جامعة سيدي محمد بن عبد الله بفاس
UNIVERSITÉ SIDI MOHAMED BEN ABDELLAH DE FES
كلية العلوم والتقنيات فاس
FACULTÉ DES SCIENCES ET TECHNIQUES DE FÈS

Projet de Fin d'Etudes

Licence Sciences et Techniques Génie Informatique

Département Informatique

Développement d'une api rest pour la gestion des partenaires

Lieu de stage : Nawra Technology

Réalisé par :

- Yendoubouam Alexandre Lalle

Encadré par :

- Pr. YOUNES DHASSI
- Mr. Mohamed KAARAR

Soutenu le 06/07/2022 devant le jury composé de :

Pr. A. ZARGHILI (Président)

Pr. M.C. ABOUNAIMA (Examinateur)

Pr. Y. DHASSI

Année Universitaire 2021-2022

Remerciements

Je tiens à remercier toutes les personnes qui ont contribué au succès de mon stage et qui m'ont aidé lors de la rédaction de ce rapport.
Tout d'abord, j'adresse mes remerciements à mon encadrant pédagogique, Mr YOUNES DHASSI, pour son encadrement et aussi pour les précieux conseils qu'il m'a prodigués tout au long de ce stage.
Je tiens à remercier vivement mon maitre de stage, Mr KAARAR Mohammed, qui n'a cessé de me faire part de ses précieux conseils, pour son accueil, le partage de son expertise au quotidien et le temps qu'il a bien voulu me consacrer. Grâce à sa confiance, j'ai pu m'accomplir totalement dans mes missions. Il fut d'une aide précieuse dans les moments les plus délicats.
Je tiens également à exprimer toute ma reconnaissance envers Mme CHAKER Ilham pour la séance de formation sur la rédaction du rapport.
Enfin, je tiens à remercier Mme MRABTI Fatiha et M. ZAHI Azeddine de m'avoir offert cette opportunité de stage.

Résumé

Dans le cadre de mon projet de fin d'étude en licence Génie informatique à la FST de Fès, j'ai effectué un stage de deux mois au sein de l'entreprise Nawra Technology. L'objectif principal de ce stage fut la réalisation d'une API REST pour la gestion des partenaires dans le cadre du projet Atlas qui vise à réaliser une plateforme digitale pour le secteur du tourisme. Pour la réalisation de ce travail, j'ai utilisé plusieurs technologies informatiques, notamment le Framework spring boot, le SGBD relationnelle PostgreSQL, le gestionnaire de version git et le logiciel Postman pour tester l'API.

Abstract

As part of my end-of-study project in Computer Engineering at the FST in Fez, I did a two-month internship at Nawra Technology. The main objective of this internship was the creation of a REST API for partner management as part of the Atlas project, which aims to create a digital platform for the tourism sector. For the realization of this work, I used several computer technologies, in particular the Spring boot framework, the PostgreSQL relational DBMS, the git version manager and the Postman software to test the API.

Sommaire

Liste des figures

Liste des acronymes

Abréviation	Désignation
JSON	**JavaScript Objet Notation**
URI	**Uniform Resource Identifier**
HTTP	**HyperText Transfert Protocol**
HATEOAS	**Hypermedia As The Engine of Application State**
DTO	**Data transfer object**
Entity	**Objet métier**

INTRODUCTION

De nos jours, il est communément admis que l'industrie touristique a un impact économique, social, culturel et environnemental de plus en plus marqué, et l'expansion continue du tourisme et des activités qui en découlent se répercute de diverses manières sur le processus de développement durable. Nawra Technology est une entreprise qui mise beaucoup sur ce secteur pour son développement. C'est en ce sens qu'elle a créé le projet Atlas qui vise à développer une plateforme digitale pour le secteur touristique.

Du 25 avril 2022 au 25 juin 2022, j'ai effectué un stage à distance au sein de l'entreprise Nawra Technology. Que j'ai obtenu à la suite d'un entretien passer avec le directeur général de la société. Les raisons qui m'ont poussé à choisir ce stage plutôt qu'un autre, sont dans un premier temps le faites que ce stage en développement pour le projet Atlas s'inscrit parfaitement dans la logique de mon projet professionnel. Et dans un second temps la difficulté et le niveau d'exigence de ce stage, me permettant ainsi de progresser et d'en apprendre plus sur les techniques de conception et développement dans une entreprise spécialisé dans le domaine.

En effet, Nawra Technology est un jeune éditeur logiciel spécialisé dans les solutions digitales modernes et innovantes. Durant mon stage chez NAWRA TECHNOLOGY mon maitre de stage était Mohamed KAARAR l'un des fondateurs de la société. J'ai pu apprendre dans d'excellentes conditions et ai bénéficié d'un soutien de qualité.

J'ai travaillé essentiellement sur la partie back du projet. Mon rôle consistait à développer une api rest pour la gestion des partenaires. Une API REST (également appelée API RESTful) est une interface de programmation d'application (API ou API web) qui respecte les contraintes du style d'architecture REST et permet d'interagir avec les services web RESTful. Pour accomplir ma tâche, j'ai été amené à m'autoformer sur l'utilisation de nombreuses technologies informatiques, en particulier le framework spring boot, à faire la conception du modèle de donnée, ainsi que l'analyse et le développement des différentes ressources de l'api.

Plus largement, cette expérience a été l'opportunité pour moi de percevoir comment une entreprise spécialisée en développement gère un projet et d'enrichir mes connaissances en programmation.

L'élaboration de ce rapport a pour principale source les différents enseignements tirés de la pratique journalière des tâches auxquelles j'étais affecté. Enfin, les nombreux entretiens que j'ai pu avoir avec les employés de la

société m'ont permis de donner une cohérence à ce rapport.
Ce rapport est constitué de trois principaux chapitres :

- Le premier chapitre intitulé « contexte général du projet » porte sur la description de l'organisme d'accueil et présente la problématique, la solution proposée ainsi que la conduite et la planification adoptée pour le projet.
- Le second chapitre intitulé « analyse et conception » présente une analyse des besoins fonctionnels et techniques ainsi que la conception adoptée.
- Le troisième chapitre intitulé « réalisation » présente la structure de l'api réalisée, les outils de développement utilisés et les services fourni par l'api.

Enfin, je terminerai par une conclusion générale et les perspectives ou améliorations que je jugerai intéressantes à apporter.

CHAPITRE I
Contexte général du projet

1. La présentation de l'organisme d'accueil

Nawra Technology est un jeune éditeur logiciel qui se lance dans l'industrie de solutions digitales modernes et innovantes. Son objectif est de concevoir, réaliser et commercialiser des produits et des solutions révolutionnaires afin de contribuer à la construction d'un avenir riche et durable. Nawra Technology est conçue par un groupe d'ingénieurs ayant un très fort potentiel technique, fonctionnel et managérial, dont objectif est de mettre en œuvre des nouveaux produits innovants répondants aux demandes et exigences du marché.
Nawra Technology, possède ses propres valeurs et son éthique. Ils lui permettent de maîtriser son domaine d'activité et de technologie, d'établir des relations constructives entre collègues et avec ses clients.
Ces valeurs ne constituent pas seulement un code de conduite, ce sont aussi des principes directeurs. Enchâssés dans son ADN, ils façonnent sa culture éthique, créant un état d'esprit partagé qui maintient l'éthique au cœur de ses décisions et actions. CHALENGE, INNOVATION et AVENIR définissent son ADN.

Nawra Technology se base sur les deux principes suivants :

- L'organisation de la production doit laisser davantage place à l'inventivité plutôt qu'à la planification.
- L'expérimentation scientifique et le retour des clients sont préférés au développement théorique des produits.

Ces valeurs et principes influent sur la manière dont elle répond aux besoins de ses clients tout en respectant les exigences réglementaires et contractuelles propres à chacune de ses activités.

Figure 1 Logo Nawra Technology

2. Présentation du projet

AtlasProject est une plateforme digitale touristique qui offre des services divers et variés à ses clients, la plateforme regroupe plusieurs types d'activités (hébergement, voyage, événements...).
Ma mission durant ce stage consistait à développer l'une des api rest du projet. Il s'agit de l'api « PartnersApi » qui permet la gestion des partenaires tels que des hôtels, des auberges, des entreprises liées au secteur du tourisme…

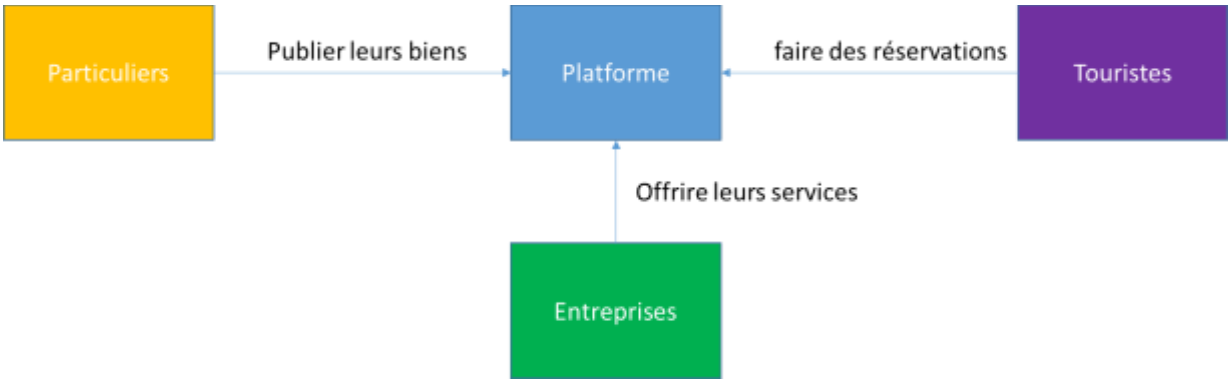


Figure 2 Présentation du projet

3. Problématique

Le Maroc propose un large éventail de paysages et d'activités qui ne laissent pas les touristes indifférents : plages et stations balnéaires sur les côtes méditerranéennes ou atlantiques, monts de l'Atlas ou dunes, oueds, et oasis du désert, ou bien ruelles secrètes de la médina des belles villes Fès, Tanger, Marrakech, Agadir, Essaouira ou Casablanca.
Cependant, après une observation continuelle, dessites web et applications utilisés dans l'industrie touristique marocaine, nous avons pu recenser les problèmes suivants :

- Il n'existe pas assez de sites web et d'applications mobile dédié au tourisme, tandis que la plupart sites web présent sur le marché sont peu fonctionnels et ne permettent pas de faire des réservations, de louer des biens, ou de proposer des activités multiples et variés aux touristes.
- La plupart des sites web présent actuellement sur le marché ne permettent pas aux entreprises et aux particuliers de publier des annonces pour leurs biens ou de mettre en valeur des activités ou des lieux culturels.

4. Les solutions proposées

Après l'étude des problèmes ci-dessus, Nawra Technology a décidé de développer une plateforme digitale touristique qui vise les tâches suivantes :

- Permettre aux touristes de passer leur séjour dans les meilleures conditions en leur permettant de faire des réservations, de louer des biens ou en leur recommandant les meilleurs hôtels et les meilleurs moyens de déplacement.
- Permettre aux entreprises et aux particuliers de publier des annonces pour leurs biens, d'offrir leurs services ou de mettre en valeur des activités touristiques ou des évènements culturels.

5. Conduite du projet

Le projet est composé d'un ensemble d'api (brique applicative) et chaque api est responsable d'un périmètre fonctionnel précis et communique avec les autres api.

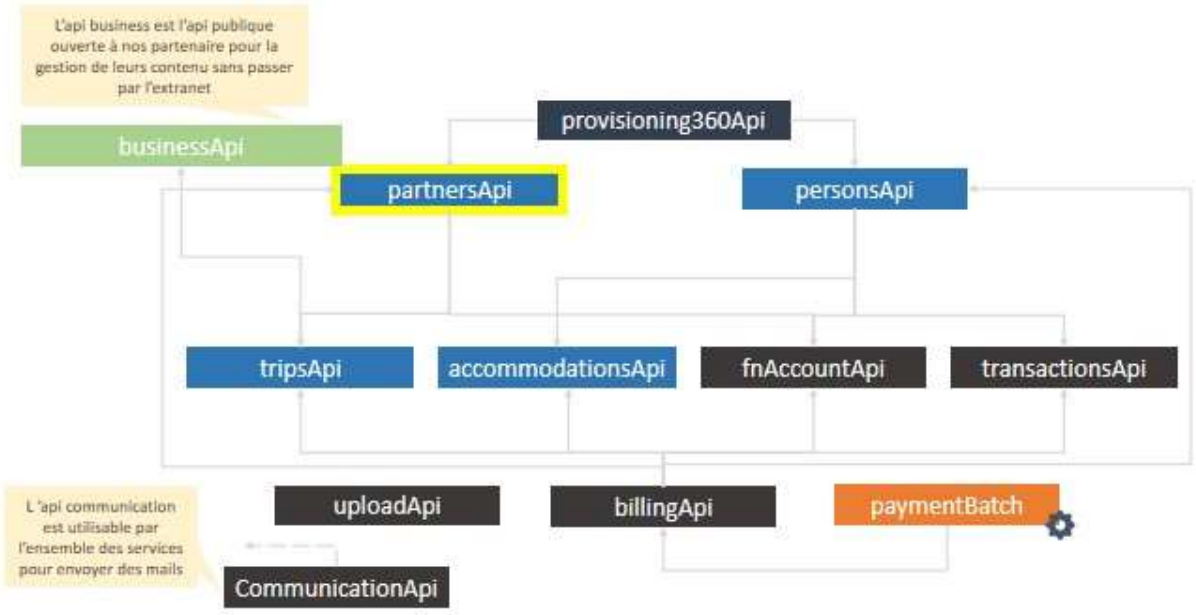


Figure 3 : architecture du projet

- L'api CommunicationAPi est l'api responsable de l'envoi de mails.
- L'api AccommmodationApi est l'api responsable de la gestion des hébergements pour tout type de profil partenaire ou particulier.

- L'api provisioning360Api est l'api responsable de la création de personnes physiques et personnes morales.
- L'api PersonsApi est l'api responsable de la gestion de personnes physique pour tout type de profil partenaire ou particulier.

Mon stage s'est déroulé en trois principales phases :

a. La phase d'autoformation

Je me suis formé sur le fonctionnement des api restful, l'utilisation du framework spring et de nombreux autres technologies informatiques.
J'ai suivi de nombreuses formations de mon maitre de stage et j'ai fait plusieurs miniprojets.

b.La phase de conception

À partir de la documentation (contrat d'interface qui décrit comment on consomme les services) de l'api PartnersApi, j'ai mis en place un modèle de données pour mon api.

Figure 4 : Documentation de l'api

c. La phase de développement

J'ai développé les différentes ressources de mon api en respectant les

consignes du contrat d'interface. Puis j'ai terminé par la mise en place de tests unitaires.

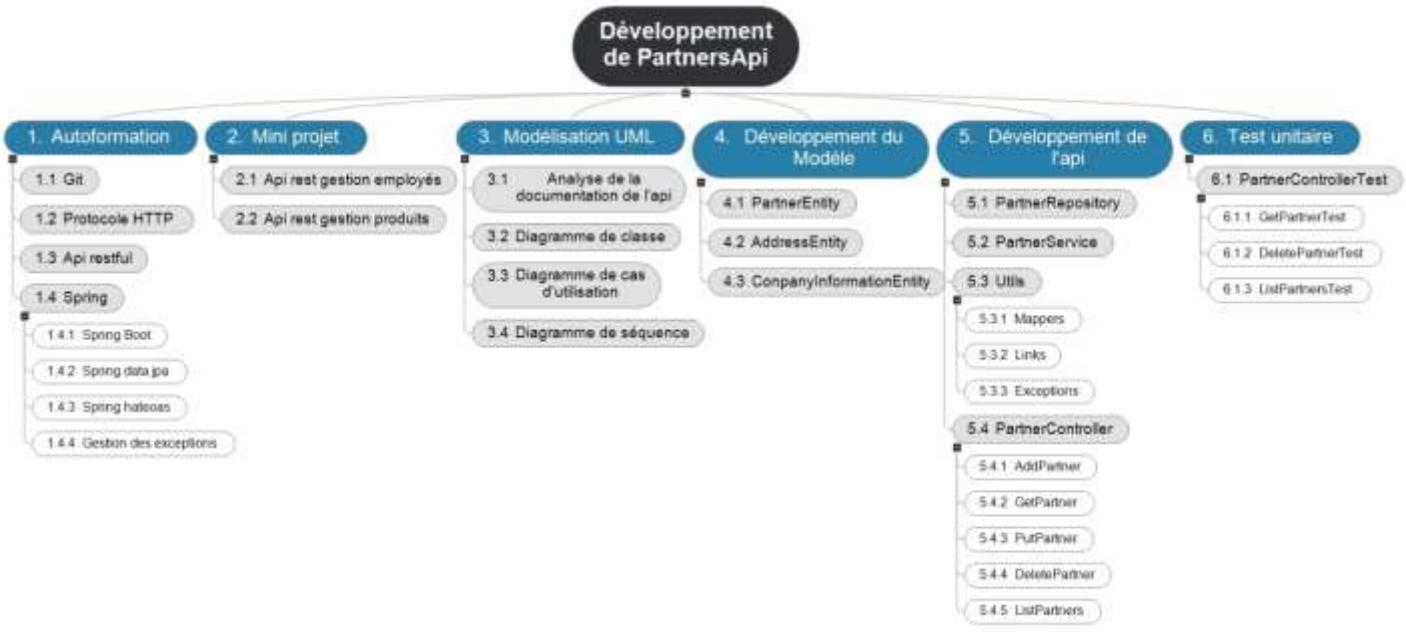


Figure 5 : Diagramme WBS

6. Planification Du Projet

Chaque projet nécessite un planning représentant visuellement l'état d'avancement des différentes activités qui constituent le projet. Mon projet de fin d'études « Développement d'une api rest pour la gestion des partenaires » est réalisé selon un planning représenté sous forme de diagramme de Gantt réalisé à l'aide du logiciel GanttProject.

Nom	Date de début	Date de fin
> Phase d'autoformation	25/04/2022	17/05/2022
⌄ Phase de modélisation	18/05/2022	24/05/2022
Formation sur le fonctionnement de l'api	18/05/2022	18/05/2022
Conception d'un modèle de données pour PartnersApi	19/05/2022	24/05/2022
⌄ Phase de développement	25/05/2022	24/06/2022
Formation sur les méthodes de développement du projet	25/05/2022	25/05/2022
Autoformation sur spring boot et spring data jpa	26/05/2022	30/05/2022
Formation sur les méthodes de développement du projet	31/05/2022	31/05/2022
Mise en place des entity et services de l'api PartnersApi	01/06/2022	14/06/2022
Formation sur les méthodes de développement des différentes ressources	15/06/2022	15/06/2022
Développement des différents endpoints	16/06/2022	21/06/2022
Présentation de l'api	22/06/2022	22/06/2022
Vérification et correction des erreurs	23/06/2022	24/06/2022

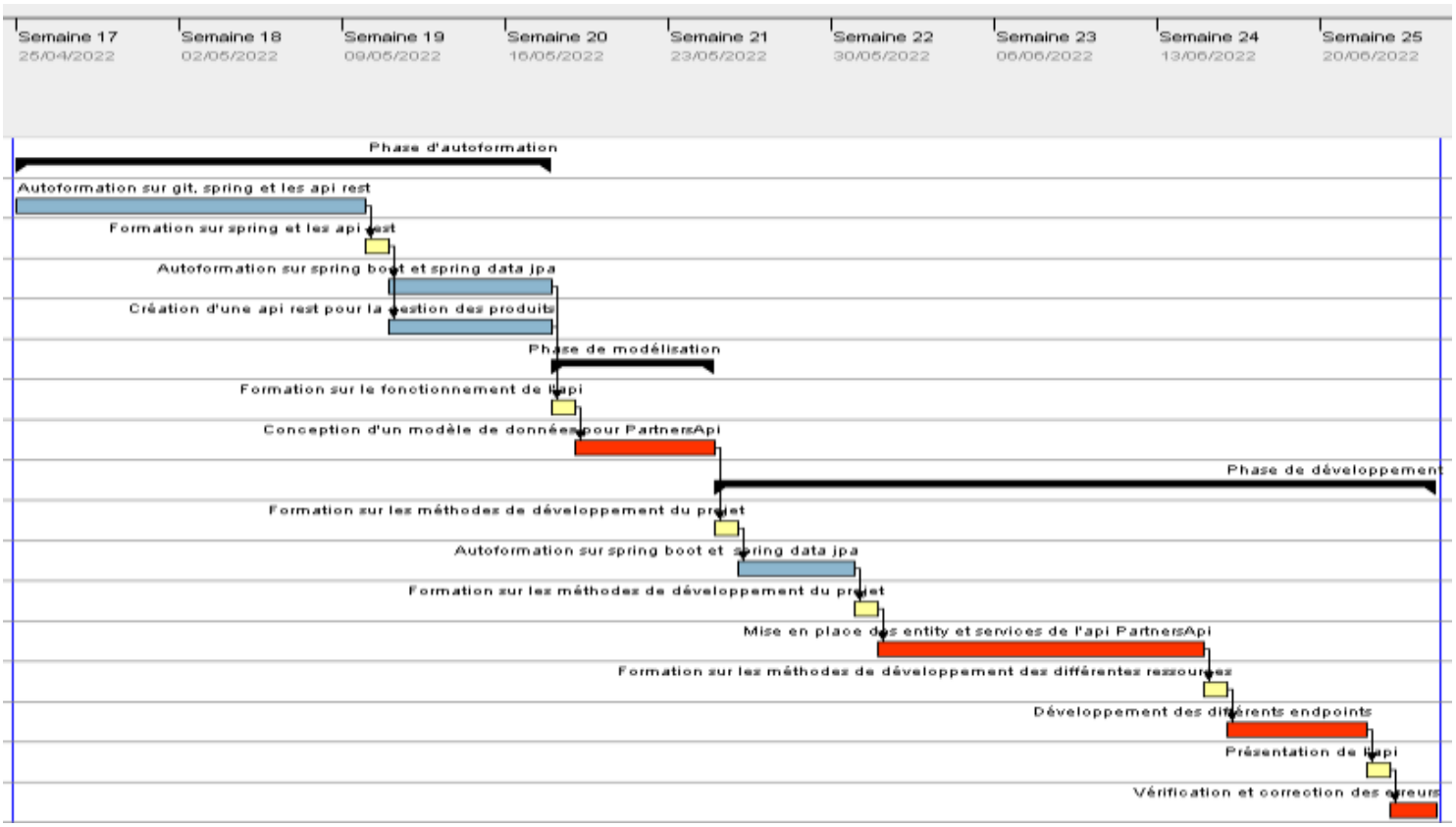


Figure 6 : Diagramme de Gantt

Autoformation Conception et Développement Cours et Formation

CHAPITRE II
ANALYSE ET CONCEPTION

1. Analyse des besoins

a. Analyse des besoins fonctionnels

L'api « PartnersApi » fournit 5 principaux services :

- **Ajouter un partenaire**

Utilise la ressource addPartner pour ajouter un nouveau partenaire dans la base de données.

- **Modifier un partenaire**

Utilise la ressource putPartner pour modifier les informations d'un partenaire dans la base de données.

- **Supprimer un partenaire**

Utilise la ressource deletePartner pour supprimer un partenaire de la base de données.

- **Afficher un partenaire**

Utilise la ressource getPartner pour afficher les informations d'un partenaire.

- **Afficher la liste des partenaires**

Utilise la ressource listPartners pour afficher la liste de tous les partenaires.

b. Analyse des besoins techniques

API est l'abréviation de "*Application Programming Interface*". C'est une **interface** de programmation, c'est-à-dire un ensemble de classes, de fonctions et de méthodes qui servent de **façade** à un logiciel. D'autres logiciels pourront donc accéder aux services de ce logiciel grâce à cette interface.

REST est un type d'architecture d'API qui signifie REpresentational State Transfer. Il a été inventé par l'Américain Roy Fielding dans les années 2000, période charnière dans la reconnaissance du potentiel des API web, afin de mettre en place des méthodes simples pour accéder à des services web.

Les API REST sont un type d'architecture d'API qui a été construit pour le web. Il utilise le protocole HTTP et des URI (comme par exemple des URL) pour identifier les ressources. Avec HTTP, nous pouvons donc utiliser 4 opérations (ou verbes) pour manipuler nos ressources via une API : GET, POST, PUT et DELETE. Ces opérations ont donc 4 fonctions résumées par l'acronyme **CRUD** (**C**reate, **R**ead, **U**pdate, **D**elete).

Pour que notre API soit considérée comme RESTful, elle doit respecter le principe d'architecture REST qui s'applique aux services web. Les principales contraintes ont été proposées par Roy Fielding et sont les suivantes :

- **Client-Serveur** – un mode de communication avec séparation des rôles entre client et serveur ;
- **Stateless Server** – les requêtes doivent contenir toutes les informations nécessaires au traitement ;
- **Cache** – la réponse du serveur doit être mise en mémoire côté client ;
- **Uniform Interface** – la méthode de communication entre client et serveur doit être sous la forme d'une URL.

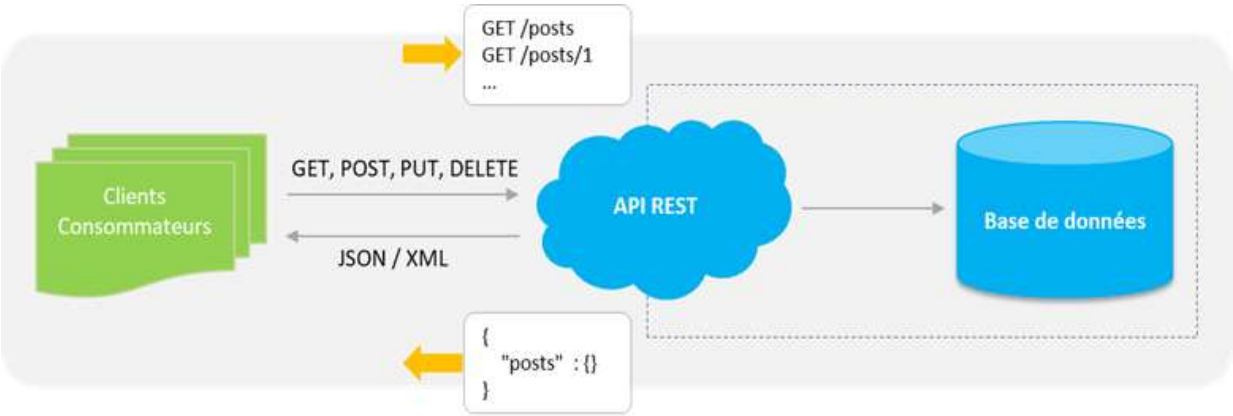


Figure 7 Api Rest

2. Conception Adoptée

a. Diagramme de cas d'utilisation

Les diagrammes de cas d'utilisation (DCU) sont des diagrammes UML utilisés pour une représentation du comportement fonctionnel d'un système logiciel. Ils sont utiles pour des présentations auprès de la direction ou des acteurs d'un projet, mais pour le développement, les cas d'utilisation sont plus appropriés. En effet, un cas d'utilisation (use cases) représente une unité discrète d'interaction entre un utilisateur (humain ou machine) et un système. Ainsi, dans un diagramme de cas d'utilisation, les utilisateurs sont appelés acteurs (actors), et ils apparaissent dans les cas d'utilisation. Le diagramme de cas d'utilisation permet d'organiser les

besoins et de recenser les grandes fonctionnalités d'un système.

Dans le cas présent, notre acteur « Client Applicatif » peut représenter une application front ou alors une autre api du projet Atlas qui consomme les services de notre api.

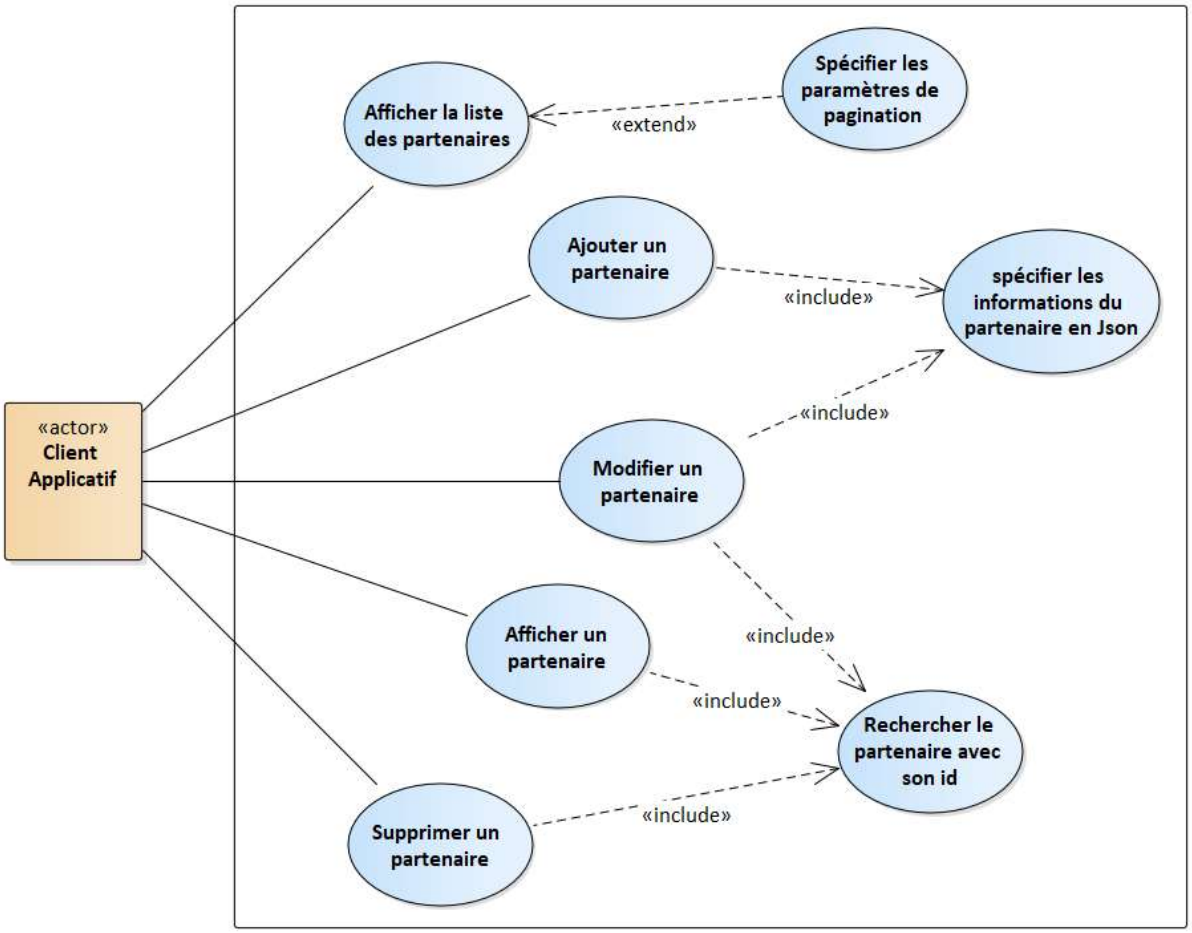


Figure 8 : Diagramme de cas d'utilisation

b. Diagrammes de Séquence

Les diagrammes de séquence UML sont des diagrammes d'interaction qui détaillent la façon dont les opérations sont effectuées. Ils capturent l'interaction entre les objets dans le cadre d'une collaboration. Les diagrammes de séquence sont centrés sur le temps et ils montrent visuellement l'ordre de l'interaction en utilisant l'axe vertical du diagramme pour représenter le temps, quels messages sont envoyés et quand.

Cas d'utilisation ajouter un partenaire

Lors de la création d'un nouveau partenaire, le client applicatif utilise la ressource addpartner. Il envoie une requête avec les informations du nouveau partenaire en format JSON avec la méthode http post. Si la requête est bien formulée et qu'on n'a aucune erreur au niveau du serveur, alors le service s'occupe ensuite de renseigner la date et le status, puis de faire appel au repository pour sauvegarder les informations du partenaire. Ensuite, le controller renvoie le code HTTP 201 CREATED et une représentation du partenaire créé avec son id et les liens hateoas pour accéder aux autres services de l'api.

Par contre, en cas d'erreur :

L'exception FailureInSave sera déclenchée pour indiquer une erreur du SGBD.

Le code de réponse d'erreur serveur HTTP 500 **Internal Server Error** indiquera que le serveur a rencontré un problème inattendu qui l'empêche de répondre à la requête.

Le code de statut de réponse HTTP 400 **Bad Request** indiquera que le serveur ne peut pas comprendre ou traiter la requête en raison d'une erreur côté client.

Le code de statut de réponse HTTP 404 **Ressource Not Found** indiquera que le partenaire n'existe pas dans la base de données.

Description textuelle	
Nom du cas d'utilisation	Ajouter un partenaire
Description	Ce cas d'utilisation permet d'ajouter un nouveau partenaire dans la base de données.
Acteur	• Application front • Une autre api rest du projet Atlas
Pré condition	Arrivé d'un nouveau partenaire
Post condition	le Controller renvoie le code HTTP 201 CREATED et une représentation du partenaire créé avec son id et les liens hateoas pour accéder aux autres

	ressources de l'api
Scénario normal	
• Le client spécifie les informations du partenaire en format JSON puis lance la requête avec la méthode HTTP post • Si la requête est bien formulée et qu'on n'a aucune erreur au niveau du serveur, alors le service fera appel au repository pour sauvegarder les informations du partenaire puis les affichées.	
Scénario Alternatif 1	
• Le client applicatif spécifie les informations du partenaire en format JSON puis lance la requête avec la méthode HTTP post • Le corps de la requête n'est pas valide • Le controller renvoie un message d'erreur 400 **Bad Request**	
Scénario Alternatif 2	
• Le client spécifie les informations du partenaireen format Json puis lance la requête avec la méthode HTTP post • La sauvegarde ne fonctionne pas à cause d'un problème coté serveur • Le controller renvoie un message d'erreur 500 **Internal Server Error**	

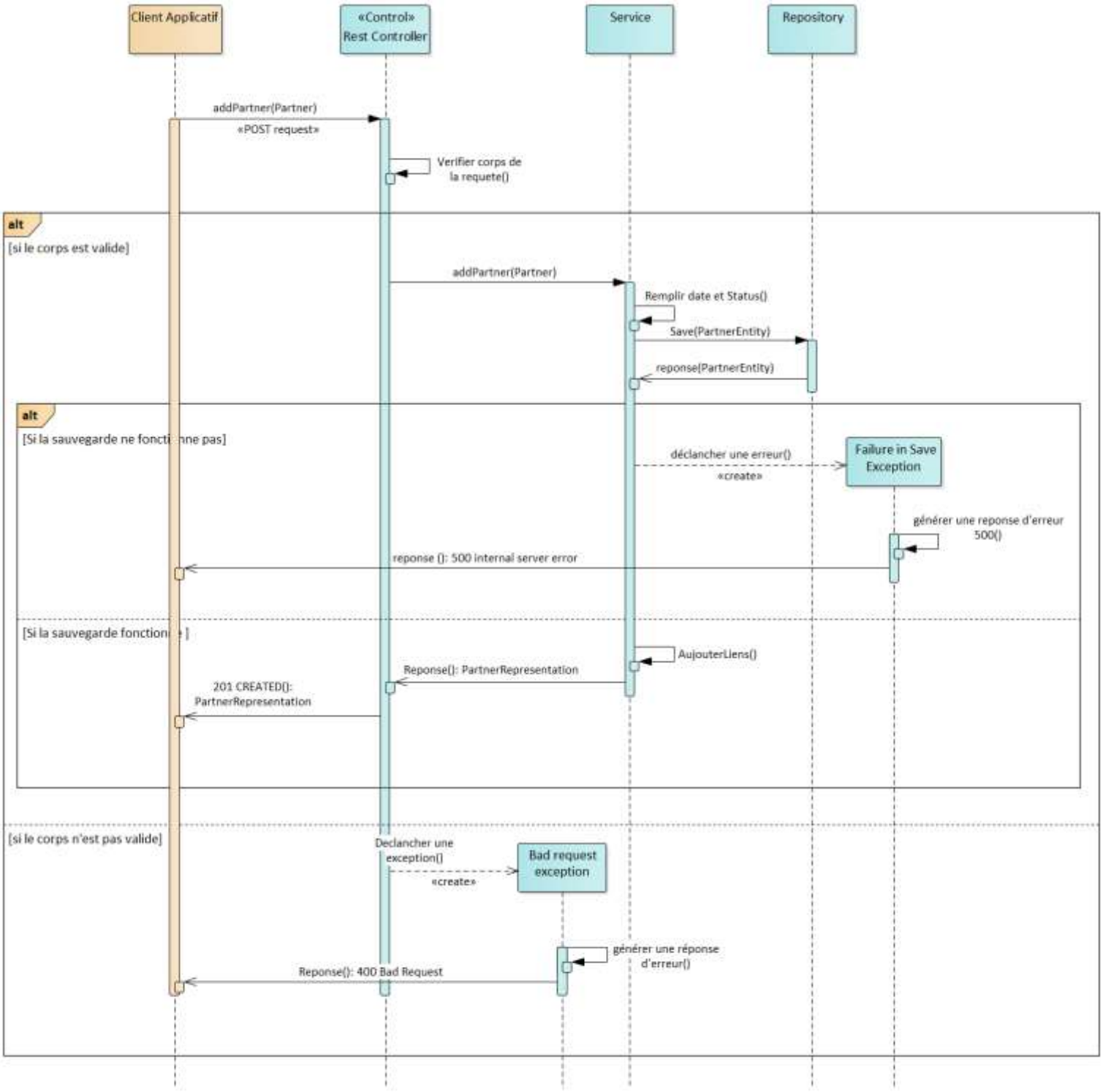


Figure 9 : Cas d'utilisation ajouter un partenaire

Cas d'utilisation modifier un partenaire

Lors de la modification d'un partenaire, le client applicatif utilise la ressource putPartner. Il spécifie l'id du partenaire dans l'URI de la requête ainsi que les informations du partenaire en format JSON avec la méthode HTTP put. Si la requête est bien formulée, que le partenaire existe et qu'on n'a aucune erreur au niveau du serveur, alors le service fait appel au repository pour sauvegarder la modification du partenaire. Ensuite, le controller renvoie le code HTTP 201 CREATED et une représentation du partenaire créé avec les liens hateoas pour accéder aux autres services de l'api.

Par contre, en cas d'erreur :

L'exception FailureInSave sera déclenchée pour indiquer une erreur du SGBD.

Le code de réponse d'erreur serveur HTTP 500 **Internal Server Error** indiquera que le serveur a rencontré un problème inattendu qui l'empêche de répondre à la requête.

Le code de statut de réponse HTTP 400 **Bad Request** indiquera que le serveur ne peut pas comprendre ou traiter la requête en raison d'une erreur côté client.

Le code de statut de réponse HTTP 404 **Ressource Not Found** indiquera que le partenaire n'existe pas dans la base de données.

Description textuelle	
Nom du cas d'utilisation	Modifier un partenaire
Description	Ce cas d'utilisation permet de modifier les informations d'un partenaire dans la base de données.
Acteur	• Application front • Une autre api rest du projet Atlas
Pré condition	
Post condition	le Controller renvoie le code HTTP 201 CREATED et une représentation du partenaire créé avec son id et les liens hateoas pour accéder aux autres ressources de l'api

Scénario normal
• Le client applicatif spécifie l'id du partenaire dans l'URI de la requête ainsi les informations du partenaire en format JSON puis lance la requête avec la méthode HTTP put. • Si la requête est bien formulée et qu'on n'a aucune erreur au niveau du serveur, alors le service s'occupe ensuite de faire le mapping, renseigner la date et le status, puis de faire appel au repository pour sauvegarder les informations du partenaire.
Scénario Alternatif 1
• Le client applicatif spécifie l'id du partenaire dans l'URI de la requête ainsi les informations du partenaire en format JSON puis lance la requête avec la méthode HTTP put. • Le corps de la requête n'est pas valide • Le controller renvoie un message d'erreur 400 **Bad Request**
Scénario Alternatif 2
• Le client applicatif spécifie l'id du partenaire dans l'URI de la requête ainsi les informations du partenaire en format JSON puis lance la requête avec la méthode HTTP put. • La sauvegarde ne fonctionne pas à cause d'un problème coté serveur • Le controller renvoie un message d'erreur **500 Internal Server Error**

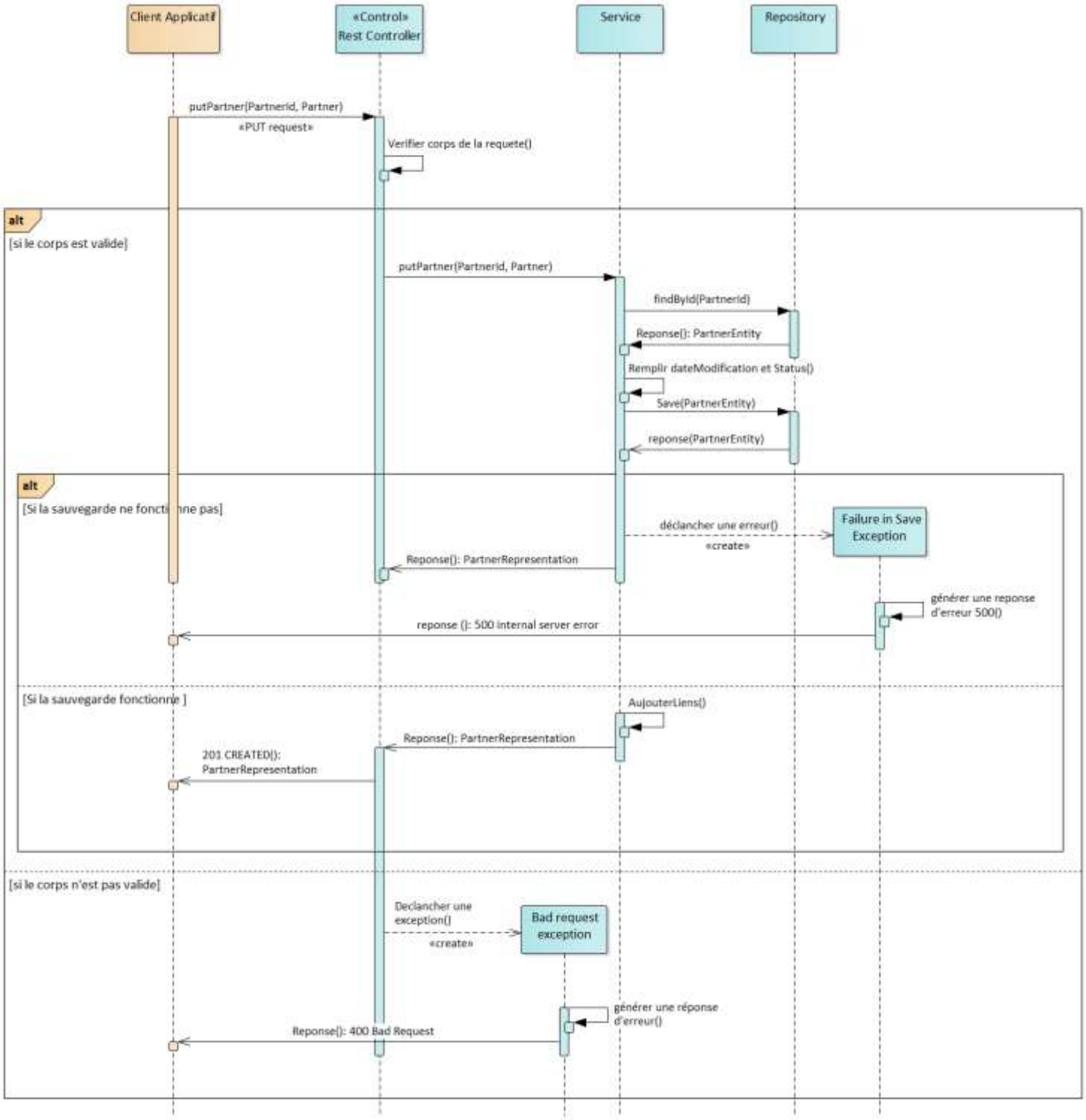


Figure 10 : Cas d'utilisation modifier un partenaire

Cas d'utilisation supprimer un partenaire

Lors de la modification d'un partenaire, le client applicatif utilise la ressource deletePartner en spécifiant l'id du partenaire dans l'URI de la requête avec la méthode http delete. Ainsi si la requête est bien formulée, que le partenaire existe et qu'on n'a aucune erreur au niveau du serveur, alors le service fait appel au repository pour supprimer le partenaire. Ensuite, le controller renvoie le code HTTP 204 NO CONTENT qui indique que la requête a réussie.

Par contre, en cas d'erreur :

L'exception FailureInSave sera déclenchée pour indiquer une erreur du SGBD.

Le code de réponse d'erreur serveur HTTP 500 **Internal Server Error** indiquera que le serveur a rencontré un problème inattendu qui l'empêche de répondre à la requête.

Le code de statut de réponse HTTP 400 **Bad Request** indiquera que le serveur ne peut pas comprendre ou traiter la requête en raison d'une erreur côté client.

Le code de statut de réponse HTTP 404 **Ressource Not Found** indiquera que le partenaire n'existe pas dans la base de données.

Description textuelle	
Nom du cas d'utilisation	Modifier un partenaire
Description	Ce cas d'utilisation permet de supprimer un partenaire de la base de données à partir de son id
Acteur	• Application front • Une autre api rest du projet Atlas
Pré condition	
Post condition	le Controller renvoie le code HTTP 204 No content qui indique que le partenaire a été supprimé avec succès
Scénario normal	
• Le client applicatif spécifie l'id du partenaire dans l'URI puis lance la requête avec la méthode HTTP delete	

• Si la requête est bien formulée, que le partenaire existe et qu'on n'a aucune erreur au niveau du serveur, alors le service fait appel au repository pour supprimer le partenaire.
Scénario Alternatif 1
• Le client applicatif spécifie l'id du partenaire dans l'URI puis lance la requête avec la méthode HTTP delete • Le corps de la requête n'est pas valide • Le controller renvoie un message d'erreur 400 **Bad Request**
Scénario Alternatif 2
• Le client applicatif spécifie l'id du partenaire dans l'URI puis lance la requête avec la méthode HTTP delete • La suppression ne fonctionne pas à cause d'un problème coté serveur • Le controller renvoie un message d'erreur **500 Internal Server Error**

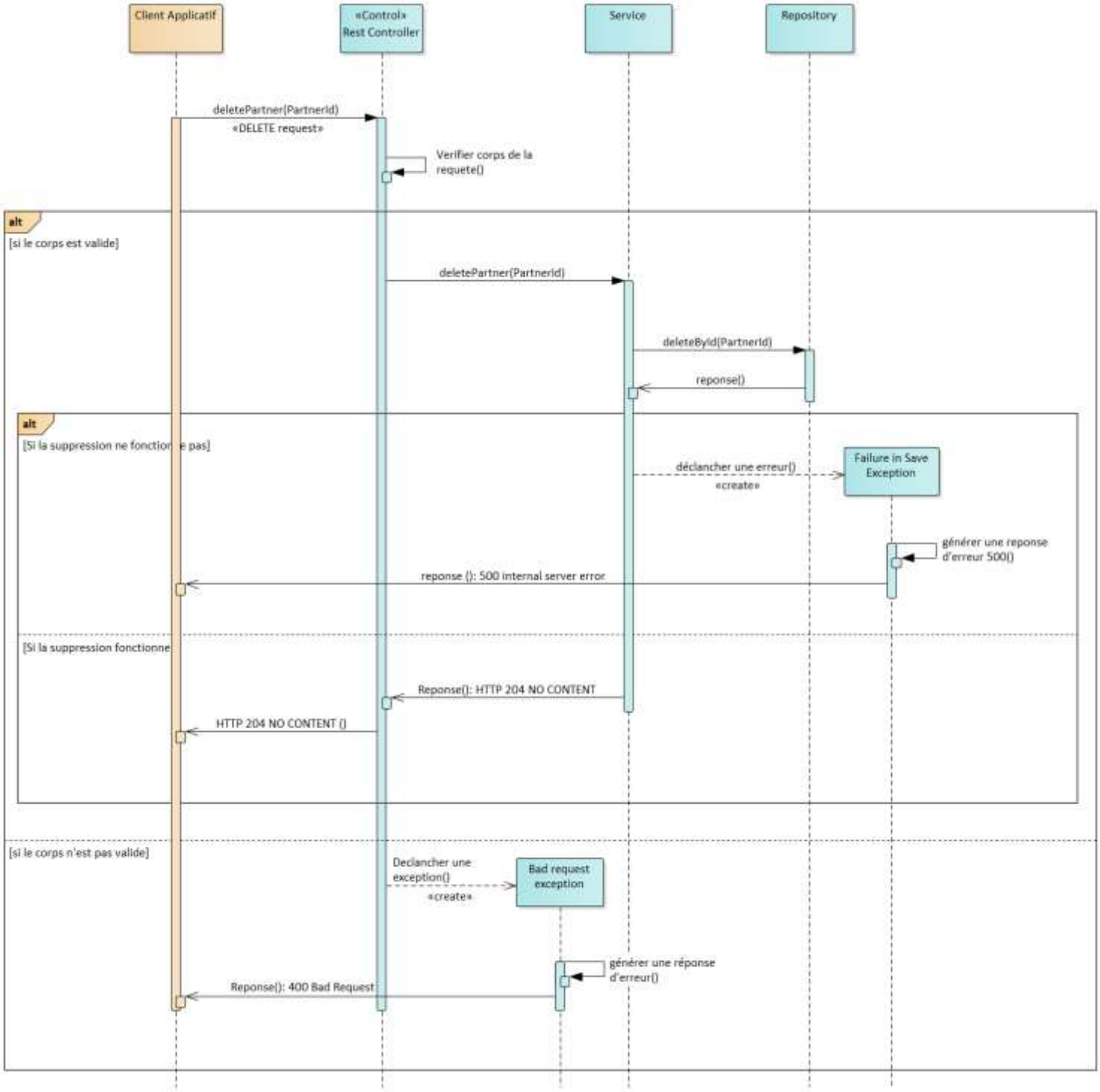


Figure 11 : Cas d'utilisation supprimer un partenaire

Cas d'utilisation afficher un partenaire

Lors de l'affichage d'un partenaire, le client applicatif utilise la ressource getPartner en spécifiant l'id du partenaire dans l'url de la requête avec la méthode HTTP get. Si la requête est bien formulée, que le partenaire existe et qu'on n'a aucune erreur au niveau du serveur, alors le service fera appel au repository pour récupérer les informations du partenaire. Ensuite, le controller renvoie le code HTTP 200 OK et une représentation du partenaire avec les liens hateoas pour accéder aux autres services de l'api.
Par contre, en cas d'erreur :
Le code de réponse d'erreur serveur HTTP 500 **Internal Server Error** indiquera que le serveur a rencontré un problème inattendu qui l'empêche de répondre à la requête.
Le code de statut de réponse HTTP 400 **Bad Request** indiquera que le serveur ne peut pas comprendre ou traiter la requête en raison d'une erreur côté client.
Le code de statut de réponse HTTP 404 **Ressource Not Found** indiquera que le partenaire n'existe pas dans la base de données.

Description textuelle	
Nom du cas d'utilisation	Modifier un partenaire
Description	Ce cas d'utilisation permet l'affichage d'un partenaire à partir de son id.
Acteur	• Application front • Une autre api rest du projet Atlas
Pré condition	
Post condition	le Controller renvoie le code HTTP 200 OK qui indique que le partenaire a été affiché avec succès
Scénario normal	
• Le client applicatif spécifie l'id du partenaire dans l'URI puis lance la requête avec la méthode HTTP get. • Si la requête est bien formulée, que le partenaire existe et qu'on n'a aucune erreur au niveau du serveur, alors le service fera appel	

au repository pour récupérer les informations du partenaire et afficher.
Scénario Alternatif 1
• Le client applicatif spécifie l'id du partenaire dans l'URI puis lance la requête avec la méthode HTTP get. • Si le corps de la requête n'est pas valide ou s'il y a une erreur coté serveur • Le controller renvoie un message d'erreur
Scénario Alternatif 2
• Le client applicatif spécifie l'id du partenaire dans l'URI puis lance la requête avec la méthode HTTP get. • Le partenaire est introuvable dans la base de données • Le controller renvoie un message d'erreur 404 **Ressource Not Found**

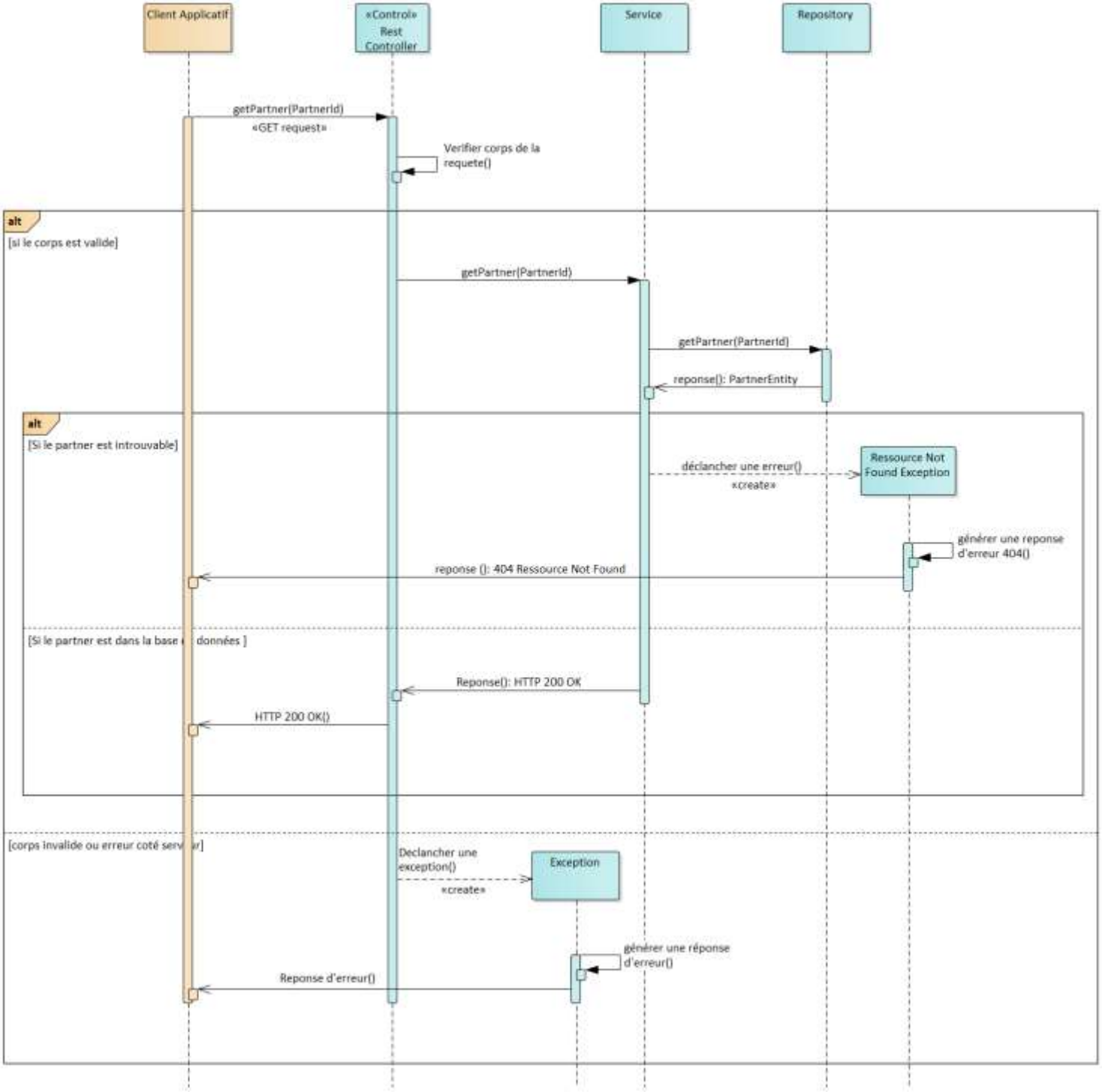


Figure 12 : Cas d'utilisation afficher un partenaire

Cas d'utilisation afficher la liste des partenaires

Pour afficher la liste des partenaires, le client applicatif utilise la ressource listPartners. Nous avons trois cas possibles :

- S'il spécifie dans l'URI le numéro de page et le nombre de partenaires par page avec la méthode http get. Si la requête est bien formulée, qu'on n'a aucune erreur au niveau du serveur, alors le service fera appel au repository pour récupérer les informations. Ensuite, le controller renvoie le code HTTP 200 OK et un objet BaseCollection qui contient les liens hateoas vers les différents partenaires.
- S'il ne spécifie pas dans l'URI le numéro de page et le nombre de partenaires par page. Alors dans ce cas, nous n'aurons pas de pagination. Le controller renvoie le code HTTP 200 OK et un objet BaseCollection qui contient la liste des liens hateoas vers tous les partenaires.
- S'il spécifie dans l'URI le numéro de page et pas le nombre de partenaires par page ou inversement, alors une erreur de type MissingRequestParameterException sera déclenchée.

En cas d'erreur :
Le code de réponse d'erreur serveur HTTP 500 **Internal Server Error** indiquera que le serveur a rencontré un problème inattendu qui l'empêche de répondre à la requête.
Le code de statut de réponse HTTP 400 **Bad Request** indiquera que le serveur ne peut pas comprendre ou traiter la requête en raison d'une erreur côté client.

Description Textuelle	
Nom du cas d'utilisation	Afficher la liste des partenaires
Description	Ce cas d'utilisation permet l'affichage de la liste des partenaires.
Acteur	• Application front • Une autre api rest du projet Atlas
Pré condition	
Post condition	le Controller renvoie le code HTTP 200 OK qui indique que la liste a été affichée avec succès

Scénario normal
• Le client applicatif spécifie le numéro de la page et le nombre de partenaires par page dans l'URI puis lance la requête avec la méthode HTTP get. • Si la requête est bien formulée et qu'on a aucune erreur au niveau du serveur, alors le service fera appel au repository pour récupérer les informations. • Ensuite, le controller renvoie le code HTTP 200 OK et un objet BaseCollection qui contient les liens hateoas vers les différents partenaires.
Scénario Alternatif 1
• Le client applicatif ne spécifie pas dans l'URI le numéro de page et le nombre de partenaires par page. Alors dans ce cas, nous n'aurons pas de pagination. • Le controller renvoie le code HTTP 200 OK et un objet BaseCollection qui contient la liste des liens hateoas vers tous les partenaires.
Scénario Alternatif 2
• Le client applicatif spécifie dans l'URI le numéro de page et pas le nombre de partenaires par page ou inversement, alors une erreur de type MissingRequestParameterException sera déclenchée. • Le controller renvoie ensuite un message d'erreur
Scénario Alternatif 3
• Le client applicatif lance la requête d'affichage de la liste avec la méthode http get. • Si le corps de la requête n'est pas valide ou s'il y a une erreur coté serveur • Le controller renvoie un message d'erreur

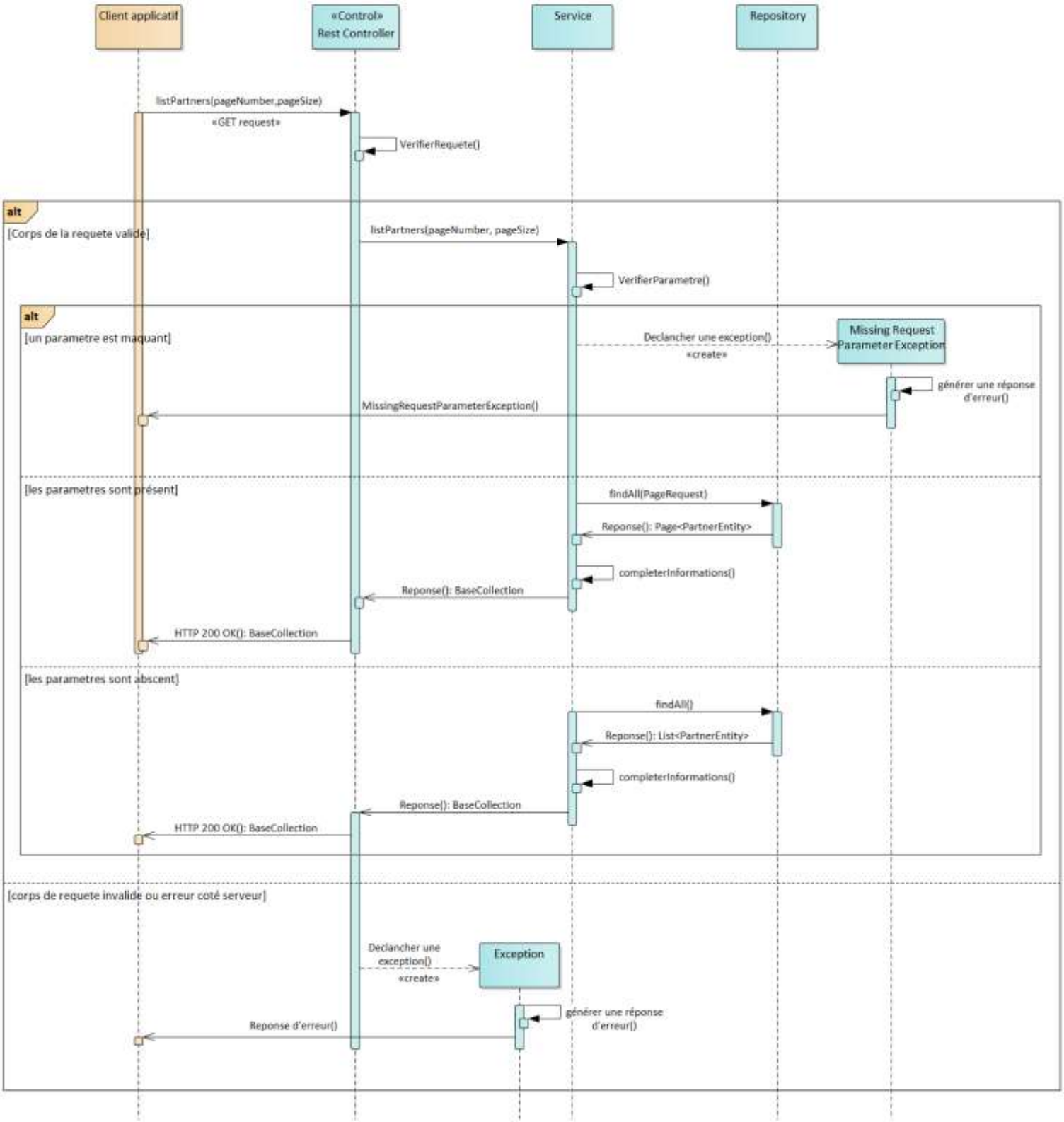


Figure 13 : liste des partenaires

3. Diagramme de classe

En génie logiciel, un diagramme de classes dans le langage de modélisation unifié (UML) est un type de diagramme de structure statique qui décrit la structure d'un système en montrant les classes du système, leurs attributs, opérations (ou méthodes) et les relations entre les objets.
Notre diagramme est constitué de 3 classes,
La classe principale « partner » contient les informations du partenaire telles que L'id, le nom, la forme juridique, le status…
La classe « address » contient les informations sur l'adresse du partenaire et la classe « companyInformation » contient des informations sur l'entreprise.
Entre partner et address, on a une agrégation et entre partner et companyInformation, une relation un à plusieurs.

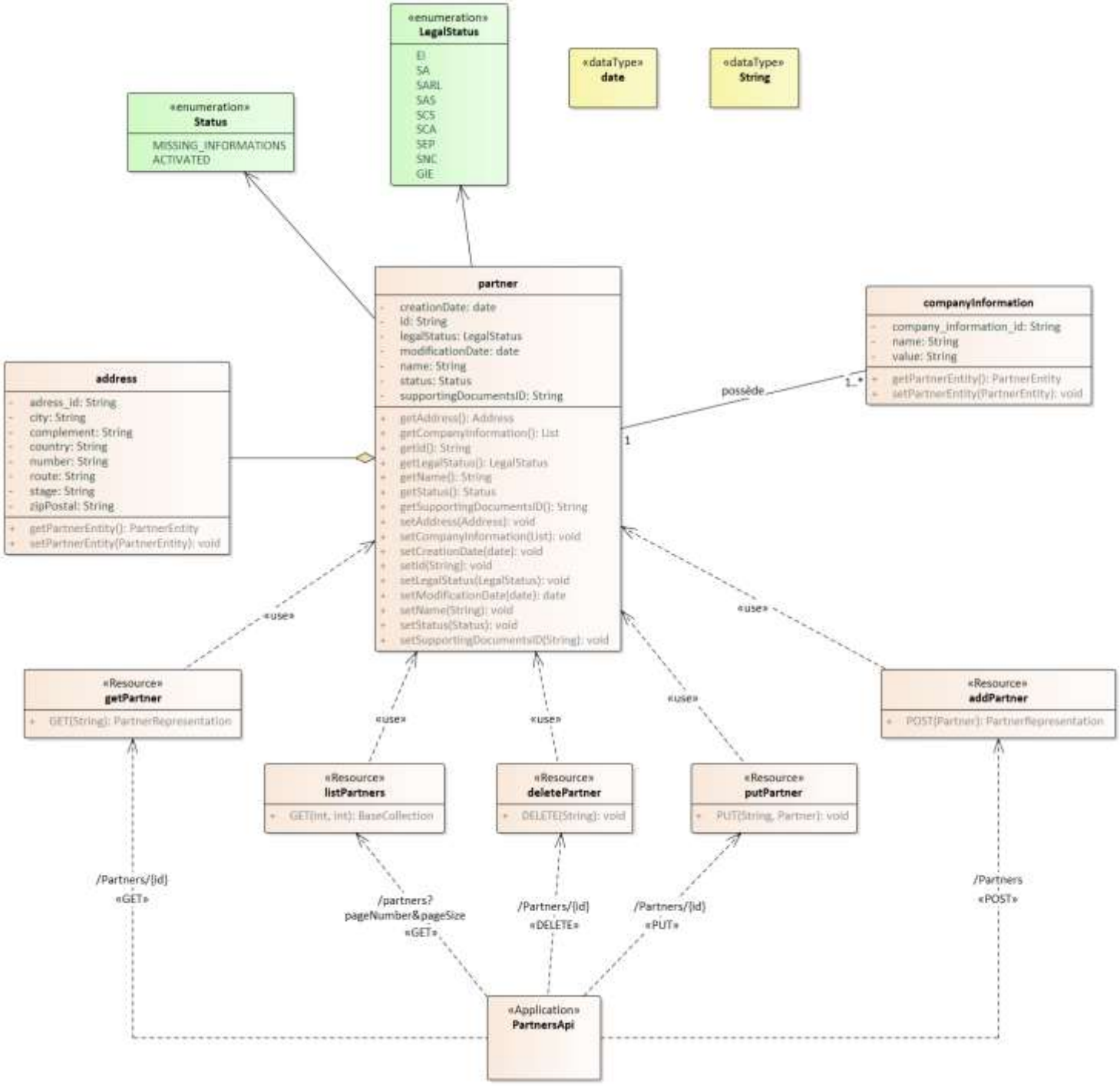


Figure 14 : Diagramme de classe

CHAPITRE III
RÉALISATION

1. Structure de l'api réalisée

J'ai opté pour une approche en couches qui permet une meilleure évolution et une meilleure maintenabilité du code.

- **Couche Controller** : gestion des interactions entre le client de l'api et l'api ;
- **Couche Service** : implémentation des traitements métiers spécifiques à l'api ;
- **Couche Repository** : interaction avec les sources de données externes ;
- **Couche Model** : implémentation des objets métiers qui seront manipulés par les autres couches ;
- **Couche Utils** : contient les mappers, les liens Hateoas, et les exceptions.

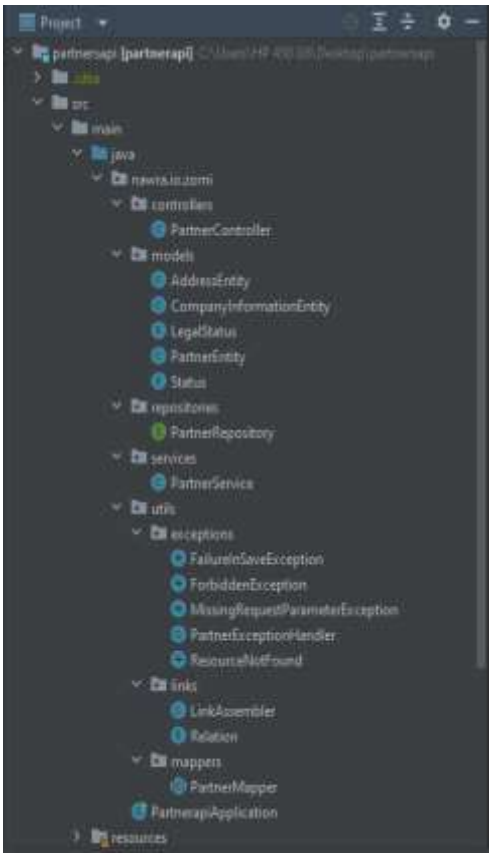


Figure 15 : Arborescence du projet

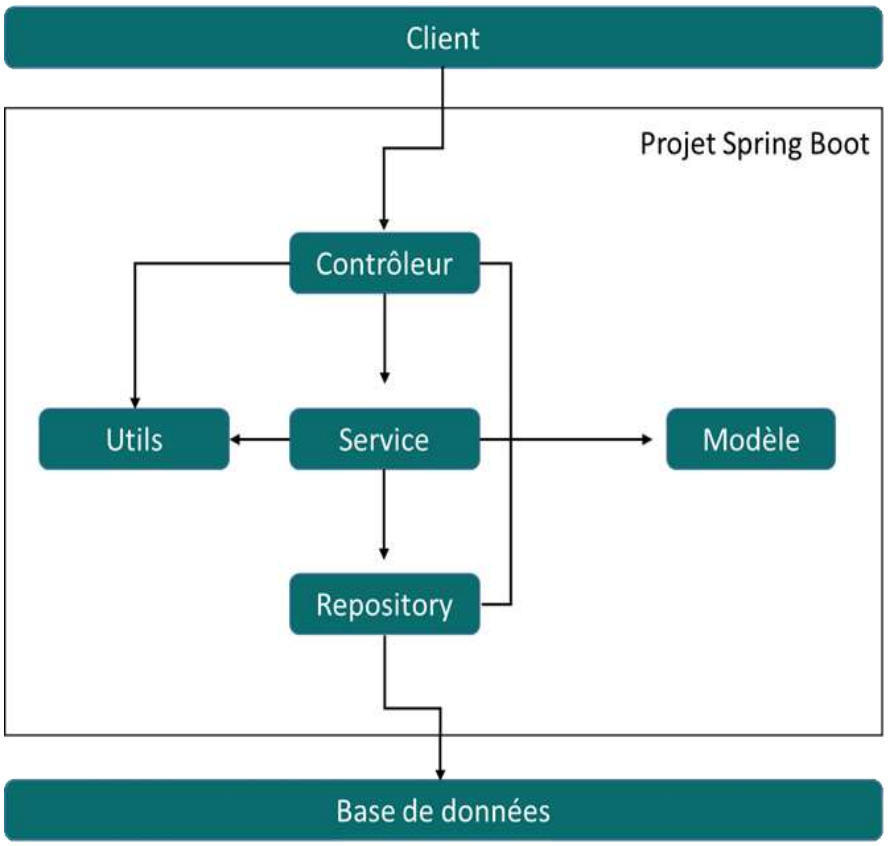


Figure 16 : Architecture de l'api

2. Outils de développement

a. Java

J’ai développé mon api en langage java. Java est un langage de programmation orienté objet et une plateforme logicielle largement utilisés qui s'exécutent sur des milliards d'appareils, notamment des ordinateurs portables, des appareils mobiles, des consoles de jeu, des appareils médicaux et bien d'autres. Les règles et la syntaxe de Java sont basées sur les langages C et C++.

Figure 17 : java

b.Spring boot

Java Spring Framework (Spring Framework) est un cadre d'entreprise populaire et open source pour la création d'applications autonomes de niveau production qui s'exécutent sur la machine virtuelle Java (JVM).

Spring Boot est un framework de développement JAVA. C'est une déclinaison du framework classique de Spring qui permet essentiellement de réaliser des microservices (ce sont la majeure partie du temps des services web qui sont regroupés en API). Les principaux avantages de spring boot sont :

- L’auto-configuration
- L’optimisation de la gestion des dépendances
- la gestion des propriétés
- le monitoring et la gestion du programme
- la simplicité du déploiement

Figure 18 : Spring Boot

Dans mon api, j'ai utilisé les starters de dépendances suivantes :

- **Spring Boot starter web** permet de fournir des dépendances qui nous permettent de faire du Restful, et d'exposer des endpoints;
- **Spring Boot starter Data JPA** est un framework ORM performant qui nous permet d'interagir avec une base de données ;
- **Spring Boot starter Actuator** correspond à une fonctionnalité de Spring Boot qui permet de monitorer et de manager notre programme pendant qu'il est en cours d'exécution ;
- **Spring Boot starter hateoas** starter pour créer une application Web RESTful basée sur l'hypermédia avec Spring MVC et Spring HATEOAS.

```
<dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
<dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
<dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-web</artifactId>
</dependency>
   <dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-hateoas</artifactId>
</dependency>
```

C. Maven

Apache Maven (couramment appelé Maven) est un outil de gestion et d'automatisation de production des projets logiciels Java en général et Java EE en particulier. Je l'ai utilisé pour automatiser l'intégration continue lors du développement.
Il offre entre autres les fonctionnalités suivantes :

- Compilation et déploiement des applications Java (JAR, WAR)
- Gestion des librairies requises par l'application
- Exécution des tests unitaires
- Génération des documentations du projet (site web, pdf, Latex)
- Intégration dans différents IDE (Eclipse, JBulder).

Figure 19 : Maven

d. Enterprise Architect

Pour créer les différents diagrammes UML de mon projet, j'ai utilisé Enterprise Architect. Enterprise Architect est un logiciel de modélisation et de conception UML, édité par la société australienne Sparx Systems. Couvrant, par ses fonctionnalités, l'ensemble des étapes du cycle de conception d'application, il est l'un des logiciels de conception et de modélisation les plus reconnus. Enterprise Architect permet la modélisation de données depuis le concept jusqu'aux niveaux physiques, l'ingénierie et la rétroingénierie des schémas de bases de données ainsi que la transformation du modèle vers les bases de données physiques, dépendantes de la plateforme sur laquelle elles sont installées.

Figure 20 Enterprise Architect

e. IntelliJ IDEA

J'ai utilisé l'environnement de développement IntelliJ IDEA pour développer mon api rest. IntelliJ IDEA est un IDE intelligent et sensible au contexte pour travailler avec Java et d'autres langages JVM comme Kotlin, Scala et Groovy sur toutes sortes d'applications. De plus, IntelliJ IDEA Ultimate peut vous aider à développer des applications Web complètes, grâce à ses puissants outils intégrés, à la prise en charge de JavaScript et des technologies associées, et à la prise en charge avancée des frameworks populaires tels que Spring, Spring Boot, Jakarta EE, Micronaut, Quarkus, Helidon.

Figure 21 : IntelliJ IDEA

f. PostgreSQL

J’ai utilisé le SGBD PostgreSQL pour la gestion de ma base de données. PostgreSQL est un système de gestion de base de données relationnelle orienté objet puissant et open source qui est capable de prendre en charge en toute sécurité les charges de travail de données les plus complexes. Alors que MySQL donne la priorité à l'évolutivité et aux performances, Postgres donne la priorité à la conformité et à l'extensibilité SQL.

Figure 22 PostgreSQL

g. Git

J’ai utilisé Git qui est un logiciel de gestion de versions décentralisé pour gérer les différentes versions de mon projet et les stockées dans un dépôt distant. Git est de loin le système de contrôle de version le plus largement utilisé aujourd'hui. Git est un projet open source avancé, qui est activement maintenu. À l'origine, il a été développé en 2005 par Linus Torvalds, le créateur bien connu du noyau du système d'exploitation Linux.

Figure 23 : git

h.Postman

Postman est un logiciel qui m’a permis d’appeler / tester mon api. Il permet de construire et d’exécuter des requêtes HTTP, de les stocker dans un historique afin de pouvoir les rejouer, mais surtout de les organiser en Collections. Cette classification permet notamment de regrouper des requêtes de façon « fonctionnelle » (par exemple enchaînement d’ajout d’item au panier, ou bien un processus d’identification).

Figure 24 Postman

i. MindView

Pour concevoir le diagramme WBS du rapport, j'ai utilisé MindView. C'est un logiciel de cartographie mentale et de gestion de projet appartenant à la société MatchWare. MindView est utilisé pour la cartographie mentale, la cartographie conceptuelle, les structures de répartition du travail, les chronologies, les diagrammes de Gantt, les organigrammes et d'autres visuels.

Figure 25 : MindView

j. GanttProject

Pour concevoir le diagramme de gantt du rapport, j'ai utilisé GanttProject. C'est un logiciel libre de gestion de projet écrit en Java, ce qui permet de l'utiliser sur divers systèmes d'exploitation (Windows, Linux, MacOS). Il permet d'éditer un diagramme de Gantt.

Figure 26 : GanttProject

3. Tests unitaires

En développement, les tests visent à vérifier que le produit codé fonctionne comme prévu selon des scénarios prédéfinis et représentatifs. Cela permet de garantir la qualité de ce qui est codé, malgré les contraintes du projet, comme les délais, par exemple.
J'ai effectué trois tests unitaires, la méthode testListPartners () permet de vérifier si le status de réponse HTTP est « 200 OK » lors de l'affichage de la liste des partenaires, la méthode testGetPartner() permet de vérifier si le status de réponse HTTP est « 200 OK » lors de l'affichage d'un partenaire par son id, et la méthode testDeletePartner() permet de vérifier si le status de réponse HTTP est « 204 NO CONTENT » lors de la suppression d'un partenaire par son id.

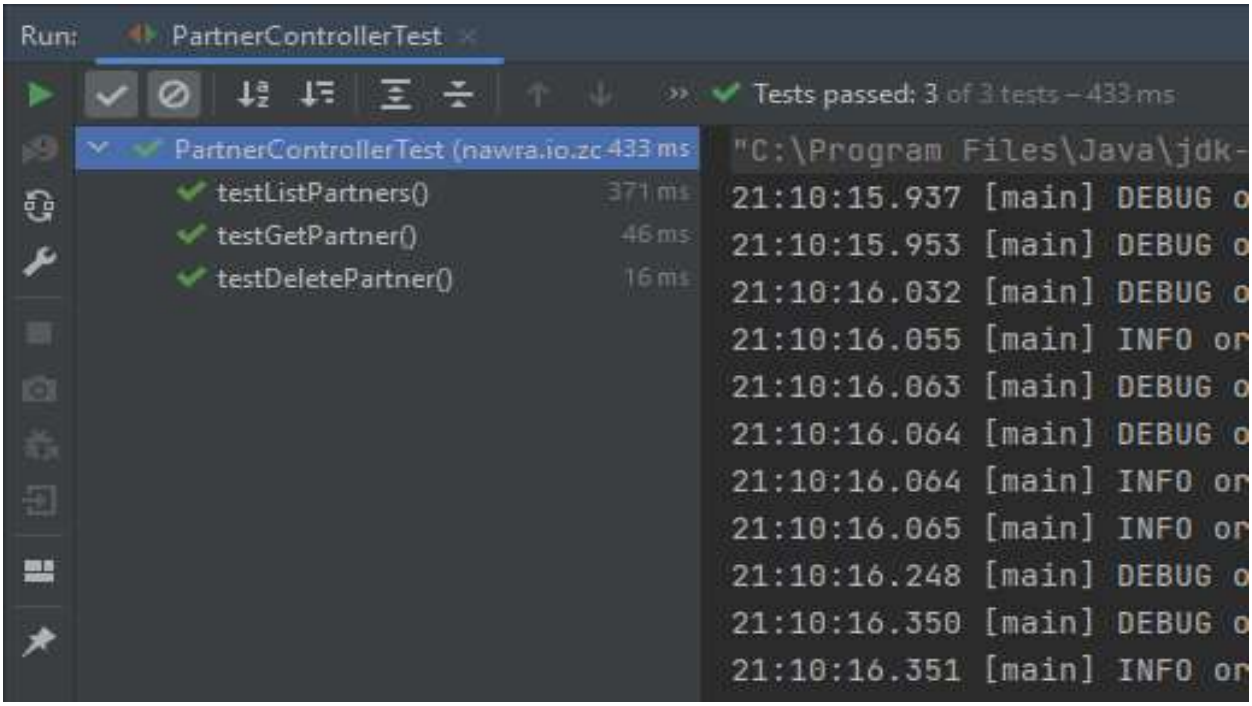


Figure 27 : test réussi

4. Présentation de l'api réalisée

Pour présenter les fonctionnalités de l'api, nous utiliserons le logiciel postman.

a. Endpoint addPartner

Pour ajouter un nouveau partenaire dans la base de donnée, on spécifie les informations du nouveau partenaire en format JSON puis on lance la requête avec la méthode HTTP post.

Figure 28 : ajouter un partenaire

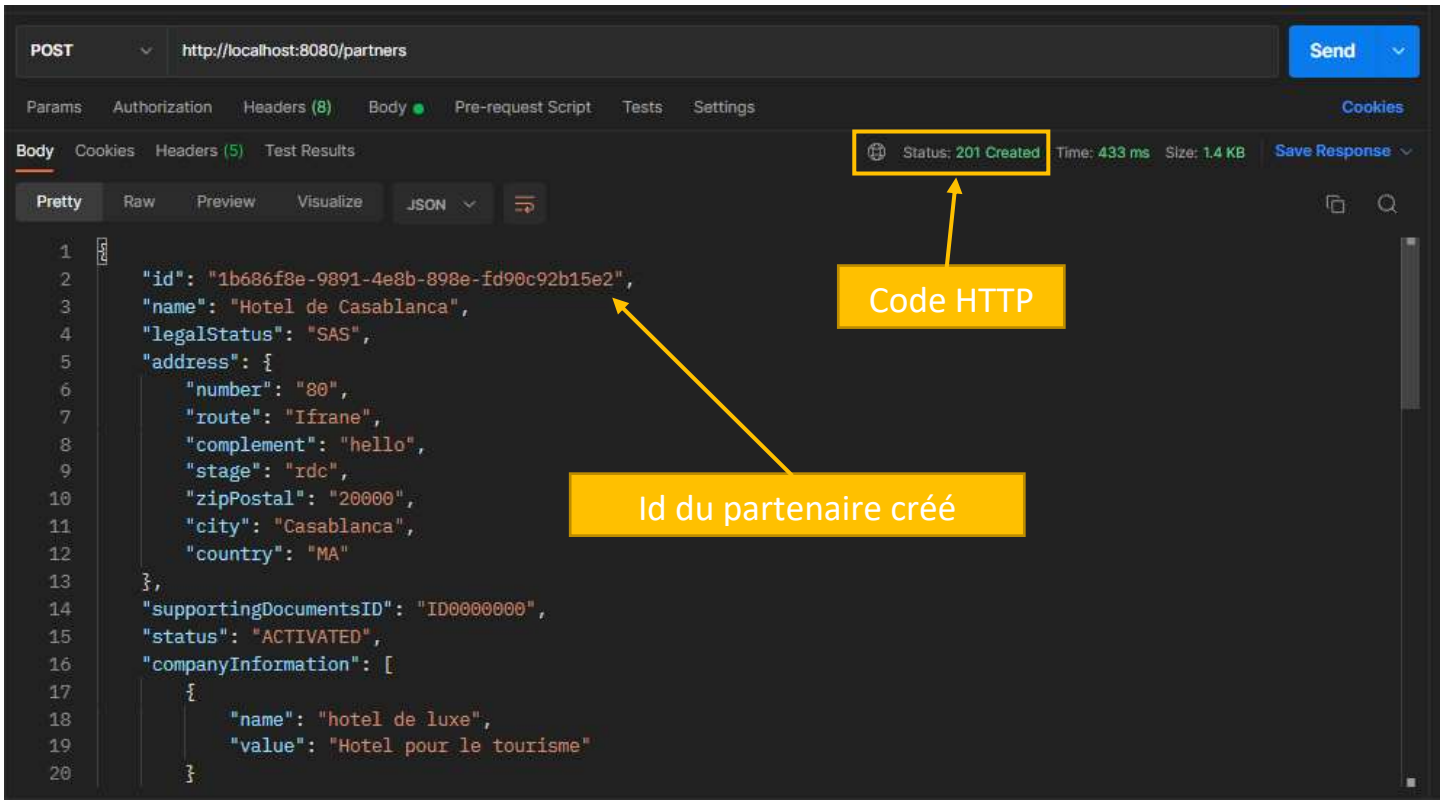


Figure 29 Le partenaire est créé

Lorsqu'on lance une requête sans spécifier les propriétés legalStatus et number, on a une erreur qui se déclenche.

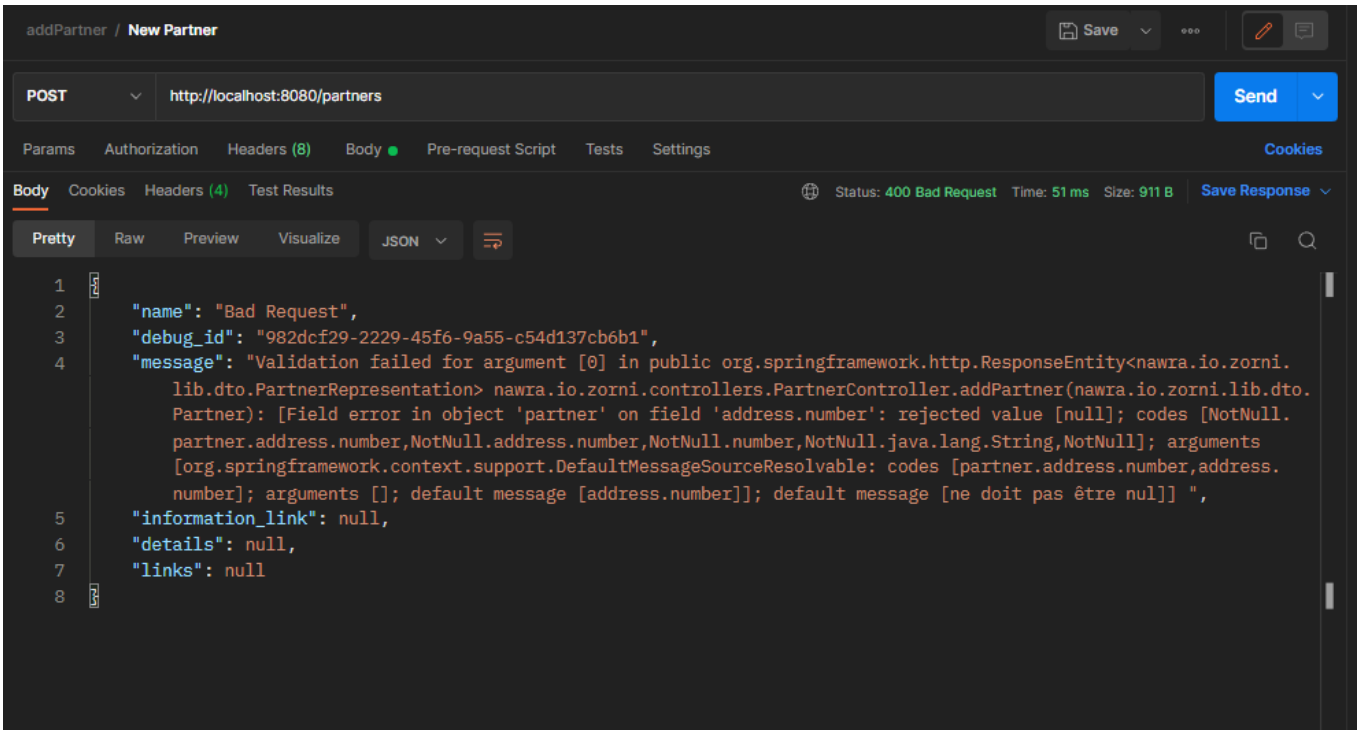


Figure 30 bad request exception

b. Endpoint listPartners

On spécifie le numéro de la page (pageNumber) et le nombre de partenaires par page (pageSize) dans l'URI puis on lance la requête avec la méthode http get.

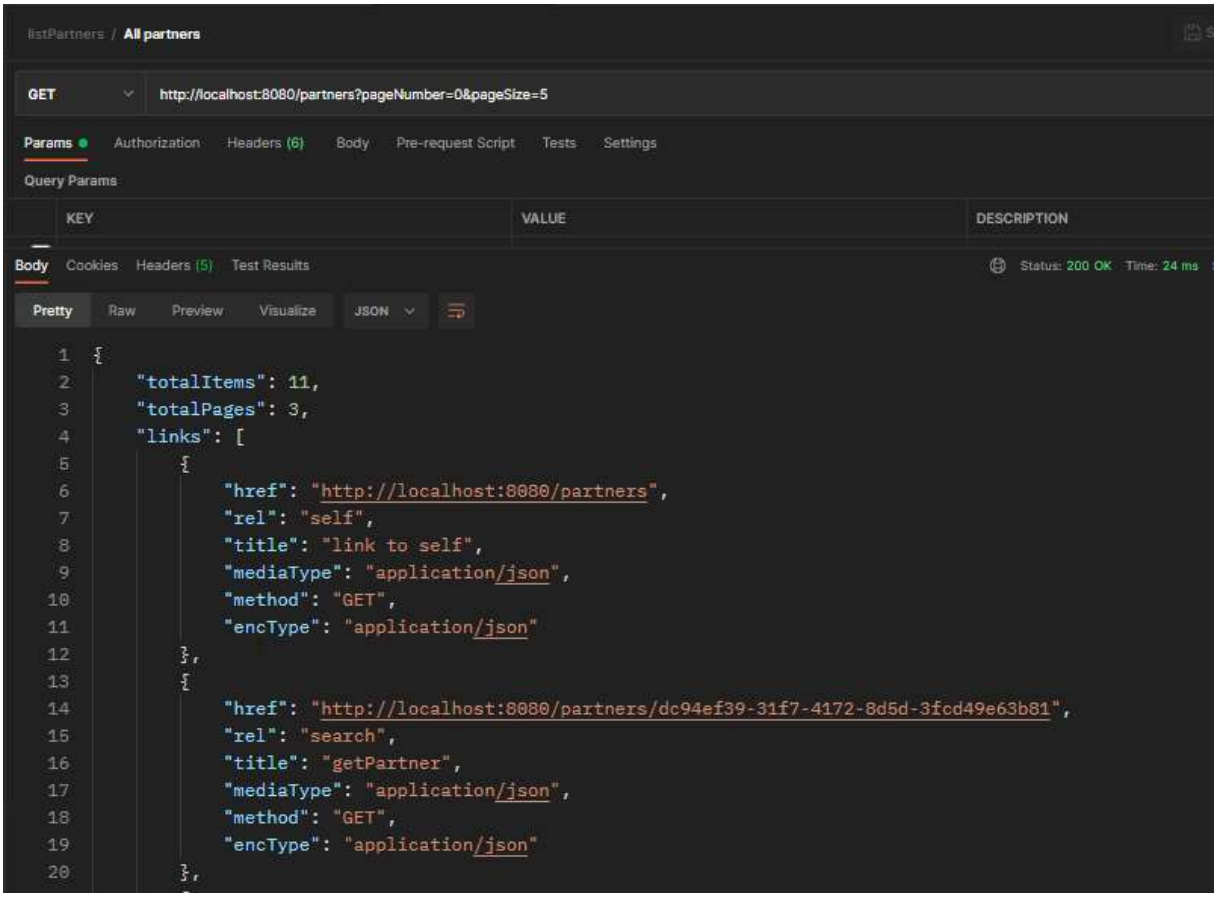
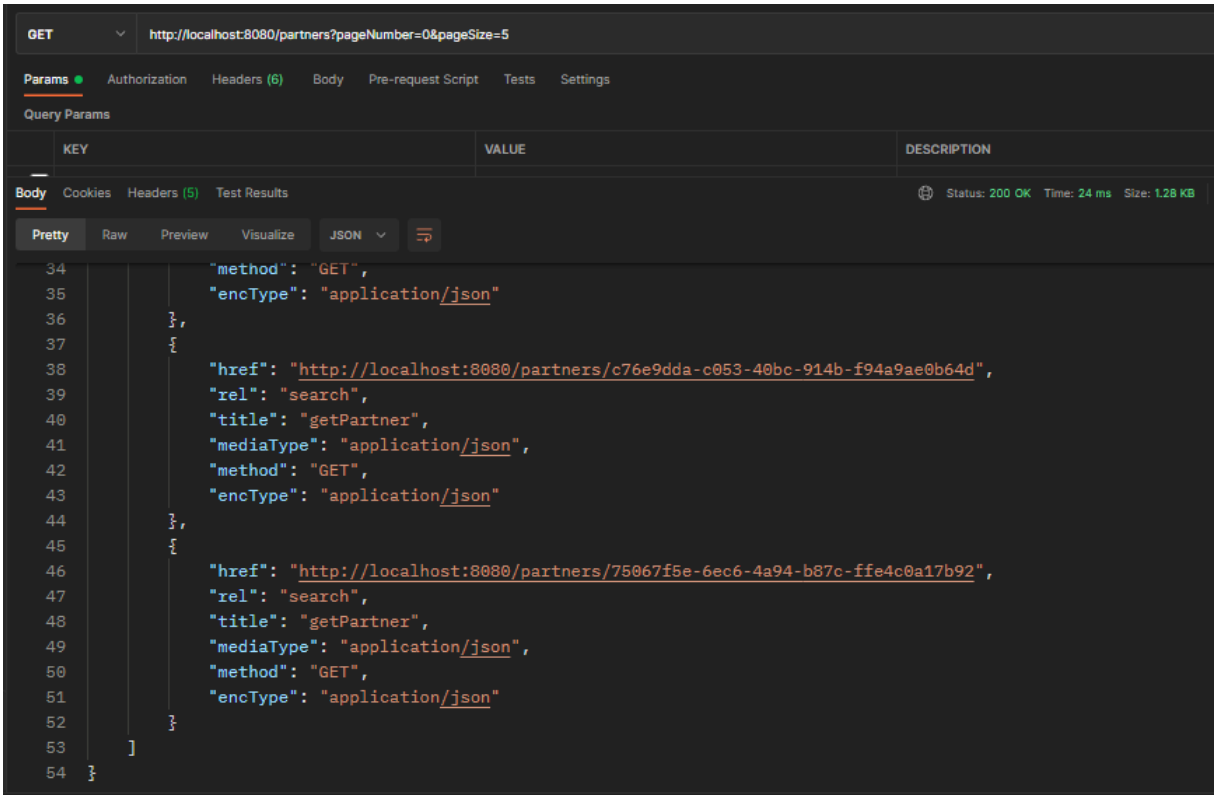


Figure 31 : Liste des partenaires



C. Endpoint getPartner

On spécifie l'id du partenaire dans l'URI puis on lance la requête avec la méthode http get.

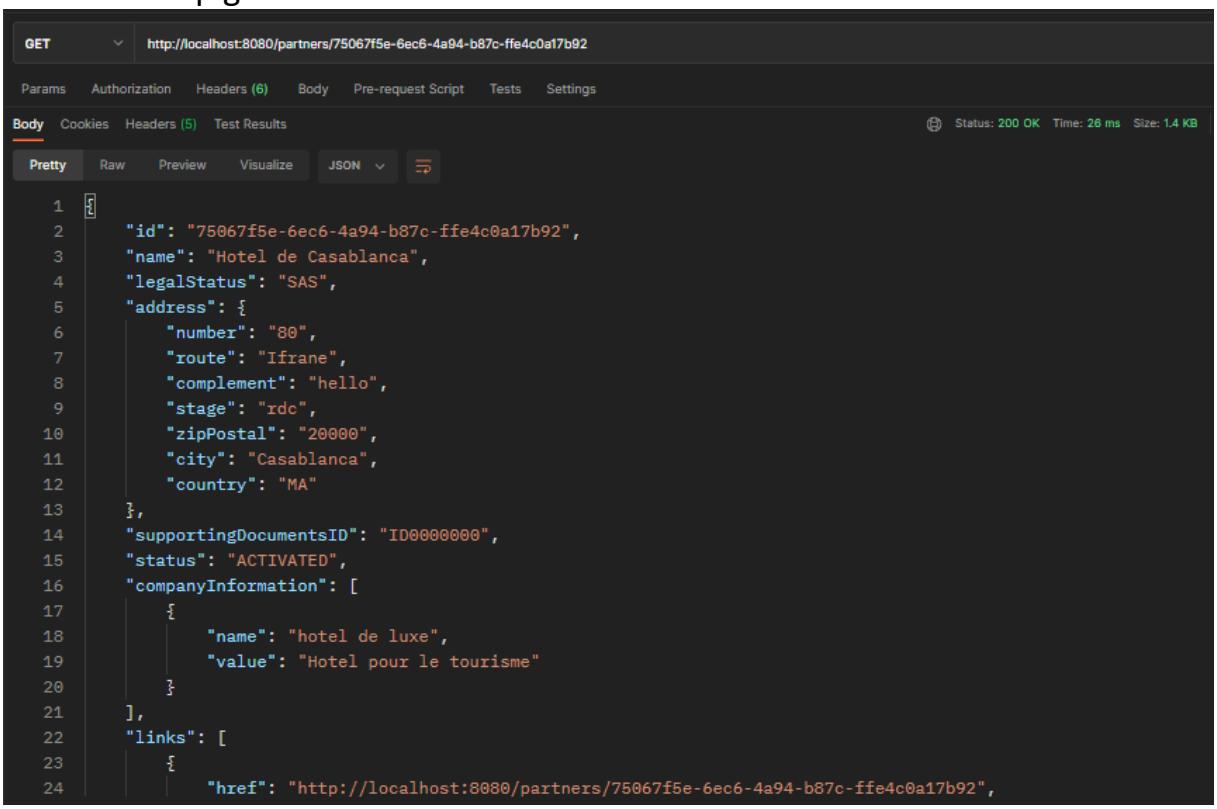


Figure 32 afficher un partenaire

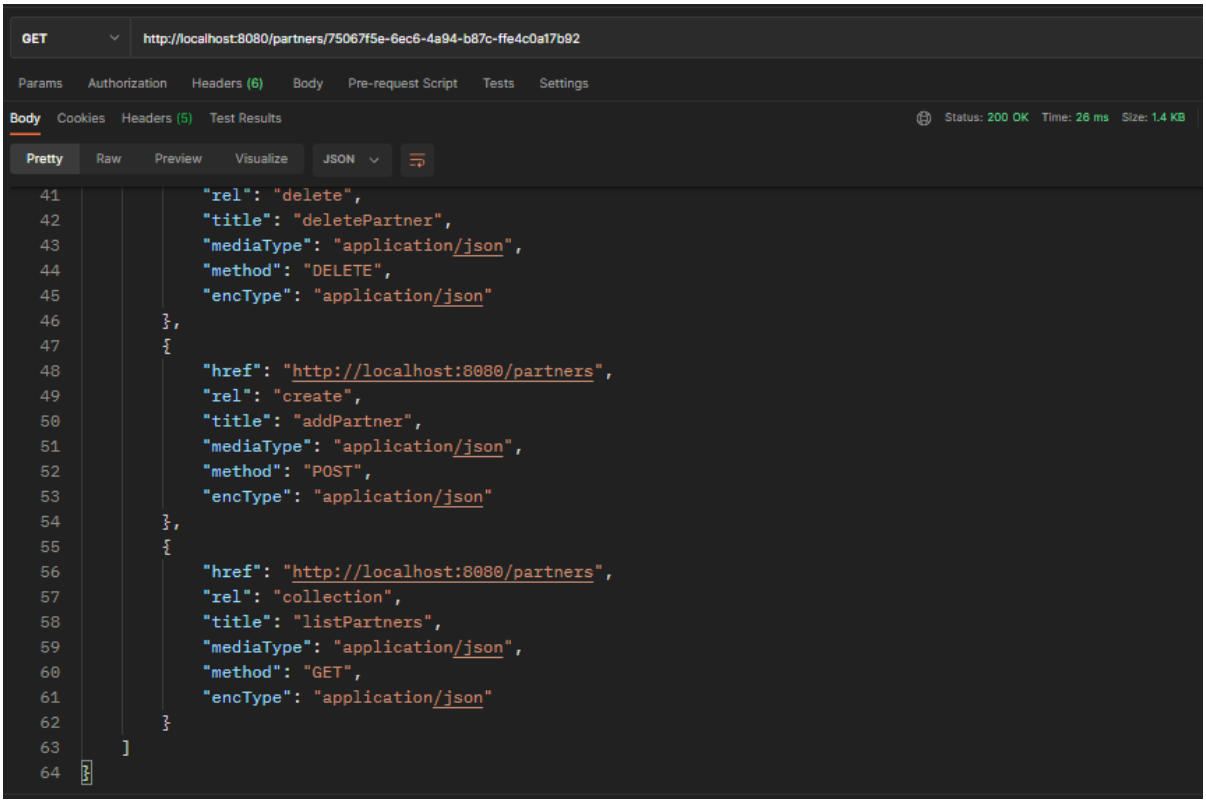


Figure 33 : Liens Hateoas

d. Endpoint deletePartner

On spécifie l'id du partenaire dans l'URI puis lance la requête avec la méthode HTTP delete.

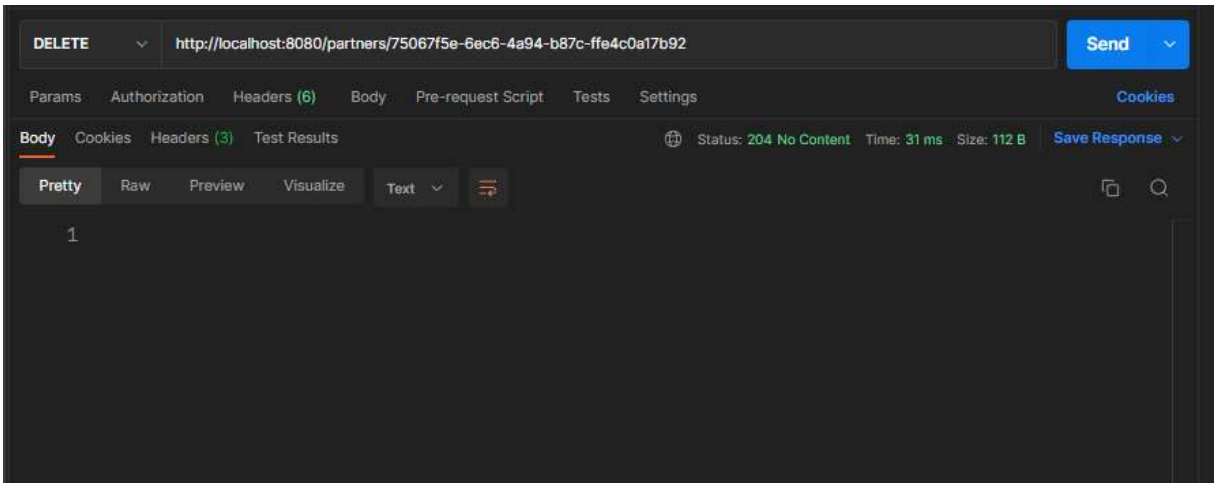


Figure 34 : supprimer un partenaire

Puisque le partenaire est supprimé, si on essaie de l'afficher, on obtient une erreur Ressource not found.

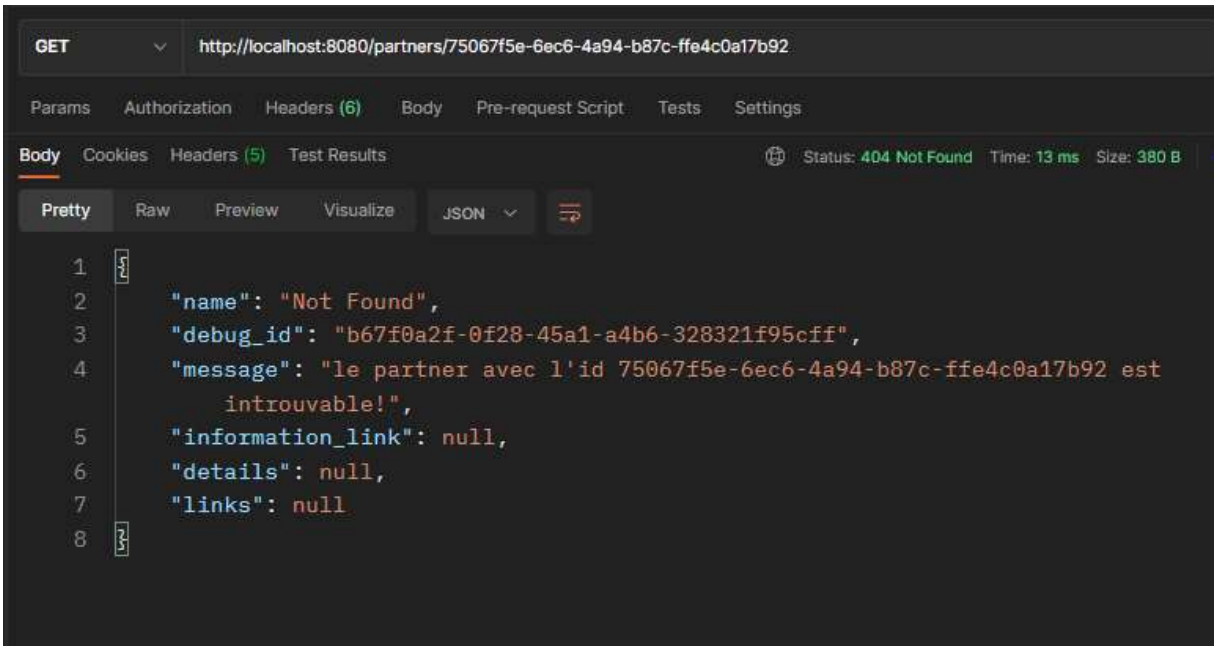
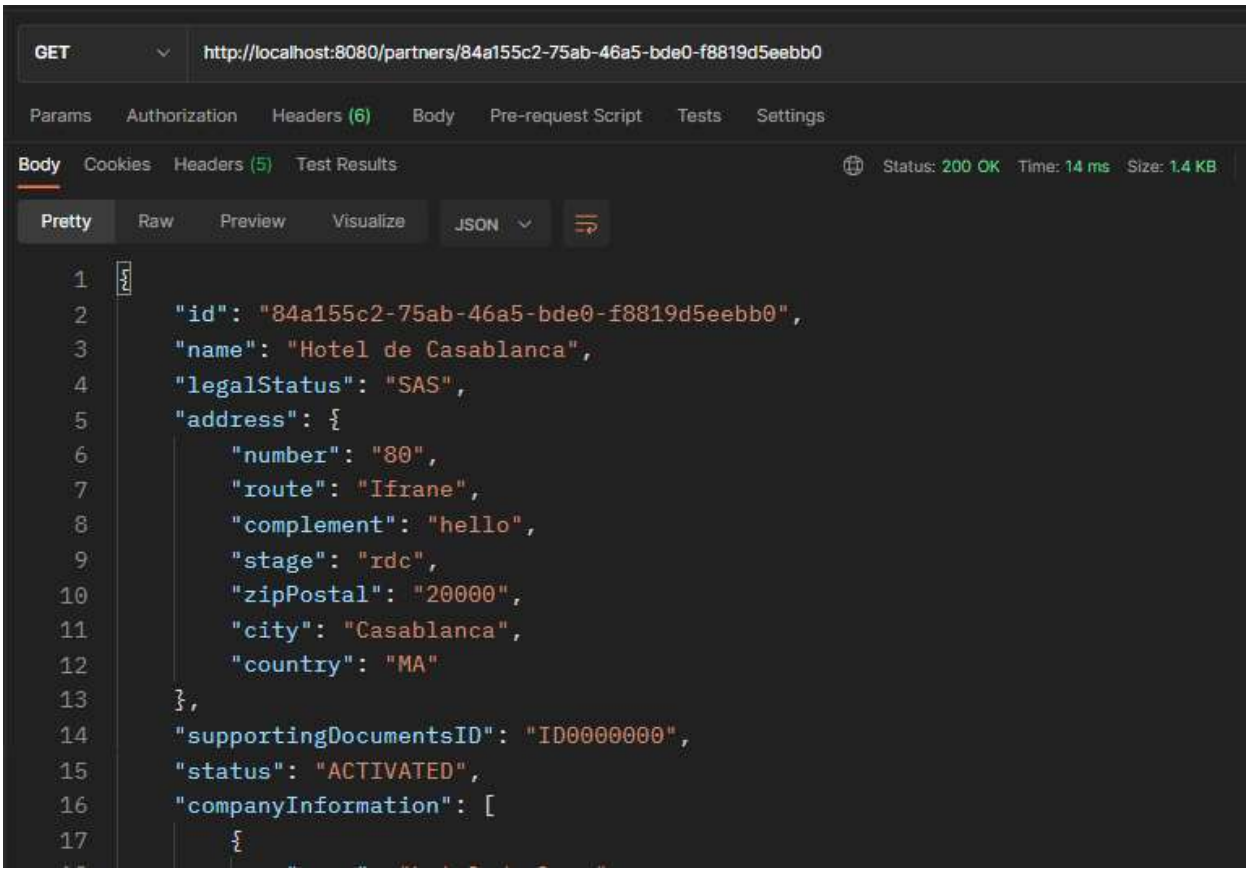


Figure 35 Ressource not found exception

e. Endpoint putPartner

On spécifie l'id du partenaire dans l'URI de la requête ainsi que le request body contenant les informations du partenaire en format JSON, puis on lance la requête.

- J'affiche les informations du partenaire



- J’écris les nouvelles informations du partenaire

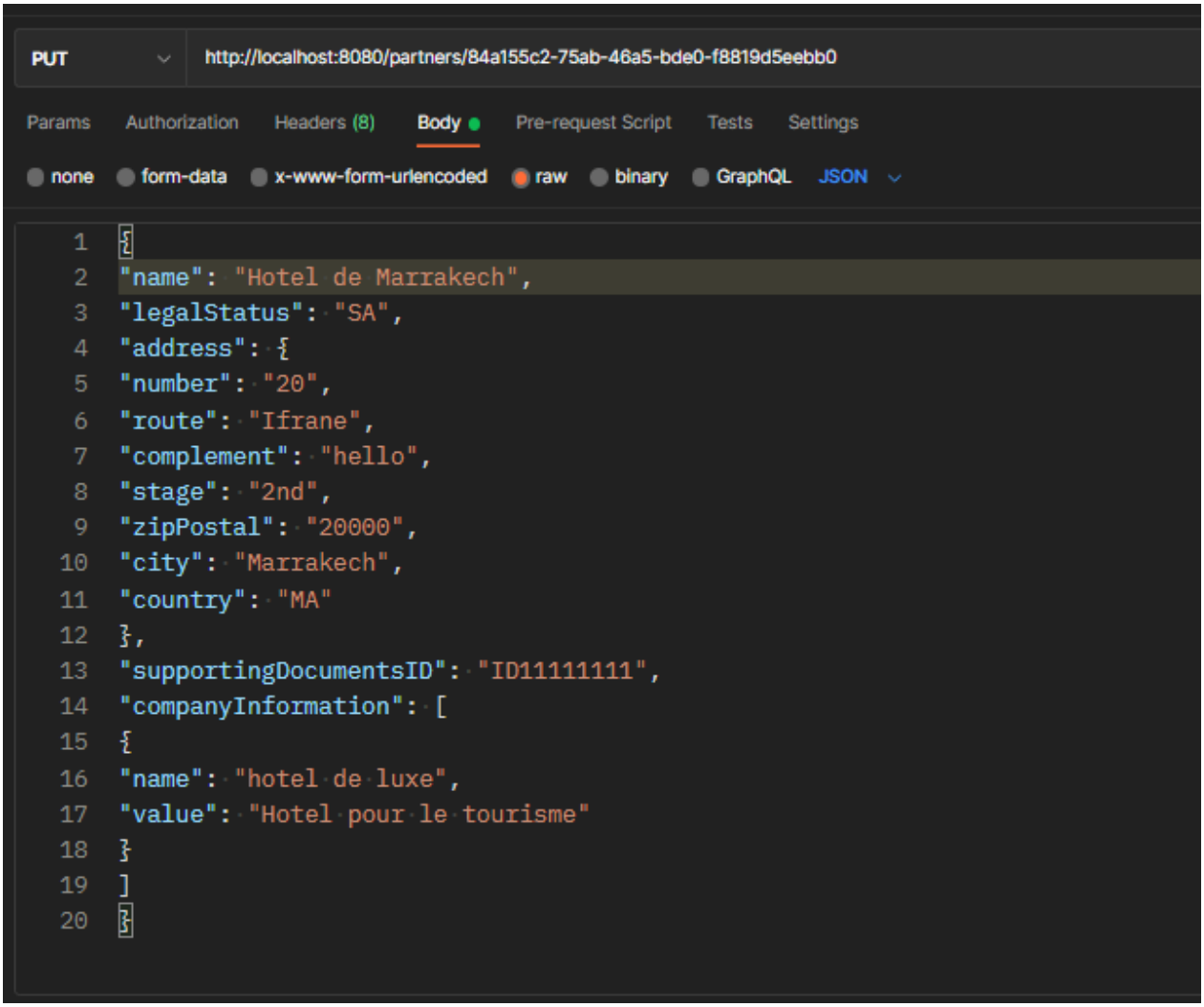


Figure 36 modifier un partenaire

- Je modifie le partenaire

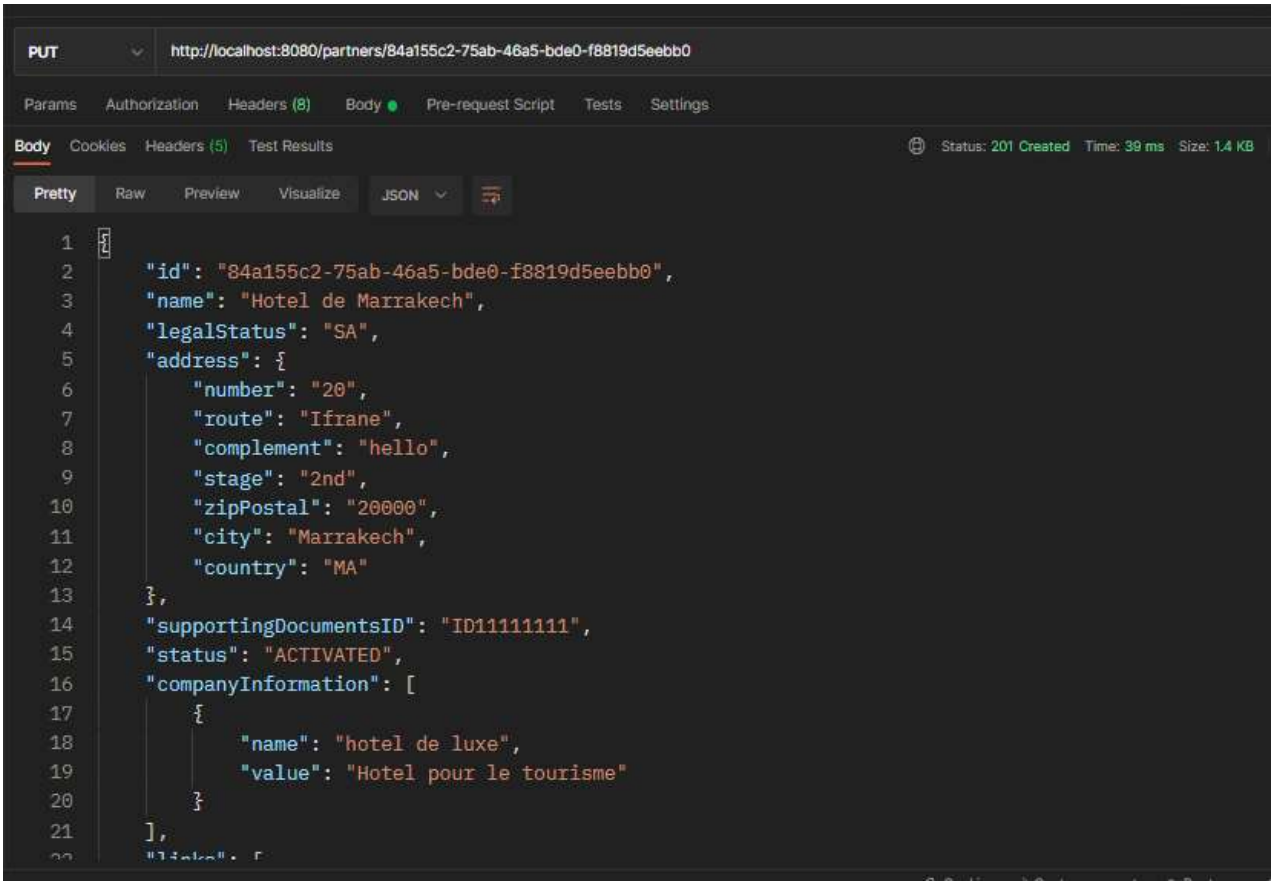


Figure 37 : Le partenaire est modifié

CONCLUSION ET PERSPECTIVES

Ce stage a été une première expérience très enrichissant pour moi, car il m'a permis de découvrir le secteur du développement informatique. Lors de ce stage de 2 mois au sein de la société Nawra Technology, j'ai pu mettre en pratique les connaissances théoriques acquises durant ma formation de licence, de plus, je me suis confronté aux difficultés réelles des projets dans le monde professionnel.

À première vue, on pourrait croire, vu le nombre de besoins fonctionnels que l'api était simple à développer. Cependant, travaillé avec le framework spring boot n'était pas une tâche aisée, car les principes de base de ce framework sont complexes et nécessite beaucoup de travail et de temps pour être maitrisé. De plus, il fallait mettre en place des classes de mapping entre entity et dto, des classes pour la création de liens hateoas, des classes pour gérer les exceptions et beaucoup d'autres fonctionnalités. Malgré l'intervalle de temps qui était très réduit, j'ai pu remplir les objectifs fixés, grâce aux directives de mon maitre de stage et aux semaines de développement intensif que j'ai effectué avec persévérance.

Ce stage a donc été très bénéfique pour moi et m'a permis d'accroitre mes connaissances en termes de programmation, de conception et de gestion de projets.

Au cours de la rédaction de ce rapport, j'ai présenté les différentes étapes de la conception et de la réalisation de mon api rest. J'ai commencé par une étude du projet dans le premier chapitre, ensuite l'analyse des besoins et à la conception dans le second chapitre et pour finir, j'ai présenté la structure, les outils de développement et les services de l'api rest réalisée.

Cette api rest peut être améliorée par la création d'une **application front** qui permettra de consommer les services de l'api rest, l'utilisation de **spring security** pour sécuriser l'api rest et l'utilisation de **Docker** pour la virtualisation de l'api rest.

WEBOGRAPHIE

- https://practicalprogramming.fr/api-rest
  Consulter le 25/05/2022
- https://spring.io/guides#tutorials
  Consulter le 02/06/2022
- https://openclassrooms.com
  Consulter le 13/06/2022
- https://www.youtube.com/playlist?list=PLjwdMgw5TTLXuY5i7RW0QqGdW0NZntqiP
  Consulter le 26/05/2022
- https://spring.io/projects/spring-boot
  Consulter le 03/06/2022
- https://stackoverflow.com/
  Consulter le 16/06/2022
- https://www.baeldung.com/
  Consulter le 19/06/2022
- https://howtodoinjava.com/
  Consulter le 23/06/2022
- https://start.spring.io/
  Consulter le 07/06/2022