FEAST EXPRESS

BY: *ANUSREE RAVICHANDRAN*, DATED: *11/09/2024*

AIM OF THE PROJECT

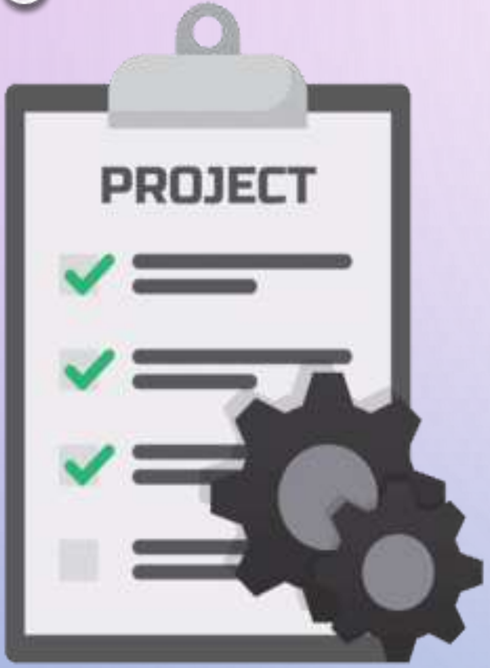
This project aims to develop an online food ordering system where customers can select menu items, place orders, and choose payment methods. It incorporates OOP principles like abstraction, encapsulation, inheritance, and polymorphism for a seamless process. Goals of this project are:

- **Provides an interactive food ordering experience**
- **Flexible payment options**
- **Discount mechanism for prime members or high-value orders**
- **User role-based functionality**

BUSINESS PROBLEM / PROBLEM STATEMENT

This project solves the need for a efficient online food ordering system that simplifies order placement and provides flexible, secure payment options. Restaurants and food delivery services struggle with manual order processing, managing discounts, and offering multiple payment methods while ensuring customer satisfaction.

PROJECT DESCRIPTION



SCOPE: This project automates the online food ordering process for restaurants and delivery services, streamlining customer orders and payments.

OBJECTIVES: To Automate Order Process, facilitate Multiple Payment Options, Personalization & OOP Principles Demonstration.

KEY FEATURES:

- Menu browsing, order placement, and discount calculation.
- Flexible payment options with secure processing.
- Personalized discounts for prime members and large orders.

FUNCTIONALITIES

- **Welcome Message:** Greeting the user prior to order process.
- **Role-based Access:** System provides access only to "***Customer***" or denies the access for "***Admin.***“
- **Customer Information Collection:** Categories the user, to personalize the experience and determine if a discount should be applied or not.
- **Menu Display:** System displays a list of available food items along with their prices.

- **Order Placement:** System tracks the items and calculates the total running cost.

FUNCTIONALITIES (Continued)

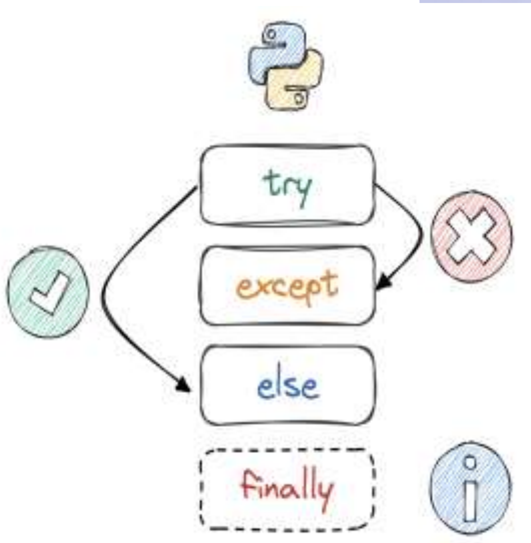
- **Order Summary and Discount Application:** Total order value exceeds a certain threshold (Rs 1200) or if the customer is a Prime Member, 30% discount or 40% discount is applied to the total amount.

- **Payment Method Selection:** Users can choose from three payment methods: Debit Card, Cash on Delivery (COD), or UPI Payment.

- **Order Completion:** Ensures the customer has clarity about the order status.

- **Error Handling:** The system includes basic error handling for invalid menu item selections and incorrect payment choices, ensuring a smooth user experience even in case of incorrect input.

INPUT VERSATILITY WITH ERRORS AND EXCEPTIONS



- **User Role Input:** Displays an error message for non-customers or Admins for ordering food

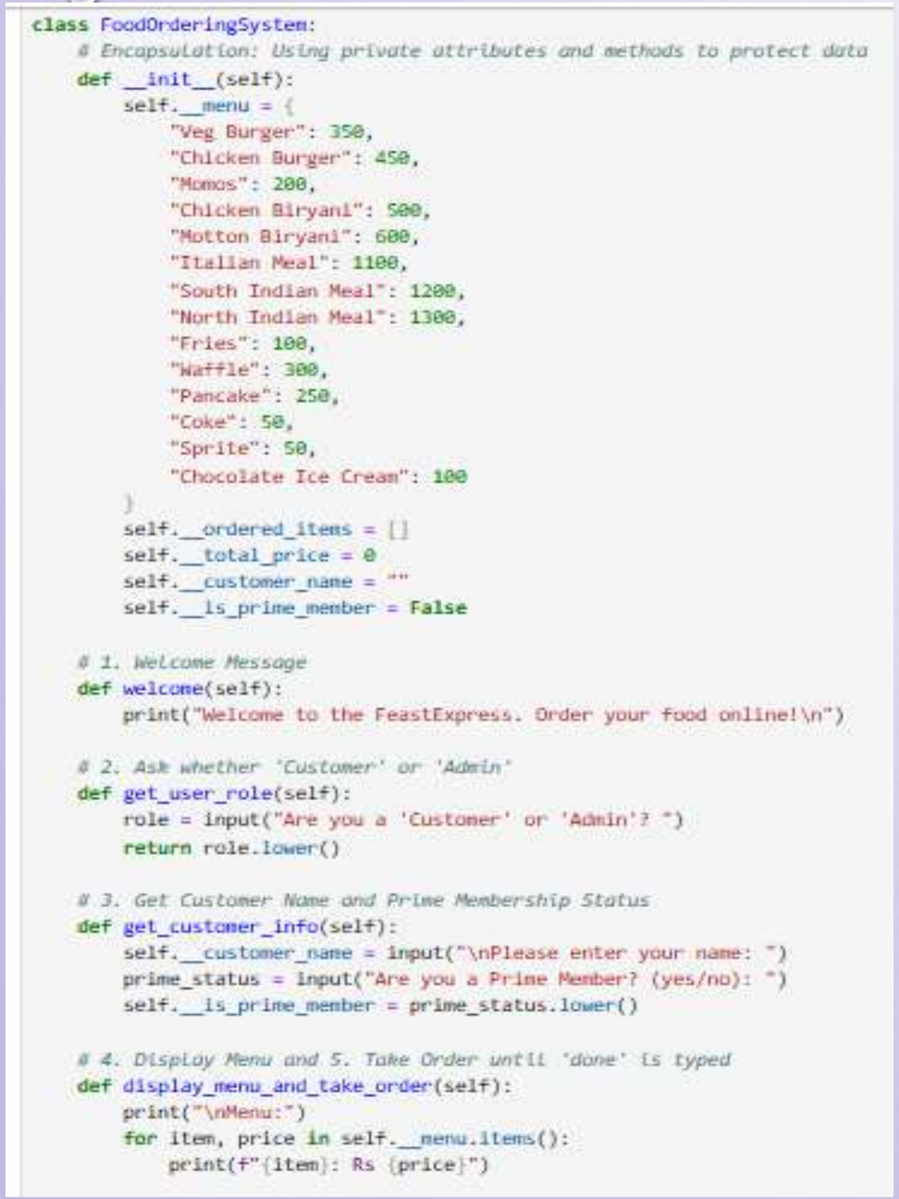
- **Food Item Selection:** Users input food items being converted to title case for consistency. Invalid entries prompt an error message, and the process continues until "**done**" is entered to finalize the order.

- **Prime Membership Status:** When prompted the user responds to "yes" or "no" option, the system checks for appropriate discounts based on total cost attributes.

Payment Method Input: Users select a payment method by entering a corresponding valid number, and the system defaults to Cash on Delivery (COD) if the entry is invalid.

Error Handling: The project handles common input errors with conditional checks but does not explicitly manage exceptions like invalid data types or unexpected inputs.

CODE IMPLEMENTATION



```python
class FoodOrderingSystem:
    # Encapsulation: Using private attributes and methods to protect data
    def __init__(self):
        self.__menu = {
            "Veg Burger": 350,
            "Chicken Burger": 450,
            "Momos": 200,
            "Chicken Biryani": 500,
            "Motton Biryani": 600,
            "Italian Meal": 1100,
            "South Indian Meal": 1200,
            "North Indian Meal": 1300,
            "Fries": 100,
            "Waffle": 300,
            "Pancake": 250,
            "Coke": 50,
            "Sprite": 50,
            "Chocolate Ice Cream": 100
        }
        self.__ordered_items = []
        self.__total_price = 0
        self.__customer_name = ""
        self.__is_prime_member = False

    # 1. Welcome Message
    def welcome(self):
        print("Welcome to the FeastExpress. Order your food online!\n")

    # 2. Ask whether 'Customer' or 'Admin'
    def get_user_role(self):
        role = input("Are you a 'Customer' or 'Admin'? ")
        return role.lower()

    # 3. Get Customer Name and Prime Membership Status
    def get_customer_info(self):
        self.__customer_name = input("\nPlease enter your name: ")
        prime_status = input("Are you a Prime Member? (yes/no): ")
        self.__is_prime_member = prime_status.lower()

    # 4. Display Menu and 5. Take Order until 'done' is typed
    def display_menu_and_take_order(self):
        print("\nMenu:")
        for item, price in self.__menu.items():
            print(f"{item}: Rs {price}")
```

- **Food Ordering System:**
  - Implements Encapsulation
  - Ensures Data Integrity
  - Protects Sensitive Information

- **Payment Method:**
  - An abstract base class which implements Abstraction Methodology
  - DebitCardPayment, CODPayment, UPIPayment
    - ❖ Implements Inheritance & Polymorphism

RESULTS & OUTCOMES

- **Results Achieved:**
  - Streamlined Ordering Process
  - Flexible Payment Options
  - Discount Application
  - Error Handling
- **Insights Gained:**
  - OOP Principles in Action
  - User-Centric Design

CONCLUSION

❑ Key Points:

- ***Automated Ordering:*** Simplifies food ordering process.
- ***Flexible Payments:*** Supports debit cards, cash and UPI payments.
- ***Discounts:*** Applies discounts for prime members and high-value orders.

❑ Significance:

- ***Efficiency:*** Streamlines the ordering process, improving user experience and operational efficiency.

❑ Future Developments:

- ***Integration with Payment Gateways:*** For real-world payment processing.
- ***Admin Features:*** To manage menu items and monitor orders.
- ***Enhanced Error Handling:*** For robust input validation and exception management.