|  |  |  |
| --- | --- | --- |
|  | Université Chouaib Doukkali  Ecole Nationale des Sciences Appliquées d’El Jadida  Département Télécommunications, Réseaux et Informatique |  |

**Projet de fin d’Année**

Filière : **2ITE**

Niveau : **1ère Année**

PFA n° : **F0-054-116**

Intitulé

**Conception et développement d’une Application Mobile**

**Smart-Assurance**

Réalisé Par :

**Omar Mazer & Omar Lahrach**

Encadré par :

**Prof. Mohammed BOUSMAH**

Présenté le **/ /2020** devant le jury composé de :

Prof., Professeur à l’ENSAJ

Prof., Professeur à l’ENSAJ

Année Universitaire : 2019/2020

Résumé

Remerciements

Table des matières

[1 Introduction 9](#_Toc39579349)

[2 Présentation du projet 10](#_Toc39579350)

[Introduction 10](#_Toc39579351)

[2.1 Présentation générale du projet 10](#_Toc39579352)

[2.1.1 Problématique 10](#_Toc39579353)

[2.1.2 Etude de l’existant 10](#_Toc39579354)

[2.2 Objectifs du projet 12](#_Toc39579355)

[2.3 Solution proposée 12](#_Toc39579356)

[Conclusion 12](#_Toc39579357)

[3 Spécification des besoins 13](#_Toc39579358)

[Introduction 13](#_Toc39579359)

[3.1 Besoins fonctionnels 13](#_Toc39579360)

[3.2 Besoins non-fonctionnels 14](#_Toc39579361)

[3.3 Traduction des besoins en diagrammes de cas d'utilisation 14](#_Toc39579362)

[3.3.1 Description des acteurs 14](#_Toc39579363)

[3.3.2 Diagramme de cas d'utilisation 15](#_Toc39579364)

[Conclusion 16](#_Toc39579365)

[4 Architecture et conception 17](#_Toc39579366)

[Introduction 17](#_Toc39579367)

[4.1 Conception générale 17](#_Toc39579368)

[4.2 Conception détaillée 18](#_Toc39579369)

[4.2.1 Diagramme de classes 18](#_Toc39579370)

[4.2.2 Digramme de séquences 19](#_Toc39579371)

[4.2.3 Digramme d’activité 20](#_Toc39579372)

[Conclusion 21](#_Toc39579373)

[5 Réalisation et méthodologies 22](#_Toc39579374)

[Introduction 22](#_Toc39579375)

[5.1 Réalisation 22](#_Toc39579376)

[5.1.1 Environnement logiciel 22](#_Toc39579377)

[5.1.2 Environnement technique 24](#_Toc39579378)

[5.1.3 Travail réalisé 26](#_Toc39579379)

[5.2 Description de l’organisation du projet 26](#_Toc39579380)

[5.2.1 Description des méthodologies utilisées 26](#_Toc39579381)

[5.2.2 Description de l’organisation du travail en équipe 26](#_Toc39579382)

[Conclusion 26](#_Toc39579383)

[6 Conclusion et perspectives 27](#_Toc39579384)

[Bibliographie 29](#_Toc39579385)

[A Code source 30](#_Toc39579386)

Liste des figures

[**Figure 1 :** Logos des projets existants. 10](#_Toc39579237)

[**Figure 2 :** Diagramme de cas d’utilisation. 15](#_Toc39579238)

[**Figure 3 :** Schéma de fonctionnement de l’application 18](#_Toc39579239)

[**Figure 4 :** Diagramme de classes 19](#_Toc39579240)

[**Figure 5 :** Diagramme de séquences 20](#_Toc39579241)

[**Figure 6 :** Diagramme d’activité 21](#_Toc39579242)

[**Figure 7 :**Le développement en temps réel avec expo. 23](#_Toc39579243)

[**Figure 8 :**Le logo créé pour l’application. 23](#_Toc39579244)

[**Figure 9 :** Le rôle de NodeJS durant le développement. 25](#_Toc39579245)

[**Figure 10 :** Remplissage du constat par deux smartphones. 26](#_Toc39579246)

Liste des tableaux

[**Tableau 1:** Tableau comparatif des projets existants 11](#_Toc39338893)

[**Tableau 2:** Tableau de description des cas d’utilisation 16](#_Toc39338894)

Liste des codes

[**Code 1:** Code source de la classe "Hello World" **Error! Bookmark not defined.**](#_Toc406137824)

Liste des abréviations

1. Introduction

De nos jours, on vit dans un monde de plus en plus moderne, rapide et connecté ; on est donc confronté à l’obligation de s’adapter rapidement à ce changement de système de vie et d’habitudes quotidiennes, ce qui va nous mener à réfléchir à automatiser et informatiser toute habitude et tout service qui se fait encore aujourd’hui de manière manuelle physique.

En ces dernières années, le monde des smartphones a connu une évolution impressionnante puisque le smartphone est devenu aujourd’hui de l’ordre d’un ordinateur de poche qui vous accompagne partout où vous allez ; c’est cette évolution qui va impliquer le fait que l’idée de l’informatisation des services va être plus orientée vers les technologies mobiles qui visent une population de plus en plus large.

Pour cela, nous avons cherché à créer une application mobile qui va informatiser et faciliter le remplissage d’un constat à l’amiable en cas d’accident entre deux véhicules.

Le remplissage d’un constat à l’amiable nécessite un l’achat d’un imprimé qui n’est pas disponible en cas d’accident hors de zones urbaines, et même en cas de possession de cet imprimé ce n’est si simple la remplir complétement sans la présence d’un constateur. Alors, notre application a comme objectif principal : simplifié le remplissage qui sera fait directement sur les smartphones des conducteurs ou bien sur celui du constateur présent au lieu de l’accident.

Le présent rapport synthétise le travail que nous avons effectué. Il est organisé en chapitres comme suit :

-Le premier chapitre donne une présentation générale du projet.

-Le deuxième chapitre présente les différents besoins auxquels notre projet doit satisfaire.

-Le troisième chapitre englobe la conception de notre projet.

-Le quatrième et le dernier chapitre va se concentrer sur la phase de réalisation du projet ainsi que les méthodologies suivis pour mener à bien notre projet.

-Nous allons terminer ce rapport par une conclusion et des perspectives.

1. Présentation du projet

Introduction

Ce premier chapitre a pour objectif de situer le projet dans son contexte général, à savoir la problématique qui a inspiré la création de notre application, la présentation du projet et les objectifs à atteindre.

* 1. Présentation générale du projet
     1. Problématique

En cas d’accident sans blessés, les conducteurs sont en besoin de remplir un constat à l’amiable afin de déterminer la responsabilité de chaque partie.

Alors, le problème réside dans le fait que dans la plupart des cas ; les conducteurs ne sont pas en possession de ce document donc ils devront en acheter un près du lieu de l’accident, mais si ce lieu d’accident est hors de zone urbaine ils ne pourront pas s’en procurer un.

Ensuite, s’ils ont réussi à trouver ce document ; le remplissage de ce dernier n’est pas totalement évident, dans ce cas les conducteurs auront besoin d’aide au remplissage ou de faire appel à un constateur pour s’en occuper.

* + 1. Etude de l’existant

Il existe pas mal d’applications qui concernent les assurances locales e–t étrangères, mais les seules applications qui mettent en œuvre le service de constat à l’amiable électronique sont les suivants :

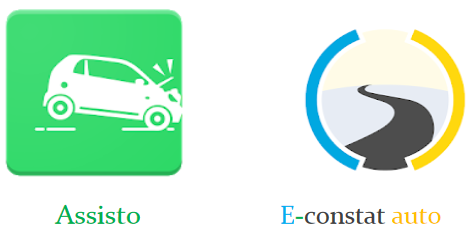


Figure 1 : Logos des projets existants.

Voici un tableau comparatif entre Assisto et E-constat auto :

|  |  |  |
| --- | --- | --- |
|  | Assisto | E-constat auto |
| **Etude Bibliographique** | Assisto est une application mobile développée par l’entreprise belge **PINCH** ,elle est disponible en 15 langues dans 42 pays Européens. | E-constat auto est une application mobile développée par l’entreprise Française **DARVA** ,elle est l’application officielle des assureurs Français. |
| **Fonctionnalités** | -Géolocalisation du lieu de l’accident.  -Prise de photos.  -La confirmation de la réclamation par SMS et un email avec la version PDF de la déclaration européenne d'accident. | -Pré-remplissage des informations personnelles de l’utilisateur.  -Aide au remplissage de la déclaration.  -Prise de photos.  -Géolocalisation du lieu de l’accident.  -Aide à la réalisation du croquis. |
| **Avantages** | -Elle est gratuite.  -Disponible sur Android et IOS.  -Facile et intuitive.  -Présente dans 42 pays.  -Disponible en plusieurs langues.  -Protection des données.  -La possibilité de suivre la réclamation sur son site <https://assis.to/#features> via le service «votre constat». | -Sa simplicité.  -Elle est gratuite.  -Disponible sur Android et IOS.  -Le constat peut être rempli avec un ou deux téléphones.  -Le remplissage du constat est simple et intuitif. |
| **Inconvénients** | Elle ne pourra pas être utilisée si un ou plusieurs des véhicules impliqués dans l’accident sont immatriculés et/ou assurés en dehors des 42 pays européens où Assisto est présent. | Elle ne pourra pas être utilisée :  -si l’accident a eu lieu en dehors de la France Métropolitaine ou des départements d’Outre-mer.  -si un ou plusieurs des véhicules impliqués dans l’accident sont immatriculés et/ou assurés en dehors de la France.  -si l’accident entraîne des dommages corporels, même légers. |

Tableau 1: Tableau comparatif des projets existants

* 1. Objectifs du projet

Notre projet vise la résolution de la problématique précédemment citée en offrant une l’aide à chaque étape du remplissage et en garantissant une interface ergonomique, simple et intuitive à utiliser, et aussi disponible sur différentes plateformes (Android et IOS), et accessible à tous les assurés marocains.

* 1. Solution proposée

Ce projet consiste à concevoir et développer une application mobile permettant de remplir un constat à l’amiable, et aussi effectuer un suivi de ce dernier après son remplissage, et finalement garder un historique des constats précédemment remplis qui peuvent être visualisée ou bien supprimés par l’utilisateur.

Notre application est destinée essentiellement à toute personne conduisant un véhicule d’immatriculation marocaine et assuré par un des assureurs marocains.

Cette application vise à rendre le remplissage d’un constat à l’amiable plus efficace en le simplifiant et l’accompagnant d’aide nécessaire à chaque étape ; ainsi diminuer le temps que prends cette opération en réalité sans oublier qu’il faut principalement soit attendre la présence d’un constateur soit essayer de se procurer le document dans les environs du lieu de l’accident.

Cette application offre la possibilité de remplir le constat à l’amiable en deux options : effectuer le remplissage par deux smartphones ou bien si les conducteurs ne disposent pas de l’application ils feront appel à un consateur qui pourra effectuer le remplissage directement sur son smartphone.

Conclusion

Tout au long de ce chapitre, nous avons mis notre projet dans son cadre général, à savoir la présentation du projet et ses objectifs, et nous allons spécifier dans le prochain chapitre les différents besoins auxquels doit répondre notre application.

1. Spécification des besoins

Introduction

Avant d’entamer la conception et la réalisation du projet, on est en grande obligation de passer par la définition des besoins fonctionnels et non fonctionnels des utilisateurs visés par ce projet ; puisque pour assurer la réussite de tout projet il est essentiel de se donner une vue claire sur les objectifs ainsi que les besoins à satisfaire à travers ce projet.

Alors, au cours de ce chapitre on va dégager les différents besoins ainsi que les fonctionnalités attendues de notre projet en définissant les différents cas d’utilisation et quelques scénarios qui expliquent ces cas.

* 1. Besoins fonctionnels

Notre application mobile peut être résumée en deux partie majeures la partie de remplissage du constat à l’amiable et celle du suivi de des constats précédemment remplis.

Au niveau de la partie du remplissage du constat à l’amiable, notre application doit mettre à la disposition des utilisateurs les fonctionnalités suivantes :

- L’utilisateur peut choisir entre deux options selon les cas :

1. Remplir le constat sur deux smartphones en même temps (si les deux conducteurs possèdent l’application).
2. Remplir le constat directement sur le smartphone du constateur (si les conducteurs ont fait appel à un constateur).

- L’utilisateur peut ajouter jusqu’à quatre témoins s’il y a des témoins.

- L’utilisateur peut consulter l’aide proposée à chaque étape du remplissage.

- L’utilisateur peut retourner à une étape pour effectuer des modifications si besoin.

- L’utilisateur peut annuler le processus de remplissage à n’importe quelle étape.

- L’utilisateur peut prendre une photo de l’accident.

- L’utilisateur peut dessiner un croquis de l’accident.

- L’utilisateur peut signer le constat directement sur son smartphone.

- L’utilisateur peut enregistrer son constat à la fin du remplissage et l’envoyer à son assurance.

Au niveau de la partie du suivi du constat à l’amiable, notre application doit mettre à la disposition des utilisateurs les fonctionnalités suivantes :

- L’utilisateur peut visualiser la liste des constats précédemment remplis sur l’application ainsi que leurs états.

- L’utilisateur peut consulter les données renseignées au niveau de chaque constat.

- L’utilisateur peut supprimer un constat de la liste des constats précédemment remplis *(à condition que le constat soit déjà traité).*

- L’utilisateur peut annuler la suppression d’un constat à partir de la corbeille.

* 1. Besoins non-fonctionnels

- Notre application doit être disponible sur différentes plateformes (iOS et Android).

- Notre application doit présenter des interfaces graphiques ergonomiques et élégantes.

- Notre application doit comporter de l’aide à chaque étape du remplissage.

- Notre application doit exploiter les données fournies par les utilisateurs en toute confidentialité et sécurité.

- Possibilité d’optimiser l’application plus tard.

* 1. Traduction des besoins en diagrammes de cas d'utilisation

Le diagramme des cas d’utilisation est un modèle simplifié du fonctionnement de l’application qui spécifie les acteurs qui l’utilisent et les services qu’il leur offre.

* + 1. Description des acteurs

Un acteur est l'idéalisation d'un rôle joué par une personne externe, un processus ou une chose qui interagit avec un système.

Dans notre cas, on peut identifier deux catégories d’acteurs : le conducteur ainsi que le constateur.

**Conducteur :** C’est l’acteur qui représente une des deux parties de l’accident ; il est donc la personne principalement concernée par le remplissage du constat à l’amiable vu qu’il permet de déterminer la responsabilité de chaque partie voir de chaque conducteur.

Notre application lui offre la majorité de ses fonctionnalités à savoir le remplissage du constat à l’amiable ainsi que le suivi de ce dernier.

**Constateur :** Il s’agit d’un employé de l’assurance automobile qui assure un des deux parties de l’accident ; il est chargé de se déplacer au lieu de l’accident et d’effectuer le remplissage du constat à l’amiable sur place.

Notre application lui offre la possibilité de remplir le constat à l’amiable directement sur son smartphone ; donc son smartphone sera son principal outil de travail dorénavant.

* + 1. Diagramme de cas d'utilisation

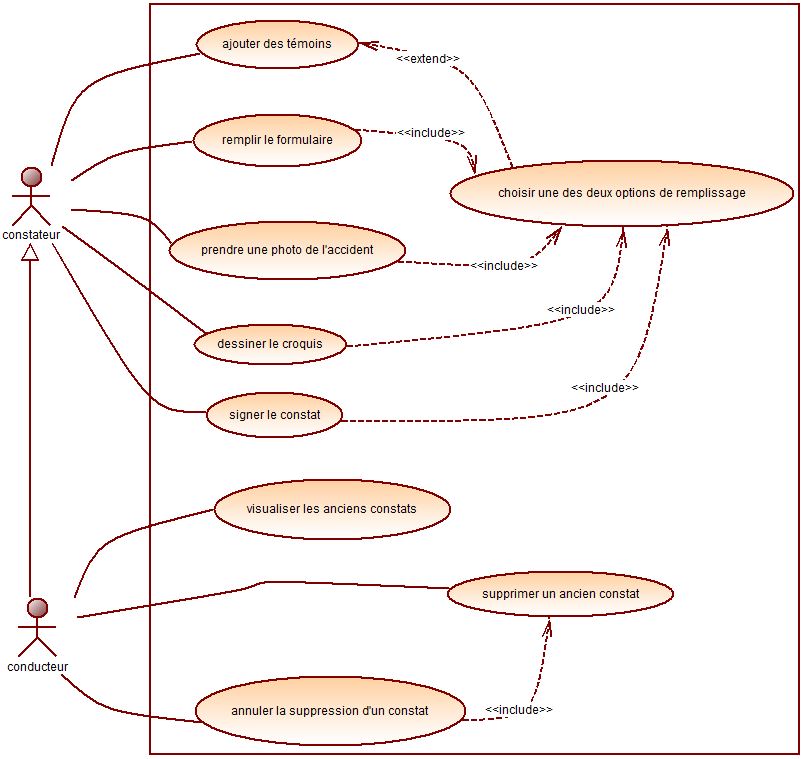


Figure 2 : Diagramme de cas d’utilisation.

Le tableau suivant explique les différents *cas d’utilisation* présentés dans ***la figure2*** :

|  |  |
| --- | --- |
| Cas d’utilisation | Description |
| **Choisir une des deux options de remplissage** | L’application offre deux différentes options de remplissage : une pour remplir à l’aide des deux smartphones des conducteurs et l’autre pour remplir directement sur le smartphone du constateur. |
| **Ajouter un témoin** | Le conducteur ou le constateur peuvent ajouter jusqu’à quatre témoins ,ou bien dépasser cette étape au cas où il n’y a pas de témoins. |
| **Remplir le formulaire** | Le formulaire fourni par l’application est un ensemble de champs à remplir ainsi que des cases à cocher , il est divisé en un nombre d’étapes qui sont toutes accompagnées par de l’aide spécifique à chacune. |
| **Prendre une photo de l’accident** | A l’aide de l’appareil photo proposé par l’application les conducteurs ou le constateur peuvent prendre des photos de l’accident et des dégâts. |
| **Dessiner le croquis** | Comme dans un constat à l’amiable classique, le constateur ou bien un des deux conducteur doit dessiner au niveau du constat un croquis illustrant l’accident et ses circonstances. |
| **Signer le constat** | A la fin du remplissage du constat les conducteurs doivent signer le constat ainsi que le constateur s’il est présent. |
| **Visualiser les anciens constats** | Les conducteurs ont la possibilité avant de créer un nouveau constat de consulter l’historique des constat précédemment remplis au niveau de l’application ainsi que leurs états actuels. |
| **Supprimer un ancien constat** | Si un des constats précédemment remplis au niveau de l’application a été traité, alors le conducteur peut le supprimer de l’historique. |
| **Annuler la suppression d’un constat** | Tous les constats supprimés sont gardés dans une corbeille accessible aux conducteurs ;c’est au niveau de cette corbeille que le conducteur peut annuler la suppression, donc elle va s’ajouter encore une fois à l’historique des constats. |

Tableau 2: Tableau de description des cas d’utilisation

Conclusion

Ce chapitre nous a permis de couvrir les différents besoins fonctionnels et non fonctionnels auxquels doit satisfaire notre application. Nous avons aussi détaillé ces besoins à travers des diagrammes de cas d’utilisation pour passer par la suite à la conception de notre application qui sera présentée dans le chapitre suivant.

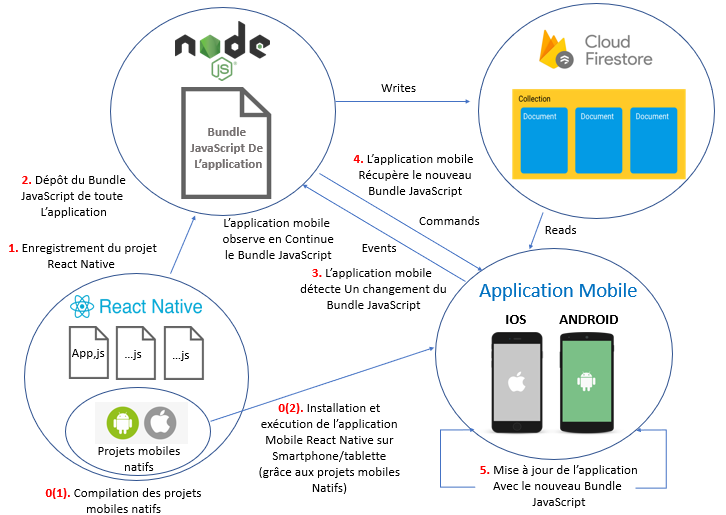
1. Architecture et conception

Introduction

Dans le présent chapitre, nous allons entamer une phase particulièrement importante du développement de l’application, Elle décrit la manière dont seront agencés les différents éléments d’une application et comment ils interagissent entre eux, Et qui constitue un pont entre la spécification et la réalisation, Elle conditionne la stabilité d’une application, son efficacité et sa pérennité Au Contraire, certaines applications peuvent connaître des faiblesses dues à une architecture mal pensée, pas ou plus adaptée au contexte, Nous allons donc présenter dans un premier temps la conception générale de notre application, puis la conception détaillée via les diagrammes de classes.

* 1. Conception générale

Sur ce graphe nous avons décrit le mode d’interaction ainsi que le fonctionnement de rechargement de notre application



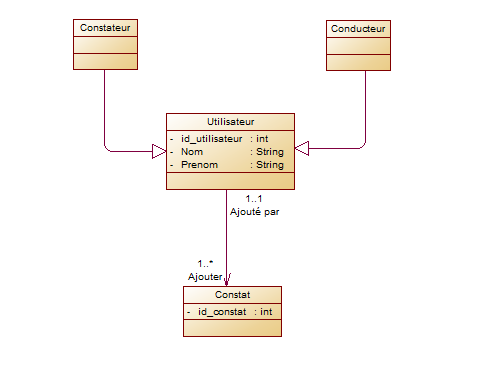
**Figure 3 :** Schéma de fonctionnement de l’application

* **0 (1)** : Lorsque vous allez lancer l'application sur un device, avant cela, les projets mobiles natifs vont être compilés.
* **0 (2)** : Puis les projets mobiles natifs vont être installés et exécutés sur le device. C'est à ce moment que la pastille de l’application va apparaître sur l'écran d'accueil du device et que l'application va être lancée.

En parallèle de ces deux étapes, un serveur Node.JS est lancé, et va héberger le code Javascript de notre application React Native. Notre application mobile, fraîchement installée, va observer ce code Javascript et être mise à jour automatiquement dès que le code change. Cela signifie que l'on va pouvoir continuer à développer et tester instantanément notre code, sans avoir à relancer l'application. Un bon point.

* 1. Conception détaillée
     1. Diagramme de classes

La **figure 4** représente le diagramme de classes des entités utilisées dans notre système :



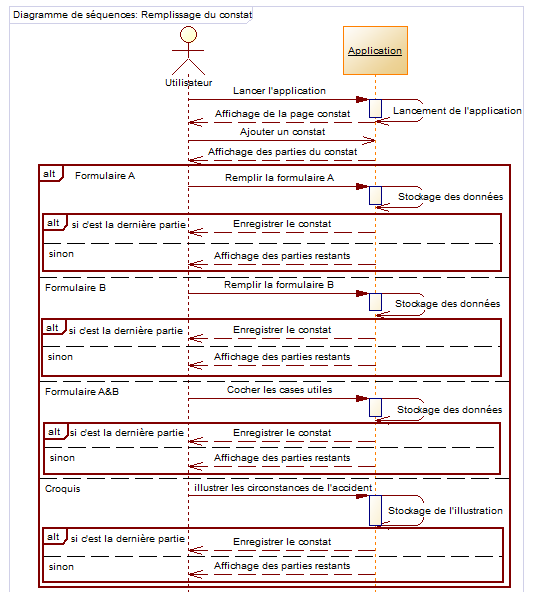
**Figure 4 :** Diagramme de classes

**-Utilisateur :** c’est la classe qui représente un utilisateur de notre application mobile qui est soit un constateur soit un conducteur.

**-Constat :** la classe constat désigne la formulaire remplit par l’utilisateur, c’est l’entité important de notre application.

* + 1. Digramme de séquences

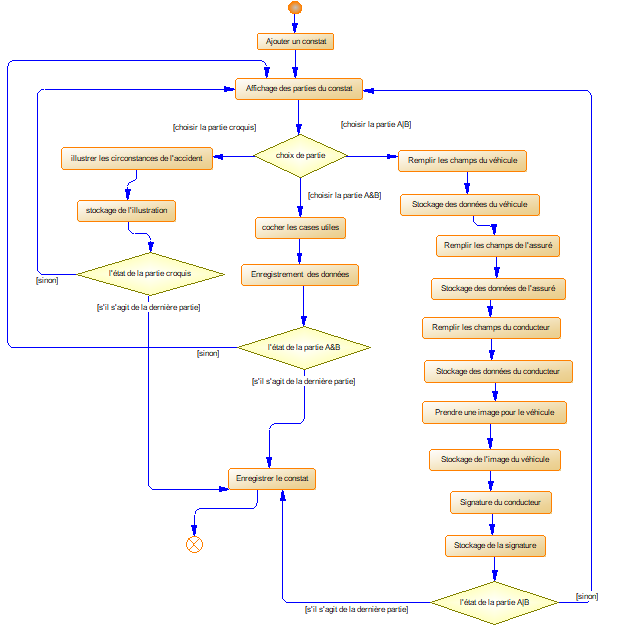
Le diagramme de la **figure 5** illustre le scénario complète de remplissage du constat à partir de lancement de l’application jusqu’à l’enregistrement du constat.



**Figure 5 :** Diagramme de séquences

* + 1. Digramme d’activité

Le diagramme de la **figure 6** illustre les activités de remplissage du constat qui en jeu l’utilisateur et l’application.



**Figure 6 :** Diagramme d’activité

Conclusion

Dans ce chapitre, nous avons modélisé le fonctionnement de l’application afin d’avoir une vue globale et simplifiée du système. Nous avons aussi détaillé les différents modules de l’application ce qui nous a permis d’organiser le travail et d’avoir une idée claire sur le travail à réaliser. Ce travail est décrit plus précisément dans le chapitre qui suit.

1. Réalisation et méthodologies

Introduction

Après avoir spécifier les besoins et fixer les objectifs de notre projet ainsi qu’avoir passer par la phase de conception, on arrive enfin à la phase finale qui n’est que la phase de réalisation qui se présente comme sujet principal de ce dernier chapitre.

Ce chapitre présente aussi les différentes méthodologies suivies durant cette phase de réalisation pour en fin de compte mener à bien notre projet et satisfaire les besoins de nos utilisateurs.

* 1. Réalisation

Dans ce paragraphe nous décrivons les différents outils et logiciels utilisés pour la mise en œuvre de l’application.

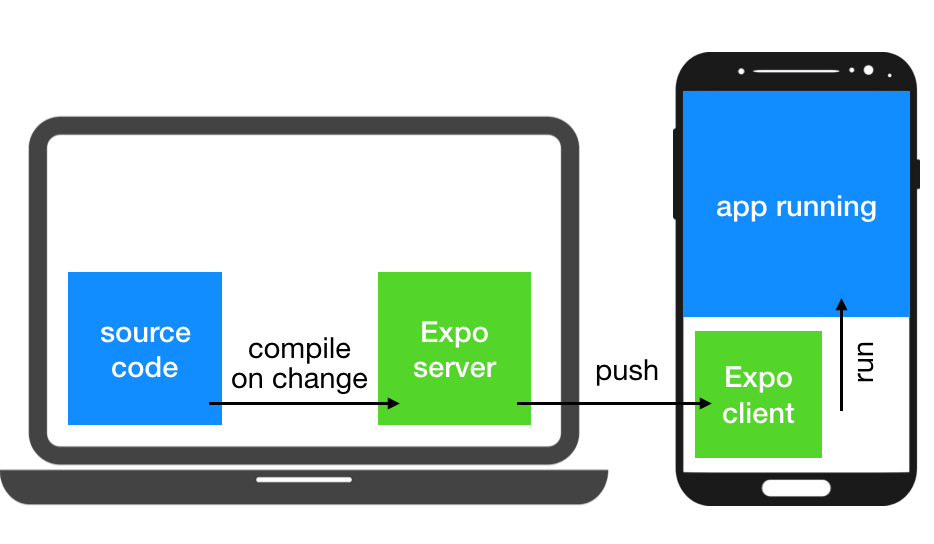
* + 1. Environnement logiciel

- Expo CLI (version: 3.11.9):

Expo est un Framework et une plateforme pour les applications React universelles. Il s'agit d'un ensemble d'outils et de services construits autour de React Native et de plateformes natives qui vous aident à développer, créer, déployer et itérer rapidement sur iOS, Android et des applications Web à partir de la même base de code JavaScript / TypeScript.

Expo CLI est un outil pour développer des applications avec Expo. En plus de l'interface de ligne de commande (CLI), il possède également une interface utilisateur graphique (GUI) basée sur le Web qui apparaît dans le navigateur Web lorsque on démarre notre projet.

Il nous a été bien bénéfique de s’appuyer sur cet outil pour développer notre application durant la période où notre application ne comportait pas encore de composants natifs.

****

*Figure 7 :* Le développement en temps réel avec expo.

-Adobe Illustrator CS6 (version: 16.0):

Adobe Illustrator est un logiciel de de création graphique vectorielle. Ce logiciel est adapté aussi bien à la création de document papier qu'à celle d'illustrations pour Internet (logos, affiches, etc.).

Il nous a permis de créer le logo de notre application ainsi que l’image d’arrière-plan de la page d’accueil.

-Adobe Photoshop CS6 (version: 13.0):

Adobe Photoshop est un logiciel de traitement d'image. Ce logiciel permet de créer et de modifier des images.

Il nous a été utile pour effectuer des retouches sur le logo de l’application ainsi que pour ajuster certaines icones et photos utilisées dans l’application.



*Figure 8 :* Le logo créé pour l’application.

-Microsoft Visual Studio Code (version : 1.44.2) :

Visual Studio Code est un éditeur de code source léger mais puissant qui s'exécute sur votre bureau et est disponible pour Windows, MacOs et Linux. Il est livré avec un support intégré pour JavaScript, TypeScript et Node.js et possède un riche écosystème d'extensions pour d'autres langages (tels que C ++, C #, Java, Python, PHP, Go) et des runtimes (tels que .NET et Unity).

-Android Studio (version: 3.5):

Android Studio est un environnement de développement pour développer des applications mobiles Android. Il est basé sur IntelliJ IDEA et utilise le moteur de production Gradle.

Il nous a été utile lors de la phase après avoir éjecter l’application par expo ; puisqu’il nous a permis de compiler notre application à chaque fois qu’on ajoutait un composant natif à notre application et la tester directement sur nos smartphones.

-Sybase PowerAMC (version : 15.1.2850) :

PowerAMC est un logiciel permettant la modélisation d'un grand nombre de processus et de modèles informatiques.

Il nous a accompagné tout au long de la phase de conception ; ainsi les différents diagrammes contenus dans ce rapport ont été réalisé à l’aide de ce logiciel.

* + 1. Environnement technique

-Node JS (version: 12.15.0):

Node.js est une technologie qui permet d’écrire du JavaScript coté serveur pour réaliser des applications web. Node JS est particulièrement adapté aux projets ayant besoin de temps réel. Il supporte nativement des technologies comme les web sockets, ce qui permet de créer facilement des applications réactives.

-React Native (version : 0.61.4) :

React Native est un Framework d'applications mobiles open source créé par Facebook. Il est utilisé pour développer des applications pour Android, iOS en permettant aux développeurs d’utiliser React avec les fonctionnalités natives de ces plateformes.

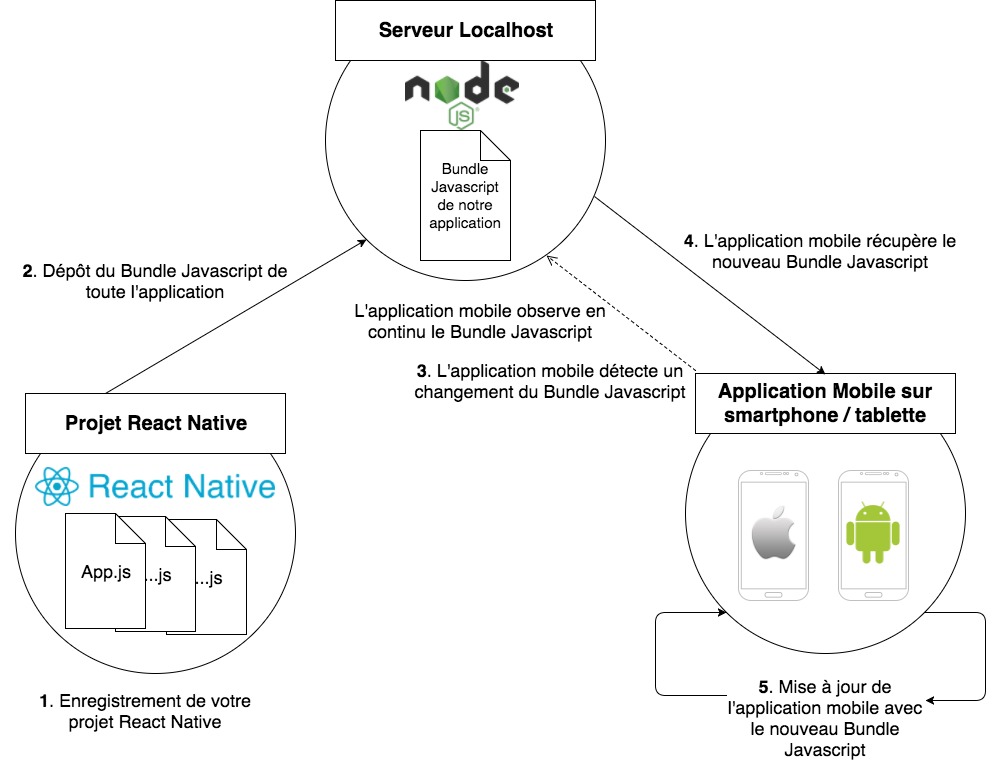


Figure 9 : Le rôle de NodeJS durant le développement.

-Firebase (version : 7.14.3) :

Firebase est un ensemble de services d'hébergement pour n'importe quel type d'application (Android, iOS, JavaScript, Node.js, Java, Unity, PHP, C++ ...). Il propose d'héberger en NoSQL et en temps réel des bases de données, du contenu, de l'authentification sociale (Google, Facebook, Twitter et GitHub), et des notifications, ou encore des services, tel que par exemple un serveur de communication temps réel.

Firebase et plus précisément Firestore nous a permis de stocker les données saisis par les utilisateurs ainsi que les photos de l’accident et les croquis et les signatures. Il nous a notamment servi pour synchroniser le remplissage des du constat par deux smartphones à l’aide de sa base de donnée en temps réel, la figure suivant illustre ce processus :

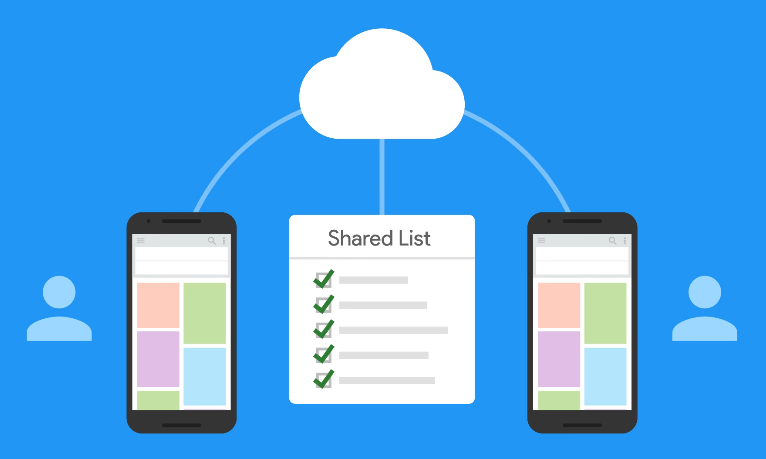


Figure 10 : Remplissage du constat par deux smartphones.

* + 1. Travail réalisé

* 1. Description de l’organisation du projet

* + 1. Description des méthodologies utilisées

* + 1. Description de l’organisation du travail en équipe

Conclusion

1. Conclusion et perspectives

Bibliographie

1. Code source