TOPIC: SRS DOCUMENT

REALTIME DOCUMENT EDITOR.

Name	Deepika Negi
Registration Number	12216699
Roll Number	RK22QDA30
Submitted To	Dr. Karthick Panneerselvam

TABLE OF CONTENT

CLIENT APPROVAL FORM.

PROJECT NAME	REALTIME TEXT EDITOR FOR DEPARTMENT OF YOUTH CAPITAL.		
JOB LOCATION	Lovely Professional University, Phagwara, Punjab		
EST. START DATE	01/03/2024	EST. FINISH DATE	01/06/2024
PROJECT LEADER	Deepika Negi	COMPANY	DYC, LPU
CONTACT NAME	Rituraj Goswami	ADDRESS	Lovely Professional University, Phagwara, Punjab
PHONE	8130241880		
EMAIL	rituraj.goswami@outlook.com		

SUMMARY	A realtime document editor that will be used to edit the document, MoU and Applications of members by multiple users at the same time.
DESIRED OUTCOME	A realtime document editor web application.
ACTION TO COMPLETION	
BENEFITS OF PROJECT	Cash amount of INR 3500, and free entry to multiple concerts and events of the campus hosted by DYC, LPU.
PROJECTED SCHEDULE	To be completed before 01/06/2024
PROJECTED BUDGET	INR 3500
PROJECTED TEAM AND RESOURCE REQUIREMENTS	Developer : Deepika Negi
PROPOSAL MAY BE WITHDRAWN IF NOT ACCEPTED BY DATE OF	04/04/2024

ACCEPTANCE OF PROPOSAL			
AUTHORIZED CLIENT SIGNATURE		DATE OF ACCEPTANCE	05/04/2024

INTRODUCTION

PROJECT GitHub URL : https://github.com/Negi-Deepika/live-document-editor

This Software Requirements Specification (SRS) document provides a comprehensive overview of the Real-Time Document Editor project. The goal of the project is to develop a web application that will facilitate joint editing of documents in real-time. Users will be able to create new rooms or join existing ones and collaborate seamlessly with others.

Project Overview:

The real-time document editor is built using Node.js and Express.js for the backend, React for the frontend, and Socket.io for real-time communication. The application allows users to create documents, share them with others by generating unique room IDs, and collaborate on editing these documents in real time.

Objectives:

1. **Real-time Collaboration:** Enable users to collaborate on document editing in real time, allowing multiple users to edit the same document simultaneously.

2. **User-friendly Interface:** Develop an intuitive and responsive user interface that facilitates easy navigation and interaction with the application.

3. **Secure Communication:** Implement secure communication protocols to ensure the confidentiality and integrity of user data transmitted over the network.

4. **Scalability:** Design the application architecture to be scalable, accommodating a growing number of users and documents without compromising performance.

5. **Reliability:** Ensure the application's reliability by minimizing downtime, handling errors gracefully, and providing robust error-handling mechanisms.

6. **Cross-platform Compatibility:** Ensure compatibility across different devices and browsers, allowing users to access and use the application seamlessly from any platform.

By outlining these objectives, this document sets the foundation for the development process, guiding the implementation, testing, and deployment phases to achieve the desired outcome of a robust and user-friendly real-time document editor.

GENERAL DESCRIPTION

The real-time document editor application is designed to facilitate collaborative document editing among multiple users in real time. It serves as a platform where users can create, share, and edit documents together, enabling seamless collaboration and communication within teams or groups. The primary purpose of the application is to enhance productivity and efficiency by eliminating the need for asynchronous editing and communication methods such as emailing documents back and forth or using separate editing tools.

Functionality:

1. Document Creation and Sharing:

- Users can create new documents within the application.
- Each document is assigned a unique Room ID, which users can share with collaborators to join the document editing session.

2. Real-time Collaboration:

- Multiple users can join the same document editing session using the shared Room ID.
- Changes made by any user to the document are instantly reflected in real-time for all other users in the session.
- Users can see the edits made by other collaborators as they occur, facilitating seamless collaboration and communication.

3. User Management:

- Users can input their usernames when joining a document editing session, allowing others to identify their contributions.
- The application maintains a list of connected users within each document editing session, enabling users to see who else is currently collaborating on the document.

4. Document Editing Features:

- The document editor provides basic text editing features such as typing, deleting, and formatting text.
- Users can collaborate on various types of documents, including plain text documents, code snippets, or rich text documents with formatting options.

5. Additional Features:

- Copying Room ID: Users can easily copy the Room ID to the clipboard for sharing with collaborators.
- Leaving the Document: Users can leave the document editing session when they are done collaborating or no longer wish to participate.

Overall, the real-time document editor application aims to streamline collaborative document editing processes, enhance teamwork, and improve productivity by providing a seamless and intuitive platform for users to collaborate on documents in real time.

SPECIFIC REQUIREMENTS

HOME PAGE :

Description:

The Home Page serves as the entry point for users to access the real-time document editor application. It provides functionalities for users to create new rooms or join existing ones by entering a Room ID and a username.

Functionality:

1. **Input Room ID and Username:**

- Users can input a Room ID and a username in designated input fields.
- These inputs are essential for creating a new room or joining an existing one.

2. **Create New Room:**

- Users have the option to create a new room by clicking on the "Create New Room" button.
- Upon clicking, a unique Room ID is generated using UUID, and the user is redirected to the Editor Page of the newly created room.

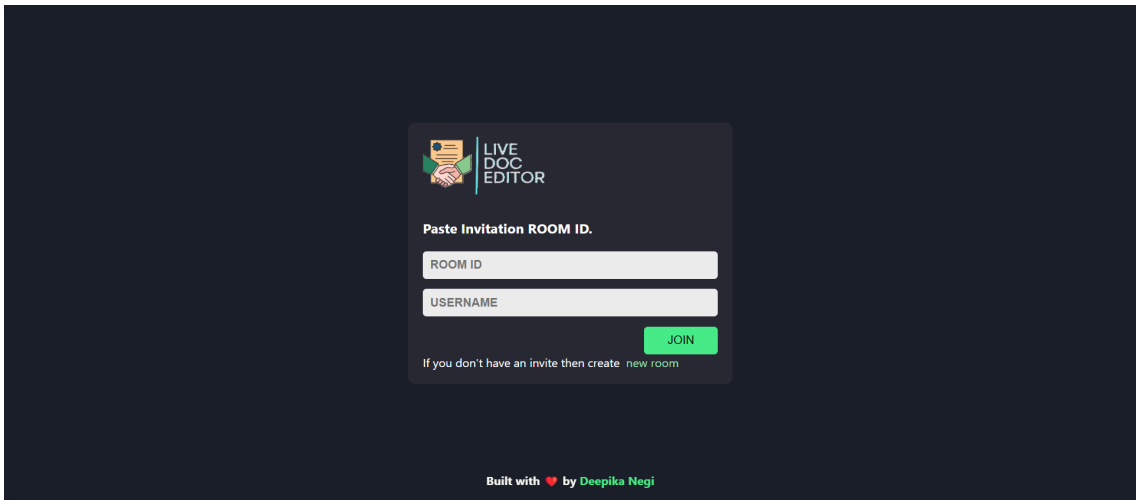
3. **Join Existing Room:**

- Users can join an existing room by entering a valid Room ID and their desired username.
- Upon joining, users are redirected to the Editor Page of the selected room.

4. **Dependencies:**

- ***React***: Used for building the user interface of the Home Page.
- ***React Router DOM***: Enables navigation within the application by providing routing functionalities.
- ***React Hot Toast***: Provides toast notifications for displaying alerts and messages to users.
- ***UUID***: Generates unique identifiers for creating new room IDs.

5. **Screenshot**:



EDITOR PAGE :

Description:
The Editor Page is the core component of the real-time document editor application, where users collaborate on editing documents within a specific room. It provides functionalities for real-time collaborative editing, user management, and interaction with the document editing session.

Functionality:

1. **User Connectivity:**
   Users can view a list of connected users within the room, allowing them to see who else is currently collaborating on the document.

2. **Copy Room ID:**
   Users have the option to copy the Room ID to the clipboard with a single click. This feature simplifies the process of sharing the Room ID with collaborators.

3. **Leave Room :**
   Users can leave the current document editing session by clicking on the "Leave" button. This action disconnects them from the room and redirects them to the Home Page.

4. **Real-time Editing:**

   The Editor Page enables real-time collaborative editing of documents, allowing multiple users to edit the same document simultaneously. Changes made by any user are instantly synchronized and reflected in real-time for all other connected users.

Dependencies:

React: Used for building the user interface and managing the state of the Editor Page.
Socket.io: Facilitates real-time communication between clients and the server, enabling synchronized editing across connected users.
Codemirror: Provides a versatile text editor component with syntax highlighting and other editing functionalities.

Screenshot:

ANALYSIS OF MODEL

The software model of the real-time document editor encompasses various components and interactions between them to facilitate collaborative document editing. This analysis includes an overview of the architecture, data flow, and technical details of the application.

Architecture:

The application follows a client-server architecture, where the frontend client (built using React) communicates with the backend server (built using Node.js and Express.js) via Socket.io for real-time communication. The server manages the creation and management of rooms, handles client connections, and facilitates real-time synchronization of document edits among connected clients.

Data Flow:

1. **User Interaction:**

Users interact with the application through the frontend interface, where they can create new rooms, join existing ones, and collaborate on document editing.

User inputs, such as Room IDs and usernames, are collected via input fields and sent to the server for processing.

2. **Server Processing:**

Upon receiving user inputs, the server generates a unique Room ID for new rooms and manages client connections.
The server stores information about connected clients within each room, including their usernames and socket IDs.

3. **Real-time Communication:**

Socket.io facilitates real-time communication between the server and clients, enabling synchronized editing of documents.
Clients emit events to the server to join rooms, send document edits, and receive updates about other connected users' actions.

4. **Document Editing:**

Clients use Codemirror, a versatile text editor component, to edit documents in real-time.
Changes made by one client are broadcasted to all other connected clients via the server, ensuring synchronized editing across all users.

5. **Technical Details:**

Frontend:
Built using React for its component-based architecture and efficient state management.
Utilizes React Router DOM for client-side routing to navigate between different pages.
Implements React Hot Toast for displaying toast notifications to users.

Backend:

Developed using Node.js and Express.js for handling HTTP requests and managing server-side logic.

Utilizes Socket.io for enabling real-time bidirectional communication between clients and the server.

Data Storage:

The application currently does not incorporate a database for storing document data. Instead, documents are stored in memory on the server and synchronized in real-time among connected clients. Future iterations of the application may integrate a database for persistent storage of documents and user data.

Conclusion:

The software model of the real-time document editor demonstrates a robust architecture for facilitating collaborative document editing. By leveraging technologies such as React, Socket.io, and Codemirror, the application provides users with a seamless and intuitive platform for real-time collaboration on documents. Further enhancements to the data storage and scalability aspects can contribute to the application's overall robustness and effectiveness in facilitating collaborative work environments.

PROOF OF CLIENT APPROVAL.

TERM SHEET

This agreement has been executed on **26th February of the year 2024** between **Deepika Negi (Registration Number : 12216699)** hereinafter referred to as "First Party" (Which expression shall unless it be repugnant to the context or meaning thereof be deemed to mean and include its successors and assigns) of the One Part.

And

Department of Youth Capital under the aegis of Division of Youth Affairs, Student Welfare Wing, Lovely Professional University: hereinafter referred to as "Second Party" (Which expression shall unless it be repugnant to the context or meaning thereof be deemed to mean and include its successors and assigns) of the other part.

The Project : DYC TEAMS DOCUMENT EDITOR, is hereby given to Deepika Negi by Department of Youth Capital, LPU. Under the terms :

1. The deployment and Data storage of the Text editor will be only stored under the Department of Youth Capital.
2. First Party can not use the project for further commercialization, but can be displayed as Portfolio projects.

All disputes, subject to the jurisdiction of Lovely Professional University, Phagwara, Punjab.

Signature of Student Head,

Rituraj Goswami,

Department Of Youth Capital,

Student Welfare Wing, LPU

Signature of First Party,

Deepika Negi (12216699)

B.Tech CSE Second Year,

Lovely Professional University

Date of Signing the Term Sheet :

26th February 2024.

Place of Signing the Term Sheet:

Lovely Professional University, Phagwara, Punjab, India.