

Auteur : Zhaojie LU

26/11/2021

Type : STR

Equipe : Virgilia Muselli, Michiel
Chinchole, Zhaojie Lu, Chengyu Yang

Chemin : PISTL/STR

Statut : Initial

Destinataire(s) : Virgilia Muselli, Michiel Chinchole,

Page(s) 11

SafetyLine-Chiffrage

STR

Version	Date	Modifications	Rédacteur(s)
1.0	11/26/2021	Initial	Zhaojie LU, Chengyu YANG

Table des matières

1. Objectif	3
2. Objets métiers	3
3. Services	5
4. Workflow	9
5. Architecture technique	10
6. Plan d'intégration	11
7. WBS de la réalisation	12
8. Annexes	13

1. Objectif

La cérémonie des “chiffrages” est une partie importante de la gestion de projet d'une entreprise. C'est le moment où les équipes se réunissent pour discuter des tâches à traiter lors du prochain sprint. Une fois les tâches acceptées, assignées par l'équipe de développement, chaque membre va planifier le temps d'achèvement prévu pour sa tâche et envoie l'ensemble de ses estimations par mail au chef de projet. Le chef de projet examine et approuve ces plans. Il va ensuite se connecter au GitLab, et ajouter les temps d'achèvement pour chaque tâche.

Pour automatiser cette cérémonie de chiffrage, nous allons construire une page d'application.

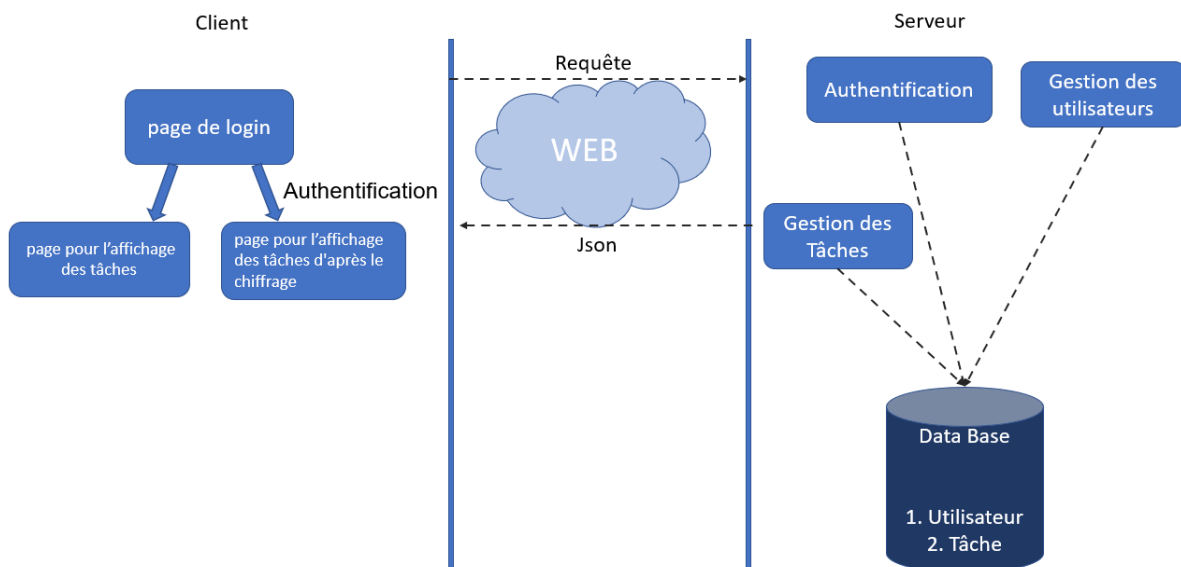
Nous allons définir approximativement les composants de notre système comme l'image ci-dessous. Nous allons le présenter en détail dans la suite.

- Client : Page Web

1. page de Login (pour distinguer le chef du projet et d'autres membres)
2. page User
3. page Admin

- Serveur : Base de données

1. utilisateur
2. tâche



2. Objets métiers

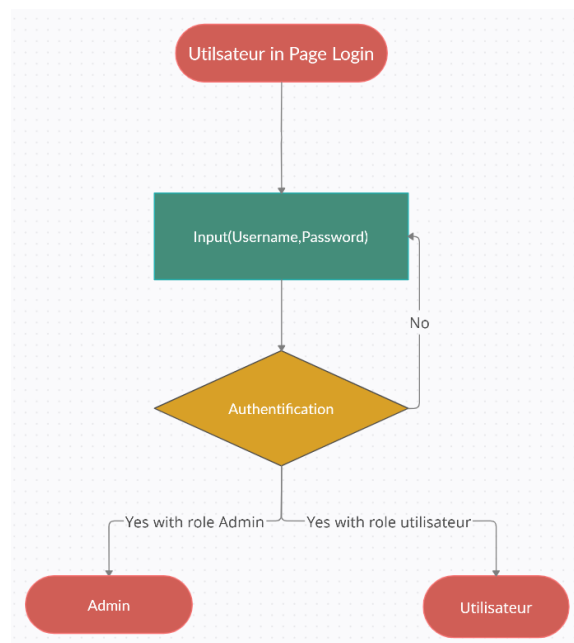
Dans notre projet, nos objets métiers sont forcément des utilisateurs qui vont utiliser cette application et ils ont deux rôles différents. Ils doivent avoir un compte pour se connecter à cette application. Afin de créer un compte, nous avons besoin des attributs correspondants. Pour les données utilisateur, nous les stockerons dans notre couche de persistance.

Nom de l'objet	Utilisateur
Définition	Membre de l'équipe dans le processus de chiffage
Attributs	Username Password Email Role

Exemple dans MySQL : Admin (rôle = 1), User (rôle = 0)

id	username	password	email	role
1	Zhaojie	123456	x1136995140@gmail.com	1
2	Tom	1136995140	tom@gmail.com	1
3	Chef	123456	chef@gmail.com	0

Sur la page initiale, nous aurons un système de connexion (composant page de login) à travers lequel nous pourrions distinguer les rôles des utilisateurs. Parce que selon le rôle de l'utilisateur, la fonction de notre application sera différente.



3. Services

La connexion de l'utilisateur sera le premier service rencontré par l'utilisateur et la seule fonction que nous ayons dans le premier composant.

Nom de Service	Authentification
Définition	L'utilisateur doit saisir les informations du compte, puis nous transmettons les informations aux backend et enfin analysons le compte pour obtenir le rôle de compte actuel.

Lorsque les utilisateurs se connectent, ils peuvent accéder aux composants qui réaliser le processus de chiffrement et la gestion des utilisateurs.

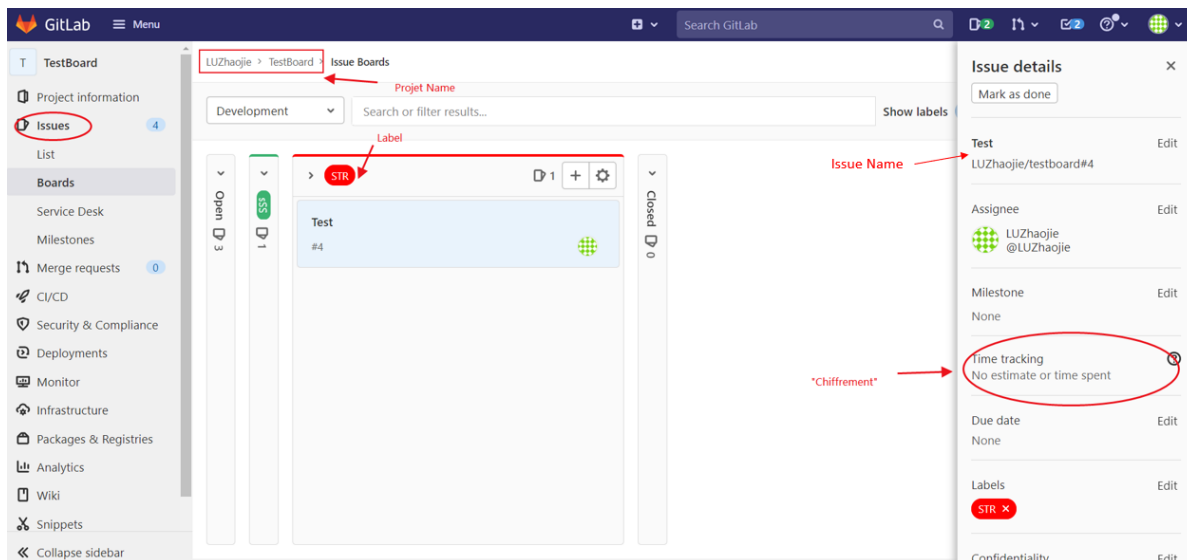
Gestion des utilisateurs :

Nom de Service	Créer/Supprimer Utilisateur
Définition	Si l'utilisateur est un administrateur, il pourra supprimer et ajouter des comptes, et le contenu de la base de données changera également.

Le processus de chiffrage :

Tout d'abord, nous devons présenter quelle est la tâche que nous devons chiffrer et où nous pouvons obtenir.

Exemple de Issue "Test" avec Label "STR" dans Projet "Testboard" :



Dans notre processus normal, toutes les tâches que nous rencontrons dans le processus sont avec des étiquettes fixées, dans notre cas, c'est le "Sprint n+1". Et nous savons que le chiffrage consiste à ajouter un temps d'achèvement estimé à une issue. Afin d'afficher ces issues à distance et d'ajouter un chiffrage sur notre application, nous devons appeler les tâches dont nous avons besoin via l'interface API de Gitlab et stocker leurs informations dans notre base de données.

Exemple : Récupérer les informations des issues avec label "STR"

gitlab api / issues from project + label

GET <https://gitlab.com/api/v4/projects/30171739/issues?labels=STR>

Résultat :

```
[{"id":98008675,"iid":4,"project_id":30171739,"title":"Test","description":"Test Issue For STR","state":"opened",
"created_at":"2021-11-26T14:37:09.334Z","updated_at":"2021-11-26T14:37:09.334Z","closed_at":null,"closed_by":null,"labels":["STR"],
"milestone":null,"assignees":[{"id":5392664,"name":"LUZhaojie","username":"LUZhaojie","state":"active","avatar_url":"https://secure.gravatar.com/
avatar/0736cf4345fa369c72b02d74416348ac?s=80\u0026d=identicon","web_url":"https://gitlab.com/LUZhaojie"}],"author":{"id":5392664,
"name":"LUZhaojie","username":"LUZhaojie","state":"active","avatar_url":"https://secure.gravatar.com/avatar/0736cf4345fa369c72b02d74416348ac?
s=80\u0026d=identicon","web_url":"https://gitlab.com/LUZhaojie"},"type":"ISSUE","assignee":{"id":5392664,"name":"LUZhaojie",
"username":"LUZhaojie","state":"active","avatar_url":"https://secure.gravatar.com/avatar/0736cf4345fa369c72b02d74416348ac?s=80\u0026d=identicon",
"web_url":"https://gitlab.com/LUZhaojie"},"user_notes_count":0,"merge_requests_count":0,"upvotes":0,"downvotes":0,"due_date":null,
"confidential":false,"discussion_locked":null,"issue_type":"issue","web_url":"https://gitlab.com/LUZhaojie/testboard/-/issues/4","time_stats":
{"time_estimate":0,"total_time_spent":0,"human_time_estimate":null,"human_total_time_spent":null},"task_completion_status":{"count":0,
"completed_count":0},"blocking_issues_count":0,"has_tasks":false,"_links":{"self":"https://gitlab.com/api/v4/projects/30171739/issues/4",
"notes":"https://gitlab.com/api/v4/projects/30171739/issues/4/notes","award_emoji":"https://gitlab.com/api/v4/projects/30171739/issues/4/
award_emoji"},"project":{"https://gitlab.com/api/v4/projects/30171739"},"references":{"short":"#4","relative":"#4","full":"LUZhaojie/testboard#4"},
"moved_to_id":null,"service_desk_reply_to":null}]
```

Exemple : Chiffrement par API

gitlab api / add estimate time

POST	▼	https://gitlab.com/api/v4/projects/30171739/issues/4/time_estimate?duration=1w3h10m
------	---	---

Résultat :

STR

1

+

⚙

Test

#4 ⌚ 43h 10m

Closed 0

Issue details

✕

Mark as done

Test

Edit

LUZhaojie/testboard#4

Assignee

Edit

LUZhaojie

@LUZhaojie

Milestone

Edit

None

Time tracking

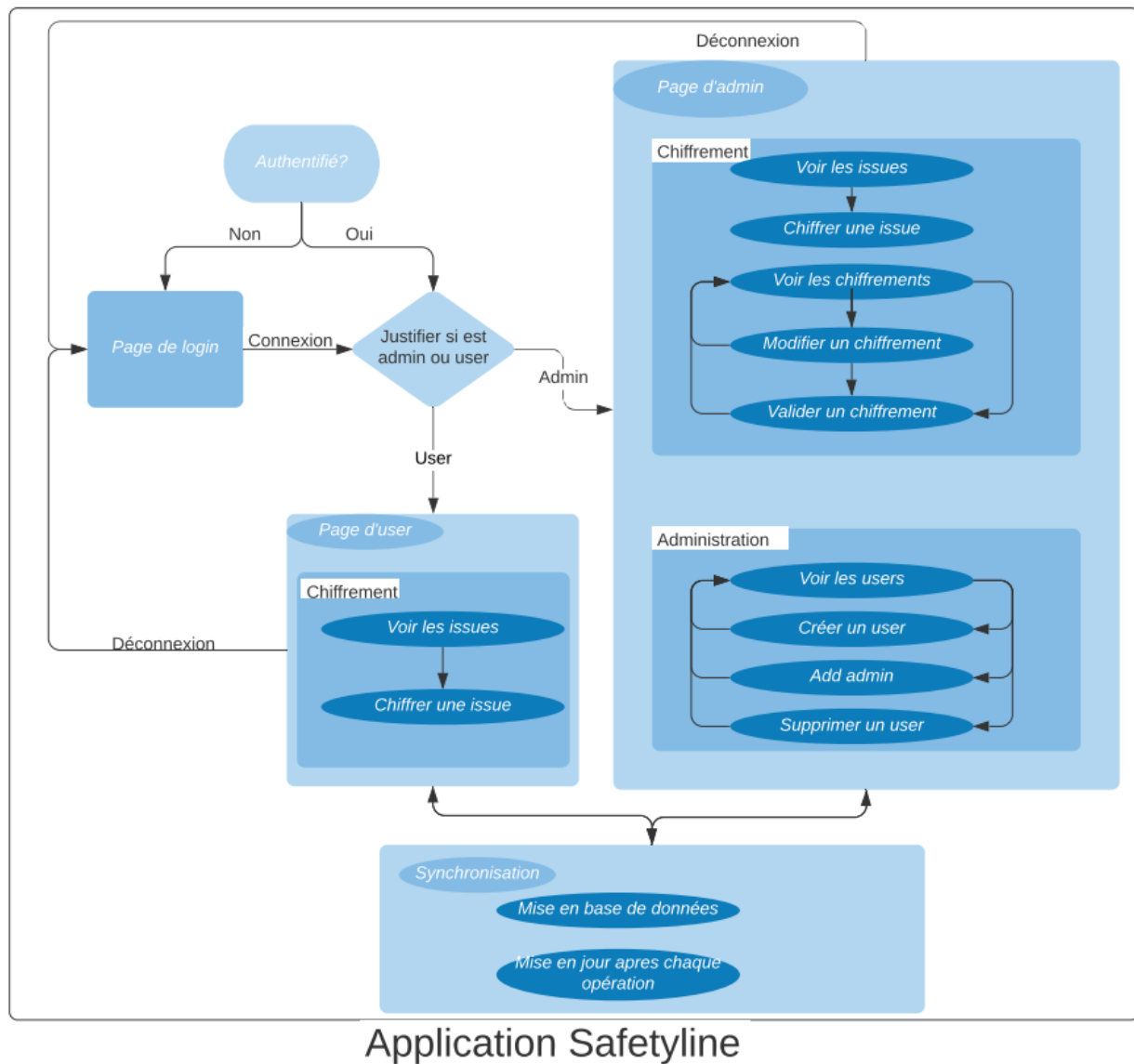
?

Estimated: 43h 10m

A la fois, mise en place de notre base de données sur les issues, nous pouvons fournir des services suivants pour réaliser le processus de chiffrage.

Nom de Service	Définition
Affichage des Issues	L'utilisateur peut voir les informations sur des tâches non chiffrées (vient de notre base de données)
Chiffrage	L'utilisateur peut ajouter un temps d'achèvement estimé à une issue selon la règle de gitlab. Cette opération va juste changer les données dans la base de données.
Affichage global	Si l'utilisateur est un administrateur, il peut voir les informations sur toutes les tâches incluant les chiffrements ajoutés par d'autres membres.
Modification	Si l'utilisateur est un administrateur, il pourra supprimer et modifier des issues chiffrées.
Validation	Si l'utilisateur est un administrateur, à la fois il est d'accord avec des issues chiffrées, il peut les mettre en ligne sur Gitlab (par API Gitlab).

4. Workflow



5. Architecture technique

Frontend:

Selon les besoins de l'équipe, nous utiliserons le Framework Angular pour construire la page frontend qui est une plateforme pour la création d'une web application au niveau de l'entreprise. Il s'agit d'un cadre frontal relativement complet, comprenant des services, des modèles, une modularisation, un routage etc.



Backend:

Pour l'environnement de développement actuel, Spring boot peut être considéré comme le framework JAVA le plus pratique aujourd'hui, c'est une déclinaison du framework classique de Spring qui permet essentiellement de réaliser des microservices. Il automatise le processus de démarrage complexe du Spring. Il permet donc de créer une API de services très simplement.

Ensuite, notre projet doit interagir avec la base de données (MySQL), nous avons donc également besoin d'un moyen pratique pour le réaliser. Nous avons donc choisi le framework MyBatis-Plus. MyBatis-Plus est un puissant outil amélioré de MyBatis. Il fournit de nombreuses opérations efficaces. MyBatis-Plus peut injecter automatiquement des fragments SQL de base, disposer d'un wrapper de condition puissant et flexible, son utilisation peut vous faire gagner beaucoup de temps de développement. Il peut être facilement ajouté à Spring boot via Maven.

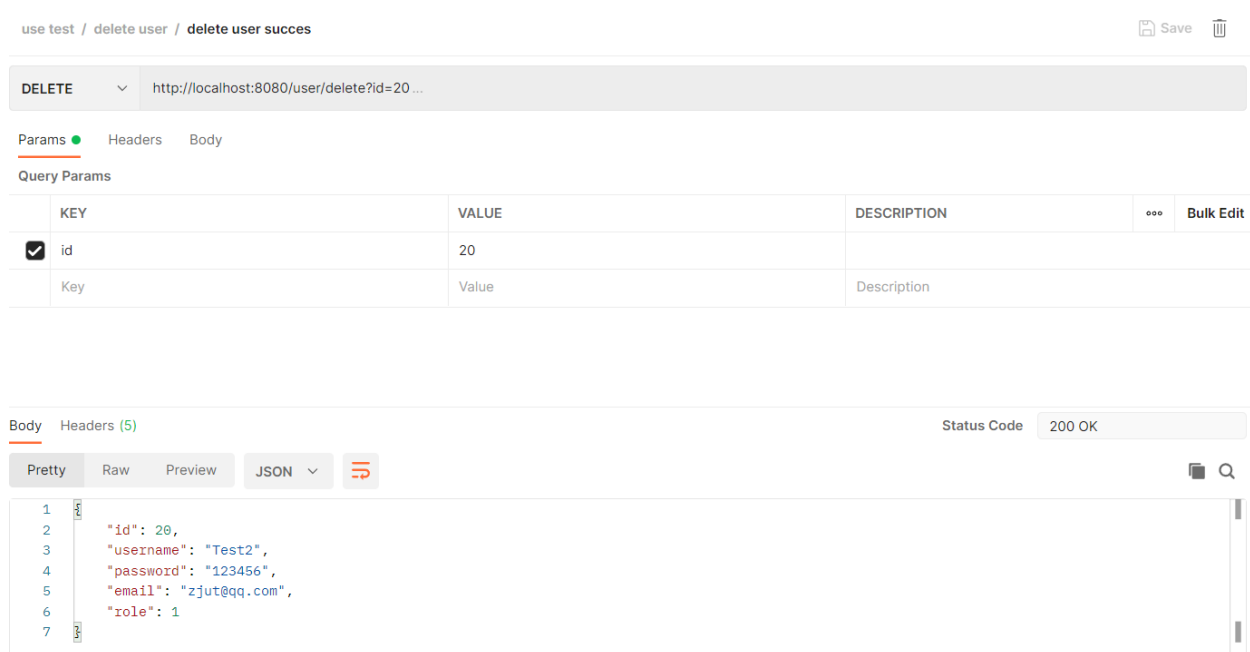


6. Tests d'intégration

Dans la partie du test, nous nous concentrons principalement sur la connexion entre le front et le back end, ainsi que la connexion entre le back end et la base de données.

Dans le processus de construction back-end, nous testerons les services API que nous avons définies une par une via le postman. Postman peut simplifier les étapes pour nous d'appeler l'API et peut facilement écrire des cas de test.

Exemple : Test Supprimer Utilisateur



use test / delete user / delete user succes Save Trash

DELETE ▼ `http://localhost:8080/user/delete?id=20...`

Params ● Headers Body

Query Params

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	id	20			
	Key	Value	Description		

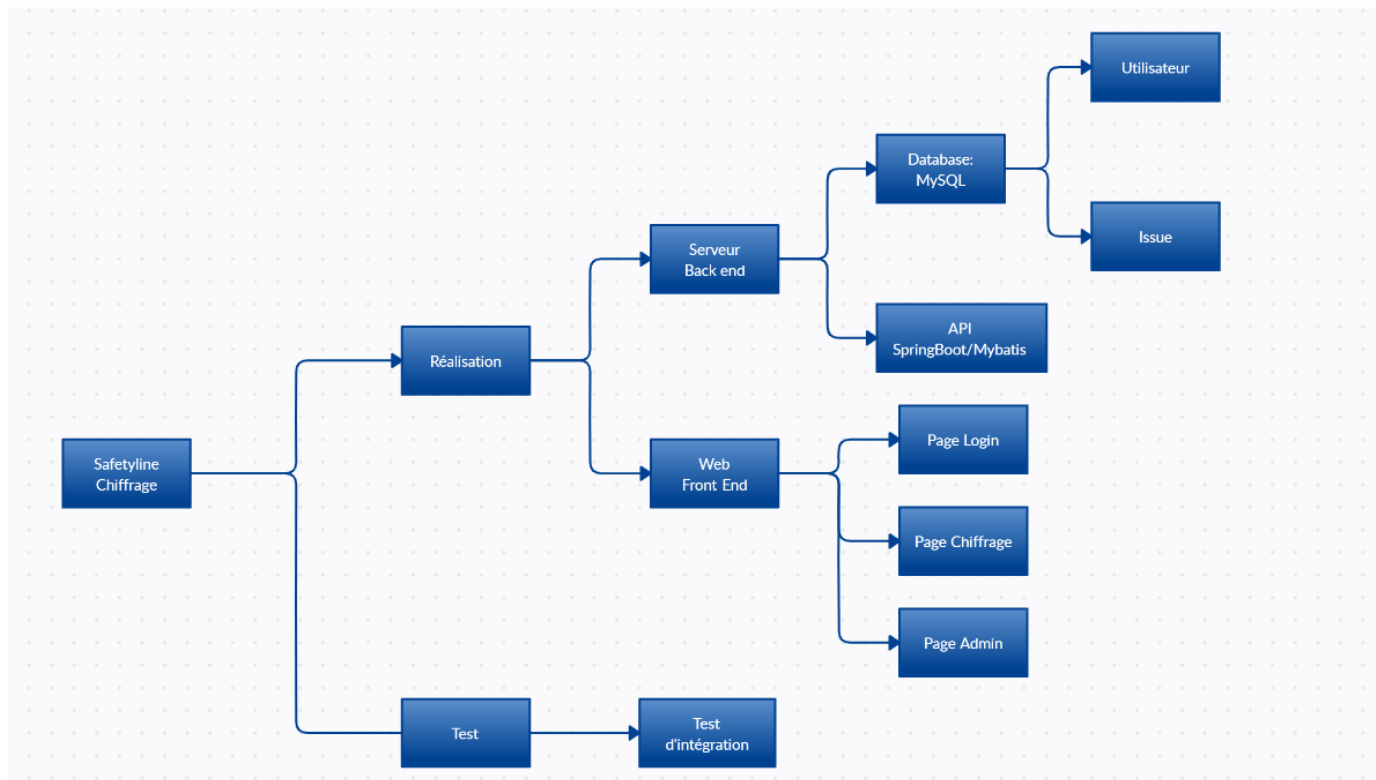
Body Headers (5) Status Code 200 OK

Pretty Raw Preview JSON ▼ ≡

```
1 {
2   "id": 20,
3   "username": "Test2",
4   "password": "123456",
5   "email": "zjut@qq.com",
6   "role": 1
7 }
```

Une fois le front-end construit, nous devons également tester des scénarios en fonction de différents utilisateurs et de différents cas d'utilisation avec notre projet Gitlab.

7. WBS de la réalisation



8. Annexes

GitlabAPI : <https://docs.gitlab.com/ee/api/>

Spring boot: <https://spring.io/projects/spring-boot>

Angular Document : <https://ng.ant.design/docs/introduce/en>

Mybatis plus: <https://baomidou.com/en/>