HOSPITAL MANAGEMENT SYSTEM

A mini project work report submitted to the University of Madras, Chennai in partial fulfillment of the requirements for the award of the degree of

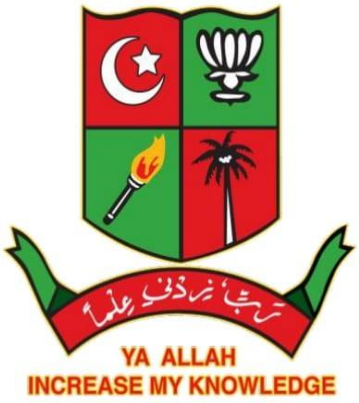
BACHELOR OF SCIENCE
IN
COMPUTER SCIENCE

By

SAMEER S
REGISTER NO: 1913181058323

Under the Guidance of

Mr. S. K. AKBAR BASHA, M.Sc., SET.,
Assistant Professor in Computer Science



DEPARTMENT OF COMPUTER SCIENCE (Shift – II)

THE NEW COLLEGE (AUTONOMOUS)

Sponsored by: The Muslim Educational Association of Southern India

(Affiliated to University of Madras & Re-Accredited by NAAC with 'A' Grade)

CHENNAI – 600 014.

APRIL – 2022

Department of Computer Science (Shift – II)

THE NEW COLLEGE (AUTONOMOUS)

Sponsored by: The Muslim Educational Association of Southern India
(Affiliated to University of Madras & Re-Accredited by NAAC with 'A' Grade)
CHENNAI – 600 014.

BONAFIDE CERTIFICATE

Register Number	**:**	**1913181058323**
Name	**:**	**SAMEER S**
Title	**:**	**HOSPITAL MANAGEMENT SYSTEM**
Subject Code	**:**	**17BJMQ601**
Subject Name	**:**	**MINI PROJECT**

Certified to be the Bonafide Record work done in the Computer Science Laboratory of The New College (Autonomous), Chennai, during the academic year 2021–2022 by the above candidate in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF SCIENCE IN COMPUTER SCIENCE

Project Guide	**Co-Ordinator**	**Head of the Department**
Mr. S. K. AKBAR BASHA	**Dr. M. WASIM RAJA**	**Dr. M. HAKKIM DIVAN MYDEEN**

Submitted for the University Viva Voce examination held at The New College (Autonomous), Chennai - 14 on _____.

Internal Examiner **External Examiner**

ACKNOWLEDGEMENTS

First and foremost, I thank the **Almighty** for providing me the strength and courage to complete this project work successfully. I take this opportunity to express my sincere thanks who made it possible.

I express my sincere gratitude to our college management, for providing excellent infrastructure. I thank our principal, **DR. S. BASHEER AHAMED, M.A., M.Phil., B.Ed., Ph.D., THE NEW COLLEGE (AUTONOMOUS), Chennai-14** for providing me an opportunity to complete the project study.

I wish to acknowledge my ineptness and express my gratitude to our esteemed **Head of Department, Dr. M. HAKKIM DIVAN MYDEEN, M.Sc., M.Phil., Ph.D.,** and **Co-Ordinator, Dr. M. WASIM RAJA, M.Sc., M.Phil., Ph.D., PGDCA.,** for their motivation and for providing me all facilities in all activities in completing this project.

I would like to express my heartfelt and sincere thanks to my guide **Mr. S. K. AKBAR BASHA, M.Sc., SET.,** Department of Computer Science, THE NEW COLLEGE (AUTONOMOUS), Chennai–14, under whose able guidance, valuable suggestions and finely help the study was carried out. It is my pleasure to acknowledge my deep sense of gratitude to him for opening my eyes on project work in computer science.

I take this opportunity to thank my friends and family members whose values support and moral guidance went a long way in helping me in accomplishing my objective.

SAMEER S

ABSTRACT

The purpose of the project entitled as “HOSPITAL MANAGEMENT SYSTEM” is to computerize the Front Office Management of Hospital to develop software which is user friendly, simple, fast and cost-effective. It does with the collections of patient’s information, diagnosis, details, etc. Traditionally, it was done manually. The main function of the system is to register and store patient details and retrieve these details as and when required, and also to manipulate these details meaningfully. System input contains patient details, diagnosis details; while system output is to get these details on the screen. This project will automate the daily operations of LIFE LINE hospital. The project keeps track of the patient details. It also take care of the ward, medical, diagnosis details. This project deals with processing of each and every department in the hospital. This project sincerely aims to reduce the manual processing of each department.

LIST OF CONTENT

a) ABSTRACT

1. INTRODUCTION

1.1. Introduction ---------- 7

1.2. Problem introduction ---------- 9

1.3. Objective ---------- 11

1.4. Modules in the project ---------- 13

2. REQUIREMENT SPECIFICATION

2.1. Introduction ---------- 22

2.2. Hardware requirements ---------- 22

2.3. Software requirements ---------- 23

3. ANALYSIS

3.1. Existing system ---------- 25

3.2. Proposed system ---------- 26

3.3. Feasibility study ---------- 26

3.4. Software specification ---------- 28

4. SYSTEM TESTING ---------- 46

5. ER DIAGRAM ---------- 51

6. TABLE STRUCTURE ---------- 53

7. SAMPLE SCREENSHOT ---------- 60

8. SOURCE CODE ---------- 96

9. CONCLUSION---------- 116

10. FUTURE ENHANCEMENT ---------- 118

11. BIBLIOGRAPHY ---------- 120

CHAPTER 1

INTRODUCTION

1.1. Introduction:

The project HOSPITAL MANAGEMENT SYSTEM includes registration of patients, storing their details into the system, and also computerized. The software has the facility to give a unique id for every patient and stores the details of every patient and the staff automatically. It includes a search facility to know the current status of each room. User can search automatically of a doctor and the details of a patient using the ID.

The Hospital Management System can be entered using a username and password. With the large amounts of data, people involved and innumerable processes, a hospital is definitely an ideal candidate for data management software. If hospitals are to run efficiently, provide top line care, ensure patient and other data confidentiality, and work seamlessly – they cannot hope to do so without an effective Hospital Management System. Reduced human intervention for paperwork, less paperwork, reduced staff headcount for jobs that can be easily managed within the HMS, speedier processes, reduction of errors, and data privacy and safety – are just some of the benefits of a Hospital Management System.

For the hospitals, HMS translates to being able to track patient history, provide better care, keep track of appointments, save patient insurance and enable doctors and clinicians to check patient history, maintain patient care continuity, and save time and effort on unnecessary tedious manual tasks. This ensures that even if a patient visits after a long break, the patient and hospital will not require going through the registration process again.

Hospital records are easily audited and kept compliant with policies and laws. In addition, the Hospital Management System is cost effective – it reduces the need for staff to manage manual entries, manage paperwork, and ensure accurate filing. This in turn significantly reduces the possibility of human error, which can prove costly on many counts. Another significant benefit/factor of the HMS is it is customization to the needs and requirements of a particular hospital/healthcare facility.

Hospital workflows are done within twenty-four hours. Due to this reason, the hospitals need efficient management. According to Toussaint (2015), hospitals can't improve without better management systems. In Toussaint's perspective, management is a significant part of today's cost and quality crisis in health care. This is the reason why suitable hospital needs and appropriate medical management must be present to deliver applicable healthcare facilities. However, there are still several hospitals in the country that uses paper works in the management.

In line with this, the researchers found out that the Malita District Hospital located at National Highway Road, Poblacion, Malita, Davao Occidental is using Microsoft Excel format in most of their computer transactions. Though they have computers on each department, the work process is still laborious and time consuming. The employees still need to check the excel files every time there are inquiries about hospital records without proper system.

Each staff will look through their Excel files or printed files for each patient profile since all the computers are not connected to each other. Though there are existing computers, updating of information is done separately. Due to this reason, recording and maintaining all the records is highly unreliable, incompetent and error-prone.

1.2 Problem Introduction

1.2. Problem Introduction:

Lack of immediate retrievals:

The information is very difficult to retrieve and to find particular information like - E.g. - To find out about the patient's history, the user has to go through various registers. This results in in convenience and wastage of time.

Lack immediate information storage:

The information generated by various transactions takes time and efforts to be stored at right place.

Lack prompt updating:

Various changes to information like patient details or immunization details of child are difficult to make as paper work is involved.

Error prone manual calculation:

Manual calculations are error prone and take a lot of time this may result in incorrect information. For example storing of patient's ID based on various treatments.

Preparation of accurate and prompt reports:

This becomes a difficult task as information is difficult to collect from various register.

1.3 Objective

1.3 Objective:

The Project "Hospital Management System" is a software to automate most of the day to day activities of the hospital.

- Define hospital
- Recording information about the Patient that come
- Recording information related to diagnosis given to Patient
- Keeping the record of the immunization provided to children/parent
- Keeping information about various disease

Scope of the Project:

1) Information about Patients is done by just writing the Patients name, age and gender. Whenever the Patient comes up his information is stored freshly.
2) Diagnosis information to patients is generally recorded on the document, which contains Patient information. It is destroyed after some time period to decrease the paper load in the office.
3) Immunization records of children are maintained in pre-formatted sheets, which are kept in a file.
4) Information about various diseases is not kept as any document. Doctors themselves do this job by remembering various medicines.

All this work is done manually by the receptionist and other operational staff and lot of papers are needed to be handled and taken care of. Doctors have to remember various medicines available for diagnosis and sometimes miss better alternatives as they can't remember them at that time.

1.4 Modules in the project

1.4. Modules in the project:

A *module* is a collection of source files and build settings that allow you to divide your project into discrete units of functionality. The project can have one or many modules, and one module may use another module as a dependency. You can independently build, test, and debug each module. Additional modules are often useful when creating code libraries within your own project or when you want to create different sets of code and resources for different device types, such as phones and wearable, but keep all the files scoped within the same project and share some code.

The entire project mainly consists of 9 modules, which are

- Login module
- Sign-up module
- Forget password module
- Home module
- Add new patient module
- Add diagnosis module
- Full history of the patient module
- Update patient record module
- Hospital information module

1.4.1 Login module:

This module will be accessible, when the user update the correct “Email” and “Password”. In general computer usage, login is the procedure used to get access to an operating system or application, usually in a remote computer. Almost always a login requires that the user have (1) a user ID and (2) a password. The Login Module is a portal module that allows users to type a email and password to log in. You can add this module on any module tab to allow users to log in to the system. Often, the user email ID must conform to a limited length such as eight characters and the password must contain at least one digit and not match a natural language word. The user email ID can be freely known and is visible when entered at a keyboard or other input device. The password must be kept secret (and is not displayed as it is entered). Some Web sites require users to register in order to use the site; registered users can then enter the site by logging on.

1.4.2. Sign-up module:

A sign-up page (also known as a registration page) enables users and organizations to independently register and gain access to your system. It is common to have multiple sign-up pages depending on the types of people and organizations you want to register. You will learn about the different types of sign-up pages, how to configure them and related functionality. A sign-up form is a web page, popup, or modal where users enter the information required to access that website's services. A login is a set of credentials used to authenticate a user. Most often, these consist of a username and password. However, a login may include other information, such as a PIN number, passcode, or passphrase. The information collected is determined by the nature of the website and the services it offers. Most sign-up forms require a name, email address, username, and password.

In this module, you will have some data field to fill for sign-in process. there are,

- Name
- Email
- Password
- Security question
- Answer for security question
- Address

1.4.3. Forget password module:

A Password is a word, phrase, or string of characters intended to differentiate an authorized user or process (for the purpose of permitting access) from an unauthorized user, or put another way a password is used to prove one's identity, or authorize access to a resource. Most programs that require a user to log in provide a link titled forgot password or another similar phrase feature. This module allows users who have forgotten their password to unlock, retrieve, or reset it, usually by answering account secret questions.
At first user have to write down the respective email and search throughout the database. Once the respective email is found, the security question will pop-up. After that the user have to fill the particular for corresponding question.

In this module, there will be few data field which user have to fill for forget password process. They are,

- Email
- Security question
- Answer
- New password

1.4.4. Home module:

The Home modules is the main context of the project. This module will be accessible, when the user update the correct “Username” and “Password”. Depending on the hospital management system software features, it can deal with a lot of tasks. It helps to outline and implement policies, guarantee communication and coordination between employees, automate routine tasks, design the patient-oriented workflow, advertise services, manage human and financial resources and provide them the uninterrupted supply chain. The components of the hospital management system can be chosen and combined in the general system that meet the needs and norms of the healthcare industry as well as quality standards.

In this module, there will be few facility that admin\user can utilize them. They are,

- Rotational arrow
- Add new patient record
- Add diagnosis information
- Full history of the patient
- Update patient record
- Hospital information
- Logout

1.4.5. Add new patient record module:

It is used to control patient flow. It can be used to register them, get the data to the patient's health condition, view the treatment and check the medical history and reports. Patients have their own system accounts where the list of various action can be performed. They are able to make online data storing. Since the hospital management system is patient-oriented, the treatment process can be less stressful. Doctor have more time for examination and interaction with patient. In addition, all the requested information can be received online.

In this module, some data has to be filled to save the individual patient's detail. They are,

- Patient ID
- Name
- Contact No
- Age
- Gender
- Blood group
- Address
- Any major disease earlier

1.4.6. Add diagnosis module:

The process of identifying a disease, condition, or injury from its signs and symptoms. A health history, physical exam, and tests, such as blood tests, imaging tests, and biopsies, may be used to help make a diagnosis. This modules contains the details of the patient. Diagnosis is the art or act of identifying a disease from its signs and symptoms. Each patient will be allocated a unique ID, by that we could search up particular data and update the additional information which includes diagnosis, medicines, ward, etc.

In this module, some data has to be filled to save the individual patient's detail. They are,

- Patient ID (to search up the particular ID)
- Sympton's
- Diagnosis
- Medicines
- Ward required
- Type of ward (Will appear when "Ward required" is tick)

1.4.7. Full history of the patient module:

A medical history includes an inquiry into the patient's medical history, past surgical history, family medical history, social history, allergies, and medications the patient is taking or may have recently stopped taking.

1.4.8. Update patient record module:

This modules contains the details of the patient. Each patient will be allocated a unique ID and it could be called by clicking the search button. After clicking the button, the details of individuals will be displayed in the particulars fields. This module will use unique patient ID to search their particular details and helps the patient/hospital member to re-correct their data.

- Patient ID
- Name
- Contact No
- Age
- Gender
- Blood group
- Address
- Any major disease earlier

1.4.9. Hospital information module:

This module shows the information about the hospital.

CHAPTER 2

SOFTWARE SPECIFICATION

2.1 Introduction:

To be used efficiently, all computer software needs certain hardware components or the other software resources to be present on a computer. These pre-requisites are known as (computer) system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements: minimum and recommended. With increasing demand for higher processing power and resources in newer versions of software, system requirements tend to increase over time. Industry analysts suggest that this trend plays a bigger part in driving upgrades to existing computer systems than technological advancements.

2.2 Hardware requirements:

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware. A hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatibility and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements

Hardware requirements for present project:

- ✧ System : AMD PRO.
- ✧ Hard Disk : 120 GB.
- ✧ Monitor : STANDARD LCD/LED MONITOR
- ✧ Input Devices : Keyboard, Mouse
- ✧ Ram : 4 GB.

2.3 software requirements:

Software Requirements deal with defining software resource requirements and pre-requisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or pre-requisites are generally not included in the software installation package and need to be installed separately before the software is installed.

Software requirements for present project:

- Operating system : Windows 10
- Front end Language : Java with swing
- Tool : Netbeans
- Database : MySQL

CHAPTER 3

ANALYSIS

3.1. Existing system:

Hospitals currently use a manual system for the management and maintenance of critical information. The current system requires numerous paper forms, with data stores spread through out the hospital management infrastructure. Often information is incomplete or does not follow management standards. Forms are often lost in transit between departments requiring a comprehensive auditing process to ensure that no vital information is lost. Multiple copies of the same information exist in the hospital and may lead to inconsistencies in data in various data stores.

3.2 Proposed system:

The Hospital Management System is designed for any hospital to replace their existing manual paper based system. The new system is to control the information of patients. Room availability, staff and operating room schedules and patient invoices. These services are to be provided in an efficient, cost effective manner, with the goal of reducing the time and resources currently required for such tasks.

3.3. Feasibility study:

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are:

3.3.1. Economic feasibility:

This study is carried out to check the economic impact will have on the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products have to be purchased.

3.3.2. Technical feasibility:

This study is carried out to check the technical feasibility, that is the technical requirements of the system. Any system developed must not have a high demand on the available available technical resources. This will lead to high demands being placed on the client. The developed systems must have a modest requirement, as only minimal or mull changes for the implementing this system.

3.3.3. Operational feasibility:

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the insers solely depends om the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which in welcomed, as he is the final user of the system

3.4. Software specification:

3.4.1. Java

JAVA was developed by James Gosling at **Sun Microsystems** Inc in the year **1995**, later acquired by Oracle Corporation. It is a simple programming language. Java makes writing, compiling, and debugging programming easy. It helps to create reusable code and modular programs.

Java is a class-based, object-oriented programming language and is designed to have as few implementation dependencies as possible. A general-purpose programming language made for developers to *write once run anywhere* that is compiled Java code can run on all platforms that support Java. Java applications are compiled to byte code that can run on any Java Virtual Machine. The syntax of Java is similar to c/c++.

History of java:

Java's history is very interesting. It is a programming language created in 1991. James Gosling, Mike Sheridan, and Patrick Naughton, a team of Sun engineers known as the **Green team** initiated the Java language in 1991. **Sun Microsystems** released its first public implementation in 1996 as **Java 1.0**. It provides no-cost -run-times on popular platforms. Java1.0 compiler was re-written in Java by Arthur Van Hoff to strictly comply with its specifications. With the arrival of Java 2, new versions had multiple configurations built for different types of platforms.

In 1997, Sun Microsystems approached the ISO standards body and later formalized Java, but it soon withdrew from the process. At one time, Sun made most of its Java implementations available without charge, despite their proprietary software status. Sun generated revenue from Java through the selling of licenses for specialized products such as the Java Enterprise System.

On November 13, 2006, Sun released much of its Java virtual machine as free, open-source software. On May 8, 2007, Sun finished the process, making all of its JVM's core code available under open-source distribution terms.

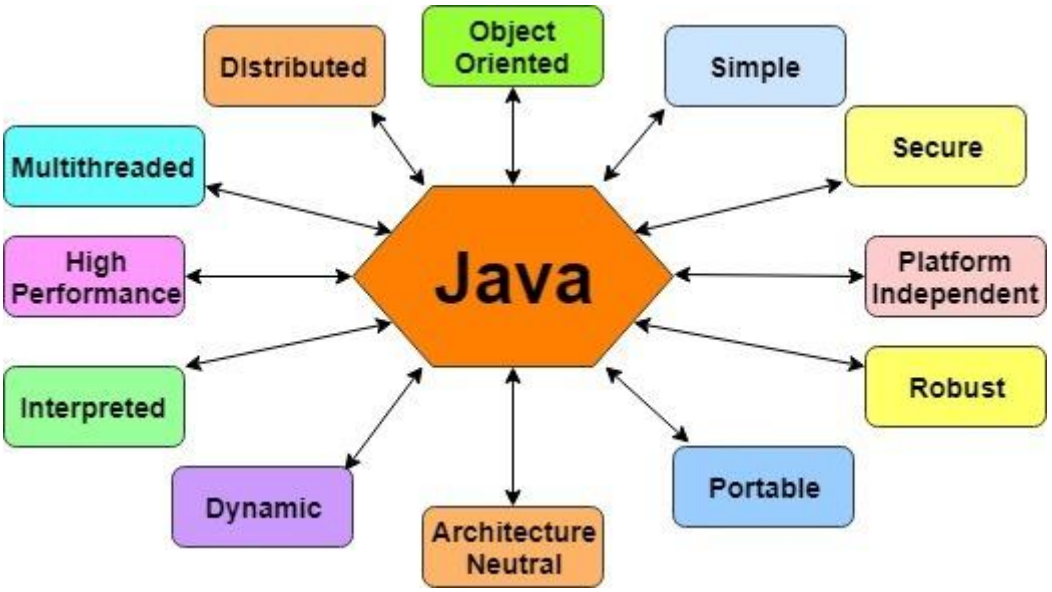
The principles for creating java were simple, robust, secured, high performance, portable, multi-threaded, interpreted, dynamic, etc. In 1995 Java was developed by **James Gosling**, who is known as the Father of Java. Currently, Java is used in mobile devices, internet programming, games, e-business, etc.

Java programming language is named JAVA. Why?

After the name OAK, the team decided to give a new name to it and the suggested words were Silk, Jolt, revolutionary, DNA, dynamic, etc. These all names were easy to spell and fun to say, but they all wanted the name to reflect the essence of technology. In accordance with James Gosling, **Java** the among the top names along with **Silk**, and since java was a unique name so most of them preferred it.

Java is the name of an **island** in Indonesia where the first coffee(named java coffee) was produced. And this name was chosen by James Gosling while having coffee near his office. Note that Java is just a name, not an acronym.

Primary/Main Features of Java:



1. **Platform Independent:** Compiler converts source code to bytecode and then the JVM executes the bytecode generated by the compiler. This bytecode can run on any platform be it Windows, Linux, macOS which means if we compile a program on Windows, then we can run it on Linux and vice versa. Each operating system has a different JVM, but the output produced by all the OS is the same after the execution of bytecode. That is why we call java a platform-independent language.

2. Object-Oriented Programming Language: Organizing the program in the terms of collection of objects is a way of object-oriented programming, each of which represents an instance of the class.

The four main concepts of Object-Oriented programming are:

- Abstraction
- Encapsulation
- Inheritance
- Polymorphism

2. **Simple:** Java is one of the simple languages as it does not have complex features like pointers, operator overloading, multiple inheritances, Explicit memory allocation.

3. **Robust:** Java language is robust which means reliable. It is developed in such a way that it puts a lot of effort into checking errors as early as possible, that is why the java compiler is able to detect even those errors that are not easy to detect by another programming language. The main features of java that make it robust are garbage collection, Exception Handling, and memory allocation.

5. Secure: In java, we don't have pointers, so we cannot access out-of-bound arrays i.e it shows **ArrayIndexOutOfBound Exception** if we try to do so. That's why several security flaws like stack corruption or buffer overflow are impossible to exploit in Java.

6. **Distributed:** We can create distributed applications using the java programming language. Remote Method Invocation and Enterprise Java Beans are used for creating distributed applications in java. The java programs can be easily distributed on one or more systems that are connected to each other through an internet connection.

7. **Multithreading:** Java supports multithreading. It is a Java feature that allows concurrent execution of two or more parts of a program for maximum utilization of CPU.

8. **Portable:** As we know, java code written on one machine can be run on another machine. The platform-independent feature of java in which its platform-independent bytecode can be taken to any platform for execution makes java portable.

9. **High Performance:** Java architecture is defined in such a way that it reduces overhead during the runtime and at some time java uses Just In Time (JIT) compiler where the compiler compiles code on-demand basics where it only compiles those methods that are called making applications to execute faster.

10. **Dynamic flexibility:** Java being completely object-oriented gives us the flexibility to add classes, new methods to existing classes and even create new classes through sub-classes. Java even supports functions written in other languages such as C, C++ which are referred to as native methods.

11. **Sandbox Execution:** Java programs run in a separate space that allows user to execute their applications without affecting the underlying system with help of a bytecode verifier. Bytecode verifier also provides additional security as its role is to check the code for any violation of access.

12. Write Once Run Anywhere: As discussed above java application generates a '.class' file which corresponds to our applications(program) but contains code in binary format. It provides ease t architecture-neutral ease as bytecode is not dependent on any machine architecture. It is the primary reason java is used in the enterprising IT industry globally worldwide.

13. Power of compilation and interpretation: Most languages are designed with purpose either they are compiled language or they are interpreted language. But java integrates arising enormous power as Java compiler compiles the source code to bytecode and JVM executes this bytecode to machine OS-dependent executable code.

3.4.2. MySQL

MySQL is developed, distributed, and supported by Oracle Corporation: MySQL s database system used on the web it runs on a server. MySQL is ideal for both small and large applications. It is very fast, reliable, and easy to use. It supports standard SQL MySQL can be compiled un a number of platforms

The data in MySQL is stored in tables. A table is a collection of related data, and it consists of columns and rows Databases are useful when storing information categorically.

- MySQL is a relational database management system
- MySQL is open-source
- MySQL is free
- MySQL is ideal for both small and large applications
- MySQL is very fast, reliable, scalable, and easy to use
- MySQL is cross-platform
- MySQL is compliant with the ANSI SQL standard
- MySQL was first released in 1995
- MySQL is developed, distributed, and supported by Oracle Corporation
- MySQL is named after co-founder Monty Widenius's daughter: My

FEATURES OF MySQL:

Internals and portability:

- .Written in C and C++
- Tested with a broad range of different compilers
- Works on many different platforms.
- Tested with Purify (a commercial memory leakage detector) as well as with Val grind, a GPL tool
- Uses multi-layered server design with independent modules.

Security:

- A privilege and password system that is very flexible and secure, and that enables host-based verification
- Password security by encryption of all password traffic when you connect to a server.

Scalability and Limits

- Support for large databases. We use MySQL Server with databases that contain 50 million records. We also know of users who use MySQL Server with 200,000 tables and about 5.000.000,000 rows.
- Support for up to 64 indexes per table (32 before MySQL 41.2). Each index may consist of 1 to 16 columns or parts of columns. The maximum index width is 767 bytes for **InnoDB** tables, or 1000 for **MyISAM**: before MySQL. 4.1.2, the limit is 500 bytes, An index may use a prefix of a column for **CHAR, VARCHAR, BLOB,** or **TEXT** column types,

MySQL workbench:

MySQL Workbench is a unified visual database designing or graphical user interface tool used for working with database architects, developers, and Database Administrators. It is developed and maintained by Oracle. It provides SQL development, data modeling, data migration, and comprehensive administration tools for server configuration, user administration, backup, and many more. We can use this Server Administration for creating new physical data models, E-R diagrams, and for SQL development (run queries, etc.). It is available for all major operating systems like Mac OS, Windows, and Linux. MySQL Workbench fully supports MySQL Server version v5.6 and higher.

MySQL Workbench covers five main functionalities, which are given below:

- SQL Development
- Data Modelling
- Server Administration
- Data Migration
- MySQL Enterprise Supports

SQL Development:

This functionality provides the capability that enables you to execute SQL queries, create and manage connections to the database Servers with the help of built-in SQL editor.

Data Modelling (Design):

This functionality provides the capability that enables you to create models of the database Schema graphically, performs reverse and forward engineering between a Schema and a live database, and edit all aspects of the database using the comprehensive Table editor. The Table editor gives the facilities for editing tables, columns, indexes, views, triggers, partitioning, etc.

Server Administration:

This functionality enables you to administer MySQL Server instances by administering users, inspecting audit data, viewing database health, performing backup and recovery, and monitoring the performance of MySQL Server.

Data Migration:

This functionality allows you to migrate from Microsoft SQL Server, SQLite, Microsoft Access, PostgreSQL, Sybase ASE, SQL Anywhere, and other RDBMS tables, objects, and data to MySQL. It also supports migrating from the previous versions of MySQL to the latest releases.

MySQL Enterprise Supports:

This functionality gives the support for Enterprise products such as MySQL firewall, MySQL Enterprise Backup, and MySQL Audit.

CONNECTIVITY:

- Clients can connect to MySQL Server using several protocols: • Clients can connect using TCP IP sockets on any platform.

- On Windows systems in the NT family (NT, 2000, XP, 2003, or Vista), clients can connect using named pipes if the server is started with the --enable-named pipe option. In MySQL 41 and higher, Windows servers also support shared-mumory connections if started with the-shared-memory option. Clients can connect through shared memory by using the-protocol-memory option.

- On UNIX systems, clients can connect using Unix domain socket files.

LOCALIZATION:

- The server can provide error messages to clients in many languages.
- All data is saved in the chosen character set

CLIENTS AND TOOLS:

- MySQL includes several client and utility programs. These include both command such line programs such as mysqldump and mysqladmin, and graphical programs as MySQL Workbench.
- MySQL Server has built-in support for SQL statements to check optimize, and repair tables. These statements are available from the command line through the mysqlcheck client. MySQL also includes myisamchk, a very fast command-line utility for performing these operations on MyISAM tables.
- MySQL programs can be invoked with the help or 2 option to obtain online assistance.

Why we use MySQL?

- Leading open source RDBMS
- Ease of use-No frills
- Fast
- Robust
- Security
- Multiple OS support
- Free
- Technical support
- Support large database-up to 50 million rows, file size limit up to 8 Million TB

3.4.3. Netbeans:

NetBeans is an open-source integrated development environment (IDE) for developing with Java, PHP, C++, and other programming languages. NetBeans is also referred to as a platform of modular components used for developing Java desktop applications.

NetBeans is coded in Java and runs on most operating systems with a Java Virtual Machine (JVM), including Solaris, Mac OS, and Linux. The IDE is designed to limit coding errors and facilitate error correction with tools such as the NetBeans FindBugs to locate and fix common Java coding problems and Debugger to manage complex code with field watches, breakpoints and execution monitoring. Although the NetBeans IDE is designed specifically for Java developers, it also supports C/C++, PHP, Groovy, and HTML5 in addition to Java, JavaScript and JavaFX.

Tools and capabilities of the NetBeans IDE include a feature-rich text editor with refactoring tools and code templates, high level and granular views of applications, a drag and drop GUI design, and versioning using out-of-the-box integration with tools such as Git. The NetBeans IDE can run on any operating system that supports a compatible JVM including Linux, Windows and OS X.

The underlying NetBeans platform supports creation of new applications and further development of existing applications using modular software components. As an application running on the NetBeans Platform, the NetBeans IDE itself is extensible and can be extended to support new languages.

The IDE and Platform were converted to open source by Sun Microsystems in 2000. Oracle continues to sponsor the NetBeans project since acquiring Sun in 2010.

NetBeans manages the following platform features and components:

- User settings
- Windows (placement, appearance, etc.)
- NetBeans Visual Library
- Storage
- Integrated development tools
- Framework wizard

NetBeans uses components, also known as modules, to enable software development. NetBeans dynamically installs modules and allows users to download updated features and digitally authenticated upgrades.

NetBeans IDE modules include NetBeans Profiler, a Graphical User Interface (GUI) design tool, and NetBeans JavaScript Editor.

NetBeans framework reusability simplifies Java Swing desktop application development, which provides platform extension capabilities to third-party developers.

History of Netbeans:

NetBeans began in 1996 as Xelfi (word play on *Delphi*), a Java IDE student project under the guidance of the Faculty of Mathematics and Physics at Charles University in Prague. In 1997, Roman Staněk formed a company around the project and produced commercial versions of the NetBeans IDE until it was bought by Sun Microsystems in 1999. Sun open-sourced the NetBeans IDE in June of the following year. Since then, the NetBeans community has continued to grow. In 2010, Sun (and thus NetBeans) was acquired by Oracle Corporation. Under Oracle, NetBeans had to find some synergy with JDeveloper, a freeware IDE that has historically been a product of the company, by 2012 both IDEs were rebuilt around a shared codebase - the NetBeans Platform. In September 2016, Oracle submitted a proposal to donate the NetBeans project to the Apache Software Foundation, stating that it was "opening up the NetBeans governance model to give NetBeans constituents a greater voice in the project's direction and future success through the upcoming release of Java 9 and NetBeans 9 and beyond". The move was endorsed by Java creator James Gosling. The project entered the Apache Incubator in October 2016.

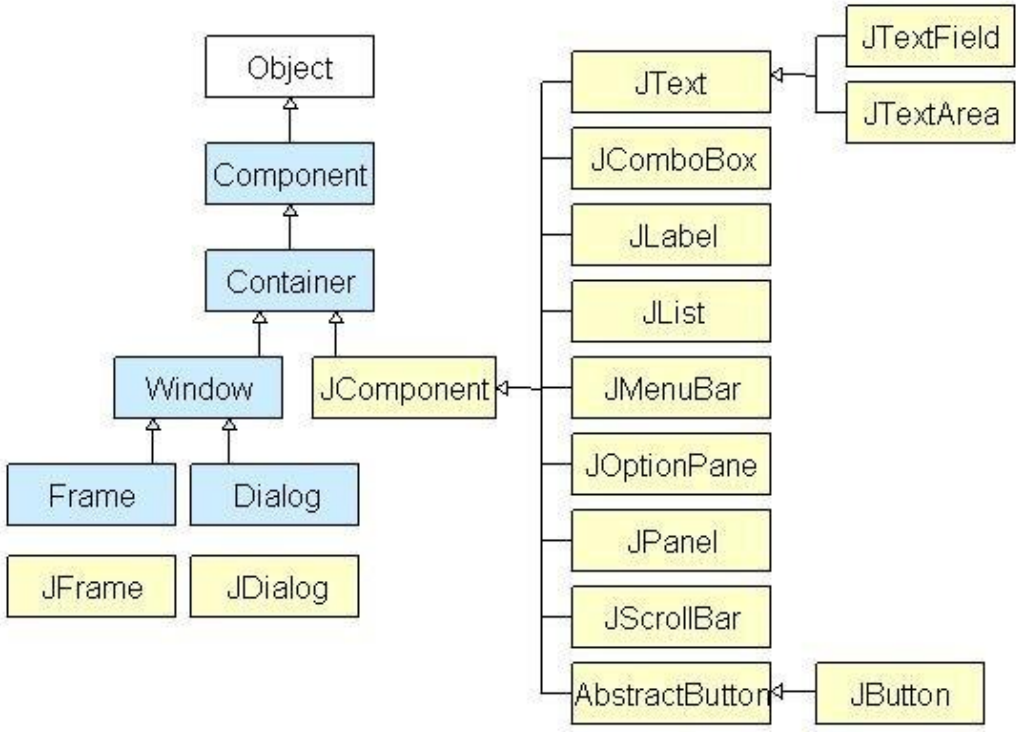
Modularity:

All the functions of the IDE are provided by modules. Each module provides a well-defined function, such as support for the Java language, editing, or support for the CVS versioning system, and SVN. NetBeans contains all the modules needed for Java development in a single download, allowing the user to start working immediately. Modules also allow NetBeans to be extended. New features, such as support for other programming languages, can be added by installing additional modules. For instance, Sun Studio, Sun Java Studio Enterprise, and Sun Java Studio Creator from Sun Microsystems are all based on the NetBeans IDE.

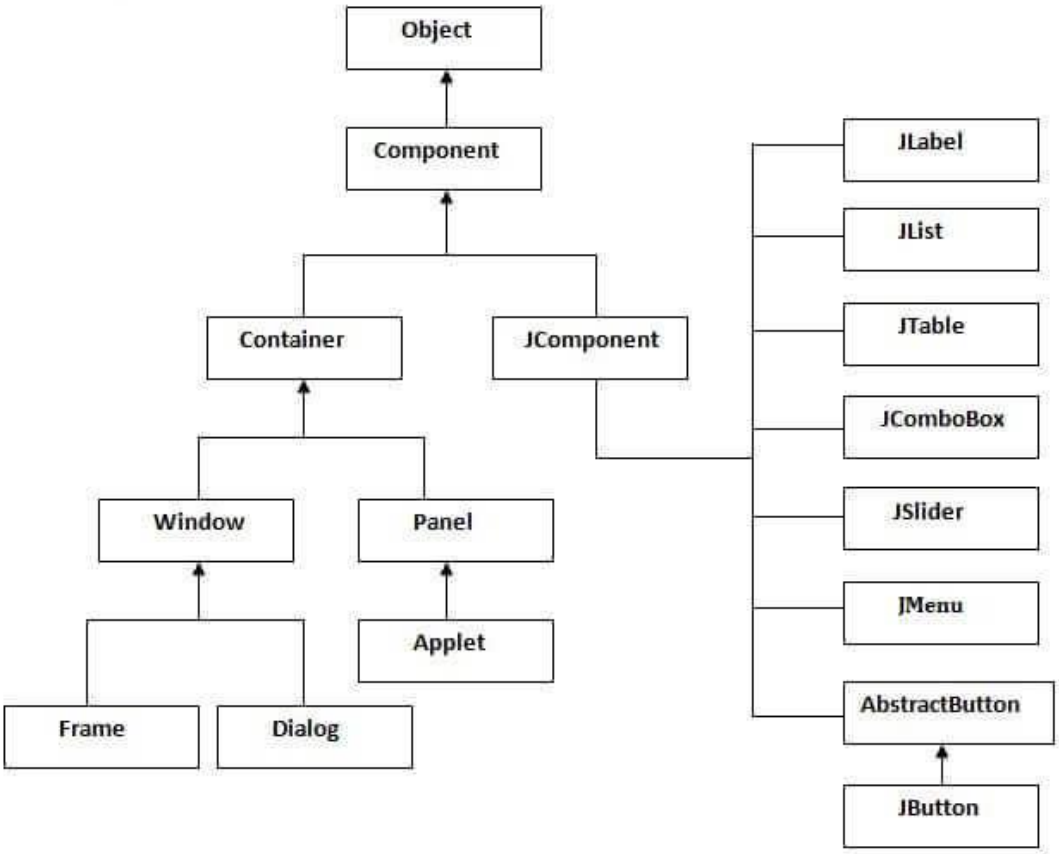
Swing:

Swing in Java is a Graphical User Interface (GUI) toolkit that includes the GUI components. Swing provides a rich set of widgets and packages to make sophisticated GUI components for Java applications. Swing is a part of Java Foundation Classes(JFC), which is an API for Java GUI programing that provide GUI.

The Java Swing library is built on top of the Java Abstract Widget Toolkit (AWT), an older, platform dependent GUI toolkit. You can use the Java simple GUI programming components like button, textbox, etc., from the library and do not have to create the components from scratch.



Java Swing Class Hierarchy Diagram



All components in Java Swing are JComponent which can be added to container classes.

`Container Class:

Container classes are classes that can have other components on it. So for creating a Java Swing GUI, we need at least one container object. There are 3 types of Java Swing containers.

1. Panel
2. Frame
3. Dialog

- **Panel:** It is a pure container and is not a window in itself. The sole purpose of a Panel is to organize the components on to a window.

- **Frame:** It is a fully functioning window with its title and icons.

- **Dialog:** It can be thought of like a pop-up window that pops out when a message has to be displayed. It is not a fully functioning window like the Frame.

Graphical User Interface (GUI):

GUI (Graphical User Interface) in Java is an easy-to-use visual experience builder for Java applications. It is mainly made of graphical components like buttons, labels, windows, etc. through which the user can interact with an application. GUI plays an important role to build easy interfaces for Java applications.

CHAPTER 4

SYSTEM TESTING

4.1. TESTING:

Testing is the major quality measure employed during the software development. After the coding phase, computer programs are available that can be executed for testing purpose. Testing is not only has to uncover error introduced during coding, but also locates errors during the previous phases. Thus the aim of testing is to uncover requirements, design or coding errors in the program.

System testing is an expensive but critical process that can take us much as fifty percent of the budget for the programs development. Consequently, different levels of testing are employed. In fact, a successful test is one that finds as errors. The system performance criteria deal with turnaround time, back, file protection and human factor. A test for the user acceptance should also be carried out.

TESTING STRATEGY:

This is the phase where bugs in the programs were to be found and corrected. One of the goals during dynamic testing is to produce a test suite, where the salary calculated with the desired outputs such as reports in the case. This is applied ensure that the modification of the program does not have any side effects. This type of testing is called fregression testing. Testing generally reeves all the residual bugs and removes the bugs and improves the bugs and improves reliability of the program.

TYPES OF TESTING:

The basic types of testing that can be done in this project is

- Unit testing
- Integration testing
- Validation testing
- Output testing
- User acceptance testing

I. UNIT TESTING:

This is the first level of testing. In this different modules are tested against the specifications produced the design of the modules. Unit testing is done for the verification of the code produced during the coding of single program module is an isolated environment. Unit testing first focuses on the modules independently of one another to locate errors.

After coding each dialog is tested and run individually. All necessary coding was removed and ensured that all the modules worked, as the programmer would expect. Logical errors found were corrected. Independently and verifying the outputs of each module in the presence of staff was concluded that the program was functioning as expected.

II. INTEGRATION TESTING:

Data can be lost access an interface, one module can have as adverse effort on the other sub functions when combine may not be produced the desired major functions. Integration testing is systematic testing is for constructing the program structure, a while at the same time conducting test to uncover errors associated with the interface. The objective are to take the unit tested as a whole. Here the correction is difficult because vast expenses of the entire program complicated isolation of causes. Thus in the integration testing step, all the errors are uncovered are corrected for the next testing step

II. VALIDATION TESTING:

This provides the final assurance that the software meets all functional, behavioral and performance requirements. The software is completely assembled as a package. Validation succeeds when the software functions in manner in which the user accepts. Validation refers to processor using software in alive environment in order to find errors. During the course of validating the system, failure may occur and sometimes the coding has to be changed according to the requirement. Thus the feedback from the validations phase generally produces changes in the software. Once the applications was made free all of logical and interface error, inputting dummy data ensure that the software developed satisfied all the requirements of the user.

IV. OUTPUT TESTING:

After performing the validation testing, the next is output testing of the proposed system since no system could be useful if it does not produced the required output generated or considered into two way; one is on screen and another Is printing format

The output format on the screen is found to be correct has the format was designed in a system design phase according to the users name. After the hard copy also the output comes out as the specified requirements by users hence output testing does not result in any correction in the system.

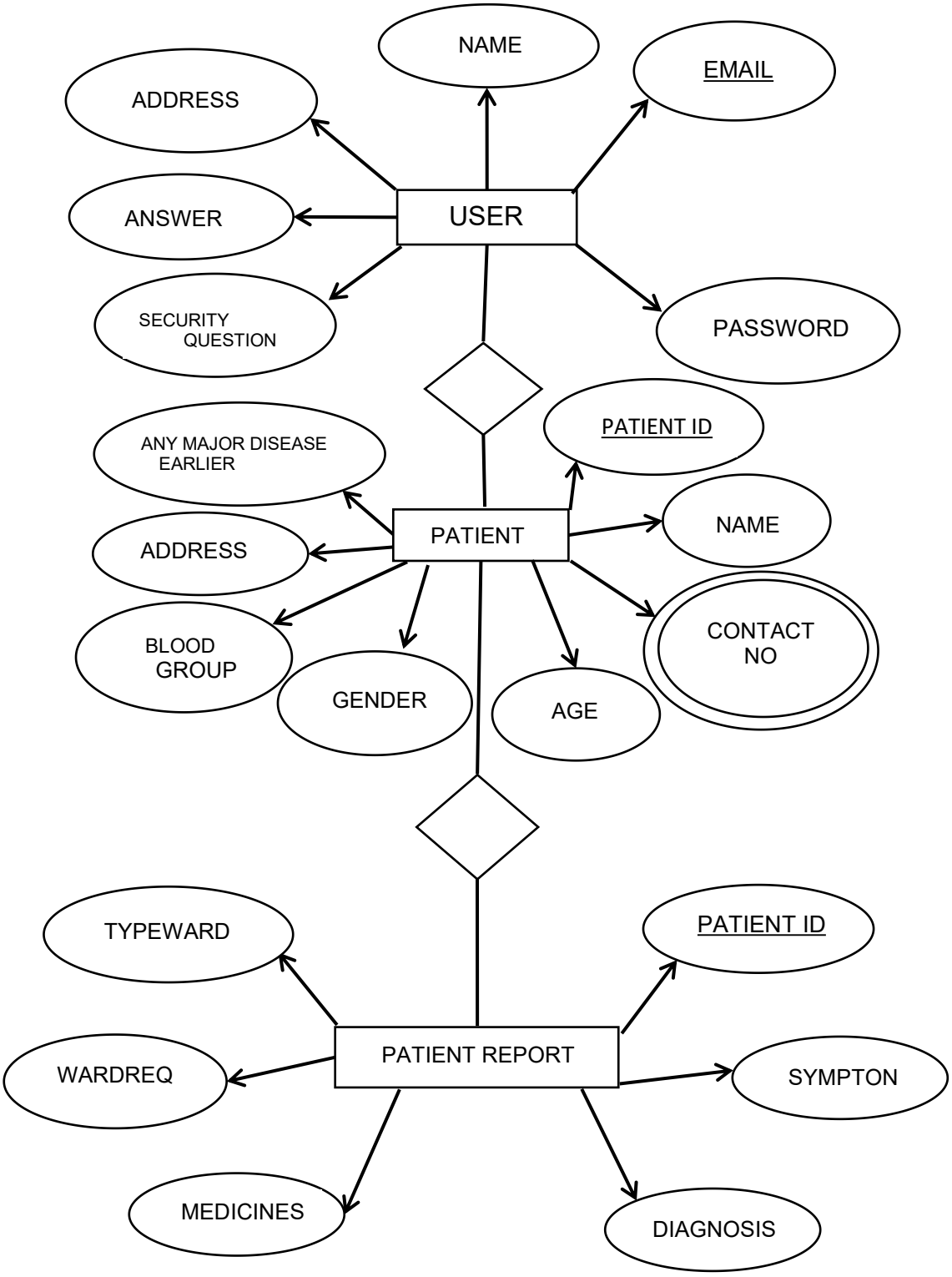
V. USER ACCEPTANCE TESTING:

User acceptance of a system is the key factor for the success of any system. the system under consideration is tested for the user acceptance by constantly keeping in touch with the prospective system user at the time of developing and making changes whenever required.

Preparation for the test data plays a vital role in a system testing. After preparing the test data the system under study is tested using the test data. By testing the system by using test data errors are again uncovered and correction are also noted for future use.

CHAPTER 5

ER DIAGRAM

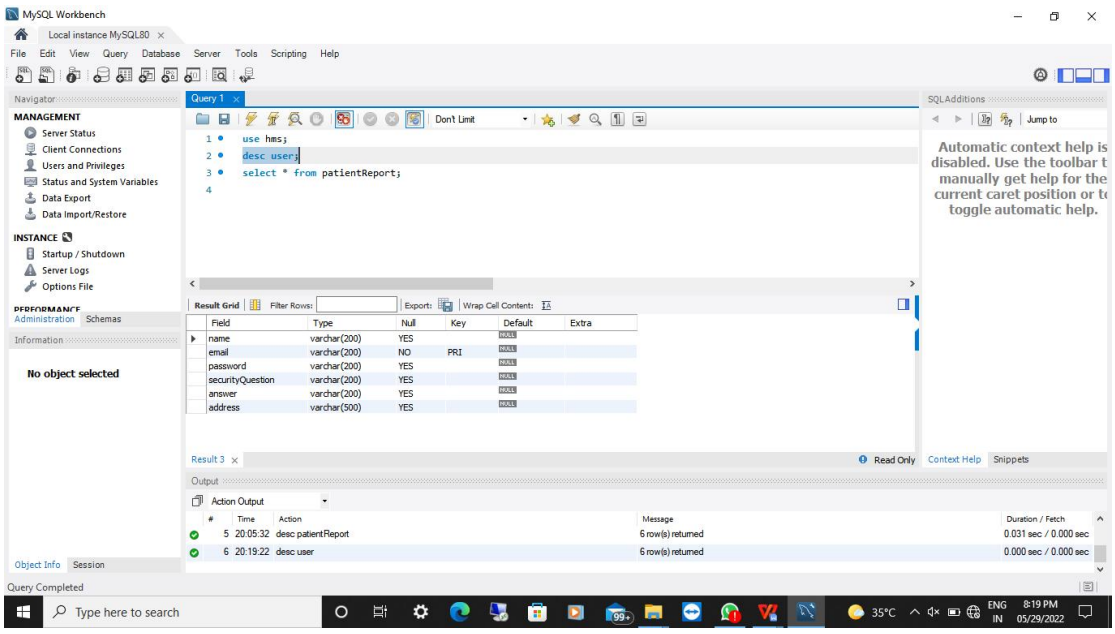


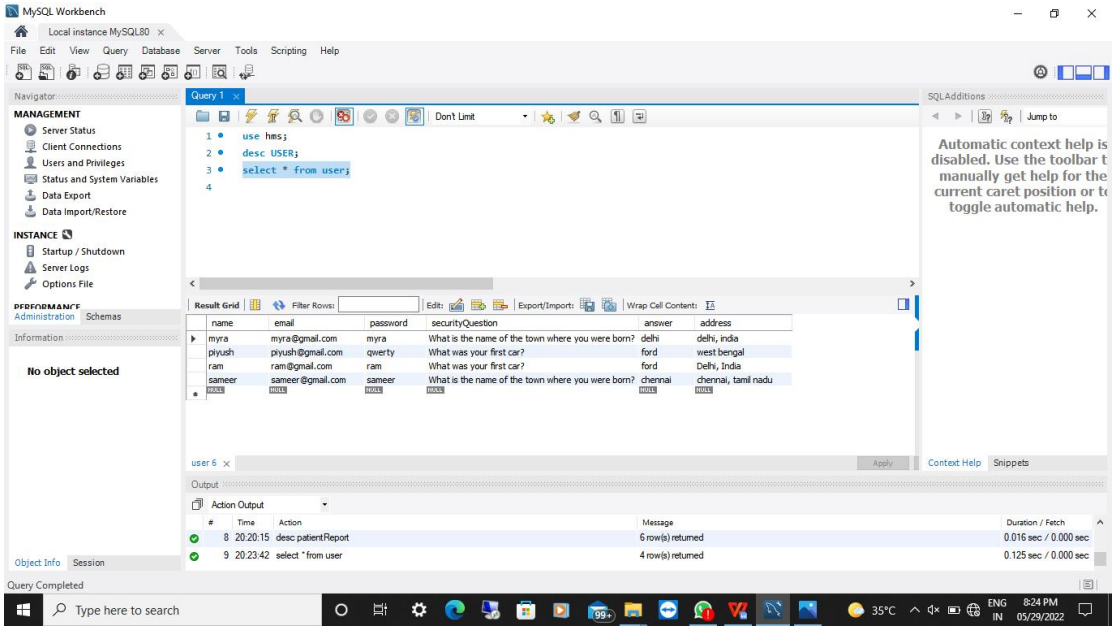
5.1 ER DIAGRAM

CHAPTER 6

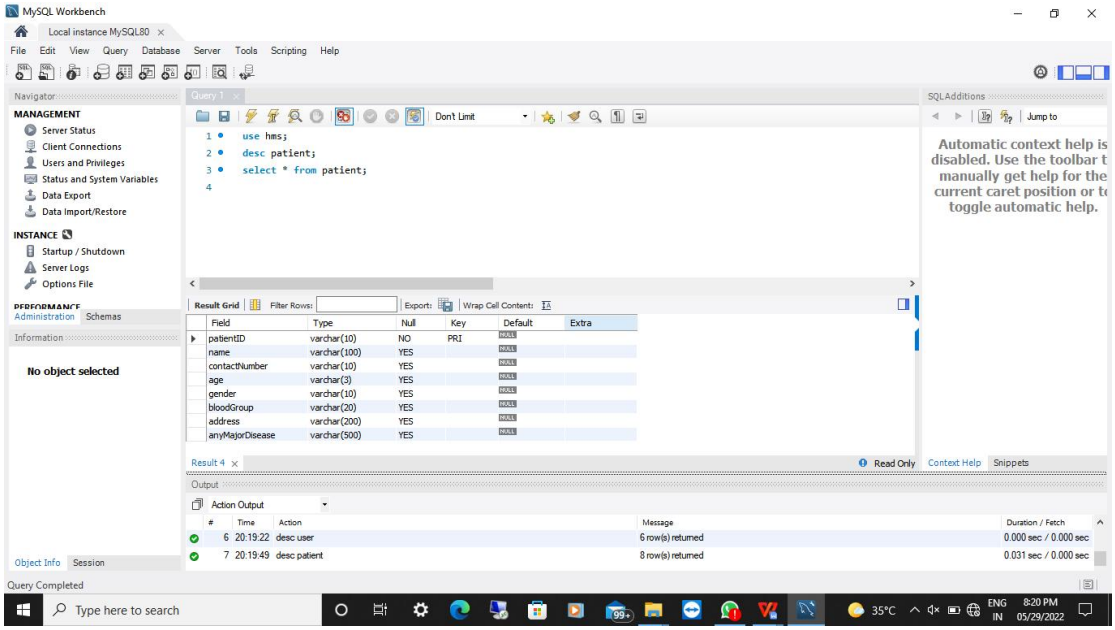
TABLE STRUCTURE

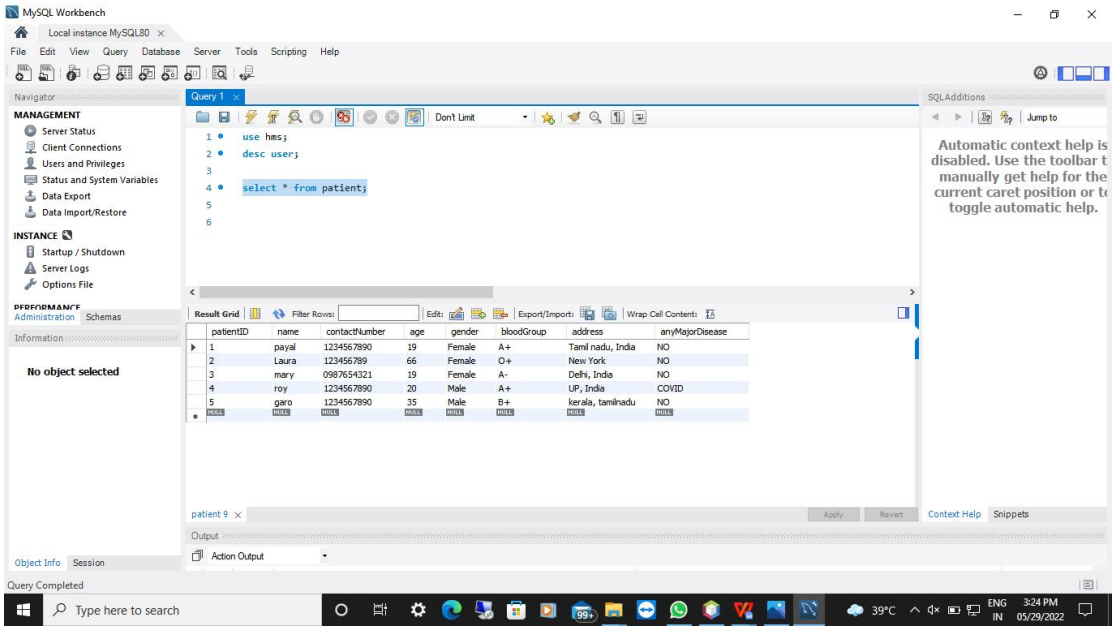
6.1. USER LIST TABLE



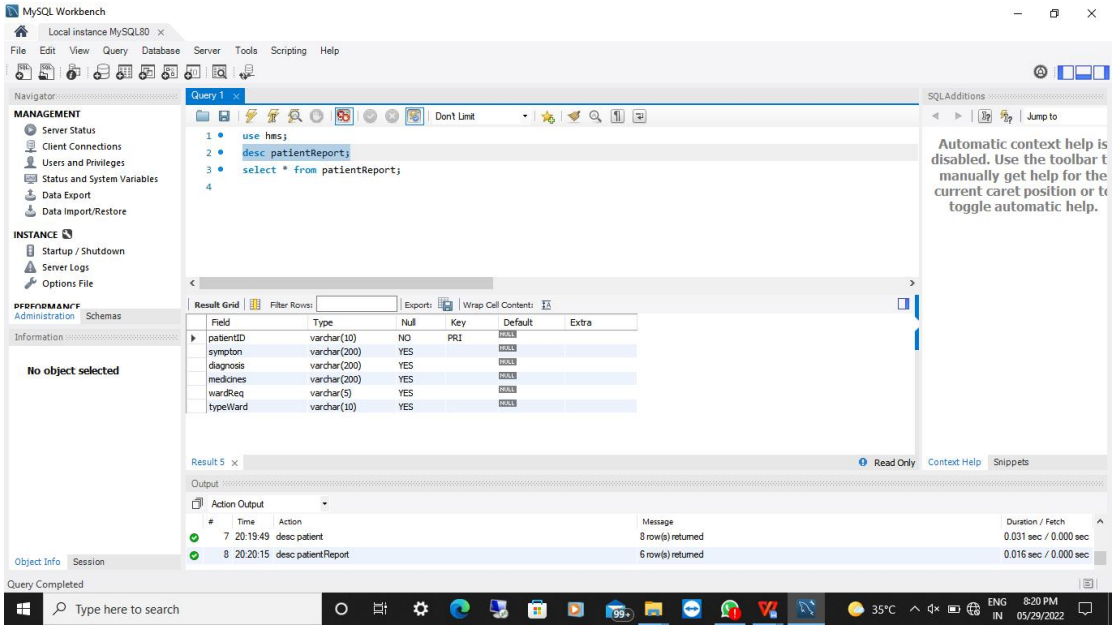

MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
MANAGEMENT
Server Status
Client Connections
Users and Privileges
Status and System Variables
Data Export
Data Import/Restore
INSTANCE
Startup / Shutdown
Server Logs
Options File
Administration Schemas
Information
No object selected
Object Info Session
Query 1
use hms;
desc USER;
select * from user;
Result Grid
name email password securityQuestion answer address
myra myra@gmail.com myra What is the name of the town where you were born? delhi delhi, india
piyush piyush@gmail.com qwerty What was your first car? ford west bengal
ram ram@gmail.com ram What was your first car? ford Delhi, India
sameer sameer@gmail.com sameer What is the name of the town where you were born? chennai chennai, tamil nadu
SQLAdditions
Automatic context help is disabled. Use the toolbar t manually get help for the current caret position or t toggle automatic help.
Context Help Snippets
Output
Action Output
Time Action Message Duration / Fetch
8 20:20:15 desc patientReport 6 row(s) returned 0.016 sec / 0.000 sec
9 20:23:42 select * from user 4 row(s) returned 0.125 sec / 0.000 sec
Query Completed
Type here to search
35°C
ENG IN
8:24 PM 05/29/2022

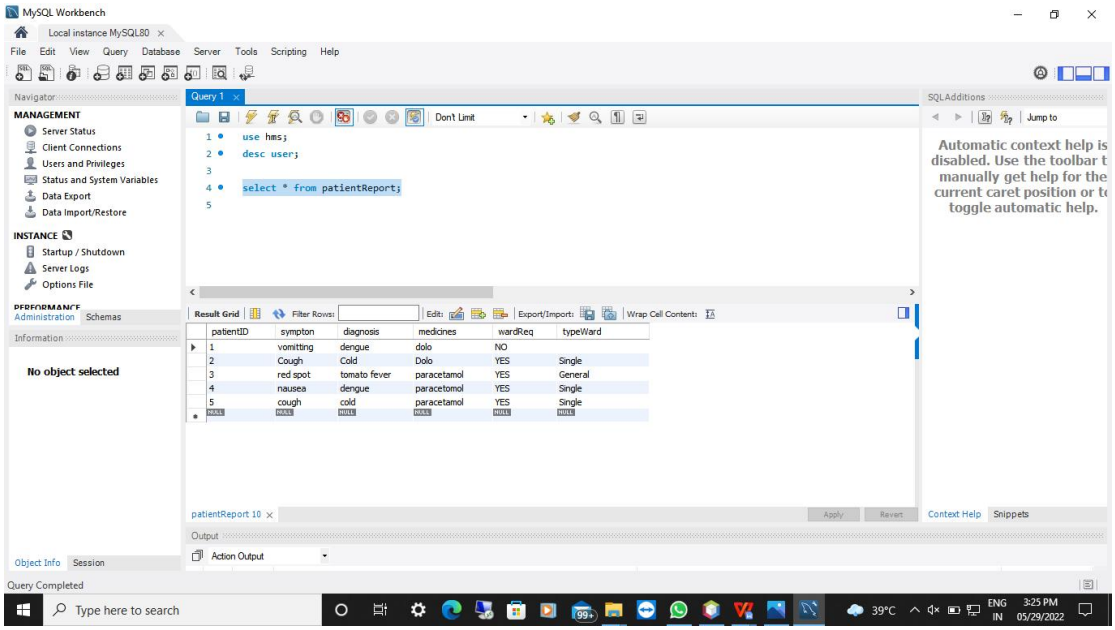
6.2. PATIENT LIST TABLE




MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
MANAGEMENT
Server Status
Client Connections
Users and Privileges
Status and System Variables
Data Export
Data Import/Restore
INSTANCE
Startup / Shutdown
Server Logs
Options File
Administration Schemas
Information
No object selected
Object Info Session
Query 1
use hms;
desc user;
select * from patient;
Result Grid
patientID name contactNumber age gender bloodGroup address anyMajorDisease
1 payal 1234567890 19 Female A+ Tamil nadu, India NO
2 Laura 123456789 66 Female O+ New York NO
3 mary 0987654321 19 Female A- Delhi, India NO
4 roy 1234567890 20 Male A+ UP, India COVID
5 garo 1234567890 35 Male B+ kerala, tamilnadu NO
patient 9
Output
Action Output
Query Completed
SQLAdditions
Jump to
Automatic context help is disabled. Use the toolbar t manually get help for the current caret position or t toggle automatic help.
Context Help Snippets
Type here to search
39°C
ENG IN
3:24 PM 05/29/2022

6.3. PATIENT REPORT LIST TABLE




MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
MANAGEMENT
Server Status
Client Connections
Users and Privileges
Status and System Variables
Data Export
Data Import/Restore
INSTANCE
Startup / Shutdown
Server Logs
Options File
Administration Schemas
Information
No object selected
Object Info Session
Query 1
Don't Limit
1 use hms;
2 desc user;
3
4 select * from patientReport;
5
Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content:
patientID sympton diagnosis medicines wardReq typeWard
1 vomitting dengue dolo NO
2 Cough Cold Dolo YES Single
3 red spot tomato fever paracetamol YES General
4 nausea dengue paracetamol YES Single
5 cough cold paracetamol YES Single
patientReport 10
Apply Revert
Output
Action Output
Query Completed
SQLAdditions
Jump to
Automatic context help is disabled. Use the toolbar t manually get help for the current caret position or t toggle automatic help.
Context Help Snippets
Type here to search
39°C
ENG IN
3:25 PM 05/29/2022

CHAPTER 7

SAMPLE SCREENSHOT

7.1. Login module:


HOSPITAL MANAGEMENT SYSTEM
Message
Every field is required!
OK
Password
Login
Signup
Close
Forget Password?


HOSPITAL MANAGEMENT SYSTEM
Email
sameer@gmail.com
Password
******
Login
Signup
Close
Forget Password?


HOSPITAL MANAGEMENT SYSTEM
Message
Logged-In successfully!!
OK
mail.com
Password
******
Login
Signup
Close
Forget Password?

7.2. Sign-up module:


Signup
Name
sameer
Email
sameer@gmail.com
Password
******
Security Question
What is the name of the town where you were bor...
Answer
chennai
Address
chennai, tamil nadu
Signup
Login
Close
Forgot Password?


Signup
Name
Email
Password
Security Questio
Answer
Address
chennai
chennai, tamil nadu
Signup
Login
Close
Forgot Password?
Message
Successfully signed up!
OK

7.3. Forget password module:


Forgot Password
Email
sameer@gmail.com
Search
Security Question
What is the name of the town where you were born?
Answer
chennai
New Password
sameer
Save
Signup
Login
Close


Forgot Password
Email
Search
Security Question
Answer
New Password
Save
Signup
Login
Close
Message
Incorrect Email!
OK

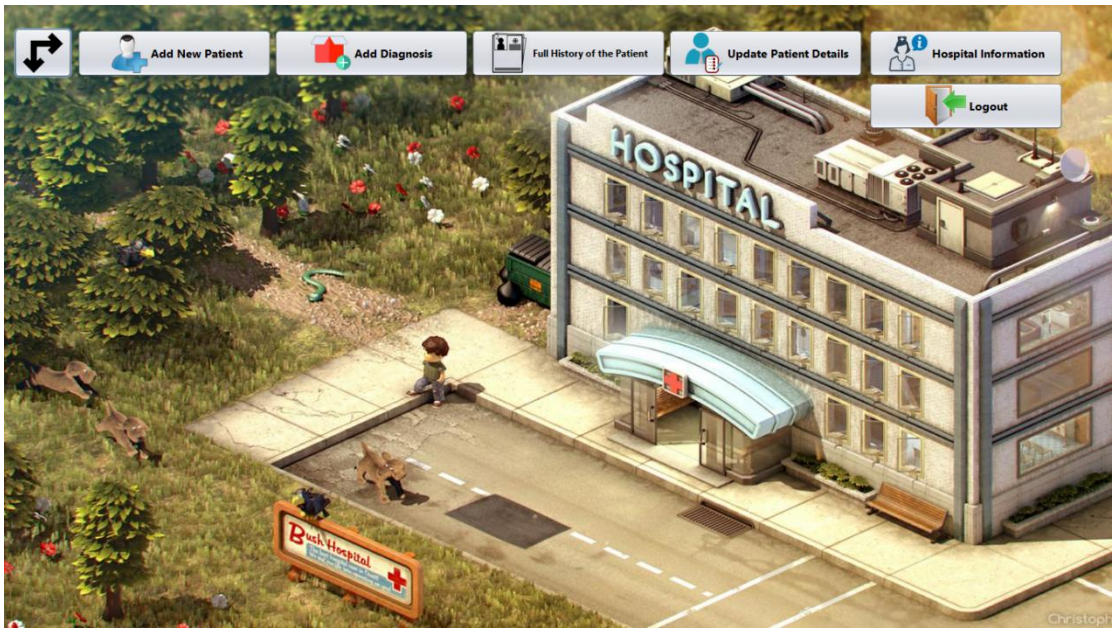

Forgot Password
Message
Email field is required*
OK
Email
Search
Security Question
Answer
New Password
Save
Signup
Login
Close


Forgot Password
Email
Search
Security Question
Answer
New Password
Save
Signup
Login
Close
Message
Email found!
OK


Forgot Password
Email
Search
Security Question
Answer
New Password
Save
Signup
Login
Close
Message
All Field is Required
OK


Forgot Password
Email
Search
Security Question
Answer
New Password
sameer
Save
Signup
Login
Close
Message
New Password set successfully!
OK

7.4. Home module:


Add New Patient
Add Diagnosis
Full History of the Patient
Update Patient Details
Hospital Information
Logout
HOSPITAL
Bush Hospital

7.5. Add new patient module:

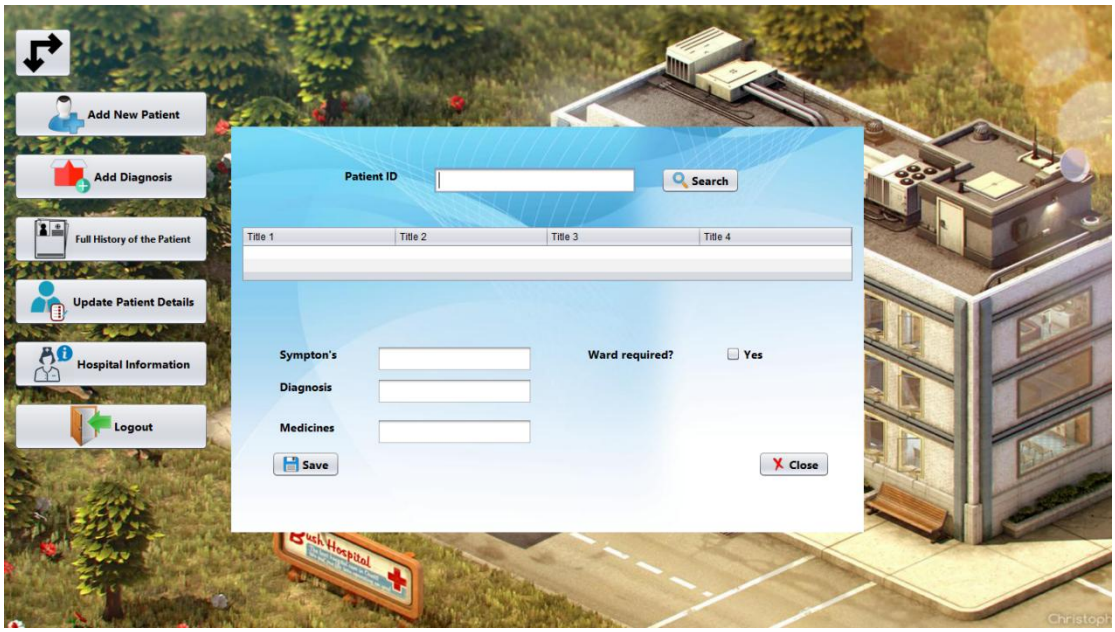

Add New Patient
Add Diagnosis
Full History of the Patient
Update Patient Details
Hospital Information
Logout
Patient ID
5
Name
garo
Contact No
1234567890
Age
34
Gender
Male
Blood Group
B+
Address
kerala, tamilnadu
Any Major Disease Earlier
NO
Save
Close


Add New Patient
Add Diagnosis
Full History of the Patient
Update Patient Details
Hospital Information
Logout
Patient ID
Name
Contact No
Age
Gender
Blood Group
Address
Any Major Disease Earlier
Save
Close
Message
Every Field is Required
OK

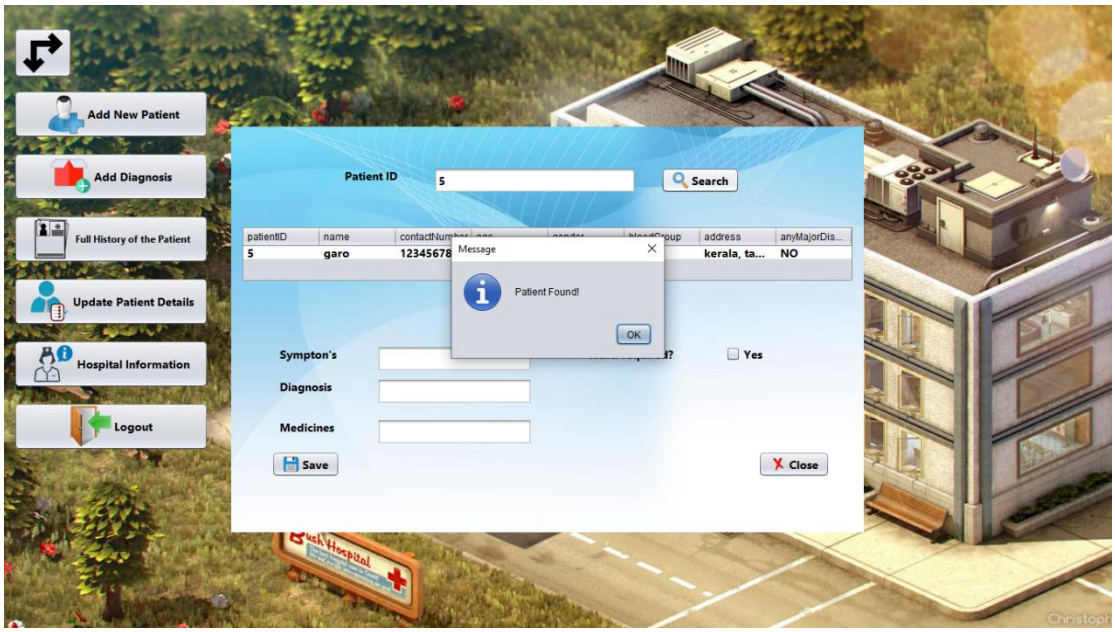

Add New Patient
Add Diagnosis
Full History of the Patient
Update Patient Details
Hospital Information
Logout
Patient ID
3
Name
garo
Contact No
Age
Gender
Blood Group
Address
kerala, tamilnadu
Any Major Disease Earlier
NO
Save
Close
Message
Please enter the Data in correct format!
OK

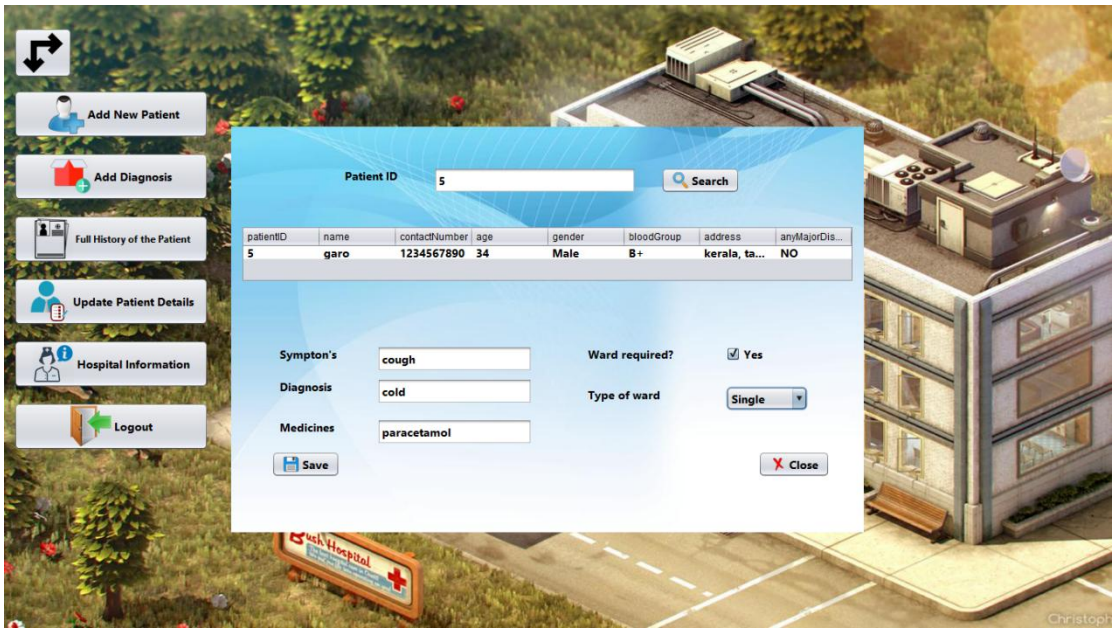

Add New Patient
Add Diagnosis
Full History of the Patient
Update Patient Details
Hospital Information
Logout
Patient ID
5
Name
garo
Contact No
Age
Gender
Blood Group
Address
kerala, tamilnadu
Any Major Disease Earlier
NO
Save
Close
Message
Successfully Updated
OK

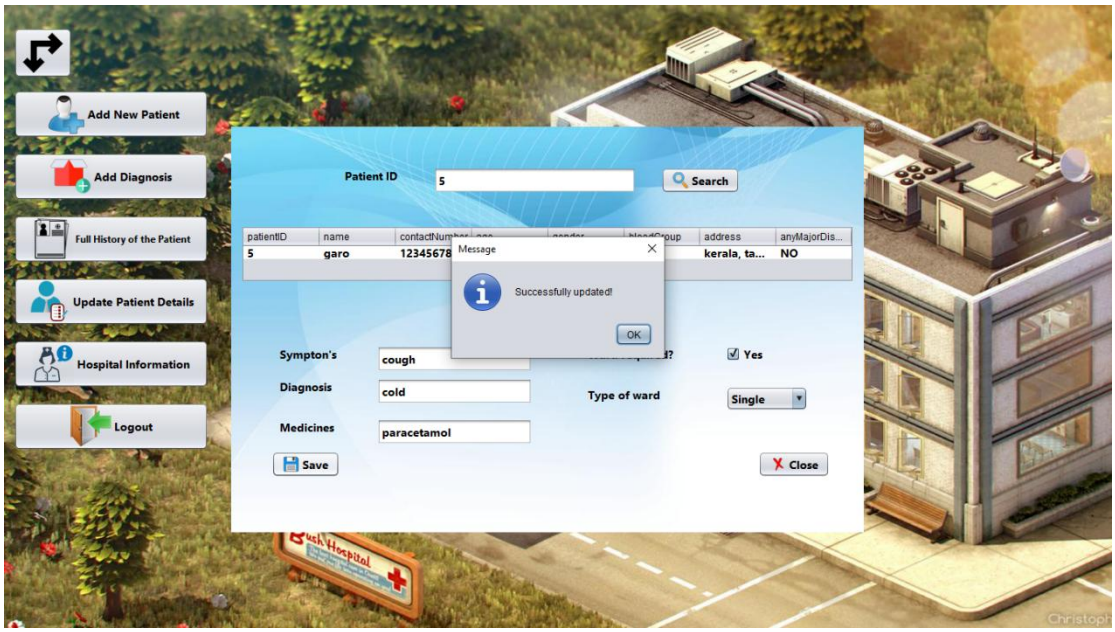
7.6. Add diagnosis module:



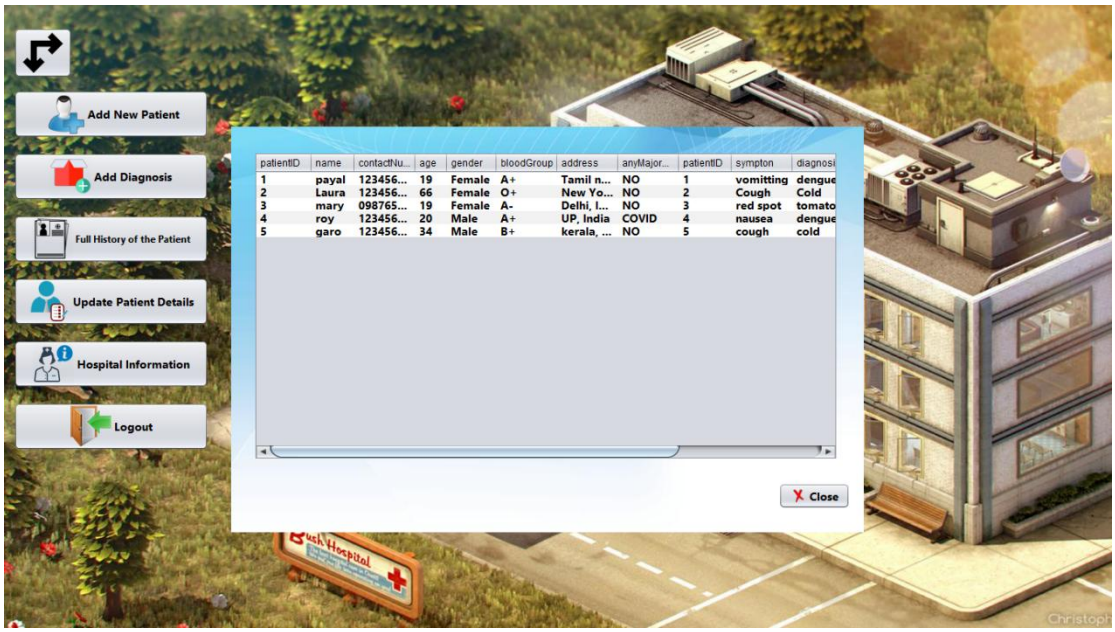

Add New Patient
Add Diagnosis
Full History of the Patient
Update Patient Details
Hospital Information
Logout
Patient ID
67
Search
Patient does not exist*
patientID
name
contactNumber
age
gender
bloodGroup
address
anyMajorDisease
Sympton's
Diagnosis
Medicines
Ward required?
Yes
Save
Close

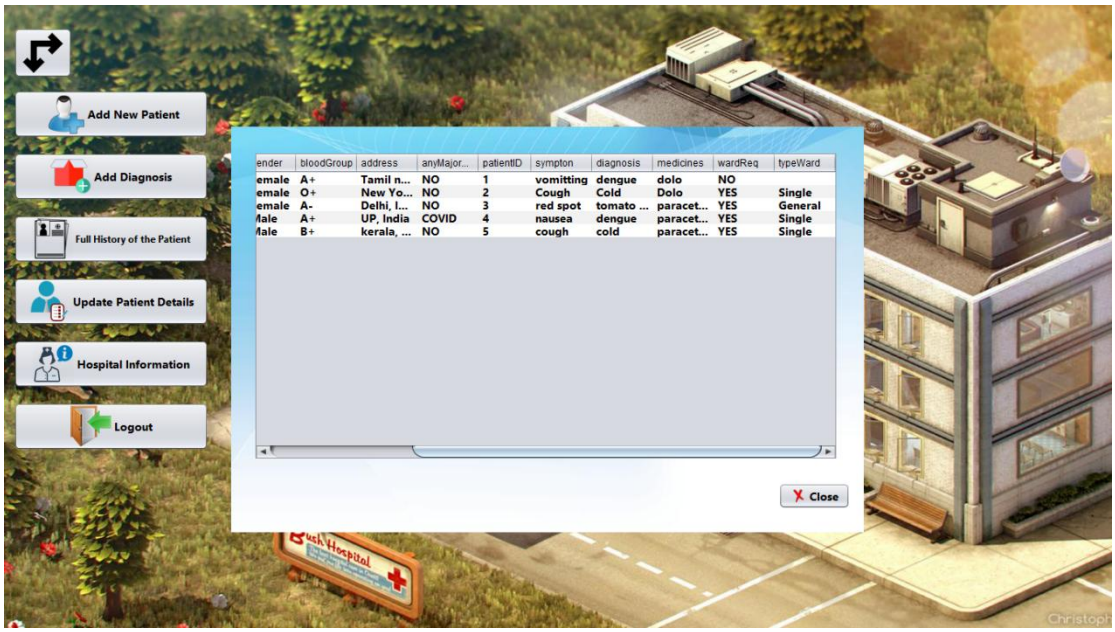

Add New Patient
Add Diagnosis
Full History of the Patient
Update Patient Details
Hospital Information
Logout
Patient ID
5
Search
patientID
name
address
anyMajorDis...
5
garo
kerala, ta...
NO
Message
Patient Found!
OK
Sympton's
Diagnosis
Medicines
Yes
Save
Close


Add New Patient
Add Diagnosis
Full History of the Patient
Update Patient Details
Hospital Information
Logout
Patient ID
5
Search
patientID
name
contactNumber
age
gender
bloodGroup
address
anyMajorDis...
5
garo
1234567890
34
Male
B+
kerala, ta...
NO
Sympton's
cough
Diagnosis
cold
Medicines
paracetamol
Ward required?
Yes
Type of ward
Single
Save
Close

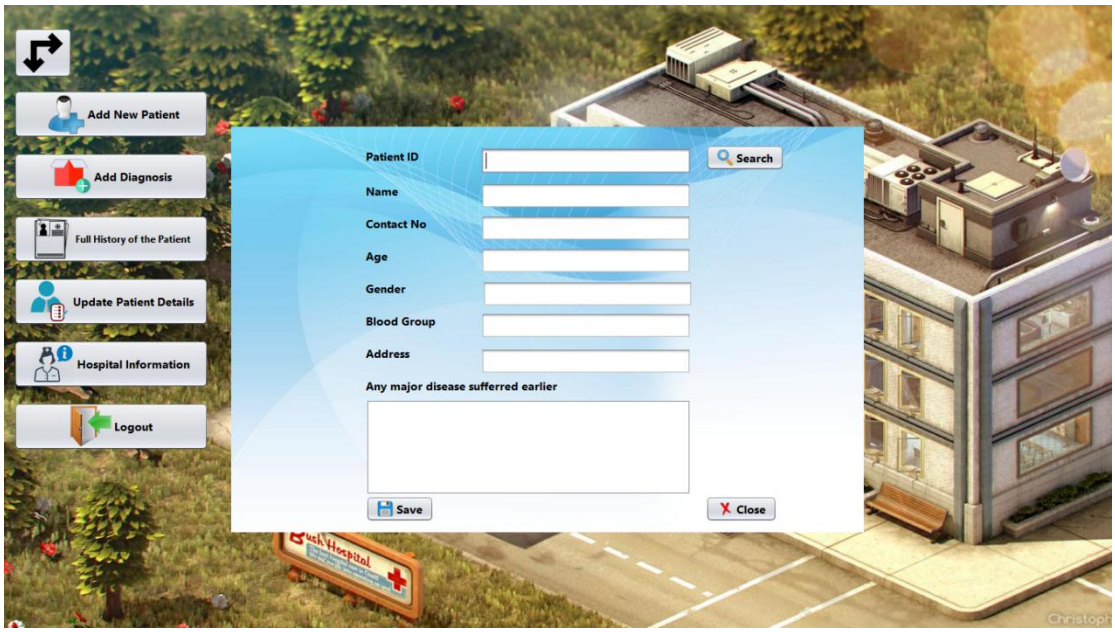

Add New Patient
Add Diagnosis
Full History of the Patient
Update Patient Details
Hospital Information
Logout
Patient ID
5
Search
patientID
name
address
anyMajorDis...
5
garo
kerala, ta...
NO
Message
Successfully updated!
OK
Sympton's
cough
Yes
Diagnosis
cold
Type of ward
Single
Medicines
paracetamol
Save
Close

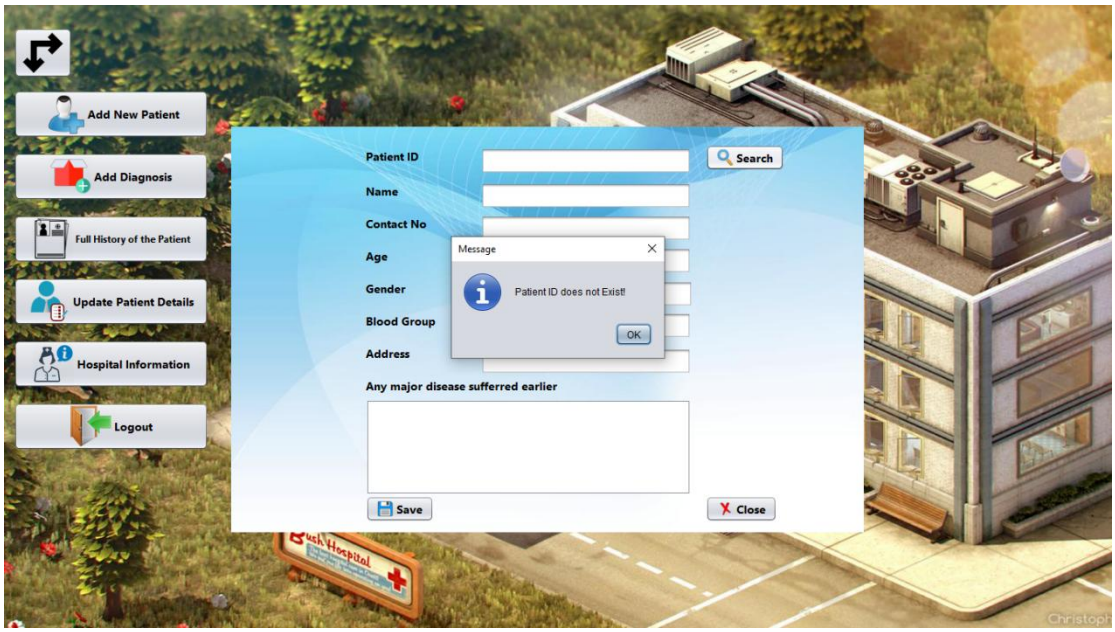
7.7. Full history of the patient module:

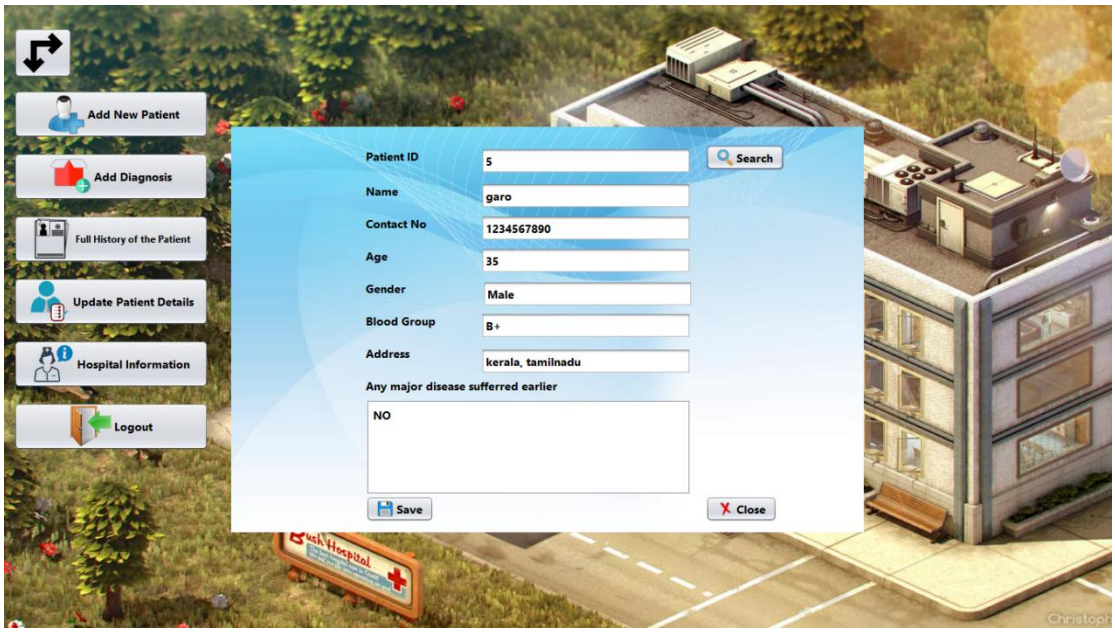


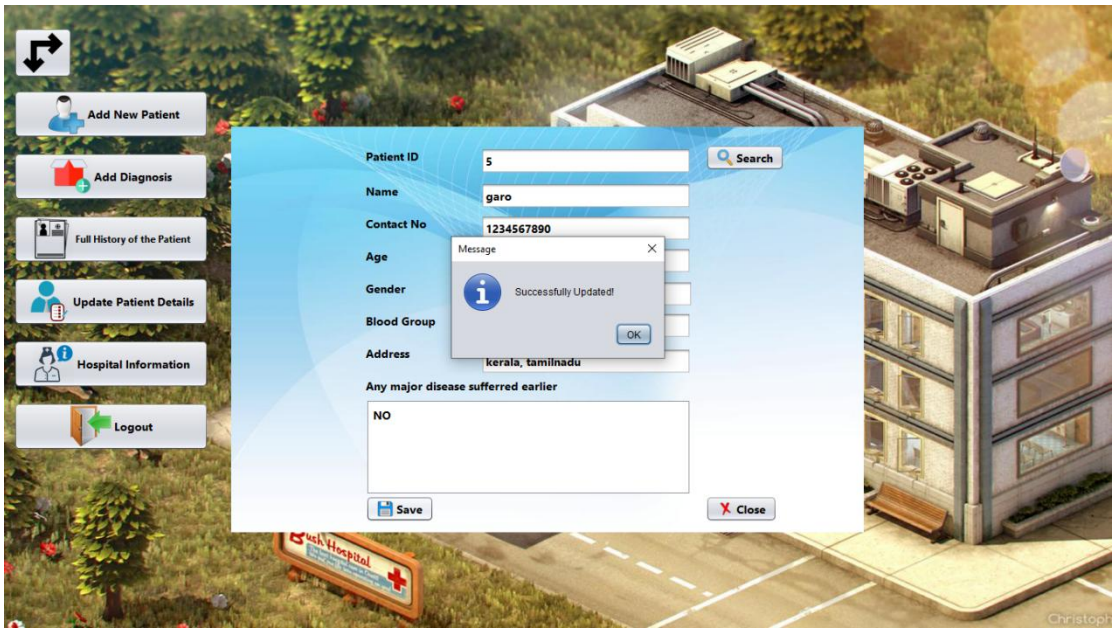

Add New Patient
Add Diagnosis
Full History of the Patient
Update Patient Details
Hospital Information
Logout
ender | bloodGroup | address | anyMajor... | patientID | sympton | diagnosis | medicines | wardReq | typeWard
emale | A+ | Tamil n... | NO | 1 | vomitting | dengue | dolo | NO
emale | O+ | New Yo... | NO | 2 | Cough | Cold | Dolo | YES | Single
emale | A- | Delhi, I... | NO | 3 | red spot | tomato ... | paracet... | YES | General
lale | A+ | UP, India | COVID | 4 | nausea | dengue | paracet... | YES | Single
lale | B+ | kerala, ... | NO | 5 | cough | cold | paracet... | YES | Single
Close

7.8. Update patient record module:

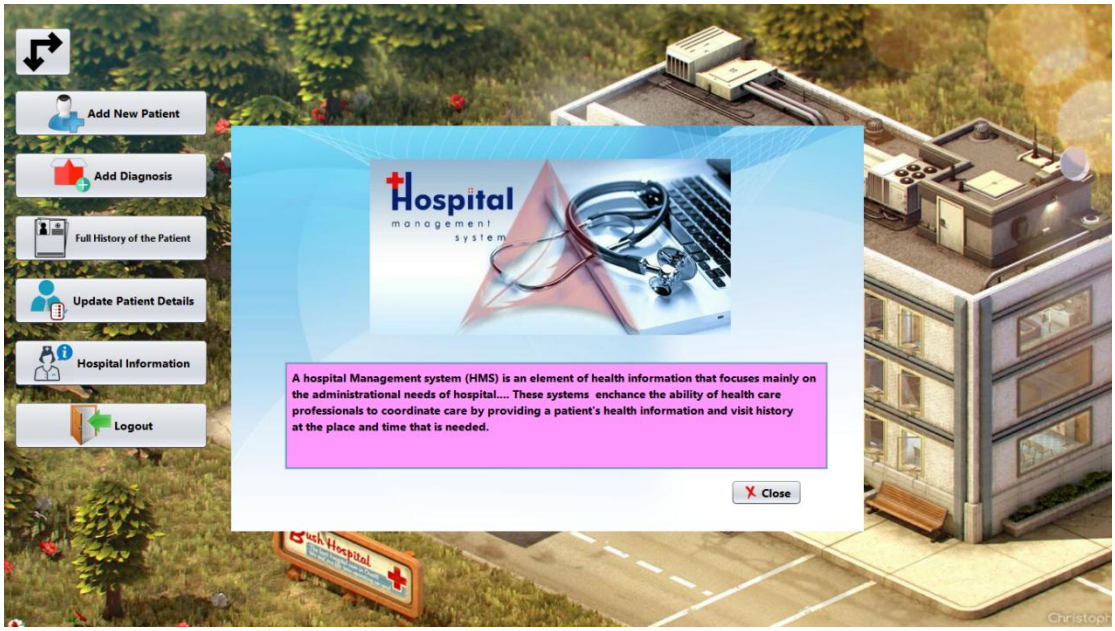



Add New Patient
Add Diagnosis
Full History of the Patient
Update Patient Details
Hospital Information
Logout
Patient ID
Search
Name
Contact No
Age
Gender
Blood Group
Address
Any major disease sufferred earlier
Save
Close
Message
Patient ID does not Exist!
OK

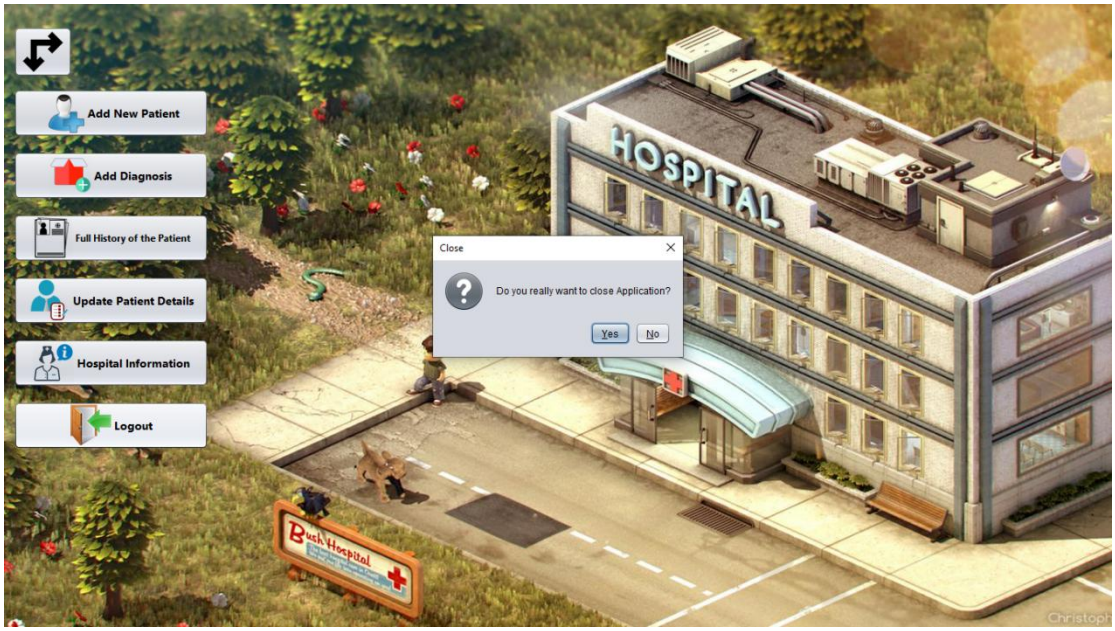

Add New Patient
Add Diagnosis
Full History of the Patient
Update Patient Details
Hospital Information
Logout
Patient ID
5
Search
Name
garo
Contact No
1234567890
Age
35
Gender
Male
Blood Group
B+
Address
kerala, tamilnadu
Any major disease sufferred earlier
NO
Save
Close


Add New Patient
Add Diagnosis
Full History of the Patient
Update Patient Details
Hospital Information
Logout
Patient ID
5
Search
Name
garo
Contact No
1234567890
Age
Gender
Blood Group
Address
kerala, tamilnadu
Any major disease sufferred earlier
NO
Save
Close
Message
Successfully Updated!
OK

7.9. Hospital information module:



7.10. Logout:


Add New Patient
Add Diagnosis
Full History of the Patient
Update Patient Details
Hospital Information
Logout
HOSPITAL
Close
Do you really want to close Application?
Yes
No
Bush Hospital

7.11. Close:


HOSPITAL MANAGEMENT SYSTEM
Close
Do you really want to close?
Yes
No
Password
Login
Signup
Close
Forget Password?

CHAPTER 8

SOURCE CODE

Login.java

```
import project.ConnectionProvider;
import java.sql.*;
import javax.swing.JOptionPane;

public class login extends javax.swing.JFrame {

      public login() {
          initComponents();
       }

       private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
                // TODO add your handling code here:
                // Login form => "Close Button"

                int n = JOptionPane.showConfirmDialog(null, "Do you really want to
                        close?","Close",JOptionPane.YES_NO_OPTION);
                if(n == 0)
                {
                     System.exit(0);
                }
        }

         private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
           // TODO add your handling code here:
           // Login form => "Login Button"

           String email = jTextField1.getText();
           String password = jPasswordField1.getText();
           if(jTextField1.getText().equals("") || jPasswordField1.getText().equals(""))
           {
                JOptionPane.showMessageDialog(null, "Every field is required!");
           }
           else
           {
                try
                {
                     Connection con=ConnectionProvider.getCon();
                     Statement st=con.createStatement();
                     ResultSet rs=st.executeQuery("select * from user where email='"+email+"'
and password='"+password+"'");
                     if(rs.next())
                     {
                          if(rs.getString(2).equals(email) && rs.getString(3).equals(password))
                          {
                               JOptionPane.showMessageDialog(null, "Logged-In successfully!!");
```

```
                    setVisible(false);
                    new home().setVisible(true);
                }
                else
                {
                    JOptionPane.showMessageDialog(null, "Invalid Username or
                     Password");
                    new login().setVisible(true);
                }
            }
        }
        catch (Exception e)
        {
            JOptionPane.showMessageDialog(null, e);
        }
    }
}

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    // Login form => "Signup Button"

      setVisible(false);
      new signup().setVisible(true);
    }

    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    // login form =>    "Forgot password"

      setVisible(false);
      new forgotPassword().setVisible(true);
    }
}
```

signup.java

```
import project.ConnectionProvider;
import javax.swing.JOptionPane;
import java.sql.*;

public class signup extends javax.swing.JFrame {

    public signup() {
        initComponents();
    }

    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:

        int a=JOptionPane.showConfirmDialog(null,"Do you really want to close
                Application?","Close",JOptionPane.YES_NO_OPTION);
        if(a==0)
        {
            setVisible(false);
            new login().setVisible(true);
        }
    }

    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        // Signup form => "Forgot password button"

        setVisible(false);
        new forgotPassword().setVisible(true);
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        // Signup form => "Login button"

        setVisible(false);
        new login().setVisible(true);
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        // Signup form => "Signup button"

        String name = jTextField1.getText();
        String email = jTextField2.getText();
        String password = jPasswordField1.getText();
```

```
            String securityQuestion = (String)jComboBox1.getSelectedItem();
            String answer = jTextField3.getText();
            String address = jTextField4.getText();
              if(name.equals("") || email.equals("") || password.equals("") ||
securityQuestion.equals("(Select)") || answer.equals("") || address.equals(""))
            {
                JOptionPane.showMessageDialog(null,"Every Field is Required");
                setVisible(false);
                new signup().setVisible(true);
            }
            else
            {
                  try{
                        Connection con= ConnectionProvider.getCon();
                        Statement st = con.createStatement();
                        st.executeUpdate("insert into user
values('"+name+"','"+email+"','"+password+"','"+securityQuestion+"','"+answer+"','"+address+"')")
;
                        JOptionPane.showMessageDialog(null, "Successfully signed up!");
                        setVisible(false);
                        new signup().setVisible(true);
                    }catch (Exception e) {
                        JOptionPane.showMessageDialog(null, "Email already exists!");
                    }
            }
        }
}
```

forgotPassword.java

```
import java.sql.*;
import javax.swing.JOptionPane;
import project.ConnectionProvider;

public class forgotPassword extends javax.swing.JFrame {

    /**
     * Creates new form forgotPassword
     */
    public forgotPassword() {
        initComponents();
    }

        private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
            // TODO add your handling code here:
            // Forgot password => "Search button"
            String email = jTextField1.getText();

            if(email.equals(""))
            {
                JOptionPane.showMessageDialog(null, "Email field is required*");
            }
            else
            {
                try
                {
                    Connection con=ConnectionProvider.getCon();
                    Statement st=con.createStatement();
                    ResultSet rs=st.executeQuery("select * from user where email=
    '"+email+"'");
                    if(rs.next())
                    {
                        JOptionPane.showMessageDialog(null, "Email found!");
                        jTextField2.setEditable(false);
                        jTextField1.setEditable(false);
                        jTextField2.setText(rs.getString(4));
                    }
                    else
                    {
                        JOptionPane.showMessageDialog(null, "Incorrect Email!");
                    }
                }
                catch (Exception e)
                {
                    JOptionPane.showMessageDialog(null, e);
```

```
                }
            }
        }

        private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
          // TODO add your handling code here:
          // Forgot password => "Signup button"

          setVisible(false);
          new signup().setVisible(true);
    }

    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
          // TODO add your handling code here:
          // Forgot password => "Login button"

          setVisible(false);
          new login().setVisible(true);
    }

    private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
          // TODO add your handling code here:
          // Forgot Password => "Close button"

          int a=JOptionPane.showConfirmDialog(null,"Do you really want to close
Application?","Close",JOptionPane.YES_NO_OPTION);
          if(a==0)
          {
                setVisible(false);
                new login().setVisible(true);
          }
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
          // TODO add your handling code here:
          // Forgot password => "Save button"

          String email = jTextField1.getText();
          String securityQuestion = jTextField2.getText();
          String answer = jTextField3.getText();
          String newPassword = jTextField4.getText();
          if(answer.equals("") || newPassword.equals(""))
          {
                JOptionPane.showMessageDialog(null, "All Field is Required");
          }
          else
          {
                try
                {
```

```
                    Connection con=ConnectionProvider.getCon();
                    Statement st=con.createStatement();
                    ResultSet rs=st.executeQuery("select * from user where email= '"+email+"'
and securityQuestion = '"+securityQuestion+"' and answer = '"+answer+"'");
                    if(rs.next())
                    {
                        st.executeUpdate("update user set password='"+newPassword+"' where
email='"+email+"'");
                        JOptionPane.showMessageDialog(null,"New Password set
successfully!");
                        setVisible(false);
                        new forgotPassword().setVisible(true);
                    }
                    else
                    {
                        JOptionPane.showMessageDialog(null, "Invalid answer!");
                    }
                }
                catch (Exception e)
                {
                    JOptionPane.showMessageDialog(null, e);
                }
            }
        }
}
```

home.java

```
import project.ConnectionProvider;
import javax.swing.JOptionPane;
import java.sql.*;

public class home extends javax.swing.JFrame {
    public int i=0; //declaring a public variable
    /**
     * Creates new form home
     */
    public home() {
        initComponents();
    }

    private void jButton8ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        // Home page => "Logout to login page"

        int a=JOptionPane.showConfirmDialog(null,"Do you really want to close
Application?","Close",JOptionPane.YES_NO_OPTION);
        if(a==0)
        {
            setVisible(false);
            new login().setVisible(true);
        }
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        // home page => "rotational button"

        if(i==0)            // Declared "i" variable in line 16
        {
            jButton3.setLocation(90, 30);
            jButton4.setLocation(333, 30);
            jButton5.setLocation(576, 30);
            jButton6.setLocation(819, 30);
            jButton7.setLocation(1066, 30);
            jButton8.setLocation(1066, 95);
//          jButton9.setLocation(1066, 160);
//          jButton10.setLocation(1066, 225);
            i=1;
        }
        else
        {
            jButton3.setLocation(10, 107);
            jButton4.setLocation(10, 184);
            jButton5.setLocation(10, 261);
```

```
                jButton6.setLocation(10, 338);
                jButton7.setLocation(10, 416);
                jButton8.setLocation(10, 490);
//              jButton9.setLocation(10, 567);
//              jButton10.setLocation(10, 644);
                i=0;
            }
        }

        private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
            // TODO add your handling code here:
            // Home page => "Add New Patient Button"

            new addNewPatientRecord().setVisible(true);
        }

        private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
             // TODO add your handling code here:
             // Home page => "Add Diagnosis button"

             new addDiagnosis().setVisible(true);
        }

        private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
             // TODO add your handling code here:
             // Home page => "Full history of the patient button"

             new fullHistoryOfThePatient().setVisible(true);
        }

        private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {
             // TODO add your handling code here:
             // Home page => "Update Patient Record"

             new updatePatientRecord().setVisible(true);
        }

        private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {
             // TODO add your handling code here:
             // Home page => "Hospital information"

             new hospitalInformation().setVisible(true);
        }
}
```

addNewPatientRecord.java

```
import project.ConnectionProvider;
import java.sql.*;
import javax.swing.JOptionPane;

public class addNewPatientRecord extends javax.swing.JFrame {
    /**
     * Creates new form addNewPatientRecord
     */
    public addNewPatientRecord() {
        initComponents();
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        // Home form => "Close button"

        setVisible(false);
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        // Add New Patient Record => "Save button"

        String patientID = jTextField1.getText();
        String name = jTextField2.getText();
        String contactNumber = jTextField3.getText();
        String age = jTextField4.getText();
        String gender = (String)jComboBox1.getSelectedItem();
        String bloodGroup = (String)jComboBox2.getSelectedItem();
        String address = jTextField5.getText();
        String anyMajorDisease = jTextField6.getText();

        if(patientID.equals("") || name.equals("") || contactNumber.equals("") ||
bloodGroup.equals("(Select)") || age.equals("") || gender.equals("(Select)") || address.equals("") ||
anyMajorDisease.equals(""))
        {
            JOptionPane.showMessageDialog(null,"Every Field is Required");
            setVisible(false);
            new addNewPatientRecord().setVisible(true);
        }
        else{
                try
                {
                    Connection con = ConnectionProvider.getCon();
                    Statement st=con.createStatement();
```

```
                    st.executeUpdate("insert into patient
            values('"+patientID+"','"+name+"','"+contactNumber+"','"+age+"','"+gender+"','"+bloodG
            roup+"','"+address+"','"+anyMajorDisease+"')");
                    JOptionPane.showMessageDialog(null, "Successfully Updated");
                    setVisible(false);
                    new addNewPatientRecord().setVisible(true);
                }
                catch(Exception e)
                {
                    JOptionPane.showMessageDialog(null, "Please enter the Data in correct
            format!");

                }
            }
        }
    }
```

addDiagnosis.java

```
import project.ConnectionProvider;
import java.sql.*;
import javax.swing.JOptionPane;
import net.proteanit.sql.DbUtils;

public class addDiagnosis extends javax.swing.JFrame {
public int flag=0;
    /**
     * Creates new form addDiagnosis
     */
    public addDiagnosis() {
        initComponents();
        jLabel2.setVisible(false);
        jLabel7.setVisible(false);
        jComboBox1.setVisible(false);
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        // Add Diagnosis form => "Close button"

        setVisible(false);
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        // addDiagnosis => "Search button"

        String patientID = jTextField1.getText();
        try
        {
            Connection con = ConnectionProvider.getCon();
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("select * from patient where
patientID='"+patientID+"'");
            jTable1.setModel(DbUtils.resultSetToTableModel(rs));
            if(!rs.first())
                jLabel2.setVisible(true);
            else
            {
                JOptionPane.showMessageDialog(null, "Patient Found!");
                jLabel2.setVisible(false);
                jTextField1.setEditable(false);
                flag=1;                                    //create flag var in line 15
            }
        }
```

```
            catch(Exception e)
            {
                JOptionPane.showMessageDialog(null, "Connection error");
            }
        }

        private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
            // TODO add your handling code here:
            // add Diagnosis => "Save Button"

            if(flag==1)
            {
                String patientID = jTextField1.getText();
                String sympton = jTextField2.getText();
                String diagnosis = jTextField3.getText();
                String medicines = jTextField4.getText();
                String wardReq;
                String typeWard;
                if(jCheckBox1.isSelected())
                {
                    wardReq = "YES";
                    typeWard = (String)jComboBox1.getSelectedItem();
                }
                else
                {
                    wardReq = "NO";
                    typeWard = "";
                }
                try
                {
                    Connection con = ConnectionProvider.getCon();
                    Statement st = con.createStatement();
                    st.executeUpdate("insert into patientreport
values('"+patientID+"','"+sympton+"','"+diagnosis+"','"+medicines+"','"+wardReq+"','"+typeWard
+"')");
                    JOptionPane.showMessageDialog(null, "Successfully updated!");
                    setVisible(false);
                    new addDiagnosis().setVisible(true);
                }
                catch(Exception e)
                {
                    JOptionPane.showMessageDialog(this, "Details is already added!");
                }
            }
            else
            {
                JOptionPane.showMessageDialog(null, "PatientID Field is empty!");
            }
        }
```

```
    private void jCheckBox1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        // addDiagnosis page => "CheckBox for ward"

        if(jCheckBox1.isSelected())
        {
            jLabel7.setVisible(true);
            jComboBox1.setVisible(true);
        }
        else
        {
            jLabel7.setVisible(false);
            jComboBox1.setVisible(false);
        }
    }
}
```

fullHistoryOfThePatient.java

```
import project.ConnectionProvider;
import java.sql.*;
import javax.swing.JOptionPane;
import net.proteanit.sql.DbUtils;

public class fullHistoryOfThePatient extends javax.swing.JFrame {
    /**
     * Creates new form fullHistoryOfThePatient
     */
    public fullHistoryOfThePatient() {
        initComponents();
    }

    private void formComponentShown(java.awt.event.ComponentEvent evt) {
        // TODO add your handling code here:
        //We're writing code here because whenever our code get execute it'll load here.

        try
        {
            Connection con = ConnectionProvider.getCon();
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("select * from patient inner join patientreport where
patient.patientID=patientreport.patientID");
            jTable1.setAutoResizeMode(jTable1.AUTO_RESIZE_OFF);
            jTable1.setModel(DbUtils.resultSetToTableModel(rs));
        }
        catch(Exception e)
        {
            JOptionPane.showMessageDialog(null, "Connection Error!");
        }
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        // Full history of the patient => "Close button"

        setVisible(false);
    }
}
```

updatePatientRecord.java

```
import project.ConnectionProvider;
import java.sql.*;
import javax.swing.JOptionPane;

public class updatePatientRecord extends javax.swing.JFrame {

    /**
     * Creates new form updatePatientRecord
     */
    public updatePatientRecord() {
        initComponents();
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        //updatePatientRecord => "Search Button"

        String patientID = jTextField1.getText();

        try
        {
            Connection con = ConnectionProvider.getCon();
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("select * from patient where
patientID='"+patientID+"'");
            if(rs.next())
            {
                jTextField1.setText(rs.getString(1));
                jTextField2.setText(rs.getString(2));
                jTextField3.setText(rs.getString(3));
                jTextField4.setText(rs.getString(4));
                jTextField5.setText(rs.getString(5));
                jTextField6.setText(rs.getString(6));
                jTextField7.setText(rs.getString(7));
                jTextArea1.setText(rs.getString(8));

            }
            else
                JOptionPane.showMessageDialog(null, "Patient ID does not Exist!");
        }
        catch(Exception e)
        {

        }
    }
```

```
    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        // Update Patient Record => "Close button"

        setVisible(false);
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        //updatePatientRecord => "Update Button"

        String patientID = jTextField1.getText();
        String name = jTextField2.getText();
        String contactNumber = jTextField3.getText();
        String age = jTextField4.getText();
        String gender = jTextField5.getText();
        String bloodGroup = jTextField6.getText();
        String address = jTextField7.getText();
        String anyMajorDisease = jTextArea1.getText();

        if(patientID.equals("") || name.equals("") || contactNumber.equals("") ||
bloodGroup.equals("") || age.equals("") || address.equals("") || gender.equals("") ||
anyMajorDisease.equals(""))
        {
            JOptionPane.showMessageDialog(null,"Every Field is Required");
            setVisible(false);
            new addNewPatientRecord().setVisible(true);
        }
        try
        {
            Connection con = ConnectionProvider.getCon();
            Statement st = con.createStatement();
            st.executeUpdate("Update patient set
name='"+name+"',contactNumber='"+contactNumber+"',age='"+age+"',gender='"+gender+"',bloo
dGroup='"+bloodGroup+"',address='"+address+"',anyMajorDisease='"+anyMajorDisease+"'
where patientID='"+patientID+"'");
            JOptionPane.showMessageDialog(null, "Successfully Updated!");
            setVisible(false);
            new updatePatientRecord().setVisible(true);
        }
        catch(Exception e)
        {
            JOptionPane.showMessageDialog(null, "Please enter the Data in correct format!");
        }
    }
}
```

hospitalInformation.java

```
public class hospitalInformation extends javax.swing.JFrame {

    /**
     * Creates new form hospitalInformation
     */
    public hospitalInformation() {
        initComponents();
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        // Hospital Information => "Close button"

        setVisible(false);
    }

    private void formComponentShown(java.awt.event.ComponentEvent evt) {
        // TODO add your handling code here:
        // Hospital Information => “Text area for hospital information”

        jTextArea1.setEditable(false);
    }
}
```

ConnectionProvider.java

```
package project;

import java.sql.*;

/**
 *
 * @author admin 1
 */

public class ConnectionProvider {
    public static Connection getCon()
    {
        try
        {
            Class.forName("com.mysql.jdbc.Driver");
            Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/hms","root","Shabittaj_786");
            return con;
        }
        catch(Exception e)
        {
            return null;
        }
    }
}
```

CHAPTER 9

CONCLUSION

9.1. Conclusion

Since we are entering details of the patients electronically in the Hospital Management System", data will be secured. Using this application we can retrieve patient's history with a single click. Thus processing information will be faster. It guarantees accurate maintenance of Patient details. It easily reduces the book keeping task and thus reduces the human effort and increases accuracy speed.

CHAPTER 10

FUTURE ENHANCEMENT

10.1. Future enhancement:

1. User will get appointment in mobile phone through the SMS.
2. Modules will use OTP verification, so it control the flow.
3. User will have the various option for do payment.
4. User will track the patient current health using new features.
5. User will get the help of AI assistant so user can ask anything related.
6. In future there will be the android application which based on web.

CHAPTER 11

BIBLIOGRAPHY

11.1 Bibliography

- Java: The Complete Reference - Herbert Schildt
- Introducing InnoDB cluster - Charles Bell
- Beginning Netbeans IDE - Geertjan Wielenga