

REPORT FILE

VITYARTHI PROJECT



TOPIC: Restaurant billing system

BY:

JHEEL SHAH

(25BAI10895)



VIT[®]
—
BHOPAL

VIT Bhopal University

Kotrikalan – 466114

Madhya Pradesh INDIA

December 2025

Project overview

- **Introduction:**

The *Restaurant Billing System* is a Python program that allows users to order food items from a predefined menu, enter quantities, and calculate the total bill. The system supports multiple orders, validates user inputs, and includes an optional discount code feature. After the ordering process, the program displays a clear and formatted final bill showing the subtotal, discount applied, and the final payable amount. This project demonstrates basic Python concepts such as functions, dictionaries, loops, conditionals, and exception handling.

- **Problem Statement:**

Restaurants often rely on manual methods for taking orders and generating bills, which can lead to errors, delays, and inefficiencies. Calculating totals, applying discounts, and managing multiple items manually increases the chances of mistakes and reduces overall customer satisfaction. There is a need for a simple, reliable, and automated billing system that can streamline the order-to-bill process without requiring advanced hardware or complex setup.

The goal of this project is to develop a **Python-based Restaurant Billing System** that automates menu display, order processing, subtotal calculation, discount application, and final bill generation. The system should be user-friendly, error-resistant, and suitable for small restaurants or educational purposes.

➤ Functional Requirements:

- Display menu with item names and prices
- Accept customer name
- Allow selection of items
- Validate selected item
- Accept item quantity with error handling
- Allow multiple orders
- Calculate subtotal
- Accept discount code and apply discount
- Generate final bill with subtotal, discount, and total amount
- Display thank-you message

➤ Non-Functional Requirements:

- **Usability:** Easy-to-use text interface
- **Reliability:** Handles invalid inputs gracefully
- **Performance:** Fast bill generation with minimal processing
- **Maintainability:** Clean, modular code using functions
- **Portability:** Runs on any OS with Python installed

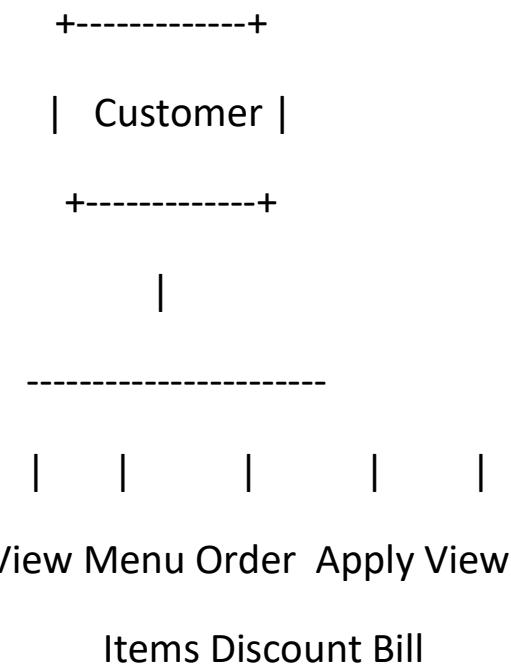
➤ System Architecture:

User → Input Handling → Menu & Order Processing → Billing Engine → Discount Logic → Output (Final Bill)

This architecture uses a linear flow where the user's data passes through processing blocks (loops, decisions, calculations) and outputs a structured bill.

➤ Design Diagrams:

A. Use Case Diagram:



B)Sequence Diagram:

Customer System

```
|           |
|----Name--->|
|<--Greet----|
|----Item---->|
|<--Validate--|
|---Quantity->|
|<--Calculate-|
|---More?----|
|---Code---->|
|<--Discount--|
|<----Bill-----|
```

C)Class Component diagram:

```
+-----+
| RestaurantBilling   |
+-----+
| - menu          |
| - total          |
+-----+
| + greet ()      |
| + order_items() |
| + apply_discount() |
| + generate_bill() |
+-----+
```

➤ Design Decisions & Rationale:

Decision	Reason
Using dictionary for menu	Fast item lookup and price mapping
Using functions	Improves readability and modularity
Using loops	Enables multiple orders
Using try-except	Prevents crashes due to invalid input
Using f-strings	Clean and readable output formatting

➤ Implementation Details

The program is implemented using Python 3.x and consists of:

- A **greet ()** function
- A **restaurant ()** function
- A dictionary-based menu
- Loops for additional orders
- Discount code mechanism
- Final bill formatting using print statements

Code handles input errors, invalid choices, and ensures billing accuracy.

➤ Screenshots/Results:

□ Code:

```
restaurant.py > 🏠 restaurant
1 def greet(name):
2     print(f"Welcome to our reastaurant, {name}!")
3
4 def restaurant():
5     menu = [
6         'Coffee':80,
7         'Pizza':120,
8         'Pasta':130,
9         'Burger':70,
10        'Maggie':40
11    ]
12
13
14    print("Here's the menu: ")
15    print(" Burger 🍔 = Rs70 \n Coffee ☕ = Rs80 \n Maggie 🍔 = Rs40 \n Pasta 🍝 = Rs130 \n Pizza 🍕 = Rs120")
16    total=0
17
18    order=input("Please select your first item: ").title()
19    if order in menu:
20        print ("Great",order,"selected. ✓ ")
21        while True:
22            try:
23                quantity= int(input("please select the quantity of your item: "))
24                if quantity>0:
25                    break
26                else:
27                    print("Quantity must be a positive number.")
28            except ValueError:
29                print("Invalid input. Please enter a whole number for the quantity.")
30            total += menu[order] * quantity
31    else:
32        print("Invalid choice! ❌ ")
33
```

```
restaurant.py > 🏠 restaurant
4 def restaurant():
35     while True:
36         again=input("Do you want to order something else: (Yes/No)").title()
37         if again != "Yes":
38             break
39
40         order = input("Please select your next item: ").title()
41         if order in menu:
42             print("Great", order, "selected.")
43             quantity= int(input("please select the quantity of your item: "))
44             total += menu[order] * quantity
45         else:
46             print("Choice not available")
47
48         discount_amount = 0
49         original_total = total
50         valid_code = "CODE10"
51         discount_percentage = 0.10
52
53         discount_code = input("\nHave a discount code? Enter it here (e.g., CODE10) or press Enter to skip: ").upper()
54
55         if discount_code == valid_code:
56             discount_amount = total * discount_percentage
57             total -= discount_amount # Apply the discount
58
59             print(f"✓ Discount Code **{valid_code}** accepted! -Rs{discount_amount:.2f} applied.")
60         elif discount_code:
61
62             print("❌ Invalid discount code. No discount applied.")
63
64
65         # --- Final Bill Display ---
66         print("\n--- FINAL BILL ---")
```

```
54  
55 # --- Final Bill Display ---  
56 print("\n--- FINAL BILL ---")  
57  
58 # You can show the detailed receipt here!  
59  
60 print("-----")  
61 print(f"Subtotal: Rs{original_total:.2f}")  
62  
63 if discount_amount > 0:  
64     print(f"Discount: -Rs{discount_amount:.2f}")  
65     print("-----")  
66  
67 print(f"** 🍔 FINAL AMOUNT DUE: Rs.{total:.2f} **")  
68  
69 print("\n✿ Thank you for dining with us! ✿")  
70 print("We hope you enjoyed your meal 😊")  
71 print("Please visit again soon! 🍔")  
72 print("👉 Got feedback? We'd love to hear from you.")  
73 print("⭐ Rate us from 1 to 5 stars before you go! ⭐")  
74  
75 user_name=input("Enter your name: ")  
76 greet(user_name)  
77 restaurant()
```

□ OUTPUT:

```
Enter your name: JHEEL SHAH
Welcome to our reastaurant, JHEEL SHAH!
Here's the menu:
Burger 🍔 = Rs70
Coffee ☕ = Rs80
Maggie 🍜 = Rs40
Pasta 🍝 = Rs130
Pizza 🍕 = Rs120
Please select your first item: Burger
Great Burger selected. ✓
please select the quantity of your item: 2
Do you want to order something else: (Yes/No)YEs
Please select your next item: pizza
Great Pizza selected.
please select the quantity of your item: 2
Do you want to order something else: (Yes/No)no

Have a discount code? Enter it here or press Enter to skip: CODE10
✓ Discount Code **CODE10** accepted! -Rs38.00 applied.

--- 📋 FINAL BILL 📋 ---
--- 📋 FINAL BILL 📋 ---
-----
Subtotal: Rs380.00
Discount: -Rs38.00
-----
** 💰 FINAL AMOUNT DUE: Rs.342.00**

✨ Thank you for dining with us! ✨
We hope you enjoyed your meal 😊
Please visit again soon! 🍔
👉 Got feedback? We'd love to hear from you.
⭐ Rate us from 1 to 5 stars before you go! ⭐
```

➤ Testing Approach:

Test cases performed:

- Valid item input
- Invalid item input
- Numeric and non-numeric quantity
- Zero or negative quantity
- Multiple orders
- Valid discount code
- Invalid discount code
- Empty discount input

All tests passed successfully.

➤ Challenges Faced:

- Handling incorrect quantity inputs
- Ensuring program does not crash on invalid entries
- Designing clean bill formatting
- Managing loop flow for multiple orders

➤ Learnings & Key Takeaways

- Use of loops and conditionals
- Importance of input validation
- Structuring programs with functions
- Using dictionaries effectively
- Basic understanding of billing systems
- Writing user-friendly console applications

➤ Future Enhancements

- Add GST and service tax
- Add a GUI using Tkinter or PyQt
- Connect to a database for storage
- Add option to remove or modify ordered items
- Print or export the bill as a PDF

➤ References

- Python Official Documentation
- Classroom notes
- Self-designed logic
- No external code copied