School of Science and Technology

B.Sc. in Computer Science and Engineering

Lab Report : 03

Designing a Complete Class Diagram of an Online Order Processing System

Submitted By	Submitted To
Name: **Md. Zubaer Ahammed** Student ID: 20-0-52-801-006 Course Title: Database Management System Lab Course Code: CSE22P9	**Mr. Samrat Kumar Dey** Lecturer (Computer Science), School of Science and Technology Bangladesh Open University Gazipur-1705
Date of Submission: 27 Mar 2024	

Objective:

The objective of this report is to design a comprehensive class diagram for an online order processing system. This diagram will represent the structure of the system, including its various components, their relationships, and interactions.

Theory:

An online order processing system facilitates the management of orders placed by customers on a digital platform. It involves various entities such as customers, items, orders, payment methods, and the system itself. Class diagrams are a fundamental tool in object-oriented design, providing a visual representation of the system's structure through classes, attributes, and relationships.

Required Software:

For designing the flowchart of the General Problem Solution Approach, EdrawMax or any equivalent software capable of creating flowcharts can be used.

Procedures:

Identify Entities: Begin by identifying the main entities involved in the online order processing system. These typically include Customers, Items, Orders, Payments, and possibly others depending on the specific requirements.

Define Classes: For each identified entity, create a class representing it in the system. Determine the attributes and methods associated with each class. For example:

- **Customer:** Attributes - customerID, name, address, phone, email; Methods - editInfo(), login(), placeOrder()
- **Order:** Attributes - dateReceived, status; Methods - calculateSubTotal(), calculateTax(), calculateTotal(), calculateTotalWeight(), generateInvoice()
- **OrderDetail:** Attributes - orderID, quantity, taxStatus; Methods - calculateSubTotal(), calculateWeight(), calculateTax()
- **Item**: Attributes - weight, description; Methods - getPriceQuantity(), inStock(), getTax()
- **Payment**: Attributes - paymentID, amount
- **Credit**: Attributes - number, type, expirationDate; Methods - authorize()
- **Cash**: Attributes - cashPaid

- **Check**: Attributes - name, bankID; Methods - authorize()
- **CashOnDelivery**: Attributes - customerName, customerAddress, amount

Establish Relationships: Determine the relationships between the classes. Use appropriate notations such as association, aggregation, or composition to represent these relationships. For instance:

- Customer places Orders (association)
- Order contains OrderDetail and Items (aggregation)
- Order is associated with Payment (association)

Refine Class Structure: Review the initial class structure and relationships. Make necessary adjustments to ensure clarity, coherence, and adherence to system requirements.

Create the Class Diagram: Utilize eDrawMax or equivalent software to construct the class diagram based on the defined classes, attributes, and relationships. Arrange the classes in a logical manner, ensuring readability and comprehensibility.

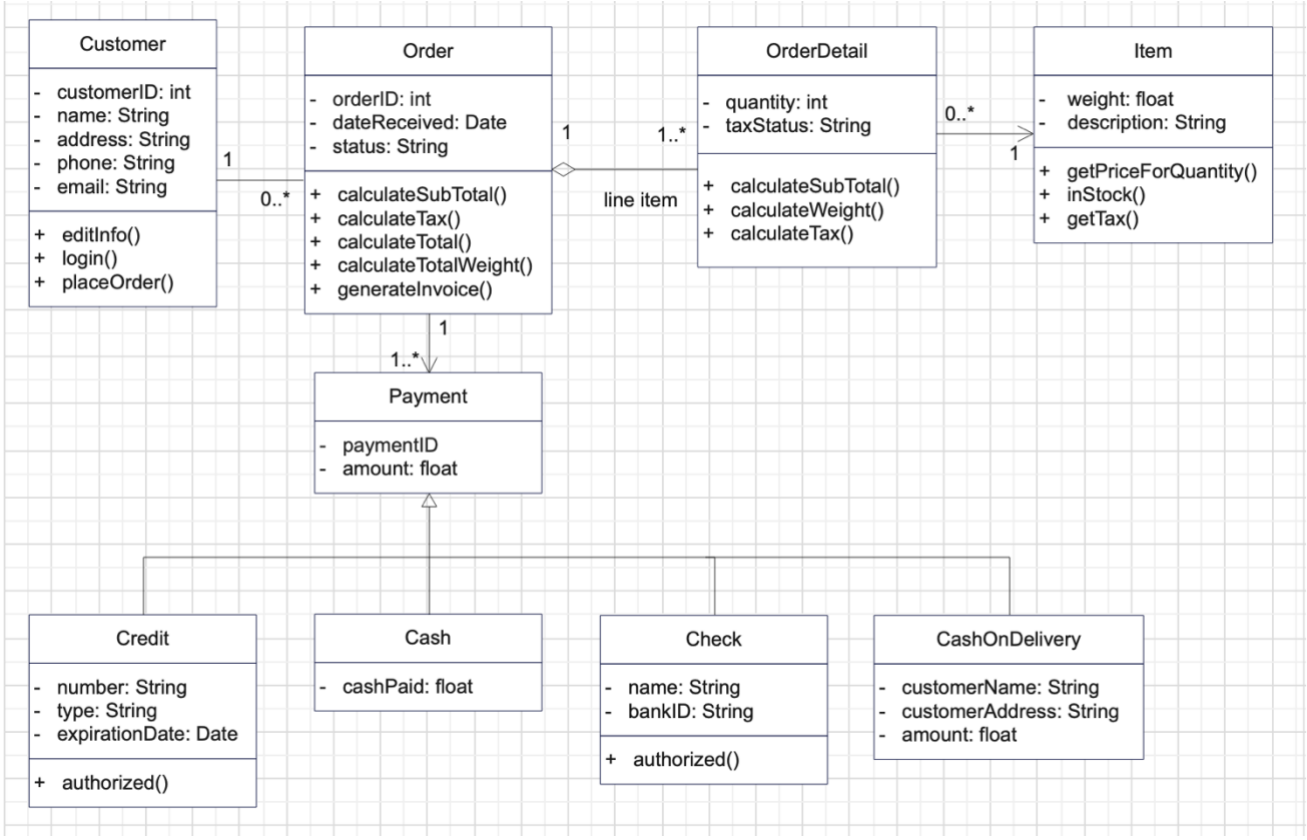

Figure: UML Class Diagram of an Online Order Processing System

Conclusion:

In conclusion, designing a UML Use Case diagram of the General Problem Solution Approach provides a structured framework for effectively addressing and solving problems. By following the defined steps, individuals and organizations can streamline their problem-solving process, leading to more efficient and successful outcomes. Utilizing software such as EdrawMax facilitates the visualization and documentation of the problem-solving process, enabling clear communication and collaboration among stakeholders. Overall, the flowchart serves as a valuable tool for guiding problem-solving efforts and driving continuous improvement.