Front End Engineering-II

Project Report

Semester-VI (Batch-2022)

YAPSTER

Supervised By:

Mr. Rahul

Submitted By:

Nischay , 2210991987 (G-24)

Nitin Goyal , 2210992000 (G-24)

Prikshit ,2210992085 (G-24)

Department of Computer Science and Engineering

Chitkara University Institute of Engineering & Technology, Chitkara University, Punjab

TABLE OF CONTENTS

1. Abstract

Yapster is a real-time chat application built using the MERN stack and Socket.io, designed to offer a seamless communication experience. The platform includes essential features such as instant messaging, media sharing, and an intuitive user interface. To ensure secure user authentication, Yapster implements access and refresh tokens along with OTP-based email verification and password recovery. The application utilizes Socket.io for low-latency real-time communication and supports media uploads via Cloudinary. The frontend is developed with React, Vite, and Tailwind CSS for a responsive and visually engaging user interface. Additionally, Yapster uses Redux Toolkit for efficient state management and integrates Postman for API testing. This project offers an excellent opportunity to enhance skills in MongoDB, Express.js, React.js, and Node.js while building a dynamic, scalable, and secure messaging platform.

2. Introduction

2.1 Background

With the growing demand for seamless communication, the need for a fast, secure, and engaging chat platform has become more essential than ever. Traditional messaging apps often face challenges related to user engagement, message delivery speed, and security. Many platforms struggle with slow response times, fragmented conversations, and unreliable notifications, leading to poor user experiences and decreased retention. Additionally, security concerns such as data breaches and weak authentication methods threaten the safety of user conversations. Yapster is developed to address these challenges by utilizing the MERN stack and Socket.io, offering a real-time, feature-rich, and scalable solution for users and administrators. By incorporating secure authentication methods such as access and refresh tokens, OTP-based email verification, and password recovery, the platform ensures robust security. Yapster also features an intuitive chat interface with real-time messaging, media sharing, and personalized notifications, enhancing user engagement. The use of Cloudinary ensures smooth media uploads, while Socket.io powers real-time, low-latency communication for a seamless chat experience.With a responsive frontend powered by React, Vite, and Tailwind CSS, Crave Cart offers an engaging shopping experience with fast performance, smooth navigation, and a visually appealing UI. The use of Redux Toolkit for state management enhances application efficiency, while

Postman aids in thorough API testing to ensure backend reliability and smooth communication between the server and client. Additionally, interactive alerts powered by Sweet Alert enhance user experience by providing real-time feedback during key processes such as message sending and media sharing. This project is designed to enhance development skills in full-stack technologies while creating a real-world chat application that meets industry standards. By incorporating modern best practices in UI/UX design, security, and performance optimization, Yapster aims to set a benchmark for scalable and user-centric messaging platforms.

2.1 Objectives

The primary objective of **Yapster** is to develop a secure, scalable, and user-friendly real-time chat application that enhances the messaging experience. The project focuses on implementing a robust authentication system using access and refresh tokens, OTP-based email verification, and password recovery to ensure data security and user trust. Another key goal is to provide an efficient admin panel for user and conversation management, including real-time message tracking, notifications, and media sharing to improve user interaction. The integration of **Cloudinary** for media uploads ensures high-quality images and videos without compromising performance, while **Socket.io** enables fast, real-time communication, promoting seamless user engagement.

Additionally, **Yapster** prioritizes a responsive, visually appealing interface using **React**, **Vite**, and **Tailwind CSS**, delivering an intuitive experience across all devices. To optimize performance and maintain efficient state management, **Redux Toolkit** is utilized, ensuring smooth interactions and data flow between components.

Beyond functionality, the project emphasizes performance optimization, API efficiency, and security best practices. By leveraging **Postman** for API testing and debugging, the reliability and integrity of backend operations are maintained. Furthermore, interactive UI elements powered by **Sweet Alert** enhance user engagement, providing instant feedback during key actions such as sending messages, media sharing, and account authentication.

Through this project, developers will gain hands-on experience in full-stack development using the MERN stack and **Socket.io**, focusing on industry-standard practices in authentication, real-time communication, and UI/UX design. **Yapster** serves as both a learning opportunity for developers and a scalable, high-performance messaging platform capable of adapting to evolving communication needs.

3. Problem Definition and Requirements

3.1 Problem Statement

In today's digital world, social networking platforms face challenges related to user engagement, security, scalability, and performance. Many existing social applications struggle with maintaining constant user interaction, providing seamless communication features, and ensuring secure user authentication. Additionally, many platforms fail to offer efficient messaging management, intuitive interfaces, and robust security measures, leading to poor user experiences and reduced platform engagement. Yapster, a real-time online communication platform, aims to address these challenges by providing a unique, secure, and scalable environment for continuous communication among friends. Yapster is designed to allow users to engage in non-stop, real-time conversations while ensuring secure authentication with features like access and refresh tokens, OTP-based email verification, and password recovery. It incorporates a seamless chat interface, efficient messaging management, and a scalable backend architecture to support an ever-growing user base. By leveraging modern web technologies such as React, Vite, Tailwind CSS, and Redux Toolkit, Yapster ensures a smooth, user-friendly**,** and high-performance messaging experience.

3.2 Software Requirements/Hardware Requirements/Data Sets

Software Requirements

Frontend Technologies:

- **React.js:** A JavaScript library used to build dynamic and responsive user interfaces, ensuring a seamless and interactive chat experience.
- **Vite:** A fast build tool to speed up development and optimize builds for better performance**.**
- **Tailwind CSS:** A utility-first CSS framework to create modern, responsive, and mobile-friendly designs for Yapster's user interface.

- **Redux Toolkit:** A library to manage the global state of the application, ensuring real-time message synchronization across the platform.
- **React Router:** A tool for client-side routing to facilitate smooth navigation between different sections of the app, like chat rooms, notifications, and profile pages.

Backend Technologies:

- **Node.js**: A JavaScript runtime environment that powers the server-side logic and manages API requests and real-time communication.
- **Express.js**: **A framework for Node.js to handle backend routing and middleware, enabling smooth interactions with the frontend.**
- **Mongoose:** A MongoDB ODM (Object Data Modeling) library for managing data models, ensuring smooth integration with the database for users and chat data.
- **JsonWebToken (JWT):** A secure authentication mechanism using access and refresh tokens for safe user login and session management.
- **Dotenv**: A module for managing environment variables, ensuring sensitive data like API keys and secrets remain secure.

Database:

- **MongoDB:** A NoSQL database designed to handle large volumes of real-time data, such as messages, user profiles, and chat history.

Authentication & Security:

- **OTP-based Email Verification:** An email verification system to confirm users' identities and ensure legitimate access to the platform using Resend for sending OTPs.

- **Password Recovery System: A secure and easy-to-use password recovery mechanism to help users reset their passwords when needed.**
- **Access & Refresh Tokens:** JWT-based tokens for secure user authentication, ensuring safe login and session management.

Messaging & Real-Time Communication:

- **Socket.io:** A real-time communication library to enable instant messaging and live chat features across the platform.

Development & Testing Tools:

- **Postman:** A tool used for testing and debugging API endpoints to ensure seamless backend interactions with the frontend.
- **Axios:** A promise-based HTTP client for handling API requests from the frontend.
- **VS Code:** A widely-used code editor for development, supporting full-stack and real-time communication app development.

User Interaction & Alerts:

- **SweetAlert:** A library for displaying interactive, customizable alerts and notifications, ensuring a smooth user experience for messages, errors, or system notifications.

Hardware Requirements

Minimum Requirements:

- **Processor:** Intel Core i3 (8th Gen) / AMD Ryzen 3 (or equivalent).
- **RAM:** 8 GB.

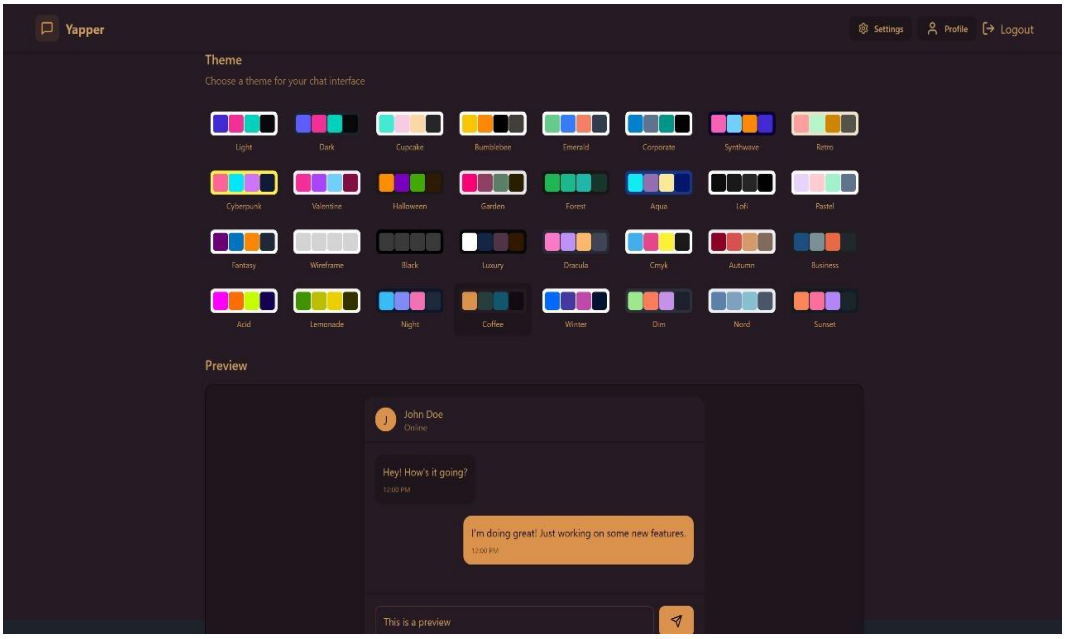
- **Storage:** 256 GB SSD or 500 GB HDD.
- **Graphics:** Integrated GPU (Intel UHD / AMD Vega).
- **Operating System:** Windows 10 / macOS / Linux (Ubuntu preferred).
- **Internet Connection:** A stable broadband connection for seamless real-time communication and API testing.

Recommended Requirements:

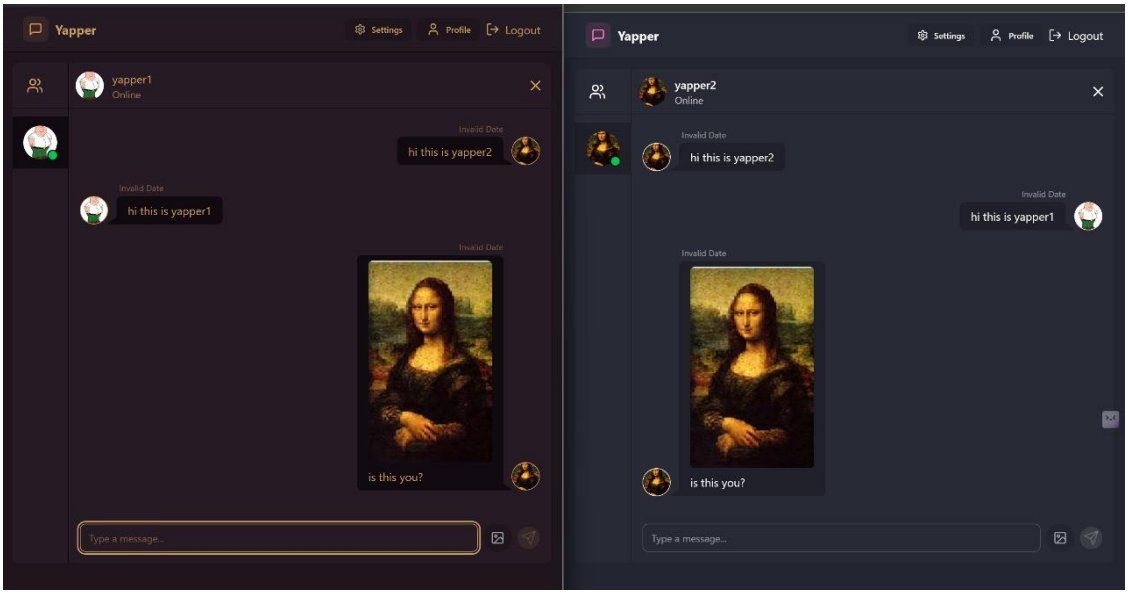
- **Processor:** Intel Core i5/i7 (10th Gen or higher) / AMD Ryzen 5/7 (or equivalent).
- **RAM:** 16 GB or higher for optimal multitasking and smoother experience with multiple applications.
- **Storage:** 512 GB SSD or higher for faster performance and storage of project files.
- **Graphics:** Dedicated GPU (NVIDIA GTX 1650 or AMD Radeon RX 5500).
- **Operating System**: Windows 11 / macOS Monterey or later / Linux (Ubuntu 22.04 LTS).
- **Internet Connection:** High-speed internet for smooth real-time messaging, API interactions, and cloud-based services.

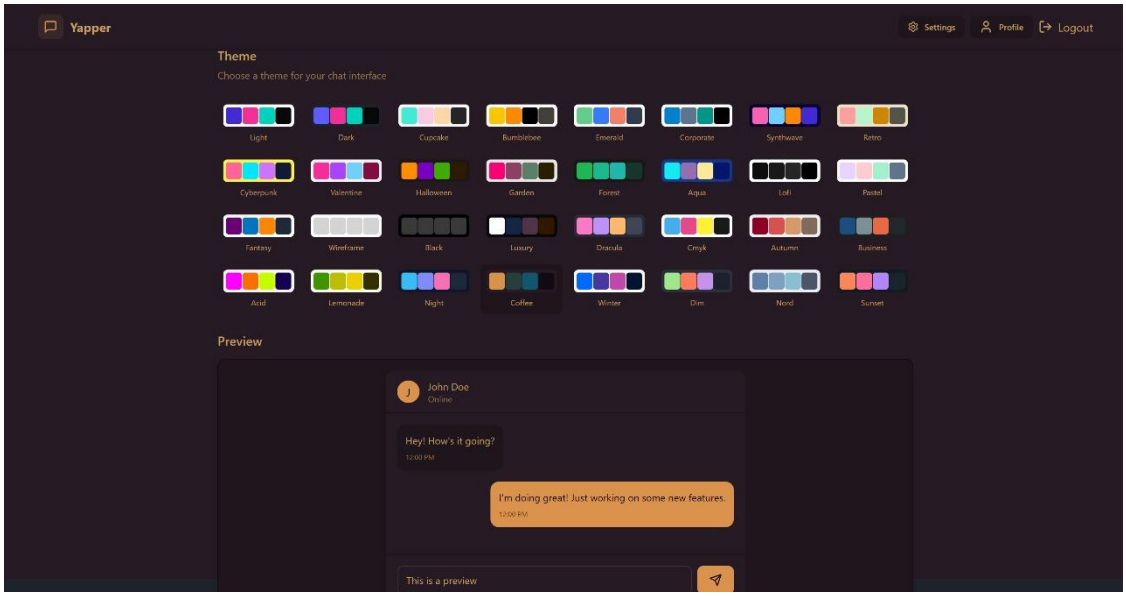
---

3. Proposed Design / Methodology



4. Results/Screenshots


Yapper
Settings
Profile
Logout
yapper1
Online
Invalid Date
hi this is yapper2
Invalid Date
hi this is yapper1
Invalid Date
is this you?
Type a message...
Yapper
Settings
Profile
Logout
yapper2
Online
Invalid Date
hi this is yapper2
Invalid Date
hi this is yapper1
Invalid Date
is this you?
Type a message...


Yapper
Settings
Profile
Logout
Theme
Choose a theme for your chat interface
Light
Dark
Cupcake
Bumblebee
Emerald
Corporate
Synthwave
Retro
Cyberpunk
Valentine
Halloween
Garden
Forest
Aqua
Lofi
Pastel
Fantasy
Wireframe
Black
Luxury
Dracula
Cmyk
Autumn
Business
Acid
Lemonade
Night
Coffee
Winter
Dim
Nord
Sunset
Preview
John Doe
Online
Hey! How's it going?
12:00 PM
I'm doing great! Just working on some new features.
12:00 PM
This is a preview

5. References

- **https://www.npmjs.com/package/jsonwebtoken**
- **https://vite.dev/**
- **https://www.npmjs.com/package/dotenv**
- **https://react.dev/**
- **https://cloudinary.com/**
- **https://www.postman.com/**