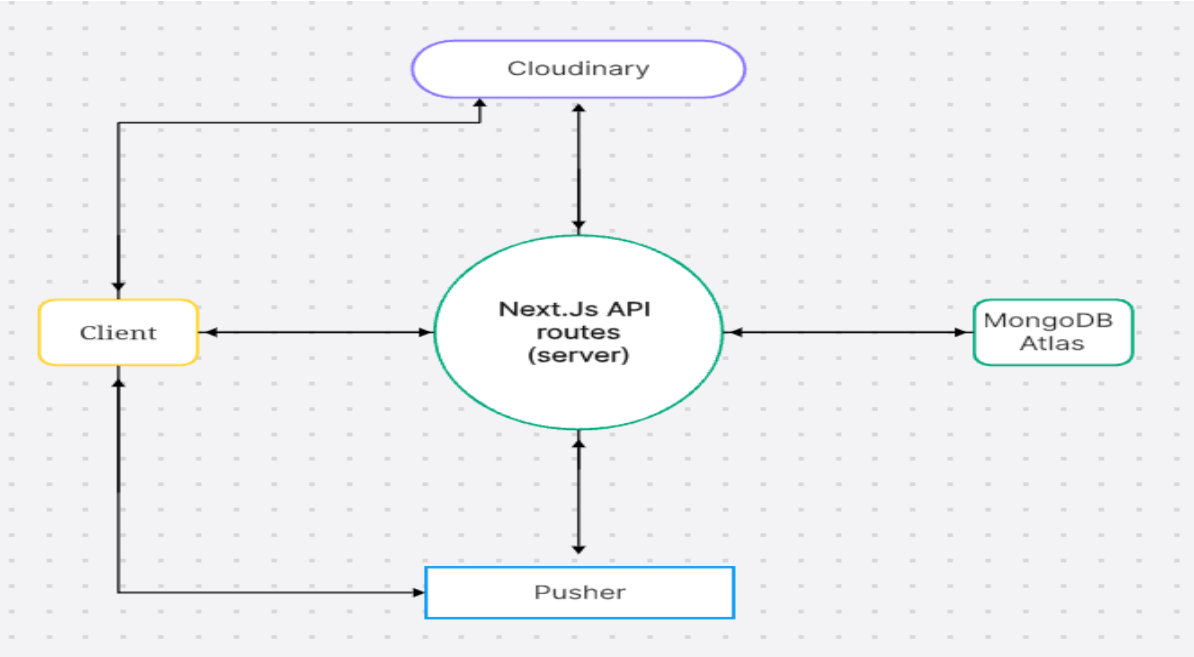
InfiTalks: Messaging Web App

Project Overview

InfiTalks is a real-time web messaging platform built with Next.js. It allows users to send messages, create group chats, and manage media using a cloud-based storage service. The application features real-time communication with **Pusher** and secure authentication through **NextAuth**.

System Architecture

The architecture of InfiTalks is designed to support real-time messaging, secure user authentication, and cloud-based image management. The application follows a modern web application architecture, leveraging Next.js for both the frontend and API routes, with MongoDB Atlas as the database and Pusher for real-time communication.



Clients interact directly with Pusher to receive and send real-time updates, and with Cloudinary for uploading media. Next.js API Routes manage interactions between the client and MongoDB Atlas for data operations, Pusher for real-time messaging, and Cloudinary for media management.

MongoDB Atlas is used by Next.js API Routes to store and retrieve application data.

Pusher provides real-time communication services and is connected to both Next.js API Routes (to publish events) and the Client (to receive real-time updates).

Key Components:

Frontend:

- Built with **Next.js**, which enables both client-side and server-side rendering (SSR) for optimal performance.
- The frontend handles user interactions, displays messages, and manages file uploads.

Backend (Next.js API Routes):

- Next.js API routes are used for handling user registration, authentication, sending messages, and managing group chats.
- RESTful API endpoints for CRUD operations on user, chat, and message data.

Database (MongoDB Atlas).

- **Users**: Stores user details like email, username, profile picture, and authentication tokens.
- **Chats**: Contains details of chat groups, including members, group names, and metadata.
- **Messages**: Stores individual chat messages with information about the sender, timestamp, and media links.

Real-Time Messaging (Pusher):

- Pusher is used to deliver real-time updates to users when a new message is sent in any chat.
- Pusher broadcasts to all participants in the relevant chat room.

Media Management (Cloudinary):

- Cloudinary handles image uploads, storage, and retrieval.
- Users can upload profile pictures or send images in chats.

Authentication (NextAuth.js):

- NextAuth.js manages user authentication via OAuth providers.
- It ensures secure session management with JWTs and session cookies.

Data Flow

The data flow outlines the sequence of interactions between the client, server, database, and external services.

User Authentication:

- Users sign up or log in using NextAuth.js.
- On successful login, NextAuth issues a session token, which is stored as a secure cookie.

Sending a Message:

- A user writes a message and sends it from the chat window.
- The frontend sends the message to the backend API endpoint (/api/messages).
- The backend processes the message and stores it in MongoDB.
- A real-time update is triggered via Pusher to notify other participants.

Receiving Real-Time Messages:

- When a message is received in the chat, Pusher broadcasts the message to all subscribed users.
- The frontend listens for the event and updates the chat interface in real-time.

Uploading Media:

- A user uploads an image (profile picture or chat attachment).
- The image is uploaded to Cloudinary, and the returned URL is stored in the MongoDB.
- The image link is broadcasted via Pusher for other users to see.

Technology Stack

Frontend:

- **Next.js**: A React framework that allows both server-side and static site generation, optimizing performance and SEO.

Backend:

- **Next.js API Routes**: Provides server-side APIs to handle requests, send and receive data, and manage user sessions.

Database:

- **MongoDB Atlas**: A NoSQL cloud database used to store user profiles, chat data, and messages.

Real-Time Communication:

- **Pusher**: Real-time messaging service that allows users to receive instant updates when a message is sent or received.

Authentication:

- **NextAuth.js:** Handles OAuth2-based authentication (e.g., Google OAuth) and manages session tokens and secure cookies.

Cloud Storage:

- **Cloudinary**: A cloud-based image management service used to store and serve user-uploaded media.

API Design

Endpoints:

/api/auth:

- Handles user authentication and session management.
- Uses NextAuth.js for OAuth providers like Google.

/api/messages:

- POST: Used to send a message.
- GET: Fetches messages for a given chat.

/api/chats:

- GET: Fetches chat details, including members and messages.
- POST: Creates a new chat.

Scalability Decisions (Why?)

Real-Time Messaging with Pusher:

- **Why Pusher**: It simplifies real-time communication with WebSocket technology and ensures low latency.
- Benefit: Pusher abstracts WebSocket management, making it easier to implement and scale.
- Pusher ensures low-latency message delivery and is scalable across multiple users and large chat rooms

Next.js API Routes:

- **Why Next.js**: It provides a unified solution for both frontend and backend with server-side rendering capabilities, which improves performance and SEO.

MongoDB Atlas:

- **Why MongoDB**: A NoSQL database is ideal for flexible, document-based data storage, which is useful for storing chat messages and user information.

NextAuth.js for Security:

- **Why NextAuth.js**: It simplifies OAuth2 integration and session management, ensuring secure authentication flows.

Deployment

- I used Vercel to deploy the app.
- You can test and use the app by clicking the link: InfiTalks.

Conclusion

InfiTalks is designed to be a scalable, real-time messaging platform with a focus on performance and security. Future enhancements could include video calls, push notifications, and enhanced scalability.

NOTE: For more details regarding project and its setup navigate to readme.md file on my Github.

About Me:

Name: **Anil Patel**
University: **Indian Institute of Technology Goa**
Department: **Computer Science and Engineering**
Email: anil.patel.22031@iitgoa.ac.in

****