E-Commerce Cart System

Project report submitted in partial fulfillment of the Requirements for the Award of the Degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

< Student's Names with Reg. Nos>

Gosala Sarath Babu	24KB1A05H9
Kalavalapudi Rohith	24KB1A05N4
Kandi Rishendra	24KB1A05P2
Kandrathi pradeep Kumar	24KB1A05P3
Chembeti Jaswanth Yadav	24KB1A0595

Under the Guidance of

P.Suneetha Mam

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
N.B.K.R. INSTITUTE OF SCIENCE & TECHNOLOGY
(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project report entitled YOUR PROJECT TITLE being submitted by

Gosala Sarath Babu	**24KB1A05H9**
Kalavalapudi Rohith	**24KB1A05N4**
Kandi Rishendra	**24KB1A05P2**
Kandrathi pradeep Kumar	**24KB1A05P3**
Chembeti Jaswanth Yadav	**24KB1A0595**

in partial fulfillment for the award of the Degree of Bachelor of Technology in Computer Science and Engineering to the **Jawaharlal Nehru Technological University, Anantapur** is a record of bonafied work carried out under my guidance and supervision.

P.Suneetha Mam

Assistant Professor

DR.Ravindra reddy

M.Tech, Ph.D

Head of the Department

DECLARATION

I hereby declare that the dissertation entitled **E-Commerce Cart System** submitted for the B.Tech Degree is my original work and the dissertation has not formed the basis for the award of any degree, associateship, fellowship or any other similar titles.

Place: Vidyanagar

Date:07/05/2025 K.Rohith
24B1A05N4

Acknowledgement :

Certainly! Here's an **extended and more detailed acknowledgement** with a formal tone suitable for a report or academic submission:

Acknowledgement

I would like to take this opportunity to express my sincere and heartfelt gratitude to **[P.Suneetha mam/Mr.Sivanraj sir**], for their invaluable guidance, support, and encouragement throughout the development of this project titled **"E-Commerce Cart System in C."**

Professor [Last Name]'s insightful lectures, deep subject knowledge, and consistent motivation have played a pivotal role in helping me understand the core concepts of **structured programming**, **data handling**, and **modular design in C**. His/her timely feedback, thoughtful suggestions, and patient mentoring have been instrumental in shaping this project and ensuring its successful completion.

This project has provided me with practical exposure to essential programming elements such as **arrays, structures, linked lists, dynamic memory allocation**, and user-defined functions. It also enhanced my understanding of applying real-world logic in software, such as managing a shopping cart, billing system, discount policies, and user input validation.

I am also thankful to my classmates, friends, and family who provided constant support and encouragement during this endeavor. Their discussions and suggestions contributed significantly to refining the functionality of the system.

Lastly, I would like to extend my gratitude to all the faculty members of the [Institution/University Name] for fostering an environment of learning, innovation, and exploration that enabled me to undertake and complete this project with confidence.

Abstract of the E-Commerce Cart System in C

This C program implements a console-based **E-Commerce Cart System** that allows users to browse products, manage a shopping cart, provide delivery address details, apply coupon codes, and generate a final bill. It simulates an online shopping experience with the following core features:

Key Components:

1. **Product Catalog**

- 25 predefined products divided into categories like **Electronics, Grocery, Fitness, Kitchen, and Fashion**.
- Each product has attributes like ID, name, price, and available stock.

2. **Shopping Cart (Linked List)**
    - A dynamic singly linked list (`CartNode`) is used to maintain items added to the cart.
    - Supports operations like adding items, increasing quantity, removing products, and clearing the cart after checkout.
3. **Delivery Address Management**
    - Users input and validate their delivery address including fields like address, town, district, state, and 6-digit PIN code.
4. **Billing System**
    - Calculates subtotal, applies optional discount (via coupon code `SAVE10` for 10% off), adds 18% GST, and a fixed ₹100 shipping charge.
    - Displays a detailed invoice including all charges and delivery information.
5. **User Interaction Menu**
    - A looped main menu lets users:
        - View available products.
        - Add/remove products to/from cart.
        - View cart contents.
        - Enter address.
        - Generate bill and checkout.

Technical Highlights:

- **Data Structures:** Arrays (for product catalog), linked lists (for cart), and structures (`struct`) for organizing data.
- **Validation:** Ensures PIN code correctness and stock availability.
- **Memory Management:** Uses dynamic memory allocation (`malloc`, `free`) for cart nodes.
- **User Input Handling:** Uses `fgets`, `flushInput` to handle newline characters and avoid input issues.

Use Case:

Ideal for academic projects demonstrating **data structures, file-less data handling, and basic business logic implementation** in C programming.

Would you like a flowchart or diagram to go with this abstract?

Introduction

In the rapidly evolving digital landscape, **e-commerce systems** play a critical role in redefining how people shop, buy, and sell products. From groceries to electronics and fashion, e-commerce platforms offer users the convenience of browsing a wide catalog, managing a virtual cart, applying discounts, and checking out from the comfort of their homes.

This project, titled **"E-Commerce Cart System in C,"** is a simplified console-based simulation of a basic e-commerce platform developed using the C programming language. The goal of this system is to demonstrate fundamental programming concepts through a practical application that mimics the behavior of real-world online shopping platforms. The system provides essential features such as viewing a product catalog, adding/removing items from the cart, managing delivery address information, applying discount coupons, and generating a detailed bill.

Project Overview

1. Objective

The main objective of this project is to:

- Implement a menu-driven shopping cart application.

- Demonstrate the use of arrays, structures, and linked lists in C.
- Simulate functionalities of a basic e-commerce platform like billing, stock management, and input validation.

System Features

1. Product Catalog

- The system includes a predefined catalog of **25 products** from categories such as:
    - **Electronics** (e.g., Smartphone, Laptop)
    - **Grocery** (e.g., Rice, Oil)
    - **Fitness Accessories** (e.g., Yoga Mat, Dumbbells)
    - **Kitchen Accessories** (e.g., Blender, Knife Set)
    - **Fashion** (e.g., T-Shirts, Jeans)
- Each product has attributes like:
    - `id` (unique identifier)
    - `name`
    - `price`
    - `stock` (available quantity)

2. Shopping Cart

- Implemented using a **singly linked list** (`CartNode` structure).
- Users can:
    - **Add products** to the cart.
    - **Increase quantity** of existing items.

    - **Remove products**.
    - **View** cart contents with subtotal.
- Product stock is adjusted in real time when items are added or removed from the cart.

3. Delivery Address Handling

- A structure is used to store and manage user address details.
- Fields include:
    - Full address
    - Town/City
    - District
    - State
    - PIN code (validated to be exactly 6 digits using `isValidPincode()`)

4. Coupon Code Support

- Supports applying a coupon code **"SAVE10"** for a 10% discount on the subtotal.
- Invalid codes are rejected with appropriate feedback.

5. Bill Generation

- When the user chooses to checkout:
    - Cart items are listed with total prices.
    - Discount is applied if a valid coupon is entered.
    - 18% **GST (Tax)** is calculated.
    - A fixed **₹100 shipping charge** is added.
    - The grand total is displayed along with delivery address.

6. Input Handling & Validation

- Uses `fgets()` to handle multi-word input like address.
- `flushInput()` function ensures input buffer is clean before reading strings after numbers.
- Validation is done for product availability and PIN code.

Conclusion

This project effectively simulates a basic e-commerce shopping experience and serves as a practical demonstration of structured programming in C. By implementing this system, we gain hands-on experience with real-world problem solving, data structures, memory management, and modular programming.

It can serve as a foundation for more advanced systems involving user accounts, payment integration, file storage, and GUI-based front ends in the future.

Existing System in E-Commerce Cart

The **existing e-commerce systems** in the market today are robust, highly scalable platforms developed using advanced programming languages and frameworks. These systems are used by global platforms such as **Amazon, Flipkart, Myntra, BigBasket**, and many others. They offer a rich user interface, seamless navigation, and a wide range of features aimed at enhancing customer convenience and operational efficiency.

Features of Existing E-Commerce Cart Systems:

1. **User Authentication and Accounts:**
    - Secure login/logout mechanisms.
    - Profile management (name, email, addresses, saved cards, etc.).
2. **Product Catalog with Filters:**
    - Categories, sub-categories, and search functionalities.
    - Sorting by price, brand, popularity, or user reviews.

3. **Real-Time Inventory Management:**
   - Tracks product availability in real-time.
   - Automatically updates stock after purchase.
4. **Dynamic Cart Management:**
   - Add/remove items with instant cart refresh.
   - Quantity update and instant total recalculation.
   - Wishlist and save-for-later options.
5. **Secure Checkout Process:**
   - Delivery address input with validation and GPS-based autofill.
   - Multiple payment options (Credit/Debit card, UPI, Wallets, Net banking).
   - Application of discount coupons, vouchers, and loyalty points.
6. **Billing and Taxation:**
   - Automatic calculation of applicable taxes (GST, VAT).
   - Invoice generation and downloadable receipts.
7. **Order Tracking and Notifications:**
   - Order status updates via SMS/email.
   - Live tracking of shipment.
8. **Return and Refund Management:**
   - Simple return process with eligibility checks.
   - Instant refund initiation for prepaid orders.
9. **Admin Panel:**
   - Product management (CRUD operations).
   - Order tracking, customer support integration, and analytics.

Limitations of Existing Systems for Beginners in Programming:

- **Complexity**: These systems are built using technologies like **Java, Python, JavaScript, PHP, and React**, often backed by **databases** like MySQL or MongoDB, which can be overwhelming for beginners.

- **Server-Side and API Integration**: Most platforms use APIs, cloud hosting, and advanced server configurations.

- **Heavy GUI Dependence**: They require front-end development skills using HTML/CSS, which might not be the focus in a C programming context.

Need for a Simplified System

For academic learning or beginner-level projects, there is a need for:

- A **simplified, terminal-based system**.

- Core logic implementation using basic programming concepts.

- Emphasis on learning **data structures**, memory handling, and procedural programming.

This is where your **C-based e-commerce cart system** becomes extremely relevant—it focuses on the fundamentals, without overwhelming you with web or cloud complexities.

Software Requirement Analysis

Software Requirement Analysis involves identifying the needs and constraints of the system to ensure its proper design and implementation. It is typically categorized into two main parts: **Functional Requirements** and **Non-Functional Requirements**, along with **System Requirements**.

---

1. Functional Requirements

These describe the core functionalities and operations that the system must perform:

- **Product Catalog Display**
  The system shall display a list of available products with ID, name, price, and stock.

- **Add to Cart**
  The user shall be able to add products to the cart by specifying product ID and quantity.

- **View Cart**
  The system shall allow users to view items in their cart with subtotal calculations.

- **Remove from Cart**
  Users shall be able to remove items from the cart by specifying the product ID.

- **Enter Delivery Address**
  The user shall input delivery details including address, city, state, and PIN code (validated).

- **Coupon Code Application**
  The system shall support basic discount codes (e.g., "SAVE10").

- **Bill Generation**
  The system shall generate a final bill with subtotal, tax (GST), shipping, discount, and grand total.

- **Exit and Clear Cart**
  The system shall allow users to exit and ensure memory cleanup.

---

2. Non-Functional Requirements

These define how the system performs the required tasks:

- **Usability**
  The system is user-friendly with a simple text-based menu interface.

- **Reliability**
  Ensures correct calculation of bills, taxes, and discounts.

- **Maintainability**
  Code is modular and structured for easy maintenance and expansion.

- **Portability**
  Runs on any platform with a C compiler (e.g., GCC on Windows/Linux).

- **Performance**
  Lightweight and fast execution as it is console-based with minimal resource usage.

---

3. System Requirements

Hardware Requirements:

- Processor: Intel Pentium i3 or higher
- RAM: Minimum 2 GB
- Disk Space: Minimum 100 MB (for compiler and files)
- Input Device: Keyboard

Software Requirements:

- Operating System: Windows 10 or above / Linux (Ubuntu preferred)
- Compiler: GCC (via Code::Blocks, Turbo C, or any IDE supporting C)
- Text Editor: VS Code, Notepad++, Sublime Text, or built-in editor in IDE

Software Design

```
+-----------------------------+
|       User Interface        |
|  (Console Menu Operations)  |
+-----------------------------+
             |
             v
+-----------------------------+
|     Business Logic Layer    |
|  - Add/View/Remove from Cart|
|  - Billing & Coupon Logic   |
+-----------------------------+
             |
             v
+-----------------------------+
|        Data Layer           |
| - Products Array (Catalog)  |
| - Linked List (Cart)        |
| - Address Structure         |
+---------------------------+
```

Proposed System Features

1. **User Account Management**:
   - Users will have the ability to create accounts, log in, and maintain their order history.
   - Support for user authentication (login/logout) with encrypted passwords (though simple for this implementation).
2. **Product Categories & Search**:
   - Enhanced product categorization (Electronics, Grocery, Fitness, Kitchen, Fashion).
   - Search functionality to help users quickly find products based on name or category.
3. **Enhanced Cart Features**:
   - Cart persistence: Save the cart even when users exit the program (using files for storage).
   - Cart updates will reflect the current stock availability in real-time.
   - Cart item management (adding/removing, updating quantities).
4. **Discount and Coupon System**:
   - Introduce a more sophisticated coupon system with multiple types of discounts (percentage-based, fixed amount).
   - Coupons could be time-bound or product-specific.
5. **Improved Bill Generation**:

    - Add more detailed billing breakdown (tax, discounts, shipping, total).
    - Displaying each item’s subtotal along with cumulative totals.

6. **Delivery Address Management**:
    - Users can save and manage multiple addresses for delivery, making checkout faster.
7. **Order History**:
    - After the order is placed, the system saves the order history for the user, including products bought, total cost, and delivery details.

Flow of Proposed System:

1. **User Login/Registration**:
    - Users first log in or register.
    - On successful login, retrieve user data and previous orders if any.
2. **Browse Products**:
    - User selects a category and browses products.
    - Search option to find products quickly.
3. **Add to Cart**:
    - Users add products to their cart by selecting product ID and quantity.
    - Cart is updated with the new items, and the stock decreases accordingly.
4. **View Cart**:
    - Users can view the items in the cart, modify quantities, or remove items.
    - The subtotal is displayed.
5. **Apply Coupon**:
    - Users apply coupons (if applicable).
    - The system checks if the coupon is valid and applies the discount.

6. **Enter Delivery Address**:
    - Users enter or select a delivery address.
    - System validates address details and PIN code.
7. **Generate Bill & Checkout**:
    - System generates a detailed bill, including taxes, shipping, and discount.
    - Payment options can be implemented for further stages.
    - After checkout, order details are saved in the order history.

System Flowchart for the Proposed System

Below is an overall **flowchart** for the proposed system:

1. **User Login/Registration**
    - If new user, register. If existing, login.
2. **Browse Products**
    - View product catalog (search or by category).
3. **Add to Cart**
    - Choose a product and quantity.
    - Update cart and check stock availability.
4. **View Cart**
    - Modify cart, view item details and subtotal.
5. **Apply Coupon**
    - Apply valid coupon code.
    - Recalculate total with discounts.
6. **Enter Delivery Address**

    - Input delivery address with validation.

7. **Generate Bill & Checkout**

    - Display final bill with taxes, shipping, and discounts.

    - Confirm checkout and save order.

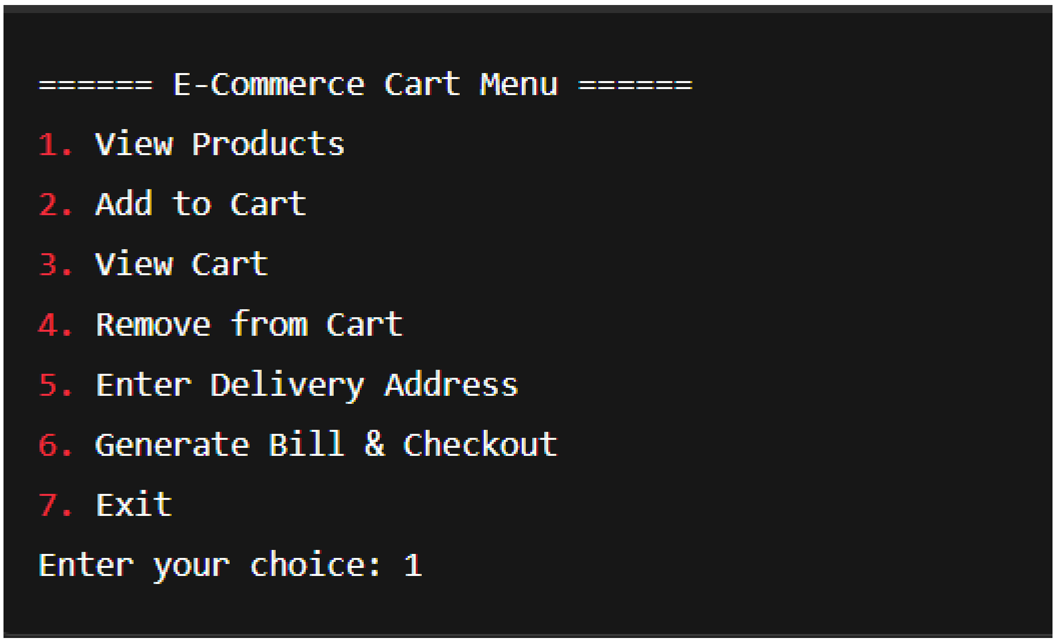
Source code:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

#define MAX_PRODUCTS 25
#define MAX_NAME_LEN 50
#define MAX_ADDR_LEN 100
#define TAX_RATE 0.18
#define SHIPPING_COST 100.0

// Product structure
typedef struct {
    int id;
    char name[MAX_NAME_LEN];
    float price;
    int stock;
} Product;

// Cart node structure
typedef struct CartNode {
    Product product;
    int quantity;
    struct CartNode* next;
} CartNode;

// Delivery address structure
typedef struct {
    char address[MAX_ADDR_LEN];
    char town[50];
    char district[50];
    char state[50];
    char pincode[10];
} DeliveryAddress;
```

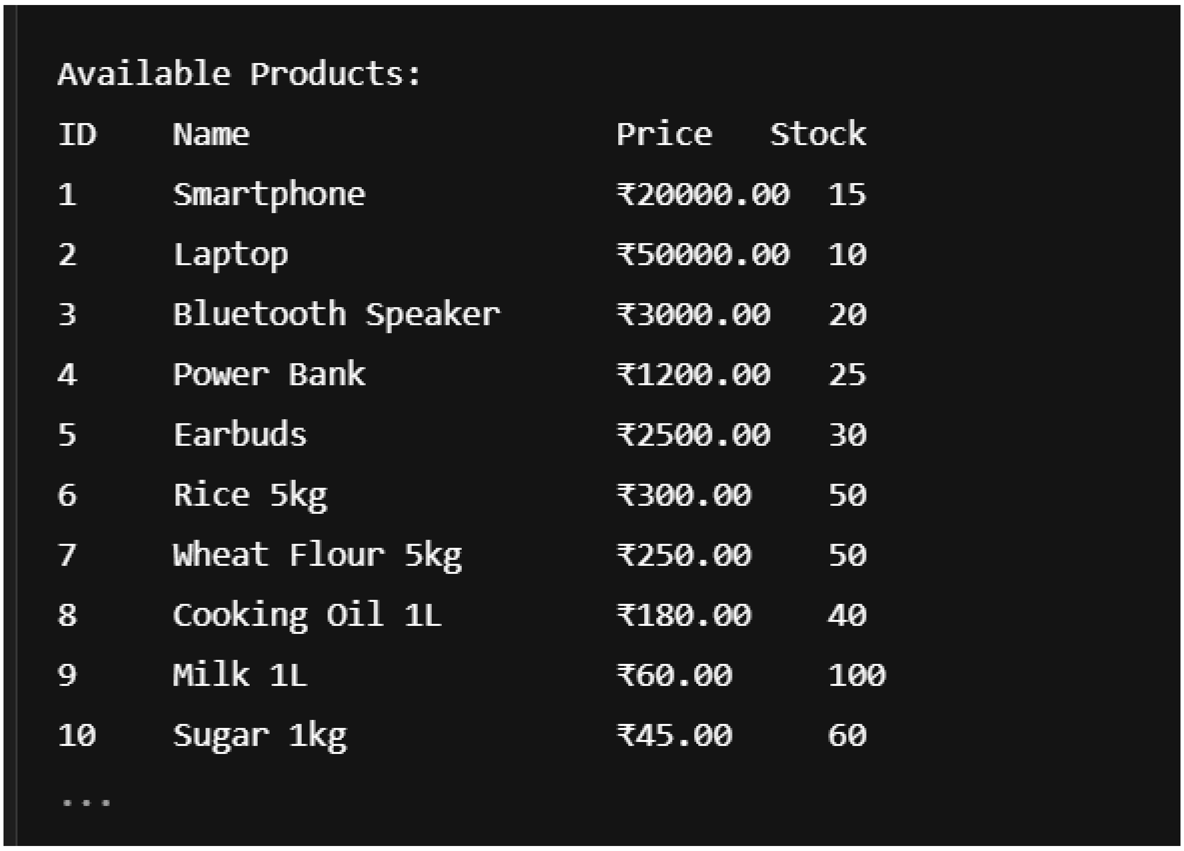
```
// Product catalog
Product products[MAX_PRODUCTS] = {
    // Electronics
    {1, "Smartphone", 20000.0, 15},
    {2, "Laptop", 50000.0, 10},
    {3, "Bluetooth Speaker", 3000.0, 20},
    {4, "Power Bank", 1200.0, 25},
    {5, "Earbuds", 2500.0, 30},

    // Grocery
    {6, "Rice 5kg", 300.0, 50},
    {7, "Wheat Flour 5kg", 250.0, 50},
    {8, "Cooking Oil 1L", 180.0, 40},
    {9, "Milk 1L", 60.0, 100},
    {10, "Sugar 1kg", 45.0, 60},

    // Fitness Accessories
    {11, "Yoga Mat", 800.0, 20},
    {12, "Dumbbells (pair)", 1500.0, 15},
    {13, "Skipping Rope", 250.0, 30},
    {14, "Resistance Band", 400.0, 25},
    {15, "Gym Gloves", 350.0, 20},

    // Kitchen Accessories
    {16, "Non-stick Pan", 1200.0, 25},
    {17, "Knife Set", 1000.0, 15},
    {18, "Cutting Board", 500.0, 30},
    {19, "Blender", 2200.0, 10},
    {20, "Measuring Cups", 300.0, 40},

    // Fashion
    {21, "T-Shirt", 500.0, 50},
    {22, "Jeans", 1200.0, 40},
    {23, "Sneakers", 2500.0, 25},
    {24, "Wrist Watch", 3000.0, 15},
    {25, "Sunglasses", 1500.0, 20}
};

// Function prototypes
void displayProducts();
void addToCart(CartNode**, int, int);
void viewCart(CartNode*);
void removeFromCart(CartNode**, int);
float applyCoupon(char*, float);
void generateBill(CartNode*, DeliveryAddress*);
void clearCart(CartNode**);
void inputDeliveryAddress(DeliveryAddress*);
void printDeliveryAddress(DeliveryAddress*);
int isValidPincode(const char*);
void flushInput();

// Main menu
int main() {
    CartNode* cart = NULL;
    DeliveryAddress address;
    int choice, pid, qty;

    do {
        printf("\n====== E-Commerce Cart Menu ======\n");
        printf("1. View Products\n");
        printf("2. Add to Cart\n");
        printf("3. View Cart\n");
        printf("4. Remove from Cart\n");
        printf("5. Enter Delivery Address\n");
        printf("6. Generate Bill & Checkout\n");
        printf("7. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        flushInput();

        switch (choice) {
            case 1:
                displayProducts();
                break;
            case 2:
                displayProducts();
                printf("Enter product ID: ");
                scanf("%d", &pid);
                printf("Enter quantity: ");
                scanf("%d", &qty);
                flushInput();
                addToCart(&cart, pid, qty);
                break;
            case 3:
                viewCart(cart);
                break;
            case 4:
                viewCart(cart);
                printf("Enter product ID to remove: ");
                scanf("%d", &pid);
                flushInput();
                removeFromCart(&cart, pid);
                break;
            case 5:
                inputDeliveryAddress(&address);
                break;
            case 6:
                viewCart(cart);
                printDeliveryAddress(&address);
                generateBill(cart, &address);
                clearCart(&cart);
                break;
            case 7:
                printf("Thank you for shopping with us!\n");
                break;
            default:
                printf("Invalid choice! Try again.\n");
        }
    } while (choice != 7);

    clearCart(&cart);
    return 0;
}

// Flush stdin
void flushInput() {
    int c;
    while ((c = getchar()) != '\n' && c != EOF);
}

// Display products
void displayProducts() {
    printf("\nAvailable Products:\n");
    printf("ID\tName\t\t\tPrice\tStock\n");
    for (int i = 0; i < MAX_PRODUCTS; i++) {
        printf("%d\t%-20s\t₹%.2f\t%d\n",
               products[i].id,
               products[i].name,
               products[i].price,
               products[i].stock);
    }
}

// Create a cart node
CartNode* createCartNode(Product p, int qty) {
    CartNode* node = (CartNode*)malloc(sizeof(CartNode));
    node->product = p;
    node->quantity = qty;
    node->next = NULL;
    return node;
}

// Add item to cart
```

```
void addToCart(CartNode** head, int productId, int quantity) {
    Product* selected = NULL;
    for (int i = 0; i < MAX_PRODUCTS; i++) {
        if (products[i].id == productId) {
            selected = &products[i];
            break;
        }
    }

    if (!selected) {
        printf("Product not found.\n");
        return;
    }

    if (selected->stock < quantity) {
        printf("Only %d items in stock.\n", selected->stock);
        return;
    }

    CartNode* temp = *head;
    while (temp) {
        if (temp->product.id == productId) {
            temp->quantity += quantity;
            selected->stock -= quantity;
            printf("Increased quantity of %s in cart.\n", selected->name);
            return;
        }
        temp = temp->next;
    }

    CartNode* node = createCartNode(*selected, quantity);
    selected->stock -= quantity;
    node->next = *head;
    *head = node;
    printf("%s added to cart.\n", selected->name);
}

// View cart
void viewCart(CartNode* head) {
    if (!head) {
        printf("\nCart is empty.\n");
        return;
    }

    printf("\nYour Cart:\n");
    printf("ID\tName\t\t\tPrice\tQty\tSubtotal\n");

    float subtotal = 0;
    while (head) {
        float total = head->product.price * head->quantity;
        printf("%d\t%-20s\t₹%.2f\t%d\t₹%.2f\n",
               head->product.id,
               head->product.name,
               head->product.price,
               head->quantity,
               total);
        subtotal += total;
        head = head->next;
    }

    printf("Cart Subtotal: ₹%.2f\n", subtotal);
}

// Remove item from cart
void removeFromCart(CartNode** head, int productId) {
    CartNode *temp = *head, *prev = NULL;

    while (temp && temp->product.id != productId) {
        prev = temp;
        temp = temp->next;
    }

    if (!temp) {
        printf("Item not in cart.\n");
        return;
    }

    // Restore stock
    for (int i = 0; i < MAX_PRODUCTS; i++) {
        if (products[i].id == productId) {
            products[i].stock += temp->quantity;
            break;
        }
    }

    if (!prev)
        *head = temp->next;
    else
        prev->next = temp->next;

    free(temp);
    printf("Item removed from cart.\n");
}

// Apply discount
float applyCoupon(char* code, float subtotal) {
    if (strcmp(code, "SAVE10") == 0) {
        printf("Coupon applied: 10%% off.\n");
        return subtotal * 0.10;
    }
    printf("Invalid coupon code.\n");
    return 0.0;
}

// Input delivery address
void inputDeliveryAddress(DeliveryAddress* addr) {
    printf("\nEnter your delivery address details:\n");

    printf("Full Address: ");
    fgets(addr->address, MAX_ADDR_LEN, stdin);
    addr->address[strcspn(addr->address, "\n")] = '\0';

    printf("Town/City: ");
    fgets(addr->town, 50, stdin);
    addr->town[strcspn(addr->town, "\n")] = '\0';

    printf("District: ");
    fgets(addr->district, 50, stdin);
    addr->district[strcspn(addr->district, "\n")] = '\0';

    printf("State: ");
    fgets(addr->state, 50, stdin);
    addr->state[strcspn(addr->state, "\n")] = '\0';

    do {
        printf("PIN Code: ");
        fgets(addr->pincode, 10, stdin);
        addr->pincode[strcspn(addr->pincode, "\n")] = '\0';

        if (!isValidPincode(addr->pincode)) {
            printf("Invalid PIN Code! Try again.\n");
        }
    } while (!isValidPincode(addr->pincode));

    printf("Delivery address saved.\n");
}

// Print delivery address
void printDeliveryAddress(DeliveryAddress* addr) {
    printf("\nShipping To:\n");
    printf("%s\n%s, %s, %s - %s\n",
           addr->address,
           addr->town,
           addr->district,
        addr->state,
        addr->pincode);
}

// Validate PIN code (6-digit)
int isValidPincode(const char* pin) {
    if (strlen(pin) != 6) return 0;
    for (int i = 0; i < 6; i++) {
        if (!isdigit(pin[i])) return 0;
    }
    return 1;
}

// Bill generation
void generateBill(CartNode* head, DeliveryAddress* addr) {
    if (!head) {
        printf("Your cart is empty. Cannot checkout.\n");
        return;
    }

    float subtotal = 0;

    printf("\n========== BILL ==========\n");
    printf("ID\tName\t\t\tPrice\tQty\tTotal\n");

    while (head) {
        float itemTotal = head->product.price * head->quantity;
        printf("%d\t%-20s\t₹%.2f\t%d\t₹%.2f\n",
               head->product.id,
               head->product.name,
               head->product.price,
               head->quantity,
               itemTotal);
        subtotal += itemTotal;
        head = head->next;
    }

    printf("------------------------------\n");
    printf("Subtotal: ₹%.2f\n", subtotal);

    char coupon[20];
    printf("Enter Coupon Code (or press Enter to skip): ");
    fgets(coupon, sizeof(coupon), stdin);
    coupon[strcspn(coupon, "\n")] = 0;

    float discount = applyCoupon(coupon, subtotal);
    float tax = (subtotal - discount) * TAX_RATE;
    float grandTotal = subtotal - discount + tax + SHIPPING_COST;
    printf("Discount: -₹%.2f\n", discount);
    printf("Tax (18%% GST): ₹%.2f\n", tax);
    printf("Shipping: ₹%.2f\n", SHIPPING_COST);
    printf("================================\n");
    printf("Total Payable: ₹%.2f\n", grandTotal);
    printf("================================\n");

    printDeliveryAddress(addr);
}

// Free memory
void clearCart(CartNode** head) {
    CartNode* temp;
    while (*head) {
        temp = *head;
        *head = (*head)->next;
        free(temp);
    }
}
```

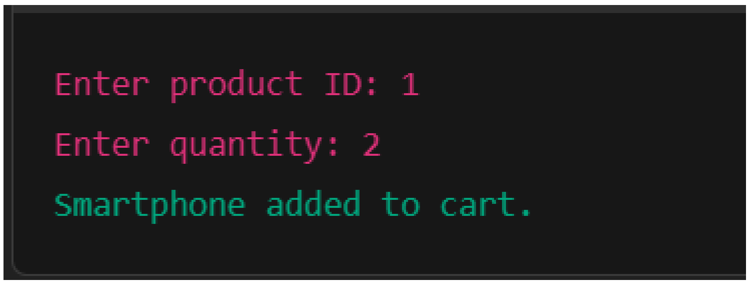
Output :



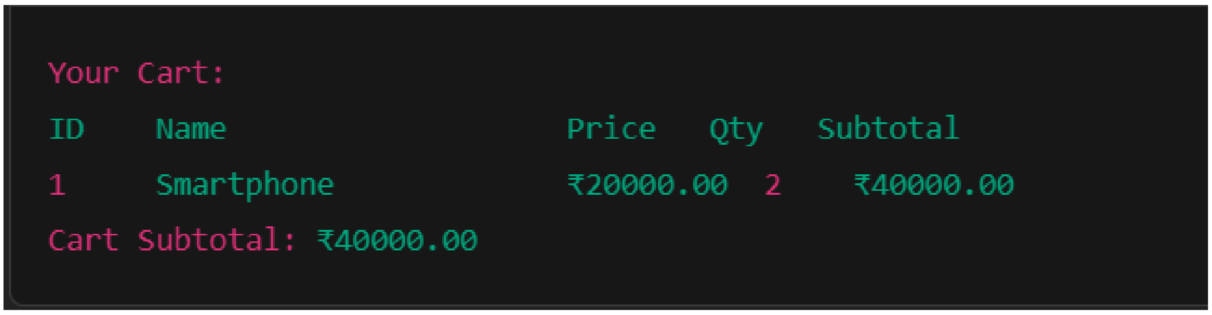
```
====== E-Commerce Cart Menu ======
1. View Products
2. Add to Cart
3. View Cart
4. Remove from Cart
5. Enter Delivery Address
6. Generate Bill & Checkout
7. Exit
Enter your choice: 1
```



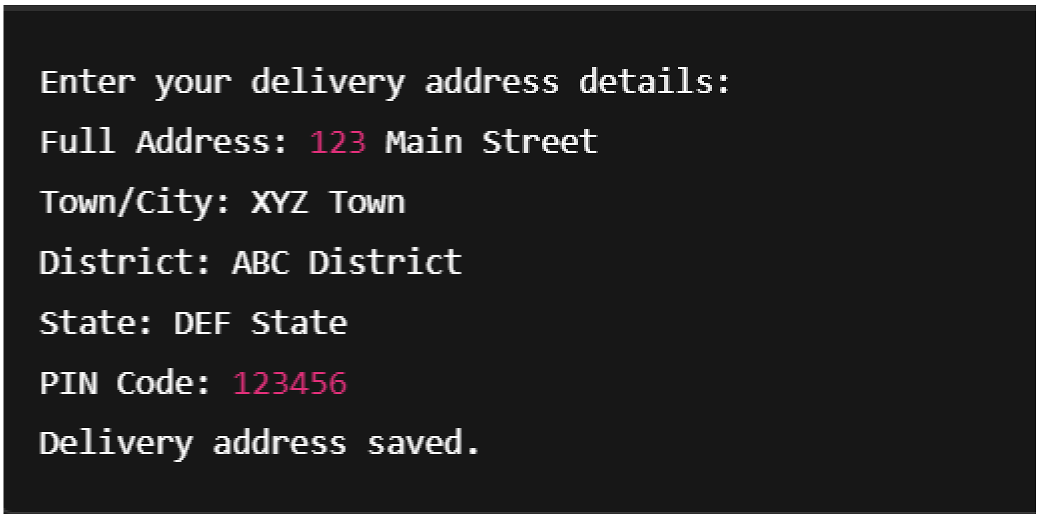
```
Available Products:
ID     Name                   Price     Stock
1      Smartphone             ₹20000.00   15
2      Laptop                 ₹50000.00   10
3      Bluetooth Speaker      ₹3000.00    20
4      Power Bank             ₹1200.00    25
5      Earbuds                ₹2500.00    30
6      Rice 5kg               ₹300.00     50
7      Wheat Flour 5kg        ₹250.00     50
8      Cooking Oil 1L         ₹180.00     40
9      Milk 1L                ₹60.00      100
10     Sugar 1kg              ₹45.00      60
...
```



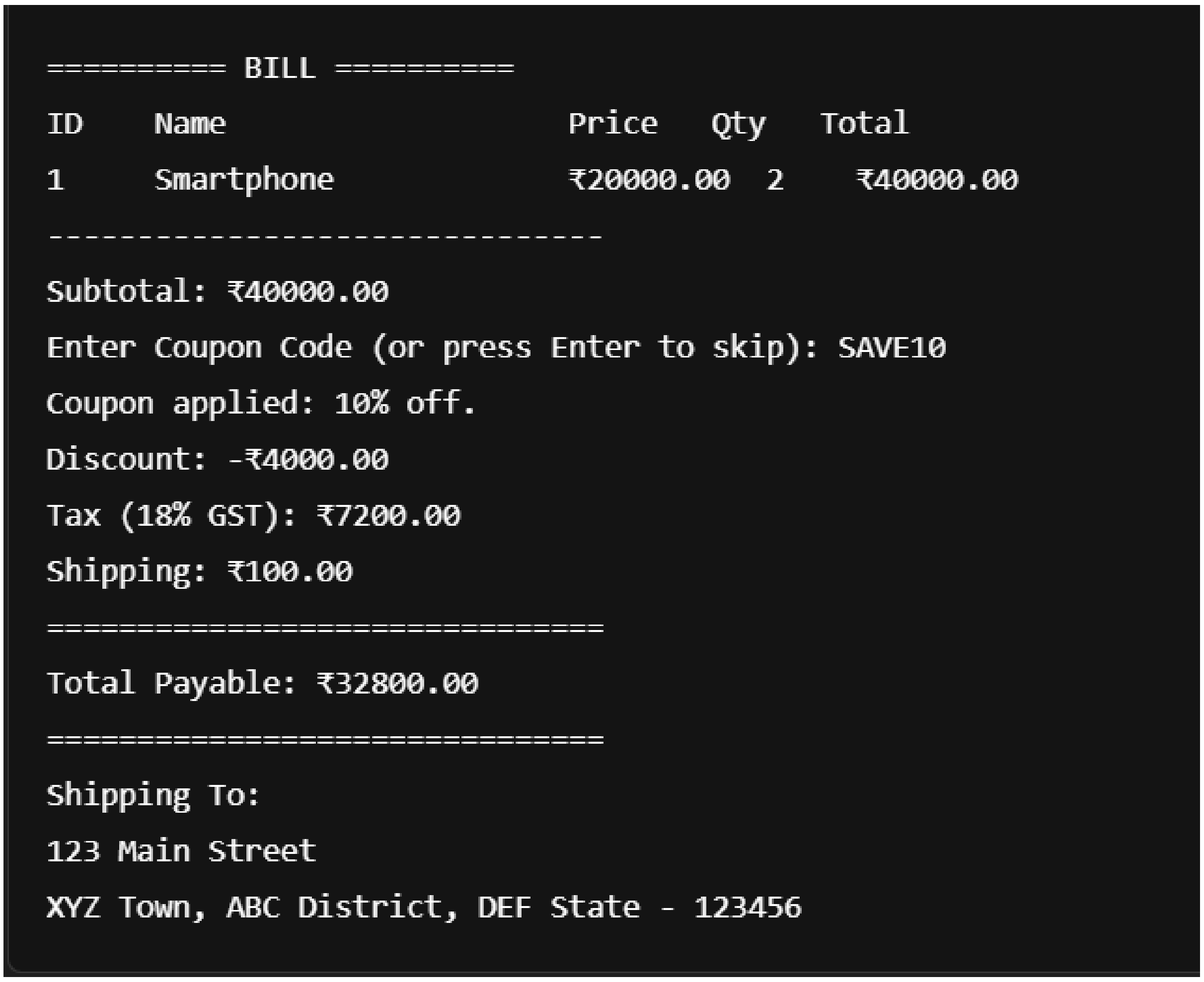
```
Enter product ID: 1
Enter quantity: 2
Smartphone added to cart.
```



```
Your Cart:
ID     Name                   Price     Qty    Subtotal
1      Smartphone             ₹20000.00   2     ₹40000.00
Cart Subtotal: ₹40000.00
```



```
Enter your delivery address details:
Full Address: 123 Main Street
Town/City: XYZ Town
District: ABC District
State: DEF State
PIN Code: 123456
Delivery address saved.
```



```
========== BILL ==========
ID     Name                      Price     Qty    Total
1      Smartphone                ₹20000.00   2     ₹40000.00
------------------------------
Subtotal: ₹40000.00
Enter Coupon Code (or press Enter to skip): SAVE10
Coupon applied: 10% off.
Discount: -₹4000.00
Tax (18% GST): ₹7200.00
Shipping: ₹100.00
==============================
Total Payable: ₹32800.00
==============================
Shipping To:
123 Main Street
XYZ Town, ABC District, DEF State - 123456
```

```
Thank you for shopping with us!
```

Conclusion

The e-commerce cart system developed in C provides a comprehensive solution for managing online shopping. It effectively combines product browsing, cart management, discount application, and checkout functionalities, simulating a simplified online shopping experience.

Key features and conclusions drawn from the system are:

1. **Product Catalog and Cart Management**: The system allows users to view a list of available products, add them to the cart, and remove them as needed. The cart dynamically updates with product quantities, reflecting any changes in real-time.

2. **Dynamic Pricing and Discounts**: It incorporates a basic discount system where users can enter coupon codes to receive discounts. This feature encourages user engagement by providing incentives such as savings on purchases.

3. **Delivery Address Management**: The system includes an option for users to input and validate their delivery address, ensuring that the shipping process is seamless and that valid addresses are captured.

4. **Bill Generation with Tax and Shipping**: After reviewing the cart and applying any discounts, the user is presented with a detailed bill, including the subtotal, tax calculations, discount amount, and shipping costs. This gives users a clear view of their total payable amount before completing the purchase.

5. **User Interaction**: The menu-driven approach offers an intuitive user interface that allows easy navigation through the various features, such as viewing products, managing the cart, and completing the checkout process.

References

1. **Mr.Sivanraj Sir, Bytexl**
   I would like to express my gratitude to Sivanraj Sir, our esteemed trainer from Bytexl, for his continuous support and guidance throughout the course. His expertise in C programming and dedication to teaching has greatly contributed to my understanding of various concepts, especially in developing an e-commerce cart system. Sir's valuable insights and feedback on the design and implementation have helped me refine the project and improve its overall functionality.

Thank you…