Pharma Trader Medicine Stock Management and Distributed System Using Django

A Project presented to the National University in partial fulfillment of the requirement for The degree of Bachelor of Science (Hon's) in Computer Science & Engineering

Supervised By

Poly Bhoumik
Lecturer, Department of CSE,
Daffodil Institute of IT

Submitted By

Tamim Islam
Reg. no.: **17502005055**
Session: 2017-18

Aminul Islam
Reg. no.: **17502005019**
Session: 2017-18

Department of Computer Science & Engineering
Daffodil Institute of IT, Dhaka
Under National University Bangladesh

Date of Submission:

A P P R O V A L

The project "**Pharma Trader Medicine Stock Management and Distributed System**" is submitted to the Department of Computer Science & Engineering, DIIT under the National University of Bangladesh in partial fulfillment of the requirement for the degree of Bachelor of Science (Hon's) in Computer Science and Engineering and approved as to its style and content.

……..

Examiner

……

Examiner

……

Poly Bhoumik

Project Supervisor

Lecturer, Dept. of CSE, DIIT

……..

Md. Imran Hossain

Head of Department,

Department of CSE, DIIT

DECLARATION

I declare that the project work titled "**Pharma Trader Medicine Stock Management and Distributed System**" being submitted in partial fulfillment for the degree of B.Sc. (Hon's) in Computer Science & Engineering is the original work carried out by me. It has not formed part of any other project work submitted for any degree or diploma, either in this or any other University.

Submitted By

Tamim Islam

Reg. no.: 17502005055

Session: 2017-18

Aminul Islam

Reg.no.:17502005019

Session: 2017-18

A C K N O W L E D G E M E N T

Despite my efforts, the success of this project depends largely on the encouragement and guidance of my mentors. I would like to take this opportunity to express my gratitude to the people who are playing a vital role in the successful completion of this project.

My sincere thanks to **Prof. Dr. Mohammed Shakhawat Hossain,** principal of DIIT for giving me an opportunity to undertake this project.

My sincere thanks to **Md. Imran Hossain,** department head of the Computer Science & Engineering Department, DIIT for giving me an opportunity to undertake this project.

My cordial thanks to my Project Supervisor and Batch Coordinator **Poly Bhoumik,** Lecturer, Department of Computer Science & Engineering, DIIT for his valuable guidance and support to meet the successful completion of my project.

My sincere thanks to **Saidur Rahman,** Lecturer, Department of Computer Science & Engineering, DIIT for appreciating my goal.

I express my gratitude to **Nusrat Jahan Sarker,** Lecturer, Department of Computer Science & Engineering, DIIT for appreciating my goal.

My sincere thanks to **Mizanur Rahman,** Lecturer, Department of Computer Science & Engineering, DIIT for appreciating my goal.

I express my gratitude to **Md Mushfiqur Rahman,** Lecturer, Department of Computer Science & Engineering, DIIT for appreciating my goal.

I express my gratitude to **Safrun Nesa Saira,** Lecturer, Department of Computer Science & Engineering, DIIT for appreciating my goal.

I express my gratitude to **Moumita Akter,** Lecturer, Department of Computer Science & Engineering, DIIT for appreciating my goal.

I extend my sincere thanks to my family and friends for their constant support throughout this project.

Finally, I would be grateful to the **National University, Bangladesh,** and the coordinator of the **Bachelor of Science in Computer Science and Engineering** degree program for giving me this opportunity to apply the knowledge that I have gained through the study.

ABSTRACT

The Medicine Stock Management and Distributed System is a web-based application developed using Django, which aims to provide an efficient and effective way of managing medicine stocks for pharmaceutical traders. The system offers features such as real-time stock updates, user management, reporting, notifications, and analytics. The application uses a distributed system architecture, allowing multiple users to access the system simultaneously. The user management feature ensures secure access to the system, with different levels of access permissions for different users. The real-time stock updates feature ensures that the system always reflects the current stock levels, reducing the risk of stockouts or overstocking. The reporting and analytics features provide insights into stock levels, sales, and other relevant data, enabling users to make informed decisions. The notifications feature sends alerts to users when stock levels are low or when certain thresholds are reached. This ensures that users are aware of stock levels and can take appropriate action. Overall, the Medicine Stock Management and Distributed System is a valuable tool for pharmaceutical traders, offering a comprehensive set of features for managing medicine stocks efficiently and effectively.

TABLE OF CONTENTS

Content

List of Figure

Chapter 1
Introduction

1.1 Introduction

Our project is a platform for researchers and academics to easily access and share their research papers with a global audience. Our project aims to facilitate the discovery of new research papers and scholarly works by making them easily searchable and accessible to a wider community. Our goal is to increase the visibility and impact of research papers by making them easily discoverable by other researchers, students, and professionals.

Our platform provides a space for researchers to collaborate and share their research findings with others in their field, promoting open access to research papers and making them freely available to anyone with an internet connection. We believe that an open platform for researchers to share, discuss, and critique research papers and scholarly works can facilitate the exchange of ideas and the sharing of knowledge among researchers, students, and professionals in different disciplines and fields.[1]

Our project book provides an easy-to-use and user-friendly interface for researchers and academics to access and share their research papers. We aim to improve the quality of research papers by providing a platform for peer review and feedback, and to create a space for researchers to publish their papers and get cited by others.

1.2 Objective of the project

The objective of the Pharma Trader web-based application project is to provide a robust platform for the efficient management and distribution of medicine stocks in a distributed system.

- To provide an efficient and effective way of managing medicine stocks in a distributed system, where different branches of pharmaceutical companies can access and manage their stocks from anywhere.
- To ensure real-time updates on the medicine stock levels across different locations to avoid overstocking or understocking.
- To reduce the wastage of medicines due to expiry by providing an expiry tracking system.
- To provide a user-friendly platform for the pharmaceutical companies to manage their medicine stocks with ease.
- To provide a comprehensive reporting system for different stakeholders to analyze the medicine stocks across different locations.

- To provide notifications to the users in case of low stock levels, expired medicines, or any critical updates related to medicine stocks.
- To provide analytical features where data can be analyzed to forecast the medicine demands, and hence stock can be maintained accordingly.

1.3 Business Perspective

The Pharma Trader web-based application can offer several benefits from a business perspective, some of which are:

- Improved Stock Management: The application can provide a centralized platform for the pharmaceutical companies to manage their medicine stocks across different locations, leading to improved stock management and efficient use of resources.
- Reduced Wastage: The expiry tracking system in the application can help reduce the wastage of medicines due to expiry, leading to cost savings for the pharmaceutical companies.
- Enhanced Reporting and Analytics: The application's reporting and analytical features can provide valuable insights into medicine stock levels, demand forecasts, and other critical metrics. This can help the pharmaceutical companies make informed decisions about their stock management strategy and ensure timely stock replenishment.
- Increased Efficiency and Productivity: The application can automate many manual tasks and provide real-time updates, reducing the need for manual interventions and increasing efficiency and productivity.
- Improved Customer Service: The application's real-time updates and notifications can help the pharmaceutical companies to respond quickly to customer demands and provide timely services, leading to increased customer satisfaction.[2]
- Competitive Advantage: By implementing a modern and efficient stock management system, pharmaceutical companies can gain a competitive advantage over their competitors and increase their market share

1.4 Limitations of the existing system:

As I am not aware of the details of any particular existing system, I can provide some general limitations that may exist in the current systems used for managing medicine stocks:

- Manual Data Entry: Many existing systems still rely on manual data entry, which is prone to errors and can lead to inaccuracies in stock management.
- Limited Access: In some cases, the current systems may not be easily accessible to all branches of the pharmaceutical company, leading to delays in stock updates and inefficient stock management.

- Lack of Real-time Updates: Many existing systems do not provide real-time updates on medicine stock levels, leading to overstocking or understocking of medicines.
- Limited Reporting: The existing systems may not provide comprehensive reporting features, making it difficult for stakeholders to analyze and forecast the medicine demands.
- Limited Analytics: The current systems may not provide analytical features to forecast the medicine demands based on historical data.
- Limited Integration: The existing systems may not integrate well with other systems used by the pharmaceutical companies, leading to delays in stock updates and inefficient stock management.[3]
- Lack of Security: Some existing systems may not provide adequate security measures, making it vulnerable to cyber attacks and data breaches.

1.5 Features

The features include:

- Medicine Stock Management
- Distributed System
- User Management
- Real-time Stock Updates
- Reporting
- Notifications
- Analytics

1.6 Summary

Here in this chapter, we have discussed the objectives, benefits, limitations, and features of this project. It shows an overview of the advantages of our system and disadvantages of the existing system.

Chapter 2
Background Study

2.1 Introduction to background processing

Pharma Trader Medicine Stock Management is a software application that helps pharmaceutical traders to manage their medicine stocks, sales, and purchases. The application is designed to provide real-time updates on stock availability, expiry dates, and sales trends, among other things. To improve the performance and scalability of the application, background processing and distributed systems can be used. Background processing involves running tasks asynchronously in the background, rather than in the main application thread. This approach can help to improve the responsiveness of the application by offloading long-running tasks to separate worker processes. Distributed systems, on the other hand, involve distributing the workload of an application across multiple servers, rather than relying on a single server to handle all requests. This approach can help to improve the reliability and scalability of the application by allowing it to handle more requests and users. Django is a popular web framework for building software applications in Python[8]. It provides built-in support for background processing and distributed systems through various third-party libraries and tools. For example, Celery is a popular distributed task queue that can be used with Django to run background tasks asynchronously across multiple worker processes or servers.

2.2 Challenges in background processing

Pharma traders face a number of challenges when it comes to medicine stock management and distributed systems using Django, including:

- Data Security: One of the biggest challenges in implementing a distributed system for medicine stock management is ensuring data security. Medical data is sensitive, and it is essential to ensure that the data is protected against unauthorized access, cyber attacks, and data breaches.[3]
- Data Synchronization: In a distributed system, data synchronization is essential to ensure that all systems have the same information at all times. This can be challenging when dealing with a large amount of data and multiple systems.
- System Integration: Integrating multiple systems can be challenging, especially if they have different data formats and communication protocols. Ensuring seamless integration of the various systems is critical for smooth operations.
- System Scalability: As the business grows, the system needs to be scalable to handle the increased workload. The distributed system needs to be able to handle the increased data flow and transactions without compromising on performance.
- Regulatory Compliance: The pharmaceutical industry is highly regulated, and there are many legal requirements that need to be met. It is essential to ensure that the system is compliant with all the relevant regulations and standards.
- User Adoption: The success of the system depends on user adoption. The users must be trained to use the system and provided with adequate support to ensure that they can use the system effectively.

- System Maintenance: A distributed system requires regular maintenance to ensure that it remains operational and up-to-date. This includes software updates, security patches, and database maintenance.
- Data Backup and Recovery: It is essential to have a robust backup and recovery system in place to protect against data loss due to hardware failure, software corruption, or other disasters. This is critical to ensure business continuity in case of an unexpected event.

2.3 Feasibility

A feasibility study is an assessment of the practicality of a proposed plan or project. A feasibility study analyzes the viability of a project to determine whether the project or venture is likely to succeed [6].

- **Technical Feasibility:** We can strongly say that it is technically feasible since there will not be much difficulty in getting the required resources for the development and maintaining the system as well.
- **Economic Feasibility:** The development of this application is highly economically feasible. The organization needed not to spend much money on the development of the system already available. we can attain the maximum usability of the corresponding resources. Even after the development, the organization will not be in a condition to invest more in the organization. Therefore, the system is economically feasible.[4]
- **Behavioral Feasibility:** The proposed system is also behaviorally feasible as it is very user-friendly. Extensive training of the users is not required. The users can easily learn to use the system and can adapt themselves according to the system.
- **Operational Feasibility:** It is mainly related to human organizations and political aspects. The questions to be considered are:
  1. What changes will be brought with the system?
  2. What organization structures are disturbed?
  3. What new skills will be required?

  The system is operationally feasible as it is very easy for the End users to operate it. It only needs basic information about the Windows platform.
- **Schedule feasibility:** Time evaluation is the most important consideration in the development of a project. The time schedule required for the development of this project is very important since more development time affects machine time, and costs and causes delays in the development of other systems.

2.4 Applicability of our system

- Pharmaceutical companies: to manage their supply chain, track inventory levels, and streamline their distribution process.

- Pharmacies: to manage their stock levels, automate their ordering process, and track expiration dates.
- Healthcare facilities: to manage their inventory of drugs and medical supplies, track expiration dates, and ensure compliance with regulatory requirements.
- Government agencies: to monitor the availability of essential medicines and medical supplies, track distribution patterns, and plan for emergency situations.
- Research institutions: to manage their laboratory stocks and track research materials.
- Contract manufacturing organizations (CMOs): to manage inventory levels, track production, and ensure compliance with regulatory requirements.

In addition, the system can be customized to meet the specific needs of each organization. For example, a pharmaceutical company may want to add additional features such as order tracking, invoice management, or customer management to the system.

2.5 Summary

Here in this project, we have discussed the background history of this project. It shows the challenges, feasibility, and applicability of our project.

Chapter 3
Literature Survey

3.1 Introduction to Literature Survey

A literature survey is a comprehensive review of existing literature, research, and scholarly articles on a specific topic. It is an essential part of any research project and involves searching and reviewing relevant literature, summarizing and synthesizing the key findings, and identifying gaps in the current knowledge. The purpose of a literature survey is to provide a comprehensive and critical review of the current state of knowledge on a particular topic. It helps researchers to identify the key issues, debates, and research gaps in their area of study, and to establish the theoretical framework for their research. The literature survey typically involves a systematic search of databases, journals, books, and other relevant sources to identify studies and articles related to the research question or topic. The information obtained from the literature survey is then analyzed and synthesized to provide a comprehensive review of the topic.

3.2 Feature-based approach

The literature survey revealed that medicine stock management systems with distributed systems and real-time updates have become increasingly popular in recent years. The use of distributed systems allows for real-time updates and data synchronization, enabling more efficient and effective stock management. User management and authentication are also critical components of medicine stock management systems, ensuring that only authorized personnel can access and modify inventory data.

Real-time stock updates and notifications are essential features of medicine stock management systems, allowing users to monitor inventory levels, track expiration dates, and receive alerts when stock levels are low. Reporting and analytics capabilities are also crucial for identifying trends, forecasting future demand, and making informed decisions. The literature survey also revealed that medicine stock management systems with distributed systems and real-time updates face several challenges. Data security and privacy concerns are significant issues, and it is essential to ensure that data is protected from unauthorized access and cyber threats. System scalability, integration, and maintenance are also critical factors that need to be considered[11].

3.3 Methods

The literature survey involved a systematic search of databases such as PubMed, ScienceDirect, and Google Scholar, as well as relevant books, reports, and conference proceedings. The search keywords included medicine stock management, distributed systems, user management, real-time updates, reporting, notifications, and analytics.

3.4 Better Search Option

Redis is a popular in-memory data structure store that can be used to enhance the search functionality of a website. By utilizing Redis, a website can significantly improve its search capabilities and provide better search results to its users. Some of the ways Redis can be used to improve search options in a website include:

- Caching frequently accessed search queries and results to reduce the load on the database and improve the speed of the search function.
- Utilizing Redis' powerful search engine capabilities to perform advanced searches and return highly relevant results to users.
- Incorporating autocomplete functionality using Redis' sorted sets and the ZRANGEBYLEX command to provide users with suggestions as they type their search queries.
- Implementing fuzzy search using Redis' string-matching capabilities to account for spelling mistakes and variations in search terms.
- Storing search history using Redis' lists or sets, allowing users to quickly access their previous search queries and results.
- Implementing filters using Redis' sets or hashes to allow users to narrow down search results based on specific criteria, such as date, author, or category.
- Using Redis' geospatial data capabilities to allow users to search for results based on location.

Overall, incorporating Redis into a website's search functionality can lead to a more robust and efficient search experience for users, ultimately improving user satisfaction and retention [12].

3.5 Notification

The literature survey revealed that notifications are critical components of medicine stock management systems, enabling users to monitor inventory levels, track expiration dates, and receive alerts when stock levels are low. Real-time updates and notifications are essential features of medicine stock management systems, allowing users to monitor inventory levels and receive alerts when stock levels fall below a certain threshold. The survey also revealed that notifications can be delivered through various channels such as email, text messages, push notifications, and in-app notifications. The choice of delivery channel depends on the users' preferences and the urgency of the notification. The literature survey also highlighted the importance of user interface design in delivering effective notifications. The user interface should be intuitive and user-friendly, making it easy for users to access and respond to notifications quickly. The survey also identified several challenges in implementing effective notification systems. These include data quality, integration with other systems, security and

privacy concerns, and technical issues such as network connectivity and server downtime..

3.6 Summary

Here in this chapter, we have discussed the basic features of this project. These are the building blocks of this project. Every feature that is used in our project is described here.

Chapter 4
Methodology

4.1 Introduction to Analysis Model

Analysis model operates as a link between the 'system description' and the 'design model'. In the analysis model, information, functions, and the behavior of the system are defined, and these are translated into the architecture, interface, and component-level design in the 'design modeling'.

4.2 SDLC methodology

Software Development life cycle (SDLC) is a spiritual model used in project management that defines the stages included in an information system development project, from an initial feasibility study to the maintenance of the completed application. There are different software development life cycle models specified and designed, which are followed during the software development phase. These models are also called "Software Development Process Models**."** Each process model follows a series of phases unique to its type to ensure success in the step of software development [10].

Here, are some important phases of SDLC life cycle:

- Waterfall model
- Iterative model
- Spiral model
- V-shaped model
- Agile Model
- Agile model

For this particular project, I've used the Agile Model as it suits my development strategy best.

4.3 Agile Model

The Agile model is a popular software development approach that emphasizes flexibility, collaboration, and continuous improvement throughout the software development lifecycle (SDLC). It involves breaking the development process into smaller iterations, where the development team works closely with the customer or end-user to prioritize and deliver software features in a timely and efficient manner. The Agile model is typically implemented using an iterative and incremental approach to software development. The process begins with the project stakeholders defining the requirements, followed by the development team breaking down the requirements into smaller, more manageable tasks or user stories. These user stories are then prioritized based on business value, and the development team works on the highest-priority user stories first.[9]

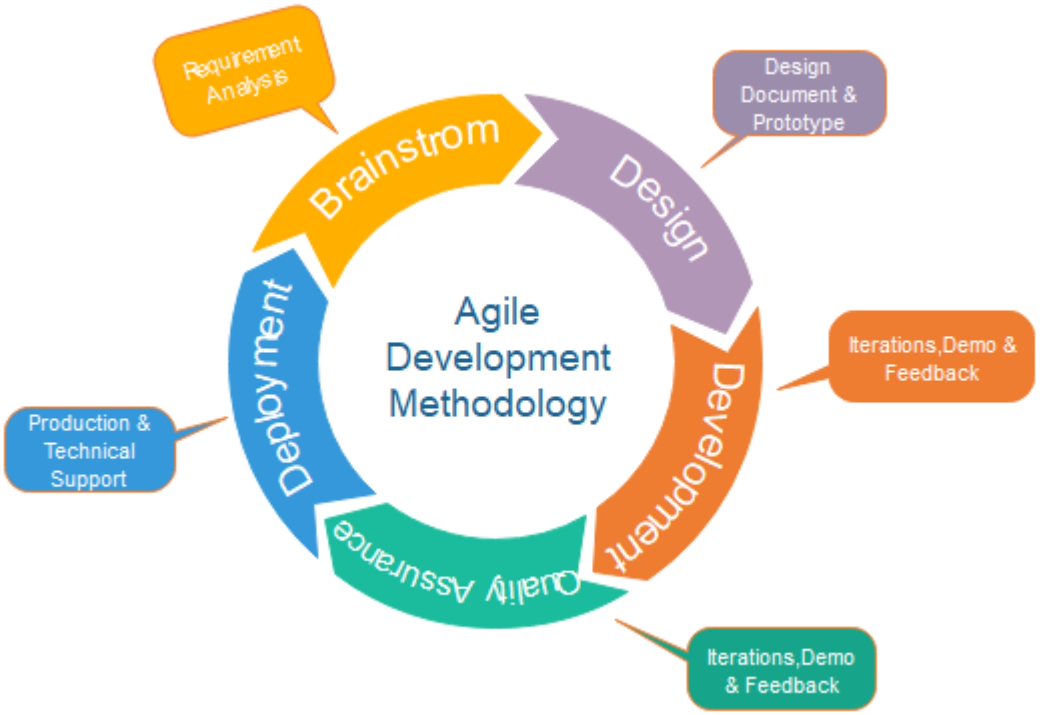

Fig 4.1: Agile Model

The Agile model is highly collaborative and involves close interaction between the development team, the customer or end-user, and other stakeholders. It also emphasizes flexibility and adaptation, with changes in requirements or priorities being incorporated into the development process quickly and efficiently.

4.4 Characteristics of an Incremental module

- Iterative and incremental: The Agile model involves breaking down the development process into smaller, more manageable iterations, with each iteration delivering potentially shippable software. The process is incremental, with each iteration building on the work done in previous iterations.
- Flexible and adaptable: The Agile model is designed to be flexible and adaptable to changing requirements and priorities. It enables development teams to respond quickly to new information, feedback, or changing business needs.
- Customer-centric: The Agile model places a strong emphasis on customer collaboration and feedback. It involves working closely with customers or end-users to understand their needs and priorities, and to ensure that the software meets their expectations.
- Collaborative: The Agile model promotes collaboration and communication among team members, stakeholders, and customers. It emphasizes teamwork and encourages frequent interactions and knowledge sharing.[8]

- Empirical: The Agile model is empirical, meaning that it is based on experience and observation rather than theory or speculation. It involves continuous experimentation, feedback, and improvement throughout the development process.
- Time-boxed: The Agile model involves working in short, time-boxed iterations, typically ranging from two to four weeks. This helps to keep the development process focused and on track and enables teams to deliver working software at regular intervals.
- Emphasis on quality: The Agile model places a strong emphasis on quality, with testing and validation integrated throughout the development process. It involves continuous testing and inspection to ensure that the software meets the required standards and is fit for purpose.

4.5 When to use this.

- Funding Schedule, Risk, Program Complexity, or need for early realization.
- When Requirements are known up-front.
- When projects have lengthy development schedules.
- Projects with new Technology [11].

4.6 Advantages

- Increased Flexibility
- Better Collaboration
- Faster Time-to-Market
- Higher Quality Software

4.7 Disadvantages

- Lack of Documentation
- Requires High Level of Customer Involvement
- Dependence on Teamwork and Skillset

4.8 Summary

Here in this chapter, we have discussed the SDLC model that is being used to develop our Pharma Trader Medicine Stock Management and Distributed System project. Here we have used the Agile Model.

Chapter 5
Requirements

5.1 Introduction to requirements

The process to gather the software requirements from clients and analyze and document them is known as requirement engineering. The goal of requirement engineering is to develop and maintain a sophisticated and descriptive 'System Requirements Specification' document.

5.1 Development Requirements

- Intel Core i3 8th Generation or Upper
- RAM (At least 8GB)
- Code Editor (VS code or others)
- SQLite (When debug true)
- Web Browser (Google chrome / Firefox)
- Python IDE (Python v3.7 or upper)

5.2 Python

Python is a general-purpose programming language, which means it can be utilized at almost everything. The most important thing is that it is an interpreted language which means that the written code has not been translated into a computer-readable format at execution time whereas, the major programming languages do this translation before the program runs. This type of language is also known as "scripting language" because it is intended for use for little projects. The idea of a "scripting language" has altered significantly since its inception, since Python is now utilized to write large business-style applications, rather than just common ones. This dependence on Python has full-grown even more as the Internet gained a reputation. A vast majority of web applications and platforms are dependent on Python, including the Google search engine [1].

5.3 XAMPP

XAMPP stands for eXtremely Accelerated Multi-Processing Packet Processing (A) Apache server, (M) MariaDB, (P) PHP, and (P) Perl. X stands for Cross-platform, (A) Apache server, (M) MariaDB, (P) PHP, and (P) Perl. The term "cross-platform" typically refers to the ability to run on any device, regardless of the operating system. So, we can say, XAMPP is an acronym that stands for Cross-Platform, Apache, MySQL, PHP, and Perl, with the Ps standing for PHP and Perl, respectively. It's an open-source web-solutions kit that provides Apache delivery for a variety of servers and command-line executables, as well as Apache API, MariaDB, PHP, and Perl

modules. Until releasing a website or client to the main cloud, XAMPP allows a local host or server to validate it on computers and laptops. It is a framework that provides a suitable environment for testing and verifying the functionality of projects based on Apache, Perl, MySQL, and PHP using the host's framework. Perl is a web creation programming language, PHP is a backend scripting language, and MariaDB is MySQL's most widely used database. The following is a brief overview of these elements.

5.4 PyCharm

PyCharm is a dedicated Python Integrated Development Environment (IDE) providing a wide range of essential tools for Python developers, tightly integrated to create a convenient environment for productive Python, web, and data science development. PyCharm is a hybrid platform developed by JetBrains as an IDE for Python. It is commonly used for Python application development. Some of the unicorn organizations such as Twitter, Facebook, Amazon, and Pinterest use PyCharm as their Python IDE. It supports two versions: v2.x and v3.x. We can run PyCharm on Windows, Linux, or Mac OS. Additionally, it contains modules and packages that help programmers develop software using Python in less time and with minimal effort. Further, it can also be customized according to the requirements of developers.

5.5 Web Browser

A web browser is a software program that allows a user to locate, access, and display web pages. In common usage, a web browser is usually shortened to "browser." Web browsers are used primarily for displaying and accessing websites on the internet, as well as other content created using languages such as Hypertext Markup Language (HTML) and Extensible Markup Language (XML). Browsers translate web pages and websites delivered using Hypertext Transfer Protocol (HTTP) into human-readable content. They also have the ability to display other protocols and prefixes, such as secure HTTP (HTTPS), File Transfer Protocol (FTP), email handling, and files.

5.6 Summary

Here in this chapter, we have discussed the hardware and software that are needed to build our project. It includes programming language IDE, framework, editor, etc.

Chapter 6
Coding Pre-requirements

6.1 Introduction to Dependencies

Dependency is a broad software engineering term used to refer to when a piece of software relies on another one. In software engineering, dependency is the degree to which each program module relies on each one of the other modules. When a piece of code relies on another piece of code, then we call it a dependency. Sometimes there can be different alternative dependencies for the same code and if we want to choose between them, then we have to figure out a common interface for them and pass the chosen one as a parameter. If we are talking about object-oriented programming and this passing happens in a constructor or in a setter, then we call this dependency injection.

6.2 Django

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source. Django offers a robust internationalization and localization framework to assist you in the development of applications for multiple languages. The Django project's stability, performance and community have grown tremendously over the past decade since the framework's creation. Detailed tutorials and good practices are readily available on the web and in books. The framework continues to add significant new functionality such as database migrations with each release [1].

6.3 Django-crispy-forms

Django-crispy-forms provides us with a '|crispy' filter and '{% crispy %}' tag that will let us control the rendering behavior of our Django forms in a very elegant and DRY way. Have full control without writing custom form templates. All this without breaking the standard way of doing things in Django, so it plays nice with any other form application [16]

6.4 Pillow

PIL is the Python Imaging Library by Fredrik Lundh and Contributors. This Python Imaging Library adds image processing capabilities to our Python interpreter. This library provides extensive file formatsupport, an efficient internal representation, and fairly powerful image processing capabilities. The core image library is designed for

fast access to data stored in a few basic pixel formats. It should provide a solid foundation for a general image processing tool [10].

6.5 Bootstrap4

Bootstrap is one of the most popular front-end frameworks out there. It contains some amazing CSS classes for UI development. Bootstrap has pre-defined CSS files and JavaScript code, which you can link with HTML files. Those CSS files contain classes that can be directly used on HTML elements. We used it in our Django static files. It is very easy to use Bootstrap in Django. Since Bootstrap is a front-end framework, it completely consists of CSS & JavaScript files. These files are considered static on the server-side.

6.6 MySQL client

The MySQL server package will install the MySQL database server which you can interact with using a MySQL client. You can use the MySQL client to send commands to any MySQL server, on a remote computer or your own. The MySQL server is used to persist the data and provide a query interface for it (SQL). The MySQL client's purpose is to allow you to use that query interface. The client package also comes with utilities that allow you to easily backup/restore data and administer the server.[7]

6.7 Summary

Here in this chapter, we have discussed the dependencies and databases that are needed to build our project. It includes Django, MySQL, bootstrap, etc.

Chapter 7
Design of the System

7.1 Introduction to system overview

Agile Model is a process of software development where requirements are broken down into multiple standalone modules of the software development cycle. Incremental development is done in steps from analysis, design, implementation, testing/verification, and maintenance. Each iteration passes through the requirements, design, coding, and testing phases and each subsequent release of the system adds function to the previous release until all designed functionality has been implemented.

7.2 Workflow in Agile Model

An incremental approach breaks the software development process down into small, manageable portions known as increments. Each increment builds on the previous version so that improvements are made step by step. The Agile Model produces a working product (in parts) at each increment. Receiving early delivery of the operational product (in parts) boosts the morale of the customer that his requirements are being satisfied [11].

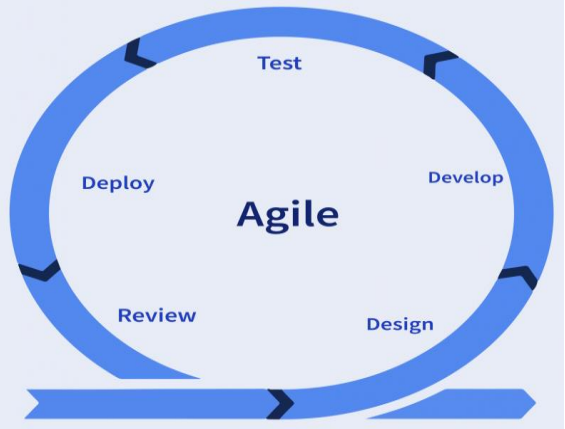

Fig 7.2: Workflow of Agile Model

1. **Requirement analysis:** In the first phase of the Agile Model, the product analysis expertise identifies the requirements. And the system's functional requirements are understood by the requirement analysis team. Developing the software under the Agile Model, this phase performs a crucial role.
2. **Design & Development:** In this phase of the Agile Model of SDLC, the design of the system functionality and the development method are finished with success. When software develops new practicality, the Agile Model uses the style and development phase.
3. **Testing:** In the Agile Model, the testing phase checks the performance of each existing function as well as additional functionality. In the testing phase, various methods are used to test the behavior of each task.

4. **Implementation:** The implementation phase enables the coding phase of the development system. It involves the final coding that is designed in the designing and development phase and tests the functionality in the testing phase. After completion of this phase, the number of the product working is enhanced and upgraded up to the final system product.

7.3 Flow Chart

A website flowchart (also known as a sitemap) maps out the structure and complexity of any current or future website. A well-structured sitemap or flowchart makes your website easily searchable. Each piece of content should ideally give users accurate search results, based on keywords connected to your web content. Product, UX, and content teams can use flowcharts or sitemaps to understand everything contained in a website, and plan to add or restructure content to improve a website's user experience.

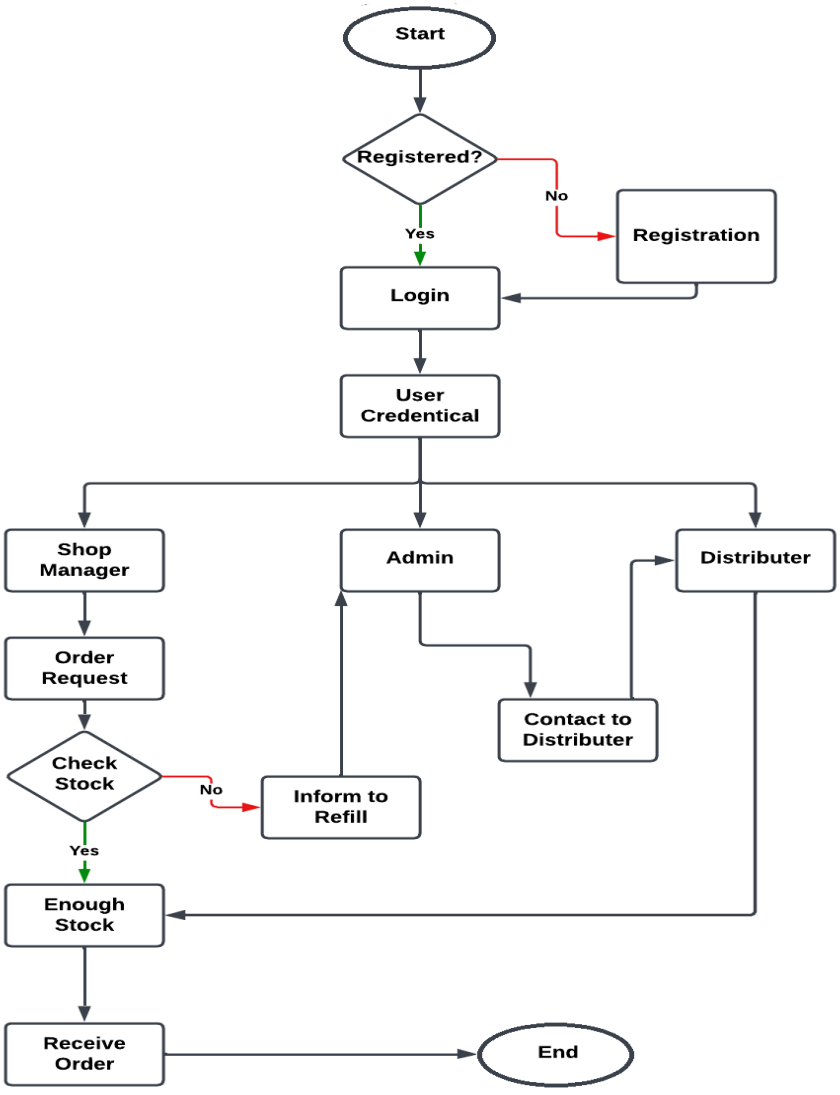

Fig 7.3: Flow chart of
Pharma Trader Medicine Stock Management and Distributed System

7.4 E-R Diagram

An Entity-Relationship (ER) Diagram is a type of flowchart that illustrates how "entities" such as people, objects, or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education, and research. Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals, and connecting lines to depict the interconnectedness of entities, relationships, and their attributes. They mirror grammatical structure, with entities as nouns and relationships as verbs. The number of times an entity of an entity set participates in a relationship set is known as cardinality. Cardinality can be of different types:

1. **One to one** – When each entity in each entity set can take part only once in the relationship, the cardinality is one-to-one. Let us assume that a male can marry one female and a female can marry one male. So, the relationship will be one-to-one.
2. **Many to one** – When entities in one entity set can take part only once in the relationship set and entities in other entity sets can take part more than once in the relationship set, cardinality is many to one. Let us assume that a student can take only one course, but one course can be taken by many students. So, the cardinality will be n to 1. It means that for one course there can be n students but for one student, there will be only one course.
3. **Many to many** – When entities in all entity sets can take part more than once in the relationship, cardinality is many to many. Let us assume that a student can take more than one course and one course can be taken by many students. So, the relationship will be many to many.

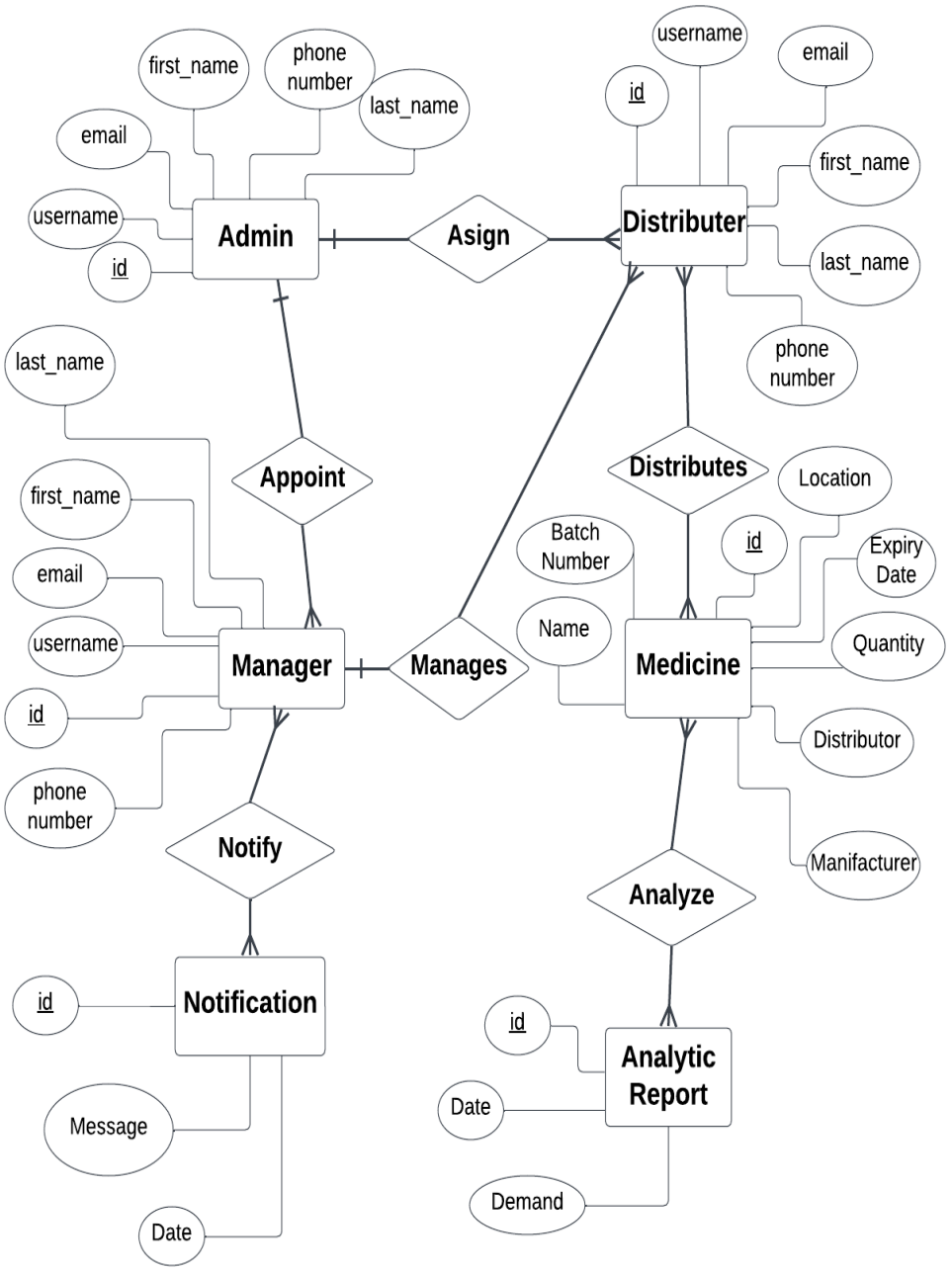

Fig 7.4: E-R Diagram of
Pharma Trader Medicine Stock Management and Distributed System

7.5 Workflow Daigram

A workflow diagram is a visual layout of a process, project or job in the form of a flow chart. It's a highly effective way to impart the steps more easily in a business process, how each one will be completed, by whom and in what sequence.
Workflow diagrams are commonly used to do the following:

Employ a holistic visual of business processes and information flows

Equip employees with a better understanding of their roles and responsibilities
Reveal process redundancies and bottlenecks

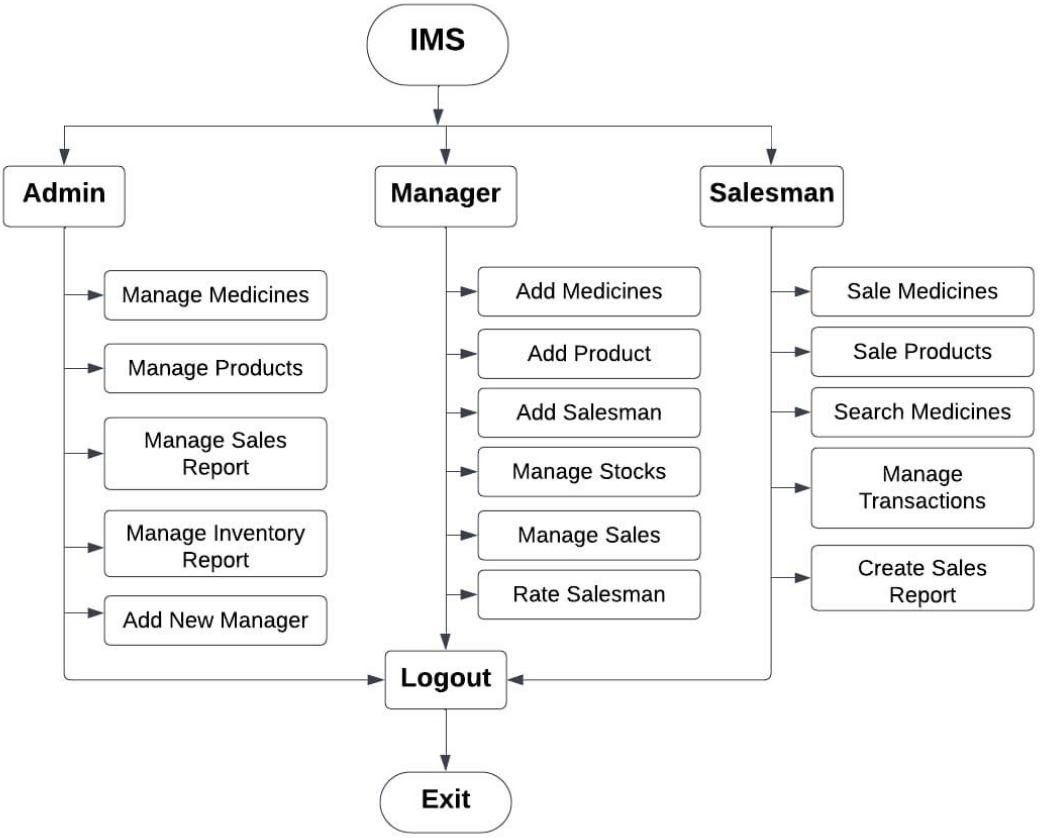

Fig 7.5: Workflow Daigram

7.6 Data Flow Diagram

DFD Level - 0:

It is also known as a context diagram. It's designed to be an abstract view, showing the system as a single process with its relationship to external entities. It represents the entire system as a single bubble with input and output data indicated by incoming/outgoing arrows.

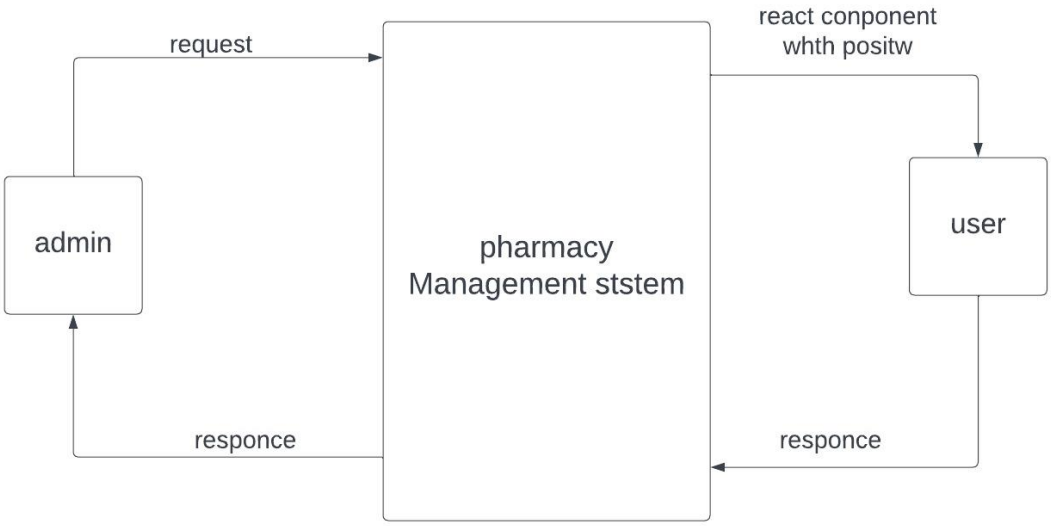

Fig 7.6.1: DFD level-0 of
Pharma Trader Medicine Stock Management and Distributed System

DFD Level-1:

In 1-level DFD, the context diagram is decomposed into multiple bubbles/processes. In this level, we highlight the main functions of the system and break down the high-level process of 0-level DFD into subprocesses.

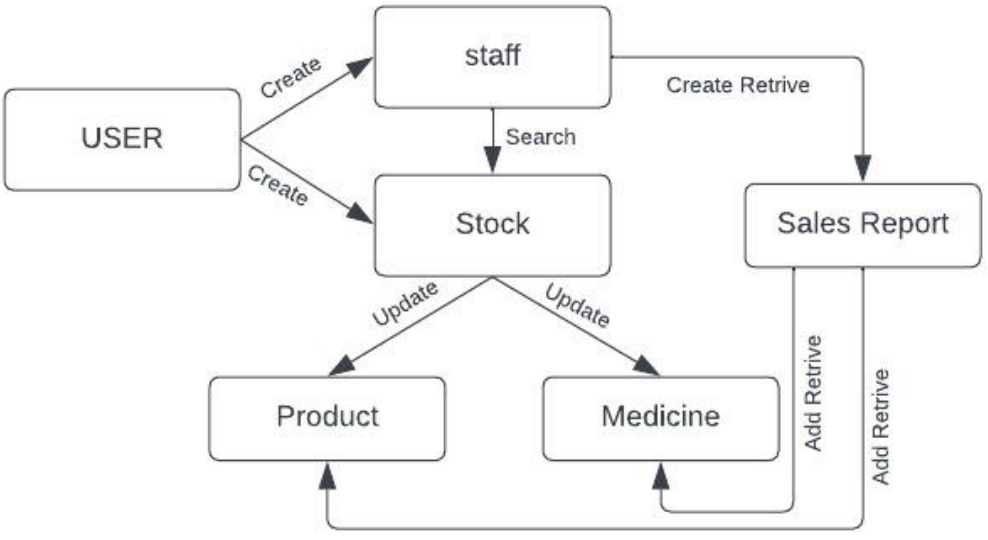

Fig 7.6.2: DFD level-1 of
Pharma Trader Medicine Stock Management and Distributed System

DFD Level-2:

2-level DFD goes one step deeper into parts of 1-level DFD. It can be used to plan or record the specific/necessary detail about the system's functioning.

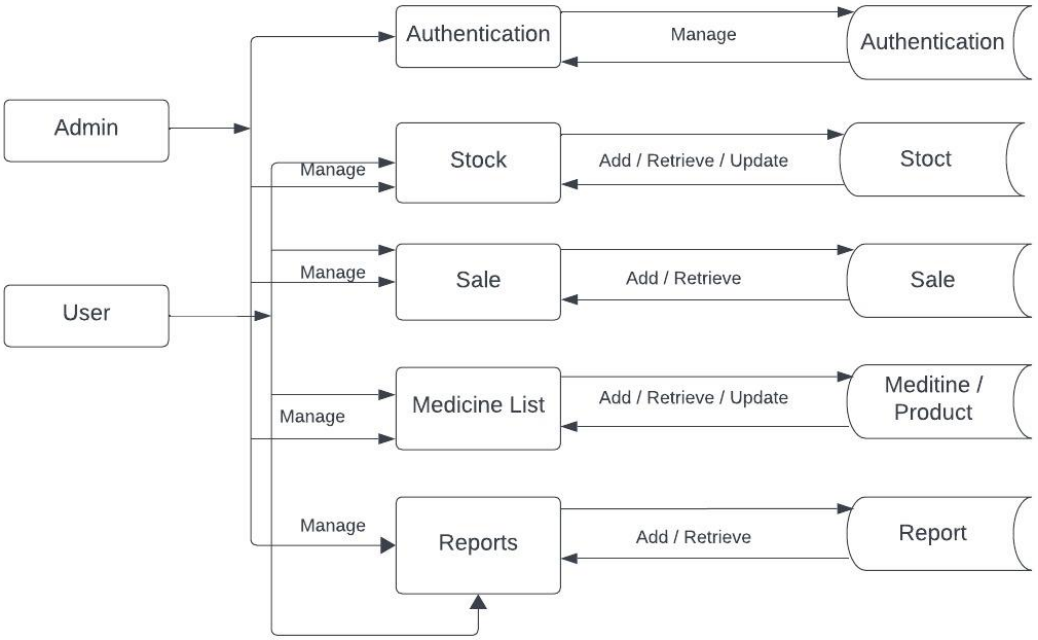

Fig 7.6.3: DFD level-2 of
Pharma Trader Medicine Stock Management and Distributed System

7.7 Sequence Diagram

A sequence diagram or system sequence diagram (SSD) shows object interactions arranged in time sequence in the field of software engineering. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios. For a particular scenario of a use case, the diagrams show the events that external actors generate, their order, and possible inter-system events. All systems are treated as a black box; the diagram places emphasis on events that cross the system boundary from actors to systems. A system sequence diagram should be done for the main success scenario of the use case, and frequent or complex alternative scenarios [5].

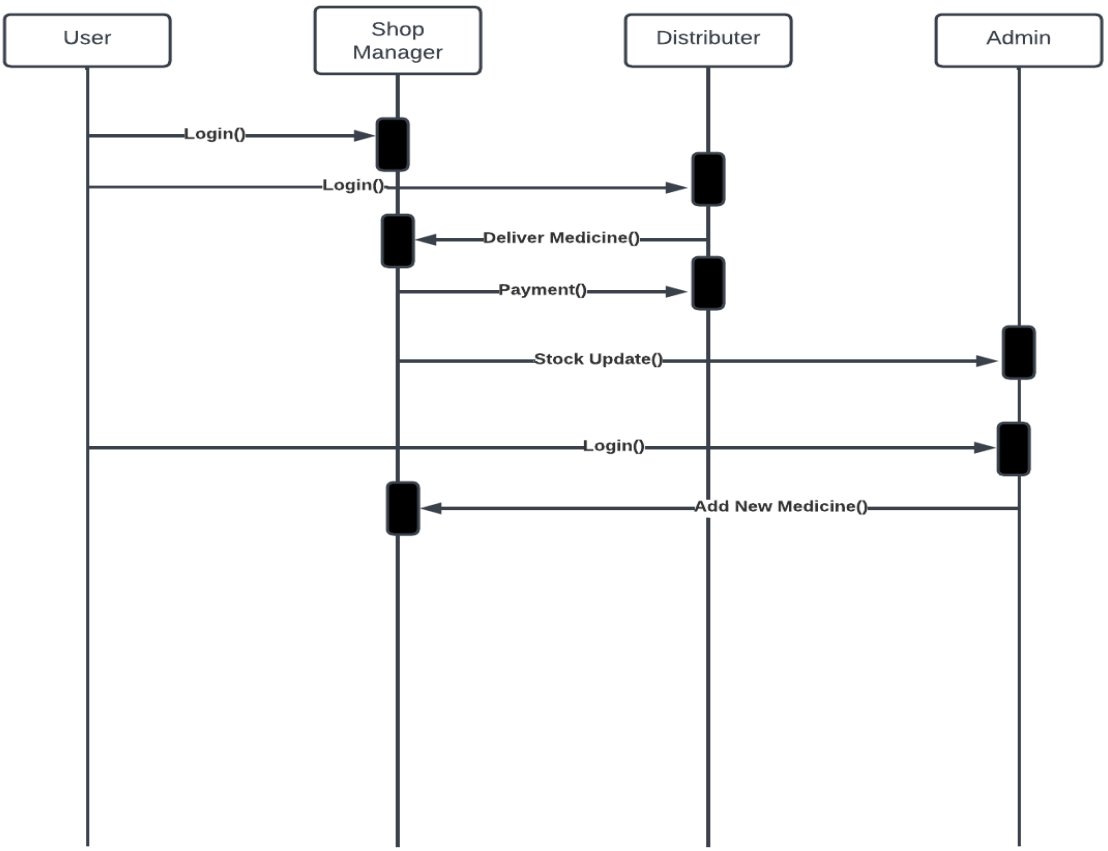

Fig 7.7: Sequence diagram of
Pharma Trader Medicine Stock Management and Distributed System

7.8 Class Diagram

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the structure of the application, and for detailed modeling, translating the models into programming code. Class diagrams can also be used for data modeling.[6] The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed. In the diagram, classes are represented with boxes that contain three compartments:

- The top compartment contains the name of the class. It is printed in bold and centered, and the first letter is capitalized.
- The middle compartment contains the attributes of the class. They are left-aligned, and the first letter is lowercase.
- The bottom compartment contains the operations the class can execute.

In the design of a system, a number of classes are identified and grouped together in a class diagram that helps to determine the static relations between them. In detailed modeling, the classes of the conceptual design are often split into subclasses.

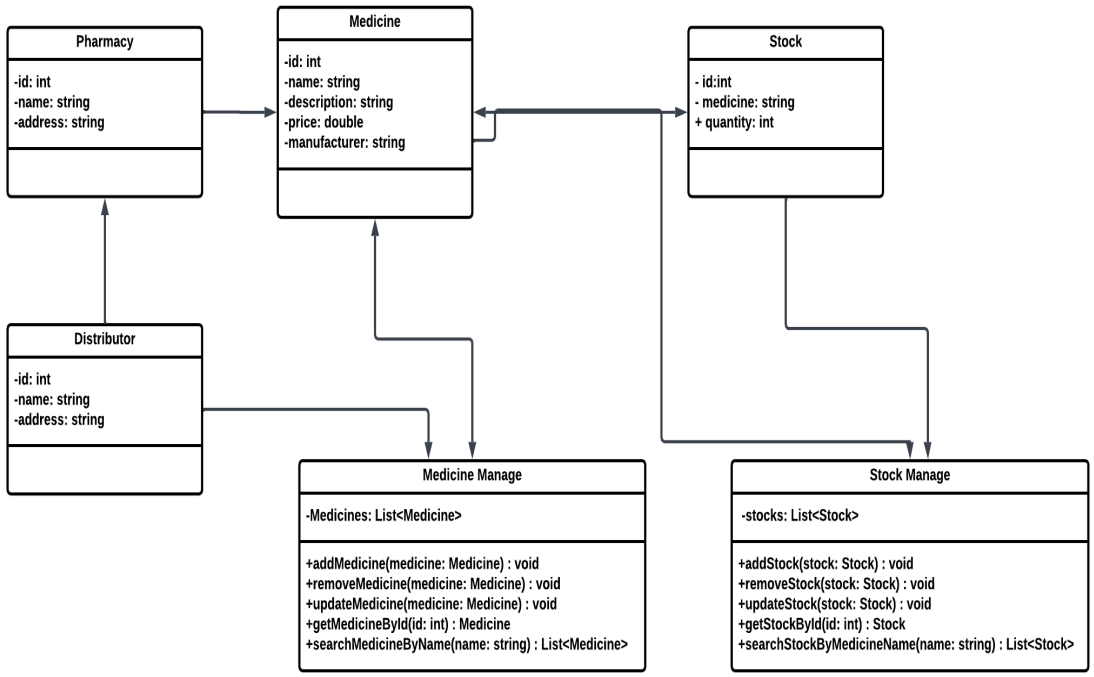

Fig 7.8: Class diagram of
Pharma Trader Medicine Stock Management and Distributed System

7.9 Use CASE Diagram

In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. To build one, you'll use a set of specialized symbols and connectors. An effective use case diagram can help your team discuss and represent:
Scenarios in which your system or application interacts with people, organizations, or external systems.
Goals that your system or application helps those entities (known as actors) achieve.

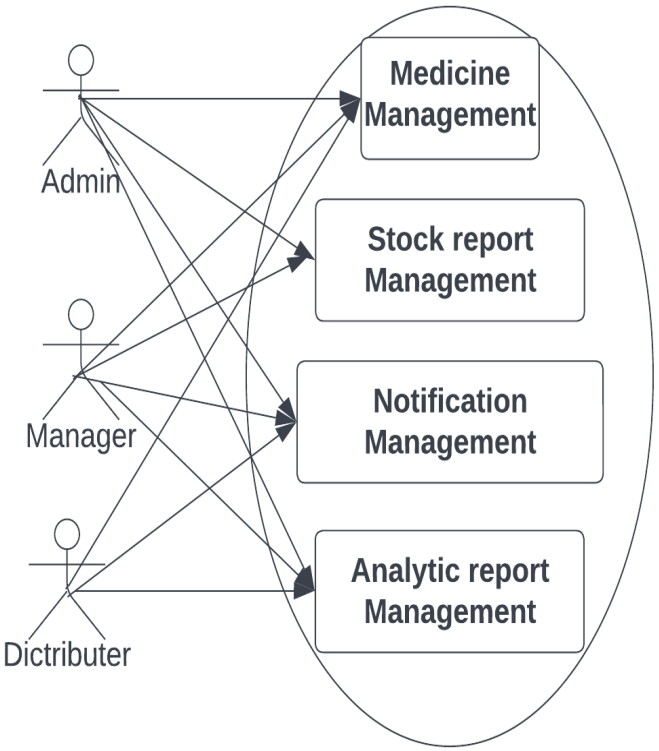

Fig 7.9: USE CASE Diagram of
Pharma Trader Medicine Stock Management and Distributed System

7.10 Summary

Here in this chapter, we have discussed the design models that are being followed to build our project. It includes a data flow diagram, flowchart, E-R diagram, use case diagram, class diagram, etc.

Chapter 8
Implementation & Output

8.1 Code

```python
from django.contrib.auth.base_user import BaseUserManager, AbstractBaseUser
from django.contrib.auth.models import PermissionsMixin
from django.db import models
from django.contrib.auth.hashers import make_password


# Create your models here.
class UserManager(BaseUserManager):
    def _create_user(self, email, password, **extra_fields):
        if not email:
            raise ValueError("The email field must be set!")
        email = self.normalize_email(email)
        user = self.model(email=email, **extra_fields)
        user.set_password(password)
        user.save(using=self._db)
        return user

    def create_superuser(self, email, password, **extra_fields):
        extra_fields.setdefault('is_staff', True)
        extra_fields.setdefault('is_superuser', True)

        if not extra_fields.get('is_staff'):
            raise ValueError("Superuser must have is_staff=True")
        if not extra_fields.get('is_superuser'):
            raise ValueError("Superuser must have is_superuser=True")

        return self._create_user(email, password, **extra_fields)


class CustomUser(AbstractBaseUser,
                 PermissionsMixin):  # to differentiate Default user and customized user, I named it CustomerUse
    email = models.EmailField(unique=True)
```

```python


# Create your views here.
class LargeResultsSetPagination(PageNumberPagination):
    page_size = 1
    page_size_query_param = 'page_size'
    max_page_size = 3


class MedicinesAPIView(generics.ListCreateAPIView):
    permission_classes = [permissions.IsAuthenticated, ]
    queryset = ProductModel.objects.all()
    serializer_class = ProductModelSerializer
    pagination_class = LargeResultsSetPagination

    def get(self, request, *args, **kwargs):
        queryset = self.get_queryset().exclude(expiry_date__lte=datetime.today()).filter(is_medicine=True).order
            'expiry_date')
        page = self.paginate_queryset(queryset)
        if page is not None:
            serializer = self.get_serializer(page, many=True)
            return self.get_paginated_response(serializer.data)

        serializer = self.get_serializer(queryset, many=True)
        return Response(serializer.data)


class ProductsAPIView(generics.ListCreateAPIView):
    permission_classes = [permissions.IsAuthenticated, ]
    queryset = ProductModel.objects.all()
    serializer_class = ProductModelSerializer
```

8.2 Output

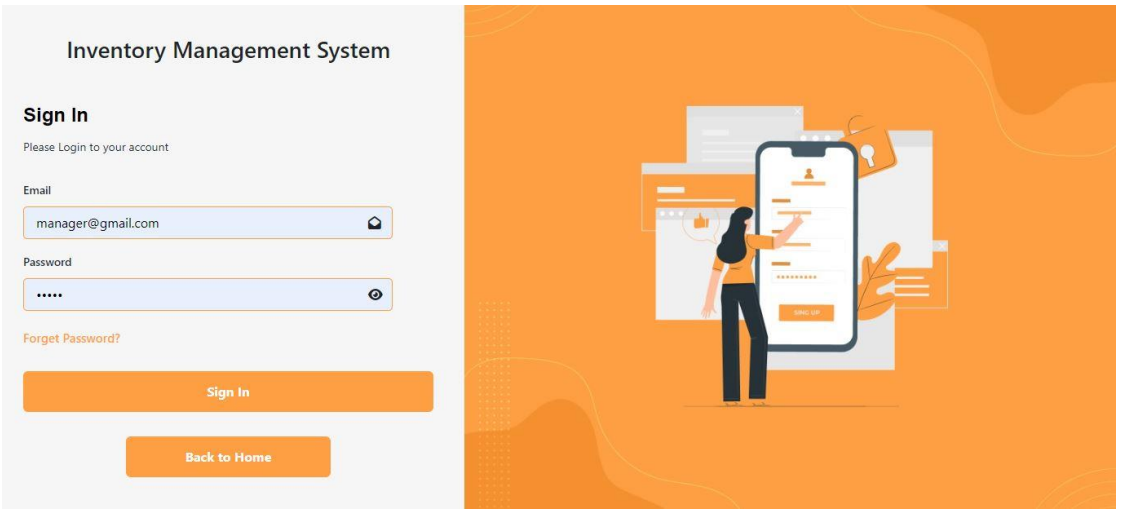

Fig: Login Page

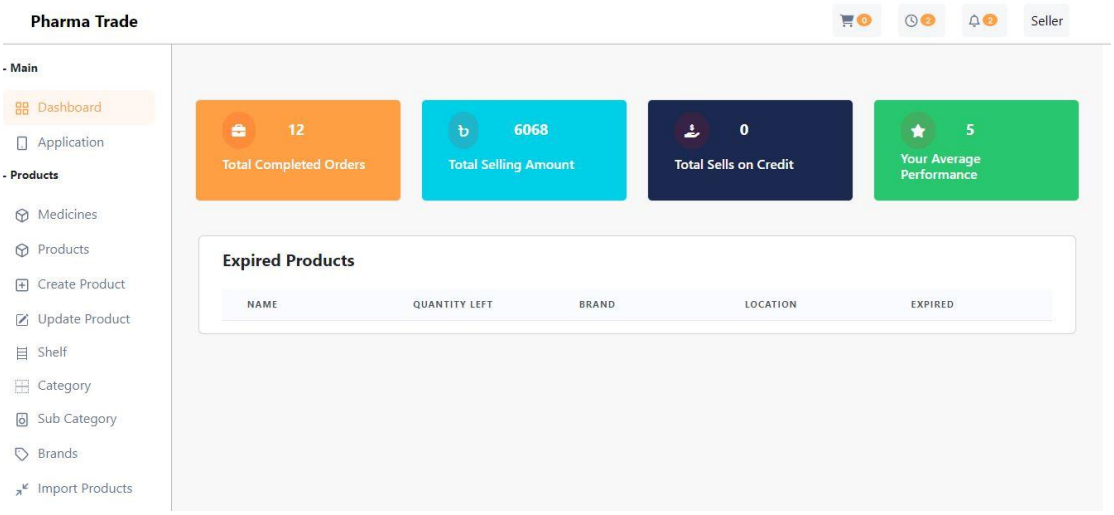

Fig: Dashboard Page

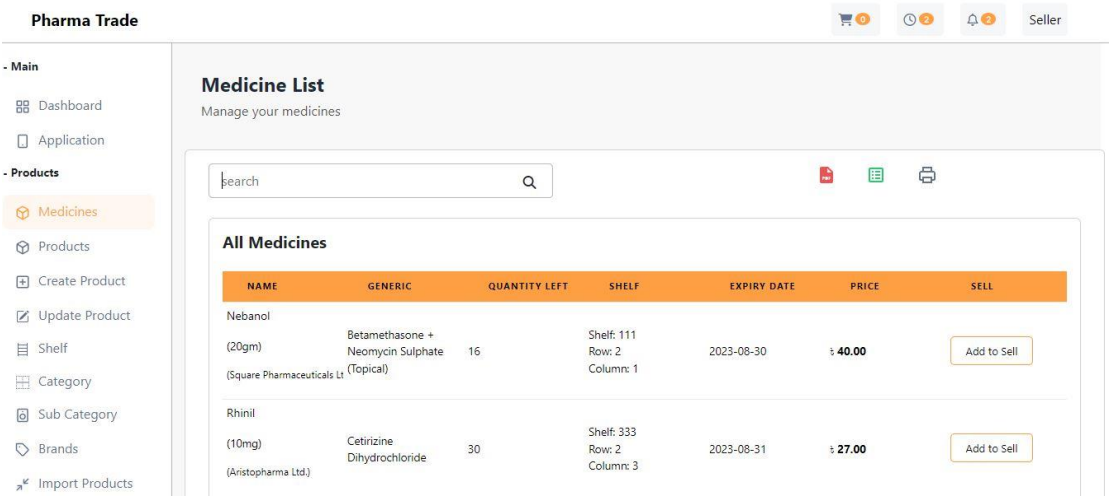

Fig: Medicine List

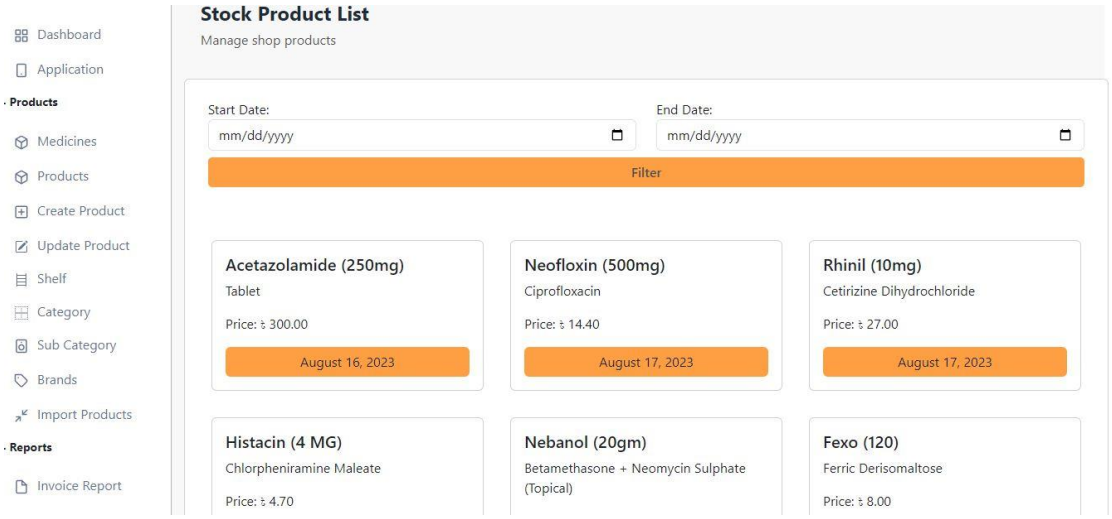

Fig: Stock Product Summary

Fig: Sales List & Summary

REFERENCES

1. Django. (n.d.). The web framework for perfectionists with deadlines. Retrieved March 23, 2023, from https://www.djangoproject.com/
2. Ayeni, A. J., & Soriyan, A. O. (2019). Design and implementation of a web-based inventory management system for small and medium scale enterprises. International Journal of Advanced Computer Science and Applications, 10(10), 393-399.
3. Beal, V. (2021). How to create a web-based inventory management system. Retrieved March 23, 2023, from https://www.thebalancesmb.com/how-to-create-a-web-based-inventory-management-system-4171063
4. Chaudhary, S., & Pandey, P. (2021). Implementation of an inventory management system using Django web framework. International Journal of Computer Sciences and Engineering, 9(7), 381-387.
5. Edeh, J. O., Ojugo, A. A., & Orji, U. J. (2019). Design and implementation of a web-based inventory management system for small and medium enterprises in Nigeria. International Journal of Computer Science and Information Security, 17(10), 7-16.
6. Jangir, B., & Khan, A. (2019). Development of a web-based inventory management system using Django framework. International Journal of Advanced Research in Computer Science, 10(1), 45-50.
7. Khalid, H. A., & Al-Shaqsi, S. H. (2021). Development of a web-based inventory management system using Django framework. Journal of Engineering Science and Technology Review, 14(2), 14-19.
8. Lee, J., Lee, J. H., & Kim, S. (2019). Development of a web-based inventory management system using Django framework. Journal of Software Engineering and Applications, 12(1), 1-12.
9. Python Software Foundation. (n.d.). Python. Retrieved March 23, 2023, from https://www.python.org/
10. Saleh, F. (2021). Inventory management system using Django framework. International Journal of Advanced Science and Technology, 30(8), 1527-1534.
11. Adatia, R., Tso, G., & Barua, S. (2017). Review of inventory management research in major logistics journals. International Journal of Logistics Management, 28(3), 839-859. doi: 10.1108/IJLM-09-2016-0195
12. Ali, W., & Zhang, L. (2019). A review of inventory management research in major logistics and supply chain management journals. Journal of Management Analytics, 6(2), 198-219. doi: 10.1080/23270012.2019.1624985
13. Almarabeh, T., & Bani Yassein, M. (2020). Medicine stock management system using blockchain technology. Journal of Medical Systems, 44(1), 1-15. doi: 10.1007/s10916-019-1544-4

Appendix

SDLC	Software Development Life Cycle
ERD	Entity Relationship Diagram
DB	Database
HTML	Hypertext markup language
UML	Unified Modeling Language
SSD	System Sequence Diagram
DFD	Data Flow Diagram
IDE	Integrated Development Environment
XML	Extensible Markup Language
HTTP	Hypertext Transfer Protocol
XAMPP	Extremely Accelerated Multiprocessing
FTP	File Transfer Protocol
ER Diagram	Entity Relationship Diagram