T11B-AERO Beans Future Features Planning

Akhil Govan, Justin Roxas, Isaac Chang, Jerald Jose, Marie Clemente

REQUIREMENTS

ELICITATION

INTERVIEW QUESTIONS

Note: Interviewees were given the link to the example implementation (http://treats-unsw.herokuapp.com/) in order to explore the current features of Beans.

1. Which communication platform do you prefer the most for communicating with a team?
    a. Discord
    b. Microsoft Teams
    c. Zoom
    d. Google Meet
    e. Other:
2. Which specific features of your chosen platform do you find most useful?
3. Which features do you think are essential for a teamwork-driven communication platform?
    a. voice channels
    b. text channels
    c. private chats/dms
    d. @mention other users
    e. video calls
    f. screen share in video calls
    g. shared calendar
    h. Other:
4. Are there any problems/issues that you have found with using your chosen platform?
5. What functional limitations (if any) did you encounter using the given example implementation?

RESPONSES

Interviewee: Patrice Clemente
Email: patriceclemente97@gmail.com
Answers:

1. Zoom
2. schedule meetings, annotate screen share, private message in video call chat
3. text channels, private chats/dms, @mention other users, video calls, screen share in video calls
4. getting kicked out of a meeting after a certain time limit, cannot set online status
5. not enough reacts, cannot search for messages from a given user, no shared calendar/meeting reminders, cannot reply to specific messages

Interviewee: Josh Lim
Email: josh.lim@student.unsw.edu.au
Answers:

1. Microsoft Teams
2. permanent chat history, calendar view
3. text channels, private chats/dms, video calls
4. None
5. lacking chat archiving, no features for collaborative brainstorming and work

Interviewee: Vivian Truong
Email: vivitrng@gmail.com
Answers:

1. Google Meet
2. Scheduling meetings in a calendar
3. voice channels, shared calendar
4. Not a great way to access the actual meeting
5. I would appreciate a tutorial on how each component works and understanding the general UI. Search bar has no real functionality - making it more intuitive to use.

RESULTS

Identified Problems	Proposed Solutions
search bar has limited functionality	- search messages by sender - filter messages by channel - search for messages sent on a specific date - add search options to a dropdown menu to make use more intuitive
unable to reply to a specific message	- Add a button to reply to a specific message in a channel or dm. - Add a header to messages that are replying to another message. - Notify the user if any of their messages have been replied to.
no features for collaborative brainstorming and work	- collaborative document mode - whiteboard for brainstorming
cannot set/view online status	- in each channel add a small icon next to each member in the list signifying online status - on profile page, add a field allowing users to edit their online status(online, idle, busy, invisible)
not enough reacts	- instead of a like button, have a react button on each message that opens up a menu of potential reacts(like, dislike, heart, etc.)
cannot schedule standups	- implement a shared calendar tab in each channel/dm where the users with owner permissions can add scheduled standups - a link will also be added on the side menu where a user can view all their scheduled meetings from joined channels/dms - send meeting reminders as emails
not understanding how each feature works	- adding a question mark that, on hover, gives a short description of a feature - add tooltip that shows what each button does since icons are not clear

ANALYSIS & SPECIFICATION - USE CASES

USER STORIES

User Story: As a Beans user, I want to be able to access a white board feature to collaborate and brainstorm new ideas with my team.

Acceptance Criteria:

- Given that I have conceived a channel, I can create an empty whiteboard with a specified time limit and all members of the channel can draw on it until the time limit ends.
- Given that I have conceived a dm, I can create an empty whiteboard with a specified time limit and all members of the dm can draw on it until the time limit ends.
- I am able to view other members' contributions to the whiteboard live.
- Other members of the channel/dm can view my contributions to the whiteboard live.
- I can view the archived whiteboard as an image message in the channel after the time limit ends.

User Story: As a Beans user, I want to display my online status so that people know when I am available to help or chat to. I also want to view others' status so that I know if I can reach out to them to ask questions.

Acceptance Criteria:

- Given that I have logged in, I can click on my profile data and change my online status, one of the following options: online, offline, idle and invisible.
- Given that I have created or joined a channel, I can see other members' online status next to their name on the members list.
- Given that I have created or got added to a dm, I can see other members' online status next to their name on the members list.

User Story: As a Beans user, I want to be able to converse naturally with an indicator showing if someone is replying to a specific message in a channel or dm.

Acceptance Criteria:

- Given that I am in a channel or dm, I can click a reply button associated with a message.
- Given that I reply to a message, my reply will have a header showing the message I'm replying to.
- I want to receive a notification indicating that someone replied to my message.

User Story: As a Beans user, I want to be able schedule standups on a shared calendar within a channel so that I can check on my group's progress and discuss any problems they are having.

Acceptance Criteria:

- Given I'm in a channel, I can view a calendar page, displaying standups scheduled within the channel.
- Given I'm logged in, I am able to view a personal calendar with all scheduled standups across channels I am a part of.
- Given there are standups scheduled for a channel I'm in, I receive an email 5 minutes before the standup to remind me to attend.
- Given I have scheduled a standup, I am able to edit the note, start time and length of the scheduled standup.
- Given I have scheduled a standup, I am able to delete the scheduled standup.

USE CASES

Use Case: Create or edit a time limited live whiteboard within a channel or dm

Goal in context: A member of a channel or dm can create a whiteboard or edit an existing whiteboard within the specified time limit.

Scope: UNSW Beans

Level: Primary task

Preconditions: The users have registered and are logged-in, channel (or dm) created and users are members of the channel (or dm).

Success End Condition: A whiteboard exists in the channel or dm chat box, where when a member hovers over it in the chat, a series of tools (draw, add a text box and open in a new window) are displayed to make live edits to the whiteboard. After the specified time limit is over, all changes are saved and the final result is locked-in and displayed in the chat.

Failed End Condition: No whiteboard exists in the channel or dm.

Primary Actor: User in a channel or dm.

Trigger: Channel or dm member types '/whiteboard X', where X is the whiteboard editing window in seconds, in the chat box.

Success Scenario 1

1. User one joins a public channel.
2. User one creates a whiteboard that has an editing time window of 10 seconds by typing '/whiteboard 10' in the message box of the channel.
3. UNSW Beans displays a whiteboard in the chat box on the channel.
4. User two joins the same channel as user one and hovers their mouse over the whiteboard.
5. UNSW Beans displays an interface of tools (draw, add a text box and open in a new window) on the whiteboard as the user hovers over it.
6. User two clicks on the draw tool and draws a house on the whiteboard.
7. UNSW Beans displays these changes live on the whiteboard in the chat for all channel members to see.
8. Whiteboard's 10 second timer ends, all changes to whiteboard are locked-in and hovering the mouse over the whiteboard displays no tools.

Success Scenario 2

1. User one and user two are invited to a dm.
2. User one creates a whiteboard that has an editing time window of 20 seconds by typing '/whiteboard 20' in the message box of the dm.
3. User two hovers their mouse over the whiteboard.
4. UNSW Beans displays an interface of tools (draw, add a text box and open in a new window) on the whiteboard as the user hovers over it.
5. User two clicks on the draw tool and draws a star on the whiteboard.
6. UNSW Beans displays these changes live on the whiteboard in the chat for all dm members to see.
7. Whiteboard's 20 second timer ends, all changes to whiteboard are locked-in and hovering the mouse over the whiteboard displays no tools.

Use Case: Schedule a standup within a channel.

Goal in context: A member of a channel can edit a shared calendar within a channel to schedule a common meeting time to have a standup. Also, users of UNSW Beans can view all their scheduled standups for all their joined channels with a private calendar in their user profile.

Scope: UNSW Beans

Level: Primary task

Preconditions: The users have registered and are logged-in, channel created and users are members of the channel.

Success End Condition: Members of a channel can see a shared calendar with a scheduled standup on it, with a note, start time and end time. A user can see all their scheduled standups for all their joined channels with a private calendar in their user profile. A reminder is sent to all the channel members' emails 5 minutes before the standup start time.

Failed End Condition: Shared calendar in a channel has no scheduled standups and users' user profiles have private calendars that have no scheduled standups.

Primary Actor: User in a channel.

Trigger: Channel member clicks on a day on the shared calendar of the channel.

Success Scenario 1

1. Users clicks the 20th of November on the channel's shared calendar.
2. UNSW Beans displays a prompt to the user to add a note, start time (time within the day) and end time.
3. User types data into the prompt and then submits it.
4. UNSW Beans displays a coloured text box on the 20th of November, in the channel's shared calendar, with the scheduled standup's note, start time and end time.
5. UNSW Beans displays a coloured text box on the 20th of November, in the user's private calendar, with the scheduled standups note, start time and end time.
6. UNSW Beans sends a reminder to all the channel members emails 5 minutes before the standups start time.

Success Scenario 2

1. User clicks the 20th of November on the channel’s shared calendar.
2. UNSW Beans displays a prompt to the user to add a note, start time (time within the day) and end time.
3. User types data into the prompt and then submits it.
4. UNSW Beans displays a coloured text box on the 20th of November, in the channel's shared calendar, with the scheduled standups note, start time and end time.
5. User changes to a second joined channel.
6. User clicks the 21st of November on the channel’s shared calendar.
7. UNSW Beans displays a prompt to the user to add a note, start time and end time.
8. User types data into the prompt and then submits it.
9. UNSW Beans displays a coloured text box on the 21th of November, in the channel's shared calendar, with the scheduled standups note, start time and end time.
10. UNSW Beans displays a coloured text box on the 20th of November, in the user’s private calendar, with the scheduled standups note, start time and end time of the first channel, and another on the 21th of November with the scheduled standups note, start time and end time of the second channel.
11. UNSW Beans sends a reminder to all the channel members emails 5 minutes before the standups start time.

VALIDATION

Each interviewee was asked to comment on the extent to which their specific use case would adequately describe the problem they're trying to solve.

Interviewee: Patrice Clemente

Use Case: Schedule a standup within a channel.

Comments: The use case adequately describes my problem of not being able to schedule meetings. There are no additional features that I would like to include. I particularly like the personal calendar which displays all of a user's scheduled meetings.

Interviewee: Josh Lim

Use Case: Create or edit a time limited live whiteboard within a channel or dm

Comments: The provided use case outlines the problem well and gives a potential solution that is practical. Some more clarification would be nice on what happens to the whiteboard after the timer ends, and also how a user might interact with the whiteboard.

Note: Edits made to the first use case using these comments are highlighted in green.

Interviewee: Vivian Truong

Use Case: Schedule a standup within a channel.

Comments: The problem is sufficiently addressed by the given use case. The provided solution seems useful and intuitive.

DESIGN

Chosen Problem: Beans users are unable to set scheduled standups for group check ins.

Solution: Implementing a calendar which allows users to schedule standups in a channel as well as a personal calendar compiling a user's scheduled meetings throughout all channels they are a member of.

INTERFACE DESIGN

INPUT/OUTPUT TYPES

Variable Name	Type
has suffix **Id**	integer
length	integer
note	string
has prefix **time**	string (time in format hh:mm)
scheduledStandups	Array of objects, where each object contains types { channelId, uId, date, note, timeStart, timeEnd } where - channelId is the channel where the standup will occur - uId belongs to the user who created the scheduled standup - date is the date that the standup will occur - timeStart is the time the standup will begin - timeEnd is the time the standup will end
date	string (date in format dd-mm-yyyy)

INTERFACE

Note: All routes that require a token should raise a 403 Error when the token passed in is invalid.

Name & Description	HTTP Method	Data Types	Exceptions
standup/schedule/create/v1 Given a standup's desired channelId, date, note, timeStart and timeEnd, creates a scheduled standup for them and returns a unique scheduleId. An email is also sent stating that the standup was created. Another email will be sent to all channel members as a reminder 5 minutes before the meeting begins.	POST	**Query Parameters:** { channelId, date, note, timeStart, timeEnd } **Return type if no error:** { scheduleId }	**400 error if:** - channelId does not refer to a valid channel - given timeStart is later than given timeEnd - there is another standup currently scheduled in the same channel in the given time frame - date is invalid - note is not between 1 and 500 characters inclusive **403 error if:** - channelId is valid and the authorised user is not a member of the channel
standup/schedule/editnote/v1 Updates the note for the scheduled standup with the given scheduleId. An email is also sent to all channel members stating that the standup was updated.	PUT	**Query Parameters:** { scheduleId, note } **Return type if no error:** { }	**400 error if:** - scheduleId is invalid - note is not between 1 and 500 characters inclusive **403 error if:** - authorised user did not create the standup and does not have owner permissions in the channel
standup/schedule/editstart/v1 Updates the timeStart for the scheduled standup with the given scheduleId. An email is also sent to all channel members stating that the standup was updated.	PUT	**Query Parameters:** { scheduleId, timeStart } **Return type if no error:** { }	**400 error if:** - scheduleId is invalid - given timeStart is later than given timeEnd **403 error if:** - authorised user did not create the standup and does not have owner permissions in the channel

standup/schedule/editend/v1 Updates the timeEnd for the scheduled standup with the given scheduleId. An email is also sent to all channel members stating that the standup was updated.	PUT	**Query Parameters:** { scheduleId, end } **Return type if no error:** { }	**400 error if:** - scheduleId is invalid - given timeStart is later than given timeEnd **403 error if:** - authorised user did not create the standup and does not have owner permissions in the channel
standup/schedule/editdate/v1 Updates the date for the scheduled standup with the given scheduleId. An email is also sent to all channel members stating that the standup was updated.	PUT	**Query Parameters:** { scheduleId, date } **Return type if no error:** { }	**400 error if:** - scheduleId is invalid - date is invalid **403 error if:** - authorised user did not create the standup and does not have owner permissions in the channel
standup/schedule/remove/v1 Deletes the standup with a given standupId. An email is also sent to all channel members stating that the standup was cancelled.	DELETE	**Query Parameters:** (scheduleId) **Return type if no error:** { }	**400 error if:** **403 error if:** - authorised user did not create the standup and does not have owner permissions in the channel
channel/getschedule/v1 Returns all the scheduled standups that belong to the given channelId.	GET	**Query Parameters:** (channelId) **Return type if no error:** { scheduledStandups }	**400 error if:** - channelId is invalid **403 error if:** - authorised user is not a member of the channel
user/getschedule/v1 Returns all the scheduled standups from all of the channels the authorised user is a member of.	GET	**Query Parameters:** ( ) **Return type if no error:** { scheduledStandups }	N/A

IMPLEMENTATION WITH FRONT-END

Creating a new scheduled standup:

1. Channel calendar will be a front-end component, where each day can be clicked to schedule a new standup on that day.
2. Authorised user will input start time, end time and a note.
3. The current channelId, chosen date, entered note, start time and end time will be passed through *standup/schedule/create/v1*.
4. A new scheduled standup will be created with a unique scheduleId.

Displaying channel/personal standups on the calendar:

1. The routes *channel/getschedule/v1* or *user/getschedule/v1* will return objects with all data needed to display a standup(note, start time, end time, date, channelId).
2. The front-end will take this data and display each standup as a div on the calendar's specific date containing the note, start time, end time and channelId(for personal calendar).

Editing schedule details:

1. Given a user has the correct permissions, they can click on a pen icon on the top right corner on the scheduled standup's div.
2. A popup will display the standup's date, start time, end time and note.
3. The user can type in any of these fields.
4. The updated value will be passed through to the corresponding *standup/schedule/edit** route to update the data.

CONCEPTUAL MODELLING - STATE DIAGRAM