

Projet de Fin d'Etudes

Licence Sciences et Techniques Génie Informatique

Département Informatique

Développement d'une API et sécurité pour une
Plateforme digital pour le secteur touristique



Lieu de stage : Nawra technologie

Réalisé par :

- Aymane Benahmed

Encadré par :

- Pr. Mohammed Talibi Alaoui
- Mr. Mohamed KAARAR

Soutenu le 04/07/2022 devant le jury composé de :

Pr. M.C. ABOUNAIMA

Pr. L. LAMRINI

Pr. Mohammed Talibi Alaoui



Dédicaces

Je tiens à dédier ce modeste travail à :

Mon père

L'homme le plus brave, droit, bon, courageux ainsi que toutes les autres qualités que l'on pourrait attribuer à un homme digne de ce nom, que je connaisse.

Ma mère

La femme la plus douce, sentimentale, responsable, dotée d'un esprit de gestion hors norme, et du meilleur esprit maternel que je connaisse.

Mes petites sœurs

Leur douceur, leur curiosité, leur intelligence et leur hyperactivité ont toujours été pour moi des raisons pour faire de ma personne un exemple pour elles.

Mes ami(e)s :

Hamza , amine, nouhaila, aya, redouan, karim, mohamed ainsi que toute personne qui a un jour croisée ma route et habitée mon cœur, Dieu m'a béni en vous mettant sur mon chemin.

A toutes personnes m'ayant attribué son amitié ou son amour, vous avez illuminé ma vie.

Et enfin à tout lecteur,



Remerciements

Avant tout remerciement, louange à Dieu.

Louange à Dieu tout puissant pour tout ce qu'il m'a donné afin que je puisse terminer ce travail. Au terme de ce celui-ci, je tiens à exprimer toute ma gratitude envers les personnes qui, de près ou de loin, m'ont apporté leur sollicitude.

J'adresse ma profonde reconnaissance à Monsieur Mohammed Talibi Alaoui pour ses précieuses directives, sa compréhension et l'incalculable aide qu'il m'a prodigué tout au long du projet.

Nous remercions aussi Mr. KAARAR Mohamed, le chef du projet pour son aide très précieuse, ses conseils et éclaircissements tout aussi pertinents qu'il m'a accordés durant mon stage.

Je remercie également, les autres membres de NAWRA TECHNOLOGY que j'ai côtoyé lors de ces deux derniers mois et qui ont tous, sans exception, contribué à faire de ce stage ce qu'il a été.

Mes remerciements s'adressent également et particulièrement à nos professeurs, enseignants, formateurs et tout le personnel de La FSTF pour leurs efforts incalculables consentis à notre profit, afin de nous assurer une formation de qualité, susceptible de nous qualifier à l'entrée au marché de l'emploi.

Finalement, toutes mes reconnaissances et mes considérations à ma famille, pour son extrême soutien et pour l'ambition qu'elle m'inspire.



Résumé

Le présent document est le fruit du travail réalisé dans le cadre du projet de fin d'études effectué au sein de la société NAWRA TECHNOLOGY. Ce projet a pour but l'étude et la mise en développement d'une API pour une plateforme web facile à utiliser dans le secteur touristique , (réservation d'hôtel réservation ticket de vol...,).

Ma mission dans ce projet c'est de développer l'api provisioning360 dont l'objectif est de gérer les comptes des personnes.

La réalisation de ce projet a été basée sur des langages de programmation tels que Spring web, Spring data pour développement d'api et Spring sécurité, keyclock pour la sécurité.



Abstract

This document is the result of the work carried out within the framework of the end- of-studies project carried out within the company NAWRA TECHNOLOGY. The purpose of this project is to study and implement Develop an API for a digital platform For the tourism sector and API security, with the aim of facilitating the tourist's process of coming to visit our country all that is hotel reservation booking flight ticket....,

My mission in this project is to develop the API provisioning360api its mission is to manage people's accounts

The realization of this project was based on programming languages such as Spring web, spring data for API development and spring security, keycloak for security



Sommaire

Dédicaces.....	2
Remerciements	3
Résumé	4
Liste Figures	9
LISTE DES ACRONYMES	11
INTRODICTION.....	12
Chapitre 1 Contexte générale du Projet	13
Introduction	14
I- Présentation de l'organisme d'accueil.....	14
II- Présentation du projet.....	15
1. Atlas Project.....	15
2. Problématique.....	15
3. Les outils de travail.....	16
3.1. L'outil REDOC.....	16
3.2. L'outil Trello	17
3.3. GITLAB.....	18
4. Missions et tâches effectués	19
Conclusion.....	19
Chapitre 2 Analyse et conception	20
Introduction.....	21
1. Analyse des besoins.....	21
1.1. Les besoins attendus Api provisioning 360	21
1.2. Les besoins fonctionnels.....	21
1.3. Les besoins non fonctionnels	22



1.4.	les besoins techniques.....	22
1.4.1.	L'architecture microservice	22
1.4.2.	API REST	22
1.5.	Outils de développement :	24
2.	Conception adoptée	28
2.1.	Diagramme de cas d'utilisation	28
2.2.	Diagramme de séquence.....	30
1.3.1.	Diagramme de séquence générale.....	31
1.3.2.	Diagramme de séquence ajouter un Partner.....	32
1.3.3.	Diagramme de séquence ajouter un Person	33
1.3.4.	Diagramme de séquence ajouter un compte	34
1.4.	Diagramme de classe	35
	Conclusion.....	35
Chapitre 3 Interfaces de l'application.....		36
1.	Introduction.....	37
1.1.	postman.....	37
1.2.	Keyckloack	38
1.2.1.	Console d'administration Keycloak	38
1.2.2.	Récupération du token	39
2.	Ajouter une Partner.....	40
2.1.	Status code : 201 created	40
2.2.	Status : 400 Bad Request	40
2.3.	Status : 500 Internal Server Erreur.....	41
3.	Ajouter une Person.....	41
3.1.	Status code : 201 created	41
3.1.1.	le Person a un Partner	41
3.1.2.	Le Person n'a pas un Partner.....	42
3.2.	Status : 404 Not Found	42
3.3.	Status : 500 Internal Server Erreur.....	43
4.	Ajouter un compte	43
4.1.	Status code : 201 created	44



4.2. Status : 409 User déjà existe	45
5. Consulter un compte	45
5.1. Status code : 200 OK	45
5.2. Status :404 Not Found	46
6. Modifier un compte.....	47
6.1. Status code : 201 created	47
6.2. Status :404 Not Found	47
7. Supprimer un compte	49
8. Envoie des emails.....	49
8.1. Email de confirmation	49
8.2. Mot de passe oublier	51
CONCLUSION ET PERSPECTIVES	53
RÉFÉRENCES.....	54



Liste Figures

Figure 1 : logo de l'entreprise.....	14
Figure 2 : schéma de générale.....	15
Figure 3 : schéma du projet détaillé.....	16
Figure 3 : documentation de l'Api.....	16
Figure 4 : l'Outils de Trello.....	17
Figure 5 : Diagramme de Gantt.....	19
Figure 6 : Description API REST	24
Figure 7 : Diagramme de cas d'utilisation générale	28
Figure 8 : Diagramme de cas d'utilisation d'api Provisioning 360.....	29
Figure 9 : Diagramme de séquence générale.....	31
Figure 10: Diagramme de cas d'utilisation ajouter un Partner	32
Figure 11: Diagramme de cas d'utilisation ajouter un Person.....	33
Figure 12: Diagramme de cas d'utilisation ajouter un compte	34
Figure 13: .Diagramme de classe	35
Figure 14: .interface de Postman.....	37
Figure 15: .interface de keycloak	38
Figure 16: .récupérer le token à travers keycloak	39
Figure 17: décoder l'Access token	39
Figure 18: ajouter un Partner status 201	40
Figure 19: ajouter un Partner status 404.....	40
Figure 20: ajouter un Partner status 500.....	41



Figure 21: ajouter un Person lié un Partner status 201	41
Figure 22: ajouter un Person n'est pas lié un Partner status 201.....	42
Figure 23: ajouter un Person status 404.....	43
Figure 24: ajouter un Person status 500	43
Figure 25: ajouter un compte status 201	44
Figure 26: ajouter un compte status 409.....	45
Figure 27: consulter un compte status 201	45
Figure 28: consulter un compte status 404 Person not found	46
Figure 28: consulter un compte status 404 user not found	46
Figure 30: modifier un compte status 201	47
Figure 31: modifier un compte status 404 Person not found	48
Figure 31: modifier un compte status 404 user not found	48
Figure 33: Supprimer un compte status 501	49
Figure 34: interface ajouter un compte.....	50
Figure 35: interface demande de verification par email	50
Figure 36: email de vérification.....	51
Figure 37: Interface mot de passe oublié.....	51
Figure 38: email de réinstaller le mot de passe	52
Figure 39: interface pour réinstaller le mot de passe.....	52



LISTE DES ACRONYMES

Acronyme	Désignation
API	Application programming interface
HTTP	HyperText Transfer Protocol
REST	Représentationnel StateTransfer
JSON	JavaScript Object Notation
SGBD	Système de Gestion des Bases de Données
UML	Unified Modeling Language
SAT	service account token
UAT	user access token



INTRODUCTION

Dans le cadre de ma troisième année de licence à la Faculté des sciences et techniques FES, génie informatique, j'ai eu l'opportunité de réaliser un stage de 8 semaines au sein du groupe "NAWRA THECHNOLOGIE" plus précisément le développement de l'api provisioning360, durant lequel j'ai réalisé un api qui permet de gérer les comptes des utilisateurs et incidents développée en SPRING BOOT

Le premier chapitre présente le cadre général du projet en décrivant l'organisme d'accueil, la problématique du sujet, et le pilotage du projet.

Le deuxième chapitre se concentre sur l'analyse des besoins fonctionnels et techniques du projet, la conception et les différents diagrammes utilisés.

Le troisième chapitre présente les différentes interfaces de l'application. Et nous terminerons ce rapport par une conclusion générale et les perspectives du travail.



Chapitre 1

Contexte
Générale du
Projet

INTRODUCTION :

Dans ce chapitre je présente le cadre général du projet en décrivant l'organisme d'accueil, la problématique du sujet, les outils de travail et finalement les missions et les tâches effectués.

Figure 1 : logo de l'entreprise



I- Présentation de l'organisme d'accueil

Nawra Technologie est une société marocaine, basée à Fès, créée par

Des jeunes marocains intéressés par le développement des nouvelles solutions Informatique dans le secteur touristique au Royaume du Maroc.

La mission de NAWRA TECHNOLOGY est de participer à informatiser le secteur

Touristique au Maroc afin de faciliter l'accès des touristes à des plateformes web faciles à utiliser.

Le but est de faciliter aux touristes de trouver les auberges et les hôtels à réserver, les locations à louer et des excursions et des voyages organisés par des établissements spécialisés dans le domaine.

II. Présentation du projet

1. Atlas Project

Atlas Project consiste à développer une plateforme digitale touristique qui offre des services divers et variés à ces clients. La plateforme regroupe plusieurs types d'activités (hébergement, voyage, événements...)

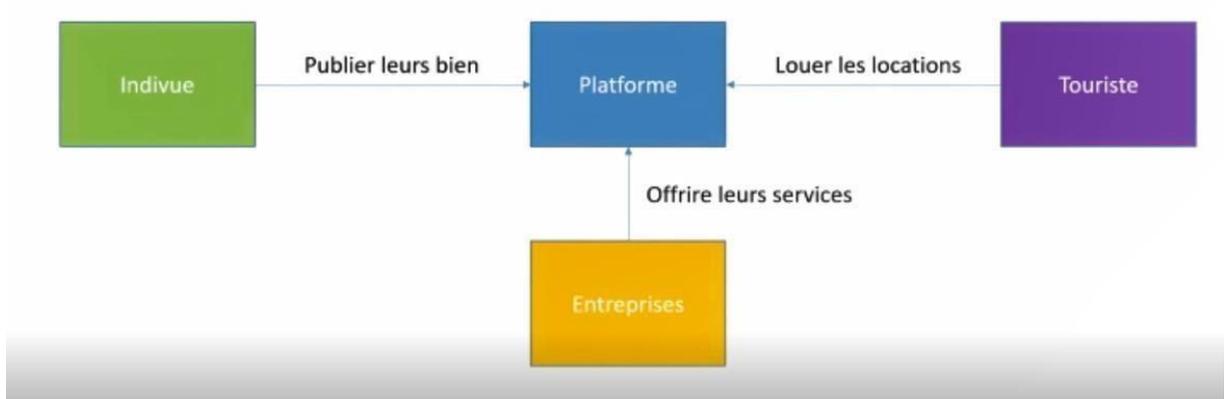


Figure 2 : schéma générale

2. Problématique

L'architecture de l'application API provisioning360 se repose sur plusieurs Micro-services (Person, Partenaire et communication) ainsi que l'enregistrement et la gestion des comptes.

En plus il était nécessaire de développer une couche de sécurité pour toutes ces applications.

La complexité du projet (plusieurs micro-services qui interagissent entre eux...) rend

l'objectif est l'optimisation du temps de traitement des incidents et aussi d'éviter la surcharge sur l'assistance en ligne. Ainsi, cette application améliorera la production et la qualité du service desk et la sécurité de leur information, et elle assurera une satisfaction et fidélisation de la clientèle.

Cet api à concevoir et à développer vient pour répondre à ce besoin majeur tout en assurant une intégration des fonctionnalités nécessaires dans l'application.

Afin de bien cerner les objectifs et le contexte du projet à réaliser, il s'impose d'effectuer une étude de l'existant produisant ainsi un ensemble de critiques et des solutions à entreprendre.

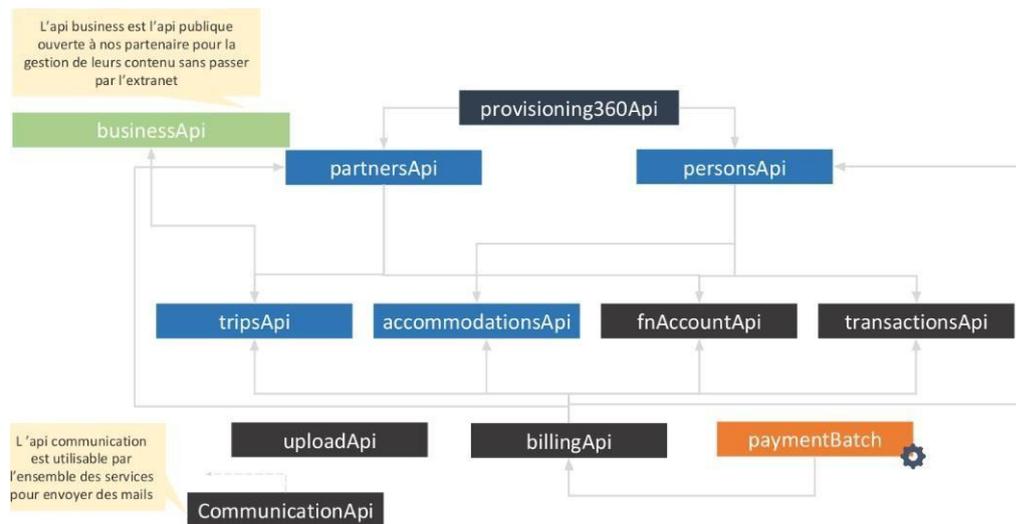


Figure 2 : schéma du projet détaillé

3. Les outils de travail

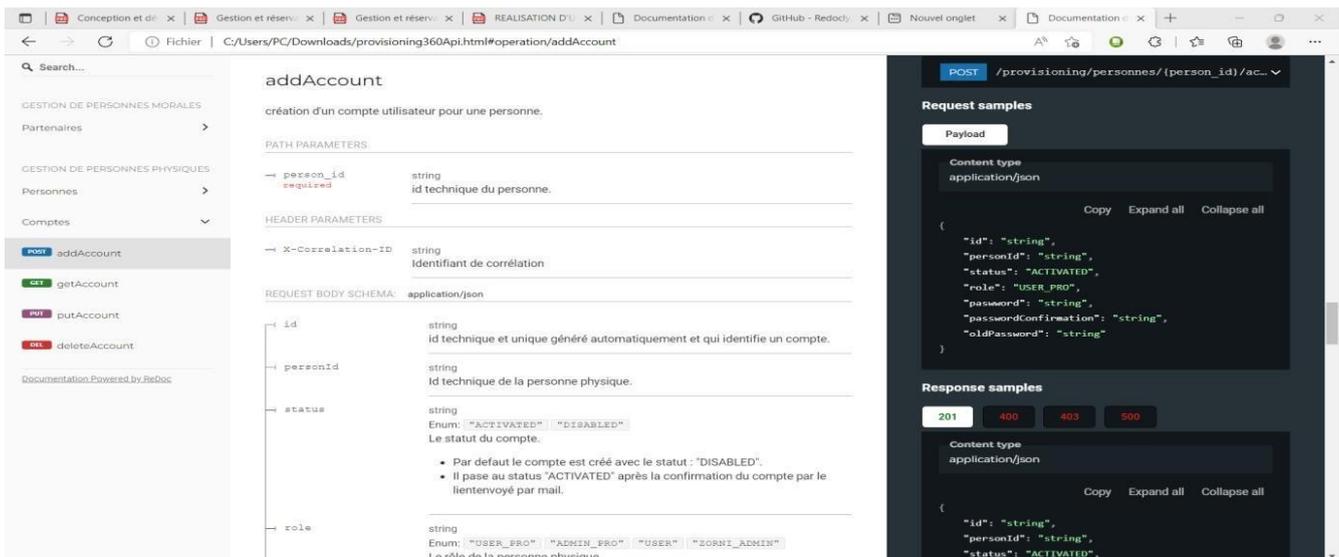
3.1. L'outil REDOC

Redoc est un outil open source permettant de générer de la documentation à partir des définitions Open API (fka Swagger).

Par défaut, Redoc propose une mise en page réactive à trois panneaux :

Le panneau de gauche contient une barre de recherche et un menu de navigation. Le panneau central contient la documentation.

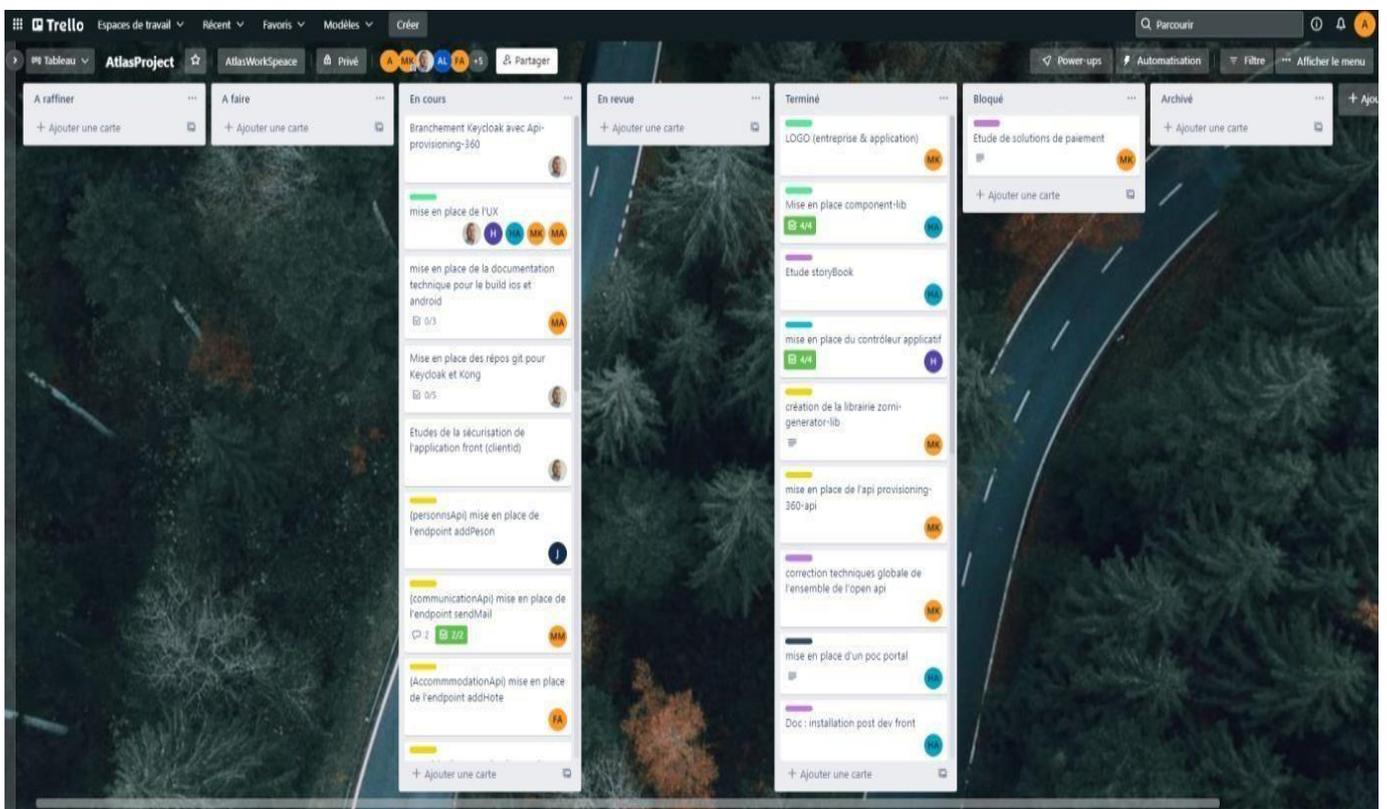
Le panneau de droite contient des exemples de requête et de réponse.



*Figure 3 : documentation de l'API

3.2. L’outil Trello

Afin de mettre en place cette méthodologie, l’outil de gestion de projet en ligne Trello était proposé par notre encadrant M. KAARAR Mohamed, afin d’assurer la coordination entre les différents acteurs contribuant dans le projet dans un souci de meilleure efficacité et de rentabilité. L’outil Trello est basé sur une organisation des projets en planches listant des cartes, représentant les tâches à réaliser. Ces cartes sont mobiles d’une planche à l’autre, première planche est la planche user stories qui contient l’ensemble des tâches à réaliser au long du projet, la deuxième planche est la planche Sprint. La méthode Scrum s’appuie sur le découpage du projet en boîtes de temps, nommées « sprints ». Les sprints peuvent durer entre une semaine et un mois (avec une préférence pour deux semaines). Chaque sprint commence par une estimation suivi d’une planification opérationnelle.



-Figure 4 : l’Outils de Trello

3.3. GITLAB

GITLAB est un service web permettant l'hébergement des développements de logiciel, utilisant Git. Git est un gestionnaire de version de fichier qui permet d'enregistrer toutes les modifications sur un fichier, permettant de revenir en arrière en cas d'erreur sur celui-ci. L'avantage de ce service est de pouvoir travailler de manière collaborative distribuée. En effet chacun peut travailler sur ses tâches en parallèle et ensuite l'envoyer sur la plateforme. Le second avantage, GitLab favorise le travail à distance car les fichiers sont hébergés sur le serveur et accessible gratuitement aux développeurs.

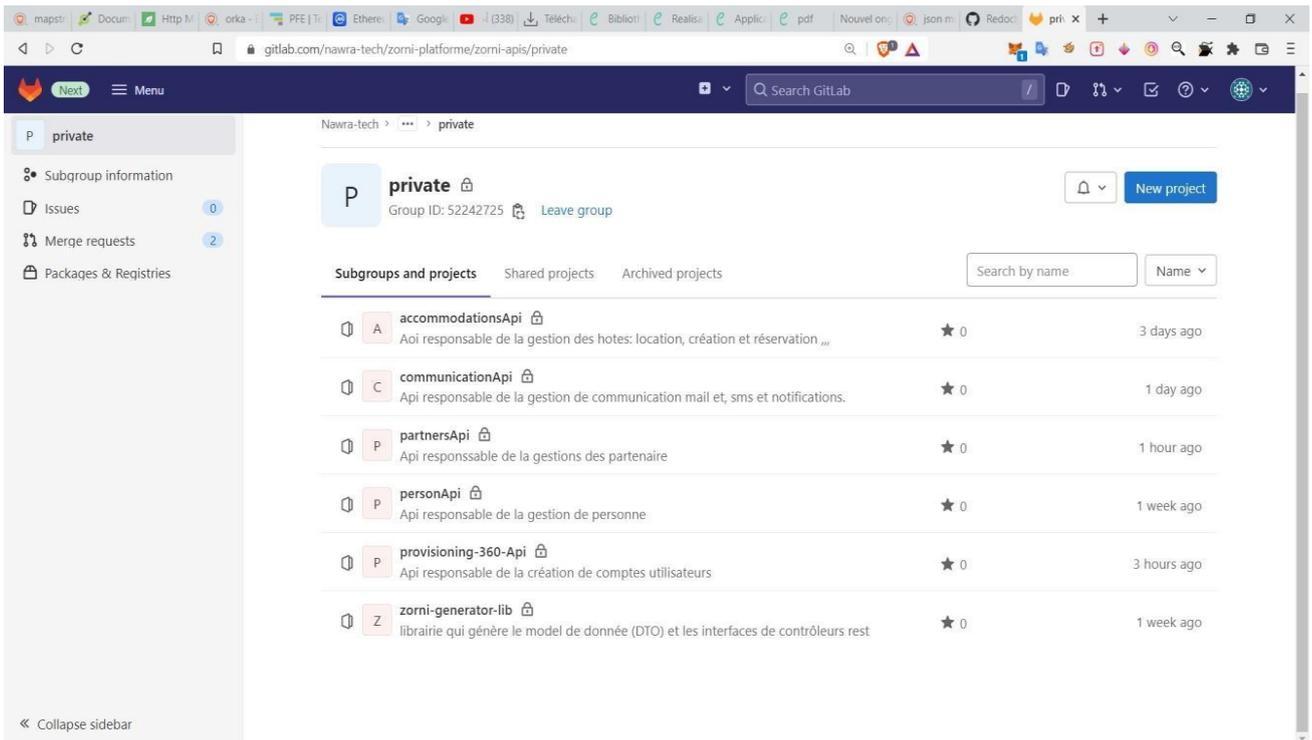


Figure 5 : GITLAB



Chapitre 2

Analyse

Et

Conception



INTRODUCTION :

Ce deuxième chapitre se concentre sur l'analyse des besoins fonctionnels, non fonctionnels et techniques du projet, la conception et les différents diagrammes utilisés.

1. Analyse des besoins

1.1. Les besoins attendus Api provisioning 360

Il est appelé par ce nom car il a une vision 360, il fait une relation entre les autres Api (Person, Partner, Commination) et au même temps avec sécurité keycloak et il doit satisfaire les besoins fonctionnels qui seront exécutés par le système et les besoins non fonctionnels qui perfectionnent la qualité logicielle du système.

1.2. Les besoins fonctionnels

Les besoins fonctionnels ou besoin métiers représentent les actions que le système doit exécuter, il ne devient opérationnel que s'il les satisfait :

- Gestion des Partner : création des Partner partir de appel d'API Partner
- Gestion des Person : création des Person à partir d'appel d'API Person
- Gestion des comptes : création des comptes, afficher un compte, modifier un compte, supprimer un compte
- Sécuriser l'api
- Envois d'email

1.3. Les besoins non fonctionnels

Ce sont des exigences qui ne concernent pas spécifiquement le comportement du système, mais plutôt identifient des contraintes internes et externes du système. Les principaux besoins non fonctionnels de notre application se résument dans les points suivants :

- L'organisation du code: Le code doit être clair et respecte la même architecture pour tous les modules.
- La fragmentation du code: Pour des raisons de lisibilité, il est nécessaire de découper un programme en plusieurs modules et fichiers, afin de faciliter la réutilisation d'une partie du code pour d'autres applications.

Les objectifs attendus et les besoins à réaliser vont être projetés dans des diagrammes de séquence.



1.4. les besoins techniques

1.4.1. L'architecture microservice

Les microservices désignent à la fois une architecture et une approche de développement logiciel qui consiste à décomposer les applications en éléments les plus simples, indépendants les uns des autres. Contrairement à une approche monolithique classique, selon laquelle tous les composants forment une entité indissociable, les microservices fonctionnent en synergie pour accomplir les mêmes tâches, tout en étant séparés.

Les avantages des microservices :

- Mise sur le marché plus rapide

Comme les cycles de développement sont plus courts, l'architecture de microservices permet des déploiements et mises à jour plus agiles ,

- Haute évolutivité

À mesure que la demande pour certains services augmente, vous pouvez étendre les déploiements sur plusieurs serveurs et infrastructures pour répondre à vos besoins.

- Résilience

Lorsqu'ils sont développés correctement, ces services indépendants n'ont aucun impact les uns sur les autres. Cela signifie que, lorsqu'un élément tombe en panne, l'ensemble de l'application ne cesse pas de fonctionner comme c'est le cas avec le modèle monolithique.

- Accessibilité

Vu que l'application est décomposée en plusieurs éléments, les développeurs peuvent plus facilement comprendre, mettre à jour et améliorer chacun de ces éléments. Résultat: des cycles de développement plus courts, surtout s'ils sont associés à des méthodes de développement agiles.

1.4.2. API REST

API : est un ensemble de définitions et de protocoles qui facilite la création et l'intégration de logiciels d'applications, permettent aux produit ou service de communiquer avec d'autres produits et services sans connaître les détails de leur mise en œuvre. Elles simplifient le développement d'applications et vous font ainsi gagner du temps et de l'argent. Lorsque vous concevez de nouveaux outils et produits, ou que vous assurez la gestion de ceux qui existent déjà, les API vous offrent plus de flexibilité, simplifient la conception, l'administration et l'utilisation, et vous donnent les moyens d'innover.

Exemple d'une API :



API Google Maps : L'API Google Maps permet aux développeurs d'intégrer Google Maps sur des pages web à l'aide d'une interface JavaScript ou Flash. L'API Google Maps est conçue pour fonctionner sur les appareils mobiles et les navigateurs de bureau.

API REST : est un type d'API, qui aide les applications de services web à communiquer entre elles. Bien qu'elle soit théoriquement compatible avec n'importe quel protocole ou format de données, l'architecture REST utilise le plus souvent le protocole HTTP pour l'envoi de requêtes relatives au CRUD (création, récupération, modification ou suppression).

L'interaction avec les ressources basées sur un serveur s'effectue à partir d'un URI. Et transfère les données en utilisant JSON. L'architecture REST est généralement retenue pour obtenir des données à partir du web en raison de sa flexibilité, de sa rapidité et de sa simplicité.

Le langage de programmation le plus communément utilisé est JSON, car, contrairement à ce que son nom indique, il ne dépend pas d'un langage et peut être lu aussi bien par les humains que par les machines.

Les 4 opérations possibles (CRUD) du protocole HTTP propose les verbes correspondant :

- Créer : POST
- Afficher : GET
- Mettre à jour : PUT
- Supprimer : DELETE

L'URI est une courte chaîne de caractères identifiant une ressource peuvent contenir jusqu'à cinq parties, parmi lesquelles seules deux sont obligatoires :

- Schème (le schéma) : indique le protocole utilisé.
- Autorité (l'autorité) : identifie le domaine.
- Path (le chemin) : indique le chemin d'accès à la ressource.
- Query (la requête) : représente une action de requête.
- Fragment (le fragment) : désigne un aspect partiel d'une ressource.

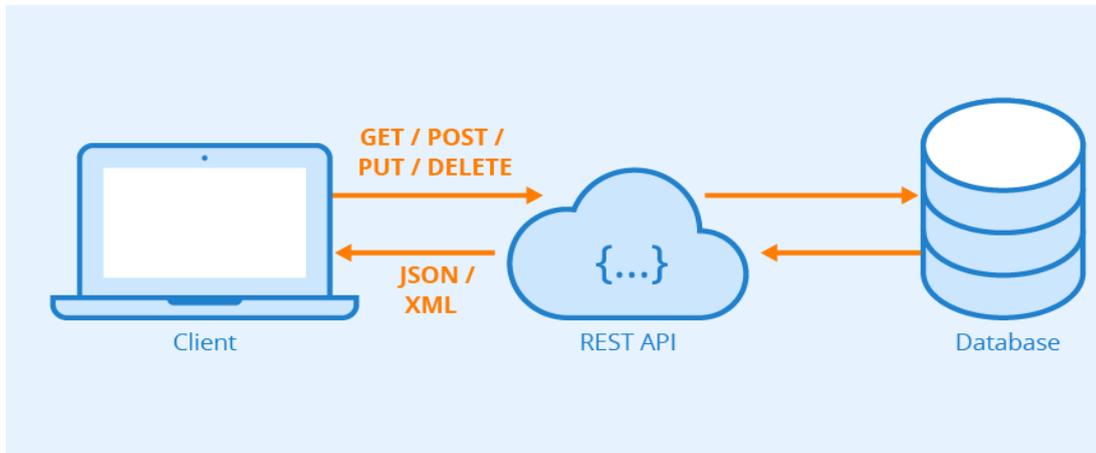


Figure 6 : Description API REST

1.5. Outils de développement :

langage

Java

Java est un langage de programmation de haut niveau, basé sur des classes et orienté objet, conçu pour avoir le moins de dépendances d'implémentation possible.



Framework	
<p>Spring :</p> <p>Spring est un Framework open source basé sur Java utilisé pour créer un micro service. Il est développé par Pivotal Team et est utilisé pour créer des applications de printemps autonomes et prêtes pour la production.</p>	
Starter	
<p>Spring Data :</p> <p>Il permet de fournir un modèle de programmation familier et cohérent basé sur Spring pour l'accès aux données. Il facilite l'utilisation des technologies d'accès aux données, des bases de données relationnelles et non relationnelles.</p>	
<p>Spring Web :</p> <p>Spring Web est le Framework Web original construit sur l'API Servlet et a été inclus dans Spring Framework depuis le tout début.</p>	

Sécurité	
<p>Spring Security :</p> <p>Spring Security est un cadre qui se concentre sur la fourniture de mécanismes d'authentification et d'autorisation aux applications Spring</p>	
<p>Keycloak :</p> <p>Keycloak est une solution open source de gestion des identités et des accès administrée par RedHat et développée en Java par JBoss.</p>	



Logiciel	
<p>Eclipse :</p> <p>Est un environnement de production de logiciels libre.</p>	
<p>Entreprise Architect :</p> <p>Est un logiciel de modélisation et de conception UML.</p>	
<p>Postman :</p> <p>Postman est une plateforme API permettant aux développeurs de concevoir, créer, tester et itérer leurs API.</p>	
<p>Postgresql :</p> <p>Est un système de gestion de bases de données relationnelles.</p>	
<p>Mail Hog:</p> <p>Mail Hog is an email testing tool for developers:</p>	

2. Conception adoptée

2.1. Diagramme de cas d'utilisation

Ce diagramme est un diagramme générale qui montre comment l'acteur réagit avec l'api

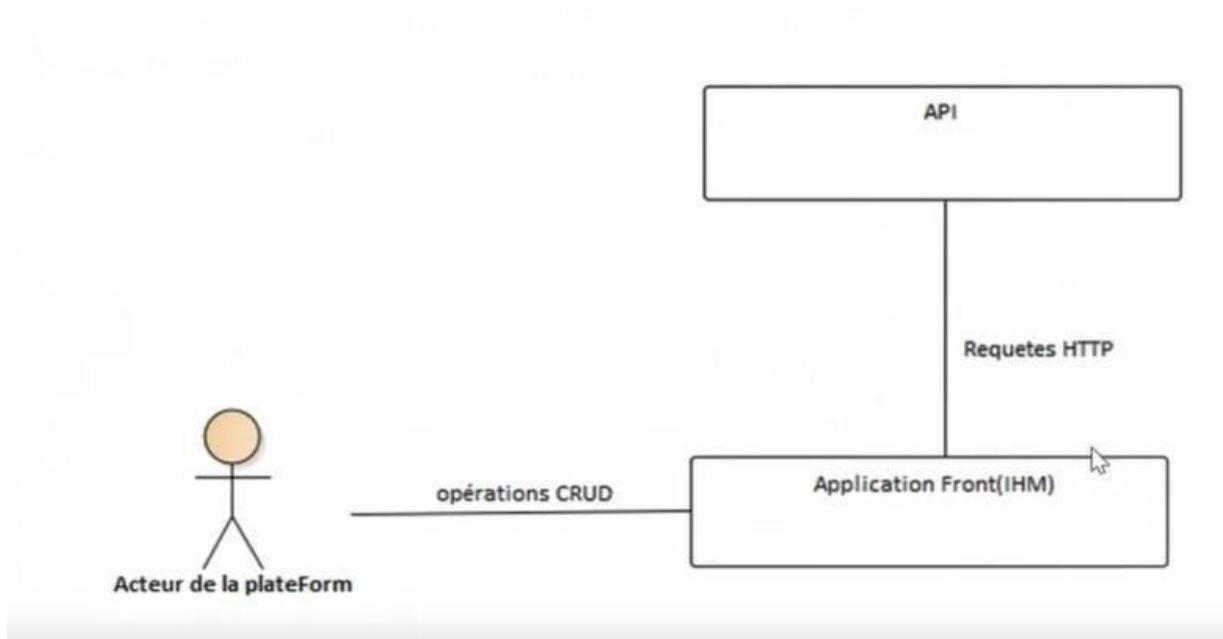


Figure 7 : Diagramme de cas d'utilisation générale

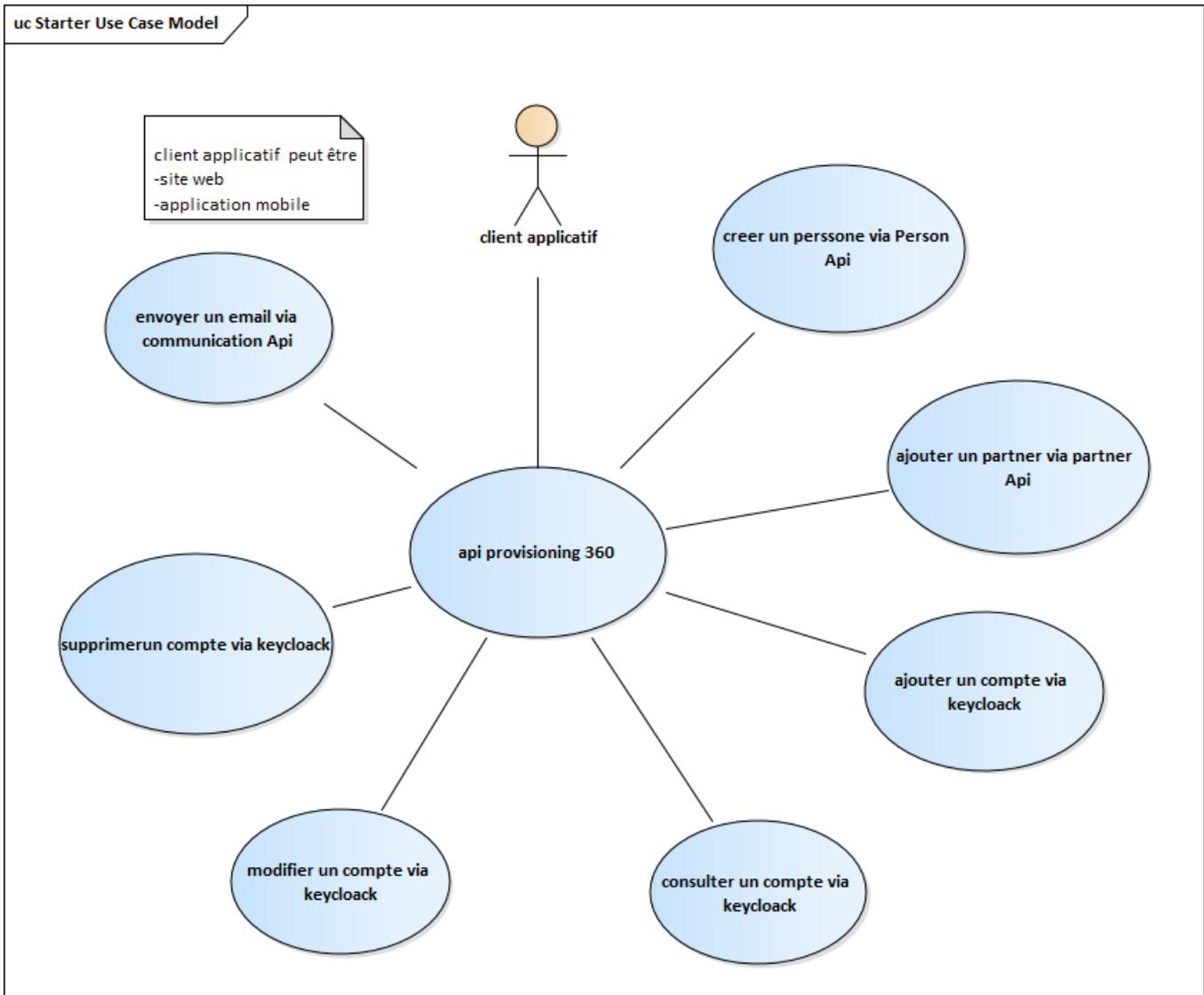


Figure 8 : Diagramme de cas d'utilisation d'api Provisioning 360

Provisioning360 doit permet une application front qui a la possibilité de faire les cas d'utilisation :

- ajouter un Partner à travers de Partner API
- Ajouter un Person à travers de Person API
- Les opérations CRUD à travers keycloak
- envoyer des emails à travers Communication Api



2.2. Diagramme de séquence

2.2.1. Diagramme de séquence générale

Nous avons réalisé des diagrammes de séquence pour décrire les cas d'utilisation les plus importants :

Ce **Diagramme de séquence** décrit comment l'utilisateur peut créer un compte

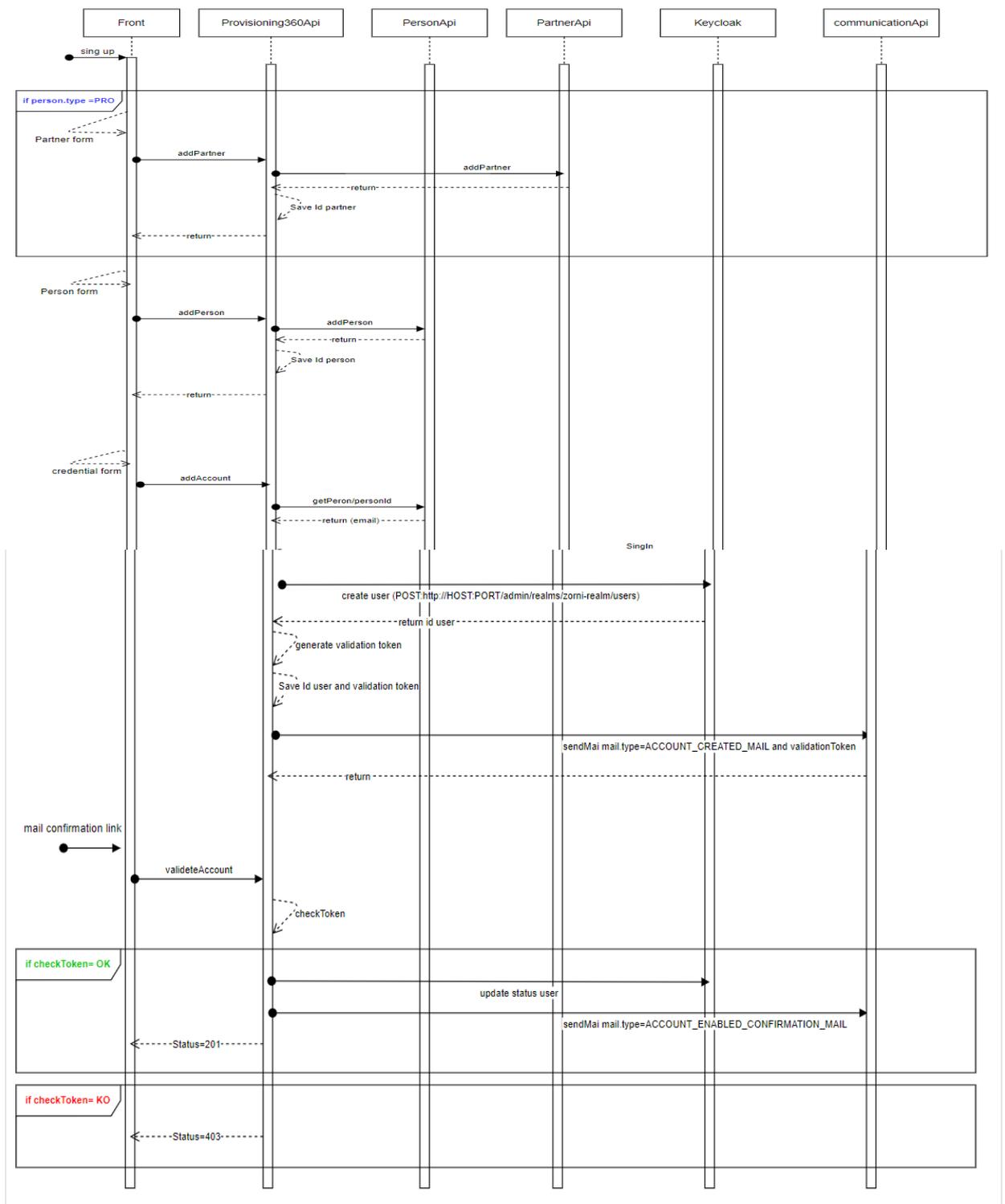


Figure 9 : Diagramme de séquence générale

2.2.2. Diagramme de séquence ajouter un Partner

Ce diagramme de séquence décrit comment ajouter un Partner et comment se fait la communication avec l'api Partner.

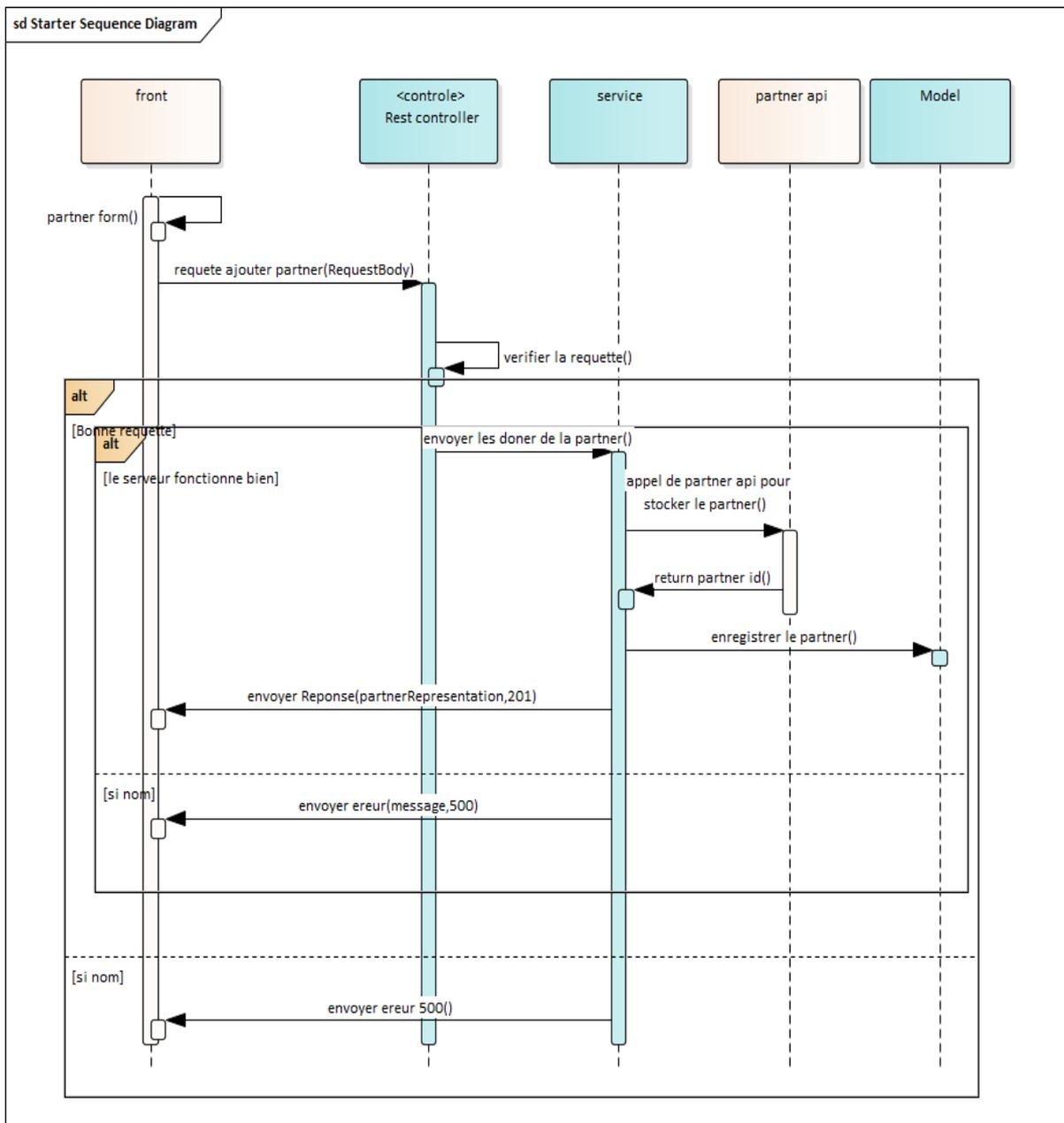


Figure 10: Diagramme de cas d'utilisation ajouter un Partner

2.2.5. Diagramme de classe

Le diagramme de classe a 3 classes principales :

- un Person : un Person ne peut gérer les autres classes seulement si un statut active (“ACTIVITED”).
- chaque Personne a le droit de créer seulement un compte
- un Person possède un Partner si de type Pro
- un Partner peut avoir plusieurs Person

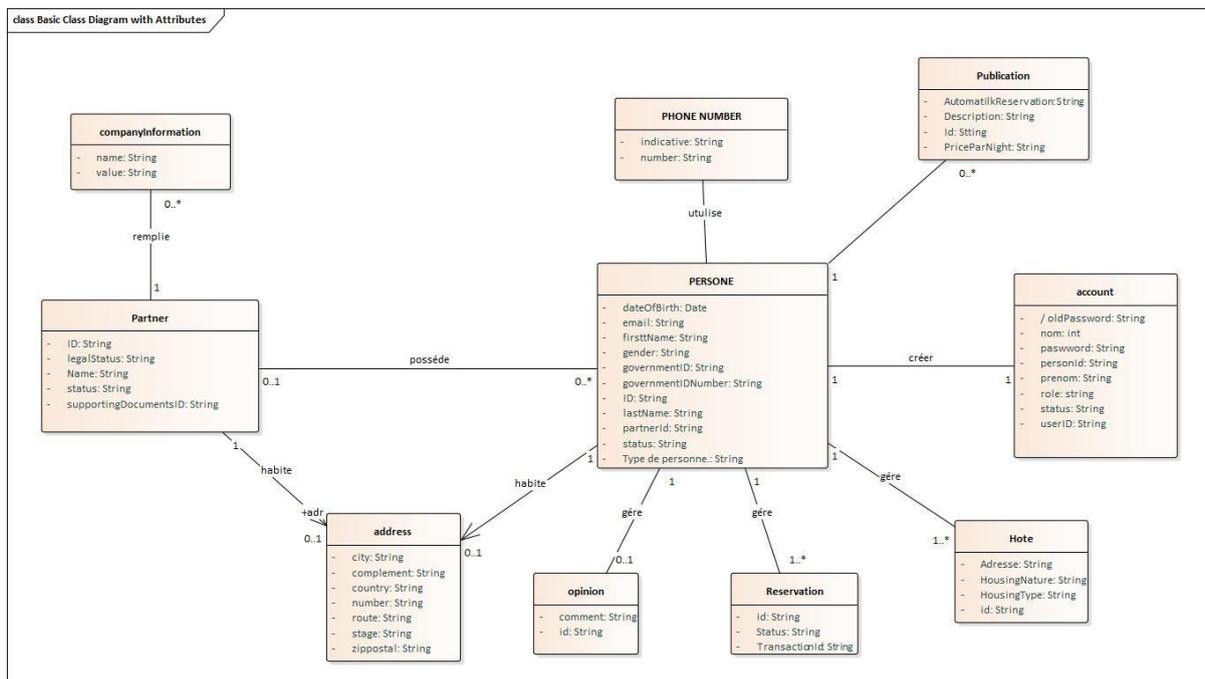


Figure 13: .Diagramme de classe

CONCLUSION :

Dans ce chapitre on a vu l'analyse des besoins fonctionnels, non fonctionnels et techniques du projet, la conception et les différents diagrammes utilisés. Dans le chapitre suivant on va voir les Interfaces de l'application.



Chapitre 3

Interfaces

De

L'application

1. Introduction

Ce chapitre présente les différentes interfaces de l'application. Nous terminerons ce rapport par une conclusion générale et les perspectives d'avenir.

Dans mon projet je me suis intéressé à la partie Back end, pour cela on utilise l'application Postman pour tester l'API et les données seront envoyées de type Json.

1.1. postman

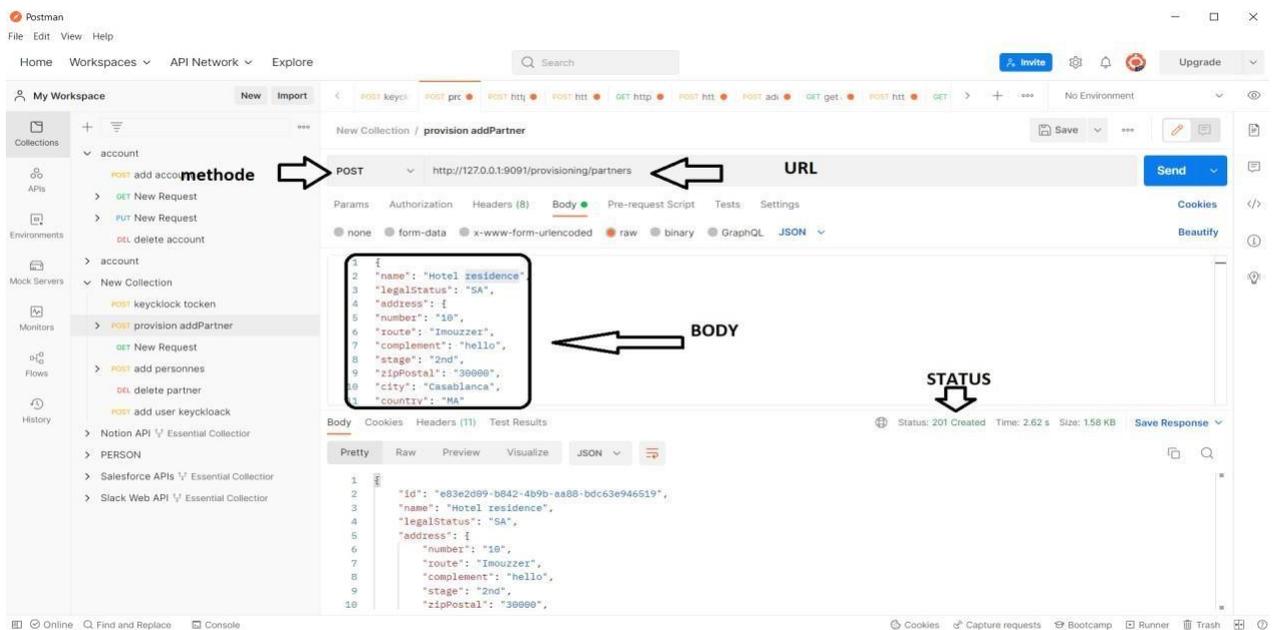


Figure 14: interface de Postman

Postman vous permet de créer et d'envoyer des requêtes API. Envoyez une demande à un point de terminaison, récupérez des données à partir d'une source de données ou testez la fonctionnalité d'une API. Vous n'avez pas besoin d'entrer des commandes dans un terminal ou d'écrire du code. Créez une nouvelle demande et sélectionnez Envoyer, et la réponse de l'API apparaît directement dans Postman.

Une requête inclut l'URL du point de terminaison de l'API et une méthode de requête HTTP. La méthode indique l'action que vous souhaitez que l'API effectue. Voici quelques-unes des méthodes les plus courantes :

- GET récupère les données d'une API.
- POST envoie de nouvelles données à une API.
- PUT mettre à jour les données existantes.
- DELETE supprime les données existantes.

1.2. Keycloak

1.2.1. Console d'administration Keycloak

C'est pour les administrateurs et les développeurs qui souhaitent configurer et gérer Keycloak.

Expliquons la section Configurer :

Royaume: Vous pouvez considérer le royaume comme un locataire. La première chose à faire est donc de créer un royaume pour l'application et les utilisateurs. Un royaume est totalement isolé (en termes de configuration, d'utilisateurs, de rôles, etc.) des autres royaumes.

Client : les clients sont des entités qui peuvent demander à Keycloak d'authentifier un utilisateur. Le plus souvent, les clients sont des applications Web, mobiles et natives qui souhaitent utiliser Keycloak pour se sécuriser et fournir une solution d'authentification unique. Les clients peuvent également être n'importe quel type de service tel que les API REST, gRPC, WebSocket qui veulent simplement demander des informations d'identité ou un token d'accès afin qu'ils puissent invoquer en toute sécurité d'autres services sur le réseau qui sont sécurisés par Keycloak.

The screenshot shows the Keycloak administration interface. On the left is a navigation sidebar with options like 'Configure', 'Clients', 'Client Scopes', 'Roles', 'Identity Providers', 'User Federation', 'Authentication', and 'Manage'. The main area is titled 'Clients' and contains a table of client configurations. A search bar and a 'Create' button are at the top right of the table. A black arrow points to the 'provisioning-360-api' client in the table.

Client ID	Enabled	Base URL	Actions		
account	True	http://127.0.0.1:8080/auth/realms/zorni-realm/account/	Edit	Export	Delete
account-console	True	http://127.0.0.1:8080/auth/realms/zorni-realm/account/	Edit	Export	Delete
admin-cli	True	Not defined	Edit	Export	Delete
broker	True	Not defined	Edit	Export	Delete
provisioning-360-api	True	Not defined	Edit	Export	Delete
realm-management	True	Not defined	Edit	Export	Delete
security-admin-console	True	http://127.0.0.1:8080/auth/admin/zorni-realm/console/	Edit	Export	Delete

Figure 15: .interface de keycloak

1.2.2. Récupération du token

Pour avoir le token il faut tout d'abord avoir le client id (c'est l'application quand veut la sécuriser) deuxièmes il faut avoir le client secret (auto génère par keycloak)

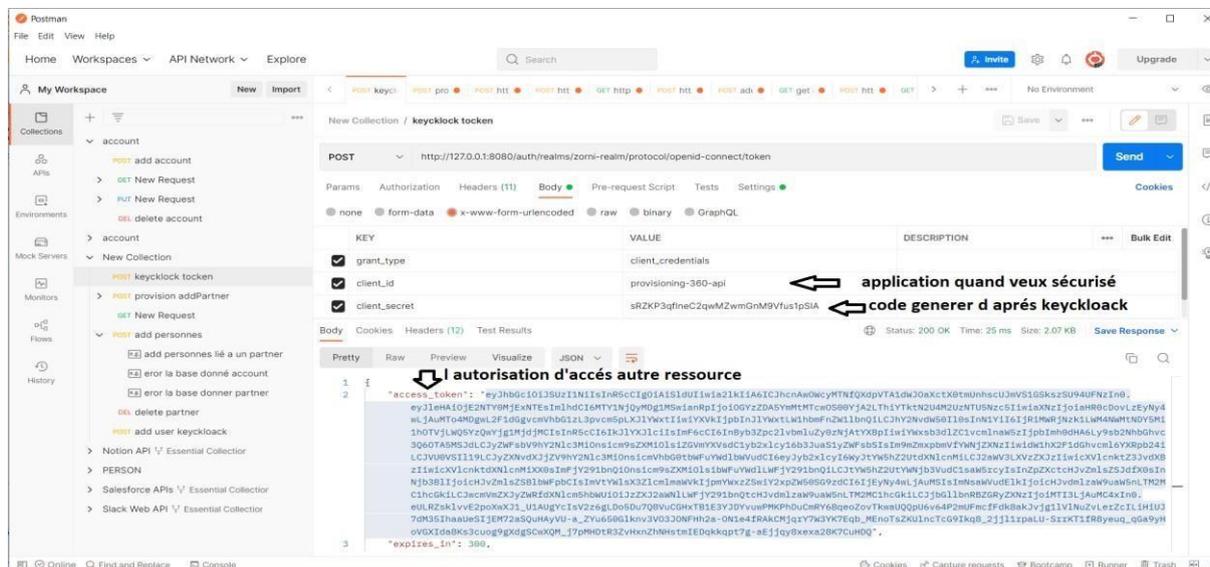


Figure 16: récupérer le token à travers keycloak

On peut décoder l'Access token à travers jwt.io. Va trouver plusieurs informations.

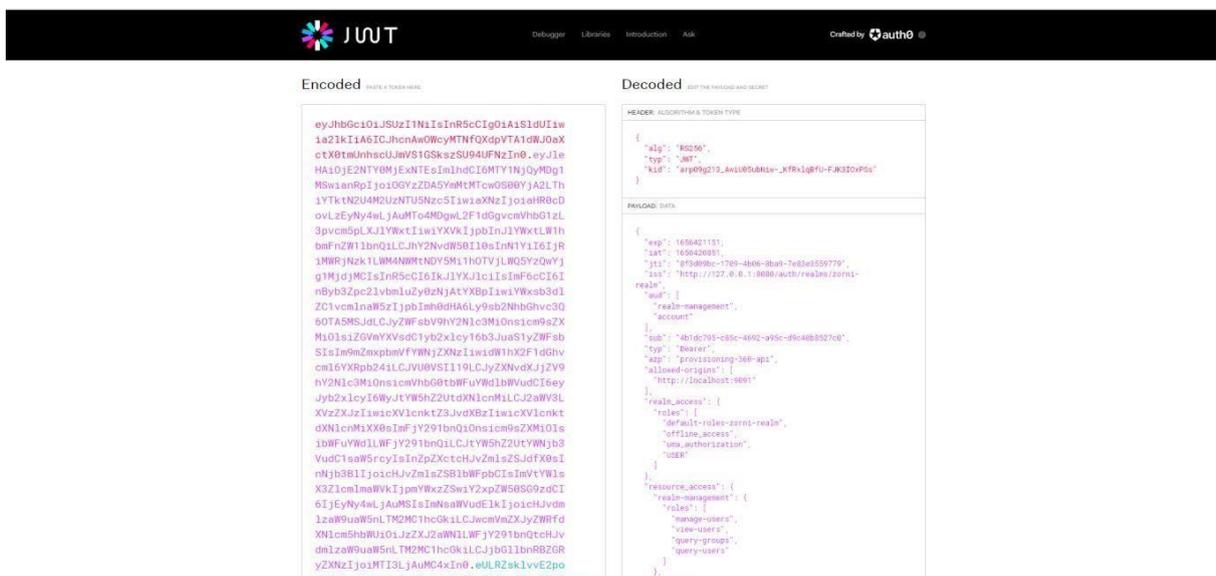


Figure 17: décoder l'Access token

Parmi l'information c'est :

- Algorithme décodage
- la date de génération de token

2. Ajouter une Partner

2.1. Status code : 201 created

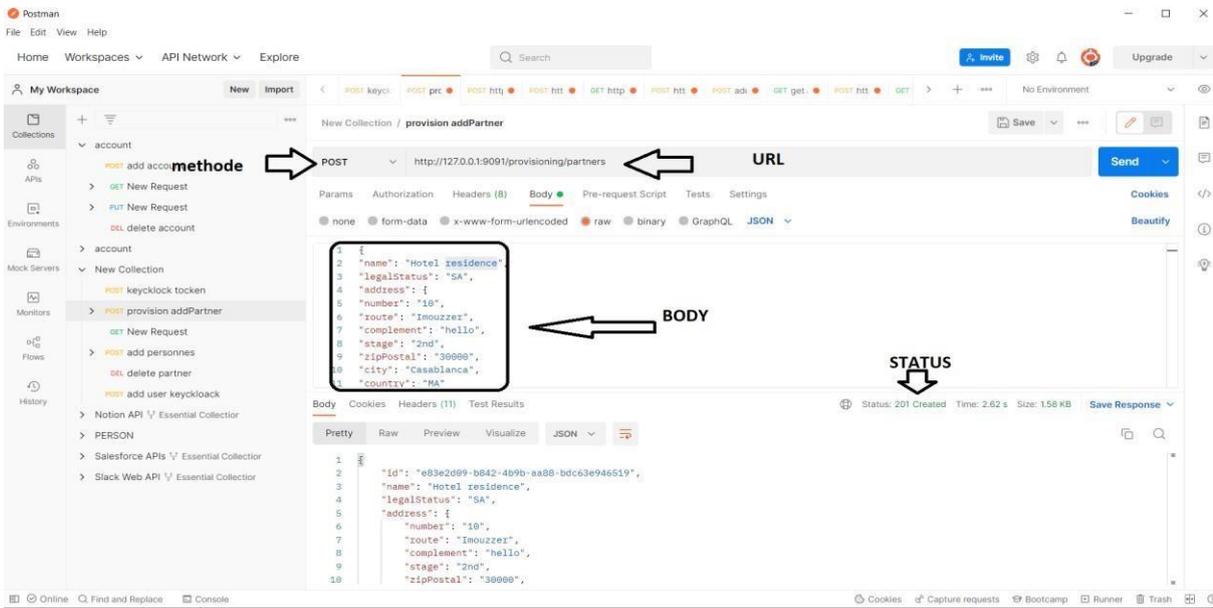


Figure 18: ajouter un Partner status 201

La réponse envoyé : on le code statut de réponse http 201 created alors que l'api provisioning360 à envoyer les données à l'api Partner et le Partner est bien enregistrée. L'api a généré une id automatique et ajouter des liens. Ces liens nous permettent de modifier, consulter et supprimer cette personne, et à la fin l'API provisioning enregistre l id de Partner.

2.2. Status : 400 Bad Request

Ce type de erreur est déclencher par postman si on en fait une erreur dans le body comme Si on ajouter ou on oublier une virgule (,) une double quote (")

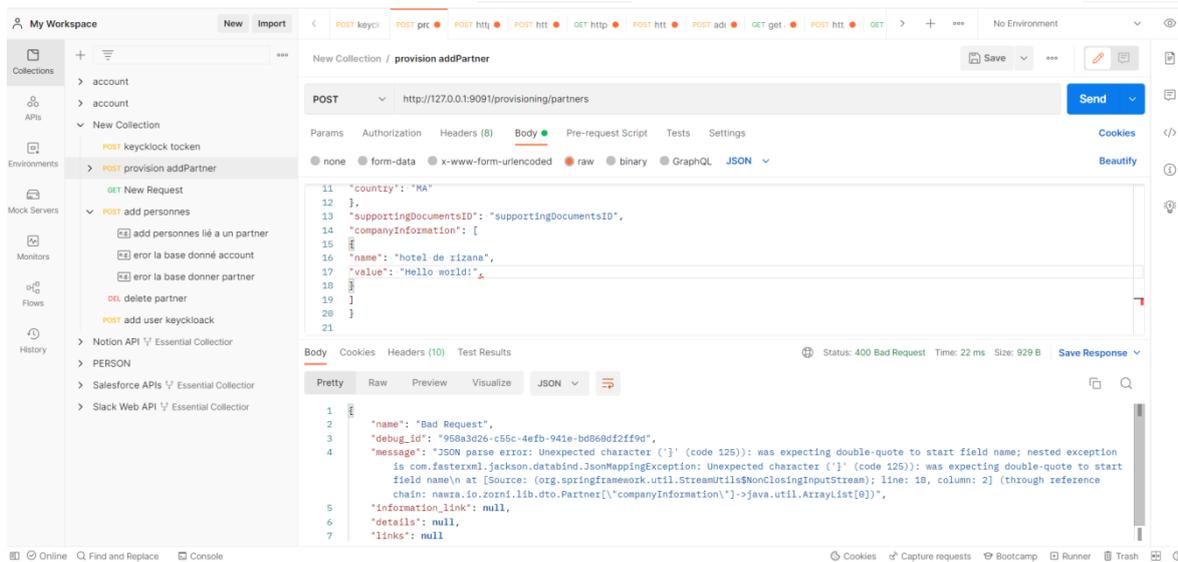


Figure 19: ajouter un Partner status 404



2.3. Status : 500 Internal Server Erreur

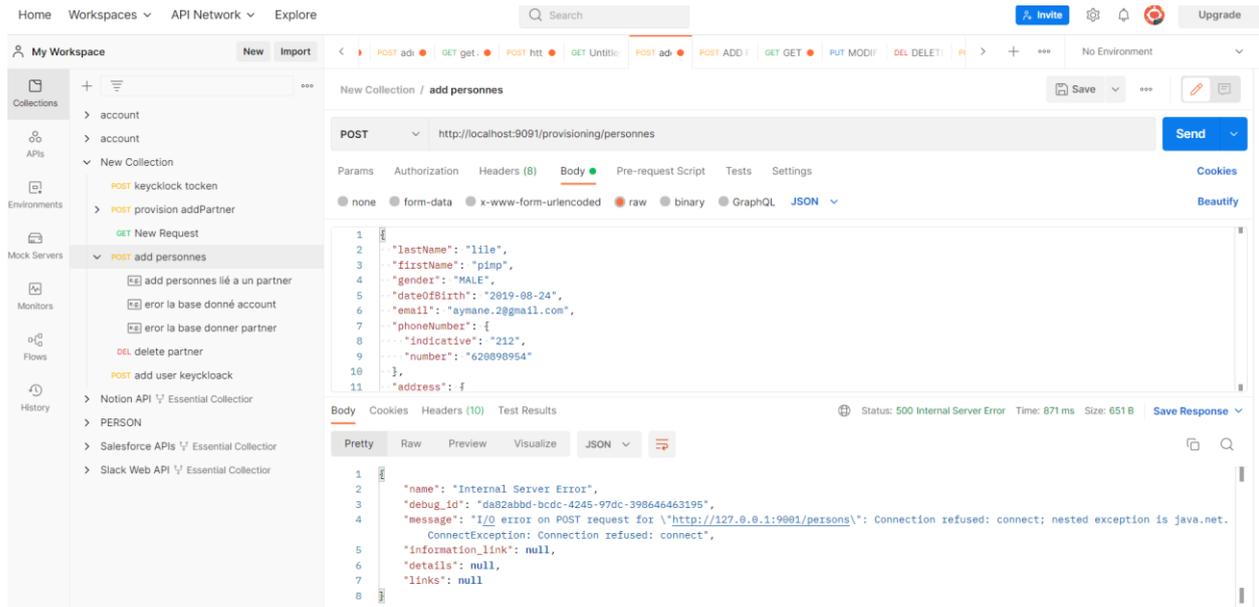


Figure 20: ajouter un Partner status 500

Cette erreur était générée par le service dans le cas où l'API provisioning ne peut pas se connecter avec l'API Partner.

3. ajouter un personne

3.1. Status code : 201 created

3.1.1. le Person a un Partner

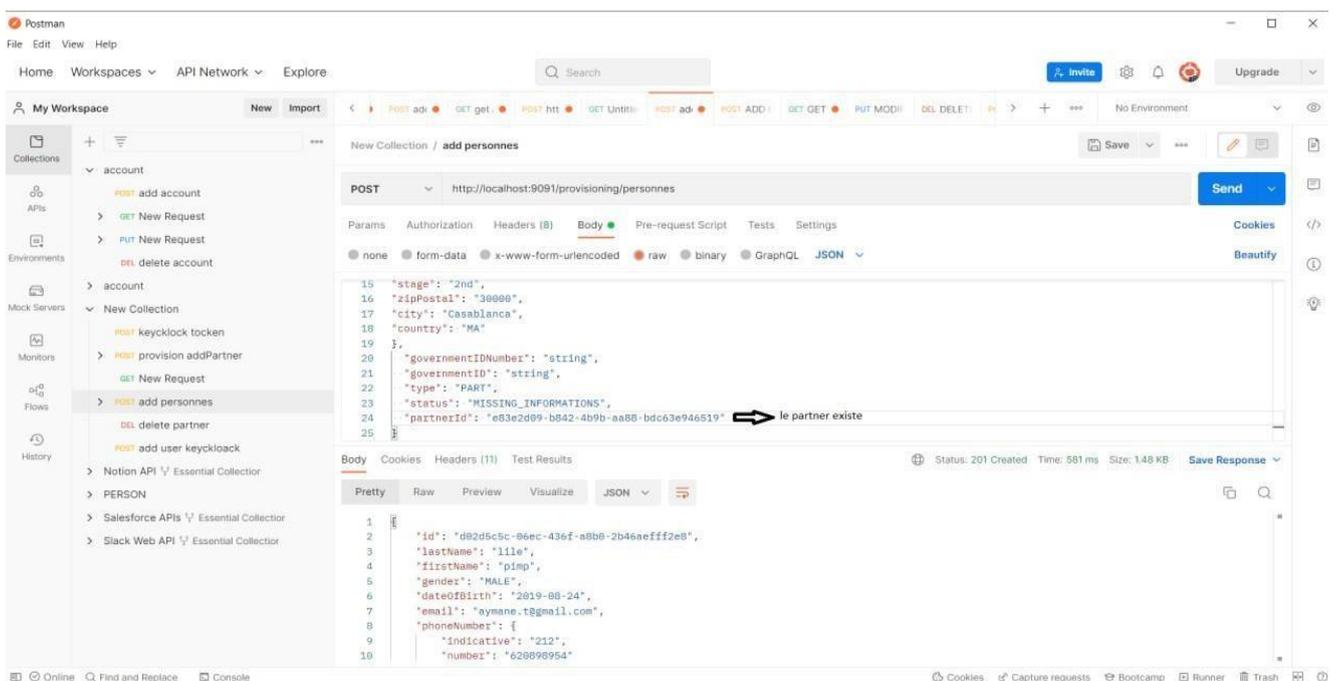


Figure 21: ajouter un Person lié un Partner status 201

La réponse envoyée : on le code statut de réponse http 201 created alors que l'api provisioning 360 fait plusieurs traitements :

- L'api provisioning fait une recherche de Partner s'il existe en local base donnée
- L'api provisioning demande à l'api Partner de rechercher de Partner est trouver si il existe (il peut que le Partner fait un compte et après un temps il l'a supprimé)
- L'api provisioning 360 à envoyer les données à l'api Person
- api provisioning 360 enregistre l'id de Person
- afficher les informations

3.1.2. Si le Person n'a pas de Partner

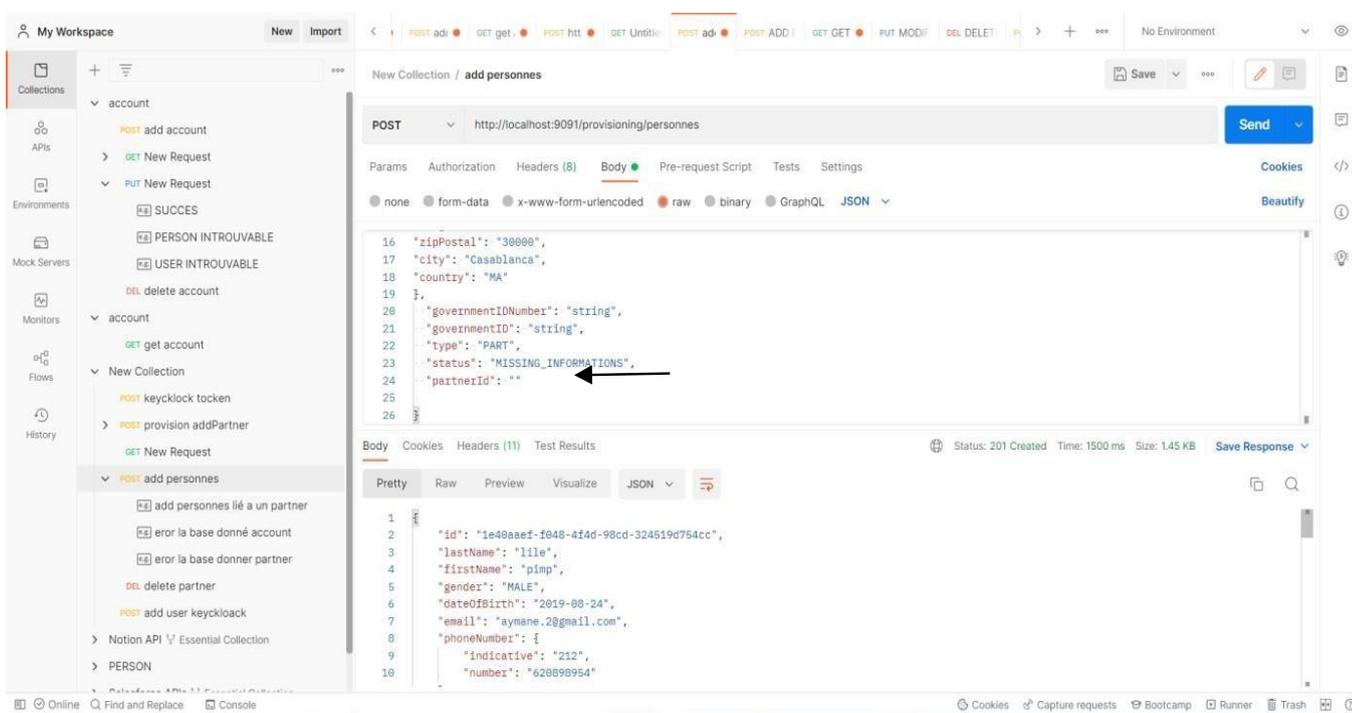


Figure 22: ajouter un Person n'est pas lié un Partner status 201

Dans ce cas il n'a pas de traitement de recherche de Partner seulement :

- envoie les données à l'api Person et le Person
- api provisioning 360 enregistre l'id de Person
- afficher les informations

3.2. Status 404 : Not Found

Si le Partner ne se trouve pas dans la base donnée provisioning

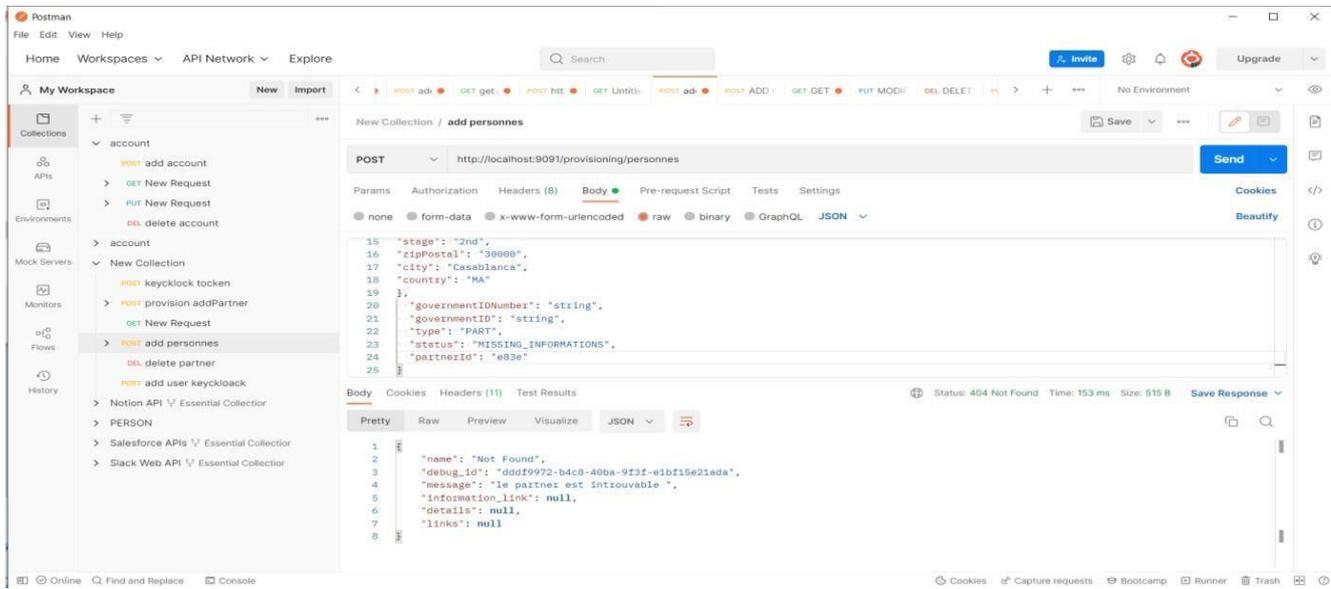


Figure 23: ajouter un Person status 404

3.3. Status 500 : Internal Server Erreur

-si le Partner est ajouter mais il a supprimé son compte

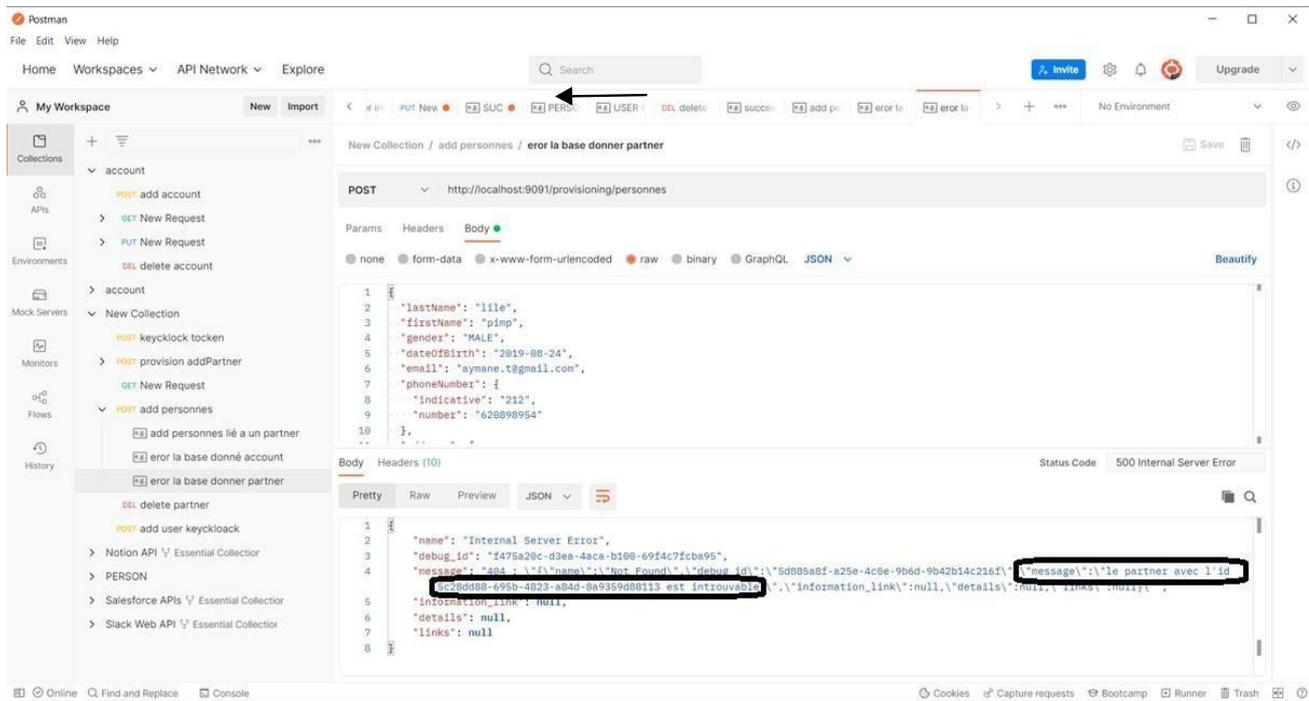


Figure 24: ajouter un Person status 500

4. ajouter compte

4.1. Staus 201 :Created

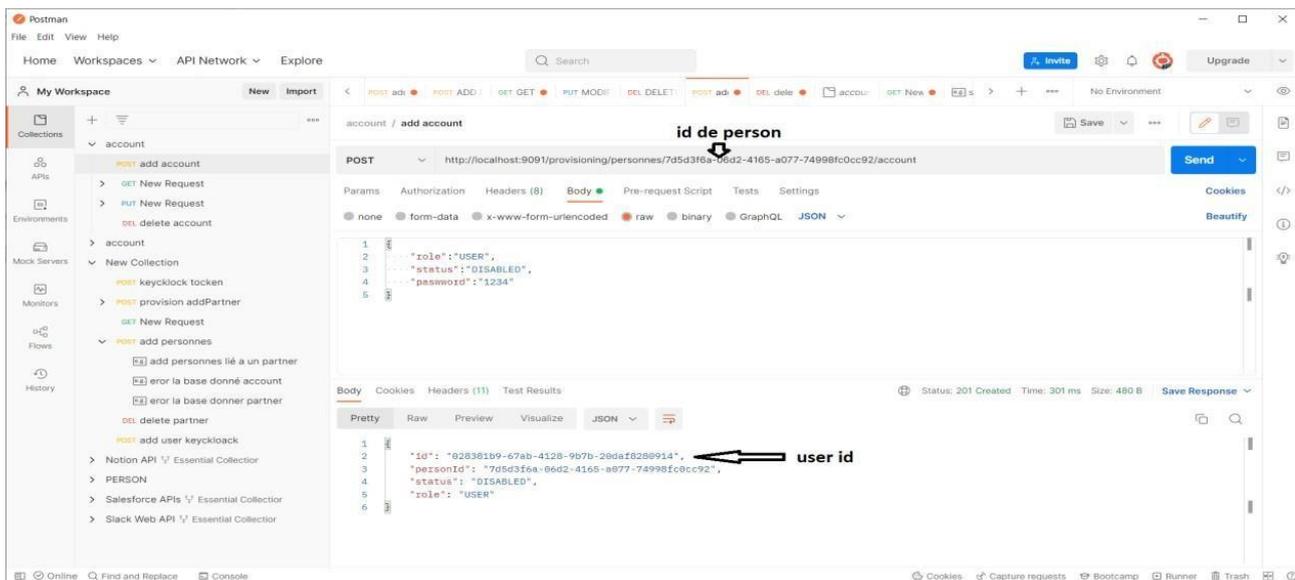


Figure 25: ajouter un compte status 201

La réponse envoyée : on le code statut de réponse http 201 created mais l'api provisioning fait plusieurs tâches :

- L'api provisioning fait une recherche de Person s'il existe en local base donnée
- L'api provisioning demande à l'api Person de recherche de Person est trouver si il existe. Il retourne son adresse mail.
- L'Api provisioning 360 demande l'accès token a Keycloak
- L'api provisioning 360 envoie l'email et mot de passe a Keycloak pour les stocker.
- api provisioning 360 enregistre le rôle le statut et user Id qui est generé par Keycloak
- api provisioning 360 afficher les informations

4.2. Status 409 : user déjà existe

-ce cas est déclenché si le Person a déjà un compte

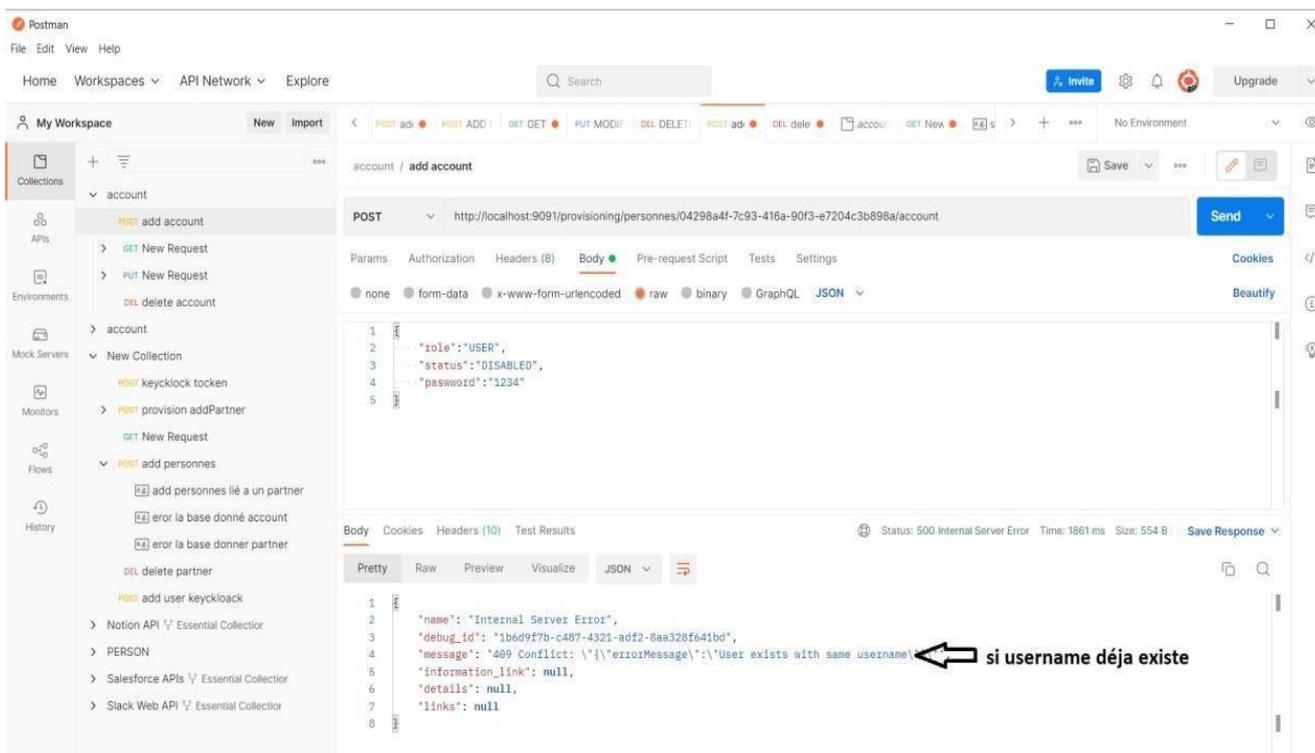


Figure 26: ajouter un compte status 409

5. consulter un compte

5.1. status 200 : OK

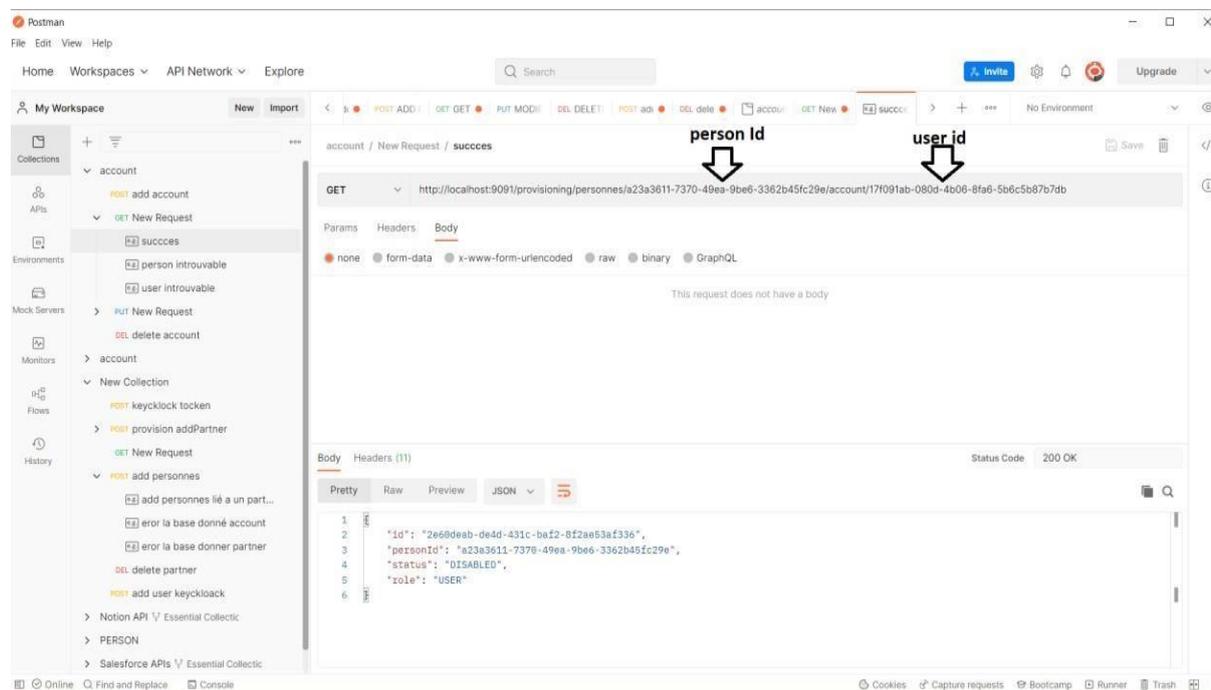


Figure 27: consulter un compte status 200

Si on veut consulter un compte, on met le protocole http GET et on ajoute l'id de la personne et Id dans le paramètre de l'URI.

La réponse envoyé : on le code statut de réponse http 200 ok alors que le compte est bien consulté

5.2. Status 404 : Not found

Cette erreur est déclenché si le id de Person ou le user id est introuvable en spécifiant Un message qui nous informe ce qui est incorrecte

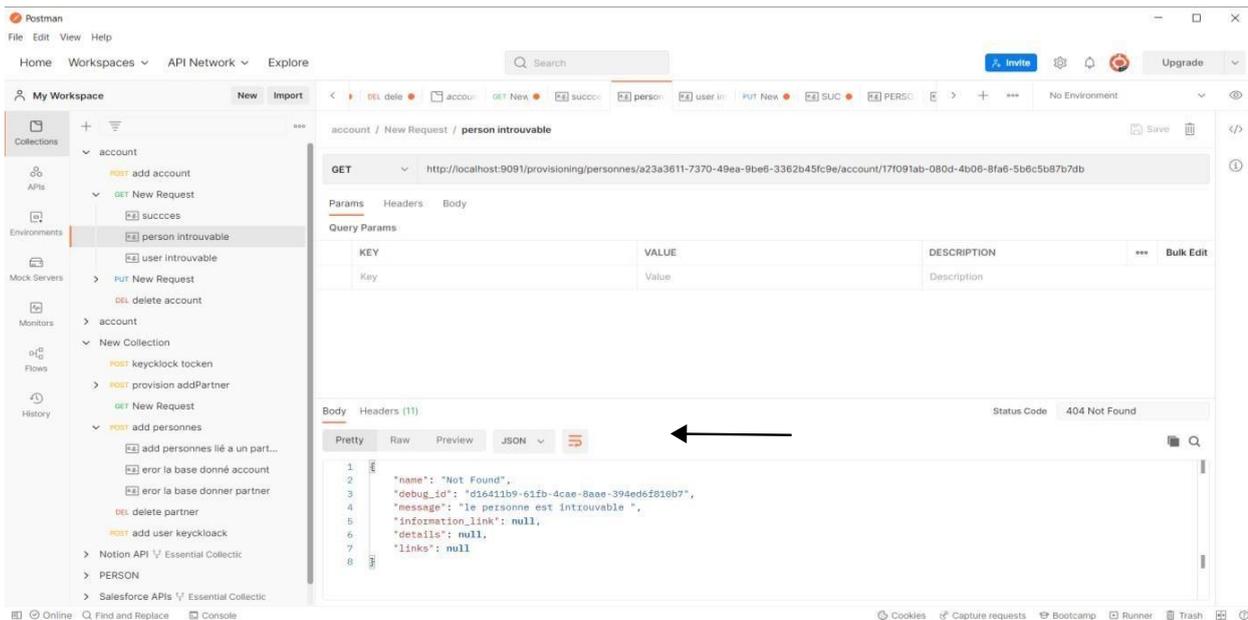


Figure 28: consulter un compte status 404 Person not found

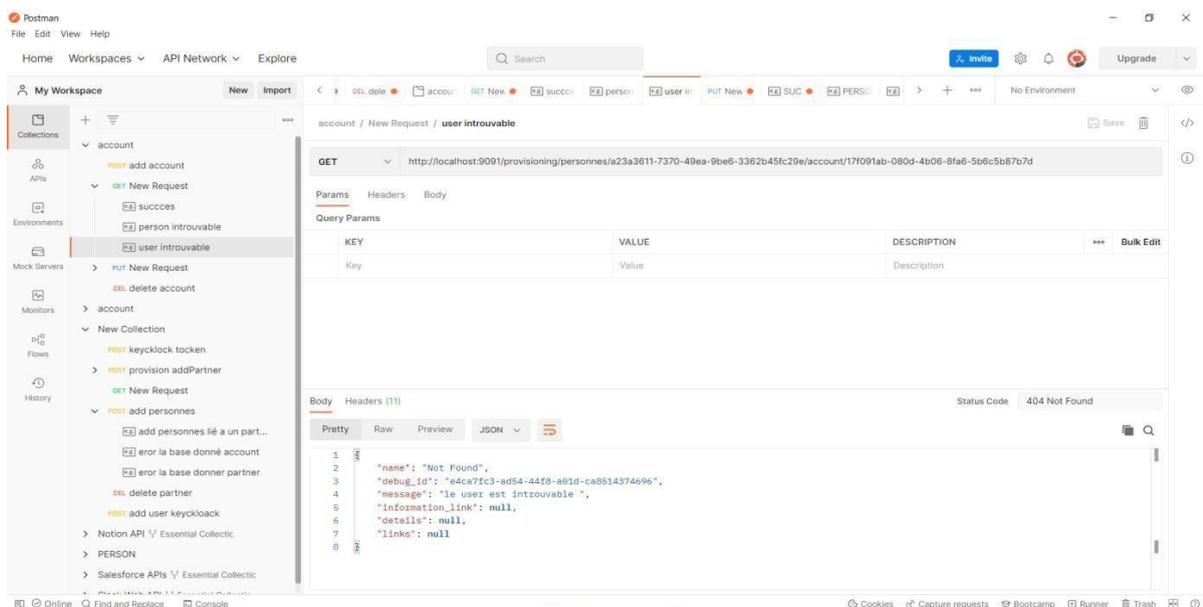


Figure 29: consulter un compte status 404 User not found

6. modifier un compte

6.1. status 201 : created

Pour modifier un compte, tous d'abord il faut consulter le compte puis le modifier. **ici** il faut premièrement choisir le Protocol, on met Put et on ajoute l'id de la personne et Id de user dans le paramètre de l'URI.

La réponse envoyé : on le code statut de réponse http 201 created alors que le compte est bien modifié.

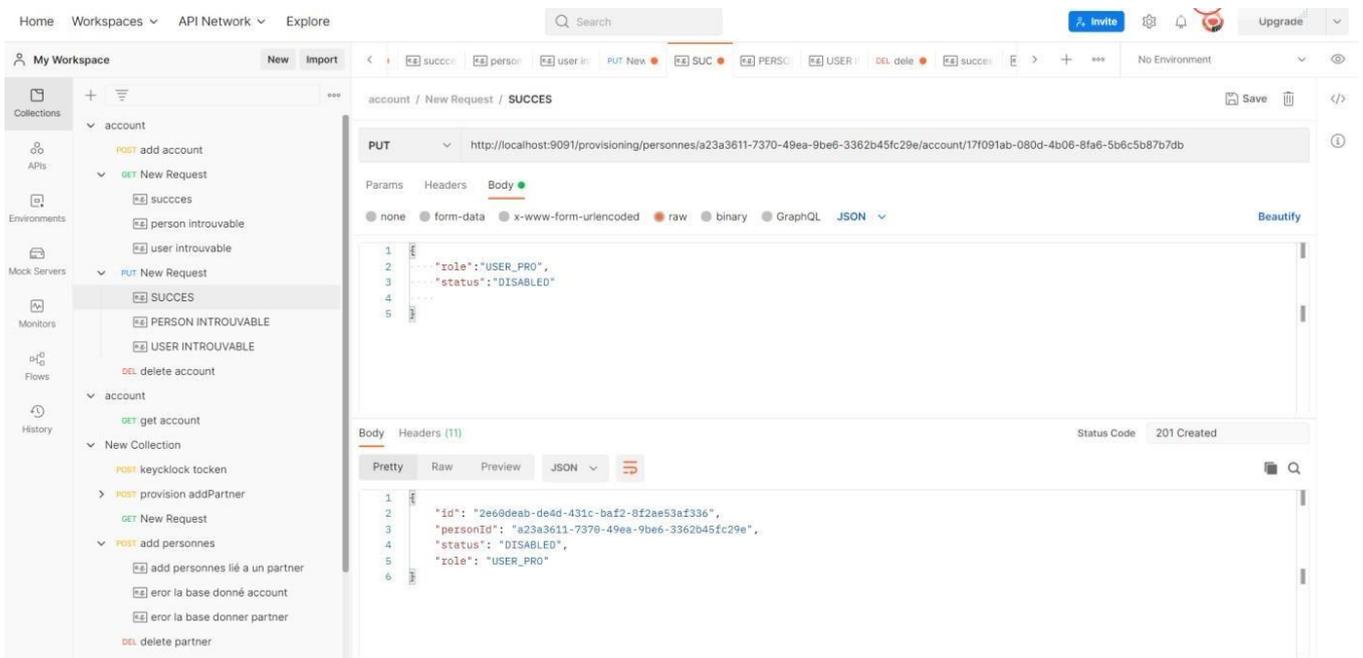


Figure 30: modifier un compte status 201

6.2. status 404 : not found

Cette erreur est déclenché si le id de Person ou le user id est introuvable en spécifiant un message qui nous informe ce qui est incorrecte.

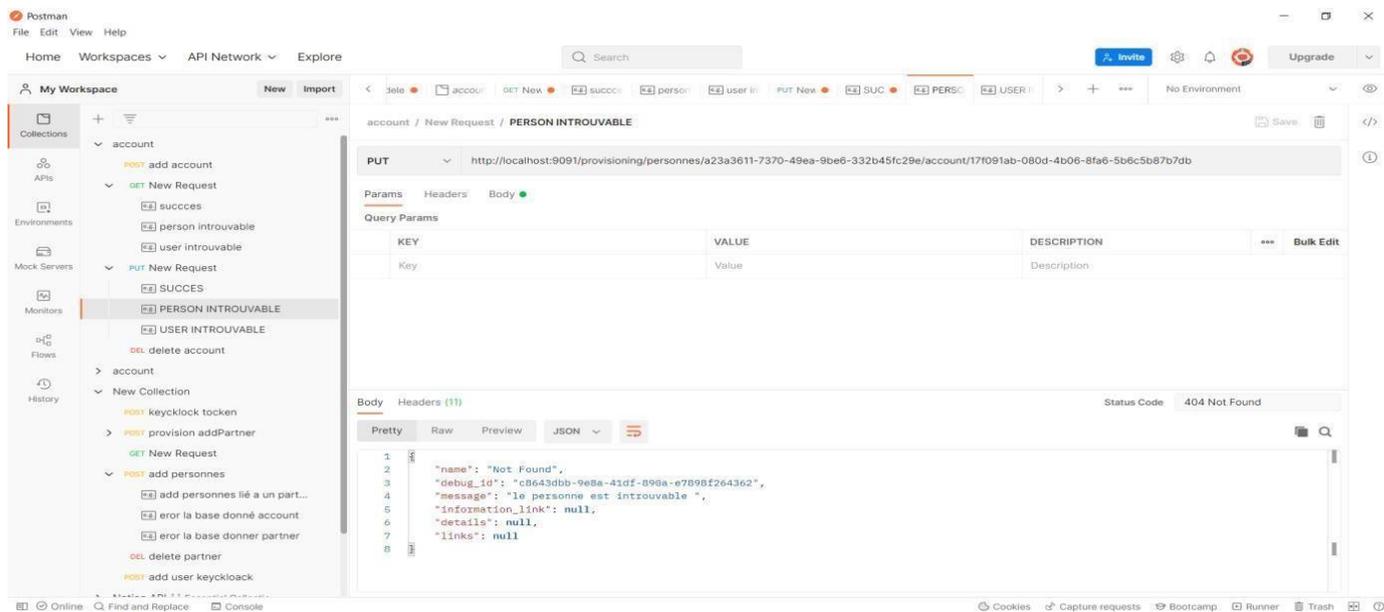


Figure 31: modifier un compte status 404 Person not found

Si le user est introuvable :

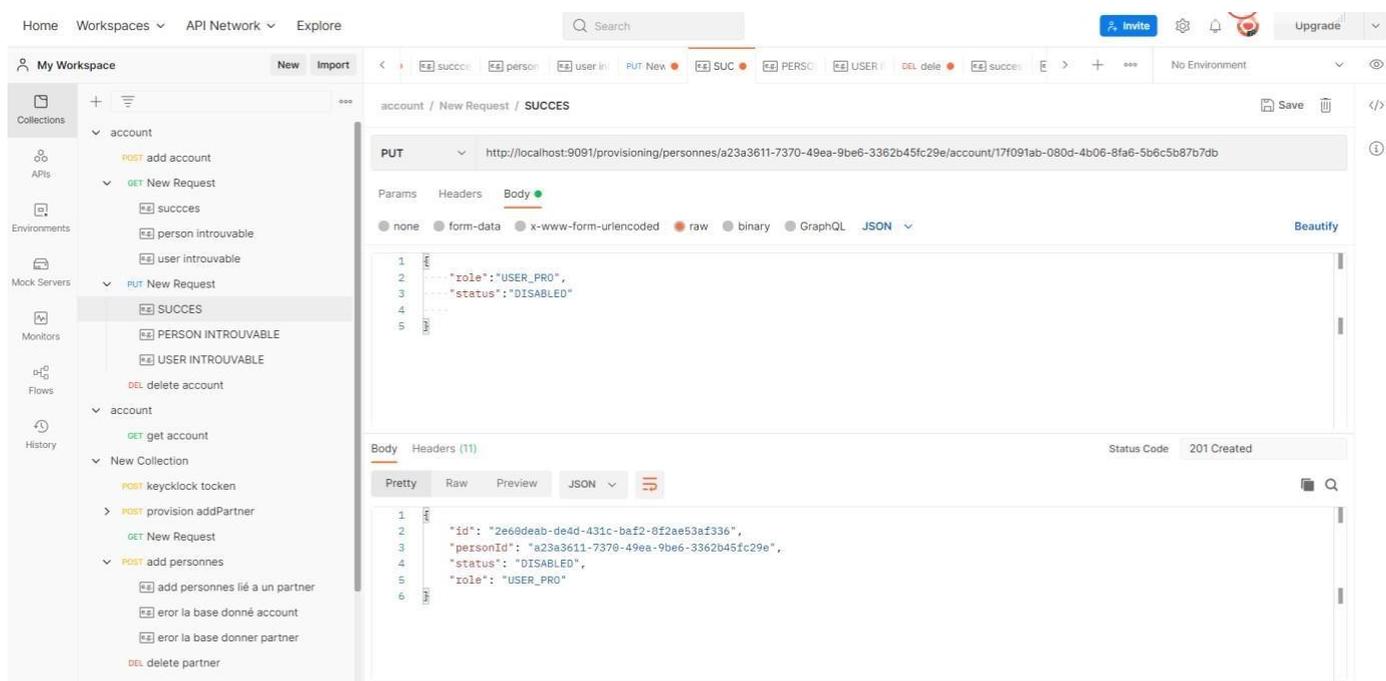


Figure 32: modifier un compte status user not found



7. supprimer un compte

Si on veut supprimer un compte, on met le protocole http DELETE et on ajoute l'id de la personne et user id dans les paramètres de l'URI.

À réponse envoyé : on le code statut de réponse http 501 not implimented alors que la personne est supprimée.

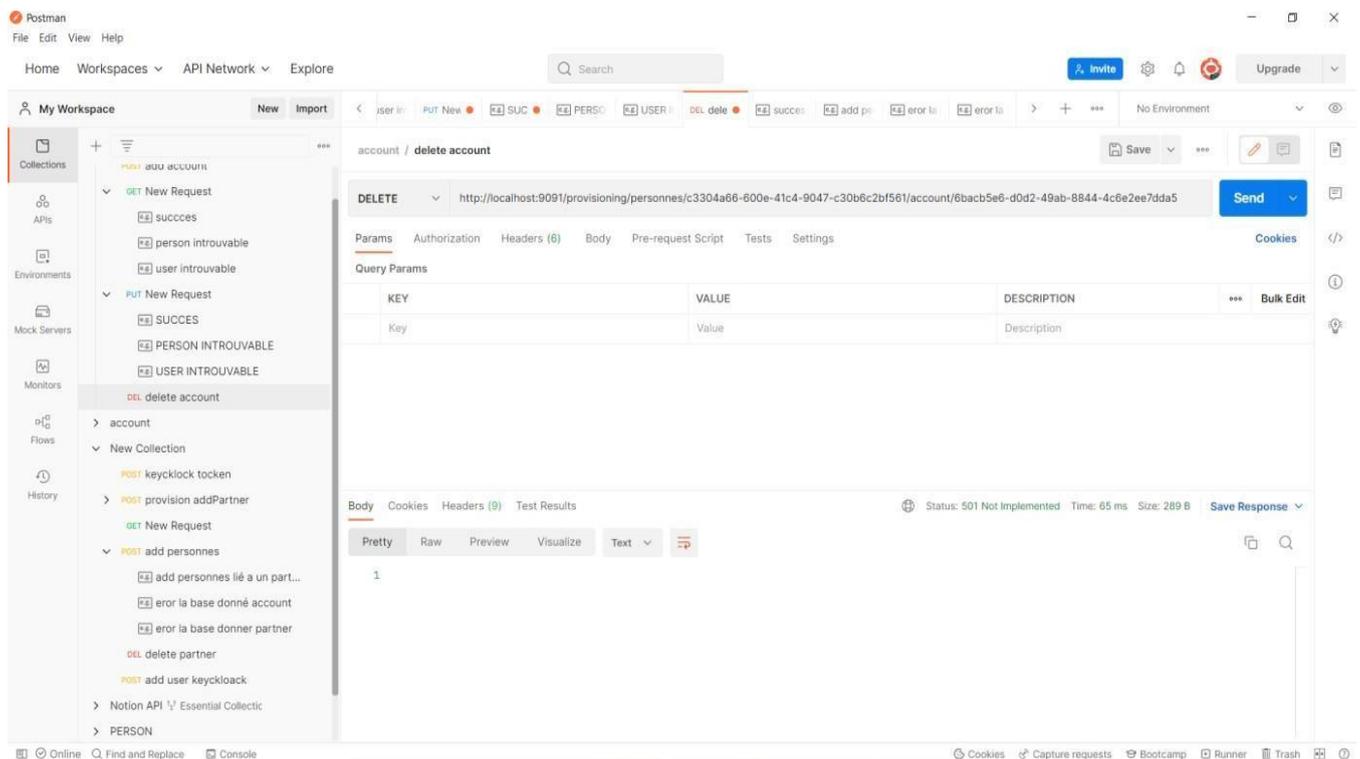


Figure 33: Supprimer un compte status 501

8. envoie d'email

8.1. email de confirmation

Dans cette partie j'ai fait une petite interface pour mieux voir comment ça marche l'envoi d'email :

ZORNI-REALM

Register

First name
aymane

Last name
benahmed

Email
aymane.15@gmail.com

Password
.....

Confirm password
.....

[« Back to Login](#)

[Register](#)

Figure 34: interface ajouter un compte

ZORNI-REALM

Email verification

⚠ You need to verify your email address to activate your account.

An email with instructions to verify your email address has been sent to your address aymane.15@gmail.com.

Haven't received a verification code in your email?
[Click here](#) to re-send the email.

Figure 35: interface demande de vérification par email

Comme il est affiché dans l'interface un message qui demande la vérification par email



Un lien est envoyé pour vérifier l'adresse mail est envoyé ce lien. Il est valable pour 5 min.

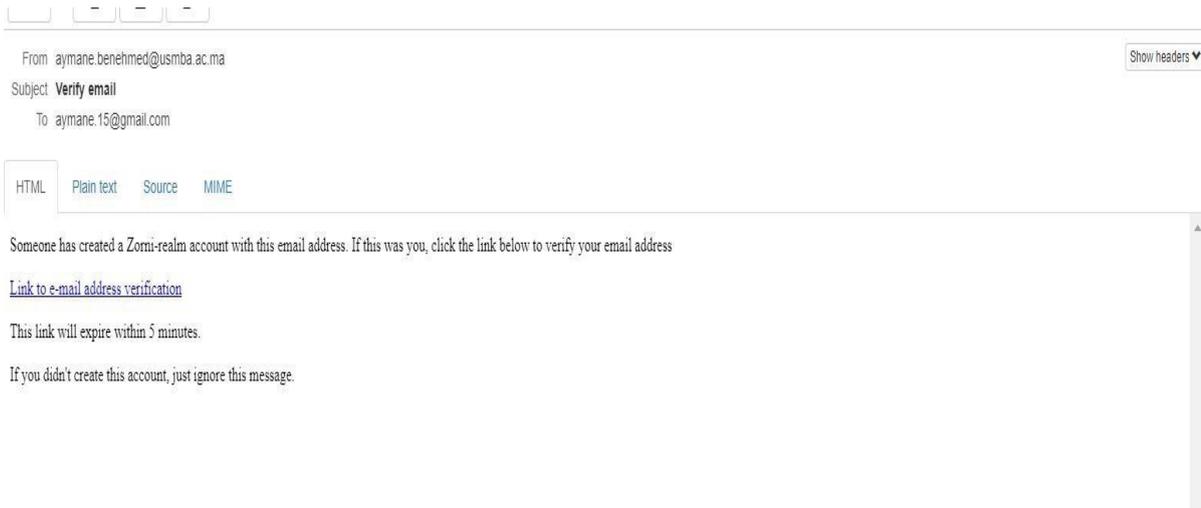


Figure 36: email de vérification

8.2. Mot de passe oublié :

Dans le cas où le mot de passe est oublié, on peut demander un email pour Réinitialiser le mot de passe.

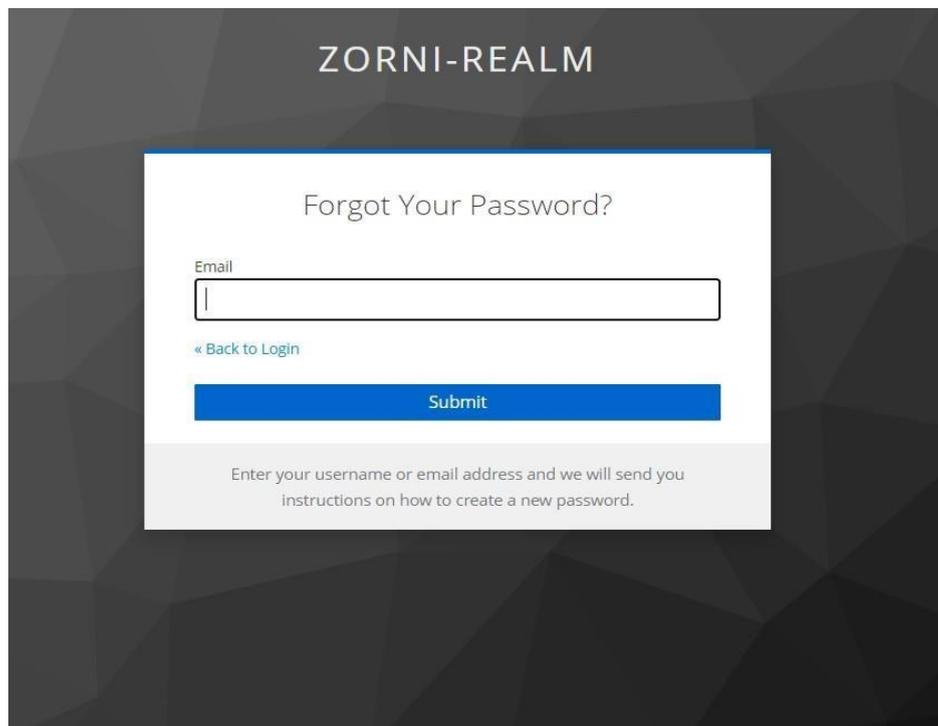


Figure 37: Interface mot de passe oublié



Un lien est envoyer pour réinstaller le mot de passe est envoyer ce lien il est valable pour 5 min



Figure 38: email de réinstaller le mot de passe

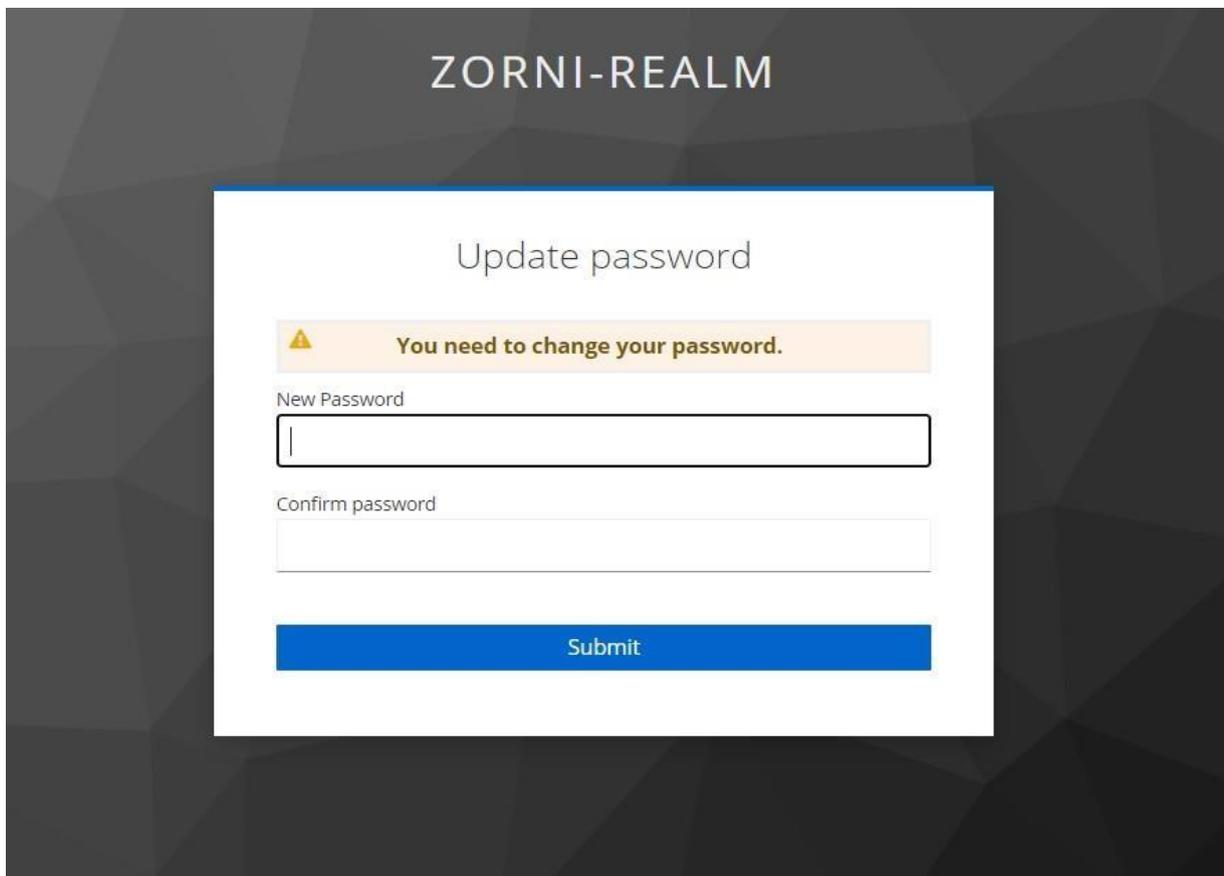


Figure 39: Interface pour réinstaller le mot de passe

Quand on clique sur le lien, il me dirige vers une interface pour changer le mot de passe.



CONCLUSION ET PERSPECTIVES

Durant ces deux mois de stage au sein du groupe « **NAWRA THECHNOLOGIE** » j'ai pu réaliser une Api provisioning 360 et la sécurité d'api.

Ce projet a été l'occasion de mettre en valeur mes connaissances acquises à la Faculté des Sciences et Techniques Fès, notamment en termes de programmation conception et modélisation, et de concrétiser mon bagage théorique.

Au cours de la réalisation de mon projet, j'ai été confronté à certaines difficultés surtout au niveau de la programmation, une autoformation s'est avéré nécessaire pour exploiter de nouveaux outils notamment la technologie Spring data, Spring security, Spring web, keycloak afin de répondre aux différents besoins fonctionnels.

Ce stage a été l'opportunité de s'initier au monde du travail et gagner une expérience aussi bien au niveau professionnel que personnel.



RÉFÉRENCES

Webographie

Documentation Spring Boot

<https://spring.io/guides/tutorials/rest/>

Ce site propose une formation de base pour Api avec Spring Boot

(Spring web, SpringData,)

Documentation XML

<https://spring.io/projects/spring-security>

Ce site propose une formation Spring Security pour sécuriser l'Api

Documentation keycloak

https://www.keycloak.org/docs/latest/server_development/

Ce site propose une formation permettant d'utiliser keycloak pour sécuriser l'api

Documentation Postman

<https://www.postman.com/documentation>

Ce site propose une formation de base pour utiliser Postman.

