MyESIEE

Projet de troisième année

Mai – juin 2015

APRAHAMIAN Kaïl

DUPONT Léo

KUHN Johann

HOUACINE Mehdi

Le Jour des Projets

ESIEE PARIS

une école de la CCI PARIS ILE-DE-FRANCE

INTRODUCTION **3**

BASE DE DONNEES **4**

STRUCTURE DE LA BDD 4
RECUPERATION DES DONNEES ET REMPLISSAGE DE LA BDD 4
SYNCHRONISATION DE LA BASE DE DONNEES AVEC ANDROID 5
Mise à jour des données *5*
Synchronisation des données *5*
Version de la base de données *6*

AURION **6**

Introduction

BASE DE DONNEES

Structure de la BDD

Cette partie a été réalisée par Johann Kuhn

La base de données est composée de quatre tables :

Récupération des données et remplissage de la BDD

Les données des tables `salle` et `prof` proviennent en très grande partie de documents récupérés auprès du service de la planification de l'école. Il s'agit de documents Excel regroupant des informations sur toutes les salles de l'école (épi, étage, numéro de bureau et service rattaché entre autre, mais surtout le type de salle, la surface occupée et le nom de l'enseignant à qui appartient ce bureau s'il s'agit d'un bureau).

Cependant, ces documents n'étaient pas suffisamment complets pour notre usage (présence d'un projecteur, d'une imprimante, type de tableau notamment y étaient absents) et ces documents étaient assez anciens : ils n'avaient pas été mis à jours depuis novembre 2014.

Cette partie a été réalisée par Mehdi Houacine

Nous avons choisi de tenir à jour notre propre document Excel avec ces informations manquantes sur les salles que nous sommes allé vérifier par nous-mêmes (notamment Johann) dans presque toutes les salles de l'école sur une durée approximative d'une semaine.

Suite à cela, j'ai réalisé en Python une extraction de ces données (sur les salles et sur les enseignants) en les convertissant sous un format de données standard exploitable, le `.csv`. Une fois toutes ces données stockées dans une structure Python, je les ai stockés dans notre base de données grâce à une bibliothèque Python permettant des interactions avec une base de données MySQL : `mysql-connector`. Voici des exemples de code assez simples provenant de la documentation de cette libraire montrant comment créer une table en Python dans une base de données par le biais d'instruction SQL standard : http://dev.mysql.com/doc/connector-python/en/connector-python-example-ddl.html.

Pour les enseignants, la table `prof` est constituée d'un numéro de bureau provenant des fichiers Excels cités plus tôt, leur nom provient d'une liste de noms extraites de ces documents Excel et d'ADE par le biais de sa Web API, et leur adresse mail est générée par un simple script Python selon l'expression suivante : les sept premières lettres du nom de famille suivies de la première lettre du prénom ou bien "nom.prénom" si la taille du nom est inférieure à 7 lettres puis "@esiee.fr".

Nous sommes conscients au vu de l'ancienneté de ces données qu'une minorité d'entre elles est obsolète, pour cela nous avons donc prévu un module de

participation sur l'application et le site Web afin que les étudiants qui l'utilisent puisse nous signaler les données erronées ou manquantes. Ces appréciations sont consignées dans une table de la base de données jusqu'à ce que nous les corrigions.

Synchronisation de la base de données avec Android

Cette partie a été réalisée par Léo Dupont

L'application Android utilise une copie de notre base de données principale afin de limiter la taille des requêtes HTTP et de proposer des prédictions pour les noms des salles et des profs (voir la partie Android).

Mise à jour des données

Afin de garder synchrones la base de données SQLite Android et la base de données MySQL du serveur, j'ai suivi ce tutoriel dans les grandes lignes : http://programmerguru.com/android-tutorial/how-to-sync-remote-mysql-db-to-sqlite-on-android/ bien qu'il ait nécessité beaucoup de changement afin de l'adapter au projet. De plus, nous avons ici deux tables à synchroniser et non une seule.

De plus, contrairement au tutoriel, nous ne pouvons pas stocker dans la base de données du serveur une information indiquant sur telle ligne a été synchronisée sur Android puisque notre application est destinée à être utilisée par plusieurs appareils (voir la partie suivante concernant la résolution de ce problème).

J'ai donc créé des scripts PHP permettant d'obtenir toutes les données des tables `salle` et `prof` au format JSON, en particulier le script `getData.php` qui est utilisé par l'application : https://mvx2.esiee.fr/mysql_sync/getdata.php?table=salle (pour récupérer la table `salle`).

Synchronisation des données

Afin de savoir si le contenu de la base de données d'un appareil Android correspond au contenu de la base de données du serveur, j'ai écrit un script PHP permettant d'obtenir un hash en SHA-256 de tout le contenu des tables `salle` et `prof`. De cette façon, il est facile de comparer le contenu de deux bases de données sans faire transiter un grand nombre de données par internet. Ce script est stocké dans le fichier bdd.php et accessible par cette URL :

https://mvx2.esiee.fr/api/bdd.php?func=getHashVersion

Le hash obtenu est stocké dans la mémoire du mobile à chaque fois qu'une mise à jour est effectuée. Il suffit alors de comparer le hash de version stocké dans l'appareil avec le hash de version obtenu par cette requête.

Version de la base de données

Afin de pouvoir afficher dans l'écran "À propos" de l'application la date de dernière mise à jour de la base de données principale, j'ai ajouté la fonction `getLastUpdate` au script `bdd.php`, accessible par cette URL :

https://mvx2.esiee.fr/api/bdd.php?func=getLastUpdate

Ce script se contente de récupérer la valeur de la clé `db_last_update` stockée dans la table `infos`.

AURION