



Noémie PERDEREAU

Étudiante en Informatique

Année 2018-2019

RAPPORT D'APPRENTISSAGE



Maître d'apprentissage :
Goulven KERBELLEC

Enseignant tuteur :
Pascal BAUDONT

Entreprise :

CyberEspar
2, rue du Vieux Four
F-56870 LARMOR BADEN

REMERCIEMENTS

Je tiens tout d'abord et tout particulièrement à remercier Goulven KERBELLEC, mon maître d'apprentissage au sein de l'entreprise CyberEspar. Il a su me donner confiance en moi et en mon travail en me laissant le plus d'autonomie possible tout en étant présent pour m'enseigner et répondre à mes questions.

Je remercie également l'équipe pédagogique de l'IUT pour la mise en place de l'alternance au sein du DUT informatique et plus particulièrement Murielle Mannevy et Pascal Baudont. Ils ont tous deux assuré un suivi régulier et personnalisé afin de s'assurer que tout se passait bien pour nous comme pour nos entreprises. J'ajoute une mention spéciale pour Pascal Baudont qui a également été mon tuteur d'alternance.

SOMMAIRE

<u>I)</u>	PRÉSENTATION DE L'ENTREPRISE	5
<u>II)</u>	PRÉSENTATION DU CLIENT	6
<u>a.</u>	La société d'Hubert Laurent	6
<u>b.</u>	Son besoin	6
<u>III)</u>	MA MISSION	8
<u>a.</u>	En général	8
<u>b.</u>	Mes outils	9
<u>IV)</u>	L'APPLICATION TALKTALKERS	12
<u>a.</u>	Mes débuts	12
<u>b.</u>	L'application	13
<u>V)</u>	GESTION DE PROJET	28
<u>VI)</u>	DIFFICULTÉS RENCONTRÉES ET SOLUTIONS	30
<u>VII)</u>	VISUELS	31
<u>VIII)</u>	AUTRES PROJETS	34
<u>IX)</u>	CONCLUSION	36
<u>X)</u>	RESUMÉ / SUMMARY	37

I) PRÉSENTATION DE L'ENTREPRISE

L'entreprise dans laquelle je travaille se nomme CyberEspar. Le nom CyberEspar s'écrit ainsi car il se compose de deux mots. Le premier (cyber) est le raccourci du mot cybernétique qui signifie selon son origine grecque « art de piloter, de gouverner un bateau » et qui, actuellement fait référence à internet. Le second est tiré du vocabulaire maritime à nouveau et fait référence à tous les éléments dépassant d'un bateau (mât, tangon, bout-dehors, etc.).

Elle a été créée en 2012 et est dirigée par Goulven Kerbellec. Son activité principale est la conception d'applications mobiles. Mais ce n'est pas sa seule activité. Son fondateur étant passionné de voile, l'activité de l'entreprise s'oriente également vers les objets connectés liés à la mer (capteurs de marées, capteurs de vent sur les voiles des bateaux, drones marins etc.). La clientèle de Goulven couvre un large panel allant de particuliers aux professionnels en passant par des associations. Goulven est président d'une SAS mais n'a pas d'employé. Ce qui lui permet de travailler selon les horaires de son choix et depuis le lieu de son choix. A la création de son entreprise, il travaillait depuis chez lui et adaptait ses horaires en fonction de ses enfants.

Ce n'est qu'en mars 2018, lorsque la commune de Larmor-Baden décide de mettre en location un espace de travail partagé que Goulven se décide à rompre ce travail solitaire. C'est avec deux de ses amis auto-entrepreneurs (Virginie Le Goascoz et Anthony Dubernet) qu'ils décident de fonder l'association « Le Catway » et de déposer leur candidature auprès de la mairie. Cette demande acceptée, cela va bientôt faire un an qu'ils travaillent tous les trois dans cette petite maison vue sur mer. Leurs trois métiers étant à la fois différents et complémentaires, ils sont souvent amenés à travailler ensemble sur des projets. Virginie est graphiste et Anthony intégrateur graphique.



Anthony DUBERNET, Goulven KERBELLEC et Virginie LE GOASCOZ

II) PRÉSENTATION DU CLIENT

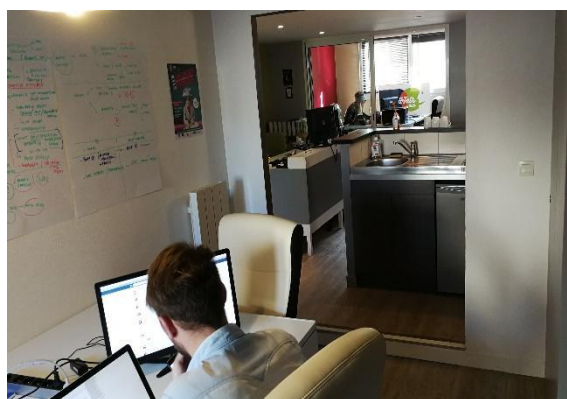
Lors de mon arrivée dans l'entreprise, Goulven venait tout juste de se lancer dans un gros projet. J'ai donc travaillé pour un même client (Hubert Laurent) pendant toute ma première période d'entreprise. Même si actuellement je découvre de nouveaux projets et de nouveaux clients, je vais essentiellement concentrer mon rapport sur ce premier projet.

a. La société d'Hubert Laurent

Hubert Laurent dirige une agence de traduction ainsi qu'un site internet depuis son bureau à Lorient. Il est très pratique pour Goulven et moi que ses bureaux soient à Lorient car nous pouvons aisément nous déplacer afin d'avancer ensemble sur le projet en ayant un retour immédiat. L'équipe d'Hubert se divise en deux parties. Une personne s'occupe de la traduction à temps plein afin de lui libérer du temps pour s'occuper de son site internet. De ce côté-ci, l'équipe se renouvelle régulièrement avec des stagiaires aux idées neuves et travaillant essentiellement sur la communication et la mercatique.



Hubert Laurent



Ses locaux

b. Son besoin

En mars 2016, Hubert se lance dans la création d'un site internet (TalkTalkBnb) mettant en relation des personnes souhaitant pratiquer les langues étrangères avec des voyageurs étrangers souhaitant être hébergés gratuitement, chez des locaux. Son site connaît un franc succès et il se constitue rapidement une solide communauté. Ne parvenant pas à trouver de modèle économique et souhaitant moderniser son activité, il prend contact avec Goulven pour transformer son site en une application mobile. Le principe n'est cependant pas tout à fait le même que sur son site. L'application permet à une personne de proposer des offres de types variés (aller boire un verre, faire une balade à vélo, un atelier de maquillage, etc.) dans sa langue maternelle.

Ainsi, un voyageur en se rendant sur l'application voit apparaître une série d'activités à proximité (grâce à la géolocalisation) lui proposant de pratiquer la langue de son choix tout en s'amusant. Prenons un exemple : je suis en vacances à Londres et je souhaite pratiquer mon anglais. Ce n'est pas toujours facile d'aborder les gens dans la rue. Je me rends donc sur l'application de mon client. On me propose alors différentes offres : une visite d'un quartier typique de la ville pour 10€, un atelier de pâtisserie typique pour 15€ ou encore une immersion de 2 nuits dans une famille pour 30€.

Notre but en tant que développeurs, est de créer cette application en conservant la charte graphique du site internet pour maintenir le plus de membres possibles.



III) MA MISSION

a. En général

Goulven et moi devons créer cette application en partant de zéro.

Le processus de création est simple. L'équipe d'Hubert nous fournit des plaquettes détaillées de ce qu'ils veulent voir sur l'application. On y trouve des textes expliquant le cheminement d'un utilisateur test, des exemples tirés de diverses plateformes (par exemple la page de modification de son profil sur twitter correspond à ce qu'Hubert souhaite mettre en place) ainsi que des maquettes qu'ils dessinent eux même.

Grâce aux textes fournis, Goulven est capable de créer une application fonctionnelle sans s'occuper de l'esthétique. Viens ensuite mon rôle. Je dois créer une page reproduisant du mieux possible les modèles que le client m'a envoyés tout en reliant les fonctionnalités que Goulven a mises en place.

Le modèle dessiné par
l'équipe d'Hubert

Mon point de départ une
fois que Goulven a
implémenté les
fonctionnalités

Ce que j'ai réalisé dans un
premier temps



b. Mes outils

Avant de vous présenter mes outils, nous allons prendre un peu de hauteur sur ce qui se fait en matière de développement d'application. Il existe deux types d'applications mobiles : les applications natives et les applications multi-plateformes.

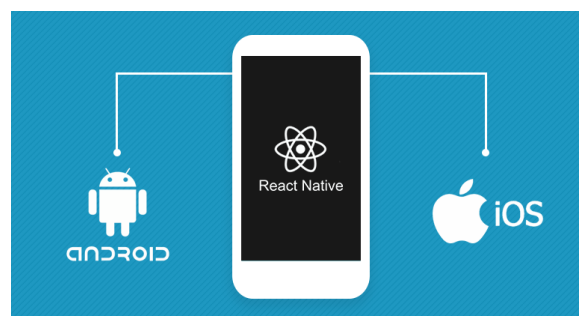
Les applications **natives** sont spécifiquement développées pour un système d'exploitation. Actuellement, les systèmes d'exploitation les plus connus sont iOS et Android. En choisissant de créer une application native, il faudra, en réalité, en développer deux (une pour iOS et une autre pour Android). Cela représente donc un gros inconvénient car la création de deux applications prend deux fois plus de temps et surtout nécessite que le développeur maîtrise les deux langages. Si ce n'est pas le cas, il faudra embaucher deux personnes différentes et donc s'attendre à ce que les deux codes sources ne suivent pas forcément la même logique de création.

C'est pourquoi les applications **multi-plateformes** ont été créées. A l'inverse des natives, le développement se fait en une seule fois (et dans le même langage) et l'application est compatible sous iOS et Android. Parfait me diriez-vous ! En réalité, pas tout à fait. Il y a tout de même un inconvénient majeur. Ces applications se sont montrées beaucoup moins fluides et performantes, et impliquent de nombreux ajustements pour passer d'une plateforme à l'autre.

En 2015, Facebook fournit une solution libre : le **React Native**. React Native permet de créer des applications multi-plateformes donc un seul code (en JavaScript) mais en utilisant des composants graphiques natifs. Par exemple, le composant permettant de définir une vue sera un mot clé de React Native dans le code (View) et sera interprété selon la plateforme comme un composant natif Android (android.view.View) ou un composant natif iOS (UIView). Cette façon de faire permet de conserver la performance et la fluidité d'une application native tout en écrivant qu'un seul code comme pour une application multi-plateforme.

S'ajoute à ces avantages le fait que React Native soit gratuit, créé par Facebook donc cela donne une bonne image pour la clientèle et testable instantanément.

En effet, tout le long du processus de création, le rendu visuel s'actualise en fonction de mon avancée sur un simulateur. Lors de gros changements ou avant les rencontres clients, nous lui fournissons un lien lui permettant d'installer la dernière version de l'application sur son téléphone portable.



La création d'une application mobile nécessite un **environnement de travail particulier**. Une fois que nous avons choisi le langage utilisé, il nous faut un éditeur de texte. Je travaille avec **Atom**. Il n'y a pas de raison particulière à ce choix si ce n'est que les visuels sont agréables (des couleurs rendent le code lisible) et le thème sombre permet de moins solliciter les yeux.

```
import styles from './Styles/ActiviteScreenStyle'

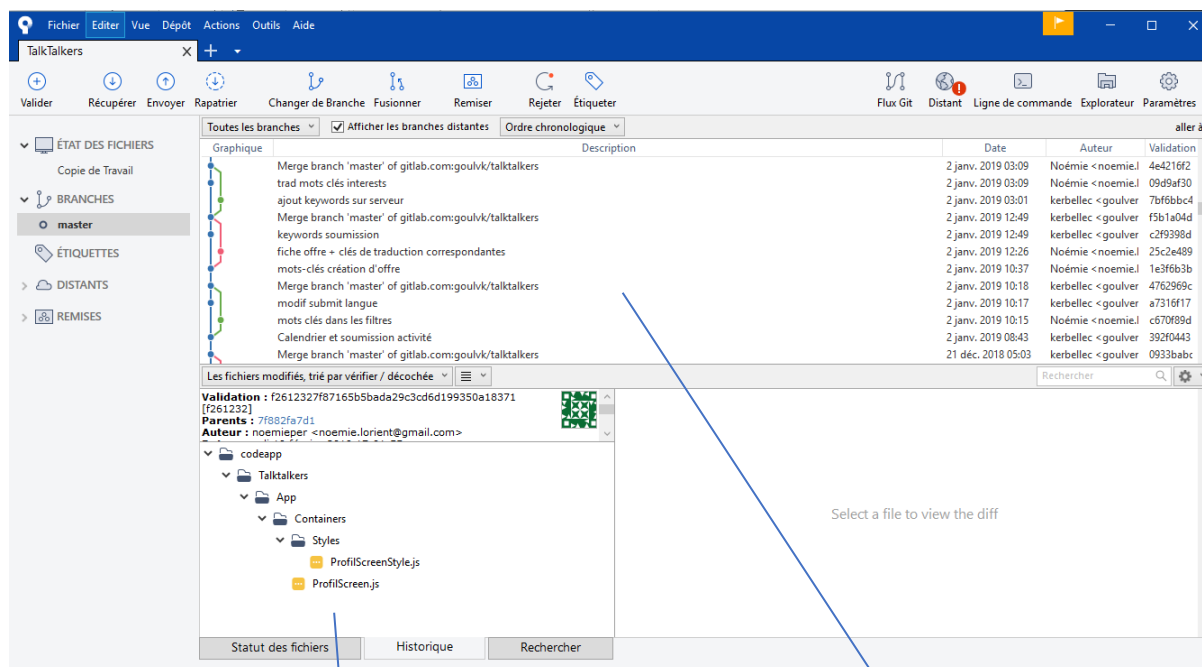
class ActiviteScreen extends Component {

  handleFieldParamsChange = (valid, params) => {
    console.log(`
      Valid: ${valid}
      Number: ${params.number || '-'}
      Month: ${params.expMonth || '-'}
      Year: ${params.expYear || '-'}
      CVC: ${params.cvc || '-'}
    `)
  }
}
```

Afin de compiler le code, il faut installer un environnement de développement. Pour la partie Android, nous travaillons avec Android Studio et pour la partie iOS avec XCode. Ces logiciels permettent d'éditer du code et de visualiser instantanément le résultat grâce à des émulateurs (smartphone fictif).

Dans la majorité des projets et afin de limiter le risque de perte, il est nécessaire de sauvegarder le code en ligne. La plateforme Git permet de garder en mémoire le code tout en traitant l'assemblage des différentes versions quand nous travaillons à plusieurs. En effet, à chaque modification importante, nous actualisons la version en ligne de notre code pour que les autres membres de l'équipe puissent avancer sur le projet le plus abouti à chaque fois. Git s'arrange pour fusionner les codes si deux personnes travaillent en parallèle sur des fichiers différents. Cependant, si deux personnes travaillent sur le même fichier, il faudra choisir manuellement quelle version conserver.

Git peut s'utiliser en ligne de commande mais pour simplifier la compréhension, nous utilisons l'interface SourceTree qui permet de mieux visualiser l'état du code.



Les fichiers que j'ai modifiés

Le suivi des modifications (avec possibilité de récupérer une ancienne version si la nouvelle ne fonctionne plus ou dans le cas où le client voudrait un retour en arrière)

D'un point de vue matériel, je travaille toute la journée sur un ordinateur. Je dispose d'un Mac et de mon propre PC afin de tester les rendus sur les deux plateformes.

IV) L'APPLICATION TALKTALKERS

a. Mes débuts

La première semaine de mon apprentissage a été consacrée à la découverte des outils, du langage ainsi que du projet et des techniques de travail. J'ai installé sur mon ordinateur divers logiciels et après de nombreuses heures d'attentes j'ai enfin pu créer une application mobile vierge depuis mon propre ordinateur.

Ma première mission était de réaliser une application mobile simple en me débrouillant par moi-même. J'ai parcouru le web et les tutoriels en ligne et j'ai réussi rapidement à me familiariser avec le langage. En fin de première semaine, je terminais une petite application d'histoire à choix multiples.

Afin de me rapprocher davantage du projet dans l'entreprise, Goulven m'a demandé par la suite de reproduire graphiquement des maquettes de pages envoyées par le client. Cette tâche m'a paru difficile au début car les maquettes ne sont pas toujours faisables. J'ai dû effectuer un gros travail de recherche pour trouver des composants adéquats.

Une fois prête, j'ai pu installer le projet sur mon ordinateur et travailler dessus. J'ai commencé par reproduire simplement les maquettes graphiques sous IOS. Lors des premières semaines je me débrouillais pour faire en sorte que cela soit ressemblant sans me soucier, ni de l'adaptabilité aux différentes tailles d'écran ni de la compatibilité avec Android. Ce n'est que lors de la première journée de travail chez le client en voyant le résultat sur son téléphone que j'ai pris conscience de l'importance d'un travail propre et adaptable.

J'ai alors passé les semaines suivantes à reprendre mon travail en utilisant des techniques plus adaptables. Par exemple pour ajuster la taille d'un bouton il vaut mieux utiliser des pourcentages qu'une taille fixe. Peu à peu, j'ai essayé d'implémenter des petites fonctionnalités comme des réactions au clic sur un bouton. Goulven m'a toujours laissé découvrir par moi-même et une fois la fonctionnalité créée il me montre d'autres façons de faire.

b. L'application

LES FICHIERS

L'application est en réalité un dossier contenant de nombreux dossiers et fichiers. Parmi eux, on retiendra le plus important.

Le dossier **App** contient l'intégralité des pages. L'application se décompose en plusieurs écrans principaux (l'écran de connexion, l'écran contenant la liste des activités, l'écran du profil etc) ainsi qu'un système de navigation entre les écrans symbolisé par un menu. Lors de l'affichage, les écrans s'empilent les uns au-dessus des autres. Une page de code contient deux parties qui communiquent entre elles.

La première partie s'apparente au modèle (traitement des données via des fonctions) et la seconde partie à la vue (affichage des composants). Le modèle se présente sous la même forme que du JavaScript. Une page correspond à une classe qui contient des attributs (pour la page de connexion : identifiant, mail, mot de passe ...), des fonctions (connexion via facebook via google, vérifier le mot de passe, ...) et enfin un affichage. L'affichage s'apparente à du HTML. Il se décompose en balises. Le style des éléments se trouve dans un fichier séparé codé en CSS.

Constructeur avec les attributs stockés dans l'objet « this.state » et fonctions

```
import firebase from 'react-native-firebase';
import { GoogleSignin } from 'react-native-google-signin';
import { AccessToken, LoginManager } from 'react-native-fbsdk';
import styles from './Styles/RegisterStyle'

class Register extends Component {
  constructor(props){
    super(props);
    this.state={};
  };

  googleLogin = async () => {}
  facebookLogin = async () => {}
  isEmailValid = () => {}
  isPassword1Valid = () => {}
  isPassword2Valid = () => {}
  createProfil = () => {}
  doRegister = () => {}
  doConnection = () => {}

  render () {
    return (
      <View>
        <Spinner
          visible={this.state.spinner}
          overlayColor={"rgba(0,0,0,0.6)"}
        />
      </View>
    )
  }
}
```

Importation des différents fichiers dont on a besoin (bibliothèques, feuilles de styles)

Affichage de la vue sous forme de balises

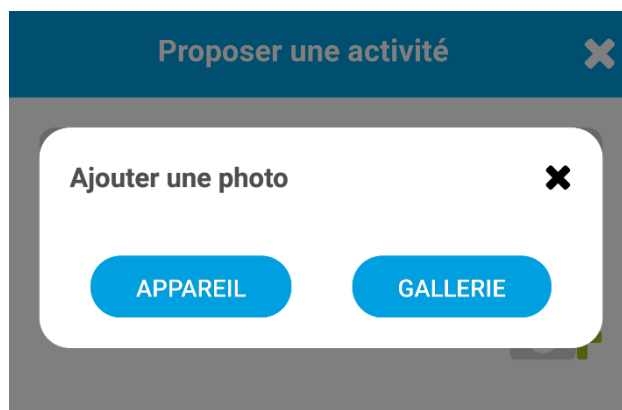
Une balise peut correspondre à plusieurs choses. Il peut s'agir d'un composant du langage lui-même comme par exemple les balises View et Text. Elles nécessitent la mention de type « `import XXX from react-native` » avant de pouvoir l'utiliser. Il peut s'agir de libraires que nous avons ajoutés nous-même ou encore de composants que nous créons.

Lorsqu'un composant va être amené à être utilisé sur plusieurs pages ; ce serait une perte de temps de l'écrire dans chaque fichier. Par exemple, le composant concernant l'ajout de photo.

Image 1 :

```
<PicsModal
  visible={this.state.modalPicsVisible}
  handlermodpics={this.handlermodpics}
  handlerPics={this.handlerPics}
/>
```

Image 2 :



Ce composant est appelé dans la page du profil (pour changer sa photo de profil et sa photo de bannière) mais aussi dans la création d'une activité (pour ajouter des photos liées à l'activité). Au lieu de le copier trois fois, nous l'écrivons dans un fichier à part. Cela permet de l'appeler de n'importe quelle page en l'utilisant comme une balise standard. Il peut également s'adapter aux différents contextes grâce aux paramètres que nous passons lors de son appel. En effet, nous lui passons en paramètre des attributs ou des fonctions spécifiques.

Par exemple sur l'image 1, l'attribut visible est un booléen. Si sa valeur est à true, le modal sera visible et si elle est à false, il ne sera pas visible. La fonction handlermodpics permet de fermer le modal en modifiant la valeur de l'attribut visible. Et enfin, la fonction handlerPics permet le stockage de l'image choisie par l'utilisateur.

L'AJOUT DE LIBRAIRIES

Le langage React Native permet d'importer des librairies. Une librairie correspond à un morceau de code déjà créé que l'on peut utiliser en tant qu'objet. Par exemple, il est possible de faire soi-même son menu de navigation mais cela prend beaucoup de temps. C'est pourquoi il existe des objets de type « menu de navigation ».








Pour l'utiliser, il faut installer la librairie dans notre dossier de travail. Dans le meilleur des cas, la commande `npm install -save <nom du module>` suffit à enregistrer les dépendances dans le fichier `package.json` et la commande `react native link` pour installer les composants dans le dossier `node-module`. Les autres utilisateurs, grâce à l'option « save » n'auront qu'à exécuter la commande `npm install` qui se chargera de remplir à nouveau le dossier « node-module » avec les nouvelles librairies.

Voici un extrait du fichier `package.json` (image 1) qui contient les versions à télécharger lors de la commande `npm install` pour chaque librairie (ici, le diaporama au lancement de l'application, le calendrier de réservation etc). Dans le dossier `node-module` (image 2) on retrouve les dossiers correspondants aux librairies et à l'intérieur de ces dossiers on retrouve entre autres les fichiers correspondant au code du composant.

Image 1 :

```
"react-native-app-intro-slider": "^1.0.1",
"react-native-calendars": "^1.22.0",
"react-native-circular-action-menu": "^0.5.0",
"react-native-config": "^0.10.0",
"react-native-datepicker": "^1.7.2",
```

Image 2 :

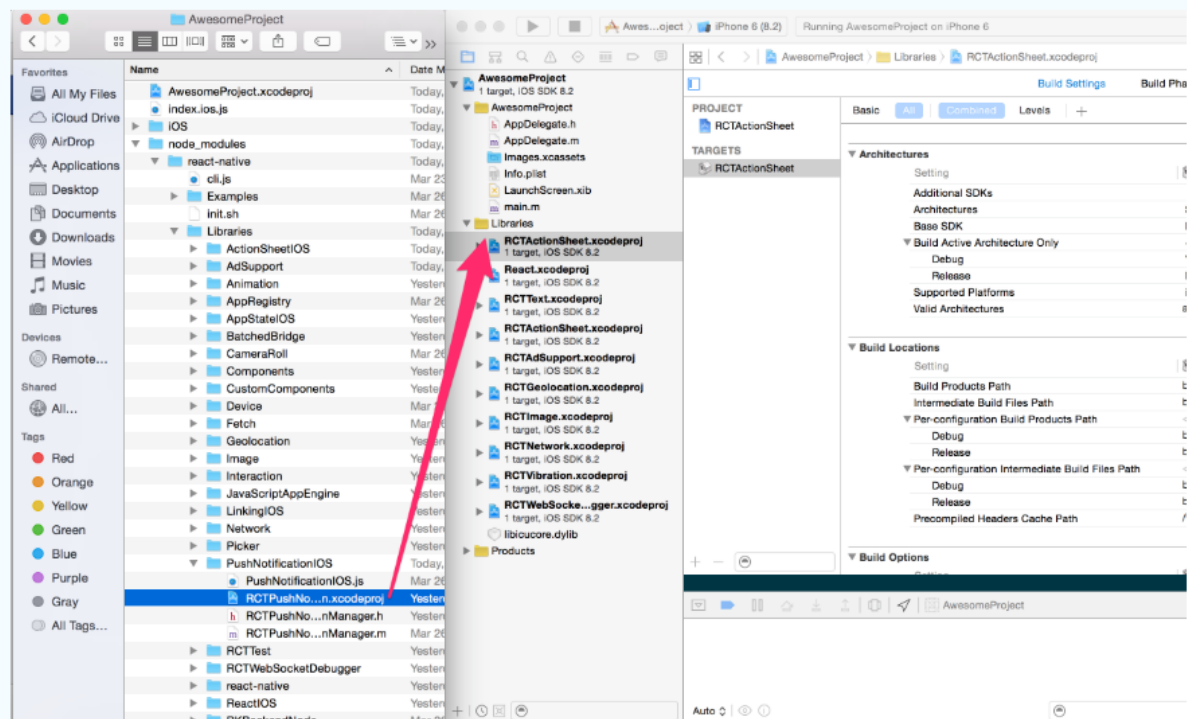
	react-native-app-intro-slider	12/04/2019 12:00	Dossier de fichiers
	react-native-button	12/04/2019 12:00	Dossier de fichiers
	react-native-calendars	12/04/2019 12:00	Dossier de fichiers
	react-native-circular-action-menu	12/04/2019 12:00	Dossier de fichiers
	react-native-compat	12/04/2019 12:00	Dossier de fichiers
	react-native-config	12/04/2019 12:00	Dossier de fichiers
	react-native-datepicker	12/04/2019 12:00	Dossier de fichiers

Néanmoins, cela ne fonctionne pas toujours aussi facilement et certaines librairies ne s'installent pas ou ne s'enregistrent pas d'un ordinateur à l'autre et il faut le faire manuellement. Heureusement, l'importation de librairie dispose de documentation complète avec les explications pour réaliser le lien manuel.

Manual linking

Step 1

If the library has native code, there must be an `.xcodproj` file inside its folder. Drag this file to your project on Xcode (usually under the `Libraries` group on Xcode);



LES LANGUES

L'application ayant une cible internationale, il est important de fournir un livrable en plusieurs langues. Nous utilisons pour cela l'internationalisation logicielle (ou i18n – 18 représentant le nombre de lettres entre le i et le n du mot internationalisation). Le but est de produire une application qui puisse être immédiatement déployé dans différentes langues en ajoutant simplement un nouveau fichier de traduction par langue supportée.

Il s'agit dans un premier temps de savoir quelle langue afficher. La librairie i18n se base sur les valeurs du téléphone afin d'en déduire la langue de l'utilisateur.

```
let languageCode = I18n.locale.substr(0, 2)
```

Une fois le langage récupéré, on lui attribue un fichier de type Json. Chaque langue à son propre fichier. Ce fichier contient des couples mot-clé/traduction.

```
case 'es':
  I18n.translations.es = require('./languages/es.json')
  break
```

```
"profil": "Profil",
"offers": "Offres",
"reservations": "Réservations",
"fav": "Favoris",
"messages": "Messagerie",
"param": "Paramètres",
"about": "À propos",
"help": "Aide",
"contact": "Contact",
"logOut": "Déconnexion",
```

Chaque clé est identique pour toutes les langues et est associée à sa traduction. En fonction du fichier que l'application appelle au lancement le texte qui va s'afficher sera dans la bonne langue. Le code de l'application ne contient pas de texte. Chaque phrase ou mot est représenté par cette unique clé qui se retrouve dans toutes les langues. Elle affichera le texte correspondant à la langue choisie à l'initialisation.

Une fois les clés créées, il a fallu remplir un fichier excel. Ce fichier contient la clé de traduction, le contexte et la traduction française écrite par le client. Il sera ensuite traduit dans toutes les langues souhaitées.

Par la suite, nous allons réaliser un script capable de lire le fichier excel et de remplir les fichiers Json correspondants aux différentes langues.

```
{ I18n.t('messages') }
```

Clé de traduction insérée dans le code source (dans une balise de texte par exemple).

Clé	Contexte	Français (fr-FR)	Anglais (en-US)
app_name	Nom de l'appliation sur le store (50)	TalkTalkers	TalkTalkers
store_description	Description brève sur le store (80 caractères)	TalkTalkers est une application collaborative sur la thématique des langues	TalkTalkers is a collaborative app all about languages.
store_presentation	Le texte de présentation pour les stores, description complète (4000 caractères)	Elle permet de rencontrer des locaux lors de vos voyages, pour converser avec eux et progresser en langue.	It helps you meet local people while abroad, for conversation and to improve your language skills.
LOGIN			
slider_text_1	Les accroches qui expliquent le concept en quelques mots	En voyage, rencontrez des locaux, pratiquez les langues et découvrez leur cultures.	Meet local people, practice your languages and discover new cultures while abroad.
slider_text_2		Inscrivez-vous à des activités dans la langue de votre choix.	Register for activities in the language of your choice.
slider_text_3		Séjournerez chez l'habitant pour une parfaite immersion linguistique.	
slider_text_4		Vous aussi, devenez hôte ou proposez une activité et gagnez de l'argent !	Why not become a host yourself or offer an activity and earn some extra money?
register	boutons sur le slider (aussi sur les modaux génériques)	Inscription	Sign up
signIn		Connexion	Log in
mail		Email	Email
password		Mot de passe	Password
passwordConfirmation		Confirmez votre mot de passe	
forgotPassword		Mot de passe oublié ?	Forgot your password?

Fichier excel regroupant toutes les clés et toutes les traductions

TRAITEMENT DES DONNÉES

Actuellement, nous n'avons pas encore de données réelles. Nous travaillons avec de faux profils. On constate cependant que l'application demande un stockage important des données avec les profils des utilisateurs, les fiches des activités, les messages échangés etc.

On peut distinguer trois types de données. Les données internes correspondent à des éléments stockés temporairement par exemple l'état d'un bouton ; les données externes correspondent aux informations liées aux personnes et aux activités. Elles doivent être accessible de n'importe quel fichier ; et enfin les informations confidentielles telles que les coordonnées bancaires.

Les **données internes** sont stockées dans un objet « **state** » contenant les variables actuelles de la page. Lors de l'écriture de notre page, on peut rajouter et modifier des éléments dans le state grâce à la méthode `setState`. Même si on change de page, les valeurs du state seront conservées. Cependant, elles ne sont accessibles que depuis la page à laquelle elles appartiennent.

```
this.state = {  
  visible: false,  
  datamembre: dataInit.data,  
  modalVisible: false,  
  modal2Visible: false,  
  modal3Visible: false,  
  spinner: false,  
}
```

```
this.setState({modalVisible: false});
```

Pour faire transiter des données d'une page à une autre, on utilise l'objet « **props** » (abréviation de propriété). Lors de la navigation vers une seconde page via la fonction `navigate`, on lui ajoute des paramètres.

```
this.props.navigation.navigate({  
  routeName: route,  
  params: {action:ac}  
})
```

Ses paramètres peuvent être utilisés par la seconde page (`this.props.nomVariable`). Contrairement au `state`, les variables contenues dans le `props` ne peuvent pas être modifiées via une fonction de type « `setProps` ». Pour contourner ses règles, il est possible d'ajouter des fonctions au `props`. Ces fonctions peuvent modifier des variables.

Lorsqu'une application ne possède pas beaucoup de pages, il est aisé de transporter les valeurs de l'une à l'autre en paramètre. Dans notre cas, cela devenait trop compliqué car certaines valeurs comme les variables du profil de la personne doivent être accessibles depuis toutes les pages de l'application (environ une vingtaine de pages). Nous avons alors décidé d'utiliser **Redux**.

Redux est une bibliothèque qui stocke des variables et des fonctions. La bibliothèque est organisée en rayons. Chaque rayon contient une série de variables et de fonctions qui lui sont propres. Par exemple, le rayon « profil » contient les variables « `username`, `avatar`, `firstName` » et des fonctions de connections.

```
export const reducers = combineReducers({
  nav: require('./NavigationRedux').reducer,
  github: require('./GithubRedux').reducer,
  search: require('./SearchRedux').reducer,
  profil: require('./LoginRedux').reducer,
  activites: require('./ActivitesRedux').reducer
})
```

Liste de différents « rayons » (fichier index de Redux)

```
export const INITIAL_STATE = Immutable({
  speakerProfil: {avatar: "", first_name: ""},
  username: null,
  error: null,
  fetching: true,
  logged: false,
  authed: false
})

export const request = (state) => {
  return state.merge({ fetching: true, speakerProfil: null })
}

export const logoutRequest = (state) => {
  console.log("danssss le logoutRequest")
  return state.merge({ authed: false, logged: false, fetching: false, speakerProfil: null })
}
```

Déclaration des variables et ensuite des fonctions (fichier correspondant au rayon Profil de Redux)

Afin d'utiliser ses variables de partout, il faut importer les rayons souhaités. En haut de chaque fichier, il faut importer la bibliothèque Redux. Puis il faut préciser quel rayon on souhaite utiliser dans une fonction nommée `mapStateToProps`. Et enfin dans la fonction `mapDispatchToProps`, il faut déclarer les fonctions que l'on souhaite utiliser.

```
import { connect } from 'react-redux'
```

Connexion à la bibliothèque Redux

```
const mapStateToProps = state => {
  return {
    profil: state.profil,
  };
};

const mapDispatchToProps = dispatch => {
  return {
    getProfil: () =>
      dispatch(ProfilActions.getProfil()),
    getProfilFirebase: (email,password) =>
      dispatch(ProfilActions.getProfilFirebase(
        {email:email, password:password}
      )),
    createProfilFirebase: (uid,profil) =>
      dispatch(ProfilActions.createProfilFirebase(
        {uid:uid, profil:profil}
      )),
  };
};
```

Fonction `mapStateToProps` qui récupère le rayon Profil ; et la fonction `mapDispatchToProps` qui déclare les fonctions.

Chaque modification d'une valeur entraîne la création d'un événement. Par exemple si un bouton implémente une fonction stockée dans Redux (fonction se connecter par exemple), en cliquant sur le bouton, on va déclencher un événement dans la librairie qui va pouvoir entre autres mettre à jour ses valeurs.

Un nouveau problème entre en jeu. Que faire si l'utilisateur clique plusieurs fois sur le bouton ? Pour gérer ce problème, nous utilisons **Saga**.

Saga est une librairie qui s'occupe de l'interface utilisateur. Elle agit avant l'appel à la bibliothèque Redux. Lorsqu'un événement Redux est créé, il est stocké sous forme d'une pile dans la librairie Saga. Selon la configuration de Saga, les données seront traitées et envoyées à Redux. Par exemple, on peut décider de traiter tous les clics un par un ou bien seulement le dernier.

```
takeLatest(StartupTypes.STARTUP, startup),
takeLatest(ProfilTypes.GET_PROFIL, getProfil, api),
takeLatest(ProfilTypes.GET_PROFIL_FIREBASE, getProfilFirebase),
```

Ici, nous demandons uniquement le traitement de la dernière action

```
export function * sendMessage (action) {
  const {message}=action;
  console.log("send to fb",message)
  const reduxSagaFirebase = new ReduxSagaFirebase(firebase);
  yield call (reduxSagaFirebase.firestore.addDocument, "messages",message)
}
```

Les fonctions de Saga appelle ensuite celles de Redux

Les **données externes** peuvent être stockées sur un serveur personnel. C'était le cas au début du projet. Redux se chargeait de récupérer et d'ajouter des données sur le serveur. Puis, nous avons découvert la base de données Firestore via le service Firebase. Firebase est un ensemble de services d'hébergement proposé par Google. Il propose un hébergement en temps réel des bases de données, de l'authentification, des notifications et enfin de la gestion des erreurs. Nous avons choisi de basculer la base de données essentiellement pour la gestion des notifications dites « push ». Voici un schéma de l'ensemble des fonctionnalités de Firebase.



FIREBASE

OUTILS DE DEVELOPPEMENT :

- Authentification
- Stockage
- Hébergement
- Tests
- Base de données en temps réel
- Fonctions supplémentaires
- Gestion des bogues



OUTILS POUR ALLER PLUS LOIN :

- Analyse des données
- Gestion des notifications
- Indexation
- Configuration personnelle de la console Firebase

Un autre avantage considérable de la base de données Firestore est qu'elle est écrite en noSQL contrairement à celle du serveur qui est en postgresSQL. Le noSQL est une façon de créer une base de données beaucoup moins compartimenté. Une même table peut stocker beaucoup d'informations même si elles n'ont pas toujours de lien. Contrairement au SQL qui permet de stocker les valeurs d'une seule façon (clé/valeur), le noSQL permet différentes méthodes de stockage (clé/valeur, orienté colonne ou encore orienté graphe). Ce stockage plus libre permet une utilisation simplifiée de l'analyse des données (bigData).

De plus, Firebase dispose d'une interface graphique complète et simple sur laquelle on peut gérer visuellement les entrées dans la base de données.

Données concernant les utilisateurs (connexion classique ou via facebook/google, mot de passe, date de création) :

noemie.lorient@gmail.com		8 avr. 2019	7 mai 2019	4WVw0r1eKHZqUL7luHWMvDyxhj...
izcephiivb_1552889856@tfb...		18 mars 2019	18 mars 2019	AHbLzCtYWJNMQG40qaN0SLpt0l...

Données concernant les activités ainsi que les profils et les messages :

The screenshot shows a web application interface with three main panels. The left panel, titled 'talktalkers', contains a sidebar with 'Ajouter une collection' and a list of collections: 'activites', 'membres' (selected), and 'messages'. The middle panel, titled 'membres', shows a list of user IDs, with '4WVw0r1eKHZqUL71uHWMvDyxhj13' selected. The right panel, titled '4WVw0r1eKHZqUL71uHWMvDyxhj13', displays the user's profile information and a collection of keywords.

User Profile Information:

- adresse: "56100 Lorient, France"
- age: null
- apropos: "Étudiante en 2ème année de DUT informatique à Vannes."
- avatar: "http://cyberespar1.2sia.be:3000/uploads/avatars/responsive/918c92dC"
- banniere: "http://cyberespar1.2sia.be:3000/uploads/avatars/responsive/135dbc"
- date: "2000-01-30"
- first_name: "Noémie"
- genre: 1

Keywords:

- 0 "sports"
- 1 "food"
- 2 "music"
- 3 "meditation"
- 4 "nature"

Gestion des notifications :

<div>Create experiment</div> <div>Nouvelle notification</div>						
Notification	État ⓘ	Plate-forme	Début/Envoi	Terminer	Envois	Ouvertures
➤ Remplissez votre profil	✓ Terminée		8 avr. 2019 18:00	—	< 1 000	0 %
➤ C'est important	✓ Terminée		8 avr. 2019 17:57	—	< 1 000	0 %
➤ test libellé test talk notif	✓ Terminée	iOS	27 févr. 2019 12:10	—	< 1 000	0 %

Nous devons aussi gérer des **données confidentielles** de type coordonnées bancaires. Qui dit paiement dit banque. Afin de procéder au paiement depuis l'application, nous utilisons la suite **Stripe** que nous implémentons grâce à la librairie tipsi-stripe.

Lorsqu'une personne veut payer pour une activité, elle renseigne ses coordonnées bancaires dans les champs correspondants.

Paiement

Coordonnées bancaires :

Numéro de carte
4242 4444 4444 4244

Date d'expiration
Mois (XX) Année (XXXX)

Cryptogramme ⓘ

Valider

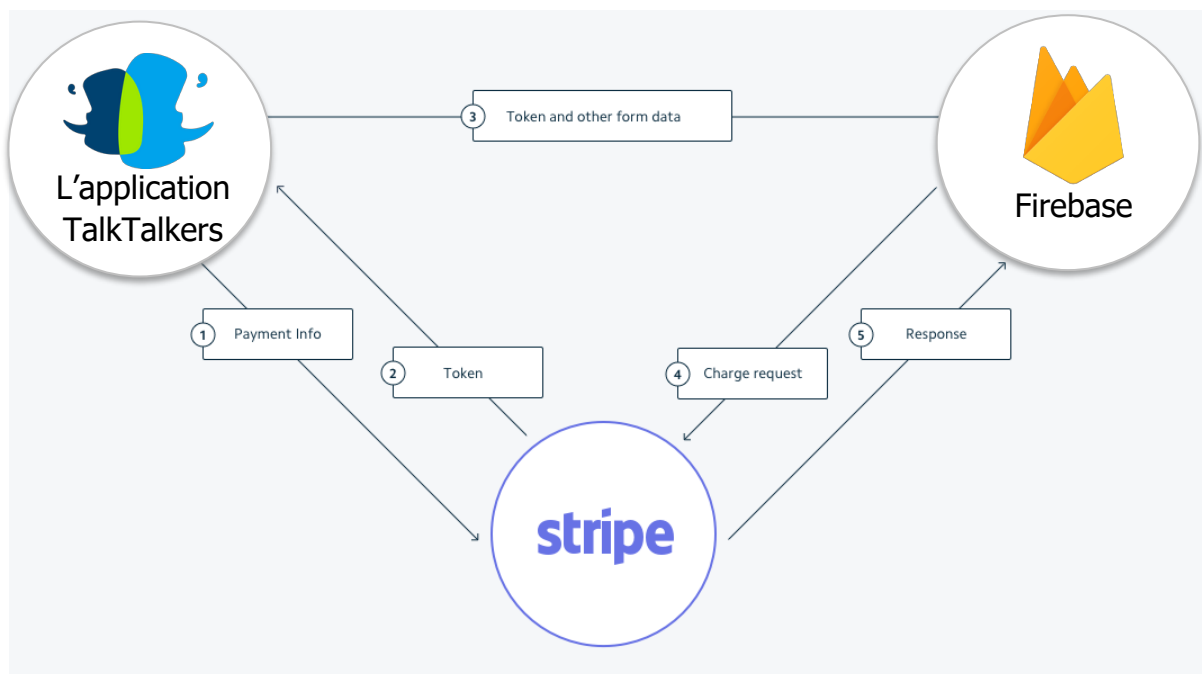
Une fois renseignés, nous ne pouvons pas les conserver. Ce sont des données confidentielles. Elles sont donc directement envoyées à Stripe. En échange, ce dernier nous renvoie ce qu'on appelle un « token ». Il s'agit d'une clé se présentant sous la forme d'une suite de chiffres et de lettres. Cette clé contient l'identifiant de la carte bancaire, le montant et la date. Une fois la clé reçue, nous pouvons la stocker.

Mais ce n'est pas aussi simple que ça. Les échanges permettant de valider le paiement doivent se faire via un serveur. Premièrement pour éviter que les données se promènent dans l'application et puisse être trouvées par une tierce personne. Et deuxièmement pour que la transaction soit toujours conservée même si Stripe ou l'application plante.

Contrairement à Paypal qui gère tout lui-même, Stripe ne met pas de serveur à disposition. On doit donc créer un serveur personnel. Notre application dispose d'une clé publique disponible dans le code, et une clé privée sur le serveur.

Une fois le token reçu, nous l'envoyons avec le montant sur notre serveur qui va se charger de discuter avec Stripe. Grâce à nos clés (public et privée), Stripe va pouvoir nous identifier et confirmer ou infirmer le paiement.

Depuis que nous utilisons Firebase, nous n'avons plus besoin de serveur personnel. Firebase se charge d'interroger Stripe grâce à un script que Goulven a écrit.



PERSISTANCE DES DONNÉES

Lorsque nous fermons l'application, il ne faut pas que les données arrêtent de transiter. Par exemple, tant que ma boîte mail n'est pas ouverte je ne reçois aucun mail et lorsque je l'ouvre les données arrivent. Cela ne sera pas intéressant du tout dans le cas d'une boîte mail. Dans le cas de l'application non plus. Que ce soit pour les notifications ou pour la messagerie, il faut que les données continuent de transiter même lorsque notre application est fermée. Cela s'appelle la persistance des données.


Firebase permet deux types de persistance. La persistance de la base de données et la persistance du cache. Pour ce qui est de la base de données, chaque modification de cette dernière envoie un événement à l'application afin de la mettre à jour. Quant au cache, grâce à Redux, le contenu de chaque bibliothèque reste enregistré. Par exemple, l'application gardera en mémoire que vous vous êtes connecté et ne vous demandera pas systématiquement vos identifiants.

V) GESTION DE PROJET

L'application TalkTalkers est un projet à grande échelle. En effet, il ne s'agit pas de fournir une base simple qui sera développée au fur et à mesure de l'utilisation. Le produit à fournir doit être complet et complexe. C'est pourquoi le planning s'étend sur de nombreux mois.

La première phase consiste à la mise en place de l'application et la seconde phase qui n'a pas encore débutée consiste à la création d'une plateforme permettant au client de valider/modifier ou supprimer les activités et profils et offrant également la possibilité d'exploiter les données pour permettre une meilleure expérience utilisateur.

Depuis septembre, je travaille sur la première phase de création de l'application. Chaque mois nous décidons avec Hubert Laurent des fonctionnalités à implémenter ainsi que des améliorations graphiques à apporter. Suite à cela, Goulven et moi rédigeons des notes sur GitLab regroupant les fonctionnalités, la personne qui va s'en charger ainsi que le temps estimé pour chacune.

logo région #134 · opened 1 week ago by kerbellec	 0 updated 1 week ago
Profil - Ajout de langue #130 · opened 2 weeks ago by Noémie	 0 updated 2 weeks ago
vérif langues non-identiques #129 · opened 2 weeks ago by kerbellec	 0 updated 2 weeks ago
Profil - Bouton Annuler l'ajout de langue ne fonctionne pas #125 · opened 2 weeks ago by Noémie	 0 updated 2 weeks ago

Lors de mes semaines de travail, je suis quotidiennement en contact avec le client. Nous discutons par mail et je lui envoie au fur et à mesure mes avancées mes idées sous forme de modèle réalisé avec l'outil Canva ou des captures d'écran de mon simulateur pour valider petit à petit le design. Evidemment, il nous arrive de revenir sur certaines choses qui ont été validées mais dans l'ensemble cela permet de bien coller à ses attentes.

En fin de semaine ou lors de grosses implémentations, nous réalisons un **fichier au format apk**. Il s'agit d'un fichier téléchargeable permettant l'installation d'une application mobile sur un téléphone Android.

Durant les 6 premiers mois (septembre à février environ), l'apk produite était envoyée par mail au client et téléchargeait l'application sur son téléphone sans passer par le magasin Google

Play. Maintenant que le projet est plus avancé, nous avons fait les démarches afin de pouvoir déposer l'application sur Google Play.

L'application est accessible depuis le magasin mais seulement pour certains comptes google que nous avons renseignés. Elle n'est donc pas téléchargeable de tous. Ce processus simplifie les démarches pour le client car nos améliorations se présentent désormais sous la forme de mises à jour (pouvant être automatiques). Pour ce qui est d'Ios, la procédure pour déposer une application sur l'Apple Store est un peu plus complexe et impose des contraintes de sécurités strictes donc nous ne nous en sommes pas occupés pour le moment.

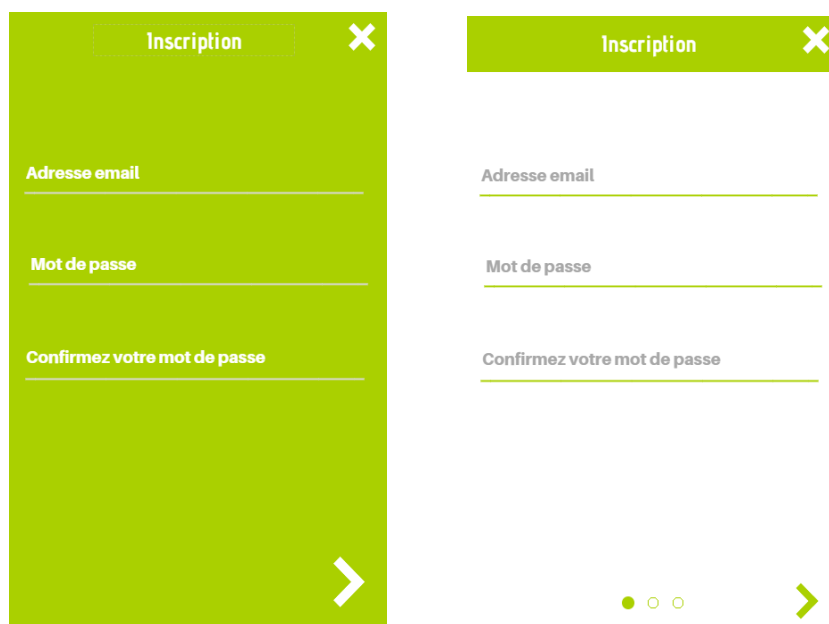
Une fois par semaine (ou une semaine sur deux lorsque nous le jugeons nécessaire), nous passons une journée avec Hubert Laurent. Lors de cette journée, nous **passons en revue toute l'application**. Ce procédé vise deux objectifs. Dans un premier temps, montrer au client l'avancée de son produit et dans un second temps avoir son avis afin de ré-ajuster la création. Nous réalisons alors un planning des missions de la semaine.

VI) DIFFICULTÉS RENCONTRÉES ET SOLUTIONS

Au cours de cette année d'apprentissage, j'ai dû faire face à différentes difficultés. La première a été l'apprentissage du langage. React-Native est un langage similaire au JavaScript pour la partie développement en arrière-plan et similaire au Html et CSS pour le côté graphique. Je me suis souvent retrouvée frustrée de ne pas savoir faire des choses que je sais possibles dans d'autres langages. S'ajoute à cela le fait que le langage est récent et la documentation n'est pas toujours très complète. Peu à peu, j'ai pris le réflexe de chercher davantage sur les forums, ou de tester ce que je pensais possible. Actuellement ce n'est plus une difficulté pour moi je suis à l'aise avec le fait de chercher constamment de la documentation.

La seconde difficulté rencontrée concerne la relation client. Je dispose de maquettes pour certaines pages mais pas toutes. Il était donc difficile pour moi de passer du temps sur un graphique qui n'allait pas du tout plaire au client. Au début du processus, je demandais au client de dessiner d'autres maquettes. Je me suis rendu compte que ce n'était pas une bonne solution car le client ne sait pas ce qu'il est possible de faire graphiquement en code. Quelque temps après, j'ai pris l'initiative de réaliser moi-même des maquettes avec l'outil Canva. Sur ces maquettes je représente différentes options ce qui permet à l'équipe d'Hubert de choisir ou de me donner d'autres idées facilement représentables.

Lorsque nous sommes arrêtés sur un graphique, je le reproduis au mieux sur l'application.

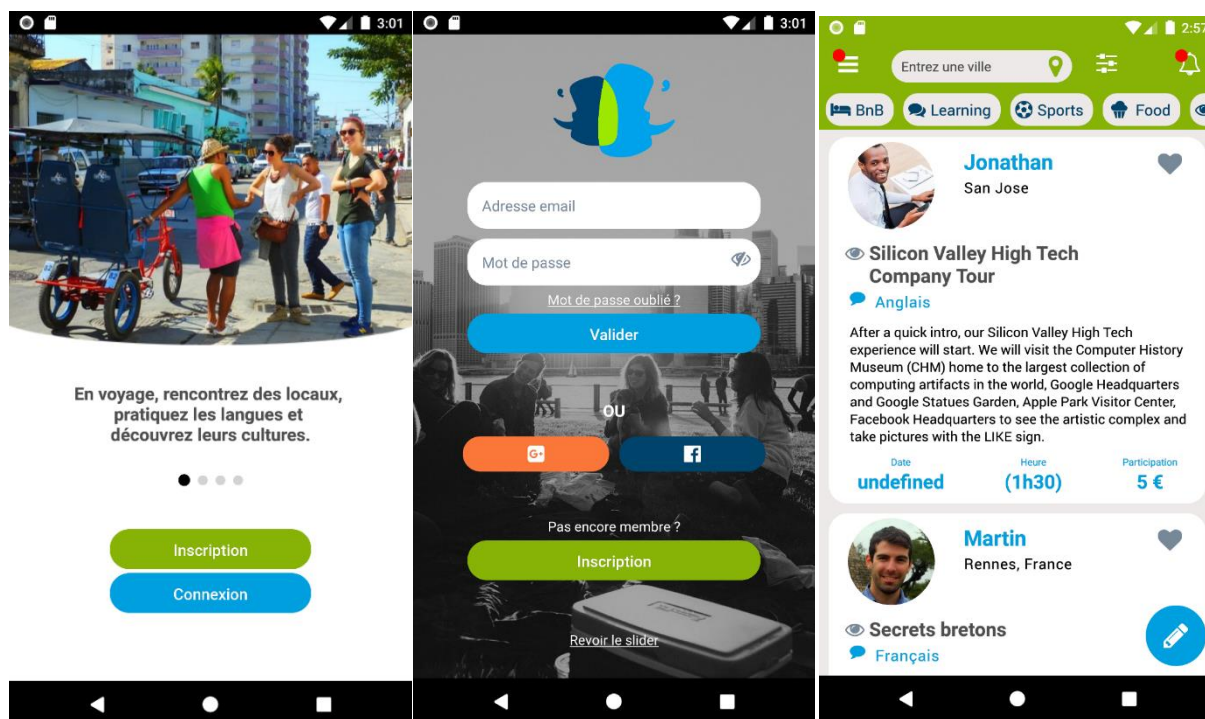


Exemples envoyés au client

VII) VISUELS

Voici quelques visuels des pages principales.

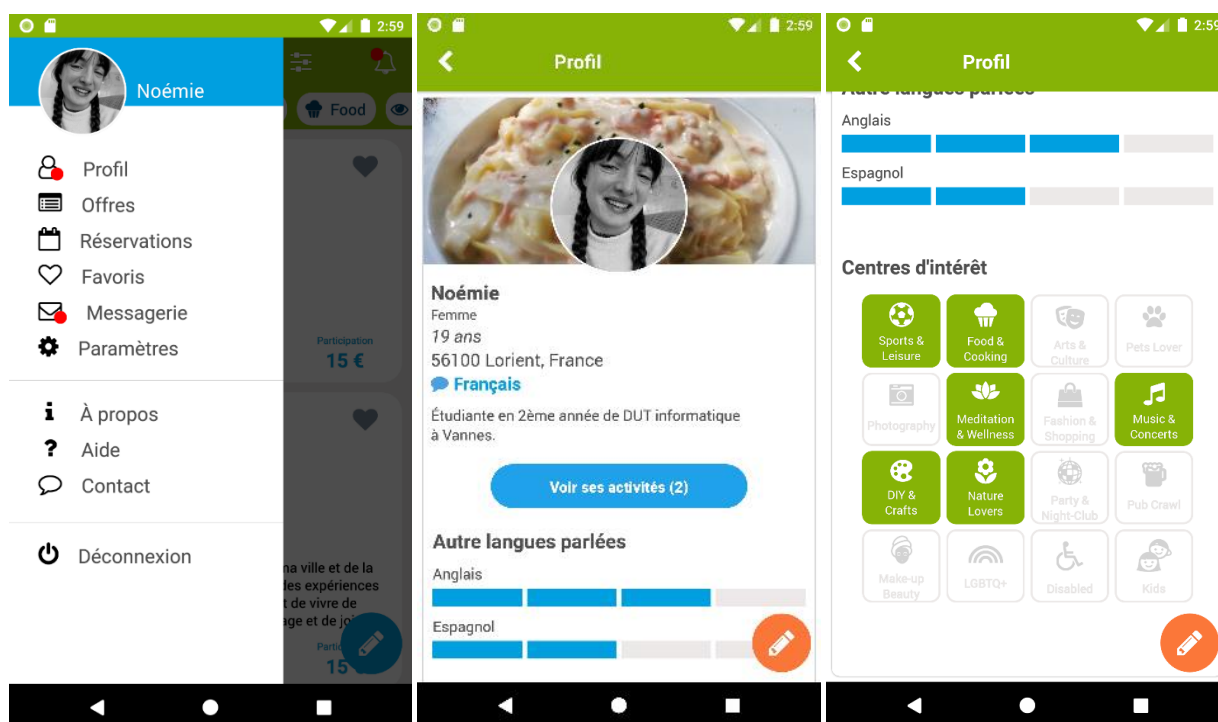
Au lancement de l'application, deux solutions. Si nous ne sommes pas connectés, nous arrivons sur les écrans 1 puis 2. Si nous le sommes et nous arrivons directement sur l'écran 3.



Écran 1 : Diaporama avec des phrases d'accroche pour présenter l'application. En cliquant sur les boutons inscription ou connexion, on arrive sur l'écran 2.

Écran 2 : Ici est représenté la page de connexion. On rentre son adresse mail et son mot de passe. On peut également se connecter via Facebook ou Google. Pour ce qui est de l'inscription, il faut simplement entrer une adresse mail, un mot de passe et le confirmer.

Écran 3 : Page d'accueil de l'application. Les activités affichées seront les activités proches de notre location géographique. Grâce au boutons bleu en bas à droite, on peut créer une activité. Avec le menu du haut on peut de gauche à droite, ouvrir l'écran 4, choisir une ville, filtrer les résultats et enfin ouvrir le panneau de notifications.



Écran 4 : Menu de navigation. En cliquant sur l'onglet profil, on arrive aux écrans suivants.

Écran 5 et 6 : Le profil utilisateur se présente de cette façon qu'il s'agisse du nôtre ou de celui d'un autre utilisateur. On y retrouve des informations personnelles, ses langues et ses centres d'intérêts. En cliquant sur le bouton orange en bas à droite on peut modifier ces informations.



Écran 7 : Édition du profil.

Tentons maintenant de réserver une activité.

Crêpes Party
français
Rennes, France
15€ participation

Bonjour, je suis Maëlle. Amoureuse de ma ville et de la culture bretonne, j'ai décidé de réaliser des expériences culturelles & culinaires, vous permettant de vivre de véritables moments épicuriens, de partage et de joie !

Votre hôte
Maëlle, 28 ans
undefined (à couper si trop longue)
[Contacter](#) [Voir son profil](#)

Prochaines sessions [Rechercher](#)
undefined - undefined
Nombre de places restantes : undefined
[Précédent](#) [Suivant](#)

Nombre de participants
- 1 +
[Réserver 1 place](#)

Paieement

Récapitulatif de la réservation :

Crêpes Party avec Maëlle
en français
le undefined
RDV : LIEU RDV
[voir sur la carte](#)

Nb de participant(s) : 1 personne
Montant : 15€

L'hôte à précisé : précisions
☐ J'accepte les [CGU](#)

Payer 15€

Écran 8 et 9 : Sur la page d'une activité, on retrouve son titre ainsi que sa description, son prix, les informations sur la personne proposant cette activité et enfin les créneaux disponibles. Il faut sélectionner un créneau, un nombre de place et cliquer sur « réserver »

Écran 10 : Avant de réserver, il y a un écran qui récapitule les informations importantes. Puis on accède au paiement.

Coordonnées bancaires :

Numéro de carte
4242 4444 4444 4244

Date d'expiration
Mois (XX) Année (XXXX)

Cryptogramme ⓘ

Valider

Écran 11 : On peut entrer ici ces coordonnées bancaires et valider le paiement.



VIII) AUTRES PROJETS

NOTINFO

Le projet Lyon Notinfo consiste en la création et la mise en production d'une application d'information, en temps réel, des actualités du Conseil régional des notaires de Lyon.

Je n'ai pas travaillé sur le visuel mais uniquement sur la mise en production et plus particulièrement sur l'Apple store. Le procédé à suivre pour déposer une application sous IOS est bien plus complexe que pour Android. En effet, il faut signer de nombreux certificats, respecter des contraintes de légalité et fournir une vingtaine d'icônes de tailles différentes.



MERAGITE

Mer Agitée est une écurie de course au large. Ils ont souhaité développer un système connecté permettant d'accéder aux réglages des voiles, de jour comme de nuit, qu'importe les conditions météo. Actuellement, les bateaux disposent de brins de laine (penons) attachés aux voiles. Selon la direction du brin de laine, on peut en déduire des informations sur le vent. Cette méthode n'est pas toujours fiable car la nuit on ne le voit pas et par temps de pluie, le penon peut rester collé à la voile.

Ils ont donc créé avec Goulven, des penons connectés. L'objet connecté se colle sur la voile. Il s'agit d'une languette de plastique attachée à un boîtier qui contient des capteurs et un émetteur. Le tout relié à une application mobile qui récupère les données en temps réel.



A la suite de ce projet, Mer Agitée a souhaité élargir le cas d'utilisation des penons. Ils souhaitent désormais les attacher sur des éoliennes et les connecter non pas à une application mais à un Raspberry pour récupérer les données et les envoyer sur un serveur.

Mon travail sur ce projet se décompose en plusieurs parties. Après avoir configuré, installé et testé le Raspberry et les logiciels nécessaires (Node essentiellement), il m'a fallu créer un script capable de récupérer les données envoyées par le penon. Une fois écrit, j'ai rajouté du code pour qu'il soit capable d'écrire ses données dans un fichier. La partie suivante consiste à faire une boucle qui permet de stocker les trames reçues chaque 10 minutes dans un même fichier. Je n'ai pas réussi cette partie car les nouvelles trames écrasaient les anciennes. La dernière étape consiste à écrire un script qui envoie ces fichiers sur un serveur.

Malheureusement je n'ai pas pu assembler tous les éléments car il me manquait une partie importante.

IX) CONCLUSION

Ce stage a été très enrichissant pour moi car il m'a permis de découvrir dans le détail le secteur du développement mobile ainsi que la vie dans une petite entreprise.

J'en retiens de nombreux points positifs. Travailler sur un projet qui me tient à cœur et que je comprends pleinement est une vraie force car cela a renforcé mon implication et ma prise d'initiative. S'ajoute à cela l'excellente relation client qui m'a permis d'avoir une grande autonomie en proposant des idées ou des maquettes.

Cette alternance a vraiment été très enrichissante pour moi j'ai adoré travailler sur ce projet, cela confirme mon désir d'orientation vers une carrière en développement web et mobile. La dimension artistique me plaît énormément ce qui me poussera à compléter mes études par une formation en web design.

Mon opinion a changé sur les conditions de travail. Je souhaitais travailler seule et je n'aimais pas travailler en équipe et finalement après avoir passé une année à travailler seule ou à deux je me suis rendu compte que ce n'était pas ce qui me convenait le mieux. Je préfère travailler avec du monde même si nous ne travaillons pas tous sur la même chose. Les journées passées chez le client me motivaient davantage que celles passées seule. J'aimerais donc effectuer mes futurs stages ou alternances dans de plus grosses entreprises ou bien travailler dans des espaces de travail partagé par exemple.

X) RESUMÉ / SUMMARY

Pour ma deuxième année de DUT informatique, j'ai expérimenté l'alternance. J'ai travaillé toute l'année au sein de l'entreprise CyberEspar à Larmor-Baden. Ma mission consiste à créer une application mobile nommée TalkTalkers.

TalkTalkers est une application collaborative sur la thématique des langues. Elle permet de rencontrer des locaux lors de vos voyages, pour converser avec eux et progresser en langue. Vous pouvez vous inscrire à des activités comme la conversation, cours de cuisine, sport ou visite culturelle dans la langue de votre choix.

During my second year of technology degree in computer science, I tried sandwich-courses. I spent my whole year working for a company called CyberEspar and based in Larmor-Baden. My mission is to develop the TalkTalkers app.

TalkTalkers is a collaborative app all about languages. It helps you meet local people while abroad, for conversation and to improve your language skills. You can register for activities like conversation, cookery course, sport or cultural visit in the language of your choice.