A close-up of a logo

Description automatically generatedUne image contenant texte, Police

Description générée automatiquement

**Sous le thème**

**Encadré par : Mr. Anass Deroussi**

Année universitaire : 2024-2025

.

Application de gestion de contact

**Rapport de Projet**

**5ème année**

**Ingénierie Informatique et Réseaux**

**Réalisé par:**

* Hanae Ray
* Mina El Mrabti

Résumé

Cette application Flutter illustre la gestion d'une liste de contacts en mettant en œuvre les opérations CRUD suivantes :

1. **Création** : Ajout de nouveaux contacts avec nom et numéro de téléphone.
2. **Lecture** : Affichage des contacts existants dans une vue en liste.
3. **Mise à jour** : Modification des informations d'un contact sélectionné.
4. **Suppression** : Suppression de contacts de la liste.

L'interface utilisateur est construite à l'aide de **ListView**, permettant une présentation dynamique et interactive des contacts. Les fonctionnalités de l'application sont implémentées en utilisant les widgets et les mécanismes de gestion d'état propres à Flutter, assurant une expérience utilisateur fluide et réactive.

Cette application sert d'exemple pratique pour comprendre l'intégration des opérations CRUD dans une application Flutter, offrant une base solide pour le développement d'applications mobiles interactives et fonctionnelles.

Abstract

This Flutter application demonstrates the management of a contact list by implementing the following CRUD operations:

1. **Create**: Adding new contacts with a name and phone number.
2. **Read**: Displaying existing contacts in a list view.
3. **Update**: Editing the details of a selected contact.
4. **Delete**: Removing contacts from the list.

The user interface is built using **ListView**, providing a dynamic and interactive presentation of the contacts. The application's functionalities are implemented using Flutter widgets and state management mechanisms, ensuring a smooth and responsive user experience.

This application serves as a practical example to understand how CRUD operations are integrated into a Flutter application, offering a solid foundation for developing interactive and functional mobile apps.

Table de matière

[Introduction Générale 6](#_Toc186366323)

[Chapitre 1 : Contexte générale 7](#_Toc186366324)

[I. Introduction 7](#_Toc186366325)

[II. Problématique 7](#_Toc186366326)

[III. Étude de l'existant 7](#_Toc186366327)

[IV. Conclusion 8](#_Toc186366328)

[Chapitre 2: Analyse et conception 9](#_Toc186366329)

[I. Introduction 9](#_Toc186366330)

[II. Analyse des besoins 9](#_Toc186366331)

[1. Besoins fonctionnels 9](#_Toc186366332)

[2. Modélisation UML 9](#_Toc186366333)

[III. Besoins techniques et graphiques 11](#_Toc186366334)

[1. Visual Studio Code (VS Code) 11](#_Toc186366335)

[2. Dart 12](#_Toc186366336)

[3. Git 12](#_Toc186366337)

[4. Enterprise Architect 12](#_Toc186366338)

[5. Flutter 13](#_Toc186366339)

[IV. Conclusion 13](#_Toc186366340)

[Chapitre 3 : Réalisation de l’application 14](#_Toc186366341)

[I. Introduction 14](#_Toc186366342)

[II. Interfaces de l’application 14](#_Toc186366343)

[1. Accueil 14](#_Toc186366344)

[2. Gestion de contact 15](#_Toc186366345)

[III. Conclusion 17](#_Toc186366346)

[Conclusion Générale 18](#_Toc186366347)

Table de figure

[Figure 1: Diagramme des cas d'utilisation 9](#_Toc186292100)

[Figure 2: Diagramme de séquence d'ajout d'un contact 10](#_Toc186292101)

[Figure 3:VSCODE 11](#_Toc186292102)

[Figure 4:Dart 12](#_Toc186292103)

[Figure 5:Git 12](#_Toc186292104)

[Figure 6:Enterprise Architect 12](#_Toc186292105)

[Figure 7:Flutter 13](#_Toc186292106)

# Introduction Générale

Dans le cadre de ce projet, nous avons développé une application mobile en utilisant **Flutter**, une technologie moderne et performante pour le développement multiplateforme. Cette application met en œuvre les opérations CRUD (Créer, Lire, Mettre à jour, Supprimer) pour la gestion d'une liste de contacts, un cas d'utilisation couramment rencontré dans les systèmes d'information.

L'objectif principal de ce projet est de démontrer la capacité à concevoir une application intuitive et fonctionnelle qui répond aux besoins des utilisateurs tout en exploitant les fonctionnalités avancées de Flutter. L'application permet :

* **L'ajout de contacts** : Les utilisateurs peuvent créer de nouveaux contacts avec un nom et un numéro de téléphone.
* **L'affichage des contacts** : Les contacts existants sont présentés dans une liste interactive.
* **La mise à jour des informations** : Les utilisateurs peuvent modifier les informations d'un contact spécifique.
* **La suppression de contacts** : Les contacts inutilisés ou incorrects peuvent être facilement supprimés.

L'application est conçue avec une interface utilisateur simple et réactive, basée sur les widgets de Flutter comme **ListView**, garantissant une expérience fluide et intuitive. Ce projet illustre également l'utilisation de mécanismes de gestion d'état pour maintenir la cohérence et les performances de l'application.

En conclusion, cette application constitue une base solide pour comprendre et développer des systèmes interactifs basés sur les opérations CRUD, tout en utilisant les outils modernes de développement mobile. Elle offre une excellente introduction aux possibilités offertes par Flutter dans la conception d'applications mobiles multiplateformes.

# Chapitre 1 : Contexte générale

1. Introduction

Dans un monde en constante évolution, les interactions sociales et professionnelles jouent un rôle fondamental dans nos vies quotidiennes. Que ce soit pour maintenir des relations personnelles ou gérer des contacts professionnels, disposer d’un outil efficace et intuitif pour organiser ces informations est devenu une nécessité. La gestion des contacts est donc une fonctionnalité essentielle dans de nombreuses applications modernes, qu'il s'agisse de solutions individuelles ou collaboratives.

Dans ce cadre, le projet à l’étude se concentre sur le développement d’une application mobile multiplateforme pour la gestion des contacts. Cette application est conçue en utilisant Flutter, un framework open-source puissant, connu pour sa capacité à produire des interfaces utilisateur riches, fluides et performantes sur plusieurs plateformes, notamment Android et iOS. Flutter permet une expérience de développement rapide et cohérente tout en garantissant une performance native optimale pour les utilisateurs finaux.

L'application proposée intègre les opérations CRUD (Création, Lecture, Mise à jour et Suppression) pour offrir une gestion complète et personnalisable d’une liste de contacts. Ces fonctionnalités de base seront enrichies par des options supplémentaires telles que la recherche rapide, le tri des contacts, ainsi que la synchronisation avec des bases de données cloud pour garantir une accessibilité et une sauvegarde optimales des données.

L'objectif principal de ce projet est d'offrir une solution innovante et conviviale, répondant aux besoins actuels des utilisateurs, tout en exploitant les avantages de Flutter pour assurer une compatibilité multiplateforme et une expérience utilisateur de haut niveau. En s’appuyant sur les principes modernes de développement logiciel, cette application s'inscrit dans une perspective de facilité d'utilisation, de performance et d'évolutivité, ouvrant ainsi la voie à des possibilités d’amélioration future pour répondre à des besoins toujours croissants.

1. Problématique

De nombreuses applications de gestion de contacts existantes souffrent de limitations en termes de performance, d’ergonomie ou de compatibilité multiplateforme. Ces contraintes limitent leur utilisation par un large public. La problématique principale de ce projet est donc la suivante :

Comment développer une application mobile intuitive, fluide et compatible multiplateforme pour la gestion des contacts, tout en assurant une expérience utilisateur optimale ?

1. Étude de l'existant

Plusieurs applications populaires comme Google Contacts ou iCloud Contacts permettent de gérer des listes de contacts. Cependant, ces solutions sont souvent liées à des écosystèmes fermés, ce qui complique leur portabilité. Contrairement à ces solutions, Flutter propose une alternative multiplateforme flexible et indépendante des écosystèmes propriétaires, permettant ainsi d’offrir une solution plus accessible et personnalisable.

1. Conclusion

À travers ce chapitre, nous avons mis en évidence l'importance croissante de la gestion des contacts dans notre quotidien et l'intérêt de concevoir des solutions adaptées pour répondre à ces besoins. Le projet présenté s'inscrit dans cette perspective en exploitant les atouts de Flutter pour développer une application multiplateforme, performante et intuitive.

En étudiant le contexte général et les défis liés aux applications existantes, nous avons pu identifier les limitations des solutions actuelles et définir une problématique claire visant à surmonter ces obstacles. Cette analyse nous permet de poser les bases solides pour la réalisation d'une application répondant aux exigences modernes, tout en offrant une expérience utilisateur optimale et une compatibilité élargie.

Chapitre 2: Analyse et conception

1. Introduction

Pour répondre à la problématique identifiée, une analyse détaillée des besoins fonctionnels, et techniques s’impose. Ce chapitre présente également la conception de l’application en s’appuyant sur des modèles UML pour mieux illustrer les interactions et les processus.

1. Analyse des besoins
2. Besoins fonctionnels

L’application doit répondre aux besoins suivants :

* Ajouter un contact : l’utilisateur doit être capable d’ajouter un contact en insérant les champs nécessaires (nom, numéro de téléphone)
* Lire la liste des contacts enregistrés.
* Modifier les informations d'un contact existant.
* Supprimer un contact.

1. Modélisation UML
2. Vue fonctionnelle (Diagramme des cas d’utilisation)

Le diagramme des cas d'utilisation illustre les interactions entre les utilisateurs et le système, mettant en évidence les principales fonctionnalités de l'application de gestion de contact.

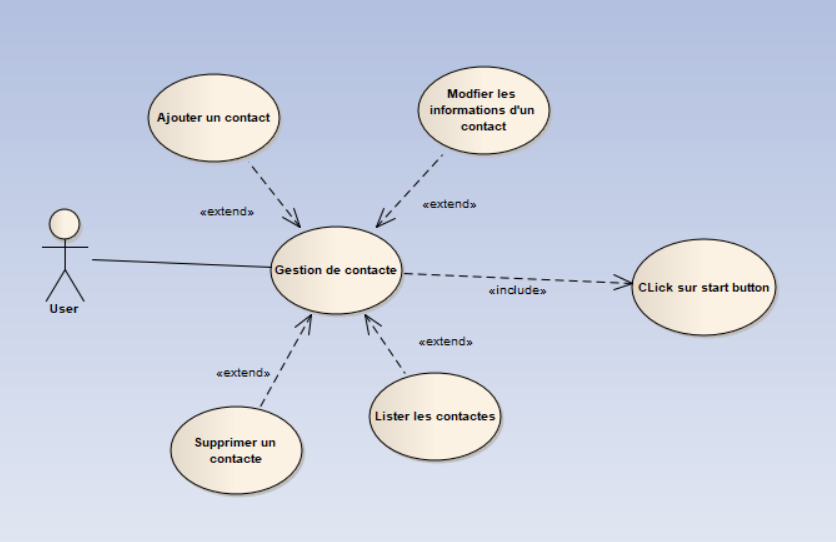


Figure 1: Diagramme des cas d'utilisation

Ce diagramme décrit les interactions entre l’utilisateur et les différents cas d’utilisation de l’application :

* Ajouter un contact.
* Modifier les informations d'un contact.
* Supprimer un contact.
* Lister les contacts existants.

1. Vue Dynamique (Diagramme de séquence)
   1. Diagramme de séquence pour ajouter un nouveau contact

A diagram of a system

Description automatically generated

Figure 2: Diagramme de séquence d'ajout d'un contact

Ce diagramme décrit les étapes liées à l’ajout d’un contact. Il inclut la vérification de l’existence du contact avant son enregistrement.

* 1. Diagramme de séquence pour modifier les informations d’un contact existant

A diagram of a system

Description automatically generated

Figure 3: Diagramme de séquence de modification d'un contact

Le diagramme suivant présente les étapes permettant de modifier un contact existant, incluant l’affichage des informations avant modification et la validation des données.

1. Besoins techniques et graphiques
   * 1. Visual Studio Code (VS Code)

A blue ribbon with a cross

Description automatically generatedUn éditeur de code source moderne et léger développé par Microsoft. Il est largement utilisé pour développer des applications dans divers langages de programmation grâce à son support étendu pour les extensions et les outils intégrés, comme le débogage, la gestion Git, et la coloration syntaxique.

Figure :VSCODE

* + 1. Dart

Un langage de programmation développé par Google, utilisé principalement pour construire des applications mobiles, web et desktop. Dart est souvent utilisé avec Flutter, offrant une syntaxe simple et des performances optimales pour le développement d'applications modernes.

A logo with a blue diamond

Description automatically generated

Figure :Dart

* + 1. Git

Un système de contrôle de version distribué qui permet de suivre les modifications apportées au code source. Git est essentiel pour le travail collaboratif, car il offre des fonctionnalités de fusion, de gestion de branches, et de sauvegarde des versions.

A red and black logo

Description automatically generated

Figure :Git

* + 1. Enterprise Architect

Un outil de modélisation UML complet utilisé pour concevoir, analyser et documenter des systèmes complexes. Il est couramment employé pour gérer les grandes architectures logicielles et les modèles de données, garantissant une planification structurée des projets.

A logo for a company

Description automatically generated

Figure :Enterprise Architect

* + 1. Flutter

Un framework open-source développé par Google qui permet de créer des applications multiplateformes (iOS, Android, web et desktop) à partir d'une base de code unique. Il est connu pour ses performances proches du natif et ses widgets personnalisables.

A logo with blue and grey letters

Description automatically generated

Figure :Flutter

1. Conclusion

L’analyse des besoins réalisée dans ce projet a été un élément crucial pour définir avec précision les objectifs fonctionnels et techniques de l’application. Elle a permis d'identifier les attentes des utilisateurs et de traduire celles-ci en exigences concrètes, garantissant ainsi que le développement réponde de manière optimale aux besoins exprimés.

Par ailleurs, la modélisation UML (Unified Modeling Language) a fourni une représentation claire et structurée des processus, des interactions, et des différents composants du système. Les diagrammes UML, tels que les diagrammes de cas d’utilisation, de classes et de séquence, ont joué un rôle essentiel en fournissant une vue d’ensemble cohérente et détaillée des aspects fonctionnels et comportementaux de l'application.

Cette phase de modélisation a servi de fondation solide pour la phase de développement, en facilitant la compréhension et la communication entre les membres de l’équipe de développement, tout en minimisant les risques d'erreurs ou de malentendus. Elle a également permis d’anticiper les problématiques potentielles et d’optimiser les choix techniques pour garantir une mise en œuvre efficace et conforme aux attentes.

Ainsi, cette combinaison d’une analyse approfondie des besoins et d’une modélisation précise constitue un pilier fondamental pour le succès du projet, assurant une transition fluide vers les étapes de conception détaillée et de développement pratique.

Chapitre 3 : Réalisation de l’application

1. Introduction

Dans ce chapitre, nous présentons les résultats du processus de développement de l’application en mettant l’accent sur les interfaces utilisateurs conçues pour la gestion des contacts. Chaque interface a été soigneusement développée pour répondre aux besoins des utilisateurs, en leur permettant d’accomplir efficacement les différentes tâches, telles que l’ajout, la modification ou la suppression de contacts.

Ce chapitre détaille les fonctionnalités de chaque interface et explique leur rôle dans le parcours utilisateur. Nous mettons en évidence l’organisation des éléments visuels et la navigation entre les écrans, en garantissant une expérience utilisateur fluide et intuitive. Un accent particulier est mis sur l’ergonomie et la clarté de l’information, des aspects cruciaux dans une application de gestion de contacts.

1. Interfaces de l’application
2. Accueil

A screen shot of a phone

Description automatically generated

Figure : Interface d'accueil

1. Gestion de contact
   1. Ajout de contact

A screenshot of a phone

Description automatically generated

Figure :Interface d'ajout

* 1. Modification de contact

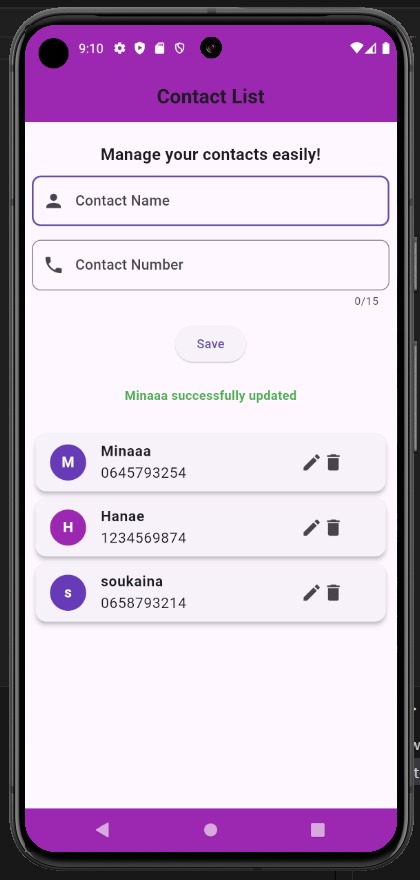


Figure : Modification d'un contact

* 1. Suppression de contact

A screenshot of a phone

Description automatically generated

Figure :Suppression d'un contact

1. Conclusion

Dans ce chapitre, nous avons examiné les interfaces de notre application. Ensuite, nous avons approfondi les fonctionnalités offertes par chacune d'entre elles.

# Conclusion Générale

En guise de conclusion, le projet de développement de l’application de gestion des contacts a permis de répondre efficacement à une problématique réelle en exploitant des technologies modernes comme Flutter. En proposant des opérations CRUD et une interface utilisateur intuitive, nous avons conçu une solution multiplateforme performante et accessible, répondant aux besoins des utilisateurs pour organiser et gérer leurs contacts de manière optimale.

La structuration du processus de développement, allant de l’analyse des besoins à la modélisation UML et à la réalisation concrète, a permis de garantir un système à la fois fonctionnel et ergonomique. Les choix technologiques, notamment l’utilisation de Visual Studio Code, Dart et Git, ont joué un rôle crucial dans la qualité et la fluidité du développement, tout en assurant une gestion efficace des versions et une collaboration harmonieuse.

Ce projet met en avant les avantages et les capacités de Flutter dans la création d’applications mobiles multiplateformes. Il ouvre également des perspectives d’évolution, comme l’ajout de fonctionnalités avancées telles que la synchronisation avec des bases de données cloud, des systèmes de recherche sophistiqués, ou encore l’intégration d’intelligence artificielle pour une gestion intelligente des contacts.

Enfin, ce projet illustre la capacité à relever des défis technologiques et conceptuels tout en offrant une solution conviviale et performante. Il constitue une base solide pour développer des applications futures et innovantes dans le domaine de la gestion mobile et au-delà.