CEBU INSTITUTE OF TECHNOLOGY

UNIVERSITY

COLLEGE OF COMPUTER STUDIES

Developer Setup Guide

for

CollaborAid

Developers

Jesson Chyd Cultura

Jhudiel Adrian Artezuela

Harold Destura

May 7, 2025

1 Overview

Welcome to the CollaborAid developer setup guide. This document provides step-by-step instructions for setting up the full-stack development environment, including tools, dependencies, database configuration, GitHub access, and how to run the backend, web frontend, and mobile frontend. Whether you're contributing to the backend (Spring Boot), web (React + TypeScript), or mobile (Kotlin + Jetpack Compose), this guide will help you get up and running efficiently.

2 Tools & Technologies

Category	Tool/Technology
Backend	Java 17, Spring Boot. *Deployed on Render*
Frontend (Web)	React + TypeScript, Tailwind CSS, ShadCN UI. (*Deployed on Vercel)*
Frontend(Mobile)	Kotlin, Jetpack Compose (Android Studio)
Database	MySQL (*Hosted on Azure)*
Websocket	Stomp over SockJS
Build Tool	Maven
Version Control	Github
Containerization Tool	Docker
AI Integration	Open AI API
Rest Testing	Postman
IDE(Backend)	IntelliJ IDEA
IDE(Frontend)	Visual Studio Code
IDE(Mobile)	Android Studio

3 Prerequisites

Ensure the following tools are installed:

- **Java 17+ (JDK)** *For running the Spring Boot backend.*
- **Node.js v18+** *Required for building and running the frontend (Vite, React, TypeScript, Tailwind CSS, ShadCN UI).*
- **npm (comes with Node.js)** *Used for installing frontend dependencies.*
- **Maven** *For building and running the backend (Spring Boot).*
- **MySQL Server 8+** *Required for local database setup (used by the backend).*
- **Git** *For cloning the project and version control.*
- **Postman** *Helpful for testing REST API endpoints.*
- **OpenAI account with API Key** *Required for integrating AI support features.*
- **Android Studio** *Recommended IDE for developing and running the Kotlin-based mobile app using Jetpack Compose.*
- **Visual Studio Code** *Recommended IDE for working with the frontend (React + TypeScript).*

4 What to Do: Step-by-Step Setup Guide

🟣 *GitHub Repository*

Github Repository: github.com/Cultura15/IT342-G5-CollaborAid

The project follows a monorepo structure, containing both frontend and backend codebases.

Open your command and follow these instructions:

```
git clone https://github.com/Cultura15/IT342-G5-CollaborAid.git
```

```
cd IT342-G5-CollaborAid
```

🔵 *Database Configuration (MySQL)*

First, log in to your MySQL instance and create a database:

```
CREATE DATABASE collaborbase;
```

Update the application.properties file in the backend (src/main/resources) with your local MySQL credentials:

```
spring.datasource.url=jdbc:mysql://localhost:3306/collaborbase
spring.datasource.username=root
spring.datasource.password=your_password
```

Lastly, ensure the following JPA property is added to auto-manage your schema updates:

```
spring.jpa.hibernate.ddl-auto=update
```

🟡 *OpenAI API Integration*

To enable the AI assistant feature in Collaboraid, you'll need access to the OpenAI API.

1. Visit the official OpenAI platform: https://platform.openai.com
2. Sign up or log in to your OpenAI account.
3. Go to your API Keys section and generate a new secret key.

In your backend project's application.properties, add the following line:

```
openai.api.key=sk-xxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

Note

The GPT-based AI features in this project require a valid OpenAI token. For development and deployment, you must use your own OpenAI API key.

The original developer (project owner) has purchased and used their personal GPT token for production usage, which is not included in this repository for security and privacy reasons.

If you're only testing locally and want to explore the AI functionalities, you'll need to obtain your own token and be aware that usage may incur billing based on OpenAI's pricing model: https://openai.com/pricing

Be sure to keep your API key private and never commit it to version control.

▶️ *Running the Backend (Spring Boot)*

To start the backend service, first navigate to the backend directory: `cd backend`

Open the project using IntelliJ IDEA, locate the *GoogleContactsCulturaApplication.java* file, and click the Run button to launch the application.

Alternatively, you can start the backend directly from the terminal using: `mvn spring-boot:run`

Once running, the backend will be accessible at http://localhost:8080

▶️ *Running the Web Frontend*

Navigate into the web frontend directory: `cd frontend-web`

Install all required project dependencies using: `npm install`

Then, launch the development server with: `npm run dev`

The frontend will now be available at: http://localhost:5173

▶️*Running the Frontend (Mobile)*

To run the mobile application, follow these steps:

1. Open Android Studio, then select "Open an existing project" and navigate to the frontend-mobile/ directory.

2. Allow the IDE to sync the project with Gradle automatically. If not, click "Sync Now" when prompted.

3. Connect a physical Android device or launch an emulator from the Device Manager.

4. Once Gradle sync is complete and all dependencies are resolved, click the Run button or press Shift + F10 to build and launch the app.

Your mobile application will now be running on the selected device or emulator.

API Testing with Postman

Base URL: http://localhost:8080/api/

Authorization:

- **Type:** Bearer Token
- **Header:** Authorization: Bearer <your_jwt_token>

Useful Endpoints:

POST /auth/register

POST /auth/login

POST /app/sendMessage

GET /task/open - Fetch open tasks

GET /task/in-progress - Fetch tasks in progress

GET /task/done - Fetch completed tasks

GET /auth/active-users - Fetch active users

GET /auth/inactive-users - Fetch inactive users

GET /auth/all - Fetch all users

GET /messages/conversation/{adminId}/{userId} - Fetch conversation between admin and user