**Day 2 – OOAD & Design**

1. **Class Diagram** (representing classes like Book, User, Cart, Order).
2. **Use Case Diagram** (depicting how different users interact with the system).
3. **ER Diagram** (depicting MySQL tables like Book, User, Cart, and Orders).

**1. Class Diagram**

In the **Student Course Registration System**, we can have the following classes:

**Classes and Attributes:**

* **Book:**
  + bookID (int)
  + title (String)
  + author (String)
  + price (double)
  + description (String)
  + genre (String)
  + stockStatus (boolean) — True if available, False if out of stock.
* **User:**
  + userID (int)
  + name (String)
  + email (String)
  + password (String, hashed)
  + role (String) — e.g., "student", "faculty", "admin"
  + address (String)
* **Cart:**
  + cartID (int)
  + userID (int)
  + totalPrice (double)
  + **Methods:**
    - addItem(Book book)
    - removeItem(Book book)
    - calculateTotal()
* **Order:**
  + orderID (int)
  + userID (int)
  + orderDate (Date)
  + totalPrice (double)
  + shippingAddress (String)
  + paymentStatus (String) — e.g., "paid", "pending"

**Class Diagram (Visual Representation):**

Let me create the class diagram for you. Here's the structure:

1. **User** can have a one-to-many relationship with both **Cart** and **Order** (i.e., a user can have multiple carts and orders).
2. **Cart** contains multiple **Books** (many-to-many), and the **Cart** has methods for adding/removing items.
3. **Order** is associated with a **User** and contains details about the books they purchased.

Now let me generate the diagram.

**2. Use Case Diagram**

Here’s a typical breakdown for the **Student Course Registration System**.

**Actors:**

* **Student**
* **Faculty**
* **Administrator**

**Use Cases:**

1. **Student:**
   * Search Courses
   * View Course Details
   * Register for Courses
   * Drop Courses
   * View Registered Courses
2. **Faculty:**
   * View Enrolled Students
   * Modify Course Information
3. **Administrator:**
   * Add/Modify/Delete Courses
   * Manage Users (Student/Faculty)

Here is a quick description of the interactions:

* **Student** searches for courses, views details, and registers for courses.
* **Faculty** can see who is enrolled in their courses and modify course details.
* **Administrator** manages users and courses.

**3. ER Diagram**

The **ER Diagram** shows how the database tables relate to each other.

**Tables & Attributes:**

1. **Users Table:**
   * userID (PK)
   * name
   * email
   * password
   * role (student, faculty, admin)
   * address
2. **Books Table:**
   * bookID (PK)
   * title
   * author
   * price
   * description
   * genre
   * stockStatus
3. **Cart Table:**
   * cartID (PK)
   * userID (FK to Users)
   * totalPrice
4. **CartItems Table** (many-to-many between Cart and Books):
   * cartID (FK)
   * bookID (FK)
   * quantity
5. **Orders Table:**
   * orderID (PK)
   * userID (FK to Users)
   * orderDate
   * totalPrice
   * shippingAddress
   * paymentStatus

**ER Diagram Design:**

* **Users** table is linked to both **Cart** and **Orders** tables via userID.
* **Cart** and **Books** are linked in a many-to-many relationship through **CartItems**.
* **Orders** stores completed transactions with **Users**.

**Deliverables**

* **Class Diagram** (Uploaded to project repo).
* **Use Case Diagram** (Uploaded to project repo).
* **ER Diagram** (Uploaded to project repo).