**PROJECT RAPPORT**

**Subject:**

**Implementation of Patient Record Management System**

**Prepared by:**

**Safa BECHCHAA & Yasmine OUAZZINE**

**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](#_Toc157280789)

[**List of Figures** 5](#_Toc157280790)

[**INTRODUCTION** 6](#_Toc157280791)

[**Chapitre I : Project** 7](#_Toc157280792)

[**1-** **Project’s Subject:** 7](#_Toc157280793)

[**2-** **Used Environment:** 7](#_Toc157280794)

[**Chapter II : Project Requirements Document:** 9](#_Toc157280795)

[**1-Introduction:** 9](#_Toc157280796)

[**2. General Features:** 9](#_Toc157280797)

[**3. User Access:** 9](#_Toc157280798)

[**4. User Interfaces:** 9](#_Toc157280799)

[**5. Database Management:** 10](#_Toc157280800)

[**6. Security:** 10](#_Toc157280801)

[**7. Languages and Tools:** 10](#_Toc157280802)

[**8. Time Constraints:** 10](#_Toc157280803)

[**9. Testing:** 10](#_Toc157280804)

[**10. Documentation(video):** 10](#_Toc157280805)

[**11. Maintenance:** 10](#_Toc157280806)

[**12. Deliverables:** 10](#_Toc157280807)

[**Chapter III : Unified Modeling Langage :** 11](#_Toc157280808)

[**1-Use Case Diagram:** 11](#_Toc157280809)

[**a-Use Case 1:** 11](#_Toc157280810)

[**a.1-Diagram:** 11](#_Toc157280811)

[**b-Use Case 2:** 12](#_Toc157280812)

[**b.1-Diagram:** 12](#_Toc157280813)

[**2-Class Diagram:** 13](#_Toc157280814)

[**a-Class :** 13](#_Toc157280815)

[**a.1-Diagram:** 13](#_Toc157280816)

[**Chapter IV : Project Implementation:** 14](#_Toc157280817)

[**1-** **Creation of Database in phpMyAdmin:** 14](#_Toc157280818)

[**2-Creation in NetBeans the project:** 14](#_Toc157280819)

[**a-Main.java :** 15](#_Toc157280820)

[**b-user.java:** 15](#_Toc157280821)

[**b.1-table user** 16](#_Toc157280822)

[**c-Login.java** 16](#_Toc157280823)

[**d-Patient.java** 17](#_Toc157280824)

[**d.1-patient table** 17](#_Toc157280825)

[**e-Doctor.java** 18](#_Toc157280826)

[**e.1-doctor table:** 18](#_Toc157280827)

[**f-createChannel : Channel.java** 19](#_Toc157280828)

[**f.1-Channel table:** 19](#_Toc157280829)

[**g-viewChannel.java:** 20](#_Toc157280830)

[**h-visit.java:** 20](#_Toc157280831)

[**h.1-visit table:** 21](#_Toc157280832)

[**i-viewDoctor.java:** 21](#_Toc157280833)

[**j-moreinformations :** 22](#_Toc157280834)

[**j.1-moreinformations table:** 22](#_Toc157280835)

[**k-viewPatientInformation.java:** 23](#_Toc157280836)

[**l- Connection Code Part:** 23](#_Toc157280837)

[**3-** **We Insert Some Informations In The Database Tables:** 25](#_Toc157280838)

[a- channel table: 25](#_Toc157280839)

[b- doctor table: 26](#_Toc157280840)

[c- patient table: 26](#_Toc157280841)

[d- visit table: 26](#_Toc157280842)

[e- moreinformations table: 26](#_Toc157280843)

[f- user table 27](#_Toc157280844)

[**CONCLUSION:** 28](#_Toc157280845)

# **List of Figures**

[Figure 1:use case 1 11](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280669)

[Figure 2:use case 2 12](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280670)

[Figure 3:class 13](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280671)

[Figure 4:database's tables 14](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280672)

[Figure 5: Project's files 14](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280673)

[Figure 6:Main.java 15](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280674)

[Figure 7:user.java 15](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280675)

[Figure 8:table user 16](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280676)

[Figure 9:login.java 16](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280677)

[Figure 10:patient.java 17](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280678)

[Figure 11:patient table 17](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280679)

[Figure 12:doctor.java 18](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280680)

[Figure 13: doctor table 18](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280681)

[Figure 14:Channel.java 19](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280682)

[Figure 15:channel table 19](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280683)

[Figure 16:viewchannel.java 20](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280684)

[Figure 17:visit.java 20](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280685)

[Figure 18: visit table 21](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280686)

[Figure 19:viewDoctor.java 21](#_Toc157280687)

[Figure 20:the combo box choices 22](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280688)

[Figure 21:moreinformations.java 22](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280689)

[Figure 22:moreinformations table 22](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280690)

[Figure 23:viewPatientInformation.java 23](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280691)

[Figure 24:connect Method 23](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280692)

[Figure 25:Patient 24](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280693)

[Figure 26:AutoID 24](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280694)

[Figure 27:patient\_table 24](#_Toc157280695)

[Figure 28:the rest of the patient\_table 25](#_Toc157280696)

[Figure 29:channel table 25](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280697)

[Figure 30:doctor table 26](#_Toc157280698)

[Figure 31:patient table 26](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280699)

[Figure 32:visit table 26](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280700)

[Figure 33:moreinformations table 26](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280701)

[Figure 34:user table 27](file:///C:\Users\gss\Documents\PROJECT%20RAPPORT%20Java%20yasminesafahospital.docx#_Toc157280702)

# **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**

## **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).

## **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.

**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 efficiently.

**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 Langage :**

## **1-Use Case Diagram:**

### **a-Use Case 1:**

#### **a.1-Diagram:**

Figure :use case 1

### **b-Use Case 2:**

#### **b.1-Diagram:**

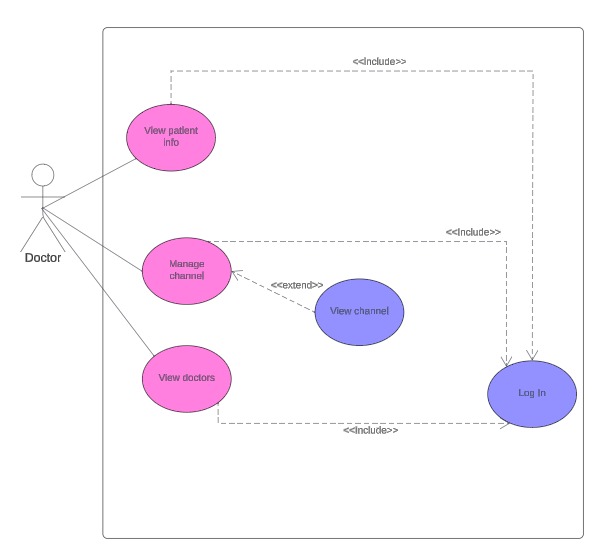


Figure :use case 2

## **2-Class Diagram:**

### **a-Class :**

#### **a.1-Diagram:**

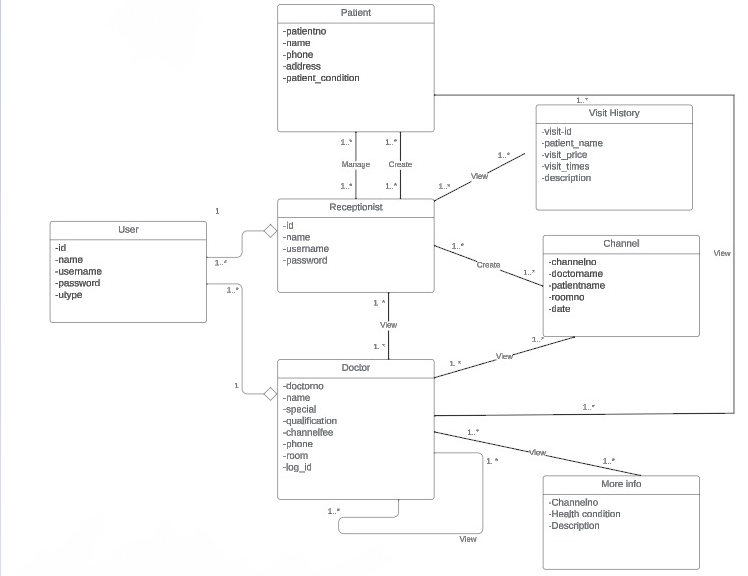


Figure :class

# **Chapter IV : Project Implementation:**

## **1-** **Creation of Database in phpMyAdmin:**

**Database:** yasminesafahospital.

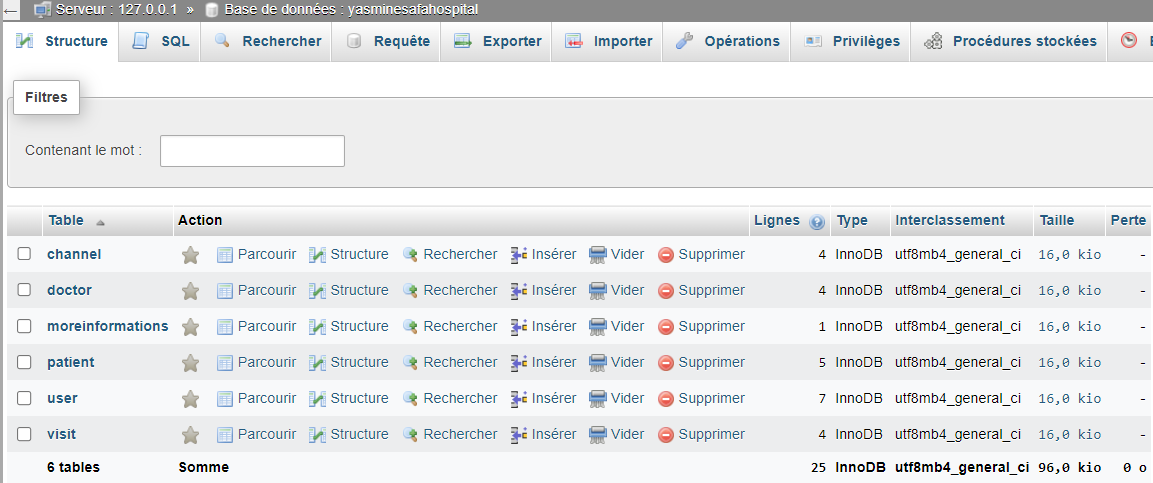
  
**Tables Created:** patient, channel, moreinformations, doctor, user, visit.

Figure :database's tables

## **2-Creation in NetBeans the project:**

**Project name:** yasminesafahospital

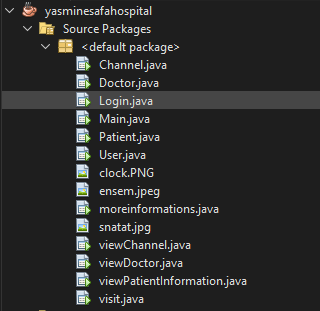
  
**Jframes created:** Channel.java , Doctor.java , Login.java , Main.java , User.java , moreinformations.java , viewChannel.java , viewDoctor.java , viewPatientInformation.java , visit.java.

Figure : 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 :Main.java

### **b-user.java:**

To create new user (doctor, receptionist)

Figure :user.java

#### **b.1-table user**

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

Figure :table user

### **c-Login.java**

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

Figure :login.java

### **d-Patient.java**

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

Figure :patient.java

#### **d.1-patient table**

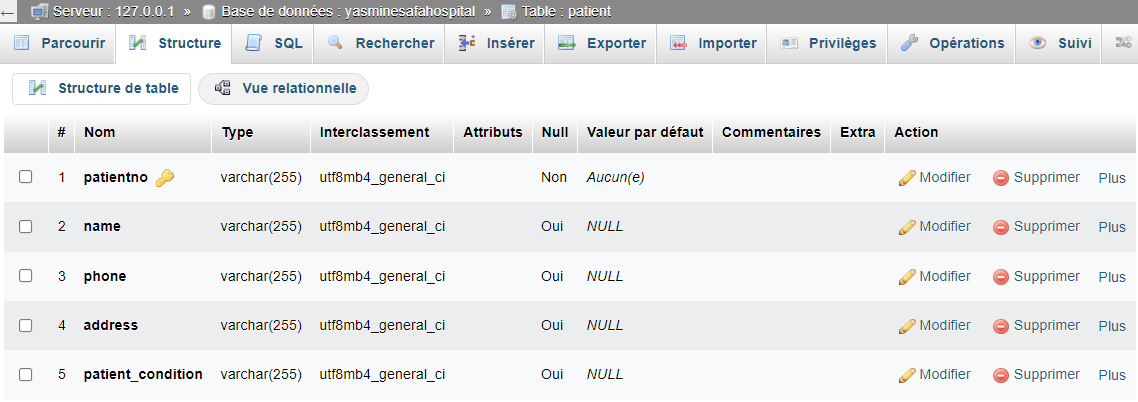


Figure :patient table

### **e-Doctor.java**

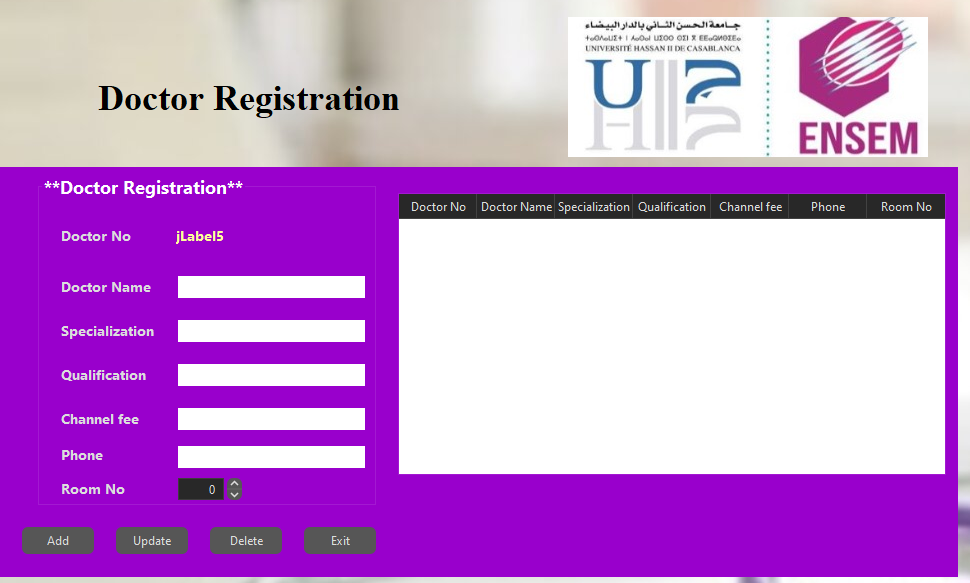
Each doctor after log in can enter his own informations in the doctor registration and the receptionist can see it in view doctor

Figure :doctor.java

#### **e.1-doctor table:**

Figure : 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 :Channel.java

#### **f.1-Channel table:**

Figure :channel table

### **g-viewChannel.java:**

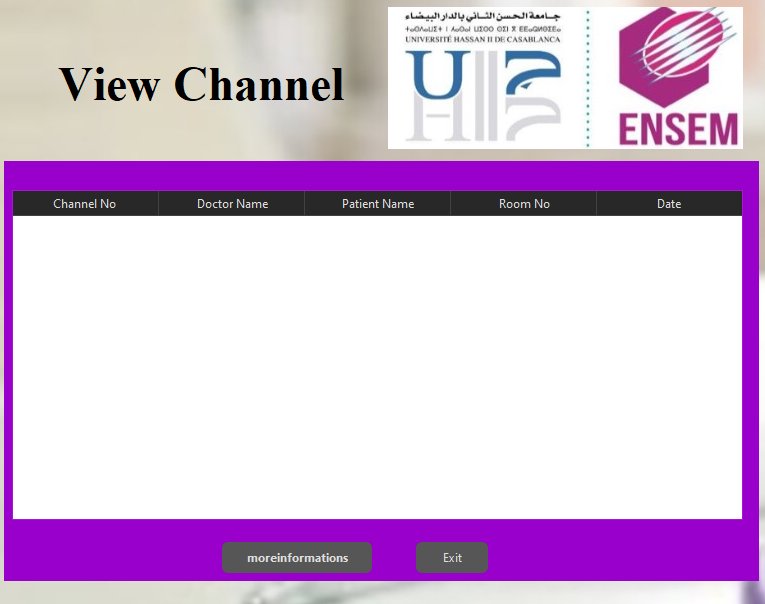
when we login as doctor we can see channel to add moreinformations to the patient in the channel chosen

Figure :viewchannel.java

### **h-visit.java:**

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

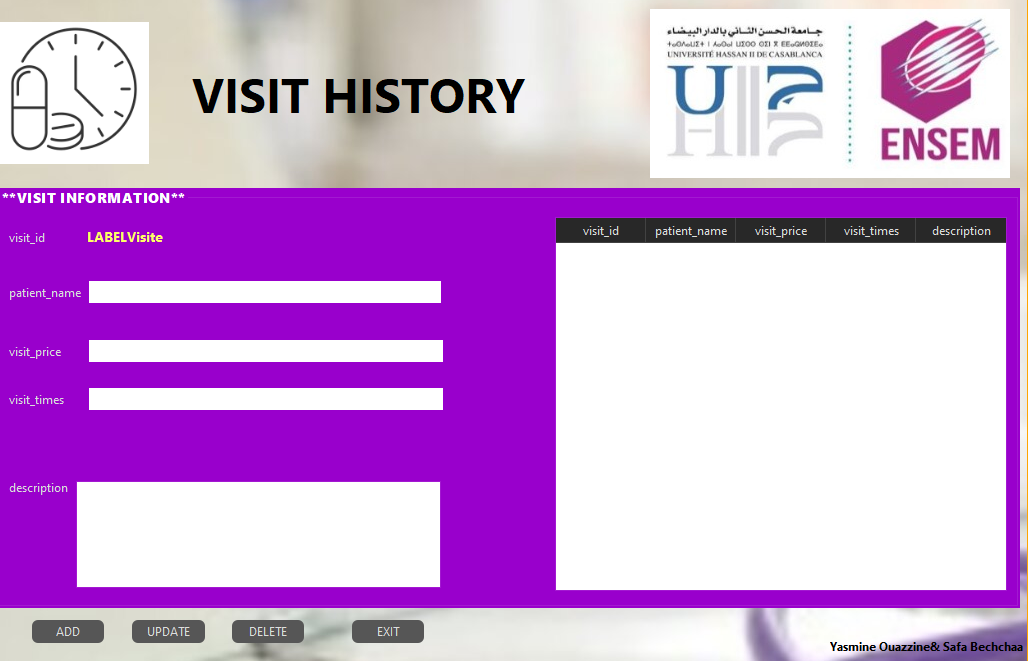
informations after the visit

Figure :visit.java

#### **h.1-visit table:**

Figure : visit table

### **i-viewDoctor.java:**

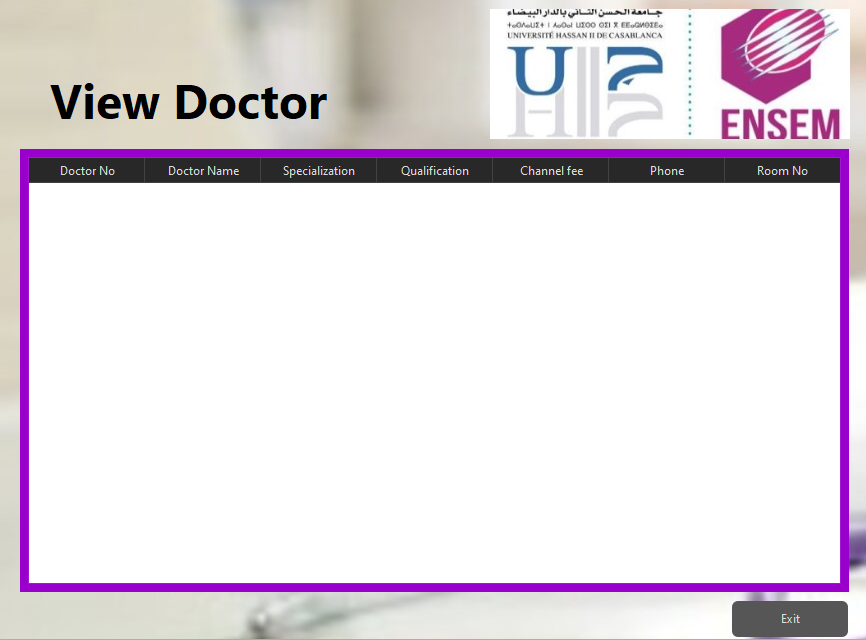
  
 The receptionist can visualize the names of doctors and there informations such as the room assigned to them and their qualifications

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

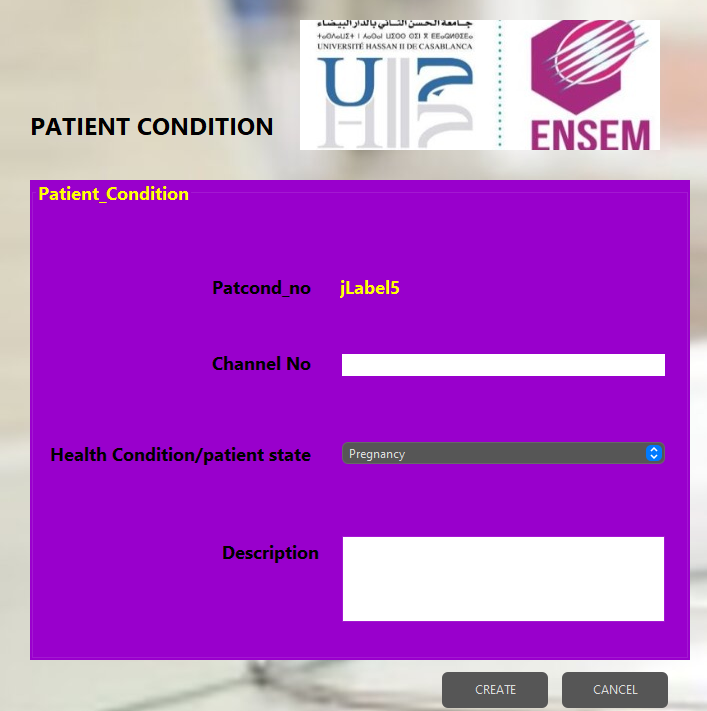
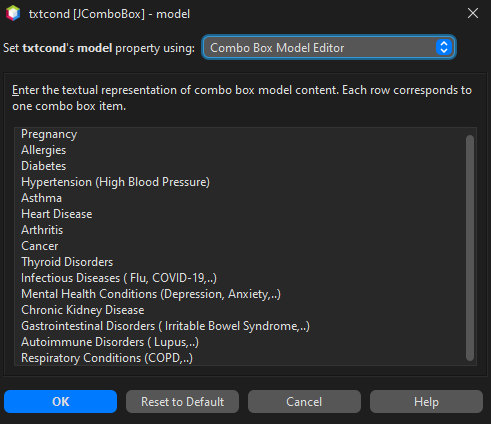


Figure :the combo box choices

Figure :moreinformations.java

#### **j.1-moreinformations table:**

Figure :moreinformations table

### **k-viewPatientInformation.java:**

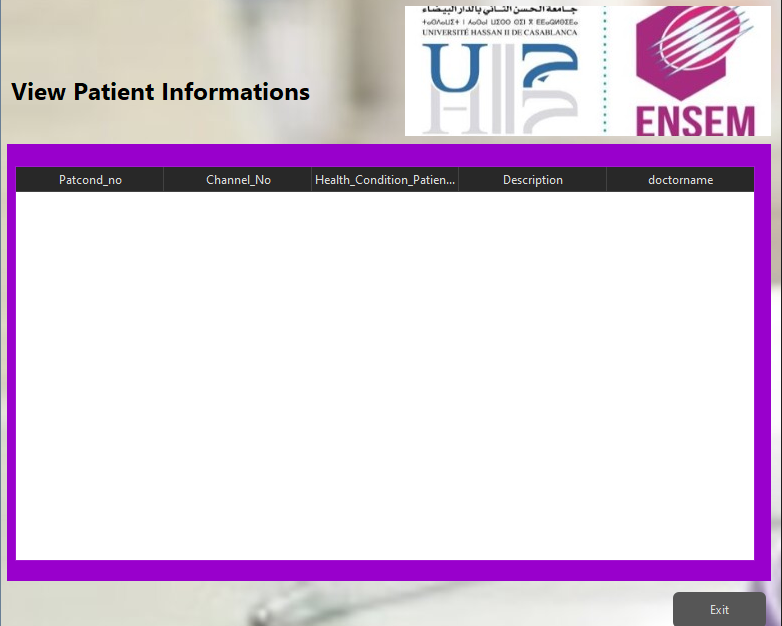
To view the moreinformations entered by a certain doctor about a certain patient only a doctor after login can visualize since the information is confidential the receptionist cannot see it.

Figure :viewPatientInformation.java

### **l- Connection Code Part:**

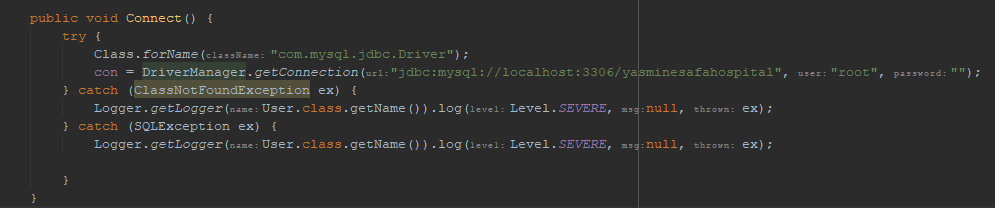
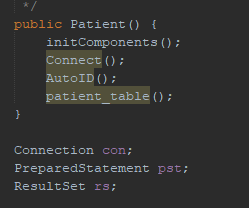
 In all the files.java we created a connection part to connected to the database we can take as an example the Patient file: Patient.java

Figure :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.



**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 :Patient

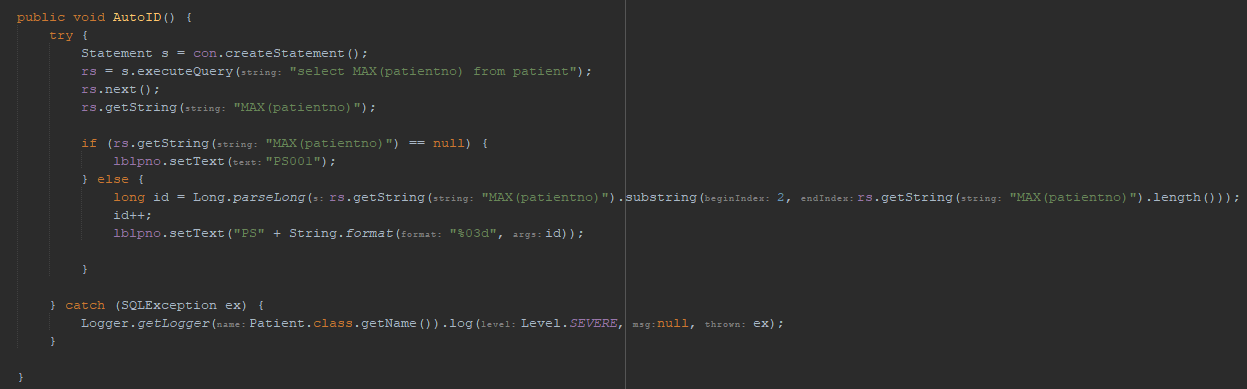


Figure :AutoID

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

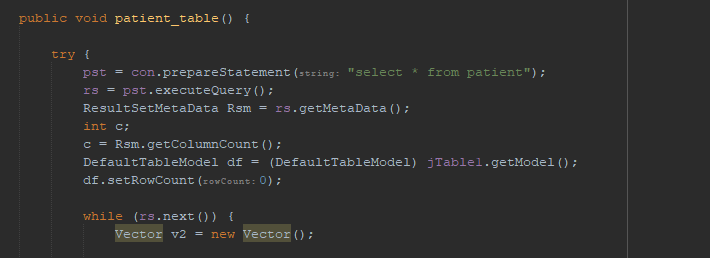


Figure :patient\_table

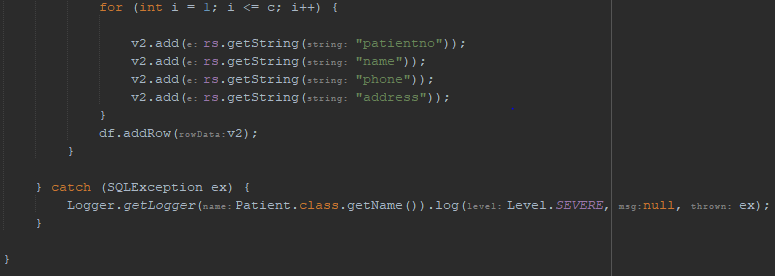


Figure 28:the rest of the patient\_table

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:**

### channel table:

Figure :channel table

1. doctor table:

Figure :doctor table

### patient table:

Figure :patient table

### visit table:

Figure :visit table

1. moreinformations table:

Figure :moreinformations table

### user table

Figure :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 you 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.