

ONLINE INTERVIEW PLATFORM

A PROJECT REPORT

Submitted by

RAHUL RAJ – 171001070

SATHYA KALYAN - 171001084

SIDDHARTH JAYAN – 171001094

in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY IN

INFORMATION TECHNOLOGY



RAJALAKSHMI ENGINEERING COLLEGE, THANDALAM

ANNA UNIVERSITY, CHENNAI

MARCH 2021

ANNA UNIVERSITY, CHENNAI

BONAFIDE CERTIFICATE

Certified that this Thesis titled “**ONLINE INTERVIEW PLATFORM**” is the Bonafide work of **RAHUL RAJ (171001070), SATHYA KALYAN (171001084) , SIDDHARTH JAYAN (171001094)** who carried out the work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

<<Signature of the HOD with date>>

<<Signature of the Supervisor with date>>

Dr. L. Priya

Mrs. Madhu Rani M

Professor and Head

Assistant Professor

Department of Information Technology

Department of Information Technology

Rajalakshmi Engineering College

Rajalakshmi Engineering College

Thandalam, Chennai

Thandalam, Chennai

This project report is submitted for viva voce examination to be held on at Rajalakshmi Engineering College, Thandalam.

EXTERNAL EXAMINER

INTERNAL EXAMINER

ACKNOWLEDGEMENT

We thank the Almighty God for the successful completion for the project. Sincere thanks to our Chairman **Mr. S. Meganathan** for his sincere endeavor in educating us in his premier institution.

We heartly thank our chairperson **Dr . (Mrs.) Thangam Meganathan**for her motivation and inspiration that paved for the completion of our project.

We also express our gratitude to our principal **Dr. S. N. Murugesan** who helped us in providing the required facilities in completing the project.

We would like to thank our Head of Department **Dr. (Mrs.) L. Priya**for her guidance and encouragement for completion of the project.

We would like to thank **Mrs. M. Madhu Rani**, our supervisor for constantly guiding us and motivating us throughout the course of the project. We express our gratitude to our parents and friends for extending their full support to us.

We thank our Project Coordinator **Dr. (Mrs.) L. Priya and (Mrs.) M. Sindhuja** for her invaluable guidance, ideas, advice and encouragement for the successful completion of this project.

ABSTRACT

Video calls have become an integral part of today's communication for attending an interview, class, community meetings and so on. Along with the increase in live video usage, the quality of interviews has dropped since the interviewees have found ways to malpractice and join the organization though they lack the skills required. Due to these reasons, candidates having the required skills are losing out to those who find alternative ways to cheat and clear the interviews. We propose a way to make the interview process easier and to make it a fair experience for all the candidates. We introduce the Online Interview Platform which will provide features such as a collaborative code editor, video/audio call facility, chat box and a collaborative whiteboard to make sure all the activities of an interview takes place in the same tab and both the interviewer and interviewee can collaborate as if they were attending an interview in person. With these facilities available, the interviewee will have to stay on the same tab on their browser and would be able to perform the tasks required for the interview rather than switching tabs and trying to cheat. With the help of this platform, we'll be able to screen out candidates, assess them properly and make decisions based on his/her performance

KEYWORDS : Video Call, Online Interview, Collaborative editor, Collaborative whiteboard, Preventing malpractice

TABLE OF CONTENTS

TITLE	PAGE NO
ABSTRACT	iv
LIST OF FIGURES	viii
LIST OF SYMBOLS	ix
1. INTRODUCTION	1
1.1 SYSTEM OVERVIEW	1
1.2 SCOPE OF THE PROJECT	3
2. LITERATURE SURVEY	4
2.1 VIDEO CONFERENCING	4
2.2 CODE EDITOR	5
2.3 COLLABORATIVE EDITOR	6
2.4 INFERENCES	7
3. SYSTEM ANALYSIS	8
3.1 EXISTING SYSTEM	8
3.2 PROPOSED SYSTEM	9
3.2.1 ARCHITECTURE DIAGRAM	10
3.3 REQUIREMENTS SPECIFICATION	11

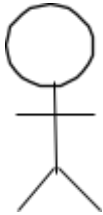
3.3.1 HARDWARE SPECIFICATION	11
3.3.2 SOFTWARE SPECIFICATION	11
3.4 SOFTWARE SPECIFICATION	12
3.4.1 JAVASCRIPT	12
3.4.2 VS CODE	12
3.4.3 NODE JS	13
3.4.4 EXPRESS JS	13
3.4.5 MONGODB	14
4. SYSTEM DESIGN	15
4.1 SYSTEM DESIGN	15
4.1.1 UML DIAGRAM	15
4.1.2 DFD DIAGRAM	16
4.1.3 SEQUENCE DIAGRAM	18
5. SYSTEM IMPLEMENTATION	19
5.1 REGISTER AND LOGIN	19
5.2 MEET ID GENERATION	19
5.3 COLLABORATIVE EDITOR, CHAT BOX, REAL TIME VIDEO/AUDIO CALLING	19
5.4 COLLABORATIVE WHITEBOARD	20

6. IMPLEMENTATION AND RESULTS	21
6.1 REGISTRATION/LOGIN	21
6.2 CREATE NEW MEET	22
6.3 REAL TIME MEET	23
7. CONCLUSION AND FUTURE ENHANCEMENT	25
7.1 CONCLUSION	25
7.2 FUTURE ENHANCEMENT	25
APPENDIX 1	26
APPENDIX 2	32
REFERENCES	35

LIST OF FIGURES

FIGURE NUMBER	FIGURE NAME	PAGE NUMBER
3.2.1.1	ARCHITECTURE DIAGRAM	12
4.1.1	UML DIAGRAM	17
4.1.2.1	DATA FLOW DIAGRAM	18
4.1.3.1	SEQUENCE DIAGRAM	20

SYMBOLS AND NOTATIONS



Actor

StandardUML icon for actor is “stick man” icon the name of the actor above or below the icon



Use case

Every use case must have a name. Use case is shown as ellipse containing the name of the use case.



An **association** between an actor and use case indicate that actor and the use case communicate with each other. An actor could be associated to one or several use cases.

CHAPTER 1

INTRODUCTION

1.1 SYSTEM OVERVIEW

Ever since the world went into hibernation owing to the sudden outbreak of the Covid-19 pandemic, a lot has changed in the way industries work. With worldwide travel restrictions and strict quarantine regulations, the medium of conducting business has had to be reimagined for the current crises. One of the many radical changes that we have witnessed came in the process of hiring new candidates, through the previously uncommon method of virtual interviews. Yes, video calls and online conferencing had always been a part of hiring in certain organizations. However, it became the mandate to be followed by almost every other company on the planet over the course of the last year.

These virtual job interviews are much akin to in-person interviews. The advantage is that he/she can take part in a video job interview from the comfort of your home without any travel costs or worrying about whether the bus route you are taking will get there on time. The interviewee may need to download and login to the video platform the interviewer has provided to you. The interviewee may have one interviewer or a panel interview with two or more interviewers on your screen.

During the interview, the interviewer will ask the interviewee general questions pertaining to your field of work. And, much like the usual interviews that were held before disaster struck, the ones who performed well in the interview are selected on the basis of the answers to the questions asked to them and also based on their attitude during the interview. Well, technology seems to

have solved the crisis of conducting interviews amidst a global pandemic, hasn't it? A big no, is the answer to that question.

Over time, candidates began to realise the shortcomings in this virtual environment and figured out unique and innovative ways to malpractice their way to a job. With more such candidates clearing interviews by their deceitful ways, talented and budding individuals who gave it their all ended up missing out on an opportunity to work at their desired firm. Not just that, those candidates who cleared their interview by trickery do not make the cut in the industry and become deadwood at the firm they end up joining.

No, we cannot preach individuals the virtues of staying true to their conscience. But, we can indeed figure out ways to contain their malpractices to ensure that truly talented people are hired by the organization. And for that reason, we have once again decided to leverage the power of technology to make interviews foolproof.

Our application aims to make interviews more straightforward, by reducing the chances of a candidate malpracticing by providing a code editor along with a built in compiler in the same tab as the one where the video call has been connected. The candidate will be expected to type their solution to the problems given by the interviewer on the editor given in our page. And any changes made by the candidate on the editor will be immediately reflected in the editor displayed on the interviewer's screen.

Thus, the interviewer can keep track of every change made by the candidate, preventing the candidate from copying and pasting the code from another source. Also, since the candidate's video is being monitored live by the interviewer, the chances of any form of malpractice becomes little to none in our application.

There is a chat box feature available in the application as well, to allow the candidate and interviewer to communicate with each other in case of any audio issues at either end of the application. Also, the interviewer can share his questions with the candidate through this chat box while conducting the interview.

Okay, so what if, the interviewer expects the candidate to not just write answers but to draw them? Models, charts, UML diagrams, flow charts, graphs, and many other forms of drawings have become a crucial part of an engineer's work. And unlike an in-person interview, an interviewer cannot usually test a candidate's mettle in drawing as such, during a virtual interview. Our application helps quell this issue by providing a draw board as well on the same page as the video interview. Similar to the code editor, any changes made on the drawing board are immediately reflected on the board at the interviewer's end thereby enabling the interviewer to rate a candidate's performance to a better standard.

Thus, our application looks to solve some of the major issues faced by interviewers and organizations while conducting interviews virtually.

1.2 SCOPE OF THE PROJECT

To develop an online interview platform that would ensure proper conduction of an interview virtually and reduce the risk of cheating and other forms of malpractice during the course of the interview.

CHAPTER 2

LITERATURE SURVEY

Online Interview Platform is a Internet-based tool offering a virtual environment for remote meeting and collaborative work among geographically dispersed participants. Remote conferencing can be used to minimize travel expenses and time for face-to-face meetings. This explains the worldwide shift towards web based interviewing tools.

Virtual platforms for video conferencing already exist. These virtual platforms are good for video conferencing but they are not as efficient when it comes to the conduction of an interview. These platforms do not test the competency of a candidate to the fullest and also do not have a proper means to filter out malpracticing candidates.

2.1 VIDEO CONFERENCING

Video conferencing is a technology that allows users from diverse locations around the world to conduct face-to-face meetings albeit without having to actually come into contact in-person. This technology is particularly useful for users in different cities or even different countries because it saves time, expenses, and hassles associated with travel. Applications for video conferencing include holding routine meetings, negotiating business deals, and interviewing job candidates.

In 2012, K.V. Rop and N.K Bett from the Department of Telecommunication and Information Engineering of Jomo Kenyatta University of Agriculture and Technology had submitted a research paper “Video Conferencing and its Application in Distance Learning”[5] putting an emphasis

on how video conferencing could be an effective way of delivering subject matter in classrooms.

Antonious G Nanos and Anne E James, part of the Distributed Systems and Modelling(DSM) Research group faculty of Engineering and Computing Coventry University had researched on the existing Virtual meeting platforms and put forth a paper “A Virtual Meeting System for the New Age”[4] back in 2013. They had summarized that the existing virtual platforms like Cisco Webex, Skype, MS Live Meeting were providing features such as audio/video communication, text chat but that these features were not enough to replace direct meetings. Hence, they decided to provide more features to improve the quality of virtual conferences with the support of Artificial Intelligence and developed a product called the V-Room. Xinggong Zhang and Yang Xu of New York University had put forth a paper “Modeling and analysis of skype video calls” [14] in the year 2013, elaborating upon the varying packet loss rate , propagations delay ,network bandwidth in Skype video calls.

2.2 CODE EDITOR

A source-code editor is considered to be a text editor program that has been designed specifically for editing the source code of computer programs. It could be a standalone application or it may be built upon an integrated development environment (IDE) or a web browser. Source-code editors are a fundamental programming tool, as the fundamental job of programmers is to write and edit source code. And when it comes to a job interview in the field of development and programming, every candidate’s coding ability needs to be scrutinized before giving the go-ahead for a role in the organization. In 2015, Aditya Kurniyawan, Christine Soesanto and Joe Erik Carla Wijaya from Bina Nusantara University had developed an Online Code Editor which was a real time code editing application developed for collaborative programming. It

supports C, C++, Java programming languages and published it in the paper “CodeR: Real-time Code Editor Application for Collaborative Programming”. [16] They have used a facebook plugin to enable logging in using their facebook profiles to access the application easily and to integrate the editor with the social network to collaborate with each other. They utilised the Operational Transform Algorithm to develop the collaborative code editor.

In the year 2017, Sonali R. Gujarkar, Samprada D. Nimrad and Shital Meshram from Shri Shankarprasad Agnihotri College Of Engineering in Wardha, Maharashtra had proposed a paper “A Review on Server Based Code Editor”. It was a paper that explained about their server based code editor which had the capability of running HTML, CSS, JavaScript and Java source codes without downloading any additional resources.

2.3. COLLABORATIVE EDITORS

A collaborative real-time editor is said to be a type of collaborative software or web application which enables real-time collaborative editing, simultaneous editing, or live editing of the same digital document, computer file or cloud-stored data – such as an online spreadsheet, word processing document, database or presentation – at the same time by different users on different computers or mobile devices, with automatic and nearly instantaneous merging of their edits.

Quang-vinh dang and Claudia-lavinia ignat of National Institute for Research in Computer Science and Control in Le Chesnay, France, had proposed a paper “Performance of real time collaborative editors at large scale” [7] in the year 2016, that provided some clarification on the issues that arise with increase in number of users.

Ajay Khunteta of Poornima group of colleges had published a paper “A Survey on Operational Transformation Algorithms: Challenges, Issues and Achievements”[11] in the year 2010, elaborating upon the Operational Transformation Algorithm which forms the basis of building collaborative applications such as a collaborative code editor.

Francis Pacull of Bag-Era, Alain Sandoz of Université de Neuchâtel and André Schiper of Ecole Polytechnique Fédérale de Lausanne came up with a paper “Duplex : A distributed collaborative editing environment” [17], in 1994. They had built DUPLEX, which was a distributed collaborative editor for users connected through a large-scale environment such as the Internet.

2.4 INFERENCES

- The virtual interview must not be restricted to being a generic video conference.
- The interview must take place flawlessly on a lower bandwidth as well.
- There must be a code editor which can be used to execute code without having to download any additional resources.
- A draw board must be implemented to test the candidates on their modelling skills.
- The interviewer must be able to collaborate with the interviewee on the editor and the draw board.
- Considering the collaborative nature of our platform, we must ensure that it functions perfectly in a large scale environment as well.

CHAPTER 3

SYSTEM ANALYSIS

3.1.1 EXISTING SYSTEM

Existing applications are available for conducting coding assessments, with proctoring features available as well. Applications such as Hacker Rank, Code Signal are used for conducting these types of coding interviews. These applications available offer online coding assessments to the candidates with proctoring facilities. But the issue of proctoring is that the camera just requires to recognize the face on the screen. The candidate taking up the test must not move out of the screen. Proctoring approach like this allows the candidate to malpractice, since candidates can use their phone or surf the web using their hands while their head is still in the camera's frame. These practices by the candidate are unethical and if they are selected, the deserving candidates lose out their spot because of such unethical practices.

Also during the interviews, the interviewer asks the candidate to share their screen and make them code, design to check their eligibility. But there have been cases where the camera doesn't work which allows the user to surf the web even though they share the screen and give out answers copied from the internet.

Using these interview platforms are really costly, each coding assessment conducted on those platforms is expensive, the rate of conducting an assessment for one candidate is around 5 dollars. Budding startups, small organizations will not be able to afford such an amount to conduct interviews for the candidates.

3.1.2 PROPOSED SYSTEM

We propose an application which solves the disadvantages of the other platforms. The Online Interview Platform is a web application developed using Node JS which is an open source, cross platform, back-end JavaScript runtime environment and Express JS which is a framework for Node JS designed for building web applications and API's. The Online Interview Platform's objective is to make the process of interview easier for both the candidate as well as the interviewer by improving the quality of interviews taking place and reducing the number of malpractice taking place. The web application provides various features such as Real Time Audio/Video calling which allows the interviewer and the candidate to connect to each other using Peer JS, which uses Web RTC's functionality for data, video and audio calls. The applications also provide a Collaborative editor and a collaborative whiteboard which allows the user to immediately see through the changes as they code or design on screen. They use a technology called Operational Transform which supports a wide range of collaborative functions in software systems. A chat box has also been included in the application which can be used by the users to send the invite code, questions for programming, etc.

3.2.1 ARCHITECTURE DIAGRAM

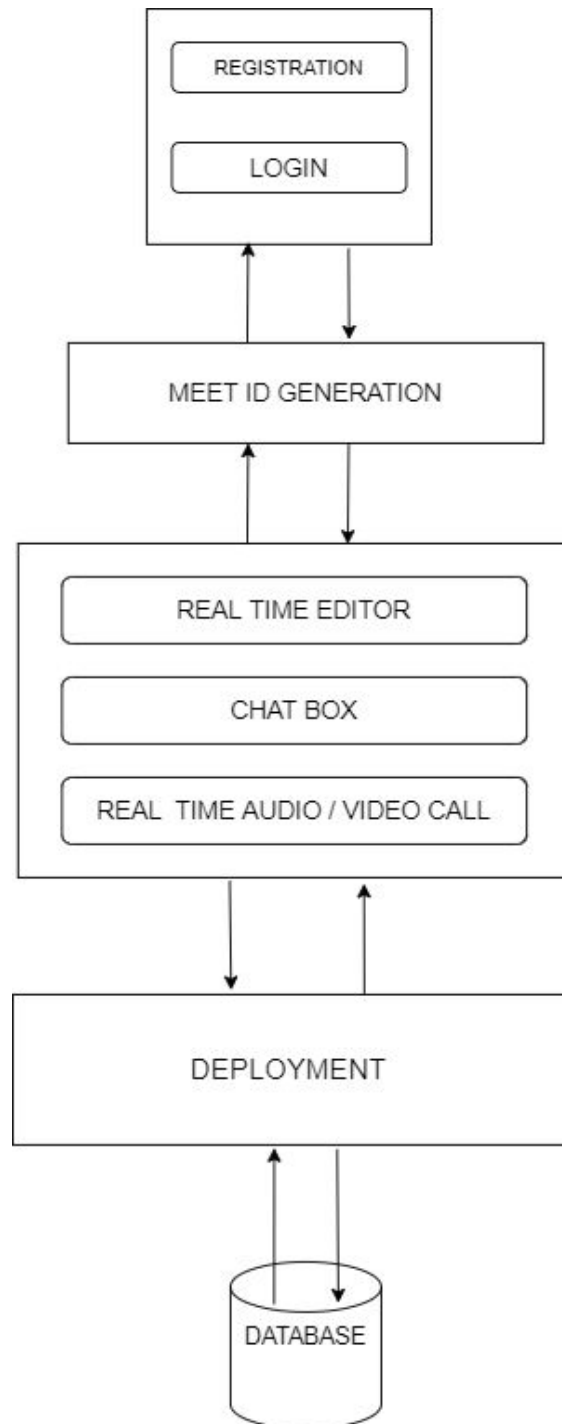


Fig 3.2.1.1 Architecture Diagram

The above diagram explains the architecture of the Online interview Platform. The interviewer has to register/login at first either using traditional email and password option or single sign on functionality using Facebook. Once the Interviewer signs in, they must create a new meet for an interview. This will result in the formation of a new page with a unique link. The link generated must be shared to the candidate appearing for the interview. The page generated consists of a Real time collaborative editor and whiteboard, real time audio/video calling functionality and a build in chat box for the application. The code and user details are stored in the Database implemented using MongoDB.

3.3 REQUIREMENT SPECIFICATION

3.3.1 HARDWARE REQUIREMENTS

1. System : Minimum Pentium IV 2.4GHz
2. Harddisk : 50 GB
3. Monitor : 15 VGAcOLOR
4. RAM : 4Gb or above
5. Camera
6. Microphone

3.3.2 SOFTWARE REQUIREMENTS

1. VS Code
2. Node JS

3. Javascript
4. Express JS
5. MongoDB

3.4 SOFTWARE SPECIFICATION

JAVASCRIPT

JavaScript is an object-based scripting language which is lightweight and cross-platform. JavaScript is not a compiled language, but it is a translated language. The JavaScript Translator (embedded in the browser) is responsible for translating the JavaScript code for the web browser. JavaScript (js) is a light-weight object-oriented programming language which is used by several websites for scripting the webpages. It is an interpreted, full-fledged programming language that enables dynamic interactivity on websites when applied to an HTML document. It has been adopted by all other graphical web browsers. With JavaScript, users can build modern web applications to interact directly without reloading the page every time. The traditional website uses js to provide several forms of interactivity and simplicity. In addition to web browsers, databases such as CouchDB and MongoDB use JavaScript as their scripting and query language.

VS CODE

Visual Studio Code is a freeware source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality. In the Stack Overflow 2019 Developer Survey, Visual Studio Code was ranked the most popular

developer environment tool, with 50.7% of 87,317 respondents reporting that they use it. Visual Studio Code was released under the MIT License, having its source code available on GitHub. On April 14, 2016, Visual Studio Code was released to the Web.

NODE JS

Node.js is a cross-platform runtime environment and library for running JavaScript applications outside the browser. It is used for creating server-side and networking web applications. It is open source and free to use. Many of the basic modules of Node.js are written in JavaScript. Node.js is mostly used to run real-time server applications. The definition given by its official documentation is as follows: Node.js is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices. Node.js also provides a rich library of various JavaScript modules to simplify the development of web applications.

EXPRESS JS

Express is a fast, assertive, essential and moderate web framework of Node.js. You can assume express as a layer built on the top of the Node.js that helps manage a server and routes. It provides a robust set of features to develop web and mobile applications. It can be used to design single-page, multi-page and hybrid web applications. It allows setting up middlewares to respond to HTTP Requests. It defines a routing table which is used to perform different actions based on HTTP method and URL. It allows to dynamically render HTML Pages based on passing arguments to templates.

MONGODB

MongoDB is an open-source document database that provides high performance, high availability, and automatic scaling. In simple words, you can say that - Mongo DB is a document-oriented database. It is an open source product, developed and supported by a company named 10gen. MongoDB is available under General Public license for free, and it is also available under Commercial license from the manufacturer. MongoDB is a scalable, open source, high performance, document-oriented database. MongoDB was designed to work with commodity servers. Now it is used by the company of all sizes, across all industries.

CHAPTER 4

SYSTEM DESIGN

4.1 SYSTEM DESIGN

4.1.1 UML DIAGRAM

This section focuses on describing the individual components of the system and how they interact with each other. The system design is explained by using UML diagrams and data flow diagrams. Unified modelling language is a rich language to model software solutions, application structures, system behavior and business processes. A data flow diagram is a graphical representation of the “flow” of data through an information system, modelling its process aspects.

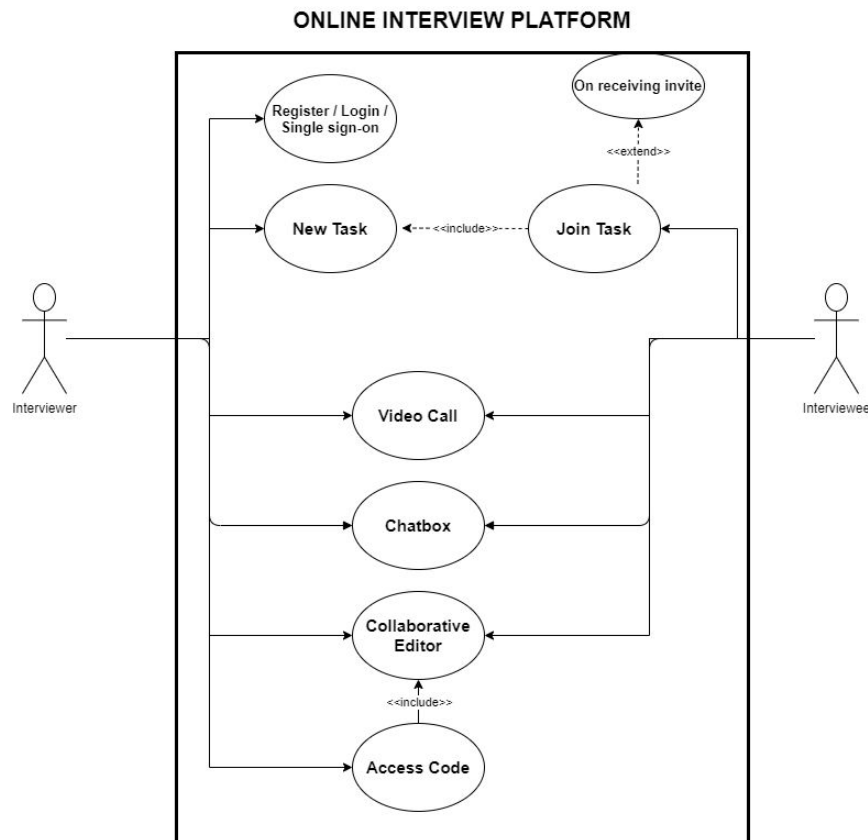


Fig 4.1.1: UML Diagram

The Unified Modeling Language (UML) is general purpose, developmental, modeling language in the field of software engineering. The actors involved with this system are the interviewer and the interviewee. The interviewer logs in to our platform through username and password authentication or Facebook single sign on functionality. After logging in, the interviewer creates a new meet and shares the meet ID with the interviewee who would be able to join the meeting through a web browser.

In the meeting the interviewer and interviewee will be able to communicate with each other through the video call feature or by messaging in the chat box. The interviewer shall test the ability of the candidates by making them solve programming problems in the inbuilt collaborative editor. Also, the interviewer shall test the modeling and planning skills of the candidates by asking them to utilize the whiteboard. Thus, our application shall enable the smooth and efficient conduction of an interview.

4.1.2 DFD DIAGRAM

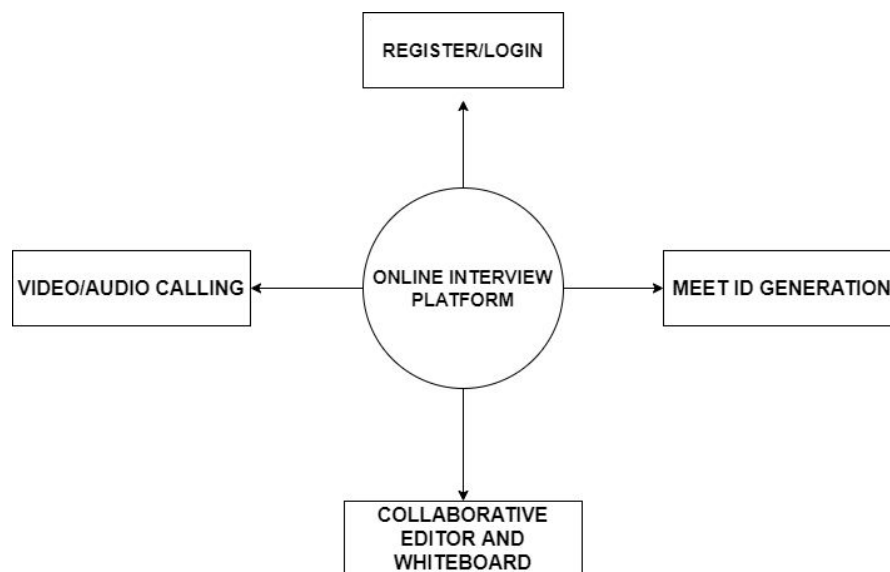


Figure 4.1.2.1 DATA FLOW DIAGRAM LEVEL 0

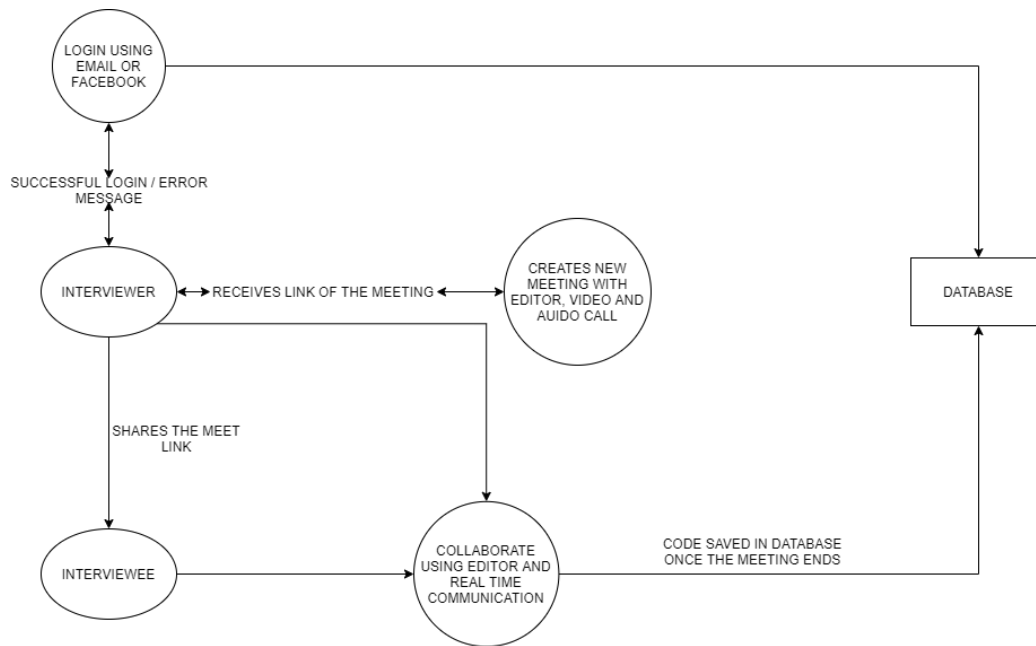


Figure 4.1.2.2 DATA FLOW DIAGRAM LEVEL 2

The data flow diagrams are traditional visual representations of the information flows within a system. In our application, the interviewer registers or logs in to the system by providing a username and password or by utilizing his Facebook credentials. This data is authenticated and on successful authentication, the user is provided with access to the system.

After logging in, the interviewer will be provided with the ability to create a new meeting and on creation of such a meeting, a new meet ID will be generated which shall be shared by the interviewer with the interview candidates. On joining the meeting, the interviewer shall prompt the interviewee to type some code in the collaborative code editor to test their coding skills. All the code that gets typed in the editor will be saved to a database for future reference.

4.1.3 SEQUENCE DIAGRAM

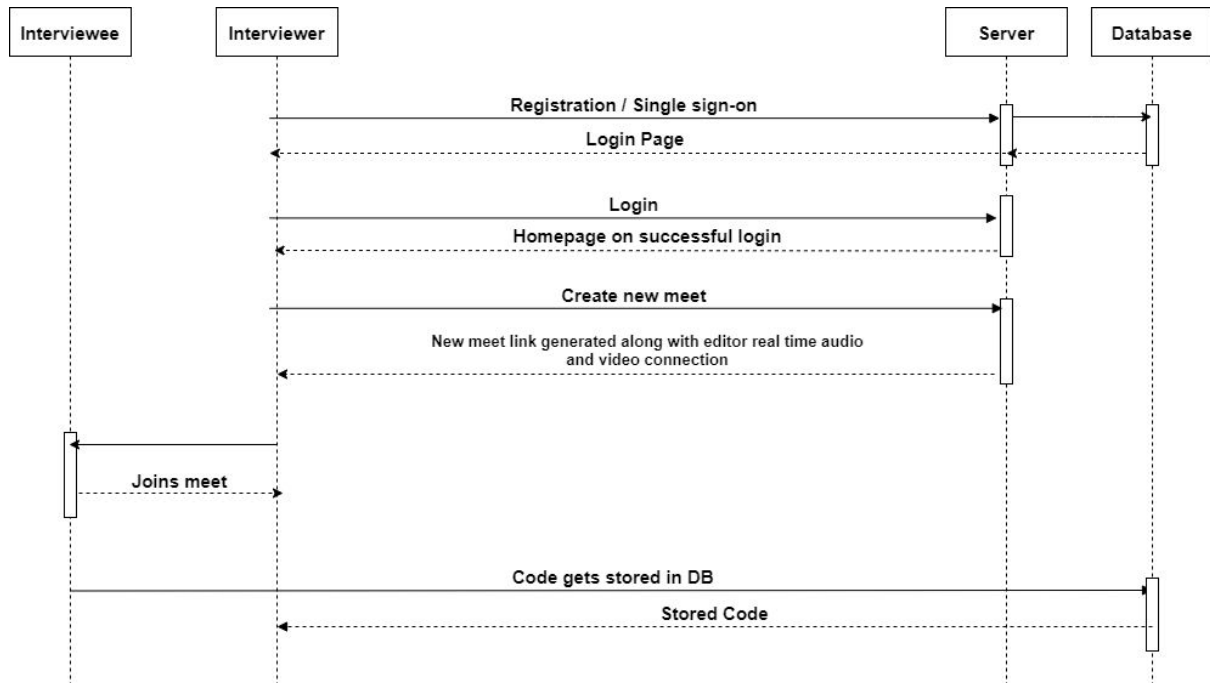


Figure 4.1.3.1 SEQUENCE DIAGRAM

The sequence diagram represents the flow of messages in the system and is also termed as an event diagram. It helps in envisioning several dynamic scenarios. It portrays the communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part at the run time. In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extends across the bottom of the page. It incorporates the iterations as well as branching.

CHAPTER 5

SYSTEM IMPLEMENTATION

The proposed model consists of four modules :

5.1 Register and Login

In this module, we have implemented the registration and login functionality in a couple of ways. One option is for the user to sign up and sign in using an email address and password. The email addresses and passwords of the users are gathered and stored in the MongoDB database, during the time of registration. During login, the information entered by the user is authenticated before providing him access to the website. Another way is to sign in using Facebook's single sign on feature which will allow the user to login using their Facebook profile. Where our application will authenticate using the details from Facebook and then store it in the database as well.

5.2 Meet ID Generation

When the interviewer wants to create a new meeting, all he needs to do is to click the button "CREATE MEET". This will lead to the generation of a new meet page with a unique ID every time the button is clicked. The ID generated will be stored in the database and can be used within the day. Once the room has been created the interviewer is supposed to share the link with the interviewee and the interviewee shall join the meeting along with the interviewer.

5.3 Collaborative Editor, Chat box and Real time Audio/Video call

Every meeting that has been created consists of a built in chat box that has been implemented using socket functionality. The page will also include the video

calling feature implemented using Peer JS built on top of Web RTC peer to peer functionality. This video call will provide the means for direct communication between the host and the candidate.

A collaborative editor will also be made available to both the interviewer and the candidate using which both of them can collaborate. The collaborative editor is created using an operational transform algorithm which allows varied software systems to collaborate. The collaborative editor allows the interviewer to test the candidate's coding skills by giving him problems through the chat box. The interviewer will be able to monitor the candidate's typing speed and coding ability by watching over the editor as he is typing the solution to a particular question.

5.4 Collaborative Whiteboard

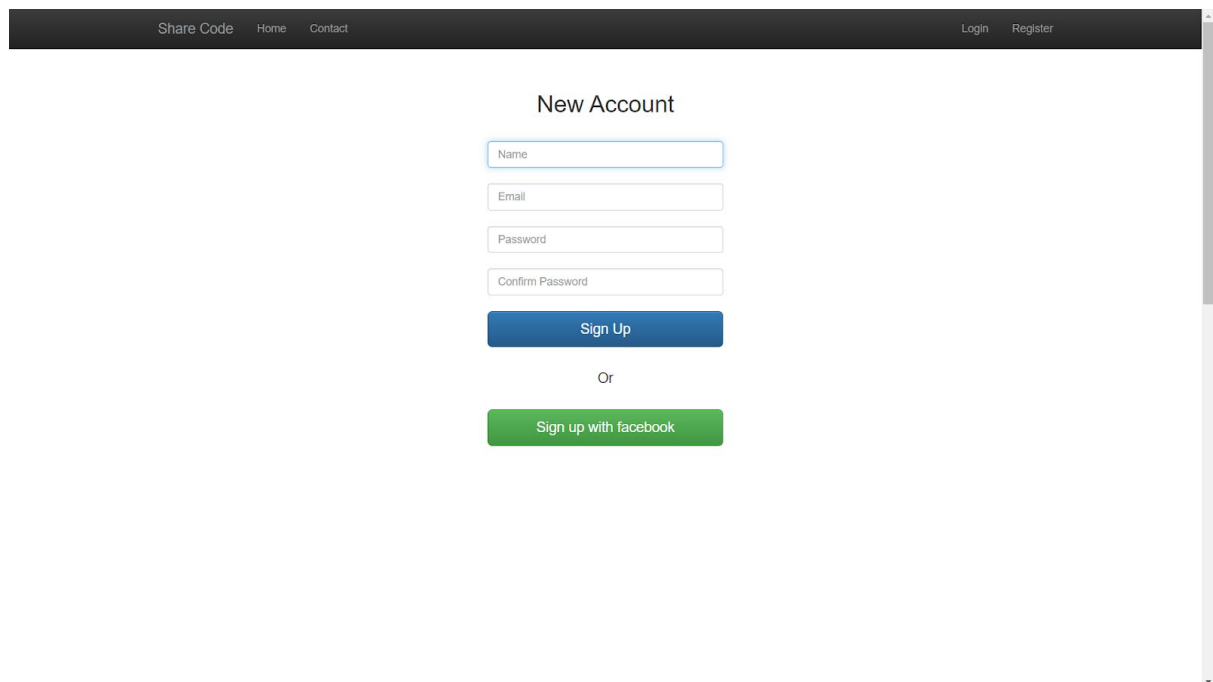
A collaborative whiteboard will also be created with the help of socket functionality and it is also included in the meeting page. Using the white board the interviewer can test the interviewee's product design abilities, ability to draw UML diagrams, etc.

CHAPTER 6

IMPLEMENTATION AND RESULTS

With the development of this application, we would be able to make the interview process easier and reduce the wrongdoings taking place during an interview. We implemented features such as collaborative editor, collaborative whiteboard, chat box and audio/video call features. With help of these features, we'd be able to reduce the amount of misconducts taking place in an interview.

6.1 REGISTRATION/LOGIN



The screenshot displays a web application's registration interface. At the top, a dark navigation bar contains links for 'Share Code', 'Home', 'Contact', 'Login', and 'Register'. The main content area is titled 'New Account' and features a vertical stack of input fields for 'Name', 'Email', 'Password', and 'Confirm Password'. Below these fields is a blue 'Sign Up' button. Underneath the button is the text 'Or', followed by a green button labeled 'Sign up with facebook'. The interface is clean and modern, with a white background and subtle shadows.

Fig 6.1.1 Registration

The screenshot shows a web page with a dark header bar. On the left, there are links for 'Share Code', 'Home', and 'Contact'. On the right, there are links for 'Login' and 'Register'. The main content area is white and contains the text 'Please Sign In' centered. Below this text are two input fields: 'Email' and 'Password'. Below the 'Password' field is a blue button labeled 'Sign In'. Below the 'Sign In' button is the text 'Or'. Below 'Or' is a green button labeled 'Sign In with facebook'.

Fig 6.1.2 : Login

6.2 CREATE NEW MEET

The screenshot shows a web page with a dark header bar. On the left, there are links for 'Share Code', 'Home', and 'Contact'. On the right, there is a user profile 'Rahul Raj' with a dropdown arrow. The main content area is white and contains the text 'Share Code - Online Interview Platform' followed by 'Welcome to Share Code - Online Interview Platform'. Below this text is a blue button labeled 'Create New Task'.

Fig 6.2.1 : Interviewer homepage

6.3 REAL TIME MEET

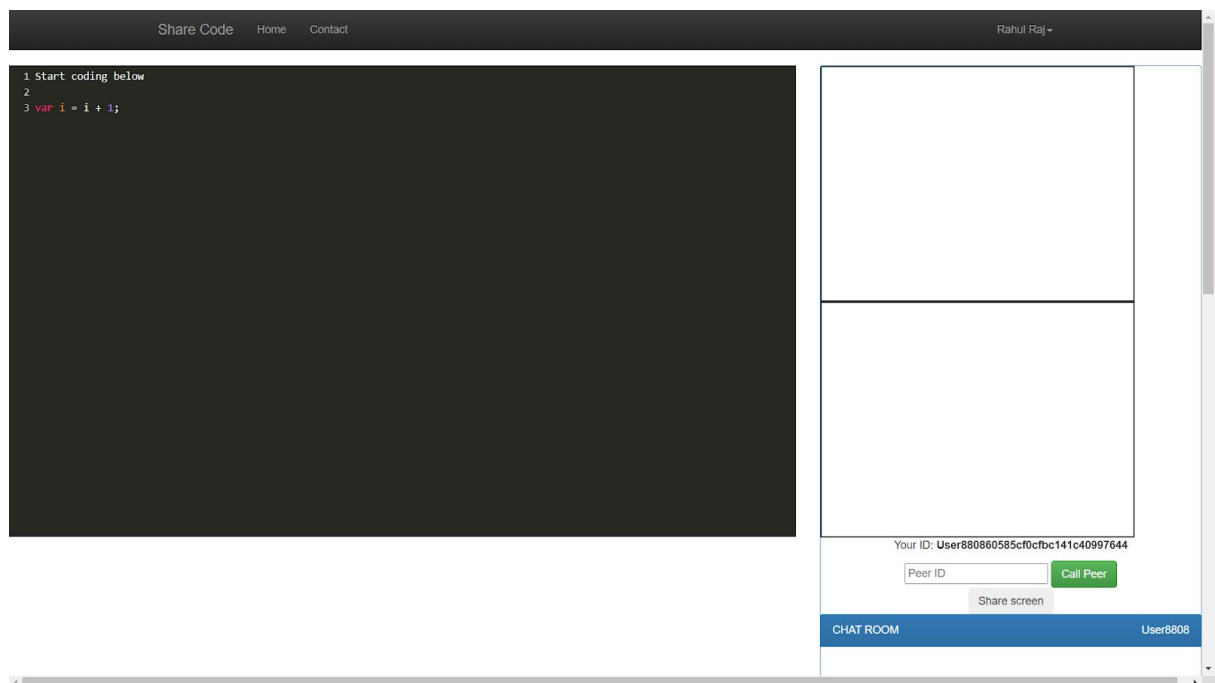


Fig 6.3.1 : Editor and Video Call

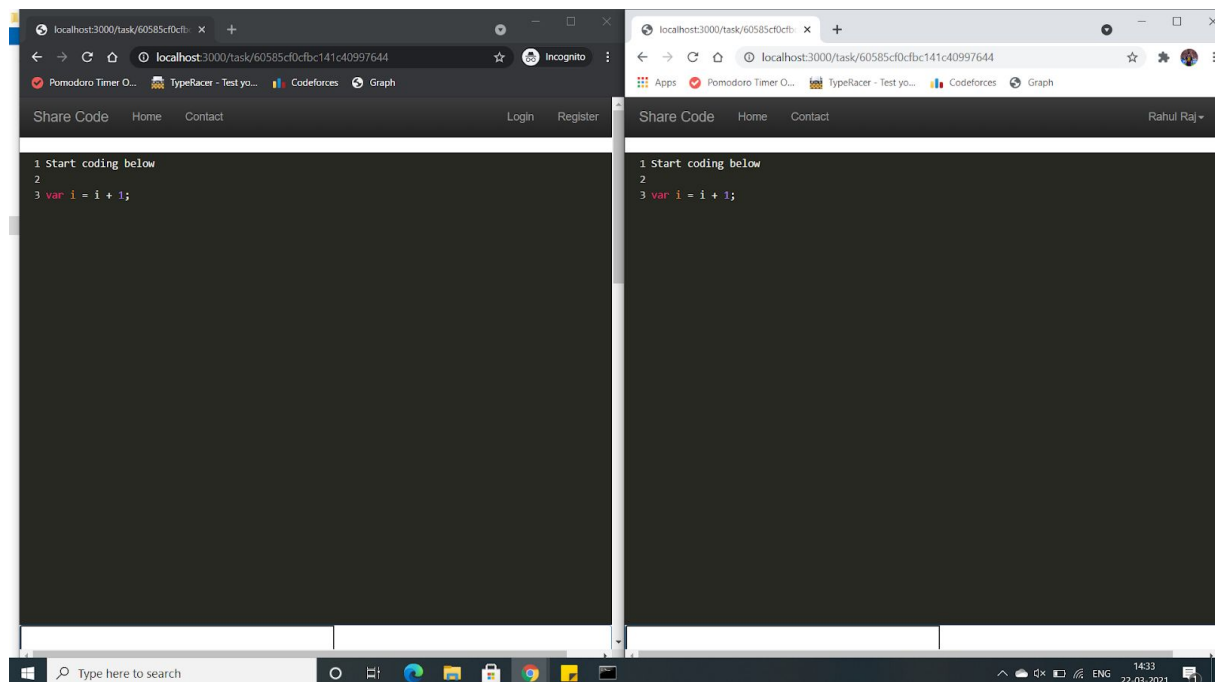


Fig 6.3.2 : Real time collaborative editor

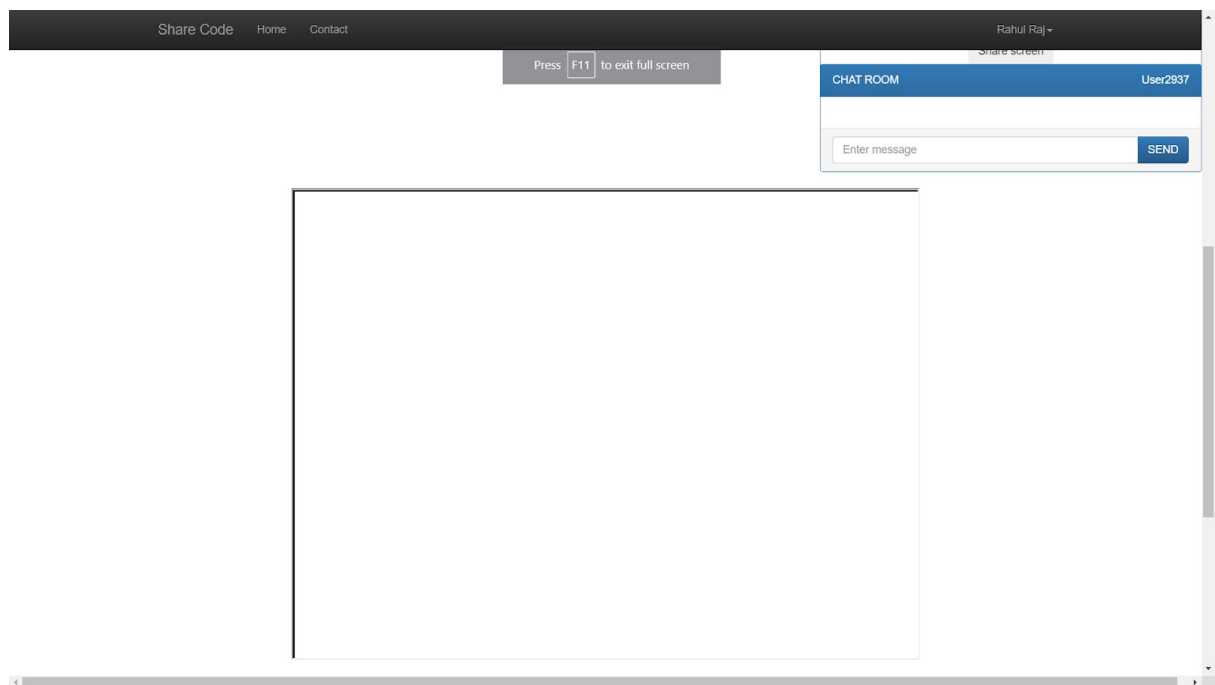


Fig 6.3.3 : Collaborative whiteboard and Chat box

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 CONCLUSION

The major goal of this application is to make the process of conducting interviews easier. By using this application we would be able to reduce the malpractices taking place during an interview. This application would be useful during the times of COVID-19 or even after it to make it easy for both the interviewer and interviewee by conducting the interview online and making interviews hassle free.

We have included several features in this application such as a collaborative code editor allowing both the interviewer and interviewee to collaborate while coding. Video/Audio camera facility to allow both of them to connect with each other and look at their movements. A collaborative whiteboard to handle design questions asked during interviews and a chat box to type and send questions. With all these features, we will be able to reduce the cheating taking place during interviews and making it easier and time efficient.

7.2 FUTURE ENHANCEMENT

Future works will aim at including a compiler along with the editor to compile and get the output of the program at the same time on the screen and improve the support of the editor to more programming languages. By improving the usability of the whiteboard by adding more shapes along with the pencil, and to include code generation features from diagrams

APPENDIX 1 – CODE

SAMPLE CODE

```
<div>
  <div class="row">
    <div class="col-md-8">
      <textarea id="code-screen">{{content}}</textarea>
    </div>
    <div class="col-md-4" id="chatbox">
      <div class="panel panel-primary">
        <!-- VIDEO CALL -->
        <div id="video-container">
          <div style="width: 400px; height:300px; border:2px solid;"
id="ourVideo"></div>
          <div style="width: 400px; height:300px; border:2px solid;"
id="remoteVideo"></div>
          <div id="step2">
            <p>Your ID: <span id="my-id">...</span></p>
            <div class="form-inline">
              <input id="peerId" placeholder="Peer ID">
              <button id="call-peer" class="btn btn-success">Call
Peer</button>
              <br>
              <button id="shareScreen" class="btn btn-secondary">Share
screen</button>
            </div>
          </div>
          <div id="step3">
          </div>
        </div>
        <!-- CHAT ROOM -->
        <div class="panel-heading">
          CHAT ROOM
          <span class="pull-right" id="chatbox-username">
            {{#if user}}
            {{user.name}}
            {{/if}}
          </span>
        </div>
      </div>
    </div>
  </div>
```

```

    </div>
    <div class="panel-body">
        <ul class="media-list" style="height: 300px; overflow-y: scroll"
id="chatbox-listMessages">

            </ul>
        </div>
        <div class="panel-footer">
            <div class="input-group">
                <input type="text" class="form-control" placeholder="Enter
message" id="userMessage" />
                <span class="input-group-btn">
                    <button type="button" class="btn btn-primary"
onclick="sendMessage()">SEND</button>
                </span>
            </div>
        </div>
    </div>
    <div id="board">
        <iframe src="http://localhost:3010" width="800" height="600"
scrolling="no"> </iframe>
    </div>
</div>
<input type="hidden" value="{ {roomId} }" id="roomId" />

<script
src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.25.0/codemirror.min.
js"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.25.0/mode/javascript/
javascript.min.js"></script>

<script src="/plugins/ot/ot.js"></script>
<script src="/socket.io/socket.io.js"></script>
<script
src="https://rocky-depths-74216.herokuapp.com/peer/peer.js"></script>

<script>
    var EditorClient = ot.EditorClient;
    var SocketIOAdapter = ot.SocketIOAdapter;

```

```

var CodeMirrorAdapter = ot.CodeMirrorAdapter;

var socket = io.connect();
var editor =
CodeMirror.fromTextArea(document.getElementById("code-screen"), {
    lineNumbers: true,
    theme: "monokai"
});

var code = $('#code-screen').val();
var cmClient;
function init(str, revision, clients, serverAdapter) {
    if (!code) {
        editor.setValue(str);
    }

    cmClient = window.cmClient = new EditorClient(
        revision, clients, serverAdapter, new CodeMirrorAdapter(editor)
    );
};

socket.on('doc', function (obj) {
    init(obj.str, obj.revision, obj.clients, new SocketIOAdapter(socket));
});

var username = $("#chatbox-username").val();
if (username === "") {
    var userId = Math.floor(Math.random() * 9999).toString();
    username = "User" + userId;
    $("#chatbox-username").text(username);
};

var roomId = $('#roomId').val();
socket.emit('joinRoom', { room: roomId, username: username });

var userMessage = function (name, text) {
    return ('<li class="media"> <div class="media-body"> <div
class="media">' +
        '<div class="media-body"' +
        '<b>' + name + '</b> : ' + text +
        '<hr/> </div></div></div></li>'
    );
};

```

```

};

var sendMessage = function () {
  var userMessage = $('#userMessage').val();
  socket.emit('chatMessage', { message: userMessage, username: username
});
  $('#userMessage').val("");
};

socket.on('chatMessage', function (data) {
  $('#chatbox-listMessages').append(userMessage(data.username,
data.message));
});

// PeerJS
window.addEventListener('load', (event) => {
  var peer = new Peer(username + roomId, {
    host: 'rocky-depths-74216.herokuapp.com',
    path: '/peerjs',
    port: 443,
    secure: true,
    key: 'code4startup',
    debug: true
  });
  var myStream;
  var currentPeer;
  var peerList = [];
  peer.on('open', function (id) {
    document.getElementById("my-id").innerHTML = id;
  })
  peer.on('call', function (call) {
    navigator.mediaDevices.getUserMedia({
      video: true,
      audio: true
    }).then((stream) => {
      myStream = stream
      addOurVideo(stream);
      call.answer(stream)
      call.on('stream', function (remoteStream) {
        if (!peerList.includes(call.peer)) {
          addRemoteVideo(remoteStream);
          currentPeer = call.peerConnection;

```

```

        peerList.push(call.peer);
    }
})
}).catch((err) => {
    console.log(err + " unable to get media.")
})
})

document.getElementById("call-peer").addEventListener('click', (e) => {
    let remotePeerId = document.getElementById("peerId").value;
    callPeer(remotePeerId);
})

document.getElementById("shareScreen").addEventListener('click', (e)
=> {
    navigator.mediaDevices.getDisplayMedia({
        video: {
            cursor: "always"
        },
        audio: {
            echoCancellation: true,
            noiseSuppression: true
        }
    }).then((stream) => {
        let videoTrack = stream.getVideoTracks()[0];
        videoTrack.onended = function () {
            stopScreenShare();
        }
        let sender = currentPeer.getSenders().find(function (s) {
            return s.track.kind === videoTrack.kind
        })

        sender.replaceTrack(videoTrack);
    }).catch((err) => {
        console.log("Unable to get display media" + err)
    })
})

function callPeer(id) {
    navigator.mediaDevices.getUserMedia({
        video: true,
        audio: true
    })
}

```

```

    }).then((stream) => {
      myStream = stream;
      addOurVideo(stream);
      let call = peer.call(id, stream);
      call.on('stream', function (remoteStream) {
        if (!peerList.includes(call.peer)) {
          addRemoteVideo(remoteStream);
          currentPeer = call.peerConnection;
          peerList.push(call.peer);
        }
      })
    }).catch((err) => {
      console.log(err + " Unable to get media");
    });
  }

```

```

function stopScreenShare() {
  let videoTrack = myStream.getVideoTracks()[0];
  var sender = currentPeer.getSenders().find(function (s) {
    return s.track.kind == videoTrack.kind;
  })
  sender.replaceTrack(videoTrack)
}

```

```

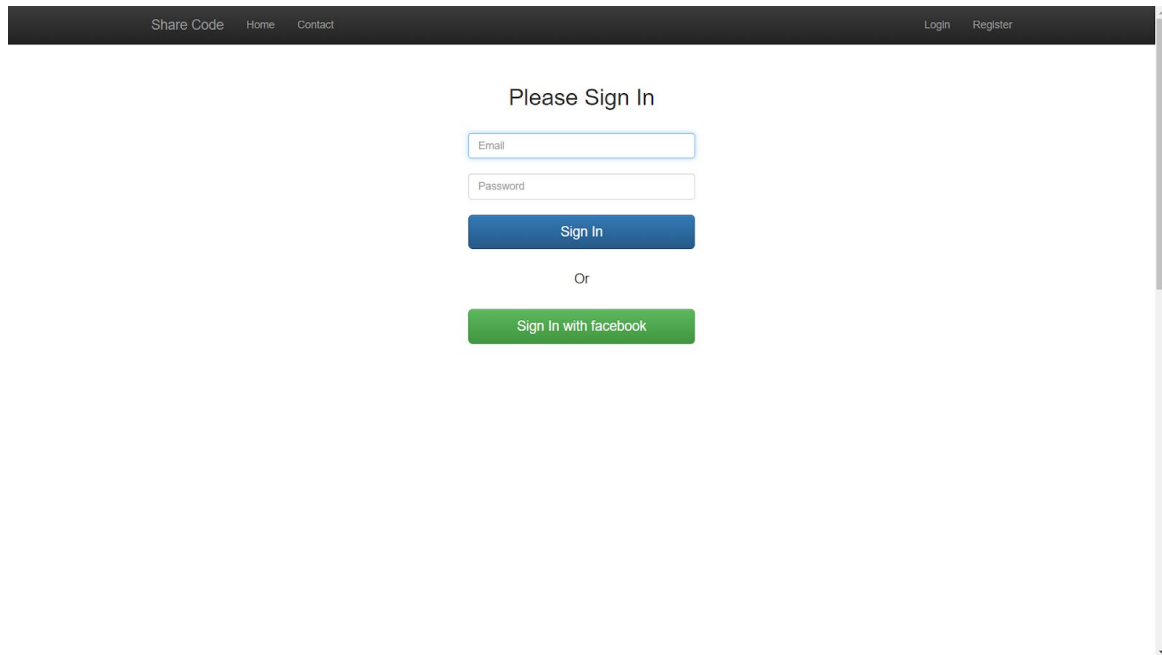
function addRemoteVideo(stream) {
  let video = document.createElement("video");
  video.classList.add("video");
  video.srcObject = stream;
  video.play();
  document.getElementById("remoteVideo").append(video);
}
function addOurVideo(stream) {
  let video = document.createElement("video");
  video.classList.add("video");
  video.srcObject = stream;
  video.play();
  document.getElementById("ourVideo").append(video);
}
});

```

</script>

APPENDIX 2 – OUTPUT SCREENSHOTS

LOGIN PAGE



The screenshot shows a web browser window displaying a login page. At the top, a dark navigation bar contains links for 'Share Code', 'Home', 'Contact', 'Login', and 'Register'. The main content area is titled 'Please Sign In'. It features a form with two input fields: 'Email' and 'Password'. Below these fields is a blue 'Sign In' button. Underneath the button is the text 'Or', followed by a green button labeled 'Sign In with facebook'. A vertical scrollbar is visible on the right side of the page.

Share Code Home Contact Login Register

Please Sign In

Email

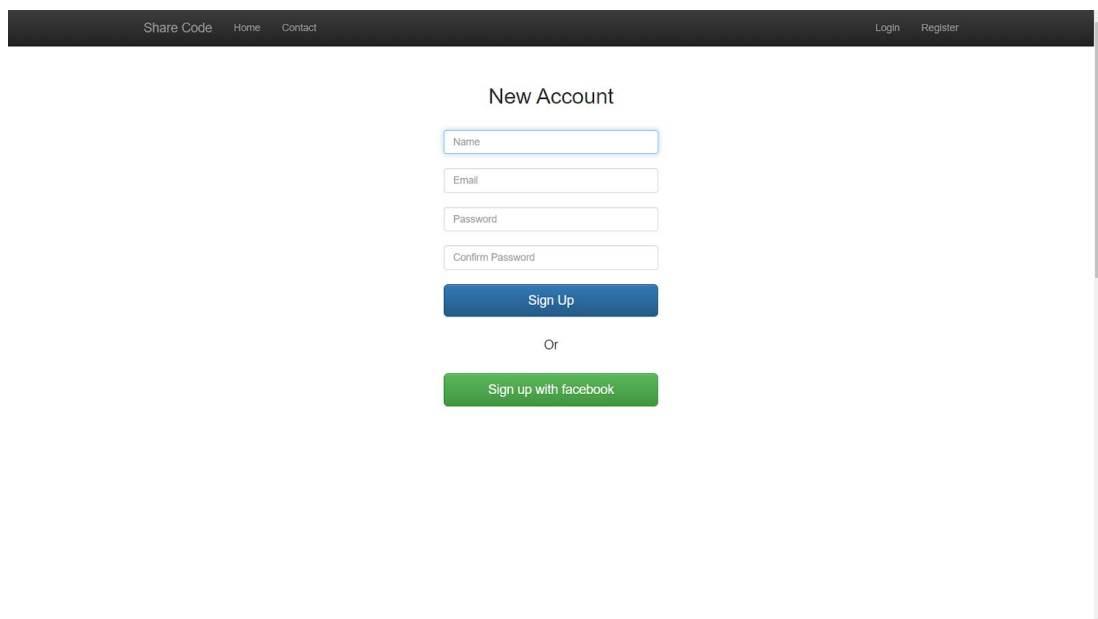
Password

Sign In

Or

Sign In with facebook

SIGN UP PAGE



The screenshot shows a web browser window displaying a sign-up page. At the top, a dark navigation bar contains links for 'Share Code', 'Home', 'Contact', 'Login', and 'Register'. The main content area is titled 'New Account'. It features a form with four input fields: 'Name', 'Email', 'Password', and 'Confirm Password'. Below these fields is a blue 'Sign Up' button. Underneath the button is the text 'Or', followed by a green button labeled 'Sign up with facebook'. A vertical scrollbar is visible on the right side of the page.

Share Code Home Contact Login Register

New Account

Name

Email

Password

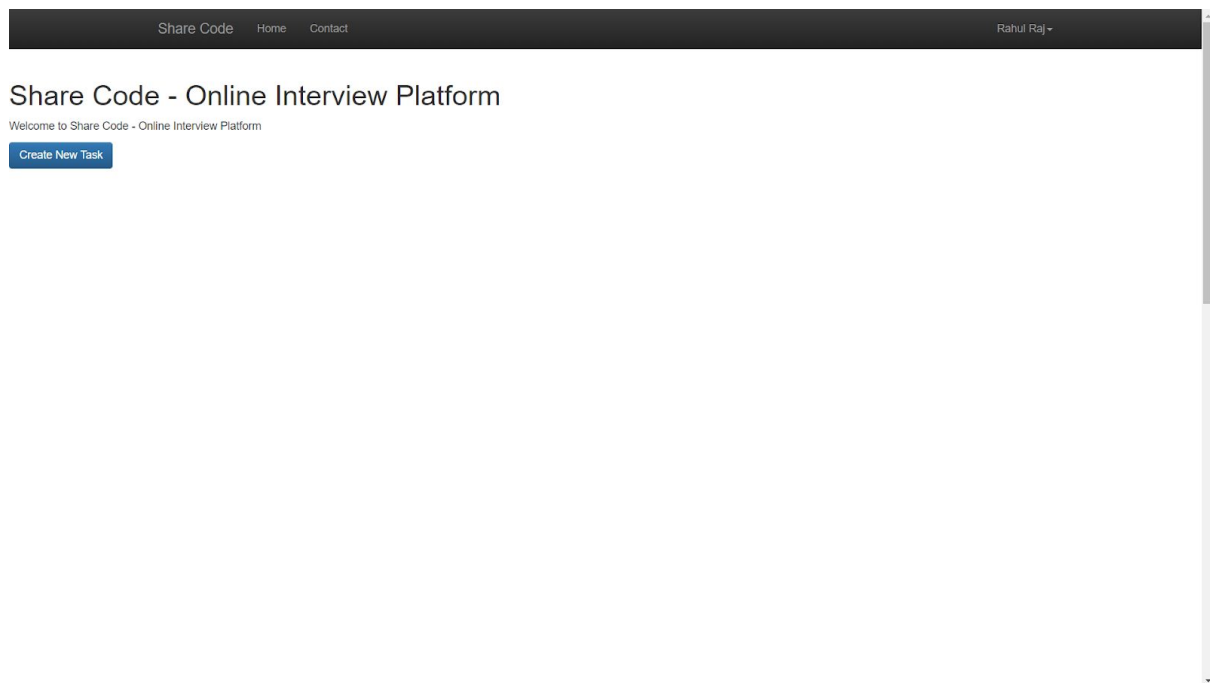
Confirm Password

Sign Up

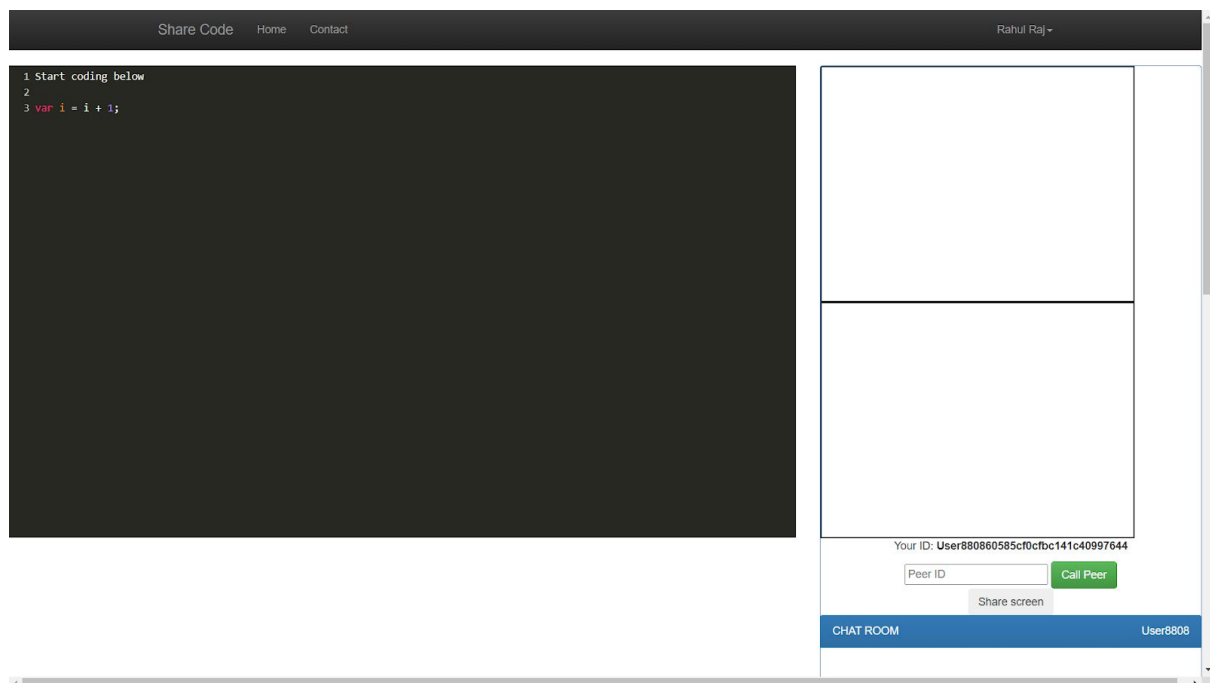
Or

Sign up with facebook

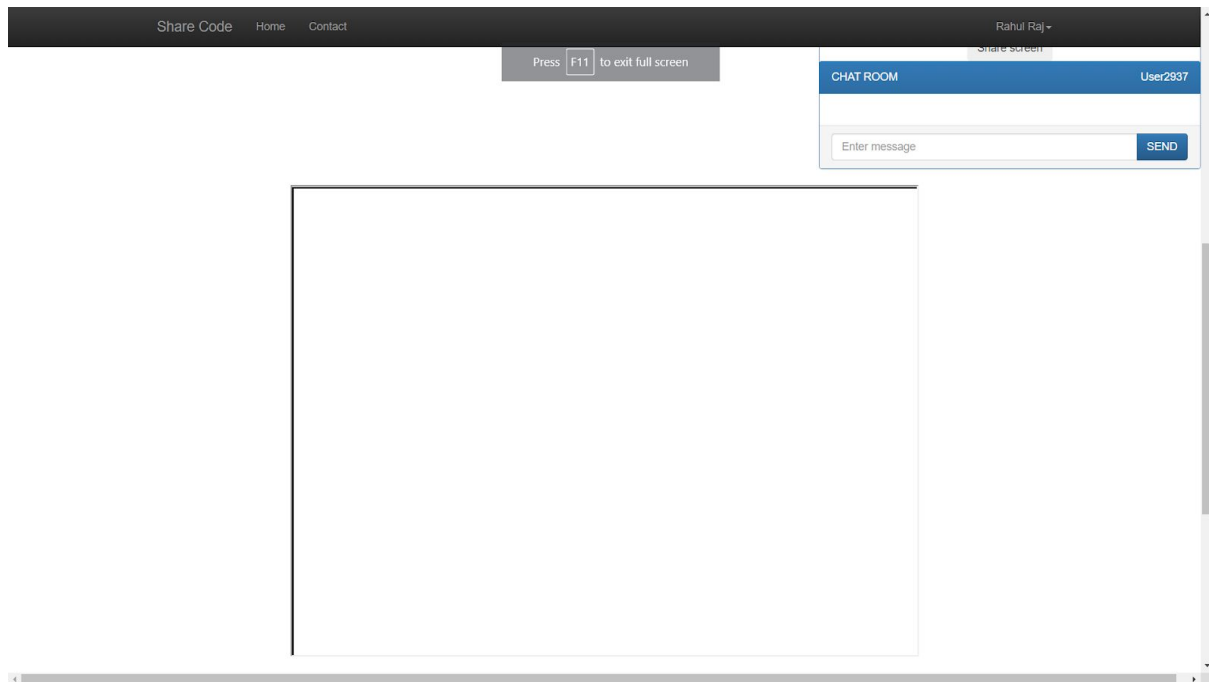
INTERVIEWER HOMEPAGE



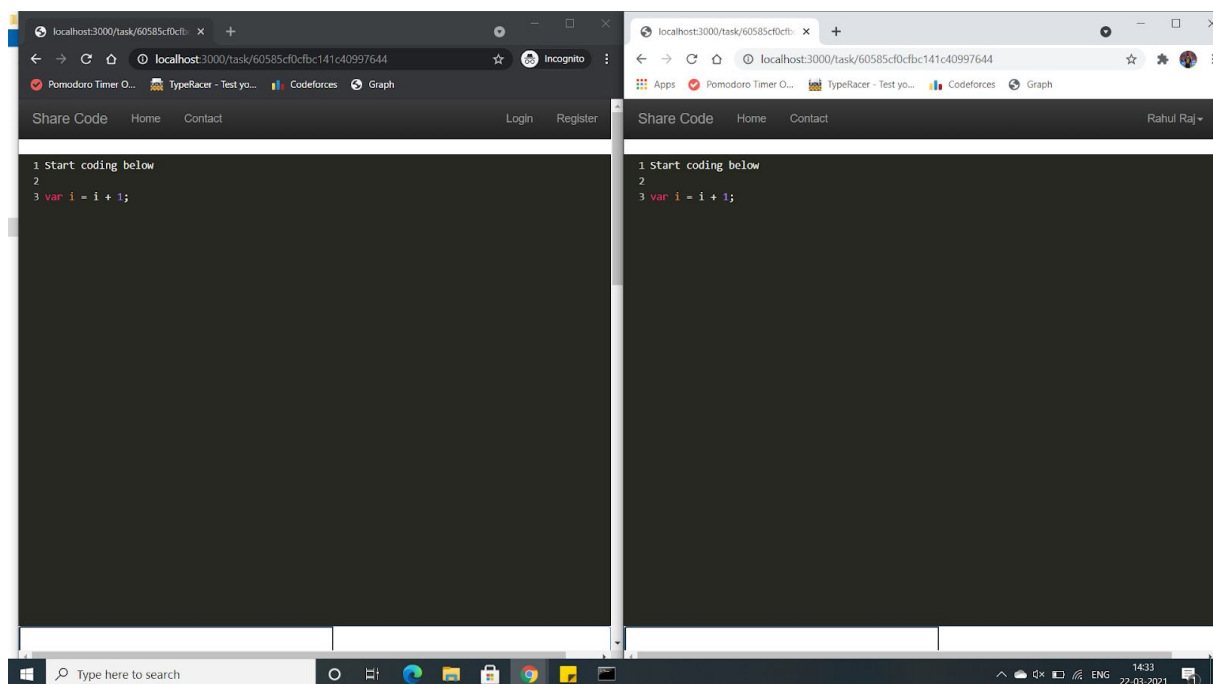
EDITOR AND VIDEO CALL



COLLABORATIVE WHITEBOARD & CHAT BOX



COLLABORATIVE EDITING



REFERENCES

- [1] Max Goldman, “Software development with Real time collaborative editing”, August 2012.
- [2] Mayank Patel, “Online Java Compiler using Cloud Computing”, January 2013.
- [3] Julian Jang-Jaccard, Surya Nepal, Branko Celler, Bo Yan, “WebRTC-based video conferencing service for telehealth”, September 2014.
- [4] Antonios G Nanos, Anne E James , “A Virtual Meeting System for the New Age”, September 2013.
- [5] KV Rop, NK Bett, “Video conferencing and its application in distance learning”, June 2012.
- [6] Ravinder Singh, Sowmya Awasthi, “Updated comparative analysis on video conferencing platforms - Zoom, Google Meet, Microsoft Teams, WebEx Teams, Go To Meetings”, August 2020
- [7] Quang-Vinh Dang, Claudia-Lavinia Ignat, “Performance of real-time collaborative editors at large scale: User perspective”, May 2016
- [8] David Sun, Steven Xia, Chengzheng Sun, David chen, “Operational transformation for collaborative word processing”, November 2014.
- [9] Du Li, Rui Li, “An Admissibility-Based Operational Transformation Framework for Collaborative Editing Systems”, February 2010.
- [10] Chengzheng sun, Clarence Ellis, “Operational transformation in real-time group editors: issues, algorithms, and achievements”, November 2008.

- [11] Ajay Khunteta, “A Survey on Operational Transformation Algorithms: Challenges, Issues and Achievements”, July 2010
- [12] Google Docs : <https://www.google.com/docs/about/>
- [13] Jon Martin Denstadli, Tom Erik Julsrud, Randi Hjorthol, “Videoconferencing as a Mode of Communication: A Comparative Study of the Use of Videoconferencing and Face-to-Face Meetings”, January 2012.
- [14] Xinggong Zhang, Yang Xu, Hao Hu, Yong Liu, Zongming Guo, Yao Wang, “Modeling and Analysis of Skype Video Calls: Rate Control and Video Quality”, October 2013.
- [15] Sanabil A. Mahmood, Ergun Ercelebe, ”Development of video conference platform based on Web RTC”, June 2018
- [16] Aditya Kurniawan, Christine Soesanto, Joe Erik, Carla Wijaya, ”CodeR: Real-time Code Editor Application for Collaborative Programming”, 2015
- [17] Francois Pacull, Alain Sandoz, Andre Schiper, “Duplex : A distributed collaborative editing environment”, October 1994