



Microsoft Teams Clone

12-06-21 to 15-07-21 (Microsoft Engage Program' 21)

[Github link of project](#) | [Live](#)

Submitted By **Akshat Garg**

[Github](#) | [LinkedIn](#)

Table of Contents

[Overview](#)

[Specifications](#)

[Milestones Completed](#)

[User Guide with Screenshots](#)

[What I learned from this project \[major\]](#)

[Agile Methodology and Weekly Plan](#)

[Problems faced](#)

Overview

A Video calling application, which has features like Group Video call, Team Creation.

The Video call interface has features like Media controls and real time meeting chatbox,

Sending notification to users to join calls.

Specifications

The Group Video chat is achieved by using peer to peer connection through WebRTC and mesh network architecture in which each node gets connected to every other node.

The signalling is handled by using sockets.

Tech Stack

Frontend : [ReactJS](#) with [Typescript](#)

Backend : [Nodejs](#) + express + Typescript

Database used : [MongoDB](#)

Packages : [Socket.io](#) , [simple-peer \(WebRTC wrapper\)](#) , [Fluent UI \(component library\)](#)



Tech Stack

Frontend : Deployed to Vercel

Backend : Deployed to Heroku

Milestones Completed

I. Group Video Call Functionality

Handled group call functionality using webrtc peer to peer connection, the architecture is a mesh network, in which each node gets connected to every other node, simple-peer is used as a helper library for webrtc implementation and signalling is done using sockets.

Main features of Group call are :

- i. Connecting two or more people through webrtc in a video call when they join a specific room with a specific url.
- ii. Users also get the features to invite other users by notification
- iii. Media controls (turning on/off video and audio)
- iv. Chat box in the video call can also be accessed through the record section of teams.

II. Complete user authentication

User Authentication is handled by using json web tokens and cookies.

Main features of auth :

- i. secured the routes at frontend
- ii. Added user model to the application

- iii. configured all other features like teams , meetings and chat with this model to provide users with custom roles

III. Team Model

Team model was added to create teams just like in microsoft teams, users can create or join a team.

Main features of team :

- i. create / delete team [only the creator can delete the team]
- ii. add / remove members from a team
- iii. All records/meetings are shown in the team page
- iv. Team chat feature is a chatting group inside all teams to chat and discuss with teammates.

IV. Notification

Notifications notify users of a specific event like Meeting or chat message and are mainly handled by socket events.

Main features of notification :

- i. user gets a notification to join meeting if invited with a join button
- ii. Receive notifications of a chat after joining the chat room of a team

V. Responsive UI/UX similar to Teams

The user interface uses components from FluentUI by Microsoft (open source), and is completely responsive to all screen sizes.

VI. Proper API Design

API is designed with a proper scalable approach and proper design with complete error handling and http status codes and error messages.

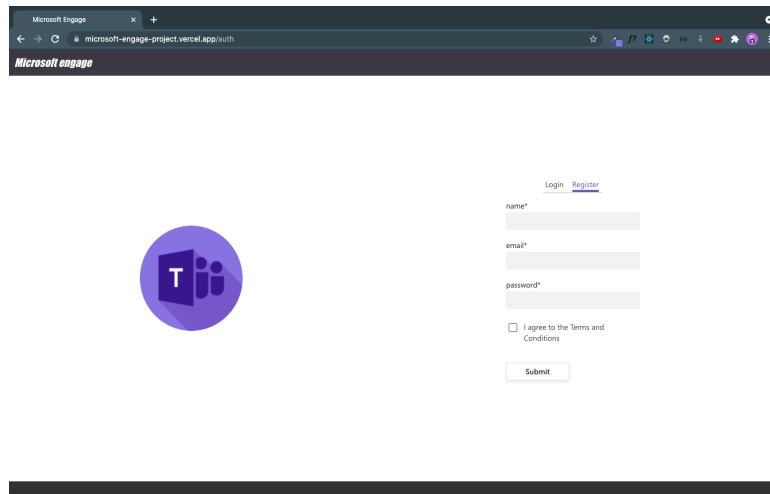
User Guide with Screenshots

1. Login/Register:

To access the user dashboard first login or register on the website, for testing purposes to login use the credentials:

Email : akshatarungarg78@gmail.com
Password : password

Email : alka@email.com
Password : password

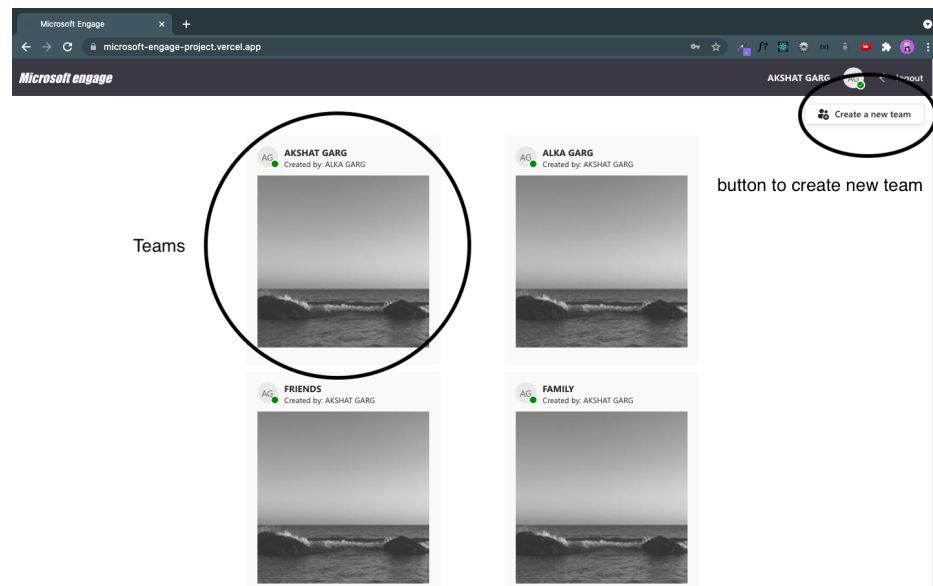


Login / register screen

2. Creating a new team:

Click on Create a new team button on the user dashboard, Enter the name and email Ids of members to be added and click on submit.

Note : News members can be added after the creation of the team also.



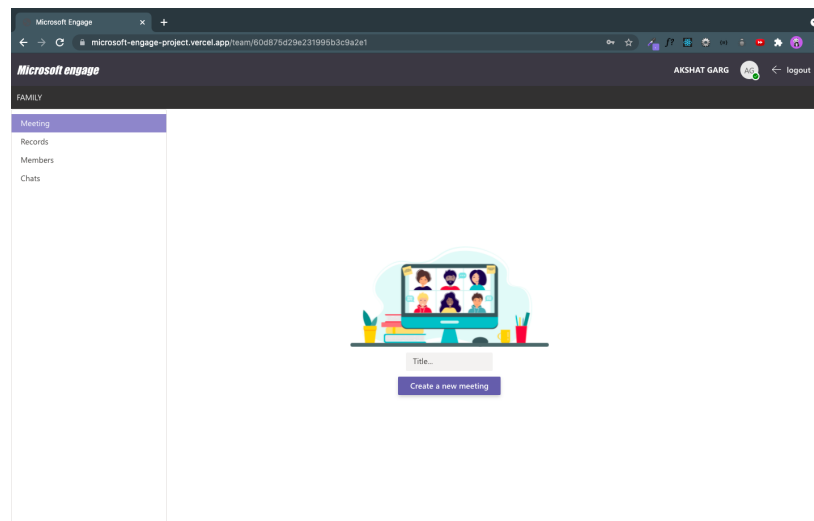
User Dashboard

The screenshot shows the Microsoft Engage Create Team Form. The form has a title 'Name of Team*' and a text input field containing 'Coders'. Below the title, there's a section 'Add Members' with a text input field containing 'vishesh@email.com'. Below the input field, there's a button 'Add a member'. Below the button, there's a list of email addresses: 'alka@email.com' and 'arun@email.com', each with a close button 'X'. At the bottom of the form, there's a button 'Submit'.

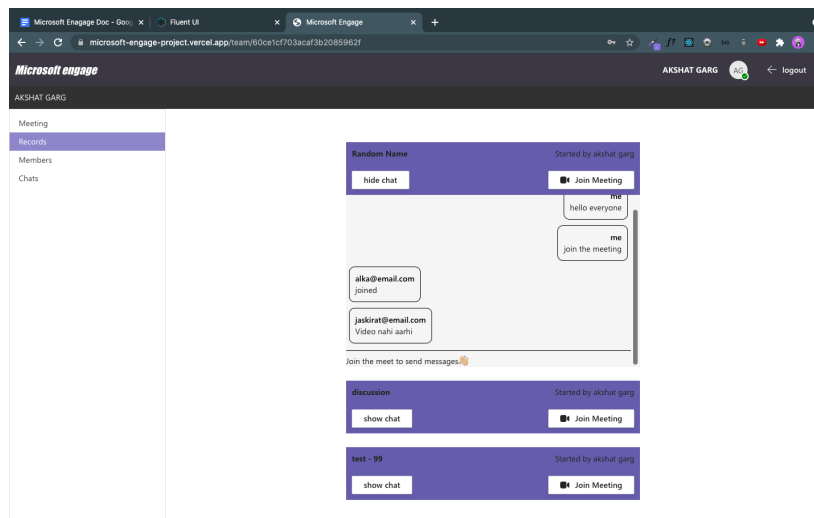
Create Team Form

3. Creating a new Meeting / Group video call:

Choose a team from the user dashboard and click on create a new meeting, add a relevant title to the new meeting. After the team is created the user will be redirected to the meeting and the video call will start immediately. This meeting will also be visible in the records section of the team-page with a join button for other users to join.

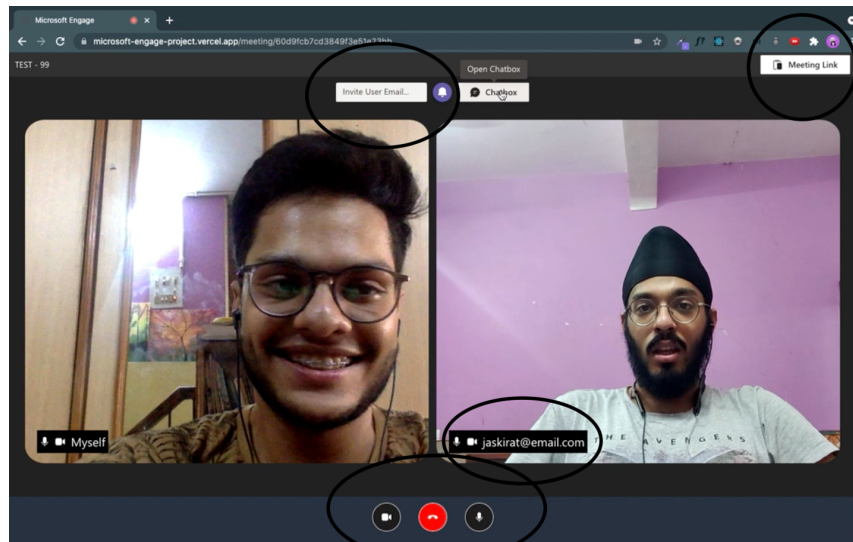


Create meeting section



4. Meeting Controls:

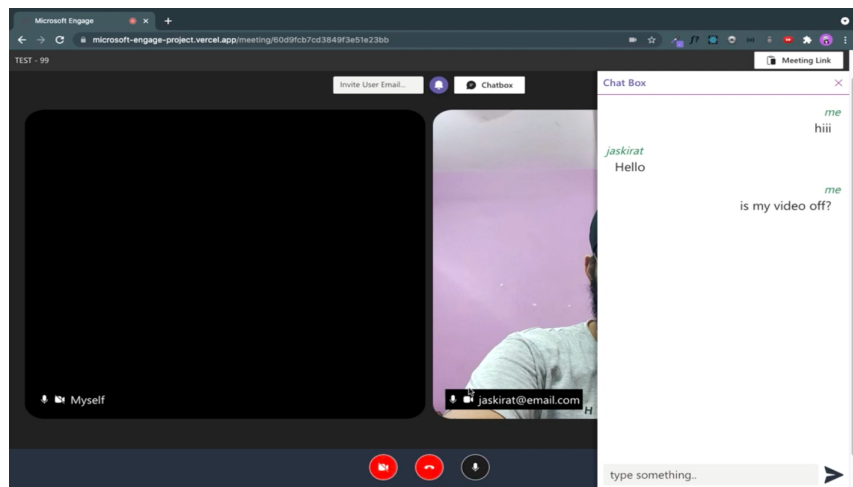
Users can control their media streams by muting or turning off their video from the meeting controls at the bottom of the meeting. They can also leave the meeting by clicking the end call button.



Meeting Screen and controls

5. Meeting Chatbox:

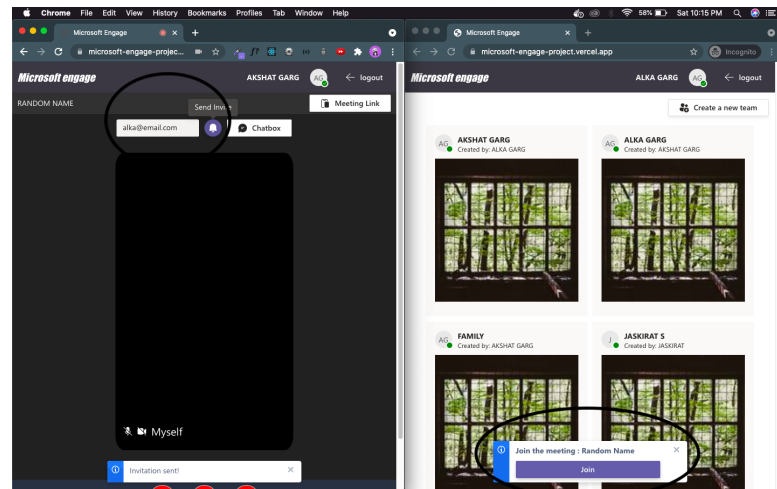
Open the chat box by clicking on the chat-icon at the top, all the chat messages sent by users in meeting will be visible here. To send the message use the input field at the bottom of the chat box and click on the send button. These chats are persistent and are visible in the records section under the meetings.



6. Invite a user to meeting:

To invite a user to the meeting, type the email of the registered user in the input at the top of the meeting page and click on the bell button to send the notification.

The user who is invited gets a notification with a join button to join the meeting.

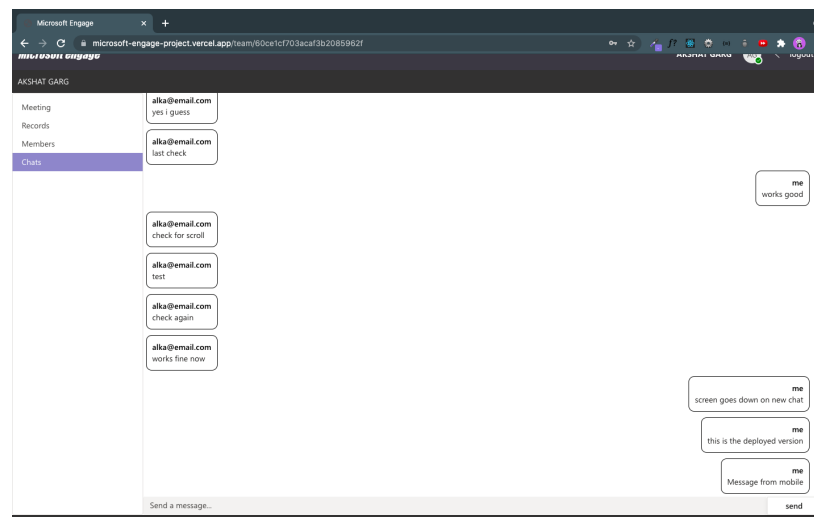


7. Team Chat:

To send or access messages from team chat open the chat section in the team page, and use the input field at bottom to send a new message.

Notifications are also enabled for these chat messages.

These messages are stored in a database for persistence unlike meeting chat.



What I learned from this project [major]

1. WebRTC:

Studied the basics and working of webRTC and implemented mesh networks to create group video chat rooms. Also got to know about mediaDevices and userMediaStreams and how they are manipulated to control streams.

2. Sockets:

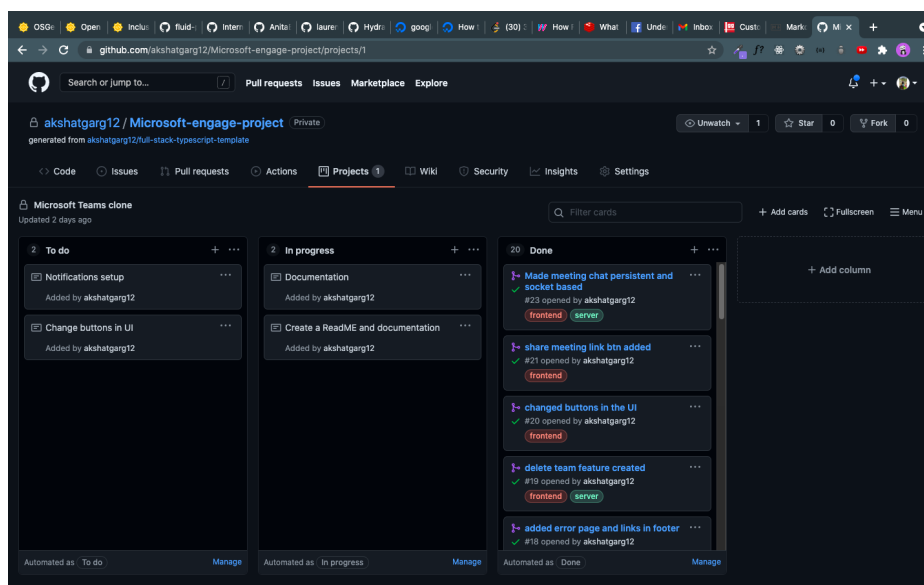
Used sockets to signal the SDP from one peer to another, also sockets were used in creating chat functionality in the teams.

3. Full-stack development:

Integration of all these technologies using an API and a UI helped me learn the fundamentals of web development.

Agile Methodology and Weekly Plan

Used github project and milestones to keep a track of the whole plan



1. Week 1: Learning and Research on WebRTC

Researched about WebRTC and libraries to use for the project, watched youtube videos on the topic and implementation examples.

2. Week 2: Implementation of the basic functionality

Implemented basic feature of connecting two users in a unique url.

Further made changes to connect multiple users using mesh network, Also worked on UI

3. Week 3: Added Meeting features and deployment

Added Media stream control features like muting , video toggle and user invite. Created user authentication and teams model around the application.

Deployed the first version of application with all basic features + additional

4. Week 4: Chat Functionality and Improvements + Documentation

Worked on creating meeting and team chat functionality and refined notifications.

Refined the meeting UI. Also worked on correcting bugs faced during meetings.

Problems faced

1. Handling user refresh during a meeting
2. Handling media stream changes like video off/on or audio on/off
3. Handling peer connections other than local network [configured turn and stun servers for that]
4. Securing the application with user models and authentication
5. Making the whole UI responsive and mobile-first

Thanks to my mentors Amit Tulshyan and Sunil Singhal for guiding me throughout the process and conducting discussion meetings which were very useful for clearing doubts.