SOFTWARE DEVELOPMENT PRACTICES

WEB BASED CHAT APPLICATION

Prepared by
Harirajan S
Rogith P M
Jai Surya
S Selva Kumar
S Kaushik
(IT A&C)

Objective:

To Build an web based chat application so that all the peoples can get globally connected.

Users of the system:

1. All people

Functional Requirements:

- Build an application that connects people from different locations.
- The application should have a sign-in/sign-up page,profile page, communities page, chat page,cloud page,settings page,social page.
- This page should have provision to maintain a database of:
    - User id’s and password
    - Shared media
    - Chat history
- An integrated platform required for admin and people.

While the above ones are the basic functional features expected, the below ones can be nice to have add-on features:

- Multi-factor authentication for sign-in process.
- The account can easily transferred to mobile devices using QR code or OTP

Non-Functional Requirements:

1.Security

- End to End encryption
- App Platform – Username / Password – Based Credentials.
- Sensitive data has to be categorised and stored in a secure manners.
- Secure connections for transmissions of any data

2.Performance

- Peak Load performance
- eCommerce -< 3 sec

- Admin application < 2 sec
- Non-Peak Load Performance
- eCommerce <2 sec
- Admin Applications < 2 sec

3.Availability

- 99.99% Available

4.Standard Features

- Scalability
- Maintainability
- Usability
- Availability
- Failover

5.Logging and Auditing

- The system should support logging (app/web/DB) and auditing at all levels.

6.Monitoring

- Should be able to monitor via as-in enterprise monitoring tools

7.Cloud:

- The Solutions should be made Cloud-ready and should have a minimum impact when moving away to Cloud infrastructure

8.Browser Compatible:

- All Latest Browsers

9.Technology stack:

- Front End HTML,CSS,JS

- Server Side Spring Boot / .Net WebAPI/ Node Js
- Database MySQL or Oracle or MSSQL

10.Applications Assumptions:

1. The sign-in or sign-up page should be the first page rendered when the application loads.
2. Manual routing should be restricted by using AuthGuard by implementing the canActivate interface. For example, if the user enters
as http://localhost:8000/signup or http://localhost:8000/home the page should not navigate to the corresponding page instead it should redirect to the login page.
3. Unless logged into the system, the user cannot navigate to any other pages.
4. Logging out must again redirect to the sign-in or sign-up page.
5. To navigate to the admin side, you can store a user type as admin in the database with a username and password as admin.
6. Use admin/admin as the username and password to navigate to the admin dashboard

Validations:
1. Basic email validation should be performed.
2. Basic mobile validation should be performed.
3.Password validations should be performed.

Project Tasks:

API Endpoints:
USER:

Action URL Method Response:

Login /login POST true/false
Signup /signup POST true/false
Chats – Home /home GET Array of chats Methods
Call history – Home /home GET Array of Call history

Chat history /Statistics/{id} POST statistics of Chat history
Reels/Statistics/{id} POST statistics of reels
Liked/Statistics/{id} POST statistics of liked
Comments/Statistics/{id} POST statistics of comments
Followed/Statistics/{id} POST statistics of followed

FRONTEND:
CUSTOMER:

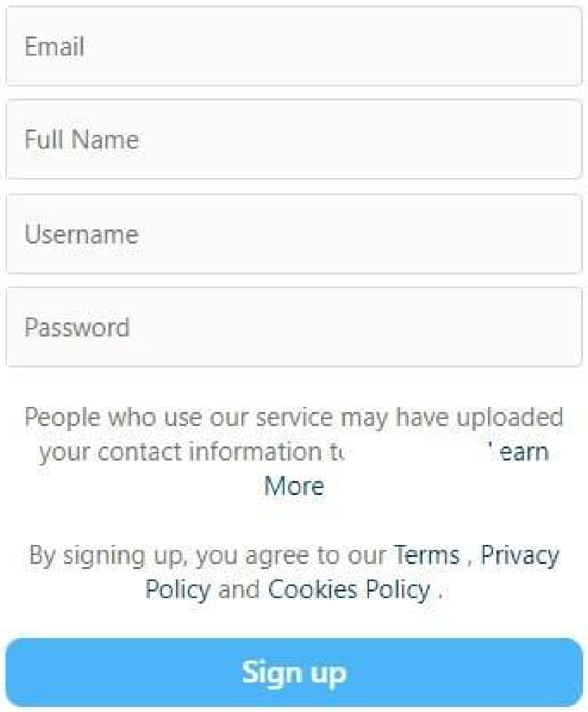
1. Auth: The customer can authenticate login and signup credential.

2. Register: The new customer has options to sign up by providing their basic details.

1. ids:
    - Email
    - Full name
    - Username
    - Password
    - Sign up button

2. API endpoint Url: http://localhost:8000/signup

3. Output screenshot:

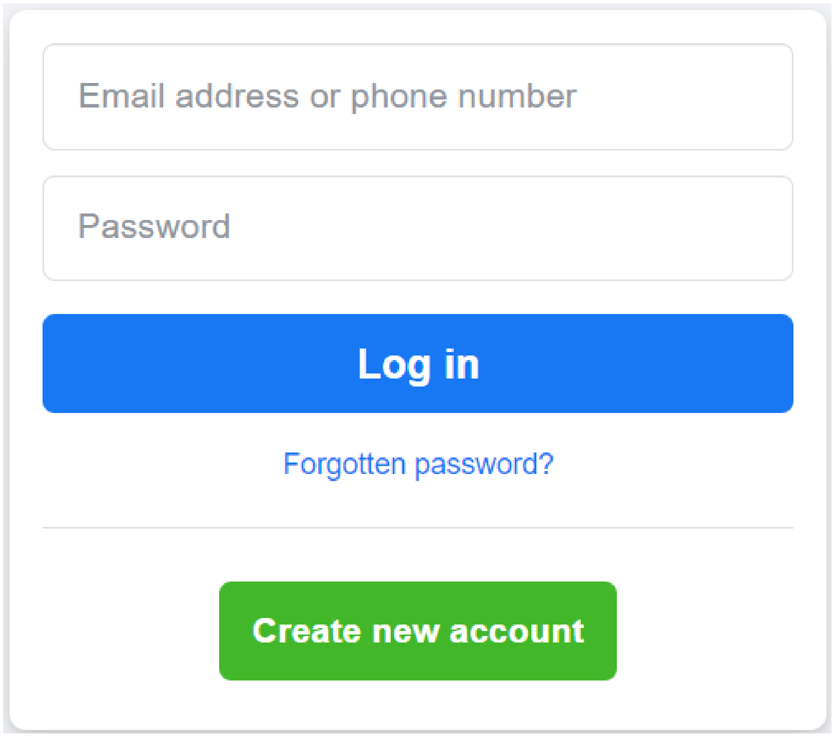

Email
Full Name
Username
Password
People who use our service may have uploaded your contact information t   ' earn More
By signing up, you agree to our Terms , Privacy Policy and Cookies Policy .
Sign up

3. Login: The existing customer can log in using the registered email id and password.

1. Ids:
    - Email
    - Password
    - LoginButton
    - Forgot password link
    - Create new account button

2. API endpoint Url: http://localhost:8000/login

3. Output screenshot:



4. Dashboard: A homepage which contains paths to

1. Ids;
   - Chats
   - Calls
   - Settings
   - Archive
   - Media
   - Community

2. API endpoint Url: http://localhost:8000/home

__BACKEND:__

__1.Model Layer:__

1. User Model: the user type (admin or customer) and all user information are stored

Attributes:

- email: String
- password: String
- username: String
- mobileNumber: String
- role: String

2. Login Model: This class contains the email and password of the user.

Attributes:

- email: String
- password: String

<u>2.Controller Layer:</u>

1. Signup Controller: This class control the user signup

Methods:

- saveUser(UserModel user): This method helps to store users in the database and return true or false based on the database transaction

2. Login Controller: This class controls the user login.

Methods:

- checkUser(LoginModel data): This method helps the user to sign up for the application and must return true or false.