

PROJECT RAPPORT

Subject:
**Implementation of Patient
Record Management System**

Prepared by:

Safa BECHCHAA & Yasmine OUZZINE

Major :

Génie Logiciel et Digitalisation

ACKNOWLEDGMENTS

I would like to express my profound gratitude to Professor Meriyem Chergui for her exceptional dedication throughout the Java course and practical sessions. Her passion for teaching and efforts to make the content accessible and engaging greatly enhanced my learning experience.

The practical sessions, led by Professor Chergui, were particularly enriching. Her ability to explain complex concepts clearly and concisely facilitated my understanding of fundamental Java programming principles. Her ongoing support, readiness to answer our questions, and encouragement created a conducive and motivating learning environment.

Furthermore, I sincerely thank her for entrusting me with the project topic "Implementation of a Patient Record Management System." This stimulating and relevant project provided me with the opportunity to apply classroom knowledge and develop practical skills in software system design.

In conclusion, I am grateful to Professor Meriyem Chergui for her exceptional dedication to teaching and for significantly contributing to my academic journey. These learning moments will remain unforgettable and serve as a solid foundation for my future projects and accomplishments.

Table Of Contents

ACKNOWLEDGMENTS.....	2
List of Figures	5
INTRODUCTION	6
Chapitre I : Project	7
1- Project's Subject:	7
2- Used Environment:.....	7
Chapter II : Project Requirements Document:	9
1-Introduction:.....	9
2. General Features:	9
3. User Access:	9
4. User Interfaces:.....	9
5. Database Management:	10
6. Security:	10
7. Languages and Tools:	10
8. Time Constraints:.....	10
9. Testing:.....	10
10. Documentation(video):.....	10
11. Maintenance:	10
12. Deliverables:.....	10
Chapter III : Unified Modeling Langage :	11
1-Use Case Diagram:.....	11
a-Use Case 1:.....	11
a.1-Diagram:	11
b-Use Case 2:.....	12
b.1-Diagram:	12
2-Class Diagram:	13
a-Class :	13
a.1-Diagram:	13
Chapter IV : Project Implementation:	14
1- Creation of Database in phpMyAdmin:.....	14
2-Creation in NetBeans the project:	14
a-Main.java :	15
b-user.java:	15
b.1-table user.....	16

c-Login.java	16
d-Patient.java	17
d.1-patient table	17
e-Doctor.java	18
e.1-doctor table:	18
f-createChannel : Channel.java	19
f.1-Channel table:	19
g-viewChannel.java:	20
h-visit.java:	20
h.1-visit table:	21
i-viewDoctor.java:	21
j-moreinformations :	22
j.1-moreinformations table:	22
k-viewPatientInformation.java:	23
l- Connection Code Part:	23
3- We Insert Some Informations In The Database Tables:	25
a- channel table:.....	25
b- doctor table:	26
c- patient table:	26
d- visit table:	26
e- moreinformations table:.....	26
f- user table.....	27
CONCLUSION:	28

List of Figures

Figure 1:use case 1	11
Figure 2:use case 2	12
Figure 3:class.....	13
Figure 4:database's tables	14
Figure 5: Project's files	14
Figure 6:Main.java	15
Figure 7:user.java	15
Figure 8:table user	16
Figure 9:login.java.....	16
Figure 10:patient.java.....	17
Figure 11:patient table.....	17
Figure 12:doctor.java.....	18
Figure 13: doctor table	18
Figure 14:Channel.java	19
Figure 15:channel table	19
Figure 16:viewchannel.java.....	20
Figure 17:visit.java.....	20
Figure 18: visit table.....	21
Figure 19:viewDoctor.java.....	21
Figure 20:the combo box choices.....	22
Figure 21:moreinformations.java	22
Figure 22:moreinformations table	22
Figure 23:viewPatientInformation.java.....	23
Figure 24:connect Method.....	23
Figure 25:Patient	24
Figure 26:AutoID	24
Figure 27:patient_table.....	24
Figure 28:the rest of the patient_table	25
Figure 29:channel table	25
Figure 30:doctor table	26
Figure 31:patient table.....	26
Figure 32:visit table.....	26
Figure 33:moreinformations table	26
Figure 34:user table	27

INTRODUCTION

The realization of a patient record management system represents a stimulating and captivating challenge to which we dedicated a defined period within the framework of our academic curriculum. The project was planned and executed with a precise schedule, reflecting the delicate balance between the ambition to explore new dimensions of Java programming and the need to adhere to imposed time constraints.

The decision to use Java Application instead of Java EE (JEE) for this project was deliberate and strategic. Having previously worked on an e-commerce project using Java EE, we identified the opportunity to deepen our Java skills in a different context. The choice of Java Application is driven by our desire to diversify our knowledge and skills, exploring a facet of Java that had been relatively less explored within our academic program.

The decision to focus on Java Application stems from the belief that this project provides a unique opportunity to deepen our understanding of Java in specific domains while contributing to the development of a robust and functional patient record management application. This approach reflects our commitment to expanding our skill set and maximizing the learning experience offered by this project.

Chapitre I : Project

1- Project's Subject:

Title: Framework for Implementation of Patient Record Management System

Description: Development of an integrated patient record management system across various services

Features:

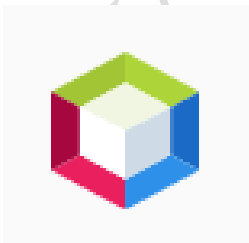
- Unique patient identification across all services
- Establishment of a shared common minimum record containing global patient information and clinical details (allergies (food, animals, etc.), pregnancy, medical history)
- Creation of patient records by specialty using different sources (patient interviews, clinical examinations, diagnostics, pathology)
- Implementation of a visit record to trace all patient visits to various hospital departments within the University Hospital Center (CHU).

2- Used Environment:

In the context of our project "Implementation of a Patient Record Management System," we have established a cohesive technological environment, integrating multiple tools to effectively meet our development needs. Here is an overview of the main components of this environment:



JAVA: Java is a versatile, object-oriented programming language known for its portability and platform independence. Developed by Sun Microsystems, it provides a robust framework for building scalable applications. Java's key features include its simplicity, security, and the ability to run on various platforms without recompilation, making it a popular choice for diverse software development projects.



NetBeans IDE 19: NetBeans is an Integrated Development Environment (IDE) that simplifies and accelerates the software development process. Equipped with comprehensive tools and supporting multiple programming languages, it streamlines coding, debugging, and project management. With its user-friendly interface, NetBeans IDE is a powerful tool for constructing diverse applications.



efficiently.

Système de Gestion de Base de Données (SGBD) -

MySQL via PHPMyAdmin : For managing data related to patient records, we integrated MySQL using the user-friendly interface of PhpMyAdmin. This combination allows us to design and administer our database



Environnement de Développement Local - XAMPP :

XAMPP is an open-source, cross-platform software stack that simplifies setting up a local server environment for web development. It includes Apache (web server), MySQL (database server), PHP, and Perl, providing a convenient package for creating and testing dynamic web applications on a personal computer.

Chapter II : Project Requirements Document:

1-Introduction:

The project aims to develop a Java application for managing patient records. The system includes six tables in the MySQL database (patient, user, doctor, visit, moreinformations, channel). The application will provide specific functionalities for receptionists and doctors.

2. General Features:

- Creation of patient records.
- Assignment of patients to doctors.
- Recording medical visits.
- Addition of extra information by doctors.

3. User Access:

- Receptionist:

- Create a new patient record.
- Manage channels (create).
- Create a user (doctor, receptionist).
- view doctor.
- visit (create, history).

- Doctor:

- View assigned channels.
- View doctor information.
- Add additional information to patient records during visits.
- View channels assigned to them.
- View Doctors.
- View Patient Information.

4. User Interfaces:

- Home Page:

- Authentication (receptionist/doctor).
- Access to role-specific functionalities.

- Receptionist:

- Create patient records.
- Manage channels.
- Create users (doctors).

- Doctor:

- View assigned channels.
- View medical information.
- Add extra information to the channel.

5. Database Management:

- Design of tables (`patient`, `user`, `doctor`, `visit`, `moreinformations`, `channel`).
- Establishment of appropriate foreign key relationships.

6. Security:

- Secure authentication and authorization.
- Encryption of sensitive data.

7. Languages and Tools:

- Use Java for application development.
- MySQL with PhpMyAdmin for database management.
- User-friendly UI design.

8. Time Constraints:

- Delivery of the final product within one month.

9. Testing:

- Detailed test plan for each functionality.
- Integration and system testing.

10. Documentation(video):

- Comprehensive code documentation.
- User and administrator manuals.

11. Maintenance:

- Ensure system maintenance after deployment.
- Address potential issues and update features as needed.

12. Deliverables:

- Complete source code.
- Technical documentation and user manuals.
- MySQL database ready for use.

Chapter III : Unified Modeling Language :

1-Use Case Diagram:

a-Use Case 1:

a.1-Diagram:

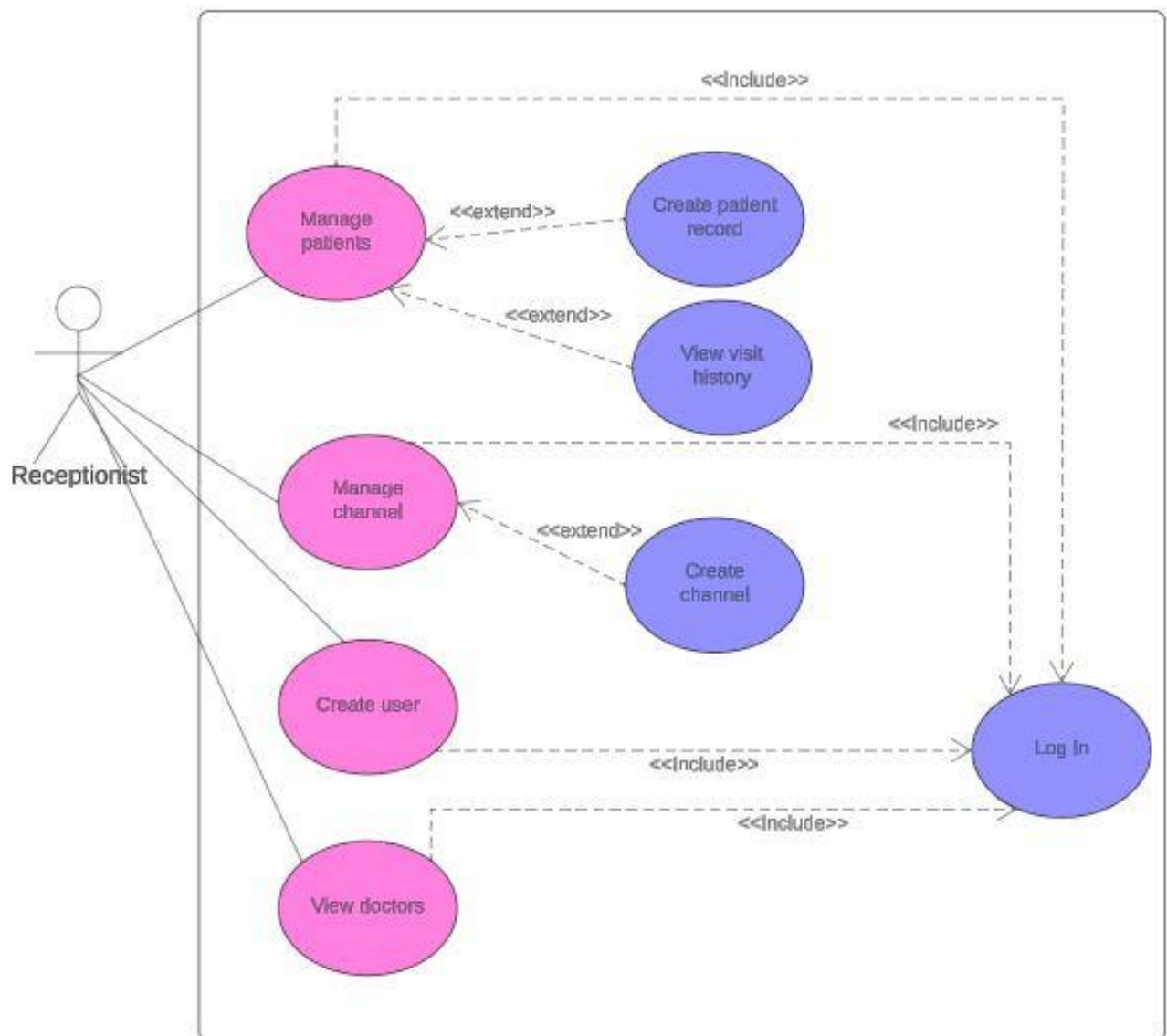


Figure 1:use case 1

b-Use Case 2:

b.1-Diagram:

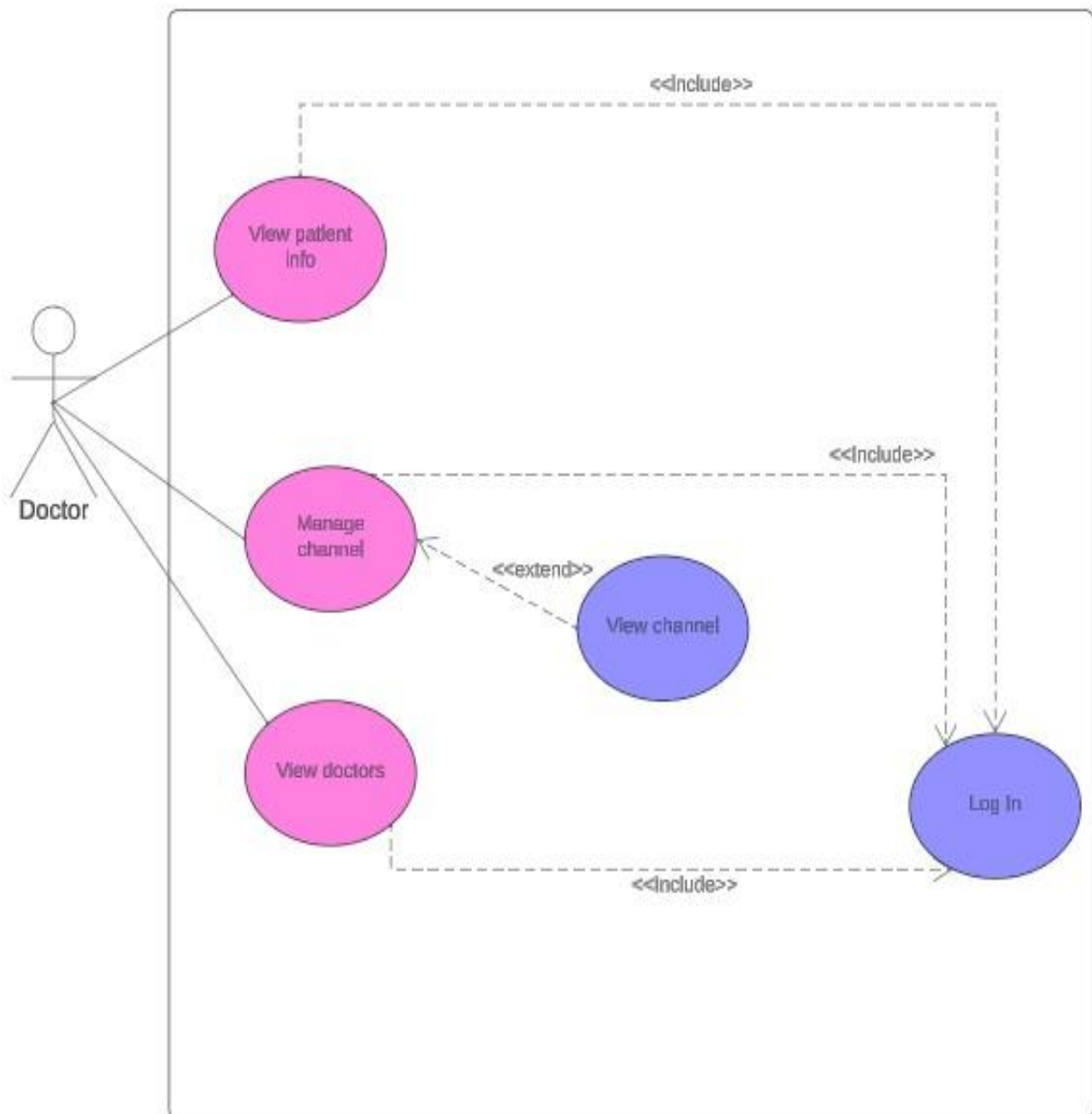


Figure 2:use case 2

2-Class Diagram:

a-Class :

a.1-Diagram:

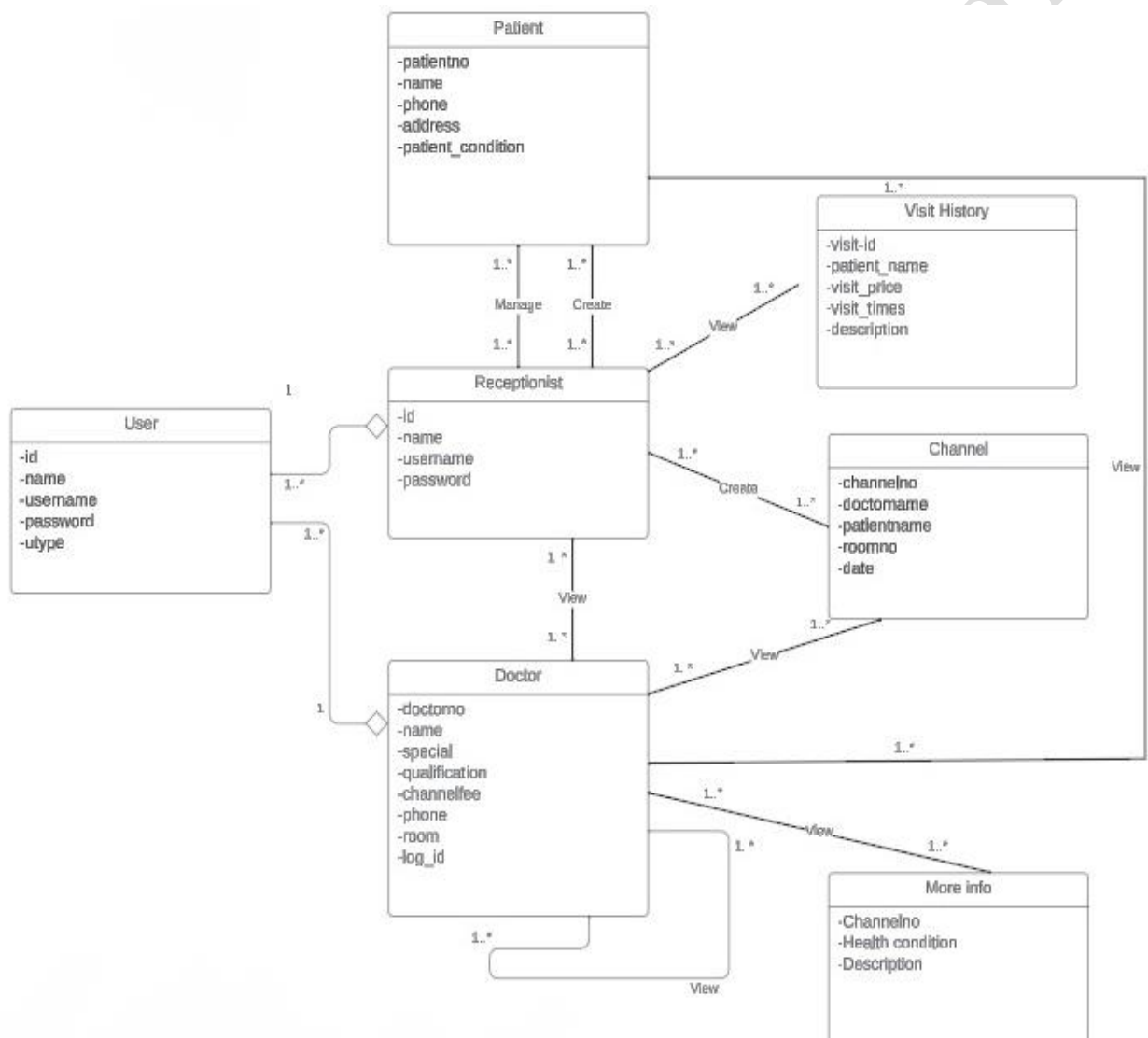


Figure 3: class

Chapter IV : Project Implementation:

1- Creation of Database in phpMyAdmin:

Database: yasminesafahospital.

Tables Created: patient, channel, moreinformations, doctor, user, visit.

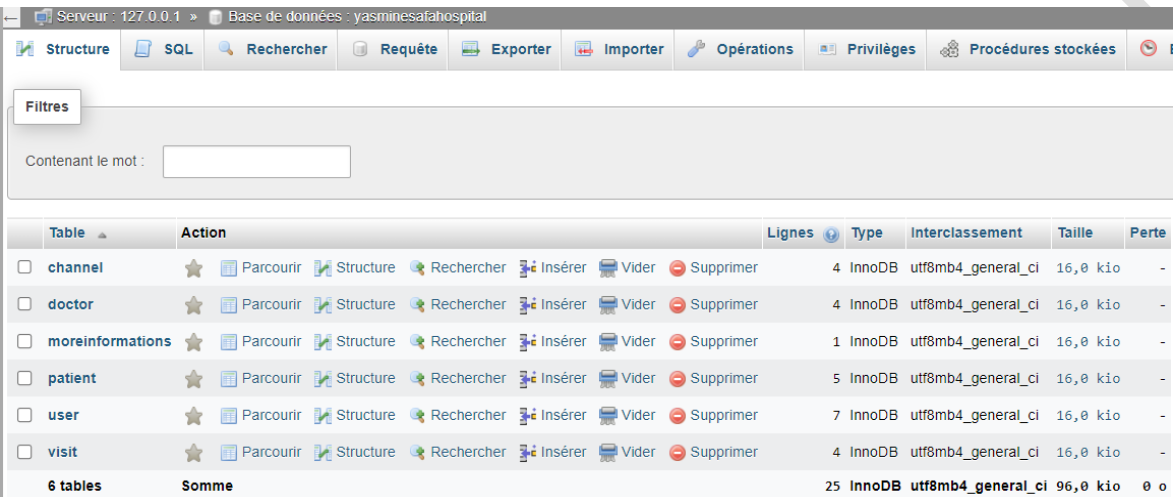


Table	Action	Lignes	Type	Interclassement	Taille	Perte
channel	Parcourir Structure Rechercher Insérer Vider Supprimer	4	InnoDB	utf8mb4_general_ci	16,0 kio	-
doctor	Parcourir Structure Rechercher Insérer Vider Supprimer	4	InnoDB	utf8mb4_general_ci	16,0 kio	-
moreinformations	Parcourir Structure Rechercher Insérer Vider Supprimer	1	InnoDB	utf8mb4_general_ci	16,0 kio	-
patient	Parcourir Structure Rechercher Insérer Vider Supprimer	5	InnoDB	utf8mb4_general_ci	16,0 kio	-
user	Parcourir Structure Rechercher Insérer Vider Supprimer	7	InnoDB	utf8mb4_general_ci	16,0 kio	-
visit	Parcourir Structure Rechercher Insérer Vider Supprimer	4	InnoDB	utf8mb4_general_ci	16,0 kio	-
6 tables	Somme	25	InnoDB	utf8mb4_general_ci	96,0 kio	0 o

Figure 4: database's tables

2- Creation in NetBeans the project:

Project name: yasminesafahospital

Jframes created: Channel.java , Doctor.java , Login.java , Main.java , User.java , moreinformations.java , viewChannel.java , viewDoctor.java , viewPatientInformation.java , visit.java.

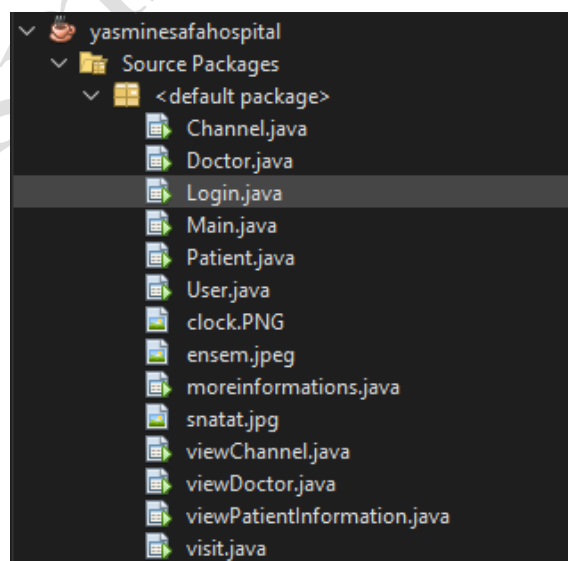


Figure 5: Project's files

a-Main.java :

Home page (to understand more who can see each button of the main page after log in watch the video)



Figure 6:Main.java

b-user.java:

To create new user (doctor, receptionist)



Figure 7:user.java

b.1-table user

we create and visualize here new users that we can use to login

Serveur : 127.0.0.1 » Base de données : yasmimesafahospital » Table : user

Parcourir Structure SQL Rechercher Insérer Exporter Importer Privilèges Opérations Suivi Déclarer

Structure de table Vue relationnelle

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
<input type="checkbox"/> 1	id	int(11)			Non	Aucun(e)		AUTO_INCREMENT	Modifier Supprimer Plus
<input type="checkbox"/> 2	name	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/> 3	username	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/> 4	password	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/> 5	utype	varchar(50)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus

Figure 8:table user

c-Login.java

To reach the home page we need to login as a doctor or receptionist

Figure 9:login.java

d-Patient.java

we create patients and add their personal informations such as patient name, phone, address

Figure 10:patient.java

d.1-patient table

Table : patient									
#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
<input type="checkbox"/> 1	patientno	varchar(255)	utf8mb4_general_ci		Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/> 2	name	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/> 3	phone	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/> 4	address	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/> 5	patient_condition	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus

Figure 11:patient table

e-Doctor.java

Each doctor after log in can enter his own informations in the doctor registration and the receptionist can see it in view doctor

Doctor No	Doctor Name	Specialization	Qualification	Channel fee	Phone	Room No

Figure

e.1-doctor table:

<div> <div> <div>Sever : 127.0.0.1 » Base de données : yasminesafahospital » Table : doctor</div> <div> <div>Parcourir</div> <div>Structure</div> <div>SQL</div> <div>Rechercher</div> <div>Insérer</div> <div>Exporter</div> <div>Importer</div> <div>Privileges</div> <div>Operations</div> <div>Suivi</div> </div> </div> <div> <div>Structure de table</div> <div>Vue relationnelle</div> </div> </div>									
#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
<input type="checkbox"/>	1 doctorno	varchar(255)	utf8mb4_general_ci		Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/>	2 name	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/>	3 special	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/>	4 qualification	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/>	5 channelfee	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/>	6 phone	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/>	7 room	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/>	8 log_id	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus

Figure 13: doctor table

f-createChannel : Channel.java

To create a channel that contains the doctor name assigned to the patient and in which room the patient is going to stay and the date channel, when he was assigned to this room

Figure 14:Channel.java

f.1-Channel table:

Serveur : 127.0.0.1 » Base de données : yasmimesafahospital » Table : channel

Parcourir Structure SQL Rechercher Insérer Exporter Importer Privileges Opérations Suivi

Structure de table Vue relationnelle

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
<input type="checkbox"/> 1	channelno	varchar(255)	utf8mb4_general_ci		Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/> 2	doctornome	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/> 3	patientname	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/> 4	roomno	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/> 5	date	date			Oui	NULL			Modifier Supprimer Plus

Figure 15:channel table

g-viewChannel.java:

when we login as doctor we can see channel to add more informations to the patient in the channel chosen

Figure 16:viewchannel.java

h-visit.java:

The visit the receptionist can reserve a visit for a patient or delete the visit or update the informations after the visit

Figure 17:visit.java

h.1-visit table:

Serveur : 127.0.0.1 » Base de données : yasminealahospital » Table : visit

Parcourir Structure SQL Rechercher Insérer Exporter Importer Privilèges Opérations Suivi

Structure de table Vue relationnelle

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
<input type="checkbox"/> 1	visit_id	varchar(255)	utf8mb4_general_ci		Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/> 2	patient_name	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/> 3	visit_price	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/> 4	visit_times	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/> 5	description	text	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus

Figure 18: visit table

i-viewDoctor.java:

The receptionist can visualize the names of doctors and there informations such as the room assigned to them and their qualifications



Figure 19:viewDoctor.java

j-moreinformations :

As in view channel there is a button of moreinformations that a doctor can click on the channel that he wants and add the informations on that patient

Figure 21:moreinformations.java

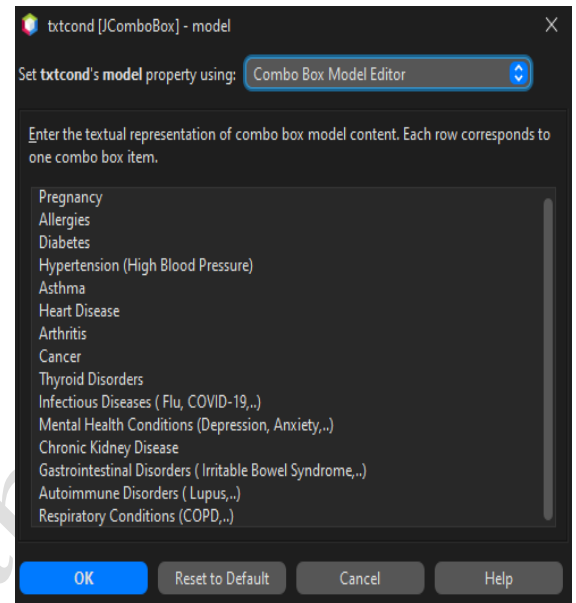


Figure 20:the combo box choices

j.1-moreinformations table:

Serveur : 127.0.0.1 » Base de données : yasminesafahospital » Table : moreinformations

Parcourir

Structure

SQL

Rechercher

Insérer

Exporter

Importer

Privileges

Operacions

Suivi

Déclencheurs

Structure de table

Vue relationnelle

	#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
<input type="checkbox"/>	1	Patcond_no	varchar(255)	utf8mb4_general_ci		Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/>	2	Channel_No	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/>	3	Health_Condition_Patient_State	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/>	4	Description	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
<input type="checkbox"/>	5	doctorname	varchar(255)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus

Figure 22:moreinformations table

k-viewPatientInformation.java:

To view the more informations entered by a certain doctor about a certain patient only a doctor after login can visualize since the information is confidential the receptionist cannot see it.

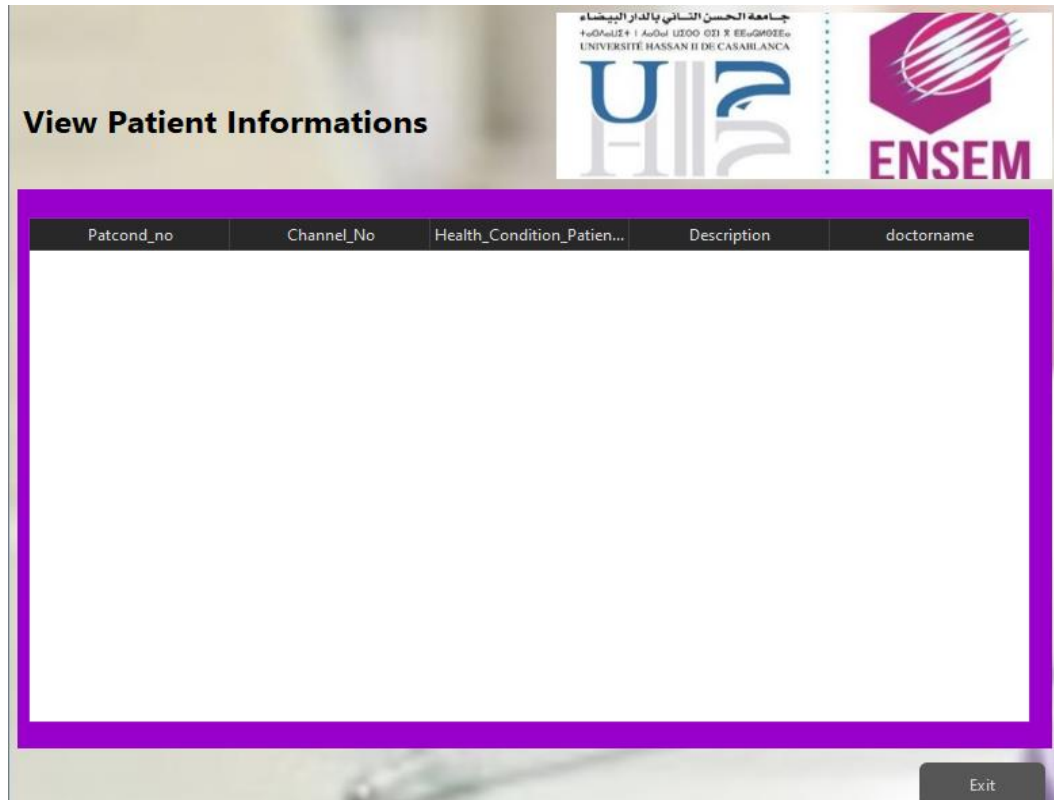


Figure 23:viewPatientInformation.java

I- Connection Code Part:

In all the files.java we created a connection part to connected to the database we can take as an example the Patient file: Patient.java

```
public void Connect() {
    try {
        Class.forName(className: "com.mysql.jdbc.Driver");
        con = DriverManager.getConnection(url:"jdbc:mysql://localhost:3306/yasminesafahospital", user:"root", password:"");
    } catch (ClassNotFoundException ex) {
        Logger.getLogger(name: User.class.getName()).log(level: Level.SEVERE, msg:null, thrown: ex);
    } catch (SQLException ex) {
        Logger.getLogger(name: User.class.getName()).log(level: Level.SEVERE, msg:null, thrown: ex);
    }
}
```

Figure 24:connect Method

The provided Java code defines a method named **`Connect`** designed to establish a connection to a MySQL database. The initial step involves loading the **MySQL JDBC** driver using the **`Class.forName("com.mysql.jdbc.Driver");`** statement. Subsequently, a connection is established through the **`DriverManager.getConnection`** method, where the URL specifies the database location

(`localhost:3306/yasminesafahospital`), and the parameters include the username (`root`) and an empty password. Exception handling is implemented for both the loading of the JDBC driver and potential errors during the database connection process. Any exceptions are logged for further analysis. It's important to note that the `con` variable, presumably an instance variable of the `User` class, is used to store the established database connection. However, the code might benefit from additional error handling and resource management practices, such as closing the connection in a `finally` block, to ensure robustness.

```

*/
public Patient() {
    initComponents();
    Connect();
    AutoID();
    patient_table();
}

Connection con;
PreparedStatement pst;
ResultSet rs;

```

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.

Figure 25: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", args.id));
        }
    } catch (SQLException ex) {
        Logger.getLogger(Patient.class.getName()).log(Level.SEVERE, null, ex);
    }
}

```

Figure 26:AutoID

We select the patientno from the database patients and to do AutoID for the Patient Number

```

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();

```

Figure 27:patient_table


```

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

            v2.add(e: rs.getString(string: "patientno"));
            v2.add(e: rs.getString(string: "name"));
            v2.add(e: rs.getString(string: "phone"));
            v2.add(e: rs.getString(string: "address"));

        }
        df.addRow(rowData:v2);

    }

} catch (SQLException ex) {
    Logger.getLogger(name: Patient.class.getName()).log(level: Level.SEVERE, msg:null, thrown: ex);
}
}

```

Figure 28:the rest of the patient_table

The code snippet `v2.add(rs.getString("patientno"));` is adding a value to some form of collection or list, denoted by `v2`. Let's break down the code:

1. rs.getString("patientno"): This retrieves the value of the column named "patientno" from the current row of a `ResultSet` (`rs`). The `ResultSet` is typically obtained from executing a SQL query on a database.

2. v2.add(...): This adds the retrieved value to the collection or list represented by `v2`. The method `add(...)` is commonly associated with List or Collection classes in Java.

Putting it together, this line of code fetches the value of the "patientno" column from the current row of a `ResultSet` (assumed to be part of a database query result) and adds that value to the collection `v2`. The specific type of collection (`List`, `ArrayList`, etc.) and the context in which this code is used would determine the exact behavior and purpose of this operation.

3- We Insert Some Informations In The Database Tables:

To have a clearer video on the functionalities of the java application i will insert some informations in the database tables

The informations inserted in the tables:

a- channel table:

☐ Tout afficher

Nombre de lignes : 25

Filtrer les lignes: Chercher dans cette table

Trier

Options supplémentaires

<div>←T→</div>	channelno	doctorname	patientname	roomno	date
<div><div><input type="checkbox"/></div><div><div>✎ Éditer</div><div>📄 Copier</div><div>🗑 Supprimer</div></div></div>	CH001	DS001	PS002	8	2024-01-06
<div><div><input type="checkbox"/></div><div><div>✎ Éditer</div><div>📄 Copier</div><div>🗑 Supprimer</div></div></div>	CH002	DS002	PS002	10	2024-01-05
<div><div><input type="checkbox"/></div><div><div>✎ Éditer</div><div>📄 Copier</div><div>🗑 Supprimer</div></div></div>	CH003	DS003	PS003	14	2024-01-19
<div><div><input type="checkbox"/></div><div><div>✎ Éditer</div><div>📄 Copier</div><div>🗑 Supprimer</div></div></div>	CH004	DS004	PS004	7	2024-01-17

Figure 29:channel table

b- doctor table:

☐ Tout afficher | Nombre de lignes : 25 | Filtrer les lignes: Chercher dans cette table | Trier par clé : Aucun(e)

Options supplémentaires

			doctorno	name	special	qualification	channelfee	phone	room	log_id
<input type="checkbox"/>	Éditer	Copier	Supprimer	DS001	yasmine ouazzine	surgeon	harvard graduated	2500	0600453470 8	1
<input type="checkbox"/>	Éditer	Copier	Supprimer	DS002	kaoutar ouazzine	Cardiology	stanford graduated	5677	0796567419 10	2
<input type="checkbox"/>	Éditer	Copier	Supprimer	DS003	ouazzine hiba	brain surgeon	pen graduated	4355	0777877707 14	3
<input type="checkbox"/>	Éditer	Copier	Supprimer	DS004	safa bechchaa	therapist	harvard degree	3000	0656765501 7	4

Figure 30:doctor table

c- patient table:

☐ Tout afficher | Nombre de lignes : 25 | Filtrer les lignes: Chercher dans cette table | Trier par clé : Aucun(e)

Options supplémentaires

				patientno	name	phone	address	patient_condition
<input type="checkbox"/>	Éditer	Copier	Supprimer	PS001	kaoutar attanan	0665443322	haj fateh lot elkhousama habiba	NULL
<input type="checkbox"/>	Éditer	Copier	Supprimer	PS002	ayoub hamaoui	0787986610	fes	NULL
<input type="checkbox"/>	Éditer	Copier	Supprimer	PS003	hanane assendale	0787665610	agadir	NULL
<input type="checkbox"/>	Éditer	Copier	Supprimer	PS004	siham aouiss	0656441927	casablanca	NULL
<input type="checkbox"/>	Éditer	Copier	Supprimer	PS005	taha mohaddere	0665342945	ain chok	NULL

Figure 31:patient table

d- visit table:

☐ Tout afficher | Nombre de lignes : 25 | Filtrer les lignes: Chercher dans cette table | Trier par clé : Aucun(e)

Options supplémentaires

				visit_id	patient_name	visit_price	visit_times	description
<input type="checkbox"/>	Éditer	Copier	Supprimer	VS001	kaoutar attanan	2500	1	first time
<input type="checkbox"/>	Éditer	Copier	Supprimer	VS002	ayoub hamaoui	5677	2	check up
<input type="checkbox"/>	Éditer	Copier	Supprimer	VS003	hanane assendale	4355	6	just a visit to check the wound
<input type="checkbox"/>	Éditer	Copier	Supprimer	VS004	siham aouiss	3000	1	first time

Figure 32:visit table

e- moreinformations table:

☐ Tout afficher | Nombre de lignes : 25 | Filtrer les lignes: Chercher dans cette table

Options supplémentaires

				Patcond_no	Channel_No	Health_Condition_Patient_State	Description	doctornome
<input type="checkbox"/>	Éditer	Copier	Supprimer	PA001	CH001	Allergies	peanuts allergie	yasmine ouazzine

Figure 33:moreinformations table

f- user table

☐ Tout afficher

Nombre de lignes : 25

Filtrer les lignes:

Chercher dans cette table

Options supplémentaires

←

T

→

id

name

username

password

utype

Éditer

Copier

Supprimer

1 ouazzine yasmine 1234567890 Doctor

Éditer

Copier

Supprimer

2 ouazzine kaoutar 23456790 Doctor

Éditer

Copier

Supprimer

3 hiba ouazzine 123456789 Doctor

Éditer

Copier

Supprimer

4 safa bechchaa 1234567890 Doctor

Éditer

Copier

Supprimer

5 karimi mohamedamine 22042002 Receptionist

Éditer

Copier

Supprimer

6 darkaoui khadija 1234567890 Receptionist

Éditer

Copier

Supprimer

7 ibtissam echchaibi 123456654321 Receptionist

Figure 34:user table

CONCLUSION

In conclusion, the implementation of the " Implementation of Patient Record Management System " project has been a journey of learning, collaboration, and practical application of our programming skills. The challenge of understanding and translating the professor's instructions into a coherent and comprehensive Java application was met with dedication and perseverance.

Throughout the development process in NetBeans IDE 19, the team invested significant time in grasping the intricacies of the project requirements, ensuring a clear understanding of each aspect described by the professor. This commitment allowed us to create an application that not only aligns with the given instructions but also adds a personal touch by naming it "yasminesafahospital" to reflect our collective effort and individual contributions.

The decision to deviate from the conventional "chu" naming convention to incorporate our names in both the project and the database serves as a testament to our commitment to personalizing and taking ownership of the project. This approach not only enhances our sense of accomplishment but also demonstrates our ability to adapt and infuse creativity into the project within the given framework.

In summary, the process of creating the patient record management system has not only deepened our understanding of Java application development but also showcased our adaptability and commitment to delivering a project that aligns with our professor's vision while reflecting our unique contributions. This experience has undoubtedly enriched our practical knowledge, preparing us for future endeavours in software development.