Ankara Yıldırım Beyazıt University
Department of Computer Engineering

CENG 201 – Object Oriented Programming
Course Project

G20: E-commerce Shopping Cart System

Analysis Report

Muhammet Enes Varol - 22050111041
Abdullah Tülek – 22050111079
Nurgül Yalman- 21050111072
Dilek Miraç Çolak - 21050111046

Instructor: Muhammed Abdullah Bülbül
Teaching Assistant: Elif Şanlıalp, Yusuf Şevki Günaydın

Date: 10/12/2024

-
-

Table of Contents

1. Introduction 2
2. Class-Responsibility-Collaboration (CRC) Cards2
3. Class Diagram 3
4. Conclusion 3

Introduction

- This report provides a comprehensive overview of the design phase of our project, which focused on developing a well-structured, efficient, and maintainable system that adheres to the principles of object-oriented programming (OOP). The main goal of this phase was to ensure that the system design was both robust and extensible, providing a solid foundation for subsequent development activities.

- During the design phase, detailed artifacts that captured the structure and behavior of the system were created. These artifacts included UML class diagrams and Class Responsibility Collaboration (CRC) maps, which were critical in defining the roles, responsibilities, and interactions of each class in the system. Collaboration during this phase ensured that all design decisions were consistent with the overall goals and requirements of the project.

- The report is divided into the following main sections:

- CRC Card: This section describes the main responsibilities assigned to each class and explains how they work together. By outlining these responsibilities, the team ensures a clear division of labor within the system and reduces potential functional overlap or redundancy.

- Class Diagram: Provides a comprehensive UML class diagram that details the structure of the system. The diagram contains the properties, methods, and relationships of all classes, such as: B. Associations, aggregations, and inheritance hierarchies. It provides a visual representation of the interaction and complementarity of system components.

- Conclusion: The final section summarizes the design process and highlights an iterative approach to improving the system architecture. It also recognizes the contributions of each team member and emphasizes the collaborative effort that is essential to achieving a comprehensive design.

- This document serves as a blueprint for the implementation phase, ensuring that all design aspects are clearly defined and ready to be implemented into code. By carefully planning and documenting the design, the team aims to streamline the development process, minimize errors, and maintain consistency throughout the project lifecycle.

Class-Responsibility-Collaboration (CRC) Cards

Class Name	Product
Responsibilities	Manage product details: ID, picture path, explanation, cost, like count, and comments.
	Provide getters and setters for attributes (`getId()`, `setCost(double)`, etc.)
	Allow user interaction: liking or unliking products (`likeProduct()`, `unlikeProduct()`).
	Handle and store user comments with `addComment(QString)` and `getComments()`.
Collaborators	**Cart**: Products can be added to or removed from the cart.
	Customer: Customers interact with products by liking, commenting, or adding them to their favorites.
	MainWindow: Displays products to the user and provides a user interface for interaction (e.g., adding to cart, liking).

Class Name	Cart
Responsibilities	Manage the collection of products in the cart.
	Add products to the cart (`addProduct(Product product)`).
	Remove products from the cart (`removeProductById(int id)`).
	Retrieve the list of products in the cart (`getProducts()`).
	Clear all products from the cart (`clearCart()`).
Collaborators	**Product**: Products are added to and removed from the cart.
	Customer: Each customer owns a cart.
	Payment: Processes the payment for the items in the cart.

Class Name	Customer
Responsibilities	Manage customer details, such as name, surname, email, password, and points.
	Interact with products by liking (`likeProduct(int)`) or adding/removing favorites (`addFavoriteProduct(Product product)`, `removeFavorite(int id)`).
	Retrieve customer information, such as their favorites (`getFavorites()`).
	Manage the associated cart (`getCart()`).
Collaborators	**Product**: Customers can like, add, or remove products from their favorites.
	Cart: Each customer is associated with a cart to manage products they intend to purchase.
	UserManager: Handles customer authentication and registration.

Class Name	NotificationSystem
Responsibilities	Display various types of messages to the user interface.
	Show informational messages (`showInfo(title, message)`).
	Display warning messages (`showWarning(title, message)`).
	Handle and display error messages (`showError(title, message)`).
Collaborators	**MainWindow**: Provides the parent widget for notifications and utilizes this class to display messages in response to user actions or system events.

Class Name	Payment
Responsibilities	Manage the payment process for the customer's cart.
	Apply discounts to the total amount (`applyDiscount(Discount discount)`).
	Calculate the total amount for the cart (`getTotal()`).
	Retrieve associated customer, cart, and discount details (`getCustomer()`, `getCart()`, `getDiscount()`).
	Generate and return payment record details (`getRecordDetails()`).
Collaborators	**Cart**: Processes payments for the products in the cart.
	PurchaseRecord: Generates a record of the payment and stores the purchase details.
	OrderDetail: Provides order-specific details and points after successful payment.

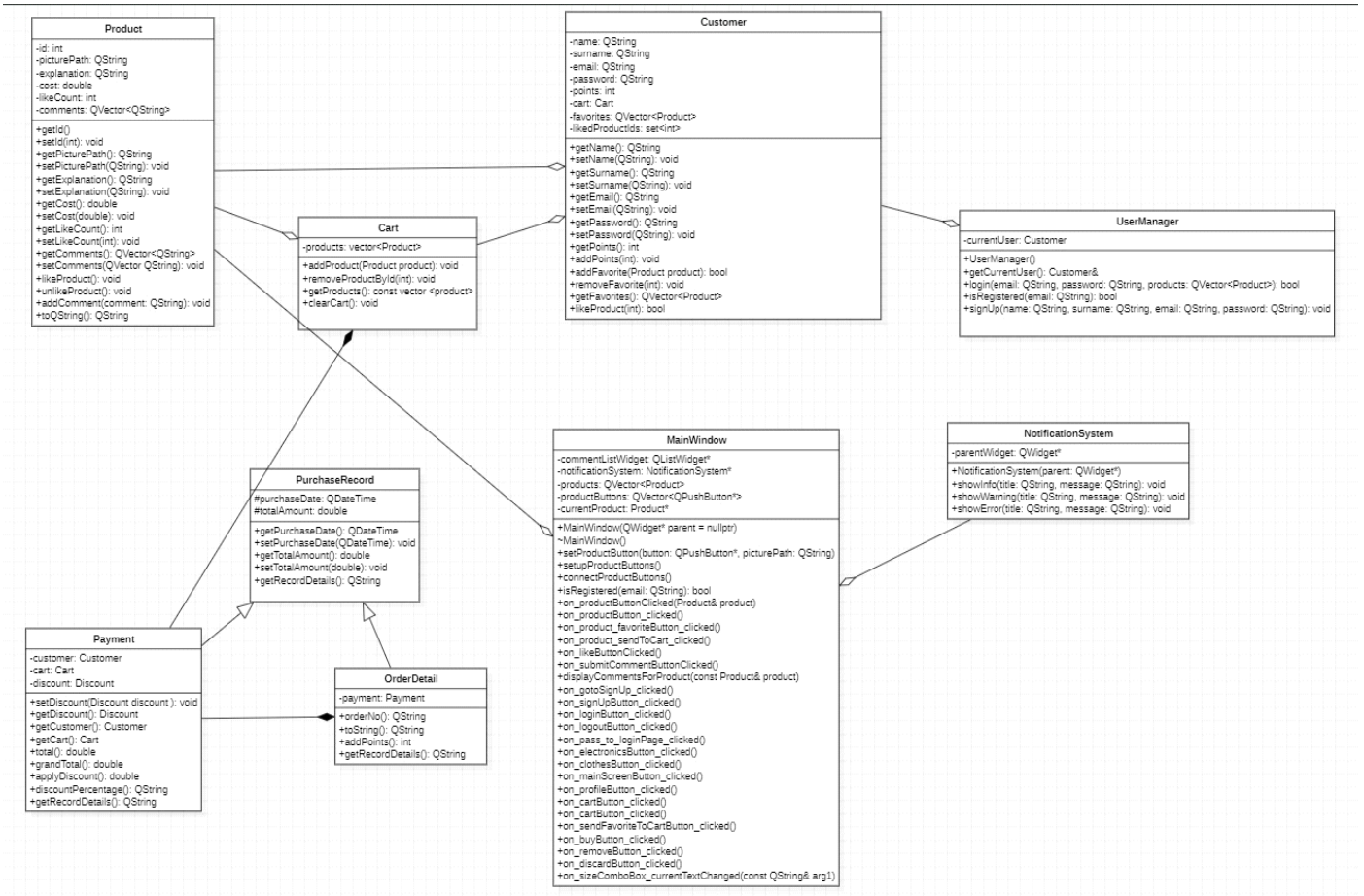
Class Name	PuchaseRecord
Responsibilities	Store details of a purchase, including the date and total amount (`setPurchaseDate(QDateTime)`, `setTotalAmount(double)`).
	Provide access to purchase details (`getPurchaseDate()`, `getTotalAmount()`).
	Generate a detailed record of the purchase (`getRecordDetails()`).
Collaborators	**Payment**: Receives payment information and generates a corresponding purchase record.
	OrderDetail: Links purchase records with specific order details for further processing.

Class Name	OrderDetail
Responsibilities	Store detailed information about an order, including payment and record details.
	Link the order to the payment (`setPayment(Payment payment)`).
	Generate and return order-specific details (`getRecordDetails()`, `toString()`).
	Add points for the customer based on the order total (`addPoints()`).
Collaborators	**PurchaseRecord**: Retrieves and uses purchase record details to complete the order.
	Payment: Handles payment information and associates it with the order.

Class Name	MainWindow
Responsibilities	Display products and manage user interactions with the graphical interface.
	Handle product-related actions, such as liking, adding to cart, or displaying comments (`on_productButton_clicked(Product product)`, `on_likeButton_clicked()`, etc.).
	Connect buttons and widgets to their respective functionalities (`setupProductButtons()`, `connectProductButtons()`).
	Show notifications and handle errors (`NotificationSystem` integration).
Collaborators	**Product**: Displays products and enables user interactions, such as liking and adding comments.
	NotificationSystem: Sends user notifications, such as success or error messages.

Class Name	UserManager
Responsibilities	Manage customer authentication and registration.
	Retrieve the currently logged-in customer (`getCurrentUser()`).
	Handle login functionality (`login(email, password, products)`).
	Check if an email is already registered (`isRegistered(email)`).
	Handle customer sign-up (`signUp(name, surname, email, password)`).
Collaborators	**Customer**: Maintains the current user session and interacts with customer data for login, registration, and authentication.

- Class Diagram



• Conclusion

The design phase of this project represents a milestone, bridging the gap between conceptual planning and implementation. It focused on crafting a robust, extensible system architecture. This phase not only addressed immediate project requirements but also ensured the system's adaptability for future enhancements, emphasizing scalability and maintainability.

Through collaborative effort, the team produced detailed design artifacts that captured the essence of the system's structure and behavior. UML class diagrams and CRC cards were pivotal in visualizing and organizing the system's components, fostering clarity and reducing redundancy. These tools, combined with iterative refinement, ensured a well-rounded design that aligns with the project's goals.

In conclusion, the design phase of this project has laid a solid foundation for the implementation phase, ensuring that all critical elements are well-structured and aligned with object-oriented programming principles. The use of UML class diagrams and CRC maps has enabled the team to clearly define the system's structure, roles, and interactions, minimizing redundancy and fostering clarity.

Below is a summary of the contributions made by each team member:

- **Dilek Miraç Çolak:**

  o Developed the Class-Responsibility-Collaboration (CRC) cards, defining the main responsibilities of each class and ensuring their roles were clearly outlined. This effort helped to minimize functional overlap and establish a clear division of labor within the system.

- **Abdullah Tülek:**

  o Authored the "Conclusion" section, summarizing the design process and emphasizing the collaborative effort of the team. His contribution provided a clear summary of the design process and underscored its alignment with the project's overall goals.

- **Muhammet Enes Varol:**

  o Wrote the "Introduction" section, providing a comprehensive overview of the design phase and its objectives. This section effectively framed the report's purpose and set the stage for subsequent sections.

- **Nurgül Yalman:**

  o Designed the UML Class Diagram, detailing the system's structure, properties, methods, and relationships. Her work provided a visual representation of the system's components, emphasizing their interactions and complementarity.