

# RAPPORT DE PROJET

## Sujet: Smart Bracelet Médical

Préparé par:

Yasmine OUZZINE

Filière:

Génie Logiciel et Digitalisation

## Remerciements

Nous tenons à exprimer notre profonde gratitude au Professeur Hicham MEDROMI pour son dévouement exceptionnel tout au long du cours de Systèmes Distribués et des séances pratiques. Sa passion pour l'enseignement et ses efforts pour rendre le contenu accessible et captivant ont grandement amélioré notre expérience d'apprentissage. Les séances pratiques, dirigées par le Professeur Medromi, ont été particulièrement enrichissantes. Sa capacité à expliquer des concepts complexes de manière claire et concise a facilité notre compréhension des principes fondamentaux des systèmes distribués. Son soutien continu, sa disponibilité pour répondre à nos questions et ses encouragements ont créé un environnement d'apprentissage propice et motivant. De plus, nous le remercions sincèrement de nous avoir confié le sujet de projet "Smart Bracelet Médical". Ce projet stimulant et pertinent nous a donné l'occasion d'appliquer les connaissances acquises en classe et de développer des compétences pratiques en conception de systèmes logiciels. En conclusion, nous sommes reconnaissantes envers le Professeur Hicham MEDROMI pour son dévouement exceptionnel à l'enseignement et pour sa contribution significative à notre parcours académique. Ces moments d'apprentissage resteront inoubliables et serviront de base solide pour nos projets et réalisations futurs.

# INTRODUCTION

La réalisation d'un Smart Bracelet Médical représente un défi stimulant et captivant auquel nous avons consacré une période définie dans le cadre de notre programme académique. Le projet a été planifié et exécuté selon un calendrier précis, reflétant l'équilibre délicat entre l'ambition d'explorer de nouvelles dimensions de la programmation Java et le besoin de respecter des contraintes de temps imposées. La décision d'utiliser une Application Java plutôt que Java EE (JEE) pour ce projet était délibérée et stratégique. Ayant déjà travaillé sur un projet de commerce électronique utilisant Java EE, nous avons identifié l'opportunité d'approfondir nos compétences en Java dans un contexte différent. Le choix de l'Application Java est motivé par notre désir de diversifier nos connaissances et compétences, explorant une facette de Java qui avait été relativement moins explorée dans notre programme académique. La décision de se concentrer sur l'Application Java découle de la conviction que ce projet offre une opportunité unique d'approfondir notre compréhension de Java dans des domaines spécifiques. Cette approche reflète notre engagement à élargir notre ensemble de compétences et à maximiser l'expérience d'apprentissage offerte par ce projet.

# Chapitre I : Projet

## 1- Sujet du Projet:

**Titre :** G-Bracelet: Bracelet Médical Connecté

**Description du Projet :** Le G-Bracelet est un dispositif médical intelligent conçu pour surveiller en temps réel les paramètres vitaux tels que la température corporelle et la pression artérielle. Il permet une détection précoce des variations significatives, une localisation en cas d'urgence, une connectivité pour la télémédecine et une gestion intelligente de l'énergie.

### **Application Mobile :**

#### **• Pour le Patient :**

- Création de compte : Requiert ID du bracelet, nom, prénom, téléphone, mot de passe.
- Connexion : Accès via ID du bracelet et mot de passe.
- Visualisation en temps réel : Affichage des paramètres vitaux.
- Historique des mesures : Consultation par intervalle de temps.
- Indicateur de normalité : Changement de couleur selon les seuils.
- Acceptation de demandes de médecins : Autorisation pour le suivi.
- Ajout de proche : Pour suivi limité à 1 personne.
- Réception de commentaires de médecins : Diagnostics et conseils des médecins.

#### **• Pour le Médecin :**

- Création de compte : Nom, prénom, numéro de téléphone, mot de passe.
- Visualisation de la liste des patients : Accès à tous les patients.
- Gestion des patients : Ajout ou suppression de patients.
- Visualisation des paramètres vitaux : Consultation en temps réel.
- Personnalisation des seuils : Réglage des seuils de normalité.
- Diagnostic : Commentaires et diagnostics basés sur les données.

#### **• Pour le Proche :**

- Création de compte : Nom, prénom, numéro de téléphone, mot de passe.
- Visualisation des données : Accès aux données de santé du patient.
- Notifications : Alertes en cas de seuils dépassés.
- Réception de commentaires de médecins : Consultation des diagnostics et commentaires.
- Visualisation de la localisation : Suivi de la localisation du patient.

## 2- Environnement Utilisé:

Dans le cadre de notre projet " Implémentation d'un Système de Smart Bracelet Médical ", nous avons établi un environnement technologique cohérent, intégrant plusieurs outils pour répondre efficacement à nos besoins de développement. Voici un aperçu des principaux composants de cet environnement :



**JAVA: Java** est un langage de programmation polyvalent et orienté objet connu pour sa portabilité et son indépendance de plateforme. Développé par Sun Microsystems, il fournit un cadre robuste pour la construction d'applications évolutives. Les principales caractéristiques de Java incluent sa simplicité, sa sécurité et sa capacité à s'exécuter sur différentes plateformes sans recompilation, ce qui en fait un choix populaire pour divers projets de développement de logiciels.



**NetBeans IDE 19:** NetBeans est un Environnement de Développement Intégré (IDE) qui simplifie et accélère le processus de développement logiciel. Doté d'outils complets et prenant en charge plusieurs langages de programmation, il rationalise la codification, le débogage et la gestion de projet.

Avec son interface conviviale, l'IDE NetBeans est un outil puissant pour la construction d'applications diverses.



### **Système de Gestion de Base de Données (SGBD) -**

**MySQL via PHPMyAdmin :** Pour la gestion des données liées aux dossiers des patients, nous avons intégré MySQL en utilisant l'interface conviviale de PhpMyAdmin. Cette combinaison nous permet de concevoir et d'administrer notre base de données efficacement.

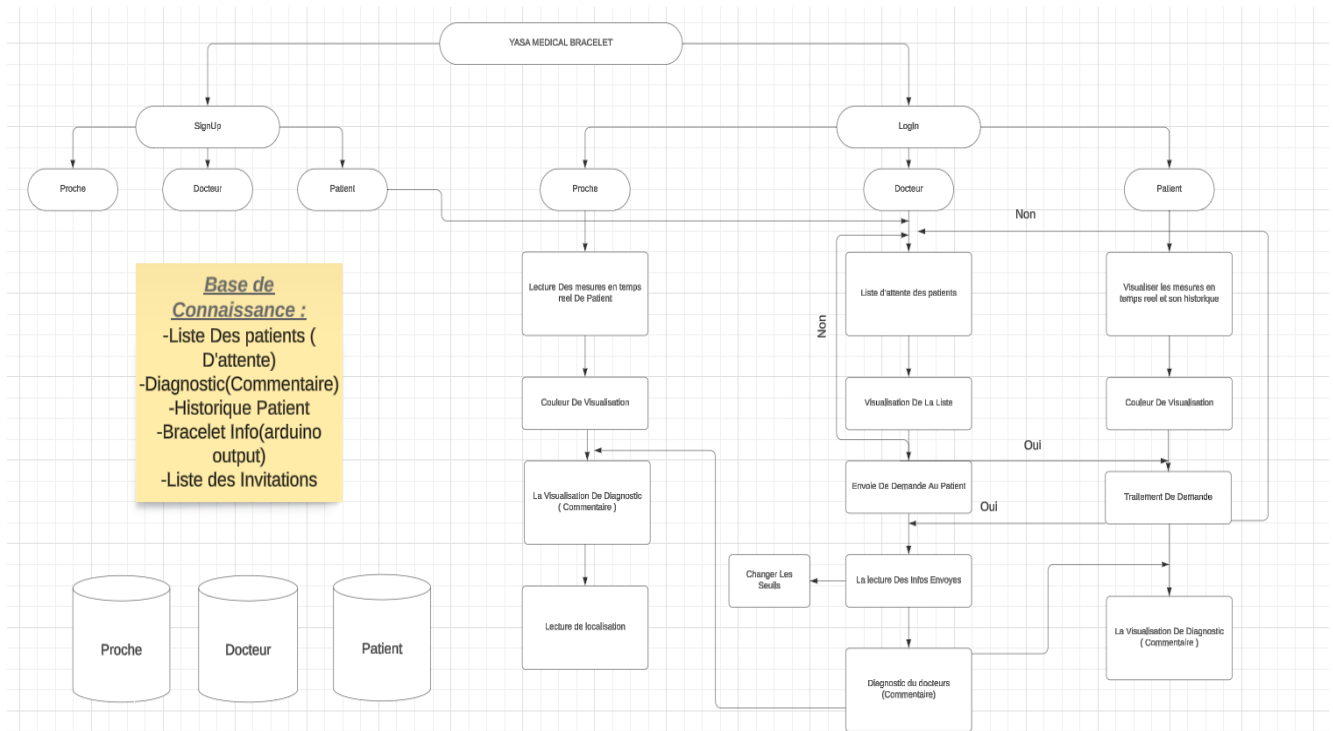


### **Environnement de Développement Local - XAMPP :**

XAMPP est un ensemble de logiciels open-source et multiplateforme qui simplifie la configuration d'un environnement de serveur local pour le développement web. Il comprend Apache (serveur web), MySQL (serveur de base de données), PHP et Perl, offrant un package pratique pour la création et le test d'applications web dynamiques sur un ordinateur personnel.

### 3- L'algorithme du projet :

YASABRACELET (YASMINESAFABRACELET)



Les bases de données proches, Docteur, Patient seront visibles dans SQL :

- Relative, Patient, Doctor

Les bases de Connaissances seront visibles dans le contenu du projet :

Où les tableau dans chaque fichier au projet contient tous les informations de chaque base de connaissance :

- *Liste des patients* base de connaissance est visible dans patientinfo dans le projet
- *Diagnostic* base de connaissance est visible dans doctDiag dans le projet
- *Braceletinfo* base de connaissance est visible dans le tableau qui contient ces infos dans viewbraceletinfo dans le projet
- *Et les liste des invitations* sera visible aussi dans patientdata ou chaque patient peut choisi le docteur voulue dans la liste afficher

## Chapitre II : Document de spécifications du projet :

### 1-Introduction :

Le projet vise à développer une application Java pour surveiller en temps réel les paramètres vitaux d'un patient. Le système comprend six tables dans la base de données MySQL (patient, user, doctor, history, moreinformations, relative, braceletinfo). L'application fournira des fonctionnalités spécifiques pour les proches, les patients et les médecins.

### 2. Fonctionnalités générales:

- Surveillance en temps réel
- Détection précoce des variations
- Capacité de localisation
- Connectivité pour la télémédecine
- Gestion intelligente de l'énergie

### 3. Accès utilisateur:

- **Patient**
  - Visualisation en temps réel les mesures
  - Visualisation l'historique
  - Ajouter proche (1 uniquement)
  - Réception des commentaires du médecin
- **Docteur:**
  - Visualiser la liste des patients
  - Ajouter nouveau patient
  - Retirer patient
  - Visualisation des paramètres vitaux
  - Changer les seuils
  - Commentaire : Faire le diagnostic
- **Proche:**
  - Visualisation des données
  - Réception des commentaires du médecin
  - Visualisation de la localisation

### 4. Interfaces Utilisateurs:

- **Home Page:**
  - Authentification (patient/doctor/relative).
  - Accès aux fonctionnalités spécifiques au rôle.

**- Patient:**

- Voir ses informations.
- Voir les informations du bracelet.
- Voir Diagnostic du docteur.
- Visualiser ses données.
- Voir la liste des docteurs.

**- Docteur:**

- Voir les informations du bracelet.
- Ajouter un docteur.
- Voir l'historique du patient.
- Voir la liste des docteurs.

**- Proche:**

- Voir Diagnostic du docteur.
- Voir les informations du bracelet.
- Voir l'historique du patient.
- Ajouter ses informations.

**5. Gestion de base de données :**

- Architecture des tables (`patient`, `user`, `doctor`, `relative`, `moreinformations`, `history`, `braceletinfo`).
- Établissement de relations de clés étrangères appropriées.

**6. Sécurité:**

- Authentification et autorisation sécurisées.
- Chiffrement des données sensibles.

**7. Langages et outils:**

- Utilisation de Java pour le développement d'applications.
- MySQL avec PhpMyAdmin pour la gestion du database.
- User-friendly UI design.

**8. Contraintes de temps:**

- Livraison du produit final sous un mois.

**9. Test:**

- Plan de test détaillé pour chaque fonctionnalité.
- Intégration et le système.

**10. Documentation:**

- Documentation complète du code.



- Manuels d'utilisation et d'administration.

### 11. Maintenance:

- Assurer la maintenance du système après le déploiement.
- Traiter les problèmes potentiels et mettre à jour les fonctionnalités si nécessaire.

### 12. Deliverables:

- Code source complet.
- Documentation technique et manuels d'utilisation
- MySQL data base prête pour l'utilisation.

## Chapitre III: Implémentation du Projet:

### 1- Création de la Base de Données sur phpMyAdmin:

**Database:** yasabracelet

**Les tables Créées:** patient, history, moreinformations, doctor, user, relative, braceletinfo.

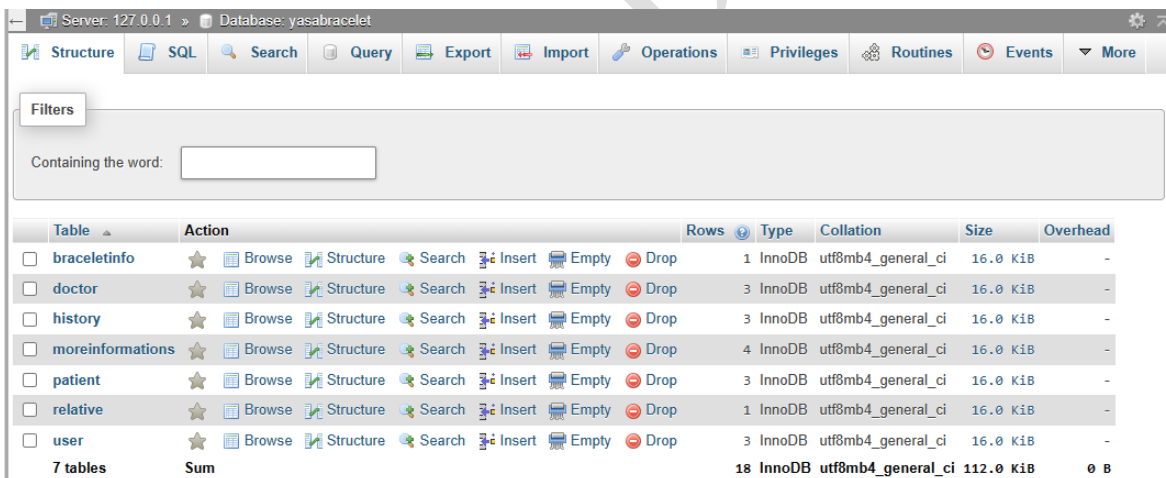


Table	Action	Rows	Type	Collation	Size	Overhead
braceletinfo	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	16.0 KiB	-
doctor	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	16.0 KiB	-
history	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	16.0 KiB	-
moreinformations	Browse Structure Search Insert Empty Drop	4	InnoDB	utf8mb4_general_ci	16.0 KiB	-
patient	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	16.0 KiB	-
relative	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	16.0 KiB	-
user	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<b>7 tables</b>	<b>Sum</b>	<b>18</b>	<b>InnoDB</b>	<b>utf8mb4_general_ci</b>	<b>112.0 KiB</b>	<b>0 B</b>

Figure 1: database's tables

### 2-Création de projet sur NetBeans :

**Nom du projet:** yasminesafabracelet

**Les Jframes créés:** Doctor.java , Braceletinfo.java , Login.java , Main.java , Signup.java , moreinformations.java , Patientdata.java , Patient.java , Relative.java , viewPatientHistory.java , viewBraceletInfo.java , viewDoctor.java , Doctordiag.java .

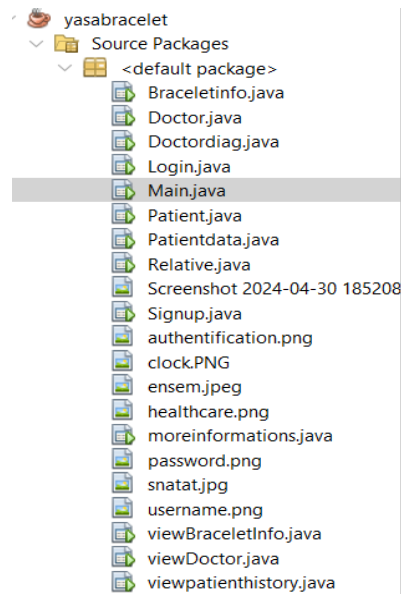


Figure 2: Project's files

a-Main.java :

Homepage on visualise apres le login dans homepage le username et le utype

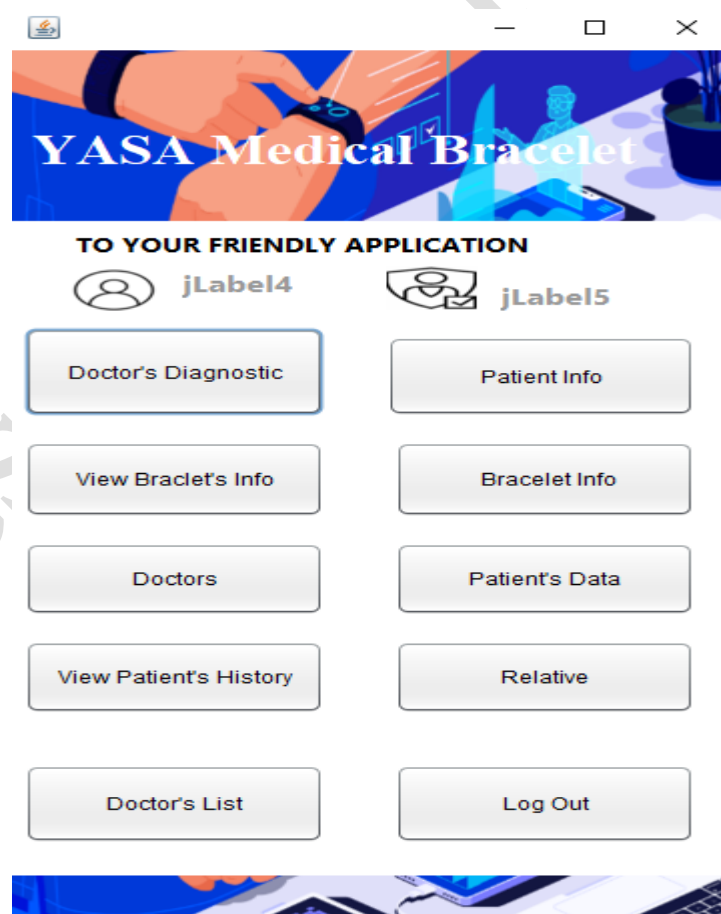


Figure 3:Main.java

b-Signup.java:

Pour Créer un nouvel utilisateur (doctor, relative, patient)

Figure 4: Signup.java

b.1-table user

On a créé et visualisé des nouveaux utilisateurs qu'on peut utiliser pour le login.

Server: 127.0.0.1 » Database: yasabracelet » Table: user

Table structure

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	name	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	username	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
4	password	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
5	utype	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More

Figure 5:table user

c-Login.java

Pour accéder au Home page, on a besoin de s'authentifier comme un docteur ; un patient ou un proche :

Figure 6:Login.java

On fait l'enregistrement d'un docteur (yasmine) , relative (safa) , patient (hicham)  
Le contenu de la table user

				id	name	username	password	utype
<input type="checkbox"/>	Edit	Copy	Delete	1	yasmine	ouazzine	2003	Doctor
<input type="checkbox"/>	Edit	Copy	Delete	2	safa	bechchaa	2002	Relative
<input type="checkbox"/>	Edit	Copy	Delete	3	hicham	medroumi	2000	Patient

d-Patient.java

On a créé Patient pour des informations telles que le nom et l'Age.

patientno	name	age
PS001	Kaoutar	36
PS002	Keltoum	44
PS003	Amine	29

Figure 7:patient.java

#### d.1-patient table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	patientno	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
2	name	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	age	int(10)			No	None			Change Drop More

Figure 8:patient table

On a le contenu de la table patient :

				patientno	name	age
<input type="checkbox"/>		Edit		Copy		Delete
				PS001	Kaoutar	36
<input type="checkbox"/>		Edit		Copy		Delete
				PS002	Keltoum	44
<input type="checkbox"/>		Edit		Copy		Delete
				PS003	Amine	29

e-Doctor.java

Après le Login; chaque docteur peut entrer ses informations dans Doctor's Informations



**\*\*Doctor Information\*\***

Doctor No: DS004

Full Name:

Specialization:

Phone:

Doctor_No	Doctor_Name	specialization	phone
DS001	yasmine	heart doctor	0665450160
DS002	Safa	brain	0641634517
DS003	Medroumi	General	0645234512

Save Exit

Figure 12: doctor info

e.1-doctor table:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1 Doctor_No	varchar(255)	utf8mb4_general_ci		No	None			Change  Drop  More
<input type="checkbox"/>	2 Doctor_Name	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change  Drop  More
<input type="checkbox"/>	3 specialization	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change  Drop  More
<input type="checkbox"/>	4 phone	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change  Drop  More
<input type="checkbox"/>	5 log_id	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change  Drop  More

Figure 13: doctor table

On a le contenu de la table docteur :

			Doctor_No	Doctor_Name	specialization	phone	log_id
<input type="checkbox"/>	Edit	Copy	Delete	DS001	yasmine	heart doctor	0665450160 1
<input type="checkbox"/>	Edit	Copy	Delete	DS002	Safa	brain	0641634517 2
<input type="checkbox"/>	Edit	Copy	Delete	DS003	Medroumi	General	0645234512 3

f-viewDoctor.java

Pour que le patient peut visualiser la liste des patient on a créer un tableau quisidplay tous les infos inserer par le docteur dans docteur.java



Figure 9:viewDoctor.java

g-Patientdata.java:

c'est ou le patient peut choisir le docteur qui traite ces bracelets info et le donne un diagnostic avec un nouveau seuil

Figure 15:Patientdata.java

Pt_No	Doctor_N...	Pt_Name	Location	Diag_Date
PN001	DS001	PS001	Lissasfa	2023-04-23
PN002	DS002	PS002	Chaymae	2023-02-23
PN003	DS003	PS003	Agdal	2023-01-12

g.1-History table:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1 Pt_No	varchar(255)	utf8mb4_general_ci		No	None			Change  Drop  More
<input type="checkbox"/>	2 Doctor_Name	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change  Drop  More
<input type="checkbox"/>	3 Pt_Name	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change  Drop  More
<input type="checkbox"/>	4 Location	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change  Drop  More
<input type="checkbox"/>	5 Diag_Date	date			Yes	NULL			Change  Drop  More

Figure 16:History table

On a le contenu de la table History :

<div><div><div></div><div></div><div></div></div><div></div></div>				Pt_No	Doctor_Name	Pt_Name	Location	Diag_Date
<div><div><div></div></div><div><div><div></div><div></div><div></div></div><div></div></div></div>	<div><div></div><div>Edit</div></div>	<div><div></div><div>Copy</div></div>	<div><div></div><div>Delete</div></div>	PN001	DS001	PS001	Lissasfa	2023-04-23
<div><div><div></div></div><div><div><div></div><div></div><div></div></div><div></div></div></div>	<div><div></div><div>Edit</div></div>	<div><div></div><div>Copy</div></div>	<div><div></div><div>Delete</div></div>	PN002	DS002	PS002	Chaymae	2023-02-23
<div><div><div></div></div><div><div><div></div><div></div><div></div></div><div></div></div></div>	<div><div></div><div>Edit</div></div>	<div><div></div><div>Copy</div></div>	<div><div></div><div>Delete</div></div>	PN003	DS003	PS003	Agdal	2023-01-12

h-viewpatienthistory.java:

Pour que le docteur et le proche peuvent visualiser la liste d'historique de patient on a créer un tableau qui display tous les infos inserer par le patient dans patientdata.java



Figure 17:view patient history .java



i-moreinformations :

Comme il ya un bouton dans viewpatienthistory c'est un bouton où après cliquer sur un des choix le docteur peut entrer sans diagnostic ( commentaire ) pour le patient choisis

Figure 108:moreinformations.java

Figure19:the combo box choices

i.1-moreinformations table:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1 Patcond_no	varchar(255)	utf8mb4_general_ci		No	None			Change  Drop  More
<input type="checkbox"/>	2 Pt_No	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change  Drop  More
<input type="checkbox"/>	3 Threshold	int(11)			Yes	NULL			Change  Drop  More
<input type="checkbox"/>	4 Diagnostic	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change  Drop  More
<input type="checkbox"/>	5 Doctor_Name	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change  Drop  More

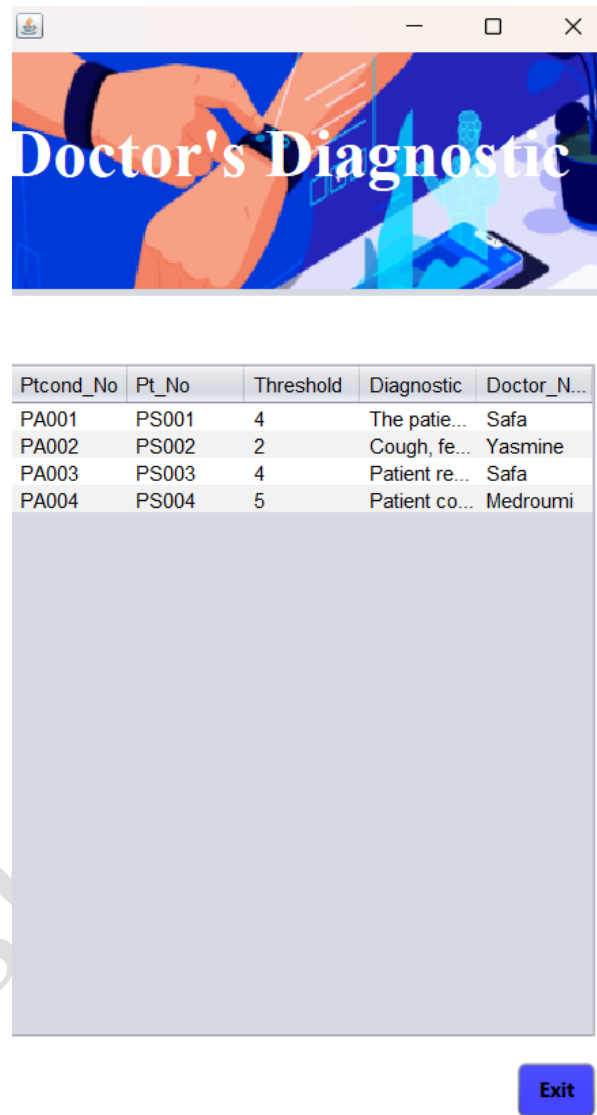
Figure 110:moreinformations table

On a le contenu de la table moreinformation qui contient le diagnostic du docteur :

			Patcond_no	Pt_No	Threshold	Diagnostic	Doctor_Name
<input type="checkbox"/>	Edit	Copy	Delete	PA001	PS001	4 The patient status is very stable at this point, n...	Safa
<input type="checkbox"/>	Edit	Copy	Delete	PA002	PS002	2 Cough, fever, and difficulty breathing. Suspected ...	Yasmine
<input type="checkbox"/>	Edit	Copy	Delete	PA003	PS003	4 Patient reports abdominal pain, nausea, and vomiti...	Safa
<input type="checkbox"/>	Edit	Copy	Delete	PA004	PS004	5 Patient complains of headache, sensitivity to ligh...	Medroumi

j-doctorDiag.java:

Pour que le patient et le proche peuvent visualiser le diagnostic/commentaire inserer par le docteur avec le suil modifier on a cree un tableau qui display tous les infos inserer dans moreinformations.java



Ptcond_No	Pt_No	Threshold	Diagnostic	Doctor_N...
PA001	PS001	4	The patie...	Safa
PA002	PS002	2	Cough, fe...	Yasmine
PA003	PS003	4	Patient re...	Safa
PA004	PS004	5	Patient co...	Medroumi

Figure 21:doctorDiag.java

k-Braceletinfo.java:

cette partie est pour les informations recus par le bracelet arduino qui contient temperature seuil et blood pressure temperature

**Bracelet Info**

Bracelet Id: **B1002**

Full Name:

Phone:

Relative's Phone:

Temp measure:

Blood pres measure:

Bracelet...	Full_N...	phone	Relativ...	Temp_...	Bloodp...
B1001	Kaouta...	645345...	666010...	37	23

Add Update Delete Exit

Figure 22:baraceletinfo.java

k.1-Braceletinfo table:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	Bracelet_id	varchar(255)	utf8mb4_general_ci		No	None			Change  Drop  More
<input type="checkbox"/> 2	Full_Name	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change  Drop  More
<input type="checkbox"/> 3	Phone	int(11)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 4	Relative_Phone	int(11)			Yes	NULL			Change  Drop  More
<input type="checkbox"/> 5	Temp_measure	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change  Drop  More
<input type="checkbox"/> 6	Bloodpress_measure	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change  Drop  More

Figure 23:baraceletinfo table

On a le contenu de la table Braceletinfo :

	Bracelet_id	Full_Name	Phone	Relative_Phone	Temp_measure	Bloodpress_measure
<input type="checkbox"/> Edit  Copy  Delete	BI001	Kaoutar Hafidi	645345678	666010234	37	23

L-Relative.java:

cette partie pour les relatives il faut qu'il insère plus informations sur leurs relation avec le patient

Relative No **RN002**

Relative name

Relationship

Phone

Relative_No	Relative_Na...	Relationship	phone
RN001	Hassan Baz	Cousin	645367299

Add Update Delete Exit

Figure 24:Relative.java

L.1-Relative table:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1 <b>Relative_No</b>	varchar(255)	utf8mb4_general_ci		No	None			Change  Drop  More
<input type="checkbox"/>	2 <b>Relative_Name</b>	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change  Drop  More
<input type="checkbox"/>	3 <b>Relationship</b>	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change  Drop  More
<input type="checkbox"/>	4 <b>Phone</b>	int(11)			Yes	NULL			Change  Drop  More

Figure 25: relative table

On a le contenu de la table relative :

← T →	Relative_No	Relative_Name	Relationship	Phone
<input type="checkbox"/> Edit  Copy  Delete	RN001	Hassan Baz	Cousin	645367299

M-viewbraceletinfo.java:

Pour que le docteur peut voir les mesures et les informations recus par le bracelet lui et les proches on a cree ce tableau qui display tous les infos que les deux seulement peuvent voir.

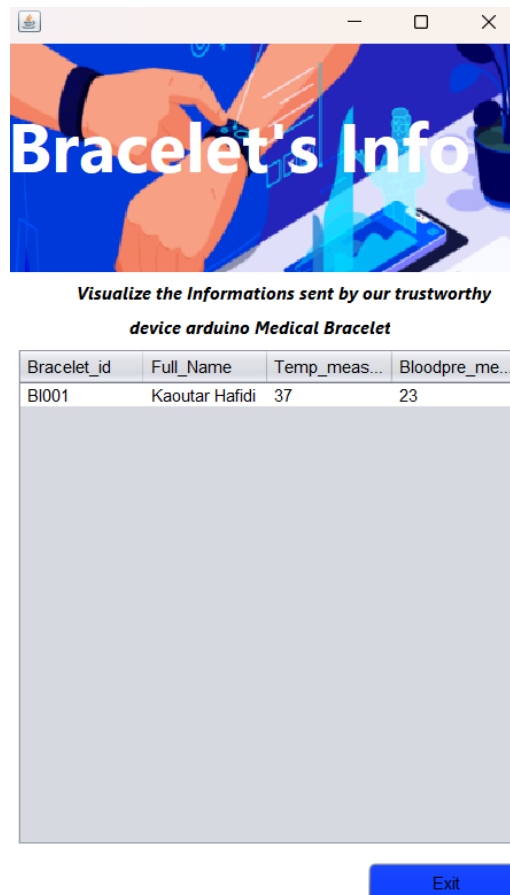


Figure 26: baraceletinfo table

N- Partie du code de connexion:

Dans tous les fichiers .java que nous avons créés, nous avons ajouté une partie de connexion pour se connecter à la base de données. Nous pouvons prendre comme exemple le fichier Patient : Patient.java

```
public void Connect() {
    try {
        Class.forName("com.mysql.jdbc.Driver");
        con = DriverManager.getConnection("jdbc:mysql://localhost:3306/yasabracelet", "root", "");
    } catch (ClassNotFoundException ex) {
        Logger.getLogger(Signup.class.getName()).log(Level.SEVERE, null, ex);
    } catch (SQLException ex) {
        Logger.getLogger(Signup.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

Figure 27: connect Method

Le code Java fourni définit une méthode nommée `Connect` conçue pour établir une connexion à une base de données MySQL. La première étape consiste à charger le pilote JDBC MySQL en utilisant l'instruction `Class.forName("com.mysql.jdbc.Driver");`. Ensuite, une connexion est établie. À travers la méthode `DriverManager.getConnection`, où l'URL spécifie l'emplacement de la base de données (`localhost:3306/yasabracelet`), et les paramètres incluent le nom d'utilisateur (`root`) et un mot de passe vide. La gestion des exceptions est mise en œuvre à la fois pour le chargement du pilote JDBC et les erreurs potentielles lors du processus de connexion à la base de données. Toutes les exceptions sont enregistrées pour une analyse ultérieure. Il est important de noter que la variable `con`, vraisemblablement une variable d'instance de la classe `User`, est utilisée pour stocker la connexion établie à la base de données. Cependant, le code pourrait bénéficier de pratiques supplémentaires de gestion des erreurs et des ressources, telles que la fermeture de la connexion dans un bloc `finally`, pour garantir la robustesse.

```
public Patient() {
    initComponents();
    Connect();
    AutoID();
    patient_table();
}
```

**Connect ():** method to connect to the database.  
**AutoID ():** for the Patient Number.  
**patient\_table ():** the table that we can visualize in it the patient information created.

```
Connection con;
PreparedStatement pst;
ResultSet rs;
```

Figure 28:Patient

```
public void AutoID() {
    try {
        Statement s = con.createStatement();
        rs = s.executeQuery("select MAX(patientno) from patient");
        rs.next();
        rs.getString("MAX(patientno)");

        if (rs.getString("MAX(patientno)") == null) {
            lblpno.setText("PS001");
        } else {
            long id = Long.parseLong(rs.getString("MAX(patientno)").substring(2, rs.getString("MAX(patientno)").length()));
            id++;
            lblpno.setText("PS" + String.format("%03d", id));
        }
    }

    catch (SQLException ex) {
        Logger.getLogger(Patient.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

Figure29:AutoID

Nous sélectionnons patientno dans la base de données des patients et nous effectuons AutoID pour le numéro de patient.

```

public void patient_table() {

    try {
        pst = con.prepareStatement("select * from patient");
        rs = pst.executeQuery();
        ResultSetMetaData Rsm = rs.getMetaData();
        int c;
        c = Rsm.getColumnCount();
        DefaultTableModel df = (DefaultTableModel) jTable1.getModel();
        df.setRowCount(0);

        while (rs.next()) {
            Vector v2 = new Vector();

            for (int i = 1; i <= c; i++) {

                v2.add(rs.getString("patientno"));
                v2.add(rs.getString("name"));
                v2.add(rs.getString("age"));
            }
            df.addRow(v2);
        }

    } catch (SQLException ex) {
        Logger.getLogger(Patient.class.getName()).log(Level.SEVERE, null, ex);
    }

}

```

Figure30:patient\_table

Le morceau de code `v2.add(rs.getString("patientno"));` ajoute une valeur à une forme de collection ou liste, désignée par `v2`. **Analysons le code :**

1. **rs.getString("patientno") :** Cela récupère la valeur de la colonne nommée "patientno" de la ligne actuelle d'un `ResultSet` (`rs`). Le `ResultSet` est généralement obtenu en exécutant une requête SQL sur une base de données.
2. **v2.add(...):** Cela ajoute la valeur récupérée à la collection ou liste représentée par `v2`. La méthode `add(...)` est couramment associée aux classes `List` ou `Collection` en Java.

*En résumé, cette ligne de code extrait la valeur de la colonne "patientno" de la ligne actuelle d'un `ResultSet` (supposé être une partie du résultat d'une requête sur une base de données) et ajoute cette valeur à la collection `v2`. Le type spécifique de collection (`List`, `ArrayList`, etc.) et le contexte dans lequel ce code est utilisé détermineraient le comportement exact et le but de cette opération.*

## CONCLUSION

En conclusion, la mise en œuvre du projet "Smart Bracelet Médical" a été un voyage d'apprentissage, de collaboration et d'application pratique de nos compétences en programmation. Le défi de comprendre et de traduire les instructions du professeur en une application Java cohérente et complète a été relevé avec dévouement et persévérance.

Tout au long du processus de développement dans NetBeans IDE 19, l'équipe a investi un temps considérable pour saisir les subtilités des exigences du projet, assurant une compréhension claire de chaque aspect décrit par le professeur. Cet engagement nous a permis de créer une application qui non seulement est conforme aux instructions données, mais qui ajoute également une touche personnelle en la nommant "**yasminesafabracelet**" pour refléter notre effort collectif et nos contributions individuelles.

La décision de dévier de la convention de dénomination conventionnelle pour incorporer nos noms à la fois dans le projet et la base de données témoigne de notre engagement à personnaliser et à prendre en charge le projet. Cette approche non seulement renforce notre sentiment d'accomplissement, mais démontre également notre capacité à s'adapter et à insuffler de la créativité dans le projet dans le cadre donné.

En résumé, le processus de création du Smart Bracelet Médical a non seulement approfondi notre compréhension du développement d'applications Java, mais a également mis en valeur notre adaptabilité et notre engagement à fournir un projet qui correspond à la vision de notre professeur tout en reflétant nos contributions uniques. Cette expérience a sans aucun doute enrichi notre connaissance pratique, nous préparant pour nos futures entreprises dans le développement de logiciels.