



Mémoire du Projet de Fin d'Étude

**MISE EN PLACE D'UN SERVICE WEB
ET UNE APPLICATION MOBILE POUR
La gestion des tâches à faire**

Réalisé par :

BENABBI SAÂD

Date de soutenance : 15/09/2020

Jury :

Nom et prénom	Etablissement	Qualité
		Président
Pr. Mohamed beraich	ENS TETOUAN	Examinateur
Pr. Ahmed BEN DAHMANE	ENS TETOUAN	Encadrant pédagogique



Remerciement

Après Dieu, Je tiens à adresser mes remerciements les plus sincères, à mon encadrant monsieur BENDAHMANE AHMED pour son attention, son orientation, son aide pendant la réalisation de ce travail et pour être source d'information et de communication sans hésiter à aucun moment de consacrer une part de son temps précieux .

Je souhaite remercier tout le corps professoral et administratif et l'équipe pédagogique de l'Ecole National Supérieur de Tétouan .

Je désire aussi remercier les professeurs de l'ENS Tétouan, qui m'ont fourni les outils nécessaires au bon déroulement de ce PFE.

Un grand merci à ma mère et mon père, pour leurs conseils ainsi que leur soutien inconditionnel, à la fois moral et économique.

Que tous ceux et celles qui ont contribué de près ou de loin à l'accomplissement de ce travail trouvent l'expression de nos remerciements les plus chaleureux .



Contents

Remerciement	ii
Contents	v
List of Figures	viii
Introduction	1
0.1 context	1
0.2 problematique	2
0.3 Travail à réaliser	2
0.4 Méthodologies du travail	2
0.4.1 Méthode en cascade	3
0.5 Organisation	3
1 Etat de l'art	5
1.1 Android	5
1.1.1 Pourquoi Android ?	5
1.2 Spring boot	6
1.2.1 Pourquoi Spring boot	6
1.2.2 Pourquoi Spring boot?	6
1.2.3 Historique du Spring boot	7
1.3 Spring Framework	9
1.3.1 spring web	10
1.3.2 spring Data JPA	12
1.3.3 Spring security	13

2	Etude de projet	15
2.1	Contexte	15
2.2	Besoins fonctionnels détaillés	15
2.3	Besoins non fonctionnels détaillés	16
2.4	Besoins Technique	17
2.5	Analyse des fonctionnalités	17
2.5.1	Identification des acteurs	17
2.6	Schéma Relationnel	18
2.6.1	Modèle logique donnés MLD	18
2.7	Diagramme de cas d'utilisation	19
2.7.1	Définition	19
3	Branche Technique	26
3.1	Contexte	26
3.2	Architecture	26
3.3	Présentation des services REST	28
3.3.1	Présentation des services REST : URL et URI	28
3.4	Diagramme de déploiement	29
3.4.1	Définition :	29
3.5	Diagramme de séquences	31
3.5.1	Definition	31
3.5.2	Diagramme de séquence «S'Authentification»	32
3.5.3	Diagramme de séquence «Sign»	33
3.5.4	Diagrammes de séquence «Messages»	33
3.5.5	Diagramme de séquence : «taches»	35
3.5.6	Diagramme de séquence : «projects»	36
3.6	Diagramme de class	36
3.6.1	Définition:	36
3.6.2	Diagramme de class coté client	37
3.6.3	Diagramme de class coté serveur	39
4	Phase de Réalisation	44
4.1	Environnement matériel	44
4.2	Environnement logiciel	45
4.3	Choix technique	47
4.4	Choix technique	48

4.5 Format De données Communiquées	48
4.6 Choix de la technologie de sécurité	49
4.7 Présentation de quelque interface	49
4.7.1 Interface Splash Screen	50
4.7.2 Interface Idntifier(Sign in) nouvel utilisateur	52
4.7.3 Authentication	52
4.7.4 Interface Acceuil	53
4.7.5 Interface Setting :	53
4.7.6 Interface Contacts	55
4.7.7 interface messages	56
4.7.8 interface ajoutée taches	56
4.7.9 interface modification tache	56
4.7.10 interface list taches	59
4.7.11 interface suppression des tache	59
4.7.12 interface notification	62
4.7.13 interface projects partagés	62
4.7.14 curl client pour récupérer des donnes from Restful API	63
4.7.15 curl client pour récupérer des donnes from Restful API	63
4.7.16 curl client pour récupérer des donnes from Restful API	63
4.7.17 403 forbidden access	64
4.7.18 les uri	64
Conclusion	66
Webography	68



List of Figures

1	Le modèle en cascade	3
2	Version Spring par rapport au temps	8
3	Spring frameworks	9
4	pom.xml	10
5	Resful Api	11
6	MVC	12
7	spring Data JPA	13
8	Spring Security	14
9	Schéma Relationnel	18
10	diagramme cas utilisation authentication	19
11	diagramme cas utilisation taches	20
12	diagramme cas utilisation Acceuil	21
13	diagramme cas utilisation message	22
14	iagramme cas utilisation projects shared	23
15	diagramme cas utilisation taches	24
16	diagramme cas utilisation notification	25
17	architecture de app	27
18	Service Rest	29
19	Diagramme de déploiement	30
20	broker mqtt	30
21	Diagramme de séquence «S'Authentification»	32
22	Diagramme de séquence «Sign»	33
23	Diagrammes de séquence «Messages»»	34

24	Diagramme de séquence : «taches»	35
25	Diagramme de séquence : «projects»	36
26	diagramme de class message	37
27	Diagramme de class	38
28	Diagramme de class client	38
29	Diagramme de class serveur	39
30	Diagramme de class Contrôleurs	40
31	Diagramme de class services	41
32	Diagramme de class entity	42
33	Diagramme de class repository	43
34	matériel	44
35	Android	45
39	Vim	46
40	Maria DB	46
41	postman	47
42	Mosquitto	47
43	Java	47
44	Spring boot	48
45	XML	48
46	Splash Screen	50
47	interfaces sign in	51
48	Interface d'authentification	52
49	interfaces acceuil	53
50	interfaces Settings	54
51	interfaces Contacts	55
52	Interface messages	56
53	interface ajoutée taches	57
54	interface modification taches	58
55	interface list taches	59
56	interface suppression des tache	60
57	interface notification	61
58	interface projects partagés	62
59	curl	63
60	curl	63
61	curl	64

62	curl	64
63	curl	64
64	les uri	65



Introduction Générale

0.1 context

Nous sommes aujourd’hui dans une situation où les médias sociaux et autres distraction en ligne facilement accessible nous empêchent de souvenir de faire nos tâches ou bien pour les gens qui perte de mémoire ceux qui ont une mémoire à court terme et les personnes âgées atteintes de la maladie d’alzheimer et par l’effet que tout le monde a un téléphone Smartphone j’ai décidé de développer une application vers les smartphones Android pour aider ces gens et faciliter de gérer les tâches; Cette application aussi utilise pour partager les tâches en des projets partagés entre deux utilisateurs ou plus ce qui faciliter d’échanger les informations et faire des tâches en commun aussi j’ai ajouté une fonctionnalité pour chat et échange des messages entre deux utilisateurs d’une manière facile et à distance en internet, et pour faire ça j’ai développé un service web utilisent Spring framework (spring boot, Spring web, Spring security, jpa)

L’objectif de cet Application c’est d’utiliser l’informatique mobile et plus précisément la plateforme de développement Android a fait pour développer une application intégrée dirigée vers les smartphones. Et aussi de créer un service web (Restfull Api) qui faire cette application mobile travaille en internet, aussi avec un système informatique qui stocke les projets des tâches, les messages, les utilisateurs et les contacts de ces utilisateurs. parmi les objectifs de notre application c’est de permis l’accès pour les personnes qui

ont dans le même réseau, utilisant un serveur web existe dans un serveur accessible sur internet travaille comme API (une interface de programmation d'application basée sur REST que l'accessibilité est pour tous les machines qui peuvent interagir avec notre api). L'importance et le but de cette application c'est la mieux adaptée à ses pratiques quotidiennes multiformes. Évitez-la de perdre de temps et faciliter l'accès rapide à l'information à distance sans besoin de transfert.

0.2 problematique

la plupart des gens ont besoin d'une application mobile pour gérer leurs tâches d'une manière simple surtout les gens âgés qui une maladie de perte de mémoire ou d'Alzheimer, l'application vont faire des rappels pour ces gens, et aussi pour les gens de partager leurs tâches avec leurs amis et aussi communiquer avec eux avec des messages..

0.3 Travail à réaliser

Pour remédier aux difficultés présentées, nous proposons de concevoir et mettre en place un service web (Api) utilisant Spring boot avec autres frameworks avec un système informatique mysql et une application Android. Cette solution doit fournir un accès à distance aux tâches, messages et aux contacts à travers une application Android connectée à internet, avec une condition que leurs authentifications soient correctes.

0.4 Méthodologies du travail

Dans cette partie, nous définissons le langage et la méthode de cycle de vie utilisée tout au long du projet afin de modéliser d'une manière claire et précise la structure et le comportement de notre système indépendamment de tout langage de programmation. Le « cycle de vie d'un logiciel» représente toutes les étapes du développement d'un logiciel.

Le découpage permet de détecter les erreurs très tôt et ainsi de maîtriser le coût, le temps, et la qualité du logiciel.

0.4.1 Méthode en cascade

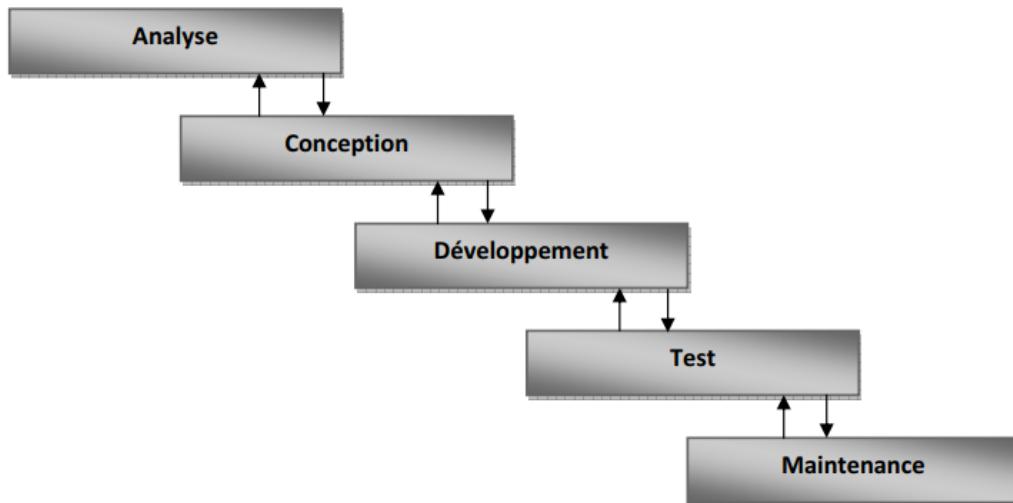


Figure 1. Le modèle en cascade

0.5 Organisation

Organisation

Le présent rapport est composé de quatre chapitres : - dans Le premier chapitre « introduction générale» consiste à mettre notre projet dans son cadre général. nous avons proposé une solution de la problématique de notre cahier des charges, en travailler par une méthodologie qui basait à la méthode en cascade

-Dans le deuxième chapitre état de l'art on a parlé des frameworks qu'on a utilisés, historique et pourquoi. on a parlé d'android, Spring boot avec ces dependencies spring web, Spring data jpa, Spring security.

- dans Le troisième chapitre « branche fonctionnelle» on a présenté les différents besoins fonctionnels et non fonctionnels détaillés, nous avons aussi réalisé le diagramme de classes pour faciliter se comprendre les tâches de chaque utilisateur. Enfin il y a schéma relationnel MCD.

- dans Le quatrième chapitre « branche technique » nous avons présenté Architecte de notre application avec la réalisation des diagrammes de séquences pour détailler la l'interaction entre les utilisateurs avec le system avec explication de chaque diagramme, et aussi le diagramme de classes.
- dans Le cinquième chapitre « réalisation » présente les différents outils de développement utilisés, la phase de test et de validation et quelques interfaces de l'application. on a Présenté les différentes techniques comme les logiciels utilisés et les choix techniques (les langages de programmation), enfin nous avons présenté quelques interfaces de notre application.

Chapter

1

Etat de l'art

1.1 Android

Android, prononcé Android, est un système d'exploitation mobile pour smartphones, tablettes tactiles, PDA, smart watches et terminaux mobiles. C'est un système open source utilisant le noyau Linux. Il a été lancé par une startup du même nom rachetée par Google en 2005.

D'autres types d'appareils possèdent ce système d'exploitation existent, par exemple des télévisions, des radioréveils, des montres connectées, des autoradios et même des voitures.

1.1.1 Pourquoi Android ?

Android est Open Source. Chaque constructeur se permet donc de le bidouiller pour attirer plus de monde. Ceux qui en tirent profit sont au final les clients ! Les options varient d'un constructeur à un autre par exemple Samsung, LG ou HTC. Android est aussi convivial, personnalisable, flexible et simple

Alors j'ai choisi Android car c'est le téléphone que j'ai utilisé, et à propos de pourquoi je ne choisis pas cross platform mobile framework qui travaille native comme native script c'est que j'ai besoin d'accéder au service alarm et j'ai utilisé pour ça la class AlarmManager et cette classe permet d'accéder au service d'alarme du système. Ceux-ci vous permettent de planifier l'exécution de votre application à un moment donné dans le futur. Lorsqu'une

alarme se déclenche, l'intention qui avait été enregistrée pour elle est diffusée par le système, démarrant automatiquement l'application cible si elle n'est pas déjà en cours d'exécution. Les alarmes enregistrées sont conservées pendant que l'appareil est en veille (et peuvent éventuellement réveiller l'appareil si elles s'éteignent pendant cette période), mais seront effacées s'il est éteint et redémarré.

autre chose c'est que j'ai besoin de deux bases de données, base de données locale et autres dans un serveur. et j'ai utilisé Sqlite pour stocker les données locales.

1.2 Spring boot

1.2.1 Pourquoi Spring boot

a) Spring Boot est un sous-projet de Spring qui vise à rendre Spring plus facile d'utilisation en éliminant plusieurs étapes de configuration.

1.2.1.1 L'objectif de Spring Boot est de permettre aux développeurs de se concentrer sur des tâches techniques et non des tâches de configurations, de déploiements.

1.2.1.2 Ce qui a pour conséquence un gain de temps et de productivité (avec Spring Boot, il est très facile de démarrer un projet n-tiers).

1.2.2 Pourquoi Spring boot?

C'est évident que Spring boot est la meilleure solution pour un projet RESTful API (notre cas) car il divise les tâches et facilite la gestion ainsi les notions de sécurité implémenter dedans mais en peut déduire en général qu'il est .

- Légèreté : Spring Boot a la particularité d'être très léger et d'embarquer avec lui le strict minimum pour faire tourner votre service.

- Intégration facilitée : Spring Boot s'intègre particulièrement bien dans une architecture orientée micro-services... et c'est l'un des seuls ! En effet, l'adoption des architectures micro-services au sein des organisations étant relativement récente, il n'existe pas dans l'univers de JAVA de framework capable de créer des services suffisamment légers et performants.
- Simplicité de prise en main : Spring Boot permet donc de créer une API de services très simplement. Il suffit d'embarquer directement le serveur d'applications dans un seul et unique Jar qui est exécutable, par exemple, directement dans un service de conteneur (exemple : Amazon Web Service).

Quelques avantages de Spring boot

- Il facilite notamment la création, la configuration et le déploiement d'une application complète. On n'a plus besoin des fichiers XML à configurer (pas besoin du fichier du descripteur de déploiement web.xml dans le cas d'une application web).
 - Spring Boot permet de déployer très facilement une application dans plusieurs environnements sans avoir à écrire des scripts. Pour ce faire, une simple indication de l'environnement (développement ou production) dans le fichier de propriétés (.properties) suffit à déployer l'application dans l'un ou l'autre environnement.
- avantages de Spring boot
- Spring Boot possède un serveur d'application Tomcat embarqué afin de faciliter le déploiement d'une application web. Il est possible d'utiliser un serveur autre ou externe, grâce à une simple déclaration dans le fichier pom.xml.
 - Spring Boot permet de mettre en place un suivi métrique de l'application une fois déployée sur le serveur afin de suivre en temps réel l'activité du serveur, ceci grâce à spring-boot-starter-actuator.

1.2.3 Historique du Spring boot

La première version a été écrite par Rod Johnson, qui a publié le cadre avec la publication de son livre < Expert One-on-One J2EE Design and Development > en octobre 2002.

Le cadre a été publié pour la première fois sous la licence Apache 2.0 en juin 2003. Le premier jalon La version 1.0 a été publiée en mars 2004 avec d'autres versions importantes en septembre 2004 et mars 2005. Le Framework Spring 1.2.6 a remporté un prix de productivité Jolt et un prix d'innovation JAX (API Java pour XML) en 2006. Le printemps 2.0 a été lancé en octobre 2006, le printemps 2.5 en novembre 2007, le printemps 3.0 en décembre 2009, le printemps 3.1 en décembre 2011 et le printemps 3.2.5 en novembre 2013. Spring Framework 4.0 a été publié en décembre 2013. Les améliorations notables de Spring 4.0 incluent la prise en charge de Java SE (Standard Edition) 8, Groovy 2, certains aspects de Java EE 7 et Web Socket.

Spring Framework 4.2.0 a été publié le 31 juillet 2015 et a été immédiatement mis à niveau vers la version 4.2.1, qui a été publiée le 01 septembre 2015. Il est "compatible avec Java 6, 7 et 8, en mettant l'accent sur les améliorations de base. et des capacités Web modernes".

Spring Framework 4.3 a été publié le 10 juin 2016 et sera pris en charge jusqu'en 2020. Il "sera la génération finale dans le cadre des exigences système générales de Spring 4 (Java 6+, Servlet 2.5+), [...]".

Spring 5 est annoncé comme étant basé sur Reactive Streams compatible Reactor Core. Et dans l'années 2013 la naissance de spring boot Framework.

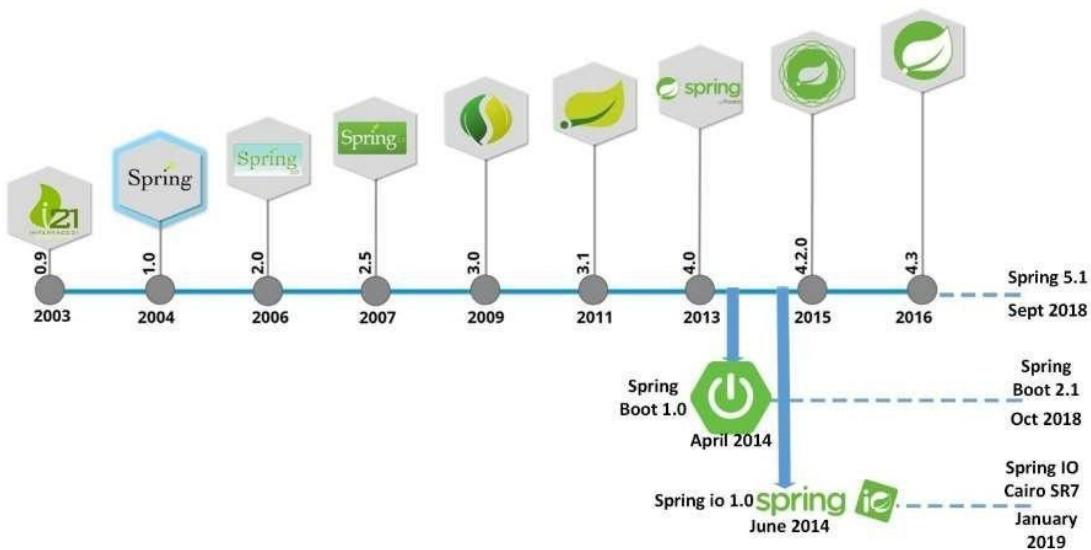


Figure 2. Version Spring par rapport au temps

1.3 Spring Framework

est un Framework de développement d'applications Java, qui apporte plusieurs fonctionnalités comme SpringSecurity, SpringMVC, SpringBatch, SpringLoc, SpringData, etc. Ces frameworks ont pour objectif de faciliter la tâche aux développeurs. les mises en œuvre sont devenues complexes à travers les fichiers de configuration XML qui ne cessent de grossir, et une gestion des dépendances fastidieuse. C'est pour répondre à cette inquiétude que le projet SpringBoot a vu le jour.

Ces frameworks ont pour objectif de faciliter la tâche aux développeurs.



Figure 3. Spring frameworks

```
<groupId>com.zer0time</groupId>
<artifactId>pfebackend</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>pfebackend</name>
<description>un projet de fin d'étude ens tetouan</description>

<properties>
    <java.version>14</java.version>
</properties>

<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
    <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-security</artifactId>
    </dependency>
    <dependency>
        <groupId>io.jsonwebtoken</groupId>
        <artifactId>jjwt</artifactId>
        <version>0.9.1</version>
    </dependency>
    <dependency>
        <groupId>com.fasterxml.jackson.dataformat</groupId>
        <artifactId>jackson-dataformat-xml</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
        <exclusions>
            <exclusion>
                <groupId>org.junit.vintage</groupId>
                <artifactId>junit-vintage-engine</artifactId>
            </exclusion>
        </exclusions>
    </dependency>
</dependencies>
```

Figure 4. pom.xml

1.3.1 spring web

Créez des applications Web, y compris RESTful, à l'aide de Spring MVC. Utilise Apache Tomcat comme conteneur intégré par défaut.

1.3.1.1 c'est quoi restful?

REST (representational state transfert) est un style d'architecture logicielle définissant un ensemble de contraintes à utiliser pour créer des services web. Les services web conformes au style d'architecture REST, aussi appelés services web RESTful, établissent une interopérabilité entre les ordinateurs sur Internet. Les services web REST permettent aux systèmes effectuant des requêtes de manipuler des ressources web via leurs représentations textuelles à travers un ensemble d'opérations uniformes et prédéfinies sans état.

Figure 5. Restful api

1.3.1.2 c'est quoi mvc

Modèle-vue-contrôleur ou MVC est un motif d'architecture logicielle destiné aux interfaces graphiques lancé en 1978 et très populaire pour les applications web. Le motif est composé de trois types de modules ayant trois responsabilités différentes : les modèles, les vues et les contrôleurs.

Un modèle (Model) contient les données à afficher.

Une vue (View) contient la présentation de l'interface graphique. dans notre cas c'est xml ou bien json format

Un contrôleur (Controller) contient la logique concernant les actions effectuées par l'utilisateur.

Model-View-Controller

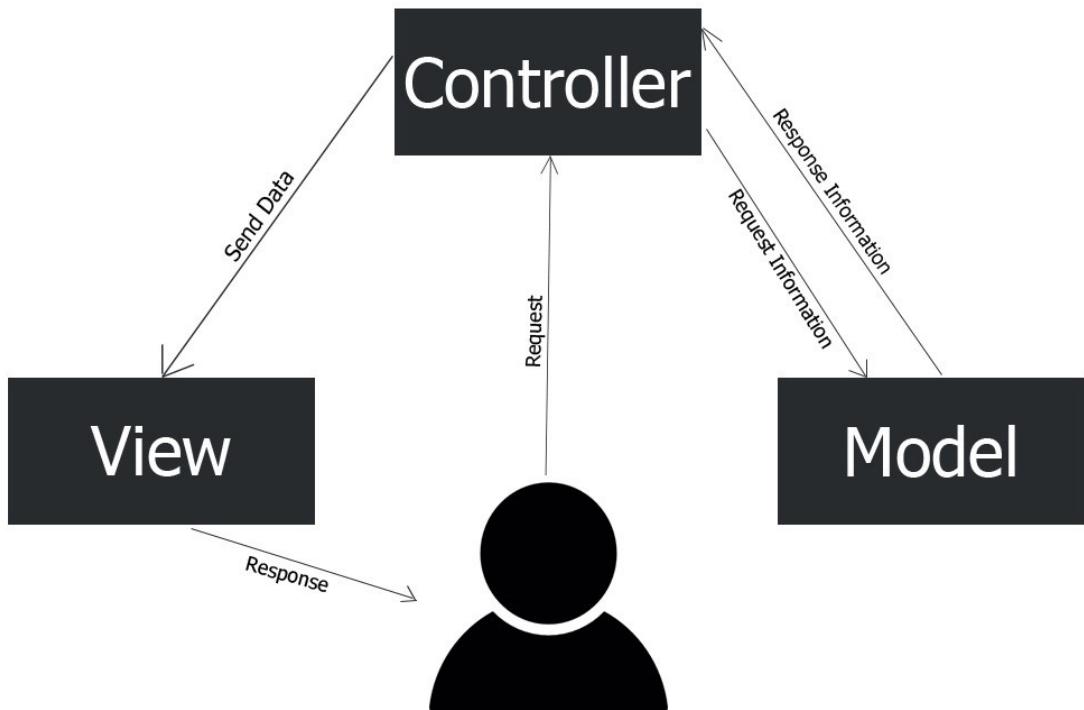


Figure 6. mvc

1.3.2 spring Data JPA

Persistance des données dans les store SQL avec l'API Java Persistence à l'aide de Spring Data et Hibernate

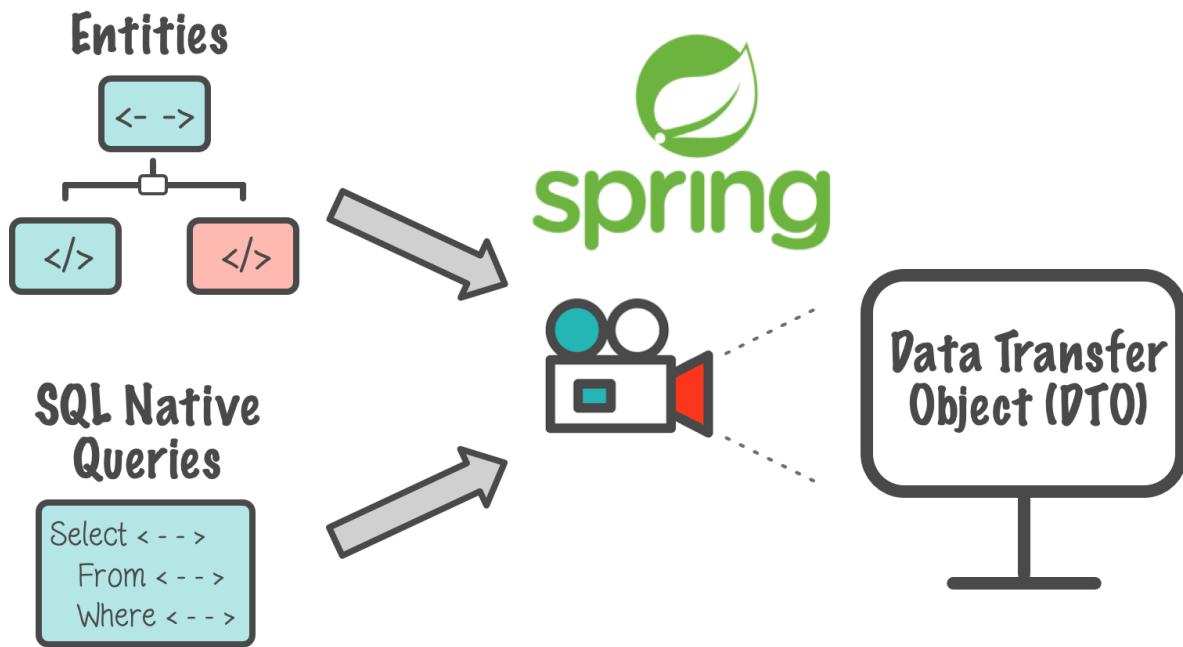


Figure 7. spring Data JPA

1.3.3 Spring security

Framework d'authentification et de contrôle d'accès hautement personnalisable pour les applications Spring. j'ai utiliser pour l'authorization des requests .

```
@EnableWebSecurity
public class WebSecurity extends WebSecurityConfigurerAdapter{
    private final UserService userDetailsService;
    private final BCryptPasswordEncoder bCryptPasswordEncoder;

    public WebSecurity(UserService userDetailsService, BCryptPasswordEncoder bCryptPasswordEncoder) {
        this.userDetailsService = userDetailsService;
        this.bCryptPasswordEncoder = bCryptPasswordEncoder;
    }

    @Override
    protected void configure(HttpSecurity http) throws Exception {
        http.csrf().disable().authorizeRequests().antMatchers(HttpMethod.POST, SecurityConstants.SIGN_UP_URL, "/users/sign")
            .permitAll().antMatchers(HttpMethod.GET, "/users/downloadFile/**").permitAll()
            .antMatchers(HttpMethod.POST, "/taches/").permitAll()
            .anyRequest().authenticated().and()
            .addFilter(getAuthenticationFilter())//authentication filter
            .addFilter(new AuthorizationFilter(authenticationManager()))//authorization filter
            .sessionManagement()
            .sessionCreationPolicy(SessionCreationPolicy.STATELESS);
    }

    @Override
    protected void configure(AuthenticationManagerBuilder auth) throws Exception {
        auth.userDetailsService(userDetailsService).passwordEncoder(bCryptPasswordEncoder);
    }

    public AuthenticationFilter getAuthenticationFilter () throws Exception {
        final AuthenticationFilter authenticationFilter = new AuthenticationFilter(authenticationManager());
        authenticationFilter.setFilterProcessesUrl("/users/login");
        return authenticationFilter;
    }
}
```

Figure 8. Spring security

on a implémenté Spring security dans notre projet on a configuré la class WebSecurity, en implémentant l'authentification et l'envoi de token à chaque correcte authentification, et aussi on a implémenté autorisation et l'accès des requêtes.

Chapter

2

Etude de projet

2.1 Contexte

Les besoins fonctionnels Il s'agit des fonctionnalités du système. Ce sont les besoins spécifiant un comportement d'entrée / sortie du système. Et les besoins non fonctionnels Il s'agit des besoins qui caractérisent le système. Ce sont des besoins en matière de performance, de type de matériel ou le type de conception. Ces besoins peuvent concerter les contraintes d'implémentation (langage de programmation, type SGBD, de système d'Exploitation...).

2.2 Besoins fonctionnels détaillés

Les besoins fonctionnels présentent les actions que le système doit assurer en répondant à une requête. Ce système se place en tant qu'interface entre les utilisateurs pour gérer leurs taches soit localement ou par internet avec la possibilité de partager ces taches avec des amis et aussi le possible se communiquer entre eux. L'application offrira un certain nombre de services via un espace personnel accessible depuis le réseau local d. L'application offre un ensemble de services telles que :

- Gestion des Contacts :Ce service permet gérant les contacts d'un utilisateur.

- Gestion des tâches : ajouter des tâches /supprimer /éditer / ajouter des tâches à des projets partagés
- Gestion des projects: ajouter des nouveaux projets partagés, et ajoute des utilisateurs à ces projets.
- Gestion des utilisateurs : Ce service permet d'enregistrer (sign in) un nouvel utilisateur pour peut authentifier avec notre application.
- Gestion des Messages : Ce service permet d'envoyer et recevoir des messages.
- Gestion des notifications : Ce service permet de push notification . Pour alerter quand un nouveau message.
- Gestion du profil : Ce service permet la consultation et la modification des informations personnelles.

2.3 Besoins non fonctionnels détaillés

La simplicité et la lisibilité représentent les principaux besoins non fonctionnels que doivent fournir notre application mobile ainsi que d'autres contraintes :

- Contraintes ergonomiques : simplicité et convivialité des interfaces graphiques.
- Contraintes de sécurité : authentification,(les mots de pass sont hachés avec bcrypt, ressaisi du mot de passe lors d'un traitement dans la base de données.) . utilisant Spring security aussi l'utilisation d'autorisation avant d'accepter une requête http, en utilisant le framework spring security aussi on a utilisé un codage en modèle MVC qui sécurise l'application
- Accès internet : Une application parfaite doit permet d'un minimum d'accès off ligne.
- Contraintes de performance : accès facile, chargement rapide.
- Contrainte de fiabilité : sans ambiguïté.
- L'application devra être capable de Tourner en réseau.

2.4 Besoins Technique

- Avoir un Smartphone tournant sous le système d'exploitation Android de Google.
- Une connexion au réseau local ou internet si le serveur déployé internet.

2.5 Analyse des fonctionnalités

2.5.1 Identification des acteurs

on a un seul acteur qui interact avec le téléphone et le service web utilisateur: C'est une personne intéressée par l'application. Et qui peut gérer (ajouter, supprimer) leurs tâches, des projets partagés avec ses tâches, ses contacts envoyés et recevoir les messages.

2.6 Schéma Relationnel

2.6.1 Modèle logique données MLD

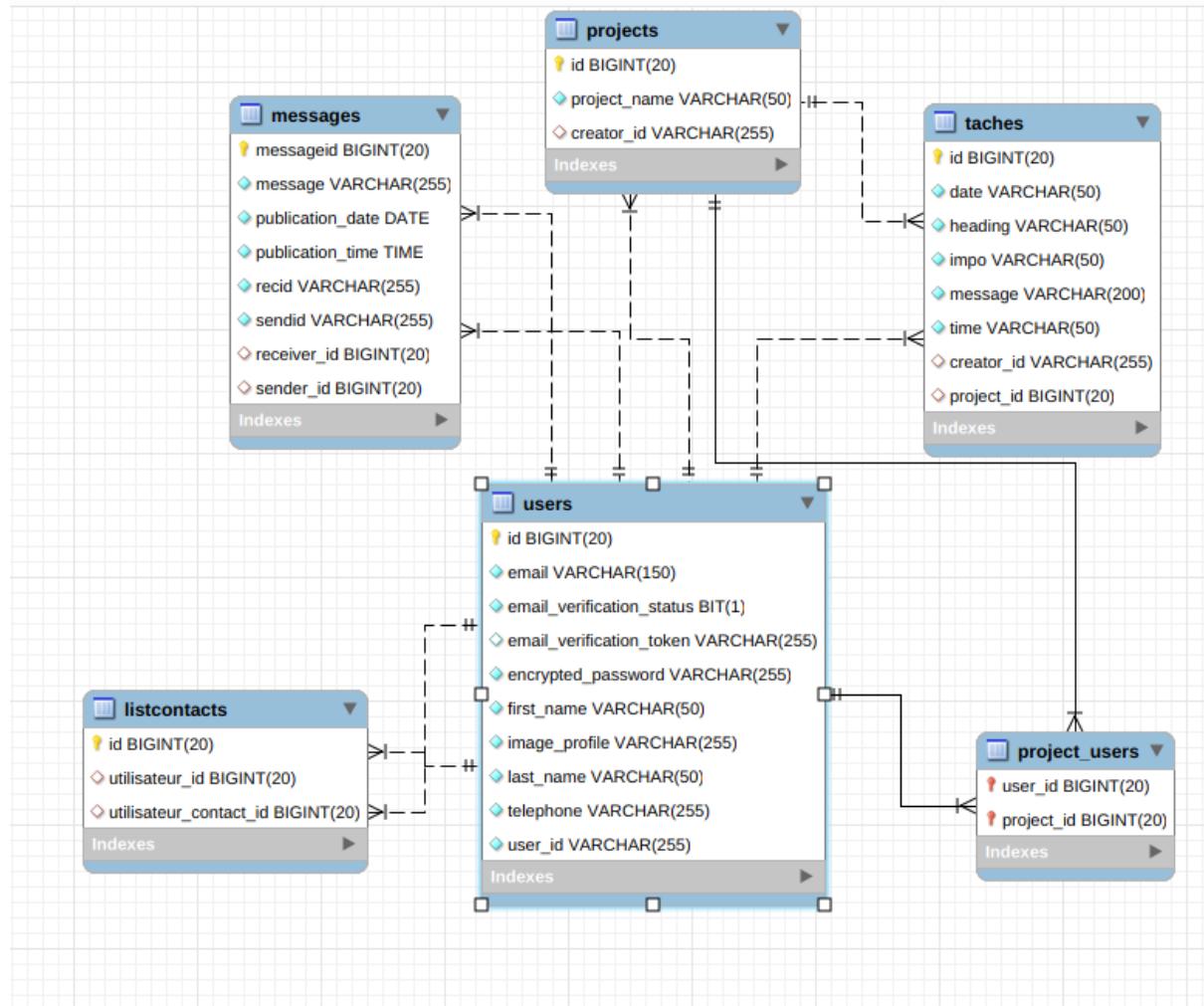


Figure 9. Schéma Relationnel

- users et projets ont une relation Many to Many.
- users et messages ont une relation One to Many
- taches et projets ont une relation Many to one
- projects et users ont une relation many to many join on table projectusers
- users and users ont une relation many to many join on table listcontacts

2.7 Diagramme de cas d'utilisation

2.7.1 Définition

Diagramme de cas d'utilisation permet de créer l'interaction entre les acteurs (l'utilisateur de cas) et le système. La représentation de cas d'utilisation est basé sur trois concepts : Acteur et cas d'utilisation et l'interaction entre Acteur et cas d'utilisation.

2.7.1.1 diagramme cas utilisation authentication

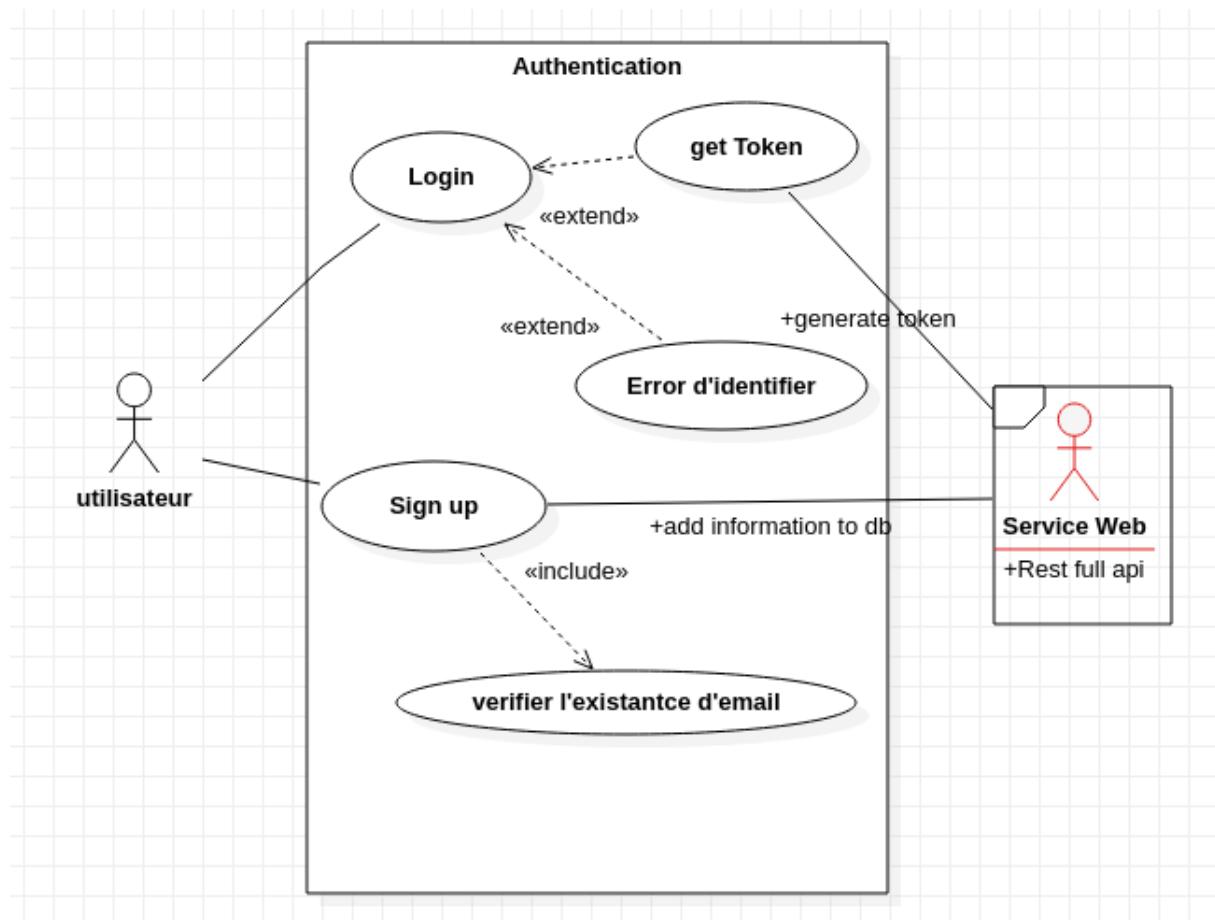


Figure 10. diagramme cas utilisation authentication

2.7.1.2 diagramme cas utilisation taches

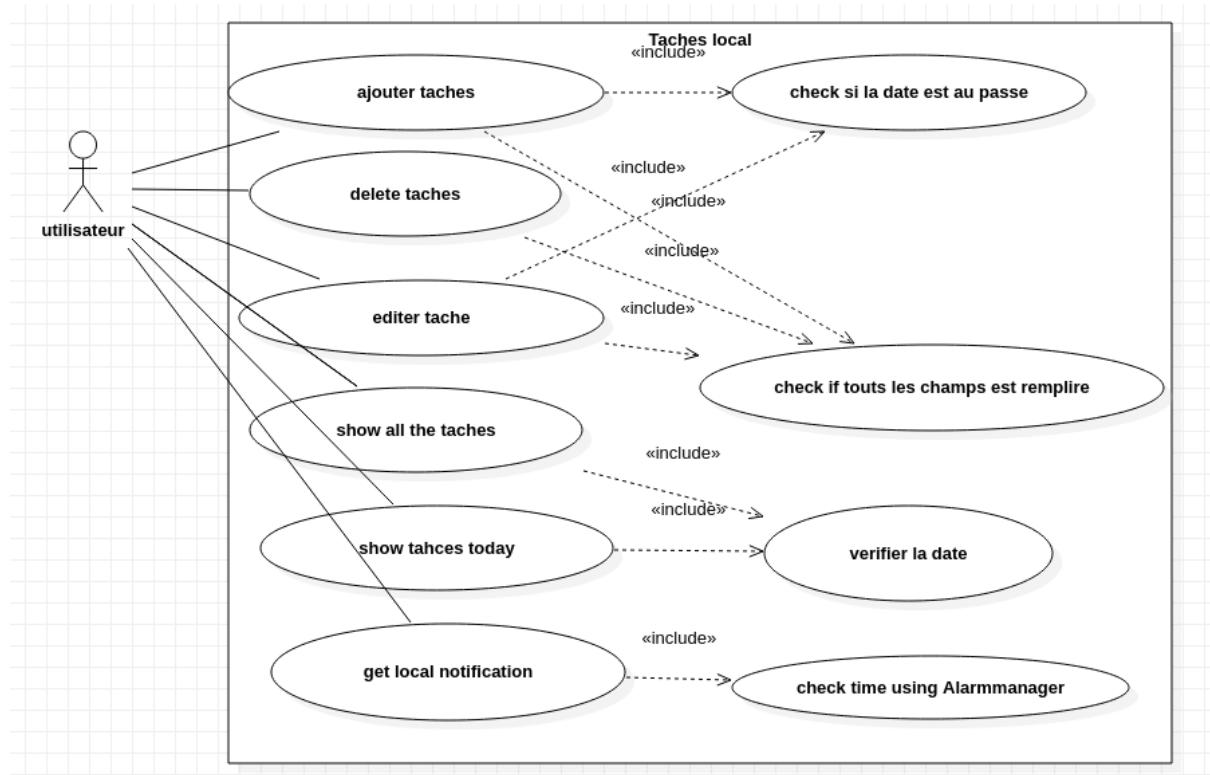


Figure 11. diagramme cas utilisation taches

2.7.1.3 diagramme cas utilisation Acceuil

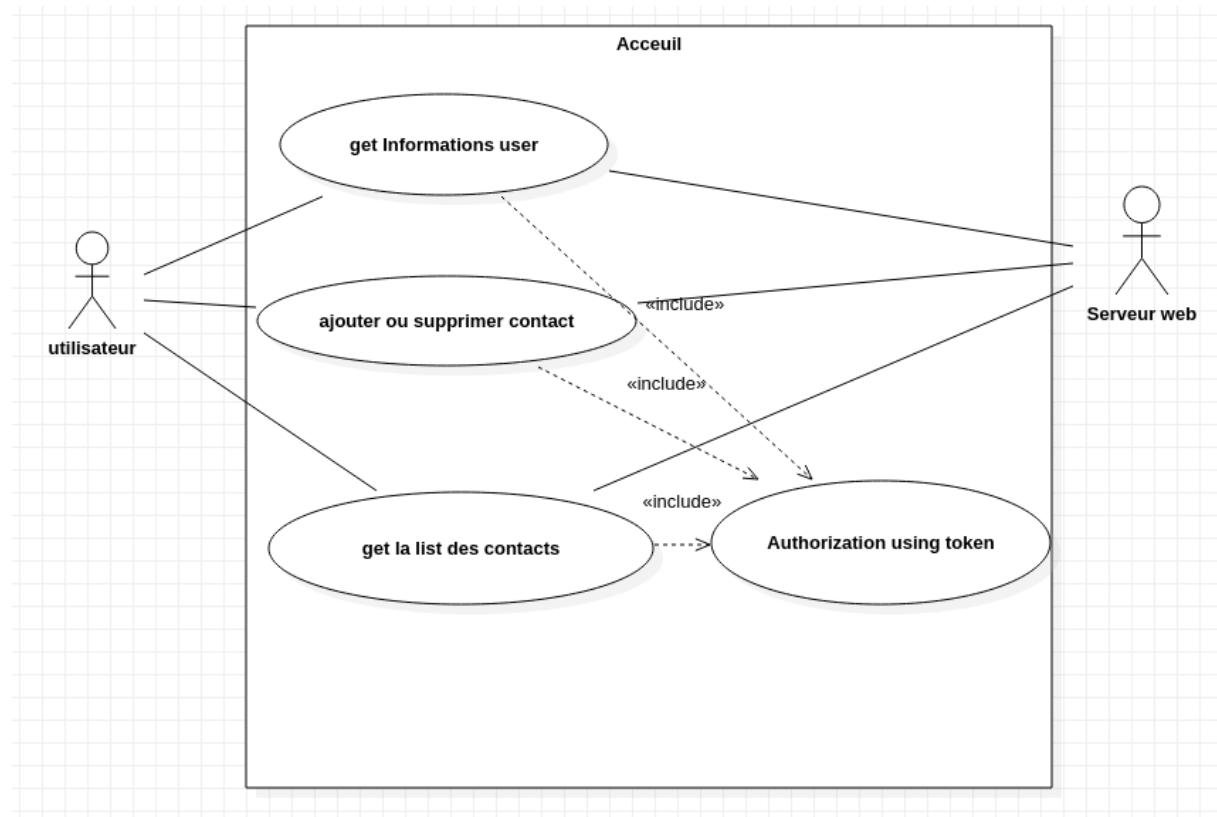


Figure 12. diagramme cas utilisation Acceuil

2.7.1.4 diagramme cas utilisation messages

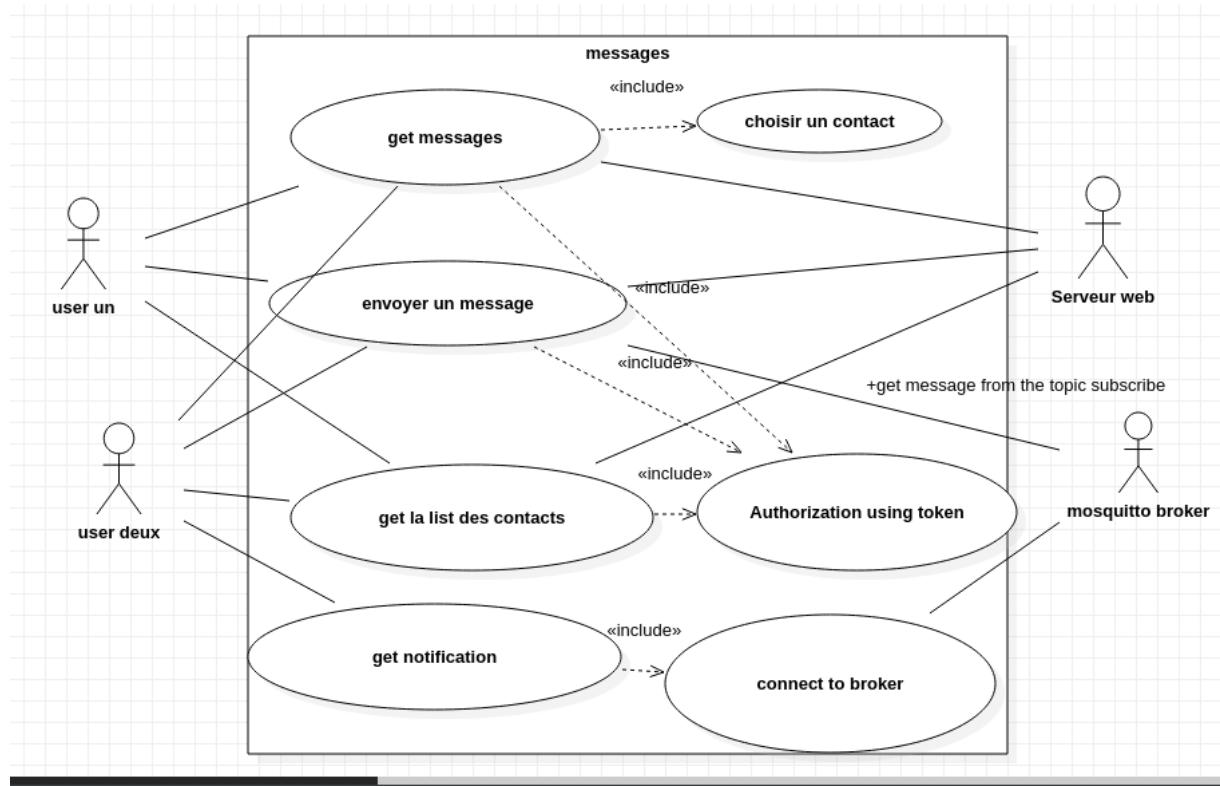


Figure 13. diagramme cas utilisation messages

2.7.1.5 diagramme cas utilisation projects shared

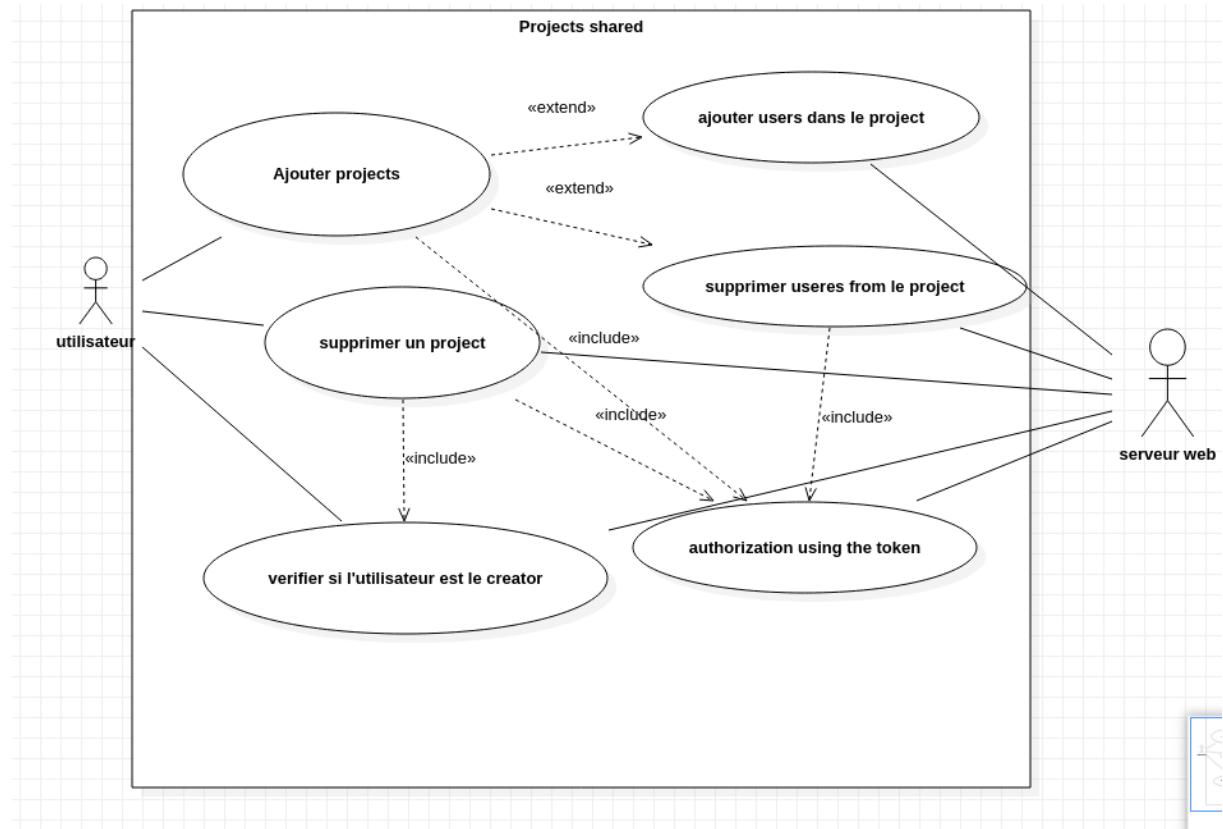


Figure 14. iagramme cas utilisation projects shared

2.7.1.6 diagramme cas utilisation tâches

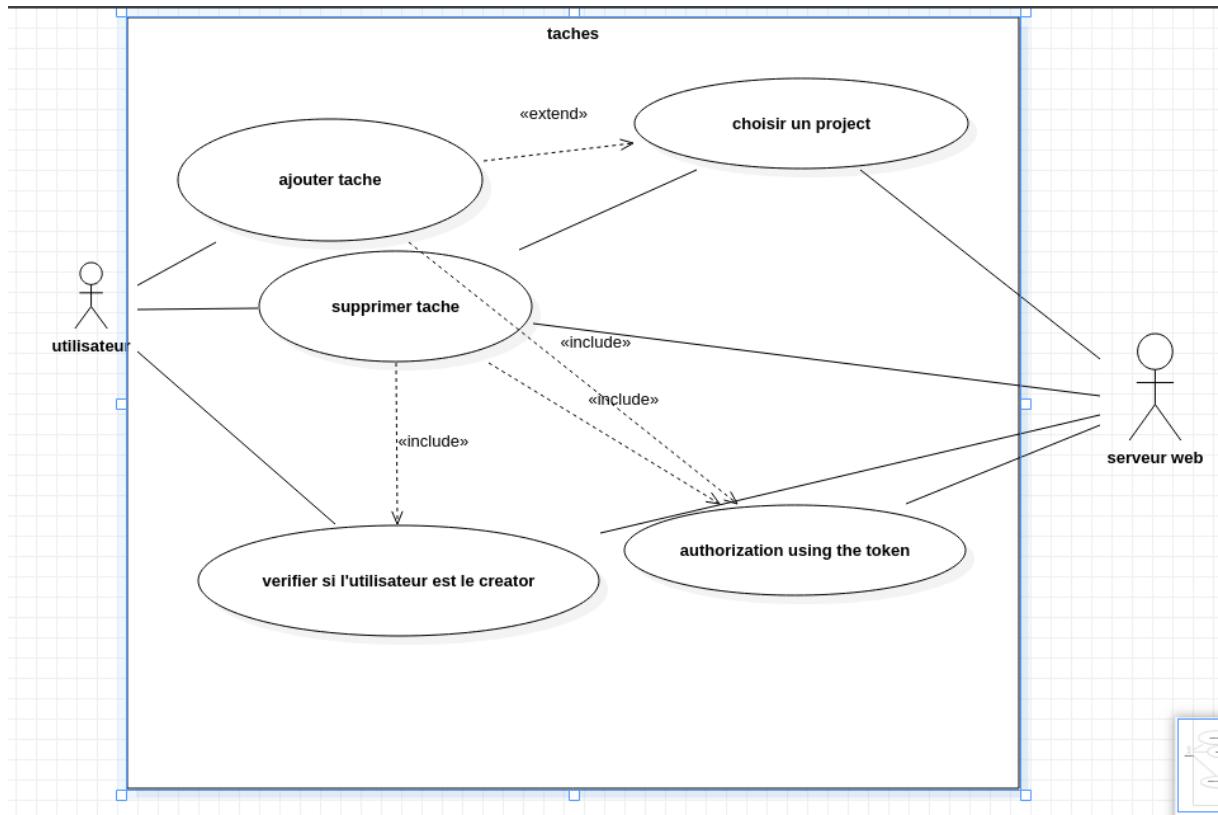


Figure 15. diagramme cas utilisation tâches

2.7.1.7 diagramme cas utilisation notification

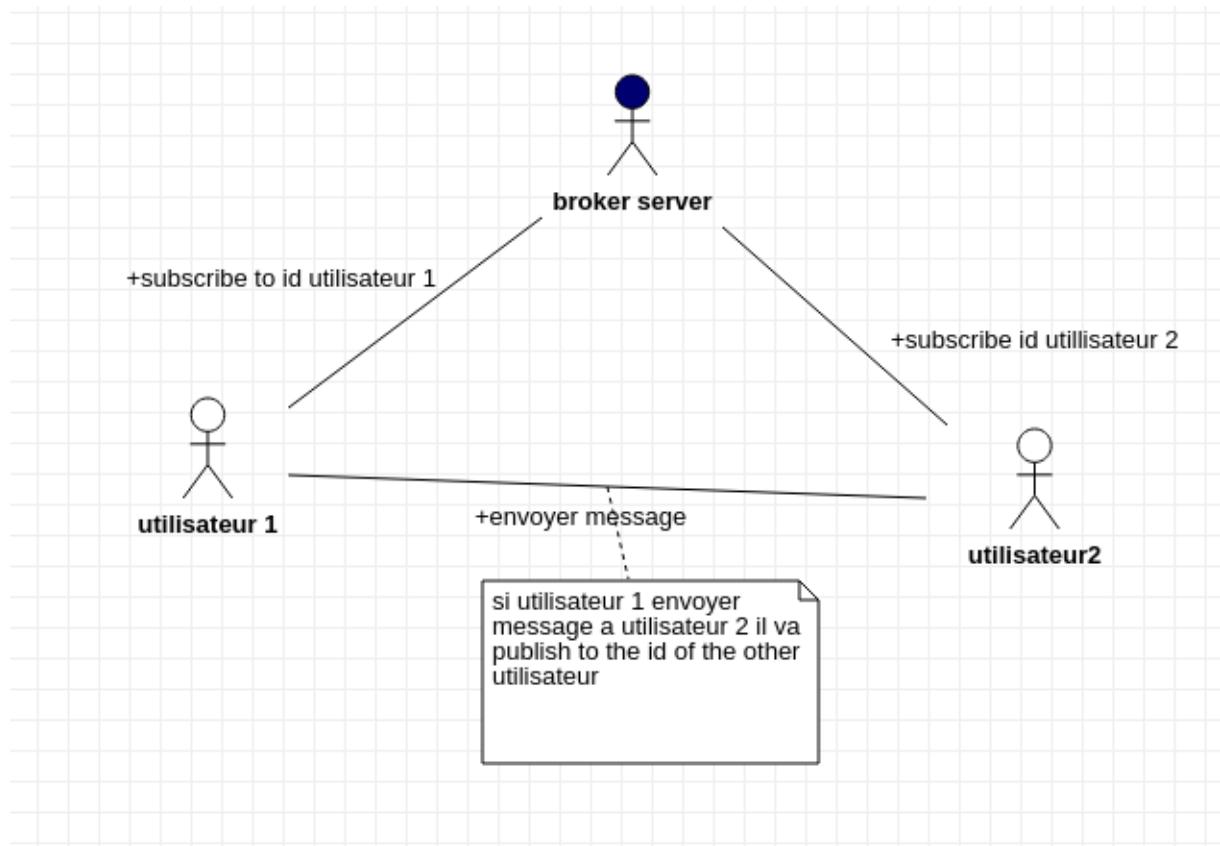


Figure 16. diagramme cas utilisation notification

si un utilisateur veut envoyer un message il a besoin de connecter à un broker qui est mosquito dans notre cas et aussi de faire subscribe au topic qui est l'id public personnel , quand l'autre utilisateur envoie un message, il faire un publish au broker dans le topic de l'id public de person qu'il veut envoyer le message.

Chapter

3

Branche Technique

3.1 Contexte

La branche technique capitalise un savoir-faire technique et/ou des contraintes techniques. Les techniques développées pour le système le sont indépendamment des fonctions à réaliser.

3.2 Architecture

Opérer une connexion directe à la base de données depuis le téléphone Android n'est pas conseillé d'un point de vue architecture logicielle. Il est en effet préférable de passer par un middleware. Cette couche serveur intermédiaire sera la seule habilitée à se connecter à la base de données, ce qui est plus sécurisé. Donc La méthode la plus répandue pour se connecter à une base de données MySQL à distance à partir d'un appareil Android, est de mettre une sorte de service dans le milieu. donc le plus évident est d'écrire un web service (Restful api pour gérer la base de données et gérer les requêtes à travers des contrôleurs en utilisant le protocole HTTP. J'ai codé les données dans le format JSON, afin de communiquer les données entre web service et Android, en exploitant les options faciles à utiliser construit dans les fonctions JSON dans les deux langages.

En fait, si je parle de l'architecture 3-tiers de point de vue technologie, le client est la

plateforme Android, le serveur web et le Spring boot et le serveur de bases de données et le Mysql. Lorsque notre application Android s'exécute, elle va être connectée au serveur Web qui va récupérer les données depuis la base de données MySQL en utilisant les services web.

Ensuite les données seront encodées au format JSON et envoyées au système Android. L'application va obtenir ces données codées. Elle les analysera et les affichera sur le mobile.

En fait notre projet est divisé en deux parties à développer le serveur web (avec spring) et le développement de l'application mobile Android. La partie serveur est composée de deux serveurs distants : le serveur web et le serveur de bases données. Le serveur Web utilisé est le serveur Tomcat, il est le serveur le plus répandu sur internet pour déployer les scripts JAVA. Le serveur de bases de données utilisé est le serveur MySql

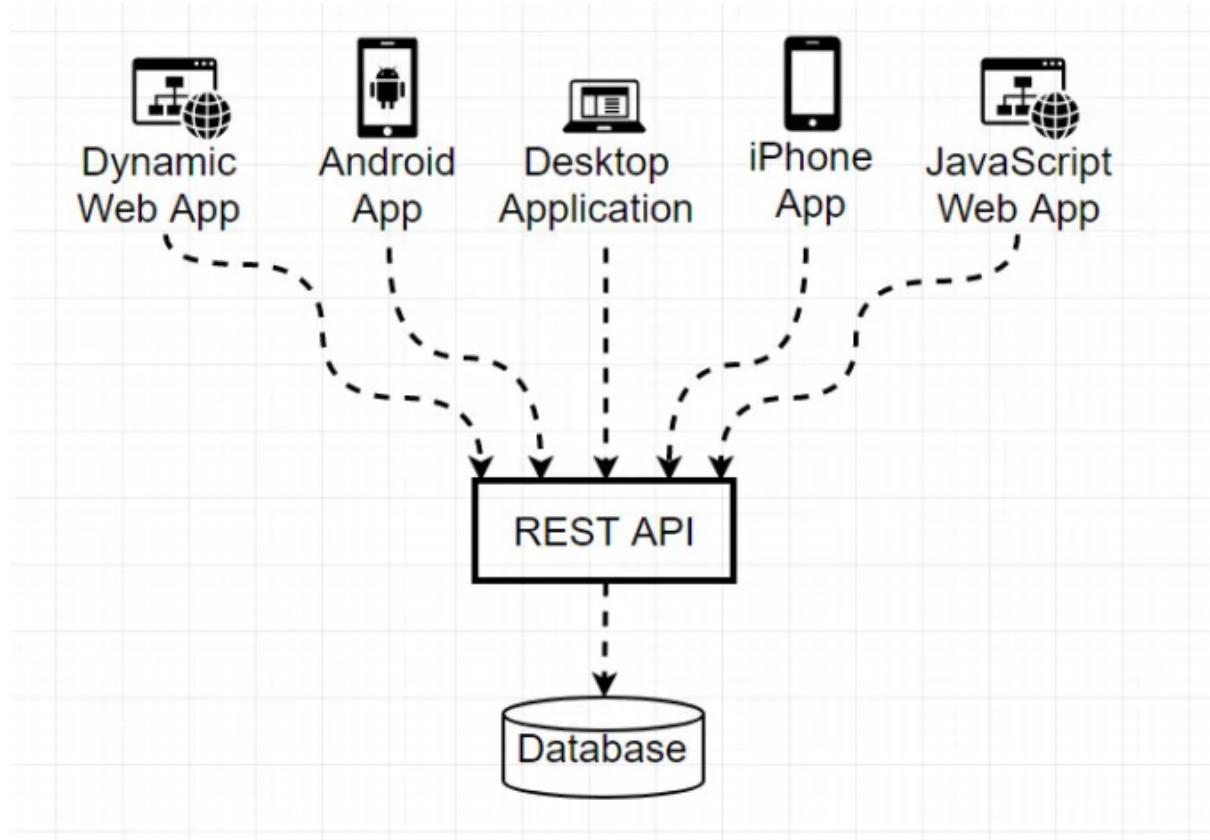


Figure 17. architecture de app

3.3 Présentation des services REST

- Les services REST (Representational State Transfer) représentent un style d'architecture pour développer des services web.
- Une API qui respecte les principes REST est appelée API-RESTful.
- Les principes clés de REST impliquent la séparation de l'API en ressources logiques. Ce qui revient à penser à comment obtenir chaque ressource.
- Une ressource est un objet ou une représentation d'objets contenant éventuellement des données. Exemple : un employé d'une société est une ressource. La manipulation de ces ressources repose sur le protocole HTTP à travers les méthodes d'actions GET, POST, PUT, PATCH, DELETE.

3.3.1 Présentation des services REST : URL et URI

- Pour obtenir une ressource, il faut déjà l'identifier à travers une URL et bien la nommer.
- Une URL (Uniform Ressource Locator) indique le chemin vers une ressource. Cette ressource n'est pas toujours disponible. Lorsque l'URL pointe vers une ressource disponible, on parle d'une URI
- Une URI (Uniform Ressource Identifier) est l'identifiant unique de ressource sur un réseau (URI = URL + Ressource). Une URI est donc une URL qui pointe vers une ressource bien identifiée.

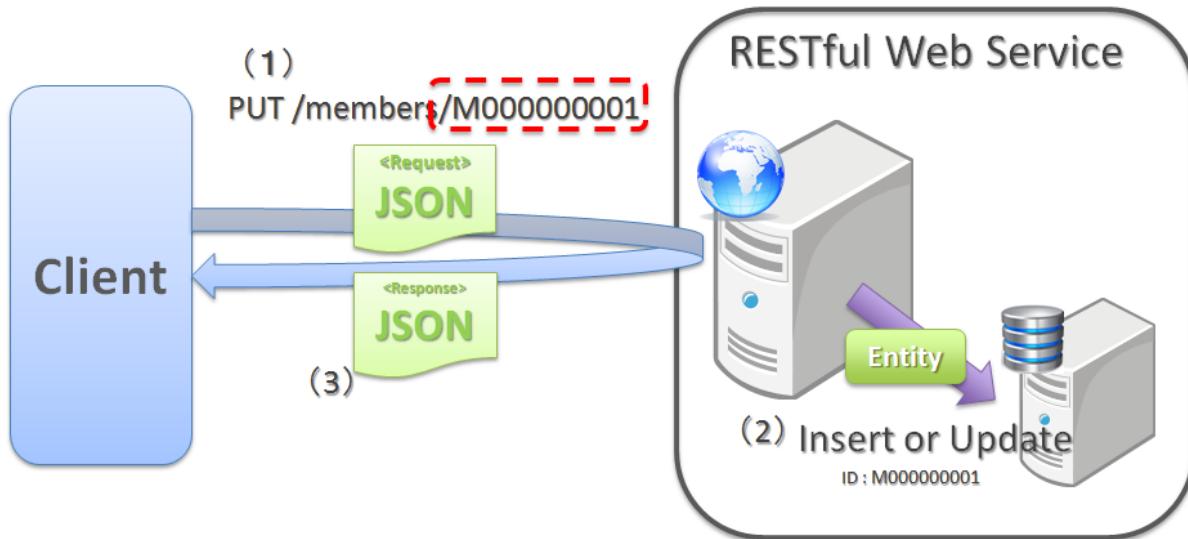


Figure 18. Service Rest

3.4 Diagramme de déploiement

3.4.1 Définition :

Le diagramme de déploiement montre la configuration physique des différents matériels qui participent à l'exécution du système, ainsi que les artefacts qu'ils supportent. Ce diagramme est constitué de « noeuds» connectés par des liens physiques. Les symboles des noeuds peuvent contenir des artefacts (et non plus des composants comme en UML 1)

RESTful Web Service in Java

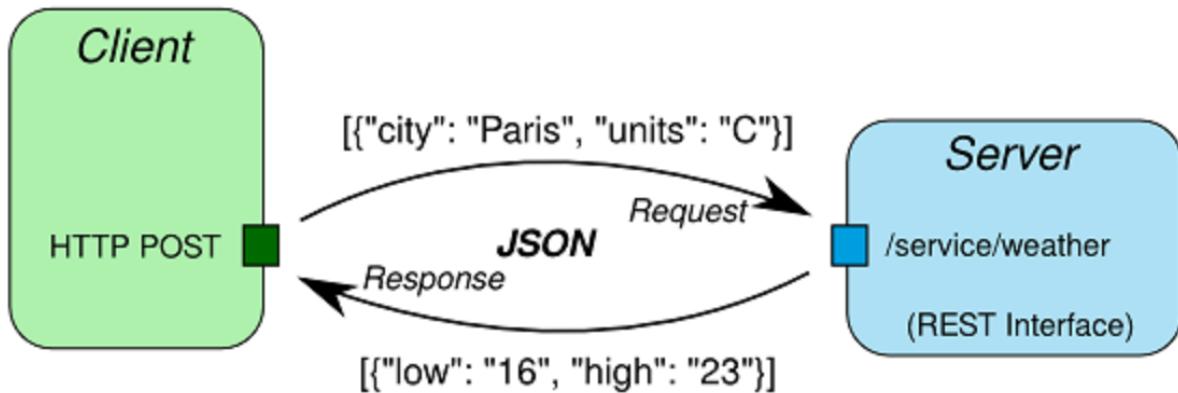


Figure 19. Diagramme de déploiement

Notre Application Android fonctionne avec un API, l'application fait des requêtes vers cet API et avec la base de données MYSQL, l'API donne la réponse avec un format JSON à notre application qui va décoder et faire ce qu'il veut avec la réponse

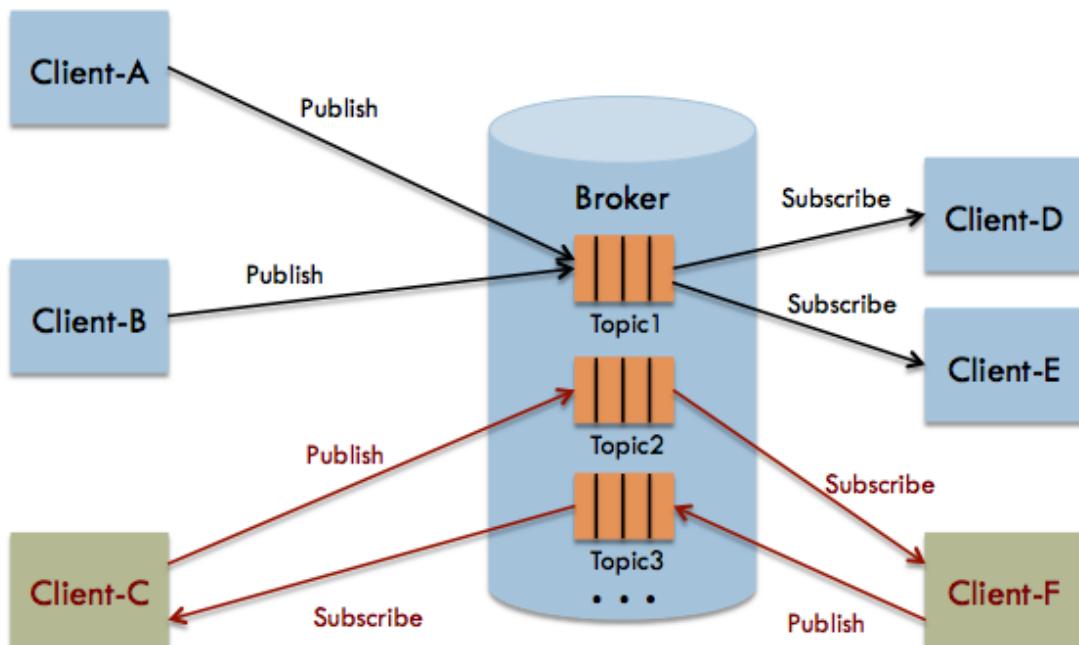


Figure 20. broker mqtt

dans notre projet quand un utilisateur envoie un message à un contact il fait une Publish sur un topic que le ce contact faisait subscribe sur . l'idée c'est de nommer le topic par l'id du contact et que chaque utilisateur subscribe à son identificateur personnel . cette idée me permettait d'envoi les messages en réel time aussi

3.5 Diagramme de séquences

3.5.1 Definition

Le diagramme de séquence est permis de représenter des scénarios d'un cas d'utilisation et aussi représenter l'interaction entre les objets en indiquant la chronologie des échanges.

3.5.2 Diagramme de séquence «S'Authentification»

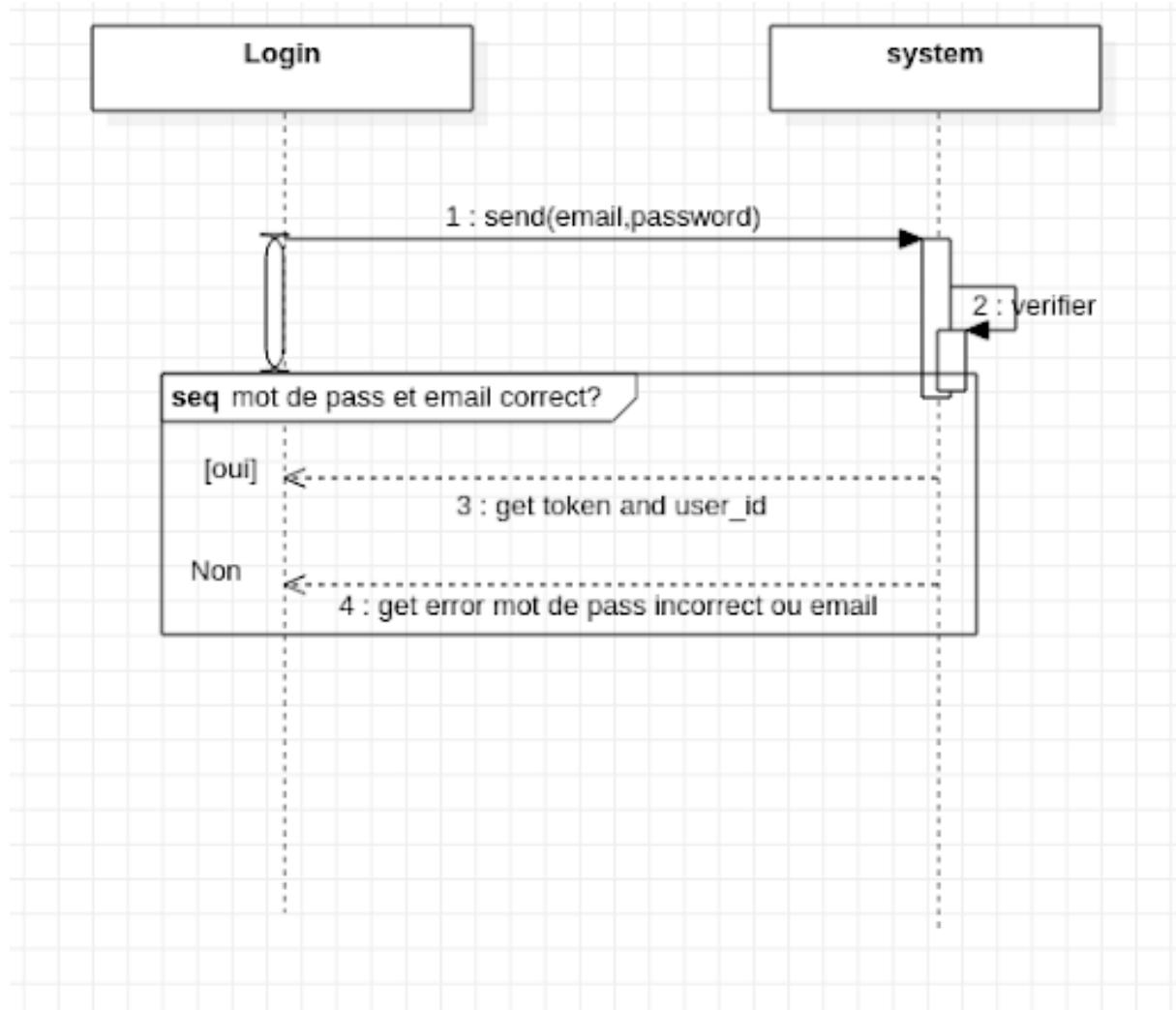


Figure 21. Diagramme de séquence «S'Authentification»

Figure 6 : Diagramme de séquence «S'Authentification». L'utilisateur (employee) choisit l'authentification si n'est pas identifiée puis saisit les données d'authentification. Le système va vérifier ses droits d'accès dans la base de données et lui répondre par la suite soit par passage à son espace soit par un passage vers même interface d'authentification.

3.5.3 Diagramme de séquence «Sign»

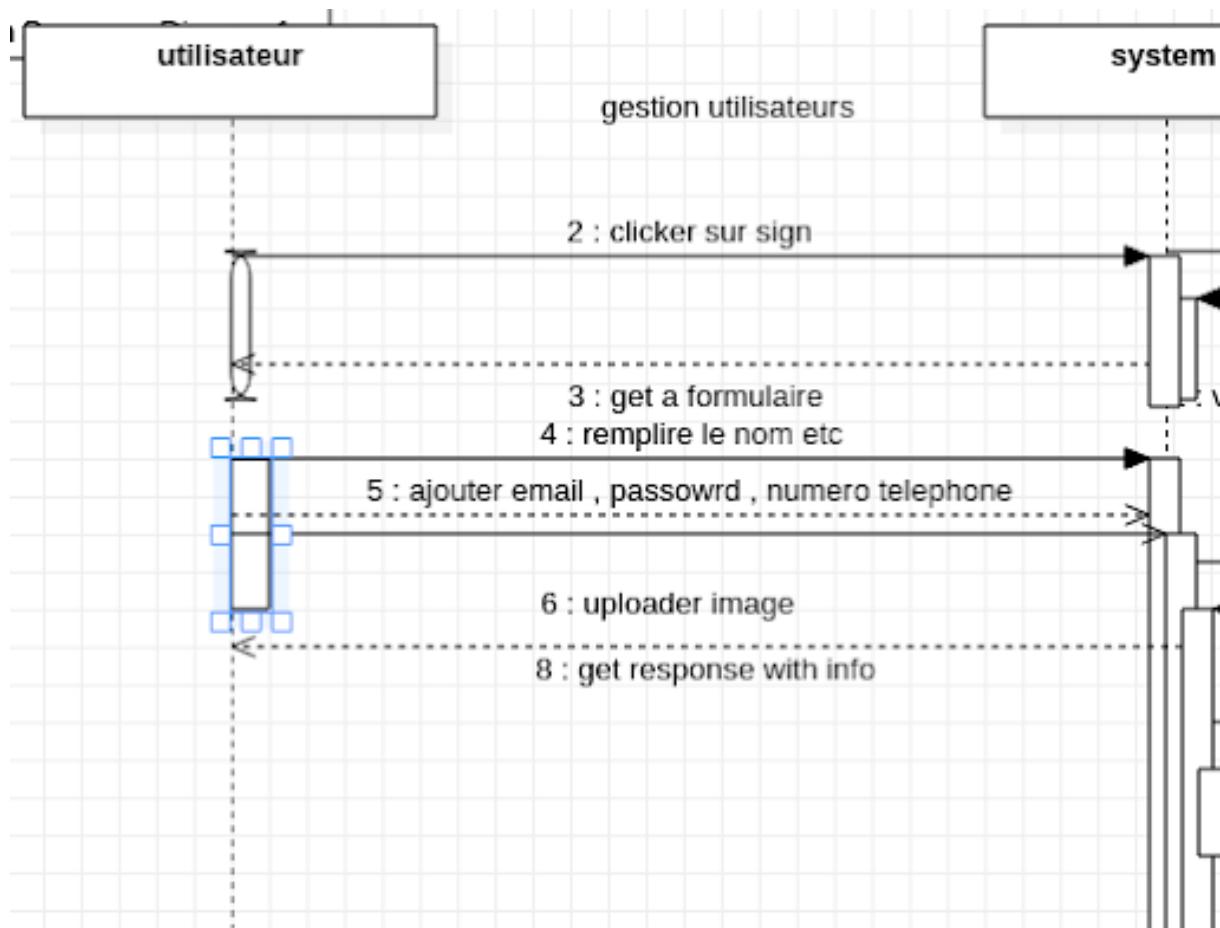


Figure 22. Diagramme de séquence «Sign»

Avant l'authentification si on veut ajouter quelqu'un à notre system a fait une clique sur le button sign . On obtient un formulaire avec quatre champs, username, email, et deux champs pour le mot de pass. l'utilisateur qui a enregistré peut accéder à son espace personnel .

3.5.4 Diagrammes de séquence «Messages»

Figure 8: Diagrammes de séquence «Messages».

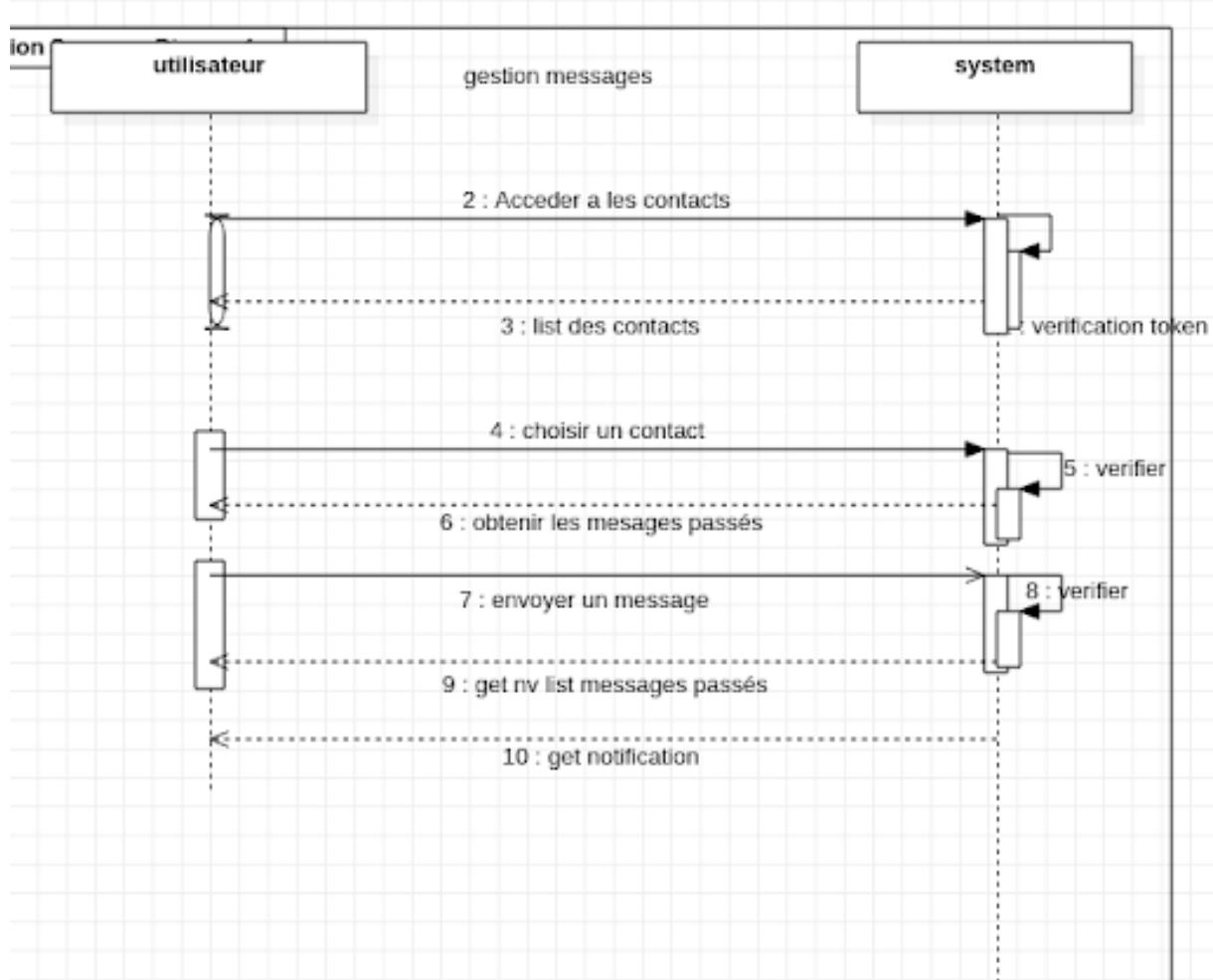


Figure 23. Diagrammes de séquence «Messages»»

L'utilisateur choisit un contact de ses contacts qui a ajouté il obtient une interface de messages avec tous les messages échangés avec la date d'envoyer et recevoir, et une EditText pour ajouter nouveau message, si un nouveau message il reçoit une notification...

3.5.5 Diagramme de séquence : «taches»

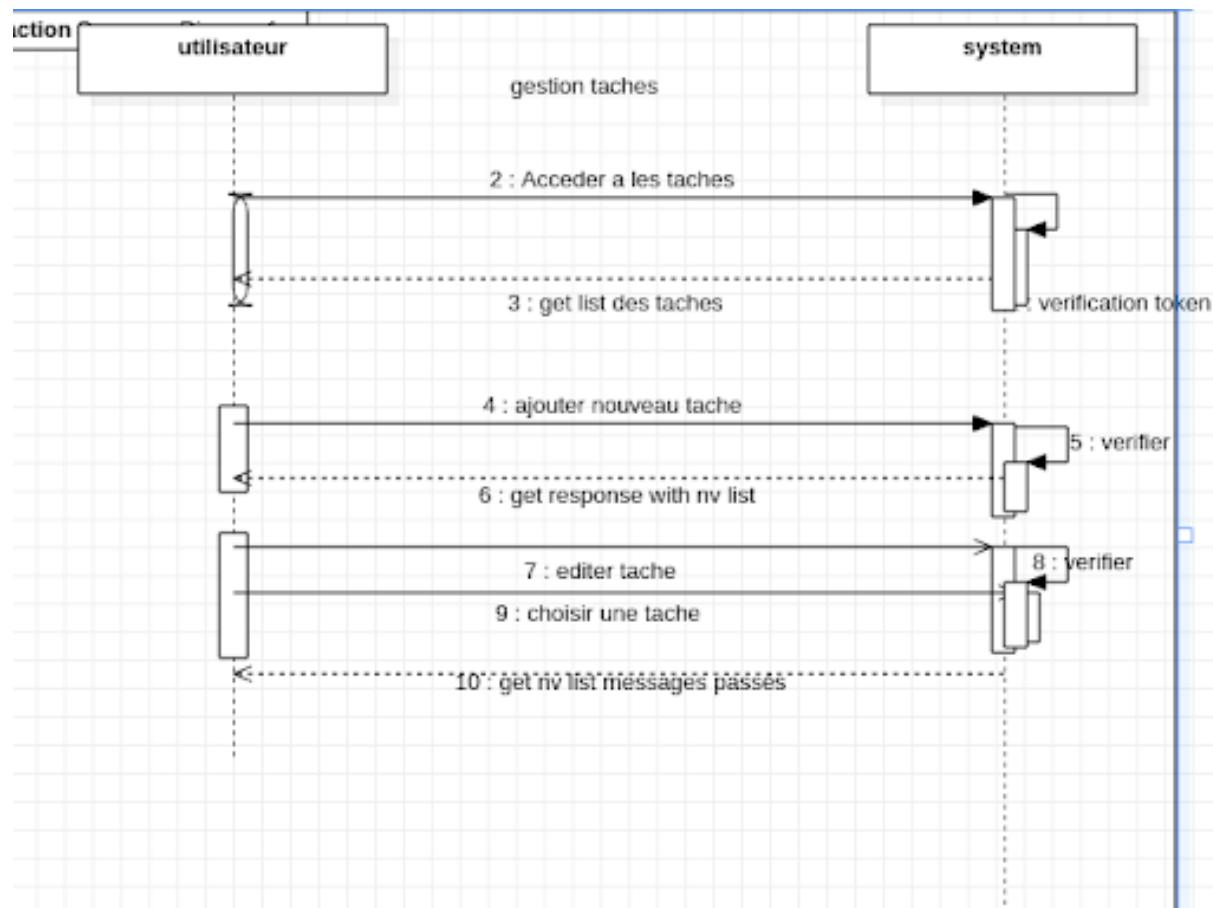


Figure 24. Diagramme de séquence : «taches»

3.5.6 Diagramme de séquence : «projects»

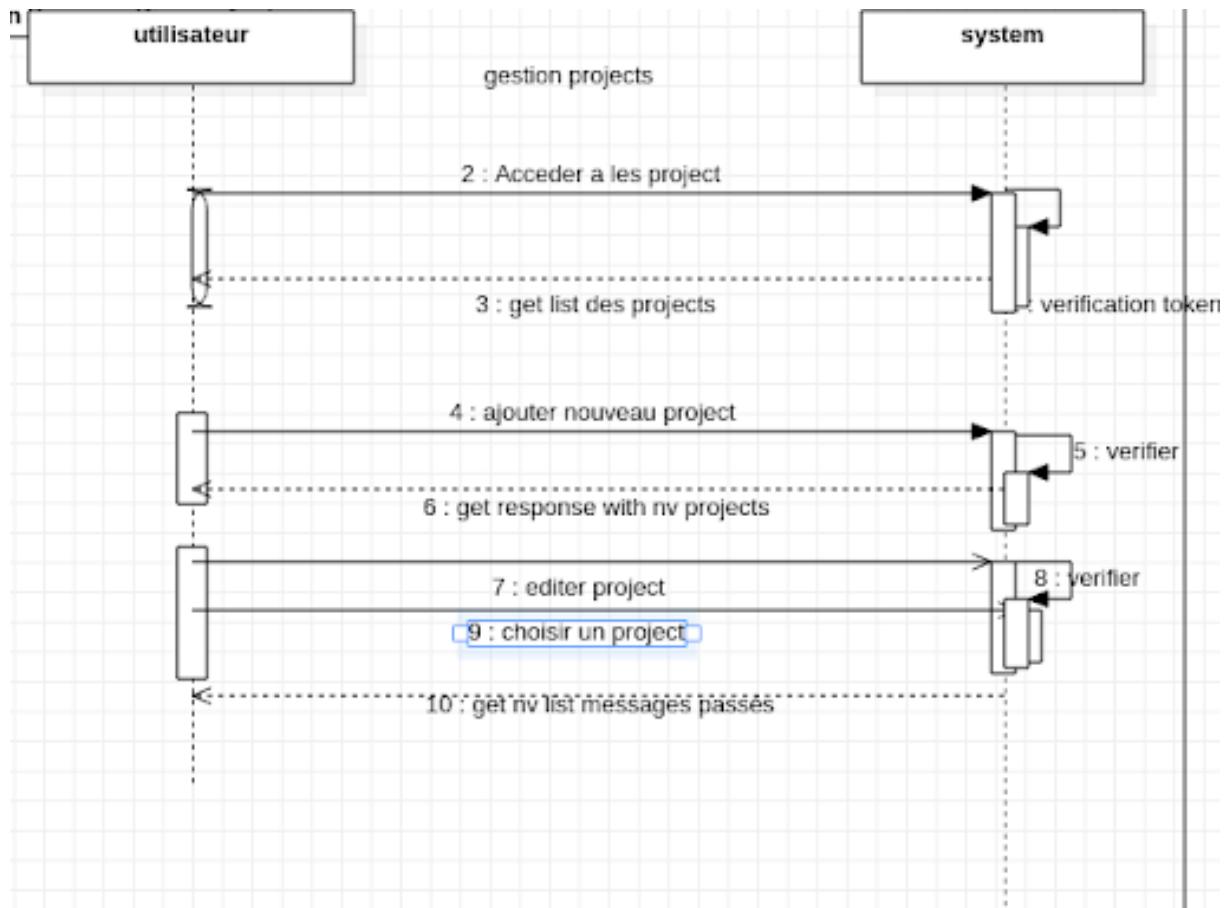


Figure 25. Diagramme de séquence : «projects»

3.6 Diagramme de class

3.6.1 Définition:

Le diagramme de classes est le point central dans le développement orienté objet. Coté analyse, il a pour objectif de décrire la structure des entités manipulées par les utilisateurs. Coté conception, le diagramme de classes représente la structure d'un code orienté objet ou, à un niveau de détail plus important, les modules du langage de développement.

3.6.2 Diagramme de class côté client

subsectionDiagramme de séquence : «taches»



Figure 26. Diagramme de séquence : «taches»

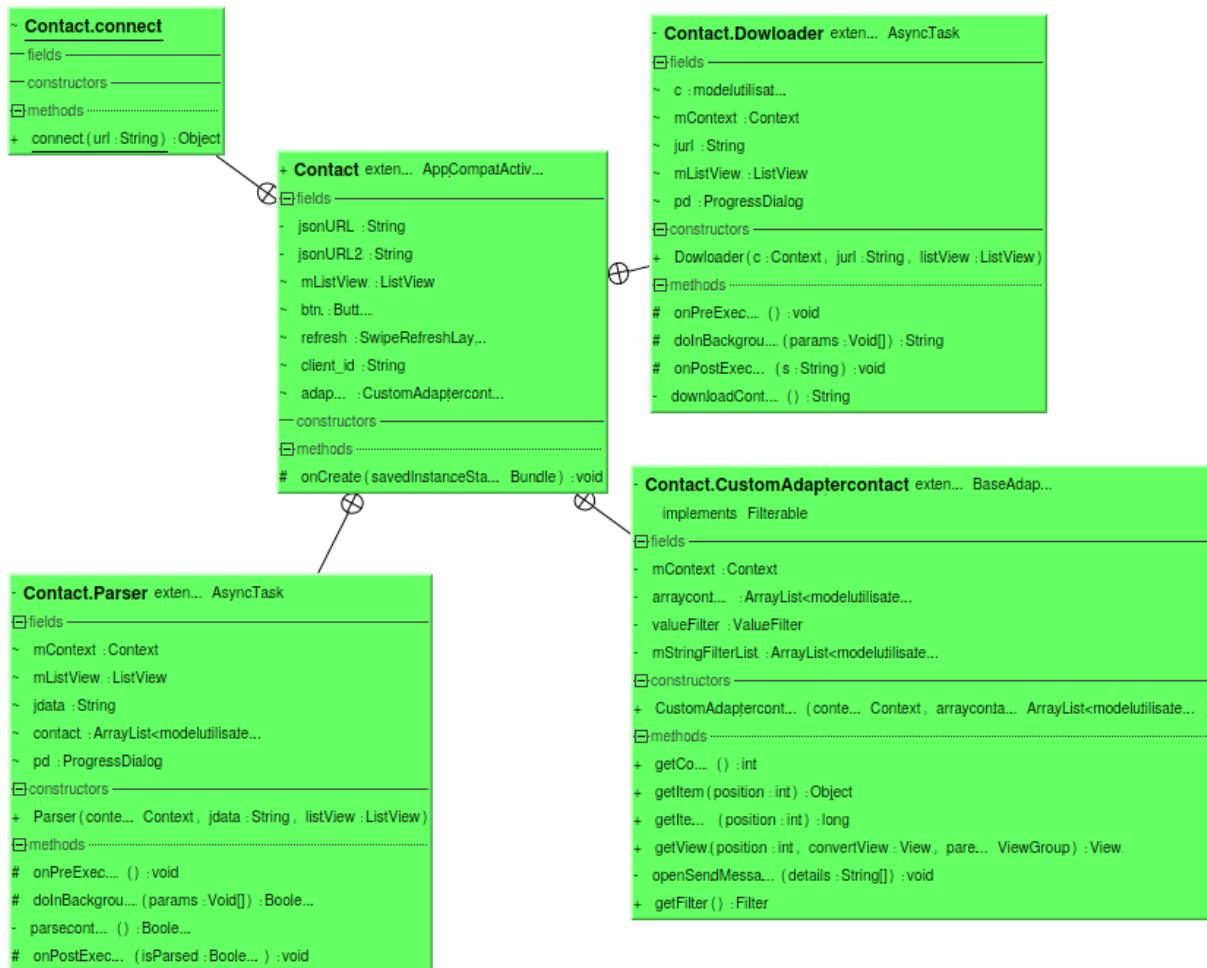


Figure 27. Diagramme de class

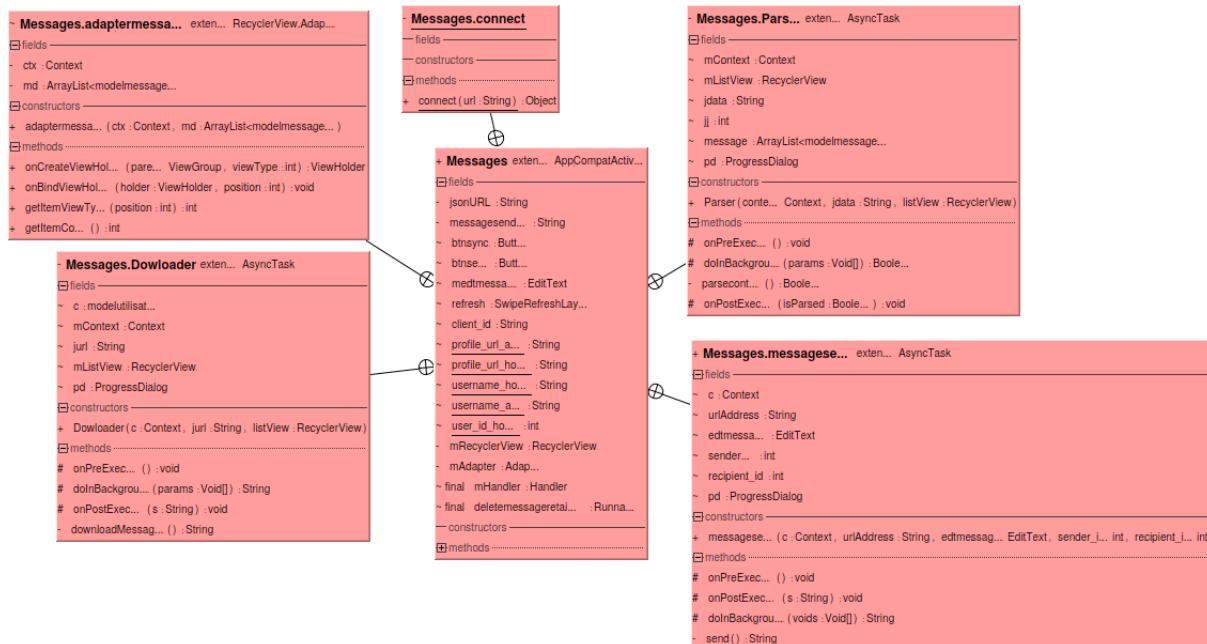


Figure 28. Diagramme de class client

3.6.3 Diagramme de class coté serveur

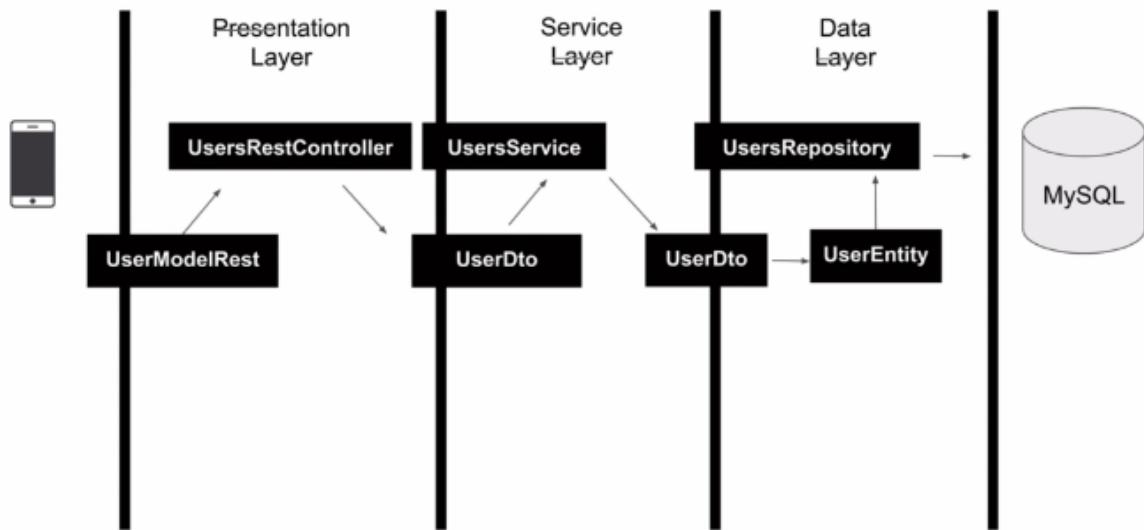


Figure 29. Diagramme de class serveur

la premiere couche c'est presentation couche et ca ou on trouve les controlleur , le telephone Android envoi une request POST,GET,DELETE,PUT , La request les donnees en format json qui est utilise par le rest controlleur qui va copier les attributs a des model partage entre le client et la base de donnees , le rest controlleur.

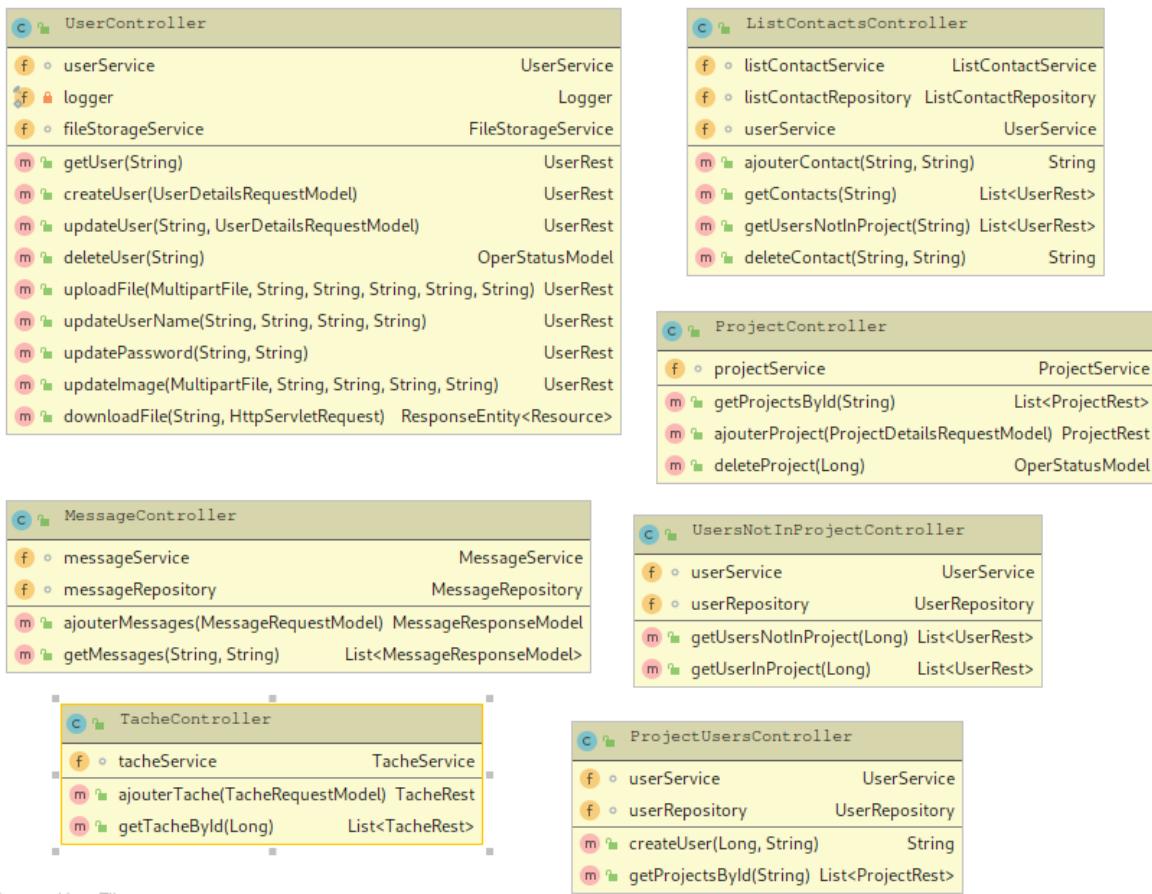


Figure 30. Diagramme de class Contrôleurs

les services c'est des classes annoté avec l'annotation @service de spring boot , et utilise pour contacter avec la base de données avec les classes annoté par repository et recuperer les données from base de données mysql

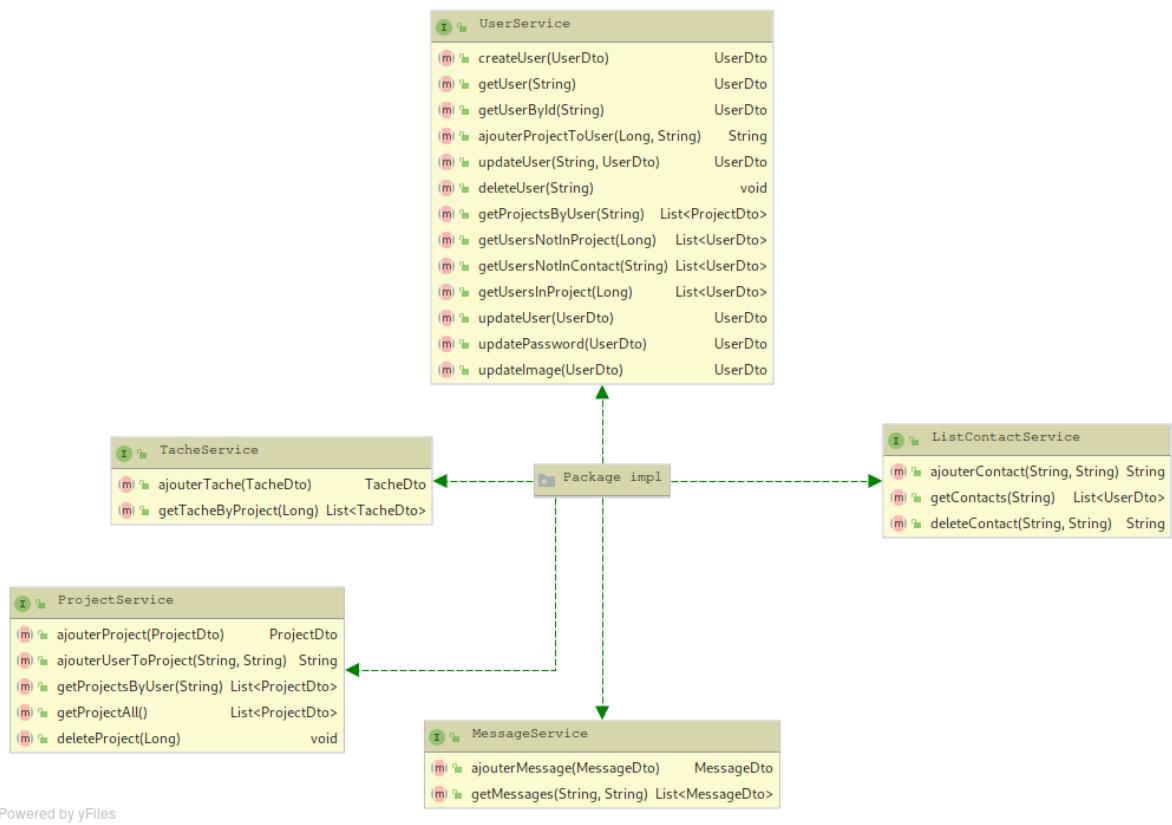
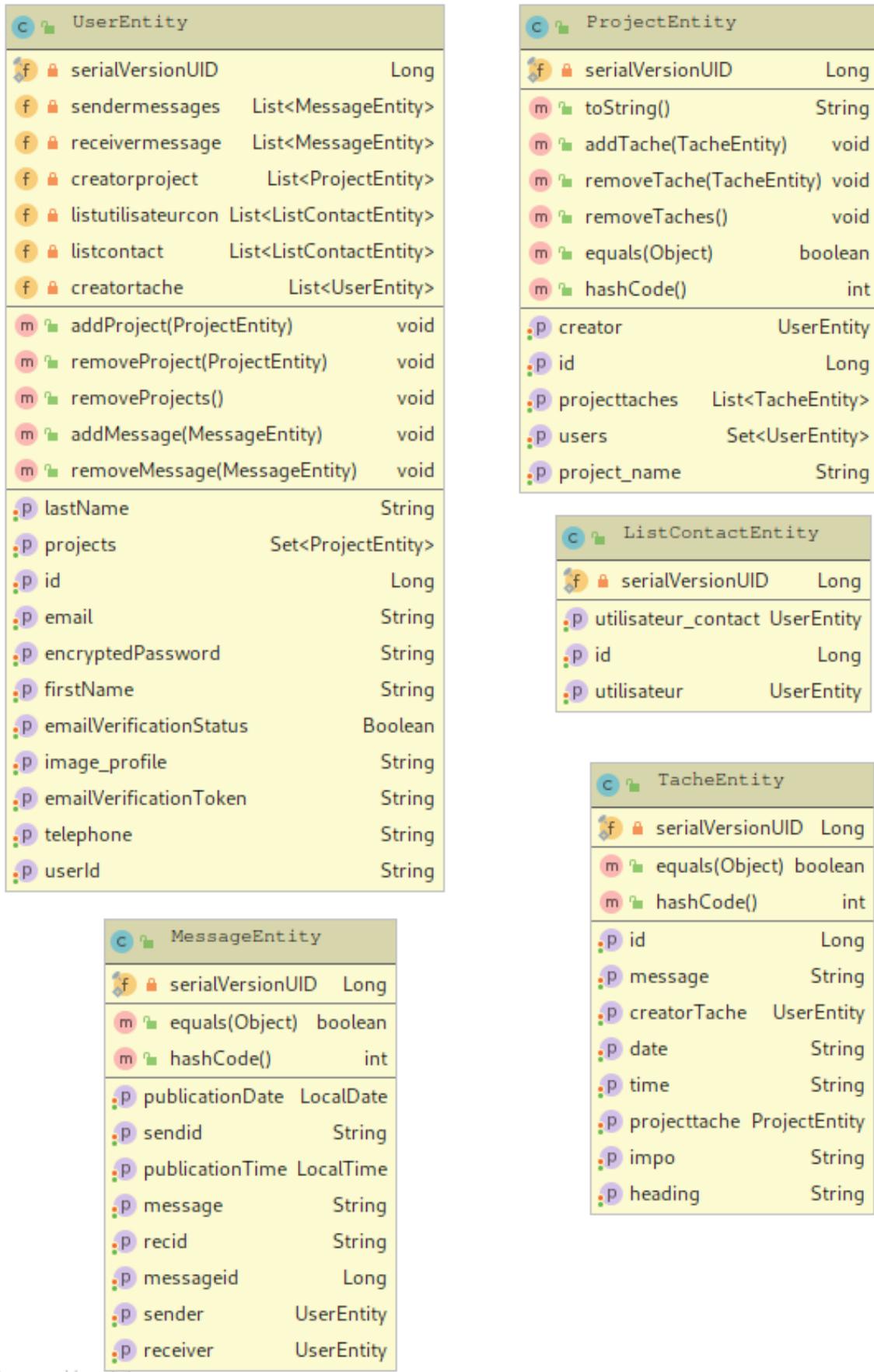


Figure 31. Diagramme de class services

les services c'est des classes annoté avec l'annotation @service de spring boot , et utilise pour contacter avec la base de données avec les classes annoté par repository et recuperer les données from base de données mysql .



Powered by yFiles

Figure 32. Diagramme de class entity

les entitys utilisent avec les repositorys pour récupérer, insérer, update dans la base de données.

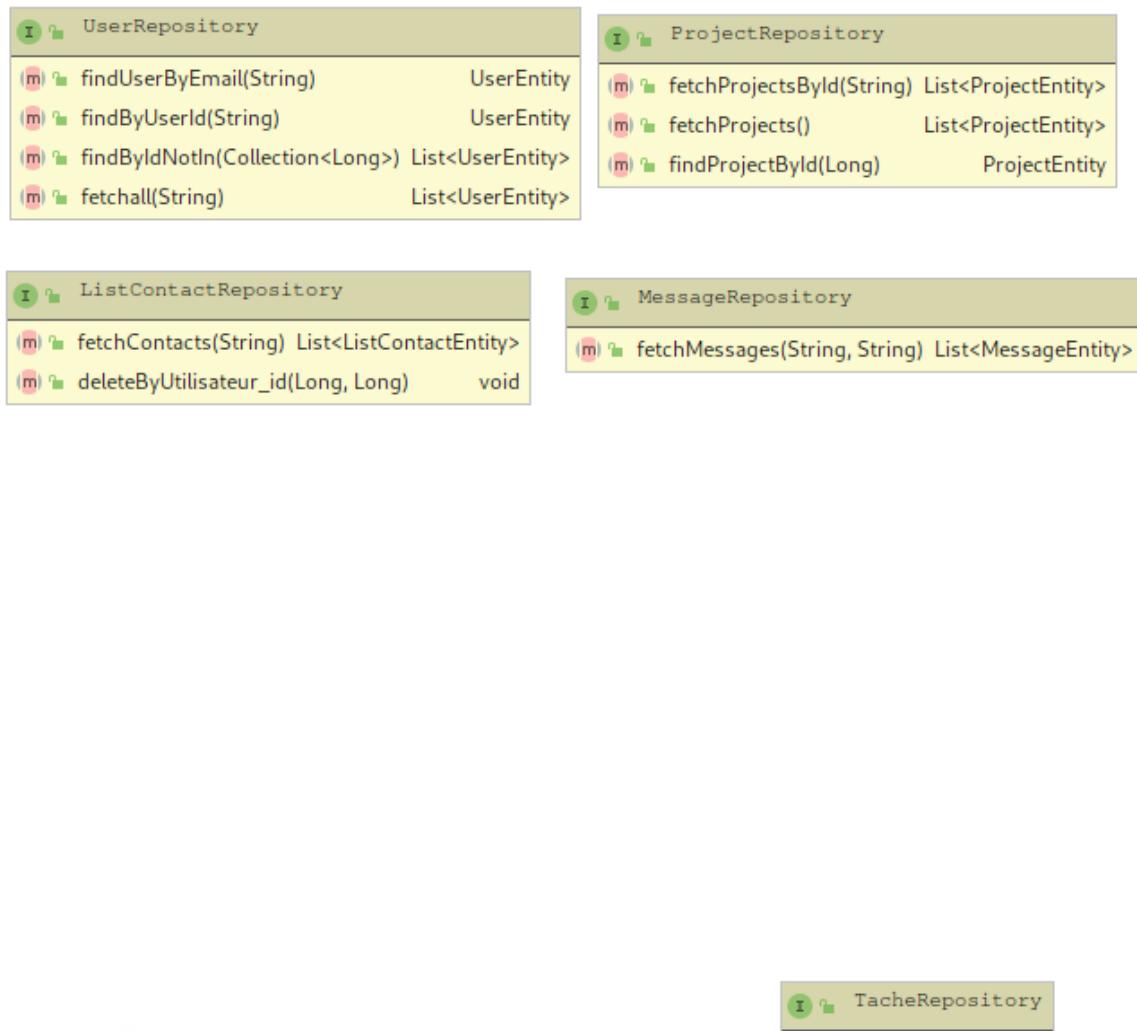


Figure 33. Diagramme de class repository

les repositorys utilise les entity pour recuperer , inserer , update dans la base de donnees

Chapter

4

Phase de Réalisation

Pour pouvoir mener à bien un projet informatique, il est nécessaire de choisir des techniques permettant de simplifier sa réalisation. Pour cela, après avoir complété l'étude conceptuelle dans le chapitre précédent, nous allons aborder la partie implémentation dans ce qui suit. J'entame ce chapitre la description des environnements matériels et logiciels qui nous ont permis de réaliser notre projet ainsi que l'architecture physique de notre système à travers le diagramme de déploiement. Nous passons ensuite à la phase d'implémentation dans laquelle Je vais présenter les différentes techniques que j'ai utilisées pour réaliser notre application.

4.1 Environnement matériel

Notre application est réalisée sur un pc portable avec un Smartphone pour l'exécution dont les caractéristiques sont les suivant:

	ordinateurs	telephone
Marque:	Hp compaq	Samsung Galaxy
Processeur:	Intel ® Core (™) i5	
Ram:	4 Go	1.5 Go
Operating system:	Gnu-linux	Android 7.1.1

Figure 34. matériel

4.2 Environnement logiciel

Les Logiciel Utiliser Pour la réalisation du Projet Sont:



Figure 35. Android

Android Studio: C'est un environnement de développement pour développer des applications mobiles Android. il est basé sur IntelliJ IDEA et utilise le moteur de production Gradle. Il peut être téléchargé sous les systèmes d'exploitation Windows, macOS et Linux



StarUML: StarUML est un logiciel de modélisation UML, qui a été cédé comme opensource par son éditeur, à la fin de son exploitation commerciale (qui visiblement continue ...), sous une licence modifiée de GNU GPL.



Apache Tomcat est un conteneur web libre de servlets et JSP. Issu du projet Jakarta, c'est un des nombreux projets de l'Apache Software Foundation. Il implémente les spécifications des servlets et des JSP du Java Community Process9, est paramétrable

par des fichiers XML et des propriétés, et inclut des outils pour la configuration et la gestion. Il comporte également un serveur HTTP.



NetBeans est un environnement de développement intégré, placé en open source par Sun en juin 2000 sous licence CDDL et GPLv2. En plus de Java, NetBeans permet la prise en charge native de divers langages tels le C, le C++, le JavaScript, le XML, le Groovy, le PHP et le HTML, ou d'autres par l'ajout de greffons



Figure 39. Vim

Vim Est un éditeur de texte, c'est-à-dire un logiciel permettant la manipulation de fichiers texte. Il est directement inspiré de vi (un éditeur très répandu sur les systèmes d'exploitation de type UNIX), dont il est le clone le plus populaire. Son nom signifie d'ailleurs Vi IMproved, que l'on peut traduire par VI Amélioré



Figure 40. Maria DB

Est un système de gestion de base de données édité sous licence GPL. Il s'agit d'un fork communautaire de MySQL : la gouvernance du projet est assurée par la fondation MariaDB, et sa maintenance par la société Monty Program AB, créateur du projet. Cette gouvernance confère au logiciel l'assurance de rester libre.



Figure 41. postman

Postman Postman un logiciel qui se focalise sur les tests des API. Il est devenu très populaire pour tester les Microservices, notamment grâce à sa simplicité et ses fonctionnalités très spécialisées.



Figure 42. Mosquitto

Est un serveur MQTT Open Source (Broker) que l'on peut installer sur un Raspberry Pi mais aussi sur presque toutes les plateformes (macOS, Windows, Linux...). MQTT facilite la communication entre objets connectés (M2M) tout en économisant la batterie.

4.3 Choix technique



Figure 43. Java

Est un langage de programmation orienté objet qui permet une programmation orientée-objet. Ce langage a l'avantage d'être modulaire, rigoureux (la plupart des erreurs se produisent à la compilation et non à l'exécution) et portable (un même programme Compilé peut s'exécuter sur différents environnements).

4.4 Choix technique



Figure 44. Spring boot

Spring Boot est un framework développé par Pivotal en 2012 qui connaît depuis environ 4 ans, une explosion du nombre d'utilisateurs. C'est un point d'entrée unique vers tous les projets de la IO Foundation (batch, ligne de commande, web..) que vous pourrez donc utiliser dans votre application.

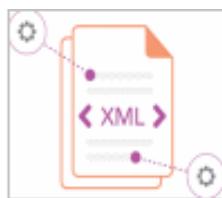


Figure 45. XML

Est un langage de balisage extensible méta langage informatique de balisage générique qui est un sous-ensemble du Standard Generalized Markup Language (SGML). Sa syntaxe est dite « extensible » car elle permet de définir différents langages avec chacun leur Vocabulaire et leur grammaire, comme XHTML, XSLT, RSS, SVG. . .

4.5 Format De données Communiquées

JSON (JavaScript Object Notation) est un format léger d'échange de données. Il peut être aisément analysé et généré par des machines.

Lorsque l'application Android s'exécute et veut récupérer des données, elle se connectera au service Web (Api) à travers des rest controllers. le contrôleur va récupérer la requête qui contient des données json et le type de requête soit POST, GET, DELETE, PUT. le rest controller va utiliser un service qui va utiliser Jpa repository api qui récupère

les données depuis la base de données My sql. Ensuite les données seront encodées au format JSON et envoyées au système Android. Ensuite, l'application Android va obtenir ces données codées. Elle les analysera et les affichera sur l'appareil Android.

Format compréhensible par tous (humain et machine). Aucun apprentissage n'est requis puisque la syntaxe n'utilise que quelques marques de ponctuations, plus sa structure en arborescence et sa syntaxe simple qui lui permet de rester très "léger" et efficace.

4.6 Choix de la technologie de sécurité

Spring Security est un Framework de sécurité léger qui fournit une authentification et un support d'autorisation afin de sécuriser les applications Spring. Il est livré avec des implémentations d'algorithmes de sécurité populaires.

JWT pour JSON Web Token est une méthode sécurisée d'échange d'informations, L'information est échangée sous la forme d'un jeton signé afin de pouvoir en vérifier la légitimité. Ce jeton est compact et peut être inclus dans une URL sans poser de problème. Afin d'assurer la sécurité des données et des informations personnelles des utilisateurs, j'ai utilisé Spring security pour authentifier les clients et générer le token Jwt pour chaque utilisateur et qui va utiliser ce token pour l'autorisation de ces requêtes à api pour récupérer les données choisies que chaque utilisateur doit saisir ses droits d'accès (login et mot de passe) pour consulter l'ensemble des services qui lui et associés.

4.7 Présentation de quelque interface

Dans ce qui suit nous présentons quelques interfaces de l'application en citant les détails de chaque imprimé écran

4.7.1 Interface Splash Screen

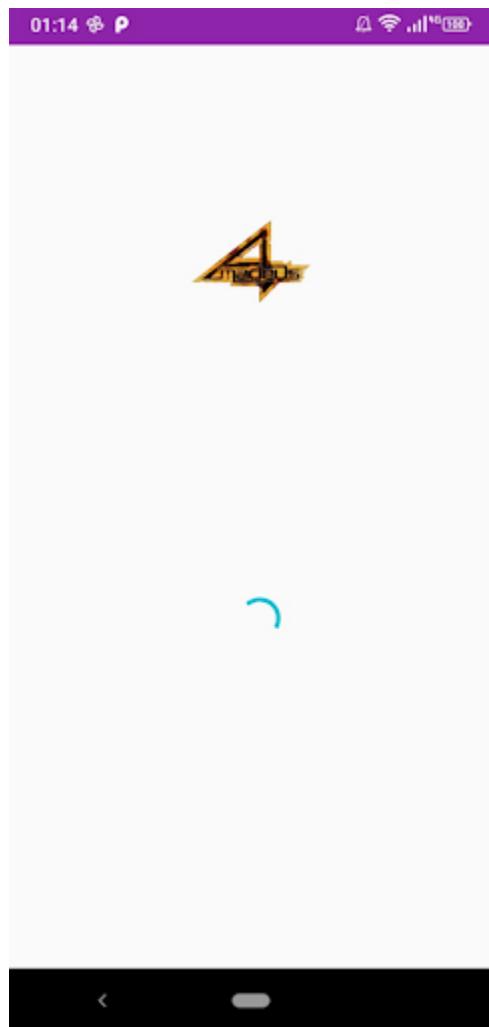
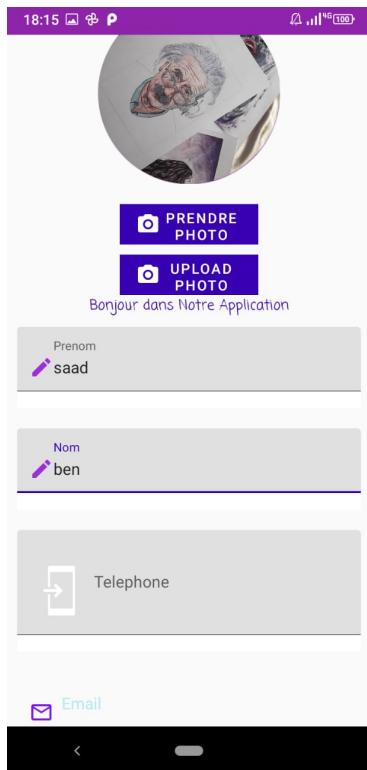
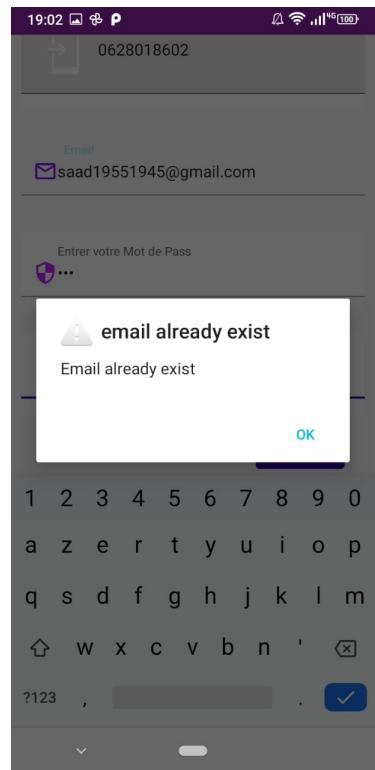


Figure 46. Splash Screen

dans cette interface il vérifie la connexion au serveur local . Et il vérifie quelle session doit afficher . Si l'utilisateur et déjà connecté il doit rester connecté à son profil . S'il déconnectait ou n'est pas coché rester connecté autant d'authentification il afficher page d'authentification.



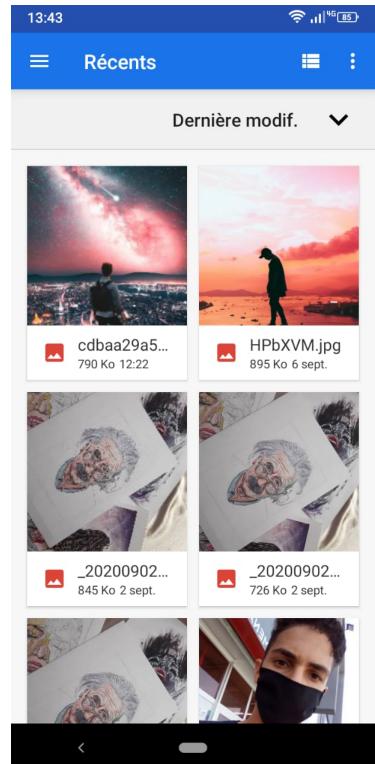
(a) remplire la formulaire



(b) email déjà existe



(c) prendre photo



(d) upload image

Figure 47. interfaces sign in

4.7.2 Interface Idntifier(Sign in) nouvel utilisateur

si quelqu'un veut enregistrer à notre système il doit utiliser cette interface pour ajouter ses informations personnelles à notre base de données , il prendre une photo ou uploader une photo aussi il a besoin de remplir les informations

4.7.3 Authentication

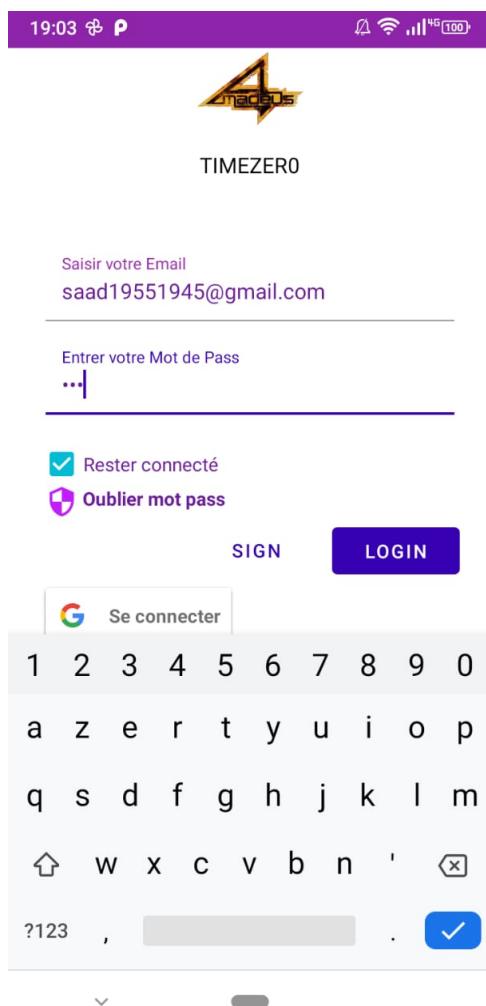


Figure 48. Interface d'authentification

dans cette interface les utilisateurs identifiaient ou choisissaient l'option identifier ou si quelqu'un oublier son mot de pass il suffit de donner son email pour initialiser le mot de pass et envoyer un message à son email électronique en utilisant le mail serveur postfix.

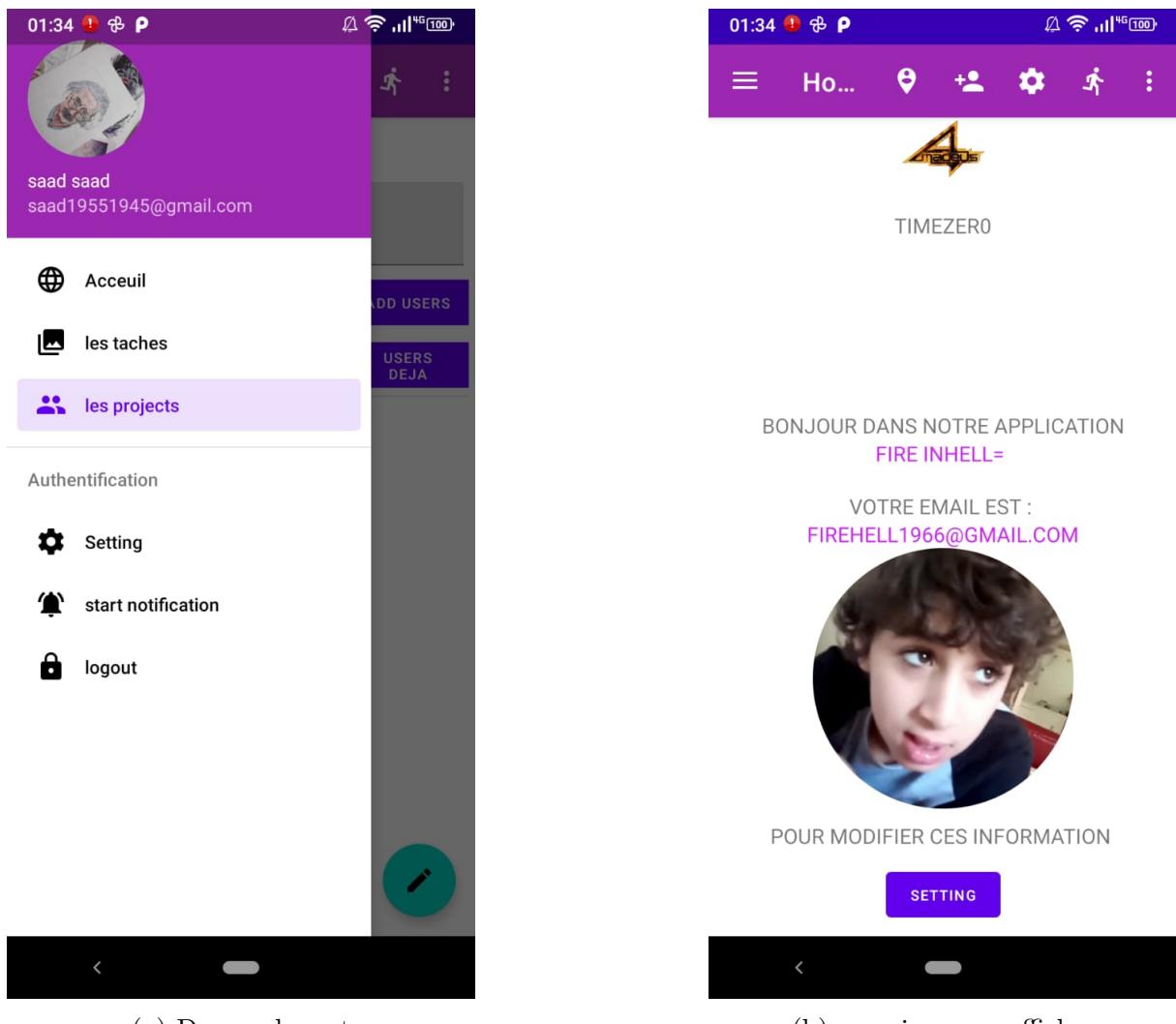


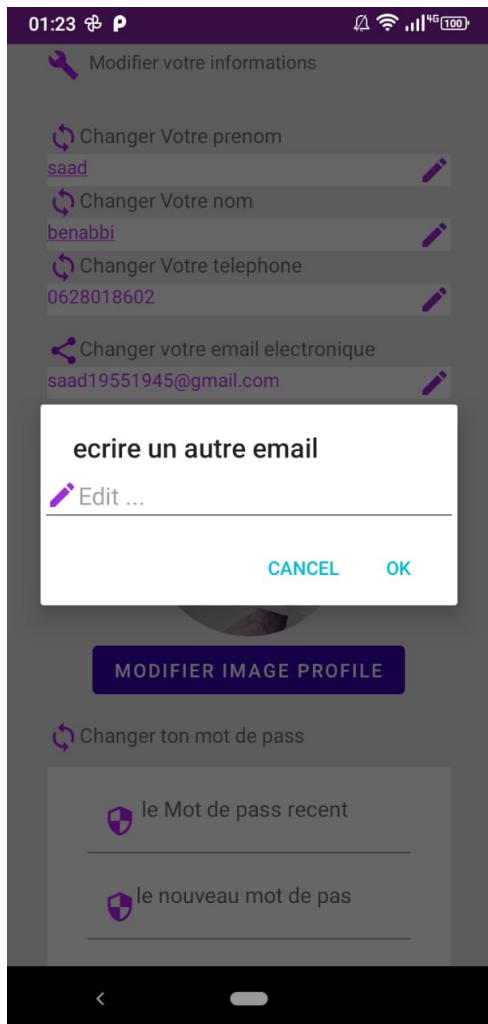
Figure 49. interfaces acceuil

4.7.4 Interface Acceuil

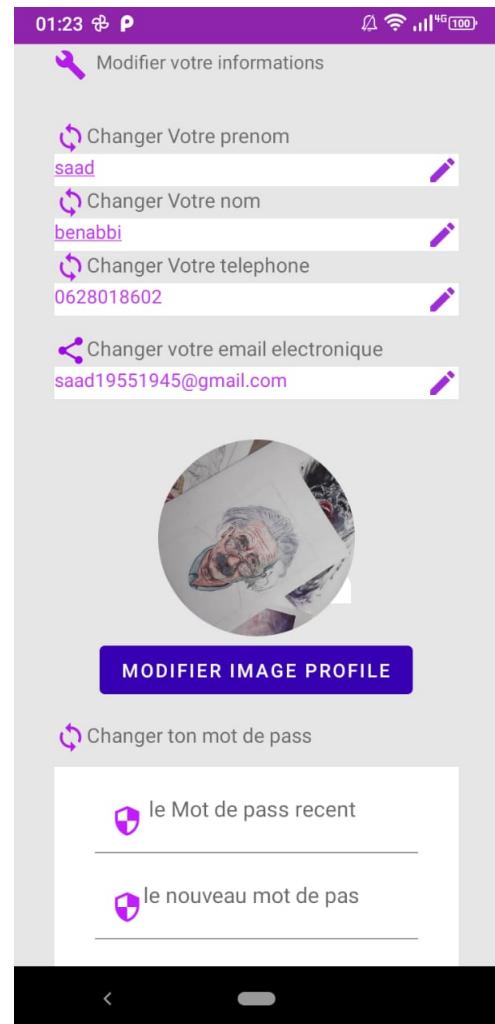
Dans cette interface on a le menu (Drawer layout) on a plusieurs items parmi eux on a l'accueil ou on a les informations de l'utilisateur connecté contact ou on a les contacts setting pour modifier des informations personnelles ; start notification pour ajouter push notification.

4.7.5 Interface Setting :

dans cette interface où l'utilisateur peut changer ses informations personnelles, photo se profile, nom, numéro téléphone.



(a) interface settings



(b) interface settings

Figure 50. interfaces Settings

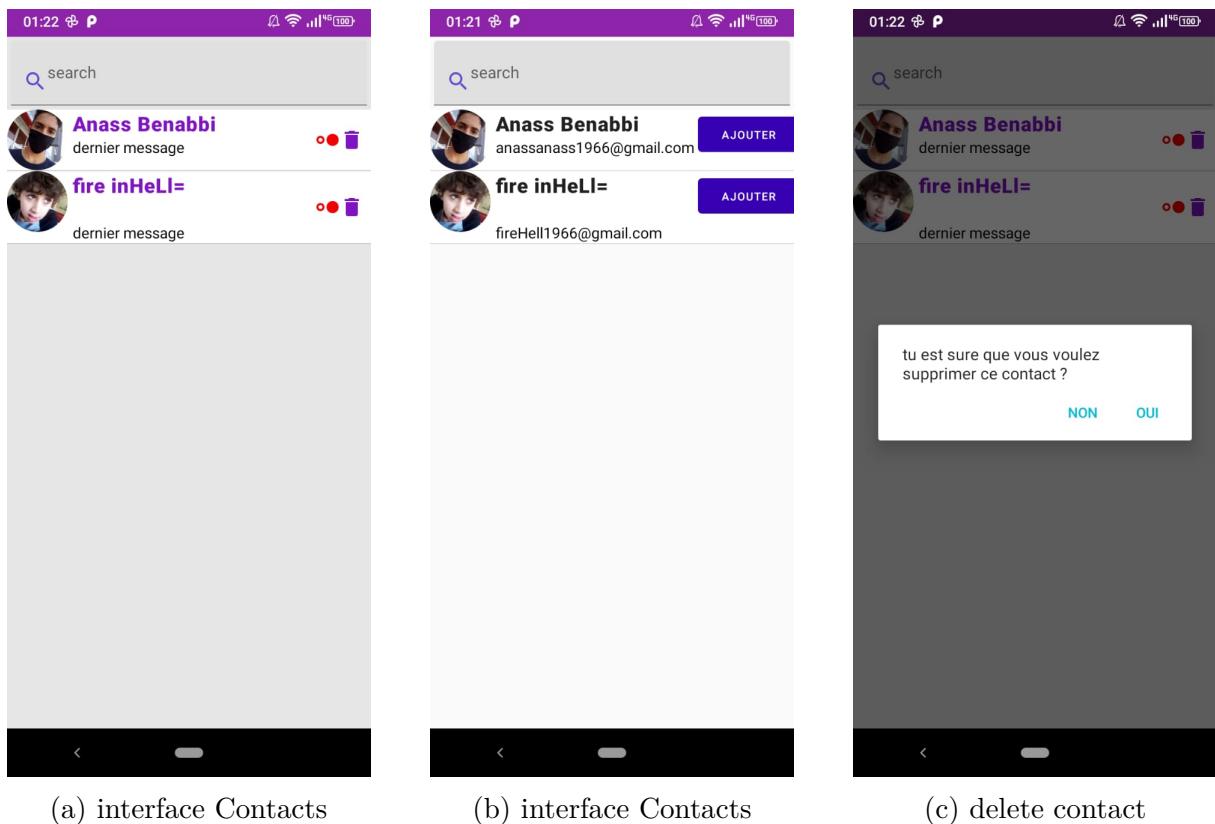


Figure 51. interfaces Contacts

4.7.6 Interface Contacts

cette interface est utilisée par l'utilisateur pour voir ses contacts, qui ont déjà ajouté par cet utilisateur.

4.7.7 interface messages

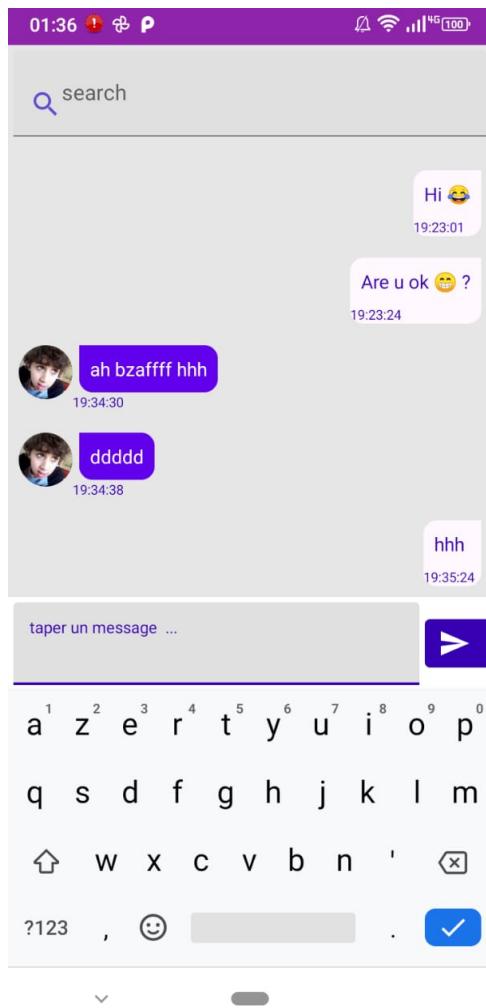


Figure 52. Interface messages

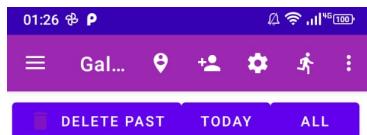
par cette interface chaque utilisateur peut envoyer en réel time des messages et reçoit des messages.

4.7.8 interface ajoutée taches

cette interface utilisait pour ajouter des nouvelles taches.

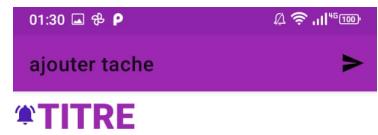
4.7.9 interface modification tache

Cette interface utilisée pour éditer une tache déjà existe



This is gallery fragment

ajouter une tache



alarme

donnez un titre
tache urgent

06 / 09 /

01 : 31

✉ MESSAGE

écrire un message descriptif
urgent pas important

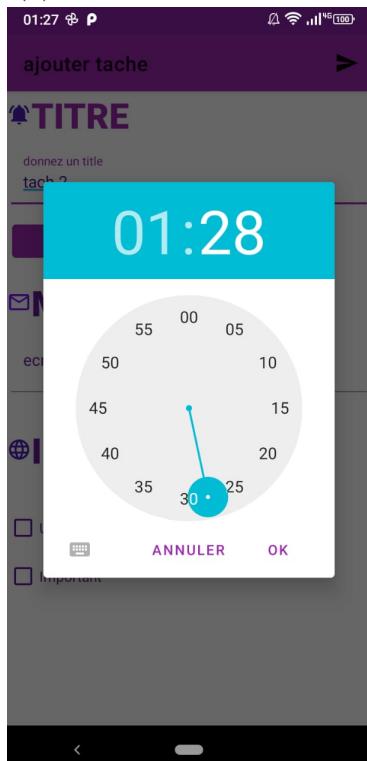
⌚ IMPORTANCE

Urgent

Important



(a) interface ajoutée taches



(c) interface ajoutée taches



(b) interface ajoutée taches



(d) interface ajoutée taches

Figure 53. interface ajoutée taches

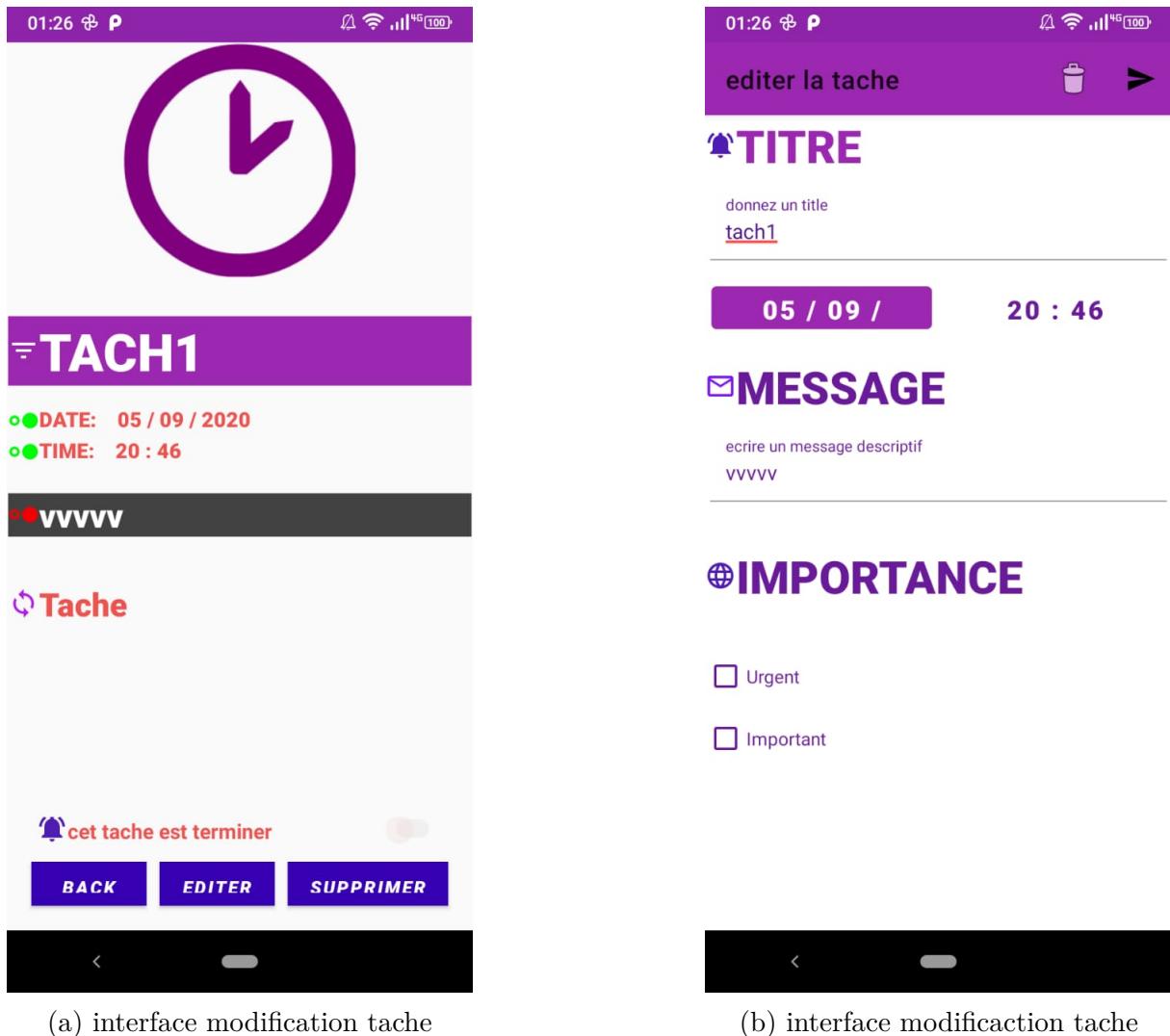


Figure 54. interface modification taches

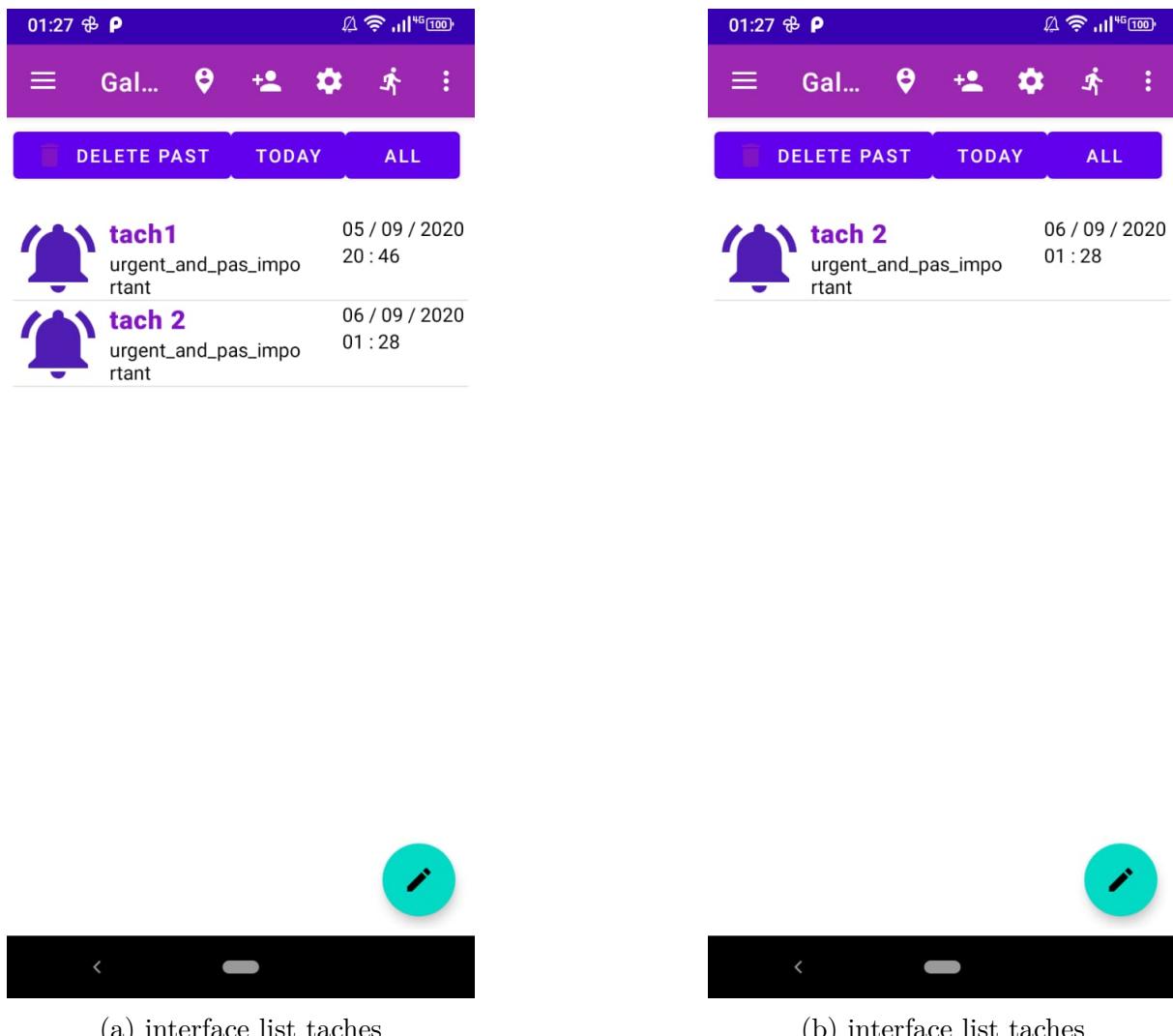


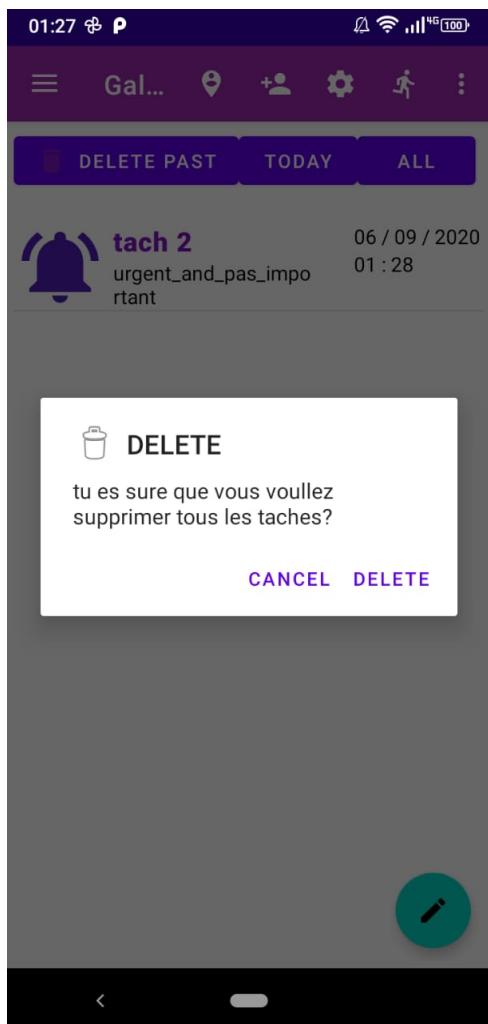
Figure 55. interface list taches

4.7.10 interface list taches

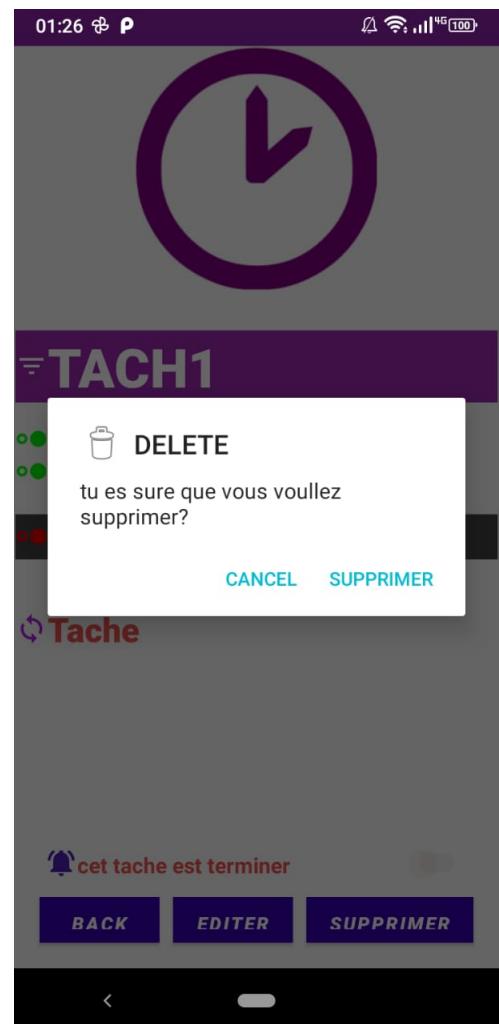
interface utilise pour liste les taches, et c'est une liste view, quand on click sur une view on pass a interfacé éditer tache.

4.7.11 interface suppression des tache

on utilise cette interface pour supprimer des taches déjà existe.

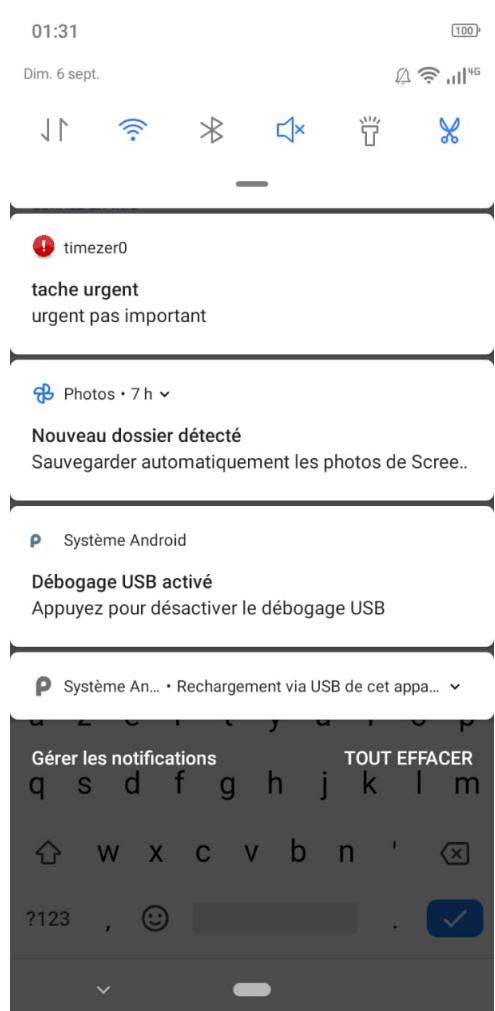


(a) interface suppression des tache



(b) interface suppression des tache

Figure 56. interface suppression des tache



(a) interface notification



(b) interface notification

Figure 57. interface notification

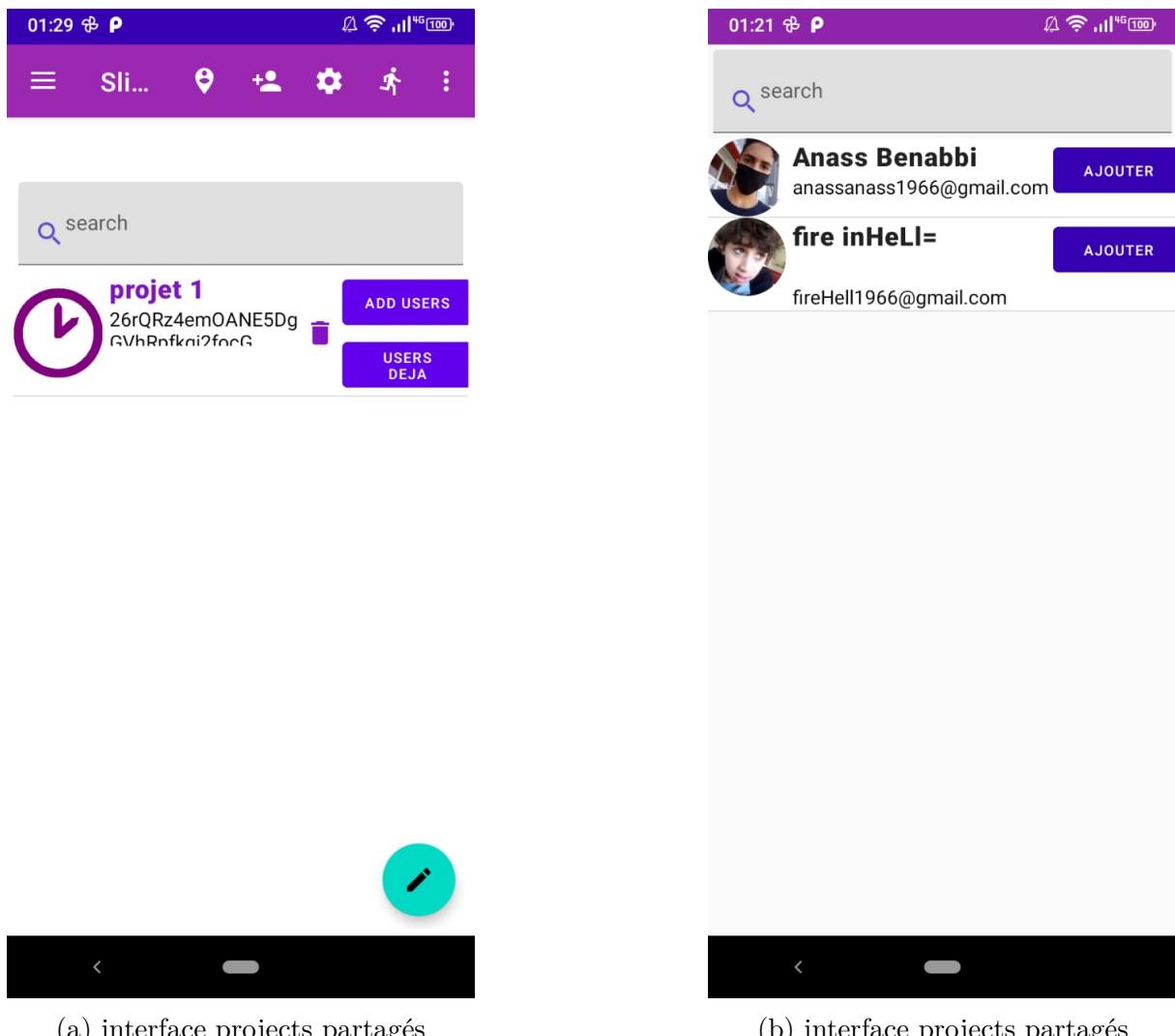


Figure 58. interface projects partagés

4.7.12 interface notification

4.7.13 interface projects partagés

on utilise cette interface pour ajouter/supprimer des projets . aussi on utilise pour ajouter des utilisateurs à des projets et supprimer des utilisateurs.

aussi c'est une liste view c'est on cliqué sur une view on obtient interface taches.

4.7.14 curl client pour récupérer des données from Restful API

```
zer0@zer0Monster:~$ curl -i -X POST http://10.0.0.1:8080/users/login -H 'Accept: application/json' -H 'Content-type: application/json' -d '{"email":"saad19551945@gmail.com","password":"123"}'
HTTP/1.1 200
Authorization: Bearer eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJzYWFkMTk1NTE5NDVAZ21haWwUY29tIiwiZXhwIjoxNjAwMjE
30DUwfQ.l8gTsAtOPj7KI0z5KpPGJFzuI-sW6bT4wKKKXWsTq9GcuVj0m0lQP3hdjsDxS9vyAxCV292MlQ9zkby7mC9oaA
UserID: 26rQRz4emOANE5DgGVhRpfkqj2focG
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Expires: 0
X-Frame-Options: DENY
Content-Length: 0
Date: Sun, 06 Sep 2020 00:57:30 GMT
```

Figure 59. curl

4.7.15 curl client pour récupérer des données from Restful API

```
zer0@zer0Monster:~$ curl -i -X POST http://10.0.0.1:8080/users/login -H 'Accept: application/json' -H 'Content-type: application/json' -d '{"email":"saad19551945@gmail.com","password":"123"}'
HTTP/1.1 200
Authorization: Bearer eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJzYWFkMTk1NTE5NDVAZ21haWwUY29tIiwiZXhwIjoxNjAwMjE
30DUwfQ.l8gTsAtOPj7KI0z5KpPGJFzuI-sW6bT4wKKKXWsTq9GcuVj0m0lQP3hdjsDxS9vyAxCV292MlQ9zkby7mC9oaA
UserID: 26rQRz4emOANE5DgGVhRpfkqj2focG
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Expires: 0
X-Frame-Options: DENY
Content-Length: 0
Date: Sun, 06 Sep 2020 00:57:30 GMT
```

Figure 60. curl

4.7.16 curl client pour récupérer des données from Restful API

la réponse est encodée soit en XML ou json, la réponse dépend sur l'header envoyé, c'est l'header accepté et application/json, la réponse est en json, c'est l'header accepté de la requête et application/XML la réponse est en XML.

```
[zer0@zer0Monster ~]
[zer0@zer0Monster ~]$ curl -H 'Authorization: Bearer eyJhbGciOiJIUzUxMiJ9eyJzdWIiOiJzYWFkMTk1NTE5NDVAZ21haWwuY29tIiwiZXhwIjoxNjAwMjE3ODUwfQ.l8gTsAtOPj7KI0z5KpPGJFzuI-sW6bT4wKKKXWsTq9GcuVj0mOlQP3hdjsDxS9vyAxCV292ML09zkby7mC9oaA' -H 'Accept: application/json' -X GET "http://10.0.0.1:8080/users/26rQRz4emOANE5DgGVhRpfkj2focG"
{"id":1,"telephone":"0628018602","image_profile":"http://10.0.0.1:8080/users/downloadFile/1599351472721.png","userId":"26rQRz4emOANE5DgGVhRpfkj2focG","firstName":"saad","lastName":"saad","email":"saad19551945@gmail.com"}[zer0@zer0Monster ~]$ 
```

Figure 61. curl

```
[zer0@zer0Monster ~]$ curl -H 'Authorization: Bearer eyJhbGciOiJIUzUxMiJ9eyJzdWIiOiJzYWFkMTk1NTE5NDVAZ21haWwuY29tIiwiZXhwIjoxNjAwMjE3ODUwfQ.l8gTsAtOPj7KI0z5KpPGJFzuI-sW6bT4wKKKXWsTq9GcuVj0mOlQP3hdjsDxS9vyAxCV292ML09zkby7mC9oaA' -H 'Accept: application/xml' -X GET "http://localhost:8080/users/26rQRz4emOANE5DgGVhRpfkj2focG"
<UserRest><id>1</id><telephone>0628018602</telephone><image_profile>http://10.0.0.1:8080/users/downloadFile/1599351472721.png</image_profile><userId>26rQRz4emOANE5DgGVhRpfkj2focG</userId><firstName>saad</firstName><lastName>saad</lastName><email>saad19551945@gmail.com</email></UserRest>[zer0@zer0Monster ~]$ 
```

Figure 62. curl

4.7.17 403 forbidden access

ici c'est l'utilisateur n'est pas autorisé, n'a pas le token qui l'autorisee, il va recevoir une response status code 403.

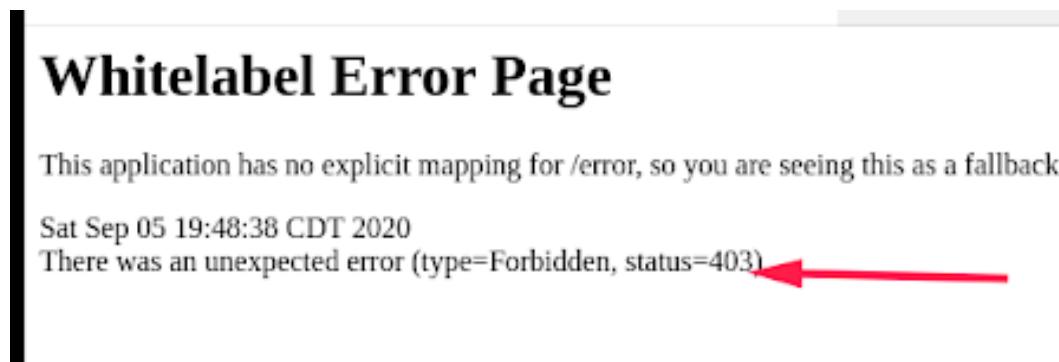


Figure 63. curl

4.7.18 les uri

ici c'est la plupart des uri utilisé pour acceder a les resources de notre service web.

```
// mqtt url :  
public static String urlmqtt = "tcp://10.0.0.1:1883";  
//url for project  
public static String ajouterProject= "http://10.0.0.1:8080/projects/";  
public static String getProjects = "http://10.0.0.1:8080/projects";  
public static String deleteproject="http://10.0.0.1:8080/projects/";  
public static String getnotinyourlistuserproject = "http://10.0.0.1:8080/projectusers/in";  
public static String ajouternvuserproject ="http://10.0.0.1:8080/projectusers/notin";  
public static String getuserbyid ="http://10.0.0.1:8080/users/";  
public static String signUrl = "http://10.0.0.1:8080/users/sign";  
public static String getnotinyourlistcontact="http://10.0.0.1:8080/contacts/notin";  
public static String ajouternvcontact="http://10.0.0.1:8080/contacts/";  
public static String getcontact22="http://10.0.0.1:8080/contacts";  
public static String deletecontact2="http://10.0.0.1:8080/contacts/delete";  
public static String ajouternvmessage2="http://10.0.0.1:8080/messages";  
public static String getmessages2="http://10.0.0.1:8080/messages";  
public static String updateimage="http://10.0.0.1:8080/users/updateimage";  
public static String updateuser="http://10.0.0.1:8080/users/updateuser";  
public static String updatepassword="http://10.0.0.1:8080/users/updatepassword";  
public static String login="http://10.0.0.1:8080/users/login";  
public static String deletemessage;
```

Figure 64. uri



Conclusion

Au cours de ce projet, et dans la première chapitre j'ai proposé une solution de la problématique de cahier des charges.

Dans le deuxième chapitre État de l'art on a parlé des frameworks qu'on a utilisés, historique et pourquoi. on a parlé d'android, Spring boot avec ces dependencies spring web, Spring data jpa, Spring

Dans le troisième chapitre j'ai présenté les différents besoins fonctionnels et non fonctionnels détaillé, j'ai aussi réalisé le diagramme de classes pour faciliter de comprendre le fonctionnement de l'application. Enfin il y a schéma relationnel MCD (Module logique de données).

Dans le quatrième chapitre de la branche technique, J'ai présenté l'architecte de cette application avec la réalisation des diagrammes de séquences pour détailler la l'interaction entre les utilisateurs avec le system avec explication de chaque diagramme, et aussi le diagramme de classes.

Dans le dernier chapitre j'ai Présenté les différentes techniques comme les logiciels utilisés et les choix techniques (les langages de programmation), enfin j'ai présenté quelques interfaces de notre application.

Au cours de ce rapport, j'ai appliqué mes connaissances et j'ai acquis des compétences relationnelles et techniques enrichissantes à notre formation. D'ailleurs, ce projet a été bénéfique à plusieurs niveaux.

Enfin je ne trouve de meilleur pour terminer mon rapport que de remercier Mon encadrand BENDAHMANE AHMED pour ses efforts.



Webography

- [1] Wikipedia. url: https://fr.wikipedia.org/wiki/JavaScript_Object_Notation (visited on 0006–2020).
- [2] developer.android. url: <https://developer.android.com/guide> (visited on 0006–2020).
- [3] spring.io. url: <https://spring.io/guides/gs/spring-boot/> (visited on 0006–2020).
- [4] wikipedia. url: https://fr.wikipedia.org/wiki/Android_Studio (visited on 0006–2020).
- [5] wikipedia. url: https://fr.wikipedia.org/wiki/Representational_state_transfer (visited on 0006–2020).
- [6] wikipedia. url: https://fr.wikipedia.org/wiki/Extensible_Markup_Language (visited on 0006–2020).
- [7] wikipedia. url: https://fr.wikipedia.org/wiki/Cas_d (visited on 0006–2020).
- [8] wikipedia. url: [https://fr.wikipedia.org/wiki/Java_\(langage\)](https://fr.wikipedia.org/wiki/Java_(langage)) (visited on 0006–2020).
- [9] wikipedia. url: <https://www.getpostman.com/downloads/> (visited on 0006–2020).
- [10] wikipedia. url: https://fr.wikipedia.org/wiki/Diagramme_de_classes (visited on 0006–2020).