Therapeutical E-Commerce :

Content:

ABSTRACT

In the dynamic landscape of modern business, it's crucial to promptly meet the demands of clients. Catering to their desire for online accessibility and swift responses is paramount. Enter Online Shopping, a lifestyle e-commerce platform designed for seamless transactions. This initiative grants users the ability to browse a diverse array of products and make purchases instantly through PayPal (Instant Pay), or opt for the convenience of Cash on Delivery (Pay Later) for their orders.

This endeavor facilitates convenient access for Administrators and Managers to monitor orders processed through both Pay Later and Instant Pay avenues. Building an e-commerce website necessitates a comprehensive exploration and comprehension of various technologies. These encompass multi-tiered architecture, scripting techniques for both server and client sides, as well as implementation tools like Spring Boot, programming languages such as Java, and relational databases. The aim of this project is to construct a fundamental website featuring a shopping cart application, while also delving into the technological frameworks essential for its development. This document will delve into each of these underlying technologies required to construct and deploy a medical e-commerce platform.

INTRODUCTION

The Therapeutical E-Commerce System project embodies a comprehensive web-based application developed using Vue.js and Spring Boot technologies within the Visual Studio Software environment. Its primary objective is to establish a platform that seamlessly facilitates the purchase and sale of medical products and services, prioritizing convenience, accessibility, and efficiency.

Key objectives and benefits include:

1. **Accessibility**: This online medical e-commerce solution offers unparalleled accessibility, allowing individuals to procure a wide array of medical goods and services from the comfort of their homes or workplaces, irrespective of their geographical location. This feature is particularly advantageous for those with mobility constraints or residing in remote areas with limited access to physical medical establishments.
2. **Convenience**: By enabling users to peruse, compare, and purchase medical essentials online, e-commerce platforms provide a level of convenience unmatched by traditional brick-and-mortar stores. Users can shop at their convenience, avoiding the constraints of physical

store hours and queues.

3. **Product Variety**: Online medical e-commerce platforms boast an extensive range of offerings, encompassing over-the-counter medications, prescription drugs, medical devices, supplements, personal care products, and wellness services. This diversity empowers users to procure specific items without constraints.
4. **Price Transparency and Competition**: E-commerce platforms promote price transparency by facilitating price and review comparisons across multiple sellers, empowering users to make informed purchasing decisions. Additionally, the competitive nature of online markets often leads to competitive pricing and promotional offers, ensuring cost-effective options for users.
5. **Privacy and Discretion**: Online medical e-commerce respects user privacy and discretion, allowing discreet browsing and purchasing of medical goods and services without the need for face-to-face interactions. This feature is particularly valuable for individuals seeking sensitive products or treatments.
6. **Information and Education**: E-commerce platforms serve as repositories of valuable information and educational resources on medical products, conditions, treatments, and

wellness practices. This educational content empowers users to make informed decisions about their health, fostering self-care and preventive healthcare practices.

7. **Contribution to Healthcare Accessibility**: Online medical e-commerce enhances overall healthcare accessibility by complementing traditional healthcare delivery models. It ensures access to essential medical supplies, particularly for individuals with chronic conditions or special healthcare needs, and serves as a vital resource during emergencies or public health crises.

that traditional brick-and-mortar stores may not provide. Users can shop at any time of the day or night, avoiding the need to travel to physical stores and wait in long queues.

Product Variety: Online medical e-commerce platforms typically offer a diverse range of medical products and services, including over-the-counter medications, prescription drugs, medical devices, health supplements, personal care products, and wellness services. This extensive product variety allows users to find and purchase the specific items they need without limitations.

Price Transparency and Competition: E-commerce platforms foster price transparency by allowing users to compare prices and reviews from multiple sellers, facilitating informed purchasing decisions. Additionally, the competitive nature of online markets often results in competitive pricing and promotional offers, providing users with cost-effective options for their medical needs.

Privacy and Discretion: Online medical e-commerce respects user privacy and

discretion by allowing individuals to browse and purchase medical products and services discreetly, without the need for face-to-face interactions with healthcare providers or pharmacists. This can be particularly important for individuals seeking sensitive products or treatments.

Information and Education: E-commerce platforms often provide valuable information and educational resources related to medical products, conditions, treatments, and wellness practices. This educational content empowers users to make informed decisions about their health and well-being, promoting self-care and preventive healthcare practices.

Contribution to Healthcare Accessibility: Online medical e-commerce plays a role in improving overall healthcare accessibility by complementing traditional healthcare delivery models. It facilitates access to essential medical supplies, particularly for individuals with chronic conditions or special healthcare needs, and may serve as a valuable resource during emergencies or public health crises.

Architecture:

Overview: The system will employ a client-server architecture, where the client-side application (web browser) communicates with the server-side application through HTTP requests. The server-side application will handle the business logic and interface with the database for data storage and retrieval purposes.

Components:

Frontend: Developed using modern web technologies (HTML, CSS, JavaScript), with frameworks Vue.js for dynamic user interfaces.

Backend: Implemented using Java Spring Boot, responsible for handling HTTP requests, processing business logic, and serving data to the frontend.

Database: Utilizes a relational database management system (MySQL) for storing structured data related to users, products, orders, etc.

Interactions: The frontend communicates with the backend through RESTful APIs, sending HTTP requests (GET, POST, PUT, DELETE) to perform CRUD (Create, Read, Update, Delete) operations on resources.

System Overview

The sales process for medical products needs an application to manage it Their works. This application should be web-based and able to accommodate various Business functions to handle daily operations. Among the requirements are the following:

1. Web Based Application
2. Database driven (SQL)
3. Secure
4. Facilitate Back Office Operations:
    a. Supply Inventory
    b. Product Inventory
    c. Shipping
    d. Receiving
    e. Accounts Payable
    f. Scheduling

Design Map

Software Methodology

The overall project requirements will be clearly defined in the outset of the project, with dated Gantt chart and clear project responsibilities. Since the customer has no need to review programming technique or progress, a design methodology of the type “Waterfall” will be used. The waterfall methodology allows the project to be developed and managed when segmented into a hierarchy of phases, activities, tasks and steps. This model lends itself well to the Gantt chart, which will help monitor the development process, and increase the likelihood of an on time delivery of the source code. The development process will be broken into 5 Phases:

Design → Code → Test → Bug Fixes → Documentation

Design Phase

The design phase consists of producing this document, which includes the software design requirements. These requirements will be reviewed by the primary contact for the final deliverable, and once approved, will continue to the “code” phase.

Code phase

The code phase involves application development, based on the design specifications contained in this document. This phase is further subdivided into three distinct portions: API Interface, Primary Interface, and Database Design.

Test phase

The testing phase will document all user processes, and verify their functionality. The customer will be involved in this phase to ensure the system is functioning as expected. Additionally, “Bugzilla” will be utilize to track and close any encountered issues.

Bug Fixes

During the testing phase, the bug fix phase will run concurrently, as development repairs any bugs, and testing verifies bug fixes.

Documentation

will include source, and user documentation, as well as known issues and limitations.

Constraints

Software

Product Classification: Medical products are classified into different categories based on factors such as intended use, risk level. The software must support accurate classification and categorization of products to ensure appropriate listing, labeling, and distribution.

Product Information Management: The software must support the storage and retrieval of comprehensive product data, including descriptions, specifications, indications, contraindications, precautions, and usage instructions.
Inventory Management: Effective inventory management is essential to ensure the availability of medical products and prevent stockouts or overstock situations.
Quality Assurance: The software must implement quality assurance measures such as product authentication, batch tracking, expiration date management, and supplier verification to mitigate the risk of counterfeit or substandard products.

Hardware

Testing and performance monitoring should be baselined on this configuration. The server will have up to two processors, and 2 GB of RAM.

User Interface Design

Input Field for Adding New products (Admin)
Buttons for Editing and Deleting products (Admin)
Buttons for Editing and Deleting the chosen products

Backend Design

application is designed as a RESTful API to facilitate communication between the client-side interface and the server-side logic.

The following endpoints and functionalities are defined:
GET /tasks: Retrieves all tasks from the database and returns them as JSON objects to the client.
POST /tasks: Receives a new task object from the client, validates it, and inserts it into the database. Returns the newly created task object with a unique identifier.
PUT /tasks/:id: Accepts an updated task object with the specified identifier from the client, validates it, and updates the corresponding task in the database. Returns the updated task object.
DELETE /tasks/:id: Deletes the task with the specified identifier from the database. Returns a success message upon successful deletion.

OVERALL DESCRIPTION

DESCRIPTION:-

- Any member can register and view available products.
- Any member can search for products
- There are three roles available: Visitor, User and Admin.
    1. Guest can view available products.
    2. Customer can view and purchase products.
    3. An Admin has some extra privilege including all privilege of Visitor and User.

        - Admin can add products, edit product information and add/remove product.

        - Admin can add User, edit User information and can remove User.

        - Admin can ship order to User based on order placed by sending confirmation mail.

Using the code:

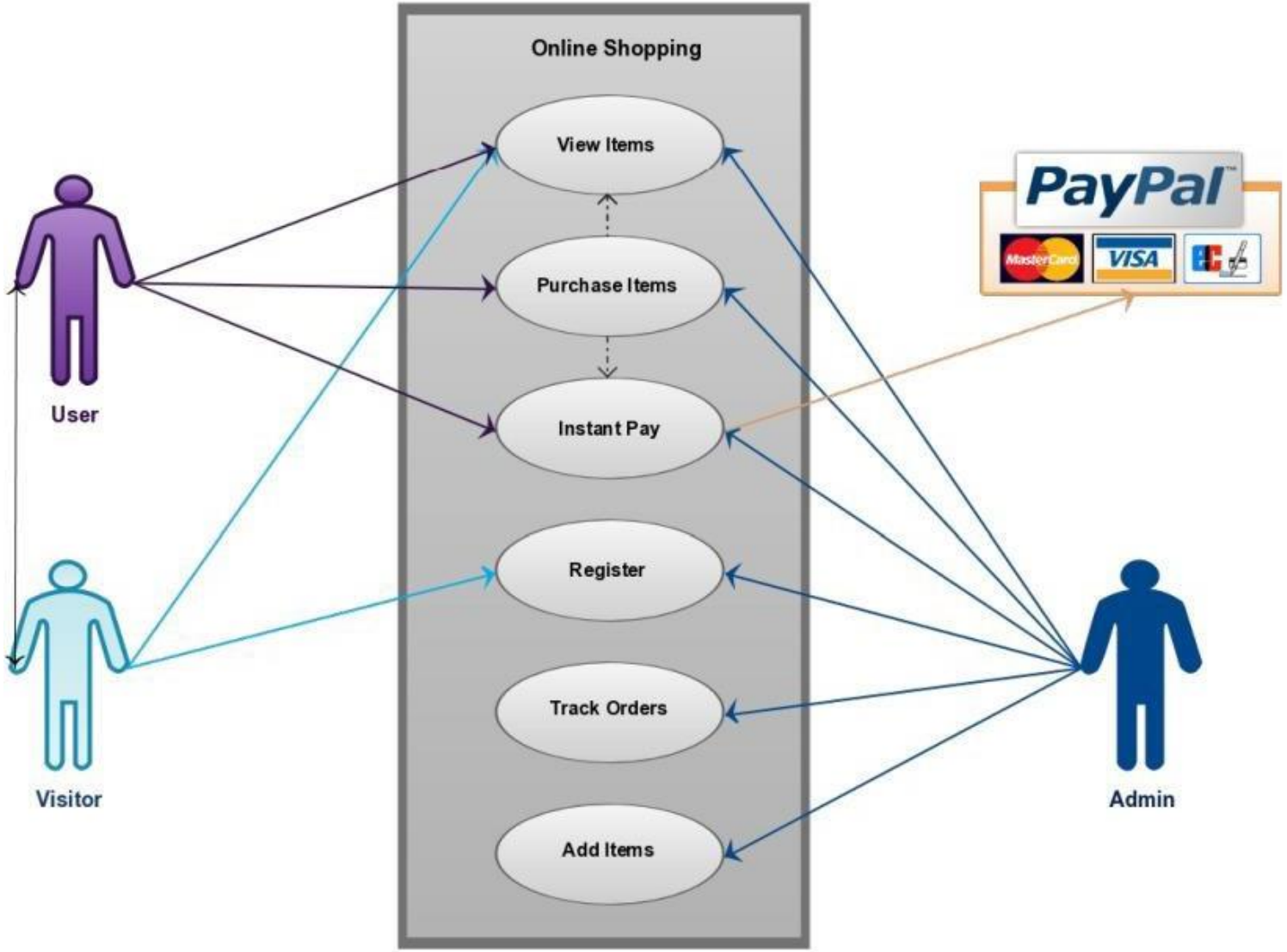
1. Attach the database in your “XAMPP Server ".
2. Run the application on Microsoft Visual Studio as web site.
3. Locate the database.

Web Pages details:

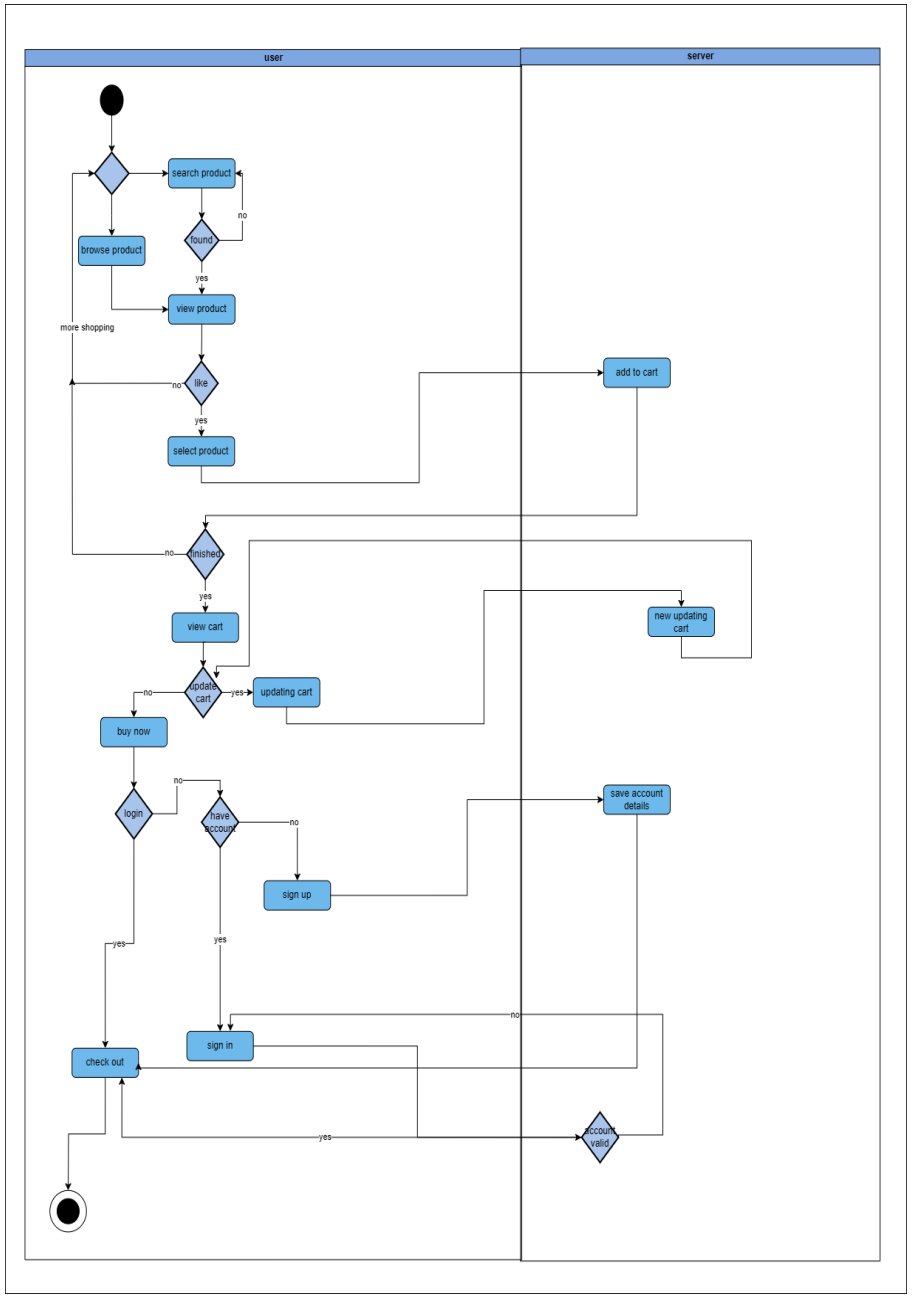
- Home Page
- About Us Page
- Blog Page
- Shop Page
- Contact Us Page
- Admin Page
- Login Page
- Your Cart Page

SYSTEM REQUIREMENTS

USE-CASE DIAGRAM:



ACTIVITY DIAGRAM :

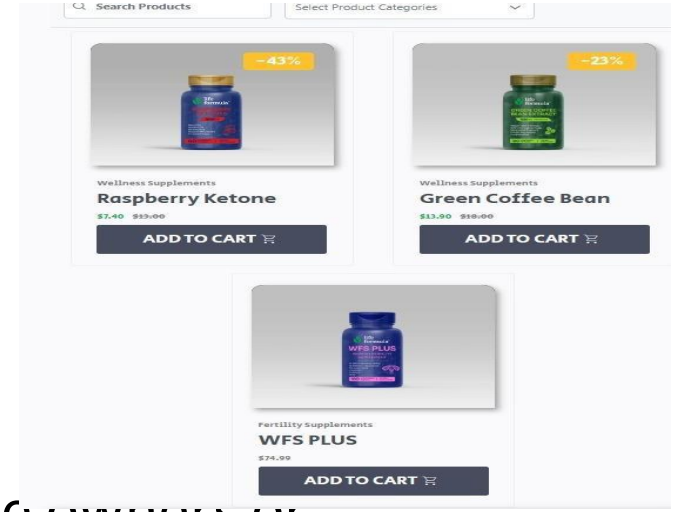


Therapeutical ONLINE SHOPPING APPLICATION

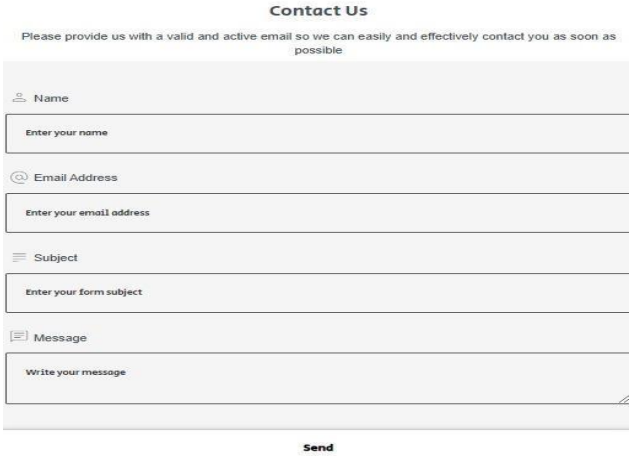
Anyone can view Online Shopping portal and available products, but every user must login by his/her Username and password in order to purchase or order products. Unregistered members can register by navigating to registration page. Only Admin will have access to modify roles, by default developer can only be an ‘Admin’. Once user register site, his default role will be ‘User’.

HOMEPAGE: The Home Screen will consist of screen were one can browse through the products which we have on our website

SHOP PAGE (PRODUCTS): This page consists of product details. This page appears same for both visitors and users.



Contact Us Page: Visitors and Registered users can contact website owners or administrators from here



ABOUT US PAGE: This page describes about website and owners .

REGISTER PAGE: New users can register here.

LOGIN PAGE: Login page for both users and administrators.

Admin Page: Only difference you see in this page is Role: Admin. User and Admin role will be checked once the page was login and Session [“role”] will be either Admin or User. If credentials belong to Admin then role will be Admin and if credentials belong to User then role will be User.

ORDER VIEW FOR USER: Once users order item they are able to see ordered products and grand total.

PAYPAL FOR PAYMENT: Once users orders products they are redirected to payment page.

Data Management

This database consists of

Users:

User and Admin information is added to database with Unique ID based on their roles.

Shop:

Complete products information is stored in this table.

Orders:

Customer ordered products, status and delivery information is stored in this table.

Data Objects

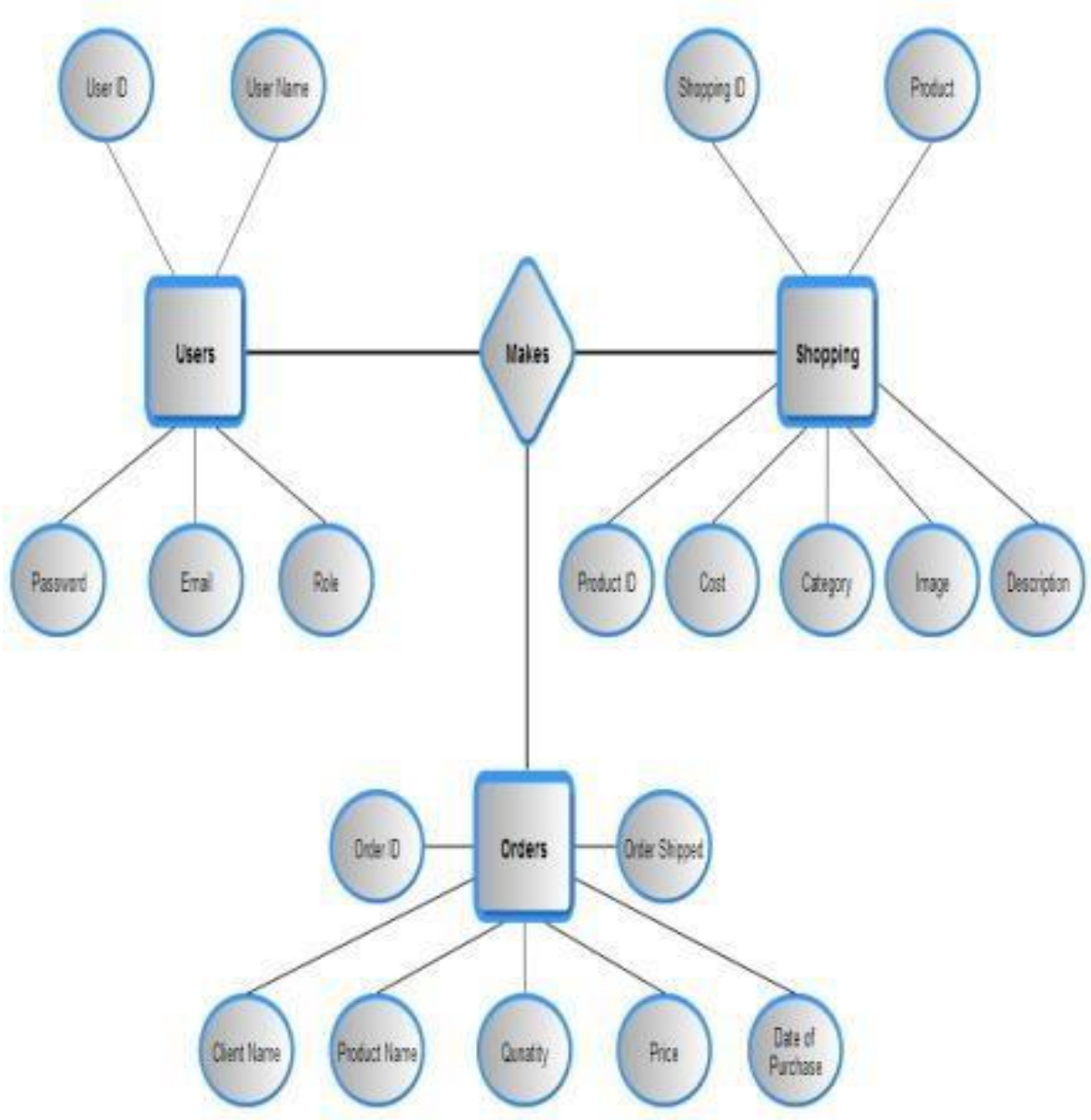
User: ID, UserName, Password, Email, Role

Shopping: ID, Product, Product ID, Cost, Category, Image, Description

Orders: ID, Client, Product, Quantity, Price, Date

Database Table Diagram

Relationships:



Functional / Operational Requirements

User Registration and Authentication

- Users should be able to register accounts with personal information.
- Authentication mechanisms should ensure secure access to user accounts.

Product Catalog

- Display a catalog of medical products with detailed descriptions.
- Categorize products for easy navigation.
- Allow users to search and filter products by category, brand, price, etc.

Product Details

- Provide detailed information about each product, including price, availability, specifications, and images.

Shopping Cart

- Allow users to add products to a shopping cart.
- Enable users to view, modify, and remove items from the cart.
- Calculate the total price including taxes and shipping costs.

Checkout Process

- Guide users through a secure checkout process.
- Collect shipping and billing information.
- Support various payment methods, such as credit/debit cards, PayPal, etc.
- Provide order confirmation and receipt.

Non-Functional / Operational Requirements

Security

- Pages of the website must be access in the way they were intended to be accessed. Included files shall not be accessed outside of their parent file.
- Administrator can only perform administrative task on pages they are privileged to access. Customers will not be allowed to access the administrator pages
- **Threat Analysis:** A thorough threat analysis will be conducted to identify potential security vulnerabilities and threats to the system.
- Defense Mechanisms: Security measures such as input validation, output encoding, parameterized queries, and prepared statements will be implemented to mitigate common security risks such as SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF).
- **Data Encryption:** Sensitive data such as user passwords and payment information will be encrypted using industry-standard encryption algorithms to protect against unauthorized access.

HTTPS Encryption: Communication between the client-side interface and the server-side API is encrypted using HTTPS protocol to ensure data confidentiality and integrity, mitigating the risk of eavesdropping and man-in-the-middle attacks.

Authentication and Authorization: To differentiate between roles between admin and users

Efficiency and Maintainability

- Page loads should be returned and formatted in a timely fashion depending on the request being made. Administrators will have the ability to edit the aspects of the order forms, product descriptions, prices and website directly

Scalability and Performance Design

The application is designed to handle multiple concurrent users.

Deployment and Maintenance

The application can be deployed on a cloud / Docker platform. Maintenance involves regular updates to fix bugs and add new features.

Performance

Ensure fast loading times for product pages and checkout process. - Handle concurrent user sessions and transactions efficiently. - Optimize database queries and server responses to minimize latency.

Scalability

Design the application to handle increased traffic and growing product inventory. - Implement scalable infrastructure and cloud services to support future growth.

Reliability

Minimize downtime and ensure high availability of the application. - Implement backup and disaster recovery mechanisms to prevent data loss.

Usability

Design an intuitive user interface that is easy to navigate and understand. - Ensure compatibility with various devices and screen sizes (responsive design). - Provide helpful tooltips, error messages, and guidance throughout the application

Accuracy

The system should accurately provide real time information taking into consideration various concurrency issues. The system shall provide 100% access reliability.

Conclusion

The internet has emerged as a pivotal tool in contemporary business, thus elevating the significance of electronic shopping from both the entrepreneur's and customer's perspectives. For entrepreneurs, electronic shopping opens new avenues for business growth, while for customers, it facilitates comparative shopping. Research indicates that most online store consumers are impulsive and tend to make decisions about site engagement within the initial moments of browsing. Just as a well-designed shop interior can enhance customer retention, a thoughtfully crafted website design is crucial for user engagement. Therefore, our project focuses on delivering seamless navigation, efficient data retrieval, and timely feedback to users.

For the implementation of this web application, we employed Spring Boot as the technology platform. Spring Boot offers numerous benefits including improved performance, scalability, built-in security features, and simplicity.

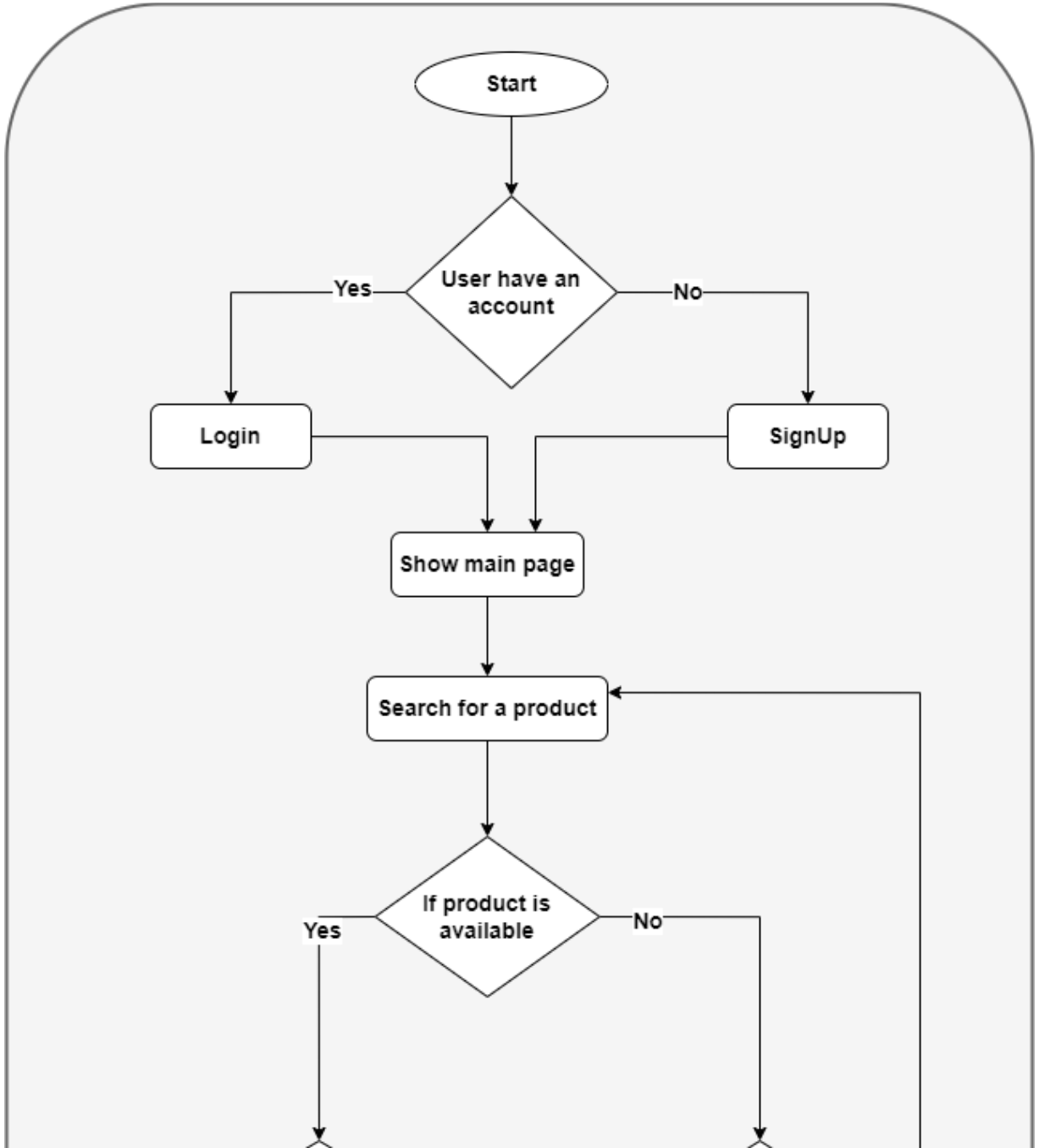
Java serves as the programming language for building web applications with Spring Boot.

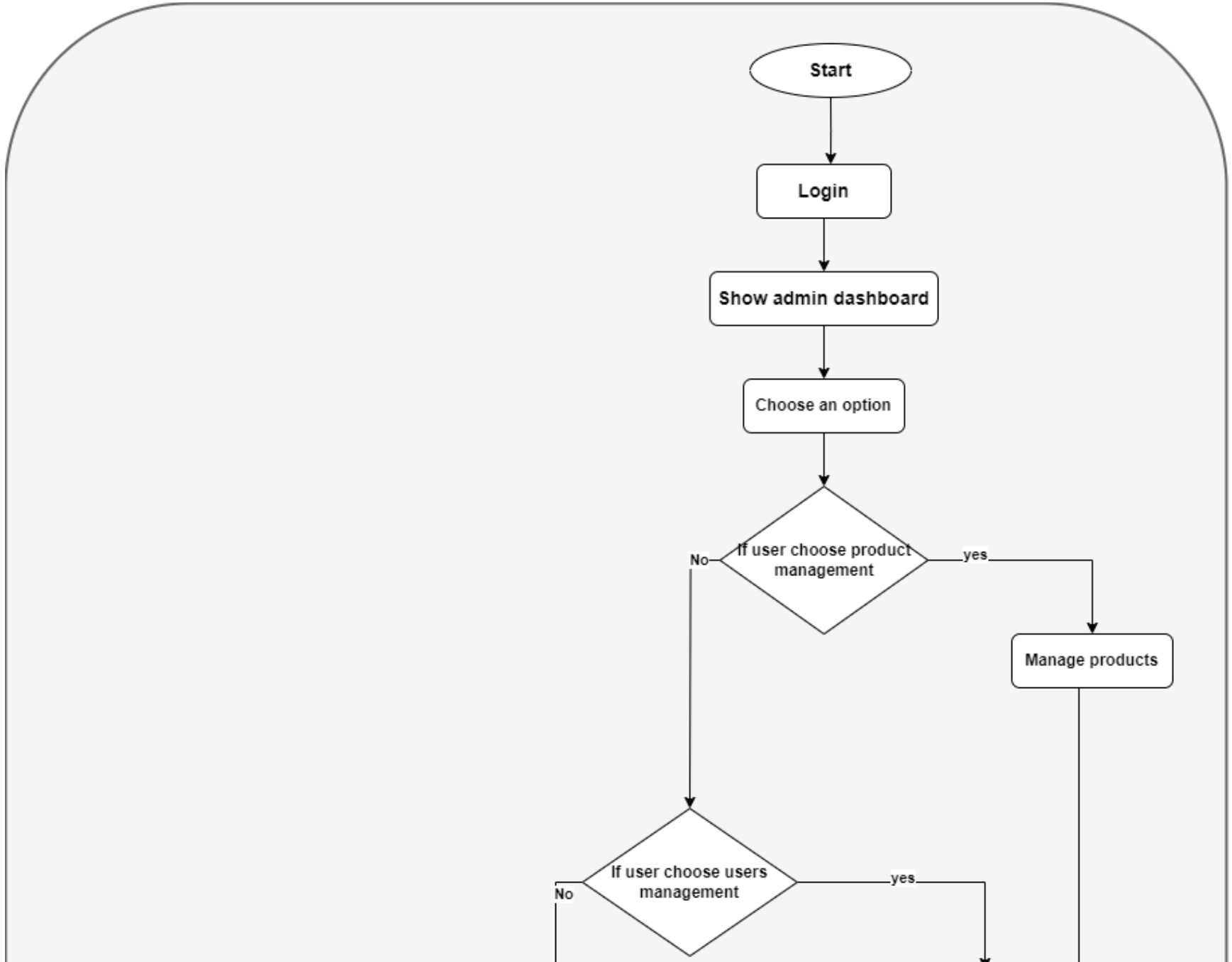
MySQL was chosen as the backend database due to its widespread popularity, fast data access capabilities, easy installation process, and user-friendly interface.

A well-designed interface should be complemented by user-friendly application logic, ensuring that customers can conveniently view, add, or remove items from their shopping cart. The shopping cart application outlined in this project incorporates various features aimed at enhancing customer convenience.

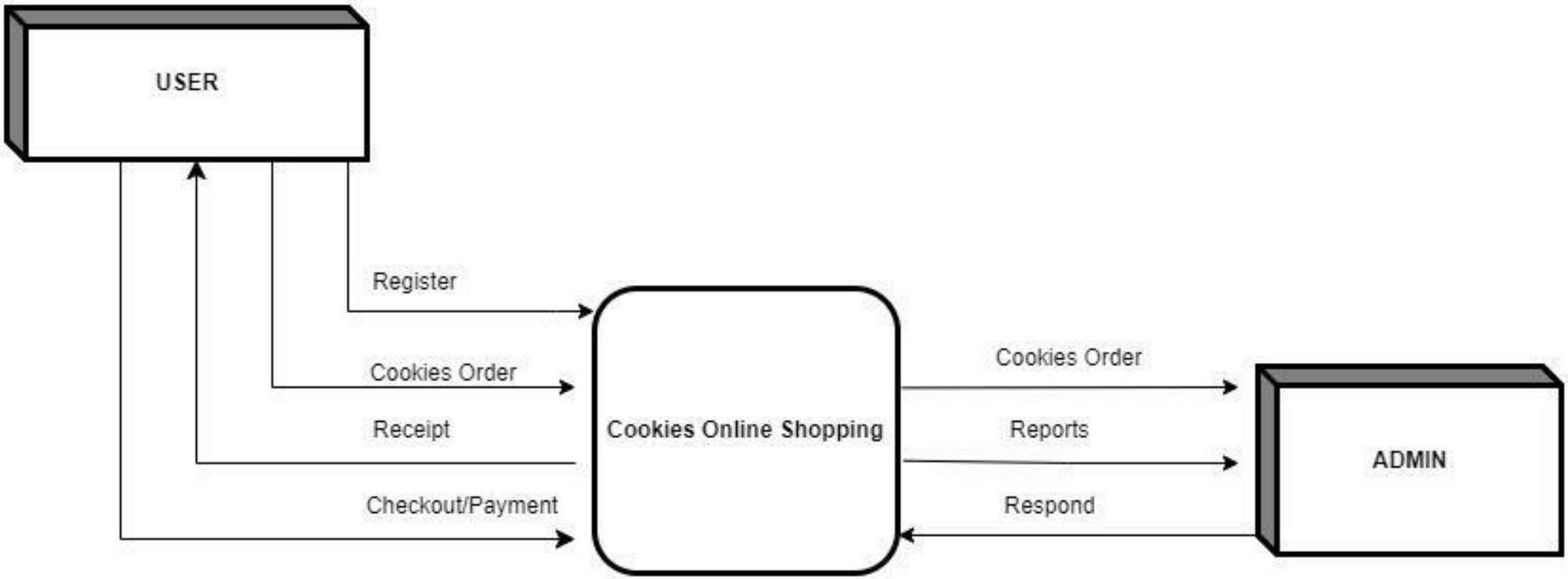
This project serves as a learning opportunity for understanding the development of interactive web pages and the underlying technologies involved in their implementation. Through its construction, I've gained invaluable insights into how Spring Boot facilitates website development, how it establishes connections with databases for data retrieval and modification, and how these components collaboratively shape the user experience of the application.

Another versions of diagrams:
Activiy diagrams for Customer and Admin:


Start
User have an account
Yes
No
Login
SignUp
Show main page
Search for a product
If product is available
Yes
No


Start
Login
Show admin dashboard
Choose an option
If user choose product management
No
yes
Manage products
If user choose users management
No
yes

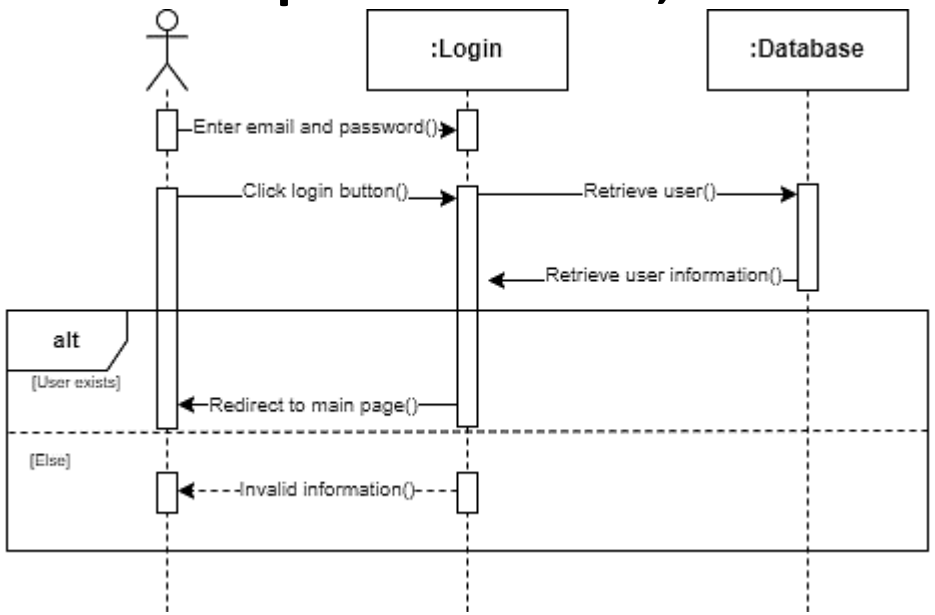
Data flow diagram:
Context Diagram:

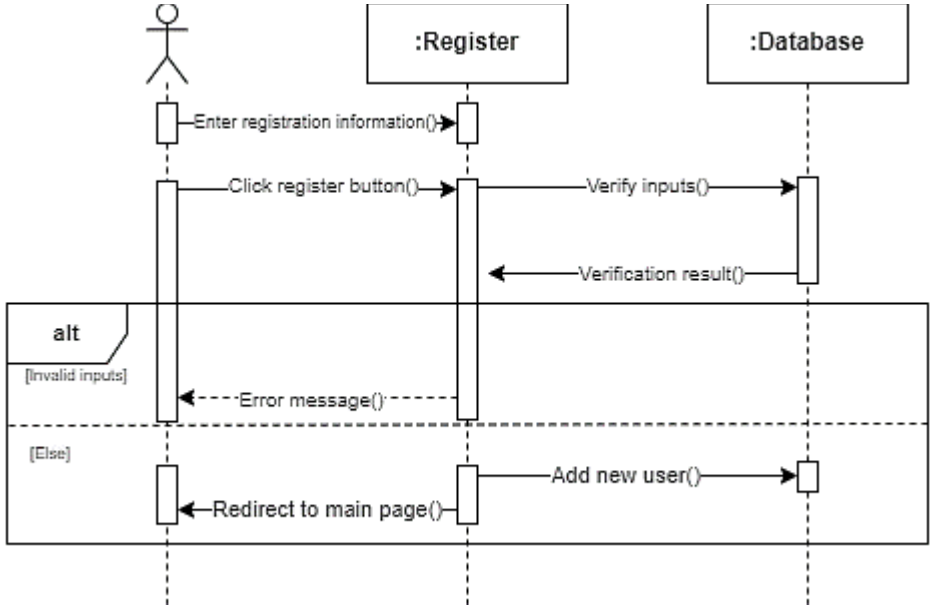

USER
Register
Cookies Order
Receipt
Checkout/Payment
Cookies Online Shopping
Cookies Order
Reports
Respond
ADMIN

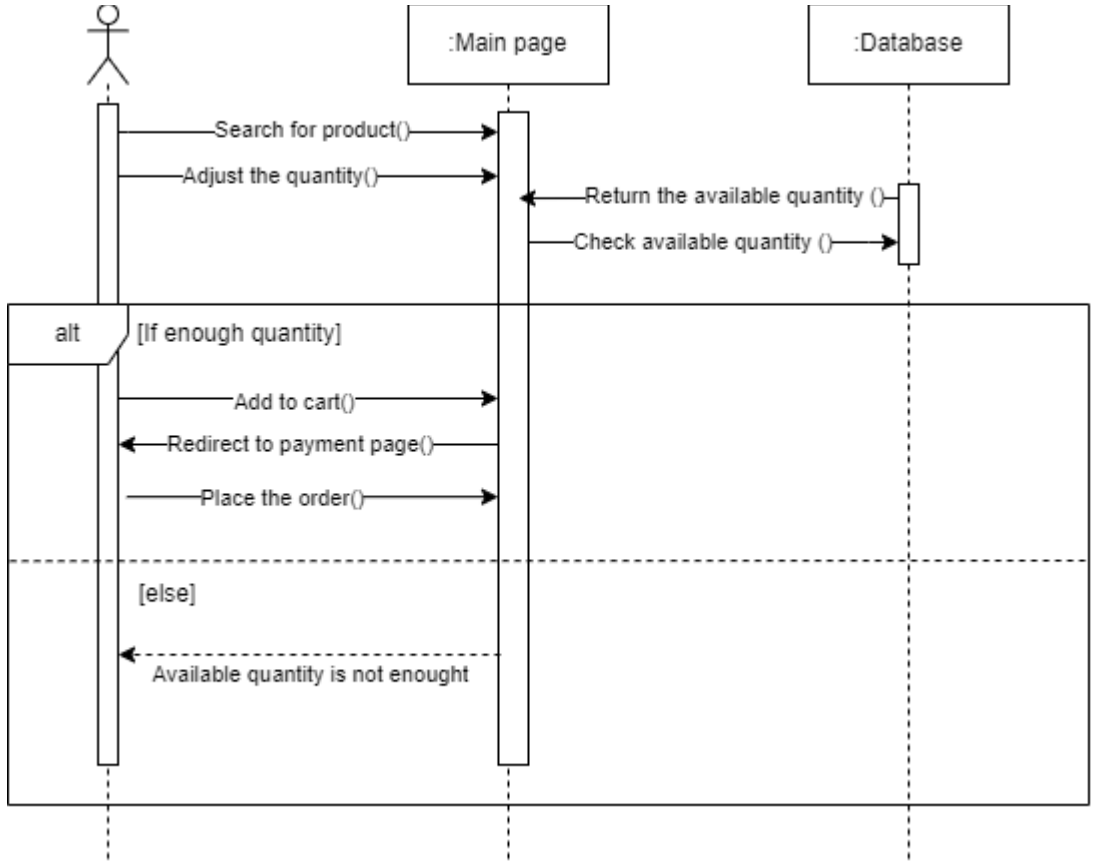
Sequence diagrams:

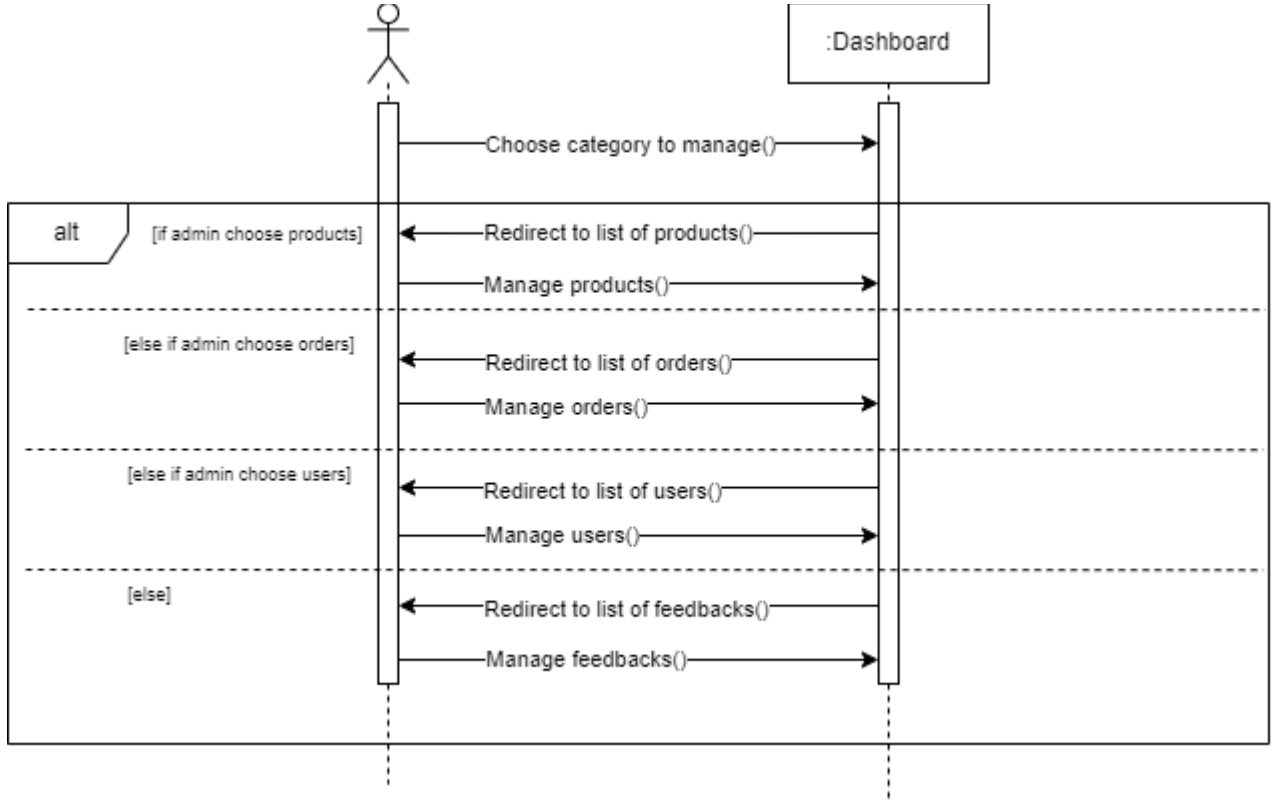
User: Login , registration

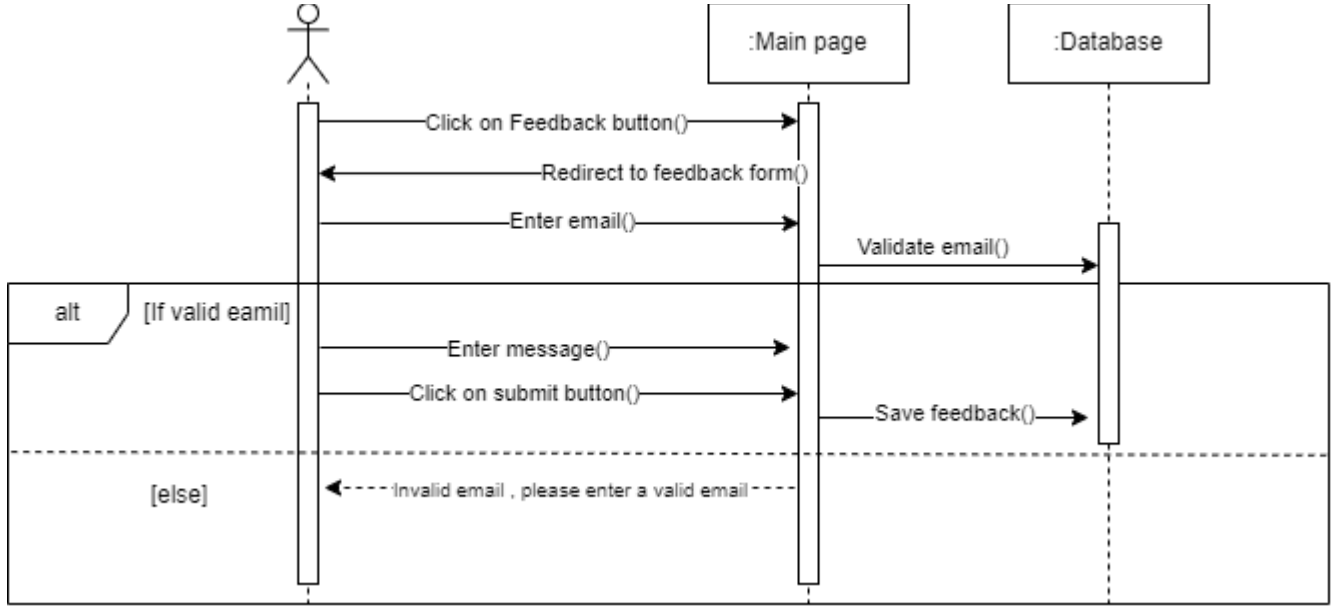
Customer: place an order, send feedback, dashboard management




:Register
:Database
Enter registration information()
Click register button()
Verify inputs()
Verification result()
alt
[Invalid inputs]
Error message()
[Else]
Add new user()
Redirect to main page()


:Main page
:Database
Search for product()
Adjust the quantity()
Return the available quantity ()
Check available quantity ()
alt
[If enough quantity]
Add to cart()
Redirect to payment page()
Place the order()
[else]
Available quantity is not enought


:Dashboard
Choose category to manage()
alt
[if admin choose products]
Redirect to list of products()
Manage products()
[else if admin choose orders]
Redirect to list of orders()
Manage orders()
[else if admin choose users]
Redirect to list of users()
Manage users()
[else]
Redirect to list of feedbacks()
Manage feedbacks()


:Main page
:Database
Click on Feedback button()
Redirect to feedback form()
Enter email()
Validate email()
alt
[If valid eamil]
Enter message()
Click on submit button()
Save feedback()
[else]
Invalid email , please enter a valid email

OCL:

context User
inv: self.age >= 18 -- Users must be adults
inv: self.address->notEmpty() -- Users must have a valid address
pre: self.password size >= 8 --Password must be strong
inv: self.email -> unique & not null

context Product
inv: self.price > 0 -- Product price must be positive
inv: self.quantityAvailable >= 0 -- Available quantity must be non-negative
inv: self.nameOfProduct -> not null

context ShoppingCart
inv: self.items->notEmpty() -- Shopping cart must not be empty
post: self.totalAmount >= 0 -- Total amount must be non-negative

```
context Order
inv: self.items->notEmpty() -- Order must contain at least one item
inv: self.status = 'Placed' or self.status = 'Shipped' or self.status = 'Delivered' -- Order status must be valid

context Payment
pre: self.amount >= 0 -- Payment amount must be non-negative
post: self.status = 'Pending' or self.status = 'Completed' or self.status = 'Failed' -- Payment status must be valid

 context Customer
Constraint: Customer email must be unique
 inv: Customer.allInstances()->select(c | c <> self)->forAll(c | c.email <> self.email)

context Admin
inv: self.role = 'SuperAdmin' or self.role = 'RegularAdmin'

context Category
inv: self.products->notEmpty() implies self.description->notEmpty()
```