# **Project Report: Project0-ShoppingCart**

# 

**Submitted By:** GROUP NO 12

Group Members:

Zain ul abideen Ali 233604 BSCS B

Arfa Imran 233596 BSCS B

Fatima Khalid 233526 BSCS B

**Course Title:** VISUAL PROGRAMMING

**Semester:** 3RD

**Session:** 2023-2027

**Department:** COMPUTER SCIENCE

**Submitted To:** MAM AATKA ALI

# **Table of Content**

Contents

[**Project Report: Project0-ShoppingCart** 1](#_Toc180752341)

[1](#_Toc180752342)

[**Table of Content** 2](#_Toc180752343)

[**Introduction** 3](#_Toc180752344)

[**Requirements** 3](#_Toc180752345)

[**Software** 3](#_Toc180752346)

[**Design and Architecture** 4](#_Toc180752347)

[ **UML DESIGN** 4](#_Toc180752348)

[ **Class Design** 4](#_Toc180752349)

[**Product Class** 4](#_Toc180752350)

[**Cart Class** 4](#_Toc180752351)

[**Main Class (User):** 5](#_Toc180752352)

[**Implementation** 5](#_Toc180752353)

[**Key Features** 5](#_Toc180752354)

[**Add Product:** 5](#_Toc180752355)

[**View Cart:** 5](#_Toc180752356)

[**Remove Product** 6](#_Toc180752357)

[**Cart Expiration** 6](#_Toc180752358)

[**Input Validation** 6](#_Toc180752359)

[**Code Explanation** 6](#_Toc180752360)

[**Testing** 6](#_Toc180752361)

[**Testing Strategies** 6](#_Toc180752362)

[**Test Cases** 6](#_Toc180752363)

[**Challenges and Solutions** 6](#_Toc180752364)

[**Challenges** 7](#_Toc180752365)

[**Solutions** 7](#_Toc180752366)

**Abstract**

This report details the development of a console-base Shopping Cart application implemented in c#. It focuses on managing products, handling user input validation, and performing basic cart operation

# **Introduction**

This project demonstrates a simple **Shopping Cart Application** using C# in a console-based environment. It allows users to add, remove, and view products in a shopping cart. Additional functionalities like product recommendations, cart expiration due to inactivity, subtotal calculation, discounts, and sales tax computation are included.

# **Requirements**

## **Software**

* C# Compiler (Visual Studio)

# **Design and Architecture**

## **UML DESIGN**

## **Class Design**

**Product Class**: Handles product details with validation for price and quantity.

**Cart Class:** Contains a list of Product objects.

Provides methods to:

* Add products to the cart.
* Remove products.
* View cart contents.
* Handle **expiration** using asynchronous tasks.

## **Main Class (User):**

* Provides a **menu-driven interface** to interact with the user.
* Handles user choices and redirects to relevant cart operations.

**Objectives:**

* To build a **user-friendly shopping cart system**.
* To add/remove products in cart.
* To practice **Object-Oriented Programming (OOP)** concepts like encapsulation and properties.
* To demonstrate **input validation** and error handling.
* To apply **task-based asynchronous operations** for cart expiration.
* To simulate **product recommendations** and real-time **checkout processing.**
* To view Cart and add Discounts, sales tax and calculate total cost.

# **Implementation**

## **Key Features**

### **Add Product:**

* The user can add predefined products (e.g., Pen, Watch) or custom products.
* The system checks if a product with the same ID already exists and updates its quantity accordingly.

### **View Cart:**

Displays all products in the cart with details like name, price, quantity, and total amount.

* + Subtotal, discount (10%), and sales tax (8%) are calculated.
  + Users can **checkout** or go back to the main menu.

**Remove Product**: Allows users to remove items from the cart by entering product IDs.

**Cart Expiration:** The cart **clears after 3 minutes** of inactivity to simulate session expiration.

**Input Validation**: Ensures valid entries for product price and quantity. Our console application only accepts greater than 0 entries for product and price otherwise our app will ask user to enter again valid value.

## **Code Explanation**

* **Properties**: Used for ensuring valid price and quantity in the Product class.
* **Cart Operations**: Methods for adding, viewing, and removing products handle user interactions and calculations.
* **Async Task**: Cart Expiration method uses asynchronous delay to clear the cart after a set period (3minutes) of inactivity.

# **Testing**

## **Testing Strategies**

* Manual testing for input validation and cart functionalities.

## **Test Cases**

* Ensure valid input handling for price and quantity.
* Verify correct cart operations for adding, viewing, and removing products.

# **Challenges and Solutions**

## **Challenges**

* Handling incorrect user inputs gracefully.

## **Solutions**

* Implemented robust input validation with prompts for re-entry.