Rapport Projet L3

PSI-Univ

Issameddine BERROUCHE

Morvan HAMONO

Tony BUAN

Guillaume VACHEZ

Introduction

Auteurs

- Berrouche Issameddine (RealHephaistos)
- Buan Tony (NoctTB, tbuan n'est pas un compte GitHub donc les commits n'apparaissent pas dans les statistiques)
- Hamono Morvan (MorvanH)
- Vachez Guillaume (xelphire)

Contexte

Actuellement en licence 3 informatique à l'université de Rennes 1 nous consacrons beaucoup de temps à rechercher des salles (essentiellement à l'ISTIC qui est notre lieu de travail principal). Pendant notre temps libre et dans le but de travailler, nous recherchons des salles à l'aide de l'emploi du temps qui nous est fournis. Cependant, l'ergonomie de cet outil pour trouver des salles n'étant pas la principale priorité de l'application ne permet pas une recherche simple et efficace. En effet, il faut vérifier l'emploi de chaque salle afin de déterminer si elles sont disponibles ou non.

Problématique

Nous nous sommes alors posé la question de savoir s'il n'existerait pas un moyen plus simple de chercher et vérifier qu'une salle est disponible ou non ?

Solution

Nous avons donc décidé de créer une application Android qui permettrait aux étudiants de pouvoir chercher ou de voir à l'aide d'une carte interactive qu'elles sont les salles disponibles à un moment précis. Pour cela nous avons pensé à utiliser les données de l'outil d'emploi du temps qui nous est fournis.

Description du projet

Application Android qui affiche les salles disponibles sur un plan du campus en se basant sur le planning de la fac. Cette application permettrait aux élèves de trouver plus rapidement quelles salles sont disponibles et dans quels bâtiments.

Cahier des charges

- Réaliser l'application mobile Android dans le délai de 6 semaines qui nous a été donné
- L'application doit fonctionner sur un grand nombre d'appareil (Android 6.0 Marshmallow à Android 11)
- L'application doit ce mettre à jour automatiquement avec les données de l'emploi du temps de l'ENT
- L'utilisateur doit pouvoir sélectionner un bâtiment soit en manipulant une carte interactive du campus (zoom, déplacement et bâtiments cliquables), soit en utilisant la barre de recherche ou la recherche avancée
- Quand un bâtiment est sélectionné, l'application affiche le plan du bâtiment. L'utilisateur peut sélectionner l'étage en scrollant et cliquer sur la salle de sont choix pour afficher sa disponibilité, ainsi que le prochain cours qui se déroulera dans cette salle.
- Réaliser un menu permettant d'accéder à la page d'accueil, recherche avancée et de paramètres
- Réaliser un système de recherche avancé avec le choix de pouvoir choisir une date et une heure précise
- Réaliser une page de paramètres permettant de changer la langue de l'application (Français et Anglais dans un premier temps) ainsi que le format de la date

Répartition des tâches

Séparation du groupe de 4 en 2 groupes de 2 personnes :

- 2 personnes sur la partie frontend, elle-même séparé en 2 groupes individuels
    1. Carte du campus et plan des bâtiments : Berrouche Issameddine
    2. Navigation et Menus : Buan Tony
- 2 personnes sur la partie backend, elle-même séparé en 2 groupes individuels
    1. Traitement des données et de la base de données côté serveur : Hamono Morvan
    2. Récupération des données et base de données côté client : Vachez Guillaume

Ces deux parties ont été créées afin d'être réalisées dans la période des 6 semaines disponibles avec la mise en commun comprise. En plus de ça, nous somme resté en contacte pendant toute la durée du projet, une à deux fois par semaine, afin de nous entre aider, de mettre notre travail en commun et de faire le point sur la progression de chacun.

Outils utilisés

- Discord, pour pouvoir communiquer entre tous les membres du groupe et de notre professeur tuteur à l'aide du chat vocal pendant les réunions. Avoir une trace écrite des idées / problèmes rencontrés avec le système de message. Nous n'avons pas choisi une autre plateforme car discord est une plateforme que tous les membres du groupe connaissaient et utilisaient régulièrement contrairement à d'autres plateforme.
- Android Studio, afin de réaliser le projet car il s'agit d'un des environnements de développement les plus connus et utilisé pour réaliser des applications mobiles Android.Une documentation importante et de nombreuses librairies sur internet qui nous ont permis d'avoir plus d'outils à disposition pour répondre à nos questions et aux problèmes rencontrés ou encore de ne pas écrire du code trop complexe qui nous prendrait trop de temps pour un projet de 6 semaines. Sachant que les deux seuls langages de programmation disponibles dans Android Studio sont java et kotlin, nous avons décidé d'utiliser java. En effet, il s'agit d'un langage de programmation que nous avons étudié ces trois dernières années. Nous disposons donc de base solide avec ce langage de programmation.
- Github, qui a permis de partager le code entre toutes les personnes du groupe ainsi que les professeurs. Ceci nous à aussi permis de travailler simultanément sur le projet sans devoir attendre que les personnes aient fini d'écrire leur code. Nous savions déjà utiliser la plateforme suite aux cours que nous avions eu sur Gitlab (GEN). Nous avons cependant préféré utiliser GitHub afin de garder une trace de notre travail après nos études (ne sachant pas mais supposant que nos comtes Gitlab allaient être supprimés). De plus, une extension était disponible dans Android Studio ce qui nous a permis de faire le lien entre les deux très facilement. Enfin, un système de hiérarchie très pratique grâce aux commits qui nous ont permis de retourner sur d'ancienne version en cas de problème avec le code actuel.
- SQLite, afin de pouvoir stocker les données nécessaires pour le bon fonctionnement de l'application, c'est-à-dire les données de l'emploi du temps. Nous avons décidé d'utiliser SQLite car nous savions déjà utilisé MySQL (DSB) et nous voulions pouvoir garder notre travail pour le futur. Nous n'avions pas besoin d'une table de données composés d'un nombre de données très importantes donc nous avons favorisé SQLite a SQL.
- Azure, afin de stocker la base de données en ligne et de ne pas avoir besoin de la stocker sur l'application et donc réduire la taille de celle-ci. Nous avons utilisé Azure

car cela est géré par Microsoft et qu'un compte gratuit était disponible car nous sommes étudiants. De plus des caractéristiques largement suffisantes pour notre base de données (débit,...).

Gestion du projet

Backend

Début du projet

Notre première étape à été de nous mettre d'accord sur la forme que prendrait nos donnée pour les manipuler de la même façon tout en travaillant séparément, tout cela en prévision de la mise en commun de chacune de nos partie. Nous avons donc rapidement mis au point différentes classes représentants un bâtiment, un étage, une salle et un cours. Toutes ces classes étant liée : un bâtiment contient des étages, un étages des salles et une salles des cours. Afin de facilité l'implémentation, chaque classe a des information sous forme de String comme par exemple le nom et chaque objet ce souvient des informations utiles de son parent. Ainsi une salle connaît le nom de son étage et de son bâtiment.

Récupération des ICS

Pour la récupération des fichiers .ics nous avons utilisé l'outil d'emploi du temps qui nous est fournis. Attention, nous n'avons pas eu de réponse pour l'autorisation des données. Si nous n'obtenons pas cette autorisation, l'application ne sera disponible que dans un contexte privé (en l'occurrence ce projet) afin de ne poser aucun problème.

Traitement des informations

Les fichiers en .ics n'étant que du texte avec des mots clefs suivi de valeurs comme par exemple "SUBJECT:Maths" pour exprimer que l'événement est un cours de maths,nous avons du les transformer en un format utilisable. Pour cela nous avons essayer d'utiliser des librairies déjà existante mais aucune ne semblait faire de l'extraction de donnée comme nous le voulions. Nous avons donc fait un "parser" nous même qui extrait les données en forme de String. Nous n'avions plus qu'à faire utiliser ces Strings pour créer des objets Java comme des salles ou des événements. Afin que tous les clients aient accès aux données et pour que le serveur des emplois du temps ne soit pas surcharger, nous avons décider de récupérer les données et de faire ce traitement qu'une seule fois qui envoie les données sur un serveur MySQL où tous les appareils pourraient ce connecter.

Base de donnée

La base de donnée a été élaborée grâce à un serveur wamp privé et local afin de faire des tests avant de passer sur Azure pour avoir une base de donnée commune et externe pour travaillé sur le projet dans une situation réelle. A terme, si l'application devait être "produite", nous aurions un serveur physique privé capable d'héberger ce serveur MySQL et d'effectuer les mises à jours régulièrement.

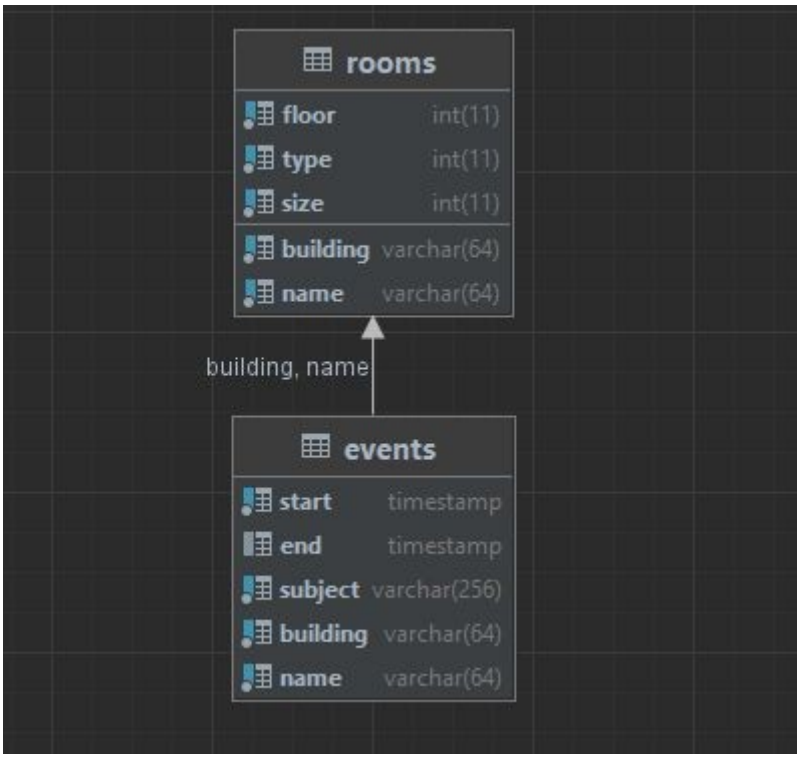

Figure 1: Schéma de la base de donnée

Base de donnée locale

Nous avions besoins de regroupé toute les informations directement sur l'appareil de l'utilisateur. Pour ce faire nous utilisons SQLite pour créer et gérer une base de donnée sous Android. La base de donnée SQLite fraîchement créé récupère la base de donnée du serveur en passant par un lien PHP hébergé sur serveur qui renvoie une String de tout l'emploie du temps que l'on doit parser pour pouvoir la re-rentrer dans la base de donnée SQLite. Ce processus de récupération se fera ensuite de façon quotidienne pour garder un emploie du temps à jour. Une fois la base de donnée SQLite complété et mis à jour, nous pouvons aller récupéré les informations dont nous avons besoin grâce à des recherches SQL nous permettant de récupérer tout un jeu de donnée ou seulement des données spécifiques comme par exemple des données correspondant à une date et une heure spécifique du calendrier.

Nous nous sommes heurtés à différentes difficultés pour faire fonctionner l'ensemble des bases de donnée comme pour aller récupérer les différents jeux de données sur le serveur externe MySQL. Nous utilisons du PHP pour pouvoir accéder à la base de donnée externe car c'est l'une des seule solution trouvé qui fonctionnait. La plupart n'était pas compatible avec ce que nous recherchions ou non entretenu et pas mis à jour depuis plusieurs années. De plus, nous devions faire attention à la durée du traitement de donnée à cause du grand jeux de donnée que nous avons. Aujourd'hui la récupération des données mais ~45 secondes pour 7500 lignes de données. Cela est sûrement encore améliorable. Nous avons du faire attention a certaine choses découlant du travail avec des calendriers, les fuseaux horaires, qui peuvent changer toutes les heures des emplois du temps, les rendant inutilisable.

Frontend

Barre de recherche

Afin de satisfaire le cahier des charges, l'utilisateur doit pouvoir chercher une salle précise de façon simple et efficace. Pour cela nous avons utilisé l'outil d'Android Studio "SearchView" qui nous a permis d'implanter la fonction de recherche. Cette barre de recherche ne fonctionne qu'avec le clic sur la salle dans le menu déroulant. Nous avons créé ce menu à l'aide de l'outil ListView d'Android Studio (identique dans les deux barres de recherche). Un filtre est disponible dans cette barre de recherche permettant une recherche rapide d'une salle précise.

Seul la barre de recherche de l'accueil possède une icône de menu qui permet d'ouvrir un menu tiroir et de naviguer entre toutes les activités possibles de l'application. Il s'agit d'une barre d'outils pour les autres activités qui va permettre la navigation.

La recherche est effectuée instantanément sur la page d'accueil mais ne l'est pas sur la page de recherche avancée. En effet, l'information reste affichée et est stockée afin de pouvoir appliquer les différents filtres à la recherche.

Menu

Il existe 2 éléments associés au menu afin de répondre aux cahiers des charges. Dans un premier temps une icône de menu permettant de faire apparaître un menu tiroir qui va permettre d'accéder aux différentes pages. Cette icône est disponible dans la barre de recherche pour la page d'accueil (comme vue précédemment). Elle est disponible dans

le coin supérieur droit des autres pages, sur la barre d'outils qui permet aussi de réaliser un retour en arrière.

Le menu tiroir qui s'affiche lorsque l'on appuie sur l'icône de menu décrite précédemment. Celui-ci peut aussi être ouvert en réalisant un scroll de gauche à droite. Différentes options apparaissent. Tout d'abord l'option accueil permettant de revenir à la page contenant le plan de l'université, la page initiale. L' option dans la catégorie lieux n'est sont pas utilisées car notre application ne traite que les données d'un bâtiment de l'université de Rennes 1. Cependant, ces options pourraient être utilisé si l'on étendait notre application à d'autres universités afin de pouvoir choisir le plan de l'université à afficher. Enfin l'option paramètre permettant d'ouvrir une page de paramètre. Toutes les options sauf les options de la catégorie lieux sont cliquables et permettent la navigation entre tous les éléments de l'application. Cette navigation a été possible avec l'utilisation d'un Drawer Layout permettant l'affichage de ce menu.

Paramètres

Les paramètres sont constitués de 2 options: Langue : permet de changer la langue de l'application entre le français et l'anglais. Sachant que la fonction utiliser pour la modifier risque d'être supprimé dans les prochaines versions ("deprecated") nous avons inclus la librairie Localisation qui devrait pouvoir la remplacer en cas de problèmes. Format : Change le format d'affichage de la date parmi trois choix possibles jj-mm-aaaa, mm-jj-aaaa, et aaaa-mm-jj. Ces modifications ne sont que visuels et n'apportent qu'un confort a l'utilisateur. Afin de réaliser ces changements nous avons utilisé les avantages des PreferencesFragment qui nous ont permis de partager les différentes valeurs sélectionnées entre tous les fichiers.

Recherche avancée

La recherche avancée est constituée d'une barre de recherche qui a déjà été évoqué précédemment. Elle contient aussi 2 autres filtres qui permettent de rechercher la disponibilité d'une salle a une heure précise, répondant ainsi au cahier des charges. Le premier filtre est un sélectionneur de temps qui fait apparaître une horloge (icône en bas à gauche de la fenêtre pour une entre manuel) afin de sélectionner l'heure désiré. Le second filtre quant à lui est un sélectionneur de date qui fait apparaître un calendrier permettant la sélection d'une date précise. Si aucun des filtres n'est utilisé alors le résultat par défaut sera le rez-de-chaussée de l'ISTIC à l'heure et date actuel.

Vue de la carte du campus

Afin de satisfaire le cahier des charges, l'utilisateur doit pouvoir zoomer et déplacer la carte du campus (idéalement avec un défilement fluide), ainsi que cliquer sur les bâtiments. La carte du campus étant trop grande et trop complexe pour être stockée sous format SVG, nous n'avons pas pu utiliser la même méthode que pour la vue des bâtiments.

Pour le déplacement de la carte, Android Studio n'offrant pas de solutions natives permettant de gérer le déplacement sur des images, nous avions initialement pensé à programmer nous-mêmes un composant permettant de zoomer et de scroller dans deux dimensions. Mais après une version bêta techniquement fonctionnelle, bien que sans mouvement fluide, nous nous sommes rendu compte qu'il serait plus facile d'implémenter la libraire PhotoView.

Pour rendre certaines parties de la cartes interactives, nous avons utilisé un outil en ligne permettant de créer des fichiers imagemap. Ce site permet de tracer des formes sur une image et génère automatiquement un fichier XML contenant les coordonnés des points du polygone. La classe MapPhotoView (notre implémentation de PhotoView) utilise ce fichier pour créer une liste de polygones. Quand le composant MapPhotoView de l'activité principale détecte un clique, il parcoure la liste des polygones pour vérifier si l'utilisateur a cliqué sur un bâtiment, et ouvre la vue bâtiment correspondante si c'est le cas.

Vue des bâtiments

La vue des bâtiments est composée de 2 RecyclerView. Le premier « LevelMapRecycler » affiche les plans des différents niveaux du bâtiment sélectionné, le deuxième « LevelNameRecycler », situé en haut à gauche affiche les numéros des Levels en fonction de l'état du LevelMapRecycler et ne peut pas être scrollé directement par l'utilisateur.

Pour rendre les salle cliquables, nous n'avons pas choisit de réutiliser la même méthode que pour la carte du campus. Créer un fichier imagemap par étage était possible mais trop laborieux. Nous avons profité du fait que les plan des bâtiment soient relativement simple pour les exporter au format SVG, pour les importer sous Android en temps que VectorDrawable. Nous avons ensuite utilisé la libraire RichPath pour pouvoir rendre certains path cliquables (ceux ayant un nom correspondant à une salle connue dans la base de donnée).

Initialement cette méthode avait un autre avantage : la librairie RichPath disposait d'une fonction permettant de changer la couleur du contenue des path dynamiquement. Cela nous aurait permis de changer la couleur des salles en fonction de leur disponibilité

afin de rendre la lecture du plan plus simple. Nous avions réussi à développer cette fonctionnalité, mais lors de nos derniers testes, et sans que nous ne touchions à la partie frontend, la fonction setFillColor de RichPath n'affichait qu'une couleur violette foncée peut importe la couleur passée en argument. Nous n'avons pas réussi à corriger ce bug à temps et nous n'avons pas gardé cette fonctionnalité dans le rendu final du projet.

La libraire RichPath n'est plus mise à jour depuis 2017 et ne fonctionne pas sur API 31, si le projet était à refaire, nous choisirions une autre approche pour la partie vue bâtiments.

Validation de l'implémentation

Afin de valider l'implémentation de notre application, nous avons testé tout d'abord la partie frontend avec les émulateurs disponibles sur Android Studio. Pour réaliser les tests sur la backend / base de données sur l'application nous avons tout d'abord créé manuellement une base de données locale puis nous avons basculé vers la base de données en ligne après avoir vérifié que les tests réalisés localement fonctionnaient. L'ensemble des tests a principalement été réalisé avec l'utilisation de l'application en se considérant comme un client / utilisateur.

Conclusion

Tony BUAN

Un projet que j'ai fortement apprécié réalisé. Un projet avec beaucoup de liberté, sur un sujet qui me plaisait et dont j'ai apprécié passer du temps dessus. N'étant pas spécialement adepte de la création de code j'ai beaucoup aimé réalisé et codé l'interface graphique de l'application grâce aux outils fournis par Android Studio. De nombreux petits détails qui posent souvent problèmes mais que j'ai trouvé satisfaisant à résoudre. Une liberté qui a cependant quelques contraintes. Comme dans tous projets il est difficile de savoir par où commencé car contrairement aux projets réalisés en cours nous ne sommes pas guidés. En conséquence, je trouve que l'on peut facilement avoir des interrogations sur l'avancement du projet. Cependant ces doutes sont souvent supprimées lors des réunions avec le professeur tuteur ce qui je trouve est une bonne chose. En effet, notre professeur nous a permis d'avoir une vision plus claire de notre application et nous a permis de définir des objectifs à réaliser toutes les semaines.

Morvan HAMONO

Le fait de pouvoir choisir le thème, la techno et les gens avec qui ont travail procure un vrai sentiment de liberté et permet de ce travaillé à fond dans quelque chose qui nous intéresse et nous appartient. Je pense d'ailleurs que cela nous a permis de travaillé plus rapidement efficacement. Le fait de devoir apprendre un langage ou un logiciel est déroutant au début car on se dit que l'on prends beaucoup de temps sans vraiment produire mais cela permet d'être en situation réelle. Sur un projet professionnel, on ne peut pas connaître le logiciel propriétaire et on passe les premières semaines de travail à se former dessus, ce projet nous a donc appris à se former nous même. La durée du projet est assez longue pour cela même si cela passe très vite et qu'il y a forcements des choses que l'on ne peut pas implémenter, on a quand même le temps d'arriver à une version fonctionnelle ou vraiment pas loin.

Guillaume VACHEZ

Ce projet aura été un réel moment de plaisir pour moi, avec ses hauts et ses bas, dans la construction comme dans la déconstruction, parce que tous ce qu'on crée ne marche pas forcement. Il est toujours agréable de créer quelque chose de ses propres mains. C'est ce que nous avons fait a travers ce projet. Le plus fastidieux a travers ce projet reste la mise en place, devoir ce familiarisé avec de nouveau concepts et outils peut rebuter au début mais une fois que le projet démarré tout ce met très vite en place. Au tout début du projet, je passais la plus part de mon temps en recherche et visionnage de tutoriel sur les différents point clé sur lesquels je devais travailler. Au final être confronté et devoir résoudre des problèmes est vraiment un point du projet que j'ai le plus apprécié même si ça en devient très vite frustrant et chronophage.