Université Internationale de Rabat

Projet de développement
Rapport

Application de Gestion de Cabinet Médical

Réalisé par :
Hajar Bezzanou
Hajar hajar

Enseignant :
Najib MEHDI

13 décembre 2025

Table des matières

Liste des figures

Liste des tableaux

1 Introduction Générale

Dans le contexte actuel de digitalisation du secteur médical, la gestion efficace d'un cabinet médical représente un enjeu majeur pour les professionnels de santé. La multiplication des patients, la complexité des plannings et le suivi des paiements nécessitent des outils informatiques adaptés et performants.

Ce projet consiste à développer une **application desktop de gestion de cabinet médical** permettant d'automatiser et d'optimiser les tâches quotidiennes du personnel médical. L'application offre une solution complète pour :

- La gestion des dossiers patients
- La planification et le suivi des consultations
- La gestion des paiements et de la facturation
- La génération de bilans et statistiques

L'application est développée en **Java** avec l'interface graphique **Swing**, utilisant **MySQL** comme système de gestion de base de données. L'architecture adoptée suit le pattern **Model-DAO-Service-Controller** garantissant une séparation claire des responsabilités et une maintenance facilitée.

Ce rapport présente les différentes étapes de réalisation du projet, depuis l'analyse des besoins jusqu'au guide d'utilisation, en passant par la conception et la structure de la base de données.

2 Analyse des Besoins et Détermination des Fonctionnalités

2.1 Introduction

Ce chapitre présente l'analyse des besoins fonctionnels et non fonctionnels de l'application de gestion de cabinet médical. Cette étape cruciale permet de définir clairement les fonctionnalités attendues et les contraintes techniques à respecter.

2.2 Besoins Fonctionnels

2.2.1 Acteurs du Système

L'application identifie deux acteurs principaux :

A. Médecin Le médecin est l'acteur principal du système. Il dispose des privilèges les plus élevés et peut effectuer les opérations suivantes :

Fonctionnalité	Description
Authentification	Se connecter au système avec ses identifiants
Gestion des patients	Ajouter, modifier, supprimer et rechercher des patients
Gestion des consultations	Créer, annuler et consulter les rendez-vous
Consultation du bilan	Visualiser les statistiques mensuelles
Visualisation des statistiques	Accéder aux graphiques et indicateurs de performance
Export PDF	Générer des rapports professionnels en PDF
Gestion de session	Se déconnecter de manière sécurisée

TABLE 1 – Fonctionnalités du Médecin

B. Assistant(e) Médical(e) L'assistant(e) dispose d'un accès limité aux fonctionnalités administratives :

Fonctionnalité	Description
Authentification	Se connecter au système avec ses identifiants
Gestion des patients	Ajouter, modifier, supprimer et rechercher des patients
Gestion des paiements	Enregistrer les paiements des consultations
Visualisation des impayés	Consulter la liste des consultations non payées
Gestion de session	Se déconnecter de manière sécurisée

TABLE 2 – Fonctionnalités de l'Assistant

2.2.2 Fonctionnalités Détaillées

Module d'Authentification

- Connexion sécurisée avec login et mot de passe
- Hachage des mots de passe avec BCrypt
- Option "Se souvenir de moi" pour connexion persistante
- Timeout automatique après inactivité
- Journal des actions utilisateur (audit trail)

Module Gestion des Patients

- Enregistrement des informations patient (nom, prénom, téléphone, adresse)
- Recherche de patients par nom
- Modification des informations
- Suppression de patients
- Affichage en tableau avec mise à jour en temps réel

Module Gestion des Consultations

- Création de consultations avec sélection de date/heure
- Catégories : Consultation normale, Consultation spécialisée, Contrôle, Visite à domicile

— Tarification personnalisée selon la catégorie
— Suivi du statut de paiement
— Annulation de consultations

Module Gestion des Paiements
— Liste des consultations impayées
— Enregistrement des paiements
— Calcul automatique du total impayé
— Historique des transactions

Module Bilan et Statistiques
— Bilan mensuel avec statistiques générales
— Graphiques d'évolution (courbes, histogrammes, camemberts)
— Indicateurs clés de performance (KPIs)
— Export des rapports en PDF

2.3 Besoins Non Fonctionnels

2.3.1 Exigences de Qualité

Exigence	Description
Fiabilité	L'application doit fonctionner de manière stable sans interruption
Maintenabilité	Le code doit être structuré et documenté pour faciliter les évolutions
Utilisabilité	L'interface doit être intuitive et accessible
Sécurité	Protection des données sensibles des patients

TABLE 3 – Exigences de Qualité

2.3.2 Exigences de Performance

Exigence	Spécification
Temps de réponse	Affichage des données en moins de 2 secondes
Disponibilité	Application utilisable en mode local (desktop)
Capacité	Support de plusieurs milliers de patients

TABLE 4 – Exigences de Performance

2.3.3 Contraintes de Conception

Contrainte	Description
Langage	Java JDK 21
Interface	Java Swing avec NetBeans GUI Builder
Base de données	MySQL avec JDBC
Architecture	Pattern Model-DAO-Service-Controller
Sécurité	BCrypt pour le hachage des mots de passe
Rapports	iTextPDF pour la génération de PDF
Graphiques	JFreeChart pour les visualisations

TABLE 5 – Contraintes de Conception

2.4 Conclusion

L'analyse des besoins a permis d'identifier clairement les fonctionnalités attendues pour chaque acteur du système. Cette étude constitue la base de la conception et du développement de l'application, garantissant une couverture complète des besoins du cabinet médical.

3 Chapitre 2 : Modélisation et Architecture du Système

3.1 Introduction

Ce chapitre présente la modélisation UML de l'application de gestion de cabinet médical. Les diagrammes de cas d'utilisation décrivent les interactions entre les acteurs et le système, tandis que les diagrammes de classes illustrent la structure statique de l'application.

3.2 Diagrammes de Cas d'Utilisation

3.2.1 Diagramme de Cas d'Utilisation Global

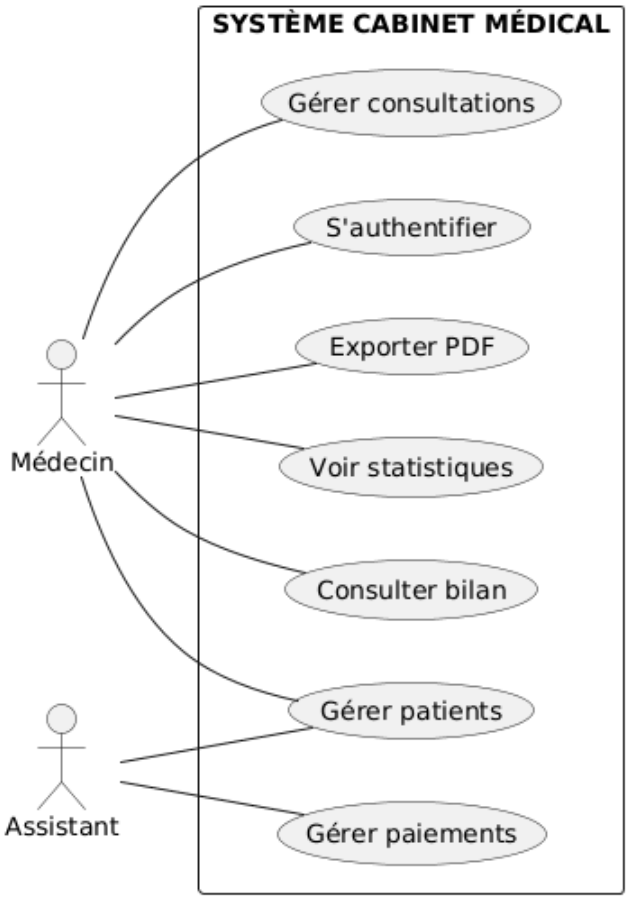

FIGURE 1 – Diagramme de cas d'utilisation global

Ce diagramme présente une vue d'ensemble des interactions entre les deux acteurs principaux (Médecin et Assistant) et le système de gestion du cabinet médical.

3.2.2 Diagramme de Cas d'Utilisation - Médecin

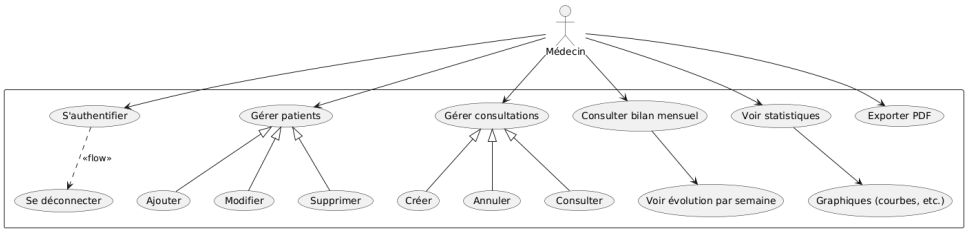

FIGURE 2 – Diagramme de cas d'utilisation Médecin

Ce diagramme détaille les cas d'utilisation spécifiques au médecin, incluant la gestion complète des patients, consultations, bilans et statistiques.

3.2.3 Diagramme de Cas d'Utilisation - Assistant

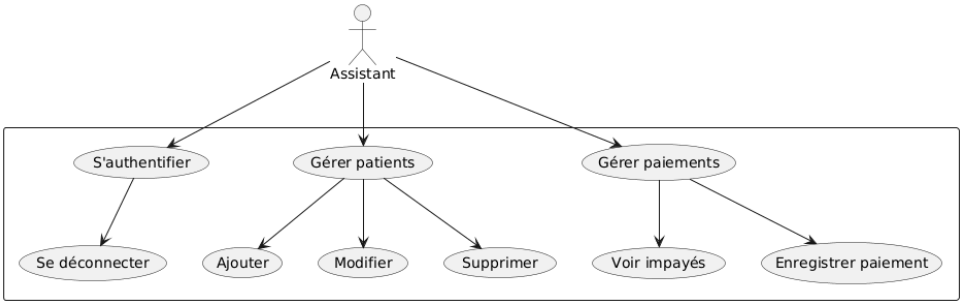

FIGURE 3 – Diagramme de cas d'utilisation Assistant

Ce diagramme présente les cas d'utilisation de l'assistant médical, axés sur la gestion des patients et des paiements.

3.3 Diagrammes de Classes

3.3.1 Diagramme de Classes - Entités (Modèles)

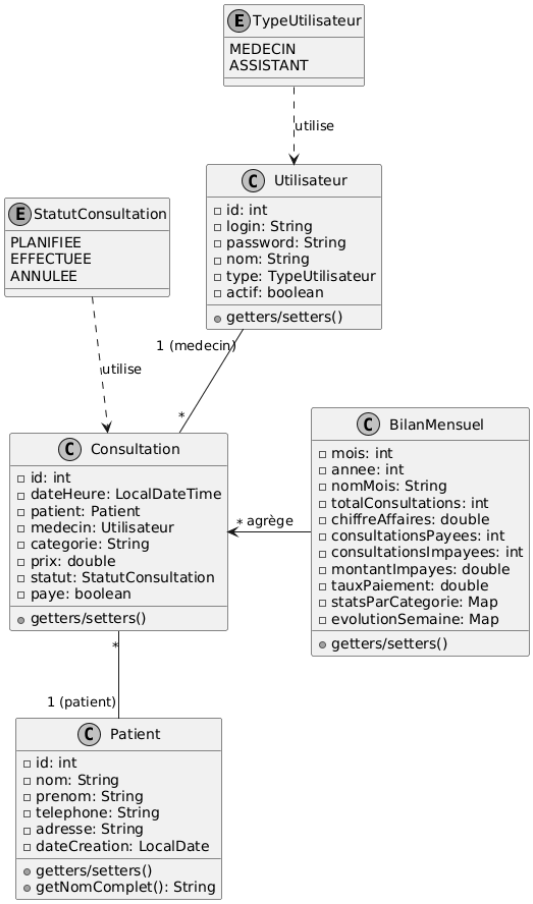

FIGURE 4 – Diagramme de classes - Entités

Ce diagramme représente les classes entités du système avec leurs attributs et relations. Il illustre la structure des données métier de l'application.

3.3.2 Diagramme de Classes - Couche DAO

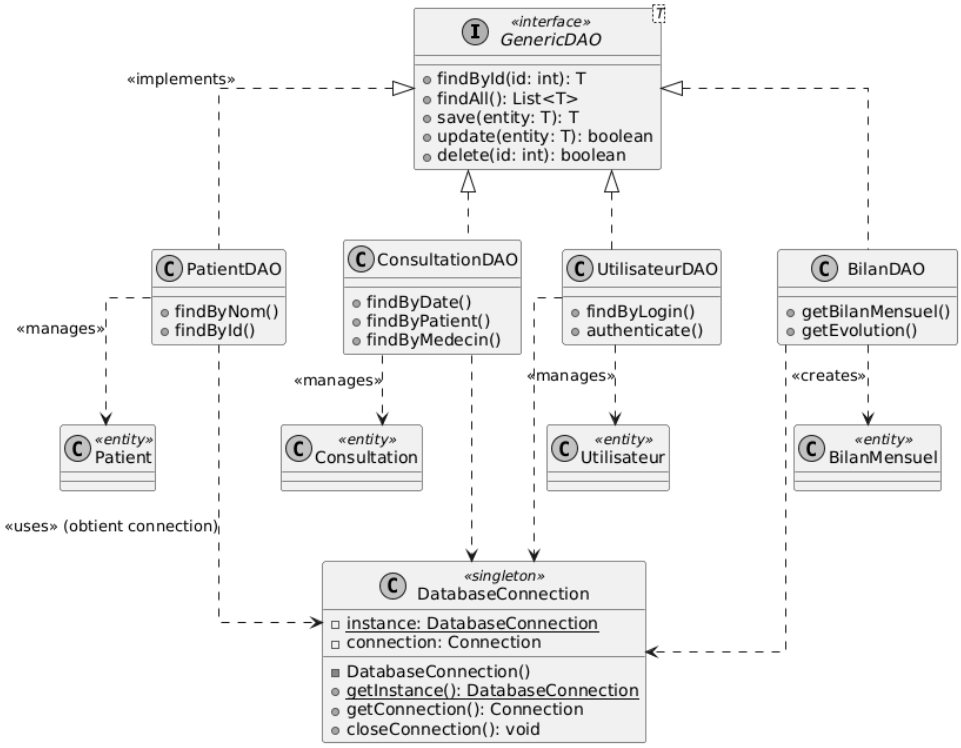

FIGURE 5 – Diagramme de classes - Couche DAO

Ce diagramme illustre la couche d'accès aux données (Data Access Object) qui encapsule toutes les opérations de persistance avec la base de données MySQL.

3.3.3 Diagramme de Classes - Couche Service

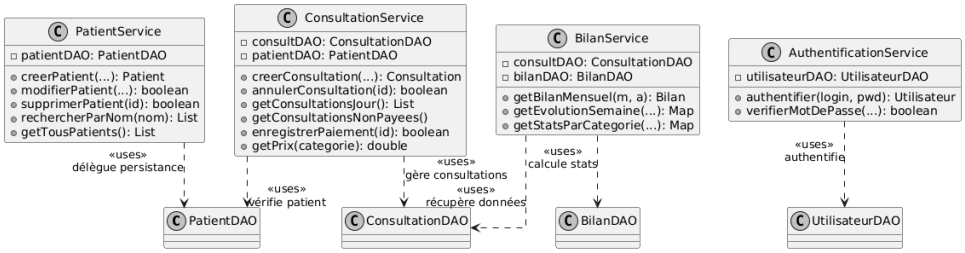

FIGURE 6 – Diagramme de classes - Couche Service

Ce diagramme représente la couche métier qui contient la logique applicative et orchestre les opérations entre les contrôleurs et la couche DAO.

3.3.4 Diagramme de Classes - Couche Controller

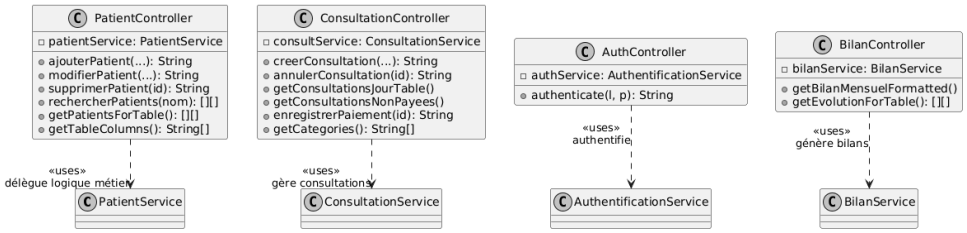

FIGURE 7 – Diagramme de classes - Couche Controller

Ce diagramme présente la couche contrôleur qui fait le lien entre l'interface utilisateur Swing et la couche service.

3.4 Conclusion

La phase de conception a permis de modéliser l'ensemble du système à travers les diagrammes UML. L'architecture en couches (Model-DAO-Service-Controller) garantit une séparation claire des responsabilités et facilite la maintenance et l'évolution de l'application.

4 Chapitre 3 : Modélisation et Structure de la Base de Données

4.1 Introduction

Ce chapitre présente la structure de la base de données MySQL utilisée par l'application. Le dictionnaire de données détaille chaque table et ses attributs, tandis que le schéma relationnel illustre les liens entre les différentes entités.

4.2 Dictionnaire de Données

4.2.1 Table `utilisateur`

Attribut	Type	Taille	Contraintes	Description
id	INT	-	PRIMARY KEY, AUTO_INCREMENT	Identifiant unique de l'utilisateur
login	VARCHAR	50	UNIQUE, NOT NULL	Nom d'utilisateur pour la connexion
password	VARCHAR	255	NOT NULL	Mot de passe hashé avec BCrypt
nom	VARCHAR	100	NOT NULL	Nom complet de l'utilisateur
type	ENUM	-	NOT NULL	Type : 'MEDECIN' ou 'ASSISTANT'
actif	BOOLEAN	-	DEFAULT TRUE	Statut actif/inactif du compte

Table 6 – Structure de la table utilisateur

4.2.2 Table `patient`

Attribut	Type	Taille	Contraintes	Description
id	INT	-	PRIMARY KEY, AUTO_INCREMENT	Identifiant unique du patient
nom	VARCHAR	100	NOT NULL	Nom de famille du patient
prenom	VARCHAR	100	NOT NULL	Prénom du patient
telephone	VARCHAR	20	-	Numéro de téléphone
adresse	TEXT	-	-	Adresse postale complète
date_creation	DATE	-	DEFAULT CURRENT_DATE	Date d'enregistrement

Table 7 – Structure de la table patient

4.2.3 Table `consultation`

Attribut	Type	Taille	Contraintes	Description
id	INT	-	PRIMARY KEY, AUTO_INCREMENT	Identifiant unique de la consultation
date_heure	DATETIME	-	NOT NULL	Date et heure de la consultation
patient_id	INT	-	FOREIGN KEY, NOT NULL	Référence vers le patient
medecin_id	INT	-	FOREIGN KEY, NOT NULL	Référence vers le médecin
categorie	VARCHAR	50	NOT NULL	Type de consultation
prix	DECIMAL	10,2	NOT NULL	Montant de la consultation en DH
statut	ENUM	-	DEFAULT 'PLANIFIEE'	Statut : PLANIFIEE, EFFECTUEE, ANNULEE
paye	BOOLEAN	-	DEFAULT FALSE	Indicateur de paiement

Table 8 – Structure de la table consultation

4.2.4 Table `categorie_consultation`

Attribut	Type	Taille	Contraintes	Description
id	INT	-	PRIMARY KEY, AUTO_INCREMENT	Identifiant unique
nom	VARCHAR	100	UNIQUE, NOT NULL	Nom de la catégorie
prix_defaut	DECIMAL	10,2	NOT NULL	Prix par défaut en DH

Table 9 – Structure de la table categorie_consultation

4.3 Structure des Tables MySQL

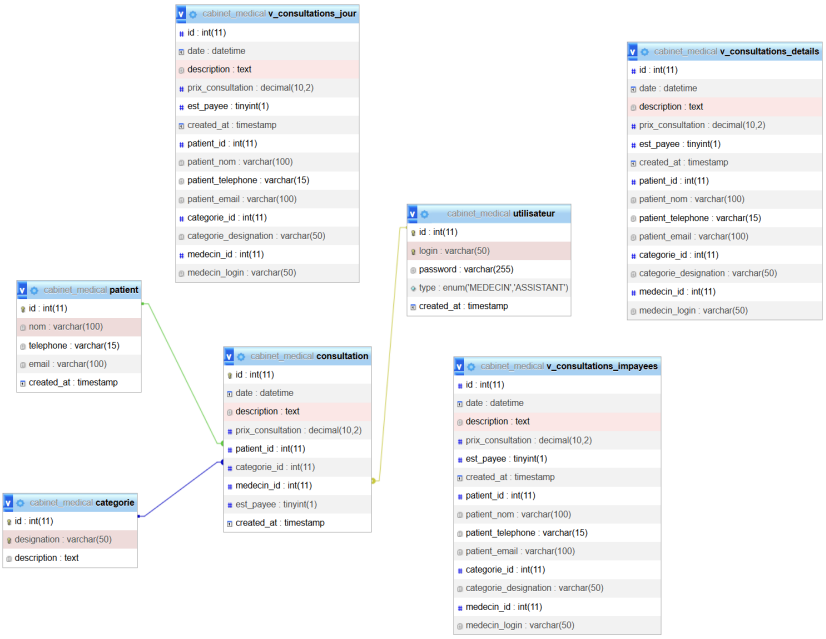

FIGURE 8 – Schéma de la base de données

Ce schéma illustre les relations entre les différentes tables de la base de données cabinet_medical, avec les clés primaires et étrangères.

4.3.1 Script de Création de la Base de Données

Les scripts SQL de création de la base de données et d'insertion des données de test sont disponibles dans les fichiers sources du projet.

4.4 Conclusion

La base de données a été conçue pour répondre aux besoins fonctionnels identifiés tout en respectant les règles de normalisation. Les relations entre les tables permettent une gestion cohérente des données patients, consultations et utilisateurs.

5 Guide Utilisateur

5.1 Introduction

Ce chapitre présente un guide complet d'utilisation de l'application de gestion de cabinet médical. Chaque fonctionnalité est illustrée par des captures d'écran avec des explications détaillées.

5.2 Authentification

5.2.1 Écran de Connexion

FIGURE 9 – Interface de connexion

L'écran de connexion permet à l'utilisateur de s'authentifier avec son login et son mot de passe. L'option "Se souvenir de moi" permet de maintenir la session active.

Éléments de l'interface :

— **Champ Login** : Saisir votre identifiant (ex : dr.alami)
— **Champ Mot de passe** : Saisir votre mot de passe (masqué par défaut)
— **Bouton Voir/Cacher** : Afficher ou masquer le mot de passe
— **Case "Se souvenir de moi"** : Maintenir la connexion active
— **Bouton "Se connecter"** : Valider l'authentification

Procédure de connexion :

1. Entrer votre login dans le champ approprié
2. Saisir votre mot de passe
3. Cocher "Se souvenir de moi" si souhaité
4. Cliquer sur "Se connecter"

5.3 Espace Médecin

5.3.1 Tableau de Bord Médecin

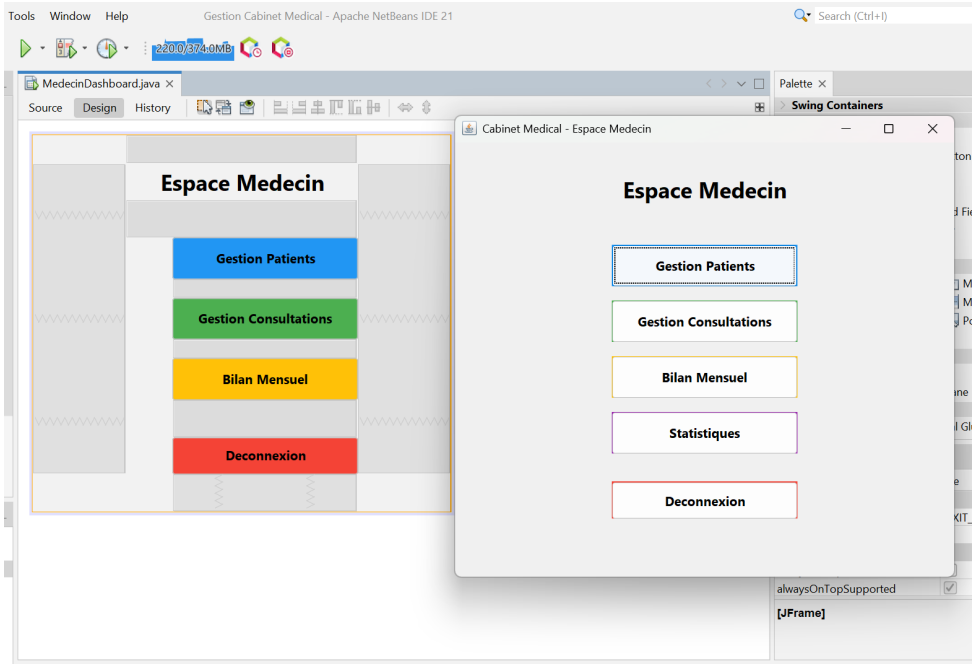

FIGURE 10 – Dashboard Médecin

Le tableau de bord du médecin donne accès à toutes les fonctionnalités principales : gestion des patients, des consultations, bilan mensuel, statistiques et déconnexion.

Boutons disponibles :

— **Gestion Patients** : Accéder à la gestion des dossiers patients
— **Gestion Consultations** : Planifier et gérer les rendez-vous
— **Bilan Mensuel** : Consulter les statistiques du mois
— **Statistiques** : Visualiser les graphiques et KPIs
— **Déconnexion** : Fermer la session en cours

5.3.2 Gestion des Patients

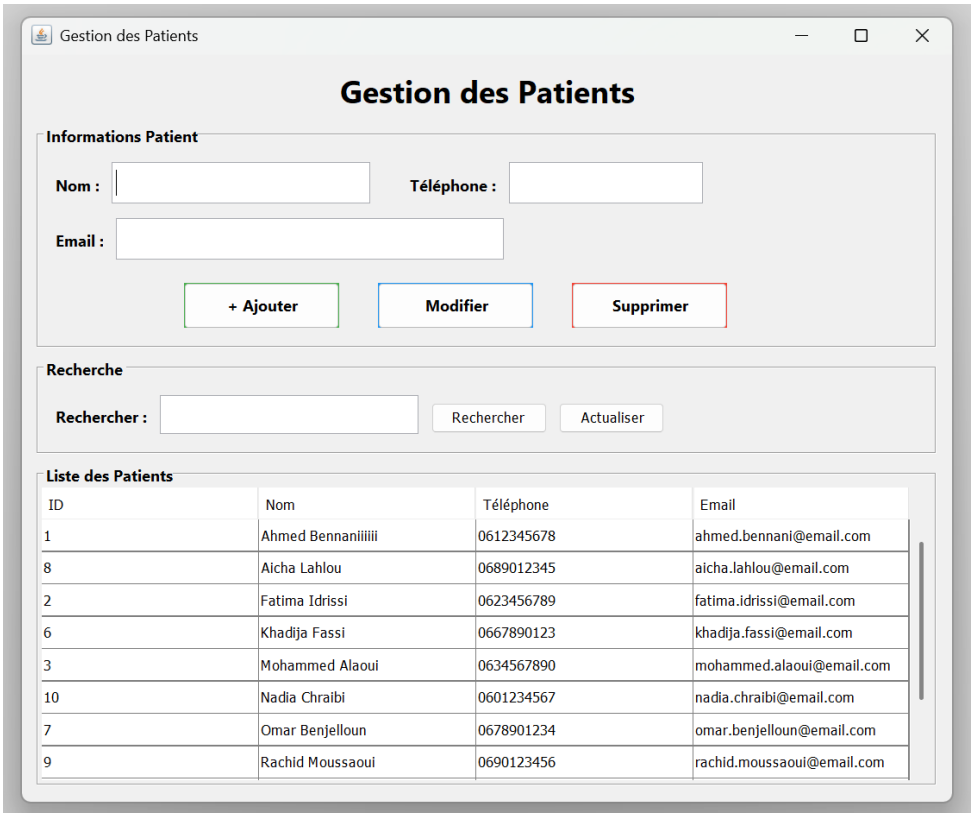

FIGURE 11 – Gestion des patients

Cette interface permet d'ajouter, modifier, supprimer et rechercher des patients. La liste des patients s'affiche dans un tableau avec mise à jour en temps réel.

Action	**Procédure**
Ajouter un patient	Remplir le formulaire (Nom, Prénom, Téléphone, Adresse) puis cliquer sur "Ajouter"
Modifier un patient	Sélectionner un patient dans le tableau, modifier les champs, cliquer sur "Modifier"
Supprimer un patient	Sélectionner un patient, cliquer sur "Supprimer", confirmer la suppression
Rechercher	Saisir un nom dans le champ de recherche, cliquer sur "Rechercher"
Actualiser	Cliquer sur "Actualiser" pour rafraîchir la liste

TABLE 10 – Procédures de gestion des patients

5.3.3 Gestion des Consultations

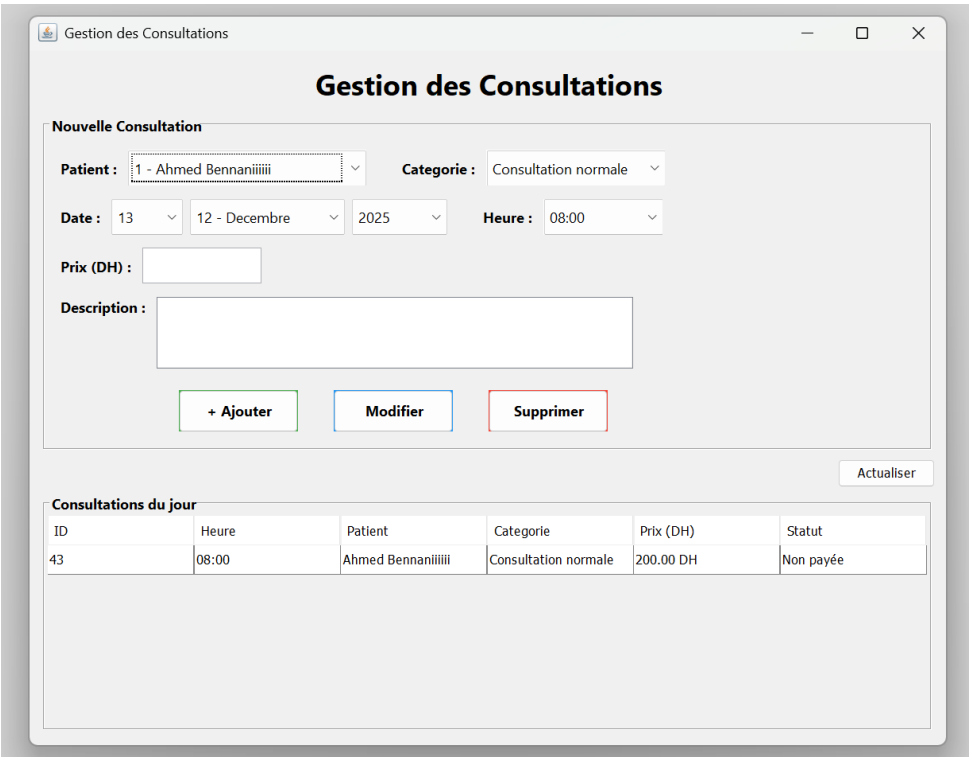

FIGURE 12 – Gestion des consultations

Cette interface permet de créer et gérer les consultations. La sélection de la date et de l'heure se fait via des listes déroulantes pour éviter les erreurs de saisie.

Éléments du formulaire :

— **Patient** : Liste déroulante des patients enregistrés
— **Date** : Sélection du jour, mois et année
— **Heure** : Créneaux horaires disponibles (08 :00 à 18 :00)
— **Catégorie** : Type de consultation
— **Prix** : Montant en DH (pré-rempli selon la catégorie)

Procédure pour créer une consultation :

1. Sélectionner le patient dans la liste
2. Choisir la date (jour/mois/année)
3. Sélectionner le créneau horaire
4. Choisir la catégorie de consultation
5. Vérifier/ajuster le prix si nécessaire
6. Cliquer sur "Créer Consultation"

5.3.4 Bilan Mensuel

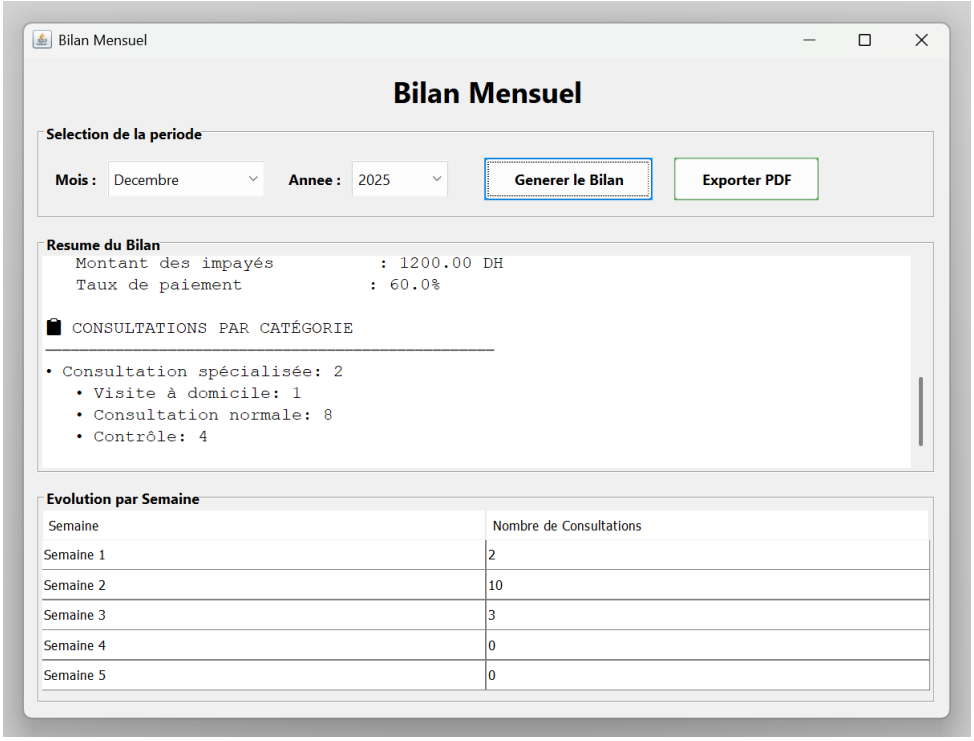

FIGURE 13 – Bilan mensuel

L'interface de bilan mensuel affiche les statistiques du mois sélectionné avec possibilité d'export en PDF.

Informations affichées :

— Nombre total de consultations
— Chiffre d'affaires total
— Prix moyen par consultation
— Consultations payées / impayées
— Taux de paiement
— Répartition par catégorie
— Évolution par semaine

Procédure :

1. Sélectionner le mois et l'année
2. Cliquer sur "Générer le Bilan"
3. Consulter les statistiques affichées
4. Cliquer sur "Exporter PDF" pour générer un rapport

5.3.5 Statistiques et Graphiques

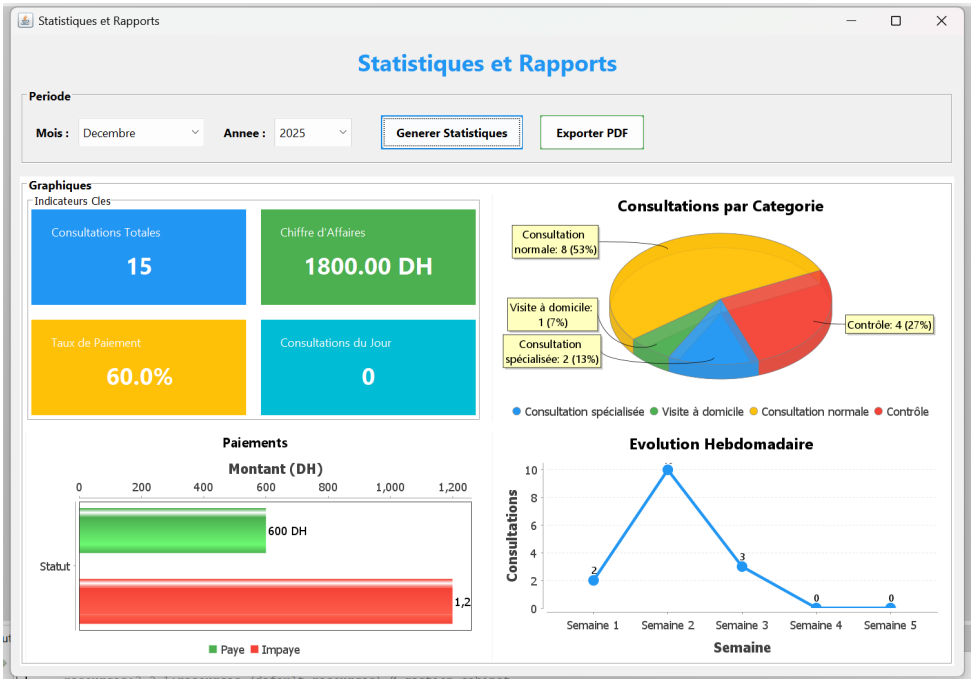

FIGURE 14 – Statistiques

L'écran des statistiques présente des visualisations graphiques (courbes, histogrammes, camemberts) et les indicateurs clés de performance (KPIs).

Types de graphiques :

— **Camembert** : Répartition des consultations par catégorie
— **Histogramme** : Chiffre d'affaires mensuel
— **Courbe** : Évolution des consultations par semaine
— **KPIs** : Indicateurs clés (total consultations, CA, taux paiement)

5.3.6 Export PDF

FIGURE 15 – Export PDF

Exemple de rapport PDF généré par l'application, contenant le bilan mensuel formaté de manière professionnelle.

Contenu du PDF :

— En-tête avec nom du cabinet et période
— Statistiques générales
— Statistiques de paiement
— Répartition par catégorie
— Date et heure de génération

5.4 Espace Assistant

5.4.1 Tableau de Bord Assistant

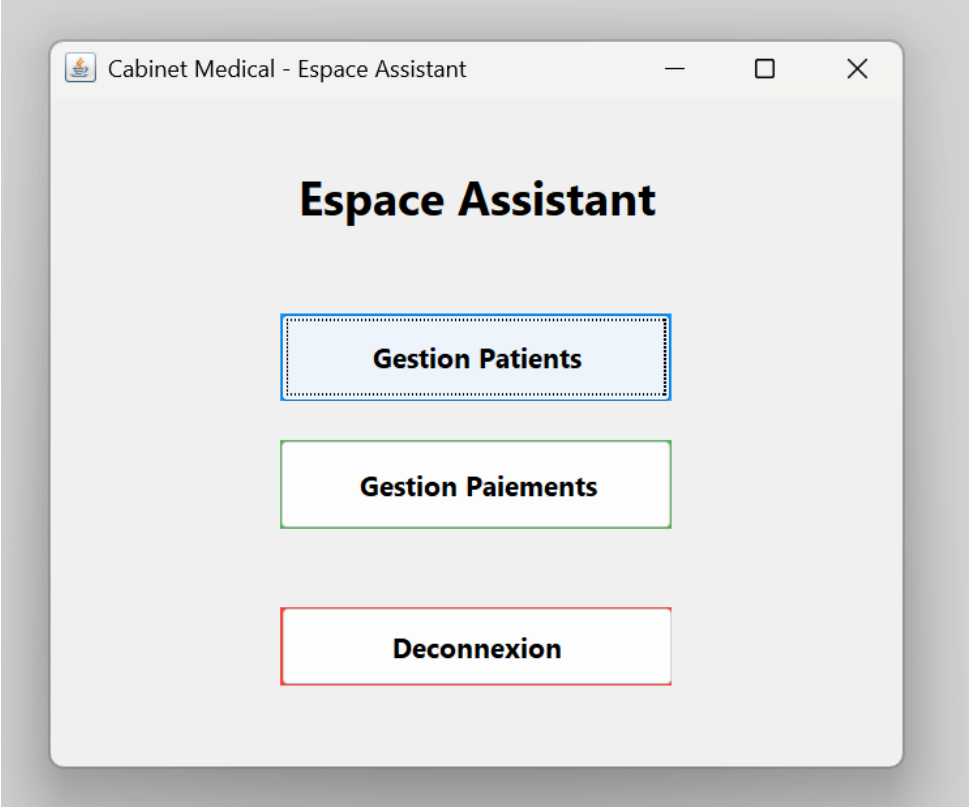

FIGURE 16 – Dashboard Assistant

Le tableau de bord de l'assistant donne accès aux fonctionnalités de gestion des patients et des paiements.

Boutons disponibles :

— **Gestion Patients** : Accéder à la gestion des dossiers patients
— **Gestion Paiements** : Enregistrer les paiements des consultations
— **Déconnexion** : Fermer la session en cours

5.4.2 Gestion des Paiements

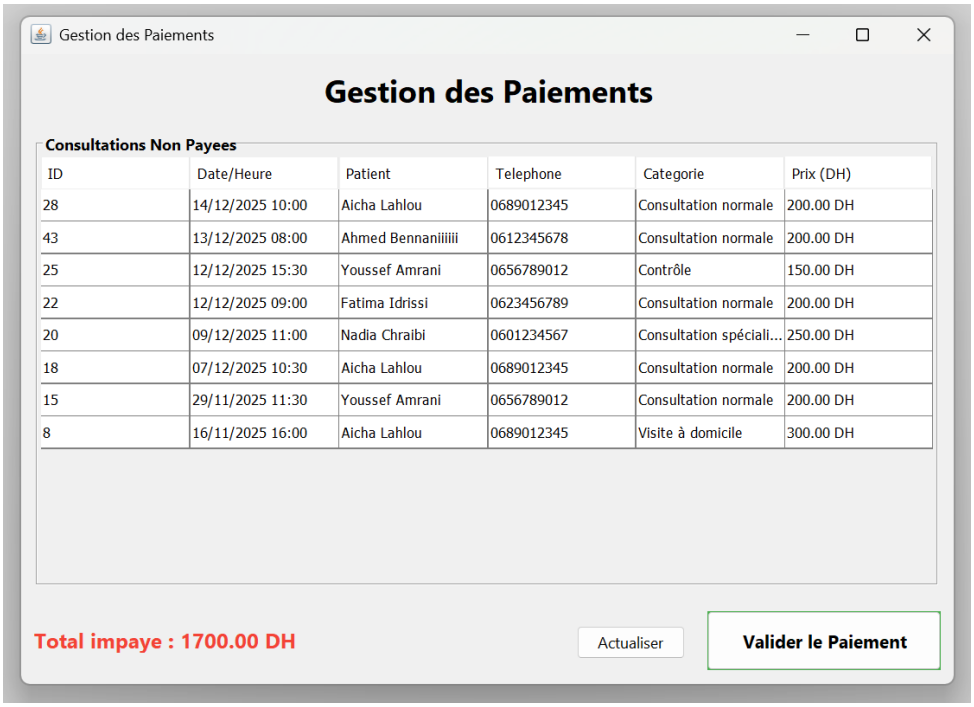

FIGURE 17 – Gestion des paiements

Cette interface affiche la liste des consultations impayées et permet d'enregistrer les paiements.

Colonnes du tableau :

— ID de la consultation
— Date et heure
— Nom du patient
— Téléphone
— Catégorie
— Prix (DH)

Procédure pour enregistrer un paiement :

1. Sélectionner la consultation dans le tableau
2. Cliquer sur "Enregistrer Paiement"
3. Confirmer l'opération
4. La consultation disparaît de la liste des impayés

Indicateur : Le total des impayés est affiché en bas de l'écran.

5.5 Conclusion

Ce guide utilisateur couvre l'ensemble des fonctionnalités de l'application. L'interface intuitive permet une prise en main rapide par le personnel médical et administratif du cabinet.

6 Conclusion Générale

Ce projet de développement d'une application de gestion de cabinet médical a permis de mettre en pratique les concepts fondamentaux de la programmation orientée objet et du développement d'applications desktop en Java.

Réalisations principales :
— Interface graphique complète avec Java Swing et NetBeans GUI Builder
— Architecture en couches (Model-DAO-Service-Controller)
— Persistance des données avec MySQL et JDBC
— Authentification sécurisée avec BCrypt
— Gestion complète des patients et consultations
— Système de paiements et suivi des impayés
— Bilans mensuels avec statistiques détaillées
— Visualisations graphiques avec JFreeChart
— Export de rapports en PDF avec iTextPDF
— Gestion de sessions avec "Se souvenir de moi"

Compétences développées :
— Conception et modélisation UML
— Développement Java avancé (JDK 21)
— Création d'interfaces graphiques avec Swing
— Gestion de bases de données relationnelles
— Application des patterns de conception
— Génération de documents PDF
— Création de graphiques et visualisations

Perspectives d'amélioration :
— Ajout d'un module de gestion d'agenda avec rappels
— Intégration d'un système de notifications par SMS/email
— Développement d'une version web complémentaire
— Ajout de la gestion des ordonnances
— Historique médical détaillé des patients

Ce projet constitue une base solide pour une application de gestion de cabinet médical, extensible et maintenable grâce à son architecture modulaire.

7 Annexes

7.1 Annexe A : Structure du Projet

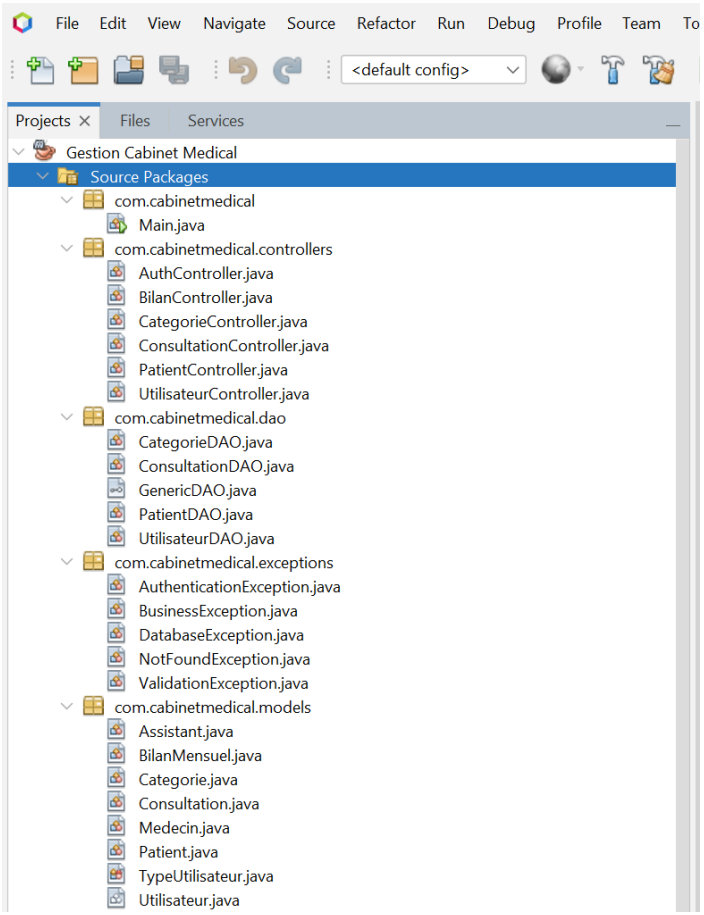

FIGURE 18

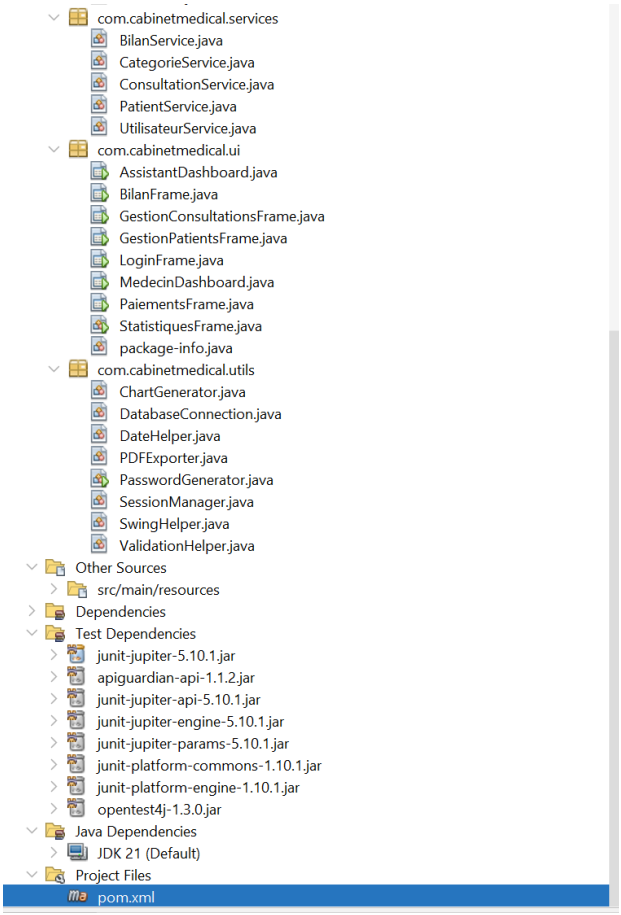
com.cabinetmedical.services
BilanService.java
CategorieService.java
ConsultationService.java
PatientService.java
UtilisateurService.java
com.cabinetmedical.ui
AssistantDashboard.java
BilanFrame.java
GestionConsultationsFrame.java
GestionPatientsFrame.java
LoginFrame.java
MedecinDashboard.java
PaiementsFrame.java
StatistiquesFrame.java
package-info.java
com.cabinetmedical.utils
ChartGenerator.java
DatabaseConnection.java
DateHelper.java
PDFExporter.java
PasswordGenerator.java
SessionManager.java
SwingHelper.java
ValidationHelper.java
Other Sources
src/main/resources
Dependencies
Test Dependencies
junit-jupiter-5.10.1.jar
apiguardian-api-1.1.2.jar
junit-jupiter-api-5.10.1.jar
junit-jupiter-engine-5.10.1.jar
junit-jupiter-params-5.10.1.jar
junit-platform-commons-1.10.1.jar
junit-platform-engine-1.10.1.jar
opentest4j-1.3.0.jar
Java Dependencies
JDK 21 (Default)
Project Files
pom.xml

FIGURE 19

7.2 Annexe B : Code Important - Authentification avec BCrypt

```
// AuthentificationService.java
public class AuthentificationService {

    private UtilisateurDAO utilisateurDAO;

    public AuthentificationService() {
        this.utilisateurDAO = new UtilisateurDAO();
    }

    public Utilisateur authentifier(String login, String motDePasse)
            throws AuthenticationException {

        if (login == null || login.trim().isEmpty()) {
            throw new AuthenticationException("Le login est requis");
        }
        if (motDePasse == null || motDePasse.trim().isEmpty()) {
            throw new AuthenticationException("Le mot de passe est
    requis");
        }

        Utilisateur utilisateur = utilisateurDAO.findByLogin(login.trim
    ());

        if (utilisateur == null) {
            throw new AuthenticationException("Login ou mot de passe
    incorrect");
        }

        if (!utilisateur.isActif()) {
            throw new AuthenticationException("Ce compte est désactivé")
    ;
        }

        // Vérification du mot de passe avec BCrypt
        if (!BCrypt.checkpw(motDePasse, utilisateur.getPassword())) {
            throw new AuthenticationException("Login ou mot de passe
    incorrect");
        }

        return utilisateur;
    }
}
```

Listing 1 – AuthentificationService.java

7.3 Annexe C : Code Important - Connexion Base de Données

```java
// DatabaseConnection.java (Singleton)
public class DatabaseConnection {

    private static DatabaseConnection instance;
    private Connection connection;

    private static final String URL = "jdbc:mysql://localhost:3306/
    cabinet_medical";
    private static final String USER = "root";
    private static final String PASSWORD = "";

    private DatabaseConnection() {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            this.connection = DriverManager.getConnection(URL, USER,
    PASSWORD);
        } catch (Exception e) {
            throw new DatabaseException("Erreur de connexion à la base
    de données", e);
        }
    }

    public static synchronized DatabaseConnection getInstance() {
        if (instance == null || instance.connection == null) {
            instance = new DatabaseConnection();
        }
        return instance;
    }

    public Connection getConnection() {
        return connection;
    }
}
```

Listing 2 – DatabaseConnection.java (Singleton)

7.4 Annexe D : Code Important - Export PDF

```
// PDFExporter.java
public class PDFExporter {

    public static void exportBilanMensuel(BilanMensuel bilan, String
   filePath)
            throws DocumentException, FileNotFoundException {

        Document document = new Document(PageSize.A4);
        PdfWriter.getInstance(document, new FileOutputStream(filePath));
        document.open();

        // Titre
        Font titleFont = new Font(Font.FontFamily.HELVETICA, 20, Font.
   BOLD);
        Paragraph title = new Paragraph("BILAN MENSUEL", titleFont);
        title.setAlignment(Element.ALIGN_CENTER);
        document.add(title);

        // Période
        Font subtitleFont = new Font(Font.FontFamily.HELVETICA, 14);
        Paragraph periode = new Paragraph(
            bilan.getNomMois() + " " + bilan.getAnnee(), subtitleFont);
        periode.setAlignment(Element.ALIGN_CENTER);
        document.add(periode);

        document.add(new Paragraph("\n"));

        // Statistiques
        PdfPTable table = new PdfPTable(2);
        table.setWidthPercentage(80);

        addTableRow(table, "Total consultations",
            String.valueOf(bilan.getTotalConsultations()));
        addTableRow(table, "Chiffre d'affaires",
            String.format("%.2f DH", bilan.getChiffreAffaires()));
        addTableRow(table, "Consultations payées",
            String.valueOf(bilan.getConsultationsPayees()));
        addTableRow(table, "Montant impayés",
            String.format("%.2f DH", bilan.getMontantImpayes()));

        document.add(table);
        document.close();
    }
}
```

Listing 3 – PDFExporter.java

7.5 Annexe E : Dépendances Maven (pom.xml)

```
<dependencies>
    <!-- MySQL Connector -->
    <dependency>
        <groupId>com.mysql</groupId>
        <artifactId>mysql-connector-j</artifactId>
        <version>8.2.0</version>
    </dependency>

    <!-- BCrypt pour le hachage des mots de passe -->
    <dependency>
        <groupId>org.mindrot</groupId>
        <artifactId>jbcrypt</artifactId>
        <version>0.4</version>
    </dependency>

    <!-- iTextPDF pour la génération de PDF -->
    <dependency>
        <groupId>com.itextpdf</groupId>
        <artifactId>itextpdf</artifactId>
        <version>5.5.13.3</version>
    </dependency>

    <!-- JFreeChart pour les graphiques -->
    <dependency>
        <groupId>org.jfree</groupId>
        <artifactId>jfreechart</artifactId>
        <version>1.5.4</version>
    </dependency>

    <!-- JUnit pour les tests -->
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.13.2</version>
        <scope>test</scope>
    </dependency>
</dependencies>
```

Listing 4 – Dépendances Maven