Software Design Specifications

[ShopFlow]

Version: [1.0]

Project Code	*SF2024*
Supervisor	Munawar Shereen
Co Supervisor	Huzaifa Faran
Project Team	*Munawar Shereen* *Huzaifa Faran* *Faraz Iqbal*
Submission Date	5-May-2024

Document History

Version	Name of Person	Date	Description of change
1.0	*Munawar Shereen*	*17th-April-2024*	*Initial Creation of Document*
1.1	*Huzaifa Faran*	*25th-April-2024*	*Added Activity diagram*
1.2	*Faraz Iqbal*	*29-April-2024*	*Added Sequence diagram*
1.3	*Huzaifa Faran*	*1st-May-2024*	*Added Component diagram*
1.4	*Munawar Shereen*	*5th-May-2024*	*Review and finalize the document for submission*

Distribution List

Name	Role
Munawar Shereen	Supervisor
Huzaifa Faran	Co Supervisor

Document Sign-Off:

Version	Sign-off Authority	Sign-off Date
1.0	*Huzaifa Faran*	*1st-May-2024*
1.6	*Munawar Shereen*	*5h-May-2024*

Document Information

Category	Information
Customer	FAST-NU
Project	<ShopFLow>
Document	Software Design Specification
Document Version	1.0
Status	Draft
Author(s)	Munawar Shereen, Huzaifa Faran and Faraz Iqbal
Approver(s)	
Issue Date	
Document Location	Karachi
Distribution	Advisor Project Coordinator’s Office (through Advisor)

Definition of Terms, Acronyms and Abbreviations

[This section should provide the definitions of all terms, acronyms, and abbreviations required to interpret the terms used in the document properly.]

Term	Description
ASP	Active Server Pages
DD	Design Specification
SDS	Software Design Specification
GUI	Graphical User Interface

Table of Contents

1 Introduction

1.1 Purpose of Document

The objective of this document is to provide an extensive depiction of the "Shopping Flow" project within the Software Design Specification (SDS). It aims to elucidate the system's objectives and characteristics, its interfaces, functionality, operational constraints, and its responsiveness to external inputs.

1.2 Intended Audience

These are the different readers for whom this SDS is intended:

***Developers**: Skilled in software development and programming, developers will utilize this SDS as a guide to create code that complies with the particular design specifications and technologies used in the development of ShopFlow.*

***Project managers:** will rely on the SDS to make sure that the system is created in accordance with the agreed design, fulfilling project milestones, financial limitations, and stakeholder expectations. Project managers are in charge of supervising project finances, timelines, and resources.*

***Marketing Staff:** In order to effectively promote ShopFlow to prospective customers and users, marketing staff will make use of the SDS to get an understanding of the features and benefits of the system. This understanding will allow them to create marketing strategies and materials that effectively emphasize the advantages of ShopFlow.*

***Users:** In order to better comprehend and utilize ShopFlow, end users will consult the SDS to obtain a better grasp of the system's features and user interface.*

***Testers:** Charged with comparing the system's performance to the design specifications, testers will use the SDS to perform extensive testing and spot any inconsistencies or problems with the system as it has been implemented.*

1.3 Document Convention

Arial font, italicized for emphasis, is used in the Software Design Specification (SDS) at a size of 10 points for the body of the text and 12 points for headers. To direct development efforts, each need is assigned a priority level (high, medium, or low). By default, higher-level needs priority levels are transferred to detailed requirements. Throughout the paper, important details and important information are highlighted in bold text. This format helps with the design and development of the ShopFlow system by ensuring readability and clarity.

1.4 Project Overview:

The "ShopFlow" semester project is a shopping application designed with a focus on object-oriented principles. The system incorporates three main classes: Admin, Customer, and Product. The Product class acts as a base class that is inherited by specialized subclasses representing product categories like medical care, accessories, electronics, and clothes. The project emphasizes a modular and hierarchical design using inheritance to organize product categories effectively. A graphical user interface (GUI) will be developed using standard Python libraries, allowing users to browse products, manage accounts (Admin and Customer), and facilitate a seamless shopping experience. The main class serves as the central component to execute the application, coordinating interactions between different classes and functionalities.

1.5 Scope

The "ShopFlow" project is an online shopping application with a graphical user interface (GUI) built on Python that combines several product categories. The program, which is meant for users who shop online, serves a wide range of customers looking to buy accessories, apparel, electronics, and health care products. Many essential features are currently absent from the "ShopFlow" project, such as deep customization, offline capability, advanced inventory management, extensive social media integration, AI-driven recommendations, extensive analytics, internationalization/localization, and scalable architecture. The main goal is to create a strong e-commerce platform with feature-rich capability that doesn't require complicated integrations, benefiting administrators and consumers alike.

2 Design Considerations

Object-Oriented Design: *The project will emphasize encapsulation and inheritance by modeling entities such as Admin, Customer, and Product categories using object-oriented programming techniques.*

Class Hierarchy: *To encourage code reuse, a hierarchical class hierarchy will be created, with Product serving as the base class that specialized subclasses (such as medical care, accessories) inherit from.*

User authentication: *To control access rights and guarantee data security, secure authentication methods will be added for the Admin and Customer roles.*

Error Handling: *To guarantee data integrity and system dependability, strong error handling and input validation will be put in place.*

User Interface Design: *To improve usability, focus will be placed on developing an intuitive and aesthetically pleasing interface.*

2.1 Assumptions and Dependencies

Assumptions:

- *Users are aware of how to use the GUI and basic software interaction to navigate the program.*
- *During system functioning, it is expected that user input is legitimate and structured appropriately.*
- *Enough resources are available in the development environment to use Python for GUI implementation.*

Dependencies:

- *Precise user input is necessary for data processing and GUI operation.*
- *Implementing GUI features and file operations requires third-party dependencies, such as Python libraries and file system access.*
- *The underlying operating system and necessary software dependencies determine whether or not the program is compatible.*

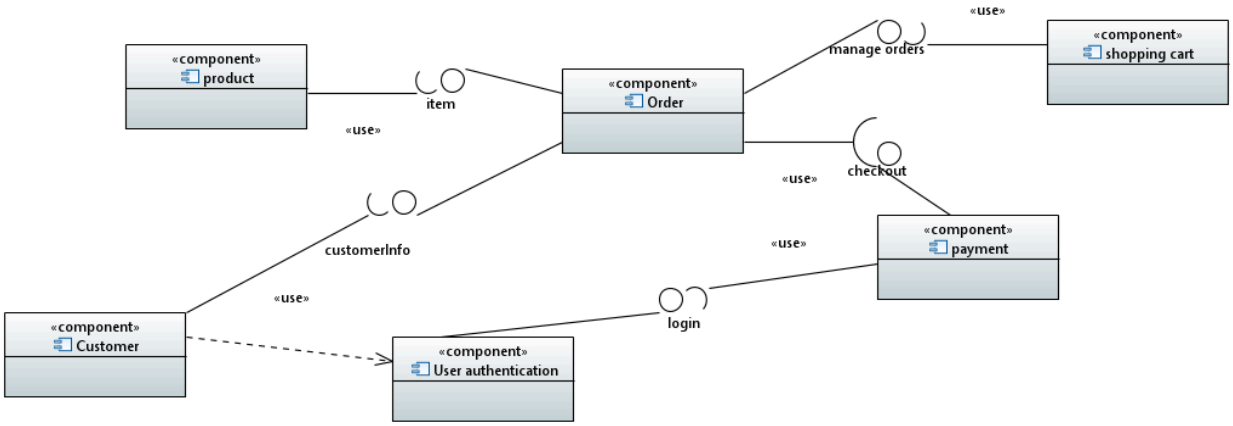
2.2 Risks and Volatile Areas

A shopping management system encompasses various components, and its design can be impacted by several factors. Following are the most likely sources of change and risk.
Enhanced Product Search and Filtering: Customers might desire more advanced search functionalities based on product attributes, price ranges, or personalized recommendations
Guest Checkout: Allowing purchases without user account creation might be a future requirement.
Subscription Services: Implementing subscription boxes or recurring purchase.
Changing Regulations: Regulations regarding online sales, data privacy, or payment security might necessitate adjustments to the system's functionalities and data handling practices.

3 System Architecture:

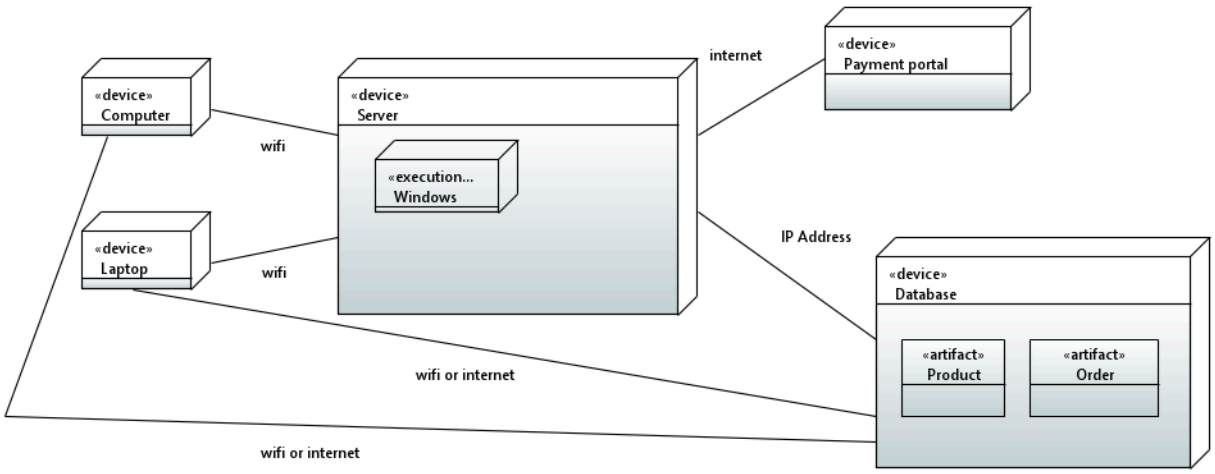
3.1 System Level Architecture

The component diagram illustrates the high-level structure of a system with components including product, customer, order, user authentication, payment, and shopping cart. Each component represents a distinct functional unit within the system. The product component manages product-related operations such as catalog management and inventory tracking. The customer component handles customer-related functionalities including user profiles and preferences. The order component manages the order processing workflow from creation to fulfillment. User authentication ensures secure access to system resources. The payment component handles payment processing and integration with external payment gateways. The shopping cart component manages the user's shopping session, including adding/removing items and calculating totals. These components interact with each other to facilitate e-commerce operations, ensuring a seamless experience for users throughout the system.



3.2 Software Architecture

The deployment diagram illustrates component distribution across devices including a computer, laptop, server, database, and payment portal. Users interact with the system through client-side components hosted on computers and laptops. Server-side components run on the server, managing business logic and coordinating system interactions. Data is stored and managed in the database hosted on a dedicated server. Payment transactions are processed through an external payment portal, integrated with the main system. This diagram visualizes how components are deployed across devices in the system architecture.



4 Design Strategy

The shopping system uses a microservices architecture with APIs for flexibility. New features can be added as independent services without affecting existing ones. This also allows for easier reuse of functionalities across different applications.

Event-driven communication between services promotes loose coupling and simplifies adding new functionalities by introducing new events or modifying how existing ones are handled.

Multiple UI options (web, mobile app, API) cater to different user preferences and access methods.

Data management strategies consider cost, performance, reliability, and security when choosing storage solutions (databases, cloud storage) for various data types (product info, customer details, orders).

Concurrency and synchronization mechanisms (locking, transactions) ensure data consistency during concurrent user access.

Trade-offs: Increased complexity vs. flexibility, maintainability, and scalability.

Security considerations are crucial throughout the design.

This approach aims to create a robust and adaptable shopping system architecture that can evolve with future needs.

5 Detailed System Design:

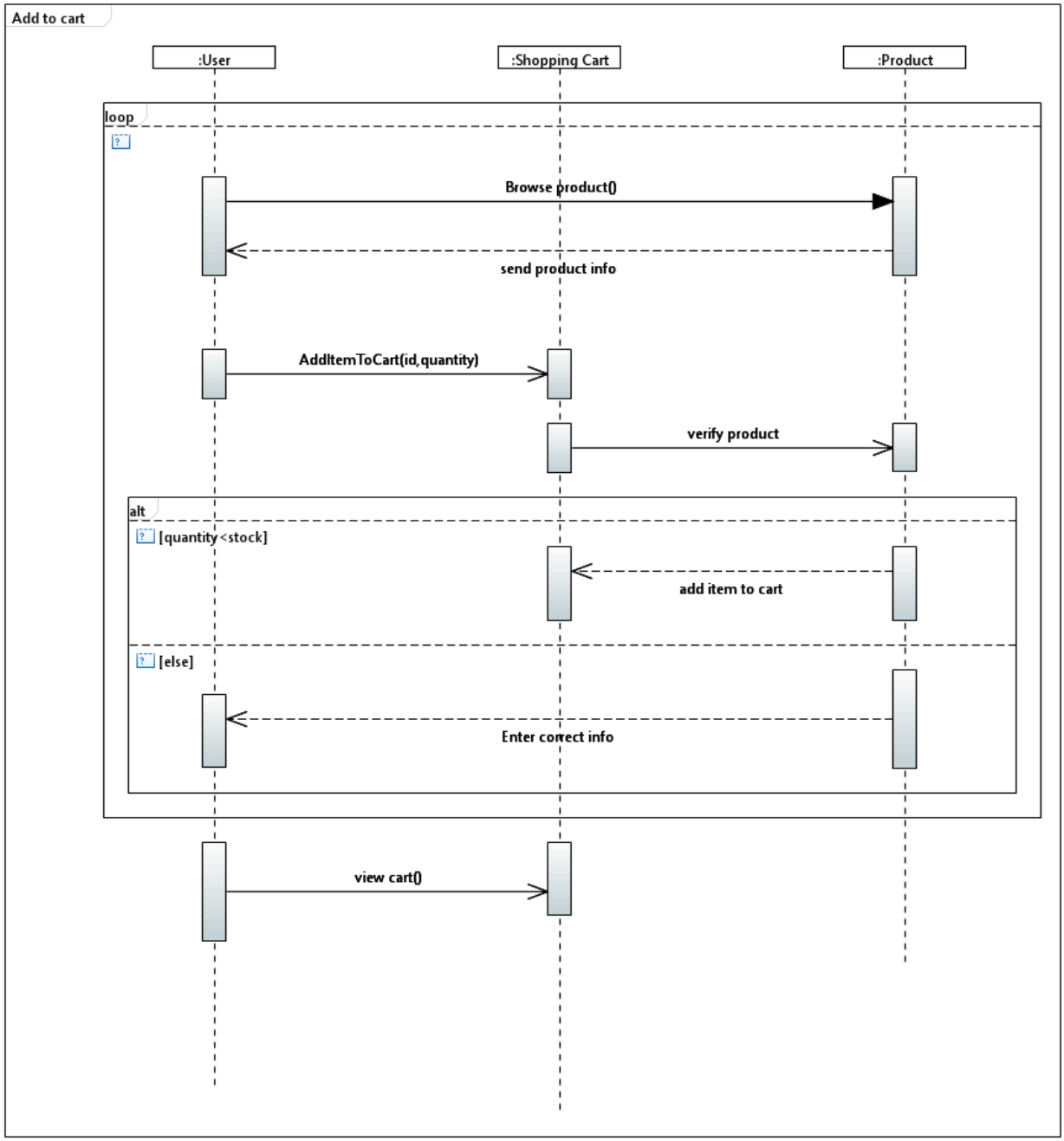
5.1 Database Design

5.1.1 Sequence Diagram

5.1.1.1 <Sequence Diagram 1>

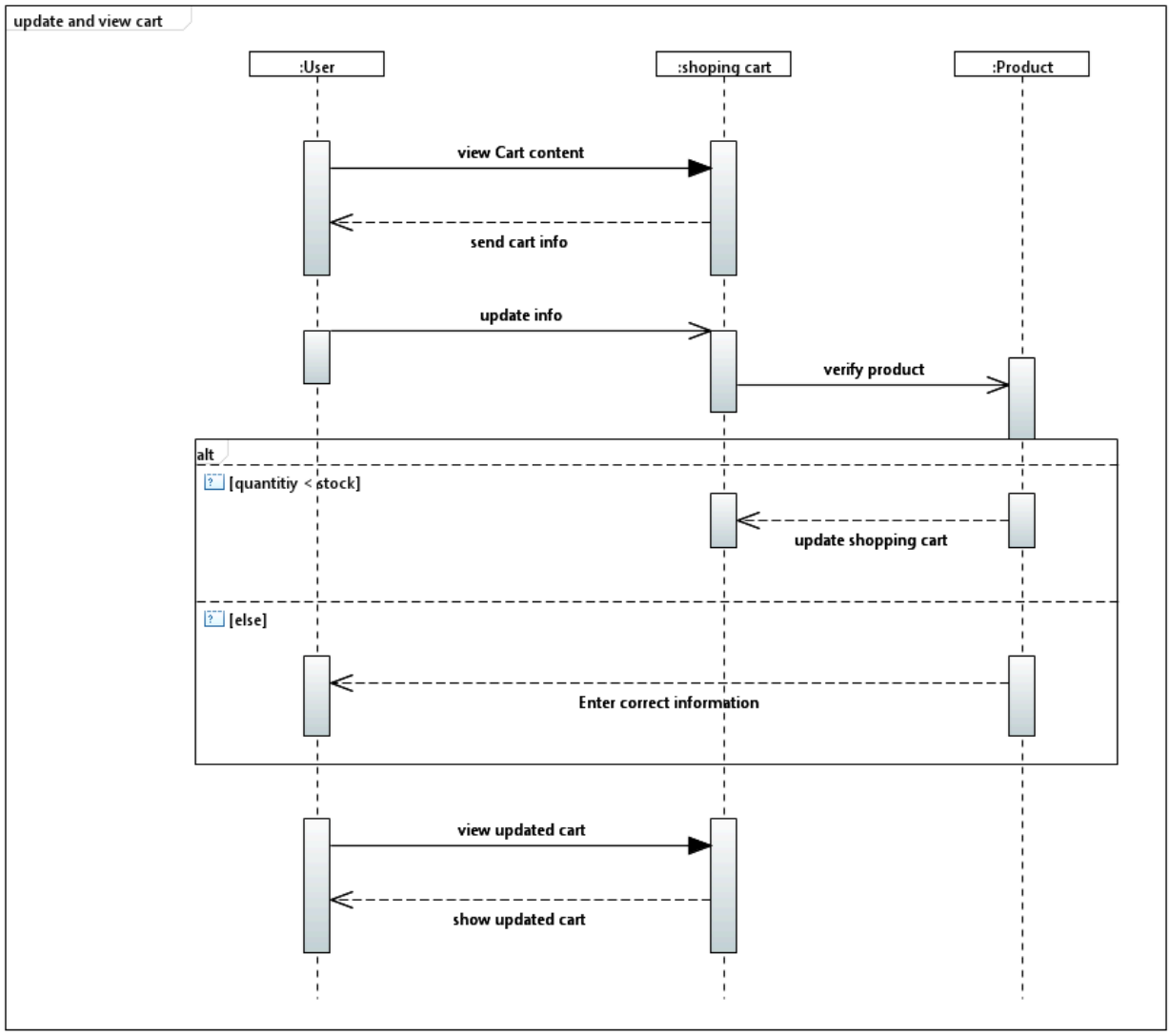
In the sequence diagram for the add to cart function, there are three main objects involved: the user, the shopping cart, and the product. The user initiates the action by selecting a product to add to their shopping cart. The request is then sent to the shopping cart object, which manages the user's cart state. The shopping cart interacts with the product object to verify the availability and details of the selected product. If the product is valid and in stock, the shopping cart updates its contents accordingly by adding the product. Finally, the updated cart status is confirmed back to the user, completing the add to cart

process.



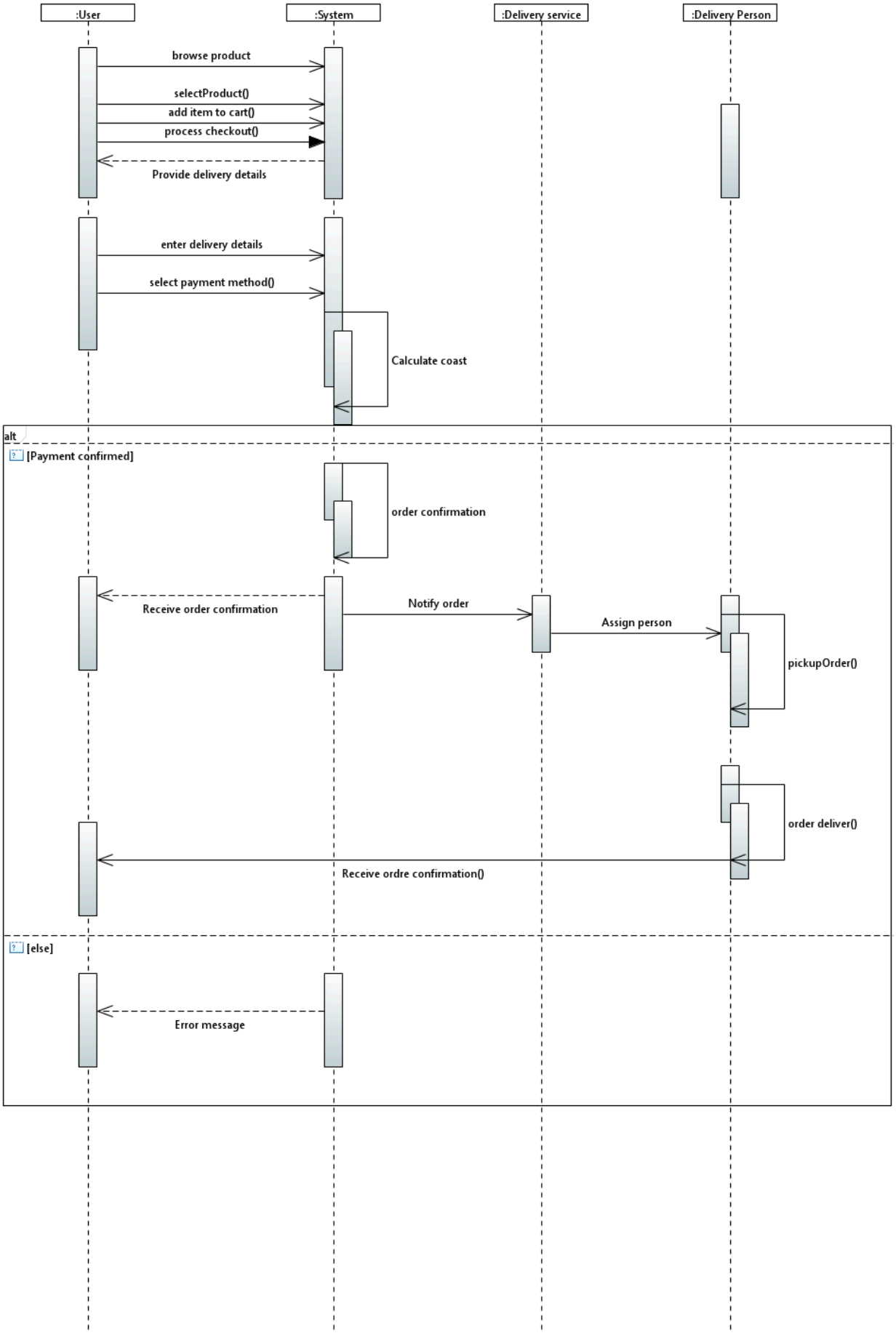
5.1.1.2 <Sequence Diagram 2>

In the sequence diagram for the update cart and view cart functionalities, involving the user, shopping cart, and product objects, the user interacts with their shopping cart to modify or view its contents. In the update cart process, the user adjusts item quantities or removes products, prompting the shopping cart to validate and apply these changes with the product object's assistance. For view cart,the user requests a display of current cart items, leading the shopping cart to retrieve and present this information.



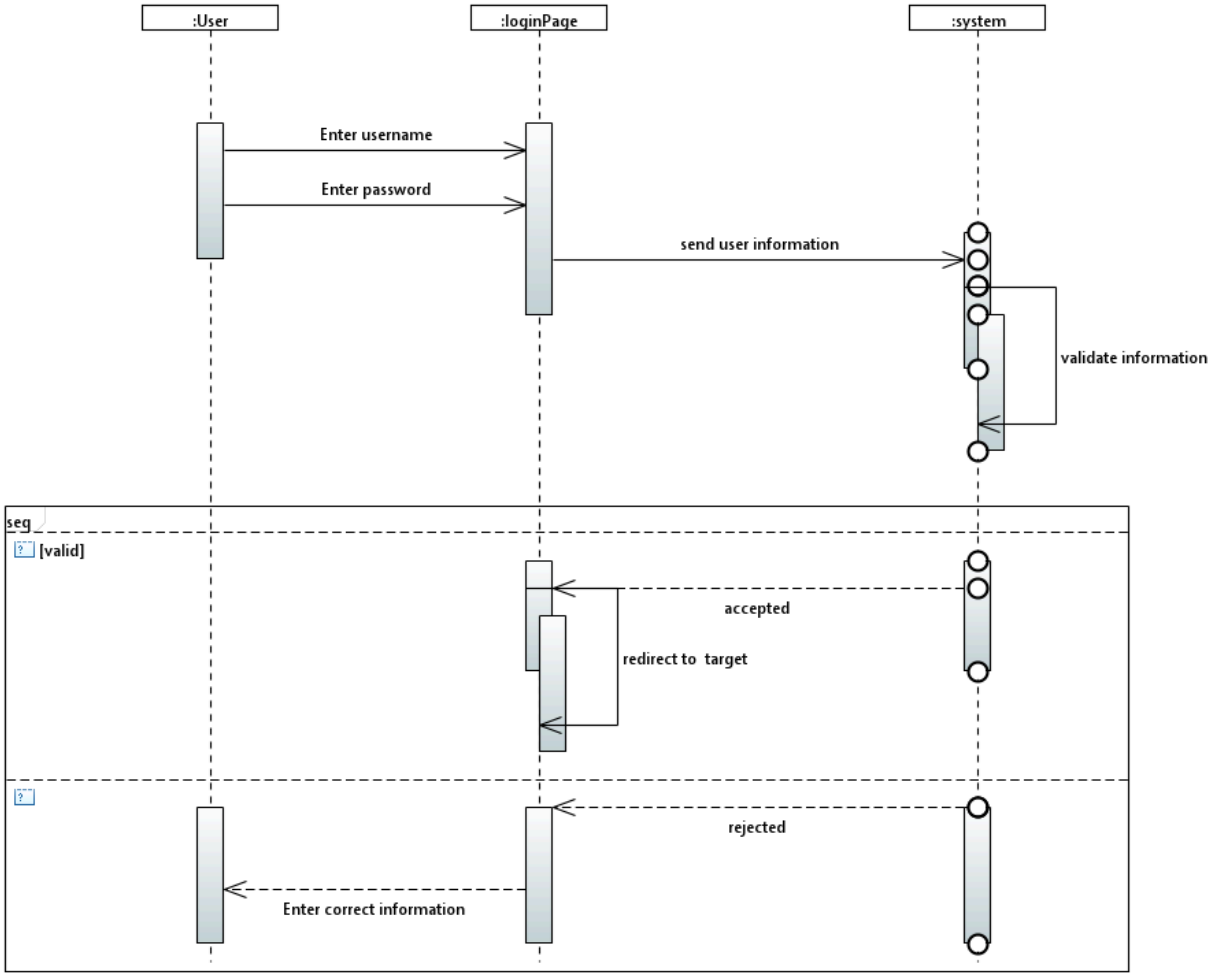
5.1.1.3 <Sequence Diagram 3>

In the sequence diagram for order delivery involving the user, system, delivery person, and delivery service objects, the process begins with the user placing an order through the system. Upon receiving the order request, the system coordinates with the selected delivery service to arrange for a delivery person. The delivery service assigns a delivery person to fulfill the order. The delivery person then retrieves the items from the designated location and proceeds to deliver them to the user's specified address.

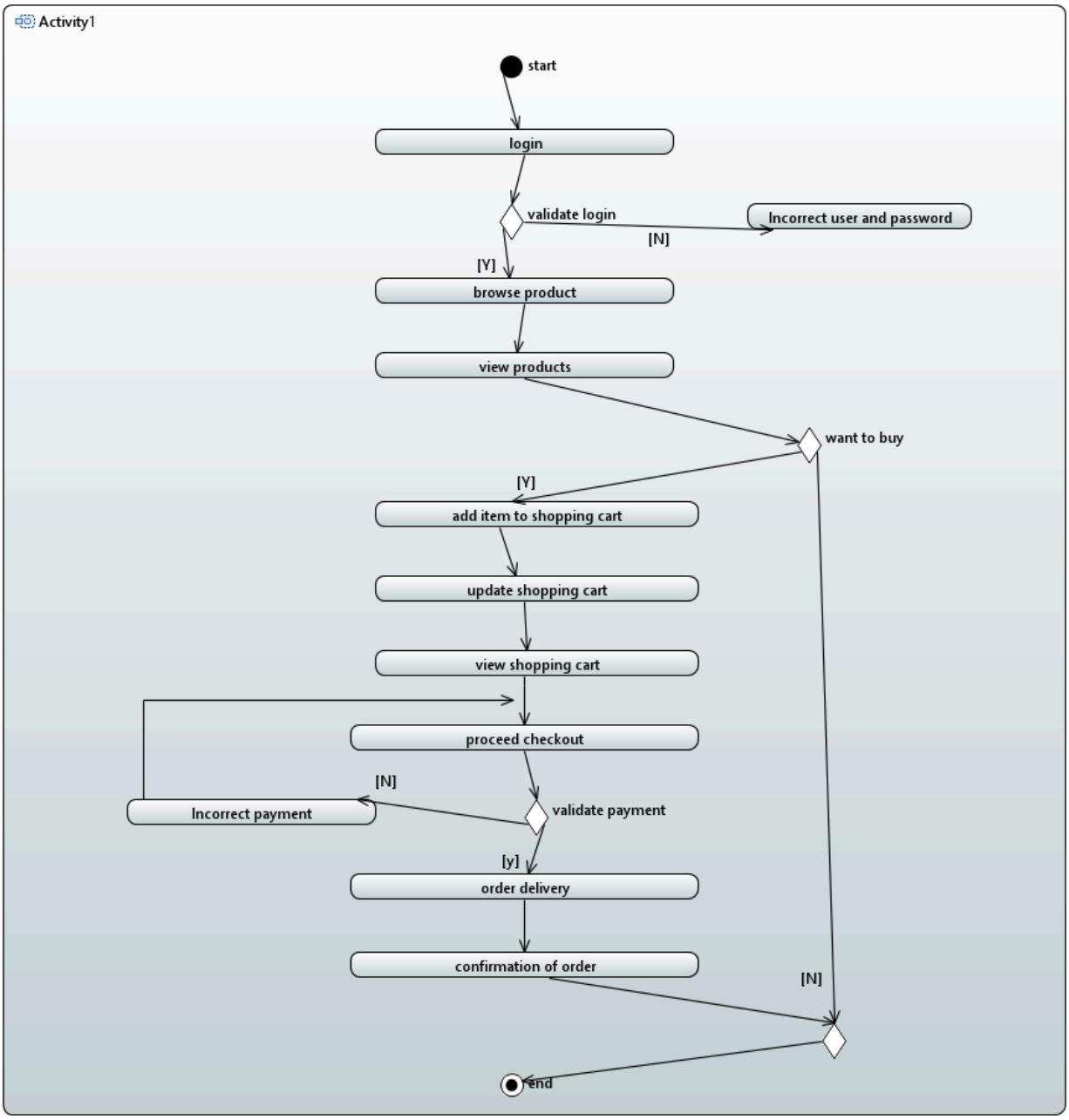

:User
:System
:Delivery service
:Delivery Person
browse product
selectProduct()
add item to cart()
process checkout()
Provide delivery details
enter delivery details
select payment method()
Calculate coast
alt
[Payment confirmed]
order confirmation
Receive order confirmation
Notify order
Assign person
pickupOrder()
order deliver()
Receive ordre confirmation()
[else]
Error message

5.1.1.4 <Sequence Diagram 4>

In the sequence diagram for the login functionality involving the user, login page, and system objects, the process unfolds as follows: The user initiates the login by entering their credentials (username and password) on the login page. The login page sends these credentials to the system for authentication. The system then verifies the credentials against its stored records. If the credentials are valid, the system grants access to the user, allowing them to proceed with the requested action. Conversely, if the credentials are invalid, the system denies access and prompts the user to retry or recover their account.



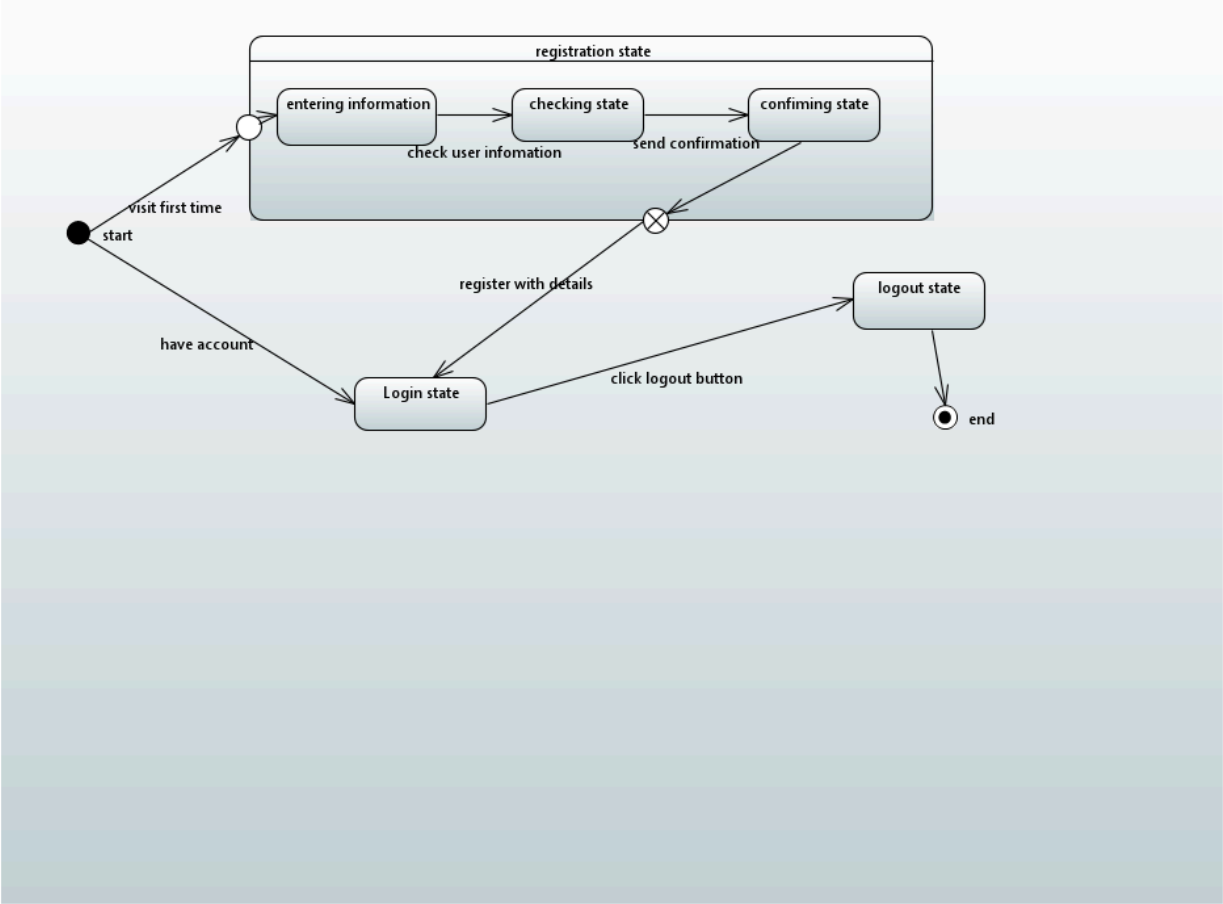
Activity Diagram:



5.1.2 State Diagram

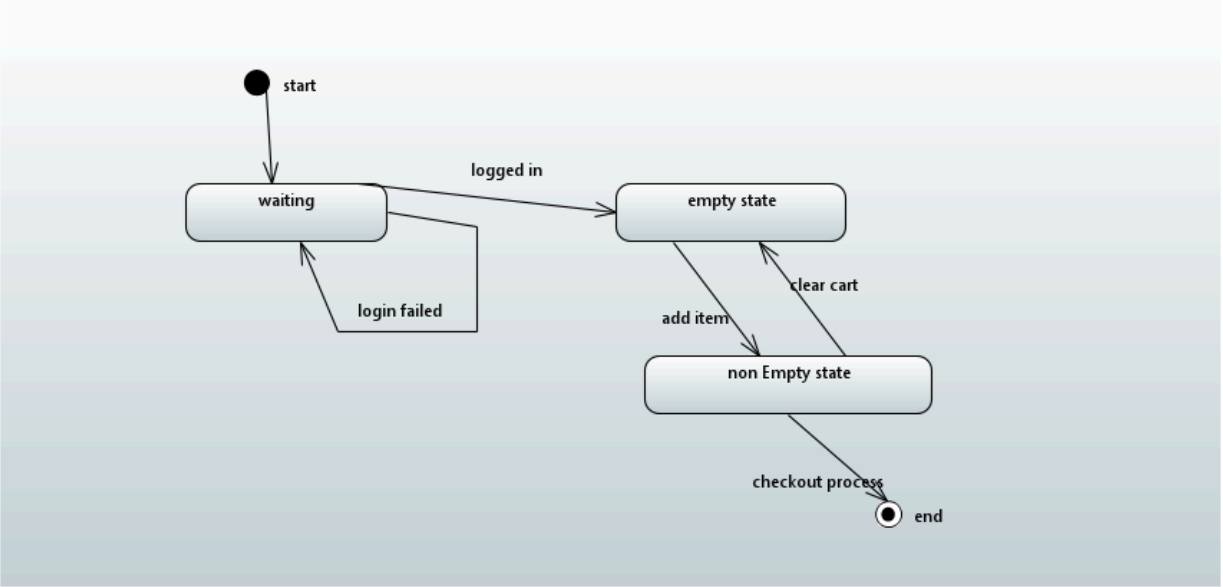
5.1.2.1 <State Diagram 1>

The User state machine diagram includes three states: login, register, and logout, all under the main User state. The login state represents the user being logged in and active within the system. register is the state where users create new accounts, and LOGOUT indicates the user has ended their current session.



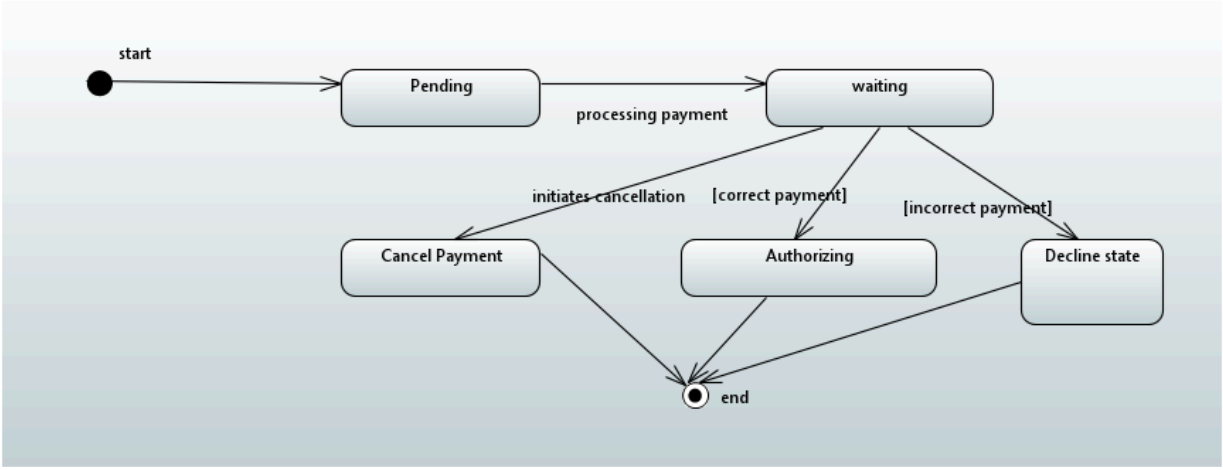
5.1.2.2 <State Diagram 2>

The shopping cart state machine diagram includes three states: waiting, empty, and non-empty, all under the main shopping cart state. The empty state indicates that the cart has no items, while the non-empty state signifies that items have been added to the cart. Transitions between these states occur based on user actions like adding or removing items, reflecting the cart's status and behavior within the system.



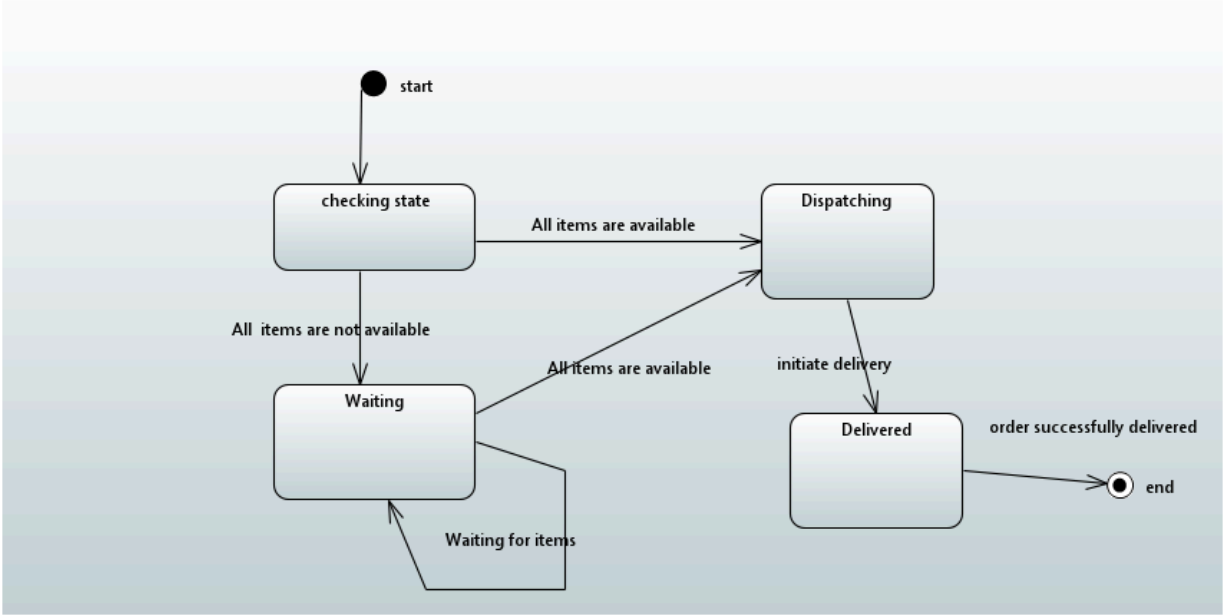
5.1.2.3 <State Diagram 3>

The payment state machine diagram includes five states: pending, waiting, cancel payment, authorizing, and decline, all under the main payment state. The pending state represents an initiated payment awaiting processing. The waiting state indicates the system is ready for user action or further information. Cancel payment allows users to abort the payment process. Authorizing signifies payment verification by the provider, while decline indicates a failed authorization. Transitions reflect system events and user interactions during the payment process.



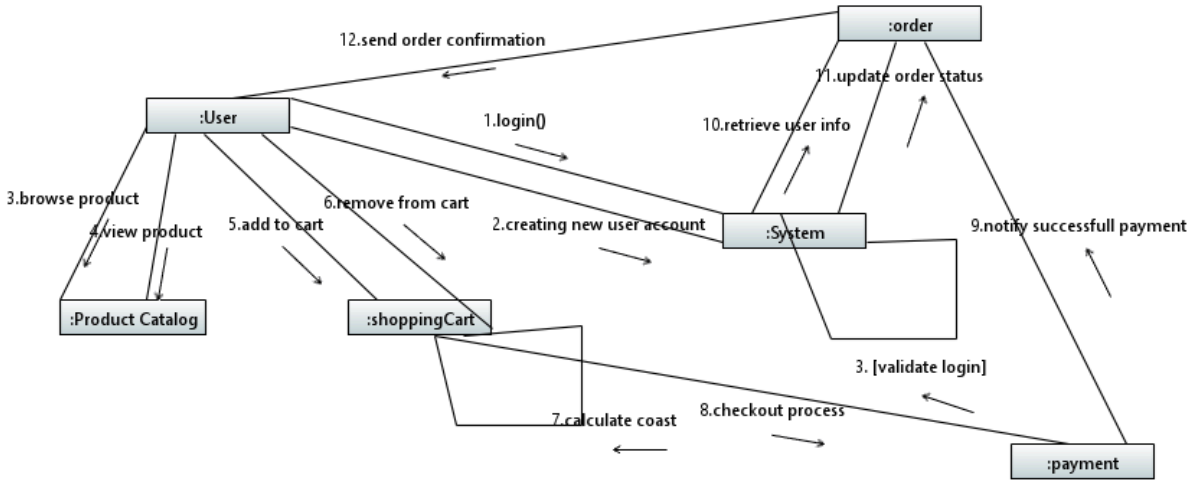
5.1.2.4 <State Diagram 4>

The order state machine diagram includes four states: checking, dispatching, waiting, and delivery, all under the main order state. The checking state verifies order details, followed by dispatching where the order is prepared for shipment. The waiting state indicates that some items in the order are currently out of stock and not available for immediate dispatch. Finally, the delivery state signifies that the order is in transit to its destination.



Collaboration and Communication Diagram:

The collaboration diagram illustrates interactions among objects including user, order, product catalog, shopping cart, system, and payment in a system.The user modifies the contents of the shopping cart by adding or removing items and proceeds to initiate the checkout process. Subsequently, the system updates the order status and sends a confirmation message to the user.



6 References

User Interface style guide:

"Huzawar UI Style Guide",author "Munawar Shereen", Version 1.0, Date October 2, 2023 ,https://www.shopflowdev.com/ui-style-guide .

Contract:
“ShopFlow Service Agreement ”,author “Huzaifa Faran " education , Version 1.0, Date September 7, 2023 https://www.shopflowdev.com/service-agereement .

Standards:
“ISO/IEC 9126 Software Engineering - Product quality ”, author “ISO and IEC ”, Version 2001, Date October 15,2001, https://www.iso.org/standard/22749.html .

System Requirements Specifications:
“ShopFlow System Requirement”,author “Faraz Iqbal”, Version 2.0, Date October 2, 2023 ,https://www.shopflow.com/system-requirement .

Use Case Documentation:
“ShopFlow Use Cases”,author “Huzaifa Faran ”, Version 2.0, Date October 23, 2023 ,https://www.shopflowdevl.com/use-cases .

Version and Scope Documentation:
“ShopFlow Version and Scope ”,author “Huzaifa Faran ”, Version 2.0, Date November 1, 2023, https://www.shopflowdevl.com/version-scope .

The links that were used when the project was being developed are as follows:

https://www.google.com/url?sa=t&source=web&rct=j&url=https://krazytech.com/projects/sample-software-requirements-specificationsrs-report-airline-database&ved=2ahUKEwjqrcGCqeH-AhVLy7sIHSTTANYQFnoECA4QAQ&usg=AOvVaw22CzBwewIGeRwWNEqReDMv

https://www.google.com/url?sa=t&source=web&rct=j&url=https://www.cse.msu.edu/~cse435/Handouts/SRSExample-webapp.doc&ved=2ahUKEwjqrcGCqeH-AhVLy7sIHSTTANYQFnoECAwQAQ&usg=AOvVaw249wPeutmbLpdGGSWXPd77

https://www.google.com/url?sa=t&source=web&rct=j&url=https://www.geeksforgeeks.org/software-requirement-specification-srs-format/amp/&ved=2ahUKEwjqrcGCqeH-AhVLy7sIHSTTANYQFnoECD4QAQ&usg=AOvVaw06tjxud9Q-fHOAwGbnbGdf

https://www.google.com/url?sa=t&source=web&rct=j&url=https://www.reqview.com/doc/iso-iec-ieee-29148-srs-example/&ved=2ahUKEwiEx4-hqeH-AhVn_rsIHdIaDugQFnoECCoQAQ&usg=AOvVaw0C3mot5kYXhvl1kGQqEq4z

7 Appendices

Error Handling Strategy:*Define a detailed error handling strategy for shopFlow, including error codes, messages, and how errors will be communicated to users. This will ensure efficient troubleshooting and user guidance in case of errors.*

User Roles and Permissions:*Clearly define user roles (e.g., admin, regular user) and their corresponding permissions within shopFlow. This will establish access control measures and ensure proper segregation of duties among users.*

Third-Party Integrations:*List and specify any third-party services or APIs to be integrated into shopFlow, outlining the scope and functionalities of each integration. This will ensure seamless interoperability with external systems and enhance the overall functionality of the platform.*

Performance Metrics:*Determine specific performance metrics to be measured for shopFlow, such as response time, throughput, and error rates. Establish acceptable thresholds for each metric to ensure optimal system responsiveness and user experience.*