System Design Document for Messenger

**Messaging-Service-IBY**

This is my repo for the Assignment for the role of Software Developer at IBY for Summer Internship 2025.

A Next.js Web Application which provides users messaging service. User should initially get registered and logging into the application can avail:

* Profile Picture upload
* Sending and receiving text messages and photo messages between users.
* Creation of Group chat
* Real time message updates
* Personalised AI Powered Chatbot

Clone this repository, extract it and then open it in VS Code.

Then open terminal in VS Code and type the following:

* cd .\Messaging-Service-IBY-main\
* npm install
* npm run dev

The npm install will install all the dependencies required and then the npm run dev will create a local server.

Run the local server and the app is ready to use.

1. **The Brain symbol in the Desktop Sidebar will redirect to https://solutions.imbesideyou.com/**
2. **The Robot Symbol below that is the Chatbot option, it can also be used from the chat button in the bottom right corner**

Note: To view the functionalities, kindly visit the Output\_Images folder in the repository for Web App Snapshots.

**Technology used**

Frontend ---> Next.js

Backend ---> Node.js

* Prisma
* Next-Auth
* Pusher
* Bcrypt

NoSQL database ---> MongoDB

**Future Implementations**

Add Video & Audio calling feature

Improve AI functionalities into our web app

**Details of Developer**

Name: Vishwaa P S

University: Indian Institute of Technology Palakkad

Department: Bachelor's in Computer Science and Engineering

**Technologies Used in the Project**

**1. Next.js**

* **Server-Side Rendering (SSR):** Renders HTML on the server to enhance performance and SEO.
* **Static Site Generation (SSG):** Pre-renders pages at build time for improved performance.
* **API Routes:** Enables the creation of API endpoints directly within the application.
* **Built-In Routing:** Utilizes a file-based routing system for easy page management.
* **Automatic Code Splitting:** Loads only necessary parts of the code to improve load times.

**2. NextAuth.js**

* **Simplified Authentication:** Facilitates integration with authentication providers like Google and GitHub.
* **Built-In Sessions:** Manages user sessions and authentication states effectively.
* **Customizable:** Supports custom databases and various authentication flows.
* **Secure:** Implements encrypted sessions and secure token handling for user data protection.

**3. Prisma**

* **Type-Safe Database Access:** Generates TypeScript types for early error detection.
* **Efficient Database Queries:** Provides a powerful query engine for optimized database interactions.
* **Database Migrations:** Eases schema management and evolution over time.
* **Intuitive API:** Offers an easy-to-use API that reduces boilerplate code and accelerates development.

**4. bcrypt**

* **Secure Password Hashing:** Employs a computationally intensive algorithm for password security.
* **Salting:** Automatically adds a salt to passwords before hashing for enhanced security.
* **Ease of Use:** Features a straightforward API for password hashing and comparison.

**5. Pusher**

* **Real-Time Capabilities:** Enables live notifications, chat, and real-time updates.
* **Easy Integration:** Simplifies the addition of real-time features with minimal setup.
* **Scalability:** Manages the infrastructure and scaling for real-time functionalities.
* **Cross-Platform Support:** Compatible with various platforms and languages.

**Additional Dependencies**

* **@headlessui/react & @headlessui/vue:** Provides headless UI components for React and Vue applications.
* **@next-auth/prisma-adapter:** Integrates NextAuth.js with Prisma for authentication.
* **@tailwindcss/forms:** Tailwind CSS plugin for enhancing form styling.
* **axios:** Promise-based HTTP client for making HTTP requests.
* **bcrypt & bcryptjs:** Libraries for secure password hashing.
* **cloudinary & cloudinary-react:** Handles image and video management in React applications.
* **clsx:** Utility for conditional application of CSS class names.
* **date-fns:** Utility library for date manipulation and formatting.
* **dotenv:** Loads environment variables from a .env file.
* **ejs:** Templating engine for generating HTML.
* **express:** Web framework for building server-side applications in Node.js.
* **lodash:** Utility library for working with arrays, objects, and functions.
* **mongoose:** MongoDB object modeling tool for schema-based data access.
* **next:** React framework with built-in support for SSR and SSG.
* **next-cloudinary:** Integrates Cloudinary with Next.js applications.
* **next-superjson-plugin:** Adds SuperJSON support to Next.js for better data serialization.
* **openai:** Client library for interacting with OpenAI's API.
* **pusher & pusher-js:** Libraries for implementing real-time communication features.
* **react:** Core library for building user interfaces.
* **react-dom:** Provides DOM-specific methods for React.
* **react-hook-form:** Manages form state and validation in React.
* **react-hot-toast:** Displays toast notifications.
* **react-icons:** Offers popular icons as React components.
* **react-select:** Customizable select input component for React.
* **react-spinners:** Provides spinner components for loading indicators.
* **zustand:** State management library with a simple API.

**DevDependencies**

* **@types/xyz:** TypeScript definitions for various libraries.
* **eslint & eslint-config-next:** Tools for linting code and ensuring quality and consistency.
* **postcss & tailwindcss:** Tools for CSS processing and utility-first styling.
* **prisma:** ORM tool for database interactions.
* **typescript:** Adds TypeScript support for type safety and development efficiency.