



**ECOLE MAROCAINE DES
SCIENCES DE L'INGENIEUR**
Membre de
HONORIS UNITED UNIVERSITIES

EMSI - ÉCOLE MAROCAINE DES SCIENCES DE L'INGÉNIEUR

CASABLANCA

Rapport

de Projet Java Avancé

SmartCity

*SYSTÈME HYBRIDE DE GESTION DES INCIDENTS URBAINS
(WEB & DESKTOP)*

RÉALISÉ PAR :

MARWA EL OMARI ALAOUI

ENCADRÉ PAR :

PR. ABDERRAHIM LARHLIMI

ANNÉE UNIVERSITAIRE : 2025-2026

Remerciements

Au terme de ce travail de fin d'étude, je tiens à exprimer ma profonde gratitude et ma reconnaissance la plus distinguée à **Monsieur le Professeur Abderrahim Larhlimi**.

Je tiens à le remercier tout particulièrement pour l'honneur qu'il m'a fait en acceptant d'encadrer ce projet avec autant de bienveillance. Ses conseils avisés, sa rigueur scientifique et son expertise technique ont été des atouts majeurs et indispensables à l'aboutissement de ce travail. Au-delà de ses compétences académiques, sa vision éclairée et son sens du détail m'ont permis de surmonter les obstacles techniques et de parfaire mes connaissances en ingénierie logicielle. Je lui suis extrêmement reconnaissant pour le temps précieux qu'il m'a accordé malgré ses nombreuses responsabilités, ainsi que pour la confiance qu'il a placée en mes capacités.

Mes remerciements s'adressent également avec une grande estime à l'ensemble du corps professoral de l'**École Marocaine des Sciences de l'Ingénieur (EMSI)**. Je les remercie pour la qualité de l'enseignement qu'ils nous ont prodigué tout au long de mon cursus et pour leur dévouement constant à faire de nous des ingénieurs compétents et responsables, prêts à affronter les défis du monde professionnel.

Je ne saurais passer sous silence le soutien inconditionnel de ma famille. Un immense merci à mes parents, qui ont toujours cru en moi et qui ont consenti d'énormes sacrifices pour que je puisse poursuivre mes études dans les meilleures conditions. Leurs encouragements quotidiens et leur présence à mes côtés ont été ma plus grande source de motivation et de persévérance.

Enfin, je tiens à exprimer ma gratitude envers mes amis et collègues de promotion, avec qui j'ai partagé des moments d'entraide et d'apprentissage inoubliables. Que tous ceux qui ont contribué, de près ou de loin, à la réalisation de ce projet de fin d'étude trouvent ici l'expression de ma sincère reconnaissance.

Table des matières

1	Introduction Générale	3
1.1	Contexte du Projet	3
1.2	Problématique.....	3
1.3	Objectifs du Projet	4
2	Analyse et Conception	5
2.1	Spécification des Besoins	5
2.1.1	Besoins Fonctionnels.....	5
2.1.2	Besoins Non-Fonctionnels	5
2.2	Conception UML.....	6
2.2.1	Diagramme de Classes	6
3	Environnement Technique	7
3.1	Langage de Programmation	7
3.2	Environnement de Développement.....	7
3.3	Gestion de Projet avec Maven.....	7
3.4	Base de Données	8
3.5	Frameworks et Outils	8
4	Architecture et Implémentation	9
4.1	Architecture Logicielle.....	9
4.2	Design Patterns Utilisés.....	9
4.2.1	Pattern Singleton.....	9
4.2.2	Pattern DAO (Data Access Object)	9
5	Interface Utilisateur et Tests	10
5.1	Présentation des Interfaces.....	10
5.1.1	Page acuiel.....	11
5.1.2	Page de Connexion.....	12
5.1.3	Tableau de Bord Utilisateur	13
5.1.4	Tableau de Bord Administrateur	14
5.2	Page Desktope.....	15
5.2.1	Dachbord admin.....	15

6 Conclusion et Perspectives	16
6.1 Bilan Technique.....	16
6.2 Bilan Personnel.....	16
6.3 Difficultés Rencontrées	16
6.4 Perspectives d'Amélioration	17
Webographie	18

Chapitre 1

Introduction Générale

1.1 Contexte du Projet

Dans le cadre de l'évolution vers les "Smart Cities", la gestion efficace des infrastructures urbaines est devenue un enjeu majeur pour les municipalités. **SmartCity Reporter** s'inscrit dans cette dynamique de modernisation en proposant une plateforme numérique permettant de centraliser les signalements citoyens (pannes d'éclairage, nids-de-poule, gestion des déchets). Ce projet vise à renforcer la proximité entre les usagers de la ville et les services techniques via l'utilisation de technologies Java avancées.

1.2 Problématique

Actuellement, le processus de signalement d'incidents urbains souffre souvent de plusieurs lacunes:

- **Manque de réactivité**: Les canaux de communication traditionnels (appels, courrier) sont lents et peu structurés.
- **Absence de suivi** : Les citoyens n'ont pas de visibilité sur l'état d'avancement de leurs requêtes.
- **Fragmentation des données**: Les services administratifs ont du mal à consolider les informations provenant de sources disparates pour prioriser les interventions.

1.3 Objectifs du Projet

L'objectif principal est de développer une solution hybride robuste répondant aux besoins de deux types d'acteurs :

1. **Interface Web (Citoyen)** : Offrir une plateforme accessible permettant aux citoyens de soumettre un signalement détaillé avec description et localisation en quelques secondes.
2. **Module Desktop (Administration)** : Fournir aux autorités un outil performant sous JavaFX pour visualiser, trier et mettre à jour le statut des incidents en temps réel.

-
3. **Centralisation Technique** : Garantir la cohérence des données grâce à une base de données MySQL partagée et une couche de persistance gérée par Hibernate.

Chapitre 2

Analyse et Conception

2.1 Spécification des Besoins

2.1.1 Besoins Fonctionnels

ID	Description du Besoin
BF01	Remplir un formulaire (type, lieu, description) via l'interface Web.
BF02	Sauvegarder automatiquement les données via Hibernate dans MySQL.
BF03	Afficher la liste des incidents en temps réel sur l'application Desktop.
BF04	L'utilisateur doit pouvoir ajouter des ressources à ses favoris.
BF05	Modifier l'état d'un signalement (En attente, Résolu, Rejeté).

TABLE 2.1 – Besoins Fonctionnels

2.1.2 Besoins Non-Fonctionnels

ID	Description du Besoin
BNF01	Le traitement d'un signalement doit se faire en moins de 2 secondes.
BNF02	Le serveur Web doit être opérationnel en permanence pour les citoyens.
BNF03	Utilisation de transactions Hibernate pour éviter toute perte de données.
BNF04	Interfaces intuitives sans besoin de formation (Thymeleaf et JavaFX).
BNF05	Le système doit informer l'utilisateur en cas de coupure de la base de données.

TABLE 2.2 – Besoins Non-Fonctionnels

2.2 Conception UML

La conception logicielle repose sur une approche orientée objet pour garantir la modularité du code.

2.2.1 Diagramme de Classes

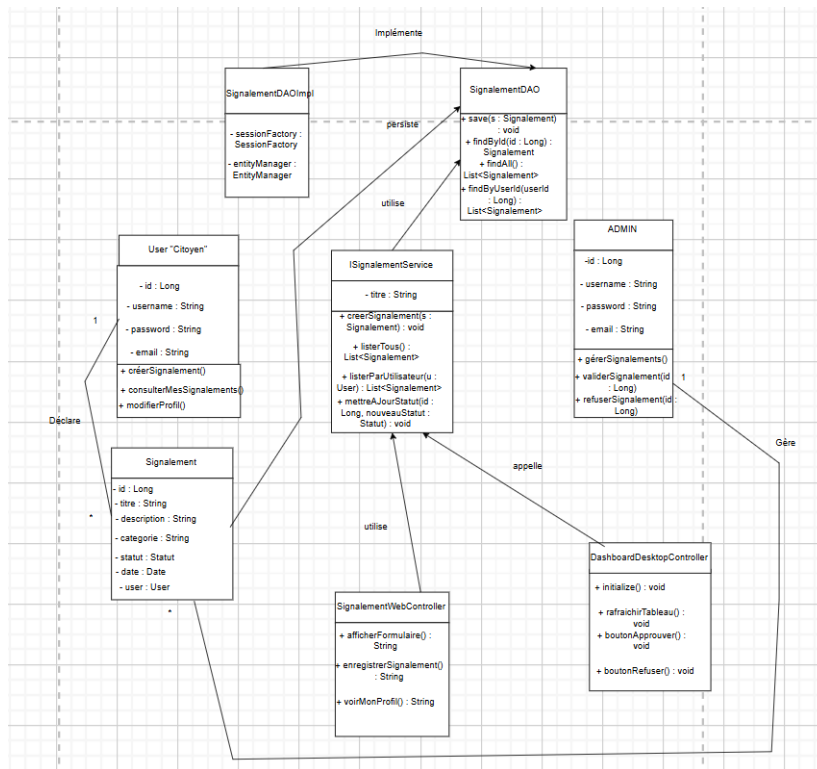


FIGURE 2.1 – Diagramme de Classes

Le diagramme de classes de l'application **SmartCity Reporter** modélise la structure statique du système en définissant les entités métier et leurs interactions selon une architecture **N-Tier**. Au cœur de ce modèle se trouve l'entité **Signalement**, qui encapsule les attributs essentiels tels que le type d'incident, la localisation et le statut. Pour assurer une séparation stricte entre la logique métier et le stockage des données, nous avons implémenté le pattern **DAO (Data Access Object)**. L'interface **ISignalementDAO** définit le contrat des opérations CRUD (Create, Read, Update, Delete), tandis que sa classe d'implémentation, **SignalementDAOImpl**, utilise l'API **Hibernate** pour interagir avec la base de données MySQL. Cette structure est complétée par une classe utilitaire **HibernateUtil**, basée sur le pattern **Singleton**, qui garantit une gestion unique et optimisée de la SessionFactory. Cette organisation garantit non seulement la modularité du code, facilitant ainsi la maintenance et l'évolution du système, mais assure également une synchronisation fluide entre les couches de présentation Web et Desktop.

Chapitre 3

Environnement Technique

3.1 Langage de Programmation

Le choix s'est porté sur le langage Java 17 (LTS) pour le développement de l'intégralité de la solution. Java offre une robustesse et une portabilité indispensables pour un système hybride. La version 17 apporte des améliorations significatives en termes de performance et de sécurité, tout en supportant pleinement les frameworks modernes comme Spring Boot et JavaFX utilisés dans ce projet.

3.2 Environnement de Développement

Pour la réalisation de ce projet, nous avons utilisé des outils professionnels garantissant une productivité optimale :

- **VS Code (Visual Studio Code)** : Utilisé comme environnement de développement principal pour sa légèreté et ses extensions puissantes pour Java et Spring.
- **Scene Builder** : Un outil de conception visuelle indispensable pour construire les interfaces utilisateur JavaFX de manière ergonomique.

3.3 Gestion de Projet avec Maven

Maven est utilisé pour automatiser la gestion des dépendances et le cycle de build. Voici les dépendances principales :

- **Gérer les dépendances** : Centraliser toutes les bibliothèques (Spring Boot, Hibernate, JavaFX, MySQL Connector) dans le fichier pom.xml.
- **Standardiser la structure** : Assurer que le projet respecte l'arborescence standard des applications Java professionnelles.
- **Faciliter le Build** : Automatiser la compilation et la génération des fichiers exécutables.

3.4 Base de Données

Pour le stockage des signalements, nous avons utilisé **MySQL**. C'est un système relationnel fiable qui permet une synchronisation parfaite entre :

- L'application **Web** (qui écrit les nouveaux signalements).
- L'application **Desktop** (qui lit et met à jour les statuts).

3.5 Frameworks Spécifiques

- Spring Boot : Pour créer la plateforme web citoyenne rapidement avec un serveur intégré.
- Hibernate (ORM) : Pour faire le lien entre nos classes Java et les tables MySQL sans écrire de requêtes SQL complexes.
- JavaFX : Pour développer une interface administrateur moderne et réactive sur ordinateur.

Chapitre 4

Architecture et Implémentation

4.1 Architecture Logicielle

Le projet SmartCity Reporter repose sur une architecture N-Tier (multi-couches). Cette approche permet de séparer les responsabilités pour rendre l'application plus évolutive et facile à maintenir :

- Couche de Présentation : Composée des interfaces Web (Thymeleaf) et Desktop (JavaFX).
- Couche Service (Métier) : Gère la logique de validation des signalements.
- Couche d'Accès aux Données (DAO) : Assure la communication avec MySQL via le framework Hibernate.

4.2 Design Patterns Utilisés

Pour garantir la qualité du code, deux patrons de conception fondamentaux ont été appliqués :

4.2.1 Pattern Singleton

Ce pattern est utilisé pour la gestion de la connexion à la base de données via la classe HibernateUtil. Il garantit qu'une seule instance de la SessionFactory est créée, évitant ainsi une consommation excessive de mémoire et des conflits de connexion.

4.2.2 Pattern DAO (Data Access Object)

Le pattern DAO permet de découpler la logique métier de la persistance des données.

- Interface ISignalementDAO : Définit les méthodes (ajouter, lister, modifier).
- Classe SignalementDAOImpl : Contient l'implémentation concrète utilisant Hibernate. Cela permet de changer de base de données sans impacter le reste de l'appli

Chapitre 5

Interface Utilisateur et Tests

5.1 Présentation des Interfaces

5.1.1 Page d'accueil

La page d'accueil constitue le point d'entrée principal de la plateforme SmartCity Reporter. Elle a été conçue pour offrir une navigation fluide et intuitive. Cette interface présente brièvement la mission du projet et propose des boutons d'appel à l'action (Call-to-Action) permettant aux citoyens d'accéder rapidement au formulaire de signalement ou à l'espace de connexion.

Éléments visibles : Barre de navigation, bannière descriptive, et accès direct aux services.



SC SmartCity

Connexion

Signaler un incident

Votre contribution aide à améliorer la qualité de vie de nos quartiers

NOM COMPLET

Mohammed Alami

EMAIL

alami@mail.com

TÉLÉPHONE

+212 6...

LOCALISATION

Rue, Quartier...

CATÉGORIE DE L'INCIDENT

Éclairage public (Lampadaire en panne)

DESCRIPTION DU PROBLÈME

Décrivez les détails de l'incident...

ENVOYER LE SIGNALEMENT

SmartCity Reporter

Solution technologique pour la gestion urbaine moderne.
Ensemble, construisons des villes plus intelligentes.

EMSI Casablanca

contact@smartcity.ma

© 2026 SmartCity Reporter – Projet d'innovation EMSI Casablanca

FIGURE 4.2 – Page de d'accueil

5.1.2 Page de Connexion

Cette interface permet de sécuriser l'accès à la plateforme. Elle utilise un système d'authentification lié à la base de données MySQL. L'utilisateur doit saisir ses identifiants pour accéder à son espace personnel. La validation des champs est gérée côté serveur pour prévenir les accès non autorisés.

Éléments visibles : Champs de saisie (Email/Login et Mot de passe), bouton de validation, et lien de récupération de compte.

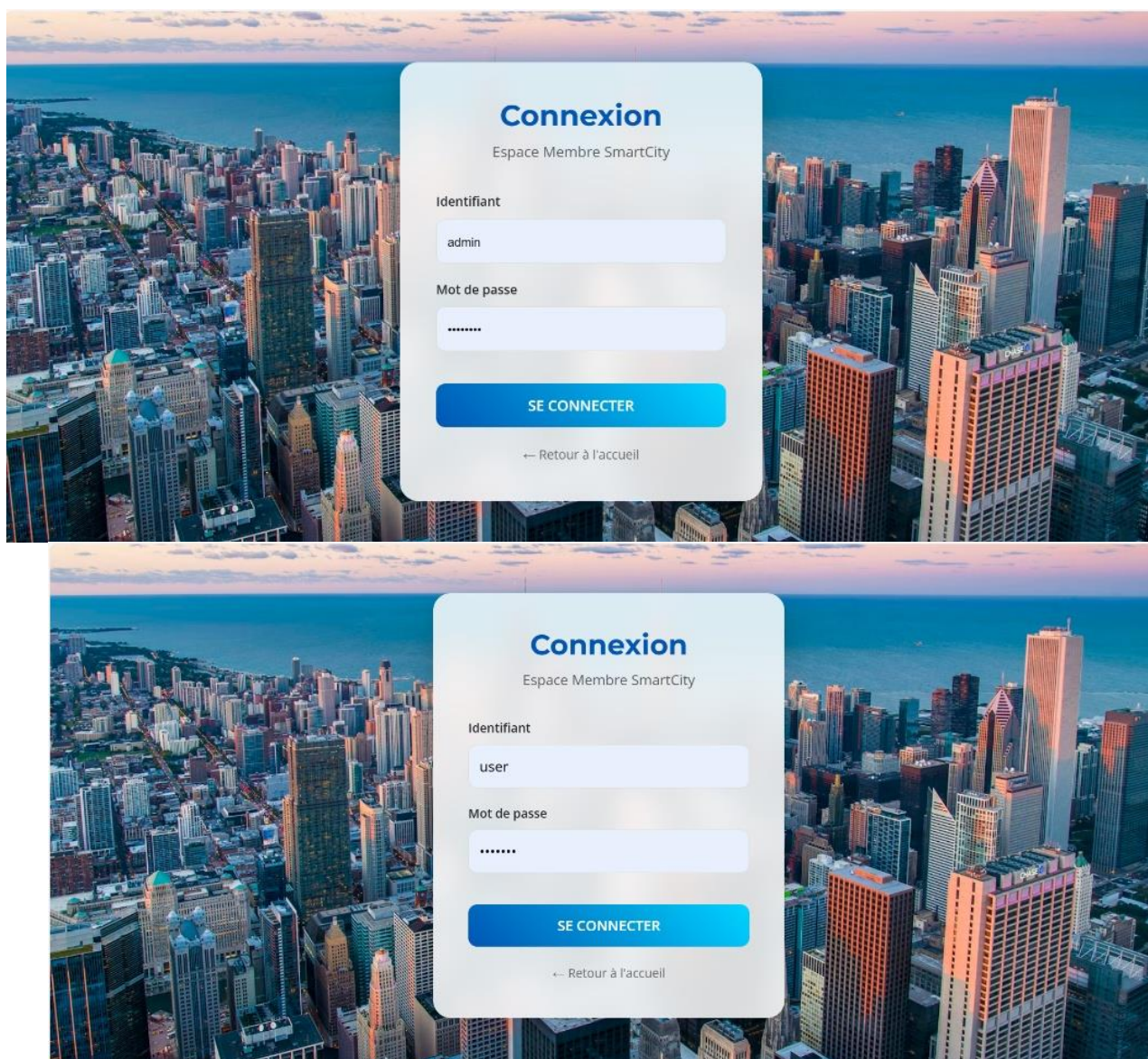



FIGURE 5.2 – Tableau de Bord Administrateur

5.1.3 Tableau de Bord Utilisateur (Web)

Une fois connecté, le citoyen accède à cet espace dédié. Il y trouve le formulaire principal pour soumettre un incident.

□ **Fonctionnalité** : Le citoyen choisit le type d'incident (via une liste déroulante), décrit le problème et précise sa localisation.

□ **Suivi** : Un tableau récapitulatif affiche l'historique des signalements envoyés par l'utilisateur avec l'état d'avancement mis à jour en temps réel.

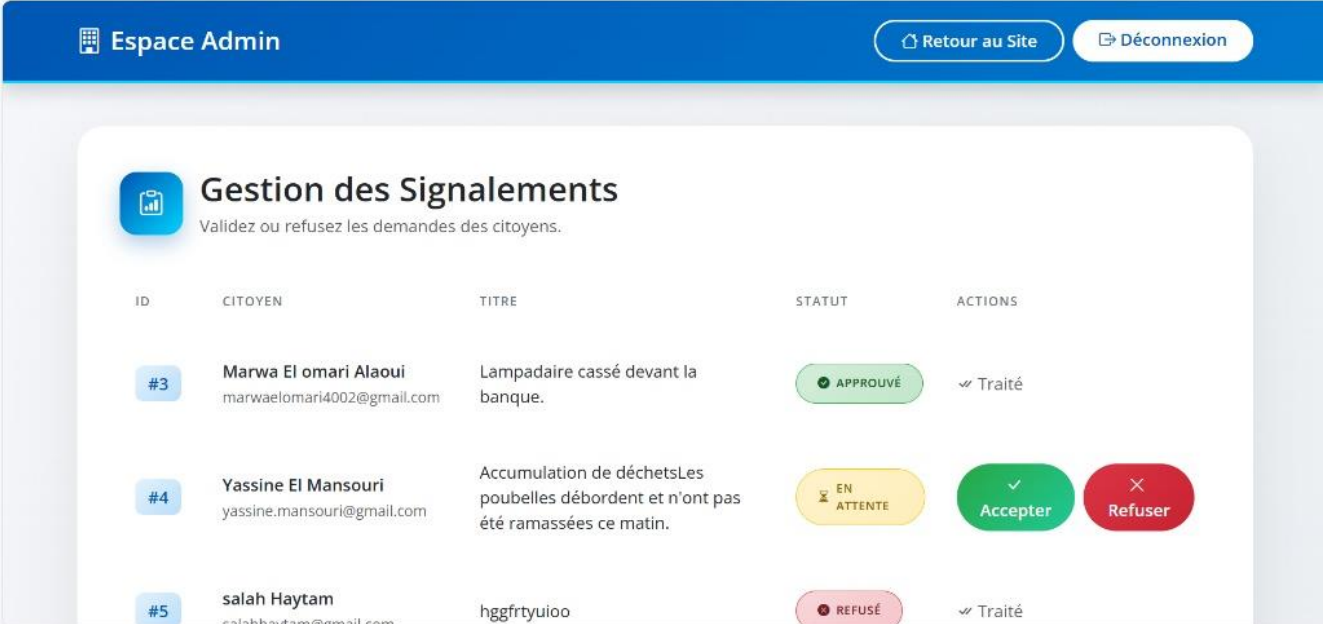


RÉF.	DATE	TITRE	TYPE	STATUT
#3	2026-01-08	Lampadaire cassé devant la banque.	ECLAIRAGE	APPROUVÉ
#4	2026-01-08	Accumulation de déchetsLes poubelles débordent et n'ont pas été ramassées ce matin.	ECLAIRAGE	EN ATTENTE
#5	2026-01-09	hggrfryuiioo	VOIRIE	REFUSÉ

FIGURE 5.3 – Tableau de Bord Utilisateur

5.1.4 Tableau de Bord Administrateur (Web)

Cette page permet une gestion simplifiée des signalements via le navigateur. Elle offre une vue d'ensemble des statistiques (nombre d'incidents ouverts, résolus, etc.) et permet de consulter les détails de chaque requête.



ID	CITOYEN	TITRE	STATUT	ACTIONS
#3	Marwa El omari Alaoui marwaelomari4002@gmail.com	Lampadaire cassé devant la banque.	APPROUVÉ	✓ Traité
#4	Yassine El Mansouri yassine.mansouri@gmail.com	Accumulation de déchetsLes poubelles débordent et n'ont pas été ramassées ce matin.	EN ATTENTE	✓ Accepter ✗ Refuser
#5	salah Haytam salahhaytam@gmail.com	hggfrtyuioo	REFUSÉ	✓ Traité

FIGURE 5.4 – Tableau de Bord Administrateur

5.1 Page Desktop (JavaFX)

5.1.1 Dashboard Admin (Interface de gestion)

L'application de bureau, développée en JavaFX, est l'outil principal des autorités pour la gestion des données massives.

- **Visualisation** : La capture montre un composant TableView qui charge dynamiquement tous les signalements stockés dans MySQL via Hibernate.
- **Interactivité** : L'administrateur peut sélectionner une ligne spécifique. Un panneau latéral ou une fenêtre contextuelle permet de modifier le statut de l'incident (ex: passer de "En attente" à "Approuvé").
- **Réactivité** : L'interface utilise le multi-threading pour garantir que l'application ne se fige pas lors de la récupération des données depuis le serveur.

Smart City Reporter - Gestion des Signalements

Tableau de Bord - Signalements

Liste des signalements

ID	Titre	Type	Statut	Date
3	Lampadaire cassé devant la banque.	ECLAIRAGE	APPROUVE	2026-01-08
4	Accumulation de déchetsLes poubelles débordent et n'ont pas été ramassées ce matin.	ECLAIRAGE	EN_ATTENTE	2026-01-08
5	hggrtyuioo	VOIRIE	REFUSE	2026-01-09
6	Le lampadaire ne s'allume plus depuis trois jours, la rue est très sombre le soir.	ECLAIRAGE	EN_ATTENTE	2026-01-09
7	hhhhhh	VOIRIE	APPROUVE	2026-01-09
8	nrjjuuu	DECHETS	EN_ATTENTE	2026-01-09
9	dghjjkkkk	VOIRIE	APPROUVE	2026-01-09
10	Accumulation de déchets Les poubelles débordent et n'ont pas été ramassées ce matin.	DECHETS	EN_ATTENTE	2026-01-09
11	Nid de poule dangereuxUn grand trou au milieu de la route qui risque d'endommager les voitures.	VOIRIE	REFUSE	2026-01-09
12	Accumulation d'ordures	ECLAIRAGE	EN_ATTENTE	2026-01-09
13	Les poubelles n'ont pas été ramassées depuis 48h et l'odeur devient insupportable.	VOIRIE	APPROUVE	2026-01-09
14	Fuite d'eau importante Une canalisation a éclaté sur le trottoir et l'eau inonde la rue.	AUTRE	EN_ATTENTE	2026-01-09
15	Nid de poule dangereuxUn grand trou au milieu de la route qui risque d'endommager les voitures.	VOIRIE	REFUSE	2026-01-10
16	hhvtftft	VOIRIE	APPROUVE	2026-01-11
17	GHDFJ VK KUUK	VOIRIE	EN_ATTENTE	2026-01-12

Détails / Nouveau

Titre:

Description:

Type:

Statut:

Auteur (Nom):

Email:

Localisation:

Date:

TABLE 5.1 – Application Desktop d'administration sous JavaFX

Chapitre 6

Conclusion et Perspectives

6.1 Bilan Technique

Le projet **SmartCity Reporter** a permis de mettre en place une solution logicielle complète et fonctionnelle, intégrant les technologies Java les plus récentes. L'objectif de créer un système hybride capable de synchroniser des données entre une plateforme Web (Spring Boot) et une application Desktop (JavaFX) a été atteint avec succès. L'utilisation du pattern DAO et d'Hibernate a assuré une gestion propre et sécurisée de la persistance des données.

6.2 Bilan Personnel

Sur le plan personnel, ce travail a été extrêmement formateur. Il m'a permis de :

- Approfondir mes connaissances sur l'architecture N-Tier.
- Maîtriser l'intégration d'un ORM (Hibernate) dans un projet réel.
- Comprendre les enjeux de la synchronisation de données multi-plateformes.
- Gérer un cycle de développement complet, de la conception UML à l'implémentation finale.

6.3 Difficultés Rencontrées

Tout au long de la réalisation, plusieurs défis techniques ont été relevés :

1. **Configuration Maven** : La gestion des dépendances JavaFX avec Spring Boot a nécessité une configuration précise du fichier pom.xml.
2. **Transactions Hibernate** : La gestion des sessions pour éviter les erreurs de type "LazyInitializationException" a été un point complexe.
3. **Threading JavaFX** : Assurer que la mise à jour de la TableView se fasse sur le thread principal de l'UI pour éviter les crashes de l'application.

6.4 Perspectives d'Amélioration

Le système actuel constitue une base solide qui pourrait évoluer vers :

- **Géolocalisation:** Intégration de l'API Google Maps pour placer les incidents sur une carte interactive.
- **Notification en temps réel:** Ajout d'un système d'alertes par email ou SMS pour informer les citoyens de l'avancement de leur signalement.
- **Application Mobile:** Développement d'un module Android/iOS pour faciliter les signalements directement sur le terrain.
- **Analyses Statistiques :** Ajout de graphiques (PieCharts) pour visualiser les zones les plus touchées par les incidents urbains.

Webographie

1. **Documentation Spring Boot** : <https://spring.io/projects/spring-boot>
2. **Documentation JavaFX** :-<https://openjfx.io/>
3. **Hibernate ORM Guide** : <https://hibernate.org/orm/>
4. **MySQL Tutorial** : <https://dev.mysql.com/doc/>