K.S.RANGASAMY COLLEGE OF TECHNOLOGY

(Autonomous)

TIRUCHENGODE-637215

A MINI PROJECT REPORT

DEPARTMENT STORE MANAGEMENT SYSTEM

60 IT L04 – C# and .NET FRAMEWORK

BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

Submitted by

NANDHAGOPALAN S (73772214166)

ROHITH S (73772214187)

K.S.RANGASAMY COLLEGE OF TECHNOLOGY

(Autonomous)
TIRUCHENGODE-637215

BONAFIDE CERTIFICATE

Certified that this project report titled "**DEPARTMENT STORE MANAGEMENT SYSTEM**" is the bonafide work of **NANDHAGOPALAN S (73772214166), ROHITH S(73772214187)** who carried out the project under my guidance.

ABSTRACT

The **Department Store Management System** is a comprehensive web-based platform designed to streamline and automate the operations of retail businesses. Built using C# with ASP.NET and powered by a reliable relational database, the system ensures efficient data handling, smooth performance, and seamless user interaction. It is tailored to meet the diverse needs of department stores, providing solutions for inventory management, sales tracking, and customer engagement. The system's modular design allows for easy scalability, making it adaptable for small stores, large retail chains, and franchise operations.

One of the key features of the Department Store Management System is its ability to handle inventory control and product categorization. Store managers can effortlessly add, update, and organize products into different categories, enabling efficient stock management. The system supports barcode scanning, automatic stock updates, and low-stock alerts, ensuring that the store maintains optimal inventory levels. This feature reduces the risk of stockouts and overstocking, improving operational efficiency and customer satisfaction.

The system also provides robust sales tracking and reporting tools. Cashiers can process transactions quickly through a user-friendly point-of-sale (POS) interface that supports multiple payment methods, including cash, credit/debit cards, and digital wallets. The application instantly records sales data, generating detailed reports on sales trends, revenue, and customer preferences. These insights help store managers make data-driven decisions, optimize product offerings, and develop targeted marketing strategies.

In addition to inventory and sales management, the Department Store Management System focuses on enhancing the customer experience. The system supports customer loyalty programs, allowing users to register for reward points and receive personalized offers. Real-time notifications and promotional updates help engage customers, fostering brand loyalty and repeat business.

The platform's intuitive interface ensures that both employees and customers have a seamless experience. Its secure, role-based access control system protects sensitive data and ensures that users only access the features they are authorized to use. With comprehensive performance reports, error-free transaction processing, and customizable modules, the Department Store Management System is a vital tool for modern retail management, promoting efficiency, profitability, and customer satisfaction.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

The **Department Store Management System**, developed using C# and the .NET framework, is a comprehensive solution designed to streamline the operations of department stores by managing sales, inventory, customer relationships, and employee activities in an integrated manner. This system empowers store managers and staff with robust tools for tracking product availability, processing transactions, and maintaining accurate inventory records, while customers benefit from an enhanced shopping experience through efficient service and personalized engagement.

The platform's scalable architecture makes it suitable for both small retail outlets and large department stores. It features modules for point-of-sale (POS), inventory control, supplier management, and customer loyalty programs, ensuring that every aspect of store management is covered. Security is a priority, with measures in place to safeguard financial transactions and customer data. By leveraging the capabilities of the .NET framework, the system ensures a seamless and responsive experience across desktop and mobile devices, enabling store personnel to access critical information anytime, anywhere.

This solution not only simplifies the daily operations of department stores but also enhances decision-making through real-time analytics and reporting. Store managers can monitor sales trends, optimize stock levels, and improve customer satisfaction through data-driven insights. The system's intuitive interface and automation of routine tasks help reduce human error and boost overall efficiency.

Whether used in single-location stores or across multi-store chains, the **Department Store Management System** empowers businesses to deliver exceptional service, maintain inventory accuracy, and drive profitability. By integrating sales, inventory, and customer management into a unified platform, it transforms traditional retail operations into a streamlined, data-driven experience that meets the demands of today's competitive retail environment.

CHAPTER 2

REQUIREMENT ANALYSIS

FUNCTIONAL REQUIREMENTS

Product Management:

- Add, edit, and delete products, including details like name, category, price, and stock quantity.
- Support for categorizing products by departments, brands, and seasonal collections.
- Manage product descriptions, images, and specifications to enhance the shopping experience.
- Track stock levels and set alerts for low inventory.

Inventory Management:

- Real-time tracking of stock across multiple locations or warehouses.
- Automated stock reordering based on predefined thresholds.
- Generate reports on stock levels, product movement, and turnover rates.
- Support for barcode scanning and batch tracking for efficient stock management.

Sales and Point-of-Sale (POS) Management:

- Process sales transactions, including discounts, promotions, and loyalty points.
- Support multiple payment methods, including cash, credit/debit cards, and digital wallets.
- Print or email receipts and invoices to customers.
- Generate daily, weekly, and monthly sales reports.

Customer Management:

- Create and manage customer profiles, including purchase history and contact details.
- Implement a loyalty program with reward points and special offers for frequent customers.
- Send personalized promotions and notifications based on customer preferences.
- Provide a feedback mechanism for customers to rate their shopping experience.

Supplier Management:

- Maintain a database of suppliers and their contact information.
- Track purchase orders, delivery schedules, and payment status.
- Generate reports on supplier performance and procurement costs.

Employee Management:

- Manage employee roles and access permissions based on job functions.
- Track employee work schedules, attendance, and performance metric

NON-FUNCTIONAL REQUIREMENTS

User Interface (UI) and User Experience (UX):

✓ Design a responsive and user-friendly interface that adapts to various devices, including desktops, tablets, and smartphones.

✓ Ensure consistent design elements and intuitive navigation to enhance the shopping experience and reduce the learning curve for users.

✓ Provide clear, easy-to-use dashboards for managers and employees to access critical information quickly.

Performance and Scalability:

✓ Support high transaction volumes and concurrent users, ensuring smooth operations during peak shopping times.

✓ Optimize system performance to minimize load times for product searches, checkout processes, and reporting.

✓ Design for scalability to accommodate future expansion, including additional stores and product lines.

Data Management and Security:

✓ Implement secure database management for storing product, customer, and transaction data.

✓ Use encryption and access controls to protect sensitive information, including payment details and personal customer data.

✓ Conduct regular security audits and vulnerability assessments to safeguard against threats.

System Reliability and Availability:

✓ Ensure high system availability with minimal downtime to provide uninterrupted service.

✓ Implement failover mechanisms and automated backups to protect data and recover quickly in case of system failures.

✓ Design for redundancy and load balancing to enhance system reliability.

Compliance and Accessibility:

✓ Adhere to data protection regulations such as GDPR or CCPA to ensure customer privacy and compliance.

✓ Follow accessibility standards, including WCAG guidelines, to ensure that the system is usable by individuals with disabilities.

✓ Maintain audit logs and reporting capabilities to support compliance and accountability.

CHAPTER 3

SYSTEM DESIGN

The design of the **Department Store Management System** ensures a clear structure for managing products, sales, inventory, employees, and customers, supported by a well-defined flow and class relationships. Below is a detailed explanation of the system's operational flow:

QUIZ FLOW

Login and User Authentication:

- Users log in based on roles (manager, cashier, staff) with appropriate access permissions.

Product Management:

- Add, edit, or delete products, including details like name, category, price, and stock.

Inventory Management:

- Monitor stock levels and receive low-stock alerts.
- Create purchase orders and update stock upon delivery.

Sales (POS) Flow:

- Scan or select products for purchase.
- Apply discounts and process payments (cash, card, digital).
- Generate and issue receipts.

Customer Management:

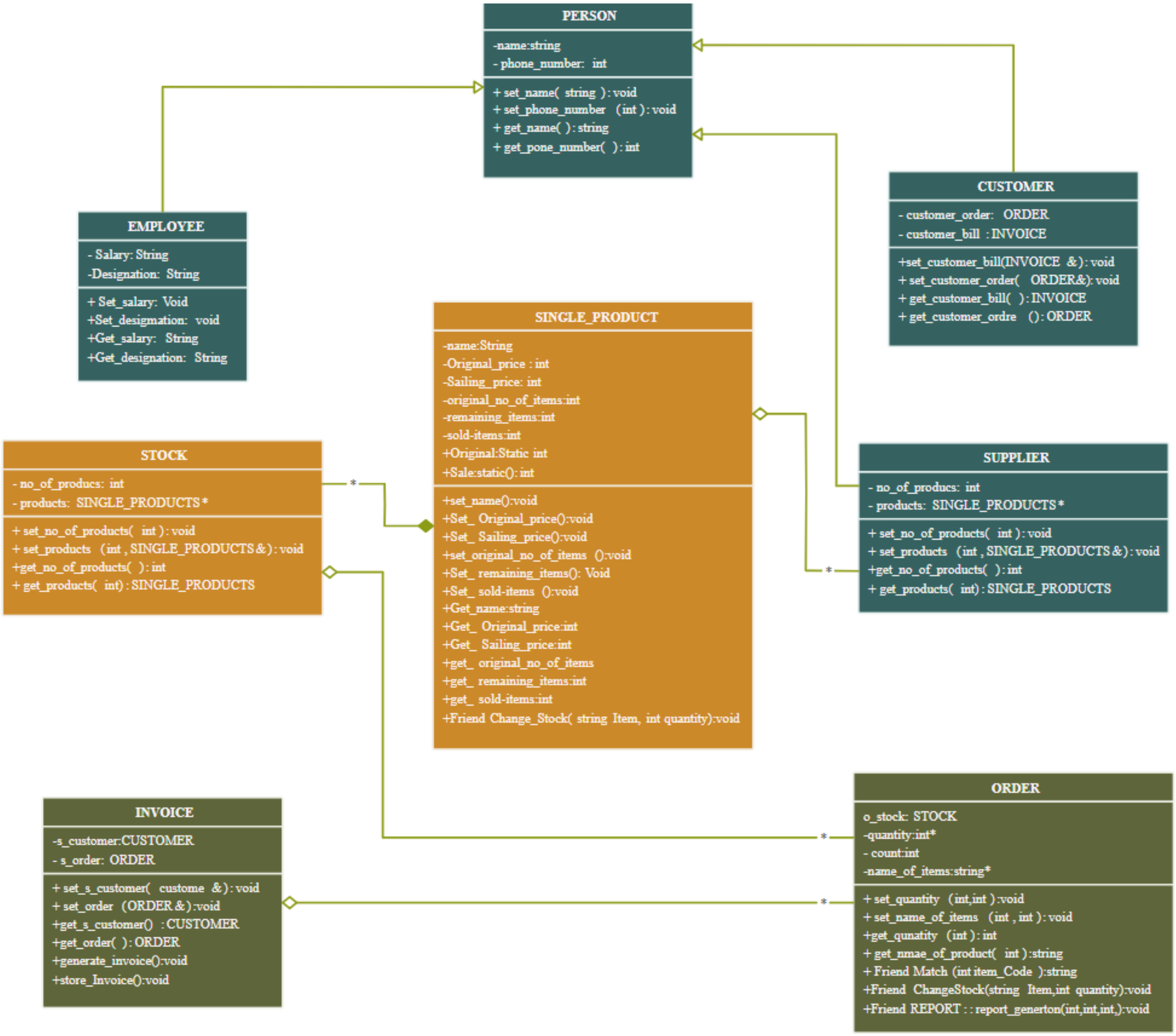
- Create/update customer profiles.
- Manage loyalty points and send personalized promotions.

Employee Management:

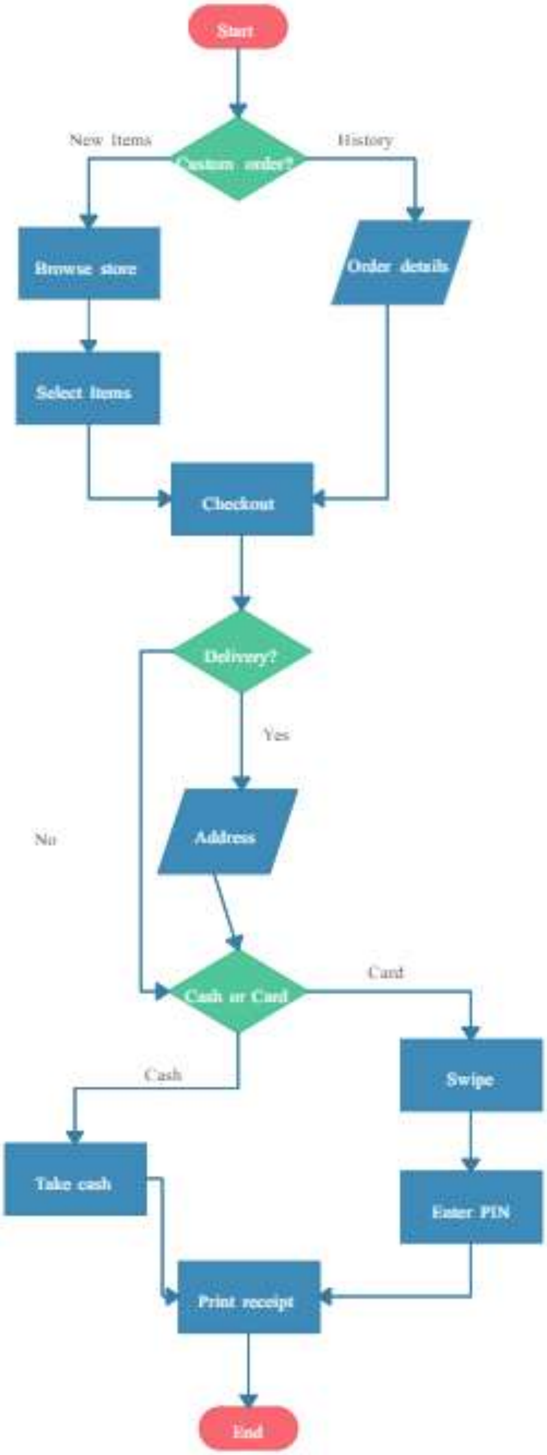
- Add/edit employee details and assign roles.
- Track attendance and performance.

UML DIAGRAM

CLASS DIAGRAM


PERSON
-name:string
- phone_number: int
+ set_name(string): void
+ set_phone_number (int): void
+ get_name( ): string
+ get_pone_number( ): int
CUSTOMER
- customer_order: ORDER
- customer_bill : INVOICE
+set_customer_bill(INVOICE &): void
+ set_customer_order(ORDER&): void
+ get_customer_bill( ): INVOICE
+ get_customer_ordre (): ORDER
EMPLOYEE
- Salary: String
-Designation: String
+ Set_salary: Void
+Set_desigmation: void
+Get_salary: String
+Get_designation: String
SINGLE_PRODUCT
-name:String
-Original_price : int
-Sailing_price: int
-original_no_of_items:int
-remaining_items:int
-sold-items:int
+Original:Static int
+Sale:static(): int
+set_name():void
+Set_ Original_price():void
+Set_ Sailing_price():void
+set_original_no_of_items ():void
+Set_ remaining_items(): Void
+Set_ sold-items ():void
+Get_name:string
+Get_ Original_price:int
+Get_ Sailing_price:int
+get_ original_no_of_items
+get_ remaining_items:int
+get_ sold-items:int
+Friend Change_Stock(string Item, int quantity):void
STOCK
- no_of_producs: int
- products: SINGLE_PRODUCTS *
+ set_no_of_products(int): void
+ set_products (int , SINGLE_PRODUCTS &): void
+get_no_of_products( ): int
+ get_products(int) : SINGLE_PRODUCTS
*
SUPPLIER
- no_of_producs: int
- products: SINGLE_PRODUCTS *
+ set_no_of_products(int): void
+ set_products (int , SINGLE_PRODUCTS &): void
+get_no_of_products( ): int
+ get_products(int) : SINGLE_PRODUCTS
*
INVOICE
-s_customer:CUSTOMER
- s_order: ORDER
+ set_s_customer(custome &): void
+ set_order (ORDER &):void
+get_s_customer() : CUSTOMER
+get_order( ): ORDER
+generate_invoice():void
+store_Invoice():void
ORDER
o_stock: STOCK
-quantity:int*
- count:int
-name_of_items:string*
+ set_quantity (int,int):void
+ set_name_of_items (int , int): void
+get_qunatity (int): int
+ get_nmae_of_product(int):string
+ Friend Match (int item_Code):string
+Friend ChangeStock(string Item,int quantity):void
+Friend REPORT : : report_generton(int,int,int,):void
*
*

FLOW DIAGRAM



CHAPTER 4
IMPLEMENTATION

CODE

USER CLASS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace DepartStoreManagementSystem.Class
{
    class User
    {
        public int UserID { get; set; }
        public string FullName { get; set; }
        public string Username { get; set; }
        public string Email { get; set; }
        public string Contact { get; set; }
        public string Password { get; set; }
        public string UserType { get; set; }
    }
}
```

INVENTORY CLASS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace DepartStoreManagementSystem.Class
{
    class Inventory
    {
        public int ProductID { get; set; }
        public decimal Quantity { get; set; }
    }
}
```

PRODUCT CLASS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace DepartStoreManagementSystem.Class
{
    class Product
    {
        public int ProductID { get; set; }
        public string ProductName { get; set; }
        public decimal Rate { get; set; }
        public decimal Quantity { get; set; }
        public string Category { get; set; }

    }
}
```

TRANSACTION CLASS

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace DepartStoreManagementSystem.Class
{
   class TransactionClass
   {
      public int TransactionID { get; set; }
      public string TransactionType { get; set; }
      public string SysUser { get; set; }
      public DateTime TransactioDate { get; set; }
      public decimal GrandTotal { get; set; }
      public DataTable TransactionDetails { get; set; }
   }
   public class TransactionDetails
   {
      public int TransactionID { get; set; }
      public int ProductID { get; set; }
      public decimal Tax { get; set; }
      public decimal Discount { get; set; }
      public decimal Rate { get; set; }
      public decimal Quantity { get; set; }
      public decimal Total { get; set; }
   }
}
```

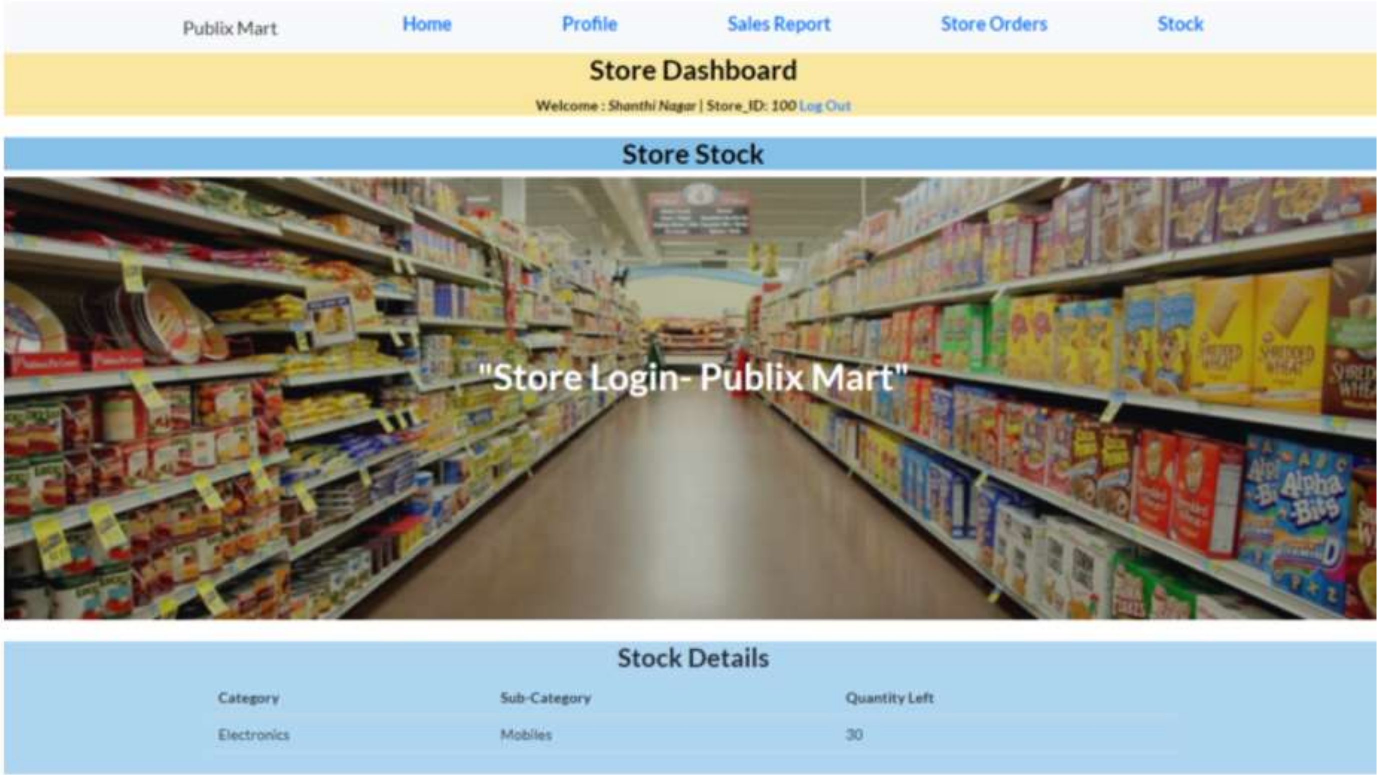
PROGRAM CLASS

```
using DepartStoreManagementSystem.UI;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace DepartStoreManagementSystem
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new frmLogin());
        }
    }
}
```

CHAPTER 5

OUTPUT



Store Dashboard

Welcome : *Shanthi Nagar* | Store_ID: *100* Log Out

Store Stock

"Store Login- Publix Mart"

Stock Details

Category	Sub-Category	Quantity Left
Electronics	Mobiles	30

CHAPTER 6
CONCLUSION

CONCLUSION

The Department Store Management System provides a practical, scalable, and efficient solution for managing the core operations of a department store, including product handling, inventory control, sales processing, and customer engagement. By streamlining processes such as stock management, employee oversight, and supplier coordination, this system ensures smooth day-to-day operations and enhances decision-making through comprehensive reporting and analytics.

Its modular architecture allows for easy maintenance and future scalability, accommodating the growing needs of the business. The system's user-friendly interface, role-based access control, and integration with essential hardware like barcode scanners ensure a seamless experience for staff and customers alike.

With a strong focus on security, performance, and compliance, the system protects sensitive data and supports adherence to industry standards. Future enhancements, such as integrating AI-driven analytics, personalized marketing, and mobile support, can further extend its functionality, catering to modern retail demands.

Overall, this project highlights how technology can optimize retail operations, improve customer service, and drive business growth. Its adaptability and potential for continuous improvement make it a valuable tool for the evolving retail landscape.

CHAPTER 7
REFERENCES

[1] M. Fowler. *Patterns of Enterprise Application Architecture.* Addison-Wesley, 2002. ISBN: 978-0321127426.

[2] S. McConnell. *Code Complete: A Practical Handbook of Software Construction.* Microsoft Press, 2004. ISBN: 978-0735619678.

[3] E. Gamma, R. Helm, R. Johnson, J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software.* Addison-Wesley, 1994. ISBN: 978-0201633610.

[4] Microsoft. "Introduction to ASP.NET Core and Entity Framework." Microsoft Learn, 2023. https://learn.microsoft.com/en-us/aspnet/core/

[5] K. Scott, J. Taylor. *Pro ASP.NET Core 6: Develop Cloud-Ready Web Applications Using MVC, Blazor, and Razor Pages.* Apress, 2022. https://doi.org/10.1007/978-1-4842-7885-1.

[6] D. Soni, K. Ng. *Mastering C# and .NET Framework.* Packt Publishing, 2021. ISBN: 978-1800568105.

/