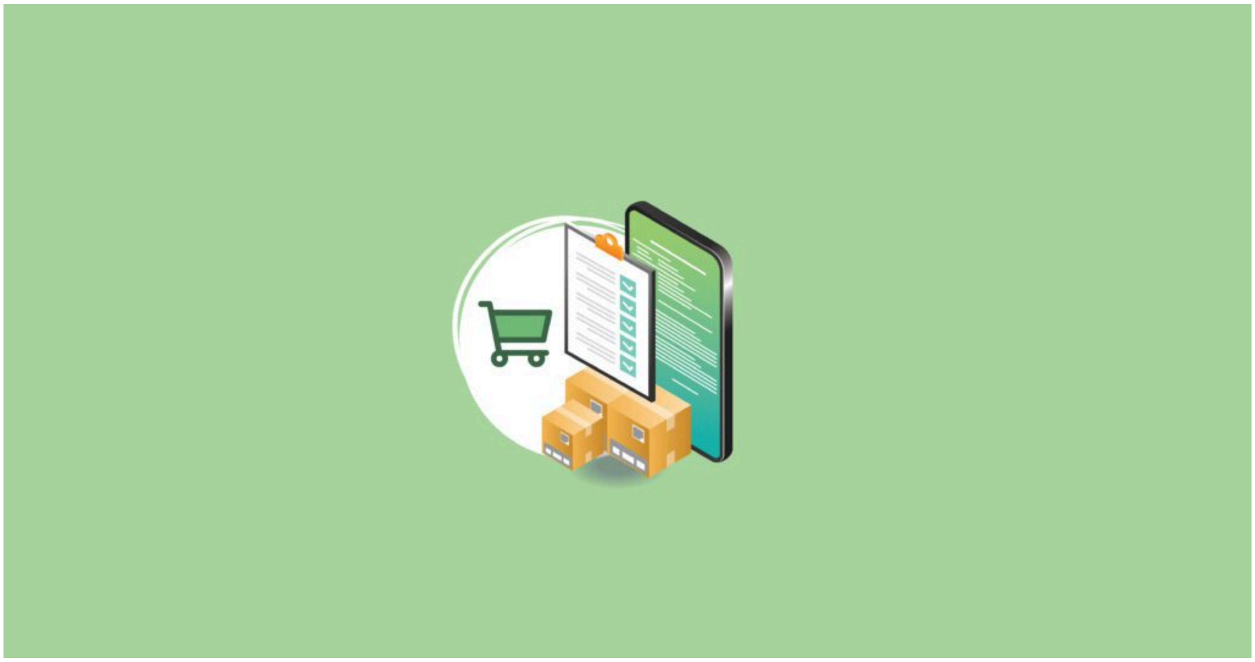

# ORDER SYSTEM

**PROJECT BY:**

Jidnyasa Patil
Ashutosh Kulkarni
Balraj Rasal
Anula Borse

# INTRODUCTION

An order system, also known as an Order Management System (OMS) is a software application that manages the order lifecycle from start to finish:

- Order entry: Tracks when an order is placed
- Inventory management: Tracks inventory levels
- Fulfillment: Tracks the status of an order as it's being fulfilled
- After-sales service: Tracks customer service interactions
- Payment collection: Tracks payment status
- Shipping: Tracks shipping status
- Invoicing: Tracks invoicing status

# ANALYSIS

1. We have written a code to accept the details of customer from user such as name, phone number and address.
2. The available items such as iPhone, Macbook and Airpods are displayed to the customer.
3. User has to enter any number from 1 to 3 to choose the item to purchase (where 1 is iPhone, 2 is Macbook and 3 is Airpods).
4. After choosing the item the user has to enter the quantity of that item to be purchased.
5. After entering the number, the user is given an option to again choose the item to be purchased and then the amount.
6. After entering the total number of items to be purchased, the code ends and the bill is displayed to the user in proper format.

# ALGORITHM

1. **Define Structures**:
    - Item Structure:
        - Fields: name (string, max length = `MAX_NAME_LENGTH`)`price` (float)
    - Customer Structure:
        - Fields:
            - name (string, max length = `MAX_NAME_LENGTH`)
            - phone (string, max length = `MAX_PHONE_LENGTH`)
            - `address` (string, max length = `MAX_ADDRESS_LENGTH`)
2. **Function: `printBill(Customer customer, Item items[], int quantities[], int itemCount)`**:
    - Initialize `total` to 0.0.
    - Print the header for the shopping bill:
        - Display customer name, phone, and address.
        - Print table headers: "Item", "Quantity", "Price (Rs)", "Total (Rs)".
    - Loop through each item (from 0 to `itemCount - 1`):
        - Calculate `itemTotal` as `items[i].price * quantities[i]`.
        - Add `itemTotal` to `total`.
        - Print item details (name, quantity, price, itemTotal).
    - Print the total amount.
    - End the function.
3. **Main Function**:
    - Declare a variable `customer` of type `Customer`.
    - Declare an array `items` of type `Item` with size `MAX_ITEMS` and initialize it with three items (name and price).
    - Declare an array `quantities` of integers with size `MAX_ITEMS` and initialize all elements to 0.
    - Initialize `itemCount` to 0.
4. **Get Customer Details**:
    - Prompt the user for `customer.name` and read input.
    - Prompt for `customer.phone` and read input.
    - Prompt for `customer.address` and read input.
5. **Show Available Items**:
    - Print the list of available items along with their prices.
6. **Input Items to Buy**:
    - Loop through up to `MAX_ITEMS` (3 iterations):
        - Prompt the user to choose an item (1-3, or 0 to finish).
        - If the user inputs `0`, exit the loop.

    - If the input is invalid (not between 1 and 3):
        - Print an error message and decrement `i` to repeat the current iteration.
    - Otherwise:
        - Prompt for the quantity of the selected item.
        - Store the quantity in the `quantities` array at the index corresponding to the selected item.
        - Increment `itemCount`.

7. **Print the Bill**:
    - Call the `printBill` function with `customer`, `items`, `quantities`, and `itemCount` to display the total bill.
8. **End of Program**.

# CODE ACCESS

- https://github.com/Jidnyasa-P/Order-System.git
- ORDER SYSTEM