Project Title	**Pharmacy Management System**
Project Definition / Goal	Pharmacy Management System is a website …
Project Guide	**Rachita mem**
Operating System	Windows 7 and Upgrade
Web Server	Apache HTTP XAMPP
Frontend Used	HTML,CSS,JAVASCRIPT,BOOTSTRAP
Backend Uses	MySQL
Connectivity	PHP
Project Duration	100 Days
Team Size	2 Person
Submitted By	**Ms.Manali Jetani D.[25260152]** **Ms.Jigna Kabirpanthi A.[25260106]**
Submitted To	Shree Swaminarayan College of Computer Science. Gurukul campus, Sardarnagar, Bhavnagar. Gujrat-364001.

PREFACE

Welcome to "Pharmacy Management System",There is 606 project subject in BCA semester 6. will be very helpful in real life, so you can get experience of how to create a project in future.

A project is any undertaking, carried out individually or collaboratively and possibly involving research or design, that is carefully planned to achieve a particular goal. In a technical word, project is collection of files and folders.

A project may be a temporary or permanent. A project consists of a temporary endeavour undertaken to create a unique product, service or result.

The Pharmacy Management System outlined in this document represents a culmination of technological innovation, industry expertise, and a deep understanding of the challenges faced by modern pharmacies.

Developed with the aim of addressing these challenges head-on, the PMS provides a comprehensive suite of features and functionalities tailored to the unique requirements of pharmacy management.

Thank you

ACKNOWLEDGEMENT

LIST OF TABLES

SR.NO	Table Name
1	customers
2	invoices
3	manager
4	medicines
5	medicines_stock
6	purchases
7	sales
8	suppliers
9	user

INDEX

LIST OF FIGURES

INTRODUCATION

1.1 Background:

The Pharmacy Management System is a comprehensive solution organizing medication processes in pharmacies. Whether standalone or integrated into a hospital's system, it's pivotal in healthcare. Key components include a user-friendly interface, data entry efficiency, and robust security for patient health information. Obtained as pre-made software or provided by wholesalers, it operates across diverse settings. The system's primary goal is aiding pharmacists in safe drug delivery. Essential features encompass web-based ordering, inventory management, insightful reporting, SMS notifications, streamlined sale/billing, e-prescriptions, and perpetual inventory. These ensure optimized workflow, accuracy, and contribute to effective pharmacy operations, reinforcing the system's significance in healthcare efficiency.

1.2 Objectives:

➢ this is a very useful application for the pharmacist, which reduces the work load and it will help you to manage all of the components of the pharmacy, such as Drug Administration, Invoicing etc.

➢ That is, the increase in the efficiency of processing. This will increase the clinical efficiency and patient convenience , in view of the fact that Ethiopia is in the direction of the pharmaceutical care of the patient.

➢ It automates tasks, and account management. In a pharmacy, and the bill inspection is an essential process. In addition, the pharmacy management system that you can easily serve as the requirements for the products.

➢ This includes the safe and secure storage of the medicinal product details, as well as a quick search, removal, and updating of products. The pharmacy management system is developed in order to ensure effective and clear the data storage and manipulation, and precision, with, Instagram, products, and medical products.

➢ The pharmacy management system is easy-to-use, so that the user can run a pharmacy without ambiguity. This is the project subject to a pharmacy management system with a high degree of minimisation of time and resources, and

with the help of that by looking at the drug information, you can use the data in the shortest amount of time possible.

➢ The main objectives of the ICP are the automation of pharmacy organisations in the creation of a good quality by minimizing or eliminating the time of the loss, and the removal of substances such as paper to add data to it, I know that there have been paper-based, to reduce lack of health care by providing relevantly.

1.3 Purpose and Scope:

Pharmacy management system is a management system that has been developed to improve the accuracy and increase the safety and efficiency of the pharmacy store.

1.3.1 PURPOSE :

It has been one of the IT-systems, which helps pharmacists improve the supply, cost, health, safety, security, etc., etc. the management of the inventory and the sales activities, so as the user, so that the production dates, and due dates to individual products, or medications.

The product displays a list of the dates after a specific period of time prior to the expiry date of the product. It will also include the launch of a new batch of drugs for a certain period of time and at the entrance to t• he pharmacy to get to the merchant. Each month's apothecary reports on the promotion of medicinal products, to receive information about the products. Expiration date is the date of retail purchase, with the remaining amount of drug in the place of the drug in the pharmacy.

1.3.2 SCOPE :

Now We are ready with 80% of all of the options available. We have 4 different types of advertising, such as Customer, Pharmaceutical industry, Pharmacy, Shops, and etc. We have a complete order system, a billing system, a system for the registration of orders in the system for your business, and pharmacy stores.

An order is a system for the customers, pharmacies, shops, etc. But there are still some jobs left which we will do in the future. Some works like –

1. Pharmacy ranking system.
2. Pharmacy safety system.

SOFTWARE DEVELOPMENT LIFE CYCLE

[SDLC]

2.1 What is SDLC ?

➢ SDLC is known as "System Analysis Life Cycle".

➢ System development, a process consisting of the two major step of system analysis and design, starts when management or sometimes systems development person realize that particular business system needs improvement.

➢ "The system development life cycle (SDLC) method is classically thought of as the set of activities that analysist, designers and users carry out to develop and implement an information system."

➢ These section study each of the eight activities that make up the systems development life cycle.
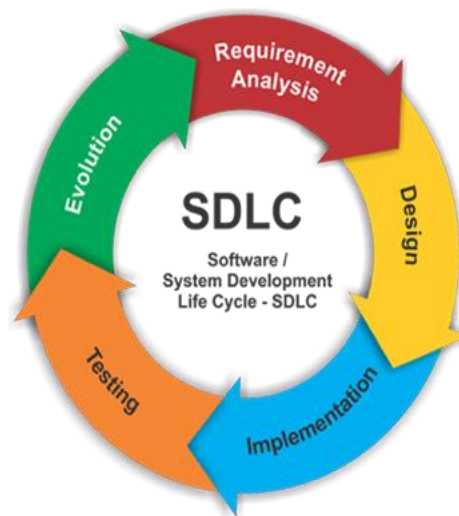
[Figure-2.1 **S**oftware **D**evelopment **L**ife **C**ycle (**SDLC**)]

2.2 Advantages Of SDLC :

- **Structured Approach:**

  SDLC offers a structured approach to software development, efficient planning, and organization of tasks for developers. This structured methodology not only minimizes errors but also enhances productivity, ensuring the timely delivery of high-quality software.

- **Risk Management:**

  A great aspect of SDLC is its ability to identify and effectively manage risks inherent in the software development process. By pinpointing potential risks early on, developers can proactively address and mitigate them, ultimately diminishing the overall risk associated with software development

  .

- **Consistency:**

  SDLC establishes a foundation for consistency in software development through a standardized framework and methodology. This consistency is instrumental in elevating the quality of the software, guaranteeing that the final product aligns seamlessly with client expectations.

- **Collaboration:**

  SDLC fosters a collaborative environment among team members by providing a common application framework and language for communication. This collaborative synergy not only enhances the overall quality of the software but also ensures that the end product precisely fulfills the client's requirements.

- **Cost-Effective:**

    SDLC proves to be a cost-effective approach by identifying potential issues early in the development process through the use of prototyping tools like Figma and others. Early issue detection allows developers to take proactive measures, significantly reducing overall development costs. This cost-effective attribute positions SDLC as a strategic choice in the realm of software development.

2.3 Phases of SDLC :

1. Planning and Requirement Analysis

Planning is a crucial step in everything, just as in software development. In this same stage, requirement analysis is also performed by the developers of the organization. This is attained from customer inputs, and sales department/market surveys.

The information from this analysis forms the building blocks of a basic project. The quality of the project is a result of planning. Thus, in this stage, the basic project is designed with all the available information.

2. Defining Requirements

In this stage, all the requirements for the target software are specified. These requirements get approval from customers, market analysts, and stakeholders.

This is fulfilled by utilizing SRS (Software Requirement Specification). This is a sort of document that specifies all those things that need to be defined and created during the entire project cycle.

3. Designing Architecture

SRS is a reference for software designers to come up with the best architecture for the software. Hence, with the requirements defined in SRS, multiple designs for the product architecture are present in the Design Document Specification (DDS).

This DDS is assessed by market analysts and stakeholders. After evaluating all the possible factors, the most practical and logical design is chosen for development.

4. Developing Product

At this stage, the fundamental development of the product starts. For this, developers use a specific programming code as per the design in the DDS. Hence, it is important for the coders to follow the protocols set by the association. Conventional programming tools like compilers, interpreters, debuggers, etc. are also put into use at this stage. Some popular languages like C/C++, Python, Java, etc. are put into use as per the software regulations.

5. Product Testing and Integration

After the development of the product, testing of the software is necessary to ensure its smooth execution. Although, minimal testing is conducted at every stage of SDLC. Therefore, at this stage, all the probable flaws are tracked, fixed, and retested. This ensures that the product confronts the quality requirements of SRS.

6. Deployment and Maintenance of Products

After detailed testing, the conclusive product is released in phases as per the organization's strategy. Then it is tested in a real industrial environment. It is important to ensure its smooth performance. If it performs well, the organization sends out the product as a whole. After retrieving beneficial feedback, the company releases it as it is or with auxiliary improvements to make it further helpful for the customers.

REQUIREMENT & ANALYSIS

3.1 Problem Definition :

➢ First, selecting the suitable medicine for the type of illness is usually take time and makes the patients or customer waiting. Therefore, the time is waste for the customer to be waiting.

➢ Second, for medicine stock management, the pharmacist must check it manually and no warning message for medicine that decrease to finish. So, PMS the provide advantage to setup process in selecting and alert program to warn about medicine stock.

➢ Finally, yet important, no analyses are done for frequency type of medicine usually buy by the customer or patient at that area. This is also important to determine the medicines that are demanded more from the customers so that pharmacist can be prepared to order more for that type of medicine.

3.2 Requirement Specification :

The requirements for the E-Health Care Management System include:

➢ The Pharmacy Management System (PMS) is designed with a set of comprehensive requirements to ensure its effectiveness and efficiency in organizing and optimizing the medication processes within pharmacies.

➢ The user interface demands an intuitive design for easy navigation and accessibility, facilitating efficient data entry and retrieval.

3.3 Software and Hardware Requirements :

3.3.1 Software Requirements:

The software requirements include a secure database system, development frameworks, communication tools, and encryption protocols.

	Requirement
Front-end Tools	HTML,CSS,JAVASCRIPT,BOOTSTARP
Back-end Tools	PHP,MYSQL(DATABASE)
Editor	VS CODE,NOTEPAD++
Browser	GOOGLE CHROME,MICROSOFT EDGE
Operating System	WINDOWS 10

3.3.1 Hardware Requirements:

The hardware requirements include servers for data storage, networking equipment, and devices for end-users.

Component	Requirement
Processor	Intel core I3,1.70GHz
RAM	512 MB
Hard Disk	20 GB

3.4 Planning and Scheduling :

The process of planning primarily deals with selecting the appropriate policies and procedures in order to achieve the objectives of the project. Scheduling converts the project action plans for scope, time cost and quality into an operating timetable.
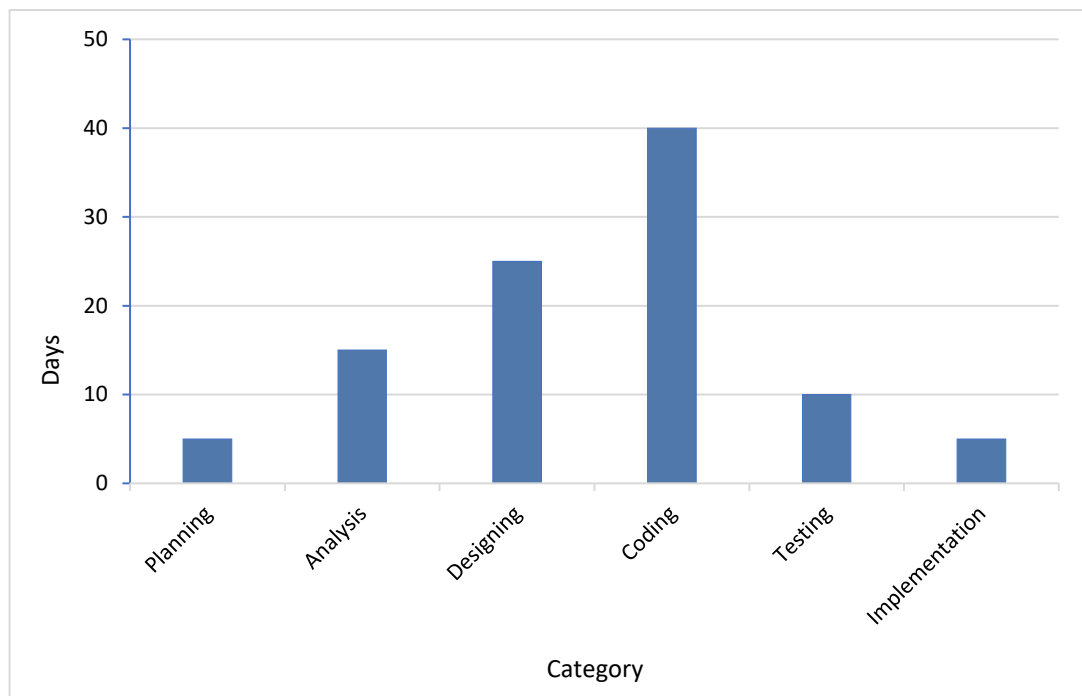
[**Figure-3.4 Planning and Scheduling**]

3.5 Functional requirement:

Functional requirements for a Pharmacy Management System (PMS) outline the specific features and capabilities that the system must possess to effectively support pharmacy operations. Here's a comprehensive list of functional requirements typically expected in a PMS:

3.5.1 Inventory Management:

- Track inventory levels of medications, medical supplies, and related products.
- Monitor stock levels and generate alerts for low stock or expiring products.
- Automate inventory replenishment processes.
- Manage product recalls and returns efficiently.
- Provide barcode scanning and integration for inventory control.

3.5.2 Drug Database:

- Maintain a comprehensive database of medications with detailed information such as drug names, dosages, indications, and interactions.
- Regularly update the database to ensure accuracy and completeness.
- Allow for easy searching and retrieval of drug information.

3.5.3 Point-of-Sale (POS) Integration:

- Support sales transactions for prescription and over-the-counter medications.
- Process payments securely.
- Integrate with insurance systems for claims processing.
- Generate invoices and receipts for transactions.

3.5.4 Billing and Insurance:

- Generate billing statements for patients.
- Verify insurance coverage and eligibility in real-time.

- ➢ Submit insurance claims electronically.
- ➢ Handle co-payments, deductibles, and other financial aspects of transactions.

3.5.5 Drug Interaction Checking:

- ➢ Perform real-time checks for potential drug interactions and contraindications.
- ➢ Provide alerts to pharmacists for any identified risks.
- ➢ Offer recommendations for alternative medications when necessary.

3.5.6 Reporting and Analytics:

- ➢ Generate reports on sales trends, inventory turnover, prescription volume, and financial performance.
- ➢ Provide data visualization tools for easy analysis.
- ➢ Support custom report generation based on specific pharmacy needs.

3.5.7 Customer Relationship Management (CRM):

- ➢ Manage customer profiles, preferences, and purchase history.
- ➢ Facilitate personalized interactions and communication with patients.
- ➢ Support loyalty programs and promotions.

By meeting these functional requirements, a Pharmacy Management System can effectively support pharmacy operations improve overall efficiency and accuracy in medication management.

3.6 Non-functional requirement:

Non-functional requirements for a Pharmacy Management System (PMS) specify the criteria that describe how the system performs rather than what it does. These requirements focus on aspects such as performance, security, usability, reliability, and scalability. Here's a list of non-functional requirements typically associated with a PMS:

3.6.1 Performance:

- Response Time: The system should respond to user actions promptly, with minimal delay.
- Throughput: It should handle a high volume of transactions efficiently, especially during peak times.
- Scalability: The system should be scalable to accommodate growth in data volume and user load.
- Resource Utilization: Efficient utilization of system resources such as CPU, memory, and storage.

3.6.2 Security:

- Access Control: Role-based access control (RBAC) should be implemented to restrict access to authorized users.
- Audit Trails: Maintain detailed logs of user activities for auditing and accountability purposes.
- Data Integrity: Ensure the accuracy and consistency of data through measures like checksums and data validation.

3.6.3 Reliability:

- Fault Tolerance: The system should continue to function properly in the event of hardware or software failures.
- High Availability: Minimize downtime by ensuring the system is available and accessible when needed.
- Backup and Recovery: Regularly backup data and provide mechanisms for data recovery in case of data loss or corruption.

3.6.4 Usability:

- User Interface (UI) Design: The interface should be intuitive, user-friendly, and visually appealing.
- Accessibility: Ensure the system is accessible to users with disabilities by complying with accessibility standards.

3.6.5 Compatibility:

- Platform Compatibility: The system should be compatible with various operating systems and web browsers.
- Integration Compatibility: It should seamlessly integrate with other healthcare systems, such as electronic health records (EHRs) and laboratory information systems (LIS).
- Device Compatibility: Support for different devices including desktops, laptops, tablets, and mobile phones.

3.6.6 Regulatory Compliance:

- HIPAA Compliance: Ensure compliance with the Health Insurance Portability and Accountability Act (HIPAA) regulations for patient data protection.
- FDA Compliance: Adhere to regulations set by the Food and Drug Administration (FDA) for pharmaceutical products and electronic prescribing.

3.6.7 Performance Efficiency:

- Optimization: Optimize system performance through techniques such as caching, indexing, and query optimization.
- Minimize Bottlenecks: Identify and eliminate performance bottlenecks to ensure smooth operation.

3.6.8 Documentation:

- Comprehensive Documentation: Provide detailed documentation including user manuals, system architecture, and troubleshooting guides.
- Version Control: Maintain version control for documentation to ensure accuracy and relevance.
- By addressing these non-functional requirements, a Pharmacy Management System can ensure robustness, security, usability, and performance, ultimately enhancing the overall effectiveness of pharmacy operations.

3.7 FACT FINDING TECHNIQUES:

Fact-finding is the formal process of using research, interviews, questionnaires, sampling, and other techniques to collect information about systems, requirements, and preferences. It is also called information gathering or data collection.A database developer commonly uses several fact-finding techniques during a single database project. There are four widely used fact-finding techniques:

1. Interviewing:

Interviewing is the most frequently used, and usually the most useful, fact-finding procedure used. We can interview to collect information from person face-to-face. There can be several objectives for using interviewing, such as finding out facts, verifying those facts, clarifying these released facts, generating enthusiasm, getting the end user involved, identifying requirements, and gathering ideas and opinions. However, using the interviewing practice must require proper communication skills for dealing effectively with people who have different values, priorities, opinions, motivations, and personalities.

2. Observation:

Observing the enterprise in action: Observation is one of the most successful fact-finding techniques carried out for understanding a system. Using this technique, it is achievable to either participate in or observe a person perform activities to learn about the system.

3. Questionnaires:

Another fabulous fact-finding method is to conduct surveys through questionnaires. Questionnaires are special-purpose documents that allow facts to be gathered from a large number of people while upholding some control over their responses. When dealing with a large number of listeners or audience, no other fact-finding technique can tabulate the same facts so efficiently. There are two types of questions that can be asked in a questionnaire, namely freeformat and fixed-format. Free format questions offer the respondent greater freedom inputting answers. Fixed-format questions require specific responses from individuals, and for the given question, the respondent must choose from the available answers.

4. Research:

A useful fact-finding technique is to research the application or the problem that you are dealing with and want to put within a database. Computer trade journals, reference books, and the Internet are good sources of information that can make available the vast quantity of information on how others have solved similar problems/issues plus whether or not any software packages exist to resolve or even partially solve your current problem.

3.8 FEASIBILITY STUDY:

Feasibility Analysis is the process of determination of whether or not a project is worth doing. Feasibility studies are undertaken within tight time constraints and normally culminate in a written and oral feasibility report.

The objective behind the feasibility study is to create the reasons for developing the software that is acceptable to users, flexible to change and conformable to established standards.

There are three aspects in feasibility study

Feasibility study protion of the priliminary investigation.

Technical

Economic

Operational
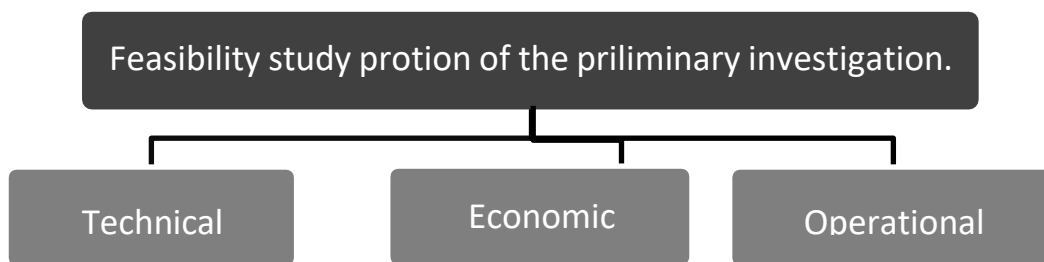
[Figure-3.8 Feasibility study]

1) **Technical Feasibility:**

Technical feasibility determines whether the work for the project can be done with the existing equipment, software technology and available personnel. Technical feasibility of proposed project refers to the software and hardware requirements.

The project is developing using HTML, CSS, PHP and other font end tool and MySQL is used for DBMS. The proposed project can beimplementing on maximum browser support.

2) **Economic Feasibility:**

This feasibility determines whether there are sufficient benefits in creating to make the cost acceptable, or is the cost of the system too high. The software using to develop the proposed system is cost efficient. HTML, CSS and PHP tools are available for free and open source.

3) Operational Feasibility:

Operational feasibility assesses the range in which the required software performs a series of levels to solve business problems and customer requirements. This feasibility is dependent on human resources (software development team) and involves visualizing whether the software will operate after it is developed and be operative once it is installed.

This system provides various functions, it is important to measure the feasibility of each function for measuring overall feasibility of this system. Status check, request form, adds mechanics etc. are easily operated using proposed project.

3.9 TOOLS & TECHNOLOGY:

3.9.1 Introduction to PHP (HYPERTEXT PRE-PROCESSOR):

PHP is a scripting language originally designed for producing dynamic web pages. It has evolved to include a command line interface

capability and can be used in standalone graphical applications.

While PHP was originally created by Rasmus Lerdorf in 1995, the main implementation of PHP is now produced by the PHP Group and serves as the defector standard for PHP as there is no Formal specifications.

[Figure-3.9.1 PHP]

PHP is free software released under the PHP license; however it is incompatible with the GNU general public license (GPL), due to restrictions on the uses of the term PHP.

PHP is a widely-used general-purpose scripting language that is especially suited for web development and can be embedded into HTML. PHP has grown from simple beginnings to a full-fledged object oriented language that can run both windows and UNIX/Linux platforms. The main advantages of using PHP over other platforms, such as Java or .net, are that it is smaller, much simpler to install, and more lightweight, needing only a fraction of the memory of the Java runtime of the .Net CLR. Science it is an open source language and we do not have to purpose it, it seemed best choice for the development of our project as we had to maintain a very low budget throughout out project.

FEATURE OF PHP:

- Simplicity
- Portability
- Speed
- Open source
- Extensible

3.9.2 Introduction to JAVASCRIPT:

Java script is a scripting language used to enable programmatic access to objects within other applications. It was developed by Brendan Eich.

It is primarily used in the form of client-side JavaScript for the development of dynamic website. Java script is a dialect of the ECMA script standard and is characterized as a dynamic, weakly typed, prototype-based language with first-class functions.

[Figure-3.9.2 JAVASCRIPT]

Java script was influenced by many languages and was designed to look like java, but be easier for non-programmers to work with.Java script, despite the name, is essentially unrelated to the java programming language even though the tow does have superficial similarities. Both languages use syntaxes influenced by that of C syntax, and java script copies many java names and naming conventions. The language's name is the result of a co-marketing deal between Netscape and Sun, in exchange for Netscape building sun's Java runtime with their then-dominant browser. The key design principles within Java Script are inherited from the self and scheme programming languages.

FEATURES OF JAVASCRIPT:

- Light Weight Scripting language

- Dynamic Typing

- Object-oriented programming support

- Functional Style

- Platform Independent

- Prototype-based

- Interpreted Language

3.9.3 Introduction to MYSQL:

MySQL is a high-performance, multiuser relational database management system for database driven software applications. Designed around three fundamental principles-speed, stability and ease of use, and freely available under the GNU (General Public License). MySQL has been dubbed "the world's most popular open-source database" by its parent company, MySQL AB.

[Figure-3.9.3 MYSQL]

Today, MySQL is available for a wide variety of platforms, including Linux, Mac OS and Windows.

FEATURE OF MYSQL:

- Speed

- Reliability

- Security

- Scalability and portability

- Ease of use

- Compliance with existing standards

- Wide application support

3.9.4 Introduction to HTML(HYPER TEXT MARKUP LANGUAGE):

HTML stands for Hyper Text Markup Language. HTML is not only way to present information on the web, but it's the glue that holds everything to gather. In addition to being a markup language for displaying text, images and multimedia, HTML provides instructions to web browsers in order to control how documents are viewed and how they relate to each other. For all its simplicity, HTML is a very powerful language.

[Figure-3.9.4 HTML]

➢ HTML stands for Hyper Text Markup Language.
➢ An HTML file is a text file containing small markup tags
➢ The markup tags tell the web browser how to display to the page.
➢ An HTML file must have an HTM or HTML extension.
➢ An html file can be created using a simple text editor.

3.9.5 Introduction to CSS(CASCADING STYLE SHEET):

Cascading Style Sheets (CSS) is a style sheet language used to describe the presentation (that is, the look and formatting) of a document written in a Markup language. It's most common application is to style web pages written in HTML and XHTML, but the language can be applied to any kind of XML document, including SVG and XUL.

CSS is designed preliminary to enable the separationif document content (written in HTML or a similar markup language) from document presentation, including elementssuch as the colours, fonts and layout.

[Figure-3.9.5 CSS]

This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple pages to share formatting, and reduce complexity and repetition in the structural content (such as by allowing for table less web design).

CSS can also allow the same markup page to be presented in different styles for different rendering methods, such as on-screen, in print, by voice (when read out by a speech-based browser or screen reader) and on braille-based, tactile devices.

3.9.6 Introduction to BOOTSTRAP:

Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first web sites. It solves many problems which we had once, one of which is the cross-browser compatibility issue.

Nowadays, the websites are perfect for all the browsers (IE, Firefox and Chrome) and for all sizes of screens (Desktop, Tablets, and Phones).

[Figure-3.9.6 BOOTSTRAP]

All Thanks to Bootstrap developers -Mark Otto and Jacob Thornton of Twitter, though it was later declared to be an open-source project.

Why Bootstrap?

- ➢ Faster and Easier Web-Development.
- ➢ It creates Platform-independent web-pages.
- ➢ It creates Responsive Web-pages.
- ➢ It designed to be responsive to mobile devices too.

SYSTEM DESIGN

4.1 Database Design :

4.1.1 Introduction of Data Dictionary

A Data Dictionary is a collection of names, definitions, and attributes about data elements that are being used or captured in a database, information system, or part of a research project. It describes the meanings and purposes of data elements within the context of a project, and provides guidance on interpretation, accepted meanings and representation. A Data Dictionary also provides metadata about data elements. The metadata included in a Data Dictionary can assist in defining the scope and characteristics of data elements, as well the rules for their usage and application.

What Is Data Dictionary?

➢ A data dictionary contains metadata i.e data about the database.

➢ The users of the database normally don't interact with the data dictionary, it is only handled by the database administrators.

➢ The data dictionary is very important as it contains information such as what is in the database, who is allowed to access it, where is the database physically stored etc.

The data dictionary generally contains the information about the following:

➢ Names of all the database tables and their schemas.

- ➢ Details about all the database, such as their owners, their security constraints, when they were created etc.
- ➢ Physical information about the table such as where they are stored and how.
- ➢ Table constraints such as primary key attributes and foreign key information etc.
- ➢ Information about the database views that are visible.

4.1.2 Types of Data Dictionary:

- ❖ The different type of data dictionary are as follows:

1. **Active Data Dictionary**:

- ➢ If the structure of the database or its specifications change at any point of time, it should be reflected in the data dictionary.

- ➢ This is the responsibility of the database management system in which the data dictionary resides.

- ➢ So, the data dictionary is automatically updated by the database management system when any changes are made in the database.

- ➢ This is known as an active data dictionary as it is self-updating.

2. **Passive Data Dictionary:**

- ➢ This is not as useful or easy to handle as an active data dictionary.

➤ A passive data dictionary is maintained separately to the database whose contents are stored in the dictionary.

➤ That means that if the database is modified the database dictionary is not automatically updated as in the case of Active Data Dictionary.

4.1.3 Why data dictionary is important?

➤ The main reason companies use data dictionaries is to document and share data structures and other information for all involved with a project or database.

➤ Using a shared dictionary ensures the same quality, meaning, and relevance for all data elements for all team members.

➤ The data dictionary will define conventions for the project and consistency throughout the dataset.

➤ Without a data dictionary, there's a higher risk of losing crucial information in translation and transition. Using a data dictionary also helps teams analyse the data easier later on.

4.1.4 Data Dictionary

Database Name : pharmacy

1. Table Name : customer

This Table Is Use To Store Detail Of customer In System .This Table Also Use to Login On System.

Sr.no	Field Name	Data Type	Size	Constraint	Description
1	ID	INT	3	Primary Key, Auto Increment	Use to Store Customer Id
2	NAME	VARCHAR	20	NOT NULL	Use to Store Customer Name
3	CONTACT_NUMBER	VARCHAR	12	NOT NULL, Unique	Use to Store Customer Contact Number
4	ADDRESS	VARCHAR	50	NOT NULL	Use to Store Customer Address
5	username	VARCHAR	15	NOT NULL, Unique	Use to Store Customer username
6	password	VARCHAR	50	NOT NULL	Use to Store Customer Password

2. **Table Name :** invoices

Invoices table is storing the information of invoices.it store invoice_id,net_total,total_amount and other information.

Sr.no	Field Name	Data Type	Size	Constraint	Description
1	INVOICE_ID	INT	3	Primary Key, Auto Increment	Use to Store Invoice Id
2	NET_TOTAL	DOUBLE		NOT NULL	Use to Store Net Total Of Invoice
3	INVOICE_DATE	DATE		NOT NULL	Use to Store Invoice Date
4	CUSTOMER_ID	INT	3	REFERENCE (TABLE customer)	Use to Store Customer Id
5	TOTAL_AMOUNT	DOUBLE		NOT NULL	Use to Store Total Amount Of Invoice
6	TOTAL_DISCOUNT	DOUBLE		NOT NULL	Use to Store Discount On Invoice

3. Table Name : manager

This Table Is Use To Store Detail Of Manger In System .This Table Also Use to Login On System.

Sr.no	Field Name	Data Type	Size	Constraint	Description
1	id	INT	3	Primary Key, Auto Increment	Use to Store Manager Id
2	username	VARCHAR	15	NOT NULL, UNIQUE	Use to Store Manager username
3	email	VARCHAR	30	NOT NULL, UNIQUE	Use To Store Email Of Manager
4	address	VARCHAR	50	NOT NULL	Use to Store Manager Address
5	password	VARCHAR	50	NOT NULL	Use to Store Manager Password
6	name	VARCHAR	20	NOT NULL	Use to Store Manager Name
7	Status	INT	1	NOT NULL	Use To Store Status Of This Manager
8	Contact_no	VARCHAR	12	NOT NULL, UNIQUE	Use to Store Manager Contact Number

4. **Table Name :** medicines

This Table Is Use To Store Detail Of Medicine In System .

Sr.no	Field Name	Data Type	Size	Constraint	Description
1	ID	INT	3	Primary Key, Auto Increment	Use to Store Medicine Id
2	NAME	VARCHAR	50	NOT NULL	Use to Store MedicineName
3	PACKING	VARCHAR	20	NOT NULL	Use to Store Medicine Packing
4	GENERIC_NAME	VARCHAR	30	NOT NULL	Use to Store Generic Name Of Medicine
5	SUPPLIER_NAME	VARCHAR	20	NOT NULL	Use to Store Medicine Supplier

5. Table Name : medicines_stock

This Table Is Use To Store Detail Of Medicine Stock In System.

Sr.no	Field Name	Data Type	Size	Constraint	Description
1	ID	INT	3	Primary Key, Auto Increment	Use to Store Medicine Stock Id
2	NAME	VARCHAR	50	NOT NULL	Use to Store Medicine Name
3	BATCH_ID	VARCHAR	10	NOT NULL	Use to Store Medicine Batch Id
4	EXPIRY_DATE	VARCHAR	10	NOT NULL	Use to Store Expiry Date Of Medicine
5	QUANTITY	INT	5	NOT NULL	Use to Store Medicine Quantity
6	MRP	DOUBLE		NOT NULL	Use to Store Medicine MRP
7	RATE	DOUBLE		NOT NULL	Use to Store Medicine Rate

6. Table Name : purchases

This Table Is Use To Store Detail Of Purchase Of Medicine In System .

Sr.no	Field Name	Data Type	Size	Constraint	Description
1	VOUCHER_NUMBER	INT	3	Primary Key, Auto Increment	Use to Store Voucher Id/Number
2	INVOICE_NUMBER	INT	3	REFERENCE (TABLE invoice)	Use to Store Invoice id
3	SUPPLIER_NAME	VARCHAR	20	NOT NULL	Use to Store Medicine Supplier
4	PURCHASE_DATE	VARCHAR	10	NOT NULL	Use to Store Purchase Date
5	TOTAL_AMOUNT	DOUBLE		NOT NULL	Use to Store Total Amount
6	PAYMENT_STATUS	VARCHAR	20	NOT NULL	Use To Store Payment Status

7. Table Name : sales

This Table Is Use To Store Detail Of Sales Of Medicine In System.

Sr.no	Field Name	Data Type	Size	Constraint	Description
1	ID	INT	3	Primary Key, Auto Increment	Use to Store Sales Id
2	CUSTOMER_ID	INT	3	REFERENCE (TABLE customer)	Use to Store Customer Id
3	INVOICE_NUMBER	INT	3	REFERENCE (TABLE invoice)	Use To Store Invoice id
4	MEDICINE_NAME	VARCHAR	50	NOT NULL	Use to Store Medicine Name
5	BATCH_ID	VARCHAR	10	NOT NULL	Use to Store Medicine Batch Id
6	EXPIRY_DATE	VARCHAR	10	NOT NULL	Use to Store Expiry Date Of Medicine
7	QUANTITY	INT	5	NOT NULL	Use to Store Medicine Quantity
8	MRP	DOUBLE		NOT NULL	Use to Store Medicine MRP
9	DISCOUNT	DOUBLE		NOT NULL	Use to Store Medicine Discount
10	TOTAL	DOUBLE		NOT NULL	Use To Store Total Amount

8. Table Name : suppliers

This Table Is Use To Store Detail Of Supplier Of Medicine In System.

Sr.no	Field Name	Data Type	Size	Constraint	Description
1	ID	INT	3	Primary Key, Auto Increment	Use to Store Suppliers Id
2	NAME	VARCHAR	20	NOT NULL	Use to Store Suppliers Name
3	EMAIL	VARCHAR	30	NOT NULL, UNIQUE	Use To Store Supplier Email
4	CONTACT_NUMBER	VARCHAR	12	NOT NULL, UNIQUE	Use to Store Suppliers Contact Number
5	ADDRESS	VARCHAR	50	NOT NULL	Use to Store Suppliers Address

9. Table Name : user

This Table Is Use To Store Detail of All User In System .This Is very
Important Table Because Of This Table User Can Login On System.

Sr.no	Field Name	Data Type	Size	Constraint	Description
1	ID	INT	3	Primary Key, Auto Increment	Use to Store Suppliers Id
2	email	VARCHAR	30	NOT NULL, UNIQUE	Use to Store Email Of User
3	USERNAME	VARCHAR	15	NOT NULL, UNIQUE	Use To Store User name Of User
3	PASSWORD	VARCHAR	50	NOT NULL	Use to Store Password Of User
4	type	CHAR	1	NOT NULL	Use to Store Type Of User [A-Admin M-Manager C-Customer]

4.2 Data Flow Diagram :

4.2.1 What is Data Flow Diagram?

- A Data Flow Diagram(DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically.

- It shows how information enters and leaves the system, what changes the information and where information is stored.

- The purpose of DFD is to show the scope and boundaries of a system as a whole.

- It may be used as a communications tool between a system analyst and any person who plays a part in the system that acts as the starting point for redesigning a system

- It usually begins with a context diagram as level 0 of the DFD diagram, a simple representation of the whole system.

- To elaborate further from that, we drill down to a level 1 diagram with lower-level functions decomposed from the major functions of the system. This could continue to evolve to become a level 2 diagram when further analysis is required.

- Progression to levels 3, 4 and so on is possible but anything beyond level 3 is not very common.

4.2.2 Component of Data Flow Diagram

➢ The two main types of notation used for data flow diagrams are Yourdon-Coad and Gane-Sarson, both named after their creators, all experts who helped develop DFD methodology: Ed Yourdon, Peter Coad, Chris Gane and Trish Sarson.

➢ There are some differences in style between the notation types. For example, Yourdon and Coad notation uses circles to represent processes, whereas Gane and Sarson notation use rectangles with rounded corners.

➢ Because DFD symbols vary, it's important to be consistent with whatever notation you choose in order to avoid confusion.

➢ Here we will be using the Yourdon-Coad notations

➢ All data flow diagrams include four main elements: entity, process, data store and data flow.

- **Entity:**
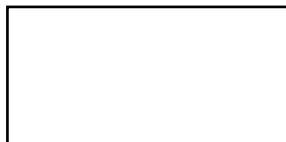
[Entity]

■ Also known as actors, sources or sinks, and terminators, external entities produce and consume data that flows between the entity and the system being diagrammed.

- Since they are external to the system being analysed, these entities are typically placed at the boundaries of the diagram.

- **Process :**
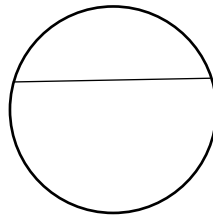
[Process]

- An activity that changes or transforms data flows. Since they transform incoming data to outgoing data, all process must have inputs and outputs on a DFD.

- Processes are typically oriented from top to bottom and left to right on a data flow diagram.

- **Data Source :**

[Data Source]

- A data store does not generate any operations but simply holds data for later access

- Data stores could consist of files held long term or a batch of documents stored briefly while they wait to be processed.

- Input flows to a data store include information or operations that change the stored data. Output flows would be data retrieved from the store.

- **Data Flow :**

⟶

[Data Flow]

- Movement of data between external entities, processes and data stores is represented with an arrow symbol, which indicates the direction of flow.

- Input and output data flows are labelled based on the type of data or its associated process or data store, and this name is written alongside the arrow.

4.2.3 Rules of Data Flow Diagram

- All Data Flows must flow to or from a process.
- A process must have at least one input flow and one output flow.
- The input to a process must be sufficient to produce output flows.
- Process must transform data.
- Data cannot flow between two entities.
- Data cannot flow directly from an entity to data store.

4.2.4 Advantage of Data Flow Diagram

Using a data flow diagram offers several benefits, especially during system analysis, design, and documentation stages. Here are some of the critical advantages of employing DFDs:

1.Visual Representation

DFDs provide a clear graphical representation of a system's processes, data flows, data stores, and external entities. This visual element helps technical and non-technical stakeholders grasp system components and their interrelationships more easily.

2.System Overview

The context diagram (level 0 DFD) offers a bird's-eye view of the entire system, facilitating a high-level understanding of system boundaries, major processes, and external interactions.

3.Modular Decomposition

DFDs allow for a top-down modular decomposition of a system. As one moves from higher-level DFDs to more detailed ones, one can delve deeper into specific system aspects without getting overwhelmed by the system's entirety.

4.Communication Tool

DFDs are an excellent communication tool between analysts, designers, developers, and other stakeholders. They ensure everyone consistently understands the system's structure and functionality.

5.Identification of Redundancies

DFDs can help identify redundant or unnecessary data processes by mapping out data flows, leading to streamlined system design.

6.Enhanced Error Detection

DFDs can aid in pinpointing inconsistencies, missing elements, or potential bottlenecks within the system, which can then be addressed during the design phase.

7. Documentation

DFDs contribute to system documentation, providing future developers, analysts, and managers with valuable insights into system operations and data flow.

8. Facilitates System Improvements

Over time, as the system needs to evolve or be upgraded, DFDs can assist in pinpointing areas of improvement, integration, or modification.

9. Boundary Clarification

DFDs help clarify a system's boundaries by distinguishing between external entities and internal processes. This distinction is crucial for defining the scope of system development projects.

10. Validation

DFDs can validate the proposed design with end-users or stakeholders, ensuring that the design aligns with the system's goals and requirements.

DFDs act as a roadmap for system development, offering clarity, facilitating communication, and ensuring the system is designed efficiently and effectively.

4.2.5 Data Flow Diagram

SYSTEM :Pharmacy Management System

❖ **Level-0 DFD :**

[Figure- 4.2.5.1 Level-0 DFD]
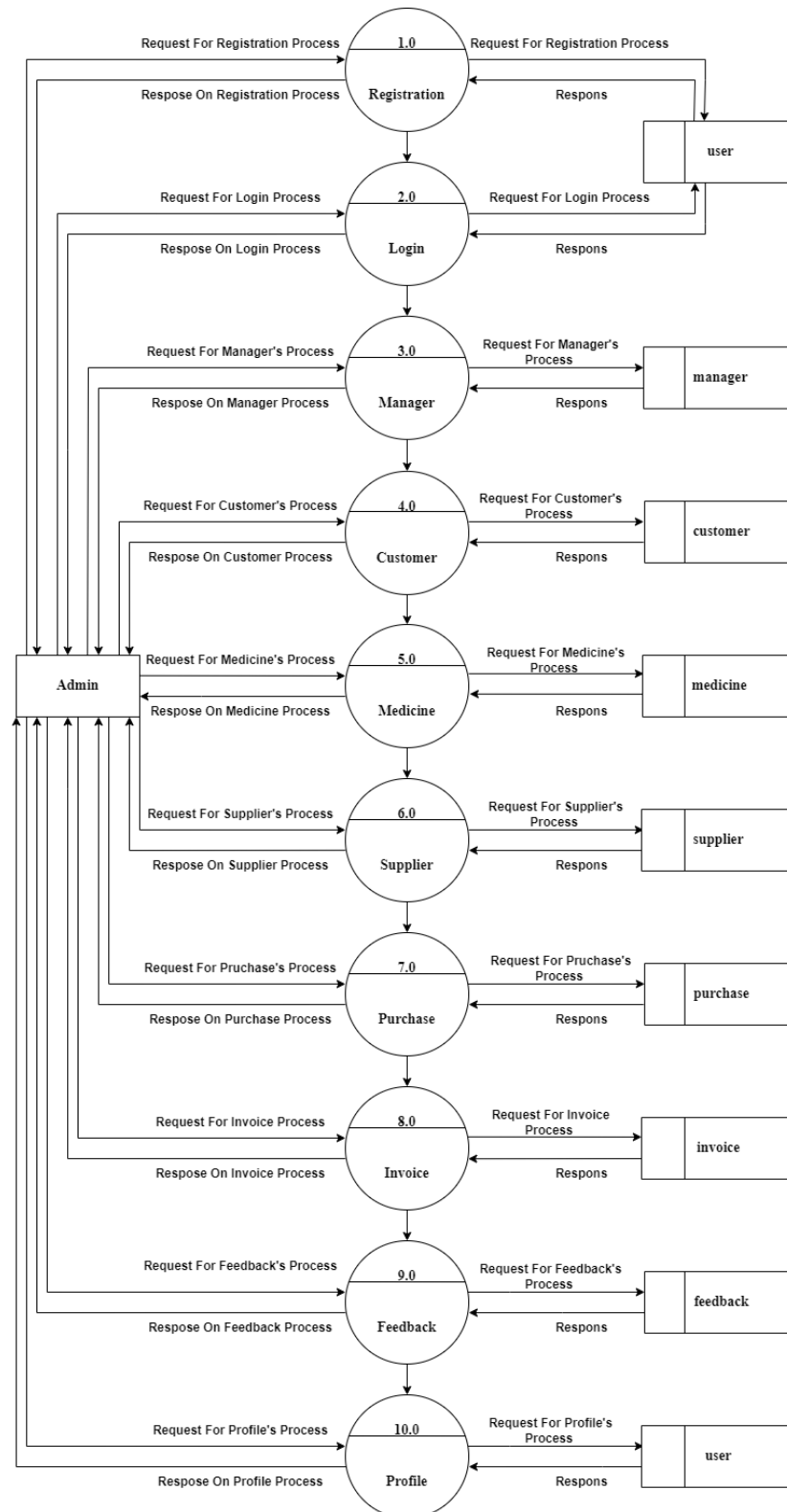
❖ **Admin Level-1 DFD:**

[Figure-4.2.5.2 Admin Level-1 DFD]
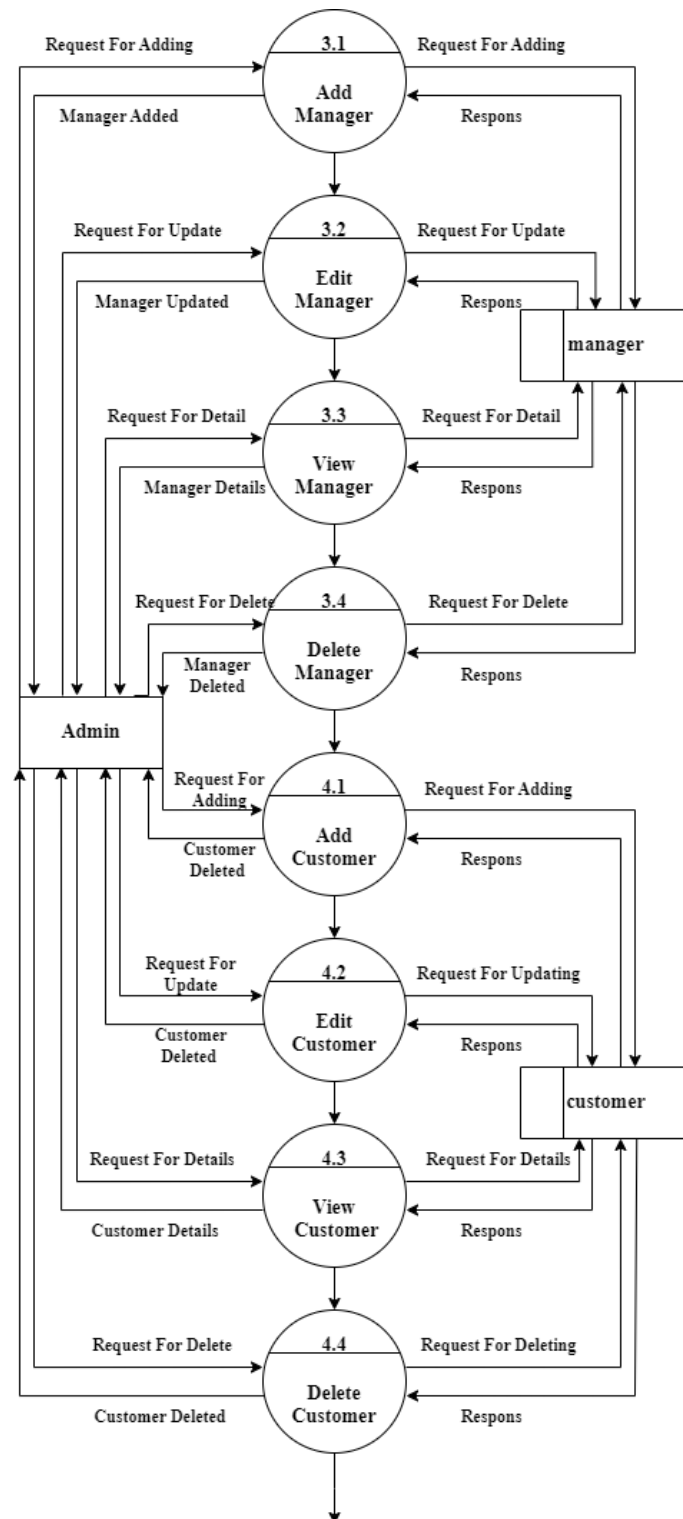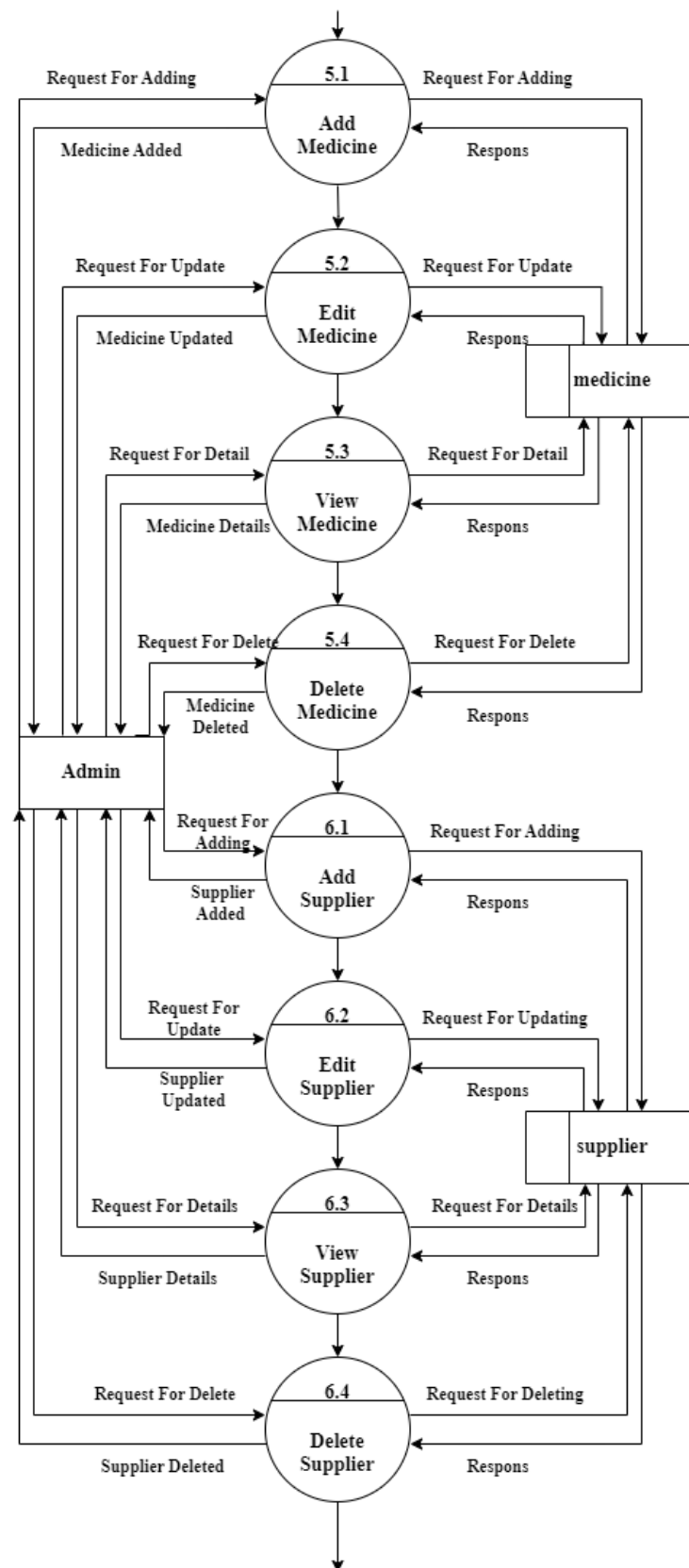
❖ **Admin Level-2 DFD :**

To Be Continue…

To Be Continue…
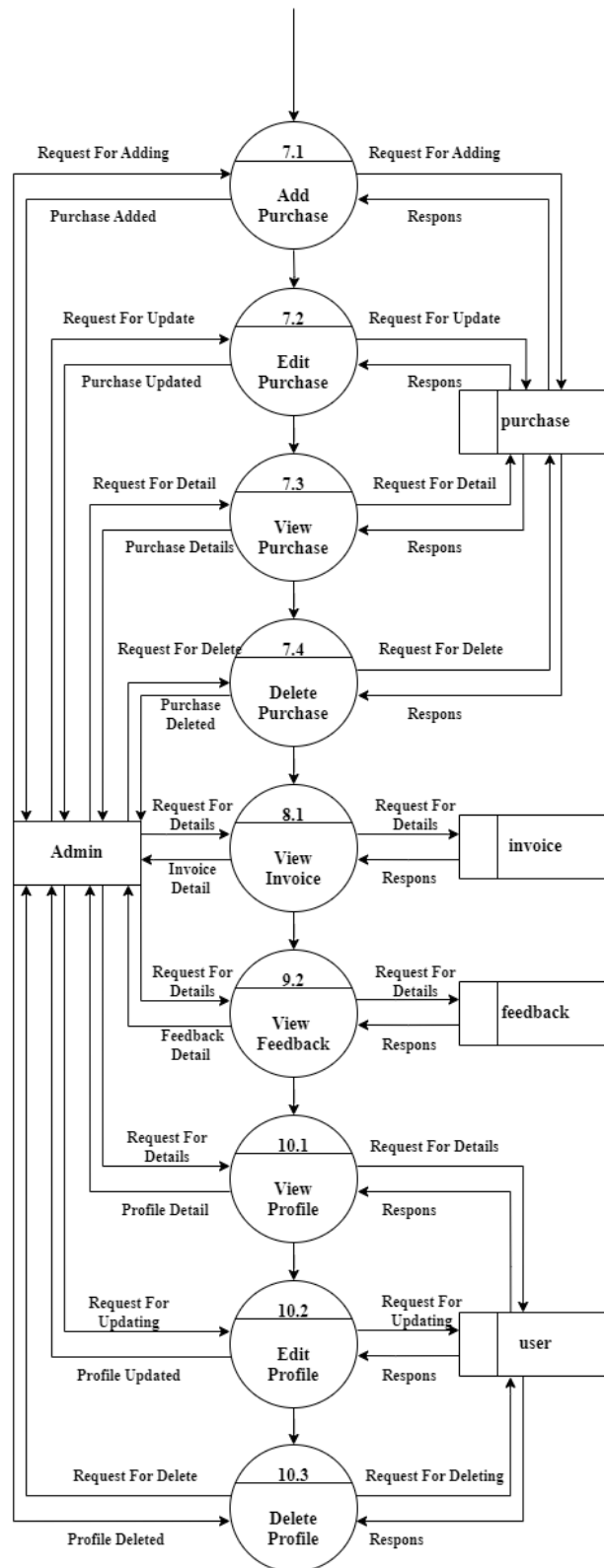
[Figure-4.2.5.3 Admin Level-2 DFD]

❖ **Customer Level-1 DFD:**

[Figure-4.2.5.4 Customer Level-1 DFD]

❖ **Customer Level-2 DFD:**
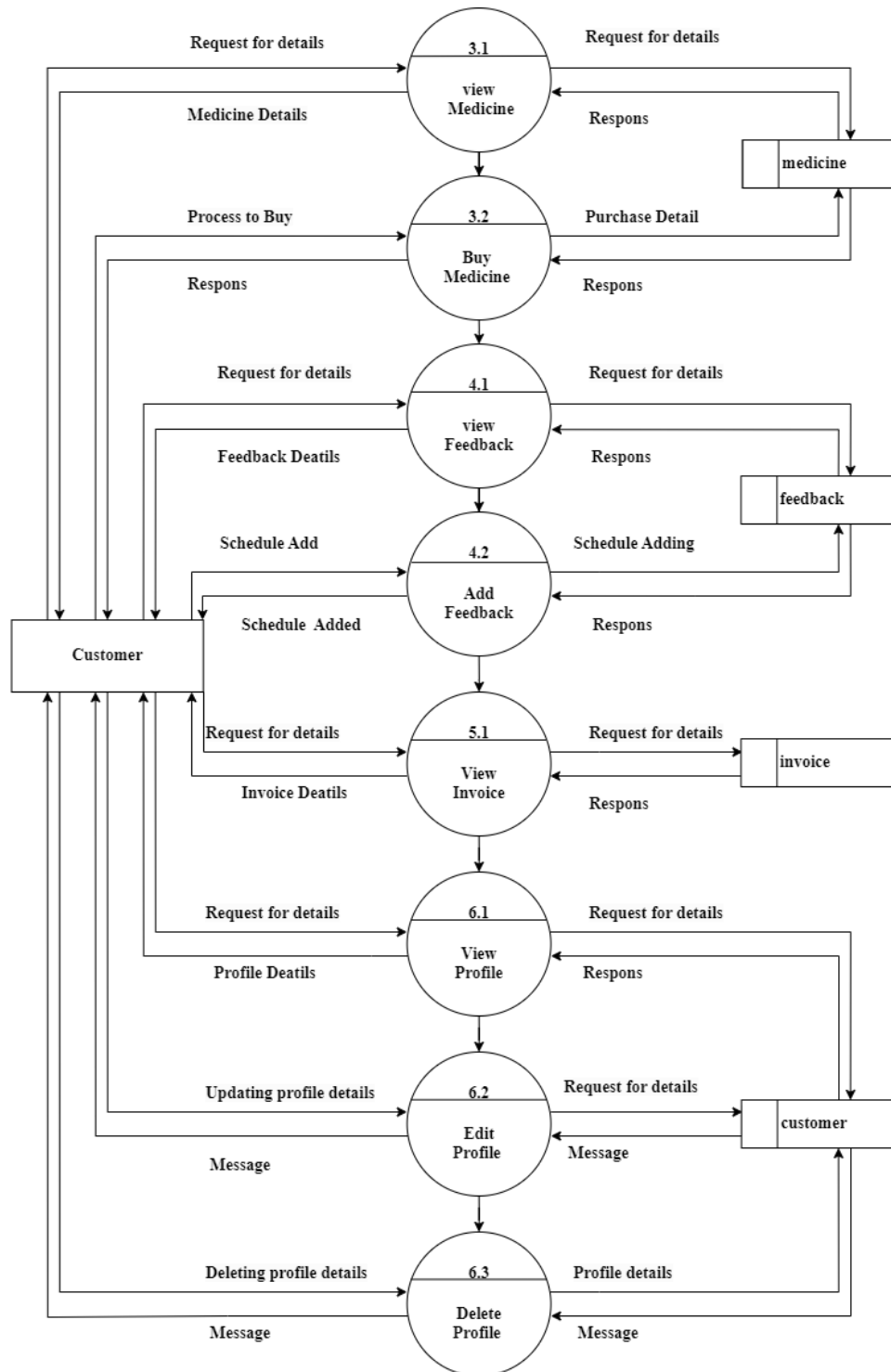
[Figure-4.2.5.5 Customer Level-2 DFD]
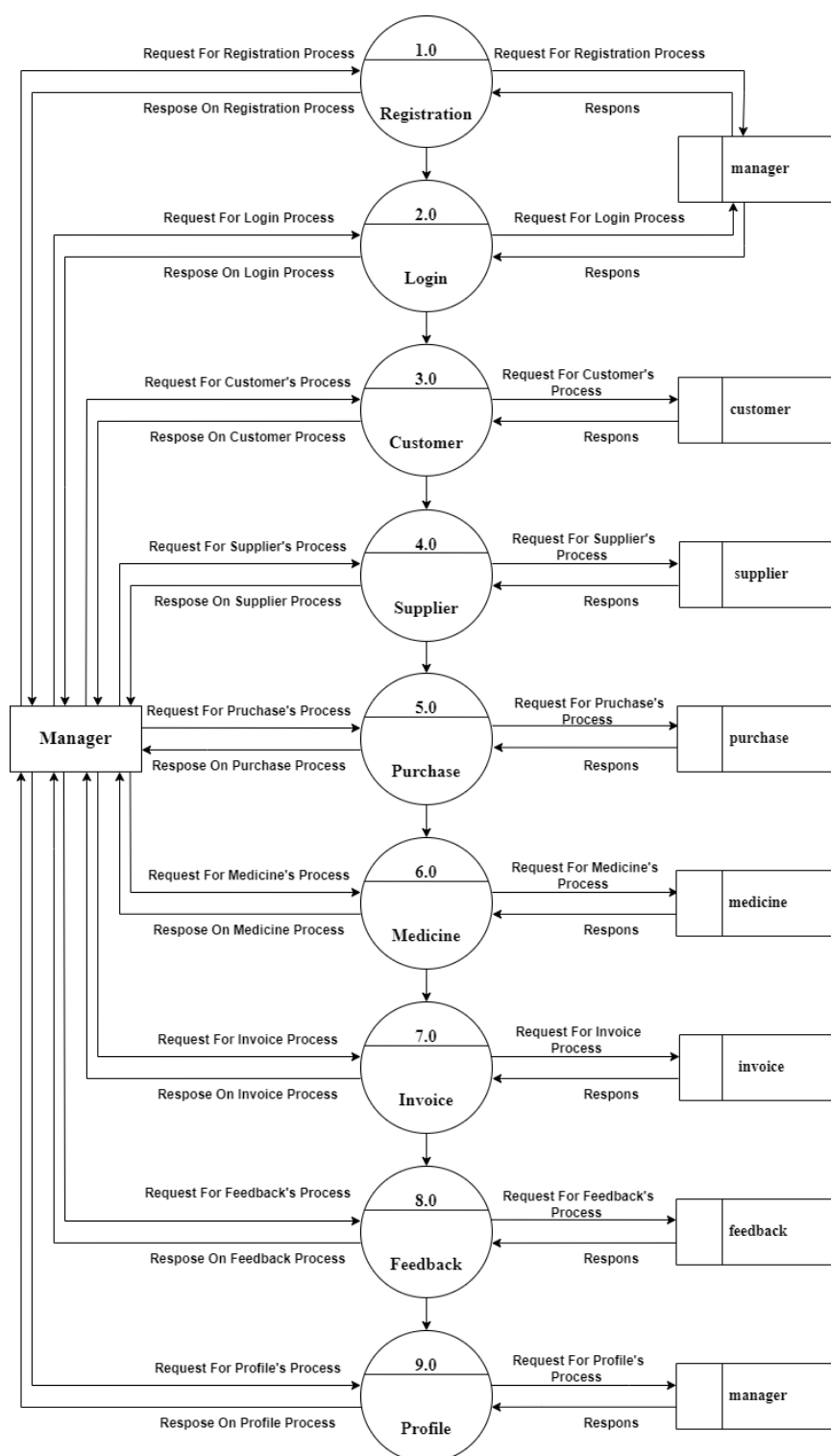
❖ **Manager Level – 1 DFD:**

[Figure-4.2.5.6 Manager Level – 1 DFD]
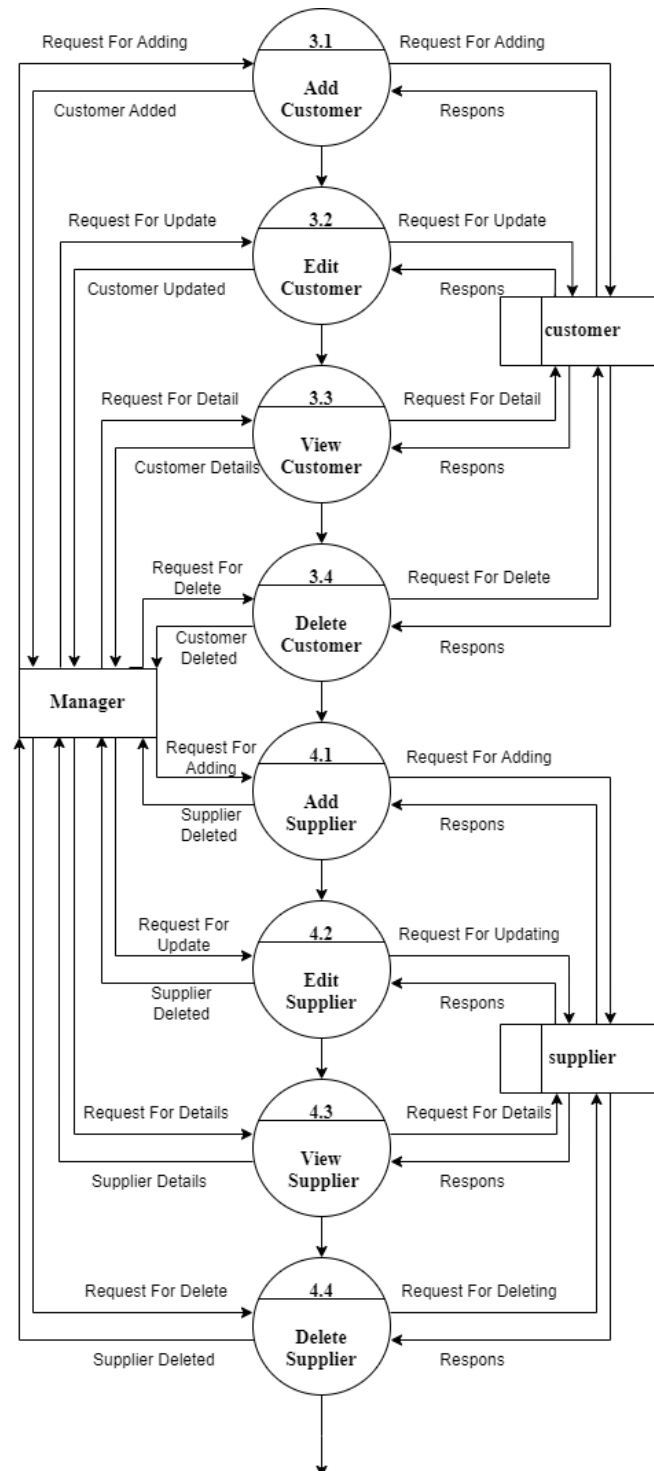
❖ **Manager Level – 2 DFD:**

To Be Continue…
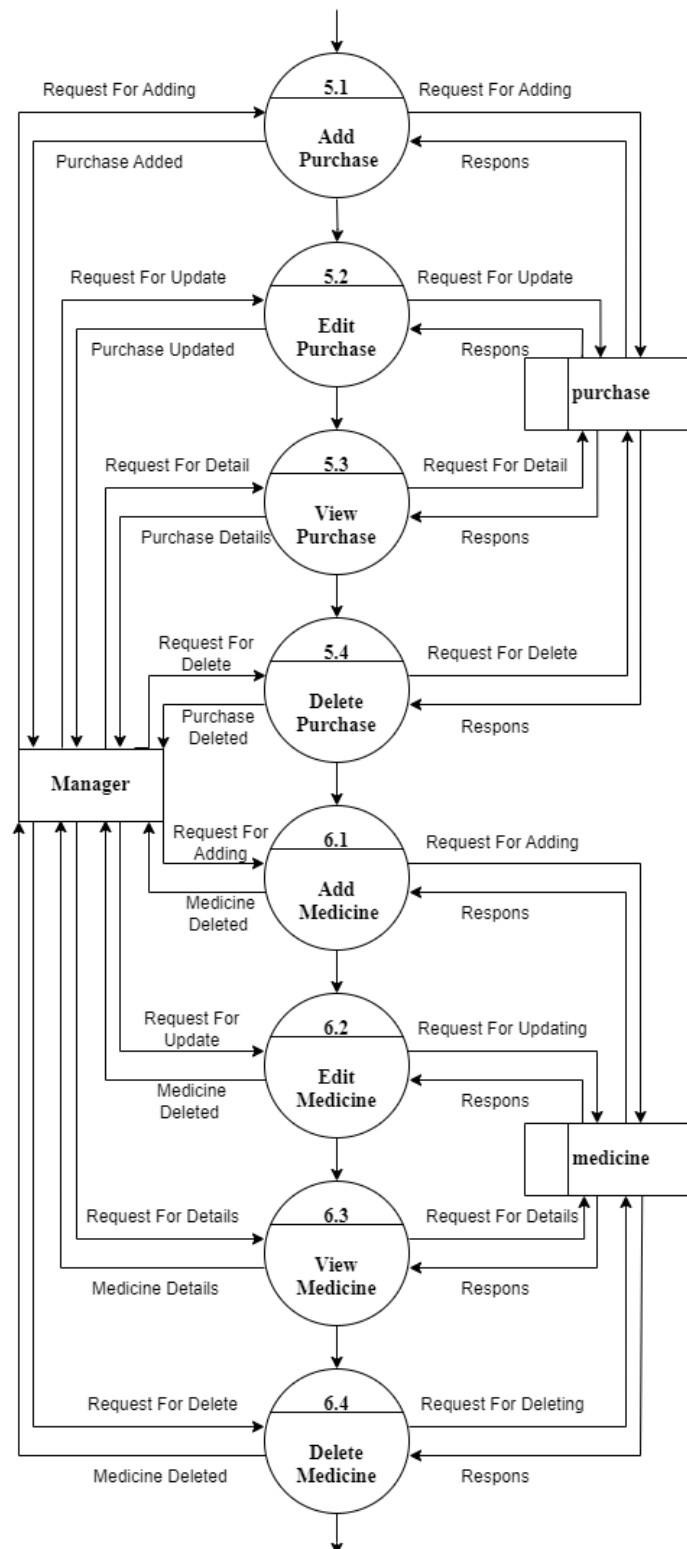
To Be Continue…
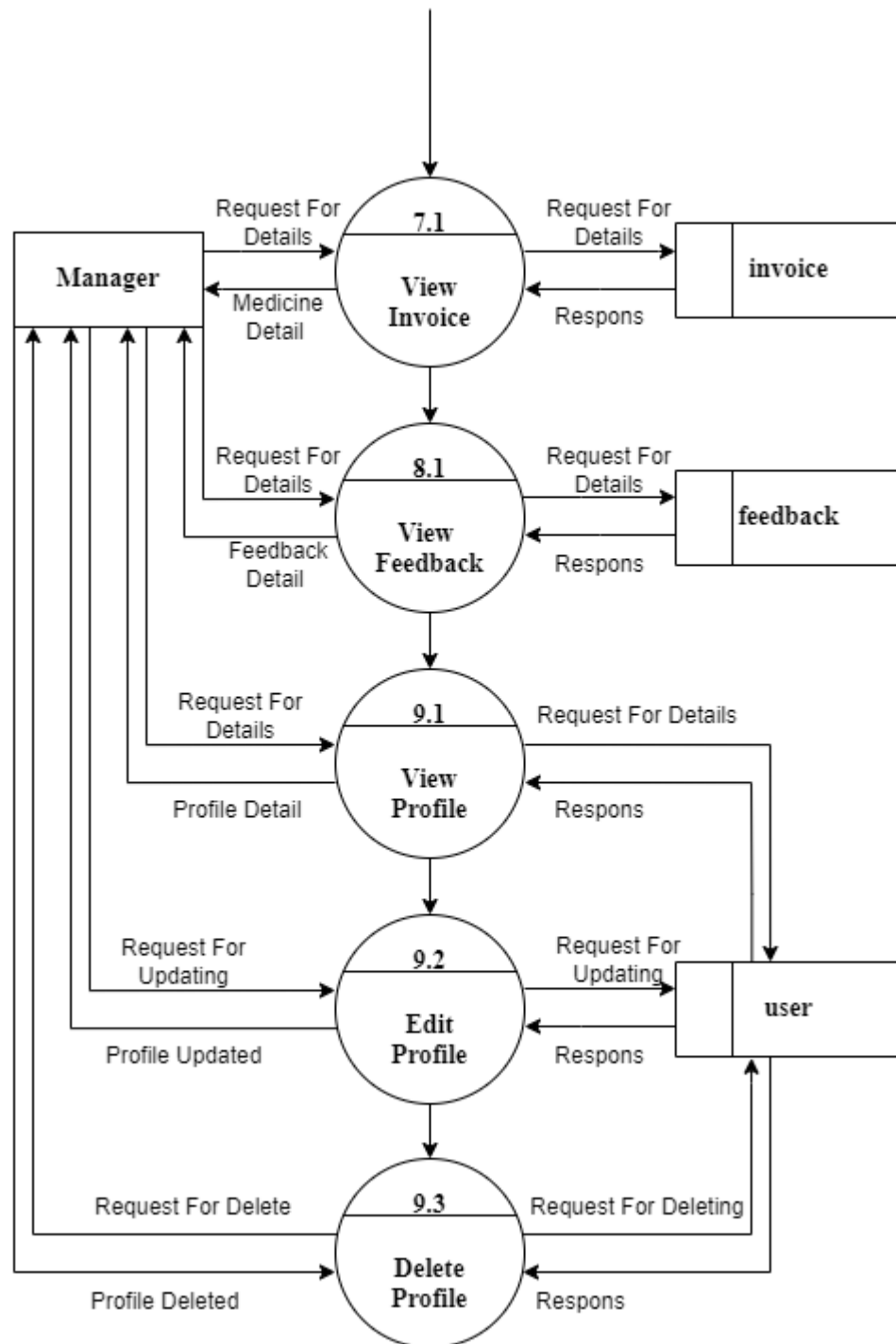
[Figure-4.2.5.7 Manager Level – 2 DFD]

4.3 E-R Diagram :

4.3.1 Introduction

An Entity Relation(ER) Diagram is a specialized graphics that illustrates the interrelationship between entities in a database. ER diagrams often use symbols to represent 3 different types of information. Boxes are commonly used to represent entities. Diamonds are normally used to represent relationships and ovals are used to represent attributes.

An Entity Relationship Model (ERM), in software engineering is an abstract and conceptual representation of data. Entity Relationship modelling is a relational schema database modelling method, used to produce a type of conceptual schema or semantic data model of a system, often a relation database, and its requirements in a top-down fashion.

4.3.2 Why use ER diagram?

Here, are prime reasons for using the ER Diagram.

- Helps you to define terms related to entity relationship modeling. It Provide a preview of how all your tables should connect, what fields are going to be on each table.

- Helps to describe entities, attributes, relationships. It ER diagrams are translatable into relational tables which allows you to build databases quickly.

- ER diagrams can be used by database designers as a blueprint for implementing data in specific software applications.

- The database designer gains a better understanding of the information to be contained in the database with the help of ERP diagram.

- ERD Diagram allows you to communicate with the logical structure of the database to users.

4.3.3 Component of ER diagram
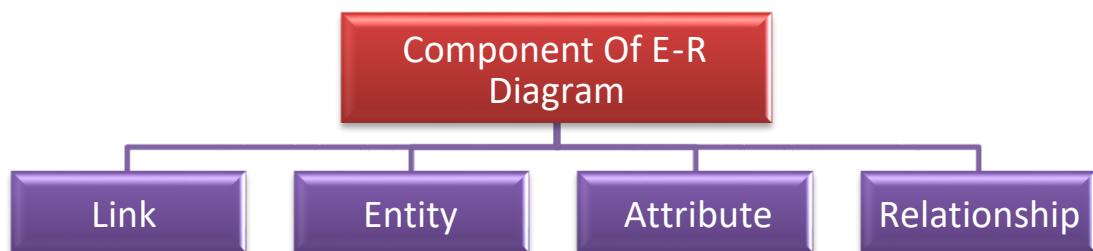
[Figure-4.3.3 Component of ER Diagram]

1. **ENTITY:**

Entity is the thing which we want to store information. It is an elementary basic building block of storing information about business process. An entity represents an object defined within the information system about which you want to store information. Entities are distinct things in the enterprise.

2. RELATIONSHIP:
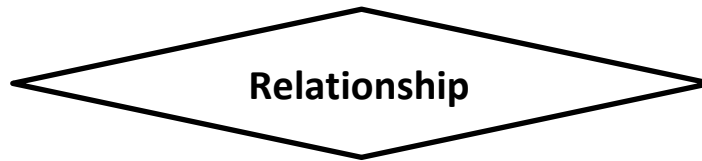
A relationship describes how entities interact. For example, the entity "carpenter" may be related to the "table" entity by the relationship "builds" or "makes". Relationship are represented by diamond shapes and are labelled using verbs.

There are four type of relationship

Types of relationship	Symbol
One to one relationship	**1 1**
One to many relationship	**1 N**
Many to one relationship	**N 1**
Many to many relationship	**N N**

3. ATTRIBUTE:

Attribute are the properties of the entities and relationship, descriptor of the entity.Attribute are elementary pieces of information attached to an entity.

4. LINK:

_____

Link is connected entity to attribute or attribute to entity.

4.3.4 ER Diagram
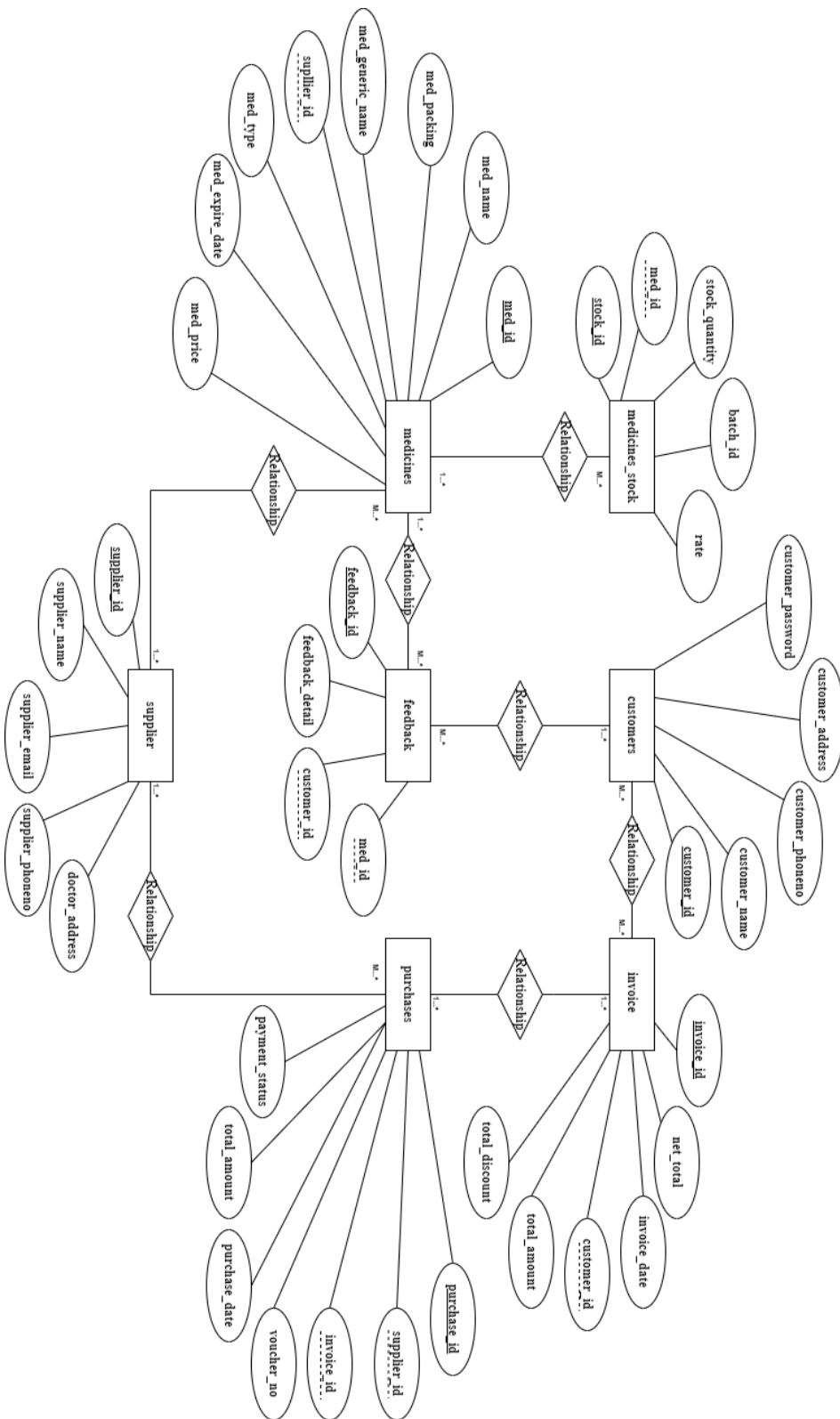
[Figure-4.3.4 ER Diagram]

4.4 USE CASE :

4.4.1 What Is Use Case Diagram:

> ➢ A UML use case diagram is the primary form of system/software requirements for a new software program underdeveloped.

> ➢ Use cases specify the expected behaviour (what), and not the exact method of making it happen.

> ➢ Use cases once specified can be denoted both textual and visual representation (i.e. use case diagram).

> ➢ A key concept of use case modeling is that it helps us design a system from the end user's perspective.

> ➢ It is an effective technique for communicating system behaviour in the user's terms by specifying all externally visible system behaviour.

4.4.2 Origin Of Use Case Diagram:

These day use case modelling is often associated with UML, although it has been introduced before UML existed. Its brief history is as follow:-

In 1986, Ivar Jacobson first formulated **textual** and **visual modelling** techniques for specifying use cases.

In 1992 his co-authored book Object-Oriented Software Engineering - A Use Case Driven Approach helped to popularize the technique for capturing functional requirements, especially in software development.

4.4.3 Symbols / Notations Of Use Case Diagram :

i. ACTOR :

```
┌─────────────────────┐
│          ○          │
│         /│\         │
│          │          │
│         / \         │
│                     │
│       [Actor]       │
└─────────────────────┘
```
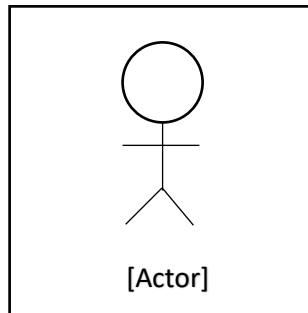
➢ The actor is an entity that interacts with the system. An actor is an entity that initiates the use case from outside the scope of a use case.

➢ It can be any element that can trigger an interaction with the use case. One actor can be associated with multiple use cases in the system. A user is the best example of an actor.

ii. USE CASE:

```
┌─────────────────────┐
│                     │
│     ⬭ Use Case      │
│                     │
│      [Use Case]     │
└─────────────────────┘
```
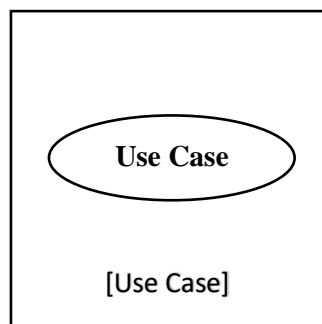
➢ A use case describes a sequence of actions that provide something to an actor.

➢ Use cases are used to represent high-level functionalities and how the user will handle the system

iii.   COMMUNICATION LINK:

[Communication-Link]
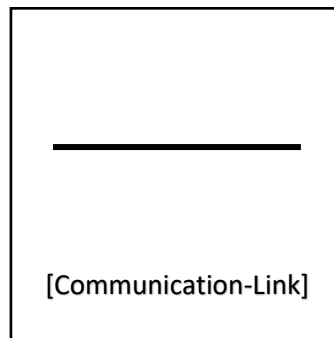
➢ The participation of an actor in a use case is shown by
connecting an actor to a use case by a solid link.

➢ Actors may be connected to use cases by associations,
indicating that the actor and the use case communicate with one
another using messages.

iv.   BOUNDARY OF SYSTEM:
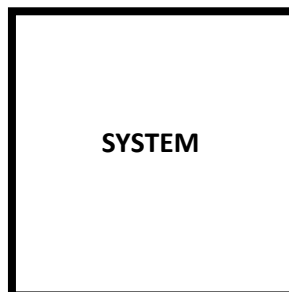
SYSTEM

➢ The system boundary is the entire system as defined in the
requirements document. It indicates the scope of your system.

➢ Anything within the box represents functionality that is in
scope and anything outside the box is not. System boundary
boxes are rarely used.

❖ **Use-case Diagram [Admin]:**
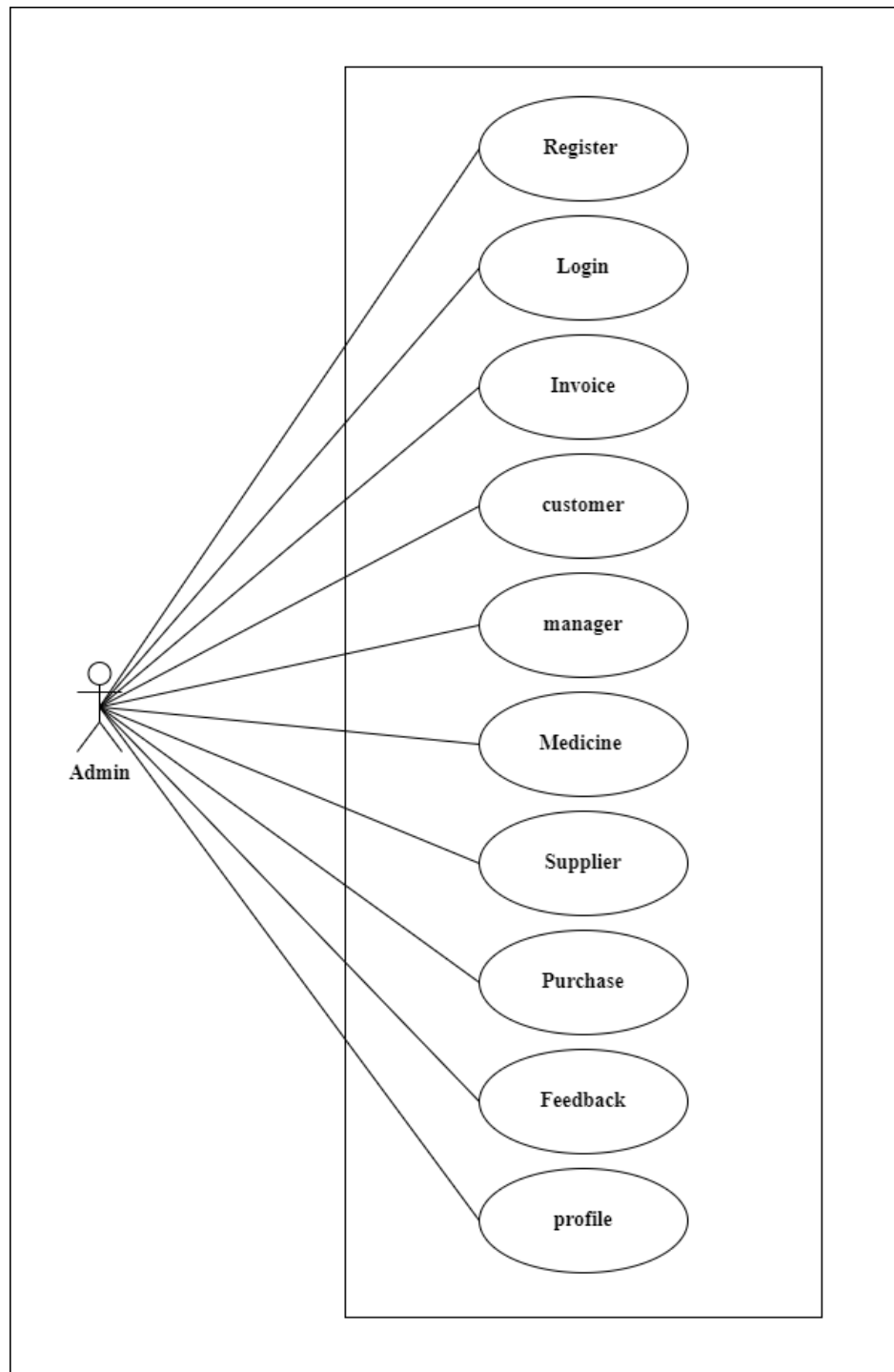
[Figure-4.4.1 Use-case Diagram-Admin]

❖ **Use-case Diagram [Customer]:**
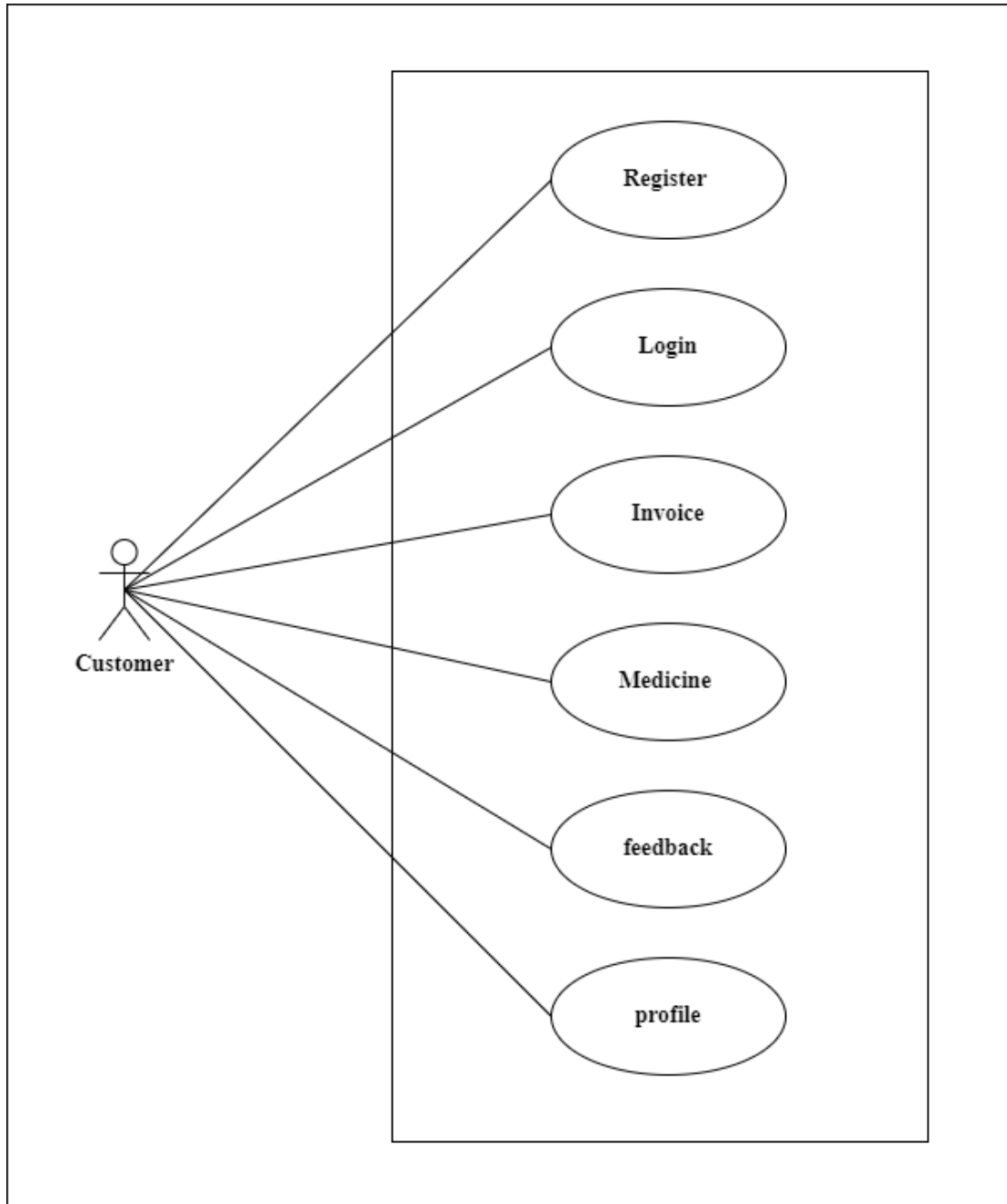
[Figure-4.4.2 Use-case Diagram -Customer]

❖ **Use-case Diagram [Manager]:**
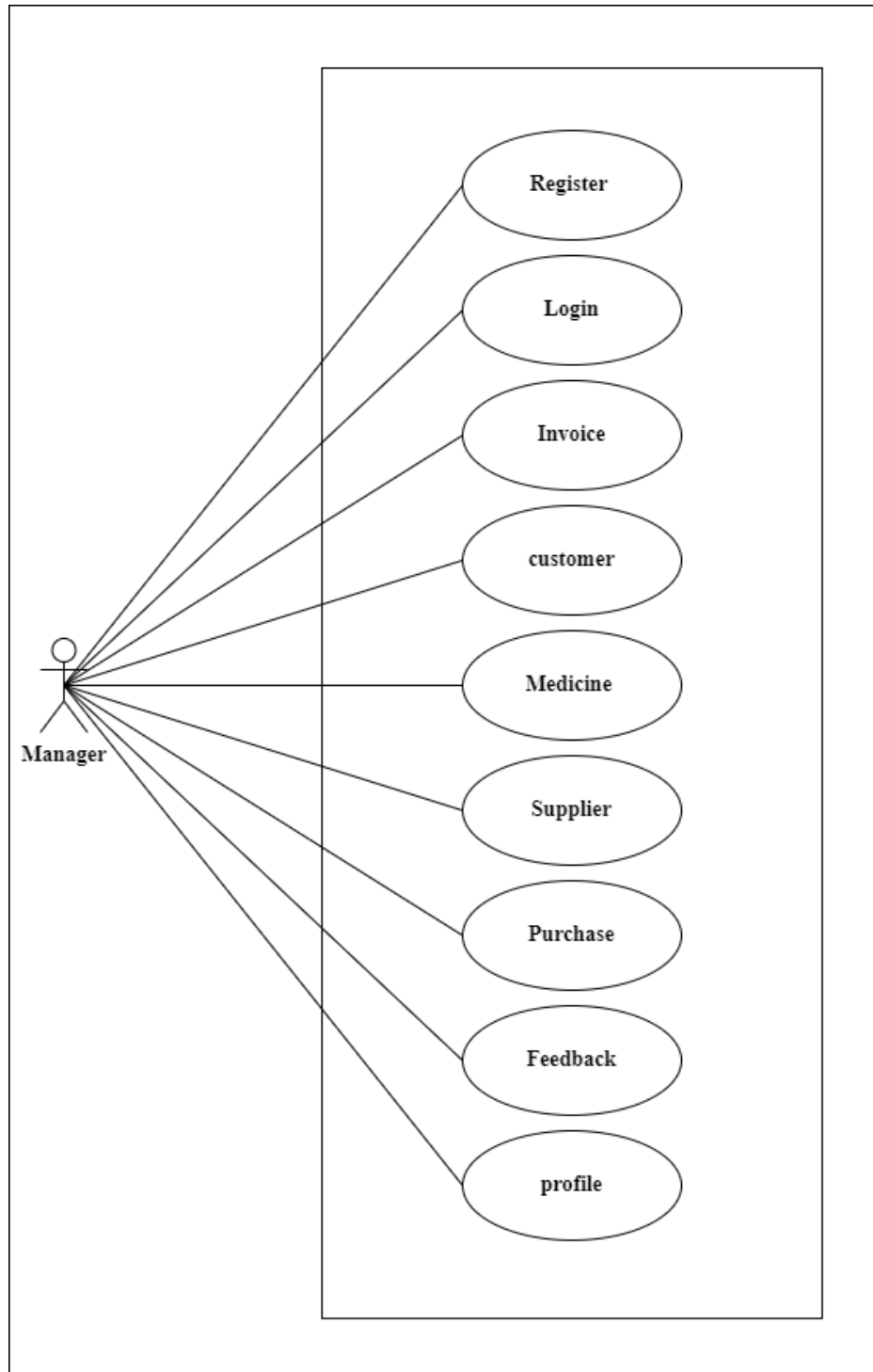
[Figure-4.4.3 Use-case Diagram-Manager]

4.5 Input /Output Design:

4.5.1 Login Page :
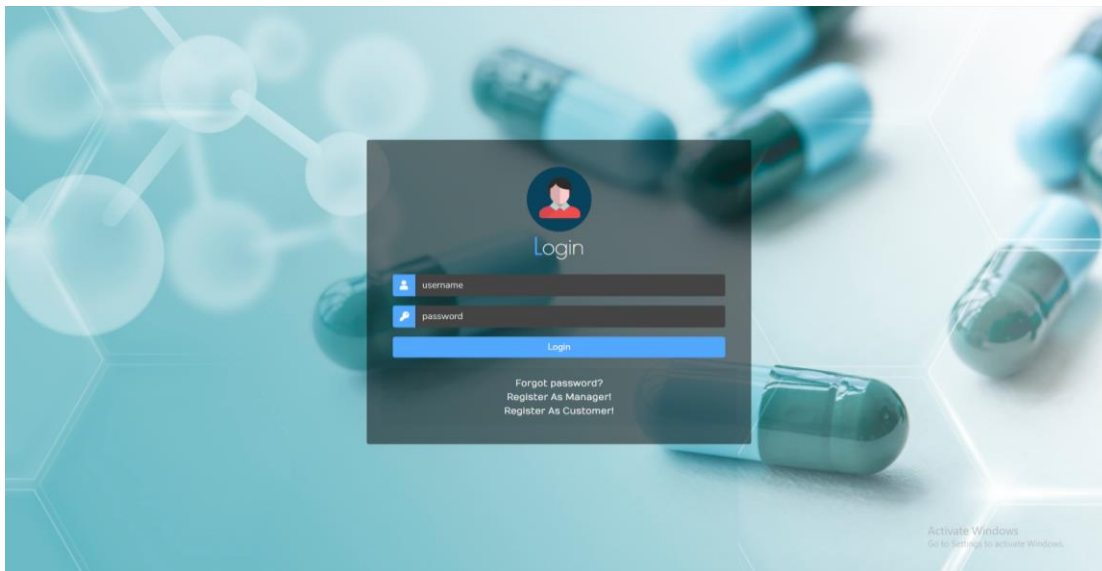
This figure is shows the login page with That Use Login On System With Their Unique User Name And Password.

4.5.2 Customer Registration :
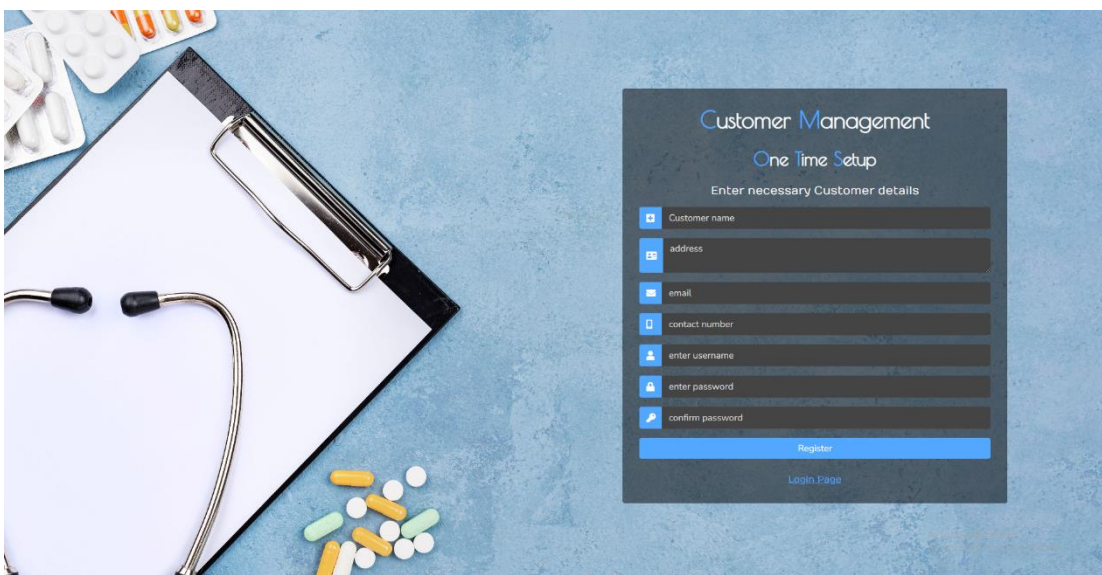
This figure is shows the Customer Registration Page With That Customer Register Him/Her Self To Login On System .

4.5.3 Manager Registration :
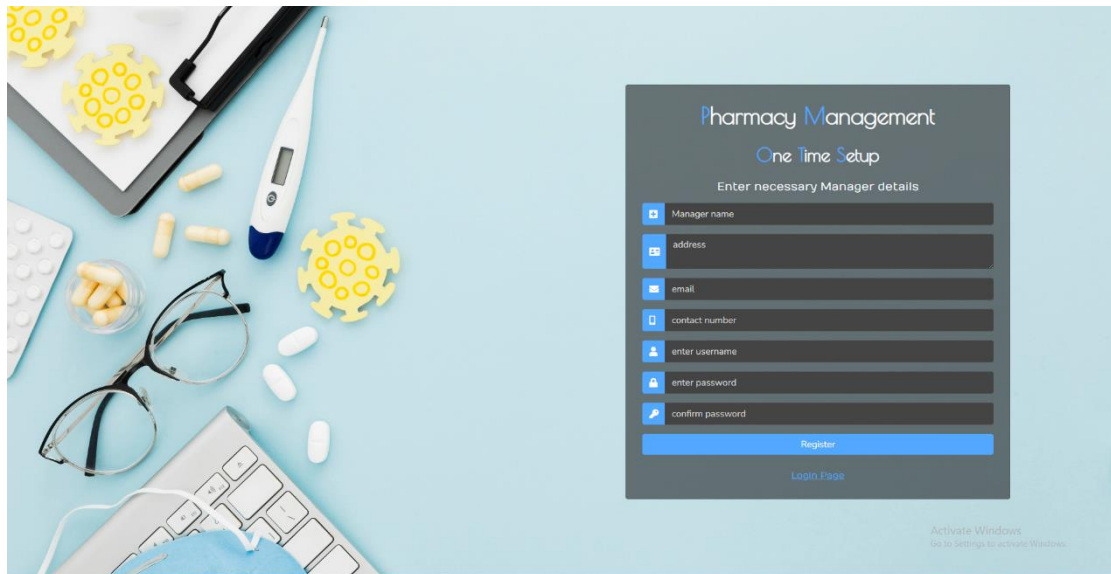
This figure is shows the Manager Registration Page With That Manager Register Him/Her Self To Login On System.

4.5.4 Forget Password :
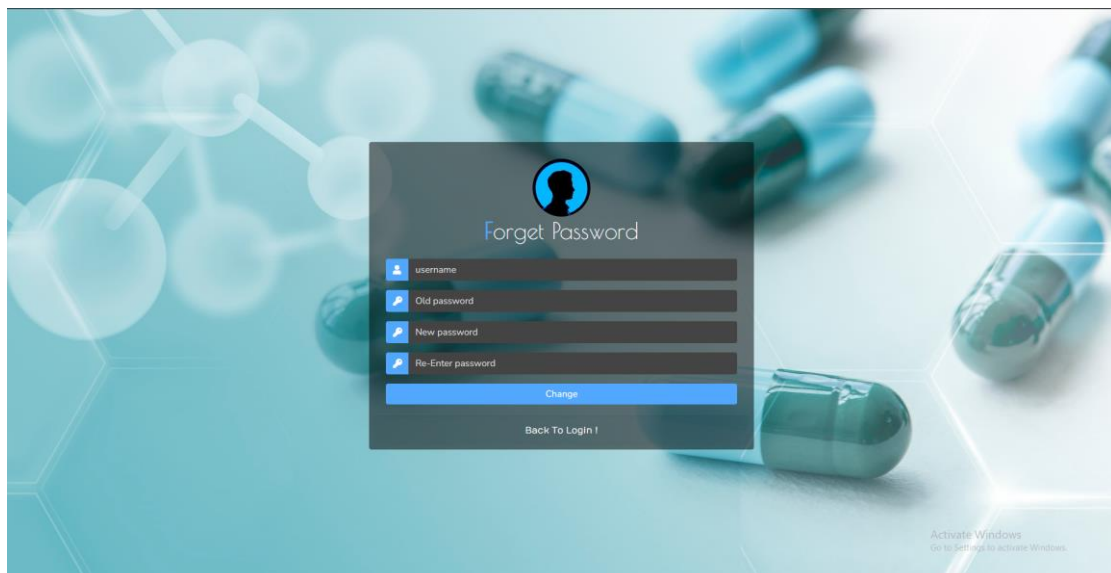
This figure is shows the Fortage Password Page With That Any user Of System Change Their Password without Login.

4.5.5 Admin Dashboard :
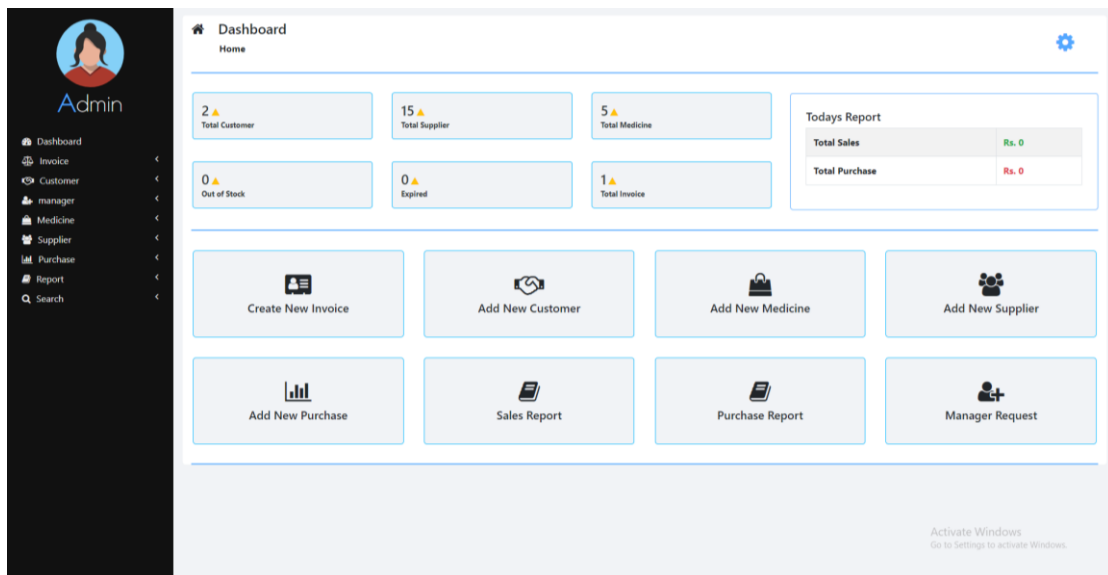
This figure is shows the Admin Dashboard page.

4.5.6 Customer Dashboard :
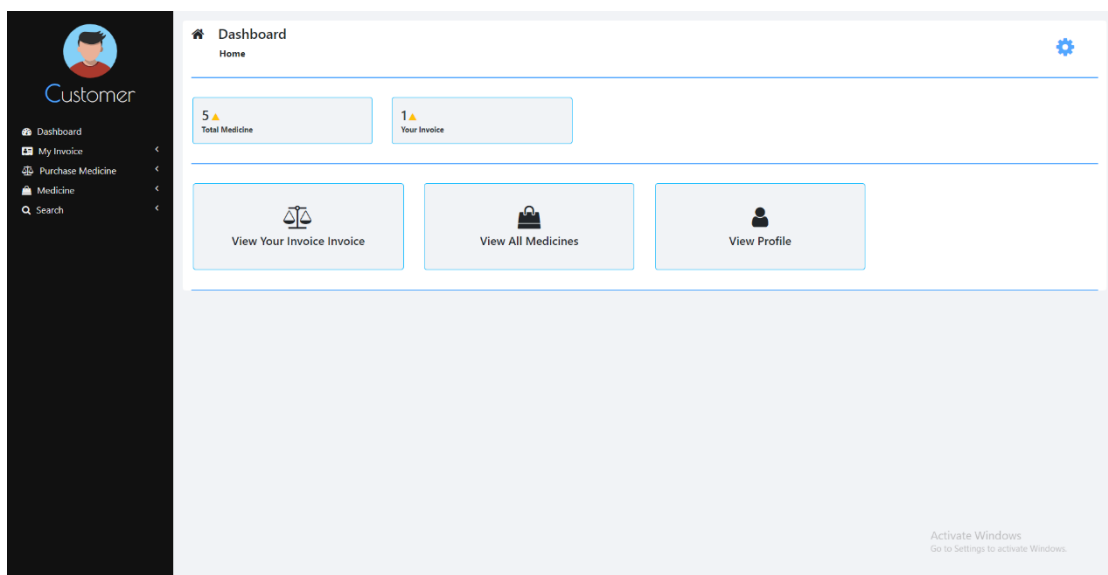
This figure is shows the Customer Dashboard page.

4.5.7 Manager Dashboard :
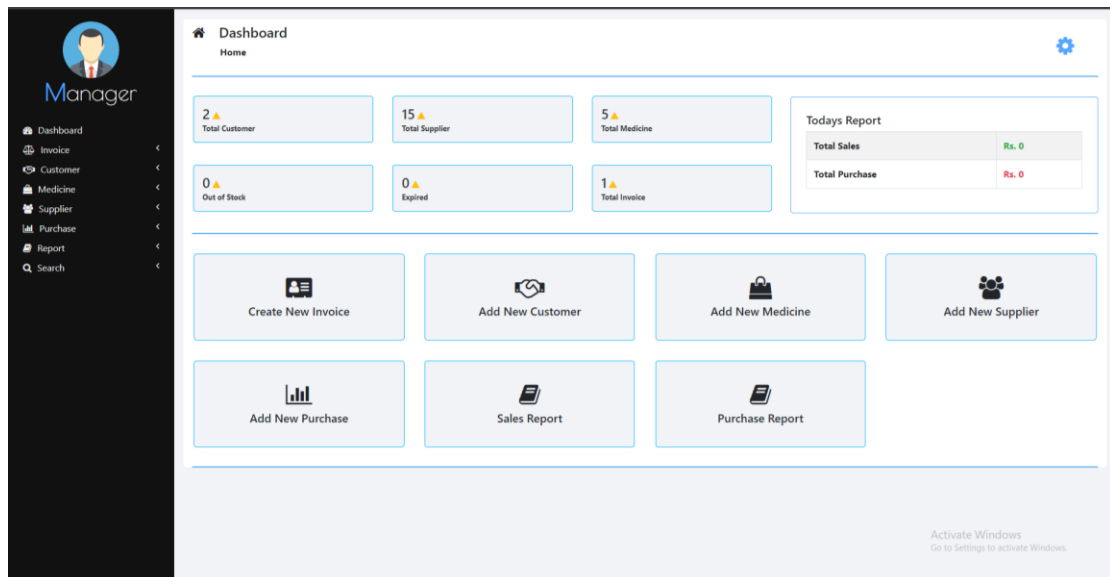
This figure is shows the Manager Dashboard page.

4.5.8 Buy Medicine :
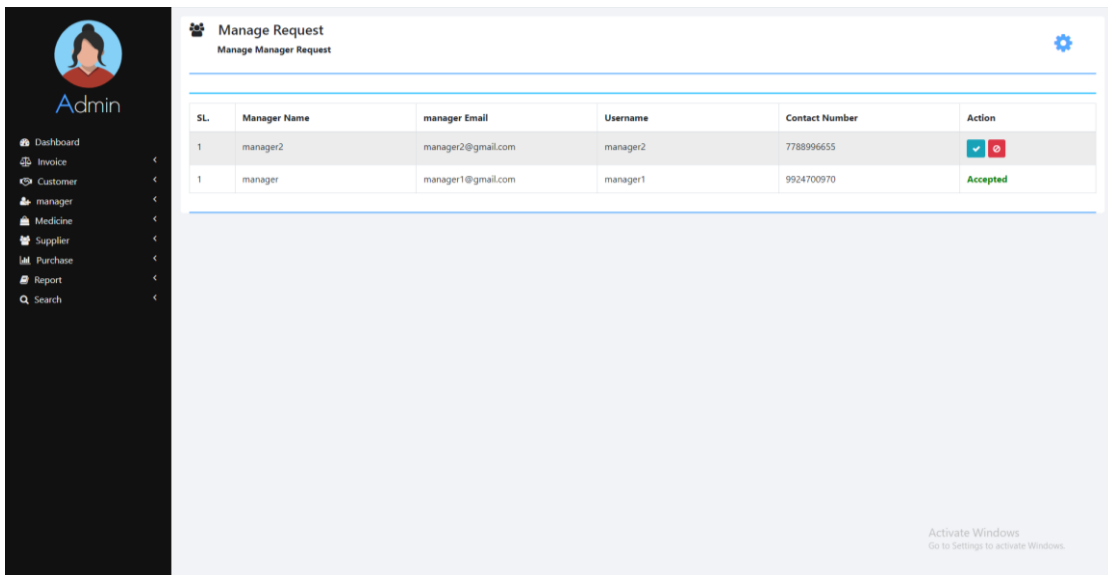
This figure is shows the Customer Side That If Any Customer
Want To Purchase Medicine
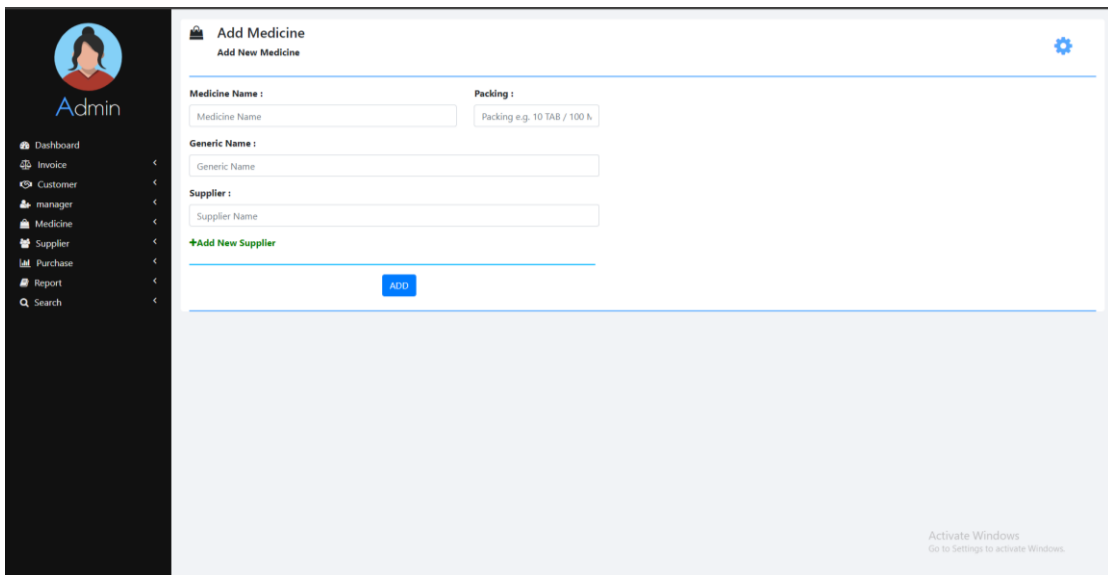
4.5.9 Manage Request :

This figure is shows the request Of Manager On System.This Only Visible TO Admin And Admin Has Right To Accept Or Denied.

4.5.10 Add Medicine :

This figure is shows the Add New Medicine On System By Admin And Also Manager Have Right To add New Medicine
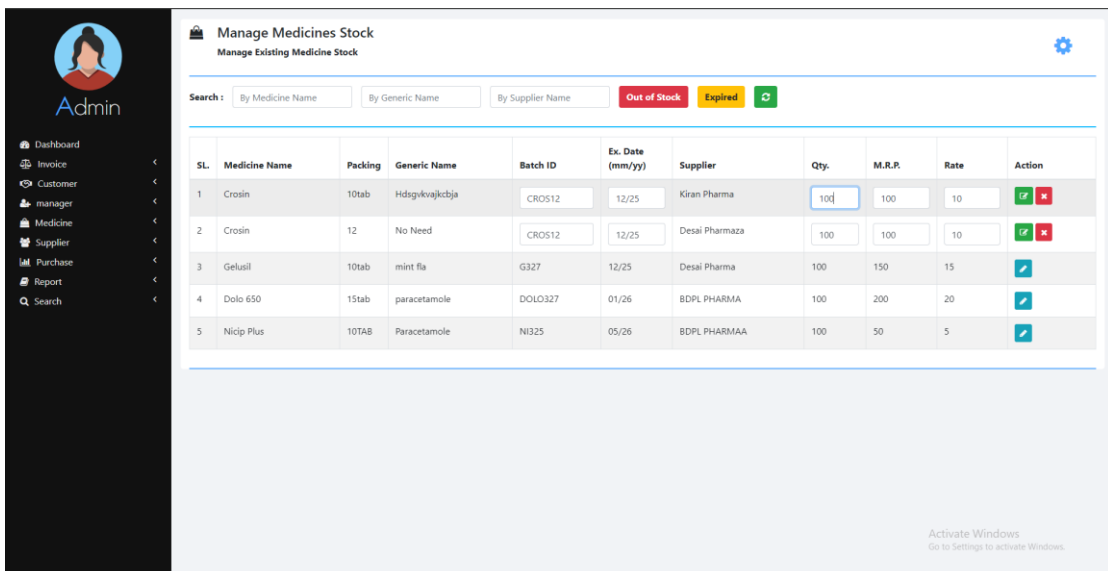
4.5.11 Manage Medicine :

This figure is shows the Manage Medicine Detail By Admin And Also Manager Have right To Do This.

4.5.12 Manage Medicine Stock :

This figure is shows the Manager Medicine Stock By Admin And Manager
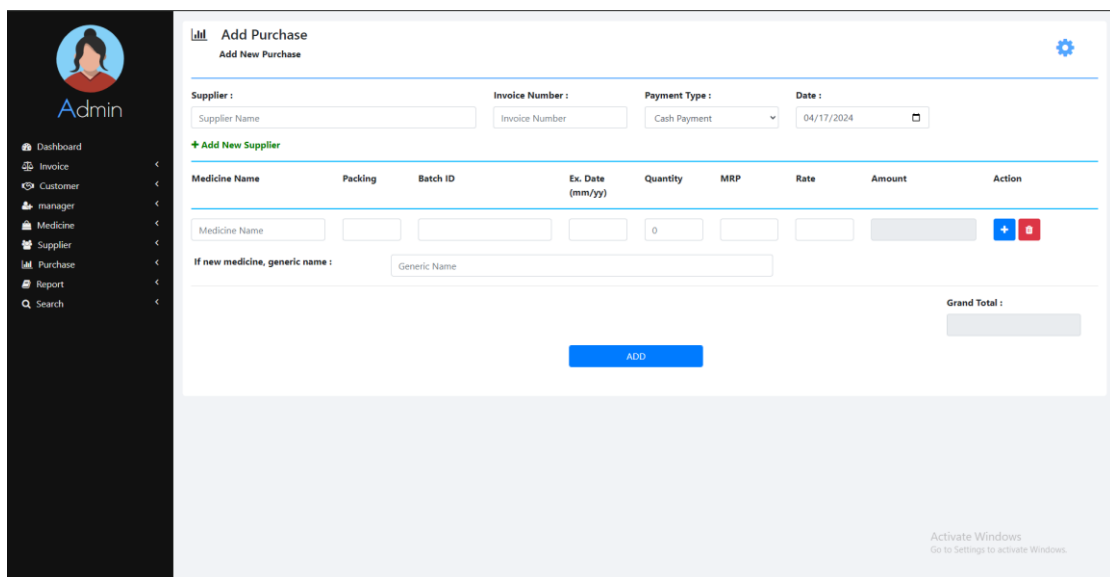
4.5.13 Add Supplier :

This figure is shows the Add New Supplier On System By Admin Or Manager

4.5.14 Add Medicine Purchase :
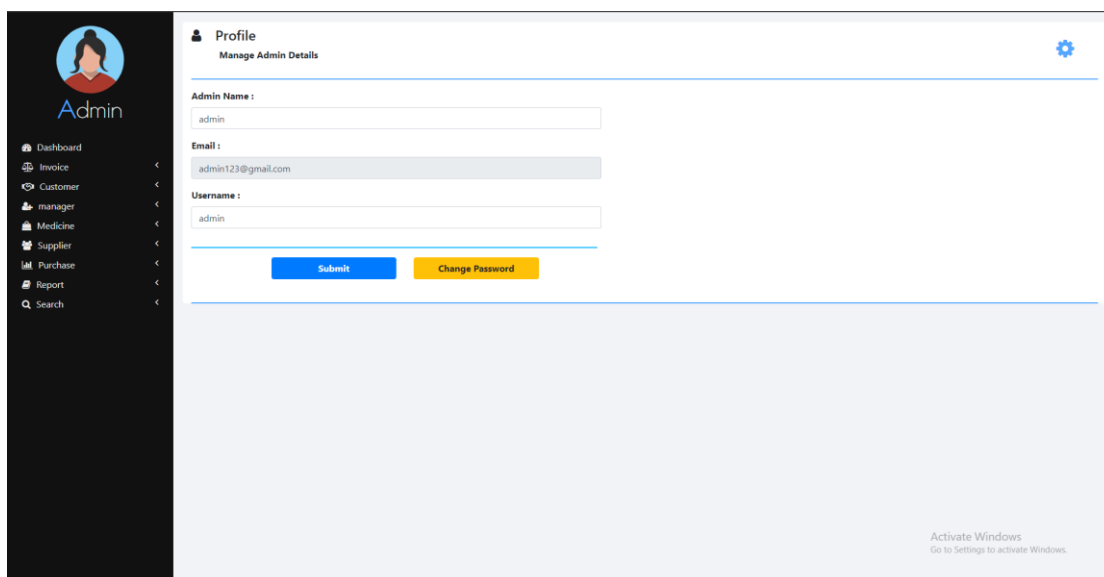
This figure is shows the Purchase Of Medicine And This Purchase Impact On Medicine Stock Also.
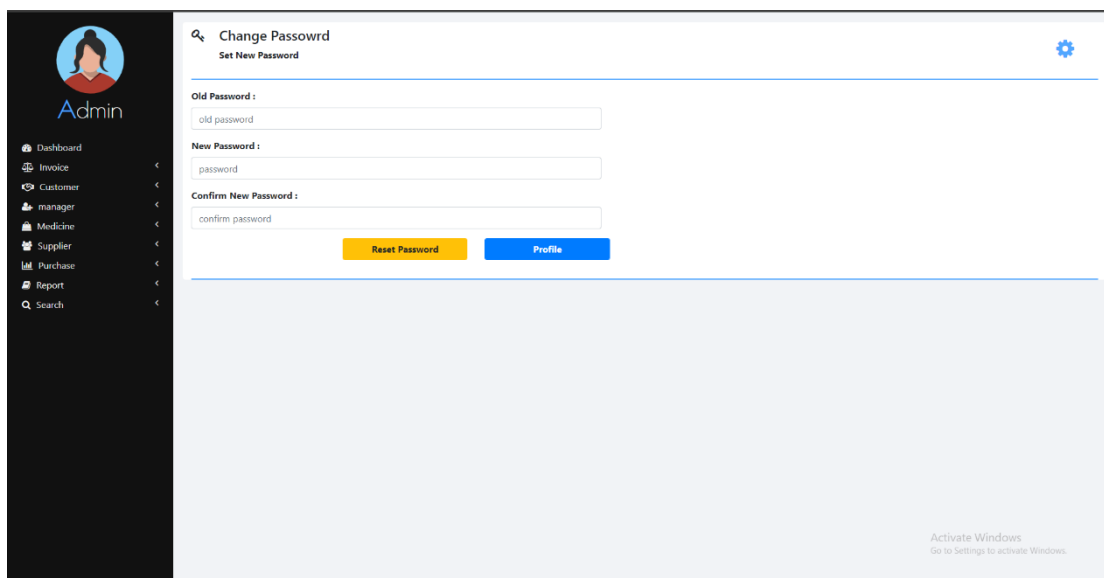
4.5.15 View Profile :

This figure is shows the Profile Page And This Page Use To Eidt Information According User On system There is Minor Change on Profile Page.

4.5.16 Change Password :

This figure is shows the Change Password After Login On System By User This Page Is Same for All User.

TESTING & IMPLEMENTATION

5.1 Introduction of Testing:

➢ Testing is a method to check whether the actual software product matches expected requirements and to ensure that software product is Defect free.

➢ It involves execution of software/system components using manual or automated tools to evaluate one or more properties of interest.

➢ The purpose of software testing is to identify errors, gaps or missing equirements in contrast to actual requirements.

Following are the characteristic that process the testing templates:

➢ The developer should conduct the successful technical reviews to perform the testing successful.

➢ Testing starts with the component level and work from outside toward the integration of the whole computer-based system.

➢ Different testing techniques are suitable at different point in time.

➢ Testing is organized by the developer of the software and by an independent test group.

➢ Debugging and testing are different activities, then also the debugging should be accommodated in any strategy of testing.

5.2 Importance of Testing:

➢ Testing is Important because if there are any bugs or errors in the software, it can be identified early and can be solved before delivery of the software product.

➢ Properly tested software product ensures reliability, security and high performance which further results in time saving, cost effectiveness and customer satisfaction.

➢ Testing is important because software bugs could be expensive or even dangerous. Software bugs can potentially cause monetary and human loss.

5.3 Benefits of Testing:

➢ **Cost-Effective:** It is one of the important advantages of software testing. Testing any IT project on time helps you to save your money for the long term. In case if the bugs caught in the earlier stage of software testing, it costs less to fix.

➢ **Security:** It is the most vulnerable and sensitive benefit of software testing. People are looking for trusted products. It helps in removing risks and problems earlier.

➢ **Product quality:** It is an essential requirement of any software product. Testing ensures a quality product is delivered to customers.

➢ **Customer Satisfaction:** The main aim of any product is to give satisfaction to their customers. UI/UX Testing ensures the best user experience

5.4 Testing Plan:

The main Objective of doing testing is to identify all defects existing in software. Basically, the testing of software consists of providing the program with a set of test inputs (test case) and observing that whether the software behaves as expected.

Testing is the process of executing a program with the explicit intention of finding errors, which makes the program fail. The tester is actually trying to make the program fail. A successful test is the one that finds errors.

Regardless of which strategies the analysts follow, there are preferred practices to ensure that the testing is useful. The levels of tests and practices to ensure that testing is useful. The levels of tests and types of test data, combined with testing libraries are important aspects of test process. Among the various testing practices or strategies that are followed by analysts, the two important ones are unit testing and system testing.

Software testing is crucial step in determining whether a software application is viable, ready for market and free of bugs. No software will be completely free of glitches but through software testing can and will make sure that it is as error free as humanly possible.

5.5 Testing Strategies:

[Figure-5.5 Testing Strategies]

- **UNIT TESTING :**

  ➢ Tests a single class or a set of closely coupled classes.
  ➢ These unit tests can either be run using the actual objects that the unit interacts with or by employing the use of test doubles or mocks.
  ➢ In Unit testing, the smallest piece of testable software is tested in the application to determine whether it behaves as expected or not.
  ➢ Tests are typically run at the class level or around a small group of related classes. In unit testing, an important distinction is seen based on whether or not the unit under test is isolated from its collaborators.

➤ Unit tests are usually written by the programmers using their regular tools. The only difference being the use of the same sort of unit testing framework.

There are further two types of testing in Unit Testing:

i. Sociable Unit Testing:

It focuses on testing the behaviour of modules by observing changes in their state. This treats the unit under test as a black box tested entirely through its interface.

ii. Solitary Unit Testing:

It looks at the interactions and collaborations between an object and its dependencies, which are replaced by test doubles.

- **INTEGRATION TESTING:**

➤ Integration tests are used to test communication between services. These tests are designed to test basic success and error paths over a network boundary.

➤ Different components interact with each other for their functional dependency, while communicating with each other integration test verifies the communication paths between the components and detect the interface defects.

➤ Here, all test modules are integrated together and tested as a subsystem. It checks that the communication paths between the subsystem work correctly while interacting with its peers. In micro service architecture, they are typically used to verify interactions between layers of integration code and the external components to which they are integrating.

➤ When the automated tests are written for the modules which are interacting with an external component, the basic goal is to verify the

modules are interacting sufficiently with the external component.

➢ It is very difficult to trigger abnormal behaviour such as a timeout or slow responses from the external component. Special tests are written to ensure that test respond as expected in the unexpected circumstances

➢ **Persistence integration tests** provide assurances that the schema assumed by the code matches that is available in the data store.

➢ With unit testing and integration testing, we can have confidence in the correctness of the logic contained in the individual modules that make up the micro service, but we cannot be sure that the micro services work together as a whole to satisfy business requirements.

- **COMPONENT TESTING:**

➢ Tests the full function of a single micro service. During this type of testing, any calls to external services are mocked in some way.

➢ A component is any well-encapsulated, coherent and independently replaceable part of a larger system. In a micro service architecture, the components are the services themselves.

➢ A component's complex behaviour is avoided by isolating it from its peers, also isolation help in controlling the test environment for the component.

- **CONTRACT TESING:**

➢ Test the agreed contract for APIs and other resources that are provided by the micro service.

➢ At the boundary of the external service, an Integrated contract test is done to verify the contract that is expected by the consuming service. This test

verifies   that   the   component   meets   a   contract.

➢ A test suite is written to verify only those aspects of the producing service that is in use. The behaviour of the service is not deeply tested, response latency and throughput should be within acceptable limits when input and output of the service call contain required attributes. This test is written by each test consuming team and then packaged. The main aim of this test is to know the impact of the changes made by the maintainers on the consumers.

- **END TO END TESTING:**

➢ End-to-End Testing, Tests a complete flow through the application or micro service. Usually used to test a golden path or to verify that the application meets external requirements.

➢ End-to-End testing tests the whole system from end to end. It verifies that the entire system meets the external requirements and eventually achieve its goal. Without bothering about the internal architecture of the application business goal should be achieved by the End-to-End testing.

➢ The system is fully deployed and is treated as a black box and the test is exercised. With Public interference through GUIs and API, the system is manipulated. End to End Tests is more business facing.

➢ This test verifies that the firewall, proxies, and load balancers are correctly configured.

In micro service architecture, for one behaviour, there are many micro services which interact to respond to that behaviour, an end-to-end testing provides value by adding coverage of gaps between the system.

5.6 Testing Methods:

I. BLACK-BOX TESTING:
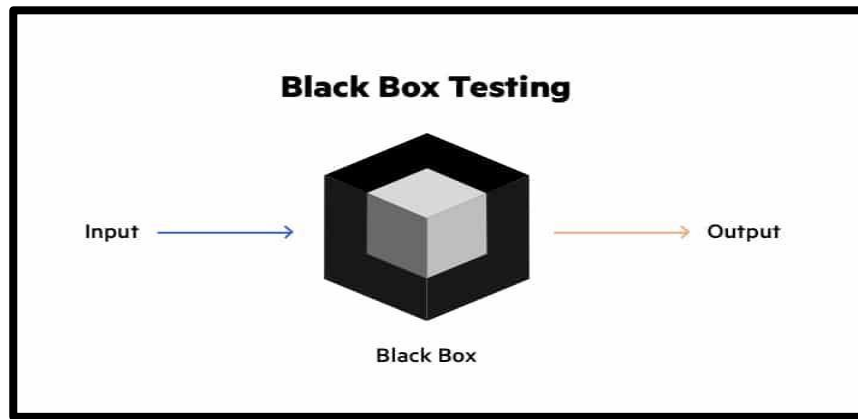
[Figure-5.6.1 BLACK-BOX TESTING]

➢ The technique of testing without having any knowledge of the interior workings of the application is called black-box testing.

➢ The tester is oblivious to the system architecture and does not have access to the source code. Typically, while performing a black-box test, a tester will interact with the system's user interface by providing inputs and examining outputs without knowing how and where the inputs are worked upon.

II. WHITE-BOX TESTING:

➢ White-box testing is the detailed investigation of internal logic and structure of the code. White-box testing is also called **glass testing** or **open-box testing**. In order to perform **white-box** testing on an application, a tester needs to know the internal workings of the code.
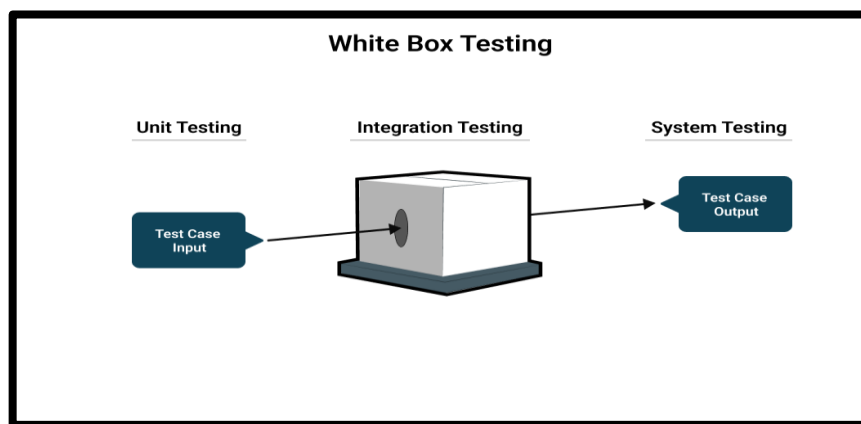
[Figure-5.6.2 WHITE-BOX TESTING]

➢ The tester needs to have a look inside the source code and find out which unit/chunk of the code is behaving inappropriately.

III. GREY-BOX TESTING:

➢ Grey-box testing is a technique to test the application with having a limited knowledge of the internal workings of an application.

➢ In software testing, the phrase the more you know, the better carries a lot of weight while testing an application.

➢ Mastering the domain of a system always gives the tester an edge over someone with limited domain knowledge.

➢ Unlike black-box testing, where the tester only tests the application's user interface; in grey-box testing, the tester has access to design documents and the database.

➢ Having this knowledge, a tester can prepare better test data and test scenarios while making a test plan.

Black Box Testing (Unknown Internal Code Structure) + White Box Testing (Known Internal Code Structure) = Grey Box Testing (Internal Code Structure Partially Known)
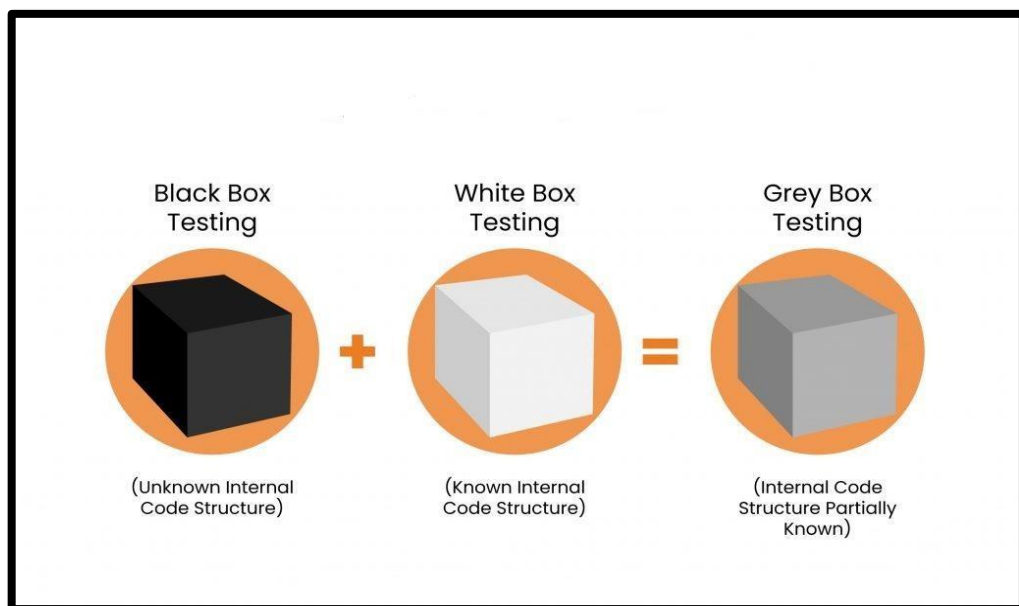
[Figure-5.6.3 GREY-BOX TESTING]

5.7 Test Case :

- **1st Test Case :**

TEST CASE ID	1-A
Name	Login Validation
Test Scenario	It Will Check Login Data Of User If Username And Password Both Are True
Test Step	1.Open Web-site 2.Click On "Log In" 3.Enter Detail 4.Click On Login
Test Data	Username: admin Password: 123
Expected Result	User Login As Admin Of System
Actual Result	As Expected
Pass / Fail	Pass

- **2ⁿᵈ Test Case:**

TEST CASE ID	1-B
Name	Login Validation
Test Scenario	It Will Check Login Data Of User If Username And Password Both Or Either One Of Them Is False
Test Step	1.Open Web-site 2.Click On "Log In" 3.Enter Detail 4.Click On Login
Test Data	Username: admin Password: 111
Expected Result	Message: Wrong username Or Password !!
Actual Result	As Expected
Pass / Fail	Pass

- **3ʳᵈ Test Case:**

TEST CASE ID	1-C
Name	Login Validation
Test Scenario	It Will Check Login Data Of User If Username And Password Both or One Of Them Is Empty
Test Step	1.Open Web-site 2.Click On "Log In" 3.Enter Detail 4.Click On Login
Test Data	Username: … Password: …
Expected Result	Message: Enter Username Or Password First!!
Actual Result	As Expected
Pass / Fail	Pass

- **4th Test Case:**

TEST CASE ID	2-A
Name	Customer Registration Validation
Test Scenario	It Will Check All Data Of Filed That Enter By user To Create Customer Account
Test Step	1.Open Web-site 2.Click On "Register As Customer" 3.Enter Detail 4.Click On Register
Test Data	Customer Name : Customer1 Address : Chitra Bhavnagar Email : Customer1@gmail.com Mobile No : 8460888834 Username : Customer1 Password : Customer123 Repeat Password : Customer123
Expected Result	Message : Account Created Successfully !!
Actual Result	As Expected
Pass / Fail	Pass

- **5ᵗʰ Test Case:**

TEST CASE ID	2-B
Name	Customer Registration Validation
Test Scenario	It Will Check All Data Of Filed That Enter By user To Create Customer Account And Check For Already Exist Email.
Test Step	1.Open Web-site 2.Click On "Register As Customer" 3.Enter Detail 4.Click On Register
Test Data	Customer Name : Customer1 Address : Chitra Bhavnagar Email : Customer1@gmail.com Mobile No : 8460888833 Username : Customer2 Password : Customer123 Repeat Password : Customer123
Expected Result	Message : Email Already Exist !!
Actual Result	As Expected
Pass / Fail	Pass

- **6th Test Case:**

TEST CASE ID	2-C
Name	Customer Registration Validation
Test Scenario	It Will Check All Data Of Filed That Enter By user To Create Customer Account And Check For Already Exist Mobile No.
Test Step	1.Open Web-site 2.Click On "Register As Customer" 3.Enter Detail 4.Click On Register
Test Data	Customer Name : Customer2 Address : Chitra Bhavnagar Email : Customer2@gmail.com Mobile No : 8460888834 Username : Customer2 Password : Customer123 Repeat Password : Customer123
Expected Result	Message : Mobile No Already Exist !!
Actual Result	As Expected
Pass / Fail	Pass

- **7th Test Case:**

TEST CASE ID	2-D
Name	Customer Registration Validation
Test Scenario	It Will Check All Data Of Filed That Enter By user To Create Customer Account And Check For Already Exist Username.
Test Step	1.Open Web-site 2.Click On "Register As Customer" 3.Enter Detail 4.Click On Register
Test Data	Customer Name : Customer2 Address : Chitra Bhavnagar Email : Customer2@gmail.com Mobile No : 8460888833 Username : Customer1 Password : Customer123 Repeat Password : Customer123
Expected Result	Message : Username Already Exist !!
Actual Result	As Expected
Pass / Fail	Pass

- **8ᵗʰ Test Case**

TEST CASE ID	2-E
Name	Customer Registration Validation
Test Scenario	It Will Check All Data Of Filed That Enter By user To Create Customer Account And Check All Filed Must Have Data
Test Step	1.Open Web-site 2.Click On "Register As Customer" 3.Enter Detail 4.Click On Register
Test Data	Customer Name : … Address : … Email : … Mobile No : … Username : … Password : … Repeat Password : …
Expected Result	Message : All Detail Must Be Filed !!
Actual Result	As Expected
Pass / Fail	Pass

- **8th Test Case:**

TEST CASE ID	3-A
Name	Manager Registration Validation
Test Scenario	It Will Check All Data Of Filed That Enter By user To Create Manager Account
Test Step	1.Open Web-site 2.Click On "Register As Manager" 3.Enter Detail 4.Click On Register
Test Data	Manager Name : Manager1 Address : Chitra Bhavnagar Email : Manager1@gmail.com Mobile No : 9924700970 Username : Manager1 Password : Manager123 Repeat Password : Manager123
Expected Result	Message : Manager Details Add For Inquiry !!
Actual Result	As Expected
Pass / Fail	Pass

- **9th Test Case:**

TEST CASE ID	3-B
Name	Manager Registration Validation
Test Scenario	It Will Check All Data Of Filed That Enter By user To Create Manager Account And Check For Already Exist Email.
Test Step	1.Open Web-site 2.Click On "Register As Manager" 3.Enter Detail 4.Click On Register
Test Data	Manager Name : Manager2 Address : Chitra Bhavnagar Email : Manager1@gmail.com Mobile No : 9924700977 Username : Manager2 Password : Manager123 Repeat Password : Manager123
Expected Result	Message : User With This Email Is Already Exist !!
Actual Result	As Expected
Pass / Fail	Pass

- **10th Test Case:**

TEST CASE ID	3-C
Name	Manager Registration Validation
Test Scenario	It Will Check All Data Of Filed That Enter By user To Create Manager Account And Check For Already Exist Mobile No.
Test Step	1.Open Web-site 2.Click On "Register As Manager" 3.Enter Detail 4.Click On Register
Test Data	Manager Name : Manager2 Address : Chitra Bhavnagar Email : Manager2@gmail.com Mobile No : 9924700970 Username : Manager2 Password : Manager123 Repeat Password : Manager123
Expected Result	Message : User With This Mobile Number Already Exist !!
Actual Result	As Expected
Pass / Fail	Pass

- **11ᵗʰ Test Case:**

TEST CASE ID	3-D
Name	Doctor Registration Validation
Test Scenario	It Will Check All Data Of Filed That Enter By user To Create Doctor Account And Check All Filed Must Have Data
Test Step	1.Open Web-site 2.Click On "Register As Doctor" 3.Enter Detail 4.Click On Register
Test Data	Manager Name : Manager1 Address : Chitra Bhavnagar Email : Manager2@gmail.com Mobile No : 9924700977 Username : Manager1 Password : Manager123 Repeat Password : Manager123
Expected Result	Message : Username Already Exist !!
Actual Result	As Expected
Pass / Fail	Pass

- **12th Test Case:**

TEST CASE ID	3-E
Name	Doctor Registration Validation
Test Scenario	It Will Check All Data Of Filed That Enter By user To Create Doctor Account And Check All Filed Must Have Data
Test Step	1.Open Web-site 2.Click On "Register As Doctor" 3.Enter Detail 4.Click On Register
Test Data	Manager Name : … Address : … Email : … Mobile No : … Username : … Password : … Repeat Password : …
Expected Result	Message : All Detail Must Be Filed !!
Actual Result	As Expected
Pass / Fail	Pass

5.8 Implementation Approach :
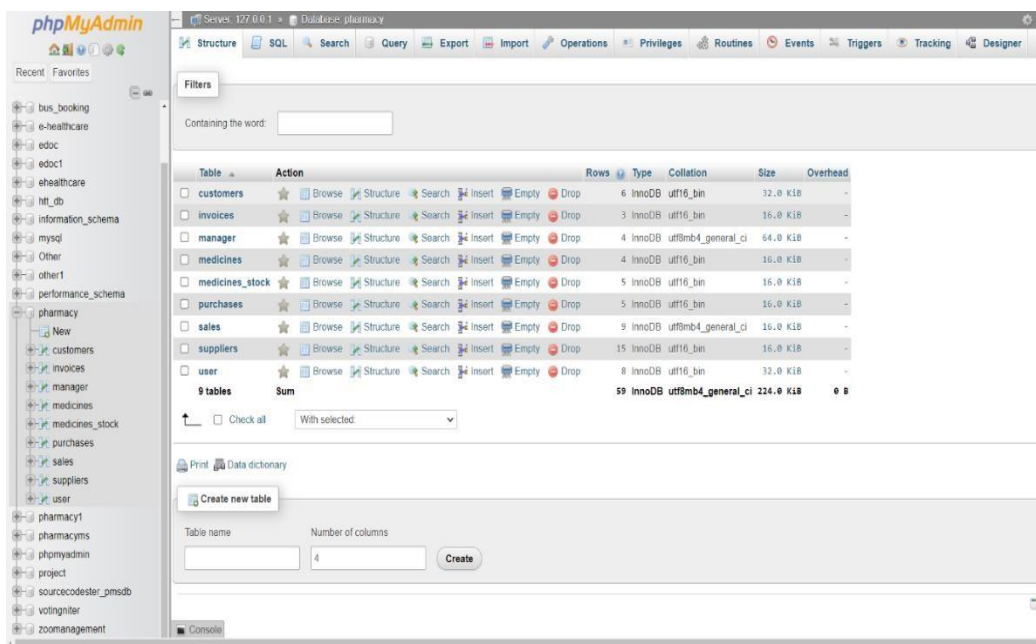
XAMPP is a completely free, easy to install Apache distribution containing MariaDB, PHP, and Perl. The XAMPP open source package has been set up to be incredibly easy to install and to use. Hi Apache friends! We just released a new version XAMPP for windows, Linux and OS X. Download from portableApps.com XAMPPL a uncheer can run from a cloud folder, external drive, or local folder without installing into windows. It's even better with the Portable Apps.com platform for easy installs and automatic updates. Double-click on the name of your hard drive, then your Xampp folder and finally your htdocs folder.

You need to move all required PHP files into your folder. If you are using a, Mac, go to your XAMPP control panel and click the volumes tab. Now click Mount, then explorer and double click on your htdocs folder. XAMPP is now become number one choice for web developer for its distribution containing MYSQL, PHP and Perl. Features of XAMPP are listed below: it is updated and lates version of Apache, MariaDB, PHP and Perl. It comes with other modules including OpenSSL, phpMyAdmin, Media Wiki, Joomla, WordPress etc. It comes in full and standard version.

CONCLUSION

6.1 Conclusion :

This is a Kind of web application Which Providing detailed information about an event in any city. events like painting exhibition, Happy Street, fairs, circus, Dance competition, comedy club etc.The event information will be delivered to the user with the help of event organists through email.
We can get the information of upcoming events online.

There are 2 types of platforms.
          1. event organiser and
        2. normal user.

There are different types of categories of different events and these categories have some sub-categories.From this, it is known which event has taken place in which city. event organiser Provide facility of subscribe event and those who are interested can participate and watch the event through the subscribe button. If someone subscribes, they should be informed by Gmail.

6.2 Limitation of System :

Technical Issues:

Like any software, pharmacy management systems are susceptible to technical glitches, bugs, or downtime, which can disrupt workflow and lead to delays in service.

Integration Challenges:

Integrating the pharmacy management system with other existing systems within a healthcare facility (e.g., electronic health records) can be complex and may require additional resources.

Data Security Concerns:

Given the sensitive nature of patient information, maintaining robust data security measures is crucial. However, pharmacy management systems can be vulnerable to cybersecurity threats if not properly secured.

Customization Limitations:

While many pharmacy management systems offer customization options, there may be limitations in tailoring the software to specific workflow preferences or unique requirements of a pharmacy.

Dependency on Technology:

Reliance on technology means that any disruptions in internet connectivity or hardware failure could potentially halt pharmacy operations temporarily.

Regulatory Compliance:

Keeping up with changing regulatory requirements and ensuring that the pharmacy management system remains compliant with industry standards can be challenging.

User Support:

Depending on the provider, the level of customer support and maintenance services provided for the pharmacy management system may vary, which could impact the resolution time for any issues that arise.

Scalability:

As pharmacies grow or expand, scalability can become a concern. Some pharmacy management systems may have limitations in accommodating the increased volume or complexity of operations.

6.3 Future Scope of System:

Personalized Medicine Integration: As the field of pharmacogenomics advances, pharmacy management systems could be tailored to support personalized medicine initiatives. They could store and analyze genetic data to optimize medication selection and dosing based on individual patient characteristics.

1.Mobile Applications:

Increasing use of mobile devices opens up opportunities for pharmacy management systems to offer mobile applications for patients and healthcare providers. These apps could enable medication reminders, refill requests, telepharmacy services, and access to health information.

2.AI and Smart Devices:

Internet of Things (IoT) devices such as smart pill bottles, medication dispensers, and wearable sensors can connect to pharmacy management systems, providing real-time data on medication adherence, vital signs, and patient health status.

3.Predictive Analytics for Demand Forecasting:

By analyzing historical data and trends, pharmacy management systems could use predictive analytics to forecast medication demand more accurately. This can help pharmacies optimize inventory levels, reduce waste, and ensure timely availability of essential medications.

4.Enhanced Patient Engagement Tools:

Future pharmacy management systems may include advanced patient engagement features such as interactive medication adherence programs, virtual counseling sessions, and personalized health education resources.

5.Interoperability with Healthcare Ecosystem:

Improved interoperability standards would enable seamless integration between pharmacy management systems and other healthcare information systems, such as electronic health records (EHRs) and medical billing platforms.

6.Continuous Quality Improvement:

Pharmacy management systems of the future will likely focus on continuous quality improvement, leveraging data analytics and feedback mechanisms to identify areas for optimization and enhance the overall quality of pharmaceutical care.

Overall, the future of pharmacy management systems lies in leveraging cutting-edge technologies to enhance patient outcomes, improve operational efficiency, and adapt to evolving healthcare needs.

6.4 Bibliography:

We took references from below describe listing websites to build successful web Application

Youtube : https://www.youtube.com/

Google : https://www.google.com/

ChatGTP : https://chat.openai.com/