

ECOLE POLYTECHNIQUE DE L'UNIVERSITÉ FRANÇOIS RABELAIS DE TOURS

Département Informatique

64 avenue Jean Portalis

37200 Tours, France

Tél. +33 (0)2 47 36 14 14

polytech.univ-tours.fr

Projet Recherche & Développement

2017-2018

Outil pour la gestion des présences

Rapport de projet recherche et développement



POLYTECH[®]
TOURS

Entreprise

DSI de l'Université François Rabelais



Tuteur entreprise

Malika LABANE (Ingénieure à la DSI)

Étudiant

Kenan SAURET (DI5)

Tuteur académique

Pascal MAKRIS

Liste des intervenants

Entreprise

DSI de l'Université François Rabelais
60, rue du Plat d'Etain - Bâtiment D
37000 - TOURS
www.univ-tours.fr



Nom	Email	Qualité
Kenan SAURET	kenan.sauret@etu.univ-tours.fr	Étudiant DI5
Pascal MAKRIS	pascal.makris@univ-tours.fr	Tuteur académique, Département Informatique
Malika LABANE	malika.labane@univ-tours.fr	Tuteur entreprise, Ingénieure à la DSI



Avertissement

Ce document a été rédigé par Kenan Sauret susnommé l'auteur.

L'entreprise DSI de l'Université François Rabelais est représentée par Malika Labane susnommé le tuteur entreprise.

L'Ecole Polytechnique de l'Université François Rabelais de Tours est représentée par Pascal Makris susnommé le tuteur académique.

Par l'utilisation de ce modèle de document, l'ensemble des intervenants du projet acceptent les conditions définies ci-après.

L'auteur reconnaît assumer l'entière responsabilité du contenu du document ainsi que toutes suites judiciaires qui pourraient en découler du fait du non respect des lois ou des droits d'auteur.

L'auteur atteste que les propos du document sont sincères et assument l'entière responsabilité de la véracité des propos.

L'auteur atteste ne pas s'approprier le travail d'autrui et que le document ne contient aucun plagiat.

L'auteur atteste que le document ne contient aucun propos diffamatoire ou condamnable devant la loi.

L'auteur reconnaît qu'il ne peut diffuser ce document en partie ou en intégralité sous quelque forme que ce soit sans l'accord préalable du tuteur académique et de l'entreprise.

L'auteur autorise l'école polytechnique de l'université François Rabelais de Tours à diffuser tout ou partie de ce document, sous quelque forme que ce soit, y compris après transformation en citant la source. Cette diffusion devra se faire gracieusement et être accompagnée du présent avertissement.



Pour citer ce document

Kenan Sauret, *Outil pour la gestion des présences: Rapport de projet recherche et développement*, Projet Recherche & Développement, Ecole Polytechnique de l'Université François Rabelais de Tours, Tours, France, 2017-2018.

```
@mastersthesis{
  author={Sauret, Kenan},
  title={Outil pour la gestion des présences: Rapport de projet recherche et développe-
    ment},
  type={Projet Recherche \& Développement},
  school={Ecole Polytechnique de l'Université François Rabelais de Tours},
  address={Tours, France},
  year={2017-2018}
}
```

Table des matières

Liste des intervenants	a
Avertissement	b
Pour citer ce document	c
Table des matières	i
Table des figures	iii
1 Introduction	1
1 Acteurs, enjeux et contexte	1
2 Objectifs	1
3 Bases méthodologiques	2
2 Description générale	3
1 Environnement du projet	3
2 Caractéristiques des utilisateurs	3
3 Fonctionnalités du système	3
4 Structure générale du système	4
3 État de l'art	6
1 Applications Android existantes sur le Google Play Store	6
2 EMA : présentation de l'application existante du SUAPS	7
2.1 Fonctionnement	7
2.2 Limitations	8
2.2.1 Utilisation des bases de données	8
2.2.2 Design de l'application	8

2.2.3	Mises à jour et nouveautés.....	8
4	Analyse et conception	9
1	Spécifications fonctionnelles.....	9
1.1	Adaptation de l'application afin de pouvoir faire des pointages dans toute l'université	9
1.2	Mises à jour	9
2	Spécifications non-fonctionnelles	11
2.1	Adaptation du design de l'application	11
2.2	Mise à jour vers le dernier SDK d'Android	11
2.3	Utiliser les codes du material design	11
5	Mise et œuvre	12
1	Mise à jour de la version d'Android	12
1.1	Migration vers le plugin Android Gradle 3.0.0	12
1.2	Mise à jour des fichiers de configuration	12
1.3	Remplacement des méthodes dépréciées.....	13
1.4	Gestion des permissions à l'exécution.....	14
1.5	Tests.....	15
2	Adaptation du design de l'application.....	15
2.1	Le responsive design et le constraint-layout	15
2.2	Swipe entre étudiants	17
3	Modification du système d'affectation.....	17
3.1	Application Android	17
3.1.1	Ajout d'un enseignant	17
3.1.2	Suppression d'un enseignant	18
3.2	Serveur	18
3.3	Problèmes rencontrés	18
4	Documents additionnels produits.....	18
6	Bilan et conclusion	19
1	Première partie	19
2	Deuxième partie	19
	Annexes	21
A	Spécifications fonctionnelles détaillées	22
1	Système d'affectation des professeurs aux appareils.....	22
B	Diagramme de Gant et contenu des lots de livrables	23
	Webographie	24



Table des figures

2 Description générale

1	Diagramme des cas d'utilisations du système existant	4
2	Diagramme de composants du système.....	5

5 Mise et œuvre

1	Schéma de fonctionnement des permissions à l'exécution.....	14
2	Linear layout.....	15
3	Constraint layout.....	16
4	Linear-layout vs. Constraint-layout.	16

A Spécifications fonctionnelles détaillées

1	Algorithme d'affectation	22
---	--------------------------------	----

B Diagramme de Gant et contenu des lots de livrables

1	Diagramme de Gant du projet	23
---	-----------------------------------	----

1

Introduction

1 Acteurs, enjeux et contexte

Dans la plupart des composantes de l'université il est nécessaire de réaliser un suivi des présences des élèves que ce soit à chaque cours pour les étudiants des IUT ou lors des examens à la Fac par exemple. Ce pointage est aujourd'hui réalisé à la main, avec un émargement sur papier qui est ensuite sauvegardé informatiquement par les services de secrétariat.

Il est possible de rendre cela automatique et dématérialisé grâce à différentes applications et notamment une application mobile grâce à laquelle les élèves peuvent pointer sur une tablette ou un smartphone. Cette solution qui existe déjà au SUAPS (Service Universitaire des Activités Physiques et Sportives). En effet, depuis plus de 3 ans maintenant, le suivi des présences lors des cours de sport est fait par pointage de la carte étudiante sur une tablette. Les professeurs ont ensuite accès aux données récoltées via une application web utilisable depuis un ordinateur. Cela simplifie le processus d'appel des professeurs et permet une sauvegarde des données sans risque de perte en vue des évaluations (qui prennent en compte le présentiel pour les cours du SUAPS).

Ce système a fait ses preuves et, bien que des améliorations puissent lui être apporté, il est vu d'un très bon œil par les autres composantes de l'université qui souhaiteraient l'adopter. De plus, la quasi-totalité des enseignants possédant aujourd'hui un smartphone, le déploiement de l'application devrait alors être possible et relativement simple.

Ce projet est proposé par Malika Labane, ingénieure à la DSI (Direction des Systèmes d'Information) de l'Université François Rabelais (UFR) de Tours. Il est encadré par Pascal Makris, enseignant-chercheur à Polytech Tours. Ils représentent la maîtrise d'ouvrage (MOA).

Il est réalisé par Kenan Sauret, étudiant ingénieur en 5^e année à Polytech Tours, maîtrise d'œuvre (MOE).

2 Objectifs

L'objectif du projet est de faire évoluer l'existant afin de le rendre utilisable dans toute l'université. Mon travail se concentrant sur l'application mobile qui permet de réaliser l'émargement.

Le projet sera divisé en deux parties. Premièrement, il sera question de mettre à jour l'application du SUAPS afin d'apporter des nouveautés. Ensuite, il faudra faire les modifications nécessaires afin de pouvoir l'utiliser dans toute l'UFR (Université François Rabelais). De plus, l'application est une application Android qui ne fonctionne actuellement que sur des tablettes de 7 pouces. Il est question de faire fonctionner l'application sur tablette et smartphone Android de toutes dimensions et il est envisagé de développer une application iOS (pour iPhone).

3 Bases méthodologiques

La modélisation du système est réalisée à l'aide du langage UML (Unified Modeling Language). Cela permet d'avoir une modélisation formelle et normée compréhensible par l'ensemble des acteurs de ce projet.

L'ensemble des sources sera versionné en utilisant Git et GitLab. L'IDE utilisé sera IntelliJ. Le projet a été développé avec IntelliJ jusqu'ici et c'est un IDE que j'apprécie.

La méthode SCRUM (méthode agile) sera employée lors du développement afin de produire régulièrement des livrables.

2

Description générale

1 Environnement du projet

Le projet est une partie d'un projet de plus grande envergure. En effet, le système EMA comporte 3 composants : l'application mobile qui permet de faire l'émargement, une application web qui permet aux administrateurs de gérer l'application mobile et une base de données. De plus, le SUAPS dispose d'une application web permettant aux professeurs d'accéder à différentes données et statistiques à propos des présences des élèves.

Le système EMA existant communique avec une base de données propre au SUAPS. L'application pour toute l'université doit communiquer avec les bases universitaires existantes notamment Apogée (base élèves), ADE (base cours) ou encore Harpège (base personnel).

2 Caractéristiques des utilisateurs

Il y a 2 types d'utilisateurs de l'application mobile :

- Les professeurs : ce sont les utilisateurs principaux de l'application. Ils se servent de l'application pour faire l'appel durant leurs cours et faire des mises à jour pour envoyer/recevoir des données sur leurs cours. Ils n'ont pas forcément de connaissance en informatique mais peuvent recevoir une formation sur l'utilisation de l'application.
- Les élèves : il se servent de l'application en pointant leur carte étudiante sur l'appareil. Ils n'ont besoin d'aucune connaissance spéciale.

3 Fonctionnalités du système

La figure 1 représente le diagramme des cas d'utilisation de l'application.

Ainsi comme on peut le voir, un professeur peut réaliser les actions suivantes :

- Se connecter à la tablette avec une carte.
- Se connecter à la tablette avec un code PIN.
- Sélectionner une offre de formation.
- Sélectionner un cours.

- Sélectionner une séance.
- Localiser le lieu d'un cours sur une carte (via Google Maps).
- Voir des informations sur les élèves.
- Afficher une vue « séances » sur laquelle on peut gérer une séance.
- Réaliser un émargement.
- Modifier un émargement.
- Activer le mode « professeur » pour modifier un émargement.
- Synchroniser l'appareil avec le serveur pour l'envoi et la récupération de données.

Un élève peut, lui, badger sa carte sur l'appareil lors d'un émargement.

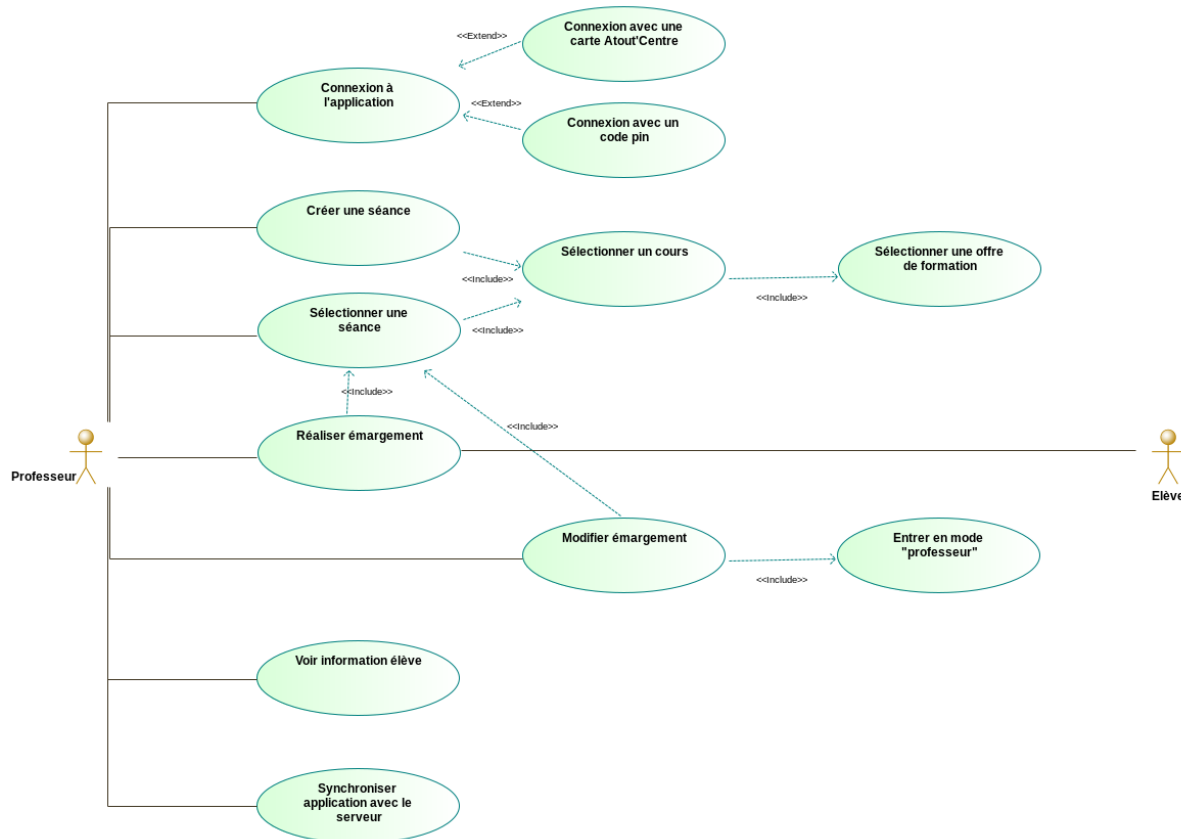


Figure 1 – Diagramme des cas d'utilisations du système existant

4 Structure générale du système

La figure 2 représente le diagramme de composants de l'application.

L'application mobile ainsi que l'application back-office (application web) interagissent avec une base de données EMA. Ces 3 composants forment le système EMA. Cette base de données EMA interagit avec une base de données SUAPS. Cette base existait avant EMA et servait à stocker les informations sur les inscriptions des étudiants de l'université aux cours de sport, informations récupérées en partie dans les bases de données de l'université telle que la base APOGEE (qui contient des informations sur les étudiants).

La différence avec le système existant est que la base de données EMA doit être capable d'interagir directement avec les bases de données universitaires. En effet, il n'est pas envisageable de reproduire la solution du SUAPS (ie. créer une nouvelle base de données) pour chaque

composante de l'université. En effet, toutes les informations nécessaires existent déjà et le coût et la complexité seraient trop élevés.

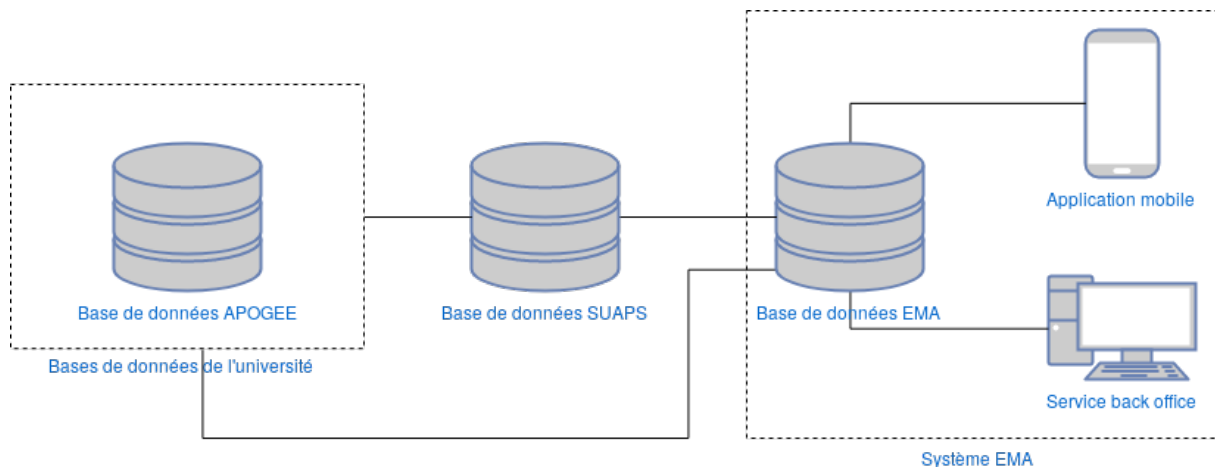


Figure 2 – *Diagramme de composants du système*

3

État de l'art

1 Applications Android existantes sur le Google Play Store

J'ai commencé mon état de l'art en cherchant si des applications similaires existaient déjà. Des applications de gestion de présence existent sur le Google Play Store mais aucune ne possède de mécanisme de pointage par carte. On aurait pu imaginer par exemple qu'une application accepte d'enregistrer des tags NFC afin de les utiliser après mais il n'en est rien.

J'ai testé plusieurs de ces applications afin de voir si on pouvait trouver des mécanismes intéressants à ajouter à l'application EMA. En voici la liste :

- RollCall - Attendance Manager <https://play.google.com/store/apps/details?id=trademila.attendit>
- Attendance Taker <https://play.google.com/store/apps/details?id=com.ferid.app.classroom>
- Attendance <https://play.google.com/store/apps/details?id=com.aor.attendance>
- Attendance Manager <https://play.google.com/store/apps/details?id=dotinc.attendancemanager2>

Les trois premières reposent, à quelques détails près sur les mêmes principes. Elles sont destinées aux professeurs afin de faire de la gestion de la présence des élèves en cours. Pour cela il faut créer manuellement des cours auxquels on ajoute des élèves et des séances de cours. On peut ensuite renseigner, toujours manuellement, la présence ou l'absence d'un élève, voir un autre statut tel que "excusé pour maladie". Une fois qu'un appel a été fait, on peut revenir dessus pour faire des modifications ou alors l'exporter dans différents formats.

Ces applications fonctionnent relativement bien une fois tous les paramètres enregistrés mais ils doivent être ajoutés manuellement et cela prend du temps. Elles proposent un système d'importation via un fichier CSV correctement formaté mais si on prend l'exemple des élèves, ils doivent être importés via ce fichier pour chaque cours. On ne retrouve en fait pas le système de groupe ou de classe français. Ces applications fonctionnent sur le système américain où les élèves choisissent les cours qu'ils veulent suivre.

La quatrième application de la liste est conçue selon une autre vision. Elle est destinée à permettre à un élève de suivre sa présence en cours. C'est lui qui ajoutera sa liste de cours et qui renseignera sa présence ou non à ceux-ci. Il doit aussi ajouter le nombre de séances total

et un objectif (en pourcentage de présence) à chaque cours. L'application lui indiquera alors le nombre de cours qu'il doit suivre pour remplir l'objectif.

Ainsi, bien que plusieurs applications de gestion de présence existent déjà, aucune ne ressemble à EMA. De plus, je n'ai pu trouver, dans les applications lui ressemblant, aucun mécanisme intéressant qu'elle ne possède pas déjà. Enfin, l'application destinée aux élèves me fait penser qu'il pourrait être possible de réaliser une version d'EMA destinée aux élèves. En effet, cette version pourrait contenir l'emploi du temps de chaque élève ainsi que son suivi des présences en cours. De plus, les professeurs pourraient même s'en servir afin de faire passer des informations. Cette application destinée aux élèves ne sera pas intégrée au projet mais je pense qu'il peut être intéressant d'y réfléchir pour le futur.

2 EMA : présentation de l'application existante du SUAPS

2.1 Fonctionnement

C'est une application Android qui fonctionne sur des appareils de 7 pouces minimum et Android 4.4 (KitKat). Elle permet de faire émarger les élèves à l'aide de leur carte étudiante lors des cours de sport.

Voici comment l'application fonctionne à l'heure actuelle :

- A chaque tablette est associé un ou plusieurs professeurs. Cette association est faite manuellement grâce à une application Web.
- Le professeur se connecte à la tablette en badgeant sa carte ou à l'aide d'un code PIN de 4 chiffres propre à l'application EMA.
- Il peut lancer une synchronisation afin de récupérer les informations nécessaires sur les cours qu'il donne.
- Il a alors accès à une liste déroulante qui présente les offres de formation (i.e. S1 Tours, S2 Blois, Stages, Évènements...)
 - On sélectionne une offre et les cours que le professeur dispense sont affichés (Badminton, Handball...)
 - On sélectionne un sport
- Une fois sur le sport plusieurs options sont présentes :
 - On voit la localisation du lieu du cours (gymnase, stade ...) sur une carte Google Maps
 - 2 icônes sont disponibles :
 - Afficher les élèves du cours et ainsi avoir des informations sur eux.
 - Voir les séances. Ici on voit les séances passées (15/09/17 10 :00 ...). On peut ajouter une séance, modifier une séance ou cliquer sur une séance pour débiter un émargement.
- Un émargement se passe de la manière suivante :
 - Le professeur crée une séance (elle est créée à la date et l'heure actuelle)
 - La liste des étudiants s'affiche, ils sont tous en rouge (absents).
 - Chaque étudiant passe un par un et badge sa carte sur la tablette. Sa photo s'affiche en grand pour que le professeur puisse faire une vérification puis il passe en vert sur la tablette pour signifier qu'il est présent. S'il n'est pas dans la liste (carte non reconnue) un son d'erreur retenti.
 - Une fois l'émargement fini on a donc les étudiants présents en vert, les absents en rouge.
 - Le professeur peut badger sa carte pour faire entrer la tablette « en mode professeur » et faire des modifications sur la séance. Ainsi il peut changer l'heure et la date de

- la séance, changer le statut des étudiants (passer un absent excusé en jaune par exemple).
- Une fonction existe pour mettre tous les élèves présents quand la séance n'a pas lieu (professeur malade par exemple). Dans ce cas les élèves passent en vert clair.
- Une synchronisation via Wifi s'effectue pour envoyer les informations.

2.2 Limitations

L'application souffre de plusieurs limitations. C'est le but de mon projet de les résoudre afin de rendre l'application plus performante et surtout qu'elle soit utilisable dans toute l'université.

2.2.1 Utilisation des bases de données

Il n'est pas possible d'accéder aux bases de données de l'université (cours, étudiants, professeurs, etc.) librement, notamment en modification. Ainsi, pour simplifier la tâche, il a été décidé de créer une base de données propre au SUAPS dans laquelle sont stockées les informations sur les inscriptions et les étudiants inscrits aux cours de sport. Les inscriptions au SUAPS se faisant via une application web sur internet depuis un certain temps déjà, cette base existait avant EMA.

C'est avec cette base SUAPS que la base de données EMA dialogue. Ainsi, l'application EMA est libre d'effectuer des modifications dans sa propre base qui seront transmises à la base SUAPS sans influencer les bases universitaires. Ce schéma fonctionne bien mais il est impossible de le garder si l'on veut étendre l'application à toute l'UFR. En effet, les informations nécessaires existant déjà dans les bases universitaires, il n'est ni utile, ni judicieux (coût et complexité de gestion) d'en créer de nouvelles. Il y aurait une duplication des informations.

2.2.2 Design de l'application

A l'heure actuelle, l'application ne fonctionne que sur des écrans supérieurs à 7 pouces. En effet, le design possède certaines parties qui sont fixes et calibrées pour un écran 7 pouces (la taille de l'écran faisait partie des contraintes lors du début du projet). Des fonctionnalités sont alors indisponibles si on utilise l'application sur un écran de plus petite taille.

2.2.3 Mises à jour et nouveautés

L'application a été développée il y a 3 ans. Android en était à sa version 4.4 (Kitkat). Depuis Android en est à sa version 8 (Oreo) et la version 8.1 sera bientôt disponible. Cette évolution s'est accompagnée de mises à jour et de nouveautés qu'il est intéressant d'intégrer à l'application. D'autre part, plusieurs améliorations de l'application ont été demandées par le SUAPS. Il faut répondre à ces demandes.

4

Analyse et conception

1 Spécifications fonctionnelles

Les spécifications fonctionnelles du projet se séparent en deux catégories : les changements à apporter pour pouvoir utiliser l'application dans toute l'université et les mises à jour et ajouts demandés par le SUAPS.

1.1 Adaptation de l'application afin de pouvoir faire des pointages dans toute l'université

Il y a deux modifications à apporter pour pouvoir utiliser l'application dans toute l'université. Premièrement, l'application doit interagir avec les bases de données universitaires et plus avec la base du SUAPS. La base EMA requiert un certain nombre d'informations pour fonctionner. Il faut récupérer ces informations dans les différentes bases de données de l'université. Pour cela, des vues seront développées. On utilisera alors ces vues pour acquérir les informations nécessaires. A priori, aucune modification de la base EMA ne sera à effectuer. Son fonctionnement est bon, les données existent dans les bases universitaires et elle possède un champ permettant d'identifier la provenance de l'information.

Deuxièmement, il faut revoir le système d'affectation des appareils aux professeurs. En effet, le système EMA possède un mécanisme d'association professeur-appareil afin de ne charger que les données des professeurs associés. Il doit être modifié. Actuellement il est réalisé manuellement et on souhaite l'automatiser. L'algorithme que doit suivre le mécanisme d'association est donné en annexe 1 (Annexe A). De plus, il faut supprimer un autre mécanisme. C'est celui d'autorisation de l'appareil à utiliser EMA. Cela simplifiera le déploiement dans toute l'université. En effet, tout le monde pourra installer l'application Android mais une authentification est nécessaire pour l'utiliser, garantissant la sécurité.

1.2 Mises à jour

8 mises à jour et ajouts m'ont été demandés de faire :

- **Signaler à l'utilisateur la connexion/non-connexion à internet** : lorsque la connexion à internet est requise, il faut afficher une pop-up pour signaler à l'utilisateur s'il n'est pas connecté. De plus, il faut ajouter à l'application une icône visible sur toutes les pages qui montre la connexion ou non au réseau. La vérification de la connexion au réseau doit aussi être faite lors des tests au démarrage.
- **Signaler à l'utilisateur l'activation/la non-activation du NFC** : il faut signaler à l'utilisateur via une pop-up la non-activation du NFC lorsqu'il démarre un émargement. De plus, il faut ajouter à l'application une icône visible sur toutes les pages qui montre l'activation ou non du NFC. La vérification de l'activation du NFC doit aussi être faite lors des tests au démarrage.
- **Signaler à l'utilisateur que l'appareil n'a pas été synchronisé récemment** : il n'y a actuellement pas de mécanisme de vérification de synchronisation de l'appareil. On souhaite ajouter un moyen de signaler à l'utilisateur la date de la dernière synchronisation lorsqu'il crée une séance afin de faire un émargement. Pour cela, la date de dernière synchronisation sera affichée à l'écran et une pop-up proposant de synchroniser sera affichée si la date de dernière synchronisation est vieille de plus d'un jour.
- **Rendre la synchronisation bloquante et afficher son résultat** : la synchronisation doit être bloquante. Ainsi, lors d'une synchronisation, il ne doit plus être possible d'interagir avec l'écran et une animation doit s'afficher au centre. A la fin de celle-ci, une pop-up doit afficher le résultat (réussite ou échec).
- **Créer un tutoriel et une aide** : les utilisateurs de l'application peuvent être novices. Ainsi, un tutoriel doit être proposé. Il doit être possible de le regarder autant de fois que voulu. Une aide doit aussi être rédigée. Elle décrira l'utilité et le fonctionnement de chaque fonction de l'application.
- **Ajouter une gestion par swipe de l'application** : une navigation par swipe vers la gauche ou la droite doit être possible entre les écrans donnant les informations sur un élève.
- **Afficher les présences ultérieures des élèves** : sur la page «séance», permettant de réaliser un émargement, les présences ultérieures des élèves du cours pendant ce semestre doivent être affichées à l'aide de carrés de couleur. Les couleurs sont : vert = présent, vert clair = présence validée par le professeur car cours annulé, jaune = absence justifiée, rouge = absence injustifiée.
- **Corriger différents bugs** : Voici une liste de bugs recensés dans l'application et à corriger :
 1. Problème de synchronisation : actuellement, lors d'une synchronisation, si un problème se passe (pas de connexion internet par exemple), l'utilisateur n'en est pas informé. Ainsi, il peut potentiellement effectuer des modifications sur des données non mises à jour. Il faut corriger ce problème en informant l'utilisateur du succès ou de l'échec de la mise à jour.
 2. Plantage de l'application lors d'un problème avec le lieu du cours : un plantage de l'application peut survenir s'il y a un problème avec le lieu du cours. Ce problème devrait être résolu en utilisant les API les plus récentes de Google Maps.
 3. L'écran du code pin ne disparaît pas : lorsque l'on veut passer la tablette en mode professeur, un écran permettant d'entrer le code pin s'affiche. Si le professeur active ce mode en badgeant sa carte, l'écran demandant le code pin ne disparaît pas alors que la tablette est bien passée en mode professeur.

2 Spécifications non-fonctionnelles

2.1 Adaptation du design de l'application

L'application actuelle a été développée pour fonctionner sur une tablette 7 pouces. On souhaite utiliser le responsive design afin qu'elle puisse être utilisée sur tous les appareils Android. Le responsive design est un principe de conception selon lequel l'affichage s'adapte à la taille de l'écran pour que l'utilisateur puisse utiliser simplement toutes les fonctions de l'application. Ainsi, il faut, en fonction de la taille de l'écran, afficher les éléments de différentes façons. Cela peut se traduire par un agencement différent, par le fait de cacher certains éléments et/ou par l'utilisation de conteneurs de tailles variables (par exemple on peut utiliser des menus déroulants, qui sont de base ouverts ou fermés en fonction de l'espace dont on dispose).

Les choix de conceptions doivent être réalisés pour donner à l'utilisateur la meilleure expérience possible et surtout la plus intuitive.

Ce point est crucial dans le développement car il permet une utilisation de l'application sur tous les supports. Ainsi, le déploiement de celle-ci est fortement facilité puisque tout le monde peut l'installer sur sa tablette ou son smartphone Android.

2.2 Mise à jour vers le dernier SDK d'Android

Aujourd'hui l'application fonctionne sous Android 4.4 (Kitkat). Bien qu'avec Android il n'y ait pas de problème pour faire fonctionner une application développée pour une ancienne version sur un appareil utilisant une version plus récente, il est important de faire cette mise à jour. En effet, l'inverse est aussi possible, c'est-à-dire de faire fonctionner une application récente sur un système plus ancien mais avoir une application à jour apporte plusieurs options intéressantes. Tout d'abord une application à jour est plus sécurisée, ensuite une application à jour peut profiter de nouvelles fonctionnalités et enfin une application à jour est souvent plus optimisée (plus rapide). C'est pourquoi il est voulu de mettre l'application dans la dernière version d'Android, Android 8.1 (Oreo).

2.3 Utiliser les codes du material design

Une application qui respecte le material design est une application dont les utilisateurs arriveront à se servir facilement. En effet, les codes du material design sont maintenant utilisés dans la plupart des applications, les utilisateurs y sont habitués.

5

Mise et œuvre

1 Mise à jour de la version d'Android

La première partie de mon travail fut de mettre à jour le code de l'application afin d'utiliser le dernier SDK d'Android, le SDK 27.

1.1 Migration vers le plugin Android Gradle 3.0.0

En décembre 2017 est sorti le plugin Gradle 3.0.0 pour Android. Celui représente une amélioration majeure des outils de build et de débogage. Pour mettre à jour mon projet et l'utiliser, j'ai suivi le guide[[WWW2](#)] fourni par Android sur son site développeur. Cependant, après avoir attendu que IntelliJ télécharge les API nécessaires et résolve les nouvelles dépendances une erreur persistait. C'était un bug connu, résolu rapidement par une mise à jour de l'IDE (source StackOverflow[[WWW1](#)]).

Les fichiers à modifier sont :

- `/gradle/gradle-wrapper.properties` pour indiquer l'URL correcte pour télécharger la version de Gradle compatible.
- Les deux fichiers `/build.gradle` et `/EmargMobileApp/build.gradle` pour indiquer la version de Gradle à utiliser (généralement la plus récente disponible).

1.2 Mise à jour des fichiers de configuration

Une fois les outils de développement les plus récents intégrés à l'application, j'ai mis à jour les fichiers de configuration afin d'indiquer au projet d'utiliser la dernière version d'Android, c'est-à-dire le SDK 27 (correspondant à Android Oreo 8.1).

J'ai d'abord modifié le numéro de version. Celui-ci permet d'identifier les versions successives de l'application. Il sert notamment lors de l'installation de l'application sur un appareil. Ce dernier ne permet pas d'installer une version plus ancienne que celle déjà présente, évitant ainsi des problèmes de cohérence sur les fonctionnalités. Pour changer ce numéro, il faut modifier la balise `<manifest>` du fichier `AndroidManifest.xml`.

Le reste des modifications s'effectuent dans le fichier `/EmargMobileApp/build.gradle`.

- On indique que l'on veut compiler dans la version 27 et utiliser les outils de compilation qui correspondent à cette version.
- On passe la version minimum de 19 à 21. Il n'existe quasiment plus d'appareils Android qui fonctionnent dans une version inférieure et cela permet certaines simplifications dans le code.
- Dans la section **dependencies** on s'aide de l'IDE pour mettre à jour les versions des librairies qu'on utilise. De plus, Android ne permet plus d'utiliser la bibliothèque **play-services** directement. En effet, au fil des versions, celle-ci s'est trop enrichie et contient maintenant des douzaines de librairies. L'importer augmenterait considérablement le poids de l'application. Il faut maintenant importer chaque librairie que l'on utilise individuellement. On trouve en ligne un guide[[WWW6](#)] permettant de connaître les API à importer.

Une fois que ces modifications ont été effectuées, l'application est prête pour la version 27. Cependant, de nombreuses méthodes ont été dépréciées au fil des versions et doivent donc être remplacées.

1.3 Remplacement des méthodes dépréciées

Il allait de paire avec la mise à jour de la version qu'il fallait remplacer les méthodes dépréciées dans le code par leur nouvelles versions fonctionnelles. Le saut de version assez important (de 19 à 27) a apporté un nombre considérable de dépréciation. J'avais sous-estimé le temps que cela me prendrait et j'ai perdu du temps par rapport à mon planning prévisionnel dans cette partie.

Tout d'abord, j'ai remplacé les méthodes étant devenues non fonctionnelles. En effet, lorsqu'on indique que la version minimale supportée est la version 21 (contre 19 auparavant), certaines méthodes ne sont plus supportées et deviennent des erreurs. Pour cela, je me suis aidé principalement de la documentation Android ainsi que du site StackOverflow sur lequel la communauté Android est très active et où la plupart des problèmes que j'ai pu rencontrer étaient résolus.

Par la suite, j'ai parcouru la totalité des classes du projet et remplacé une à une les méthodes indiquées comme dépréciées. Si ce travail fût un peu long et fastidieux, je n'ai cependant rencontré que peu de problèmes sur cette partie. Encore une fois, la communauté Android étant très active, la plupart des modifications à faire se trouvent aisément sur internet. Et ce, même pour les changements récents apportés par la version 27 (sortie en octobre 2017) car la documentation est correctement remplie.

L'une des modifications qui a prit le plus de temps est la gestion de la géolocalisation. En effet, entre les versions 19 et 27, le système a quasiment complètement changé 2 fois. Ainsi, j'ai d'abord suivi un tutoriel[[WWW3](#)] pour passer de la première version (qui ne fonctionnait plus) vers une version dépréciée, puis un second tutoriel[[WWW5](#)] pour arriver à la méthode la plus à jour. La dernière version permet de localiser l'appareil de façon totalement asynchrone (donc non-bloquante) et d'utiliser au choix une localisation très précise mais qui consomme un peu plus de batterie ou une localisation moins précise mais moins gourmande.

Enfin j'ai laissé certaines méthodes dépréciées dans le code. Ce sont des méthodes pour faire des requêtes HTTP. Dans le code existant, une librairie Apache est utilisée pour faire cela car à l'époque il n'existait pas de librairie Android pour le faire. Cependant, une librairie existe depuis donc la librairie Apache est déprécié. En revanche, il existe une librairie Apache Http-legacy qui permet de pouvoir encore utiliser les anciennes méthodes. J'ai choisi de ne pas changer cette partie de code car cela m'aurait pris du temps (il faut complètement changer de librairie donc il faut beaucoup de modifications dans le code) et que la librairie Http-legacy existe (Apache continue donc à soutenir son ancienne API). J'ai estimé avoir déjà pris assez de retard et j'ai décidé de passer au développement de la fonctionnalité suivante.

1.4 Gestion des permissions à l'exécution

Une des grosses nouveautés apporté par Android Marshmallow (version 23) est la gestion des permissions à l'exécution. Afin de donner plus d'informations et plus de liberté à l'utilisateur sur ce à quoi peut accéder une application, la méthode d'acceptation des permissions à changé. Auparavant, c'était lors de l'installation que l'utilisateur autorisait toutes les permissions nécessaires à l'application pour son fonctionnement (il ne pouvait pas l'installer sinon). Maintenant, une liste de permissions dites dangereuses[WWW4] doivent être autorisés par l'utilisateur à l'exécution de l'application. Cela permet de donner plus d'information à l'utilisateur en affichant pourquoi cette permission est nécessaire au fonctionnement. Il peut alors choisir de l'accepter ou pas, il faut désactiver la ou les fonctionnalités correspondantes dans le cas négatif. Le fonctionnement de ce système suit le schéma de la figure 1.

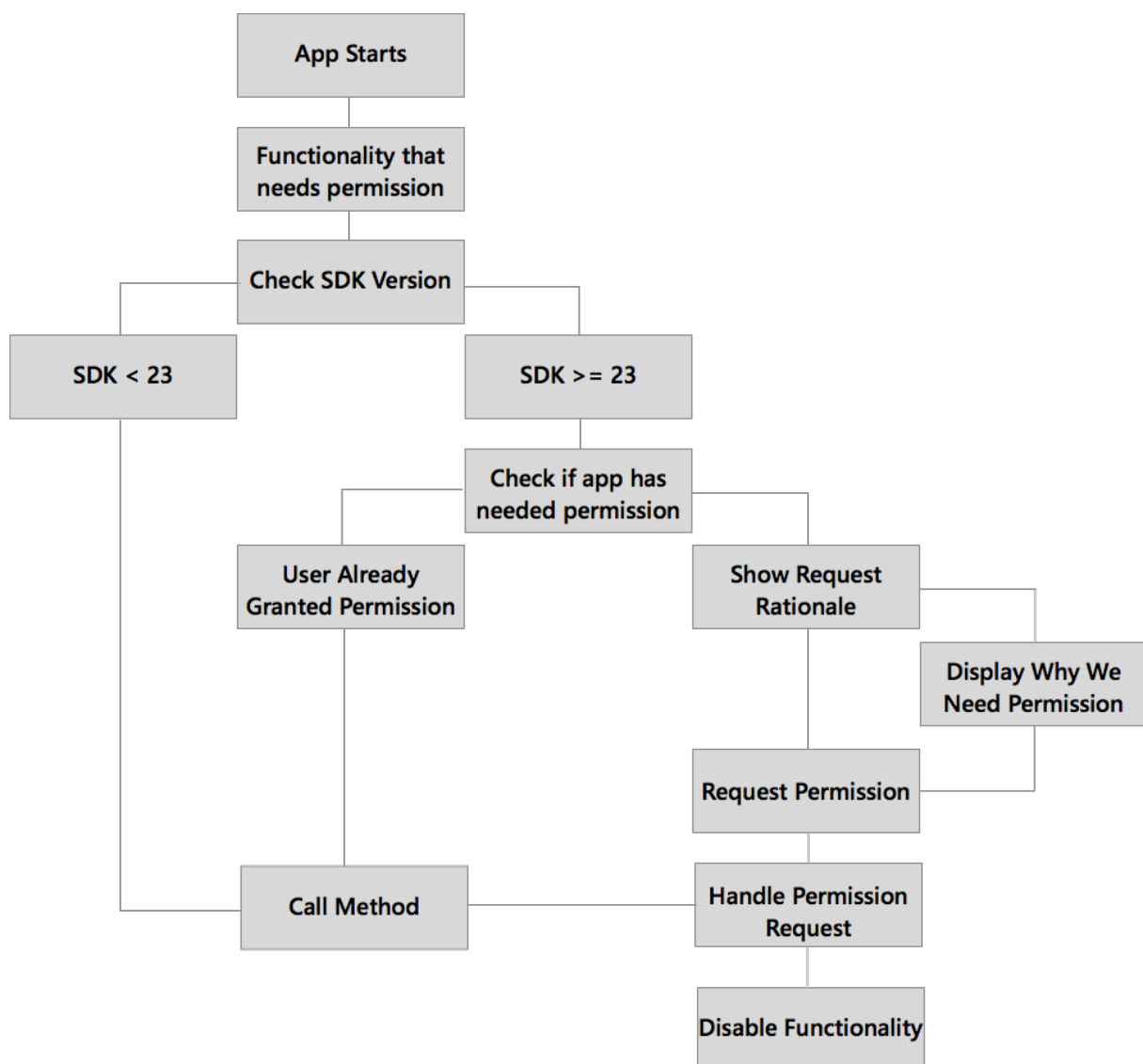


Figure 1 – Schéma de fonctionnement des permissions à l'exécution.

Ce nouveau comportement amène plusieurs modifications dans le code. Tout d'abord, il faut rendre les fonctionnalités concernées dépendantes des permissions. Ainsi, avant de s'exécuter, elles doivent pouvoir vérifier si les permissions sont accordées. Pour certaines fonctionnalités, il

faut aussi pouvoir les désactiver si les permissions nécessaires ne sont pas données. Dans cette application, 3 permissions dangereuses sont requises :

- Accéder à la position de l'appareil. Cette permission est obligatoire et l'application ne fonctionne pas si elle n'est pas donnée. En effet, la localisation de l'appareil sert notamment lors de la première utilisation afin d'enregistrer l'appareil en base de données.
- Accéder à l'état du téléphone. Cette permission est obligatoire et l'application ne fonctionne pas si elle n'est pas donnée. Elle permet de récupérer un identifiant unique pour chaque appareil et ainsi les identifier.
- Accéder au stockage du téléphone. Cette permission n'est pas obligatoire. Elle permet de stocker des informations de log pour les développeurs sur l'appareil.

De plus, les demandes faites à l'utilisateur ainsi que la gestion de sa réponse doivent être faites de manière asynchrone.

1.5 Tests

Pour tester les modifications que j'ai apporté au code existant, et surtout lors du remplacement des méthodes dépréciées, j'effectuais la manipulation suivante. J'observais le comportement de la fonction atteinte par la modification en l'utilisant sur mon appareil, je faisais les modifications, puis je reproduisais l'utilisation pour vérifier que le comportement n'avait pas changé. Si je conçois que cette méthode de test n'était pas très conventionnelle elle s'est révélée efficace. Cependant, si j'avais à refaire la chose, j'essaierai de faire des tests plus rigoureux.

2 Adaptation du design de l'application

La deuxième partie du développement fut de changer le design de l'application afin quelle puisse être utilisée sur des appareils ayant une taille d'écran différente.

2.1 Le responsive design et le constraint-layout

Un des problèmes majeurs de l'application était que certaines fonctionnalités n'étaient pas accessibles si l'écran de l'appareil était trop petit. Dans le code, des **linear-layout** (figure 2) étaient principalement utilisés. Il affichent à l'écran les éléments les uns sous les autres mais ne prennent pas en compte la largeur de l'écran.

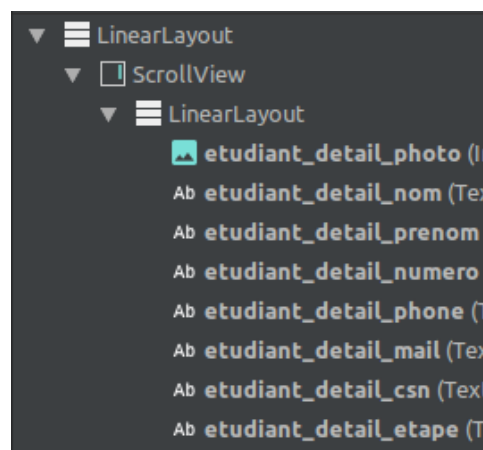


Figure 2 – Linear layout.

Cependant, il existe en Android un outil de design qui s'appelle le **constraint-layout** (figure 3). Celui-ci permet de lier les éléments les uns par rapport aux autres (par exemple spécifier que ce champ texte doit se trouver 8 pixels à droite de cette photo) mais aussi par rapport aux bords de l'écran. Mais la force de cet outil ne s'arrête pas là. Dépendamment des valeurs de contraintes fixées et des dimensions de l'écran, il ajuste le design de façon autonome (par exemple placer le champ texte sous la photo et non à droite car il serait coupé par le bord de l'écran sinon). Cela permet d'avoir un design responsive. C'est-à-dire que le design de l'application dépend de la taille de l'écran sur lequel il s'affiche.

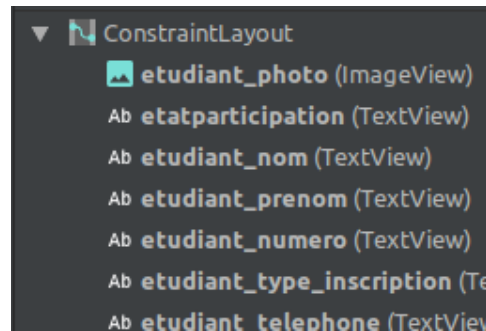


Figure 3 – Constraint layout.

Pour faire ce changement, j'ai modifié les fichiers de design afin qu'ils utilisent le constraint-layout. J'ai ensuite fixé les différentes contraintes afin que l'affichage corresponde à ce qui était proposé avant, tout en réglant les problèmes observés auparavant. La figure 4 illustre un de ses changements, les boutons qui sortaient de l'écran sont maintenant accessibles.

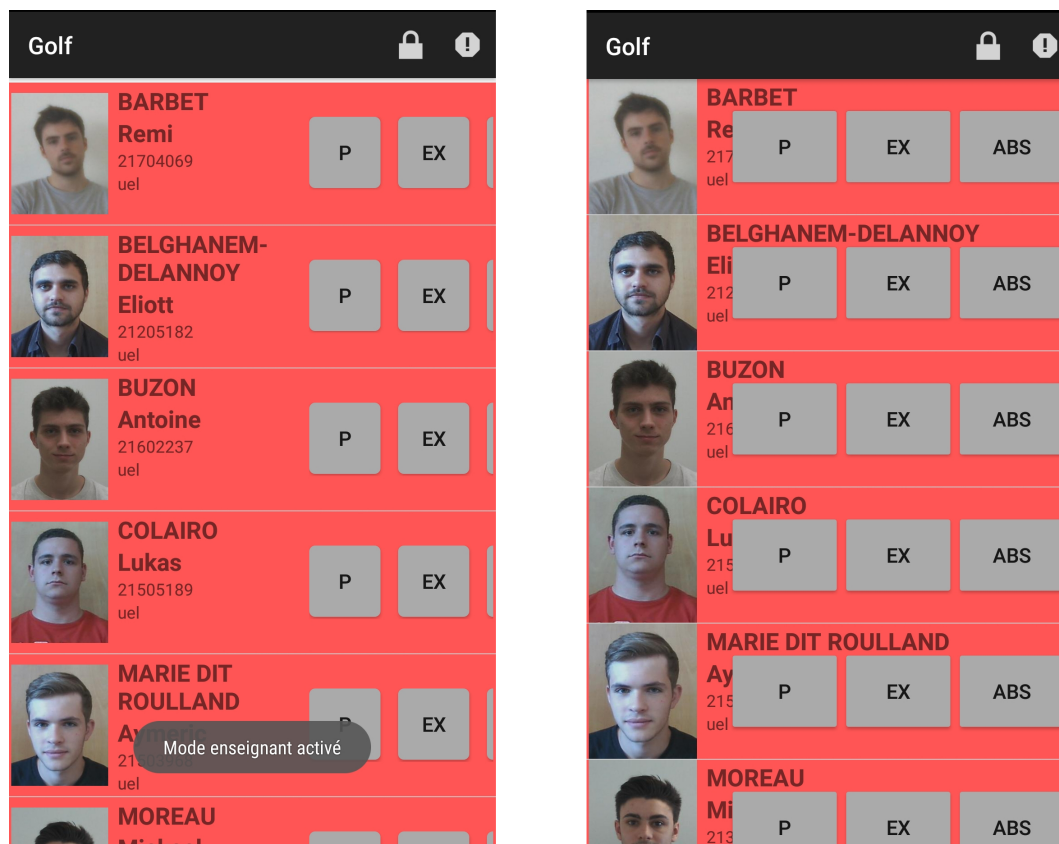


Figure 4 – Linear-layout vs. Constraint-layout.

J'ai aussi fait le choix, lorsque des boutons doivent s'afficher, de les afficher par dessus l'élément auxquels ils correspondent (voir figure 4). Ce choix me paraît judicieux car l'utilisateur sait sur

quoi il agit.

2.2 Swipe entre étudiants

Une autre requête d'évolution dans le design de l'application était de permettre une navigation par swipe (glissé du doigt) à droite ou à gauche entre les étudiants d'une liste. J'ai pu ajouter cette fonctionnalité au fruit de quelques modifications.

En Android, la brique de base de ce qui est affiché à l'écran s'appelle une **activité**. Dans le code existant, lorsqu'on affichait les informations d'un étudiant à l'écran, une activité complète était affichée. Afin de permettre le swipe, il faut transformer ces activités en **fragment**. Ces derniers sont des éléments de design qui s'attachent au dessus d'une activité. Ensuite il faut créer une activité vide qui servira de socle aux fragments. Il suffit alors d'ajouter dans le code un élément qui permet d'interchanger les fragments sur l'activité socle lorsque l'utilisateur glisse son doigt sur l'écran. Cet élément existe et il est simple d'utilisation, il s'appelle **FragmentManager**.

3 Modification du système d'affectation

Le troisième point fut la modification du système d'affectation enseignant/appareil. Ce point a nécessité la modification à la fois du code de l'application Android et du serveur. Pour rappel, il est nécessaire de rendre automatique le système d'affectation et d'autoriser par défaut l'application sur l'appareil pour utiliser l'application dans toute l'université, voir 1.1 (Chapitre 4).

3.1 Application Android

Pour commencer j'ai créé 2 activités que j'ai ajouté à l'écran d'accueil. L'une permet d'ajouter un enseignant à l'appareil, l'autre d'en supprimer un. J'ai aussi ajouté deux boutons à l'écran d'accueil pour lancer ces activités.

3.1.1 Ajout d'un enseignant

Le fonctionnement se veut extrêmement simple. L'utilisateur n'a qu'à passer la carte de l'enseignant à ajouter pour qu'il le soit. Il faut ensuite synchroniser l'application pour ajouter l'enseignant en base de données et récupérer les données qui lui correspondent.

Plusieurs choses sont à noter. Premièrement, l'ajout ne peut se faire que par la carte Atout'Centre de l'enseignant. Contrairement à la connexion à l'application qui peut se faire avec un couple login/code pin. Cela permet de garder quasiment le même niveau de sécurité que l'on avait avec l'ajout manuel par un administrateur (fonctionnalité par ailleurs toujours utilisable). En effet, n'importe qui peut télécharger et installer l'application mais si on ne possède pas de carte autorisée (c'est-à-dire celles des enseignants), il est impossible d'accéder à l'application. De plus, si un enseignant perd ou se fait voler sa carte, il va en demander une autre ce qui désactivera l'ancienne. J'ai discuté de cela avec mon client et nous avons décidé de cette solution ensemble.

Deuxièmement, il faut penser à faire deux synchronisations. Une après l'ajout pour s'assurer que cet ajout de nouvelle association est correctement stocké en base de données et une pour s'assurer de la récupération des données.

3.1.2 Suppression d'un enseignant

De même que pour l'ajout le processus est très simple. La page de suppression présente une liste des enseignants associés à l'appareil. Il suffit d'appuyer sur l'enseignant à supprimer puis sur une confirmation et enfin de synchroniser l'appareil pour que l'association soit supprimée.

3.2 Serveur

Quelques modifications doivent aussi être faites côté serveur. Tout d'abord j'ai changé l'autorisation de base lorsqu'un appareil est ajouté en base de données. Comme dit dans les spécifications, on souhaite que par défaut n'importe quel appareil puisse utiliser l'application afin de rendre le déploiement dans l'université plus simple. Pour cela, il a suffi de changer un booléen dans le code du serveur. On garde cependant le moyen de supprimer cette autorisation en cas de problème.

Par la suite j'ai cherché à modifier le code de façon à enregistrer les modifications d'association apportées via l'application Android. Cependant, j'ai eu du mal à comprendre le code du serveur. Je n'avais jamais vu de code de serveur Glassfish et je ne pouvais pas déployer le serveur localement et faire des tests avec l'application puisque le fonctionnement repose sur des web-services (une connexion internet est obligatoire). Ceci fait que je n'ai pas pu finir le développement de cette fonctionnalité. Cependant, je propose dans la section suivante des pistes pour y arriver.

3.3 Problèmes rencontrés

Le gros problème que j'ai rencontré dans cette partie est la communication entre application et serveur. En effet, comme dit plus haut, il m'était impossible de déployer le serveur pour effectuer des tests. Je n'ai pas non plus réussi à simuler une communication JSON via une classe de test. En revanche, je sais qu'il est possible d'utiliser des web-services depuis une application Android et que les fonctionnalités d'ajout/suppression d'enseignant d'un appareil existent via l'application web d'administration. Je pense donc qu'il est possible d'appeler ces fonctionnalités dans le code des activités d'ajout/suppression de l'application Android et je pense que j'aurai réussi si j'avais eu plus de temps.

4 Documents additionnels produits

J'ai rédigé 2 documents en parallèle du développement :

- Un guide du développeur qui vise à aider un développeur à prendre en main plus rapidement le projet. Il liste notamment les modifications à faire dans les fichiers de configuration pour pouvoir compiler le code sur sa machine et donne une description rapide des classes principales du projet.
- Un guide utilisateur qui est un tutoriel agrémenté de screenshots afin de guider un utilisateur novice dans l'utilisation de l'application.

J'ai aussi produit une documentation Javadoc du projet (principalement sur les classes importantes ou difficile à comprendre) et commenté mon code afin d'aider un futur développeur. Cependant, le projet contenant beaucoup de code déjà écrit mais vierge de commentaire, je n'ai pas pu tout commenter.

6

Bilan et conclusion

1 Première partie

Pour faire le bilan de la première partie du projet je dirai que j'ai connu plusieurs difficultés. Tout d'abord mon projet consiste à reprendre une application existante afin de la modifier et de l'améliorer. Cette tâche n'est pas aisée car il faut commencer par bien s'approprier l'existant, son fonctionnement et ses subtilités. Ensuite, il faut travailler en ajoutant ce que l'on veut. Dans le cas de la rédaction du cahier de spécifications je ne savais pas à quel point il fallait parler de l'existant, ayant peur d'en dire trop ou pas assez car lors du développement de l'application existante ce document n'avait pas été rédigé.

De plus, mon client étant très occupé, j'ai parfois eu quelques petits problèmes de communication et de délais, pour obtenir des informations notamment. Cependant, j'ai pu passer outre ces problèmes pour rédiger un cahier de spécifications dont mon client et mon encadrant sont satisfaits.

Concernant l'état de l'art et la recherche, la plupart du travail a consisté en la prise en main et la compréhension de l'application existante puisqu'aucune application similaire n'existe et que peu de recherches ont été nécessaires pour le moment.

Enfin, je pense avoir pris un peu de retard pour la suite car je n'ai pas encore eu vraiment le temps de regarder le code de l'application. J'aurai aimé faire cela durant cette première partie afin de pouvoir directement commencer à coder dans la deuxième. Le planning de développement et la composition des lots de livrables sont disponibles en annexe **B**.

2 Deuxième partie

Concernant cette deuxième partie du PRD, la partie développement, je dirais qu'elle s'est plutôt bien passée. Premièrement, j'ai pu finir plusieurs tâches qui m'avaient été demandées, à savoir la mise à jour de la version d'Android et l'adaptation du design. Ces deux points étaient les points principaux de mon projet. De plus, travailler pendant 3 mois de manière intensive sur le développement d'une application Android m'a permis de grandement faire évoluer mes compétences sur le sujet et c'est un domaine qui me plaît beaucoup.

En revanche, je n'ai pas pu développer tout ce qui m'avait été demandé. J'ai passé beaucoup plus de temps sur la mise à jour de la version d'Android que je ne l'aurai pensé. Cela est

principalement du au fait qu'un saut de 8 versions était fait. Cela entraîne énormément de modifications car Android est une plateforme jeune, en constante et rapide évolution. Je pense que de reprendre le développement d'une vieille application Android n'est vraiment pas une chose facile et qu'il est préférable et plus aisé de maintenir régulièrement ses applications à jour. Ainsi, je n'ai pas pu modifier complètement le système d'affectation enseignant-appareil ni ajouter les petites fonctionnalités supplémentaires telles que la signalisation à l'utilisateur de la non-connexion à internet.

Au niveau la gestion de projet, je pense avoir su utiliser correctement git et en tirer profit. Le stockage de mes documents dans le cloud m'a évité les pertes et au final je n'ai pas perdu de temps sur ces aspects. En revanche, je n'ai pas réellement rendu de livrables à mon client afin qu'il les teste. Puisque je travaillais avec la DSI de l'université et que le git était géré par mon client notamment, je me suis contenté de création de branches différentes.

C'est un projet sur lequel j'ai pris plaisir à travailler et qui a su, je pense, me faire progresser sur plusieurs points. C'était un sujet intéressant mais très orienté développement. Je me suis alors heurté à la version actuelle des PRD qui contiennent une grande partie recherche. Je pense que la formule devrait évoluer vers une séparation recherche-développement variable selon les projets. Il me semblerait correct de commencer par une partie recherche qui s'arrêterait quand le tuteur considère quelle est finie. Ainsi, les projets demandant plus de développement ou plus de recherche ne seraient pas lésés. Cela n'empêcherait en rien une évaluation à mi-année pour contrôler l'avancée.

Annexes

A

Spécifications fonctionnelles détaillées

1 Système d'affectation des professeurs aux appareils

L'algorithme est présenté par la figure 1 :

```
Si c'est la première connexion d'une carte Atout'Centre sur l'appareil Alors
    Demander si on veut associer l'enseignant de la carte à l'appareil
    Si on veut associer Alors
        Associer
        Charger les données correspondantes
Sinon
    Connecter le professeur
    Charger les données correspondantes
```

Figure 1 – Algorithme d'affectation

B

Diagramme de Gant et contenu des lots de livrables

La figure 1 est le diagramme de Gant de ce projet. 5 livrables seront effectués et 7 tâches sont prévues en tout :

- Du 20/12/17 au 09/01/18 : Compréhension et appropriation du code existant.
- Du 10/01/18 au 18/01/18 : Tâche 1, mise à jour de l'application vers le SDK le plus récent.
- Du 19/01/18 au 01/02/18 : Tâche 2, changement du design de l'application vers le responsive design et le material design.
- Du 02/02/18 au 15/02/18 : Tâche 3, utilisation des bases de données universitaires et changement du système d'affectation.
- Du 16/02/18 au 08/03/18 : Tâche 4, correction des bugs et réalisation du tutoriel et de l'aide.
- Du 09/03/18 au 22/03/18 : Tâche 5, signalisation à l'utilisateur de la connexion internet, de l'activation du NFC, de la date de dernière synchronisation ; rendre la synchronisation bloquante ; ajouter la gestion de l'application par swipe et afficher les présences ultérieures des élèves.
- Du 23/03/18 au 29/03/18 : Rédaction du rapport final.

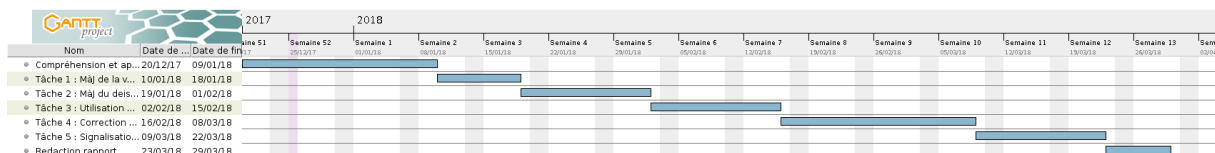


Figure 1 – Diagramme de Gant du projet



Webographie

- [WWW1] *Gradle 3.0.0 and IntelliJ error.* URL : <https://stackoverflow.com/questions/47178724/the-sdk-platform-tools-version-26-0-2-is-too-old-to-check-apis-compiled-with-a> (visité le 30/03/2018).
- [WWW2] *Migrate to Android Plugin for Gradle 3.0.0.* URL : <https://developer.android.com/studio/build/gradle-plugin-3-0-0-migration.html> (visité le 30/03/2018).
- [WWW3] *Migrating from LocationClient to FusedLocationProviderApi.* URL : <http://sanastasov.blogspot.fr/2014/12/migrating-from-locationclient-to.html> (visité le 01/04/2018).
- [WWW4] *Permissions Overview.* URL : <https://developer.android.com/guide/topics/permissions/overview.html#normal-dangerous> (visité le 01/04/2018).
- [WWW5] *Retrieving Location with LocationServices API.* URL : https://github.com/codepath/android_guides/wiki/Retrieving-Location-with-LocationServices-API (visité le 01/04/2018).
- [WWW6] *Set Up Google Play Services.* URL : <https://developers.google.com/android/guides/setup> (visité le 01/04/2018).

Outil pour la gestion des présences : Rapport de projet recherche et développement

Kenan Sauret

Encadrement : Pascal Makris



En collaboration avec DSI de
l'Université François Rabelais

Objectifs

Le projet vise à améliorer une application Android de gestion de présence via les cartes Atout'centre. Les objectifs:

- Rendre l'application utilisable dans toute l'université et sur n'importe quel appareil Android
- Ajouter les mises à jour demandées par le SUAPS.



Mise en œuvre

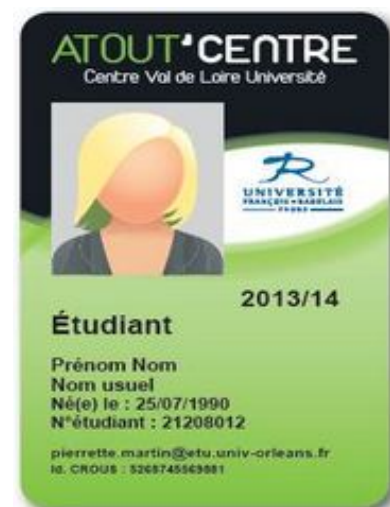
Ce projet est mon projet de fin d'étude à Polytech Tours sur lequel je travaille durant toute l'année 2017-2018. Il comporte deux parties : une phase de recherche et d'état de l'art, suivi d'une phase de développement.



Résultats attendus

On attend à la rentrée 2018 que :

- L'application soit utilisable dans toute l'université.
- L'on puisse se servir de l'application sur n'importe quel appareil Android.



Outil pour la gestion des présences : Rapport de projet recherche et développement

Kenan Sauret

Encadrement : Pascal Makris

Objectifs

Le projet vise à améliorer une application Android de gestion de présence via les cartes Atout'centre. Les objectifs:

- Rendre l'application utilisable dans toute l'université et sur n'importe quel appareil Android
- Ajouter les mises à jour demandées par le SUAPS.

Mise en œuvre

Ce projet est mon projet de fin d'étude à Polytech Tours sur lequel je travaille durant toute l'année 2017-2018. Il comporte deux parties : une phase de recherche et d'état de l'art, suivi d'une phase de développement.

Résultats attendus

On attend à la rentrée 2018 que :

- L'application soit utilisable dans toute l'université.
- L'on puisse se servir de l'application sur n'importe quel appareil Android.



En collaboration avec DSI de
l'Université François Rabelais



android

Logo d'Android



POLYTECH[®] TOURS



ECOLE POLYTECHNIQUE DE L'UNIVERSITÉ FRANÇOIS RABELAIS DE TOURS
Département Informatique
64 avenue Jean Portalis, 37200 Tours, France
polytech.univ-tours.fr



POLYTECH[®] TOURS

Informatique

Outil pour la gestion des présences

Rapport de projet recherche et développement

Résumé

Ce projet de recherche et développement consiste à développer une application mobile de gestion de présence à l'aide d'un appareil Android et d'une carte Atout'centre. Pour cela, l'application Android "EMA" (Emargement Mobile Application) sera mise à jour. Cette application a été développée par la DSI de l'Université François Rabelais de Tours et permet aux professeurs de sport du SUAPS de faire pointer les élèves avec leur carte étudiante pendant leurs cours. Elle fonctionne sur des tablettes 7 pouces et on souhaite étendre son champ d'application à toute l'Université. Ainsi, il est question de changer les bases de données avec lesquelles elle interagit, modifier le design de l'application pour qu'elle fonctionne sur n'importe quel appareil et enfin d'ajouter les mises à jours demandées par les professeurs du SUAPS. Les technologies utilisées sont notamment Android pour l'application, NFC et RFID pour la communication entre l'appareil et la carte, et le responsive design.

Mots-clés

Gestion de présence, Android, NFC

Abstract

This project of research and development consist on developing a mobile application for attendance management with an Android device and an university card. In order to do that, the Android application "EMA" will be updated. This application was developed by the DSI of François Rabelais' University and allows sport teachers of the SUAPS to make students check for attendance with their student cards. It works on 7 inches tabs and it is wanted to extend its scope of utilization at the whole university. So, it is question to change the databases the application interacts with, modify the design to make it runnable on any device and to add the updates asked by SUAPS's members. The technologies used are Android for the application, NFC and RFID for the communication between the device and the card, and the responsive design.

Keywords

Attendance management, Android, NFC

Entreprise

DSI de l'Université François Rabelais



Tuteur entreprise

Malika LABANE (Ingénieure à la DSI)

Étudiant

Kenan SAURET (DI5)

Tuteur académique

Pascal MAKRIS