Documentation technique (AP4 : GSB MEDECINS)

Nom : Diatta

Prénom : Sidy James

Classe : BTS SIO2 B

A) Présentation du contexte

Le laboratoire Galaxy Swiss Bourdin (GSB) est amené dans ses différentes activités à contacter les médecins installés, par exemple les visiteurs médicaux du laboratoire se déplacent afin de présenter les nouveaux produits pharmaceutiques.

Une base de données de médecins est utilisée dans ces différents processus. Cette base de données évolue fréquemment, ceci à cause de départ à la retraite de médecins ou d'installation de nouveaux praticiens. Les applications doivent intégrer ces évolutions des données.

Il a été décidé de confier à une société de services informatiques la responsabilité de développer deux applications donnant accès aux informations de la base de données.

1. Partie 1 : la consultation sur une application de type client lourd, des médecins.

2. Partie 2 : la consultation sur un téléphone (ou une tablette) Android des médecins.

Un certain nombre de contraintes ont été fixées :

- Le développement sera réalisé dans un langage orienté objet (JSP). L'architecture applicative sera en MVC.

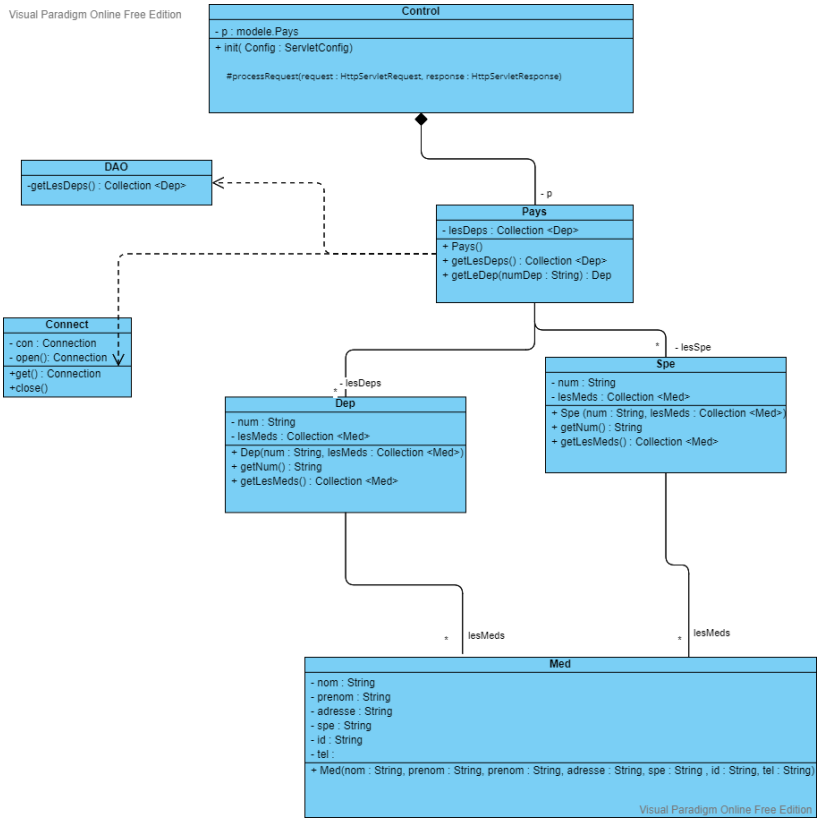
- L'application utilisera la base de données PostgreSQL fournie.

- Les informations exposées indiqueront simplement les nom, prénom, adresse, téléphone et spécialité médicale complémentaire des médecins généralistes.

B) Application réalisée

Cette application réalisée permet au laboratoire GSB de pouvoir rechercher des médecins par département, par spécialité avec leurs coordonnées mais aussi de pouvoir effectuer une recherche à partir du début de nom des médecins.

C) Diagramme de classe



D) MVC

1- Modele

Connexion à la base de données en singleton dans Connect.java

```
package model;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class Connect { //singleton qui garantit qu'il n'y aura qu'une seule instance de cette classe.

    private static Connection cnx = null;

    private static Connection open() {
        try {
            Class.forName("org.postgresql.Driver");
        } catch (ClassNotFoundException e) {
            System.out.println("Pilote mal installÃ©... " + e);
        }
        try {
            cnx = DriverManager.getConnection("jdbc:postgresql://localhost:5433/ap4", "postgres", "[redacted]");
        } catch (SQLException e) {
            System.out.println("Erreur SQL : " + e);
        }
        return cnx;
    }
//Méthode qui va nous retourner notre instance et la créer si elle n'existe pas
    public static Connection get() {
        if (cnx == null) {
            cnx = Connect.open();
        }
        return cnx;
    }

    public static void close() {
        try {
            cnx.close();
        } catch (SQLException e) {
            System.out.println("Erreur SQL : " + e);
```

Ceci nous permet d'avoir une seule instance (objet avec un comportement et un état, tous deux définis par la classe) dans le programme et y avoir accès partout dans l'application.

Dans la page DAO.java, nous avons la création de collection avec l'utilisation de TreeSet (avec un temps d'accès beaucoup plus rapide) pour les départements, spécialités et médecins.

```java
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Collection;
import java.util.TreeSet;
import java.util.logging.Level;
import java.util.logging.Logger;

public class DAO {
//Utilisation de treeset qui permet un temps d'accès rapide
    public static Collection<Dep> getLesDeps() {
        Collection<Dep> d = new TreeSet<Dep>();
        try {

            Connection cnx = Connect.get();

            //Requête
            java.sql.Statement reqD = (java.sql.Statement) cnx.createStatement();
            ResultSet rsD = reqD.executeQuery("Select DISTINCT departement from medecin");
            // Parcours de la collection des departements
            while (rsD.next()) {
                String dep = rsD.getString("departement");
                d.add(new Dep(dep));
            }
            rsD.close();
            //fermeture de la Base de Données
            reqD.close();
        } catch (SQLException ex) {
            Logger.getLogger(Pays.class.getName()).log(Level.SEVERE, null, ex);
        }
        return d;
    }
```

```java
public static Collection<Spe> getLesSpes() {
    Collection<Spe> s = new TreeSet<Spe>();
    try {
        Connection cnx = Connect.get();

        java.sql.Statement reqS = (java.sql.Statement) cnx.createStatement();
        ResultSet rsS = reqS.executeQuery("Select DISTINCT specialitecomplementaire from medecin where specialitecomplementai
        // Parcours de la collection des specialités
        while (rsS.next()) {
            String spe = rsS.getString("specialitecomplementaire");
            s.add(new Spe(spe));
        }
        rsS.close();
        //fermeture de la Base de Données
        reqS.close();
    } catch (SQLException ex) {
        Logger.getLogger(Pays.class.getName()).log(Level.SEVERE, null, ex);
    }
    return s;
}
```

```
    public static Collection<Med> getLesMeds() {
        Collection<Med> m = new TreeSet<Med>();
        try {
            Connection cnx = Connect.get();

            Statement reqM = cnx.createStatement();
            ResultSet rsM = reqM.executeQuery("select * from medecin");
            while (rsM.next()) {
                String id = rsM.getString("id");
                String nom = rsM.getString("nom");
                String prenom = rsM.getString("prenom");
                String adresse = rsM.getString("adresse");
                String tel = rsM.getString("tel");
                String spe = rsM.getString("specialitecomplementaire");
                String dep = rsM.getString("departement");
                m.add(new Med(nom, prenom, adresse, tel, spe, dep, id));
            }
            rsM.close();
            //fermeture de la Base de Données
            reqM.close();
        } catch (SQLException ex) {
            Logger.getLogger(Pays.class.getName()).log(Level.SEVERE, null, ex);
        }
        return m;
    }
}
```

Dans le fichier Dep.java, nous avons une classe Dep avec implements Comparable<Dep> qui nous permet de faire comparaison entre les numéros de départements avec une création de collection pour les médecins.

```
package model;

import java.util.Collection;
import java.util.TreeSet;
import java.util.HashSet;

// La class Dep implémente Comparable qui nous permet de faire une comparaison dans la collection des départements
public class Dep implements Comparable<Dep> {

    private String num;
    private Collection<Med> lesMeds = new TreeSet<Med>();

    public Dep(String num) {
        this.num = num;
    }

    public String getNum() {
        return num;
    }

    public Collection<Med> getLesMeds() {
        return lesMeds;
    }

    public void addUnMed(Med unMed) {
        lesMeds.add(unMed);
    }

    public Collection getLesMedsR(String med) {
        Collection<Med> medSearch = new HashSet<Med>();
        for (Med unMed : lesMeds) {
            if (unMed.getNom().startsWith(firstLetterCaps(med))) {
                medSearch.add(unMed);
            }
```

```java
public Collection getLesMedsR(String med) {
    Collection<Med> medSearch = new HashSet<Med>();
    for (Med unMed : lesMeds) {
        if (unMed.getNom().startsWith(firstLetterCaps(med))) {
            medSearch.add(unMed);
        }
    }
    return medSearch;
}

public String firstLetterCaps(String data) {
    String firstLetter = data.substring(0, 1).toUpperCase();
    String restLetters = data.substring(1).toLowerCase();
    return firstLetter + restLetters;
}

@Override
// Comparaison de l'objet Dep en fonction de son numero
public int compareTo(Dep d) {
    return num.compareTo(d.num);
}

    public void setNum(String num) {
            this.num = num;
    }

    public void setLesMeds(Collection<Med> lesMeds) {
            this.lesMeds = lesMeds;
    }
```

Le même processus est répété pour les médecins dans Med.java mais dans ce cas, on compare d'abord le nom des médecins. S'ils ont le même nom alors nous pouvons comparer les prénoms sinon nous comparons les noms.

```java
public class Med implements Comparable<Med> {

    private String nom;
    private String prenom;
    private String adresse;
    private String tel;
    private String spe;
    private String dep;
    private String id;

    public Med(String nom, String prenom, String adresse, String tel, String spe, String dep, String id) {
        this.nom = nom;
        this.prenom = prenom;
        this.adresse = adresse;
        this.tel = tel;
        this.spe = spe;
        this.dep = dep;
        this.id = id;
    }

    public String getAdresse() {
        return adresse;
    }

    public String getNom() {
        return nom;
    }

    public String getPrenom() {
        return prenom;
    }

    public String getSpe() {
        return spe;
```

```java
public String getSpe() {
    return spe;
}

public String getDep() {
    return dep;
}

public String getTel() {
    return tel;
}

@Override
public int compareTo(Med m) {
    if (nom.compareTo(m.nom) == 0) { //s'ils ont le même nom
        return prenom.compareTo(m.prenom); //comparer les prenoms
    } else {
        return nom.compareTo(m.nom); //sinon comparer les noms
    }
}

    @Override
    public String toString() {
            return "Med [nom=" + nom + ", prenom=" + prenom + ", adresse=" + adresse + ", tel=" + tel + ", spe=" + spe
                    + ", dep=" + dep + ", id=" + id + "]";
```

Dans pays.java nous retrouvons la collection des départements et spécialités comme dans le diagramme de classe ci- dessus.

```java
public class Pays {

    private Collection<Dep> lesDeps;
    private Collection<Spe> lesSpes;


    public Pays() {
        lesDeps = DAO.getLesDeps();
        lesSpes = DAO.getLesSpes();
        this.assocMedecins(DAO.getLesMeds());
    }

    public Collection<Dep> getLesDeps() {
        return lesDeps;
    }

    public Collection<Spe> getLesSpes() {
        return lesSpes;
    }

    private void assocMedecins(Collection<Med> lesMeds) {
        for (Med unMed : lesMeds) {
            getLeDep(unMed.getDep()).addUnMed(unMed);
            Spe uneSpe = getLaSpe(unMed.getSpe());
            if (uneSpe != null) {
                uneSpe.addUnMed(unMed);
            }
        }
    }

    public Dep getLeDep(String numDep) {
        for (Dep unDep : lesDeps) {
            if (unDep.getNum().equals(numDep)) {
                return unDep;
            }
```

2- Controleur

```java
public class Control extends HttpServlet {

    private Pays gsb;

    public void init(ServletConfig config) throws ServletException {
        super.init(config);
        gsb = new Pays(); // instancie les objets utiles
    }

    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        String page;
        String action = request.getParameter("action");

        if ("listeMedecinsDep".equals(action)) {
            String choixDep = request.getParameter("choixDep");
            if (choixDep == null || choixDep.equals("-1")) {
                Collection<Dep> d = gsb.getLesDeps();
                request.setAttribute("listDeps", d);
                page = "selectDep.jsp";
            } else {
                Collection<Med> m = gsb.getLeDep(choixDep).getLesMeds();
                request.setAttribute("leDep", gsb.getLeDep(choixDep));
                request.setAttribute("listMeds", m);
                page = "listMedDep.jsp";
            }

        } else if ("listeMedecinsNom".equals(action)) {
            String nomMed = request.getParameter("nomMed");
            if (nomMed == null) {
                page = "searchMed.jsp";
            } else {
```

Ici, notre classe Control hérite de HttpServlet qui reçoit une requête du client, effectue des traitements et renvoie le résultat.

Nous chargeons les médecins du département choisi et nous retournons la page listMedDep.jsp.

```java
} else if ("listeMedecinsNom".equals(action)) {
    String nomMed = request.getParameter("nomMed");
    if (nomMed == null) {
        page = "searchMed.jsp";
    } else {
        Collection<Med> m = new TreeSet<Med>();
        for (Dep unDep : gsb.getLesDeps()) {
            Collection medR = unDep.getLesMedsR(nomMed);
            m.addAll(medR);
        }
        request.setAttribute("listMeds", m);
        request.setAttribute("nomR", nomMed);
        page = "listMedNom.jsp";
    }

} else if ("listeMedecinsSpe".equals(action)) {
    String choixSpe = request.getParameter("choixSpe");
    if (choixSpe == null || choixSpe.equals("-1")) {
        Collection<Spe> s = gsb.getLesSpes();
        request.setAttribute("listSpes", s);
        page = "selectSpe.jsp";
    } else {
        Collection<Med> m = gsb.getLaSpe(choixSpe).getLesMeds();
        request.setAttribute("laSpe", gsb.getLaSpe(choixSpe));
        request.setAttribute("listMeds", m);
        page = "listMedSpe.jsp";
    }
} else {
    page = "index.jsp";
}
```

3- Vue

En-tête de la page d'accueil :

```jsp
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE>
<html lang="fr">
  <head>
    <title> Laboratoire Galaxy-Swiss Bourdin</title>
    <meta http-equiv="content-type" content="text/html; charset=utf-8" />
    <link href="./css/style.css" rel="stylesheet" type="text/css" />
    <link rel="shortcut icon" type="image/x-icon" href="./img/favicon.ico" />
  </head>
  <body>
    <div id="page">
      <div id="entete">
        <img src="./img/logo.jpg" id="logoGSB" alt="Laboratoire Galaxy-Swiss Bourdin" title="Laboratoire Galaxy-Swiss Bourdi
        <h1>Recherche des médecins</h1>
      </div>
        <jsp:include page="sommaire.jsp"/>
```

```jsp
<!-- Utilisation des tags pour iterer sur la collection et empêcher d'utiliser du code java directement -->
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<jsp:include page="entete.jsp"/>
<div id="contenu">
    <!-- Retour à la page selectDep.jsp -->
    <a href="Control?action=listeMedecinsDep">Retour à la selection </a>
    <br />
    <table class="listeLegere">
        <!-- Affichage de la collection des médecins sous forme de tableau -->
        <caption>Medecins pour le departement n° ${leDep.num}</caption>
        <tr>
            <th>Nom</th>
            <th>Prenom</th>
            <th>Adresse</th>
            <th>Specialition</th>
            <th>Telephone</th>
        </tr>
        <c:forEach var="med" items="${listMeds}">
            <tr>
                <td>${med.nom}</td>
                <td>${med.prenom}</td>
                <td>${med.adresse}</td>
                <td>${med.spe}</td>
                <td>${med.tel}</td>
            </tr>
        </c:forEach>
    </table>
</div>
<jsp:include page="pied.jsp"/>
```

Dans listMedDep.jsp, nous interagissons avec le controleur avec items="${listMeds}" pour récupérer les coordonnées des médecins et l'afficher sous forme de tableau.

```jsp
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<jsp:include page="entete.jsp"/>
<div id="contenu">
    <h3>Résultat(s) pour "<i>${nomR}</i>"</h3>
    <table class="listeLegere">
        <caption>Liste des médecins par nom</caption>
        <tr>
            <th>Nom</th>
            <th>Prénom</th>
            <th>Adresse</th>
            <th>Spécialité</th>
            <th>Téléphone</th>
        </tr>
        <c:forEach var="med" items="${listMeds}">
            <tr>
                <td>${med.nom}</td>
                <td>${med.prenom}</td>
                <td>${med.adresse}</td>
                <td>${med.spe}</td>
                <td>${med.tel}</td>
            </tr>
        </c:forEach>
    </table>
    <br />
    <a href="Control?action=listeMedecinsNom">Retour à la recherche</a>
</div>
<jsp:include page="pied.jsp"/>
```

La page listMedNom.jsp renvoie les coordonnées du médecin recherché sous forme de tableau. Le bouton retour à la recherche permet de retourner à la page de recherche qui est ci-dessous :

```jsp
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<jsp:include page="entete.jsp"/>
<div id="contenu">
    <h2>Recherche des médecins par nom</h2>
    <form action="Control?action=listeMedecinsNom" method="post">
        <div class="corpsForm">
            <p>
                <label for="nomMed">Nom du médecin : </label>
                <input type="text" name="nomMed">
            </p>
        </div>
        <div class="piedForm">
            <p>
                <input id="ok" type="submit" value="Valider" size="20" />
                <input id="annuler" type="reset" value="Effacer" size="20" />
            </p>
        </div>

    </form>
    <jsp:include page="pied.jsp"/>
```

```jsp
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<jsp:include page="entete.jsp"/>
<div id="contenu">
    <a href="Control?action=listeMedecinsSpe">Retour à la sélection</a>
    <br />
    <table class="listeLegere">
        <caption>Médecins pour la spécialité "<i>${laSpe.libelle}</i>"</caption>
        <tr>
            <th>Nom</th>
            <th>Prénom</th>
            <th>Adresse</th>
            <th>Spécialité</th>
            <th>Téléphone</th>
        </tr>
        <c:forEach var="med" items="${listMeds}">
            <tr>
                <td>${med.nom}</td>
                <td>${med.prenom}</td>
                <td>${med.adresse}</td>
                <td>${med.spe}</td>
                <td>${med.tel}</td>
            </tr>
        </c:forEach>
    </table>
</div>
<jsp:include page="pied.jsp"/>
```

Dans la page listMedSpe.jsp, nous affichons les coordonnées des médecins selon la spécialité choisie.

```jsp
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<jsp:include page="entete.jsp"/>
<div id="contenu">
    <h2>Recherche des médecins par département</h2>
    <h3>Département à sélectionner : </h3>
    <form action="Control?action=listeMedecinsDep" method="post">
        <div class="corpsForm">
            <p>
                <label for="lstDeps" accesskey="n">Département : </label>
                <select id="lstDeps" name="choixDep">
                    <option value="-1">Choisir un département...</option>
                    <c:forEach var="dep" items="${listDeps}">
                        <option value="${dep.num}">${dep.num}</option>
                    </c:forEach>
                </select>
            </p>
        </div>
        <div class="piedForm">
            <p>
                <input id="ok" type="submit" value="Valider" size="20" />
                <input id="annuler" type="reset" value="Effacer" size="20" />
            </p>
        </div>

    </form>
<jsp:include page="pied.jsp"/>
```

La page selectDep.jsp récupère les départements avec ${listDeps} et les affiche sous forme d'une liste de même que dans la page selectSpe.jsp avec ${listSpes} ci-dessous :

```jsp
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<jsp:include page="entete.jsp"/>
<div id="contenu">
    <h2>Recherche des médecins par spécialité</h2>
    <h3>Spécialité à sélectionner : </h3>
    <form action="Control?action=listeMedecinsSpe" method="post">
        <div class="corpsForm">
            <p>
                <label for="lstSpes" accesskey="n">Spécialité : </label>
                <select id="lstSpes" name="choixSpe">
                    <option value="-1">Choisir une spécialité...</option>
                    <c:forEach var="spe" items="${listSpes}">
                        <option value="${spe.libelle}">${spe.libelle}</option>
                    </c:forEach>
                </select>
            </p>
        </div>
        <div class="piedForm">
            <p>
                <input id="ok" type="submit" value="Valider" size="20" />
                <input id="annuler" type="reset" value="Effacer" size="20" />
            </p>
        </div>

    </form>
<jsp:include page="pied.jsp"/>
```