



**CHANDIGARH
UNIVERSITY**

Discover. Learn. Empower.

UNIVERSITY INSTITUTE OF COMPUTING

PROJECT REPORT ON Café Billing System

Program Name: BCA
Computing Aptitude/23CAP-308

Submitted by:

Name: Aashika Singh

UID: 23BCA10222

Section: 23BCA-2 “B”

Submitted to:

Name: Mr. Suraj Parkash

Designation: Assist. prof

Project Title: - Café Billing System

2. Aim of the Project: -

To design and implement a simple billing system for a café that allows the user to view the menu, take customer orders, calculate the total bill including GST, and store all orders efficiently.

3. Objective: -

- To automate the café billing process.
- To display and manage café menu items.
- To generate bills with tax calculations.
- To store and view all previous orders.
- To enhance customer service through quick billing.

4. Tools and Technologies Used: -

- Programming Language: C++
- IDE: Code::Blocks
- Operating System: Windows
- Compiler: GCC (GNU Compiler Collection)

5. System Requirements

Hardware Requirements:

- Processor: Intel i3 or higher
- RAM: 4 GB or above
- Storage: Minimum 100 MB free space

Software Requirements:

- Windows OS
- C++ Compiler (Code::Blocks)

6. Algorithm / Logic

1. Display the café menu.
2. Take the customer's name and order details.
3. Store item IDs and quantities.
4. Calculate the subtotal using the item price and quantity.
5. Add GST (20%) to get the final total.
6. Display the detailed bill.
7. Save the order for future reference.
8. Option to view all previous orders.

7. Code Overview: -

The program uses structures for MenuItem and Order.

- The menu stores item details such as ID, name, and price.
- The order structure holds customer information, ordered items, and total amount.
- Functions used:
 - displayMenu() – Displays café menu items.
 - takeOrder() – Takes order details from the customer.
 - calculateTotal() – Calculates subtotal and total amount.
 - printBill() – Prints the formatted customer bill.
 - viewAllOrders() – Displays all stored orders.
- The system runs in a menu-driven format, allowing users to perform operations easily.

Code: -

```
#include <iostream>

#include <iomanip>

#include <string>

using namespace std;

// Structure to store each menu item

struct MenuItem {

    int id;

    string name;

    double price;
```

```
};
```

```
// Structure to store customer order
```

```
struct Order {  
    string customerName;  
    int itemIds[10];  
    int quantities[10];  
    int totalItems;  
    double totalAmount;  
};
```

```
// Function prototypes
```

```
void displayMenu(MenuItem menu[], int size);  
void takeOrder(MenuItem menu[], int size, Order &order);
```

```
double calculateTotal(MenuItem menu[], Order &order);  
void printBill(MenuItem menu[], Order &order);  
void viewAllOrders(Order orders[], int totalOrders);
```

```
int main() {  
    const int MENU_SIZE = 5;
```

```
const int MAX_ORDERS = 20;
```

```
// Initialize menu
```

```
MenuItem menu[MENU_SIZE] = {  
    {1, "Cappuccino", 120.0},  
    {2, "Espresso", 100.0},  
    {3, "Cold Coffee", 90.0},  
    {4, "Brownie", 150.0},  
    {5, "Sandwich", 130.0}  
};
```

```
Order orders[MAX_ORDERS];
```

```
int totalOrders = 0;
```

```
int choice;
```

```
do {
```

```
    cout << "\n===== CAFE BILLING SYSTEM  
===== \n";
```

```
    cout << "1. Display Menu\n";
```

```
    cout << "2. New Order\n";
```

```
    cout << "3. View All Orders\n";
```

```
    cout << "4. Exit\n";
```

```
cout << "Enter your choice: ";
```

```
cin >> choice;
```

```
switch(choice) {
```

```
    case 1:
```

```
        displayMenu(menu, MENU_SIZE);
```

```
        break;
```

```
    case 2:
```

```
        if(totalOrders < MAX_ORDERS) {
```

```
            cout << "\nEnter customer name: ";
```

```
            cin.ignore();
```

```
            getline(cin, orders[totalOrders].customerName
```

```
);
```

```
            takeOrder(menu, MENU_SIZE,  
orders[totalOrders]);
```

```
            orders[totalOrders].totalAmount =  
calculateTotal(menu, orders[totalOrders]);
```

```
            printBill(menu, orders[totalOrders]);
```

```
            totalOrders++;
```

```
        } else {
```

```
            cout << "\nOrder limit reached!\n";
```

```
        }
```

```
break;
```

```
case 3:
```

```
    viewAllOrders(orders, totalOrders);
```

```
    break;
```

```
case 4:
```

```
    cout << "\nThank you for visiting our Cafe! ☕\n";
```

```
    break;
```

```
default:
```

```
    cout << "\nInvalid choice! Try again.\n";
```

```
}
```

```
} while(choice != 4);
```

```
return 0;
```

```
}
```

```
// Function to display the cafe menu
```

```
void displayMenu(MenuItem menu[], int size) {
```

```
    cout << "\n----- MENU ----- \n";
```

```
    cout << left << setw(5) << "ID" << setw(20) << "Item" <<  
    "Price (Rs)\n";
```



```
cout << "-----\n";  
for(int i = 0; i < size; i++) {  
    cout << left << setw(5) << menu[i].id << setw(20) <<  
menu[i].name << menu[i].price << endl;  
}  
cout << "-----\n";  
}
```

// Function to take customer order

```
void takeOrder(MenuItem menu[], int size, Order &order) {
```

```
    displayMenu(menu, size);
```

```
    cout << "\nHow many items do you want to order? ";
```

```
    cin >> order.totalItems;
```

```
    for(int i = 0; i < order.totalItems; i++) {
```

```
        cout << "\nEnter item ID for item " << i+1 << ": ";
```

```
        cin >> order.itemIds[i];
```

```
        cout << "Enter quantity: ";
```

```
        cin >> order.quantities[i];
```

```
    }
```

```
}
```

```
// Function to calculate total bill

double calculateTotal(MenuItem menu[], Order &order) {
    double subtotal = 0.0;
    for(int i = 0; i < order.totalItems; i++) {
        int id = order.itemIds[i];
        int qty = order.quantities[i];
        subtotal += menu[id - 1].price * qty;
    }

    return subtotal;
}

// Function to print bill for the customer

void printBill(MenuItem menu[], Order &order) {
    double subtotal = calculateTotal(menu, order);
    double tax = subtotal * 0.20; // 5% GST
    double total = subtotal + tax;

    cout << "\n===== BILL =====\n";
    cout << "Customer: " << order.customerName << endl;
    cout << "-----\n";
```

```
cout << left << setw(20) << "Item" << setw(10) << "Qty"
<< "Price\n";
```

```
cout << "-----\n";
```

```
for(int i = 0; i < order.totalItems; i++) {
```

```
    int id = order.itemIds[i];
```

```
    int qty = order.quantities[i];
```

```
    cout << left << setw(20) << menu[id - 1].name
```

```
<< setw(10) << qty
```

```
    << menu[id - 1].price * qty << endl;
```

```
}
```

```
cout << "-----\n";
```

```
cout << "Subtotal: Rs " << subtotal << endl;
```

```
cout << "GST (20%): Rs " << tax << endl;
```

```
cout << "Total: Rs " << total << endl;
```

```
cout << "=====\n";
```

```
}
```

```
// Function to display all previous orders
```

```
void viewAllOrders(Order orders[], int totalOrders) {
```

```
if(totalOrders == 0) {  
    cout << "\nNo previous orders.\n";  
    return;  
}
```

```
cout << "\n===== ALL ORDERS =====\n";  
for(int i = 0; i < totalOrders; i++) {
```

```
    cout << i+1 << ". " << orders[i].customerName  
        << " - Rs " << fixed << setprecision(2)  
        << orders[i].totalAmount << endl;  
}
```

```
    cout << "===== \n";  
}
```

8. Output: -

```
"C:\Users\ashika singh\OneD x + v

===== CAFE BILLING SYSTEM =====
1. Display Menu
2. New Order
3. View All Orders
4. Exit
Enter your choice: 1

----- MENU -----
ID  Item          Price (Rs)
-----
1   Cappuccino     120
2   Espresso       100
3   Cold Coffee    90
4   Brownie        150
5   Sandwich       130
-----

===== CAFE BILLING SYSTEM =====
1. Display Menu
2. New Order
3. View All Orders
4. Exit
Enter your choice: 2

Enter customer name: Aashika Singh

----- MENU -----
ID  Item          Price (Rs)
-----
1   Cappuccino     120
2   Espresso       100
3   Cold Coffee    90
4   Brownie        150
5   Sandwich       130
-----

How many items do you want to order? 2
Enter item ID for item 1: 2
Enter quantity: 2
```

```
"C:\Users\ashika singh\OneD x + v

Enter your choice: 2

Enter customer name: Aashika Singh

----- MENU -----
ID  Item          Price (Rs)
-----
1   Cappuccino     120
2   Espresso       100
3   Cold Coffee    90
4   Brownie        150
5   Sandwich       130
-----

How many items do you want to order? 2

Enter item ID for item 1: 2
Enter quantity: 2

Enter item ID for item 2: 2
Enter quantity: 2

===== BILL =====
Customer: Aashika Singh
-----
Item          Qty    Price
-----
Espresso      2      200
Espresso      2      200
-----
Subtotal: Rs 400
GST (20%): Rs 80
Total: Rs 480
=====

===== CAFE BILLING SYSTEM =====
1. Display Menu
2. New Order
3. View All Orders
4. Exit
Enter your choice:
```

9. Conclusion

The Cafe Billing System successfully automates the café billing process. It provides an easy way to handle multiple customer orders, calculate totals with tax, and generate bills quickly. The project demonstrates efficient use of structures, arrays, and functions in C++.

10. Learning Outcomes

- Gained practical knowledge of C++ programming.
- Learned how to use structures and arrays to manage data.
- Understood file-less order management in C++.
- Developed problem-solving and logical thinking skills.