École pluridisciplinaire internationale

EPI Digital school
Rapport

Conception et réalisation d'une application mobile e-constat

réalisé par :
Baccouche Siwar

Enseignant :
Mr. Chtioui Houssem

Année universitaire: 2023-2024

Dédicace

Je dédie ce travail à mes parents .

Remerciements

J'adresse mes sincères remerciements à Mr.Chtioui pour son encadrement et son soutien inestimable tout au long de ce projet.

Table des matières

Table des figures

Introduction

L'industrie automobile est caractérisée par une évolution continue redéfinissant en permanence le marché. Cette dynamique exige que la procédure de déclaration d'accidents soit transparente, fluide et efficace, influencé par de nombreux facteurs, notamment la disponibilité des parties impliquées, les assurances, les documents et bien d'autres encore. C'est dans ce contexte que le projet AssurTN est né. En tant qu'étudiante en ingénierie

en génie logiciel, nous avons entamé la création d'une application mobile innovante visant à simplifier le processus de constatation des accidents. Cette application sera un outil pratique et accessible à tous, mettant en œuvre des technologies en développement frontend et backend pour fournir une méthode de déclaration plus fiable et pratique. En somme,

le projet AssurTN incarne notre engagement envers l'amélioration de l'expérience des conducteurs. Nous espérons que ce rapport vous permettra de comprendre en profondeur la création de cette application innovante. Le présent rapport abordera donc les différentes phases de la genèse et la réalisation de ce projet. Il compte quatre principaux chapitres décrits brièvement comme suit :

- Le premier chapitre présente le cadre du projet, l'étude préliminaire et la méthodologie de travail utilisée.
- Le deuxième chapitre contient l'analyse et la spécification des exigences fonctionnelles et non fonctionnelles, détaillées à travers des diagrammes UML.
- Le troisième chapitre élabore une description de l'architecture du système et une conception détaillée à travers un diagramme de classes d'entités, des diagrammes de classes participantes et des diagrammes de séquences.
- Le quatrième chapitre décrit la réalisation de notre application en présentant l'environnement logiciel et les choix technologiques ainsi que les différentes interfaces réalisées.

Finalement nous concluons ce rapport par une conclusion générale du travail accompli suivie des perspectives de notre projet.

Chapitre 1

Aperçu général et cadre du projet

Intoduction

Dans ce chapitre nous présenterons en premier lieu, l'entreprise d'accueil au sein de laquelle ce projet a été réalisé. Ensuite nous metterons en lumière la problématique au centre de notre projet, en examinant les solutions existantes, en comparant les méthodes agiles, et en sélectionnant la méthodologie adoptée. De plus, nous procéderons à l'élaboration du backlog du produit et à la création du diagramme de Gantt.

1.1 Contexte du projet

1.1.1 Cadre du projet

Ce projet représente une excellente opportunité pour enrichir nos connaissances académiques et acquérir une précieuse expérience concrète dans le domaine de l'informatique. En effet, il s'inscrit dans le cadre de la validation du projet de fin d'année du quatrième semestre pour l'obtention du diplôme national d'ingénieur en Génie Logiciel à l'EPI (École Pluridisciplinaire Internationale). Ce projet s'est déroulé au sein de notre école, l'EPI, et prend la forme d'une application mobile intitulée "AssurTN", développée pour les plateformes iOS et Android.

1.1.2 Présentation de la société d'acceuil : EPI

"EPI SUP" est un groupe universitaire pluridisciplinaire et international qui a été lancé en 2011, comprenant aujourd'hui quatre écoles. Parmi ces écoles, nous présentons l'École Digitale (EPI-Digital School) dédiée à la formation des professionnels du monde du digital (licences, masters, ingénieurs), au sein de laquelle nous développons notre application.

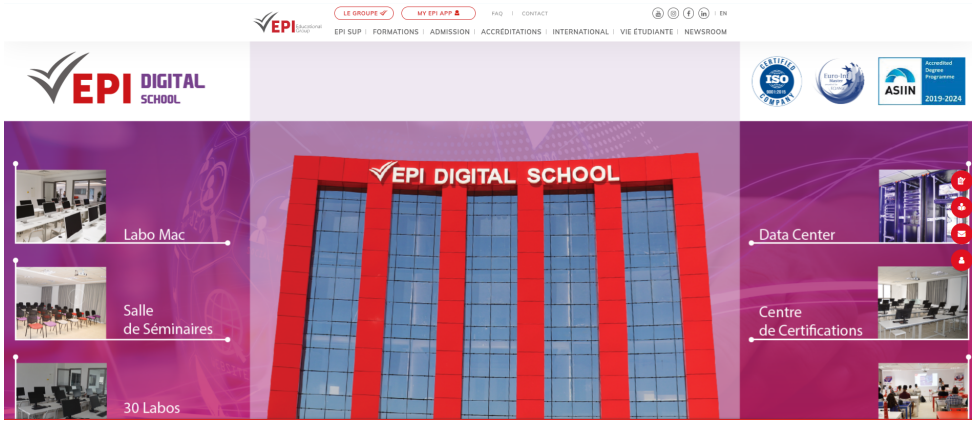


FIGURE 1.1 – Site web de l'EPI

1.2 Présentation du projet

1.2.1 Problématique

L'industrie automobile est marquée par une évolution constante, rendant primordiale la modernisation des procédures qui lui sont reliées, telles que la déclaration d'accidents matériels.
Traditionnellement, un constat amiable est un document papier à remplir, expliquant les circonstances d'un accident, utilisé par les assurances pour déterminer les responsabilités et les indemnisations des parties impliquées. Cette méthode peut parfois être fastidieuse et peu efficace, ce qui rend les exigences en termes de rapidité et de fiabilité dans le traitement des sinistres de plus en plus pressantes. Notre problématique découle de cette réalité : Comment simplifier, accélérer et moderniser le processus de déclaration d'accidents automobiles ? Comment s'inspirer d'un constat amiable papier pour créer une application plus conviviale améliorant l'expérience utilisateur ?
Dans ce cadre, nous sommes confrontés à la nécessité de concilier les exigences technologiques modernes avec les besoins pratiques des utilisateurs. Notre objectif est de proposer une solution innovante qui simplifie et accélère le processus de gestion des sinistres, tout en renforçant la confiance des assurés dans leur compagnie d'assurance.

1.2.2 Etude de l'existant

L'étude de l'existant joue un rôle crucial dans le développement de notre projet. Avant de commencer à créer notre solution, il est impératif d'observer et d'examiner l'écosystème existant dans le domaine de la constatation automobile. Cette démarche nous permettra de bénéficier d'un aperçu concret et approfondi des solutions existantes , ainsi que mieux comprendre les besoins des utilisateurs potentiels.

Analyse de l'existant

Bien que le constat amiable papier reste largement utilisé en Tunisie , des efforts sont déployés par certain organismes pour moderniser ce processus. Nous pouvons siter à titre d'exemple des applications mobiles qui sont disponibles pour permettre aux conducteurs de déclarer rapidement et facilement un accident à leur assureur. Ces applications offrent

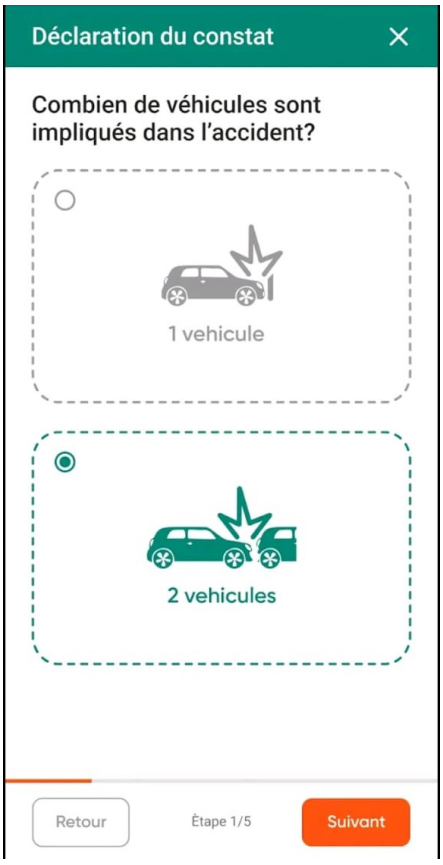
souvent des fonctionnalités supplémentaires, telles que la possibilité de prendre des photos des dommages, de géolocaliser l'accident, ou encore de contacter directement les services d'assistance en cas d'urgence.

DigiConstat

Aujourd'hui, **AVIDEA** offre une application mobile intitulée **DigiConstat**. Elle est intégrée à la solution DigiClaim et disponible sur GOOGLE Playstore. Cette solution offre les fonctionnalités suivantes :

— Identification du véhicule par numéro de contrat et immatriculation
— Géolocalisation de l'accident
— Prise de photos de l'accident et des papiers de l'assuré
— Décrire les circonstances par message vocal
— Suivi du reste du parcours sur l'application (expertise, réparation, etc.)

(a) Interface de suivi des constats



(b) Interface d'une étape de déclaration d'un constat

FIGURE 1.2 – Interfaces de l'application DigiConstat

STAR E-CONSTAT

STAR Assurances avait pour ambition de révolutionner la façon dont les accidents automobiles sont déclarés en Tunisie en lançant son application mobile, **STAR E-CONSTAT**, qui est censée être une première sur le marché tunisien. Cependant, cette application n'est pas encore disponible.

Prévue pour être accessible sur Google Play en 2016, elle aurait permis aux utilisateurs de :

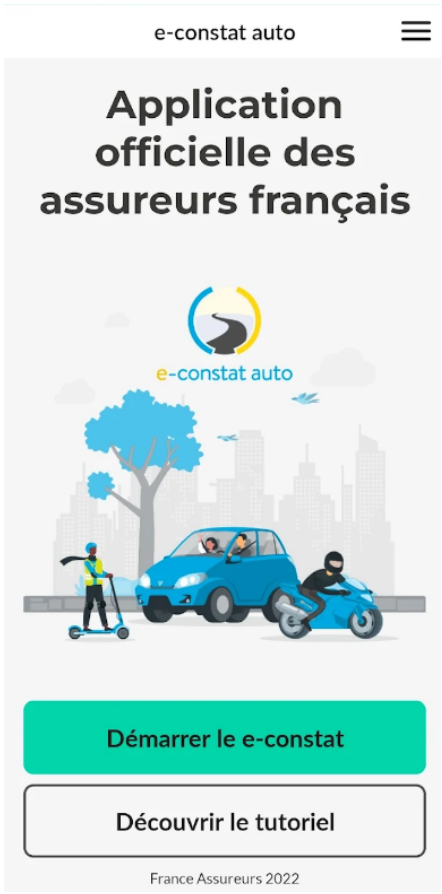
- déclarer un sinistre automobile
- suivre l'avancement du dossier en temps réel
- localiser les agences et centres d'expertise les plus proches
- contacter facilement les conseillers et partenaires de Star par téléphone ou vidéo-conférence.

FIGURE 1.3 – Interface de l'application Star e-constat

E-CONSTAT AUTO

E-CONSTAT AUTO est l'application officielle des assureurs français développée sous l'égide de **France Assureurs** dédiée aux accidents n'ayant pas entraîné de dommages corporels. Toutefois, dans les cas d'accident à l'étranger ou avec un véhicule étranger, d'accident corporel ou lorsque plus de deux véhicules sont impliqués, il est encore nécessaire d'utiliser un constat papier. Cette application offre plusieurs fonctionnalités pratiques, notamment :

- Pré-remplissage des informations personnelles
- Prise de photos de l'accident
- Géolocalisation de l'accident
- Réalisation d'un croquis



(a) Interface d'acceuil

(b) Interface de l'étape de description de l'accident

FIGURE 1.4 – Interfaces de l'application E-CONSTAT AUTO

Ci-joint un tableau comparatif des solutions mentionnées :

TABLE 1.1 – Comparaison des solutions existantes

	DigiConstat	**STAR E-CONSTAT**	**e-constat auto**
Sociéte	AVIDEA	STAR Assurances	France Assureurs
Pays	Tunisie	Tunisie	France
Déploiement	Oui	Non	Oui
Fonctionnalités	- Identification par numéro de contrat et immatriculation - Géolocalisation de l'accident - Prise de photos - Messagerie vocal - Suivi du parcours	- localisation des agences proches - Contact via téléphone ou vidéo-conférence - Suivi en temps réel - mémorisation des données de l'assuré et du conducteur	- Ajout de 3 véhicules au maximum - Pré-remplissage des informations -Prise de photos de l'accident - Réalisation d'un croquis
Liste des assurances	Assurances MAGHREBIA	Non disponible	Aucune

Critique de l'existant

En étudiant les solutions existantes, nous avons constaté que le marché tunisien manque de solutions complètes. En effet, l'application déployée ne propose qu'une seule assurance disponible, ce qui limite le nombre des utilisateurs potentiels. De plus, la procédure de rédaction peut-être longue et ennuyeuse puisque l'utilisateur est obligé de retaper les informations à chaque fois. De surcroît, certaines solutions ne sont pas déployées, les rendant ainsi indisponibles pour les clients. Parmi ces solutions étudiées,e-constat auto semble être la plus intérssante mais elle est disponible pour les français, son design n'est pas très attrayant et elle souffre fréquement de bugs.

1.2.3 Solution proposée

Nous proposons une application mobile offrant une solution complète pour la déclaration facile et rapide des accidents automobiles. Conçue pour être accessible aux utilisateurs possédant ou utilisant un véhicule, elle offre la possibilité d'enregistrer les données des véhicules telles que l'immatriculation et le numéro de série ainsi que celles du conducteur. Les utilisateurs pourront ensuite rédiger un constat pour le véhicule sélectionné, consulter l'avancement de leurs dossiers ainsi que vérifier les détails de leurs contrats d'assurance. En parallèle, les agents de chaque compagnie d'assurance pourront répondre aux constats après les avoir évalués. Notre application vise à offrir une expérience transparente et fiable, contribuant ainsi à renforcer la confiance des assurés envers leurs assurances et vice versa.

1.3 Methodologie adoptée

La méthodologie joue un rôle primordial dans la gestion de tout projet. Elle permet de définir les étapes, les processus et les outils nécessaires pour atteindre les objectifs fixés. Cette section fournira une vue d'ensemble détaillée des approches.

1.3.1 Étude des méthodes existantes

Dans le cadre de l'étude de notre projet, nous essayons d'étudier quelques méthodologies pour pouvoir decider celle qui convient le mieux avec notre projet :

Méthode en cascade

"La méthode en cascade" ou "cycle en V" est une approche classique et traditionelle qui repose sur le traitement séquentiel des phases d'un projet. Elle est souvent privilégiée dans les projets où les besoins sont bien compris et relativement stables dès le début, et où les modifications sont coûteuses ou difficiles à intégrer une fois que le processus est en cours. Ces étapes sont :

— Spécification et analyse des besoins
— Conception ou planification générale
— Développement et production
— Test et corrections
— Livraison

Toutes ces étapes dépendent les unes des autres, une tâche ne pourra être commencée que lorsque la précédente a bien été validée.

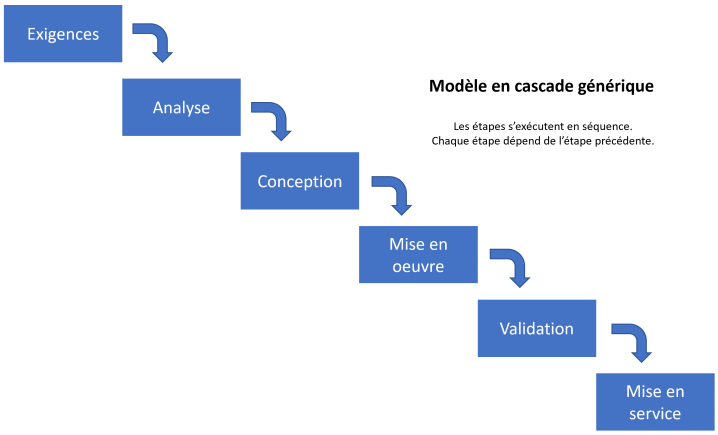


FIGURE 1.5 – La méthode en cascade

SCRUM

Scrum est un cadre de travail (framework) lié aux méthodes agiles. Reposant sur trois piliers fondamentaux : la transparence, l'inspection et l'adaptation, cette méthode permet de répondre à des problèmes complexes dans un environnement incertain et turbulent, tout en livrant de manière productive et créative des produits de la plus grande valeur possible. Elle se caractérise par son approche itérative et incrémentielle. En plus, elle organise

le processus de développement en utilisant une série d'événements, de rôles et d'artéfacts. Voici les principales étapes ou éléments constitutifs de Scrum :

- Définition du cadre du projet
- Préparation le backlog
- Travail sur les tâches de votre sprint
- Récolte des feedbacks
- Recommencer

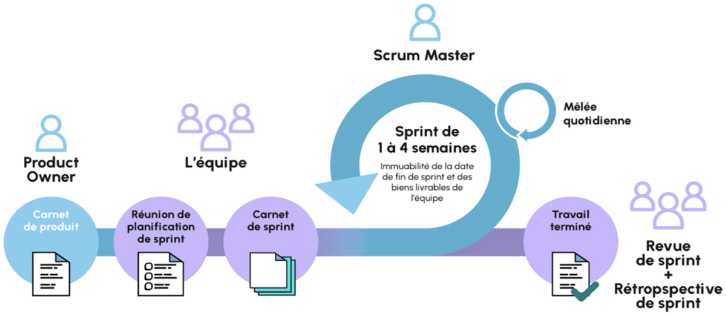


FIGURE 1.6 – Processus SCRUM

Two Track Unified Process (2TUP)

2TUP (2 track unified process) est une adaptation de Unified Process (UP). Il conserve certains aspects de UP, mais introduit également un cycle de développement en Y, qui dissocie les aspects techniques des aspects fonctionnels.Le processus s'articule ensuite autour de trois phases essentielles :

- ***Une branche technique*** qui se concentre sur les aspects techniques du projet : l'architecture logicielle, la conception détaillée et le développement des composants du système.
- ***Une branche fonctionnelle*** se concentre sur les aspects fonctionnels du projet : l'analyse des besoins, la spécification des fonctionnalités et la validation.
- ***Une phase de réalisation*** qui est la fusion des deux branches technique et fonctionnelle pour former une version fonctionnelle et technique du produit. Elle implique le développement, les tests et la livraison du produit final.

Chaque phase comprend ses propres étapes spécifiques, qui lui sont adaptées.

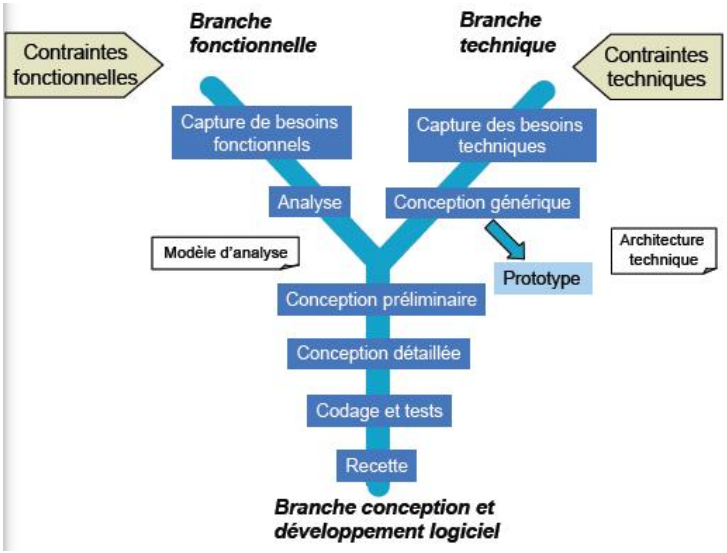


FIGURE 1.7 – Processus 2TUP

1.3.2 Comparaison entre les méthodologies

Avant de plonger dans la comparaison entre les Méthode de gestion d'un projet, il est essentiel de comprendre qu'elles diffèrent dans leur philosophie, leur structure et leur exécution. Chacune de ces méthodologies offre des avantages uniques en fonction des besoins spécifiques du projet et l'équipe de développement.

Voici un tableau comparatif entre la Méthode en cascade, Scrum et 2TUP en fonction de plusieurs critères :

TABLE 1.2 – Comparaison entre Méthode en Cascade, Scrum et 2TUP

Critère	**Cascade**	**Scrum**	**2TUP**
Type de projet	Moyen à grand	Variété de tailles	Variété de tailles
Exigences	Définies en amont et peu susceptibles de changer	Peuvent évoluer au fil des itérations	Flexibles et évoluer grâce à la planification incrémentielle
Approche de développement	Séquentielle, chaque phase dépendante de la précédente	Itérative et incrémentielle	Une approche hybride en Y combinant développement itératif et validation
Type d'équipe	Taille et expérience variables	Petite équipe auto-organisée	Taille et niveau d'expérience variable
Gestion du temps	Rigide et difficiles à ajuster	Les sprints ont une durée fixe, mais les priorités peuvent être réévaluées	Flexible en intégrant des cycles de développement et de validation simultanés
Communication	Limitée	Des réunions régulières	Continue
Adaptabilité	Faible	Elevée	Elevée
Risques	Identifiés tardivement et coûteux à corriger	Gérés de proactivement	Identifiés et gérés à travers les branches
Implication du client	Au début de projet	Tout au long du projet	Dans les phases de développement et de validation

1.3.3 Méthodologie adoptée : 2TUP

Suite à cette étude et afin de contrôler les risques et mener à bien ce projet, nous avons décidé d'opter pour le processus 2TUP (Two Track Unified Process) pour plusieurs raisons importantes. En effet, notre projet présente des besoins fonctionnels qui peuvent évoluer ou changer au fil du temps. Avec 2TUP, nous pouvons intégrer une approche hybride qui combine le développement itératif et la validation, ce qui nous permet de gérer efficacement les changements des exigences. De plus, cette méthode offre une flexibilité permettant l'adaptation du processus aux compétences de l'équipe. En conclusion, le processus 2TUP offre la flexibilité, la collaboration et la qualité nécessaires pour mener notre projet à bien dans un environnement dynamique et évolutif.

Conclusion

Ce chapitre présente l'organisme d'accueil et expose la problématique du projet **AssurTN**, abordant l'étude des solutions existantes, la comparaison des méthodes de gestion de projet, et la sélection de 2TUP comme méthodologie. Il présente aussi la solution envisagée, une application mobile e-constat.

Chapitre 2

Analyse et spécification des besoins

Intoduction

La spécification de besoins est une étape fondamentale dans l'édification de notre projet. En effet, elle incarne la transition de la vision abstraite du concept à la réalisation concrète de l'application. Ce chapitre sera une cartographie qui nous aidera à explorer d'abord les acteurs impliqués , ensuite leurs tâches et finalement les besoins non fonctionnels que notre système doit satisfaire pour assurer le fonctionnement attendu.

2.1 Identification des acteurs

Notre solution sera composée de deux applications : une application mobile dédiée aux conducteurs et une application web destinée aux assureurs.

2.1.1 Application mobile pour les conducteurs

Elle sera utilisée par un seul acteur qui sera :
Le conducteur : C'est l'utilisateur qui s'inscrit sur notre application afin de pouvoir bénéficier de toutes les fonctionnalités qui lui sont proposé lors de la déclaration d'un constat.

2.1.2 application web pour l'agent de la compagnie d'assurance

Cette application sera destinée à :
L'agent de la compagnie d'assurance : C'est le responsable de l'évaluation des constats et la prise de décision.

2.2 Spécification des besoins

Cette partie vise à rassembler, analyser et définir les besoins de haut niveau ainsi que les caractéristiques clés du projet. Nous nous focalisons sur les exigences fonctionnelles et non fonctionnelles qui sont indispensables à la création d'une application non seulement performante, mais également aisément accessible.

2.2.1 Les besoins fonctionnels :

Les besoins fonctionnels représentent les capacités et les actions concrètes que l'application doit fournir pour accomplir les tâches attendues et pour répondre aux exigences de l'utilisateur.

Partie Conducteur

Pour l'application mobile du côté conducteur, les principaux besoins fonctionnels se résument aux points suivants :

- **S'inscrire dans l'application :** l'utilisateur s'inscrit obligatoirement dans l'application mobile afin de bénéficier des fonctionnalités existantes.
- **S'authentifier :** l'utilisateur accède à l'application en saisissant ses identifiants.
- **Gérer profil :** Une fois connecté, l'utilisateur a la possibilité de consulter et de mettre à jour son profil.
- **Ajouter un véhicule :** L'utilisateur a la possibilité d'ajouter un ou plusieurs véhicules qu'il possède ou conduit simplement en saisissant leurs détails.
- **Rédiger un constat :** L'utilisateur a la possibilité de rédiger un constat de manière spontanée ou pour un véhicule déjà enregistré.
- **Consulter l'état de la demande :** Après avoir rédiger un constat, l'utilisateur aura à sa disposition une page qui lui permettra de consulter l'état de sa demande.

Partie agent de l'assurance

Pour la partie application mobile, côté agent , les principaux besoins fonctionnels se résument aux points suivants :

- **S'authentifier :** l'agent accède à l'application en saisissant ses identifiants fournis par leur assurance.
- **Gérer profil :** Une fois connecté, l'agent a la possibilité de consulter et de mettre à jour son profil.
- **Consulter et gérer les les demandes de constatation :** L'agent aura un aperçu sur la liste des demande et pourra prendre une décision (accepter ou réfuser)

2.2.2 Les besoins non fonctionnels

Les besoins non fonctionnels vont au-delà des fonctionnalités opérationnelles . En effet, Ils jouent le rôle d'un indicateur de qualité et de niveau de service du fonctionnement d'un système. Dans cette partie, nous aborderons les principaux exigences de notre application.

Exigences de qualité

— ***Interfaces conviviales :*** Afin de réaliser cet objectif, nous avons utilisé Figma pour la conception d'interfaces utilisateur. Comme nous souhaitons que notre application soit aisément exploitable par des conducteurs de différents âges, niveaux culturels et compétences technologiques, nous avons conçu ces interfaces simples et conviviales. Nous avons sélectionné des couleurs, des polices et des icônes familières pour l'œil de l'utilisateur. De plus, nous avons aménagé la disposition de manière à ce que l'expérience utilisateur soit ergonomique, compréhensible et accessible à tous.

— ***Auto-remplissage des questions :*** Lors de la rédaction d'un constat, certaines données peuvent être redondantes, ce qui rends le processus long et ennuyeux. Donc,nous avons réfléchi à une fonction d'auto-remplissage de certaines questions, telles que les informations du conducteur, de l'assuré, et du véhicule. Par conséquent, l'utilisateur gagnera du temps et d'effort.

Exigences de sécurité

— ***authentification à facteurs multiples :*** Puisque notre application contient des données confidentielles des utilisateurs, nous devons intégrer une authentification à facteurs multiples (MFA), comprenant l'authentification par carte d'identité et empreinte digitale, afin d'assurer un niveau élevé de sécurité.

— ***JSON Web Tokens :*** Afin de sécuriser les sessions et garantir l'intégrité et la confidentialité des données, nous utilisons les JSON Web Tokens (JWT). Cette méthode de gestion de l'authentification repose sur l'autonomie du token, sa signature et son chiffrement. Cela permet de vérifier que le token n'a pas été altéré et que les données contenues restent confidentielles.

Exigences d'extensibilité et de maintenabilité :

Notre solution doit être facilement mise à jour ou étendue afin de répondre aux nouveaux besoins sans nécessiter de modifications majeures de son architecture. Par suite, notre application doit adopter le principe du couplage faible et des bonnes pratiques de programmation. En outre, le code doit être lisible, commenté et documenté.

2.3 Diagrammes de cas d'utilisation

Après avoir identifié les acteurs principaux de notre système et les cas d'utilisation, nous allons illustrer le diagramme comportemental de cas d'utilisation général afin de représenter les interactions entre les acteurs et le système.

2.3.1 Diagramme de cas d'utilisation global Conducteur

La figure suivante présent le diagramme de cas d'utilisation du conducteur.

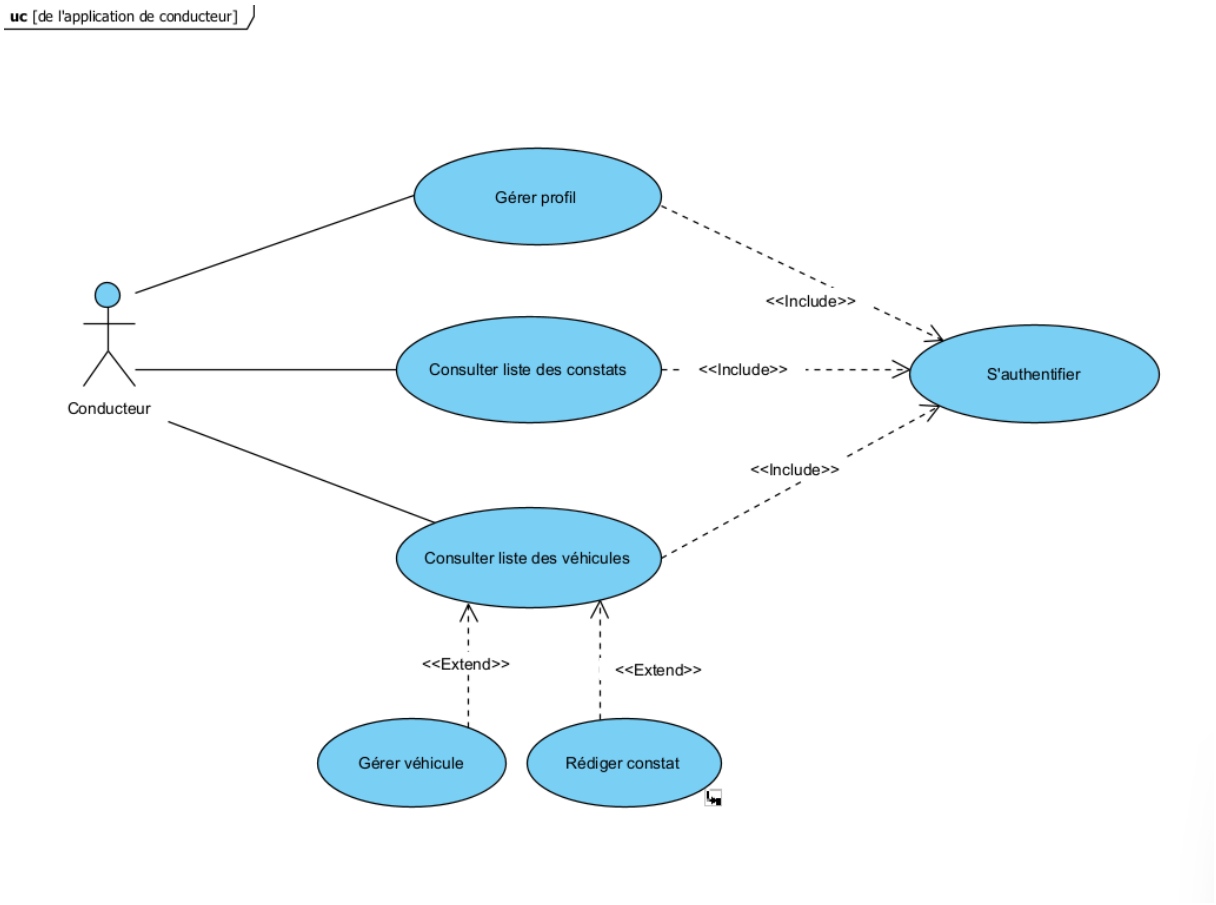


FIGURE 2.1 – Diagramme de cas d'utilisation Conducteur

En accèdant à notre application, l'utlisateur pourra rédiger un constat en suivant des étapes organisées. Ces étapes comprennent : l'authentification, la consultation de la liste des voitures , la sélection de la voiture concernée ou le démarrage un spontanné puis la réponse au formulaire. Après avoir validé et signé le constat, l'utilisateur pourra consulter la liste des demandes déjà envoyées.

Diagramme de cas d'utilisation Rédiger constat

Le cas d'utilisation "Rédiger constat" estla fonctionnalité principale de notre application, conçue pour permettre aux utilisateurs de documenter les détails d'un accident de manière précise et efficace. La figure suivante présente le diagramme de cas d'utilisation "Rédiger constat " d'une manière plus détaillée :

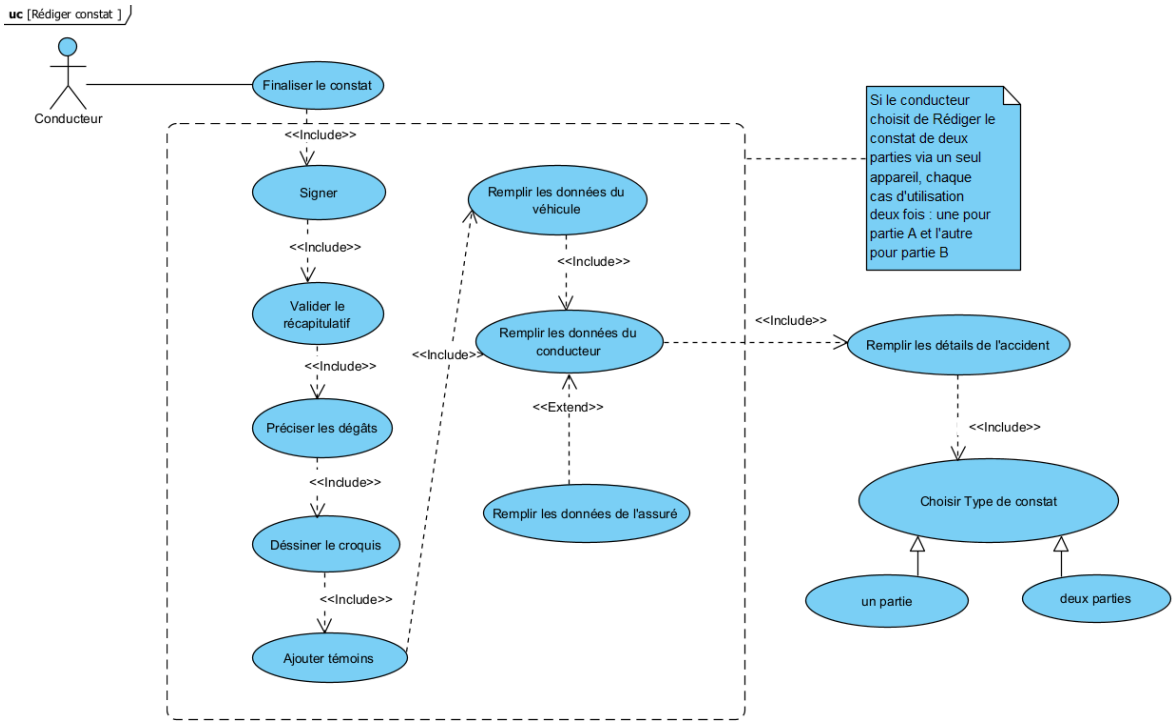


FIGURE 2.2 – Diagramme de cas d'utilisation Rédiger constat

Acteur principal	Conducteur
Objectif	L'utilisateur veut déclarer un accident à son assureur en rédigeant un constat.
Préconditions	1. L'utilisateur est authentifié dans l'application. 2. L'utilisateur navigue vers l'interface de la liste des voitures.
Déclencheur	L'utilisateur sélectionne la fonction de rédaction de constat.
Scénario principal	1. L'utilisateur s'authentifie. 2. L'utilisateur navigue vers l'interface de la liste des voitures. 3. L'utilisateur sélectionne la fonction de rédaction soit par la carte de voiture concernée, soit en tappant le bouton d'un nouveau constat. 4. L'utilisateur choisit entre "Choisir constat par deux parties" et "Choisir constat par une seule partie" 5. Si l'utilisateur choisit "Choisir constat par deux parties", il peut sélectionner "Rédiger constat sur un appareil" ou "Rédiger constat sur deux appareils séparés" 6. Si l'utilisateur choisit "Choisir constat par une seule partie", il peut sélectionner "Utiliser un code fourni par le système" ou "Créer un nouveau constat" 7. L'utilisateur répond aux questions du formulaire.
Postconditions	1. Si l'utilisateur valide l'envoie du constat, la demande s'affiche dans la partie liste des constats 2. Si Le conducteur n'est pas l'assuré, l'assuré doit recevoir une notification.

2.3.2 Diagramme de cas d'utilisation Agent de l'assurance

La figure suivante présent le diagramme de cas d'utilisation de l'agent de l'assurance :

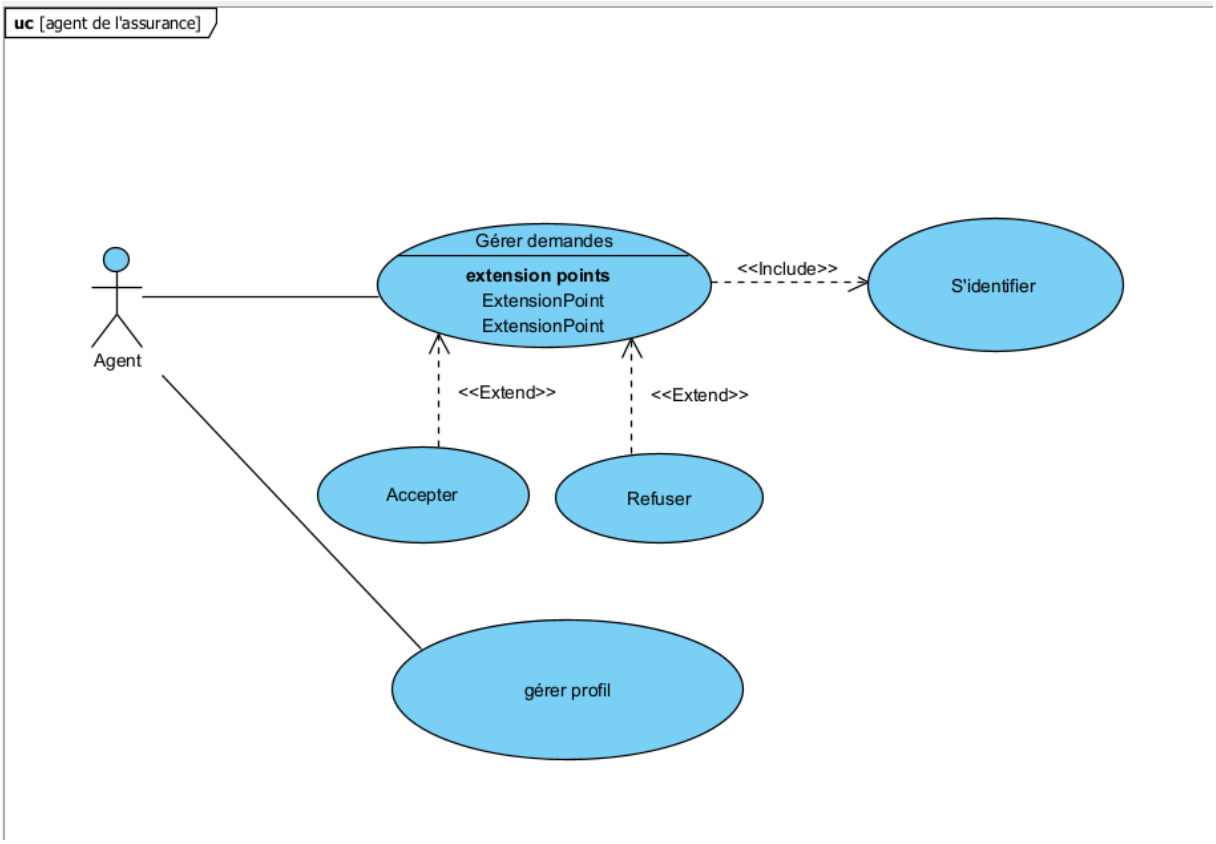


FIGURE 2.3 – Diagramme de cas d'utilisation de l'agent de l'assurance

En accèdant à l'application dédiée à l'agent, l'utlisateur pourra consulter de la liste des demandes , et prendre la décision, soit accèpter ou réfuser.

Acteur principal	Agent de l'assurance
Objectif	L'admin veut consulter et gérer les fiches techniques.
Préconditions	L'utilisateur s'est identifié dans l'application.
Déclencheur	L'admin sélectionne la marque, modèle et la version auquels la fiche appartient.
Scénario principal	1. L'agent s'identifie avec les données déjà fournis. 2. L'agent navigue vers la page de gestion des demandes. 3. Après avoir consulter la demande, l'agent décide d'accepter ou réfuser.
Postconditions	L'agent est redirigé vers la page des listes des demandes.
Autres scénarios	L'agent peut annuler le processuss.

2.4 Diagrammes de séquence système

Les diagrammes de séquence se concentrent sur les lifelines, les processus et les objets qui coexistent simultanément, ainsi que les messages échangés pour exercer une fonction avant la fin de la ligne de vie. Dans cette section, nous avons présenté les diagrammes de séquence les plus critiques de notre application.

2.4.1 Diagramme de séquence système “Rédiger un constat”

Le diagramme de séquence système suivant montre l’interaction entre le conducteur et l’application.

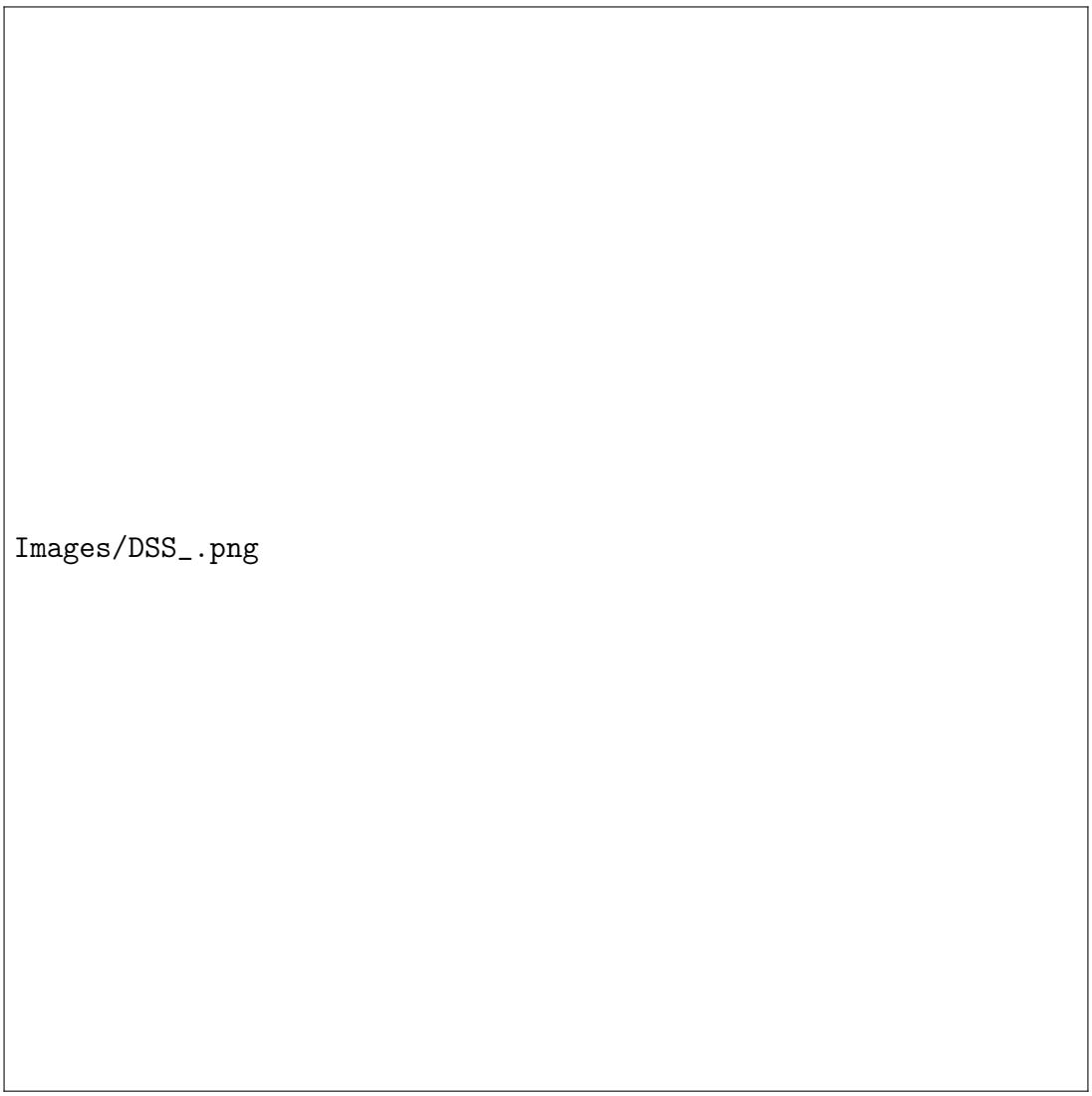


FIGURE 2.4 – Diagramme de séquence système “Rédiger un constat”

Le diagramme de séquence système correspondant au client montre que ce dernier, authentifié, et passant par l’interface des listes de voiture, peut

2.4.2 Diagramme de séquence système “Répondre à la demande”

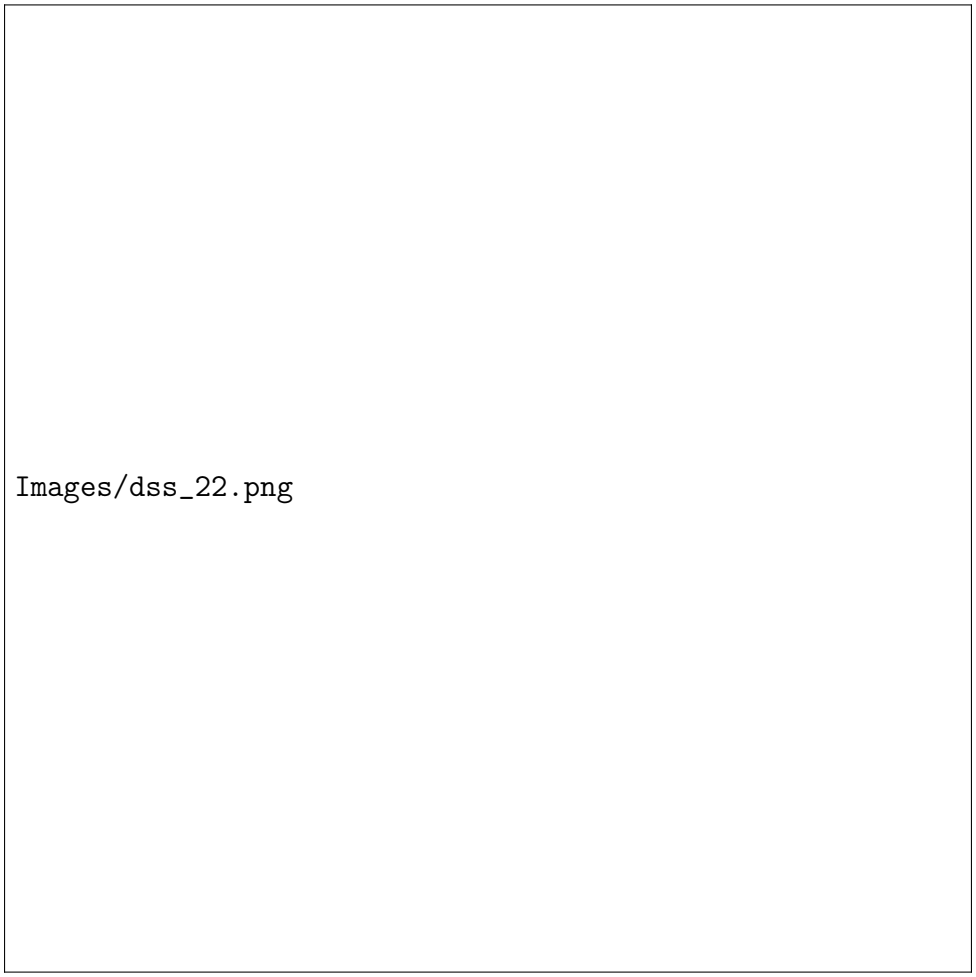


FIGURE 2.5 – Diagramme de séquence système “Répondre à la demande”

Le diagramme de séquence système correspondant à l’agent montre que ce dernier, authentifié, peut consulter, lire puis changer l’état du constats.

Conclusion

Ce chapitre a couvert la spécification des besoins pour le projet **AssurTN**, en identifiant les acteurs, leurs rôles, et les besoins fonctionnels et non fonctionnels. Les diagrammes de cas d’utilisation et de séquence système ont été présentés pour illustrer les interactions entre les acteurs et le système. Cette phase de spécification jette les bases nécessaires pour le développement de l’application **AssurTN** .

Chapitre 3

Aperçu conceptuel

Introduction

Après avoir précisé les différentes fonctionnalités de notre projet dans le chapitre précédent, nous visons dans ce chapitre à explorer en détail son architecture en mettant en évidence les patrons de conception adoptés. Nous suivons également l'évolution de la conception UML, offrant une vue approfondie des choix de conception. De plus, nous allons illustrer une compréhension visuelle de l'interface utilisateur.

3.1 Architecture du projet

L'architecture logicielle décrit de manière symbolique et schématique les différents éléments et leurs interactions. Cette étape est particulièrement cruciale du développement, car elle va influencer la stabilité, la robustesse ainsi que la scalabilité de notre application.

3.1.1 Architecture générale : 3-tiers

L'architecture 3-tiers est l'application du modèle plus général qu'est le multi-tiers. Cette approche divise l'application en trois couches distinctes : **la présentation**, **la logique métier** et **la couche de données**.

- **La couche de Présentation (UI - User Interface) :**
  C'est la partie de l'application avec laquelle l'utilisateur interagit directement. Elle est responsable de l'affichage des informations et de la collecte des entrées de l'utilisateur. Dans notre application mobile, cette couche comprend les vues, les contrôleurs et les widgets d'interface utilisateur qui constituent les interfaces.
- **La couche de Logique Métier (Business Logic) :**
  Cette couche est responsable de la logique applicative de l'application : la manipulation des données et de l'exécution des opérations métier. Elle veille à ce que les données soient traitées de manière appropriée et ne doit pas être directement liée à la source de données, mais plutôt s'appuyer sur une couche de données distincte pour accéder aux données.
- **La couche de Données (Data Layer) :**
  Cette couche est responsable de la gestion et de l'accès aux données de l'application. Elle interagit directement avec la source de données, telle qu'une base de données ou un fichier, etc.

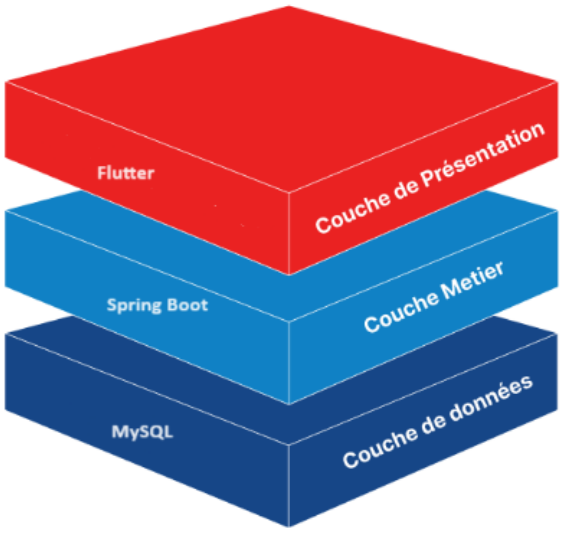


FIGURE 3.1 – Architecture 3-tiers

3.2 Patrons de conception

Les "patrons de conception" sont des solutions éprouvées pour résoudre des problèmes courants dans la conception de logiciels. Ils sont comme des modèles qui aident à organiser chaque couche d'une application de manière efficace. Ils simplifient la création d'applications robustes, en aidant à structurer et à organiser chaque couche pour une meilleure maintenabilité.

3.2.1 Le modèle MVC

Le modèle Modèle-Vue-Contrôleur (MVC) est un modèle d'architecture logicielle qui divise une application en trois composants interconnectés pour améliorer la structure, la maintenabilité et la flexibilité du code.

1. **Modèle** Le modèle représente les données et la logique métier de l'application. Il encapsule la gestion des données, les règles métier et les opérations sur les données. Le modèle est indépendant de l'interface utilisateur ou de la présentation.
2. **Vue** La vue est responsable de l'affichage des données au sein de l'interface utilisateur. Elle réagit aux changements dans le modèle et affiche les données de manière appropriée. La vue est généralement passive et ne contient pas de logique métier significative.
3. **Contrôleur** Le contrôleur agit comme un intermédiaire entre le modèle et la vue. Il reçoit les entrées de l'utilisateur, interagit avec le modèle pour mettre à jour les données, puis met à jour la vue en conséquence. Le contrôleur contient souvent la logique de gestion des événements et des interactions utilisateur.

3.2.2 Repository

Le modèle de conception Repository est un modèle de conception créationnel qui fournit une couche d'abstraction entre la logique métier de l'application et le code d'accès aux données. L'objectif principal du modèle Repository est de séparer la logique qui récupère les données du magasin de données sous-jacent (par exemple, une base de données, une API, un système de fichiers) du reste de l'application

3.3 Conception UML

UML (Unified Modeling Language) est un langage graphique standardisé utilisé pour modéliser, concevoir et documenter les systèmes logiciels. Il offre une notation visuelle permettant de représenter les différentes perspectives d'un système, y compris ses classes, objets, interactions, et structures, facilitant ainsi la communication entre les membres d'une équipe de développement. UML comprend divers diagrammes, dans cette partie nous présentons les diagrammes de classes, de séquence et le diagramme de classes de conception, offrant une vue globale du système.

3.3.1 Diagramme de classes

Le **Diagramme de Classe** UML représente graphiquement la structure statique d'un système logiciel, illustrant les classes du système, leurs relations, les attributs, et les méthodes. Vous trouvez ci-dessous le diagramme de classe de notre application.

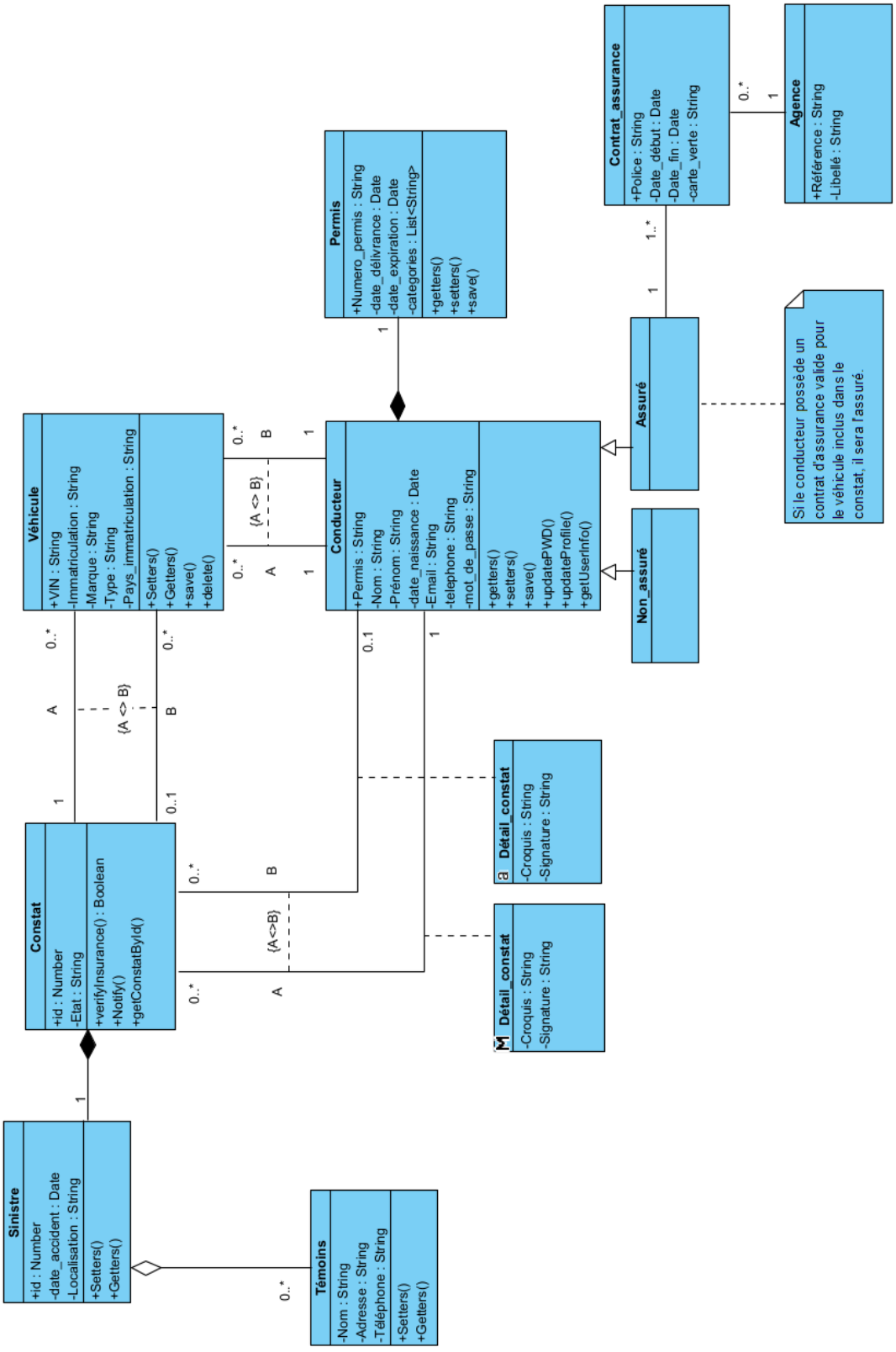


FIGURE 3.2 – Diagramme de classes

3.3.2 Diagrammes de séquences

Le **Diagramme de Séquence** UML représente graphiquement les interactions entre les objets d'un système au fil du temps. Il illustre la séquence des messages échangés entre les objets, avec des lignes de vie pour chaque objet participant. Dans cette partie nous illustrons quelques diagrammes de séquence de notre application.

Diagramme de sequence Rédiger Constat

Ce Diagramme illustre les différentes intéractions entre les composants de notre système afin de ...

********* inserer les figures **********

3.4 Conclusion

En résumé, ce chapitre explore l'architecture du projet en détaillant les choix de conception et l'évolution UML. L'adoption de l'architecture orientée services (SOA) est mise en avant, accompagnée de l'explication des patrons de conception tels que le modèle MVC et le modèle Repository. La conception UML, illustrée par des diagrammes de classes, de séquences, et de classes de conception, offre une vision complète du système.

Chapitre 4

Réalisation

Intoduction

Ce chapitre a pour but de spécifier les différentes technologies ainsi que les languages de programmation utilisées tout au long de la réalisation de ce projet. Cette partie sera suivi par la précision de l'architecture globale de l'application. Enfin, nous clôturerons par la présentation de quelques interfaces.

4.1 Environnement de développement

Dans cette partie, on va présenter l'environnement logiciel du développement de notre application.

4.1.1 Technologie de programmation

Spring Boot

FIGURE 4.1 – Logo Spring Boot

Spring Boot est framework accélère et facilite le développement d'applications et de microservices avec Java Spring Framework.

Flutter

FIGURE 4.2 – Logo Spring Boot

Développé par Google en 2017, Flutter[13] est un KIT de développement logiciel d'interface utilisateur open-source, multiplateforme et native. Il utilise le langage Dart pour concevoir des applications pour Android, iOS, Web et desktop.

4.1.2 Langages de programmation

Java

FIGURE 4.3 – Logo Java

Le Java est un langage de programmation orientée objet basé sur le C++ et développé par Sun Microsystems en 1995, et racheté par Oracle en 1995. Parmis ces avantages, on cite son interopérabilité puisqu'il fonctionne parfaitement sur Windows ainsi que sur Mac et Linux, et sur une divers types d'appareils (centres de données, ordinateur, téléphone mobile, ...).

Dart

FIGURE 4.4 – Logo Dart

Dart est un langage de programmation optimisé pour les applications sur plusieurs plateformes. Il est développé par Google et est utilisé pour créer des applications mobiles, de bureau, de serveur et web. Dart est un langage orienté objet à ramasse-miettes avec une syntaxe de type C++.

4.1.3 *Outils et Logiciels*

Dans cette partie, nous allons mentionner les logiciels qui nous ont aidés à coder et tester notre application.

Android Studio

FIGURE 4.5 – Logo Android Studio

Basé sur IntelliJ IDEA, Android Studio est un environnement de développement intégré (IDE) conçu spécifiquement pour les applications mobiles Android. . Cet envronnement utilise le moteur de production Gradle, qui à son tour, offre des fonctionnalités robustes pour gérer et automatiser le processus de création des applications.

IntelliJ

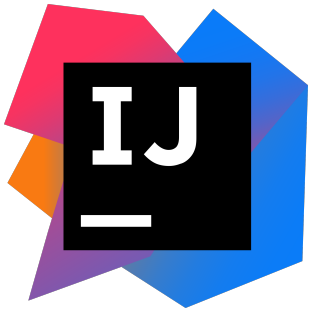


FIGURE 4.6 – Logo IntelliJ

IntelliJ IDEA est un environnement de développement intégré (IDE) utilisé pour créer des applications basées sur Java. Il offre divers fonctionnalités avancées qui facilitent la création d'applications et de logiciels Java fiables.

WampServer

FIGURE 4.7 – Logo WampServer

WampServer est un ensemble d'outils de développement web de type WAMP permettant la création d'un serveur de développement local. En plus, il intègre phpMyAdmin, une interface web qui facilite l'administration et la gestion des bases de données MySQL.

Postman

FIGURE 4.8 – Logo Postman

Postman est un outil qui permet de tester les interfaces de programmation d'applications (API). Il offre une plateforme très simple à manipuler facilitant le test, le débogage, et la gestion des API.

Github

FIGURE 4.9 – Logo Github

GitHub est une plateforme en ligne qui aide les utilisateurs à partager, collaborer, réviser et contribuer à des projets logiciels, en permettant notamment de stocker du code, de gérer les problèmes et de suivre les activités des projets.

Overleaf

FIGURE 4.10 – Logo Overleaf

Overleaf est une plateforme de rédaction collaborative en ligne et gratuite qui utilise le système LaTeX. Elle permet de produire des documents tels que des articles de recherche et des rapports techniques de haute qualité.

Visual Paradigm

FIGURE 4.11 – Logo Overleaf

Visual Paradigm est un logiciel de modélisation et de conception. Il est prend en charge de nombreux diagrammes commerciaux et techniques comme UML, BPMN, URD...

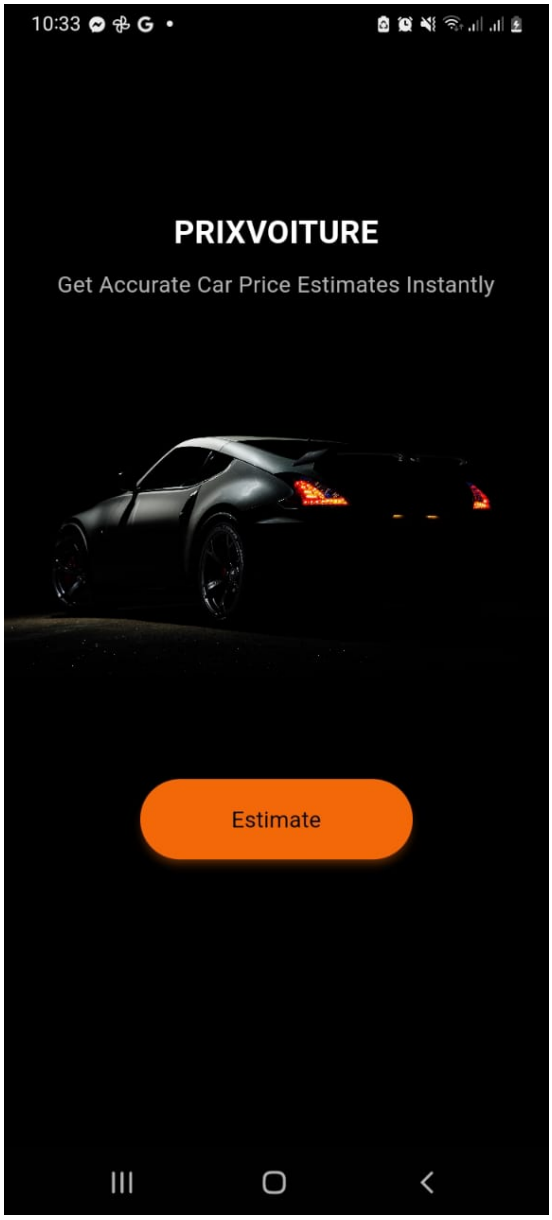
4.2 Aperçu sur le travail réalisé

4.2.1 Interfaces des application

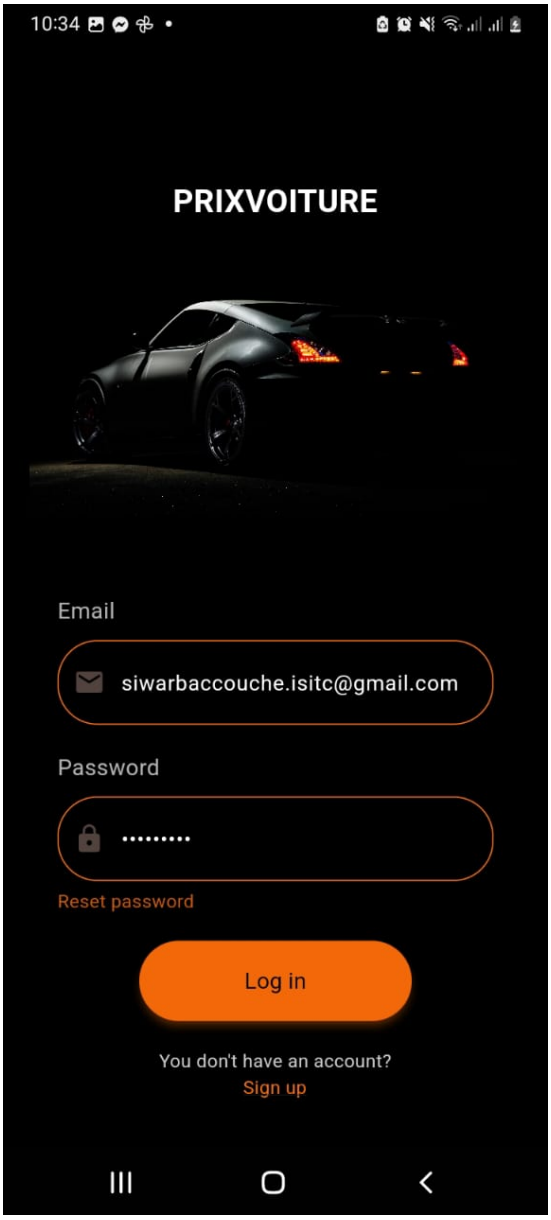
Dans cette partie, nous préssenterons les différentes captures d'écran des interfaces de l'application mobile.

Interfaces conducteur

Etape 1 : Pour commencer, après avoir passé la page d'acceuil, le visiteur est invité à se connecter ou à créer un compte s'il l'en a pas. Si toutes ces données sont correctement saisies, il sera dérigé vers la page de la liste des voitures.



(a) Interface de page d'acceuil



(b) Interface de saisi de données

Conclusion

Ce chapitre a détaillé l'environnement de développement utilisé, mettant en avant des technologies telles que Spring Boot et Flutter, ainsi que les langages Java et Dart. Il a également présenté des outils essentiels comme Android Studio, IntelliJ, WampServer, Postman, et GitHub. Enfin, des captures d'écran ont offert un aperçu des interfaces utilisateur, décrivant le flux depuis la page d'accueil jusqu'au choix du type de quiz.

Conclusion

L'évolution constante de l'industrie automobile .....