

CHAPTER - 1

COMPANY'S PROFILE

CHAPTER – 1

COMPANY’S PROFILE

Magic Resorts Private Limited is a Private incorporated on 30 January 2012. It is classified as non-govt company and is registered at Registrar of Companies, Delhi. Its authorized share capital is Rs. 100,000 and its paid-up capital is Rs. 100,000. It is involved in Real estate activities with own or leased property. [This class includes buying, selling, renting and operating of self-owned or leased real estate such as apartment building and dwellings, non-residential buildings, developing and subdividing real estate into lots etc. Also included are development and sale of land and cemetery lots, operating of apartment hotels and residential mobile home sites.(Development on own account involving construction is classified in class 4520).]

Magic Resorts Private Limited's Annual General Meeting (AGM) was last held on 30 November 2021 and as per records from Ministry of Corporate Affairs (MCA), its balance sheet was last filed on 31 March 2021.

Directors of Magic Resorts Private Limited are Vivek Vazirchand Khanna, Munish Banwarilal Gupta, Satish Kumar Sharma and .

Magic Resorts Private Limited's Corporate Identification Number is (CIN) U70100HP2012PTC007435 and its registration number is 7435. Its Email address is info@magicresorts.co.in and its registered address is C/O MAGIC LANDBASE PVT LTD VILL DIKTU 2223 JHEOL Kangra HP 176001 IN , - , .

Current status of Magic Resorts Private Limited is - Active.

1.0 Company Details

CIN	<u>U70100HP2012PTC007435</u>
Company Name	MAGIC RESORTS PRIVATE LIMITED
Company Status	Active
RoC	RoC-Delhi
Registration Number	7435
Company	Company limited by Shares

CIN

U70100HP2012PTC007435

Category	
Company Sub Category	Non-govt company
Class of Company	Private
Date of Incorporation	30 January 2012
Age of Company	10 years, 3 month, 27 days
Activity	Real estate activities with own or leased property. [This class includes buying, selling, renting and operating of self-owned or leased real estate such as apartment building and dwellings, non-residential buildings, developing and subdividing real estate into lots etc. Also included are development and sale of land and cemetery lots, operating of apartment hotels and residential mobile home sites.(Development on own account involving construction is classified in class 4520).

CHAPTER – 2

INTRODUCTION TO PROJECT

CHAPTER – 2

INTRODUCTION TO PROJECT

2.0 Organization the thesis was written to

The requirement for a Bachelor of Engineering's degree in Information Technology is that student must do training in order to gain experience necessary in the technological field. The work must be theoretical and practical work.

The thesis project was carried by me and my friend separately during our training, and my practical training was completed officially starting on 1st November 2021 for about 4 months. The experience and skills that are gained and challenges which are faced in solving problems during the training. My training included various activities but mainly were learning and practicing, Website development, Project Handling, And Conversational Skills. I express my gratitude to Magic Resorts Pvt Ltd for giving me this opportunity to do Information Technology training with a partial fulfillment of the requirements for the Bachelor of Engineering in Information Technology.

Throughout this training, I was very honored and lucky with the encouragement and guidance from the firm's Director, Vivek Khanna. I thank the staff of Magic Resorts Pvt Ltd for being guiding and sharing the knowledge in technology. I also thank to other staff for creating a good learning environment.

Today Developers around the world are making efforts to enhance user experience of using application as well as to enhance the developer's workflow of designing applications to deliver projects and rollout change requests under strict timeline.

Stacks can be used to build web applications in the shortest span of time. The stacks used in web development are basically the response of software engineers to current demands. They have essentially adopted pre-existing frameworks (including JavaScript) to make their lives easier. While there are many, MEAN and MERN are just two of the popular stacks that have evolved out of JavaScript. Both stacks are made up of open-source components and offer an end-to-end framework for building comprehensive web apps that enable browsers to connect with databases.

The common theme between the two is JavaScript and this is also the key benefit of using either stack. One can basically avoid any syntax errors or any confusion by just coding in one programming language, JavaScript. Another advantage of building web projects with MERN is the fact that one can benefit from its enhanced flexibility. In order to understand MERN stack, we need to understand the four components that make up the MERN stack (fig.2.1), namely – MongoDB, Express.js, React and Node.js.

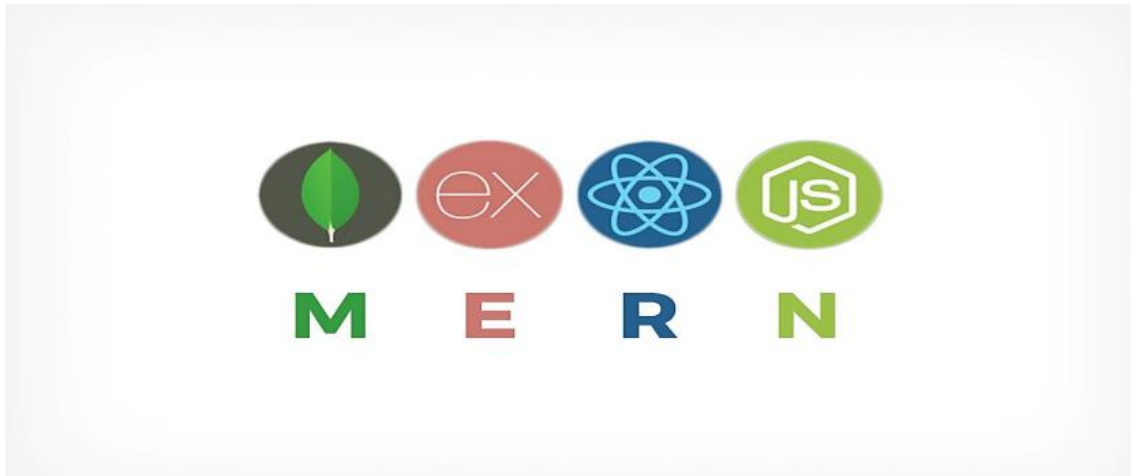


Figure 2.1 MERN Stack

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's.
- 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 organizational field where employees can connect through LAN.
- The main purpose of this project is to provide multi chatting functionality through network.

2.2 Innovative Ideas of Project

- **GUI:** Easy to use GUI (Graphical User Interface), hence any user with minimal knowledge of operating a system can use the software.
- **Platform independence:** The messenger operates on any system irrelevant of the underlying operating system.
- **Unlimited clients:** "N" number of users can be connected without any performance degradation of the server.

2.3 Project Objective

- **Communication:** To develop an instant messaging solution to enable users to seamlessly communicate with each other.
- **User friendliness:** The project should be very easy to use enabling even a novice person to use it.

2.4 Scope of the Project

- Broadcasting Chat Server Application is going to be a text communication software; it will be able to communicate between two computers using point to point communication.
- The limitation of Live Chat is it does not support audio conversations. To overcome this limitation, we are concurrently working on developing better technologies.
- Companies would like to have communication software wherein they can communicate instantly within their organization.
- The fact that the software uses an internal network setup within the organization makes it very secure from outside attacks.

2.5 Used Technologies

In this Section, all technologies of the Chat App are mentioned with the background.

2.5.1 Chat Background

In this section, the background history of the chat is discussed

2.5.1.1 SMS

Text message is the short electronic message which is called as SMS. It allows user to send and receive small text messages with little cost. It is a more secure and reliable way to send and receive short message that consists of around 160 characters. For SMS does not require an App and the internet and it is available on all mobiles while MMS (multimedia services) require the internet or a WIFI service. MMS is the enhanced form of the SMS, which enables users to send and receive video and pictures images.

SMS is being used for the last 20 years on GSM standard as communication technology enhanced to 3G, 4G and 5G SMS developed to MMS which allows the user to send and receive video and pictures messages. While Chat App is a free of cost, it occurs via a desktop or a mobile app.

2.5.1.2 Instant Messaging

Instant messaging platform was created in the 1960s by MIT allowing up to 30 users to log in at a time. Instant messaging is a one-to-one conversation-based platform which allows user to connect with another person's device for sharing the text and picture image. In other words, instant messaging is a sort of real-time online chat which is enabled via a special software application. Since technology has improved the instant messaging has gained popularity and it is used in every life. The downside of instant messenger or IM is that the other user must have the same device or software program for the communication. Examples of some famous instant messenger applications are shown in Figure.



Figure 2.2 Instant message categories

2.5.1.3 Chat App

The word communication has been derived from the Greek word, “Communicate”, meaning the exchange or sharing of ideas or expression. Chat was created at the University of Illinois in 1973. Chat App is a platform that enable messaging either one-to-one or to group members usually occurring in a chat room where multiple people can join and shares their interests. In other words, Chat App is also called “Real- Time” chat because it is on-going message service which processes the hundreds of messages between the sender and the receiver without any delay.

Some examples of popular Chat Apps are WhatsApp, Viber, LINE, Skype, WeChat, IM, tt, C, Tango, N, kik, TALK, talk and ebuddy in Figure 3.2 .



Figure 2.3 Chat App programs

Chat App figure is different than the old tradition instant messaging applications because it requires only a desktop and a laptop, for example, Yahoo messenger and Hotmail messenger, while Chat App is enabled via mobile apps on smart phones such as Android, IOS as well as a desktop.

2.5.1.4 Benefits of Chat

Benefits of chat include many companies and schools rely on live chat today because it has the following benefits.

- **Real-time text examine:** It is one of the benefits between customer and agent. A customer sends the request for a solution and an agent view and thinks about the solution of the issue and replies to the customer immediately.
- **Instant customer feedback:** Customer can rate the company service right after using the chat service.
- **No waiting queues:** Customers can reach the company agent immediately.
- **On a browser site:** The customer and agent conversation take place on the website during a chat, the agent can see where the customer is currently looking the issue so the agent can easily guide the user by copying the link or screenshot image.
- **Purchasing solution:** On online shopping point of view, is one of the important advantages of live chat? A customer can ask the agent about the product cost in the middle of online shopping
- **File transfer:** A customer can provide file, document and images of the questioned issue upon asking by the company agent.
- **Mobile Messaging Integrations:** Now, companies add Live Chat messenger like Viber, WhatsApp and Facebook to a company website. Customers are connected with the company's central live chat system via advance live software programs.
- **24/7 Support Service:** A company is available for the customers 24/7 without any time zone restrictions. Customers can talk to a chatbot even out of office hours.
- **Less cost:** Live chat is cheaper than the traditional phone call for the customer to access the company agent for questioning and solving issues.

2.5.1.5 Disadvantages of Chat

Disadvantages of Chat are:

- **Internet Access:** The user needs constant access to the internet during a chat conversation with colleague or company's help desk. Also, users have internet access difficulties in developing countries.
- **Communication difficulties:** Some users have difficulties in chat communication with the help desk, particularly elder and disable people.
- **Virus Risk:** Companies can have virus risks attached files, pictures or images when transferring them from the user to company during chat.
- **User Unfriendly:** Live chat is unfriendly for the novice computer users
- **Unfriendly:** It is observed unfriendly on mobile platforms.

2.5.2 HTML

HTML is a programming language which was developed by Berners-Lee in 1980 but standardized in 1995. HTML stands for Hyper Text Markup Language. It is one of the most popular languages on the planet used for developing web pages and websites. HTML is used for the structure of web documents i.e., headlines, paragraphs and images etc. HTML is known as the basics of all programming language in the globe. HTML is highly used in other programming languages such as JavaScript and CSS for the its display attributes. It consists of opening and closing tags as with the file extension .html.

2.5.2.1 Advantages of HTML

- HTML is easy to learn and use without any prerequisite's knowledge in programming.
- HTML has no license fees.
- HTML is supported by all browsers on the web and supported by all platforms.
- HTML is lightweight code so that fast to download.
- HTML is very easy to edit in any text editor such as Notepad or Notepad ++.
- HTML is the most search-friendly programming on the globe.

2.5.2.2 Disadvantages of HTML

- HTML script is easy to write while it can take much time to write a single webpage,
- HTML is very easy to write as compare to other programming languages while it needs unaccountable words to develop even a single webpage on the internet,
- HTML needs a server for saving a file,
- HTML is easy to write but it is difficult to find an error when a webpage does not open.

2.5.3 CSS

CSS stands for Cascading Style Sheets. It is the advanced form of HTML language and little easier to use than HTML, and it is also compatible with HTML. CSS is used for the presentation of web pages i.e., design, colors, layout and fonts etc. CSS has a special standard of a sector and declaration such that selector, property, declaration block and values. CSS has three ways of inserting style sheets into HTML code i.e., External CSS, Internal CSS and Inline CSS. CSS file extension is file.css.

2.5.3.1 Benefits of CS

- CSS has more attributes than HTML that make a better-looking web page style format than HTML,
- Its faster webpage speeds.

- It has less lightweight code that makes webpage takes less downloading time 2 It makes updates, modification and maintenance a lot easier,
- It is compatible with multiple devices as well as platform environment,
- CSS is extremely used in E-commerce, social media and web-based online services,
- It is easy to handle image files by CSS. Files such as jpeg, png and gif can be easily modified by using the CSS frameworks,
- It is easy to handle to Dynamic Website Templates with the help of CSS frameworks,
- It is easy to handle Flash Animation and Effects in web pages with the help of CSS Flash files such as button effects, loaders and animated background,
- It is used for the client-side and server-side styling display which gives better display in the websites,
- CSS has three types of inserting styles i.e., Internal CSS, External CSS and Inline CSS. Internal CSS is a ruleset which needs to add section of the HTML and it is mainly used a single page. External CSS which links to an external .CSS file extension. It is extremely useful for creating a large webpage. Inline CSS means adding CSS into the HTML file and located in the head of the document between style tags.

2.5.3.2 Disadvantages of CSS

- CSS have many types like CSS, CSS 1 and CSS 3. These different kinds of CSSs create confusion among developers and web browsers in different platforms,
- CSS web pages is not compatible on all browsers,
- CSS is an open text-based code which does not have built-in security. It means anyone can read and write the CSS file and change the website contents which can cause the major issues in the websites.

2.5.4 MongoDB

- MongoDB is a cross-platform document-oriented NoSQL database used for high volume data storage that provides high performance, high availability and easy scalability.
- MongoDB stores data in flexible, JSON-like documents, meaning fields can vary from document to document and data structure can be changed over time. The document model maps to the objects in the application code, making data easy to work with.
- The data model available within MongoDB allows users to represent hierarchical relationships, to store arrays, and other more complex structures more easily.
- MongoDB works on concept of collections and documents. Each database contains collections which in turn contains documents. Each document can have varying number of fields. The size and content of each document can also be different from each other.

2.5.4.1 Key Components of MongoDB Architecture

- **_id** – This is a 24-digit unique identifier field required in every MongoDB document in the collection. The **_id** field is like the document's primary key. If the user creates a new document without an **_id** field, MongoDB will automatically create the field.
- **Collection** - Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Typically, all documents in a collection are of similar or related purpose.
- **Document** - A document is a set of key-value pairs. Documents have dynamic schema. Dynamic schema means that documents in the same collection do not need to have the same set of fields or structure, and common fields in a collection's documents may hold different types of data.
- **Database** - Database is a physical container for collections. Each database gets its own set of files on the file system. A single MongoDB server typically has multiple databases.
- **Field** - A name-value pair in a document. A document has zero or more fields. Fields are analogous to columns in relational databases.
-

2.5.5 Express.js

- Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications. It is an open-source framework developed and maintained by the Node.js foundation.
- Express provides us the tools that are required to build our app, be it single-page, multi-page or hybrid web applications. It is flexible as there are numerous modules available on npm(Node Package Manager), which can be directly plugged into Express
- Unlike its competitors like Rails and Django, which have an opinionated way of building applications, Express has no "best way" to do something. It is very flexible and pluggable.
- Pug (earlier known as Jade) is a terse language for writing HTML templates. It produces HTML, supports dynamic code and code reusability (DRY). It is one of the most popular template languages used with Express.
- Express can be thought of as a layer built on the top of the Node.js that helps manage a server and routes. It allows users to setup middleware to respond to HTTP Requests and defines a routing table which is used to perform different actions based on HTTP method and URL.
- Express allows to dynamically rendering HTML Pages based on passing arguments to templates.
- Express is asynchronous and single threaded and performs I/O operations quickly.
- Ultra-fast I/O.
- Asynchronous and single threaded.
- MVC like structure.
- Robust API makes routing easy.

2.5.6 React

- ReactJS is a declarative, efficient, and flexible JavaScript library for building reusable UI components. It is an open-source, component-based front-end library which is responsible only for the view layer of the application. It was initially developed and maintained by Facebook and later used in its products like WhatsApp & Instagram.
- A ReactJS application is made up of multiple components, each component responsible for outputting a small, reusable piece of HTML code. The components are the heart of all React applications. These Components can be nested with other components to allow complex applications to be built of simple building blocks. ReactJS uses virtual DOM based mechanism to fill data in HTML DOM. The virtual DOM works fast as it only changes individual DOM elements instead of reloading complete DOM every time.
- Instead of using regular JavaScript, react codes are written in something called JSX (JavaScript Syntax Extension). JSX is basically a syntax extension of regular JavaScript and is used to create React elements. These elements are then rendered to the React DOM. JSX is faster than normal JavaScript as it performs optimizations while translating to regular JavaScript.

2.5.7 Advantage of React

- Uses virtual DOM which is a JavaScript object. This will improve apps performance, since JavaScript virtual DOM is faster than the regular DOM.
- Can be used on client and server side as well as with other frameworks.
- Component and data patterns improve readability, which helps to maintain larger apps.

2.5.8 Key Features of React JS

The React JS offers plenty of benefits but below there are some key benefits.

- **Easy to learn:** It is easy to learn as compare to other front-end frame works technologies.
- **JSX:** JSX is shorthand of JavaScript XML. React JS uses JSX instead of HTML which enables the user to read and write the components easily and clearly.
- **Speed:** It allows the developers reuse the individual components on both client side and server-side without effects on the main application.
- **Flexibility:** React JS code is easy to maintain and flexible due to its framework structure as compared to other front-end framework technologies.
- **Performance:** React JS uses virtual DOM and server-side rendring thus it makes complex web apps runs on high speed.
- **Code stability:** It follows unidirectional data flow or downward data flow which means parent structure unaffected by any modifications in its child structure.
- **Integration:** React code easy to integrate with other frameworks like Angular.
- **SEO- friendly:** SEO stands for Search Engine Optimization. React offers less page load time and fast rendering speed

- It is easy to test UI cases with React code.
- React renders at client side while SEOs do not use the data.

2.5.9 Life Cycle of React Components

In life, we humans as well as tree have a life cycle like we are born, grow and finally we die. It is the same phenomena in software application's life cycle, which takes birth as initialization, it grows by updating and changes occurs during the different stages by mounting and unmounting. React JS has four life cycle classifications as below.

- **Initialization:** This is first and most important phase in which developer setup or constructs the component with the given props (properties) and default state.
- **Mounting:** In this second phase, after creating state and props by developer. React JS component is ready to mount in the browser DOM.
- **Update:** This is third stage, once the component gets added in browser DOM. React JS components updates it when a state or prop changes in the application.
- **Unmounting:** This is fourth and final phase of the life cycle of React JS component in which the component is removed from browser DOM when it is not needed anymore.

2.5.10 User Interface

User Interface (UI) is the interaction and communication between human and any device. Some common examples are

- Command Line User Interfaces (CLI)
- Graphical User Interfaces (GUI)
- Touch screen Interfaces
- Menus Interfaces
- Voice User Interfaces (VUI)
- Form-fill Interfaces
- Natural-Language Interfaces

2.5.11 Flux

Flux is an architectural pattern which was developed by Facebook to work with React in 2011. FLUX's application architecture is more suitable for building user interface application also client-side web-based applications.

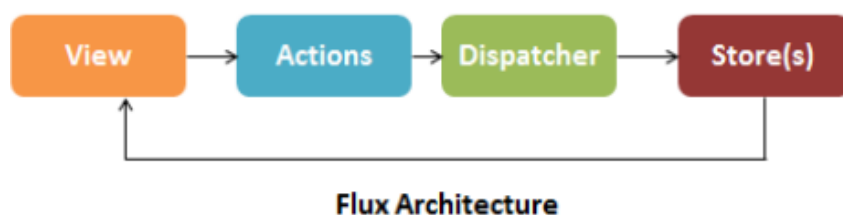


Figure 2.4. Flux Architecture

2.5.12 Features of Flux

Flux's architectural pattern is based on four segments in unidirectional data flow. Its values and objects can be mutated or changeable without creating any new values and objects. It is

Integrated with Tuxedo JS, Flux xor and React as shown in Figure.

Flux's four segments are as following:

- **Actions:** Action is a simple JavaScript object in Flux pattern which must pass through dispatcher.
- **2 Dispatcher:** Dispatcher is the central hub of the application in Flux pattern which dispatches all the data and sends it to the stores.
- **Stores:** Flux pattern has multiple stores for each application while each store is a singleton type object.
- **View:** View is the element of the Flux pattern which sends the data from the store to the application.

2.5.13 Node.js

- Node.js is a very powerful JavaScript-based platform built on Google Chrome's JavaScript V8 Engine. It is used to develop I/O intensive web applications like video streaming sites, single page applications, and other web applications. Node.js is open source, completely free, and used by thousands of developers around the world.
- Node.js is a server-side platform built on Google Chrome's JavaScript Engine (V8 Engine). Node.js was developed by Ryan Dahl in 2009.
- Node.js applications are written in JavaScript and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.
- Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.

2.5.13.1 Features of Node.js

- **Extremely fast:** Node.js is built on Google Chrome's V8 JavaScript Engine, so its library is very fast in code execution.
- **I/O is Asynchronous and Event Driven:** All APIs of Node.js libraries 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.
- **Single threaded:** Node.js follows a single threaded model with event looping.
- **Highly Scalable:** Node.js is highly scalable because event mechanism helps the server to respond in a non-blocking way.

- **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.
- **Open source:** Node.js has an open-source community which has produced many excellent modules to add additional capabilities to Node.js application.

2.5.11 Styled-Components

Styled-components are the result of wondering how we could enhance CSS for styling React component systems. By focusing on a single use case, we managed to optimize the experience for developers as well as the output for end users.

Apart from the improved experience for developers, styled-components provides:

- **Automatic critical CSS:** styled-components keeps track of which components are rendered on a page and injects their styles and nothing else, fully automatically. Combined with code splitting, this means your users load the least amount of code necessary.
- **No class name bugs:** styled-components generates unique class names for your styles. You never have to worry about duplication, overlap or misspellings.
- **Easier deletion of CSS:** it can be hard to know whether a class name is used somewhere in your codebase. styled-components makes it obvious, as every bit of styling is tied to a specific component. If the component is unused (which tooling can detect) and gets deleted, all its styles get deleted with it.
- **Simple dynamic styling:** Adapting the styling of a component based on its props or a global theme is simple and intuitive without having to manually manage dozens of classes.
- **Painless maintenance:** You never have to hunt across different files to find the styling affecting your component, so maintenance is a piece of cake no matter how big your codebase is.
- **Automatic vendor prefixing:** Write your CSS to the current standard and let styled-components handle the rest.

You get all of these benefits while still writing the CSS you know and love, just bound to individual components

2.5.11.1 Installation Styled-Component

Installing styled-components only takes a single command and you're ready to roll:

with npm

npm install --save styled-components

with yarn

Yarn add styled-components

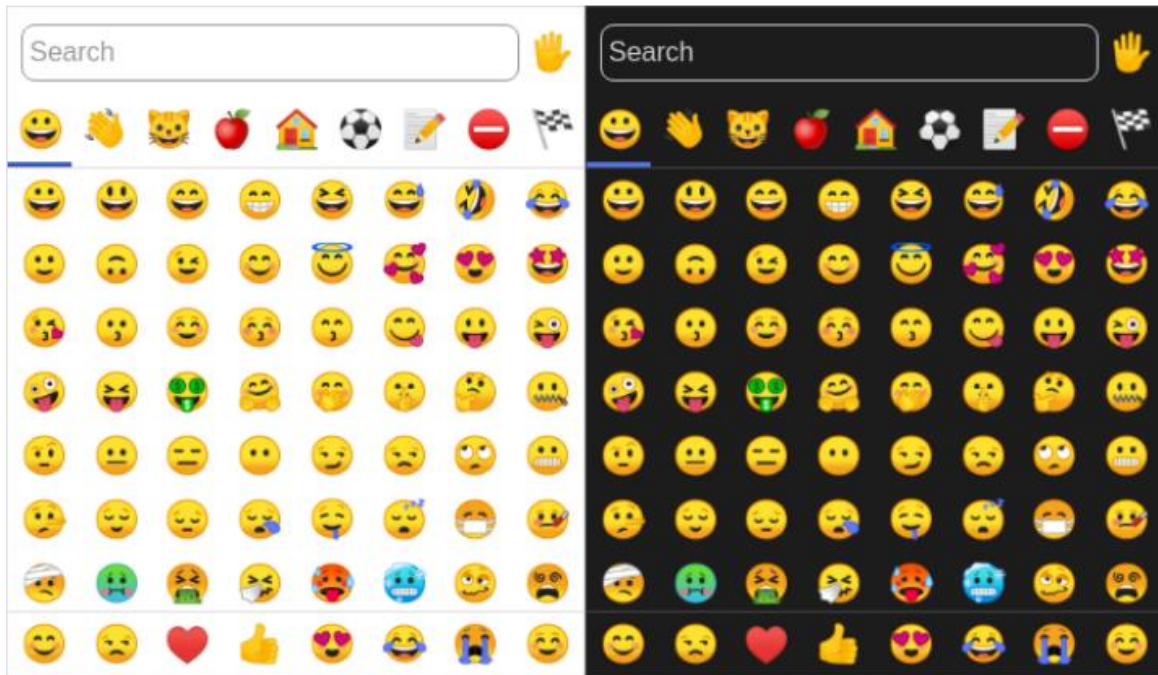


Figure 3.4. Emoji Picker

2.5.12 Dependencies

Emoji-picker: emoji-picker-element is a lightweight emoji picker for the modern web which gives you the access of lots of emoji in your react project. Emoji pickers are ubiquitous. It seems that every social media and messaging app needs to have a little grid of cartoon faces you can click on.

Installation: *npm install emoji-picker-element*

Babel: Babel is a free and open-source JavaScript trans-compiler that is mainly used to convert ECMA Script 2015+ code into a backwards compatible version of JavaScript that can be run by older JavaScript engines. Babel is a popular tool for using the newest features of the JavaScript programming language.

Installation: *npm install*

"babel-preset-node8": "^1.2.0",

"body-parser

"compression"

"cors

"express

"moment

"mongoose

"yup

Body-parser: Body-parser is the Node.js body parsing middleware. It is responsible for parsing the incoming request bodies in a middleware before you handle it.

Installation: *npm install body-parser*

2.5.13 Visual Studio Code

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages (such as C++, C#, Java, Python, PHP, Go) and runtimes (such as .NET and Unity).

Visual Studio Code is a code editor in layman's terms. Visual Studio Code is “a free-editor that helps the programmer write code, helps in debugging and corrects the code using the intelli-sense method”. In normal terms, it facilitates users to write the code in an easy manner. Many people say that it is half of an IDE and an editor, but the decision is up to the coders. Any program/software that we see or use works on the code that runs in the background. Traditionally coding was used to do in the traditional editors or even in the basic editors like notepad! These editors used to provide basic support to the coders.

Some of them were so basic that it was very difficult in writing basic English level programs in them. As time went by, some programming languages needed a specific framework and support for further coding and development it, which was not possible using these editors. VI Editor, Sublime Text Editor, is one of the many kinds of editors that came into existence. The most prominent and which supports almost every coding language is VISUAL STUDIO CODE. Its features let the user modify the editor as per the usage, which means the user is able to download the libraries from the internet and integrate it with the code as per his requirements.

2.5.13.1 Features

Visual Studio Code has some very unique features. They are listed as below:

- **Support for multiple programming languages:** Supports multiple programming languages. So earlier, programmers needed Web-Support: a different editor for different languages, but it has built-in multi-language support. This also means it easily detects if there's any fault or cross-language reference, it'll be able to detect it easily.
- **IntelliJ-Sense:** It can detect if any snippet of code is left incomplete. Also, common variable syntaxes and variable declarations are made automatically. Ex: If a certain variable is being used in the program and the user has forgotten to declare, IntelliJ-sense will declare it for the user.
- **Cross-Platform Support:** Traditionally, editors used to support either Windows or Linux or Mac Systems. But Visual Studio Code is cross-platform. So it can work on all three platforms. Also, the code works on all three platforms; else, the open-source and proprietary software codes used to be different.
- **Extensions and Support:** Usually supports all the programming languages but, if the user/programmer wants to use the programming language which is not supported then, he can download the extension and use it. And performance-wise, the extension doesn't slow down the editor as it runs as a different process.

- **Repository:** With the ever-increasing demand for the code, secure and timely storage is equally important. It is connected with Git or can be connected with any other repository for pulling or saving the instances.
- **Web-Support:** Comes with built-in support for Web applications. So, web applications can be built and supported in VSC.
- **Hierarchy Structure:** The code files are located in files and folders. The required code files also have some files, which may be required for other complex projects. These files can be deleted as per convenience.
- **Improving Code:** Some code snippets can be declared a bit differently, which might help the user in the code. This function prompts the user, wherever necessary, to change it to the suggested option.
- **Terminal Support:** Many of the times, the user needs to start from the root of the directory to start with a particular action, in-built terminal or console provides user support to not to switch in-between two screens for the same.
- **Multi-Projects:** Multiple projects containing multiple files/folders can be opened simultaneously. These projects/folders might or might not be related to each other.
- **Git Support:** Resources can be pulled from Git Hub Repo online and vice-versa; saving can be done too. Resource pulling also means cloning the code which is made available on the internet. This code can later be changed and saved.
- **Commenting:** A common feature, but some of the languages do not support it. Commenting on the code helps the user to recall or track according to the sequence he wants.

2.5.13.2 Advantages

- There are many advantages over any other IDE; they are as follow:
- Cross-platform support:
- Windows
- Linux
- Mac
- Light-weight
- Robust Architecture
- IntelliJ-Sense
- Freeware: Free of Cost- probably the best feature of all for all the programmers out there, even more for the organizations.
- Many users will use it or might have used it for desktop applications only, but it also provides great tool support for Web Technologies like; HTML, CSS, JSON.

2.5.13.3 Disadvantages

There are not many issues with Visual Studio Code but one area that can be definitely improved is high memory and CPU consumption. Being an electron app, Visual Studio Code consumes very high memory and CPU.

CHAPTER - 3

REQUIREMENT ANALYSIS

CHAPTER - 3

REQUIREMENT ANALYSIS

4.1 Project Aims

There is no better way to understand a framework, library and its features, than by making projects.

- To create the Web Application Just like WhatsApp.
- To create chat room or groups between two or many users.
- Sign in with Google account.
- To create the Messaging Box and Contact List.
- Sent Message in groups and to individual person.
- To design the UI like the original WhatsApp web have.
- To learn & Implementation of basic and advanced library of React JS and MangoDB. And other technology.
- Understanding & Implementation the concept of messaging application.
- Understanding & Implementation the concept of modern application and their working process
- Learning & Implementation the Communication process with the server and user.
- Learning & Implementation the internet protocol and their uses.

4.2 Expected outcomes for this project are

- Sign In with Google Account.
- Profile Picture of Sender and Receiver should be visible.
- The contact list should be visible on left side with recent chat.
- It should be able to display data from database.
- It should be able to add rooms with a prompt to the database.
- The chat box should take the input and display it as chat in real time.

4.3 Software and Hardware Requirements

4.3.1 Software Requirements

- **Platform:** Web platform
- **Language:** Html, Java Script, CSS
- **IDE:** Visual Studio Code
- **Database:** Mongoo DB
- **Operating System:** Window/IOS/Linux

4.3.2 Hardware Requirements

- **CPU Cores:** quad-core CPU
- **RAM:** 8GB

4.4 Additional Requirements

- HTML, CSS (Specially Flex-Box)
- Java Script
- Any Web Browser
- A Stable Internet Connection
- Use of Github as industry standard
- Knowledge of Basic Programming and Problem Solving
- Visual Studio IDE
- Node Js
- React (Component, JSX, State, Props etc...)
- The Resource to learn React Hooks, Metoer, and Styled Components

4.5 Cost Estimation of Project

Building an application across platforms could attract a lot of investment, and bulk investment results in a higher amount of risks. This problem can be brought into control by either selecting a single platform to build an app or then after plowing back profits into building another app for different platforms.

Understanding the requirements could sum up different factors, and these factors play a very specific role in crafting an application. An application cost has a simple calculation, i.e.

$$\text{App Cost} = \text{Development Time} * \text{Hourly Rate}$$

A simple calculation structure with detailed factors that are described below.

4.5.1 Online OR Offline Application

Internet-dependent application attracts a lot cost in an earlier time as compared to no internet-dependent application. Also, in an earlier time, these internet-dependent applications were the ones that require a lot of costs but with backend technologies like Firebase, the app design and development have become affordable.

4.5.2 Number of Features

The beauty of the application is dependent on the features you add in it, and the more features you add, the more investment you have to bring in, instead what you can do is, bring the updates in the version format and slowly increase the ROI.

4.5.3 Number of Screens

The number of screens decides the investment in terms of time and money, as the number of screens could attract a higher monetary investment. It is advisable to use fewer screens and feed in more information that could save a lot of time and money.

4.5.4 The complexity of Backend

Backend plays an important role in any application as it stores data, manages information and credentials, etc. The front-end sometimes looks simple & easy but the backend becomes complex and that takes out a lot of investment which in turn becomes costly for an app owner.

4.5.5 UI, Animation, Special effects

The application that has great animations, gestures and special effects in it attracts higher investment and thus, the gaming apps are the costliest one while building.

Most of the technology stack that is used in the project is open-sourced. And as open-source applications do not cost any fortune and are freely available to the public, thus, there would be no cost included along with the development of the project.

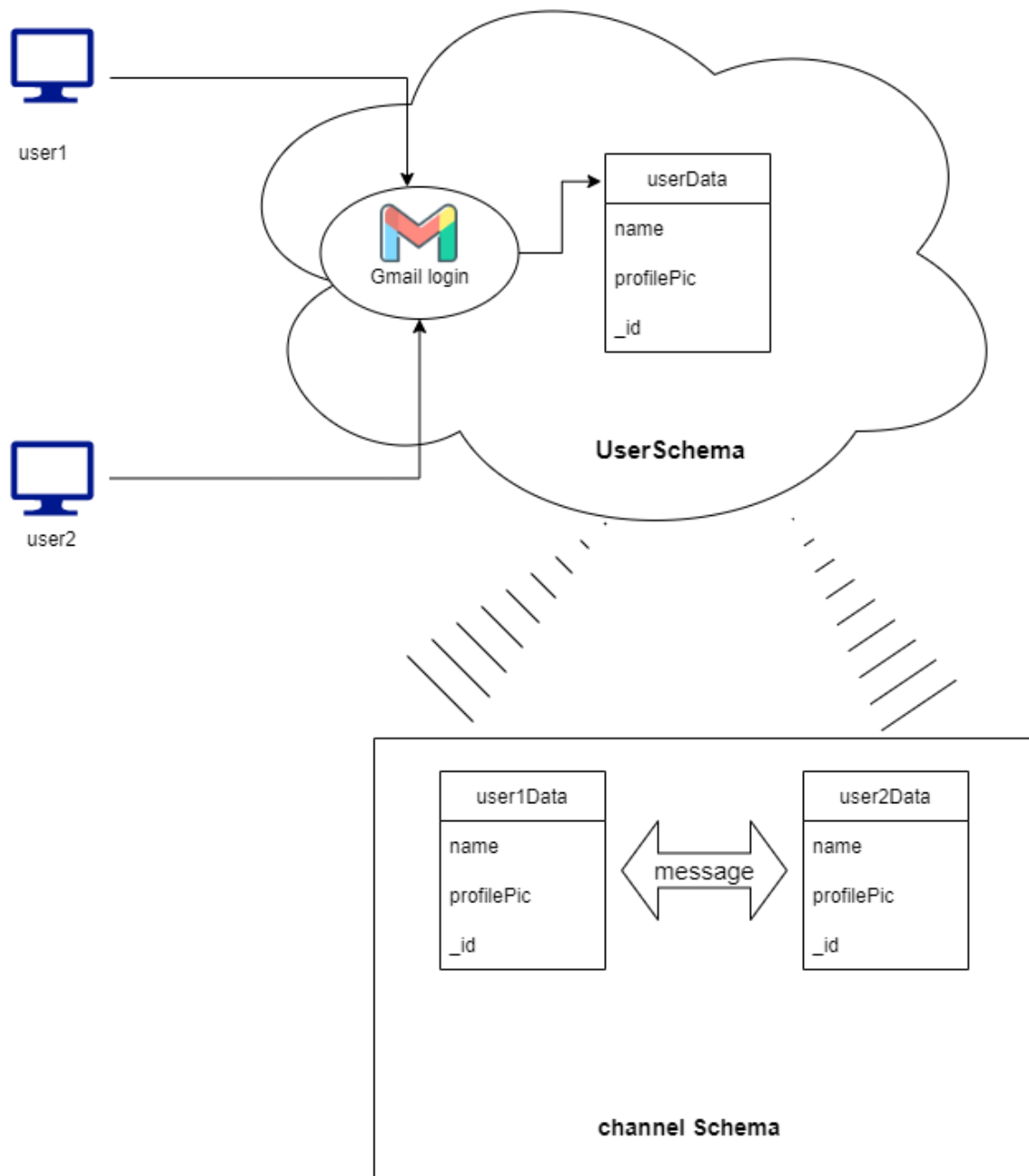
CHAPTER – 4

PROJECT DESIGN

CHAPTER – 4

PROJECT DESIGN

Software design is basically a mechanism of preparing a plan, a layout for structuring the code of your software application. It is as simple as that to define, but is this that easy to perform. Well, that is a whole different scenario but if you are to perform software design for your new software project then it is a necessity to know its importance, otherwise, you might just miss the crisp of it.



Software design is a multi-stage concept. The software itself is a multi-layer, multi-dimensional spectrum, and its design has multiple intermediate steps, therefore; different types of software level design.

4.1 Types of Software Level Design

- **Architectural Design:** Architectural design is the first step in software-level design. This involves the construction of a software application structure in order to have the correct interconnection of each element of code according to the required function.
- **High-level Design:** This is the second step in software-level design. In this step, the designer splits the theoretical concepts, into multiple entities and ensures their functions are inter-related to get an optimum result. This module identifies a modular structure of different entities and creates a design to inter-connect all modules for a specific output.
- **Detailed Design:** This is the third and last step of the software-level design. It involves the compilation of all modular outputs and arrangement or rearrangement of functional modules for the final outcome. This level of software design determines more of a logical structure out of constructed modules. The software outcome accomplishment part is the motive of the detailed design level of software design. What Is The Importance Of Software Design.

The functionality of the chat application is to give the ability to chat with whoever is online on the application. The users and stakeholders will be a small group for now, the use cases will be what is available to the user, and the functional/nonfunctional requirements will be covered, as well as the milestones of the chat application.

4.2 Users and Stakeholders

This section will deal with the users and stakeholders. The users will be using the chat application and the stakeholders will develop, maintain, and test the chat application.

Team and I	We will be developing, maintaining, and testing the chat application through its phases of development.
Users	The users will be anyone who has the chat application and registers for it.

Table 4.1: *Users and Stakeholder*

4.3 Project Perspective

- The system to be developed here is a Chat facility. It is a centralized system. It is Client-Server system with centralized database server. All local clients are connected to the centralized server via LAN.
- There is a two-way communication between different clients and server. This chat application can be used for group discussion. It allows users to find other logged in users.

4.5 Interface

- This application interacts with the user through G.U.I. The interface is simple, easy to handle and self-explanatory.
- Once opened, user will easily come into the flow with the application and easily uses all interfaces properly. However, the basic interface available in our application(fig.2) is: -
- Header with User Profile Picture.
- Search bar for Contact Search.
- Contact list panel.
- Conversations panel.
- Message Typing Dialog box with emoji features.

4.6 Functional and Non-Functional Requirements

4.6.1 Functional Requirements

User Registration: User must be able to register for the application through an Email, Username and Password. On Opening the application, user must be able to register them or they can directly login if there have an account already. If user skips this step, user should able to chat. The user's email will be the unique identifier of his/her account on Chat Application.

Adding New Contacts: The application should detect all contacts from the server database. If any of the contacts have user entered with Chat Application, those contacts must automatically be added to the users contact list on Chat Application.

Send Message: User should be able to send instant message to any contact on his/her Chat Application contact list. User should be notified when message is successfully delivered to the recipient by coloring message.

Broadcast Message: User should be able to create groups of contacts. User should be able to broadcast messages to these groups.

Privacy: Messages shared between users should be encrypted to maintain privacy.

Robustness: In case user's system crashes, a backup of their chat history must be stored.

Performance: Application must be lightweight and must send messages instantly.

4.7 Use Case Table

<i>Level 0</i>	<i>Level 1</i>	<i>Level 2</i>	<i>Actor</i>
<i>Chat Application</i>	Authentication System	Registrar, Login, Logout	User and Admin
	Contact Form	Search/Find Users, Adding Users	User
	Chat Form	Send Message	User
	Monitor	Chat History	User and Admin

• *Table 4.2: Use Case*

CHAPTER - 5

IMPLEMENTATION OF MODULES

CHAPTER – 5

IMPLEMENTATION OF MODULES

The implementations of the Chat App are discussed with the codes in this chapter.

5.1 Installing Visual Studio Code

The IDE stands for Integrated Development Environment. It is used for the text editor for building software applications, compiler and debugger with help of various integrated tools. In other words, IDE is a software tool which can perform multi functions such as to write code, modify code, compile and debug code etc.

Now, there are many IDEs available in the market which are used for the different performs. Most important, Visual Studio Code (VSC) is the best choice for Chat App. It is free and open-source and it works perfectly on cross-platforms. It is recommended to install visual studio code from its official website.

5.2 Environment Setup

The environment setup of WhatsApp clone requires the installation npm first.

5.2.1 Installation of npm

NodeJS is JavaScript runtime open-source environment which helps developers to build web applications with the help of frameworks like web pages, images, video clips, social network sites and application forms. It is based on Chrome's V8 engine which runs on cross-platform such as windows, Linux and Mac. The V8 engine compiles the JavaScript code into machine code that makes our program code runs smoother and faster.

NodeJS is an open source library which is currently used by unaccountable developers around the globe. NodeJS uses a special programming technique which is called asynchronous programming. Asynchronous programming means that developers can execute multiple codes at a time without finishing the current execution can execute the parallel code. NodeJS offers popular services like Netflix, PayPal, Uber and eBay.

Npm stands for Node Package Managers. It is the world's largest open source library which is used for Nodejs. It was developed by Rebecca Turner, Kat Marchan in 2010. It is written in JavaScript itself and free for use.

It is highly recommended to download NodeJS directly from its official website i.e. <http://nodejs.org>. NPM comes with the installation of NodeJS.

```
C:\Users\akshay>node -v
```

```
v16.14.2
```

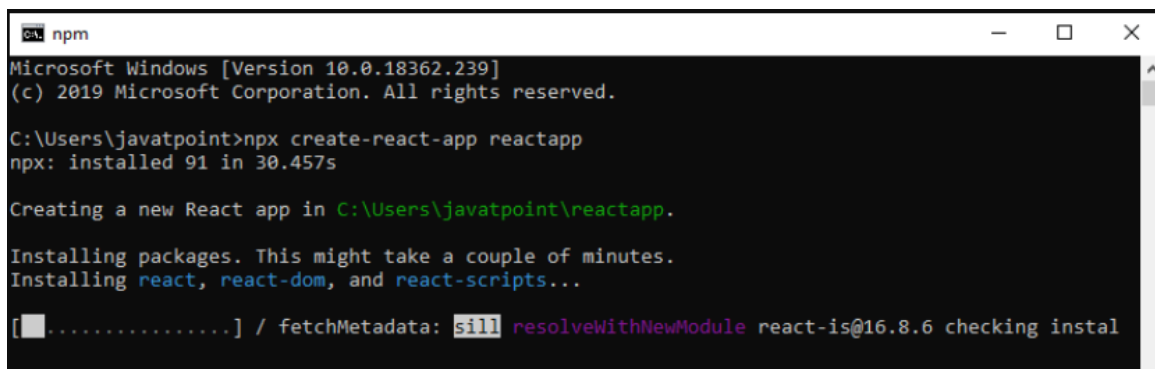
```
C:\Users\akshay>npm -v
```

```
8.6.0
```

The above commands show that npm and node is installed with the latest version.

5.2.2 Create React App

The official way to create an app in React JS is “create-react-app”, which was first deployed in Facebook for testing the real-time chat of a user. It helps to save a lot of time, so we do not need to install or configure extra tools like, for example, web packs, Babel and other dependencies.



```
npm
Microsoft Windows [Version 10.0.18362.239]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\javatpoint>npx create-react-app reactapp
npx: installed 91 in 30.457s

Creating a new React app in C:\Users\javatpoint\reactapp.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts...

[█.....] / fetchMetadata: sill resolveWithNewModule react-is@16.8.6 checking instal
```

Figure 5.1 npm start commande

A simple command is only needed to install either in command prompt or power shell in windows.

- *npm install -g create-react-app (Chat-app) Create a project name then install a new app inside the project*
- *D:\chatapp\chat-app>mkdir chat-app*
- *Installing packages of dependencies such as Installing react, react-dom, and react-scripts. Wait for the couple of minutes.*

cd chat-app

npm start

Happy hacking! The Chat App file structure in VS code created by create-react-app.

```
Command Prompt
added 1385 packages from 675 contributors and audited 902050 packages in 305.218s
found 0 vulnerabilities

Success! Created reactapp at C:\Users\javatpoint\reactapp
Inside that directory, you can run several commands:

  npm start
    Starts the development server.

  npm run build
    Bundles the app into static files for production.

  npm test
    Starts the test runner.

  npm run eject
    Removes this tool and copies build dependencies, configuration files
    and scripts into the app directory. If you do this, you can't go back!

We suggest that you begin by typing:

  cd reactapp
  npm start

Happy hacking!
C:\Users\javatpoint>
```

Figure 5.2 happy hacking – react app installed

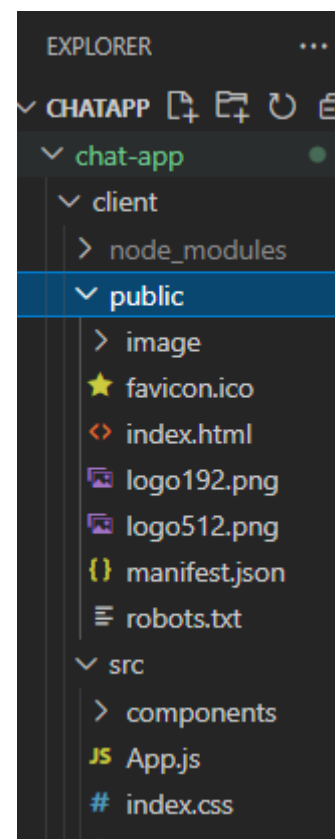
5.2.3 User Story Case

The user story of the Chat App explains the file structure of the application and how the user can use the application.

5.2.3.1 Project Structure

In this section, the deployment of this project is discussed. The whole project structure and different components of the project will be explained in Visual Studio Code. All component files of the project are in the source (.src) folder on the left hand.

In source (.src) folder, App.js is the main app file of the project, SignUp.js and LoginForm.js component for creating a user in chat application and SendMessageForm.js component is for writing the message to another user, which is in the main panel on right side. While, the SideBar.js component contains Logout button and during a chat it enables creating a new chat Room. Fire.js component is for backend and hosting service of a user's chat and firebase. json contains the json file of the backend hosting service of firebase while the firebaseconfig file shows the default chat project.



5.1 Send Chat

This topic discusses how users send chat messages in WhatsApp Clone. React Map is a JavaScript library function, the map method represented as `.map()` method creates a new array by calling a provided function on every message Key in project. The Key is a special string attribute which is required to include creating lists of messages in a project. The important feature in JSX is the spread operator attribute. The object such as message's property is provided by JSX spread operator attributes in the project which is represented by ... (three dots).

```
const Message = ({message}) => {  
  return(  
    {message.email} @ {moment(message.created).format('h:mm a')}  
  )  
}  
  
const MessageList = ({ messages }) => (  
  {Object.keys(messages)  
    .map(messageKey => ({...messages[messageKey], id: messageKey}))  
    .map(message => <Message key={message.id} message={message}/>)  
  }  
)
```

Figure 5.3 Screenshot of MessageList

In order to reach this goal, two constructors are built namely as `const Message List` and `const Message` as in the above screenshot in Figure 5.1 of Chat App also on Change (e) event argument is modified in the `render ()` function of Send Message Form. At first, a constructor is built as Message List, in which message is sent by a user as an object like a Key string attribute value and applying `map ()` method on every message Key then spread Key string attribute(...) is applied on every message Key and id. Again, `map()` method is applied over the messages such as id of the message and message text.

Another constructor is built as Message by which message is returned with email id, what time message is created with the proper time format. Time format is accomplished with `moment.js`, which must be installed first in npm.

Now, on Change (e) the event handler is modified in Send Message Form. Whenever the change in input field as text is changed, the event of target value is changed which is updated the state by this. Set State as shown in the below screenshot in Figure 5.3.

```

render() {
  return(
    <form onSubmit={this.onMessageSend}>
      <div className='control'>
        <input type='text'
          value={this.state.text}
          onChange={(e)
            =>this.setState({text:

```

Figure 5.4 Screenshot of on Change function

Finally, on Message Send (e) an event handler is built, which is prevented from refreshing the whole browser by simply pressing Send Message From.

Const message is created as a constructor, showing the user's message created time, text state, author property of uid, the room Id where chat is occurred, the property of email by this function, the property of send Message as message and text is updated by set State as shown in the screenshot in Figure 5.5.

```

onMessageSend= (e) =>
{
  e.preventDefault();
  const message = {created: Date.now(),
    text: this.state.text, author: this.props.uid, roomId: this.props.roomId, email: this.pr
    ops.email}this.props.sendMessage(message); this.setState({text: ""});

```

Figure 5.5 Screenshot of on Message Send function

5.2 Receive Chat

This section explains how the user's chat is refreshed automatically. In order to achieve this purpose, Firebase's listener method "on" and "child_added, orderByChildandkey" methods are reused. These methods are implemented and only adding some conditions in the selected room is used in the project.

```

messageRef
    .on("child_added", snapshot=>{

        const message=snapshot.val();

        const key=snapshot.key;

        if(message.roomId===this.state.selectedRoom){ this.setState({

            messages:{

                ...this.state.messages,

                [key]: message }

```

Figure 5.4. Screenshot of snapshot setRoom

The first step is: setRoom (id), .orderByChild which is roomId, .equalTo (id) for the room,. Once('value')for onetime message,.then(snapshot)is a snapshot of the value or message is taken an empty {} object means that a value can be nothing, state of the selectedRoom and corresponding messages is updated by this. SetState while .catch(err => catches error in the browser console as shown in Figure 5.6

```

return messageRef

    .orderByChild('roomId')

    .equalTo(selectedRoom)

    .once('value');

```

Figure 5.5. Screenshot of message Ref

The second step is configuring the message Ref. It is accomplished by adding Firebase's "on" listener, which takes child_added method and snapshot, snapshot of val() is taken by message constructor while a snapshot of the key can be a value or a message of the user which is taken by the key method constructor. Message Ref is updated by set State, message state and key such as message.

The third step, return message, on refresh is promised by Firebase backend as below in Figure 5.6.

```
setRoom=(id) => { messageRef
  .orderByChild('roomId')
  .equalTo(id) .once('value')
  .then(snapshot => { const messages = snapshot.val() || {};
    this.setState({
      selectedRoom: id, messages });}) .catch(err => console.error(err));}
```

Figure 5.6 Screenshot of Firebase backend promise

The final screen shot of the sender's and the receiver's chat message with user email and time stamp is shown in Figure 5.7.

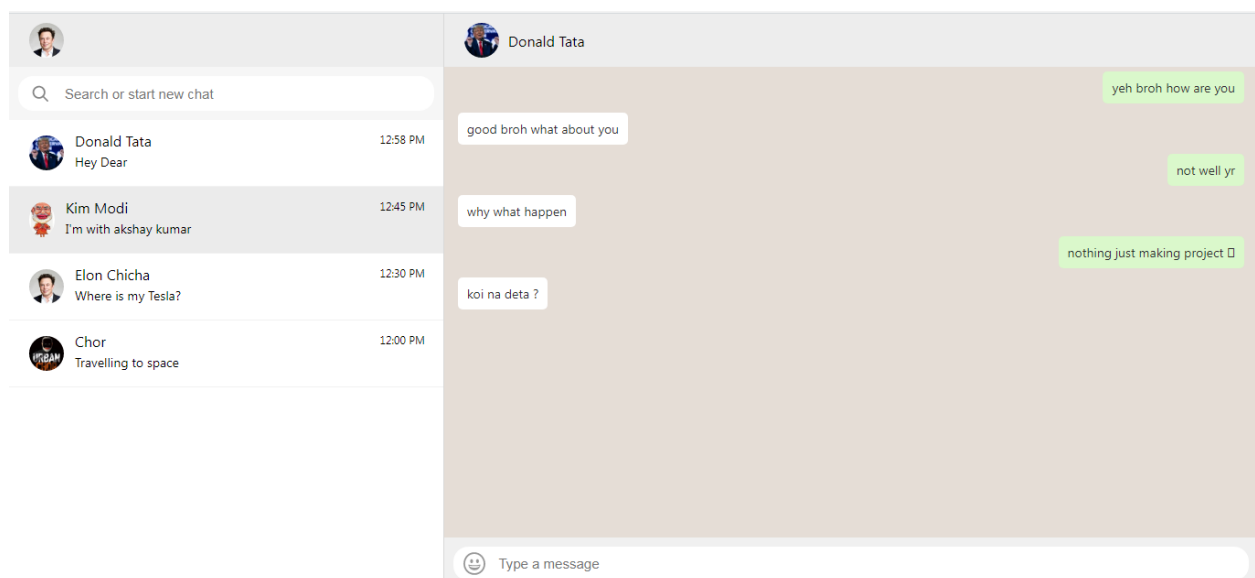


Figure 5.7. Screenshot of Receiver chat message

In this application we have separately designed API'S for every component are listed below:

- **Login API:** Login API helps us to log in this application with the help of email validation dependencies which helps us to validate the emails so that the user can be unique and identified in data base
- **Create User API :** Create User is API that helps to create user and store the data of user in our database. Which contains name profilePic and email id and create an unique id for specific user.
- **Search-user:** Search user help us to search the user which is logged in our database using email validation.
- **Channel:** This API is create channels between two use so that two user can chat with each othe.

- **channel-list** : Channel list provide us to channel list that has build using channel api and give all the details about channels that has created between users such like username, email, id, profilepic etc.
- **message** : message is API that use channel and send text message between user those who are connected in the channel.
- **Chat App or client part**, which is a Web chat application. Build on React.
- Chat Server Engine or server part, which is a pool of external servers responsible for the chat operation. This is the place where all the chat magic happens.

Both parts contain various components that communicate to each other and bring the chat into action

5.3 Chat App or Client Side

Chat App is the other major part of the chat architecture, the one that users directly interact with. It's split into two separate root components:

- **Chat Client Engine** (fig.3.2) handles all the communication with the Chat Server Engine via its internal components: Chat REST API Client Library and Chat WebSocket Client Library.
- Chat UI displays data to users: **Chat Contact List UI**, **Chat Dialog UI**.

5.4 Component

Components are the building blocks of any React app and a typical React app will have many of these. Simply put, a component is a JavaScript class or function that optionally accepts inputs i.e. properties (props) and returns a React element that describes how a section of the UI (User Interface) should appear.

App.js is the starting point of our React app.

A package.json file (fig.4.2):

- lists the packages your project depends on
- Specifies versions of a package that your project can use.
- Makes your build reproducible, and therefore easier to share with other developers.

A package.json file may look similar to this:

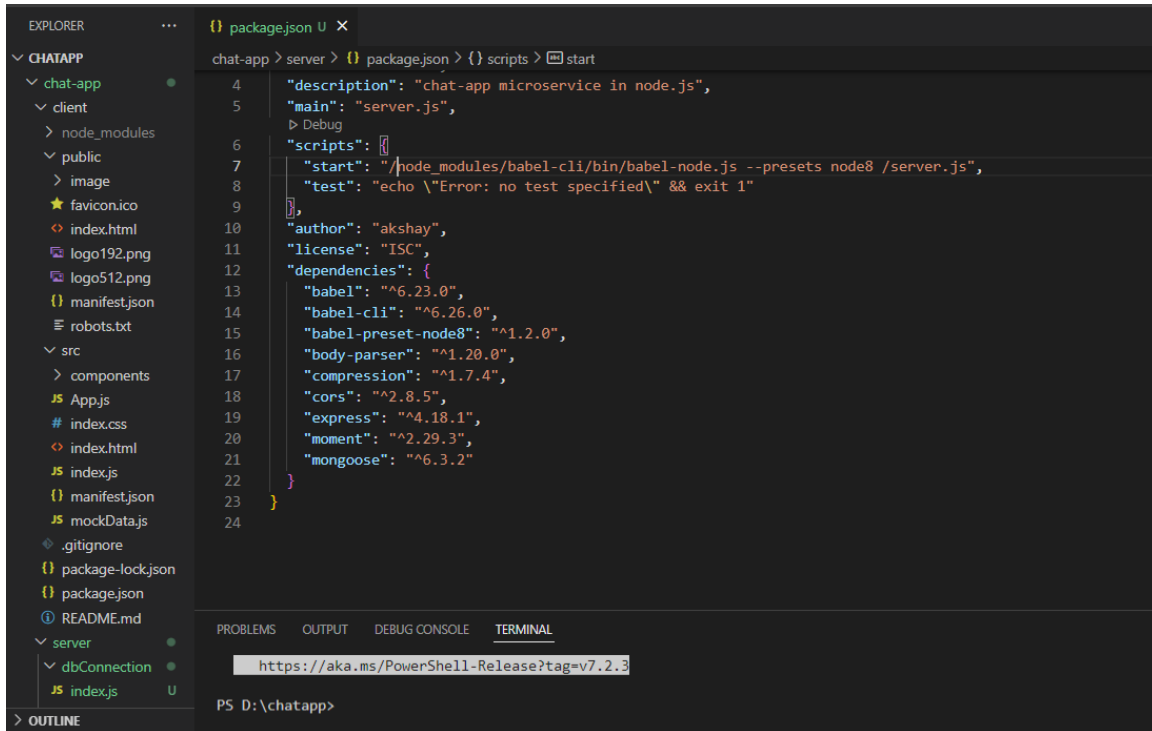


Figure 5.8 package.json

5.5 Chat Server Engine

This is a core of the chat architecture that handles message delivery and dispatch. In our version of chat architecture, it includes the following components.

- Chat REST API handles the tasks that are not connected directly to message dispatch and delivery, such as user authentication, changing of user settings, friendsinvitation, downloading sticker packs, etc. The Chat App (the chat client part) communicates with the Chat REST API via the Chat REST API Client Library.
- Chat Web Socket Server is responsible for transmitting messages between users. The Chat App communicates with the Chat Web Socket Server via the Chat Web Socket Client Library. This connection is open two ways; that means users don't have to make requests to the server if there are any messages for them; they just get them right away

5.6 Dependencies

The third-party package or modules installed using npm are specified in this segment. The package.json file is the heart of Node.js system. It is the manifest file of any Node.js project and contains the metadata of the project. The package.json file is the essential part to understand, learn and work with the Node.js. It is the first step to learn about development in Node.js.

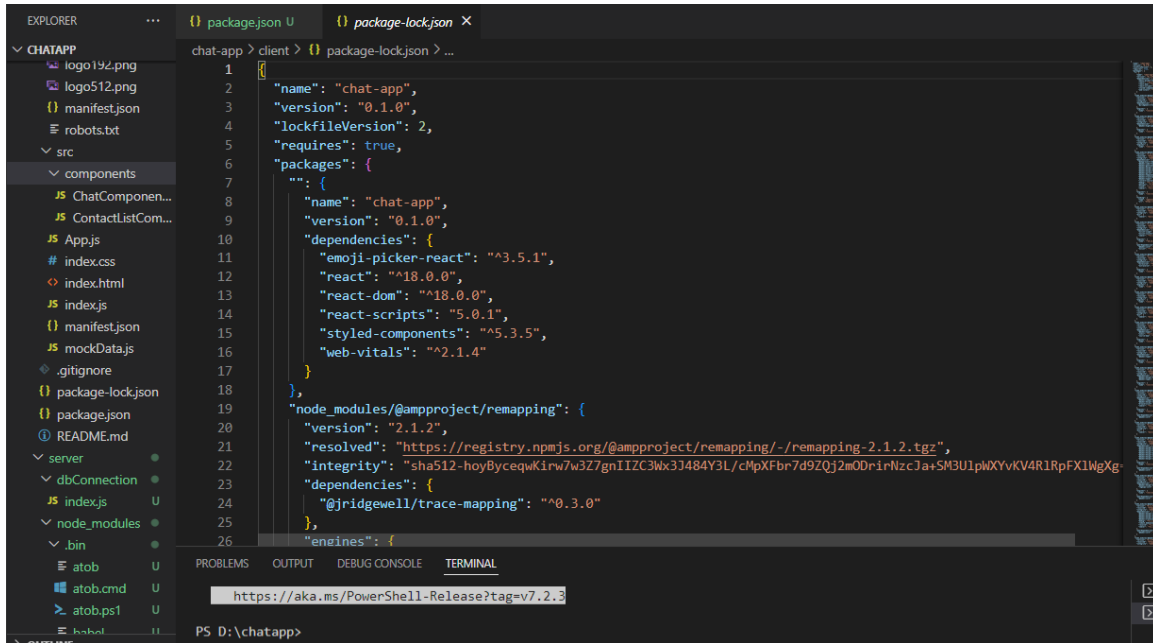


Figure 5.9 *package.json*

Whatsapp clone is a chat-based web application. Once you know the design of Whatsapp application you would design any chat-based application with newer features. The special things about our project are,

- **One to one Chat** — You can make a chat with your friends or any logged in person which is in database.
- **Group Messaging** — Group Messaging is not available in this clone. But we are sure that in future we will try to add this feature.
- The person is online or the person last online start.
- Store profile information about users like status, profile picture, profile ID, and contact details.

5.7 API

- **Profile Service (API)**
Have multiple endpoints which help retrieve details about the user. Get — retrieve profile information, post/put — add/update profile information.
- **Mapping Database**
Before we go to understand deeper, do you have the concept of the general communication architecture? When 2 clients (A and B) want to communicate and send messages to each other, they have to know the address of each other (IP, MAC, or any customized unique identity) and exchange messages over a network, in this case, it is the internet. And they will use the bi-directional connection. Then it will go for many evolution processes, finalize it like the picture below. Server, client, load balancer, and databases.

The HTTP protocol is not used because every user needs to send a request to the server, once a response is received from the client-server, then the connection is closed, it means every message needs to wait for the full transaction between server-client. But, we don't want to waste time and resources creating connection, so the WebSocket protocol is used because the connection is not closed immediately. Web Socket Handler WSH1 will be connected by the user. Web socket handler is a lightweight server which keeps an open connection with all the active users.

- **Media type messages**

chat service takes the messages and finds out the type of message. Once the message type or chat service detects the format of the message as media, it is stored in an S3 bucket which is an object storage service. The links to these media are then stored in a SQL or NoSQL db with mapping to the user details. We can use an HTTP protocol to transfer these messages.

CHAPTER - 6

DATABASE AND DICTIONARY

CHAPTER - 6

DATABASE AND DICTIONARY



DATA BASE DESIGN FOR WHATSAPP CLONE

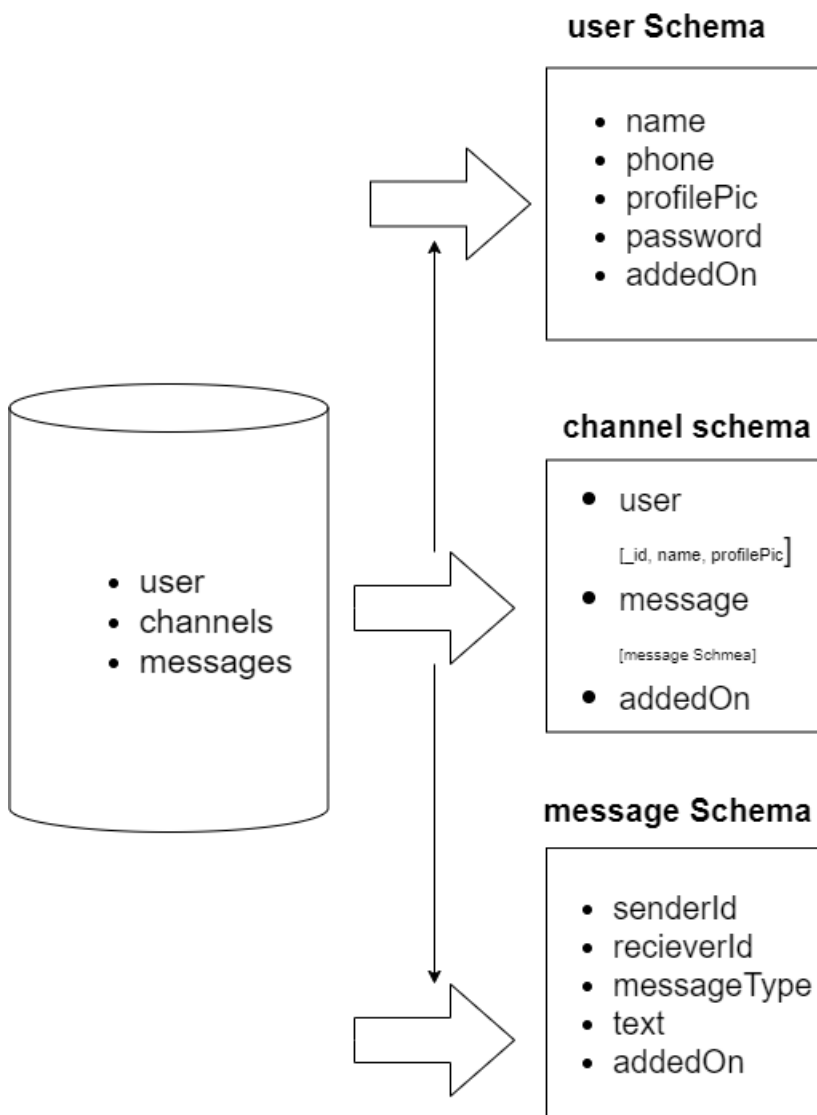


Figure 6.1 Data base structure

A database is a systematic collection of data. They support electronic storage and manipulation of data. Databases make data management easy.

Database servers are used to store and manage databases that are stored on the server and to provide data access for authorized users. This type of server keeps the data in a central location that can be regularly backed up. It also allows users and applications to centrally access the data across the network. A large number of the databases used in your organization can be kept on one server or a group of servers that are specifically configured to protect data and service client requests.

Databases are becoming increasingly widespread, and practically every company now uses them for a variety of purposes. Simultaneously, database technologies have advanced over time and now offer a wide range of capabilities. A Relational Database, whether Oracle, SQL Server, or DB2, has been the dominating back end of commercial systems since the 1980s.

With Big Data being the most important thing in IT since the Internet, developers all around the world are working on applications that can support a wider range of data formats. As Relational Databases generally don't cater to the growing Unstructured Data, NoSQL databases are becoming a new favourite among businesses in recent years. In comparison to Traditional Relational Databases, proponents of NoSQL platforms claim that they allow easier scalability and better performance. This category includes Open-Source solutions like Cassandra, MongoDB, and Redis, which excel at storing Unstructured Data.

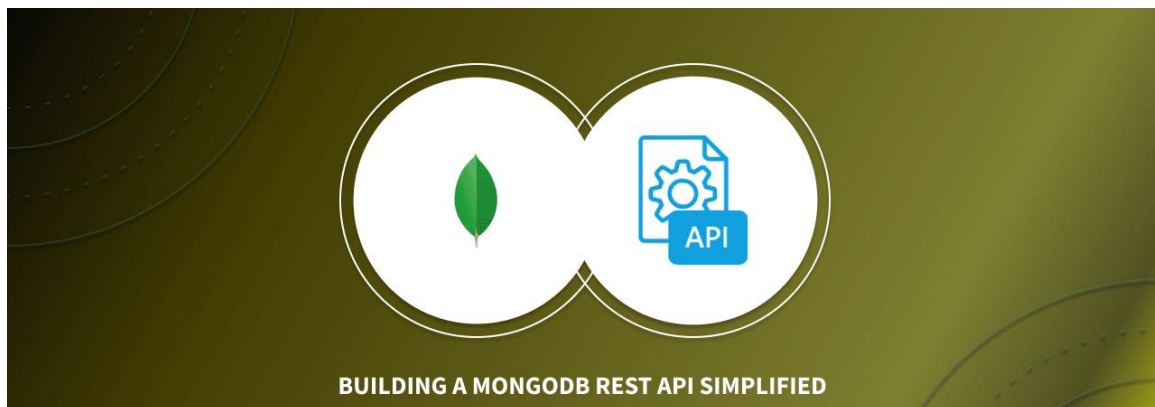


Figure 6.2 MongoDB as API

The MongoDB Atlas Data API lets you read and write data in Atlas with standard HTTPS requests. To use the Data API, all you need is an HTTPS client and a valid API key.

Clients send requests to specific endpoints, which each represent a MongoDB operation. The API includes endpoints that create, read, update, delete, and aggregate documents in your cluster. Requests specify a collection and include operation-specific parameters like a query filter or update description. Additionally, all incoming requests must include a Data API key to authenticate the calling user.

For each request, the Data API authenticates the user and checks the cluster's read/write permissions. If the user's request is authorized, then the API validates the request, runs the corresponding database operation, and returns the result in an HTTPS response.

6.1 Architecture of MongoDB API's used In Project

6.1.1. User API

user API store the data of user in terms of string entered by user or it can be fetch from gmail after gmail validation, usderData consist userId, username, phoneNumber, profilePic.

Code:

```
1  const mongoose = require('mongoose')
2
3  const userSchema = new mongoose.Schema({
4    name: { type: String, default: '' },
5    email: { type: String, default: '' },
6    profilePic: { type: String, default: '' },
7    addedOn: { type: Number, default: Date.now() }
8  })
9
10 userSchema.method({
11   saveData: async function () {
12     return this.save()
13   }
14 })
15 userSchema.static({
16   findData: function (findObj) {
17     return this.find(findObj)
18   },
19   findOneData: function (findObj) {
20     return this.findOne(findObj)
21   },
22   findOneAndUpdateData: function (findObj, updateObj) {
23     return this.findOneAndUpdate(findObj, updateObj, {
24       upsert: true,
25       new: true,
26       setDefaultsOnInsert: true
27     })
28   },
29 })
30 export default mongoose.model('wc-user', userSchema)
```

6.1.2. Channel API

Channel API is responsible for creating channel between two user whose data is stored in data base or those who have logged in using gmail. Its create a tunnel between two user so that they can hava chats.

Code:

```
1  const mongoose = require("mongoose");
2
3  const channelSchema = new mongoose.Schema({
4    channelUsers: [
5      {
6        email: { type: String, default: "" },
7        name: { type: String, default: "" },
8        profilePic: { type: String, default: "" },
9      },
10   ],
11   messages: [
12     {
13       senderEmail: { type: String, default: "" },
14       messageType: { type: String, default: "TEXT" },
15       text: { type: String, default: "" },
16       addedOn: { type: Number, default: Date.now() },
17     },
18   ],
19   addedOn: { type: Number, default: Date.now() },
20 });
21
22 channelSchema.method({
23   saveData: async function () {
24     return this.save();
25   },
26 });
27
28 channelSchema.static({
29   findData: function (findObj) {
30     return this.find(findObj);
31   },
32   findOneData: function (findObj) {
33     return this.findOne(findObj);
34   },
35   findOneAndUpdateData: function (findObj, updateObj) {
36     return this.findOneAndUpdate(findObj, updateObj, {
37       upsert: true,
38       new: true,
39       setDefaultsOnInsert: true,
40     });
41   },
42 });
43
44 export default mongoose.model("wc-channel", channelSchema);
```

6.1.3 Message API

Message API send the request to channel and once response get success message sent to destination by using email validation that validate the sender and receiver email to create channel between them and send receive message .

Code:

```
1  const sendResponse = (res, data, message, success, code) => {
2    const responseObj = {
3      responseData: data,
4      message: message,
5      success: success,
6      responseCode: code,
7    };
8    res.format({
9      json: () => {
10        res.send(responseObj);
11      },
12    });
13  };
14  const sendError = (res, data, msg) => {
15    if (!res) {
16      return false;
17    }
18    sendResponse(res, data, msg || "Request Failed", false, 400);
19  };
20  export { sendResponse, sendError };
```

6.2 Controllers

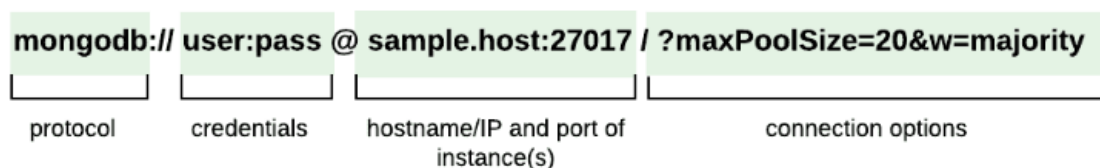
We have separately defined controller in a .js file that control whole API's and responsible to connect all the API's to server and with frontend.

Code:

```
1  /**
2   * Created by AyushK on 26/07/21.
3   */
4   import * as Controller from "../app/controllers";
5   import * as Validation from "../utility/validator";
6
7   module.exports = (app) => {
8     app.get('/', (req, res) => res.send(`API running fine`));
9
10    app.post('/user', Validation.validateCreateUser, Controller.createUser);
11
12    app.get('/search-user', Validation.validateSearchUser, Controller.searchUser);
13
14    app.post('/channel', Validation.validateCreateChannel, Controller.createChannel);
15
16    app.get('/channel-list', Validation.validateGetChannelList, Controller.getChannelList);
17
18    app.post('/message', Validation.validateAddMessage, Controller.sendMessage);
19  };
```

6.3 Connection URL

The **connection URI** provides a set of instructions that the driver uses to connect to a MongoDB deployment. It instructs the driver on how it should connect to MongoDB and how it should behave while connected. The following example explains each part of a sample connection URI:



Code:

```
1  import mongoose from "mongoose";
2
3  const DB_CONNECTION_URL = `mongodb+srv://whatsappclone:yqVF0kialZqsXA30@localdb.3dk81.mongodb.net/whatsapp-clone?retryWrites=true&w=majority`
4
5  const connectDB = () => {
6    console.log("DB trying to connect on " + new Date());
7
8    const options = {
9      keepAlive: 1,
10     autoReconnect: true,
11     poolSize: 10,
12     useNewUrlParser: true,
13     useUnifiedTopology: true,
14   };
15   return mongoose.connect(DB_CONNECTION_URL, options);
16 };
17 export default connectDB;
```

CHAPTER – 7

TESTING STRATEGY

CHAPTER – 7

TESTING STRATEGY

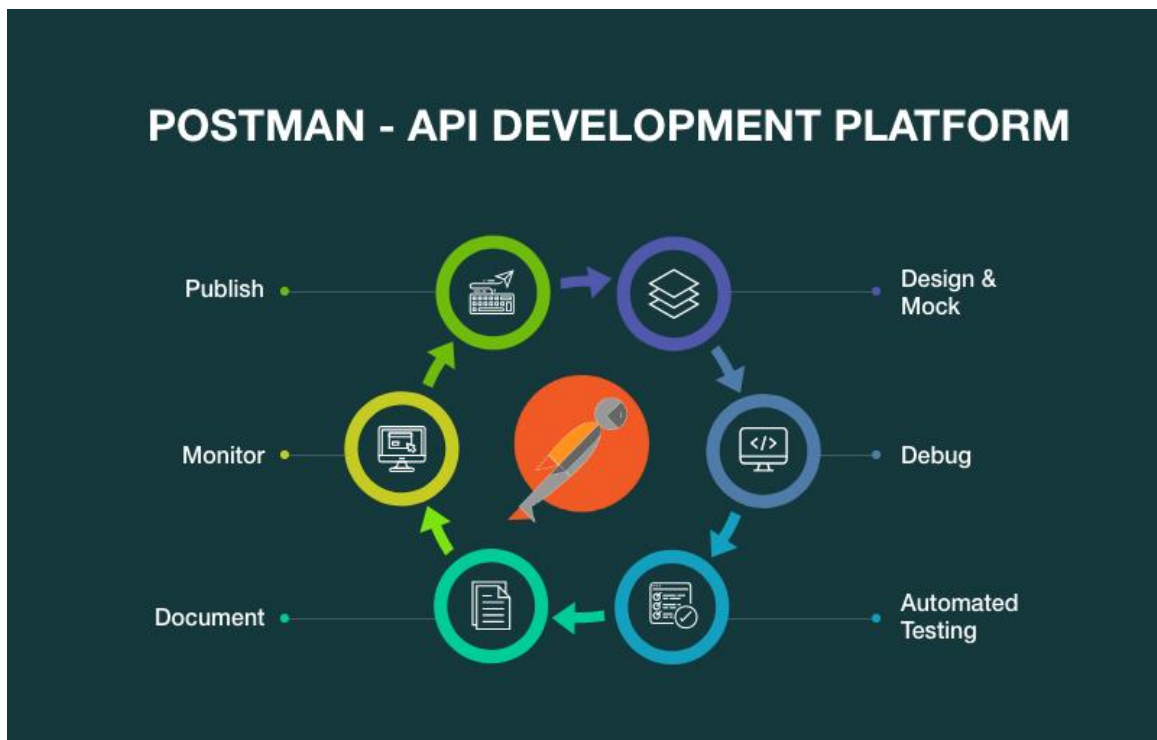
In this chapter, test cases of the application are examined.

7.1 System Testing

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. In simple words, testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

For testing we have used Postman tool:

The **Postman** testing tool is a complete API development platform with various built-in tools that support every stage of the API lifecycle. Postman tool allows you to design, mock, debug, automated testing, document, monitor and publish the APIs - everything From one place.



7.1.1 Cases 1 Sign Up

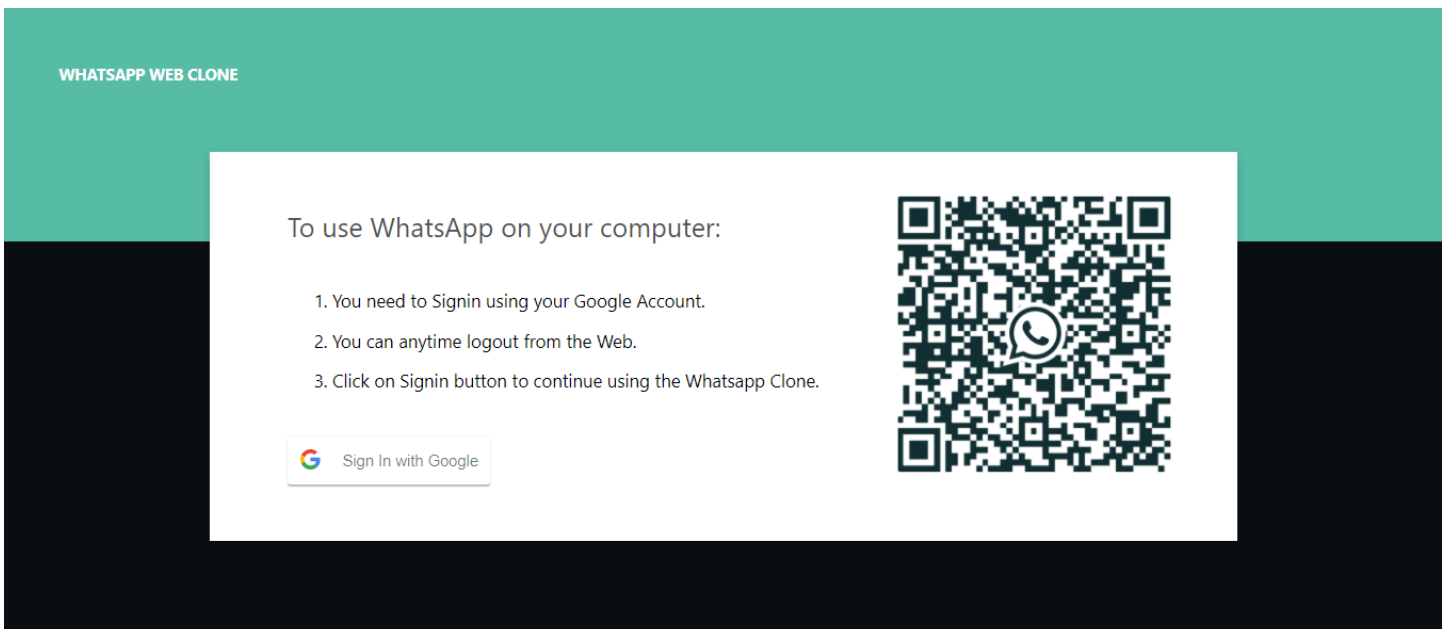
7.1.1.1 New User

Test Step

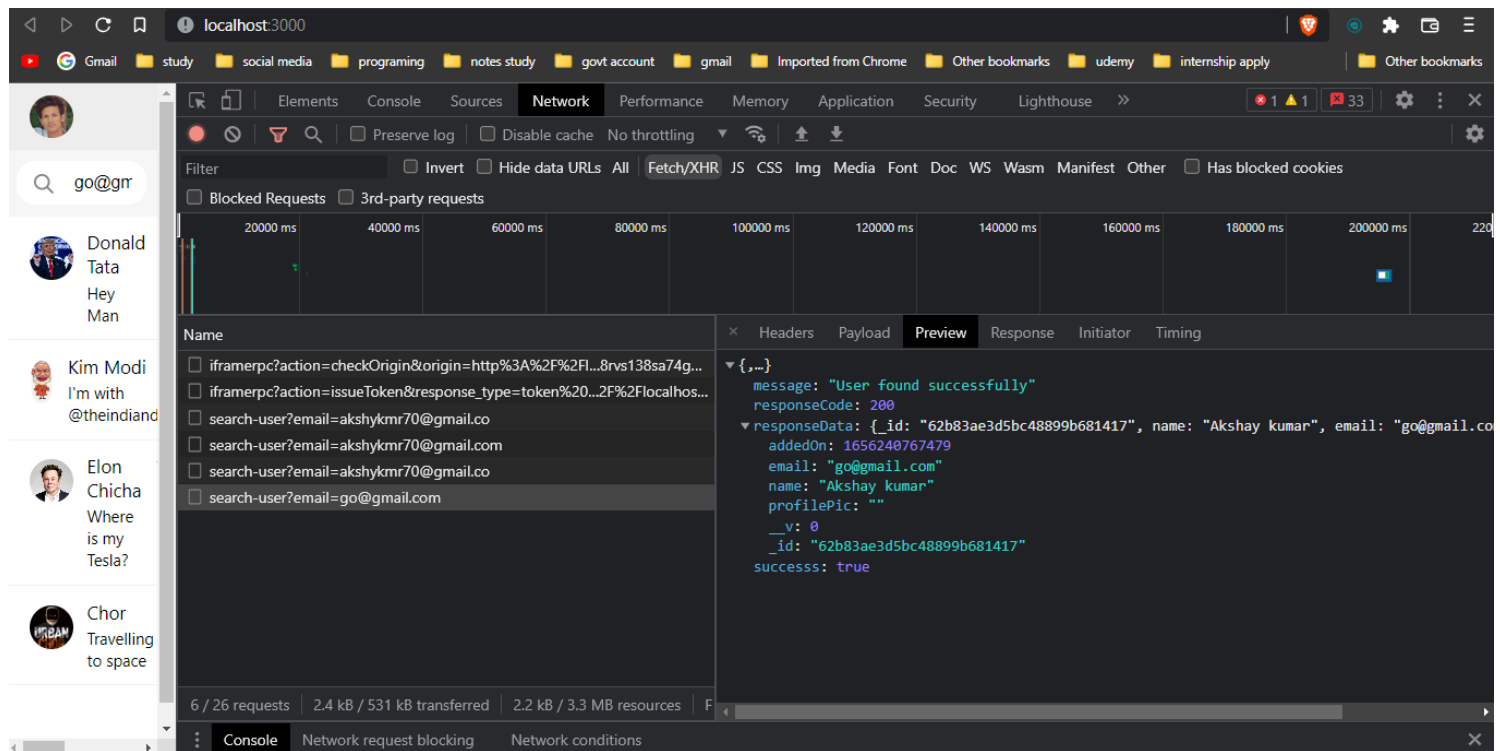
- Go to the whatsapp page and Sign Up with a valid email and a six-character password and hit click it will automatically validate you email and will store your data in database.

Expected Result

- It should allow the user and enter the chat panel
- Actual Outcome



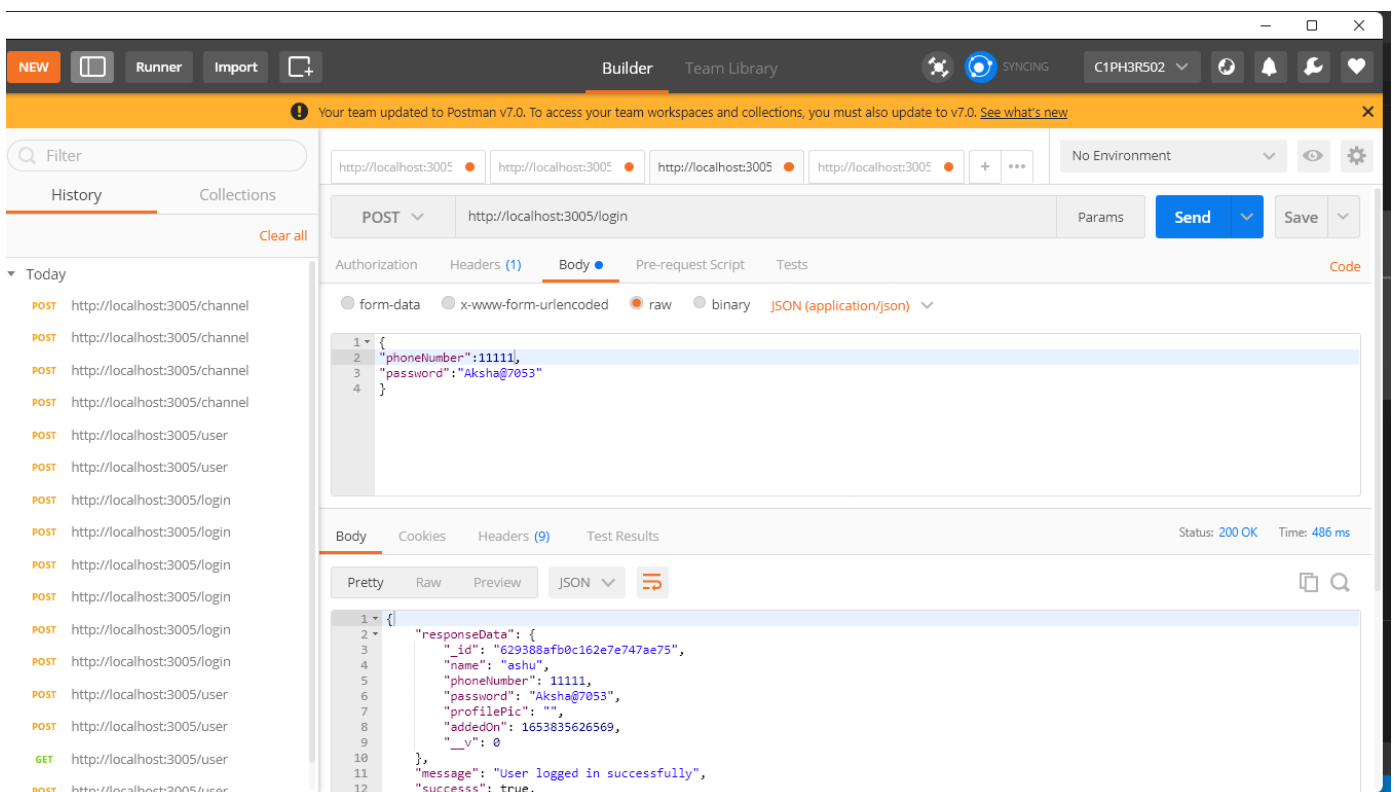
A new user is entered in the chat panel.



7.1.2 Check New Created User in database.

Test Step

- Open postman tool and entered the url of local server that is localhost:5000



- After entering user data post the request and below panel will show the result with user data and if it exists.

Expected Result

- Newly Created User email, created date and unique uid

Actual Outcome.

- A new Created User is available in data Authentication user list.

7.1.3 Login new User without Sign Up

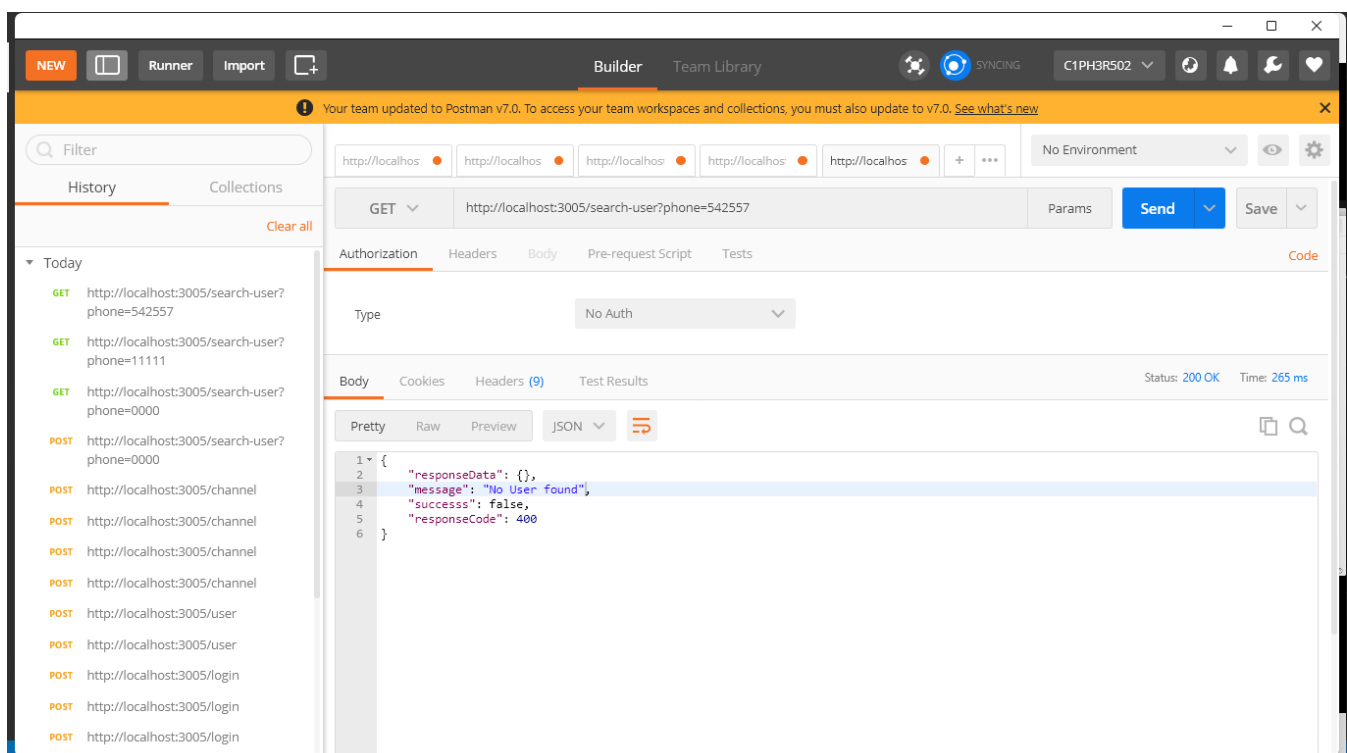
Test Step

- Go to the whatsapp page and Login with a valid email and a six-character password and just hit enter button.

Expected result

It should not allow the user to enter the chat panel

- Actual Outcome: User is not allowed to enter chat panel.



7.1.4 User messages with email and time

Test Step

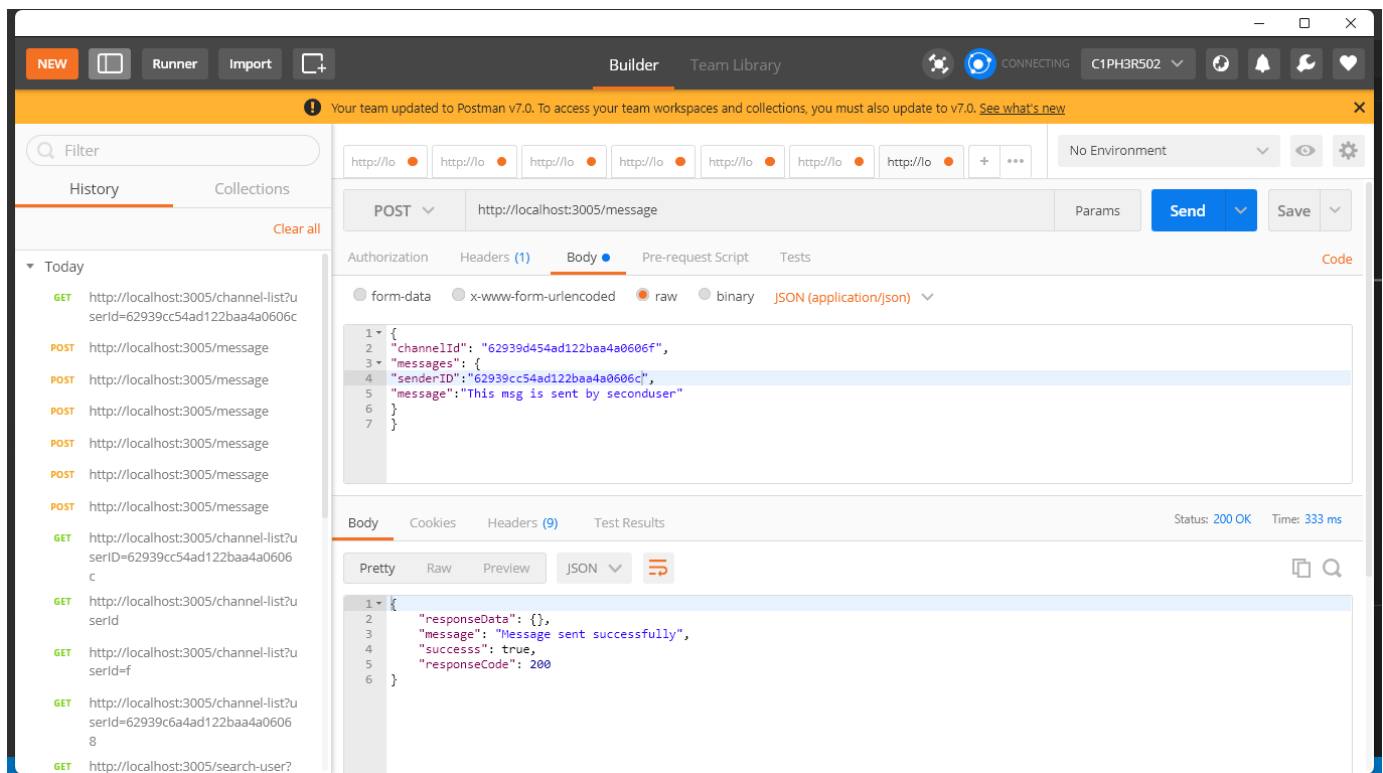
- Logged in user writes the message in Send Message Form

Expected Result

- Users write any message and receiver views the message with sender's email and time

Actual Outcome

- Sender's email id with time is viewed by receiver.



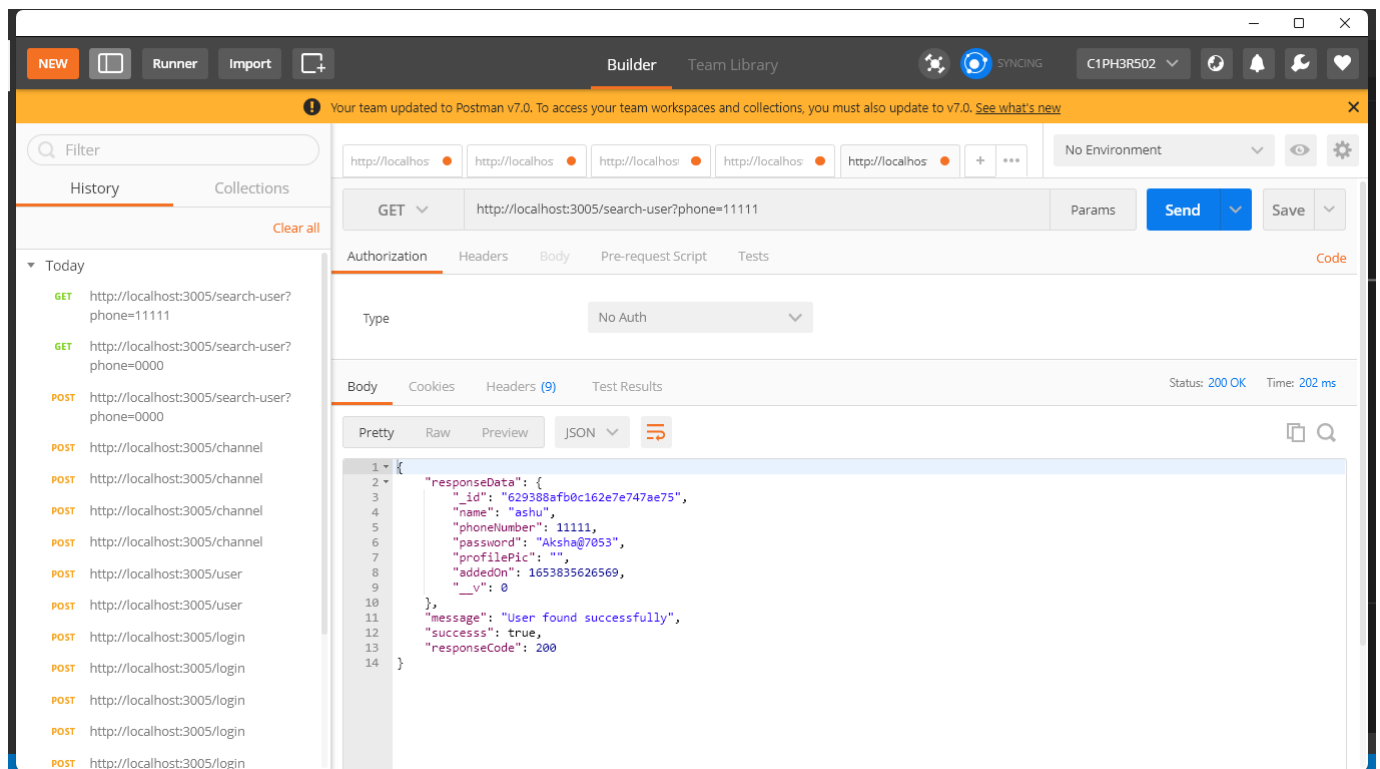
7.1.5 Logged in User able to create channel.

Test Step

- After log in user will search other user using email id.

Expected Result

Logged in user's data should be responded by the api with userData.



7.1.6 Channel between two users

Test Step

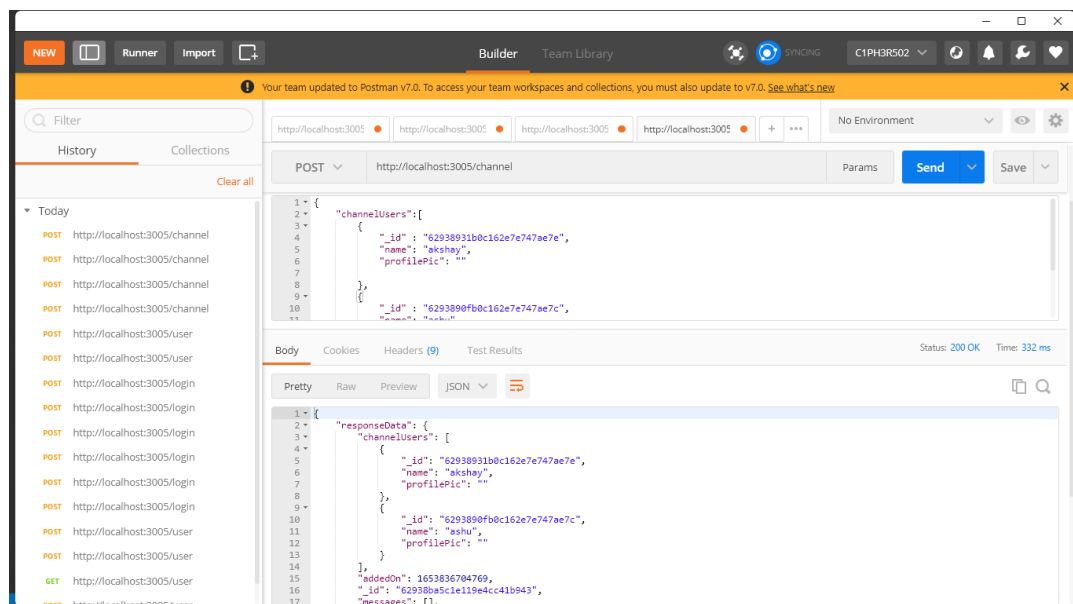
- Go to the postman tool and entered the url and also entered the user details between channel is to be created.

Expected Outcome

- Channel details should be visible with user data and channel id.

Actual Outcome

- Channel found successfully.

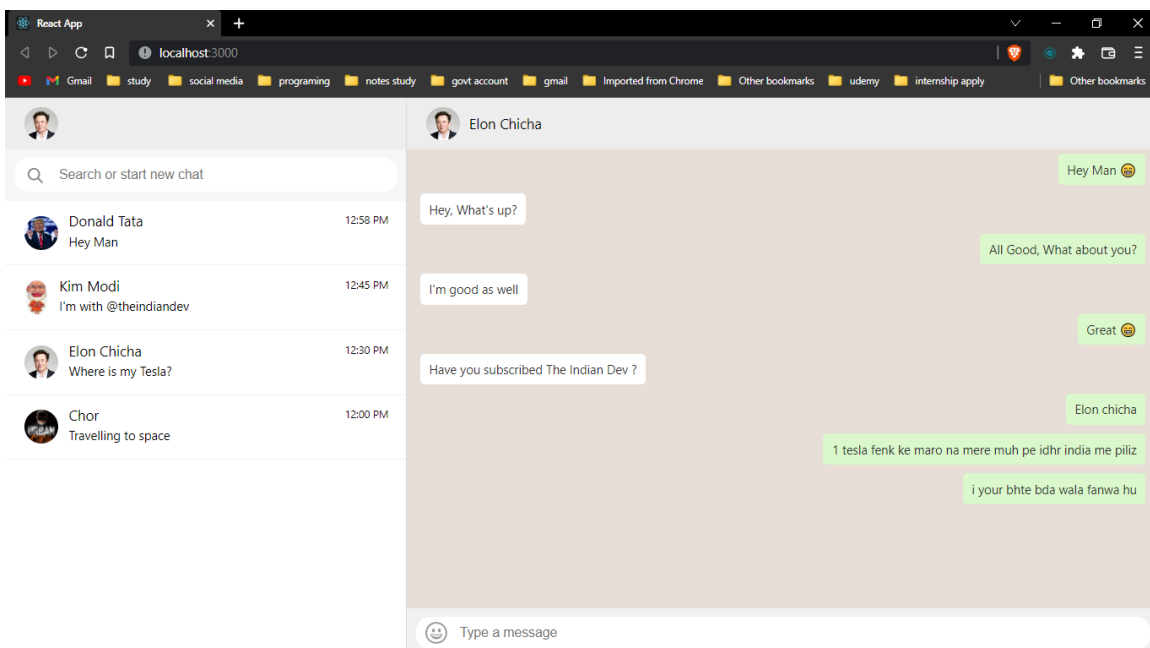
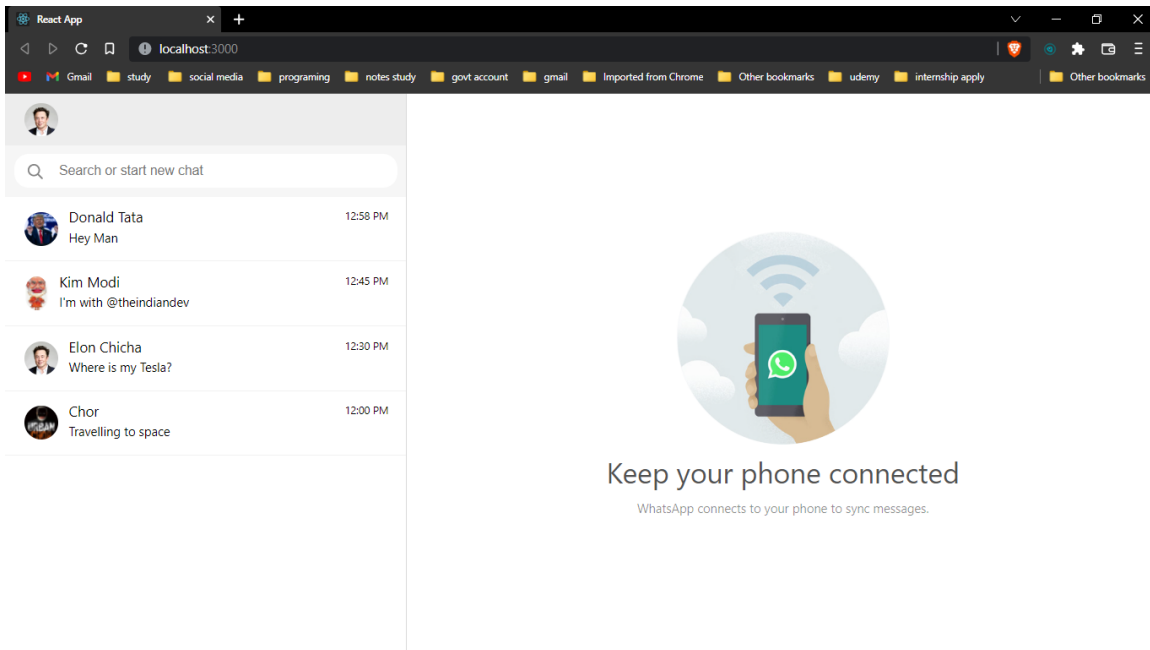


CHAPTER - 8

SNAPSHOTS OF PROJECT

CHAPTER – 8

SNAPSHOTS OF PROJECT



CHAPTER – 9

REFERENCES

CHAPTER - 9

REFERENCES

- SMS, Instant message – Last access 27/02/2022
<https://www.lifewire.com/difference-between-chat-and-instant-messaging-3969422>
- SMS, Instant message – Last access 27/02/2022
<https://www.trickstrend.com/popular-instant-messenger-apps/>
- Chat Apps pictures – Last access 27/02/2022
<http://xvdx.blogspot.com/2014/08/en-iyi-mobil-chat-sohbet-programi.html>
- Chat Features – Last access 27/02/2022
<https://www.userlike.com/en/blog/live-chat-advantages-disadvantages>
- HTML – Last access 27/02/2022
<https://www.babujitechnolife.in/2019/05/html-ladvantages-and-disadvantages/>
- CSS Features – Last access 27/02/2022
<https://connectusfund.org/6-advantages-and-disadvantages-of-cascading-style->
- BULMA – Last access 27/02/2022
<https://mobiosolutions.com/what-is-bulma-advantages-disadvantages-of-using-bulma-2/>
- Visual Studio Code – Last access 27/02/2022
<https://code.visualstudio.com/>
- Visual Studio Code – Last access 27/02/2022
https://en.wikipedia.org/wiki/Visual_Studio_Cde
- React JS Features – Last access 27/02/2022
<https://curatti.com/pros-cons-reactjs>
- React JS Features – Last access 27/02/2022
<https://www.peerbits.com/blog/reasons-to-choose-reactjs-for-your-web-development-project.html>
- React JS Features – Last access 27/02/2022
<https://mindmajix.com/reactjs-interview-questions>

CHAPTER – 10

SCOPE AND FUTURE ENHANCEMENT

CHAPTER -10

SCOPE AND FUTURE ENHANCEMENT

React JS is one of the most popular and powerful front-end technology around the world, today. It offers excellent performance in any application. In addition to this, React JS offers good compatibility on different platforms, browsers and devices. After developing the Chat App, it is clear that React JS is an easy to learn and convenient to implement.

Firebase played a crucial role in providing real-time database backend service. It is easy to use because extra code writing for the server is not required while it is a ready-made API. Visual Studio Code was an excellent IDE in this project.

I conclude by saying that this thesis application is a simple project but needs development in the future if the developer gets an opportunity to put effort into it.

CHAPTER – 11

BRIEF PROFILE

CHAPTER - 11

BRIEF PROFILE

I, Akshay Kumar, student of B. Tech Computer Science and Engineering, have created the above project report during my 8 th semester industrial training under MAGIC RESORTS PRIVATE LIMITED – A real state company for the partial fulfillment of award for B. Tech Degree.

During my industrial training, my role was as a website developer and other IT related issue management -

- Use extraction techniques to extract data from APIs, to use scripts to convert various files to .csv, .pdf and more
- Able to create a app with good interface.
- Able to detect bugs from big data.
- Able to work with third parties libraries.
- Show willpower and tenacity to explore all existing data for the specific indicator I am working on.
- Maintain clear and coherent communication, both verbal and written, to understand data needs and report results
- Produce visualizations using tool of my choice
- Create clear summaries reporting the progress of my work

I am hereby submitting my project report.

Yours Sincerely,
Akshay Kumar

