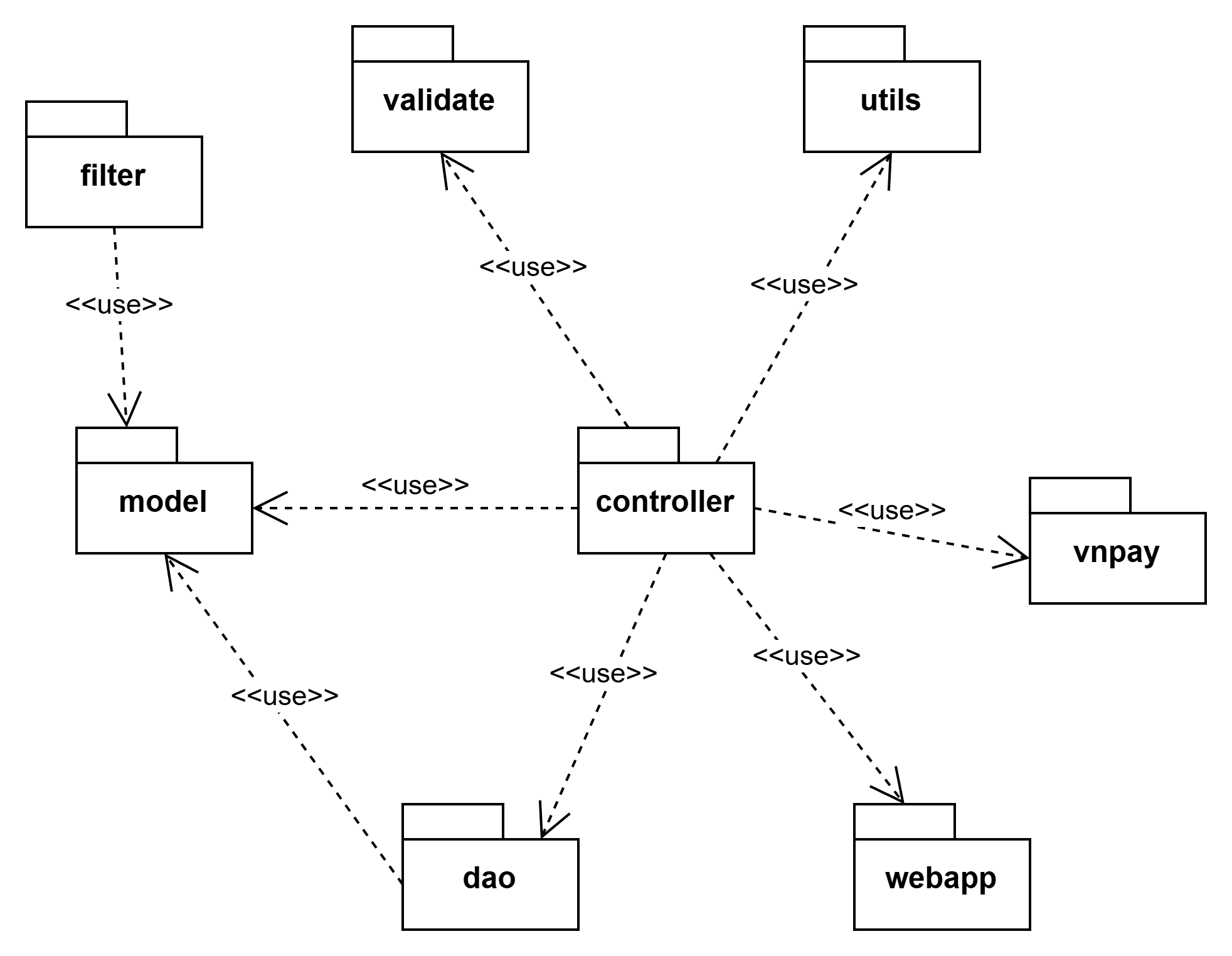
# **I. Overview**

## **1. Code Packages**



***Package descriptions***

| **No** | **Package** | **Description** |
| --- | --- | --- |
| *01* | controller | A controller package, in the context of software development, refers to a component or module within an application or system that plays a crucial role in managing and directing the flow of data and interactions between various parts of the software. |
| *02* | util | A util package is a collection of reusable, general-purpose functions, classes, or modules that provide common functionalities and tools that can be used across different parts of an application or system. These utilities are typically designed to simplify coding tasks, improve code organisation, and promote code reusability. |
| *03* | dao | A DAO package is responsible for abstracting and encapsulating the methods and operations necessary to access, manipulate, and interact with data. |
| *04* | model | A model package typically refers to a component or module within an application that is responsible for defining and encapsulating the data structures, business logic, and rules that represent the core functionality and behaviour of the application. |
| *05* | webapp | A "web package" is a broad term that can refer to various components, modules, or directories within a web application or web-based project. The contents and purpose of a web package can vary depending on the specific framework, technology stack, or project structure being used. |
| *06* | vnpay | Package for configuration params for processing payment via vn pay api. |
| *07* | filter | Package contain the filter for user authentication |
| *08* | validate | Package contain the validation method for ensuring input data match the right format of system |

## **2. Database Design**

### ***a. Database Schema***

### ***b. Table Description***

| **No** | **Table** | **Description** |
| --- | --- | --- |
| 01 | Users | This table stores information about users registered on the platform, including personal details like first and last names, email, phone number, address, and authentication details. It also tracks whether the user is active and links them to a specific role. |
| 02 | Roles | This table stores the different roles available in the system (e.g., Admin, Customer, Supplier). Each user is assigned a role which governs their access and permissions. |
| 03 | Suppliers | This table holds information about suppliers who provide products for sale or rent. Each supplier is linked to a user, as every supplier must have a registered account. |
| 04 | Products | This table contains details about the products available for sale or rental on the platform. Products are linked to categories and suppliers. |
| 05 | Categories | This table categorises products for easier search and organisation, such as "Educational Toys", "Action Figures", etc. |
| 07 | Orders | This table records customer orders that result from completed requests. It tracks the total amount, order status, and the date the order was placed. |
| 08 | OrderItems | This table contains information about the individual items in each order, including the product details, quantity, and pricing. |
| 09 | Transactions | This table records payment transactions related to orders, including the payment status and the amount paid. |
| 10 | Reviews | This table stores customer reviews of products, including the rating and review text. |
| 11 | ShoppingCart | This table represents the shopping cart of a user, allowing the user to add or remove items before finalising the order. |
| 12 | CartItems | This table contains information about the items within a shopping cart, including product details and quantity. |
| 13 | Inventory | This table manages stock levels for products, tracking the available quantity for sale and rental. |
| 14 | WishLists | This table represents a user's wishlist, where they can save products they are interested in but are not ready to purchase yet |
| 15 | WishlistItems | This table stores items added to a user's wishlist, allowing them to track items they are interested in |

# **II. Code Designs**

## **1. Register User**

### **AuthController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | register() | Handles user registration requests and forwards them to the service layer. |

### **AuthService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | register() | Executes the registration logic and stores user data. |

### **UsersRepository Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | save() | Saves the new user's data into the database. |

### **CreateUserDto Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | CreateUserDto | Defines the required data structure for user registration. |

### **AuthModule Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | AuthModule | Registers authentication components, including registration services. |

## **2. Login**

### **AuthController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | login() | Handles user login requests. |

### **AuthService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | validateUser() | Validates user credentials. |
| 2 | generateJwtToken() | Generates a JWT token upon login. |

### **UsersRepository Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | findByEmail() | Finds a user by email from the database. |

### **LoginDto Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | LoginDto | Defines the data structure for login. |

### **AuthModule Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | AuthModule | Registers login-related components. |

## **3. Logout**

### **AuthController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | logout() | Logs out the user from the current session. |

### **AuthService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | logout() | Executes the logout process. |

### **AuthModule Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | AuthModule | Includes the logout functionality. |

## **4. Create Guest Order**

### **OrderController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | createGuestOrder() | Creates a new order without user registration. |

### **OrderService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | createOrder() | Handles order creation logic. |

### **OrderRepository Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | save() | Saves the order in the database. |

### **OrderModule Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | OrderModule | Registers the order management components. |

## **5. Update Order Status**

### **OrderController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | updateStatus() | Updates the status of an existing order. |

### **OrderService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | updateOrderStatus() | Executes status updates for orders. |

### **OrderRepository Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | update() | Updates the order status in the database. |

## **6. Cancel Expired Orders**

### **OrderController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | cancelExpiredOrders() | Cancels orders that are past their expiry date. |

### **OrderService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | cancelOrders() | Performs the logic to cancel expired orders. |

## **7. Get All Users**

### **UsersController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | findAll() | Retrieves all users from the system. |

### **UsersService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | findAllUsers() | Handles the retrieval of all user data. |

## **8. Search User by Name**

### **UsersController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | searchByName() | Searches for a user by name. |

### **UsersService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | findByName() | Implements the logic to search for users by name. |

## **9. View Profile**

### **UsersController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | viewProfile() | Displays the profile information of the current user. |

### **UsersService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | getProfile() | Retrieves the profile details of the user. |

## **10. Update Profile**

### **UsersController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | updateProfile() | Updates the user profile with new information. |

### **UsersService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | updateUserProfile() | Executes the logic for updating the profile. |

## **11. Upgrade to Supplier**

### **UsersController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | upgradeToSupplier() | Upgrades the user account to a supplier role. |

### **UsersService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | promoteToSupplier() | Handles the upgrade logic to supplier. |

## **12. Downgrade to User**

### **UsersController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | downgradeToUser() | Downgrades the supplier account back to a user role. |

### **UsersService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | demoteToUser() | Handles the downgrade logic to regular user. |

## **13. Remove User**

### **UsersController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | remove() | Removes a user from the system. |

### **UsersService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | deleteUser() | Executes the logic to delete a user. |

## **14. Create Product**

### **ProductsController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | createProduct() | Creates a new product in the system. |

### **ProductsService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | addProduct() | Handles product creation logic. |

## **15. Get All Products**

### **ProductsController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | findAll() | Retrieves all products available in the system. |

### **ProductsService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | getAllProducts() | Fetches the list of all products. |

## **16. Find Product by ID**

### **ProductsController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | findOne() | Finds a specific product by its ID. |

### **ProductsService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | getProductById() | Retrieves product details using the product ID. |

## **17. Update Product**

### **ProductsController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | updateProduct() | Updates the details of an existing product. |

### **ProductsService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | modifyProduct() | Handles the logic to update product details. |

## **18. Remove Product**

### **ProductsController Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | removeProduct() | Removes a product from the system. |

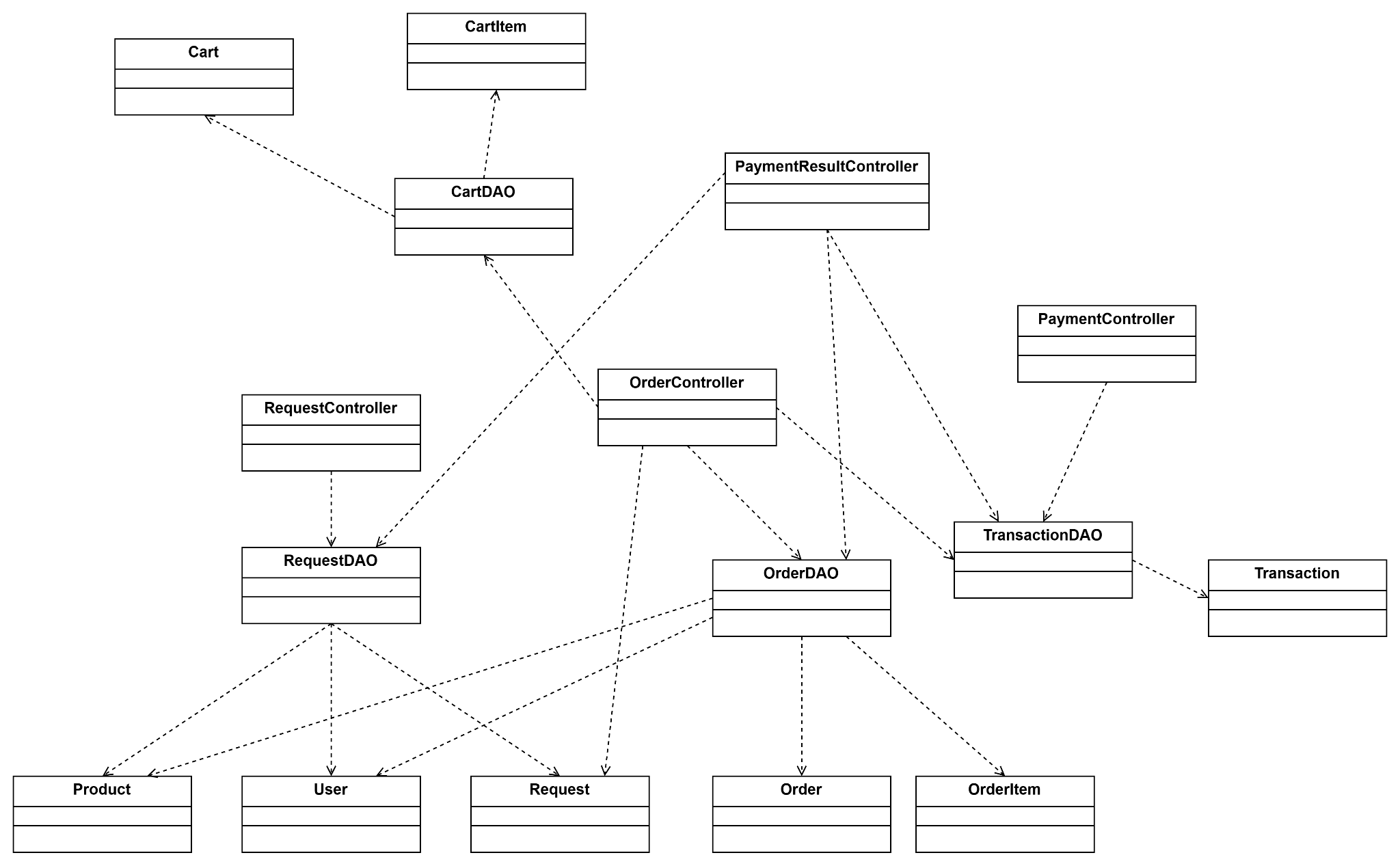
### **ProductsService Class**

| No | Method | Description |
| --- | --- | --- |
| 1 | deleteProduct() | Executes the logic to delete a product. |

## **19.** **Rent Toy**

Customers can request to rent educational toys, specifying the rental period (1 day, 1 week, or 2 weeks) and corresponding rental prices.

### ***a. Class Diagram***



### ***b. Class Specifications***

#### ***PaymentController Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | doPost | This method is used to request to the 3rd party for payment processing |

#### 

#### ***PaymentResultController Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | doPost | This method is used to update transaction status based on result returned from payment process |

#### 

#### ***RequestController Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | doPost | This method is used to get the request for renting toys from the user and perform relation action. |

#### 

#### ***OrderController Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | doPost | This method is used to get the order request from the user and perform relation action |

#### ***CartController Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | doPost | This method is used to perform cart manage action |

#### 

#### ***CartDAO Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | addToCart | This method is used to create a cart for an user. |
| 02 | updateCartItemQuantity | This method is used to update cart item quantity based on the cart id and cart item id. |
| 03 | removeFromCart | This method is used to remove cart items based on the cart id and cart item id. |

#### ***Cart Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of a cart object’s attributes. |
| 02 | Setter | These methods are used to set the values of a cart object’s attributes. |

#### ***CartItem Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of a cartItem object’s attributes. |
| 02 | Setter | These methods are used to set the values of a cartItem object’s attributes. |

#### 

#### ***RequestDAO Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | createRequest | This method is used to create row data of request and return the inserted id. |
| 02 | updateRequest | This method is used to update status data of request by request ID. |

#### 

#### ***OrderDAO Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | createOrder | This method is used to create row data of order and return the inserted id. |
| 02 | insertOrderItem | This method takes userID, list product, orderID and requestID as input and is used to insert all items of an order. |
| 03 | updateOrderStatus | This method is used to update order status based on the validation request of the payment process |

#### 

#### ***TransactionDAO Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | createTransaction | This method is used to create row data of transactions and return the id inserted. |
| 02 | updateTransaction | This method is used to update transaction status based on the validation request of the payment process. |

#### 

#### ***Product Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of product object’s attributes. |
| 02 | Setter | These methods are used to set the values of product object’s attributes. |

#### ***Order Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of the order object’s attributes. |
| 02 | Setter | These methods are used to set the values of order object’s attributes. |

#### 

#### ***OrderItem Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of order item object’s attributes. |
| 02 | Setter | These methods are used to set the values of order item object’s attributes. |

#### 

#### ***Transaction Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of a transaction object’s attributes. |
| 02 | Setter | These methods are used to set the values of a transaction object’s attributes. |

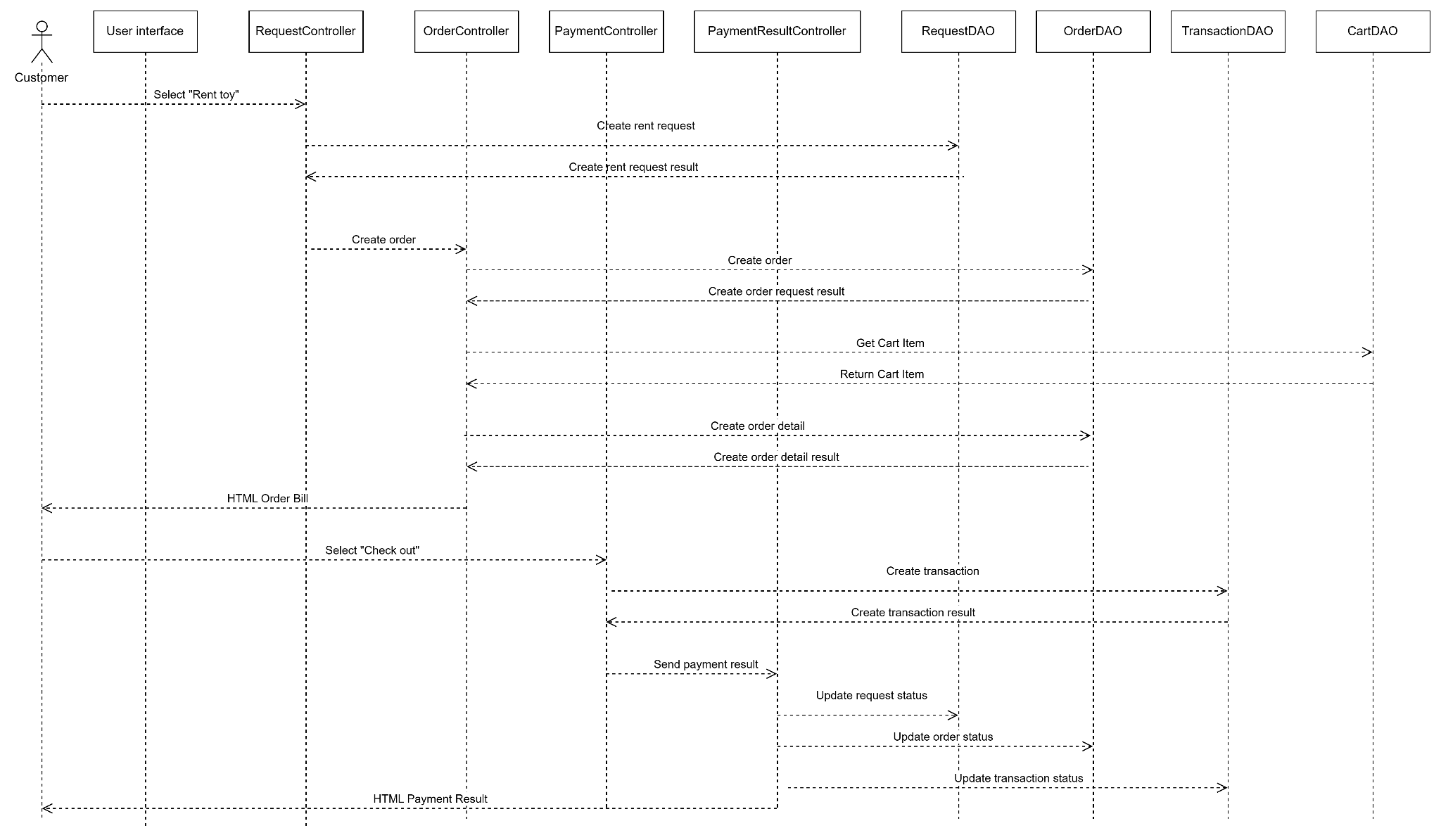
#### ***Request Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of the request object’s attributes. |
| 02 | Setter | These methods are used to set the values of request object’s attributes. |

#### ***User Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of the user object’s attributes. |
| 02 | Setter | These methods are used to set the values of the user object’s attributes. |

### ***c. Sequence Diagram(s)***



### ***d. Database Queries***

SELECT i.quantityInStock, i.availableForRental

FROM Inventory i

WHERE i.productId = 'TOY\_ID';

INSERT INTO Requests (userId, productId, requestType, quantity, rentalDuration, requestDate, status, totalAmount, createdDate)

VALUES ('USER\_ID', 'TOY\_ID', 'Rental', 1, 'RENTAL\_DURATION', CURRENT\_TIMESTAMP, 'Pending', 'TOTAL\_AMOUNT', CURRENT\_TIMESTAMP);

UPDATE Requests

SET status = 'NEW\_STATUS', updatedDate = CURRENT\_TIMESTAMP

WHERE id = 'REQUEST\_ID';

INSERT INTO Orders (userId, requestId, orderDate, totalAmount, status, createdDate)

VALUES ('USER\_ID', 'REQUEST\_ID', CURRENT\_TIMESTAMP, 'TOTAL\_AMOUNT', 'Pending', CURRENT\_TIMESTAMP);

UPDATE Order

SET status = 'Completed'

WHERE id = 'Order\_ID';

INSERT INTO OrderItems (id, orderId, productId, quantity, unitPrice, totalPrice, rentalDuration, rentalReturnDate)

VALUES ('ORDER\_ITEM\_ID', 'ORDER\_ID', 'PRODUCT\_ID', 2, 15.99, 31.98, 7, NULL);

INSERT INTO Transactions (id, orderId, paymentDate, amount, status)

VALUES ('TRANSACTION\_ID', 'ORDER\_ID', CURRENT\_TIMESTAMP, 59.99, 'Completed');

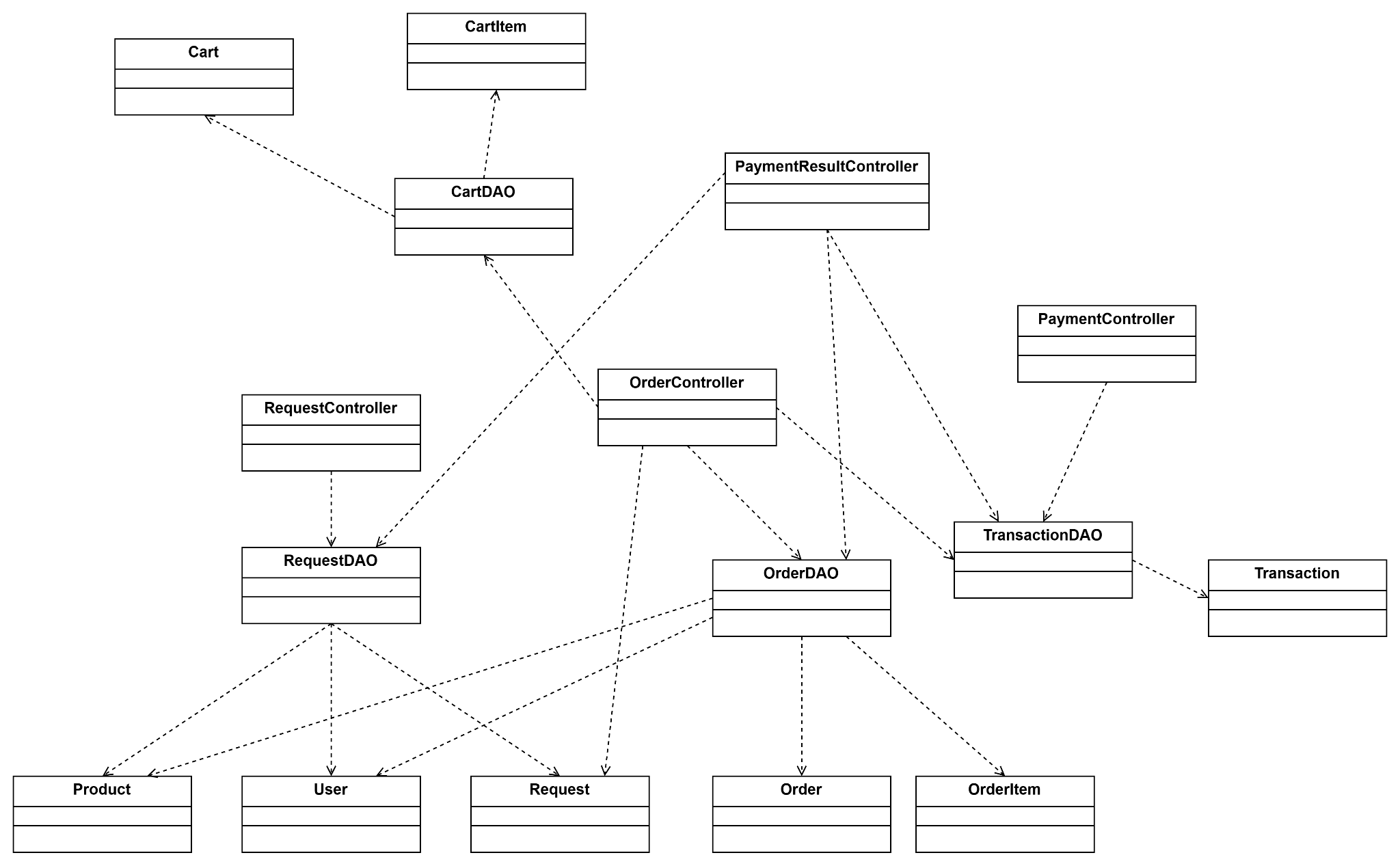
UPDATE Transactions

SET status = 'Completed'

WHERE id = 'TRANSACTION\_ID';

## **20.** **Buy Toy**

### **a. Class Diagram**



### **b. Class Specifications**

#### ***RequestController Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | doPost | This method is used to get the request for renting toys from the user and perform relation action. |

#### ***CartController Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | doPost | This method is used to perform cart manage action |

#### 

#### ***CartDAO Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | addToCart | This method is used to create a cart for an user. |
| 02 | updateCartItemQuantity | This method is used to update cart item quantity based on the cart id and cart item id. |
| 03 | removeFromCart | This method is used to remove cart items based on the cart id and cart item id. |

#### ***Cart Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of a cart object’s attributes. |
| 02 | Setter | These methods are used to set the values of a cart object’s attributes. |

#### ***CartItem Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of a cartItem object’s attributes. |
| 02 | Setter | These methods are used to set the values of a cartItem object’s attributes. |

#### 

#### ***OrderController Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | doPost | This method is used to get the order request from the user and perform relation action |

#### ***PaymentController Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | doPost | This method is used to request to the 3rd party for payment processing |

#### 

#### ***PaymentResultController Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | doPost | This method is used to update transaction status based on result returned from payment process |

#### 

#### ***RequestDAO Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | createRequest | This method is used to create row data of request and return the inserted id. |
| 02 | updateRequest | This method is used to update status data of request by request ID. |

#### 

#### ***OrderDAO Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | createOrder | This method is used to create row data of order and return the inserted id. |
| 02 | insertOrderItem | This method takes userID, list product, orderID and requestID as input and is used to insert all items of an order. |

#### 

#### ***TransactionDAO Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | createTransaction | This method is used to create row data of transactions and return the id inserted. |
| 02 | updateTransaction | This method is used to update transaction status based on the validation request of the payment process. |

#### 

#### 

#### ***Product Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of product object’s attributes. |
| 02 | Setter | These methods are used to set the values of product object’s attributes. |

#### ***Order Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of the order object’s attributes. |
| 02 | Setter | These methods are used to set the values of order object’s attributes. |

#### 

#### ***OrderItem Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of order item object’s attributes. |
| 02 | Setter | These methods are used to set the values of order item object’s attributes. |

#### 

#### ***Transaction Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of a transaction object’s attributes. |
| 02 | Setter | These methods are used to set the values of a transaction object’s attributes. |

#### ***Request Class***

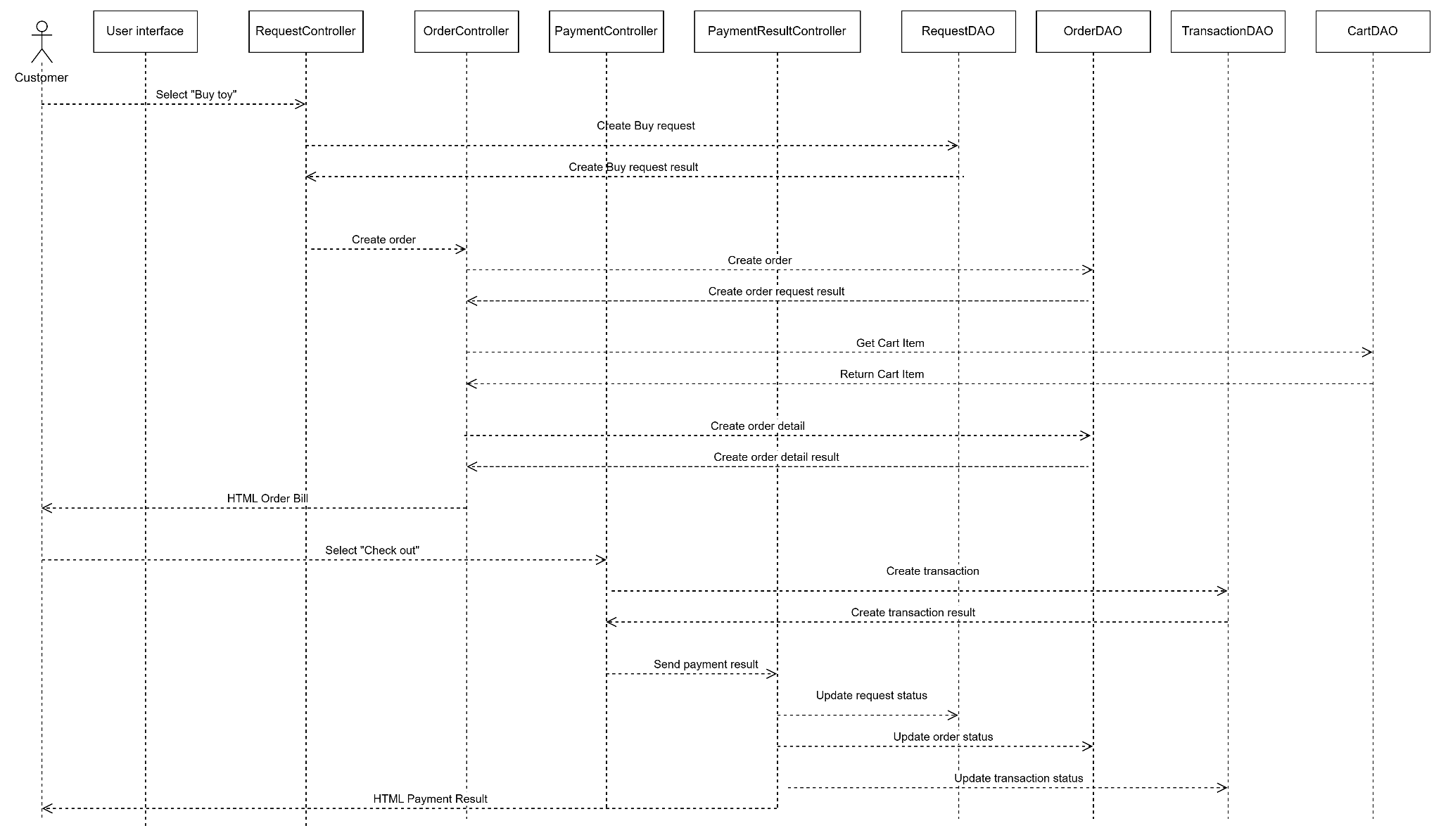
| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of the request object’s attributes. |
| 02 | Setter | These methods are used to set the values of request object’s attributes. |

#### ***User Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of the user object’s attributes. |
| 02 | Setter | These methods are used to set the values of the user object’s attributes. |

### 

### **c. Sequence Diagram(s)**



### **d. Database Queries**

SELECT i.quantityInStock, i.availableForRental

FROM Inventory i

WHERE i.productId = 'TOY\_ID';

INSERT INTO Requests (userId, productId, requestType, quantity, rentalDuration, requestDate, status, totalAmount, createdDate)

VALUES ('USER\_ID', 'TOY\_ID', BuyToy, 1, 'RENTAL\_DURATION', CURRENT\_TIMESTAMP, 'Pending', 'TOTAL\_AMOUNT', CURRENT\_TIMESTAMP);

UPDATE Requests

SET status = 'NEW\_STATUS', updatedDate = CURRENT\_TIMESTAMP

WHERE id = 'REQUEST\_ID';

INSERT INTO Orders (userId, requestId, orderDate, totalAmount, status, createdDate)

VALUES ('USER\_ID', 'REQUEST\_ID', CURRENT\_TIMESTAMP, 'TOTAL\_AMOUNT', 'Pending', CURRENT\_TIMESTAMP);

UPDATE Order

SET status = 'Completed'

WHERE id = 'Order\_ID';

INSERT INTO OrderItems (id, orderId, productId, quantity, unitPrice, totalPrice, rentalDuration, rentalReturnDate)

VALUES ('ORDER\_ITEM\_ID', 'ORDER\_ID', 'PRODUCT\_ID', 2, 15.99, 31.98, 7, NULL);

INSERT INTO Transactions (id, orderId, paymentDate, amount, status)

VALUES ('TRANSACTION\_ID', 'ORDER\_ID', CURRENT\_TIMESTAMP, 59.99, 'Completed');

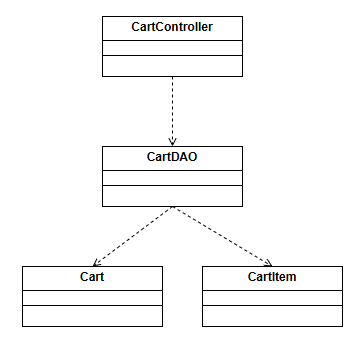
UPDATE Transactions

SET status = 'Completed'

WHERE id = 'TRANSACTION\_ID';

## **3. Manage Cart**

### **a. Class Diagram**



### **b. Class Specifications**

#### ***CartController Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | doPost | This method is used to perform cart manage action |

#### 

#### ***CartDAO Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | addToCart | This method is used to create a cart for an user. |
| 02 | updateCartItemQuantity | This method is used to update cart item quantity based on the cart id and cart item id. |
| 03 | removeFromCart | This method is used to remove cart items based on the cart id and cart item id. |

#### ***Cart Class***

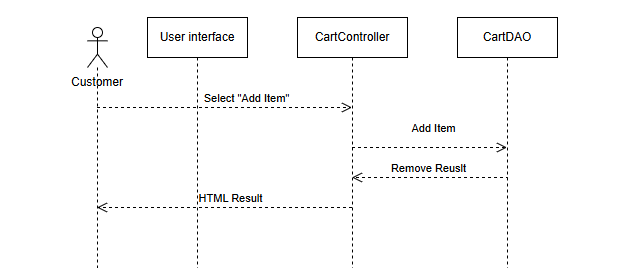
| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of a cart object’s attributes. |
| 02 | Setter | These methods are used to set the values of a cart object’s attributes. |

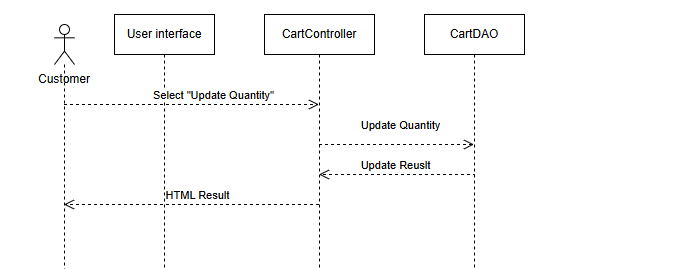
#### ***CartItem Class***

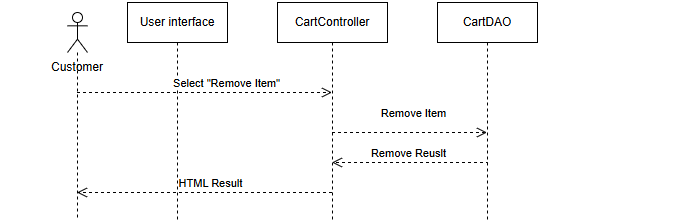
| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of a cartItem object’s attributes. |
| 02 | Setter | These methods are used to set the values of a cartItem object’s attributes. |

#### 

### **c. Sequence Diagram(s)**







### **d. Database Queries**

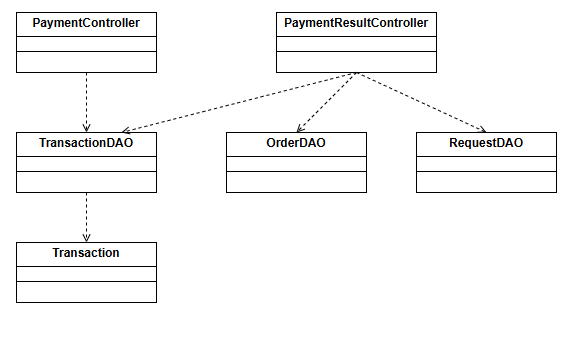
INSERT INTO CartItems (id, cartId, productId, quantity, isRental, rentalDuration) VALUES ('CART\_ITEM\_ID', 'CART\_ID', 'PRODUCT\_ID', 1, TRUE, 7);

UPDATE CartItems SET quantity = new\_quantity WHERE id = 'CART\_ITEM\_ID';

SELECT ci.id AS cartItemId, p.id AS productId, p.name AS productName, ci.quantity, p.rentalPrice, ci.isRental, ci.rentalDuration FROM CartItems ci JOIN Products p ON ci.productId = p.id WHERE ci.cartId = 'CART\_ID';

## **4. Checkout**

### **a. Class Diagram**



### **b. Class Specifications**

#### ***PaymentController Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | doPost | This method is used to request to the 3rd party for payment processing |

#### 

#### ***PaymentResultController Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | doPost | This method is used to update transaction status based on result returned from payment process |

#### 

#### ***TransactionDAO Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | createTransaction | This method is used to create row data of transactions and return the id inserted. |
| 02 | updateTransaction | This method is used to update transaction status based on the validation request of the payment process. |

#### ***OrderDAO Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 02 | updateOrder | This method is used to update order status based on the validation request of the payment process. |

#### ***RequestDAO Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 02 | updateRequest | This method is used to update request status based on the validation request of the payment process. |

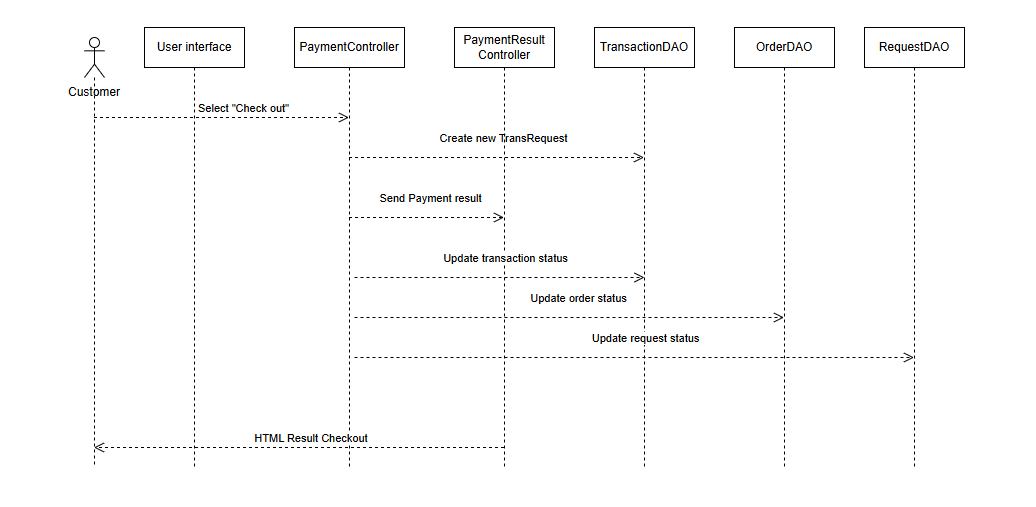
#### 

#### ***Transaction Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of a transaction object’s attributes. |
| 02 | Setter | These methods are used to set the values of a transaction object’s attributes. |

#### 

### **c. Sequence Diagram(s)**



### **d. Database Queries**

INSERT INTO Transactions (id, orderId, paymentDate, amount, status)

VALUES ('TRANSACTION\_ID', 'ORDER\_ID', CURRENT\_TIMESTAMP, 59.99, 'Completed');

UPDATE Transactions

SET status = 'Completed'

WHERE id = 'TRANSACTION\_ID';

UPDATE Order

SET status = 'Completed'

WHERE id = 'Order\_ID';

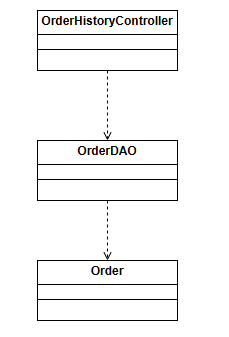
UPDATE Request

SET status = 'Completed'

WHERE id = 'Request\_ID';

## **5.** **View Order history**

### **a. Class Diagram**



### ***b. Class Specifications***

#### ***OrderHistoryController Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | doPost | This method is used to get data through dao and send it to the user for viewing. |

#### 

#### ***OrderDAO Class***

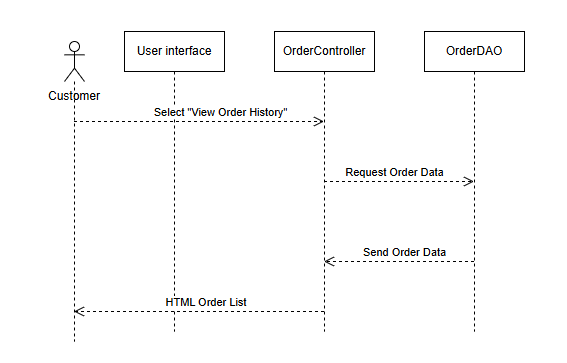
| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getAllOrder | This method is used to retrieve all order data from database |

#### ***Order Class***

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | Getter | These methods are used to get the values of an order object’s attributes. |
| 02 | Setter | These methods are used to set the values of an order object’s attributes. |

#### 

### ***c. Sequence Diagram(s)***



### ***d. Database Queries***

SELECT \*

FROM Order

WHERE User\_ID = 'UserID\_ID';