JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

# Multi User Code Editor

**T. Harshith Kumar, S. Advaith Sai, S. Samuel Anurag, G. Kalyan Reddy, Mr. Mohammed Zaheer Ahmed**

Dept. of Computer Science and Engineering, Vidya Jyothi Institute of Technology, Hyderabad, India

**Abstract -** The world of Internet is growing rapidly, many applications that previously created on the desktop started moving to the web. Many applications could be accessed anytime and anywhere easily using Internet. Developers need tools to create their applications, one of them named code editor. In order to facilitate user collaboration while working on the project, the goal of this research is to design and construct a real-time code editor application employing web socket technology. This programme offers a feature that allows users to work together in real-time on a project.This application provides a feature where users can collaborate on a project in real-time. It's a full stack web application that provides a workspace to writing and sharing code to another developer by connecting in the same room. Every room will have a unique id which helps the developers to connect by sharing it between each other.

**Key words: Web Socket, Collaboration, Full Stack Application, Code Editor**

## 1 INTRODUCTION

### 1.1 Introduction

A multi user code editor is a software tool that allows multiple developers to work on the same codebase simultaneously. It enables real-time collaboration, where developers can view and edit code in real-time, share their screens, and communicate through chat or voice calls. Multi user code editors provide a centralized workspace for the development team, where everyone can access the same codebase and work together on it.

The use of multi user code editors has become increasingly popular as software development teams have become more distributed and remote. These tools help to bridge the gap between team members who may be in different locations, time zones, or even countries. With multi user code editors, developers can work together seamlessly, without

the need for lengthy email chains, file sharing, or frequent check-ins.

Some common features of multi user code editors include version control, syntax highlighting, code completion, debugging tools, and integration with other development tools. Examples of popular multi user code editors include Codeanywhere, Koding, Cloud9, CodeSandbox, and Collabedit.

Overall, multi user code editors have revolutionized the way software development teams collaborate, making it easier and more efficient for developers to work together on complex projects.

Any software solution or application relies on its programme, and today we create programmes or code utilising a variety of code editors. The code editor that is installed on a person's computer allows for code editing. But what if several individuals are working on the code at the same time, and modifications need to be made by all of them? Do we have to save each piece of code separately before exporting it to the next person to make changes? Using web socket technology, we are developing a real-time code editor tool that will allow several users to simultaneously interpret the same code while working on the project. This programme offers a feature that allows users to work together in real-time on a project.It's a web application that provides a workspace to write and share code with another developer by connecting in the same room. Every room will have a unique id which helps the developers to connect and work on the same project.

Multi user code editors have several advantages over traditional code editors. Here are some more benefits of using multi user code editors:

Real-time collaboration: Multi user code editors enable real-time collaboration, allowing multiple developers to work on the same codebase simultaneously. This improves productivity, reduces communication overhead, and eliminates the need for time-consuming handovers.

Enhanced communication: Multi user code editors offer built-in chat and voice calling features, making it easier for team members to communicate and collaborate.

### 1.2 Scope of Project

This Multi User Code Editor can be used in MNC's for efficient handling of projects where multiple persons can work on the same code simultaneously and share the knowledge

This can also be used by trainees, teachers who train students to write code usually in institutions and laboratories.

It can be used efficiently by a group of developers (students or professionals) for successful completion of the project.

## 2 LITERATURE SURVEY

Most of the existing collaborative code editors have followed client-server architecture, where the server holds a persisted copy of the shared document, and each client has their own local copy

**Saros - Distributed Party Programming:**

Saros is a plugin for eclipse that supports two or more programmers to collaborate and develop projects in real-time by sharing each other's changes. It is built using Java as a programming language and XMPP as a communication layer.

It follows a client-server architectural model for sending message updates to other clients. For performance reasons, it uses a peer-peer model for file transfer. In case of failure in direct connection, file transfer uses the client-server model.

**Google Docs:**

It is a widely known tool, where multiple users can edit documents. Google Docs also offers additional features like commenting on the text, lighting, changing document style, etc. Our project is inspired by the basic google doc functionality of real-time synchronous text editing. Google Docs uses operational transformation technology to enable concurrent editing. It is an optimistic consistency control algorithm and doesn't have a locking system while a user edits the doc. Apache wave also uses OT for its editor. We are also using the operational transformation in our project for concurrency control.

It also follows a client-server architectural model for sending message updates to other clients. Regarding performance reasons, it uses a peer-peer model for file transfer. File transfer uses the client-server model in case of failure in direct connection.

**Real-Time Collaborative Coding with Teletype for Atom** by M. McDonough and I. Geary. This paper presents Teletype for Atom, a real-time collaborative coding tool that enables multiple users to edit code simultaneously within the Atom text editor. The authors describe the architecture and implementation of the tool, and present a user study that evaluates its usability and effectiveness.

**CodeBubbles**: A Multi-User Code Editor for Collaborative Software Development by M. K. Stein, R. C. Miller, and M. C. Harrison. This paper presents CodeBubbles, a multi-user code editor designed to support collaborative software development. The authors describe the features of the tool that make it suitable for this application, and present a user study that evaluates its effectiveness.

## 3 OVERVIEW OF THE SYSTEM

### 3.1 Proposed System

Real Time Code Sync Editor is not only an editor where normally all developers write their code for their project, but it also provides a real time code sharing with other developers who were working on the same project. This editor helps the developers in sharing the real time code among themselves by joining a particular room by using a unique room id. Suppose one developer had created a room for sharing the code with other developers then he had to send them a room id so that the other developers join the room. In this editor as many as developers can join at the same time and can edit, read and delete the code. This editor supports some basic languages like Python, JavaScript, C, etc. which makes the code highlighted according to the keywords present in the programming languages. It also notifies which developer had join the room or left the room. And for security purpose all the code will be removed/ deleted from the database, so that the admin can't see the codes of other developers and also it free the space in the memory which helps in using low storage.

The functional components required for Real Time Code Sync Editor website are:

- Generating a room Id
- Providing a unique username
- Copying the room Id for sharing or inviting the other developers
- Analyzing the text in the text area field and highlighting the code based on the programming languages.

- Notifying the host who have join the room and who have left the room.

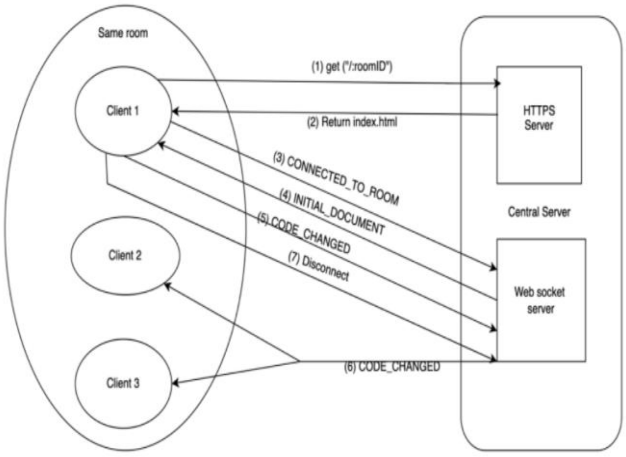


**Fig. 1** Client Server Architecture of multiple users who are connected to the same room.

Collaborative editing refers to a concept where a group of people scattered geographically can simultaneously work on the same set of files. Users used to send an entire copy of a file via email or other internal communication channels when they wanted to share a file in the past. When a different user wants to make changes to the file, that user downloads the copy, makes the changes locally, and then multicasts the updated copy to the interested group via a communication channel.

The user must manually identify the changes made by the prior user, handle any collisions, and send back the edited copy if another user is working on the old copy and has just received a new copy throughout this process. This procedure takes a long time and is tedious. This issue can be resolved by using collaborative code editing tools, which enable several people to access the same file and make adjustments, allowing other users to see the changes right away. These days, they are relatively well-known due to the clear advantages of pair programming.

# 4 RESULT SCREENS

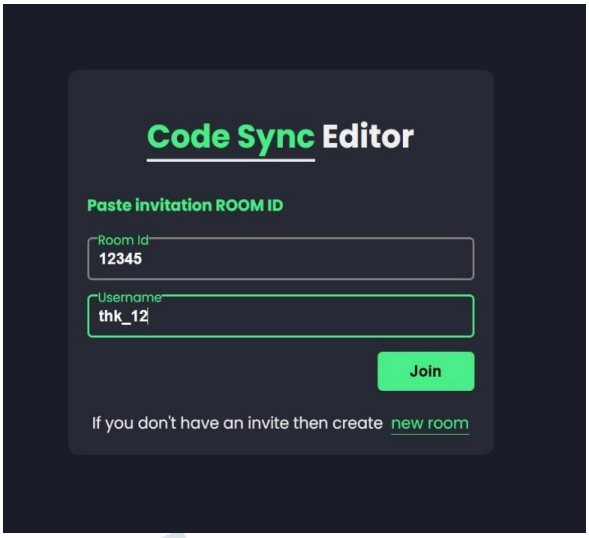


**Fig. 2** Home Screen

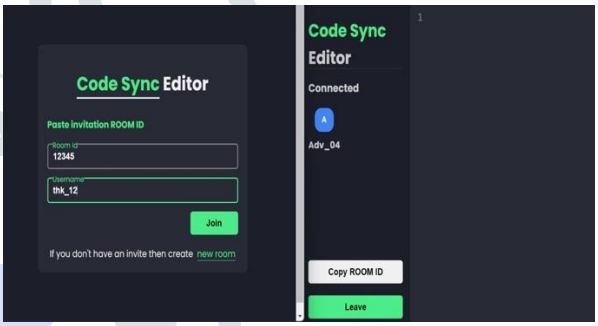


**Fig. 3** One user created the room and the user is joining the same room with the id

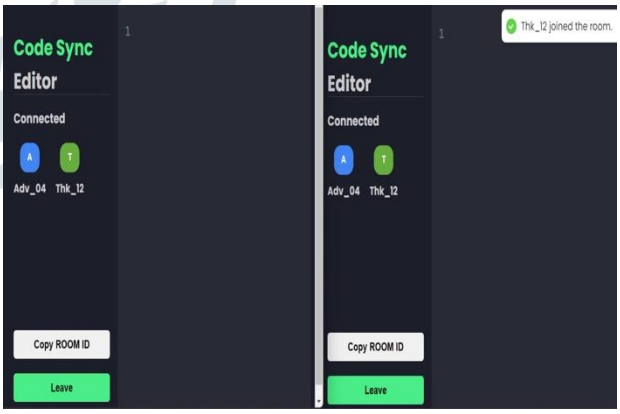


**Fig. 4** Notification pops up when a user joins a room


**Fig. 5** Two users collaborate and write code at same time

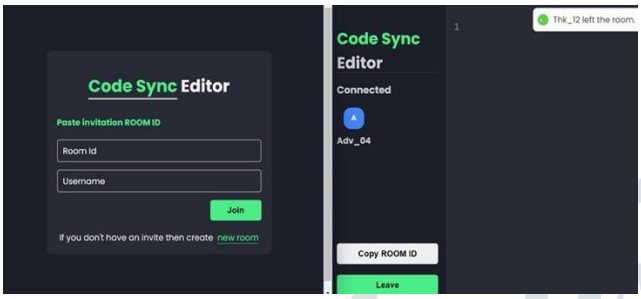


**Fig. 6** User gets notified when any other user leaves the room

## 5 CONCLUSION

In this project, our goal was to develop a collaborative editor tool that offers a seamless code editing experience to the users. In the process of building the tool, we researched various types of architectures and algorithms which gave a broad perspective of existing systems, their pros, and cons, the design constraints, and tradeoffs.

Our analysis of many designs presented in research articles helped us to figure out the architecture best suited for our specific application. The most challenging part was developing algorithms for consistency across multiple users. We achieved it through implementing CDRT. We could achieve faster updates across the users through the successful implementation of CDRT. Though CCE has basic text editing features, they are implemented robustly. The future work for the project would include transitioning from stateful to stateless and implementing a global cache and using a Redis pub sub-system to notify sockets on other servers on code-change events.

Branch creation functionality to create a branch atany point of time which would create a new child session with the current document's snapshot.User authentication and Chat/Call features could also be added to provide a better user experience and security.

## 6 REFERENCEES

[1] M. Doernhoefer, "Surfing the Net for Software Engineering Notes", ACM SIGSOFT Software Engineering Notes, Vol. 24,
No. 3, (1999), pp. 15–24.

[2] L. C. L. Kats, R. G. Vogelij, K. T. Kalleberg, and E. Visser, "Software development environments on the web", in
Proceedings of the ACM international symposium on New ideas, new paradigms, and reflections on programming and
software - Onward! '12, (2012), pp. 99
.

[3] M. Goldman, "Role-based interfaces for collaborative software development", in Proceedings of the 24th Annual ACM
Symposium Adjunct on User Interface Software and Technology - UIST '11 Adjunct, (2011), pp. 23.

[4] F. Frößler, "A Practice Theoretical Analysis of Real Time Collaboration Technology: Skype and Sametime in Software
Development Projects", Göttingen: Cuvillier, (2008).

[5] S. Klein, N. Vehring, and M. Kramer, "Introducing Real Time Communication: Frames, Modes & Rules", in Proceedings
23nd Bled eConference eTrust: Implications for the Individual, (2010), pp. 591–606.

[6] K. Riemer and F. Frößler, "Introducing Real-Time Collaboration Systems: Development of a Conceptual Scheme and
Research Directions", Communications of the Association for Informations Systems (CAIS), Vol. 20, (2007), pp. 204–225.

[7] M. Goldman, G. Little, and R. C. Miller, "Real-time Collaborative Coding in a Web IDE", in Proceedings of the 24th
Annual ACM Symposium on User Interface Software and Technology, (2011), pp. 155–164.

[8] H. Fan, C. Sun, and H. Shen, "ATCoPE: Any-time Collaborative Programming Environment for Seamless Integration of
Real-time and Non-real-time Teamwork in Software Development", in Proceedings of the 17th ACM International
Conference on Supporting Group Work, (2012), pp. 107–116.

[9] H. Bani-Salameh, C. Jeffery, Z. Al-Sharif, and I. Abu Doush, "Integrating Collaborative Program Development and

Debugging within a Virtual Environment", in Proceedings of the 14th Collaboration Researchers' International Workshop
on Groupware, Vol. 5411, (2008), pp. 107–120.

[10] A. Sarma, "A Survey of Collaborative Tools in Software Development, Technical Report, UCI-ISR-05-3", Irvine,
California, (2005).

[11] S. Goel and V. Kathuria, "A Novel Approach for Collaborative Pair Programming", Journal of Information Technology
Education, Vol. 9, (2010), pp. 183–196.

[12] H. B. Salameh and C. Jeffery, "Collaborative and social development environments: a literature review", International
Journal Computer Applications in Technology, Vol. 49, No. 2, (2014), pp. 89.

[13] S. Kumawat, M. T. Scholar, and A. Khunteta, "A Survey on Operational Transformation Algorithms: Challenges, Issues
and Achievements", International Journal of Engineering Science and Technology, Vol. 2, No. 7, (2010), pp. 3311–3319.

[14] D. Sun, S. Xia, C. Sun, and D. Chen, "Operational Transformation for Collaborative Word Processing", in Proceedings of
the 2004 ACM Conference on Computer Supported Cooperative Work, (2004), pp. 437–446.

[15] H. S. Molli, P. Molli, and G. Oster, "Semantic Consistency for Collaborative Systems", in Proceedings of the International
Workshop on Collaborative Editing Systems - CEW 2003, (2003).

[16] J. Sung-Jae, B. Yu-Mi, and S. Wooyoung, "Web Performance Analysis of Open Source Server Virtualization Techniques",
International Journal of Multimedia and Ubiquitous Engineering, Vol. 6, No. 4, (2011), pp. 45–52