FINAL SEMESTER ASSESSMENT (FSA)
B.TECH. (CSE)
VI SEMESTER

UE18CS355 – OBJECT ORIENTED ANALYSIS AND DESIGN WITH SOFTWARE ENGINEERING LABORATORY

PROJECT REPORT

ON

Amazon (Online Shopping)

SUBMITTED BY

NAME	SRN
1) Sai Eashwar K S	PES1201801910
2) Srinath M P	PES1201801946
3) Abhay Pratap Singh	PES1201802019

JANUARY – MAY 2021

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
RR CAMPUS,
BENGALURU – 560100, KARNATAKA, INDIA

TABLE OF CONTENTS

Abstract

Amazon.com is the online shopping platform that simplifies buying and selling of products from the convenience of our houses. It provides many features such as searching for and viewing the products, adding items to cart, buying the items, making online payments etc. There are three types of users of Amazon.com :

- Customer
- Seller
- Administrator

Every user can perform certain allowed actions.

Coming to the project, we were first given a Software Requirements Specifications document from our client (peer). We went through the entire document and cleared a few doubts which we had. We then proceeded to make diagrams such as activity diagram, state diagram, class diagram etc which helped us in implementing the project.

We implemented 3 use cases :

- Product management
- Item selection and adding to cart
- Managing cart

We implemented the functionalities using Python and used Tkinter to generate basic User Interface. We did intensive testing which included Unit testing, Integration testing and System testing and fixed the many bugs that had arisen during coding.

CHAPTER-1

Software Requirements Specification

1. Introduction

Purpose

Amazon.com is a vast Internet-based enterprise, owned by Jeff Bezos. Amazon is the largest Internet company by revenue in the world. It mainly focuses on e-commerce(amazon.in), cloud computing(amazon web services), digital streaming(amazon prime), and artificial intelligence(recommendations).Retail remains Amazon's primary source of revenue, with online and physical stores accounting for the biggest share.With separate retail websites spread across different continents such as America, Europe, Asia and Australia. The tech giant is looking to simplify its online shopping system to make it user friendly and easy to use for everyone. The purpose of the Software Requirement Specification is to provide a clear, documented model of the requirements for the online shopping system. This document also provides top level use cases for a web customer making purchases online. It also describes nonfunctional requirements and other factors necessary to provide a complete and comprehensive description of the requirements for the software.

Intended Audience

This project is a prototype for the Online Shopping System. The system is worth using by an audience , who are interested in buying online products and benefit from facilities offered.
The intended readers of this document are the developers of the site, testers, website owners, managers and coordinators.

Product Scope

The scope of the project is to design and develop a system that allows users to perform some functionalities . The application to be created is an Online Shopping System. It will perform basic functions that will allow the Web customers to browse through the website to make purchases online. Top level use cases are Client Register, View Items and Make Purchase. It also includes financial transactions that call for other third party services like NetBanking. This system provides an easy solution for customers to buy the product without going to the shop and to pay virtually.

References

- IEEE 830-1998 standard for writing SRS documents.
- https://en.wikipedia.org/wiki/Amazon_(company)
- amazon.com
- https://www.altexsoft.com/blog/business/functional-and-non-functional-requirements-specification-and-types/

2. Overall Description

2.1 Product Perspective

The origin of the product The suggested framework is a solution for the online purchase/sale of goods.It is a web based system implementing client - server model. Customers must have an account to purchase the product. This application stores all the information in the database which can be retrieved whenever needed for further transactions.Product is intended to be a standalone product and should not depend on the availability of other software. It should run on both linux and windows based platforms.

2.2 Product Functions

- **Use Case Name: Login , Sign Up**
    - **Summary :** The login feature allows the user of the software to use the website as a registered user.

- **Preconditions:** create an account.
- **Triggers:** User selects the sign in button.

- **Use Case Name: Search items**
  - **Summary:** The search items allows the user of the software to find products on the website.
  - **Preconditions:** None.
  - **Triggers:** User clicks on the search box and type in the required item.

- **Use Case Name: Browse items**
  - **Summary:** The search items allows the user of the software to browse items in the catalog on the website.
  - **Preconditions:** None.
  - **Triggers:** User select the ‘category’ down button.

- **Use Case Name: Buy item now**
  - **Summary:** The buy item now feature allows the actor of the software to quickly order items using 1-Clicking ordering.
  - **Preconditions:** Actors need to 'Turn on 1-Click ordering for this browser' to use the feature.
  - **Triggers:** Actor selects the ‘Buy Now’ button.

- **Use Case Name: Add to shopping cart**
  - **Summary:** The add to shopping cart feature allows the actor of the software to create add items , products they wish to buy.
  - **Preconditions:** None.
  - **Triggers:** Actor clicks the ‘Add to Cart’ button.

- **Use Case Name: View shopping cart**

- **Summary:** The view shopping cart feature allows the actor of the software to view a list of items in the shopping cart.
- **Preconditions:** None.
- **Triggers:** Actor clicks the ‘edit your cart’ button.

- **Use Case Name: Proceed to checkout**
  - **Summary:** The proceed to checkout feature allows the actor of the software to make payment and complete order.
  - **Preconditions:** None.
  - **Triggers:** Actor click the proceed to checkout button.

- **Use Case Name: Make payment**
  - **Summary:** The make payment feature allows the actor of the software to pay for items in the shopping cart.

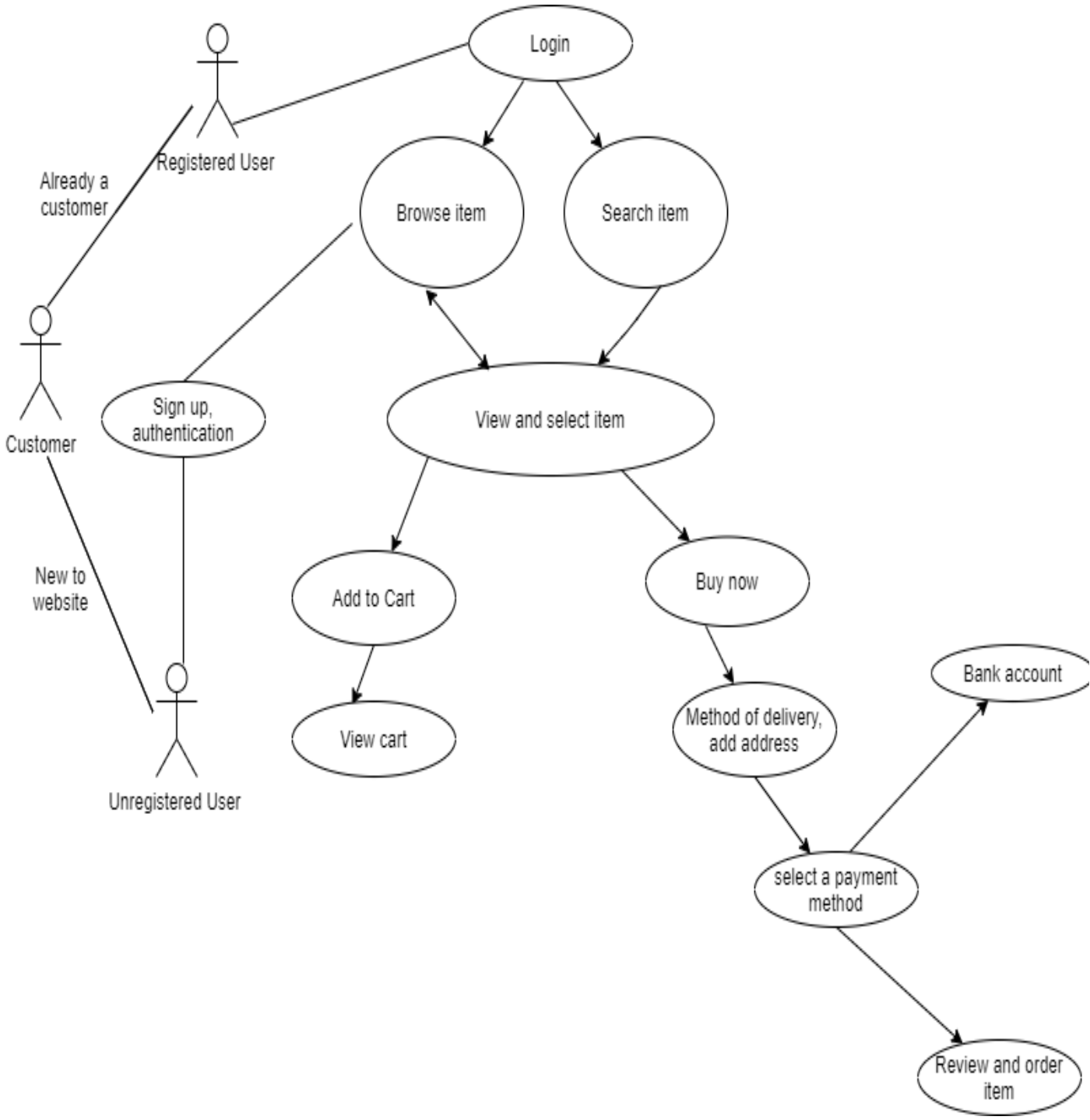


Fig. 1. Product functions

2.3 User Classes and Characteristics

The users of the online shopping-cart application, based on their roles, are customers (users) and the administrator (owner). These users are identified based on their experience and technical expertise.

1. Admin:

The administrator is the owner of this online shopping-cart application. One must have a basic understanding of computers and the internet as well as prior knowledge for operating the eclipse and Java programming languages. The administrator is responsible for maintaining all the training documents required for the system.
The administrator can perform the following functions:

- Assign or change the price of the items, update the items in the list, and delete the items.
- Assign sales tax for different states at the time of checkout.
- View the history of the customers who purchased the items.

2. Registered customers:

The users of this online shopping-cart application are all customers who would shop to test the application. These users are anyone with shopping experience and the know-how to browse through a shopping-cart application.The users should be able to perform the following functions using this system:

- View, browse, and select a category on the home page.
- View, add, and update items in the cart.
- Delete items from the cart.
- Check out the items from the application or continue shopping.
- Sign-on/login using a username and password.
- Place the order by completing the order form.

3. Unregistered customers

Unregistered users can only search, browse, view the items, but cannot purchase or buy an item as their entry is not made in the database.

2.4 Operating Environment

The software that is designed is a Distributed database system and client/server system.
The hardware, software and technology used should have following specifications:
• Ability to create a new account
• Mobile sms services for convenience.
• Ability to validate users.

Server side (Back end)	Operating system : Windows DBMS : PostgreSQL Web server : Apache
Customer side (Front end)	Operating system : Windows , which supports networking. Browser : Google Chrome, Firefox

2.5 Design and Implementation Constraints

Internet connection is required to browse through the website.
This system will work on client-server architecture. It will require an internet server and which will be able to run PHP applications. The system should support some commonly used browsers such as Mozilla Firefox, chrome etc.

2.6 Assumptions and Dependencies

OSS requires a back-end database server for storing details of a registered customer which includes , username, password, address, phone number, and other data. The system must be able to respond to the database server within reasonable time.
This system will use third-party software, and it is assumed that system users are familiar with the software.

3. External Interface Requirements

3.1 User Interfaces

3.1.1 : Customers

Registration :

If a customer wants to buy the product then he/she must be registered, an unregistered user can't go to the shopping cart.

Login :

Customer logins to the system by entering valid user id and password for the shopping.

Changes to Cart :

Changes to cart means the customer after login or registration can make order or cancel order of the product from the shopping cart.

Payment :

For customers there are many types of secure billing will be prepaid as debit or credit card, post paid as after shipping, check or bank draft.

Proceed to checkout :

Select this option to get to the payment and checkout use case

Enter method of delivery :

Used by customer to specify where item will be delivered

Make payment :

Make payment for the item in the shopping cart

Logout :

After the payment or surf the product the customer will logged out.

Giving Feedback :

Can give feedback to the 24X7 Customer Care Service center about their impression for the site and services.

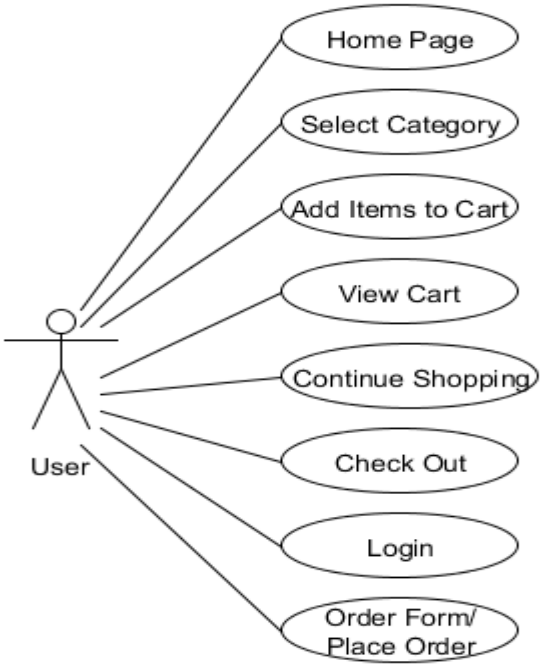


Fig. 2. Online Shopping-Cart System Use-Case Diagram: User.

3.1.2: Administrator

Database Management:- Control the database and keep track of all records of customers and employee details.

Contact and Giving Permission to Vendors:- Contact with the vendors and give permission to sell their product under the site after testing the product's quality.

View all details and make modifications:- View the details of all employees and customers and control the whole site.

Add/remove/update item category:- can add new items or remove the existing items .

shipping order:- can provide shipment services to customers.

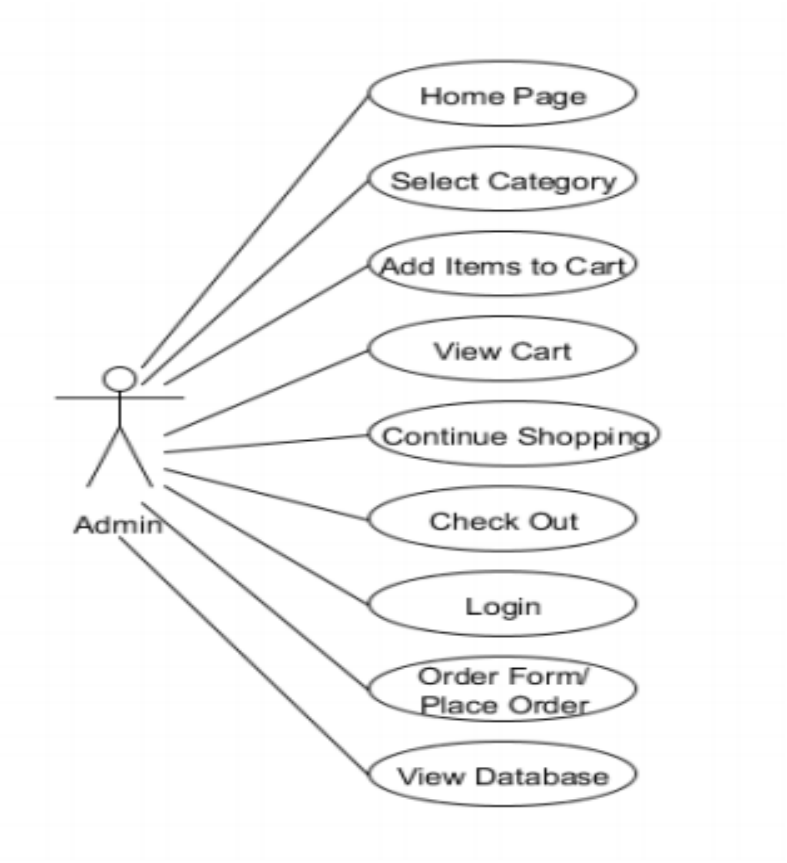


Fig. 3. Online Shopping System Use-Case Diagram : Administrator

3.1.3: Visitors

Visiting the Site: Can only visit the site without registration.

create a new account: users can create a new account if required.

3.2 Software Interfaces

1) Operating System: Unix, Linux, Mac, Windows, etc.

2) Development tool: PHP: Hypertext Preprocessor, JavaScript, Ajax,java development tool kit etc.

3) Database: MySQL.

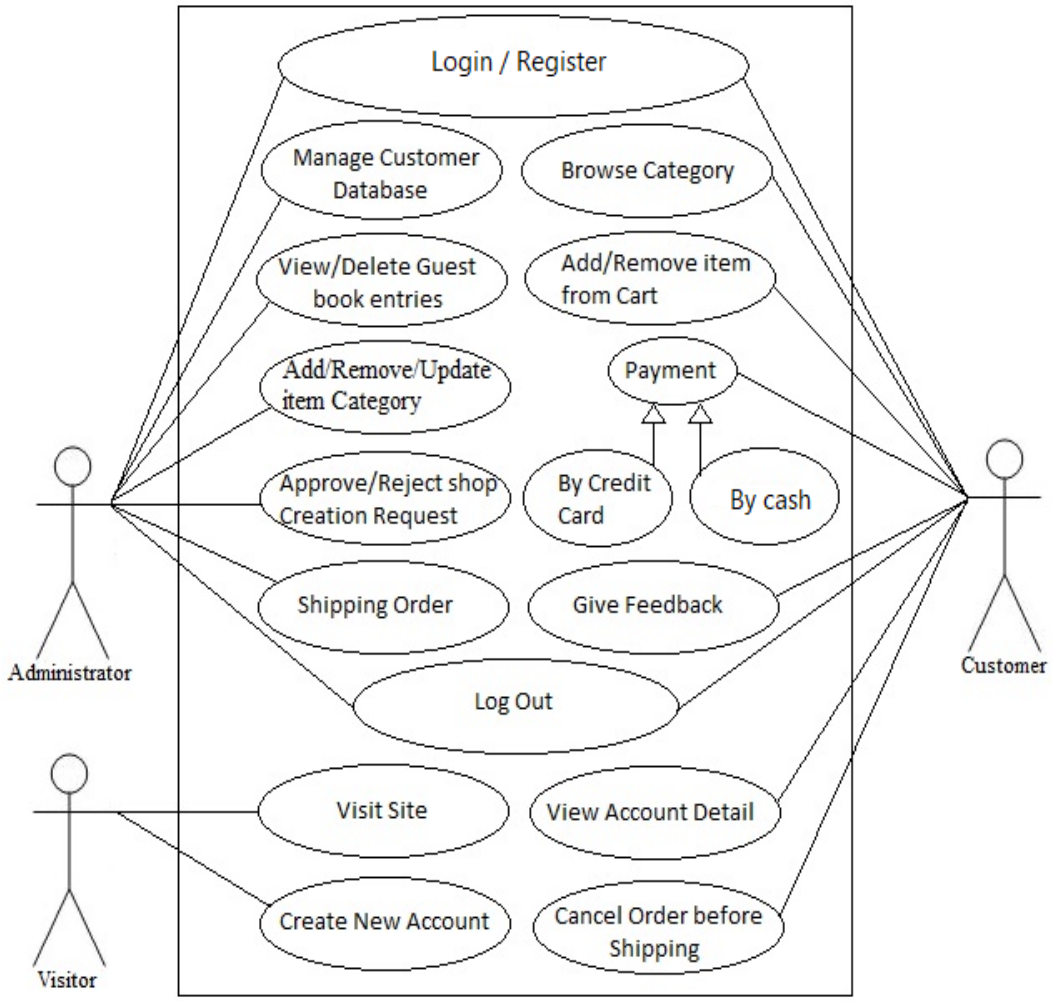
3.3 Communications Interfaces

Users need a web browser to interact with the system.The system shall use the HTTPS protocol for communication over the internet and for the intranet communication will be through TCP/IP protocol suite. The SSL Certificate for licensing purposes.

3.4 Hardware Interface

The System must run over the internet, all the hardware shall be required to connect to the internet will be a hardware interface for the system. As for e.g. Modem, WAN – LAN, Ethernet Cross-Cable

4. Analysis Models



Use case diagram for Customer, Visitor and Administrator.

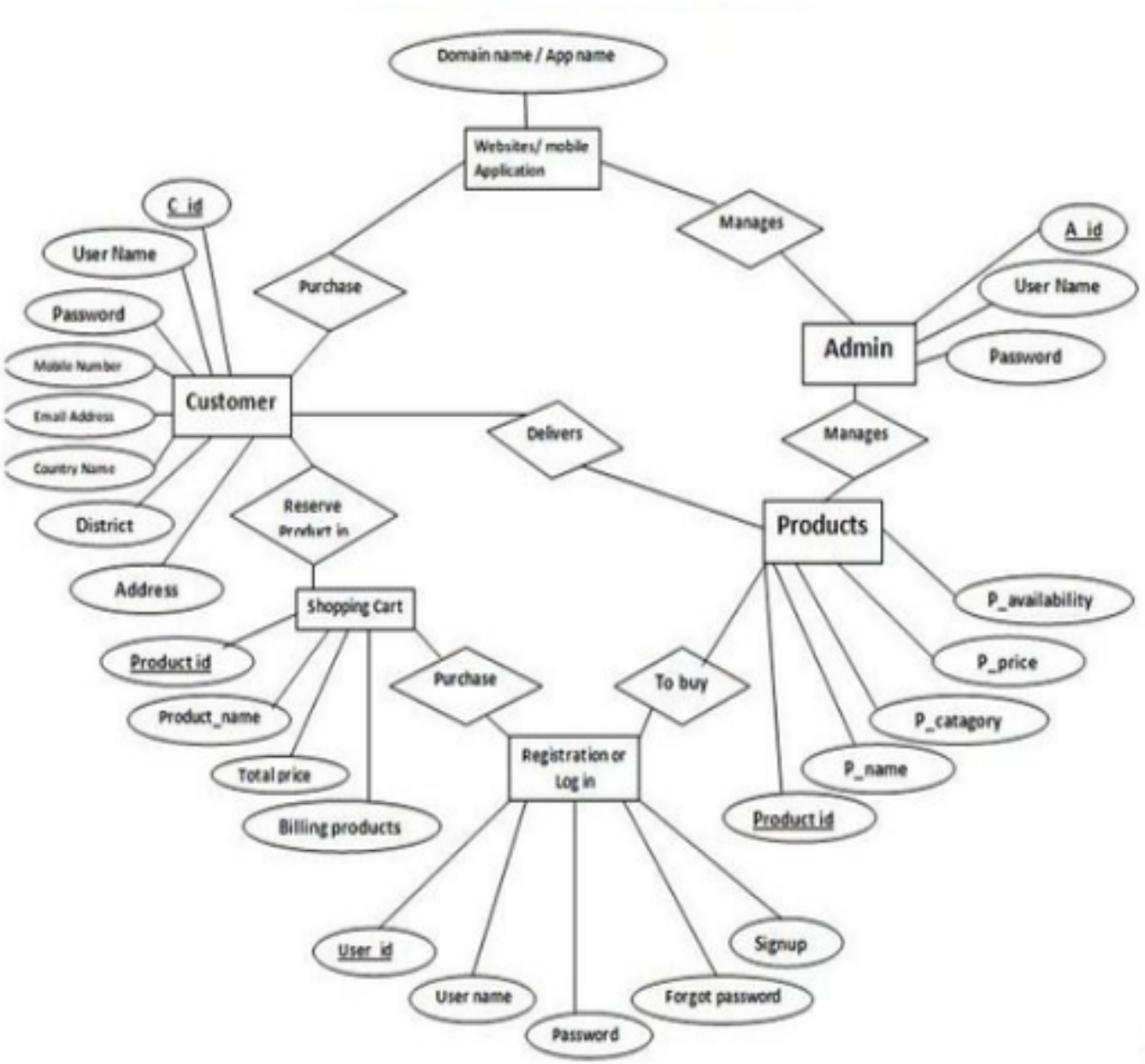


Fig:-ER Diagram

5. System Features

5.1 System feature 1: Registration[Priority HIGH]

Description and Priority:

Customers have to be registered in order to buy any product, unregistered users can't go to the shopping cart.

Priority: HIGH

Stimulus/Response sequences:

When the registration page pops up, users need to enter valid credentials such as username, email id.phone number and then the user can set the password for their account. Once OTP is received the users can register themselves, they can now login and continue shopping further.

Alternate Flow:

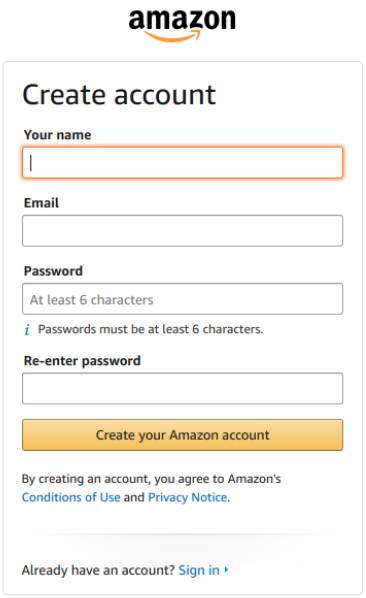
If user credentials already exists then it will be redirected to the login page

Business Rules:

Any user with the correct email id/phone number can register themselves when OTP is received for registration.

Functional Requirements:

1. The system should allow any user with proper email id and phone number to register
2. OTP will be sent to the provided email id/phone number
3. The system should be able to authenticate the users account with valid credentials
4. The system should be able to store the data in the database.



5.2 System Feature 2: Login [Priority HIGH]

Description and Priority:

After registration, customers should login to the system by entering valid user id(email id/phone number) and password for further shopping.Users can login only if credentials are valid.The database maintains these login credentials for authentication purposes.

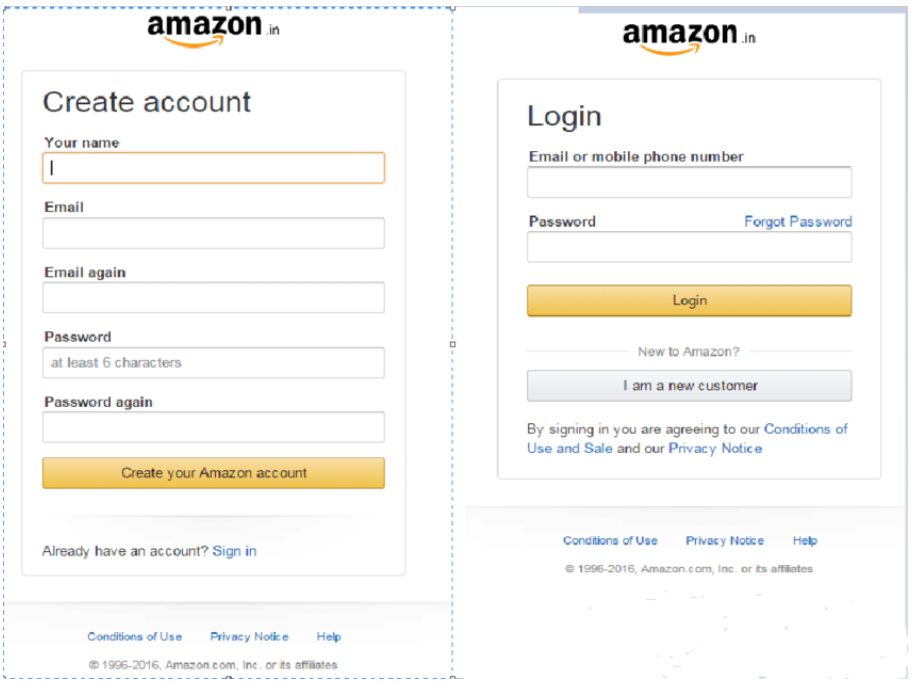
Priority: HIGH

Stimulus/Response sequences:

Once the login credentials are valid, users will be redirected to the home page, where they can continue further shopping.Mainly email id/phone number and password are the valid credentials.

Functional Requirements:

1. The system should allow user to login if the credentials entered are valid
2. The system should be able to authenticate users with valid credentials



5.3 System Feature 3: Shopping Cart [PRIORITY MEDIUM]

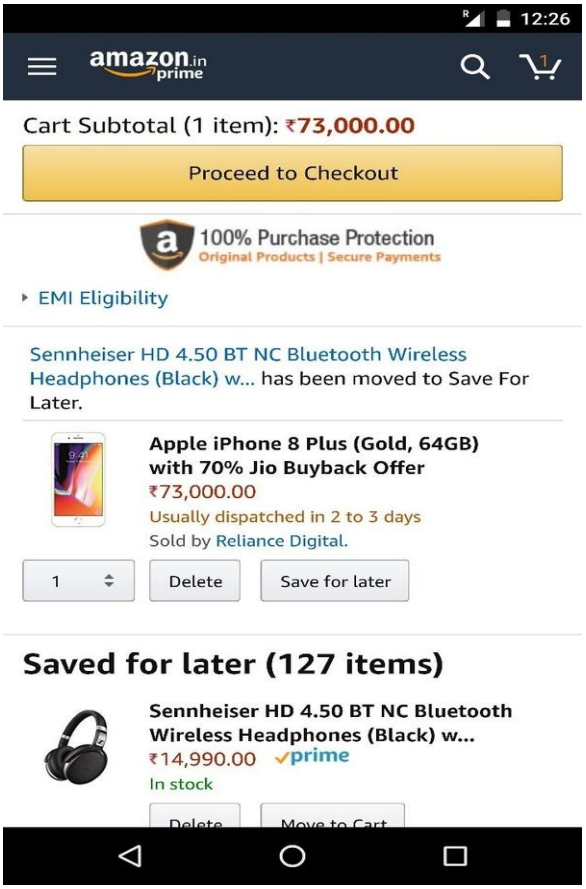
Description and Priority:

Changes to cart means the customer after login or registration can make order or cancel order of the product from the shopping cart.
Priority: Medium

Stimulus/Response sequences:

If the user is interested in buying any product then they can add it to the shopping cart.
If the user wants to buy that particular product, then it will be redirected to the payment section,
If not then it will be in the cart, and at any times the user can purchase from the cart.
Functional Requirements:

1. The system should maintain a database for the shopping cart.
2. The system should allow the user to select the required quantity of the product.
3. The system should allow the user to delete the particular product from the cart.
4. The system should allow the user to save the product for future shopping/buying.



5.4 System Feature 4: Payment [PRIORITY: HIGH]

Description and Priority:

In this system we are dealing with the mode of payment by cash on delivery, credit/debit cards.

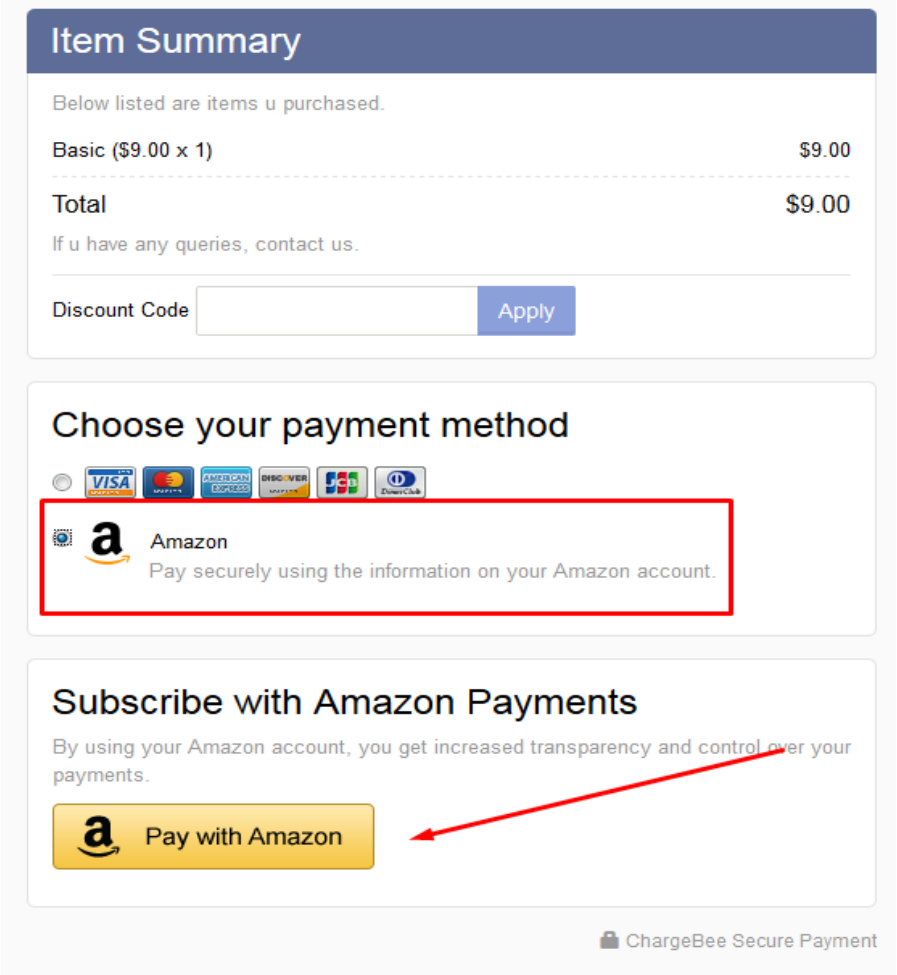
Priority: HIGH

Stimulus/Response sequences:

When the user will select a particular product from the shopping cart, the user will be redirected to the payment page.This system will allow payment through cash, credit/debit cards, Amazon Pay.

Functional Requirements:

1. The system should calculate total of the purchase by price * quantity
2. The system should provide various modes of payment.
3. If the payment is through online, then bank details/UPI details will be retrieved from the database(only if the user has provided the details during registration).



5.5 System Feature 5: Logout [PRIORITY: MEDIUM]

Description and Priority:

After ordering or surfing for the product, customers can logout.

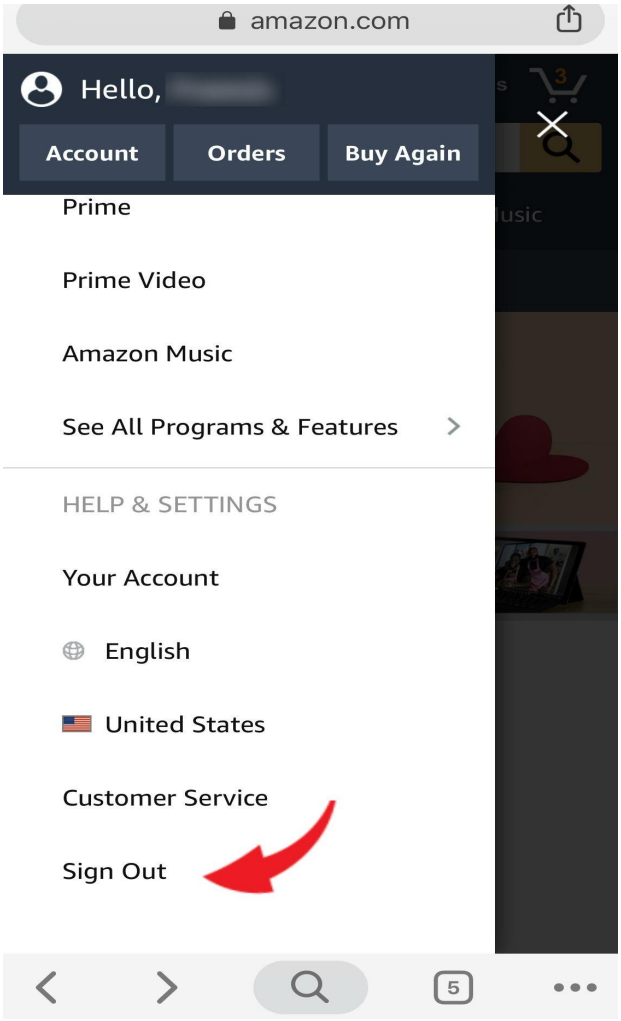
Priority: Medium

Stimulus/Response sequences:

After surfing through the products, users can decide whether to logout or not.

Functional Requirements:

1. The system should maintain a database of the user.
2. After logging out, the user must enter the login credentials again.
3. The system must maintain the database for user activities(Shopping cart, Wishlist etc)



5.6 System Feature 6: Report Generation [PRIORITY HIGH]

Description and Priority:

After ordering for the product,the system will send one copy of the bill to the customer's Email-address and another one for the system database.

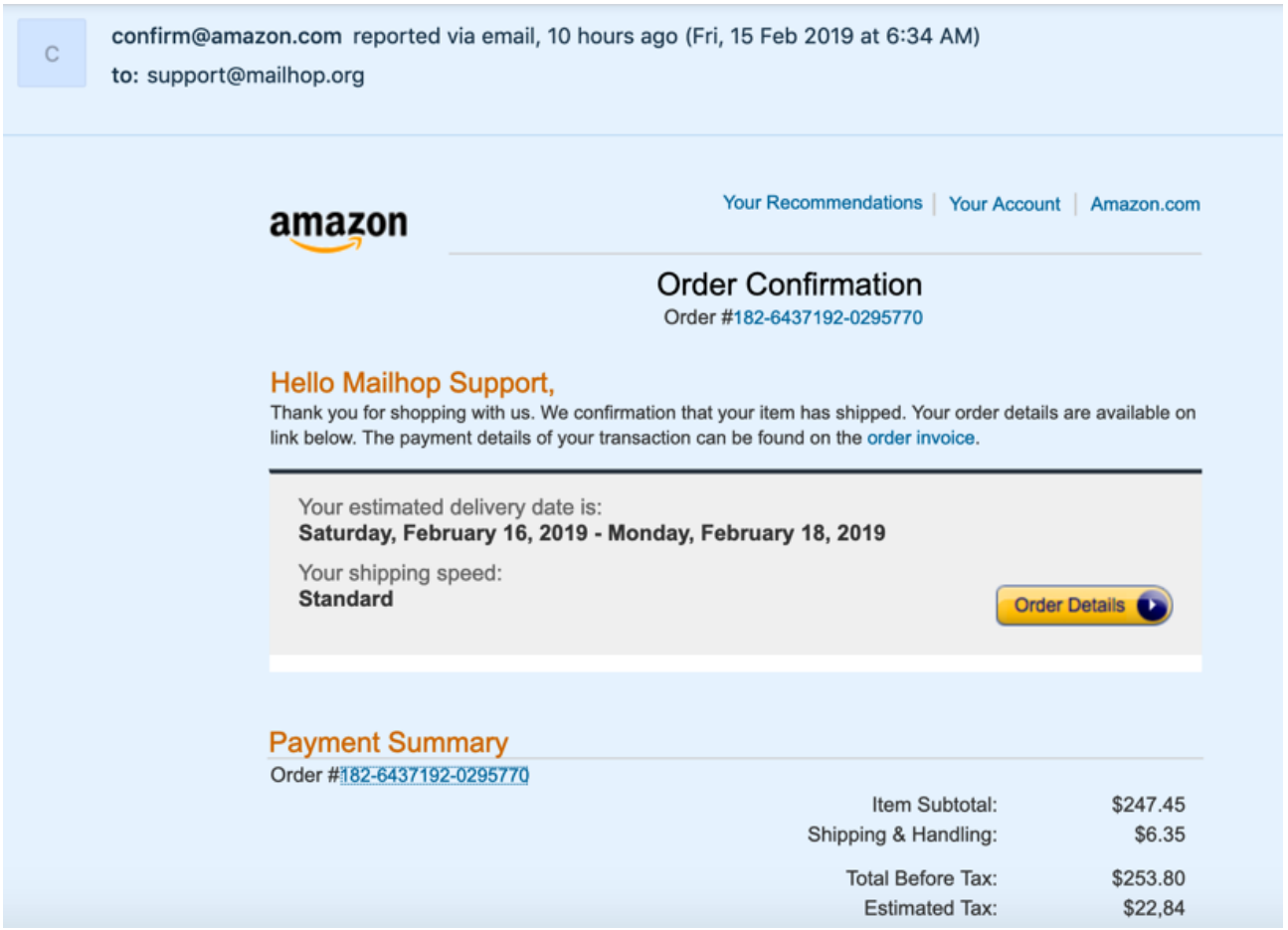
PRIORITY: HIGH

Stimulus/Response sequences:

When the order is successful the user will get an order confirmation report through mail or SMS. Also the same report will be stored in the database.

Functional Requirements:

1. The system will store a separate database for the order/transactional details.
2. The system should send a copy of order details for the user.



6. Other Nonfunctional Requirements

Following Non-Functional Requirements will be there in the insurance to the internet:
(i) Secure access to consumer's confidential data.
(ii) 24X7 availability.
(iii) Better component design to get better performance at peak time.
(iv) Flexible service based architecture will be highly desirable for future extension.

Non-Functional Requirements define system properties and constraints.
Various other Non-Functional Requirements are:

- **Security**
- **Reliability**
- **Maintainability**
- **Portability**
- **Extensibility**
- **Reusability**
- **Compatibility**
- **Resource Utilization**

6.1 Performance Requirements

There are no performance requirements in this system as it depends on the internet connection between the end users that is server request and response is totally based on the internet connection between the end users.

6.2 Safety Requirements

Safe in terms of security aspects.
Emergency customer service number in terms of any emergency complaints.
There must be an emergency customer care service which is 24*7 in service.
All the information regarding the customers must be kept safe in the database.

6.3 Security Requirements

This system uses SSL(Secured System Layer) in all transactions that include any confidential customer information.
After a period of inactivity, customers must be automatically logged out of the system.
The system should not leave any cookies on the customer's computer containing the user’s password.
Only authenticated administrators will be able to access the system’s back-end.
While transferring the data over insecure channels like the internet, sensitive data must be encrypted.

6.4 Software Quality Attributes

6.4.1 Reliability

The system provides storage of all databases on redundant computers with automatic switchover.
The reliability of the overall program depends on the reliability of the separate components. The main pillar of reliability of the system is the backup of the database which is continuously maintained and updated to reflect the most recent changes.
Thus the overall stability of the system depends on the stability of the container and its underlying operating system.

6.4.2 Availability

The system should be available at all times, meaning the user can access it using a web browser, only restricted by the down time of the server on which the system runs. In case of a hardware failure or database corruption, a replacement page will be shown. Also in case of a hardware failure or database corruption, backups of the database should be retrieved from the server and saved by the administrator. Then the service will be restarted. It means 24 X 7 availability.

6.4.3 Maintainability

A commercial database is used for maintaining the database and the application server takes care of the site. In case of a failure, a re-initialization of the program will be done. Also the software design is being done with modularity in mind so that maintainability can be done efficiently.

6.4.4 Portability

The application is HTML and scripting language based. So The end-user part is fully portable and any system using any web browser should be able to use the features of the system, including any hardware platform that is available or will be available in the future.
An end-user uses this system on any OS; either it is Windows or Linux.
The system shall run on PC, Laptops, and PDA etc.

6.5 Business Rules

There are mainly two types of users using this system, such as the user and an administrator.
Administrator has full permission of controlling the system whereas users can only utilize the services provided by the system.

6.6 Technical Issues:

This system will work on client-server architecture. It will require an internet server and which will be able to run PHP applications. The system should support some commonly used browsers such as IE, Mozilla Firefox,chrome etc.

Other Requirements

Data Base

The system must be able to use several data formats according to the data formats that are provided by the databases of different customers. A transaction should have all the properties of a database transaction (Atomicity, Consistency, Isolation, and Durability).

Appendix A: Glossary

Definitions :

- Register : The action of enrolling for something or of enrolling or employing someone.
- Login : An act of logging in to a computer, database, or system. A password or code used when logging in.
- Actor : An actor can be a user, or another software system that interfaces with the system being designed
- Feedback : Information about reactions to a product, a person's performance of a task, etc. which is used as a basis for improvement.
- Log Out : When someone who is using a computer system logs out or logs off, they finish using the system by typing a particular command.
- Website – A address that is connected to the internet to provide one or more web pages or other content.
- Online – Connected to a computer network or accessible by computer
- Browse – Reading superficially or at random

Abbreviations

Throughout this document the following abbreviations are used :

SRS	Software Requirement Specification
OSS	Online Shopping System
UPI	Unified Payments Interface

Appendix B: Field Layouts

Field	Length	Data Type	Description	Is Mandatory
First Name	32	String	First name of the customer	YES
Last Name	32	String	Last name of the customer	YES
Phone Number	10	Integer	Phone number of the customer	YES
Email id	32	Varchar	email -id of the customer	YES
Address	128	Varchar	Address (Location) of the customer to see product availability in the area	YES
Password	32	Varchar	Password to authenticate the user	YES
Account Number	16	Numeric	Based on the payment method	NO
Product ID	12	Varchar	ID of the product	YES
Price	Variable	Variable	Price of the product	YES

Appendix C: Requirement Traceability Matrix

Sl. No	Requirement ID	Brief Description of Requirement	Architecture Reference	Design Reference	Code File Reference	Test Case ID	System Test Case ID
1	RQ-1	Item Selection and Adding to Cart	1.1 (section 3)	1.3, 1.5.1, 1.5.2 (section 3)	amazon_1.py	1-9	10
2	RQ-2	Managing Cart	1.1 (section 3)	1.3, 1.7.1, 1.7.2 (section 3)	amazon_1.py	11-19	20
3	RQ-3	Product Management	1.1 (section 3)	1.3, 1.6.1, 1.6.2 (section 3)	amazon_1.py	21-31	32

CHAPTER-2

Project Plan

1. Lifecycle

1.1 Prototype Model

In Software Engineering, Prototype methodology is a software development model in which a prototype is built, tested and then reworked when needed until an acceptable prototype is achieved. Prototype models should be used when the desired system needs to have a lot of interaction with the end users.

Typically, online systems, web interfaces have a very high amount of interaction with end users, and are best suited for Prototype models. Customer feedback is used to Develop or refine a prototype which is used for testing of prototype.

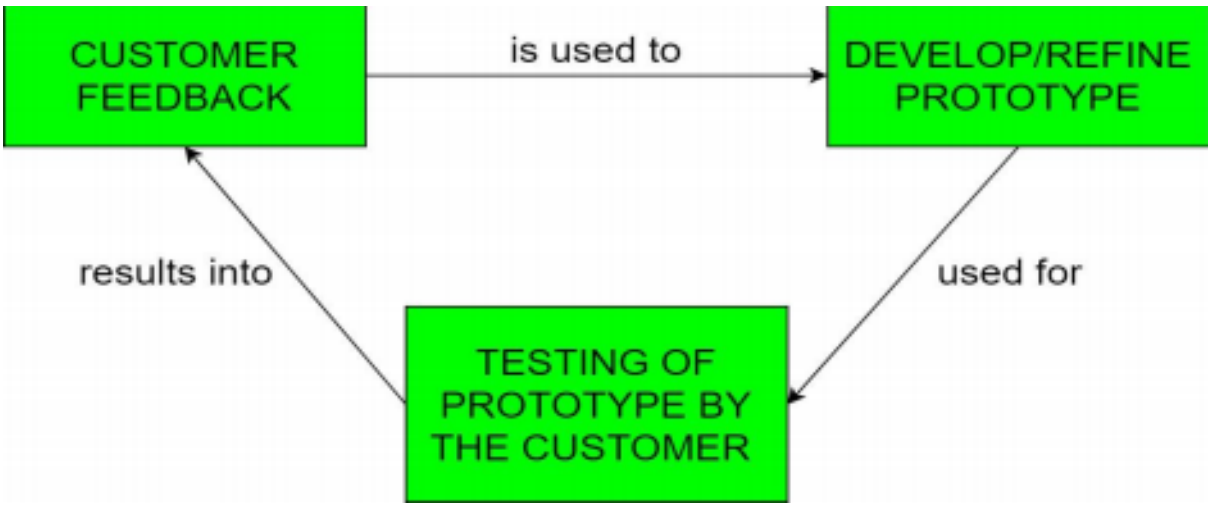


Fig 1.1

6 steps of the prototyping process are:

Step 1: Requirements gathering and analysis

Step 2: Quick design

Step 3: Build a Prototype

Step 4: Initial user evaluation

Step 5: Refining prototype

Step 6: Implement Product and Maintain

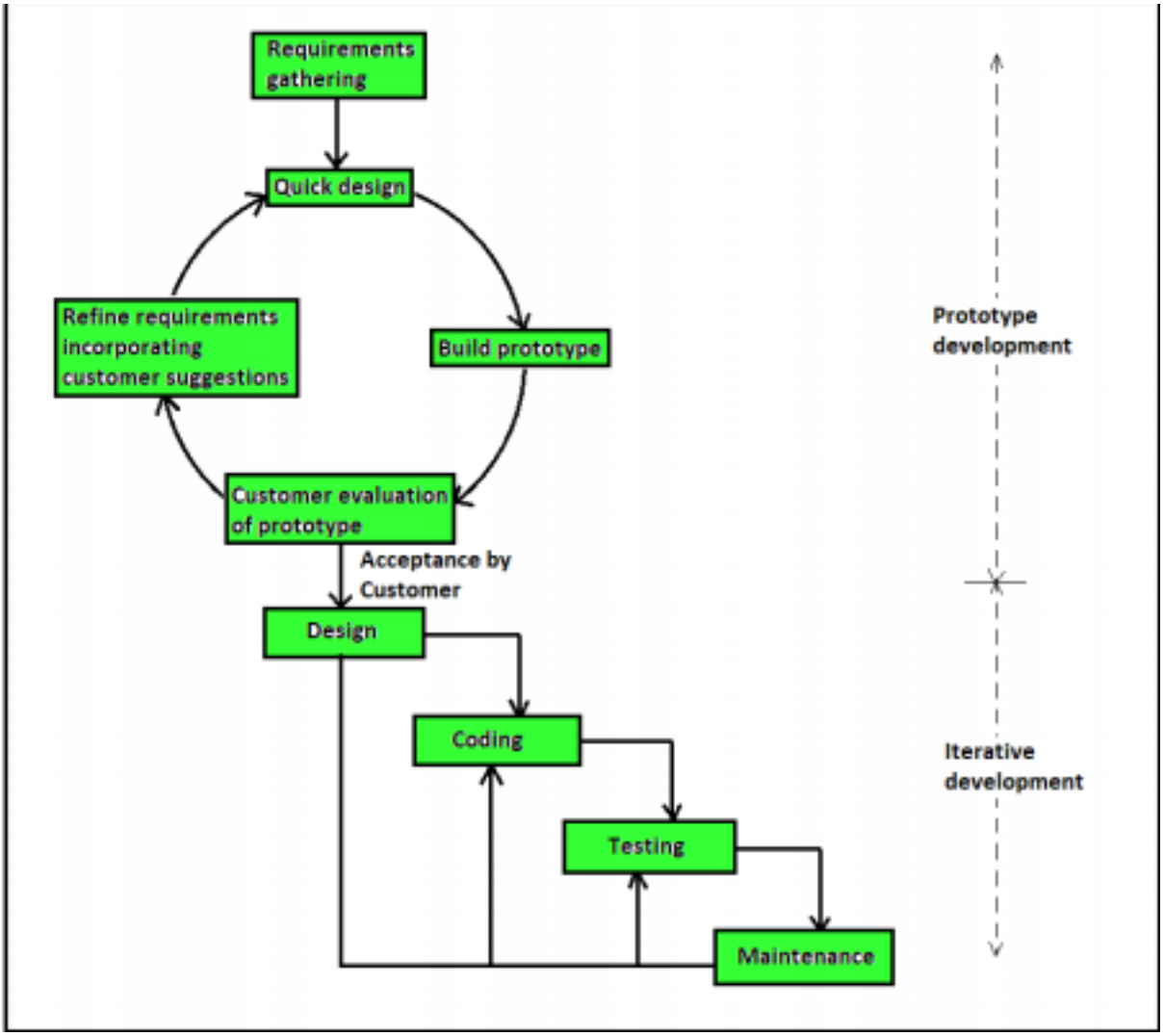


Fig 1.2

Prototype Development:

- It starts with an initial requirements gathering phase
- Design will be carried out and a prototype is built
- Then, this prototype will be submitted to the

customer for evaluation/feedback

- Based on the feedback given by the customer, requirements are considered and the prototype will be modified accordingly
- This cycle of obtaining feedback from the customer and modifying the prototype continues till the prototype is approved by the customer

Iterative Development:

- Once the prototype is approved by the customer, the actual software is developed using the iterative waterfall approach.
- Through these stages, experience will be gathered from developing the prototype and will be helpful for further development of software • That is, by constructing the prototype and submitting it for the customer evaluation, many requirements will be defined and fulfilled properly • Hence this model minimises the later change requests from customers and the associated redesign costs

Advantages:

- The customers get to see the partial product early in the life cycle. This ensures a greater level of customer satisfaction and comfort. • New requirements can be easily accommodated as there is scope for refinement.
- Missing functionalities can be easily figured out.
- Errors can be detected much earlier thereby saving a lot of effort and cost, besides enhancing the quality of the software.
- The developed prototype can be reused by the developer for more complicated projects in the

future.

- Flexibility in design.

Use:

- This model is used when the requirements of the product are not clearly understood or are unstable
- This model can be used when the requirements are changing quickly • For developing user interfaces, high technology software-intensive systems and systems with complex algorithms and interfaces. • In developing Graphical User Interface(GUI) part of the software, as the user can experiment with a working user interface and they can suggest any change if needed using the prototype provided
- Used when the exact technical solutions are unclear to the development team

There are several types of prototyping models, we are choosing Extreme Prototyping as:
Extreme prototyping method is mostly used for web development. It consists of three sequential phases.

1. Basic prototype with all the existing pages is present in the HTML format.

2. You can simulate a data process using a prototype services layer. 3. The services are implemented and integrated into the final prototype. Missing functionality can be identified, which helps to reduce the risk of failure as Prototyping is also considered as a risk reduction activity in SDLC.

2. Tools

2.1 Planning:

nTask:

ntask is the selected tool for planning. It is a tool for planning and managing a series of projects. It is really good at providing the big picture of all ongoing deliverables, tracking the timeline. List of tasks or a simple Gantt chart can be created to arrange in the modules of the project. The assignments can also be viewed as a grid. Team members can be able to log the time spent on a particular task.

Features:

Task Comments & File Attachments
Control and Manage Securely
Task & Project Management
Meeting Management
Timesheet Management
Team management

2.2 Design:

Bluefish:

It is a free software advanced text editor with a variety of tools for programming and website development as we are working on the project which requires arrangement of modules, website development. It supports coding languages including HTML, CSS, XML, PHP, C, C++, JavaScript etc.

It is available for many platforms, including Linux, macOS and windows. Bluefish is lightweight, fast and easy to learn, while providing many IDE features.

2.3 Version Control

Git:

It is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

It is usually used for coordinating work among team members collaboratively developing source code during software development.

Its goals include speed, data integrity which is most required, and support for distributed, non-linear workflows .

2.4 DEVELOPMENT:

2.4.1 Code Editors

- **Notepad ++**

  It is a free test and code editing software available on Windows and Ubuntu which supports about 27 programming languages and several file types along with interesting features like:

  - Syntax highlighting
  - Multiple views
  - Synchronized edits
  - Macros and more.

- **Visual Studio Code**

  Visual Studio Code is possibly the best JavaScript ide for Windows, Mac, and Linux. Not only does it support JavaScript, but it also supports Node.js, TypeScript, and it comes with a whole ecosystem of extensions for other languages including C++, C#, Python, PHP etc.

It provides great syntax highlighting and auto-complete with IntelliSense based on variable types, function definitions, and imported modules.

Useful features like:

- Debugging
- Syntax highlighting
- Smart code completion
- Snippets
- Cross platform availability and much more.

- **RJ TextEd**

RJ TextEd is high on our list fighting for the spot of the best IDE for JavaScript. It is a full-featured text and source editor with Unicode support and all in all, a great IDE for web development. It supports not only JavaScript, but also PHP, ASP, HTML and CSS.

Some of the most important features of this web development IDE include but are not limited to:

- Auto Completion
- Multi edit and multi select
- CSS and HTML wizards
- File explorer
- Text clips
- Code explorer
- Project manager

- Tools available like syntax editor, color picker, charmap

2.4.2 Frameworks

- **Bootstrap**

It is an open source CSS framework which assists in creation of responsive websites which consists of CSS and JS based design templates and other interface components.

- **Flask**

It is a micro web framework without requirement of particular tools or libraries. It is a 3rd party python library used to deploy python models onto web applications.

2.4.3 Backend

- **Apache**

It is a free and open source cross platform web server used to deliver web content over the internet.

- **MySQL**

It is an open source relational database management system, used to handle the product's database.

- **XAMPP**

It is a local development environment which is an open source and cross platform web server solution stack with integrated Apache and MySQL and compatible with PHP.

2.4.4 Programming Languages

- **HTML , CSS**

The primary language used for web development is HTML for the structure which shall be assisted with CSS for design and styling of the pages.

- **Javascript**

Javascript shall be used along with HTML for client side scripting to help make the web pages

more interactive.

- **PHP**

  PHP is the server side scripting language which is adopted to manage the dynamic content, databases, databases, session tracking and more by simply embedding in HTML.

2.5 Bug Tracking

- **Bugzilla**

  Bugzilla is an open-source Bug tracking tool. Bugzilla is a defect/bug tracking tool. Defect tracking systems allow developers and testers to track all the outstanding defects.

2.6 Testing

- **Selenium**

  The best free automation testing tools for web application testing. It is an open-source platform that is compatible with numerous browsers, operating systems, and programming languages. Selenium features detailed and advanced automation scripts, supports execution of parallel tests, and integrates other software testing tools. As the best web testing tool, The main thing is the Selenium Framework, it will help you to make code maintenance easy.

- **Postman**

  It is a Google Chrome app for interacting with HTTP APIs. It presents you with a friendly GUI for constructing requests and reading responses. The API testing tool is Postman. Postman offers a web version as well as a desktop app, and can also be used for testing API services.

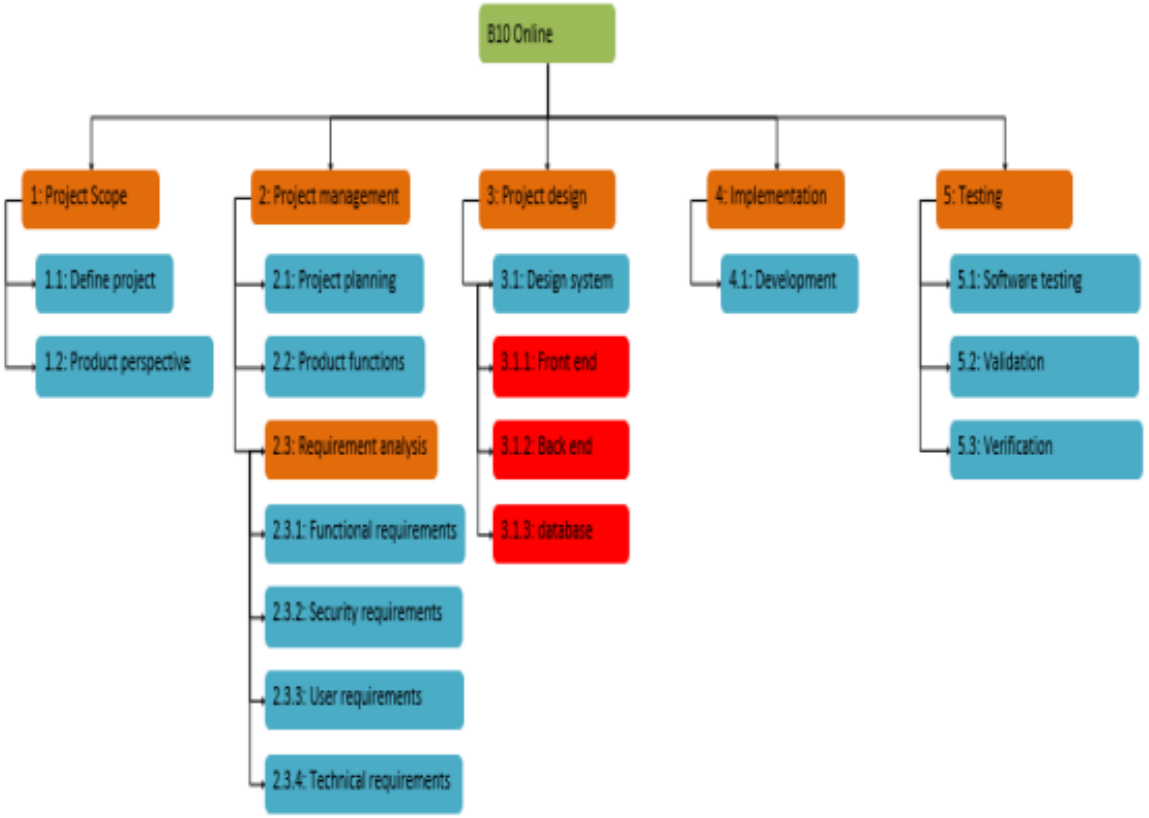
- **Pylot**

  Performance and Scalability testing of a web application can be well tested using Pylot. It generates concurrent HTTP requests and performs testing that is useful for analysis and system optimization. You can use a GUI or shell console to execute and monitor test suites.

- **NMap (Network Mapper)**

  A cross-platform web security scanner, written by Gordon Lyon (Fyodor) founder of hosts and services on a computer network. This scanner delivers correct packets to the target machine and examines the response coming from there

3. Work Breakdown Structure



4. Deliverables

Deliverable	Description	Build/Reuse
Software requirement document	SRS document is to provide a detailed overview of our software product, its parameters and goals. This document describes the project's target audience and its user interface, hardware and software requirements.	SRS document is written from scratch.
Project Plan	This document specifies how the objectives will be achieved, what resources will be needed and timelines for the completion.	Project plan document is written from scratch keeping the objectives to be delivered.

Architecture	Defines the structure, behaviour, and more views of the system.	Built from scratch in accordance with system requirements
High-level design (HLD)	Explains the architecture that would be used for developing the software product.	The architecture diagram provides an overview of an entire system, identifying the main components that would be developed for the product and their interfaces .
User Interface	Efficient functions are implemented, Which are easy for the user to interact with.	User interface is built from scratch.
Application interface	Includes Items Interface, View Cart Interface, Checkout (payment) Interface	The system will be implemented using some existing software to fasten the process.
Database	Using this, the administrator can access information of all users , purchases made by the users, which is updated for every order.	Database shall be built , which is easy to access, update , and authorized due to the security issues.

5. Effort Estimation

5.1 Cocomo

Cocomo (Constructive Cost Model) is a regression model based on LOC, i.e **number of Lines of Code**. It is a procedural cost estimate model for software projects and often used as a process of reliably predicting the various parameters associated with making a project such as size, effort, cost, time and quality.

The key parameters which define the quality of any software products, which are also an outcome of the Cocomo are primarily Effort & Schedule:

- **Effort:** Amount of labour that will be required to complete a task. It is measured in person-months units.
- **Schedule:** Simply means the amount of time required for the completion of the job, which is, of course, proportional to the effort put. It is measured in the units of time such as weeks, months.

Our project comes under organic type: A software project that is said to be an organic type if the team size required is adequately small, the problem is well understood and has been solved in the past and also the team members have a nominal experience regarding the problem.

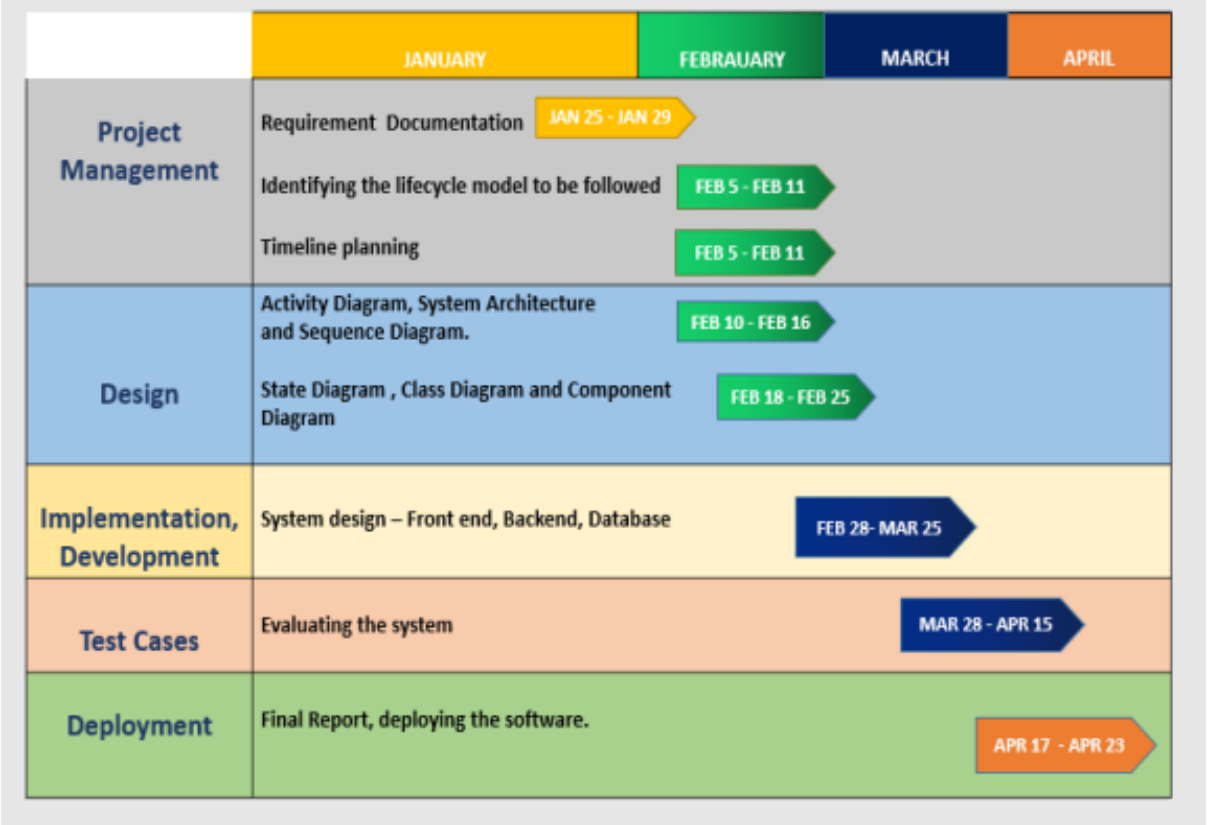
5.2 Formula:

Efforts= a*(KLOC)^b
Time = c*(effort)^d
For organic software projects, a = 2.4, b

= 1.05, c = 2.5, d = 0.38. **5.3 Estimation**

Assuming KLOC for our project is 2:
Effort= 4.97 person-months
Time= 4.59 months
Person required= Effort/ Time = 1.08= 1 person

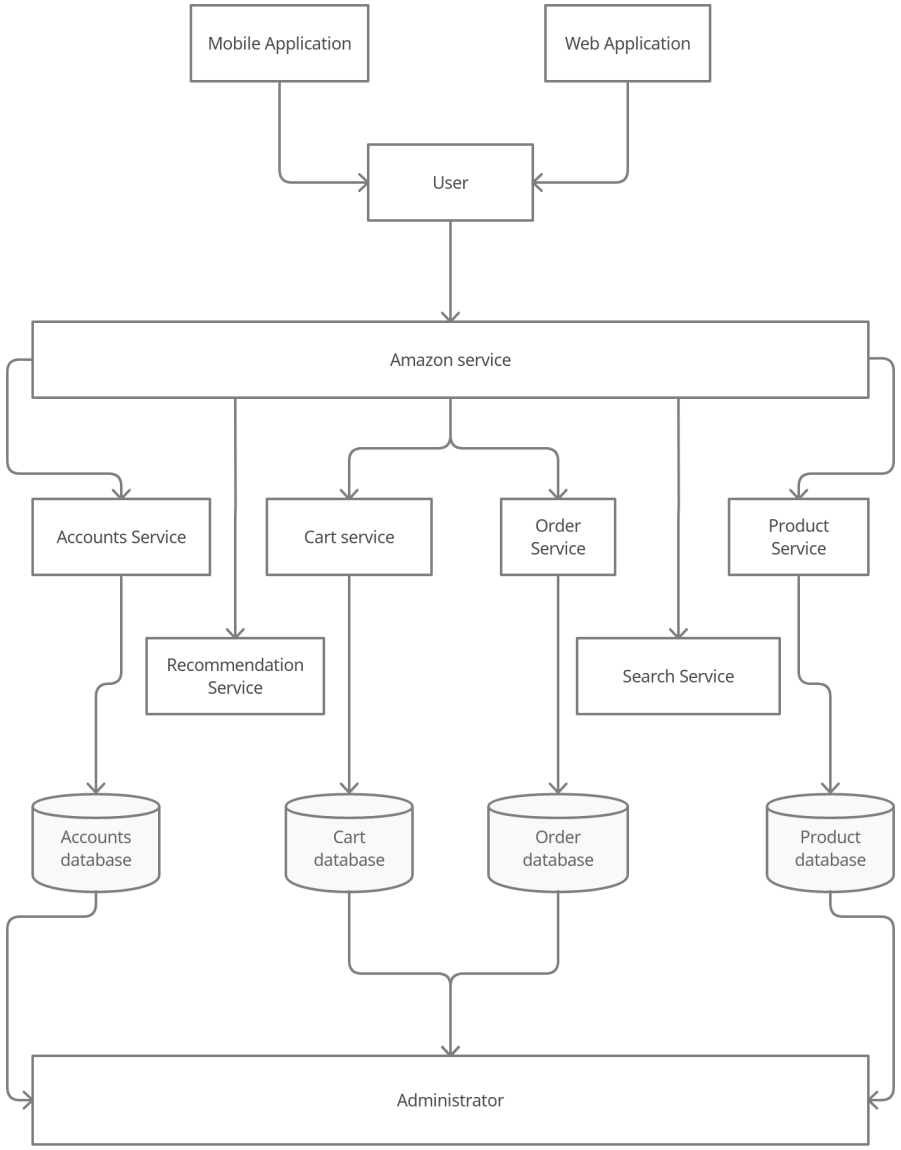
6. Gantt chart


JANUARY
FEBRAUARY
MARCH
APRIL
Project Management
Requirement Documentation
JAN 25 - JAN 29
Identifying the lifecycle model to be followed
FEB 5 - FEB 11
Timeline planning
FEB 5 - FEB 11
Design
Activity Diagram, System Architecture and Sequence Diagram.
FEB 10 - FEB 16
State Diagram , Class Diagram and Component Diagram
FEB 18 - FEB 25
Implementation, Development
System design – Front end, Backend, Database
FEB 28- MAR 25
Test Cases
Evaluating the system
MAR 28 - APR 15
Deployment
Final Report, deploying the software.
APR 17 - APR 23

CHAPTER-3

Design Diagrams

1.1 Architecture Diagram

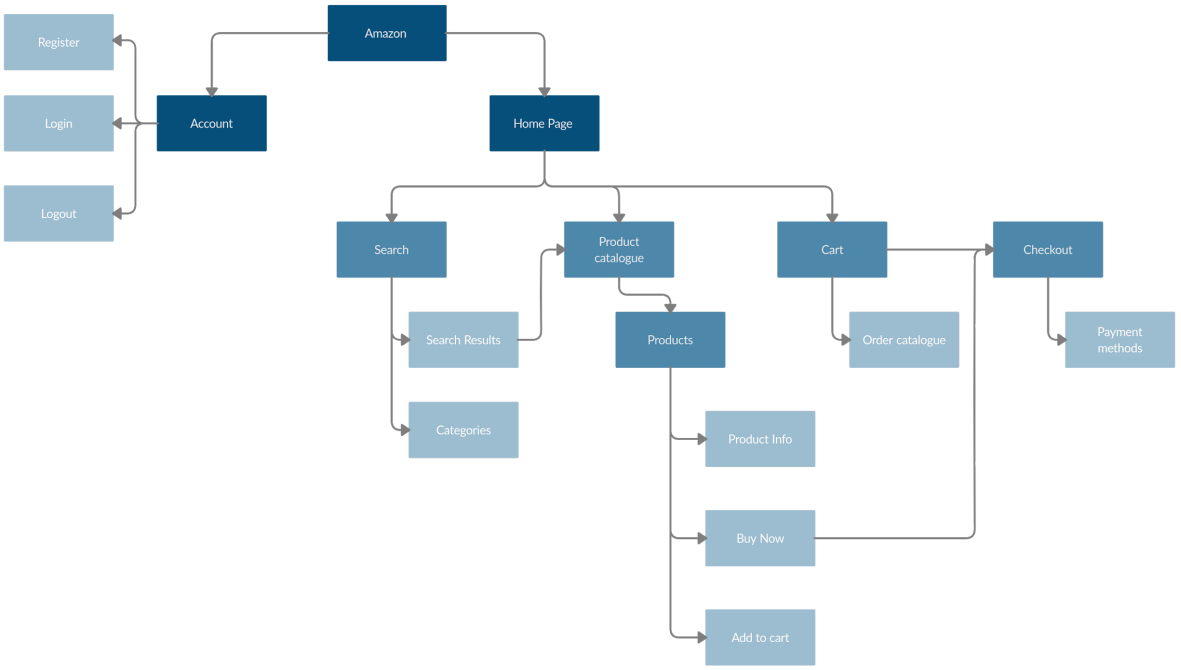


Micro-service Architecture

The architecture diagram as the name says describes the constructional architecture of the entire application at a rudimentary level. The primary elements of the diagram are user , and the amazon application. It describes the interaction among various micro services that make up the backend of the application mainly account service, cart service, order service, product service, recommendation service, search service, databases for account, order and

product . The diagram also explains the access hierarchy of the various micro services and their components for example the administrator has access to the databases for account, order, cart and product while the user does not. The structure is decided based on the understanding

1.2 INTERFACE DIAGRAM

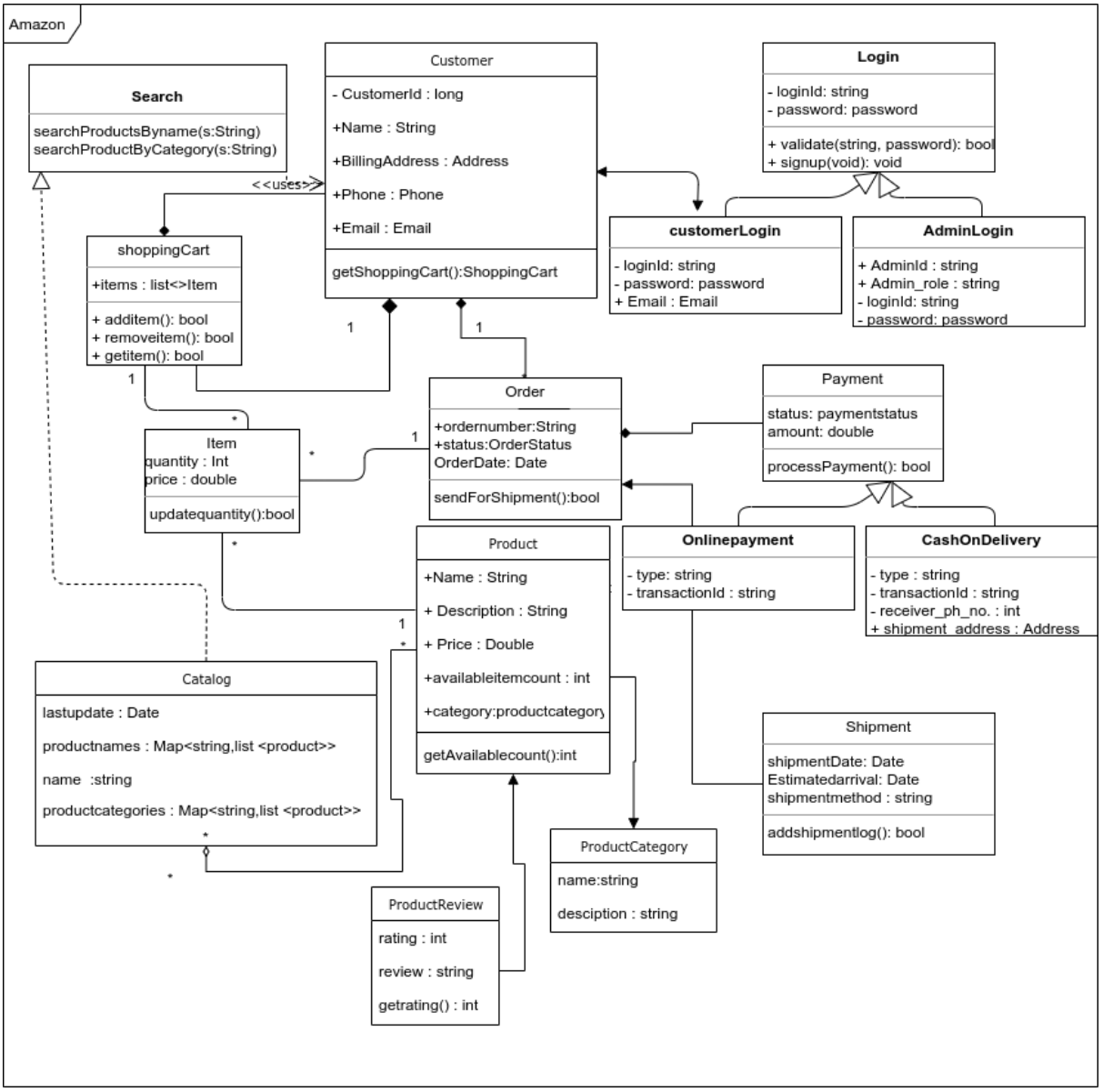


The interface diagram describes the various structural elements that make up the application and how they interact amongst themselves and also their dependencies . It describes which implementational elements can interact with the corresponding elements based on the inferences from the software requirements specifications. The interface diagram helps us in the planning phase to decide the timeline of the development activities based on dependency as an element can't be implemented without its dependencies being successfully implemented and test and is ready to be deployed . It also helps in understanding different hierarchies that exist and their distinctions . The diagram helps avoid repetitive access and understanding of the software requirements specifications document.

1.3 CLASS DIAGRAM

A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram in software engineering that depicts the structure of a system by displaying the system's classes, attributes, operations (or methods), and relationships among objects. The classes were selected after careful selection and

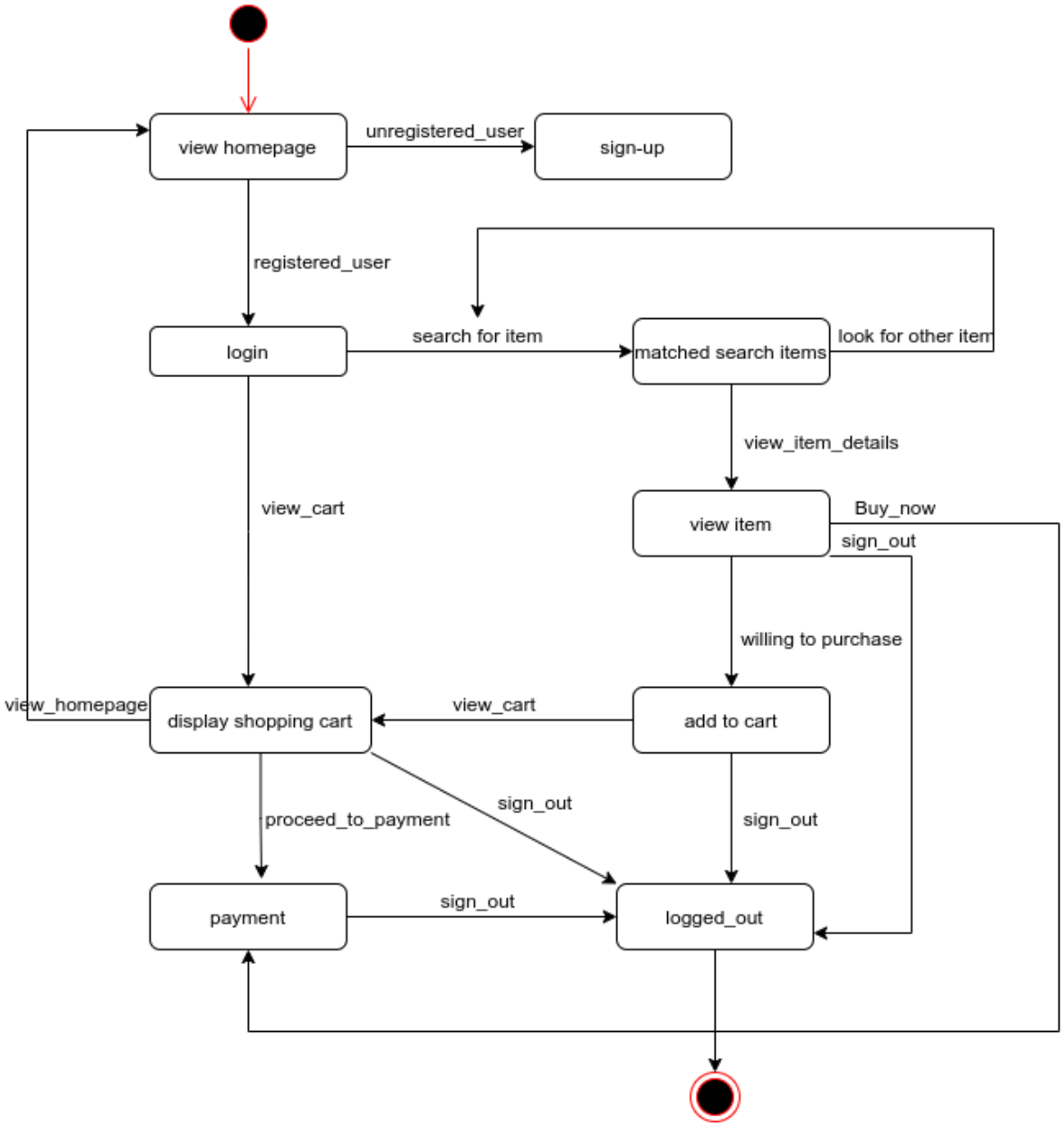
elimination of classes that were inappropriate and a data dictionary was made , then associations were identified which were subject to filtering . The main classes were selected and their respective attributes were identified. The main classes along with their attributes and their relationships with other classes are depicted below using UML notations and conventions.



1.4 State Diagram

A state diagram is used to represent the condition of the system or part of the system at finite instances of time. It's a behavioral diagram and it represents the behavior using finite state transitions. State diagrams are also referred to as State machines and State-chart Diagrams. The diagram below describes the major states and their transitions . The states shown here cover all the transitions that may occur while realising the use cases and take into considerations the activities and interactions that happen among the states. The state diagram shown below

describes all the states that may be active from the beginning of the home page with user logging in to the user making payment and logging out. Multiple paths of flow can be taken based on the states that a user lands on based on their choice.

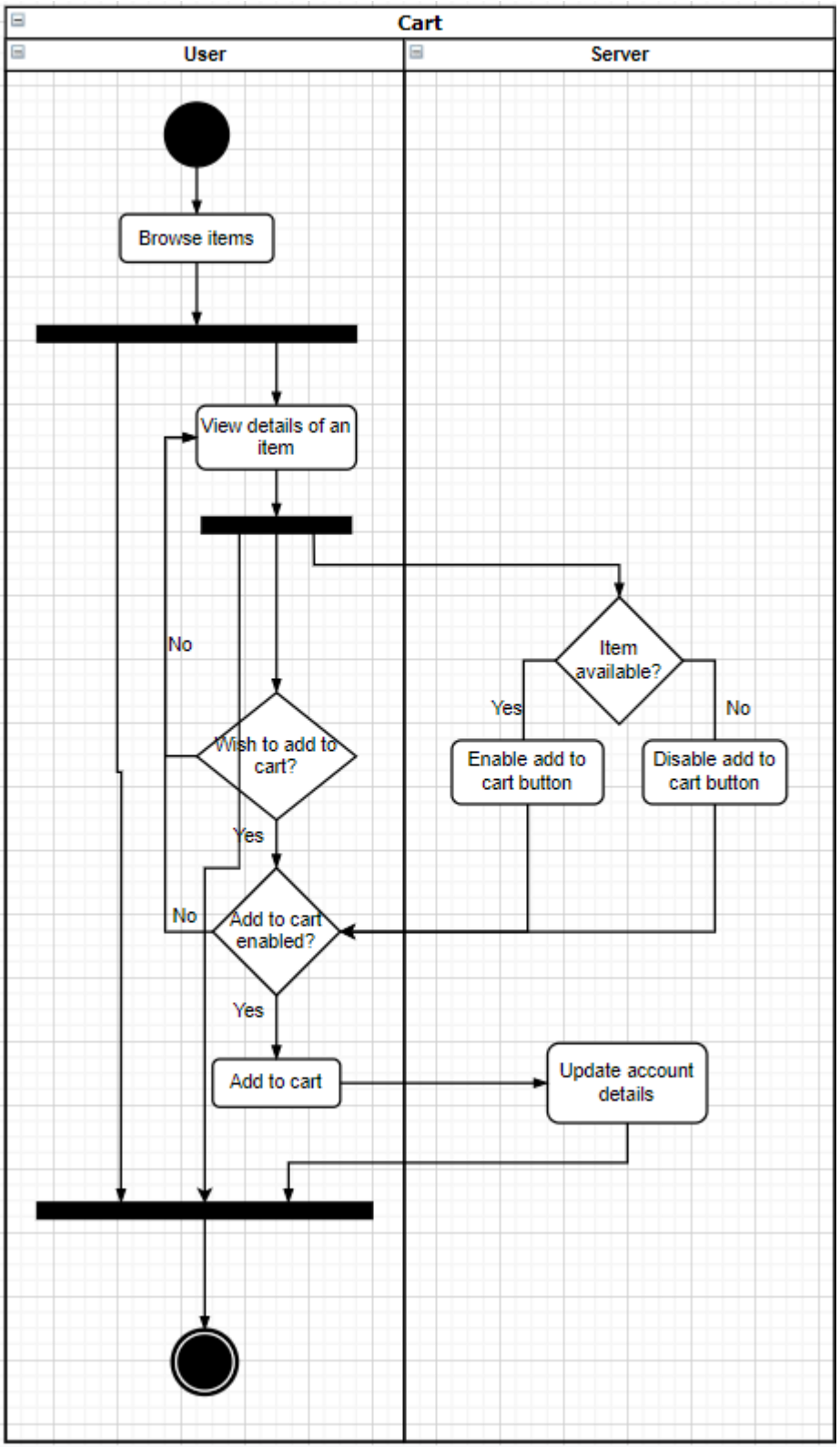


1.5 Usecase - Add to cart

1.5.1 Activity Diagram

An activity diagram visually presents a sequence of actions or flow of control in a structure similar to a flowchart or a data flow diagram. Activity diagrams are also used in business process modelling.The activity diagram shown below has two swimlanes mainly user, serverand the application server that group together activities that come under the swimlane's purview. The activity diagram describes the activities of the user executing concurrently while browsing the items available. The user can add the product to cart if the option is enabled, meaning the item is

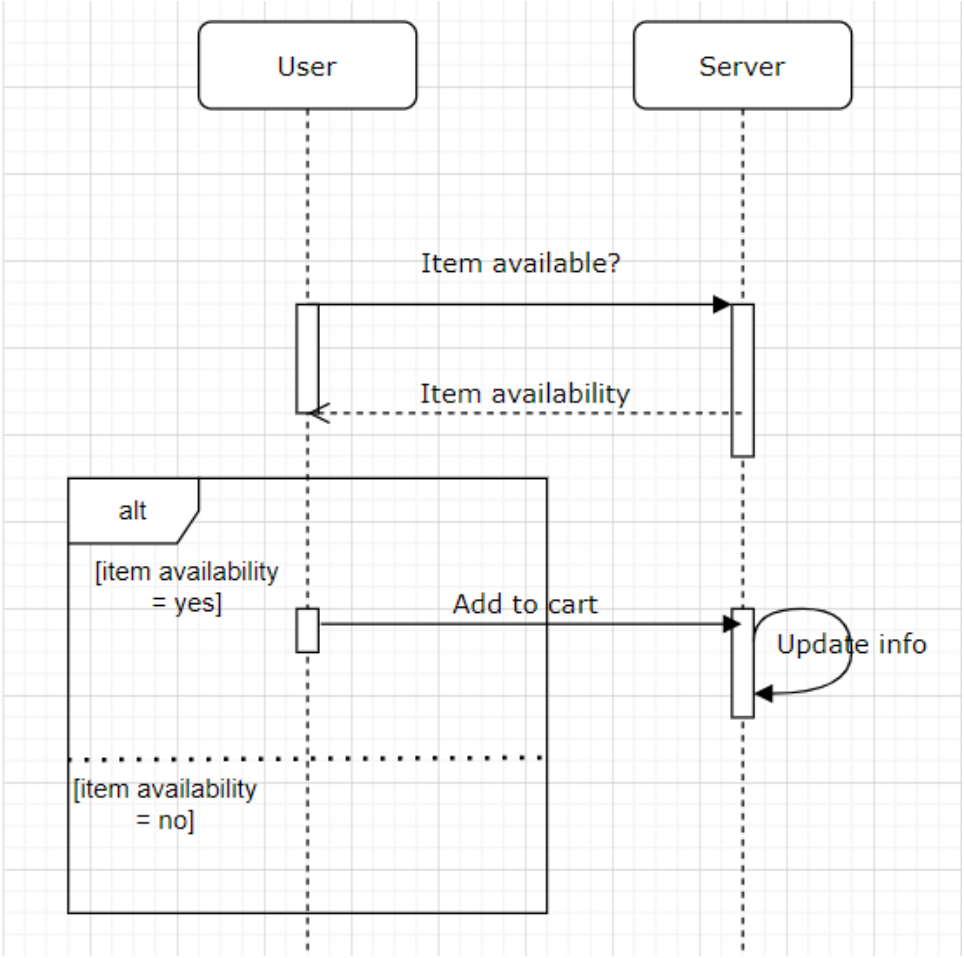
available in the inventory and if it's not then the option is disabled. The check for availability would access the application database for the required product.



1.5.2 Sequence Diagram

Interaction diagrams such as UML Sequence Diagrams show how operations are carried out. They capture the interaction between objects in a collaborative setting.

Sequence Diagrams are time-focused and visually depict the order of an interaction by using the vertical axis of the diagram to illustrate time and the messages sent and received.The sequence diagram shown below describes interactions between objects of the classes represented in the class diagram. The objects here described here are server and user.The note “Alt” here is used to convey that based of the value of the condition the messages are sent to carry out transactions. The user who implicitly queries the availability of the product in the server sends a message for the same. Based on the reply message by the server, user can either add the item to cart or not.

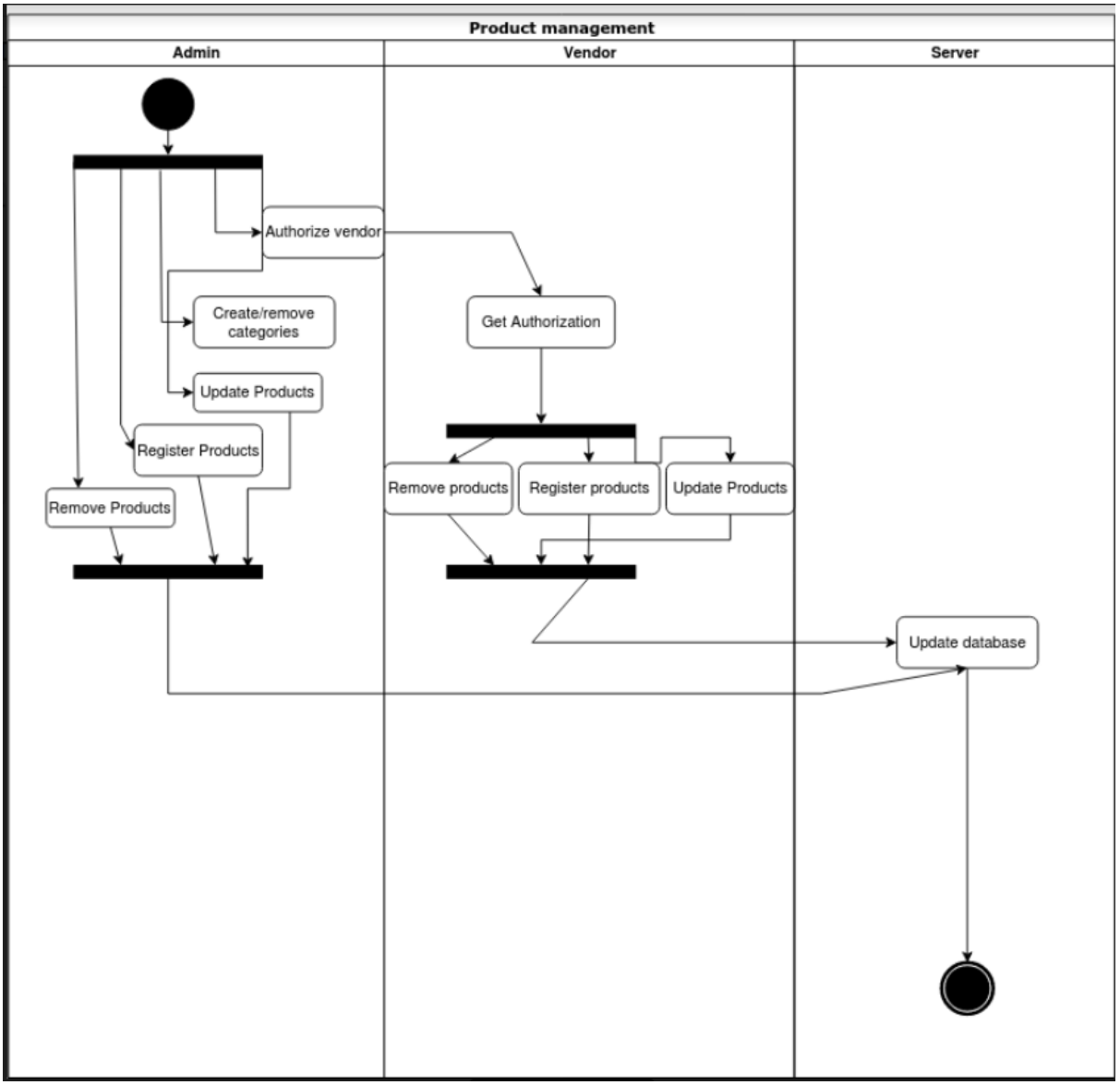


1.6 Use Case - Product Management

1.6.1 Activity Diagram

An activity diagram visually presents a sequence of actions or flow of control in a structure similar to a flowchart or a data flow diagram. Activity diagrams are also used in business process modelling.The activity diagram shown below has three swimlanes mainly admin , vendor and the application server that group together activities that come under the swimlane’s purview.The activity diagram describes the activities of the admin executing concurrently - registering, creating, removing, updating products , which then merge together to be updated in the database on successful operation. The activity diagram describes the activities of the vendor who upon getting authorization from the admin which are executing concurrently -

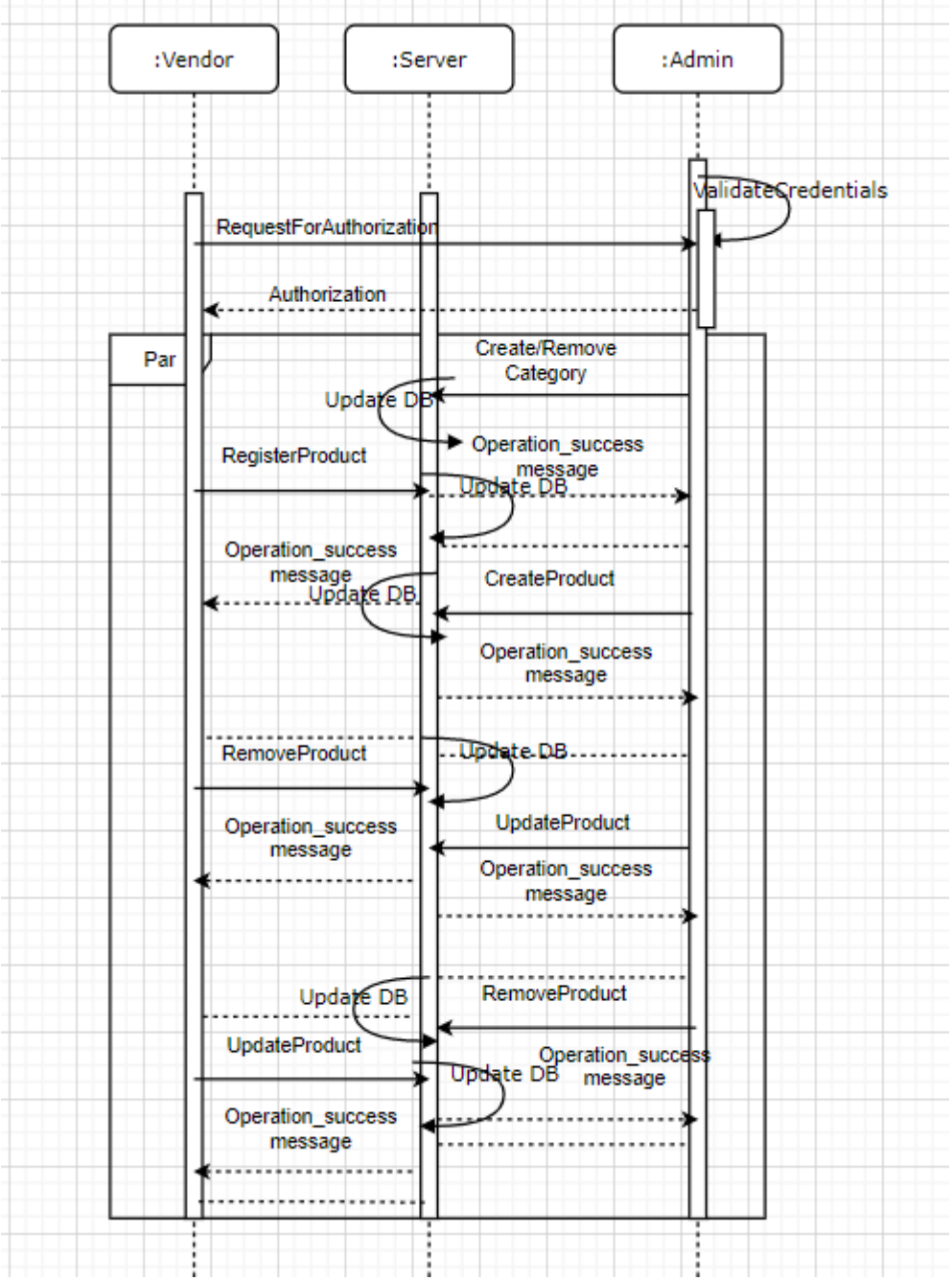
creating, removing, updating products which then merge together to be updated in the application database on successful operation. These operations being atomic does not result in rac conditions or undefined behavior due to deadlocks.



1.6.2 Sequence Diagram

Interaction diagrams such as UML Sequence Diagrams show how operations are carried out. They capture the interaction between objects in a collaborative setting. Sequence Diagrams are time-focused and visually depict the order of an interaction by using the vertical axis of the diagram to illustrate time and the messages sent and received.The sequence diagram shown below describes interactions between objects of the classes represented in the class diagram. The objects here described here are vendor, server and admin. The vendor after successful validation of credentials by the admin can send messages to the server for performing registering , updating and deleting of products that involve updating the database for every transaction. The note “Par” here is used to convey that the transactions are carried out parallely

during the activation of the lifeline. The admin can send messages to the server for performing creation of category and product , updation and removal.

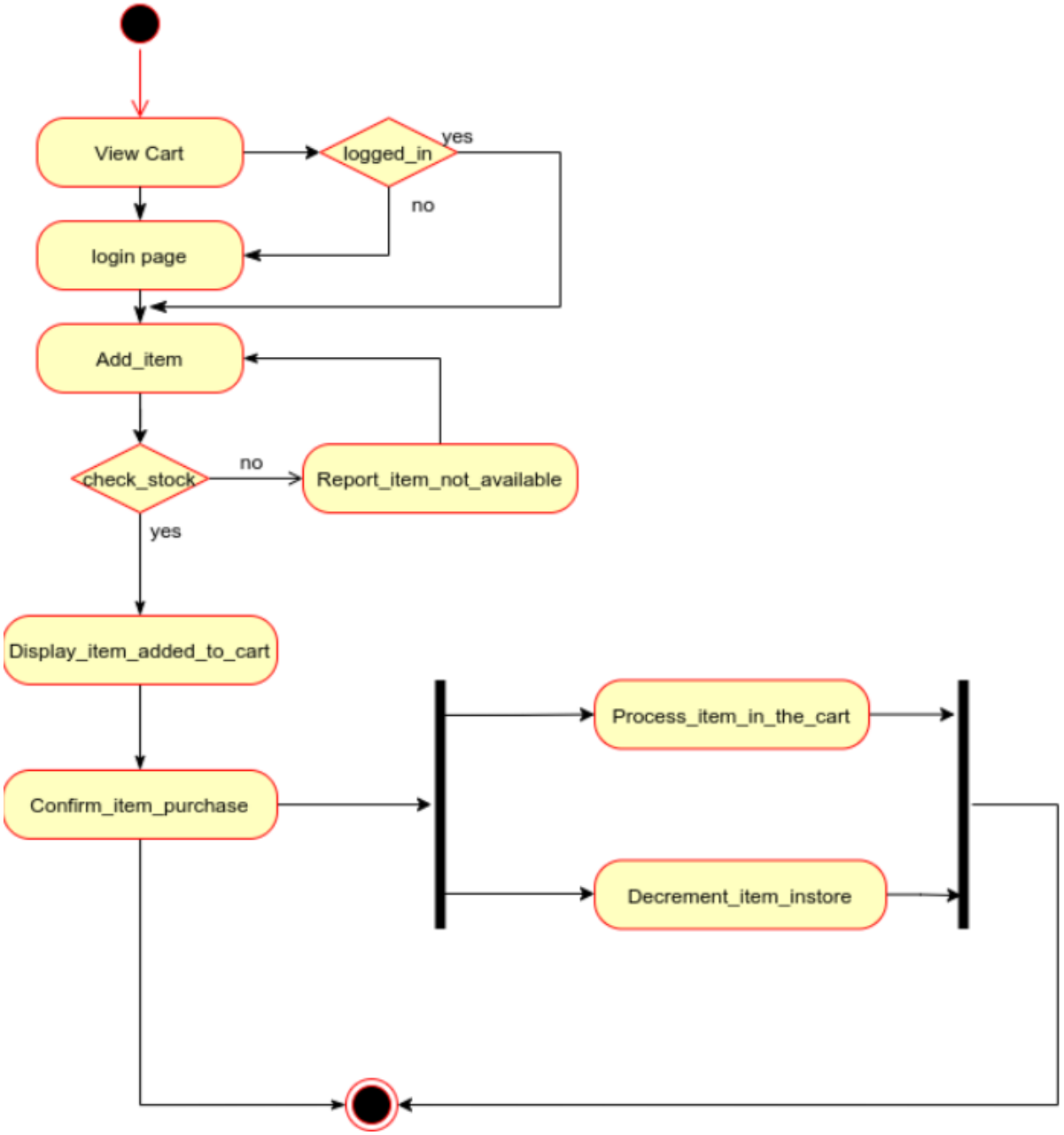


1.7 Use Case - Manage Cart

1.7.1 Activity Diagram

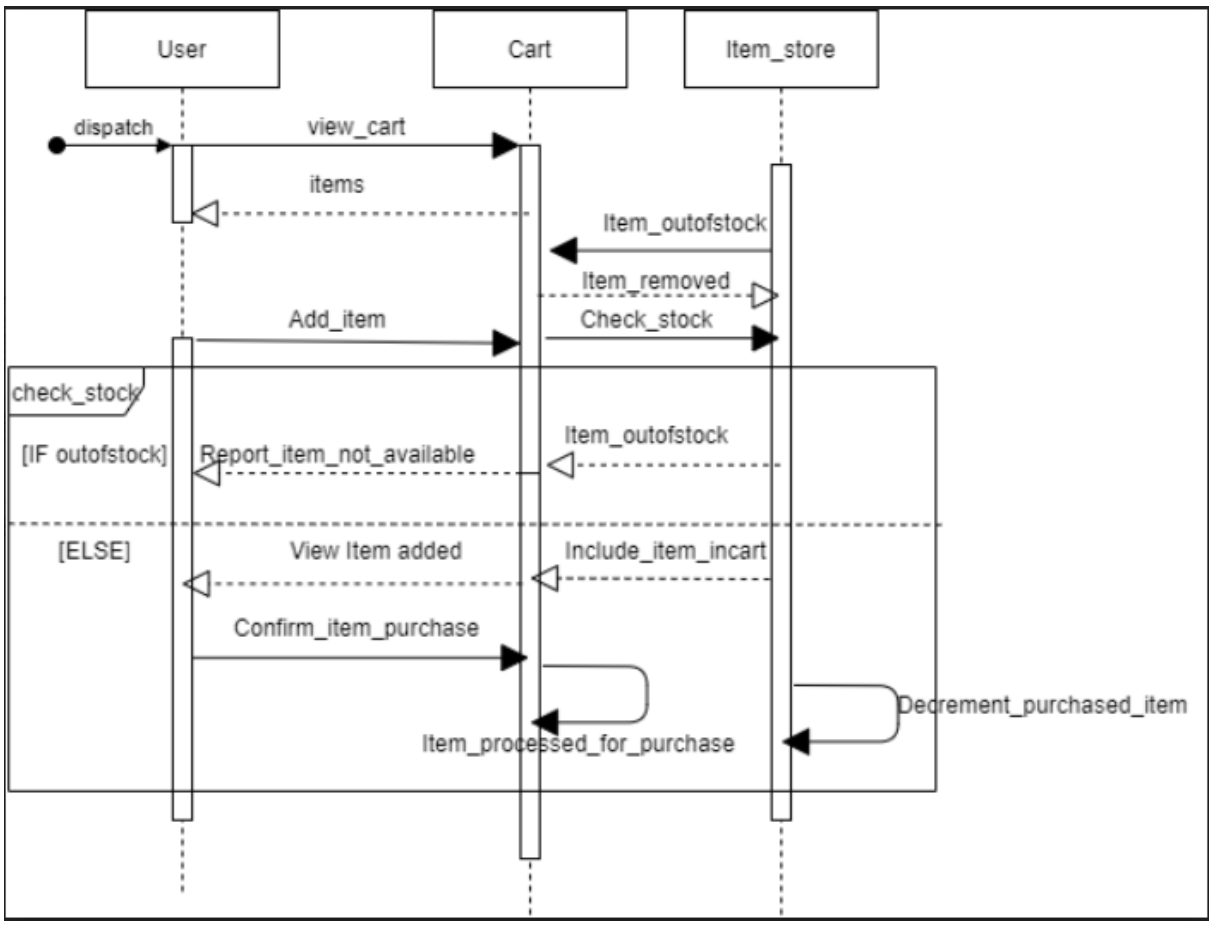
An activity diagram visually presents a sequence of actions or flow of control in a structure similar to a flowchart or a data flow diagram. Activity diagrams are also used in business process modelling. The activity diagram shown below depicts the various activities associated with cart management use case. Here the system is designed under the assumption that the user may or may not be logged in when viewing the cart, hence the decision node following the activity- “view cart“ handles this situation. The fork node following the activity-”confirm item purchase” performs

two operations concurrently - processing the item in cart and updating the application database.



1.7.2 Sequence Diagram

Interaction diagrams such as UML Sequence Diagrams show how operations are carried out. They capture the interaction between objects in a collaborative setting. Sequence Diagrams are time-focused and visually depict the order of an interaction by using the vertical axis of the diagram to illustrate time and the messages sent and received.The sequence diagram shown below describes interactions between objects of the classes represented in the class diagram. The objects described here are user cart and Item store. The note “check stock” here is used to convey that based on the reply message from the cart to the user, the user performs the following actions.
If an item is available the Item store returns “include item in cart “ message to cart which then returns view item in cart message to user.


User
Cart
Item_store
dispatch
view_cart
items
Item_outofstock
Item_removed
Add_item
Check_stock
check_stock
[IF outofstock]
Report_item_not_available
Item_outofstock
[ELSE]
View Item added
Include_item_incart
Confirm_item_purchase
Decrement_purchased_item
Item_processed_for_purchase

CHAPTER-4

Module Description

The three use cases implemented are :

1. Adding item to cart : The use case involves addition of items or products to the users cart from the available products that are visible to the user. These products might be either queried by the user or presented to the user as recommendations. This process should periodically monitor the availability of the products in the main application database . When an item goes out of stock , it should no longer be available to the user to add to cart and when an item already in cart goes out of stock it should be removed from the cart while notifying the user about the same.
2. Product management: The use case involves the management of the application database/ inventory when the products and their respective categories are added. Updation of product attributes and removal of products are also present under this use case . The changes in the application database must be reflected in the associated use cases and components and dependent services.
3. Cart management: The use case involves the management of the cart of the user. The user's cart should be in coherence with the inventory's stock count and must notify the user when there is an update in stock that no longer permits the [presence of the item in the cart .

CHAPTER-5

Test Cases

5.1 USE CASE : ITEM SELECTION AND ADDING TO CART

Test Case ID	Name of Module	Test Case Description	Pre-Conditions	Test Steps	Test Data	Expected Results	Actual Result	Test Result
1	Item Selection	Test product viewing functionality	User opened the website on browser and logged in	1. Navigate to www.amazon.com 2. Enter username and password. 3. Click submit.	NA (logging in functionality not implemented)	All the available products in home screen displayed	All the available products in home screen displayed	Pass
2	Add to cart	Test adding a single product to cart	User is logged in and is viewing the products	User clicks on the Add to Cart button of a product	ID of product being added to cart : 8	Product whose ID is 8 is added to cart	Product whose ID is 8 is added to cart	Pass
3	Add to cart	Test adding multiple products to cart	User is logged in and is viewing the products	User clicks on the Add to Cart button of multiple products	ID of products being added to cart : 8, 18	Products whose IDs are 8 and 18 are added to cart	Products whose IDs are 8 and 18 are added to cart	Pass
4	Add to cart	Test adding a single product number of times which is less than the stock available	User is logged in and is viewing the products	User clicks on the Add to Cart button of the same product for a number of times which is less than the current stock of that product	Product ID: 3 Current stock : 10 Number of times added to cart : 4	The count of the product becomes 1 the first time it is added. In the end, the count becomes 4.	The count of the product becomes 1 the first time it is added. In the end, the count becomes 4.	Pass
5	Add to cart	Test adding a product to cart which is	A product is present in the cart	User clicks on the “Add to	Product ID: 3 Count in	Each time the button is clicked,	Each time the button is clicked,	Pass

		already present in the cart and the quantity in cart is already equal to the current stock of the product in the inventory.	and its quantity in cart is equal to the current stock of the product in the inventory.	Cart" button of this product multiple times.	Cart: 10 Stock in inventory: 10	an Error window is shown which says "Only 10 items are in stock!"	an Error window is shown which says "Only 10 items are in stock!"	
6	Add to cart	Test adding multiple products to cart multiple number of times.	User is logged in and is viewing the products.	User clicks on Add to Cart button of multiple products multiple times	Product IDs: 3, 7 Number of times added to cart (each) : 3	The count of the products in cart becomes 1 when added for the first time. The count increases each time the button is clicked and the count for ID 3 becomes 3. When product 7 is added twice, the count becomes 2 and when the button is clicked again, an error window with the message "Only 2 items are in stock!" is shown	The count of the products in cart becomes 1 when added for the first time. The count increases each time the button is clicked and the count for ID 3 becomes 3. When product 7 is added twice, the count becomes 2 and when the button is clicked again, an error window with the message "Only 2 items are in stock!" is shown	Pas s
7	Add to cart	Test adding a product multiple	A product is added to cart	User clicks on the "Add to	Product ID: 7 Initial	Product is added to cart the	Product is added to cart the	Pas s

		times while the stock in the inventory reduces.	multiple times while the stock of the product in the inventory reduces (but still remains greater than the count in cart)	cart" button of a product. The stock of the product is reduced by one. User again clicks on "Add to cart" button of the same product	count in cart: 1 Initial stock in inventory: 2 Final stock in inventory: 1	first time. If the button is clicked again, the error window is shown with the message "Only 1 items are in stock!"	first time. If the button is clicked again, the error window is shown with the message "Only 1 items are in stock!"	
8	Add to cart	Test adding a product multiple times and the stock of the product reduces to less than the current count of the product in the cart.	A product is added to cart multiple times. The stock of the product reduces to less than the count of the product in cart.	User clicks on the "Add to cart" button of a product multiple times. The stock of the product reduces to lower than its count in the cart. User clicks the button again.	Product ID: 3 Initial stock : 10 Initial count in cart: 7 Updated Stock : 5	When the product is added to cart 7 times, the count becomes 7. When the stock reduces to 5, the count becomes 5. When the user clicks on the button again, the error window is shown with the message "Only 5 items are in stock!"	When the product is added to cart 7 times, the count becomes 7. When the stock reduces to 5, the count becomes 5. When the user clicks on the button again, the error window is shown with the message "Only 5 items are in stock!"	Pas s
9	Add to cart	Test adding a product to cart again when the product's stock is 0	The user is logged in and is viewing the products	The user clicks on the "Add to Cart" button of a product whose stock is 0.	Product Id : 7	An error window is shown with the message "The product is currently not in stock!"	An error window is shown with the message "The product is currently not in stock!"	Pas s

10	Entire System	Testing the entire system from start to end	User logs in and enter the home page and can add products to cart.	The user logs in by giving correct credentials. He can view the products. He can add any product any number of times depending on the stock of the product.	Products added to cart: ID : 8 Count: 2 ID : 7 Count : 1 ID : 18 Count : 5 Stock of ID 18 is reduced to 3 Stock of ID 7 is made 0	Final cart contents: ID : 8 Count ; 2 ID : 18 Count : 3	Final cart contents: ID : 8 Count ; 2 ID : 18 Count : 3	Pas s

5.2 USE CASE : MANAGING CART

Test Case ID	Name of Module	Test Case Description	Pre-Conditi ons	Test Steps	Test Data	Expected Results	Actual Result	Test Res ult
11	View Cart content s	Keep the cart products persistent, unless not out of stock.	User to be logged in to view his products added to the cart	1. Navigate to www.amaz on.com 2. Enter username and password. 3. Cart will be displayed	NA (logging in not implemen ted, assume the user to be logged in)	Products added to the cart are displayed in the cart	Products added to the cart are displayed in the cart	Pass
12	Managi ng Cart	Test increasing the current stock of the product, in the backend	User is logged in and viewing the product in the cart. The product exists in cart and admin only allowed to make the	Admin increases the stock and updates it by clicking submit button	Product existing in the cart is with ID := 7	On click of submit button, Product with ID = 7, updates its current stock field with the existing stock + increment value, in	On click of submit button, Product with ID = 7, updates its current stock field with the existing stock + increment value, in	Pass

			modificatio n			the cart window	the cart window	
13	Managi ng Cart	Test decreasing the current stock of the product, in the backend	User is logged in and viewing the product in the cart. The product exists in cart and admin only allowed to make the modificatio n	Admin decreases the stock and updates it by clicking submit button	Add another product ID = 3 in the cart, will serve as the test data for this case	On click of submit button, Product with ID=3, updates its current stock field with the existing stock - decrement value, in the cart window	On click of submit button, Product with ID=3, updates its current stock field with the existing stock - decrement value, in the cart window	Pass
14	Managi ng Cart	Test using values other than the integer values in the increase and reduce stock field	User is logged in and viewing the product in the cart. The product exists in cart and admin only allowed to make the modificatio n	Admin enters float or string data type value in the increase or the decrease stock and clicks submit button	Product ID=3, taken as test data which exists in the cart	The notification window pops-up displaying an error message stating : please enter valid value in the fields.Also no modificatio n takes into effect in the cart.	The notificatio n window pops-up displaying an error message stating : please enter valid value in the fields. Also no modificati on takes into effect in the cart.	Pass
15	Managi ng cart	Test reducing stock in the update field below the current stock value for the product existing in the cart	User is logged in and viewing the product in the cart. The product exists in cart and admin only allowed to make the modificatio	Enters the value in the reduce stock field below the current stock of the product mentioned in the product id field and click submit	Product ID=3, taken as test data present in the cart window	When the submit button is clicked the notification window pops-up stating: Update value increases beyond current stock.Also	When the submit button is clicked the notificatio n window pops-up stating: Update value increases beyond current stock.Also	Pass

			n	button		no modificatio n takes into effect in the cart.	no modificati on takes into effect in the cart.	
16	Managi ng cart	Test entering both the increase and decrease stock field simultaneou sly of the product existing in the cart	User is logged in and viewing the product in the cart. The product exists in cart and admin only allowed to make the modificatio n	Admin goes to the update window and enters both the increase stock and decrease stock field with the valid data type(i.e. integer). Then clicks the submit button	Product ID:=3, taken as test data which is present in the cart window	When the submit button is clicked, the notification window pops-up stating: Please enter either increase or decrease stock field and not both simultaneo usly. Also no modificatio n takes into effect in the cart.	When the submit button is clicked, the notificatio n window pops-up stating: Please enter either increase or decrease stock field and not both simultane ously. Also no modificati on takes into effect in the cart.	Pass
17	Managi ng Cart	Test reducing the stock of the item existing in the cart to zero.	User is logged in and viewing the product in the cart. The product exists in cart and admin only allowed to make the modificatio n and is aware of the current stock of the particular product	In the update window, the reduce stock field is entered with the value of the current stock of the product under test and clicking the submit button.	Product ID:=3, taken as the test data which is existing in the cart window	When the submit button is clicked the stock reduces to zero in the Inventory window and a notification message pops-up stating that the product ID(its name displayed) which was in the cart is out of	When the submit button is clicked the stock reduces to zero in the Inventory window and a notificatio n message pops-up stating that the product ID(its name displayed) which was in the cart	Pass

						stock. This notification can be used as a service to let the user know about the status of the product which is kept in the cart. That product also disappears from the cart window.	is out of stock. This notification can be used as a service to let the user know about the status of the product which is kept in the cart. That product also disappears from the cart window.	
18	Managing Cart (Compute Total)	Users can compute the total amount of the products in the cart individually based on the quantity of the products and then finally compute the total cost. Uses product cost details from product inventory	User is logged in and viewing the products in the cart, at least one product present in the cart.	First add items to the cart and User clicks the “Compute Total” button at the bottom of the cart, the notification message pops-up.	Product ID=7, it’s count = 2. Product ID=18, it’s count = 1. Product ID=3, it’s count = 1.	After the click of the button the message box pops-up listing out the total of each product in the cart and the grand total amount. Displays Total ID=7 is 8000. Total ID=18 is 3000 and Total ID=3 is 2000. Overall total = 13000	After the click of the button the message box pops-up listing out the total of each product in the cart and the grand total amount. Displays Total ID=7 is 8000. Total ID=18 is 3000 and Total ID=3 is 2000. Overall total = 13000	Pass
19	Managing Cart (Compute	Users compute the total when there is no	The user is logged in and is viewing the	When the cart is empty the user clicks	Empty cart window	On click of button the message pops-up	On click of button the message pops-up	Pass

	Total)	product in the cart, either being deleted from cart or removed due to out of stock or no item added and compute total button clicked. This tests the edge case.	products in the cart window. No restriction on whether an item is in the cart initially and then removed or item never added.	the “Compute total” button at the bottom of the cart window.		only displaying the total as 0.	only displaying the total as 0.	
20	Entire System	Testing the entire system from start to end	User logs in and enter the home page, then selects the cart window	The user logs in by giving correct credentials. He can view the products. He can add any product any number of times depending on the stock of the product. The admin can update the product's details, and users can change contents in their cart directly using product id. This system integration test checks the behaviour of the model	Multiple products are added to the cart which forms the test data and then managing of the cart mostly done by the admin on it	Random testing performed and based on the multiple test yields expected **consisten t** results	Random testing performed and based on the multiple test yields expected **consisten t** results	Pass

				under the above specified test cases, when all the features are put together				

5.3 USE CASE: Product Management

Test Case ID	Name of Module	Test Case Description	Pre-Conditions	Test Steps	Test Data	Expected Results	Actual Result	Test Result
21	Add to inventory	Test adding product to inventory functionality	User opened the website on browser and logged in as Vendor	1. Navigate to www.amazon.com 2. Enter username and password for vendor. 3. Click submit.	NA (logging in functionality not implemented)	All the available products in home screen displayed	All the available products in home screen displayed	Pass
22	Add to inventory	Test adding product to inventory functionality	User opened the website on browser and logged in as Vendor	1. Enter product Id 2. Enter product Name 3.Enter product price 4.Enter product stock qty	1.Id - 123 2.Name - Gatorade 3.Price-100.00 4.Stock qty Inc - 250 5.Stock qty Dec -”Empty”	Product with Id 123 is displayed in the inventory display box along with already added products	Product with Id 123 is displayed in the inventory display box along with already added products	Pass
23	Add to inventory	Test adding product to inventory function	User opened the website on browser and	1. Enter product Id 2. Enter product Name 3.Enter product	1.Id - 123 2.Name - “Empty” 3.Price-100.00 4.Stock qty Inc - 250	A popup window is shown having message - “All fields mandatory,	A popup window is shown having message - “All fields mandatory,	Pass

		ality	logged in as Vendor	price 4.Enter product stock qty	5.Stock qty Dec -”Empty”	Please enter all fields with valid values”	Please enter all fields with valid values”	
24	Add to inventory	Test adding product to inventory functionality	User opened the website on browser and logged in as Vendor	1. Enter product Id 2. Enter product Name 3.Enter product price 4.Enter product stock qty	1.Id - Id 2.Name - Gatorade 3.Price-100.00 4.Stock qty Inc - 250 5. Stock qty Dec-”Empty”	A popup window is shown having message -”Please enter valid values for fields,Valid types: Id:Integer,name:String, price:floatStockqty:Integer”	A popup window is shown having message -”Please enter valid values for fields,Valid types: Id:Integer,name:String,price:floatStockqty:Integer”	Pass
25	Update product in inventory	Test updating product in inventory functionality	User opened the website on browser and logged in as Vendor and has added some products to inventory successfully(one among them is of Id-125)	1. Enter product Id 2. Enter product Name to update 3.Enter product price to update 4.Enter product stock qty to update (Fields are optional)	1.Id - 125 2.Update Name - “Empty” 3.Update Price-200.00 4.Update Stock qty - “Empty” 5.UpdateStock qty Dec-”Empty”	Product with Id 125 is displayed in the inventory display box and also in the cart display box if added , with updated price of product .	Product with Id 125 is displayed in the inventory display box and also in the cart display box if added , with updated price of product .	Pass
26	Update product in inventory	Test updating product in inventory functionality	User opened the website on browser and logged in as Vendor and has	1. Enter product Id 2. Enter product Name to update 3.Enter product price to update	1.Id - 125 2.Update Name - RedBull 3.Update Price-200.00 4.Update Stock qtyInc - “Empty”	Product with Id 125 is displayed in the inventory display box and also in the cart display box if added ,	Product with Id 125 is displayed in the inventory display box and also in the cart display box if added ,	Pass

			added some products to inventory successfully(one among them is of Id-125)	4.Enter product stock qty to update (Fields are optional)	5.Update Stock qty Dec-”Empty ”	with updated price and name of product .	with updated price and name of product .	
27	Update product in inventory	Test updating product in inventory functionality	User opened the website on browser and logged in as Vendor and has added some products to inventory successfully(one among them is of Id-125)	1. Enter product Id 2. Enter product Name to update 3.Enter product price to update 4.Enter product stock qty to update (Fields are optional)	1.Id - RedBullId 2.Update Name - RedBull 3.Update Price-200.00 4.Update Stock qtyInc - “Empty” 5.Update Stock qty Dec-”Empty ”	A popup window is shown having message -”Please enter valid values for fields,Valid types: Id:Integer,name:String, price:floatStockqty:Integer”	A popup window is shown having message -”Please enter valid values for fields,Valid types: Id:Integer,name:String,price:floatStockqty:Integer”	Pass
28	Delete product in inventory	Test deleting product in inventory functionality	User opened the website on browser and logged in as Vendor and has added some products to inventory successfully(one among them is of Id-125)	1. Enter product Id in delete Entry	1.Id-125	The product is removed from Inventory display box.	The product is removed from Inventory display box and cart if added.	Pass
29	Update	Test	User	1. Enter	1.Id - 125	Product	Product	Pass

	product in inventory	updating product in inventory functionality	opened the website on browser and logged in as Vendor and has added some products to inventory successfully(one among them is of Id-125,StockQty = 10)).Prod is added to cart	product Id 2. Enter product Name to update 3.Enter product price to update 4.Enter product stock qty to update (Fields are optional) 5.Add to cart	2.Update Name - “Empty” 3.Update Price-“Empty” 4.Update Stock qty - “Empty” 5.Update Stock qty Dec-”10”	StockQty reduced to 0 in the Inventory display box and product is removed from Cart .	StockQty reduced to 0 in the Inventory display box and product is removed from Cart .	
30	Update product in inventory	Test updating product in inventory functionality	User opened the website on browser and logged in as Vendor and has added some products to inventory successfully(one among them is of Id-125,StockQty = 0)).Try to add to cart.	1.Add product to cart whose StockQty is 0	1.Click add to cart of product whose Stock Qty is 0	A popup message is shown with message-” This product is currently not in stock”	A popup message is shown with message-” This product is currently not in stock”	Pass
31	Delete product in inventory	Test deleting product in inventor	User opened the website on	1. Enter product Id in delete Entry	1.Id-125	The product is removed from Inventory	The product is removed from Inventory display box	Pass

		y function ality	browser and logged in as Vendor and has added some products to inventory successfu lly(one among them is of Id-125<st ockQty=1 0) and is added to cart			display box and also from cart.	and also from cart.	
32	Entire System	Testing the entire system from start to end	User logs in as vendor on authentic ation by admin and accesses vendor view of the site	The user logs in by giving correct credentials. He can view the products. He can add any product any number of times depending on the stock of the product.	Actions performed are reflected in the inventory display window and cart display window in accordance with the type of input given by the user as a vendor.	Random testing performed and based on the multiple test cases covering many permutation s yields expected **consistent** results	Random testing performed and based on the multiple test cases covering many permutation s yields expected **consistent** results	

CHAPTER-6

Screenshots of Output

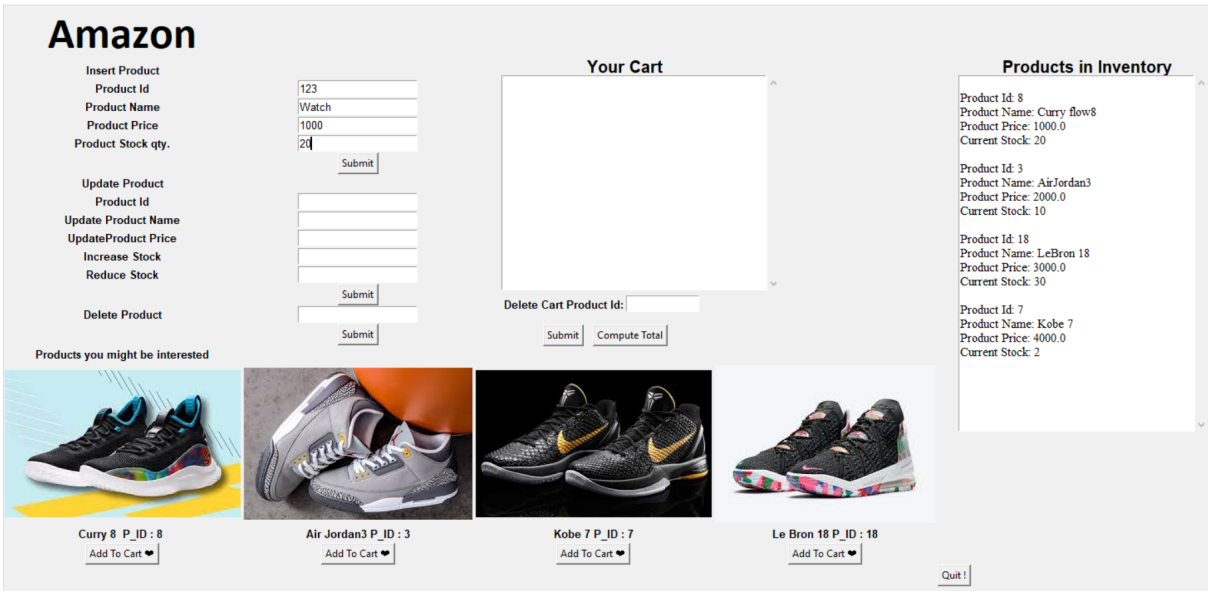


Fig: 6.1 Default page on load of website

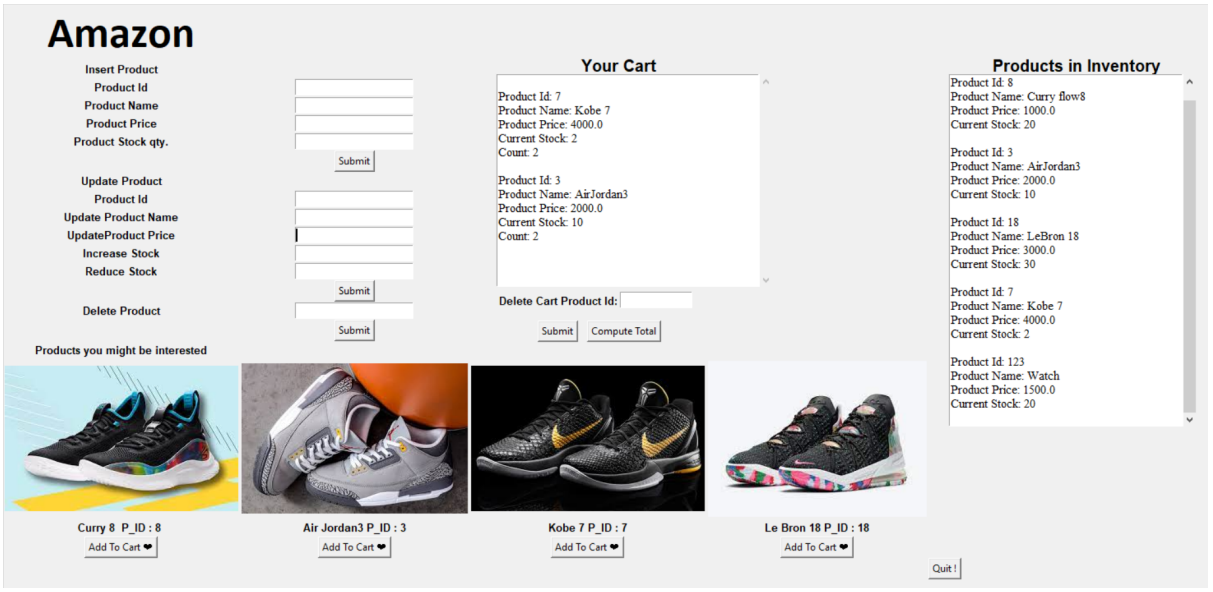


Fig: 6.2 Addition of products to inventory and cart

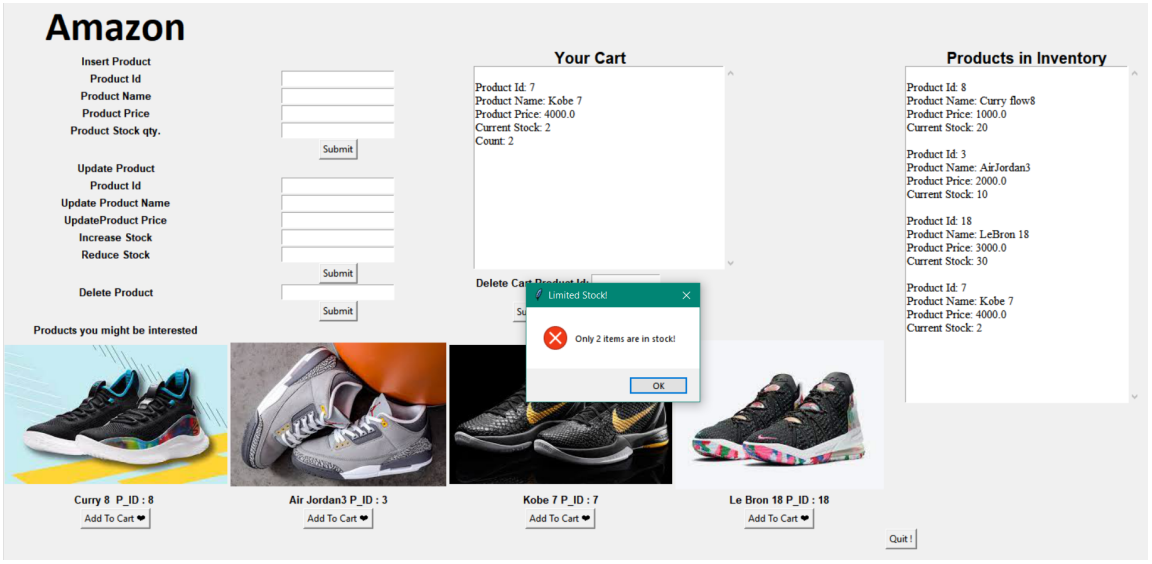


Fig: 6.3 Adding a product more times than its current stock in inventory

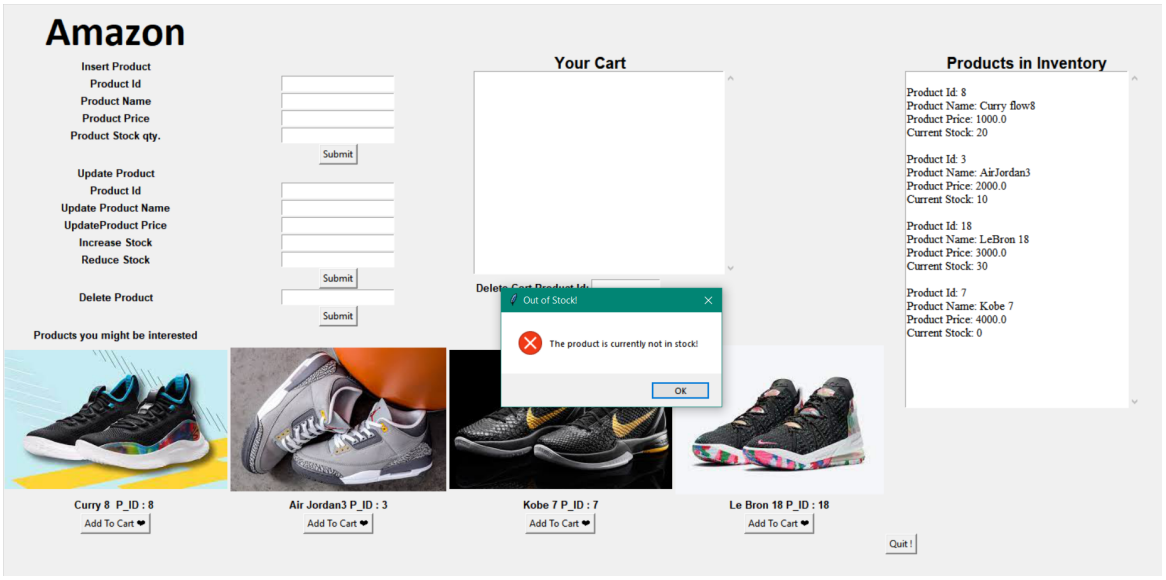


Fig: 6.4 Adding a product when it is out of stock

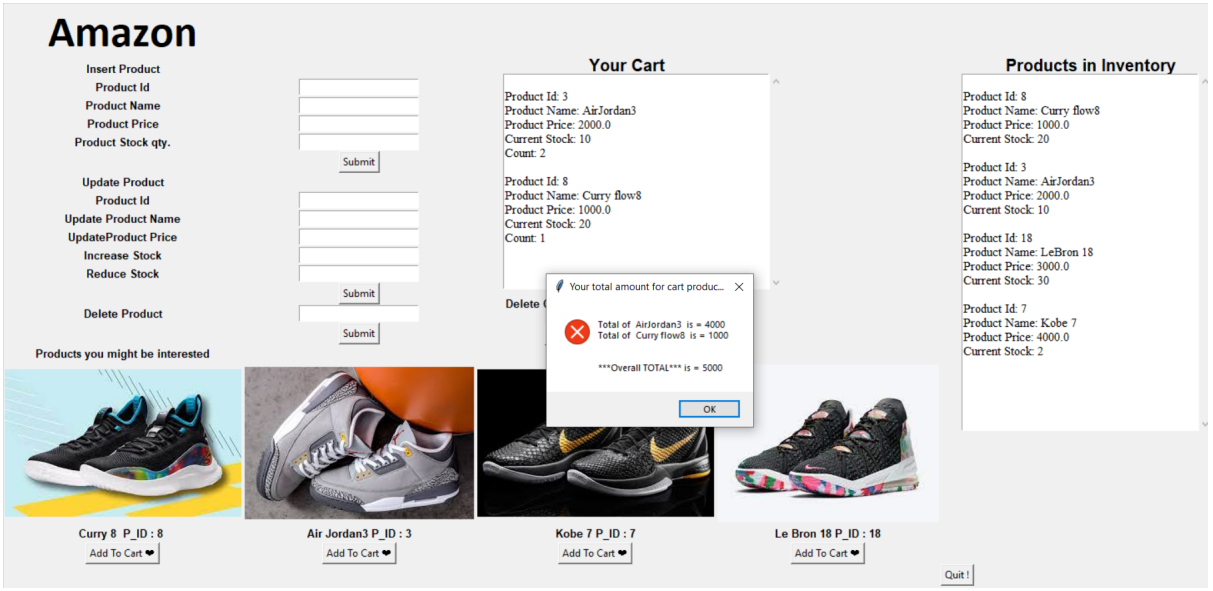


Fig: 6.5 Computing cart total cost

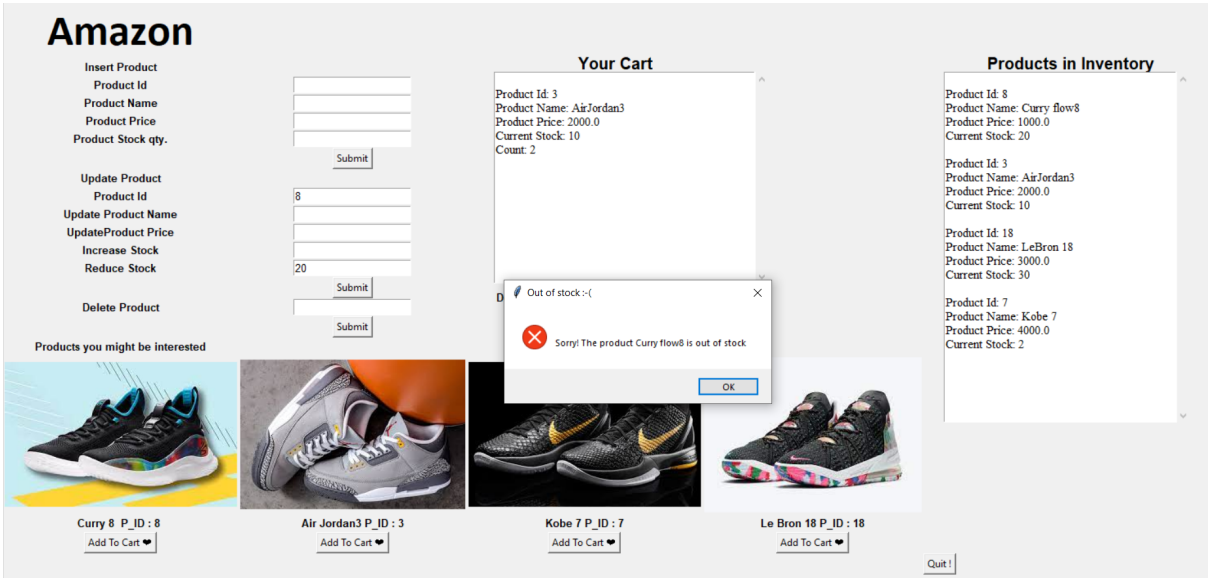


Fig: 6.6 Item in cart goes out of stock

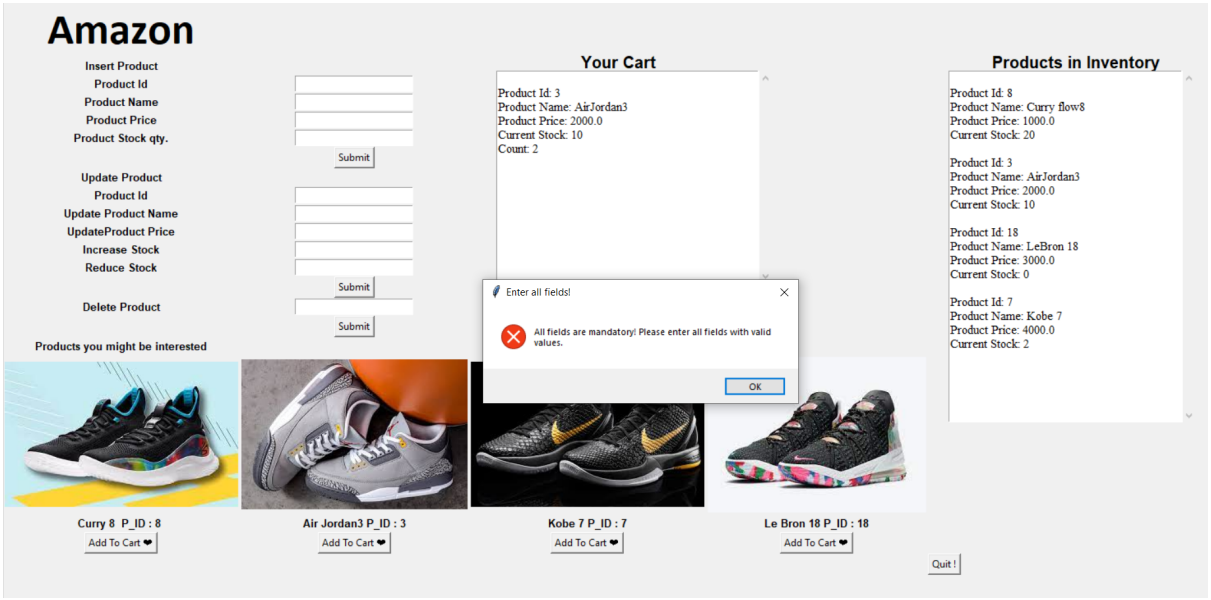


Fig: 6.7 Insufficient number of arguments passed