**BTS SIO SLAM 3 PPE : Gestion des rapports de visite GSB en Java**

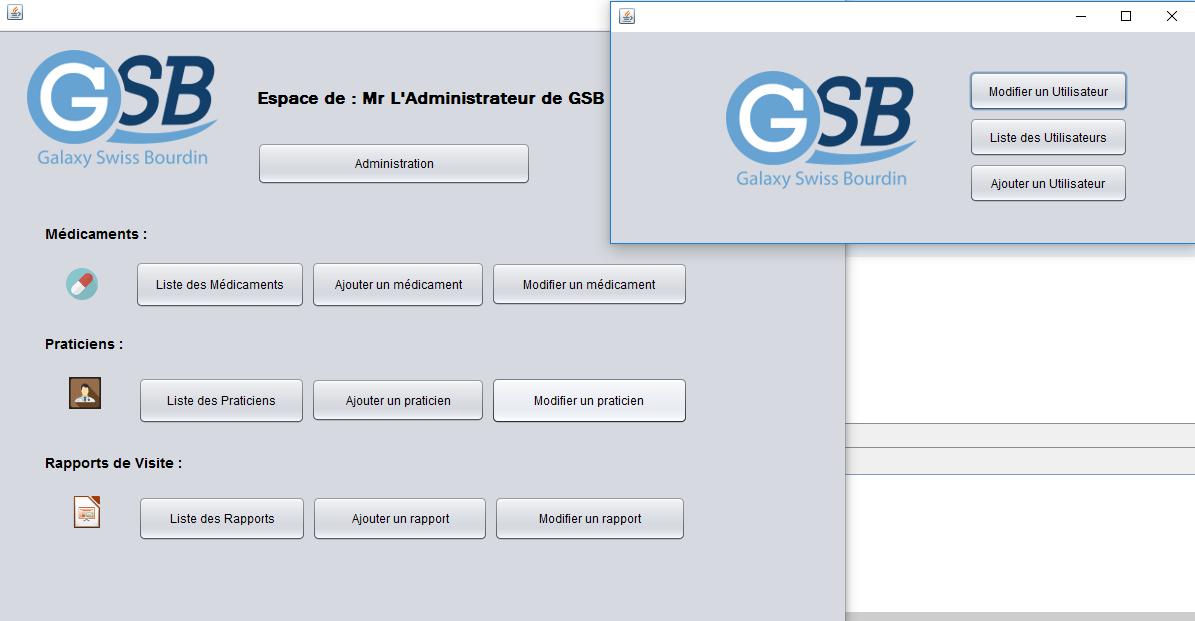
### CAHIER DES CHARGES

|  |  |
| --- | --- |
| **Information sur le projet** | |
| Groupe : | JADEAU Anne-Jeanne  * BERNATENE Gaël * BOUZAINE Emir * GAUDET Bryan |
| Nom du Projet : | Application de rapports de visite |
| Type de document : | Cahier des charges |
| Professeur référent : | Mme. Gaddouri |
| Etablissement : | UFIP Business School, Nice |

|  |
| --- |
| Diffusion du groupe |

|  |  |  |
| --- | --- | --- |
| Élève | Email | Rôle |
| Anne-Jeanne JADEAU | anne@jadeau.fr | Chef de Projet  Développement Modèle, Vue, Contrôleurs de la partie Utilisateur et Rôle |
| *Gaël BERNATENE* | gael.bernatene@gmail.com | Développement Modèle, Vue, Contrôleurs de la partie Authentification et Rapport de Visite |
| Emir BOUZAINE | emir.bouzaine@ufip.fr | Développement Modèle, Vue, Contrôleurs de la partie Visiteur |
| Bryan Gaudet | bryan.gaudet@ufip.fr | Développement Modèle, Vue, Contrôleurs de la partie Praticien |

### Sommaire



**I. Présentation et mise en situation du système**…………………………………………………………….3-4

1.1 Contexte…………………………………………………………………………………………………………………………………3

1.2 Définition du besoin……………………………………………………………………………………………………………….3

1.3 Forme de l’objet……………………………………………………………………………………………………………………..3

1.4 Diagramme de cas d’utilisation………………………………………………………………………………………………4

1.5 Diagramme de Modèle Conceptuel de Données……………………………………………………………………..4  
**II. Contraintes fonctionnelles et techniques**………………………………………………………………………5  
 2.1 Architecture……………………………………………………………………………………………………………………………5  
 2.2 Ergonomie………………………………………………………………………………………………………………………………5  
 2.3 Sécurité………………………………….……………………………………………………………………………………………….5

**III. Environnement technologique**………………………………………………………………........................6  
**IV. Les Fonctionnalités**……………………………………………………………….......................................7-9

**V. La base de données**………………………………………………………………...................................10-12  
 5.1 Création des tables………………………………………………………………………………………………………….10-11  
 5.2 MLD……………………………………………………………………………………………………………………………………..12

**I. Présentation et mise en situation du système**

1.1 Contexte

Le laboratoire Galaxy Swiss Bourdin (GSB) désire mettre à disposition des visiteurs médicaux une application Bureau permettant de centraliser les comptes-rendus de visite. Cette base d'information sera utilisée à des fins d'élaboration de la démarche de communication auprès des praticiens et donnera une vision individuelle et synthétique de l'activité de représentation.

1.2 Définition du besoin

Pour permettre une aide au renseignement des rapports, l'application fournira une description des produits du laboratoire, les coordonnées précises des praticiens et des informations détaillées les concernant.

L’application bureau offrira aux visiteurs médicaux les possibilités suivantes :

* S’authentifier avec un compte utilisateur qui détient le rôle visiteur médical attitré via un login et un mot de passe, afin d’accéder au menu de l’application lié à sa session.
* Accéder aux options du menu comme ceux des Médicaments, des Praticiens, des Visiteurs ou encore des Rapport de visites
* Créer, modifier, supprimer, visionner, afin de gérer les différents paramètres des entités selon sa volonté (médicament, praticien, visiteur médical, rapport de visite)

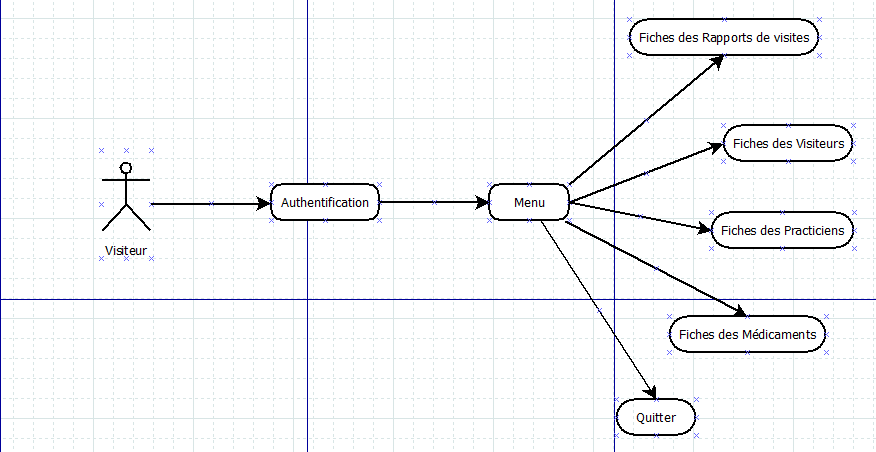
L’administrateur aura aussi accès à certaines fonctionnalités en fonction de son rôle :

* L’authentification avec un compte utilisateur qui détient le rôle admin attitré via un login et un mot de passe, afin d’accéder au menu de l’application lié à sa session.
* Accéder aux même options du menu comme ceux des Médicaments, des Praticiens ou encore des Rapport de visites
* Créer des comptes utilisateur aux rôles visiteur médical ou admin,

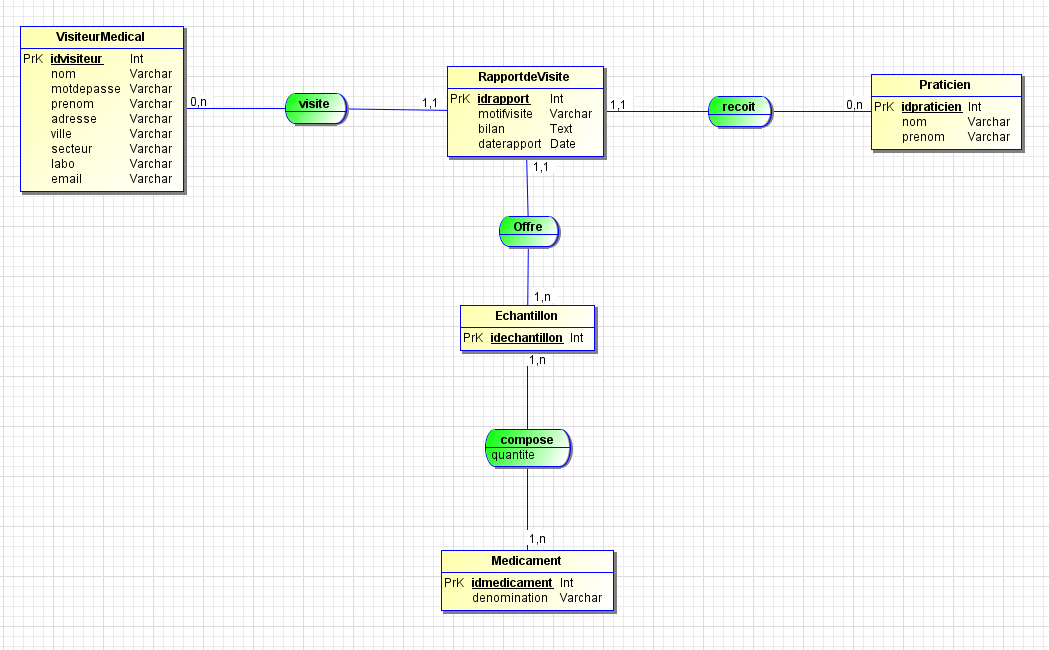
1.3 Forme de l'objet

L’application se fera sous forme d'un programme en Java sécurisé et authentifié, dont le design se composera de plusieurs fenêtres exécutables, créable grâce à la bibliothèque graphique Swing.

1.4 Diagramme de cas d’utilisation



1.5 MCD



**II. Contraintes fonctionnelles et techniques**

2.1 Architecture

L’application suit l’organisation d’un design pattern MVC (Modèle Vue Contrôleur).

**Modèle(1, 1) :** Celui-ci comprend les données et un certain nombre de méthodes pour les lire et les modifier. Le modèle n’a aucune connaissance de la façon dont les données sont présentées à l’utilisateur (la Vue). Le modèle peut toutefois enregistrer une ou plusieurs vues (une liste de dépendants) qui sont notifiées quand les données du modèle ont subi une modification

**Vue (1, N) :** fournit une représentation graphique de tout ou partie des données du modèle. Chaque vue obtient les données à afficher en en faisant la demande au modèle. Quand un utilisateur manipule une vue d’un modèle, la vue informe le contrôleur du changement désiré.

**Contrôleur (1, N) :** Le contrôleur interprète les actions de l’utilisateur et appelle les méthodes requises du modèle pour le mettre à jour. Le modèle informe alors toutes les vues qu’un changement est intervenu. Ces dernières se mettent à jour. Il peut aussi vérifier les actions de l’utilisateur et modifier la vue en conséquence si besoin.

Les vues sont aussi associées à des contrôleurs qui mettent à jour le modèle. Chaque vue est associée à un unique contrôleur.

2.2 Ergonomie

L’application s’inspire fortement d’une application Access existante en terme d'ergonomie et de couleurs, et elle est donc constituée d’un certain nombre d’écrans auxquels on accède à travers un écran de menu principal. Il n’y a pas de barre de menu, ni d’animation particulière.

2.3 Sécurité

Les mots de passe des utilisateurs seront stockés dans la base de données, ce qui induit le fait de

devoir les crypter afin que personne ne puisse les connaître.  A noter avant cela que chaque mot de passe crypté dans la base de données ne pourront être

décryptés, donc il n’y a pas moyen de récupération. Ainsi, nous vous prions de bien garder votre

mot de passe.  Pour se faire, nous avons décidé d’utiliser le salage du mot de passe, avant de le crypter pour

éviter tout risque de brute force dessus, ce qui augmentera sa sécurité.

**III. Environnement Technologique**

=> WampServer

-WampServer est une plate-forme de développement Web sous Windows pour des applications Web/logiciel dynamiques à l’aide du serveur Apache, du langage de scripts PHP et d’une base de données MySQL.

-Il possède également PHPMyAdmin pour paramétrer plus facilement les bases de données.

=> Apache

Apache est un logiciel permettant de créer un serveur HTTP, il est conçu pour prendre en charge de nombreux modules, lui donnant des fonctionnalités supplémentaires : interprétation des langages Perl, Php, Python, JAVA serveur proxy, réécriture d'URL, négociation de contenu, protocoles de communication...

=> MySQL

MySQL est un Système de Gestion de Base de Données (SGBD), qui sera un support de base de

données utilisé pour stocker les tables et les informations nécessaires au bon fonctionnement de l’application.

=> JAVA  JDK 8

- Langage de programmation orienté objet.

- Fonctionne automatiquement sous Windows, Solaris, Mac, Linux, etc..

=> SwingX

- C’est une [bibliothèque graphique](https://fr.wikipedia.org/wiki/Biblioth%C3%A8que_graphique) pour le [langage de programmation Java](https://fr.wikipedia.org/wiki/Java_(langage))

- Il utilise la bibliothèque AWT(Abstract Window Toolkit)

=>WebLAF

-C’est une bibliothèque basée sur Java qui permet de créer des applications en utilisant de multiples effets et des éléments.

- Elle comprend des barres de progression, des curseurs, un système de calendrier et de nombreux autres éléments qui peuvent être intégrés dans vos propres applications.

=>NetBeans 8.2

-NetBeans est un environnement de développement intégré (ou IDE) qui supporte plusieurs langages de programmation tel que  [C](https://fr.wikipedia.org/wiki/C_(langage)), le [C++](https://fr.wikipedia.org/wiki/C_plus_plus), Java, [le JavaScript](https://fr.wikipedia.org/wiki/JavaScript), [le XML](https://fr.wikipedia.org/wiki/Extensible_Markup_Language), [le Groovy](https://fr.wikipedia.org/wiki/Groovy_(langage)), [le PHP](https://fr.wikipedia.org/wiki/PHP) et [le HTML](https://fr.wikipedia.org/wiki/Hypertext_Markup_Language) et qui permet donc de développer diverses applications.

-Il est compatible avec [Windows](https://fr.wikipedia.org/wiki/Microsoft_Windows), [Linux](https://fr.wikipedia.org/wiki/Linux), [Solaris](https://fr.wikipedia.org/wiki/Solaris_(informatique)) , [Mac OS X](https://fr.wikipedia.org/wiki/Mac_OS_X) ou avec une version indépendante des systèmes d'exploitation qui dispose d’une machine virtuelle Java.

-Un environnement Java Development Kit ([JDK](https://fr.wikipedia.org/wiki/JDK)) est requis pour les développements en Java.

**IV. Les Fonctionnalités**

|  |
| --- |
| **Fonctionnalité 1 : Authentification d’un compte utilisateur (rôle : visiteur médical ou administrateur)** |

|  |  |
| --- | --- |
| Fonctionnement | Permet de se connecter grâce à un compte déjà existant et d’accéder aux options du menu proposées selon le rôle de l’utilisateur (visiteur médical ou admin) |
| Description | Affiche une fenêtre de connexion (Authentification.java), qui propose de saisir les champs Login et Mot de passe afin de s’authentifier. Le contrôleur MysqlConnection.java va établir la connexion avec la base de données donc  si les valeurs saisies sont exactes avec celles enregistrées dans celle-ci, alors l’application va ouvrir l’accès à une nouvelle jFrame MenuVisiteur.java |
| Localisation/Dépendances | Modèle : Utilisateur.java  Vue : Authentification .java  Contrôleur : UtilisateurController.java, MysqlConnection.java |
| Etat | Fonctionnel |

|  |
| --- |
| **Fonctionnalité 2 : Gestion des médicaments (rôle : visiteur médical ou administrateur)** |

|  |  |
| --- | --- |
| Fonctionnement | Permet d’ajouter un médicament, de le modifié et d’afficher une liste des médicament |
| Description | Apparition d’une jFrame MedicamentView.java, qui présente une liste avec toutes les informations d’un médicament. La jFrame MedicamentAdd.java affiche tous les champs textes nécessaires pour informer un médicament : Dénomination, Laboratoire, Quantité. L’exécution de la requête d’insertion se fait via un bouton.  Et une troisième jFrame MedicamentEdit.java permet de modifier antérieurement saisie en sélectionnant grâce un panneau déroulant le médicament souhaité. |
| Localisation/Dépendances | Modèle : Medicament.java  Vue : MedicamentView.java, MedicamentAdd.java, MedicamentEdit.java  Contrôleur : MedicamentController.java, MysqlConnection.java |
| Etat | Fonctionnel |

|  |
| --- |
| **Fonctionnalité 3 : Gestion des praticiens (rôle : visiteur médical)** |

|  |  |
| --- | --- |
| Fonctionnement | Permet d’afficher la liste de tous les praticiens et d’en ajouter, modifier et supprimer. |
| Description | La jFrame PraticienView.java affiche un tableau de tous les praticiens pouvant être trié par un menu déroulant et inclut une option pour souscrire et retirer tous les praticiens favoris du visiteur.  La jFrame PraticienAdd.java autorise seulement un visiteur à ajouter un praticien en fournissant toutes informations nécessaire d’un profil praticien.  La jFrame PraticienModif.java permet la modification ou la suppression des information liée à un praticien |
| Localisation/Dépendances | Modèle : Praticien.java  Vue : PraticienView.java, PraticienAdd.java, PraticienModif.java  Contrôleur : PraticienController.java, VisiteurController.java, MysqlConnection.java |
| Etat | Fonctionnel |

|  |
| --- |
| **Fonctionnalité 4 : Gestion rapports de visite (rôle : visiteur médical et administrateur)** |

|  |  |
| --- | --- |
| Fonctionnement | Permet d’afficher, d’établir un rapport de visite |
| Description | La fenêtre RapportAdd.java donne la possibilité à l’utilisateur d’entrer les valeurs nécessaire pour enregistrer un rapport de visite en associant au préalable le praticien et le médicament : date de rapport, motif de visite, bilan.  L’autre fenêtre RapportVisiteList.java affiche une liste de tous les rapports de visites du logiciel |
| Localisation/Dépendances | Modèle : Rapportdevisite.java, Praticien.java, Echantillon.java, Visiteurmedical.java  Vue : RapportAdd.java, RapportVisiteListe.java  Contrôleur : RapportController.java, PraticienController.java, VisiteurController.java, EchantillonController.java |
| Etat | Fonctionnel |

|  |
| --- |
| **Fonctionnalité 5 : Gestion de visiteurs médicaux (rôle : administrateur)** |

|  |  |
| --- | --- |
| Fonctionnement | Permet de créer, modifier un compte d’utilisateur de visiteur médical et de voir une liste de tous les utilisateurs enregistrés dans l’application |
| Description | Une fois authentifié en tant qu’Admin, une fenêtre modifié Authentification.java autorise via un bouton (lance UtilisateurAdd.java) la création d’un compte visiteur médical en fournissant des informations : identifiant, mot de passe, email, nom, prénom, adresse, ville, secteur, laboratoire.  Un autre bouton (lance UtilisateurModif.java) permet la modification ou la suppression d’un compte utilisateur.  La fenêtre UtilisateurList.java affiche la liste de tous les utilisateurs enregistrés. |
| Localisation/Dépendances | Modèle : Role.java, Utilisateur.java, Visiteurmedical.java  Vue : Authentification.java, UtilisateurAdd.java, UtilisateurList.java  Contrôleur : RoleController.java, UtilisateurController.java, MysqlConnection.java |
| Etat | Fonctionnel |

**IV. La base de données**

5.1 Création des tables

La table : compose

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Attributs** | **Null** | **Défaut** | **Extra** | **Relié à** |
| quantite | int(11) |  | Oui | Null |  |  |
| id\_medicament | int(11) |  | Non | Aucune |  | medicament |
| id\_echantillon | int(11) |  | Non | Aucune |  | echantillon |

La table : echantillon

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Attributs** | **Null** | **Défaut** | **Extra** | **Relié à** |
| id\_echantillon | int(11) |  | Non | Aucune |  |  |
| ref\_echantillon | varchar(55) |  | Oui | Null |  |  |
| id\_rapport | int(11) |  | Oui | Null |  | rapportdevisite |

La table : medicament

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Attributs** | **Null** | **Défaut** | **Extra** | **Relié à** |
| id\_medicament | int(11) |  | Non | Aucune | auto\_increment |  |
| denomination | varchar(255) |  | Oui | Null |  |  |
| laboratoire | varchar(55) |  | Non | Aucune |  |  |
| quantiteStock | int(255) |  | Non | Aucune |  |  |

La table : praticien

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Attributs** | **Null** | **Défaut** | **Extra** | **Relié à** |
| id\_praticien | int(11) |  | Non | Aucune | auto\_increment |  |
| nom | varchar(255) |  | Oui | Null |  |  |
| prenom | varchar(25) |  | Oui | Null |  |  |
| adresse | varchar(55) |  | Oui | Null |  |  |
| ville | varchar(25) |  | Oui | Null |  |  |
| code\_postal | varchar(25) |  | Oui | Null |  |  |
| telephone | varchar(25) |  | Oui | Null |  |  |
| secteur | text |  | Oui | Null |  |  |

La table : rapportdevisite

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Attributs** | **Null** | **Défaut** | **Extra** | **Relié à** |
| id\_rapport | int(11) |  | Non | Aucune | auto\_increment |  |
| motifvisite | varchar(255) |  | Oui | Null |  |  |
| bilan | text |  | Oui | Null |  |  |
| datevisite | date |  | Oui | Null |  |  |
| id\_visiteur | int(11) |  | Oui | Null |  | visiteur |
| id\_praticien | int(11) |  | Oui | Null |  | praticien |
| id\_echantillon | int(11) |  | Oui | Null |  | echantillon |

La table : role

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Attributs** | **Null** | **Défaut** | **Extra** | **Relié à** |
| id\_role | int(11) |  | Non | Aucune | auto\_increment |  |
| nom\_role | varchar(25) |  | Oui | Null |  |  |

La table : utilisateur

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Attributs** | **Null** | **Défaut** | **Extra** | **Relié à** |
| id\_utilisateur | int(11) |  | Non | Aucune | auto\_increment |  |
| login | varchar(25) |  | Oui | Null |  |  |
| password | varchar(255) |  | Oui | Null |  |  |
| email | varchar(25) |  | Oui | Null |  |  |
| id\_role | int(11) |  | Oui | Null |  | role |

La table : visiteurmedical

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Colonne** | **Type** | **Attributs** | **Null** | **Défaut** | **Extra** | **Relié à** |
| id\_visiteur | int(11) |  | Non | Aucune | auto\_increment |  |
| nom | varchar(255) |  | Oui | Null |  |  |
| prenom | varchar(25) |  | Oui | Null |  |  |
| adresse | varchar(25) |  | Oui | Null |  |  |
| ville | varchar(255) |  | Oui | Null |  |  |
| secteur | varchar(255) |  | Oui | Null |  |  |
| labo | varchar(25) |  | Oui | Null |  |  |
| id\_utilisateur | int(11) |  | Oui | Null |  | utilisateur |

5.2 MLD

