**The Islamia University of Bahawalpur**

**Department of Computer Science**

****

**SOFTWARE DESIGN DESCRIPTION**

**(SDD DOCUMENT)**

**for**

**<Hospital Management System>**

Version 1.0

***By***

**Student Name**

**Roll No**

**Session Spring/Fall 20xx – 20xx**

***Supervisor***

**Supervisor Name**

***Bachelor of Science in Computer Science***

Table of Contents

[1. INTRODUCTION 3](#_Toc153204668)

[1.1 Purpose 3](#_Toc153204669)

[1.2 Scope 3](#_Toc153204670)

[2. Overall description 4](#_Toc153204671)

[2.1 Product perspective 4](#_Toc153204672)

[2.2 Overview 4](#_Toc153204673)

[3. Design methodology and software process model 5](#_Toc153204674)

[3.1 Design Methodology: 5](#_Toc153204675)

[3.2 Process Model: 5](#_Toc153204676)

[3.3 Justification 5](#_Toc153204677)

[3. SYSTEM OVERVIEW 6](#_Toc153204678)

[3.1 ARCHITECTURAL DESIGN 6](#_Toc153204679)

[3.1.1 Architectural Design 6](#_Toc153204680)

[7](#_Toc153204681)

[3.1.2 Decomposition Description 7](#_Toc153204682)

[3.2 Process flow 8](#_Toc153204683)

[4. Design models 10](#_Toc153204684)

[4.1 Class Diagram 10](#_Toc153204685)

[4.2 Sequence Diagram 11](#_Toc153204686)

[4.3 Data flow diagrams (DFD): 12](#_Toc153204687)

[5. Data Design: 16](#_Toc153204688)

[5.1 Data Dictionary 17](#_Toc153204689)

[6. Algorithm & Implementation 20](#_Toc153204690)

[7. Software requirements traceability matrix 22](#_Toc153204691)

[8. HUMAN INTERFACE DESIGN 23](#_Toc153204692)

[7.1 Screen Images 25](#_Toc153204693)

# INTRODUCTION

A Hospital Management System (HMS) is a sophisticated software solution designed to digitize and optimize the administrative and clinical processes within a healthcare institution. This system facilitates efficient management of patient records, billing, and inventory. By centralizing these functions, an HMS helps enhance overall hospital efficiency, reduce paperwork, and improve patient care. It provides a user-friendly interface for healthcare professionals and administrative staff, contributing to a more organized and streamlined healthcare environment.

## 1.1 Purpose

The primary purpose of a **Hospital Management System (HMS)** is to streamline and automate the administrative and clinical operations within a healthcare facility. This includes managing patient information, maintaining medical records, handling billing and invoicing, and overseeing inventory. The overarching goal is to improve overall operational efficiency, reduce manual paperwork, enhance accuracy in data management, and ultimately, provide better patient care. The HMS aims to create a centralized and integrated platform that enables healthcare professionals and administrators to access, manage, and share critical information swiftly, contributing to the overall effectiveness and productivity of the healthcare institution.

## 1.2 Scope

Information about Patients is done by just writing the Patients name, age and gender. Whenever he Patient comes up his information is stored freshly. Bills are generated by recording price for each facility provided to Patient on a separate sheet and at last they all are summed up. 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. 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 must remember various medicines available for diagnosis and sometimes miss better alternatives as they can’t remember them at that time.

# Overall description

## 2.1 Product perspective

The Hospital Management System (HMS) represents a new and innovative product designed to enhance the efficiency and effectiveness of healthcare management. It is not a replacement for an existing application but rather a fresh solution addressing the growing needs of modern healthcare facilities. The system stands as an independent and standalone product, not necessarily part of an existing product line. It introduces a novel approach to hospital management, leveraging advanced technologies to streamline patient data management, appointment scheduling, medical record updates, billing, and inventory management. The HMS is envisioned as a comprehensive tool catering to the evolving requirements of healthcare professionals and administrative staff, providing a user-friendly interface for seamless interaction with the system.

## 2.2 Overview

The Hospital Management System (HMS) is a digitized healthcare solution designed for efficient administration and clinical operations. It employs Object-Oriented Programming (OOP) principles, using C# in Visual Studio to create a user-friendly interface. The system's key functionalities include patient management, appointment scheduling, medical records maintenance, billing, and inventory management. With a modular design for easy scalability, the HMS aims to streamline healthcare processes, reduce paperwork, and enhance patient care. It serves as a centralized hub for stakeholders, promoting data accuracy and tailored access for different user roles. The project responds to the evolving complexities of healthcare data management, providing a robust, secure, and adaptable solution for healthcare institutions.

# Design methodology and software process model

## 3.1 Design Methodology:

For a Hospital Management System (HMS) project, Object-Oriented Programming (OOP) is often the preferred design methodology. OOP is well-suited for projects that involve complex, interconnected entities with behaviors and attributes, making it an ideal choice for modeling the various components of a healthcare system.

In OOP, the system is conceptualized as a collection of objects, each representing a real-world entity, with properties (attributes) and behaviors (methods). For an HMS, entities such as patients, doctors, appointments, and medical records can be modeled as objects. This approach promotes encapsulation, modularity, and reusability, allowing for a more intuitive representation of the healthcare domain and facilitating easier maintenance and expansion.

## Process Model:

The choice of a process model depends on the project's requirements, complexity, and the need for flexibility. For a Hospital Management System project, an Agile development process model is often justified. Agile methodologies, such as Scrum or Kanban, are well-suited for projects that require frequent adjustments, close collaboration with stakeholders, and the ability to adapt to changing requirements.

## Justification

1. **Iterative Development:** Agile allows for iterative development, which is crucial for an HMS as requirements might evolve, and feedback from healthcare professionals and administrators is essential for refining the system.
2. **Continuous Stakeholder Involvement:** Agile emphasizes continuous collaboration with stakeholders. In healthcare projects, involving healthcare professionals and administrators throughout the development process ensures that the system aligns with their needs and expectations.
3. **Flexibility in Requirements:** Healthcare systems often face evolving regulatory and compliance standards. Agile's flexibility in accommodating changing requirements is advantageous in adapting the HMS to comply with any emerging standards.
4. **Incremental Delivery:** Agile facilitates incremental delivery of functionality. This allows for the early deployment of essential features, enabling users to start benefiting from the system sooner.
5. **Risk Mitigation:** Agile's iterative nature enables early identification and mitigation of risks. This is particularly important in healthcare projects, where data security, privacy, and accuracy are critical concerns.

# 3. SYSTEM OVERVIEW

The Hospital Management System (HMS) is a digitized healthcare solution designed for efficient administration and clinical operations. It employs Object-Oriented Programming (OOP) principles, using C# in Visual Studio to create a user-friendly interface. The system's key functionalities include patient management, appointment scheduling, medical records maintenance, billing, and inventory management. With a modular design for easy scalability, the HMS aims to streamline healthcare processes, reduce paperwork, and enhance patient care. It serves as a centralized hub for stakeholders, promoting data accuracy and tailored access for different user roles. The project responds to the evolving complexities of healthcare data management, providing a robust, secure, and adaptable solution for healthcare institutions.

## 3.1 ARCHITECTURAL DESIGN

The Hospital Management System (HMS) adopts a modular program structure to ensure a scalable and maintainable architecture. The system comprises key subsystems that collaborate to achieve comprehensive functionality.

### 3.1.1 Architectural Design

|  |  |
| --- | --- |
| Notation | Description |
|  | This shape is called actor. Its play role in a system. Actor play role as a human users or other subject and external hardware. One actor may be played different activities. Its show with stick man icon. |
|  | Ellipse shape is used as use case. Its show the functionality of any system.  Each use case must have a name. |
|  | This shape shows association. It shows the relationship between actor and use case. Through this use case show interaction. |

## 

User Case Diagram

|  |  |
| --- | --- |
| Use Case | UC1 |
| Use case name | How system will work. |
| actor | Admin, Doctor, Pharmacist. |
| Description | Admin can save edit update and delete records. Pharmacist can sale delete and update the medicines record. |

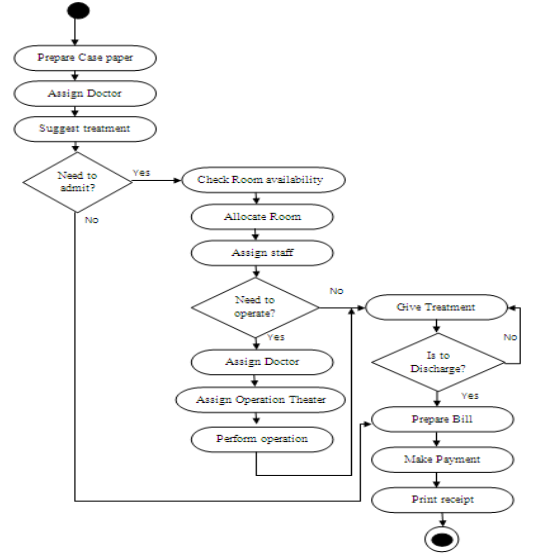
### 3.1.2 Decomposition Description

The decomposition of the Hospital Management System (HMS) was executed to enhance the system's manageability, security, and scalability. By breaking down the system into modular components like Patient Management, Appointment Scheduling, and Billing, each module addresses a specific concern, ensuring clear separation of functionalities. Encapsulation is employed to secure sensitive data within modules, promoting a controlled access approach. The modular design allows for scalability, accommodating future enhancements without disrupting the entire system. This approach improves system maintainability, as updates to one module do not necessitate changes in others, facilitating efficient collaboration and reducing overall complexity. The decomposition strategy is a deliberate effort to streamline development, enhance security, and support the evolving needs of healthcare management.

## 3.2 Process flow

In the Hospital Management System (HMS), the major processes are briefly represented through an activity diagram. The patient registration process is initiated by the receptionist, gathering essential information and creating a unique patient ID. Appointment scheduling, facilitated by healthcare professionals, involves selecting available time slots and linking them to patient records. Following patient appointments, medical records are updated with diagnoses, treatments, and test results. The billing process is triggered by completed medical services, generating invoices based on the services provided, which subsequently impacts inventory management. Report generation, both periodic and on-demand, compiles information on appointments, billing, and inventory. Emergency handling involves quick patient registration, appointment scheduling, and medical records updates during critical situations. Routine tasks like system maintenance, end-of-day procedures, and system shutdown are seamlessly integrated into the HMS workflow, ensuring efficient and organized hospital operations.

**Activity Diagram**



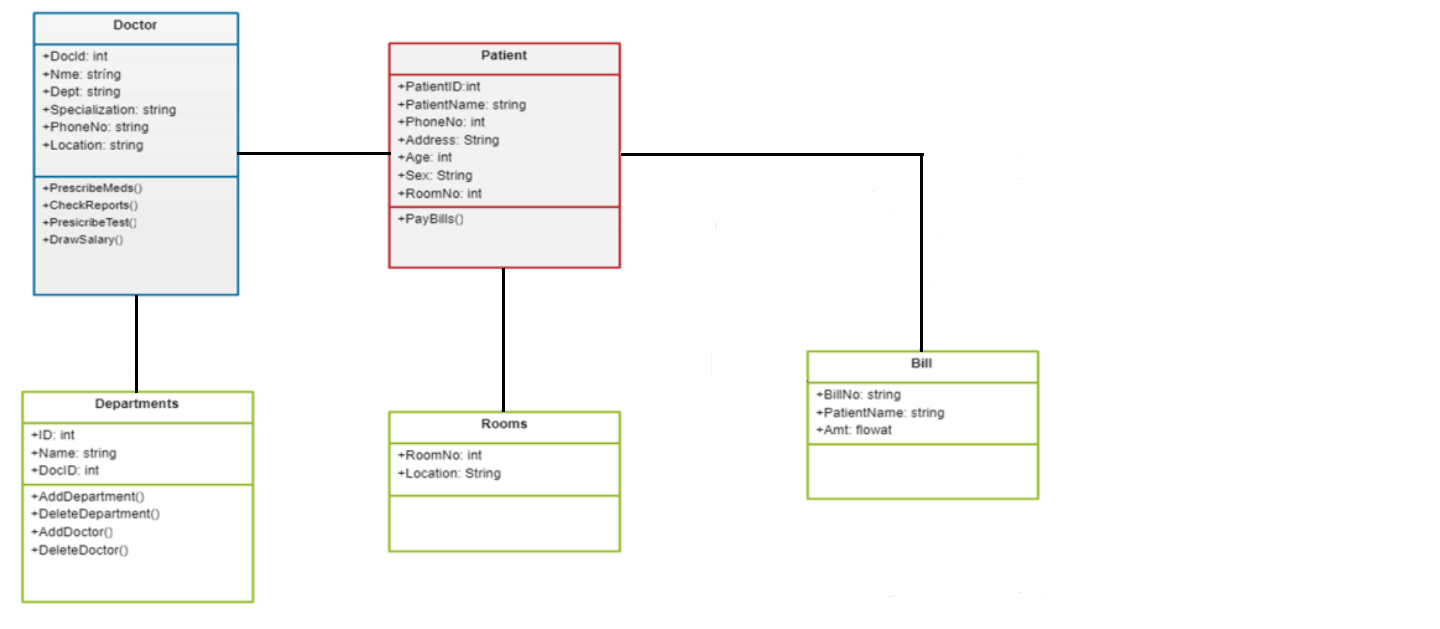
**Activity Diagram**

# 4. Design models

The design models for the Hospital Management System (HMS) encompass a range of visual representations tailored to capture distinct facets of the system's architecture and functionality. The Class Diagram delineates the object-oriented structure, illustrating classes, relationships, and inheritance. Sequence Diagrams offer insights into the dynamic interactions between system components during processes such as patient registration and billing. State Transition Diagrams illuminate the various states and transitions within the HMS, particularly focusing on appointment status changes. Data Flow Diagrams provide a comprehensive view of information flow, showcasing processes, data stores, and external entities.

## Class Diagram

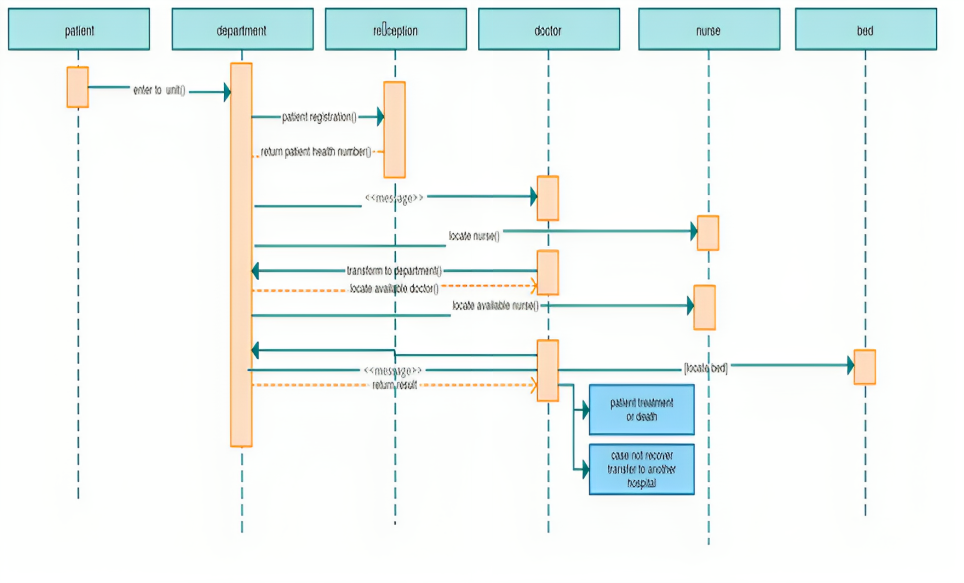
The class diagram illustrates the object-oriented design of the HMS, depicting classes like Patient, Appointment, and Billing. Relationships such as aggregation, composition, and inheritance are used to represent associations and hierarchies between classes. A Class is a category or group of things that has similar attributes and common behavior. A Rectangle is the icon that represents the class it is divided into three areas. The upper most area contains the name, the middle; area contains the attributes and the lowest areas show the operations. Class diagrams provides the representation that developers work from. Class diagrams help on the analysis side, too.



**Class Diagram**

## Sequence Diagram

A Sequence Diagram is an interaction diagram that emphasis the time ordering of messages; a collaboration diagram is an interaction diagram that emphasizes the structural organization of the objects that send and receive messages. Sequence diagrams and collaboration diagrams are isomorphic, meaning that you can take one and transform it into the other.

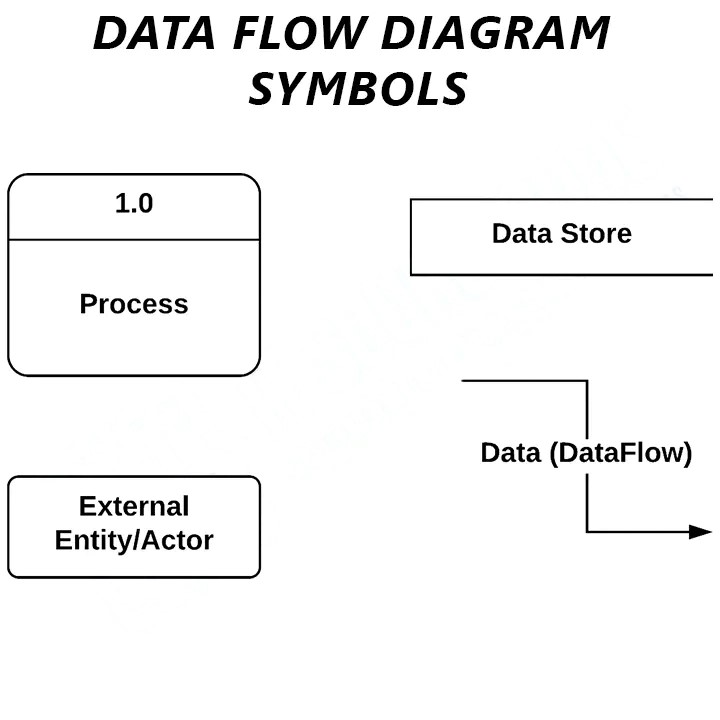


**Sequential Diagram**

## Data flow diagrams (DFD):

DFDs provide a visual representation of how data flows through the system. In the HMS, a multi-level DFD can showcase processes like Patient Registration, Appointment Scheduling, and Billing, along with data stores such as Patient Database and Inventory.

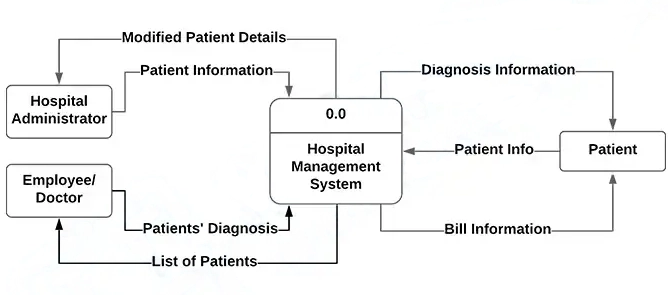
This DFD Diagram for the hospital management system uses defined symbols that present the system’s processes.

These symbols are the following:

**Data Flow Diagram**

* **External Entities** are the entry and exit points for data entering and leaving the system. Entities are referred to as terminators, sources, sinks, and actors.
* The **process** is the portion of DFD that modifies and generates output from data.
* **A Data Store (database)** is a table that stores the files or repositories for future use.
* **Data Flow** is the flow of data between external entities, processes, and data stores.
  + 1. **Level Zero Data Flow Diagram**

The Zero Level DFD for hospital management system depicts the overview of whole hospital anagement system. It is supposed to be an abstract view of overall system. This is also called as context diagram for hospital management system in which entire system is represented as single process with its relationship with external entities such as admin, staff, doctor etc.

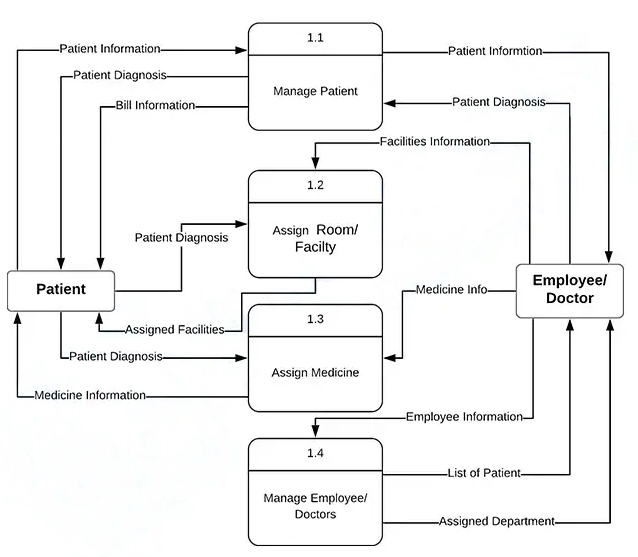


**Zero Level Data Flow Diagram**

* + 1. **Level 1 Data Flow Diagram**

The first level DFD (1St Level) of Hospital management system shows more details of processing. Level 1 DFD list all the major sub processes that makes the entire system. It also identifies data store of hospital master data that contains all records of patient, doctor, staff, room, treatment etc. that will be used to during different other process like patient registration and patient discharge.

**Level 1 Data Flow Diagram**

This Diagram shows the sub-processes of hospital management. These sub-processes comprise the important functions of the system which complete the whole project.

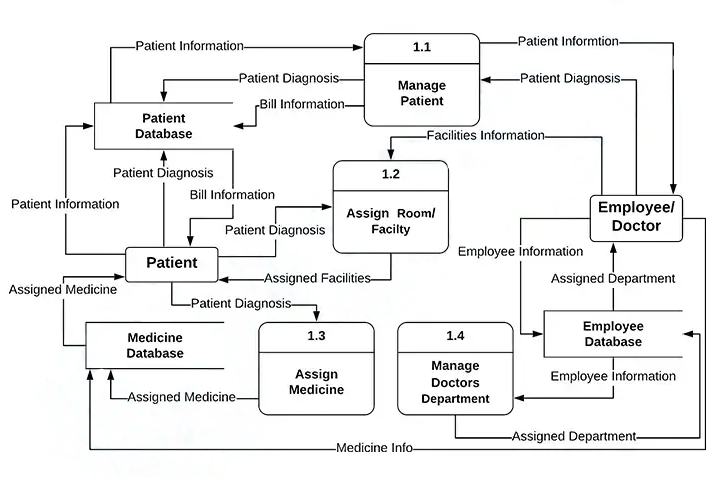
Hospital Management System Sub-processes:

* Patient Management
* Assigning Room or Facility
* Assigning Medicine
* Employee (Doctor and Nurses) Management

These processes specify the paths (flows) of data that may enter and exit the system.

* + 1. **Level 2 Data Flow Diagram**

DFD level 2 is the highest abstraction among the data flow diagram levels. It focuses more on discussing the processes of the specific sub-process in the DFD level 1.

In short, this level provides a specific explanation of the specific sub-process.

**Level 2 Data Flow Diagram**

In this example, the databases of the hospital management system are as follows:

* Patient Database
* Medicines Database
* Employee Database

Each of these databases (data store) becomes the storage of data that comes in and the source of data that comes out.

# Data Design:

The data design for the Hospital Management System (HMS) involves transforming the information domain into organized data structures, focusing on how major entities are stored, processed, and managed.

1. **Patient Data:**

* **Storage**: Patient information, including demographics and medical history, is stored in a centralized Patient Database.
* **Processing**: The system processes patient data during registration, appointment scheduling, and medical record updates.
* **Organization**: Patient data is organized by unique identifiers, facilitating quick retrieval and updating.

1. **Appointment Details:**

* **Storage**: Appointment information is stored in the Appointment Database.
* **Processing**: The system processes appointment data during scheduling, updating, and cancellation.
* **Organization**: Appointments are organized by date, time, and associated patient, ensuring a systematic arrangement.

1. **Medical Records:**

* **Storage**: Electronic health records are stored in the Medical Records Database.
* **Processing**: The system processes medical records during updates, ensuring accurate documentation of diagnoses, treatments, and test results.
* **Organization**: Medical records are organized by patient ID and chronologically for easy retrieval.

1. **Billing Information:**

* **Storage**: Financial transactions and billing details are stored in the Billing Database.
* **Processing**: The system processes billing information during service provision, generating invoices.
* **Organization**: Billing data is organized by patient ID, date, and service type.

1. **Inventory Management:**

* **Storage**: Information about medical supplies and equipment is stored in the Inventory Database.
* **Processing**: The system processes inventory data during usage, restocking, and monitoring.
* **Organization**: Inventory data is organized by item categories and quantities.

1. **Databases and Data Storage Items:**

* **Patient Database:** Stores patient demographics and medical history.
* **Appointment Database:** Stores information about scheduled appointments.
* **Medical Records Database:** Stores electronic health records.
* **Billing Database:** Stores financial transactions and billing details.
* **Inventory Database:** Stores information about medical supplies and equipment.

## Data Dictionary

This data dictionary provides a concise overview of the major entities in the HMS, including their types, descriptions, and attributes. It serves as a reference for understanding the structure and characteristics of key data elements within the system.

1. **Appointment:**
   * **Type:** Entity
   * **Description:** Represents a scheduled appointment for a patient.
   * **Attributes:**
     + AppointmentID (Integer): Unique identifier for the appointment.
     + PatientID (Integer): Identifier of the associated patient.
     + Date (Date): Date of the appointment.
     + Time (Time): Time of the appointment.
     + Status (String): Current status of the appointment (e.g., scheduled, completed).
2. **Billing Record:**
   * **Type:** Entity
   * **Description:** Records financial transactions and billing details.
   * **Attributes:**
     + TransactionID (Integer): Unique identifier for the transaction.
     + PatientID (Integer): Identifier of the associated patient.
     + Date (Date): Date of the transaction.
     + ServiceType (String): Type of healthcare service provided.
     + Amount (Float): Amount charged for the service.
3. **Inventory Item:**
   * **Type:** Entity
   * **Description:** Represents an item in the medical inventory.
   * **Attributes:**
     + ItemID (Integer): Unique identifier for the inventory item.
     + ItemName (String): Name or description of the item.
     + Category (String): Category to which the item belongs (e.g., medication, equipment).
     + QuantityAvailable (Integer): Current available quantity of the item.
4. **Medical Record:**
   * **Type:** Entity
   * **Description:** Stores electronic health records of patients.
   * **Attributes:**
     + RecordID (Integer): Unique identifier for the medical record.
     + PatientID (Integer): Identifier of the associated patient.
     + Diagnosis (String): Medical diagnosis recorded in the health record.
     + Treatment (String): Treatment information.
     + TestResults (String): Results of medical tests conducted.
5. **Patient:**
   * **Type:** Entity
   * **Description:** Represents a patient within the system.
   * **Attributes:**
     + PatientID (Integer): Unique identifier for the patient.
     + FirstName (String): First name of the patient.
     + LastName (String): Last name of the patient.
     + DateOfBirth (Date): Date of birth of the patient.
     + ContactNumber (String): Contact number of the patient.

# 6. Algorithm & Implementation

In this section, the algorithms for critical functions in the Hospital Management System (HMS) are systematically summarized. The Patient Registration algorithm creates a new patient, capturing essential details and adding them to the Patient Database. For scheduling appointments, the system checks patient existence, creates a new appointment, and updates the Appointment Database. The Update Medical Record function ensures accurate health record documentation, creating a new record linked to the patient. Generating invoices involves creating billing records tied to patient information. Inventory management is implemented by creating new inventory items with specified details. Each algorithm rigorously checks for data validity before proceeding, ensuring the integrity of the HMS operations.

**Function: Patient Registration**

* **Algorithm (PDL):**

1. *PROCEDURE RegisterPatient(FirstName, LastName, DateOfBirth, ContactNumber):*
2. *CREATE new Patient*
3. *SET Patient.FirstName = FirstName*
4. *SET Patient.LastName = LastName*
5. *SET Patient.DateOfBirth = DateOfBirth*
6. *SET Patient.ContactNumber = ContactNumber*
7. *ADD Patient to Patient Database*
8. *END PROCEDURE*

* **Function: Schedule Appointment**
  1. *PROCEDURE ScheduleAppointment(PatientID, Date, Time):*
  2. *IF PatientID EXISTS in Patient Database:*
  3. *CREATE new Appointment*
  4. *SET Appointment.PatientID = PatientID*
  5. *SET Appointment.Date = Date*
  6. *SET Appointment.Time = Time*
  7. *SET Appointment.Status = "Scheduled"*
  8. *ADD Appointment to Appointment Database*
  9. *ELSE:*
  10. *PRINT "Patient not found"*
  11. *END PROCEDURE*

# 7. Software requirements traceability matrix

The Software Requirements Traceability Matrix (RTM) serves as a vital tool to establish the connections between specified requirements and the corresponding design components detailed in this document. This tabular representation facilitates a clear understanding of how each software requirement aligns with specific design elements, ensuring a one-to-one or one-to-many relationship. The RTM acts as a navigational guide, systematically demonstrating the fulfillment of each requirement through reference to relevant design components and their associated items. This comprehensive matrix enhances transparency and aids in validating that the design comprehensively addresses the specified software requirements.

|  |  |  |  |
| --- | --- | --- | --- |
| **Req. Number** | **Ref. Item** | **Design Component** | **Component Items** |
| FR01 | Class Diagram | Patient Registration | ClassName: Patient, FunctionName: RegisterPatient |
| FR02 | DFD | Level 1: System | DiagramNumber: 1, FunctionName(s): Patient Registration, Schedule Appointment, Update Medical Record, Generate Invoice, Manage Inventory |
| FR03 | State Transition Diagram | Appointment System | DiagramNumber: 2, FunctionName(s): ScheduleAppointment (State transitions: Scheduled, Completed, Canceled) |
| FR04 | Data Flow Diagram | Level 2: Patient Registration | DiagramNumber: 3, FunctionName(s): RegisterPatient |
| FR05 | Data Flow Diagram | Level 2: Appointment Scheduling | DiagramNumber: 3, FunctionName(s): ScheduleAppointment |
| FR06 | Data Flow Diagram | Level 2: Medical Records | DiagramNumber: 3, FunctionName(s): UpdateMedicalRecord |
| FR07 | Data Flow Diagram | Level 2: Billing | DiagramNumber: 3, FunctionName(s): GenerateInvoice |

# 8. HUMAN INTERFACE DESIGN

From the user's perspective, the Hospital Management System (HMS) is designed with an intuitive and user-friendly interface, ensuring seamless interaction and efficient utilization of the system's features. Users, including healthcare professionals and administrative staff, can easily navigate through the system to accomplish various tasks.

1. **Patient Registration:**
   * *Functionality:* Users can register new patients by entering essential information such as name, date of birth, and contact number.
   * *User Interaction:* The system provides a user-friendly form with input fields for data entry. After submission, a confirmation message or any error prompts are displayed for user feedback.
2. **Appointment Scheduling:**
   * *Functionality:* Users can schedule appointments for patients, specifying the date and time.
   * *User Interaction:* An interactive calendar or date-picker simplifies the scheduling process. After successful scheduling, a confirmation message is displayed, and any conflicts or errors are communicated clearly.
3. **Update Medical Records:**
   * *Functionality:* Healthcare professionals can update patient medical records, including diagnoses, treatments, and test results.
   * *User Interaction:* A user-friendly interface presents relevant patient data, allowing easy updates. Confirmation messages validate successful updates, while error messages guide users in case of discrepancies.
4. **Generate Invoice:**
   * *Functionality:* Administrative staff can generate invoices for healthcare services rendered.
   * *User Interaction:* An invoice generation form provides a straightforward input process. Upon completion, users receive a detailed invoice, and any errors or missing information are highlighted for correction.
5. **Manage Inventory:**
   * *Functionality:* Users can manage the medical inventory, adding new items and updating quantities.
   * *User Interaction:* An inventory management interface offers easy data entry. Confirmation messages validate additions, and any issues, such as low stock, are flagged for attention.

## 7.1 Screen Images

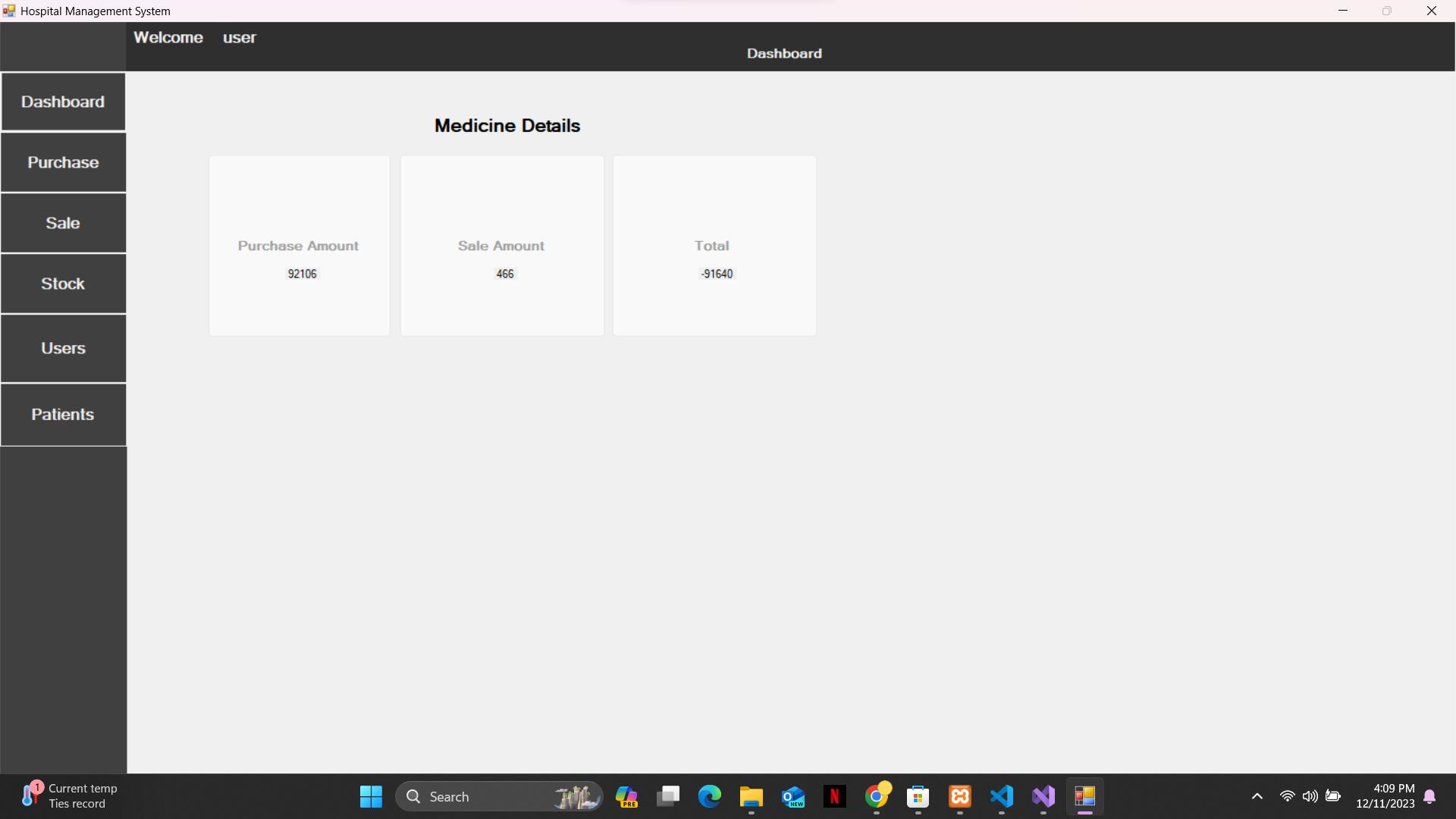


Figure 1

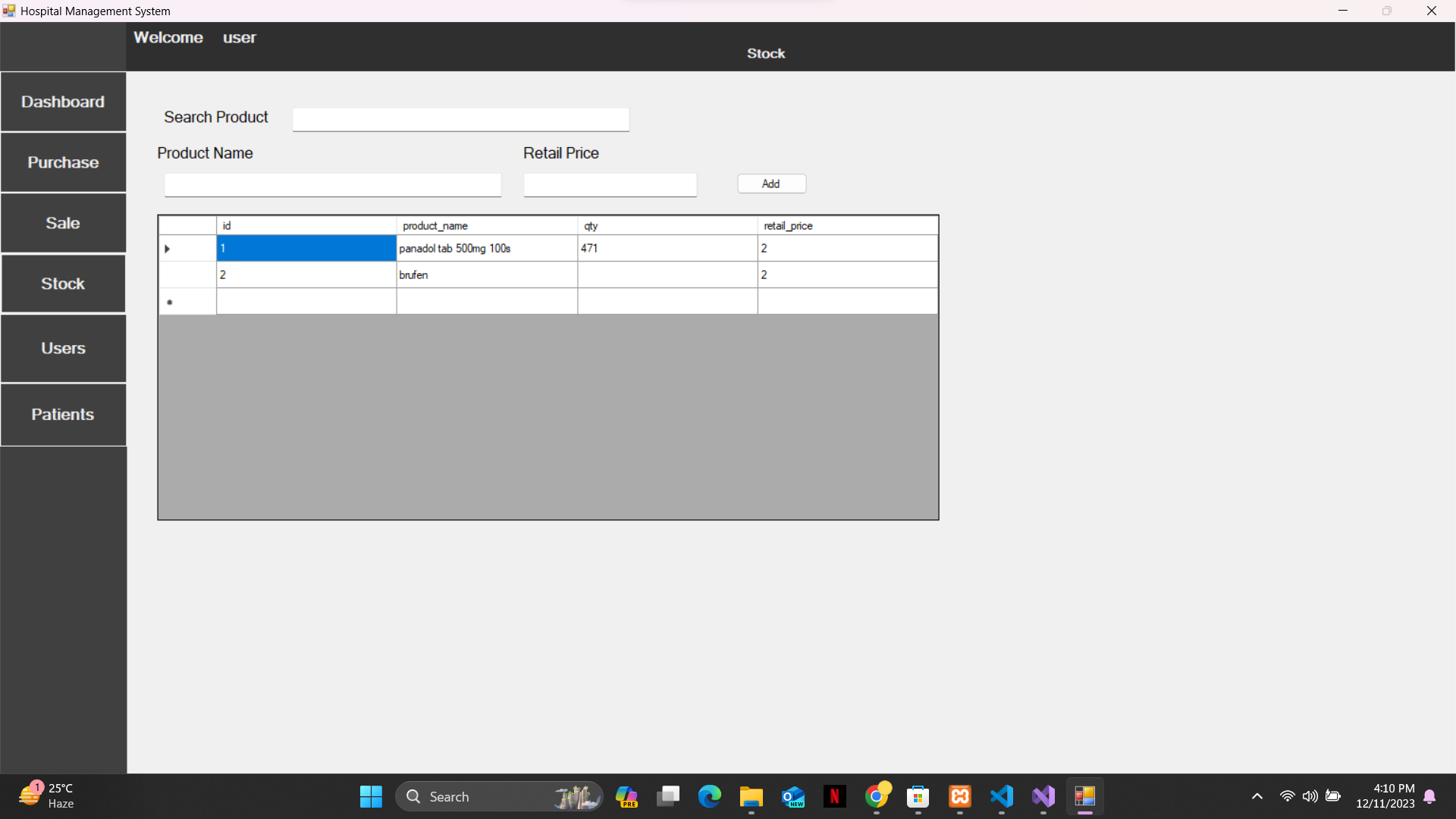


Figure 2

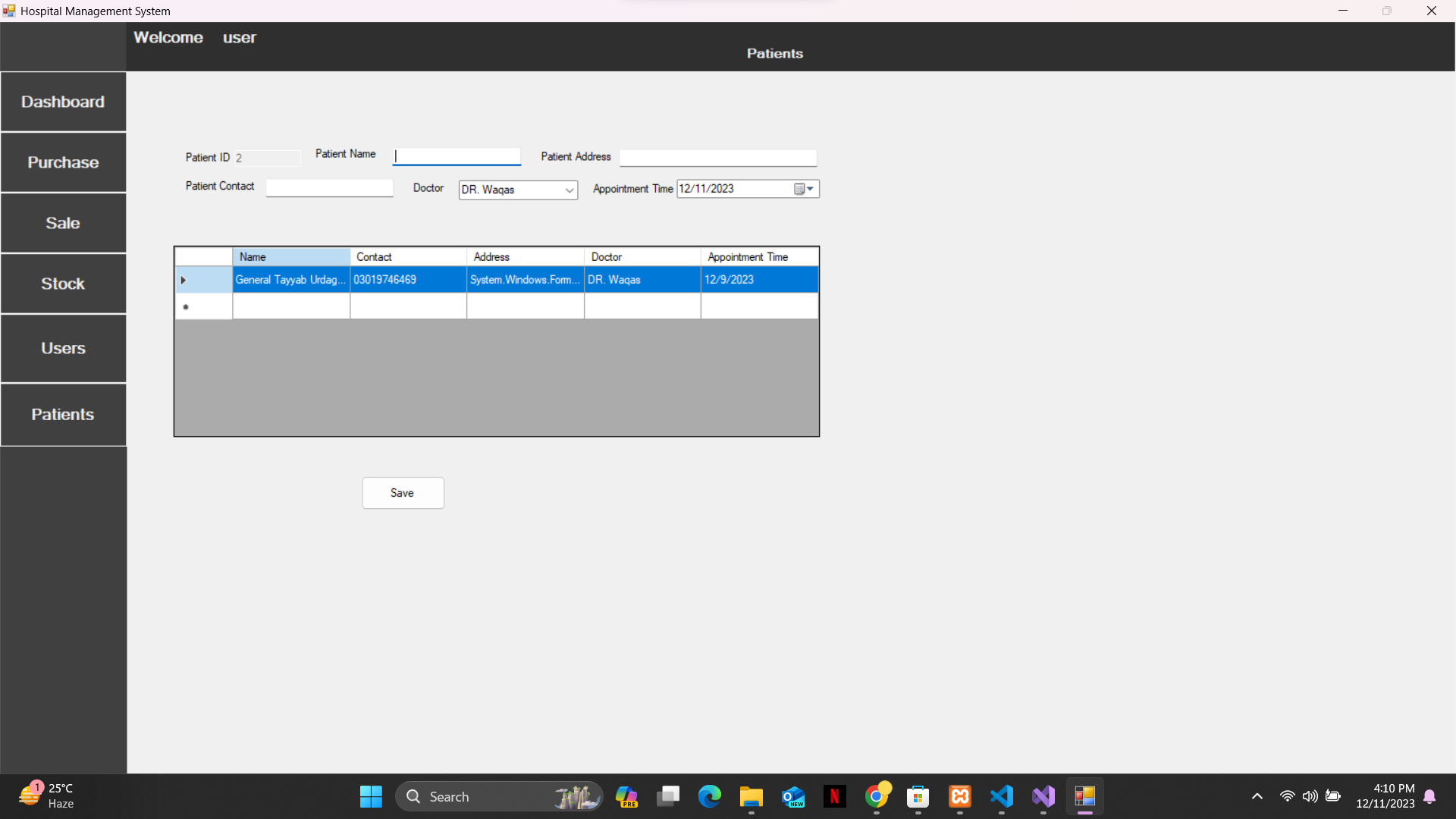


Figure 3

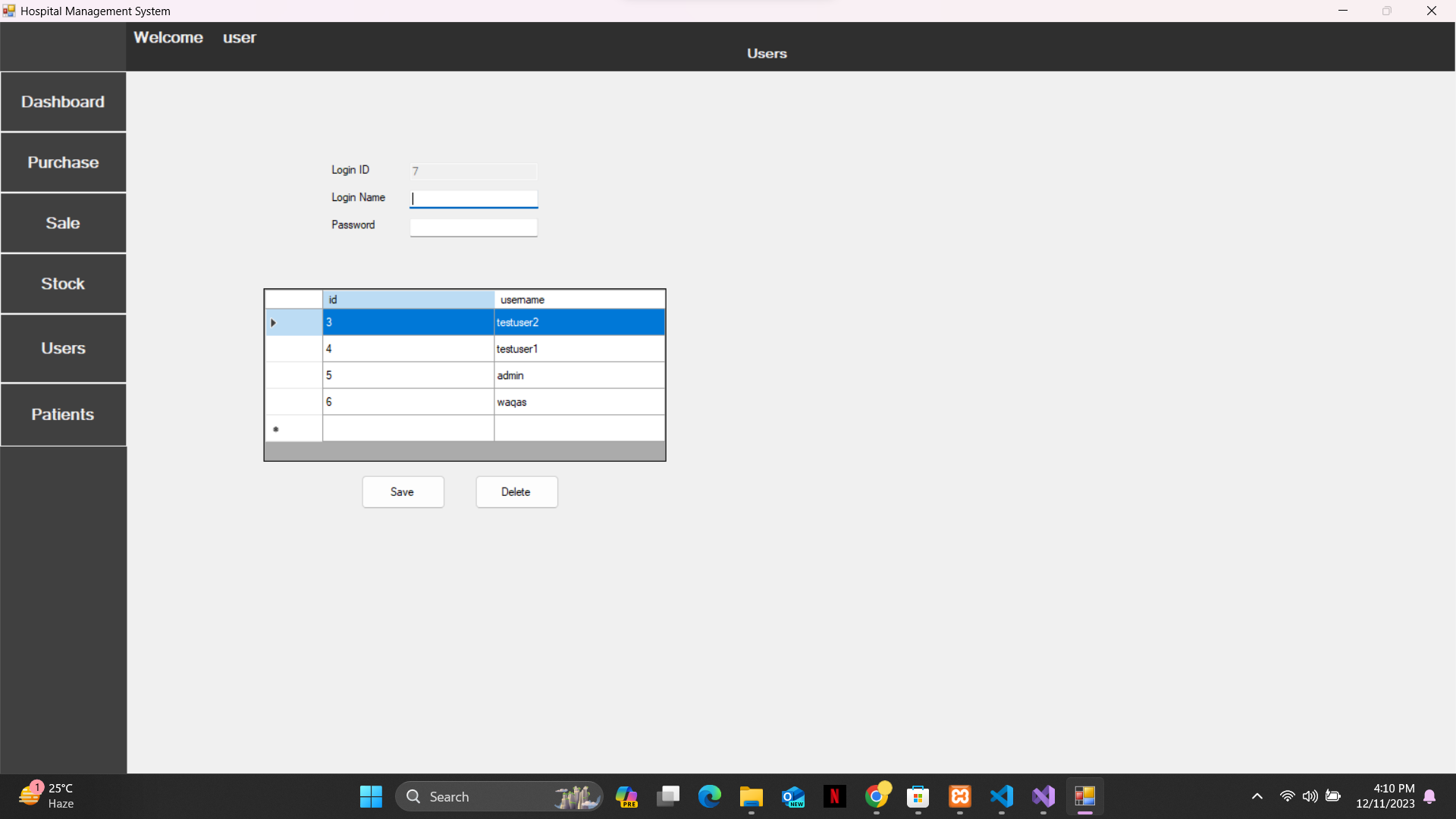


Figure 4