



École Supérieure Multinationale des Télécommunications



BÉNIN



BURKINA FASO



GUINÉE



MALI



MAURITANIE



NIGER



SÉNÉGAL

MÉMOIRE FIN DE FORMATION POUR L'OBTENTION DU DIPLÔME DE LICENCE PROFESSIONNELLE



OPTION : TELECOMMUNICATIONS ET INFORMATIQUE

SPÉCIALITÉ : EF FMPQQFNFO !E QQMD PQSFQ S FT!

THÈME

Conception et réalisation d'une application de covoiturage : Ecovoiturage

Sous la direction de

M. Moustapha DER

Enseignant

!! FTN

Présenté et soutenu par

Mme Bark-wende Melissa D.ZANNE

Promotion 2018- 2021
Decembre 2022

DEDICACE

A mon PAPA et à ma MAMA, pour tous leurs sacrifices, leur amour, leur tendresse, leur soutien et leurs prières tout au long de mes études,

A mon frère, ma sœur et ma cousine pour leurs encouragements permanents, et leur soutien moral dans les moments difficiles,

Merci à mon Oncle et ma Tante ici à Dakar, toujours présent pour moi,

A toute ma famille, mes amis et ma colocataire pour leur soutien tout au long de mon parcours universitaire,

Que ce travail soit l'accomplissement de vos vœux tant allégués, et le fruit de votre soutien infailible,

Merci d'être toujours là pour moi.

REMERCIEMENTS

On remercie Dieu le tout puissant de nous avoir donné la santé et la volonté d'entamer et de terminer ce mémoire.

Tout d'abord, ce travail ne serait pas aussi riche et n'aurait pas pu voir le jour sans l'aide et l'encadrement de Mr DER je le remercie pour la qualité de son encadrement exceptionnel, pour sa patience, sa rigueur et sa disponibilité durant la préparation de ce mémoire.

Mes remerciements s'adressent également à tous nos professeurs pour leurs générosités et la grande patience dont ils ont su faire preuve malgré leurs charges académiques et professionnelles.

LISTE DES CAPTURES

Capture 1: Image montrant les plugins de react-native-dotenv	40
Capture 2: Capture montrant la bibliothèque dotent, jailbreaké et jail-monkey	41
Capture 3: Page de choix de statut	42
Capture 4: Page d'inscription du client.....	43
Capture 5: Page d'inscription en tant que société	44
Capture 6: Page de publication des différentes offres de covoiturages	45
Capture 7: Détails d'un post.....	46
Capture 8: Liste des véhicules disponible d'une société.....	47
Capture 9: Détails des informations d'ajout de voiture d'une société	48
Capture 10: Suite des détails des informations d'ajout de voiture d'une société	49

LISTE DES FIGURES

Figure 1 : Diagramme de cas d'utilisation de l'administrateur.....	12
Figure 2 :Diagramme de cas d'utilisation de la société et du client ..	13
Figure 3 : Diagramme de classe	14
Figure 4: Logos de react native et IONIC	17
Figure 5: Courbe comparative entre utilisateurs de react native et IONIC	18
Figure 6: Logos de Flutter et react native.....	23
Figure 7 : Comparaison des frameworks cross-platform en 2020	27
Figure 8 : Illustration de react native sur smartphone.....	28
Figure 9 : Logo Firebase.....	30
Figure 10 : Logo mongodb	30
Figure 11 : Logo dynamodb	31
Figure 12 : Logo SQLite.....	32
Figure 13 : Architecture de l'application	36
Figure 14 : Logo Visual Studio Code.....	38
Figure 15 : Logo Github	38
Figure 16 : Logo Modelio.....	

SIGLES ET ABREVIATIONS

API	Application Programming Interface
BDD	Base de Données
CSS	Cascading Style Sheets
DAC	Diagramme d'activités
DCU	Diagramme des cas d'utilisation
DCL	Diagramme de classe
DES	Diagramme de Séquence
DET	Diagramme état-transition
DPL	Diagramme de déploiement
DOM	Document Object Model
HTML	HyperText Markup Language
J2ME	Java 2 Platform, Micro Edition
JS	JavaScript
JSX	JavaScript Syntax eXtension
MVC	Modèles Vues Contrôleurs
SGBD	Système de Gestion de Base de Donnée
SQL	Structured Query Language
SMS	Shorts Message Service
UML	Unified Modeling Language
SDK	Software Development Kit

LISTE DES TABLEAUX

Tableau 1 : Tableau récapitulatif de Ionic et React Native	22
Tableau 2 : Tableau comparatif entre Flutter et React Native	

AVANT-PROPOS

L'Ecole Supérieure Multinationale des Télécommunications (ESMT) a pour vocation de former des diplômés (Techniciens supérieurs, Licences Professionnelles, Ingénieurs, Masters, Masters spécialisés) dans les domaines techniques et managériaux des télécommunications/TIC. Elle accueille en formation initiale ou continue des stagiaires qui proviennent de l'ensemble des pays francophones d'Afrique, recrutés au niveau des écoles, des universités ou directement chez les opérateurs de télécommunications.

Le mémoire de fin d'études de l'ESMT est très important dans le parcours de l'étudiant.

Ce mémoire a pour but premier d'évaluer et d'attester de nos compétences et capacités acquises lors des trois années de formation effectuées. En effet lors de ce mémoire nous associerons toutes nos compétences acquises pour l'analyse et le développement du thème de ce mémoire. Ce mémoire nous permet aussi de faire valoir nos capacités à réfléchir sur la problématique dégagée par un sujet ainsi que sa maîtrise et par la même occasion nos connaissances

Sommaire

INTRODUCTION.....	1
Partie 1 : ETUDE PREALABLE SPECIFIQUE ET ANALYSE DES BESOINS	2
1.1 Contexte du sujet.....	2
1.2 Problématique du sujet.....	3
1.3 Solutions à la problématique.....	3
1.4 Enquête sur la mise en place de l'application	3
1.5 Cahier de charge.....	4
1.6 Aspect économique	7
Partie 2 : LA MODELISATION STRUCTURELLE ET COMPORTEMENTALE	9
2.1 Choix de la technologie de modélisation	9
2.2 Présentation de l'UML.....	9
2.3 Diagramme des cas d'utilisations (DCU)	11
2.4 Diagramme de classe	14
Partie 3 : LA REALISATION ET CONCEPTION	15
3.1 Les technologies mobiles	15
3.2 Applications natives	15
3.3 Application web ou connectés	16
3.4 Les applications mobiles hybrides	16
3.5 Concepts sur les technologies multiplateformes.....	16
3.6 Choix de technologie de réalisation	27
3.7 Choix du back-end	29
3.8 L'architecture de EcoVoiturage	34
3.9 Mise en place de l'environnement de développement.....	37
3.10 Outils de développement.....	37
3.11 Sécurité	39
3.12 Interface de EcoVoiturage	42
Partie 4 : APPORTS	50
4.1 Apports sur le plan technique.....	50
4.2 Apports au niveau de la conception et du développement.....	50
Partie 5 : CONCLUSION ET PERSPECTIVES	51

INTRODUCTION

Le Sénégal est fortement touché par les conséquences du changement climatique, à l'instar des grandes villes africaines, et vit une urbanisation galopante plus marquée dans la capitale Dakar, mais aussi dans d'autres villes du pays.

Dans un contexte d'urbanisation massive, et grandement touché par le changement climatique nous avons mis en place une application mobile de covoiturage. Le covoiturage est un mode de déplacement où plusieurs personnes utilisent une seule voiture pour faire le même trajet ou presque. La principale difficulté du covoiturage réside dans son essence même, c'est-à-dire trouver un partenaire avec qui faire le trajet. Pour pallier ce problème, nombreux sites Internet proposent des solutions qui ne sont pas toujours efficaces pour l'utilisateur qui vit au Sénégal. Celle-ci a pour but de réduire les files interminables des bouchons, soulager un peu notre portefeuille ainsi que donner un déclic sur les questions écologiques et réchauffement climatique que subit le Sénégal et la TERRE.

Notre présentation sera répartie en 5 parties. Dans la première partie nous ferons une étude préalable, spécifique et l'analyse des besoins ; dans la deuxième partie pour une bonne analyse et une bonne architecture de notre application nous aurons à faire la modélisation et la conception de l'application ; dans la troisième partie nous aurons les réponses sur les choix des technologie utilisées pour la réalisation de l'application, pour la quatrième partie nous parlerons des différents apport et enfin dans la dernière partie nous aurons la conclusion et les perspectives.

Partie 1 : ETUDE PREALABLE SPECIFIQUE ET ANALYSE DES BESOINS

Nous procédons maintenant à une présentation générale du sujet : elle contiendra une étude de l'existant, qui nous permettra de cerner la problématique et de présenter le projet

1.1 Contexte du sujet

Face à la montée du prix de l'essence, les kilomètres d'embouteillages, la pollution, le réchauffement climatique, le covoiturage est une solution. Pourtant ce système n'a pas vraiment la côte en Afrique surtout au Sénégal.

Le covoiturage ne fait l'objet d'aucune définition officielle, ni de législation spécifique. Selon Séverine Millet, auteur de « La stratégie du Colibri » le covoiturage "a pour vocation de mettre en relation des personnes effectuant seules tout ou partie d'un trajet identique, afin qu'elles voyagent désormais à plusieurs. Ce système permet finalement de diminuer le nombre de voitures en circulation pour un même déplacement". Dans notre application nous prendrons en considération les situations réelles du Sénégal c'est à dire en ce qui concerne les raisons de sécurité, l'offre et la demande.

Les trajets peuvent être de différentes natures, réguliers ou irréguliers : domicile-travail, domicile-université, domicile-supermarché, domicile-loisirs, domicile-gare, point de rencontre a un lieu X mais aussi trajets longues distances.

Pour les utilisateurs, les principaux atouts sont souvent le partage des frais de déplacement liés à la voiture et la convivialité du trajet effectué à plusieurs. C'est aussi et surtout un moyen de réduire son empreinte carbone.

D'un milieu à l'autre, le covoiturage n'est pas implanté de la même manière. Dans le but d'offrir un service qui correspond à la demande sénégalaise, une analyse de leurs habitudes de transport a été faite. À partir des données recueillies, on pourra définir un système adapté à leur besoin.

1.2 Problématique du sujet

Comme nous l'avons spécifié précédemment, le covoiturage est un Système de transport qui consiste à partager l'utilisation d'une voiture entre plusieurs personnes. Nous avons essayé d'adapter ce concept aux réalités du Sénégal. Aussi, nous nous sommes posées pleins de questions en ce qui concerne son aspect écologique, économique et utilitaire.

Il se pose alors le problème qui suit : “Comment résoudre tous ces problèmes dans une applications mobile tout en donnant notre modeste contribution à la réduction de production d'effet de serre, des bouchons interminables tout en allégeant nos portefeuilles ?

1.3 Solutions à la problématique

Aujourd'hui, les villes, face à une sur-urbanisation, voient leur système de mobilité en saturation, qu'il s'agisse des réseaux de transports public ou automobile. Les solutions anciennes ne fonctionnent plus : on ne peut plus créer de nouvelles avenues. Il y a donc un besoin, qui se trouve décuplé par les enjeux environnementaux, et notamment celui de la mobilité décarbonée. La mobilité est donc aujourd'hui un enjeu premier pour les villes, quelle que soit leur taille. Les enjeux tournent autour de l'accessibilité des territoires, la capacité de relier petites villes et agglomérations... Cependant, nous avons la chance de vivre une période où la numérisation peut faire émerger des solutions

C'est dans ce sens que se penchera notre application. Elle permettra de mettre en relation des utilisateurs non véhiculés proche de chez vous qui partent dans la même direction ou presque pour faire un trajet en communs.

De mettre en relation des voyageurs voulant faire de grandes distances avec des sociétés de taxi préalablement inscrites dans l'application.

Ainsi Ecovoiturage nous évitera de faire un trajet seul, de faire des economies, réduire les bouchons interminables, les accidents, Socialiser en voiture.

1.4 Etat de l'art

Pour un développeur, mettre en place une application est un processus assez complexe. Elle se fait après une réflexion, une analyse des besoins et difficultés quotidiennes autour de lui et selon son vécu.

C'est dans ce sens que pour nous rassurer de la pertinence de EcoVoiturage, nous avons entrepris des enquêtes aux niveau de la société allo taxi, des particuliers et des clients.

Entretien avec la société Allo taxi

Cet entretien avait pour but de faire connaître notre projet et vérifier si l'idée de concevoir une telle application était bénéfique pour les professionnelles du domaine.

Au cours de l'entretien avec monsieur Ousmane et monsieur Aziz SENI PDG de Allo taxi et investisseurs de différents projets à succès, nous avons eu à aborder les points suivants :

Les questions écologiques

L'efficacité du covoiturage, à savoir les questions de sécurité, temps

L'impact économique sur sa société

Mais aussi nous avons eu à le convaincre sur les points suivants :

- + La qualité de l'application ;
- + La sécurité ;
- + L'intégrité ;
- + Le bon fonctionnement ;
- + La fiabilité ;
- + L'accessibilité.

1.5 Cahier de charge

Dans cette section nous décrirons les caractéristiques, les attentes et les contraintes liées au développement et à la mise en place de notre solution de covoiturage. Nous présenterons d'abord les objectifs de notre solution, puis les utilisateurs qui interagiront avec elle avant de présenter les besoins fonctionnels auxquels doit répondre la solution. Enfin, nous présenterons l'aspect économique de notre application pour la monétisation de cette dernière.

1.5.1 Les Objectifs

Les objectifs de notre solution ont été définis autour des axes suivants : le sociale, l'environnement et l'économie.

- ✚ **Le social** : Ecovoiturage permettra à ces utilisateurs d'avoir des trajets plus conviviaux notamment par un système de feedback pour que les pourvoyeurs de véhicules ainsi que les passagers puissent se noter mutuellement et ainsi forcer un minimum de savoir vivre. Cela améliorera aussi la sécurité pour nos utilisateurs (surtout celle de la gente féminine) qui restent dubitatif soit plus rassuré de la fiabilité de notre solution.
- ✚ **L'environnement** : Ecovoiturage a pour objectif pour l'environnement de réduire l'empreinte carbone de ses utilisateurs. Nous arriverons à ce résultat grâce au système du covoiturage lui-même qui permet de réduire le nombre de véhicule en circulation et par conséquent réduit les émissions de gaz carbonique.
- ✚ **L'économie** : L'un des aspects qui séduisent les pratiquants du covoiturage est l'une des caractéristiques de cette dernière leurs permettant de faire des économies d'échelle. Nous avons donc pour objectif de mes en places à moyens-long terme d'améliorer cet aspect de notre solution avec des fonctionnalités plus poussées d'analyse des trajets les plus communs et par exemple fixer un prix pour ces derniers.

1.5.2 Les Utilisateurs

Nous pouvons les diviser en 3 grandes parties

- ✚ Les entreprises de taxi : ce sont des entreprises de taxi qui mettent à la disposition de l'applications des taxis.
- ✚ Les clients : nous avons 2 types de clients à savoir ceux qui proposent un covoiturage avec leurs véhiculés personnels et ceux qui demandes des covoiturations à travers les sociétés de taxi.
- ✚ Les administrateurs : est en charge de la gestion des traitements et des données liées à l'application. Son rôle est très important, à savoir définition des accès utilisateur (ajouter, supprimer, bannir un utilisateur).

1.5.3 Les Fonctionnalités

Tout nouvel utilisateur doit d'abord s'inscrire. Nous aurons deux options au niveau de l'inscription. La possibilité de s'inscrire en tant que client ou de s'inscrire en tant que société de taxi.

+ Les clients :

- S'inscrire, se connecter
- Gestion de son compte : Modifier son profil et ses informations de connexion
- Rechercher un covoiturage
- Suivre le trajet
- Lancer un covoiturage

+ Pour les sociétés de taxi :

- S'inscrire, se connecter
- Gestion de son compte : Modifier son profil et ses informations de connexion
- Gérer ses taxis et services : Créer, supprimer, mettre à jour ses taxi et les différents conducteurs
- Gérer ses commandes : Consulter, confirmer des trajets

+ Pour les administrateurs :

- Se connecter
- Gestion de son compte : Modifier son profil et ses informations de connexion
- Gérer les clients et les sociétés de taxi : Créer, mettre à jour, consulter, supprimer, activer et désactiver un compte

1.5.4 Données utilisateurs collectés

Au-delà du respect écologique cette dernière nous voulons être transparent sur les données utilisées par EcoVoiturage. Ici nous présentons les principales données collectées pour le fonctionnement de notre solution.

- + Le client souhaitant le covoiturage doit permettre sa localisation afin que EcoVoiturage lui propose les combinaisons les plus adéquates pour son déplacement.

- ✚ Pour chaque société, on souhaite avoir comme informations sur le chauffeur tel que :
 - Son nom, prénom,
 - Numéro de téléphone,
 - La catégorie du permis.
- ✚ Pour chaque véhicule, on souhaite connaître :
 - Sa marque, son modèle,
 - Sa catégorie (Berline, Citadine, Monospace, ...),
 - Sa motorisation (Essence ou Diesel),
- ✚ Pour chaque covoiturage proposer par les sociétés de taxi, on souhaite connaître :
 - La ville de départ et d'arrivée,
 - La date du voyage,
 - La distance,
 - Le nombre de places disponibles,
 - Le prix.

1.6 Aspect économique

Lorsque nous décidons de créer une application mobile il est important de se poser des questions par rapport à la rémunération de celle-ci. Elle doit être adaptée en fonction de nos critères et du type de revenu que nous souhaitons.

Ainsi, nous avons 5 manières de pouvoir monétiser notre application mobile :

- ✚ **Les applications payantes** : lorsque vous voudriez la télécharger vous êtes obligé de payer un montant. Par exemple sur App store il y a un montant bien déterminé juste à côté de l'application. Il est important de savoir qu'avec ce type de monétisation Apple perçoit 30% de commission.
- ✚ **Les applications par abonnement** : ici l'application est gratuite mais lorsqu'on voudra plus de fonctionnalités nous devons payer un abonnement. Comme exemple nous avons Spotify dans ce type de monétisation Apple prend également une commission de 30%
- ✚ **Les Applications séparées** : en effet nous avons 2 applications ici. La 1^{ère} est gratuite mais par contre lorsque l'utilisateur voudra plus de fonctionnalités nous

mettrons un pop-up et nous lui demanderons de télécharger la 2^{ème} application. Nous pouvons assimiler la première application à une offre basique avec des fonctionnalités limitées et la seconde comme une offre premium disposant de l'intégralité des fonctionnalités.

✚ **Les in-Approchasse** : il s'agit de faire des achats à l'intérieur de l'applications. Cette manière de faire est plus utilisée avec les jeux vidéo. Nous avons affaire à une application gratuite mais lorsque nous voudrions atteindre un autre niveau ou vouloir payer des accessoires nous sommes obligés de payer. Ce sont des achats qui se font à l'intérieur de l'application, nous avons l'exemple du jeu vidéo de Kim Kardashian.

✚ **Les publicités** : c'est un modèle assez classique. L'application est offerte de manière gratuite mais en échange des pubs. Pour EcoVoiturage, nous avons décidé d'utiliser ce type de rémunération. Elle est utilisée par les plus grosses entreprises tel que Facebook ou Google. Il est important de savoir qu'il y a plusieurs types de publicités. Nous retiendrons les 4 les plus utilisées :

- Les Banners : c'est un petit bandeau qu'on met juste au-dessous de l'écran, elle n'est pas vraiment dérangeante
- L'interstitial : elle occupe toute l'écran, on est obligé de cliquer sur une croix pour la faire disparaître
- Le Reword : c'est l'utilisateur même qui demande à la regarder
- In app : elles se trouvent en plein milieu des postes et passent souvent inaperçues. Méthode utilisée par Facebook

Partie 2 : LA MODELISATION STRUCTURELLE ET COMPORTEMENTALE

2.1 Choix de la technologie de modélisation

Dans ce chapitre, nous présenterons la conception détaillée du projet. Pour cela nous avons utilisé le langage UML comme approche de conception. Par la suite nous présenterons ce langage ainsi que le pourquoi de notre choix.

2.2 Présentation de l'UML

L'UML (pour Unified Modeling Language, ou "langage de modélisation unifié" en français) est un langage permettant de modéliser nos classes et leurs interactions. Autrement c'est un ensemble de notations graphiques s'appuyant sur des diagrammes et permettant de spécifier, visualiser et de documenter les systèmes logiciels orientés-objet. De plus, la modélisation UML permet de vulgariser les aspects liés à la conception et à l'architecture, propres au logiciel, au client. Aussi, elle apporte une compréhension rapide du programme à d'autres développeurs externes en cas de reprise du logiciel et facilite sa maintenance. Pour finir, elle facilite la représentation et la compréhension de solution objet. UML propose, en outre, un méta modèle de tous les concepts et notations associées utilisés dans les treize diagrammes du langage de modélisation regroupés en deux ensembles (diagrammes structurels et diagrammes de comportement).

2.2.1 *Les diagrammes structurels*

Les diagrammes structurels représentent l'aspect statique d'un système.

- ✚ **Diagramme de classes (DCL)** : représente la description statique du système en intégrant dans chaque classe la partie dédiée aux données et celle consacrée aux traitements ;
- ✚ **Diagramme d'objets (DOB)** : représente les instances des classes (objets) et des liens entre instances ;
- ✚ **Diagramme de composants (DCP)** : représente les différents constituants (fichiers sources, bibliothèques exécutables, etc.) du logiciel au niveau de l'implémentation du système ;

- ✚ **Diagramme de déploiements (DPL)** : montrent la disposition physique du matériel qui compose le système et la répartition des composants sur ce matériel ;
- ✚ **Diagramme de paquetages (DPA)** : donne une vue d'ensemble du système structuré en paquetage ;
- ✚ **Diagramme de structures composites (DSC)** : décrit la structure interne d'un ensemble complexe composé par un exemple de classes ou d'objets et de composants techniques.

2.2.2 Les diagrammes de comportement

Les diagrammes de comportement représentent la partie dynamique d'un système réagissant aux événements.

- ✚ **Diagramme des cas d'utilisation (DCU)** : identifient les utilisateurs du système (acteurs) et leurs interactions avec le système ;
- ✚ **Diagramme d'état-transitions (DET)** : représente les différents états des objets en réaction aux événements ;
- ✚ **Diagramme d'activités (DAC)** : représente le comportement d'une méthode ou le déroulement d'un cas d'utilisation ;
- ✚ **Diagramme de séquences (DSE)** : décrit les scénarios de chaque cas d'utilisation en mettant l'accent sur la chronologie des opérations en interaction avec les objets ;
- ✚ **Diagramme de communications (DCO)** : ce diagramme est une autre représentation des scénarios des cas d'utilisations qui met plus l'accent sur les objets et les messages échangés ;
- ✚ **Diagramme global d'interactions (DGI)** : fournit une vue générale des interactions décrites dans le diagramme de séquence et des flots de contrôle décrits dans le diagramme d'activités ;
- ✚ **Diagramme de temps (DTP)** : ce diagramme permet de représenter les états et les interactions d'objets dans un contexte où le temps a une forte influence sur le comportement du système à gérer.

2.3 Diagramme des cas d'utilisations (DCU)




Un diagramme de cas d'utilisation (DCU) permet de décrire l'interaction entre les acteurs (utilisateurs) et le système.

2.3.1 *Identificateurs des acteurs*

Un acteur représente une entité externe qui interagit avec l'application (fournir, recevoir, échanger de l'information). En observant les utilisateurs directs de l'application, nous trouvons ces acteurs qui opèrent avec l'application : l'Administrateur, les clients et les différentes sociétés de taxi.

2.3.2 *e foug j!bujpo!e ft!dbt!e vujn! bujpo*

Pour chaque acteur nous distinguons les cas d'utilisations suivants :

-  -Administrateur
-  -clients
-  -société

Pour faciliter la compréhension nous avons procédé à la décomposition en plusieurs diagrammes

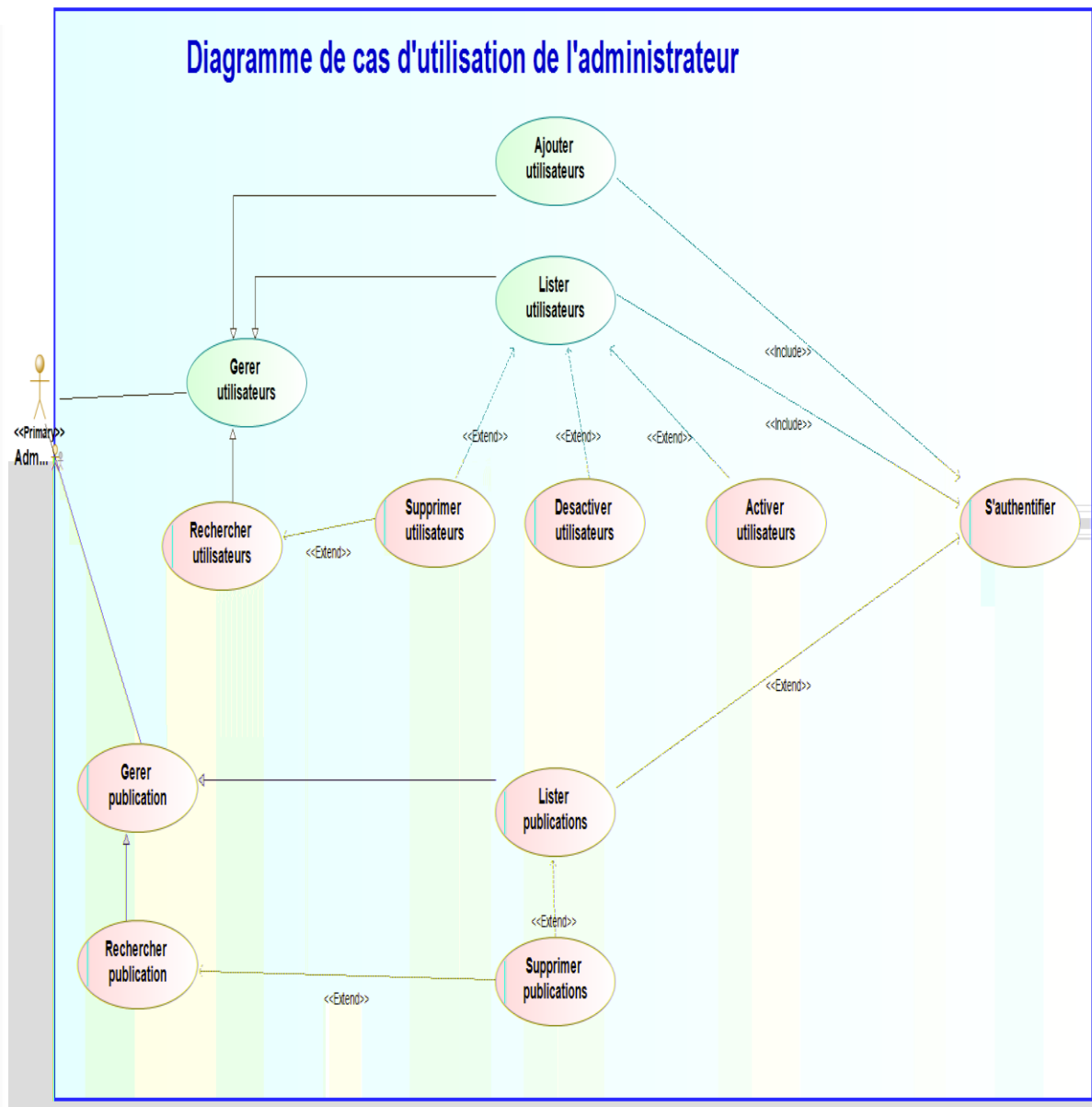


Figure 1 : Diagramme de cas d'utilisation de l'administrateur

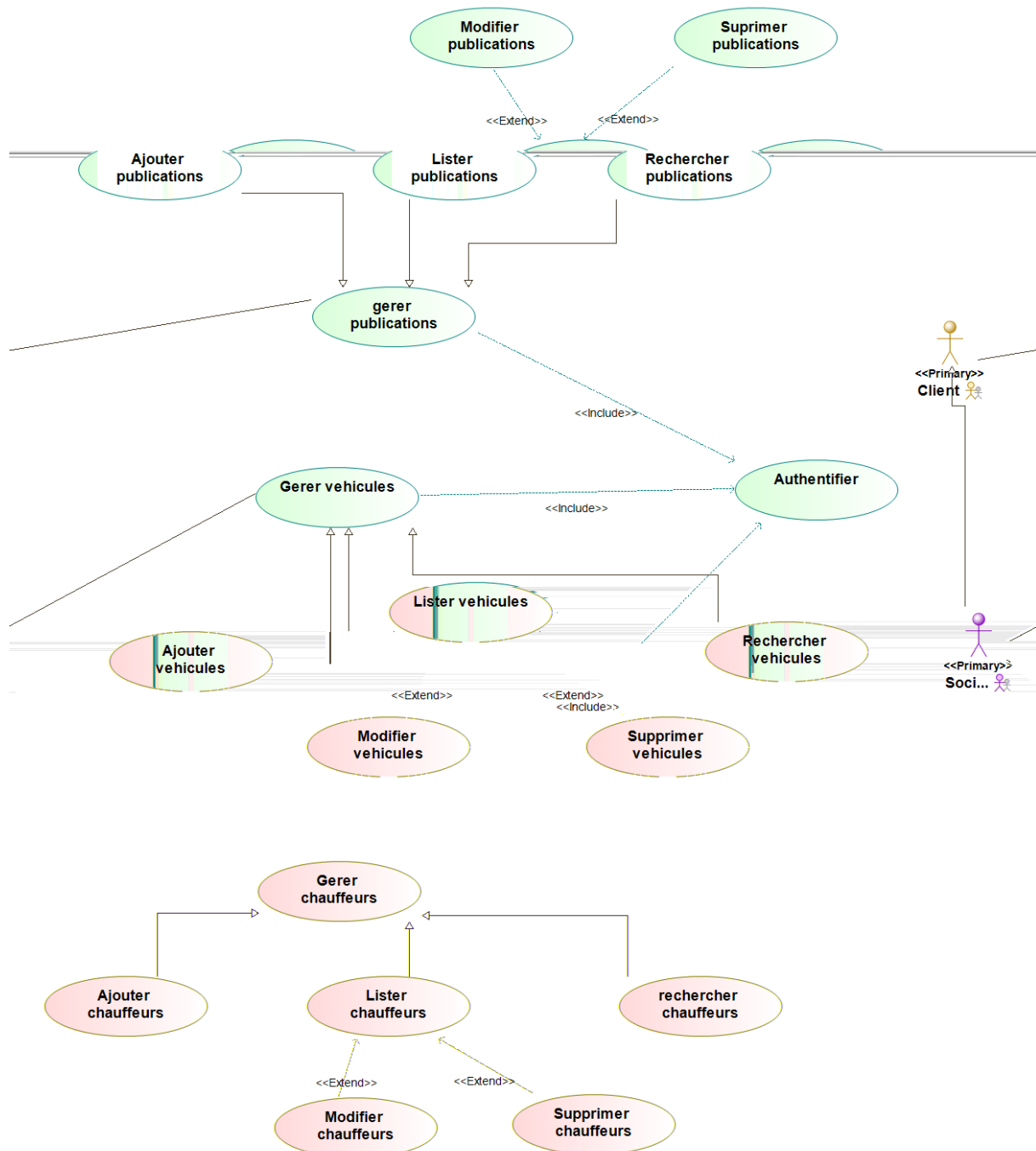


Figure 2 : Diagramme de cas d'utilisation de la société et du client

2.4 Diagramme de classe

Un diagramme de classes fournit une vue globale d'un système en présentant ses classes, interfaces et collaborations, et les relations entre elles. Les diagrammes de classes sont statiques : ils affichent ce qui interagit mais pas ce qui se passe pendant l'interaction.

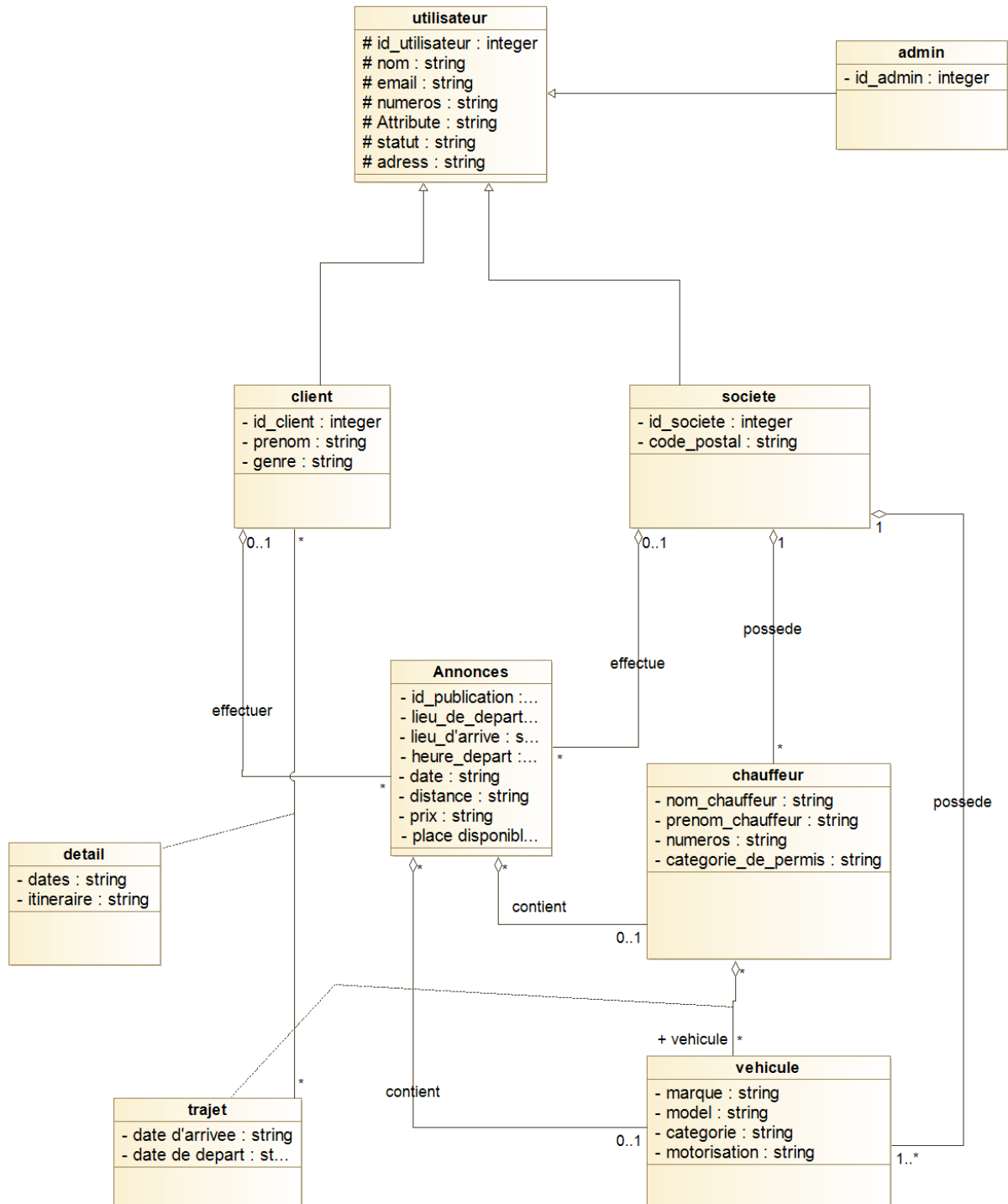


Figure 3 : *Diagramme de classe*

Partie 3 : LA REALISATION ET CONCEPTION

3.1 Les technologies mobiles

Ce n'est un secret pour personne : on accède de plus en plus à Internet au moyen d'appareils mobiles. Comme un téléphone intelligent ou une tablette plutôt que des traditionnels ordinateurs de bureau ou portables. Essentiellement, chez les utilisateurs, la commodité l'emporte sur la performance. Ce changement dans le comportement des consommateurs a une incidence profonde sur la conception de sites Web. Toute stratégie de sensibilisation ou de communication sur le Web doit prendre en considération les appareils mobiles dès le départ, et non après coup. Ainsi cela va consister en la mise à disposition, l'amélioration et la mise à jour fréquentes d'outils et d'environnements de développement spécialisés.

3.2 Applications natives

Tout d'abord comment définir une application native ?

Les applications natives mobiles sont des programmes développés pour fonctionner sur les systèmes d'exploitation utilisés par les smartphones et tablettes. Ces programmes, installés localement sur le "device", utilisent les capacités de ce dernier, couplées à celui de son OS, pour pouvoir fonctionner normalement.

Les deux principaux systèmes d'exploitation (OS) mobiles sont aujourd'hui :

Android (76,5% de part de marché en France), présent sur la plupart des smartphones. Les applications Android sont développées en langage de programmation Java ou Kotlin.

iOS (23,5% de part de marché en France), présent uniquement sur les iPhones de la marque Apple. Les applications iOS sont développées en langage Swift (auparavant Objective-C).

Les applications mobiles natives ont des finalités très larges : aider l'utilisateur à s'organiser, à communiquer, à utiliser des services, à acheter des biens, à interagir avec son environnement, à créer quelque chose, à accéder à de l'information pertinente pour ses prises de décisions...

3.3 Application web ou connectés

Les applications Web ou connectés ont une interface utilisateur qui permet aux utilisateurs de se connecter à divers sites Web via une connexion Internet et d'accéder aux données ou au contenu disponibles. ... Ils sont accessibles depuis n'importe quel appareil avec une connexion Internet de n'importe où. Ces applications sont entièrement construites avec le langage Web et fonctionnent avec tous les navigateurs pour mobile. En revanche pas profiter pleinement des possibilités graphiques des appareils et requiert généralement une connexion internet

3.4 Les applications mobiles hybrides

Une application hybride est une application logicielle qui combine des éléments d'applications natives et d'applications Web.

Les applications hybrides sont populaires car elles permettent aux développeurs d'écrire du code pour une application mobile une fois tout en prenant en charge plusieurs plates-formes.

Ce type d'application offre une facilité dans la maintenance parce qu'il n'y a qu'une seule version pour plusieurs plateformes : « cross-platform », React native, flutter, PhoneGrap (...); sont des technologies multiplateformes.

Étant donné que les applications hybrides ajoutent une couche supplémentaire entre le code source et la plate-forme cible, elles peuvent fonctionner légèrement plus lentement que les versions natives ou Web de la même application.

3.5 Concepts sur les technologies multiplateformes

Nous sommes dans une ère où les technologies de l'informations et de la communication, les smartphones et les tablettes font partis intégrants de notre quotidien.

Un logiciel multiplateforme est un logiciel conçu pour fonctionner sur plusieurs plates-formes, c'est-à-dire le couple liant ordinateur et système d'exploitation

Proposer des applications mobiles devient un enjeu stratégique. Surtout que le marché des applications mobiles ne finit pas de s'accroître. La diversité qui existe dans le domaine mobile, notamment le nombre important de systèmes d'exploitation qui utilisent des technologies différentes, engendre cependant une fragmentation : environnement

IOS/Objective-C pour l'iPhone et l'iPad, SDK Java spécifique pour Android, J2ME pour Symbian, etc.

En effet, le fait de pouvoir déployer un logiciel sur plusieurs environnements permet non seulement d'économiser du temps (temps passé à reproduire le code pour chaque plateforme cible) mais aussi d'économiser des ressources (coût de production). De plus, un gain non négligeable émane de cette notion de développement multiplateforme : la maintenance. Dans le domaine informatique, la maintenance constitue en moyenne près de 70% des coûts, avoir ainsi un seul logiciel à maintenir reviendrait considérablement moins cher que si on en avait plusieurs. Conscients de ses avantages et soucieux d'optimiser les processus de conception des applications mobiles, l'idée de développer une application unique qui fonctionne partout (ou presque partout) est devenu un objectif qui était beaucoup plus difficile à atteindre, mais qui reste aussi attractif que jamais.

L'objectif de l'étude comparative des technologies est de mettre en évidence les avantages et les inconvénients des différents Framework cross-platform les plus répandus tel que Ionic, PhoneGap, react native, et flutter dans le monde mobile afin d'expliquer notre choix.

3.5.1 *React native contre IONIC*



Figure 4: Logos de react native et IONIC

Les deux ont une large communauté autour d'eux, avec une forte adoption par les grandes entreprises et des applications mobiles utilisées par un très grand nombre de clients. Voici ce que nous pouvons percevoir à partir de la courbe :

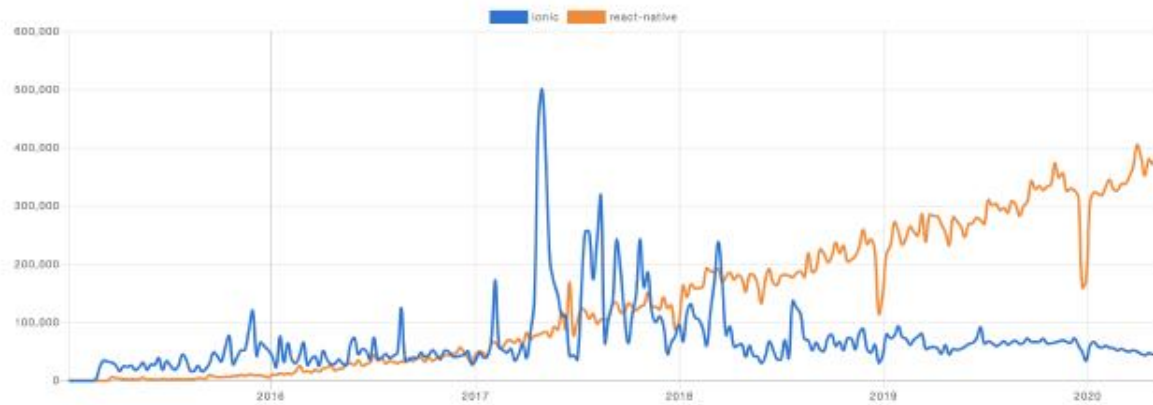


Figure 5: Courbe comparative entre utilisateurs de react native et IONIC

Comme le montre le graphique ci-dessus, **Ionic** a enregistré bien plus de téléchargements que **React Native** en 2017. Cela s'est produit car **Ionic** est plus ancien que **React Native**. A cette époque, il était plus mature, prêt à être utilisé dans le développement mobile, avec moins de bugs et plus stable. Après 2019, Facebook a fait plus d'efforts pour faire de **React Native** une bibliothèque stable, en la mettant à jour fréquemment, en l'utilisant dans ses propres produits et en présentant des vitrines sur les résultats.

Ionic propose également de nombreuses applications publiées sur le Play Store (Android) et l'App Store (iOS). L'application Untappd est l'un des exemples.

Les deux ont des caractéristiques différentes et des écosystèmes différents. Parlons de chacun d'eux individuellement pour mieux comprendre comment fonctionnent Ionic et React Native et ses avantages et inconvénients:

IONIC :

Ionic Framework est une boîte à outils d'interface utilisateur open source pour la création d'applications mobiles et d'applications **Web progressives (PWA)** à l'aide de technologies Web telles que HTML, CSS et **JavaScript**. Il fournit des composants basés sur la technologie Web

Aujourd'hui, vous pouvez créer une application mobile hybride dans Ionic avec Angular, React.js, VueJs ou même simplement avec JavaScript.

Voyons maintenant ses avantages :

➤ **Avantages :**

Possède une bonne documentation, car c'est une entreprise spécialisée dans la création d'outils qui aident les entreprises et les développeurs à créer des applications mobiles.

Ces outils sont bien compris par les développeurs Web car ils utilisent la vue Web pour rendre l'application (permettant aux développeurs d'utiliser des technologies quotidiennes telles que HTML, CSS et **JavaScript**).

Facilite le portage du code dans une **application Web progressive (PWA)**, car tout le code est conçu pour s'exécuter dans un navigateur web. Ionic nécessite juste quelques étapes supplémentaires pour pouvoir avoir un PWA. Ionic nécessite juste quelques étapes supplémentaires pour pouvoir avoir un PWA.

Bien que cela présente des avantages, nous pouvons également rencontrer certains défis, comme ceux énumérés ci-dessous :

➤ **Les inconvénients :**

Étant donné que l'organisation derrière le framework en vit, outre les fonctionnalités *fremium*, il existe également des fonctionnalités premium que vous ne pouvez utiliser que si vous payez un coût supplémentaire.

Ionic est construit sur le "navigateur Web". Le code de l'application ne peut pas accéder facilement aux fonctionnalités natives.

React Native :

Avec **React Native**, les développeurs peuvent créer des applications mobiles pour les plates-formes Android et iOS à l'aide de **JavaScript**, mais ils peuvent également implémenter des fonctionnalités avec du code natif. Il permet des builds multi-plateformes en fournissant des composants natifs indépendants de la plate-forme qui correspondent aux blocs de construction natifs de l'interface utilisateur des plates-formes.

Les développeurs peuvent également créer des applications TV avec **React Native**. Cela signifie qu'avec quelques étapes supplémentaires, il est possible de livrer des applications pour **Apple TV** et **Android TV**.

Maintenant, plongeons-nous dans quelques avantages de **React Native** :

➤ **Avantages**

Communauté massive autour de l'écosystème. À l'heure actuelle, il y a des nombres impressionnants sur **GitHub** repo [facebook/react-native](https://github.com/facebook/react-native). Cela signifie qu'il est très probable que les développeurs trouvent des réponses aux problèmes auxquels ils sont confrontés.

Capacité d'être intégré dans des applications natives existantes, comme montré dans cette présentation. En plus de cela, il est également possible de créer des modules natifs, ce qui rend l'apparence de votre application mobile aussi fluide qu'une application native.

React Native est traduit en code natif, avec l'avantage d'atteindre 60 images par seconde. Cela donne à l'utilisateur l'impression d'une application native pas du tout lente !

Il permet aux développeurs de fournir des applications pour Apple TV et Android TV, avec seulement quelques étapes supplémentaires.

Examinons maintenant les problèmes que nous avons rencontrés comme ci-dessous :

➤ **Les inconvénients :**

Puisqu'il y a une grande entreprise privée derrière cela, de nouveaux outils sont publiés en fonction de leurs propres besoins. Cependant, la communauté est libre d'étendre les fonctionnalités souhaitées, pour atteindre ce qui est nécessaire pour les technologies natives Android et iOS.

React Native est encore en version bêta. Oui, vous avez bien lu. Bien qu'il ait été créé il y a cinq ans, l'équipe de Facebook n'avait toujours pas la confiance nécessaire pour en sortir une version "production". Cela peut ressembler à : "être conscient des risques possibles".

Cela conduit souvent à des changements de rupture dans les outils et les dépendances entre les versions. Mais bon, Gmail était en version bêta depuis presque 5 ans et cela a plutôt bien fonctionné.

Tableau récapitulatif

	Ionic	React Native
Multiplateforme	Yes	Yes
Date de lancement	2013	2015
Entreprise	Ionic	Facebook
Applications connues	McLaren Automotive Diesel McDonald's	Facebook Uber s Airbnb
Web First	Oui	Non
Applications web progressives	Oui	Non
Catégorie	Web hybride	Hybride Native
Avantages	<ul style="list-style-type: none"> – Web First – Facile à personnaliser – Fonctionne sur l'environnement du navigateur 	<ul style="list-style-type: none"> – De meilleures performances – L'interface avec les commandes de l'interface utilisateur de l'application native est plus naturelle – Plus proche des applications natives

Inconvénients	<ul style="list-style-type: none"> – Peut être moins performant – Peut ne pas avoir exactement le même aspect et la même convivialité qu'une application native – Difficile à déboguer 	<ul style="list-style-type: none"> – Plus difficile à personnaliser – Pas un véritable environnement de navigation – Les bibliothèques ne sont pas portables
Approche	« Écrire une fois, fonctionne partout »	« Apprendre une fois, écrire n'importe où »
Plates-formes supportées	Android, iOS, Desktop, Progressive Web	Android, iOS
Éléments de l'interface utilisateur	Partagé	Non partagé
Taille de la communauté	Petite	Grande

Tableau 1 : Tableau récapitulatif de Ionic et React Native

3.5.2 Flutter contre react native

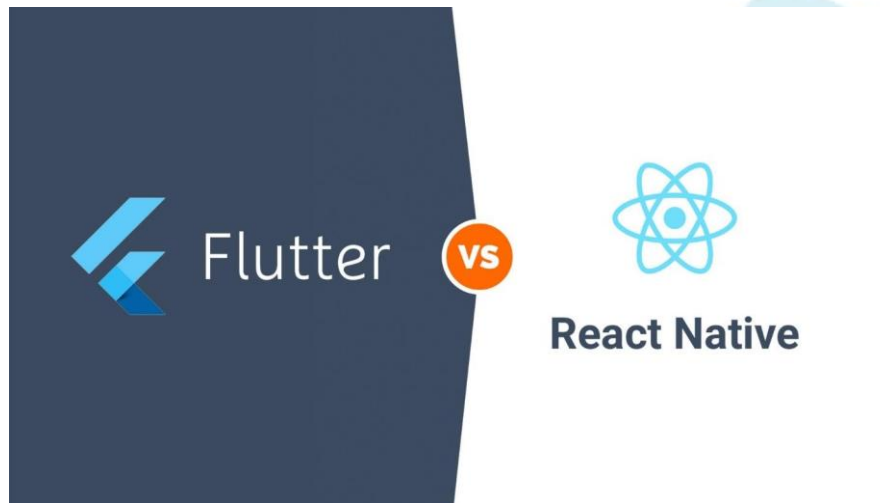


Figure 6: Logos de Flutter et react native

Flutter est un kit de développement logiciel relativement nouveau par rapport à React Native et n'a pas encore pénétré complètement le marché du développement d'applications. Il utilise le langage de programmation Dart au lieu du langage le plus répandu, JavaScript. React Native utilise JavaScript, ce qui le rend facile à utiliser pour la plupart des développeurs d'applications

➤ **FLUTTER :**

Il s'agit d'un kit de développement logiciel (SDK) gratuit permettant de créer des applications performantes pour iOS, Android, Mac, Windows, Google Fuchsia, Linux et le web à partir d'un seul code. Il s'agit d'un logiciel multiplateforme écrit en langage Dart et soutenu par Google.

➤ **Native :**

Il s'agit d'une structure JavaScript à code source ouvert pour le développement d'applications mobiles natives pour iOS et Android, soutenue par Facebook.

Parlons des similitudes

Ces structures vous aident à développer des applications mobiles multiplateformes pour iOS et Android

Les logiciels Flutter et React Native sont tous deux des codes sources ouverts et gratuits.

Vous pouvez être sûr que ces structures resteront en place pendant des décennies, car les plus grands géants de la technologie les soutiennent. Flutter est soutenu par Google, et React Native par Facebook.

Flutter et React Native supportent le hot reloading pour maintenir votre application en marche pendant que vous injectez des modifications au fichier.

Les deux structures offrent une excellente interface utilisateur pour le développement de belles applications mobiles.

Vous trouverez une documentation détaillée sur ces deux structures ainsi que des références sur les interfaces de programmation d'applications (API).

Parlons des différences

Flutter utilise le langage de programmation dart, tandis que React native utilise JavaScript pour développer ses applications.

Les emplois de Flutter et React Native

Flutter se classe en dessous de React Native en termes de disponibilité des emplois. Cela s'explique par le fait que Flutter est entré sur le marché du développement d'applications bien plus tard que React Native. De plus, en utilisant JavaScript et React, les outils de développement d'applications les plus utilisés, la structure est plus apte à créer de nombreuses opportunités d'emploi.

Les points de référence : Flutter contre React Native

Dans la comparaison des performances entre Flutter et React Native, Flutter est clairement gagnant car il n'utilise pas de pont JavaScript pour interagir avec les composants natifs. Le code Dart utilisé par Flutter a été compilé en code machine natif pour offrir des performances plus rapides.

Flutter vs. React Native, leur taille

Dans le monde des applications mobiles, il est très important de réduire la taille de l'application au minimum. Flutter a l'avantage sur React Native dans le développement

d'applications avec des tailles APK minimales. En effet, Flutter se compile à 100 % en code natif, alors que React Native communique à travers un pont en utilisant une combinaison de JavaScript et de code natif.

Avantages de Flutter

Il offre des performances plus rapides grâce à une communication directe avec les composants natifs.

Il est soutenu par Google.

Flutter offre des modifications rapides pour votre application avec sa recharge à chaud dynamique.

Il offre une personnalisation maximale grâce à son vaste ensemble de widgets.

Il est facile à mettre en place, et une machine bas de gamme peut facilement manipuler la structure.

Inconvénients de Flutter

Il est essentiel d'apprendre Dart avant de développer une application avec Flutter.

Comme la structure est relativement nouvelle, il manque des bibliothèques tierces.

Avantages de React Native

Il utilise le langage de programmation le plus couramment utilisé.

Il est soutenu par Facebook.

Il offre des modifications rapides grâce à son système de rechargement à chaud.

Le développement de l'interface utilisateur est fluide avec React Native.

Inconvénients de React Native

Il utilise un pont pour communiquer avec les composants natifs.

Il faut du temps pour que la synchronisation entre React Native et les nouveaux kits de développement logiciel soit complète.

La documentation disponible pour l'intégration de React Native avec des outils supplémentaires est très limitée.

Tableau comparatif 2020 Flutter versus React Native

Base	Flutter	React Native
Langage de programmation	Il utilise Dart.	Il utilise JavaScript.
Performance	Elle est plus rapide grâce à la communication directe avec les composants natifs.	Il est relativement plus lent en raison du pont JavaScript.
Taille	La taille minimale de l'APK des applications est plus petite.	La taille minimale de l'APK des applications est plus importante.
Opportunités d'emplois	Il y a moins de possibilités d'emploi pour les développeurs d'applications Flutter.	Plus d'offres d'emploi sont disponibles pour les développeurs d'applications React Native.
Soutien	Il est soutenu par Google.	Il est soutenu par Facebook.
Adoption	Il n'est pas aussi largement adopté que le React Native mais a été utilisé par Alibaba, Google Ads, Birch Finance, et plus encore.	Il a été largement adopté par de nombreux développeurs d'applications comme Bloomberg, Airbnb, Facebook Ads Manager, et bien d'autres.

Ensuite nous avons les modifications en live

Une des problématiques du développement est de devoir constamment déployer des modifications de code pour tester et mettre en production des changements sur votre application et cela peut parfois prendre beaucoup de temps.

C'est la raison pour laquelle les équipes de React Native ont mis en place un déploiement instantané qui permet aux développeurs, et aux équipes de recette, de visualiser directement les modifications apportées sur votre application.

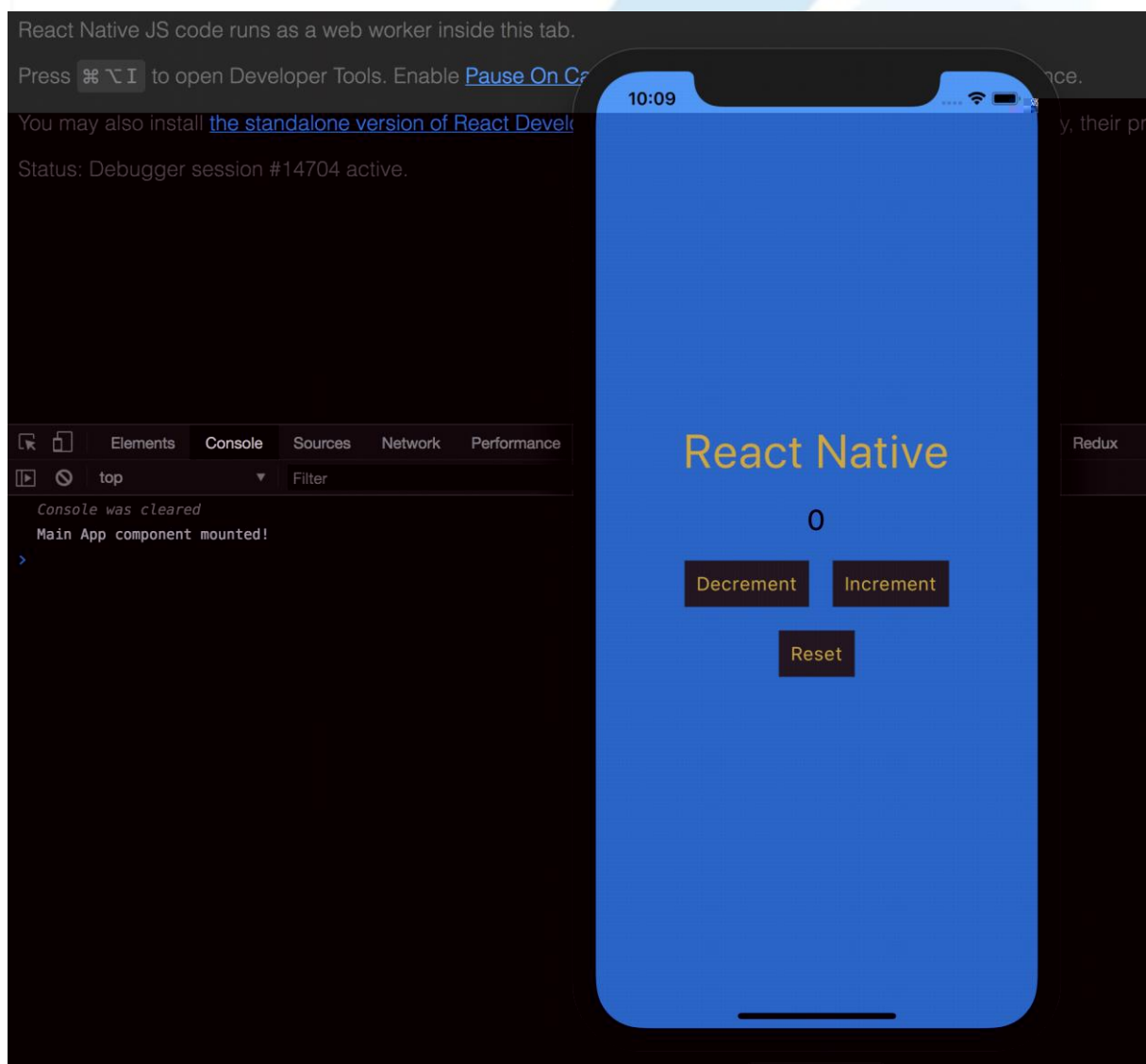


Figure 8 : Illustration de react native sur smartphone

Plus qu'un gain de temps, cette fonctionnalité permet aux équipes de développeurs d'être totalement transparents et de vous montrer en temps réel l'évolution de votre application. Ainsi, vos équipes peuvent faire les retours nécessaires pour corriger et améliorer votre application mobile quotidiennement.

Enfin grâce à sa facilité d'apprentissage

Voir le style de codage et vu que nous avons déjà des bases en JavaScript, HTML et CSS et de plus les documents officiels sont adaptés aux débutants et ont des exemples de code. Outre la documentation officielle.

Des exemples de plateformes pour apprendre

Le cours complet React Native + Hooks sur Udemy

Développement d'applications mobiles multiplateformes sur Coursera

Tutoriel React Native pour les débutants sur YouTube

Redux tutoriel vidéo

3.7 Choix du back-end

Tout comme les langages de programmation il existe une très grande panoplie de possibilités pour les SGBG.

Le backend, c'est toute la partie que l'utilisateur ne voit pas, mais qui lui permet de réaliser des actions sur un site ou une application

Dans la suite de notre travail nous verrons les différentes bases de données recommander par react native et expliquerons le pourquoi de notre choix.

3.7.1 *Cloud Firebase Firestore.*

En ce qui concerne la synchronisation des données, firebase fait partie des meilleurs. La base de données Firebase Realtime est une base de données hébergée dans le cloud. Elle prend en

charge une base de données NoSQL en temps réel pour les applications React Native et est suffisamment capable de répondre aux exigences MVC pour celles-ci.

Un autre avantage important c'est l'existence d'une API multiplateforme avec des exigences minimales pour la configuration. De plus, il est facilement accessible directement en tant que base de données en temps réel à partir d'un appareil mobile.



Figure 9 : Logo Firebase

3.7.2 *Mongodb*

MongoDB est une base de données côté serveur conçue pour les applications volumineuses et complexes. Son approche principale consiste à utiliser un magasin clé-valeur et une base de données relationnelle pour stocker des objets dans des documents JSON avec des schémas dynamiques. Cette base de données offre une solution précieuse pour les applications mobiles React Native évolutives.



Figure 10 : Logo mongodb

3.7.3 AWS DynamoDB

Dynamo DB est une base de données côté serveur NoSQL conçue pour s'exécuter sur la plateforme cloud Amazon Web Services. Il est complètement décentralisé et ne nécessite pas beaucoup d'administration

Son modèle d'interrogation est faible, l'interrogation des données est extrêmement limitée.



Figure 11 : Logo dynamodb

3.7.4 SQLite

Cette base de données a été initialement conçue pour offrir un stockage local à diverses applications des smartphones. Il possède une bibliothèque légère et ne nécessite que peu d'efforts pour être mis en place. Un avantage majeur de cette base de données est qu'elle peut être utilisée directement dans l'application mobile pour accéder directement à la base de données.

L'architecture de la bibliothèque est simple et tous les types de données peuvent être manipulés facilement. Un autre point intéressant est que SQLite peut activer la persistance hors ligne via les développeurs.

En outre, il permet aux développeurs d'utiliser un plugin de stockage basé sur React Native qui s'exécute sur SQLite. Son travail principal est de gérer les données avec l'application.

Figure 12 : Logo SQLite

3.7.5 *Le choix de notre base de données*

Les quatre SGBD étudiés plus haut présentent de nombreux avantages pour notre application. Mais notre choix est porté sur Firebase car firebase est un moteur en temps réel avec une connectivité en arrière-plan qui prend en charge tout un écosystème pour le développement d'applications mobiles et Web. Google possède actuellement Firebase et a créé une solution beaucoup plus complète avec de nombreux autres

applications multiplateformes avec les SDK iOS, Android et JavaScript, tous vos clients partagent une instance de base de données en temps réel et reçoivent automatiquement des mises à jour avec les données les plus récentes.

Capacités clés	Description
Temps réel	Au lieu des requêtes HTTP classiques, la base de données en temps réel Firebase utilise la synchronisation des données : chaque fois que les données changent, tout appareil connecté reçoit cette mise à jour en quelques millisecondes. Offrez des expériences collaboratives et immersives sans penser au code de mise en réseau.
Hors ligne	Les applications Firebase restent réactives même lorsqu'elles sont hors ligne, car le SDK Firebase Realtime Database conserve vos données sur le disque. Une fois la connectivité rétablie, le périphérique client reçoit tous les changements qu'il a manqués, en les synchronisant avec l'état actuel du serveur.
Accessible depuis les appareils des clients	La base de données en temps réel Firebase est accessible directement à partir d'un appareil mobile ou d'un navigateur Web ; il n'y a pas besoin d'un serveur d'applications. La sécurité et la validation des données sont disponibles via les règles de sécurité de la base de données en temps réel Firebase, des règles basées sur des expressions qui sont exécutées lorsque les données sont lues ou écrites.

Évolutivité sur plusieurs bases de données	Avec Firebase Realtime Database sur le plan tarifaire Blaze, vous pouvez répondre aux besoins de données de votre application à grande échelle en répartissant vos données sur plusieurs instances de base de données dans le même projet Firebase. Rationalisez l'authentification avec Firebase Authentication sur votre projet et authentifiez les utilisateurs sur vos instances de base de données. Contrôlez l'accès aux données de chaque base de données avec des règles de base de données en temps réel Firebase personnalisées pour chaque instance de base de données.
--	--

Tableau 2 : Tableau recapitulatif de firebase

3.8 L'architecture de EcoVoiturage

Une architecture d'application décrit les modèles et les techniques utilisés pour concevoir et créer une application. L'architecture fournit une feuille de route ainsi que les meilleures pratiques à suivre pour créer une application bien structurée.

Une architecture d'application comprend les services front-end et back-end. Le développement front-end concerne l'expérience utilisateur de l'application, tandis que le développement back-end permet de fournir l'accès aux données, aux services et à d'autres systèmes dont dépend l'application.

L'architecture est le point de départ ou la feuille de route pour créer une application. Vous devrez cependant effectuer les choix de mise en œuvre que l'architecture n'inclut pas. Par exemple, le choix du langage de programmation est la première étape pour écrire une application.

Il existe de nombreux langages de programmation pour le développement logiciel. Certains langages sont utilisés pour créer certains types d'applications, par exemple Swift pour les applications mobiles ou JavaScript pour le développement front-end.

JavaScript, associé aux langages HTML et CSS, figure actuellement parmi les langages de programmation pour notre travail

D'autres langages de programmation sont aussi largement utilisés : Ruby, Python, Swift, TypeScript, Java, PHP et SQL. Le choix du langage dépend du type d'application à créer, des ressources de développement disponibles ainsi que d'autres facteurs.

Auparavant, les applications étaient rédigées sous la forme d'une seule unité de code, où tous les composants partageaient les mêmes ressources et le même espace mémoire. C'est ce que l'on appelle une architecture de type monolithique.

Notre application repose sur une architecture de type client-serveur, où le client (frontend) représente l'entité chargée d'interagir directement avec l'utilisateur final et le serveur (backend) représente l'entité chargée du traitement et du stockage des données. La communication entre le backend et le frontend s'effectue grâce au protocole HTTP.

Le backend comprend un « serveur d'applications » et un « serveur de base de données ». Le serveur d'applications est composé d'un « serveur web » (dont le rôle principal est de recevoir et de répondre à des requêtes HTTP) et d'un « interpréteur » ou « moteur de scripts » (qui permet d'exécuter des programmes écrits dans un langage de programmation).

Le serveur de base de données quant à lui sert de conteneur au Système de Gestion de Base de Données (SGBD) qui n'est rien d'autre qu'un logiciel système servant à stocker, à manipuler ou gérer, et à partager des informations dans une base de données, en garantissant la qualité, la pérennité et la confidentialité des informations, tout en cachant la complexité des opérations.

Le frontend est représenté par l'application cliente qui tourne sur l'OS du mobile. Il gère principalement toutes les interactions avec l'utilisateur et est chargé de lui afficher les informations dont il a besoin. Il communique régulièrement avec le backend pour ajouter, modifier, récupérer ou supprimer des données.

L'architecture logicielle implique aussi la définition d'une solution structurée qui répond à toutes les exigences techniques et opérationnelles, tout en optimisant les attributs de qualité courants comme la performance, la sécurité

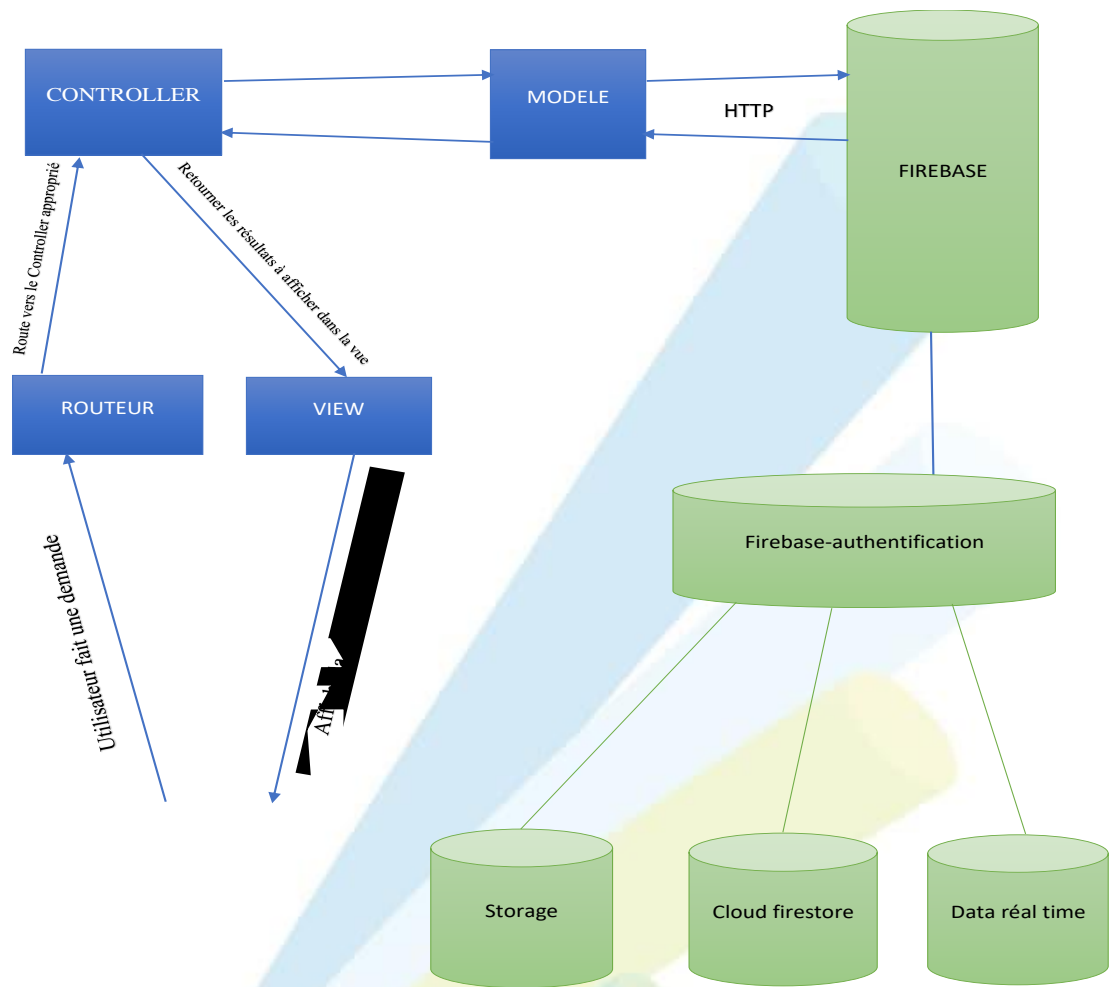


Figure 13 : Bsdj jufdws fle flmbqq nxbujpo

3.9 Mise en place de l'environnement de développement

Avant avoir à entamer la conception de l'application, nous avons l'obligation dans un premier temps mettre en place l'environnement de travail. Il nous permettra de pouvoir développer chaque partie notre application. Il s'agira de créer un nouveau projet React Native dans lequel se trouve des fichiers de base qui sont indispensables pour démarrer tout projet. Dans les écritures à suivre nous détaillerons tout dans les moindres détails.

3.9.1 *Création et structuration de base du projet de React Native*

Ce dossier comporte différents fichiers et sous-dossiers essentiels pour démarrer notre application. Parmi ces fichiers et sous-dossiers, on a :

3.9.2 *Ajout de fichier complémentaire au projet*

Il est question ici d'ajouter des fichiers manuellement qui seront utilisés. Même si le dossier de base contient déjà des fichiers par défaut, il est possible d'en rajouter au besoin sans pour autant modifier ceux qui existent déjà. Le capture d'écran ci-dessous montre une partie d'organisation de notre application.

3.10 Outils de développement

3.10.1 *Visual Studio Code*

Visual Studio Code est un éditeur de code source développé par Microsoft pour Windows, Linux et MacOS. Il inclut la prise en charge du débogage, du contrôle git intégrer, de la coloration syntaxique, de la complétion du code intelligent, des extraits et du refactoring du code. Il est également personnalisable afin que les utilisateurs puissent modifier le thème de l'éditeur, les raccourcis clavier et les préférences. Il est gratuit et open-source, bien que le téléchargement officiel soit sous une licence propriétaire.

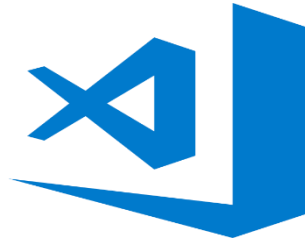


Figure 14 : Logo Visual Studio Code

3.10.2 GitHub

GitHub est un service web d'hébergement et de gestion de développement de logiciels. Ainsi, grâce à GitHub, n'importe quel développeur qui a accès à un code peut avoir les différentes versions et surtout l'historique des modifications facilement.

Cette possibilité offre aux programmeurs l'opportunité de travailler sur une copie du code et de pouvoir la réintégrer ensuite après validation des modifications.



Figure 15 : Logo Github

3.10.3 MODELIO

Modelio est un outil de modélisation complet UML sans limite et gratuit, il n'impose aucune limitation de taille de modèle.



Figure 16 : Logo Modelio

3.11 Sécurité

L'expression Sécurité des applications décrit les mesures de sécurité au niveau des applications qui aident à empêcher le vol ou le détournement de données ou de code contenus dans les applications.

La sécurité des applications est importante, car les applications d'aujourd'hui sont souvent disponibles sur divers réseaux et connectées au Cloud, ce qui augmente leur vulnérabilité aux menaces et aux violations de sécurité. On constate de plus en plus de pression et d'incitations à ne pas uniquement garantir la sécurité au niveau du réseau, mais également au sein des applications elles-mêmes. L'une des raisons à cela est le fait que les hackers s'attaquent davantage aux applications que par le passé. Les tests de sécurité des applications peuvent révéler les défauts au niveau des applications, afin d'empêcher ces attaques.

La sécurisation d'une application ou d'un système s'attache à 6 aspects primordiaux :

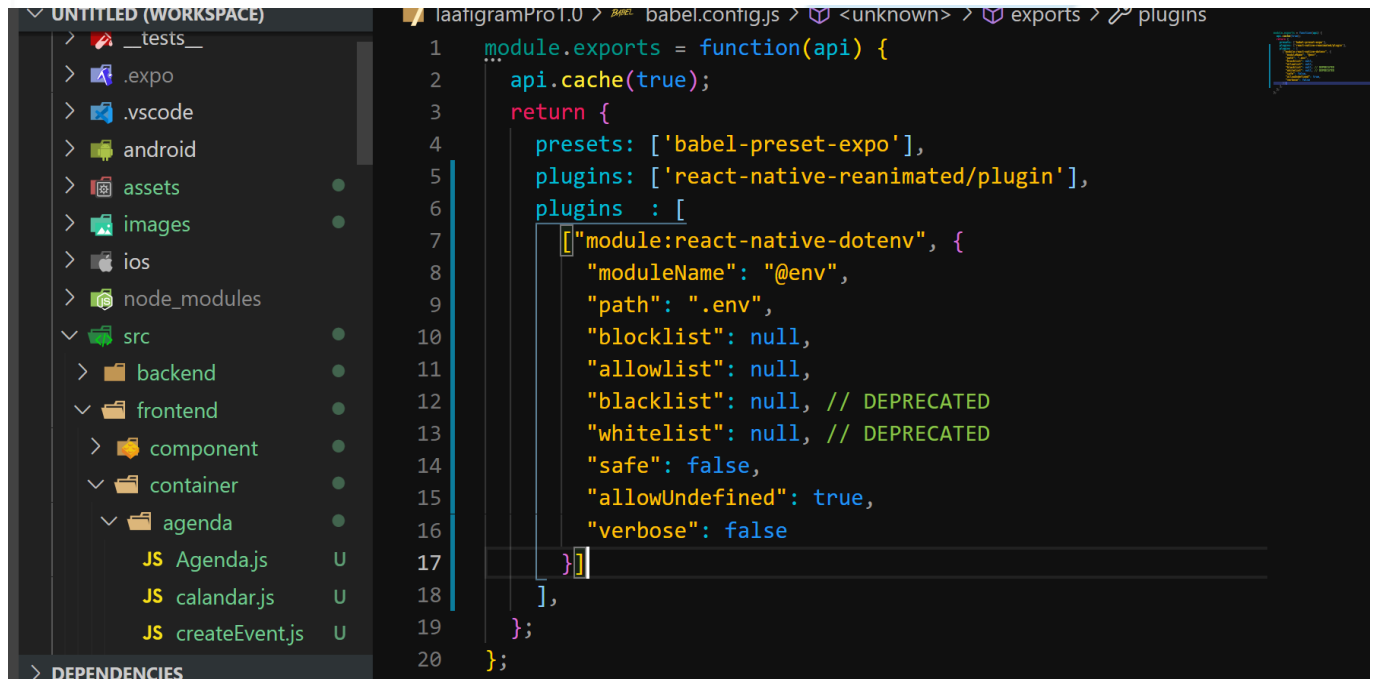
- L'authentification ;
- Le contrôle d'accès ;
- L'intégrité des données ;
- La confidentialité des données ;
- La non-répudiation ;
- La protection contre l'analyse du trafic

Il est impossible de nos jours de créer un logiciel 100% sécurisé. La probabilité d'être victime d'une attaque ou d'être exposé à une faille de n'est pas une chose à prendre à la légère.

Créer une application mobile ne consiste pas seulement à créer des écrans, à créer des animations sophistiquées et à traiter des données à partir d'une API ou d'une base de données. Il est plus complexe que cela. L'un des plus grands défis du processus de développement est la sécurité. En outre, c'est l'un des aspects les plus souvent négligés lors de la création de logiciels. Pour la sécurité de notre application nous allons utiliser l'une des bibliothèques populaires, comme react-native-dotenv pour sécuriser notre application

- React-native-dotenv est un module que nous utiliserons pour accéder aux variables d'environnement dans notre application. Lorsqu'une application **react native** s'exécute, elle injecte une variable globale appelée **process.env** qui contient des

informations sur l'état de l'environnement dans lequel l'application s'exécute. **Dotenv** nous permettra de charger nos variables d'environnement stockées dans le fichier **.env** dans **process.env**



```
1 module.exports = function(api) {
2   api.cache(true);
3   return {
4     presets: ['babel-preset-expo'],
5     plugins: ['react-native-reanimated/plugin'],
6     plugins : [
7       ["module:react-native-dotenv", {
8         "moduleName": "@env",
9         "path": ".env",
10        "blocklist": null,
11        "allowlist": null,
12        "blacklist": null, // DEPRECATED
13        "whitelist": null, // DEPRECATED
14        "safe": false,
15        "allowUndefined": true,
16        "verbose": false
17      }],
18    ],
19  };
20 };
```

Capture 1 : Image montrant les plugins de react-native-dotenv

- -Empêcher l'accès pour les appareils racines

Il existe de nombreuses raisons pour lesquelles les personnes routent leurs téléphones. Etant donné que notre application fonctionne avec des données très sensibles, nous envisageons une protection par identification, que l'appareil soit enraciné ou non. Ces types d'appareils peuvent obtenir un accès non autorisé aux données stockées dans notre application.

Pour protéger notre application, nous allons utiliser une bibliothèque appelée jail-monkey. Cette bibliothèque nous permet de :

- -Identifier si un téléphone a été jail-monkey ou rooté pour iOS/Android ;
- -Détecter si l'appareil simule sa position GPS ;


- -Détecter si l'application s'exécute sur un stockage externe (Android uniquement).

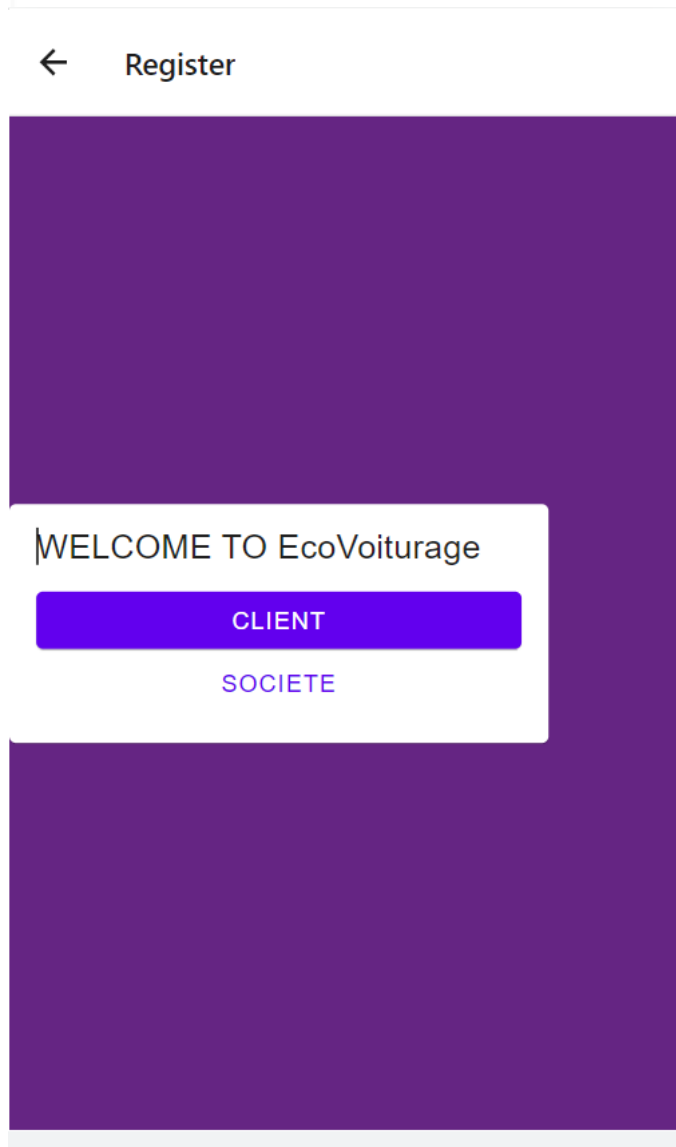
```
5 "react-native-date-picker": "^3.3.2",  
7 "react-native-datepicker": "^1.7.2",  
8 "react-native-dotenv": "^3.2.0",  
9 "jail-monkey": "^2.6.0",  
0 "react-native-jailbreak": "^0.1.0",  
1 "react-native-elements": "^3.4.2",  
2 "react-native-events-calendar": "^1.0.8"
```

Capture 2 : Capture montrant la bibliothèque dotent, jailbreaké et jail-monkey

3.12 Interface de EcoVoiturage

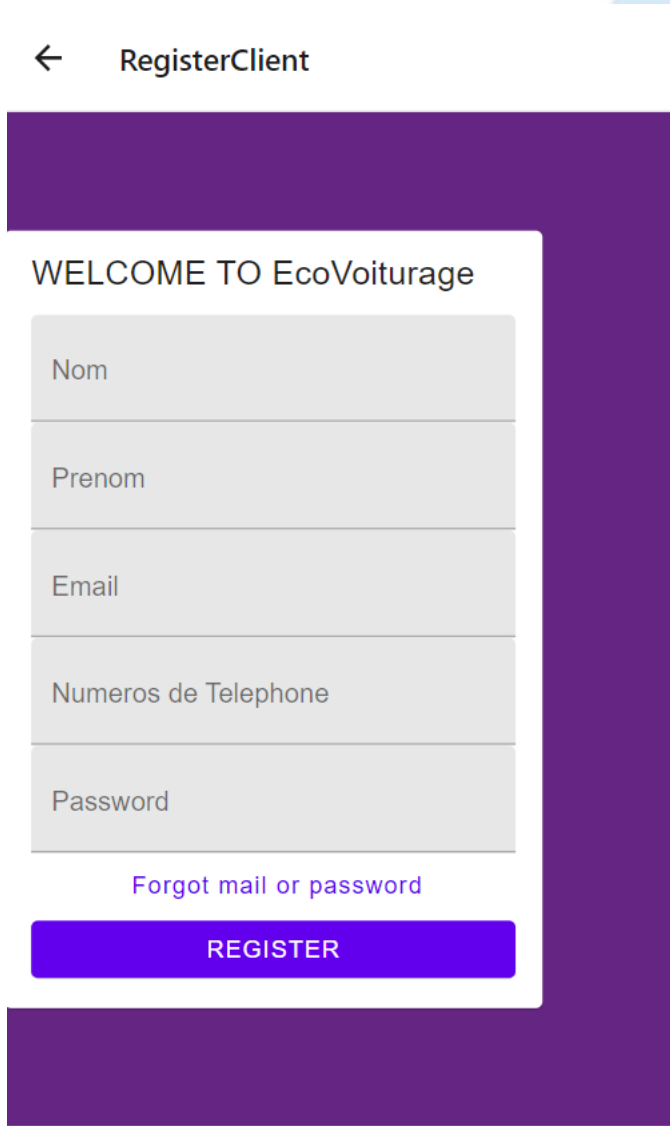
Les images ci-dessous illustrent les interfaces de quelques fonctionnalités de EcoVoiturage

 Interface de choix de statut



Capture 3 : Page de choix de statut

Interface d'inscription



The screenshot shows a mobile application interface for client registration. At the top, there is a back arrow and the text "RegisterClient". Below this is a purple header bar. The main content area is white and contains the text "WELCOME TO EcoVoiturage". Underneath, there are five input fields for "Nom", "Prenom", "Email", "Numeros de Telephone", and "Password". A link "Forgot mail or password" is positioned below the password field. At the bottom of the form is a blue "REGISTER" button. The entire form is set against a background of a purple L-shaped bar and a light blue background with abstract green and blue curved shapes.

← RegisterClient

WELCOME TO EcoVoiturage

Nom

Prenom

Email

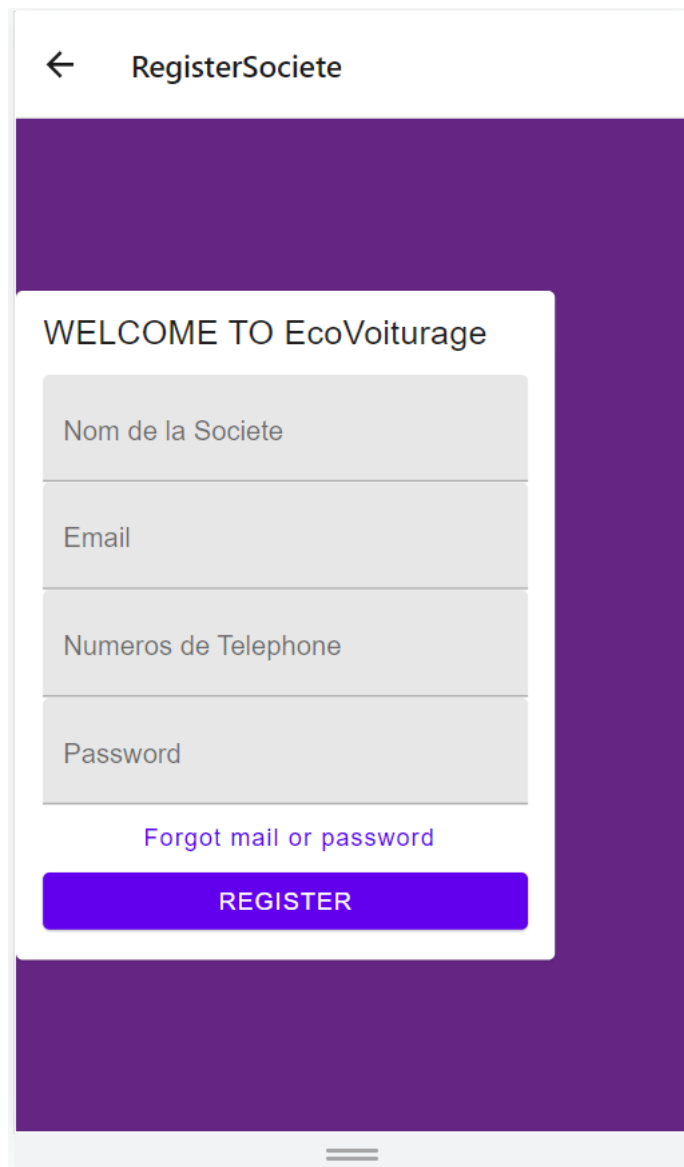
Numeros de Telephone

Password

[Forgot mail or password](#)

REGISTER

Capture 4 : Qbh fle pt ds q uipo!ev!dmjfou



The image shows a mobile application interface for 'RegisterSociete'. At the top, there is a back arrow and the title 'RegisterSociete'. Below this is a large purple header. A white card in the center contains the text 'WELCOME TO EcoVoiturage'. Underneath the card are four input fields: 'Nom de la Societe', 'Email', 'Numeros de Telephone', and 'Password'. Below the input fields is a link that says 'Forgot mail or password' in purple. At the bottom of the card is a blue button labeled 'REGISTER'. The bottom of the screen features a grey bar with a hamburger menu icon.

← RegisterSociete

WELCOME TO EcoVoiturage

Nom de la Societe

Email

Numeros de Telephone

Password

[Forgot mail or password](#)

REGISTER

Capture 5 : Qbh fle jpt ds ij ujppl fo!iboulrv flt pld jfuji

EcoVoiturage



ouedraogoluc@gmail.com



medina plateau
Reserver



demo@gmail.com



medina plateau
Reserver



samata@yahoo.com



dakar thies
Reserver



samata@yahoo.com



dakar thies
Reserver



Home



rechercher

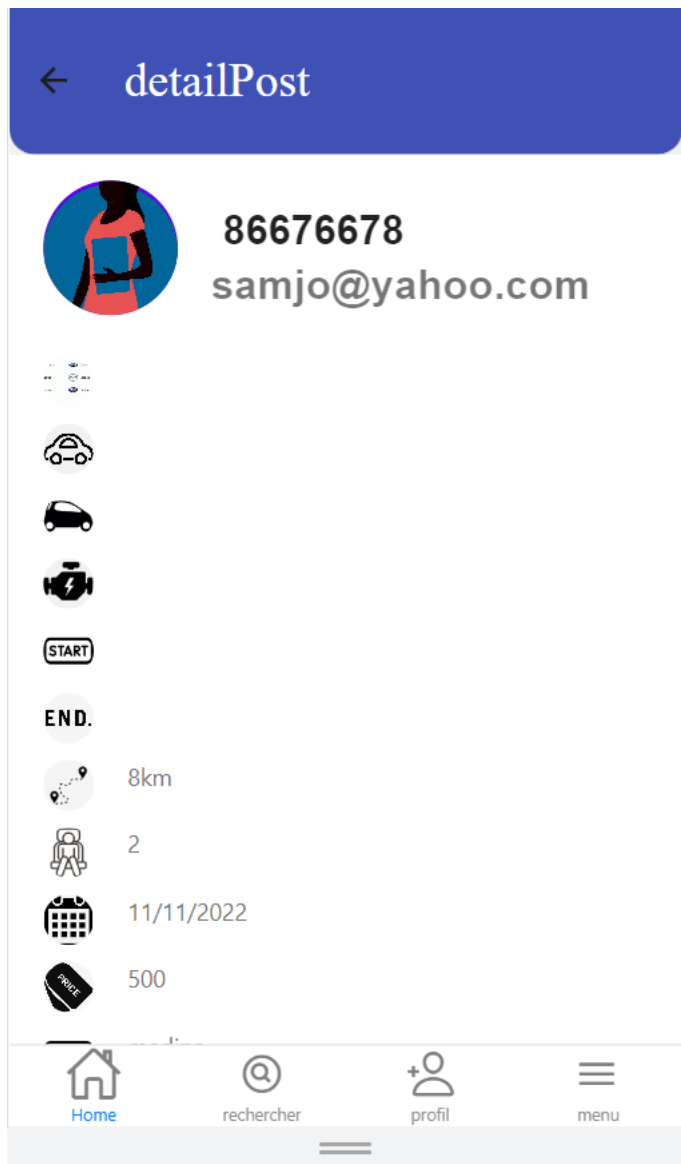


profil

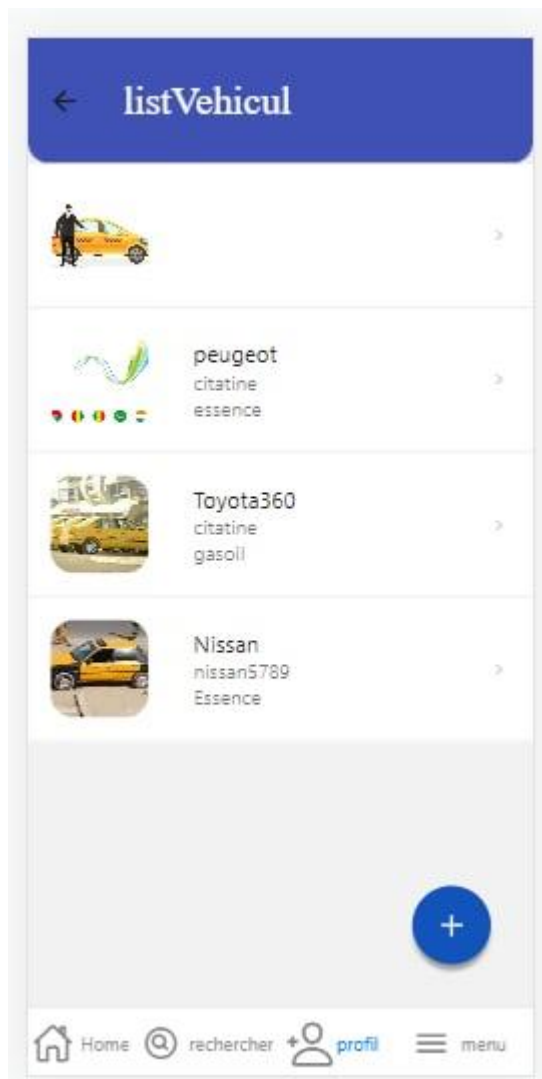


menu

Capture 6 : Page de publication des différentes offres de covoiturages



Capture 7 : E fib jnle volq p u



Capture 8 : L'ajout de véhicules dans la liste

←

AddPost

INFO CHAUFFEUR

nom complet du chauffeur

Taper message

Numeros du chauffeur

Taper message

Categorie Permis

Taper message

VEHICUL

Marque vehicule

Taper message

Modele vehicule

Taper message

Categorie Vehicule

Taper message

Motorisation Vehicule

Taper message

TRAJET

Save ✓

home

rechercher

profil

menu

Capture 9 : E fib jtnle ft ! jgpsn bujpot !e bkpvule f'wpjws fle vo f't pd jfuft

←

AddPost

TRAJET

date

Taper message

L'heure

Taper message

Distance

Taper message

Nombre de Place

Taper message

Numeros

Taper message

Prix

Taper message

Lieu de depart

Taper message

Lieu d'arrivee

Taper message

Save ✓

Home

rechercher

profil

menu

Capture 10 : v jf!e ft!e fib jn!e ft! jgpsn bujpot!e bkpvile f!wpjws f!e vo ft pd jfu

Partie 4 : APPORTS

Ce projet de fin d'études nous a permis de nous familiariser avec un certain nombre de concepts tout en se basant sur nos connaissances acquises au cours de notre formation au sein de notre école. Notre projet de fin d'étude a été une occasion de développer et d'exercer nos capacités d'observation, d'analyse, de conception, de développement et de rédaction.

4.1 Apports sur le plan technique

Ce projet de fin d'étude nous a permis de :

- ✚ Mettre en œuvre les notions et les connaissances acquises au sein des modules « Méthodologie de conception des bases de données » ;
- ✚ Découvrir les technologies NoSql et SQL (Firebase et PostgreSQL) ;
- ✚ Découvrir la librairie ReactJS ;
- ✚ Améliorer nos connaissances théoriques sur la communication client/serveur ;
- ✚ Apprendre à gérer un projet.

4.2 Apports au niveau de la conception et du développement

Au niveau de la conception et du développement, cette étude nous a permis de :

- ✚ Mener une conception orientée objet représentée avec le langage UML ;
- ✚ Apprendre à maîtriser la méthodologie de conception et de développement des applications clientes ;
- ✚ Maîtriser les étapes de développement et de conception d'une application mobile ;

Partie 5 : CONCLUSION ET PERSPECTIVES

Ce projet nous a permis de mettre en évidence toute la complexité qui se cache derrière une application mobile. En partant de l'analyse et la conception, en passant par le choix des architectures et modèles de conception à utiliser jusqu'à la réalisation concrète de l'application. Le processus peut s'avérer compliqué à gérer surtout lorsqu'on ne démarre pas sur de bonnes bases. Heureusement, grâce à l'ESMT qui nous a donné une bonne formation, et à la vaste documentation et les outils mis en avant par la communauté de développeurs, il a été plus aisé de se retrouver dans ce long processus.

Notre projet « Conception et réalisation d'une application mobile de covoiturage » nous a donné des résultats satisfaisants et testables.

Notre démarche a été en un premier lieu d'effectuer une étude sur la mise en place de l'application, puis en deuxième lieu de faire la conception. En troisième lieu nous avons procédé aux choix des technologies et à la réalisation ainsi que la conception de l'API. Enfin, nous avons parlé de l'apport du projet.

La majorité des fonctionnalités décrites dans la présentation du sujet ont été développées et validées. Néanmoins, notre projet pourra être amélioré via une étude des méthodes de collaboration et de communication entre les différents concerner ; dévoiler les défaillances rencontrées durant l'utilisation en collaborant avec les utilisateurs de application d'EcoVoiturage; constater des anomalies au niveau de la communication, la gestion des informations pour assurer plus de sécurité aux utilisateurs ;créer une bibliothèque dans notre application et stocker les informations de notre API.

Nous n'avons pas implémenté certaines fonctionnalités comme la localisation, le paiement en ligne

BIBLIOGRAPHIE

[B1] : Définition des diagrammes cours de M. Ghislain AKINOCHO - Esmt, 2019 0 -
Chapitre 1 – Introduction à UML slide 38

[B2] : documentation et informations sur l'expériences des taximan à dakar : Monsieur Fall et son équipe

[B3] : *React native contre Ionic en 2017*

*npm install de React-Native et Ionic-Angular en 2017 (source: **NPM Trends**)*

[B3] : Mémoire de Charbel ZINGAN

WEBOGRAPHIE

Documentation UML consulté le 12/07/2021

[W1] : <https://www.ibm.com/docs/fr/rsas/7.5.0?topic=model-modeling-user-workflow-by-using-activity-diagrams> consulté le 20/06/2022 à 12H

[W2] : <https://cordova.apache.org>

Documentation sur Ionic consulté le 20/06/2022

[W3] : <https://ionicframework.com>

[W4] : Nombre de commits sur le projet ionic sur github (source: [github](#))

React native contre Flutter en 2022

[W5] : <https://www.monterail.com/blog/flutter-vs-react-native-mobile-development>

Comparaison des frameworks cross-platform

[W6] : <https://www.ftxinfotech.com/react-native-vs-ionic-vs-flutter>

Meilleurs SGBD pour react native

[W7] : <https://blog.back4app.com/fr/les-10-meilleures-bases-de-donnees-pour-votre-application-react-native/>

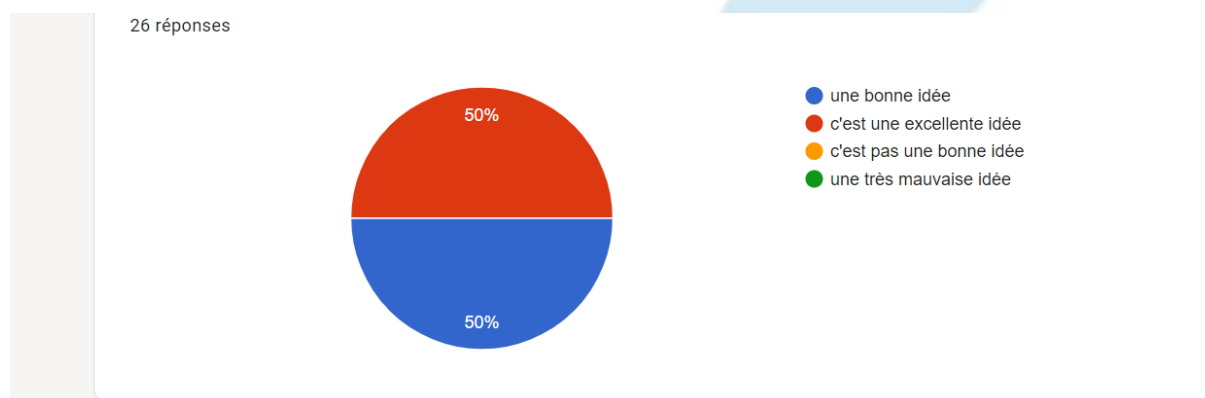
[W8] : <https://webdevdesigner.com/q/what-are-my-options-for-storing-data-when-using-react-native-ios-and-android-30399/>

Introduction à Flask Consulté le 8/12/2022

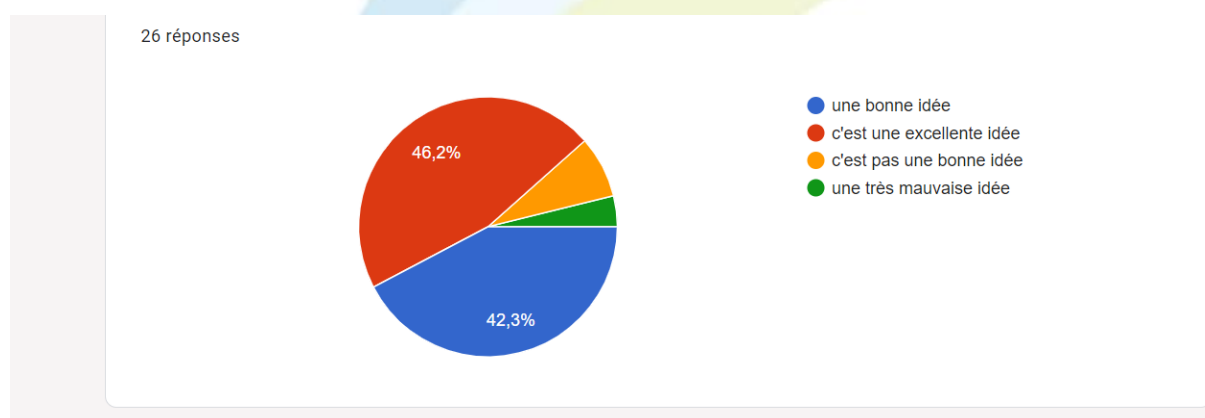
ANNEXE A

Les images ci-dessous illustrent les réponses lors des enquêtes pour la mise en place de l'application.

Que pensez de l'application EcoVoiturage après avoir lu le petit résumé ?



Pensez-vous que cette action est une bonne idée pour l'environnement ?



Cette application vous permettez de faire des économies sur votre budget transport ?

27 réponses

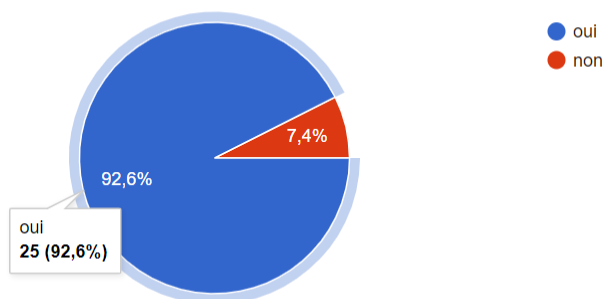


Table des matières

DEDICACE.....	II
REMERCIEMENTS.....	II
LISTE DES CAPTURES	III
LISTE DES FIGURES.....	IV
SIGLES ET ABREVIATIONS	V
LISTE DES TABLEAUX.....	VI
AVANT-PROPOS	VII
Sommaire	VIII
INTRODUCTION.....	1
Partie 1 : ETUDE PREALABLE SPECIFIQUE ET ANALYSE DES BESOINS	2
1.1 Contexte du sujet.....	2
1.2 Problématique du sujet.....	3
1.3 Solutions à la problématique.....	3
1.4 Etat de l’art.....	3
1.5 Cahier de charge.....	4
1.6 Aspect économique	7
Partie 2 : LA MODELISATION STRUCTURELLE ET COMPORTEMENTALE.....	9
2.1 Choix de la technologie de modélisation	9
2.2 Présentation de l’UML.....	9
2.3 Diagramme des cas d’utilisations (DCU)	11
2.4 Diagramme de classe	14
Partie 3 : LA REALISATION ET CONCEPTION	15
3.1 Les technologies mobiles.....	15
3.2 Applications natives.....	15
3.3 Application web ou connectés	16
3.4 Les applications mobiles hybrides	16
3.5 Concepts sur les technologies multiplateformes.....	16
3.6 Choix de technologie de réalisation	27
3.7 Choix du back-end	29
3.8 L’architecture de EcoVoiturage	34
3.9 Mise en place de l’environnement de développement	37
3.10 Outils de développement.....	37

3.11 Sécurité	39
3.12 Interface de EcoVoiturage	42
Partie 4 : APPORTS	50
4.1 Apports sur le plan technique.....	50
4.2 Apports au niveau de la conception et du développement.....	50
Partie 5 : CONCLUSION ET PERSPECTIVES	51
BIBLIOGRAPHIE	A
WEBOGRAPHIE.....	B
ANNEXE A.....	C



MÉMOIRE FIN DE FORMATION POUR L'OBTENTION DU DIPLOME DE LICENCE PROFESSIONNELLE

OPTION : TELECOMMUNICATIONS ET INFORMATIQUE
SPÉCIALITÉ : DEVELOPPEMENT D'APPLICATIONS REPARTIES

Auteur : Mme ZANNE B-W Melissa Diane

Thème : Conception et réalisation d'une application mobile de covoiturage

Directeur de Mémoire : M. Moustapha DER

RESUME :

Le thème abordé dans ce projet de fin d'études consiste à concevoir et à développer une application mobile de covoiturage. L'application réalisée prend en charge les tâches suivantes les publications, les échanges d'information (messagerie), réservation. Ce qui a été prévu et qui n'a pas pu être réalisée concerne la localisation, le paiement en ligne.

La méthode de conception utilisée est UML, en ce qui concerne les technologies de développement, nous avons opté React Native pour les interfaces de l'application, firebase pour la gestion des données de l'application, FLask pour la conception de l'API, et PostgreSQL comme base de données.



MÉMOIRE FIN DE FORMATION POUR L'OBTENTION DU DIPLOME DE LICENCE PROFESSIONNELLE

OPTION : TELECOMMUNICATIONS ET INFORMATIQUE
SPÉCIALITÉ : DEVELOPPEMENT D'APPLICATIONS REPARTIES

Auteur : Mme ZANNE B-W Melissa Diane

Thème : Conception et réalisation d'une application mobile de covoiturage

Directeur de Mémoire : M. Moustapha DER

SUMMARY:

The theme of this graduation project is to design and develop a mobile carpooling application. The application carried out supports the following tasks: publications, information exchange (messaging), reservation. What was planned and which could not be carried out concerns localization, online payment.

The design method used is UML, as far as development technologies are concerned, we have opted for React Native for the application interfaces, firebase for the application data management, Flask for the API design, and PostgreSQL as the database.