e-commerce class diagram:

⧉ Key classes and their attributes:

1. User: Represent a customer or administrator interacting with the system.

- Attributes:

  * userID: Unique identifier for the user.
  * username: User's login name.
  * password: User's passwords
  * email: User's email address.

- Methods:

  * register( ): Allows a new user to create an account.
  * login ( ): Authenticates a user.
  * update Profile ( ): Updates user information.

2. Product: Represents an item available for purchase.

- Attributes:

  * ProductID: ~~Unin~~ Unique identifier for the product.

  * name: Name of the product.

  * description: Detailed information about the product.

  * price: cost of the product.

  * stockQuantity: Number of iteam available.

- Methods:

  * updateStock(): Modify the stock quantity

  * applyDiscount(): Applies a discount to the product price.

3. Category: Organizes products into groups.

- Attributes:

  * cetegoryID: Unique identifier for the cetegory.

*name: Name of the cetegory.

* description: Description of the cetegory.

- Methods:

  * addProduct(); Adds a products to the cetegory.

  * removeProduct(): Removes a product from the cetegory.

4. Shopping cart: Holds products a user intends to purchase.

- Attributes:

  * cartID: Unique identifier for the cart.

  * userID: Identifier linking the cart to a user.

- Methods:

  *addIteam(): Adds a product to the cart.

  * removeIteam(): Removes a product from the cart.

* calculateTotal( ): compute the total cost of iteams in the cart.

5. Order: Records completed purchases.

- Attributes:

  * orderID: Unique identifier for the order
  * userID: Identifier linking the order to a user.
  * OrderDate: Date when the order was placed.
  * status: Current status of the order (e.g, pending, shipped).

- Methods:

  * placeOrder( ): finalizes the order.
  * cancelOrder( ): cancels the order.

6. Payment: Manages .payment information for orders.

- Attributes:

  * paymentID: Unique identifier for the payment.

  * orderID: Identifier linking the payment to an order.

  *amount: Total amount paid.

  * paymentMethod: Method used for payment (e.g: credit card)

  * paymentStatus: Status of the payment (e.g: completed, pending)

- Methods:

  * ~~Pro~~ processPayment(): Processes the payment for an order.

7. Shipping: Handles delivery details for orders.

- Attributes:

  * ~~Sh~~ shippingID: Unique identifier for the shipping records.
  * orderID: Identifier linking the shipping to an order.
  * shippingAddress: Delivery address.
  * shippingStatus: Status of the shipment.

- Methods:

  * shipOrder(): Initiates the shipping process.

8. Admin: Manages the platform. ~~Ind~~ Including the users and products.

- Attributes:

  * adminID: Unique identifier for the administrator.

  * username: Admin's login name.

  * password: Admin's password.

- Methods:

  * login( ): Authenticates the admin.

  * manageUser ( ): Manages users accounts

  * manageProduct ( ): Manages product listings.

Relationships:

  * User to Shopping Cart: A User has one Shopping Cart.

  * User to Order: A user can place multiple Orders.

* Order to Product : An Order can include multiple ~~Orders~~. Products and a Product can be a part of multiple Orders (many-to-many relationship)

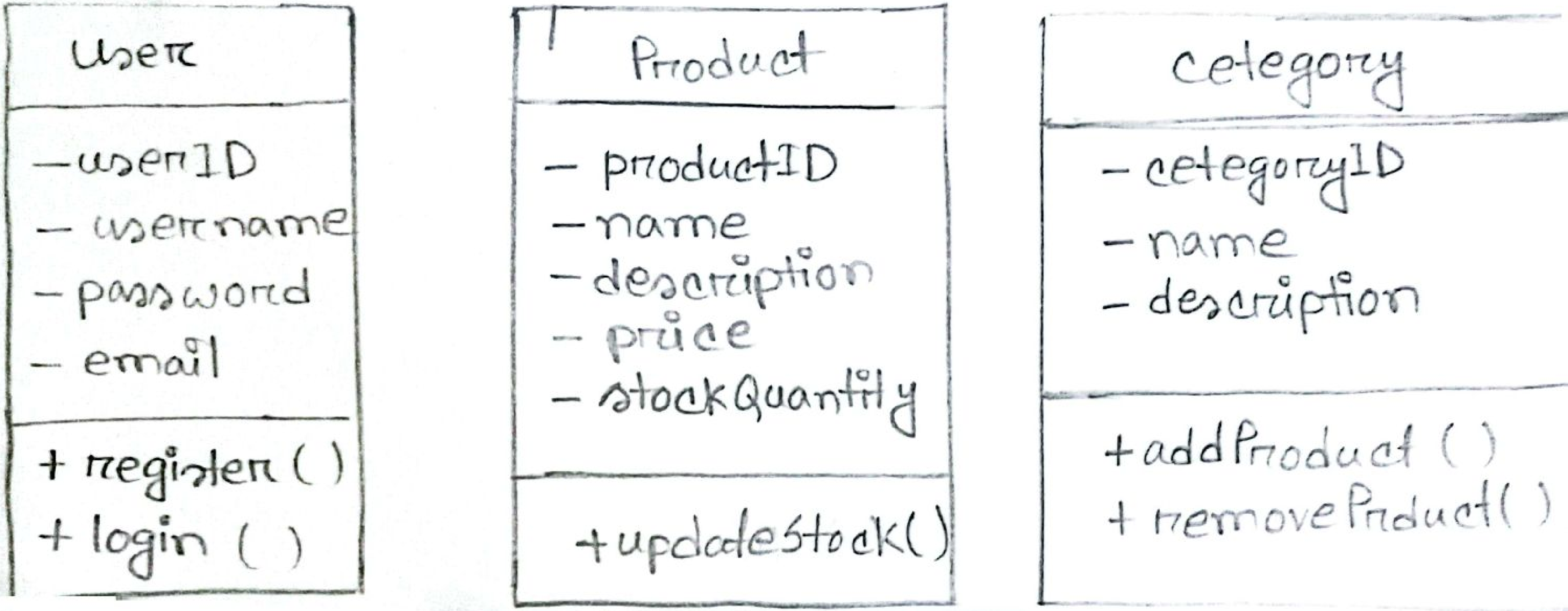
* Shopping Cart to Product : A shopping cart can contain multiple products, and a product can be in multiple Shopping Carts (many-to-many relationship)

* Order to Payment : An order has one payment.

* Order to Shipping : An order has one Shipping record.

* Admin to user/ Product / Order : An Admin can manage multiple Users, Products, or Orders.

UML class diagram:


User
- userID
- username
- password
- email
+ register ( )
+ login ( )
Product
- productID
- name
- description
- price
- stockQuantity
+ updateStock( )
cetegory
- cetegoryID
- name
- description
+ addProduct ( )
+ removePrduct( )