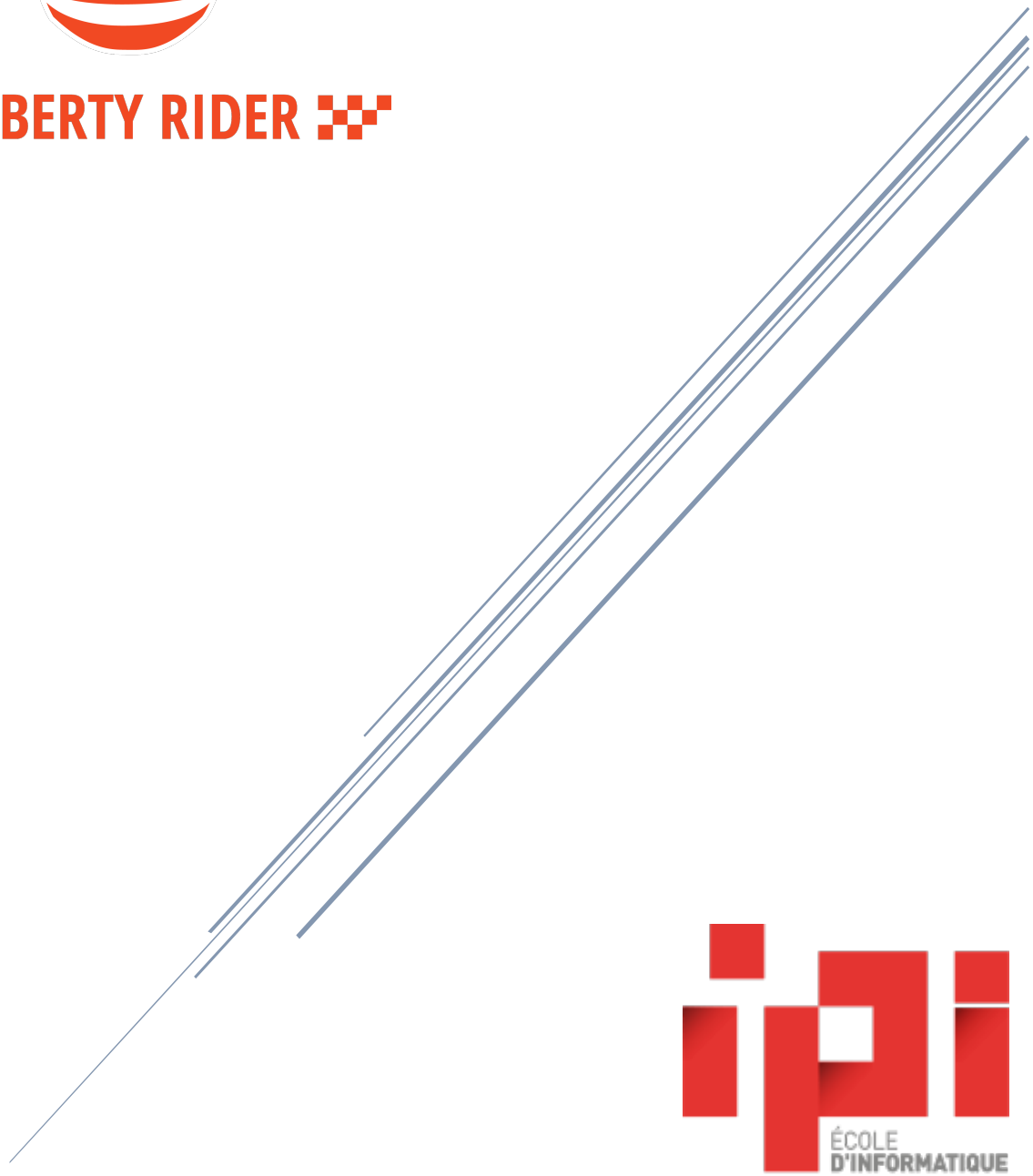
RAPPORT D'ALTERNANCE

Titre de Chef de projet ingénierie logicielle



Institut de Poly-Informatique, Toulouse

Julien Sergent

PAGE A SUPPRIMER

LEGENDE

Lorem ipsum	Texte à revoir ou à supprimer
[image:label]	Image à insérer, décrite selon le label
[annexe:numero:label]	Insérer une annexe complétant la section

Remerciements

LIBERTY RIDER

ÉCOLE D'INFORMATIQUE
Certificateur de compétences

Grille de compétences

Gérer les processus et la qualité

- Pratiquer des audits en respectant une méthode.....
- Être capable de gérer et d'optimiser les procédures existantes en respectant les normes ISO 20 000 .....
- Formaliser des procédures et garantir leur respect en respectant les normes ISO 900 .....
- Architecturer et gérer un réseau d'entreprise.....
- Concevoir et proposer des solutions innovantes de reconstruction des processus.....

Gérer les ressources du projet

- Prévoir des ressources humaines, recruter des collaborateurs .....
- Manager une équipe, évaluer les collaborateurs.....
- Prévoir et trouver les moyens logistiques nécessaires au projet .....
- Maîtriser les aspects juridiques des contrats à passer.....

Gérer le budget des projets

- Élaborer, faire valider un budget .....
- Contrôler les tableaux de bord de suivi budgétaire.....
- Savoir rendre compte de l'état d'avancement du budget.....

Communiquer

- Élaborer un cahier des charges ou y répondre..... 19
- Organiser des réunions, capacité à négocier 19
- Mobiliser l'équipe pour favoriser l'avancement du projet, savoir pratiquer une écoute active et gérer les conflits.....
- Rédiger des documents et des présentations en français et en anglais .....

Manager le projet

- Être capable d'utiliser une méthode Agile .....
- Transmettre les informations et accompagner le changement .....
- Prévoir les charges de travail et le planning de réalisation, lancer l'ordonnancement et assurer son suivi .....
- Contrôler les écarts de délai et le respect des contraintes .....
- Synthétiser les indicateurs des tableaux de bord, prendre ou faire prendre des décisions pour garantir les bonnes fins du projet .....

Technique métiers spécifiques

- Posséder des compétences métier spécifiques associées aux projets à gérer.....
- Être capable de maintenir ses compétences ou connaissances métier.....

Glossaire

Entreprise

CEO : Chief Executive Officer

R&D : Recherche et Développement

B2B : Business to Business

B2C : Business to client

CPO : Chief Product Owner

CMO : Chief Marketing Officer

IMA : Inter Mutuelles Assistance

Propre à l'entreprise

FLOOZ : Système monétaire virtuel de l'application n'ayant aucune valeur en euro

Technique

BACKEND : Partie d'une application qui concerne le serveur, l'architecture, base de données, etc. Tout ce qui n'est pas visible ni tangible par le client

FRONTEND : Partie d'une application qui concerne la partie graphique, l'interface utilisateur

API : Application Programming Interface. Interface disponible pour le client (mobile, application web, etc.) afin de pouvoir interagir avec le serveur.

FULLSTACK : Backend + frontend

JAVASCRIPT : Langage de programmation

SWIFT : Langage de programmation

JAVA : Langage de programmation

DEADLINE : Date limite / Date butoir (souvent appliquées pour parler d'une date limite pour rendre/terminer quelque chose)

NodeJS : Plateforme logicielle permettant d'exécuter du JavaScript côté serveur

Firebase : Plateforme logicielle en ligne simplifiant toute la partie base de données pour un projet

SGBD : Système de gestion de base de données

Outils

Slack : Outils permettant de communiquer sur différents canaux avec les membres de notre équipe

CircleCI : Outils permettant d'exécuter des commandes et d'effectuer un déploiement continue

GitHub : Plateforme en ligne permettant de versionner son code avec une stratégie git

Sommaire

Introduction

Dans le cadre de mon mastère au sein de l'école IPI, j'ai effectué deux années consécutives en alternance, dans deux entreprises différentes. Lors de mon mastère 1, j'ai effectué une année d'alternance au sein du groupe Capgemini, puis au sein de Liberty Rider pour mon mastère 2. J'ai pu changer d'entreprise car Capgemini n'avait pas désiré me prendre deux années consécutives en alternance, ce qui m'a finalement permis de pouvoir découvrir le monde de la startup en plus de celui de la SSII, afin de pouvoir effectuer un choix plus adapté à mon fonctionnement lors de la prise de mon premier emploi.

J'ai donc effectué une année d'alternance chez Liberty Rider, où j'ai occupé le poste de développeur web Fullstack. Durant cette année d'alternance, j'ai pu contribuer à plusieurs projets, qui m'ont permis d'approfondir mes connaissances techniques, mes compétences à travailler en équipe et à gérer un projet.

Les débuts ont été quelques peu difficiles, à commencer par comprendre comment fonctionnait l'entreprise, apprendre l'environnement global de l'application et savoir prendre les bonnes décisions. Rapidement, j'ai pu m'adapter à toutes ces contraintes et nouveautés afin de pouvoir apprendre pleinement tout ce dont Liberty Rider avait à m'offrir, mais aussi de pouvoir contribuer et faire évoluer cet environnement.

Liberty Rider

LIBERTY RIDER

L'histoire

Liberty Rider prit naissance un soir, lorsque Emmanuel Petit, son CEO et porteur originel de l'idée, rentrait chez ses parents en moto, en empruntant des chemins sinueux, dangereux et peu fréquentés. Il s'est alors dit que s'il venait à chuter à cet endroit, il aurait de grandes chances de ne pas s'en sortir. L'idée lui est donc venu de créer une startup, avec trois de ses amis, à savoir : Julien LE, Martin D'Allens et Jérémie Fourmann.

Liberty Rider était donc né, à la tête quatre fondateurs, tous prêt à porter le projet jusqu'au bout.

Les débuts de Liberty Rider ont été quelques peu houleux, des difficultés à trouver des finances, ce qui était le rôle de Emmanuel, de définir le périmètre de départ à réaliser pour avoir une première version stable, prévoir les évolutions, étudier le marché et les concurrents.

Les débuts de la startup se déroulèrent dans de petits locaux, les premiers tests ont même été effectués chez eux, sur leur canapé.

A force de volonté et de persévérance, la startup a pu intégrer les locaux de AtHome, un incubateur sur Toulouse. C'est à partir de ce moment que tout prit forme. Les premiers stagiaires et employés ont été recrutés, les premiers rôles ont été attribués, c'était donc les prémisses de la jeune startup comme on a la connaît actuellement.

La startup

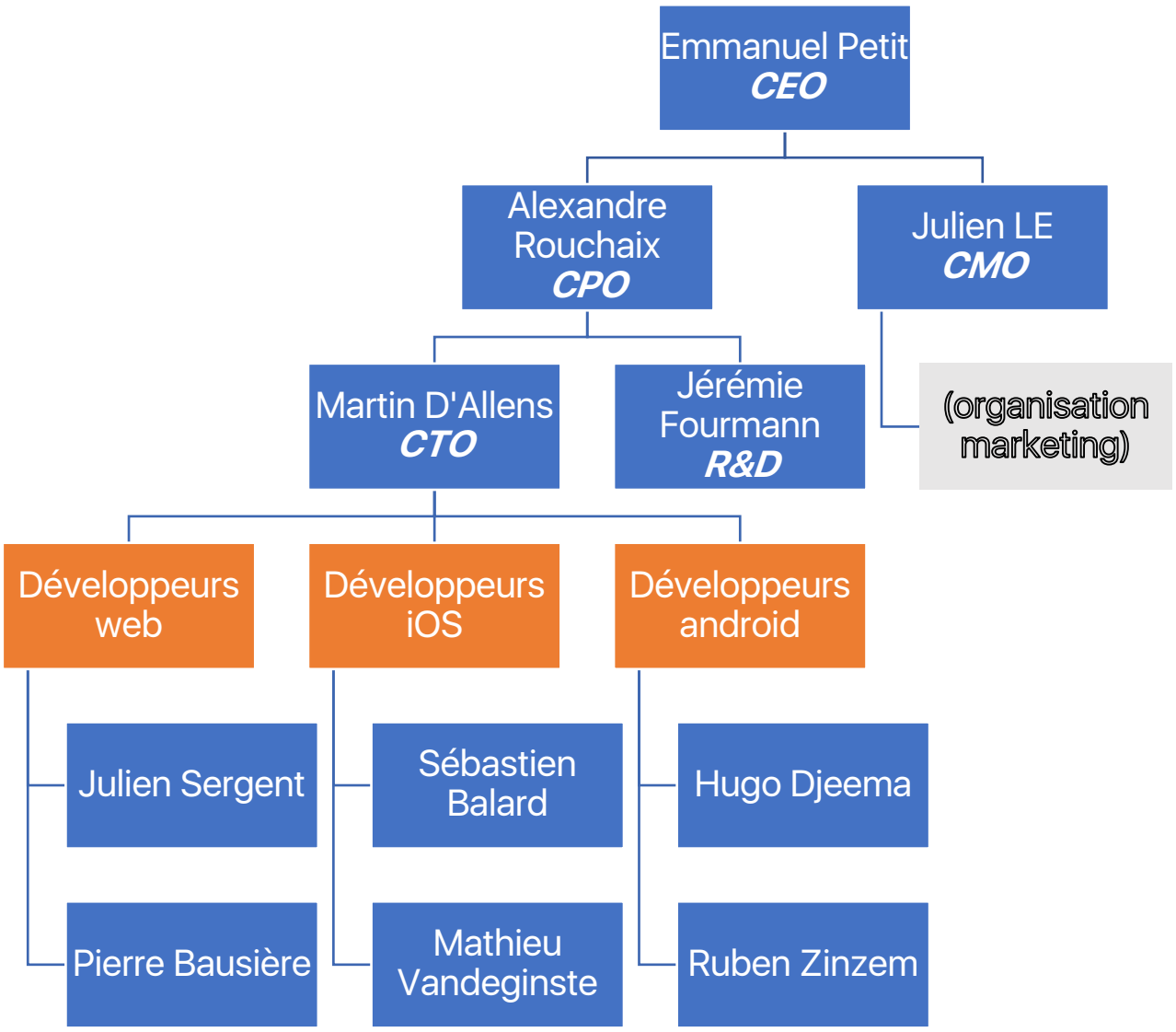
Effectif et organisation

Liberty Rider est actuellement composé de dix-sept personnes ; dont le CEO, trois développeurs web, quatre développeurs mobiles, un CPO, un responsable R&D, quatre personnes chargées de la partie B2C, deux personnes chargées de la partie B2B et un happiness officer.

La startup est répartie en trois grands pôle d'activités :

- Le secteur développement (regroupe les développeurs web, mobile et R&D)
- Le secteur commercial / client (regroupe les commerciaux B2B et B2C)
- Le secteur de l'entreprise elle-même (regroupent le happiness officer, le CEO)

Hiérarchie


Emmanuel Petit
CEO
Alexandre Rouchaix
CPO
Julien LE
CMO
Martin D'Allens
CTO
Jérémie Fourmann
R&D
(organisation marketing)
Développeurs web
Développeurs iOS
Développeurs android
Julien Sergent
Sébastien Balard
Hugo Djeema
Pierre Bausière
Mathieu Vandeginste
Ruben Zinzem

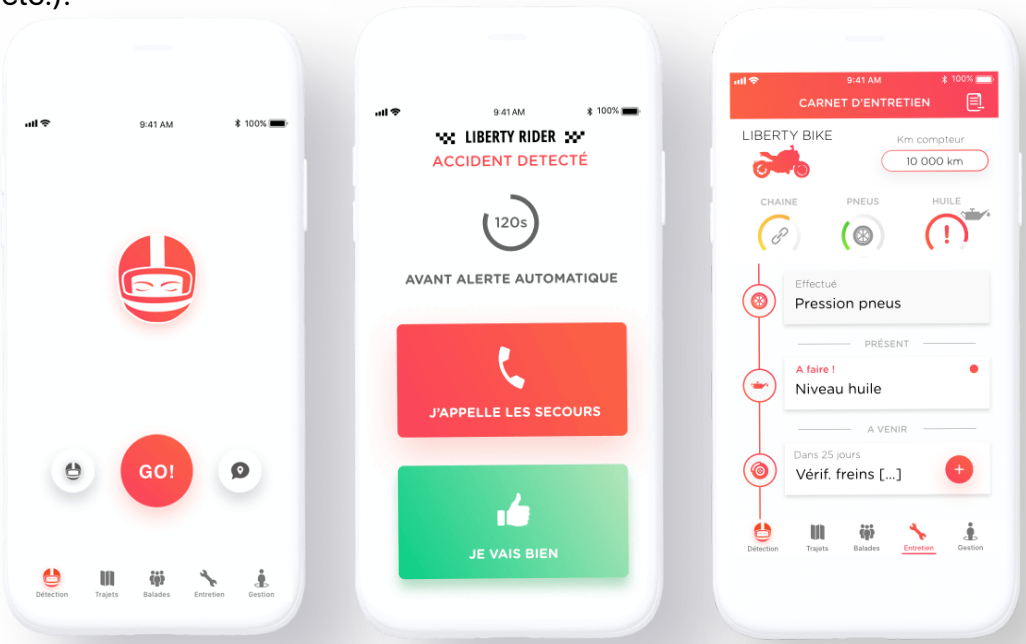
L'application

Liberty Rider est à la fois la société mais aussi l'application, cette application est donc le cœur de métier de la startup.

L'objectif principal de l'application est la détection de chute à moto. Pour ce faire, un motard peut donc télécharger gratuitement sur n'importe quel store (App Store, Play Store), s'inscrire, remplir quelques informations sur son profil et il sera enfin en capacité de pouvoir démarrer une session durant laquelle Liberty Rider sera son ange gardien. Au démarrage de la session, un sms est envoyé à certains contacts du motard (choisi par ses soins), leur indiquant qu'il vient de démarrer une session, un lien de suivi de trajet sur le web leur est fourni dans le message, ils peuvent donc observer tout ce qu'il se passe pendant la session.

En cas d'accident détecté par l'application, l'équipe de Liberty Ride est immédiatement alertée et prévient les secours, en indiquant la position GPS de l'accident, afin de pouvoir sauver le motard.

Cependant, l'application n'a pas pour seule fonctionnalité la détection d'accident, elle permet aussi de pouvoir gérer le carnet d'entretien de sa moto, être alerté quand il est nécessaire de faire la vidange, ou tout autre changement nécessaire, et il est aussi possible de pouvoir organiser des balades avec plusieurs motards et ainsi pouvoir tous être visible sur le même trajet. Afin de rendre l'application plus attractive, un système de ludification a été mis en place, à savoir des flooz à gagner lors de chaque voyage, lors du remplissage intégral du profil, de participation à des balades, être organisateur de balades, etc. Tous ces flooz permettront au motard, via l'application, de pouvoir commander des cadeaux sur la boutique Liberty Rider (des casques, mugs, blousons, etc.).



Vision, mission et ambition

Le projet Liberty Rider est basé sur un secteur nouveau, avec peu de concurrents, mais aussi où le marché n'est pas encore à l'écoute ni habitué à voir naître ce type d'application dans leur écosystème. C'est pourquoi la vision de la startup est de devenir le principal leader dans le domaine de la détection de chute à moto, mais aussi dans le domaine plus général de la moto et notamment de la sécurité.

Sa mission est donc de savoir faire éclore ce nouveau marché, convaincre la grande majorité des motards (français dans un premier temps) d'installer l'application afin d'assurer leur sécurité mais aussi de pouvoir rassurer leurs proches. Tout ceci s'accompagnera d'une croissance de communauté, qui permettra de pouvoir faire naître de nouvelles fonctionnalités, comme le partage et proposition de balades au sein de la communauté.

Aujourd'hui l'application ne fonctionne qu'exclusivement en France, pour des raisons de proximité avec les secours et d'échanges avec les utilisateurs, mais il est envisagé, à moyen terme, d'une expansion en Europe de l'application, grâce notamment à un nouveau partenariat avec IMA (Inter Mutuelles Assistance) qui permettrait à Liberty Rider de pouvoir agir en termes d'intervention des secours en cas d'accident détecté dans des pays d'Europe.

Clients et partenaires

Liberty Rider est une application gratuite destinée aux particuliers, elle repose sur un modèle B2C, cependant comme le marché n'est pas encore pleinement ouvert pour ce nouveau type d'application, Liberty Rider collabore avec des partenaires déjà présents sur différents types de marchés de la moto (Honda, Scorpion, GS27, etc.).

De cette manière, la startup peut donc utiliser les données récoltées auprès de leurs utilisateurs pour les proposer à ses partenaires afin que ces derniers puissent mieux cibler leur clientèle, promouvoir des produits aux bonnes dates et ainsi optimiser leurs campagnes publicitaires.

La startup s'entoure aussi d'investisseurs, lui permettant de pouvoir augmenter ses effectifs régulièrement et assurer l'augmentation de charge des nouveaux utilisateurs qui s'accompagne de plus de bugs, de mécontentement, d'accidents et de nouvelles fonctionnalités à développer.

Tout ceci engendre une importante charge de travail pour les développeurs, qui se retrouvent submergés de travail et il devient donc nécessaire de recruter plus d'effectif.

Contexte et problématiques

Gestion de projet

L'organisation au sein d'une startup est bien souvent chaotique, et Liberty Rider n'y échappe pas.

A mes débuts dans la startup, j'ai commencé par le projet *Gamification*, le but du projet était de fournir une boutique dans l'application, où le motard pouvait y dépenser ses flooz gagner durant ses sessions avec l'application, le tout afin de rendre l'application plus ludique, et favoriser la rétention des utilisateurs. Ce projet était malheureusement dirigé par une personne dont les compétences n'était pas la gestion de projet, ce qui est bien souvent le problème dans une startup, on ne peut pas se permettre d'embaucher des compétences à chaque besoin. J'ai rapidement ressenti les problématiques engendrées dès les débuts du projet : manque de spécifications qui nécessitait des allers-retours constants avec le chef de projet afin de connaître le cadre des fonctionnalités à développer, manque de connaissances globales qui engendrait un retard conséquent sur les deadlines, et bien souvent des parties de codes à totalement refaire car elles n'étaient pas conformes aux attentes du projet.

En ce qui concerne l'organisation au sein de l'équipe de développeurs, on s'organisait le plus souvent par oral, en mettant le plus possible de comptes rendus sur Slack, tout en essayant de se mettre d'accord entre les équipes web et mobiles.

Il est donc convenable de dire qu'au début de mon alternance au sein de Liberty Rider, la gestion de projet était quelque peu empirique, mais fort heureusement tout ceci à rapidement évoluer.

Environnement technique

Les solutions techniques adoptées chez Liberty Rider étaient les suivantes :

[image:technologies logos]

WEB : NodeJS, Express, JavaScript, Firebase

MOBILE : Java, Swift

DEVOPS : CircleCI, GitHub, Asana

ARCHITECTURE : Amazon Web Services (AWS)

La startup avait donc fait le choix d'utiliser des technologies modernes, mais tout ceci n'était pas sans contrepartie, puisque en effet l'utilisation de Firebase a engendré de lourdes problématiques quant à la manière de pouvoir traiter les données utilisateur (filtrer, réorganiser, etc.), il est même devenu impossible de pouvoir récupérer certains nœuds de données sous crainte de faire tomber la base de données pour dix minutes, rendant l'application totalement inutilisable.

De plus, la maintenabilité et l'évolutivité étaient parfois compromises dues aux dettes techniques trainantes. Tout ceci sera rapidement remis en question et de lourds projets de migration viendront réorganiser le système.

Nouvelle version de détection d'accidents

Contexte et objectifs

Liberty Rider est avant tout une application de détection d'accident, son but est donc d'être l'application la plus performante possible sur ce sujet, et pour ce faire des algorithmes complexes doivent être mis en place.

J'ai donc commencé chez Liberty Rider sur ce premier projet, la nouvelle version de détection d'accident. Ce projet avait pour but d'améliorer les contraintes et problématiques présentent, à savoir des fausses détections, ou parfois même des non détections alors qu'il y avait accident. Raison supplémentaire, la startup était en passe de concrétiser un partenariat avec IMA, qui serait en charge d'intervenir sur le lieu de l'accident une fois que nous leur aurions transmis l'alerte de l'accident et la localisation.

Les nouveaux enjeux de l'entreprise étaient donc importants pour cette dernière, car il s'agit du cœur même de l'application qui en fait sa réputation, mais aussi de la conséquence directe de leur première levée de fond ayant comme objectif de pouvoir en avoir d'autres par la suite.

D'autres objectifs étaient attendus à l'issu de ce projet. Une nouvelle architecture du code côté serveur, afin de rendre hautement disponible l'API d'accident, qui est un des points clés de l'application et qui ne peut pas se permettre d'être indisponible pour une raison quelconque sous peine de rendre l'application privée de sa fonction première et par conséquent l'incapacité à pouvoir secourir des motards accidentés.

Un autre objectif de ce projet était d'amorcer la migration et l'insertion d'un nouveau backend, afin de commencer à se détacher au fur et à mesure de Firebase. Il était important de garder à l'esprit que Firebase coûtait 700€/mois, ce qui n'est pas négligeable quand on est une startup, mais aussi du fait de son manque de souplesse et de prise de contrôle, il provoquait des problèmes de performance sur tout le réseau de l'application car certaines des requêtes dans l'application (par exemple la possibilité à un utilisateur de pouvoir visionner tous ses anciens trajets) entrainé un ralentissement de Firebase du fait d'un traitement de beaucoup données, cette problématique avait pour conséquence de rendre l'application indisponible pendant dix minutes environ, un problème majeur.

Liberty Rider étant dans les locaux de l'incubateur AtHome, au milieu de beaucoup de personnes et d'autres startup, nous étions souvent dérangés par le milieu de travail,

afin de pouvoir mener à bien ce projet et pouvoir être concentré pleinement dessus, nous avons décidé de partir une semaine, une dev week, dans une maison isolé à Pau. Cette semaine avait donc pour but de dégrossir le projet, en produire les spécifications et commencer à les implémenter afin d'obtenir une première version bêta sur mobile.

Travail R&D en amont

Comme expliqué dans le chapitre précédent, la détection d'accident est le cœur de l'application, et n'est pas une des parties les plus simple à réaliser. En effet il s'agit d'algorithmes complexes visant à analyser correctement les signaux envoyés par le téléphone dans un but de détecter et reconnaître un vrai accident d'un faux.

Le problème est que de tels algorithmes requièrent un important travail préparatoire, il a donc fallu réaliser des recherches, préparer tous les aspects des nouveaux algorithmes, anticiper les contraintes techniques, et rédiger des documentations techniques afin de simplifier l'implémentation pour les développeurs.

Ces travaux de recherches ont été effectués pendant une année avant leur implémentation définitive, des tests en amont ont été réalisés pour ajuster les paramètres des algorithmes et les améliorer.

Jérémie Fourmann, responsable R&D et co-founder, était en charge de la réalisation des travaux.

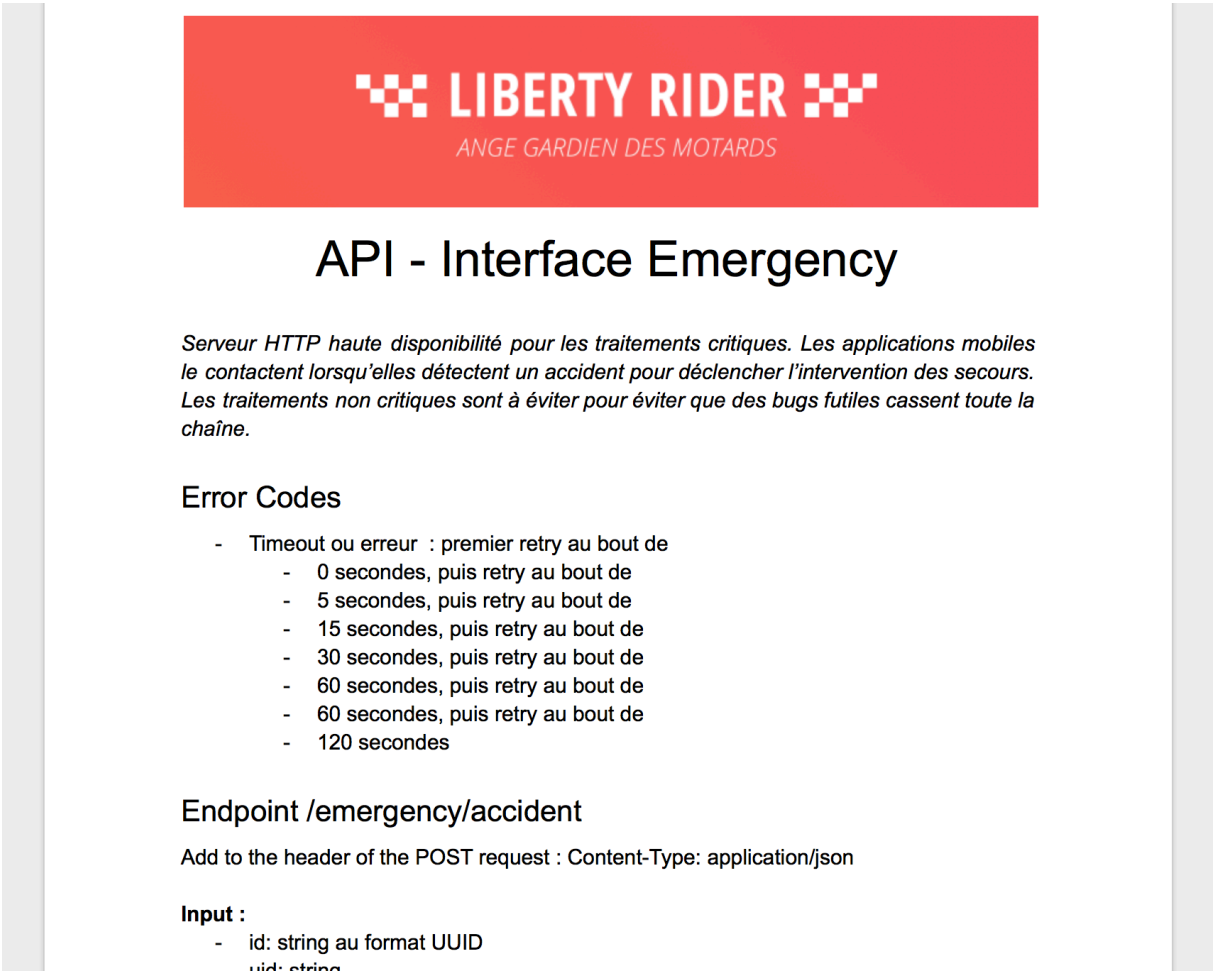
Point technique : Les recherches ont été réalisés à l'aide du langage de programmation Python.

Spécifications

Rédaction

Avant de commencer le projet, et afin de mieux le réaliser, nous avons décidé de rédiger des spécifications, pour mieux prévoir tous les aspects de ce dernier, les éventuels points bloquants, le temps que nous allions y consacrer et des documents en anglais et français (anglais pour la partie technique, axés développeurs, et français pour la partie commerciale) permettant de recenser la manière dont serait menés les travaux, les temps attribués aux différentes tâches, et les différents comportements du nouvel algorithme.

Pour mieux réaliser ces spécifications, nous avons donc effectué plusieurs réunions, chacune avait pour but de mieux prévoir toutes les frontières de la nouvelle version de détection d'accidents (ce qui allait devoir être migré, modifié et ajouté), et aussi se mettre d'accord sur les aspects techniques, et comment nous allions procéder.
J'ai donc assisté et animé des réunions de planifications, contribué à rédiger les spécifications du cahier des charges, ainsi que savoir discuter avec l'équipe des éventuels conflits, problèmes, que nous aurions pu rencontrer lors du développement de la solution.



LIBERTY RIDER

ANGE GARDIEN DES MOTARDS

API - Interface Emergency

Serveur HTTP haute disponibilité pour les traitements critiques. Les applications mobiles le contactent lorsqu'elles détectent un accident pour déclencher l'intervention des secours. Les traitements non critiques sont à éviter pour éviter que des bugs futiles cassent toute la chaîne.

Error Codes

- Timeout ou erreur : premier retry au bout de
  - 0 secondes, puis retry au bout de
  - 5 secondes, puis retry au bout de
  - 15 secondes, puis retry au bout de
  - 30 secondes, puis retry au bout de
  - 60 secondes, puis retry au bout de
  - 60 secondes, puis retry au bout de
  - 120 secondes

Endpoint /emergency/accident

Add to the header of the POST request : Content-Type: application/json

Input :

- id: string au format UUID
- uid: string

Aperçu du document des spécifications de l'API du serveur en charge de gérer les accidents (voir Annexe 2)

Architecture

Une des résultantes des spécifications est l'architecture des services web exposés par les serveurs de Liberty Rider.

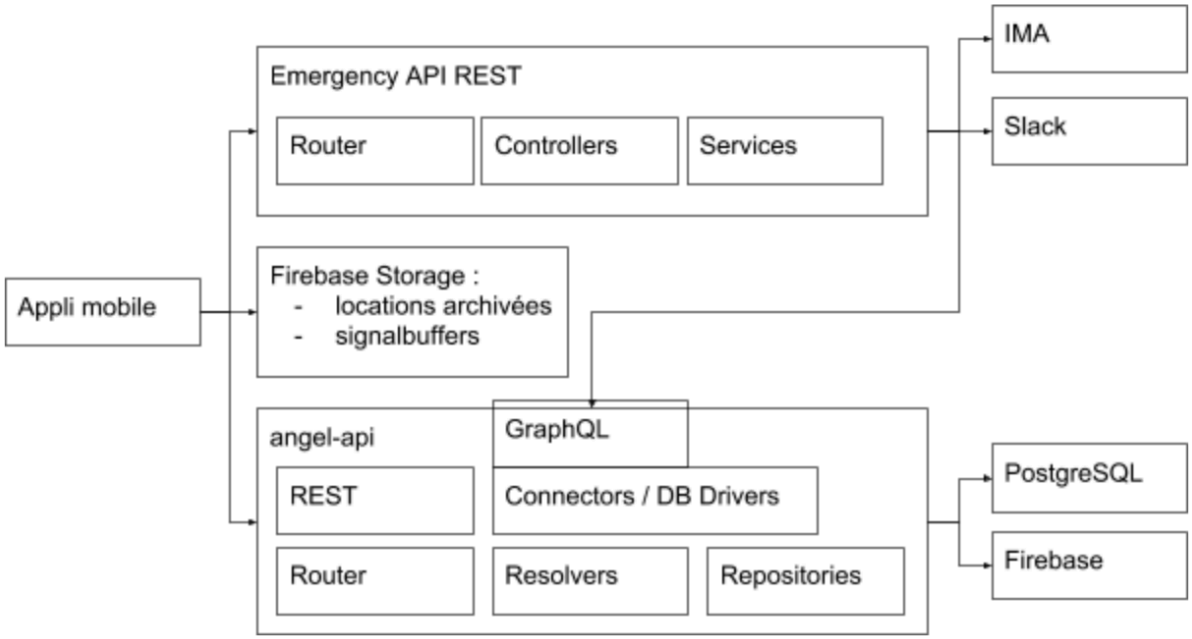


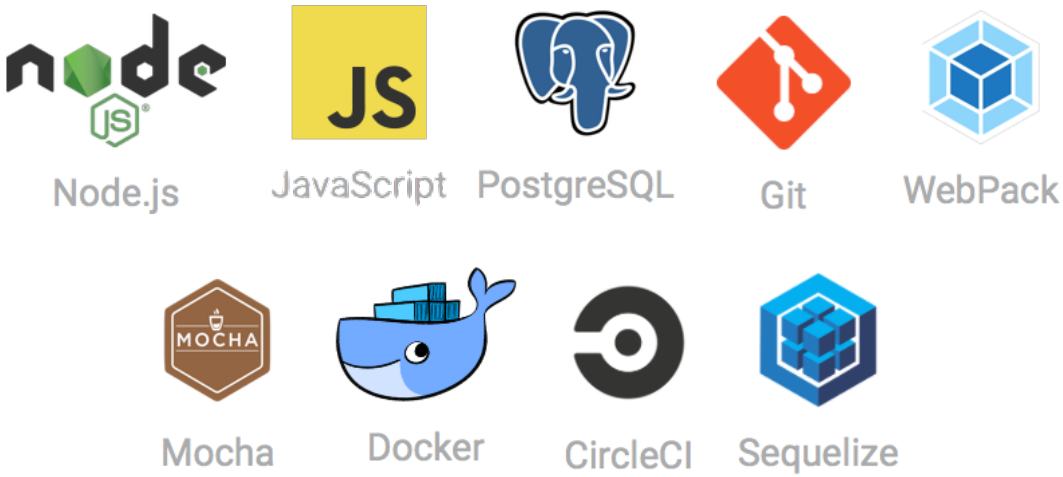
Schéma de l'architecture des différents services web exposés par les serveurs de Liberty Rider

Cette architecture est l'issue de plusieurs réunions de l'équipe web.

Dans le précédent schéma on représente donc les clients qui utilisent les services web par la cadre « Appli mobile ». Ces clients vont pouvoir consommer les services qui leur sont exposés, à savoir la partie accident avec le serveur « Emergency », les services globaux de l'application avec le serveur « angel-api », et tout ce qui concerne le stockage d'images sera disponible avec le service « Firebase Storage ». On remarque également qu'en sortie du schéma, nous avons « PostgreSQL », qui est un SGBD (Système de gestion de base de données), et « Firebase », qui est un système de base de données temps réel. Ces deux sorties, qui font le même travail, sont la preuve de la migration du système de Liberty Rider, cet aspect sera présenté en détails dans la présentation du projet **Migration et nouvelles fonctionnalités**.

Environnement technique

L'environnement technique choisi pour ce projet est le suivant :



L'environnement présenté ci-dessus concerne uniquement l'environnement web choisi par l'équipe web.

Nous avons décidé de garder quelques éléments déjà utilisés dans des projets de Liberty Rider (à savoir Docker, CircleCI, JavaScript, WebPack et Git). En ce qui concerne les nouveaux éléments techniques ajoutés (Sequelize, Mocha,et PostgreSQL), ils ont été choisi de manière élaboré. Leur choix provient de leur popularité au sein de la communauté des développeurs ; Sequelize est le nouvel ORM (Object-Relational Mapping) de tous les systèmes de Liberty Rider, du fait de sa popularité au sein de la communauté des développeurs ; PostgreSQL est un SGBD fiable et robuste, Mocha est un outil permettant d'exécuter des tests écrit avec le langage JavaScript.

Les raisons de ces choix sont dans un but précis. Dans un premier objectif, harmoniser les outils sur tout le système de Liberty Rider, dans un second temps, utiliser des outils robustes et fiables, et dernièrement, entrer dans un cadre DevOps pour aligner les différentes équipes grandissantes (par exemple l'utilisation de Docker permet d'augmenter l'environnement DevOps).

Réalisation et gestion

Équipe

Choix techniques

Contraintes

Réalisation backend

Réalisation mobile

Problématiques d'absence de gestion de projet

Mise en production

Analyse des erreurs

Délais courts

Surcharge

Complexité et difficulté inégales selon la plateforme

Migration et nouvelles fonctionnalités

Contexte et objectifs

Préparatifs

Recensement des fonctionnalités

Estimation des charges et du budget

Recrutement de compétences

Logistique

Spécifications

Environnement technique

Réalisation et gestion

Équipe

Gestion de projet

Sprint planning

Daily meeting

Rétrospective de sprint

Revue de planning

Suivi des KPI

Choix techniques

Collaboration des équipes

Contraintes et complexité

Compréhension inter-équipes

Mise en production

Procédure

Contraintes

Évolution

Retours sur investissement

Augmentation des performances

Réduction des coûts

Transformation des processus

Gestion de projet 2.0

Adoption des méthodes agile

Méthode Kanban

Planning et dead line

Utilisation de Asana

Nouvel environnement technique

Solidité

Évolutivité

Projets personnels

SnipHub

Problématique récurrente et naissance de l'idée

Étude du marché et recherche de concurrents

Spécification, estimation, coûts

Réalisation et contraintes

Protection des données personnelles

Mise en production

Maintenance et évolutions

Premier bilan

Conclusion

Annexes

Annexe 1

[image:map competences projects]

LIBERTY RIDER

ÉCOLE D'INFORMATIQUE
Certificateur de compétences

Annexe 2

LIBERTY RIDER

ANGE GARDIEN DES MOTARDS

API - Interface Emergency

Serveur HTTP haute disponibilité pour les traitements critiques. Les applications mobiles le contactent lorsqu'elles détectent un accident pour déclencher l'intervention des secours. Les traitements non critiques sont à éviter pour éviter que des bugs futiles cassent toute la chaîne.

Error Codes

- Timeout ou erreur : premier retry au bout de
  - 0 secondes, puis retry au bout de
  - 5 secondes, puis retry au bout de
  - 15 secondes, puis retry au bout de
  - 30 secondes, puis retry au bout de
  - 60 secondes, puis retry au bout de
  - 60 secondes, puis retry au bout de
  - 120 secondes

Endpoint /emergency/accident

Add to the header of the POST request : Content-Type: application/json

Input :

- id: string au format UUID
- uid: string
- firstName: string
- lastName: string
- email: string
- phoneNumber: string
- isDuo: boolean // if user is in duo
- isTest: boolean // if accident is a test
- sessionId: string // can also be sent as rideId
- shockId: string au format UUID
- appVersion: string
- appPlatform: string
- clientTime : double // ms
- clientUploadTime : double // ms
- lastLocations: Location[] // Les **10** dernières locations, au moins **1** location. L'ordre est important. Les locations les plus **récentes** doivent être à la **fin**. Le timestamp de location doit être arrondi à la minute.
  - latitude: decimal
  - longitude: decimal
  - accuracy: decimal
  - timestamp: int // ms
- brandName: string (''Kawasaki'')
- modelName: string (''ER6'')

Response status code: 204
Pas de data en output

Traitements déclenchés :

- Enregistrer serverUploadTime
- Si erreur imprévue, renvoyer 500
- Vérifier l'idToken
- Si erreur id token: renvoyer 401
- Vérifier intégrité des données (bon format, type, pas vide)
- Si erreur de données: renvoyer 400
- Build XML pour IMA
- Check XML pour IMA
- Si erreur : renvoyer 400
- Envoyer le XML à IMA (config staging seulement) CRITICAL
- Envoyer un email sur alarms@liberty-rider.com (pour déclencher la chaine Zapier), (config staging et production seulement), comme dans le worker. CRITICAL
- Contacter healthchecks.io si test=true (config différentes par environnement)
- Renvoyer 204 (succès) au client
- Ensuite, poster les infos sur l'interface algorithme (on fait ça après avoir retourné 204 car la télémétrie n'est pas critique), cf API - Interface algorithme

Architecture Interne

Liste des services :

- IMA service : build, check, send
- Email service : send
- angel-api service : send
- http service : send response

Tests unitaire :

- Test de chaque service
- Test des controllers

Tests système :

- Teste l'intégralité de la chaîne sous forme de scénarios. Envoi des requêtes http, permet d'émuler le comportement d'un client (mobile, web, etc.).

Environments Variations

Paramètre REST isHealthcheck : Le body de la requête emergency contient le paramètre isHealthcheck=true lorsqu'on veut faire un **test automatique** de bout en bout. Envoyé uniquement par angel-scheduler (toutes les 5 min). Comme l'envoi est très fréquent les données ne sont pas stockées. Donc ça se comporte pas à 100% comme une vraie alerte.

Paramètre REST isTest : Le body de la requête emergency contient le paramètre isTest=true lorsqu'on veut faire un **test manuel** de la chaine de bout en bout. La seule différence de traitement avec une vraie alerte c'est qu'on transmet le paramètre isTest. C'est donc plus réaliste que isHealthcheck. Le seul moyen de faire tourner ce test est d'utiliser un APK dédié.

Interdit d'envoyer à la fois isTest=1 et isHealthcheck=1.

	NODE_ENV=test (circleci et TU) les services externes mockés en TU			NODE_ENV=development (développeur)		
	isTest=0 isHealth=0	**isTest=1** isHealth=0	isTest=0 **isHealth=1**	isTest=0 isHealth=0	**isTest=1** isHealth=0	isTest=0 **isHealth=1**
IMA						
Email						
healthchecks.io						✓
angel-api				✓	✓	

	NODE_ENV=staging info "staging" ajoutée ?			NODE_ENV=production		
	isTest=0 isHealth=0	**isTest=1** isHealth=0	isTest=0 **isHealth=1**	isTest=0 isHealth=0	**isTest=1** isHealth=0	isTest=0 **isHealth=1**
IMA	✓	✓	✓	2018	2018	2018
Email	✓	✓		✓	✓	
healthchecks.io			✓			✓
angel-api	✓	✓		✓	✓	

IMA : envoi de l'alerte à IMA via une API REST en XML. Ils ont un paramètre "isTest" qui correspond au notre, et un paramètre "supervision" qui correspond à notre isHealthcheck.
NOTE: Actuellement IMA est activé seulement en staging !

NOTE TESTS IMA DEBUTANT 28/03 :

- **Création d'une application Android flavor IMA TEST**
- **Envoi d'un paramètre spécial : isIMATest**
- **Notifier l'environnement de production IMA des alertes uniquement si l'accident est flaggé isIMATest = true**
  - **Asana : https://app.asana.com/0/606735458576959/600688449298686**
  - **PR : https://github.com/liberty-rider/emergency-api/pull/42**

Email : envoi d'un email à un Google group dédié alarms@liberty-rider.com, auquel les fondateurs et Zapier sont inscrits. Zapier déclenche l'envoi d'un message Slack sur **#alertes** et l'appel du téléphone d'alerte. C'est le système qu'on utilisait avant IMA, et on le garde en parallèle.
En staging, l'email arrive sur debug+staging+alarms@liberty-rider.com et ne déclenche pas zapier.

healthchecks.io : envoi à https://healthchecks.io/checks/, un service qui nous alerte sur Slack dans le canal #monitoring lorsque le healthcheck est en retard par rapport à l'intervalle de 5 minutes qui est prévu.

angel-api : envoi des détails de l'alerte à notre API pour être stocké dans notre base de donnée, et affiché sur la carte de la session.

LIBERTY RIDER

ÉCOLE D'INFORMATIQUE
Certificateur de compétences

Annexe 3