COMP90015 Distributed Systems Assignment 2

Distributed Shared White Board

*Team: 9523*

**System Description**

The purpose of the project is to design and implement a shared whiteboard system which allows multiple users to draw concurrently. In this system, people can create a new whiteboard or join one after allowed by its manager. Also, the users who are drawing the same canvas can communicate with each other.

Based on the peer to peer architecture, our team creatively set a central server to achieve the management of multiple users drawing the same time. RMI is implemented for communication, including the synchronous painting and chat function.

In this paper, the designing of the architecture will be firstly shown. Then the how RMI is used in this project will be talked about, including the design of an RMI object and the message format. Also, the implement details are explained. And we will then talk about the innovative part and the further improvements. Finally, the contribution of each member will be outlined.

**Architecture Design**

***Peer to Peer Model***

Peer-to-peer Model has been used in this project. Whether the system user wants to be a manager or a user, he is equivalent to a peer in the system.

To be more specific, in a shared whiteboard, the creator of the whiteboard is connected with every user who joins his whiteboard through the peer to peer model, and it is shown below.

Once there are some changes exist in the white board from a user, the change will first be shown at his white board. After that, the manager should be informed. Then, the manager calls the remote object to pass the change to any other white board except the change maker. Hence, there are not obvious difference between them except the manager keeps the user's remote object.

A close up of a map

Description automatically generated

***Central Server***

In this project, a central server is needed to control the system, as they can clearly handle the state of each user and the information of the shared whiteboards.

Firstly, when a user wants to login, he needs to input a username. However, it must be ensured that users who log in at the same time have different username, which can help to distinguish them. And this is the reason why a user array has been maintained by central server.

Secondly, as we want each user can select whether they want to become a client or a manager of a white board, the central server is also necessary.

If the user would like to create a new whiteboard or open his exist whiteboards, the information should be recorded in the central server. Also, once a client wants to join the white board, he needs to select among a series of available managers. The central server is able to provide the list of available white board in this procedure.

**Communication Design**

***Java RMI***

***Message Format***

**Function Design**

***Login***

In this system, the username is the only basis for identifying users. To realize this function, a central server should be used to maintain the unique username. A user array is used to store user information for each user. Once a user login, the username he enters needs to determine if it already exists in the user array. If it already exists, the user will need to change his username.

A screenshot of a cell phone

Description automatically generated

***Client Management***

For the client who want to join a white board are managed by server. Central server generates the available server white board for client. Once the user selects which whiteboard they want to join, the corresponding manager will receive the user's join request. The manager maintains an array of online users and an array of requesting users. If the manager wants to accept the user, the user will be removed from the application array and added to the online user array. When the manager want to close the white board, the close function of each remote object will be invoked, after each client exit from white board, the manager will exit from the white board.

***Painting***

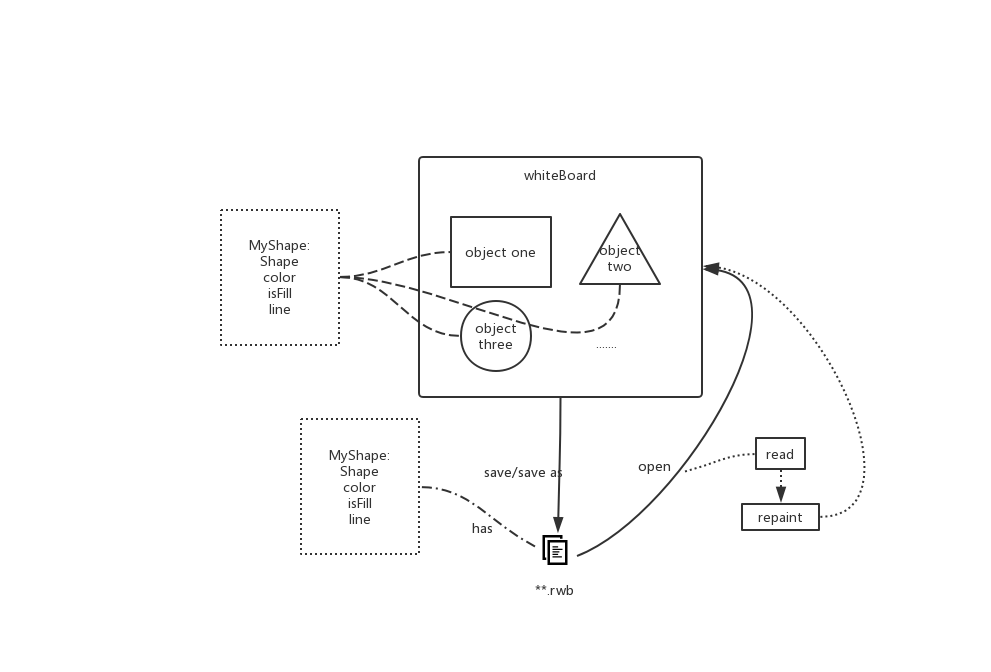
As for the painting function, it realizes by graphic2d application interface. Firstly, the mouse pressed and released coordinates has been captured. In order the draw different shapes on the whiteboard, we use class “Rectangle2D”, “Ellipse2D” and “Line2D” to create the corresponding graphic object based on the coordinates of the mouse. Each shape is stored in Vector container with painting attributes. In addition, the shape will keep changing during the mouse dragging process. In order to achieve such an effect, a temporary shape is drawn in a short interval. In the same time, the previous temporary shape has been overwritten. Unlike drawing shapes, the brush function records the path that the mouse drags through.

*File*

The manager of a whiteboard has the ability to control their blackboards files. As is mentioned before, in this project, we set each shape as an object named “MyShape” to represent it. To be more specific, for each shape, its shape, color, filled or not and line thickness are its attributes.

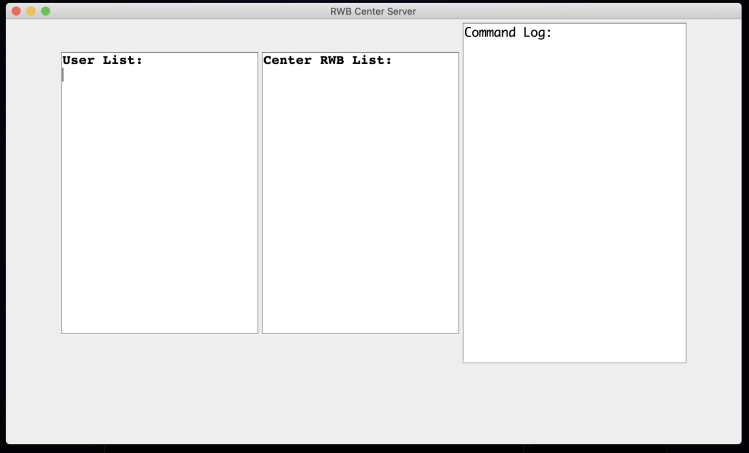
Once the whiteboard is needed to be saved, every “MyShape” object would be saved into the file with a name from manager and a specific file suffix name: “.rwb”. What’s more, “save as” function also allow the manage to choose a dictionary for the file.

As for the “open” function, the client can open the exist whiteboard by reading the file of the suffix name: “.rwb”, and repaint each “MyShape” object in this file.



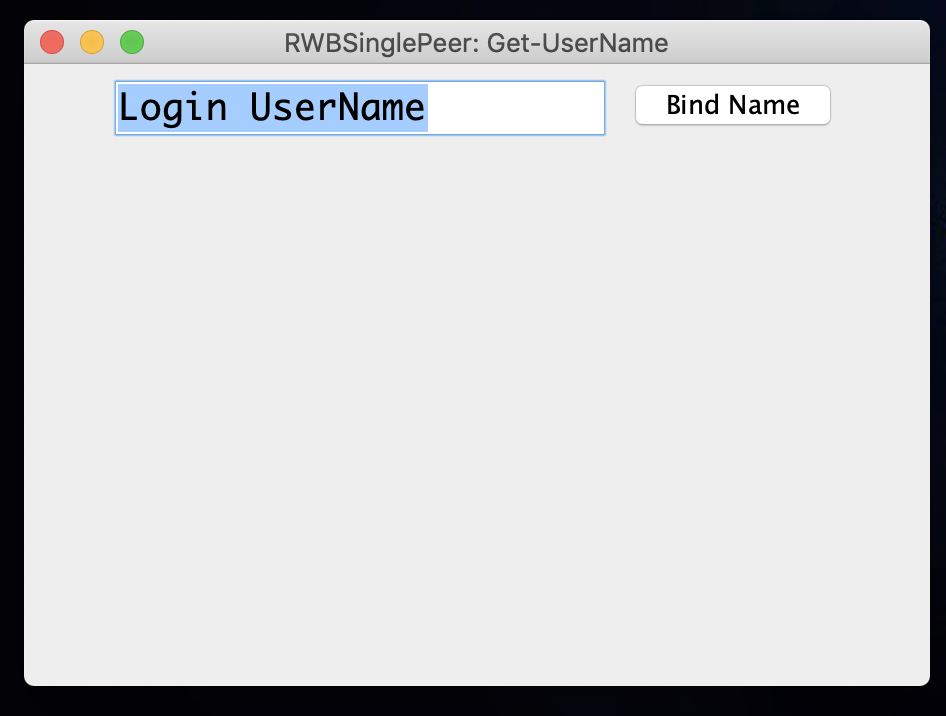
***Communication***

**User Interface Design**

****

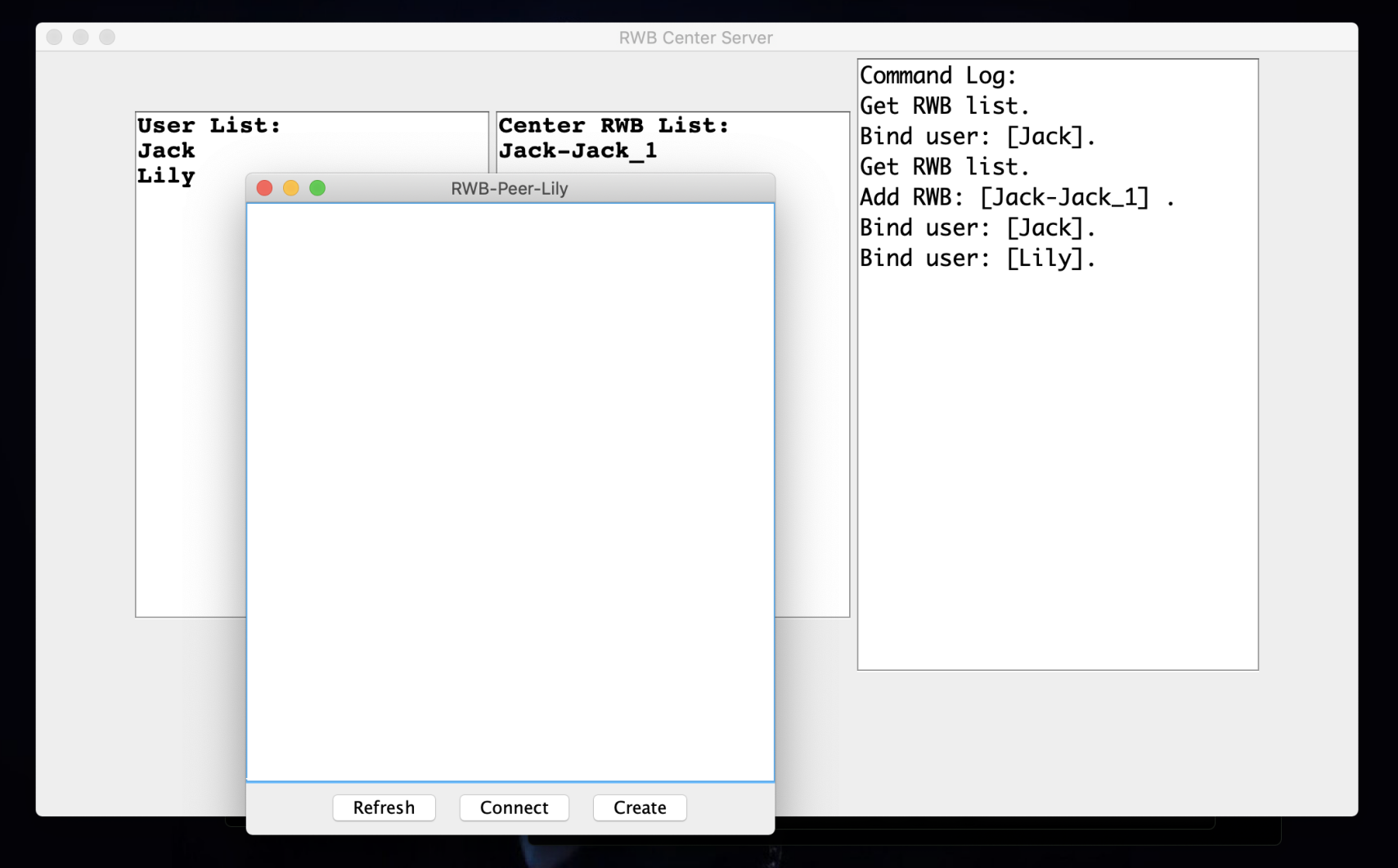
---Figure 1---

In Figure 1, it shows the user interface for central server. It contains the field to record the user list and the center remote white board list. It will continue to update the contents of the list as the user joins.



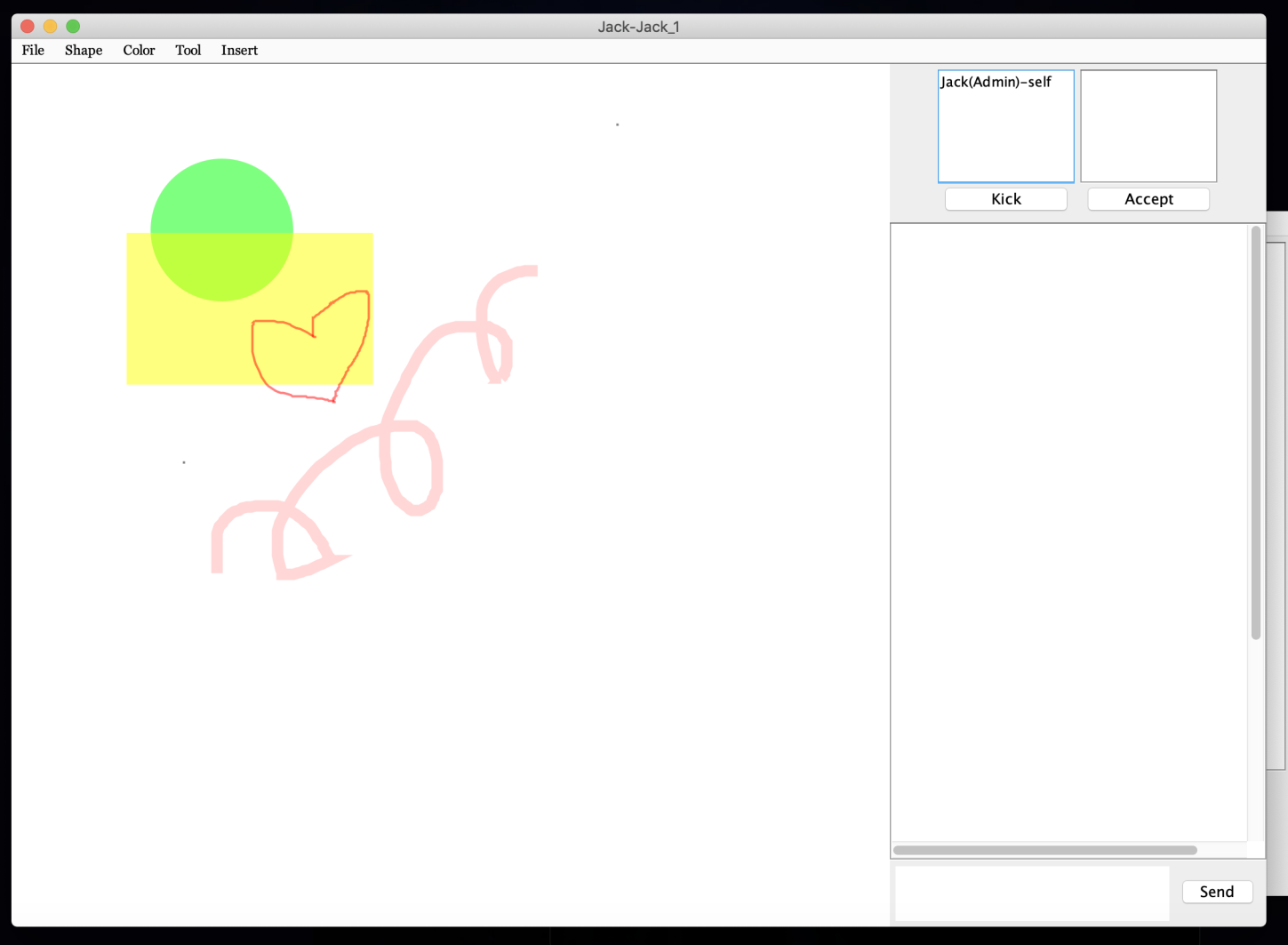
---Figure 2---

Figure 2 shows the page for login function, the user can input the username. If the username already exists, it will say that username already exist. Otherwise, the system will jump to next page, which has been shown in Figure 3.



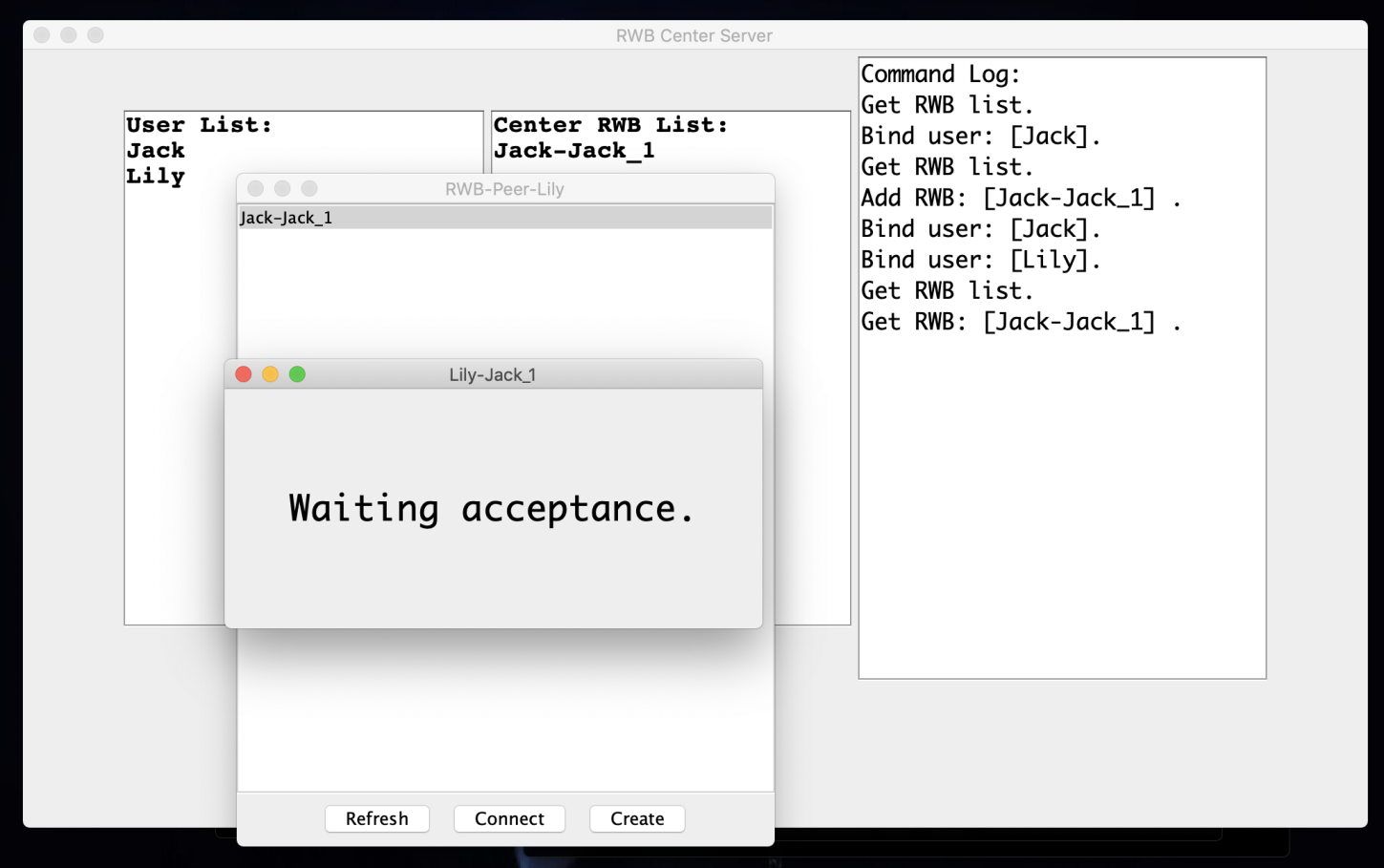
---Figure 3---

In this page, user can select whether he want to be a manager or a client of a white board through “create” or “connect” button. The refresh button use to get the latest available remote white board. If the user clicks create, a manager white board page will be shown as Figure 4. It has an white board on the left. The right side shows the online user of current white board. Kick button can move the user out of the white board. Another text field shows the applied user, accept button can move them into online user field. The field in right bottom is a message box. If user input the message can send to white board through send button. On the top of this page, a tool bar has been shown, user can select the shape color or tool.



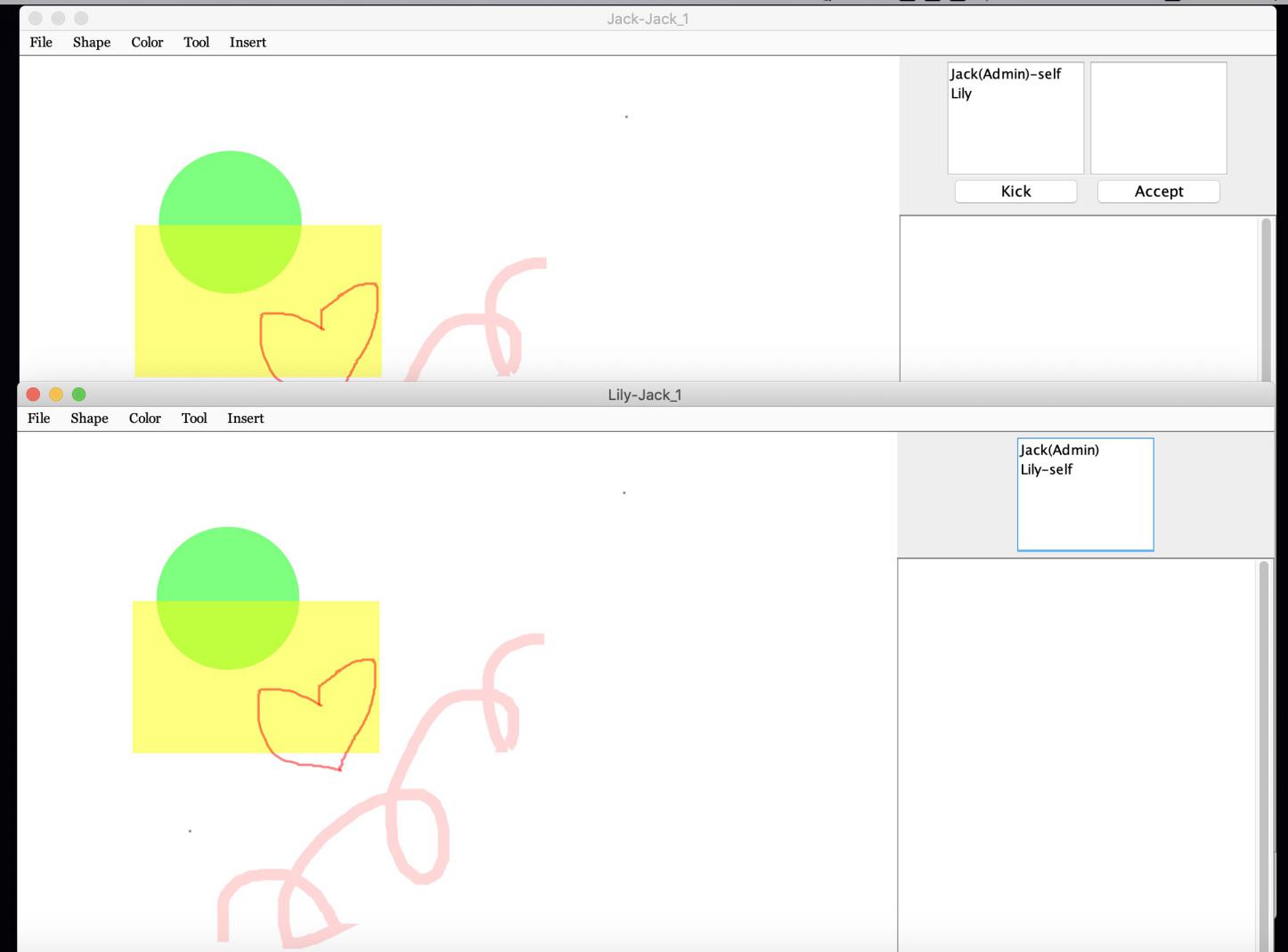
---Figure 4---

If the user wants to join a white board, a list of available white board has been shown. After the user selects the server, a prompt box will show like Figure 5.



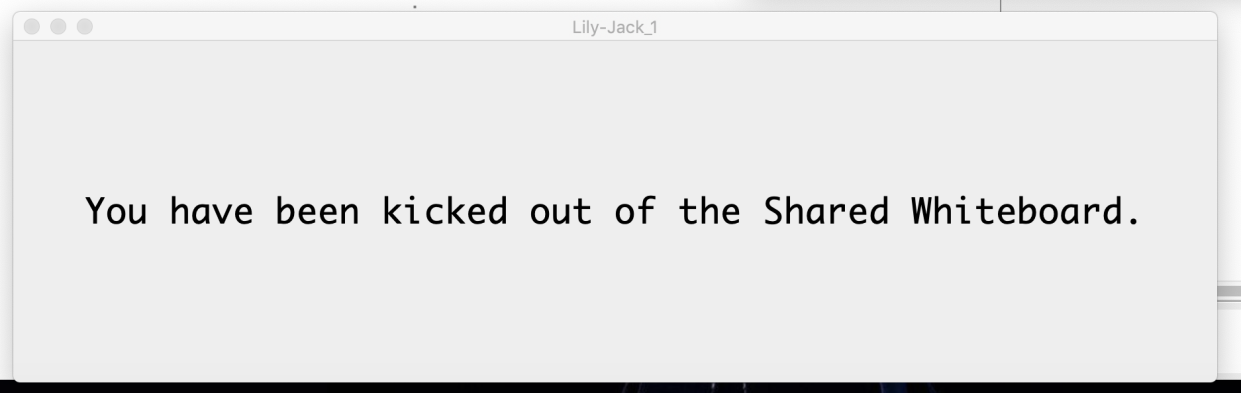
---Figure 5---

Once the manager accept the client, the client white board page will be shown. In Figure 6, it pointed out the difference between the manager white board and the client white board. Client white board only can see the online white board users. However, the manager white board has a field to control the online users.



---Figure 6---

Once manager select the online user and clicked the kick button, corresponding client will jump to prompt page which has been shown in Figure 7.



---Figure 7---

**Creative Design**

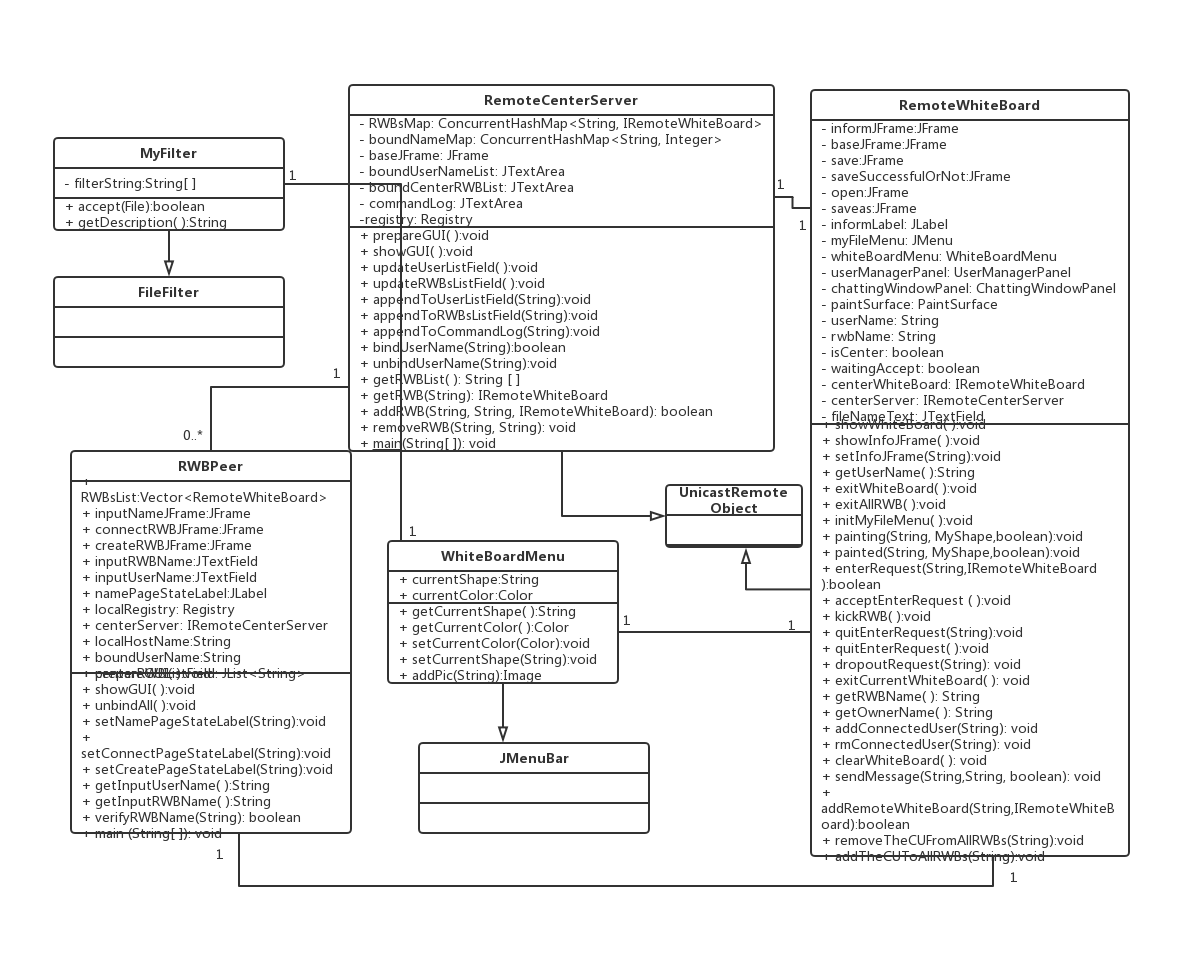
***Central Server Interface***

In our system, the central server has an interface to shows the online user and available remote white board. Once there are some new user join the system or a new white board has been created, the latest information will be shown in the interface. Hence, central server responsible for managing all users and whiteboards.

**Conclusion**

**Appendix**

***Class Diagram***

******

***Member Contribution***

|  |  |  |
| --- | --- | --- |
| Name & Std No, | Contribution Area | Overall Contribution to Project |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |