Rapport de stage

Du 4 Janvier au 26 Février

A l'Aéroclub Limoges Bellegarde

81 Avenue de l'Aéroport, 87 100 Limoges

Développement d'une application mobile

Par : Florent Martinot

BTS Services Informatiques aux Organisations

Option : SLAM

Plan du rapport :

L’objet du stage

1. Le sujet

L’aéroclub Limoges Bellegarde recherchait un moyen de simplifier l’accès à l’information et le traitement en interne de la sécurité et de la mécanique. Le président de l’aéroclub avait pour idée de concevoir deux applications mobiles, qui peuvent être installées sur téléphone ou tablette que les pilotes auraient en permanence sur eux. Ces deux applications mobiles auraient chacune un objectif précis.

Le premier objectif est d’apporter aux pilotes un outil numérique contenant l’ensemble des informations permettant de faciliter et d’assurer la sécurité du vol pour le pilote.

Le deuxième objectif est d’apporter aux gestionnaires et formateurs de l’association un outil numérique dédié à la sécurité et aux échanges d’information inhérente à l’utilisation de la flotte d’avions.

2. Le cahier des charges

Un premier cahier des charges a ainsi été établi avec les cibles principales de cette application :

- Les pilotes : pour les pilotes l’idée est d’avoir toutes les informations indispensables à la réalisation d’un vol en sécurité
- Les mécaniciens, pour les mécaniciens c’est d’avoir l’information, le suivi des dysfonctionnements des aéronefs de telle façon à pouvoir intervenir et/ou programmer les réparations résultantes de ces dysfonctionnements
- Les formateurs, pour les formateurs l’outil doit permettre d’indiquer facilement aux mécaniciens les dysfonctionnements identifiés sur la flotte, mais également d’alerter les pilotes (élèves ou pilotes brevetés) de danger particulier sur les aéronefs, et/ou de difficultés particulières identifiées sur

la plateforme aéroportuaire de Limoges et/ou de l'espace aérien autour de Limoges.

Pour la réalisation de cette application je n'ai eu que comme contrainte une charte graphique à respecter qui est la suivante :

Couleur, code Hex:

- Noir « 000000 »,
- Bleu foncé « 3869A2 »,
- Bleu clair « dee8f6 »
- Rouge « FF4000 »
- Gris « 424242 ».

Polices de caractère : « Oswald pour les titres, et Roboto pour les corps de texte.

Après réception du cahier des charges j'ai discuté avec les différents acteurs de l'aéroclub pour savoir qui aimerait voir quoi dans une application mobile et j'ai établi une liste de fonctionnalités qu'il faudrait implémenter dans une première version de l'application dont voici la liste :

Application mobile Aéroclub Limoges Bellegarde

Implémenté dans une 1ère version :

Page d'accueil avec des articles pouvant être rédigés par les administrateurs

Un système de checklist interactives et de MEL (minimum equipment list) sur tous le avions.

Une page pour voir tous les incidents sur les avions présenté sur la forme de tableau avec les incidents qui ont été traités par la mécanique et les incidents qui n'ont pas encore été résolus.

La mécanique doit pouvoir gérer les incidents survenus sur les avions, en les définissants comme « traités » une fois l'intervention terminée.

Lorsqu'un formulaire est envoyé, il est traité par la mécanique ou le chef pilote qui décide de garder ou non cet incident pour stockage dans l'application.

Une page avec les liens utiles à la préparation d'un vol, que ce soit une navigation ou un vol local.

Après analyse des besoins, je me suis rendu compte qu'il serait beaucoup plus simple de réaliser une seule application englobant toutes les fonctionnalités.

A partir de cette liste j'ai pu commencer la création d'une maquette avant de commencer le développement de l'application.

La maquette de l'application

1. Conception

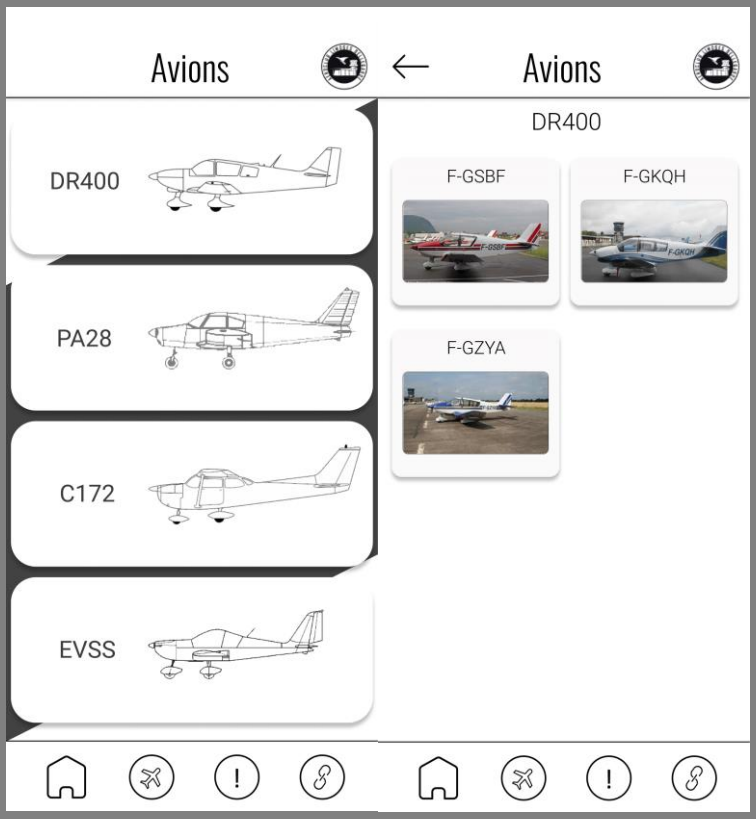
Pour la réalisation de la maquette j'ai utilisé le logiciel « Figma » qui est un logiciel gratuit permettant de concevoir des maquettes pour n'importe quel support numérique.

La principale difficulté à la réalisation de l'interface de l'application était de la rendre accessible pour tous. L'aéroclub a des pilotes de tous âges et une application mobile se doit d'être facile d'utilisation et intuitive pour tous les utilisateurs. J'ai donc dû simplifier un maximum l'interface de l'application et la structurer de telle façon à tout de suite trouver ce que l'on cherche.

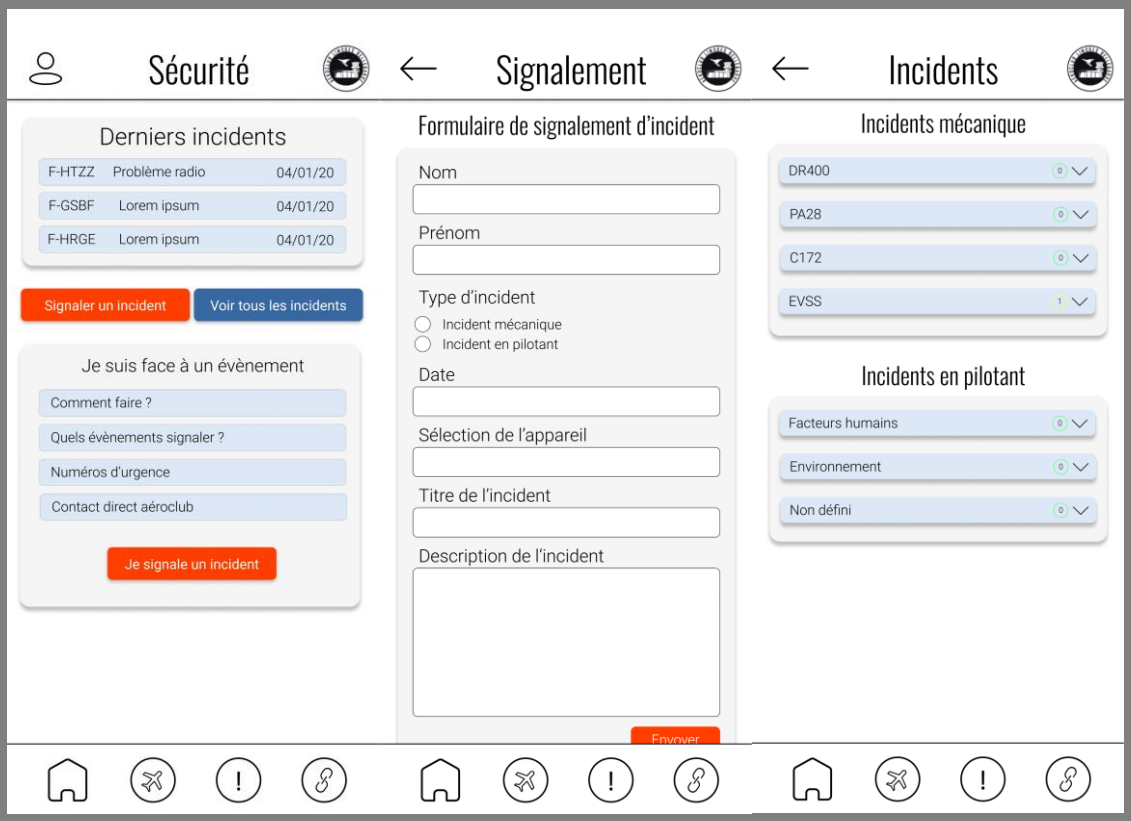
Cette application se compose donc en 4 pages différentes.

Une page d'accueil avec les différents articles qui permettent d'informer les utilisateurs d'évènements dans l'aéroclub ou sur l'aéroport.

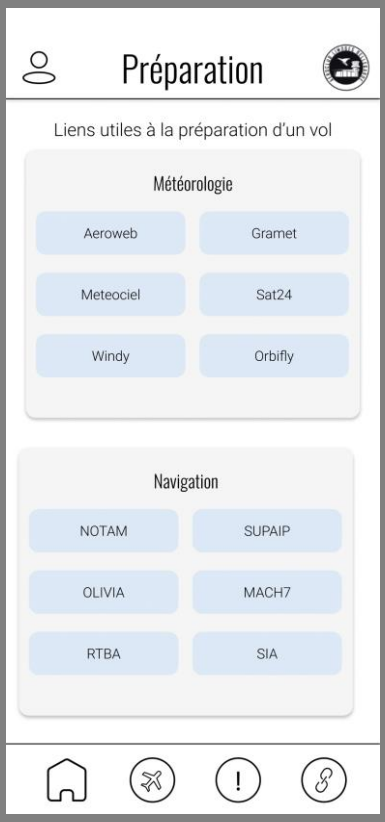
Une page avec la liste de nos avions classé par type d'avions, l'aéroclub possédant plusieurs avions de même catégorie ils sont ensuite classé par immatriculation. Chaque avion a ensuite une page avec tous ses documents.



Une page sécurité répertoriant les derniers évènements survenus sur les avions, donnant accès au formulaire de signalement d'incident et différentes informations utiles aux pilotes



Une page préparation recensant les liens utiles dont peuvent avoir besoin les pilotes.



A l'issue de la réalisation de la maquette, j'ai organisé une réunion afin de la présenter aux différents acteurs de l'aéroclub pour d'éventuelles améliorations. Quelques précisions ont été apportées et la maquette a bien été reçue.

Le développement de l'application

1. Le choix d'une solution

En tant que novice dans le domaine, le premier challenge au développement d'une application mobile est de choisir comment nous allons la développer. Mon objectif étant d'avoir une application cross-platform fonctionnant aussi bien sur IOS que Android, j'avais le choix de développer soit dans le langage natif de ces deux systèmes d'exploitation, mais cela m'aurait obligé d'apprendre deux langages distincts et de développer deux applications, soit de développer une application dite de « single code base » en utilisant un framework. L'avantage de la deuxième solution est que comme son nom l'indique, il n'y a qu'un seul langage et un seul code à écrire pour faire une application à la fois sur android et IOS. Un deuxième avantage est la facilité de mise en œuvre et de maintenance, lorsqu'il y a un changement à faire sur l'application, il n'y a pas besoin de le faire sur les deux plateformes indépendamment. C'est donc vers cette deuxième solution que j'ai décidé de m'orienter.

Mais il me restait encore un choix à faire : le choix du framework avec lequel j'allais développer mon application. Les plus connus sont React Native, Ionic, Xamarin et Flutter. Et c'est vers ce dernier, le petit nouveau (mai 2017) que je me suis orienté. Créé par Google il est encore tout récent dans le monde du développement mobile mais après beaucoup de recherches il est très prometteur, facile de prise en main et possède déjà énormément de fonctionnalités et tutoriels un peu partout sur internet. De plus, il est très facile de connecter une application crée avec Flutter à Firebase qui propose un système de base de données venant aussi de Google et je savais que j'aurai besoin d'une base de données dans mon application afin de stocker les formulaires et les articles. Mon choix est donc fait, il ne me reste plus qu'à commencer à apprendre à développer une application mobile.

2. L’apprentissage

Développer une application mobile n’est pas aussi simple qu’il n’y paraît. Les premières semaines du stage m’ont surtout servi à prendre en main Flutter et son langage propriétaire qu’est le Dart. Heureusement, il y a toujours des réponses à mes questions sur Stack Overflow, et si je ne trouve pas de réponse je peux toujours poser ma question et en moins de 24h mon problème sera solutionné. Il y a aussi beaucoup de vidéos sur Youtube expliquant comment développer une application avec Flutter et abordant pleins de fonctionnalités.

Flutter se compose de Widgets, pour écrire du texte on utilise le Widget Text(), pour faire un bouton on a le choix entre plusieurs Widget… Il existe un Widget pour presque tous les cas de figure.

Construire l’interface de l’application peut prendre un peu de temps quand on a pas vraiment pris en main Flutter, mais une fois le système de Widget maîtrisé ça va très vite, on dispose de deux Widget très pratiques : Column() et Row(), à l’intérieur de ces Widget on peut y placer toute une multitude d’autres Widgets qui seront soit en colonne soit en rangée.

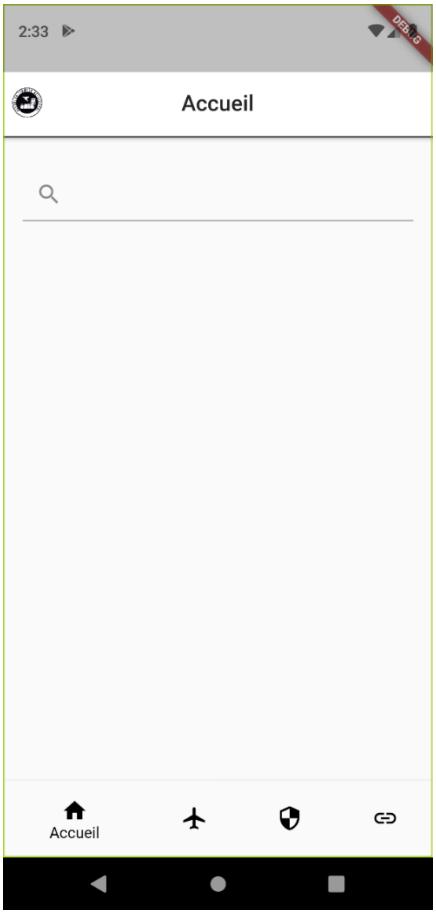
Le plus difficile est en fait de reproduire la maquette, de la retranscrire en code. L’un des gros points fort de Flutter est que lorsque l’on code quelque chose, on peut faire ce qu’on appelle un « hot reload » et en fonction de si l’on utilise un émulateur ou même un vrai téléphone le résultat apparaîtra directement sur l’écran. Lors des premières semaines de développement j’ai utilisé un émulateur de Google Pixel directement intégré dans Android Studio, mais au bout de quelques semaines j’ai utilisé mon ancien téléphone qui est un Samsung Galaxy S7, c’est assez bluffant de voir le résultat de son code en direct sur son téléphone.

3. Les premiers pas

Ayant voulu y aller progressivement, j’ai d’abord commencé par essayer de concevoir le système de menu avec les boutons en bas de l’application qui vont permettre de naviguer vers les différentes pages de l’application.

J’ai réalisé moi-même le design de ces boutons avec un logiciel gratuit « Paint.net » et des logos libres de droit.

Le premier bouton permet d’accéder à l’accueil, le deuxième à la sélection des avions, le troisième d’accéder à la page sécurité et le dernier d’aller sur la page avec les liens utiles aux pilotes.



Voici la première esquisse de l’application depuis l’émulateur de Google Pixel. On y voit les 4 boutons de navigation qui permettent d’accéder aux différentes pages, mais qui n’ont pas encore leurs logos définitifs.

Voici le code associé au menu une fois terminé :

```
class _NavigationState extends State<Navigation> {
  int _selectedIndex = 0;

  List<Widget> _widgetOptions = [
    Accueil(), Avions(), Securite(), Preparation()
  ];

  void _onItemTap(int index) {
    setState(() {
      _selectedIndex = index;
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Center(
        child: _widgetOptions.elementAt(_selectedIndex),
      ),
      bottomNavigationBar: SizedBox(
        height: 70,
        child: BottomNavigationBar(
          type: BottomNavigationBarType.fixed,
          showSelectedLabels: false,
          showUnselectedLabels: false,
          items: [
            BottomNavigationBarItem(
              icon: Image.asset('assets/logos/home.png', height: 40, width: 40),
              label: 'Accueil',
            ),
            BottomNavigationBarItem(
              icon: Image.asset('assets/logos/plane.png', height: 40, width: 40),
              label: 'Avions',
            ),
            BottomNavigationBarItem(
              icon: Image.asset('assets/logos/danger.png', height: 40, width: 40),
              label: 'Sécurité',
            ),
            BottomNavigationBarItem(
              icon: Image.asset('assets/logos/link.png', height: 40, width: 40),
              label: 'Préparation',
            ),
          ],
          currentIndex: _selectedIndex,
          onTap: _onItemTap,
        )
      )
    );
  }
}
```

On peut y voir le système de Widget. Les différents boutons sont dans le Widget BottomNavigationBar() et sons appelés des Items. Chaque bouton est contenu à l'intérieur du Widget BottomNavigationBarItem() et j'y ai associé un logo ainsi qu'un label qui est obligatoire pour que le menu puisse fonctionner. Le label est normalement affiché en dessous de l'icône, mais ayant choisi de garder l'application le plus simple possible, j'ai choisi de le masquer dans ses deux états, lorsque l'on est sur la page associée au bouton et lorsqu'on y est pas.

```
showSelectedLabels: false,
showUnselectedLabels: false,
```

La structure est assez simple et le code est facilement compréhensible par quelqu'un d'initié mais qui n'aurait pas écrit le code.

4. La base de données

Lier une base de données Firebase avec une application Flutter est un jeu d'enfant, nous sommes guidés du début à la fin. Une fois la liaison établie avec un fichier à placer dans l'application, il faut installer le plugin « firebase_core » ainsi que tous les plugins dont nous pouvons avoir besoin, j'ai choisi d'utiliser la fonction du cloud qui s'appelle « cloud_firestore » présente dans firebase pour y stocker les données de mon application, j'ai donc également installé le plugin « cloud_firestore » et le tour est joué, il ne reste plus qu'à écrire des requêtes.

J'ai besoin de faire appel à ma base de données pour l'envois de formulaire de rapport d'incident ou de dysfonctionnement sur les avions, pour traiter ces formulaires en les affichant dans directement dans l'application et pour la page d'accueil avec les articles.

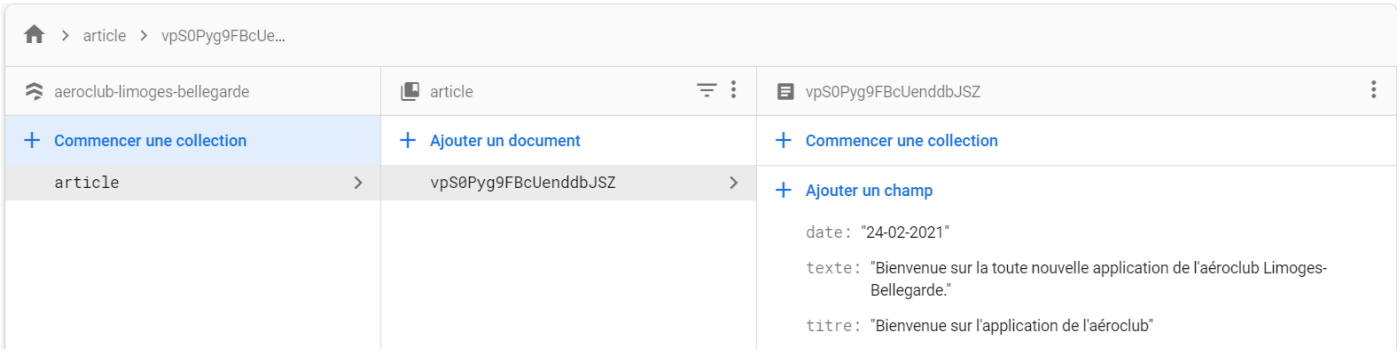
L'envois de données est assez simple, par exemple pour l'envois d'article qui se fait également via un formulaire, on créé une collection comme ceci :

```
CollectionReference article = FirebaseFirestore.instance.collection('article');
```

Et on y ajoute des données comme ceci :

```
await article.add(
 {
  'titre': _titre,
  'texte': _texte,
  'date': now,
 }
);
```

Une collection est une collection de documents qui sont générés avec un id unique à chaque envoi de formulaire. Et à l'intérieur d'un document on y retrouve les données que l'on a stocké, ici le titre, le texte et la date. Voici le rendu dans Firebase :



On y retrouve notre collection d'articles, avec un document qui contient des données que j'ai envoyé à l'aide de l'application.

Voici la requête que j'ai utilisé pour avoir les deux articles les plus récents qui s'affichent sur la page d'accueil :

```
FirebaseFirestore.instance
 .collection('article')
 .orderBy('date', descending: true)
 .limit(2)
 .snapshots(),
```

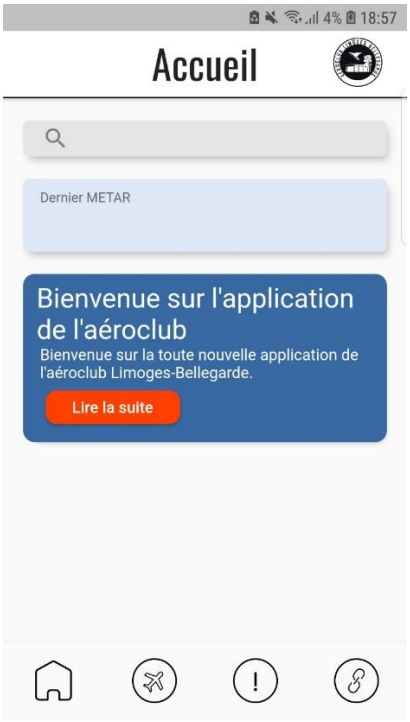
On y voit bien que l'on va chercher les données dans la collection « article », on les classe par date et on en veut que deux.

Et voici comment afficher le contenu de la requête :

```
Align(
 alignment: Alignment.topLeft,
 child: Padding(
  padding: const EdgeInsets.only(left: 12, top: 8.0, right: 12),
  child: Text(document.data()['titre'], style: TextStyle(color: Colors.white, fontSize: 24
)),
 ),
),
Align(
 alignment: Alignment.topLeft,
 child: Padding(
  padding: const EdgeInsets.only(left: 15.0, right: 15.0),
  child: Text(document.data()['texte'], maxLines: 2, style: TextStyle(color: Colors.white)
),
 )
),
```

Document.data()['titre'] permet d'avoir le contenu du champ titre.

Le tout est assez simple d'utilisation et de compréhension et tout fonctionne en temps réel, si je publie un nouvel article il sera directement affiché dans l'application et si je supprime un article il disparaîtra aussitôt.

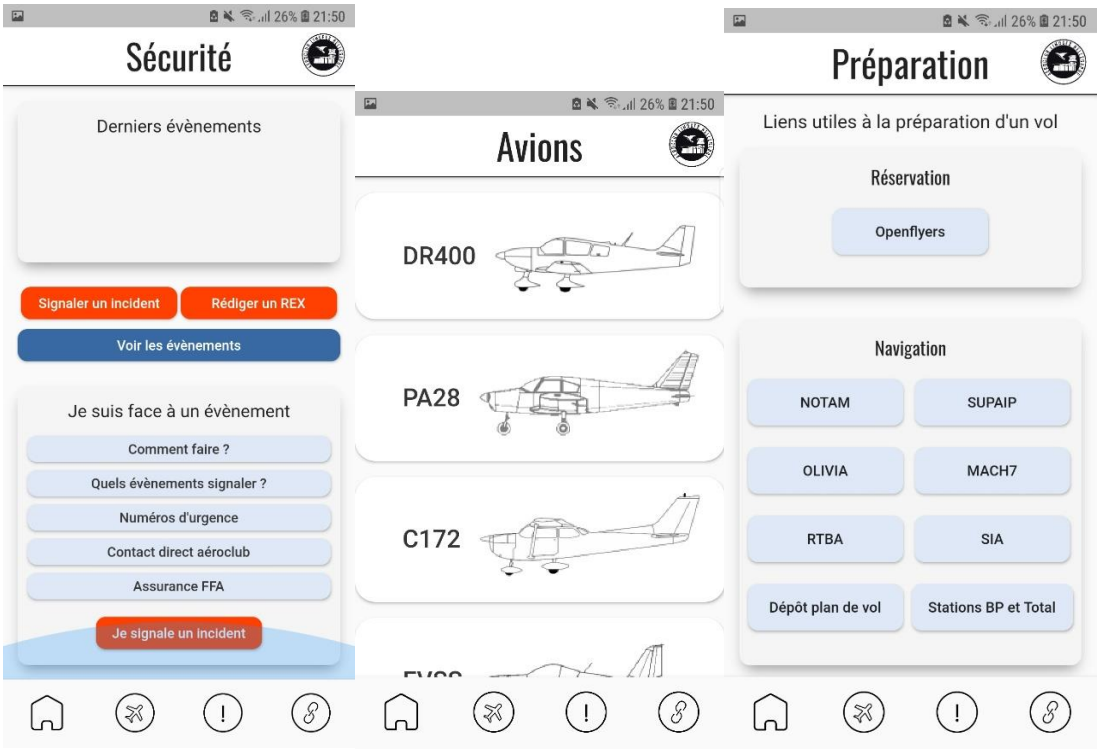


Voici le rendu de l'application avec le dernier article sur la page d'accueil.

On peut y voir le titre « Bienvenue sur l'application de l'aéroclub » ainsi que le texte en plus petit en dessous.

5. Rendu final

Voici sans rentrer dans les détails car il y aurait beaucoup trop de pages à montrer, quelques pages de l'application sur mon téléphone.



Conclusion

Je suis parti de zéro et en deux mois j'ai été capable de réaliser une application mobile plutôt complexe avec beaucoup de fonctionnalités, notamment une partie administration.

Ce stage m'a beaucoup appris et réaliser une application mobile n'est pas aussi facile qu'il n'y paraît. L'avantage que j'ai eu est qu'en tant que pilote au sein de l'aéroclub, je me servirai de cette application, je l'ai donc conçue non pas en tant que développeur, mais en tant qu'utilisateur.

Toutes les fonctionnalités du cahier des charges sont implémentées dans l'application et sont fonctionnelles. L'application a été uniquement testée sur Android par manque de matériel, je n'ai pas de Mac à disposition et je n'ai pas installé d'émulateur MacOs sur mon ordinateur. Il est tout à fait possible de développer une application IOS sur un PC Windows, mais pour la tester et la publier sur l'App Store il faut obligatoirement un Mac.

L'application devrait être publiée sur les différents stores dans les semaines à venir et sera en période de test, l'application est évolutive et je continuerai de travailler dessus pour y apporter de nouvelles fonctionnalités.

Annexe 1

CARTE D'IDENTITE D'ENTREPRISE

Dénomination : Aéroclub Limoges Bellegarde

Siège social : 81 Avenue de l'aéroport, 87100 Limoges

Adresse du lieu de stage : 81 Avenue de l'aéroport, 87100 Limoges

Nationalité : Française

Secteur d'activité : Tertiaire

Objet : L'association a pour objet: De promouvoir, de faciliter et d'organiser la pratique de l'aviation légère et sportive et des différentes activités s'y rattachant, notamment par des opérations de découverte de l'aviation auprès du public et par la formation de pilotes, l'entraînement, le voyage et l'instruction technique nécessaires, tant à l'aide de moyens privés que de moyens d'État, à effet de développer l'aviation générale comme de préparer aux carrières ou métiers y ressortissant; De participer à l'étude, la réalisation et la gestion d'infrastructures aéronautiques: aérodromes, avitaillements, installations techniques et d'accueil

Forme juridique : Privé et associative, Loi 1901

Capital : 700 000€

Effectif : 2

Chiffre d'affaires : 450 000€

Environnement juridique : Autonome