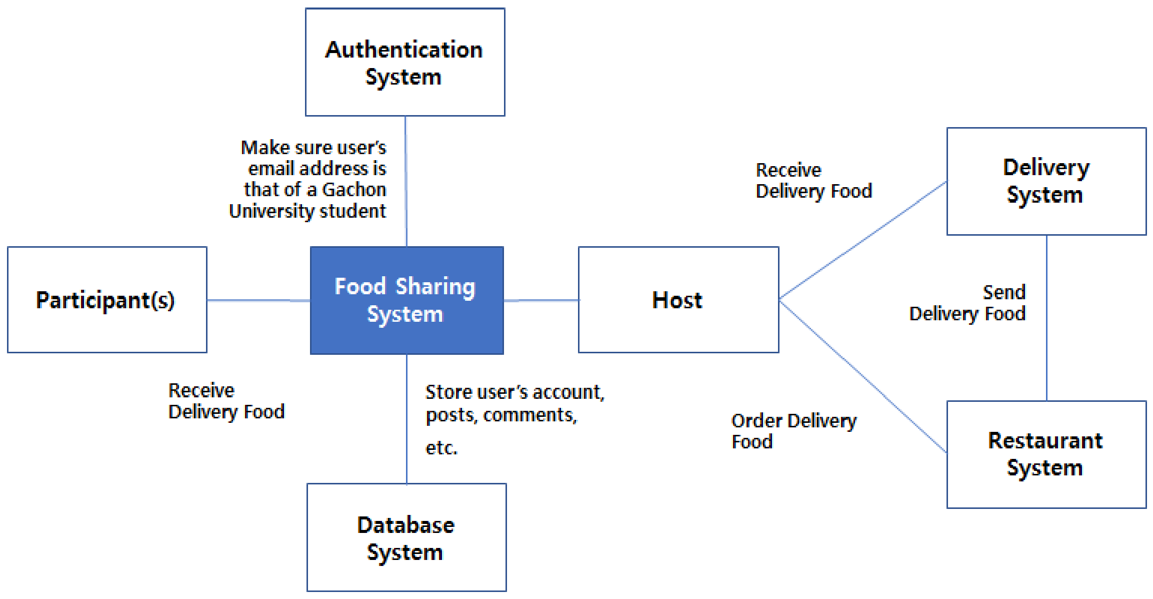
System Modeling for Term Project

Software Engineering

Team 9 | 노민하, 박다정, 안세훈, 장건

Content

1. Context Model

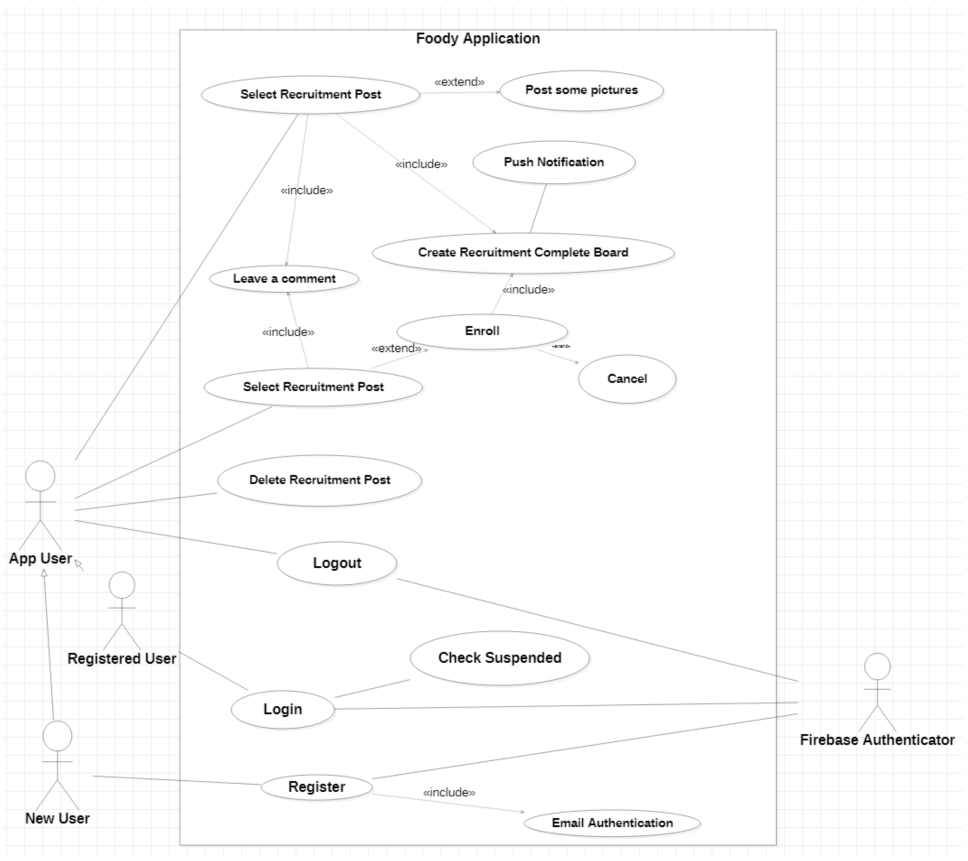


Description - External Systems

[Database system] - Contain a user's account, postings, comments, and etc.
[Authentication system] - Make sure the user's email address is that of a Gachon University student.
[Host] - A user who creates the posting to share the delivery food.
[Participant(s)] - A user who participates in the posting to share the delivery food.
[Restaurant system] - Get orders and send delivery food to the delivery system.
[Delivery System] - Get the delivery food and send the food to the host.

2. Interaction Model

a. Usecase diagram



Description

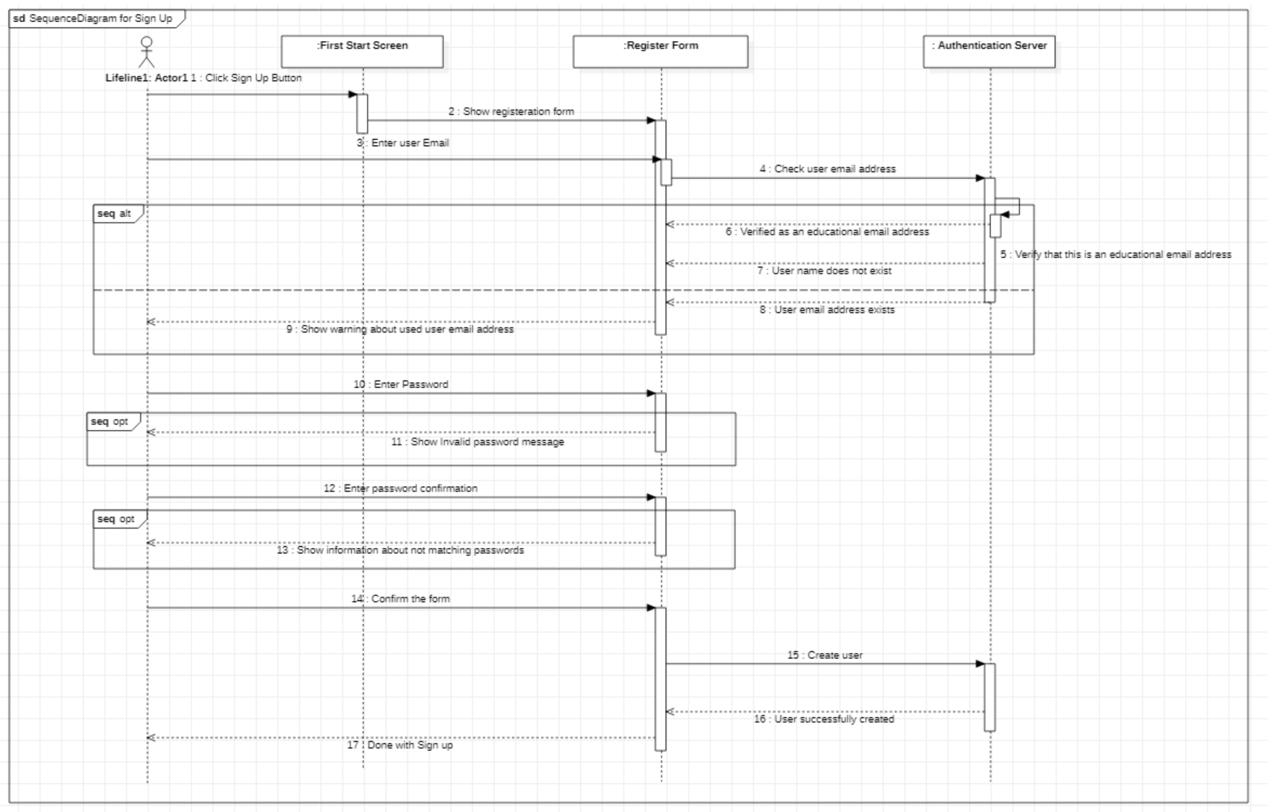
[New User] Register a new account. At that time, Firebase authorizes the user as a university student or not.

[Registered User] Check whether the user is suspended or not.

[App User] Select Recruitment Post. If not, User Enrolls a new post with some pictures. Then the user becomes the host of the order. Host can cancel the post. All participants can leave a comment and discuss their order. If all participants join(maximum number of participants), the post is moved to the recruitment complete board. Then, the host sends a push notification about the delivery process.

b. Sequence Diagrams

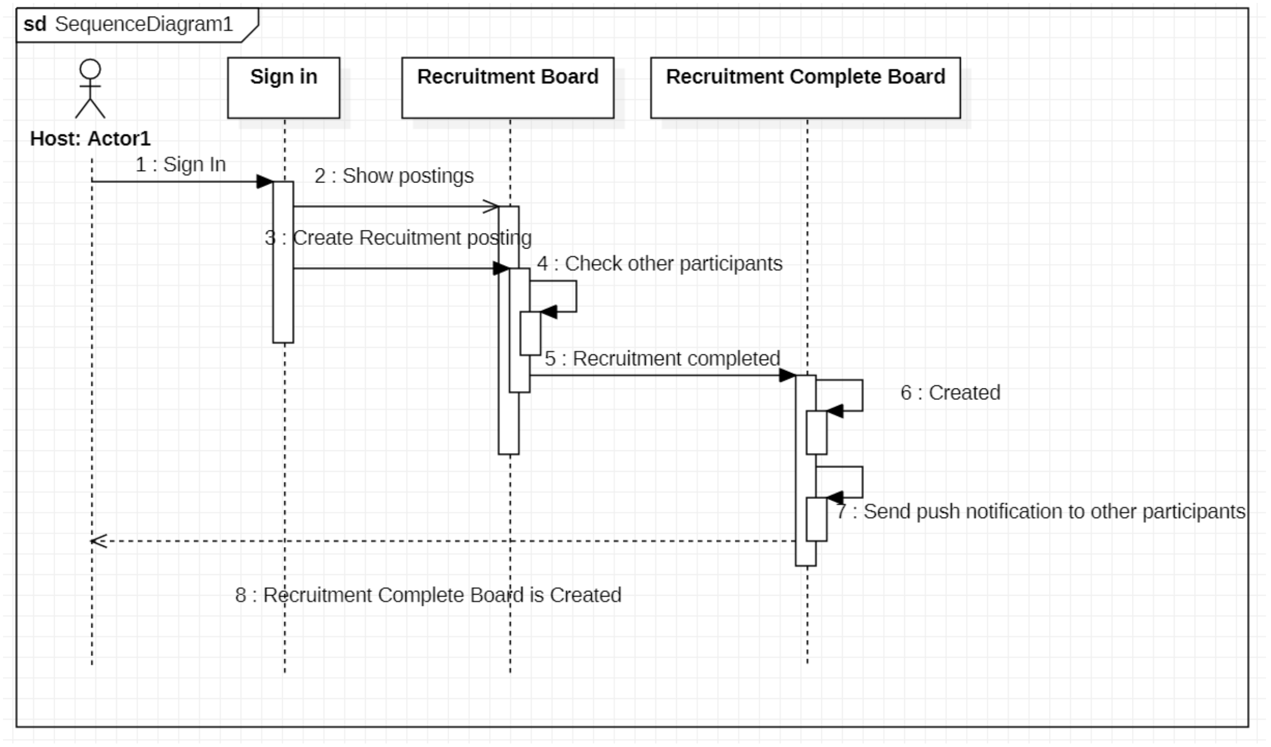
* Diagram for Sign up


sd SequenceDiagram for Sign Up
:First Start Screen
:Register Form
: Authentication Server
Lifeline1: Actor1
1 : Click Sign Up Button
2 : Show registeration form
3 : Enter user Email
4 : Check user email address
seq alt
6 : Verified as an educational email address
5 : Verify that this is an educational email address
7 : User name does not exist
8 : User email address exists
9 : Show warning about used user email address
10 : Enter Password
seq opt
11 : Show Invalid password message
12 : Enter password confirmation
seq opt
13 : Show information about not matching passwords
14 : Confirm the form
15 : Create user
16 : User successfully created
17 : Done with Sign up

Description

1. Users who have not signed up for membership press the sign-up button.
2. Send authentication mail to the user account for authentication.
3. If the email account is a university account and the user clicks the Approval button, authentication is complete.
4. The user enters the password and password verification.
5. User presses confirm to send information to the server.
6. An account is created on the server and ends the sign up process.

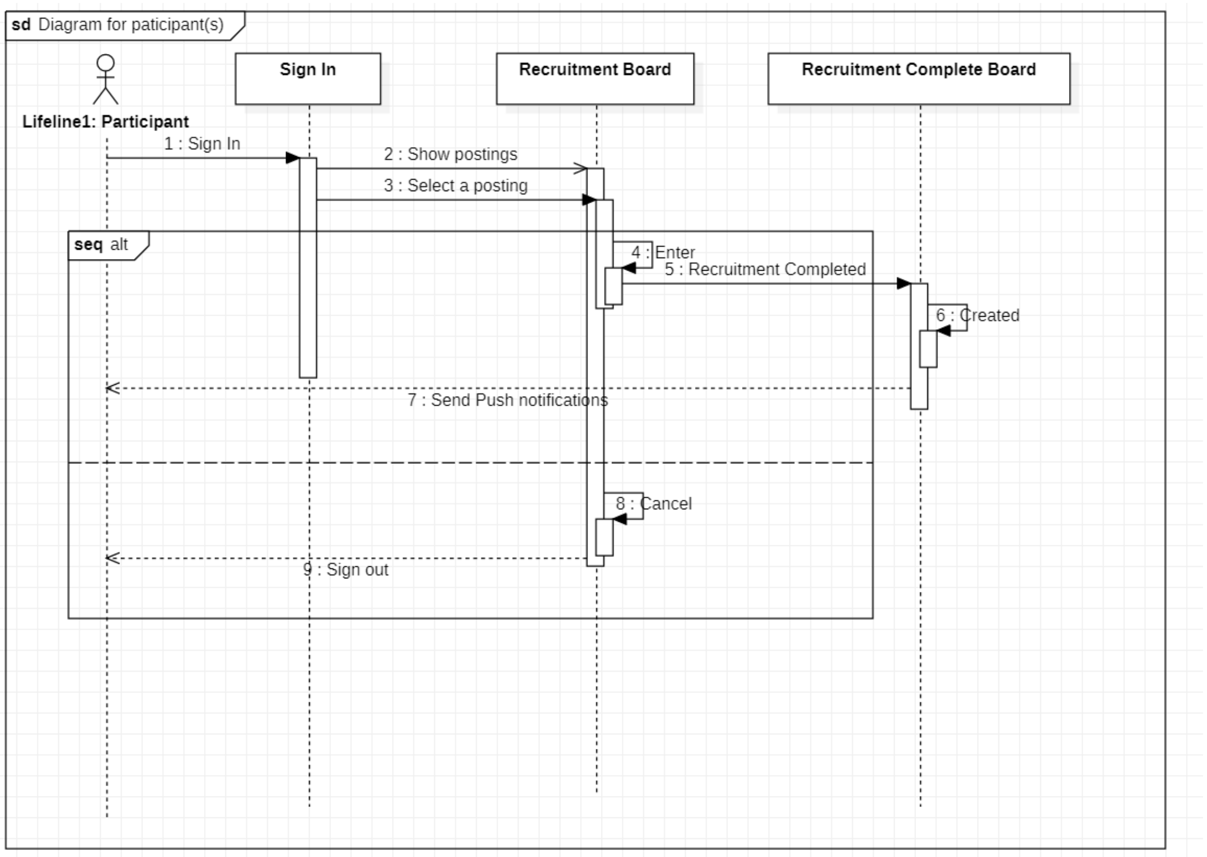
* Diagram for Host


sd SequenceDiagram1
Sign in
Recruitment Board
Recruitment Complete Board
Host: Actor1
1 : Sign In
2 : Show postings
3 : Create Recuitment posting
4 : Check other participants
5 : Recruitment completed
6 : Created
7 : Send push notification to other participants
8 : Recruitment Complete Board is Created

Description

1. Users run the app. Users have to log in to use the app, so the user enters userID and PW. If the user doesn’t have an account, create a new account and authorize that he or she is a university student.
2. After login, the system displays the posts(recruiting posts) and waits for the next movement of the user.
3. If the user closes the app, the process ends.
4. If a user writes a new post, the post is uploaded to the recruiting board. The user is the writer(or host) of the post, so only he or she can edit or delete the post, and edit maximum participants. The recruiting is finished when the maximum number of participants are joined.
5. The post is moved to the recruitment complete board.
6. If it ends, the host places their orders to the restaurant, and pushes the *order placed* button. The system will send push notification that the host just placed an order for delivery.
7. When the delivery food arrives, host push *arrived* button so participants can receive the push notification.

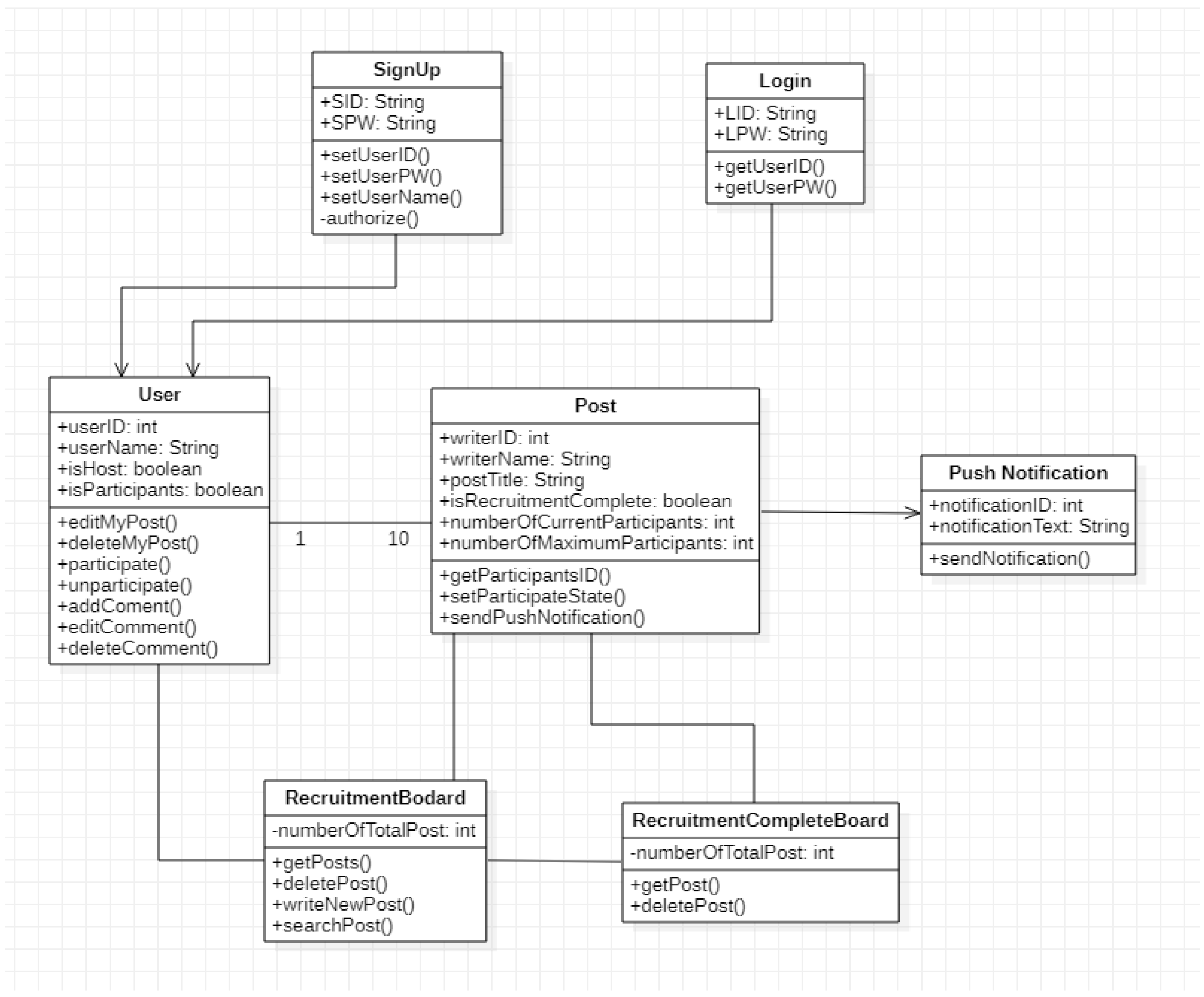
* Diagram for participant(s)



Description

1. If the user has finished signing up, the user login to use the app, so the user can enter user ID and PW.
2. After login, the system shows the recruiting posts and waits for the next movement of the user.
3. If the user wants to participate, press the join button. Users can undo joining at any time before the recruitment is over.
4. Host and participants can communicate with each other through comments. They can leave their special orders in the comments and discuss it with each other.
5. If it ends, the host pushes the *order placed* button. The system will send push notification that the host just placed an order for delivery.

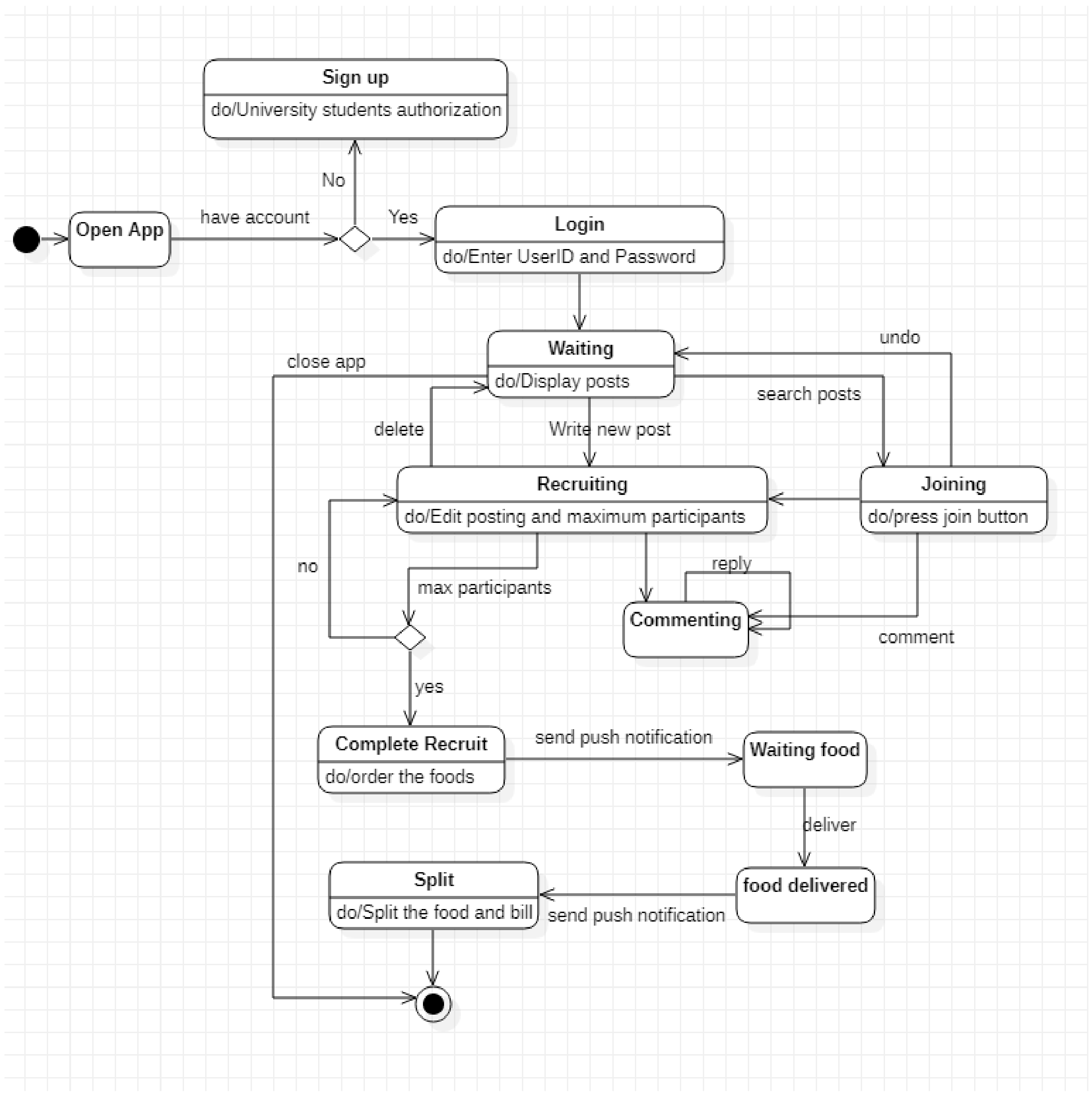
3. Structural model



Description

[SignUp] Register new user; input new ID and PW. Set it as UserID, UserPW. Check if the user is a university student.
[Login] New users try to log in; Compare the LoginID & PW with UserID & PW.
[User] Every user has userID, userName, boolean type of host/participants. Users can edit/delete the post if he/she is the host. If the user is not a host, the user can participate in the other recruitments. Users can cancel their participation at any time. The participants can add/edit/delete their comment on the post.
[Post] Post has writerID, writerName, Title of the post. The post checks the current participants and maximum participants. It has a recruitment state, either Complete or not.
[RecruitmentBoard] New posts are posted on the recruitmentBoard.
[RecruitmentCompleteBoard] If recruitment is finished, the post is moved to the recruitment Complete Board.

4. Behavior model



Description

1. Users run the app. Users have to log in to use the app, so the user enters userID and PW. If the user doesn't have an account, create a new account and authorize that he or she is a university student.
2. After login, the system displays the posts(recruiting posts) and waits for the next movement of the user.
3. If the user closes the app, the process ends.
4. If a user writes a new post, the post is uploaded to the recruiting board. The user is the writer(or host) of the post, so only he or she can edit or delete the post, and edit maximum participants. The recruiting is finished when the maximum number of participants are joined.

5. If the user wants to participate, press the join button. Users can undo joining at any time before the recruitment is over.
6. Host and participants can communicate with each other through comments. They can leave their special orders in the comments and discuss it with each other.
7. If it ends, the host places their orders to the restaurant, and pushes the *order placed* button. The system will send push notification that the host just placed an order for delivery.
8. When the delivery food arrives, the host pushes the *arrival* button so participants can receive the push notification.
9. All the participants gather and receive their own food.