**Rapport Détaillé du Projet Clinique Vétérinaire**

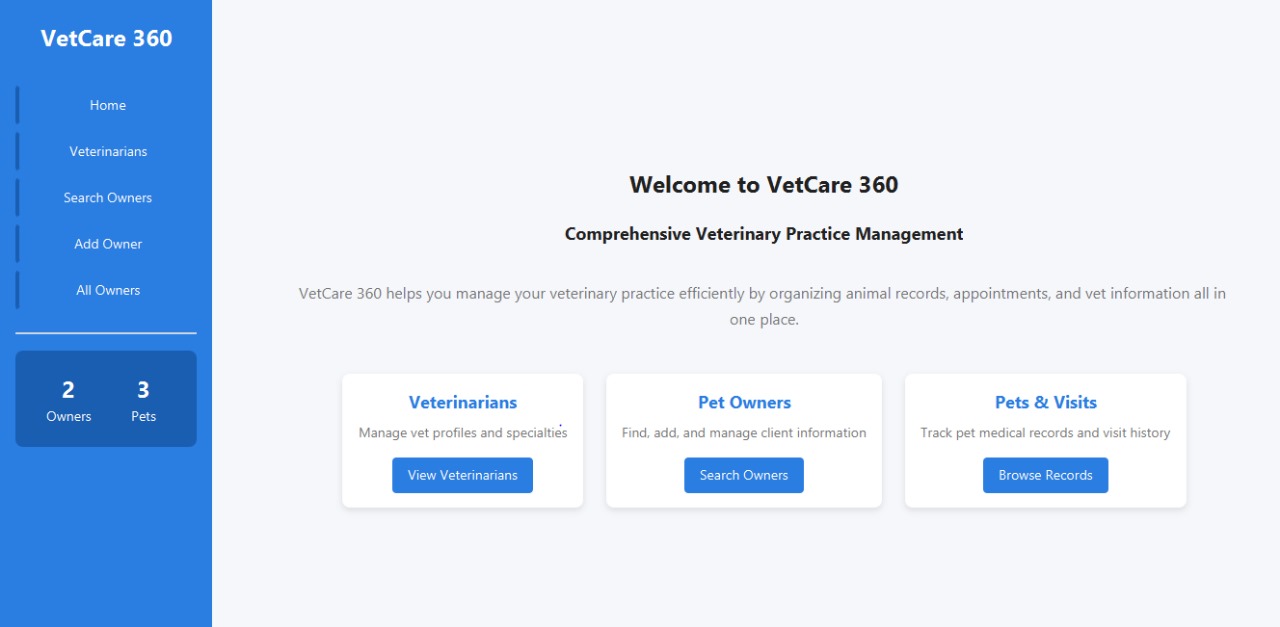
**VetCare 360 java oop**

**Filière : Ingénierie Logicielle et Cybersécurité (ILCS)**

**Sous la supervision du professeur : Redouane ESBAI**

**Nom : EL OUAHBI et AGHADDAD**

**prénom : FATIMA et SIHAM**



# VetCare 360

Une application de gestion complète pour les cliniques vétérinaires, développée en **Java 11** avec **JavaFX**,

## Table des Matières

1. -------------------------------------------Introduction
2. ------------------------------------------ Objectifs du Projet
3. ------------------------------------------ Technologies Utilisées
4. ------------------------------------------ Architecture du Projet
5. ------------------------------------------ Fonctionnalités Principales
6. ------------------------------------------ Structure du Code
7. -------------------------------------------Modèles de Données
8. ------------------------------------------ Interfaces Utilisateur
9. ----------------------------------------- Gestion des Données
10. ---------------------------------------- Tests et Validation
11. ------------------------------------------ Conclusion et Perspectives

## Introduction

Dans un monde de plus en plus digitalisé, où les technologies de l’information transforment profondément tous les secteurs d’activité, le domaine de la santé animale accuse encore un certain retard en matière de modernisation et de numérisation de ses processus. En effet, de nombreuses cliniques vétérinaires continuent de s’appuyer sur des méthodes traditionnelles, souvent manuelles, pour la gestion des dossiers médicaux, la planification des rendez-vous ou le suivi des consultations. Ces pratiques, bien qu'encore répandues, présentent plusieurs limites : elles sont chronophages, augmentent le risque d’erreurs humaines, et rendent difficile un suivi rigoureux et centralisé de l’historique médical des animaux.

Dans ce contexte, le projet **VetCare 360** vient répondre à un réel besoin de transformation digitale. Il s’agit d’une solution informatique moderne, conçue pour offrir aux cliniques vétérinaires une plateforme intuitive, efficace et sécurisée pour la gestion complète de leurs activités quotidiennes. VetCare 360 ambitionne d’automatiser et d’optimiser la gestion des consultations, des dossiers médicaux, des clients (propriétaires d’animaux), ainsi que des plannings, tout en assurant un accès rapide et fiable à l’information. Ce projet s’inscrit donc dans une dynamique d’innovation au service de la santé animale et du confort de travail des professionnels vétérinaires.

### Objectif général du projet

### 

### Le projet VetCare 360 vise à concevoir et développer une application de bureau complète, intuitive et ergonomique, spécifiquement adaptée aux besoins des cliniques vétérinaires. Ce logiciel ambitionne de dépasser les limites des méthodes traditionnelles de gestion en proposant une solution numérique efficace qui permet :

* La centralisation des données médicales et administratives : au lieu de s’appuyer sur des fichiers papier ou des tableaux éparpillés, l’application regroupe toutes les informations relatives aux animaux, aux propriétaires, aux consultations et aux traitements dans une base de données unique. Cela facilite la recherche, la mise à jour et l’analyse des informations.
* Un accès rapide et sécurisé à l’information : grâce à une interface claire et bien organisée, les vétérinaires et leurs assistants peuvent accéder aux dossiers médicaux en quelques secondes, tout en respectant les normes de sécurité et de confidentialité. Le système d’authentification basé sur les rôlesgarantit quechaque utilisateur accède uniquement aux données qui lui sont autorisées.
* Un suivi complet et précis de l’état de santé de chaque animal : l’application conserve l’historique détaillé de chaque visite, les examens réalisés, les traitements prescrits et les résultats obtenus. Cela permet au vétérinaire de suivre l’évolution de l’état de santé de l’animal dans le temps et de prendre des décisions éclairées.
* Une gestion optimisée des rendez-vous et du personnel : VetCare 360 permet de planifier les emplois du temps des vétérinaires, de suivre les horaires du personnel et de gérer efficacement les rendez-vous des clients, avec la possibilité d’envoyer des rappels ou de reprogrammer facilement en cas d’imprévus.

### Public cible

### VetCare 360 s’adresse à une large gamme de professionnels du domaine vétérinaire, notamment :

* Les cliniques vétérinaires de petite et moyenne taille : souvent limitées en ressources, elles ont besoin d’une solution simple, performante et adaptée à leur réalité.
* Les vétérinaires libéraux : travaillant seuls ou en petits groupes, ils recherchent une solution locale, facile à installer et à utiliser, sans infrastructure complexe.
* Les centres de soins animaliers et les refuges : qui prennent en charge un grand nombre d’animaux et nécessitent un suivi médical rigoureux et bien organisé pour chacun.

### Pourquoi ce projet ?

### 

### Le projet VetCare 360 est le fruit d’une réflexion approfondie menée dans le cadre d’une formation en développement informatique. Il ne s’agit pas simplement d’un exercice académique, mais d’un véritable défi technique visant à transformer les compétences acquises en une application concrète et utile. L’objectif était de :

* Mettre en œuvre une méthodologie de développement logiciel complète, en passant par toutes les étapes : analyse, conception, codage, tests et documentation.
* Utiliser le langage Java et la technologie JavaFX pour développer une interface graphique interactive et professionnelle.
* Créer une base de données structurée et performante, capable de gérer des données variées et évolutives.
* Adopter une architecture logicielle propre et évolutive (Clean Architecture), qui facilite la maintenance et l’ajout de nouvelles fonctionnalités à l’avenir.

### VetCare 360 est ainsi un exemple concret de la manière dont une idée peut évoluer pour devenir un outil numérique au service d’un secteur essentiel, en contribuant à la transformation digitale du domaine vétérinaire.

### 

### 

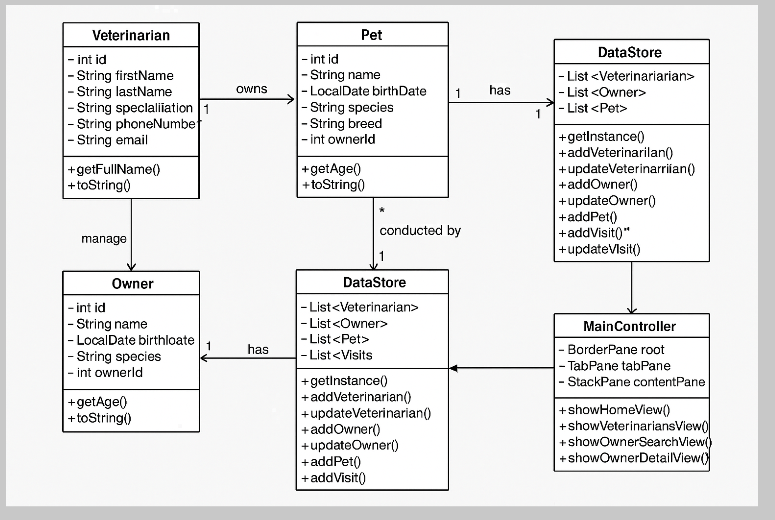
### Technologies utilisées

Dans le cadre du développement de l’application VetCare 360, plusieurs technologies ont été choisies de manière stratégique pour garantir performance, lisibilité du code, et facilité de maintenance.

* **Java 11** Le langage Java a été retenu comme base principale du projet en raison de sa robustesse, sa portabilité et sa richesse en bibliothèques standard. Java 11, version LTS (Long-Term Support), offre une stabilité et un support à long terme, ce qui est idéal pour un projet structuré et évolutif. De plus, Java est un langage orienté objet, ce qui facilite l’organisation du code et la modularité de l’application.
* **JavaFX** JavaFX a été utilisé pour concevoir l’interface graphique de l’application. Cette technologie permet de créer des interfaces utilisateur modernes, réactives et bien structurées. Contrairement à Swing (ancien framework Java), JavaFX offre plus de flexibilité dans la mise en page, l’intégration de styles CSS, et l’utilisation de composants graphiques personnalisés, ce qui améliore grandement l’expérience utilisateur.
* **ArrayList (stockage temporaire des données)** Pour le stockage temporaire des informations (avant intégration d’une base de données complète), des collections de type ArrayList ont été utilisées. Elles permettent de manipuler dynamiquement des listes d’objets (comme les animaux, les clients ou les rendez-vous) de manière simple et efficace. Ce choix a été motivé par la simplicité d’implémentation pendant les phases initiales de développement et de test.
* **Architecture MVC (Modèle-Vue-Contrôleur)**  L’application suit le modèle d’architecture **MVC**, qui sépare clairement la logique métier (Modèle), l’interface utilisateur (Vue) et le contrôle des interactions (Contrôleur).  
   Cette séparation favorise :  
    
  + Une meilleure organisation du code,
  + Une maintenance facilitée,
  + Et la possibilité de faire évoluer l’application (changer la vue sans toucher au modèle, par exemple).  
     Grâce à cette architecture, chaque partie du code a un rôle précis, ce qui rend l'application plus claire et modulaire.

**Architecture du Projet**

**Diagramme de Classes**



classDiagram

class Veterinarian {

-int id

-String firstName

-String lastName

-String specialization

-String phoneNumber

-String email

+getFullName()

+toString()

}

class Owner {

-int id

-String firstName

-String lastName

-String address

-String city

-String phoneNumber

-String email

+getFullName()

+getFullAddress()

+toString()

}

class Pet {

-int id

-String name

-LocalDate birthDate

-String species

-String breed

-int ownerId

+getAge()

+toString()

}

class Visit {

-int id

-int petId

-int veterinarianId

-LocalDate date

-LocalTime time

-String description

-String diagnosis

-String treatment

+toString()

}

class DataStore {

-List~Veterinarian~ veterinarians

-List~Owner~ owners

-List~Pet~ pets

-List~Visit~ visits

+getInstance()

+addVeterinarian()

+updateVeterinarian()

+addOwner()

+updateOwner()

+addPet()

+updatePet()

+addVisit()

+updateVisit()

}

class MainController {

-BorderPane root

-TabPane tabPane

-StackPane contentPane

+showHomeView()

+showVeterinariansView()

+showOwnerSearchView()

+showOwnerDetailView()

}

Owner "1" -- "\*" Pet : owns

Pet "1" -- "\*" Visit : has

Visit "\*" -- "1" Veterinarian : conducted by

DataStore -- Veterinarian : manages

DataStore -- Owner : manages

DataStore -- Pet : manages

DataStore -- Visit : manages

MainController -- DataStore : uses

**Structure des Packages**

com.vetcare360

├── controllers/ # Contrôleurs de l'application

├── models/ # Modèles de données

└── utils/ # Utilitaires et helpers

**Composants Principaux**

L'application *VetCare 360* repose sur une architecture claire et modulaire, structurée autour de trois grandes catégories de composants : **les contrôleurs (Controllers)**, **les modèles (Models)** et **les utilitaires (Utils)**. Chacune de ces catégories joue un rôle essentiel dans le bon fonctionnement et l’organisation de l’application.

### 1. Controllers – *Gestion de la logique métier et des interactions utilisateur*

Les contrôleurs sont responsables de la gestion des actions de l'utilisateur et de la navigation entre les différentes vues de l'application. Ils traitent les événements, mettent à jour l’interface et communiquent avec les modèles pour effectuer les opérations nécessaires.

* **MainController** : C’est le contrôleur principal qui orchestre la navigation entre les différentes sections de l’application. Il gère l'affichage de l'interface utilisateur globale (menus, onglets, vues dynamiques).
* **VeterinarianController** : Gère les opérations relatives aux vétérinaires, telles que l’ajout, la modification, la suppression et l’affichage des informations sur les vétérinaires.
* **OwnerController** : Responsable de la gestion des propriétaires d’animaux. Il permet d'ajouter de nouveaux propriétaires, de consulter leurs données et de les mettre à jour.
* **PetController** : Permet de gérer les animaux, en lien avec leurs propriétaires. On peut y consulter l’historique médical, ajouter de nouveaux animaux, ou modifier les informations existantes.
* **VisitController** : Gère les visites médicales, en reliant un vétérinaire, un animal et une date de consultation. Ce contrôleur s’occupe également du suivi des diagnostics et traitements prescrits.

### 2. Models – *Représentation des données de l’application*

Les modèles définissent la structure des entités manipulées dans l’application. Ils encapsulent les données et peuvent également contenir de la logique liée à ces données (ex. validation).

* **Veterinarian** : Contient les informations relatives aux vétérinaires, comme le nom, la spécialité et les disponibilités.
* **Owner** : Représente les propriétaires d’animaux, incluant leurs coordonnées et la liste de leurs animaux.
* **Pet** : Représente chaque animal suivi dans la clinique, avec ses caractéristiques (nom, espèce, race, âge) et son lien avec un propriétaire.
* **Visit** : Modèle des visites médicales. Il contient des détails comme la date, le vétérinaire responsable, l’animal concerné et les notes ou diagnostics associés.

### 3. Utils – *Fonctionnalités auxiliaires et support technique*

Les classes utilitaires offrent des services transversaux qui facilitent le développement, le test ou le stockage temporaire de données.

* **DataStore** : Joue le rôle de base de données temporaire en mémoire. Il permet de stocker et d’accéder aux entités (vétérinaires, propriétaires, animaux, visites) durant l’exécution de l’application, en l’absence d’un vrai système de base de données.
* **SampleDataGenerator** : Permet de générer automatiquement des données fictives pour faciliter les tests et la démonstration de l’application. Il est très utile en phase de développement ou pour des démonstrations sans avoir besoin d’une base réelle

**Fonctionnalités**

## Fonctionnalités Clés de l’Application

L’application *VetCare 360* propose une gestion complète et intuitive des différentes entités d’une clinique vétérinaire. Elle est organisée en quatre modules fonctionnels principaux, chacun dédié à un aspect spécifique du métier.

### 1. Gestion des Vétérinaires

Ce module permet une gestion efficace du personnel médical de la clinique. Il vise à centraliser les informations relatives aux vétérinaires et à organiser leur emploi du temps.

* **Ajout de nouveaux vétérinaires** : Possibilité d’enregistrer un nouveau vétérinaire avec ses coordonnées, sa spécialité et sa disponibilité.
* **Modification des informations** : Mise à jour des données personnelles ou professionnelles (nom, spécialité, horaires…).
* **Consultation des spécialisations** : Visualisation des compétences de chaque vétérinaire afin de faciliter l’attribution des cas spécifiques.
* **Attribution des rendez-vous** : Affectation d’un vétérinaire à une visite médicale, selon sa spécialité et sa disponibilité.

### 2. Gestion des Propriétaires

Ce module permet d’administrer les fiches clients (propriétaires d’animaux), de suivre leur historique, et d'assurer une relation personnalisée.

* **Recherche par nom** : Recherche rapide d’un propriétaire via son nom ou prénom.
* **Création de nouveaux profils** : Enregistrement d’un nouveau client avec ses coordonnées complètes.
* **Modification des informations** : Possibilité de mettre à jour les données du client (adresse, téléphone, etc.).
* **Association avec les animaux** : Chaque propriétaire peut être lié à un ou plusieurs animaux.
* **Historique des visites** : Accès à l’historique des consultations liées à un propriétaire et ses animaux.

### 3. Gestion des Animaux

Ce module assure un suivi complet et précis des animaux pris en charge par la clinique.

* **Création de dossiers médicaux** : Chaque animal dispose d’un dossier médical contenant ses informations générales et cliniques.
* **Suivi des traitements** : Suivi des prescriptions, traitements en cours ou antérieurs.
* **Historique des visites** : Affichage de toutes les consultations passées avec diagnostics et traitements.
* **Informations détaillées** : Description précise de l’animal : nom, espèce, race, sexe, âge, poids, etc.

### 4. Gestion des Visites

Ce module gère les rendez-vous médicaux, de leur planification jusqu’à l’archivage des diagnostics.

* **Planification des rendez-vous** : Enregistrement d’une nouvelle visite en choisissant la date, le vétérinaire, et l’animal concerné.
* **Enregistrement des diagnostics** : Ajout des observations cliniques, résultats d’examens, et diagnostics établis.
* **Suivi des traitements** : Lien avec les traitements prescrits à l’issue de la visite.
* **Historique médical complet** : Accès global à l’ensemble des visites pour un suivi longitudinal de la santé de l’animal.

## Structure du Code

L’architecture de *VetCare 360* est pensée selon une séparation claire des responsabilités, en suivant le modèle **MVC** (Modèle–Vue–Contrôleur). Cette structure facilite la maintenance, l’évolutivité et la lisibilité du code.

### Controllers – *Logique métier et gestion des interactions*

Les contrôleurs sont responsables de l’orchestration des actions entre l’utilisateur et les données. Ils traitent les événements issus de l’interface, effectuent des vérifications et déclenchent les opérations correspondantes sur les modèles.

#### MainController

* Gère la **navigation principale** de l’application entre les différentes vues.
* Assure la **gestion des transitions d’interface** et la mise en place des composants graphiques.
* Coordonne les autres contrôleurs pour garantir une synchronisation fluide entre les modules.

#### VeterinarianController

* Centralise la **gestion complète des vétérinaires** (ajout, modification, suppression, affichage).
* Met en place une **interface CRUD** (Create, Read, Update, Delete) conviviale et efficace.
* Effectue la **validation des champs** saisis (nom, spécialité, téléphone…).

#### OwnerController

* Permet la **création et la gestion des propriétaires d’animaux**.
* Intègre des fonctionnalités de **recherche et de filtrage** par nom ou critères spécifiques.
* S’assure de la **cohérence et la validation des informations** saisies.

### Models – *Représentation des données métier*

Les modèles encapsulent les entités principales manipulées dans l’application. Ils décrivent la structure des données et servent d’intermédiaires entre les contrôleurs et le stockage.

#### Veterinarian

* Contient un **identifiant unique** pour chaque vétérinaire.
* Stocke les **informations personnelles** (nom, prénom).
* Gère les données liées à la **spécialisation** et aux **coordonnées de contact**.

#### Owner

* Dispose d’un **ID unique** pour chaque propriétaire.
* Regroupe les **données personnelles** (adresse, téléphone, etc.).
* Gère une **liste d’animaux** associés.
* Conserve un **historique des interactions** et des visites passées.

#### Pet

* Possède un **identifiant unique** par animal.
* Regroupe les **informations de base** (nom, espèce, race, âge, sexe).
* Fait le lien avec son **propriétaire associé**.
* Contient un **dossier médical** avec un historique des visites, traitements et diagnostics.

#### Visit

* Identifié par un **ID unique**.
* Contient les **informations de planification** : date, heure, vétérinaire, animal concerné.
* Permet la saisie des **diagnostics, observations médicales et traitements prescrits**.

## Interfaces Utilisateur

L’interface graphique de *VetCare 360* a été conçue avec une attention particulière à l’**ergonomie**, à la **simplicité d’utilisation** et à la **réactivité**. Elle vise à offrir une expérience fluide tant pour les vétérinaires que pour le personnel administratif.

### Interface Principale

L’interface principale est le cœur de la navigation dans l’application. Elle est conçue pour offrir un accès rapide et clair à toutes les fonctionnalités essentielles.

* **Navigation intuitive** : Une structure claire permet aux utilisateurs de trouver rapidement ce dont ils ont besoin, même sans formation préalable.
* **Menu latéral** : Accessible à tout moment, il donne un accès direct aux différents modules : vétérinaires, propriétaires, animaux, visites, etc.
* **Zone de contenu principale** : C’est dans cette zone que s’affichent dynamiquement les vues ou formulaires en fonction des actions de l’utilisateur.
* **Barre d’outils contextuelle** : Elle s’adapte en fonction de la vue affichée (ajouter, filtrer, imprimer, rafraîchir, etc.), pour plus d’efficacité.

### Formulaires

Les formulaires permettent la saisie et la modification des données tout en garantissant leur cohérence grâce à des mécanismes de validation intégrés.

* **Validation en temps réel** : Les champs sont contrôlés dès la saisie, pour éviter les erreurs avant la soumission.
* **Messages d'erreur explicites** : Les utilisateurs sont guidés par des messages clairs et compréhensibles en cas d’erreur de saisie.
* **Auto-complétion** : Dans certains champs (par exemple, recherche de vétérinaire ou de propriétaire), une suggestion automatique s'affiche pour accélérer la saisie.
* **Sauvegarde automatique** : Dans les formulaires longs ou critiques, les données sont automatiquement enregistrées périodiquement pour éviter toute perte accidentelle.

### Tableaux de Données

Les tableaux affichent les données sous forme structurée, avec des outils interactifs pour faciliter leur consultation et gestion.

* **Tri par colonnes** : L’utilisateur peut trier les données (ex. par date, nom, spécialité) en un seul clic.
* **Filtrage** : Possibilité d’appliquer des filtres pour n’afficher que les éléments pertinents selon certains critères (ex. visites du jour, propriétaires par ville).
* **Pagination** : Les longues listes sont divisées en pages pour une navigation fluide, avec options de saut rapide.
* **Actions contextuelles** : Chaque ligne propose des actions rapides comme "Modifier", "Supprimer", "Voir détails", accessibles via un menu ou un bouton.

## Gestion des Données

La gestion des données au sein de *VetCare 360* repose sur une architecture légère mais structurée, combinant un système de **stockage temporaire en mémoire** et des **mécanismes de persistance** simulés pour les phases de développement et de test. L’objectif est d’assurer un flux de données fluide entre les entités et les vues de l’application.

### DataStore – *Stockage en mémoire*

Le **DataStore** agit comme une base de données temporaire intégrée à l’application. Il facilite l’accès rapide aux données sans nécessiter de système de gestion de base de données externe (pendant la phase de développement ou démonstration).

* **Stockage temporaire en mémoire** : Toutes les données sont conservées dans des structures internes pendant l'exécution, sans être enregistrées de manière permanente.
* **Collections ArrayList** : Les entités (vétérinaires, propriétaires, animaux, visites) sont stockées dans des listes dynamiques pour permettre un accès rapide, des recherches, et des modifications aisées.
* **Gestion des relations entre entités** : Le DataStore gère les liens logiques entre les entités (ex : un propriétaire et ses animaux, une visite liée à un vétérinaire et un animal).
* **Opérations CRUD** : Le module permet d’effectuer toutes les opérations de base (Créer, Lire, Mettre à jour, Supprimer) sur les données.

### Persistence – *Simulation de sauvegarde et de cohérence*

Même si l’application ne dispose pas encore d’un système de base de données complet, des mécanismes de persistance simulée ont été mis en place pour offrir une expérience réaliste et garantir la cohérence des données.

* **Stockage temporaire** : Les données peuvent être simulées comme persistantes pendant une session, via sérialisation ou maintien en mémoire.
* **Données de test** : Grâce à un générateur (SampleDataGenerator), des jeux de données fictives sont injectés pour tester les interfaces et les fonctionnalités sans saisie manuelle.
* **Gestion de la cohérence** : Lors des manipulations (ajouts, modifications), des vérifications sont effectuées pour garantir la validité des relations (par exemple, un animal ne peut pas exister sans propriétaire).

## Tests et Validation

Afin de garantir la **qualité** et la **stabilité** de l'application, une série de tests rigoureux a été mise en place sur l’ensemble des composants du système, que ce soit au niveau de la logique métier ou de l’interface utilisateur.

### Tests Unitaires

Des tests indépendants ont été effectués sur chaque composant du système pour vérifier la validité de la logique applicative et l'exactitude des interactions entre les entités.

* **Validation des modèles** : Vérification que les entités (animal, vétérinaire, visite, etc.) sont correctement stockées et traitées.
* **Tests des contrôleurs** : Assurance que chaque contrôleur gère efficacement les données et la navigation.
* **Vérification des règles métier** : Contrôle du respect des règles logiques (ex. : impossibilité de planifier une visite pour un animal inexistant).

### Tests d’Interface

Ces tests visent à s'assurer que l'utilisateur peut interagir avec l'application de manière fluide et sans erreurs.

* **Validation des formulaires** : Test de la saisie, de l’envoi et des messages d’erreur en cas de données incorrectes.
* **Tests de navigation** : Vérification de la cohérence et de l’accessibilité de toutes les vues.
* **Vérification des interactions** : Test du bon fonctionnement des boutons, menus, liens, et autres éléments interactifs.

### Données de Test

Des données fictives variées ont été utilisées pour simuler des scénarios réels et tester le comportement de l’application dans différents cas.

* **Génération automatique** : Grâce au module SampleDataGenerator, des données sont créées dynamiquement pour les tests.
* **Scénarios réalistes** : Simulation de l’utilisation quotidienne dans une clinique vétérinaire.
* **Validation des cas limites** : Test du système face à des données manquantes ou incohérentes afin d’en évaluer la robustesse.

### Points Forts

* Interface intuitive et facile à prendre en main
* Architecture modulaire et évolutive
* Gestion complète des entités (vétérinaires, propriétaires, animaux, visites)
* Flexibilité et extensibilité du système pour de futures améliorations

### Perspectives d’Évolution

* Intégration d’un système de **persistance réelle** des données (base de données)
* Ajout de **rapports statistiques** pour le suivi de l’activité
* Développement d’un **module de facturation**
* Création d’une **application mobile complémentaire** pour les vétérinaires et assistants

## Conclusion

L’application **VetCare 360** représente une avancée significative vers la **digitalisation et la modernisation de la gestion des cliniques vétérinaires**. Elle propose un système complet qui allie **simplicité d’utilisation** et **efficacité opérationnelle**. Conçue pour répondre aux besoins des vétérinaires et des gestionnaires, elle facilite l’organisation des tâches quotidiennes, depuis l’accueil des clients jusqu’au suivi médical des animaux.

Grâce à une **architecture bien structurée** basée sur la séparation des composants (modèles, contrôleurs, interfaces), l’équipe a pu garantir une **lisibilité du code**, une **maintenance aisée** et une **facilité d’évolution** à long terme. L’utilisation de **tests rigoureux** et de **données de simulation réalistes** a également permis de s'assurer de la **stabilité** et du **bon fonctionnement** de l’ensemble du système.

Bien que la version actuelle repose sur un stockage temporaire et des données simulées, elle ouvre la voie à une **intégration plus poussée avec des bases de données réelles**, ainsi qu’à l’ajout de **fonctionnalités avancées** telles que les rapports financiers, la facturation électronique, ou encore des applications mobiles.

En somme, **VetCare 360** est bien plus qu’une simple application : c’est une **vision d’avenir** pour une gestion vétérinaire intelligente, précise et évolutive, posant les bases d’un système d’information moderne, adaptable et durable.