

**A Project Report**

**On**

**MEDICAL STORE MANAGEMENT SYSTEM**



Submitted in partial fulfilment of the  
requirements for the award of the degree  
of

**MASTER OF COMPUTER APPLICATIONS**

**SUBMITTED BY**

**DAMARASINGU.GOWTHAM BHASKAR**

**(22P31F0009)**

**Under the Esteemed Guidance of**

**Ms.Mohammad.Gousia, MCA**

**Assistant Professor**



**DEPARTMENT OF MCA**

**ADITYA COLLEGE OF ENGINEERING AND TECHNOLOGY**

(Approved by AICTE, Affiliated to JNTUK & Accredited by NBA, NAAC with 'A+'Grade)

Recognized by UGC under the sections 2(f) and 12(B) of the UGC act 1956

SURAMPALEM-533437, East Godavari District, ANDHRA PRADESH.

2022-2024

# ADITYA COLLEGE OF ENGINEERING AND TECHNOLOGY

(Approved by AICTE, Affiliated to JNTUK & Accredited by NBA, NAAC with 'A+' Grade)

Recognized by UGC under the sections 2(f) and 12(B) of the UGC act 1956

SURAMPALEM-533437, E.G.Dist, ANDHRA PRADESH.

2022-2024

## DEPARTMENT OF MCA



### CERTIFICATE

This is to certify that the project work entitled, “**MEDICAL STORE MANAGEMENT SYSTEM**”, is a Bonafide work carried out by **DAMARASINGU.GOWTHAM BHASKAR** bearing Regd.No:**22P31F0009** submitted to the requirements for the award of the Computer Applications in partial fulfilment of the requirements for the award of degree of **MASTER OF COMPUTER APPLICATIONS** from **Aditya College of Engineering and Technology**, Surampalem during the academic year **2023-2024**.

#### Project Guide

**Ms.Mohammad.Gousia, MCA**

Assistant Professor,

Department of MCA,

Aditya College of Engg & Tech,

Surampalem-533437.

#### Head of the Department

**Mr.R.V.V.N.BheemaRao, M.Tech**

Associate Professor,

Department of MCA,

Aditya College of Engg & Tech,

Surampalem-533437.

#### External Examiner

## **DECLARATION**

I hereby declare that the project entitled “**MEDICAL STORE MANAGEMENT SYSTEM**” done on my own and submitted to **Aditya College of Engineering & Technology, Surampalem** has been carried out by me alone under the guidance of **Ms.Mohammad.Gousia**.

Place: Surampalem

Date:

**(D.GOWTHAM BHASKAR)**

**22P31F0009**

## ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that I have now the opportunity to express my gratitude for all of them.

The first person I would like to thank is my guide **Ms.Mohammad.Gousia, MCA, Assistant Professor, Aditya College Of Engineering And Technology, Surampalem**, Her wide knowledge and logical way of thinking have made a deep impression on me. Her understanding, encouragement, and personal guidance have provided the basis for this project. She is a source of inspiration for innovative ideas and his kind support is well known to all her students and colleagues.

I wish to thank **Mr.R.V.V.N.Bheemaroo, M.Tech, Head of the Department, Aditya College Of Engineering And Technology, Surampalem**, has extended his support for the success of this project.

I also wish to thank **Dr.D.Sanjay, Principal, Aditya College Of Engineering And Technology, Surampalem**, who has extended his support for the success of this project.

I would like to thank all the **Management & Technical Supporting Staff** for their timely help throughout the project.

# **ABSTRACT**

## **ABSTRACT**

The Medical store management system is designed to streamline operations and improve customer experience in the healthcare sector. The system includes key features such as inventory management, online ordering, online payments and customer management.

The main aim of medical store management system is to improve the efficiency of medical stores by providing real-time access to stock information, supporting online transactions, and automating routine tasks. The system will focus on the overall healthcare service by ensuring the availability of medicines and providing a convenient platform for customers to access medical supplies.

During the development of my website the technologies that I used for front-end are, HTML, CSS, JAVASCRIPT and for back-end I used Python with Django framework.

## **LIST OF CONTENTS**

|  | <b>PAGE NO</b> |
|--|----------------|
| <b>Chapter 1: Introduction</b>         |                |
| 1.1 Brief Introduction of The Project  | 01             |
| 1.2 Motivation of The Project          | 01             |
| 1.3 Objective of The Project           | 02             |
| 1.4 Organization of The Project        | 02             |
| <b>Chapter 2: Literature Survey</b>    | 04             |
| <b>Chapter 3: System Analysis</b>      |                |
| 3.1 Existing System                    | 06             |
| 3.2 Proposed System                    | 06             |
| 3.3 Feasibility Study                  | 07             |
| 3.4 Functional Requirements            | 08             |
| 3.5 Non-Functional Requirements        | 08             |
| 3.6 Software And Hardware Requirements | 08             |
| <b>Chapter 4: System Design</b>        |                |
| 4.1 Introduction                       | 10             |
| 4.2 Data Dictionary                    | 10             |
| 4.3 Data Flow Diagram                  | 10             |
| 4.4 System Architecture Diagram        | 11             |
| 4.5 UML Diagram                        | 11             |
| 4.5.1 Use Case diagram                 | 13             |
| 4.5.2 Class Diagram                    | 14             |
| 4.5.3 Sequence Diagram                 | 16             |
| 4.5.4 Collaboration Diagram            | 17             |
| 4.5.5 Activity Diagram                 | 19             |

|  |    |
|--|----|
| <b>Chapter 5: Technology Description</b> | 20 |
| <b>Chapter 6: Sample code</b>            | 24 |
| <b>Chapter 7: Testing</b>                | 43 |
| <b>Chapter 8: Screen Shots</b>           | 47 |
| Conclusion                               | 58 |
| Bibliography                             | 59 |



## LIST OF FIGURES

| S.NO | Figure No | Name Of The Figure          | Page No |
|------|-----------|-----------------------------|---------|
| 01   | 4.4       | System Architecture Diagram | 11      |
| 02   | 4.5.1     | Use Case Diagram            | 13      |
| 03   | 4.5.2     | Class Diagram               | 14      |
| 04   | 4.5.3     | Sequence Diagram            | 16      |
| 05   | 4.5.4     | Collaboration Diagram       | 17      |
| 06   | 4.5.5     | Activity Diagram            | 19      |

## LIST OF SCREEN SHOTS

| S.NO | Screen No | Name of the Screen       | Page No |
|------|-----------|--------------------------|---------|
| 01   | 8.1       | Home Page                | 48      |
| 02   | 8.2       | Login Page               | 49      |
| 03   | 8.3       | Sign up Page             | 50      |
| 04   | 8.4       | Medicines Page           | 51      |
| 05   | 8.5       | Products Page            | 52      |
| 06   | 8.6       | Orders Page              | 53      |
| 07   | 8.7       | Contact Us Page          | 54      |
| 08   | 8.8       | Admin Login Page         | 55      |
| 09   | 8.9       | Site Administration Page | 56      |

# **CHAPTER-1**

## **INTRODUCTION**

## 1.INTRODUCTION

Medical store management system is a digital platform used to streamline the operations of medical stores or pharmacies in the online realm. This innovative solution empowers users to conveniently browse, purchase, and receive a wide array of medical products and services remotely, offering a seamless and efficient experience.

### 1.1 Brief Introduction Of The Project:

The main objective of the project is to create an medical store management system that allows users to search and purchase medicines based on requirements, prescription and needs. The selected medicines are displayed in a list format and the user can order their medicines using a web browser and then purchase it using cash on delivery option. Mobile friendly design for accessibility across various devices and screen sizes. The rise of ecommerce played a pivotal role in the growth of medical store management system. This is used to reduce the time complexity. By harnessing the power of technology and implementing a robust architectural framework, the system effectively balances functionality, usability, 24/7 availability and security to provide a highly satisfying and secure online shopping experience for users worldwide. The medical store management system project is a comprehensive initiative designed to create a dynamic and user-friendly platform for medicines to explore, purchase medicines. This project aims to capitalize on the convenience and accessibility offered by online platforms, providing a seamless experience for both buyers and sellers. The Administrator will have additional functionalities when compared to the common user such as adding medicines and deleting medicines from the store. An effective medical store management system streamlines the medicine-buying process, enhances user engagement, and provides a platform for both buyers and sellers to interact in the digital marketplace. It's a great way to purchase medicines with price comparisons and diverse product range!.

### 1.2 Motivation Of The Project:

It is important to approach the project with a commitment to ethical and legal considerations, ensuring the well-being of users. Medical store management system project was motivated by several factors each with its unique appeal:

- **Healthcare Accessibility:** A primary motivation is to improve healthcare accessibility. medical store management system can provide a convenient platform for individuals to access medications and healthcare products.
- **Convenience and Efficiency:** The convenience of ordering medications from home can significantly improve the overall efficiency of obtaining necessary healthcare supplies.
- **Global Impact:** Creating a medical store management system with global reach can have a positive impact on healthcare on an international scale. It allows people around the world to access a diverse range of medications and healthcare products.
- **Innovation in Healthcare:** Developing a medical store management system involves innovating within the healthcare sector. This can include implementing features like personalized health information.
- **Public Health Improvement:** By ensuring the availability of genuine medications and promoting responsible usage, a medical store management system project contributes to public health improvement, potential side effects, and proper usage.

- **Digital Health Solutions:** Contributing to the growing field of digital health, a medical store management system project can be a part of a broader ecosystem of health-related technologies, fostering innovation and improvement in overall healthcare delivery.
- **Personal Experience or Passion:** If you have a personal experience with challenges in obtaining medications or a genuine passion for improving healthcare access, this can be a powerful motivation to embark on a medical store management system project.

### 1.3 Objective Of The Project:

Interest to develop a good user friendly website with many online transactions using a database. To increase my knowledge horizon in technologies like DJANGO, SQLITE, CSS, HTML. To gain good experience in DJANGO before joining in a fulltime job.

- **Convenience and Accessibility:** Enable users to access a vast number of medicines from the comfort of their homes, avoiding the need to visit a physical pharmacy.
- **Secure and Seamless Transactions:** Ensure secure transactions by integrating reliable payments, safeguarding user data, and providing a seamless checkout process for a trustworthy purchasing experience.
- **Global Marketplace:** Establish a global marketplace that transcends geographical boundaries, allowing users to explore and purchase medicines from various brands, companies, and price range.
- **Information Availability:** Users can access detailed information about medications, potential side effects, and usage instructions, empowering them to make informed decisions about their health.
- **Inventory Management for Sellers:** Provide a user-friendly dashboard for sellers, including brands and retailers, to efficiently manage their medicine inventory, update product information, and track sales.

### 1.4 Organization of the Project:

- **Literature Survey:** This chapter contains base information of previous models.
- **System Analysis:** The description of the current system, the planned system, and the required specifications make up the majority of this chapter.
- **System Design:** This chapter is consisting of modules description and algorithms with example and use case diagrams, class diagrams, sequence diagrams, Collaboration diagrams and activity diagrams.
- **Technology Description:** This chapter mainly consists of the technology description of this project.
- **Sample Code:** This chapter consisting of sample code for the few modules.
- **Testing:** This chapter contains of testing techniques and test cases for modules.
- **Screen Shots:** This chapter is mainly consisting of output screens of this project.
- **Conclusion:** Main Conclusion of the project is to segmenting the customers based on characteristics.

## **CHAPTER-2**

### **LITERATURE SURVEY**

## **2. LITERATURE SURVEY**

### **2.1 Pharmacy Management System**

The purpose of the pharmacy management system is to computerise manual systems and replace them with one. The system should be able to carry out tasks as directed by the pharmacy manager in an economical, practical, and efficient manner. The software handles every aspect of running a pharmacy, including sales, entering new inventory, creating invoices, figuring out taxes and debt, calculating employee compensation, providing product information, creating charts that show various statistics, and overseeing staff work.

### **2.2 Medical Store Management System**

This particular project deals with the problems on managing a medical shop and avoids the problems which occur when carried manually. The Medical Management System is a windows-based software designed for registration and management of patient's records and easy access of the records. The system will be used to assist the register, doctors, lab technicians to store and manage patient records in a hospital or clinic for easier access and reference. All these activities are done routinely and would be cumbersome on the employees if done manually hence need of an efficient easy to use management software that will help ease the work load on employees in clinic/hospital.

### **2.3 Generic Medicine Finder**

Generic medicine are those medicine that people can Afford it in less and affordable cost There are two types of medicine first is Generic Medicine and second is Brand medicine Brand Medicine are those who's Cost are very high and at unaffordable price. Means if the Brand medicine are 100Rs cost of per medicine then the Generic medicine are 10Rs cost of per medicine. Means the Generic Medicine are 90 to 80 percent Less in cost then Brand medicine.

### **2.4 Advances in Pharmacy Practice**

In today's specialised and globalised world, pharmacy education in India faces significant challenges and shortcomings. There is an urgent need to start an academic activity to achieve curriculum revision that keeps up with current and upcoming developments in the pharmacy profession. Sadly, throughout the years, when developing curricula for teaching at the diploma and degree levels, not enough attention was placed on bolstering the components of community pharmacy, hospitals, and clinical pharmacy. The academic-practice relationship can create new and inventive practise innovations that will enhance patient outcomes with the right leadership and support, a change in present professional education and training processes, and a dedication to nurturing future innovators.

## **CHAPTER-3**

### **SYSTEM ANALYSIS**



### 3. SYSTEM ANALYSIS

#### 3.1 Existing System

The project refers to the system that is being followed till now. The current existing system includes only the branding and name of medicine and its difficult for uneducated and normal people to understand for what purpose the specific medicine is being used. Some of the current existing systems do not provide expiry dates.

**Draw backs:**

- Limited accessibility, especially for individuals those who are not in remote areas.
- Manual prescription handling can lead to errors in data entry, and record-keeping.
- Managing inventory, leads to issues such as overstocking, stockouts, and expired medications.
- Manual processes, including prescription verification, and customer service, can be time-consuming.
- Limited information availability storing and retrieving patient information and prescription history.

#### 3.2 Proposed System

The project that is being developed will contain the complete information about the medicine along with its expiry details. The user can also share the feedback by logging into our website.

Online medical stores offer various advantages of digital technology to streamline processes and enhance accessibility to healthcare products and services. Here are some key advantages of medical store management system.

**Advantages:**

- Detailed information about medications including expiry date available on the platform.
- User privacy in online platforms provide a way for users to order sensitive medications without the need for face-to-face interactions.
- Users can browse and purchase medications and healthcare products from the comfort of their homes, 24/7.
- Online medical stores eliminate the need for physical travel to saving time for users.

- Online platforms often provide transparent pricing, allowing users to compare the costs of medications across different pharmacies.

### **3.3 Feasibility Study**

A Feasibility study evaluates the project's potential for success; therefore, perceived objectivity is an important factor in the credibility of the study for potential investors and lending institutions. It must therefore, be conducted with an objective, unbiased approach to provide information upon which decisions can be based. Here, we discuss 3 major feasibility studies required for our project.

- Economic Feasibility
- Technical Feasibility
- Operational Feasibility

#### **ECONOMIC FEASIBILITY**

Economic Feasibility defines whether the expected benefit equals or exceeds the expected costs. It is also commonly referred to as cost/benefit analysis. The procedure is to determine the benefits and the savings expected from the system and compare them with the costs. A proposed system is expected to outweigh the costs. This is small project with no cost for development. The system is easy to understand use. Therefore, there is no need to spend on training to use the system. This system has the potential to grow by adding functionalities for persons. Hence, the project could have economic benefits in future.

#### **TECHNICAL FEASIBILITY**

Technical feasibility is carried out to determine whether the project is feasible in terms of software, hardware, personnel, and expertise, to handle the completion of the project. It considers determining resources for the proposed system. As the system is developed using python, it is platform independent. Therefore, the users of the system can have average processing capabilities, running on any platform. The technology is one of the latest hence the system is also technically feasible.

#### **OPERATIONAL FEASIBILITY**

Operational feasibility is the measure of how well a proposed system solves the problems with the users. Operational feasibility is dependent on human resources available for the project and involves projecting whether the system will be used if it is developed and implemented. The project is operationally feasible for the users as nowadays almost all the humans are familiar with digital technology.

### 3.4 Functional Requirements

- The system should keep records of registration of customers.
- The system should keep the records of medicines.
- The system should keep the record of daily sell.
- The system should keep the record of products.
- The system should store the feedback given by the customer.
- The system should keep details about the product it is delivered or not etc.

### 3.5 Non Functional Requirements

- Only authorized employees of the firm can access the subsystem's secured page.
- Simple user interface has be created to make it easier to use.
- The systems must always be accessible around the clock, every day of the week.
- If a significant system fault occurs, it should be fixed within 1 to 2 working days to minimize disruption to business operations.

### 3.6 Software and Hardware requirements

#### **MINIMUM SOFTWARE REQUIREMENTS :-**

- Operating System : Windows 10
- Technologies : Python 3.9
- Tools : Visual Studio

#### **MINIMUM HARDWARE REQUIREMENTS :-**

- Processor : intel core i3
- RAM : 4 GB
- Hard Disk : 500 GB
- Speed : 2.4 GHZ

## **CHAPTER-4**

### **SYSTEM DESIGN**

## 4.SYSTEM DESIGN

### 4.1 Introduction

The purpose of the design phase is to plan a solution of the problem specified by the requirement document. This phase is the first step in moving from the problem domain to the solution domain. In other words, starting with what is needed, design takes us toward how to satisfy the needs. The design of a system is perhaps the most critical factor affecting the quality of the software; it has a major impact on the later phase, particularly testing, maintenance. The output of this phase is the design document. This document is similar to a blueprint for the solution and is used later during implementation, testing and maintenance. The design activity is often divided into two separate phases System Design and Detailed Design.

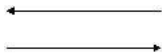
### 4.2 Data Dictionary

A data dictionary contains metadata and data about the database. The data dictionary is very important as it contains information such as what is in the database, who can access it, where is the database physically stored etc. The users of the database normally don't interact with the data dictionary, it is only handled by the database administrators. The data dictionary in general contains information about the following Names of all the database tables and their schemas.

### 4.3 Data Flow Diagram

Data flow-oriented techniques advocate that the major data items handled by a system must be first identified and then the processing required on these data items to produce the desired outputs should be determined. The DFD (also called as bubble chart) is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on these data, and the output generated by the system. It was introduced by De Macro (1978), Gane and Sarson (1979). The primitive symbols used for constructing DFD's are:

**Data flow:**



**Process:**



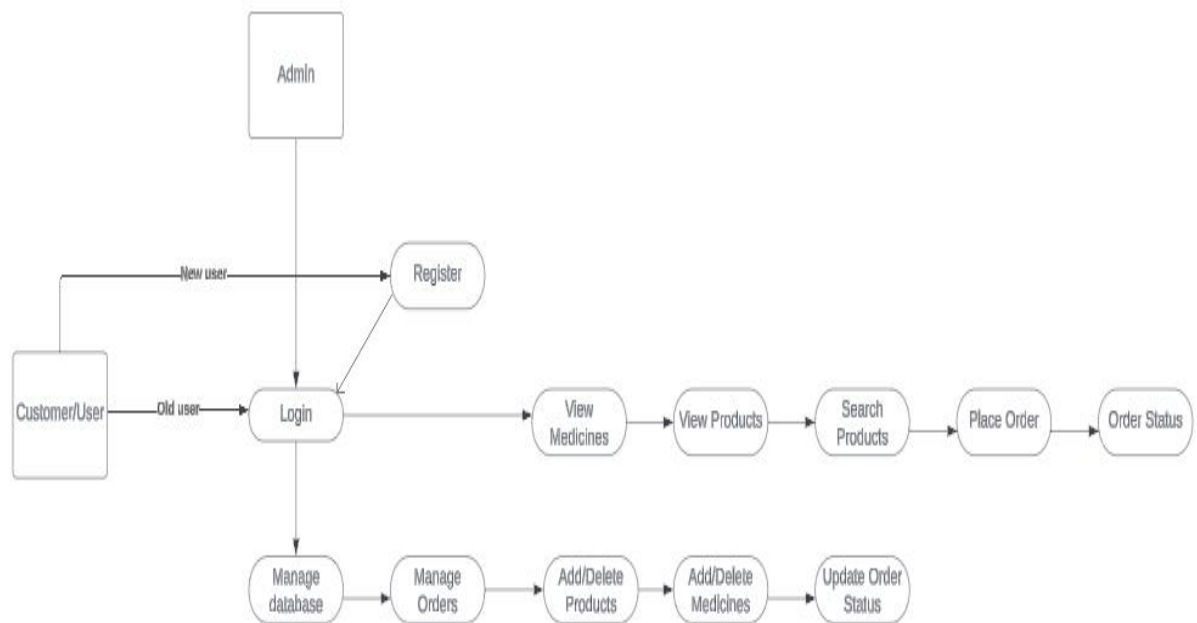
**Source:**



**Rhombus:**



#### 4.4 System Architecture Diagram



#### 4.5 UML Diagrams

The unified modelling language allows the software engineer to express an analysis model using the modelling notation that is governed by a set of syntactic semantic and pragmatic rules. A UML system is represented using five different views that describe the system from distinctly different perspective. UML is specifically constructed through two different domains UML Analysis modelling, this focuses on the user model and structural model views of the system UML design modelling, which focuses on the behavioural modelling, implementation modelling and environmental model views. These are divided into the following types.

- **Use case Diagram**
- **Class Diagram**

## 4.5.1 Use case Diagram

Use Case diagrams identify the functionality provided by the system (use cases), the users who interact with the system (actors), and the association between the users and the functionality. Use Cases are used in the Analysis phase of software development to articulate the high-level requirements of the system. The primary goals of Use Case diagrams include:

- Providing a high level view of what the system does.
- Identifying the users (“actors”) of the system.
- Determining the areas needed human-computer interfaces.

### Graphical Notation:

The basic components of Use Case diagrams are the

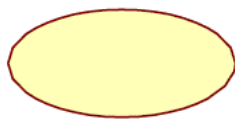
- Actor
- Use Case
- Association

**Actor:** An Actor, as mentioned, is a user of the system, and is depicted using a stick figure. The role of the user is written beneath the icon. Actors are not limited to humans. If a system communicates with another application, and expects input or delivers output, then that application can also be considered an actor.



**Actor Role Name**

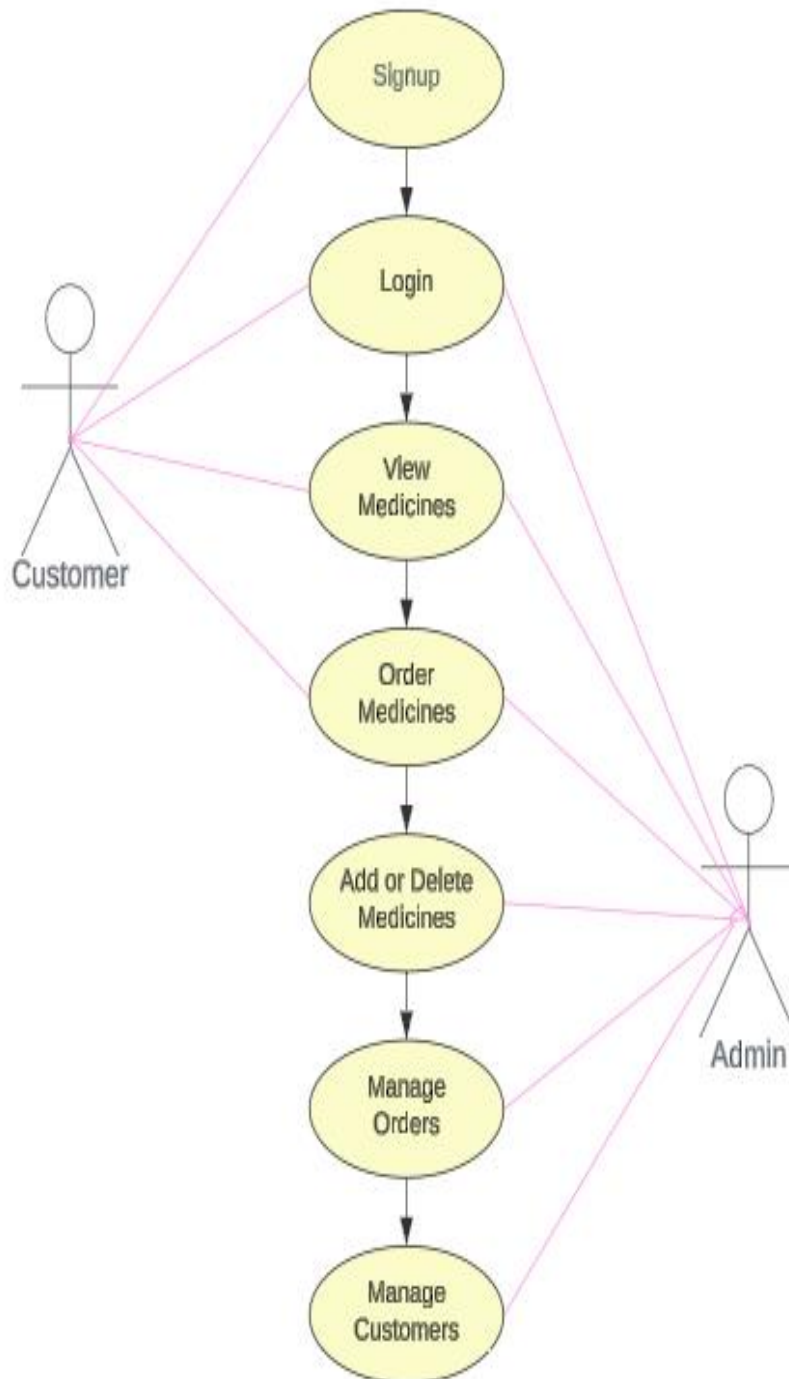
**Use Case:** A Use Case is functionality provided by the system; Use Cases are depicted with an ellipse. The name of the use case is written within the ellipse.



**Association:** These Associations are used to link Actors with Use Cases, and indicate that an Actor participates in the Use Case in some form.



Use case Diagram

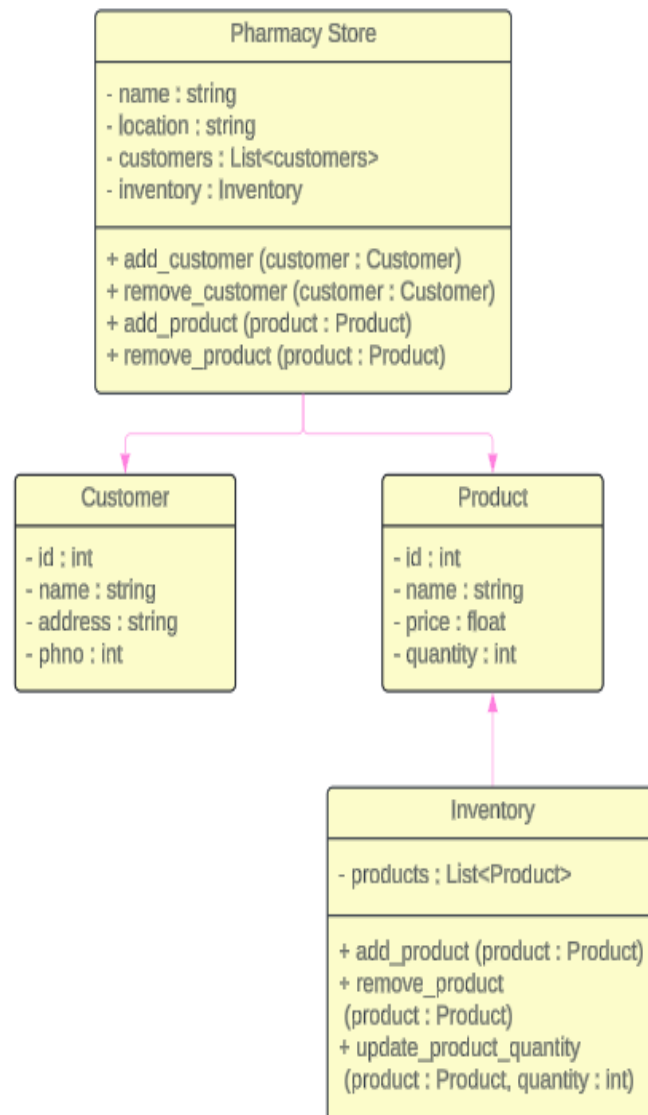




#### 4.5.2 Class Diagram

Class diagrams identify the class structure of a system, including the properties and methods of each class. Also depicted are the various relationships that can exist between classes, such as an inheritance relationship. Part of the popularity of Class diagrams stems from the fact that many CASE tools, such as Rational XDE, will autogenerate code in a variety of languages, these tools can synchronize models and code, reducing the workload, and can also generate Class diagrams from object-oriented code.

#### Class Diagram



### 4.5.3 Sequence Diagram

Sequence diagrams document the interactions between classes to achieve a result, such as a use case. Because UML is designed for object-oriented programming, these communications between classes are known as messages. The Sequence diagram lists objects horizontally, and time vertically, and models these messages over time.

**Graphical Notation:** In a Sequence diagram, classes and actors are listed as columns, with vertical lifelines indicating the lifetime of the object over time.

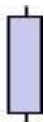
**Object** Objects are instances of classes, and are arranged horizontally. The pictorial representation for an Object is a class (a rectangle) with the name prefixed by the object name (optional) and a semi-colon.



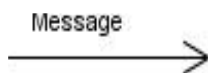
**Lifeline:** The Lifeline identifies the existence of the object over time. The notation for a Lifeline is a vertical dotted line extending from an object.



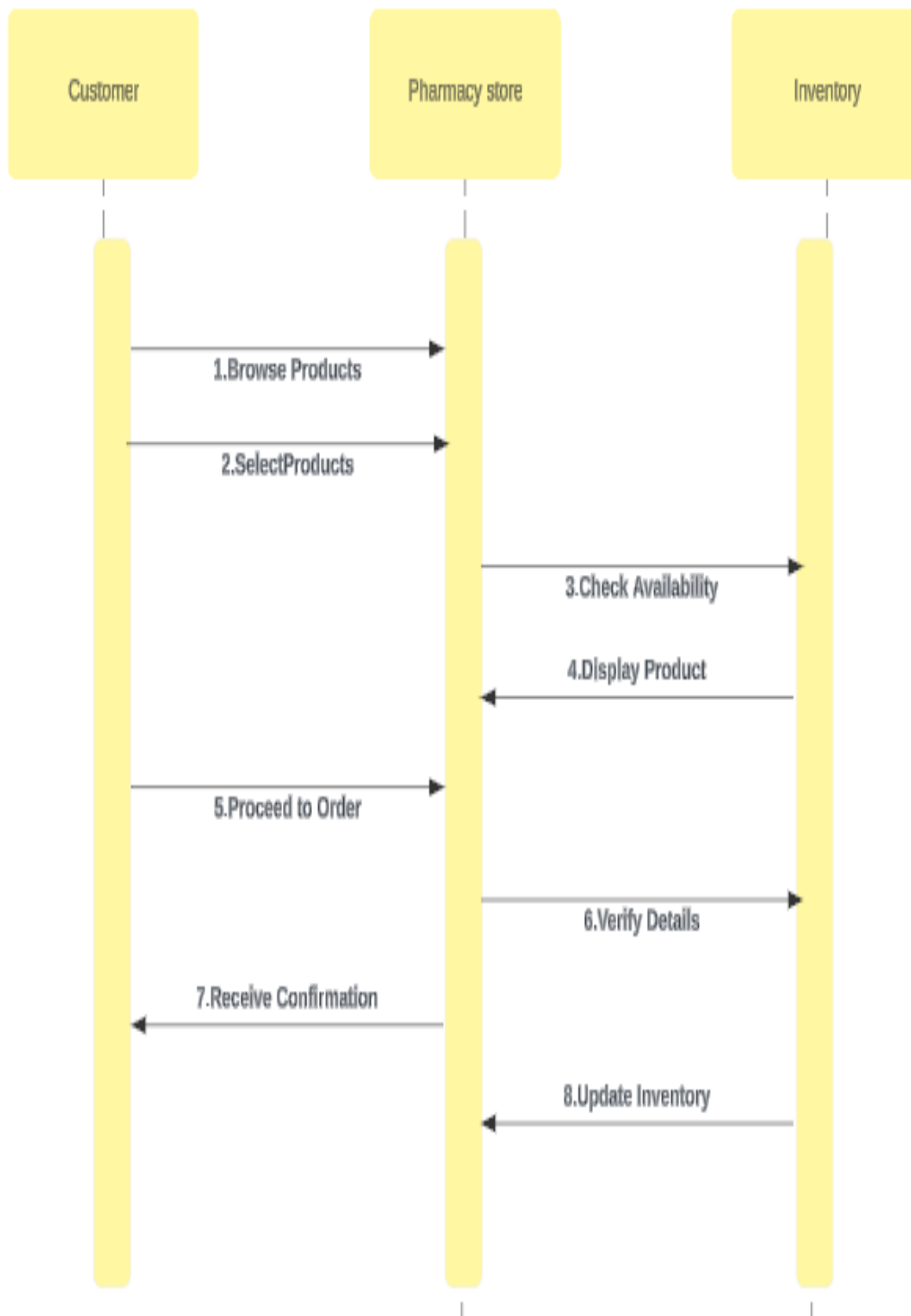
**Activation:** Activations, modelled as rectangular boxes on the lifeline, indicate when the object is performing an action.



**Message:** Messages, modeled as horizontal arrows between Activations, indicate the communication between objects.



## Sequence Diagram



#### 4.5.4 Collaboration Diagram

Like the other Behavioural diagrams, Collaboration diagrams model the interactions between objects. This type of diagram is a cross between an object diagram and a sequence diagram. Unlike the Sequence diagram, which models the interaction in a column and row type format, the Collaboration diagram uses the free- form arrangement of objects as found in an Object diagram. This makes it easier to see all interactions involving a particular object.

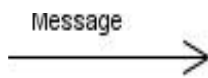
**Object** Objects are instances of classes, and are arranged horizontally. The pictorial representation for an Object is a class (a rectangle) with the name prefixed by the object name (optional) and a semi-colon.



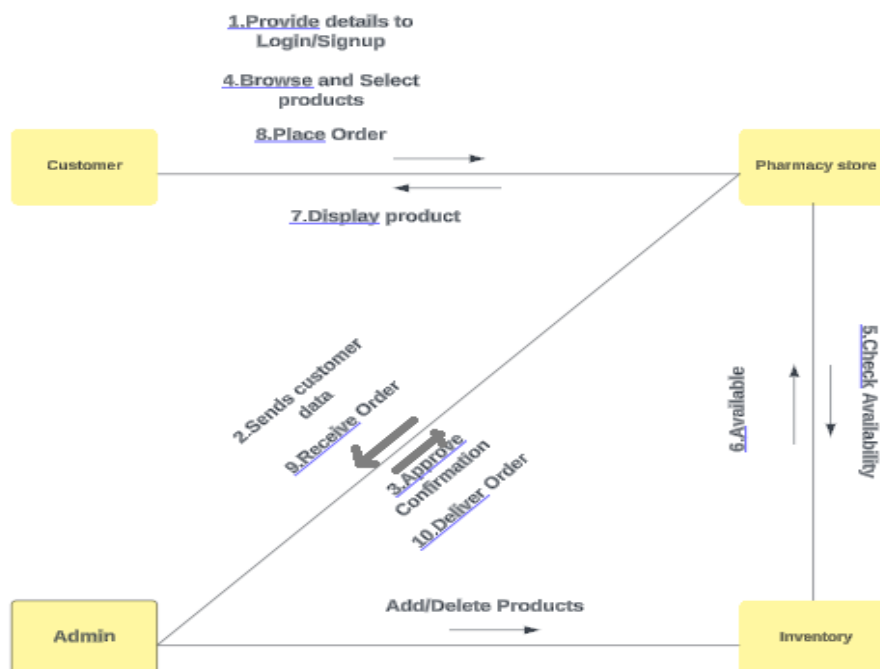
**Activation:** Activations, modelled as rectangular boxes on the lifeline, indicate when the object is performing an action.



**Message:** Messages, modeled as horizontal arrows between Activations, indicate the communication between objects.



#### Collaboration Diagram



#### 4.5.5 Activity Diagram

A use case's or an object's behaviour's activities normally take place in a particular order. A simplified view of what happens throughout a process or an operation is provided by an activity diagram. The processing within each activity moves to completion before being automatically transmitted to the following, which is represented by a rounded rectangle. The changeover between two activities is represented by an arrow. A system is described in terms of activities in an activity diagram.

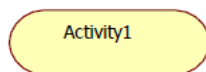
Activities are the state that denotes the performance of a sequence of actions. These are comparable to dataflow and flow chart diagrams.

**Initial state:** which state is starting the process?



#### Action State:

An action state denotes the performance of an atomic action, commonly the calling of an operation. A simple state containing an entry action and one exit transition is referred to as an action state. The exit transition is brought about by the implicit event of the entry action's completion.



#### Transition:

A directed relationship between the vertex of a source state and the vertex of a target state is referred to as a transition. It might be a component of a compound transition that changes the static machine's configuration from one static state to another, summarising how the static machine responded to a specific event instance.



#### Final state:

A final state represents the last or "final" state of the enclosing composite state. There may be more than one final state at any level signifying that the composite state can end in different ways or conditions.

When a final state is reached and there are no other enclosing states it means that the entire state machine has completed its transitions and no more transitions can occur.

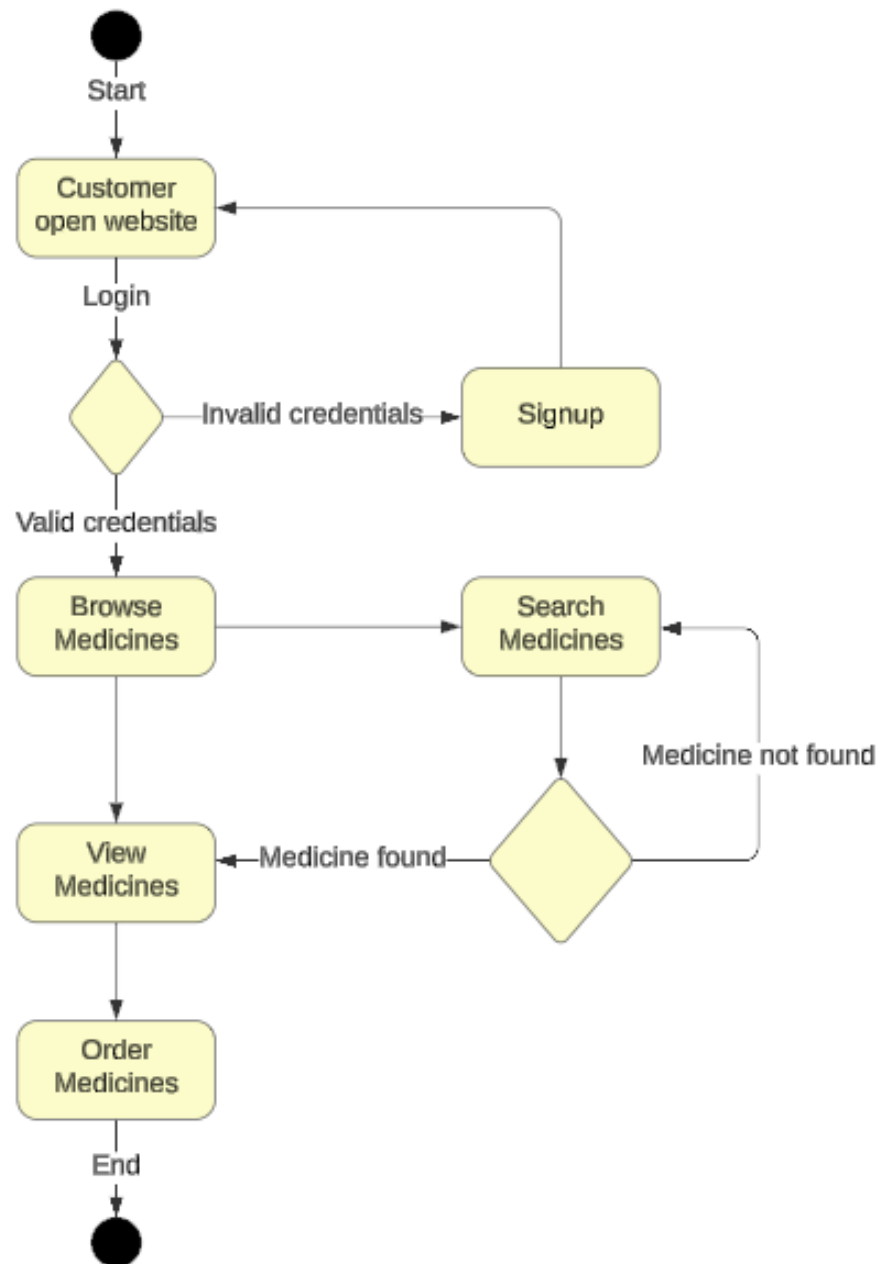


#### Decision:

A state diagram (and by derivation an activity diagram) expresses decision when guard conditions are used to indicate different possible transitions that depend on Boolean conditions are used to indicate different possible transitions that depend on Boolean conditions of the owning object.



# Activity Diagram



## **CHAPTER-5**

### **TECHNOLOGY DESCRIPTION**

## 5. TECHNOLOGY DESCRIPTION

### 5.1 HTML Introduction:

Hypertext Mark-up Language (HTML), the languages of the world wide web (WWW), allows users to produce web pages that included text, graphics and pointer to other web pages (Hyperlinks). HTML is not a programming language but it is an applications of ISO standard 8879, SGML (Standard Generalized Mark -up Language), but specialized to hypertext and adapted to the web. The idea behind Hypertext one point to another point. We can navigate through the information based on our interest and preference. A mark-up language is simply a series of enclosed within the elements should be displayed.

Hyperlinks are underlined or emphasized words that load to other documents or some portions of the same document. HTML can be used to display any type of document on the host computer, which can be geographically at a different location. It is a versatile language and can be used on any platform or desktop. HTML provides tags (special code) to make the document look attractive. HTML provides or not case –sensitive. Using graphics, fonts, different sizes, colour, etc. can enhance the presentation of the document. Anything that is not a tag is part of the document itself.

#### 5.1.1 Basic HTML Tags:

< ! ----> Specific comments.

<A>....</A> Creates Hypertext links.

<B>....</B> Creates Hypertext links.

<Big>....</Big> Formatted text in large-font.

<Body>.....</Body> contains all tags and text in the HTML-document.

<Center>.....</Center> Creates Text.

<DD>....</DD> Definition of a term.

<TABLE>...</TABLE> Creates table.

<Td>.....</Td> indicates table data in a table.

<Tr>.....</Tr> designates a table row.

<Th>.....</Th> creates a heading in a table.

### ADVANTAGES



- A HTML document is a small and hence easy to send over the net. It is a small because it does not include formatted information.
- HTML is a platform independent.

## 5.2 CSS Introduction

**CSS** is used to control the style of a web document in a simple and easy way. **CSS** is the acronym for "**Cascading Style Sheet**". This tutorial covers both the versions CSS1, CSS2 and CSS3, and gives a complete understanding of CSS, starting from its basics to advanced concepts. **Cascading Style Sheets**, fondly referred to as **CSS**, is a simple design language intended to simplify the process of making web pages presentable. **CSS** is a **MUST** for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain. I will list down some of the key advantages of learning CSS:

## 5.3 PYTHON Introduction

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics developed by Guido van Rossum. It was originally released in 1991. Designed to be easy as well as fun, the name "Python" is a nod to the British comedy group Monty Python. Python has a reputation as a beginner-friendly language, replacing Java as the most widely used introductory language because it handles much of the complexity for the user, allowing beginners to focus on fully grasping programming concepts rather than minute details.

Python is used for server-side web development, software development, mathematics, and system scripting, and is popular for Rapid Application Development and as a scripting or glue language to tie existing components because of its high-level, built-in data structures, dynamic typing, and dynamic binding. Program maintenance costs are reduced with Python due to the easily learned syntax and emphasis on readability. Additionally, Python's support of modules and packages facilitates modular programs and reuse of code. Python is an open source community language, so numerous independent programmers are continually building libraries and functionality for it.

## 5.4 Modules Description

A medical store management system typically consists of several interconnected modules to facilitate various functionalities. Here are the key modules commonly found in such systems:

1. **Inventory Management:** This module handles the tracking of medicines and medical supplies, including stock levels, expiration dates, and supplier information.
2. **Order Management:** Allows users to place orders, track order status, and manage order processing workflows from order placement to fulfilment.
3. **Customer Management:** Manages customer profiles, including personal information, purchase history, preferences, and loyalty programs.
4. **Reporting and Analytics:** Provides insights into sales performance, inventory turnover, customer behaviour, and other key metrics through customizable reports and analytics tools.

5. **Security and Access Control:** Implements security measures to protect sensitive data, enforce access control policies, and ensure compliance with regulatory requirements.
6. **User Administration:** Allows administrators to manage user accounts, permissions, roles, and other administrative tasks related to system access and usage.
7. **Customer Support and Communication:** Provides channels for customer support, including live chat, email support, and helpdesk ticketing systems, as well as communication tools for sending notifications and updates to customers.

These modules work together to streamline operations, enhance efficiency, and provide a seamless shopping experience for customers in a medical store management system.

**CHAPTER-6**  
**SAMPLE CODE**

## 6. SAMPLE CODE

### 1.BASE.HTML (Contains details about the apps and authorizations.)

```
{ % load static % }
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- Bootstrap CSS -->
    { % comment % } <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOMLASjC"
crossorigin="anonymous"> { % endcomment % }

    <link rel="stylesheet" href="{ % static 'css/style.css' % }">

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.1.1/css/all.min.css" integrity="sha512-
KfkfwYDsLkIlwQp6LFnl8zNdLGxu9YAA1QvwINks4PhcElQSwqcyVLLD9aMhXd13uQjo
XtEKNosOWaZqXgel0g==" crossorigin="anonymous" referrerpolicy="no-referrer" />

  <title>

    { % block title % }

    { % endblock title % }
  </title>

</head>
<body>

  <nav class="navbar navbar-expand-lg navbar-dark bg-dark p-4">
    <div class="container-fluid">
```

```
<button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
```

```
<span class="navbar-toggler-icon"></span>
```

```
</button>
```

```
<div class="collapse navbar-collapse" id="navbarSupportedContent">
```

```
<ul class="navbar-nav me-auto mb-2 mb-lg-0">
```

```
<li class="nav-item">
```

```
<a class="nav-link active" aria-current="page" href="/">Home</a>
```

```
</li>
```

```
<li class="nav-item">
```

```
<a class="nav-link active" aria-current="page" href="/medicines">Medicines</a>
```

```
</li>
```

```
<li class="nav-item">
```

```
<a class="nav-link active" aria-current="page" href="/products">Products</a>
```

```
</li>
```

```
<li class="nav-item">
```

```
<a class="nav-link active" aria-current="page" href="/orders">Order</a>
```

```
</li>
```

```
<li class="nav-item">
```

```
<a class="nav-link active" href="contact">Contact</a>
```

```
</li>
```

```
<li class="nav-item">
```

```
<a class="nav-link active" href="about">About</a>
```

```
</li>
```

```
{ % if request.user.is_authenticated % }
```

```
<li class="nav-item dropdown">
```

```

    <a class="nav-link dropdown-toggle" href="signup" id="navbarDropdown" role="button"
data-bs-toggle="dropdown" aria-expanded="false">
    Welcome {{user.email}}
</a>
<ul class="dropdown-menu" aria-labelledby="navbarDropdown">

    <li><a class="dropdown-item" href="logout">Logout</a></li>
</ul>
</li>

{% else %}

    <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle" href="signup" id="navbarDropdown"
role="button" data-bs-toggle="dropdown" aria-expanded="false">
        New User?
        </a>
        <ul class="dropdown-menu" aria-labelledby="navbarDropdown">
            <li><a class="dropdown-item" href="signup">Sign Up</a></li>
            <li><a class="dropdown-item" href="login">Login</a></li>
        </ul>
    </li>

    {% endif %}

</ul>
<form action="/search" method="get" class="d-flex">
    <input class="form-control me-2" type="search" placeholder="Search" aria-
label="Search" name="getdata" required>
    <button class="btn btn-outline-success" type="submit" >Search</button>
</form>
</div>
</div>
</nav>

```

```
{% block body %}
```

```
{% endblock body %}
```

```
<br><br>
```

```
<!-- Optional JavaScript; choose one of the two! -->
```

```
<!-- Option 1: Bootstrap Bundle with Popper -->
```

```

                                {%           comment           % }           <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script> {% endcomment % }
```

```
<!-- Option 2: Separate Popper and Bootstrap JS -->
```

```
<!--
```

```

    <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.2/dist/umd/popper.min.js"
integrity="sha384-
IQsoLX15PILFhosVNubq5LC7Qb9DXgDA9i+tQ8Zj3iwWAwPtgFTxbJ8NT4GN1R8p"
crossorigin="anonymous"></script>
```

```

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.min.js"
integrity="sha384-
cVKIPhGWIC2Al4u+LWgxfKTRIfu0JTxR+EQDz/bglldoEyl4H0zUF0QKbrJ0EcQF"
crossorigin="anonymous"></script>
```

```
-->
```

```
<script src="static/js/bootstrap.js"></script>
```

```
</body>
```

```
</html>
```

**2.CONTACT.HTML** (Helpful for user to provide the feedback about the website and orders.)

```
{% extends "base.html" %}
{% block title %} Contact {% endblock title %}
{% block body %}
<Center><h1 class="display-4">CONTACT US</h1></Center>

<div class="container mt-2">

    <div class="row">
        <div class="col-md-3"></div>

        <div class="col-md-6 bg-light p-5">

            {% include "messages.html" %}

            <form action="/contact" method="post">{% csrf_token %}

                <label for="name">Name</label>

                <div class="form-group">
                    <input type="text" id="name" class="form-control mb-3" placeholder="Enter Your
Name" name="name" required>
                </div>

                <label for="email">Email</label>

                <div class="form-group">
                    <input type="email" id="email" class="form-control mb-3" placeholder="Enter Your
Email Address" name="email" value="{{ user.username }}" required>
                </div>

                <label for="num">Phone Number</label>
```



```

<div class="form-group">
  <input type="number" id="num" class="form-control mb-3" placeholder="Enter Your
Mobile no." name="num" required>
</div>

<label for="desc">How can i help?</label>

<div class="form-group">
  <textarea type="text" id="desc" class="form-control mb-3" name="desc"
required></textarea>
</div>

<div class="d-grid gap-2">
  <button class="btn btn-dark" type="submit">Submit</button>

</div>

</form>

</div>

<div class="col-md-3"></div>
</div>
</div>
<br>
<br>

{ % endblock body % }

```

### 3.HOME.HTML (User interface of different categories Home page.)

```
{% extends "base.html" %}
{% block title %} Home {% endblock title %}
{% block body %}
{% load static %}
<div id="carouselExampleCaptions" class="carousel slide" data-bs-ride="carousel">
  <div class="carousel-indicators">
    <button type="button" data-bs-target="#carouselExampleCaptions" data-bs-slide-to="0"
class="active" aria-current="true" aria-label="Slide 1"></button>
    <button type="button" data-bs-target="#carouselExampleCaptions" data-bs-slide-to="1"
aria-label="Slide 2"></button>
    <button type="button" data-bs-target="#carouselExampleCaptions" data-bs-slide-to="2"
aria-label="Slide 3"></button>
  </div>
  <div class="carousel-inner">
    <div class="carousel-item active">
      
      <div class="carousel-caption d-none d-md-block">
        <h5 style="color: black;"><b>WELCOME TO</b></h5>
        <p style="color: black;"><b>MEDICAL STORE MANAGEMENT SYSTEM</b></p>
      </div>
    </div>
    <div class="carousel-item">
      
      <div class="carousel-caption d-none d-md-block">
        <h5 style="color: black;"><b>WELCOME TO</b></h5>
        <p style="color: black;"><b>MEDICAL STORE MANAGEMENT
SYSTEM</b></p>
      </div>
    </div>
    <div class="carousel-item">
      
      <div class="carousel-caption d-none d-md-block">
        <h5 style="color: black;"><b>WELCOME TO</b></h5>
        <p style="color: black;"><b>MEDICAL STORE MANAGEMENT SYSTEM</b></p>
      </div>
    </div>
  </div>
</div>
{% endblock body %}
```

**4.LOGIN.HTML** (Existing user can directly login to account by giving login credentials and clicking on login button.)

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <!-- Bootstrap CSS -->
    <link
      href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
      rel="stylesheet"
      integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
      crossorigin="anonymous"
    />
    <title>Login</title>
  </head>
  <body>
    <div class="container mt-5">
      <div class="row">
        <div class="col-md-4"></div>
        <div class="col-md-4 bg-light p-5">
          <hr />
          <h5 class="display-4 text-center bg-dark text-light">Login</h5>
          { % include "messages.html" % }
          <hr />
          <form action="/login" method="post">{ % csrf_token % }
            <label for="email">Email</label>
            <div class="form-group">
              <input type="email" id="email" class="form-control mb-3" placeholder="Enter Your
Email Address" name="email" required>
            </div>
            <label for="pass1">Password</label>
            <div class="form-group">
              <input type="password" id="pass1" class="form-control mb-3" placeholder="Enter
Your Password" name="pass1" required>
            </div>
          </form>
        </div>
      </div>
    </div>
  </body>
</html>
```

```

</div>

<div class="d-grid gap-2">
  <button class="btn btn-dark" type="submit">Login</button>

</div>
<hr>
</form>
<br>
<br>
</div>
<div class="col-md-4"></div>
</div>
</div>
<!-- Optional JavaScript; choose one of the two! -->

<!-- Option 1: Bootstrap Bundle with Popper -->
<script
  src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
  integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
  crossorigin="anonymous"
></script>
<!-- Option 2: Separate Popper and Bootstrap JS -->
<!--
  <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.2/dist/umd/popper.min.js"
  integrity="sha384-
IQsoLX15PILFhosVNubq5LC7Qb9DXgDA9i+tQ8Zj3iwWAwPtgFTxbJ8NT4GN1R8p"
  crossorigin="anonymous"></script>
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.min.js"
  integrity="sha384-
cVKIPhGWIC2A14u+LWgxfKTRICfu0JTxR+EQDz/bglDoEyl4H0zUF0QKbrJ0EcQF"
  crossorigin="anonymous"></script>
-->
</body>
</html>

```

**5.ORDERS.HTML**(Contains details of every order of an user in our website.)

```

{% extends "base.html" %}
{% block title %}
Order-Products/Medicines
{% endblock title %}
{% block body %}

<div class="row">

    <div class="col-md-4">

<div class="container mt-2">
    <h2 class="display-4"></h2>
    {% include "messages.html" %}
    <div class="card bg-light p-4">
        <div class="card-body">
            <form action="orders" method="post">
                {% csrf_token %}

                <div class="form-group">
                    <label for="name">Full Name</label>
                    <input
                        type="text"
                        id="name"
                        name="name"
                        class="form-control mb-3"
                        placeholder="Enter Your Name"
                        value="{{ user.first_name }} {{ user.last_name }}"
                        required
                    />
                </div>
                <div class="form-group">
                    <label for="name">Email</label>
                    <input
                        type="text"
                        id="email"

```

```

name="email"
class="form-control mb-3"
value="{{user.username}}"
required
/>
</div>

<div class="form-group">
  <label for="items">Select Medicines/Products</label>
  <select id="items" class="form-control" name="items">
    <option selected>Choose...</option>

    {% for i in mymed %}
    <option value="{{i.medicine_name}}">{{i.medicine_name}}</option>
    {% endfor %}
    {% for j in myprod %}
    <option value="{{j.prod_name}}">{{j.prod_name}}</option>
    {% endfor %}
  </select>
</div>

<div class="form-group">
  <label for="quantity">Quantity?</label>
  <select id="quantity" class="form-control" name="quantity">
    <option selected>1</option>
    <option value="2">2</option>
    <option value="3">3</option>
    <option value="4">4</option>

  </select>
</div>

<div class="form-group">
  <label for="num">Phone Number</label>
  <input

```

```

        type="tel"
        id="num"
        name="num"
        class="form-control mb-3"
        pattern="[0-9]{3}[0-9]{3}[0-9]{4}"

        required
    />
</div>

<div class="form-group">
    <label for="address">Delivery Address</label>
    <textarea
        type="text"
        id="address"
        name="address"
        class="form-control mb-3"
        required
    /></textarea>
</div>
<div class="form-group form-check">
    <input type="checkbox" class="form-check-input" id="exampleCheck1">
    <label class="form-check-label mb-2" for="exampleCheck1">Cash on delivery</label>
</div>
<div class="d-grid gap-2">
    <button class="btn btn-dark" type="submit">Place Order</button>
</div>
</form>
</div>
</div>
</div>
<div class="col-md-8">
    <div class="container mt-2">
        <h2 class="display-4">Orders</h2>
        <table class="table table-danger table-striped">
            <thead>
                <tr>

```

```

        <th scope="col">Order ID</th>
        <th scope="col">Name</th>
        <th scope="col">Products</th>
        <th scope="col">Price</th>
        <th scope="col">Cancel</th>
        <th scope="col">Status</th>
    </tr>
</thead>
<tbody>
    {% for item in items %}
    <tr>
        <th scope="row">{{ item.id }}</th>
        <td>{{ item.name }}</td>
        <td>{{ item.items }}</td>
        <td>{{ item.price }}</td>
        {% if item.delivery %}
            <td><button type="button" class="btn-success p-2" href="/orders/{{ item.id }}"
disabled> <i class="fa-solid fa-trash-can"></i></button></td>
        {% else %}
            <td><a type="button" class="btn-danger p-2" href="/orders/{{ item.id }}"> <i class="fa-
solid fa-trash-can"></i></a></td>
        {% endif %}
        {% if item.delivery %} <td><i class="fa-solid fa-circle-check"></i> Delivered</td>
        {% else %}
            <td> <i class="fa-solid fa-truck"></i> Not Delivered</td>
        {% endif %}
    </tr>
    {% endfor %}
</tbody>
</table>
</div>
</div>
</div>
{% endblock body %}

```



**6.SEARCH.HTML**(Contains details of every medicine or product in our website.)

```

{% extends "base.html" %}
{% block title %} About {% endblock title %}
{% block body %}<br>
<h2 class="text-center bg-dark text-success">Your Search results.....</h2>
<br />{% if allItems|length < 1%}
<h1>Your Search did not match <span class="badge bg-secondary">Try Again</span></h1>
<ul><li>Make Sure You have Spelled Correct</li>
</ul>{% endif %}
<br /><div class="container mt-2">
  <div class="row">
    {% for i in Med %}
      <div class="col-md-4 bg-light p-5">
        <div class="card" style="width: 20rem;">
          
          <div class="card-body">
            <h6 class="card-title display-4 text-primary">{{ i.medicine_name }}</h6>
            <p class="text-success bg-light">Price : {{ i.medicine_price }} Rs</p>
            <p class="card-text">{{ i.medicine_descripton }}</p>
            <p class="text-danger"><b>Expiry Date : {{ i.medicine_exp }}</b></p>
            <a href="/orders" class="btn btn-outline-success btn-lg">Order Now</a>
          </div>
        </div>
      </div>
    {% endfor %}
  </div>
  <div class="row">
    {% for i in Prod %}
      <div class="col-md-4 bg-light p-5">
        <div class="card" style="width: 20rem;">
          
          <div class="card-body">
            <h6 class="card-title display-4 text-primary">{{ i.prod_name }}</h6>
            <p class="text-success bg-light">Price : {{ i.prod_price }} Rs</p>
            <p class="card-text">{{ i.prod_descripton }}</p>
            <p class="text-danger"><b>Expiry Date : {{ i.prod_exp }}</b></p>
            <a href="/orders" class="btn btn-outline-success btn-lg">Order Now</a>
          </div>
        </div>
      </div>
    {% endfor %}
  </div>
</div>
{% endblock body %}

```

**7.SIGNUP.HTML**(New user can signup to account by entering valid credentials and clicking on signup button.)

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />

    <!-- Bootstrap CSS -->
    <link
      href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
      rel="stylesheet"
      integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTWfSpd3yD65VohhpuuCOMLASjC"
      crossorigin="anonymous"
    />

    <title>Signup</title>
  </head>
  <body>
    <div class="container mt-5">
      <div class="row">
        <div class="col-md-4"></div>

        <div class="col-md-4 bg-light p-5">
          <hr />
          <h5 class="display-4 text-center bg-dark text-light">SignUp</h5>

          <hr />
          { % include "messages.html" % }

          <form action="/signup" method="post">
            { % csrf_token % }

            <label for="username">UserName</label>
```

```
<div class="form-group">
  <input
    type="text"
    id="username"
    name="username"
    class="form-control mb-3"
    placeholder="Spaces Not Allowed"
    required
  />
</div>
```

```
<label for="email">Email</label>
```

```
<div class="form-group">
  <input
    type="email"
    id="email"
    name="email"
    class="form-control mb-3"
    placeholder="Enter Your Email Address"
    required
  />
</div>
```

```
<label for="fname">First Name</label>
```

```
<div class="form-group">
  <input
    type="text"
    id="fname"
    name="fname"
    class="form-control mb-3"
    placeholder="Enter Your First Name"
    required
  />
</div>
```

```
<label for="lname">Last Name</label>
```

```
<div class="form-group">
  <input
    type="text"
    id="lname"
    name="lname"
    class="form-control mb-3"
    placeholder="Enter Your Last Name"
    required
  />
</div>
```

```
<label for="pass1">Password</label>
```

```
<div class="form-group">
  <input
    type="password"
    id="pass1"
    name="pass1"
    class="form-control mb-3"
    placeholder="Enter Your Password"
    required
  />
</div>
```

```
<label for="pass2">Confirm Password</label>
```

```
<div class="form-group">
  <input
    type="password"
    id="pass2"
    name="pass2"
    class="form-control mb-3"
    placeholder="Confirm Your Password"
    required
  />
</div>
```

```
<div class="d-grid gap-2">
```

```

        <button class="btn btn-dark" type="submit">Signup</button>
    </div>

    <hr />
    Already User? <a href="/login">Login</a>
</form>
<br />
<hr>
    Back to <a href="/">Home</a>
<br />
</div>

<div class="col-md-4"></div>
</div>
</div>

<!-- Optional JavaScript; choose one of the two! -->

<!-- Option 1: Bootstrap Bundle with Popper -->
<script
    src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
    integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
    crossorigin="anonymous"
></script>

<!-- Option 2: Separate Popper and Bootstrap JS -->
<!--
    <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.2/dist/umd/popper.min.js"
    integrity="sha384-
IQsoLX15PILFhosVNubq5LC7Qb9DXgDA9i+tQ8Zj3iwWAwPtgFTxbJ8NT4GN1R8p"
    crossorigin="anonymous"></script>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.min.js"
    integrity="sha384-
cVKIPhGWIC2A14u+LWgxfKTRlcfu0JTtR+EQDz/bglldoEyl4H0zUF0QKbrJ0EcQF"
    crossorigin="anonymous"></script>
-->
</body>
</html>

```

## **CHAPTER-7**

### **TESTING**

## **7.TESTING**

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are corrected, the goal will be successfully achieved. Inadequate testing non testing leads to errors that may not appear until months later.

The testing of this project ensures that the data received by the user is accurate. The project gives details of different books available in the book stall according to the customer wish. This is ensured in the testing. Testing should systematically uncover different classes of errors in a minimum amount of time with a minimum amount of efforts. Two classes of inputs are provided to test the process.

A software configuration that includes a software requirement specification, a design specification and source code. A software configuration that includes a test plan and procedure, any testing tool and test cases and their expected results.

Testing is divided into several distinct operations:

### **TYPES OF TESTING**

#### **UNIT TESTING**

In computer programming, a unit test is a procedure used to validate that a particular module of source code is working properly. The procedure is to write test cases for all functions and methods so that whenever a change causes a regression, it can quickly be identified and fixed. Ideally, each test case is separate from others.

This project has so many modules like new book insertion, deletion, update of the book details. All the individual modules are tested and validated for checking whether it gives the desired output. After validating these modules, we can say that all the modules of these system are working perfectly and giving the desired output required by the administrator.

#### **WHITE BOX TESTING**

Integration testing (sometimes called integration and testing and abbreviated I&T) is the phase of software testing in which individual software modules are combined and tested as a group. It follows unit testing and precedes system testing. Integration testing takes as its input values that have been unit tested, groups them in larger aggregates, applies tests defined in and integrated test plan to those aggregates and delivers as its output the integrated system ready for system testing the purpose of integration testing is to verify functional performance and reliability requirements placed in major design items.

Form validation involves checking all the form constraints like arithmetic, syntax, logical errors. Database validation involves checking constraints like primary key, foreign key, and all the database validations avoiding data redundancy. All the modules of the system are tested and now the modules are combined together and the integrated module is tested and validated for checking whether the combined modules are working perfectly or not.

### **BASIC PATH TESTING**

Validation testing is a concern which overlaps with integration testing ensuring that the application fulfils its specification, is a major criterion of integration testing. Validation, testing also overlaps to a large extent with system testing, where the application is tested with respect to its typical working environment.

Consequently, for many processes no clear division between validation and system testing can be made validation testing.

Specific tests can be performed in either or both stages include the following:

### **DATA FLOW TESTING**

where the software is deliberately interrupted in a number of ways, for example taking its hard disk off line or even turning the computer off, to ensure that the appropriate techniques for restoring any lost data will function.

It is a system that forces the software to fail in a variety of ways and verifies that the recovery is properly performed. where unauthorized attempts to operate the software or part of it, are attempted. It might also include attempts to obtain access the data or harm the software installations or even the system software.

With all types of security it is recognized that someone sufficiently determined will be able to obtain unauthorized access and the best that can be achieved is to make this process as difficult as possible. It attempts to verify that protection mechanisms built into a system will in fact protect it from improper penetration. The system's security must of course be tested from its vulnerability from frontal attack.

### **LOOP TESTING**

where abnormal demands are made upon the software by increasing the rate at which it is asked to produce information. More complex tests may attempt to create very large data sets or cause the software to make excessive demands on the operating system. Stress tools are designed to confront programs with abnormal situations. Stress testing executes a system in a manner that demands resources in abnormal quantity and volume.

### **PERFORMANCE TESTING**

Performance requirements, if any, are checked. These may include the size of the software when installed the amount of main memory and/or secondary storage it requires and the demands made of the operating system when running within normal limits of the response Time.

### **BLACK BOX TESTING**

Black box testing is done to find out the following information as shown in below:

1. Incorrect or missing functions.
2. Interface errors.
3. Errors or database access.
4. Performance error.
5. Termination error.



The mentioned testing is carried out successfully for this application according to the user's requirement specification.

## TEST DATA OUTPUT

After preparing test data, the system under study is tested using the test data. While testing the system using test data, errors are again uncovered and corrected by using above testing and corrections are also noted for future use.

## IMPLEMENTATION

Implementation is the process of converting the system design into code, testing the system and giving the user training. Implementation of a new system design is a crucial phase in the system development life cycle.

## TESTING TABLE

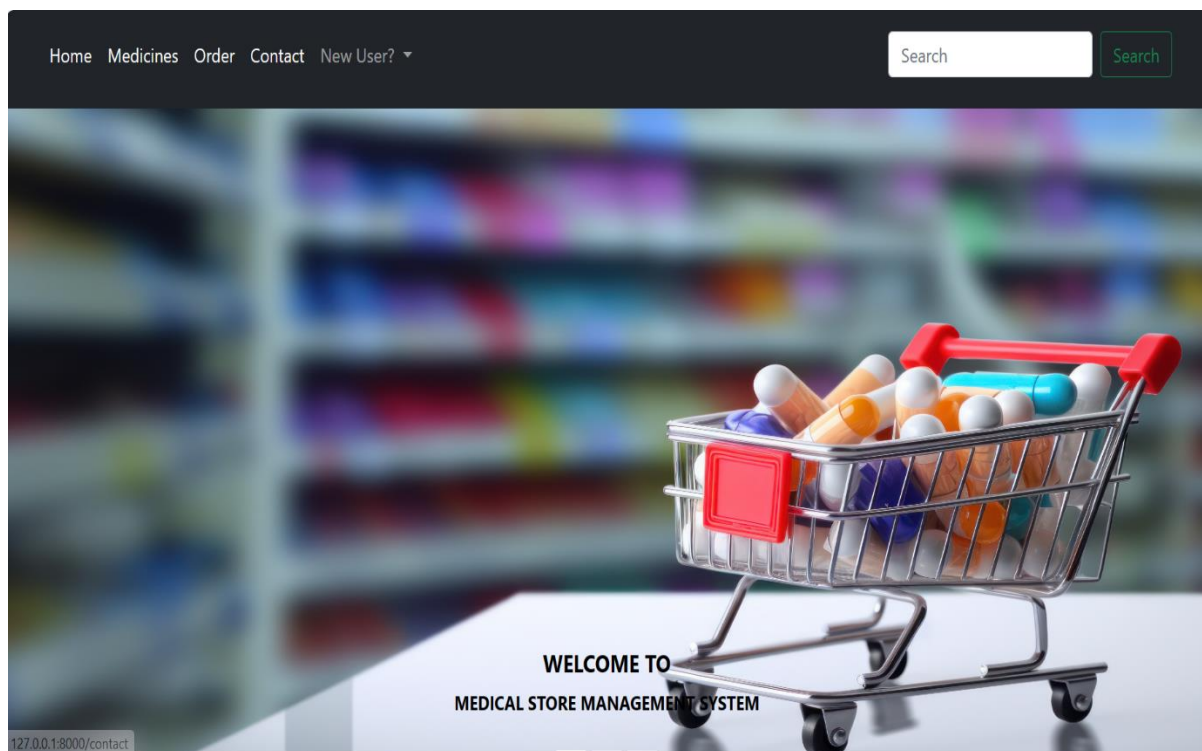
| ID | TEST CASE   | EXPECTED OUTPUT  | ACTUAL OUTPUT   | CONCLUSION                       |
|----|---|--|---|----------------------------------|
| 01 | To show main page.  | Any user can access this page and view the list of main sections provided on the platform. | User can access this page and view the list of main sections provided on the platform properly. | This function works effectively. |
| 02 | To give the users permission to view the login page and enter their details.        | Any user can view the login page and input their details based on their user type/role.    | Users can view the login page and input their details properly.                                 | This function works effectively. |
| 03 | To give the users permission to view the registration page and enter their details. | Any user can view the registration page.   | Users can view the registration page and input their details properly.                          | This function works effectively. |

## **CHAPTER-8**

### **SCREENSHOTS**

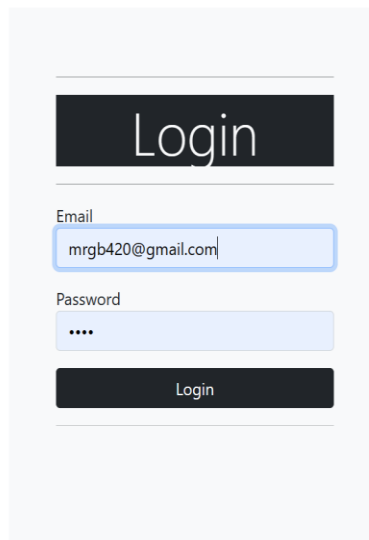
## 8.SCREENSHOTS

**Fig 8.1 HOME PAGE**



Description:- User interface of different categories Home page.

**Fig 8.2 LOGIN PAGE**



The image shows a login page interface. At the top, there is a dark rectangular button with the word "Login" in white text. Below this, there are two input fields. The first is labeled "Email" and contains the text "mrgb420@gmail.com". The second is labeled "Password" and contains four dots. Below these fields is a dark rectangular button with the word "Login" in white text.

Description:- Existing user can directly login to account by giving login credentials and clicking on login button.

**Fig 8.3 SIGN UP PAGE**

SignUp

UserName  
Gowtham

Email  
mrgb420@gmail.com

First Name  
Gowtham

Last Name  
Gowtham

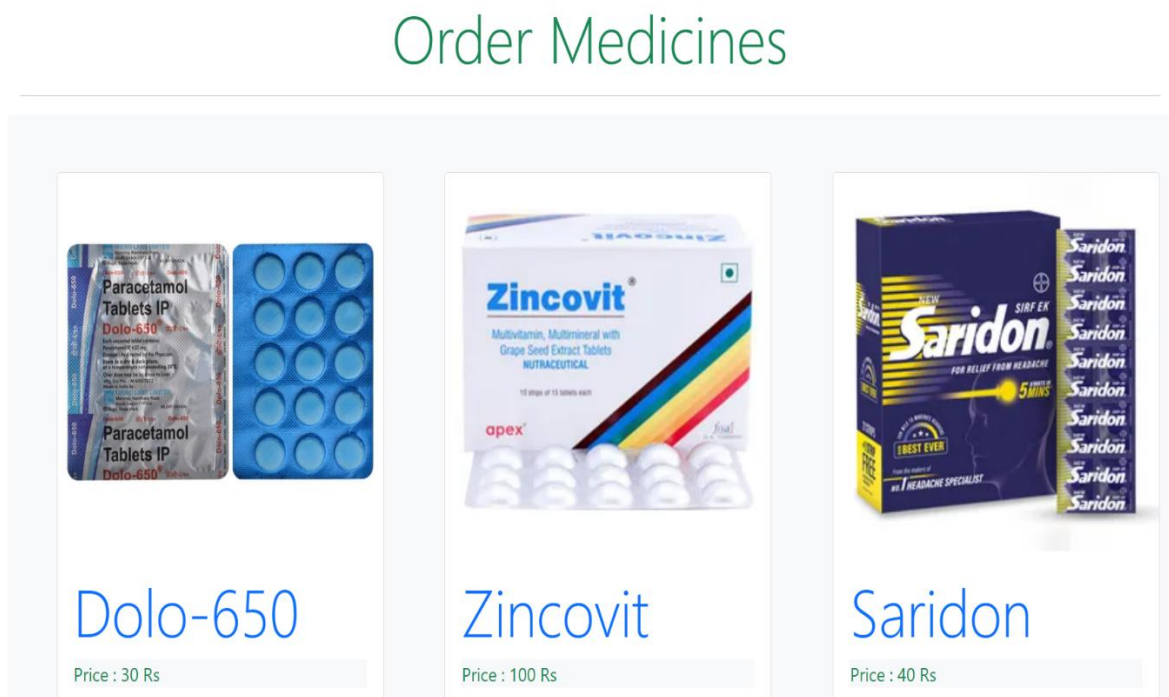
Password  
\*\*\*\*\*

Confirm Password  
\*\*\*\*\*

Signup

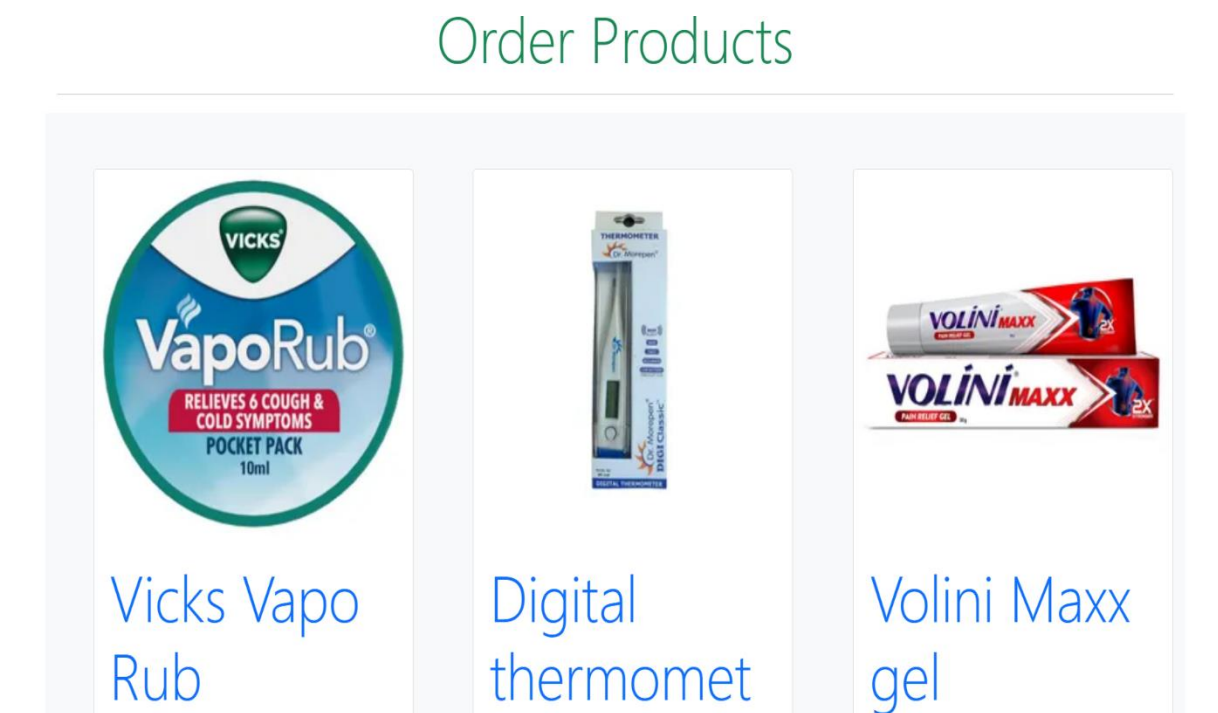
Description:- New user can signup to account by entering valid credentials and clicking on signup button.

**Fig 8.4 MEDICINES PAGE**



Description:- Displays all the Medicines available in our website.

**Fig 8.5 PRODUCTS PAGE**



Description:- Displays all the Products that are available in our website.

**Fig 8.6 ORDERS PAGE**

Full Name

Gowtham Gowtham

Email

mrgb420@gmail.com

Select Medicines/Products

Dolo-650

Quantity?

1

Phone Number

9182603675



Delivery Address

Kakinada

☒ Cash on delivery

Place Order

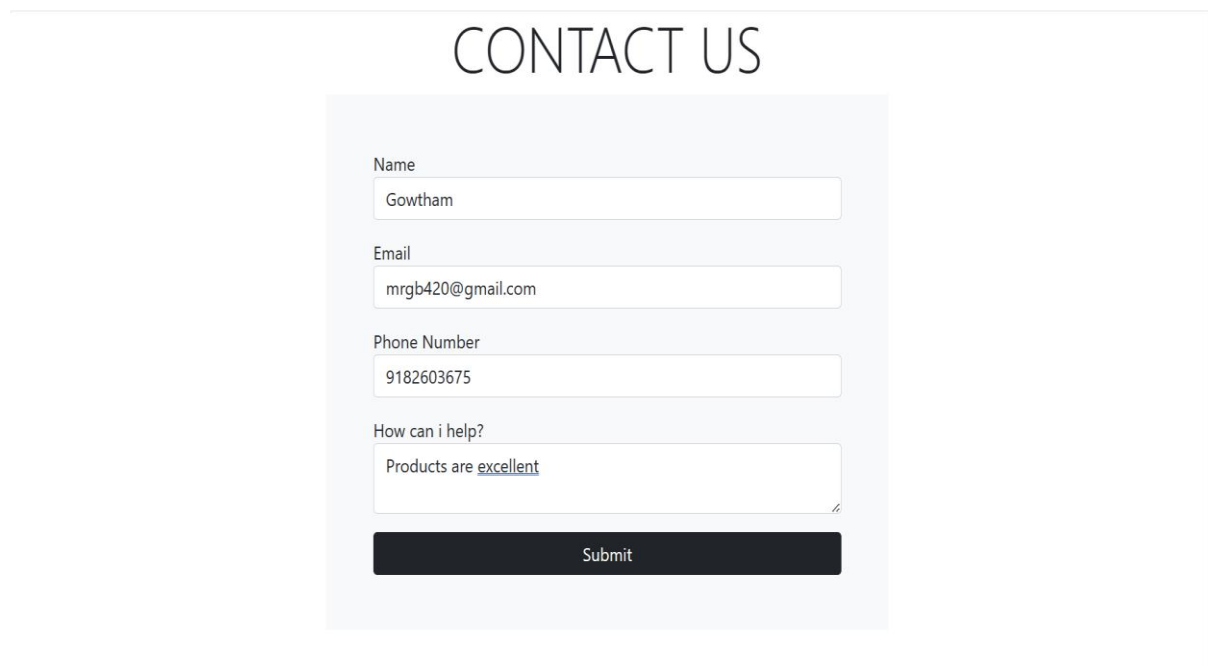
## Orders

| Order ID | Name            | Products | Price | Cancel  | Status  |
|----------|-----------------|----------|-------|---|---|
| 22       | Gowtham Gowtham | Dolo-650 | 120   |  |  Delivered |

Description:- Contains details of every order of an user in our website.



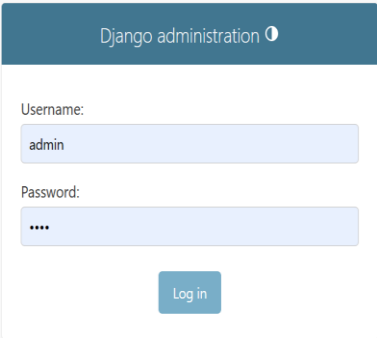
**Fig 8.7 CONTACT US PAGE**



The screenshot displays a 'CONTACT US' page. At the top, the title 'CONTACT US' is centered in a large, dark font. Below the title is a light gray rectangular box containing a contact form. The form has four input fields, each with a label above it: 'Name' (containing 'Gowtham'), 'Email' (containing 'mrgb420@gmail.com'), 'Phone Number' (containing '9182603675'), and 'How can i help?' (containing 'Products are [excellent](#)'). Below these fields is a dark gray 'Submit' button.

Description:- Helpful for user to provide the feedback about the website and orders.

**Fig 8.8 ADMIN LOGIN PAGE**



The image shows a screenshot of the Django administration login page. It features a dark blue header bar with the text "Django administration" and a small help icon. Below the header, there are two input fields: "Username:" with the value "admin" and "Password:" with masked characters "\*\*\*\*". A blue "Log in" button is positioned below the password field. The entire form is centered on a light gray background.

Description:- Administrator can login by Giving details and clicking on login button.

**Fig 8.9 SITE ADMINISTRATION PAGE**

Django administration

WELCOME: **ADMIN** [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

Site administration

| APP            |  |
|----------------|--|
| Contacts       | <a href="#">+ Add</a> <a href="#">Change</a> |
| Mediciness     | <a href="#">+ Add</a> <a href="#">Change</a> |
| My orderss     | <a href="#">+ Add</a> <a href="#">Change</a> |
| Product itemss | <a href="#">+ Add</a> <a href="#">Change</a> |

| AUTHENTICATION AND AUTHORIZATION |  |
|----------------------------------|--|
| Groups                           | <a href="#">+ Add</a> <a href="#">Change</a> |
| Users                            | <a href="#">+ Add</a> <a href="#">Change</a> |

Recent actions

**My actions**

- [MyOrders object \(28\)](#)  
My orders
- [MyOrders object \(27\)](#)  
My orders
- [MyOrders object \(26\)](#)  
My orders
- [MyOrders object \(25\)](#)  
My orders
- [MyOrders object \(24\)](#)  
My orders
- [MyOrders object \(22\)](#)  
My orders
- [MyOrders object \(23\)](#)  
My orders
- [MyOrders object \(22\)](#)  
My orders
- [+ Womens Horlicks](#)  
Product items

Description:- Contains details about the apps and authorizations.

## **CONCLUSION**

## **CONCLUSION**

In conclusion, the medical store management system has brought about significant changes in the way individual access and manage their healthcare needs. The convenience and accessibility offered by these platforms have proven to be valuable, especially in the context of modern, fast-paced lifestyles.

Medical store management system provide a wide range of benefits, including the ability to order medications from the comfort of one's home, access to a diverse selection of products, and often competitive pricing as the medical store management system landscape continues to evolve, it is essential for both consumers and healthcare providers to stay informed about best practices, regulations, and advancements in technology.

## **BIBLIOGRAPHY**

### **TEXT BOOKS :**

1. "Python Crash Course" by Eric Matthes: For a solid introduction to Python programming.
2. "Fluent Python" by Luciano Ramalho: To deepen your understanding of Python's features and best practices.

### **WEBSITES :**

<https://www.w3schools.com>

<https://www.python.org>

<https://www.upgrad.com>

### **REFERENCES :**

[1] Pharmacy Management System by Risha Patidar, Shohel Akhtar Qureshi, Anash Kuraishi, ISSN: 2582-7421.

[2] Medical Store Management System by Sathya, Gayathri, Infanta Aalvina, Kowsika, Supriya Shree, ISSN: 2582-7421.

[3] Generic Medicine Finder by Preet Patil, Swayam Jadhav, Faraaz Khan, Nilesh Vispute, ISSN: 2582-7421.

[4] Advances in Pharmacy Practice by Mr. Ram.B. Gayke, Mr. Ajit. P. More, Mr. Shakib Bagwan, ISSN: 2582-7421.