



Projet annuel 3AL

Axel DJENADI, Fidel MONTEIRO, Halisia HALIFA

3ESGI
2020 - 2021

Table des matières

Table des matières	1
Présentation générale	2
1.1 Introduction du projet	2
1.2 Présentation des fonctionnalités	2
1.2.1 Authentification	2
1.2.2 Partager ou récupérer un plat	2
1.2.3 Communication à l'aide d'un chat	2
1.2.4 Laisser une évaluation	3
Gestion de projet	3
2.1 Présentation de l'outil mis en place	3
2.2 Explications des différentes classes	3
2.3 Exemple d'utilisation	4
Base de données de l'application	8
Organisation des classes	8
Partie API	9
Fonctionnement	9
Webscraping	10
5.1 Utilisations	10
5.2 Exemples	10

1. Présentation générale

1.1 Introduction du projet

Mercifood est une application web qui permet de partager de la nourriture autour de soi. Un utilisateur souhaitant cuisiner aura la possibilité de mettre à la disposition d'autres utilisateurs, proches de chez lui, les informations relatives à son plat.

Ces informations contiennent les aliments qui composent son plat à partager, le nombre de personnes maximum, il aura aussi la possibilité d'ajouter des photos.

Les utilisateurs intéressés pourront ensuite réserver le plat qui leur convient, et auront la possibilité de laisser une évaluation sur leur expérience.

1.2 Présentation des fonctionnalités

1.2.1 Authentification

Afin de proposer le partage ou d'avoir la possibilité de réserver un plat, il est nécessaire d'avoir un compte sur Mercifood et de se connecter.

Une adresse mail ou un numéro de téléphone est donc obligatoire afin de pouvoir obtenir des informations sur la réservation en cours au cas où l'utilisateur se déconnecte avant de récupérer son plat.

Chaque compte sera accompagné d'un profil, dans lequel les exigences de chaque utilisateur sera indiqué, notamment pour pouvoir réaliser des filtres sur leurs recherches de plats.

1.2.2 Partager ou récupérer un plat

En tant qu'utilisateur connecté, je souhaite avoir accès à ces deux fonctionnalités. Sur l'écran d'accueil j'aurais la possibilité de choisir quelle action je souhaite réaliser :

- Si je choisis de partager un plat, j'aurai accès à un formulaire à remplir avec les informations de celui-ci.
- Si je choisis de rechercher un plat, j'aurai accès à un listant les plats disponibles. Les données affichées seront cliquables afin d'avoir accès aux informations données par l'utilisateur qui partage et de vérifier la compatibilité entre le plat et notre profil (allergies, régime alimentaire, etc).

A ce jour, les deux fonctionnalités suivantes n'ont pas encore été implémentées:

1.2.3 Communication à l'aide d'un chat

En tant qu'utilisateur connecté, je souhaite pouvoir échanger avec d'autres utilisateurs par un chat afin de faciliter le processus de réservation. Lorsqu'un utilisateur a partagé un plat et que je le réserve, je pourrai si besoin lui poser des questions ou répondre aux siennes.

Ce chat permet surtout à l'utilisateur qui partage d'avoir toutes les conversations entre tous les utilisateurs ayant réservé réunies dans une seule application, ce qui peut aussi être utile pour envoyer des messages génériques qui les concernent tous.

1.2.4 Laisser une évaluation

En tant qu'utilisateur connecté et après avoir effectué une réservation, j'aurai la possibilité de laisser une évaluation une fois mon plat récupéré. Cette évaluation est composée d'une note sur 5 ainsi que d'un commentaire (temps d'attente, communication avec l'utilisateur qui partage, qualité/conformité du plat, etc). Cette évaluation sera visible sur le profil de l'utilisateur qui a fourni le plat.

2. Gestion de projet

2.1 Présentation de l'outil mis en place

Afin de gérer l'équipe de projet, un outil réalisé en Java a été mis en place en parallèle de l'application. Cet outil nous permettra de gérer l'ensemble des tâches définies pour ce projet, soit en ligne de commande, soit via une interface.

Le code est divisé en plusieurs packages : database, models et view:

- Le package models contient les différentes classes du projet
- Le package view contient les fichiers fxml utilisés pour l'interface graphique ainsi que leurs contrôleurs

2.2 Explications des différentes classes

Classe Home:

Cette classe représente l'accueil, elle permet de regrouper l'ensemble des projets créés ainsi que tous les membres des projets.

L'accueil est composé de :

- un nom
- une liste de projets
- une liste de membres

Classe Project:

Cette classe permet de réunir les informations relatives aux membres ainsi qu'aux tâches d'un projet.

Un Projet est donc composé de :

- un nom
- une liste de membres
- une liste de tâches
- un prochain point de rendez-vous
- une deadline
- un accueil

Classe Task:

Cette classe regroupe tout ce qui concerne une tâche d'un projet.

Une tâche est composée de:

- un nom
- un propriétaire
- une deadline
- une date de création
- un état (cf. Énumération State)
- un commentaire
- un projet

Classe Member:

Cette classe regroupe tout ce qui concerne un membre d'un projet

Un membre est composé de:

- un nom
- une liste des tâches qui lui ont été assignées

Enumeration State:

Cette énumération permet de lister les différents états qu'une tâche peut avoir :

- To do : l'état par défaut attribué à une tâche lors de sa création
- In progress
- Done

Classe Database:

Cette classe sert de lien entre le projet et la base de données MySQL. Au lancement du programme, elle permet de faire la connexion à la base de données et de récupérer les valeurs existantes. Elle contient également les fonctions nécessaires à l'écriture dans la base.

2.3 Exemple d'utilisation

Afin de commencer, il est nécessaire de réaliser les étapes suivantes :

- 1) Se placer dans le répertoire backend/java/planning et ouvrir un terminal
- 2) Il faut mettre en place la base de données. Deux options sont possibles :
Si votre docker est installé sur votre machine, la base de données peut être lancée à l'aide du docker-compose disponible dans le projet.
Il suffira de lancer la commande `docker-compose up`, et si besoin, d'ouvrir phpMyAdmin dans le navigateur avec l'adresse suivante : localhost:8081. (mêmes identifiants)

Dans le cas échéant, créez une base de données mySQL avec:

- nom de la base de données : home_projects
- utilisateur : root
- mot de passe root

- port : 3306

En se positionnant à la racine du projet, un export de la base de données "home_projects.sql" est disponible dans le répertoire /backend/database.

- 3) Pour lancer le projet, ouvrir un autre terminal et exécuter la commande suivante:
`java -jar planning.jar` afin de lancer le programme en ligne de commande, ou alors
`java -jar planning_interface.jar` afin de le lancer avec l'interface graphique
- 4) Pour se connecter en tant qu'invité :
 - user : guest
 - password : guest3AL

Dans la version en lignes de commandes, la première question posée à l'utilisateur est de choisir si il veut créer un nouveau projet où gérer un projet existant :

```
What do you want to do ?  
1. Create a new project  
2. Update an existing project  
3. Nothing
```

Menu principal

Si l'utilisateur choisit 1, un nouveau projet sera créé et aura accès au menu d'édition d'un projet.

S'il choisit 2 il choisira un projet à modifier et aura ensuite accès au menu d'édition.

S'il choisit 3, il aura un récapitulatif de l'ensemble des projets existants.

```
What do you want to do in this project ?  
1. Change the name  
2. Add a member  
3. Remove a member  
4. Add a task  
5. Update an existing task  
6. Update next appointment  
7. Update the deadline  
8. Remove from home  
9. Nothing
```

Menu d'édition d'un projet

Ce fonctionnement sera le même pour les autres actions. Si l'utilisateur souhaite ajouter une tâche à ce projet, après l'avoir créée il aura accès au menu d'édition pour cette tâche, de même pour modifier une tâche existante.

```
What do you want to do with this task ?
1. Update the state
2. Update the owner
3. Update the deadline
4. Write a comment
5. Remove from project
6. Nothing
```

Menu d'édition d'une tâche

Lorsque l'utilisateur termine une action, il appuie sur l'option "Nothing" qui sert de retour. Après avoir édité une tâche, il sera redirigé vers le menu d'édition du projet en cours, et s'il appuie encore sur "Nothing", il obtiendra le récapitulatif des projets existants.

Exemple de récapitulatif obtenu:

```
[PROJECT Mercifood
-----
Members : - Halisia - Fidel - Axel
Next Appointment : 2021-06-25
Deadline : null
Tasks : [
  Api
  ---
  Owner : null
  State : DONE
  Date : 2021-06-25
  Deadline : 2021-04-23
  Comment :
,
  JavaFx interface
  ---
  Owner : null
  State : DOING
  Date : 2021-06-25
  Deadline : 2021-06-25
  Comment :
]
, PROJECT WebScrapper
-----
Members :
Next Appointment : null
Deadline : null
Tasks : []
]
```

Sur l'interface graphique, le principe est le même. On crée ou on édite un projet et nous avons accès à l'ensemble de ses informations :

The 'Project Planning' window features a table on the left and a details pane on the right.

Projects	
Mercifood	
WebScrapper	

Project Details:

Name: Mercifood

Deadline:

Next appointment: 2021-06-25

Members: - Halisia
- Fidel
- Axel

Tasks: - Api
- JavaFx interface

Buttons: New... Edit... Delete

Page d'accueil de l'interface graphique

The 'Edit Project' dialog box contains the following fields and buttons:

Name:

Deadline:

Next appointment:

Tasks:

Members:

Buttons: OK Cancel

Page d'édition d'un projet

3. Base de données de l'application

3.1 Organisation des classes

Afin de développer les fonctionnalités décrites ci dessus, le diagramme des classes se présente comme ceci:

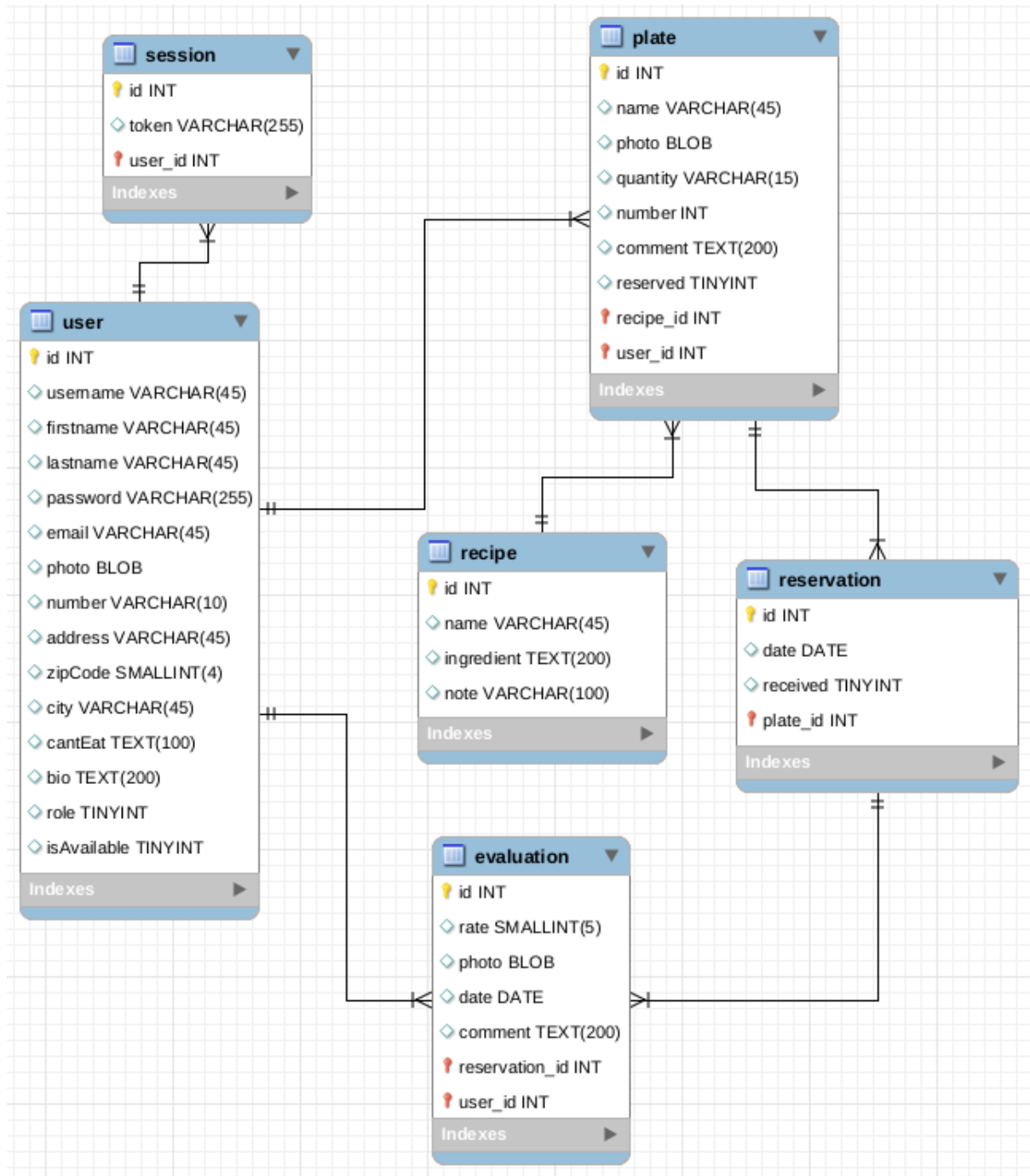


Table User:

Cette table représente un utilisateur possédant un compte sur Mercifood. Certains attributs sont optionnels tels que photo, bio ou encore cantEat. Ces informations ne sont pas demandées lors de l'inscription mais peuvent être mises à jour si besoin.

Table Session:

C'est la table qui gère les sessions pour un utilisateur donné.

Table Plate:

Cette table représente un plat appartenant à un utilisateur. Lorsqu'un plat est mis en ligne, il apparaît dans la liste des plats disponibles avec l'attribut reserved égal à false par défaut.

Table Recipe:

Une recette représente l'ensemble des informations relatives à un plat.

Table Reservation:

Cette table contient les informations relatives à une réservation pour un plat donné.

Table Evaluation:

Cette table sert de lien entre la note qu'un utilisateur donne à un autre utilisateur après avoir choisi un de ces plats.

3.2 Utilisation

En se positionnant à la racine du projet, un export de la base de données "mfdb.sql" est disponible dans le répertoire /backend/database.

4. Partie API

4.1 Les différentes routes

Le document ci-dessous liste l'ensemble des endpoints accessibles depuis notre API.

CRUD Mercifood API			
Method	Path	Role	Description
POST	/auth/subscribe	0	Create a new user
POST	/auth/login	0	Create a new session
POST	/recipes	0	Create a new recipe

POST	/plates	0	Create a new plate
POST	/evaluations	0	Create a new evaluation
POST	/reservations	0	Create a new reservation
POST			
GET	/users	0	Show all users
GET	/users/{id}	0	Show a specific user by id
GET	/users/{name}	0	Show a specific user by name
GET	/users/availables		Show all available user
GET	/recipes	0	Show all recipes
GET	/recipes/{id}	0	Get a specific recipe
GET	/recipes/{name}	0	Get a specific recipe by name
GET	/plates	0	Show all plates
GET	/plates/{id}	0	Show a specific plate by id
GET	/plates/{name}	0	Show a specific plate by name
GET	/evaluations	0	Show all evaluations
GET	/evaluations/{username}	0	Create a new plate evaluation by username
GET	/reservations	0	Show all reservations
GET	/reservations/{id}	0	Get a specific reservation
GET			
PUT	/users/{id}	1	Update an existing user
PUT	/recipes/{id}	1	Update an existing recipe
PUT	/plates/{id}	1	Update an existing plate by id
PUT	/plates/{name}	1	Update an existing plate by name
PUT	/reservations/{id}	1	Update an existing reservation
PUT		1	
DELETE	/users/{id}	1	Delete an existing user
DELETE	/users/{id}	0	Unsubscribe an existing user
DELETE	/recipes/{id}	1	Delete an existing recipe
DELETE	/plates/{id}	1	Delete an existing plate id
DELETE	/plates/{name}	1	Delete an existing plate by name
DELETE	/reservations/{id}	1	Delete an existing reservation
DELETE			

Role => 0: User 1: Admin

4.2 Utilisation

4.2.1 Lancement

Pour pouvoir utiliser l'API, il faut lancer le projet backend :

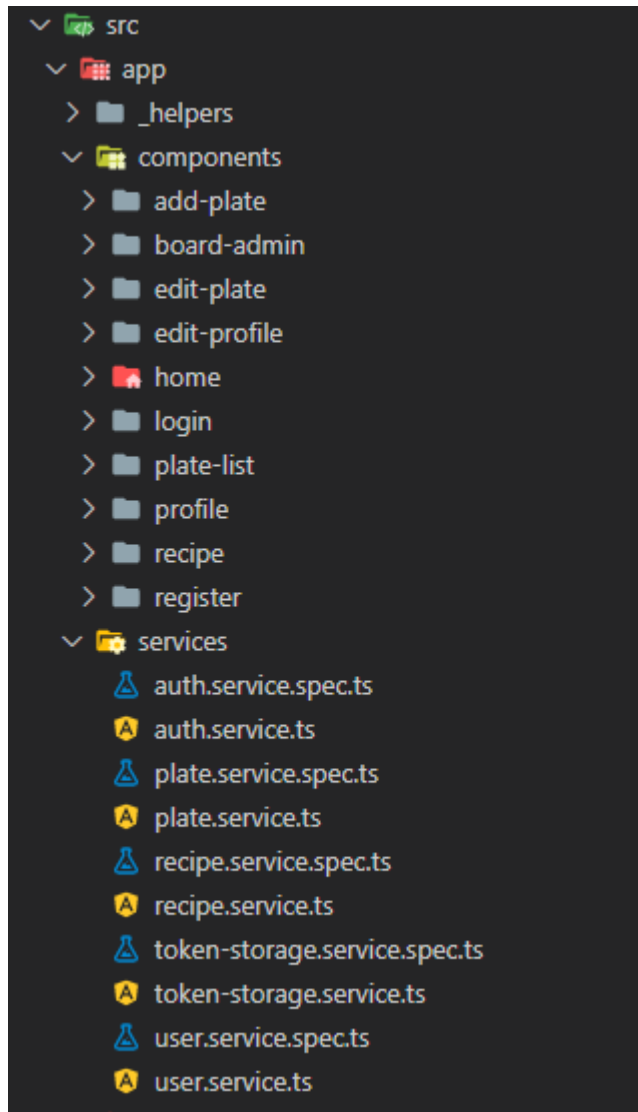
- Depuis la racine du projet, se placer dans le répertoire backend/api/mercifood et ouvrir un terminal
- Exécuter la commande : **npm i** puis **npm start**
- L'utilisation peut commencer une fois que le serveur est à l'écoute sur le port 3000

4.2.2 Exemple de requêtes

POST	/auth/subscribe	0	Create a new user
{ "firstname":"Alice", "lastname":"Bob", "username":"aliceb", "password":"mypassword", "email":"aliceb@gmail.com" }			
POST	/auth/login	0	Create a new session
{ "username":"aliceb", "password":"mypassword" }			
POST	/recipes	0	Create a new recipe
{ "name":"myrecette", "ingredient":"ing1,ing2,ing3", "note":"come notes" }			
POST	/plates	0	Create a new plate
{ "name":"delicious plate", "quantity":4, "number":5, "photo":"/path/picture", "comment":"comment" }			

5. Partie Front-end

5.1 Architecture du code



5.1.1 Les components

- add-plate: Ajouter un nouveau plat dans la liste des plats disponibles
- edit-plate: Éditer un plat existant
- edit-profile: Mettre à jour ses informations d'utilisateur
- home: La page d'accueil
- login: Le formulaire de connexion
- plate-list: La liste des plats disponibles
- profile: Les informations de l'utilisateur
- register: le formulaire d'inscription

5.1.2 Les services

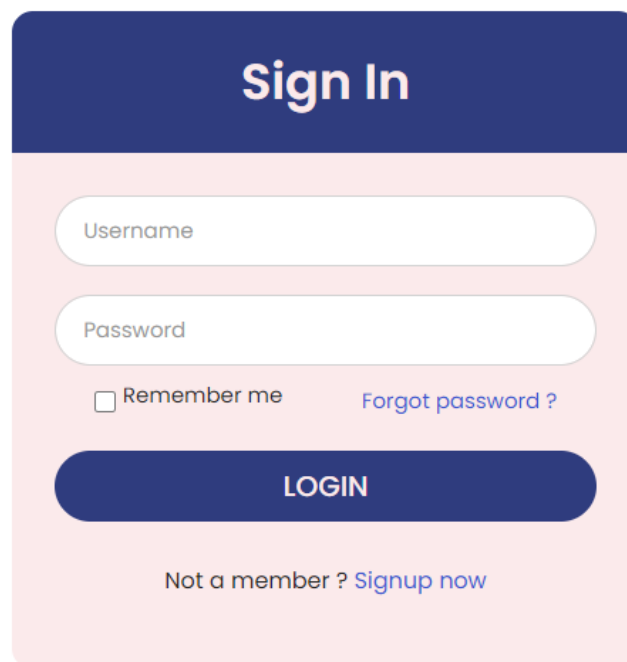
- auth: S'inscrire et se connecter
- plate: Requêtes CRUD d'un plat
- recipe: Ajouter une recette
- token-storage: Sauvegarder un token/utilisateur et obtenir un token/utilisateur
- user: Requêtes CRUD d'un utilisateur

5.2 Utilisation

Pour pouvoir accéder au site, il faut lancer le projet front-end:

- Depuis la racine du projet, se placer dans le répertoire frontend/mercifood et ouvrir un terminal
- Exécuter la commande : `npm i` puis `npm start`
- Une fois que le server est à l'écoute sur le port 4200, ouvrir un navigateur et accéder à l'adresse : `https://localhost:4200`

Pour commencer, appuyer sur Se connecter en haut à droite afin d'accéder au formulaire de connexion:

A sign-in form with a dark blue header containing the text "Sign In". Below the header, there are two white input fields with rounded corners, labeled "Username" and "Password". Under the "Password" field, there is a checkbox labeled "Remember me" and a link "Forgot password ?" in blue. A dark blue button with the text "LOGIN" in white is positioned below these elements. At the bottom of the form, there is a link "Not a member ? Signup now" in blue.

Appuyer sur Signup now si besoin, pour créer un compte. Une fois la connexion établie, on arrive sur la page d'accueil.

Le menu à gauche permet de naviguer sur le site, par exemple pour partager un plat.

Mercifood

Partager un plat

Plats à réserver

Ce qui nous renvoie vers la page suivante:

Partager un plat

Nom du plat

Quantité (en grammes)

Nombre de personnes

Commentaires

Photo

Choisir un fichier

Aucun fichier choisi

Partager

Pour accéder aux détail d'un plat publié, cliquer sur Plats à réserver puis cliquer sur le plat désiré :

Search by name

Search

Liste des Plats

test
new plat
new plat
test
gregeg

Remove All

Plate

Name: new plat

Photo:

Quantity: 4

Number: 5

Comment: comment

Status: Pending

Created: 2021-07-24T17:27:11.000Z

Edit

6. Trello

Boards

Jump to...

Board

Mercifood

Private Workspace

Private

FM HH AD

Invite

À faire

...

Progress bar

Plugins - Java

+ Add another card

En cours

...

Progress bar

Documentation de l'API

FM

Progress bar

Pages web - Angular

Progress bar

Idées pour le micro langage

AD

Progress bar

Documentation globale

AD FM HH

Progress bar

Documentation de la BD

AD FM HH

Progress bar

Dessins des IHM

AD

Progress bar

Mise en place de Webscrapping

AD

+ Add another card

Terminé

...

Progress bar

Schéma des classes de la BD

Progress bar

Client Java - Interface graphique

Progress bar

Client Java - Application en lignes de commandes (pouvoir rajouter projets)

HH

Progress bar

Client Java - Sauvegarde et Backup base de données

HH

Progress bar

API - TypeScript

FM

+ Add another card