

Ministry of Higher Education, Research and Innovation

**THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF BACHELOR OF SCIENCE**

**Specialty:** Computer Science

|  |
| --- |
| **Topic: Design of a mobile application for breeders** |

Presented and publicly defended:

**Zida Assomption Désiré Wendlasida**

**Academic year: 2023 – 2024**

**Supervised by**: Dr Pegdwendé N. Sawadogo

**Under the supervision by:** Dr Pegdwendé N. Sawadogo

Je dédie ce projet à mes parents et à mon frère pour leur soutien inébranlable et leur amour constant. Leur encouragement et leur confiance en moi ont été essentiels tout au long de ce parcours. Merci pour être ma source de force et d'inspiration.

**Remerciements**

Je tiens tout d’abord à exprimer ma profonde gratitude à Dieu, pour m’avoir accordé la santé et la force nécessaires pour mener à bien ces trois années de travail. Sans Sa grâce, ce parcours n'aurait pas été possible.

Je remercie également mes parents pour leur soutien indéfectible, leurs encouragements constants, et leur amour inconditionnel tout au long de cette aventure. Vous avez été ma plus grande source d'inspiration et de motivation.

Mes remerciements vont également à mon directeur de mémoire, Monsieur Sawadogo, pour son accompagnement, ses conseils précieux, et sa disponibilité tout au long de ce projet. Votre expertise et votre bienveillance ont été des atouts inestimables dans l’accomplissement de ce travail.

Je suis reconnaissant envers l'école BIT pour la formation de qualité que j'y ai reçue, ainsi qu'au chef de département pour son engagement et son leadership. Merci de nous avoir fourni un environnement propice à l'apprentissage et à l'épanouissement.

Je souhaite aussi remercier mes camarades de promotion, avec qui j'ai partagé tant de moments de travail, de doutes, et de réussite. Ensemble, nous avons surmonté les défis de ces années d'études, et c'est avec fierté que je regarde notre parcours.

Enfin, je remercie tous mes proches qui ont contribué de près ou de loin à ce projet. Votre soutien, votre compréhension, et vos encouragements ont été d'une grande aide tout au long de ce processus.

**Resumé**

AnimalCare est une application mobile innovante dédiée à la gestion complète des animaux de compagnie et du bétail. Conçue pour les éleveurs, les propriétaires d'animaux, et les passionnés d'animaux, l'application propose une série de fonctionnalités essentielles pour assurer le bien-être et l'entretien des animaux. AnimalCare permet aux utilisateurs de suivre l’état de santé de leurs animaux en enregistrant des informations telles que l’espèce, l’âge, les dates de vaccination, et l’historique des soins. Les utilisateurs peuvent ajouter facilement de nouveaux animaux et mettre à jour leurs informations à tout moment. L'application propose également une section de vente où les éleveurs peuvent exposer leurs animaux et produits dérivés à la vente via une plateforme dédiée, avec des annonces détaillées incluant des descriptions, des prix, et des photos. AnimalCare facilite également la connexion entre éleveurs, clients, ou amateurs d’animaux grâce à une fonctionnalité de chat qui permet de discuter en temps réel pour des ventes, des échanges de services, ou des conseils sur l’élevage. Les utilisateurs peuvent planifier et suivre les soins de leurs animaux, organiser les séances de vaccination, et recevoir des rappels pour les soins à venir. Avec une interface intuitive et un design moderne, l'application offre une navigation facile et une expérience utilisateur agréable, enrichie par des animations et des effets visuels. AnimalCare vise à transformer la gestion des animaux en simplifiant les tâches quotidiennes et en centralisant les informations cruciales dans une seule application conviviale.

**TABLE DE MATIERES**

Remerciement **i**

Résumé**1**

**Partie I étude théorique8**

**Chapitre 1 : Information générale sur le domaine de l’élevage** 9

1. Contexte D’étude **9**
2. Problématique 9
3. Objectifs9
4. Objectif Général9
5. Objectifs spécifiques10
6. Etat de l’art 10
7. Le concept de santé et de bien-être des animaux d’élevage10
8. Bien-être des animaux d’élevage 11
9. Importance d’une Gestion pour assurer la qualité et la productivité11
10. Revue des études existantes sur les plateformes intégrées pour l’élevage 12
11. Améliorations de la gestion et de la productivité12
12. Amélioration de la santé et du bien-être des animaux 13
13. Prévention des Maladies 13
14. Efficacité, Gain de temps et accès au conseil pour les Eleveurs 14
15. Analyse des Applications Existantes 14
16. Conclusion 15

**Chapitre 2 : Méthodologies et Approche Conceptuelles**16

1. Introduction à UML16
2. Définition16
3. Conception Orienté Objet 17
4. Langage de Modélisation UML17
5. Diagramme de classe 17
6. Diagramme de séquence 19
7. Diagramme d’activité 19
8. Application des diagramme UML dans le développement 20
9. Méthode de processus et de développement 20
10. Approche Conceptuelle Méthodologique 22
11. Conclusion 24

**Partie II : Conception et mise en œuvre**25

**Chapitre 1 : Conception, modélisation et approche conceptuelle** 26

1. Conception et Modélisation 26
2. Diagramme des cas d’utilisation26
3. Description textuelle du cas d’utilisation de la gestion des animaux 27
4. Diagramme de séquence27
5. Diagramme de classe 28
6. Approche Conceptuelle30
7. Conclusion32

**Chapitre 2 : Mise en œuvre et Résultat** 33

1. Mise en œuvre 33
2. Environnement de travail33
3. Environnements Matériels33
4. Environnement logiciel34
5. Résultat36

**Conclusion**39

**Liste des acronymes et abréviations**

**UML** : Unified Modeling Languag

**OO** : Conception orientée objet

**RAD** : Rapid Application Development

**JIT** : Just-In-Time (Compilation)

**AOT** : Ahead-Of-Time (Compilation)

**SCRUM** : Méthode agile de gestion de projet

**VS Code** : Visual Studio Code

**Firebase** : Plateforme de développement d'applications mobiles et web

**NoSQL** : Non-relational SQL (Système de gestion de base de données)

**SQL** : Structured Query Language

**Liste des Tableaux**

**Tableau 1 : Comparaison des méthodes de Développement** 38

**Tableau 2 : Scrum vs RAD** 38

**Tableau 3 : Cas d’utilisation vs User stories** 38

**Tableau 4 : Description textuelle du cas d’utilisation de la gestion des animaux** 38

**Tableau 5 : Flutter vs Réact Native** 38

**Tableau 6 : Firestore vs MySQL** 38

**Tableau 7 : Caractéristique des méthodes utilisées** 38

**Liste des Figures**

Figure 1 : Logo de AgriWebb15

Figure 2 : Logo de Moocall15

Figure 3 : Logo de UML17

Figure 4 : illustration d’un diagramme de classe19

Figure 5 : illustration d’un diagramme de séquence20

Figure 6 : illustration d’un diagramme d’activité20

Figure 7 : illustration du cas d’utilisation de l’application 27

Figure 8 : un diagramme de séquence pour l’achat d’un produit18

Figure 9 : Le diagramme de classe du projet AnimalCare30

Figure 10 : écran de chargement à l’entrée de l’application36

Figure 11 : écran d’accueil (catalogue)36

Figure 12 : écran login de l’application37

Figure 13 : écran Register de l’application 37

Figure 14 : Gestion des vaccins 37

Figure 15 : Ajout de vaccins 37

Figure 15 : Ajout de publication 37

**Introduction Générale**

Le domaine de l’élevage joue un rôle crucial dans l’économie et la sécurité alimentaire de plusieurs régions à travers le monde. Cependant, la gestion efficace des animaux, de leur santé, de leur reproduction, et de leur produit reste un défi important pour de nombreux éleveurs, Principalement ceux opérant dans des environnements aux ressources limitées. Avec l’avancer des technologies numériques, il devient de plus en plus évident que des solutions innovantes peuvent être développées pour répondre à ces défis. C’est dans ce contexte que s’inscrit AnimalCare, une application mobile dédiée à la gestion globale des animaux d’élevage.

AnimalCare est conçue pour fournir aux éleveurs une plateforme complète et intuitive pour suivre l’état de santé de leurs animaux, planifier les séances de vaccinations géré les produits dérivés des animaux, et facilité la vente de ces animaux ou produit. L’application offre également des fonctionnalités pour optimiser les interactions entre éleveurs et les clients, contribuant ainsi a amélioré l’efficacité et la rentabilité des opérations.

Ce document de soutenance présente le processus de conception, de développement et de déploiement de l’application AnimalCare. Nous y développerons les différents modules de l’application. Les choix technologiques effectués, ainsi que les défis rencontrés et les solutions apportées tout au long du développement. Ce projet s’inscrit dans une vision plus large de modernisation du secteur de l’élevage, en exploitant les possibilités offertes par les technologies mobiles pour améliorer la gestion quotidienne des animaux et la connectivité entre les acteurs de secteur.

En outre, AnimalCare s’intègre avec des technologies en matière de cloud computing et de bases de données en temps réel, assurant que les données des utilisateurs sont accessibles de n’importe où. En toute sécurité. Cette connectivité permet également aux éleveurs de collaborer avec d’autres acteurs su secteurs, d’accéder à des services de conseil en ligne et de rester informés a des dernières tendances et innovations en matière d’élevage. En combinant des fonctionnalités avancées avec une interfaces utilisateur simple et conviviale, AnimalCare vise à transformer la manière dont les éleveurs gèrent leurs activités au quotidien.

**Partie I : Etude Théorique**

**Chapitre 1 : Information générale sur le domaine de l’élevage**

1. **Contexte D’étude**

Le marché des animaux d’élevage connait actuellement une croissante rapide, Caractérise par une demande accrue pour des services diversifiés. Cette demande r assemble des soins de santé et d’hygiène pour les animaux, mais aussi des produits de vente variée=s et des conseils professionnels adaptés aux besoins des éleveurs. Ce phénomène est alimenté par une prise de conscience croissante de l’importance du bien-être de leur animaux. Aussi par une forte exigence de qualité et de traçabilité des produits animaux de la part des consommateurs.

Dans ce cas, les éleveurs se trouve face a des défis majeurs pour répondre a des obligations tout en gérant convenablement leurs opérations quotidiennes. Ils ont besoin de solution qui facilitent non seulement la gestion de santé et du bien-être de leur animaux, mais aussi la commercialisation de leurs produits et l’accès à des conseils spécialisés.

Une plateforme intégrée, telle que AnimalCare pourrait jouer un rôle important en répondant de manière plus efficaces et accessible a ces besoins variés. En misant sur les services de vente d’hygiène de santé et de conseil, cette application permettrait d’optimiser les interactions entre éleveur, client et fournisseur, créant ainsi un écosystème plus conforme et fonctionnel.

1. **Problématique**

Comment une plateforme intégrée comme AnimalCare peut-elle amélioré la vente des animaux d’élevage tout en optimisant les services offerts aux éleveurs, tout en optimisant les interactions entre éleveurs, client et fournisseurs et aussi en facilitant la commercialisation des produits ?

1. **Objectifs**
2. **Objectif Général**

Evaluer l’impact d’une plateforme intégrée sur la santé et le bien-être des animaux d’élevage ainsi que sur l’efficacité des interactions et des transactions entre éleveur, client et fournisseurs.

1. **Objectifs spécifiques**

* Analyser les avantages commerciaux d’une telle plateforme pour les fournisseurs de service et de produit pour animaux.
* Evaluer la satisfaction des éleveurs, des clients et des fournisseurs utilisant la plateforme AnimalCare.
* Identifier les défis et les obstacles à l’implémentation d’une plateforme intégrées.
* Proposer des améliorations basées sur les retours d’expérience des utilisateurs.

1. **Etat de l’art**

L’élevage moderne vise à renforcer la production animale tout en assurant le bien-être des animaux tout au long de leur vie. La santé et le bien-être des animaux d’élevage sont des concepts dispensables qui influenceront directement la quantité et la productivité des produits animaux. Dans cette section, nous explorerons en détails les notions de santé et de bien-être des animaux d’élevage en mettant l’accent sur l’importance d’une bonne gestion pour assurer la qualité et la productivité des productivité des produit animaux.

1. **Santé des animaux d’élevage**

La santé des animaux d’élevages est importante pour prévoir une production optimale. Voici quelques aspects de la gestion de la gestion de la santé animale :

* Prévention et traitement des Maladies : La prévention des maladies par la vaccination, les soins vétérinaires régulier et les pratiques d’hygiène est fondamentale. Une fois les maladies identifiées, un traitement efficace et rapide essentiel pour éviter la propagation et minimiser les pertes économiques.
* Condition de Vie Adéquates : Les conditions de logement des animaux doivent être propre, Sécurisées et confortables. Un bon espace de vie réduit le stress et le risque de maladies.
* Alimentation Equilibrée : Une bonne Nutrition est cruciale pour la croissance, la reproduction et la production des animaux.

1. **Bien-être des animaux d’élevage**

Le bien-être des animaux d’élevage va au-delà de la simple physique et inclut leur état mental et émotionnel. Voici les éléments majeurs :

* Les cinq libertés : Le bien-être animal est souvent évaluer selon les cinq libertés que sont :

-Liberté de la faim et la soif : chaque animal doit avoir accès a l’eau et a une nutrition adéquate.

-Liberté de l’inconfort : il faut fournir un environnement ayant un abri et une zone de repos propre.

-Liberté de la douleur, des blessures et des maladies : il faut une prévention et un traitement rapide des animaux.

-Liberté de la peur et de la détresse : Eviter les conditions de stress et assurer des soins appropriés.

1. **Importance d’une Gestion pour assurer la qualité et la productivité**

Une bonne gestion de la santé et du bien-être animal est essentielle pour maintenir des standards élevés de qualité et de productivité. Voici une bonne gestion qui peut avoir un impact positif :

* Amélioration de la productivité :

Un animal en bonne santé et bien traité sont plus productifs. Par exemple une poule pondeuse bien entretenu produit plus d’œufs, ainsi va de même que pour tous les animaux. Une meilleure croissance des animaux donne un taux élever des reproductions et une meilleure croissance des animaux.

* Qualité des produits :

Le bien-être des animaux a un impact direct sur la qualité des produits dérivées. La viande d’un animal stressés n’est pas meilleure que celle d’un animal en bonne forme.

* Réduction des couts :

Un bon programme de santé et de bien-être animal peut réduire les couts vétérinaires et les pertes dues aux maladies. Moins d’animaux malades signifie moins de dépenses en médicaments et traitement, ainsi qu’une minimisation des pertes de production.

* Acceptation social et règlementaire :

Dans plusieurs pays, le bien-être animal est devenu un enjeu sociétal majeur. Les consommateurs sont de plus en plus soucieux des conditions dans lesquelles les animaux sont élevé. Une gestion respectueuse du bien-être animal peut améliorer l’image de marque des produits et répondre aux attentes des consommateurs et des régulateurs. Cela peut également aider à accélérer a de nouveaux marchés ou les normes de bien-être animal sont rigoureux.

* Durabilité :

Le bien-être animal contribue également à la durabilité de l’élevage. Des pratiques d’élevages respectueuses de l’animal sont souvent en ligne avec des pratiques respectueuses de l’environnement, réduisant ainsi l’empreinte écologique de l’élevage. Par exemple des habitudes qui minimise le stress animal peuvent également diminuer les besoins en ressources et l’impact environnemental.

1. **Revue des études existantes sur les plateformes intégrées pour l’élevage**

L’intégration des plateformes numériques dans l’industrie de l’élevage a connu une montée en popularité, répondant a la demande croissante de solutions innovent pour améliorer la gestion des élevages, le bien-être animal et aussi la productivité. Cette revue met en lumière les bénéfices observés et les défis rencontrés dans cette transformation numérique, en s’appuyant sur des études et de recherches antérieures.

1. **Améliorations de la gestion et de la productivité**

* **Optimisation des ressources**

Les plateformes intégrées jouent un rôle crucial dans l’optimisation de la gestion des ressources, notamment en ce qui concerne l’alimentation et l’eau. Elles fournissent des données en temps réel qui permettent aux éleveurs de prendre des décisions éclairées. Par exemple, en Australie l’utilisation de capteurs connectés pour surveiller les besoins nutritionnelles des troupeaux a permis de réduire les couts d’alimentations de 15% tout en augmentant la productivité. Cette optimisation se traduit par une utilisation plus efficace des ressources et une amélioration des rendements.

1. **Amélioration de la santé et du bien-être des animaux**

L’utilisation de plateforme intégrées, combinant des capteurs biométriques et des algorithmes de machine Learning, joue un rôle crucial dans la prévention des maladies. Ces technologies permettent de surveiller en continu les indicateurs de santé des animaux, identifiant les risques avant l’apparition des symptômes. Une étude réalisée en Amérique du Nord a révélé que l’intégration de ces outils numériques a permis de réduire la mortalité des veaux de 20%. Cela met en évidence l’importance des technologies avancé pour une gestion proactive et efficace de la santé animale, minimisant ainsi les risques de maladie graves.

1. **Réduction du stress des animaux**

La surveillance constante et l’environnement contrôlés offerts par les plateformes intégrées permettent de minimiser les facteurs de stress pour les animaux. Au japon, également une étude a montré que l’utilisation de ces plateformes a réduit le taux d’hormone de stress chez les bovins de 25%. Un environnement moins stressant favorise la santé et le bien-être générale des animaux, ce qui peut également améliorer la qualité des produits animaux.

1. **Efficacité, Gain de temps et accès au conseil pour les Eleveurs**

Les plateformes offrent également des modules de formations et de conseils en ligne pour promouvoir les meilleures pratiques d’élevage. En Inde, une étude a révélé que l’accès à ces ressources a considérablement amélioré les compétences et les connaissances des éleveurs, leur permettant d’adopter des pratiques plus rentables.

Les plateformes intégrées automatisent de nombreuses tâches quotidiennes telle que l’alimentation et le nettoyage, libérant ainsi du temps pour les éleveurs pour se concentrer sur des activités a plus forte valeur ajoutée. En nouvelle Zélande, une étude a également indiqué que les éleveurs utilisant des plateformes intégrées ont réduit leur temps de travail manuel de 40% l’automatisation contribue à une gestion plus efficace de l’élevage et réduit la charge de travail physique des éleveurs.

1. **Analyse des Applications Existantes**

Plusieurs applications ont été développées pour résoudre les besoins de l’industrie de l’élevage en termes de gestion et d’efficacité. Nous avons un aperçu de certaine applications existantes :

* AgriWebb : elle est destinée à la gestion des fermes, ce qui offre des outils pour le suivi des animaux, la gestion des champs, et l’enregistrement des traitements vétérinaires. Elle facilite également la conformité réglementaire.

Lien : <https://www.agriwebb.com>



*Figure 1 : logo de Agriwebb*

* Moocall : Un système de surveillance basé sur des capteurs qui alerte les éleveurs sur le début du travail chez les vaches. Cela aide à minimiser les pertes dues à des naissances non supervisées.

Lien : <https://www.moocall.com>



*Figure2 : logo de Moocall*

1. **Conclusion**

En comparaison avec les applications existante, notre application se départage par son intégration complète des fonctionnalités essentielles à la gestion de l’élevage, incluant :

* Optimisation des contacts entre éleveur et clients : En facilitant les interactions, notre application permet une meilleure mise en relation pour la vente d’animaux et de produits animaux.
* Approche personnalisée de la gestion des animaux : Contrairement a certaines applications généralistes, notre solution offre une personnalisation approfondie pour s’adapter aux besoins spécifiques des éleveurs.
* Interface utilisateur intuitive et moderne : Notre application met l’accent sur l’expérience utilisateur fluide et moderne, ce qui est souvent un point faible dans les applications existantes.
* Fonctionnalité de formation et d suivis en temps réel : Notre application met intègre des modules de formations directement dans l’application, nous offrons un accès direct à l’éducation continue pour les éleveurs.

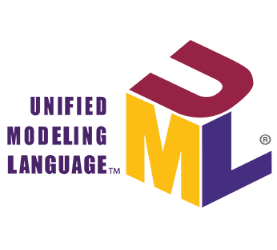
En résumé, AnimalCare intègre les meilleures pratiques de l’industrie tout en offrant des fonctionnalités uniques qui répondent aux besoins spécifiques des éleveurs modernes. Elle représente une avancée significative dans l’utilisation de la technologie pour transformer l’industrie de l’élevage.

**Chapitre 2 :** **Méthodologies et Approche Conceptuelles**

1. **Introduction à UML**
2. **Définition**

Le Unified Modeling Language (UML) est un outil essentiel pour la modélisation de système logiciels complexes. Il offre une série de notations graphique qui aident les concepteurs et les développeurs à représenter différent aspect d’un système de manière compréhensible. En utilisant UML, les équipes peuvent créer des représentations visuelles qui facilitent la communication, la documentation et la compréhension des systèmes. Ce qui est important pour le succès des projets de développement.

UML comprend divers types de diagrammes qui se concentre sur la structure, le comportement et les interactions du système. Ces diagrammes aident à capturer les exigences à concevoir les architectures, à planifier les interactions et à définir les flux de travail. Les principaux diagrammes incluent les diagrammes de classes, de séquence et d’activités, chacun ayant un rôle spécifique dans le processus de modélisation.



*Figure 3 : logo de UML*

1. **Conception Orienté Objet**

La conception orienté objet (OO) est une approche qui permet de modéliser des systèmes logiciels en termes d’objet qui interagissent entre eux.

* Modélisation des Entités : Nous avons utilisé des concepts orienté objet pour modéliser les entités principales du système, telle que les éleveurs, les animaux et les produits. Chaque entité principale du système, telle que les éleveurs, les animaux, les produits. Chaque entité est représentée par une classe qui englobe les données et les comportements qui y sont associés.
* Principe SOLID : Les principes SOLID sont des ensembles de bonnes pratiques de conception qui visent à rendre le code plus compréhensible, flexible et maintenable. Ils renferment des concepts tels que la responsabilité unique des dépendances.

1. **Langage de Modélisation UML**

UML offre un ensemble diversifié de diagramme qui facilitent la visualisation et la spécialisation des différents aspects du système.

1. **Diagramme de classe**

Le diagramme de classe est un élément essentiel de l’UML (Unified Modeling Language) qui montre la structure statique d’un système. Il illustre les classes du système, leur attributs, méthodes et les interactions, ce qui facilite la conception de la base de données et l’implémentation du système.

**Composant d’un diagramme de classe**

* **Classe**

-Description : Représente une entité du système avec ses propriétés et comportement. Chaque classe est décrite par un rectangle qui indique le nom de la classe.

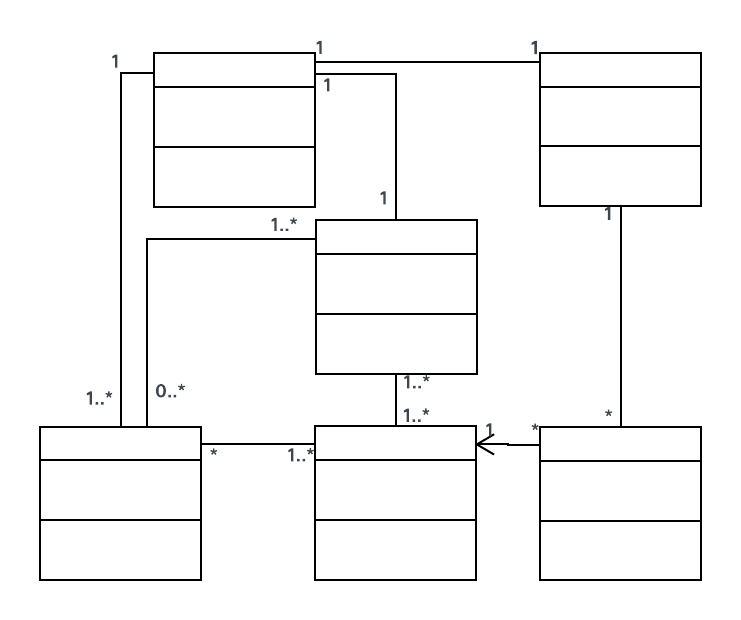
-Section

. Nom de la classe : La partie supérieur du rectangle qui indique le nom de la classe.

. Attributs : la partie du milieu qui liste les propriétés ou caractéristiques de la classe, Généralement avec le format nom : type.

. Méthode : La partie inferieur qui énumère les opérations ou fonctions que la classe peut réaliser, souvent avec le format nom(paramètre) : typeRetour.

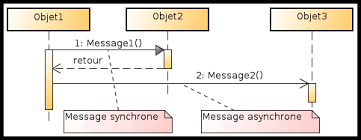
* Association :
* Description : Montre une relation entre deux classes. Elle est représentée par une ligne reliant les deux classes, et peut inclure une flèche pour indiquer la direction de la relation.
* Détails : Contrairement à la composition, les parties peuvent exister indépendamment de l’ensemble. La durée de vie de la partie n’est strictement liée à celle du tout.
* Dépendance :
* Description : Indique qu’une classe utilise ou dépend d’une autre classe. Cette relation est représentée par une ligne en pointillé avec une flèche pointant vers la classe dont la dépendance est faite.



*Figure 4 : illustration d’un diagramme de classe*

1. **Diagramme de séquence**

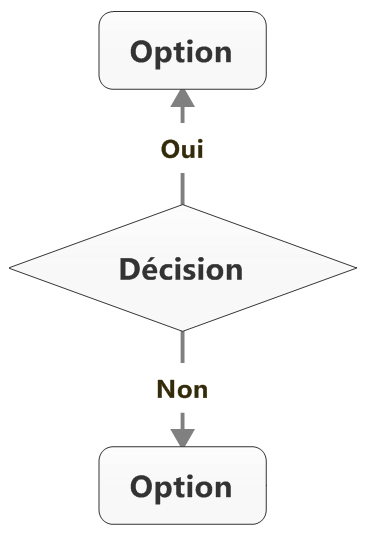
Le diagramme de classe est un élément central de l’UML utilisé pour représenter la structure statique d’un système. Il montre les classes du système, leur attributs, méthodes et les relations qui les relient. Le diagramme est essentiel pour comprendre la base de données et la mise en œuvre du système.



*Figure 5 : illustration d’un diagramme de séquence*

1. **Diagramme d’activité**

Les diagrammes d’activité sont des outils clés en UML utiliser pour capturer les flux de travail et les processus métiers. Il représente les étapes séquentielles, les choix conditionnels et les parallélismes au sein d’un processus. Ces diagrammes permettent de visualiser de manière claire et détailler de la séquence des activités et les interactions entre elles.



*Figure 6 : illustration d’un diagramme d’activité*

1. **Application des diagramme UML dans le développement**

Les diagramme UML sont des outils puissants qui jouent un rôle dominant dans le développement de logiciel en facilitant la spécification, la visualisation et la documentation des systèmes.

* Spécification et Visualisation : Les diagrammes UML permettent de spécifier clairement les exigences et de visualiser les comportements attendus du système. Ils offrent une représentation graphique qui aide à la communication et a la compréhension entre les développeurs et les parties prenantes.
* Documentation et Communication : En fournissant une documentation détaillée et standardisée, UML améliore la communication entre les membres de l’équipe de développement et les autres parties prenantes, assurant une compréhension commune des fonctionnalités et de l’architecture du système.

En intégrant UML dans notre méthodologie, nous avons créer une architecture logicielle bien structurée, flexible et prête à évoluer avec les besoins futurs, tout en assurant une communication claire et efficace au sein de l’équipe.

1. **Méthode de processus et de développement**

Le développement de notre application de gestion de l’élevage a nécessité l’adoption de méthodologies moderne pour assurer flexibilité et efficace. Nous avons choisi de méthodes qui favorisent l’adaptabilité aux besoins changeant des utilisateurs tout en garantissant une livraison rapide de solutions fonctionnelles.

* **Méthode Agile**

La méthode agile est une approche de développement logiciel qui privilégie les interactions humaines, les livraisons rapides et une capacité d’adaptation aux changements. Voici comment nous l’avons appliquée :

* Principe Agile : L’agilité repose sur douze principes, tels que la satisfaction du client par une livraison rapide et continue de logiciel fonctionnel, et l’acceptation du changement, même tard dans le développement. Ces principes ont guidé notre processus, assurant que nous restons alignés avec les attentes des utilisateurs finaux.
* Scrum : Scrum est un cadre agile spécifique qui structure le travail en sprints court et irrégulier, généralement de deux a quatre semaines. Chaque sprint commence par une planification ou les priorités sont définies, suivie d’un développement intensif, et se termine par
* Une planification ou les priorités sont définies, suivie d’un développement intensif, et se termine par une revue et une rétrospective pour évaluer le travail accompli et identifier les améliorations possibles.

|  |  |  |
| --- | --- | --- |
| Critère | Méthode Agile | Méthode traditionnelle |
| Flexibilité | Haute, permet des changements même tardif | Faible, difficile d’incorporer des changements |
| Livraison | Livraison fréquente | Livraison unique en fin de cycle |
| Collaboration | Forte collaboration entre les équipes | Collaboration limitée à certaines étapes |
| Documentation | Allégée, évolue avec le projet | Lourde, établie en début de projet |
| Approche | Itérative et incrémentale | Linéaire |

*Tableau 1 : Comparaison des méthodes de Développement.*

|  |  |  |
| --- | --- | --- |
| Critère | Scrum | RAD |
| Durée des cycles | Courts sprints de 2 à 4 semaines | Phase rapide avec des prototypes |
| Focus | Livraison d’une version fonctionnelle a chaque sprint | Livraison rapide grâce à des prototypes |
| Adaptabilité | Haut niveau flexibilité | Flexibilité élevée, mais moins structurée |
| Documentation | Adaptative, ajustée à chaque sprint | Minimaliste, basée sur les prototypes |
| Communication | Forte communication continue | Forte, mais parfois centrée sur l’interface utilisateur |

*Tableau2 : Scrum vs RAD.*

1. **Approche Conceptuelle Méthodologique**

Une approche conceptuelle méthodologique bien établie est cruciale pour structurer et organiser le développement de logiciel complexe. Notre approche se concentre sur l’analyse des exigences et la conception orienté objet.

* **Analyse des Exigences**

L’analyse des exigences est la première étape importante dans le développement de logiciels, permettant de définir ce que le système doit accomplir

* Collecte des Exigence : Nous avons mené des ateliers et des entrevues avec les parties prenantes pour recueillir des exigences détaillées. Ces intéractions ont permis de comprendre les besoins spécifiques des éleveurs, des clients et des fournisseurs de services.
* Cas d’utilisation : Les cas d’utilisation sont des descriptions narratives des interactions entre les utilisateurs et le système. Ils ont été utilisés pour documenter comment le système devrait se comporter dans divers scénarios, offrant une base solide pour la conception et le développement.
* **Analyse des Exigence : Cas d’utilisation vs User storie**

|  |  |  |
| --- | --- | --- |
| **Critère** | **Cas d’utilisation** | **User Stories** |
| Format | Narratif détaillé avec diagramme | Simple phrase : « En tant que…Je veux…afin de…. » |
| Niveau de détails | Très détaillé | Moins détaille, focalisé sur les besoins utilisateurs |
| Documentation | Lourde, nécessaire pour les systèmes complexes | Légère, évolue avec les besoins |
| Objectif | Capturer tous les scénario possible | Capturer les besoins utilisateurs de manière simple |
| Utilisation | Projet complexe avec beaucoup d’interaction | Projet agile, agile centré sur les utilisateurs |

*Tableau3 : Cas d’utilisation vs User stories.*

1. **Conclusion**

Dans ce chapitre dédié a la méthodologie et a l’approche conceptuelle, nous avons exploré les outils essentiels pour la conception et la modélisation de systèmes complexes. Les diagrammes de classe et d’activité jouent un rôle fondamental dans la structuration et la visualisation des processus et des entités du système.

Les diagrammes de classe permettent de définir la structure statique du système en illustrant les classes, leur attributs, méthodes, et les relations entre elles. Cette représentation est cruciale pour la conception de la base de données et les relations entre elles. Cette représentation est cruciale pour la conception de la base de données et l’organisation des différentes entités du domaine, facilitant ainsi une compréhension claire de la structure et des interactions des classes.

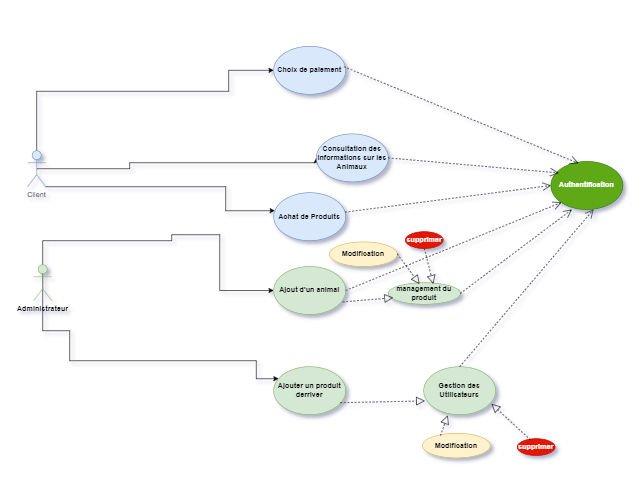
En parallèle, les diagrammes offrent une vue dynamique des processus métiers, montrant les flux de travail, les choix conditionnels, et les tâches parallèles. Ils sont particulièrement utiles pour modéliser des processus complexes, permettant ainsi une meilleure visualisation des étapes séquentielles et des interactions dans les processus d’affaire.

En intégrant ces éléments dans la phase de conception, nous pouvons mieux comprendre les exigences du système, optimiser les processus et garantir une mise en œuvre cohérente et efficace. La combinaison des diagrammes de classe et d’activité assure une approche systématique et complète, offrant une base solide pour l’élaboration de solution et adaptées aux besoins spécifiques du projet.

**Partie II : Conception et mise en Œuvre**

**Chapitre 1 : Conception, modélisation et approche conceptuelle**

1. **Conception et Modélisation**
2. **Diagramme des cas d’utilisation**

****

*Figure 7 : illustration de cas d’utilisation de l’application AnimalCare*

Bien avant l’inscription, l’utilisateur n’a que la possibilité de regarder le catalogue de produit présent sur la plateforme. Ensuite il pourra s’inscrire pour devenir un client sur notre application mobile.

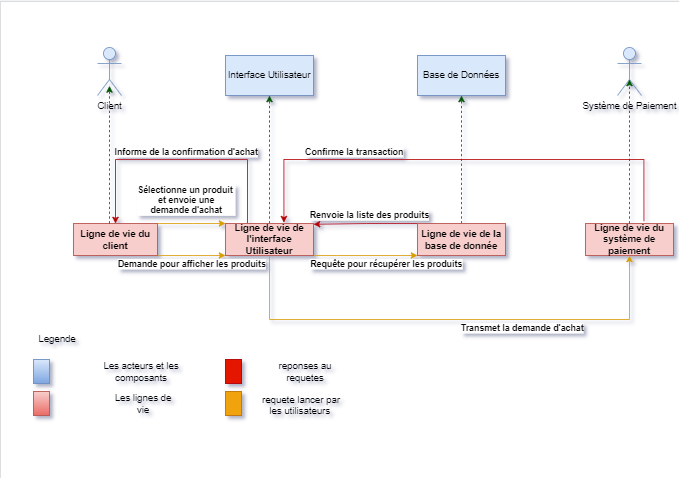
Après s’être enregistrer il a le droit maintenant de poursuivre les opérations jusqu’à l’achat du produit en ligne sur notre application.

1. **Description textuelle du cas d’utilisation de la gestion des animaux**

|  |  |
| --- | --- |
| **Titre** | **Description Textuelle** |
| Nom du cas d’utilisation | Gestion des animaux |
| Acteurs | Utilisateur, Système de gestion, Base de données |
| Objectif | Permettre à l’utilisateur de gérer les informations de ses animaux |
| Préconditions | Les informations de l’animal sont mises à jour et enregistrer dans la base de données |
| Flux Principal | 1.L’utilisation accède à la section de gestion des animaux.  2.il sélectionne un animal dans la liste.  3.l’utilisateur modifie les informations de l’animal.  4.le système enregistre les modifications. |
| Extension Alternatif | 1.l’utilisateur n’est pas connecté, le système redirige vers la page de connexion.  2.Si la l’animal sélectionner n’existe n’existe pas, le système affiche un message d’erreur. |

*Tableau4* : *Description textuelle du cas d’utilisation de la gestion des animaux*

1. **Diagramme de séquence**



*Figure 8 : le diagramme de séquence pour l’achat d’un produit sur AnimalCare.*

1. **Diagramme de classe**

Le diagramme de classe comprend les entités suivantes :

Classe utilisateur Comprend les attributs

 :

* Nom, Prénom, email, motDePasse, et adresse, avec des méthodes seConnecter(), créerCompte(), et mettreAJourInformation().

Classe Animal Comprend les attributs

 :

* Nom, age,aspece,dateNaissance, photo, et historiqueSoins, avec les methodes planifierViste() et ajouterVaccination().

Classe Vétérinaire Comprend les attributs

 :

* Nom, contact, et spécialité, ainsi que les méthodes planifierVisite() et ajouterVaccination().

Classe Vaccination Comprend les attributs

 :

* Date, type de vaccin et vétérinaire avec la méthode enrégistrerVaccination().

Classe Produit Comprend les attributs

:

* Nom, description, prix, photos, avec les méthodes ajouterProduit() et modifierProduit().

Classe Vente Comprend les attributs

 :

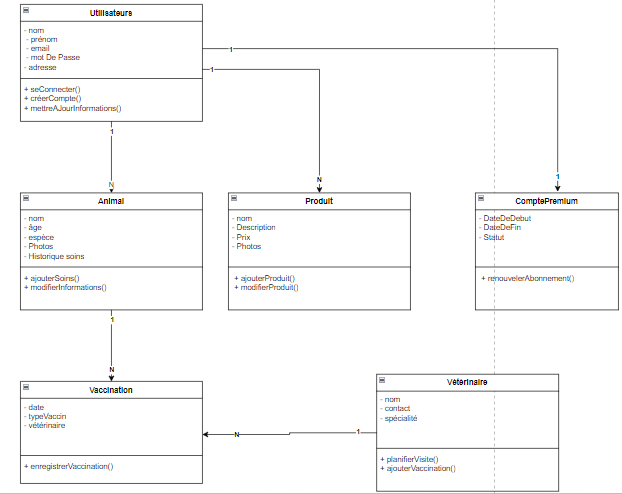
* Animal, produit, prix, et dateVente avec la méthode enregistrerVente().

Classe ComptePrenium comprends les attributs :

* dateDébut, dateExpiration et statut, avec la méthode renouvelerAbonnement().

Relation entre les classes :

* Un utilisateur peut posséder plusieurs Animaux (1 : N).
* Un Animal peut recevoir plusieurs vaccinations, chaque vaccination étant administrée par un vétérinaire (1 : N).
* Un Utilisateur peut vendre plusieurs Produits (1 : N).
* Un Utilisateur peut vendre plusieurs Produit (1 : N).
* Un Produit peut être associé à plusieurs ventes, chaque vente étant spécifique a un produit et un acheteur (1 : N).
* Un Utilisateur peut avoir un ComptePrenium (1 : N).



*Figure 9 : le diagramme de classe du projet.*

1. **Approche Conceptuelle**
2. **Comparaison de différentes méthodes de développement**

**Tableau de comparaison entre flutter et Réact**

|  |  |  |
| --- | --- | --- |
| **Critère** | **Flutter** | **Réact Native** |
| Language | Dart | JavaSrcipt |
| UI ET UX | Proche du natif grâce a la compilation JIT et AOT | Bonne performance, mais parfois inférieur au natif |
| Hot Reload | Oui, permet de voir les changements instantanément | Oui, permet de voir les changements instantanément |
| Ecosystème | Plus jeune, en croissance rapide | Mûr, large communauté et écosystème établi |
| Plugins et Bibliothèques | En croissance, certains plugins nécessitent du code natif | Principalement pour Ios et Android, le support web est limité |
| Apprentissage | Langage Dart peut nécessiter un apprentissage supplémentaire | JavaScript est largement connu, plus facile pour les développeurs web |
| Support des plateformes | Support natif pour Ios et Android, ainsi que pour le web et le desktop | Bonne intégration avec les fonctionnalités natives grâce à des Wrappers et des modules |
| Intégration Native | Peut nécessiter du code natif pour certaine fonctionnalités via des plugins | Bonne intégration avec les fonctionnalités native grâce à des Wrappers et des modules natifs |

*Tableau5 : Flutter vs Réact Native.*

**Tableau de comparaison entre firestore et MySQL**

|  |  |  |
| --- | --- | --- |
| Critère | Firestore | MySQL |
| Temps réel | Support natif des mises a jour en temps réel | Pas de support natif, mais peut être réaliser avec solutions externes |
| Transactions | Transaction atomique, mais avec limitations sur les opérations complexes | Transaction ACID robuste et support complet des opérations complexes |
| Flexibilité du schéma | Schéma flexible, idéal pour des données non structurées ou évolutives | Schéma fixe, nécessite des migrations pour modifier la structure des données |

*Tableau6 : Firestore vs MySQL*

Pour le développement de notre application nous avons choisi flutter comme méthode de développement et firestore pour la base de données.

1. **Conclusion**

Ce premier chapitre a permis de poser les bases essentielles pour la réalisation du projet en mettant l'accent sur la conception, la modélisation et l'approche conceptuelle. Nous avons d'abord défini les objectifs et les besoins spécifiques du projet, ce qui a conduit à l'élaboration d'une architecture conceptuelle adaptée. La modélisation des données et des processus a été abordée de manière rigoureuse, garantissant une structure solide et cohérente pour le développement ultérieur. L'approche conceptuelle adoptée a permis de clarifier les différentes composantes du système, tout en assurant une cohérence entre les besoins des utilisateurs et les fonctionnalités offertes par l'application. Les outils et méthodes utilisés, tels que les diagrammes UML et les modèles de données, ont contribué à structurer et formaliser la solution envisagée, minimisant ainsi les risques d'incohérences ou de malentendus lors des phases de développement. En somme, ce chapitre a jeté les bases théoriques et méthodologiques indispensables pour la suite du projet, garantissant une progression fluide et cohérente vers la réalisation des objectifs fixés.

**Chapitre2 : Mise en Œuvre et Résultat**

1. **Mise en œuvre**
2. **Environnement de travail**

**Environnements Matériels**

****Processeur :Intel(R) Celeron (R) N4000 CPU @

1.10GHz 1.10 GHz Mémoire RAM : 8,00 Go

Disque dur : 230go

Mémoire RAM : 12go

Disque Dur: 256go

Version android: 13

1. **Environnement logiciel**

* **Flutter**



Flutter est un ensemble de Framework open-source développé par Google qui permet de créer des applications mobiles, web et desktop à partir d’une base de code. En utilisant le langage Dart, Flutter offre des capacités de développement s’interfaces d’utilisateur riches et performantes. Le Framework se départage des autres de par sa capacité de hot Reload, qui permet aux développeurs de voir instantanément les changements apportés au code sans avoir à redémarrer l’application. Bien que Flutter soit très puissant pour le développement cross-plateforme, Il est aussi jeune par rapport à d’autres Framework comme Réact Native, et certains plugins peuvent ne pas être aussi bien maintenus ou complets.

Lien d’accès : <https://flutter.dev>

* **Draw.io**

Draw.io est un outil en ligne gratuit conçu pour la création de diagramme et de maquettes. Il est particulièrement utile pour concevoir des diagrammes de flux, des organigrammes, et des maquettes d’applications. Draw.io est accessible directement depuis un navigateur web et permet la collaboration en temps réel avec plusieurs utilisateurs. Il s’intègre facilement avec des services de stockage cloud comme Google Drive et Dropbox. Cependant, bien que Draw.io soit gratuit et accessible, il peut manquer de certaines fonctionnalités avancées présentes dans des outils payant comme Microsoft Visio et peut devenir lent lorsqu‘on travaille avec des diagrammes très complexes.

Lien d’accès : <https://app.diagrams.net/>

* **Visual Studio Code**

Visual studio code est un éditeur de code-open source développée par Microsoft. Il est largement utilisé pour sa légèreté, ses performances rapides et ses capacités de personnalisation. Vs code prend en charge une très grande variété de langage de programmation grâce a un large éventail d’extension, et il offre une bonne intégration avec des systèmes de contrôle de version comme Git, ainsi que des outils de développement comme les débutants en raison de ses nombreuses fonctionnalités et options, et peut nécessiter une configuration et une personnalisation spécifique pour certain langages et Framework.

Lien d’accès : <https://code.visualstudio.com/>

* **Firestore**

. Firestore est une base de données NoSql cloud développée par google, concue pour stocker et synchroniser des données en temps réel.Firestore s’intègre facilement avec les autres services Firebase comme Firebase Authentification et cloud Function et offre un modèle de données flexible qui convient bien aux structures de données non rigides.Cependant, le coût d’utlisation peut devenir élévé avec une utilisation intensive, et les requêtes complexes peuvent être plus difficiles a optimiser par rapport aux base de données relationnelle comme MySql.

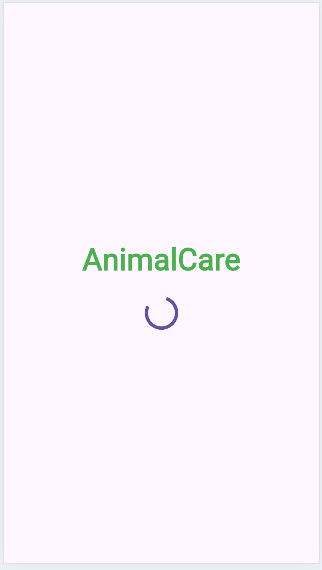
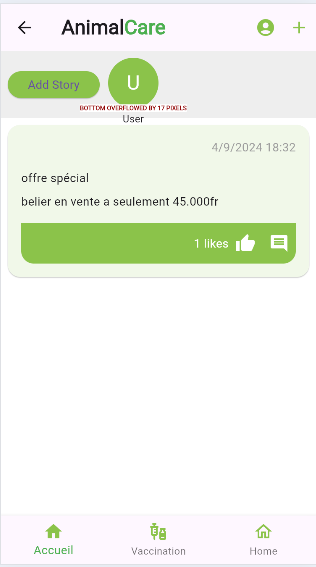
Lien d’accès : <https://firebase.google.com/docs/firestore>

1. **Caractérisque des méthodes utilisée**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Outil/Technologie** | **Type** | **Langage** | **Description** | **Principales Fonctionnalités** |
| Flutter | Framework UI | Dart | Framework open-source pour le développement d'applications mobiles, web et desktop. | Développement cross-platform,  Hot Reload, Widgets riches et personnalisables |
| Draw.io | Outil de Diagramme | N/A | Outil en ligne pour la création de diagrammes, maquettes et organigrammes. | Collaboration en temps réel, Intégration avec le stockage cloud, Gratuit et accessible |
| Visual Studio Code (VS Code) | Éditeur de Code | N/A | Éditeur de code open-source avec support pour divers langages de programmation et outils de développement. | Extensions variées, Intégration Git, Performances rapides, Personnalisation |
| Firestore | Base de Donné NoSql | N/A | Base de données cloud pour le stockage et la synchronisation des données en temps réel. | Scalabilité élevée, Intégration avec Firebase, Modèle de données flexible |

*Tableau7 : Caractérisque des méthods utilisée*

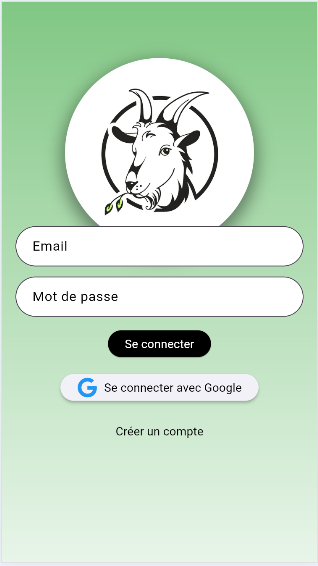
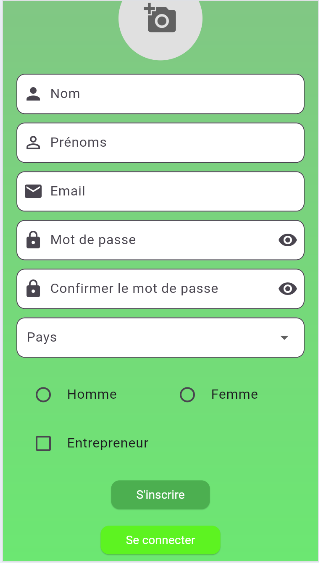
1. **Résultat**

 ****

*Figure 10 : écran de chargement à l’entrée de l’app Figure 11 : écran*

Lorsque l’application est démarrée la page splash *d’accueil (catalogue)*

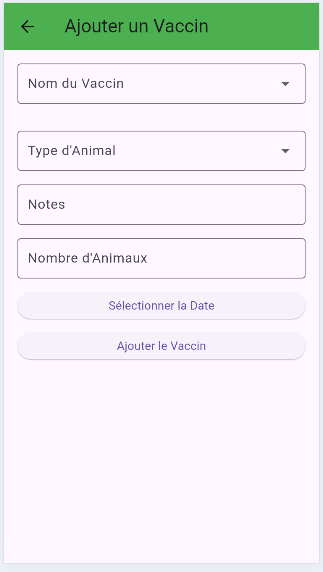
S’affiche durant 5s

 ****

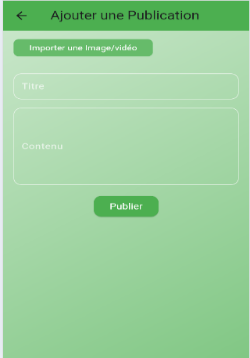
*Figure 12 : écran de login Figure 13 : écran de Registre.*

Cette page est nommée Login. Cette interface permettra Cette page permettra aux

Aux Utilisateurs de se connecter ou de créer un compte nouveaux venu de s’enregistrer.

** **

*Figure 14 : Gestion des vaccins Figure 15 : l’ajout d’un vaccin*

****

*Figure 16 : l’ajout d’une publication*

1. **Conclusion**

Le projet "AnimalCare" s'inscrit dans une démarche visant à répondre aux défis contemporains de la gestion des animaux domestiques et d’élevage. Grâce à une approche intégrant à la fois la gestion des soins, l'organisation des séances de vaccination, et une plateforme de vente, cette application mobile offre une solution complète et innovante aux éleveurs, vétérinaires, et propriétaires d'animaux. Les résultats obtenus au cours de ce projet montrent que "AnimalCare" facilite non seulement la gestion quotidienne des animaux, mais contribue également à améliorer la communication et les transactions au sein de ce secteur. En apportant une réponse adaptée aux besoins actuels, cette application a le potentiel de devenir un outil incontournable dans le domaine de la gestion animale. Les perspectives d'évolution de "AnimalCare" incluent l'intégration de nouvelles fonctionnalités et l'expansion à d'autres marchés, afin de continuer à accompagner les utilisateurs dans la gestion efficace et moderne de leurs animaux.

**Bibliographie**

**Google Firebase - "Firebase Documentation." Google. Accessible via :** [**https://firebase.google.com/docs**](https://firebase.google.com/docs)

**Flutter - "Flutter Documentation." Google Developers. Accessible via :** [**https://flutter.dev/docs**](https://flutter.dev/docs)

**Draw.io - "Draw.io Online Diagram Software." Accessible via :** [**https://www.diagrams.net/**](https://www.diagrams.net/)

**Visual Studio Code - "Visual Studio Code Documentation." Microsoft. Accessible via:** [**https://code.visualstudio.com/docs**](https://code.visualstudio.com/docs)

**Firestore - "Cloud Firestore Documentation." Google. Accessible via :** [**https://firebase.google.com/docs/firestore**](https://firebase.google.com/docs/firestore)

**Meat & Livestock Australia (MLA**

[**https://www.mla.com.au/research-and-development/Animal-Health-welfare/Digital-technology**](https://www.mla.com.au/research-and-development/Animal-Health-welfare/Digital-technology)