



HONORIS UNITED UNIVERSITIES

2021 / 2022

SPÉCIALITÉ :

Systèmes Informatiques et Mobiles

Application Mobile : SPN – Cars

Réalisé par : Alaa Abdelbaki

Encadré par : Swiss Premium Negoce

Encadrant ESPRIT : Mme. Nouha Samet

Encadrant Entreprise : Mr. Ghaith Ben Abdessalem

SPN<sup>®</sup>

Swiss Premium Negoce  
GENÈVE

# Remerciements

Au terme de ce projet, je tiens à adresser mes remerciements les plus sincères et à exprimer ma gratitude envers mon encadrant pédagogique Madame Nouha Samet dont l'assistance et la disponibilité étaient présentes tout au long du stage.

Je la remercie aussi pour ses directives et els conseils qu'elle m'a prodiguée pour atteindre les objectifs du stage dans les délais convenus. Je dois aussi une grande partie de mon travail à Monsieur Ghaith Ben Abdessalem. Je le remercie pour pour sa disponibilité, ses remarques constructives qui m'ont aidé à surmonter beaucoup de difficultés et à améliorer les fonctionnalités de l'application ainsi que ses qualités humaines d'écoute et de compréhension tout au long de ce travail.

J'exprime également ma gratitude envers tous les membres de la société Swiss Premium Negoce qui m'ont apporté leur soutien et leur savoir faire. J'ai découvert dans ce service une équipe jeune, dynamique et conviviale qui est devenue, pour moi, une famille. Je remercie également la directrice de département IT de la société Madame Jihene Ben Abderrazek pour sa disponibilité et ses conseils qui ont été très bénéfiques tout au long de ce stage.

Je tiens également à remercier tous les enseignants qui ont participé à mon évolution scientifique durant ses quatres années que j'ai passé au sein de ESPRIT. Je remercie finalement les membres du jury en espérant qu'ils apprécient ce rapport.

*Alaa Abdelbaki*

---

## ***Table des matières***

---

<b>Introduction générale</b>	<b>1</b>
<b>1 Cadre du projet</b>	<b>3</b>
1.1 Introduction . . . . .	4
1.2 Présentation de l'organisme d'accueil . . . . .	4
1.2.1 Présentation générale . . . . .	4
1.2.2 Activités . . . . .	4
1.3 Présentation du projet . . . . .	5
1.3.1 Présentation générale . . . . .	5
1.3.2 Problématique . . . . .	5
1.3.3 Étude de l'existant . . . . .	5
1.3.4 Critique de l'existant . . . . .	7
1.3.5 Solution Proposée . . . . .	8
1.4 Méthodologie de travail . . . . .	9
1.5 Conclusion . . . . .	11
<b>2 Conception du projet</b>	<b>13</b>
2.1 Introduction . . . . .	14
2.2 Spécification des besoins . . . . .	14
2.2.1 Besoins fonctionnels . . . . .	14
2.2.2 Besoins non fonctionnels . . . . .	15

2.3	Diagrammes UML . . . . .	15
2.3.1	Diagramme de cas d'utilisation . . . . .	15
2.3.2	Création de compte . . . . .	16
2.3.3	Authentification . . . . .	17
2.3.4	Gérer profil . . . . .	18
2.3.5	Créer réservation . . . . .	19
2.3.6	Gérer réservation . . . . .	20
2.3.7	Diagramme de classes . . . . .	22
2.3.8	Diagrammes de séquences . . . . .	23
2.4	UI/UX Design . . . . .	31
2.5	Conclusion . . . . .	34
<b>3</b>	<b>Technologies et outils utilisés</b>	<b>35</b>
3.1	Introduction . . . . .	36
3.2	Conception des interfaces utilisateur . . . . .	36
3.2.1	Adobe Xd . . . . .	36
3.3	Développement de l'application mobile . . . . .	37
3.3.1	Flutter . . . . .	37
3.4	Développement du serveur back-end . . . . .	38
3.4.1	Express JS . . . . .	38
3.4.2	MongoDB . . . . .	40
3.4.3	JSON Web Token (JWT) . . . . .	41
3.4.4	Firebase . . . . .	42
3.4.5	Swagger . . . . .	43
3.5	Outils et logiciels . . . . .	43
3.5.1	Visual Studio Code . . . . .	43
3.5.2	Postman . . . . .	45
3.5.3	Git . . . . .	46
3.6	Conclusion . . . . .	46
<b>4</b>	<b>Réalisation de l'application</b>	<b>47</b>
4.1	Introduction . . . . .	48
4.2	Création de compte . . . . .	48

<i>TABLE DES MATIÈRES</i>	5
4.2.1    Création de compte avec email et mot de passe . . . . .	48
4.2.2    Création de compte avec Google ou Apple . . . . .	49
4.3    Authentification . . . . .	49
4.4    Page d'accueil . . . . .	51
4.5    Gestion de profil . . . . .	52
4.6    Demander un service . . . . .	52
4.7    Affichage des voitures disponibles . . . . .	54
4.8    Signature numérique de contrat de location . . . . .	54
4.9    Paiement . . . . .	57
4.10    Localiser une voiture . . . . .	57
4.11    Messagerie instantanée . . . . .	58
4.12    Documentation des API . . . . .	59
4.13    Conclusion . . . . .	63
<b>Conclusion et perspectives</b>	<b>65</b>



---

## ***Table des figures***

---

1.1	Logo Swiss Premium Negoce . . . . .	4
1.2	Logo Sixt . . . . .	5
1.3	Logo Blacklane . . . . .	6
1.4	Logo Uber . . . . .	6
1.5	Sélection de voiture avec Uber . . . . .	8
1.6	Sélection de voiture avec Blacklane . . . . .	8
1.7	Positionnement de l'application SPN-Cars par rapport aux alternatives.	9
1.8	Méthode en cascade . . . . .	11
2.1	Diagramme de cas d'utilisation généralisé . . . . .	16
2.2	Diagramme de cas d'utilisation spécifique : Création de compte. . . . .	17
2.3	Diagramme de cas d'utilisation spécifique : Authentification. . . . .	18
2.4	Diagramme de cas d'utilisation spécifique : Gestion de profil. . . . .	19
2.5	Diagramme de cas d'utilisation spécifique : Créeer réservation. . . . .	20
2.6	Diagramme de cas d'utilisation spécifique : Gérer réservation. . . . .	21
2.7	Diagramme de classe . . . . .	22
2.8	Diagramme de séquence : Création de compte avec E-mail et mot de passe. . . . .	23
2.9	Diagramme de séquence : Création de compte avec un compte Google / Apple. . . . .	24
2.10	Diagramme de séquences : Authentification. . . . .	25

2.11	Diagramme de séquences : Page d'accueil. . . . .	26
2.12	Diagramme de séquences : Demander une location. . . . .	27
2.13	Diagramme de séquences : Demander un transfert. . . . .	28
2.14	Diagramme de séquences : Choisir une voiture . . . . .	29
2.15	Diagramme de séquences : Envoi du contrat de location. . . . .	30
2.16	Diagramme de séquences : Paiement des services. . . . .	31
2.17	Première page. . . . .	32
2.18	Page de connexion. . . . .	32
2.19	Page d'accueil. . . . .	32
2.20	Page d'accueil lors d'un transfert en cours. . . . .	32
2.21	Sélection de type de transfert. . . . .	33
2.22	Suivi de la position actuelle du chauffeur avec la voiture. . . . .	33
2.23	Liste des voitures disponibles. . . . .	33
2.24	Détails de la voiture sélectionnée. . . . .	33
2.25	Messagerie instantanée avec le chauffeur. . . . .	34
3.1	Logo Adobe Xd . . . . .	36
3.2	Logo Flutter . . . . .	37
3.3	Logo Express . . . . .	38
3.4	Logo MongoDB . . . . .	40
3.5	Logo JWT . . . . .	41
3.6	Logo Firebase . . . . .	42
3.7	Logo Swagger . . . . .	43
3.8	Logo Visual Studio Code . . . . .	43
3.9	Logo Postman. . . . .	45
3.10	Logo Git. . . . .	46
4.1	Formulaire de création de compte : 1ère étape. . . . .	49
4.2	Formulaire de création de compte : 2ème étape. . . . .	49
4.3	Page de Login. . . . .	50
4.4	Validation des champs de Login. . . . .	50
4.5	Login avec Google. . . . .	51
4.6	Formulaire de demande de transfert. . . . .	53
4.7	Utiliser le bouton «Localiser» pour choisir la position actuelle. . . . .	53

*TABLE DES FIGURES*

9

4.8	Rechercher un emplacement à l'aide de Google Places. . . . .	53
4.9	Afficher la meilleure route entre le point de départ et le point d'arrivée. . . . .	53
4.10	Liste des voitures disponibles selon le point de départ sélectionné. . . . .	54
4.11	Affichage si aucune voiture n'est disponible. . . . .	54
4.12	Création de réservation. . . . .	55
4.13	Page de vérification de signature . . . . .	55
4.14	Vérification de signature avec un document non signé. . . . .	56
4.15	Email contenant les documents à signer. . . . .	56
4.16	Choix de signature. . . . .	56
4.17	confirmation de signature. . . . .	56
4.18	Interface de paiement. . . . .	57
4.19	Détails de paiement. . . . .	57
4.20	Affichage des détails de la réservation. . . . .	58
4.21	Liste de messages de l'utilisateur. . . . .	59
4.22	Conversation avec l'utilisateur. . . . .	59
4.23	Schéma de documentation des API. . . . .	60
4.24	Documentation des API. . . . .	61
4.25	Détails API ajout voiture. . . . .	62



---

## ***Liste des tableaux***

---

1.1	Comparaison des méthodologies de travail . . . . .	10
2.1	Scénario d'utilisation : Création de compte . . . . .	16
2.2	Scénario d'utilisation : Authentification . . . . .	17
2.3	Scénario d'utilisation : Création de compte . . . . .	18
2.4	Scénario d'utilisation : Créeer une réservation . . . . .	19
2.5	Scénario d'utilisation : Gérer une réservation . . . . .	20
3.1	Tableau comparatif : Flutter et React Native . . . . .	38
3.2	Tableau comparatif : ExpressJS et Spring Boot . . . . .	39
3.3	Tableau comparatif : MongoDB Code et MySQL . . . . .	41
3.4	Tableau comparatif : Visual Studio Code et Android Studio . . . . .	44
3.5	Tableau comparatif : Postman et Insomnia REST Client . . . . .	45

---

## ***Introduction générale***

---

**L**e marché mondial de location de voitures continue, chaque année, à se développer et à s'amplifier, et le besoin d'atteindre un nombre maximal de clients est devenu une nécessité. Surtout avec l'avancement technologique qui a permis à tout le monde d'avoir un smartphone. Cet appareil est capable de réaliser plusieurs fonctionnalités en simples clics.

Pendant les années précédentes, avec la pandémie mondiale de COVID-19, qui a forcé des milliards de personnes à rester chez eux, il est devenu indispensable de digitaliser plusieurs services afin de les rendre plus accessibles, plus fiables, et continuer à exister dans un marché qui continue à évoluer et innover. Plusieurs personnes ont pu se bénéficier de ces services sans quitter leurs maisons et risquer d'être atteintes par la COVID, grâce aux différentes entreprises qui ont choisi de continuer à servir leurs clients à l'aide des applications mobiles.

L'entreprise Swiss Premium Negoce a vu cet événement mondial comme une opportunité pour s'introduire dans le monde vaste de nouvelles technologies et présenter leurs différents services sous forme numérique à l'aide des sites web et applications mobiles. Cette étape qui leur permettra de rester toujours proche de sa clientèle, et continuer à fonctionner sans tenir compte de la pandémie.

L'un des services offerts par Swiss Premium Negoce est le service de location de voitures de luxe. Ce service permet aux utilisateurs de trouver leurs voitures de rêves et la louer même pour une courte durée avec la possibilité d'avoir un chauffeur personnel. Tout cela sera possible à l'aide de l'application Swiss Premium Negoce : Cars ou tout simplement : SPN Cars. Le présent rapport qui présentera cette application, s'étalera sur quatre chapitres :

- Le premier chapitre présentera l'organisme d'accueil, ainsi le projet à réaliser et une étude sur les différents aspects de ce projet.
- Le deuxième chapitre fera l'objet de présenter les différents besoins qui seront achevés par cette application.

- Le troisième chapitre sera à propos la conception de l'application, et les technologies choisies pour réaliser le projet suite à cette application.
- Le quatrième et dernier chapitre présentera la réalisation de ce projet, qui est une explication détaillée de chacune des fonctionnalités offerts par l'application avec des captures d'écran pour illustrer le travail réalisé.

Le projet sera clôturé par une conclusion générale.

# *Chapitre 1*

---

## *Cadre du projet*

---

### **Contenu**

---

<b>1.1</b>	<b>Introduction</b>	.....	4
<b>1.2</b>	<b>Présentation de l'organisme d'accueil</b>	.....	4
1.2.1	Présentation générale	.....	4
1.2.2	Activités	.....	4
<b>1.3</b>	<b>Présentation du projet</b>	.....	5
1.3.1	Présentation générale	.....	5
1.3.2	Problématique	.....	5
1.3.3	Étude de l'existant	.....	5
1.3.4	Critique de l'existant	.....	7
1.3.5	Solution Proposée	.....	8
<b>1.4</b>	<b>Méthodologie de travail</b>	.....	9
<b>1.5</b>	<b>Conclusion</b>	.....	11

---

## 1.1 Introduction

Vu que ce projet de fin d'études sera réalisé au sein d'une entreprise accueillante, il s'avère être indispensable d'avoir une idée sur cette dernière. Par la suite, nous allons mettre l'accent sur le sujet et les motivations derrière ce projet.

## 1.2 Présentation de l'organisme d'accueil

Ce projet a été réalisé au sein du tout nouveau département IT de ***Swiss Premium Negoce*** [1].

Ce département se compose d'une équipe très innovante, très talentueuse qui est responsable des différentes applications et sites web de tous les différents services proposés par SPN.

### 1.2.1 Présentation générale



FIGURE 1.1 – Logo Swiss Premium Negoce

***Swiss Premium Negoce SA*** (Société Anonyme) est une société de conciergerie basée à Genève, Suisse. Créée en 2011 avec passion et enthousiasme, l'entreprise se spécialise en services de luxe. En 2022, SPN a inauguré SPN Tunis avec son nouveau département IT, un département qui permettra d'atteindre plus d'audience à travers la digitalisation des différents services offerts par SPN.

### 1.2.2 Activités

SPN offre plusieurs services de luxe tel que :

- Conciergerie
- Hospitalité de luxe
- Location de voitures
- Santé et services médicaux
- Camps d'été

- Éducation
- Jets privés
- Yachting
- Events

## 1.3 Présentation du projet

### 1.3.1 Présentation générale

L'application SPN-Cars est la solution trouvée par SPN pour offrir l'un de ses services les plus demandés plus facilement et convenablement grâce à l'utilisation de plusieurs nouvelles technologies qui permettront à l'entreprise de mieux performer et atteindre plus d'utilisateurs. On doit alors définir ce projet, son objectif, ses concurrents, les problèmes qu'ils créent et comment l'application SPN-Cars pourra les corriger et offrir un service meilleur que les autres applications disponibles sur le marché.

### 1.3.2 Problématique

L'application SPN-Cars vise à devenir l'un des leaders dans les domaines de location de voiture en ligne, réservation de chauffeurs et des services taxis. Ce qui pose un grand défi : **Comment fournir aux utilisateurs de l'application ses services en tenant compte de la rapidité de l'application ainsi que la facilité des procédures ?**

### 1.3.3 Étude de l'existant

Il existe déjà plusieurs applications qui offrent des services similaires, chacune de ses applications présente des avantages et des inconvénients qu'on les explorera dans les chapitres suivants.

**Sixt**



FIGURE 1.2 – Logo Sixt

Sixt [2] est un fournisseur international de voitures de location, avec plus de 2000 véhicules dans 110 différents pays.

Sixt offre plusieurs services en rapport avec la location des voitures :

- Location des voitures.
- Covoiturage.
- Service taxi.
- Location de voiture par abonnement mensuel.

### **Blacklane**



FIGURE 1.3 – Logo Blacklane

Blacklane [3] est une entreprise allemande qui offre à ses clients la possibilité de réserver un chauffeur avec une voiture luxueuse et vivre une expérience VIP.

L'entreprise offre principalement trois services :

- Trajets longue distance.
- Chauffeurs à la demande.
- Transfert aéroport.

### **Uber**



FIGURE 1.4 – Logo Uber

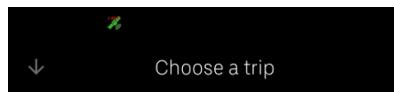
Uber [4] est une entreprise de technologies américaine qui propose à ses clients une solution afin de trouver des chauffeurs pour les transporter vers leurs destination choisie grâce à son application mobile.

Uber a commencé en tant qu'une application qui joue le rôle d'intermédiaire entre ses utilisateurs et ses chauffeurs indépendants qui offrent leurs services.

#### 1.3.4 Critique de l'existant

Même si les exemples mentionnés ci-dessus sont les leaders mondiaux dans le domaine des services de transport, ils sont tous différents. Surtout en termes de qualité de services, du marché visé.

Uber, par exemple, est à la recherche d'attirer un nombre maximal d'utilisateurs, c'est pourquoi l'application offre une sélection très variée de moyens de transport (voire fig. 5). Blacklane, par contre, vise une clientèle plus exclusive, une clientèle qui est prête à payer pour avoir un service de luxe. Ce qui est visible lors de la sélection de voitures à l'aide de l'application Blacklane comme le montre la figure ci-dessous (voir fig. 6)



Popular

	<b>Moto</b> <span>1</span>	<b>€74.67</b>
Professional drivers on high-end motorbikes and maxi scooters		
	<b>Uber Pet</b> <span>3</span>	<b>€59.50</b>
12:05	Affordable rides for you and your pet	
	<b>UberX</b> <span>3</span>	<b>€55.50</b>
12:06	Affordable everyday trips	
	<b>Berline</b> <span>3</span>	<b>€64.64</b>
12:08	Premium trips in luxury cars	
	<b>Comfort</b> <span>3</span>	<b>€64.05</b>
12:05	Comfortable cars, with top rated drivers	
	<b>Van</b> <span>6</span>	<b>€64.64</b>
12:10	Premium rides in spacious cars for groups up to 6	
	<b>Green</b> <span>3</span>	<b>€54.87</b>



Business Class

Mercedes-Benz E-Class or similar  
max. 3 max. 2



- ✓ Includes 15 min of complimentary wait time
- ✓ All-inclusive rates
- ✓ Free cancellation up until 1 hour before pickup

[Learn more](#)



FIGURE 1.5 – Sélection de voiture avec Uber

FIGURE 1.6 – Sélection de voiture avec Blacklane

### 1.3.5 Solution Proposée

L’objectif de l’application SPN-Cars est de fournir à ses utilisateurs des expériences luxueuses en simples clics.

L’application doit être simple et permet aux utilisateurs d’avoir une expérience rapide et satisfaisante.

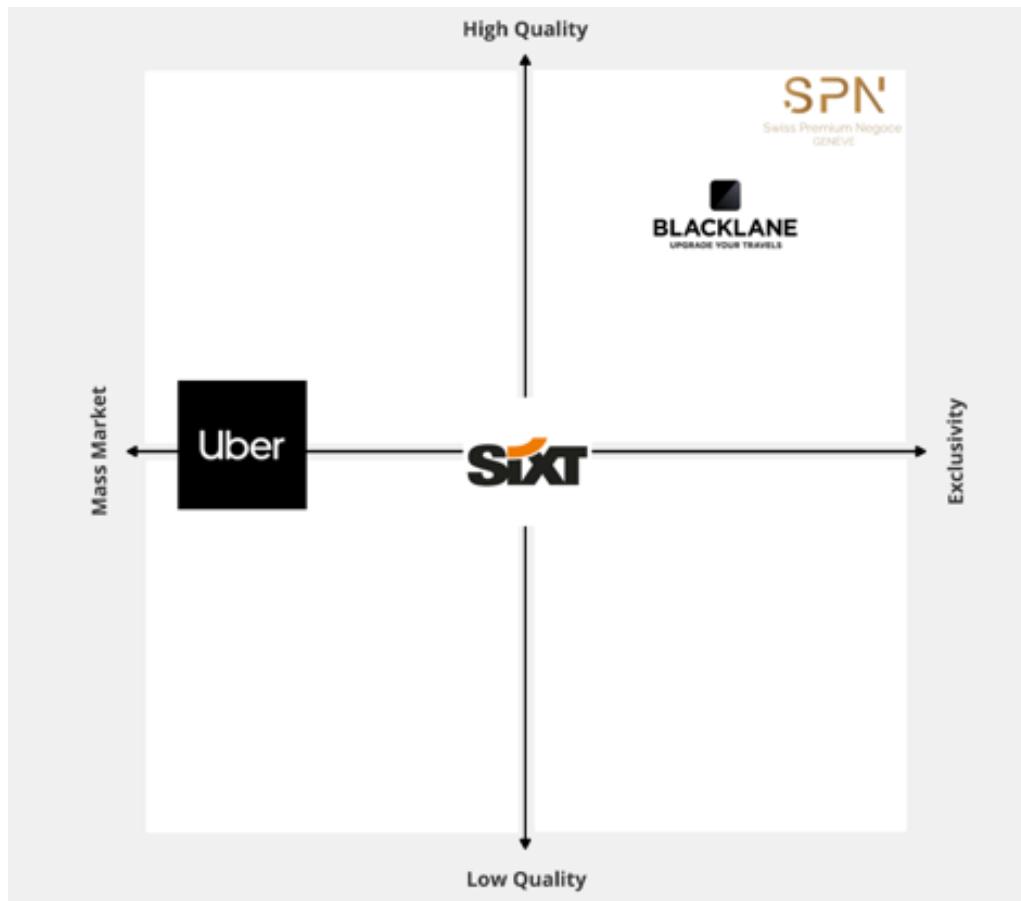


FIGURE 1.7 – Positionnement de l’application SPN-Cars par rapport aux alternatives.

## 1.4 Méthodologie de travail

L’objectif est de pouvoir mener ce projet en respectant les délais et les budgets alloués. Pour atteindre cet objectif, il faut adopter une méthode de gestion de projet. Il existe de nombreuses méthodes à choisir dont on cite :

- **Les méthodes traditionnelles** : Ces méthodes sont les plus utilisées en gestion de projet. Elles sont appelées également «En cascade» car chaque étape, car chaque étape doit être terminée pour passer à la suivante. En appliquant cette méthodologie, l’équipe projet suit le cahier de charges à la lettre et travaille sur la totalité du projet jusqu’à sa livraison. Il n’y a pas d’interaction avec le client qui recevra son projet une fois que celui-ci est terminé.
- **Les méthodes agiles** : Ces méthodes sont moins rigides que les méthodes traditionnelles, elles placent les besoins du client au centre des priorités du projet. Elles offrent une plus grande flexibilité et une meilleure visibilité dans la gestion du projet, ce qui permet à l’équipe d’être plus réactive aux attentes du client. Le projet sera découpé en

mini-projets chacun nécessitant la validation du client pour passer au suivant.

### Comparaison des méthodes

Chaque méthode présente des avantages et des inconvénients, il faut alors les comparer pour identifier la méthode appropriée pour ce projet.

Dans le tableau suivant on présente une comparaison des méthodes :

	Méthodes classiques	Méthodes agiles
Cahier de charges	Un cahier de charge complet doit être disponible.	Se concentrent sur la phase de développement, le cahier de charges est souvent incomplet.
Temps passé	Le client n'est impliqué qu'au phases de conception et livraison de l'application.	L'équipe de développement et le donneur d'ordre communiquent constamment pour échanger sur le projet et planifier les sprints.
Type de projet	Ces méthodes sont préférées pour les projets complexes où rien n'est laissé au hazard.	Cette méthode convient aux projets incertains et innovants.

TABLE 1.1 – Comparaison des méthodologies de travail

### Méthode retenue

Comme pour ce projet il existe un cahier de charges bien défini, et un temps de développement très strict, le choix d'une méthode classique, en particulier la méthode en cascade, est imminent.

La méthode en cascade est une méthode linéaire des différentes phases du projet nécessaires

pour la création du livrable.

La méthode en cascade est composée de six étapes :

- **Étape des exigences** : C'est la première étape où on définit les exigences et les besoins.
- **Étape de l'analyse** : Faire une analyse plus approfondie des solutions qui répondent aux besoins et écarter celles qui ne correspondent pas aux exigences.
- **Étape de conception** : Dans cette étape on va définir d'une façon détaillée les différents besoins.
- **La mise en œuvre** : Une fois tout est défini et accepté par tous les participants, on commence la partie de mise en œuvre où on développe les différentes fonctionnalités.
- **La validation** : Cette étape consiste à tester les fonctionnalités développées dans l'étape précédente une par une et corriger les problèmes rencontrés.
- **La mise en service** : À la fin de l'étape de validation le projet est mis en service.

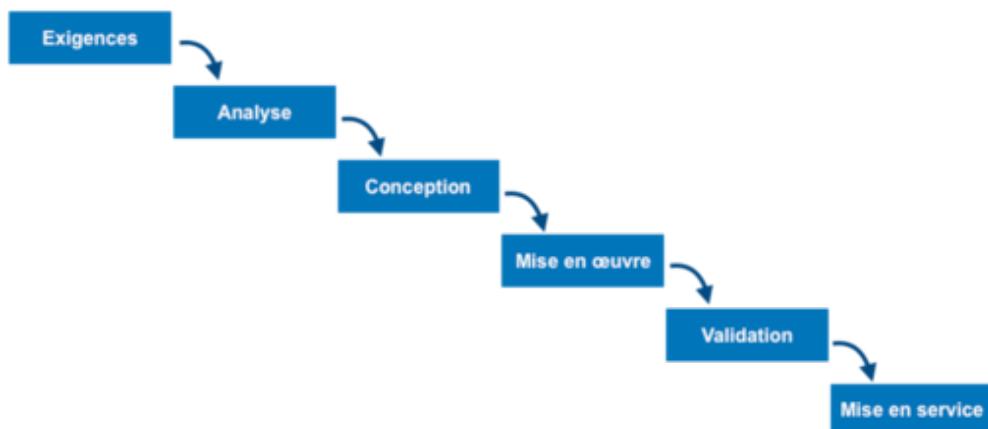


FIGURE 1.8 – Méthode en cascade .

[5]

## 1.5 Conclusion

Dans ce chapitre, on a commencé par la présentation de l'entreprise accueillante et ses différentes activités. Puis on a présenté le projet offert et la problématique qu'il pose. Suite à la problématique, il fallait faire une étude de l'existant où on a pris l'exemple de différentes applications qui offrent un service similaire et déduisent leurs faiblesses. Une fois ces défauts dégagés, on a pu passer à la présentation de la solution proposée qui permettra de résoudre les points manquants des autres applications. Finalement on a présenté la méthodologie de travail suivie qui assurera la fluidité au niveau de la gestion de ce projet.

Suite à ce chapitre, on passera vers l'étape de conception où on définira les différents aspects du projet.

## *Chapitre 2*

---

### *Conception du projet*

---

#### **Contenu**

---

<b>2.1</b>	<b>Introduction</b>	.....	<b>14</b>
<b>2.2</b>	<b>Spécification des besoins</b>	.....	<b>14</b>
2.2.1	Besoins fonctionnels	.....	14
2.2.2	Besoins non fonctionnels	.....	15
<b>2.3</b>	<b>Diagrammes UML</b>	.....	<b>15</b>
2.3.1	Diagramme de cas d'utilisation	.....	15
2.3.2	Création de compte	.....	16
2.3.3	Authentification	.....	17
2.3.4	Gérer profil	.....	18
2.3.5	Créer réservation	.....	19
2.3.6	Gérer réservation	.....	20
2.3.7	Diagramme de classes	.....	22
2.3.8	Diagrammes de séquences	.....	23
<b>2.4</b>	<b>UI/UX Design</b>	.....	<b>31</b>
<b>2.5</b>	<b>Conclusion</b>	.....	<b>34</b>

---

## **2.1 Introduction**

Afin de créer une application respectant les besoins définis dans le cahier de charges, il faut d'abord passer par l'étape de conception. Cette étape permettra de décortiquer le cahier de charges et définir les acteurs, les cas d'utilisation, et finalement les différentes classes qui définissent les différents attributs de chaque entité qui interagit avec l'application.

## **2.2 Spécification des besoins**

La première étape de la conception est de dégager les différents besoins de l'utilisateur. Ces besoins sont divisés en deux catégories : fonctionnels et non fonctionnels.

### **2.2.1 Besoins fonctionnels**

#### **L'authentification**

Le client doit s'authentifier pour utiliser les différents services de l'application.

#### **Consulter les voitures disponibles**

Le client peut consulter les voitures disponibles selon sa position actuelle.

#### **Créer une réservation**

Le client peut créer une réservation en choisissant une voiture.

#### **Paiement en ligne**

Le client peut payer pour sa réservation à l'aide des méthodes de paiement en ligne.

#### **Signature numérique de contrat de location**

Le client peut signer son contrat de location en ligne en toute sécurité.

#### **Suivre la position de la voiture**

Le client peut suivre la position de la voiture louée en temps réel.

#### **Contacter le chauffeur**

Le client peut contacter le chauffeur de sa voiture louée par appel téléphonique ou messagerie instantanée.

## 2.2.2 Besoins non fonctionnels

### Interfaces graphiques

L'utilisateur peut rapidement commencer à utiliser l'application sans difficulté à l'aide d'interfaces graphiques claires et simples.

### Performances de l'application

L'application ne doit pas présenter des imperfections en termes de rapidité et de fluidité en cours de son exécution.

### Maintenabilité et scalabilité

L'application doit être simple à maintenir et facilement scalable dans le futur.

### Sécurité

Toutes les informations traitées dans l'application doivent être protégées, et une vérification de l'utilisateur est impérative avant chaque opération.

## 2.3 Diagrammes UML

### 2.3.1 Diagramme de cas d'utilisation

L'application SPN-Cars a un seul acteur qui est l'utilisateur, ce dernier utilisera toutes les fonctionnalités offertes par l'application.

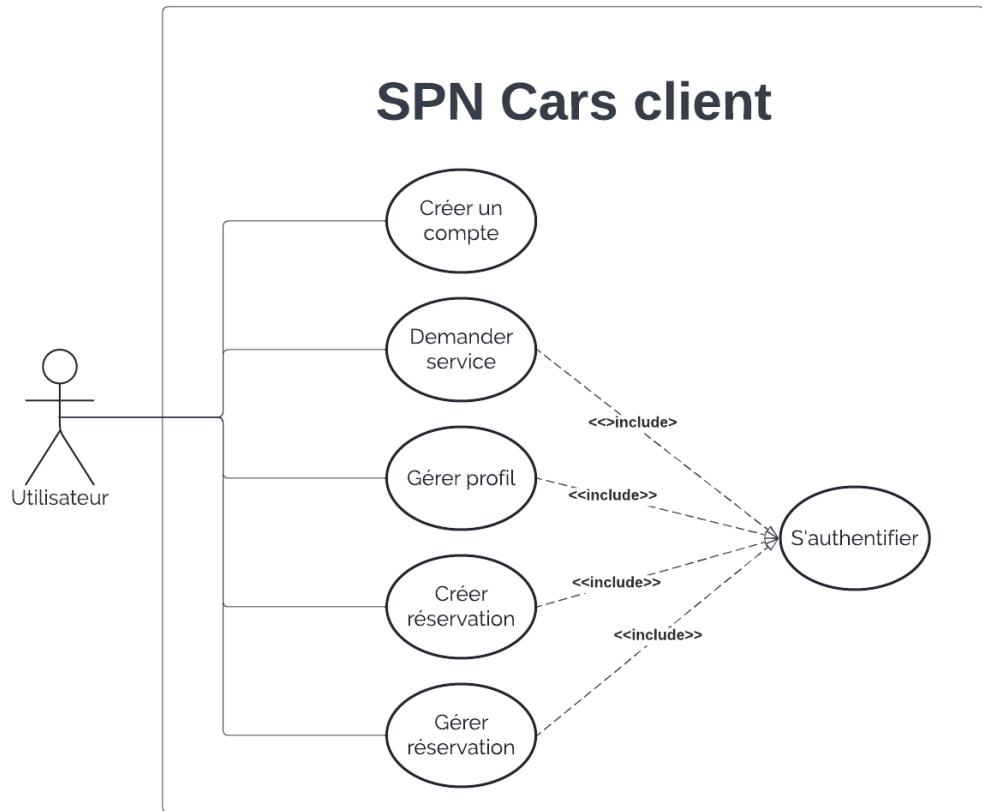


FIGURE 2.1 – Diagramme de cas d'utilisation généralisé

### 2.3.2 Crédation de compte

Avant d'utiliser les services offerts par l'application par l'application SPN Cars, l'utilisateur doit créer un compte.

Cas d'utilisation	Scénario optimal	Scénario d'exception
Création de compte	Compte créé avec succès	Compte non créé suite à une erreur : Email déjà utilisé / Serveur hors ligne.

TABLE 2.1 – Scénario d'utilisation : Crédation de compte

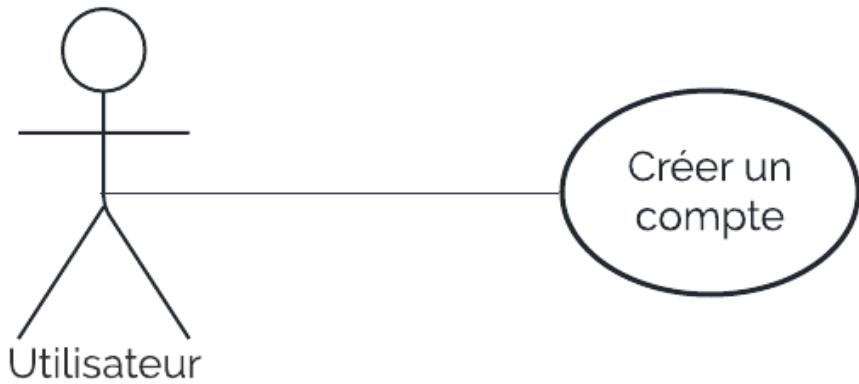


FIGURE 2.2 – Diagramme de cas d'utilisation spécifique : Crédit de compte.

### 2.3.3 Authentification

Pour accéder aux différents services de l'application SPN-Cars, l'utilisateur doit s'authentifier. Cette étape consiste à vérifier les informations de l'utilisateur dans la base de données.

Cas d'utilisation	Scénario optimal	Scénario d'exception
Authentification	Authentification réussie et un token d'authentification est retourné.	Retour d'un erreur : Coordonnées incorrectes / méthode de connexion incorrecte (Ex : Utilisation email et mot de passe au lieu du bouton Google ou Apple).

TABLE 2.2 – Scénario d'utilisation : Authentification



FIGURE 2.3 – Diagramme de cas d'utilisation spécifique : Authentification.

### 2.3.4 Gérer profil

L'utilisateur peut gérer son profil : Il peut changer sa photo de profil et ses informations personnelles.

Cas d'utilisation	Scénario optimal	Scénario d'exception
Changement de photo de profil	Photo ajoutée avec succès	Photo non ajoutés : Connexion au serveur échouée.
Changer les informations personnelles	Informations changées avec succès	Informations non changées : Erreur de connexion vers le serveur .

TABLE 2.3 – Scénario d'utilisation : Crédation de compte

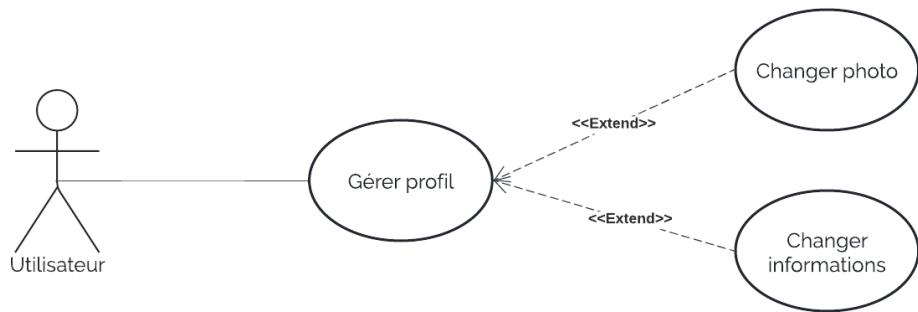


FIGURE 2.4 – Diagramme de cas d'utilisation spécifique : Gestion de profil.

### 2.3.5 Créer réservation

Pour créer sa réservation, l'utilisateur doit louer une voiture, effectuer un paiement et signer le contrat numérique de location.

Cas d'utilisation	Scénario optimal	Scénario d'exception
Louer une voiture	Voiture louée avec succès.	Voiture non louée : Voiture non disponible / Erreur de connexion.
Effectuer paiement	Paiement effectué avec succès.	Paiement non effectué : Solde insuffisant, Service de paiement indisponible.
Signer contrat	Contrat signé avec succès.	Contrat non signé par l'utilisateur.

TABLE 2.4 – Scénario d'utilisation : Créer une réservation

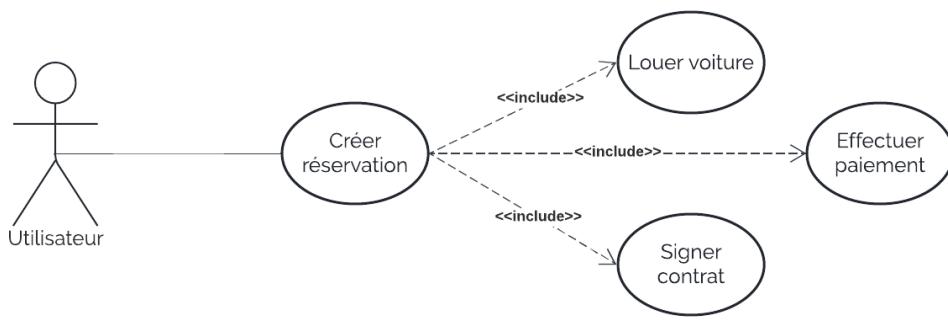


FIGURE 2.5 – Diagramme de cas d'utilisation spécifique : Créer réservation.

### 2.3.6 Gérer réservation

L'utilisateur peut gérer sa réservation, il peut suivre la position de la voiture louée en temps réel, et il peut aussi annuler sa réservation.

Cas d'utilisation	Scénario optimal	Scénario d'exception
Suivi véhicules	Véhicule suivie.	Véhicule non suivie : Signal GPS faible / problème de connexion avec chauffeur.
Annuler réservation	Réservation annulée avec succès.	Réservation non trouvée / Problème de connexion avec le serveur.

TABLE 2.5 – Scénario d'utilisation : Gérer une réservation

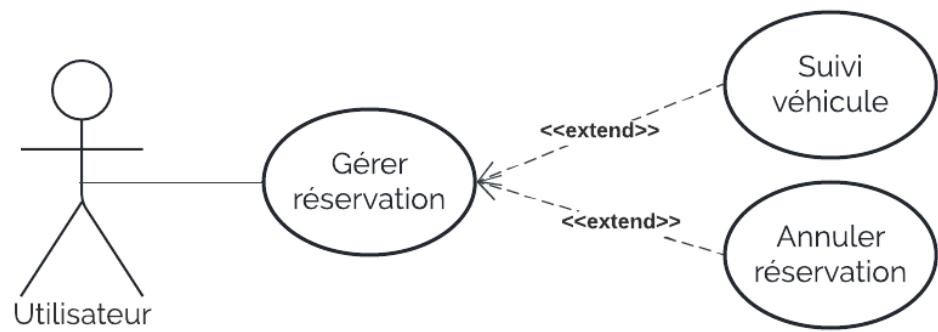


FIGURE 2.6 – Diagramme de cas d'utilisation spécifique : Gérer réservation.

### 2.3.7 Diagramme de classes

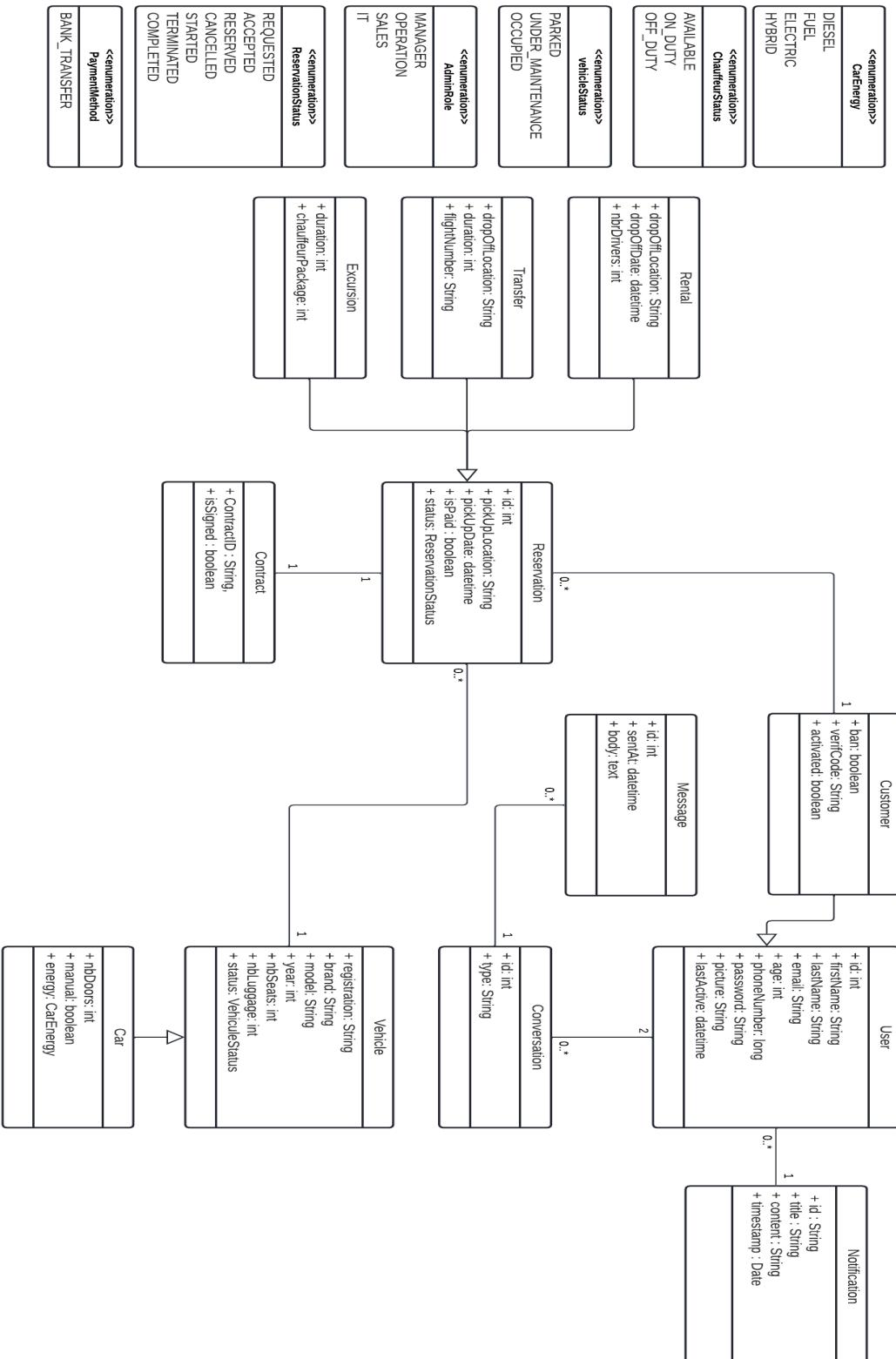


FIGURE 2.7 – Diagramme de classe

### 2.3.8 Diagrammes de séquences

Les diagrammes de séquences sont le moyen qui permettent de décrire d'une manière détaillée les différents cas d'utilisation

#### Création de compte

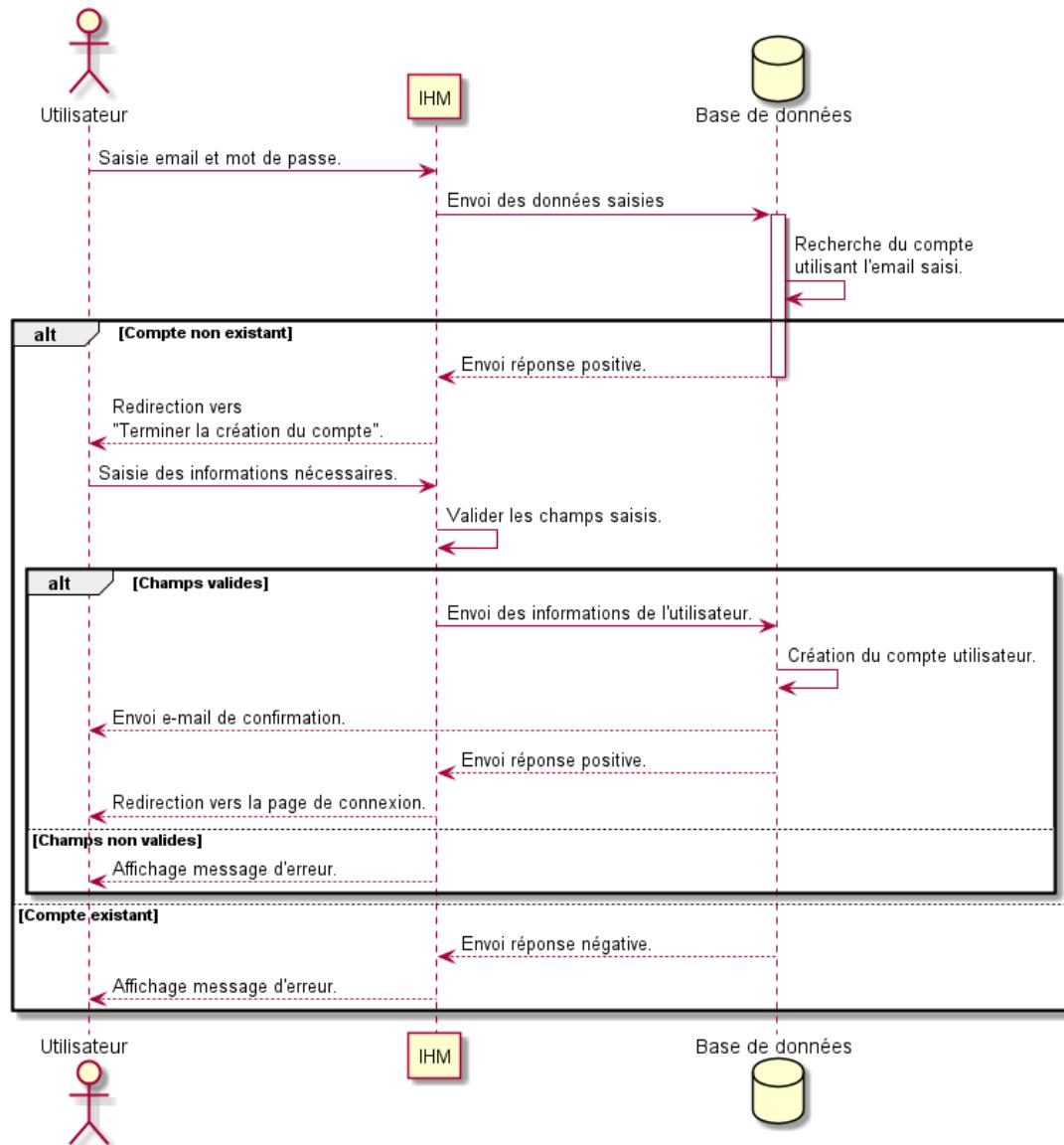


FIGURE 2.8 – Diagramme de séquence : Crédit de compte avec E-mail et mot de passe.

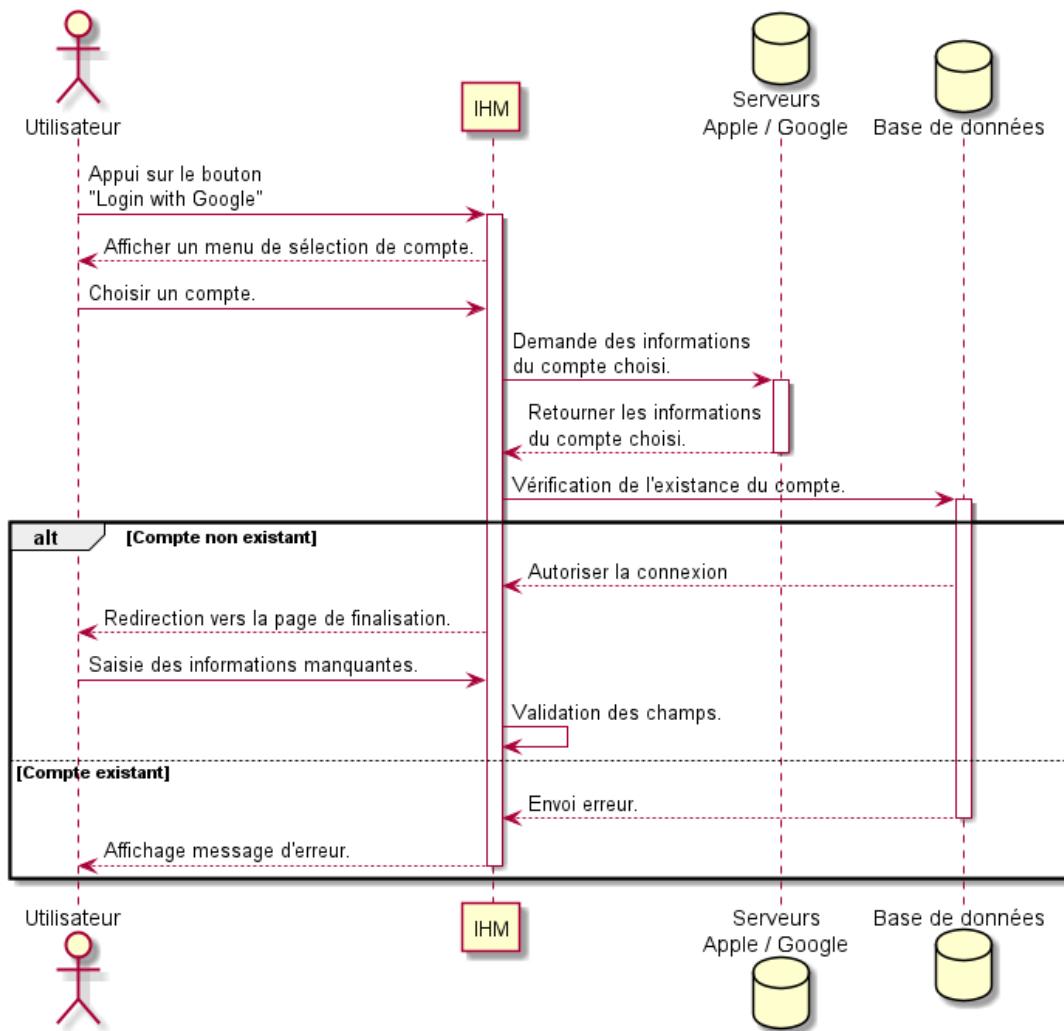


FIGURE 2.9 – Diagramme de séquence : Création de compte avec un compte Google / Apple.

## Authentification

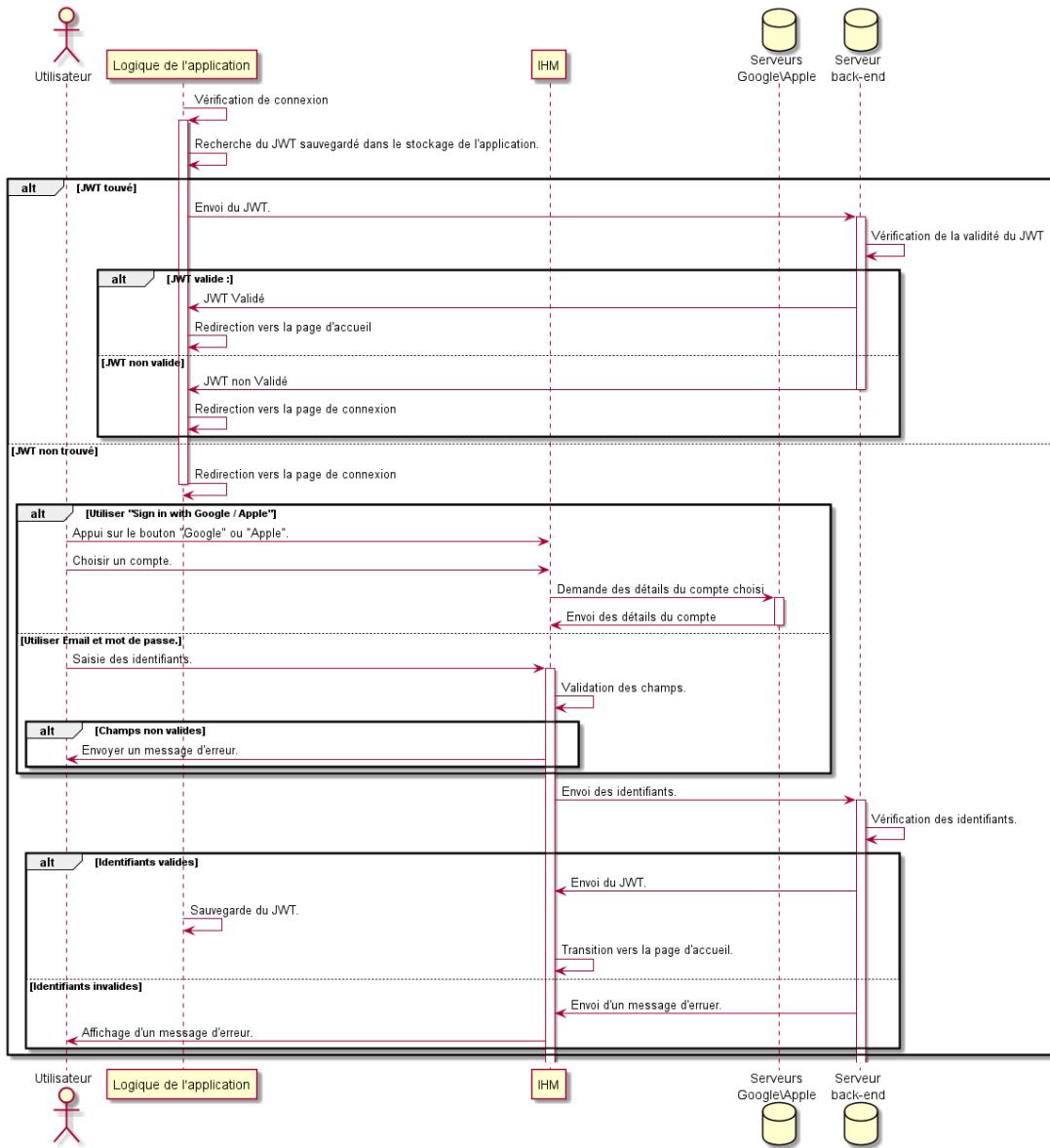


FIGURE 2.10 – Diagramme de séquences : Authentification.

### Page d'accueil

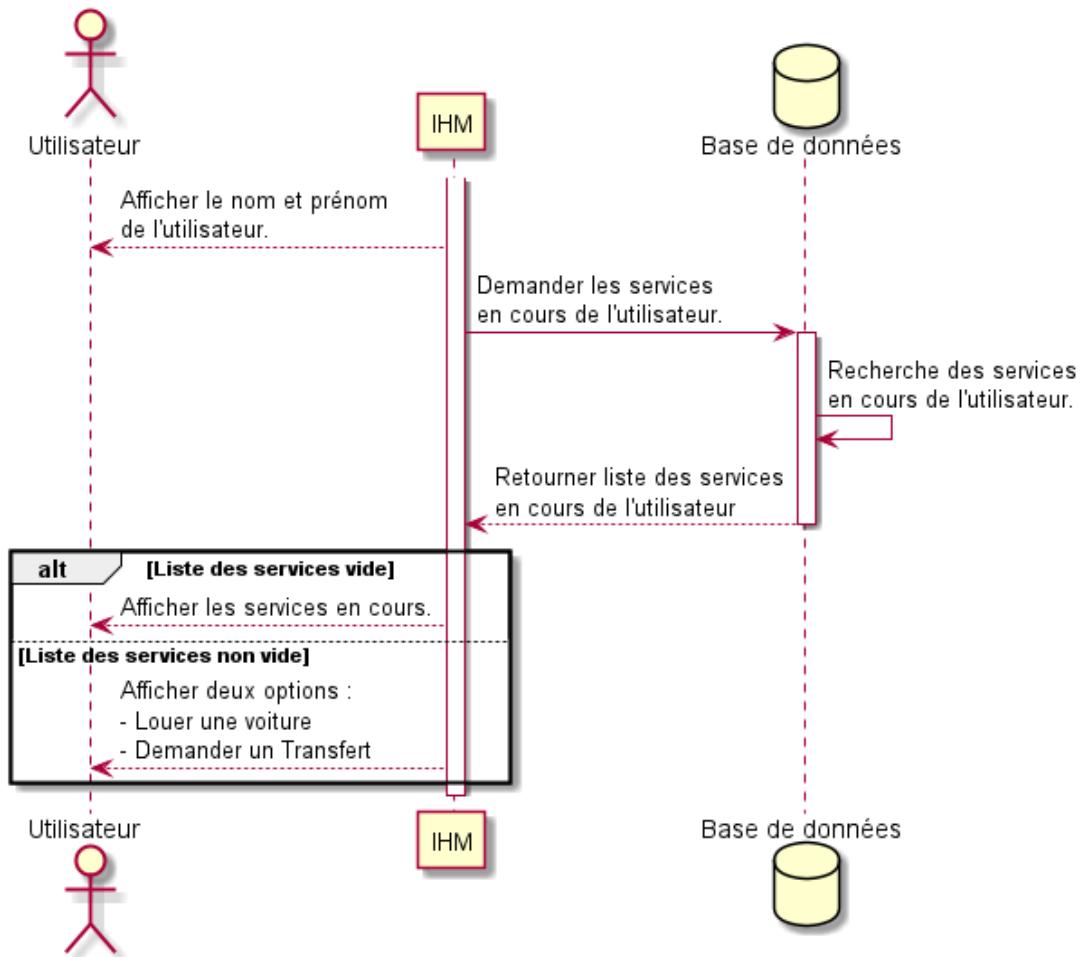


FIGURE 2.11 – Diagramme de séquences : Page d'accueil.

### Demander un service

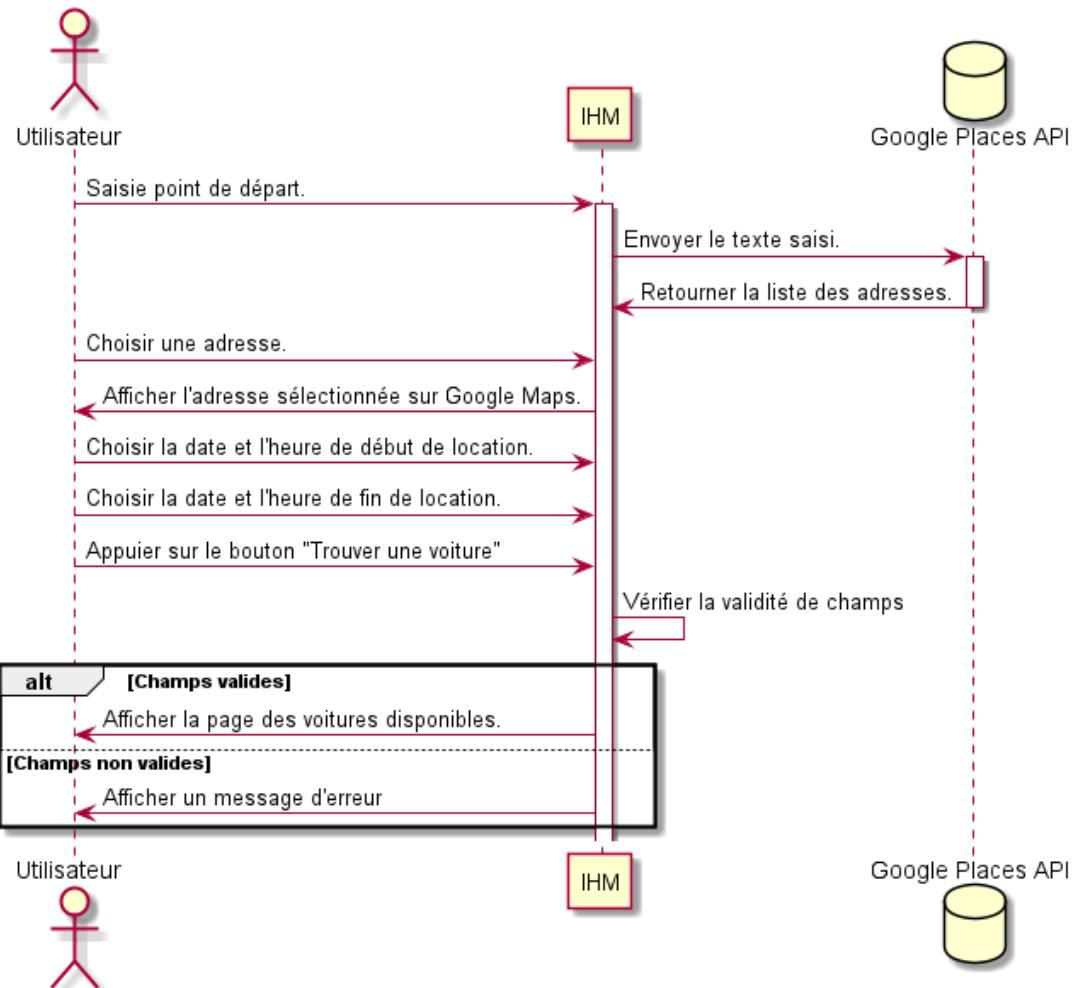


FIGURE 2.12 – Diagramme de séquences : Demander une location.

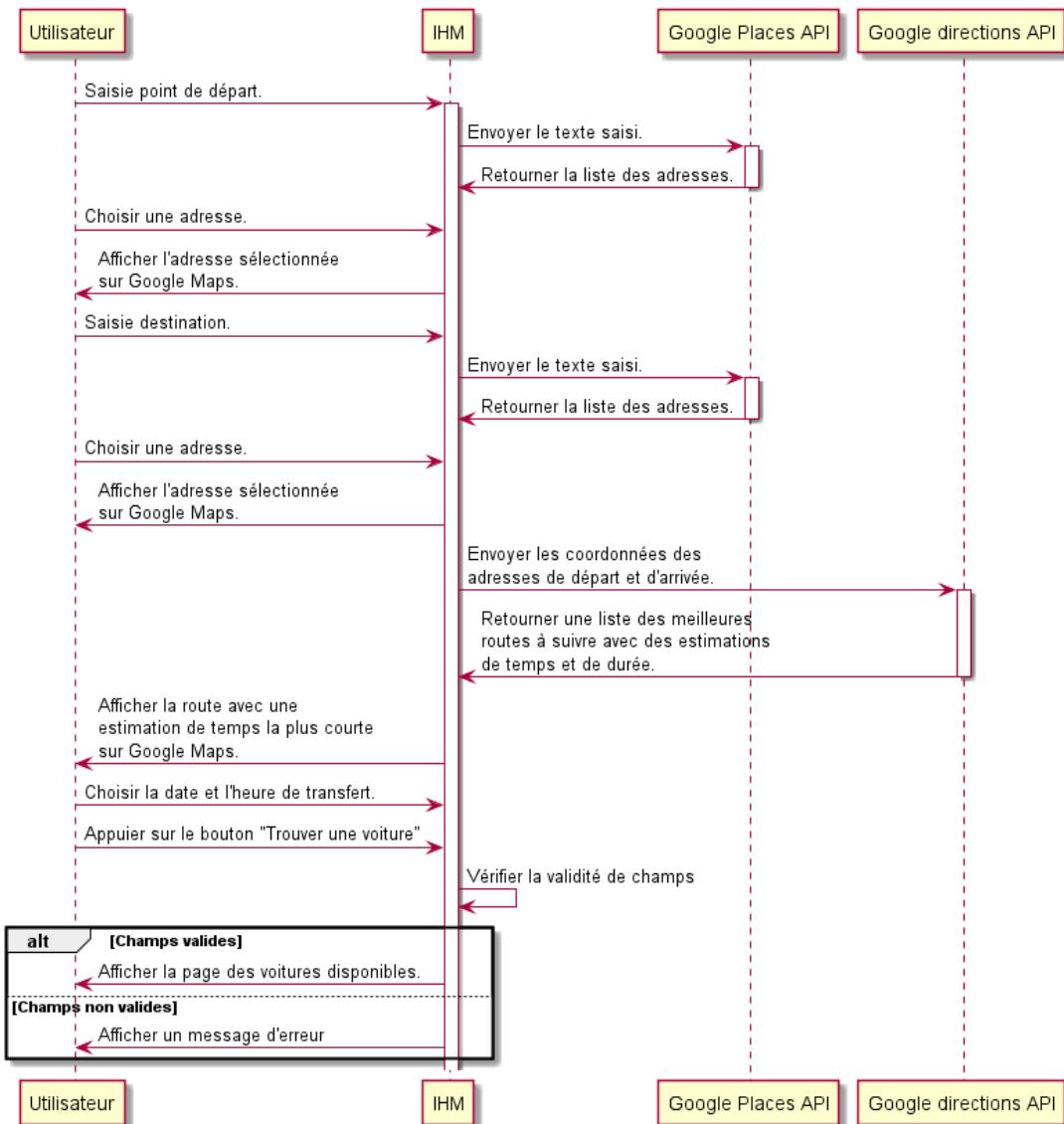


FIGURE 2.13 – Diagramme de séquences : Demander un transfert.

### Affichage des voitures disponibles

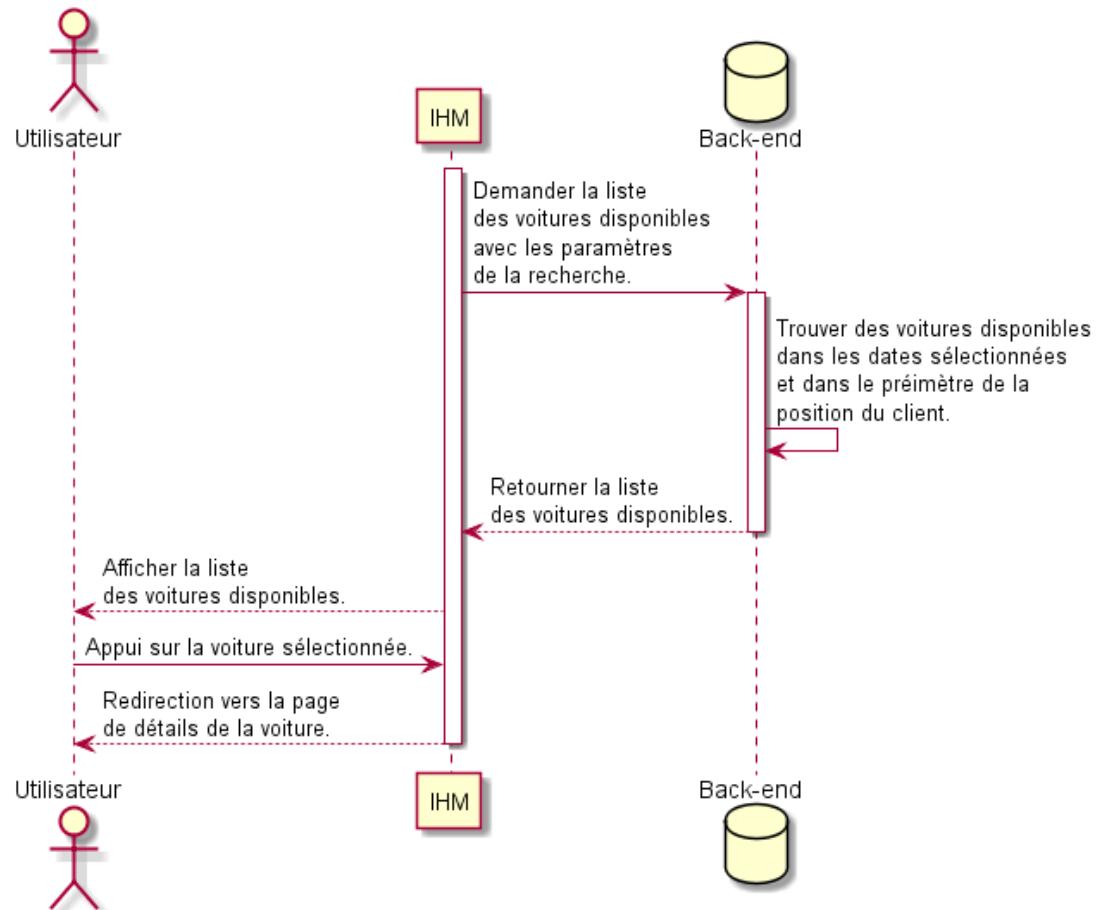


FIGURE 2.14 – Diagramme de séquences : Choisir une voiture

### Signature numérique de contrat de location

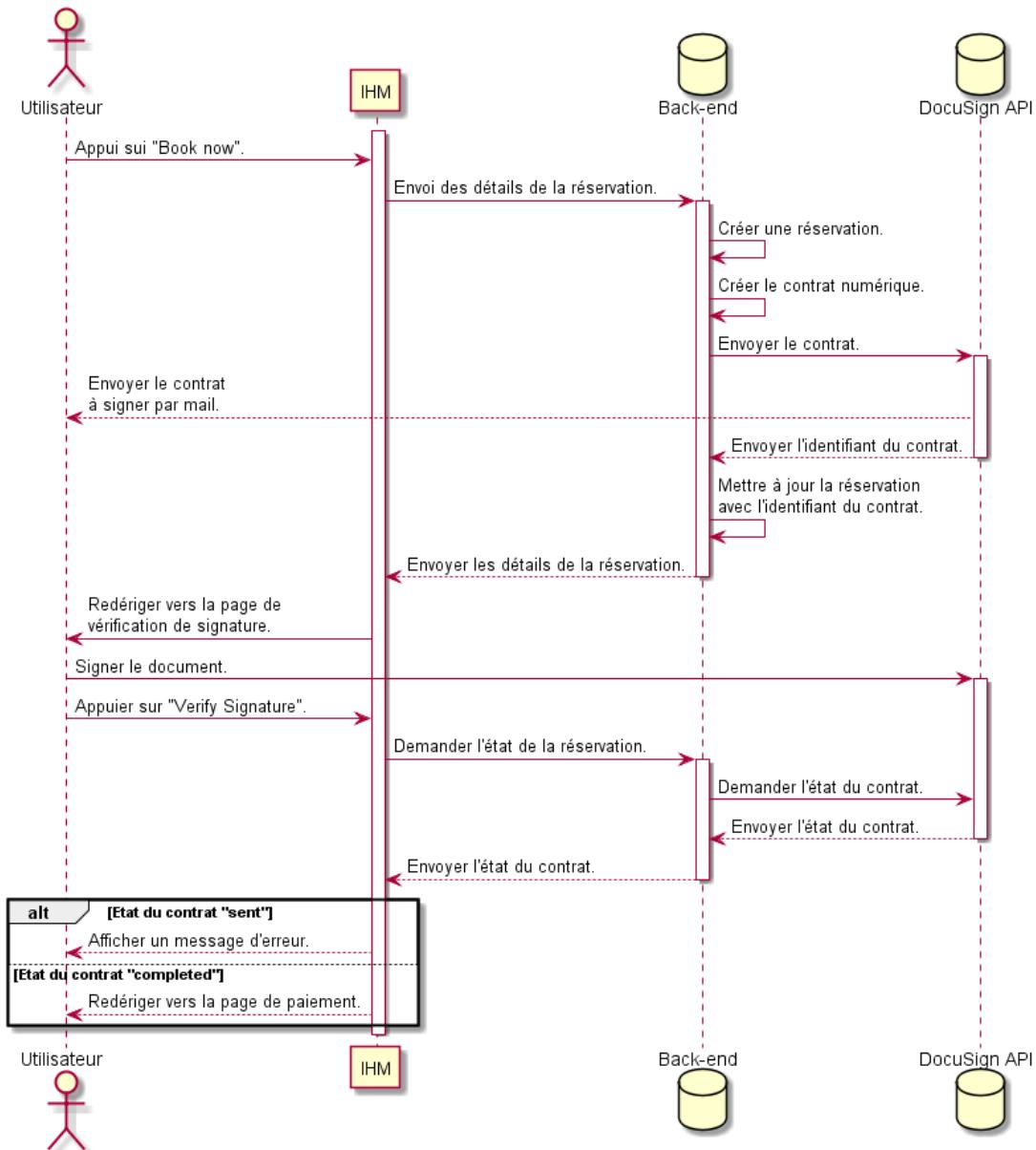


FIGURE 2.15 – Diagramme de séquences : Envoi du contrat de location.

### Paiement en ligne

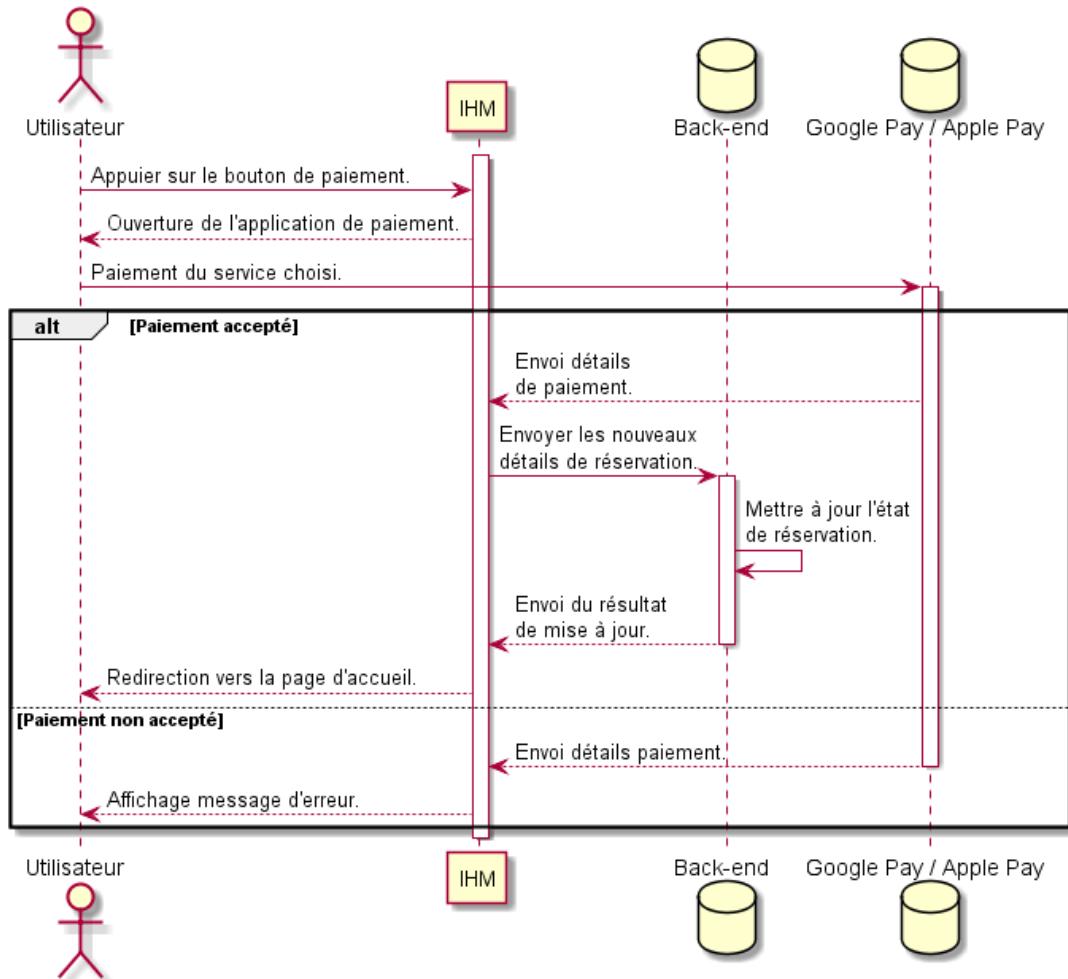


FIGURE 2.16 – Diagramme de séquences : Paiement des services.

### Localiser une voiture

## 2.4 UI/UX Design

Avant de passer au développement de l’application, il faut créer d’abord les prototypes des interfaces utilisateur.

Cette étape est nécessaire pour tester plusieurs approches dans les interfaces de l’application, s’assurer d’offrir une expérience optimale pour l’utilisateur.

Suite à une collaboration avec l’équipe de design UI/UX, les interfaces suivantes ont été créées.



Welcome to SPN  
luxury cars.

Fast service, highly skilled and  
professional chauffeurs.

[Get started](#)

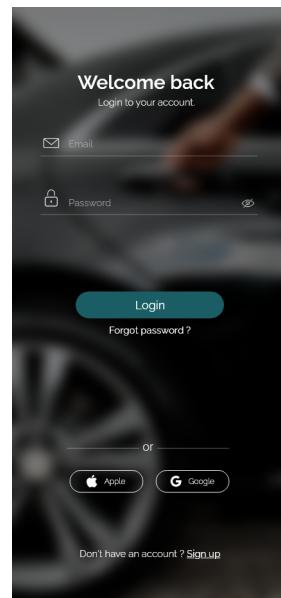


FIGURE 2.17 – Première page.

FIGURE 2.18 – Page de connexion.

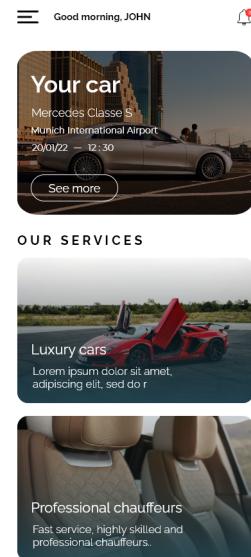
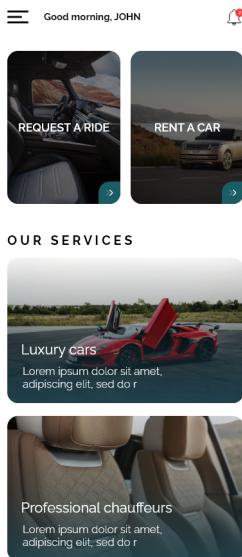
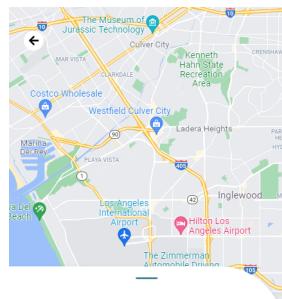


FIGURE 2.19 – Page d'accueil.

FIGURE 2.20 – Page d'accueil lors d'un transfert en cours.

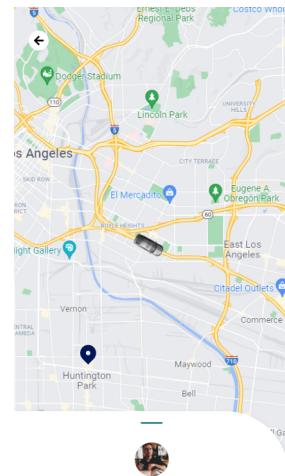


Pickup

Departure date Time

Requested duration

**Find car**



cancel the trip ?

FIGURE 2.21 – Sélection de type de transfert.

FIGURE 2.22 – Suivi de la position actuelle du chauffeur avec la voiture.



FIGURE 2.23 – Liste des voitures disponibles.



Mercedes Classe S

From 500€

fuel  
interior beige  
5 items max

#### GALLERY



**Book now**

FIGURE 2.24 – Détails de la voiture sélectionnée.

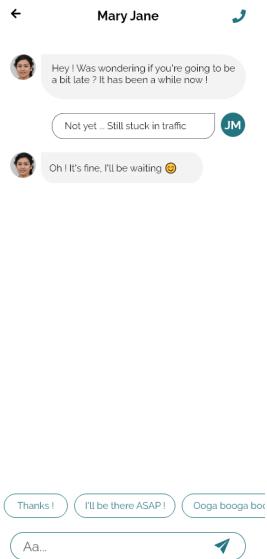


FIGURE 2.25 – Messagerie instantanée avec le chauffeur.

## 2.5 Conclusion

A travers ce chapitre, on a établi la conception de l'application SPN-Cars : On a dégager les besoins fonctionnels et non fonctionnels, qui, à travers eux on a pu créer les diagrammes de cas d'utilisation, les diagrammes de classes, et les diagrammes de séquences. Suite au développement de ces diagrammes, on est passé vers l'étape de conception des interfaces utilisateur. Toutes ces différentes étapes permettront de choisir les technologies et outils qui seront utilisés pour la création de cette application, qui seront présentés dans le prochain chapitre.

## *Chapitre 3*

---

### *Technologies et outils utilisés*

---

#### **Contenu**

---

<b>3.1</b>	<b>Introduction</b>	36
<b>3.2</b>	<b>Conception des interfaces utilisateur</b>	36
3.2.1	Adobe Xd	36
<b>3.3</b>	<b>Développement de l'application mobile</b>	37
3.3.1	Flutter	37
<b>3.4</b>	<b>Développement du serveur back-end</b>	38
3.4.1	Express JS	38
3.4.2	MongoDB	40
3.4.3	JSON Web Token (JWT)	41
3.4.4	Firebase	42
3.4.5	Swagger	43
<b>3.5</b>	<b>Outils et logiciels</b>	43
3.5.1	Visual Studio Code	43
3.5.2	Postman	45
3.5.3	Git	46
<b>3.6</b>	<b>Conclusion</b>	46

---

## 3.1 Introduction

Comme cette application sera une application mobile, il est nécessaire que la performance soit la priorité lors du développement. C'est pourquoi on a choisi les technologies suivantes pour offrir une application rapide, performante, facile à utiliser.

Ses différentes technologies sont classifiées dans trois domaines principalement : Conception des interfaces graphiques, développement de l'application, et développement du back-end de l'application.

## 3.2 Conception des interfaces utilisateur

### 3.2.1 Adobe Xd



FIGURE 3.1 – Logo Adobe Xd

**Adobe Xd** [6] est un outil de conception et modélisation des interfaces utilisateur des applications web et mobiles, développé par Adobe Inc.

Grâce aux outils fournis par Adobe Xd, la conception, l'amélioration et la rectification des interfaces graphiques et l'expérience de l'utilisateur de l'application sera plus facile, plus rapide et plus efficace. Dans le cadre de ce projet, Adobe Xd a été utilisé pour la création des prototypes des interfaces graphiques, qui seront, par la suite, construits en application mobile à l'aide de Flutter.

### 3.3 Développement de l'application mobile

#### 3.3.1 Flutter



FIGURE 3.2 – Logo Flutter

**Flutter** [7] est un kit de développement (SDK) open-source créé par Google et publié en 2017.

Flutter permet de créer des applications mobiles (Android / iOS), web et même desktop (Windows / Linux / MacOS), avec une seule base de code en Dart, un langage de programmation développé aussi par Google.

Flutter présente plusieurs avantages qui permettent de créer des applications mobiles performantes et réduit aussi le coût et le temps de développement nécessaires, grâce au langage de programmation utilisé **Dart** qui est très facile à maîtriser et qui offre plusieurs avantages, dont le plus important la fonctionnalité de «**Hot Reload**» qui permet de recharger l'application et afficher les changements sur l'écran sans passer par la recompilation du code source.

Pour créer des applications mobiles crossplatform, Flutter n'est pas la seule technologie qui facilite le développement de ces applications. Il existe plusieurs autres alternatives dont on cite React Native qui permet aussi de créer une application mobile iOS et Android avec une seule base de code.

	Flutter	React Native
Avantages	<ul style="list-style-type: none"> <li>— crossplatform.</li> <li>— Créé et maintenu par Google.</li> <li>— Applications fluides avec 60 FPS.</li> <li>— Compilé vers code natif.</li> </ul>	<ul style="list-style-type: none"> <li>— crossplatform.</li> <li>— Créé et maintenu par Facebook.</li> <li>— Utilise Javascript.</li> <li>— Composants UI Natives</li> </ul>
Inconvénients	<ul style="list-style-type: none"> <li>— Besoin d'ap-prendre Dart.</li> <li>— Documentation complexe.</li> </ul>	<ul style="list-style-type: none"> <li>— Limité à 30 FPS.</li> <li>— Application volumineuse.</li> </ul>

TABLE 3.1 – Tableau comparatif : Flutter et React Native

Comme le montre le tableau comparatif, Flutter présente plusieurs avantages en terme de fluidité et taille de l'application, ces deux avantages joueront un rôle très important avec l'expérience de l'utilisateur qui sera mieux avec une application Flutter qu'une application avec React Native.

## 3.4 Développement du serveur back-end

### 3.4.1 Express JS



FIGURE 3.3 – Logo Express

**Express JS** [8] est un framework backend gratuit et open-source pour NodeJS. Créeée par TJ Holowaychuk, la première version publique d'Express JS a été introduite au public en 2010.

Express JS est un framework minimaliste, très léger pour garantir une performance optimale et une exécution rapide. Ce framework est aussi très flexible, même s'il fournit que quelques fonctionnalités, grâce à **NPM**, le gestionnaire de paquets de NodeJS, il peut être complété par plusieurs librairies disponibles.

Grâce à son minimalisme et facilité d'implémentation, ExpressJS est utilisé par plusieurs par nombreuses sociétés dans le monde, pour développer tout type d'applications, parmi ses sociétés il y a des géants de technologies tels que **IBM**, **Uber**, et plusieurs autres.

Il existe plusieurs alternatives d'ExpressJS, l'un de ses alternatives est Spring Boot, le framework web de Java qui est aussi utilisé dans plusieurs projets.

	ExpressJS	Spring Boot
Avantages	<ul style="list-style-type: none"> <li>— Utilise Javascript.</li> <li>— Fonctionne sous NodeJS qui est très minimaliste.</li> </ul>	<ul style="list-style-type: none"> <li>— Utilise le Multithreading.</li> <li>— Utilise Java.</li> </ul>
Inconvénients	<ul style="list-style-type: none"> <li>— N'utilise pas le Multithreading.</li> <li>— Ne contient pas de vérification de type avec Javascript.</li> </ul>	<ul style="list-style-type: none"> <li>— Grande consommation de mémoire.</li> <li>— Un fichier de déploiement de grande taille.</li> </ul>

TABLE 3.2 – Tableau comparatif : ExpressJS et Spring Boot

Comme la rapidité est l'un des objectifs de cette application, le temps de réponse à partir du serveur doit être minimal, ce qui offre ExpressJS grâce à NodeJS qui est très léger et qui peut être amélioré grâce aux dépendances de NPM.

### 3.4.2 MongoDB



FIGURE 3.4 – Logo MongoDB

**MongoDB** [9], est un système de gestion de base de données NoSQL, orientée documents. Une base de données NoSQL est utilisée pour le stockage de volumes massifs de données, elle se distingue des bases de données relationnelles par sa flexibilité et ses performances. Le système MongoDB est développé par la société qui porte le même nom en 2007. Cette entreprise travaillait sur un système de cloud computing à données largement réparties. Il est depuis devenu l'un des systèmes de gestion de base de données les plus utilisés, notamment pour des sites web très populaires tels que : *SourceForge.net*, *eBay* et *The New York Times*. Contrairement à une base de données relationnelle SQL traditionnelle, MongoDB ne repose pas sur des tableaux et des colonnes. Les données sont stockées sous forme de collections et de documents. Les documents sont des paires de valeurs / clés servant d'unité de données de base. Les collections quant à elles contiennent des ensembles de documents et de fonctions. Elles sont l'équivalent des tableaux dans les bases de données relationnelles classiques.

	MongoDB	MySQL
Avantages	<ul style="list-style-type: none"> <li>— N'a pas besoin d'un modèle d'entité.</li> <li>— Sauvegarde ses documents sous forme de JSON.</li> </ul>	<ul style="list-style-type: none"> <li>— Plus utilisé.</li> <li>— Haute disponibilité.</li> </ul>
Inconvénients	<ul style="list-style-type: none"> <li>— Plus lent si on utilise des jointures.</li> <li>— Utilise son propre langage de requêtes.</li> </ul>	<ul style="list-style-type: none"> <li>— Risque de sécurité avec les injections SQL.</li> <li>— On ne peut pas changer les modèles.</li> </ul>

TABLE 3.3 – Tableau comparatif : MongoDB Code et MySQL

Vu que l'application SPN Cars n'utilise pas beaucoup de jointures dans sa conception, et l'utilisation des données non structurées sera plus bénéfique qu'utiliser des données structurées, le choix de MongoDB comme base de données est le choix le plus pertinent.

### 3.4.3 JSON Web Token (JWT)



FIGURE 3.5 – Logo JWT

**JSON Web Token** ou tout simplement connu comme "JWT" est un standard créé en 2015, qui permet d'encapsuler des données dans des jetons et les échanger entre plusieurs parties en toute sécurité.

Un jeton est composé principalement de trois parties :

- Un en-tête qui décrit le jeton en format JSON.
- Le contenu de ce jeton qui est également en format JSON.
- Une signature numérique.

### 3.4.4 Firebase



FIGURE 3.6 – Logo Firebase

**Firebase** [10] est une plateforme créée en 2011, puis acquise et développée par Google en 2014. Firebase facilite la création de back-end à la fois scalable et performant.

L'objectif de firebase est d'offrir aux professionnels et aux particuliers un moyen d'éviter l'engagement dans un processus complexe de création et de maintenance d'une architecture serveur. Firebase offre des API intuitives regroupées dans un SDK unique. Ces API permettent de gagner du temps et de réduire le nombre d'intégrations qu'on doit gérer dans l'application.

Firebase offre plusieurs services que tout le monde peut utiliser gratuitement grâce à sa politique «pay as you go» qui nécessite le paiement de ses services seulement si l'utilisation des ressources dépasse le quota du plan gratuit offert. Les services de Firebase les plus utilisés sont :

— **Firestore :**

Une base de données NoSQL, bénéficiant d'un hébergement cloud et permettant le stockage et la synchronisation de données des utilisateurs.

— **Firebase Authentification :**

Un SDK prêt et facile à exploiter qui permet d'authentifier les utilisateurs en offrant plusieurs méthodes d'authentification tels que Google, Apple, Facebook, Email et mot de passe, numéro de téléphone et plusieurs d'autres méthodes pour assurer l'authentification de l'utilisateur.

— **Firebase Cloud Messaging :**

Permet de connecter plusieurs périphériques au serveur dans les meilleures conditions (fiabilité et économie de batterie). Ce service permet de recevoir et envoyer des notifications sur les différentes plateformes (Web / iOS / Android). Avec Firebase Cloud Messaging, il est possible aussi d'assurer un service de messagerie instantanée entre les utilisateurs.

### 3.4.5 Swagger



FIGURE 3.7 – Logo Swagger

**Swagger** [11] est un langage permettant de créer une description des API Restful à l'aide de JSON. A l'aide de ses outils Open-Source, Swagger permet de concevoir, créer et décrire des API REST.

Swagger a été créé en 2011 par Tony Tam, cofondateur du site de dictionnaires Wordnik, suite à un besoin d'automatisation de documentation de l'API qui est devenue de plus en plus frustrante. Juste après sa création, le projet Swagger est devenu open-source en septembre 2011. Swagger est maintenant maintenu par la société **SmartBear Software** qui, en novembre 2015, a créé une organisation appelée **OpenAPI initiative** dont diverses entreprises tels que Google, IBM et Microsoft sont les membres fondateurs.

## 3.5 Outils et logiciels

### 3.5.1 Visual Studio Code

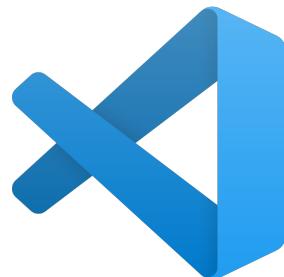


FIGURE 3.8 – Logo Visual Studio Code

**Visual Studio Code** [12] est un éditeur de texte open-source développé par Microsoft [13] pour Windows Mac OS et Linux en 2015. Riche en fonctionnalités tels que le support de débogage, complétion intelligente du code, intégration de système de contrôle de version Git, et la

capabilité d'installation d'extensions qui permettent d'améliorer l'expérience de l'utilisateur, Visual Studio Code est devenu l'un des premier choix pour plusieurs développeurs pour travailler sur leurs différents projets.

Pour développer des application mobiles avec Flutter, on peut utiliser Visual Studio Code ou Android Studio. Ces deux éditeurs offrent chacun des avantages et des inconvénients qu'on va les présenter dans le tableau suivant.

	Visual Studio Code	Android Studio
Avantages	<ul style="list-style-type: none"> <li>— Démarrage rapide.</li> <li>— Intégration du terminal.</li> <li>— Une sélection de extensions ajoutés par les utilisateurs qui améliorent l'expérience utilisateurs.</li> </ul>	<ul style="list-style-type: none"> <li>— Puissant et robuste.</li> <li>— Editeur officiel pour les applications Android.</li> <li>— Intégration des émulateurs Android.</li> </ul>
Inconvénients	<ul style="list-style-type: none"> <li>— Il faut installer des extensions pour améliorer l'expérience.</li> <li>— Lenteur avec plusieurs fichiers ouverts à la fois.</li> </ul>	<ul style="list-style-type: none"> <li>— Consommation énorme de ressources.</li> <li>— Complexe pour les débutants.</li> </ul>

TABLE 3.4 – Tableau comparatif : Visual Studio Code et Android Studio

Suite à cette comparaison, on a choisi Visual Studio Code grâce aux avantages qu'il offre tels que : La rapidité et fluidité d'usage est les extensions qui facilitent le développement des applications. Ces avantages permettront d'accélération du cycle de développement.

### 3.5.2 Postman



FIGURE 3.9 – Logo Postman.

**Postman** [14] est une plateforme permettant de concevoir, développer, et tester des API. Créé en 2012 comme une extension pour Google Chrome, Postman a gagné une grande popularité, qui lui a permis de migrer vers une application pour Windows Mac OS, et Linux. Postman est maintenant le premier choix pour la plupart des développeurs, avec plus de 20 millions d'utilisateurs [15].

	Postman	Insomnia REST Client
Avantages	<ul style="list-style-type: none"> <li>— Simple à utiliser.</li> <li>— Sauvegarde de l'historique des requêtes.</li> <li>— Travail collaboratif.</li> </ul>	<ul style="list-style-type: none"> <li>— Interface utilisateur simple.</li> <li>— Open Source.</li> <li>— Cross platform : Disponible sous Windows Mac OS et Linux</li> </ul>
Inconvénients	<ul style="list-style-type: none"> <li>— Gratuit pour les équipes moins de 5 membres.</li> <li>— Plusieurs fonctionnalités qui sont parfois ambiguës.</li> </ul>	<ul style="list-style-type: none"> <li>— Pas de collaboration en équipe.</li> </ul>

TABLE 3.5 – Tableau comparatif : Postman et Insomnia REST Client

### 3.5.3 Git



FIGURE 3.10 – Logo Git.

**Git** [16] est un système de contrôle de version open-source créé en 2005 par **Linus Torvalds**, le créateur du noyau du système d’exploitation Linux.

Git permet de gérer les ajouts et changements apportés au code source de manière tracée. Ainsi, si une erreur est commise, les développeurs peuvent revenir en arrière et comparer les versions antérieures du code, ce qui leur permet de corriger l’erreur tout en minimisant les perturbations pour tous les membres de l’équipe.

## 3.6 Conclusion

Suite à l’étape de conception vue dans le chapitre précédent, on a présenté les outils et technologies qui ont été choisis pour le développement de ce projet.

Chaque outil et technologie a été choisi suite à une étude du besoin et une comparaison avec les différents compétiteurs disponibles.

Une fois ces technologies choisies, on passera vers la dernière étape qui est la réalisation de cette application qui sera présentée dans le chapitre suivant.

## *Chapitre 4*

---

### *Réalisation de l'application*

---

#### **Contenu**

---

<b>4.1</b>	<b>Introduction</b>	48
<b>4.2</b>	<b>Création de compte</b>	48
4.2.1	Création de compte avec email et mot de passe	48
4.2.2	Création de compte avec Google ou Apple	49
<b>4.3</b>	<b>Authentification</b>	49
<b>4.4</b>	<b>Page d'accueil</b>	51
<b>4.5</b>	<b>Gestion de profil</b>	52
<b>4.6</b>	<b>Demander un service</b>	52
<b>4.7</b>	<b>Affichage des voitures disponibles</b>	54
<b>4.8</b>	<b>Signature numérique de contrat de location</b>	54
<b>4.9</b>	<b>Paiement</b>	57
<b>4.10</b>	<b>Localiser une voiture</b>	57
<b>4.11</b>	<b>Messagerie instantanée</b>	58
<b>4.12</b>	<b>Documentation des API</b>	59
<b>4.13</b>	<b>Conclusion</b>	63

---

## **4.1 Introduction**

Dans ce chapitre, on décrira toutes les fonctionnalités de l'application, leurs fonctionnement, les différents scénarios d'exécution avec des captures d'écran de chaque interface d'utilisateur avec un diagramme de séquences qui décrit en détail la fonctionnalité mentionnée

## **4.2 Crédation de compte**

La première étape pour utiliser l'application SPN-Cars est de créer un compte. Ce compte permettra aux utilisateurs de bénéficier de tous les services offerts par l'application.

Pour créer un compte, l'utilisateur a la possibilité de choisir trois méthodes : Créer un compte avec son email et choisir un mot de passe, ou créer un compte tout simplement en utilisant l'option de création de compte avec son compte Google ou Apple.

### **4.2.1 Crédation de compte avec email et mot de passe**

C'est la méthode la plus basique qui existe depuis toujours. Pour créer un compte, l'utilisateur doit tout d'abord entrer son adresse email, un mot de passe, et une confirmation de ce mot de passe. Lors de l'appui sur le bouton de «Créer un compte», l'application envoie une requête vers le serveur back-end afin de vérifier l'existence d'un compte d'utilisateur utilisant la même adresse e-mail. Après une recherche effectuée sur les utilisateurs, le serveur back-end envoie une réponse positive s'il n'a trouvé aucun compte d'utilisateur utilisant l'adresse e-mail entrée par l'utilisateur, sinon une réponse négative sera envoyée. Selon la réponse retournée par le serveur l'application redirigera l'utilisateur vers une page pour compléter l'étape de création de compte si la réponse du serveur est positive, ou affiche un message d'erreur avec l'erreur adéquate si la réponse du serveur est négative.

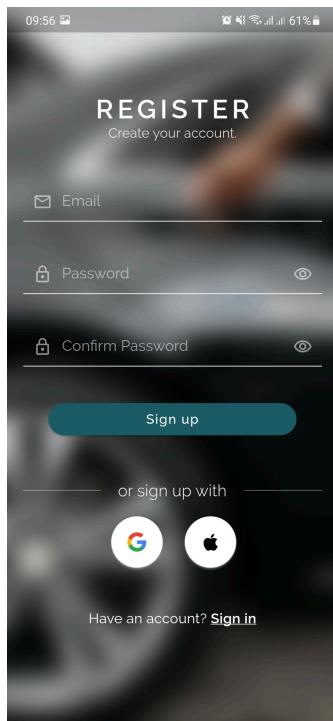


FIGURE 4.1 – Formulaire de création de compte : 1ère étape.

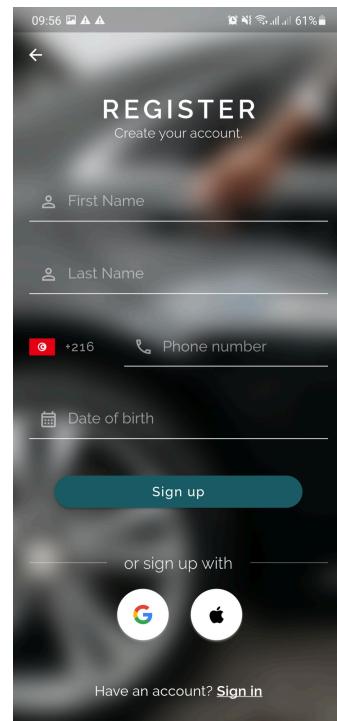


FIGURE 4.2 – Formulaire de création de compte : 2ème étape.

## 4.2.2 Crédation de compte avec Google ou Apple

Cette méthode est la plus facile et la plus rapide pour créer un compte ou s'authentifier. Avec un simple appui sur le bouton adéquat, une requête envoyée aux services de Google ou Apple pour récupérer les données du compte choisi. Ses données sont :

- Un identifiant unique du compte Google ou Apple.
- L'adresse email du compte.
- Le nom et prénom utilisés avec le compte.
- La photo de profil utilisée avec ce compte.

Ses informations seront nécessaires pour passer la première étape de création de compte et avec eux l'utilisateur gagnera un peu de temps lors de la création de son compte.

## 4.3 Authentication

L'authentification est la première étape dans le cycle de vie de l'application, lors du premier démarrage de l'application il est nécessaire de vérifier si l'utilisateur est déjà connecté à l'application. Grâce à cette étape on peut identifier l'utilisateur, et limiter les requêtes envoyées au serveur back-end.

Pour s'authentifier l'utilisateur peut choisir trois méthodes différentes : Email et mot de passe, avec son compte Google, ou avec son compte Apple.

La validation des champs est une étape nécessaire pour s'assurer que l'utilisateur n'envoie que des valeurs correctes vers les API de connexion, ce qui permet d'éviter les erreurs inattendues.

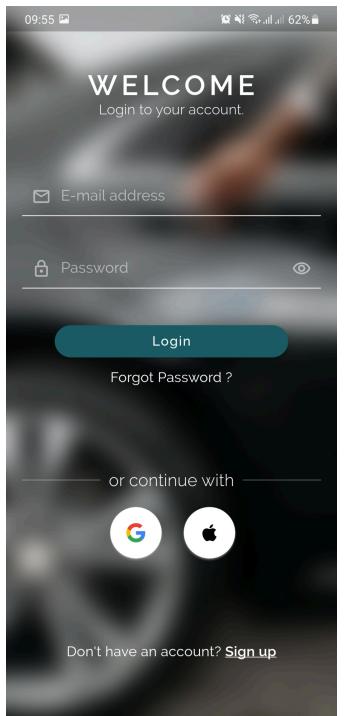


FIGURE 4.3 – Page de Login.

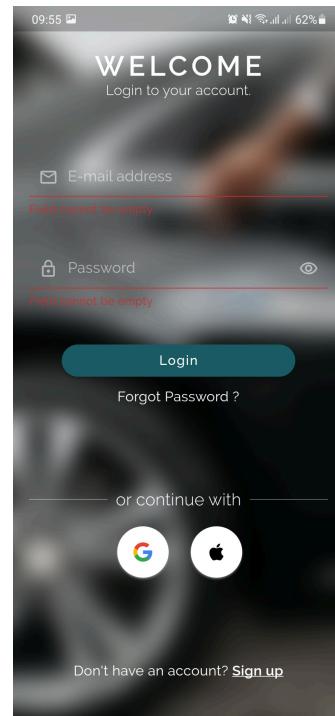


FIGURE 4.4 – Validation des champs de Login.

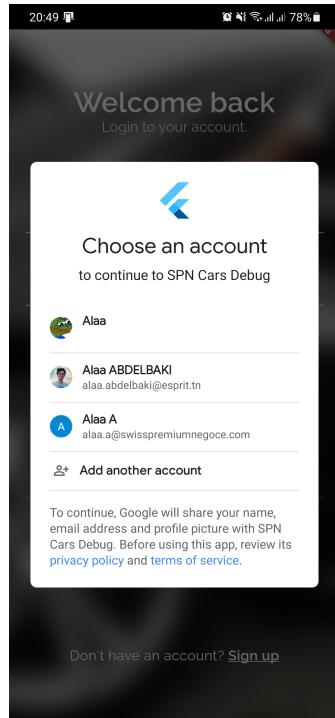


FIGURE 4.5 – Login avec Google.

## 4.4 Page d'accueil

Une fois l'utilisateur est réussi à s'authentifier, la page d'accueil s'affiche. Cette page diffère d'un utilisateur à un autre selon les services demandés par l'utilisateur : Si l'utilisateur a un service actif le moment de sa connexion, la liste de voitures louées avec les détails de chaque service actif demandé, et s'il n'a pas de services actif le moment de sa connexion, deux boutons seront affichés : «Rent a car» pour louer une voiture sans chauffeur et «Request a transfer» pour demander un chauffeur avec une voiture.

## 4.5 Gestion de profil

Chaque utilisateur possède un profil personnel, ce profil affichera les informations nécessaires pour faciliter la communication entre utilisateurs dans le futur (Ex : Contact entre chauffeur et client).

La page de profil, dans l'application SPN-Cars, est une page très simple qui contient les informations de base de l'utilisateur :

- La photo de profil.
- Le nom.
- Le prénom
- L'adresse E-mail.
- Le numéro de téléphone.
- La date de naissance.

De toutes ses données seuls le nom, prénom, photo de profil et numéro de téléphone seront accessibles aux chauffeurs pour assurer une communication avec les clients.

L'utilisateur peut modifier tous ses informations personnelles tout simplement en modifiant les champs contenant l'information à changer, et pour la photo de profile il suffit d'appuyer sur la photo pour la remplacer par une autre, soit prendre une nouvelle photo en utilisant l'appareil photo du smartphone, ou en choisissant une photo existante depuis la galerie. Une fois l'image est choisie, l'utilisateur sera redirigé vers une page pour recadrer l'image et choisir la zone qui sera affichée dans la page de profile, une fois l'image est recadrée, elle sera compressée pour réduire sa taille et faciliter son transfert vers le serveur distant.

## 4.6 Demander un service

Pour louer une voiture, l'utilisateur a besoin de spécifier tout d'abord les paramètres suivants :

- Le type de service demandé (Location / Transfert / Excursion / Long Ride).
- L'adresse de départ.
- L'heure de départ.
- L'adresse d'arrivée (Pas toujours disponible selon le type de service).
- L'heure d'arrivée (Pas toujours disponible selon le type de service).
- La durée du service demandé (Pas toujours disponible selon le type de service).

#### 4.6. DEMANDER UN SERVICE

53

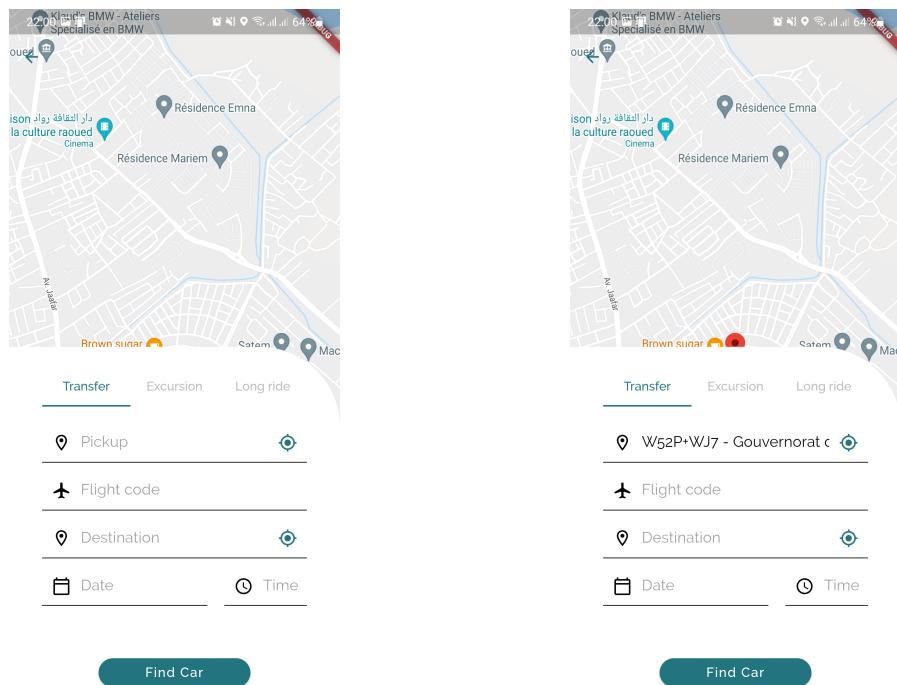


FIGURE 4.6 – Formulaire de demande de transfert.

FIGURE 4.7 – Utiliser le bouton «Localiser» pour choisir la position actuelle.

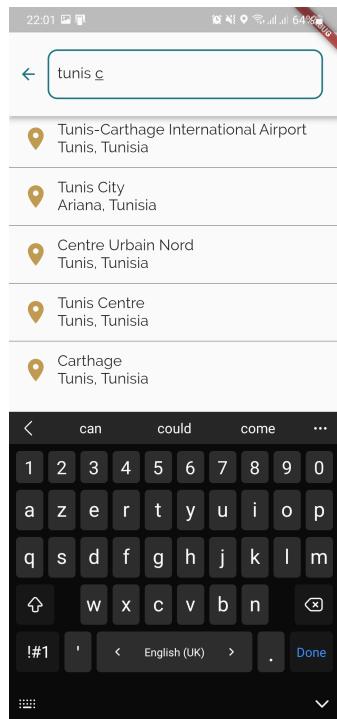


FIGURE 4.8 – Rechercher un emplacement à l'aide de Google Places.

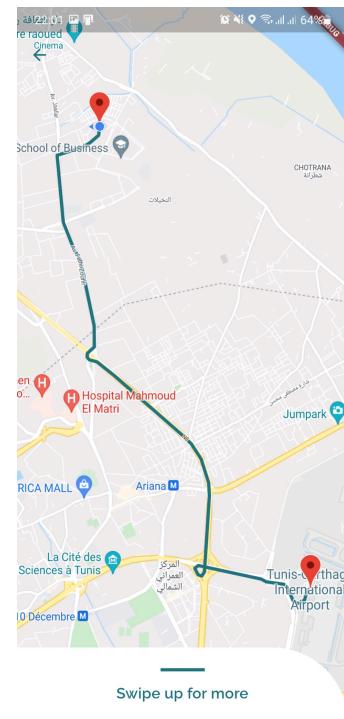


FIGURE 4.9 – Afficher la meilleure route entre le point de départ et le point d'arrivée.

## 4.7 Affichage des voitures disponibles

Après sélection des informations nécessaires par l'utilisateur, une recherche des voitures qui répondent aux critères de recherche choisis. Une fois qu'une liste de voitures est prête, les voitures seront affichées. L'utilisateur peut appuyer sur une voiture pour découvrir ses caractéristiques et choisir ensuite de la louer ou continuer sa recherche.

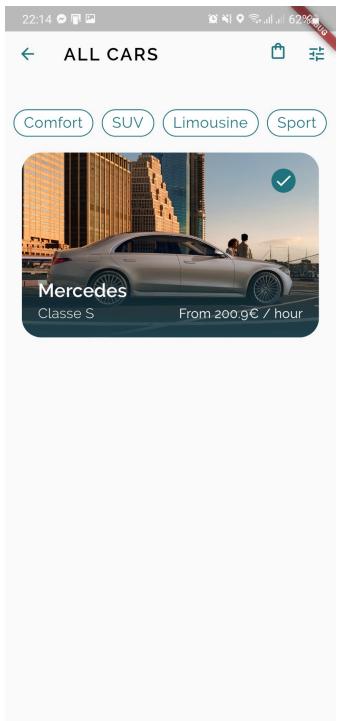


FIGURE 4.10 – Liste des voitures disponibles selon le point de départ sélectionné.

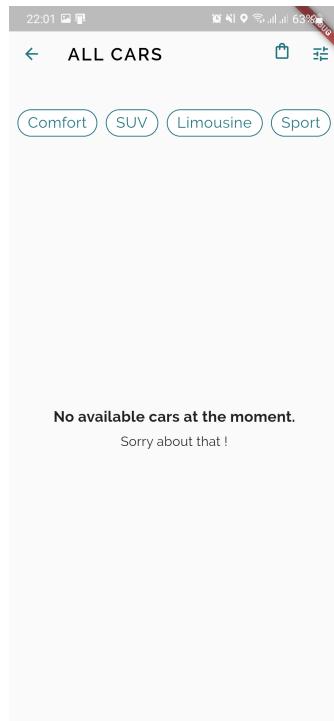


FIGURE 4.11 – Affichage si aucune voiture n'est disponible.

## 4.8 Signature numérique de contrat de location

Pour le cas de location de voitures, un contrat doit être signé entre le client et la société. Pour s'assurer de la signature du contrat, on a opté pour la solution de signature numérique à l'aide du service offert par DocuSign, qui se spécialise dans le domaine de la signature électronique et la gestion des transactions numériques pour faciliter les échanges et les validations électroniques des contrats et de documents.

Pour créer une réservation de location de voitures, il suffit d'appuyer sur le bouton de «Book now» dans la page de détails de la voiture choisie, et une réservation sera créée avec tous les détails : La position GPS, la date de récupération et retour de voiture et les informations de contact du client. Une fois ses informations sont enregistrées, un email contenant le contrat sera envoyé au client pour le signer, et l'identifiant de ce contrat sera enregistré avec les informations

de la réservation.

Dans l'application, le client est redirigé vers la page de vérification de signature où il doit appuyer sur le bouton de vérification de l'état de contrat. Pour signer ce contrat, il suffit d'ouvrir le document envoyé à l'adresse email du client, le signer, et retourner vers l'application. Lors de l'appui sur le bouton «Verify Signature», une requête sera envoyée au back-end qui contactera les APIs de «DocuSign» pour vérifier l'état de signature du contrat à l'aide de l'identifiant du contrat fourni lors de la création de la réservation. Si l'état de contrat est «sent», le contrat n'est pas encore signé, et un message d'erreur sera affiché à l'utilisateur. Sinon si l'état du contrat est «completed», les contrat est, donc, signé et l'utilisateur est redirigé vers la page de paiement.

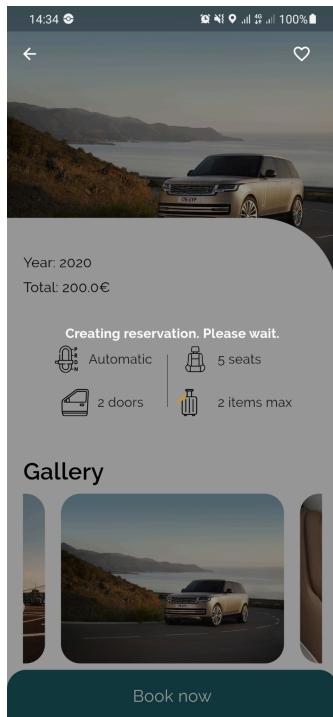


FIGURE 4.12 – Crédit de réservation.

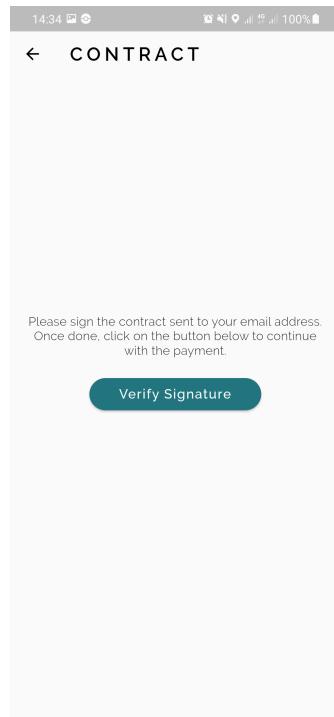


FIGURE 4.13 – Page de vérification de signature

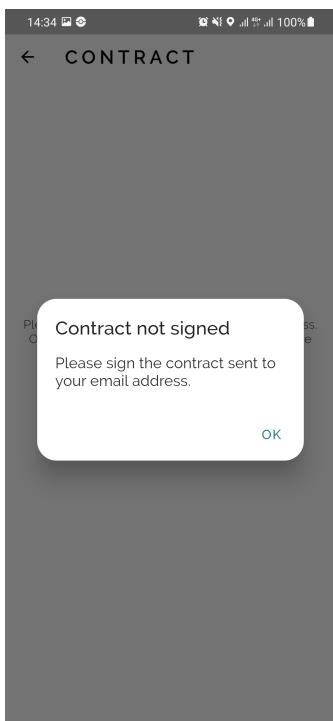


FIGURE 4.14 – Vérification de signature avec un document non signé.

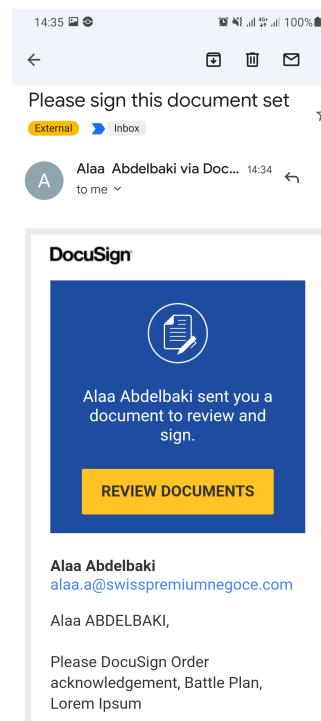


FIGURE 4.15 – Email contenant les documents à signer.



FIGURE 4.16 – Choix de signature.

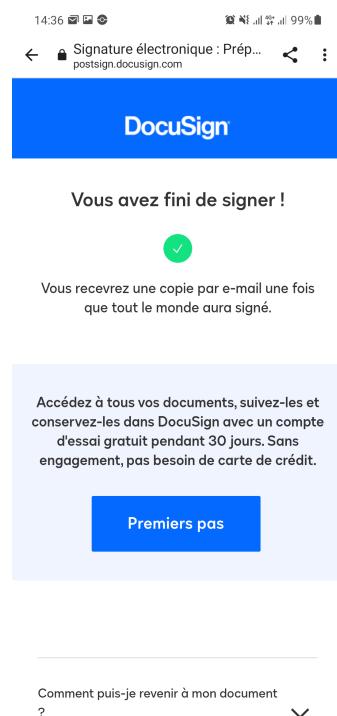


FIGURE 4.17 – confirmation de signature.

## 4.9 Paiement

Pour le paiement des services offerts par SPN Cars, l'utilisateur utilisera «Apple Pay» s'il utilise un iPhone ou «Google Pay» s'il utilise un smartphone Android, qui sont, grâce à leurs haute disponibilités, le choix optimal pour effectuer des paiements en ligne avec son propre smartphone.

Pour effectuer un paiement avec l'application SPN Cars, il suffit de signer le contrat de location dans le cas de location de voitures, ou sélectionner une voiture dans le cas de demande de transfert, après la page de paiement avec le bouton de paiement selon le type de smartphone sera affiché. Lors de l'appui, L'interface des applications de paiement adéquates sera affichée pour continuer la transaction. Une fois terminé, l'utilisateur sera redirigé vers la page d'accueil si le paiement est effectué, ou un message d'erreur sera affiché si non.

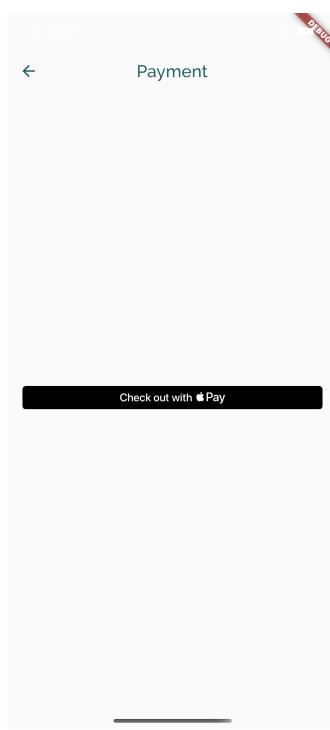


FIGURE 4.18 – Interface de paiement.

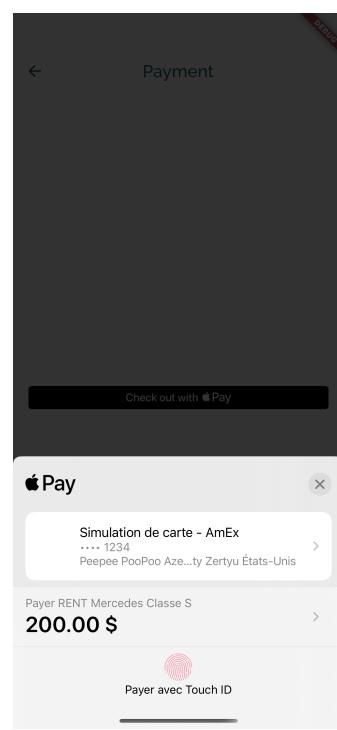


FIGURE 4.19 – Détails de paiement.

## 4.10 Localiser une voiture

Lorsqu'une réservation atteint la date de début et sera affichée dans la page d'accueil, le client peut appuyer sur la réservation et voir plus de détails sur sa réservation et sa voiture. Une fois appuyé sur le bouton «See more», le client sera redirigé vers une page qui affiche la position de sa voiture sur un plan. Cette position sera mise à jour en temps réel si le chauffeur

est en train de conduire la voiture.

La page permet aussi d'afficher les informations basiques du chauffeur avec une possibilité de le contacter soit par appel téléphonique soit par message.



FIGURE 4.20 – Affichage des détails de la réservation.

## 4.11 Messagerie instantanée

Une fois que l'utilisateur a appuyé sur le bouton pour contacter le chauffeur par messagerie instantanée, l'utilisateur est redirigé vers une page de messagerie avec le chauffeur où ils peuvent échanger des messages.

L'utilisateur peut aussi retourner vers sa liste de conversations et afficher ses anciens messages depuis la page d'accueil.

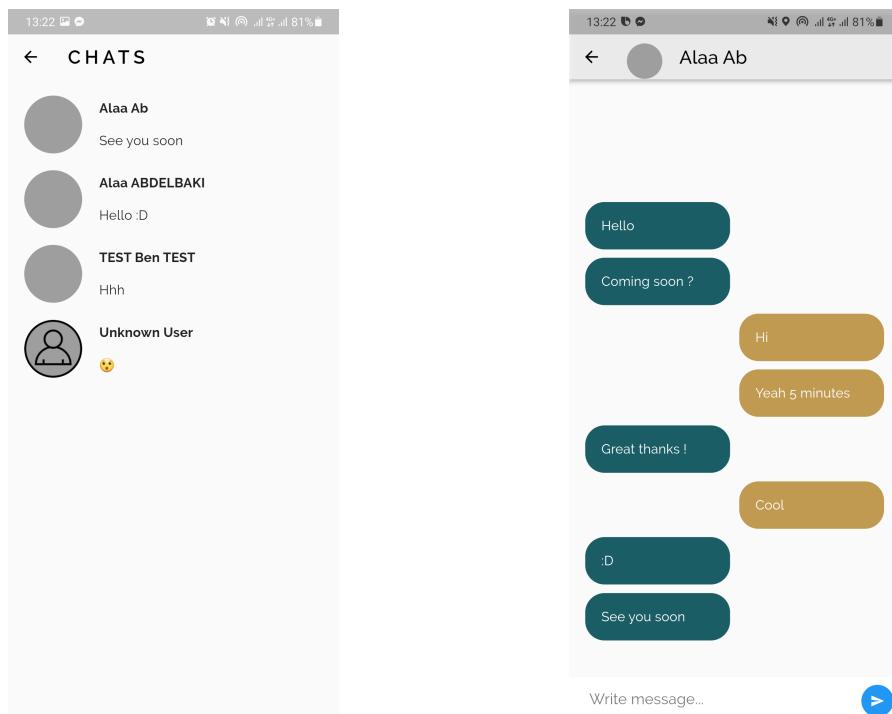


FIGURE 4.21 – Liste de messages de l’utilisateur.

FIGURE 4.22 – Conversation avec l’utilisateur.

## 4.12 Documentation des API

La documentation des APIs est une étape nécessaire dans le développement de n’importe quelle application. Une documentation accessible et facilement exploitable est une condition préalable au développement des API. Il est important d’avoir une documentation toujours à jour quand le code/les fonctionnalités de l’API évoluent.

Et puisque le modèle de développement chez SPN est centré sur le travail collaboratif, il est très important d’avoir une documentation claire et facile à exploiter des APIs.

On va prendre un exemple de documentation de l’API de l’entité «Voiture» où on a suivi les spécifications de Open Api pour créer la documentation de ses APIs.

```
{
  "openapi": "3.0.0",
  "security": [
    // Les normes de sécurité à suivre pour accéder à cet API.
    // Exemple: Bearer Token
    {
      //BasicAuth est défini dans "components"
      "BasicAuth:[]"
    }
  ],
  "info": {
    // Informations générales sur l'API
    "title": "SPN-Backend Cars",
    "version": "1.0.0",
    "description": "The documentation for using the SPN-Cars backend."
  },
  //La liste des serveurs pour accéder l'API
  "servers": [
    {
      "url": ""
    }
  ],
  //La liste des chemins d'accès pour accéder les différentes méthodes
  "paths": {
    //Exemple : Ajouter une voiture dans la base de données.
    "/api/cars/add": {
      "summary": "Add Car",
      // Type de requête : POST dans cet exemple
      "post": {
        "summary": "Add car.",
        // Description de fonctionnement de la requête
        "description": "# Adds a new car to the database.",
        "requestBody": {
          "required": true,
          "content": {
            "application/json": {
              "schema": {
                "$ref": "#/components/schemas/Car"
              }
            }
          }
        },
        // Différentes réponses du serveur
        "responses": {
          "200": {
            "description": "Car is added successfully"
          },
          "400": {
            "description": "Validation error"
          }
        }
      }
    }
  },
  "components": {
    "securitySchemes": {
      "BasicAuth": {
        "type": "http",
        "scheme": "bearer",
        "bearerFormat": "JWT"
      }
    },
    "schemas": {
      "Car": {
        //Définition de l'entité "Car"
        "type": "object",
        "properties": {
          "title": {
            "type": "String"
          },
          // ...
        }
      }
    }
  }
}
```

FIGURE 4.23 – Schéma de documentation des API.

## SPN-Backend Cars 1.0.0 OAS3

The documentation for using the SPN-Cars backend

Servers  Authorize

**default** ^

**POST** /api/cars/add Add car. ✓

**DELETE** /api/cars/delete Delete car. ✓

**POST** /api/cars/find Find car. ✓

**PUT** /api/cars/update Updates car ✓

**Schemas** ^

Car >

ExtraServices >

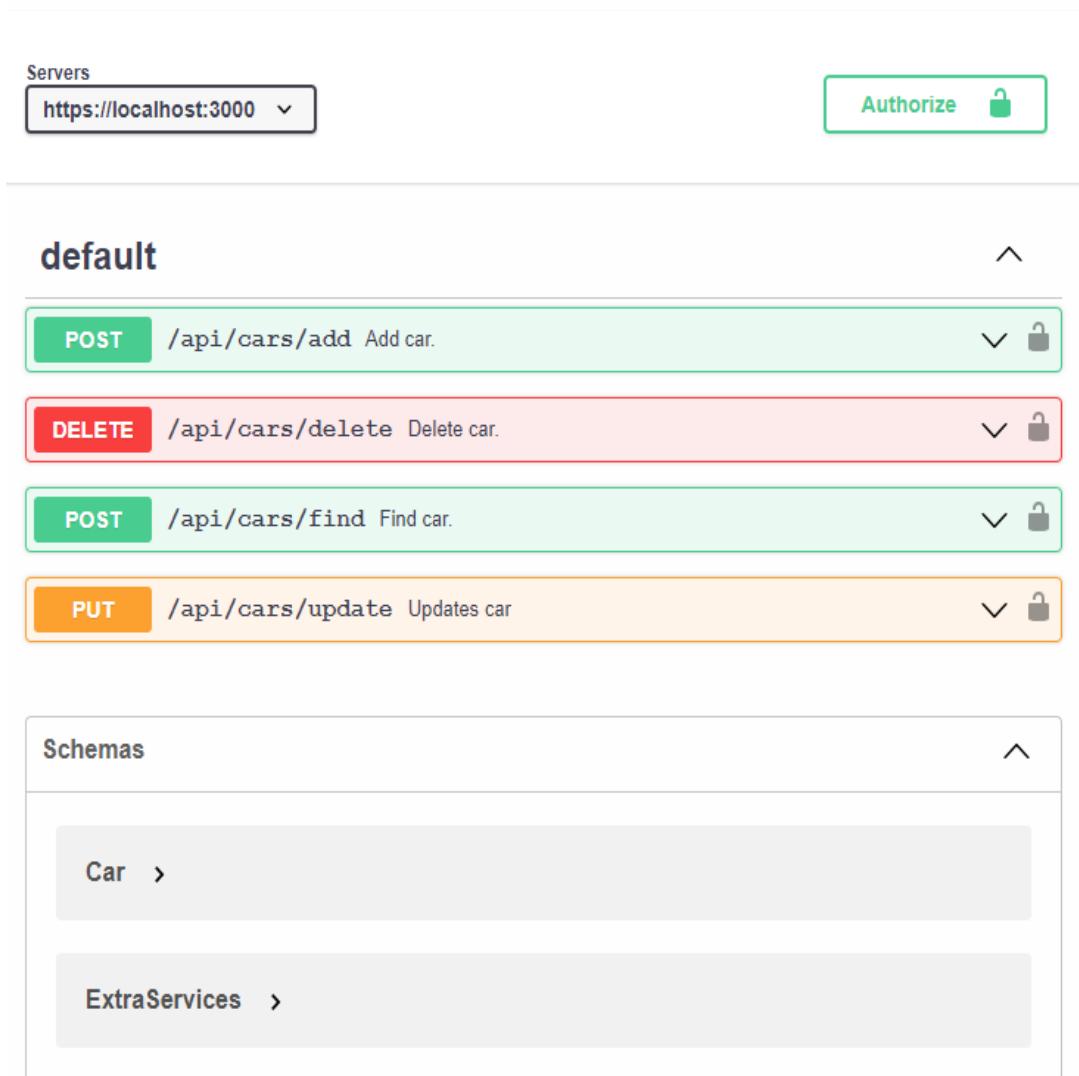


FIGURE 4.24 – Documentation des API.

**POST** /api/cars/add Add car.

Adds a new car to the database.

**Parameters**

No parameters

**Request body** required

application/json

**Example Value** | **Schema**

```
{
  "title": "string",
  "description": "string",
  "commission": 0,
  "enabled": true,
  "registration": "string",
  "brand": "string",
  "model": "string",
  "year": 0,
  "nbSeats": 0,
  "nbLuggage": 0,
  "price": 0,
  "status": "PARKED",
  "nbDoors": 0,
  "manual": true,
  "energy": "DIESEL",
  "image": "string",
  "extraServices": [
    {
      "title": "string",
      "price": 0
    }
  ]
}
```

**Responses**

Code	Description	Links
200	Car is added successfully	No links
400	Validation error	No links

FIGURE 4.25 – Détails API ajout voiture.

## 4.13 Conclusion

Dans ce chapitre on a présenté la réalisation de l'application, où on a décrit chaque fonctionnalité de l'application avec des captures d'écran qui montrent les différentes interfaces. On a commencé par l'étape de création de compte et connexion qui permettra au client d'utiliser les différents services offerts par l'application et on a passé par les différentes fonctionnalités une par une.

On a présenté aussi l'étape de documentation des APIs qui permettra de faciliter la collaboration avec d'autres équipes et l'amélioration de leur fonctionnement dans le futur.



---

## ***Conclusion et perspectives***

---

**D**ans le présent rapport, on a dressé le bilan complet de ce travail qui se situe dans le cadre de mon projet de fin d'études. L'objectif principal de ce projet étant de concevoir et de développer une application mobile de location de voitures de luxe pour les clients de Swiss Premium Negoce, l'entreprise accueillante.

On a entamé ce rapport par une présentation du cadre général du projet où on a présenté l'entreprise accueillante et ses activités, puis on a présenté différentes applications qui offrent des services de location de voitures et service taxi, ces applications ont présenté des inconvénients. On a dégagé d'après ses inconvénients la problématique ayant engendré le besoin d'une telle application. Pour réaliser cette application, on a dû ensuite séparer ses besoins en besoins fonctionnels et non fonctionnels qui ont permis de commencer l'étape de conception, où on a décrit les différentes fonctionnalités nécessaires. Cette étape a permis d'identifier les technologies et outils à utiliser pour arriver à réaliser ce projet dans les délais choisis.

Une fois les outils et technologies choisies, on a passé au développement de l'application, où on a commencé par le développement des interfaces graphiques avec Flutter et leurs fonctionnalités avec Javascript et ExpressJS en côté serveur, au fin de développement de chaque interface et ses fonctionnalités, elle est testée pour dégager les différentes erreurs qui peuvent se déclencher et les corriger, jusqu'au développement de la dernière interface de l'application.

Ce projet fut une expérience enrichissante et fructueuse qui m'a permis d'acquérir de nouvelles compétences de grande valeur dans plusieurs domaines à la fois : Le développement web, développement mobile et le design des interfaces graphique et l'expérience utilisateur (UI/UX Design), en particulier le développement des applications côté serveur avec ExpressJS, développement des applications mobiles avec Flutter et la création des maquettes des interfaces utilisateur avec Adobe XD. Ainsi, ce stage qui représente une nouvelle expérience professionnelle,

m'a été bénéfique, il a été une nouvelle chance pour consolider mes compétences techniques et mettre en pratique mes compétences théoriques.

J'ai appris à être autonome, ouvert aux autres, et plus conscient de la vie professionnelle qui nécessite la ponctualité, le sérieux et le travail acharné. J'ai pris la responsabilité et j'ai développé l'application qui répondait aux besoins définis dans le cahier de charges, les exigences, et les délais.

Comme tout projet informatique, des contraintes ont été rencontrées et elles ont été surmontées tout au long de la réalisation du projet. Elles ont constitué un défi pour acquérir de nouvelles connaissances techniques et développer les capacités opérationnelles. Ces défis concernent notamment la familiarisation avec les nouveaux frameworks et la maîtrise de certains outils techniques.

Pour le futur de cette application, on a plusieurs objectifs planifiés, dont on cite :

- Une migration vers firebase pour utiliser tous les services offerts par cette plateforme afin de réduire le nombre de bases de données utilisées.
- Le développement des applications SPN-Cars Driver et SPN-Cars Admin où il y aura une communication entre ces trois applications et assurer un service rapide et de qualité pour le client.

Ses objectifs, une fois atteints, permettront d'améliorer l'expérience de l'utilisateur et vont ouvrir les portes pour trouver plusieurs autres améliorations.

J'espère que le présent rapport soit suffisamment clair et structuré pour que le lecteur ait une idée précise et complète sur les différentes tâches que j'ai effectuées.

---

## **Bibliographie**

---

- [1] *Swiss Premium Negoce.* <https://www.swisspremiumnegoce.com>. Consulté le 02 Juillet 2022.
- [2] *Sixt Car hire.* <https://www.sixt.com/>. Consulté le 02 Juillet 2022.
- [3] *Blacklane.* <https://www.blacklane.com/>. Consulté le 02 Juillet 2022.
- [4] *Uber.* <https://www.uber.com/>. Consulté le 02 Juillet 2022.
- [5] *Méthode en cascade.* <https://blog-gestion-de-projet.com/modele-en-cascade/>. Consulté le 02 Juillet 2022.
- [6] *Adobe XD.* <https://www.adobe.com/products/xd.html>. Consulté le 02 Juillet 2022.
- [7] *Flutter.* <https://flutter.dev/>. Consulté le 02 Juillet 2022.
- [8] *ExpressJS.* <https://expressjs.com/>. Consulté le 02 Juillet 2022.
- [9] *Mongodb.* <https://www.mongodb.com/>. Consulté le 02 Juillet 2022.
- [10] *Firebase.* <https://firebase.google.com/>. Consulté le 02 Juillet 2022.
- [11] *Swagger.* <https://swagger.io/>. Consulté le 02 Juillet 2022.
- [12] *Visual Studio Code.* <https://code.visualstudio.com/>. Consulté le 02 Juillet 2022.
- [13] *Microsoft.* <https://www.microsoft.com>. Consulté le 02 Juillet 2022.
- [14] *Postman.* <https://www.postman.com/>. Consulté le 02 Juillet 2022.
- [15] *Postman atteint 20 million d'utilisaateurs.* <https://venturebeat.com/2022/04/19/postman-api-platform-hits-20m-users-helps-drive-the-api-economy/>. Consulté le 02 Juillet 2022.
- [16] *Git.* <https://git-scm.com/>. Consulté le 02 Juillet 2022.



## ESPRIT SCHOOL OF ENGINEERING

**www.esprit.tn - E-mail : contact@esprit.tn**

**Siège Social : 18 rue de l'Usine - Charguia II - 2035 - Tél. : +216 71 941 541 - Fax. : +216 71 941 889**

**Annexe : 1-2 rue André Ampère - 2083 - Pôle Technologique - El Ghazala - Tél +216 70 250 000 - Fax +216 70 685454**

