



Université Mohammed V  
École nationale supérieure d'informatique et des  
systèmes  
Département de génie logiciel



## Rapport de projet

---

# Productivity Booster

---

Réalisé par :

YASSINE OUHADI

Yassine LAKHDACHI

Oussama Bakri

CHARIFE Mehdi

Encadré par :

Mme. Mounia ABIK

Année universitaire : 2022/2023

” Le succès vient généralement à ceux qui sont trop occupés pour le chercher. ”

Henry David Thoreau

## Dédicace

À Nos chers parents en signe de témoignage d'un grand respect et d'une profonde reconnaissance pour les sacrifices et les efforts qu'ils ont déployés pour notre éducation et notre formation.

À nos chers frères et sœurs.

À tous nos amis et collègues.

À tous nos professeurs.

À toute la famille.

Aucun mot ou expression ne serait valoriser les sacrifices que vous avez consentis pour notre bien-être et notre études.



## Remerciements

Nous tenons à exprimer nos sincères remerciements à notre professeur encadrant, Mme. Mounia ABIK, pour son soutien et son accompagnement tout au long de ce projet. Sa disponibilité, ses conseils et ses encouragements ont été précieux pour mener à bien cette réalisation.

Nous souhaitons également remercier l'ensemble de l'équipe pédagogique du département de Génie Logiciel de l'ENSIAS pour la qualité de leur enseignement, leur expertise et leur disponibilité.

Enfin, nous tenons à exprimer notre gratitude envers toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce projet, notamment nos camarades de groupe, ainsi que nos amis et notre famille pour leur soutien et leur encouragement.

## Résumé

Notre projet Android vise à créer une application de stimulation de la productivité. Notre projet se concentre sur permettre aux utilisateurs de planifier leurs tâches quotidiennes et d'optimiser leur temps en recevant des suggestions de l'application. De plus, l'application envoie également des rappels via des notifications pour s'assurer que l'utilisateur reste sur la bonne voie avec ses tâches.

# Table des matières

Introduction générale . . . . .	VII
<b>Chapitre 1: Contexte du projet</b>	<b>1</b>
1.1 Présentation de la mission du projet . . . . .	1
1.1.1 Problématique . . . . .	1
1.1.2 Objectif du projet . . . . .	2
1.1.3 Planification de projet . . . . .	3
1.2 Méthodologie suivie . . . . .	4
1.2.1 La méthodologie Agile . . . . .	4
1.2.2 Principes de l'agilité . . . . .	4
1.3 Conclusion . . . . .	5
<b>Chapitre 2: Analyse et conception</b>	<b>1</b>
2.1 Introduction . . . . .	1
2.2 Cahier des charges . . . . .	1
2.2.1 Besoins fonctionnels . . . . .	1
2.2.2 Besoins non fonctionnels . . . . .	2
2.3 Conception . . . . .	2
2.3.1 Diagramme de séquences . . . . .	2
2.4 Conclusion . . . . .	5
<b>Chapitre 3: Réalisation</b>	<b>1</b>
3.1 Introduction . . . . .	1
3.2 Environnement du travail . . . . .	1
3.3 Fonctionnalités Android . . . . .	2

---

3.4	L'architecture générale des IHM . . . . .	3
3.5	Interfaces Graphiques . . . . .	4
3.6	Optimisation de l'application . . . . .	10
3.7	Conclusion . . . . .	11
	Conclusion . . . . .	12

## List of Figures

2.1	Diagramme des séquences de connexion . . . . .	3
2.2	Diagramme des séquences d'inscription . . . . .	3
2.3	Diagramme des séquences de la création d'une tâche . . . . .	4
2.4	Diagramme des séquences de lister les tâches . . . . .	5
3.1	Android Studio . . . . .	1
3.2	Fonctionnalités Android . . . . .	3
3.3	Interface de connexion . . . . .	5
3.4	Interface d'inscription . . . . .	5
3.5	Interface de la liste des tâches . . . . .	6
3.6	Interface de détails de la tâche . . . . .	6
3.7	DatePicker . . . . .	7
3.8	TimePicker . . . . .	7
3.9	Calendar . . . . .	8
3.10	Alarm . . . . .	8
3.11	Compléter une tâche . . . . .	9
3.12	Popup . . . . .	9
3.13	tâche en arrière-plan . . . . .	10
3.14	Using RecyclerView . . . . .	10



## **Introduction générale**

Notre projet de développement s'intéresse à la gestion des absences à l'ENSIAS en utilisant une technologie innovante, le système RFID. Ce projet a été réalisé dans le cadre du module "Plateformes de développement" du semestre 4 de la filière Génie Logiciel à l'ENSIAS. L'objectif de ce projet est de proposer une solution efficace pour la gestion des absences des étudiants et des enseignants dans l'établissement. Dans ce rapport, nous présenterons les différentes étapes de la réalisation de ce projet, de la conception à l'implémentation de l'application web de gestion des absences et du système RFID.

# Chapitre 1

## Contexte du projet

Dans ce chapitre, nous commencerons tout d'abord par la présentation de la problématique et le contexte général de notre projet ainsi que la méthodologie suivie pour trouver une solution à la problématique.

### 1.1 Présentation de la mission du projet

Notre projet a pour mission de développer une application Android visant à stimuler la productivité des utilisateurs. Nous sommes conscients des défis auxquels les gens sont confrontés dans leur vie quotidienne, notamment la gestion du temps et l'accomplissement de leurs tâches. C'est pourquoi nous avons entrepris de créer une application qui les aide à planifier efficacement leurs activités et à optimiser leur emploi du temps.

#### 1.1.1 Problématique



La vie moderne est souvent synonyme de rythme effréné, de multiples responsabilités et d'un emploi du temps chargé. Les individus sont confrontés à de nombreux défis pour gérer leur temps de manière efficace et atteindre leurs objectifs. C'est dans ce contexte que se pose la problématique suivante : Comment aider les utilisateurs à optimiser leur productivité et à mieux gérer leur temps dans un monde en perpétuel mouvement ?

La problématique se décline en plusieurs questions :

1. Comment les utilisateurs peuvent-ils planifier leurs tâches de manière efficace et structurée pour éviter l'oubli ou le manque d'organisation ?

2. Comment peuvent-ils hiérarchiser leurs tâches en fonction de leur importance et de leur urgence pour maximiser leur efficacité ?
3. Comment peuvent-ils identifier les moments les plus propices pour effectuer certaines tâches afin d'optimiser leur temps et leur énergie ?
4. Comment peuvent-ils rester concentrés sur leurs tâches malgré les nombreuses distractions de la vie quotidienne ?
5. Comment peuvent-ils être régulièrement rappelés de l'accomplissement de leurs tâches pour éviter les retards ou les oublis ?
6. Comment peuvent-ils obtenir des suggestions personnalisées et des conseils pratiques pour améliorer leur gestion du temps et leur productivité ?

La problématique centrale réside donc dans la recherche de solutions permettant aux utilisateurs de surmonter ces défis et d'optimiser leur productivité au quotidien. Notre projet vise à apporter une réponse concrète à cette problématique en développant une application Android qui offre des fonctionnalités avancées de planification, de suggestion et de rappel, afin d'aider les utilisateurs à mieux gérer leur temps et à atteindre leurs objectifs de manière plus efficace.

### 1.1.2 Objectif du projet

L'objectif principal de notre projet est de développer une application Android qui agit comme un booster de productivité pour les utilisateurs. Nous souhaitons fournir un outil pratique et efficace qui les aide à optimiser leur temps, à planifier leurs tâches et à atteindre leurs objectifs de manière plus efficace. Les objectifs spécifiques de notre projet sont les suivants :

1. Créer une interface conviviale et intuitive : Nous voulons développer une application avec une interface utilisateur conviviale, facile à naviguer et agréable à utiliser. Une expérience utilisateur intuitive favorise l'adoption et l'utilisation régulière de l'application.
2. Permettre la planification des tâches : Notre application permettra aux utilisateurs de créer des listes de tâches, de définir des priorités et d'organiser leurs activités quotidiennes de manière structurée. Cela leur permettra d'avoir une vision claire de leurs tâches à accomplir.
3. Fournir des suggestions personnalisées : Grâce à des algorithmes d'intelligence artificielle avancés, notre application analysera les habitudes et les préférences de l'utilisateur pour lui proposer des suggestions personnalisées. Ces suggestions aideront les utilisateurs à optimiser leur temps et à prendre des décisions éclairées quant à la planification de leurs tâches.
4. Envoyer des rappels et des notifications : L'application enverra des rappels et des notifications aux utilisateurs pour les aider à rester sur la bonne voie avec leurs tâches.

Ces rappels personnalisables les aideront à ne pas oublier les échéances et à se tenir responsables de l'accomplissement de leurs tâches.

En atteignant ces objectifs, notre projet vise à offrir aux utilisateurs un outil complet et fiable pour améliorer leur gestion du temps, optimiser leur productivité et les aider à atteindre leurs objectifs personnels et professionnels de manière plus efficace.

Enfin, pour garantir le bon déroulement du projet et respecter les délais, nous avons élaboré une planification globale de conduite du projet, en utilisant un diagramme de Gantt qui nous a permis de hiérarchiser les tâches et de quantifier les délais d'exécution à chaque étape du projet.

### 1.1.3 Planification de projet

La conception et la réalisation d'une solution répondant à l'ensemble des besoins imposés par la problématique constituaient la principale motivation de ce projet. Il s'agissait de concevoir une solution propriétaire, afin de nous donner la liberté de la modifier, l'optimiser et la personnaliser à notre guise.

Comme pour tout projet, une phase de planification s'avère nécessaire pour en assurer le bon déroulement. Cela permet de définir les tâches à réaliser, de maîtriser les risques et de rendre compte de l'état d'avancement du projet. Afin de garantir le respect des délais impartis, nous avons élaboré une planification globale de conduite du projet.

Cette planification a été représentée sous la forme d'un diagramme de Gantt, qui décrit l'ordonnancement prévu des différentes phases du projet.

Nous avons planifié le déroulement des différentes étapes de la manière suivante :

1. Phase de conception :
  - (a) Effectuer une analyse des besoins et des attentes des utilisateurs.
  - (b) Définir les fonctionnalités clés de l'application, les interfaces utilisateur et l'architecture globale.
  - (c) Créer des maquettes ou des wireframes pour visualiser l'apparence de l'application.
2. Phase de développement :
  - (a) Développer la structure de base de l'application en utilisant les meilleures pratiques de développement Android.
  - (b) Implémenter les fonctionnalités de base telles que la création de tâches, la gestion des listes, les priorités, les notes supplémentaires, etc.
  - (c) Mettre en place le système de rappels et de notifications.
  - (d) Assurer la synchronisation en temps réel entre les appareils.
3. Phase de test et de correction :
  - (a) Effectuer des tests approfondis de l'application pour détecter les éventuels bugs, erreurs ou problèmes de performance.

- (b) Corriger les problèmes identifiés et améliorer la stabilité et la convivialité de l'application.
- 4. Phase de polissage et d'optimisation :
  - (a) Affiner l'interface utilisateur pour une meilleure expérience utilisateur.
  - (b) Optimiser les performances de l'application pour assurer une utilisation fluide et réactive.
  - (c) Effectuer des tests de charge pour vérifier la capacité de l'application à gérer un grand nombre de tâches et d'utilisateurs.
- 5. Phase de déploiement et de lancement :
  - (a) Préparer la version finale de l'application pour le déploiement sur Google Play Store.
  - (b) On vise à lancer l'application sur le Google Play Store et rendre l'application accessible aux utilisateurs.

## 1.2 Méthodologie suivie

Dans cette partie, nous allons aborder la méthodologie de travail mise en place lors du projet.

### 1.2.1 La méthodologie Agile

Pour mener à bien ce projet, nous avons choisi de mettre en place une méthode de travail agile, qui nous permet de travailler de manière itérative, avec une grande flexibilité pour adapter le projet aux changements éventuels. Cette méthode nous a permis de travailler efficacement en équipe et de synchroniser notre travail en utilisant des outils tels que GitHub, Git et JIRA, ce qui nous a permis de collaborer à distance en toute transparence. Grâce à cette approche, nous avons pu répondre rapidement aux besoins du projet et garantir sa réussite. Mon encadrant est impliqué dans le projet du début à la fin ce qui lui donne de la visibilité. Cela permet d'éviter « l'effet tunnel », c'est-à-dire de se lancer dans un projet et arriver à terme avec un produit qui ne correspond pas aux attentes des clients (les encadrants dans mon cas) ou ne pas mener le projet à terme.

### 1.2.2 Principes de l'agilité

Bien que le projet ait été effectué en équipe, plusieurs principes de l'agilité ont été respectés, notamment les suivants :

1. La concentration sur la qualité du développement logiciel, en veillant à ce que chaque fonctionnalité ajoutée soit testée de manière approfondie avant d'être mise en production.

2. L'adaptation au changement en restant ouvert aux retours des utilisateurs et en ajustant les fonctionnalités en conséquence, tout en gardant un plan général du projet.
3. La collaboration étroite entre les membres de l'équipe de développement et les utilisateurs finaux de l'application, pour garantir que les besoins de ces derniers sont pris en compte dans chaque itération du développement.
4. L'utilisation de méthodes de développement Agile telles que Scrum pour planifier et suivre le travail, en veillant à ce que l'équipe reste alignée sur les objectifs globaux du projet.

### 1.3 Conclusion

Après avoir présenté le contexte général du projet, nous allons mettre en exergue l'analyse du cahier de charge et la conception dans le chapitre suivant.

# Chapitre 2

## Analyse et conception

### 2.1 Introduction

La conception est une étape primordiale dans le cycle de vie d'une application, elle a pour objectif de faire l'étude des données et des traitements à effectuer. C'est en général dans cette phase que s'appliquent les techniques de modélisation. Je commencerai d'abord par l'analyse du sujet puis je passerai à la conception.

### 2.2 Cahier des charges

Dans cette partie du rapport de notre projet, nous allons décrire en détail les besoins et les attentes, ainsi que les exigences et les fonctionnalités spécifiques du projet.

#### 2.2.1 Besoins fonctionnels

Après l'étude du système, cette partie est réservée à la description des exigences fonctionnelles des utilisateurs.

- Création de tâches : Permettre aux utilisateurs de créer des tâches avec des titres, des descriptions et des dates d'échéance.
- Planification et organisation : Permettre aux utilisateurs de planifier leurs tâches en les assignant à des jours spécifiques ou en les regroupant dans des listes thématiques.
- Priorisation des tâches : Offrir la possibilité aux utilisateurs de définir des priorités pour leurs tâches, par exemple en utilisant des codes couleur ou des étiquettes.
- Suggestions de productivité : Fournir des suggestions personnalisées aux utilisateurs pour optimiser leur temps, basées sur des algorithmes d'intelligence artificielle qui analysent leurs habitudes et leurs préférences.
- Rappels et notifications : Envoyer des rappels et des notifications aux utilisateurs pour leur rappeler les tâches à accomplir et les aider à rester sur la bonne voie.

- Recherche des tâches : Les utilisateurs doivent pouvoir rechercher des tâches spécifiques en utilisant des mots-clés, des filtres ou d'autres critères de recherche.
- Notification des tâches : L'application doit pouvoir envoyer des notifications aux utilisateurs pour leur rappeler les tâches à effectuer. Ces notifications doivent être configurables en termes de fréquence et de moment de la journée.

### 2.2.2 Besoins non fonctionnels

Les exigences « non fonctionnelles » sont celles qui sont importantes voire critiques aux yeux des utilisateurs et qui assurent le bon fonctionnement du système.

On peut les répartir par catégories, notamment :

- Sécurité : Assurer la confidentialité des données des utilisateurs en mettant en place des mesures de sécurité robustes, telles que le chiffrement des données et l'authentification sécurisée.
- Performance : L'application doit être rapide et réactive, offrant une expérience utilisateur fluide sans délais significatifs lors de la création ou de la modification des tâches.
- Disponibilité : Garantir une disponibilité élevée de l'application, en minimisant les temps d'arrêt planifiés pour les mises à jour ou les maintenances.
- Convivialité : Offrir une interface utilisateur intuitive et conviviale, avec une navigation facile, des icônes et des libellés clairs, pour faciliter l'utilisation de l'application par les utilisateurs de tous niveaux de compétence.
- L'évolutivité : Le système doit être « scalable » avec la possibilité d'ajout de fonctionnalité à tout temps.

## 2.3 Conception

### 2.3.1 Diagramme de séquences

DIAGRAMME DES SÉQUENCES DE CONNEXION

DIAGRAMME DES SÉQUENCES D'INSCRIPTION



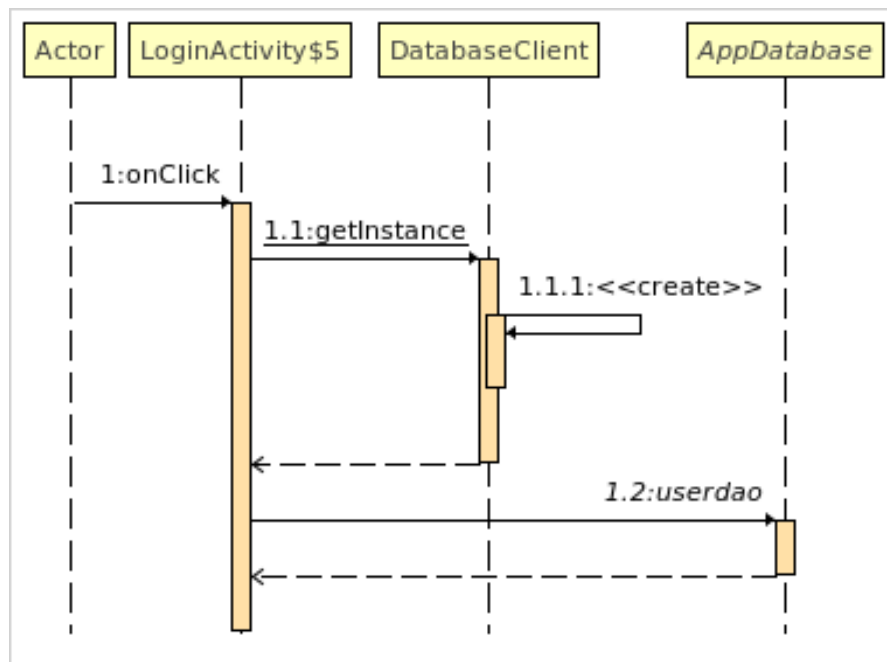


FIGURE 2.1 – Diagramme des séquences de connexion

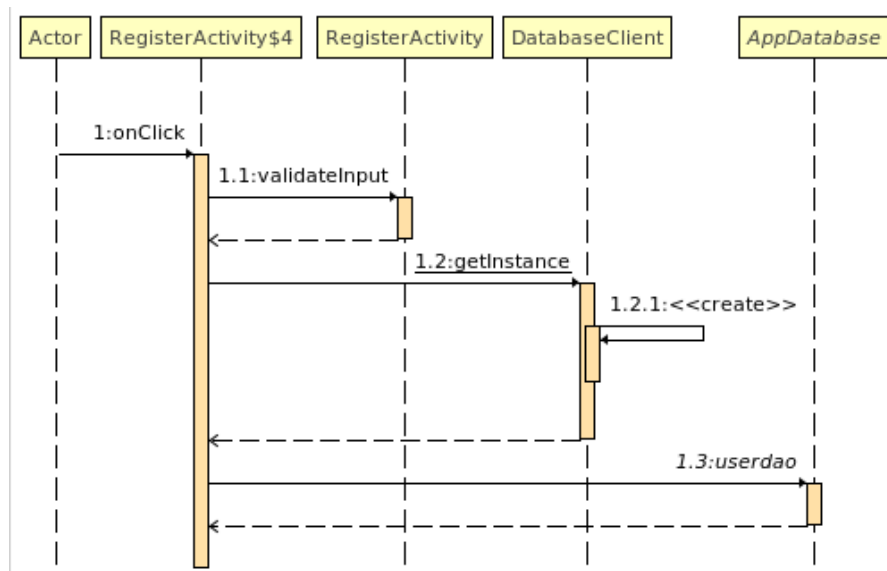


FIGURE 2.2 – Diagramme des séquences d'inscription

## DIAGRAMME DES SÉQUENCES DE LA CRÉATION D'UNE TÂCHE

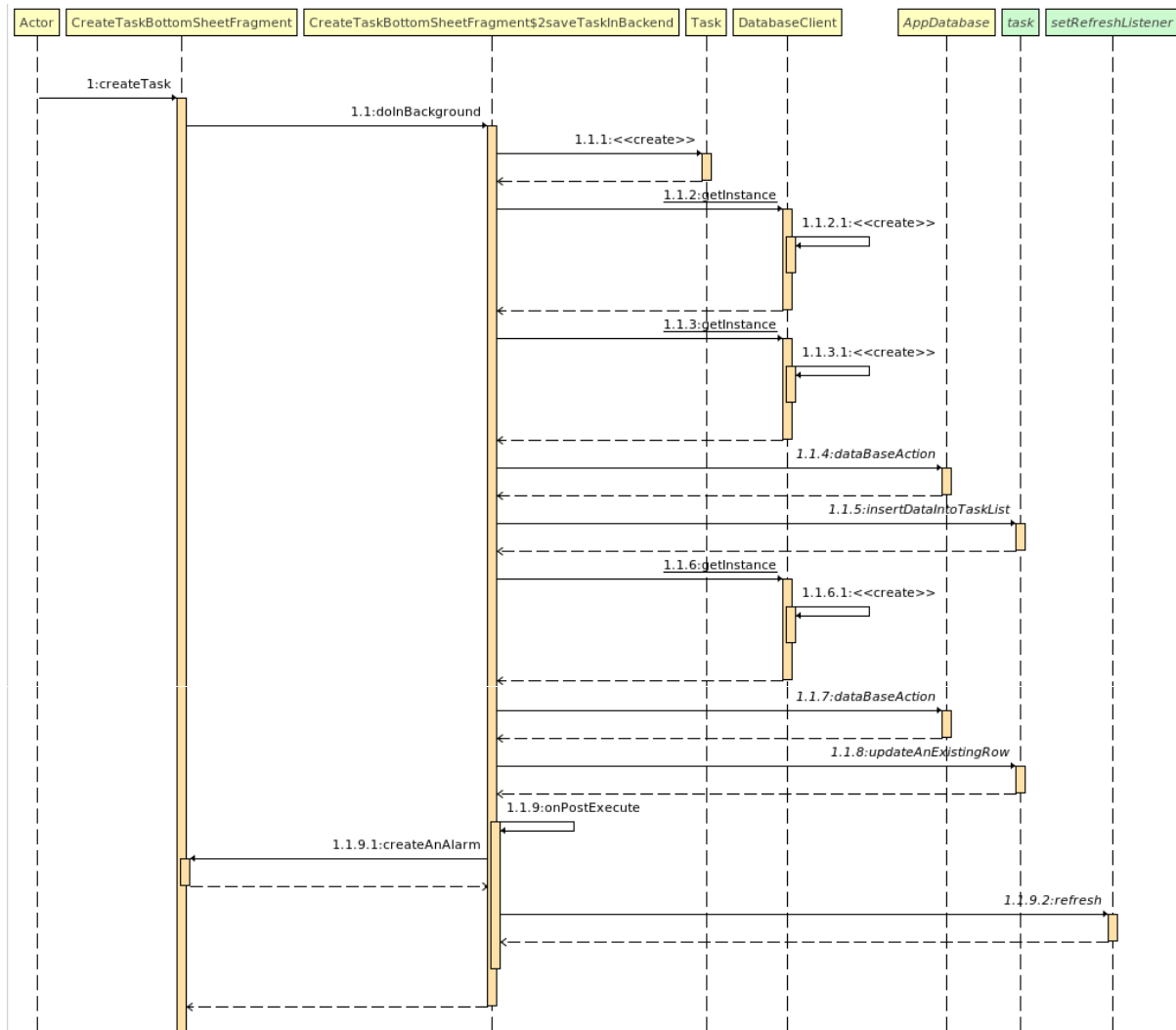


FIGURE 2.3 – Diagramme des séquences de la création d'une tâche

## DIAGRAMME DES SÉQUENCES DE LISTER LES TÂCHES

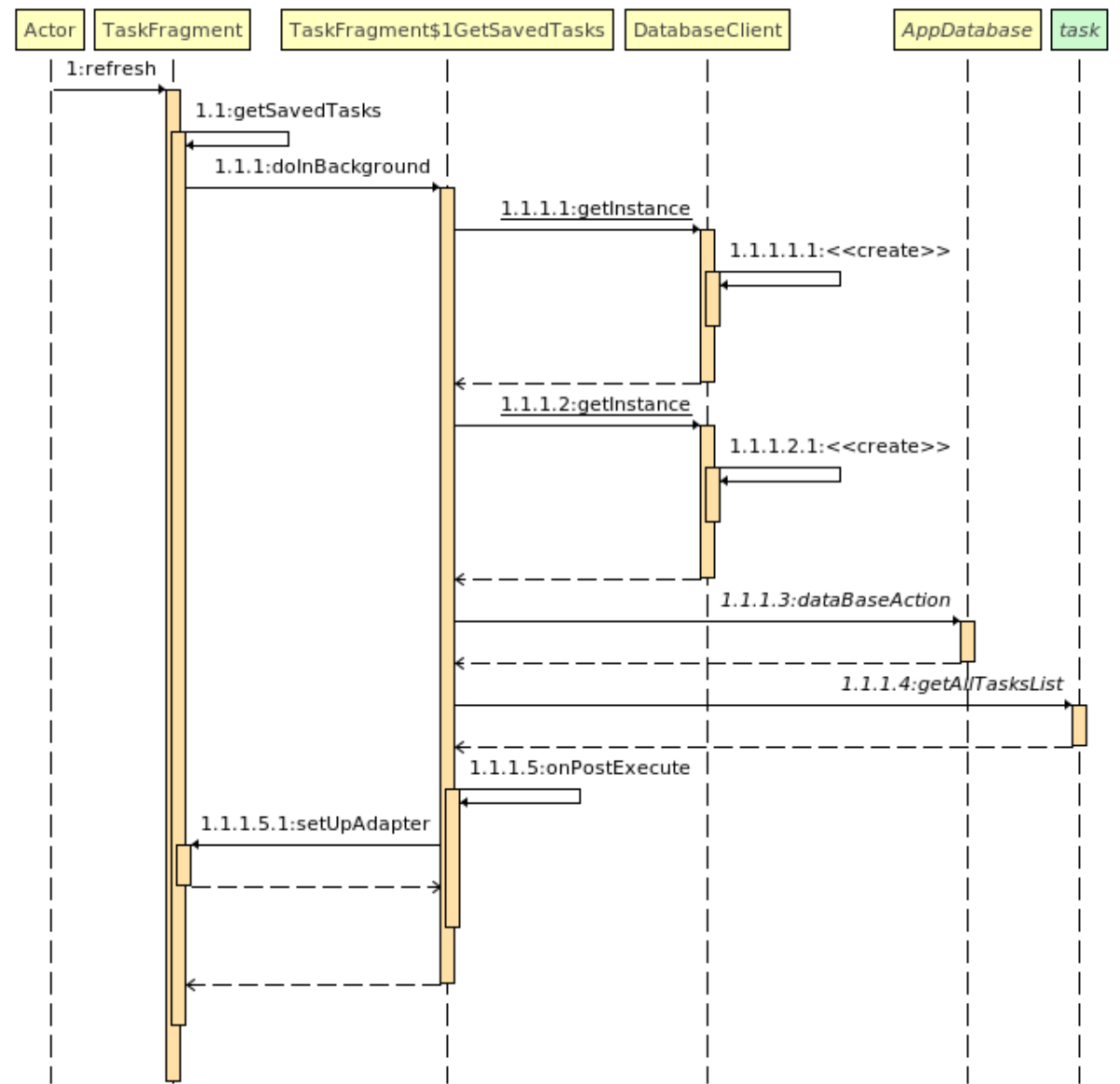


FIGURE 2.4 – Diagramme des séquences de lister les tâches

## 2.4 Conclusion

Dans ce chapitre, nous allons aborder le cahier des charges, ainsi que la modélisation des données et des traitements en utilisant la l'analyse UML, qui nous permettent de donner une vision bien détaillée sur le fonctionnement du système. À présent, nous allons entamer la phase de réalisation.

# Chapitre 3

## Réalisation

### 3.1 Introduction

Au niveau de cette dernière partie, nous nous intéressons aux outils de développement utilisés pour la réalisation de notre application, ainsi qu'aux principales fonctionnalités et interfaces de l'application et l'architecture générale des IHM.

### 3.2 Environnement du travail

#### ANDROID STUDIO



FIGURE 3.1 – Android Studio

Pour le développement de notre application To-Do List, nous avons utilisé Android Studio comme environnement de développement. Android Studio est l'IDE (Integrated Development Environment) recommandé par Google pour le développement d'applications Android. Cet outil puissant et complet offre de nombreuses fonctionnalités spécifiques à la plateforme Android, ce qui en fait un choix idéal pour notre projet.

#### ROOM DATABASE

Nous avons utilisée comme une base de données locale pour stocker les tâches et gérer les opérations CRUD (Create, Read, Update, Delete).

### 3.3 Fonctionnalités Android

Avant de présenter le tableau des fonctionnalités Android utilisées dans notre application, il est essentiel de souligner l'importance de ces fonctionnalités dans le développement de notre application. Grâce à ces fonctionnalités, nous avons pu offrir une expérience utilisateur fluide et conviviale, tout en répondant aux nos besoins.

Nous avons exploité les fonctionnalités offertes par le framework Android pour créer une application robuste et efficace :

#### RECYCLERVIEW

Le RecyclerView, par exemple, nous a permis de présenter la liste des tâches de manière dynamique, avec un défilement fluide et une gestion optimisée de la mémoire :

#### LES FRAGMENTS

Les fragments ont été utilisés pour organiser les différentes vues de l'application, permettant aux utilisateurs de naviguer facilement entre la liste des tâches, le planning, les paramètres et le profil.

#### ALERTDIALOG ET POPUPMENU

Pour améliorer l'interaction utilisateur, nous avons utilisé des fonctionnalités telles que l'AlertDialog et le PopupMenu, qui ont facilité les actions contextuelles, telles que la modification, la complétion ou la suppression d'une tâche.

#### L'ALARMMANAGER

L'AlarmManager nous a permis de mettre en place des rappels d'alarme pour les tâches récurrentes, garantissant ainsi que les utilisateurs ne manquent aucune échéance importante.

#### BOTTOMSHEETDIALOGFRAGMENT

De plus, nous avons tiré parti du BottomSheetDialogFragment pour offrir aux utilisateurs une flexibilité accrue lors de l'ajout de nouvelles tâches, en leur permettant de choisir l'heure et la date de manière pratique.

#### CALENDAR, GREGORIANCALENDAR ET EVENTDAY

En utilisant des fonctionnalités telles que Calendar, GregorianCalendar et EventDay, nous avons pu gérer les dates et les événements liés aux tâches, facilitant ainsi leur suivi et leur visualisation sur un calendrier.

#### NOTIFICATIONMANAGER ET AU NOTIFICATIONCOMPAT

Enfin, nous avons intégré des fonctionnalités de notification grâce au NotificationManager et au NotificationCompat, qui ont permis d'alerter les utilisateurs des rappels de tâches ou des tâches urgentes, même lorsque l'application n'était pas active.

Dans le tableau suivant, nous présentons de manière concise les fonctionnalités Android que nous avons utilisées dans notre application To-Do List, en spécifiant où et comment nous les avons implémentées :

Fonctionnalité	Utilisation
RecyclerView	Utilisé pour afficher la liste des tâches à accomplir, permettant un défilement fluide et efficace.
Fragments	Utilisés pour organiser et gérer les différentes vues de l'application, tels que les fragments pour la liste des tâches, le planning, les paramètres et le profil.
BottomNavigationView	Utilisé pour la navigation entre les différentes sections de l'application, telles que la liste des tâches, le planning, les paramètres, etc.
PopupMenu	Utilisé pour afficher un menu contextuel avec des options supplémentaires pour chaque tâche, permettant la modification, la complétion ou la suppression d'une tâche.
AlertDialog	Utilisé pour afficher des boîtes de dialogue contextuelles pour des actions telles que l'indication d'une tâche comme annulée, complétée ou supprimée.
AlarmManager	Utilisé pour définir des rappels d'alarme pour les tâches qui se répètent à des intervalles spécifiques.
DatePickerDialog	Utilisé pour afficher un sélecteur de date pour choisir la date d'une tâche.
TimePickerDialog	Utilisé pour afficher un sélecteur d'heure pour choisir l'heure d'une tâche.
AsyncTask	Utilisés pour effectuer des opérations asynchrones, telles que la sauvegarde ou la suppression de tâches en arrière-plan.
BottomSheetDialogFragment	Utilisé pour afficher une feuille de bas de page avec des options flexibles pour ajouter une nouvelle tâche à différents moments.
Calendar	Utilisés pour gérer les dates et les événements liés aux tâches, tels que le marquage des tâches terminées sur le calendrier.
NotificationManager	Utilisés pour afficher des notifications pour les rappels de tâches ou les tâches urgentes.

FIGURE 3.2 – Fonctionnalités Android

### 3.4 L'architecture générale des IHM

L'architecture générale de l'interface utilisateur de notre application To-Do List repose sur l'utilisation de fragments et d'activités pour créer une expérience utilisateur cohérente et intuitive. Les différents fragments et activités interagissent entre eux pour offrir les fonctionnalités clés de l'application.

Lorsque l'utilisateur lance l'application, il est d'abord dirigé vers l'interface de connexion. S'il dispose déjà d'un compte, il peut saisir ses informations de connexion, et l'application vérifie les identifiants et le redirige vers l'activité principale. En revanche, s'il est nouveau et souhaite créer un compte, il peut accéder à l'interface d'inscription (register), où il saisit ses informations personnelles et crée son compte. Une fois le compte créé, l'utilisateur est automatiquement connecté et redirigé vers l'activité principale.

L'activité principale est composée de plusieurs fragments qui sont accessibles via un menu de navigation situé dans le bas de l'écran, utilisant le composant `BottomNavigationView`. Les fragments principaux sont les suivants :

1. Fragment de la liste des tâches : Ce fragment affiche toutes les tâches enregistrées par l'utilisateur, utilisant un `RecyclerView` pour une gestion efficace de l'affichage des tâches. Les utilisateurs peuvent visualiser, ajouter, modifier, compléter ou supprimer des tâches à partir de cet écran. Lorsqu'une tâche est sélectionnée, l'application affiche les détails de la tâche dans un autre fragment.
2. Fragment du planning : Ce fragment permet aux utilisateurs de planifier leurs tâches sur un calendrier. Il utilise la bibliothèque `CalendarView` pour afficher les jours et les événements planifiés. Les utilisateurs peuvent ajouter, modifier ou supprimer des événements à partir de cet écran.
3. Fragment des paramètres : Ce fragment offre aux utilisateurs la possibilité de personnaliser diverses options de l'application, telles que les préférences d'affichage, les rappels de tâches, etc.
4. Fragment du profil : Ce fragment affiche les informations personnelles de l'utilisateur, telles que son nom, son adresse e-mail, etc. Les utilisateurs peuvent modifier leurs informations de profil à partir de cet écran.

Cette architecture basée sur les fragments et les activités permet une modularité et une réutilisabilité accrues des composants de l'interface utilisateur. Elle offre une navigation fluide entre les différentes fonctionnalités de l'application et permet aux utilisateurs de gérer leurs tâches de manière efficace et conviviale.

## 3.5 Interfaces Graphiques

Pour notre projet, nous sommes intéressés à fournir des interfaces utilisateur conviviales, attrayantes et simples. Concernant cette dernière partie, nous allons présenter différentes interfaces de notre application.

## INTERFACE DE CONNEXION

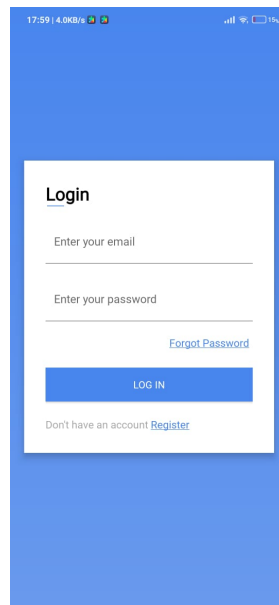
A mobile application interface for login. The background is a solid blue color. In the center, there is a white rectangular card. At the top of the card, the word "Login" is written in bold black text. Below it, there are two input fields: "Enter your email" and "Enter your password", each with a light gray border. To the right of the password field, there is a blue link that says "Forgot Password". Below the input fields is a blue button with the text "LOG IN" in white. At the bottom of the card, there is a line of text: "Don't have an account" followed by a blue link that says "Register". The status bar at the top of the screen shows the time "17:59", signal strength, and battery level.

FIGURE 3.3 – Interface de connexion

## INTERFACE D'INSCRIPTION

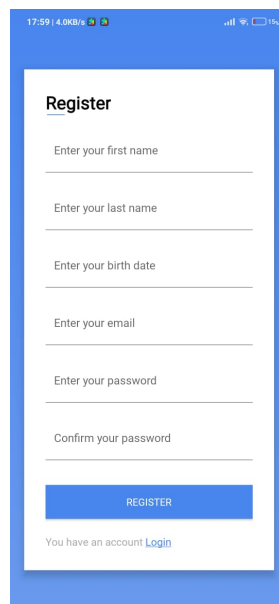
A mobile application interface for registration. The background is a solid blue color. In the center, there is a white rectangular card. At the top of the card, the word "Register" is written in bold black text. Below it, there are five input fields: "Enter your first name", "Enter your last name", "Enter your birth date", "Enter your email", and "Enter your password", each with a light gray border. Below the password field is another input field labeled "Confirm your password". Below all input fields is a blue button with the text "REGISTER" in white. At the bottom of the card, there is a line of text: "You have an account" followed by a blue link that says "Login". The status bar at the top of the screen shows the time "17:59", signal strength, and battery level.

FIGURE 3.4 – Interface d'inscription



## INTERFACE DE LA LISTE DES TÂCHES

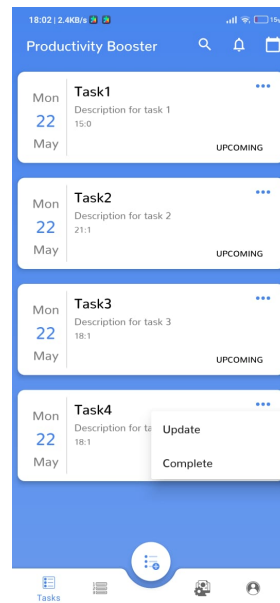


FIGURE 3.5 – Interface de la liste des tâches

## INTERFACE DE DÉTAILS DE LA TÂCHE

18:02 | 2.1KB/s

≡+

**Add a task**

Fill the details below to add a task into your TODO

Task title

Task description

Task date

Task time

Add an event

ADD TASK

FIGURE 3.6 – Interface de détails de la tâche

DATEPICKER

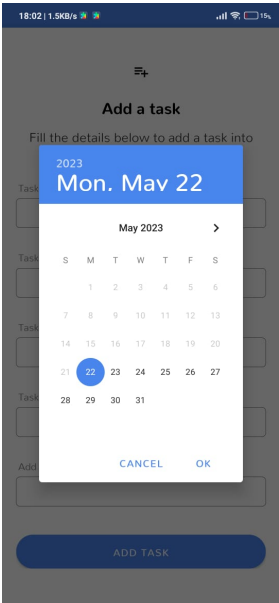


FIGURE 3.7 – DatePicker

TIMEPICKER

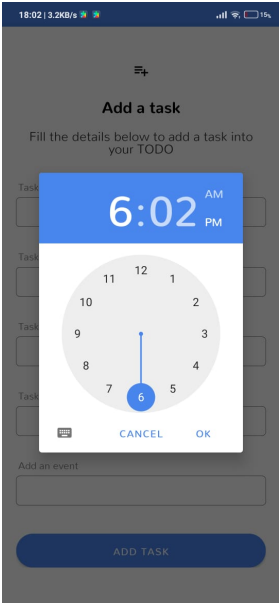


FIGURE 3.8 – TimePicker

CALENDAR

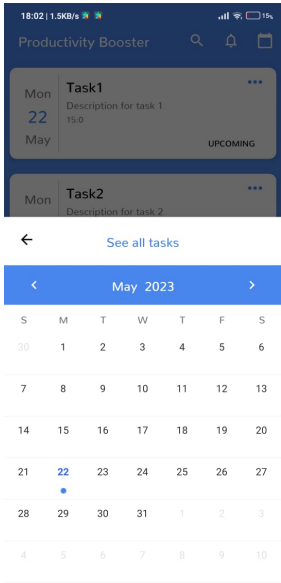


FIGURE 3.9 – Calendar

ALARM

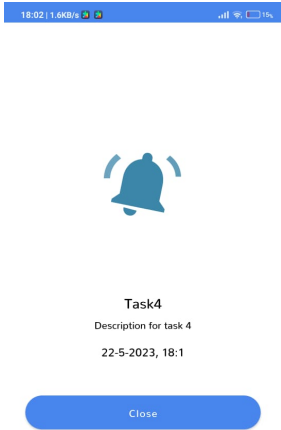


FIGURE 3.10 – Alarm

## COMPLÉTER UNE TÂCHE

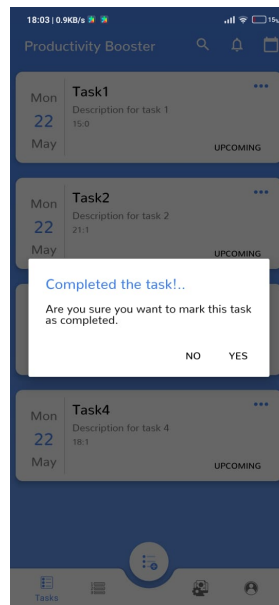


FIGURE 3.11 – Compléter une tâche

## POPUP

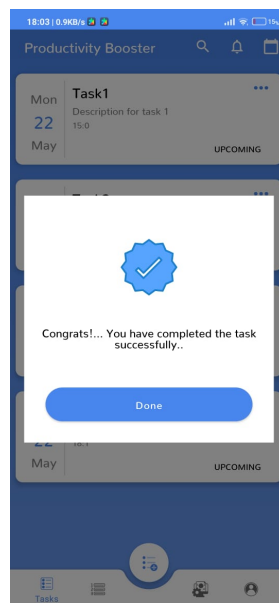


FIGURE 3.12 – Popup

## 3.6 Optimisation de l'application

Pour optimiser les performances de notre application To-Do List, nous avons mis en œuvre plusieurs mesures visant à améliorer l'efficacité et la réactivité de l'application. Voici les principales mesures que nous avons prises :

### UTILISATION DE TÂCHES EN ARRIÈRE-PLAN

Pour améliorer la réactivité de l'interface utilisateur, nous avons délégué certaines tâches intensives en ressources à des tâches en arrière-plan. Par exemple, lors de la synchronisation des données avec le serveur, nous avons utilisé des tâches asynchrones pour exécuter ces opérations sans bloquer l'interface utilisateur. Cela a permis aux utilisateurs d'interagir avec l'application de manière fluide même pendant les opérations de synchronisation.

```
private void createTask() {  
    // yassine *  
    class saveTaskInBackend extends AsyncTask<Void, Void, Void> {  
        // yassine *  
        @SuppressWarnings("WrongThread")  
        @Override  
        protected Void doInBackground(Void... voids) {...}  
        // 7 usages // yassine  
        @Override  
        protected void onPostExecute(Void aVoid) {...}  
    }  
    saveTaskInBackend st = new saveTaskInBackend();  
    st.execute();  
}
```

FIGURE 3.13 – tâche en arrière-plan

### UTILISATION DE STRUCTURES DE DONNÉES OPTIMISÉES

Nous avons veillé à utiliser des structures de données appropriées pour optimiser les performances. Par exemple, pour gérer la liste des tâches, nous avons utilisé le composant RecyclerView avec une approche de recyclage des vues, ce qui a permis une utilisation efficace de la mémoire et une meilleure réactivité lors du défilement de la liste.

```
@Override  
public View onCreateView(LayoutInflater inflater, ViewGroup container,  
    Bundle savedInstanceState) {  
    View rootView = inflater.inflate(R.layout.tasks, container, attachToRoot: false);  
    taskRecycler = rootView.findViewById(R.id.taskRecycler);  
    noDataImage = rootView.findViewById(R.id.noDataImage);  
    setUpAdapter();  
    getSavedTasks();  
    return rootView;  
}
```

FIGURE 3.14 – Using RecyclerView

## GESTION EFFICACE DE LA MÉMOIRE

Nous avons pris des mesures pour optimiser l'utilisation de la mémoire de l'application. Cela comprend la libération appropriée des ressources non utilisées, la gestion des références, la suppression des objets inutiles et la limitation de la consommation de mémoire lors du chargement des données. Nous avons également évité les fuites de mémoire potentielles en nous assurant de libérer correctement les ressources lorsque nous n'en avons plus besoin.

En prenant ces mesures d'optimisation, nous avons réussi à améliorer considérablement les performances de notre application To-Do List, réduisant les temps de chargement, minimisant la consommation de mémoire et offrant une expérience utilisateur plus fluide et réactive.

## 3.7 Conclusion

Dans ce chapitre, nous avons montré et détaillé l'architecture et les fonctionnalités utilisées pour la réalisation de notre projet, ainsi que l'environnement de travail. Nous avons également présenté les principales interfaces qui permettent de répondre aux besoins de notre application.

## **Conclusion**

En conclusion, notre projet Android vise à créer une application qui booste la productivité des utilisateurs en leur permettant de planifier leurs tâches quotidiennes, d'optimiser leur temps grâce à des suggestions et de recevoir des rappels pour rester sur la bonne voie. Notre objectif est de fournir une expérience conviviale et intuitive, avec une synchronisation multi-appareils, pour aider les utilisateurs à maximiser leur productivité et à atteindre leurs objectifs. En offrant ces fonctionnalités essentielles, notre application vise à simplifier la gestion du temps et à améliorer l'efficacité des utilisateurs dans leur vie quotidienne.

## **Webographie**

*[https ://github.com/devandroidboot/Productivity-Booster](https://github.com/devandroidboot/Productivity-Booster)*