المدرسة العليا للتكنولوجيا الناظور
École Supérieure de Technologie de Nador
ⵜⵉⵎⵙⵙⵉⵍⵜ ⵜⴰⵏⴰⵎⵓⵔⵜ ⵏ ⵜⵓⵜⵜⵓⵔⵉⵙⵜ ⵅⵓⵏⴰⵜⵓⵔ

Rapport sur l'application web

VetCare360

Java

Réalisé par : **LETRACH IBTIHAL**

OUKHSSANE IHSSANE

Encadrant : **Professeur M.ESBAI REDOUANE**

Année universitaire: **2024 - 2025**

Etablissement : **Ecole supérieure de technologie Nador**

Filière : **Ingénierie de logiciels et cybersécurité**

Résumé :

Le projet **VetCare 360** est une application de bureau développée en **JavaFX** avec une base de données **MySQL**, destinée à faciliter la gestion quotidienne d'une clinique vétérinaire. Face aux besoins croissants en matière d'organisation, de suivi médical et de gestion numérique des informations, ce projet vise à fournir une solution logicielle efficace, intuitive et adaptée aux professionnels de la santé animale.

L'application permet la gestion complète des entités suivantes : **vétérinaires**, **propriétaires d'animaux**, **animaux**, et **visites médicales**. Chaque module de l'application propose des fonctionnalités essentielles telles que l'ajout, la modification, la suppression et la recherche dynamique d'enregistrements.

Sur le plan technique, VetCare 360 repose sur une architecture claire combinant **JavaFX pour l'interface graphique**, **FXML pour la séparation de la vue**, **CSS pour le design**, et **MySQL** pour le stockage persistant des données. Le projet suit une démarche modulaire et structurée, facilitant la maintenabilité et l'extensibilité.

Ce rapport retrace l'ensemble du processus de développement, depuis l'analyse des besoins jusqu'à la réalisation concrète, en passant par la conception de la base de données, le choix des technologies, et la mise en œuvre des interfaces utilisateurs. Il met également en lumière les choix ergonomiques effectués afin d'assurer une expérience utilisateur fluide et agréable.

VetCare 360 ambitionne ainsi de répondre aux besoins des cliniques vétérinaires modernes en proposant un outil simple, performant et personnalisable.

Table des matières

Introduction :

Dans un contexte où la digitalisation touche tous les secteurs, le domaine vétérinaire ne fait pas exception. La gestion d'une clinique vétérinaire implique un suivi rigoureux de nombreuses données : fiches des animaux, historique des visites médicales, informations des propriétaires, agenda des vétérinaires, etc. Or, de nombreuses structures fonctionnent encore avec des méthodes manuelles ou des outils non spécialisés, rendant la gestion chronophage, sujette à l'erreur, et difficile à maintenir à jour.

C'est dans cette optique qu'est né **VetCare 360**, une application de bureau moderne développée en **JavaFX** avec **MySQL** pour répondre aux besoins spécifiques des cliniques vétérinaires. Elle vise à centraliser les informations, automatiser les tâches courantes, et offrir un environnement de travail fluide aux vétérinaires et à leur personnel.

VetCare 360 se veut simple à utiliser, ergonomique et visuellement attrayant. Elle permet notamment :

- La gestion des **vétérinaires** (coordonnées, spécialités),
- La gestion des **propriétaires** et de leurs **animaux** (fiche complète),
- Le suivi des **visites médicales** (date, motif, vétérinaire responsable),
- Une **interface intuitive** avec recherche dynamique, tableaux filtrables et boutons clairs.

Ce rapport retrace l'ensemble du développement de l'application, depuis la **définition du besoin** jusqu'à la **mise en œuvre technique**, en passant par le **choix des outils**, la **conception de la base de données**, l'**interface utilisateur**, et les **fonctionnalités implémentées.** Il permet de mettre en lumière le travail réalisé, les solutions apportées, ainsi que les possibilités d'évolution du projet.

Objectifs du projet :

Le développement de **VetCare 360** s'inscrit dans une volonté de répondre à des besoins précis rencontrés dans les structures vétérinaires. Ce projet a été initié avec une vision claire : offrir un outil efficace, intuitif et professionnel permettant d'améliorer la gestion quotidienne d'une clinique vétérinaire.

Objectif général

Créer une **application de gestion complète** pour une clinique vétérinaire, permettant de centraliser, consulter, ajouter, modifier et supprimer les informations relatives aux vétérinaires, aux propriétaires, aux animaux, et aux visites médicales, dans une interface moderne et responsive.

Objectifs spécifiques

1. **Digitalisation des processus** :
   - Supprimer la gestion manuelle des dossiers.
   - Permettre un accès rapide et structuré aux informations.
2. **Gestion multi-entités** :
   - Implémenter des modules pour les vétérinaires, les propriétaires, les animaux et les visites.
   - Lier les entités entre elles de manière cohérente (ex. : un animal est rattaché à un propriétaire, une visite à un vétérinaire).

3. **Interface conviviale** :
    - Créer une interface utilisateur claire avec **JavaFX**.
    - Utiliser une **navigation fluide** (navbar, pages dédiées, effets visuels).
    - Appliquer un **design homogène** à toutes les pages (thème de couleurs, tableau responsive, boutons stylisés).
4. **Fonctionnalités interactives** :
    - Mise en place de la **recherche dynamique** (filtrage en temps réel).
    - Ajout / modification / suppression en quelques clics avec retour visuel.
5. **Persistance des données** :
    - Utilisation de **MySQL** pour stocker et gérer les données de manière fiable.
    - Connexion sécurisée à la base de données via JDBC.
6. **Préparation à l'évolution** :
    - Organiser le code selon une **structure MVC** (Modèle - Vue - Contrôleur) claire.

Technologies utilisés :

Le choix des technologies a été guidé par les objectifs de performance, de maintenabilité et de professionnalisme. Le projet **VetCare 360** repose sur un ensemble cohérent de technologies modernes, bien établies dans le domaine du développement d'applications de bureau.

Java :

Le langage principal utilisé est **Java**, pour sa robustesse, sa portabilité et sa large adoption dans le monde professionnel.

- **Version utilisée** : Java 21
- **Avantages** :
  - Orienté objet
  - Très bon support pour les applications de gestion
  - Compatibilité avec de nombreux frameworks (JavaFX, JDBC, etc.)

JavaFX :

JavaFX est utilisé pour la création de l'interface graphique.

- **Modules utilisés** :
  - javafx.controls
  - javafx.fxml
- **Atouts** :
  - Composants graphiques modernes
  - Support du CSS pour le style
  - FXML pour séparer la vue du contrôleur (MVC)

MySQL :

Pour la base de données relationnelle, **MySQL** a été choisi pour sa simplicité, sa stabilité et sa performance.

- **Connecteur JDBC utilisé** : mysql-connector-j-9.3.0.jar
- **Structure** :
  - Tables bien normalisées
  - Relations entre propriétaires, animaux, vétérinaires et visites

JDBC (Java Database Connectivity) :

Le connecteur **JDBC** est utilisé pour la communication entre l'application Java et la base de données MySQL.

- Gestion des requêtes SQL : SELECT, INSERT, UPDATE, DELETE
- Connexion centralisée dans une classe DatabaseConnection

L'application est structurée selon les conventions MVC :

Structure des fichiers :

- model/ → classes métiers (Veterinaire, Animal, etc.)
- dao/ → classes d'accès aux données
- controller/ → logique métier et événements liés à l'UI
- resources/fxml/ → vues définies en FXML
- resources/style/ → fichiers CSS
- resources/images/ → images de l'application (logo, icônes)

CSS :

L'interface utilisateur est personnalisée à l'aide d'un fichier style.css :

- Couleurs : tons clairs (#ecf0f1) et sombres (#2c3e50)
- Boutons colorés (ajouter = vert, modifier = jaune, supprimer = rouge)

Structure de l’application :

L’architecture du projet **VetCare 360** suit le modèle **MVC (Modèle-Vue-Contrôleur)**, ce qui permet une séparation claire des responsabilités, facilite la maintenance et l’évolution de l’application.

1. Modèle (Model)

Les classes du **modèle** représentent les entités manipulées par l'application. Elles contiennent uniquement des attributs, des constructeurs et des accesseurs (getters/setters).

Exemples de classes du modèle :

- Veterinaire.java
- Proprietaire.java
- Animal.java
- Visite.java

Chaque classe correspond à une table de la base de données MySQL.

2. Accès aux données (DAO)

Les classes **DAO** (Data Access Object) contiennent toutes les opérations SQL (CRUD) et sont responsables de l’interaction avec la base de données.

Exemples :

- VeterinaireDAO.getAll()
- AnimalDAO.insert(animal)
- VisiteDAO.delete(id)

Ces classes permettent d'abstraire complètement la base de données du reste de l'application.

3. Contrôleur (Controller)

Les classes **contrôleurs** sont liées aux vues (FXML) et gèrent les actions utilisateur (ajouter, modifier, supprimer, rechercher...).

Exemples :

- VeterinaireController.java
- AnimalController.java

Chaque contrôleur :

- Initialise les composants de la vue
- Récupère et affiche les données via DAO
- Met à jour le modèle en réponse aux actions utilisateur

4. Vue (FXML + CSS)

Les fichiers **FXML** décrivent l'interface graphique. Ils sont associés à des contrôleurs Java.

Vues principales :

- main_menu.fxml : menu de navigation
- accueil.fxml : page d'accueil
- ListeVeterinaires.fxml : gestion des vétérinaires
- proprietaire_list.fxml, animal.fxml, visite.fxml

Le style est géré dans un fichier unique : style.css.

5. Navigation

La navigation entre les vues est gérée dans MainMenuController.java grâce à un BorderPane qui affiche dynamiquement les différentes pages.

Structure de la base de données :

L'application **VetCare 360** repose sur une base de données **MySQL** relationnelle, composée de plusieurs tables représentant les entités principales du domaine vétérinaire.

Tables principales

1. **veterinaire**
    - id *(INT, PK, AUTO_INCREMENT)* : identifiant unique du vétérinaire
    - nom *(VARCHAR)* : nom du vétérinaire
    - prenom *(VARCHAR)* : prénom
    - specialite *(VARCHAR)* : spécialité médicale
    - telephone *(VARCHAR)* : numéro de téléphone
    - email *(VARCHAR)* : adresse mail
2. **proprietaire**
    - id *(INT, PK, AUTO_INCREMENT)* : identifiant du propriétaire
    - nom *(VARCHAR)* : nom
    - prenom *(VARCHAR)* : prénom
    - adresse *(VARCHAR)* : adresse complète
    - telephone *(VARCHAR)* : numéro de téléphone
    - email *(VARCHAR)* : adresse email

3. **animal**
    - id *(INT, PK, AUTO_INCREMENT)* : identifiant de l’animal
    - nom *(VARCHAR)* : nom de l’animal
    - espece *(VARCHAR)* : espèce (chien, chat, etc.)
    - race *(VARCHAR)* : race de l’animal
    - sexe *(VARCHAR)* : sexe (M/F)
    - date_naissance *(DATE)* : date de naissance
    - id_proprietaire *(INT, FK)* : clé étrangère vers proprietaire(id)
4. **visite**
    - id *(INT, PK, AUTO_INCREMENT)* : identifiant de la visite
    - date *(DATE)* : date de la consultation
    - motif *(VARCHAR)* : raison de la visite
    - id_animal *(INT, FK)* : clé étrangère vers animal(id)
    - id_veterinaire *(INT, FK)* : clé étrangère vers veterinaire(id)

Relations entre les tables

- Un **propriétaire** peut posséder plusieurs **animaux** *(relation 1-n)*.
- Un **animal** peut avoir plusieurs **visites** *(relation 1-n)*.
- Une **visite** est effectuée par un seul **vétérinaire** *(relation n-1)*.

Architecture Technique de l'application :

L'application **VetCare 360** repose sur une architecture **JavaFX** orientée objet et bien structurée, avec une séparation claire des responsabilités. Voici les principales couches de l'architecture :

1. Interface Utilisateur (JavaFX + FXML)

- Utilisation de fichiers .fxml pour la définition des interfaces graphiques.
- Le style est géré via un fichier CSS personnalisé (style.css) appliquant une charte graphique cohérente.
- Chaque vue possède un **contrôleur dédié** chargé de gérer les événements utilisateur et d'interagir avec les données.

Exemples de vues :

- main_menu.fxml : page d'accueil et navigation
- ListeVeterinaires.fxml : gestion des vétérinaires
- animal.fxml, visite.fxml, proprietaire_list.fxml : autres entités

2. Contrôleurs JavaFX

Les classes de contrôleurs comme VeterinaireController.java, AnimalController.java, etc. assurent :

- La gestion des événements (clics sur les boutons)
- La liaison entre les champs de saisie et les entités Java
- L'appel aux fonctions de la couche DAO

3. Modèle de Données (Modèle MVC)

Chaque entité de la base de données possède une classe Java correspondante :

- Veterinaire.java, Animal.java, Proprietaire.java, Visite.java
- Ces classes contiennent des attributs, des constructeurs et des getters/setters.

4. Accès aux Données (DAO)

Les classes DAO (Data Access Object) encapsulent toutes les opérations avec la base de données :

- Connexion à MySQL via JDBC (gérée par DatabaseConnection.java)
- Requêtes SQL pour : insertion, sélection, mise à jour, suppression

Exemples :

VeterinaireDAO.getAll();

VeterinaireDAO.insert(Veterinaire v);

5. Connexion à la base (JDBC)

- Configuration de la connexion via DatabaseConnection.java
- Utilisation du driver mysql-connector-j-9.3.0.jar

Organisation du Projet (Structure)

vetcare360/

```
├── src/
│   ├── controller/    → Contrôleurs JavaFX
│   ├── dao/           → Accès aux données
│   ├── model/         → Entités Java
│   ├── utils/         → Connexion MySQL
│   └── Main.java      → Point d'entrée
├── resources/
│   ├── fxml/          → Interfaces FXML
│   ├── style/         → Fichier CSS
│   └── images/        → Logo et icônes
└── .gitignore
```

Description Fonctionnelle :

L'application **VetCare 360** a pour but de faciliter la gestion quotidienne d'une clinique vétérinaire en permettant aux utilisateurs d'enregistrer, consulter et modifier les données relatives aux vétérinaires, propriétaires d'animaux, animaux eux-mêmes, et visites médicales. Elle propose une interface graphique intuitive et bien structurée autour de plusieurs modules.

1. Tableau de bord (Accueil)

- Présente une page d'accueil conviviale mettant en avant l'identité visuelle de la clinique.
- Affiche un logo, une image illustrative, un message de bienvenue ou une phrase d'accroche.
- Sert de point d'entrée vers les autres modules de l'application via une barre de navigation.

2. Module Vétérinaire

- Liste tous les vétérinaires enregistrés.
- Permet de :
    - **Ajouter** un vétérinaire avec son nom, prénom, spécialité, téléphone et email.
    - **Modifier** les informations d'un vétérinaire existant.
    - **Supprimer** un vétérinaire.
    - **Rechercher** un vétérinaire en filtrant dynamiquement par nom, prénom, spécialité, etc.

3. Module Propriétaires

- Permet la gestion des propriétaires d'animaux :
    - **Ajout, modification, suppression** et **recherche** dynamique.
- Chaque propriétaire est défini par : nom, prénom, adresse, téléphone, email.
- La recherche se fait en temps réel dans le tableau.

4. Module Animaux

- Gestion des animaux associés à des propriétaires.
- Informations enregistrées :
    - Nom
    - Espèce
    - Race
    - Sexe
    - Date de naissance
    - Propriétaire (via un menu déroulant)
- Possibilité de :
    - **Ajouter, modifier, supprimer** un animal
    - **Rechercher** un animal selon différents critères

5. Module Visites Médicales

- Enregistre les visites réalisées par les vétérinaires pour les animaux.
- Informations :
  - Date de la visite
  - Description ou diagnostic
  - Animal concerné
  - Vétérinaire ayant effectué la visite
- Actions disponibles :
  - **Ajout, modification, suppression**
  - **Filtrage** via un champ de recherche
- Ce module permet un suivi médical rigoureux des animaux.

Interface utilisateur et design :

L'interface de **VetCare 360** a été conçue pour offrir une expérience utilisateur claire, agréable et intuitive. Un soin particulier a été apporté à l'organisation visuelle, à la cohérence des couleurs et à la facilité de navigation.

1. Ligne Graphique

- **Palette de couleurs :**
  - Fond principal : #ecf0f1 (gris très clair)
  - Titres et éléments importants : #2c3e50 (bleu foncé)
  - Boutons :
    - Ajouter : vert (#27ae60)
    - Modifier : jaune (#f1c40f)
    - Supprimer : rouge (#e74c3c)
- **Police :**
  - Utilisation d'une typographie lisible et moderne.
  - Titres en gras, de grande taille.
- **Boutons et champs de texte :**
  - Coins arrondis, survol coloré pour un meilleur retour visuel.
  - Alignement centré, espacement cohérent.

2. Navigation

- Une **barre de navigation supérieure** est présente sur toutes les pages.
  - Elle permet un accès rapide aux sections :
    - Accueil
    - Vétérinaires
    - Propriétaires
    - Animaux
    - Visites

3. Tableaux

- Tous les modules principaux utilisent des **TableView JavaFX** :
  - Colonnes **centrées et égales**.
  - En-têtes de colonnes **blanches sur fond bleu foncé** pour rappeler la navbar.
  - Alternance de lignes (zébrage visuel) pour améliorer la lisibilité.
  - Résize automatique des colonnes (resize policy activée).

Conclusion :

Le développement de l'application **VetCare 360** marque l'aboutissement d'un projet ambitieux, à la fois formateur et porteur de solutions concrètes. Conçue dans le cadre d'un besoin réel de digitalisation dans les cabinets vétérinaires, cette application vise à améliorer la gestion quotidienne des structures de soins animaliers, en centralisant et simplifiant toutes les opérations liées aux **vétérinaires**, **propriétaires**, **animaux** et **visites médicales**.

Au fil du projet, l'accent a été mis sur l'**expérience utilisateur**, la **fiabilité des traitements**, ainsi que la **lisibilité de l'interface**. Grâce à **JavaFX**, l'application offre une interface moderne, claire et interactive, tandis que **MySQL** assure une persistance efficace et structurée des données. L'architecture mise en place, respectant les principes **MVC** et **DAO**, a permis de garantir une organisation modulaire, facilitant la maintenance, l'évolutivité et la réutilisabilité du code.

D'un point de vue pédagogique, ce projet a représenté une expérience complète, mobilisant un large éventail de compétences :

- Conception de base de données relationnelle.
- Développement Java orienté objet.
- Intégration avec JavaFX pour la création d'interfaces graphiques conviviales.
- Utilisation de FXML pour séparer clairement logique et présentation.
- Application de styles CSS pour personnaliser l'interface.
- Mise en œuvre de filtres dynamiques, de gestion des événements et de mécanismes CRUD performants.

En parallèle de l'aspect technique, VetCare 360 illustre également la capacité à mener un projet logiciel de bout en bout : **analyse des besoins**, **conception**,

implémentation, **test**, et **documentation**. C'est aussi un exemple concret de la manière dont des outils de développement classiques (comme Git et GitHub) peuvent être intégrés à une démarche professionnelle.

En somme, VetCare 360 ne constitue pas une simple application d'exercice, mais une véritable **base de travail évolutive** pour un projet informatique professionnel dans le domaine vétérinaire.

Ce projet a renforcé notre rigueur, notre capacité à **structurer un code lisible et fonctionnel**, ainsi que notre confiance dans la création de **logiciels orientés utilisateur**. Il nous a également sensibilisés à l'importance du **design**, de la **maintenabilité du code** et de l'**expérience utilisateur**. Il représente ainsi un **jalon important dans notre parcours en développement logiciel**, et constitue un **tremplin vers des projets encore plus ambitieux**.

Bibliographie :

1. **Oracle Java Documentation**
   https://docs.oracle.com/javase/
   Documentation officielle de Java SE.

2. **JavaFX Documentation**
   https://openjfx.io/
   Référence complète sur l'utilisation de JavaFX pour les interfaces graphiques.

3. **Logo et icônes utilisés dans l'application :**

- **Logo personnalisé VetCare** :

  Généré avec l'aide de l'outil Sora d'OpenAI (génération d'image par intelligence artificielle).

- **Icône chien (dog.png) et utilisateur (user.png)** :
  Icônes provenant de https://www.flaticon.com, utilisés sous licence gratuite avec attribution :
  - Dog icon by Freepik - https://www.flaticon.com/free-icon/dog_616408
  - User icon by Smashicons - https://www.flaticon.com/free-icon/user_149071

Remerciements:

Nous tenons à exprimer notre profonde gratitude à **Monsieur le Professeur ESBAI REDOUANE** pour son accompagnement attentif, sa disponibilité constante et ses conseils éclairés tout au long de ce projet.

Ses remarques constructives, sa pédagogie rigoureuse et son soutien moral nous ont grandement aidés à structurer notre travail, à approfondir nos compétences techniques et à mener à bien ce projet dans les meilleures conditions.

Grâce à son encadrement, nous avons pu vivre une expérience enrichissante, formatrice et stimulante. Nous lui adressons nos sincères remerciements pour la confiance qu'il nous a accordée.