5.3. Planning for the next problems to solve
Tue17 Grape – Team 3

Samuel He
Jayden Leung
Steven Luong
Christopher Nassif
Nicholas Tandiono

Table of Contents

[Requirements] Elicitation

Questions:

1. What is a messaging system like flockr you use?
    a. What features do you commonly use?
    b. What features do you like in these platforms?
        i. Why do you like these features?
    c. What features do you dislike in these platforms?
        i. Why do you dislike these features?
2. What features do you wish were added or changed in flockr?
    a. What would this feature need to do?
    b. When would this feature be used?
    c. Who would use this feature?
    d. How would you like to interact with this feature?

Interviewee 1:

Name: Matthew Leung

Email: z5164311@ad.unsw.edu.au

Responses:

1. Discord
    a. My channels, profile
    b. I like the channels function and the profile function
        i. They help me communicate with the people I want to.
    c. None
        i. None
2. I would like there to be private messaging
    a. Enable messaging directly to profiles without needing to create a channel
    b. When I want to disclose personal information or just messages without interrupting a channel.
    c. Me
    d. I would like to be able to send text, numbers, symbols, emojis, stickers, GIFS, images and videos as messages

Name: Isaac Seeto

Email: z5310048@ad.unsw.edu.au

Responses:

1. Discord, messenger
    a. File sharing and messages
    b. Dark mode, online/offline status, voice call, editing
        i. convenient and easy to use
    c. Music bots, spam notifications, size limit to files
        i. I just hate groovy bots. notifications whilst they can be muted can be annoying to come back to, requires file compression or another tedious mode of file transfer
2. I wish that I could see if users are online, idle, do not disturb or offline.
    a. It would need to display the users current status
    b. Day to day use
    c. Everyone
    d. Being able to change my own status in my profile settings or something alike

Interviewee 3:

Name: Thinh Do

Email: thinh.do@gmail.com

Responses:

1. Discord, skype
    a. Messages
    b. Screen sharing feature
        i. You can stream your screen for showing and explaining things
    c. Also screen sharing feature
        i. Low frame rate, has to pay for higher frame rate
2. Being able to add people
    a. Add someone else to be your friend
    b. Whenever I feel like adding someone to my friend list
    c. Me
    d. I would like to search for users where I can choose to add them

User story for Matthew Leung:

As a student I want to be able to message text, numbers, symbols, emojis, stickers, GIFS, images and videos to other users directly without the need of a channel so that I can disclose personal information privately while also not disturbing others.

User Acceptance Criteria

- Private messages can be accessed on the side bar
- Ability to private message everyone on the platform
- Ability to find someone to message with their first name and last name
- Ability to input text, numbers and emojis into the message box
- Stickers can be sent with a stickers selection tab
- GIF can be sent with a GIF selection tab
- Images and videos can be uploaded with a drag and drop into the message box

Use Case:	Private messaging
Goal in Context:	For users to directly message each other without the need of a channel.
Scope:	Flockr infrastructure
Level:	Primary Task
Preconditions:	The user is already registered in Flockr and there is more than one user in Flockr
Success End Condition:	The correct message is displayed on the recipient private message page.
Failed End Condition:	No message is displayed
Primary Actor:	User
Trigger:	Pressing the send button

MAIN SUCCESS SCENARIO

Step 1. User clicks on private message tab

Step 2. User searches for a user in search bar

Step 3. User opens a new conversation with that user

Step 4. User types messages, numbers, symbols and emojis in message box

Step 5. User selects a sticker from sticker selection tab

Step 6. User selects a GIF from GIF selection tab

Step 7. User drags image into message box

Step 8. User drags video into message box

Step 9. User presses the send button

Step 10. The other user receives the message in correct order

User story for Isaac Seeto:

As a busy student, I want to see if people are online or offline so I can know if my message will be seen immediately or later.

User Acceptance Criteria

- When the user has set "online", a green circle will appear around the profile picture
- When the user has set "do not disturb", a red circle will appear around the profile picture
- When the user has set "idle", a yellow circle will appear around the profile picture
- When the user has set "offline", a clear circle will appear around the profile picture

Use Case:	Status User Status
Goal in Context:	For users to see the activity of each other
Scope:	Flockr infrastructure
Level:	Primary
Preconditions:	The user is already registered in Flockr
Success End Conditions:	A green circle will appear around the profile picture when the user has set "online" status

	A red circle will appear around the profile picture when the user has set "do not disturb" status
	A yellow circle will appear around the profile picture when the user has set "idle" status
	A clear circle will appear around the profile picture when the user has set "offline" status
Failed End Condition:	No green circle will appear around the profile picture when the user has set "online" status
	No red circle will appear around the profile picture when the user has set "do not disturb" status
	No yellow circle will appear around the profile picture when the user has set "idle" status
	No clear circle will appear around the profile picture when the user has set "offline" status
Primary Actor:	User
Trigger:	Selecting user status

MAIN SUCCESS SCENARIO

Step 1. User clicks on user profile tab

Step 2. User clicks on user status drop down box

Step 3. Selects online status

Step 4. User is able to see a green glowing circle around the profile picture of the current user.

As a student, I want to be able to add friends on Flockr so I can easily message them.

Create a User Acceptance Criteria

- Each user has a friends list
- Ability to send a friend request to other users
- Ability to respond to a friend request, either to 'accept' or 'ignore'
- Ability to view pending friend requests

Use Case:	Add friends
Goal in Context:	Add users into their friend list.
Scope:	Flockr infrastructure
Level:	Primary Task
Preconditions:	The user is already registered in Flockr and there is more than one user in Flockr
Success End Condition:	A friend request is sent to the second user and a pending friend request is visible to the seconds user. Option for the second user to respond to the friend request (accept or ignore) When the user accepts the friend request, the user will be moved from the pending requests list to the friend's list.
Failed End Condition:	The second user received no friend request Both user appeared on each other's friend list despite the 'ignore' response Both user did not appear on each other's friend list despite the 'accept' response
Primary Actor:	User
Trigger:	Entering user's name into the 'add friend' field and pressing the add button.

MAIN SUCCESS SCENARIO

Step 1. Typing user's name onto the 'add friend' field

Step 2. Pressing add button on selected user

Step 3. Going onto pending friends list on second users account

Step 4. Accepting the friend request

[Requirements] Validation

User 1: Matthew Leung

This user requested a direct message feature, with many interactive sub features such as the inclusion of emojis, GIFS, symbols, photos and videos etc. In resolving this we have planned to implement a direct message feature, that can be accessed by clicking the private message tab on the sidebar.

In pitching our solution to the original user, they commented:

"This is a great solution to my problem, which will allow me to communicate with other flockr members, in an interactive way using emojis, GIFS, photos and videos"

User 2: Isaac Seeto

This user requested a status feature, which gives users the abilities to see if a user is online or offline. In resolving this we have planned to implement a status circle that changes colour based on the users status. When the user is online the circle will be green and when the user is offline the clear circle will be absent.

In pitching out solution to the original user, they commented:

"This is a great idea, however I feel the addition of other status' would be greatly helpful. As well as online and offline, maybe you could consider adding do not disturb, away and appear offline."

User 3: Thinh Do

This user requested a friend adding feature, which allows users to add other users into a friend list. In resolving this we have planned to implement a friend list, and the option to add people using their names. The users receiving friend requests will be able to either choose 'accept' or 'ignore' the friend requests. Added users will appear in the friend list.

In pitching out solution to the original user, they commented:

"This is awesome, I can now view my friends list so I can easily message them. I would also like to view my pending friend requests so I know who has sent me a friend request"

[Design] Interface Design

| Function Name	HTTP Method	Parameters	Return type	Exceptions	Description
user/private/message/send	POST	(token, u_id, message)	{ message_id }	**InputError** when any of**:** Message is more than 1000 characters u_id is not valid user	Given a u_id, send a message directly from one user to another.
user/private/message	GET	(token, u_id, start)	{ messages, start, end }	**InputError** when any of: u_id is not valid user start is greater than the total number of messages	Given u_id, return up to 50 messages between index "start" and "start + 50". Message with index 0 is the most recent message. This function returns a new index "end" which is the value of "start + 50", or, if this function has returned

					the least recent messages, returns -1 in "end" to indicate there are no more messages to load after this return.
user/friend/list	GET	(token)	{ user }	N/A	Returns a list of friends which contains information about their u_id, email, first name, last name, and handle.
user/friend/add	POST	(token, name_first, name_last)	{}	**InputError** when any of: name_first is not alphabetical name_last is not alphabetical	Sends a friend request to a user given their first name as name_first and last name as name_last.
user/friend/pending/list	GET	(token)	{ user }	N/A	Returns a list of pending friend requests which

					contains information about their user_id, email, first name, last name, and handle.
user/friend/pending/add	POST	(token, name_first, name_last)	{}	**InputError** when any of: name_first is not alphabetical name_last is not alphabetical name_first and name_last do not correspond to a pending friend request	Accepts a friend request of a user given their first name as name_first and last name as name_last
user/status/update	POST	(token, status)	{}	**InputError** when any of**:** Status is not a valid status type	Updates the current users status given a status parameter.

[Design] Conceptual Modelling (State)

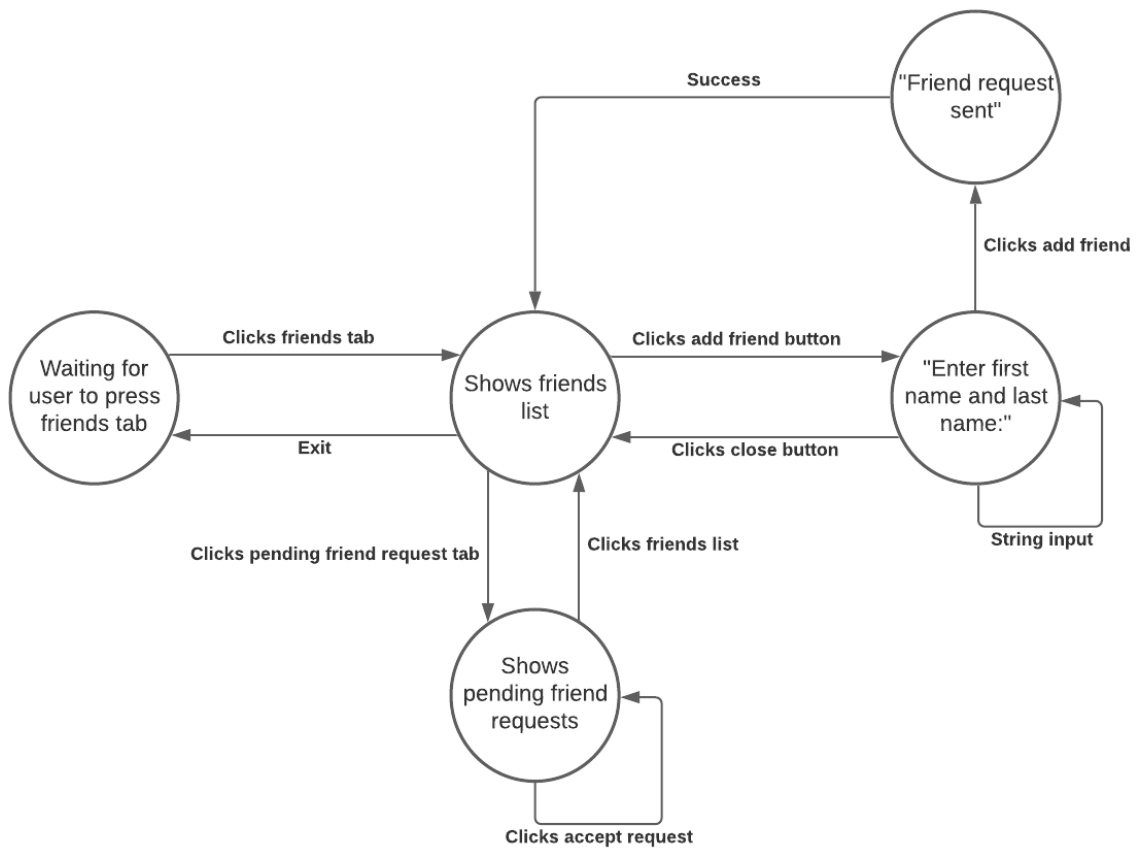**State diagram for private message feature**
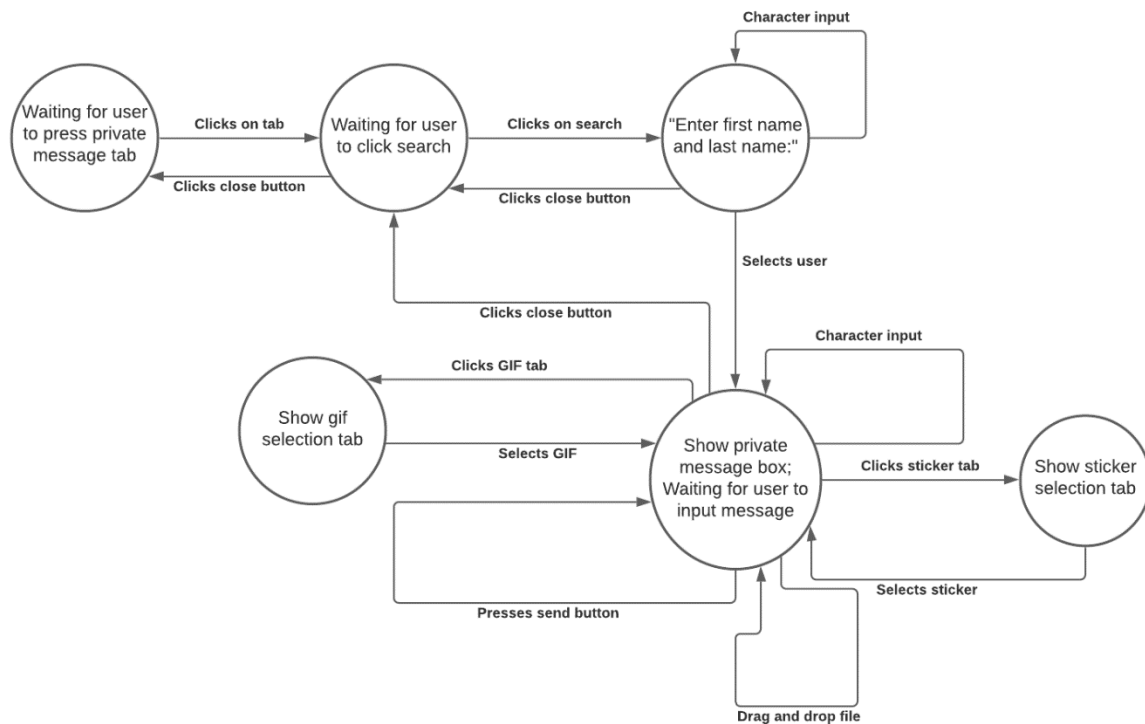
State diagram for user status feature
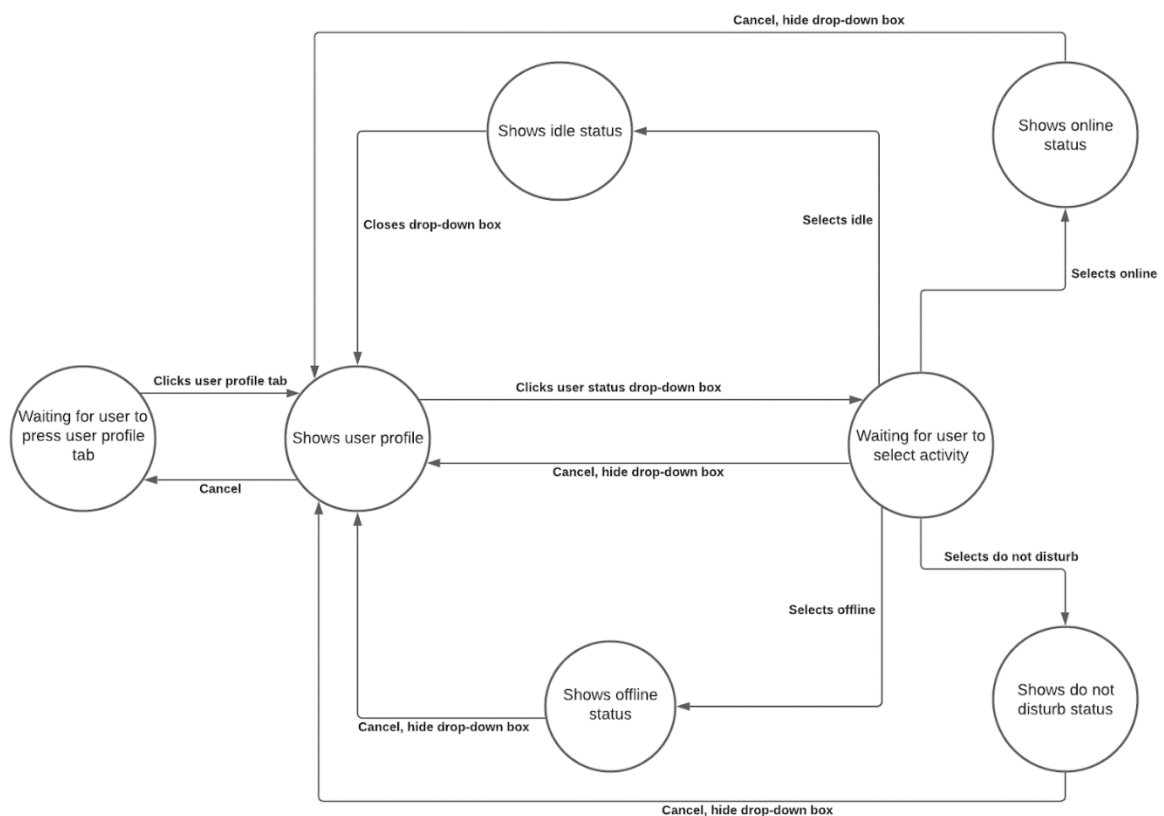
State diagram for adding friends