WhatsApp Clone

TEAM :

G.Rohit
B.Devika
G.Laxmikanth
Ooha Rani

Guidance :

Mr.Ajay Reddy Sir.

Introduction:

This documentation serves as a guide to understand and work with the WhatsApp Clone project, which is built using the MERN (MongoDB, Express.js, React.js, Node.js) stack. The project aims to replicate some of the core functionalities of the popular messaging application WhatsApp.

Project Overview:

The WhatsApp Clone project is a web based application that mimics the core functionalities of the WhatsApp messaging platform. Users can register, login, send messages, share media and more.

Contents:

1. Technologies Used:

- MongoDB: Database to store user information, messages, and group data.
- Express.js: Backend framework to handle serverside logic and API requests.
- React.js: Frontend library for building user interfaces.
- Node.js: JavaScript runtime for serverside development.
- Socket.IO: Realtime bidirectional eventbased communication library.
- JWT (JSON Web Tokens): For user authentication. (jwt= google credentials + secret key)
- MaterialUI: UI component library for React.
- Axios: Making HTTP Requests and handling Responses.

MERN STACK:

MongoDB is a popular open-source NoSQL database management system. It is designed to store and manage large volumes of unstructured data. MongoDB uses a document-oriented data model, which means data is stored in flexible, JSON-like documents. This allows for easy scalability and high performance.

Some key *features* of MongoDB include:

1. Flexible data model: MongoDB's document model allows for dynamic and flexible schemas, making it easy to store and retrieve complex data structures.

2. Integration with programming languages: MongoDB has official drivers for many programming languages, making it easy to integrate with your application.

Overall, MongoDB is a flexible and scalable database solution that is widely used in modern web applications, big data analytics, and real-time data processing.

Express.js is a popular web application framework for Node.js. It provides a simple and minimalistic approach to building web applications and APIs. Express.js is known for its flexibility, scalability, and ease of use.

Here are some key *features* and benefits of using Express.js:

1. Error handling: Express.js provides built-in error handling mechanisms that allow you to handle errors in a centralized manner. You can define error-handling middleware functions to catch and handle errors that occur during the request-response cycle.

2. Static file serving: Express.js allows you to serve static files such as HTML, CSS, JavaScript, and images. This is useful for serving frontend assets like stylesheets and client-side JavaScript files.

3. Middleware ecosystem: Express.js has a large and vibrant ecosystem of middleware modules that can be easily integrated into your application. These middleware modules provide additional functionality such as authentication, session management, logging, and more.

Express.js is widely used in building web applications and APIs due to its simplicity, flexibility, and extensive community support. It is a great choice for developers who want to quickly build scalable and efficient web applications using Node.js.

React.js is a JavaScript library for building user interfaces. It is widely used for creating interactive and dynamic web applications. React.js is known for its component-based architecture, virtual DOM, and efficient rendering.

Here are some key features and benefits of using React.js:

1. Component-based architecture: React.js follows a component-based approach, where the UI is divided into reusable and independent components. Each component encapsulates its own logic and state, making it easier to manage and maintain complex UI structures.

2. Virtual DOM: React.js uses a virtual DOM (Document Object Model) to efficiently update and render UI components. The virtual DOM is a lightweight representation of the actual DOM, and React.js compares the virtual DOM with the real DOM to determine the minimal set of changes needed to update the UI. This approach improves performance and reduces unnecessary re-rendering.

3. JSX syntax: React.js uses JSX (JavaScript XML) syntax, which allows you to write HTML-like code within JavaScript. JSX makes it easier to define the structure and appearance of UI components, and it provides a seamless integration between JavaScript and HTML.

4. Reusable components: React.js promotes reusability by allowing you to create reusable UI components. These components can be composed together to build complex UI structures, making it easier to maintain and update the application.

6. React ecosystem: React.js has a large and active community, which has resulted in a rich ecosystem of libraries, tools, and resources. There are numerous third-party libraries and frameworks that integrate well with React.js, providing additional functionality and enhancing development productivity.

React.js is widely used in modern web development for building single-page applications, mobile applications, and complex UIs. It is supported by major tech companies and has a strong community backing, making it a popular choice for developers.

Node.js is a JavaScript runtime environment that allows developers to run JavaScript code on the server-side. It provides an event-driven, non-blocking I/O model that makes it lightweight and efficient for building scalable and high-performance web applications.

Here are some key features and benefits of using Node.js:

1. JavaScript everywhere: Node.js allows developers to use JavaScript on both the client-side and server-side, which promotes code reusability and reduces the learning curve for full-stack development. This means that developers can use the same language and libraries throughout the entire application stack.

2. NPM ecosystem: Node.js has a vast ecosystem of open-source libraries and modules available through the Node Package Manager (NPM). NPM allows developers to easily install, manage, and share reusable code, making it faster and more efficient to build applications.

3. Fast and lightweight: Node.js is built on the V8 JavaScript engine, which is known for its speed and performance. Node.js applications are lightweight and have a small memory footprint, making them ideal for building microservices and serverless architectures.

4. Cross-platform compatibility: Node.js is cross-platform, which means that it can run on various operating systems such as Windows, macOS, and

Linux. This allows developers to build applications that can be deployed on different environments without major modifications.

Node.js is widely used in web development for building server-side applications, APIs, real-time applications, microservices, and more. Its combination of JavaScript, scalability, and performance makes it a popular choice for building modern web applications.

Socket.IO is a JavaScript library that enables real-time, bidirectional communication between the client and the server. It is built on top of the WebSocket protocol and provides additional features such as fallback options, event-based communication, and room-based messaging.""" Here are some reasons why Socket.IO is important:

1. Real-time communication: Socket.IO allows for real-time communication between the client and the server. This means that data can be sent and received instantly, enabling real-time updates, notifications, and collaborative features in applications.

2. Bi-directional communication: Unlike traditional HTTP requests, which are unidirectional (client to server), Socket.IO enables bidirectional communication. This means that both the client and the server can send data to each other at any time, allowing for more interactive and dynamic applications.

3. Event-based communication: Socket.IO uses an event-driven architecture, where the client and server can emit and listen to events. This makes it easy to define custom events and handle them on both ends. Events can be used to trigger actions, update data, or notify clients about specific events happening on the server.

""Socket.IO is widely used in various applications that require real-time communication, such as chat applications, collaborative tools, real-time analytics, multiplayer games, and more. Its ease of use, flexibility, and wide browser support make it an important tool for building interactive and real-time web applications.""

JWT stands for **JSON Web Token**. It is an open standard for securely transmitting information between parties as a JSON object. JWTs are commonly used for authentication and authorization purposes in web applications.

Here's how JWT works:

1. Authentication: When a user logs in to a web application, the server generates a JWT containing the user's identity and any additional information needed for authentication. This JWT is then sent back to the client and stored, typically in local storage or a cookie.

2. Authorization: For subsequent requests to protected routes or resources, the client includes the JWT in the request header or as a query parameter. This allows the server to verify the authenticity of the request and determine if the user has the necessary permissions to access the requested resource.

Benefits of using JWT:

1. Stateless and scalable
2. Security
3. Cross-domain usage
4. Customizable and extensible
5. Widely supported

It's important to note that JWTs should be used carefully and securely. Proper implementation and validation of JWTs are crucial to ensure the security and integrity of the authentication and authorization process.

Material-UI is a popular open-source library for building user interfaces in React.js. It provides a set of pre-designed and customizable UI components that follow the Material Design guidelines created by Google. Material-UI allows developers to create visually appealing and responsive web applications with ease.

Material-UI is widely used in building modern and visually appealing web applications. Its combination of Material Design principles, pre-designed components, and customization options make it a popular choice for developers who want to create beautiful and responsive user interfaces in React.js.

Axios is a popular JavaScript library used for making HTTP requests from the browser or Node.js. It provides a simple and intuitive API for performing asynchronous HTTP requests and handling responses.

Here are some key *features* and benefits of using Axios:

1. Promise-based: Axios is built on top of JavaScript promises, which allows for cleaner and more readable asynchronous code. It supports both the traditional promise-based approach and the newer async/await syntax for handling asynchronous operations.

2. Browser and Node.js support: Axios can be used in both browser-based JavaScript applications and server-side Node.js applications. This makes it a versatile choice for making HTTP requests in various environments.

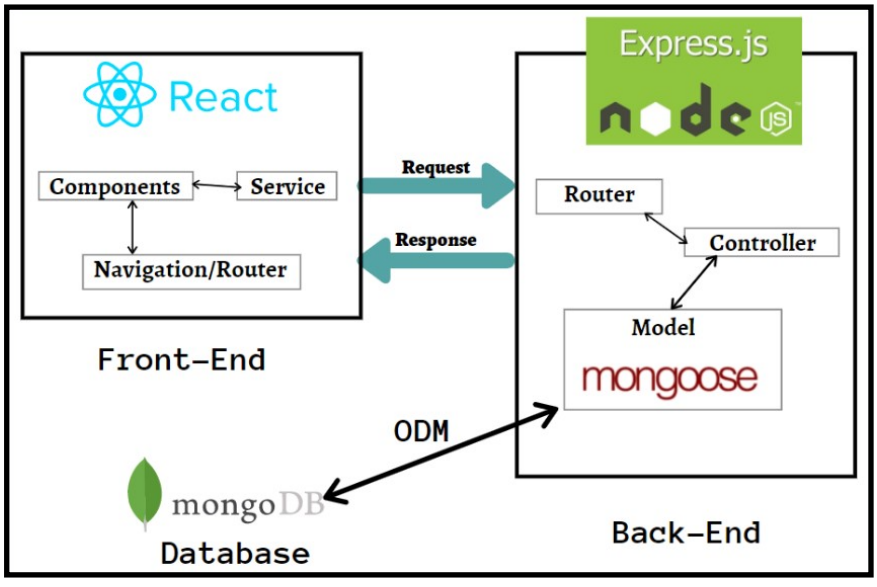
3. Easy to use: Axios provides a simple and intuitive API for making HTTP requests. It supports all the common HTTP methods like GET, POST, PUT, DELETE, etc. and allows for easy customization of request headers, query parameters, and request bodies.

4. Interceptors: Axios allows you to define interceptors for both requests and responses. Interceptors can be used to modify request or response data, add headers, handle errors, or perform other custom logic. This makes it easy to implement global request/response handling and authentication mechanisms.

5. Error handling: Axios provides built-in error handling capabilities. It automatically rejects promises and throws errors for common HTTP error status codes (e.g., 404, 500). Additionally, you can define custom error handling logic to handle specific error scenarios.

6. Cancelation support: Axios supports canceling requests, which can be useful in scenarios where you want to abort an ongoing request or prevent unnecessary requests from being sent. This can help improve performance and reduce network congestion.

Axios is widely used in web development for making HTTP requests to APIs, fetching data from servers, and handling AJAX operations. Its simplicity,



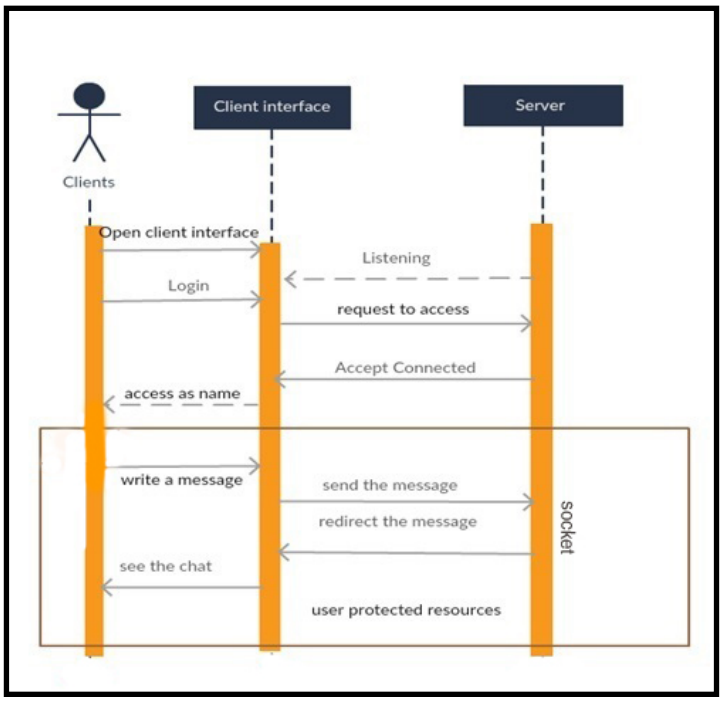
flexibility, and extensive community support make it a popular choice among developers for handling HTTP communication in JavaScript applications.

2. Project Structure:

- client: Frontend React js application.
- server: Backend Express js application.
- socket: Socket IO event-driven library.
- public: Public assets for the frontend.
- src: Source code for the frontend.
- components: All Front-end components.
- services: For handling API requests(Axios) from front-end.
- context: For storing account context.
- models: MongoDB schemas(mongoose).
- routes: Backend API routes.
- controllers: Logic for API routes.
- database: MongoDB connection.
- utils: Utility functions.

SetUp Instructions:

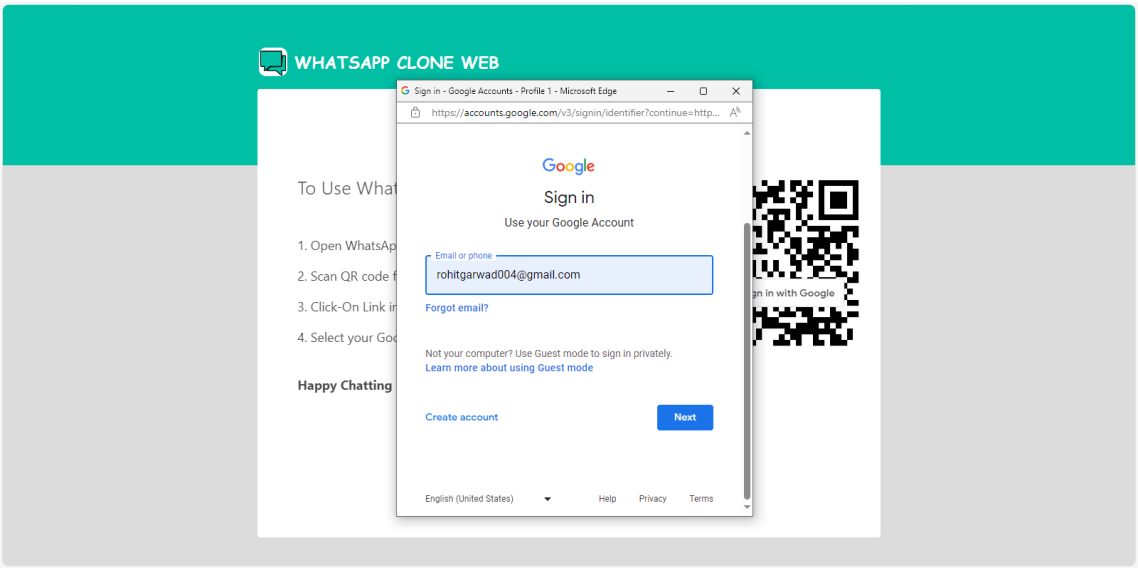
1.Clone the repository: `git clone < https://github.com/rohitgarwad/CAPSTONE_PROJECT-GROUP-2-WhatsApp_Clone >`
2. Navigate to the project directory: `cd CAPSTONE_PROJECT-GROUP-2-WhatsApp_Clone `
3. Install dependencies:
   - For the client: `cd client && npm install`
   - For the server: `cd server && npm install`
   - For the socket: `cd socket && npm install`
4. Setup MongoDB database and obtain connection URI.
5. Configure environment variables:
   - Create a `.env` file in the server directory.
   - Define variables:
   - `MONGODB_URI=<mongodb+srv://<username>:<password>@clone-whatsapp.ukgm3b2.mongodb.net/>`
   - `JWT_SECRET=<your-jwt-secret>`
6. Start the server: `cd server && npm start`
7. Start the client: `cd client && npm start`
8. Start the socket: `cd socket && npm start`
9. Access the application at `http://localhost:3000`.



Sequence Diagram

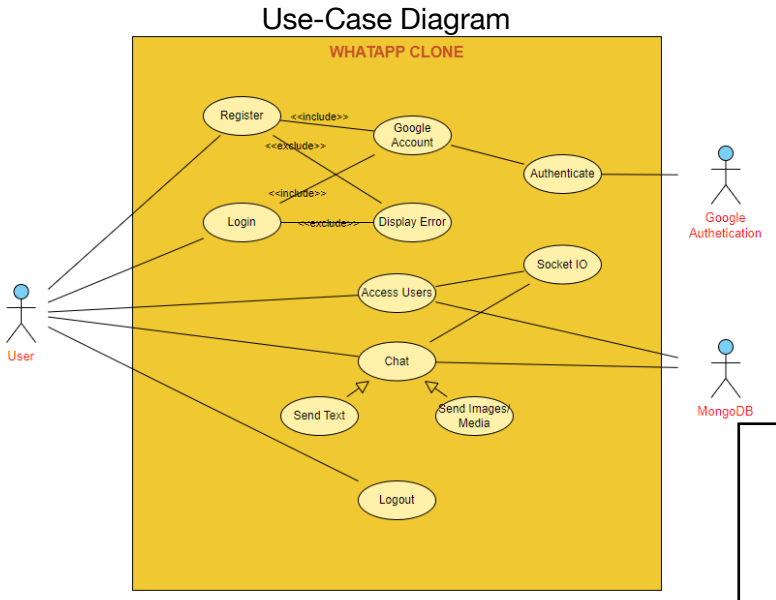
3. Key Features:

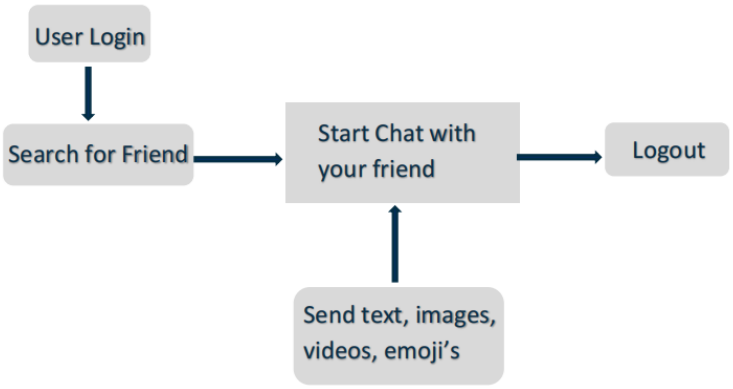
- User registration and authentication using Google OAuth service.
- Real-time chatting and bi-directional communication using Socket.IO.
- Sending and receiving messages in chat.
- Attachments support (images, documents).
- View online/offline status of users.
- View message timestamps.
- Notification for latest message
- Responsive design for various screen sizes.



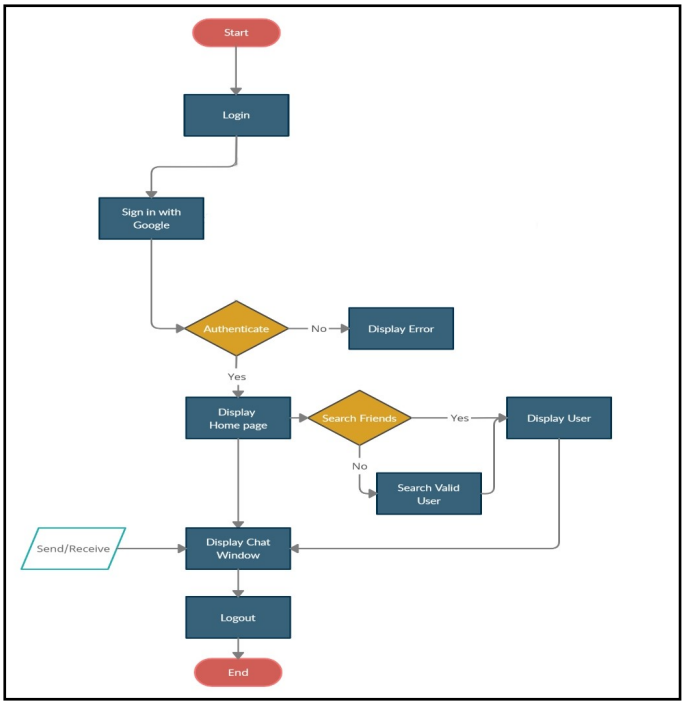
4. Diagrams:

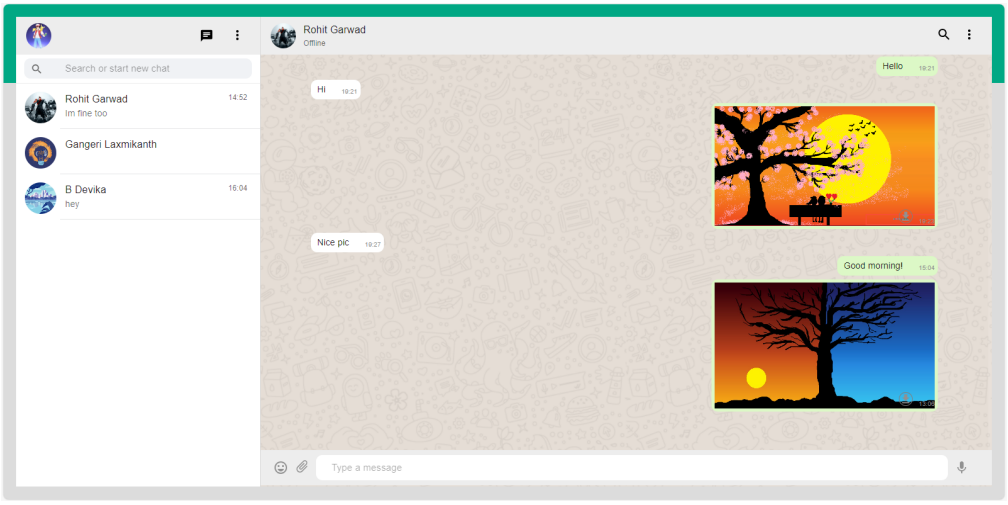
Use-Case Diagram





Flow-Chart Diagram





5. Security Measures:

To safeguard data security and user privacy, the WhatsApp Clone implements the following security measures:

- Secure API Calls: Utilizes secure protocols for fetching and transmitting event data.
- Limited User Data Storage: Minimizes the storage of userspecific data to protect privacy.
- Compliance: Adheres to relevant data protection regulations.

- Google OAuth Provides account security through JWT encryption.

- .env file hides the login credentials of mongoDB

6. Future Enhancements:

To further elevate the platform's capabilities, future enhancements may include:

- End-to-end encryption for messages.
- Message deletion and editing.
- Attaching emojis in chat.
- User profile customisation.
- Notification system.

7.Conclusion:

- The WhatsApp clone project is a web-based application that mimics the features and functionality of the popular messaging app WhatsApp.
- The project uses HTML, CSS(Material UI), JavaScript (React js, Express js, Node js) , Socket IO and MongoDB as the main technologies to create a responsive and real-time chat interface.
- The project demonstrates the use of various web development concepts, such as HTML(MUI) elements, CSS styling, JavaScript events, React Hooks , Google authentication, and MongoDB database.
- The project also showcases the design and layout of the WhatsApp web UI, including the chat window, message input box, and conversations .
- The project aims to provide a learning experience for web developers who want to create their own chat applications using modern web technologies.

8. Reference:

- https://www.youtube.com/@codeforinterview
- https://www.mongodb.com/cloud/atlas
- https://expressjs.com/
- https://react.dev/learn
- https://nodejs.org/en/learn/getting-started/introduction-to-nodejs
- https://mui.com/material-ui/all-components/
- https://developers.google.com/identity/protocols/oauth2
- https://learn.techsaksham.org/login
- https://portal.azure.com/#home
- https://github.com/ajay4ugit