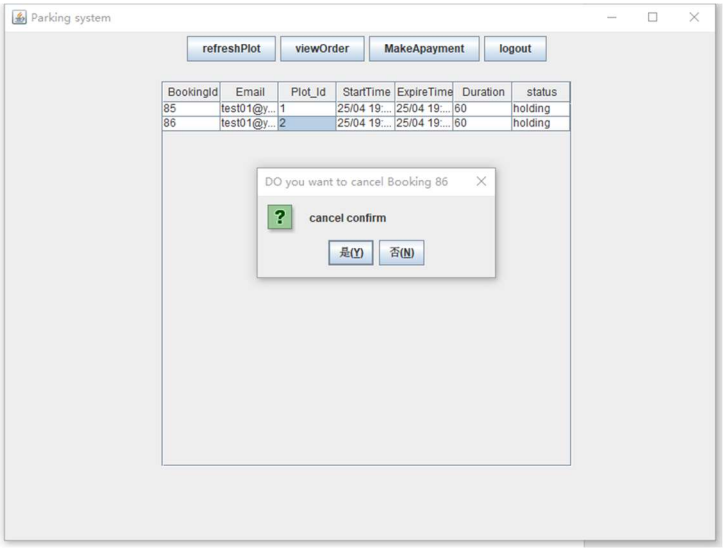
11. Booking successfully, and refresh



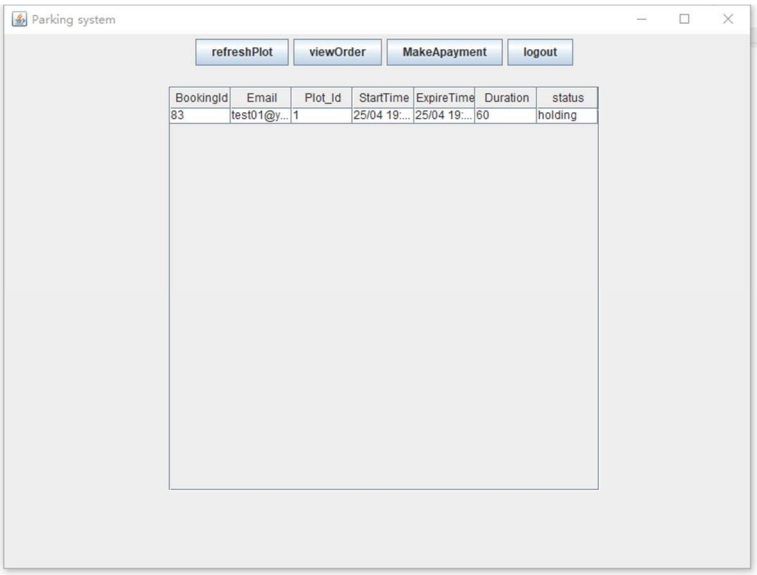
12. Book parking plot 2 again and view order



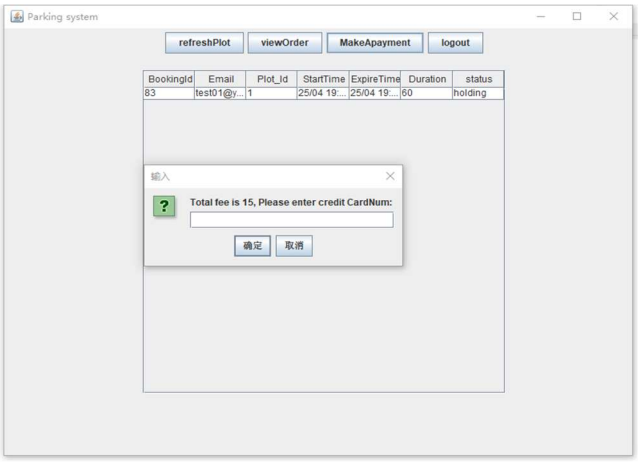
13. Cancel order, just tap order to cancel.



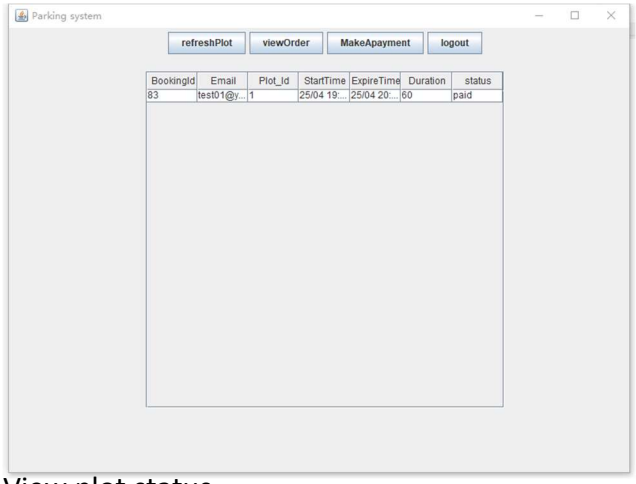
14. Tap viewOrder to refresh order.



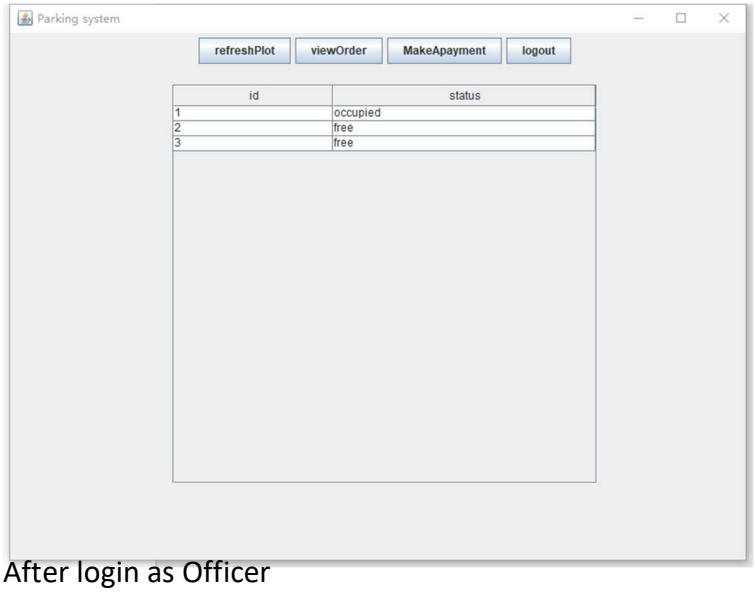
15. Pay for the order.



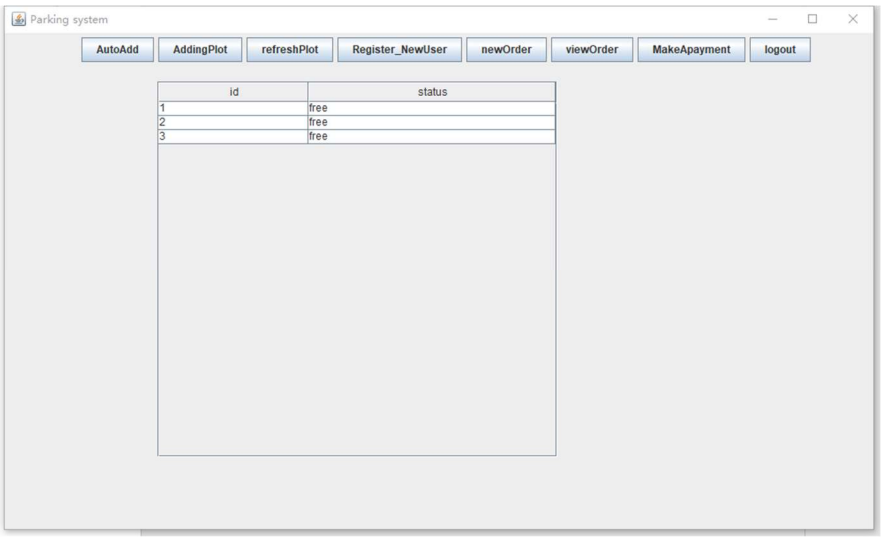
16. View order status



17. View plot status

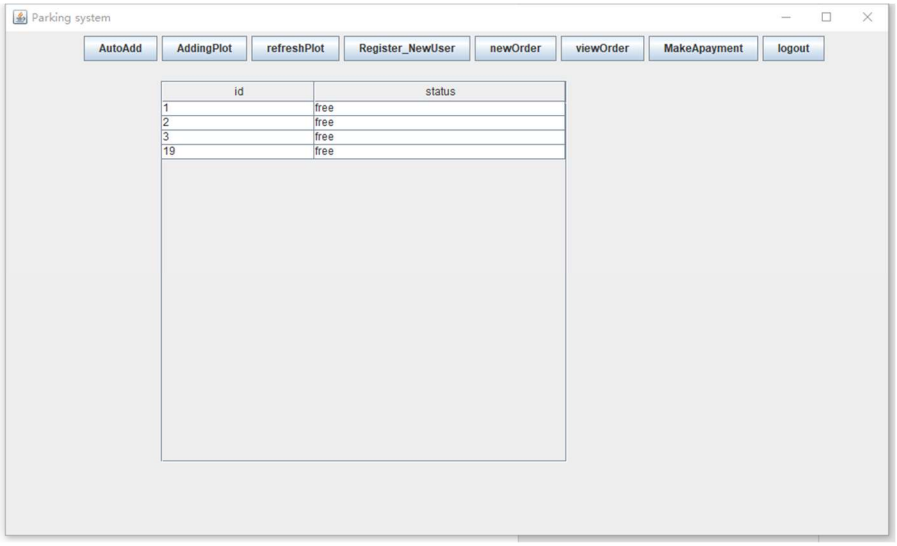


18. After login as Officer



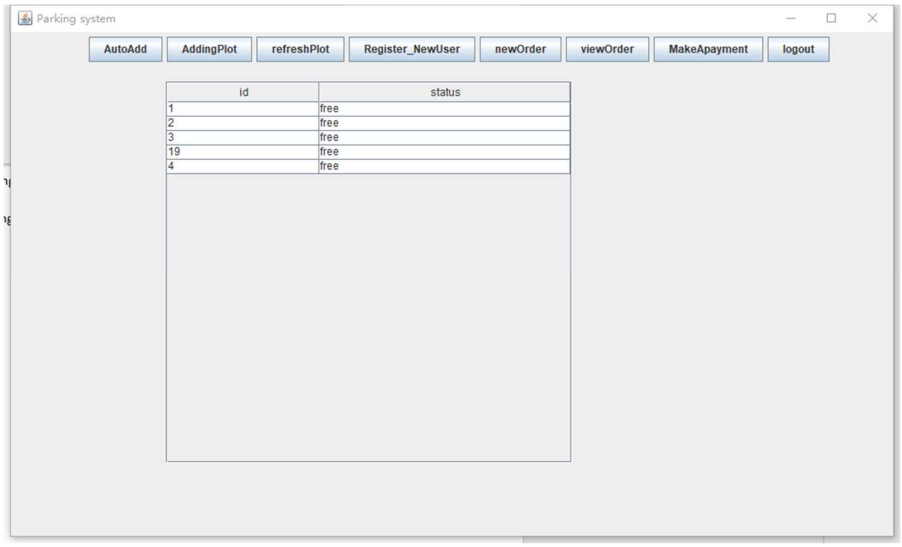
Officer can add plot by tap adding plot with specific non-existing number

19. Adding plot 19

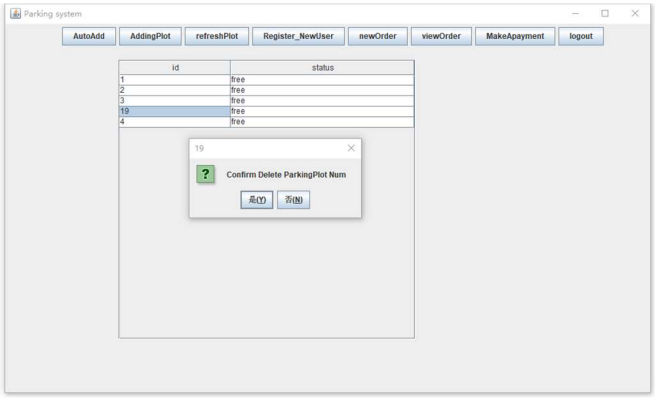


After adding just refresh it.

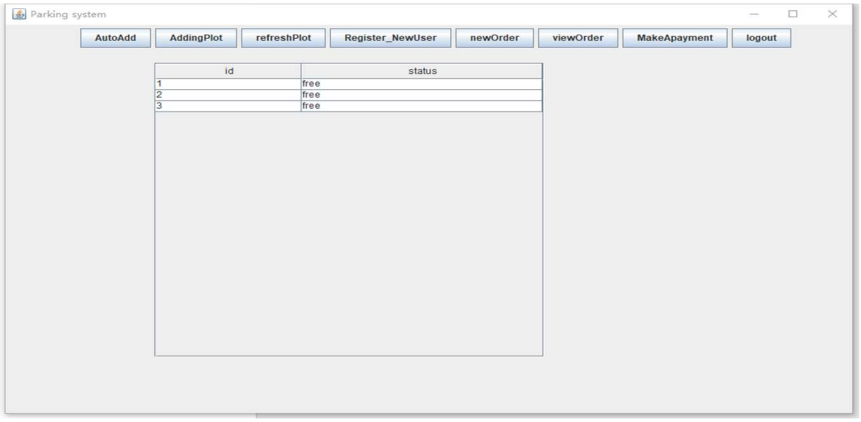
20. Auto adding a plot



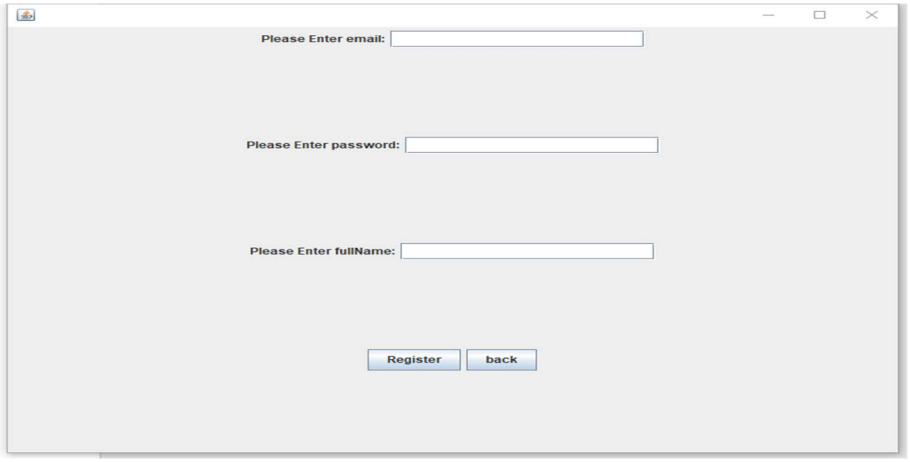
21. Tap one of plot to remove.



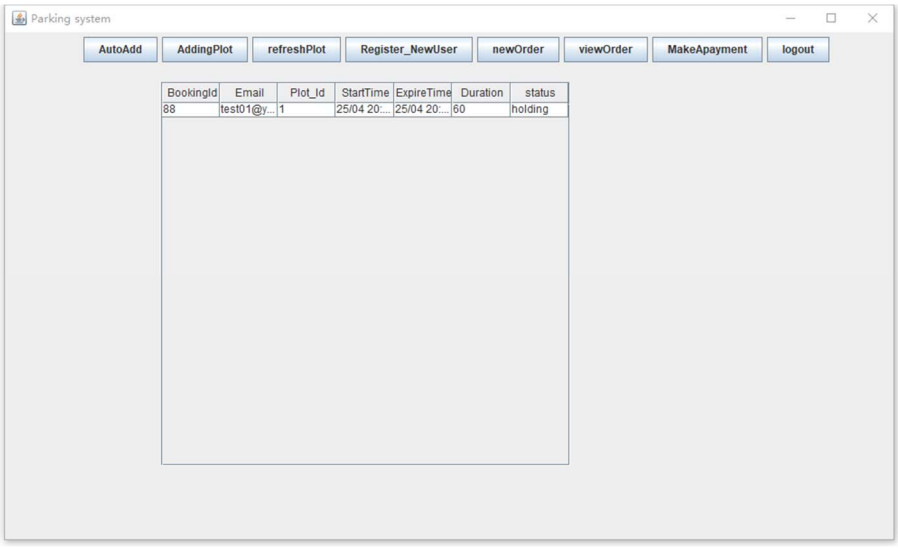
22. After removing, refresh the plot.



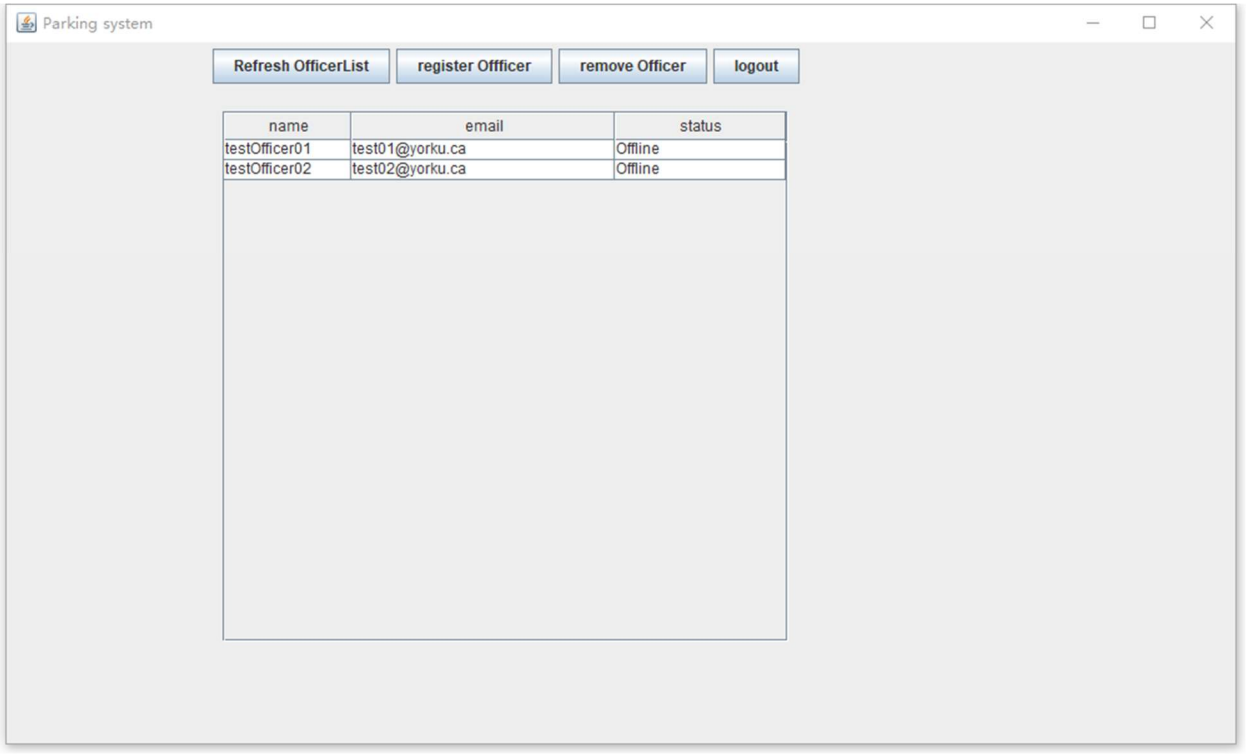
23. Officer can register for new customers.



24. Officer can book new order for customer by enter email, then the system will check whether the email is registered or not. If not, booking failed. If yes, officer need to tap id of parking plot and the duration to finish booking. After finishing booking successfully, officer can tap vieworder and enter the valid email to check the order.



25. Officer can tap makeAplayment, enter the valid email and then enter the cardnumber to help user finishing booking.
26. Admin mode



Admin can register Officer by tapping register Officers. It will add officer info both in Customer Database and Officer Database. Admin can tap refresh button to check current officer condition and tap one of officer to remove it like that officer removed parking plot. Admin also can tap remove button to remove specific officer by enter valid email. Officer can only be removed when they are offline.

D. Junit test

I test as customer, officer, and admin. All the function including payment booking register, parking plot management are covered.

Element	Coverage	Covered Instructi...	Missed Instructi...	Total Instructions
finalProject	35.9 %	3,046	5,429	8,475
src	35.9 %	3,046	5,429	8,475
view.officer	0.0 %	0	1,333	1,333
view.user	0.0 %	0	1,138	1,138
view.admin	0.0 %	0	1,033	1,033
view.bookingJPanel	0.0 %	0	737	737
view.table	0.0 %	0	737	737
parkingDB	90.4 %	2,250	239	2,489
view	0.0 %	0	212	212
junitTest	100.0 %	796	0	796