Projet Services Web
Manuel développement

Ferreira Marilyne
Moussa Ousmane Issoufou
Mpati Landry

Table des matières

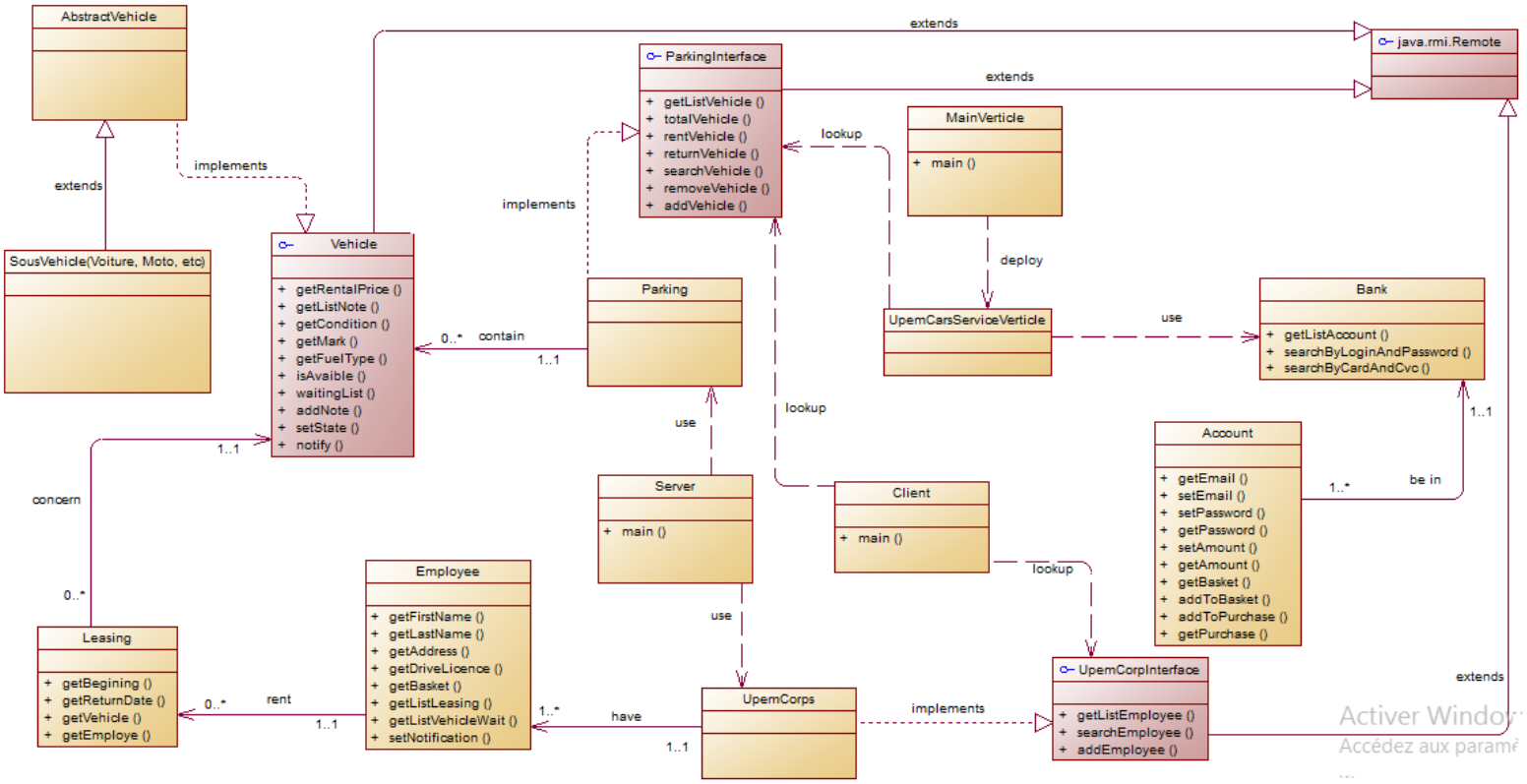
1. Introduction

Pour répondre aux besoins exprimés dans le sujet du projet, nous avons utilisé plusieurs outils et Framework notamment dans la partie web service. Nous avons développé le web service en REST. Afin de séparer les deux (2) parties du projet, nous avons développé chacune de ces 2 parties dans un package dédié. Le projet requière JAVA 11 pour compiler.

2. Conception

Ci-dessous notre diagramme de classe. Pour des raisons de lisibilité sur ce diagramme :

- ✓ Certaines méthodes ne figurent pas dans l'interface Vehicle ainsi que les paramètres des méthodes.
- ✓ Les classes implémentant les interfaces ainsi que les classes filles dans le cas de l'héritage ne redéfinissent pas les méthodes de ces interfaces ou des classes mères.
- ✓ D'autres classes du package rmi tel que AppWindow, Basket, JOptionPaneMultiInput, ListVehicles, NoteWindow, et RentedVehicle ne sont représentées sur ce diagramme car elles servent juste à créer des composants swing.



3. Réalisation

1. RMI

Tout le code lié à cette partie se trouve dans le package rmi.

Pour la développer nous avons procédé comme suit :

- Le registry est créé dans la classe Server du package rmi
- Nous avons utilisé SWING pour réaliser l'interface utilisateur de location des voitures.

2. REST

Le serveur qui expose les véhicules à vendre récupère ces véhicules depuis le serveur RMI.

1. Backend

Nous avons utilisé les outils suivants :

- Vert.x : un Framework qui permet de créer des serveurs web et des routes REST de façon aisée. En plus ce Framework supporte nativement des objets JSON permettant de créer des données dans ce format. Il faut ajouter les jars de Vert.x dans le classpath du projet soit manuellement soit via un outil dédié comme maven ou gradle.

- Currencylayer : une API pour la conversion de devise en temps réel. Nous avons pris la version gratuite de cette API qui a un nombre d'appel par mois limité pour implanter le service de choix de devise.

2. Frontend (requière l'installation de node.js)

Nous avons utilisé les outils suivant pour construire l'interface web permettant d'effectuer des achats :

- Reactjs : une librairie permettant de construire des interfaces web orientées composant et performantes.
- Material-ui : une librairie react qui fournit des composants react prêt à l'emploi
- TypeScript : un langage permettant de typer les données javascript. Nous l'avons utilisé pour typer nos classes react et variables.
- Axios : un client HTTP. Nous l'avons utilisé pour appeler les routes de notre backend.

Tous les composants du frontend se trouve dans le sous-répertoire src du répertoire UPEMCarsService.

4. Conclusion

De manière objective, le regard que nous portons sur l'état final du projet est satisfaisant. La totalité des fonctionnalités ont été implémentés avec succès, Certaines parties du code ont été particulièrement soignés, aussi bien dans le code que dans le design. La cohésion du groupe nous a permis de mener à bien ce projet.