Messaging Service Prototype Documentation

November 30, 2024

Contents

1 Project Overview

This project involves developing a real-time messaging service prototype with features like user registration, one-to-one messaging, group chats, and optional functionalities such as AI-powered chatbots and audio/video calling.

2 System Design

2.1 Architecture

- **Frontend:** Next.js (web app) or Ionic React (mobile app).
- **Backend:** REST APIs developed in Python (FastAPI) or Node.js (Express).
- **Database:** SQL (PostgreSQL) or NoSQL (MongoDB).
- **Real-Time Updates:** WebSockets via Socket.IO or FastAPI WebSocket.
- **AI Chatbot (Optional):** Integrated with OpenAI's GPT-4 API.
- **Deployment (Optional):** Hosted on Heroku or AWS.

2.2 Key Features

- **Core Features:**
  - User registration and authentication (JWT-based).
  - Sending and receiving text messages.
  - Group chat functionality.
  - Real-time message updates.
- **Optional Features:**
  - AI-powered chatbot.
  - Audio and video calling.

2.3 Design Choices

- **Frontend Framework:** Next.js for SEO and SSR; Ionic React for cross-platform mobile apps.
- **Database Selection:** SQL for structured data; NoSQL for flexibility.
- **Atomic Design Principles:** Ensuring scalable and reusable UI components.

3 APIs

- **Authentication:**
  - `POST /register`
  - `POST /login`
- **Messaging:**
  - `GET /messages/:userId`
  - `POST /messages`
- **Groups:**
  - `POST /groups`
  - `GET /groups/:groupId`

4 Setup Instructions

4.1 Prerequisites

- Node.js
- Python (if using FastAPI)
- Docker (optional for deployment)

4.2 Installation

1. Clone the repository:

```
git clone <repository_url>
cd messaging-service
```

2. Install dependencies:
   - **Backend:**

```
cd backend
npm install  # For Node.js
# OR
pip install -r requirements.txt  # For Python
```

   - **Frontend:**

```
cd frontend
npm install
```

3. Configure environment variables: Create a `.env` file in the root directory:

```
DATABASE_URL=<database_connection_string>
JWT_SECRET=<your_secret_key>
OPENAI_API_KEY=<your_api_key>  # Optional for AI chatbot
```

4. Run the application:

   - **Backend:**

     ```
     npm start  # For Node.js
     # OR
     uvicorn main:app --reload  # For Python
     ```

   - **Frontend:**

     ```
     npm run dev
     ```

4.3 Deployment (Optional)

Deploy the app using Docker:

```
docker-compose up --build
```