Mini Project - II

(2021-2022)

Kwiky!

PROJECT REPORT

Institute of Engineering &Technology

Submitted to:-

Mr. Mandeep Singh

(Assistant Professor)

Submitted by :-

Prashant(191500574)

Surya Raj(191500832)

Priya Tomar (191500603)

Pushpendra Tiwari(191500613)

Department Of Computer Science Engineering & Applications

DECLARATION

I/we hereby declare that the work which is being presented in the Bachelor of technology. Project **"Kwiky"**, in partial fulfilment of the requirements for the award of the ***Bachelor of Technology*** in Computer Science and Engineering and submitted to the Department of Computer Engineering and Applications of GLA University, Mathura, is an authentic record of my/our own work carried under the supervision of **Mr. Mandeep Singh, Assistant Professor, Dept. of CEA, GLA University.**

Name Of Candidate:
Prashant
(191500574)

Name Of Candidate:
Surya Raj
(191500832)

Name Of Candidate:
Priya Tomar
(191500603)

Name Of Candidate:
Pushpendra
(191500613)

Department of Computer Engineering and Applications
GLA University,17 km.Stone NH#2,Mathura-DelhiRoad,
Chaumuha, Mathura – 281406 U.P (India)

Certificate

This is to certify that the project entitled “Chat Application” carried out in Mini Project – II Lab, is a bonafide work by Prashant, Surya Raj, Priya Tomar, Pushpendra is submitted in partial fulfilment of the requirements for the award of the degree Bachelor of Technology (Computer Science & Engineering).

Signature of Supervisor:

Name of Supervisor: Mr. Mandeep Singh

Date:

ACKNOWLEDGMENT

It gives us a great sense of pleasure to present the Final Report of the B.Tech Mini Project **Kwiky** undertaken during B.Tech IIIrd Year. This project in itself is going to be an acknowledgement to the inspiration, drive and technical assistance will be contributed to it by many individuals. We owe Special debt of gratitude to **Mr. Mandeep Singh**, Assistant Professor Department of CEA , for providing us with an encouraging platform to develop this project , which thus helped us in shaping our abilities towards a constructive goal and for his constant support and guidance to our work. His sincerity, thoroughness and perseverance has been a constant source of inspiration for us. We believe that he will shower us with all his extensively experienced ideas and insightful comments at different stages of the project & also taught us about the latest industry-oriented technologies. We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind guidance and co-operation.

Prashant(191500574)

Surya Raj(191500832)

Priya Tomar (191500603)

Pushpendra(191500613)

ABSTRACT

The latest development of the Internet has brought the world into our hands. Everything happens through the internet from passing information to purchasing something. The Internet made the world a small circle. This project is also based on the internet. This paper shows the importance of chat applications in day today life and its impact in the technological world. This project is to develop a chat system based on Web Technologies such as Socket.io, NodeJS, Express server, HTML, CSS and Vanilla JavaScript. The application allows people to transfer messages one on one among people. This online system is developed to interact or chat with one another on the Internet. It is much more reliable and secure than other traditional systems available. One can be both anonymous or talk freely as one pleases. Chatting applications are very popular among Internet users and Smartphone's owners. Hundred million smartphone owners use chat applications on amonthly basis. These chat applications offer the communication free of charge and majority of them are free to install which makes it very appealing for the potential customers. These chat applications offer different services and built- in features to their users while in majority of the cases, they neglect security aspects of their usages and messages.

CONTENTS

CHAPTER-1

INTRODUCTION

1.1 ABSTRACT

This Web Application “Kwiky” has been submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering at **GLA University**, Mathura supervised by **Mr.Mandeep Singh.** This project has been completed within a month and has been executed in modules, meetings have been organised to check the progress of the work and forinstruction and guidelines.

1.2 MOTIVATION

In recent years, we have realized the importance of chat app and how important it is for us to have our resources online. In the century we are living in, the world is progressing at a really great pace, a lot of technologies come up every single day. To keep up with technology is also important to survive in this world of digitalization and learning. Along with this we need to have a place to keep the resources for areas of our interest so we thought of developing an application which could provide us with unique users. In today’s world, online chatting is booming with the rise of the internet. So, we came up with a chat application.

1.3 OBJECTIVE

Our goal is to build a Web Application for Chatting Purposes which holds a single chat room as a single server. This helps us to use limited resources and same way this can be scaled to build a bigger applications, one that can compete with the likes of Discord or Slack. Any user can connect to it from an open window/tab and is able to send their messages. Each connected user will see instantly any message sent, and new connected users will see only the new messages from the moment they join to facilitate privacy. Express(.js) is that it enables you to easily create web applications by providing a slightly simplerinterface for creating your request endpoints, handling cookies, etc. than vanilla Node(.js). React is an open-source, component-based JavaScript library used to create quick and interactive interfaces or UI components for users for web and mobile-based applications. It is a declarative and highly accurate library that provides reusable code, which enhances the efficiency and flexibility of single-pageapplications. Node.js is an open-source and cross-platform JavaScript runtime environment. Node.js with Express.js can also be used to create classic web applications on the server-side.

1.4 EXISTING SYSTEM

There are multiple systems which already exist based on the concept of Chat Applications but what we offer is anonymity or freedom in a different way, which is currently similar to Omegle Application or such.

Technologies would be similar too but it matters more on the side of how they are used in

application to application. So we have implemented a totally different form of Chatting where anyone can join with a simple browser and chat freely at their will.

The backend interface is implemented with two servers: client and Server. Server side is responsible for the database retrieval, database maintenance and their services to the client side, whereas the page maintenance and the application interface will be performed by the client side. Website is a set of pages provided through interconnected network that is

available for everyone to access around the globe through network facility. First, the communication between client end and server ends through Node.js which acts as an

intermediator by the usage of JSON. JSON (JavaScript Object Notation) is a lightweight data interchange format, readable and writable by humans, as well as easily translated and made (generated) by the computer. The services are provided by socket.io library, Express.js, React and Node.js. Dual path communication between the client side and server side is possible through this channel. Here, every configuration is implemented only through JavaScript and socket.io, because both server and client sides support the usage of common language. If the user starts using a web chat application and performs any steps in the page, this will initiate a request to JSON, JSON would then initiate the service, thenafter obtaining the required information from the server end, it will provide the results in the page.

1.5 SOURCES

The source of our project (including all the project work, documentations and presentations) will be available at the following link –

https://github.com/pttiwari11/kwiky

CHAPTER -2

SOFTWARE REQUIREMENT ANALYSIS

2.1 PROBLEM STATEMENT

This project is to create a chat application with a server and users to enable the users to chat with each other. To develop an instant messaging solution to enable users to seamlessly communicate with each other. The project should be very easy to use enabling even a novice person to use it. This project can play an important role in the organizational field where employees can connect through LAN, WAN or PAN. The main purpose of this project is to provide multi-chatting, free or anonymous chatting functionality through the network.

2.2 HARDWARE AND SOFTWARE REQUIREMENTS

Hardware Requirements

- Hard Disk: 100 MB
- RAM: 1 GB
- Processor: 1.6 GHz or faster processor

Software Requirements

- Operating System: Windows/Linux/MAC OS
- Application Software: Visual Studio Code (version 1.61.0)
- Web Browser: Google Chrome, Mozilla Firefox, Any Other

2.3 MODULES AND FUNCTIONALITIES

Functional Requirements

- Web Browser
- NodeJS, Socket.io, Express, any IDE to work with these

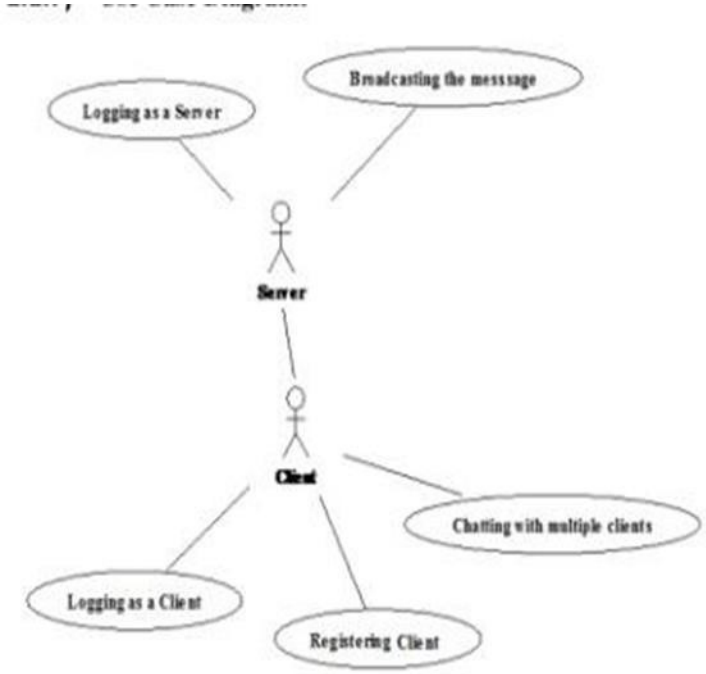
Non-Functional Requirements

- Privacy: Messages shared between users should be temporary to maintain privacy and no means to export chats or view history once chat-room closes.
- Robustness: If a user's System Crashes then he can join the same chat-room through joining same server again but not really be able to see his chat history again due to privacy concern.
- Performance: Application must be lightweight and must send messages instantly network.
- Send Message: User should be able to send instant message to any person available on his/her Server.
- Group Message: The chats work in a server-enabled-room kind of way so there can be multiple servers, each for a different group which get scaled on the go and one/two or more than two users can join one room at the same time**.**

CHAPTER – 3

Software Design

3.1 Use case Diagram:



CHAPTER-4

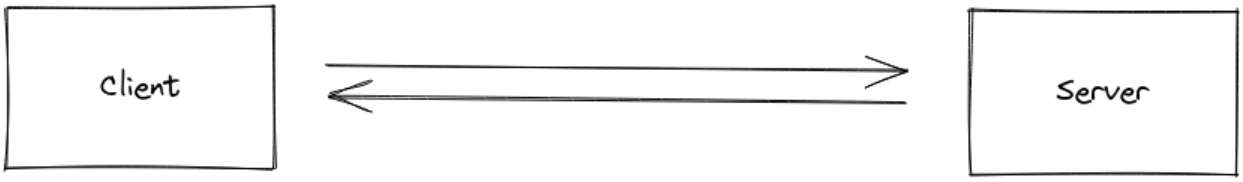
TECHNOLOGY USED

4.1 BASIC TERMINOLOGIES

NodeJS - Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser. It is designed to make the development process smoother and easier. It provides an end-to-end framework for the developers to work. This technology plays a big part in the development of web applications.

Socket.io

Socket.IO is a library that enables **low-latency**, **bidirectional** and **event-based** communication between a client and a server.



It is built on top of the WebSocket protocol and provides additional guarantees like fallback to HTTP long-polling or automatic reconnection.

INFO

WebSocket is a communication protocol which provides a full-duplex and low-latency channel between the server and the browser. More information can be found here.

There are several Socket.IO server implementations available:

- JavaScript (whose documentation can be found here on this website)
  - Installation steps
  - API
  - Source code
- Java: https://github.com/mrniko/netty-socketio
- Java: https://github.com/trinopoty/socket.io-server-java
- Python: https://github.com/miguelgrinberg/python-socketio
- Golang: https://github.com/googollee/go-socket.io

And client implementations in most major languages:

- JavaScript (which can be run either in the browser, in Node.js or in React Native)
  - Installation steps
  - API
  - Source code
- Java: https://github.com/socketio/socket.io-client-java
- C++: https://github.com/socketio/socket.io-client-cpp
- Swift: https://github.com/socketio/socket.io-client-swift
- Dart: https://github.com/rikulo/socket.io-client-dart
- Python: https://github.com/miguelgrinberg/python-socketio
- .Net: https://github.com/doghappy/socket.io-client-csharp
- Rust: https://github.com/1c3t3a/rust-socketio
- Kotlin: https://github.com/icerockdev/moko-socket-io

Express: Back-End Framework

Express is a Node.js framework. Rather than writing the code using Node.js and creating loads of Node modules, Express makes it simpler and easier to write the back-end code. Express helps in designing great web applications and APIs. Express supports many middle wares which makes the code shorter and easier to write.

- Asynchronous and Single-threaded.
- Efficient, fast & scalable
- Has the biggest community for Node.js

- Express promotes code reusability with its built-in router.
- Robust API
- Create a new folder to start your express project and type below command in the command prompt to initialize a package.json file. Accept the default settings and continue.

Vanilla JS

Vanilla JS is a fast, lightweight, cross-platform framework. for building incredible, powerful JavaScript applications.

Vanilla JavaScript refers to using plain Javascript without any additional libraries or frameworks. The term became popular when Eric Wastl created the Vanilla JS site in 2012 as a joke. The site tries to bring attention to the fact that you can use just plain Javascript in many cases.

Learning "vanilla" JavaScript means learning the fundamentals of JavaScript. Frameworks and libraries help abstract and simplify some features of JavaScript which makes writing code faster and easier. But in the long run, it is important to know the fundamentals of JS to be able to learn the use of new frameworks faster and decide if a framework is needed at all.

Node.js: JS Runtime Environment

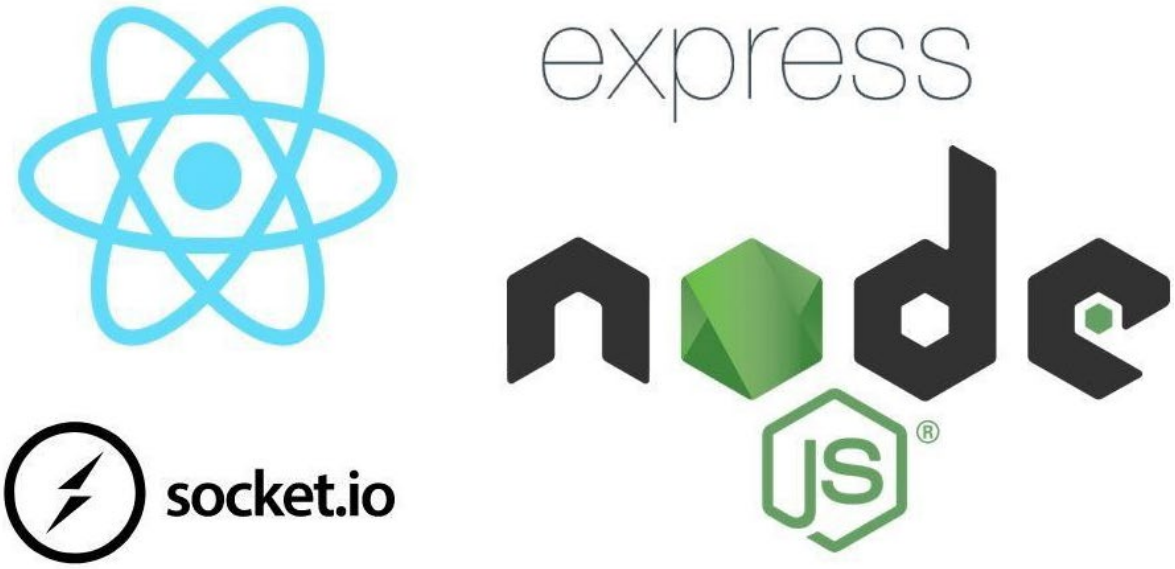
Node.js provides a JavaScript Environment which allows the user to run their code on the server. Node pack manager i.e., npm allows the user to choose from thousands of free packages to download.

- Open-source JavaScript Runtime Environment
- Single threading – Follows a single-threaded model.
- Data Streaming
- Fast – Built on Google Chrome's JavaScript Engine, Node.js has a fast code execution.
- Highly Scalable

Chapter-5

IMPLEMENTATION AND USER INTERFACE

Our goal is to build a web chat that holds a single chat room. Any user can connect to it from an open window/tab, is able to send multiple messages during the chat, each connected user will see instantly any message sent from multiple users.



5.1 Implementation of the Chat Application

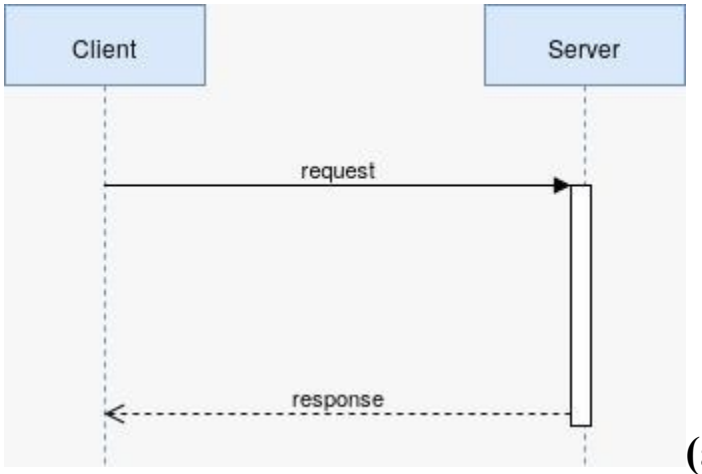
The web app will consist of 2 parts, the client and the server.

The client will contain 2 main components: a login form, where the user can write a use name to use for the chat and the option to upload an image to use as an avatar throughout the conversation. The second component will consist of the chat messages, where each message that is sent, is viewed from anyone that's connected to the chat room.

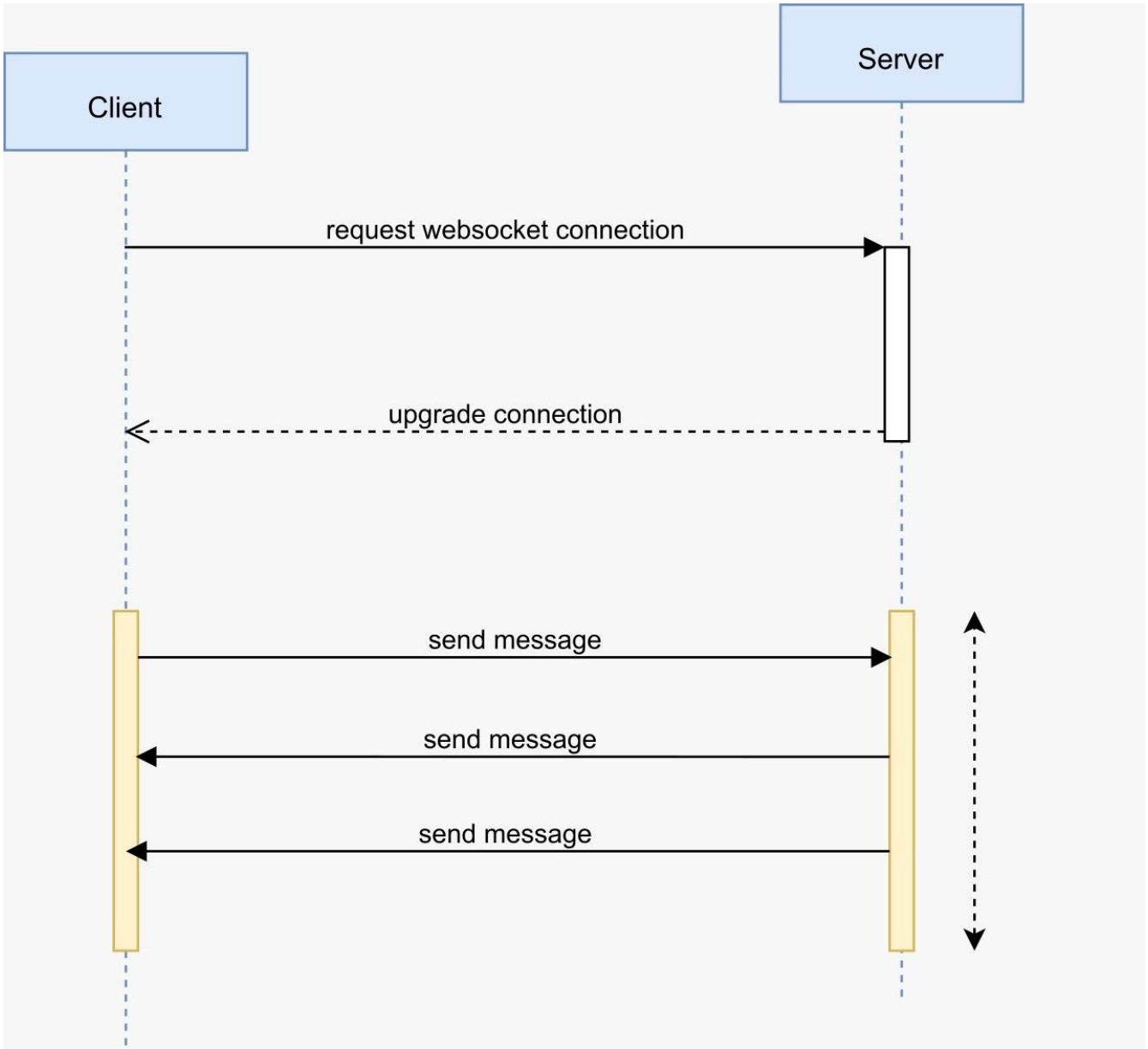
The server will accept two types of connections: an HTTP request from both client side or user side. The second connection theserver manages is a WebSocket port to manage incoming connections and broadcast messages to all connected clients, in real time.

Socket.IO and WebSocket — what's the difference?

Most websites you visit use HTTP to make API calls, which means the client sends a request to the server, and the server sends back a response. This kind of communication can only be initiated by the client. It suits most needs for a website, e.g., to get data from a server once. What if there are other needs? What if the client needs to check often for data? There is a solution for that, called AJAX long-polling, but it's not efficient. Long-polling involves sending periodic HTTP requests for data, introducing latency and increasing server load. That's where WebSocket comes in. It's a communications protocol that offers full-duplex communication channels over a single TCP connection. In other words, a WebSocket connection remains open as long as both the client and the server choose not to close it. While the connection is open, messages can be exchanged both ways. This allows to be used for applications like push messages to clients or real-time chat between clients. Socket.IO is an event-driven JavaScript library that offers an abstraction layer for using bi-directional communication between web clients and servers, and provides an easy configuration of WebSocket implementation.

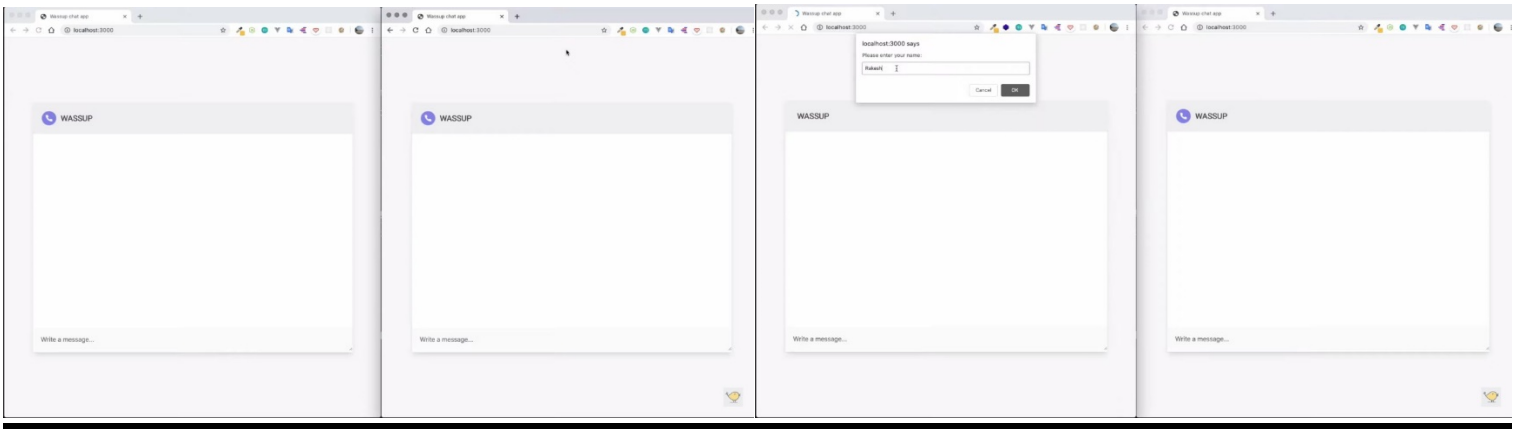


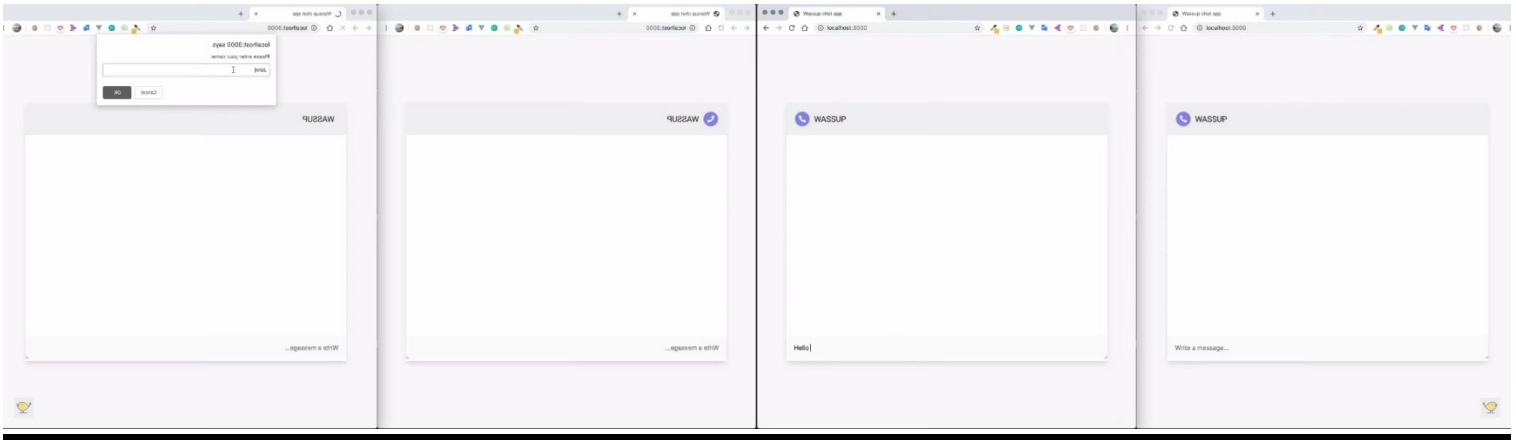
(a) Sequential Diagram

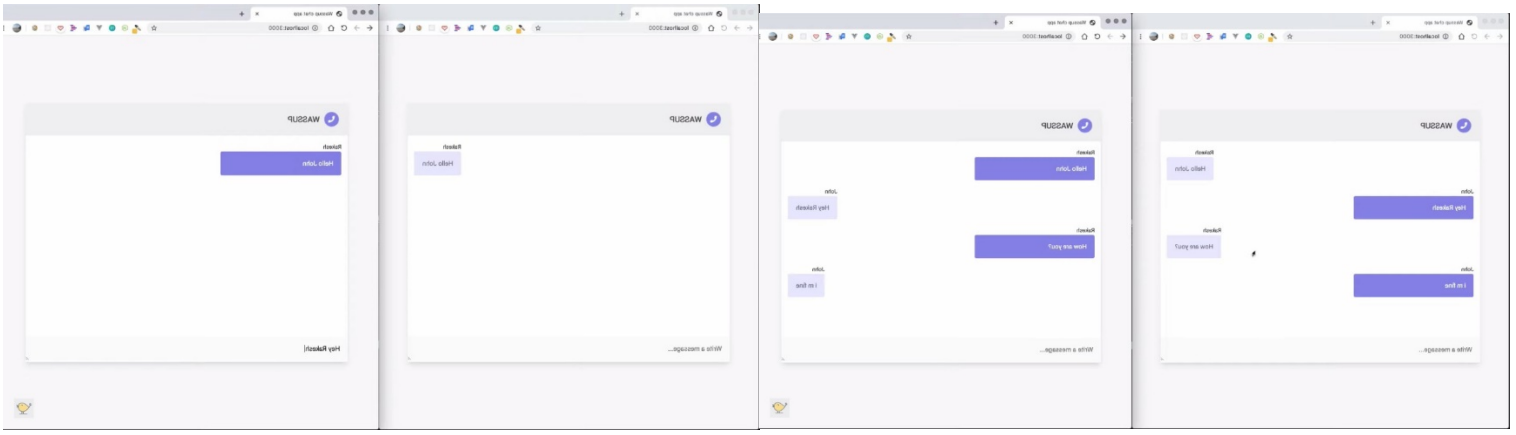


(b) Client-Server Interaction

5.2 USER INTERFACE







Conclusion

As a conclusion, We can say that this undergoing project is giving great experience. Thanks to this project, We are acquiring deep knowledge concerning my technical skills but we also benefited personally. Currently Web Technologies are common technologies of web applications and chat applications, and one of the most popular technologies for development used by developers worldwide. If we surf internet we can see millions of websites, applications and games built with JavaScript. We are learning to live in a different environment from the one we are used to. Indeed, we are growing more independent in work and also in everyday life, realizingthat we could do more things than we thought like learning new things by myself. There are huge opportunities available for the students who want to work in this field. Many private and public organizations hire web designer and app designer for theironline work and development. With the rapid advent of online industry, the demand of web development and app development professionals is increasing and this has created a huge job opportunity for the aspirants in the upcoming days. Also, an experienced person in this field can also work as a freelancer; there are many online companies which provide online projects to the individuals.

We have completed our project within time limit with the coordination of our team members under the supervision of our mentor **Mr.Mandeep Singh.**

References/Bibliography

- ExpressJS: https://expressjs.com/en/guide/routing.html
- Npm: https://www.npmjs.com/
- NodeJS: https://nodejs.org/en/docs/
- Socket.io: https://socket.io/

Faculty Guidelines

Mr. Mandeep Singh

(Assistant Professor)

THANK YOU