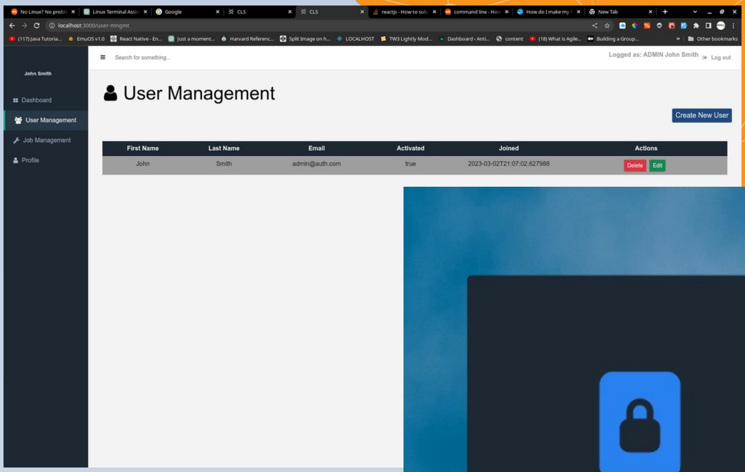
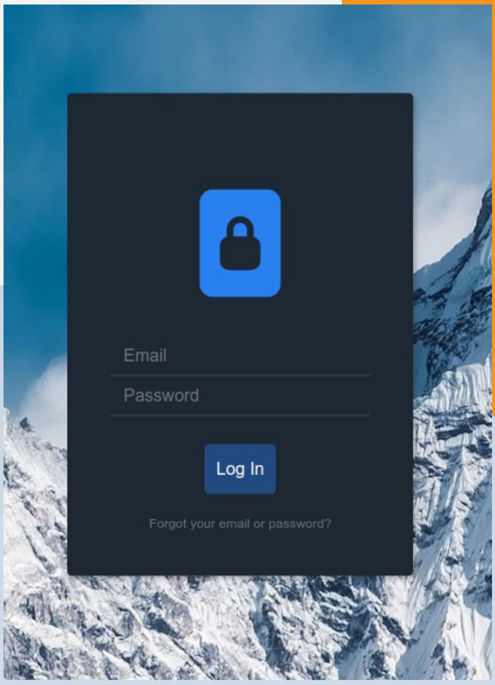
Crew Logistic System
Project Artefact Description



User Management

First Name	Last Name	Email	Activated	Joined	Actions
John	Smith	admin@auth.com	true	2023-03-02T21:07:02.627988	Delete Edit



CLS

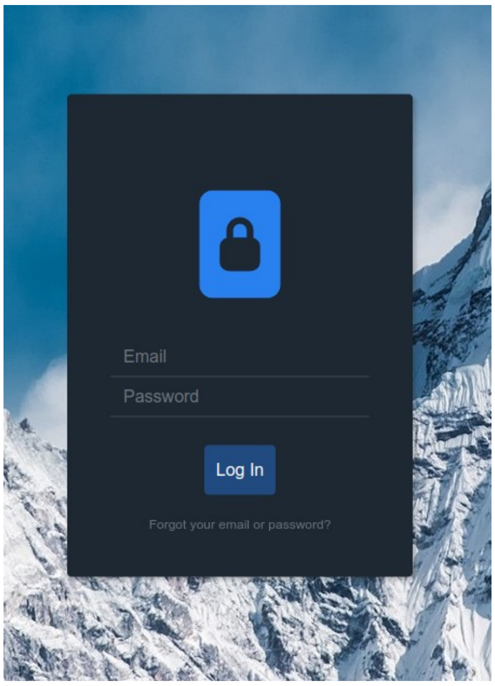
Name:	**Arkadiusz Grudzien**
ID Number :	**20001306**
Date:	**12/03/2023**
Module Leader:	**Dr. Ela Homayounvala**
First Supervisor:	**Victor Sowinski-Mydlarz**
Second Supervisor :	**Sandra Fernando**

Table of Contents

Features

Features which has been developed at this stage:

- Login: Users can log in to the system by providing their email and password in the login fields provided. The system verifies the user's credentials and then grants access to the appropriate dashboard.



- User Management: The admin user can add new users to the system by filling out a specific form. The form collects basic user information such as name, email, etc. and allows the admin to specify the user's role (either as an admin or a crew member). The admin can also edit or delete existing user accounts.
- Admin Dashboard: The admin user has access to a dashboard that displays information about the users in the system. The dashboard shows how many users have been created in total, as well as how many of those users are admin or crew members. The admin can also view a list of all the users in the system and edit or delete their accounts from the dashboard.
- Profile: The user can log in to the system and view their own dashboard. The profile section information such as the user's name, email, phone number, and address. The user can update their own profile information from the profile section.

- Job Management: The admin user can create new job assignments for the crew members by filling out a specific form. The form collects details such as the job number, date, time, location, and the drivers and crew chief assigned to the job. The admin can also edit or delete existing jobs.

Remaining Features

- Sending jobs to specific crew members allows for better task assignment and streamlined workflow.
- Being able to see (as a crew member) accepted jobs helps to track progress and ensure timely completion of tasks.
- A calendar with upcoming jobs provides better planning and coordination of team workflow, and prevents scheduling conflicts.
- Account recovery: If a user forgets their login credentials, the login feature may also include functionality to recover their account, such as password reset or account recovery options.

Functionalities

Login

Login has following functionalities:

User identification: The login feature first identifies the user by prompting them to enter their login credentials, such as a username and password. This information is then validated against the user's account information in the system's database.

Credential verification: The system verifies the user's credentials by checking if they match the records in the database. If the credentials are valid, the user is granted access to their account.

The authentication which is implemented in this project is JWT.

JWT stands for JSON Web Token, which is a standard for securely transmitting information between parties in the form of a JSON object.

In this system, when a user logs in, the server generates a JWT that contains the user's identity and any relevant metadata. This JWT is then signed with a secret key known only to the server. The server then sends the JWT back to the client, which stores it in local storage.

For every subsequent request that the client makes to the server, it includes the JWT in the request header.(For development puprose this is swiched off) The server then verifies the JWT's signature and decodes the contents to identify the user and authorize the request.

Using JWTs has several advantages over other authentication methods, including:

- Stateless: The server does not need to store any session data, making it more scalable.
- Secure: JWTs are signed and can be encrypted, making them tamper-proof and ensuring that only authorized parties can decode it.

The remaining functionality for this feature is:

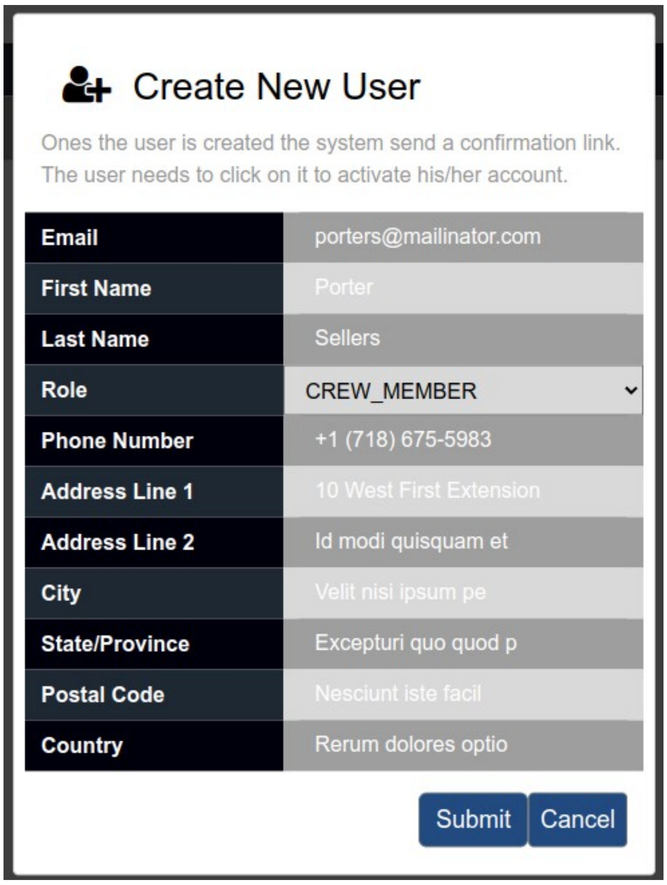
Account recovery: If a user forgets their login credentials, the login feature may also include functionality to recover their account, such as password reset or account recovery options.

The website features a login system that allows users to access different levels of functionality based on their assigned role. Upon login, user credentials are authenticated and their role is determined. The website then redirects them to the appropriate dashboard with corresponding access to features and information.

Currently, the system is designed such that admins have access to more functionalities than crew members. This ensures that admins have the necessary tools and permissions to effectively manage the system, while also maintaining the security and privacy of stored data and information.

Register New User

The clients requests was that the user can only be created via admin account, not as usualy is that the registration is available to everyone. This functionality require admin to log in to the system and go to section User Management and click on the button called “Create New User”. The admin have to fill the form which have been appeared and choose what type of user needs to be created: admin or crew member. Ones the user is created, the user receives an email to verify his email address. One this is done the user can normally log in.


Create New User

Ones the user is created the system send a confirmation link. The user needs to click on it to activate his/her account.

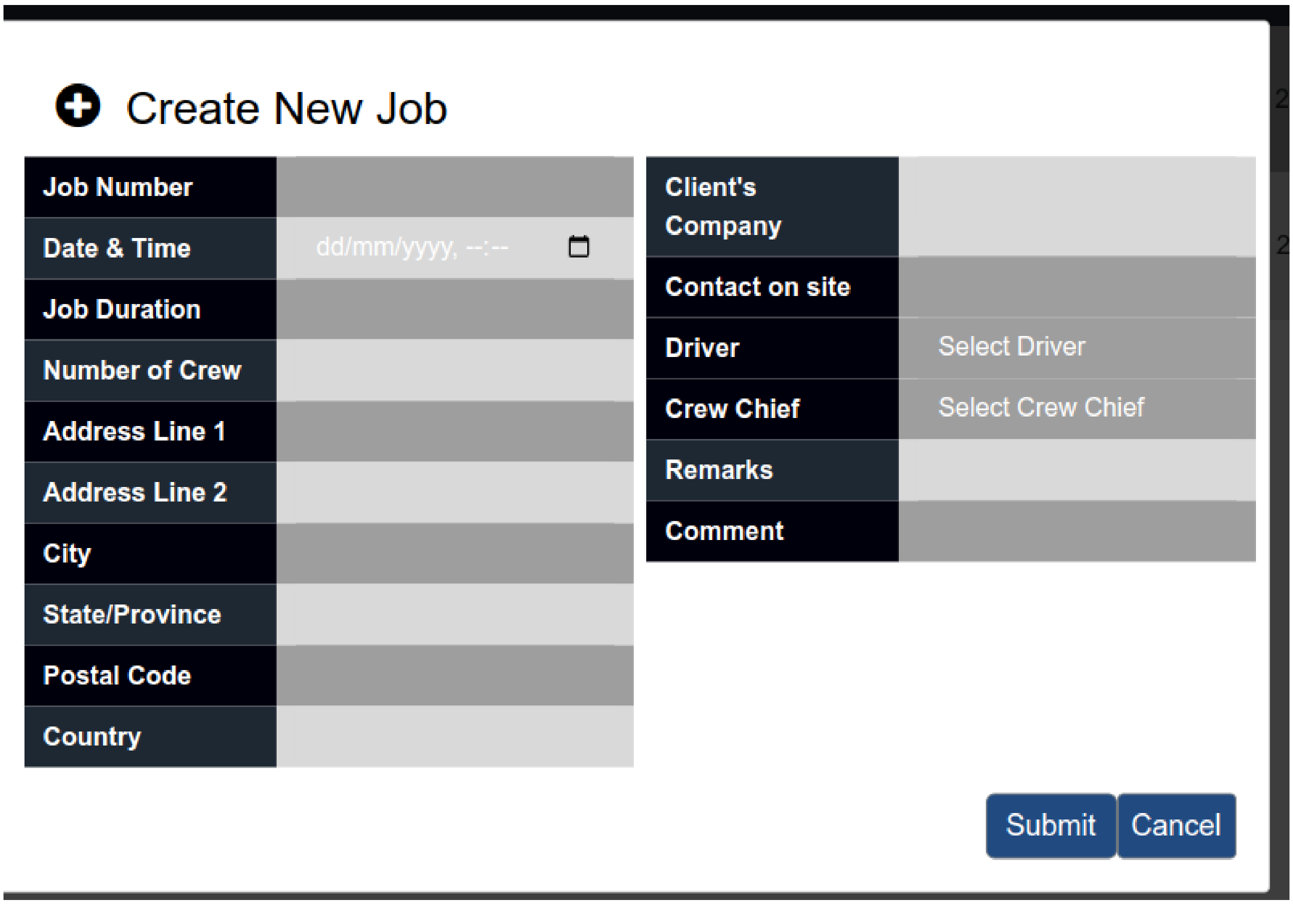
Email	porters@mailinator.com
First Name	Porter
Last Name	Sellers
Role	CREW_MEMBER
Phone Number	+1 (718) 675-5983
Address Line 1	10 West First Extension
Address Line 2	Id modi quisquam et
City	Velit nisi ipsum pe
State/Province	Excepturi quo quod p
Postal Code	Nesciunt iste facil
Country	Rerum dolores optio

Figure 1: Creating new user

For development purpose there has been setup email server called maildev where all emails are sent.

Create a Job

The app enables an admin account to effortlessly add job postings to the database. By logging in as an admin, the user gains access to the "Add Job" page where they can enter details such as the job number, address etc. Once the information is submitted, the job posting is saved to the database, ready for candidates to view and apply. This process is designed to simplify the job listing management.



Create New Job

Job Number		Client's Company	
Date & Time	dd/mm/yyyy, --:--	Contact on site	
Job Duration		Driver	Select Driver
Number of Crew		Crew Chief	Select Crew Chief
Address Line 1		Remarks	
Address Line 2		Comment	
City			
State/Province			
Postal Code			
Country			

Figure 2: Create new job form view

Technical details

Spring Boot Profiles provide a way to configure an application for different environments such as development, testing, staging, or production. Profiles allows to define different configurations for each environment and to activate them at runtime. In Spring Boot, a profile is a set of properties and configurations that define the behaviour of the application for a specific environment.

There are two profiles that have been implemented for the software, namely the deploy and dev profiles. The dev profile, which stands for development, has been designed to facilitate making changes on the fly. To achieve this state, certain dependencies need to be installed, including:

OpenJdk version 17.0.6 (Java),
NodeJS version 16.19.1,
NPM 8.19 (which comes with Node),

MySQL 8.0.32, and Apache Maven 3.6.3.

Profile “dev”

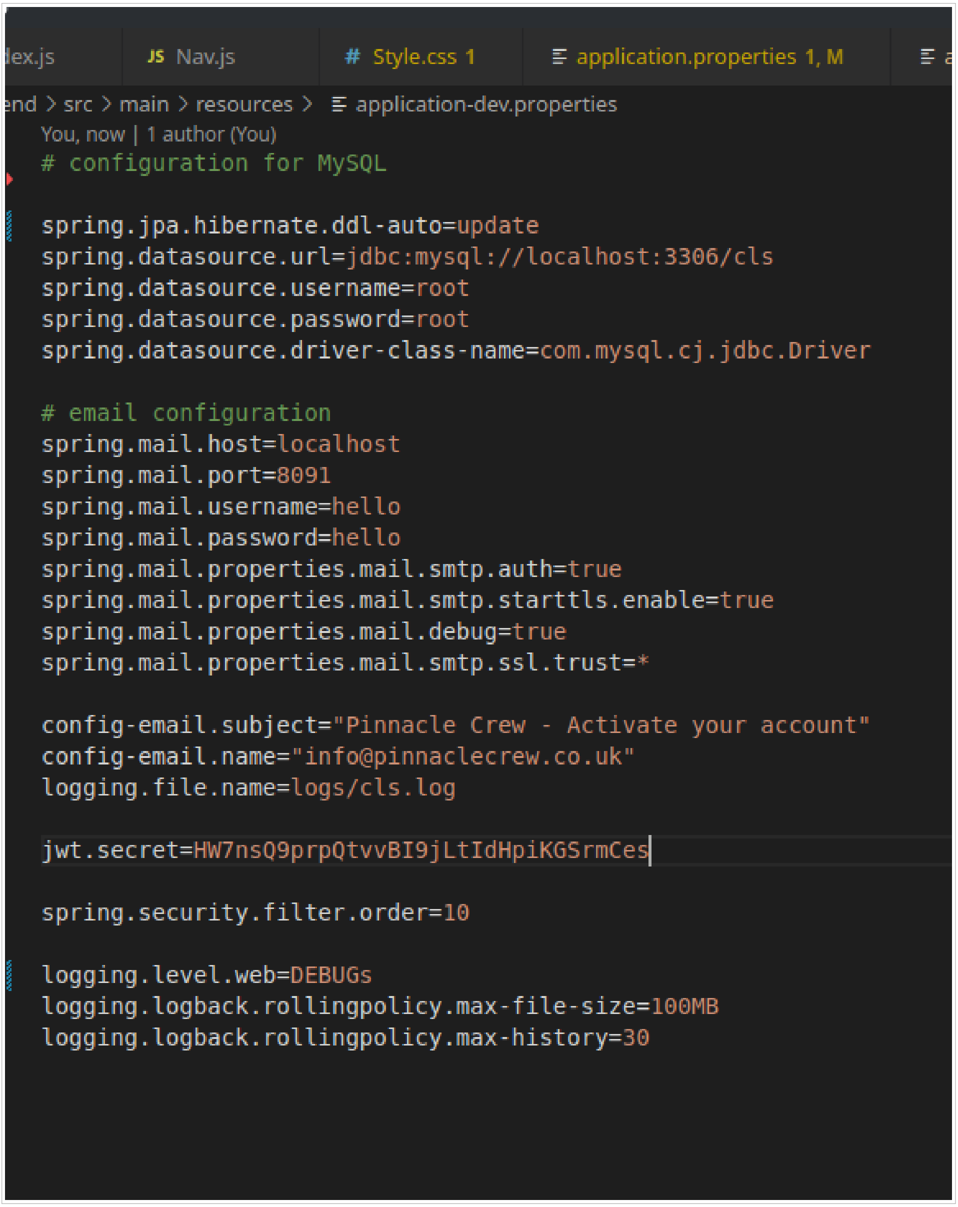
To run the software in this profile (dev), several steps need to be followed. Needs to be sure that the profile is switched on in aplication.properties.

spring.profiles.active=dev

application.properties is a configuration to configure various properties of the application. It is a plain text file that is located in the src/main/resources directory of the application.

The file application.properties contains key-value pairs that configure various aspects of the Spring Boot application, such as the server port, database connection settings, logging configuration, security settings, and more. It allows to customize and configure the application without modifying the source code. In this case application.properties include common configuration for both profiiles. The divided configurations are included in aplication-dev.properties and aplication-deploy.properties repsectivly. To swich between the profiles easly to change one value as mentioned before.

The profile “dev” has following configuration:

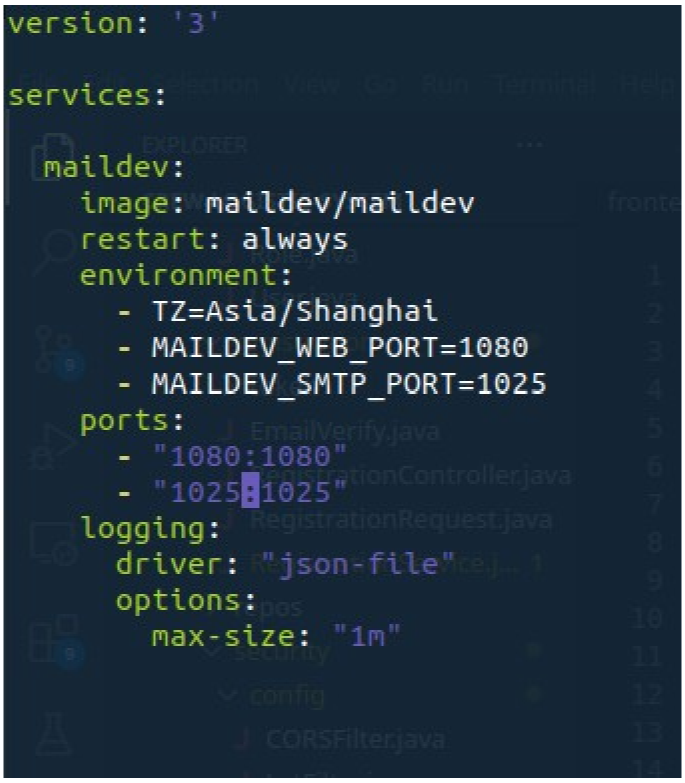

lex.js
JS Nav.js
# Style.css 1
application.properties 1, M
end > src > main > resources > application-dev.properties
You, now | 1 author (You)

```
# configuration for MySQL

spring.jpa.hibernate.ddl-auto=update
spring.datasource.url=jdbc:mysql://localhost:3306/cls
spring.datasource.username=root
spring.datasource.password=root
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# email configuration
spring.mail.host=localhost
spring.mail.port=8091
spring.mail.username=hello
spring.mail.password=hello
spring.mail.properties.mail.smtp.auth=true
spring.mail.properties.mail.smtp.starttls.enable=true
spring.mail.properties.mail.debug=true
spring.mail.properties.mail.smtp.ssl.trust=*

config-email.subject="Pinnacle Crew - Activate your account"
config-email.name="info@pinnaclecrew.co.uk"
logging.file.name=logs/cls.log

jwt.secret=HW7nsQ9prpQtvvBI9jLtIdHpiKGSrmCes

spring.security.filter.order=10

logging.level.web=DEBUGs
logging.logback.rollingpolicy.max-file-size=100MB
logging.logback.rollingpolicy.max-history=30
```

The first lines are configuring the database providing the way how the database need to behave, update that the data will be added there is also variable called create-drop. It does when spring boot reboots the data is lost usually used for testing when are created new entities.
The other line configuring the data source credentials to log in to the database.

Email configuration is for the mail server which has been implemented for developing purpose. Ones the user is created the email server receives the mail for confirm the email address of the user who is being registered.

The mailadev (https://github.com/maildev/maildev.git) can be run via docker with folowing configuration :



```
version: '3'

services:

  maildev:
    image: maildev/maildev
    restart: always
    environment:
      - TZ=Asia/Shanghai
      - MAILDEV_WEB_PORT=1080
      - MAILDEV_SMTP_PORT=1025
    ports:
      - "1080:1080"
      - "1025:1025"
    logging:
      driver: "json-file"
      options:
        max-size: "1m"
```

command: docker-compose up

To be bale to run the program in mentioned profile the next step is to go to /backend folder and run a following command in terminal:

mvn spring-boot:run

The above command runs the Spring Boot application if all the configurations has been set up correctly following the guide above.

The next step is to run the frontend. To be able to do that neeed to "be" in the frontebd folder and run command :

npm install && npm start

This command is responsible at first to download all dependences for the react framework and the it runs the web application at http://localhost:3000. Its important to have setup correct value in package.json:

"proxy": "http://localhost:8080",

The Spring Boot system creates a default user. The credentials are

Email: admin@auth.com
Password: 1234

Profile “deploy” (docker)

This profile has been made for viewing the program without configuring the machine where will be run, however docker needs to be installed *Docker Documentation. (2023)*.

Ones the docker is installed simply following command needs too be run:

```
docker-compose up
```

but needs to be sure that the profile is switched on in aplication.properties:

```
spring.profiles.active=deploy
```

and in package.json for

```
"proxy": "http://10.5.0.5:8080",
```

The delivered files should already have switched on this profile.

The proxy value is the IP and port where the Spring Boot application runs. The fron-tend “knows” where to send all the requests.

Source Control

In this project, Git has been utilized as the source control system. Git is a popular ver-sion control tool that enables keeping track of changes to the source code over time. With Git, developers can work on different features and changes simultaneously and merge them seamlessly, reducing the likelihood of conflicts and errors. Furthermore, Git provides a comprehensive history of changes made to the codebase, enabling de-velopers to easily revert to earlier versions if needed.

Project’s structure.

Spring Boot Backed - UML

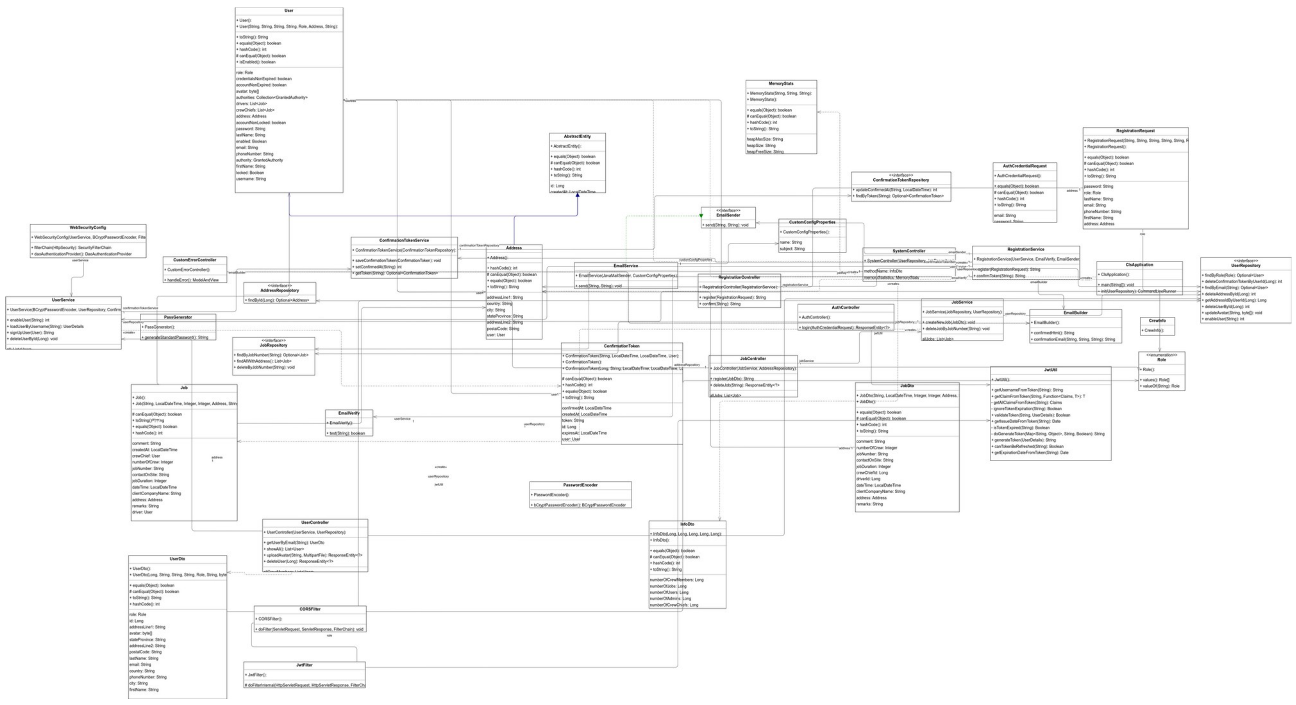


Figure 3: This show the structure of the web application - UML

Packages - backend

The table below describes all packages in backend included in this projects.

Package	Description
controller	The Controller package in Spring Boot handles HTTP requests, with annotated classes and methods mapping to endpoints. It communicates with other components in the application, such as Services and Repositories, to process requests and generate responses, promoting consistency and simplifying implementation.
dto	The DTO (Data Transfer Object) package in Spring Boot transfers data between layers using classes with private fields and getter/setter methods annotated with @Data. This promotes loose coupling, reduces data

	transfer, and enhances maintainability.
email	This package generates and sends emails to users upon registration, converting plain text to HTML format.
model	The Model package in the project contains several classes, including an Abstract Entity class that defines common entity fields, an Address class representing user addresses, a Crew Info class with information about a crew, a Job class representing job details, a Role class defining user roles, and a User class containing user data. These classes define the structure and behavior of the application's data and are used to communicate with other components in the application, such as Services and Repositories. By organizing data into these classes, the Model package promotes consistency and maintainability throughout the project.
registration	The registration package is responsible for managing user registration functionality. It contains classes that handle user input validation, user creation, and email confirmation.
repos	The repos package in the project is responsible for managing the persistence of data to and from the application's database. It contains interfaces or classes that define the methods for interacting with the database.
security	The security package is responsible for managing the authentication and authorization of users accessing the application. It contains classes that handle user authentication, authorization.
service	This package implementing the business logic of the application. It contains classes that handle data processing and manipulation, as well as interactions between different components of the application.
util	This package provides additional classes which are helping certain things during development such as random password generator.

Frontend

React Components and packages:

Header: The header component appears at the top of each page and contains elements such as the logo, navigation links, and search bar. It provides a consistent look and feel across the application and helps users quickly find what they're looking for. The logic for it is not yet implemented, anyway its additional feature for future development.

Nav: The navigation component is responsible for displaying links to different pages within the application. It incudes the sections of the app.

The Pages package is responsible for displaying the main content of your application. Each page is composed of multiple smaller components, such as text, images, forms, and other interactive elements. The Pages package is further divided into three subpackages based on user roles:

- Admin Pages: This package contains content that is only available to the admin user. It include pages for managing users, jobs.
- Crew Pages: This package contains content that is only available to the crew members of the application. It include pages for managing jobs, schedules, inventory, and other features that are specific to the crew members.
- Common Pages: This package contains content that is available to all users of the application.

The Util package is a collection of utility functions that are used throughout the application. It includes functions that perform common tasks such as data validation, formatting, request to server etc.

The frontend includes also the the requests in certain points of the application. It could not be developed in other ways.

Rest of the files

As previously mentioned, this project includes files for Docker configuration. The project has been divided into four containers: Frontend, Backend, Database, and Mail Server. The Dockerfile for the Backend container is responsible for downloading the necessary files to run the program.

The main Docker configuration file is located in the root folder and is called docker-compose.yml. This file is responsible for configuring all the containers and ensuring that they can communicate with each other. Additionally, there is a configuration for the network that enables communication between the four sub-containers mentioned earlier.

In the project, there is a script named "run" which can be used to run all the programs in development mode. This script is very useful as it saves the trouble of manually executing each command one by one.

Using the "run" script, can easily start the development server for the frontend, backend, and other necessary programs with a single command. This streamlines the development process and saves valuable time that can be spent on other tasks.

Note

There were certain aspects of the project that could not be explained in as much detail as I would have liked, but that I have tried my best to provide a comprehensive overview within the given word limit.

References

Docker Documentation. (2023). Install on Windows. [online] Available at: https://docs.docker.com/desktop/install/windows-install/.

Docker (2020). *Install Docker Engine on Ubuntu*. [online] Docker Documentation. Available at: https://docs.docker.com/engine/install/ubuntu/.