Elicitation

Develop a series of questions to ask these target users to understand what *problems* they might have with teamwork-driven communication tools that are currently unsolved by Streams. Give these questions to your target users and record their answers.

Current tools that are available in Streams

- DM functionality;
- Create private and public channels;
- Inviting users to channels
- Join channels
- Editing messages
- Removing messages
- Change some aspects of profile

Questions to ask

- Which platform do you currently use to communicate in a team?
  - Of these platforms, what functionality do you **need** most?
- What features do you wish to have on a platform like Streams?
  - What will you use each of these features for?
- What problems do you encounter using platforms like MS Teams, or discord or Facebook? And why?
  - List these problems in the order of what you would like fixed first

Interviewee 1

Name: Robert Huang → Student in Economics

Email:

- Which platform do you currently use to communicate in a team?

Only used MS teams for any group work from university. Uses discord for any informal team work like playing games or helping friends with their hw.

  - Of these platforms, what functionality do you **need** most?

Voice comms, much easier to give and get info.

- What features or abilities do you wish to have on a platform like Streams?

Voice comms, file sending, screenshot and picture sending

  - What will you use each of these features for?

Voice comms: Easy communication and more succinct, Filesharing: Actually send what you are working on to work together: Sending pictures: Send graphs and diagrams.

- What problems do you encounter using platforms like MS Teams, or discord or Facebook? And why are they problems?

Hard to navigate on MS teams: Puts him off using MS teams, not user friendly. Can't send large files and documents on discord, and therefore harder to work collaboratively.

  - List these problems in the order of what you would like fixed first

Send files, and then fix ease of navigation

Interviewee 2

Name: Bishoy Ibrahim → Student in Mechanical Engineering

Email:

- Which platform have you used to communicate in a team?

Messenger and MS teams

  - Of these platforms, what functionality do you **need** most?

Send messages to each other

- What features do you wish to have on a platform like Streams?

Send files to other people, send pictures such as screen shots. Link more information about yourself in profile like discord @, or Facebook @.

  - What will you use each of these features for?

Send files and screenshots allow for better collaboration, and pictures can show physical models. Linking can let him talk to other people outside of project

- What problems do you encounter using platforms like MS Teams, or discord? And why?

MS teams is hard to work collaboratively because the user interface is bad; On messenger, hard to keep focused on one thing when everyone talks about something different, can't share screen which makes it harder to help other teammates.

  - List these problems in the order of what you would like fixed first

Screen share, then have a better user interface.

Both Robert and Bishoy have problems revolving around communicating in media other than words. This may be resolved by adding file sharing, picture sharing and screen sharing functionality. Bishoy in particular, also has issues in keeping the topic in chatrooms focused which may be resolved by creating a list board.

Analysis and Specification

As a student in Engineering, I want to have a task board in the chatroom so that topics are not skipped over.

- The task board is placed on the top right of the channel
- The task board is always visible when typing
- Only 8 things are on the task board at one time
- The task name is under 40 characters.
- You can finish a task by clicking an X next to the individual task field
- You can add a task by typing in an empty task field

As a student in Mechanical Engineering, I want to send pictures so that my team can receive pictures of physical models.

- There is a “send picture” button on the end of any send message field
- The button can be pressed to open the users file system
- The user can then choose an image file and send it
- The user cannot send a non-image file
- The image will appear after the last message sent

As a student, I want to link my other socials on my profile page so that my team can communicate with me outside the team project.

- The profile page will have an extra field named “About me” below the handle
- The field will accept up to 400 characters
- The field will have a default message “Get your snags at Bunnings”
- The field will accept emojis
- The About me field can be typed like a normal text file
- The default message can be edited like normal text file

As a student in Engineering, I want to share screens so that my team can collaborate in real time.

- There is a “share screen” button on the top left, the clicking of which starts a stream of your screen in the channel the user is in
- A green text appears in the centre top of the screen reading “You are screen sharing”
- The viewer can see the entirety of the screen shared in real time
- There is a red X button on the bottom right corner after the screen is being shared which terminates the screen sharing if pressed
- The green text disappears after screen sharing is terminated

As a student in Economics, I want to share documents so that I can keep my team up to date with my work.

- There is a “send file” button to the right of any send message field
- The button can be pressed to open the users file system
- The user can then choose a documents file which sends
- The file appears after the last message sent
- The user cannot select a non-document file

As a student in Economics, I want to have voice communication so that I can explain complicated ideas to my team more effectively

- There is a button named “Voice” on the right of any send message field

- The clicking of the button starts recording
- The button turns red
- The clicking of the red button ends the voice recording and the button turns yellow
- The clicking of the yellow button plays the voice recording back
- Pressing enter sends the voice recording and the button turns back to default colour
- The voice recording will be a button named “voice recording” posted after the last message sent
- Pressing the button will play the recording and pressing it again will pause the recording
- The voice recordings have a max limit of 15 minutes before automatically ending
- A blue opaque timer accurate to seconds appears on the top left when recording begins

As a student in Economics, I want to have picture sending support so that my team can receive my graphs and diagrams.

- There is a “send picture” button on the end of any send message field
- The button can be pressed to open the users file system
- The user can then choose an image file and send it
- The user cannot send a non-image file
- The image will appear after the last message sent

Use-Case background

- Use Case: Send and surmise an essay
- Goal in Context: Economics students write very large essays that need to be sent to each other to ensure everyone is up-to-date with relevant information. These files can be long, so a voice recording can sum up what is in the essay quickly and effectively.
- Scope: The backend datastore, and back end functions to send the file and voice recording.
- Level: Primary task
- Preconditions: The user has an account in Streams, is in a Channel or is in a DM. The other backend and front-end infrastructure is complete.
- Success end condition: There is a text document and voice recording in the messages sent by the user.
- Failed end condition: There is no text document and voice recording in the messages sent by the user.
- Primary actor: User
- Trigger: User clicks the “send file” button, and clicks the “Voice” button afterwards.

Use-Case written form

MAIN SUCCESS SCENARIO

Step 1. User clicks the “send file” button

Step 2. The backend opens the file system

Step 3. User finds the file on his file system

Step 4. User sends the file from his file system

Step 5. The backend checks that it is a document file being sent

Step 6. The backend sends the file to the messages

Step 7. User clicks the “Voice” button

Step 8. User records his message before 15 mins

Step 9. The backend continually records the message and stores it

Step 10. User clicks the red voice button

Step 11. The backend stops recording

Step 12. The user presses enter

Step 13. The backend sends the voice recording to the messages

Validation

Robert Huang

Q:

On a scale from 1 – 10 how well does this use case describe the problem you are trying to solve? And explain how you reached this score.

A:

8 / 10, The file system is perfect, does exactly what is wanted. Did not want a voice recording system but a live call as “no one uses voice recordings” however “if people used it, it probably gets the job done”. Rated it 5/5 for file system issue and 3/5 for voice recording.

Bishoy Ibrahim

Q:

On a scale from 1 – 10 how well does this use case describe the problem you are trying to solve? And explain how you reached this score.

A:

7/10, The file system is “basically what I would need to send files” and the voice recordings are “useful but I want real live chats and screen records because I don’t want to listen to a recording or my own voice”. Rated it 7/10 because his “gut feeling felt right at 7”.

Interface Design

Name and Description	**HTTP Method**	**Data Types**	**Exceptions**

message/fetchfile/v1
Attaches a file to the next message sent by the user of a channel into the Streams chat. Can be used for either a dm or channel.
	POST	**Parameters:**
		{token, loc_id, file, is_dm}
		Return Type:
		{}
			InputError when:
			- channel_id does not refer to a valid channel
			- the given file is not a valid document type (pdf, docx, etc)
			AccessError when:
			- token is invalid
			- the authorised user is not a member of a validated channel_id

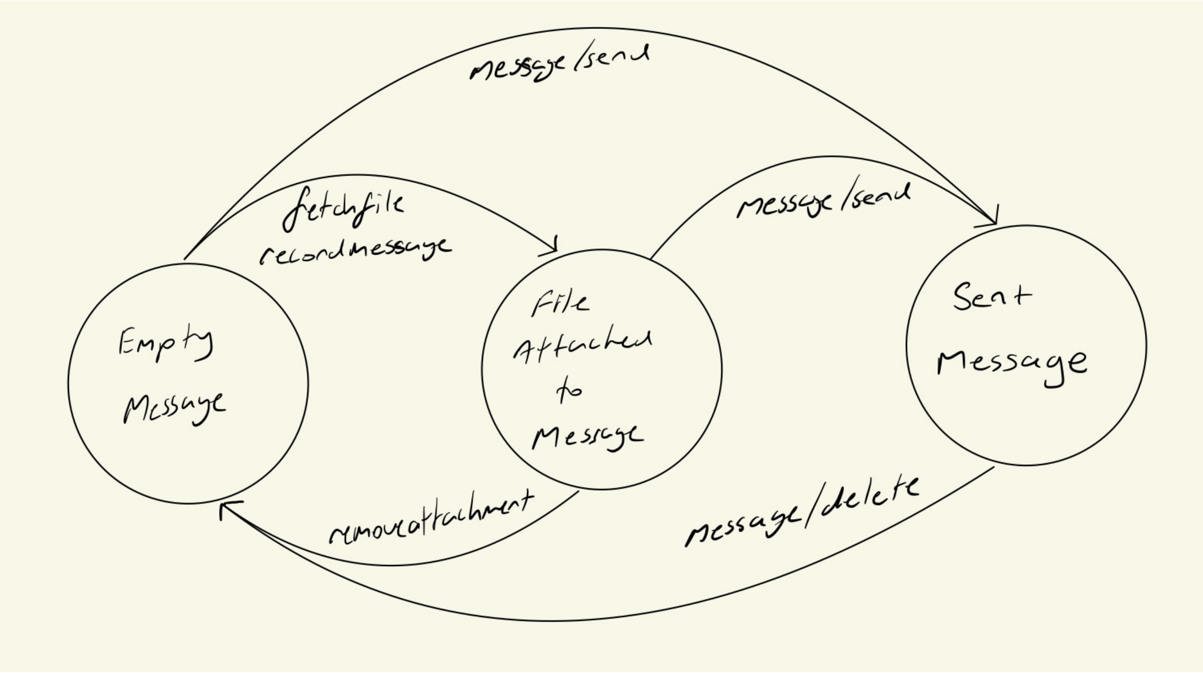
message/recordmessage/v1
Records a message for up to 15 minutes, with this file then being attached to the next sent message.
	POST	**Parameters:**
		{token, loc_id, file, is_dm}
		Return Type:
		{ }
			InputError when:
			- channel_id does not refer to a valid channel
			AccessError when:
			- token is invalid

			the authorised user is not a member of a validated channel_id

message/removeattachment/v1
Removes the currently attached file with name file_name, from the next sent message.
	GET	**Parameters:**
		{token, loc_id, file_name, is_dm}
		Return Type:
		{}
			InputError when:
			- channel_id does not refer to a valid channel
			AccessError when:
			- token is invalid
			the authorised user is not a member of a validated channel_id

Conceptual Modelling (State)

Messages State Diagram:


message/send
fetchfile
recordmessage
message/send
Empty Message
File Attached to Message
Sent Message
removeattachment
message/delete