



UNIVERSITÉ DE NANTES

Rapport final

Application *ShareTaskManager*

Module « Projet de développement 2 » - X8IM081

Réalisation :

Grégoire Bejjanji
Boubaker Mentache
Laurie Nicolas
Valentin Rétiveau

Encadrants :

M. ANDRE
M. DELAHAYE

Avril 2017

Sommaire

I	Introduction.....	4
II	Présentation de l'application	5
1.1	Origine de l'idée.....	5
1.2	Cahier des charges.....	6
1	Fonctionnalités.....	6
2	Spécificités.....	6
3	Interface graphique.....	7
III	Mise en place de l'application professionnelle.....	8
1.1	Points à améliorer et fonctionnalités à implémenter	8
1	Fonctionnalités.....	8
2	Interface graphique.....	8
3	Données.....	8
1.2	Développement de l'application professionnelle.....	9
1	Modification de la base de données locale	9
2	Mise en place du serveur distant	9
3	Développement de l'API REST.....	11
4	Appel à l'API REST via Android Studio.....	13
1.3	Tests et avis d'utilisateurs.....	14

1	Tests unitaires.....	14
2	Tests d'interface utilisateur automatisés	15
3	Tests d'utilisateurs	15
1.4	Gestion des tâches et du travail de groupe.....	16
1	Planning et gestion du temps	16
2	Répartition des tâches et méthodes de travail.....	17
IV	Rétrospective de l'ensemble du projet	18
1.1	Vision globale de notre produit et retour sur expérience.....	18
1.2	Travail de groupe.....	18
1.3	Evolution entre les deux semestres.....	18
1.4	Avenir de l'application	20
V	Conclusion	22

Annexes

Tables des figures

Figure 1 - Interface de cPanel [Source : cPanel.com]	10
Figure 2 - Schéma des communications.....	11
Figure 3 - Exemple de données sous format JSON [Source : fr.wikipedia.org]	12
Figure 4 - Calendrier des périodes critiques	16

I Introduction

Dans le cadre du module de “Projet de développement”, un projet est mené sur une année scolaire, en deux phases. Le premier semestre a été consacré à la première phase du projet. Il a pour but de mieux appréhender l’environnement de développement Android Studio et le développement d’applications mobiles sous Android. L’objectif tout aussi important est la mise en place d’une stratégie de gestion de projet afin de mener à bien le développement d’une application par groupe de quatre personnes. Ainsi, un prototype avancé de l’application *ShareTaskManager* a été présenté par un rapport écrit et une présentation orale devant les encadrants du groupe.

Le second semestre est tourné vers l’enrichissement de l’application présentée au premier semestre. Cet enrichissement doit permettre au groupe de présenter une application professionnelle, mais aussi de mettre en pratique les conseils donnés au cours du bilan du premier semestre. La prise en compte des remarques des encadrants du groupe a permis de mieux gérer le projet. Ainsi, l’ensemble des membres peut faire un retour sur l’ensemble du travail effectué, concernant le développement, la modélisation, la mise en place des tâches et la gestion de projet.

II Présentation de l'application

L'application développée est nommée *ShareTaskManager*. Son objectif est de permettre à des utilisateurs de créer des tâches et de les partager au sein d'un même groupe. Ils peuvent ainsi affecter des membres de groupe à des tâches, afin que ceux-ci les effectuent.

1.1 Origine de l'idée

Durant le premier semestre, il a été demandé de trouver une idée d'application à créer. Celle-ci devait être réalisable, utile et assez ambitieuse afin que le projet dure dans le temps. En effet, une application a pour finalité d'être déployée sur un magasin d'applications en ligne, permettant à des utilisateurs de l'utiliser, de la noter, et de la recommander (ou non). Le choix de télécharger une application est fait sur la réponse aux besoins, l'innovation, l'originalité et l'aspect graphique de celle-ci. Le choix du thème de l'application devait donc présenter une réelle utilité pour de potentiels utilisateurs. Ce choix a donc été le plus crucial du projet.

Pour ce faire, il a fallu utiliser l'expérience personnelle du groupe afin de trouver une idée d'application non présentée sur le marché, ou présentée mais ne répondant pas à assez de critères pour la rendre la plus utile possible.

C'est pour cela qu'il a été fait le choix de développer une application de gestion de tâches. En tant qu'étudiants, peu d'applications sont proposées sur le système Android pour gérer des tâches au sein d'un groupe d'élèves ou au sein d'une classe. Il a donc été effectué une étude par rapport aux différentes applications proposées et aux besoins auxquels répondre à travers l'application.

Il a ainsi été mis en place un cahier des charges afin de présenter l'application à développer, sans ambiguïté, afin de faciliter sa présentation auprès des encadrants, et au sein du groupe. Cela permet d'avoir un fil conducteur au fur et à mesure du développement. Y sont alors présentées les fonctionnalités de base de l'application telle que l'ajout de tâches ou la modification de tâches, et d'autres fonctionnalités plus complexes telles que la gestion de groupes de membres et l'affectation de tâches à plusieurs utilisateurs.

1.2 Cahier des charges

Le cahier des charges présenté ici est très simplifié et permet de présenter clairement les fonctionnalités que doit présenter l'application *ShareTaskManager* afin d'être viable.

1 Fonctionnalités

L'application permet de gérer trois entités : des utilisateurs, des groupes et des tâches. Les tâches sont affectées à des personnes, qui doivent les réaliser ou qui sont directement concernées par celles-ci. Chaque utilisateur fait partie d'au moins un groupe, nommé « Personnel ». Celui-ci regroupe toutes les tâches qu'un utilisateur souhaite créer pour lui-même. Bien sûr, un utilisateur peut créer autant de groupes qu'il le souhaite. Pour affecter une personne à une tâche, il faut que ces deux entités soient rattachées au même groupe. Voici les fonctionnalités qui doivent être disponibles sur l'application :

- ❖ Inscription d'un utilisateur,
- ❖ Connexion d'un utilisateur,
- ❖ CRUD¹ sur les tâches, les utilisateurs, les groupes,
- ❖ Accès au groupe « Personnel » dès l'inscription d'un utilisateur,
- ❖ Possibilité de personnaliser le tri des tâches (par priorité, par état, par date d'échéance par exemple),
- ❖ Affectation d'une tâche à un utilisateur autre que l'utilisateur courant de l'application,
- ❖ Ajout d'un membre dans un groupe dont l'utilisateur courant est administrateur,
- ❖ Gestion des membres d'un groupe par l'administrateur de celui-ci

2 Spécificités

L'application implique l'utilisation d'une base de données. Android Studio intègre de base la gestion d'une base de données SQLite. Ce SGBD permet de gérer des bases de données embarquées sur les terminaux Android (ici). Ainsi, l'application *ShareTaskManager* stocke l'ensemble des données sur le terminal utilisé. Cependant, cela ne fonctionne qu'en local.

¹ CRUD (Create – Read – Update – Delete) : Façon de nommer l'ensemble des quatre opérations basiques effectuées sur des objets. Ces quatre opérations sont la création, la lecture (consultation des informations), la modification et la suppression.

Afin de proposer l'ensemble des fonctionnalités du cahier des charges aux utilisateurs, il faut proposer une application qui accède à un serveur distant. Cela permet de centraliser l'ensemble des données sur un même serveur. Un utilisateur peut ainsi avoir accès aux tâches d'un groupe même si elles ne lui sont pas affectées par exemple.

Mais l'application ne peut pas fonctionner que sur serveur. En effet, si l'utilisateur souhaite consulter l'ensemble des tâches affectées à son groupe sans avoir de connexion Internet, il doit être en mesure de le faire.

Il a donc été fait comme choix d'avoir une synchronisation entre la base de données sur serveur et la base de données en local. Lorsque l'utilisateur ne possède pas de connexion Internet sur son terminal, il passe en mode « Consultation » et ne peut effectuer aucune modification. Lui permettre la modification de tâches aurait rendu la tâche du développement plus longue, et l'application n'aurait pas pu être livrée dans les temps demandés. Ainsi, l'utilisateur peut effectuer des modifications lorsqu'il est connecté à Internet. A chaque ajout, modification ou suppression, les tâches sur le serveur sont mises à jour automatiquement puis la base de données en local est modifiée en conséquences.

L'implémentation de l'API et la gestion de la base de données sur un serveur distant sont traités dans le paragraphe 1.4 de la section III.

3 Interface graphique

Comme énoncé précédemment, l'aspect graphique d'une application est très important pour l'utilisateur. Il a donc été décidé de créer des modèles d'interface graphique de l'application pour trouver le meilleur agencement afin d'attirer les utilisateurs potentiels. Ces modèles d'interfaces graphiques sont disponibles en annexe (*Cf Annexe 1 - Interfaces graphiques de l'application ShareTaskManager*).

III Mise en place de l'application professionnelle

Durant ce second semestre, il faut faire passer l'application rendue au premier semestre de prototype à application professionnelle. Pour ce projet, une application professionnelle est une version fonctionnelle, pratique et testée.

1.1 Points à améliorer et fonctionnalités à implémenter

A la fin du premier semestre, un prototype bien avancé de l'application *ShareTaskManager* est rendu. Ce prototype présente un stockage des tâches dans une base de données en local. Il n'y a aucun accès à une base de données sur serveur donc l'utilisateur n'a pas besoin de s'authentifier pour utiliser l'application. La gestion des groupes n'est donc pas non plus nécessaire. Seul le groupe « Personnel » est implémenté à la création de l'application. Ainsi, l'ensemble des tâches de l'utilisateur sont rattachées à ce groupe. Les tâches sont affichées grâce à un code couleur qui présente l'urgence de la tâche, et une icône présentant l'état de la tâche (en cours, validée, en retard).

1 Fonctionnalités

Afin de répondre aux besoins du cahier des charges, il est donc important de mettre en place une communication entre l'application et un serveur distant contenant une base de données. Ceci implique l'implémentation de la gestion des groupes et des utilisateurs, notamment avec une authentification de la part de celui-ci lorsqu'il démarre l'application.

2 Interface graphique

Concernant l'interface graphique, celle-ci semble correspondre aux attentes d'utilisateurs potentiels selon un sondage effectué dans notre entourage. Il a donc été établi de continuer sur ce modèle (*cf Annexe 1 – Interfaces graphiques de l'application ShareTaskManager*).

3 Données

Une revue des besoins implique la modification de la base de données locale. Il faut donc prendre en compte le fait qu'une table *MembreGroupe* est nécessaire. Cette table permet de lier des utilisateurs à des groupes. En effet, les groupes permettent à des utilisateurs membres de ceux-ci de partager des tâches communes. Chaque tâche étant liée à un membre du groupe. Le nom a été choisi car « *UtilisateurGroupe* » (concaténation du nom des deux tables à lier) aurait pu porter à confusion et ne pas présenter le réel objectif de cette table. Ici, il s'agit bien d'une table

permettant de rassembler les membres des groupes. Ainsi, si le membre d'un groupe souhaite affecter une tâche à un autre utilisateur, l'application peut directement chercher dans la base de données locale si cet utilisateur est membre du groupe concerné ou non. Cela permet un traitement de la tâche plus rapide que s'il fallait appeler le serveur distant. De plus, l'utilisateur aura ainsi accès à l'ensemble des groupes desquels il fait partie.

1.2 Développement de l'application professionnelle

1 *Modification de la base de données locale*

Comme énoncé précédemment, une nouvelle table, MembreGroupe, a été implémentée dans l'application. Elle permet ainsi à l'utilisateur d'avoir l'ensemble des groupes qu'il a rejoins. Il peut aussi maintenant connaître l'identifiant des utilisateurs faisant partie des mêmes groupes que lui.

Un attribut « dateDerniereModification » sur la table « taches » permet de savoir à quel moment la tâche a été modifiée pour la dernière fois.

2 *Mise en place du serveur distant*

Afin de faire fonctionner l'application avec une base de données distante, il a fallu trouver un serveur distant sur lequel héberger la base de données. Par définition, un serveur distant est un serveur auquel on accède via Internet. Leur utilisation nécessite des frais, et en fonction de la disponibilité du serveur et de sa capacité, les frais peuvent être plus ou moins importants. Afin de rendre notre application la plus disponible possible, il fallait donc avoir un serveur disponible à toute heure, pour lequel il fallait régler des frais mensuels.

Par exemple, l'hébergement le moins cher trouvé coûtait 35€ par an, mais certains coûtent 50€ par mois. Cela dépend donc du service, de la personnalisation et de la disponibilité des serveurs que nécessite l'utilisation de ceux-ci.

Une personne de l'entourage d'un des membres du groupe possède elle-même des accès à un serveur. Cette personne a donc proposé de créer un accès au groupe afin d'héberger leur base de données en ligne, sachant que l'application était plus dédiée à la réalisation d'un projet pédagogique qu'en la commercialisation de celle-ci.

L'administration de ce serveur est effectuée par le service CPANEL. Cet outil permet entre autres de créer une base de données MySQL et d'y stocker un grand nombre de données. La base de données a été créée durant le premier semestre, en suivant le diagramme de classes établi lui aussi au début du projet. La gestion de la base de données est effectuée via PHPMyAdmin, un outil déjà appréhendé durant les années d'étude précédentes de l'ensemble du groupe.

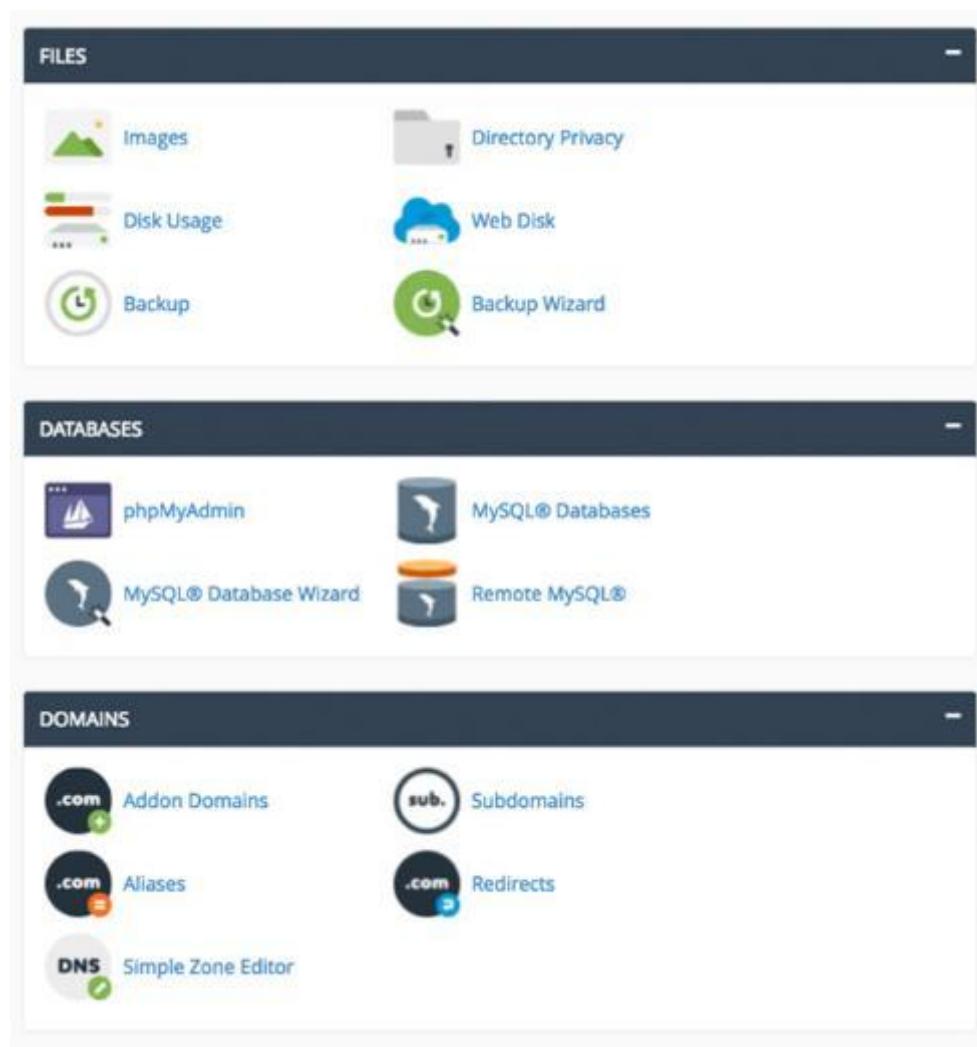


Figure 1 - Interface de cPanel [Source : cPanel.com]

3 Développement de l'API REST

Pour que l'application Android puisse récupérer les données stockées sur le serveur distant, il faut créer un lien, une interface entre les deux.

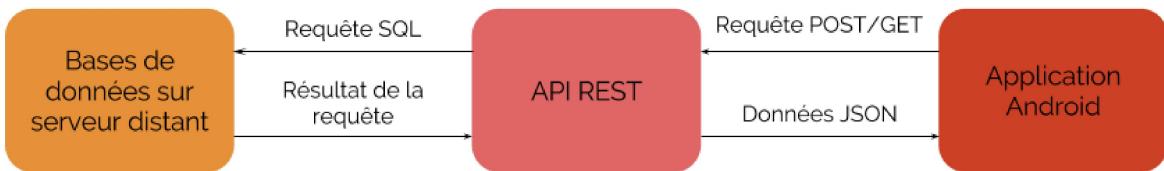


Figure 2 - Schéma des communications

Pour cette interface, il a donc fallu développer une API. « *Une API (Application Programming Interface) est un ensemble normalisé de classes, de méthodes ou de fonctions qui sert de façade par laquelle un logiciel offre des services à d'autres logiciels* » [Source : fr.wikipedia.org]

Choix du langage de développement de l'API

Durant le cursus qu'ont suivi les membres du groupe, ceux-ci ont déjà implémenté des fonctionnalités d'accès, de connexion à une base de données et de traitements liés à celle-ci en PHP. Il a donc été naturel de choisir ce langage pour le développement de l'API.

Choix du type de l'API

Il existe de nombreux types d'API, il a donc fallu effectuer un choix quant au type d'API à implémenter. La variété de documentation et la popularité de celui-ci a été un réel critère de choix afin de développer au plus vite et le plus facilement possible. La création d'une API n'étant pas l'objectif final du projet. Le groupe s'est donc arrêté sur les API REST, qui sont très polyvalentes, assez simples à développer pour ce type d'application, et très utilisées.

API RESTful et contenu de l'API

Les principes d'une API RESTful ont été théorisés par Roy Fielding² dans une thèse³. Une API RESTful présente les caractéristiques suivantes.

² Roy Fielding est un informaticien américain. Il est membre fondateur de la fondation Apache, a participé au développement des premiers serveurs Web et a effectué des recherches sur le fonctionnement du Web.

³ Cette thèse est disponible ici : http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm

- ❖ Sans état : la communication doit être sans état, chaque demande de client doit contenir toutes les informations nécessaires pour comprendre la demande. L'état de session est donc maintenu entièrement sur le client.
- ❖ Cacheable : Les données contenues dans une réponse à une demande sont étiquetés comme cacheable ou non cacheable. Si une réponse est cacheable, le client peut réutiliser celle-ci au lieu de refaire appel à l'API.
- ❖ Orientée client-serveur : Il y a une séparation des préoccupations entre le client et le serveur. Ainsi, le client s'occupe du stockage des données et le serveur s'occupe de les lui fournir.
- ❖ Avec une interface uniforme : C'est la principale différence entre le REST et les autres types d'API. Les informations sont transférées suivant une norme généralisée, ce qui permet une meilleure maintenance et évolutivité du code.
- ❖ Avec un système de couche : Ceci permet d'avoir une architecture composée de couches hiérarchiques en limitant le comportement des composants tels que chaque composant ne peut pas « voir » au-delà de la couche immédiate avec laquelle il interagit.

L'API présente ainsi deux fichiers principaux :

- ❖ dbHandler.php : Requêtes sur la base de données, accès à celle-ci et renvoi du résultat
- ❖ index.php : Traitement de l'ensemble des requêtes API, renvoi des résultats en format JSON.

Chaque valeur de retour est un tableau JSON, formé de la manière suivante :

```
{
  "menu": {
    "id": "file",
    "value": "File",
    "popup": {
      "menuitem": [
        { "value": "New", "onclick": "CreateNewDoc()" },
        { "value": "Open", "onclick": "OpenDoc()" },
        { "value": "Close", "onclick": "CloseDoc()" }
      ]
    }
  }
}
```

Figure 3 - Exemple de données sous format JSON [Source : fr.wikipedia.org]

Tests de l'API

Afin de débugger et tester l'API avant de créer les appels à celle-ci dans Android Studio, il faut les tester. Pour tester une API, de nombreux outils sont disponibles. Le plus intuitif trouvé est Advanced Rest Client. Chaque appel à une URL peut être paramétré. Des formulaires sont prévus pour envoyer des paramètres à l'API, comme la méthode d'appel (POST ou GET). Un exemple de test de l'API via l'outil Advanced Rest Client est présenté en Annexe 2.

Pour rappel, les méthodes d'appel GET et POST sont bien différentes et la compréhension de deux est importante pour développer une API. A savoir que ces méthodes sont des méthodes d'accès définies dans le protocole http.

GET : Cette méthode est utilisée lorsque la requête n'entraîne aucune modification de données (lecture seule). L'ensemble des données du formulaire sont encodées dans une URL.

POST : Un en-tête et un corps de message sont envoyés au serveur. En général, le corps du message est constitué des données du formulaire envoyé par le client. Aucune de celles-ci n'apparaissent dans l'URL. Elle est recommandée, par le W3C⁴, lorsque la requête doit modifier la base de données ou pour le transfert de données sensibles.

4 Appel à l'API REST via Android Studio

Une fois l'API REST développée et testée, il faut implémenter l'appel à l'API. L'application Android est donc qualifiée de « cliente ». De nombreux outils sont proposés pour effectuer des appels à une API à partir d'Android Studio tels que :

- ❖ Volley, une librairie développée par Google en raison d'une absence dans le SDK d'Android, d'une classe réseau capable de fonctionner sans interférer avec l'expérience utilisateur
- ❖ RESTDroid, une librairie développée par PCreations afin de gérer les appels REST
- ❖ RoboSpice, basée sur le module Android Spring
- ❖ Retrofit, développée par Square, présente une documentation simple et intuitive, une gestion des objets tout aussi rapide une fois la notation assimilée

⁴ W3C (World Wide Web Consortium) : Organisme de standardisation à but non lucratif, fondé en octobre 1994 chargé de promouvoir la comptabilité des technologies du World Wide Web. [Source : fr.wikipedia.org/]

Après avoir étudié la documentation de chacun de ces outils, il a été établi d'utiliser Retrofit, une librairie développée par Square, un éditeur d'outils open source. Cet outil a paru être le plus simple d'utilisation.

Afin d'utiliser Retrofit, il faut implémenter deux classes :

- ❖ STMAPI : déclaration de l'URL de base de l'API et de la méthode getClient() qui instancie un Builder d'objet GSON et un Builder d'objet Retrofit,
- ❖ ApiInterface : interface présentant l'ensemble des URL que l'application peut appeler, des méthodes liées et des paramètres à leur fournir.

A chaque appel au serveur, une instance de STMAPI est créée, afin de construire les traducteurs d'objets GSON⁵ et l'objet Retrofit.

1.3 Tests et avis d'utilisateurs

1 Tests unitaires

Les tests unitaires permettent de valider le code source quant à sa conformité avec les spécifications de celui-ci. Pour chacune des classes à tester, une nouvelle classe de test est implémentée. Ce type de tests s'effectue grâce à des outils, notamment JUnit. Cet outil est un framework⁶ de tests unitaires dédié au langage de programmation Java.

Chaque méthode est ainsi testée via une méthode de test qui lui est propre. Celle-ci prend des paramètres en compte, crée les instances nécessaires à son exécution, et vérifie, au travers d'une assertion, que le résultat obtenu correspond à celui attendu. L'exécution du code source est ainsi en adéquation avec les spécifications de ce dernier.

Ce type de tests est donc intéressant pour les systèmes qui effectuent des traitements et des calculs. Notre application ne réalisant que des appels au serveur, et des traitements sur la base de données, l'ensemble du groupe a jugé que la mise en place des traitements unitaires aurait coûté un temps supérieur à l'efficience de ce type de tests sur le code source.

⁵ GSON : Librairie Java utilisée pour convertir des objets Java en JSON et vice versa.

⁶ Framework : En français, « cadre d'application » définit un ensemble de composants logiciels servant à créer les grandes lignes d'une partie d'un logiciel. Un framework est générique, et guide l'utilisateur à respecter certains patrons de conception. [Source : fr.wikipedia.org]

2 Tests d'interface utilisateur automatisés

Pour une application mobile, il est important que l'interface utilisateur soit conforme aux spécifications du système. Pour s'en assurer, il est possible d'automatiser des tests d'interface utilisateur, au lieu de les faire à la main.

Pour cela, de nombreux outils sont mis à la disposition des développeurs. Cependant, ayant très peu de temps pour terminer l'application, le groupe a préféré privilégier les tests utilisateurs non automatisés. En effet, il était plus simple pour les membres du groupe de proposer à des personnes de leur entourage de tester l'application, et de recueillir directement les avis de ces testeurs. Ainsi, la mise en place des tests ne coûte rien par rapport à la mise en place d'un outil informatique.

Cependant, si le groupe avait pu disposer de plus de temps, des outils automatisés auraient été bénéfiques pour en apprendre plus sur les tests et valider les spécifications de l'application de manière plus formelle.

3 Tests d'utilisateurs

Conformément aux choix du groupe présentés précédemment, des pratiques de tests via des utilisateurs potentiels de l'application ont été mise en place. Le fichier APK⁷ a été mis à disposition de ces testeurs. Un formulaire a été créé via l'outil Google Forms. Ainsi, les testeurs peuvent répondre à un questionnaire et noter l'application (de 1 à 5) sur les points suivants :

- ❖ Utilité de l'application,
- ❖ Aspect graphique,
- ❖ Fluidité,
- ❖ Facilité d'utilisation,
- ❖ Fiabilité (crash, bugs, ...),
- ❖ Note générale à attribuer à l'application

Les testeurs peuvent aussi ajouter des commentaires libres afin de faire remonter les problèmes rencontrés lors de l'utilisation de l'application, les mauvais et les bons points de celle-ci.

⁷ APK : Format de fichier (.apk) que l'on dit installable. Les fichiers APK sont en effet des fichiers permettant d'installer une application sur un terminal Android. [Source : android-france.fr]

L'application étant lancée en tests le week-end des 1^e et 2 Avril, les résultats aux tests utilisateurs seront présentés lors de la présentation orale du projet.

1.4 Gestion des tâches et du travail de groupe

L'objectif principal de ce module était de se former à la réalisation d'un projet sur une période plus longue que sur d'autres projets pédagogiques. Cela impose obligatoirement d'avoir une organisation clairement définie entre les membres du groupe pour être efficaces. Pour cela, la même ligne conductrice que le premier semestre a été gardée concernant la gestion de projet, tout en essayant de corriger les points négatifs qui avaient pu ressortir lors de la première partie du projet.

1 Planning et gestion du temps

L'un des principaux problèmes mis en avant était l'identification des contraintes extérieures qui pouvaient ralentir le rythme de développement du projet. Par exemple, les contraintes liées aux rendus de projets, les contrôles continus et les périodes d'exams en fin de semestre. Un calendrier a été mis en place (*voir schéma ci-contre*) afin de représenter les phases où le groupe avait plus de temps libre à passer sur le projet (en vert), et les phases dites « critiques » (en rouge), où il fallait se concentrer sur d'autres travaux. Effectivement, la période de Janvier-Février offrait peu de temps libre comparée à la période du mois de Mars. En effet, cette période peu chargée en cours permet au groupe de s'adonner au développement de l'application. L'ensemble du calendrier prévisionnel a été effectué en parallèle à ce calendrier afin de prendre en compte ces périodes.

Ce point avait été le plus gros problème rencontré durant le premier semestre et les encadrants du groupe ont permis d'appréhender une nouvelle façon de réaliser un calendrier prévisionnel en prenant en compte ces contraintes.

	Janvier	Février	Mars	Avril
1 D		1 M	1 M	1 S
2 L	2 J	2 J	2 D	
3 M	3 V	3 V	3 L	
4 M	4 S	4 S	4 M	
5 J	5 D	5 D	5 M	
6 V	6 L	6 L	6 J	
7 S	7 M	7 M	7 V	
8 D	8 M	8 M	8 S	
9 L	9 J	9 J	9 D	
10 M	10 V	10 V	10 L	
11 M	11 S	11 S	11 M	
12 J	12 D	12 D	12 M	
13 V	13 L	13 L	13 J	
14 S	14 M	14 M	14 V	
15 D	15 M	15 M	15 S	
16 L	16 J	16 J	16 D	
17 M	17 V	17 V	17 L	
18 M	18 S	18 S	18 M	
19 J	19 D	19 D	19 M	
20 V	20 L	20 L	20 J	
21 S	21 M	21 M	21 V	
22 D	22 M	22 M	22 S	
23 L	23 J	23 J	23 D	
24 M	24 V	24 V	24 L	
25 M	25 S	25 S	25 M	
26 J	26 D	26 D	26 M	
27 V	27 L	27 L	27 J	
28 S	28 M	28 M	28 V	
29 D		29 M	29 S	
30 L		30 J	30 D	
31 M		31 V		

Figure 4 - Calendrier des périodes critiques

Le diagramme de Gantt disponible en Annexe 4 présente l'ensemble des répartitions prévisionnelles et effectives des tâches entre les membres du groupe.

2 Répartition des tâches et méthodes de travail

Comme énoncé précédemment, la répartition des tâches a été effectuée de la même manière que lors du premier semestre. La liste des tâches à effectuer est faite en groupe lors d'une réunion hebdomadaire. Chaque tâche est distribuée en fonction des compétences et des envies de chacun. La semaine suivante, une autre réunion hebdomadaire permet de faire le point sur les problèmes rencontrés, l'avancement de chacun et les futures tâches à effectuer.

Une nouvelle pratique de développement a cependant été adoptée. En effet, lors d'une réunion de compte-rendu avec les encadrants du groupe, ceux-ci ont prodigué la mise en place du *Pair Programming*. Le *Pair Programming*, ou *Programmation en Binôme* en français, est une méthode de travail où deux personnes développent en même temps. Ayant un créneau réservé au travail du projet du mercredi une fois par semaine, la mise en place de cette technique de travail a été effectuée durant cette journée du mercredi, les deux semaines avant la fin du projet. Cela s'est fait tard, mais a permis au groupe d'appréhender cette nouvelle technique pour ce projet. A savoir que de nombreux travaux pédagogiques impliquent cette méthode de travail, l'ensemble du groupe était donc familier avec cette façon de développer.

Pour le reste du temps, le groupe a décidé de mettre en place des binômes, comme au premier semestre. Ainsi, les deux personnes du binôme travaillent sur des points différents mais chacun des deux connaît ce que développe l'autre élément du binôme. Ainsi, si une des deux personnes rencontre un problème, l'autre personne est capable de lui venir en aide.

IV Rétrospective de l'ensemble du projet

A la fin du semestre, l'application compte un bon nombre de fonctionnalités. Cependant certains aspects ne sont pas encore traités, faute de temps et de connaissances dans le sujet. En effet, plus de temps aurait permis au groupe de se former à la gestion de la sécurité pour les bases de données, les appels entre l'API et l'application Android, et l'application en elle-même.

1.1 Vision globale de notre produit et retour sur expérience

L'application *ShareTaskManager* présente aujourd'hui les fonctionnalités prévues dans le cahier des charges. Comme énoncé précédemment, l'aspect sécurité n'est cependant pas traité, faute de temps.

Le développement de cette application a présenté un grand nombre d'éléments sur lesquels il a été appris de nombreux processus. La mise en place d'une API REST et la gestion d'une base de données distante a permis d'aborder des points non vus en cours durant le cursus. L'application présentant ainsi des appels à l'API, il a été important d'apprehender le fonctionnement d'une librairie permettant ces appels.

L'application présente donc un réel objectif pédagogique, et a permis aux membres du groupe d'aborder un très grand nombre de notions.

1.2 Travail de groupe

Le travail de groupe s'est bien déroulé, mais la constitution du groupe aurait pu bénéficier d'une plus grande diversité des profils, notamment concernant les compétences en développement. Cependant, la cohésion du groupe est restée intacte au fur et à mesure du projet. Chacun a su prendre en compte les remarques effectuées à la fin du premier semestre.

1.3 Evolution entre les deux semestres

Le tableau des compétences établi lors du rapport du premier semestre a aidé l'ensemble des membres à monter en compétences au long du développement de l'application. Afin de faire le bilan, un second tableau a été effectué en fin de projet pour le second semestre. Ce tableau est créé et rempli par l'ensemble des membres du groupe, qui peuvent apporter des éléments sur les autres membres du groupe. Ainsi, chaque personne a un avis extérieur sur ses montées en compétences, ce qui présente un réel attrait.

Les deux tableaux de compétences (celui du premier semestre et celui du second), sont présentés en Annexe 3, afin de pouvoir les comparer.

Au début du semestre, des objectifs ont été fixés à chaque membre du groupe, afin d'améliorer leur efficacité. Ces objectifs concernent le développement mais aussi la gestion de projet et l'ensemble des compétences liées à un projet (communication, rédaction, écoute par exemple).

En fin de semestre, un autre bilan est effectué. Après comparaison de ces deux bilans, il est conclu que cette pratique a permis aux membres du groupe de se concentrer chacun sur ses objectifs et de mettre en place des outils afin de les atteindre. Ainsi, la plupart des objectifs ont été atteints, même s'il reste bien sûr de nombreuses choses à améliorer au cours de l'expérience du groupe.

Pour atteindre ces objectifs, une décision de groupe a été effectuée. En effet, pour que chacun soit plus impliqué dans le projet, il a été décidé d'envoyer chaque semaine un compte-rendu rédigé par une personne différente. Cela permet aux membres du groupe de faire un bilan de la semaine et de s'intéresser à ce que chacun produit pour le projet. La rédaction de ce compte-rendu permet donc de mettre en œuvre des compétences de gestion de projet, d'écoute d'autrui, de communication mais aussi de synthèse afin que les encadrants soient en connaissance du travail effectué.

Ensuite, chaque membre du groupe a pris des décisions individuellement afin d'atteindre ses objectifs.

Pour améliorer ses compétences en maîtrise des outils utilisés, Boubaker a approfondi ses connaissances en suivant des formations en ligne (vidéos tutoriels) et la documentation Android, plus en profondeur.

Grégoire présentait des lacunes en gestion de projet, il a donc décidé d'écouter plus les autres membres du groupe afin de connaître davantage les problèmes qu'ils rencontrent et le travail qu'ils effectuent. Il s'est aussi grandement amélioré en rédaction, notamment grâce à l'écriture de comptes-rendus hebdomadaire.

Afin d'être plus efficace dans son développement, Laurie a profité de la mise en place du *Pair Programming* afin d'apprendre de nouvelles méthodes de développement et d'être ainsi plus efficace. Concernant la gestion du temps, qui présente des lacunes, il a été utilisé des tableaux afin d'avoir une vision plus générale du projet. Enfin, afin d'améliorer ses compétences en recherche de documentation, Laurie a consulté ses camarades de classe afin d'en apprendre plus

sur la recherche. Cela a grandement amélioré son efficacité, notamment pour la recherche de solution d'un problème.

Enfin, Valentin devait améliorer ses compétences en gestion de projet, communication et efficacité. De la même manière que Laurie, Valentin a lui aussi tiré profit de la mise en place du *Pair Programming* pour améliorer son efficacité dans le développement. Une meilleure gestion de son temps en suivant des calendriers a aussi permis à Valentin d'améliorer son efficacité et ses compétences en gestion de projet. La communication de Valentin par rapport aux membres du groupe s'est aussi améliorée au fur et à mesure du projet.

1.4 Avenir de l'application

La consigne première de ce projet était le développement d'une application professionnelle, pensée par l'ensemble du groupe. A la fin de ce semestre, l'application est fonctionnelle, mais présente quelques lacunes qui ne permettent pas à celle-ci d'être présentée sur un magasin d'applications à disposition d'utilisateurs.

Chaque membre partira en stage et aura du travail à effectuer pendant la période qui va suivre. De plus, certaines personnes ne seront pas souvent présentes à Nantes pour le week-end. Cela ne permet pas au groupe de continuer à développer l'application. De plus, aucun des membres ne voit son avenir professionnel dans le développement d'applications mobiles. Il a donc été décidé de laisser l'application telle qu'elle est. Si un des membres du groupe souhaite, à l'avenir, revenir sur cette application et continuer de la développer, il pourra bien sûr en informer ses camarades afin de prendre une décision commune quant à l'avenir de cette application.

De plus, chacun des membres sait que cette application pourrait servir, notamment à des étudiants, mais le manque de temps ne permet pas, pour l'instant, de la développer et de la maintenir, ce qui oblige les membres du groupe à être disponibles, même après (et surtout après) le développement de l'application.

Si un ou plusieurs des membres du groupe venaient à continuer l'application, de nombreuses fonctionnalités seraient intéressantes, en plus de la gestion de la sécurité :

- ❖ Retrouver son mot de passe si l'utilisateur l'a oublié,
- ❖ Envoyer des notifications par email et sur le terminal à l'utilisateur lorsqu'une tâche vient à échéance, qu'il est ajouté à un groupe, ou toute autre opération nécessitant une notification
- ❖ Afficher l'ensemble des membres d'un groupe quand on est sur le détail de ce groupe,
- ❖ Afficher directement sur la liste des tâches les personnes affectées à celle-ci,
- ❖ Revoir un affichage plus fluide de l'application,
- ❖ Joindre des fichiers PDF à des tâches,
- ❖ Pouvoir générer un historique des tâches avec des dates d'exécution prévisionnelles et effectives

Si l'application est amenée à être commercialisée sur un magasin d'applications, il sera aussi important de s'informer sur les aspects juridiques et commerciaux de cet acte.

V Conclusion

Ce rapport constitue un des derniers travaux de ce projet. Une présentation orale fera ainsi un complément au présent document afin d'exposer l'ensemble du travail effectué sur ce semestre et les enseignements tirés pour l'ensemble du groupe.

Concernant la montée en compétences fonctionnelles apportée par ce projet, les membres du groupe sont tous satisfaits du niveau d'autonomie, et des éléments appris tout au long du développement. En effet, ces éléments concernent de nombreux domaines tels que le développement Java et Android, la gestion de bases de données, et le développement d'une API REST. De plus, des compétences en gestion de projet ont été nécessaires pour mener à bien le développement de l'application. Les conseils des encadrants et l'application de ces conseils ont permis aux membres de s'améliorer concernant ces compétences organisationnelles.

Le travail de groupe s'est passé relativement facilement, avec une aisance à communiquer pour chacun des membres du groupe croissante au fur et à mesure du développement du projet. Chacun a pu, jour après jour, faire preuve de remise en question sur ses méthodes de travail. Ceci a permis à chacun de monter en compétences, notamment dans les points les plus critiques présentés à la fin du premier semestre.

Enfin, ce travail a été bénéfique pour les quatre membres du groupe, d'un côté relationnel et fonctionnel. Chacun arrive maintenant à se remettre en question et apprend de plus en plus à se fixer des objectifs afin de s'améliorer au fil de son expérience.

Annexe 1 - Interfaces graphiques de l'application ShareTaskManager



Affichage des tâches d'un utilisateur



Affichage des groupes d'un utilisateur



Affichage des tâches pour un groupe choisi



Menu de l'application

Annexe 2 - Fonctionnement des tests de l'API sur l'outil Advanced Rest Client

1 https://sharetaskmanager.gallenne.fr/v1/creerTache

2 GET POST PUT DELETE PATCH Other methods application/x-www-form... Headers form Headers sets Variables

Content-Type application/x-www-form-urlencoded ADD HEADER

A 47 bytes Raw payload Data form Files

ENCODE PAYLOAD DECODE PAYLOAD

Form data for x-www-form-urlencoded parameters

intituleT	testLaurie	X
descriptionT	Je suis un test à 21:20	X
prioriteT	1	X
etatT	1	X
echeanceT	27-03-2017	X
refGroupe	1	X
idUtilisateur	u_58d7d0eb6db0c	X

ADD ANOTHER PARAMETER 3

4 SEND

5 200 OK 1313.00 ms DETAILS

Raw JSON

```
{
  "idTache": "t_58db69c0b953a",
  "intituleT": "testLaurie",
  "dateCreationT": "29-03-2017",
  "descriptionT": "Je suis un test à 21:20",
  "prioriteT": "1",
  "etatT": "1",
  "echeanceT": "27-03-2017",
  "refGroupe": "1",
  "dateDerniereModification": "29-03-2017"
}
```

6 1. URL de la requête
2. Méthodes de requête
3. Paramètres
4. Envoi de la requête
5. Code HTTP du résultat
6. Résultat au format JSON

Annexe 3 – Tableaux des compétences

1.1 Tableau des compétences pour le semestre 1

	Laurie	Valentin	Boubaker	Grégoire
Points forts au début du semestre	Encadrement, Analyse, Rédaction, Adaptation rapide au contexte	Analyse, Synthèse, Développement, Stratégie (Challenge Sopra Steria)	Idées innovantes, Esprit de synthèse, Evaluation des résultats, Recherches	Design des interfaces, Maîtrise rapide des outils de développement
Points faibles au début du semestre	Concentration sur une longue période, Rapidité de développement	Adaptation à un nouvel outil, Communication	Développement Java, Disponibilité	Java, Grammaire, français écrit
Montée en compétences au cours du semestre	Encadrement du groupe, Esprit de synthèse, Développement	Développement, Synthèse	Développement Java, Elaboration des diagrammes	Développement, Elaboration de diagrammes, Java
Axes d'amélioration pour le prochain semestre	Rapidité de développement, Adaptation à de nouveaux outils, Recherche de documentation	Gestion de projet, Communication, Efficacité	Gestion de projet, Meilleure maîtrise des outils, Rédaction	Gestion de projet, Rédaction

1.2 Tableau des compétences pour le semestre 2

	Boubaker	Grégoire	Laurie	Valentin
Axes d'amélioration prévus pour le semestre	Gestion de projet Maîtrise des outils Rédaction	Gestion de projet Rédaction	Rapidité de développement Adaptation à de nouveaux outils Recherche de documentation Management d'équipe,	Gestion de projet Communication Efficacité
Montée en compétences au cours du semestre	Gestion du planning personnel	Travail d'équipe, rédaction	fonctionnement d'une API et des appels à celle-ci, recherche de documentation	Maitrise des outils, efficacité, gestion BDD avec Android
Axes d'amélioration pour le futur	Qualité de documentation	Bonnes pratiques de développement	Clarté d'expression, organisation	Ecoute d'autrui

Annexe 4 - Calendrier prévisionnel et effectif du groupe

