REPUBLIQUE TOGOLAISE

Travail-Liberté-Patrie

07 BP 12456 Lomé 07, TOGO

Tel: (+228) 22 20 47 00

E-mail: iaitogo@iai-togo.tg

Site web: www.iai-togo.com

Agoè-Nyivé Toganim, Lomé-TOGO

Tel : (+228) 98 91 17 17

E-mail : info@leaderdigitalgroup.com

Site web : www.leaderdigitalgroup.com

RAPPORT DE STAGE PRATIQUE EN ENTREPRISE

Type de stage: Génie Logiciel

Mise en place d'une API de gestion des lignes de transport urbain

Période : 06 Juin 2022 – 05 Août 2022

Rédigé et présenté par:

KODJO Afiwa Aimée

Etudiante en deuxième année Tronc Commun

Superviseur :

M. DAVON Essè

Enseignant à IAI-TOGO

Maître de stage:

M. WILSON Komlan

Directeur de LEADER DIGITAL SARL

REMERCIEMENTS:

Je voudrais profiter de ces quelques lignes et de cette occasion qui m'est donnée pour remercier toutes les personnes qui, de près ou de loin, ont permis la réalisation de ce stage dans de meilleures conditions.

J'adresse ainsi mes plus sincères remerciements à :

Monsieur Komlan WILSON, directeur de l'entreprise LEADER DIGITAL SARL et mon maître de stage, pour sa confiance et son accompagnement dont j'ai pu bénéficier tout au long de ce stage. Je le remercie aussi pour sa disponibilité et la qualité de son encadrement en entreprise.

L'Institut Africain d'Informatique (IAI-TOGO), plus précisément son corps professoral et administratif pour leur soutien et la qualité de l'enseignement offert.

Monsieur Essè DAVON, enseignant à IAI-TOGO et mon superviseur, pour m'avoir accordé de son temps, conseillé et orienté tout au long de mon stage.

Enfin, j'ai également une pensée chaleureuse pour mes amis, les membres de ma famille qui n'ont cessé de me soutenir et de m'encourager au quotidien dans toutes mes démarches, à la fois sur le plan professionnel et sur le plan personnel.

API de gestion des lignes de transport urbain

Sommaire

Liste des figures

Liste des tableaux

Introduction :

Aujourd'hui, l'informatique a pris une place prédominante dans le monde. Aucune entreprise, petite, moyenne ou grande ne peut se permettre de poursuivre ses activités sans ce dernier. Cet outil a rendu le traitement des tâches plus aisé. C'est ce qui explique l'engouement de nombreux jeunes à se tourner vers les filières technologiques.

C'est ce qui explique mon choix d'être à l'Institut Africain d'Informatique (IAI-TOGO). Etant actuellement en deuxième année de licence, il est indispensable pour les nécessités de ma formation d'effectuer un stage au sein d'une entreprise dans le but d'allier la pratique à la théorie. D'où, ma présence au sein de la structure LEADER DIGITAL SARL. J'ai donc voulu effectuer mon stage au sein de cette entreprise qui répondait aux besoins de ma formation.

L'entreprise LEADER DIGITAL SARL se situe dans le quartier d'Agoè-Nyivé, Lomé - TOGO et est spécialisée dans le cloud computing, le génie logiciel.

La mission qui m'a été confiée dans le cadre de ce stage est de mettre en place une API afin de gérer les lignes de transport, d'où le thème : « **Mise en place d'une API de gestion des lignes de transport urbain** ».

Je vais donc dans ce rapport vous présenter le travail que j'ai pu réaliser. Dans un premier temps, j'aborderai la partie du cahier des charges, ensuite celle de la pré-programmation et enfin le rapport de réalisation et de mise en œuvre.

PARTIE 1:
CAHIER DES CHARGES

Chapitre 1

Partie 1 : Cahier des charges

1.1. Présentation du sujet

Une ligne de transport urbain est un itinéraire prédéfini de transport en commun. Elle comprend plusieurs arrêts desservant autant de quartiers, de ville ou lieux-dits différents qu'elle traverse. En ce sens, la gestion des lignes de transport urbain consiste à définir les lignes de transport, les quartiers et les points d'arrêt. Cette tâche revêt une importance capitale pour toutes les entreprises de transport voulant se faire une place sur le marché togolais. Ainsi, au travers de du thème « **Mise en place d'une API de gestion des lignes de transport urbain** », nous nous intéresserons aux sociétés de transport du Togo, dont nous développerons les problèmes rencontrés et les solutions que nous aurons à leur proposer.

1.2. Problématique du sujet

Ayant à cœur les difficultés de la population dans le domaine des transports, les sociétés de transport du Togo cherchent à gérer au mieux leurs lignes de transport. En effet, ces dernières sont confrontées à de nombreux problèmes et s'articulent au prisme de la problématique de **gestion efficace des lignes de transport**. Nous sommes donc amenés à nous poser les questions suivantes :

- Comment gérer efficacement les lignes de transport ?
- Ne peut-on pas définir des points d'arrêt pour chaque ligne de transport ?
- Comment gérer de façon plus précise les quartiers afin de rendre plus explicit les renseignements sur chaque ligne de transport ?

1.3. Intérêt du sujet

1.3.1. Objectifs

Ce projet a pour objectif de gérer au mieux les lignes de transport urbain des sociétés de transport du Togo en mettant en place une Interface de Programmation d'Application (API). Il sera donc de façon plus spécifique que ce projet permette de:

- Gérer efficacement les lignes de transport ;
- Définir les points d'arrêt de chaque ligne de transport ;

- Connaître les quartiers de chaque point d'arrêt ;
- Documenter l'Interface de Programmation d'Application(API).

1.3.2. Résultats

Au terme de ce projet, les résultats suivants pourront être observés :

- Les lignes de transport sont gérées efficacement ;
- Les points d'arrêt de chaque ligne de transport sont définis ;
- Les quartiers de chaque point d'arrêt sont connus ;
- L'Interface de Programmation d'Application(API) est documentée.

PARTIE 2:
PRE-PROGRAMMATION

Chapitre 2

Partie 2 : Pré-programmation

2.1. Etude de l'existant

Comme toutes les sociétés de transport du monde, certaines sociétés de transport du Togo disposent également d'un site web. Leur site web présente la société en général et leurs offres de service. Les utilisateurs s'en servent souvent pour connaître le moyen de transport à prendre afin d'aller à destination. Par exemple, pour la société de transport SOTRAL, les passagers se rendent sur le site web et consultent les bus à prendre afin de se rendre à leur destination (à un point d'arrêt).

2.2. Critique de l'existant

Depuis leur mise en place, la procédure existante a atteint leurs objectifs puisqu'elle a répondu un temps soit peu aux attentes de la population. L'étude de l'existant nous a permis de relever les insuffisances sur les aspects suivants :

- **Précision :**

Les passagers des moyens de transport connaissent leur point d'arrêt mais pas le quartier dans lequel se trouve ce dernier. Il serait mieux que chaque passager sache dans quel quartier se trouve leur destination ;

- **Multiplicité d'information :**

Pour chaque ligne choisie par un passager, il doit pouvoir avoir accès à tous les points d'arrêts et aussi le quartier de chaque point d'arrêt afin de mieux se situer.

2.3. Planning prévisionnel de la réalisation

1: Planning prévisionnel

Date : Du au	Nombre de jours	Tâche	Observation
09/06/2022 au 10/06/2022	1 jour	Intégration : Connaissance du centre d'accueil et sur ses activités	Bien compris
13/06/2022 au 20/06/2022	7 jours	Apprendre des notions du framework	Effectué
20/06/2022 au 23/06/2022	3 jours	Création de chaque classe avec leurs attributs	Effectué
23/06/2022 au 27/06/2022	4 jours	Relation entre chaque classe	Effectué
27/06/2022 au 30/06/2022	3 jours	Création des services et Contrôleurs pour la classe Quartier	Effectué
30/06/2022 au 04/07/2022	4 jours	Création des services et Contrôleur pour la classe PtArret	Effectué
04/07/2022 au 06/07/2022	2 jours	Création des services et contrôleur pour la classe Ligne	Effectué

Date : Du au	**Nombre de jours**	**Tâche**	**Observation**
06/07/2022 au 11/07/2022	5 jours	Gestion des exceptions de la classe Quartier	Effectué
11/07/2022 au 13/07/2022	2 jours	Gestion des exceptions de la classe PtArret	Effectué
13/07/2022 au 15/07/2022	2 jours	Gestion des exceptions de la classe Ligne	Effectué
15/07/2022 au 20/07/2022	5 jours	Documentation de l'API	Effectué
20/07/2022 au 05/08/2022	16 jours	Front-end de l'application	Effectué

2.4. Etude détaillée de la solution

Dans cette partie, nous aurons à utiliser UML (Unified Modeling Language) qui est un langage de modélisation et expliquer le pourquoi nous avions fait ce choix.

2.4.1. Présentation de la méthode d'analyse

A. Le langage UML

Le langage de modélisation unifié, en anglais Unified Modeling Language est un langage de modélisation graphique à base de pictogrammes (dessins figuratifs stylisés ayant fonction de signe) conçu pour fournir une méthode normalisée pour visualiser la conception d'un système. Il est couramment utilisé en développement logiciel et en conception orientée objet. UML est le résultat de la fusion de précédents langages de modélisation objet : Booch, OMT, OOSE. Principalement issu des travaux de Grady Booch, James Rumbaugh et Ivar Jacobson, UML est à présent un standard adopté par l'Object Management Group.

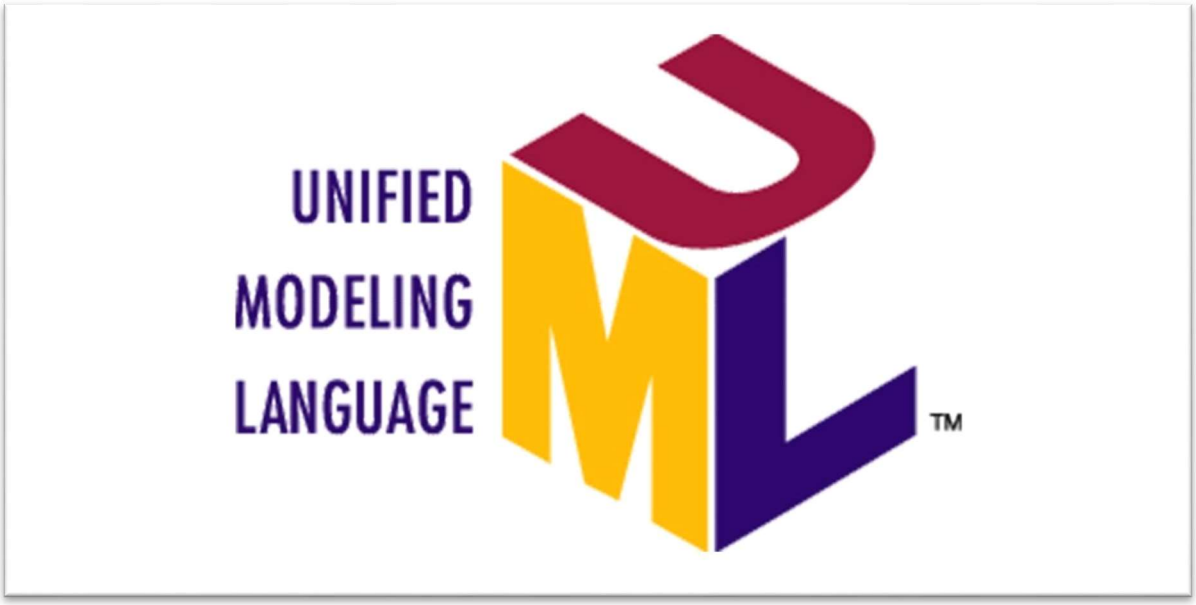

Figure 1: Logo de UML

B. Les diagrammes UML

UML propose 14 types de diagramme. Ces diagrammes sont dépendants hiérarchiquement et se complètent, de façon à permettre la modélisation d'un projet tout au long de son cycle de vie. Ces diagrammes se présentent comme suit:

a. Diagrammes structurels

- **Diagramme de classe :** représentation des classes intervenant dans le système ;
- **Diagramme d'objets :** représentation des instances de classes (objets) utilisées dans le système ;
- **Diagramme de composants :** représentation des composants du système d'un point de vue physique, tels qu'ils sont mis en œuvre (fichiers, bases de données, bibliothèques…) ;
- **Diagramme de déploiement :** représentation des éléments matériels (ordinateurs, périphériques, réseaux, système de stockage…) et la manière dont les composants du système sont répartis sur ces éléments matériels et interagissent entre eux ;
- **Diagramme des paquets :** représentation des dépendances entre paquets c'est-à-dire entre les ensembles de définitions ;
- **Diagramme de structure composite :** représentation sous forme de boîte blanche des relations entre les composants d'une classe ;
- **Diagramme de profils :** spécialisation et personnalisation pour un domaine particulier d'un Méta modèle de référence UML.

b. Diagrammes de comportement :

- **Diagramme états-transitions :** représentation sous forme de machine à états finis du comportement du système ou de ses composants ;
- **Diagramme d'activité :** représentation sous forme de flux ou d'enchaînement d'activités du comportement du système ou de ses composants ;
- **Diagramme de cas d'utilisation :** représentation des possibilités d'interaction entre le système et les acteurs.

c. Diagramme d'interaction

- **Diagramme de séquence :** représentation de façon séquentielle du déroulement des traitements et des interactions entre les éléments du système et/ou de ses acteurs ;
- **Diagramme de communication :** représentation de façon simplifiée d'un diagramme de séquence se concentrant sur les échanges de messages entre les objets ;
- **Diagramme global d'interaction :** représentation des enchaînements possibles entre les scénarios préalablement identifiés sous forme de diagrammes de séquence ;
- **Diagramme de temps :** représentation des variations d'une donnée au cours du temps.

2.4.2. Présentation de l'outil de modélisation

Dans la modélisation, on pouvait se contenter d'utiliser uniquement UML, mais sur quel support ?. De nos jours, on ne peut pas se contenter des dessins faits sur papier ; c'est là qu'intervient les logiciels de modélisation qui permettent la réutilisation des modèles, leur portabilité, etc. L'outil de modélisation retenu est : Sybase PowerAMC version 15.1.

A. Qu'est-ce que PowerAMC ?

PowerAMC est un environnement graphique de modélisation d'entreprise très simple d'emploi. Il a été créé par la société SDP sous le nom AMC*Designor, racheté par Sybase en 1995. PowerAMC prend en compte plusieurs outils de modélisation tels que UML et MERISE. PowerAMC permet d'effectuer les tâches suivantes :

- Modélisation intégrée via l'utilisation de méthodologie et de notation standard :
  - Données (E/R, Merise) ;
  - Métiers (BPMN, BPEL, ebXML) ;
  - Application (UML).
- Génération automatique de code via des templates personnalisables :
  - SQL (avec plus de 50 SGBD) ;
  - Java ;
  - .NET.

- Fonctionnalités de reverse engineering pour documenter et mettre à jour des systèmes existants :
  - Une solution de référentiel d'entreprise avec des fonctionnalités de sécurité et de gestion des versions très complètes ;
  - Fonctionnalités de génération et de gestion de rapports automatisés et personnalisables ;
  - Un environnement extensible qui vous permet d'ajouter des règles, des commandes, des concepts et des attributs à vos méthodologies de modélisation et de codage.

B. Modélisation avec PowerAMC

PowerAMC fournit un jeu unique d'outils de modélisation professionnels qui associent les techniques et notations standards de la modélisation de processus métier, de la modélisation des données et la modélisation des diagrammes UML et d'autres fonctionnalités puissantes afin d'aider à analyser, concevoir, construire et maintenir des applications en utilisant les techniques les plus élaborées d'ingénierie logicielle. La solution de modélisation PowerAMC permet d'intégrer étroitement la conception et la maintenance des couches de données centrales de l'application et exigences de projet, processus métiers, code orienté objet, vocabulaires XML et informations de réplication de bases de données.

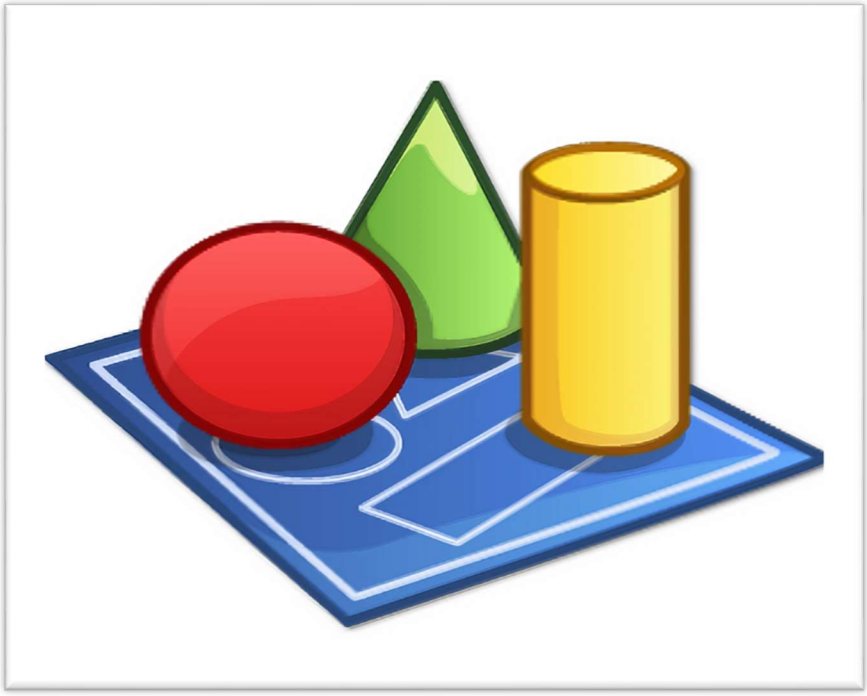

Figure 2: Logo de powerAMC

2.4.3. Diagramme de cas d'utilisation

Un cas d'utilisation traduit tout ce que l'utilisateur exprime comme action sur le logiciel ou le système à modéliser. C'est une représentation faisant intervenir les acteurs et les cas d'utilisation. Il traduit les besoins des utilisateurs vis-à-vis du système développé.

A. Les acteurs

Un acteur est une personne ou un système qui interagit avec le système en échangeant des informations en entrée comme en sortie. Le diagramme des cas d'utilisation d'UML distingue deux types acteurs à savoir :

- Les acteurs principaux (qui modifient l'état du système ou qui consultent cet état) ;
- Les acteurs secondaires (acteurs auxquels le système fait appel pour répondre aux sollicitations d'un acteur principal).

Dans notre projet, nous avons identifié les acteurs suivants:

- Utilisateurs
- Administrateur

B. Les cas d'utilisation

Un cas d'utilisation exprime le comportement du système en termes d'actions et réactions face à un besoin d'un utilisateur. Dans notre projet, nous avons identifié les cas d'utilisation suivants :

- **Consulter les quartiers**. Ce cas d'utilisation est constitué des opérations suivantes :
    1. Afficher la liste des quartiers
    2. Afficher un quartier
- **Consulter les lignes**. Ce dernier consiste entre autre à :
    1. Afficher la liste des lignes
    2. Afficher une ligne
- **Consulter les points d'arrêt**. Ce cas d'utilisation comprend les opérations suivantes :
    1. Afficher la liste des points d'arrêt
    2. Afficher un point d'arrêt

- **Gérer un quartier.** Il comprend les opérations suivantes :
    1. Ajouter un quartier
    2. Supprimer un quartier
    3. Afficher la liste des quartiers
    4. Afficher un quartier
- **Gérer une ligne.** Ce cas d'utilisation comprend les opérations suivantes :
    1. Ajouter une ligne
    2. Supprimer une ligne
    3. Afficher la liste des lignes
    4. Afficher une liste
- **Gérer un point d'arrêt.** Ce dernier consiste entre autre à :
    1. Ajouter un point d'arrêt
    2. Supprimer un point d'arrêt
    3. Afficher la liste des points d'arrêt
    4. Afficher un point d'arrêt

C. Diagramme de cas d'utilisation par acteur

❖ Diagramme de cas d'utilisation des utilisateurs

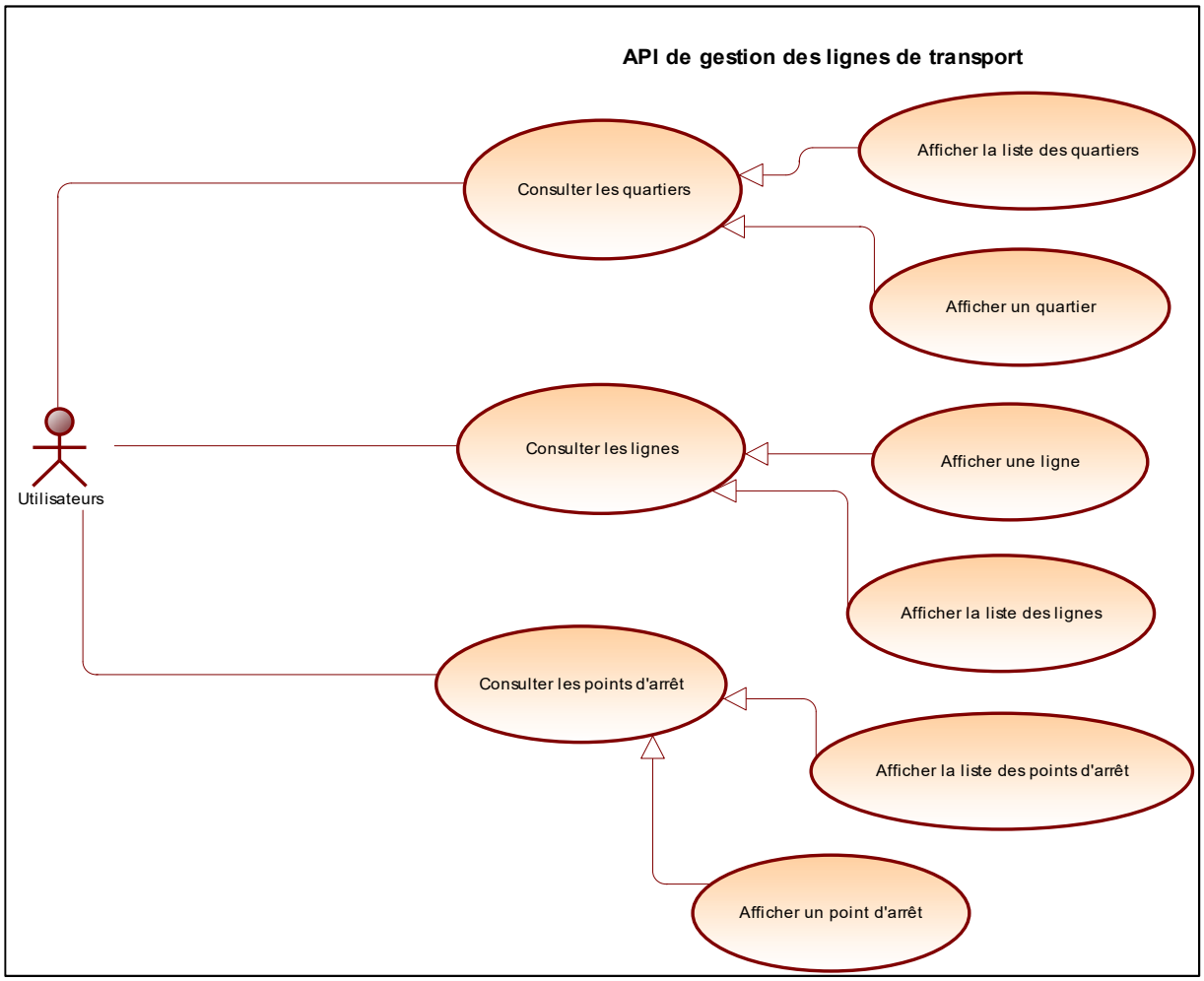

Figure 3: Diagramme de cas d'utilisation des utilisateurs

❖ Diagramme de cas d'utilisation de l'administrateur

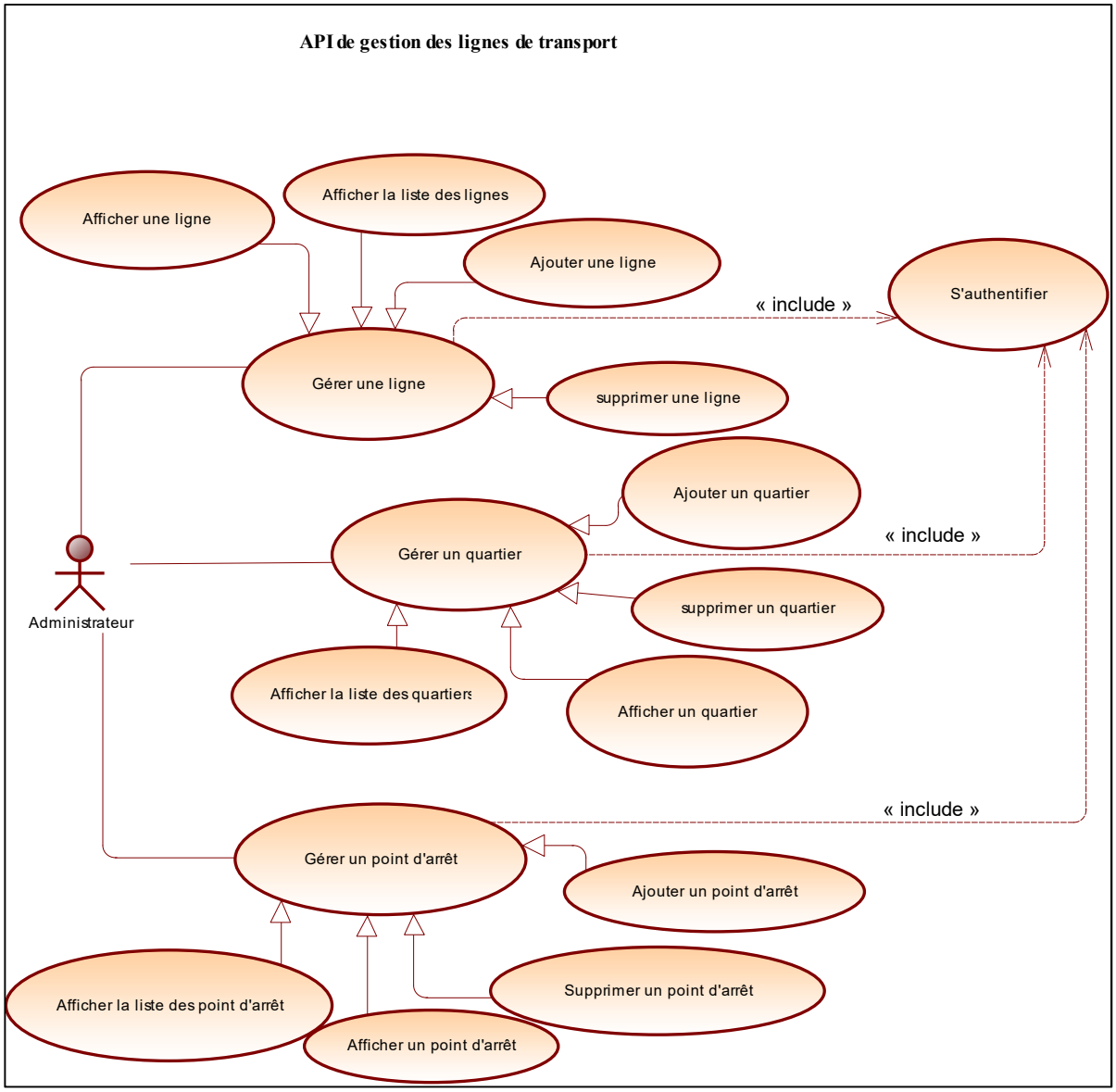

Figure 4: Diagramme de cas d'utilisation de l'administrateur

Description textuelle : Le cas d'utilisation 'Consulter les quartiers' du système.

Partie 1 : Description.

- Titre : Consulter les quartiers.

- Résumé : Ce cas d'utilisation permet d'afficher les quartiers.

- Acteur principal : Les utilisateurs.

- Version : 1.0.

Partie 2 : Description des scénarios

- **Pré-conditions :**
- L'API est opérationnelle.
- L'application est fonctionnelle.
- **<u>Scénario nominal</u> :**
    1) L'utilisateur se connecte et demande l'affichage des quartiers.
    2) L'application fait appel à l'API.
    3) L'API vérifie que la requête d'affichage fournit est valide. **(SE1)**
    4) L'API envoie la liste des quartiers à l'application.
    5) L'application affiche aux utilisateurs la liste des quartiers.
- **<u>Scénarios d'exception</u> :**
    - **<u>Scénario d'exception SE1</u> :** Requête non valide.
      SE commence au point 3 du scénario nominal. L'application envoie un message d'erreur et met fin au cas.
- **<u>Post-conditions</u> :**
  L'API enregistre les détails des requêtes en cas de succès comme en cas d'échec.

Description textuelle : Le cas d'utilisation 'Consulter les lignes' du système.

Partie 1 : Description.

- Titre : Consulter les lignes.

- Résumé : Ce cas d'utilisation permet de consulter les lignes de transport.

- Acteur principal : les utilisateurs.

- Date : 07/08/2022

- Responsable : KODJO Afiwa Aimée

- Version : 1.0

Partie 2 : Description des scénarios.

- **Pré-conditions :**
- L'API est opérationnelle.
- L'application est fonctionnelle.

- **Scénario nominal :**
    1) L'utilisateur se connecte et demande l'affichage des lignes de transport.
    2) L'application fait appel à l'API.
    3) L'API vérifie que la requête d'affichage fournit est valide. **(SE2)**
    4) L'API envoie la liste des lignes de transport à l'application.
    5) L'application affiche aux utilisateurs la liste des lignes de transport.
- **Scénarios d'exception :**
    - **Scénario d'exception SE2 :** Requête non valide.
      SE commence au point 3 du scénario nominal. L'application envoie un message d'erreur et met fin au cas.
- **Post-conditions :**
  L'API enregistre les détails des requêtes en cas de succès comme en cas d'échec.

Description textuelle : Le cas d'utilisation 'Gérer une ligne' du système.

Partie 1 : Description.

- Titre : Gérer une ligne.

- Résumé : Ce cas d'utilisation permet de gérer les lignes de transport.

- Acteur principal : l'administrateur.

- Date : 07/08/2022

- Responsable : KODJO Afiwa Aimée

- Version : 1.0

Partie 2 : Description des scénarios.

- **Pré-conditions :**

- L'API est opérationnelle.

- L'application est opérationnelle.

- **Scénario nominal :**

    1) L'administrateur s'authentifie.
    2) L'API vérifie que l'authentification de l'administrateur est correcte. **(SE3)**

3) L'administrateur effectue des opérations sur les lignes.
4) Les opérations sont effectuées sur les lignes.

- **Scénario d'exception :**
  - **Scénario d'exception SE3 :** Erreur lors de l'authentification.
    SE commence au point 2 du scénario nominal. L'application envoie un message d'erreur et met fin au cas.
- **Post-conditions :**
  L'API enregistre les détails des requêtes en cas de succès comme en cas d'échec.

2.4.4. Diagramme de classes

Le diagramme de classe permet de spécifier la structure et les liens entre les objets dont le système est composé : il spécifie « qui » sera à l'œuvre dans le système pour réaliser les fonctionnalités décrites par les diagrammes de cas d'utilisation.

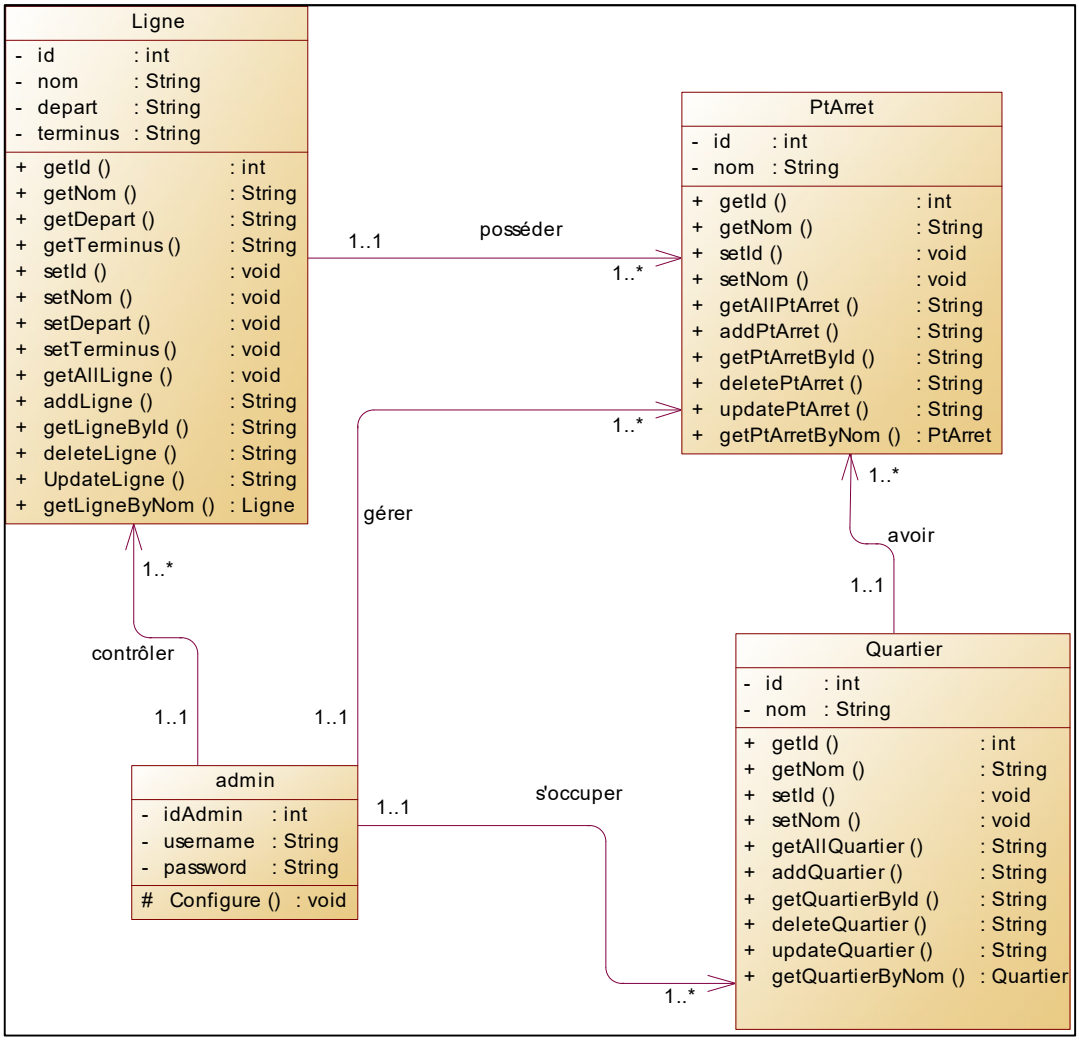

Figure 5: Diagramme de classes

2.4.5. Diagramme de séquence

Un diagramme de séquence permet de compléter et de visualiser simplement et intuitivement la description textuelle.

❖ **Consulter les quartiers**

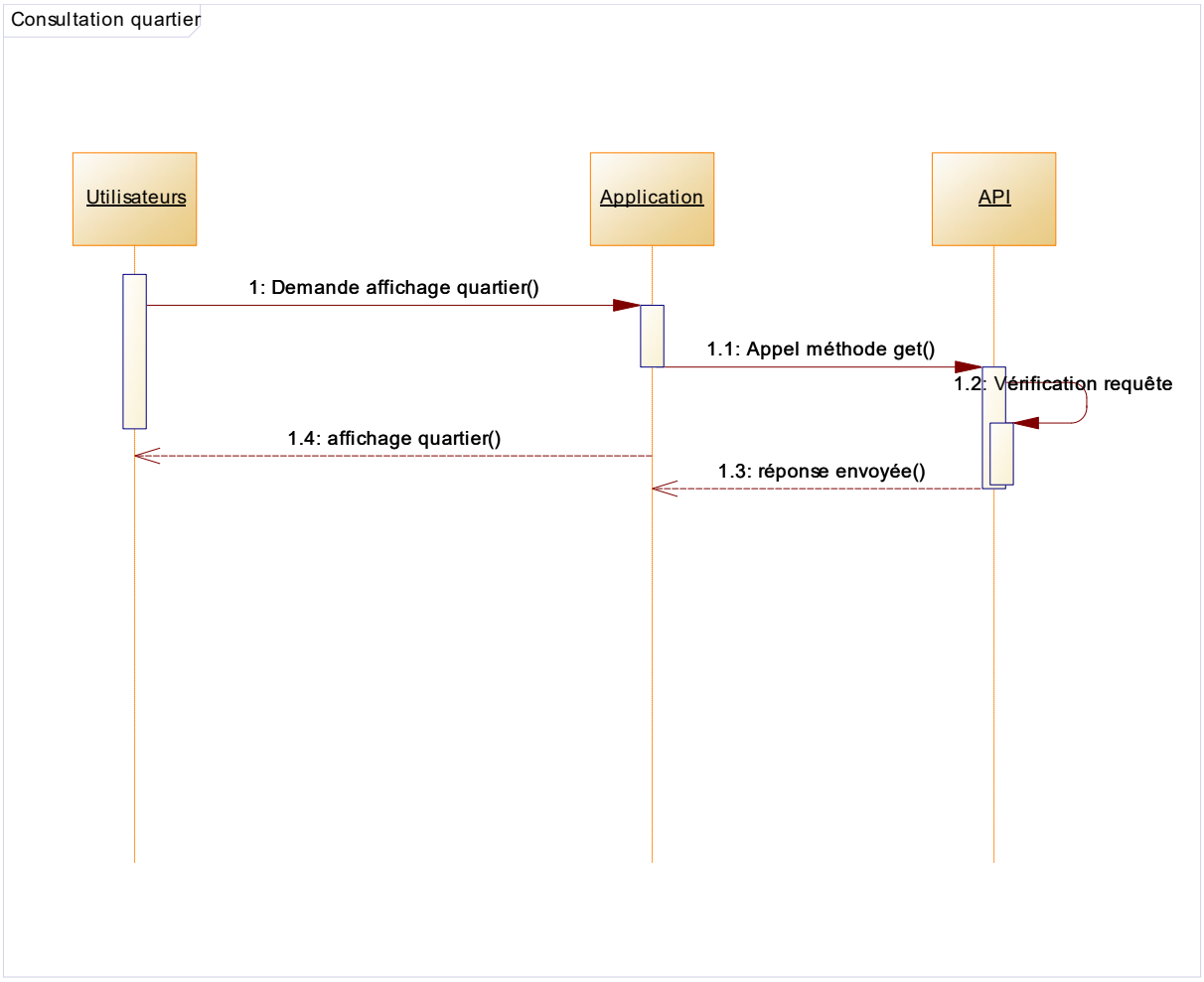

Figure 6: Diagramme de séquence de consulter les quartiers

❖ Consulter les lignes

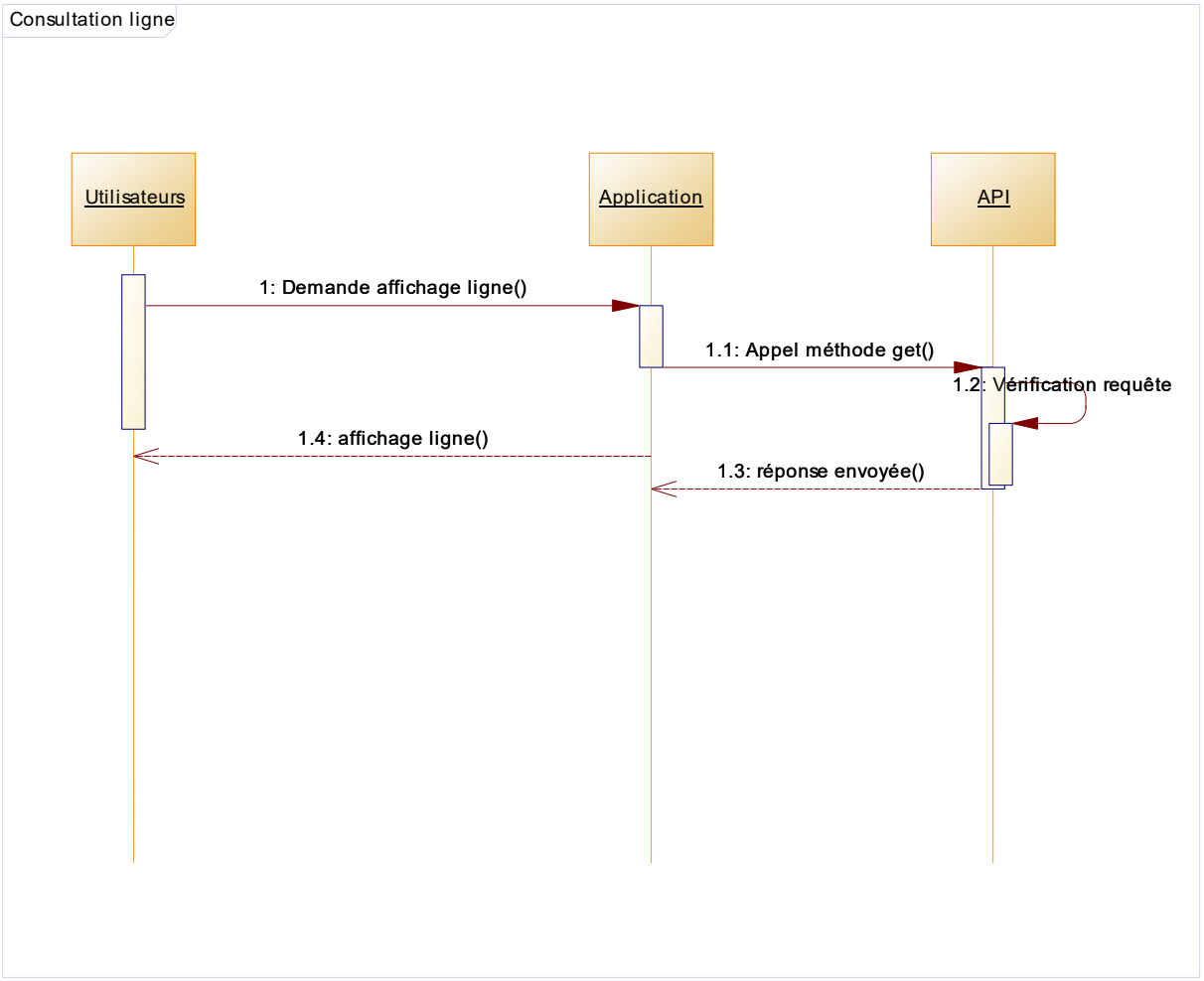

Figure 7:Diagramme de séquence de consulter les lignes

❖ Gérer une ligne

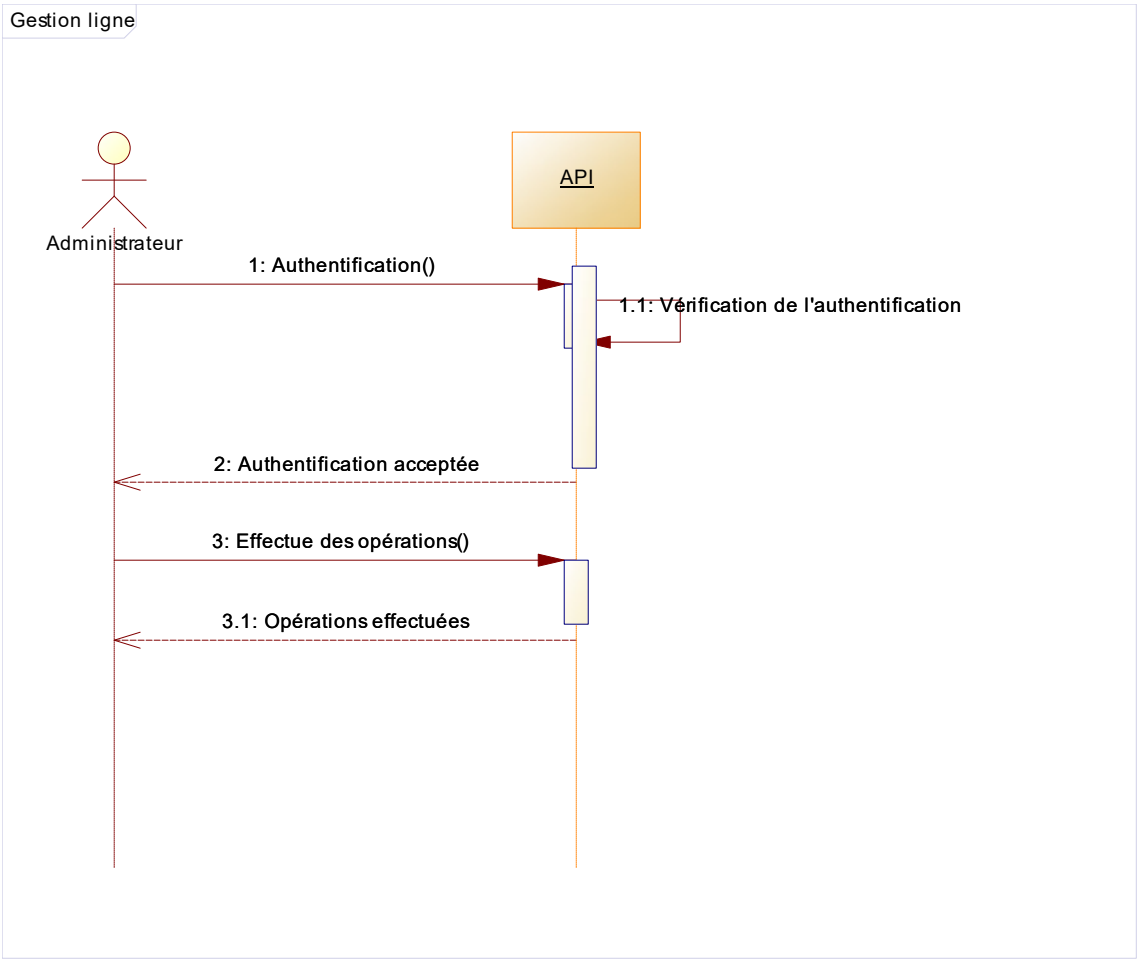

Figure 8: Diagramme de séquence de gérer une ligne

2.4.6. Diagramme d’activité

❖ Consulter les quartiers

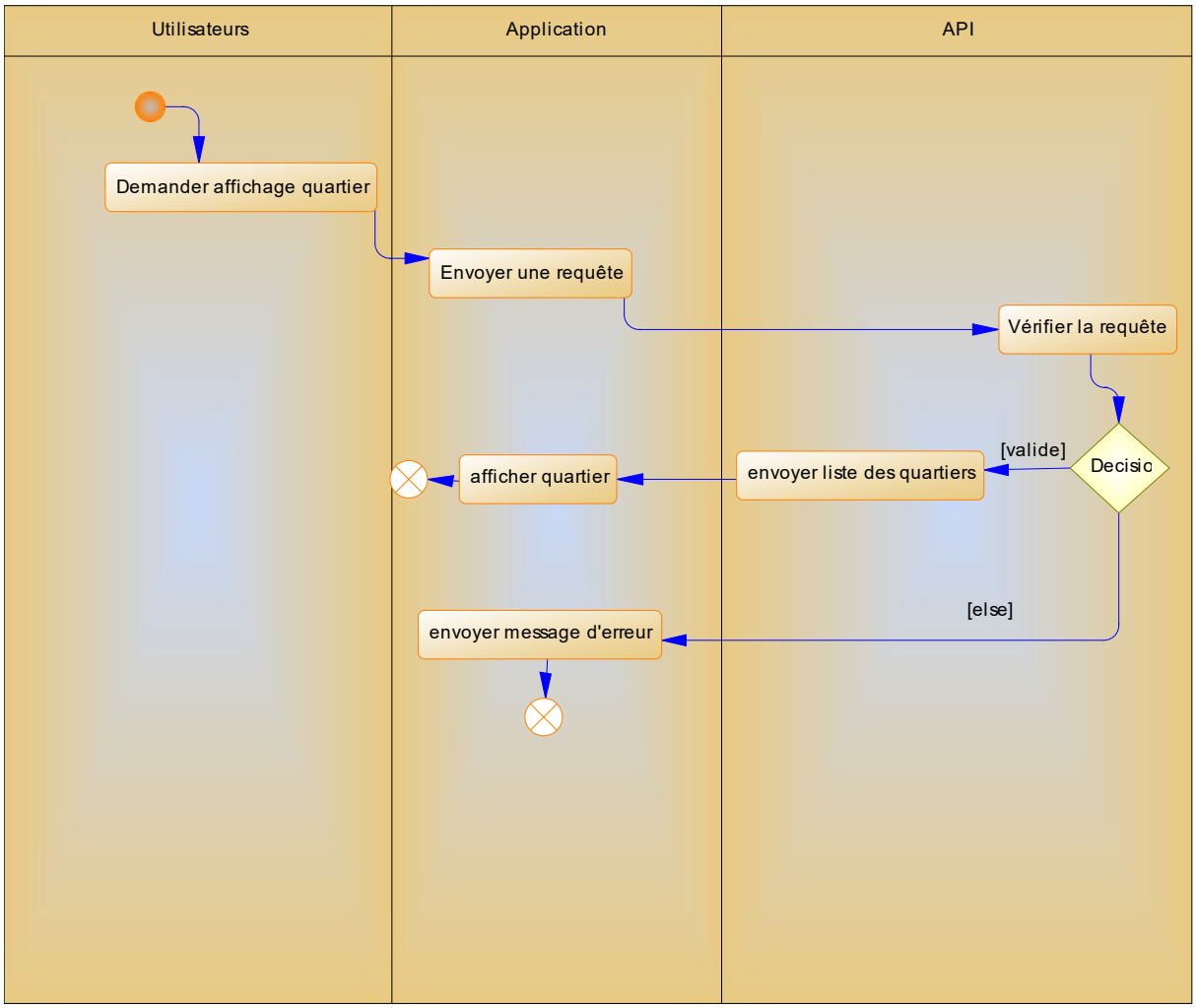

Figure 9: Diagramme d'activité consulter les quartiers

❖ Consulter les lignes

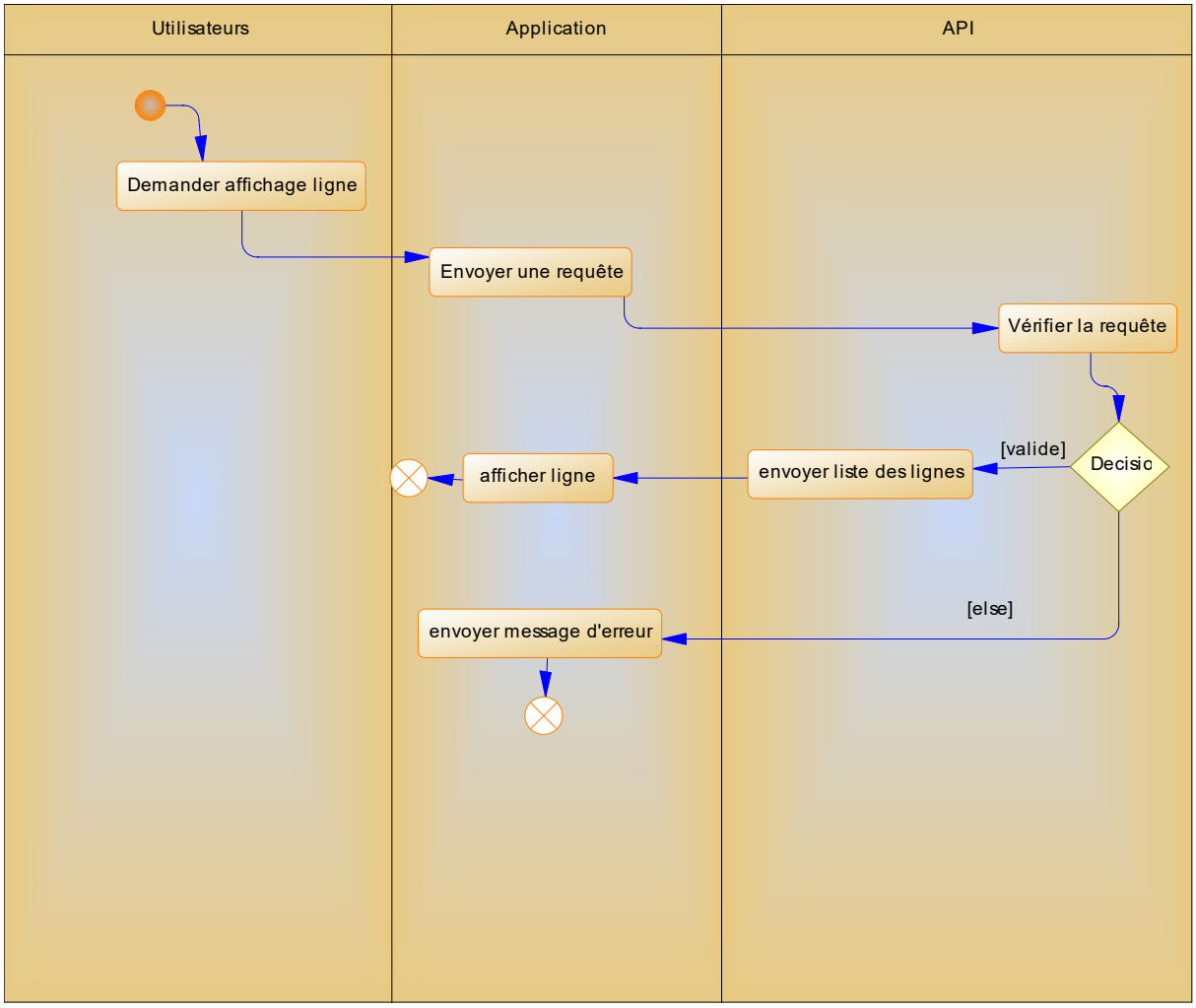

Figure 10: Diagramme d'activité consulter les lignes

❖ Gérer une ligne

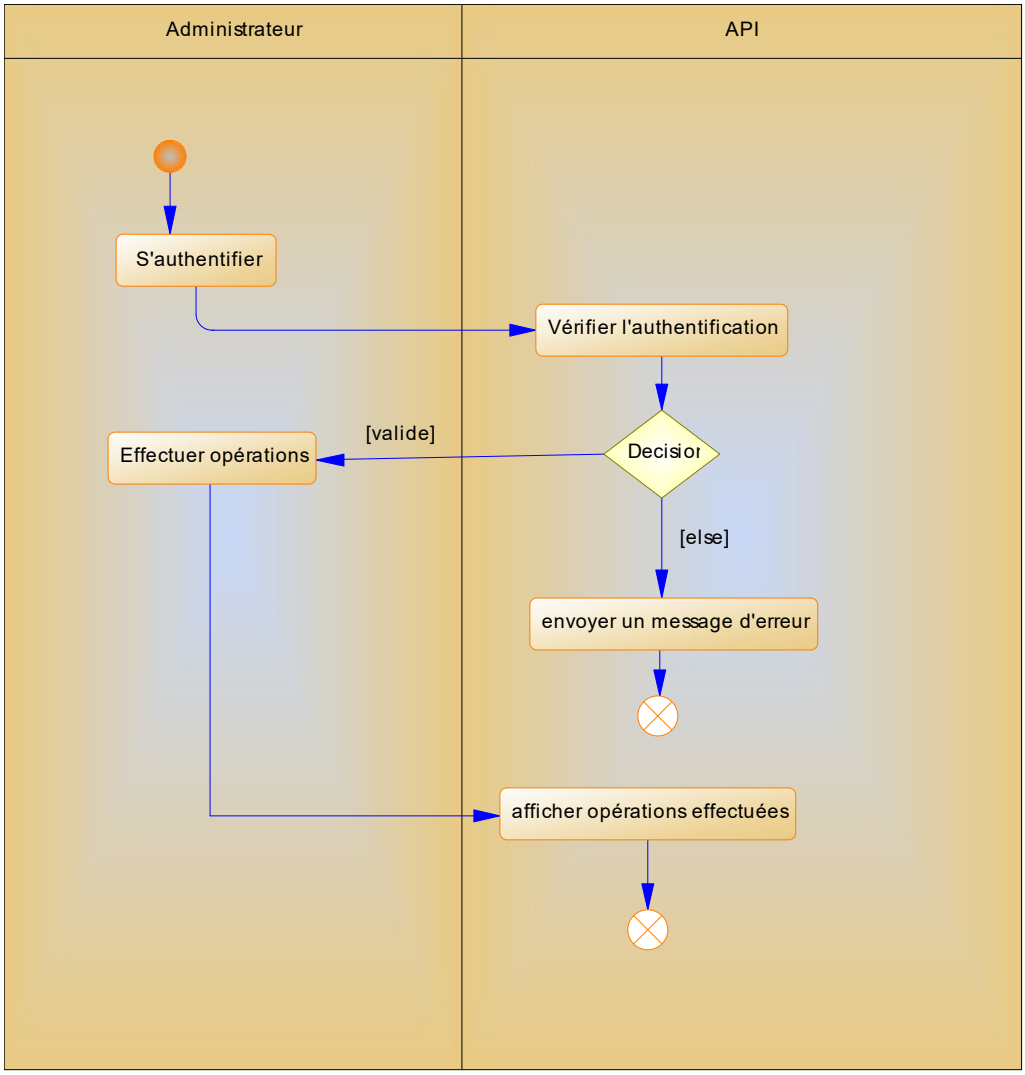

Figure 11:Diagramme d'activité gérer une ligne

PARTIE 3: REALISATION ET MISE EN OEUVRE

Chapitre 3

Partie 3 : Réalisation et mise en œuvre

3.1. Matériels et logiciels utilisés

3.1.1. Matériels utilisés

Notre matériel de développement durant la période de stage est un laptop dont les caractéristiques sont les suivantes :

- Fabricant : HP
- Processeur : Intel(R) Core(TM) i5-3230M CPU @ 2.60GHz 2.60 GHz
- Mémoire RAM installée : 4,00 Go (3,89 Go utilisable)
- Disque dur : 297,6 Go
- Type du système : Système d'exploitation 64 bits, processeur x64

3.1.2. Logiciels utilisés

Plusieurs outils ont fait partie de l'ouvrage de notre projet. Nous présenterons d'une part les outils d'implémentation de la base de données et d'autre part les outils de programmation et de développement.

1. Les outils d'implémentation de la base de données

Une base de données (en anglais database) permet de stocker et de retrouver des données brutes ou de l'information, souvent en rapport avec un thème ou une activité. La base de données est au centre des dispositifs informatiques de collecte, de mise en forme, de stockage et d'utilisation d'informations. Le dispositif comporte un Système de Gestion de Base de Données (SGBD) : un logiciel moteur qui manipule la base de données et dirige l'accès à son contenu. Le SGBD retenu ici est **MySQL.**

a. MySQL

SQL (Structured Query Language) est un langage informatique normalisé servant à exploiter des bases de données relationnelles. La partie langage de manipulation des données de SQL permet de rechercher, d'ajouter, de modifier ou de supprimer des données dans la base de données relationnelles. Outre le langage de manipulation des données, la partie de définition des données permet de créer et de modifier l'organisation des données dans la base de données, la partie langage de contrôle de transaction permet de commencer et de terminer des transactions et la partie langage de contrôle de données permet d'autoriser ou d'interdire l'accès de certaines données

à certaines personnes. Créé en 1974, normalisé depuis 1986, le langage est reconnu par la grande majorité des Systèmes de Gestion de base de Données Relationnelles (SGBDR) du marché.

Figure 12: Logo de MySQL

2. Les outils de programmation et de développement

a. Visual Studio Code

Visual Studio Code est un éditeur de code extensible développé par Microsoft pour Windows, Linux et MacOs. Les fonctionnalités impliquent la prise en charge du débogage, la mise en évidence de la syntaxe, la complétion intelligente du code, les snippets, la refactorisation du code et Git intégré. Les utilisateurs peuvent modifier le thème, les raccourcis clavier, les préférences et installer des extensions qui ajoutent des fonctionnalités supplémentaires. Visual Studio Code est un éditeur de code source qui peut être utilisé avec une variété de langages de programmation, notamment Java, JavaScript, Go, Node.js et C++.

Figure 13: Logo de Visual Studio Code

b. XAMPP

XAMPP est un ensemble de logiciels permettant de mettre en place un serveur Web local, un serveur FTP et un serveur de messagerie électronique. Il s'agit d'une distribution de logiciels libres (X (cross) Apache MariaDB Perl PHP) offrant une bonne souplesse d'utilisation, réputée pour son installation simple et rapide. Ainsi, il est à la portée d'un grand nombre de personnes puisqu'il ne requiert pas de connaissances particulières et fonctionne, de plus, sur les systèmes d'exploitation les plus répandus.

Figure 14: Logo de XAMPP

c. Spring boot framework

En informatique, Spring est un framework open source pour construire et définir l'infrastructure d'une application Java, dont il facilite le développement et les tests. Spring est effectivement un conteneur dit « léger », c'est-à-dire une infrastructure similaire à un serveur d'applications J2EE. Il prend donc en charge la création d'objets et la mise en relation d'objets par l'intermédiaire d'un fichier de configuration qui décrit les objets à fabriquer et les relations de dépendances entre ces objets. Le gros avantage par rapport aux serveurs d'application est qu'avec Spring, les classes n'ont pas besoin d'implémenter une quelconque interface pour être prises en charge par le

framework (au contraire des serveurs d'applications J2EE et des EJBs). C'est en ce sens que Spring est qualifié de conteneur « léger ».

Figure 15: Logo de Spring Boot

d. Swagger

Swagger est un langage de description d'interface permettant de décrire des API RESTful exprimées à l'aide de JSON. Swagger est utilisé avec toute une série d'outils logiciels open source pour concevoir, créer, documenter et utiliser des services Web RESTful. Swagger inclut des outils de documentation automatisée, de génération de code (dans de nombreux langages de programmation) et de génération de cas de test.

Figure 16: Logo de Swagger

e. Angular

Angular (communément appelé "**Angular 2+**" ou "**Angular v2 et plus**") est un framework côté client, open source, basé sur TypeScript, et codirigé par l'équipe du projet « Angular » à Google et par une communauté de particuliers et de sociétés. Angular est une réécriture complète d'AngularJS, cadriciel construit par la même équipe. Il permet la création d'applications Web et plus particulièrement d'applications web monopage : des applications web accessibles via une page web unique qui permet de fluidifier l'expérience utilisateur et d'éviter les chargements de pages à

chaque nouvelle action. Le Framework est basé sur une architecture du type MVC et permet donc de séparer les données, le visuel et les actions pour une meilleure gestion des responsabilités. Un type d'architecture qui a largement fait ses preuves et qui permet une forte maintenabilité et une amélioration du travail collaboratif.

Figure 17: Logo d'Angular

f. Node.js

Node.js est une plateforme logicielle libre en JavaScript, orientée vers les applications réseau évènementielles hautement concurrentes qui doivent pouvoir monter en charge. Parmi les modules natifs de Node.js, on retrouve http qui permet le développement de serveur HTTP. Ce qui autorise, lors du déploiement de sites internet et d'applications web développés avec Node.js, de ne pas installer et utiliser des serveurs web tels que Nginx ou Apache.

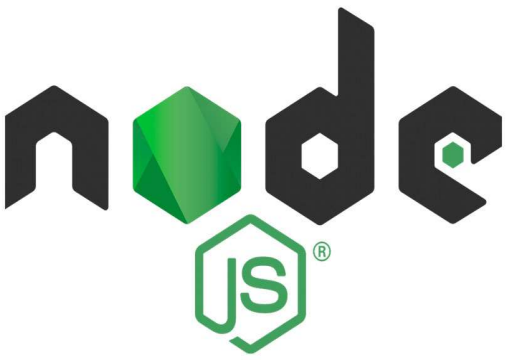

Figure 18: Logo de Node.js

g. Postman

Postman est une application permettant de tester des API. Il sert à exécuter des appels HTTP directement depuis une interface graphique.

Figure 19: Logo de Postman

3.2. Sécurité de l'application

La sécurité constitue une étape indispensable pour la réalisation. Notre API bien qu'elle soit publique présente elle aussi un modèle de sécurité basé sur le code. Sur ce, nous avons entre autres :

- L'application ne permet pas d'agir sur les fonctionnalités; autrement dit que les visiteurs ou les internautes à travers l'application web n'ont pas accès aux modules des administrateurs,
- L'application ne fournit aux internautes que des données strictes nécessaires,
- Lors de l'accès de l'administrateur à sa page, ce dernier doit s'authentifier,
- L'administrateur peut se déconnecter à la fin de chacune de ses sessions,
- L'API demande à ce que l'administrateur puisse confirmer sa demande de déconnexion.

3.3. Evaluation Financière de la solution

a. Coût du materiel

Ici, nous explorons le coût des matériels nécessaires à l'hébergement de l'application.

2: Coût du matériel

N°	Désignation	Temps mis	Source Des coûts	Montant Total (FCFA)
1	onduleur	30 jours	www.infosec-ups.com/fr/cms/conseils-dexperts/quel-onduleur-pour-un-serveur.html	325.000
2	Connexion internet	2 mois	D'après le fournisseur : TOGOCOM	100.000
TOTAL				425.000

b. Coût de conception et de développement

3: Coût de conception

N	Désignation	Temps mis	Source des coûts	Montant Total (FCFA)
1	Conception de l'application	2 mois	www.axiocode.com/estimer-valeur-financiere-app/	500.000
2	Analyse et programmation	150 heures	www.axiocode.com/estimer-valeur-financiere-app/	1.000.000
3	Hébergement et déploiement	37 jours	www.webhosting.net/fr/	245.000
4	Conception + Matériel			2.170.000

3.4. Présentation de l'application

3.4.1. Mise en place de la base de données

-- **Base de données : `transport`**--

-- Structure de la table `**lignes**`--

CREATE TABLE `lignes` (

`id` int (11) NOT NULL,

`depart` varchar (255) DEFAULT NULL,

`nom` varchar (255) DEFAULT NULL,

`terminus` varchar (255) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- Index pour la table `**lignes**`--

ALTER TABLE `lignes`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `UK_nukniexo97y06w36c8sjh7phx` (`nom`);

COMMIT;

-- Structure de la table `**pt_arrets**`--

CREATE TABLE `pt_arrets` (

`id` int(11) NOT NULL,

`nom` varchar(255) DEFAULT NULL,

`quartier_id` int(11) NOT NULL,

`ligne_id` int (11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- Index pour la table `**pt_arrets**`--

ALTER TABLE `pt_arrets`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `UK_5o8k9at80fa52h6x4nvhnhqky` (`nom`),

ADD KEY `FK8d06dm7nmt0bggoltecptns94` (`quartier_id`),

ADD KEY `FKqdku1elx70tnjkx0afsqt1sor` (`ligne_id`);

-- Contraintes pour la table `**pt_arrets**`

ALTER TABLE `pt_arrets`

ADD CONSTRAINT `FK8d06dm7nmt0bggoltecptns94` FOREIGN KEY (`quartier_id`) REFERENCES `quartiers` (`id`),

ADD CONSTRAINT `FKqdku1elx70tnjkx0afsqt1sor` FOREIGN KEY (`ligne_id`) REFERENCES `lignes` (`id`);

COMMIT;

-- Structure de la table `**quartiers**`--

CREATE TABLE `quartiers` (

`id` int(11) NOT NULL,

`nom` varchar(255) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- Index pour la table `**quartiers**`--

ALTER TABLE `quartiers`

ADD PRIMARY KEY (`id`),

ADD UNIQUE KEY `UK_2p1o1s03w0u9p4m714t8sc5yy` (`nom`);

COMMIT;

-- Index pour la table `**admin** `--

CREATE TABLE `admin ` (

`idAdmin` int(11) NOT NULL,

`username` varchar(255) NOT NULL,

`password ` int(11) NOT NULL),

ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

3.4.2. Plan de navigation

Le plan de navigation se présente dans une barre de navigation horizontale se trouvant en haut de la page.

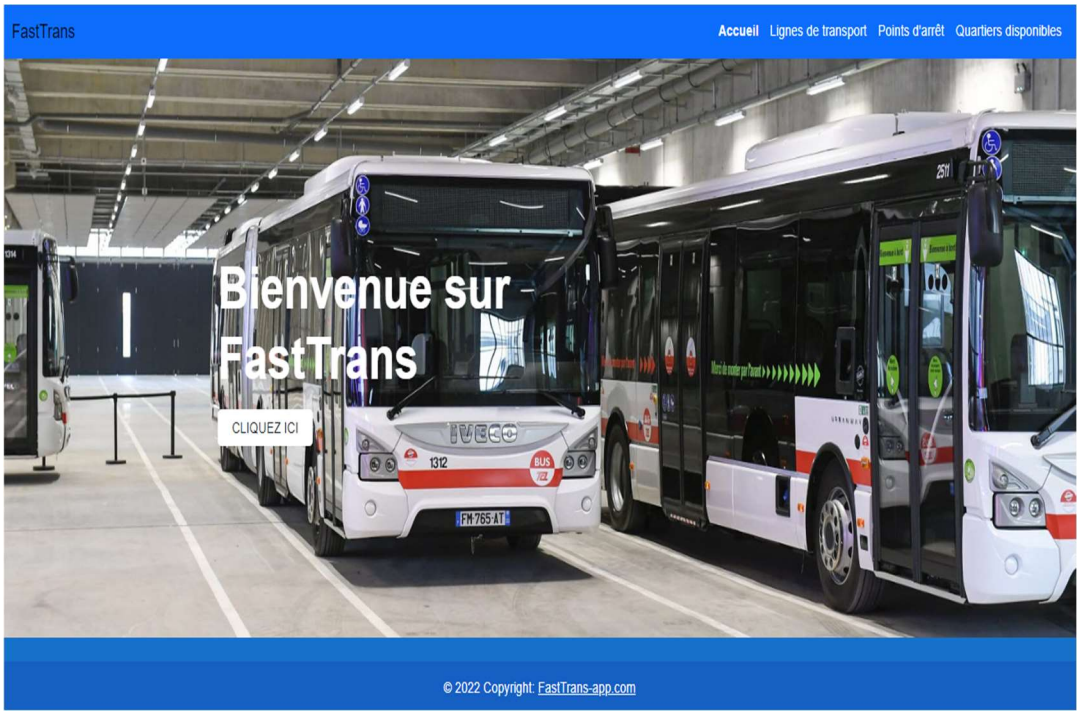

Figure 20: Plan de navigation

3.4.3. Quelques masques de saisie

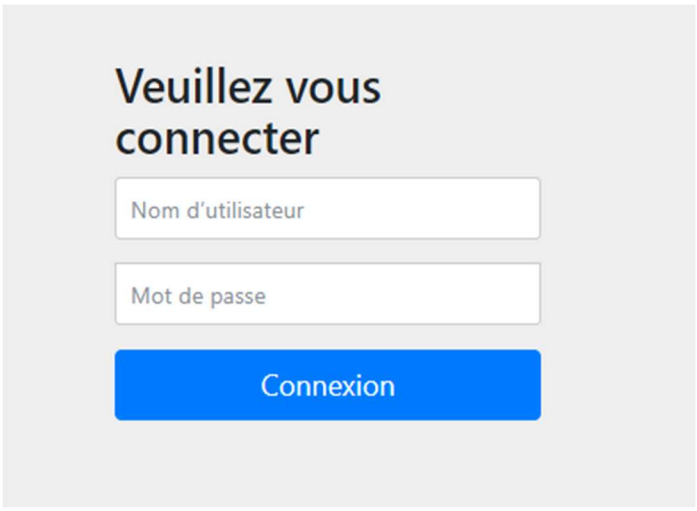

Figure 21: Page d'authentification

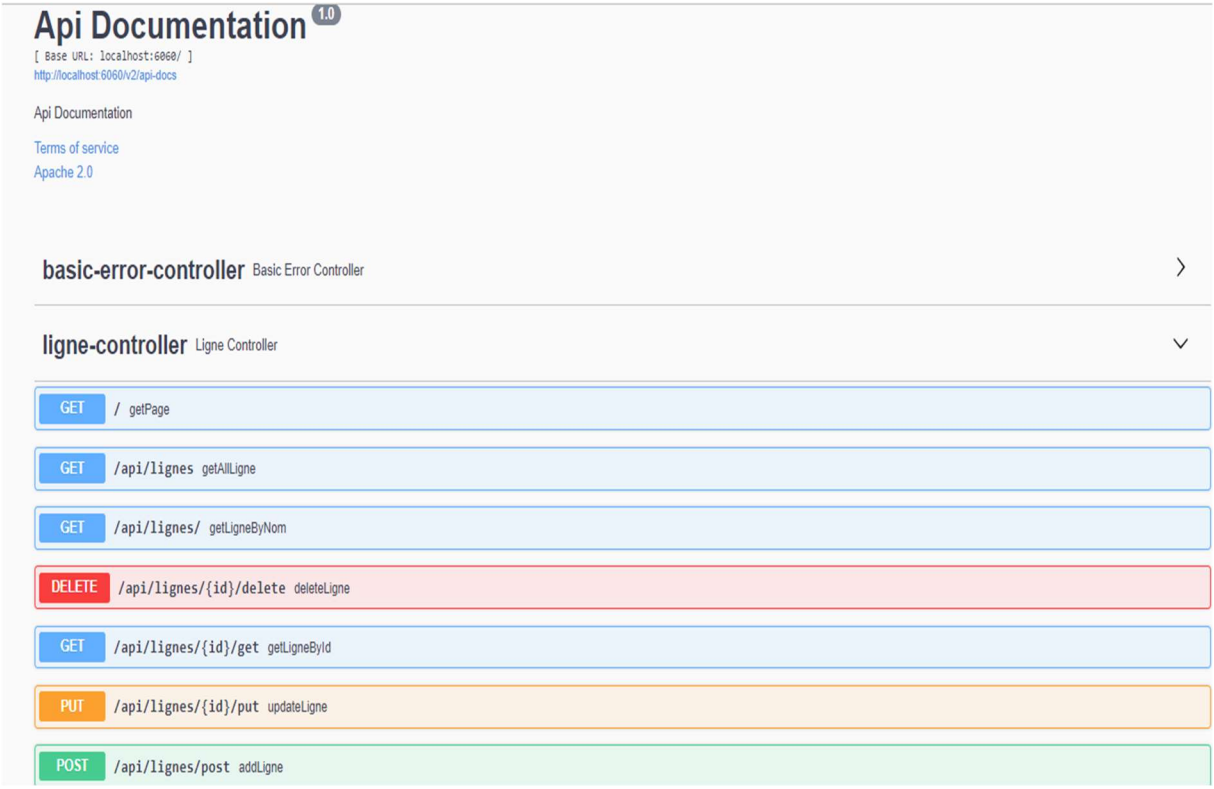

Figure 22: Documentation de l'API(pour les lignes de transport)

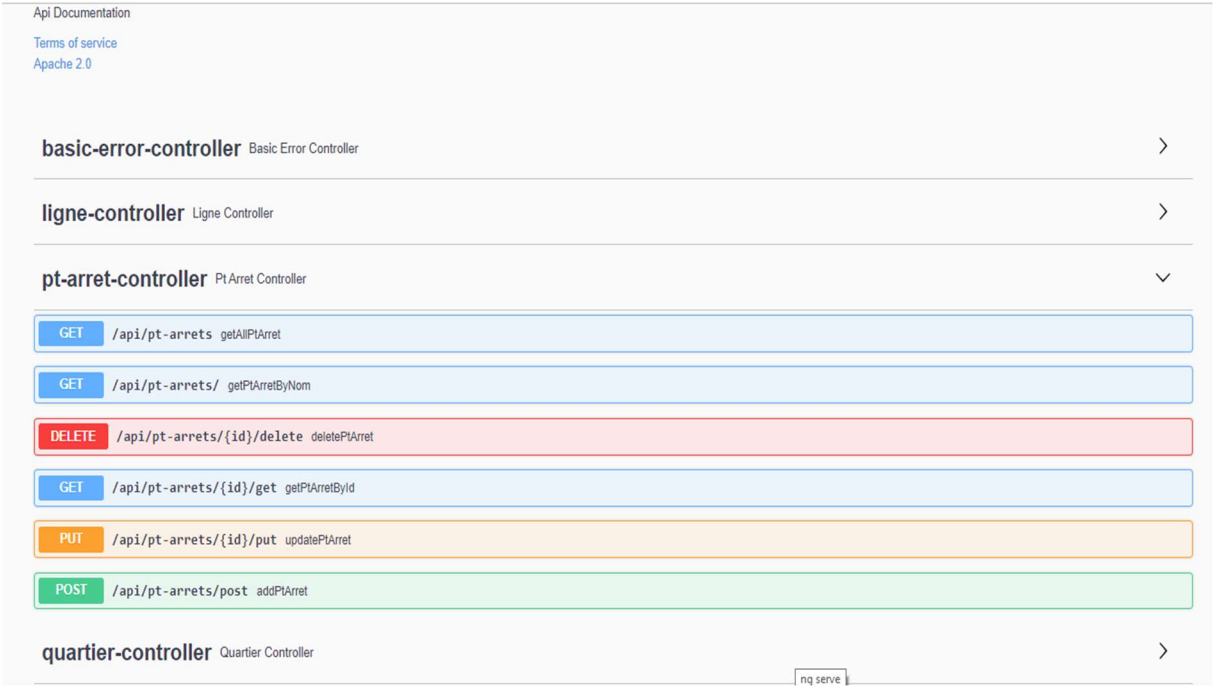

Figure 23: Documentation de l'API(pour les points d'arrêt)

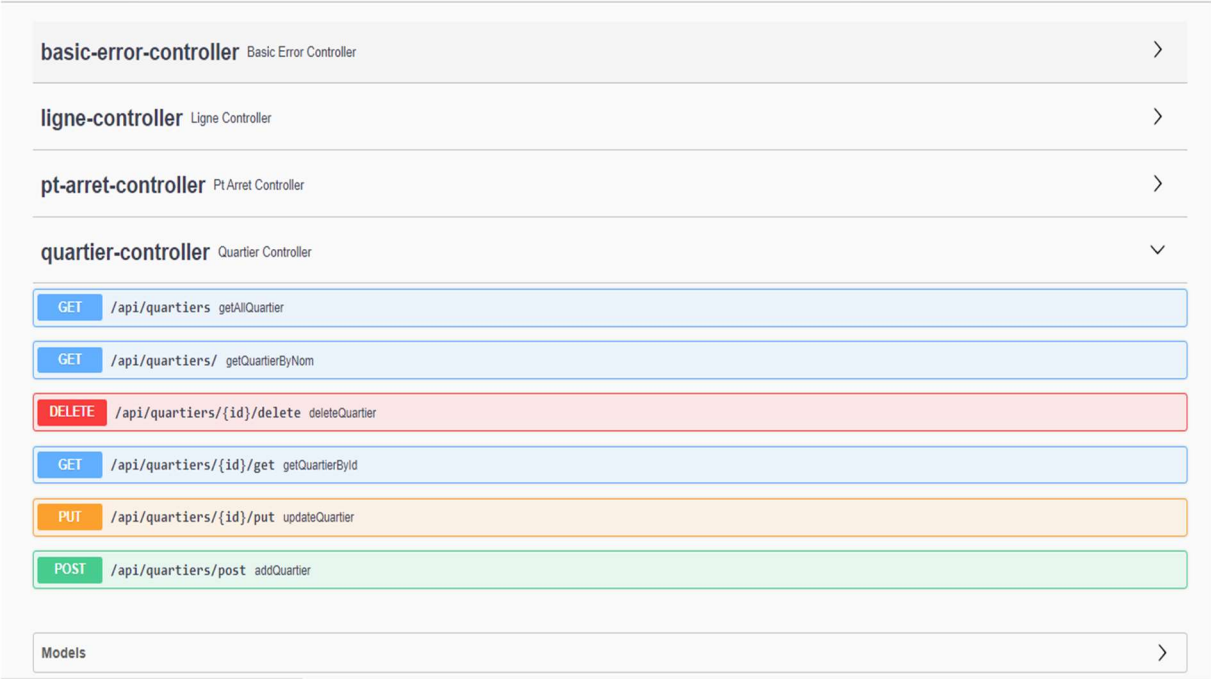

Figure 24: Documentation de l'API(pour les quartiers)

3.4.4. Quelques états et statistiques

- **Lignes de transport**

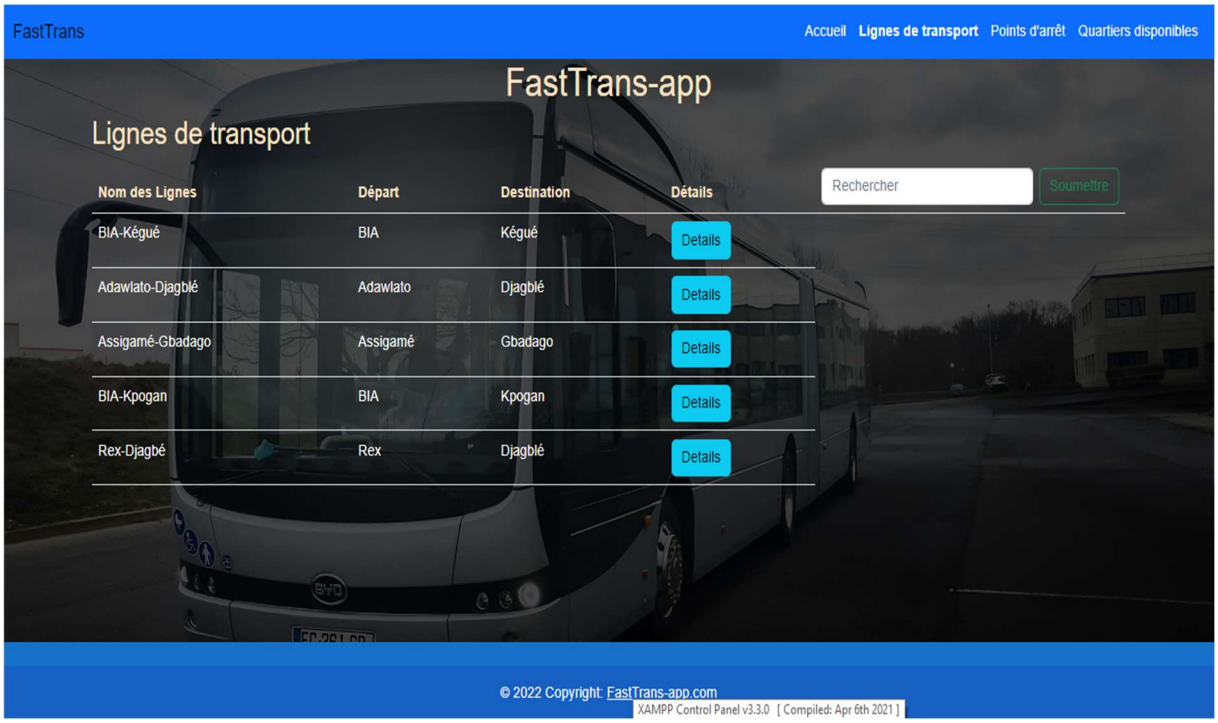

Figure 25: Etat de la liste des lignes de transport

❖ Points d'arrêt

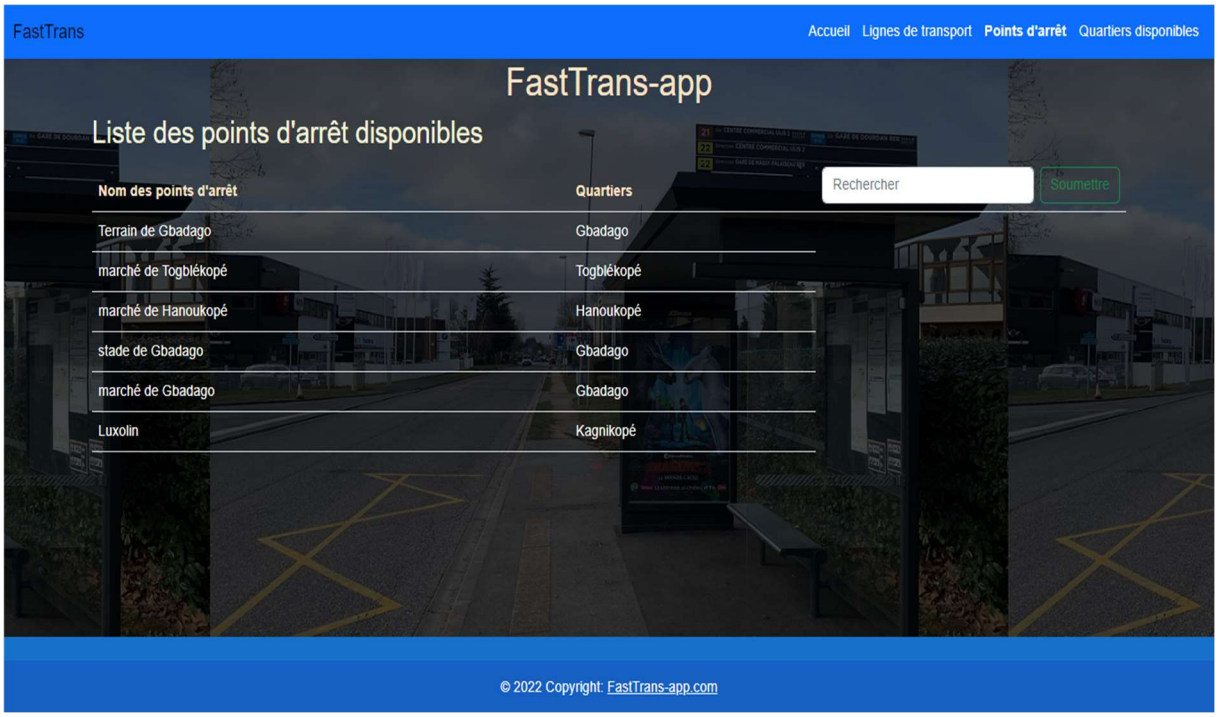

Figure 26: Etat de la liste des points d'arrêt

❖ Quartiers

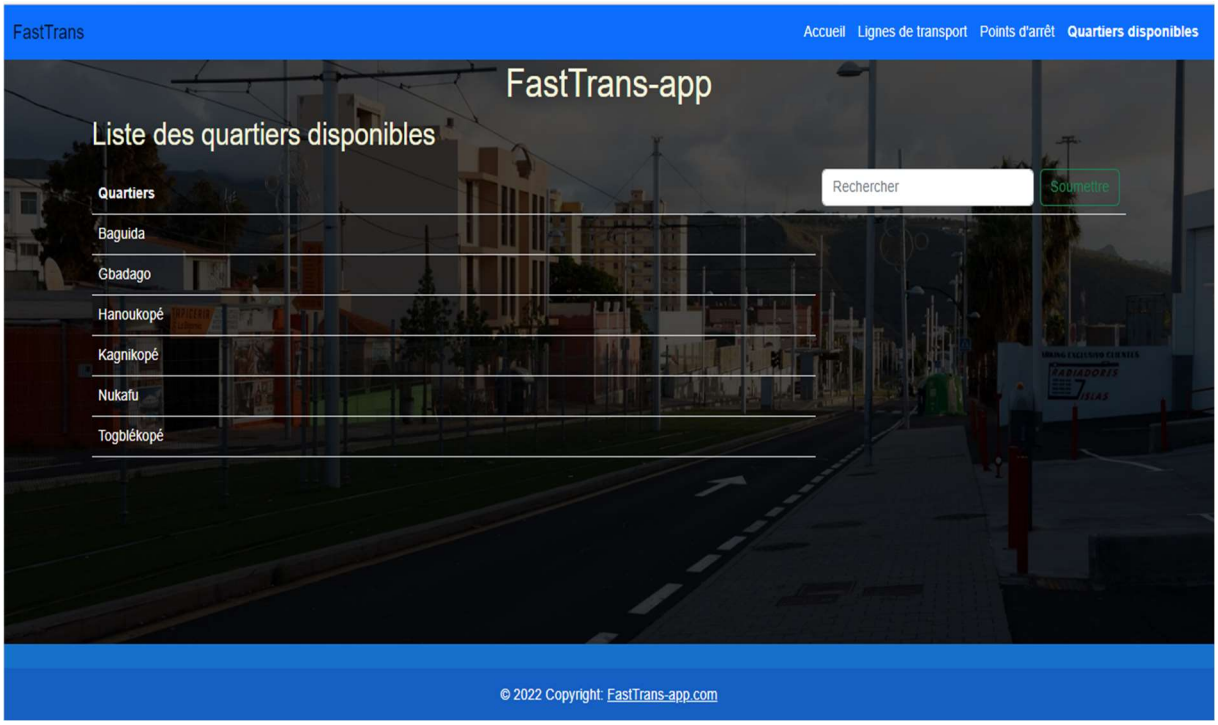

Figure 27: Etat de la liste des quartiers

❖ Détails des lignes de transport

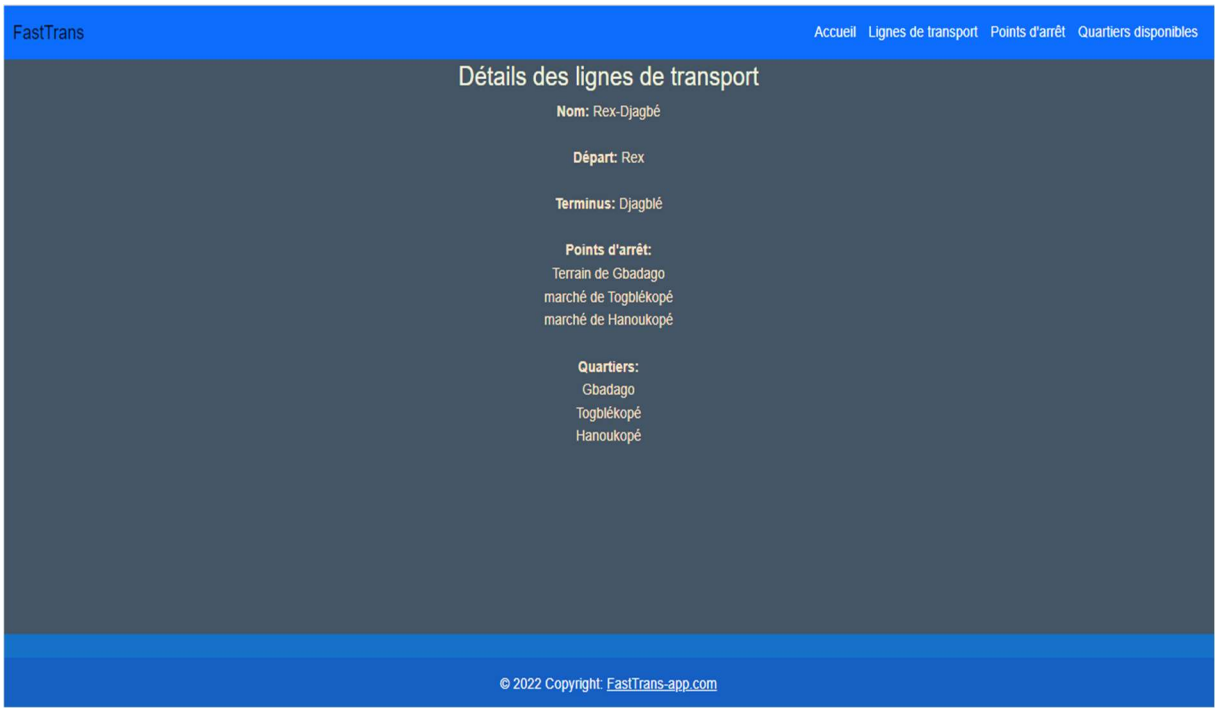

Figure 28: Etat du détail des lignes de transport

❖ Accueil

Figure 29: Page d'accueil

Conclusion :

En somme, ce stage de programmation a été pour nous l'occasion de mettre en pratique les connaissances acquises au cours des deux années de formation. L'objectif du projet que nous avons réalisé était de mettre en place une API afin de gérer les lignes de transport. Nous avons étudié le problème, trouvé une solution que nous avons implémenté en utilisant le framework SpringBoot de JAVA, Angular pour la programmation, UML pour la modélisation combinée avec MySQL, phpMyAdmin et XAMPP pour les bases de données. Tout ceci nous a permis d'accroître nos connaissances en analyse et programmation. Les divers problèmes et erreurs rencontrés au cours de ce projet nous ont forgé et servi de leçons pour la réalisation de nos futurs projets.

BIBLIOGRAPHIE INDICATIVE

- **Ouvrages :**
  - Apprenez spring-boot, Alexandre Iwanesko
  - Créer L'API avec les bons starters, OpenClassrooms, Romain Sessa
  - Débutez avec Angular, OpenClassrooms, Will Alexander
- **Notes de cours :**
  - Cours de UML dispensé par M. SEWAVI Kokou Maurice, 2021-2022
  - Cours d'implémentation de base de données dispensé par M. GBODUI Roland, 2021-2022
  - Cours de JAVA dispensé par M. N'SOUGAN Folly Woèdè, 2021-2022
  - Cours de Conception de Base de données dispensé par M. TIDJANI Ganiou, 2021-2022
- **WEBOGRAPHIE INDICATIVE :**
  - https://www.baeldung.com/swagger-2-documentation-for-spring-rest-api,(tout le long du projet)
  - https://start.spring.io/ , (tout le long du projet)
  - https://spring.io/projects/spring-security, (tout le long du projet)
  - https://angular.io/ , (tout le long du projet)
  - https://www.youtube.com(tout le long du projet)
  - https://youtu.be/YehZI9Hbtn8 (tout le long du projet)
  - https://youtu.be/Gx4iBLKLVHk, (tout le long du projet)
  - https://youtu.be/_rMAnZIcRiU, (tout le long du projet)
  - https://youtu.be/mrU0w28aOTU, (tout le long du projet)
  - https://youtu.be/9SGDpanrc8U , (tout le long du projet)
  - https://youtu.be/sm-8qfMWEV8, (tout le long du projet)
  - https://youtu.be/VVn9OG9nfH0, (tout le long du projet)
- **Anciens rapports de stage consultés :**
  - Rapport de stage de Mme BANAVAÏ Roumanatou (2019-2020) du thème : plateforme centralisée de transport routier et de réservation de tickets en milieu interurbain.

- Rapport de stage de Mme OUKPEDJO Amirah (2019-2020) du thème : mise en place d'un système de géolocalisation et de réservation de places dans un restaurant.

TABLE DES MATIÈRES