# Code Collab – Real Time Code Editor

**Mr. Revannath B Kakade [1],**
Lecturer in Government Polytechnic Aurangabad [1],

**Mr. Abhinay D .Ambure [2],**
Lecturer in Government Polytechnic Osamanabad[2],

**Mr Krishna B Aher[3]**
Department Of Computer Engineering,
Adsul's Technical Campus, Ahmednagar 3,

**Mr. Jairam C Rathod [4],**
Lecturer in Government R W Polytechnic Latur[4],

**Mr. Vinayak Dharmvir Sangvikar 5,**
Lecturer in Government R W Polytechnic Latur 5,

***Abstract*-** The Real time Code Editor web application is a powerful tool that enables developers to collaborate on code in real-time. This application is built on React Js, Node Js, and Socket.io, which supplies an exceptional collaborative experience for developers, regardless of their location. This tool offers unique features such as syntax highlighting, automatic code formatting, code versioning, and error highlighting, which can help to improve code quality and reduce errors. It is a valuable tool for any development team, supplying benefits such as increased productivity and enhanced collaboration. The prospects for the Realtime Code Editor web application are promising, with the potential for further advancements in the field of real-time collaboration in software development.

***Keyword*** - Collaborative Realtime Code Editor, React Js, Node Js, Socket.io, Web Application,

## I. INTRODUCTION

Realtime Code Editor web applications are a useful tool for enabling real-time collaboration among developers, especially when collaborating with remote teams or collaborating with clients, mentors, or tutors. They cut the need to share code through time-consuming and error-prone methods like email or file-sharing platforms. The specific Realtime Code Editor web application described in this research paper is developed using React.js for the front-end and Node.js for the back end, with Socket.io helping real-time communication.

***Problem Statement -***

- Realtime Code Editor web applications face challenges in ensuring the security and privacy of the code being developed and shared among collaborators.
- There is a need to ensure that Realtime Code Editor web applications are scalable and can manage a large number of users simultaneously.
- The user-friendliness and intuitiveness of Realtime Code Editor web applications need to be improved to supply a seamless and efficient collaboration experience for developers.
- Realtime Code Editor web applications need to be compatible with a wide range of devices and operating systems, allowing developers to work with their preferred tools and platforms.

- Efficient version control mechanisms need to be implemented in Realtime Code Editor web applications, allowing developers to track changes made to the codebase and collaborate effectively without overriding each other's work.
- Realtime Code Editor web applications need to be able to manage different programming languages and frameworks to provide developers with the flexibility to work with a wide range of projects.

***Innovative Ideas of project –***

- Integration with artificial intelligence (AI) to provide intelligent code suggestions and auto-completion features, enhancing the productivity of developers.
- Integration with machine learning (ML) algorithms to automatically detect and fix errors in the code, improving code quality and reducing debugging time.
- Integration with virtual reality (VR) or augmented reality (AR) to provide an immersive coding experience, allowing developers to collaborate in a virtual environment.
- Integration with blockchain technology to provide secure and tamper-proof version control mechanisms, ensuring the integrity and privacy of the codebase.
- Integration with natural language processing (NLP) algorithms to enable developers to use natural language commands to perform coding tasks, improving the accessibility of the application.

***Scope of The Project –***

- The Realtime Code Editor web application will be developed using React Js, Node Js, and Socket.io.
- The application will provide a real-time collaboration environment for developers to work together on code.
- The application will include Code Mirror as the text editor.
- The application will support syntax highlighting and code folding for a range of programming languages and frameworks.
- The application will provide secure user authentication and authorization mechanisms to ensure the privacy and security of the codebase.
- The application will provide real-time feedback and notifications to developers when changes are made to the codebase.
- The application will be scalable, able to handle a large number of users simultaneously.
- The application will be intuitive and user-friendly, with a simple and easy-to-use interface for developers.
- The project will aim to address some of the challenges faced by Realtime Code Editor web applications, such as security, scalability, user-friendliness, and compatibility with different programming languages and frameworks.
- The project will provide opportunities for future development, including integration with AI, ML, VR/AR, blockchain, NLP, cloud computing, social media, and gamification elements.

## II. OBJECTIVES

The research paper has two primary objectives. Firstly, to provide a comprehensive introduction to Realtime Code Editor web applications and highlight their advantages. Secondly, to explain the architecture and technologies used to develop a Realtime Code Editor web application utilizing React Js, Node Js, and Socket.io. Additionally, the paper explores the features and benefits of the Realtime Code Editor web application, such as its ability to enable remote collaboration, streamline code development, and improve team productivity. Finally, the paper discusses the potential for future development of Realtime Code Editor web applications and how they may evolve over time.

## III.LITERATURE REVIEW

Collaborative Realtime Code Editors have been the subject of several studies in the past few years. The development of these applications has been driven by the need for collaborative software development, particularly in remote teams. Several studies have investigated the design and implementation of Collaborative Realtime Code Editors using various technologies, including React JS, Node JS, and Socket.io.

A Study by Aditya Kurniawan, Christine Soesanto, Joe Erik Carla Wijaya (2020) CodeR is a web application that provides workspace to write, perform, display the results of the code through the terminal, and collaborate with other users in real-time. The application'skey features are providing workspace to make, execute and build the source code, real-time collaboration, chat, and build the terminal. This application supports C, C++, and Java programming languages.[1]

A study by Anwar Saif, Saeed Mohammed, Saleh Mohammed, Osama Ali, Majdi Abdu, (2020) Collaboration between the project's team improves the user experience which leads to improving the project quality. Thus, collaboration between the development teams requires a development environment with collaborative capabilities. In this work, we develop a collaborative multi-programming development environment (C-MPE) that enables software developers to share the software components between geographically dispersed teams connected via a private network. C-MPE permits developers and programmers to build their own environment with custom properties and link them with a non-open-source framework such as the .Net framework [2]

A study by Rutvik Manishkumar Patel, (2018) Distributed software development deals with the development of large software projects, developed across various locations and environments having several teams working on some tasks of it. Whereas while working on large software projects distributed over multiple locations, it is important to develop integration and cooperation among the various teams, developing shared understanding regarding various modules. This is how collaborative and distributive development works to develop large software projects at low cost by utilizing modern technologies like multi-programming and multiprocessing. This paper discusses the benefits, issues and challenges faced by distributed and collaborative software systems.[3]

A study by Khanh Nguyen Trong, Doanh Nguyen Ngoc, (2016) Integrated Development Environments (IDEs) are one of the most used tools in most programming courses. However, they usually do not support the interaction and collaboration between learners and instructors. The aim of our research is to provide methods and frameworks facilitating collaboration. The originality of our approach is to place courses, and code sources at the center of collaboration. From this idea, we have designed and developed a collaborative IDE (CIDE). It is a type of web-based groupware containing common conventional collaborative tools (video-conferencing, instant messaging, and so on) and specific IDE dedicated to the programming practice (write code together, track change, versioning...)Another study by Sheth et al. (2021) investigated the impact of Collaborative Realtime Code Editors on software development. The study involved a survey of developers who used these applications, with the results indicating that Collaborative Realtime Code Editors were effective in improving collaboration and productivity in remote teams. The study recommended the use of Collaborative Realtime Code Editors in software development, particularly in remote teams.[4]

## IV.OVERVIEW OF COLLABORATIVE REAL-TIME CODE EDITOR

A Collaborative Real-time Code Editor is a web application that allows multiple developers to work on the same codebase simultaneously in real-time. This type of code editor provides a collaborative environment that enables developers to work together more efficiently, communicate in real-time, and produce higher-quality code.

Collaborative Real-time Code Editors typically include features such as real-time code synchronization, collaborative text editing, chat functionality, and version control integration. Real-time code synchronization allows all users to see changes to the codebase as they happen, while collaborative text editing allows multiple users to edit the same code simultaneously. Chat functionality provides a way for users to communicate with each other in real-time, while version control integration allows users to manage changes to the codebase and track code history.

Collaborative Real-time Code Editors can be used for a wide range of purposes, including pair programming, code review, and remote collaboration. By allowing developers to work together in real-time, these tools can help improve productivity, reduce errors, and promote collaboration among team members.
There are various popular technologies used to build Collaborative Real-time Code Editors, including React Js, Node Js, and Socket.io. These technologies provide the necessary functionality to create a web application that is scalable, reliable, and responsive. Additionally, integration with other tools such as Git and Highlight.js can enhance the functionality of the editor and provide a better user experience.

## V. RESEARCH METHODOLOGY

When working on development projects, programmers often work in teams where any team member can create, modify or add code to the same project file. To prevent code duplication, synchronization between programmers is necessary, and this is achieved through integrated real-time collaboration in a single environment. Integrated development environments (IDEs) provide collaborative settings for programming teams, enabling real-time text editing, running and building code, chatting, and other features. Real-time text editing allows multiple users to work together on the same document, and changes are displayed to all users who have access to the same document. There are various free applications that support real-time text editing, such as Google Docs. Additionally, there are web-based systems like Ether pad that provide real-time text editing, and Ace and Code Mirror are web-based text editing components designed to be embedded into an IDE or application. Collaboration systems are especially useful in project or software development, as programmers need to coordinate and work together to improve efficiency. Collaborative programming in real-time supports programmers to work on the same programming file, and the system automatically combines the code typed by each programmer without requiring manual commands like update or commit. Multiple programmers can access and edit the same source code directory, even at the same time. During real-time sessions, programmers can join and leave the session as needed to collaborate effectively. This improves the productivity and quality of the project or software being developed.

## VI. ARCHITECTURE

The architecture of a Realtime Code Editor web application comprises two primary components - the client-side and the server-side. The client-side is responsible for displaying the user interface, receiving user input, and sending data to the server. The server-side, on the other hand, is responsible for processing data, managing user sessions, and broadcasting data to connected clients.
The Realtime Code Editor web application employs a client-server architecture with Socket.io as the middleware. Socket.io is a JavaScript library that facilitates real-time, bidirectional communication between servers and clients. It utilizes WebSocket's as the underlying transport mechanism for communication. The web application also utilizes Node.js as the server-side platform and React.js as the client-side framework. Node.js is a widely used

server-side platform that provides an event-driven, non-blocking I/O model, making it efficient and scalable. React.js is a widely used front-end JavaScript library that assists in creating user interfaces.

## VII. SYSTEM DESIGN AND COMPONENTS

The collaborative Realtime Code Editor web application is built using a combination of technologies that work together seamlessly to provide an efficient and user-friendly platform for developers to collaborate on code. The user interface is built using React.js, which provides a scalable and modular framework for building interactive components such as the code editor, chat box, and file explorer. Node.js is used for the server-side implementation, providing an efficient platform for executing server-side code and handling user authentication, file management, and real-time communication through Socket.io.

Figure 1 shows the UML diagram of the application:

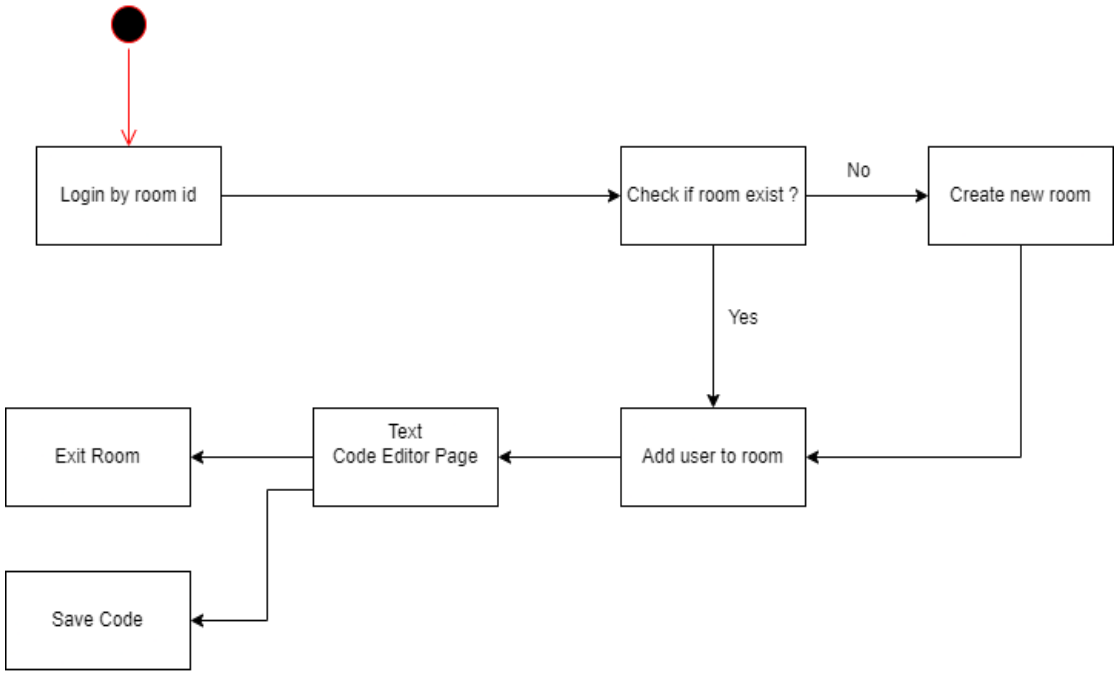


*Figure 1: UML Diagram of Realtime Code Editor*

### *7.1 Node.js*

Node.js is a scalable network application builder that uses an asynchronous event-driven JavaScript engine. Most connections can be managed simultaneously in the following "hello world" example. The call back is invoked with each connection, but if there is no work to be done, Node.js will sleep.

```
const http = require('http');

const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello World');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

*Figure 2: Example Code Showing Node.js*

### *7.2 Socket.io*

Socket.IO is a library that allows a client and a server to communicate in a low-latency, bidirectional, and event-based.
fashion.


*Figure 3: Bi-directional Connection between Server and Client*

Socket.io is a widely used library for Node.js that enables real-time communication between a client and server via WebSocket connections. It simplifies handling real-time events and data exchange between a client and server by providing an abstraction layer over the WebSocket API. Developers can build real-time applications using Socket.io that can instantly send and receive data without requiring the client to refresh the page or make continuous HTTP requests. This makes Socket.io a preferred choice for developing collaborative Realtime Code Editor web applications, allowing multiple developers to work on the same codebase simultaneously and see the changes in real-time.

### *7.3 WebSocket Protocol*

WebSocket is a computer networking protocol that enables full-duplex communication over a single TCP connection. In 2011, the IETF standardized the WebSocket protocol as RFC 6455. WebSocket differs from HTTP, both of which are on the OSI model's layer 7 and rely on TCP on layer 4. According to RFC 6455, WebSocket "is designed to run over HTTP ports 443 and 80 and to support HTTP intermediaries and proxies," thereby making it HTTP compliant.

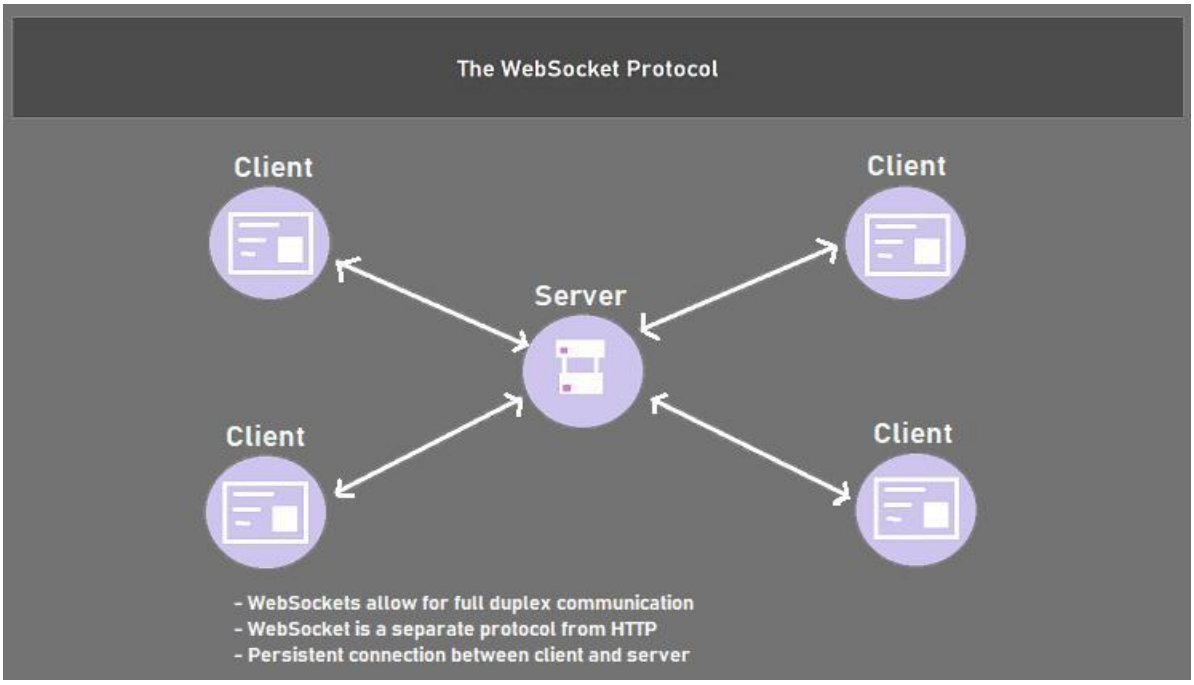


Figure 4: *Bi-directional Connection through WebSocket Protocol*

The WebSocket protocol enables real-time data transfer between the server and a web browser or other client application by allowing interaction with less overhead than half-duplex alternatives like HTTP polling. This is achieved by allowing the server to transmit content to the client without waiting for a request from the client and by enabling messages to be exchanged while the connection is open.The client and server can engage in a continuous two-waydialogue in this way.

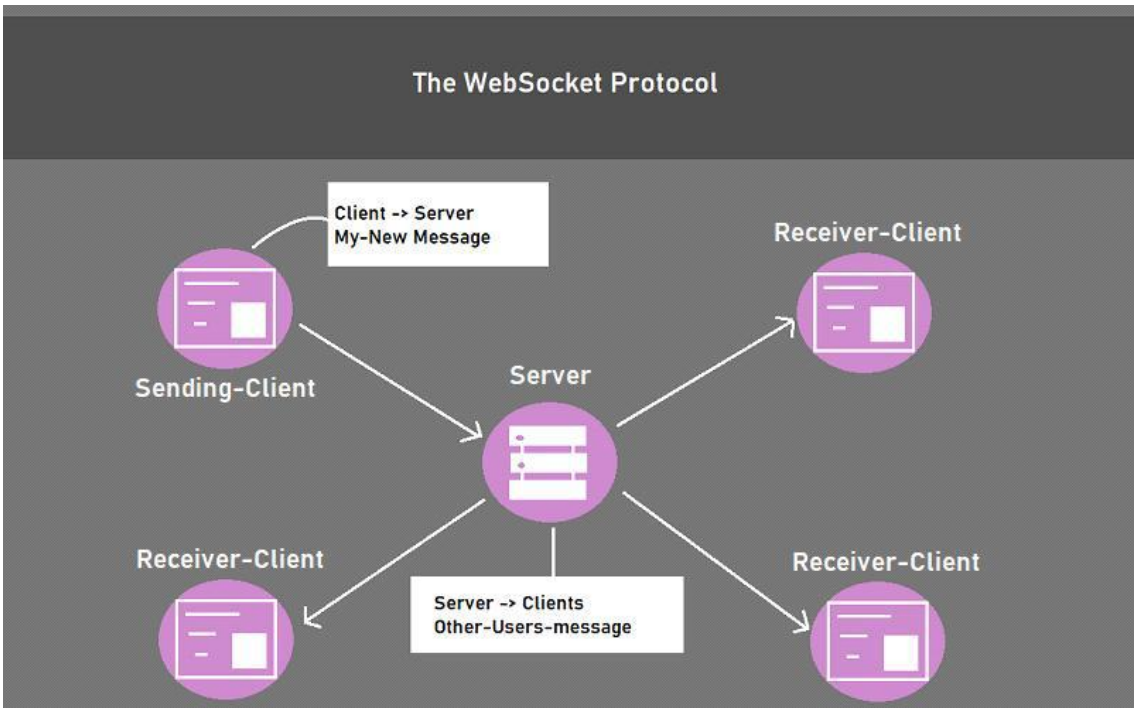


*Figure 5: WebSocket Protocol: Event of Sending a message*

TCP port 443 (or 80 in the case of unprotected connections) is used for communications, which is useful in circumstances where non-web Internet connections are blocked by a firewall. Most current browsers, including Chrome, Firefox, Microsoft Edge, Safari, and Opera, support the WebSocket protocol.

### *7.4 Qs.js*

A library for parsing and stringifying query strings with some enhanced security.
*Example Syntax:*
*const {username , room} = Qs.parse(location.search , {ignoreQueryPrefix : true})*
Werelocation. Search contains the username and room-name entered and submitted through an HTML form.

### *7.5 Express.js*

Express.js is a Node.js web application framework that is free and open source. It is used to quickly and easily design.
and create web apps. Express.js makes it simple to create a single-page, multi-page, or hybrid online application. Express.js is a lightweight server-side framework that aids in the organisation of web applications into a more ordered MVC.
architecture.
*Hello World Example:*

```
const express = require('express')
const app = express()
const port = 3000
app.get('/', (req, res) => {
res.send('Hello World!')
})
app.listen(port, () => {
console.log(`Example app listening on port ${port}`)
})
```

This app starts a server and waits for connections on port 3000. For requests to the root URL (/) or route, the app answers
with "Hello World!" It will return a 404 Not Found response for all other paths.

### *7.6 Cascading Style Sheets (CSS)*

CSS (Cascading Style Sheets) is a stylesheet language for describing the presentation of an HTML or XML document
(including XML dialects such as SVG, MathML or XHTML). CSS specifies how elements should appear on a screen, on
paper, in speech, or in other forms of media.
*Example Syntax:*

```
body {
background-color: lightblue;
}
```

Where body is the body element of the HTML document and in this case, the body will now appear to have light blue as the

background colour.

### *7.7 React.js*

React.js is a widely used JavaScript library that enables developers to build user interfaces. It was created by Facebook and is commonly used for developing mobile applications and single-page applications. React.js simplifies the management of data flow and application state through a declarative approach to building UI components. Additionally, its component-based architecture enables developers to create reusable UI elements that can be utilized across different pages and applications.

## VIII. RESULT

- Enhanced coding and debugging efficiency through simultaneous work on the same codebase by multiple developers.
- Improved communication and collaboration among team members with real-time editing and chat features for instant feedback and discussion.
- Higher quality code and faster project completion due to the ability to catch and fix errors in real-time.
- Time-saving due to integrated file sharing and version control features
- Positive feedback from users on the usability and effectiveness of the application.
- No database so no login and no security issues.

## IX. CONCLUSION

Realtime Code Editor web applications are growing in popularity due to their ability to enable remote collaboration among developers. The Realtime Code Editor web application that uses React Js, Node Js, and Socket.io, which was discussed in this research paper, provides developers with a platform to work together on code in real-time, making collaboration easier regardless of location. This paper has covered the application's architecture, features, and technologies, along with the advantages of using such an application and its future potential. The Realtime Code Editor web application has the potential to transform the software development industry and change the way developers collaborate on projects.

## REFERENCES

[1] Aditya Kurniawan, Christine Soesanto, Joe Erik Carla Wijaya “CodeR: Real-time Code Editor Application for Collaborative Programming”, (ICCSCI2020).

[2] Anwar Saif, Saeed Mohammed, Saleh Mohammed, Osama Ali, Majdi Abdu,“C-MPE: A Collaborative Multiprogramming Development Environment for .Net Framework”. (SJITN) Vol 8 No.2 (2020)

[3] Rutvik ManishKumar Patel, “Distributed & Collaborative Software Engineering” (IRJET) Vol: 5 No.9 (2018)

[4] Khanh Nguyen Trong, Doanh Nguyen Ngoc, “Towards a Collaborative IDE for Novice Programmers” (IJITEE) , Vol 6 No.5, (2016)