Item Class:

Java

```
class Item {
    private String name;
    private double unitPrice;
    private int quantity;

    // Constructor
    public Item(String name, double unitPrice, int quantity) {
        this.name = name;
        this.unitPrice = unitPrice;
        this.quantity
 = quantity;
    }

    // Getters and setters
    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public
 double getUnitPrice() {
        return unitPrice;
    }

    public void setUnitPrice(double unitPrice) {
        this.unitPrice = unitPrice;
    }

    public int getQuantity() {
        return quantity;
    }

    public void setQuantity(int quantity) {
        this.quantity = quantity;

    }
}
```

Use code with caution.

User Class:

Java

```
class User {
    private String name;
    private String contactNumber;
    private ArrayList<Item> cart;

    // Constructor
    public User(String name, String contactNumber) {
        this.name = name;
        this.contactNumber = contactNumber;
        this.cart = new ArrayList<>();
    }

    // Getters and setters
    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String
 getContactNumber() {
        return contactNumber;
    }

    public void setContactNumber(String contactNumber) {
        this.contactNumber = contactNumber;
    }

    public ArrayList<Item>
 getCart() {
        return cart;
    }

    public void setCart(ArrayList<Item> cart) {
        this.cart = cart;
    }

    // Methods for cart operations
    public void addItemToCart(Item item) {
        cart.add(item);
    }

    public void removeItemFromCart(Item item) {
        cart.remove(item);
    }

    public double calculateBill() {
        double total = 0;
        for (Item item : cart) {
            total += item.getUnitPrice() * item.getQuantity();
        }
        return total;
    }

    // Method for payment
    public void pay(String paymentMethod, double amountPaid) {
        double billAmount = calculateBill();

        if (paymentMethod.equalsIgnoreCase("cash")) {
            double change = billAmount - amountPaid;
            System.out.println("Change: " + change);
        } else if (paymentMethod.equalsIgnoreCase("credit card")) {
            String cardNumber = ""; // Get card number from user
            if (cardNumber.length() == 10) {
                System.out.println("Payment successful.");
            } else {
                System.out.println("Invalid card number.");
            }
        } else {
            System.out.println("Invalid payment method.");
        }

        // Clear the cart after successful payment
        cart.clear();
    }
}
```

Use code with caution.

Main Class (Example Usage):

Java

```
public class Main {
    public static void main(String[] args) {
        User user = new User("John Doe", "1234567890");

        // Add items to cart
        user.addItemToCart(new Item("Apple", 2.0, 5));
        user.addItemToCart(new Item("Banana", 1.5, 3));

        // View bill
        double billAmount = user.calculateBill();
        System.out.println("Total bill: " + billAmount);

        // Pay with cash
        user.pay("cash", 15.0);

        // Or pay with credit card
        // user.pay("credit card", billAmount);
    }
}
```

Use code with caution.

This implementation provides a basic online grocery shop system with features like adding items to the cart, removing items, calculating the bill, and handling cash and credit card payments. You can extend this by adding more features like user login/registration, order tracking, inventory management, and more.