Documentation

React Web Chat App

Sahil Gupta , B21125

Github repo Frontend

Github repo Backend

Table Of Contents

Brief Overview:

My chat application is a modern and feature-rich communication platform built using React. It offers a comprehensive set of functionalities designed to enhance user interactions and streamline communication. In addition to standard chat features, the application also includes:

Real-Time Messaging:
Users can engage in real-time text-based conversations, ensuring swift and efficient communication.

User Authentication:
Robust authentication mechanisms safeguard user data and privacy.

User Profiles:
Users can create and personalize their profiles, fostering a sense of identity.

Chat Types:
The app supports both one-on-one and group chats, catering to various communication needs.

Media Sharing:
Users can seamlessly share images, documents, links, and other media types during conversations.

Notifications:
Real-time notifications keep users informed about new messages, ensuring they stay up-to-date.

Online Status:
Users can see the online status of their contacts, enhancing visibility and availability.

Message Replies:
The application supports message replies, making it easier to respond to specific messages within a conversation.

Search Functionality:
Users can search for specific messages or keywords within their chat history for efficient message retrieval.

Customization:
Personalization options allow users to tailor their chat experience to their preferences.

Calling Feature:
The application includes a calling feature that enables users to make voice and video calls within the platform, further enhancing communication capabilities.

Mobile Responsiveness: The platform is designed to be fully responsive, offering a seamless user experience on both desktop and mobile devices.

System Architecture Overview:

Frontend Layer:

React Frontend:
The user interface of the chat application is built using React.js, a popular JavaScript library for building user interfaces.

Backend Layer:

Node.js Server:
The backend of the chat application is powered by Node.js, which provides a non-blocking, event-driven architecture for handling real-time features.

Express.js:
Express.js is used as the web application framework to handle routing, middleware, and server-side logic.

Socket.IO:
Socket.IO is utilized for real-time communication between the server and clients, enabling features like real-time messaging and calling.

Database (MongoDB):
A database system, such as MongoDB, stores user data, chat messages, and other relevant information.

Authentication Services:
Authentication services are integrated to handle user registration, login, and secure access to the application.

Zegocloud for Calling:
WebRTC (Web Real-Time Communication) is incorporated for voice and video calling features using Zegocloud. It allows direct peer-to-peer communication between users.
Third-Party Services:

Key Modules and Interactions:

User Management:
Users interact with the frontend to create profiles, log in, and manage their accounts. Authentication modules verify user identities and ensure secure access.

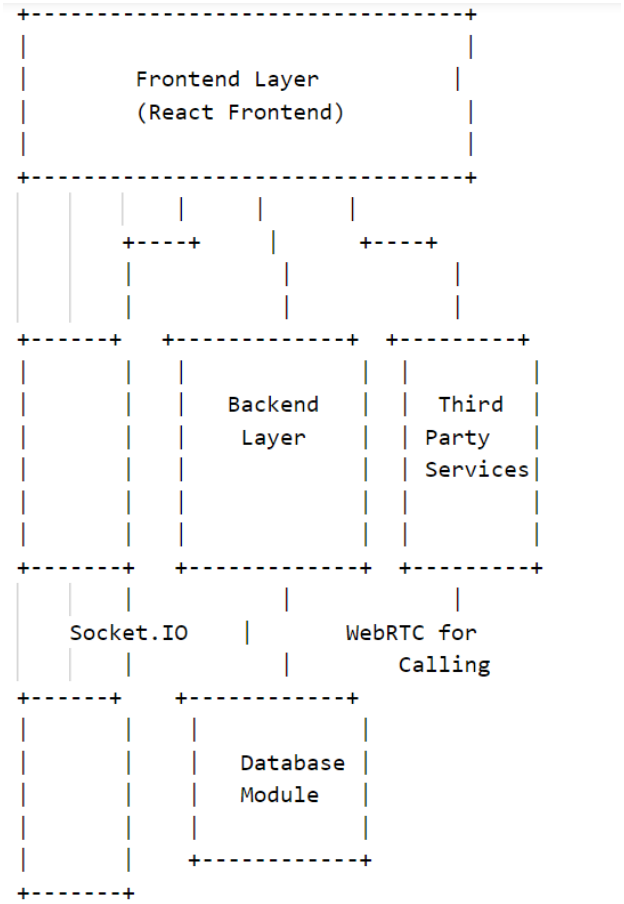
Messaging Module:
Users send and receive messages in real-time.
The frontend communicates with the backend via WebSocket connections provided by Socket.IO. Messages are stored in the database for message history.

Calling Module (WebRTC): Zegocloud
Users initiate voice and video calls. WebRTC handles the direct communication between callers, including media streaming.

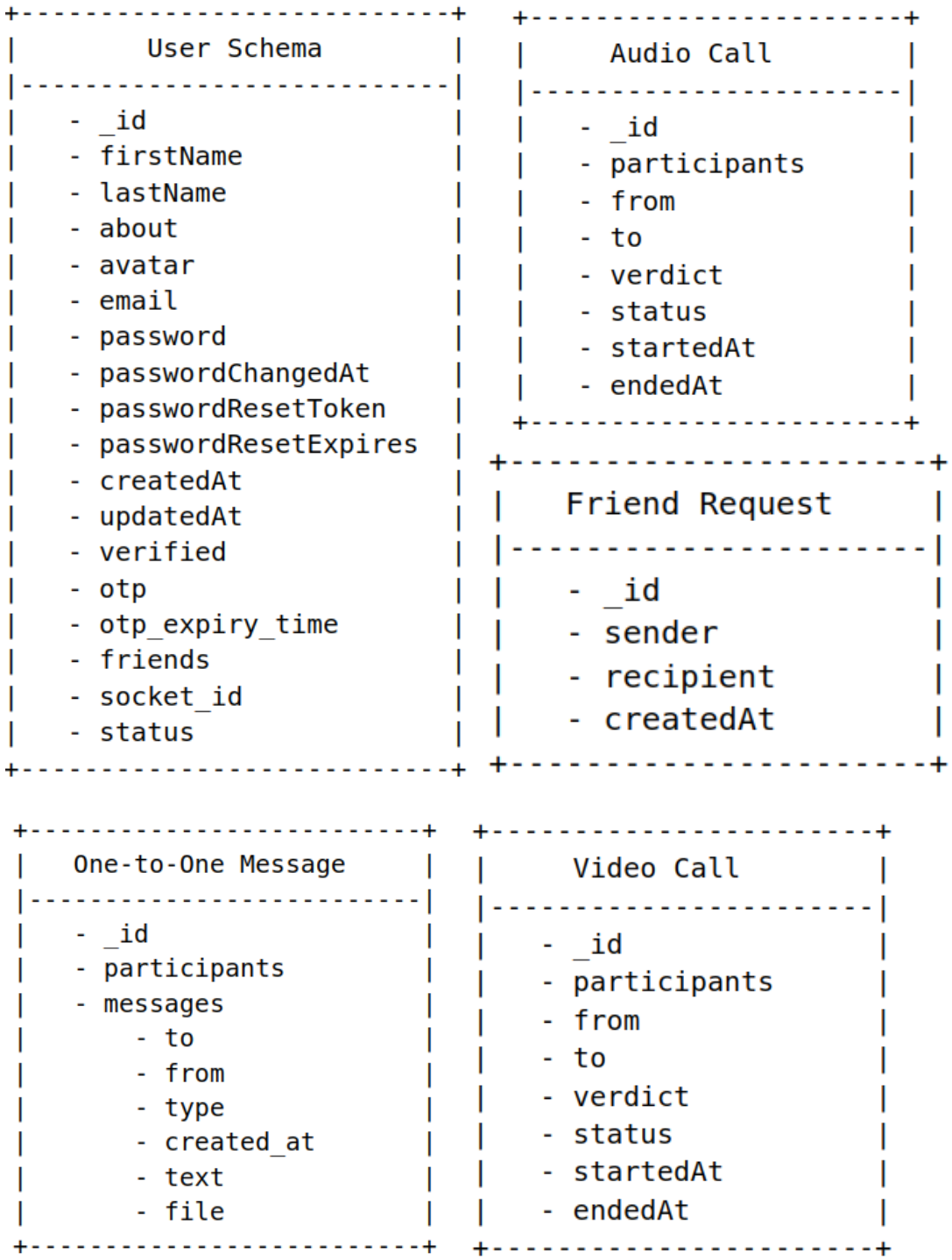
Database Module:
Stores user profiles, chat messages, and other application data. Backend services interact with the database using appropriate database drivers.

Architecture Diagram:

A typical architecture diagram for this chat application might look like this:

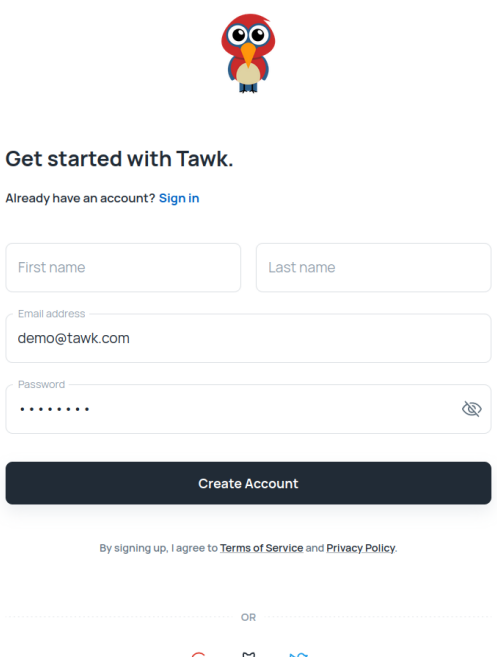


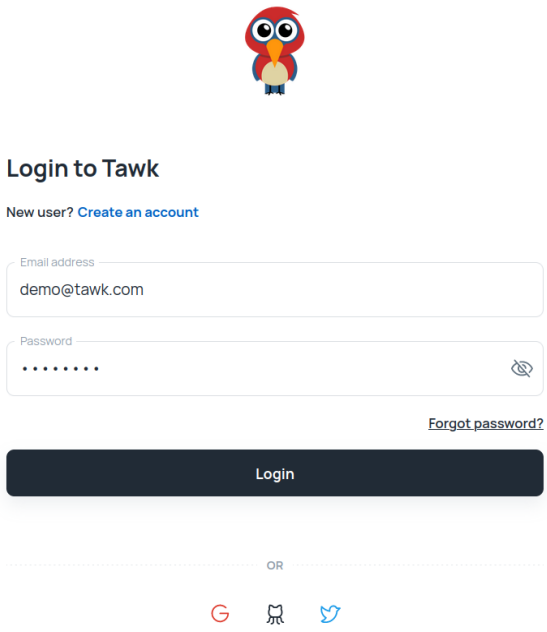
Database Schema:

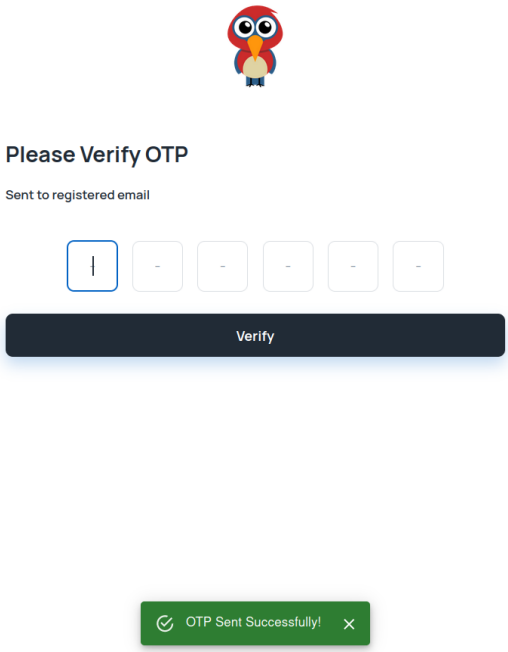


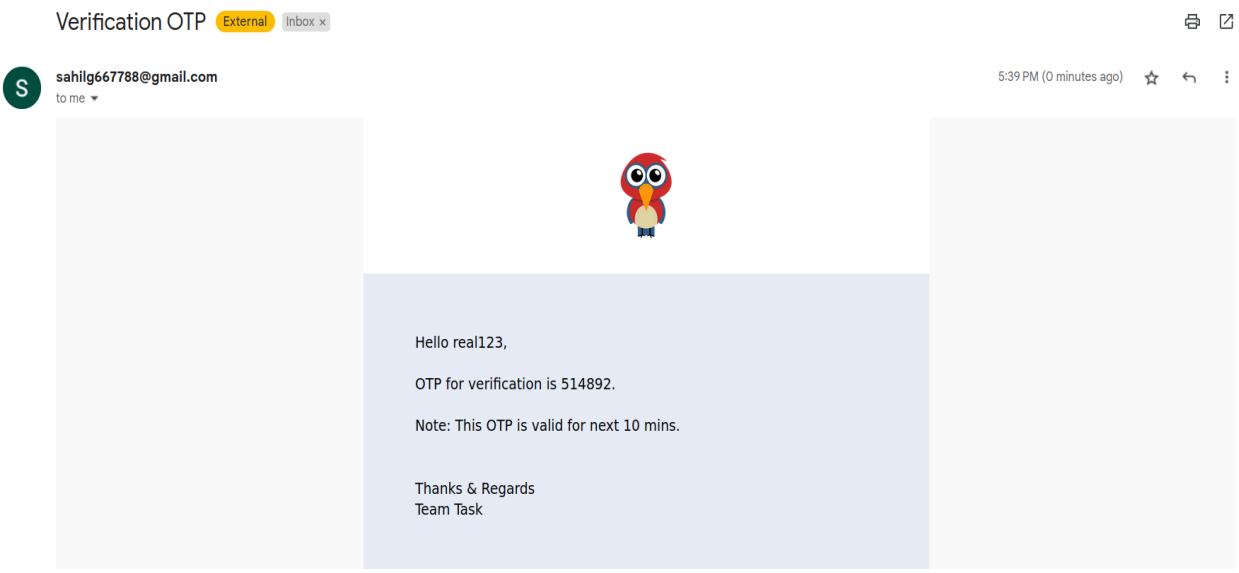
Demo :

Authentication Page:

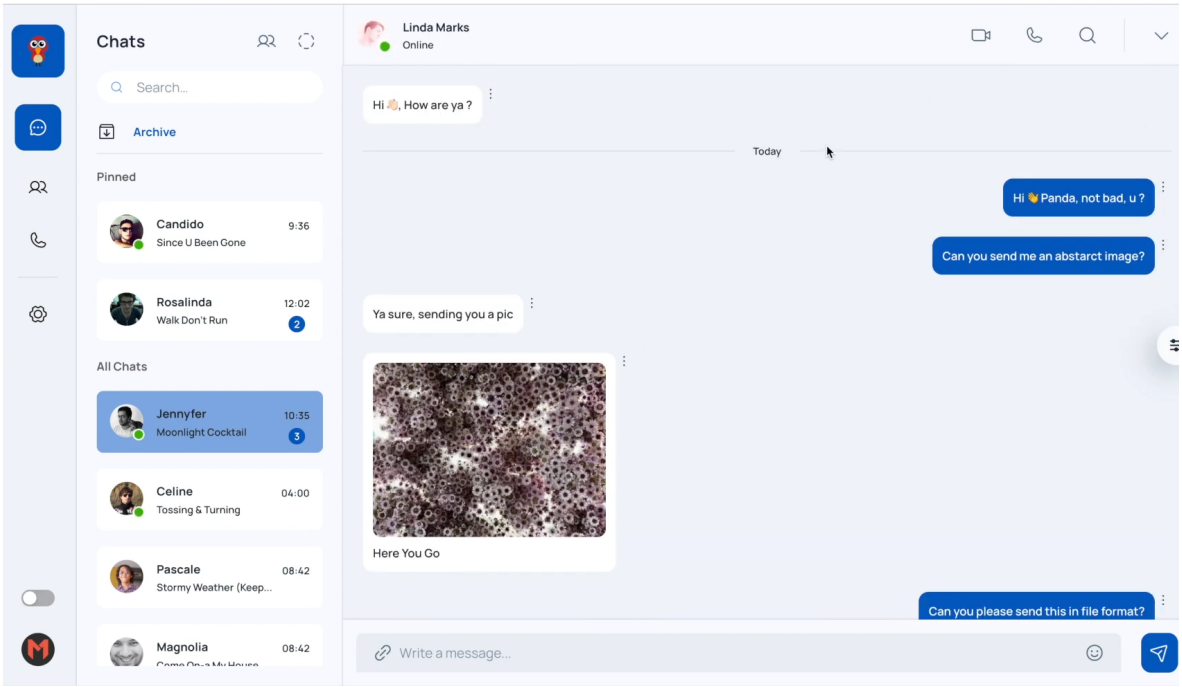


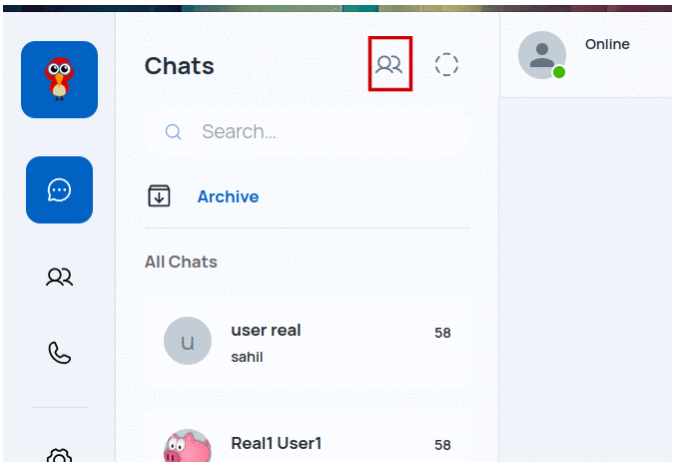


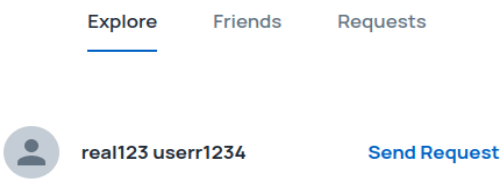


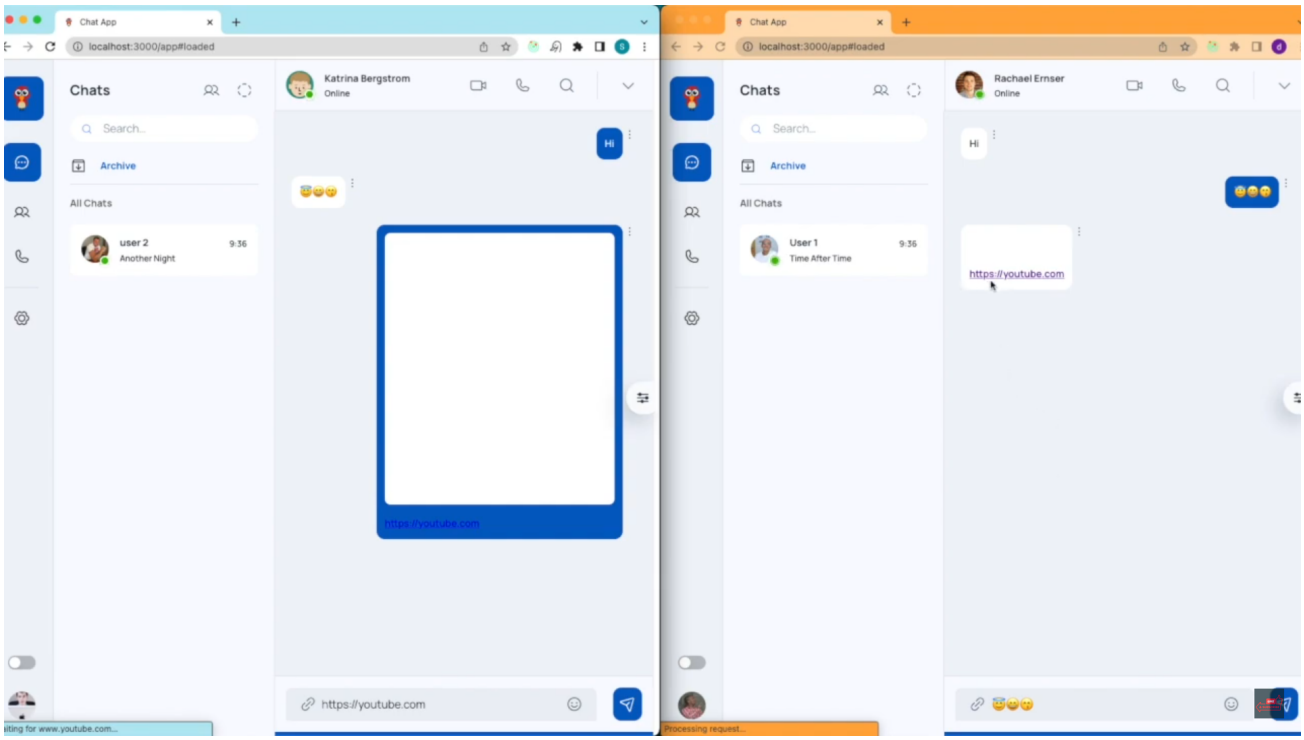

Verification OTP
External
Inbox ×
sahilg667788@gmail.com
to me
5:39 PM (0 minutes ago)
Hello real123,
OTP for verification is 514892.
Note: This OTP is valid for next 10 mins.
Thanks & Regards
Team Task

Home Page:

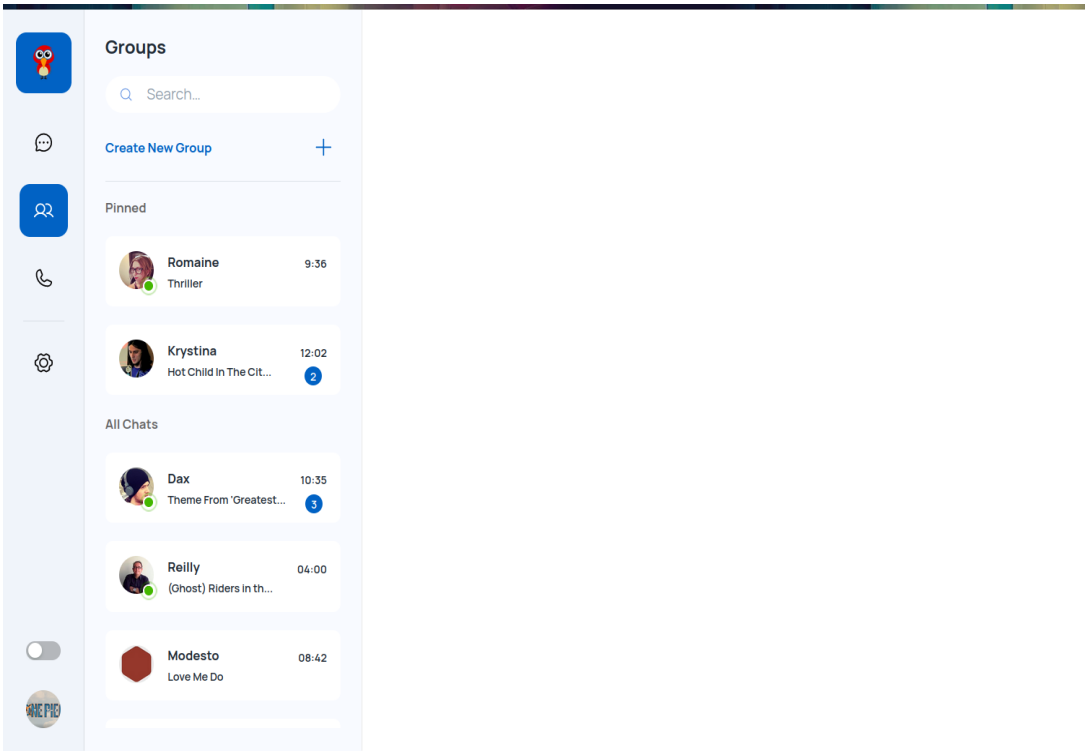

Chats
Search...
Archive
Pinned
Candido
9:36
Since U Been Gone
Rosalinda
12:02
Walk Don't Run
2
All Chats
Jennyfer
10:35
Moonlight Cocktail
3
Celine
04:00
Tossing & Turning
Pascale
08:42
Stormy Weather (Keep...
Magnolia
08:42
Linda Marks
Online
Hi, How are ya ?
Today
Hi Panda, not bad, u ?
Can you send me an abstarct image?
Ya sure, sending you a pic
Here You Go
Can you please send this in file format?
Write a message...

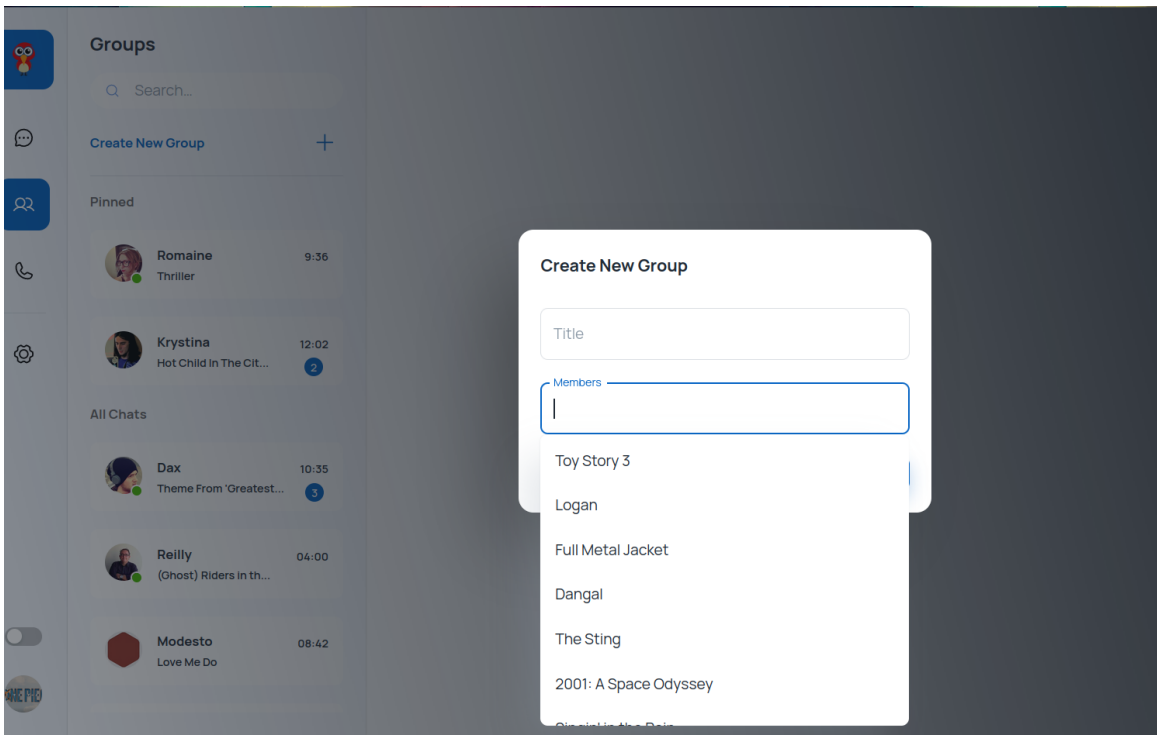

Chats
Online
Search...
Archive
All Chats
user real
sahil
58
Real1 User1
58


Explore
Friends
Requests
real123 userr1234
Send Request

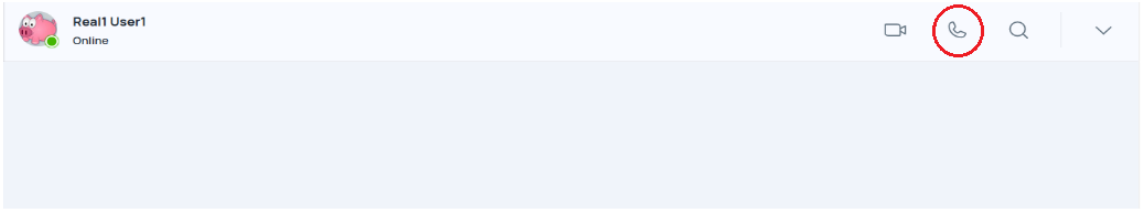

Chats
Katrina Bergstrom
Online
Archive
All Chats
user 2
Another Night
9:36
Hi
https://youtube.com
Chats
Rachael Ernser
Online
Archive
All Chats
User 1
Time After Time
9:36
Hi
https://youtube.com

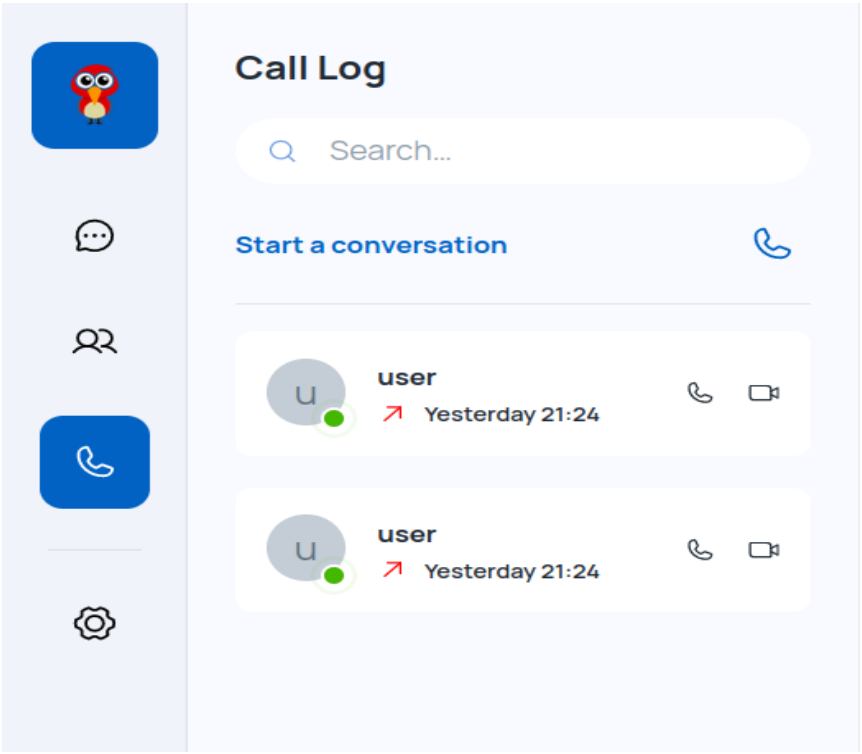
Groups:

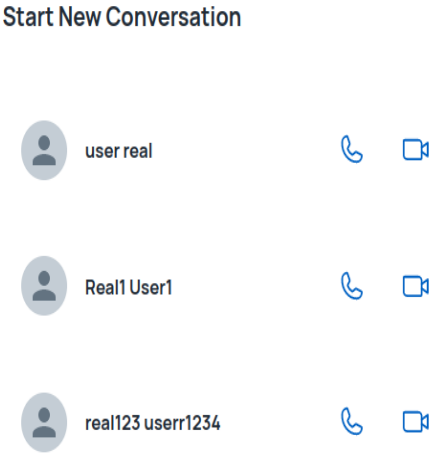

Groups
Search...
Create New Group
Pinned
Romaine
Thriller
9:36
Krystina
Hot Child In The Cit...
12:02
2
All Chats
Dax
Theme From 'Greatest...
10:35
3
Reilly
(Ghost) Riders in th...
04:00
Modesto
Love Me Do
08:42


Groups
Search...
Create New Group
Pinned
Romaine
Thriller
9:36
Krystina
Hot Child In The Cit...
12:02
2
All Chats
Dax
Theme From 'Greatest...
10:35
3
Reilly
(Ghost) Riders in th...
04:00
Modesto
Love Me Do
08:42
Create New Group
Title
Members
Toy Story 3
Logan
Full Metal Jacket
Dangal
The Sting
2001: A Space Odyssey

Calling:

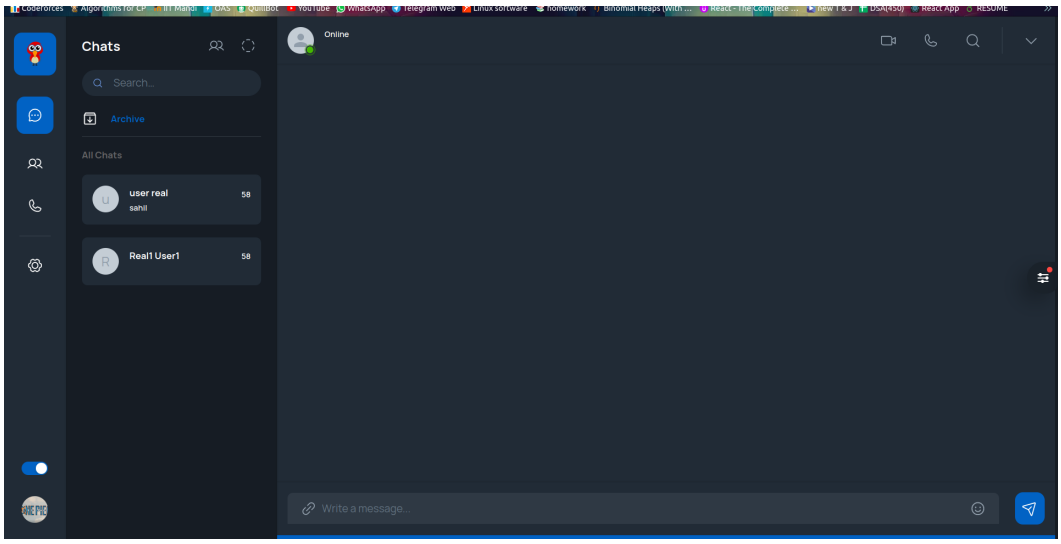

Real1 User1
Online


Call Log
Search...
Start a conversation
u
user
Yesterday 21:24
u
user
Yesterday 21:24

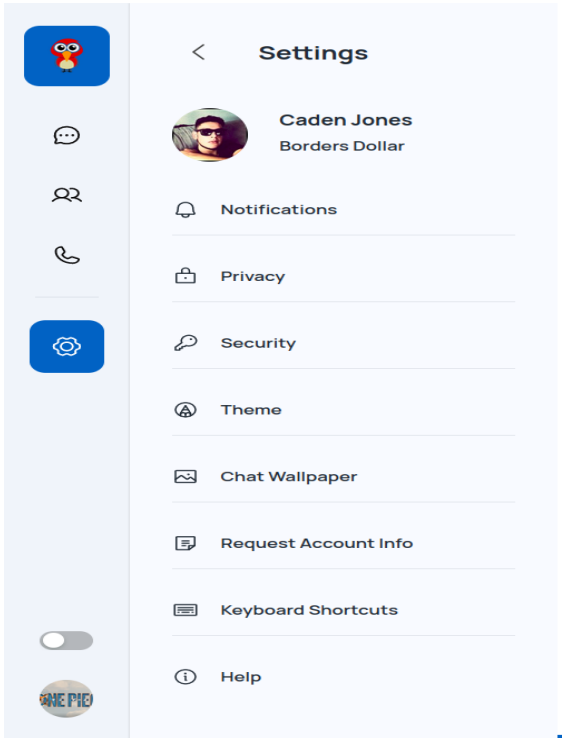

Start New Conversation
user real
Real1 User1
real123 userr1234

Optional features:

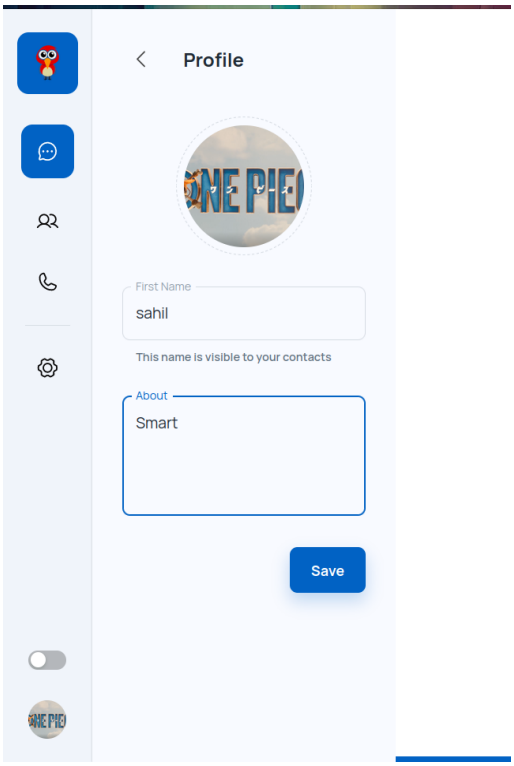
Themes:



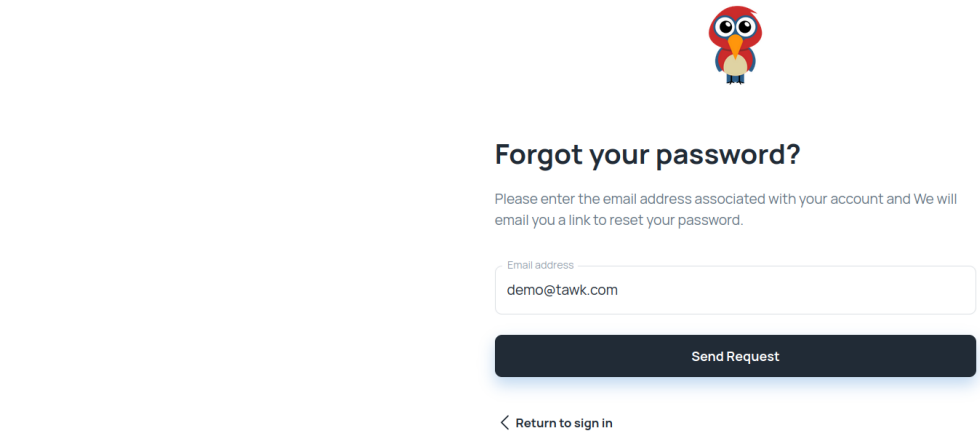
Settings:

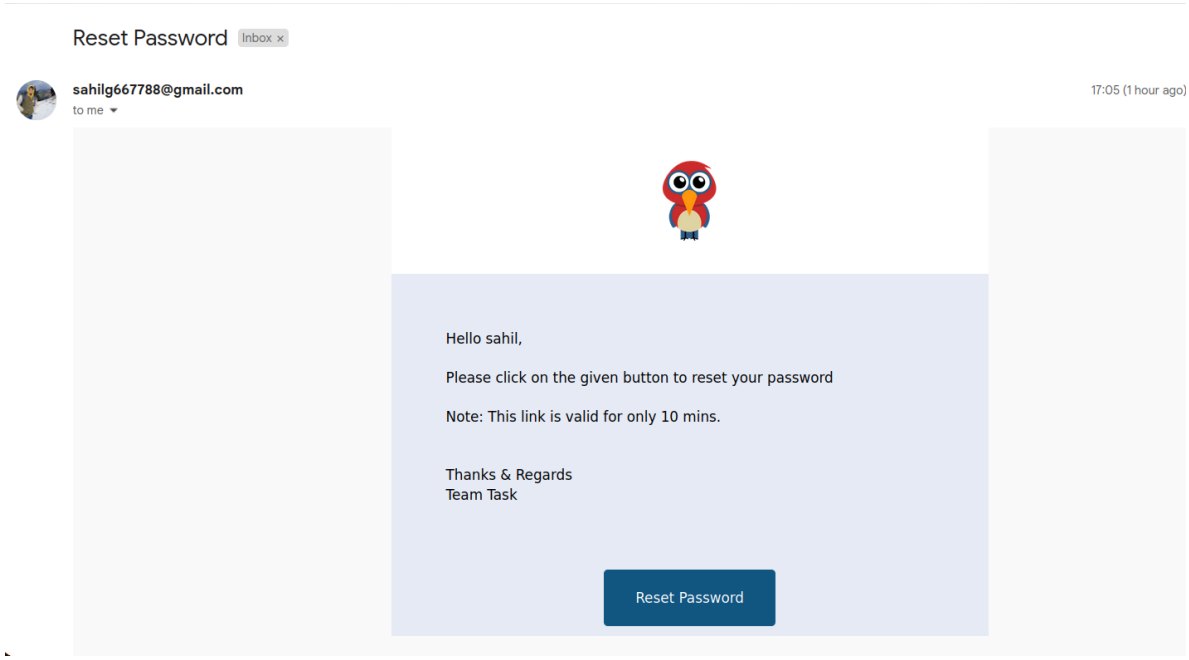


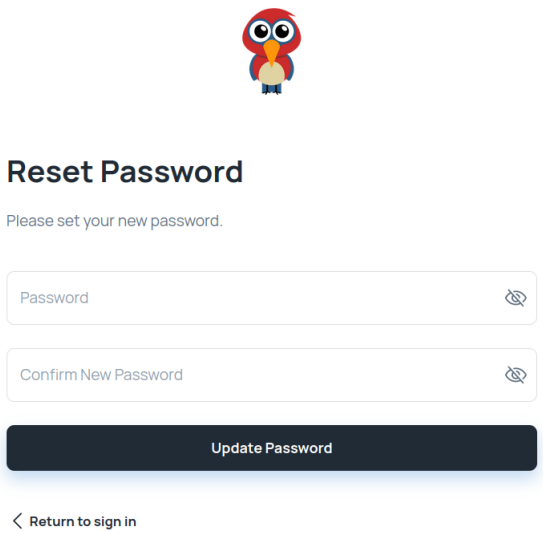
Profile:


Profile
First Name
sahil
This name is visible to your contacts
About
Smart
Save

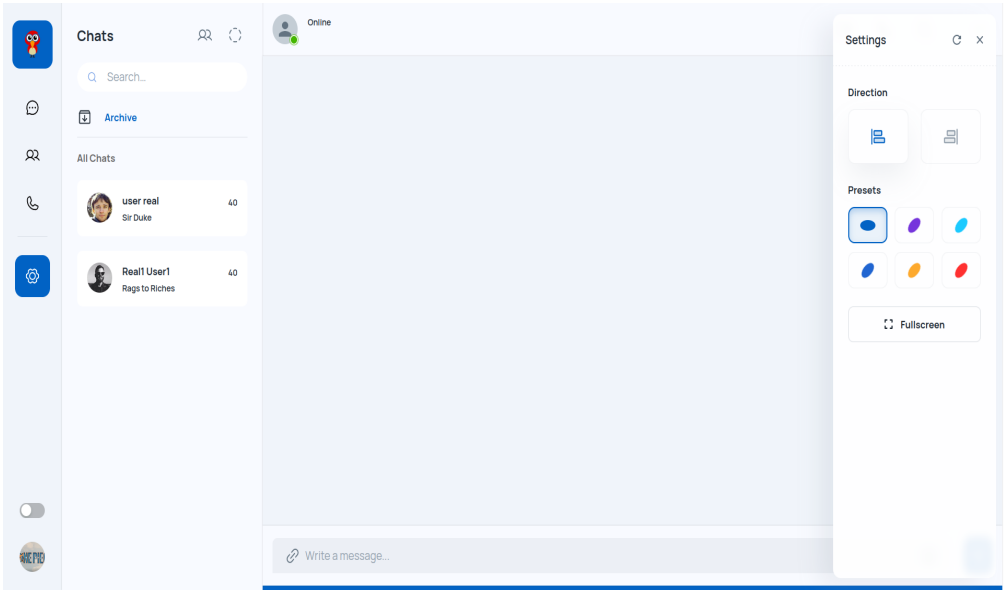
Reset password:

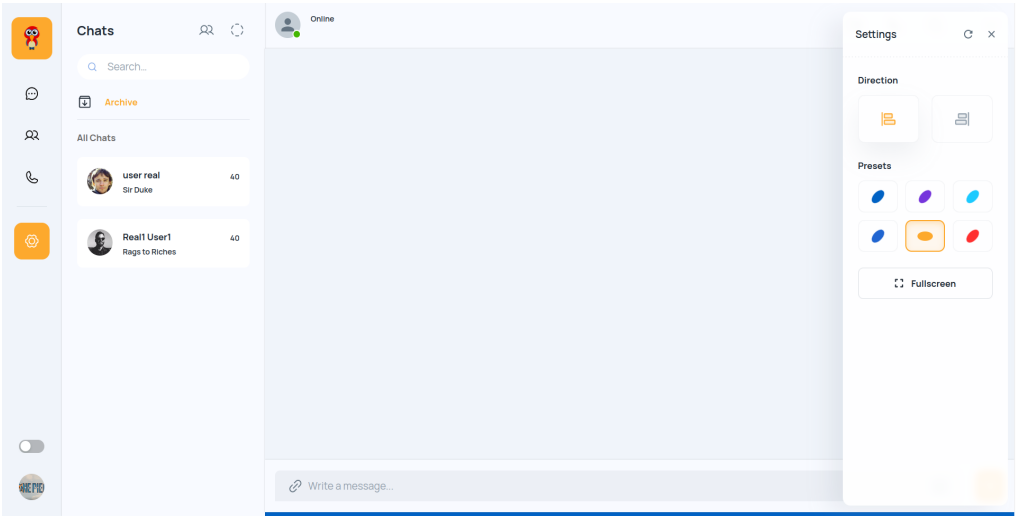






User customization options





User Roles and Permissions :

1. Admin:

- Can manage user accounts, including creating, updating, or deleting them.
- May have access to system-level settings and configurations.
- Can moderate and monitor chat rooms or group conversations.
- Has the authority to enforce community guidelines and policies.
- Typically has access to user analytics and system logs.

2. Regular User:

- Can create an account and log in.
- Has the ability to initiate one-on-one chats.
- Can join or create group chats.
- Can send and receive text messages, media, documents, and links within chats.
- May have the option to customize their profile and settings.
- Can invite friends or contacts to the application.
- Access to chat history and message search functionality.

Deployment Instructions:

These step-by-step instructions will guide you through the process of setting up and running the prototype. Please follow each step carefully to ensure a successful deployment.

Setting up the Development Environment:

Install Node.js:
Ensure you have Node.js installed on your machine. You can download it from Node.js website. Verify the installation by running the following commands in your terminal:

- node -v
- npm -v

Database Configuration:
Begin by accessing MongoDB Cloud and establishing a cluster, ensuring a connection with the database through the supplied credentials - both the database password and URL. Subsequently, integrate these credentials into the `config.env` file for streamlined configuration. This pivotal step ensures an efficient and secure database setup for your application, contributing significantly to the overall report.

Download and configure the source code:
Clone the repository from the url given above then navigate to the project directory, install the required dependencies using npm i command in the terminal

Running the web application:
In development mode: npm run dev
In production mode : npm run build, npm start

Accessing the application:
Open your web browser and navigate to the application's URL.
In development mode, it's typically http://localhost:3000 .

Security Considerations:

1) JWT Tokens for OTP Generation:

Secure OTPs with digitally signed JWTs.
JWTs with defined expiration times.

2) Two-Factor Authentication (2FA):

Extra layer during user registration.
Requires second verification factor.

Conclusion:

In summary, this project marks a significant achievement in delivering a secure and user-friendly chat application. My focus on security, including the use of JWT tokens and Two-Factor Authentication (2FA), ensures user data protection.

The clear deployment instructions make it easy for users to set up the prototype. I will remain dedicated to improving and evolving the application based on the user feedback.