CAFE ORDER MANAGEMENT SYSTEM

SUBMITTED BY

Name: Aryan Arora

Registration Number: 25BAS10008

UNDER THE SUPERVISION OF

Prof. Sandip Mal

VIT BHOPAL UNIVERSITY

(Madhya Pradesh, India)

2025

PAGE 2: INDEX

PAGE 3: DESIGN

1. System Overview

The **Cafe Order Management System** is a console-based Point of Sale (POS) application designed to streamline the billing process for a small cafe. The system utilizes Python's standard libraries to manage menu items, calculate totals using typed arrays for efficiency, and generate timestamped receipts.

2. Modules and Logic

The project is structured into modular functions to ensure code reusability and clarity:

- **Data Structure (Global Variables):**
  - MENU (Dictionary): Stores item names as keys and prices as values (Float).
  - current_order (List): A dynamic list to store the items selected by the customer.
  - STAFF_NAMES (List): Used to randomly assign a cashier for the session.
- **Core Functions:**
  - add_item(order_list): Takes user input, validates it against the MENU dictionary, and appends valid items to the current order list.
  - get_total_price(order_list): Converts the prices of selected items into a typed array (datatype double) to perform high-precision arithmetic summation.

    - print_bill(is_checkout): Generates a formatted receipt. If is_checkout is true, it generates a unique Order ID and appends the current system timestamp using the datetime module.
- **Managerial Statistics:**
    - show_manager_stats(): Demonstrates the use of the array module to analyze pricing data. It calculates the Minimum, Maximum, and Average price of items currently on the menu.

PAGE 4: SOURCE CODE

File Name: cafe_pos.py

Language: Python 3.x

Python

```
"""
Project: Cafe Order Management System
Description: A simple POS system for a cafe.
Allows users to add items, view bills, and checkout with a timestamp.
Submitted by: Aryan Arora (25BAS10008)
"""


import datetime
import random
import array


# --- DATA CONFIGURATION ---
MENU = {
    "Coffee": 80.00,
    "Espresso": 120.00,
    "Tea": 50.00,
    "Muffin": 90.00,
    "Croissant": 85.00,
    "Sandwich": 150.00,
    "Water": 60.00
}

current_order = []
STAFF_NAMES = ["Sarah", "Ben", "Chloe", "Mike", "Alex"]

# --- SYSTEM FUNCTIONS ---

def show_menu_header():
    """Prints the system header with current time and staff name."""
    now = datetime.datetime.now()
    time_str = now.strftime("%I:%M %p")
    staff = random.choice(STAFF_NAMES)

    print("\n" + "="*30)
    print(f" CAFE POS SYSTEM - {time_str}")
    print(f" Cashier on duty: {staff}")
    print("="*30)

def print_main_options():
    """Just shows the numbered options."""
    print("1. Show Menu")
    print("2. Add Item")

    print("3. View Bill")

    print("4. Checkout")

    print("5. Manager Stats (Array Mode)")

    print("6. Exit")

    print("-" * 30)


def show_cafe_menu():

    """Iterates through the menu dict and displays options."""

    print("\n--- MENU ---")

    for item, price in MENU.items():

        print(f"{item:<12} : ₹{price:.2f}")

    print("-" * 12)


def add_item(order_list):

    """Asks user for an item and adds it to the order if valid."""

    item_input = input("Item name: ").strip().title()


    if item_input in MENU:

        order_list.append(item_input)

        print(f"--> Added 1 {item_input}")

    else:

        print(f"!! Error: '{item_input}' is not on the menu.")


def get_total_price(order_list):

    """
    Calculates total price.

    Converts the list of prices to a typed array (float) for calculation.

    """

    if not order_list:

        return 0.0



    prices_list = [MENU[item] for item in order_list]

    prices_array = array.array('d', prices_list)



    return sum(prices_array)



def print_bill(order_list, is_checkout=False):

    """Displays the current list of items and the total price."""

    print("\n--- RECEIPT ---")

    if not order_list:

        print("(No items added yet)")

        return



    for item in order_list:

        print(f"{item:<12} : ₹{MENU[item]:.2f}")



    print("-" * 20)



    total = get_total_price(order_list)

    print(f"TOTAL        : ₹{total:.2f}")
    if is_checkout:
        order_id = random.randint(1000, 9999)
        timestamp = datetime.datetime.now().strftime("%Y-%m-%d %H:%M")
        print(f"\nOrder #{order_id} | {timestamp}")
        print("Thank you for visiting!")

def show_manager_stats():
    """
    Calculates basic stats (Avg, Min, Max) about the menu pricing.
    """
    print("\n--- MANAGER STATS ---")

    prices = list(MENU.values())
    price_arr = array.array('d', prices)

    total_items = len(price_arr)
    avg_price = sum(price_arr) / total_items
    min_price = min(price_arr)
    max_price = max(price_arr)

    print(f"Total Items   : {total_items}")
    print(f"Average Price : ₹{avg_price:.2f}")
    print(f"Lowest Price  : ₹{min_price:.2f}")
    print(f"Highest Price : ₹{max_price:.2f}")

# --- MAIN EXECUTION FLOW ---
```

```
def main():
    while True:
        show_menu_header()
        print_main_options()

        choice = input("Select (1-6): ").strip()

        if choice == '1':
            show_cafe_menu()

        elif choice == '2':
            add_item(current_order)

        elif choice == '3':
            print_bill(current_order)

        elif choice == '4':
            if current_order:
                print_bill(current_order, is_checkout=True)
                current_order.clear() # Reset for next customer
            else:
                print("Cannot checkout an empty order!")

        elif choice == '5':
            show_manager_stats()
        elif choice == '6':
            print("System shutting down... Bye!")
            break

        else:
            print("Invalid choice, try again.")

        input("\n[Press Enter]")

if __name__ == "__main__":
    main()
```

PAGE 5: REFERENCES

1. **Python Software Foundation.** (2025). *Python 3.12 Documentation*. Retrieved from https://docs.python.org/3/
2. **Lutz, M.** (2013). *Learning Python* (5th ed.). O'Reilly Media.
3. **Python Standard Library.**
    - datetime module: Basic date and time types.
    - array module: Efficient arrays of numeric values.
    - random module: Generate pseudo-random numbers.