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: 03/12/2024

•

•

Table of Contents

• Introduction

The project aims to develop a comprehensive e-commerce platform that facilitates a seamless online shopping experience for users. The system allows users to browse products, add items to their shopping carts, complete purchases, and manage returns. With a focus on user-friendliness, the platform is designed to be accessible to a broad audience, ensuring functionality across different devices and operating systems.

This report details the project's scope, requirements, and design. It covers the functional and non-functional requirements, system scenarios, use cases, and class models, providing a holistic view of the development process. By adhering to best practices in software development, this project strives to deliver a reliable and secure solution for online shopping.

☐ Requirements

• Functional Requirements

User Registration and Authentication:

- Users must be able to register by providing necessary information such as email, password, and name.
- Registered users can log in using their credentials and log out when needed.
- The system should include a password recovery mechanism.

Product Browsing and Search:

- Users must be able to search for products using keywords and apply filters like category, price range, or brand.
- The system should display detailed product information, including price, description, and availability.

Shopping Cart:

- Users should be able to add, remove, and update quantities of products in their shopping cart.
- The system should calculate the total price, including discounts if applicable.

Order Placement and Payment:

- Users should be able to place orders and securely provide payment details.
- The system should confirm the order and provide an estimated delivery time.

Order Tracking:

- The system must allow users to track their orders and view their delivery status.

Return and Refund Process:

- Users must be able to initiate returns for purchased products and provide a reason for the return.
- Refunds should be processed, and users should be notified upon completion.

Notifications:

- The system should send notifications for order confirmations, shipping updates, and return approvals.

Non-Functional Requirements

- Usability: The interface should be user-friendly and easy-to-use.
- Performance: The system should respond quickly and handle operations without delay.
- Security: User data and shopping records must be securely stored.
- Scalability: The system should be easily expandable to meet future needs.
- Compatibility: The system should operate on different operating systems.
- Maintainability: The system should be easy to maintain and update.

System Models

Scenarios

Scenario 1: User Makes a Purchase
Steps:

1. The User logs into the system.
2. Searches for a product using the search bar or browses through categories.
3. Selects a product from the search results and views its details.
4. Adds the product to the shopping cart by clicking "Add to Cart".
5. Proceeds to the checkout page and reviews their cart.

6. Applies a discount code (if available) and calculates the total price.
7. Enters shipping and payment details.
8. Confirms the purchase by clicking the "Make Payment" button.
9. The system processes the order and notifies the User.

Scenario 2: User Returns an Item
Steps:

1. The User logs into the system and views their order history.
2. Selects the product they wish to return.
3. Clicks the "Return Product" button and provides a reason for the return.
4. The system processes the return request and updates the order status.
5. The User monitors the return progress through their profile.
6. Once approved, the refund is processed, and the system notifies the User.

- Use Cases

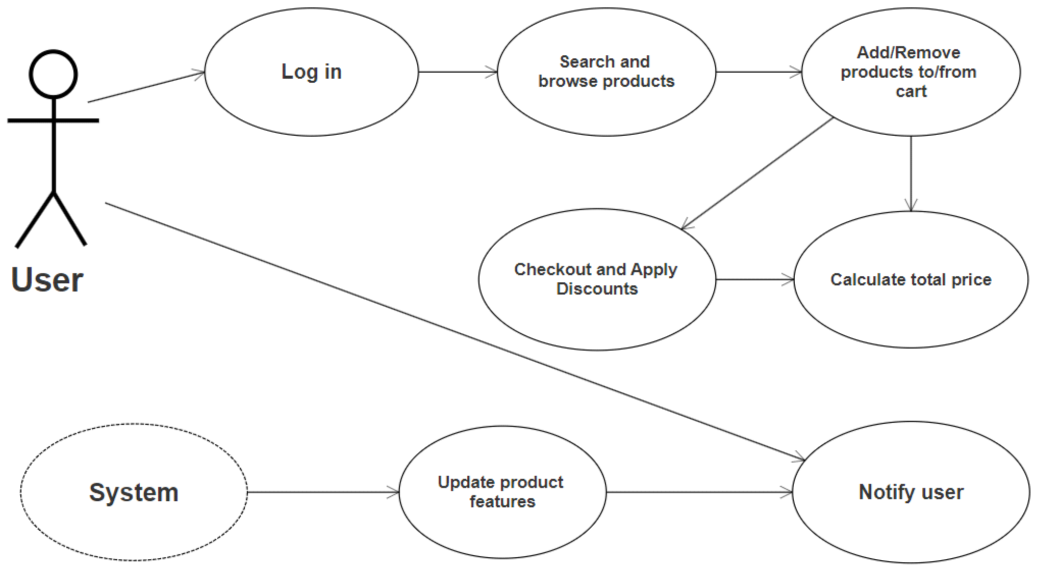
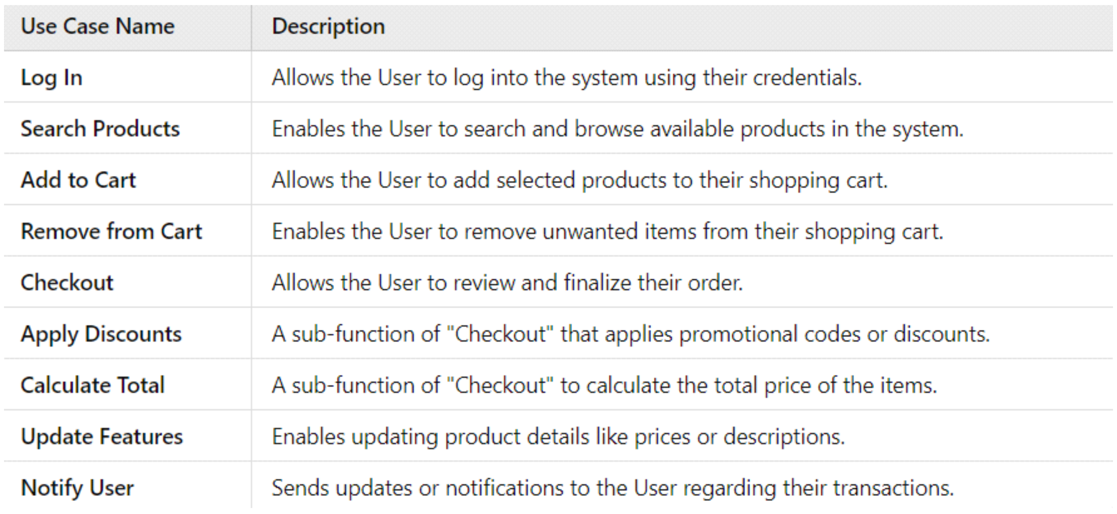

User
Log in
Search and browse products
Add/Remove products to/from cart
Checkout and Apply Discounts
Calculate total price
System
Update product features
Notify user

Figure 1: UML Use Case Diagram for the E-Shopping Cart System.



Use Case Name	Description
Log In	Allows the User to log into the system using their credentials.
Search Products	Enables the User to search and browse available products in the system.
Add to Cart	Allows the User to add selected products to their shopping cart.
Remove from Cart	Enables the User to remove unwanted items from their shopping cart.
Checkout	Allows the User to review and finalize their order.
Apply Discounts	A sub-function of "Checkout" that applies promotional codes or discounts.
Calculate Total	A sub-function of "Checkout" to calculate the total price of the items.
Update Features	Enables updating product details like prices or descriptions.
Notify User	Sends updates or notifications to the User regarding their transactions.

Figure 2:Use Case Chart for the E-Shopping Cart System.

- ## Object and Class Model

Main Classes:

User:
Attributes: userID, name, email, password, userType
Methods: login(), logout(), searchProduct()

Customer: Inherits from User.
Methods: buyProduct(), returnProduct(), viewNotifications()

Product:
Attributes: productID, name, brand, category, availability
Methods: updateAvailability()

PuchaseRecord:
Attributes: purchaseID, productID, userID, purchaseDate, estimatedDeliveryDate
Methods: calculateEstimatedTime()

Notification:
Attributes: notificationID, userID, message, date
Methods: sendNotification()

Cart:
Attributes: addToCart, removeFromCart, checkOut, applyDiscounts, calculateTotal
Methods: checkOut(), applyDiscount(), calculateTotal()

Design Patterns Used:

Observer Pattern:

Purpose: To automatically send notifications for updates during delivery and shipping.

Implementation: PurchaseRecord objects should observe the Notification system.

Factory Pattern:

Purpose: To dynamically create different user types.
Implementation: A UserFactory class creates Member or Librarian instances based on the required user type.

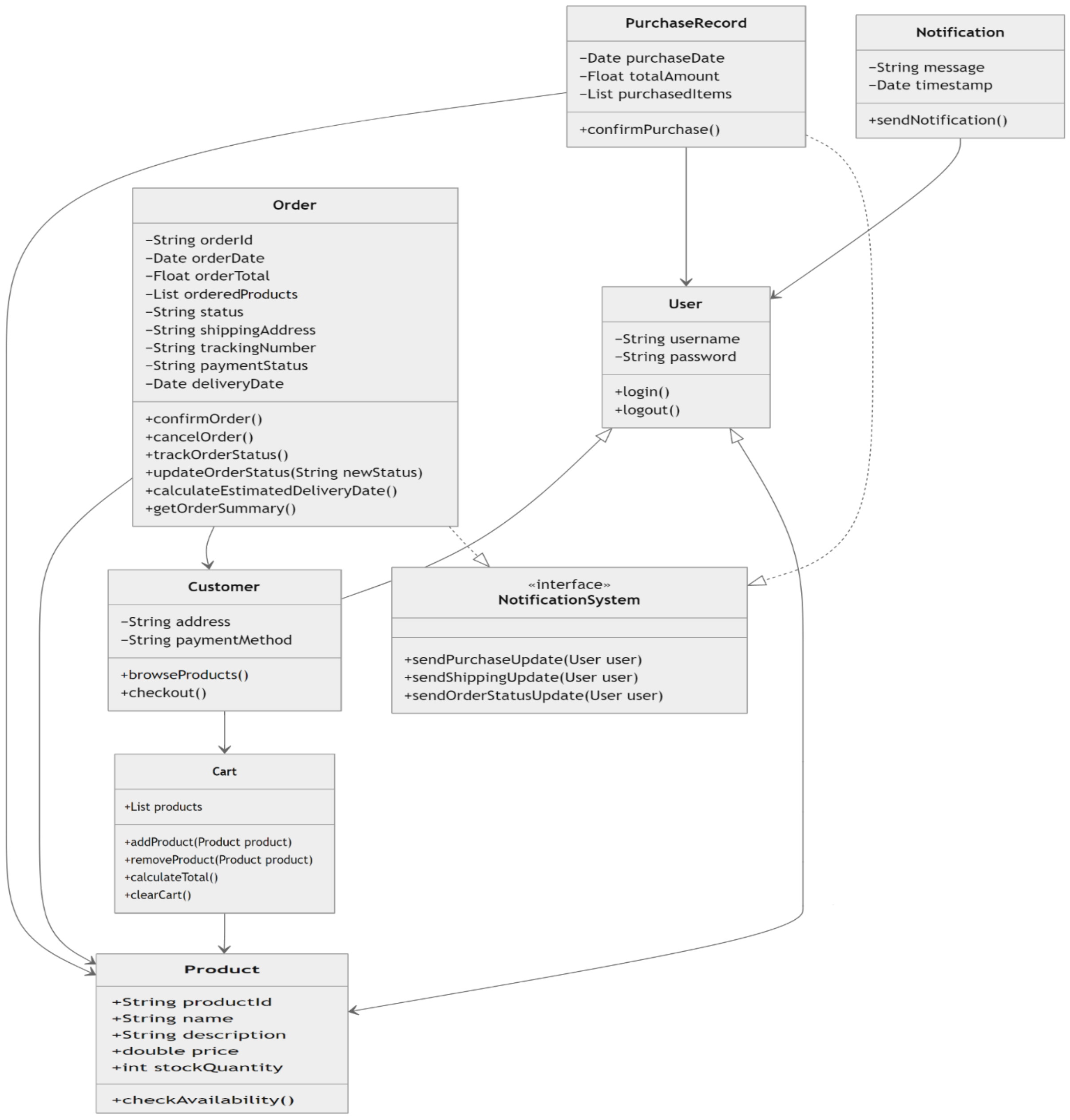
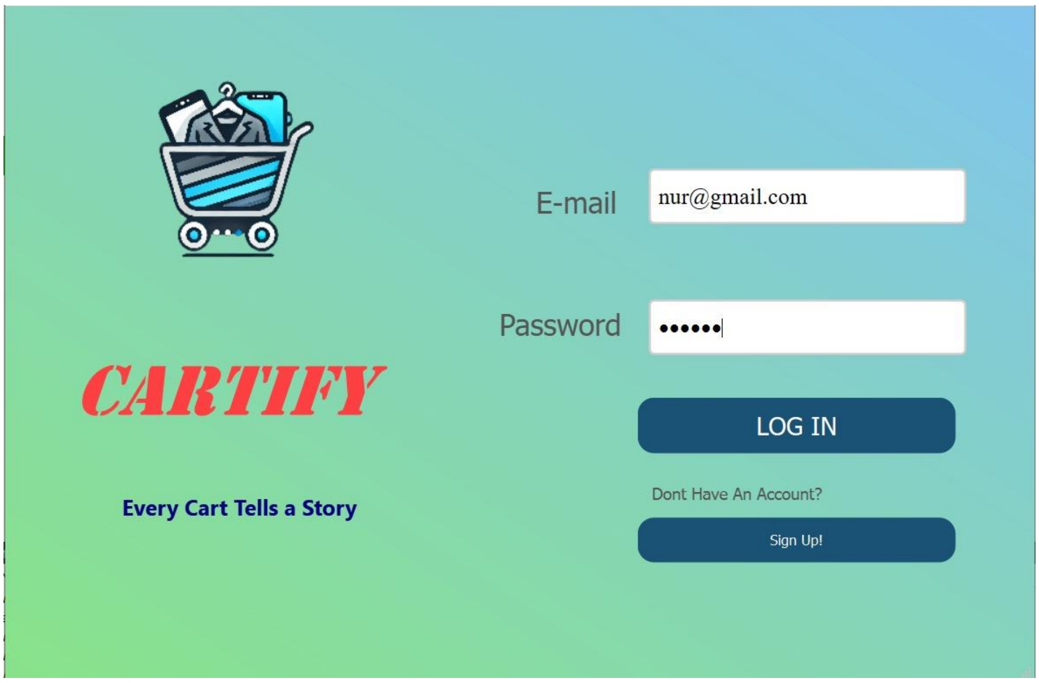


Figure 3:UML Class Diagram for the E-Shopping Cart System.

- ## User Interfaces

The aim is to provide a shopping experience where users can easily log in, view products, add them to their carts, save them to favorites and leave comments. The focus is on user-friendly interfaces with a simple design, and basic functions are implemented in a simple but effective way.



E-mail nur@gmail.com

Password ••••••

Figure 4: Login page.

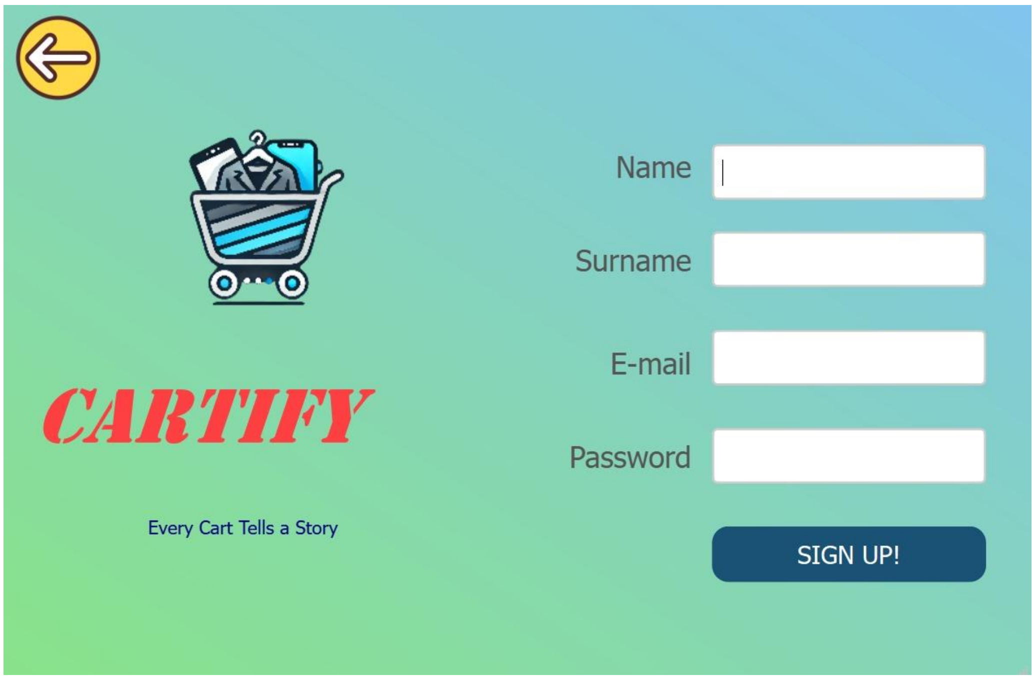


Figure 5: Sign up page.

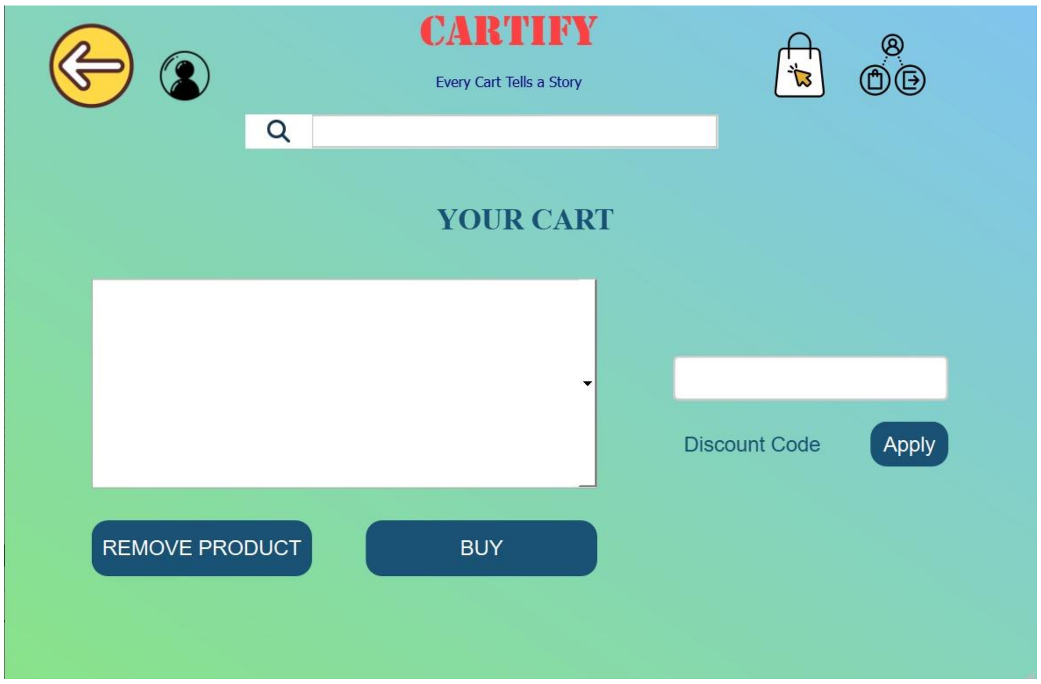


Figure 6: Purchase page.

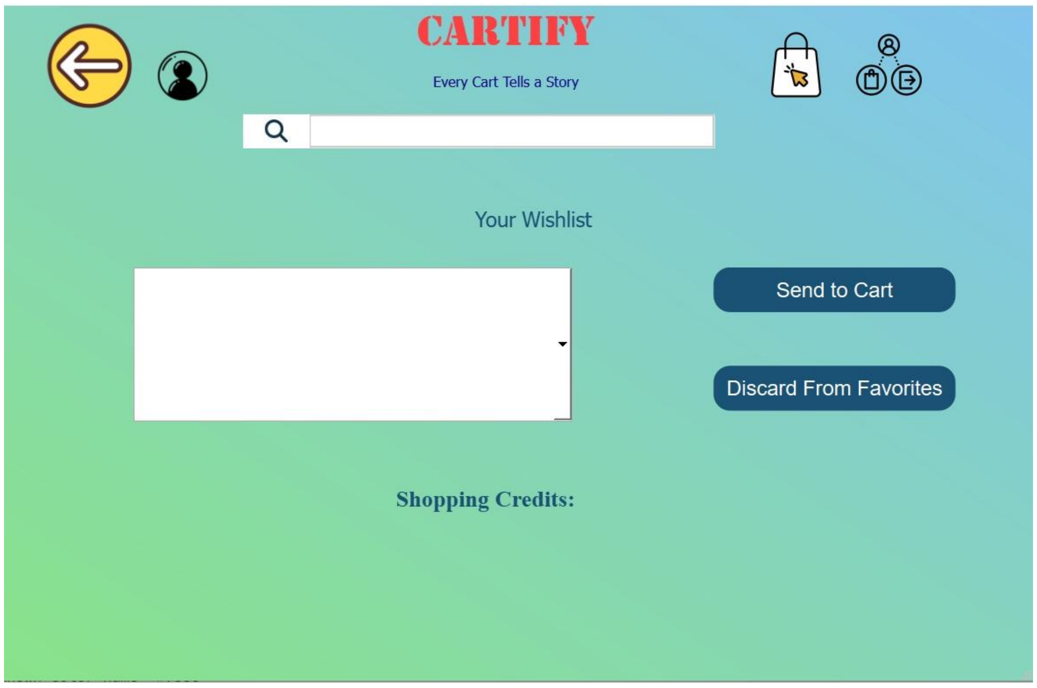


Figure 7: Wishlist page.

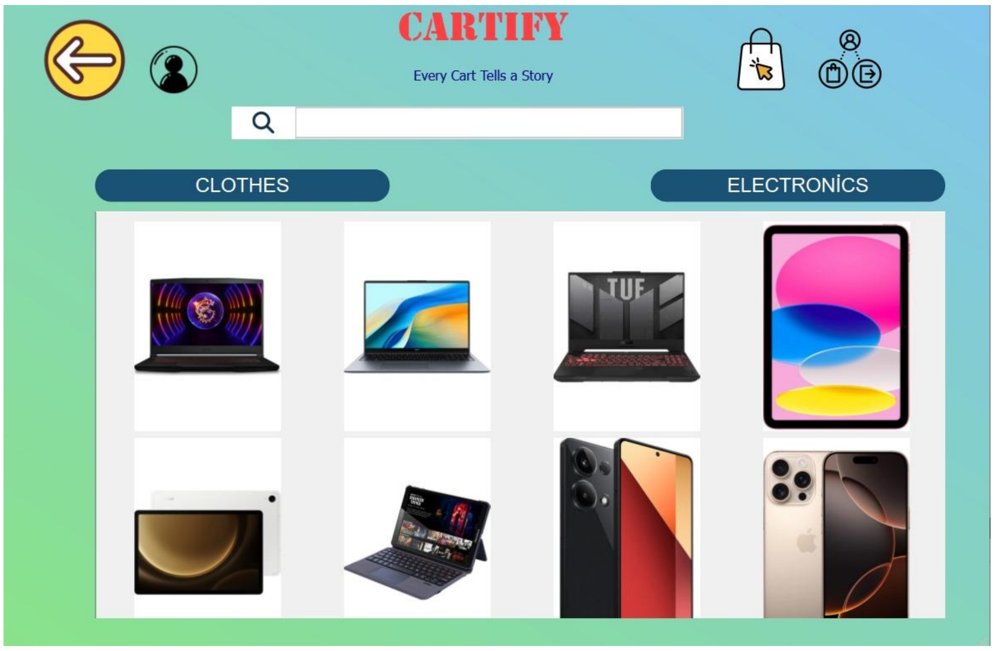


Figure 8: Page with product types.

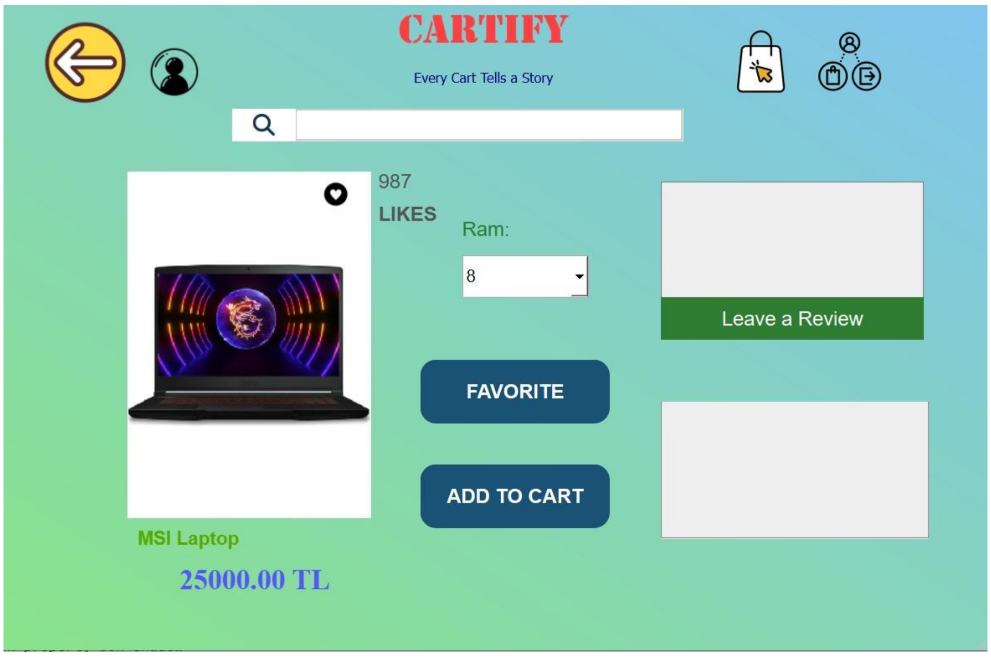


Figure 9: Product Page.

• Conclusion

This report provides a comprehensive overview of the e-commerce project, detailing its functional and non-functional requirements, system models, and design considerations. Each section of the report was crafted with the collaborative effort of the team, reflecting our collective dedication to delivering a user-friendly and robust system. Below is a summary of the contributions made by each team member:

- **Dilek Miraç Çolak**:
    - Authored the "Introduction" section, providing a concise overview of the project and its structure.
    - Compiled the "Functional Requirements" section, detailing the expected functions of the system.
    - Authored this "Conclusion" section, summarizing the report and acknowledging team contributions.
- **Abdullah Tülek**:
    - Documented the "Non-Functional Requirements," focusing on aspects such as performance, capacity, reliability, and usability.
    - Developed the "Object and Class Model" section, outlining the relationships between system classes with supporting diagrams.
- **Muhammet Enes Varol**:
    - Created detailed system scenarios in the "Scenarios" section, illustrating the system's functionality step by step.
    - Designed and described use-case diagrams in the "Use Cases" section, clarifying how users interact with the system.
- **Nurgül Yalman**:
    - Designed and created mockups for the 'User Interfaces,' researching the Qt framework to explore its GUI development capabilities including, layout management, and widget library. Focused on aligning the interface design with the system's purpose while ensuring a user-friendly and visually appealing experience.