BORDEAUX INP Ensc

Projet Informatique Individuel

Création d'une application mobile

EasyLock - Rendu final

Valentine Espié

2022 - 2023

SOMMAIRE

1. Rappel du contexte **3**

2. Choix techniques **3**

3. Architecture de l'application **4**

4. Plannings **5**

5. Pistes d'évolution **5**

6. Bilan **5**

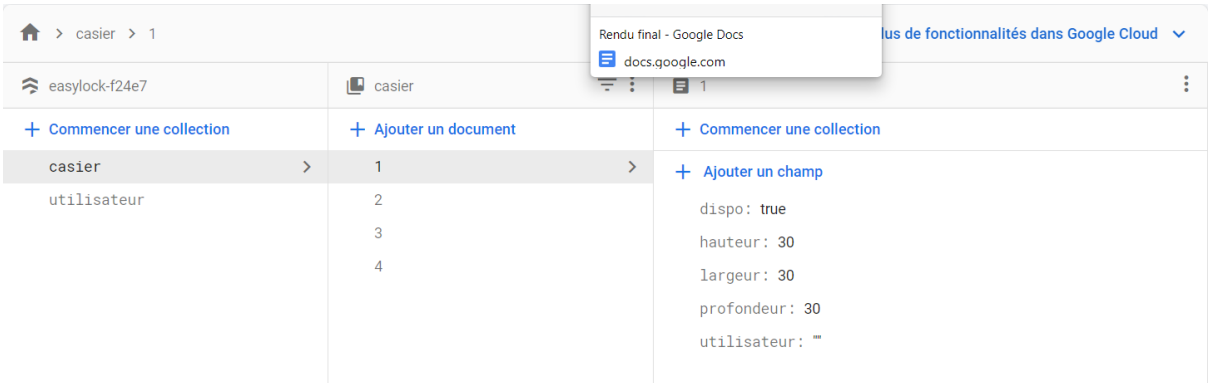
1. Rappel du contexte

Lors d'une sortie shopping avec ma mère, étant de grandes adeptes de cette pratique, elle m'a fait remarquer un des plus gros problèmes auquel nous sommes régulièrement confrontées lors de ce genre de sortie. Après le premier magasin, premier d'une longue série, nos bras sont d'ors et déjà chargés et nous sommes alors contraintes de les porter pour tout le reste de la journée. Ajoutez à cela une météo pluvieuse et des sacs en papier très peu solides et la journée de rêve peut rapidement tourner au cauchemar. C'est dans ce contexte qu'elle me soumet une idée qui solutionnerait notre problème : un système de consigne placé au centre de la ville, avec plusieurs casiers de différentes tailles pouvant se louer pendant quelques heures ou même quelques jours. En effet, au-delà de nos sacs de shopping, ces casiers peuvent être utiles pour y déposer les trottinettes électriques, les sacs de voyage ou encore un manteau lors d'une soirée en bar où celui-ci peut être encombrant mais aussi très utile lors d'un retour dans la nuit, lorsque les températures baissent.

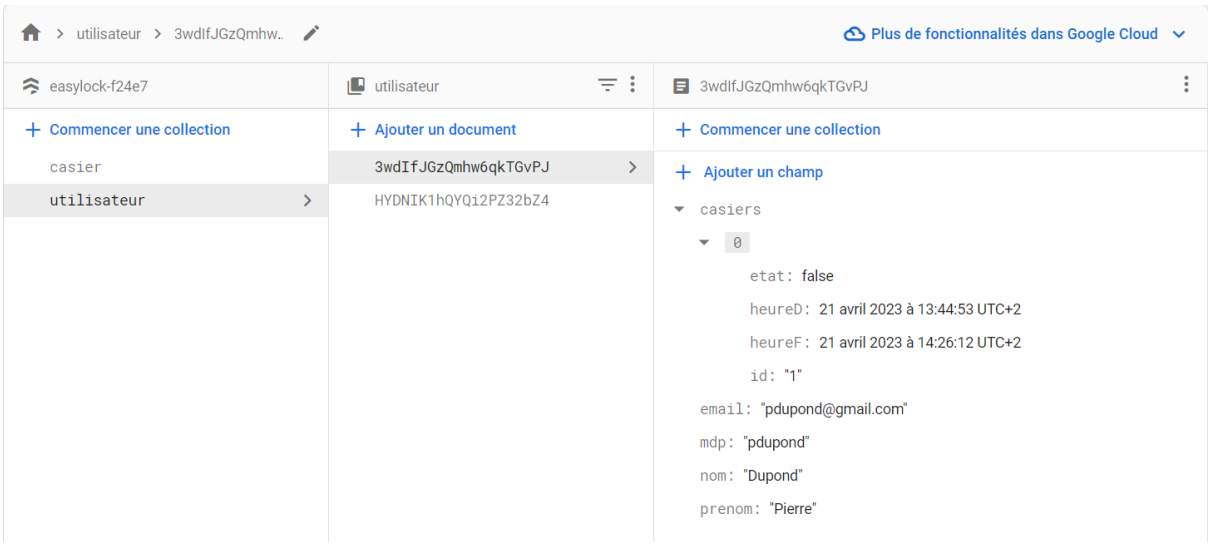
Il m'a donc semblé que cette idée pouvait tout à fait servir de sujet pour le projet informatique individuel. En effet, en ne s'attardant pas sur le côté matériel et sur les potentiels besoins des utilisateurs puisque ce n'est pas l'objectif de ce projet, il est tout de même possible de se pencher sur l'interface entre ce service et l'utilisateur. Celle-ci serait donc une application mobile permettant d'initier, réserver ou encore mettre fin à l'utilisation d'un casier du locker. Les différentes fonctionnalités seront détaillées par la suite.

2. Choix techniques

Afin de mettre au point mon application mobile j'ai utilisé le framework d'applications mobiles open source React Native et le principal langage utilisé est JavaScript. La base de données a été réalisée via Firebase. J'ai choisi de créer une base de données firestore cloud, plus simple d'utilisation. Ce service de base de données NoSQL fourni par Google est conçu pour stocker et synchroniser des données en temps réel. J'ai créé deux collections : les utilisateurs et les casiers, comme présenté ci-dessous.



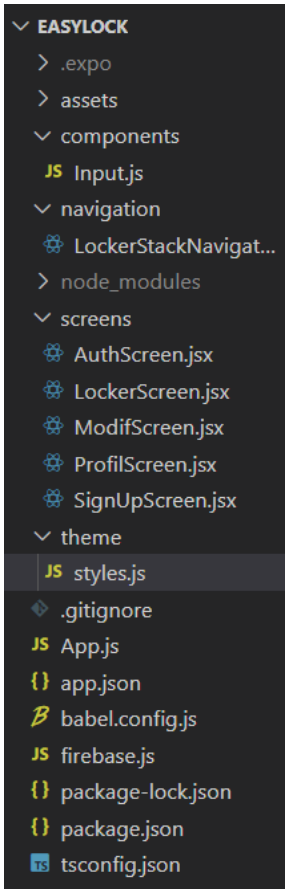
Un casier est défini par un identifiant unique, sa disponibilité (booléen), ses dimensions (hauteur, largeur et profondeur) ainsi que par un champ utilisateur, vide lorsque le casier est libre, ou contenant l'identifiant de l'utilisateur étant en train d'occuper le casier.



Un utilisateur est défini par un identifiant unique généré automatiquement par la base de données, un e-mail, un mot de passe, un nom et un prénom, tous ces champs étant des chaînes de caractères. Chaque utilisateur possède également un champ casiers : ce dernier est un tableau composé de plusieurs mappage. À chaque casier utilisé par l'utilisateur, celui-ci est intégré au tableau casiers de l'utilisateur, son identifiant est récupéré ainsi que l'heure à laquelle sa réservation a été initiée puis terminée. Un booléen état permet de signifier si ce casier est toujours en train d'être utilisé ou s' il s'agit d'une ancienne réservation, afin de constituer un historique.

Ma base de donnée firestore cloud est disponible via ce lien : [EasyLock - Cloud Firestore - Console Firebase (google.com)]

3. Architecture de l'application



Mon application est constituée de cinq fichiers écrans et d'un fichier navigation permettant de faire le lien entre ces écrans. L'appel à ce fichier navigation est fait dans le fichier App.js. De cette façon, l'écran d'authentification est celui qui est affiché en premier lieu en ouvrant l'application, les écrans suivants venant ensuite se superposer à celui-ci. Dans le dossier thème on retrouve le fichier styles.js, importé dans chaque fichier écran, qui gère tous les styles de l'application. Le fichier Input.js dans le dossier components permet de gérer les champs de tous les formulaires de l'application.

4. Plannings

Planning prévisionnel :

		Janvier			Février				Mars					Avril			
Intitulé de la tâche	Date de rendu	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1. Mise en place du projet																	
1.1 Réflexion sur le projet																	
1.2 Rédaction du cahier des charges	23/01																
1.3 Mise en place des différents éléments du projet																	
2. Formation																	
2.1 Formation NodeJS																	
2.2 Formation ReactNativ et JS																	
3. Développement																	
3.1 Modélisation des données																	
3.2 Développement de l'application																	
3.3 Mise en ligne et publication																	
4. Rendu et finalisation du projet																	
4.1 Point sur l'avancement	17/03																
4.2 Point avant rendu final	04/04																
4.3 Rendu final	24/04																
4.4 Soutenance finale	25-26/04																

Planning mis à jour en cours de projet :

		Janvier			Février				Mars					Avril			
Intitulé de la tâche	Date de rendu	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1. Mise en place du projet																	
1.1 Réflexion sur le projet																	
1.2 Rédaction du cahier des charges	23/01																
1.3 Mise en place des différents éléments du projet																	
2. Formation																	
2.1 Formation NodeJS																	
2.2 Formation ReactNativ et JS																	
2,3 Formation Firebase																	
3. Développement																	
3.1 Modélisation des données																	
3.2 Développement de l'application																	
4. Rendu et finalisation du projet																	
4.1 Point sur l'avancement	17/03																
4.2 Point avant rendu final	04/04																
4.3 Rendu final	24/04																
4.4 Soutenance finale	25-26/04																

La mise en ligne et publication de l'application a été oubliée entre temps car elle n'était pas réalisable à mon échelle. Ainsi, mis à part la modélisation des données qui a pris plus de temps que prévu à cause de la prise en main plus longue que prévu de firebase, le planning prévisionnel a globalement été respecté.

5. Pistes d'évolution

Initialement, j'aurais souhaité que le premier écran affiché à l'ouverture de l'application, après la connexion, soit une carte interactive de la zone où l'utilisateur se situe, avec les différents lockers disponibles dans les alentours. Il aurait ensuite été possible de sélectionner le locker souhaité puis d'entrer dans ce locker pour réserver le casier souhaité. J'aurai également souhaité que l'affichage du locker ne se fasse pas sous forme de liste mais sous la forme du locker, c'est -à -dire avec des carrés et des rectangles qui représentent la forme du locker en réalité. De façon générale, c'est surtout le front de mon application qui reste à améliorer car je n'ai pas eu le temps de m'y consacrer.

Il y a également un souci à régler avec l'inscription : l'ajout d'un nouvel utilisateur se fait correctement dans la base de données mais la connexion directement après l'inscription ne fonctionne pas.

Enfin, je souhaitais initialement pouvoir permettre aux utilisateurs de réserver un casier à distance en fournissant une heure d'arrivée devant le casier afin d'être sûr que le casier souhaité soit disponible, je n'ai pas pu mettre cette fonctionnalité en place mais elle me paraît toujours intéressante à mettre en place.

6. Bilan

a) Bilan technique

Le tableau ci-dessous présente la liste des fonctionnalités disponibles dans la version finale de mon application :

Code	Description
EF_01	Un utilisateur peut s'inscrire et se connecter à son compte.
EF_02	Un utilisateur connecté peut se déconnecter s'il n'est pas en train d'initier l'utilisation d'un casier.
EF_03	Un utilisateur peut utiliser un casier si il est disponible.

EF_03.1	Un utilisateur peut choisir le casier qu'il souhaite utiliser via une vue des casiers disponibles (liste ou tableau)
EF_03.2	Un utilisateur ayant finalisé sa demande de casier pour une durée définie peut ensuite valider la fermeture de son casier. Ce casier n'est alors plus disponible pour les autres utilisateurs.
EF_03.3	Une fois l'utilisation du casier terminée, l'utilisateur peut mettre fin à l'utilisation de son casier, déverrouiller son casier et récupérer ses affaires.

J'ai supprimé la fonctionnalité permettant à un utilisateur d'utiliser l'application sans se connecter car il était nécessaire pour moi d'enregistrer des informations concernant l'utilisateur qui réserve le casier dans le casier utilisé pour signifier qu'il est indisponible. De plus, n'ayant qu'un seul locker à disposition et non plusieurs à différentes localisations, la possibilité de mettre des localisations différentes en favoris a donc été abandonnée.

Ainsi, dans mon application, il est possible de se créer un compte, de se connecter, de réserver un casier dans la liste des casiers disponibles, d'accéder à son profil pour modifier ses informations personnelles, libérer le casier en cours d'utilisation et voir son historique de casiers réservés.
Deux comptes ont déjà été créés pour entrer dans l'application :

Identifiant	pdupond@gmail.com	jgaillard@gmail.com
Mot de passe	pdupond	jgaillard

b) Bilan personnel

J'ai débuté ce projet en étant très peu à l'aise avec la programmation en règle générale. J'attendais donc de ce projet de solidifier les bases apprises en cours afin de me sentir plus en confiance dans ce domaine. Malgré beaucoup de difficultés rencontrées par manque de compétences et des fonctionnalités oubliées par manque de temps, j'ai appris à coder mon application mobile de bout en bout et cela m'a apporté beaucoup de satisfaction personnelle.