BENTALEB Farès, GRANDJEAN Hélène, CLOAREC Margaux

Technical Report: Collaborative Task Management Application

1. Project Environment Configuration

This project was developed to meet as much as possible the requirements for the “Project Algorithmics and Java Programming" module.

The project is basically separated in 2 different java projects : the backend API (for the requests and the database in PostgreSQL and using docker, + spring boot in java to configure the API) and the second part with the JavaFX application with the API integrations inside it (using localhost API calls and making everything work, and designing the pages... etc)

- **IDE and Tools:** Visual Studio Code and IntelliJ IDEA were used for the development of this project, because of their excellent support for JavaFX and Spring Boot.
- **Java Version:** The Java Development Kit (JDK) version 17 was used to compile the application.
- **Build Tool:** The JavaFX application was build using the JavaFX standard library and no build tool was used. The backend Spring Boot application uses Maven.

2. Libraries and Dependencies

The 2 java projects use the following libraries:

- **JavaFX**: JavaFX version 23.0.1 was used to create the application user interfaces, providing a complete toolkit to build the UI.
- **Jackson:**
    - com.fasterxml.jackson.core:jackson-databind is used to serialize Java objects to JSON strings and deserialize JSON strings to Java objects. This library enables seamless communication between the backend API and the JavaFX application.
    - com.fasterxml.jackson.datatype:jackson-datatype-jsr310 is used to enable Jackson to correctly parse Java 8 Date and Time API objects.
- **Apache HTTP Client 5**:
    - org.apache.httpcomponents.client5:httpclient5 is the primary library used for sending and receiving HTTP requests from the Spring Boot project.
    - org.apache.httpcomponents.core5:httpcore5 provides support for the core components of the HTTP framework.
- **SQL Driver**:
    - com.microsoft.sqlserver:mssql-jdbc: was used for interacting with the Microsoft SQL Server database.
- **Spring Boot Starter Web:** This is a starter library that enables building RESTful APIs with Spring Boot, and also includes Tomcat.

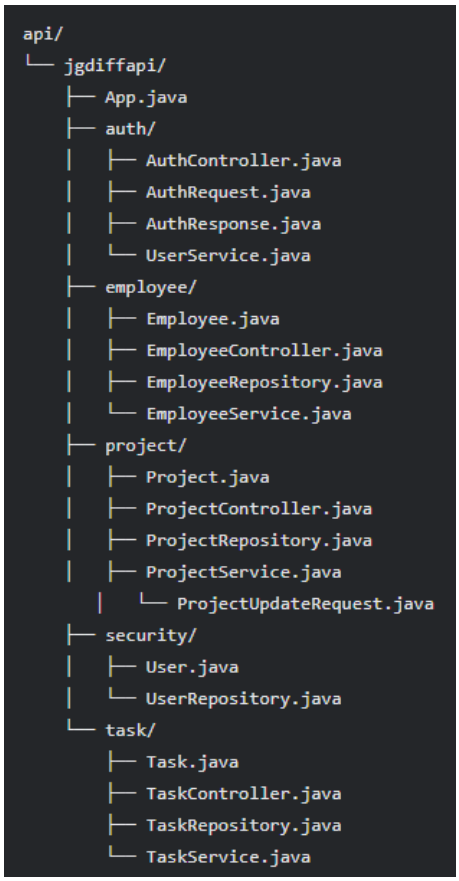
- org.springframework.boot:spring-boot-starter-web
- **Spring Boot Data JPA** : This starter libary enables database interactions.
    - org.springframework.boot:spring-boot-starter-data-jpa
- **Lombok** : A library that reduces boilerplate code like getters, setters and constructors (making things take less time to code basically).
    - org.projectlombok:lombok.
- **Jakarta Persistence** : This library provides a set of annotations that allows you to map Java objects to database.
    - jakarta.persistence:jakarta.persistence-api
- **Java SQL API**
    - java.sql : This library provides a set of interfaces for interacting with SQL databases.

3. Class Diagram

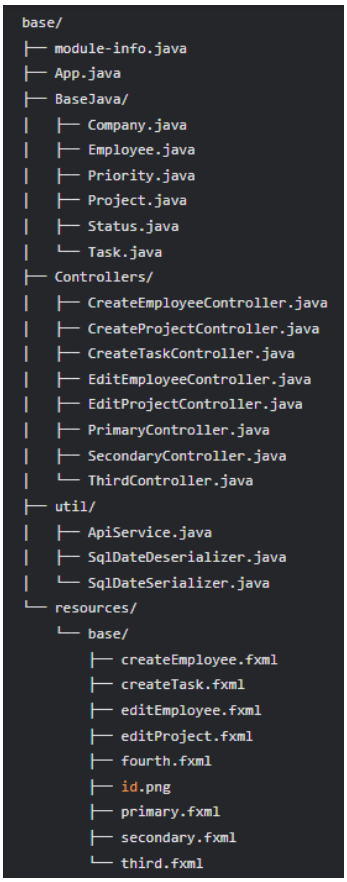
Since making a diagram would be too long and too big (because we have too many classes, constructors, functions... etc), here is instead the Project Structure (it still shows how the project works basically, how everything is working together but without detailing the inside of the java classes)

Please go to the next page to see the rest (sorry if the document is 5 pages, it's to include the Project Structure correctly and images)

For the Spring Boot Project :



For the JavaFX Project :



4. Package Architecture

The project uses the following main packages:

Spring Boot API Packages:

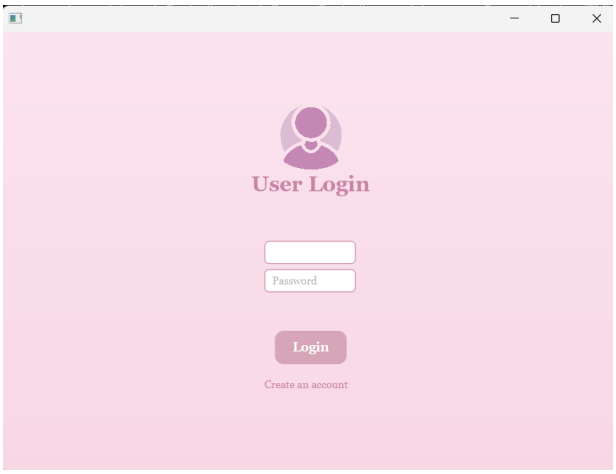
- **api.jgdiffapi**: The root package that contains the App.java for the Spring Boot application.
- **api.jgdiffapi.auth**: This package contains classes that handle authentication related functionality, which contains, user creation, login.
- **api.jgdiffapi.employee**: This package contains classes related to employee management such as Employee, EmployeeController, EmployeeRepository, and EmployeeService.
- **api.jgdiffapi.project**: This package includes the classes to handle project related functionality, such as Project, ProjectController, ProjectRepository, and ProjectService. It also includes the ProjectUpdateRequest DTO.
- **api.jgdiffapi.security**: Contains the classes that handles user security, and which contains the User and UserRepository classes.
- **api.jgdiffapi.task**: This package is responsible for task management and includes Task, TaskController, TaskRepository, and TaskService classes.

JavaFX Project Packages:

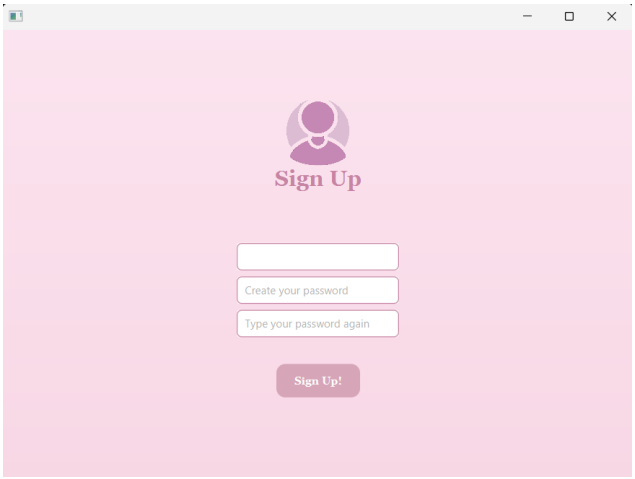
- **base**: This is the root package that contains the main application class (App.java) and the module-info.java file, acting like the main package of your java project.
- **base.BaseJava**: This package contains all the data model classes that you are using, such as Employee, Project, Task, Priority, Status, and Company.
- **base.Controllers**: This package includes all JavaFX controller classes, such as PrimaryController, CreateTaskController, EditProjectController, CreateEmployeeController, EditEmployeeController, and all other FXML controllers.
- **base.util**: This package includes utility classes, for example, the ApiService for making API calls, and SqlDateSerializer and SqlDateDeserializer for converting java SQL date objects.
- **base.resources**: This package includes the files related to the resources, like images and the FXML files for the JavaFX application.

5. Application Interfaces

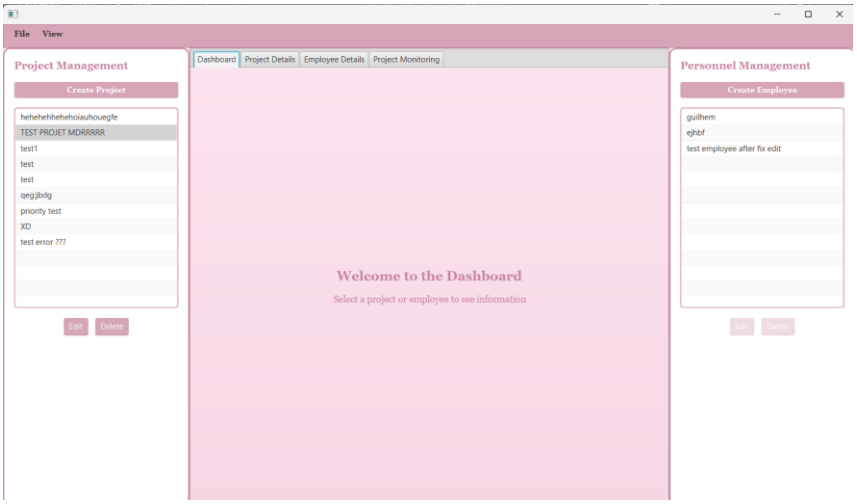
- **Login Page:** This interface allows users to authenticate into the application. The authentication process is implemented using an API that logs in the users. It contains a username text field, a password text field, a create an account hyperlink that opens the sign-up page, and to finish we have a button to login.



- **Signup Page:** This interface allows users to create an account to be able to login into the application later. It also uses the API to create the users. It contains a username text field, a password text field and a repeat password text field, and to finish the sign-up button.



- **Main Application View:** This is the main interface of the application; it displays two columns. The left shows the project management interface, which is used to manage projects. The right shows the personnel management interface, which is used to manage employees. This screen also shows different tabs, where you can view the different functionalities.



- **Create Project Dialog:** This interface allows you to create new projects. You can enter the project name, description, deadline, and priority.
- **Create Employee Dialog:** This interface allows you to create new employees by entering their names.
- **Create Task Dialog:** This interface allows you to create new tasks by entering task details like name, description, priority, due date, status and the assignee. This page is accessible from the Project Details tab.
- **Add Member Dialog:** This interface allows you to add a member to a selected project by selecting between all the employees.