COMP1531 Project Planning

By T13B AERO: Michael, Raymond, Anthony, Cara and Mohique

Elicitation

Find 2-3 people to interview as target users. Target users are people who currently use a tool like ***Dreams****, or intend to. Collect their name and email address.*

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 ***Dreams****. Give these questions to your target users and record their answers.*

Questions

1. What is your reasoning for wanting to use a teamwork-driven communication such as Dreams?
2. What are features you consider mandatory when using communication tools such as Dreams?
3. What are use cases and features you find unsatisfactory with communication tools you currently use and would appreciate in Dreams?

Target Users

Name	**Email**	**Specialisation/Profession**
Steven Stevenson	**Stevenson@gmail.com**	**Computer science student**
John Smith	**J.smith@gmail.com**	**Teacher**
Emily Truong	**Emily.T@gmail.com**	**Team Manager**

Target User	**Answers**
Steven	I mainly use these tools to talk to my friends but when I have to do group projects for my computer science course I would also use them to communicate with my group members. Being able to create multiple channels with different users and send private messages. Most apps I use don’t have many inbuilt tools for organisation and teamwork such as tasks

	and to-do lists which I feel would be very useful when working on group projects.
John	I’m a school teacher so it would be great to use Dreams as a mode of communication between my students and colleagues at all times. Having different channels for my classes and staff as well as being able to moderate behaviour of students in said classes and communicate with them privately if needed. I find it’s harder to find tools which are easy to use as communication apps and have great organization and task oriented tools. Apps such as google classroom have great management tools but are difficult to use as a communication app, so being able to give students work in Dreams would be nice.
Emily	I would like a tool that allows me to manage my employees as well as allowing them to work together as effective teamwork is crucial to our overall performance. Features I consider mandatory are platforms to send messages and tools to create tasks and organise my team. I would like to be able to create channels specifically for certain tasks and have to-do lists or tasks which are intrinsic to those channels to help create a more organised environment.

[Requirements] Analysis & Specification - Use Cases

Once you've elicited this information, it's time to consolidate it.

Take the responses from the elicitation and express these requirements as ***User Stories****. Document these user stories. For each user story, add User Acceptance Criteria as notes so that you have a clear definition of when a story has been completed.*

Once documented, generate at least one use case that attempts to tell a story of a solution that satisfies the requirements elicited. You can generate a visual diagram or a more written-recipe style, as per lectures.

Steven

- As a University student I want to be able to have casual chats with my friends
    - Given my friends and I are Dreams users, When I send private messages to my friends I expect them to be able to view it
- As a university student I want to be able to communicate with group members for group projects and create an organised list of tasks so our group is efficient
    - Given my group members and I are Dreams users, I expect a to-do list I create in a channel to be visible to them and any changes a user makes to the tasks to also be visible to each member of the channel

John

- As a teacher I want to be able to send messages both public and privately to students and colleagues.
    - Given that a student/colleague are registered on Dreams, when I send private messages to them, I expect that only we are able to see those messages
    - Given that students/colleagues are registered Dreams, when I send messages to a public channel, I expect that everyone who has joined that channel is able to see those messages.
- As a teacher, I have to manage many groups of students so I want to be able to allocate tasks and to do lists for groups and individual students
    - Given that my students and I are registered on Dreams, when I create or modify a to-do list or task in a channel, I expect members of the channel to see the change and view the tasks.
- As a teacher, I have to manage poorly behaving students by moderating their actions in a channel
    - Given that my Students and I are users of Dreams, and that I am the owner of the channel, I want to be able to remove offensive messages created by students, and prevent students from messaging for certain amounts of time

Emily

- As a team manager I want to be able to send updates in the form of messages to all the members of my team in a channel
  - Given the members of my team and I are Dreams users, and I send a message in a channel we are all in, I expect each member to be able to view the message and reply to it
- As a team manager, I want to be able to create channels centered around certain tasks and have inbuilt lists to display the tasks to the team members
  - Given my team members are Dreams users, I want to be able to create multiple channels which they are able to join, inside these channels I want to be able to allocate

Use Cases

Use case 1: Create a task list for members in a channel.
Goal in context: To allow users to organise themselves around a task and view progress of a task in a project.
Scope: Dreams, channel.
Level: Subfunction .
Precondition: All users are members of the channel and are registered in Dreams.
Success End Condition: A list is pinned to the channel showing all users the tasks inputted.
Failed End Condition: List is not pinned, list does not contain tasks, list is not available to all channel members.
Primary Actor: Channel member
Trigger: Task list is created in channel

Use case 2: Change a channel members permissions
Goal in context: To allow channel owners to deny or allow certain privileges to other channel members
Scope: Dreams, channel.
Level: Subfunction .
Precondition: User is an owner of a channel with registered Dreams users as members.
Success End Condition: The targeted users ability to send messages is either removed or allowed.
Failed End Condition: Members permission doesn’t change, members who aren't owners receive access to this function.
Primary Actor: Channel owner
Trigger: User is muted/unmuted in channel.

[Requirements] Validation

With your completed use case work, reach out to the 2-3 people you interviewed originally and inquire as to the extent to which these use cases would adequately describe the problem they're trying to solve. Ask them for a comment on this, and record their comments in the PDF.

Steven

Being able to create to-do lists in a channel I'm in for a group project would help us to stay on track with our assignments and to see if any members are falling behind or slacking off in the group. Having these alongside a solid messaging app would make this really appealing for me as an alternative to the usual messenger for group projects which is comparatively lacking in functionality.

John

This seems perfect! Being able to make channels around lists seems like a great way to encourage my students to stay organised when I place them into groups for projects as they would have all the tools needed to do so and I would be able to write sample tasks to help lost students along. The ability to mute members also seems incredibly useful if some students are misbehaving in the channels as I could just stop them from posting disruptive messages and talk to them privately before allowing them to talk again.

Emily

The second feature doesn't appeal to my needs much, however the first use case is more applicable to my line of work. Being able to pin a list of tasks to a text channel is a feature I find quite valuable as it would remove some unnecessary communication regarding what is required of certain teams as it would be simply displayed to them and they can instead focus on completing the work or bringing up issues which move the project forward. That said I would still want a few more features, namely being able to allocate specific tasks in the lists to individuals and being able to view more details about the tasks such as comments the allocated person has left regarding their process. Aside from that this application looks promising and I look forward to seeing it grow.

[Design] Interface Design

Now that we've established our problem (described as requirements), it's time to think about our solution in terms of what capabilities would be necessary. You will specify these capabilities as HTTP endpoints, similar to what is described in 6.2. There is no minimum or maximum of what is needed - it will depend on what problem you're solving.

Data types are the same as what is describer in 6.2 with the addition of a new Tasks data type

Tasks
List of dictionaries where each contains types {task_id, task_name, due_timestamp, assignees, notes}

- task_id is an integer unique to a certain task
- task_name is a string which holds the name of the task
- due_timestamp is datetime object which shows the due date of the task and can be left empty to indicate no due date
- assignees is a list of user ids which are allocated to work on the task
- notes is a string containing comments made by creator or assignees

In addition to this a boolean 'is_muted' would have to be added to the user dict in channels['all_members']. This boolean would need to be checked when a user is sending a message to determine whether or not they can send

<u>Endpoints</u>

Name and description	**HTTP method**	**Data types**	**Exceptions**
todo/createtask Adds a task with task name to channel. Due_timestamp, assignees and notes are optional fields which can be left empty	POST	**Parameters:** (token, channel_id, task_name, due_timestamp, assignees, notes) **Return Type:** { task_id }	**InputError** when any of: ● Channel ID is not a valid channel ● Task_name is empty ● Task_name is greater than 1000 characters ● Due_timestamp is in the past ● Assignees includes users not in channel **AccessError** when any of: ● Authorised user is not a member of channel with channel_id
todo/edittask Changes data of a specified task	PUT	**Parameters:** (token, task_id, task_name, due_timestamp, assignees, notes) **Return Type:**	**InputError** when any of: ● Task_id is not a valid task ● Task_name is greater than 1000 characters ● Due_timestamp is in the past ● Assignees includes users not in

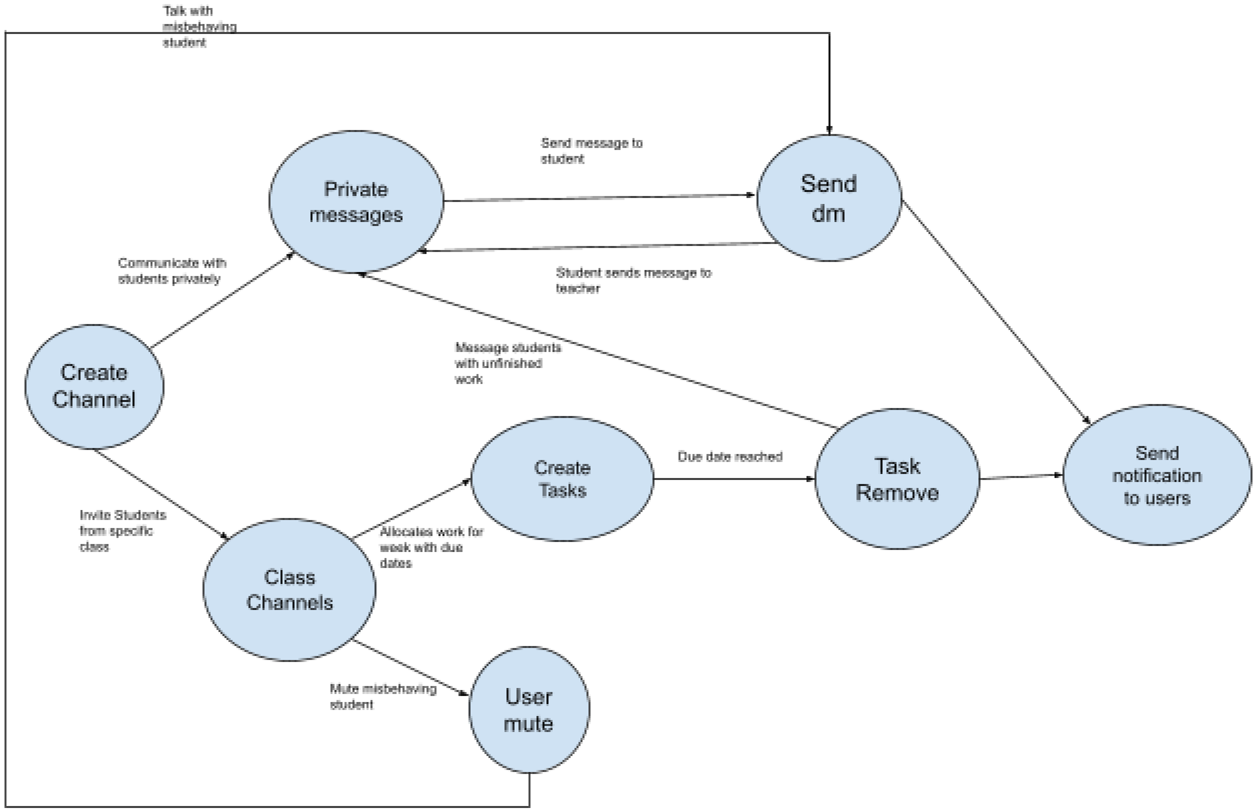
		{}	channel **AccessError** when any of: ● Authorised user is not a member of channel with channel_id
todo/listtasks Returns the list of tasks for the requested channel	GET	**Parameters:** (token, channel_id) **Return Type:** {Tasks}	**InputError** when any of: ● Channel ID is not a valid channel **AccessError** when any of: ● Authorised user is not a member of channel with channel_id
todo/removetask Removes task from channels task list	DELETE	**Parameters:** (token, task_id) **Return Type:** {}	**InputError** when any of: ● Task_id is not a valid task **AccessError** when any of: ● Authorised user is not a member of channel with channel_id
channel/usermute Sets the is_muted boolean in the target users dict in channel to False so they can no longer send messages in that channel	PUT	**Parameters:** (token, channel_id, user_id) **Return Type:** {}	**InputError** when any of: ● channel_id is not a valid channel ● User_id is not a member of channel with channel_id ● User_id is an owner of the channel with channel_id ● User_id indicates a user which is already muted **AccessError** when any of: ● Authorised user is not a member of channel with channel_id ● Authorised user is not an owner of channel with channel_id
channel/userunmute Sets the is_muted boolean in the target users dict in channel to True so	PUT	**Parameters:** (token, channel_id, user_id)	**InputError** when any of: ● channel_id is not a valid channel ● User_id is not a member of

they can send messages in that channel		**Return Type:** {}	channel with channel_id - User_id is an owner of the channel with channel_id - User_id indicates a user which is already unmuted **AccessError** when any of: - Authorised user is not a member of channel with channel_id - Authorised user is not an owner of channel with channel_id

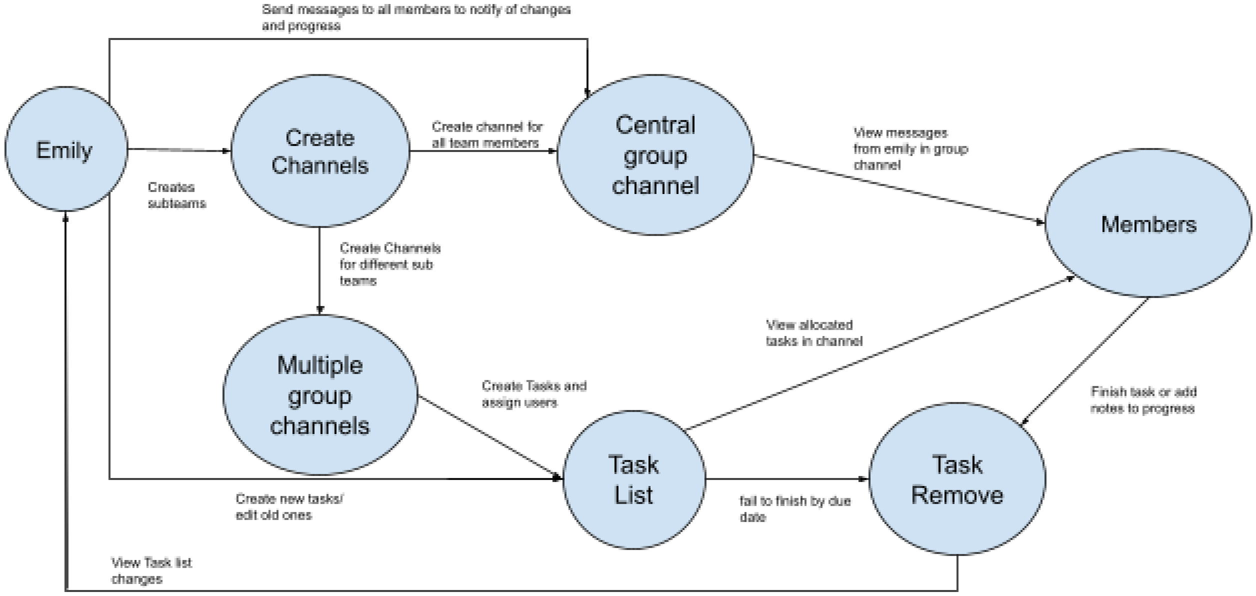
[Design] Conceptual Modelling (State)

Now that you have a sense of the problem to solve, and what capabilities you will need to provide to solve it, add at least one state diagram to your PDF to show how the state of the application would change based on user actions. The aim of this diagram is how to a developer understand the different states the user or application.

John’s State Diagram


Talk with misbehaving student
Send message to student
Private messages
Send dm
Communicate with students privately
Student sends message to teacher
Create Channel
Message students with unfinished work
Create Tasks
Due date reached
Task Remove
Send notification to users
Invite Students from specific class
Allocates work for week with due dates
Class Channels
Mute misbehaving student
User mute

Emily’s State Diagram


Send messages to all members to notify of changes and progress
Emily
Create Channels
Central group channel
Members
Creates subteams
Create channel for all team members
View messages from emily in group channel
Create Channels for different sub teams
Multiple group channels
View allocated tasks in channel
Create Tasks and assign users
Finish task or add notes to progress
Task List
Task Remove
Create new tasks/ edit old ones
fail to finish by due date
View Task list changes