

System Design Document

Project- Messaging Service Web App

Author- Sohum Bansal

University- Indian Institute of Technology(IIT) Patna

Department- Civil and Environmental Engineering

Roll No.- 2201CE60

Institute Mail-ID- 2201ce60_sohum@iitp.ac.in

Personal Mail ID- sohumbansal147@gmail.com

Github Repo Link- <https://github.com/sohumbansal/ChatApp>

Demo Video- [Demo video](#)

Overview

This document describes the **system design** for a messaging service app named ChatApp, which is made in Next Js and allows users a **real-time** text chat experience. Users are required to sign up/sign in using their **email** and **password**. The app provides a simple, easy-to-use UI. The database and authentication are handled in **Firebase**.

This document lists all the technologies and dependencies and explains their use. It provides a detailed description of the processes, data handling, and design principles.

Table of Content

- Design Principles
- System Architecture
- Workflow Database
- Dependencies used
- Use Cases
- Future Developments

Design Principles

- The App is made in next.js. Next.js provides the ability to split a website into small parts called components and reuse them.
- The Styling is done in CSS.
- This web app has one main folder -Src. The src folder has an app folder divided into different folders for accessible and reusable code.

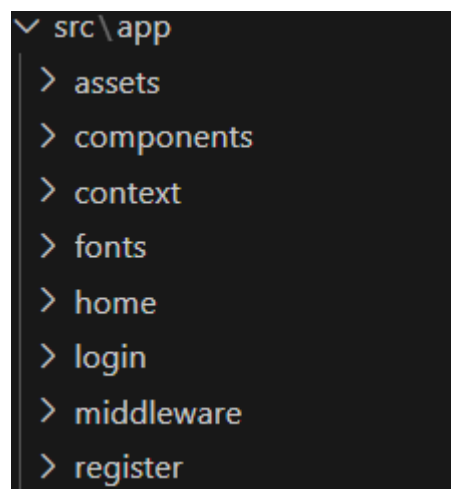


Fig: project structure

The app folder is divided into -

1. Components - Each page is divided into small react components interacting with the user.
2. Context - React context to use a value as a global value in the whole app.
3. Pages- The home, login, and register pages consist of pages.js for that particular use.
4. Each page has been written in a structured way. For example-

Parent: AuthContextProvider

Child: ChatContextProvider

Child: Form (input, button)

5. A snippet of the style.css file which has also been written in a structured manner.

```
src > app > # style.css > .formContainer .formWrapper
1  .formContainer {
2      background-color: #17B169;
3      height: 100vh;
4      display: flex;
5      align-items: center;
6      justify-content: center;
7      opacity: 0.8;
8
9      background-position: 16px 0, 16px 0, 0 0, 0 0;
10     background-size: 16px 16px;
11     background-repeat: repeat;
12 }
13 formContainer formWrapper
```

System Architecture

The system uses Next.js to provide a server-side rendering framework with React components. Each component serves a specific purpose, allowing interaction with the user interface.

The web app consists of several pages, components, and contexts, organized as follows:

- **Pages-**

1. **Register:**

- Users create an account with an email and password.
- If the email already exists, the system prevents duplicate account creation.

2. **Login:**

- After account creation, users log in using their credentials.
- Once logged in, the user is redirected to the homepage.

3. **Home:**

- Accessible only after logging in.
- Displays all the user's chats and provides real-time chat functionality, divided into a sidebar and chat window.

- **Components**

1. **Sidebar:**

- The left pane of the home page displaying user details, a list of all users who have existing chats, and a search bar.
- The sidebar is divided into **Navbar**, **Search**, and **Chats** components.

2. **Chat:**

- Displays the current conversation along with input fields for sending new messages.
- Divided into **Messages** and **Input** components.

3. **Navbar:**

- Part of the sidebar, showing the logged-in user's details and a sign-out option.

4. **Search:**

- Allows users to search for other users by name.

5. **Chats:**

- Displays a list of users with whom the current user has existing conversations.

6. **Messages:**

- Shows the current conversation between two users. The chat updates in real-time as messages are sent and received.

7. **Input:**

- Provides an input field for typing messages, with a button to send them.

8. **Message:**

- Each message contains details like sender, receiver, and content.

- **Contexts**

1. **AuthContext:**

- Tracks whether the user is logged in.
- If the user is authenticated, their details are stored and available globally within the app.

2. **ChatContext:**

- Tracks whether two users have previous chat history.
- Loads and displays the chat history if available.

Database

This solution uses **Firebase Firestore**, a NoSQL cloud database, to store users and chat data. Firestore offers **real-time synchronization**, ensuring chat messages are instantly updated across all users. Its **scalable, flexible** structure—using documents and collections—makes it ideal for dynamic chat applications.

Firebase Cloud Functions are employed to handle server-side logic like real-time updates or interactions with Firestore.

Firestore's key benefits include real-time data sync, **scalability** and seamless integration with other Firebase services. Its document-oriented NoSQL model allows for efficient, dynamic data handling.

Dependencies

- Next.js: The server-side rendering, routing, and static page generation framework.
- Firebase: To handle user authentication, real-time data sync with Firestore, and file storage.
- React: The core React library used with Next.js.
- React-dom: For rendering React components in the browser.
- React-icons: For using icons like send, camera, and attach.
- Next-router: For client-side routing and navigation between different pages in Next.js.
- CSS: Used for styling the application.

Workflow-

Given below is the workflow of a one-to-one chat-

Use Cases

The following are some of the use cases for the messaging service app:

- A user can signup/sign in using their email and password
- A user can search for other users.
- A user can start a chat conversation with another user as well as multiple users at a time(group)
- A user can send and receive text messages in real time.

Future Developments:

The following are some of the features that are planned for future development:

- Option to attach file of any format.
- Voice and video calls.
- AI chatbot integration.
- Availability status.
- Message forwarding and deletion.
- Privacy features

Thank you

For any further assistance or queries, please contact Sohum Bansal at sohumbansal147@gmail.com