

**Project Report**  
**On**  
**“REAL TIME CHAT APP”**  
**Submitted in Fulfillment for the award of**  
**BACHELOR OF TECHNOLOGY**  
**In**  
**Computer Science & Engineering**



**SUBMITTED TO**

**Ms. SHILPA**

**(ASSISTANT PROFESSOR)**

**SUBMITTED BY**

**Tanishq Sharma & Aman Raj**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**  
**SCHOOL OF ENGINEERING & TECHNOLOGY**

**GANGA TECHNICAL CAMPUS**

**SOLDHA, BAHADURGARH**

**Year 2018-2022**

## **CANDIDATE'S DECLARATION**

WE, **TANISHQ SHARMA AND AMAN RAJ**, bearing roll no 18BTCSE034 & 19BTCSE003, students of B.Tech of Computer Science and Engineering department hereby declare that WE own the full responsibility for the information, results etc. provided in this PROJECT titled “REAL TIME CHAT APP” submitted to M.D. University, Rohtak for the award of B.Tech (CSE) degree. WE have taken care in all respect to honour the intellectual property right and have acknowledged the contribution of others for using them in academic purpose and further declare that in case of any violation of intellectual property right or copyright WE, as a candidate, will be fully responsible for the same. OUR supervisor should not be held responsible for full or partial violation of copyright or intellectual property right.

NAME:- TANISHQ SHARMA, AMAN RAJ

ROLL No:- 18BTCSE034, 18BTCSE003

DATE:-

## **CERTIFICATE**

Certified that the PROJECT entitled “**REAL TIME CHAT APP**” submitted by **TANISHQ SHARMA AND AMANRAJ** bearing roll no 18BTCSE034 & 18BTCSE003, in fulfilment of the requirements for the award of the degree of Bachelor of technology (Computer Science and Engineering) of M.D.U. Rohtak, is a record of the student’s own work carried out under my supervision and guidance. To the best of our knowledge, this project has not been submitted to M.D.U. or any other university or institute for award of a degree. It is further understood that by his/her certificate the undersigned does not endorse or approve of any statement made, opinion expressed or conclusion drawn therein but approve the dissertation only for the purpose for which it is submitted

**PROJECT GUIDE:- MS SHILPA**

**SIGNATURE BY PROJECT GUIDE:-**

**HEAD OF DEPARTMENT:- MS VANDANA**

**SIGNATURE BY HEAD OF DEPARTMENT:-**

## **ACKNOWLEDGEMENT**

First and foremost, WE would like to thank our Teacher **Ms SHILPA** who guided us in doing these projects. She provided us with invaluable advice and helped us in difficult periods. Her motivation and help contributed tremendously to the successful completion of the project. Besides, we would like to thank all the teachers who helped us by giving us advice and providing the equipment which we needed. Also we would like to thank my family and friends for their support. Without that support we couldn't have succeeded in completing this project. At last but not in least, we would like to thank everyone who helped and motivated us to work on this project.

## ABSTRACT

Real-time chat is virtually any online communication that provides a real-time or live transmission of text messages from sender to receiver. A variety of software programs are available to enable real-time chat between individuals using Internet services.

Real-time chat can be any direct text-based or video-based (using webcams) one-to-one chat or one-to-many group chats.

This project is a web based Real time chat application system for browser.

The project objective is to allow real-time communication between clients and servers.

Real-time chat application is a process where clients directly communicate with each other from a server in real-time, without an intermediary service, over the internet. This project is an attempt to provide the advantages of online chatting application to clients. It helps communication with each other any where through internet by using **web browser**.

### INTRIDUCTION

Real-time chat is virtually any online communication that provides a real-time or live transmission of text messages from sender to receiver. A variety of software programs are available to enable real-time chat between individuals using Internet services.

Real-time chat can be any direct text-based or video-based (using webcams) one-to-one chat or one-to-many group chats.

This project is a web based Real time chat application system for browser.

The project objective is to allow real-time communication between clients and servers.

Real-time chat application is a process where clients directly communicate with each other from a server in real-time, without an intermediary service, over the internet. This project is an attempt to provide the advantages of online chatting application to clients. It helps communication with each other any where through internet by using **web browser**.

#### 1.1 PROJECT OBJECTIVES:

The objective of the project is to make a **web-based** application to communicate with clients. In order to build such an application, complete web support need to be provided. A complete and efficient web application which can provide the real-time chat experience is the basic objective of the project. The web application can be implemented in the form of a web-based application and it can also be implemented in the form of android application.

#### 1.2 PROJECT OVERVIEW:

The central concept of the application is to allow the clients to connect with each other virtually using the Internet and allow the clients to send messages and receive messages from each other. The messages pertaining to the clients has been stored directly on the webpage.

The Local server process the messages and other things of the clients. The application is currently designed into a single module in which the clients can

directly communicate with each other and update the information pertaining to their chatting and those of the clients through the software. The application is hosted on the web.

### **1.3 PROJECT SCOPE:**

This project can be mainly divided into two modules:

- 1.Server
- 2.Client

This project is mainly depended on client/server model. The client requests the server and server responses by granting the clients request. The proposed system should provide both of the above features along with the followed ones: **SERVER:** The server should be able to perform the following features: The first and foremost problem is to find the server. We should identify the program in the server which processes the client's request. Administrator Client who will be acting as a super user. Creating of private room with the password facility to enable private chats with the users online. The server is always waiting for clients requests .The clients come and go down but the server remains the same. **CLIENT:** The client should be able to perform the following features: Should be able to send message to anybody in the room with clients unique chat name created in the server for chatting purpose. Should be provided with the drawing tools like free hand, rectangle, oval , line and also sending text message over the room. In all the network applications, we find two sort program where the first i.e., server sends the information and the second i.e., client receives the information

Messaging apps are surging in popularity. The past few years have brought apps like WhatsApp, Telegram, Signal, and Line.

People seem to prefer chat-based applications because they allow for real-time interaction. They also add a personal touch to the experience.

I recently attended a workshop conducted by the Free Software Movement Karnataka in Bangalore where I mentored a group of college students.

During the interactions, I observed certain things:

1. Despite encouraging students to interact with the mentor, communication was always one-sided.
2. Students often felt too shy to ask questions during the sessions.
3. They were more comfortable asking questions and getting feedback in one-on-one conversations.

So we had to find a solution to break the ice between mentors and students. A local chat application came handy in this situation. People love to be anonymous, which gives them more power to express themselves and ask anytime anywhere. This is the same mantra used by most of the popular forums on the internet, such as Reddit and 4chan. These are just a few giant examples of semi-anonymous apps.

So I started thinking about this idea. I came up with some of the basic requirements and features.

1. Users register by giving a handle, which is unique to every user (a dummy name). Only the handle will be revealed to other users. So people are free to choose any handle and hence they stay anonymous.
2. A member can see other members who are online. They have an option to go public, which broadcast the message to all online members in the chat.
3. For private messages, the sender should first send a request to the other member. Other members upon accepting the request can have private chat with them.



### TECH-SATCK

We have used several technologies for building this project, which are:

1. HTML
2. CSS
3. JavaScript
4. Nodejs

And we also used some Nodejs packages in this project, which are:

1. Express.js
2. Nodemon
3. Socket.io

## 2.1 FRONT-END DEVELOPMENT

**Front End Development:** The part of a website that the user interacts with directly is termed the front end. It is also referred to as the 'client side' of the application.

It includes everything that users experience directly: text colors and styles, images, graphs and tables, buttons, colors, and navigation menu. HTML, CSS, and JavaScript are the languages used for Front End development. The structure, design, behavior, and content of everything seen on browser screens when websites, web applications, or mobile apps are opened up, is implemented by front End developers. Responsiveness and performance are two main objectives of the Front End. The developer must ensure that the site is responsive i.e. it appears correctly on devices of all sizes no part of the website should behave abnormally irrespective of the size of the screen.

Front end development manages everything that users visually see first in their browser or application. Front end developers are responsible for the look and feel of a site.

Front end development is mostly focused on what some may coin the "client side" of development. Front end developers will be engaged in analyzing code, design, and debugging applications along with ensuring a seamless user experience. You

manage what people first see in their browser. As a front end developer you are responsible for the look, feel and ultimately design of the site.

### 2.1.1 HTML

HTML is an acronym which stands for **Hyper Text Markup Language** which is used for creating web pages and web applications. Let's see what is meant by Hypertext Markup Language, and Web page.

**Hyper Text:** HyperText simply means "Text within Text." A text has a link within it, is a hypertext. Whenever you click on a link which brings you to a new webpage, you have clicked on a hypertext. HyperText is a way to link two or more web pages (HTML documents) with each other.

**Markup language:** A markup language is a computer language that is used to apply layout and formatting conventions to a text document. Markup language makes text more interactive and dynamic. It can turn text into images, tables, links, etc.

**Web Page:** A web page is a document which is commonly written in HTML and translated by a web browser. A web page can be identified by entering an URL. A Web page can be of the static or dynamic type. **With the help of HTML only, we can create static web pages.**

Hence, HTML is a markup language which is commonly used for creating attractive web pages with the help of styling, and which looks in a nice format on a web browser. An HTML document is made of many HTML tags and each HTML tag contains different content.

#### Features of HTML

- 1) It is a very **easy and simple language**. It can be easily understood and modified.
- 2) It is very easy to make an **effective presentation** with HTML because it has a lot of formatting tags.
- 3) It is a **markup language**, so it provides a flexible way to design web pages along with the text.
- 4) It facilitates programmers to add a link on the web pages (by html anchor tag), so it enhances the interest of browsing of the user.

5) It is **platform-independent** because it can be displayed on any platform like Windows, Linux, and Macintosh, etc.

6) It facilitates the programmer to add graphics, videos, and sound to the web pages which makes it more attractive and interactive.

7) HTML is a case-insensitive language, which means we can use tags either in lower-case or upper-case.

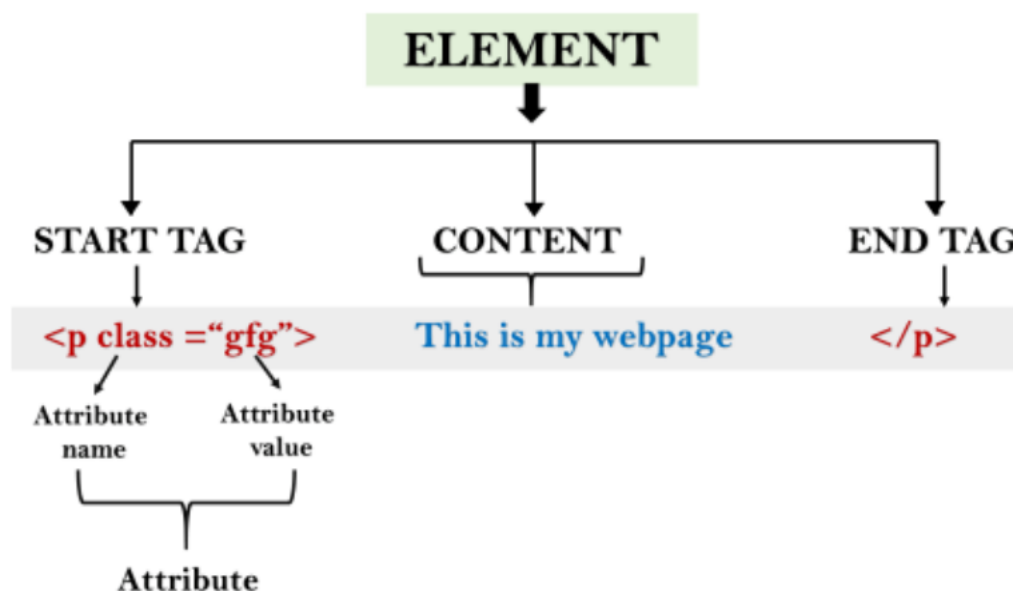
## Building blocks of HTML

An HTML document consists of its basic building blocks which are:

- o **Tags:** An HTML tag surrounds the content and applies meaning to it. It is written between < and > brackets.

- o **Attribute:** An attribute in HTML provides extra information about the element, and it is applied within the start tag. An HTML attribute contains two fields: name & value.

- o **Elements:** An HTML element is an individual component of an HTML file. In an HTML file, everything written within tags is termed as HTML elements.



## 2.1.2 CSS

CSS stands for Cascading Style Sheets. It is a style sheet language which is used to describe the look and formatting of a document written in markup language. It provides an additional feature to HTML. It is generally used with HTML to change the style of web pages and user interfaces. It can also be used with any kind of XML documents including plain XML, SVG and XUL.

CSS is used along with HTML and JavaScript in most websites to create user interfaces for web applications and user interfaces for many mobile applications.

- o CSS stands for Cascading Style Sheet.
- o CSS is used to design HTML tags.
- o CSS is a widely used language on the web.
  
- o HTML, CSS and JavaScript are used for web designing. It helps the web designers to apply style on HTML tags.

### **What does CSS do:**

- o You can add new looks to your old HTML documents.
  
- o You can completely change the look of your website with only a few changes in CSS code.

### **Why use CSS:**

These are the three major benefits of CSS:

#### 1) Solves a big problem

Before CSS, tags like font, color, background style, element alignments, border and size had to be repeated on every web page. This was a very long process. For example: If you are developing a large website where fonts and color information are added on every single page, it will become a long and expensive process. CSS was created to solve this problem. It was a W3C recommendation.

#### 2) Saves a lot of time

CSS style definitions are saved in external CSS files so it is possible to change the entire website by changing just one file.

### 3) Provide more attributes

CSS provides more detailed attributes than plain HTML to define the look and feel of the website.

## 2.1.3 JAVASCRIPT

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 was introduced in the year 1995 for adding programs to the webpages in the Netscape Navigator browser. Since then, 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.

Although, JavaScript has no connectivity with Java programming language. The name was suggested and provided in the times when Java was gaining popularity in the market. In addition to web browsers, databases such as CouchDB and MongoDB uses JavaScript as their scripting and query language.

### Features of JavaScript:

There are following features of JavaScript:

1. All popular web browsers support JavaScript as they provide built-in execution environments.
2. JavaScript follows the syntax and structure of the C programming language. Thus, it is a structured programming language.
3. JavaScript is a weakly typed language, where certain types are implicitly cast (depending on the operation).
4. JavaScript is an object-oriented programming language that uses prototypes rather than using classes for inheritance.
5. It is a light-weighted and interpreted language.
6. It is a case-sensitive language.
7. JavaScript is supportable in several operating systems including, Windows, macOS, etc.

8. It provides good control to the users over the web browsers.

### **Application of JavaScript:**

JavaScript is used to create interactive websites. It is mainly used for:

- Client-side validation,
- Dynamic drop-down menus,
- Displaying date and time,
- Displaying pop-up windows and dialog boxes (like an alert dialog box, confirm dialog box and prompt dialog box),
- Displaying clocks etc.

## **2.2 BACKEND**

**Backend Development:** Backend is the server-side of the website. It stores and arranges data, and also makes sure everything on the client-side of the website works fine. It is the part of the website that you cannot see and interact with. It is the portion of software that does not come in direct contact with the users. The parts and characteristics developed by backend designers are indirectly accessed by users through a front-end application. Activities, like writing APIs, creating libraries, and working with system components without user interfaces or even systems of scientific programming, are also included in the backend.

Back end development refers to the server side of an application and everything that communicates between the database and the browser.

Back end Development refers to the server side of development where you are primarily focused on how the site works. Making updates and changes in addition to monitoring functionality of the site will be your primary responsibility. This type of web development usually consists of three parts: a server, an application, and a database. Code written by back end developers is what communicates the database information to the browser. Anything you can't see easily with the eye such as databases and servers is the work of a back end developer. Back end developer positions are often called programmers or web developers.

## 2.2.1 NODEJS

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.

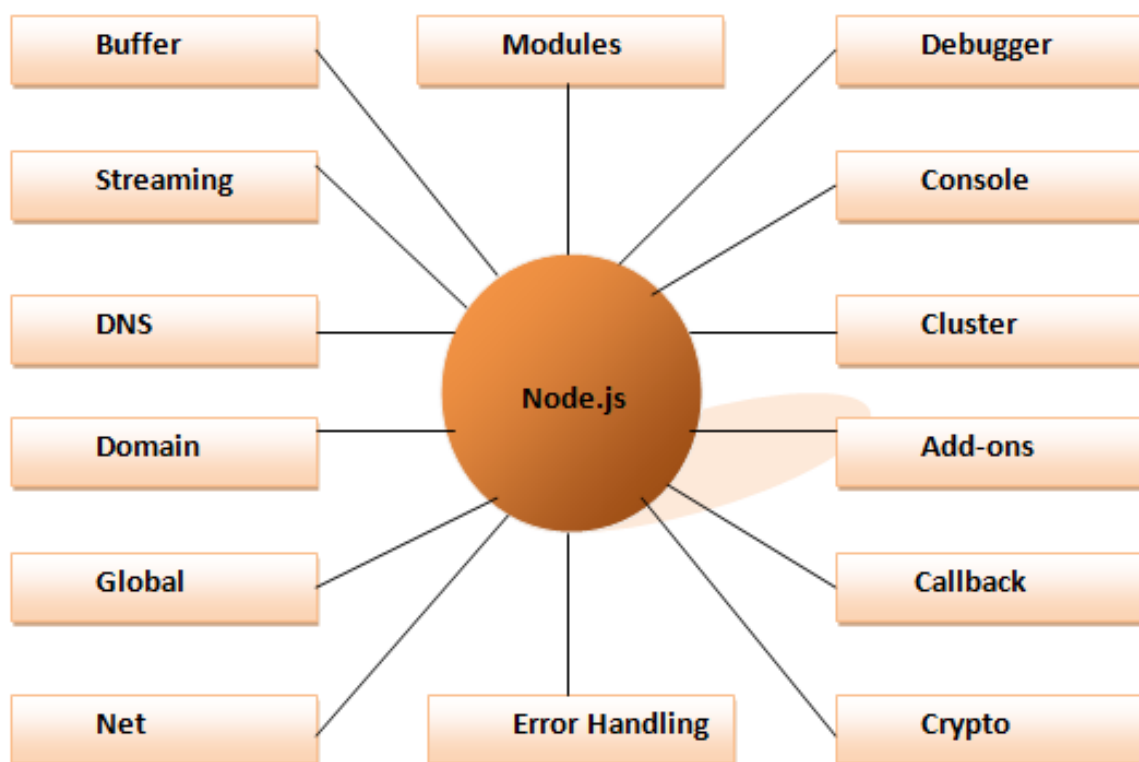
The definition given by its official documentation is as follows:

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.

### Different parts of Node.js

The following diagram specifies some important parts of Node.js:



## Features of Node.js

Following is a list of some important features of Node.js that makes it the first choice of software architects.

1. **Extremely fast:** Node.js is built on Google Chrome's V8 JavaScript Engine, so its library is very fast in code execution.
2. **I/O is Asynchronous and Event Driven:** All APIs of Node.js library are asynchronous i.e. non-blocking. So a Node.js based server never waits for an API to return data. The server moves to the next API after calling it and a notification mechanism of Events of Node.js helps the server to get a response from the previous API call. It is also a reason that it is very fast.
3. **Single threaded:** Node.js follows a single threaded model with event looping.
4. **Highly Scalable:** Node.js is highly scalable because event mechanism helps the server to respond in a non-blocking way.
5. **No buffering:** Node.js cuts down the overall processing time while uploading audio and video files. Node.js applications never buffer any data. These applications simply output the data in chunks.
6. **Open source:** Node.js has an open source community which has produced many excellent modules to add additional capabilities to Node.js applications.
7. **License:** Node.js is released under the MIT license.

### 2.2.2 EXPRESSJS

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.

Let's see some of the core features of Express framework:

- It can be used to design single-page, multi-page and hybrid web applications.
- It allows to setup middlewares to respond to HTTP Requests.



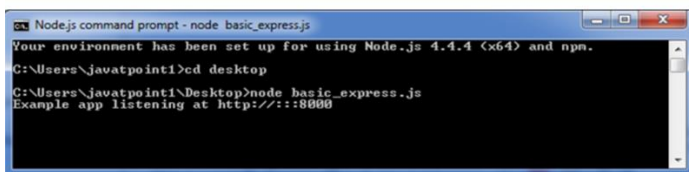
- It defines a routing table which is used to perform different actions based on HTTP method and URL.
- It allows to dynamically render HTML Pages based on passing arguments to templates.

---

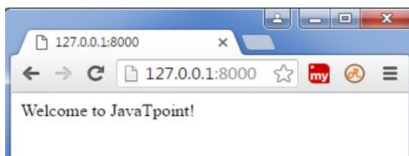
## Why use Express

- Ultra fast I/O
- Asynchronous and single threaded
- MVC like structure
- Robust API makes routing easy

```
var express = require('express');
var app = express();
app.get('/', function (req, res) {
  res.send('Welcome to JavaTpoint!');
});
var server = app.listen(8000, function () {
  var host = server.address().address;
  var port = server.address().port;
  console.log('Example app listening at http://%s:%s', host, port);
});
```



Output:



## 2.2.3 SOCKET.IO

Socket.IO is a JavaScript library for **real-time web applications**. It enables real-time, bi-directional communication between web clients and servers. It has two parts – a **client-side library** that runs in the browser, and a **server-side library** for node.js. Both components have an identical API.

## Real-time Applications:

A real-time application (RTA) is an application that functions within a period that the user senses as immediate or current.

Some examples of real-time applications are –

- **Instant messengers** – Chat apps like Whatsapp, Facebook Messenger, etc. You need not refresh your app/website to receive new messages.
- **Push Notifications** – When someone tags you in a picture on Facebook, you receive a notification instantly.
- **Collaboration Applications** – Apps like google docs, which allow multiple people to update same documents simultaneously and apply changes to all people's instances.
- **Online Gaming** – Games like Counter Strike, Call of Duty, etc., are also some examples of real-time applications.

## Why Socket.IO?

Writing a real-time application with popular web applications stacks like LAMP (PHP) has traditionally been very hard. It involves polling the server for changes, keeping track of timestamps, and it is a lot slower than it should be.

Sockets have traditionally been the solution around which most real-time systems are architected, providing a bi-directional communication channel between a client and a server. This means that the server can push messages to clients. Whenever an event occurs, the idea is that the server will get it and push it to the concerned connected clients.

Socket.IO is quite popular, it is used by **Microsoft Office, Yammer, Zendesk, Trello,** and numerous other organizations to build robust real-time systems. It is one of the most powerful **JavaScript frameworks** on **GitHub**, and most depended-upon NPM (Node Package Manager) module. Socket.IO also has a huge community, which means finding help is quite easy.

## 2.2.4 NODEMON

The nodemon Module is a module that develops node.js based applications by automatically restarting the node application when file changes in the directory are detected. Nodemon does not require any change in the original code and method of development.

## Advantages of Using nodemon Module:

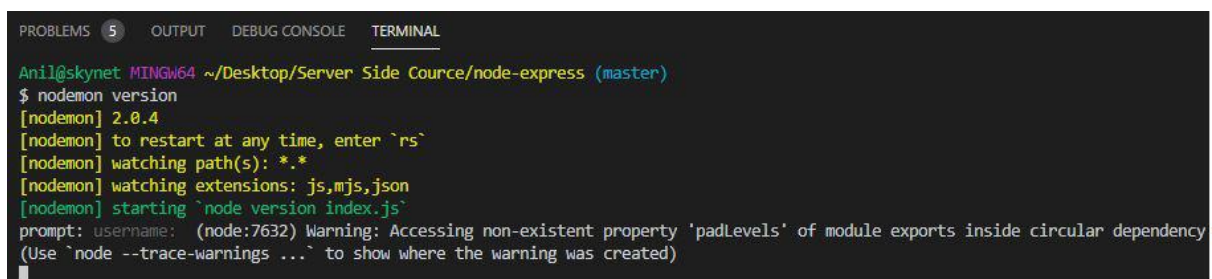
1. It is easy to use and easy to get started.
2. It does not affect the original code and no instance require to call it.
3. It help to reduce the time of typing the default syntax node <file name> for execution again and again.

**Installation:** Install the module using the following command:

```
npm install -g nodemon
```

After installing the module you can check the current version of the module by typing on console as shown below:

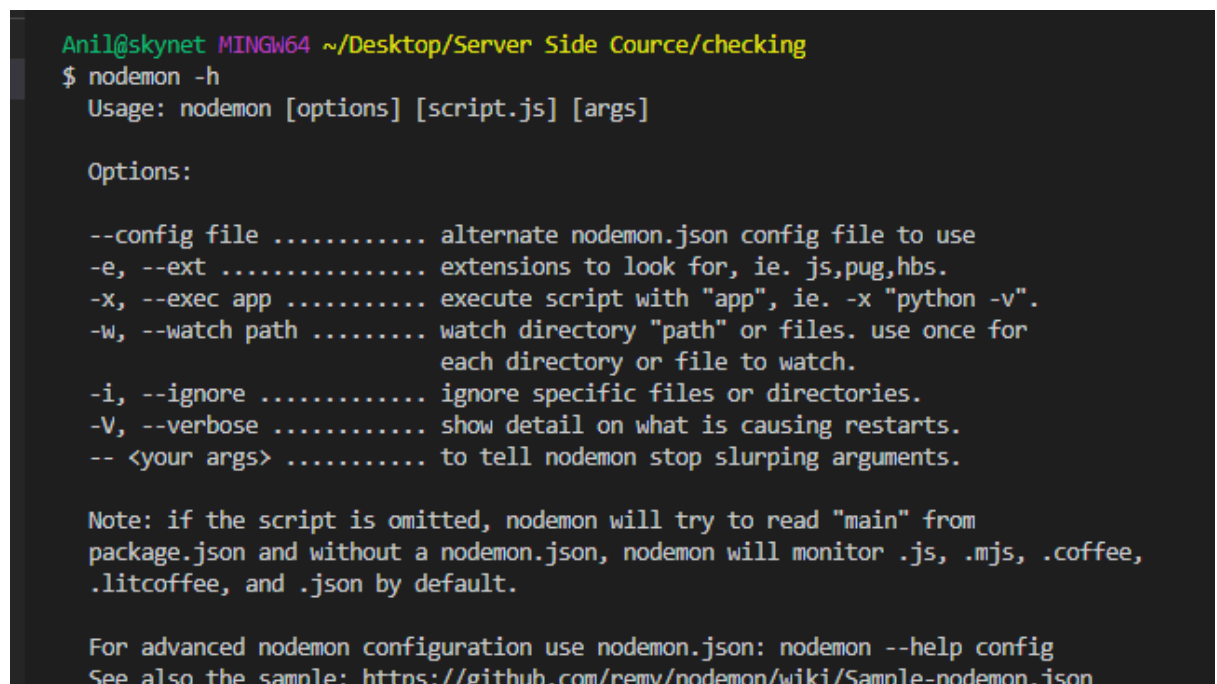
nodemon version



```
PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL
Anil@skynet MINGW64 ~/Desktop/Server Side Course/node-express (master)
$ nodemon version
[nodemon] 2.0.4
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node version index.js`
prompt: username: (node:7632) Warning: Accessing non-existent property 'padLevels' of module exports inside circular dependency
(Use `node --trace-warnings ...` to show where the warning was created)
```

## Usage:

1. The nodemon wraps your application, so you can pass all the arguments you would normally pass to your app:
2. nodemon [your node app]
3. Options available for nodemon are shown below:
4. nodemon -h



```
Anil@skynet MINGW64 ~/Desktop/Server Side Course/checking
$ nodemon -h
Usage: nodemon [options] [script.js] [args]

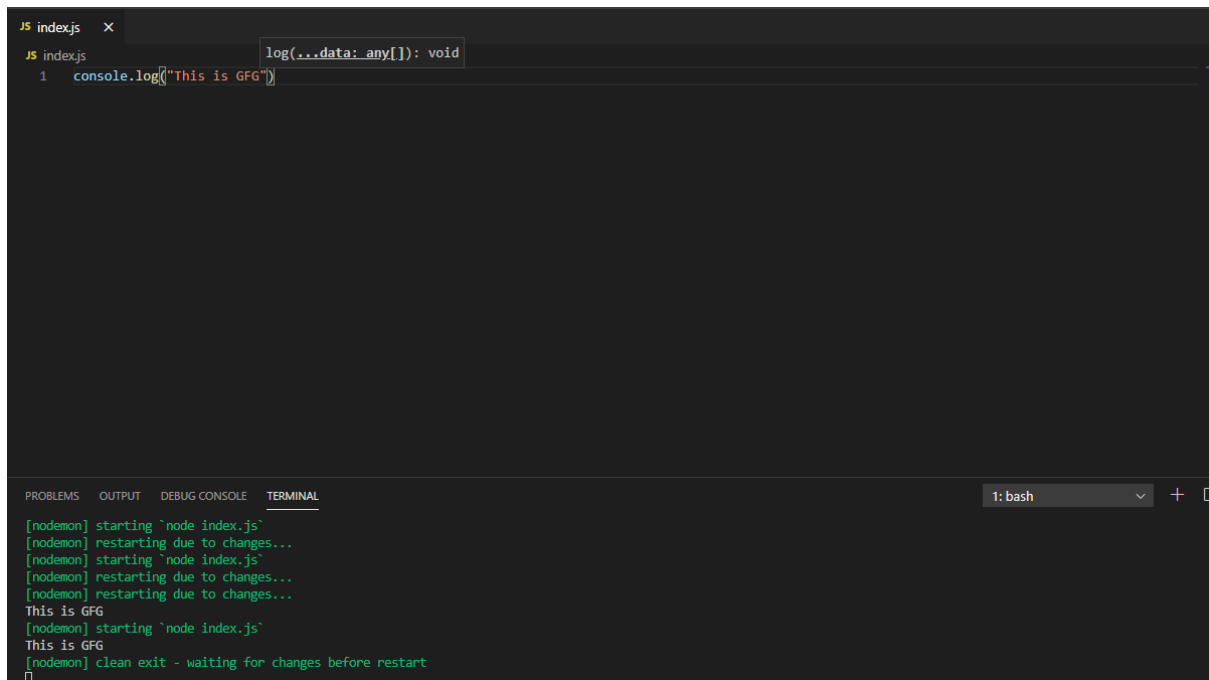
Options:

--config file ..... alternate nodemon.json config file to use
-e, --ext ..... extensions to look for, ie. js,pug,hbs.
-x, --exec app ..... execute script with "app", ie. -x "python -v".
-w, --watch path ..... watch directory "path" or files. use once for
                        each directory or file to watch.
-i, --ignore ..... ignore specific files or directories.
-V, --verbose ..... show detail on what is causing restarts.
-- <your args> ..... to tell nodemon stop slurping arguments.

Note: if the script is omitted, nodemon will try to read "main" from
package.json and without a nodemon.json, nodemon will monitor .js, .mjs, .coffee,
.litcoffee, and .json by default.

For advanced nodemon configuration use nodemon.json: nodemon --help config
See also the sample: https://github.com/remy/nodemon/wiki/Sample-nodemon.json
```

**Steps to run the program:** Use the following command to run the file as shown below:  
nodemon index.js



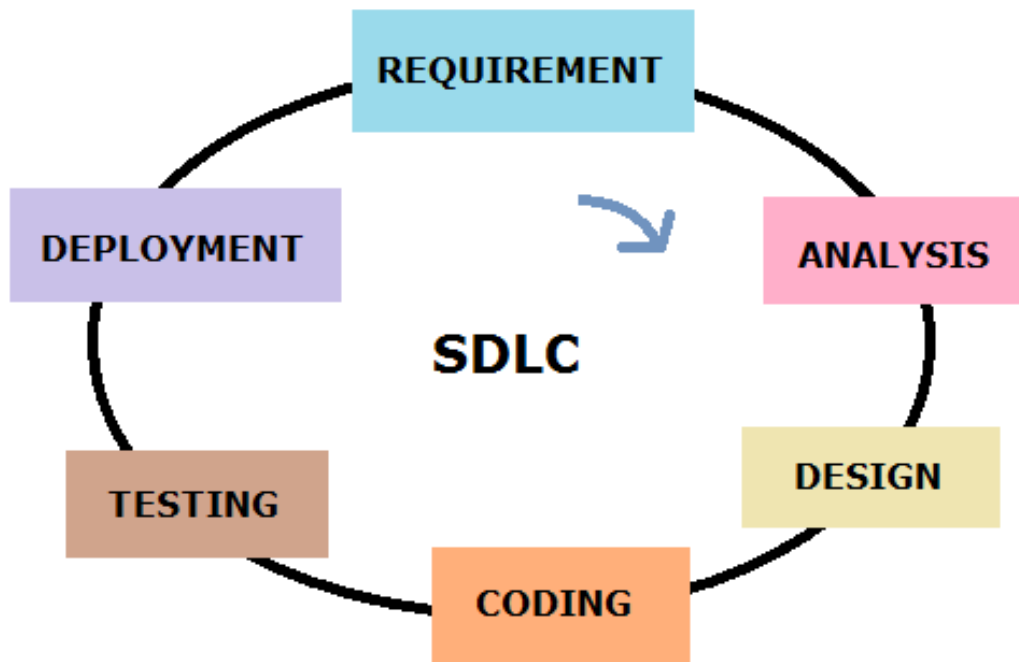
```
JS index.js x
JS index.js log(...data: any[]): void
1 console.log('This is GFG')

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: bash
[nodemon] starting `node index.js`
[nodemon] restarting due to changes...
[nodemon] starting `node index.js`
[nodemon] restarting due to changes...
[nodemon] restarting due to changes...
This is GFG
[nodemon] starting `node index.js`
This is GFG
[nodemon] clean exit - waiting for changes before restart
```

It automatically check the statements and the syntax of the program while writing new statements and show the result on the console.

## 2.3 SDLC (SOFTWARE DEVELOPMENT LIFE CYCLE)

SDLC is a process followed for making a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.



A typical Software Development Life Cycle consists of the following stages

### Stage 1: Requirement

Requirement analysis is the most important and fundamental stage in SDLC. It is performed by the senior members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry. This information is then used to plan the basic project approach and to conduct product feasibility study in the economical, operational and technical areas.

Planning for the quality assurance requirements and identification of the risks associated with the project is also done in the planning stage. The outcome of the technical feasibility study is to define the various technical approaches that can be followed to implement the project successfully with minimum risks

### Stage 2: Analysis

Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts. This is done through an **SRS (Software Requirement Specification)** document which consists of all the product requirements to be designed and developed during the project life cycle.

### Stage 3: Designing

SRS is the reference for product architects to come out with the best architecture for the product to be developed. Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a DDS - Design Document Specification.

This DDS is reviewed by all the important stakeholders and based on various parameters as risk assessment, product robustness, design modularity, budget and time constraints, the best design approach is selected for the product.

A design approach clearly defines all the architectural modules of the product along with its communication and data flow representation with the external and third party modules (if any). The internal design of all the modules of the proposed architecture should be clearly defined with the minutest of the details in DDS.

#### **Stage 4: Building or Coding**

In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per DDS during this stage. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle.

Developers must follow the coding guidelines defined by their organization and programming tools like compilers, interpreters, debuggers, etc. are used to generate the code. Different high level programming languages such as C, C++, Pascal, Java and PHP are used for coding. The programming language is chosen with respect to the type of software being developed.

#### **Stage 5: Testing**

This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC. However, this stage refers to the testing only stage of the product where product defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS.

#### **Stage 6: Deployment**

Once the product is tested and ready to be deployed it is released formally in the appropriate market. Sometimes product deployment happens in stages as per the business strategy of that organization. The product may first be released in a limited segment and tested in the real business environment (UAT- User acceptance testing).

Then based on the feedback, the product may be released as it is or with suggested enhancements in the targeting market segment. After the product is released in the market, its maintenance is done for the existing customer base.

## **CHAPTER 3**

### **SYSTEM-ANALYSIS**

System analysis is the process of gathering and interpreting facts, diagnosing problems and using the information to recommend improvements on the system. System analysis is a problem solving activity that requires intensive communication between the system users and system developers.

System analysis or study is an important phase of any system development process. The system is viewed as a whole, the inputs are identified and the system is subjected to close study to identify the problem areas. The solutions are given as a proposal. The proposal is reviewed on user request and suitable changes are made. This loop ends as soon as the user is satisfied with the proposal.

#### **3.1 EXISTING SYSTEM:**

Chat applications and messaging apps are surging in popularity. The reason for this is simple—people love to chat. It's the preferred method of communication in a multitude of different scenarios from collaborating with a colleague to checking in on a loved one.

Chat and messaging applications help foster a sense of community and connection that other forms of communication can't reproduce.

Understanding the system design & architecture of a real-time chat application can be an intimidating task. Fortunately, it's a task that we know well. In this article, we're going to break down chat app architecture so that you can identify the best way to add chat to your website or app.



### **3.1.1 DRAWBACKS:**

1. Less user friendly.
2. User has to enter some code of line to run it globally.
3. It is not encrypted and decrypted based application.
4. Time consuming process. User has to plan accordingly.
5. Not in reach of distant user.

## **3.2 TECHNICAL REQUIREMENT:**

### **3.2.1 SOFTWARE REQUIREMENT:**

Operating System : Windows 7/8/10 or MacOS or Linux

Software : Visual Studio Code

User Interface : HTML, CSS

Programming : JavaScript

Packages : NodeJs, ExpressJs, Nodemon

### **3.2.2 HARDWARE REQUIREMENT:**

Processor : Any

Hard Disk : 250GB

RAM : 4GB or More

## **3.3 FEASIBILITY STUDY:**

Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economic feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited

resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- o Economic Feasibility
- o Operational Feasibility
- o Technical Feasibility

### **3.3.1 ECONOMIC FEASIBILITY:**

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economic feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs.

The system is economically feasible. It does not require any addition hardware or software.

Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economic feasibility for certain.

### **3.3.2 OPERATIONAL FEASIBILITY**

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following:-

- o Is there sufficient support for the management from the users?
- o Will the system be used and work properly if it is being developed and implemented?
- o Will there be any resistance from the user that will undermine the possible application benefits?

This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits.

The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

### **3.3.3 TECHNICAL FEASIBILITY:**

The technical issue usually raised during the feasibility stage of the investigation includes the following:

- o Does the necessary technology exist to do what is suggested?
- o Does the proposed equipment have the technical capacity to hold the data required to use

the new system?

- o Will the proposed system provide adequate response to inquiries, regardless of the number

or location of users?

- o Can the system be upgraded if developed?
- o Are there technical guarantees of accuracy, reliability, ease of access and data security?

Earlier no system existed to cater to the needs. The current system developed is technically feasible. It is a web based user interface. Thus it provides an easy access to the users. The database's purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users

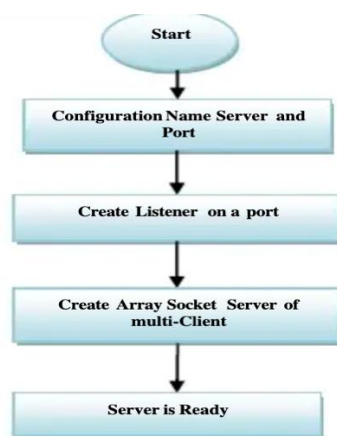
would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security. The software and hardware requirements for the development of this project are not many and are already available in-house at NIC or are available as free as open source.

### CHATCORD SPECIFICATION

Socket.io is a Javascript library for web apps that allows real-time communication between clients and servers. It's built on top of the Web sockets API (client) and Node.js (server). The most common use cases for Web sockets and socket.io are chat applications or a social media feeds in which a user's page (client) receives messages or posts from other users.

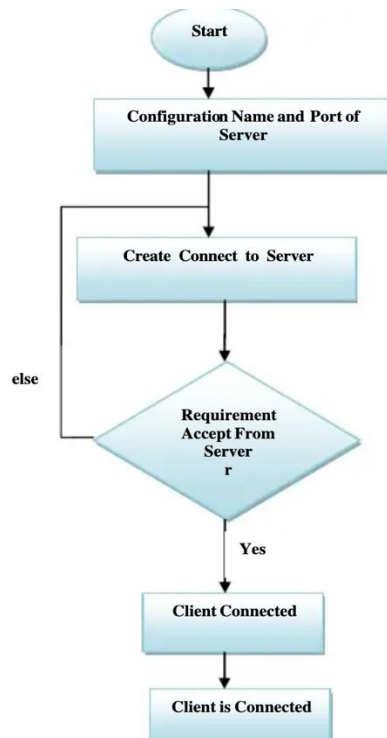
#### 4.1 CHAT SERVER RESPONSIBILITIES:

- Serve the HTML, CSS and JavaScript client files to the users.
- Start the Socket.io connection.
- Receive events from clients (like a new chat message) and broadcast them to other clients.



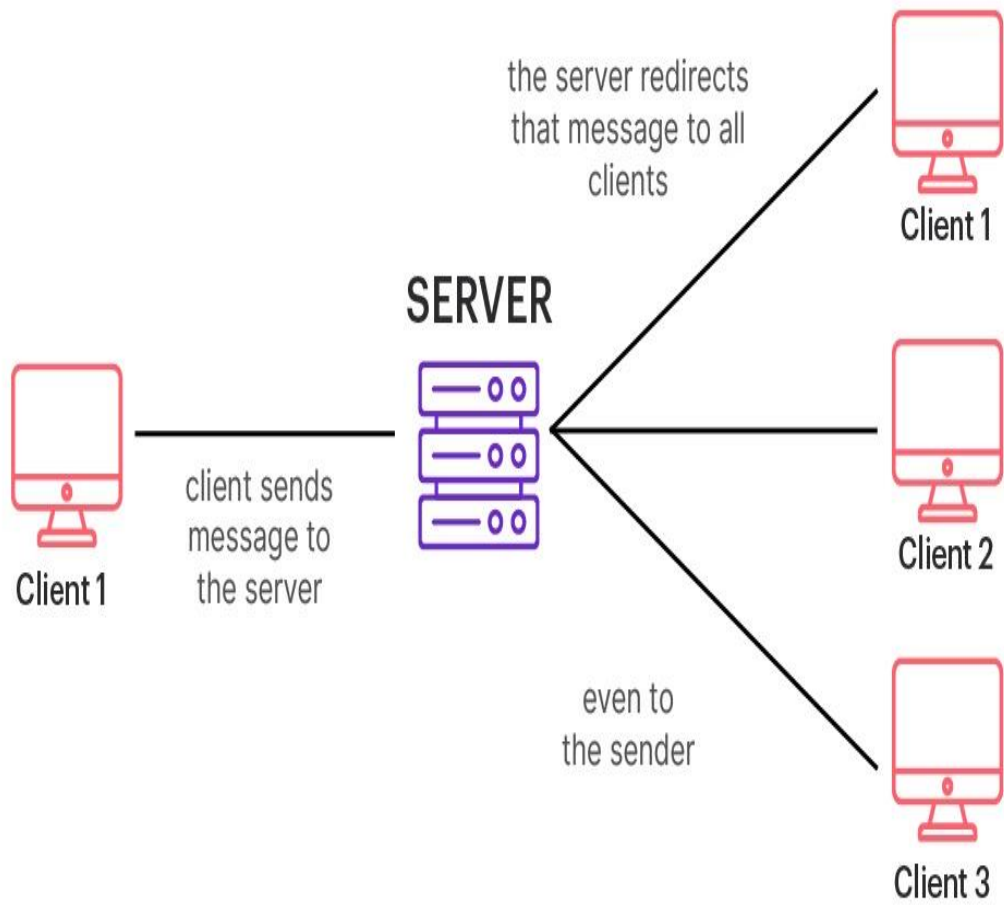
#### 4.2 CHAT CLIENT RESPONSIBILITIES:

- Load socket.io client library from a CDN.
- Establish connection with the Socket.io running in our server.
- Ask the user to enter his name so we can identify him in the chat.
- Emit and receive events to/from Socket.io running in our server.
- Add our own messages to the chat via JavaScript.

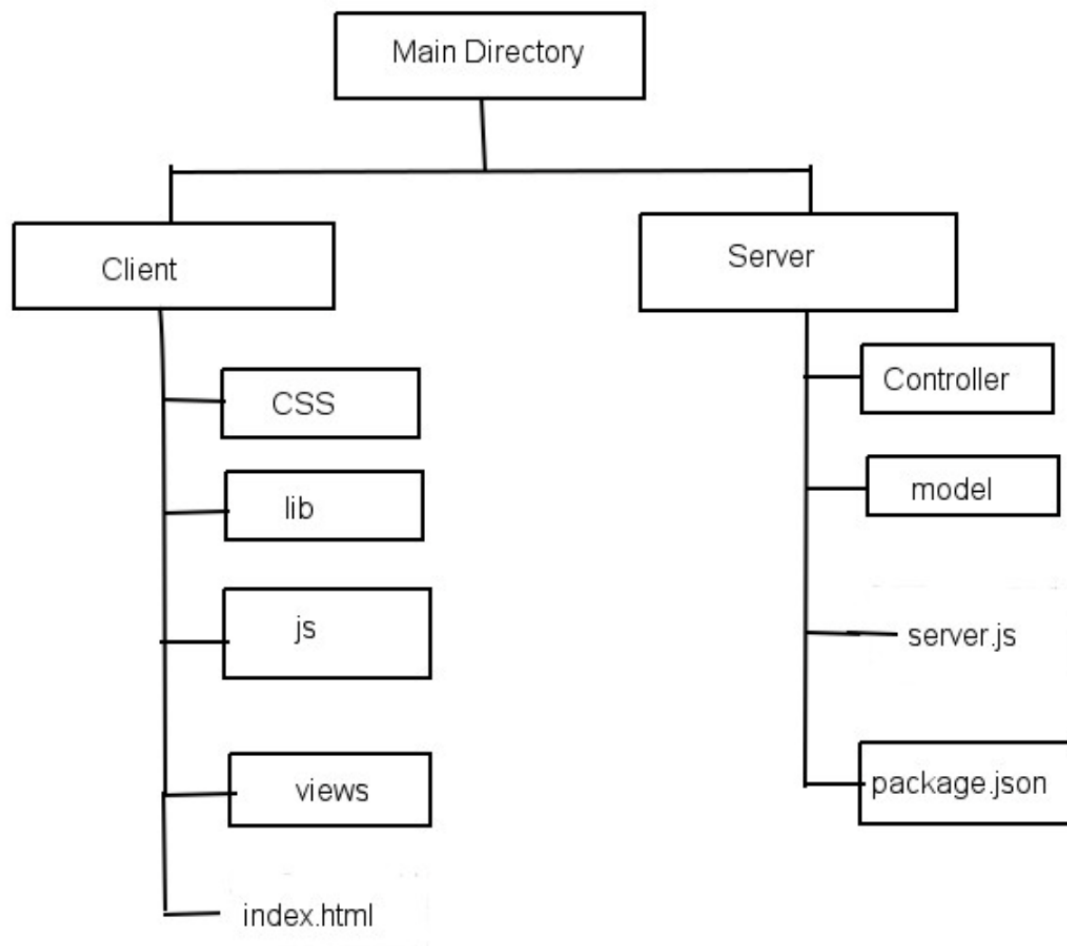


### 4.3 Components of Chat Application Infrastructure

While the system design of a chat app is unique in how it deals with the idiosyncratic business needs, you can always break it down to two major components: the chat client and the chat server.



## 4.4 DIRECTORY STRUCTURE OF PROJECT:



Directory structure of the project

## 4.5 Building the Server:

**Step 1** — Start the project :

Go to Server directory and run this command:

```
npm init
```

This will start a new project. Provide all the details required.  
The *package.json* will be created and will look something like this.

```
{
  "name": "chat",
  "version": "1.0.0",
  "description": "Chat application",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "Your name",
  "license": "ISC"
}
```

## Step 2 — Install the dependencies.

- **socket.io** — is a *javascript* library for real-time web applications. It enables real-time, bi-directional communication between web clients and servers.
- **express** — is a *Node.js* web application framework. It provides the set of features to develop the web and mobile applications. One can respond to HTTP request using different middlewares and also render HTML pages.

```
npm install --save socket.io
```

```
npm install --save express
```

This will install required dependencies and add those to *package.json*. An extra field will be added to *package.json* which will look like this:

```
"dependencies": {
  "express": "^4.14.0",
  "socket.io": "^1.4.8"
}
```

## Step 3 — Creating the Server

Create a server which serves at port 3000 and will send the html when called.

Initialize a new socket connection by passing HTTP object.

Event *connection* will be listening for incoming sockets.

Each socket emits *disconnect* event which will be called whenever a client disconnects.

- **socket.on** waits for the event. Whenever that event is triggered the callback function is called.



- **io.emit** is used to emit the message to all sockets connected to it.

The syntax is:

```
socket.on('event', function(msg){})  
io.emit('event', 'message')
```

Create a server with name *server.js*. It should:

- print a message to the console upon a user connecting
- listen for *chat message* events and broadcast the received message to all sockets connected.
- Whenever a user *disconnects*, it should print the message to the console.

The server will look something like this:

```
var app = require('express')();  
var http = require('http').Server(app);  
var io = require('socket.io')(http);  
  
app.get('/', function(req, res){  
  res.sendFile('index.html');  
});  
  
io.on('connection', function(socket){  
  console.log('user connected');  
  socket.on('chat message', function(msg){  
    io.emit('chat message', msg);  
  });  
  socket.on('disconnect', function(){  
    console.log('user disconnected');  
  });  
});  
  
http.listen(3000, function(){  
  console.log('listening on *:3000');  
});
```

## 4.6 Building the Client:

Create the index.html in the client directory, style.css in the css directory and app.js in js directory in the client.

## index.html:

Let us write a simple HTML which can take our message and also display it.

Include *socket.io-client* and *angular.js* in your HTML script.

```
<script src="/path/to/angular.js"></script>
<script src="/socket.io/socket.io.js"></script>
```

**socket.io** serves the client for us. It defaults to connect to the host that serves the page. Final HTML looks something like this:

```
<!doctype html>
<html ng-app="myApp">
  <head>
    <title>Socket.IO chat</title>
    <link rel="stylesheet" href="/css/style.css">
    <script src="/lib/angular/angular.js"></script>
    <script src="/socket.io/socket.io.js"></script>
    <script src="http://code.jquery.com/jquery-1.11.1.js">
  </script>
  <script src="/js/app.js"></script>
</head>
<body ng-controller="mainController">
  <ul id="messages"></ul>
  <div>
    <input id="m" ng-model="message" autocomplete="off" />
    <button ng-click="send()">Send</button>
  </div>
</body>
</html>
```

## css/style.css:

Give it some styling so that it looks like a chatbox. You can make use of any libraries.

```
* { margin: 0; padding: 0; box-sizing: border-box; }
body { font: 13px Helvetica, Arial; }
div { background: #000; padding: 3px; position: fixed; bottom: 0; width: 100%; }
div input { border: 0; padding: 10px; width: 90%; margin-right: .5%; }
```

```
div button { width: 9%; background: rgb(130, 224, 255); border: none; padding: 10px; }
#messages { list-style-type: none; margin: 0; padding: 0; }
#messages li { padding: 5px 10px; }
#messages li:nth-child(odd) { background: #eee; }
```

## js/app.js:

Create an angular.js app and initialize a socket connection.

- **socket.on** listens for a particular event. It calls a callback function whenever that event is called.
- **socket.emit** is used to emit the message to the particular event.

Basic syntax of both are:

```
socket.on('event name', function(msg){});
socket.emit('event name', message);
```

So whenever the message is typed and the button is clicked, call the function to send the message.

Whenever the socket receives a message, display it.

The JavaScript will look something like this:

```
var app=angular.module('myApp',[]);

app.controller('mainController',['$scope',function($scope){
  var socket = io.connect();
  $scope.send = function(){
    socket.emit('chat message', $scope.message);
    $scope.message="";
  }
  socket.on('chat message', function(msg){
    var li=document.createElement("li");
    li.appendChild(document.createTextNode(msg));
    document.getElementById("messages").appendChild(li);
  });
});
```

## 4.7 RUNNING THE APPLICATION:

Go to server directory where our server is present. Run the server using the following command:

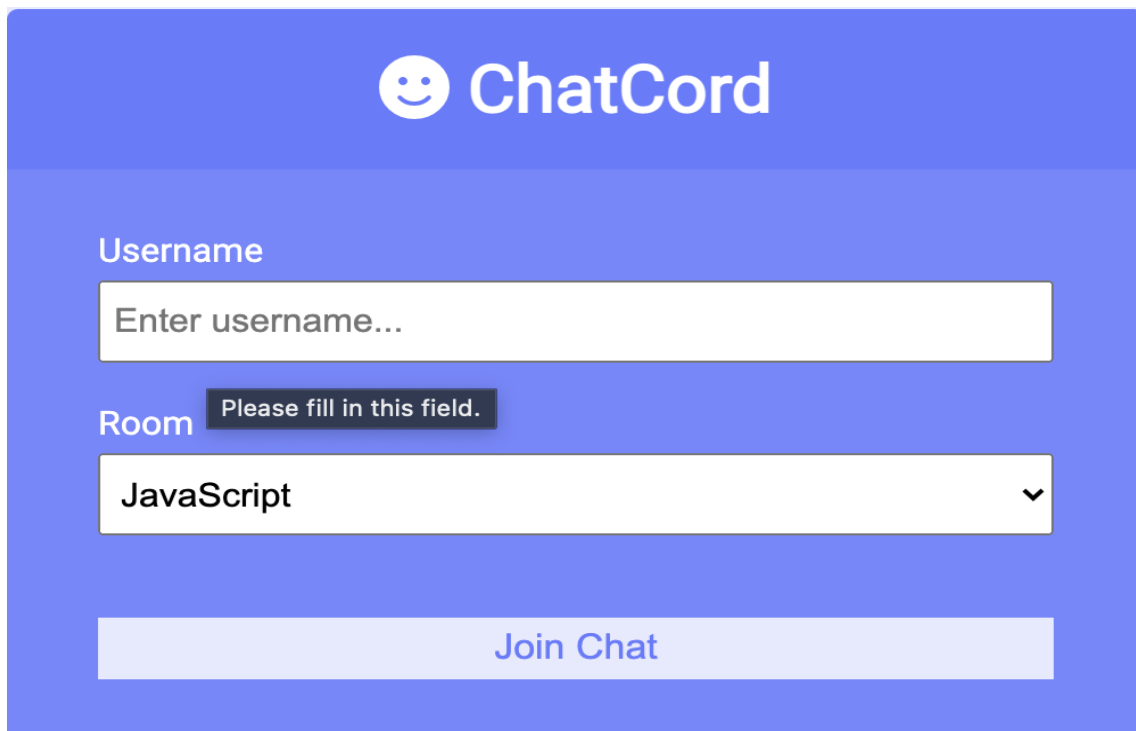
```
node server.js
```

The server starts running on port 3000. Go to the browser and type the following url:

```
http://localhost:3000
```

# SCREENSHOTS

## 1.Home Page



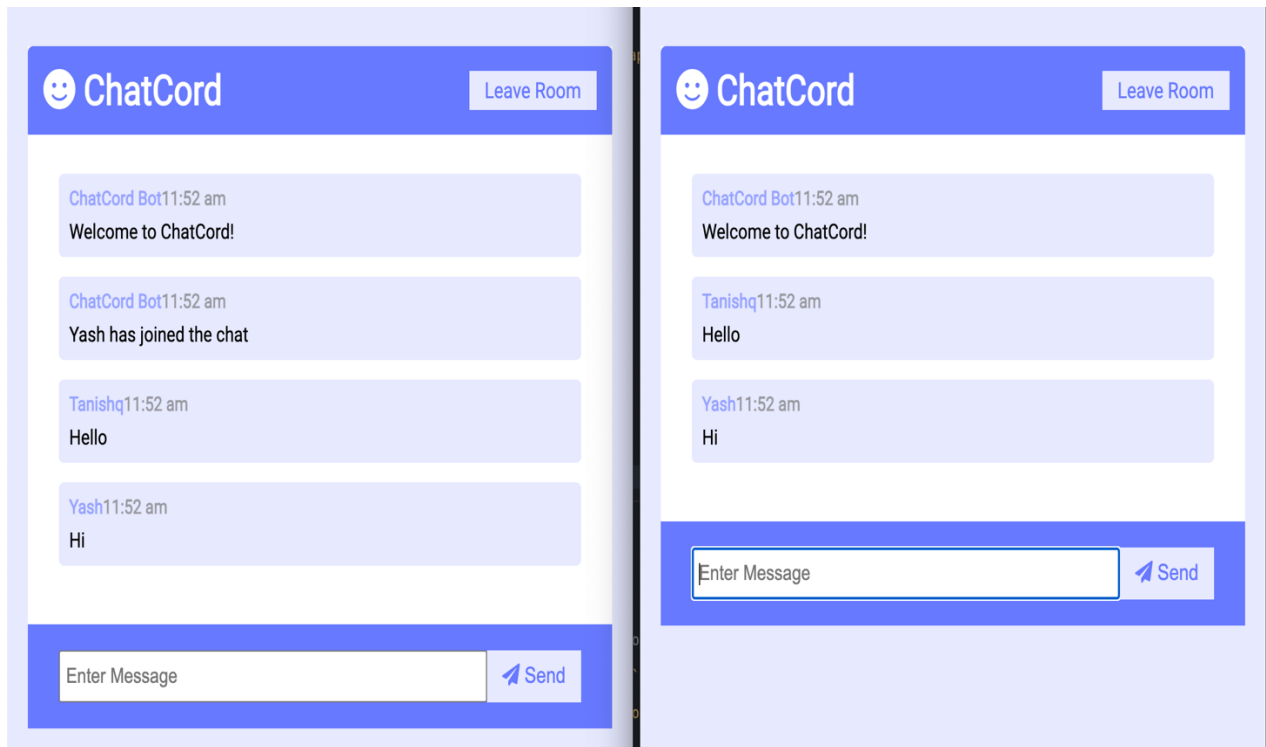
The screenshot shows the ChatCord home page. It has a blue header with the ChatCord logo (a smiley face icon) and the text "ChatCord". Below the header, there is a form with two input fields. The first field is labeled "Username" and contains the placeholder text "Enter username...". The second field is labeled "Room" and contains the text "JavaScript". A small black tooltip with the text "Please fill in this field." is visible next to the "Room" label. Below the input fields, there is a large blue button labeled "Join Chat".

## 2. Chat Room



The screenshot shows the ChatCord chat room interface. It has a blue header with the ChatCord logo and the text "ChatCord". In the top right corner of the header, there is a button labeled "Leave Room". Below the header, there is a sidebar on the left with a blue background. The sidebar contains the text "Room Name:" followed by "JavaScript" (which is highlighted with a blue background), and "Users" followed by "Tanishq". The main area of the chat room is white and contains a message from "ChatCord Bot" at "10:58 am" that says "Welcome to ChatCord!". At the bottom of the chat room, there is a blue input field labeled "Enter Message" and a blue button labeled "Send".

### 3. Clients Chatting



## CONCLUSION

The project entitled **REAL-TIME CHAT APP (CHATCORD)** was completed successfully. The system has been developed with much care and free of errors and at the same time it is efficient and less time consuming. The purpose of this project was to develop a web application for real time communication .

This project helped me in gaining valuable information and practical knowledge on several topics like designing web pages using html & css, usage of responsive templates, using JavaScript, Nodejs for coding and socket.io. Also the project helped me understanding about the development phases of a project and software development life cycle. We learned how to test different features of a project.

## REFERENCES

- [1] Ian V. Sandoval CLIENT/SERVER TECHNOLOGY, ,2005
- [2] Waleed Farah, DEVELOPING A CHAT SERVER, Fall 2000
- [3] Rahman, M. M., Development of Dialogue Based Object Oriented Client/server Applications On the Internet: An Online Interviewing System, AITMasters Thesis CS- 98-17, 1998.
- [4] Berson, Alex., Client-server architecture. McGraw-Hill, 1994.
- [5] Chat Rooms, [http://en.wikipedia.org/wiki/Chat\\_rooms](http://en.wikipedia.org/wiki/Chat_rooms),2002
- [6] Kendall & Kendall, Systems Analysis and Design, Sixth Edition, 2005 Pearson Prentice Hall
- [7] McGraw-Hill Osborne Media ,The Complete Reference JAVASCRIPT 2 fifth edition, 2002
- [8] Ian Darwin, Publisher: O'Reilly, JavaScript Cookbook (e-book), First Edition June 2001
- [9] Disadvantages of Using Skype ,[http://www.ehow.co.uk/list\\_5907068\\_disadvantages-using-skype.html](http://www.ehow.co.uk/list_5907068_disadvantages-using-skype.html)
- [10] Advantages and Disadvantages of Skype ,<http://wadams6.blogspot.com/2011/02/movie-platoon.html>
- [11] System Development Life Cycle (SDLC) Methodologies ,[http:// www.slepi.net/blog/system-development/system-development-life-cycle-sdlc-methodologies.html](http://www.slepi.net/blog/system-development/system-development-life-cycle-sdlc-methodologies.html)