

Développement d'application mobile de calendrier partagé, React Native ou Flutter ?

Travail de Bachelor réalisé en vue de l'obtention du Bachelor HES

par :

Hervé NEUENSCHWANDER

Conseiller au travail de Bachelor :

Rolf Hauri, professeur HES

Haute École de Gestion de Genève, 25 mai 2020

Haute École de Gestion de Genève (HEG-GE)

Filière IG

Déclaration

Ce travail de Bachelor est réalisé dans le cadre de l'examen final de la Haute école de gestion de Genève, en vue de l'obtention du titre « Bachelor of Science HES-SO en Informatique de gestion ».

L'étudiant a envoyé ce document par email à l'adresse remise par son conseiller au travail de Bachelor pour analyse par le logiciel de détection de plagiat URKUND, selon la procédure détaillée à l'URL suivante : <https://www.orkund.com>.

L'étudiant accepte, le cas échéant, la clause de confidentialité. L'utilisation des conclusions et recommandations formulées dans le travail de Bachelor, sans préjuger de leur valeur, n'engage ni la responsabilité de l'auteur, ni celle du conseiller au travail de Bachelor, du juré et de la HEG.

« J'atteste avoir réalisé seul le présent travail, sans avoir utilisé des sources autres que celles citées dans la bibliographie. »

Fait à Lancy, le 23 mai 2020

Hervé Neuenschwander



Remerciements

J'aimerais remercier Monsieur Hauri, pour son accompagnement pendant ce travail et pour les retours qu'il a pu me faire concernant ce mémoire.

J'aimerais aussi remercier Monsieur Balli pour son idée de projet et pour son soutien pour la réalisation de l'application.

Je remercie aussi Fabien Henaeur pour les relectures qu'il a pu faire de ce document, pour les corrections d'orthographe et les reformulations proposées.

Résumé

L'idée de ce projet est de développer une application permettant de partager des événements au sein d'un groupe de personnes. Ces événements peuvent être uniques ou récurrents. Chaque personne membre du groupe dans lequel est cet événement peut indiquer sa présence. Cette application permet aussi de trouver les disponibilités des personnes d'un certain groupe pour la semaine à venir et ainsi pouvoir trouver un moment ou se rencontrer.

Il existe deux possibilités pour créer une application mobile qui soit disponible sur plusieurs OS : faire une application par OS ou faire un code hybride. La première option est plus performante mais demande plus de travail et de connaissances. La deuxième option peut être un peu moins performante mais facilite beaucoup le travail. De nos jours quasiment plus d'applications sont développée en natif. Pour créer une application hybride il existe deux possibilités : React Natif ou Flutter. Dans ce travail nous allons explorer leur fonctionnement, leur documentation et leur popularité chez les développeurs.

Firebase est une plateforme proposée par Google qui permet principalement de faire du stockage de données. Mais cette plateforme propose d'autres fonctionnalités comme l'envoi d'email, l'authentification et des informations sur l'utilisation de l'application par les utilisateurs. Cette plateforme est pratique car elle est gratuite jusqu'à un certain niveau d'utilisation, et passé cette limite il suffit de payer pour avoir plus d'espace de stockage ou débloquent de nouvelles fonctionnalités.

Table des matières

Développement d'application mobile de calendrier partagé, React Native ou Flutter ?	1
Déclaration.....	i
Remerciements	ii
Résumé	iii
Table des matières.....	iv
Liste des tableaux	vii
Liste des figures.....	vii
1. Introduction.....	1
2. React Native ou Flutter ?	2
3. Natif et hybride : quelles différences ?	3
3.1 Le développement natif	3
3.2 Le développement hybride.....	3
3.3 Pourquoi un développement hybride ?	3
4. Étude des framework	5
4.1 Le framework.....	5
4.2 React Native	5
4.2.1 Avantages du rendu en langage natif	6
4.2.2 Architecture de React Native.....	6
4.2.2.1 JavaScript VM	6
4.2.2.2 React Native Bridge	6
4.2.2.3 Native Modules.....	6
4.2.3 Les threads dans React Native	7
4.2.3.1 Main thread.....	7
4.2.3.2 JavaScript thread	7
4.2.3.3 Native Modules Thread	7
4.2.4 React Native Fabric.....	8
4.2.4.1 La nouvelle architecture de React Native	8
4.2.4.2 JSI	9
4.2.4.3 Turbo Modules.....	9
4.2.4.4 Fabric.....	9
4.2.4.5 CodeGen	9
4.2.5 Hermes	10
4.2.5.1 Pré-compilation du bytecode.....	10
4.2.5.2 Pas de compilation à la volée.....	11
4.2.5.3 Stratégie de garbage collector	11
4.3 Flutter.....	11
4.3.1 Architecture.....	12
4.3.1.1 L' « Engine ».....	12
4.3.2 Dart.....	13

4.3.3	Skia.....	13
4.3.4	Les widgets.....	14
4.3.4.1	Widgets stateful et stateless.....	14
4.4	Comparaison	15
4.4.1	Résumé des frameworks.....	15
4.4.2	Documentation.....	15
4.4.3	Communauté	16
4.4.4	Intérêt de la communauté.....	16
5.	Problématiques à résoudre	18
5.1	Partager des événements.....	18
5.2	Événements unique et récurrents.....	18
5.3	Trouver une date pour un événement.....	18
6.	Concurrents	20
6.1	Google Agenda.....	20
6.2	Calendrier pour iOS	20
6.3	Doodle.....	21
6.4	NeedToMeet.....	21
6.5	Heja	21
6.6	TimeTree.....	21
6.7	Résumé des concurrents	22
7.	Fonctionnalités de haut niveau	23
7.1	Use-cases	23
7.1.1	Créer un compte	23
7.1.2	Connexion.....	24
7.1.3	Réinitialiser son mot de passe.....	24
7.1.4	Créer un groupe.....	24
7.1.5	Créer une disposition	24
7.1.6	Accepter et refuser une invitation	24
7.1.7	Indiquer sa présence à un événement	25
7.1.8	Indiquer ses disponibilités dans une disposition	25
7.1.9	Modifier et supprimer une disposition dont il est l'auteur	25
7.1.10	Créer un événement.....	25
7.1.11	Créer un événement depuis une disposition	26
7.1.12	Modifier et supprimer un événement.....	26
7.1.13	Modifier et supprimer une disposition	27
7.1.14	Envoyer une invitation	27
7.1.15	Modifier et supprimer un groupe	27
7.1.16	Modifier le rôle d'un membre	27
7.1.17	Retirer un membre.....	28
7.2	Vues	28

7.2.1	Écran de connexion	28
7.2.2	Écran de création de compte	29
7.2.3	Écran de profil.....	30
7.2.4	Écran de création de groupe	31
7.2.5	Écran de liste des groupes.....	32
7.2.6	Écran de détails d'un groupe.....	33
7.2.7	Écran de modification de groupe.....	34
7.2.8	Écran d'accueil.....	35
7.2.9	Écran de création de disposition	36
7.2.10	Écran de liste des dispositions.....	37
7.2.11	Écran de détails de disposition	38
7.2.12	Écran de modification de disposition.....	39
7.2.13	Écran de création d'évènement d'après une disposition	40
7.2.14	Écran de création d'évènement	41
7.2.15	Écran de détails d'évènement.....	42
7.2.16	Écran de modification d'évènement.....	43
7.2.17	Écran de liste d'invitations	44
8.	Perspectives d'avenir.....	45
8.1	S'inscrire en utilisant les réseaux sociaux.....	45
8.2	Synchroniser les calendriers	45
8.3	Envoie de notification dans l'application	45
8.4	Chats dans les événements	45
9.	Utilisation des frameworks	46
9.1	Apprentissages	46
9.1.1	React Native	46
9.1.2	Flutter.....	46
9.1.2.1	Prototype	46
9.2	Difficultés.....	47
9.2.1	React Native	47
9.2.2	Flutter.....	47
10.	Conclusion	49
	Bibliographie	50

Liste des tableaux

Tableau 1 : Résumé des frameworks	15
Tableau 2 : Résumé des concurrents	22

Liste des figures

Figure 1 : Architecture de React Native	6
Figure 2 : Nouvelle architecture de React Native	9
Figure 3 : Améliorations apportées par Hermes	10
Figure 4 : Séquence de compilation du bytecode sans Hermes	10
Figure 5 : Séquence de compilation du bytecode avec Hermes	11
Figure 6 : Architecture de Flutter	12
Figure 7 : Intérêt des recherches de React Native et Flutter sur Google.....	17
Figure 8 : Pourcentage par mois de post sur StackOverflow tagués flutter ou react-native	17
Figure 9 : Use-case de l'application.....	23
Figure 10 : Écran de connexion.....	28
Figure 11 : Écran de création de compte.....	29
Figure 12 : Écran de profil	30
Figure 13 : Écran de création de compte.....	31
Figure 14 : Écran de liste des groupes	32
Figure 15 : Écrans de détails d'un groupe	33
Figure 16 : Écran de modification de groupe	34
Figure 17 : Écran d'accueil	35
Figure 18 : Écran de création de disposition.....	36
Figure 19 : Écran de liste des dispositions	37
Figure 20 : Écran de détails de disposition	38
Figure 21 : Écran de modification de disposition	39
Figure 22 : Écran de création d'évènement d'après une disposition	40
Figure 23 : Écran de création d'évènement	41
Figure 24 : Écran de détails d'évènement	42
Figure 25 : Écran de modification d'évènement.....	43
Figure 26 : Écran de liste d'invitations	44

1. Introduction

Pour ce travail, je voulais pouvoir réaliser une application pour un mandant. Je cherchais à faire une application concrète et pas seulement un travail de recherche. J'ai donc commencé par chercher un mandant dans mes connaissances, mais je n'en ai pas trouvé. Je me suis donc tourné vers le site « Reddit » et j'ai demandé sur le groupe de Genève si quelqu'un avait une idée d'application qui pourrait être réalisée dans le cadre d'un travail de Bachelor. Monsieur Balli a répondu à ce message et nous nous sommes ensuite rencontré pour discuter de cette application.

Le développement d'applications mobile prend de plus en plus de place dans le monde de l'informatique. C'est donc sans surprise que les géants d'internet ont décidé d'offrir des outils pour soutenir ce marché. Le développement de ce marché permet de répondre à de nombreux besoins, malheureusement la qualité de ces applications mobile est parfois moyenne.

C'est le constat qu'a pu faire le mandant, Monsieur Balli, lorsqu'il cherchait une application lui permettant de partager des événements avec différents groupes de connaissances. Il a donc réfléchi aux fonctionnalités qui doivent être présentes dans ce genre d'application. L'application permettra donc de pouvoir créer des événements à partager avec des groupes de personnes mais permettra aussi de trouver des créneaux avec le maximum de personnes disponible pour créer ces événements. L'application doit permettre de créer des événements uniques mais aussi des événements récurrents. Ces événements récurrents doivent cependant garder une certaine indépendance les uns des autres, par exemple un utilisateur doit pouvoir être présent à un seul de ces événements. L'application permettra aussi de créer des événements à partir des disponibilités des membres d'un groupe.

J'avais personnellement déjà pensé à faire une application de la sorte, c'est donc avec enthousiasme que j'ai accueilli la proposition du mandant. Mais comment développer une application en très peu de temps et pour le maximum d'utilisateurs possibles ? Les frameworks qui permettent de faire du développement hybride étaient une évidence. Mais quel framework choisir ? C'est une question à laquelle je vais essayer de répondre en partie en comparant deux de ces frameworks : React Native et Flutter.

2. React Native ou Flutter ?

Ces deux frameworks ont de grandes similarité mais aussi de nombreuses différences. Ils permettent tous les deux de créer des applications pour des smartphones et sont tous les deux composés de blocs. Dans le cas de React Native ces blocs sont appelés des composants, dans le cas de Flutter ils sont appelés des widgets. Ils ont aussi des différences de par leurs architectures et leurs manières de fonctionner mais surtout au niveau des entreprises qui les ont créés. En effet React Native a été développé par Facebook en 2015 et Flutter par Google en 2018.

Il sera donc intéressant d'étudier leurs différences et leurs points communs. Nous pourrons les comparer selon plusieurs points de vue, comme leurs maturités ou la vision de la communauté. Il est aussi intéressant de voir que Google a souhaité créer un framework permettant de construire des applications hybrides alors qu'un autre géant d'internet avait déjà créé un outil de la sorte.

Malgré le fait qu'ils cherchent à proposer une solution au même problème, qui est de faciliter la production d'applications tout en réduisant les couts, ils fonctionnent de manières bien différentes avec une architecture propre à chacun. Pourtant cette architecture doit régler le même problème qui est de pouvoir exécuter un code sur plusieurs plateformes. Nous nous intéresserons donc à ces architectures pour comprendre comment ces frameworks règlent ce problème.

3. Natif et hybride : quelles différences ?

3.1 Le développement natif

Le développement natif est la création d'une application pour une plateforme en particulier. Elles doivent être écrites dans le langage de la plateforme cible, soit : Java pour Android et Objective-C pour iOS. Le développement en natif permet d'avoir accès aux fonctionnalités du smartphone par défaut, comme la caméra ou le microphone. Pour développer une application qui puisse être installée sur tous les OS il faudra donc coder autant d'applications différentes qu'il y a d'OS. Cela demande donc une plus grosse équipe de développeurs.

Les applications natives interagissent directement avec les APIs natives et n'ont pas besoin d'interface. Puisqu'il n'y a pas ces intermédiaires, les applications sont plus rapides et ont un meilleur affichage.

Pour développer une application native, il faut utiliser les SDK (software development kit) natifs. L'interface utilisateur est donc la même pour toutes les applications natives et aussi très semblables à l'OS lui-même.

Apple a commencé, en 2017, à ne pas accepter certaines applications hybrides. Il sera donc plus simple pour une application native d'être publiée sur cet app store. (Understanding native app development, 2019)

3.2 Le développement hybride

Les applications hybrides sont un mix de technologies consacrées au web et au natif. L'application est écrite en utilisant des langages généralement utilisés dans le cadre des sites web, comme le CSS, l'HTML et le JavaScript. En fait, lorsqu'une application hybride est lancée sur un smartphone, c'est un navigateur qui est lancé et c'est dans ce navigateur que l'affichage de l'application se fait.

Le développement hybride offre des solutions pour accéder aux différents composants d'un smartphone. Il est possible d'installer des plugins qui permettent de les utiliser.

Le plus gros avantage du développement hybride est de n'avoir à faire qu'un seul code pour les différents OS. Cela permet de développer une application qui pourra toucher un maximum d'utilisateurs tout en réduisant drastiquement les coûts. (IONIC, 2019)

3.3 Pourquoi un développement hybride ?

Dans le cadre du développement de cette application, nous voulons pouvoir toucher un maximum d'utilisateurs, pour cela il faut donc que l'application soit disponible sur toutes

les plateformes. Puisqu'il n'y a pas plusieurs développeurs sur le projet et que le temps de développement est restreint, la solution qui permet de remplir le cahier des charges est donc de créer une application hybride.

4. Étude des framework

4.1 Le framework

Un framework est un ensemble de composants qui offre des fonctionnalités qui peuvent ensuite être modifiées par l'utilisateur. Afin de travailler dans les meilleures conditions, il faut travailler avec l'outil qui répond le mieux à nos besoins. Pour cela, il faut explorer les différentes possibilités qui s'offrent à nous. Pour ce travail un des points le plus important est d'avoir un framework « cross-plateforme », soit un framework qui permette de créer un seul code pour plusieurs supports, dans notre cas iOS et Android. Il existe plusieurs de ces framework :

- Xamarin par Microsoft
- React Native par Facebook
- Flutter par Google
- Ionic par Drifty

Nous allons nous intéresser uniquement à deux de ces frameworks : React Native et Flutter.

4.2 React Native

La première version de React Native date du 26 mars 2015. Ce framework est développé par Facebook.

React Native est un framework qui utilise des technologies asynchrones, cela permet d'avoir plusieurs tâches qui se déroulent en même temps. Par exemple, on peut demander des données à un serveur et en attendant d'avoir la réponse continuer de répondre aux différents gestes que peut faire l'utilisateur. Cela permet d'avoir une application qui est plus fluide et qui ne se bloque pas en attendant qu'une tâche soit terminée.

La technologie peut être utilisée explicitement par le développeur en créant des fonctions asynchrones, comme l'exemple ci-dessus pour demander des données à un serveur. Mais React Native est un framework asynchrone dans son architecture, car il utilise plusieurs thread qui peuvent s'échanger des informations. Cela est plus détaillé dans la suite de ce document.

Une application utilisant React Native est écrite en JavaScript, cependant React Native fait le rendu de l'application en langage natif, soit Java ou Kotlin pour les appareils Android et Swift ou Objective-C pour les appareils iOS. Les avantages de cette méthode sont nombreux. (React Native, 2020)

4.2.1 Avantages du rendu en langage natif

Les appareils mobiles sont prévus pour fonctionner avec un certain langage. Ils sont donc plus performants lorsqu'ils travaillent directement avec ce langage-ci. De plus, l'affichage des éléments se fera selon les standards de la plateforme.

4.2.2 Architecture de React Native

Les smartphones (ou tablettes) ne peuvent pas interpréter directement du JavaScript, il faut donc d'une certaine manière transformer ce code en un code compréhensible par l'OS. Afin de faire cette transformation React Native implémente une architecture qui peut être découpée en trois couches. (React Native Internals)

4.2.2.1 JavaScript VM

La première couche est appelée « JavaScript VM ». Comme son nom l'indique, c'est une machine virtuelle qui exécute le code JavaScript. La machine virtuelle qui est utilisée est « JavaScriptCore ». Cette machine virtuelle est utilisée par le navigateur Safari par exemple.

4.2.2.2 React Native Bridge

La deuxième couche est appelée « React Native Bridge ». Cette couche permet de faire la communication entre la couche JavaScript et la couche native. Les communications qui se font au travers du Bridge sont asynchrones, sérialisées et par lots. Pour envoyer un objet du JS au natif, il devra être sérialisé en JSON. Pour appeler une fonction synchrone d'un objet natif depuis du JS, il faudra le faire de manière asynchrone, car le Bridge fonctionne de manière asynchrone. Parce que le Bridge fonctionne de manière asynchrone, il n'y a pas de risque que l'application se bloque pendant un appel à un objet par exemple.

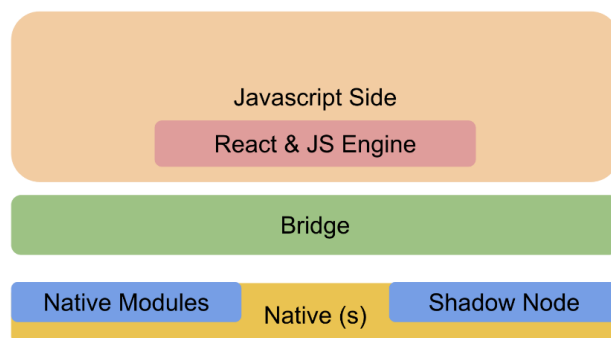
4.2.2.3 Native Modules

La troisième et dernière couche est appelée « Native Modules ». Cette couche est la couche native de l'OS. Elle est en communication directe avec les composants du smartphone ou de la tablette. Si des composants spécifiques de l'appareil doivent être utilisés tels que la caméra ou simplement l'écran, c'est cette couche qui fera le nécessaire.

Figure 1 : Architecture de React Native



Before - Current Architecture



(FELDMAN, 2019)

4.2.3 Les threads dans React Native

Mais comment fait notre smartphone pour exécuter ces trois couches en même temps ? Lorsqu'une application React Native est exécutée sur un smartphone trois threads sont lancés. Un thread est une tâche qui peut virtuellement être exécutée en même temps que d'autres tâches.

4.2.3.1 Main thread

(Comme son nom l'indique, c'est le thread principal, c'est le premier à être lancé. React Native va charger le code JavaScript et lancer un thread pour exécuter ce code. Ce thread est aussi celui qui permet de réaliser l'affichage et de passer des événements (comme le clic d'un bouton) au thread qui exécute la logique métier.

4.2.3.2 JavaScript thread

Ce thread exécute le code qui a été écrit en JavaScript dans notre application, c'est en principe la logique métier. Les mises à jour de l'affichage entraînées par ce thread sont faites par paquet à chaque fin de boucle d'événement. Étant donné que les mises à jour sont envoyées par paquet, si la boucle prend trop de temps à se terminer, des baisses d'images par secondes apparaîtront. Si l'écran de l'appareil est capable de dessiner 60 images par seconde, cela veut dire que la boucle en JavaScript doit s'exécuter en 16.6 millisecondes. (KUMAR, 2018)

4.2.3.3 Native Modules Thread

Ce thread est utilisé lorsque l'application a besoin d'accéder à certains API natifs. Par exemple, s'il faut accéder à la caméra ou au GPS, cela se fera sur ce thread

4.2.4 React Native Fabric

Les programmeurs qui travaillent sur React Native se sont rendu compte que de travailler avec de l'asynchrone peut présenter des problèmes. Par exemple, une application peut afficher des cases vides dans une liste si on fait défiler trop vite une liste, car un premier thread doit envoyer qu'on fait défiler, ensuite un deuxième thread doit récupérer les données et enfin un troisième thread doit dessiner ces données à l'écran. Cela est un processus relativement long.

À partir de 2018, Facebook a dévoilé ce qu'ils appellent « React Native Fabric ». C'est une nouvelle architecture de React Native qui permet de régler plusieurs problèmes que rencontrait React Native. Trois gros changements vont être apportés.

Premièrement, il sera désormais possible pour n'importe quel thread d'appeler directement le thread JavaScript pour faire des mises à jour synchrone et donc prioritaires.

Deuxièmement, il sera possible d'avoir des rendus asynchrones qui permettront de mieux contrôler les données asynchrones, mais aussi d'avoir plusieurs priorités de mises à jour d'affichage.

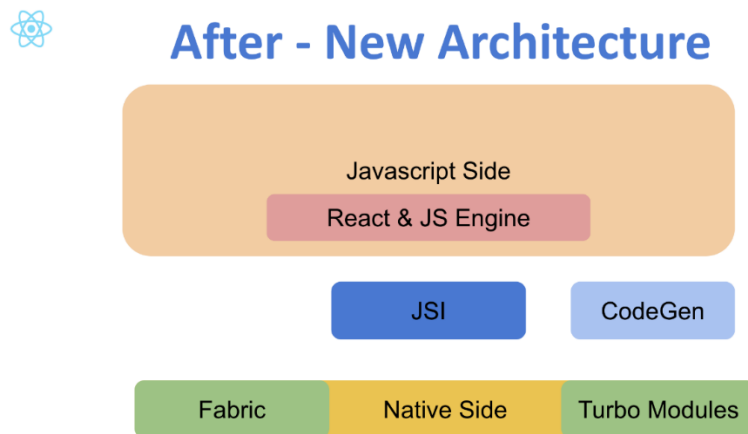
Troisièmement, le Bridge sera simplifié pour le rendre plus rapide et plus léger. (State of React Native 2018)

Cette nouvelle architecture n'est pas encore disponible. Facebook veut d'abord utiliser en interne cette nouvelle architecture avant de la proposer au public. Cela leur permet de tester la nouvelle architecture et de continuer à l'améliorer pour la rendre plus stable. Facebook pense pouvoir commencer à déployer petit à petit cette nouvelle architecture en début 2020. (React Native EU, 2019)

4.2.4.1 La nouvelle architecture de React Native

Dans la nouvelle architecture, le Bridge est remplacé par le JSI qui veut dire « JavaScript Interface », le nouveau manager d'interface utilisateur sera appelé « Fabric ». Les composants natifs sont renommés en « Turbo Modules » et « CodeGen » fait son apparition

Figure 2 : Nouvelle architecture de React Native



(FELDMAN, 2019)

4.2.4.2 JSI

JSI est le nouvel outil qui va remplacer le Bridge actuel. La plus grosse différence avec le Bridge est qu'il permettra au JavaScript et au natif d'être conscients de l'existence l'un de l'autre. JSI permet de créer des objets, d'accéder aux variables et d'appeler des fonctions entre le JS et le natif. En permettant la communication directe entre le JS et le natif, JSI permet de réduire le temps de communication entre ces deux couches.

4.2.4.3 Turbo Modules

Ces nouveaux modules offrent la possibilité d'être initialisés uniquement lorsqu'ils sont utilisés. Anciennement, tous les modules étaient initialisés au démarrage de l'application. Cela permet de réduire le temps de démarrage de l'application.

4.2.4.4 Fabric

Fabric est le nouveau manager d'interface utilisateur. Il n'y a plus besoin d'avoir un thread pour faire les mesures et accéder au composant du smartphone et les mesures peuvent être faites de manière synchrone. Ce nouveau manager sera aussi initialisé uniquement lorsqu'il sera utilisé, ce qui permettra un démarrage plus rapide.

4.2.4.5 CodeGen

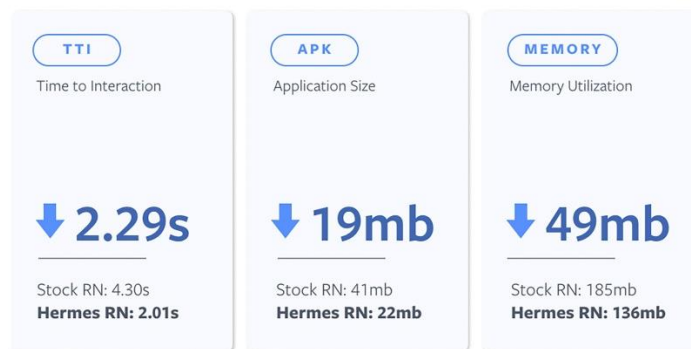
Cet outil permet d'automatiser la compatibilité entre le JS et le natif. Cet outil permet de générer des interfaces qui seront utilisées par Fabric et les Turbo Modules qui permettent de s'assurer que les données sont dans le type requis. Cela rendra la communication plus rapide puisqu'il ne sera plus nécessaire de vérifier à chaque fois le type de donnée.

4.2.5 Hermes

Les développeurs qui travaillent sur React Native chez Facebook cherchent toujours à améliorer ce framework. En analysant les données à leur disposition, ils se sont rendus compte que la JavaScript VM (JavaScriptCore) ralentissait le démarrage des applications et que la taille des applications pouvait être améliorée. Ils ont donc travaillé à l'optimisation des performances de JavaScript et on créé Hermes.

Hermes est donc une nouvelle machine JavaScript qui permet d'optimiser les applications React Native sur Android. Elle est construite spécifiquement pour rendre le démarrage des applications plus rapide.

Figure 3 : Améliorations apportées par Hermes

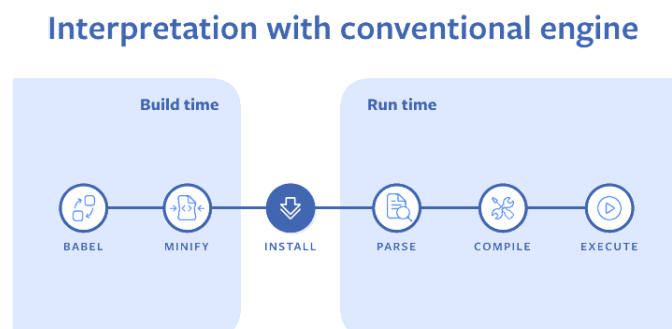


(Hermes, 2019)

4.2.5.1 Pré-compilation du bytecode

De manière générale, les machines JavaScript créent le bytecode après avoir été installées. Cela ralentit le démarrage de l'application, afin de résoudre ce problème Hermes fait une compilation du bytecode avant l'installation de l'application. En faisant la compilation avant l'installation de l'application, la génération du bytecode peut prendre plus de temps et ainsi être plus optimisée ce qui le rend aussi plus petit.

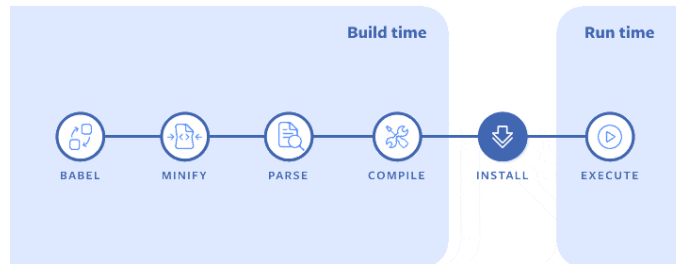
Figure 4 : Séquence de compilation du bytecode sans Hermes



(Hermes, 2019)

Figure 5 : Séquence de compilation du bytecode avec Hermes

Bytecode precompilation with Hermes



(Hermes, 2019)

4.2.5.2 Pas de compilation à la volée

La compilation à la volée permet d'accélérer l'exécution d'un programme en compilant les fichiers uniquement lorsqu'il y en a besoin. Hermes fait la compilation du bytecode avant de l'installer sur un appareil, il n'y a donc aucunement besoin d'avoir une compilation à la volée.

4.2.5.3 Stratégie de garbage collector

La mémoire peut être plus limitée sur les appareils mobiles et pour économiser de la batterie les systèmes d'exploitation arrêtent les applications qui utilisent trop de mémoire. Afin de palier à ces problèmes, Hermes implémente quelques modèles qui permettent d'optimiser la mémoire.

Premièrement, l'allocation de mémoire est faite au besoin. C'est-à-dire que l'application commencera avec un minimum de mémoire et s'il en faut plus, la mémoire sera allouée au fur et à mesure.

Deuxièmement, le garbage collector fonctionne avec un système de générations. Un récupérateur à génération fonctionne en classant les nouvelles données créées dans des générations. Le récupérateur scanne les données qui sont de la génération la plus jeune et supprime les données qui ne sont plus utilisées. Les données encore utilisées sont placées dans la génération plus âgée. Le garbage collector de Hermes ne scanne pas toutes les données à chaque génération. Cela le rend plus rapide. (Hermes, 2019)

4.3 Flutter

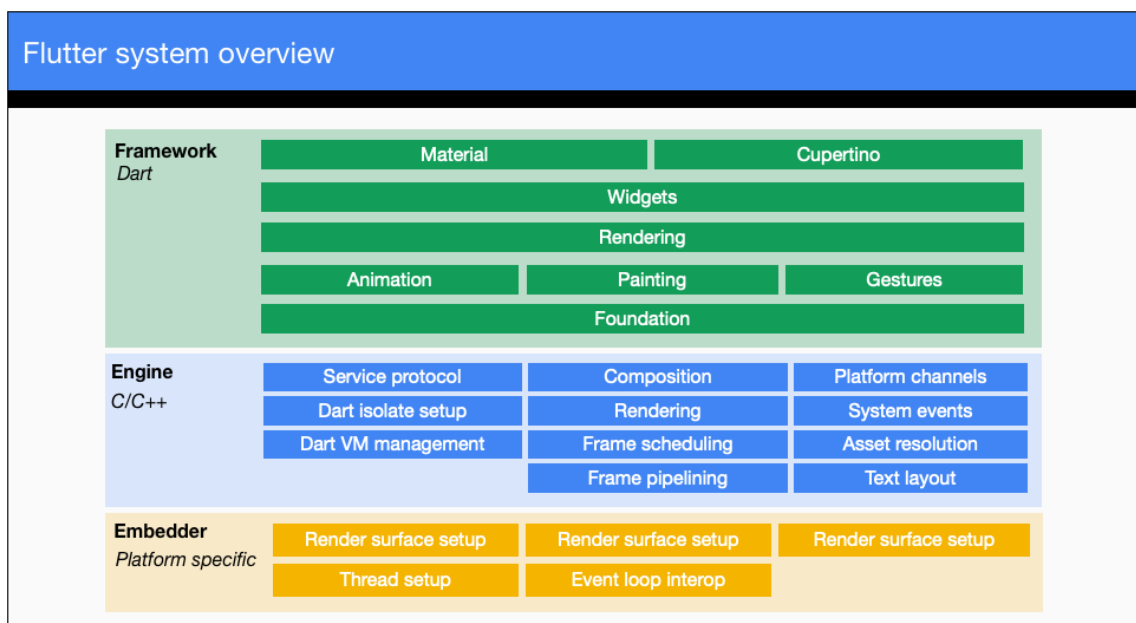
La première version de Flutter date de décembre 2018, c'est un framework développé par Google. Ce framework permet, tout comme React Native, de développer des applications mobiles pour différentes plateformes comme Android, iOS, Windows ou MacOS. Flutter malgré son jeune âge est utilisé par de grands groupes internationaux comme : Alibaba, BMW, eBay ou encore Tencent. (Flutter)

Les widgets sont au cœur de ce framework. Un widget est n'importe quel composant de l'application. Un widget peut représenter un bouton, mais un widget permet aussi de styliser un interface en permettant de centrer un élément. Un widget peut aussi être le composant qui permet le défilement d'une liste. Par exemple, pour centrer un bouton, le widget « bouton » devra être un enfant du widget « centrer ».

4.3.1 Architecture

L'architecture de Flutter est en trois couches. La première couche est appelée « Dart Framework », c'est simplement le code de l'application écrite en langage Dart (voir chapitre 6.3.2). La deuxième couche est appelée « Engine », elle permet de faire le lien entre le code écrit par le développeur et l'appareil mobile. Enfin, la troisième couche est appelée « Embedder », elle permet de faire, comme son nom l'indique, l'intégration de la deuxième couche, en faisant la connexion avec le système natif de l'appareil mobile.

Figure 6 : Architecture de Flutter



(The Engine architecture, 2019)

4.3.1.1 L' « Engine »

Cette couche embarque Skia et Dart et les "encapsules". Elle permet de faire de la lecture et de l'écriture sur des fichiers ou sur le réseau. Elle permet aussi de faire des actions spécifiques à la plateforme comme par exemple, la communication avec les claviers des différents OS.

L' « Engine » ne s'occupe pas de créer et de gérer les différents threads nécessaires au fonctionnement de l'application, ceux-ci sont gérés par la troisième couche. Il est nécessaire d'avoir 4 threads qui sont :

- Le thread de la plateforme
- Le thread de l'interface utilisateur
- Le thread du GPU
- Le thread d'écriture et de lecture

Le thread de la plateforme est le thread principal. Il permet de recevoir et d'exécuter les messages de l'OS, la plupart des API peuvent être atteints de manière sécurisée seulement depuis le thread principal d'une application.

Le thread de l'interface utilisateur permet de construire ou de modifier les éléments qui seront affichés à l'écran. Il permet donc de modifier les animations, de reconstruire les widgets qui ont été modifiés et de faire la mise en pages. Ces composants et ces modifications sont ensuite passés au thread du GPU.

Le thread du GPU reçoit des informations de composant à afficher à l'écran. Il construit les commandes qui seront passées au GPU de l'appareil. Il est quasiment impossible pour le développeur d'avoir un accès direct à ce thread afin de ne pas avoir des bases de fluidités.

Le thread d'écriture et de lecture permet de lire ou d'écrire dans des fichiers. Ce sont en général des opérations assez lentes et afin de ne pas bloquer l'application elles sont faites dans un thread spécifique. (The Engine architecture, 2019)

4.3.2 Dart

Dart est un langage de programmation créé par Google. C'est un langage qui permet de créer des applications sur n'importe quelle plateforme. Ce langage est optimisé et spécialisé dans la création d'interfaces utilisateur.

Dart est un langage qui permet de créer des fonctions asynchrone qui permettent donc de ne pas bloquer le rendu de l'application. Contrairement à d'autres langages qui offrent la possibilité d'exécuter des processus en parallèle, Dart propose l'utilisation d'« isolates ». Un Isolates est un peu comme un thread à la différence que sa mémoire n'est pas partagée avec les autres Isolates, c'est d'ailleurs pour cela qu'il porte ce nom. Deux Isolates peuvent quand même se passer des informations via des messages et lorsqu'un Isolate reçoit un message, il peut exécuter une certaine portion de code. (WALRATH, 2019)

4.3.3 Skia

Skia est une librairie graphique 2D produite initialement par le groupe Ski Graphic Engine. Skia a ensuite été acheté par Google en 2005 et est maintenant utilisé par Google Chrome, Chrome OS, Firefox, Android et évidemment Flutter. Flutter peut

prendre le contrôle de l'entièreté de l'écran de l'appareil mobile, cela permet d'avoir des animations plus fluides. Grâce à Skia, Flutter peut donc faire un affichage indépendant de l'OS et avoir par exemple des messages d'alertes Android sur un smartphone iOS. Skia n'as pas besoin d'échanger avec le smartphone pour faire un affichage et donc l'affichage est plus rapide. (KURIAN, 2019)

4.3.4 Les widgets

Comme déjà expliqué au paravent, dans Flutter tout est un widget. Les widgets sont composés d'autres widgets plus simples qui eux même sont composé de widgets plus simples et ainsi de suite. Cette hiérarchie de widget est appelée le « render tree » et a comme racine le widget nommé « RenderObjectWidgets ».

Ce composant est extrêmement basique ce qui lui permet de s'adapter à des mises en pages différentes sans avoir à le modifier. Par exemple ce widget, malgré qu'il soit responsable de l'affichage de tous les widgets de l'application, n'as pas de système de coordonnées fixe. (Rendering in Flutter)

Le « render tree » doit être représenté dans une structure de données qui puisse fonctionner efficacement avec beaucoup d'éléments. Puisque chaque widgets est composé de plusieurs widgets, le render tree peut facilement atteindre une très grande taille. Les programmeurs de Flutter ont donc cherché à optimiser cet objet.

L'arbre de widgets est parcouru une seule fois par frame de haut en bas et une seule fois de bas en haut. Lors de la descente de l'arbre, les parents transmettent à leurs enfants la taille maximale et minimale qu'ils peuvent prendre. En remontant la seule information qui transite des enfants vers leurs parents est la taille qu'ils ont pris. À ce moment le parent choisi la position de l'enfant. Afin d'optimiser le parcours de cet arbre, il existe plusieurs conditions qui font qu'une branche ne sera parcourue. Par exemple si la taille minimale possible et la taille maximale possible sont les mêmes, un parent ne devra pas recalculer sa mise en page même si celle de l'enfant a changé. Un autre exemple est que les widgets ont des flags qui permettent d'indiquer si eux même ou un de leurs enfants a changé, si le flag indique que rien n'a changé la branche n'est pas parcourue. (Inside Flutter)

4.3.4.1 Widgets stateful et stateless

Il existe deux grands types de widgets qui sont les stateful et les stateless. Les widgets stateless ou sans état en français sont des objets qui ne peuvent pas changer. Par exemple, du texte ou un bouton. Les widgets stateful sont au contraire des widgets qui peuvent changer comme une checkbox.

4.4 Comparaison

4.4.1 Résumé des frameworks

Tableau 1 : Résumé des frameworks

	React Native	Flutter
Année de sortie	2015	2018
Langage	JavaScript	Dart
Créateur	Facebook	Google
IDE compatibles	Atom, Visual Studio Code, ...	Intellij IDEA, Android Studio, Visual Studio Code...
Maturité	Forte communauté, beaucoup d'aide autre que le site officiel	Communauté encore jeune, quantité modéré d'aide autre que le site officiel

4.4.2 Documentation

Flutter propose une documentation fournie et assez détaillée. Bien entendu, il y a un guide pour faire sa première application et cela nous guide aussi dans l'installation des différents éléments requis. Il existe différentes pages sur le site officiel de la documentation de Flutter pour aider les développeurs qui ont des connaissances dans d'autres langages. Il existe donc une page pour aider les développeurs qui ont l'habitude de travailler sur Android ou iOS, une page pour ceux qui ont l'habitude de travailler avec React Native, pour les développeurs web et quelques autres pages du même type. La plupart des API de Flutter ont une page dédiée avec leur fonctionnement et des exemples.

Il existe aussi différentes sections pour chaque fonctionnalité que l'on peut apporter à notre application. Ainsi, il y a une section concernant l'interface avec des détails sur la mise en page ou les interactions, une section pour les données avec des détails sur la sérialisation en JSON et le lien avec Firebase, etc (Flutter Documentation)

La documentation de React Native présente grossièrement les mêmes chapitres et les mêmes guides. Il y a une page qui permet d'installer les éléments nécessaires au fonctionnement et au développement des applications avec React Native. Les chapitres pour faire sa première application sont un peu plus détaillés et une suite de pages

permettent d'apprendre petit à petit des connaissances qui seront utiles pour n'importe quelle application. Il existe aussi des guides détaillés pour des fonctionnalités telles que les animations, du code spécifique aux plateformes ou la navigation entre des écrans. Tout comme Flutter il y a aussi des pages qui expliquent les API comme la géolocalisation ou les notifications push. (Getting Started · React Native)

La tendance des articles qui comparent ces deux frameworks est de dire que la documentation proposée par React Native est moins précise que celle proposée par Flutter, cependant les deux frameworks ont une documentation très complète et claire.

4.4.3 Communauté

Les communautés de ces deux frameworks sont nombreuses et actives sur Internet. Il existe plusieurs sites qui proposent des groupes pour discuter sur ces sujets ou pour poser des questions et avoir des réponses de la communauté.

Sur le site officiel de Flutter, il existe un onglet « Community » qui permet de trouver des endroits où poser nos questions, mais aussi où la communauté peut répondre aux problèmes des utilisateurs. Le GitHub officiel compte environ 550 contributeurs et plus de 5000 problèmes. Sur StackOverflow il y a plus de 37'000 posts avec « flutter » comme tag, cependant seul environ 21'000 questions sont marquée comme résolues. (Newest “flutter” Questions)

Sur le site officiel de React Native, il y a aussi un onglet « Community » qui permet de trouver quelques liens utiles de la communauté. Il y a aussi un lien qui redirige directement vers leur GitHub, sur lequel nous pouvons voir qu'il y a plus de 2000 contributeurs différents et un peu moins de 700 problèmes. Sur StackOverflow il y a plus de 70'000 questions ayant comme tag « react-native » parmi lesquelles 37'000 sont marquées comme résolues. (Newest “react-native” Questions)

Il existe aussi un groupe sur le site spectrum, qui est un site de chat, avec plus de 3500 membres. (React Native community)

Nous pouvons voir que les deux communautés sont nombreuses, cependant en se basant sur les GitHub officiels nous pouvons constater que Flutter a bien plus de problèmes en cours et bien moins de contributeurs. Cela montre que c'est un framework encore jeune.

4.4.4 Intérêt de la communauté

Logiquement, puisque React Native est apparu environ 3 ans avant Flutter, sa communauté est plus grande. Cependant comme nous pouvons le voir dans les

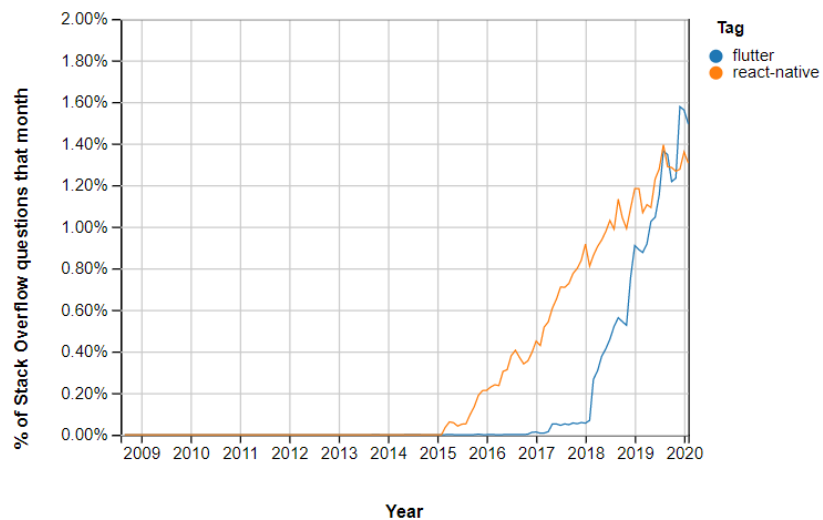
graphiques ci-dessous cette tendance est en train de s'inverser et l'intérêt des développeurs pour Flutter est grandissante, et même en train de surpasser celui porté à React Native.

Figure 7 : Intérêt des recherches de React Native et Flutter sur Google



(Google Trends)

Figure 8 : Pourcentage par mois de post sur StackOverflow tagués flutter ou react-native



(StackOverflow Trends)

5. Problématiques à résoudre

Monsieur Balli s'est confronté à plusieurs problèmes que cette application va chercher à résoudre. Le premier est de pouvoir facilement partager des événements avec des proches. Il faut aussi pouvoir être en mesure de créer des événements unique, mais aussi des événements qui seront récurrents tout en étant indépendant les uns des autres. Un autre problème est de pouvoir trouver une date ou une plage horaire permettant à un groupe de personne de se retrouver.

5.1 Partager des événements

Partager des événements peut paraître simple et c'est quelque chose qui existe déjà. Cependant, cela n'existe pas sous la forme recherchée. Le but est de pouvoir créer des groupes en ajoutant des personnes et d'avoir un calendrier commun à ce groupe. Par exemple pour une équipe de sport, il faudrait pouvoir ajouter dans le groupe tous les membres de l'équipe. Ensuite en créant un événement tel qu'un entraînement, une notification sera envoyée aux membres du groupe pour les avertir et leur demander une confirmation de présence.

5.2 Événements unique et récurrents

Créer un événement unique ne pose pas de problème particulier, le problème est plutôt par rapport aux événements récurrents. Le problème est d'avoir un événement récurrent mais cependant indépendant les uns des autres. En reprenant notre exemple d'entraînement pour une équipe de sport, nous pouvons facilement illustrer le problème des événements unique et récurrents. Par exemple, nous créons l'événement récurrent "entraînement" tous les mardis. Les membres du groupe doivent pouvoir indiquer pour chaque entraînement individuellement s'ils seront présents ou pas.

5.3 Trouver une date pour un événement

Cette partie de l'application peut faire penser à Doodle. Ce site web permet de proposer des dates ou des plages horaires et les utilisateurs qui reçoivent une invitation peuvent mettre leurs disponibilités. Le problème est que c'est un site web à part et qu'il faut inviter chaque personne. Il est aussi impossible d'informer un groupe de personne de ses disponibilités.

Dans notre application il sera possible tout comme dans Doodle de proposer des dates ou plages horaire pour une activité en particulier directement dans un groupe de personne et chaque personne pourra ajouter ses disponibilités personnelles. De plus, il sera possible d'informer un groupe de nos disponibilités pour les jours à venir. Cela

informera le groupe de nos disponibilités pour essayer de prévoir une rencontre selon nos disponibilités.

6. Concurrents

Il existe déjà des applications qui permettent de prévoir et de partager des événements avec d'autres utilisateurs. Nous recherchons les applications qui sont disponibles sur IOS et Android, les applications pour être appelées des concurrents doivent représenter certaines fonctionnalités qui sont les suivantes :

- Possibilité de créer des calendriers de groupes
- Possibilité de créer des événements uniques
- Possibilité de créer des événements récurrents
- Possibilité de trouver une date pour un événement
- Disponible sur Android et IOS

6.1 Google Agenda

Google Agenda est l'application d'agenda créé par Google. Cette application est installée par défaut sur une majorité des smartphones Android. Google a lancé cette application en version Beta en avril 2006 et en version officielle en juillet 2009 pour Android. L'agenda Google est arrivé sur les appareils IOS en octobre 2019. (Google Calendar, 2020)

Cet agenda ne permet pas de créer un calendrier pour un groupe et ne permet même pas de créer de groupe.

Cet agenda permet de créer des événements uniques en permettant d'ajouter un lieu, une description, des rappels et d'ajouter en pièce jointe un document présent sur notre Google Drive. Il permet aussi de créer des événements récurrents avec les mêmes possibilités qu'un événement unique.

Pour les utilisateurs de la G suite de Google, l'application offre une fonctionnalité qui propose des horaires pour un événement. Pour utiliser cette fonctionnalité, il faut premièrement ajouter des participants à l'événement. Puis dans le champ d'horaire, il faut utiliser l'option « Trouver un horaire », l'application proposera des horaires selon les disponibilités des participants. Malheureusement pour que cette fonctionnalité soit utile, il faut que les participants partagent leur calendrier et il faut que leur calendrier soit à jour. (Federico Asara, 2016)

6.2 Calendrier pour iOS

Ce calendrier est celui développé pour les appareils d'Apple. La première version a été publiée le 10 septembre 2002 pour les ordinateurs fixes Mac. Cette application est disponible uniquement sur les appareils iOS.

Cet agenda permet la création d'événements uniques mais aussi des événements récurrents en proposant des fréquences fixées ou en permettant d'en choisir une personnalisée.

Cette application ne permet pas de créer des calendriers de groupes, mais il offre la possibilité de s'abonner aux calendriers de nos contacts, ce qui permet de voir leur emploi du temps. Nous pouvons aussi inviter des personnes à un événement que nous créons.

6.3 Doodle

Doodle est un site web qui offre des services de planifications et de sondages. Ce site a été créé en 2003 par Michael Näf pour son usage privé, puis mis à disposition du public en 2006.

Ce site web n'est pas un calendrier et n'offre donc pas la possibilité de créer des calendriers de groupes, de créer des événements uniques et de créer des événements récurrents. Cependant, il permet de trouver une date pour un événement et est disponible sur Android et iOS puisque c'est un site web.

6.4 NeedToMeet

NeedToMeet est disponible en site web, mais aussi en application. Le but de ce site est de faciliter la prise de rendez-vous. Les fonctionnalités sont proches de celles proposées par Doodle. Ce service ne permet donc pas de créer des calendriers, des événements récurrents ou uniques. Cependant tout comme Doodle, il permet de proposer des moments pour un événement et les participants peuvent renseigner leurs disponibilités.

6.5 Heja

Heja est une application centrée sur les équipes de sport. Elle permet de créer une équipe et de prévoir les entraînements et les matchs. Les membres d'une équipe peuvent indiquer s'ils seront présents lors d'un entraînement ou d'un match. Cette application ne permet pas de trouver une date pour un événement.

6.6 TimeTree

TimeTree est une application qui permet de partager des événements avec d'autres utilisateurs. Afin de partager des événements, il faut créer un calendrier dans lequel nous pouvons inviter des utilisateurs. En créant un événement dans ce calendrier partagé les autres membres pourront voir et rejoindre un événement. Cette application ne propose pas de fonctionnalité permettant de trouver un horaire pour un événement selon les disponibilités des autres membres du calendrier.

6.7 Résumé des concurrents

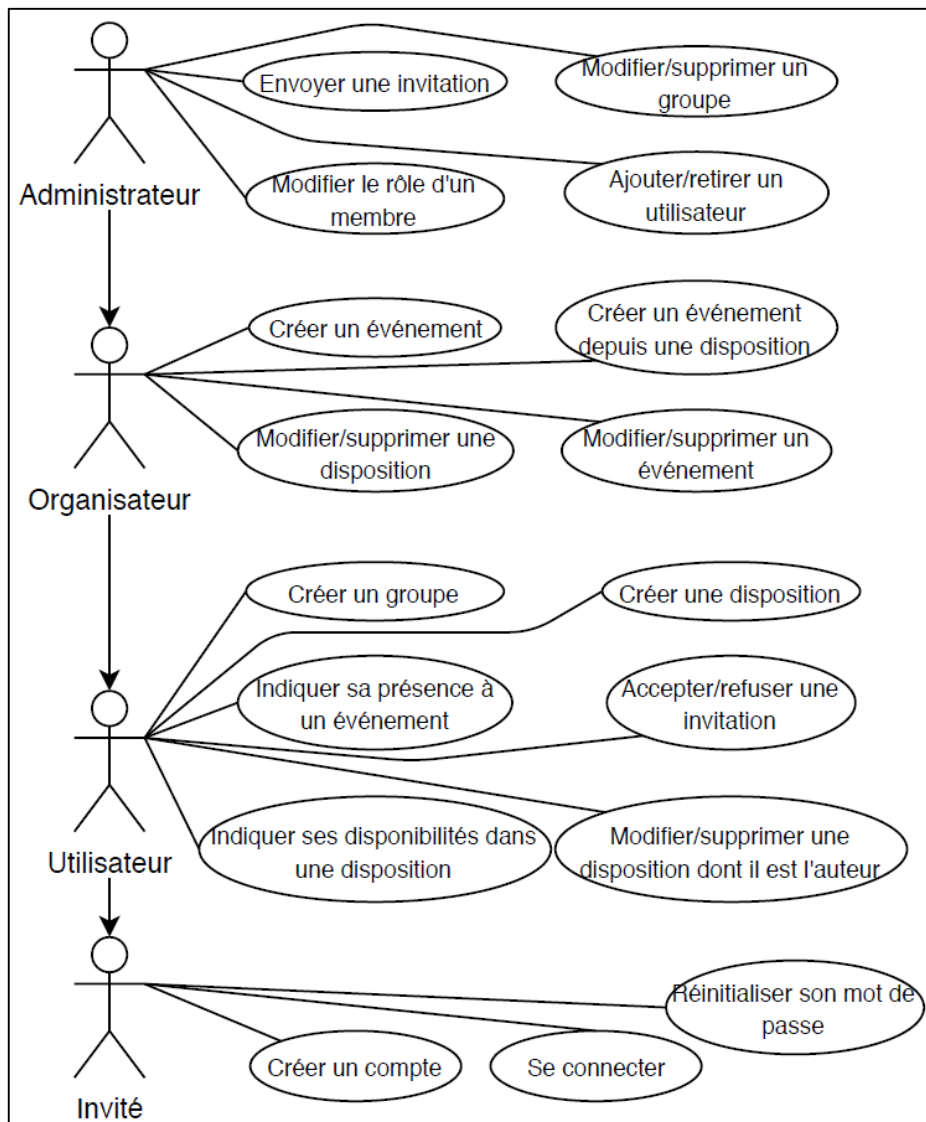
Tableau 2 : Résumé des concurrents

	Google Agenda	iCal	Doodle	NeedToMeet	Heja	TimeTree
Calendriers de groupes	Non	Non	Non	Non	Oui	Oui
Événements uniques	Oui	Oui	Non	Non	Oui	Oui
Événements récurrents	Oui	Oui	Non	Non	Oui	Oui
Trouver une date	Oui - payant	Non	Oui	Oui	Non	Non
Android et iOS	Oui	Non	Oui	Oui	Oui	Oui

7. Fonctionnalités de haut niveau

Voici un digramme représentant les cas d'utilisation ou use-cases de l'application.

Figure 9 : Use-case de l'application



(Hervé Neuenschwander, 2020)

7.1 Use-cases

7.1.1 Créer un compte

Lorsque l'utilisateur démarre l'application il arrive sur la page de connexion. Depuis cette page il a la possibilité de naviguer vers la page de création de compte. Sur cette page il doit renseigner son adresse email, un pseudo, son mot de passe et il doit valider son mot de passe en le rentrant une seconde fois. En cliquant sur le bouton pour créer un compte, la vérification des champs est effectuée. Si les champs ne sont pas remplis correctement un message en informe l'utilisateur. S'ils sont remplis correctement

l'utilisateur est automatiquement connecté et redirigé vers la page d'accueil de l'application. Il est possible de retourner à la page de connexion depuis cette page.

7.1.2 Connexion

Sur la page de connexion l'utilisateur doit entrer son email et son mot de passe. Si les données sont correctes, il est connecté et redirigé vers la page d'accueil. Si les données ne correspondent pas, un message informe l'utilisateur et il peut réessayer.

7.1.3 Réinitialiser son mot de passe

Sur la page de connexion l'utilisateur doit renseigner son email et peut cliquer sur un lien pour réinitialiser son mot de passe. Si l'adresse email est vide, le champ d'adresse email est souligné en rouge pour indiquer que l'adresse email doit être remplie. Si l'adresse email est remplie mais que l'email n'est pas dans la base de donnée un message en informe l'utilisateur. Si l'adresse email est valide, un message informe l'utilisateur qu'un email de récupération lui a été envoyé.

7.1.4 Créer un groupe

Un utilisateur authentifié peut se rendre dans l'onglet groupe et créer un groupe. Dans ce formulaire il doit renseigner un nom et une catégorie, il peut rajouter une description mais ce n'est pas obligatoire. En cliquant sur le bouton créer, les champs sont vérifiés. Si un nom et une catégorie sont indiqués le groupe est créé et l'utilisateur en devient automatiquement administrateur. Si un nom de groupe n'est pas indiqué un message informe l'utilisateur que le champ doit être rempli. Une fois le groupe créé, l'utilisateur est redirigé vers la liste des groupes.

7.1.5 Créer une disposition

Si l'utilisateur est membre d'au minimum un groupe, il a la possibilité de créer une disposition depuis la page d'accueil en cliquant sur le bouton « Indiquer mes dispositions ». Dans cet écran il choisit un groupe, un nom, peut indiquer une description et indiquer ses disponibilités pour les 6 prochains jours. Le champ du nom est obligatoire. Lorsque l'utilisateur clique sur le bouton « Créer », les champs sont vérifiés. Si un nom est rempli alors la disposition est créée et un email est envoyé aux membres du groupe pour les en avertir. Sinon un message informe l'utilisateur de remplir les champs obligatoires. Après la création de la disposition l'utilisateur est redirigé vers la page d'accueil.

7.1.6 Accepter et refuser une invitation

Lorsqu'une ou plusieurs invitations sont en attente, un bouton apparaît sur la page d'accueil pour les consulter. L'utilisateur a le choix entre accepter l'invitation ou la

refuser. Si l'utilisateur accepte, l'invitation est supprimée et il est ajouté au groupe. S'il refuse, l'invitation est simplement supprimée. S'il n'y a plus d'invitation en attente, l'utilisateur est redirigé vers la page d'accueil.

7.1.7 Indiquer sa présence à un événement

Si l'utilisateur est membre d'au moins un groupe et qu'il existe au moins un événement dans ses groupes alors l'utilisateur voit ces événements sur la page d'accueil et il peut y indiquer sa présence. Il peut indiquer sa présence sans ouvrir l'événement ou en cliquant dessus pour y voir les détails. Dans les deux cas, il y a trois boutons pour trois possibilités de présence : « Présent », « Peut-être » et « Absent ». En cliquant sur un de ces boutons la présence de l'utilisateur sera enregistrée. L'utilisateur peut changer de présence à tout moment et autant de fois qu'il le souhaite.

7.1.8 Indiquer ses disponibilités dans une disposition

Si l'utilisateur est membre d'au moins un groupe et qu'il existe au moins une disposition dans ses groupes alors l'utilisateur peut cliquer sur le bouton « Nouvelles dispositions ». En cliquant sur ce bouton l'utilisateur voit une liste des dispositions par lesquelles il est concerné. En cliquant sur une des dispositions, il en voit les détails et peut indiquer ses disponibilités. Les disponibilités sont enregistrées quand l'utilisateur appuie sur le bouton sauvegarder. Il peut modifier autant de fois qu'il veut ses disponibilités.

7.1.9 Modifier et supprimer une disposition dont il est l'auteur

En allant sur les détails d'une disposition l'utilisateur peut, s'il en est l'auteur ou s'il a le rôle « Organisateur » dans le groupe, modifier ou supprimer cette disposition. En cliquant sur le bouton modifier il est redirigé vers un écran d'où il peut modifier le nom et la description. Au moment de cliquer sur le bouton « Enregistrer », l'application vérifie que le nom ne soit pas vide. Si le nom est vide un message informe l'utilisateur qu'il doit être rempli. Sinon les modifications sont enregistrées et il est redirigé vers la page de détails.

En cliquant sur le bouton supprimer, un message demande si l'utilisateur est sûr de vouloir supprimer cette disposition. S'il clique sur oui la disposition est supprimée et il est redirigé vers la page d'accueil. S'il clique sur non l'action est annulée et rien ne se passe.

7.1.10 Créer un événement

Si l'utilisateur est organisateur d'au moins un groupe, il peut créer un événement. Sur l'écran de création d'un événement il peut indiquer un nom, une description, le groupe, une date, une heure de début et de fin ou indiquer que l'événement dure toute la journée, un nombre de participants minimum et maximum et la possibilité d'autoriser les

commentaires. Il est aussi possible d'indiquer que l'événement est récurrent et dans ce cas il faut indiquer la fréquence en jours et une date de fin de récurrence. Le champ du nom peut être souligné en rouge si il n'est pas rempli, le champ d'heure de fin peut lui aussi être souligné si l'heure de début est après l'heure de fin. Si le nombre minimum de participants est plus grands que celui maximum, ce champ est aussi souligné en rouge. Si l'événement est récurrent le champ pour la date de fin peut être souligné en rouge s'il est avant la date de l'événement. Si les données sont correctes, l'utilisateur peut créer l'événement et il est ensuite redirigé vers la page d'accueil.

7.1.11 Créer un événement depuis une disposition

Un organisateur a la possibilité de créer un événement depuis une disposition. Cette fonctionnalité est accessible via l'écran de détail d'une disposition. Le formulaire reprend le nom et la description donnés à la disposition. L'organisateur peut indiquer une heure de début et de fin ainsi qu'un nombre minimum et maximum de participants. Il y a aussi une liste déroulante avec les 6 dates possibles dans la disposition. Lorsque l'organisateur clique sur le bouton « Créer » l'application vérifie que l'heure de fin soit après l'heure de début, qu'un nom soit renseigné et que le nombre maximum de participants soit plus grand que le nombre minimum. Si ces conditions sont respectées, l'événement est créé. Sinon les champs qui ne sont pas bien remplis sont soulignés en rouge.

7.1.12 Modifier et supprimer un événement

Si l'utilisateur est organisateur du groupe dans lequel est l'événement alors il peut le modifier et le supprimer. Ces fonctions sont disponibles depuis l'écran de détail d'un événement.

Lors de la modification il est possible de modifier le nom, la description, la date, les heures de fin et de début ainsi que le nombre maximum et minimum de participants. Si l'événement sélectionné pour la modification fait partie d'événements récurrents, il est possible de modifier tous les événements d'un coup. Mais dans ce cas il n'est plus possible de modifier la date. La validation des champs est la même que pour la création d'un événement.

Lors de la suppression d'un événement, une fenêtre demande la confirmation de la suppression. Si l'événement sélectionné fait partie d'événements récurrents il est possible de supprimer tous les événements d'un coup.

7.1.13 Modifier et supprimer une disposition

Si l'utilisateur est organisateur du groupe de la disposition, il peut modifier ou supprimer une disposition. Les actions sont les mêmes que dans le chapitre 7.1.9

7.1.14 Envoyer une invitation

Si l'utilisateur est administrateur d'un groupe il peut inviter un autre utilisateur dans le groupe. Cette option est dans l'onglet « Groupes » dans les détails d'un groupe. En cliquant sur le bouton pour ajouter un utilisateur, un champ apparaît pour indiquer l'adresse email de la personne à inviter. Si le champ n'est pas rempli et que l'administrateur clique sur le bouton de validation, le champ disparaît et aucune invitation n'est envoyée.

Si le champ n'est pas rempli avec une adresse email, ou que l'adresse email n'est pas connue de l'application, un message indique à l'administrateur qu'aucun utilisateur n'est inscrit avec cette adresse email.

Si l'adresse email est connue de l'application, un message indique qu'une invitation a bien été envoyée.

7.1.15 Modifier et supprimer un groupe

Il est possible de modifier un groupe depuis les détails pour les administrateurs. Ils peuvent modifier le nom, la description et la catégorie. Le nom et la catégorie sont obligatoires. Si le formulaire est correctement rempli, les modifications sont enregistrées au moment où l'administrateur clique sur « Enregistrer » autrement un message indique que le nom et la catégorie doivent être renseignés. Après que les modifications aient été enregistrées, l'administrateur est redirigé vers la page des détails du groupe.

Il est aussi possible pour un administrateur de supprimer un groupe depuis la page des détails d'un groupe. En cliquant sur le bouton pour supprimer, un message indique à l'administrateur que les événements et les dispositions liés au groupe seront aussi supprimés. L'administrateur peut confirmer la suppression ou bien annuler l'action. Si l'administrateur confirme la suppression il est redirigé vers la page d'accueil.

7.1.16 Modifier le rôle d'un membre

Depuis l'écran de détails d'un groupe il est possible pour les administrateurs de ce groupe de modifier le rôle d'un autre utilisateur. En cliquant sur les options à côté du nom du membre un menu apparaît permettant de définir l'utilisateur comme un autre rôle que celui qu'il a actuellement.

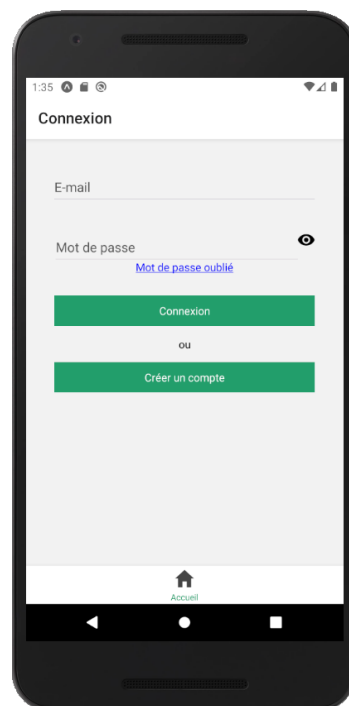
7.1.17 Retirer un membre

Comme pour modifier le rôle d'un membre il est possible aux administrateurs de retirer un membre du groupe. L'option se trouve aussi dans les options à côté du nom de l'utilisateur. Un message demande la confirmation de la suppression de l'utilisateur du groupe. Si l'administrateur annule, l'action est annulée. S'il confirme, l'utilisateur est retiré du groupe, de toutes les dispositions et tous les événements du groupe.

7.2 Vues

7.2.1 Écran de connexion

Figure 10 : Écran de connexion

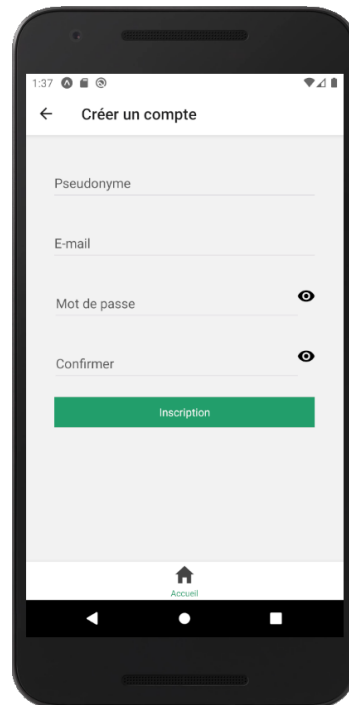


(Hervé Neuenschwander, 2020)

Cet écran est l'écran par défaut lorsque l'utilisateur ouvre l'application sur son smartphone. Le formulaire de connexion contient un champ pour indiquer l'adresse email et un deuxième pour le mot de passe. Il est possible de cliquer sur l'icône de l'œil pour afficher son mot de passe. Le lien « Mot de passe oublié » permet d'envoyer un email de récupération de mot de passe vers l'email renseigné dans le premier champ. Le bouton « Connexion » permet de faire une tentative de connexion avec les informations du formulaire. Le bouton « Créer un compte » permet d'accéder au formulaire de création de compte.

7.2.2 Écran de création de compte

Figure 11 : Écran de création de compte

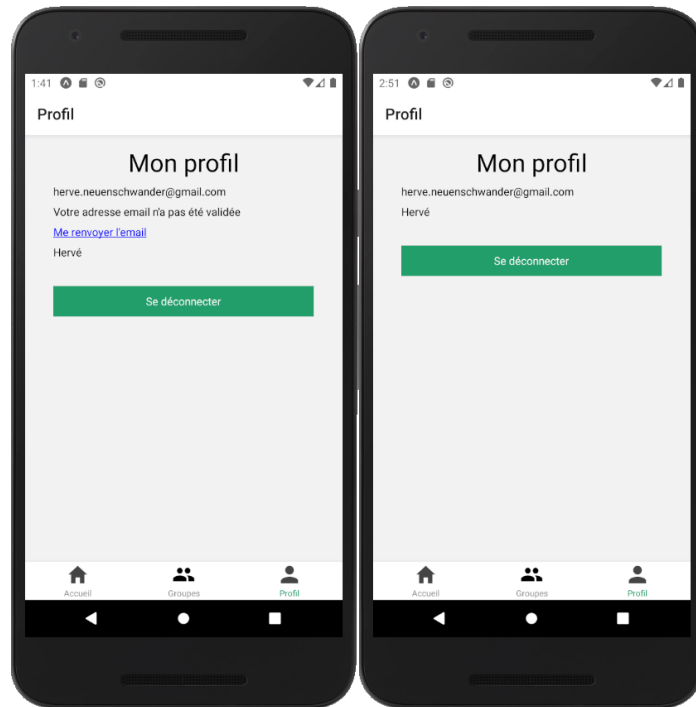


(Hervé Neuenschwander, 2020)

On peut accéder à cet écran via le bouton « Créer un compte » de l'écran de connexion. Le formulaire est composé de 4 champs. Le premier est pour le pseudonyme de l'utilisateur, le deuxième est pour son adresse email. Le troisième est pour son mot de passe et le quatrième pour la confirmation du mot de passe. Les deux icônes d'œil permettent d'afficher le mot de passe de leurs champs respectifs. Le bouton « Inscription » permet de faire une tentative de création de compte avec les données du formulaire. Il est possible de revenir sur l'écran de connexion du chapitre 7.2.1

7.2.3 Écran de profil

Figure 12 : Écran de profil

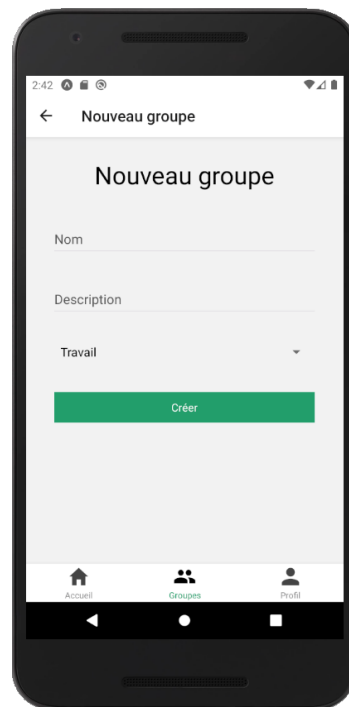


(Hervé Neuenschwander, 2020)

Cet écran est l'unique écran de l'onglet « Profil ». Il est composé d'un titre et des informations de l'utilisateur. Dans le cas de cette image, l'adresse email n'a pas été validée. Un message est donc affiché et le lien « Me renvoyer l'email » permet de renvoyer l'email de confirmation. Si l'email est validé ces deux lignes ne sont pas affichées. Le bouton « Se déconnecter » permet de se déconnecter et redirige vers l'écran « Connexion » du chapitre 7.2.1

7.2.4 Écran de création de groupe

Figure 13 : Écran de création de compte

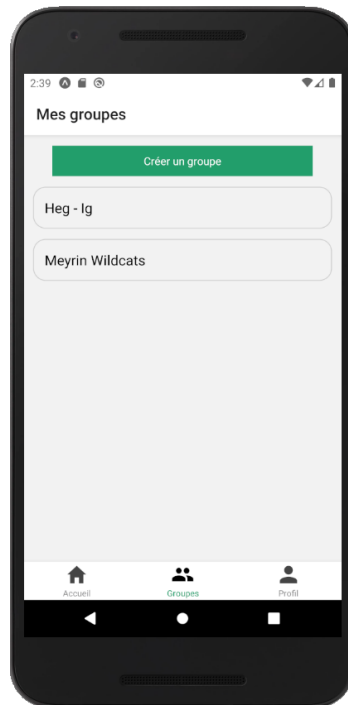


(Hervé Neuenschwander, 2020)

Cet écran est accessible via le bouton « Créer un groupe » sur l'écran de liste des groupes. Ce formulaire est composé de deux champs et d'une liste déroulante. Le premier champ est pour le nom du groupe et le deuxième pour la description. La liste déroulante permet de choisir une catégorie. Le bouton « Créer » permet de faire une tentative de création de compte avec les informations du formulaire.

7.2.5 Écran de liste des groupes

Figure 14 : Écran de liste des groupes

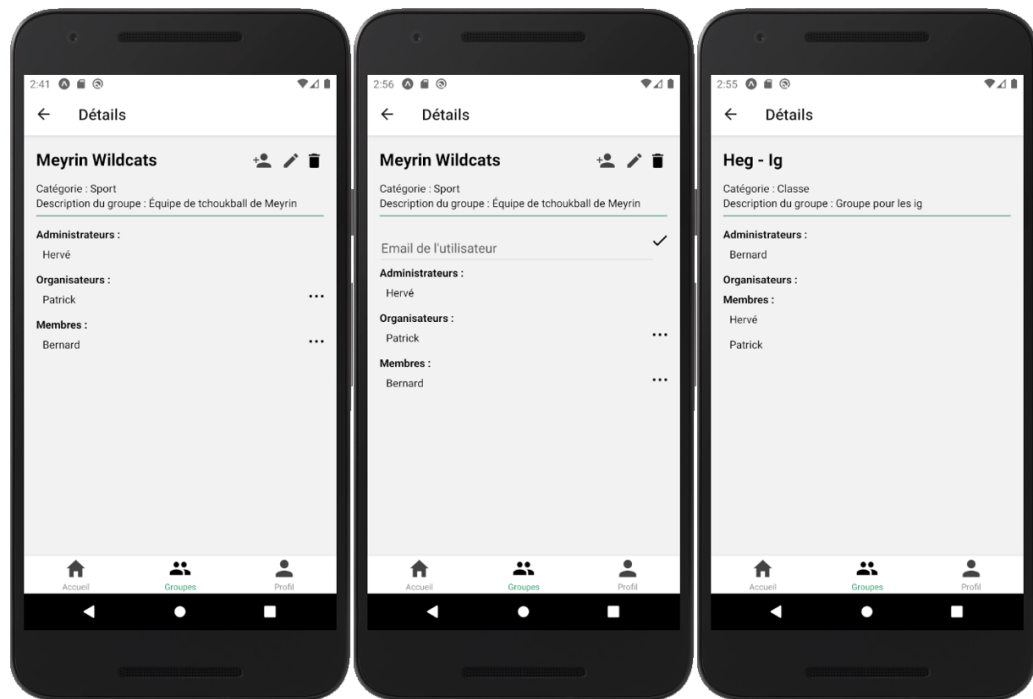


(Hervé Neuenschwander, 2020)

Cet écran est l'écran par défaut de l'onglet « Groupes ». Le bouton « Créer un groupe » permet d'accéder à l'écran de création de groupe. En dessous les groupes de l'utilisateur sont affichés sous forme de liste. En cliquant sur un des groupes l'utilisateur peut accéder à l'écran de détails d'un groupe.

7.2.6 Écran de détails d'un groupe

Figure 15 : Écrans de détails d'un groupe

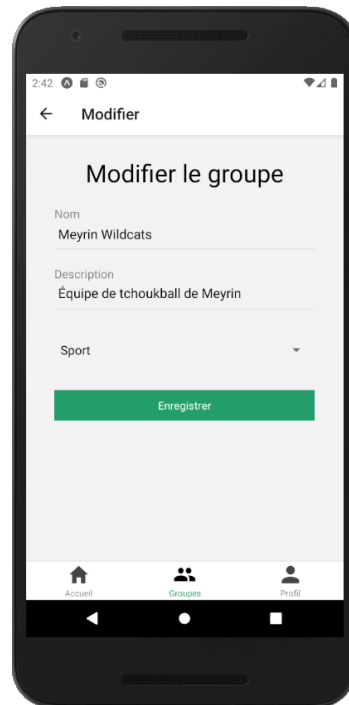


(Hervé Neuenschwander, 2020)

L'écran de détails d'un groupe est accessible en cliquant sur un groupe dans l'écran liste des groupes. Il est composé des informations concernant le groupe et en dessous, d'une liste des membres du groupe. L'administrateur peut ajouter un membre, modifier le groupe ou le supprimer. Quand l'administrateur clique sur l'icône pour ajouter un membre un champ apparaît pour indiquer l'email du nouvel utilisateur. L'icône de crayon permet d'accéder à l'écran de modification de groupe. L'icône de poubelle permet de supprimer le groupe. L'administrateur peut aussi cliquer sur le bouton d'options à côté du nom d'un utilisateur pour afficher un menu qui permet de modifier le rôle de l'utilisateur ou de le retirer du groupe. Les membres du groupe avec un autre rôle n'ont pas ces options.

7.2.7 Écran de modification de groupe

Figure 16 : Écran de modification de groupe

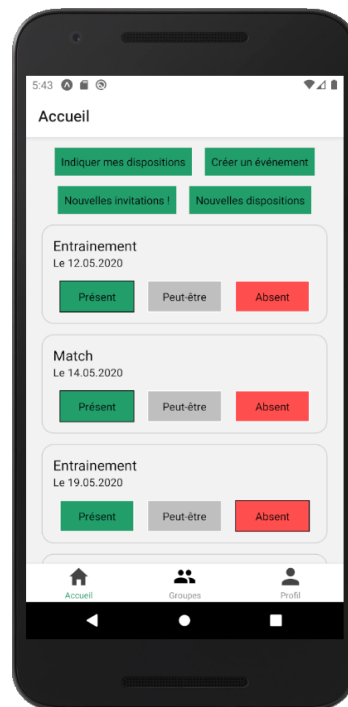


(Hervé Neuenschwander, 2020)

Cet écran est accessible pour les administrateurs en cliquant sur l'icône de crayon sur l'écran de détails d'un groupe. Le formulaire est pré-rempli avec les données actuelles du groupe modifié. L'administrateur peut changer ces informations. Le bouton « Enregistrer » permet de faire une tentative de modification du groupe.

7.2.8 Écran d'accueil

Figure 17 : Écran d'accueil

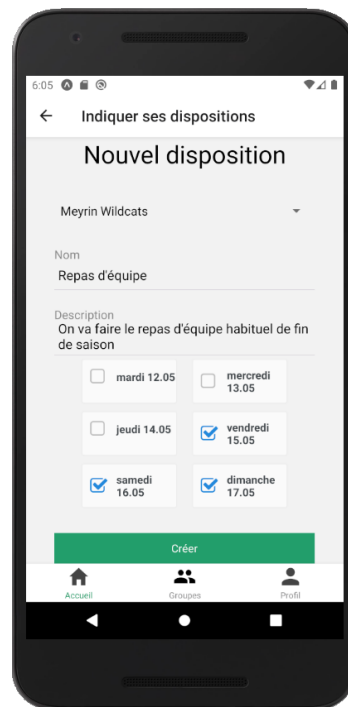


(Hervé Neuenschwander, 2020)

Cet écran est celui par défaut après la connexion de l'utilisateur. En haut, les boutons permettent d'accéder vers d'autres écrans. En dessous les événements concernant l'utilisateur sont affichés.

7.2.9 Écran de création de disposition

Figure 18 : Écran de création de disposition

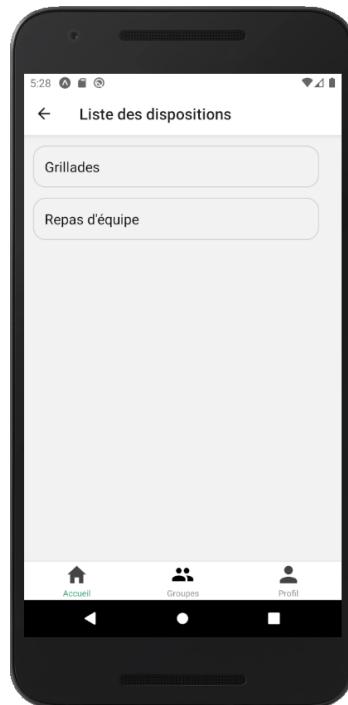


(Hervé Neuenschwander, 2020)

Cet écran est accessible via le bouton « Indiquer mes dispositions » sur la page d'accueil. En premier il y a une liste déroulante qui permet de choisir dans quel groupe la disposition sera créée. Ensuite on peut indiquer le nom de l'évènement et une description. En dessous l'utilisateur peut indiquer ses propres disponibilités pour les 6 jours à venir. Le bouton « Créer » permet de faire une tentative de création de la disposition avec les données du formulaire.

7.2.10 Écran de liste des dispositions

Figure 19 : Écran de liste des dispositions

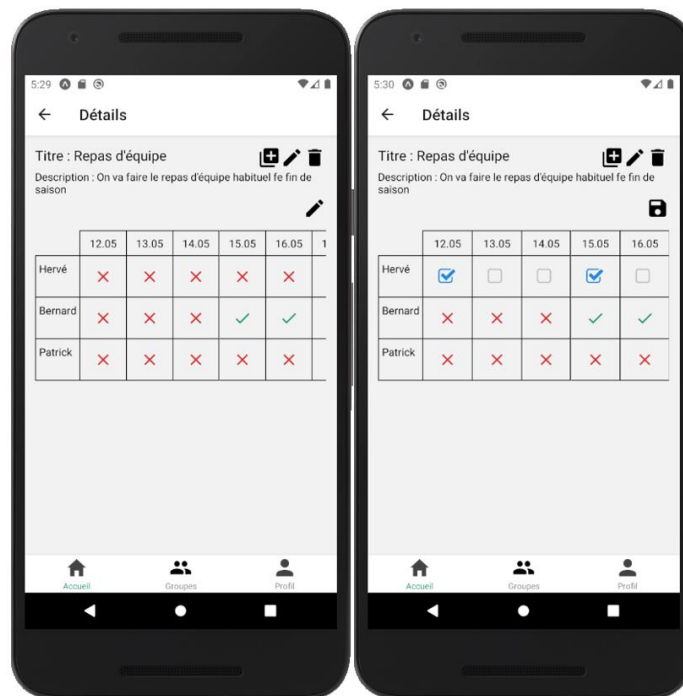


(Hervé Neuenschwander, 2020)

Cet écran est accessible en cliquant sur le bouton « Nouvelles dispositions » sur l'écran d'accueil. Il affiche les dispositions sous forme de liste. En cliquant sur une des dispositions l'utilisateur accède à l'écran de détails de cette disposition.

7.2.11 Écran de détails de disposition

Figure 20 : Écran de détails de disposition

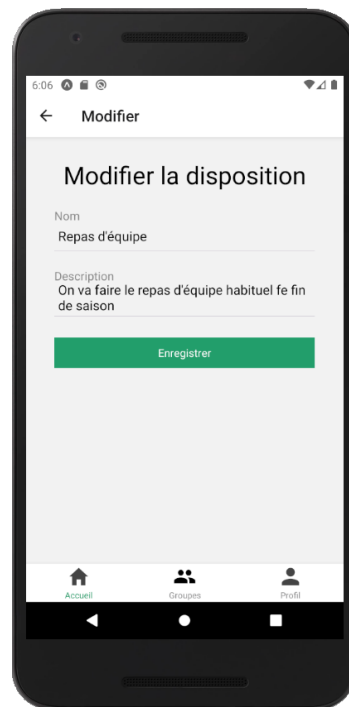


(Hervé Neuenschwander, 2020)

Cet écran est accessible en cliquant sur une des dispositions présentes dans l'écran de liste des dispositions. Il affiche les détails d'une disposition et il est possible de modifier ses disponibilités en cliquant sur l'icône de crayon en dessus du tableau. La ligne de l'utilisateur actif est remplacée par des cases à cocher, ensuite en cliquant sur l'icône de sauvegarde les modifications sont enregistrées. Si l'utilisateur est organisateur du groupe ou le créateur de cette disposition, il peut modifier ou supprimer la disposition en cliquant sur l'icône de crayon en haut ou sur l'icône de poubelle. Si l'utilisateur est un organisateur alors il peut créer un évènement depuis cette disposition en cliquant sur l'icône avec un +.

7.2.12 Écran de modification de disposition

Figure 21 : Écran de modification de disposition

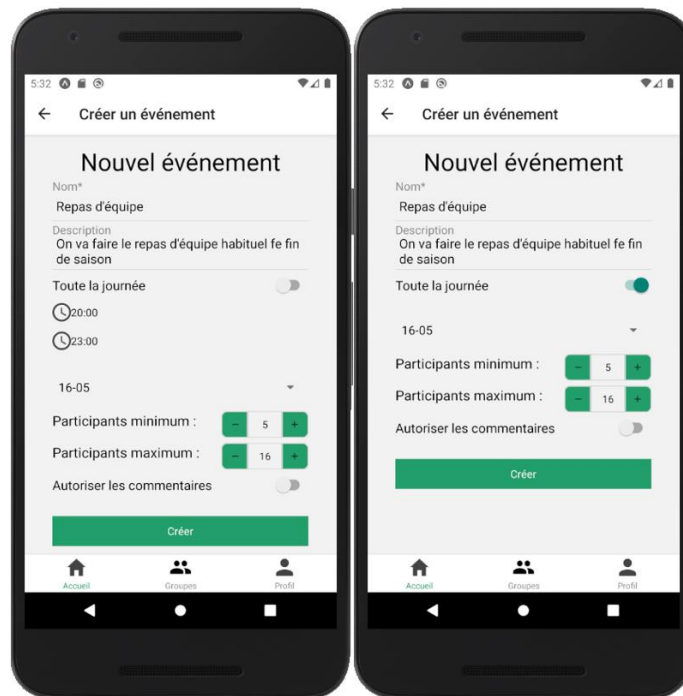


(Hervé Neuenschwander, 2020)

Cet écran est accessible en cliquant sur l'icône de crayon sur l'écran de détails d'une disposition. Le formulaire est pré-rempli avec le nom et la description de la disposition sélectionnée. L'utilisateur peut modifier ces deux champs et en cliquant sur le bouton « Enregistrer » il peut enregistrer ces modifications.

7.2.13 Écran de création d'évènement d'après une disposition

Figure 22 : Écran de création d'évènement d'après une disposition

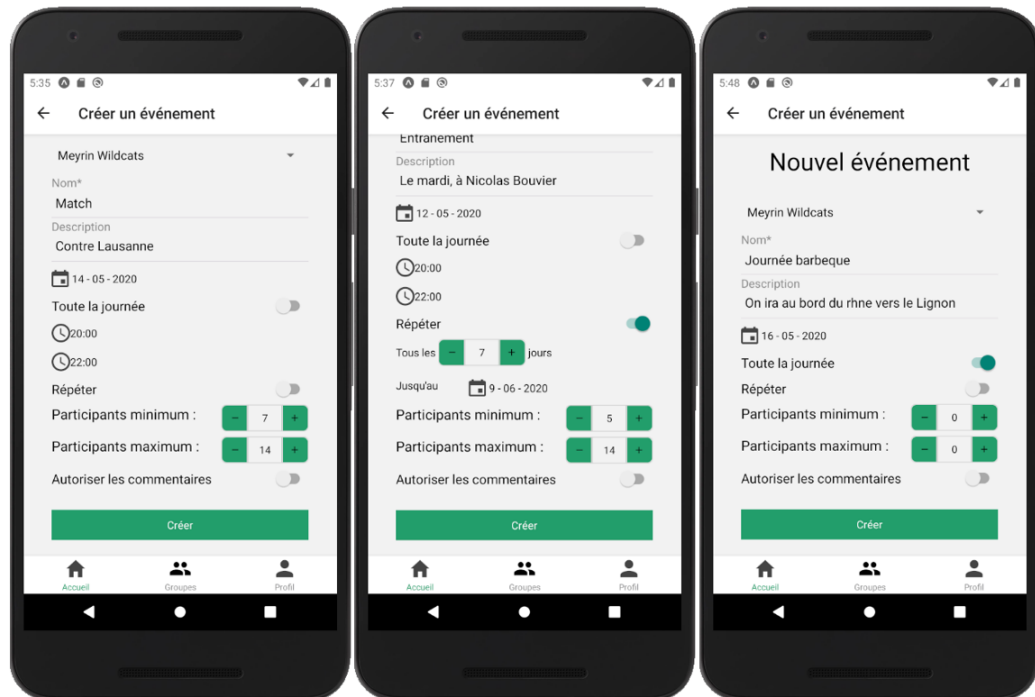


(Hervé Neuenschwander, 2020)

Cet écran est accessible en cliquant sur l'icône avec un + dans l'écran de détails d'une disposition. Il permet de créer un évènement à partir d'une disposition. Le nom et la description sont pré-remplis avec les données de la disposition. Il faut ensuite renseigner une heure de début et de fin ou indiquer que l'évènement dure toute la journée. Pour choisir la date, il y a une liste déroulante qui reprend les 6 jours possibles dans la disposition. Il y a aussi la possibilité de choisir un nombre de participants maximum et minimum. Le bouton « Créer » permet de faire une tentative de création de l'évènement.

7.2.14 Écran de création d'événement

Figure 23 : Écran de création d'événement

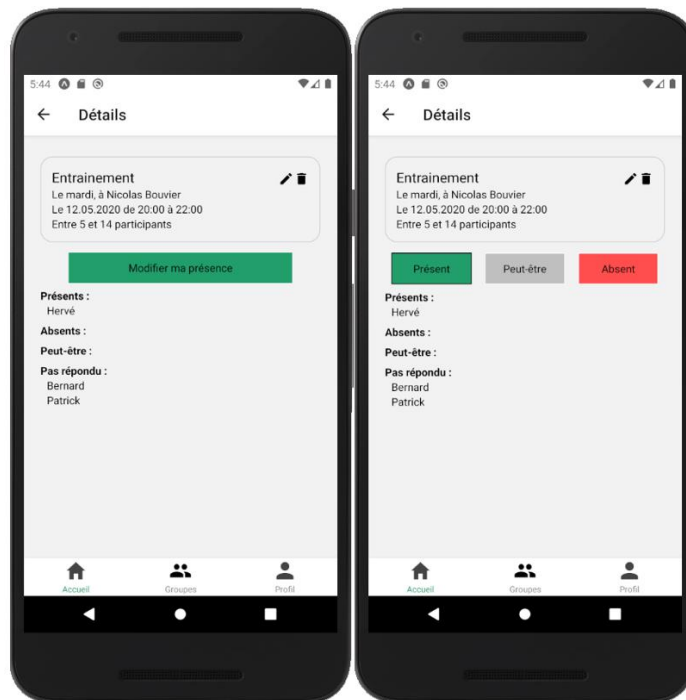


(Hervé Neuenschwander, 2020)

Cet écran est accessible pour les organisateurs en cliquant sur le bouton « Créer un évènement » sur la page d'accueil. C'est l'écran le plus complet de l'application. Il y a une liste déroulante pour choisir le groupe, un champ pour le nom et un autre pour la description. Ensuite l'icône de calendrier permet de choisir une date et la section suivante permet de choisir un horaire ou de d'indiquer que l'évènement dure toute la journée. Il est ensuite possible d'indiquer que l'évènement est récurrent et dans ce cas un champ permettant de choisir la fréquence et un deuxième calendrier apparaissent. Le deuxième calendrier permet de choisir jusqu'à quand l'évènement doit se répéter. Ensuite il y a la possibilité de choisir un nombre de participants minimum et maximum. Le bouton « Créer » permet de faire une tentative de création du ou des événements.

7.2.15 Écran de détails d'évènement

Figure 24 : Écran de détails d'évènement

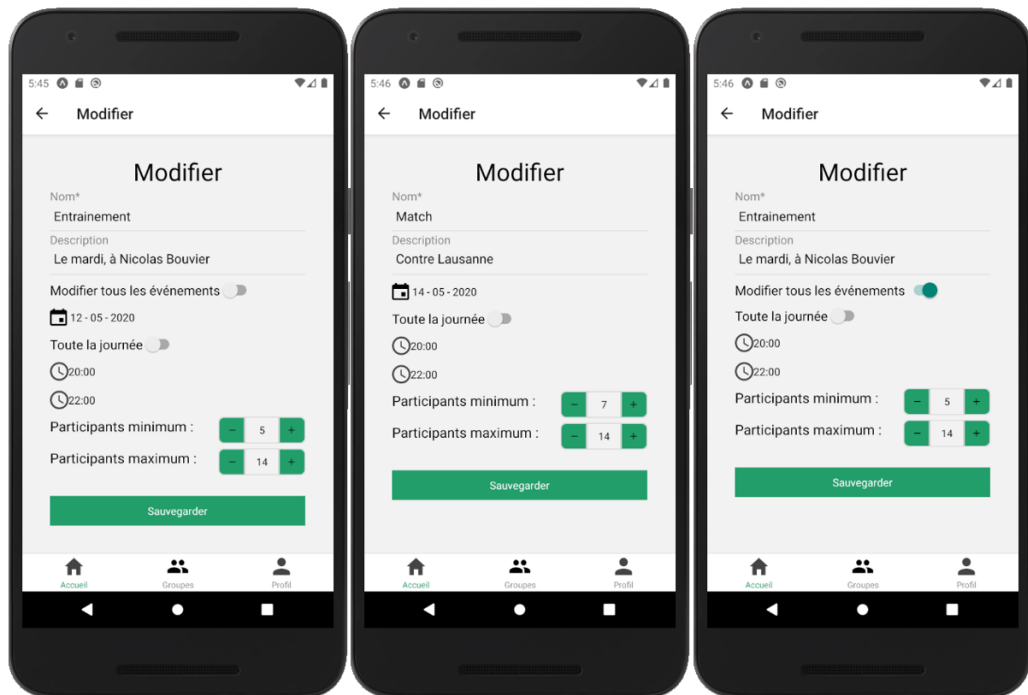


(Hervé Neuenschwander, 2020)

Cet écran est accessible en cliquant sur un des événements affichés sur l'écran d'accueil. En première partie l'évènement est résumé et en deuxième partie nous avons la liste des présences. En cliquant sur le bouton « Modifier ma présence » le bouton est remplacé par trois autres boutons qui permettent d'indiquer sa présence. Si nous avons déjà indiqué notre présence le bouton concerné est entouré de noir. Si l'utilisateur est un organisateur il peut modifier l'évènement en cliquant sur le crayon ou le supprimer en cliquant sur la poubelle.

7.2.16 Écran de modification d'évènement

Figure 25 : Écran de modification d'évènement

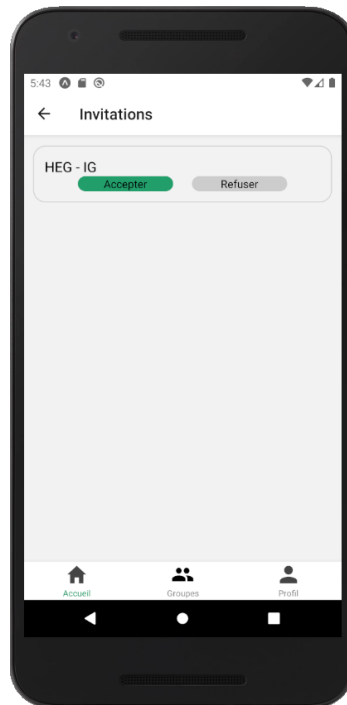


(Hervé Neuenschwander, 2020)

Cet écran est accessible en cliquant sur l'icône de crayon dans les détails d'un évènement. Tout le formulaire est pré-rempli avec les informations de l'évènement en cours de modification. Cet écran reprend une grande partie de l'écran de création d'un évènement mais offre une possibilité supplémentaire si l'évènement fait partie d'un évènement récurrent. Dans ce cas nous pouvons choisir d'appliquer les modifications sur tous les évènements et ne pouvons donc plus modifier la date de l'évènement. Le bouton « Sauvegarder » permet d'enregistrer les modifications.

7.2.17 Écran de liste d'invitations

Figure 26 : Écran de liste d'invitations



(Hervé Neuenschwander, 2020)

Cet écran est accessible en cliquant sur le bouton « Nouvelles invitations ! » de l'écran d'accueil. Ce bouton est visible uniquement si des invitations sont en attente. Cet écran affiche les invitations sous forme de liste. Chaque invitation comporte un bouton « Accepter » et un autre « Refuser ». En cliquant sur le premier bouton l'utilisateur est ajouté au groupe et à ses dispositions et événements et l'invitation est supprimée. En cliquant sur le deuxième bouton, l'invitation est supprimée. Lorsqu'il n'y a plus d'invitations en attente l'utilisateur est redirigé vers la page d'accueil.

8. Perspectives d'avenir

L'application est utilisable dans l'état actuel mais plusieurs fonctionnalités proposées par le mandant seront implémentées à l'avenir. C'était un point dont nous avons déjà discuté avec le mandant et lorsqu'il m'avait proposé son projet, je lui avais indiqué les fonctionnalités que je développerai dans le cadre du travail de Bachelor et celle qui seraient développées après.

8.1 S'inscrire en utilisant les réseaux sociaux

Afin d'améliorer l'expérience utilisateur, il sera possible de s'inscrire en utilisant Facebook ou un compte Google. Cela permettra à l'utilisateur de s'inscrire plus rapidement et de ne pas forcément avoir à remplir un formulaire complet pour pouvoir s'inscrire.

8.2 Synchroniser les calendriers

Cette future fonctionnalité permettra de ne pas avoir de conflits avec d'autres événements qui sont dans d'autres calendriers. Cela facilitera l'utilisation de l'application et permettra aux utilisateurs de ne pas se tromper dans leurs disponibilités.

8.3 Envoie de notification dans l'application

Dans l'état actuel il n'est pas possible de faire des notifications directement dans l'application. Le mandant a donc proposé de faire des notifications par email. Afin de ne pas mélanger des notifications avec les emails des utilisateurs, les notifications par email disparaîtront au profit des notifications dans l'application qui sont beaucoup plus répandues et moins envahissantes.

8.4 Chats dans les événements

Afin de faciliter la préparation d'un événement ou de demander des informations supplémentaires à l'organisateur, il sera possible de discuter dans l'écran de détails d'un événement. Cette fonctionnalité peut être activée ou pas au moment de la création. Il est déjà possible de choisir cela lors de la création mais il n'est pas encore possible de chatter. Cela va donc être rajouté dans une future version.

9. Utilisation des frameworks

9.1 Apprentissages

9.1.1 React Native

Ce travail m'a permis d'être plus confiant avec React Native. Avant de commencer j'avais déjà développé un petit peu, mais ce travail est de loin le plus complet que j'ai réalisé avec ce framework. J'ai aussi appris à utiliser Redux qui permet d'avoir un état global dans toute l'application.

Afin de pouvoir naviguer d'un écran à un autre j'ai utilisé « React Navigation ». Cette bibliothèque permet de créer des routes et d'indiquer à ces routes quel composant de notre application afficher. Ensuite nous pouvons naviguer vers cette route et le bon composant sera affiché. React Navigation permet aussi de créer des onglets ou un menu tiroir qui s'ouvre depuis le bord de l'écran. Dans le cadre de cette application j'ai utilisé trois onglets qui se trouvent en bas de l'écran.

Il était nécessaire pour cette application de pouvoir envoyer des notifications. Dans le cadre de ce travail l'application ne peut pas être installée sur un smartphone, il est donc impossible de recevoir des notifications si Expo n'est pas en cours d'exécution. Pour pallier à ce problème nous avons décidé avec le mandant d'utiliser des mails comme moyen de notification. L'envoi d'email est très simplifié avec firebase grâce à l'extension « Trigger Email ». Cette extension observe une collection de notre base de données et essaye d'envoyer les mails qui y sont stockés sous forme de document. Le document doit avoir un champ « to » qui contient l'adresse email du destinataire. Il faut aussi un champ message qui contient trois champs : « subject » qui est l'objet, un champ « text » et un champ « html » qui sont le corps de l'email.

9.1.2 Flutter

Lors du développement du prototype avec Flutter j'ai appris à utiliser Flutter car je n'avais jamais utilisé ce framework auparavant.

9.1.2.1 Prototype

Un prototype a été réalisé avec Flutter pour découvrir ce framework, mieux le comprendre et ainsi pouvoir comparer les deux frameworks. J'ai voulu intégrer Firebase dans le prototype car je pense que c'est une partie qui présente de la complexité et surtout il est primordial pour l'application d'être lié à cette base de données. Pour ce faire j'ai donc suivi un tutoriel sur le site « medium.com » qui explique comment faire une connexion et une création de compte.

9.2 Difficultés

9.2.1 React Native

J'ai rencontré deux difficultés principales lors du développement de l'application. Le premier concernait l'affichage d'icônes et le deuxième concernait la mise à jour des données en temps réel.

Sur d'autres projets auxquelles j'avais participé, il n'y avait pas eu besoin de faire de manipulations spécifiques pour afficher des icônes comme celles proposées par Material Icons ou Ionicons. Lorsque j'ai essayé pour la première fois d'afficher des icônes dans cette application, j'ai eu un message d'erreur qui disait que la police Material Icons ou Ionicons suivant laquelle j'essayais d'utiliser, n'était pas une police du système. Il m'a donc fallu les charger dans le fichier App.js et ne pas faire le rendu de l'application tant que ces polices n'étaient pas chargées. Puis lorsque j'ai testé l'application sur un appareil iOS certaines icônes ne s'affichaient pas correctement. J'ai donc dû remplacer certaines icônes.

Le deuxième problème est apparu lorsque j'ai implémenté la modification d'un groupe. Après avoir fait la modification du groupe, l'administrateur est automatiquement redirigé vers la page de détail de ce groupe. Malheureusement cet écran ne se mettait pas à jour avec les nouvelles données, car l'affichage se rafraichissait avant que les données aient été mises à jour dans la base de données. Pour pallier à ce problème, j'ai utilisé des « listener » qui écoutent les changements sur un ou plusieurs documents de la base de données et retournent le document modifié aussitôt. Ainsi dès que la modification est effectuée dans la base de données, l'affichage peut être modifié et mis à jour avec les nouvelles données. Après avoir découvert et implémenté cette fonctionnalité pour ce cas, j'ai pu la réutiliser dans une grande partie de l'application, pour la modification des événements et des disponibilités, mais aussi pour afficher un nouvel événement automatiquement ou pour recevoir une invitation.

9.2.2 Flutter

L'apprentissage de Flutter a été compliqué car je n'avais jamais utilisé ceci auparavant. De plus il est toujours plus facile d'apprendre avec une personne qui connaît déjà le framework que seul. Ainsi Flutter en lui-même a été une difficulté.

Il a fallu suivre la documentation officielle pour installer le nécessaire. Il m'a donc fallu installer des extensions sur Visual Studio Code et installer Android Studio pour pouvoir exécuter un émulateur. Il a ensuite fallu être capable d'exécuter le code de Visual Studio Code vers l'émulateur. J'ai rencontré plusieurs problèmes lors de cette étape qui

n'étaient pas documentés dans la documentation officielle et j'ai donc trouvé des solutions pour mes problèmes grâce à la communauté.

Ensuite il a fallu comprendre le fonctionnement de Flutter et la manière dont organiser les fichiers pour une meilleure compréhension et une séparation des responsabilités.

10. Conclusion

Ce travail a été très enrichissant car je n'avais jamais fait une application du début à la fin seul et en utilisant ces technologies. Ce projet m'a permis de me rendre compte de l'importance de la planification et de la documentation au début du projet, cela m'a aidé à pouvoir développer toute les parties de l'application dont on avait discuté avec le mandant. En ayant une vision sur le travail restant j'ai pu prioriser les tâches et savoir si je pouvais accepter ou non de nouvelles fonctionnalités proposées par le mandant au fur et à mesure des versions de l'application que je lui fournissais.

Cela m'a aussi permis de mettre en pratique et de mieux comprendre des points vus en théorie pendant ces trois dernières années, par exemple l'importance de séparer les responsabilités au sein du code afin de faciliter la compréhension.

J'ai découvert le fonctionnement des frameworks qui ont été étudiés dans ce document et cela était enrichissant. Je ne m'étais jamais intéressé au fonctionnement de certains outils que j'utilisais auparavant mais cela était très intéressant et très impressionnant aussi. Le travail fourni par les développeurs de ces frameworks est très technique et complexe, ils cherchent à optimiser chaque partie de l'outil qu'ils proposent.

Je me réjouis de continuer à travailler sur cette application avec le mandant. Les fonctionnalités à ajouter paraissent complexes et sont donc intéressantes. J'espère pouvoir proposer cette application dans les différents magasins d'applications dans le futur et pouvoir commencer une collaboration avec Monsieur Balli.

Je n'ai pas ressenti de différence de performance entre les deux frameworks, dans une application plus gourmande, il serait possible qu'une technologie soit plus performante que l'autre.

Personnellement j'ai préféré utiliser React Native car je connaissais déjà cela. Ce framework est aussi plus abouti que Flutter pour l'instant, mais dans quelques années, il serait important d'étudier à nouveau ces deux outils pour voir leurs évolutions et peut être changer d'avis personnel.

Bibliographie

Google Calendar. *Wikipedia : l'encyclopédie libre* [en ligne]. Dernière modification de la page le 6 mars 2020 à 12:21. [Consulté le 9 Mars 2020]. Disponible à l'adresse: https://en.wikipedia.org/w/index.php?title=Google_Calendar&oldid=944212776

Federico Asara, 2016. Save time with smart scheduling in Google Calendar. *Google* [en ligne]. 29 Septembre 2016. [Consulté le 9 Mars 2020]. Disponible à l'adresse: <https://blog.google/products/calendar/save-time-with-smart-scheduling-in-google-calendar/>

React Native - A framework for building native apps using React, [pas de date]. [en ligne]. [Consulté le 9 Mars 2020]. Disponible à l'adresse: <https://reactnative.dev/>

KUMAR, Saket, [27 août 2018]. How React Native Works ? | Codementor. [en ligne]. [Consulté le 10 Mars 2020]. Disponible à l'adresse: <https://www.codementor.io/@saketskumar95/how-react-native-works-mhjo4k6f3>

React Native Internals | React Made Native Easy, [pas de date]. [en ligne]. [Consulté le 10 Mars 2020]. Disponible à l'adresse: <https://www.reactnative.guide/3-react-native-internals/3.1-react-native-internals.html>

State of React Native 2018 React Native, [14 juin 2018]. [en ligne]. [Consulté le 10 Mars 2020]. Disponible à l'adresse: <https://reactnative.dev/blog/2018/06/14/state-of-react-native-2018>

React Native EU 2019: Emily Janzer - The New React Native, [1 octobre 2019]. [en ligne]. [Consulté le 10 Mars 2020]. Disponible à l'adresse: <https://www.youtube.com/watch?v=52EI0EUI6D0&feature=youtu.be&t=221>

FELDMAN, Chen, 2019. React Native — A Bridge To Project Fabric — Part 3. *Medium* [en ligne]. 30 Juillet 2019. [Consulté le 10 Mars 2020]. Disponible à l'adresse: <https://medium.com/@chenfeldmn/react-native-a-bridge-to-project-fabric-part-3-ae495794c6b0>

Hermes: A new open source JavaScript engine optimized for mobile apps, 2019. *Facebook Engineering* [en ligne]. [Consulté le 11 Mars 2020]. Disponible à l'adresse: <https://engineering.fb.com/android/hermes/>

Flutter - Showcase, [pas de date]. [en ligne]. [Consulté le 12 Mars 2020]. Disponible à l'adresse: <https://flutter.dev/showcase>

WALRATH, Kathy, 2020. Dart asynchronous programming: Isolates and event loops. *Medium* [en ligne]. 25 Juillet 2019. [Consulté le 13 Mars 2020]. Disponible à l'adresse: <https://medium.com/dartlang/dart-asynchronous-programming-isolates-and-event-loops-bffc3e296a6a>

KURIAN, George T., 2019. Flutter: How cross-platform made possible. *Medium* [en ligne]. 17 Décembre 2019. [Consulté le 13 March 2020]. Disponible à l'adresse: <https://medium.com/@georgetk1996/flutter-how-cross-platform-made-possible-47576976c5c1>

The Engine architecture, [1 août 2019]. *GitHub* [en ligne]. [Consulté le 14 Mars 2020]. Disponible à l'adresse: <https://github.com/flutter/flutter>

Rendering in Flutter, [pas de date]. [en ligne]. [Consulté le 14 Mars 2020]. Disponible à l'adresse: <https://flutter.dev/docs/resources/rendering>

Inside Flutter, [pas de date]. [en ligne]. [Consulté le 14 March 2020]. Disponible à l'adresse: <https://flutter.dev/docs/resources/inside-flutter>

Google Trends, [pas de date]. *Google Trends* [en ligne]. [Consulté le 14 Mars 2020]. Disponible à l'adresse: <https://trends.google.fr/trends/explore?date=2018-12-12%202020-03-14&q=Flutter,React%20Native>

Stack Overflow Trends, [pas de date]. [en ligne]. [Consulté le 15 Mars 2020]. Disponible à l'adresse: <https://insights.stackoverflow.com/trends?tags=flutter%20Creact-native>

Flutter Documentation - Flutter, [pas de date]. [en ligne]. [Consulté le 15 Mars 2020]. Disponible à l'adresse: <https://flutter.dev/docs>

Getting Started · React Native, [pas de date]. [en ligne]. [Consulté le 15 Mars 2020]. Disponible à l'adresse: <https://reactnative.dev/>

Newest “react-native” Questions, [pas de date]. *Stack Overflow* [en ligne]. [Consulté le 15 Mars 2020]. Disponible à l'adresse : <https://stackoverflow.com/questions/tagged/react-native>

React Native community, [pas de date]. *Spectrum* [en ligne]. [Consulté le 15 Mars 2020]. Disponible à l'adresse: <https://spectrum.chat/react-native?tab=posts>

Newest “flutter” Questions, [pas de date]. *Stack Overflow* [en ligne]. [Consulté le 15 Mars 2020]. Disponible à l'adresse: <https://stackoverflow.com/questions/tagged/flutter>

Understanding native app development - what you need to know in 2019, [19 Mars 2019]. *Raygun Blog* [en ligne]. [Consulté le 15 Mars 2020]. Disponible à l'adresse: <https://raygun.com/blog/native-app-development/>

IONIC, [7 Février 2019]. Ionic Article: What is Hybrid App Development? *Ionic Framework* [en ligne]. [Consulté le 15 Mars 2020]. Disponible à l'adresse: <https://ionicframework.com/resources/articles/what-is-hybrid-app-development>