UNIVERSITY INSTITUTION OF COMPUTING

BRANCH:BACHELOR OF COMPUTER APPLICATION

Mini Project: Restaurant biling system

OBJECT ORIENTED PROGRAMMING

SUBMITTED BY-	SUBMITTED TO-
NAME: Ritika Rastogi	Mr.Jitendra Sir
UID: 24BCU10027	Assistant Professor
SECTION: 24BCU-1	
NAME: Tamanna Sareen	
UID: 24BCA20011	
SECTION: 24BCV-1	
SEMESTER: 3rd	

INTRODUCTION

The Restaurant Billing System is a mini project developed in C++ using the fundamental concepts of Object-Oriented Programming

The main goal of this project is to automate the process of generating restaurant bills, making it quick, easy, and error-free.

Instead of calculating totals manually, this system allows users to select menu items, enter quantities, and view the total bill instantly through a console-based interface.

In today’s fast-paced world, restaurants and cafes require efficient billing systems to handle multiple orders quickly and accurately. This project demonstrates how OOP concepts such as classes, objects, inheritance, constructors, destructors, and dynamic memory allocation can be used to design such a system effectively.

The system allows users to:

- View the available menu items,
- Select the food items and enter quantities,
- Automatically calculate and display the total amount, and
- Generate a final bill for the customer.

This project serves as a simple and practical example of how C++ can be used to develop real-world applications that are both structured and efficient.

ABSTRACT

The Restaurant Billing System is a console-based mini project created in C++ to demonstrate the practical use of Object-Oriented Programming (OOP) concepts in solving daily life problems.
The system helps restaurant staff or customers calculate bills easily by selecting menu items and quantities. Each item has a name, price, and quantity, and the program automatically computes the total amount due.

The project makes use of:

- Constructors to initialize menu items,
- Inheritance to differentiate between categories like *Drinks* and *Main Course*,
- Polymorphism to handle price calculations differently for various types of food,
- Destructors to clean up memory after billing is done, and
- Dynamic memory allocation to create and manage objects during runtime.

This project aims to help learners understand how to implement a simple and functional billing system using C++. It can be further expanded in the future with features like GST calculation, item removal, discounts, and digital receipts.

SYSTEM CONFIGURATION

The Restaurant Billing System is a lightweight and portable C++ program that can be easily compiled and executed on any computersystem.
It does not require high-end hardware or additional libraries — only a standard C++ compiler.

Hardware Requirements:

- Processor: Dual-core or above
- RAM: Minimum 2 GB
- Storage: 500 MB free space

Software Requirements:

- Operating System: Windows / Linux / macOS
- Compiler: Turbo C++, GCC, or Code::Blocks
- Tools: Any IDE or text editor such as Dev C++, Visual Studio Code, or Code::Blocks

This configuration makes it an ideal mini project for students and beginners learning the basics of OOP in C++.

CODE

```
#include <iostream>

#include <string>

using namespace std;

class MenuItem {

   string name;

   float price;
```

```
public:
   MenuItem(string n, float p) {
      name = n;
      price = p;
   }
   void display() {
      cout << name << " - Rs." << price << endl;
   }
   float getPrice() { return price; }
   string getName() { return name; }
};

class Order {
   string customerName;
   MenuItem *item;
   int quantity;
public:
   Order(string cname, MenuItem *i, int q) {
      customerName = cname;
      item = i;
      quantity = q;
   }
   void showBill() {
      cout << "\n--- BILL DETAILS ---\n";
```

```
        cout << "Customer Name: " << customerName << endl;
        cout << "Item: " << item->getName() << endl;
        cout << "Quantity: " << quantity << endl;
        cout << "Total Amount: Rs." << item->getPrice() * quantity <<
endl;
    }
    ~Order() {
        cout << "\nMemory cleared for " << customerName << "'s
order.\n";
    }
};

int main() {
    cout << "------ Welcome to Foodie's Restaurant ------\n";
    MenuItem m1("Burger", 120.0);
    MenuItem m2("Pizza", 250.0);
    MenuItem m3("Cold Drink", 60.0);

    cout << "\n--- MENU ---\n";
    m1.display();
    m2.display();
    m3.display();

    string name;
    int choice, qty;

    cout << "\nEnter your name: ";
    getline(cin, name);

    cout << "Enter item number (1-Burger, 2-Pizza, 3-Cold Drink): ";
    cin >> choice;

    cout << "Enter quantity: ";
    cin >> qty;

    MenuItem *selectedItem;
    if (choice == 1) selectedItem = &m1;
    else if (choice == 2) selectedItem = &m2;
    else selectedItem = &m3;

    Order *o1 = new Order(name, selectedItem, qty);
    o1->showBill();

    delete o1;
    return 0;
}
```

OUTPUT

```
------ Welcome to Foodie's Restaurant ------

--- MENU ---
Burger - Rs.120
Pizza - Rs.250
Cold Drink - Rs.60

Enter your name: Ritika
Enter item number (1-Burger, 2-Pizza, 3-Cold Drink): 2
Enter quantity: 2

--- BILL DETAILS ---
Customer Name: Ritika
Item: Pizza
Quantity: 2
Total Amount: Rs.500

Memory cleared for Ritika's order.
```

CONCLUSION

The Movie Ticket Booking System project demonstrates the practical application of Object-Oriented Programming (OOP) concepts in C++. Through this project, we learned how constructors can initialize objects, destructors can handle memory cleanup, and dynamic memory allocation can be used for creating objects at runtime.

The system allows users to book movie tickets efficiently, displaying booking details and calculating the total cost automatically. It also highlights the use of encapsulation and data abstraction by keeping movie and booking details private, accessible only through public methods.

Overall, this project provides a simple yet effective simulation of a real-world scenario, making it easier for beginners to understand OOP principles and apply them in practical programming tasks. The system can be further enhanced in the future by adding features like seat selection, multiple movie shows, online payment integration, and a database to store booking history.

REFERENCES

1. *Programming in C++* by E. Balagurusamy
2. *Object-Oriented Programming with C++* by Robert Lafore
3. Official C++ Documentation — https://cplusplus.com
4. TutorialsPoint – https://www.tutorialspoint.com/cplusplus
5. GeeksforGeeks — https://www.geeksforgeeks.org/oops-concepts-in-cpp

---