Group 14 - ELEC_5619 Object Oriented Application Frameworks

Flicker - a translation messaging webapp.

Table of content

Libraries and version

Frontend

Library	Version
@reduxjs/toolkit	^1.9.5
@stomp/stompjs	^7.0.0
@types/node	20.5.9
@types/react	18.2.21
@types/react-dom	18.2.7
@types/react-infinite-scroll-component	^5.0.0
antd	^5.9.4
autoprefixer	10.4.15
axios	^1.5.0
emoji-picker-react	^4.5.2
eslint	8.48.0

Library	Version
eslint-config-next	13.4.19
next	13.4.19
postcss	8.4.29
react	18.2.0
react-dom	18.2.0
react-infinite-scroll-component	^6.1.0
react-redux	^8.1.2
tailwindcss	3.3.3
typescript	5.2.2

Backend

Library	Version
spring boot	2.7.7
spring boot starter	2.7.8
spring-boot-starter-actuator	
spring-boot-starter-validation	
spring-boot-starter-data-jpa	
spring-boot-starter-security	
spring-boot-devtools	
spring-boot-starter-web	
spring-boot-starter-websocket	
spring-boot-starter-mail	2.7.7
spring-boot-starter-test	
spring-security-test	
lombok	
jjwt	0.9.1
log4j	1.2.17
freemarker	2.3.28
swagger	3.0.0
expiring.map	0.5.10

Library	Version
junit5	5.9.0
jacoco	0.8.2
surefire	2.19.1
google-cloud-translate	2.20.0
google-cloud-speech	4.18.0
firebase-admin	7.0.0
google-cloud-storage	1.96.0
google-api-client	1.31.1
modelmapper	2.4.4
springdoc-openapi-ui	1.6.14
commons-collections4	4.4
guava	23.0
postgresql	

Working functionalities of the project

User Registration and Authentication:

- Users can create accounts using their email.
- Users are authenticated securely before accessing the app's features.

Contact Management:

- Users are able to add, remove, and organize contacts.

Messaging:

- Users are able to send text messages to individual contacts or groups.
- Users are able to send multimedia messages such as images, videos, and audio recordings.
- Conversations are organized and user-friendly, with timestamps and read receipts.
- Users are able to use emojis to show their emotional message.

Language Support:

- An integrated language translation feature that translates messages in real-time, allowing users who speak different languages to communicate seamlessly.
- A feature that transcribes voice messages into text in multiple languages, promoting seamless communication across language barriers.
- The transcribed text is displayed alongside audio messages.
- The application automatically detects the source language of each message.

Notifications:

- Users receive real-time notifications for incoming messages and other relevant events.
- Notifications are customizable, allowing users to choose their preferred notification settings.

Search and Filters:

- Users are able to search for specific messages, conversations, contacts, or media within the app.
- The app offers filtering options to sort conversations, messages, and media.

Sync Across Devices:

- User data, including messages, contacts, and conversation history, are synchronized and stored on cloud service.

How to run

Database

1. Download PostgreSQL following https://www.postgresql.org/download/ (or on Mac, you can use Homebrew to install postgresql: `brew install postgresql`).
2. Download pgAdmin from https://www.pgadmin.org/download/.
3. Start the PostgreSQL services (downloaded from step 1, or on Mac you can use `Homebrew` to start: `brew services start postgresql`).
4. Open pgAdmin app just downloaded.
5. Configure the server connection (hostname: localhost, port: 5432, username: postgres, password: admin), or you can use your own configuration, but it must be on localhost port 5432.
6. Create database with name `flicker_chatapp`.

Backend

1. Ensure you have Java JDK 8 or higher installed.
2. Install Maven following https://maven.apache.org/install.html.
3. Update the application.properties file with your database credentials and other environment-specific settings.
4. From the root folder, go to `back-end/Flicker`.
5. Then, install all the dependencies, skipping all the tests, using the command: `mvn clean install -DskipTests`.
6. After that, to run the application, just locate the file: FlickerApplication.java (inside the directory src/main/java/com/elec5619/group14/flicker) and run the main method; or you can run the application via command line: `mvn spring-boot:run`.
7. Execute the following SQL commands in pgAdmin to initiate the roles:

```
INSERT INTO role (ROLE_NAME) VALUES ('ROLE_USER') ON CONFLICT
(ROLE_NAME) DO NOTHING;
INSERT INTO role (ROLE_NAME) VALUES ('ROLE_ADMIN') ON CONFLICT
(ROLE_NAME) DO NOTHING;
```

8. Finally, access the application backend at: http://localhost:9014.

Frontend

1. Install `nodejs` version `20.6.1` following https://nodejs.org/en/download.
2. From the root folder, go to `front-end`.
3. In command line, run `npm install`.
4. Then, `npm run dev`.
5. Start back-end server.
6. In your browser, open `http://localhost:3000` and enjoy the app.