

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](http://polytech.univ-tours.fr)

## Projet Recherche & Développement 2018-2019

# Application d'émargement : EMA

## Outil pour la gestion des présences



**Entreprise**  
DSI de l'Université François Rabelais



**Tuteur entreprise**  
Malika LABANE (Ingénieure à la DSI)

**Étudiant**  
Yan LIU (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](http://www.univ-tours.fr)



Nom	Email	Qualité
Yan LIU	<a href="mailto:yan.liu-3@etu.univ-tours.fr">yan.liu-3@etu.univ-tours.fr</a>	Étudiant DI5
Pascal MAKRIS	<a href="mailto:pascal.makris@univ-tours.fr">pascal.makris@univ-tours.fr</a>	Tuteur académique, Département Informatique
Malika LABANE	<a href="mailto:malika.labane@univ-tours.fr">malika.labane@univ-tours.fr</a>	Tuteur entreprise, Ingénieure à la DSI



# Avertissement

Ce document a été rédigé par Yan LIU 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

Yan LIU, *Application d'émargement : EMA : Outil pour la gestion des présences*, Projet Recherche & Développement, Ecole Polytechnique de l'Université François Rabelais de Tours, Tours, France, 2018-2019.

```
@mastersthesis{
  author={LIU, Yan},
  title={Application d'émargement : EMA : Outil pour la gestion des présences},
  type={Projet Recherche \& Développement},
  school={Ecole Polytechnique de l'Université François Rabelais de Tours},
  address={Tours, France},
  year={2018-2019}
}
```



# Table des matières

<b>Liste des intervenants</b>	<b>a</b>
<b>Avertissement</b>	<b>b</b>
<b>Pour citer ce document</b>	<b>c</b>
<b>Table des matières</b>	<b>i</b>
<b>Table des figures</b>	<b>iv</b>
<b>1 Introduction</b>	<b>1</b>
1 Contexte de la réalisation .....	1
2 Objectifs .....	2
3 Hypothèses .....	2
4 Bases méthodologiques .....	2
<b>2 Description générale</b>	<b>4</b>
1 Environnement du projet .....	4
2 Caractéristiques des utilisateurs .....	4
3 Fonctionnalités du système .....	4
4 Structure générale du système .....	5
<b>3 Etat de l'art</b>	<b>7</b>
1 NFC .....	7
2 Applications Android existantes .....	8
3 EMA : le système EMA déjà mis en œuvre du SUAPS .....	10
3.1 Fonctionnement .....	10
3.2 Limitations .....	11

3.2.1	Utilisation des bases de données .....	11
3.2.2	Design de l'application .....	11
3.2.3	Mises à jour et nouveautés.....	11
<b>4</b>	<b>Analyse et conception</b> .....	<b>12</b>
1	ApplicationServer .....	12
1.1	Base de données EMA.....	12
1.2	Base de données APOGEE .....	13
1.3	Serveur d'applications Web : Glassfish .....	13
1.4	Conception de la fonction de synchronisation.....	14
2	MobileApplication.....	15
<b>5</b>	<b>Mise en oeuvre</b> .....	<b>16</b>
1	Mise en oeuvre du semestre 9 .....	16
2	Mise en oeuvre du semestre 10 .....	18
2.1	Pour ApplicationServer .....	18
2.1.1	Déploiement .....	18
2.1.2	Ajouter des liens entre des bases de données .....	19
2.1.3	Types des synchronisations .....	19
2.1.4	Résultats de la synchronisation.....	19
2.1.5	Autres fonctionnalités de ApplicationServer.....	20
2.2	Pour MobileApplication .....	22
2.2.1	Réalisation des fonctions.....	22
2.2.2	Test.....	26
<b>6</b>	<b>Bilan et conclusion</b> .....	<b>30</b>
1	Bilan du semestre 9 .....	30
2	Planning du semestre 10 .....	30
3	Bilan du semestre 10 .....	30
4	Bilan sur la qualité .....	31
5	Travail futur.....	31
6	Conclusion .....	32
	<b>Annexes</b> .....	<b>33</b>
<b>A</b>	<b>Planification</b> .....	<b>34</b>
1	Aperçu de gestion de projet .....	34
2	Découpage des tâches.....	34

<b>B</b>	<b>Description des interfaces externes du logiciel</b>	<b>37</b>
1	Interfaces matériel/logiciel .....	37
2	Interfaces homme/machine .....	37
3	Interfaces logiciel/logiciel .....	37
<b>C</b>	<b>Spécifications fonctionnelles</b>	<b>39</b>
<b>D</b>	<b>Spécifications non fonctionnelles</b>	<b>41</b>
1	Contraintes de développement et conception .....	41
2	Contraintes de fonctionnement et d'exploitation .....	41
2.1	Performances.....	41
2.2	Capacités.....	41
2.3	Contrôlabilité.....	42
2.4	Sécurité .....	42
<b>E</b>	<b>Analyse avec les détails et compléments</b>	<b>43</b>

# Table des figures

## 2 Description générale

1	Diagramme de cas d'utilisation .....	5
2	Diagramme de composants du système.....	6

## 3 Etat de l'art

1	Logo NFC .....	7
2	RollCall - Attendance Manager.....	9
3	Attendance.....	9
4	Attendance Taker.....	10

## 4 Analyse et conception

1	La structure de la base de données EMA .....	12
2	La structure et les données de la table properties .....	13
3	La structure de la base de données APOGEE .....	13
4	Logo GlassFish.....	14
5	Organigramme de synchronisation.....	14
6	Table <etudiant> dans la BDD EMA .....	15
7	Table <emarg etudiant> dans la BDD APOGEE .....	15

## 5 Mise en oeuvre

1	Structure de la base de données Emargement .....	16
2	Structure de Emargappserver .....	17
3	Ajout la base de données APOGEE .....	18
4	Ajouter des liens entre des bases de données .....	19

5	Les codes synchronisation réussie.....	20
6	Les codes synchronisation réussie.....	20
7	Apogee Synchro tables réussie .....	21
8	Apogee Synchro numeroEnseignant réussie.....	21
9	Apogee Synchro csu réussie .....	21
10	Apogee Synchro photo réussie .....	21
11	Autoriser les appareils mobiles par défaut à utiliser EMA .....	21
12	La table Deviceenseignant.....	22
13	Ajouter les exceptions .....	22
14	Ajouter les recherches .....	23
15	Recherche par nom ou prénom .....	23
16	Défilement du nom du cours .....	24
17	Ajouter 3 boutons pour Swipe .....	25
18	Modifier statut de présence avec succès .....	25
19	Afficher le mode enseignant .....	26
20	Les onglets RUN et Profiler dans android studio .....	27
21	Test Espresso.....	27
22	Enregistrer l'opération.....	28
23	Test réussi .....	28
24	Les tests que j'ai fait.....	29
 <b>A Planification</b>		
1	Le diagramme de Gantt.....	34
2	Les tâches dans le diagramme de Gantt.....	35
 <b>B Description des interfaces externes du logiciel</b>		
1	Diagramme de l'application mobile.....	38
 <b>E Analyse avec les détails et compléments</b>		
1	Diagramme E-R de la base de données EMA.....	43
2	Diagramme de classe de persistance.entities.emargement .....	44
3	Diagramme de classe de persistance.entities.suaps.....	45
4	Diagramme de classe de synchro.suaps .....	46
5	Diagramme de classe de persistance.entities.apogee .....	47
6	Diagramme de classe de synchro.apogee.....	48

# 1

## Introduction

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, et est exécuté par Yan LIU, élèves ingénieurs en 5ème année de spécialité Informatique à Polytech Tours.

Le client du projet est Mme.MALIKA LABANE qui est ingénieure à la DSI de l'Université François Rabelais.

## 1 Contexte de la réalisation

L'application d'émargement :

- EMA est déjà développé, cela fonctionne bien et peut être utilisé.
- Elle a été modifiée et améliorée par un étudiant de l'année dernière.

Cette application est déployée actuellement uniquement pour le contrôle d'assiduité aux séances d'activités sportives gérées par le Service Universitaire des Activités Physiques et Sportives (SUAPS)

La base de données de EMA est actuellement synchronisée avec celle de l'application SUAPS. Il est possible de mettre en place un autre connecteur avec Apogée, application de gestion des étudiants et des formations, qui permettrait de mettre à disposition des enseignants de tout l'établissement ce moyen de contrôle d'assiduité pour des cours autres que ceux du SUAPS à partir des inscriptions pédagogiques réalisées dans Apogée.

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 4 ans maintenant, le suivi des présences lors des cours de sport est fait par pointage de la carte étudiante sur une tablette. Après avoir été modifié par l'étudiant de

l'année dernière, il peut être utilisé sur les téléphones mobiles. 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, l'application a été développée pour la base de données suaps, qui attire des enseignants d'autres écoles, et les enseignants d'autres écoles souhaitent utiliser l'application pour le émargement des élèves. Nous ajoutons donc la base de données Apogee au programme afin que d'autres composants puissent également utiliser ce programme.

## 2 Objectifs

A partir de inscriptions des étudiants par internet à des cours par l'application spécifique aux UEO de sports, appelée SUAPS, l'application EMA permet d'établir des listes d'étudiants inscrits à ces séances et de valider leur assiduité à chaque séance via une tablette NFC qui récupère le numéro d'étudiant dans la carte multiservice et les informations relatives aux étudiants dans le système d'information à partir du n° d'étudiant.

Pour tout le système EMA, nous avons deux parties :

- ApplicationServer : Synchronisez la base de données EMA avec d'autres bases de données et gérez-la par l'administrateur.
- MobileApplication : Application Android pour l'émargement.

Pour la partie ApplicationServer :

- L'objectif du projet est d'ajouter une base de données Apogee aux projets existants pour les rendre disponibles dans l'ensemble de l'université, et pas seulement pour les suppléments.
- Mon objectif est d'ajouter une base de données Apogee à l'application et d'apporter certaines améliorations nécessaires à l'affichage de l'interface.
- La base de données Apogee contient plus de données que la base de données Suaps et je traiterai des problèmes fonctionnels pouvant être causés par de grandes quantités de données.

Pour la partie MobileApplication :

- Mon objectif est d'ajouter et modifier des fonctionnalités afin d'améliorer l'application
- L'application est une application Android qui fonctionne sur les smartphones et les tablettes Android. Et envisagez de développer des applications IOS à l'avenir (pour iPhone).

## 3 Hypothèses

Notre objectif principal est d'ajouter la base de données APOGEE à l'application et de créer les mêmes fonctionnalités pour APOGEE que la base de données SUAPS. Premièrement, nous écrivons un code similaire pour la base de données APOGEE de la même manière que le code SUAPS dans Serveur, mais la base de données APOGEE dispose de davantage de stockage de données que la base SUAPS. Il peut donc sembler que les fonctions disponibles pour SUAPS ne sont pas disponibles pour APOGEE. Nous allons trouver une solution à ce problème possible.

## 4 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.

Le code existant provient de GitLab et l'ensemble des sources sera versionné en utilisant GitLab, Git et son application graphique Gitkraken.

Le code existant est écrit avec IntelliJ IDEA et continuera à être codé avec IntelliJ IDEA.

Pour la base de données, utiliser le logiciel wampserver pour ouvrir phpMyAdmin et gérer mysql.



# 2

## Description générale

### 1 Environnement du projet

L'environnement de développement consiste aux points suivants :

- Le système d'exploitation : Windows ;
- Le langage de programmation : JAVA JDK 1.8.0 ;
- L'IDE pour le développement : IntelliJ IDEA ;

### 2 Caractéristiques des utilisateurs

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

- Les professeurs(Administrateur) : 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(Utilisateur) : 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 suivant représente le diagramme des cas d'utilisation de l'application.

Un professeur(Administrateur) peut réaliser les actions suivantes :

- Se connecter à le smartphone avec une carte.
- Se connecter à le smartphone avec un mot de passe(PIN code).
- Sélectionner une offre de formation.
- Sélectionner un cours.
- Sélectionner une séance.
- Localiser le lieux 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, mettre sa carte sur l'appareil lors d'un émargement.



Figure 1 – Diagramme de cas d'utilisation

## 4 Structure générale du système

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

L'application mobile ainsi que l'application web serveur interagissent avec une base de données EMA. Ces 3 composants forment le système EMA. Cette base de données EMA interagit avec les bases de données SUAPS et APOGEE.

La base de données SUAPS avant EMA et servait à stocker les informations sur les inscriptions des cours des activités physiques et sportives.

La base de données APOGEE existait avant EMA et servait à stocker les informations sur les inscriptions des étudiants de l'université.

Il existe des liens complexes entre la base de données SUAPS et la base de données APOGEE. Mais pour développer notre application, nous traitons les bases de données SUAPS et APOGEE en tant que bases de données distinctes et nous interagissons avec la base de données EMA séparément.

La différence avec le système existant est que la base de données EMA ajoute une interaction directe avec la base de données de l'université(APOGEE).

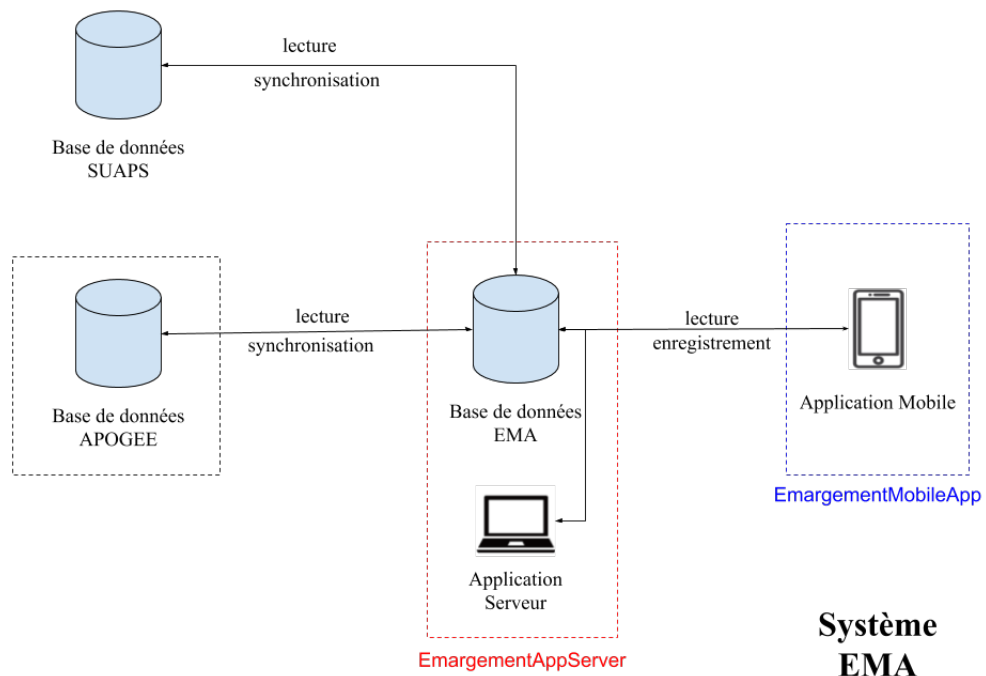


Figure 2 – Diagramme de composants du système

# 3

## Etat de l'art

Cette section présente les développements actuels de la technologie appliquée au projet, qui comprend trois parties : la technologie NFC, les applications Android existantes et le système EMA déjà mis en œuvre.

### 1 NFC

La technologie NFC (communication en champ proche) est une technologie émergente dans laquelle les appareils utilisant la technologie NFC (tels que les téléphones mobiles) peuvent échanger des données très proches les uns des autres.

La technologie a évolué depuis l'intégration de la technologie d'identification par radiofréquence (RFID) sans contact et de la technologie d'interconnexion.

En intégrant des lecteurs de cartes inductifs, des cartes inductives et une communication entre homologues sur une seule puce, les terminaux mobiles peuvent être utilisés pour mettre en œuvre des applications telles que la reconnaissance d'identité mobile.



Figure 1 – Logo NFC

La communication en champ proche est une technologie de communication sans fil à courte portée développée sur la base de la technologie RFID. La portée de transmission de la communication en champ proche est inférieure à celle de la RFID, qui peut aller de 0 à 1 m. Cependant, comme la technologie NFC adopte une technologie d'atténuation du signal unique, la technologie NFC présente les caractéristiques suivantes : faible coût, bande passante élevée et faible consommation d'énergie.

Les principales caractéristiques de la technologie de communication en champ proche sont les suivantes :

- Technologie de communication sans fil pour une communication sécurisée à une courte distance (moins de 10 cm).
- fréquence RF : 13,56 MHz.
- Compatible RF : ISO 14443, ISO 15693, norme de Felica.
- Vitesse de transmission des données : 106 kbit / s, 212 kbit / s, 424 kbit / s.

#### Application NFC

Les applications NFC comprennent Touch and Go, Touch and Pay, Touch and Connect et Touch and Explore.

Pour les systèmes de contrôle d'accès, NFC utilise Touch and Go. Pour le paiement sans contact par téléphone mobile, NFC adopte Touch and Pay, pour ce projet, Touch and Explore, qui permet à l'enseignant de consulter les informations relatives à cet élève. Facile à mettre en œuvre la vérification de l'émargement.

## 2 Applications Android existantes

La deuxième partie de mon analyse de l'état de la technique consistait à déterminer si une application mobile Android similaire existait déjà. J'ai donc recherché un système d'enregistrement des présences similaire dans le magasin d'applications Google et trouvé 4 applications mobiles similaires aux ce objectifs de mise en œuvre du projet :

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

Les trois premières applications, leur objectif principal est très similaire à ce projet. Ils sont conçus pour aider les enseignants à gérer l'assiduité des élèves en classe. Pour ce faire, le programme nécessite que l'utilisateur ajoute manuellement des étudiants et des séances des cours. L'enseignant peut ensuite sélectionner manuellement l'élève à assister ou absent, ou choisir de s'absenter pour cause de maladie. Le programme permet également aux utilisateurs d'exporter des enregistrements de présence sous différents formats ou de modifier des enregistrements de présence à tout moment.

Lorsque ces applications mobiles entrent tous les paramètres, elles peuvent mieux fonctionner, mais ces applications mobiles obligent l'utilisateur à saisir manuellement des paramètres (tels que des cours, des étudiants, etc.), ce qui prend beaucoup de temps. Ou bien l'utilisateur peut également importer les paramètres pertinents via le fichier csv, mais pour ce projet, nous utilisons la base de données pour stocker ces paramètres. Par conséquent, cela ne répond pas à nos exigences de développement. Et l'application prend uniquement en charge la saisie et la sauvegarde des informations en anglais, mais pas en français.

Bien qu'il existe déjà de nombreuses applications mobiles similaires aux objectifs de ce projet, aucune d'entre elles n'est similaire au système à écrire dans ce projet. Cependant, basées sur ces applications mobiles, elles fournissent des idées pour la gestion de la participation. Par exemple, nous pouvons également créer une version EMA pour les étudiants qui permettra aux étudiants de gérer leurs propres horaires et d'afficher leur présence actuelle. En même temps, je pense que les enseignants peuvent également transmettre des informations aux étudiants via l'application mobile. La version EMA pour étudiants ne fait pas partie des objectifs principaux du projet, mais peut être utilisée comme orientation de développement futur.



Figure 2 – RollCall - Attendance Manager

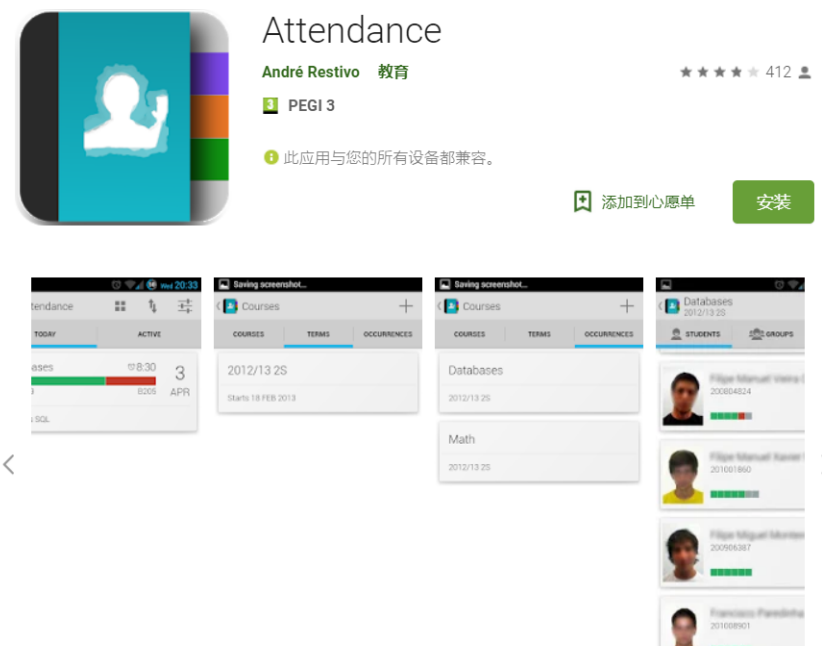


Figure 3 – Attendance

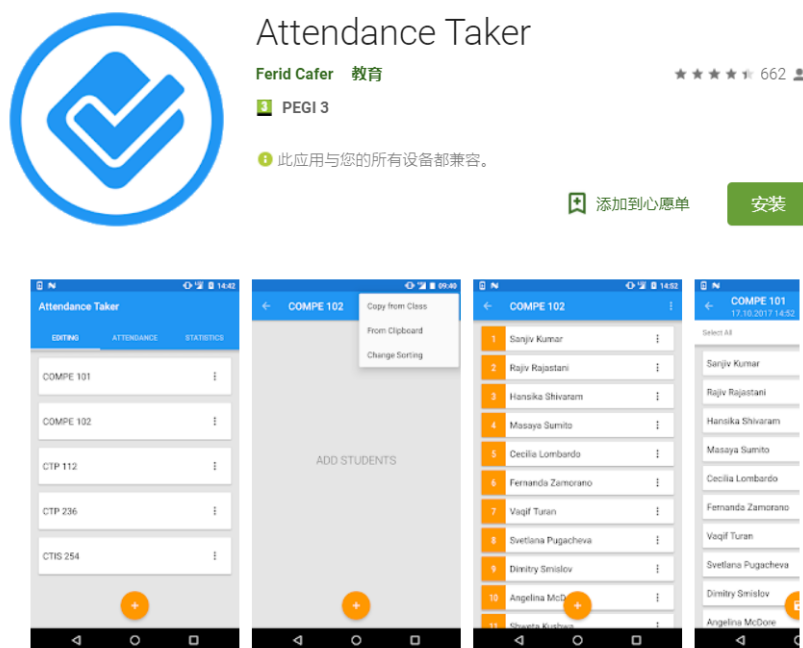


Figure 4 – Attendance Taker

### 3 EMA : le système EMA déjà mis en œuvre du SUAPS

#### 3.1 Fonctionnement

C'est une application Android qui fonctionne sur des appareils de Android 8.1 (Oreo). 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 retentit.
- 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.

## 3.2 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 pour tous les cours(APOGEE).

### 3.2.1 Utilisation des bases de données

Le programme contacte actuellement la base de données SUAPS pour gérer l'enregistrement pour tous les cours d'éducation physique, mais l'objectif de ce projet est de gérer la présence à tous les cours dans l'université (il ne se limite pas à l'éducation physique). Informations d'inscription au cours pour tous les étudiants de l'université.

C'est avec cette base APOGEE que la base de données EMA dialogue. Ainsi, l'application EMA est libre d'effectuer des modifications dans sa propre base qui seront synchronisées à la base de données APOGEE sans influencer les bases universitaires si on veut étendre l'application à toute l'UFR. En effet, les informations nécessaires existant déjà dans les bases de données de Universitaires.

### 3.2.2 Design de l'application

À l'heure actuelle, l'interface de l'application mobile convient à la téléphonie mobile, mais la planification de l'interface est très simple; il convient donc de modifier l'interface actuelle et de rendre l'interface de l'application mobile plus claire et plus pratique à utiliser.

### 3.2.3 Mises à jour et nouveautés

Actuellement, l'application dispose déjà de fonctionnalités de base pour l'émargement, mais certaines fonctionnalités ont encore des bugs qui doivent être corrigés, et doivent encore améliorer les détails des fonctionnalités, ajouter de nouvelles fonctionnalités et être plus ergonomique pour mieux utilisation et fonctionnement.



# 4

## Analyse et conception

### 1 ApplicationServer

Pour la partie serveur, nous utilisons ApplicationServer pour établir la connexion entre la base de données EMA et la base de données APOGEE. Nous avons besoin :

- La structure de la base de données EMA et ses données.
- La structure de la base de données APOGEE et ses données.

Dans le même temps, nous avons besoin :

- Serveur d'applications Web : Glassfish

#### 1.1 Base de données EMA

Le nom de la base de données EMA : emargement. Nous pouvons connaître la structure de la base de données EMA à l'aide de la figure suivante. Elle contient 18 tables.

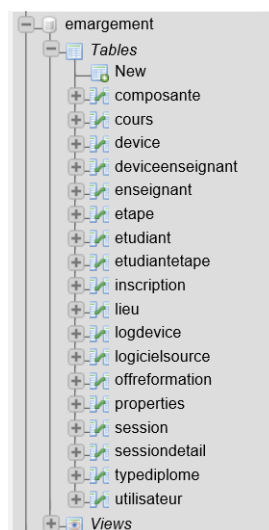


Figure 1 – La structure de la base de données EMA

Il est à noter que, pour la table `properties`, elle stocke des sites Web sur lesquels des informations peuvent être obtenues, lesquelles sont fournies par des serveurs de l'Université de Tours.

Par exemple, des informations sur la photo peuvent être obtenues par `photoUrl`; des informations sur CSN, nous pouvons l'obtenir via `csnUrl`, etc.

La table `properties` est la clé de notre synchronisation.

idProperties	photoUrl	csnUrl	numeroPersonnelUrl	currentVersion	betaVersion	urlCurrentVersion	urlBetaVersion
1	https://unicampus.univ-tours.fr/Unicampus/photos/%...	http://atoutweb.univ-tours.fr/resources/v1/cartes/...	http://atoutweb.univ-tours.fr/resources/v1/ldap/us...	42	42	http://emarg.univ-tours.fr/app/current/app.apk	http://emarg.univ-tours.fr/app/beta/app.apk

Figure 2 – La structure et les données de la table `properties`

## 1.2 Base de données APOGEE

Parce que la base de données APOGEE est une base de données plus grande et plus complexe que la base de données SUAPS. Nous n'avons pas encore obtenu la structure et les données de la vraie base de données APOGEE.

Par conséquent, j'ai simulé les données APOGEE en fonction de la structure et des données des données SUAPS pour synchroniser la base de données EMA avec la base de données APOGEE.

Le nom de la base de données APOGEE : `ueli-test`.

Nous pouvons connaître la structure de la base de données APOGEE à travers la figure suivante. Elle contient 12 tables.

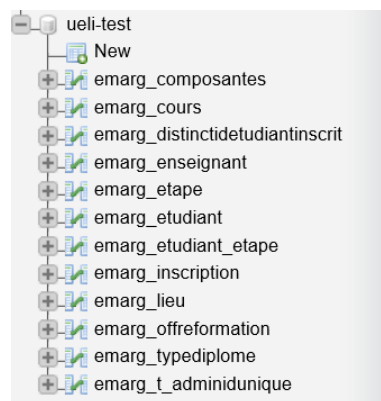


Figure 3 – La structure de la base de données APOGEE

## 1.3 Serveur d'applications Web : Glassfish

But de l'utilisation de Glassfish : pour lancer le `Emargappserver`.

Nous devons configurer Glassfish et déployer le `Emargappserver` à l'aide de Glassfish.

Donc l'utilisation de glassfish ça veut dire créer une connectivité entre les bases de données et avec le serveur de l'université.

Afin de synchroniser la base de données EMA avec la base de données APOGEE et la base de données de l'Université de Tours, nous devons utiliser un serveur Glassfish pour établir la connectivité entre eux.

Nous utilisons un serveur Glassfish :

- En configurant glassfish, la connexion entre le code ApplicationServer et la base de données est réalisée.
- Implémentez l’affichage du site Web du gestionnaire en déployant le code ApplicationServer.



Figure 4 – Logo GlassFish

#### 1.4 Conception de la fonction de synchronisation

Conception pour la synchronisation de la base de données EMA et de la base de données APOGEE.

- Premièrement, nous devons configurer et déployer glassfish afin que le serveur d’applications puisse se connecter à d’autres bases de données.
- Lors de la synchronisation, nous copions les données de les mêmes colonne de la base de données APOGEE à la base de données EMA.
- Ensuite, les colonnes dans la base de données EMA, mais n’ai pas dans la base de données APOGEE, sont synchronisées à partir de la serveur de l’Université de Tours via une URL.

L’organigramme de synchronisation est présenté ci-dessous(Figure 5).

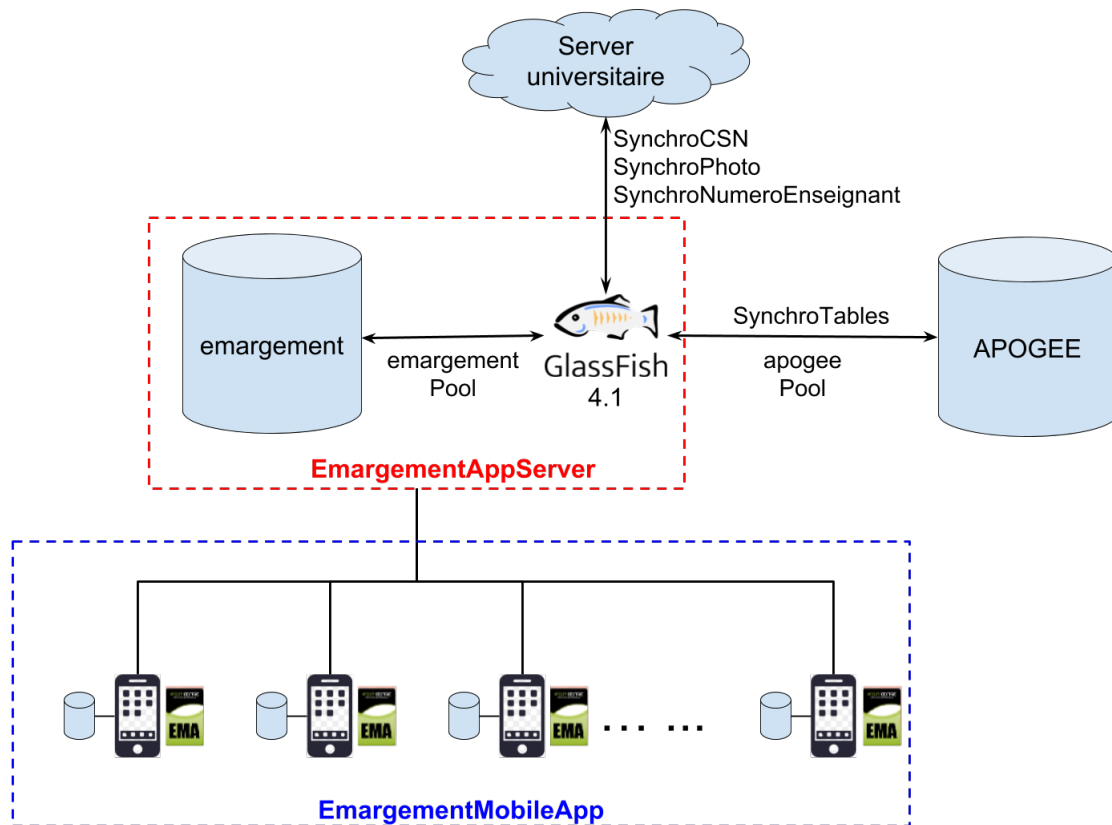


Figure 5 – Organigramme de synchronisation

Par exemple, pour la table étudiant.

- La table <etudiant> dans la BDD EMA.

idEtudiant	numeroEtudiantSource	idLogicielSource	nom	prenom	email	tel	photo	datePhoto	CSN	dateMAJ	deleted
5467	21102247	1					/9j/4A/	2011-07-12 10:05:16	04698	2019-01-01	1
5476	21201605	1				06	/9j/4A/	2014-09-01 07:45:35	04675	2019-01-01	1
5498	21200832	1				06	/9j/4A/	2012-07-10 15:09:48	04704	2019-01-01	1

Figure 6 – Table <etudiant> dans la BDD EMA

- La table <emarg etudiant> dans la BDD APOGEE.

numero	nom	prenom	email	tel
21102247			etu.univ-tours.fr	
21201605			tu.univ-tours.fr	
21200832			@etu.univ-tours.fr	

Figure 7 – Table <emarg etudiant> dans la BDD APOGEE

Nous synchronisons toutes les colonnes de la base de données APOGEE sur les colonnes correspondantes de la base de données EMA.

- numero —> numeroEtudiantSource
- nom —> nom
- prenom —> prenom
- email —> email
- tel -> tel

Ensuite, les colonnes photo et CSN nécessaires dans la base de données EMA sont obtenues auprès du serveur de l'Université de Tours via photoUrl et csUrl dans la table properties (Figure 2).

## 2 MobileApplication

Pour la partie application, ajouter et modifier des fonctionnalités pour améliorer l'application.

- 1.Rendre possible l'association d'un mobile et d'un enseignant directement depuis l'application mobile EMA : table deviceenseignant renseignée depuis l'appareil mobile.
- 2.Erreurs possibles :
  - CSN introuvable dans la table enseignant : message d'erreur, contacter admin.
  - Pas de numéro de device : créer la ligne dans la table device avant.
- 3.Ajout boîte de recherche et barre sur le côté avec l'alphabet pour les pages <Etudiantlist>
- 4.Résoudre le problème que le nom du titre est trop long pour s'afficher de manière incomplète.
- 5.Résoudre la fonction <Swipe> pour <EtudiantDetail>
- 6.Résoudre le bug dans changer un statut d'un étudiant sur une séance dans <Etudiantdetail>

# 5

## Mise en oeuvre

### 1 Mise en oeuvre du semestre 9

En S9, j'ai principalement fait :

- J'ai installé le logiciel nécessaire pour utiliser le projet : IntelliJ IDEA, Wampserver64, Git, GitKraken
- J'ai appris le rapport rédigé par les étudiants l'année dernière pour comprendre le contenu spécifique du projet.
- J'apprends la structure de la base de données EMA et je comprends la signification de chaque tableau de la base de données EMA.



Figure 1 – Structure de la base de données Emargement

- J'ai appris le code de l'application Web Serveur, qui apprend principalement :
  - Classes d'entités : `persistence.entités.margement` et `persistence.entités.suaps`

- Classe de synchronisation universelle : `synchro.suaps`
- La classe d'entité contient toutes les tables et colonnes de la base de données EMA et de la base de données SUAPS, y compris la méthode `get () /set ()`, la méthode `euqal ()`, etc.
- La classe de synchronisation générique est une classe d'implémentation Java qui synchronise la base de données EMA avec les entités partagées de la base de données SUAPS. Pour chaque entité, il existe des fonctions telles que ajouter/ mettre à jour/ supprimer dans cette classe.
- Structure Emargappserver et toutes les classes

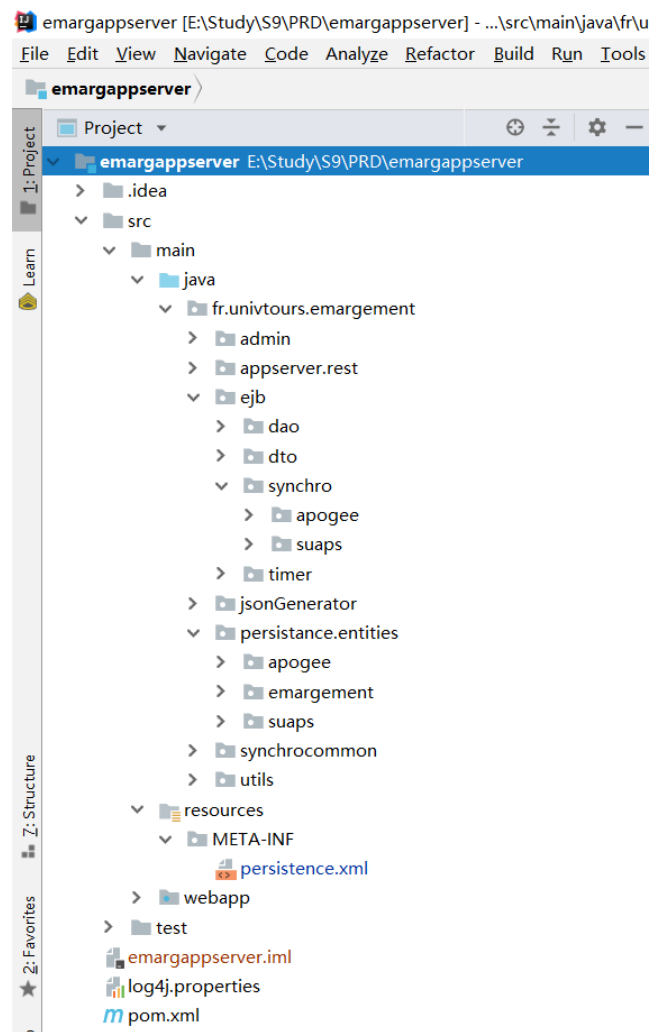


Figure 2 – Structure de Emargappserver

- Pour la fonction de synchronisation consistant à ajouter la base de données APOGEE à la base de données EMA, au cours de ce semestre, j'ai essayé d'écrire du code pour cette fonction sur le modèle de la structure de la base de données SUAPS. Cependant, comme la structure de base de données APOGEE correcte n'a pas encore été obtenue, le code doit encore être modifié après l'obtention de la structure de base de données APOGEE correcte.

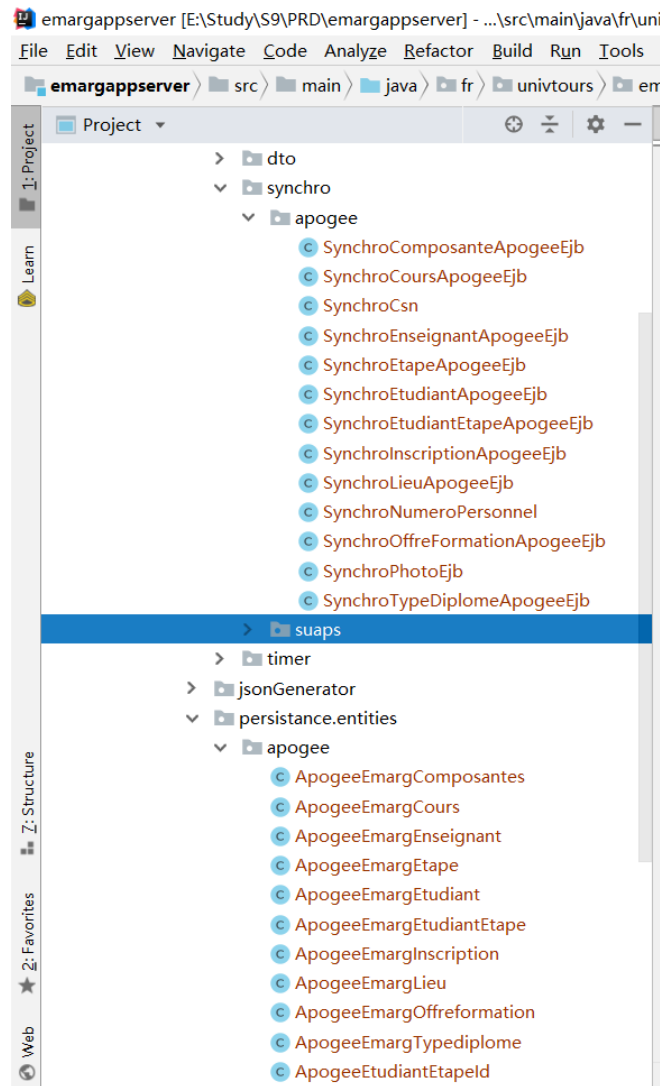


Figure 3 – Ajout la base de données APOGEE

## 2 Mise en oeuvre du semestre 10

### 2.1 Pour ApplicationServer

#### 2.1.1 Déploiement

Parce que c'est impossible d'obtenir l'autorisation pour la base de données sur le serveur de l'université, j'utilise glassfish pour déployer mon serveur localement.

Pour le port, 4848 est le port d'administration officiel de Glassfish.

Mon port est 8081.

L'adresse de serveur : localhost :8081/EmargeAppServer

Pour savoir comment configurer et déployer glassfish, on peut se référer au document fourni par tuteur : Config BDD glassfish.

### 2.1.2 Ajouter des liens entre des bases de données

Nous devons ajouter un lien vers la base de données APOGEE dans la base de données EMA.

J'ai ajouté la base de données Apogee à la table logicielsource de la base de données EMA.

De cette manière, lors de la synchronisation, nous pouvons savoir où se trouve la source des données, si elles sont synchronisées depuis la base de données SUAPS ou dans la base de données APOGEE.

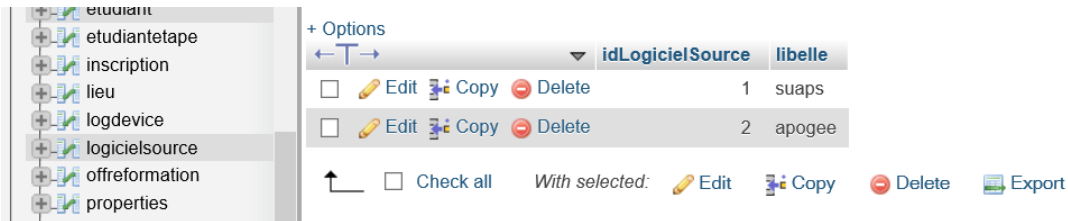


Figure 4 – Ajouter des liens entre des bases de données

### 2.1.3 Types des synchronisations

Il y a 4 types de synchronisation :

- Synchro Tables : Synchroniser toutes les tables de la base de données APOGEE à la base de données emargement.
- Synchro CSN : Rechercher le CSN par le numéro d'étudiant ou du numéro de l'enseignant sur le serveur de l'université, puis synchroniser le CSN sur les tables <enseignant> et <étudiant> dans BDD emargement.
- Synchro Photo : Rechercher l'URL de la photo par le numéro d'étudiant sur le serveur de l'université, puis synchronisez l'URL de la photo sur la table <etudiant> dans BDD emargement.
- Synchro numeroEnseignant : Trouver le numéro d'enseignant par <uid> de l'enseignant sur le serveur de l'université et synchroniser le numéro de l'enseignant sur la table <enseignant> dans BDD emargement.

### 2.1.4 Résultats de la synchronisation

En général, les services Web sont exploités en entrant une URL pour accéder à la page correspondante.

Nous définissons l'URL de l'opération de synchronisation sur localhost :8081/EmargAppServer/rest/v1/bashsynch

Afin de vérifier que la synchronisation a été effectuée avec succès, l'URL nous a donné des informations, le code est le suivant (Figure5 et Figure6).

Ainsi, lorsque les quatre types de synchronisation auront réussi, nous obtiendrons les résultats suivants.

- Apogee Synchro tables
- Apogee Synchro numeroEnseignant
- Apogee Synchro csn
- Apogee Synchro photo

Les images ci-dessous sont les résultats d'une synchronisation réussie lorsque nous avons saisi l'URL dans le navigateur (Figure 7, 8, 9 et 10).



```

switch (typeSynchro) {
    // "tables" synchronisées avec succès
    case "tables" :
        deleteActionComposantApogee();
        deleteActionCoursApogee();
        deleteActionEnseignantApogee();
        deleteActionEtapeApogee();
        deleteActionEtudiantApogee();
        deleteActionLieuApogee();
        deleteActionOffreFormationApogee();
        deleteActionTypeDiplomeApogee();
        deleteActionEtudiantEtapeApogee();
        deleteActionInscriptionApogee();

        addOrUpdateActionComposantApogee();
        addOrUpdateActionCoursApogee();
        addOrUpdateActionEnseignantApogee();
        addOrUpdateActionEtapeApogee();
        addOrUpdateActionEtudiantApogee();
        addOrUpdateActionLieuApogee();
        addOrUpdateActionOffreFormationApogee();
        addOrUpdateActionTypeDiplomeApogee();
        addOrUpdateActionEtudiantEtapeApogee();
        addOrUpdateActionInscriptionApogee();
        reponse = "\\Apogee Synchro tables\\";
        break;
}

```

Figure 5 – Les codes synchronisation réussie

```

// "numeroEnseignant" synchronisées avec succès
case "numeroEnseignant" :
    synchroNumeroPersonnel();
    reponse = "\\Apogee Synchro numeroEnseignant\\";
    break;

// "csn" synchronisées avec succès
case "csn" :
    synchroCsnEnseignant();
    synchroCsnEtudiant();
    reponse = "\\Apogee Synchro csn\\";
    break;

// "photo" synchronisées avec succès
case "photo" :
    synchroPhoto();
    reponse = "\\Apogee Synchro photo\\";
    break;
}

```

Figure 6 – Les codes synchronisation réussie

### 2.1.5 Autres fonctionnalités de ApplicationServer

Autoriser les appareils mobiles par défaut à utiliser EMA : table device authorized=1 par défaut (Figure 11).

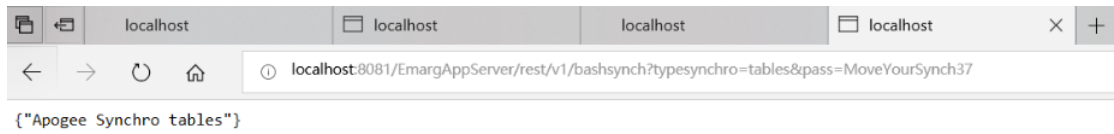


Figure 7 – Apogee Synchro tables réussie

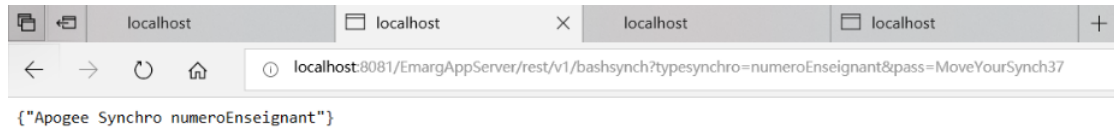


Figure 8 – Apogee Synchro numeroEnseignant réussie

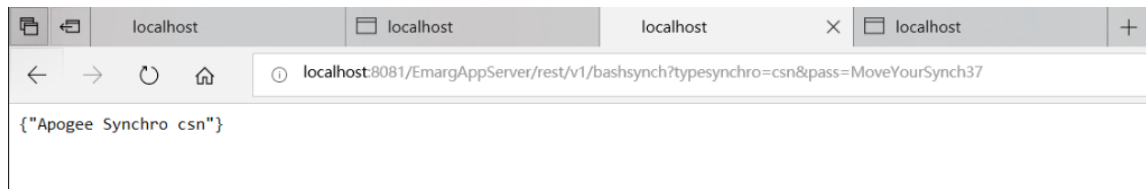


Figure 9 – Apogee Synchro csn réussie

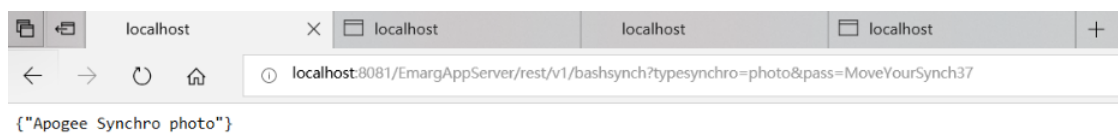


Figure 10 – Apogee Synchro photo réussie

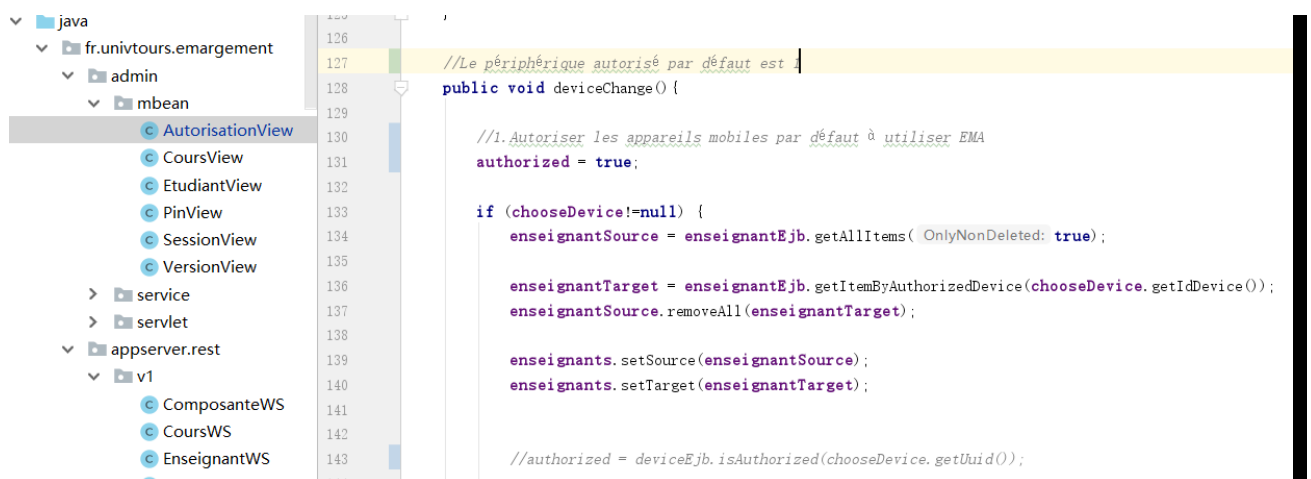


Figure 11 – Autoriser les appareils mobiles par défaut à utiliser EMA

## 2.2 Pour MobileApplication

### 2.2.1 Réalisation des fonctions

- 1. Rendre possible l'association d'un mobile et d'un enseignant directement depuis l'application mobile EMA : table deviceenseignant renseignée depuis l'appareil mobile.  
Résoudre :  
Ajouter la classe d'entité Enseignant et son contrôleur-DeviceenseignantDAO dans le code de MobileApplication.

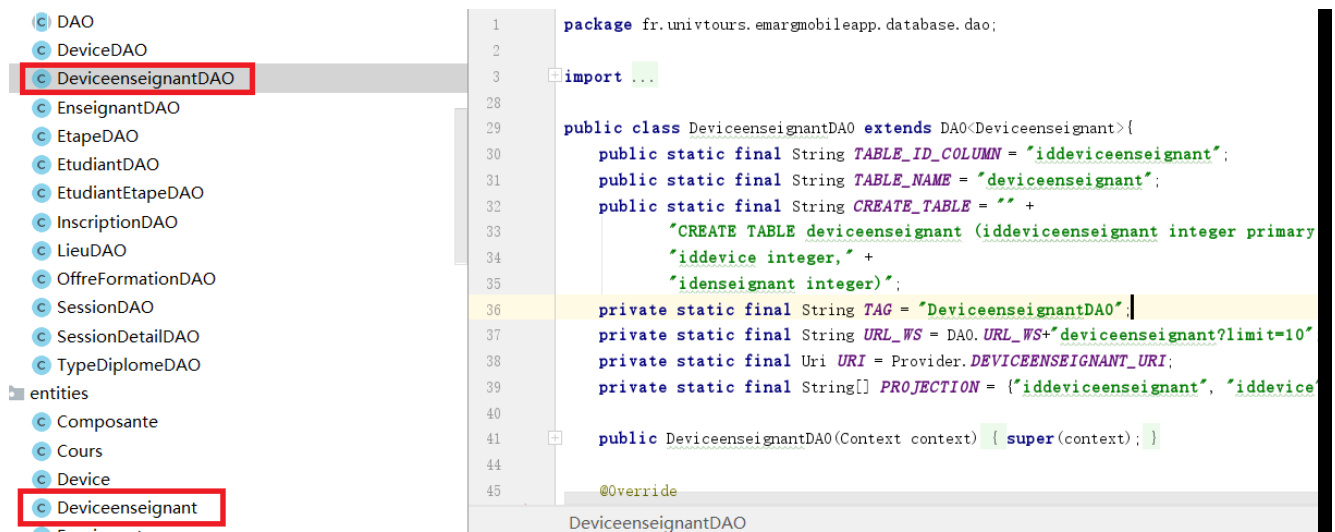


Figure 12 – La table Deviceenseignant

- 2. Erreurs possibles :
  - CSN introuvable dans la table enseignant : message d'erreur, contacter admin.
  - Pas de numéro de device : créer la ligne dans la table device avant.
 Ajouter les rappels d'exception dans MobileApplication pour CSN introuvable et numéro de device introuvable.

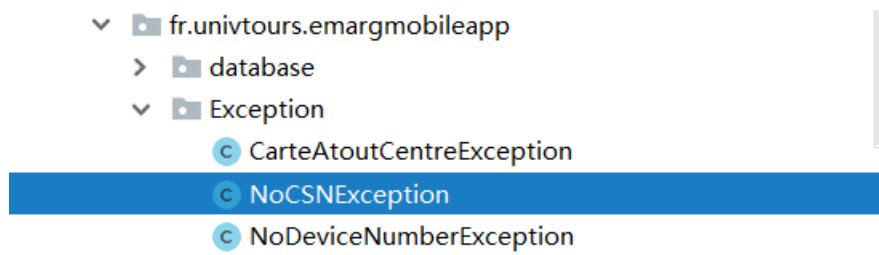


Figure 13 – Ajouter les exceptions

- 3. Ajout boîte de recherche et barre sur le côté avec l'alphabet pour les pages `<Etudiantlist>`
  - Ajout boîte de recherche  
Cliquez sur l'icône de recherche à gauche de la boîte de recherche.  
Nous pouvons effectuer une recherche par nom ou prénom de l'étudiant.
  - Ajout barre sur le côté avec l'alphabet A-Z  
La liste des étudiants a été organisée dans l'ordre de A à Z. Appuyez longuement sur la barre sur le côté pour faire glisser la liste des élèves de A à Z.

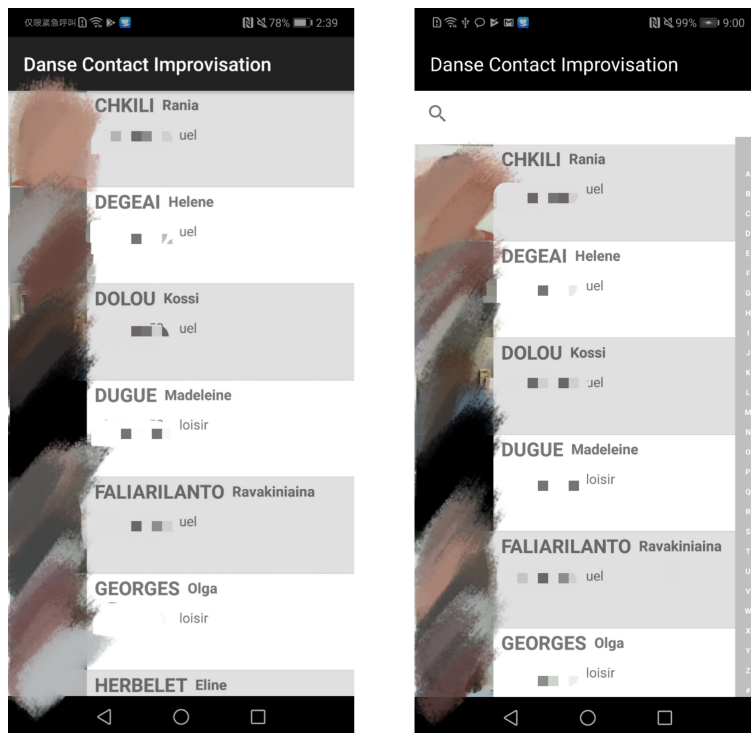


Figure 14 – Ajouter les recherches



Figure 15 – Recherche par nom ou prénom

- 4. Résoudre le problème que le nom du titre est trop long pour s'afficher de manière incomplète.
  - Dans la version précédente, lorsque le nom du cours était très long, le nom du cours ne pouvait pas être entièrement affiché dans la barre de menus et la partie non affichée était remplacée par "...".

- Après avoir ajouté la fonction, le défilement du nom du cours sur la barre de menus est réalisé et tous les noms longs peuvent être affichés.
- Cette fonction est implémentée dans la page <CoursDetail> et <EtudiantList>.

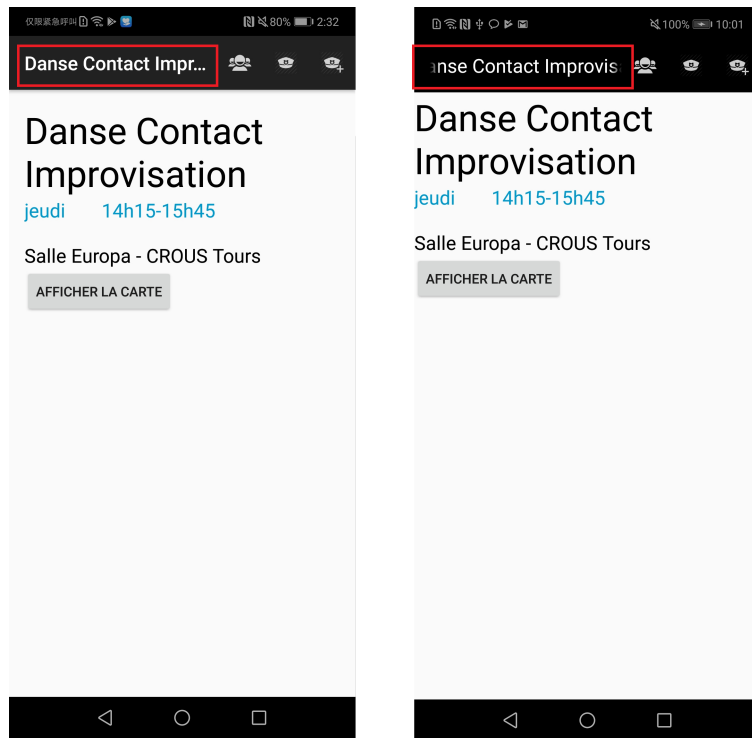


Figure 16 – Défilement du nom du cours

- 5. Résoudre la fonction <Swipe> pour <EtudiantDetail>
  - Dans la version précédente, dans <EtudiantDetail>, les glissades gauche et droite des détails de l'étudiant dans la liste ne pouvaient pas être correctement implémentées et la même page de l'étudiant était toujours affichée lors de la glissade.
  - Dans la version précédente, <EtudiantDetail> était défini comme composant de fragment, puis imbriqué dans l'activité pour implémenter le glissement, mais le composant de fragment n'était pas compatible avec la fonction d'implémentation de boutons dans <EtudiantDetail> dans ce projet.
  - Par conséquent, afin de résoudre ce problème, je définis directement <EtudiantDetail> en tant qu'activité et ajoute 3 boutons dans <EtudiantDetailActivity> : AVANT, RETOUR, PROCHAINE (Figure 17).
  - Nous cliquons sur ces trois boutons pour faire glisser à gauche et à droite les différentes pages de détails sur les étudiants.
    - Cliquez sur AVANT pour passer aux détails de l'élève précédent ;
    - Cliquez sur PROCHAINE pour passer aux détails du prochain élève.
    - Cliquez sur RETOUR pour revenir à la page <EtudiantList>.
- 6. Résoudre le bug dans changer un statut d'un étudiant sur une séance dans <Etudiantdetail>
  - Dans <EtudiantDetail>, nous souhaitons modifier le statut de présence de toutes les séances d'un élève d'un cours.
  - Dans les versions précédentes, lorsque nous apportons des modifications à la page <EtudiantDetail>, celle-ci devenait un écran blanc et quittait le programme.
  - J'ai corrigé ce bug afin que l'enseignant puisse modifier le statut de présence de toutes les séances d'étudiants dans la page <EtudiantDetail> (Figure 18).
  - Dans le même temps, il reste un problème dans cette partie. Pour plus de sécurité,

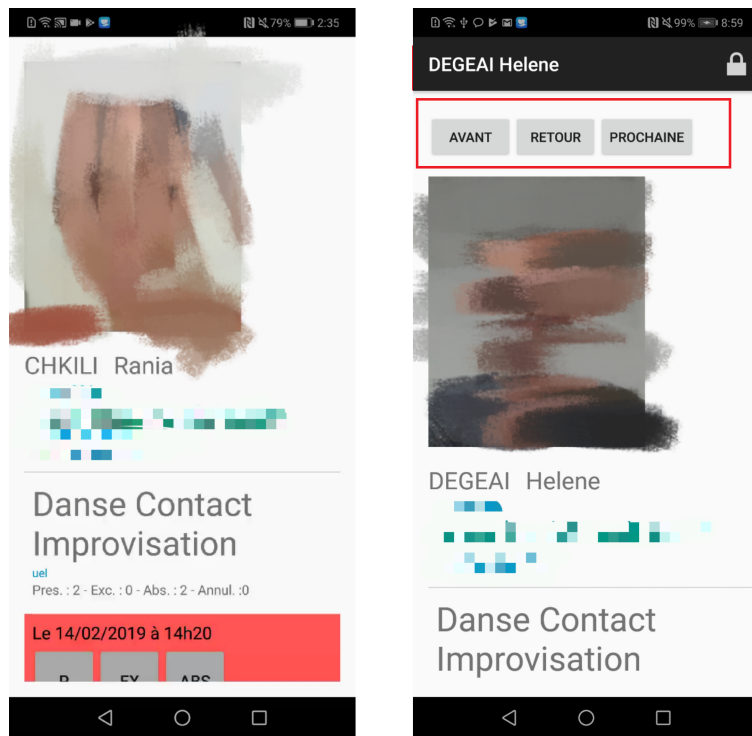


Figure 17 – Ajouter 3 boutons pour Swipe

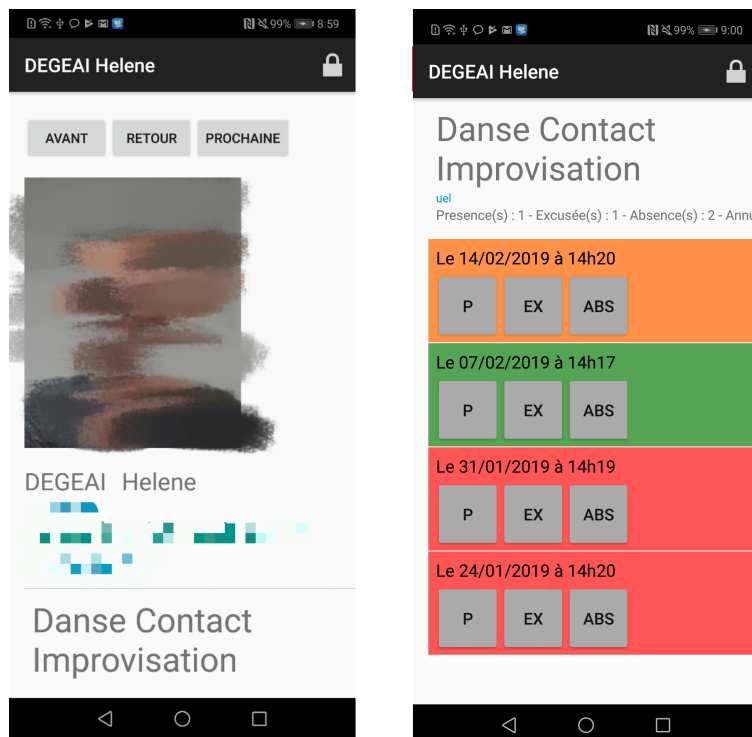


Figure 18 – Modifier statut de présence avec succès

j'espère que l'enseignant pourra se connecter en mode enseignant lorsque l'enseignant modifie le statut de présence de toutes les séances des étudiants dans la page <EtudiantDetail>.

- Je l'ai fait pour afficher le mode enseignant sur cette page, mais il n'est pas possible de le connecter correctement au mode enseignant. Cette fonctionnalité doit donc être

améliorée pour être plus sécurisée (Figure 19).

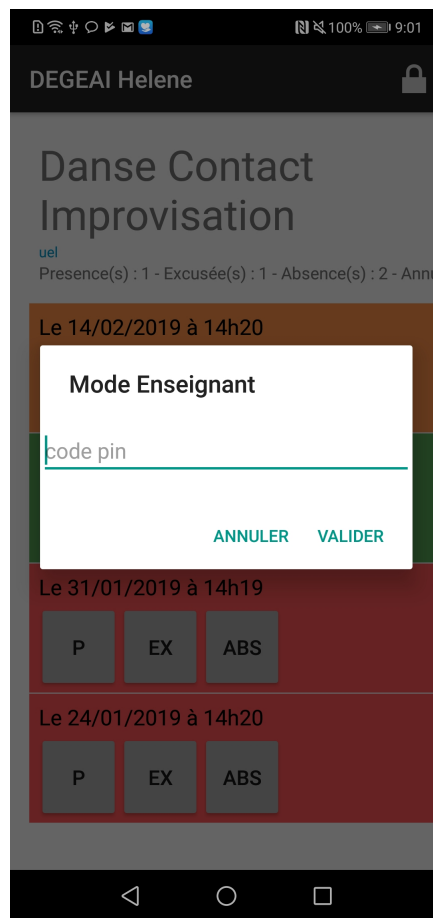


Figure 19 – Afficher le mode enseignant

### 2.2.2 Test

- Test manuel :
  - Après avoir terminé l'écriture d'une partie du code de fonction, j'ai d'abord effectué le test manuel.
  - J'ai connecté le téléphone à l'ordinateur via USB et téléchargé l'apk sur le téléphone.
  - Puis ouvrez le programme pour déboguer la fonction nouvellement écrite, vérifiez si elle a des erreurs et des bugs ;
  - Dans le même temps, en observant l'exécution sur les onglets RUN et Profiler dans android studio pour la surveillance, lorsque le code est incorrect, l'erreur est signalée dans les onglets (Figure 20).
  - Grâce à la répétition de cette méthode, la fonction de l'application est modifiée et améliorée en permanence.
- Test par la méthode Espresso :
  - Après avoir terminé le test manuel, nous pouvons fondamentalement garantir l'utilisation normale de la fonction.
  - À ce stade, j'ai utilisé la méthode Espresso pour tester automatisé toute l'application (Figure 21).
  - Nous pouvons enregistrer le fonctionnement du programme avec la méthode Espresso (Figure 22).

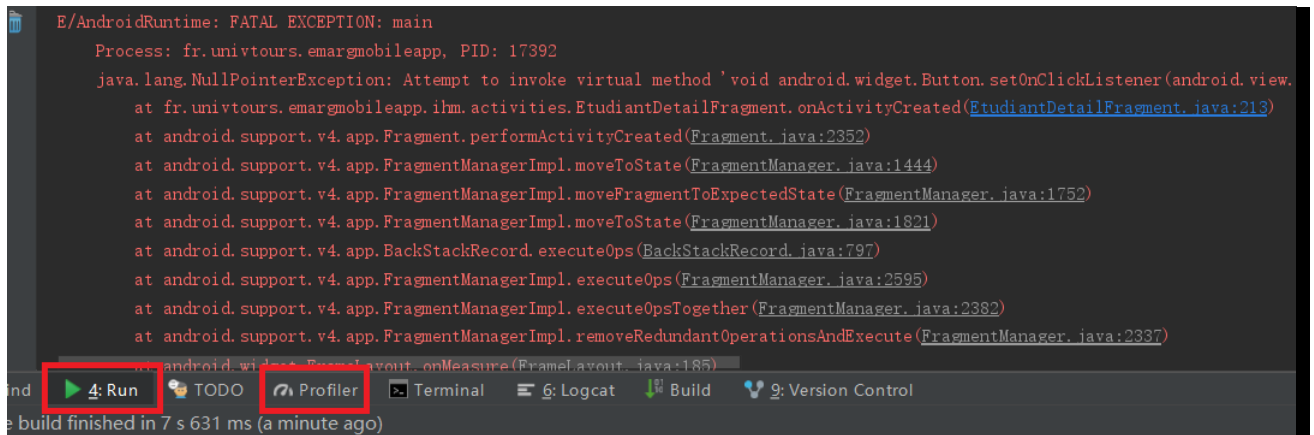


Figure 20 – Les onglets RUN et Profiler dans android studio

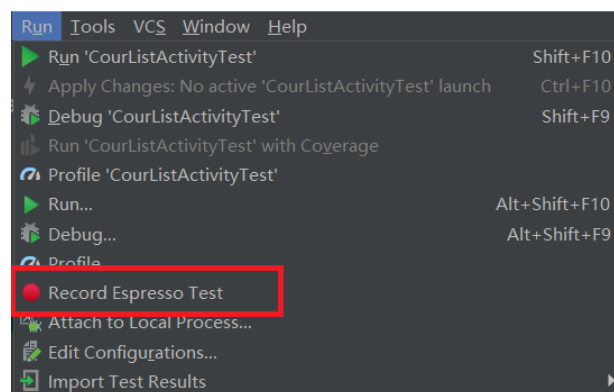


Figure 21 – Test Espresso

- Lorsque le test est lancé, le programme démarre automatiquement l'application et fonctionne conformément à l'enregistrement précédent (Figure 23).
- Cette méthode est utilisée pour détecter s'il existe un problème de connexion entre chaque composant de l'application et chaque fonction.
- J'ai testé la connectivité des fonctionnalités que j'ai ajoutées ou modifiées avec d'autres fonctionnalités du logiciel (Figure 24).
- La méthode de test Espresso est le test d'interface utilisateur, le test d'intégration et le test automatisé.



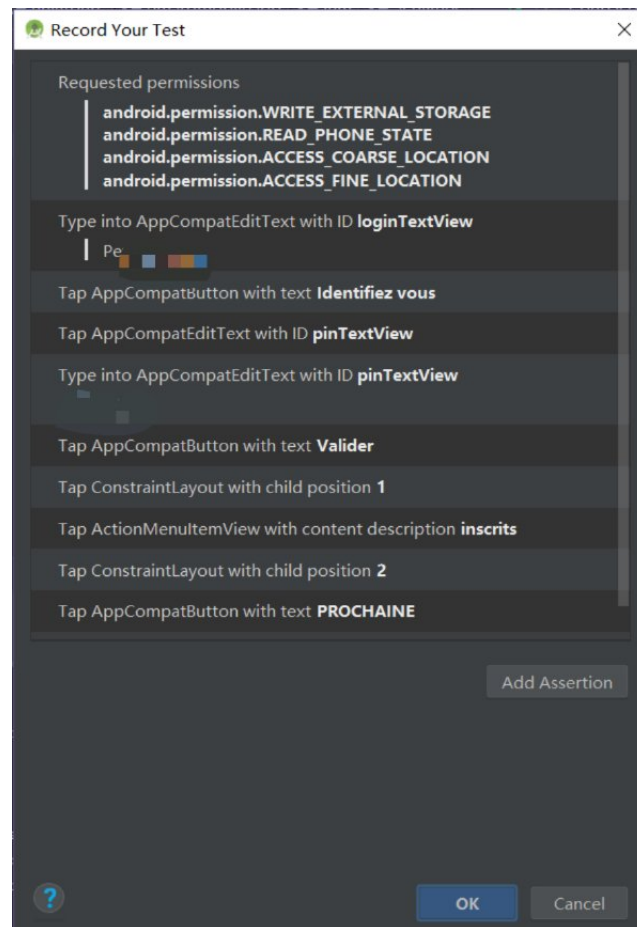


Figure 22 – Enregistrer l'opération

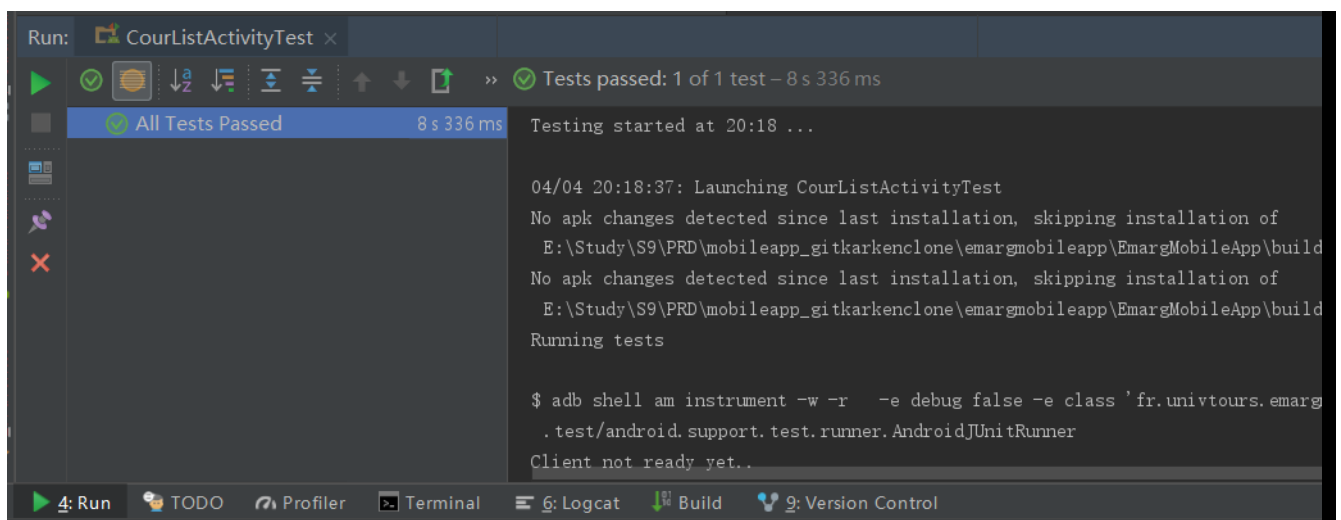
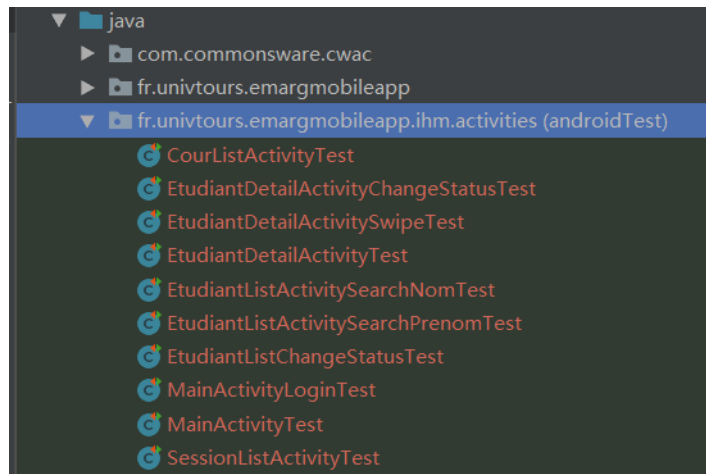


Figure 23 – Test réussi



**Figure 24** – *Les tests que j'ai fait*

# 6

## Bilan et conclusion

### 1 Bilan du semestre 9

En S9, j'ai effectué le travail de base pour la mise en œuvre du code du semestre suivant.

- J'apprends la structure de la base de données EMA et ce que chaque tableau représente.
- J'ai appris la structure de code de l'application Web Serveur, principalement :
  - Classes d'entités : `persistence.entités.margement` et `persistence.entités.suaps`
  - Classe de synchronisation universelle : `synchro.suaps`
- J'ai essayé d'écrire le code de la fonction de synchronisation de la base de données APOGEE et de la base de données EMA côté serveur :
  - Classe d'entité : `persistence.entités.apogee`
  - Classe de synchronisation universelle : `synchro.apogee`
- Mais parce que la structure de base de données APOGEE correcte n'est pas obtenue, cette écriture est simplement une tentative et sera modifiée après avoir obtenu la structure correcte.
- Pour le retard, je pense que je devrais commencer à apprendre le code de l'application émargement à l'étape S9, mais je n'ai pas le temps de le terminer, j'ai mis l'énergie principale sur l'application Web Serveur ce semestre.

### 2 Planning du semestre 10

En S10, je vais faire :

- Tout d'abord, réaliser la fonction de synchronisation de la base de données APOGEE et la base de données EMA dans l'application Web Serveur
- Ensuite, apprenez et écrivez le code de l'application émargement, qui permet à tous les composants de l'université (tous les cours) d'émargement à l'aide de ce programme.
- En outre, apportez les améliorations nécessaires à l'interface interactive de l'application.
- Enfin, testez la fonctionnalité du code (si elle est correcte et parfaite) et les performances.

### 3 Bilan du semestre 10

En S10, j'ai fait :

J'ai travaillé sur ApplicationServer et MobileApplication.

- Pour ApplicationServer :
  - J'ai simulé la structure et les données de la base de données APOGEE à partir de la base de données SUAPS.
  - J'ai configuré et déployé Serveur d'application Glassfish localement.
  - La synchronisation de la base de données apogee et de la base de données EMA sur le serveur local est implémentée.
- Pour MobileApplication :
 

J'ai mis en œuvre le plan pour la partie MobileApplication.

J'ai ajouté des fonctionnalités à l'application, modifié des fonctionnalités et corrigé des bugs importants du point.

Ces fonctionnalités sont :

  - 1.Rendre possible l'association d'un mobile et d'un enseignant directement depuis l'application mobile EMA : table deviceenseignant renseignée depuis l'appareil mobile.
  - 2.Erreurs possibles :
    - CSN introuvable dans la table enseignant : message d'erreur, contacter admin.
    - Pas de numéro de device : créer la ligne dans la table device avant.
  - 3.Ajout boîte de recherche et barre sur le côté avec l'alphabet pour les pages <Etudiantlist>
  - 4.Résoudre le problème que le nom du titre est trop long pour s'afficher de manière incomplète.
  - 5.Résoudre la fonction <Swipe> pour <EtudiantDetail>
  - 6.Résoudre le bug dans changer un statut d'un étudiant sur une séance dans <Etudiantdetail>

Dans le même temps, j'ai testé la partie sur laquelle j'ai travaillé et rédigé des documents livrables.

## 4 Bilan sur la qualité

Pour tout le projet, j'ai fait le cahier de spécification pour un meilleur projet.

Pour la partie du code que j'ai écrite, j'ai ajouté des commentaires afin que les futurs développeurs puissent comprendre le code.

J'ai aussi des documents livrables :

- Technologies utilisées
- Guide du développeur
- Guide d'utilisateur

J'ai testé les fonctionnalités que j'ai ajoutées dans la partie MobileApplication.

## 5 Travail futur

Pour ApplicationServeur :

- Après avoir obtenu la structure correcte de la base de données Apogee, synchroniser la structure correcte de la base de données Apogee afin que l'ensemble de l'université puisse utiliser l'application.

Pour MobileApplication :

- Lorsque l'enseignant souhaite modifier le statut d'émargement de l'étudiant dans le <Etudiant Detail Activity>, il est plus sécurisé de se connecter en mode enseignant et d'entrer le mot de passe à vérifier avant.

## 6 Conclusion

Pendant le Projet Recherche et Développement de ces deux semestres, j'ai appris beaucoup de connaissances et amélioré mes capacités.

Dans le métier de l'informatique :

- Pour recherche : J'ai amélioré des compétences de recherche et d'analyse.
- Pour développement : J'ai appris à propos du développement d'applications Android et du développement de Webservice, ainsi que de GlassFish. Et même le contrôle de la qualité du projet.
- Pour gestion de projet :
  - En cours de PRD, j'ai compris l'importance de la gestion de projet.
  - Nous devons bien couper ces tâches, suivre le plan et le modifier en fonction de la situation.
  - Faire du bon travail dans la gestion de projet signifie promouvoir le projet de manière ordonnée.

Dans le métier d'autre :

- J'ai amélioré des compétences en communication.
- J'ai aussi amélioré capacité à trouver des problèmes et à les résoudre.

Dernier et le plus important :

Je suis très reconnaissant à mon tuteur pour son aide lors du PRD et à tous les enseignants qui m'ont aidée au PRD.

## Annexes

# A

## Planification

En respectant le modèle en « cascade », on fait 3 cycles : l'étude de faisabilité, la spécification des besoins, et l'analyse pendant le semestre 9, et on fait les autres 4 cycles pendant le semestre 10 : la conception détaillée, l'implémentation, le test et la mise en place.

### 1 Aperçu de gestion de projet

Le diagramme de Gantt pour la planification de ce projet est comme Figure1.

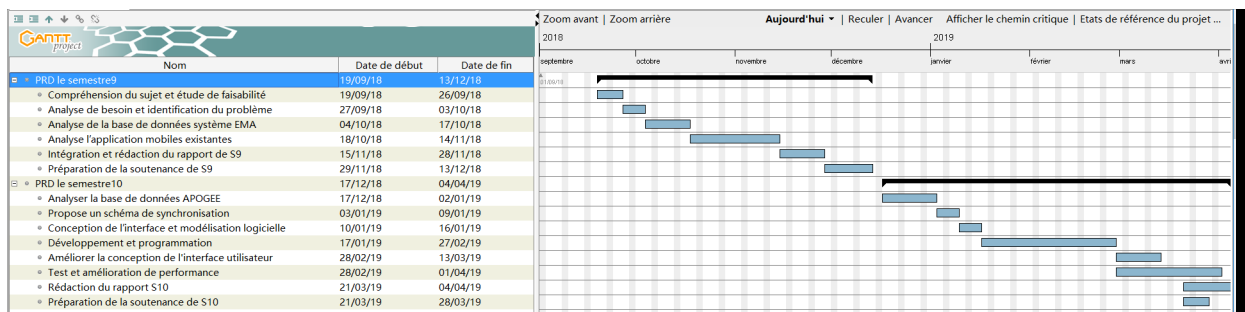



Figure 1 – Le diagramme de Gantt

Pour le S9, le projet commence le 19/09/2018 par la compréhension du sujet et finit le 13/12/2018 par la soutenance de S9.

Pour le S10, le projet commence le 17/12/2018 en Test ARCore sur votre appareil mobile et le projet finit le 05/04/2019 par la soutenance de S10. En moyen, chaque tâche dure 1 ou 2 semaines, mais pour la programmation et le test, ils prendront environ 5 ou 6 semaines.

### 2 Découpage des tâches

1. Compréhension du sujet et étude de faisabilité  
Date de début : le 19 septembre 2018;



Nom	Date de début	Date de fin
PRD le semestre9	19/09/18	13/12/18
• Compréhension du sujet et étude de faisabilité	19/09/18	26/09/18
• Analyse de besoin et identification du problème	27/09/18	03/10/18
• Analyse de la base de données système EMA	04/10/18	17/10/18
• Analyse l'application mobiles existantes	18/10/18	14/11/18
• Intégration et rédaction du rapport de S9	15/11/18	28/11/18
• Préparation de la soutenance de S9	29/11/18	13/12/18
PRD le semestre10	17/12/18	04/04/19
• Analyser la base de données APOGEE	17/12/18	02/01/19
• Propose un schéma de synchronisation	03/01/19	09/01/19
• Conception de l'interface et modélisation logicielle	10/01/19	16/01/19
• Développement et programmation	17/01/19	27/02/19
• Améliorer la conception de l'interface utilisateur	28/02/19	13/03/19
• Test et amélioration de performance	28/02/19	01/04/19
• Rédaction du rapport S10	21/03/19	04/04/19
• Préparation de la soutenance de S10	21/03/19	28/03/19

Figure 2 – Les tâches dans le diagramme de Gantt

Date de fin : le 26 septembre 2018;

Durée : 7 jours;

2.Analyse de besoin et identification du problème

Date de début : le 27 septembre 2018;

Date de fin : le 03 octobre 2018;

Durée : 7 jours;

3.Analyse de la base de données système EMA

Date de début : le 04 octobre 2018;

Date de fin : le 17 octobre 2018;

Durée : 14 jours;

4.Analyse l'application mobiles existantes

Date de début : le 18 octobre 2018;

Date de fin : le 14 novembre 2018;

Durée : 28 jours;

5.Intégration et rédaction du rapport de S9

Date de début : le 15 novembre 2018;

Date de fin : le 28 novembre 2018;

Durée : 14 jours;

6.Préparation de la soutenance de S9

Date de début : le 29 novembre 2018;

Date de fin : le 13 décembre 2018;

Durée : 14 jours;

7.Analyser la base de données APOGEE

Date de début : le 17 décembre 2018;

Date de fin : le 02 janvier 2019;

Durée : 14 jours;

8.Propose un schéma de synchronisation

Date de début : le 03 janvier 2019;

Date de fin : le 09 janvier 2019;

Durée : 7 jours;

9.Conception de l'interface et modélisation logicielle



Date de début : le 10 janvier 2019;

Date de fin : le 16 janvier 2019;

Durée : 7 jours;

10.Développement et programmation

Date de début : le 17 janvier 2019;

Date de fin : le 27 février 2019;

Durée : 42 jours;

11.Améliorer la conception de l'interface utilisateur

Date de début : le 28 février 2019;

Date de fin : le 13 mars 2019;

Durée : 14 jours;

12.Test et amélioration de performance

Date de début : le 28 février;

Date de fin : le 01 avril;

Durée : 32 jours;

13.Rédaction du rapport S10

Date de début : le 21 mars 2019;

Date de fin : le 04 avril 2019;

Durée : 14 jours;

14.Préparation de la soutenance de S10

Date de début : le 21 mars 2019;

Date de fin : le 28 mars 2019;

Durée : 7 jours;

# B

## Description des interfaces externes du logiciel

### 1 Interfaces matériel/logiciel

- Connexion de l'enseignant ou émargement de l'élève : technologie NFC  
Dans le cadre de ce projet, l'interface matérielle/logicielle principale est que l'application doit utiliser la technologie NFC pour identifier les informations de carte des enseignants et des étudiants.  
Il est nécessaire que le terminal mobile de l'enseignant lise des informations sur la carte d'étudiant, laquelle utilisera la technologie NFC pour lire la carte d'étudiant, l'application EMA permet d'établir des listes d'étudiants inscrits à ces séances et de valider leur assiduité à chaque séance via une tablette ou mobile NFC qui récupère le numéro d'étudiant dans la carte multiservice et les informations relatives aux étudiants dans le système d'information à partir du n° d'étudiant. L'enseignant peut également effectuer des opérations manuelles pour vérifier les élèves déjà venus en classe.
- L'application web Serveur : utilisé sous Windows
- L'application émargement : utiliser sur les smartphones ou les tablettes sous Android

### 2 Interfaces homme/machine

Lors de la réalisation de ce projet l'application mobile sera en plate-forme Android.

La principale interface homme-machine est :

- L'enseignant se connecte en entrant un nom d'utilisateur ou en glissant la carte de l'enseignant et en entrant le code PIN.
- Les étudiants s'enregistrent en glissant la carte d'étudiant
- Les enseignants peuvent modifier les informations d'émargement pour tous les élèves.

La navigation pour l'application mobile :

### 3 Interfaces logiciel/logiciel

Le transfert de données entre le périphérique mobile du client et la base de données s'effectue via le système EMA. Les données ne sont pas transférées directement du périphérique mobile

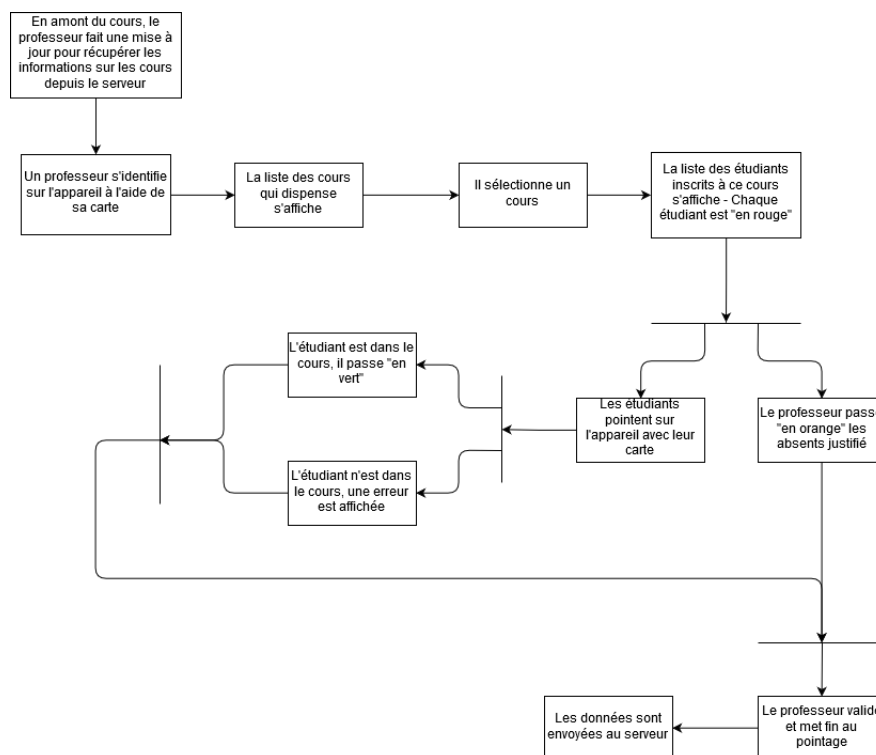


Figure 1 – Diagramme de l'application mobile

du client vers la base de données. Par conséquent, les applications mobiles peuvent utiliser les données de la base de données et les manipuler (insérer, modifier, supprimer ou consulter).

- L'application web Serveur utilise la base de données EMA pour autoriser les utilisateurs de l'application (enseignants).
- La base de données EMA lit et synchronise la base de données SUAPS et la base de données APOGEE

# C

## Spécifications fonctionnelles

C'est une application Android qui fonctionne sur des appareils smartphones ou tablettes. Elle permet de faire émarger les élèves à l'aide de leur carte étudiante lors des tous les cours.

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

- A chaque appareil est associé un ou plusieurs professeurs. Cette association est faite manuellement grâce à une application Web.
- Le professeur se connecte à l'appareil en badgeant sa carte et à l'aide d'un mot de passe(PIN CODE) 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
  - On sélectionne une offre et les cours que le professeur dispense sont affichés
  - On sélectionne un cours
- Une fois sur le cours il y a plusieurs options sont présentes :
  - On voit la localisation du lieu du cours
  - 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. 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 l'appareil. Sa photo s'affiche en grand pour que le professeur puisse faire une vérification puis il passe en vert sur l'appareil pour signifier qu'il est présent. S'il n'est pas dans la liste (carte non reconnue) un son d'erreur retentit.
  - Une fois l'émargement fini on a donc les étudiants présents en vert, les absents en rouge.
  - Le professeur peut utiliser sa carte pour faire entrer l'appareil « 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 étudiants passent en vert clair.

- Une synchronisation via Wifi s'effectue pour envoyer les informations.

# D

# Spécifications non fonctionnelles

## 1 Contraintes de développement et conception

- Language :
  - Java 8 pour l'application web Serveur
  - Java avec Android 8.1 SDK 27 pour l'application émargement
- Plateforme :
  - Windows pour l'application web Serveur
  - Appareil Android (smartphone et tablette) pour l'application émargement

## 2 Contraintes de fonctionnement et d'exploitation

Cette section décrit les contraintes de fonctionnement, qui contiennent 4 enjeux. Ce sont les performances, les capacités, les contrôlabilités et la sécurité.

### 2.1 Performances

Pour les programmes existants,

- Synchroniser les tables de la base de données SUAPS prend 17 minutes
- Synchroniser le nombre d'enseignants dans la base de données SUAPS prend 53 minutes
- La synchronisation de CSN prend 59 minutes
- La synchronisation des photos dans la base de données prend 23 heures

Pour la base de données APOGEE, où le nombre de tables et la quantité d'informations stockées est supérieur à la base de données SUAPS, nous optimiserons autant que possible sa synchronisation et son temps de réponse.

### 2.2 Capacités

Le SI doit être capable de supporter l'utilisation de tous les cours en même temps dans toute l'université et de synchroniser les résultats de chaque enseignant. Le SI doit pouvoir recevoir

les résultats de toutes les interfaces de l'enseignant, par exemple, pour synchroniser toutes les données sur le serveur une fois que l'enseignant a collecté les résultats du cours pendant une journée.

Pour les programmes existants, la base de donnée SUAPS

- Il y a 75 enseignants
- De 8 000 à 9 000 étudiants sont inscrits au SUAPS

Pour la base de donnée APOGEE :

- Il y a environ 1 300 enseignants
- Il y a environ 27990 étudiants

Ainsi, une fois que l'application a rejoint la base de données APOGEE, la base de données EMA :

- La capacité de stockage de l'enseignant sera comprise entre 2400 et 2500
- La capacité de stockage des étudiants sera comprise entre 39 000 et 40 000

### 2.3 Contrôlabilité

Pour contrôler l'accès de l'utilisateur, on doit donner manuellement à l'enseignant des autorisations d'application de connexion dans l'application web Serveur, et on peut concevoir des fichiers log qui sont nommés par la date actuelle. Lorsqu'un utilisateur se connecte et fait des opérations, le système enregistrera la date, le temps, l'identifiant et le nom de l'utilisateur, les opérations, les fichiers qu'il a consultés et les commentaires des opérations de synchronisation, etc.

### 2.4 Sécurité

Le système demande à l'utilisateur(l'enseignant) de se connecter avant de l'utiliser, mais parce qu'il n'y a qu'un type d'utilisateur, l'utilisateur peut faire toutes les opérations une fois il se connecte au système avec succès.

# E

# Analyse avec les détails et compléments

Dans ce chapitre, je vais ajouter quelques images nécessaires au chapitre 4.

— Diagramme E-R(relation d'entité) de la base de données EMA

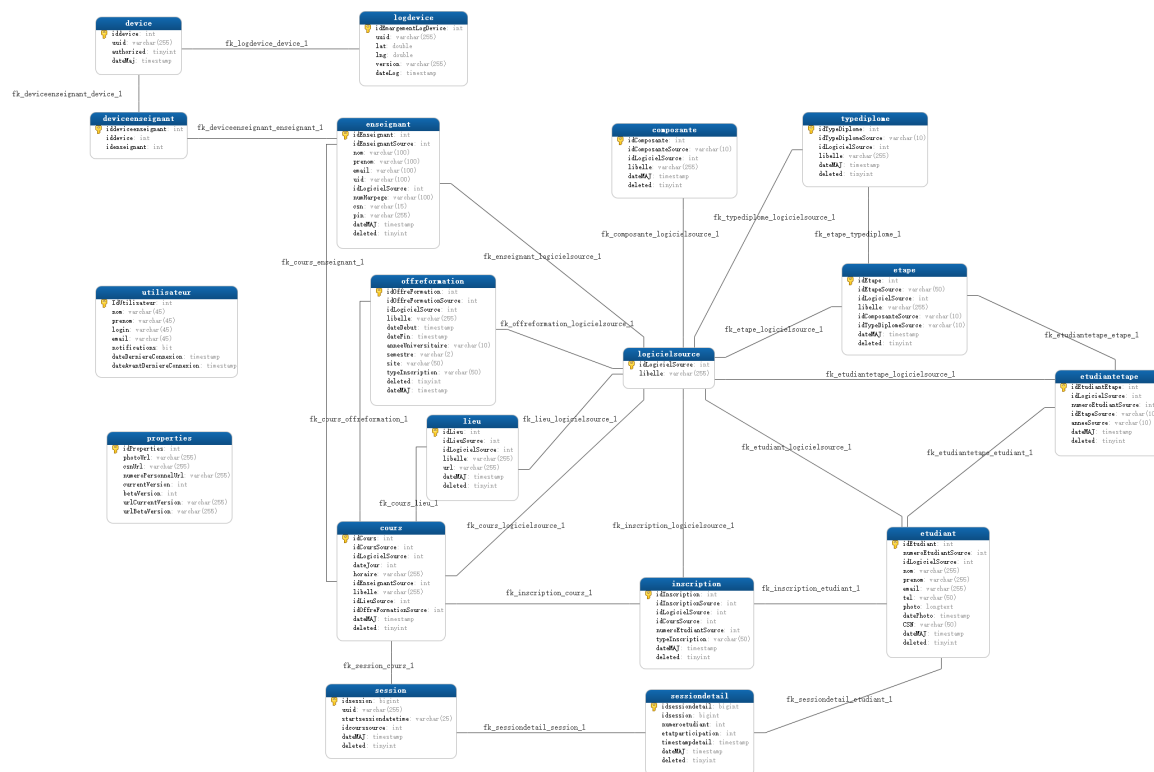


Figure 1 – Diagramme E-R de la base de données EMA

- Diagramme de classe de persistance.entities.emargement
- Diagramme de classe de persistance.entities.suaps
- Diagramme de classe de synchro.suaps
- Diagramme de classe de persistance.entities.apogee
- Diagramme de classe de synchro.apogee



## ANNEXE E. ANALYSE AVEC LES DÉTAILS ET COMPLÉMENTS



**Figure 2** – Diagramme de classe de persistance.entities.emargement

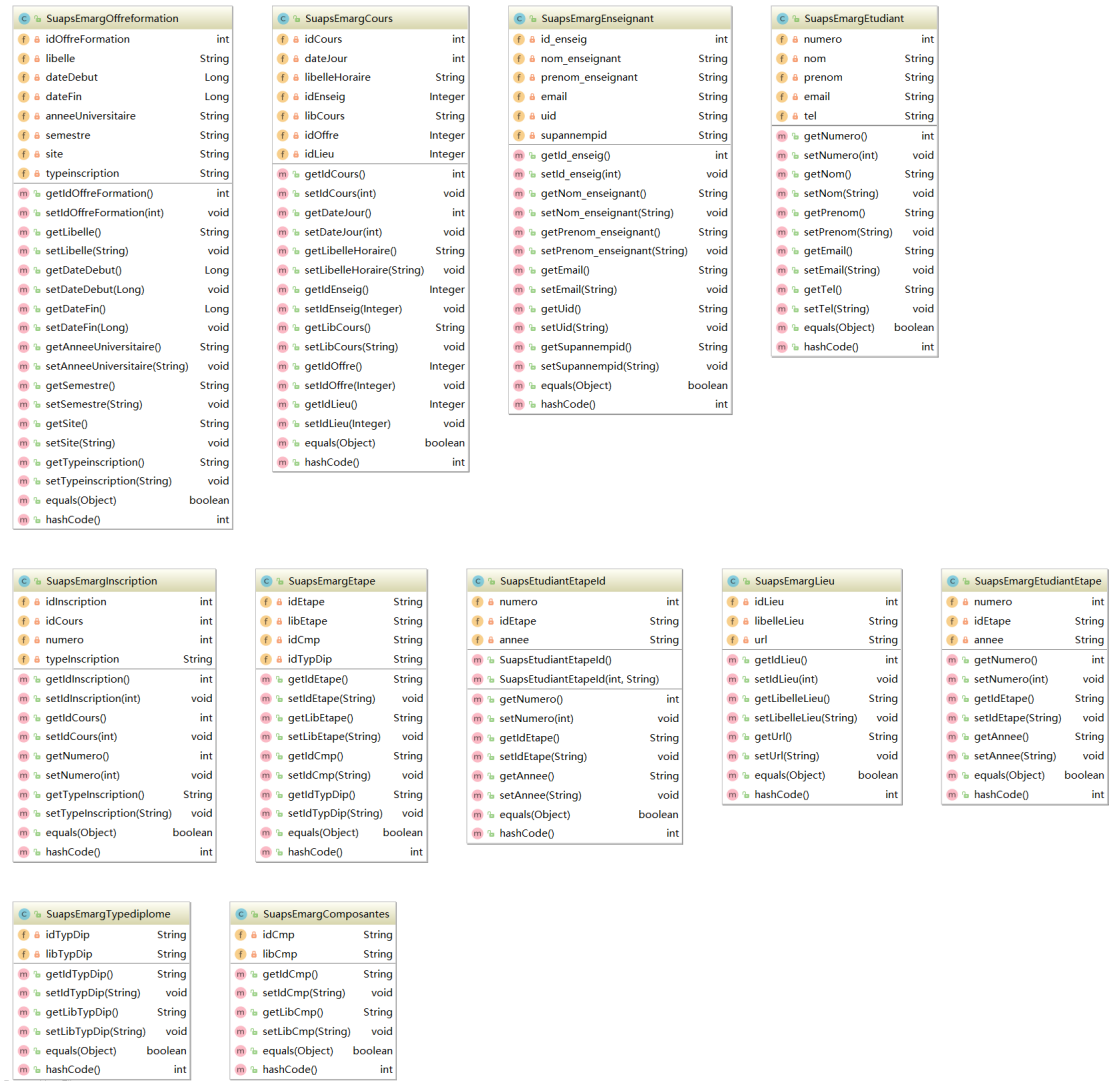


Figure 3 – Diagramme de classe de persistance.entities.suaps

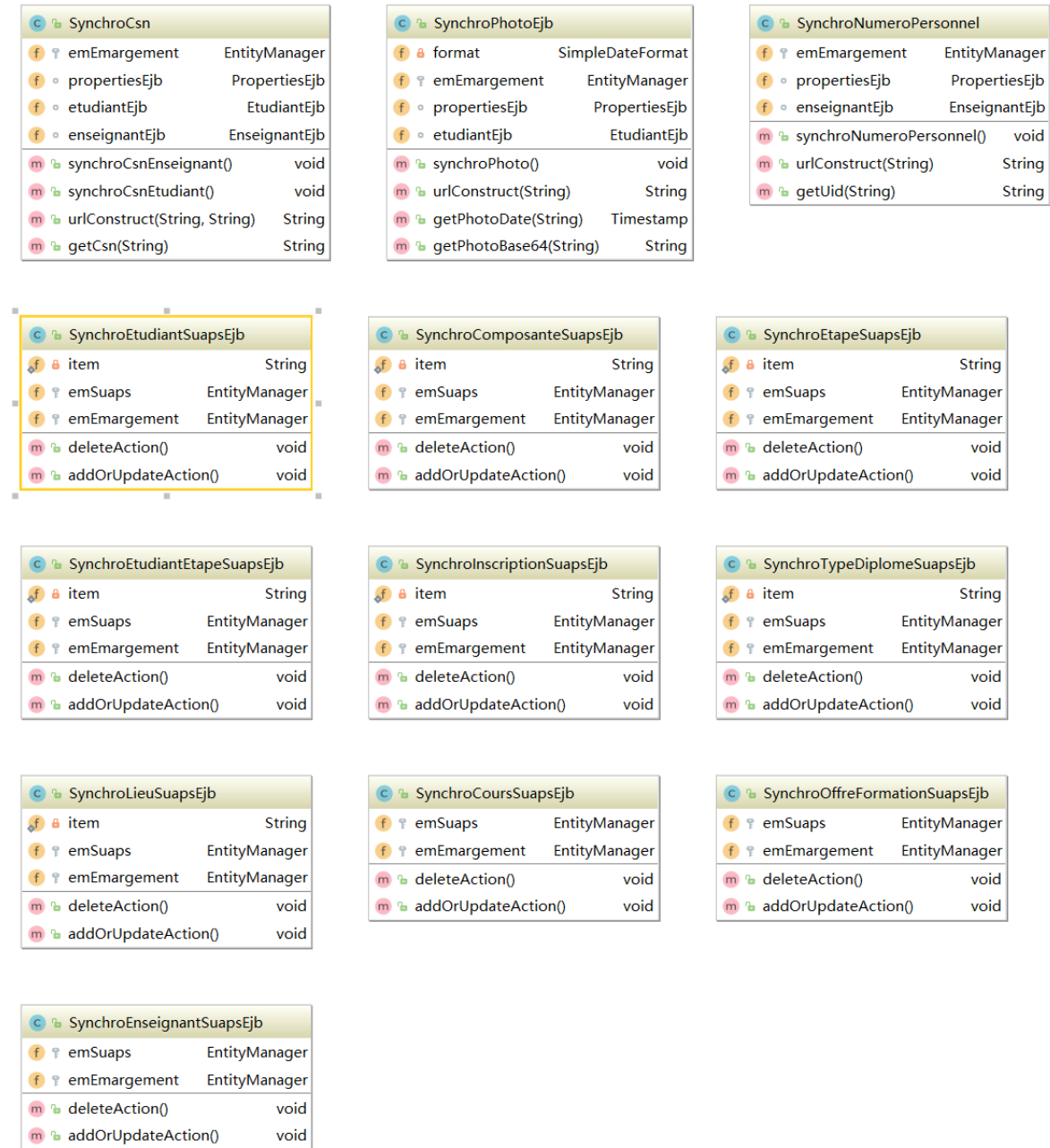


Figure 4 – Diagramme de classe de synchro.suaps

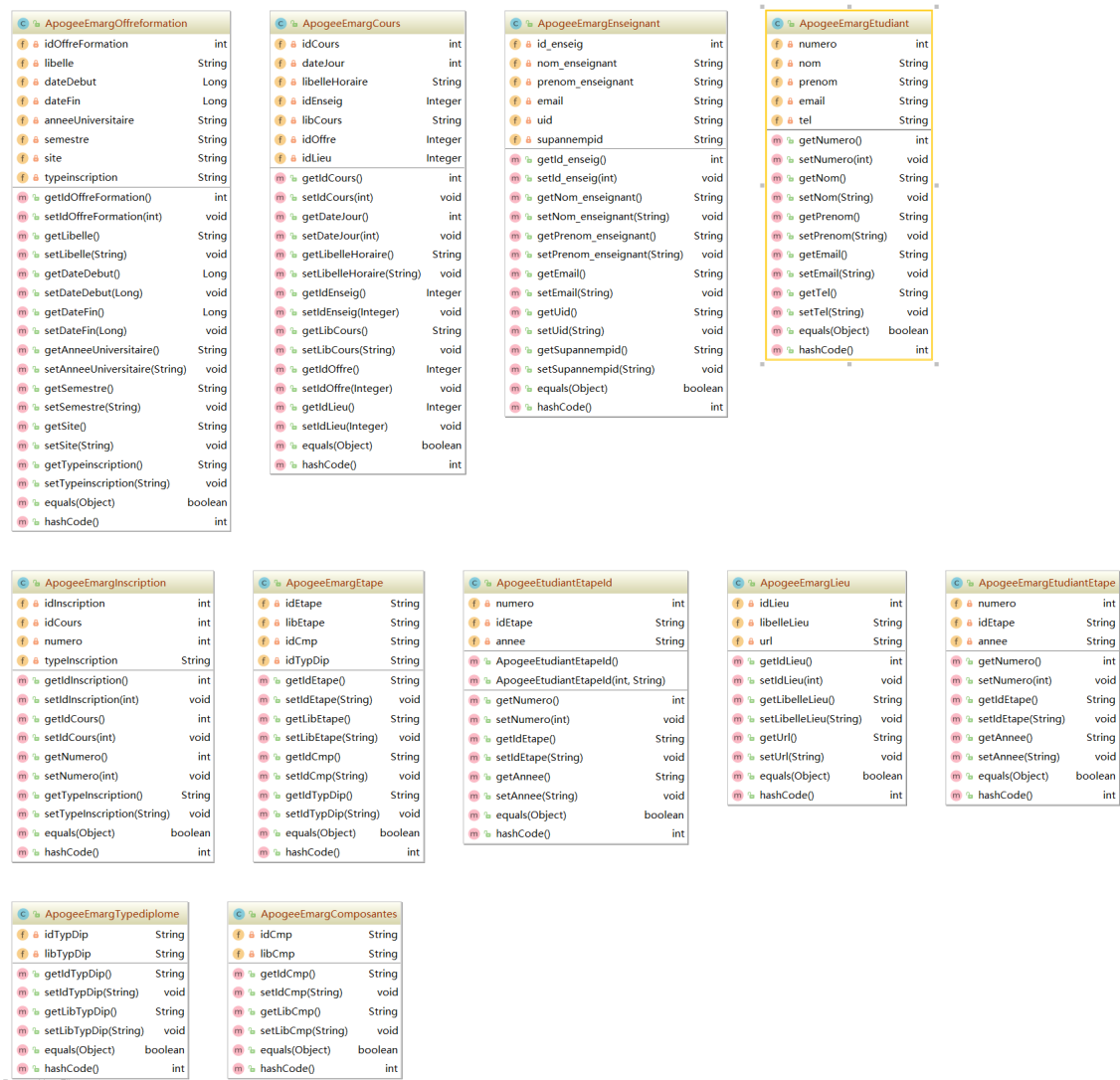


Figure 5 – Diagramme de classe de persistance.entities.apogee

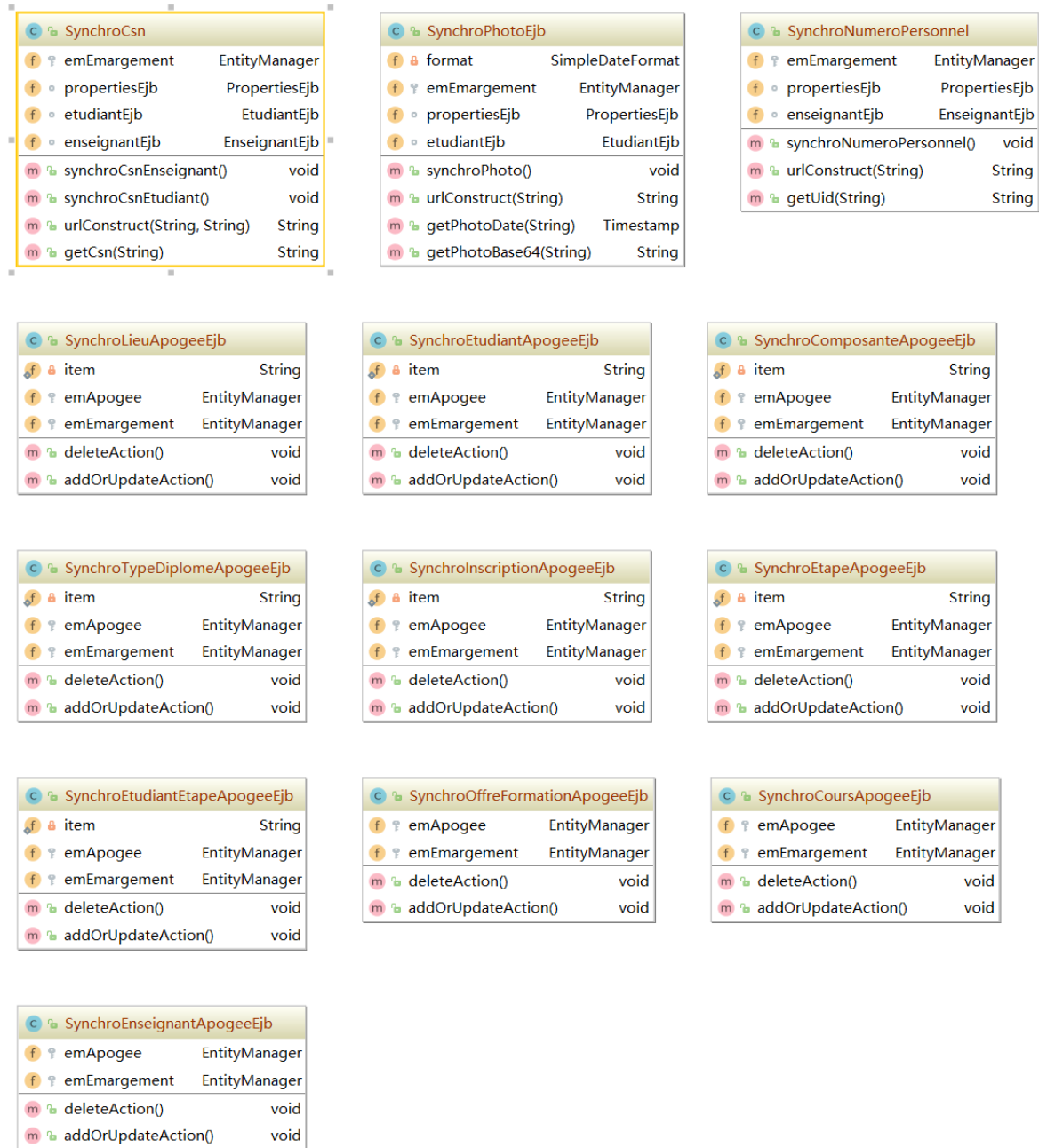


Figure 6 – Diagramme de classe de synchro.apogee

# Application d'émargement : EMA : Outil pour la gestion des présences

Yan LIU

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 pour l'université:

- Ajouter une BDD **APOGEE** de toutes les composantes de l'université. Faire la **synchronisation**.
- Ajouter et modifier des fonctionnalités pour **améliorer** l'application.



Logo Android

## Mise en œuvre

- La synchronisation entre la base de données EMA et APOGEE sur le serveur local est réalisée en simulant la structure et les données de la base de données APOGEE.
- Ajoutez et modifiez des fonctionnalités de l'application et corrigez les bogues importants.



Université de TOURS

## Résultats attendus

- L'application mobile soit utilisable dans *toute l'université*.
- L'application mobile peut connecter avec la base de données APOGEE **Réel**.
- L'interface utilisateur de l'application mobile est plus claire et plus facile à utiliser.



Logo APOGEE

# Application d'émargement : EMA : Outil pour la gestion des présences

Yan LIU

Encadrement : Pascal Makris

## Objectifs

Le projet vise à améliorer une application Android de gestion de présence pour l'université :

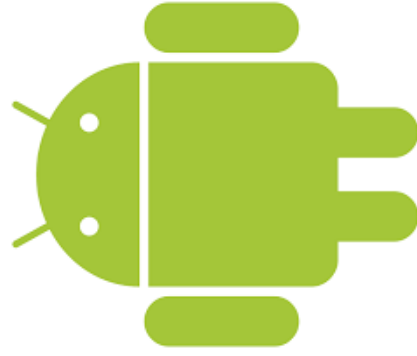
- Ajouter une BDD **APOGEE** de toutes les composantes de l'université. Faire la **synchronisation**.
- Ajouter et modifier des fonctionnalités pour **améliorer** l'application.

## Mise en œuvre

- La synchronisation entre la base de données EMA et APOGEE sur le serveur local est réalisée en simulant la structure et les données de la base de données APOGEE.
- Ajoutez et modifiez des fonctionnalités de l'application et corrigez les bogues importants.

## Résultats attendus

- L'application mobile soit utilisable dans *toute l'université*.
- L'application mobile peut connecter avec la base de données APOGEE **Réel**.
- L'*interface utilisateur* de l'application mobile est plus claire et plus facile à utiliser.



Logo Android



Université de TOURS



Logo APOGEE





# Application d'émargement : EMA

## Outil pour la gestion des présences

### 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 permet aux professeurs de sport du SUAPS de faire émargement les élèves avec leur carte étudiante pendant leurs cours. 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 à base de données APOGEE, ajouter et modifier des fonctionnalités de l'application et corriger les bogues importants, modifier le design de l'interface de l'application. Les technologies utilisées sont notamment Android pour l'application, NFC pour la communication entre l'appareil et la carte.

### Mots-clés

EMA, Synchronisation, base de données APOGEE

### Abstract

This research and development project consists in developing a mobile application for presence management using an Android device and an Atout'centre card. For this, the Android application "EMA" (Emargement Mobile Application) will be updated. This application allows sports teachers of SUAPS to emargement students with their student card during their classes. We want to extend its scope to the entire University. Thus, it is a question of changing the database with which it interacts with database APOGEE, add and edit application's functions and fix important bugs, modify the design of the interface of the application. The technologies used are in particular Android for the application, NFC for communication between the device and the map.

### Keywords

EMA, Synchronization, database APOGEE

### Entreprise

DSI de l'Université François Rabelais



### Tuteur entreprise

Malika LABANE (Ingénieure à la DSI)

### Étudiant

Yan LIU (DI5)

### Tuteur académique

Pascal MAKRI