Even Semester (2023)



**BINUS UNIVERSITY**

**BINUS INTERNATIONAL**

**Assignment Cover Letter**

**(Individual Work)**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | |  | |  | |  |
| **Student Information**: **Surname** | | | | | **Given Names**    **Ravel** | | **Student ID Number**  **2301890320** | |  |
| 1. | | **Tanjaya** |  | |  |
|  |  |  |
| **Course Code** | **: COMP6510** |  |  | | **Course Name** | | **: Programming Languages** | |  |
| **Class** | **: L2AC** |  |  | | **Name of Lecturer(s)** | | **: Jude Joseph Lamug Martinez** | |  |
|  |  |  |  | |  | |  | |  |
| **Major** | **: CS** |  |  | |  | |  | |  |
| **Title of Assignment**  (if any) | : | |  |  | |  | |  | |
| **Type of Assignment**    **Submission Pattern** | **: Final Project** |  |  | |  | |  | |  |
| **Due Date** | **: 20/06/20** |  |  | | **Submission Date** | | **: 15/06/20** | |  |

The assignment should meet the below requirements.

1. Assignment (hard copy) is required to be submitted on clean paper, and (soft copy) as per lecturer’s instructions.
2. Soft copy assignment also requires the signed (hardcopy) submission of this form, which automatically validates the softcopy submission.
3. The above information is complete and legible.
4. Compiled pages are firmly stapled.
5. Assignment has been copied (soft copy and hard copy) for each student ahead of the submission.

# Plagiarism/Cheating

BiNus International seriously regards all forms of plagiarism, cheating and collusion as academic offenses which may result in severe penalties, including loss/drop of marks, course/class discontinuity and other possible penalties executed by the university. Please refer to the related course syllabus for further information.

# Declaration of Originality

By signing this assignment, I understand, accept and consent to BiNus International terms and policy on plagiarism. Herewith I declare that the work contained in this assignment is my own work and has not been submitted for the use of assessment in another course or class, except where this has been notified and accepted in advance.

Signature of Student: (Name of Student)

1. Ravel Tanjaya

# 1. Introduction

*CGX Taskview* is a Kanban(看板, lit. A board to look) like application that allows us to create and manage tasks. CGX Taskview is more focused on **low latency** real time data communication, this is especially useful for task which require low latency communication.

You can view the demo of the app on <https://www.youtube.com/watch?v=QvrJXd9-xA4>

## 1.1 Background

In the age of technology internet has became a very crucial part of our life, it even replaces the traditional communication way, people start to prefer communication via internet, and even now the trend of real time communication is uprising.

In the other hand, task management is a very important task to do in each project. This ensure the whole project work as intended and this also work as a bidirectional communication channel. One of the most popular way of project management is Kanban, which is being used by several Application like Trello.

This is the reason why I try to combine both concept to create a real time Kanban-like application with low latency.

## 1.2 Implementation

The app that I am going to create is a JavaFX based GUI application, that will connect to JSON(Javascript Object Notation) REST API to gather and alter data, I also created 2 companion\* app for this project which is for Web and Mobile.

**NOTE: The companion app is not written in Java, *Web client* (Vue.js) and *Mobile* (Flutter) is created as a proof of concept to show the real-time capabilities of the main JavaFx client.**

### 1.2.1 Java Version

In the app is use JDK 8, this is done as JDK 8 if the most widely installed JDK, and because JDK 8 is the last version of JDK that is supported by Oracle that still have JavaFX library packed inside

Since JDK 11, JavaFx is not included in the JDK, We need to manually add the library using OpenJFX.

https://www.oracle.com/technetwork/java/javase/11-relnote-issues-5012449.html

### 1.2.1 GUI Library

The GUI(Graphical User Interface) that I use here is JavaFx, my main reason to choose JavaFx over Swing or SWT Toolkit is:

* JavaFx is more modern that Swing or SWT
* Even though Swing is included in every JDK releases, JavaFx uses a more modern design pattern resulting in a better development environment.
* JavaFx uses FXML markup and CSS, making it really easy to create and style the GUI (Which can also be implemented with Glunon Scene Builder)
* SWT is not only an abstraction layer, but it also depens on native library of each platform, thus not permitting cross compilation/universal compilation.
* JavaFx solves GUI scaling better than SWING.

### 1.2.2 REST Server

For this application I do not implement my own rest server solution with Java, as this may take too much time, so I outsource the task to Google's Firebase.

Google Firebase itself offer 2 types of database which is Realtime Database and Cloud Firestore.

In order to fulfill the target of getting a real time system I decided to use Realtime Database as it provides lower latency in comparison to Cloud Firestore.

https://firebase.google.com/

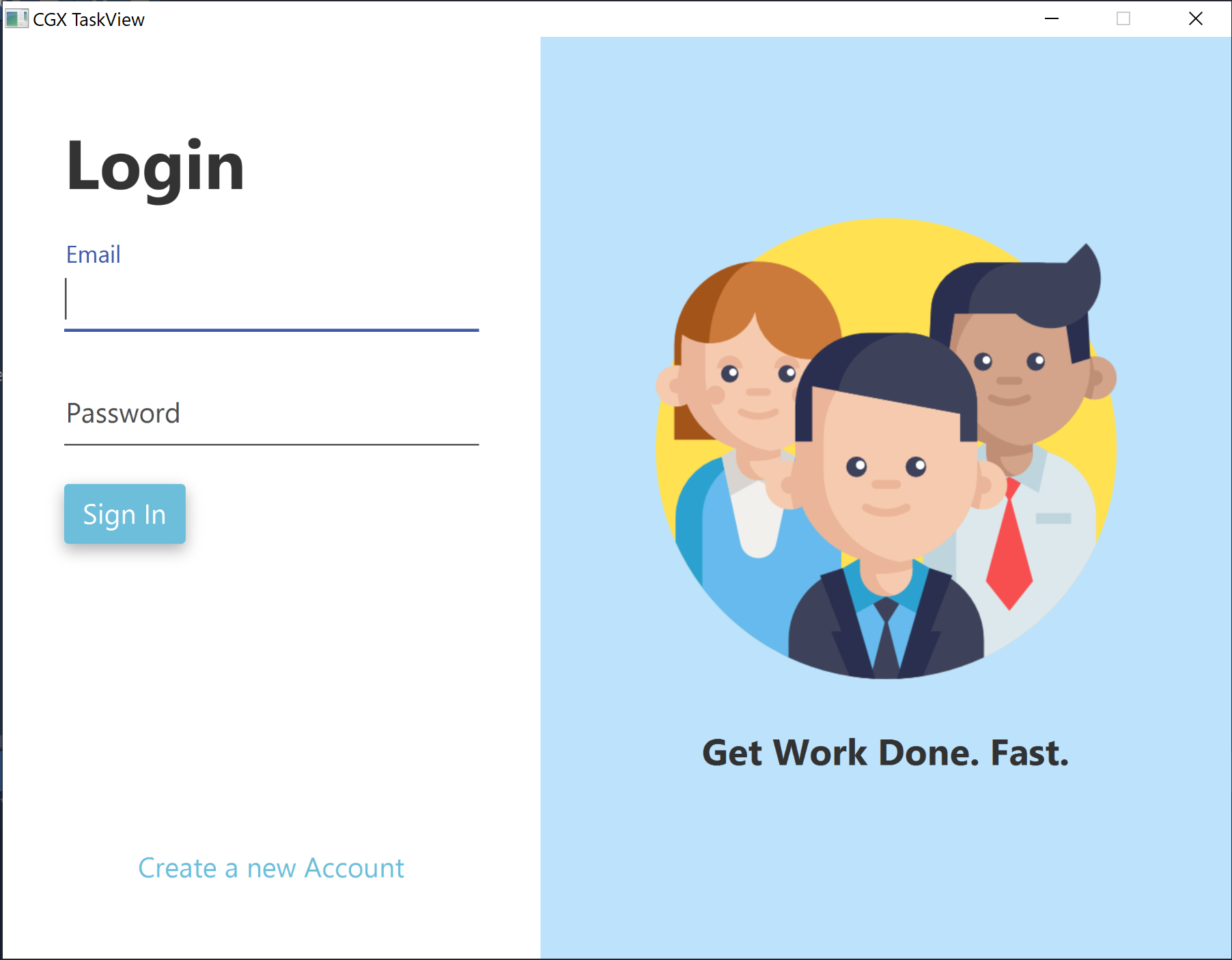
# 2. Documentation

## 2.1 Scenes

The App itself consisted of 3 Screen(Scenes):

* Login/Register
* Project View
* Project Editor

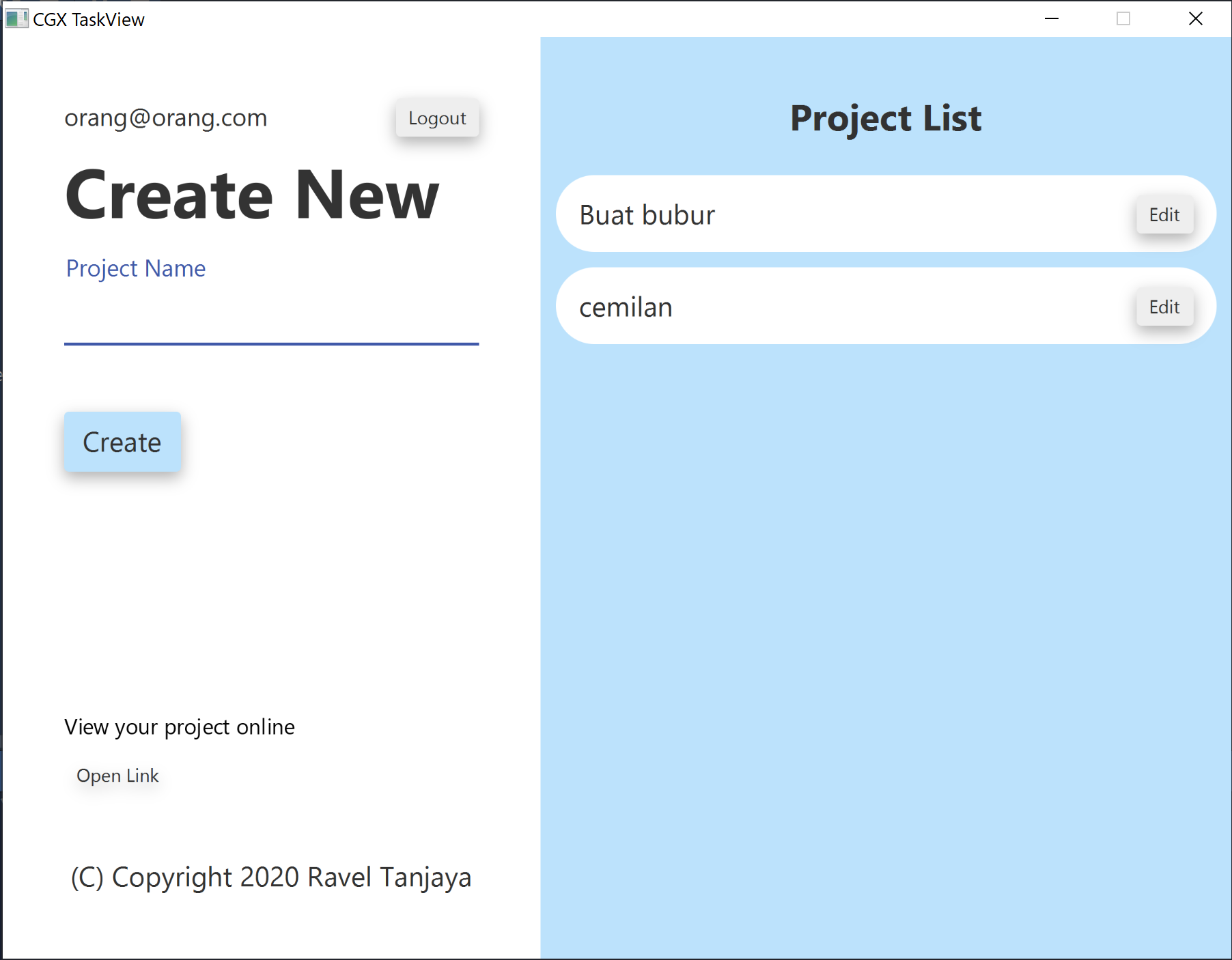
### 2.1.1 Login/Register View



Login/Register view allows user to register or login, this screen will also write user token to a file on login/register.

This View also support input validation, to prevent unwanted request to the server.

### 2.1.2 Project View

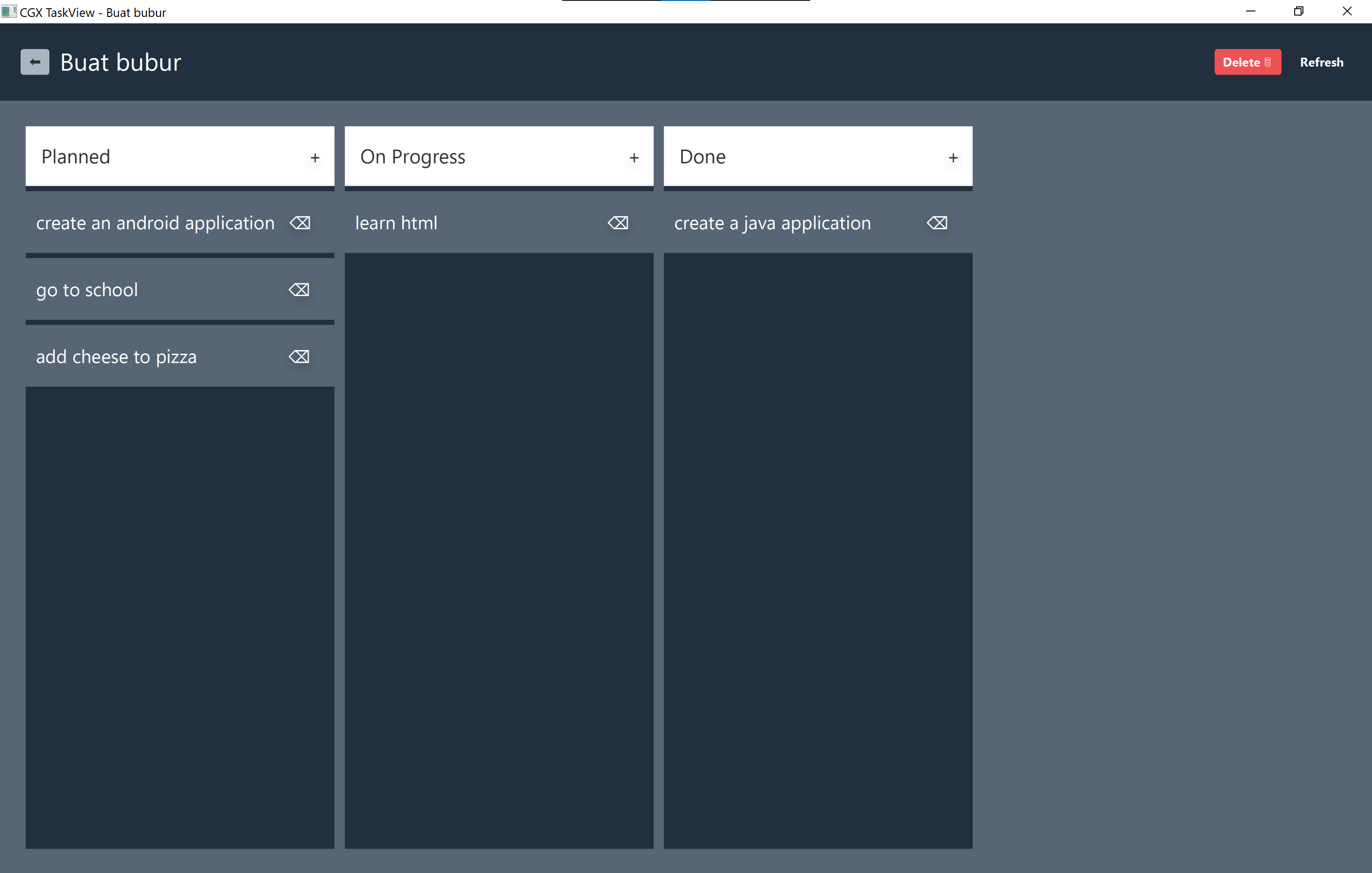


Project View

After the user login, they will be taken to the project view, here they can create and edit their projects, there is also the open link button that will open a browser and allows other user to view their projects.

There is also logout button, this will logout the user and delete the token file.

### 2.1.3 Project Edit



Project Edit

If the user is taken to the project edit page, they will be taken to the project edit page, here the can delete a task or add a task, they can also delete the whole project.

The task that they add here is updated real time on the other companion app.

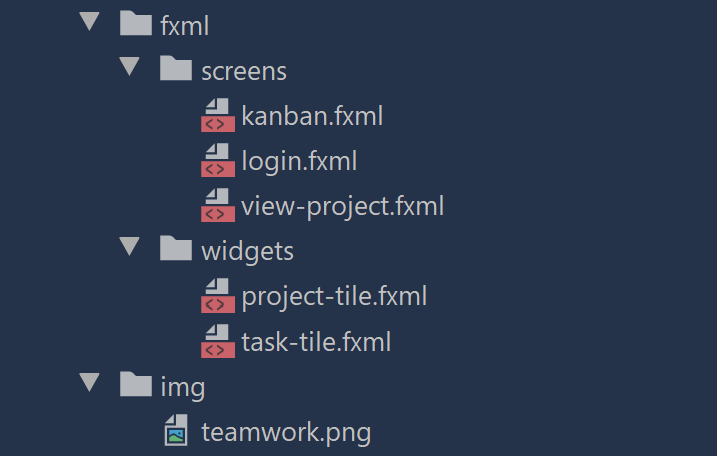
## 2.2 Code Documentation

All documentation is also available as a **JavaDoc** created on /Desktop/doc/index.html

The JavaFx client consists of 2 main file type:

* Java class (The logic of the applicaiton)
* FXML & CSS (The Interface of the application)

### 2.2.1 FXML & CSS



FXML Structure

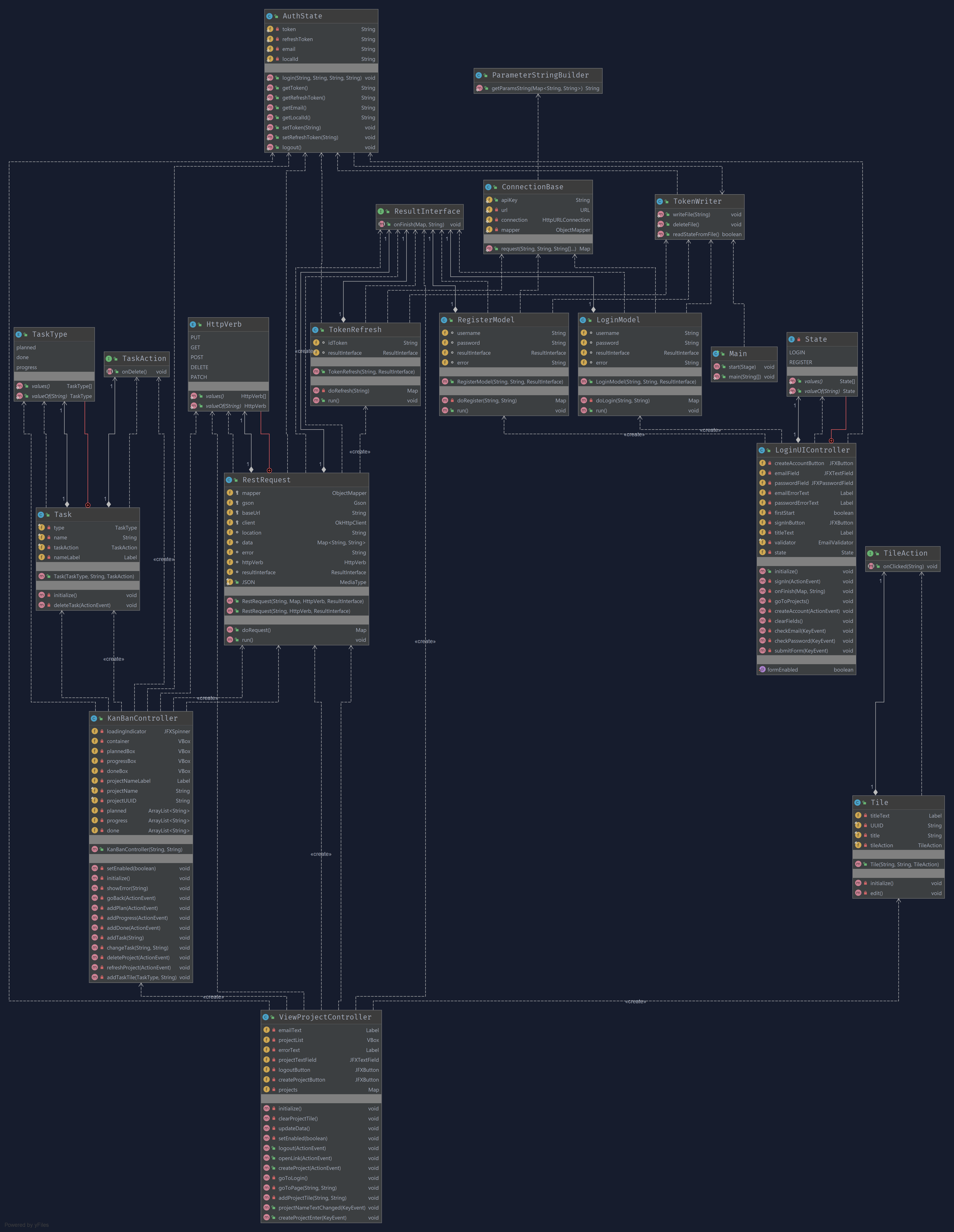
The FXML structure consists of screens and widgets,

Screens contains FXML markup for all scene/view.

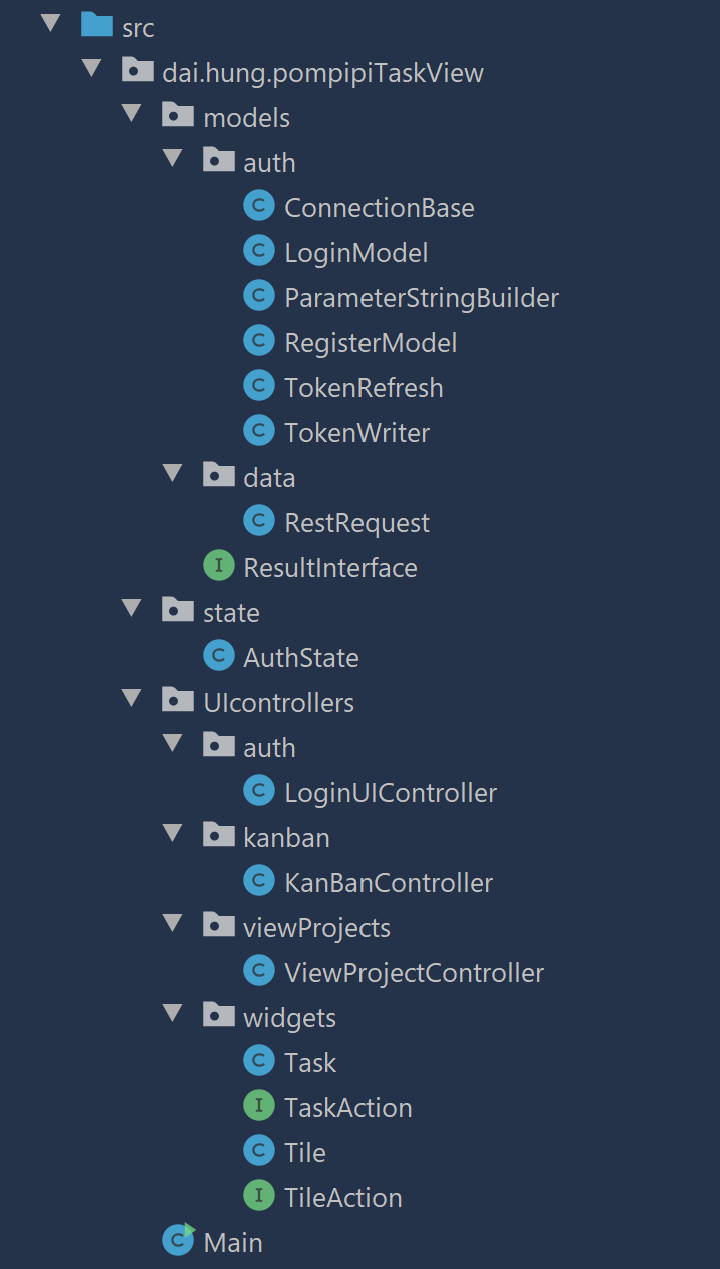
Widgets contains the components of the project view and/or project edit.

### 2.2.2 Java Classes

**The UML of the project is also available on /UML.png or /CGX-Taskview.uml**



The code is structured as follows:



Code

#### 2.2.2.1 Main.java

Main.java is the main entry point of the application, it also determines where the app should go, for example if an JWT token file is found the app will redirect to the project view file, else it will show login screen.

#### 2.2.2.2 UIContollers package

This package contains all ui controller (FXML Controller for the JavaFx application) including the reusable widget.

##### 2.2.2.2.1 Auth package

Contains login related FXML controller

###### 2.2.2.2.1.1 LoginUiController

This is the FXML Controller that controls the login screen.

it manages:

* Form validation
* Call REST Request class

##### 2.2.2.2.2 KanBan package

This is the package that contains the FXML controller related to the project edit.

###### 2.2.2.2.2.1 KanBanController

This is the FXML Controller for project edit controller.

it manages:

* Calling REST request for addition and deletion
* Fetch data on startup

##### 2.2.2.2.3 viewProject package

This package contains the FXML controller related to project View

###### 2.2.2.2.3.1 ViewProject

The FXML Controller of viewproject

it manages:

* Logout and calling delete JWT
* fetch all projects
* open a browser to view the link online

##### 2.2.2.2.4 Widget packege

Contains all FXML controller that manages resuable widget.

###### 2.2.2.2.4.1 Task

A FXML controller that show a task

###### 2.2.2.2.4.2 TaskAction (Interface)

A Interface to implements a callback for the task class as event handler.

###### 2.2.2.2.4.1 Tile

A FXML controller that show a tile

###### 2.2.2.2.4.2 TileAction(Interface)

A Interface to implements a callback for the tileclass as event handler.

#### 2.2.2.3 State

This package contains a class for authentication state management,

##### 2.2.2.3.1 AuthState

this is a static class that contains all of the current authentication state

all function in this class is static as this implement the global access pattern in order to have 1 source of uniform information for the whole app.

#### 2.2.3.4 Models

This package contains all data management related classes

##### 2.2.3.4.1 Result Interface

the Interface that acts as a callback for all of models network request.

##### 2.2.3.4.2 Auth

###### 2.2.3.2.4.1 Connection Base

A base static class mainly used by the authentication system to do http request.

###### 2.2.3.2.4.2 Login model

A class that uses method from connection base to login and request id token and request token from firebase authentication.

###### 2.2.3.2.4.3 Parameter String builder

A class that is used by connection based to build the request body for the HTTP request that is generated by the connection base.

###### 2.2.3.2.4.4 Register Model

A class that uses method from connection base to register new user and request id token and request token from firebase authentication.

###### 2.2.3.2.4.5 Token Refresh

A class that refresh the expired id token using the refresh token. this is automatically run if the token is expired.

###### 2.2.3.2.4.6 Token Writer

A class that manage the file that contains the JWT token.

##### 2.2.3.4.2 Data

###### 2.2.3.2.4.1 Data Request

A class that handle all of the http request which is not part of the authentication.

All documentation is also available as a **JavaDoc** created on /Desktop/doc/index.html

# 3. How to use

### Desktop

Download the release binary then run the app with jdk8.

### Mobile

Install the apk from the release page

## How to compile

### Desktop

This application only works with jdk8, as jdk8 still package javafx inside it's distribution. **OpenJDK is not supported**

**To delpoy app with your own server, please change the url and api key from the application**

git clone https://github.com/raveltan/cgx-taskview.git  
cd cgx-taskview/Desktop  
// open your favourite editor

### Mobile

Please make sure you have install [flutter sdk](https://flutter.dev/)

git clone https://github.com/raveltan/cgx-taskview.git  
cd cgx-taskview/Mobile  
flutter build apk

# 4. Attribution

Icons made by [Freepik](https://www.flaticon.com/authors/freepik) from <www.flaticon.com>