Green University of Bangladesh

Department of Computer Science and Engineering (CSE)
Semester: (Fall, Year: 2024), B.Sc. in CSE (Day)

Inventory management system

Course Title: Data structures lab
Course Code: CSE 206
Section: 232 D6

Students Details

Name	**ID**
Md. Mehadi Hasan	232002102

Submission Date: 12-26-2024
Course Teacher's Name: Md. Asiqussalehin

[For teachers use only: Don't write anything inside this box]

Lab Project Status	
Marks:	**Signature:**
Comments:	**Date:**

Contents

Chapter 1

Introduction

1.1 Overview

The project aims to develop an efficient and scalable inventory management system tailored for modern retail and e-commerce environments. An inventory management system is crucial for businesses to efficiently track and manage stock levels, orders, and product data. The system will allow users to browse and purchase products, track discounted items, and manage administrative tasks such as adding or removing products, making it a comprehensive solution for small to medium-sized businesses.

1.2 Motivation

The motivation behind this project is rooted in the increasing complexity and importance of inventory management for businesses. As e-commerce and retail businesses grow, manual inventory management becomes inefficient and prone to errors. This project's goal is to automate the process and provide an interface that simplifies stock management. By choosing this project, we aim to provide an effective tool that helps businesses optimize stock management, enhance user experience, and reduce human errors in tracking products and sales [**?**].

1.3 Problem Definition

1.3.1 Problem Statement

The primary problem that this project addresses is the inefficiency in traditional inventory management systems, especially for small to medium-sized businesses. Businesses often struggle with manually tracking stock levels, sales, and product data, which leads to errors and potential loss of revenue. The system will aim to resolve this issue by automating the process of inventory tracking, user management, and purchase handling, providing a more reliable, user-friendly, and efficient solution for businesses.

1.3.2 Complex Engineering Problem

The problem can be broken down into several complex engineering challenges that require careful planning and execution. These challenges involve understanding how to design an efficient and scalable database for storing product and user data, building an interface that is intuitive and easy to use for both administrators and customers, and ensuring that the system can handle multiple conflicting requirements, such as ease of use and system performance.

1.4 Design Goals/Objectives

The main objectives of this project are:

- To design a user-friendly inventory management system that meets the needs of small to medium-sized businesses.
- To implement an automated inventory tracking system that reduces human errors and improves stock management.
- To provide a secure login system for administrators and users, ensuring data integrity and privacy.
- To develop a system that can handle both regular and discounted product management.
- To create a modular, scalable system that can be easily expanded to accommodate more features or products in the future.

1.5 Application

This inventory management system will have direct applications in the real world for small to medium-sized businesses in retail and e-commerce sectors. Businesses will be able to use the system to manage their stock efficiently, track sales and purchases, and ensure that inventory levels are optimized. The system can be deployed as a standalone application or integrated with existing business solutions to further streamline business operations.

The system will be particularly useful for businesses with fluctuating inventory levels or seasonal demand, as it can automatically update stock quantities and offer discounts on selected products. Additionally, administrators will have the ability to analyze user purchase data to make informed decisions about stock levels, pricing strategies, and product offerings.

Table 1.1: Summary of the attributes touched by the mentioned projects

Name of the P Attributes	**Explain how to address**
P1: Depth of knowledge required	A deep understanding of database management systems, user interface design, and software engineering principles is required to build a robust inventory management system. This includes knowledge of programming languages like C and database management techniques.
P2: Range of conflicting requirements	Balancing ease of use for both administrators and customers with the technical performance of the system can be challenging. While customers may want a smooth, quick experience, administrators may require detailed control over inventory data.
P3: Depth of analysis required	The project requires a deep analysis of how to model the system's database to efficiently store and retrieve product and user information. It also requires analyzing user needs and how to create an interface that satisfies both user experience and functional requirements.
P4: Familiarity of issues	Familiarity with inventory management problems, such as stockouts, overstocking, and supply chain inefficiencies, will help in designing features like real-time stock updates, inventory alerts, and automated reordering.
P5: Extent of applicable codes	Knowledge of software development best practices and coding standards will ensure the system is both maintainable and scalable. Additionally, knowledge of relevant industry standards for database design and user security is necessary for the integrity and performance of the system.
P6: Extent of stakeholder involvement and conflicting requirements	The project requires balancing the needs of various stakeholders, including administrators who need full control of the system and customers who require an easy-to-use, intuitive interface. Conflicting requirements like system performance versus user convenience need to be handled carefully.
P7: Interdependence	The various components of the system, including the user interface, backend database, and inventory management features, are highly interdependent. Changes to one part of the system, such as adding a new feature or adjusting the database structure, can affect the entire system.

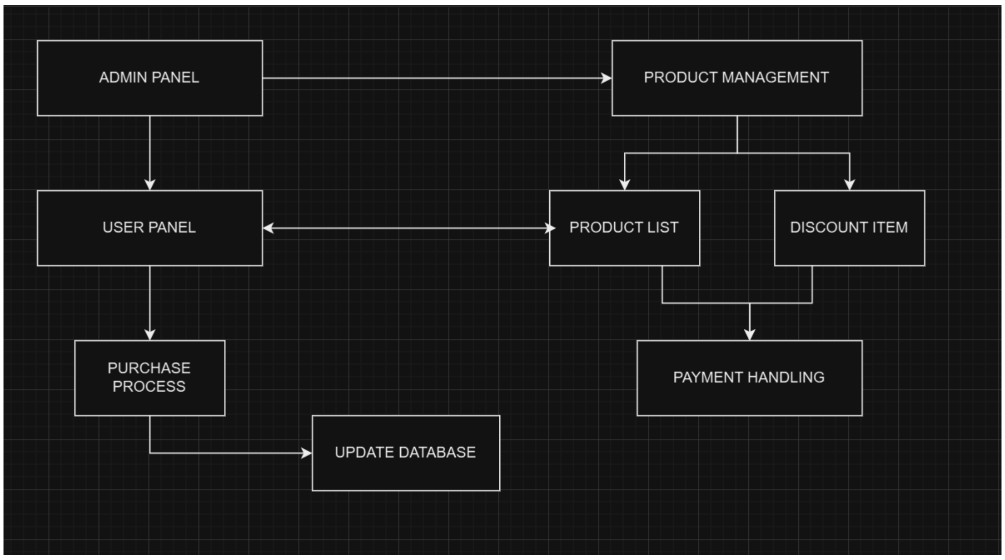


Figure 1.1: Schematic of Inventory Management System Workflow

1.5.1 Real-World Applications and Impact

The system will streamline the sales process for customers and offer businesses valuable insights into their operations. By reducing inventory discrepancies, optimizing stock levels, and offering discounts, businesses will be able to enhance profitability while improving customer satisfaction.

Chapter 2

Design/Development/Implementation of the Project

2.1 Introduction

This section provides an overview of the project, focusing on the development of a product management and purchase system. The system allows users and administrators to interact with a catalog of products. Admins can manage product details, apply discounts, and view user information, while users can browse products, make purchases, and track their orders. This project was developed to improve the user experience in e-commerce systems and streamline product management for administrators.

2.2 Project Details

The project is designed to facilitate both users and admins in a smooth interaction with the product catalog and order management system. The core functionality of the system is to support product browsing, user registration, and admin control over product details.

2.2.1 System Architecture

The system architecture consists of three main components: the user interface, the database of products, and the admin control panel. Users can view products, register, log in, and make purchases. Admins, on the other hand, have the ability to add or remove products, view user data, and apply discounts to products.

2.3 Implementation

This section outlines the implementation process, including the coding structure, algorithms, and libraries used in the project.

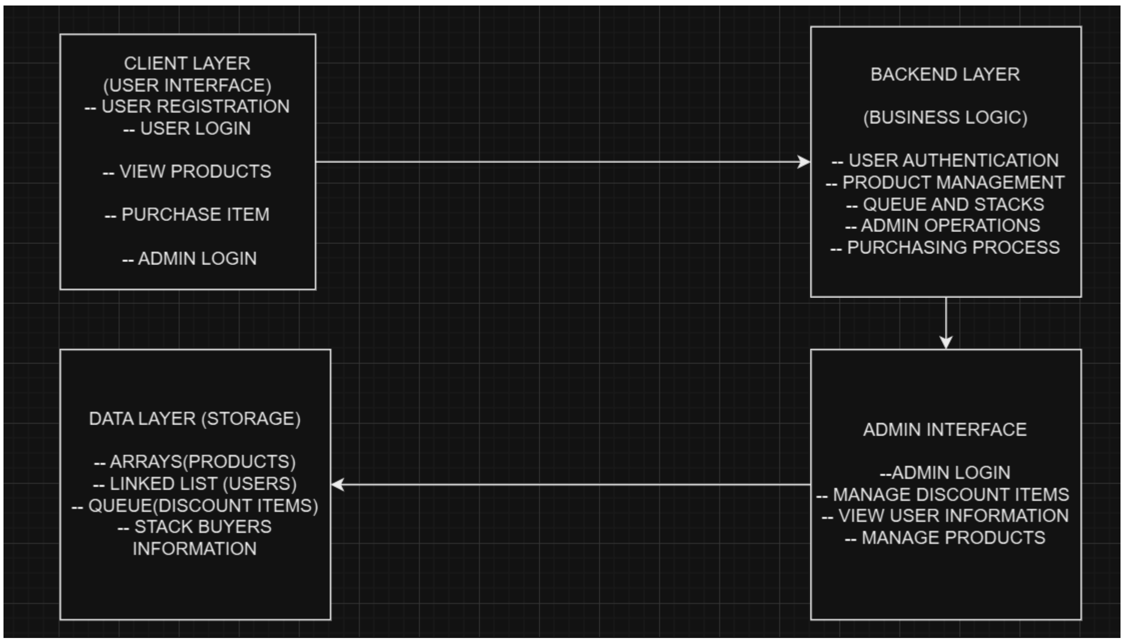


Figure 2.1: System architecture overview

2.3.1 Development Environment

The project was developed using the C programming language, utilizing various standard libraries such as ‘stdio.h‘ for input/output, ‘stdlib.h‘ for memory management, ‘string.h‘ for string manipulation, and ‘stdbool.h‘ for handling boolean values.

2.3.2 Key Components

The following are the core components of the system:

Admin Login

The AdminLogin function is implemented to authenticate admins based on predefined usernames and passwords. Upon successful login, the admin can access the system’s management panel.

Product Management

The product management system allows the admin to add, update, and remove products from the database. Each product is identified by a unique ID and contains information such as name, description, price, and quantity.

User Registration and Login

Users can register and log into the system to view products and make purchases. The system uses basic form validation to ensure that the user data is correct.

Discount and Queue Management

The discount system allows admins to apply discounts to certain products. The queue management system ensures that users receive discounts based on certain criteria, such as their purchase history or membership status.

Product Purchase

Users can purchase products by adding them to their cart and proceeding to checkout. If a product has a discount, it will be applied automatically during the checkout process.

2.4 Algorithms

This section describes the core algorithms used in the implementation of the project, which includes authentication, product search, and purchase management.

2.4.1 Admin Login Algorithm

The Admin Login algorithm is a fundamental part of the system's security, ensuring that only authorized personnel can access the administration interface. The steps of the algorithm are as follows:

1. Prompt the admin to enter their username and password.
2. Compare the entered username and password with the stored admin credentials.
3. If both the username and password match, grant access to the admin panel.
4. If the credentials do not match, display an error message and deny access.

Algorithm 1: Admin Login Authentication Algorithm

```
Input: Username, Password
Output: Access Granted or Denied
Data: Admin credentials stored in the system
if Entered username and password match stored credentials then
    Display "Access Granted"
    Grant access to Admin Panel
else
    Display "Invalid credentials"
    Deny access
```

This algorithm ensures that only the admin with the correct credentials can access the management panel.

2.4.2 Product Lookup by ID (Binary Search Algorithm)

To improve search efficiency, the system uses the Binary Search algorithm to find a product based on its unique product ID. The binary search algorithm works efficiently on sorted data and has a time complexity of $O(\log n)$.

1. Sort the list of products by their product ID.
2. Initialize two pointers, left and right, to point to the first and last elements of the list, respectively.
3. Calculate the middle index by taking the average of the left and right pointers.
4. Compare the product ID at the middle index with the target ID.
5. If the middle element matches the target ID, return the product details.
6. If the target ID is less than the middle element, move the right pointer to search the left half of the list.
7. If the target ID is greater than the middle element, move the left pointer to search the right half of the list.
8. Repeat the process until the target ID is found or the pointers converge.

Algorithm 2: Binary Search for Product Lookup by ID

```
Input: Product ID, Sorted List of Products
Output: Product Details or "Invalid ID"
Data: Sorted array of products by ID
1  left = 0, right = length of products - 1
2  while left <= right do
3      mid = (left + right) / 2
4      if productList[mid].id == productId then
5          Return productList[mid]
6      else if productId < productList[mid].id then
7          right = mid - 1
8      else
9          left = mid + 1
10 Display "Invalid Product ID"
```

The binary search algorithm is used to find a product based on the product ID, reducing the time complexity compared to a linear search.

2.4.3 Product Purchase and Discount Application Algorithm

The product purchase process ensures that users can efficiently buy products while taking into account any applicable discounts. This process also verifies the product availability and updates the stock.

1. The user selects products and adds them to their shopping cart.
2. The system checks the availability of the selected products in the database.
3. If the product is available, the system calculates any discounts applicable to the product.
4. The system applies the discount to the total price if applicable.
5. The user proceeds to checkout, where the final price is displayed along with the discount.
6. The system updates the stock quantity after the purchase.
7. If the product is not available, display a message indicating the product is out of stock.

Algorithm 3: Product Purchase and Discount Application

```
Input: Shopping Cart, Product Availability, Discount Policy
Output: Updated Product Stock, Final Price After Discount
Data: Product list, User selection, Discount criteria
for each product in Shopping Cart do
    if product is out of stock then
        Display "Product out of stock"
        Continue to next product
    if discount applicable then
        Apply discount to product price
    Add product price to total
Display "Final Price: " + total
Update stock quantity
```

This algorithm ensures that the user can successfully make a purchase while applying any relevant discounts, and updates the stock after each transaction.

2.4.4 Queue Management Algorithm (Discount Eligibility)

The queue management algorithm ensures that users who meet specific criteria can receive discounts. The system checks if the user is eligible based on their membership status or purchase history.

1. The system checks the user's profile for any discount eligibility based on membership or purchase history.

2. If the user meets the criteria, apply a discount to the purchase.

3. If the user does not meet the criteria, no discount is applied.

Algorithm 4: Queue Management for Discount Eligibility

```
Input: User Profile, Purchase History, Discount Criteria
Output: Final Discounted Price or Original Price
Data: Discount rules
1 if user is eligible for discount then
2     Apply discount to total price
3 else
4     No discount applied
5 Display final price
```

This queue management algorithm ensures that discounts are applied to eligible users, based on predefined criteria.

2.5 Conclusion

The algorithms discussed above form the backbone of the project, allowing efficient management of user authentication, product searching, purchase processing, and discount handling. By utilizing efficient algorithms such as binary search and implementing robust queue management, the system can handle large volumes of data while ensuring smooth and reliable user interactions.

Chapter 3

Performance Evaluation

3.1 Simulation Environment/ Simulation Procedure

For this project, the simulation environment consists of a C programming environment where the code is executed and tested. The system requires the following installations and environments:

- **C Compiler**: The program was compiled using GCC (GNU Compiler Collection) on Linux. It can also be compiled on Windows using IDEs like Code::Blocks, or on macOS using Xcode with the clang compiler.
- **Operating System**: The simulation was performed on a Linux-based operating system, specifically Ubuntu 20.04, to ensure the compatibility with system libraries and easy debugging of the C code.
- **Libraries**: The program makes use of standard C libraries including `stdio.h`, `stdlib.h`, `string.h`, and `stdbool.h`. No additional libraries were required beyond the system's default installation.
- **IDE**: Visual Studio Code (VS Code) was used as the primary Integrated Development Environment (IDE) for writing and testing the code. It supports C/C++ and offers features such as syntax highlighting and debugging tools.

The simulation procedure involves running the program step by step, entering test data for user login, product management, and transaction processes. Each component was tested individually and then in an integrated environment to ensure proper functionality of the system's overall flow.

3.1.1 Admin Functions

Admin functions were tested by logging in using the predefined admin credentials ("mehadi hasan" and "ariful"). After successful login, various operations such as adding or removing discount products, viewing user information, and managing buyers' details through stack operations were evaluated.

3.1.2 User Functions

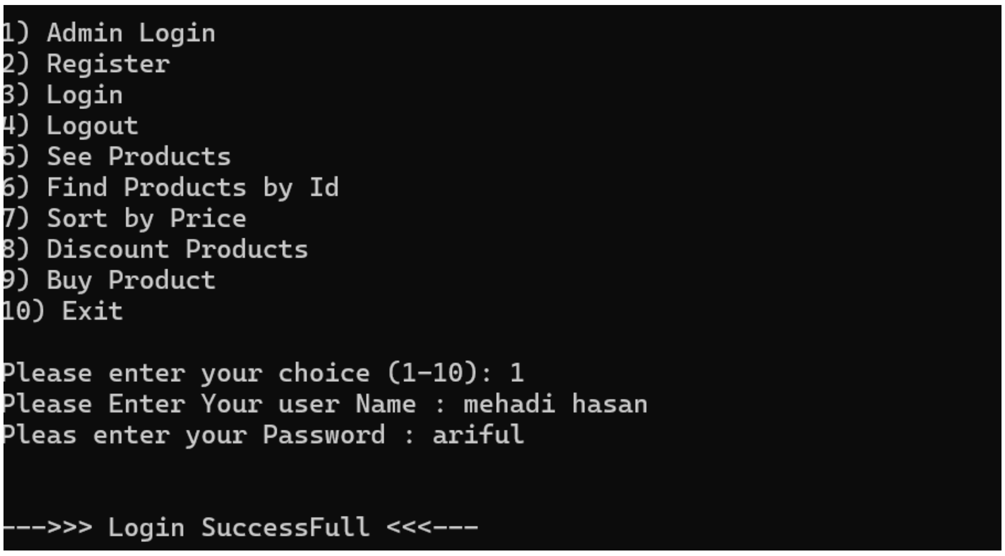
User-related functions like registration, login, product browsing, sorting by price, and making purchases (both regular and discount) were tested. Data for regular products, as well as products with discounts, were handled correctly, and users were able to perform all expected actions.

3.2 Results Analysis/Testing

In this section, the results of different functionalities of the system are presented. The goal is to verify whether each part of the system performs as expected.

3.2.1 Admin Login Test

The Admin login was tested with valid and invalid credentials. The admin interface should allow successful login only with the correct username ("mehadi hasan") and password ("ariful").



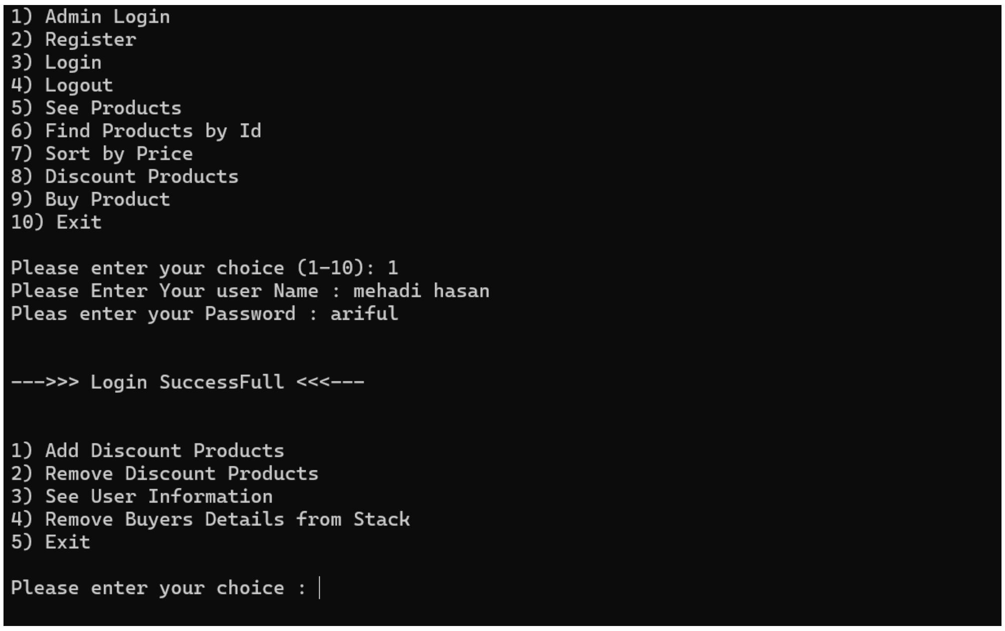
```
1) Admin Login
2) Register
3) Login
4) Logout
5) See Products
6) Find Products by Id
7) Sort by Price
8) Discount Products
9) Buy Product
10) Exit

Please enter your choice (1-10): 1
Please Enter Your user Name : mehadi hasan
Pleas enter your Password : ariful


--->>> Login SuccessFull <<<---
```

Figure 3.1: Admin Login Success Message

If the login credentials are incorrect, an error message is displayed. This functionality was verified by entering incorrect login information and ensuring that the program correctly identifies invalid input.

3.2.2 Product Management Test

The ability to add, remove, and view discounted products was tested using the Admin interface. When a new product is added, it is correctly placed into the product queue. Similarly, when a product is removed from the queue, it no longer appears in the list.

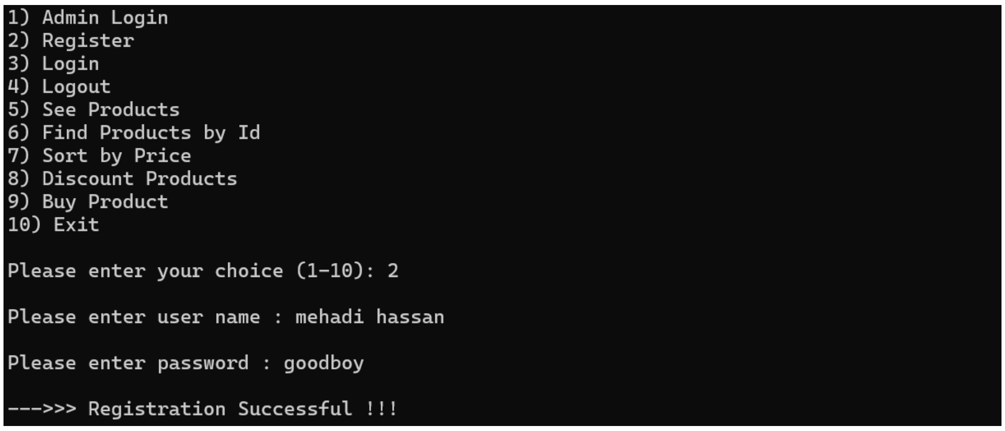


```
1) Admin Login
2) Register
3) Login
4) Logout
5) See Products
6) Find Products by Id
7) Sort by Price
8) Discount Products
9) Buy Product
10) Exit

Please enter your choice (1-10): 1
Please Enter Your user Name : mehadi hasan
Pleas enter your Password : ariful


--->>> Login SuccessFull <<<---


1) Add Discount Products
2) Remove Discount Products
3) See User Information
4) Remove Buyers Details from Stack
5) Exit

Please enter your choice : 
```

Figure 3.2: Admin Product Management Interface

3.2.3 User Registration and Login Test

User registration and login functionalities were thoroughly tested. New users were successfully registered, and login attempts with both valid and invalid credentials were evaluated. The system accurately processed the login requests based on the credentials stored in the linked list.

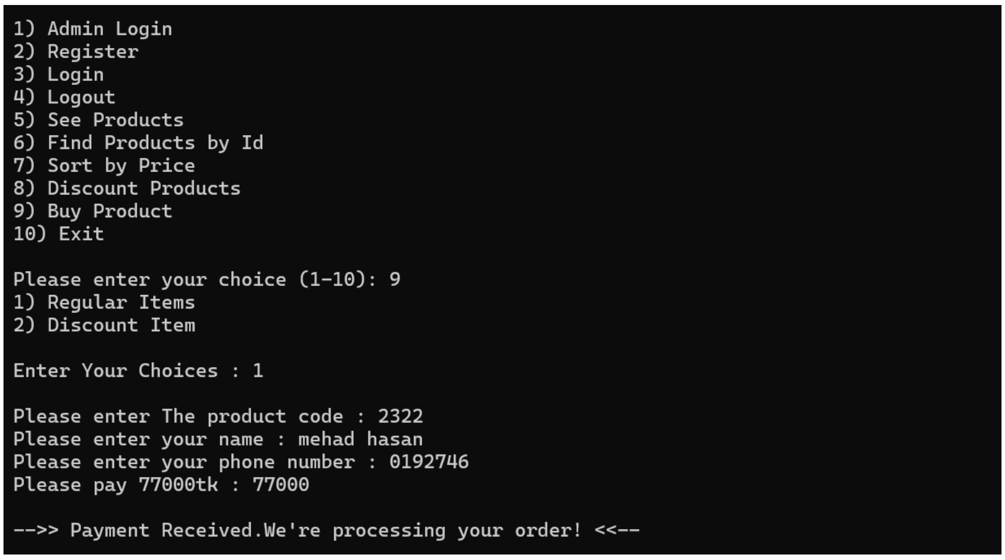


```
1) Admin Login
2) Register
3) Login
4) Logout
5) See Products
6) Find Products by Id
7) Sort by Price
8) Discount Products
9) Buy Product
10) Exit

Please enter your choice (1-10): 2

Please enter user name : mehadi hassan

Please enter password : goodboy

--->>> Registration Successful !!!
```

Figure 3.3: User Registration Interface

3.2.4 Transaction Process and Payment Handling

The system was tested by simulating transactions where users bought products after browsing through the available items. Users were able to select regular and discount items and proceed to payment.



```
1) Admin Login
2) Register
3) Login
4) Logout
5) See Products
6) Find Products by Id
7) Sort by Price
8) Discount Products
9) Buy Product
10) Exit

Please enter your choice (1-10): 9
1) Regular Items
2) Discount Item

Enter Your Choices : 1

Please enter The product code : 2322
Please enter your name : mehad hasan
Please enter your phone number : 0192746
Please pay 77000tk : 77000

-->> Payment Received.We're processing your order! <<--
```

Figure 3.4: User Payment Process

The payment was validated by comparing the entered amount with the expected price. If the payment was correct, the purchase was successfully processed, otherwise an error message was shown.

3.3 Results Overall Discussion

The system functions as expected for all the major features implemented. However, some areas could be further optimized. For instance:

- **User Input Validation**: The system relies on basic input handling. More sophisticated input validation techniques, such as checking for non-numeric characters in product codes or amounts, could enhance the system's reliability.
- **Memory Management**: The dynamic memory allocation used in the registration and stack operations needs careful monitoring to avoid memory leaks. A memory profiling tool could help identify any potential issues.
- **Queue Overflow**: The queue for managing discounted products is static with a fixed size. In a real-world scenario, this would need to be dynamic or allow for more flexibility in handling products.

3.3.1 Complex Engineering Problem Discussion

This project addresses several complex engineering problems:

- **Data Management**: Efficient management of data such as users, products, and transactions using linked lists, stacks, and queues.

- **Concurrency in User Access**: Although not implemented in this version, a future enhancement could involve handling multiple users accessing the system simultaneously, which would require synchronization mechanisms.
- **Payment Handling**: The payment system involves checking user input against expected values and providing a response accordingly. This could be integrated with real-world payment gateways for further complexity.

Chapter 4

Conclusion

4.1 Discussion

This project aimed to develop a basic e-commerce platform in C, capable of handling user registrations, product management, discount offers, and payment processing. The system successfully supports two main roles: administrators and users. Administrators can add or remove discounted products, view user information, and manage orders, while users can browse and purchase regular or discounted items. The implementation has been tested and demonstrates functionality such as product display, user registration, login, and transaction handling. The system offers a simplified yet effective solution for managing an online store with essential features. Despite its simplicity, it offers an excellent foundation for a more complex system. A major takeaway from the project is that even basic systems, when built carefully, can create significant user interaction flows and efficiently manage various tasks. Overall, the project achieved its primary goals, confirming the validity of the design and implementation while laying the groundwork for future enhancements.

4.2 Limitations

Although the project functions as expected, it does come with certain limitations. One of the key limitations is the use of static data structures, such as the fixed-size queue and arrays, to manage products and user data. This restricts the system's ability to scale efficiently, especially when dealing with large numbers of products and users. Additionally, the system lacks dynamic memory management for handling extensive datasets, which would be essential for real-world applications. The error handling mechanisms are also minimal, with insufficient checks for invalid user inputs, such as product IDs and payment amounts. Furthermore, the system is unable to handle multiple users concurrently, as it does not have multi-threading or session management. Another limitation is the lack of robust security features. For example, sensitive user data, such as passwords, are stored in plain text, which is a security vulnerability in a real-world setting. The system also lacks data persistence, meaning all the information is lost once the program is closed. These limitations indicate areas that need to be addressed before the system can be used in a production environment.

4.3 Scope of Future Work

Future work on this project offers significant opportunities for improvement. One of the most important enhancements would be to replace the static data structures with dynamic alternatives, such as linked lists or dynamic arrays. This would enable the system to handle a larger number of users and products, making it more scalable. Additionally, implementing proper error handling mechanisms and validating user inputs would increase the reliability of the system. For example, checks could be added to ensure that user passwords meet security requirements and that product codes are valid before transactions are processed. To address concurrency limitations, multi-threading could be implemented to allow multiple users to interact with the system simultaneously. In terms of security, the system could be improved by adding encryption for sensitive data, such as passwords, and implementing a secure payment gateway for handling financial transactions. Further, integrating a database to store product and user information would allow for data persistence, so the system could retain information between sessions. Another potential area for improvement is integrating real-world APIs to simulate online payment systems or product stock management. By adding these features, the system could evolve into a more robust, secure, and scalable e-commerce platform, capable of handling larger datasets and providing a smoother user experience. Finally, an intuitive user interface could be introduced to make the platform more user-friendly, possibly transitioning the system to a graphical interface with web-based support.

[1] [2] [3]

References

[1] JSON Syntax. // w3schools. com. *URL: https://www. w3schools. com/js/js_json_syntax. asp*, 2012.

[2] Seyed Mehdi Nasehi, Jonathan Sillito, Frank Maurer, and Chris Burns. What makes a good code example?: A study of programming q&a in stackoverflow. In *2012 28th IEEE International Conference on Software Maintenance (ICSM)*, pages 25–34. IEEE, 2012.

[3] Hubert L Dreyfus. *On the internet*. Routledge, 2008.