Gladiators

A web app built understanding tough situations faced by teachers, students and professionals during video meetings and solve them.

A project under Microsoft Engage 2021

Acknowledgement

I would like to sincerely thank my mentors sir Naresh Bonagiri and ma'am Shraddha Gupta without whom this project would never have been completed and without whose guidance I would have never been able to tackle the difficulties I faced.

I would also like to thank Microsoft for giving me this opportunity to be able to work on this wonderful project and meet an amazing set of peers to interact with.

I would also extend my gratitude to my peers who helped me with any problems I faced during the project.

Last but not the least, I would like to thank my family for having supported me at all times.

Table of Contents

Understanding the problems and their solutions:

- Problems faced by students: Students spend a lot of time preparing for their placement season. One of the most important parts of preparation is having mock interviews and facing situations that they could probably be during interviews through the mock practice sessions. Gladiators eases the preparation by incorporating a live Code IDE along with normal video calling app features which can be simultaneously used to practice mock interviews alongside a video conference.
- Problems faced by teachers (specially nursery students): The teachers face difficulties in teaching alphabets and shapes during online lectures to students who are seeing them for the first time. Gladiators eases this problem by incorporating a Drawing Palette alongside normal videoconferencing features.
- Multi featured for professionals: Gladiators incorporates many features so as to ensure a hassle-free video meeting with messaging, screen sharing, creating rooms feature along with getting the total number of people attending the meeting along with names of the active participants.

Features:

- Video Call Feature: Allows multiple people to connect to rooms and talk to each other.
- Room creation Feature: Creates room for different set of people as per their requirements.
- Screen Sharing Feature: Gladiators also allows you to share screen during the video call.
- Chat Feature: There is also a chat feature which allows you to send text messages to each other.
- Number of participants Feature: Shows total number of participants in the room.
- Active Participants Feature: shows the participants name who are active in the room.
- Live Coding Feature: A coding IDE where all the people connected to a room can code simultaneously.
- Code IDE with syntax highlighting Feature: Allows you to choose from a set of multiple languages and compilers, and highlights syntax on the basis of them. Also takes in customized input, runs the code and gives output.
- Drawing palette Feature: The interface also consists of a whiteboard which all the members in the room can use to draw stuff.

- The whiteboard also contains multiple features like

- colorful pens,

- pens of different sizes,
- clear canvas,
- redo and
- undo drawing options.

Hence, the app incorporates a total of **14** working features implemented in it.

Implementation Details:

Main Backend: Python Flask;

FrontEnd: HTML + CSS + JS

- Video call feature is implemented using Twilio API and WebRTC Technology, which is one of the easiest and the best ways to implement webRTC in python (flask)
- Live coding is implemented using cloud firestore of Google Firebase which makes the changes in real time.
- Coding IDE is based on Ace.js javascript library, and executing is done using Judge0 API hosted on sntc servers.
- Drawing Palette is implemented using WebSockets technology.
- Other features such as Join Room, Leave room etc are also implemented using Flask and WebSockets
- Chat feature is also implemented using WebSockets.

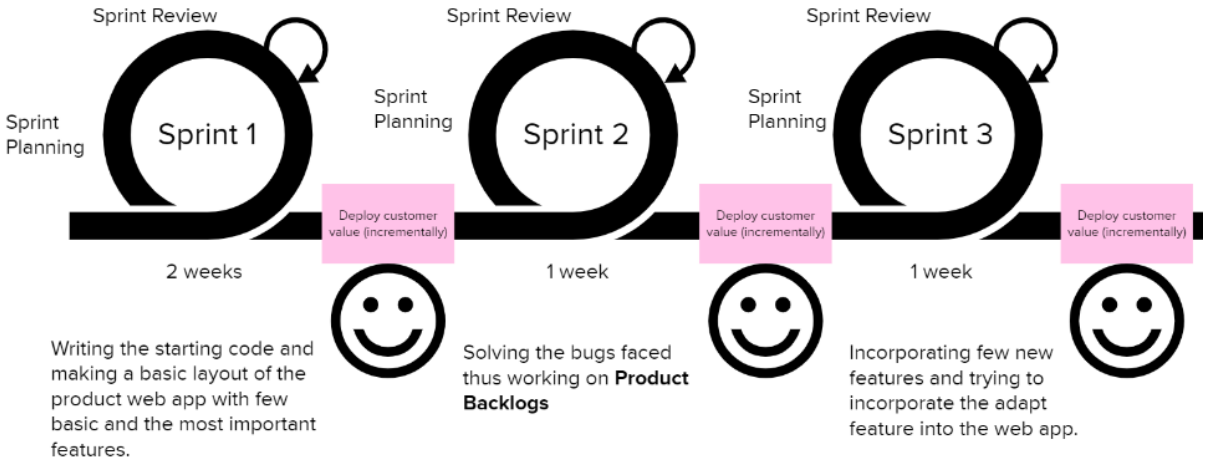
Agile Methodologies used:

The first thing done was planning for the course of 4 weeks. The plan implemented took into account the **SCRUM** process wherein the whole process was broken down into 3 important parts.

- Software devcycle: This part included dividing the time period into **Sprints** of 1-2 weeks each. Each sprint was planned beforehand at the start (**Sprint planning**). I devoted the first two weeks (first sprint) in writing the starting code and making a basic layout of the product web app with few basic and the most important features.
- Work Breakdown: This part includes solving bugs and incorporating deliverables. I devoted the third week (second sprint) in solving the bugs I faced (one was in the code IDE and other one in the screen sharing) thus working on **Product Backlogs**. I gave the fourth week (last sprint) to incorporating few new features and trying to incorporate the adapt feature into my web app.
- Roles and responsibilities: SCRUM recognizes in teams, 2 groups of members. During my project I was the person who was managing both the chicken and pig team by doing the managerial aspects and code part of the project.

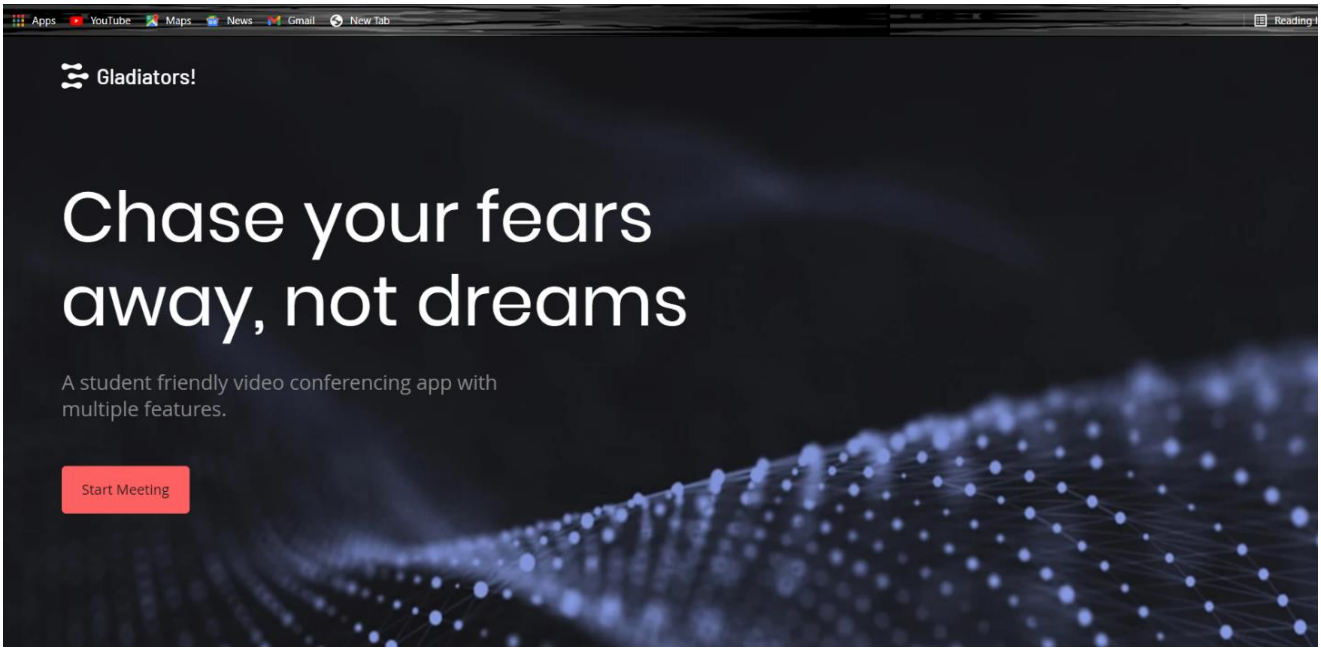
Also, another process of agile methodology allowed me to write code that didn't require much changes so as to incorporate new features.

The following mural image shows the whole summary of 4 weeks.

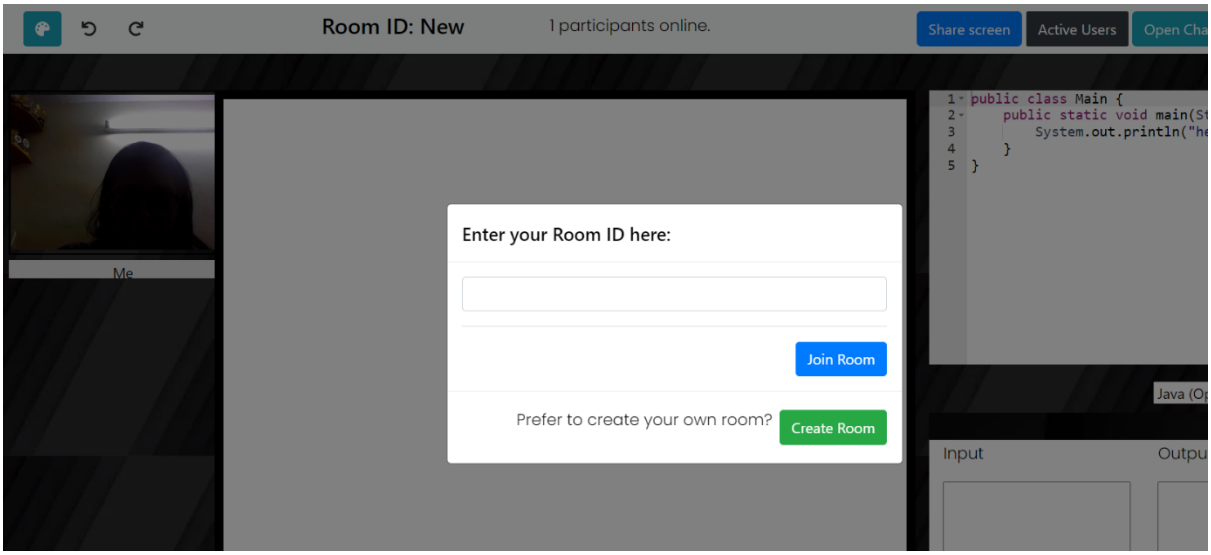


Detailed demo of all the features:

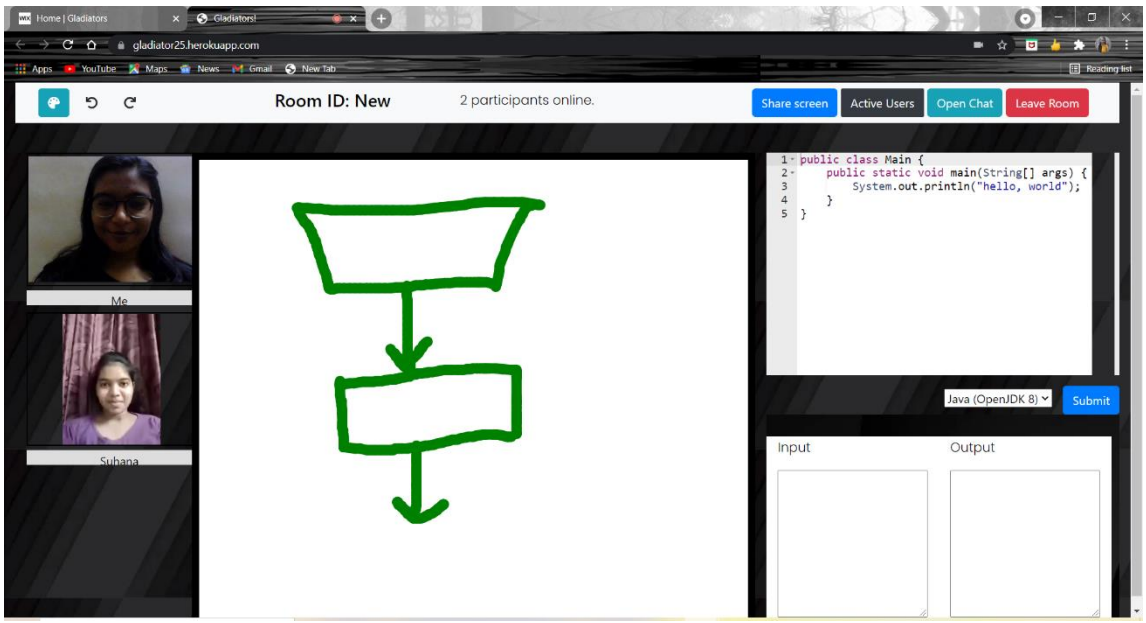
The main website of Gladiators looks like below. You can click the button ‘Start Meeting’ so as to start the meeting.



On clicking start meeting, you get 2 pop-ups. The first ones asks you to enter your full name and the second one will ask you to enter a customized room id (possibly given by the admin) to access a specific room and click ‘Join Room’ or you can click on ‘Create Room’ which will create a random id for your room.

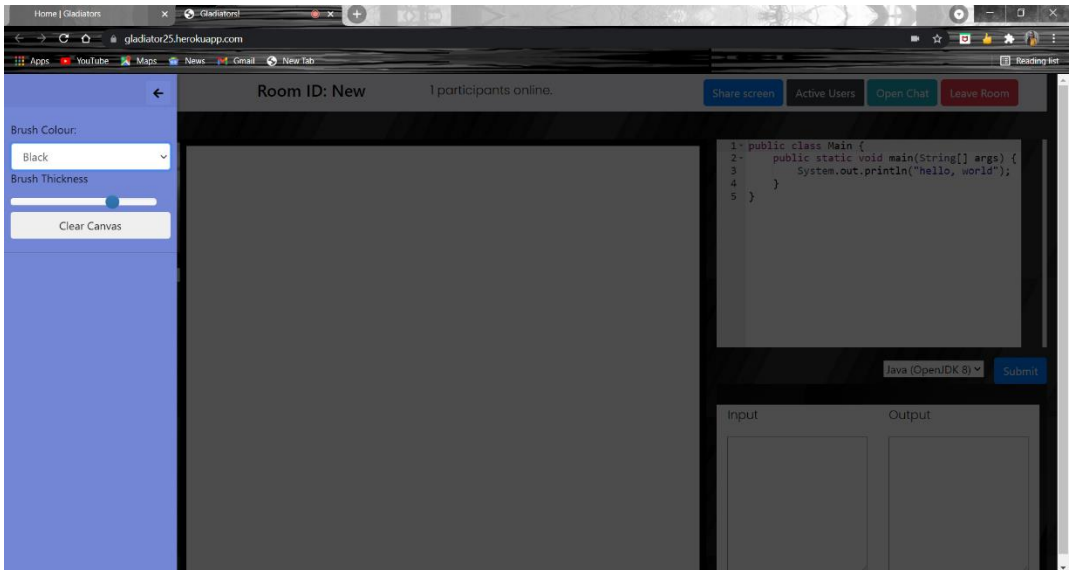


Once, the room gets created and you see that the middle top bar (in the below image you can see that area with ‘2 participants online’ written in it) shows the total number of participants online, the features are ready to function and you are good to go. If this area shows ‘disconnected’ written then refresh it until it shows the number of participants online. The features won’t work until the word disconnected disappears from there.



As can be seen in the above picture, the leftmost div shows the video of all the connected participants in the room. The middle div has the drawing palette and the rightmost div has the code IDE. The size of the video and drawing palette div gets adjusted as more people accumulate in.

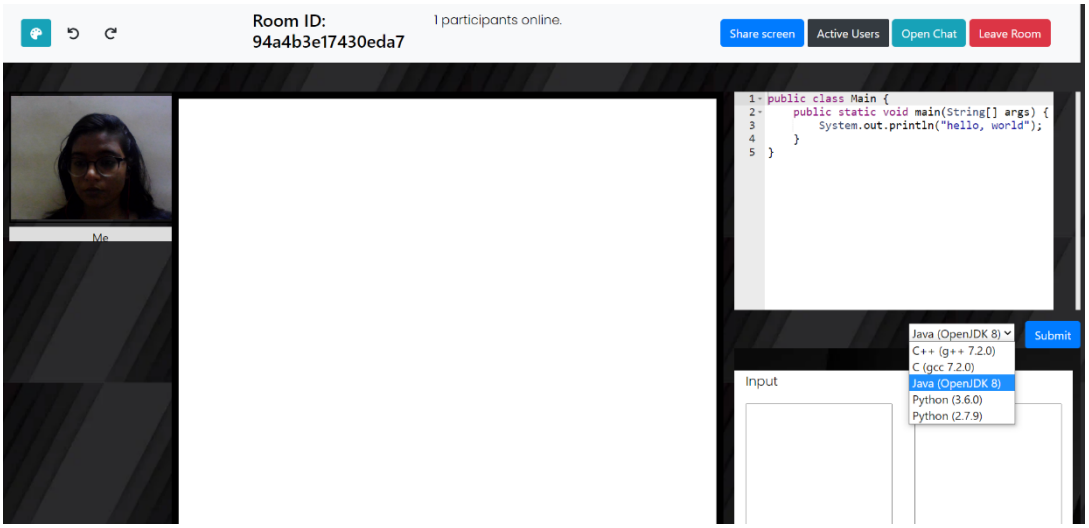
- Using the drawing palette:
  The drawing palette can be used by brushes of different colors. The brush color and size can be adjusted by clicking on the palette icon at the leftmost top bar which opens another side bar (as can be seen in the below image) to adjust the features. The palette icon is accompanied by undo and redo options next to it for the drawing palette.



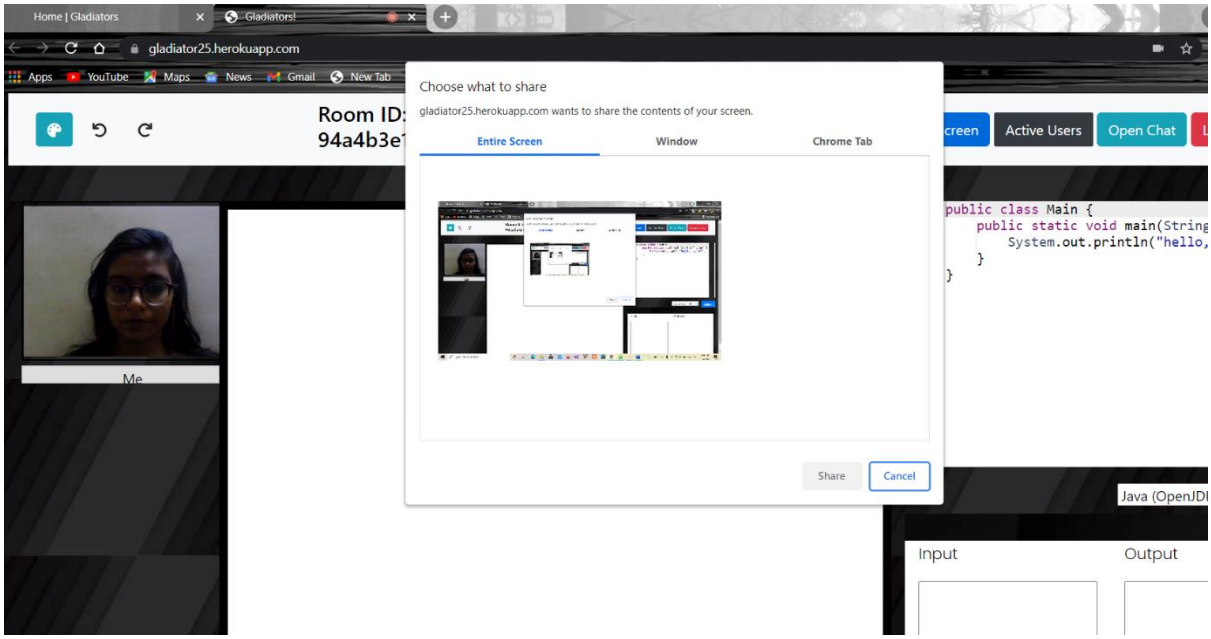
  The canvas can be completely cleared by the ‘Clear Canvas’ option as can be seen in the sidebar above.

- Using Code IDE:

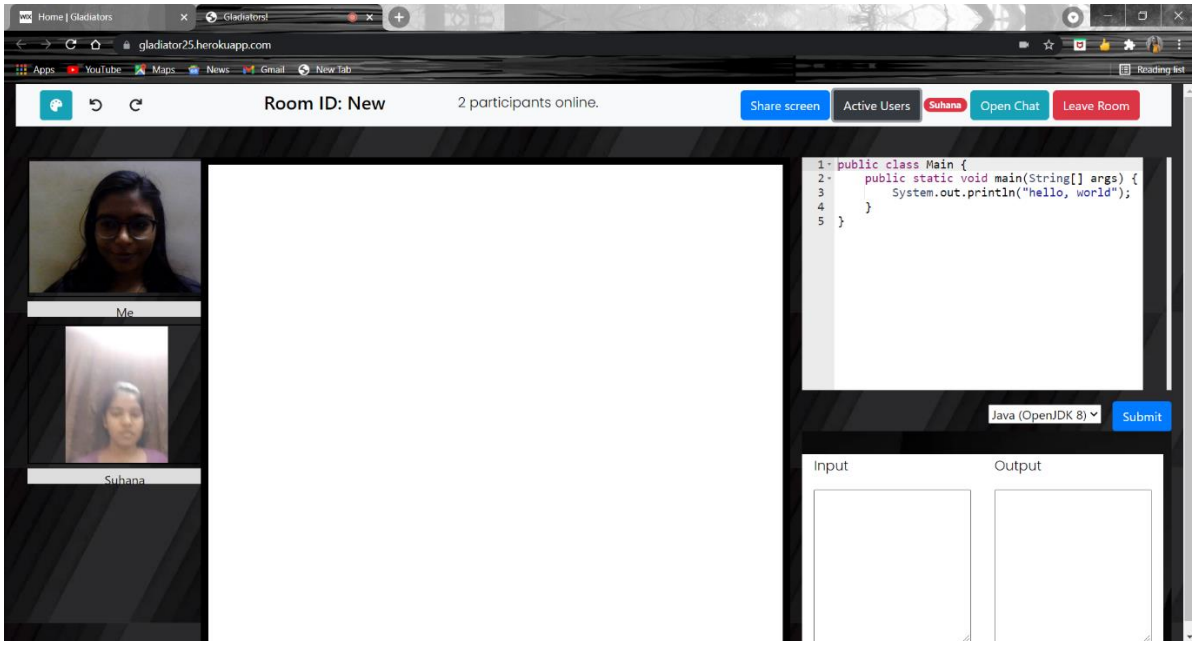
  The code IDE can be used to write codes in 5 different languages and also compile custom inputs along with the written code. The average time taken to compile the code after clicking the ‘Submit’ button is 2.4 seconds but can vary anywhere from 0.9 seconds to 6.3 seconds.



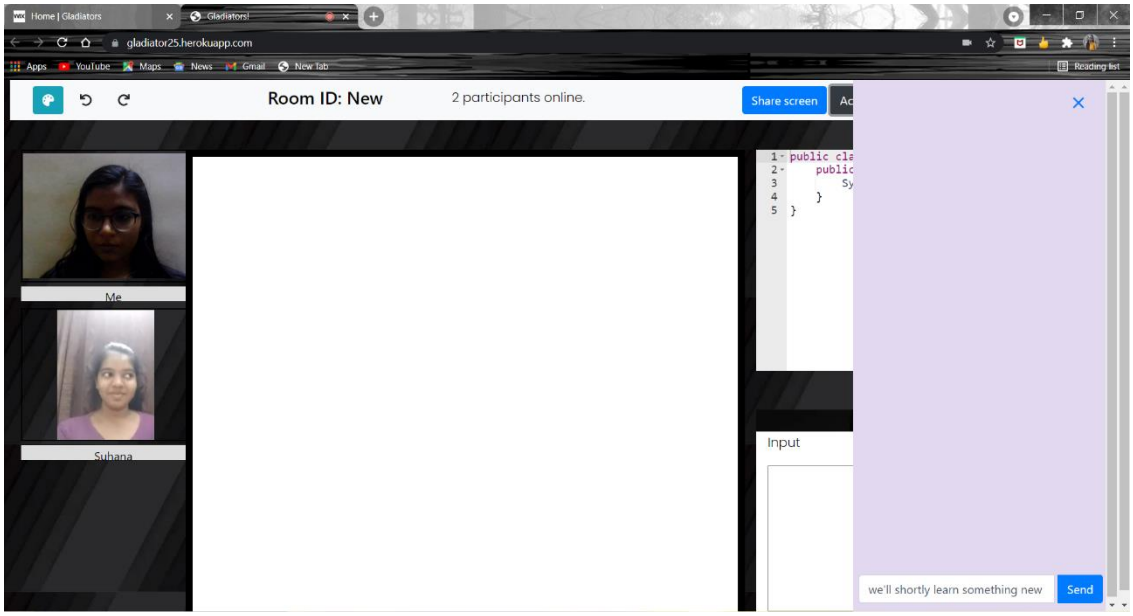
On clicking ‘Share Screen’ option on the right top bar we can share either our entire screen, just a window or a chrome tab using the options provided (refer to the image below).



On clicking ‘Active Users’ option on the right top bar we can see the participant’s name other than us who are present in the room (refer image below). The image below tells that suhana is a participant in the meeting who is online in the room.



On clicking ‘Open Chat’ option on the right top bar we can see a side bar open up where we can chat with other members in the room (refer image below).



On clicking ‘Leave’ option on the right top bar we can leave the room.

Video Implementation:

Refer to this link so as to find the video demo of the app.

How to run the project

- Clone the project
- This Project Requires Python3 to run, so make sure to install it.
- Install the libraries using `pip install -r prerequisites.txt`
- Run the app using `flask run`.
- The project runs on localhost:5000

A project done by

Tushika Singh

Undergrad at IIT Mandi

https://www.linkedin.com/in/tushikasingh/

****