Project development phase

Date	5 NOVEMBER 2023
Team ID	NM2023TMID02112
Project Name	Food tracking system
Maximum Marks	2 Marks

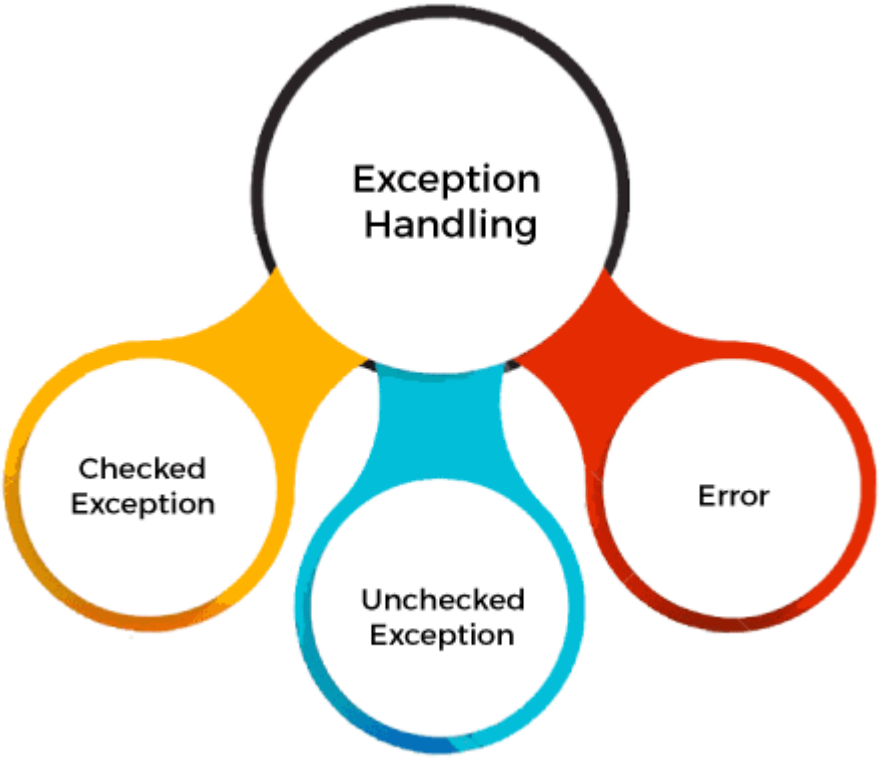
EXCEPTION HANDLING :

Exception handling in a food tracking system is crucial for ensuring smooth and error-free operation. It involves managing unexpected or exceptional situations that might occur during the execution of the software. Here are some key areas where exception handling can be applied in a food tracking system:

1. **Input Validation:** Validate user input to ensure that the data entered (e.g., food items, quantities) is in the correct format and within acceptable ranges. Handle cases where users might input incorrect data, such as non-numeric characters in a quantity field or invalid food names.
2. **Database Interactions:** Handle exceptions that might occur when retrieving or storing data in the system's database. This could involve dealing with connectivity issues, data integrity problems, or database query failures.
3. **External Service Integration:** When integrating with external services like nutritional databases or APIs for barcode scanning, implement exception handling to manage cases of unavailability, timeouts, or unexpected responses.
4. **File Handling:** If the system deals with file uploads (e.g., importing recipes, images of food items), handle exceptions related to file access, format errors, or corrupt files.
5. **Calculations and Algorithms:** Validate and handle errors during calculations of nutritional values, meal recommendations, or other algorithm-based functionalities within the system.
6. **Security and Authentication:** Handle exceptions related to authentication failures, unauthorized access attempts, or security breaches.
7. **User Interface:** Implement user-friendly error messages or notifications to guide users when unexpected errors occur, helping them understand what went wrong and how to proceed.
8. **Logging and Monitoring:** Set up proper logging mechanisms to capture exceptions, errors, and unexpected behaviors for analysis and debugging.
9. **Graceful Degradation:** Design the system to degrade gracefully in case of failures, providing basic functionalities even when certain features are temporarily unavailable.

10. **Testing for Exceptions:** Perform thorough testing, including boundary testing, negative testing, and stress testing, to ensure that the system can handle a variety of exceptional scenarios.

Proper exception handling is about gracefully managing errors, preventing crashes, maintaining data integrity, and providing a good user experience by informing users about issues while keeping the system running as smoothly as possible. It's essential to anticipate potential issues and handle them appropriately to ensure the reliability and robustness of the food tracking system.



EXCEPTION HANDLING PROGRAM :

```
class Food Tracking System:
  def log food item (self, item name, quantity):
    try:
      # Check if quantity is a positive number
      if quantity <= 0:
        raise Value Error ("Quantity must be a positive number.")

      # Check if the item name is not empty
```

```
            if not item name:
                raise Value Error ("Please provide a valid food item name.")

            # If both conditions are met, log the food item
            Self. save to database (item name, quantity)
            Print f ("Logged {quantity} of {item name} successfully.")
        except Value Error as e:
            print f ("Error logging food item: {e}")

    def save to database (self, item name, quantity):
        # Simulated database saving process
        # This could involve actual database interactions
        Print f ("Simulating saving {quantity} of {item name} to the database.")

# Example usage:
Tracking system = Food Tracking System ()

# Test cases with try-except blocks
Tracking system.log food item ("Apple", 2) # Valid input
Tracking system.log food item ("", 3) # Empty item name - will trigger an exception
Tracking system.log food item ("Pizza", -1) # Negative quantity - will trigger an exception
```