COMP1531

Next Stage: Requirements & Design

Team 5

1. Abstract

Software development is an iterative process - we're never truly finished. As we complete the development and testing of one feature, we're often then trying to understand the requirements and needs of our users to design the next set of features in our product.

For iteration 3 you are going to produce a short report in planning.pdf and place it in the repository. The contents of this report will be a simplified approach to understanding user problems, developing requirements, and doing some early designs.

2. Requirements

Planning for the next problems to solve involved developing and understanding a set of requirements that targeted users need, which can be used to design a new set of features for flockr.

2.1. Elicitation

The following series of questions was generated to interview target users and understand what problems they might have with teamwork-driven communication tools that are currently missing/unsolved by flockr.

1. What is your name and email address?
2. What messaging apps do you use most frequently for teamwork-driven communication?
3. How often do you use messaging apps?
4. What do you like about this app?
5. How many people and groups have you messaged last week?
6. Why do you use this messaging app (uni? work?)
7. What features do you use most frequently? (calling, messaging, reacting...)
8. What are some of the challenges you encounter when you're working with a team?
9. Does the messaging app cause/contribute to the challenges you face?
10. Does the messaging app help solve some of the challenges? If so, how? If not what can be done?
11. What don't you like? What would you change?
12. How reliable is this messaging app (do you experience problem logging in? Or reading messages etc)

Person 1: Dev Saldanha (devsaldanha@gmail.com)

Frequently used messaging apps:	"Facebook Messenger and sometimes Discord"
How often?	"Almost every day"
What do you like about this app?	" Messenger gives good expression through reactions and stickers, and it's fast and responsive so I know when people are available to talk and aren't. It also gives me quick access to voice and video calls to groups meaning I can easily talk when needed. Discord is even better, offering all that alongside more useful work functions like easy file sharing, screen sharing and coding languages."
How many people and groups have you messaged last week?	"Estimate at least 15"
Why do you use this messaging app?	"University, work and friends"
What feature do you use most frequently?	"Messaging and reacting, and on discord file sharing is really handy as well"
Challenges you encounter when you're working with a team?	"People never get their jobs done. It's so annoying I end up doing all the work and still get a C"
Does the messaging app cause/contribute to the challenges you face?	"Not entirely but because we are doing it at a distance there's no way for me to keep track of the work they're doing"
Does the messaging app help solve some of the challenges? If so, how? If not what can be done?	"No, but a provided measure of work completed or a system by which to easily and consistently check in with people would make it a lot easier. "
What don't you like? What would you change?	"The option to not appear as online and not tell people when I've read messages would be nice. "
How reliable is this messaging app (do you experience problem logging in? Or reading messages etc)	"Both apps are usually super reliable! never really experienced many issues but my friends messenger does get hacked quite often"

Person 2: Kent Thomson (kentj@live.com.au)

Frequently used messaging apps:	"Facebook Messenger"
How often?	" 2-3 time a week"
What do you like about this app?	" Easy group chats with a reply function.."
How many people and groups have you messaged last week?	"Within last week: 0 Before last week: 1 (physics lab ended last week)"
Why do you use this messaging app?	"Uni Physics Lab"
What feature do you use most frequently?	"Messaging"
Challenges you encounter when you're working with a team?	"It can be a little hard to follow several conversations within a larger group"
Does the messaging app cause/contribute to the challenges you face?	"The single chat group format of Facebook Messenger doesn't help this issue. "
Does the messaging app help solve some of the challenges? If so, how? If not what can be done?	"The reply feature somewhat helps this issue, but doesn't solve it completely "
What don't you like? What would you change?	"Adding a subgroup feature in addition to the main group chat would solve this issue."
How reliable is this messaging app (do you experience problem logging in? Or reading messages etc)	"There aren't any issues present in my experience with the app"

2.2. Analysis & Specification - User story and acceptance criteria

User Story:	**Acceptance criteria:**
As an user of a messaging app used for group work, I would like to be able to keep track of the tasks that the group is completing, and who is in charge of each task, so that we can assign tasks fairly.	Ensure that our channel functions allows users to 1. Add tasks to the channel checklist 2. Ability to edit checklist 3. Assign people to tasks on the checklist 4. Update progress on task
User story: As an of a messaging group, I want to make sure that I can keep track of different conversations that is happening in the same group, so that I am up to date with everything that is going on.	Ensure that our channel and message functions allows users to 1. Reply to a message 2. Being able to see the message which a message was replying to

2.3. Analysis & Specification - Use Cases

The responses from the elicitation were consolidated and documented as user stories. User Acceptance Criteria were added so that we will have a clear definition of when a story has been completed

Use case: Adding tasks to a checklist
Goal in context: User needs to have a checklist of tasks, assign tasks, and update progress for effective teamwork
Scope: Flockr frontend and backend
Precondition: User needs a flocker account, and needs to be member of at least one channel
Success end condition: A checklist of tasks is created, and members of a channel has the ability to assign tasks and update status
Failed end condition: The user cannot create a task, maybe because they are not a part of the channel, or does not have an account
Primary actor: User
Trigger: User clicks on create task

1. User registers and login to flockr
2. User joins a channel
3. User calls a add_task function, and passes in token, channel_id, and the name of task
4. Backend checks that the user is valid, and is in the channel, assign a task_id and add task to checklist
5. Any users in the channel can edit status or assign task by clicking the corresponding buttons
6. Backend is called, and received input with token, channel_id, task_id, and new status, or u_id for the person who the task was assigned to
7. Data.py is edited with the new information, and information is displayed on frontend
8. Users can now access the updated checklist, and make further changes when needed.

Use case: Reply to messages and view chain of replies
Goal in contect: User wants to keep track of separate conversations going on in bigger group chats
Scope: Flockr frontend and backend
Precondition: User needs a flocker account, and need to be member of at least one channel with other users
Success end condition: User can reply to messages, and can also view change of replies
Fail end condition: The user cannot reply to messages, maybe because there aren't any messages in the channel, or they are not in a channel
Primary actor: User
Trigger: user clicks on a message reply button to reply to a particular message

1. User registers and login to flockr
2. User joins a channel
3. User take part in a conversation by clicking message_reply to another messages
4. The user's token, channel_id, message, and the message_id of the message that the user is replying to is sent to the backend
5. Backend checks that the user is valid, logged in, and is a member of the channel. It also checks that the message_id is valid
6. A new message_id is created for the user's new message, and a link is created between the new message, and the message it was replying to
7. The information is send to data.py
8. The new message is displayed on the channel, and frontend also shows which message it is replying to

2.4. Validation

With the completed use case work, we reach out to the people we interviewed originally and inquire as to the extent to which these use cases would adequately describe the problem they're trying to solve.

Person 1: It describes my problem, and a task list can definitely be used to keep track of everyone's progress.

Person 2: A reply function could be helpful, but it could still be an issue if I can't see the chain of conversation/replies.

3. Design

3.1. Interface Design

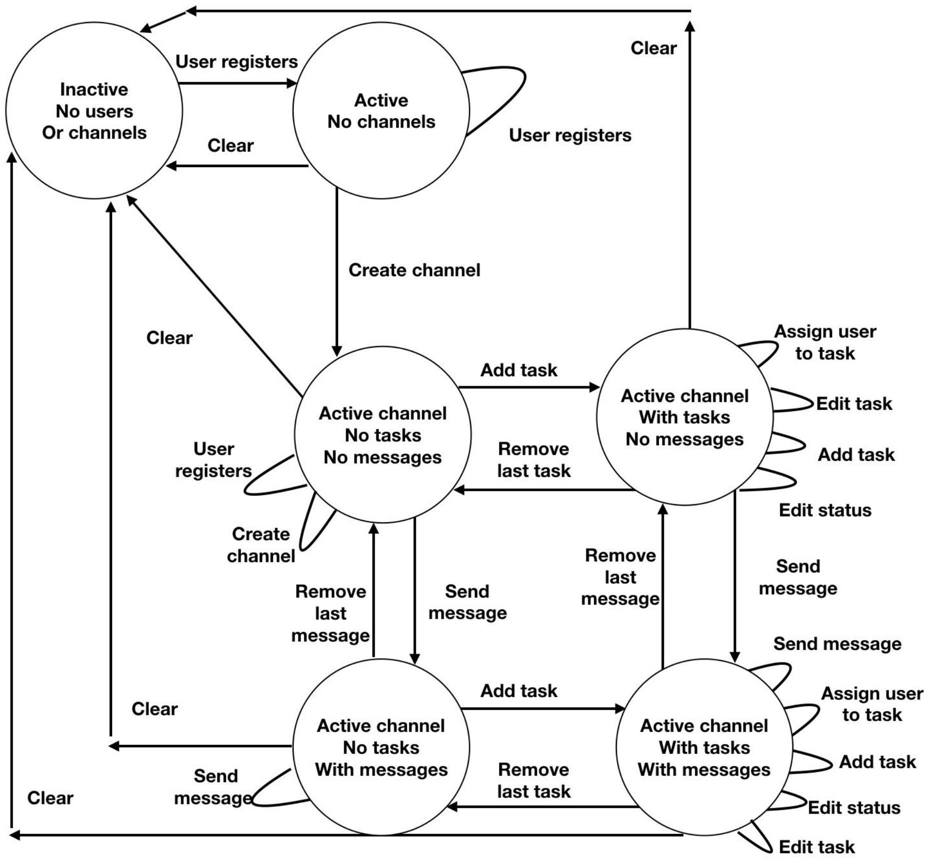
Function Name	**HTTP Method**	**Parameters**	**Return Type**	**Exceptions**	**Description**
checklist/addtask	POST	(token, channel_id, task_name)	{ task_id }	**InputError** when any of: ● channel_id does not refer to a valid channel ● task_name is more than 50 characters long	Given a channel_id of a valid channel, adds a task to the checklist
checklist/removetask	DELETE	(token, channel_id, task_id)	{}	**InputError** when any of: ● channel_id does not refer to a valid channel ● task (based on the task_id) does not exist	Given a channel_id of a valid channel and a task_id, removes the task from the checklist
checklist/editstatus	PUT	(token, channel_id, task_id, status_id)	{}	**InputError** when any of: ● channel_id does not refer to a valid channel ● task (based on the task_id) does not exist ● status_id is not a valid Status ID.	Given a task within a channel that the user is part of, set the status of the task Status ID: 0 - uncomplete 1 - ~~completed~~

checklist/assignuser	POST	(token, channel_id, task_id, u_id)	{}	**InputError** when any of: <ul><li>channel_id does not refer to a valid channel</li><li>task (based on the task_id) does not exist</li><li>u_id does not refer to a valid user within the same channel</li></ul>	Given a task within a channel, allows the authorised user to assign tasks to another user that is part of the same channel
checklist/edittask	PUT	(token, channel_id, task_id, task_name)	{}	**InputError** when any of: <ul><li>channel_id does not refer to a valid channel</li><li>task (based on the task_id) does not exist</li><li>task_name is more than 50 characters long</li></ul>	Given a task within a channel, allows the authorised user to edit and update the task text. If the new task_name is an empty string, the task is deleted.
message/reply	POST	(token, channel_id, message_id, message_reply)	{message_id}	**InputError** when any of: <ul><li>message_reply is more than 1000 characters</li><li>message_id is not a valid message</li></ul> **AccessError** when: <ul><li>the authorised user has not joined the channel they are trying to reply to</li></ul>	Given a message_id within a channel, allows a user to send a reply to the message associated with that message_id. Note: message_id returned is the id of the reply message.

New Data Types

Variable Name	**Type**
Named exactly **task**	List of dictionaries within a channel where each dictionary contains types { task_id, task_name, status, users_assigned }
Named exactly **users_assigned**	List of dictionaries within a task where each dictionary contains types { u_id, name_first, name_last, profile_img_url }
Named exactly **status_id**	Integer: - 0 represents a state where the task that has not been completed, while - 1 represents a task that has been completed (displayed on the frontend as task_name with a strikethrough e.g. "~~example task completed~~")

3.2. *Conceptual Modelling (State)*


Inactive No users Or channels
User registers
Active No channels
User registers
Clear
Clear
Create channel
Clear
Active channel No tasks No messages
Add task
Active channel With tasks No messages
Assign user to task
Edit task
Add task
Edit status
Remove last task
User registers
Create channel
Remove last message
Send message
Remove last message
Send message
Send message
Active channel No tasks With messages
Add task
Active channel With tasks With messages
Assign user to task
Add task
Edit status
Edit task
Remove last task
Clear
Send message
Clear