Dossier de projet professionnel

Réalisation de l'application web et mobile

—

TODOLIST

Présenté par Boris Melono

SOMMAIRE

Introduction

Présentation personnelle

Je m'appelle Boris Melono , j'ai 27 ans. J'ai découvert la programmation suite à une formation que j'ai suivie avec Simplon, qui m'a permis de découvrir tous les métiers liés au numérique. J'ai suivi le cursus de Développeur web/Mobile en 2020 à Simplon. Aujourd'hui je suis en Coding School 2 afin de préparer mon titre de concepteur / développeur d'application web et en alternance au sein de mon centre de formation La Plateforme.

Présentation du projet en anglaise

In business I have only worked on cms, suddenly in agreement with my trainers I decided to offer you an application project developed during the year by me.
is literally a to-do list. These tasks can be independent of each other.
I used the React Framework for the development of the mobile application and for the website.
For database and deployment, I used Firebase.

Compétences couvertes par le projet

Ce projet couvre les compétences du titre suivantes :

- Maquetter une application
- Développer des composants d'accès aux données
- Développer la partie front-end d’une interface utilisateur web
- Développer la partie back-end d’une interface utilisateur web
- Concevoir une base de données
- Mettre en place une base de données
- Développer des composants dans le langage d’une base de données
- Concevoir une application
- Développer des composants métier
- Construire une application organisée en couches
- Développer une application mobile
- Préparer et exécuter les plans de tests d’une application
- Préparer et exécuter le déploiement d’une application

Organisation et cahier des charges

Analyse de l'existant

Il existe déjà de nombreuses applications de type todolist mais chacune avec des besoins différents, pour ce projet, j'ai choisi pour mon applis un projet répondant aux besoins des compétences

Les utilisateurs du projet

Ce projet se décompose en une partie application mobile, La partie application mobile sera utilisée avec une connexion de type facebook et la partie web selon mes configurations on ne pourra pas avoir accès à l'application avec une authentification par facebook pour avoir accès à l'application.

Les fonctionnalités attendues

Application mobile & web

Page d'accueil login :

Lorsque l'utilisateur lance l'application mobile, il est redirigé vers une page d'accueil vers laquelle il y'a une procédure d'authentification.

Page Todolist :

Sur cette page nous avons la possibilité de créer une todolist avec des choses à faire, de pouvoir noté les éléments déjà fait ou pas, de les supprimer ou non.

Page logout :

Pour pouvoir, se déconnecter de son espace todolist.

Contexte technique

L’application mobile devra être accessible sur tous les systèmes d’exploitation Android et IOS ainsi que sur le navigateur web.

Conception du projet

Choix de développement

Choix des langages

Pour réaliser ce projet, j'ai décidé d'utiliser uniquement du javascript.

Avec Node Js comme environnement de développement.

J'ai fait ce choix pour plusieurs raisons : Javascript est un langage riche avec de nombreux concepts, me permettant une montée en compétences. C'est un langage beaucoup utilisé par les géants du web, ces derniers créer de nombreuse bibliothèque de code open source facilitant ainsi le développement de certaines fonctionnalités. Il est présent dans toutes les applications web mais aussi mobiles. Il n'existe à ce jour plus aucunes pages web qui n'utilisent pas cette technologie pour dynamiser son contenu.

Choix des frameworks

Pour la création de mon API j'ai choisi d'utiliser firebase ainsi qu'EXPO qui est le nom d'une plateforme mobile de Google qui facilite la création de back-end, qui permet de développer, de façon rapide et efficace, une API. Ce qui correspond parfaitement à mon besoin.

Pour la création de mon application j'ai choisi d'utiliser le framework React native qui permet de créer des applications mobiles open source créée par Facebook. Il est utilisé pour développer des applications pour Android, iOS et UWP.

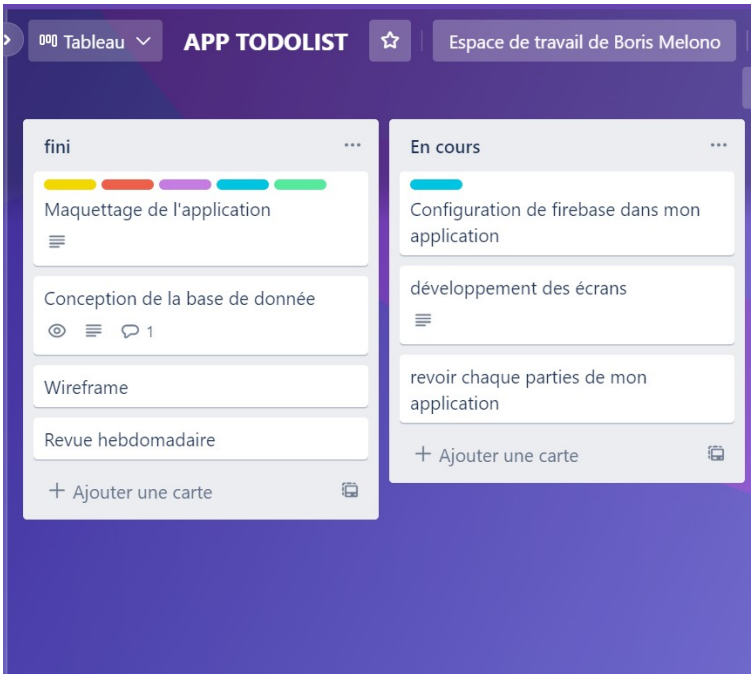
Logiciels et autres outils

Dans le cadre de ce projet, j'ai dû utiliser d'autres outils:

- Visual Studio Code pour écrire mon code;
- Postman pour effectuer les requêtes API;
- GitHub pour le versionning de mon code;
- Trello pour organiser mon travail ;
- NPM pour installer les paquets ;
- Figma pour la création de mes maquettes ;
- Draw.io pour le maquettage de ma base de données et canva pour la création de ma charte graphique et logo
- Expo pour émuler mon application mobile sur mon téléphone ;

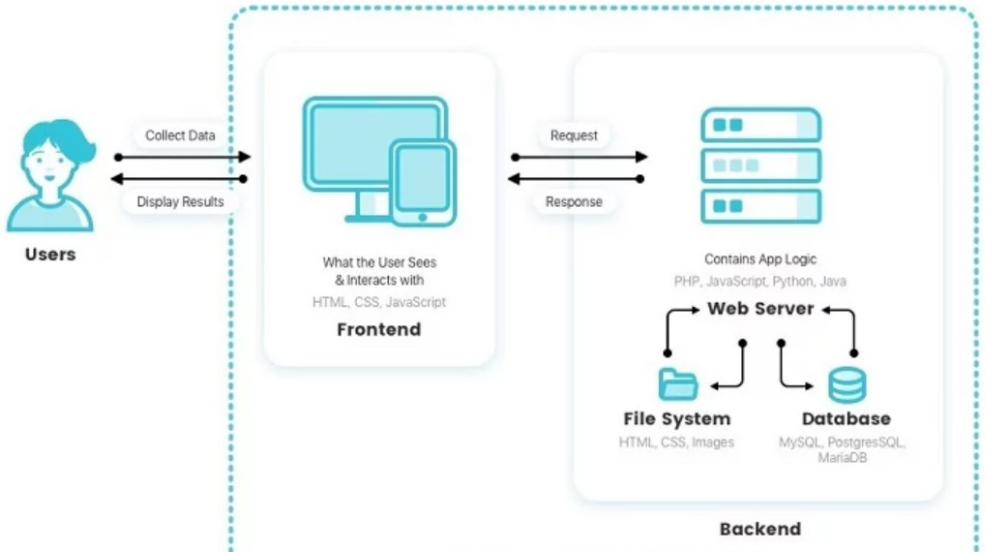
Organisation du projet

Ce projet a été réalisé durant mon année de formation, avec des temps d'entreprise et d'autres projets à rendre. Donc l'organisation de mon travail à été essentielle.
J'ai commencé le projet par un listing des tâches principales et essentielles.
Pour ce faire, j'ai tout d'abord créé Trello.
C'est un outil qui m'a permis de planifier mon projet, ainsi d'avoir une vue d'ensemble des tâches à effectuer. Il m'a permis de suivre l'avancement de mon projet, et faire des ajustements afin d'optimiser le temps que j'avais sur ce projet.



Architecture logicielle

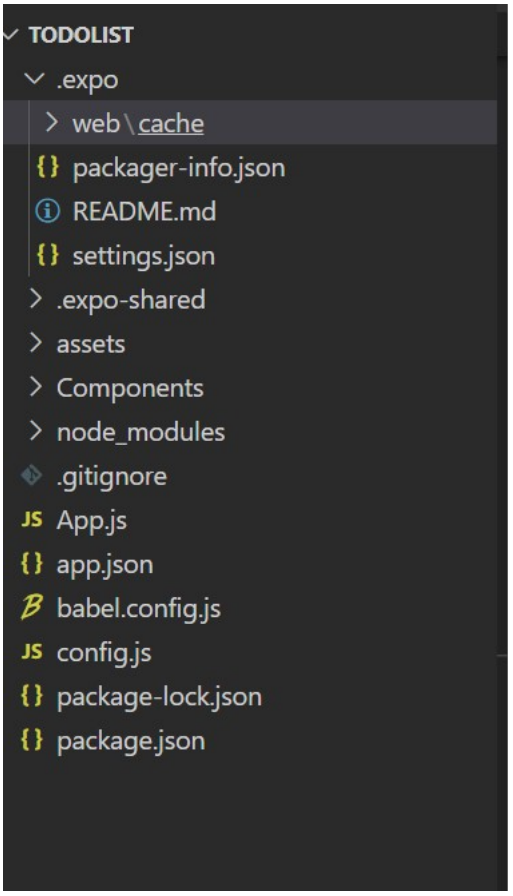
Les utilisateurs s'authentifieront, L'application mobile et le site web utilisent la même API. L'API communique avec ma base de données



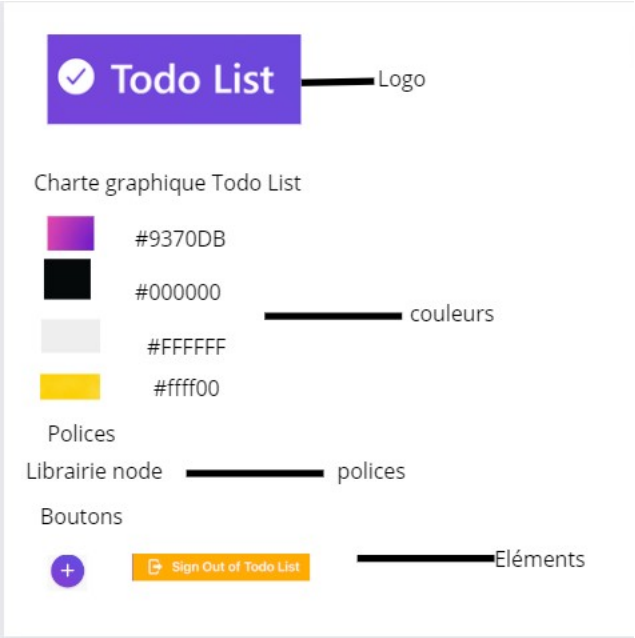
Conception du front-end de l’application

Pour créer la maquette et la charte graphique, j’ai choisi d’utiliser Canva, car il est gratuit, plutôt simple d’utilisation et permet la réalisation de maquettes réalistes.

Arborescence du projet



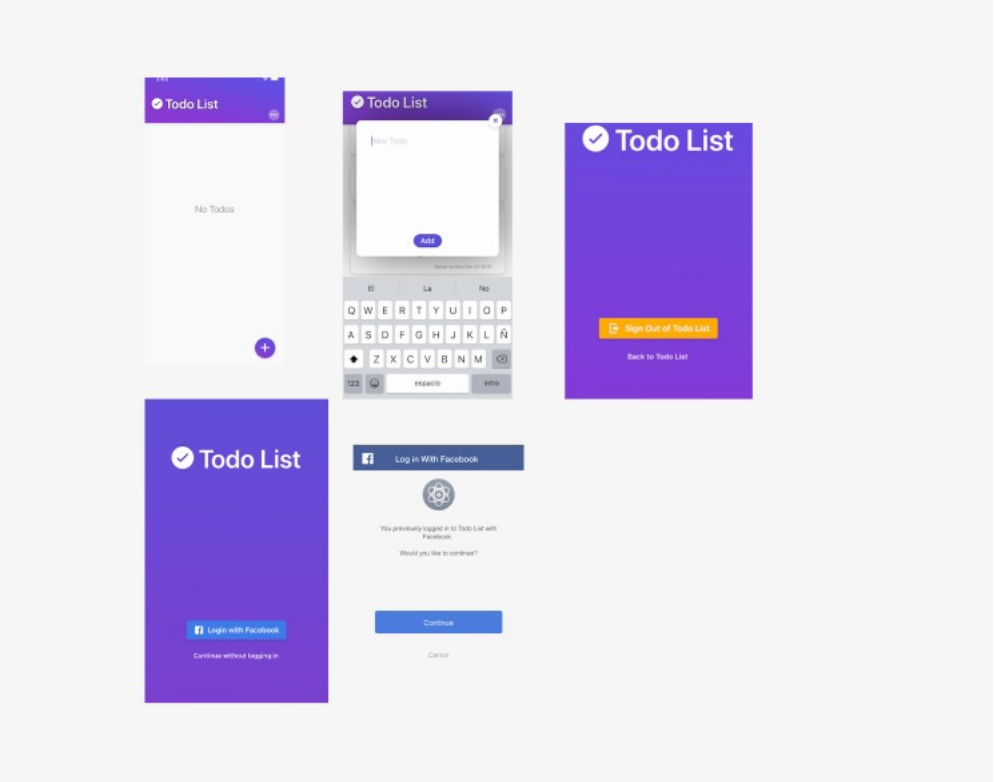
Charte graphique



Après la création de l'arborescence du projet, j'ai commencé par la création du logo à l'aide de canva. J'ai ensuite récupéré les couleurs de celui-ci pour créer la charte graphique et rajouté d'autres couleurs.

Maquettage

Les maquettes ont été créées à l'aide de Figma. Elles se trouvent en annexe de ce dossier.



Conception backend de l’application

La base de données

Mise en place de la base de données

Pour ce projet, il est indispensable d’avoir une base de données, notamment au niveau de l’authentification et les personnes qui vont créer des tâches.

Du au temps court et le fait que je sois seul sur ce projet je me suis tournée vers la base de données Firebase qui utilise un système NoSQL.

Durant mon apprentissage, j’ai beaucoup travaillé sur les bases de données MySQL, mais pour ce projet, j’ai dû me tourner vers firebase .

Conception de la base de données

Pour concevoir ma base de données j’ai utilisé firebase en configurant, par la suite en ajoutant un bout de code permettant l’authentification pour accéder à mon application.

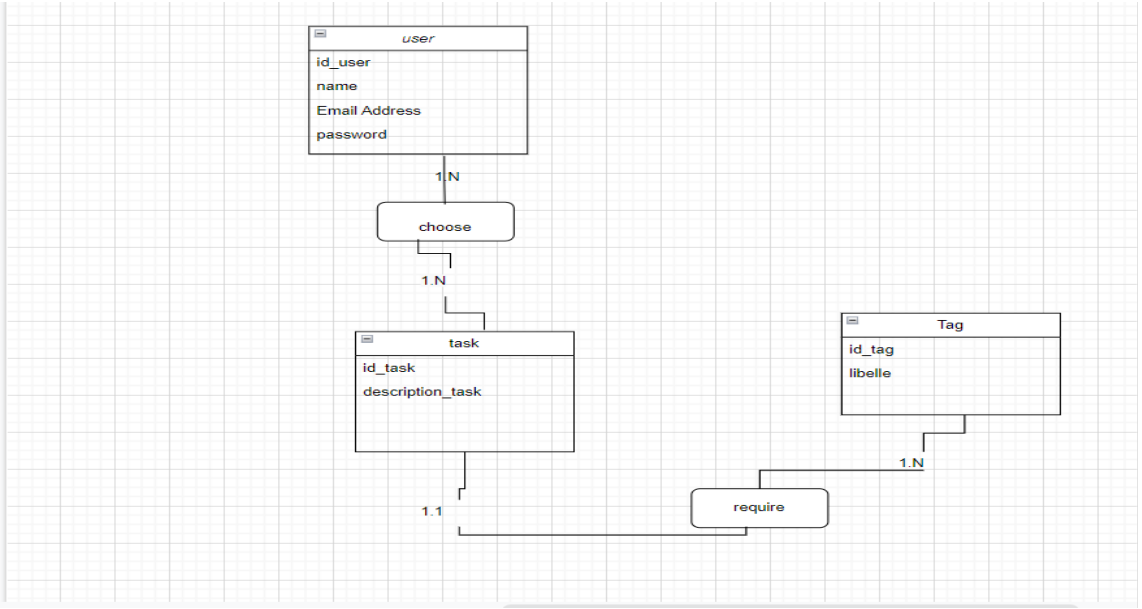
Voici un aperçu des données pertinentes qui m’ont aidé à construire une représentation claire des besoins :

- Les utilisateurs peuvent se connecter à l'application en utilisant la connexion sociale de Facebook. Lors de l'authentification, un nouveau profil utilisateur dans la base de données de Firebase sera créé. Cela permettra aux utilisateurs d'enregistrer leurs tâches dans le cloud, permettant aux données d'être poussées et extraites de n'importe quel appareil compatible sans avoir à créer un nouveau profil spécifique au service.

J’ai construit ma base de données, en procédant en deux étapes :

Modèle conceptuel de données

La première étape a été la création du modèle conceptuel des données. J’ai créé des entités en fonction du dictionnaire de données récoltés. J’ai donc dessiné dans un premier temps mes entités en leurs donnant un nom.



Modèle logique de données

J'ai ensuite continué la création de ma maquette de base de données en créant le modèle logique de données.

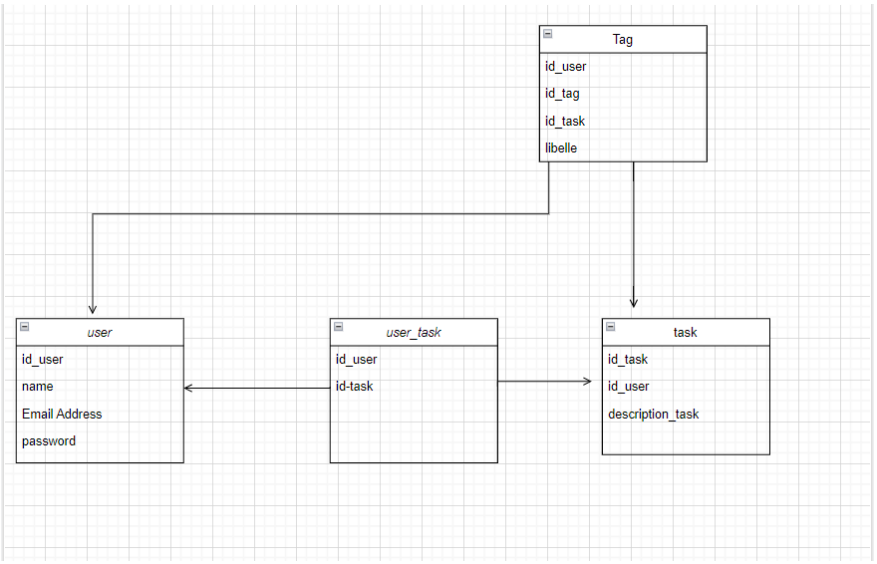
J'ai créé une clé primaire pour chaque entité qui permet d'identifier sans ambiguïté chaque occurrence de cette entité.

Ensuite, j'ai rempli chaque entité avec des attributs en reprenant les informations de mon recueil de données.

J'ai procédé à la création des cardinalités de chaque entité.

Exemple :

Un utilisateur peut avoir 1 tache ou N taches et 1 ou N taches peuvent être attribués à 1 ou N utilisateurs



Développement du backend de l'application

Organisation

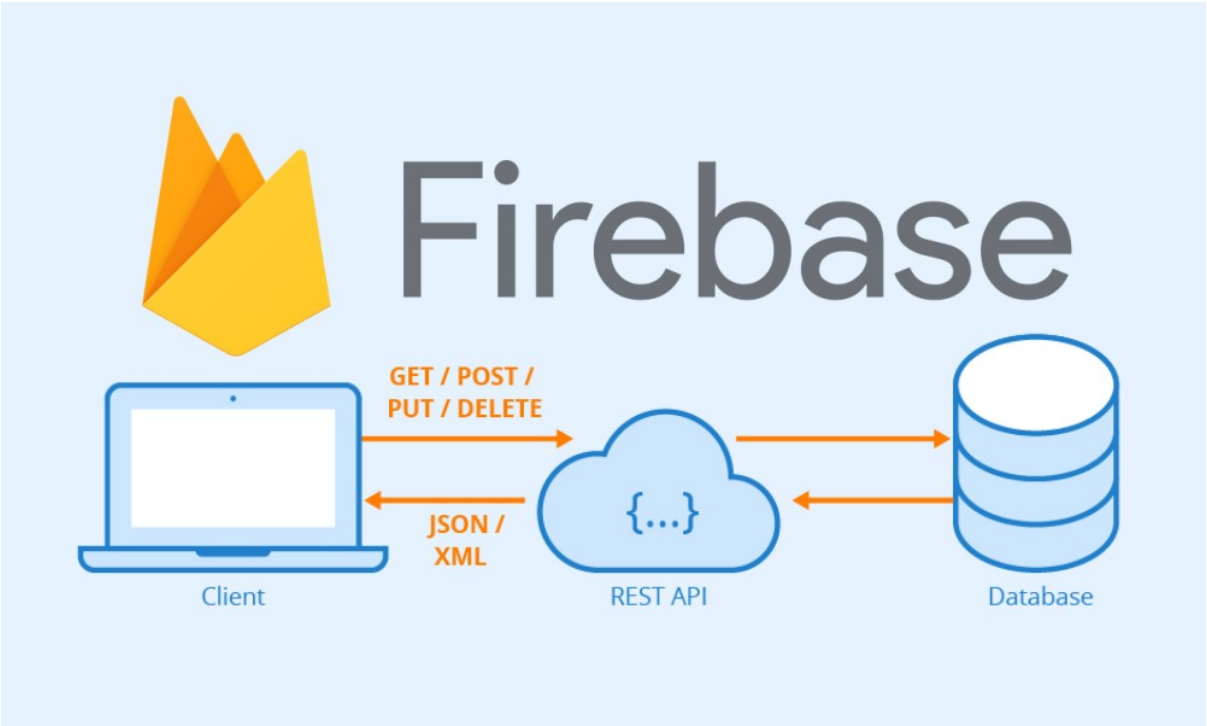
Mon back end a pour but d'être utilisé à la fois pour mon application web et mon application mobile. J'ai décidé de créer une API Firebase et par la suite j'ai déployé mon projet sur Firebase .

Mon back end est donc composer des dossiers suivants :

- **Firebase** : regroupant tous les fichiers pour la connexion à ma base de données
- **Fonctions** : regroupant tous les fichiers contenant les fonctions
- **Test** : destiné aux test unitaire - Jest

Fonctionnement de l'API

La base de données Firebase Realtime est une base de données hébergée dans le cloud. Les données sont stockées au format JSON et synchronisées en temps réel avec chaque client connecté. Lorsque vous créez des applications multiplateformes avec nos plates-formes Apple, Android et les SDK JavaScript, tous vos clients partagent une instance de base de données en temps réel et reçoivent automatiquement des mises à jour avec les données les plus récentes.



Les différents statuts utilisés dans ce projet sont :

- **200** : OK
  Indique que la requête a réussi
- **201** : CREATED
  Indique que la requête a réussi et une ressource a été créé
- **204** : NO - CONTENT
  Indique que la requête a bien été effectué et qu'il n'y aucune réponse à envoyer
- **400** : BAD REQUEST
  Indique que le serveur ne peut pas comprendre la requête à cause d'une mauvaise syntaxe
- **401** : UNAUTHORIZED
  Indique que la requête n'a pas été effectuée car il manque des informations d'authentification
- **403** : FORBIDDEN
  Indique que le serveur a compris la requête mais ne l'autorise pas
- **404** : NOT FOUND
  Indique que le serveur n'a pas trouvé la ressource demandée
- **405** : METHOD NOT ALLOWED
  Indique que la requête est connue du serveur mais n'est pas prise en charge pour la ressource cible
- **500** : INTERNAL SERVER ERROR
  Indique que le serveur a rencontré un problème.

Les différentes méthodes HTTP utilisées dans ce projet :

- **GET** - Pour la récupération de données
- **POST** - Pour l'enregistrement de données
- **PUT** - Pour mettre à jour l'intégralité des informations d'une donnée
- **PATCH** - Pour mettre à jour partiellement une donnée
- **DELETE** - Pour supprimer une donnée

Routage

Pour ce qui est du routage, voici le code de notre configuration firebase pour l'accès à travers l'authentification à notre base de données voici ci-dessous notre code :

```
import 'firebase/compat/auth';
import 'firebase/compat/firestore';

import LoginScreen from './Components/LoginScreen';
import ToDoList from './Components/TodoList';

const firebaseConfig = {
    apiKey: "AIzaSyA-UNc5Qguslo99Jc3-mKHfJr39eeq85Ps",
    authDomain: "todolist-b4339.firebaseapp.com",
    projectId: "todolist-b4339",
    storageBucket: "todolist-b4339.appspot.com",
    messagingSenderId: "955932952316",
    appId: "1:955932952316:web:5fa00a5d890b8ccfb23979",
    measurementId: "G-R16SX02SXQ"
  };

```

Sécurité

Les API sont un moyen d'appeler des informations provenant de la base de données ainsi, il existe de nombreux risques.

Les différents risques sont :

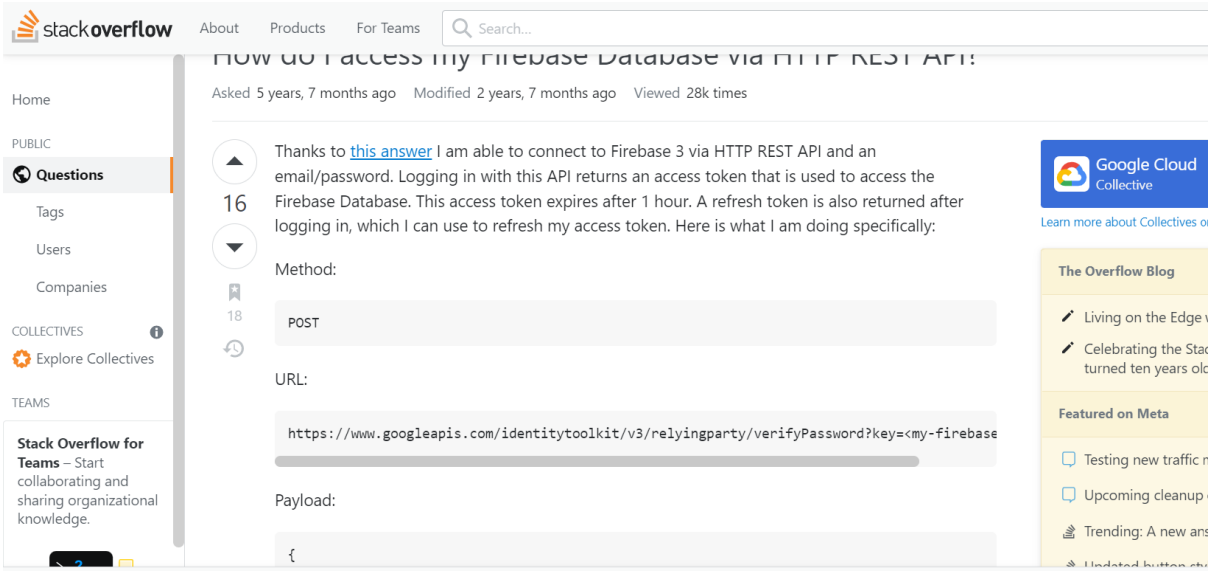
- les injections SQL : Les injections consistent à insérer dans les codes un programme un autre programme malveillant permettant ainsi d'attaquer directement une base de données et y prendre le contrôle. L'attaquant peut alors se servir librement de toutes les informations récoltées.
- Credential stuffing : vol du login et password.
- Attaque DDOS : envoie de trafic en vue de surcharger le trafic d'une api
- Man-in-the-moddle : consiste à pousser un utilisateur à se connecter a un service compromis, qui pourra alors s'emparer du jeton ou de la clés de l'utilisateur.

Pour rendre mon API sécurisé j'ai mis en place un système d'authentification avec l'aide de firebase l'authentification va souvent de pair avec l'autorisation ; alors que le premier est utilisé pour *VÉRIFIER L'IDENTITÉ* d'un utilisateur d'une plate-forme, le second est utilisé pour autoriser ou refuser l' *ACCÈS* aux ressources en fonction de l'identité de l'entité demandant la ressource.

```
firebase.auth().onAuthStateChanged((user) => {
    if (user != null) {
        console.log('We are authenticated now!');
    }
    // Do other things
});

const MainNavigator = createStackNavigator(
    {
        LoginScreen: LoginScreen,
        ToDoList: ToDoList,
    },
    {
        initialRouteName: 'LoginScreen',
```

Recherches anglophones

Pour résoudre certains problèmes notamment sur l'utilisation et la configuration de mon API firebase , je me suis aidé de qui est en anglais.



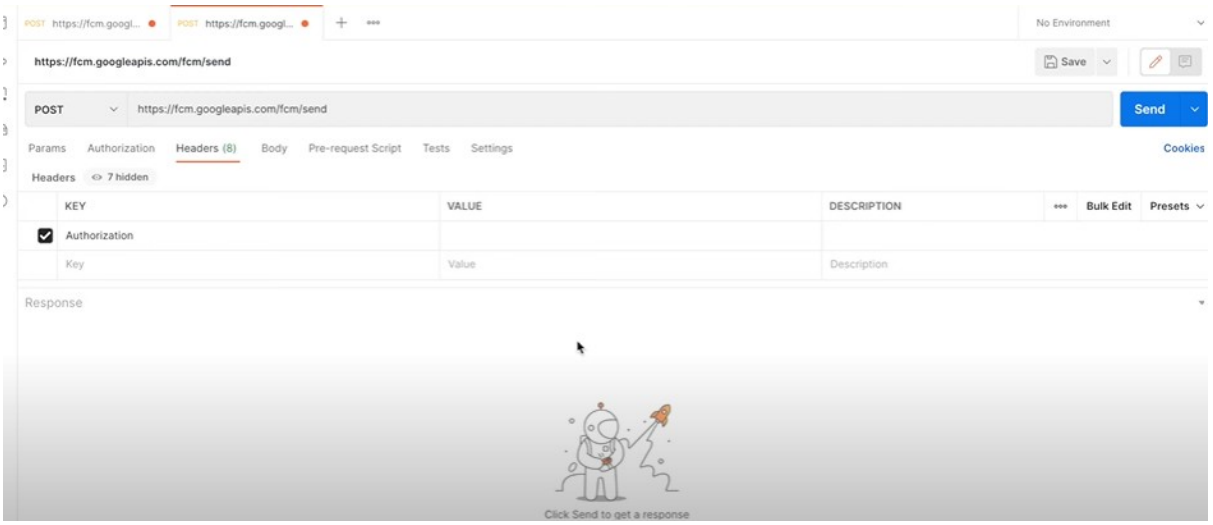
Documentation

Le but d'une API est d'être utilisé par d'autres développeurs. Ainsi il est important d'avoir une documentation.
La documentation fait référence aux contenus techniques avec des instructions claires sur le fonctionnement de celle-ci.
Il est nécessaire qu'elle soit mise à jour quand le code évolue ou que d'autres fonctionnalités sont ajoutées.
La documentation de l'API est disponible sur le site https://firebase.google.com/docs/reference c'est un outils permettant d'aider les développeurs dans la conception, le build, la documentation d'API.

Tests

Postman

Mon API expose des données, il est donc important de tester son API afin de s'assurer de la qualité des données exposées.
A chaque modification ou création au niveau de mon application, j'ai effectué des tests avec Postman afin de m'assurer du fonctionnement ou non de mon API.



A chaque test, je vérifie que les données envoyées sont bien celles attendues, qu'elles envoient les données, que le statut de la requête HTTP est correct. Je vérifie aussi que les erreurs sont bien gérées.

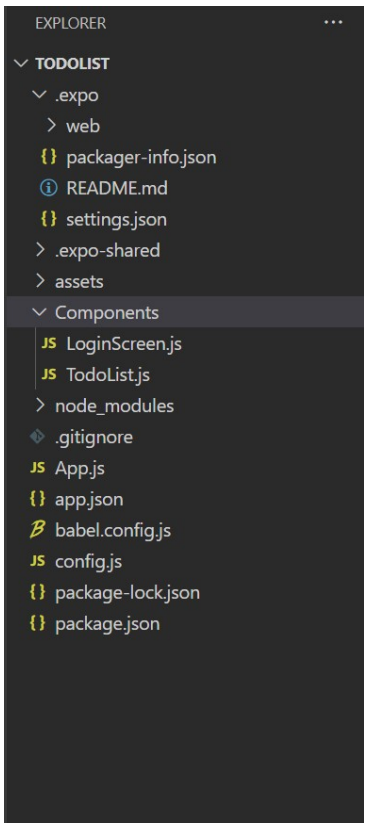
Jest

Mon projet étant complètement écrit en javascript, il est important d'avoir une couverture de test suffisante.
C'est pour cela que j'utilise Jest.

Jest est une librairie javascript conçue pour les tests front end et back end.
Je l'utilise donc pour tester les services de mon projet.

Développement du front-end de l'application

Arborescence



- Dans le dossier components on trouve tous les composants réutilisables ou pas.
- Dans le dossier fonctions se trouvent les différentes fonctions utilisées dans ce projet.

Pages et composants

Un composant permet de pouvoir découper une page en éléments indépendants et réutilisables.
Conceptuellement, les composants sont comme des fonctions JavaScript. Ils acceptent des entrées quelconque (appelées « props ») et renvoient des éléments

React décrivant ce qui doit apparaître à l'écran.

Le principe de react native est d'écrire un maximum de composants réutilisables.

Ainsi on évite de dupliquer le code.

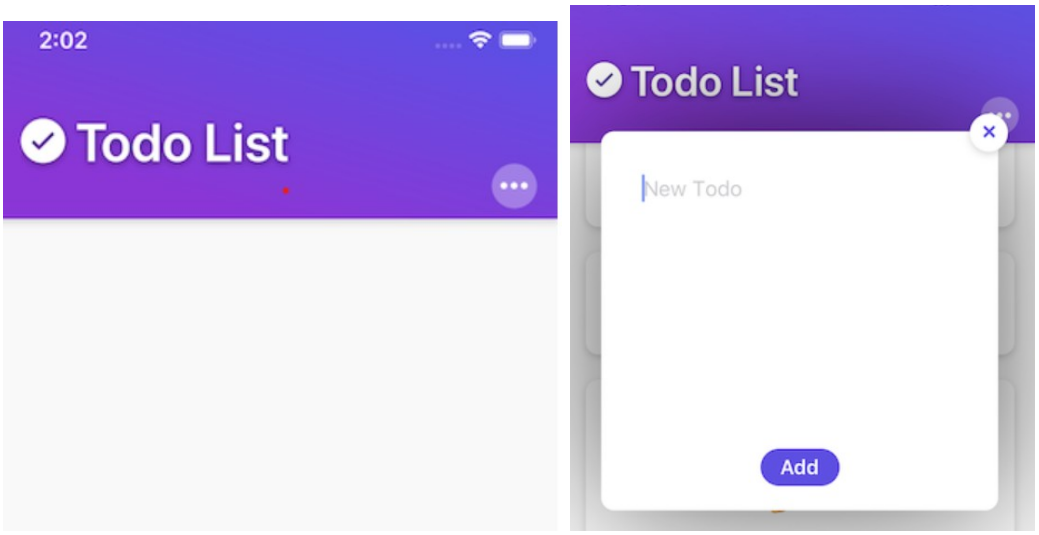
```
import {
    StyleSheet,
    View,
    Text,
    TouchableOpacity,
    StatusBar,
    ActivityIndicator
} from 'react-native';
import { LinearGradient } from 'expo-linear-gradient';
import { Ionicons, MaterialCommunityIcons } from '@expo/vector-icons';
```

```
          <Text
                              accessibilityRole='header'
                                      style={styles.header}
                    >
                                                <Ionicons name='ios-checkmark-
                circle' size={36} color="#fff" /> Todo List
                     </Text>
                      <TouchableOpacity style={styles.menuButton}
     onPress={actionSheet}>
                      <MaterialCommunityIcons name='dots-horizontal'
 size={27} color='rgba(255, 255, 255, 1)' style={{ top: 1 }} />
                   </TouchableOpacity>
          <Text
```

Voici un exemple d'un composant réutilisable, il permet l'affichage d'un bouton.

Dans mon projet, se trouvent de nombreux boutons avec des actions différentes.

Mais ils sont composés des mêmes éléments, TouchableOpacity et un Text.



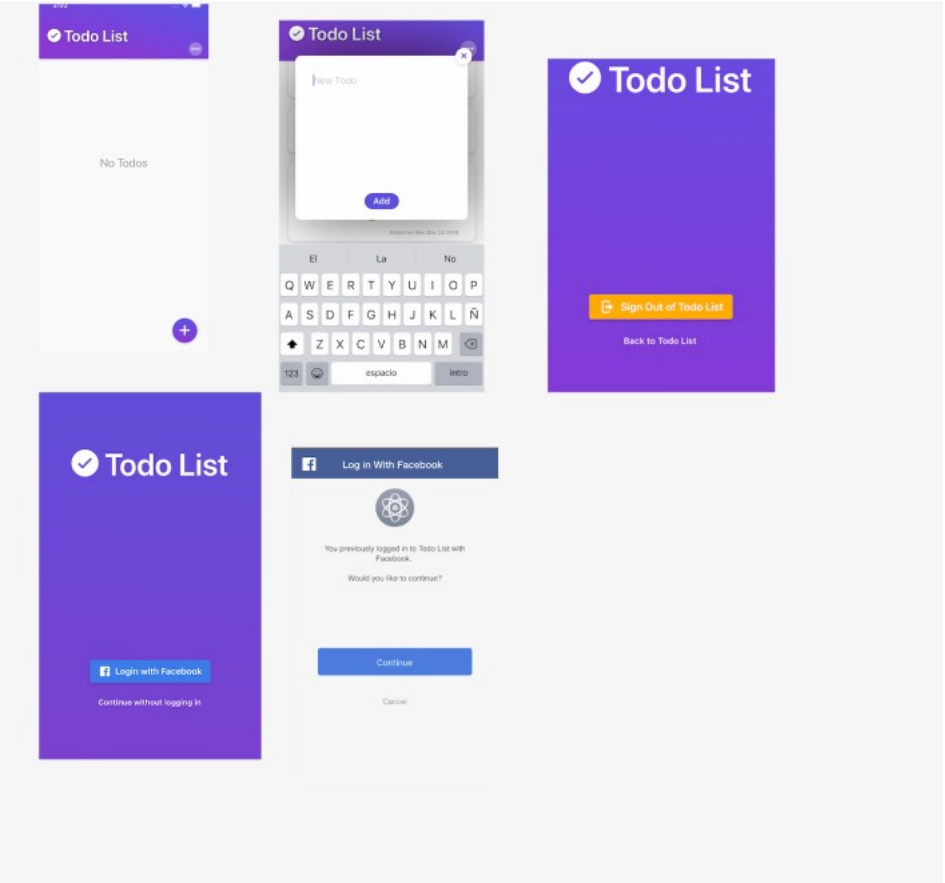

```
 <TouchableOpacity style={styles.menuButton} onPress={actionSheet}>
                        <MaterialCommunityIcons name='dots-horizontal'
size={27} color='rgba(255, 255, 255, 1)' style={{ top: 1 }} />
                          </TouchableOpacity>


          <TouchableOpacity onPress={addOrEditTodo}>
                        <Text style={styles.submitButton}> {state.editMode ?
'Update' : 'Add'} </Text>
                    </TouchableOpacity>
```

Comme on peut le remarquer sur l'image ci- dessus, les composants **menuButton et submittButton** permettent de créer les nombreux boutons de mon application.
Cet exemple il représente le bouton de mon menu , de mon bouton ajout et upadate.

Problématiques rencontrées

Durant le développement de mon application mobile, j'ai rencontré différentes difficultés. L'une de ces difficultés à été la navigation au sein de mon application.

Comme on peut le remarquer sur les maquettes du projet, voici comment naviguer dans mes différentes view :



Pour ce faire, j'ai dû comprendre la navigation dans react native mais aussi les différentes navigations dans celle-ci.
J'ai utilisé une bibliothèque React Navigation qui est très populaire dans React Native. Cette bibliothèque à résoudre le problème de navigation entre les pages et le transfert des datas entre celles-ci.

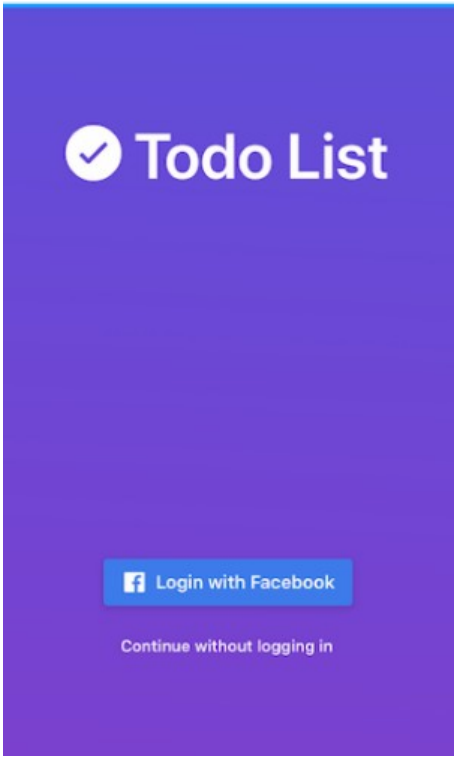
Exemple navigation imbriquée :

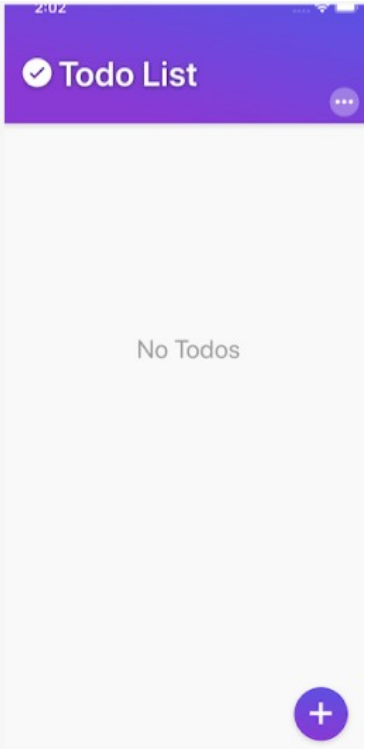
```
import React from 'react';
import { createAppContainer } from 'react-navigation';
import LoginScreen from './Components/LoginScreen';
import ToDoList from './Components/TodoList';

import React from 'react';
import { createAppContainer } from 'react-navigation';
import { createStackNavigator } from 'react-navigation-stack';
```

```
        const MainNavigator = createStackNavigator(
                {
        LoginScreen: LoginScreen,
                    ToDoList: ToDoList,
                },
    {
                    initialRouteName: 'LoginScreen',
                defaultNavigationOptions: {
                header: null
 },
    }
);
```

Ce composant est chargé de la navigation dans mon application mobile.

Pour qu'elle puisse fonctionner, il faut dans un premier temps importer react navigation ainsi que les différentes vues. Mon LoginScreen permet d'avoir accès à ma page d'authentification.





Lorsque l'utilisateur se connecte une autre view qui, s'ouvre c'est celle de l'ajout de taches

Stack Navigator

Conception de l'espace administrateur

Conception de la partie administration

Mon projet est composé d'une partie administration.
Celle-ci est gérée grâce à Firebase qui permet de gérer toutes les personnes qui pourront se connecter à mon application et les différentes données entrées.
J'ai commencé par créer une user story afin d'organiser mon travail.

User Story

Une personne qui peut se connecter aura accès à l'espace todolist qui permettra la création de todolist, sans cela la connexion sera compliqué, lorsque l'utilisateur est connecté ses identifiants sont enregistrés en base de données , avec toute les taches enregistrées.

Choix du langage et de ma base de données

Pour réaliser ce site j'ai utilisé React-Native pour le front , pour le back end , j'utilise le Firebase.

Conclusion

Pour conclure, ce projet m'a permis de découvrir de nouveaux frameworks et aussi de pouvoir monter en compétence sur javascript.

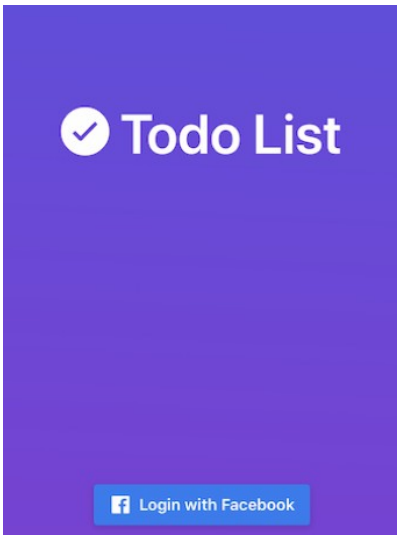
Ce projet qui s'est déroulé sur l'année de formation m'a appris à organiser mon travail.
En effet, durant cette année je n'ai pas seulement développé ce projet, j'ai aussi développé, énormément d'applications, développement d'une API en Symfony, une calculatrice en Réact js.
Malheureusement j'ai eu d'énorme problème de santé du au COVID donc j'ai dû développer mes projets tout seul pour ne pas retarder mes collègues.

De plus, le projet présenté dans ce dossier m'a permis de voir que je pouvais m'adapter aux différentes situations car j'ai dû m'adapter notamment sur le choix d'une base de données.

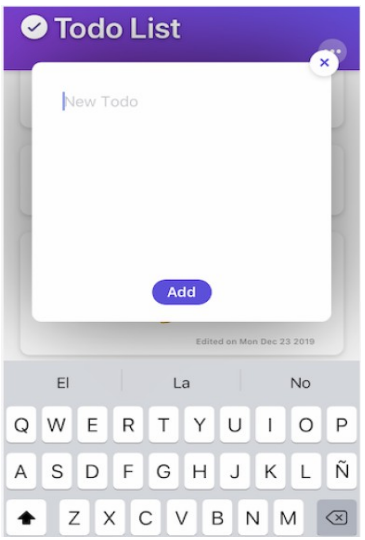
ANNEXE

Annexe 1 : Maquette de l'application mobile – TODO LIST

Page de Login :



Page ajout todolist :



Page de logout :