Newbies

NBTech
Software Architecture Document

Version 1.0

Revision History

Date	Version	Description	Task	Author
02/08/2025	1.0	Fulfill required information for the software architecture in PA3.	Section 2	Đỗ Đăng Nhật Tiến
			Section 1 and Section 3	Trần Minh Quang
			Section 4.1.1, 4.1.2, Section 4.2.1.5 Section 4.3.1 Section 4.5.1 Section 4.6.3, 4.6.5	Nguyễn Thanh Owen
			Section 4.1.5, 4.1.6 Section 4.2.1.1, 4.2.1.2 Section 4.2.2 Section 4.4.1, 4.4.3 Section 4.6.1	Lê Nguyên Thảo
			Section 4.1.3, 4.1.4 Section 4.2.1.3, 4.2.1.4 Section 4.4.2, 4.4.4 Section 4.6.2, 4.6.4	Nguyễn Ngọc Hưng Phát

Table of Contents

Software Architecture Document

1. Introduction

1.1. Purpose

This document aims to deliver a comprehensive architecture design of the NBTech e-commerce website, emphasizing the program's model and all associated packages. These characteristics were selected due to their significance in the application's design and development.

1.2. Scope

This document intends to help the development team determine how the online shopping system will be structured at the highest architectural level. It is also intended for the project sponsors to review and approve the overall system design before moving into the detailed implementation phase. Finally, the project coach or supervisor can use this document to ensure that the development team is aligning with the agreed-upon functional and non-functional requirements throughout the development of the e-commerce website.

1.3. Definitions

This document will address the background for this project, and the architecturally significant functional requirements. Each package will be documented using a complete set of class diagrams, followed by an architectural overview that includes the point of view of and a detailed discussion of the patterns that will be utilized to address the fundamental quality aspects. This will be followed by a look at some other perspectives on the system.

1.4. Acronyms

- MVC: Model - View - Controller model
- SAD: Software Architecture Document
- UI: User Interface
- API: Application Programming Interface, a protocol used as an interface to allow communication between different components.
- REST: Representational State Transfer, web API featuring a stateless client-server infrastructure.

1.5. References

- HTML-CSS: OReilly Head First HTML with CSS XHTML
- JavaScript: OReilly Head First JavaScript
- Nodejs: nodejs.org and youtube: The Complete Node.js Bootcamp: From Beginner to Expert | Unloc…
- Use-Case model, Package diagram, class diagram: draw.io

1.6. Overview

This Software Architecture Document presents an overview of the system architecture for the e-commerce website project. It includes key architectural goals and constraints derived from non-functional requirements, the use-case model that guides system behavior, and a logical view describing system components and their interactions. While the deployment and implementation views are placeholders in this version, they are reserved for future development. The document aims to provide a clear understanding of the system's structure to support design, development, and communication among stakeholders.

2. Architectural Goals and Constraints

2.1. Architectural Goals

- **Cross-platform Compatibility:**
  - The system must run on all major web browsers such as Google Chrome, Mozilla Firefox, and Microsoft Edge, and be fully functional on Windows-based desktop and laptop environments.

- **Performance and Scalability:**
  - The system should support at least 100 concurrent users accessing key functionalities like product search, account login, and checkout without noticeable delays. It should also be scalable to accommodate future growth in both users and data volume.

- **Security and Data Protection:**
  - The architecture must ensure that all user passwords are securely stored and sensitive operations (e.g., administrative functions) are access-restricted to authorized personnel only.

- **Modularity and Reusability:**
  - Key software components such as product cards, search modules, cart and checkout functions should be modular and reusable across the application.

- **Comprehensive Documentation:**
  - The system should include clear user manuals and help sections for both customers and administrators, along with detailed technical documentation to support installation, configuration, and maintenance.

2.2. Architectural Constraints

- **Limited Client Resource Usage**
  - The system must operate with no more than 512MB RAM on the client side and function reliably over a standard internet connection.

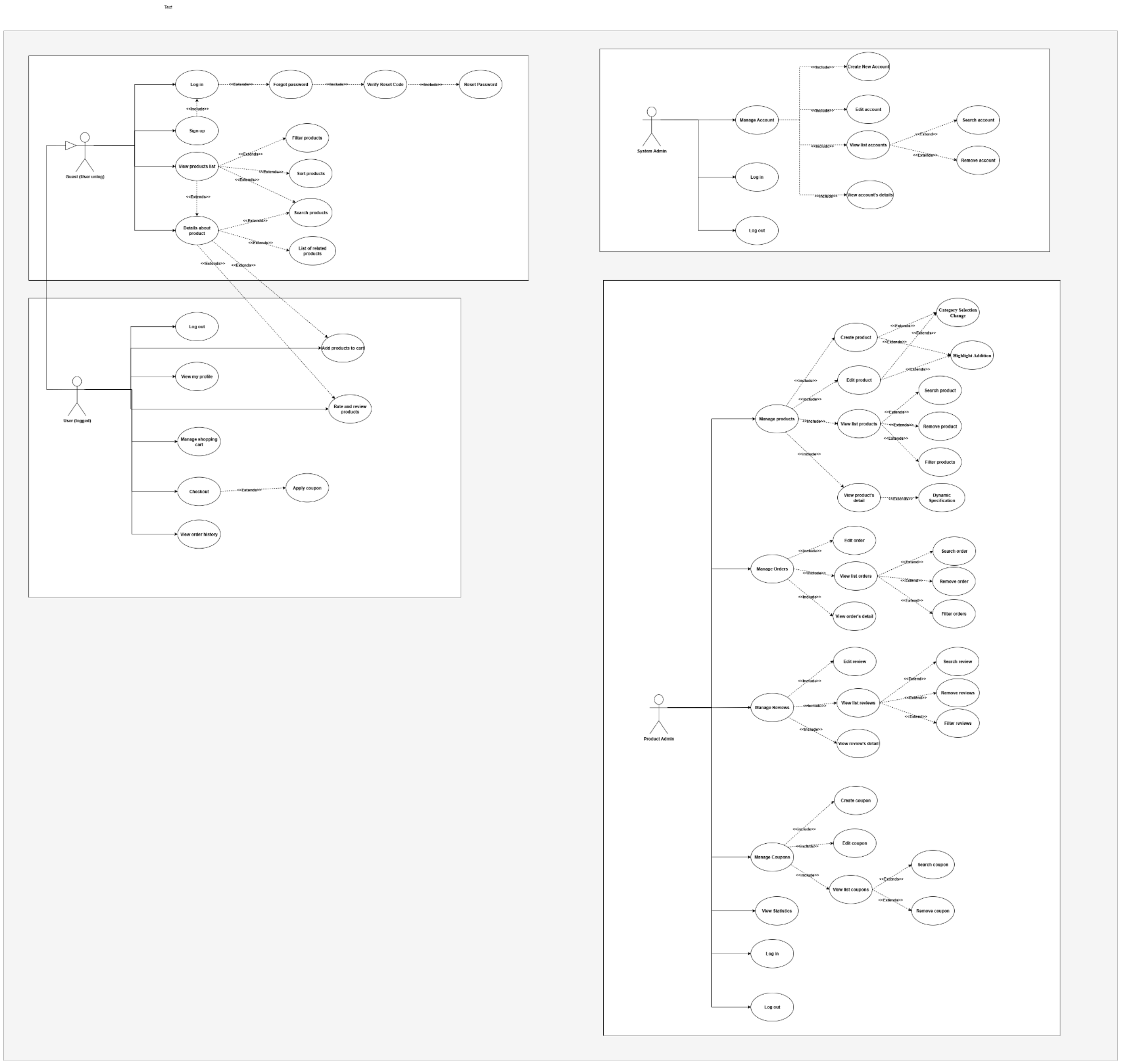
- **Network Connection Independence**
  - The client interface must be optimized for both wired and wireless internet connections.

- **Browser and OS Scope**
  - The web application is intended to work on major browsers and Windows-based systems only; support for other platforms is not prioritized.

- **Small Development Team**
  - The system architecture must remain maintainable and relatively simple due to a limited development team size (3–5 members).

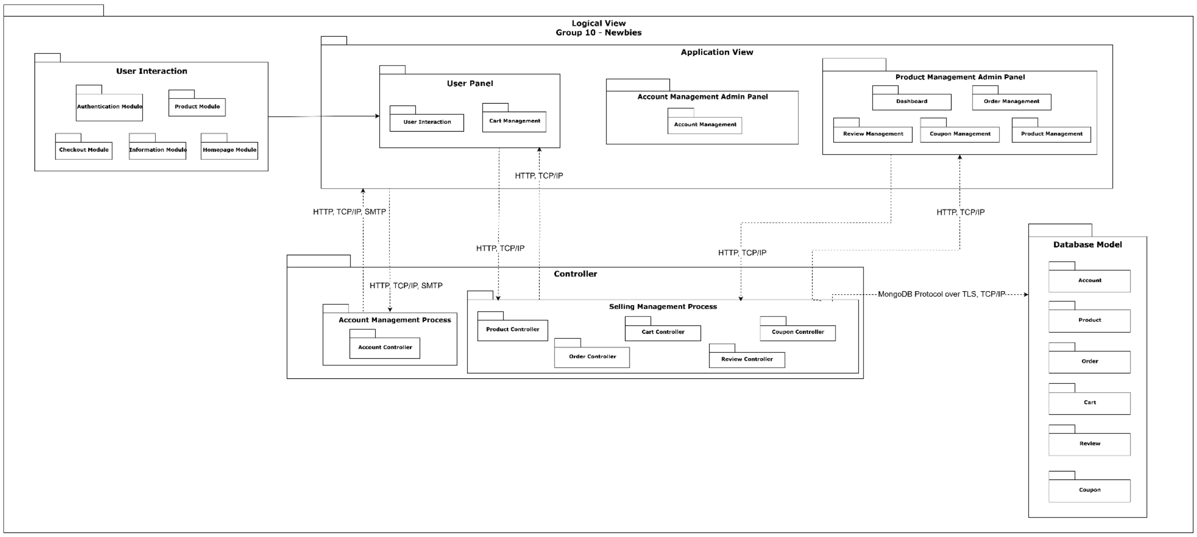
- **Cloud-Based Deployment**
  - The system must be designed for deployment on a cloud infrastructure (e.g., AWS, Azure) to ensure scalability and uptime reliability.

3. Case Model



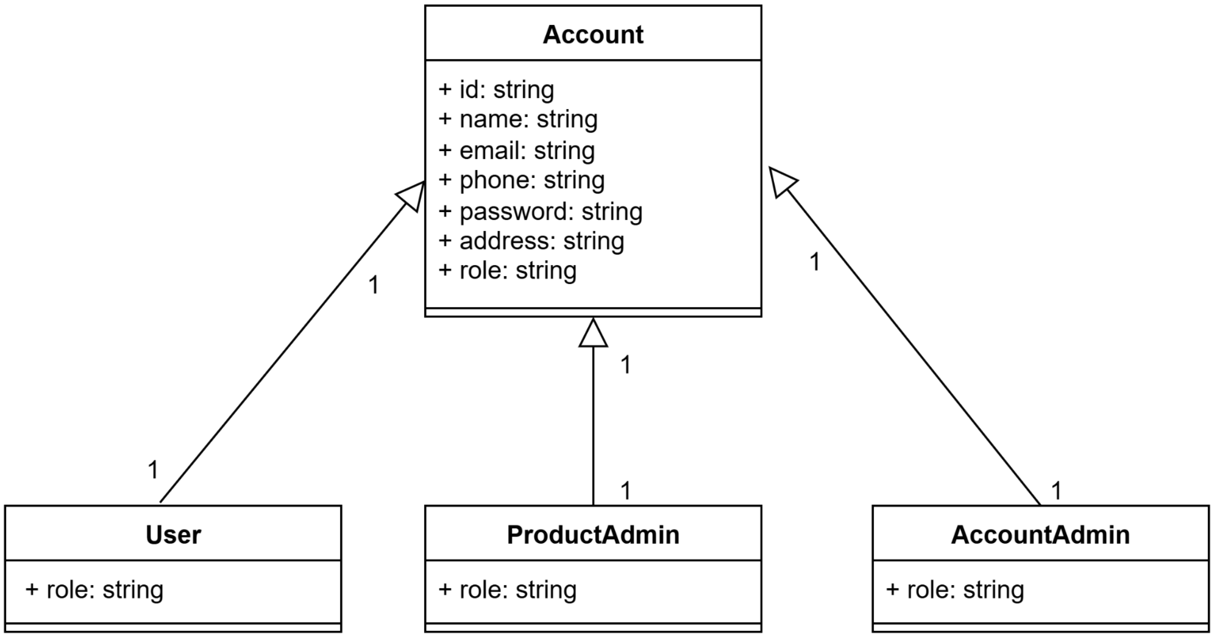
Text
Guest (User unlog)
Log in
<<Extends>>
Forgot password
<<Include>>
Verify Reset Code
<<Include>>
Reset Password
<<Include>>
Sign up
View products list
<<Extends>>
Filter products
<<Extends>>
Sort products
<<Extends>>
Search products
<<Extends>>
Details about product
<<Extends>>
<<Extends>>
List of related products
<<Extends>>
<<Extends>>
User (logged)
Log out
Add products to cart
View my profile
Rate and review products
Manage shopping cart
Checkout
<<Extends>>
Apply coupon
View order history
System Admin
Manage Account
<<Include>>
Create New Account
<<Include>>
Edit account
<<Include>>
View list accounts
<<Extend>>
Search account
<<Extends>>
Remove account
<<Include>>
View account's details
Log in
Log out
Product Admin
Manage products
<<include>>
Create product
<<Extends>>
Category Selection Change
<<Extends>>
<<Extends>>
Highlight Addition
<<include>>
Edit product
<<Extends>>
<<Include>>
View list products
<<Extends>>
Search product
<<Extends>>
Remove product
<<Extends>>
Filter products
<<include>>
View product's detail
<<Extends>>
Dynamic Specification
Manage Orders
<<Include>>
Edit order
<<Include>>
View list orders
<<Extend>>
Search order
<<Extend>>
Remove order
<<Extend>>
Filter orders
<<Include>>
View order's detail
Manage Reviews
<<Include>>
Edit review
<<Include>>
View list reviews
<<Extend>>
Search review
<<Extend>>
Remove reviews
<<Extend>>
Filter reviews
<<Include>>
View review's detail
Manage Coupons
<<include>>
Create coupon
<<include>>
Edit coupon
<<include>>
View list coupons
<<Extends>>
Search coupon
<<Extends>>
Remove coupon
View Statistics
Log in
Log out

4. Logical View



4.1. Database Model

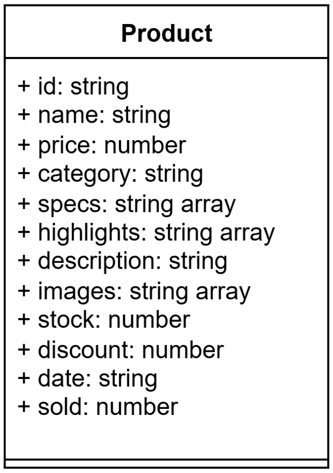
4.1.1. Account



- Class: Account (Base Class):
    - Description: This class is the base class that represents a generic user account in the system. It defines all the common attributes shared by all types of users, regardless of their specific roles.
    - Attributes:
        - Id: string
        - Name: string
        - Email: string
        - Phone: string

        - Password: string
        - Address: string
        - Role: string
- Class: User (Inherited):
    - Description: The User class inherits from the Account class and represents a **regular customer** or general user of the platform.
    - Attributes:
        - Role: user
- Class: Product Admin (Inherited):
    - Description: This class extends the Account class and represents a user with permissions to manage product-related data.
    - Attributes:
        - Role: p_admin
- Class: Account Admin (Inherited):
    - Description: This class also extends the Account class and represents an administrator responsible for managing user accounts and roles.
    - Attributes:
        - Role: a_admin

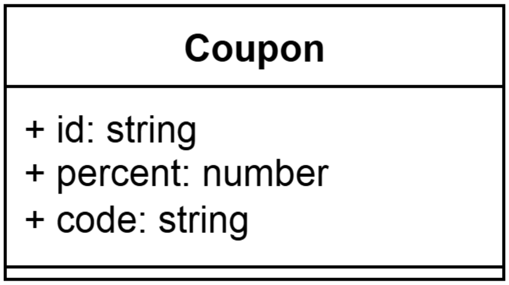
4.1.2. Product



- Class: Product
    - Description: The Product class represents an item available for sale in the system. It contains detailed information about the product.
    - Attributes:
        - id: string (Unique identifier for the product.)
        - name: string (Name of the product.)
        - price: number (Price of the product.)
        - category: string (Product category.)
        - specs: string[] (List of product specifications or technical details.)
        - highlights: string[] (Key features or selling points of the product.)
        - description: string (Detailed description of the product.)
        - images: string[] (Array of image URLs representing the product.)
        - stock: number (Quantity of the product available in stock.)
        - discount: number (Discount percentage applied to the product.)
        - date: string (Date of product addition or release.)

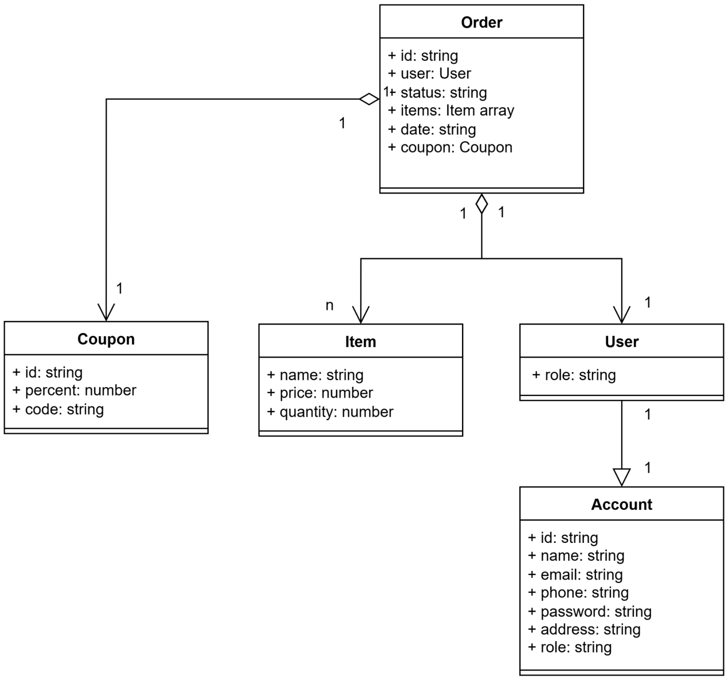
- sold: number (Number of units sold.)

4.1.3. Coupon



- Class: Coupon
  - Description: The Coupon class represents a structure for coupons.
  - Attributes:
    - id: string (Unique identifier for the coupon)
    - percent: number (the percentage of the discount)
    - code: string (the code of the coupon)

4.1.4. Order

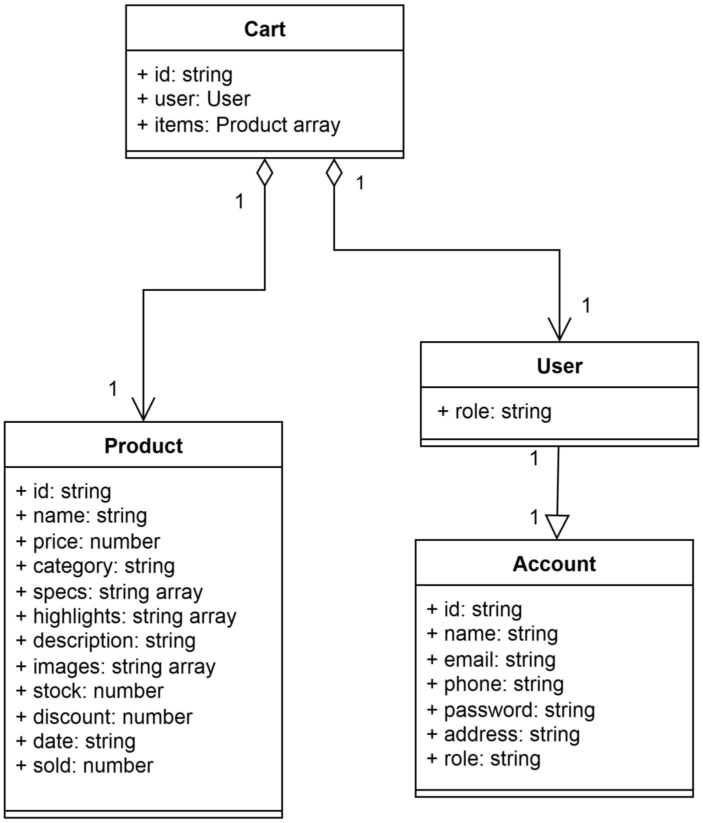


- Class: Order
  - Description: The Order class represents a structure for managing each order of a user. It contains the user's information, a list of items that were bought, and the coupon code if it was applied.
  - Attributes:
    - Id: string (unique identifier for the order)
    - User: User (represents the current user)

- Status: string (the status of the order)
- Items: item array (a list of products that were bought)
- Date: string (date when the user placed the order)
- Coupon: Coupon (a coupon code)

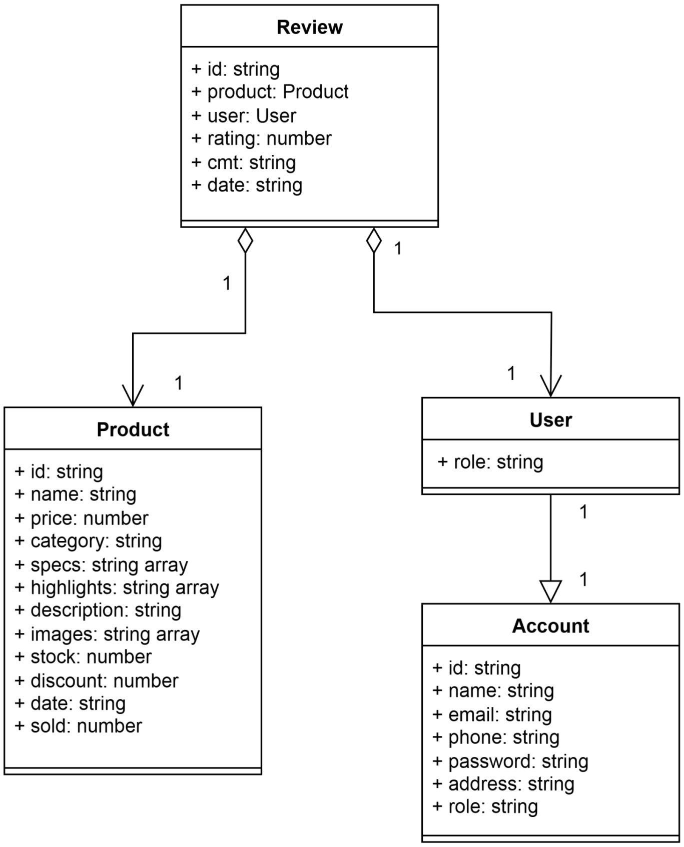
- Class: Item
  - Description:The Item class manages the information of the product that was bought.
  - Attributes:
    - name: string (name of the product)
    - price: number (the product's cost)
    - quantity: number (the number of products was chosen)

4.1.5. Cart



- Class: Cart
  - Description: The Cart class represents a structure for managing products selected by a user before purchase. It is linked to a user account and stores a list of items.
  - Attributes:
    - Id: string (Unique identifier for the cart)
    - User: User (The user who owns the cart)
    - Items: Product array (The list of products added to the cart)

4.1.6. Review

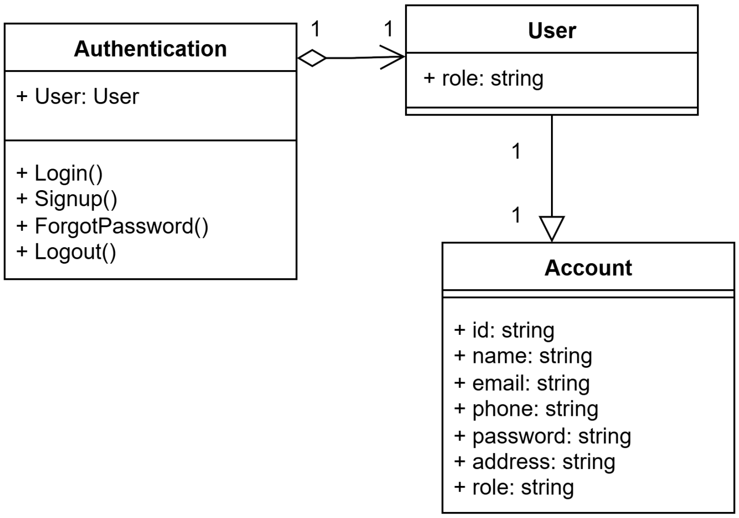


- Class: Review
  - Description: The Review class represents user feedback for a specific product. Each review is submitted by a user, linked to a product, and includes a rating, comment, and posting date.
  - Attributes:
    - Id: string (Unique identifier for the review)
    - Product: Product (The product being reviewed)
    - User: User (The user who wrote the review)
    - Rating: number (The numeric rating given by the user)
    - Cmt: string (The comment content of the review)
    - Date: string (The date when the review was posted)

4.2. Package: User Panel

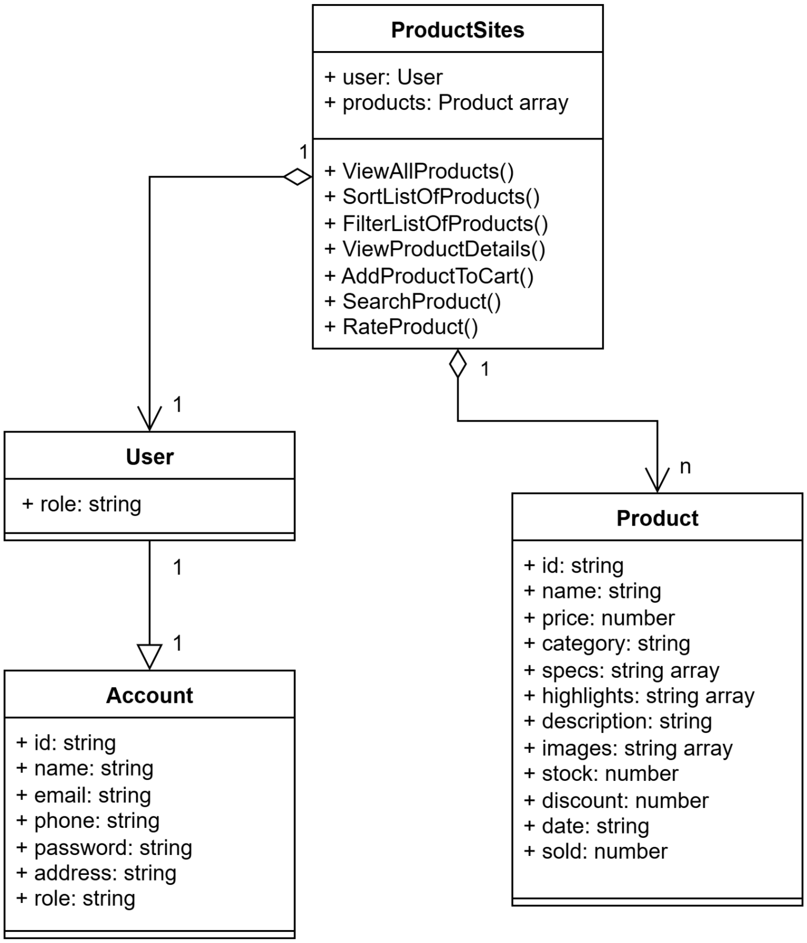
4.2.1. User Interaction

4.2.1.1. Authentication Module



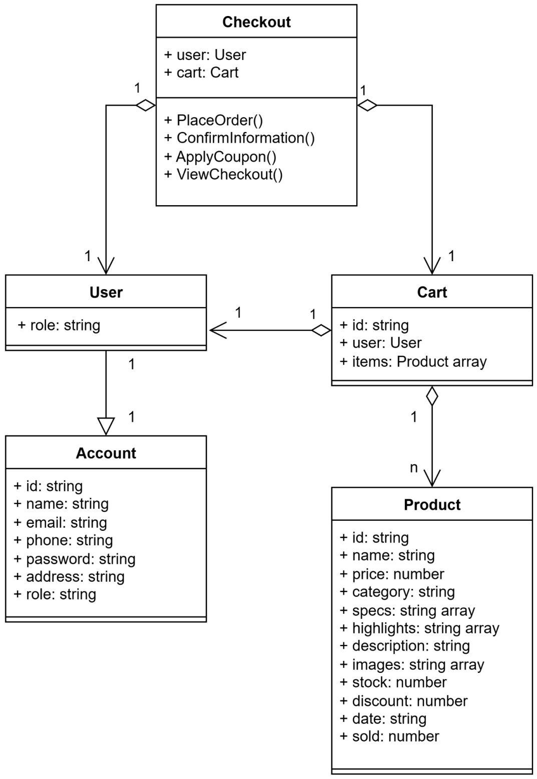
- Class Name: Authentication
  - Description: The Authentication class handles all login-related operations, including user login, registration, password recovery, and logout.
  - Attributes:
    - User: User (The user who is attempting authentication)
  - Methods:
    - Login(): Authenticates a user by verifying their email and password.
    - Signup(): Registers a new user account with the provided information.
    - ForgotPassword(): Sends a reset code to the user’s email to recover their password.
    - Logout(): Ends the user’s session and clears their authentication data.

4.2.1.2. Product Module



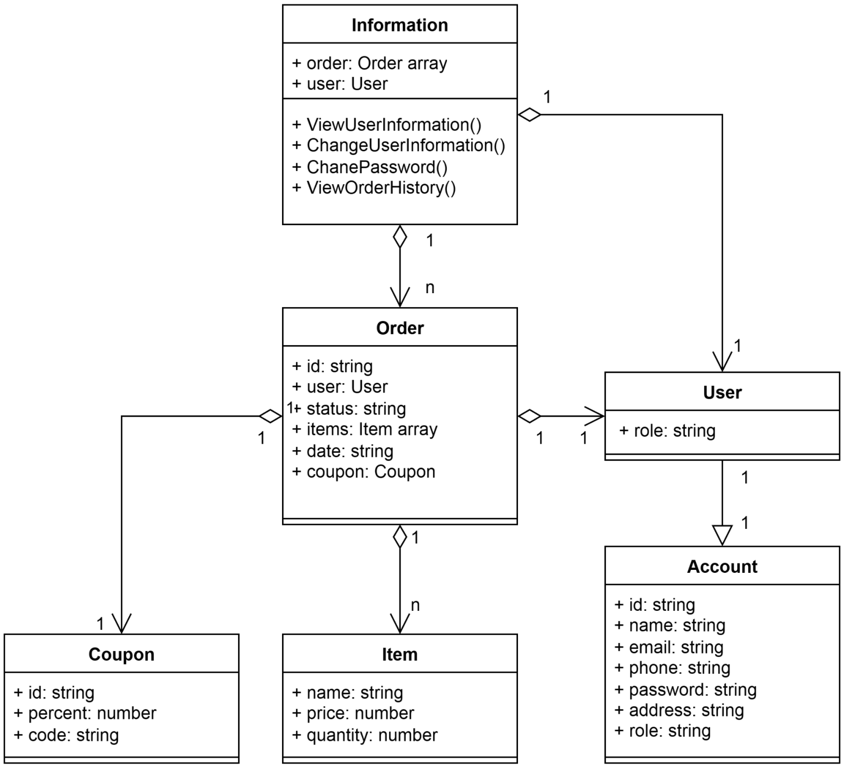
- Class: Product Module
  - Description: The ProductSites class handles product browsing features, allowing users to view, sort, filter, search products, add items to the cart, and submit reviews. It serves as the interface between users and the product catalog.
  - Attributes:
    - User: User (The user viewing the products)
    - Products: Product array (List of products displayed on the site)
  - Methods:
    - ViewAllProducts(): Displays all products available on the site
    - SortListOfProducts(): Sorts the product list by selected criteria
    - FilterListOfProducts(): Filters the product list based on chosen conditions
    - ViewProductDetails(): Shows detailed information of a selected product
    - AddProductToCart(): Adds a selected product to the user’s cart
    - SearchProduct(): Searches for products by keyword
    - RateProduct(): Allows the user to submit a review and rating for a product

4.2.1.3. Checkout Module



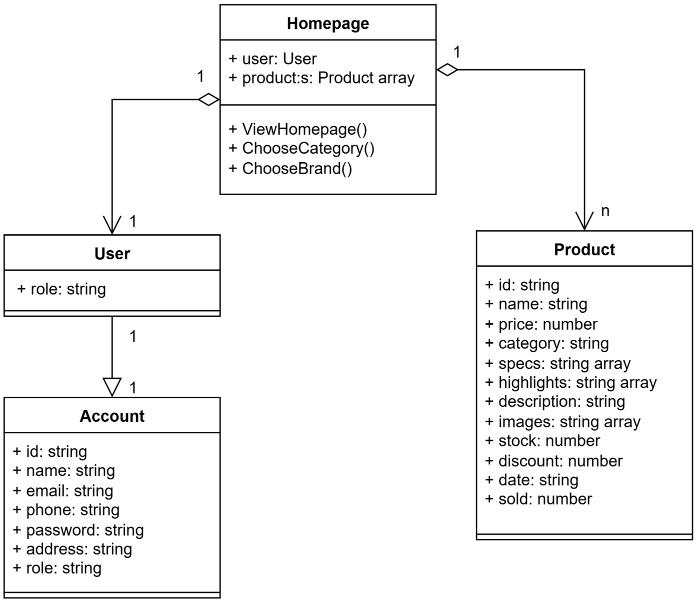
- Class: Checkout
  - Description: This allows the user to place an order. It displays an information form to fill, a list of products that were in the user's cart, and the total price. Moreover, the user can apply a coupon to get a discount.
  - Attributes:
    - user: User (Represents the current user)
    - cart: Cart (Containing all of the products the user chose)
  - Methods:
    - PlaceOrder(): Allow the user to place the order.
    - ConfirmInformation(): Let the user enter the information form for shipping.
    - ApplyCoupon(): Allow the user to apply a coupon to get the discount for the bill.
    - ViewCheckout(): Displays an information form, a list of products, the total price before and after applying a coupon.

4.2.1.4. Information Module



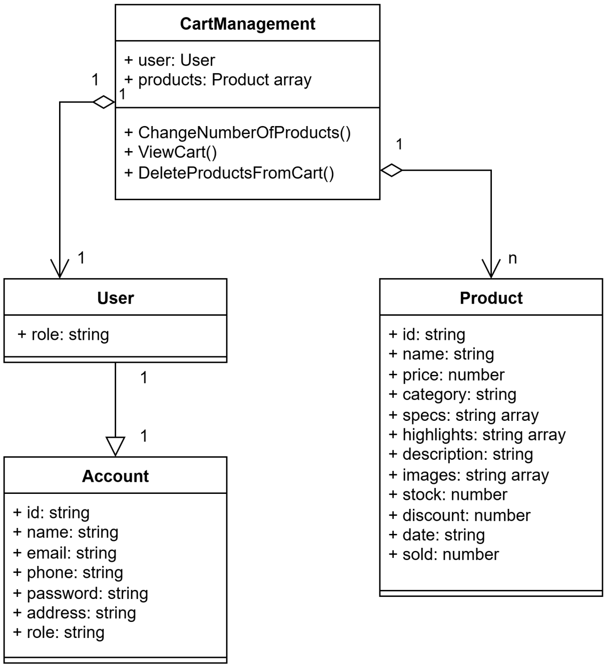
- Class: Information
  - Description: The Information class takes the responsibilites for displaying the user's information and allows the user to change the password in need. Besides, the user can view the order history by clicking on the link in the navigation bar.
  - Attributes:
    - user: User (Represents the current user)
    - Order: Order array (a list containing all of the user’s orders)
  - Methods:
    - ViewUserInformation(): Display the user’s information page, including fields of information and a button for changing password.
    - ChangeUserInformation(): Allow the user to edit the information in each field.
    - ChangePassword(): Allow the user to change the password in need.
    - ViewOrderHistory(): Display the user’s order history, including the user’s information and all of the previous orders. Each order contains a list of products, the price, date, and the status of the order.

4.2.1.5. Homepage Module



- Class: Homepage
  - Description: The Homepage class represents the main interface of the system that users see upon visiting the website. It is responsible for displaying available products, allowing users to filter by category or brand, and providing access to user account information.
  - Attributes:
    - user: User (Represents the current user interacting with the homepage.)
    - products: Product array (A list containing all the products displayed on the homepage)
  - Methods:
    - ViewHomepage(): Displays the homepage interface, including user information, banners, and the list of products.
    - ChooseCategory(): Allows the user to select a product category and filters the displayed products accordingly.
    - ChooseBrand(): Lets the user choose a specific brand and shows only products from that brand.

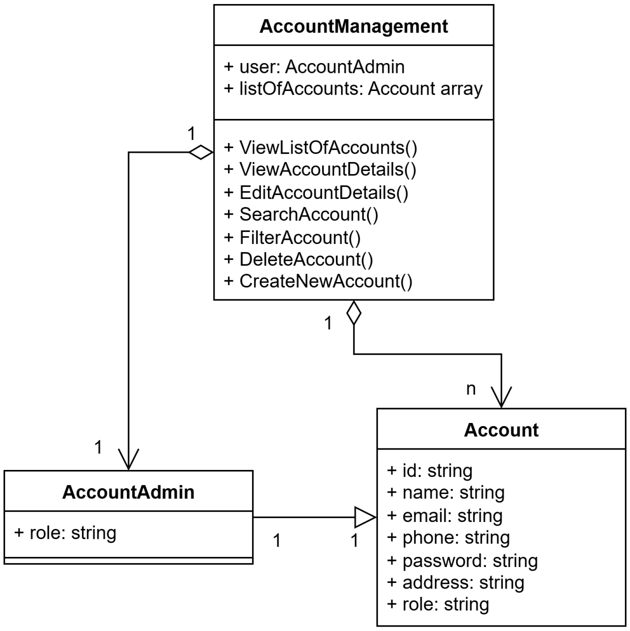
4.2.2. Cart Management



- Class: Cart Management
  - Description: The Cart Management class allows users to manage the contents of their shopping cart, including viewing, updating, or removing products.
  - Attributes:
    - user: User (The user who owns the cart)
    - products: Product array (The list of products in the cart)
  - Methods:
    - ChangeNumberOfProducts(): Updates the quantity of a specific product in the cart
    - ViewCart(): Displays all products currently in the cart
    - DeleteProductsFromCart(): Removes selected products from the cart

4.3. Package: Account Management Admin Panel

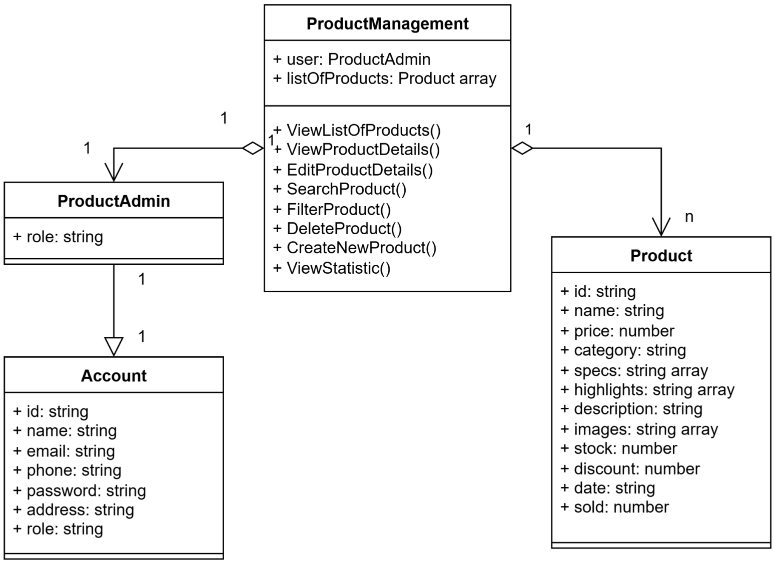
4.3.1. Account Management



- Class: AccountManagement
  - Description: The AccountManagement class handles all operations related to user accounts within the system. It allows an administrator (AccountAdmin) to view, create, edit, search, filter, and delete user accounts
  - Attributes:
    - user: AccountAdmin (The administrator responsible for managing accounts.)
    - listOfAccounts: Account array (A collection of accounts existing in the system)
  - Methods:
    - ViewListOfAccounts(): Displays the full list of user accounts.
    - ViewAccountDetails(): Shows detailed information for a specific account.
    - EditAccountDetails(): Updates information for a specific account.
    - SearchAccount(): Finds accounts matching a search keyword.
    - FilterAccount(): Filters accounts based on specified criteria (role).
    - DeleteAccount(): Removes an account from the system.
    - CreateNewAccount(): Adds a new account to the system.

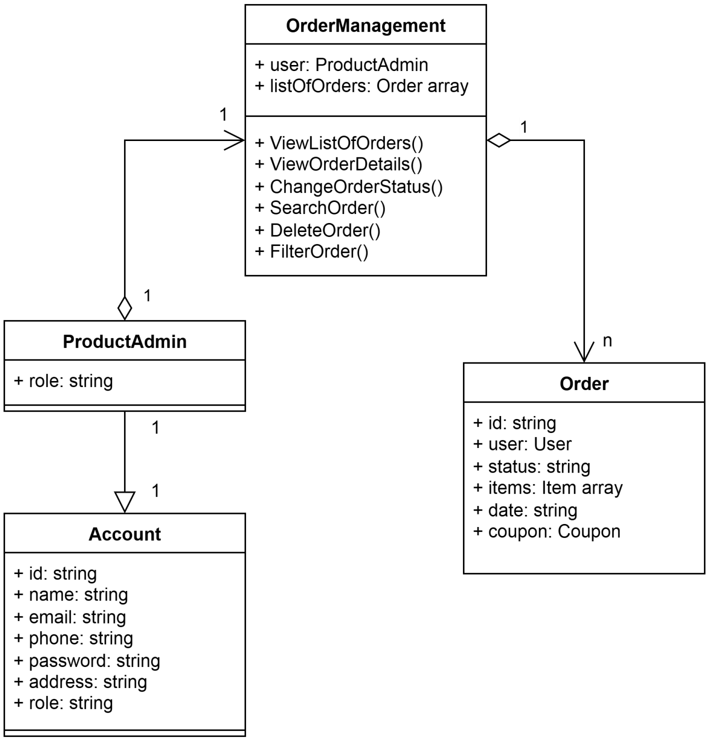
4.4. Package: Product Management Admin Panel

4.4.1. Product Management



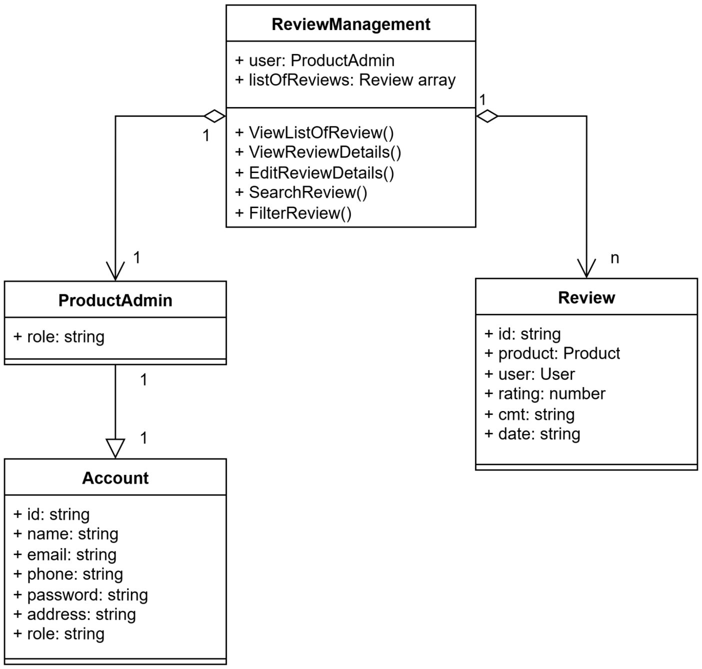
- Class: Product Management
  - Description: The Product Management class is used by product administrators to control product-related data such as listing, editing, creating, and deleting products.
  - Attributes:
    - user: ProductAdmin (The admin user responsible for product management)
    - listOfProducts: Product[] (List of products being managed)
  - Methods:
    - ViewListOfProducts() (Displays all products in the system)
    - ViewProductDetails() (Shows detailed information of a product)
    - EditProductDetails() (Allows editing of product attributes)
    - SearchProduct() (Finds products by criteria)
    - FilterProduct() (Filters product list based on attributes)
    - DeleteProduct() (Removes a product from the catalog)
    - CreateNewProduct() (Adds a new product to the catalog)
    - ViewStatistic() (Shows product-related statistics)

4.4.2. Order Management



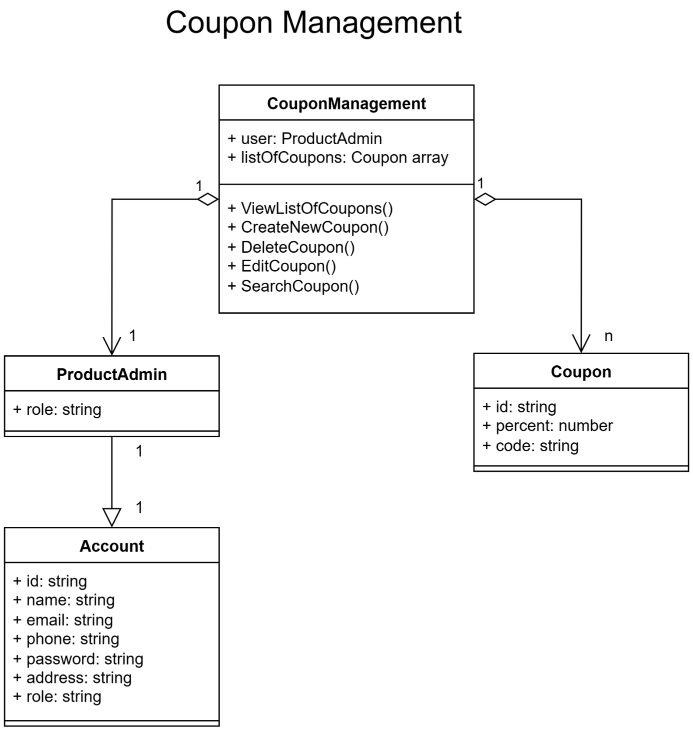
- Class: OrderManagement
  - Description: The OrderController class is used for the admin managing all the orders of the users. It allows the admin to view a list of orders, and to be able to change the status of an order.
  - Attributes:
    - user: ProductAdmin (represents for the admin managing the orders)
    - listOfOrders: Order array (a list of all users' orders)
  - Methods:
    - ViewListOfOrders(): Displays a list of orders of all the users.
    - ViewOrderDetails(): View the details of an order of one user.
    - ChangeOrderStatus(): Allow the admin to change the status of an order.
    - SearchOrder(): Let the admin find the order based on the ID, Date, Full Name, Phone Number, Total fields of an order.
    - DeleteOrder(): Allow the admin to delete an order from the database.
    - FilterOrder(): Let the admin find the orders based on each status.

4.4.3. Review Management



- Class: Review Management
  - Description: The Review Management class allows product administrators to manage product reviews, including viewing, editing, searching, and
  - Attributes:
    - user: ProductAdmin (The admin managing the reviews)
    - listOfReviews: Review[] (List of reviews under management)
  - Methods:
    - ViewListOfReview(): Displays all submitted product reviews
    - ViewReviewDetails(): Shows detailed content of a selected review
    - EditReviewDetails(): Allows the admin to modify an existing review
    - SearchReview(): Finds reviews based on criteria such as product or user
    - FilterReview(): Filters reviews using specific parameters

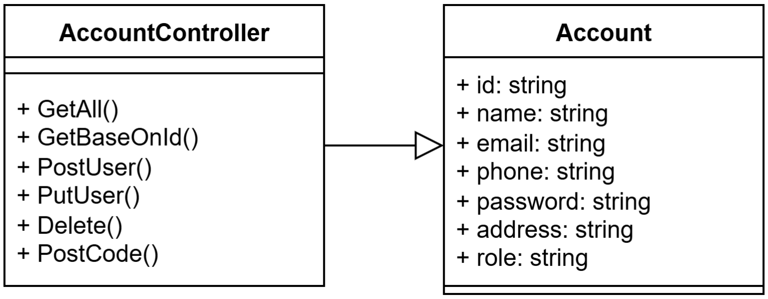
4.4.4. Coupon Management



- Class: CouponManagement
  - Description: The CouponManagement helps the admin manage all the coupons.
  - Attributes:
    - User: ProductAdmin (represents for the admin managing the coupons)
    - listOfCoupons: Coupon array (a list of all the existing coupons)
  - Methods:
    - ViewListOfCoupons(): Displays all the existing coupons.
    - CreateNewCoupon(): Allow the admin to create a new coupon (containing a coupon code and the percentage of the discount).
    - DeleteCoupon(): Allow the admin to delete an existing coupon from the database.
    - EditCoupon(): Let the admin change the coupon code or its percentage.
    - SearchCoupon(): The admin can search a coupon based on the ID, coupon code, or the percentage.

4.5. Package: Account Management Process

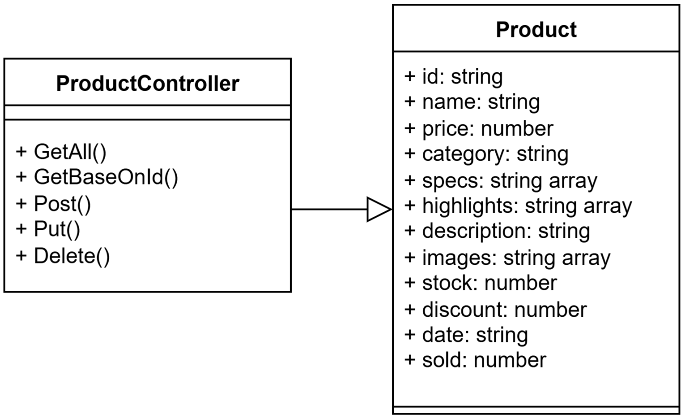
4.5.1. Account Controller



- Class: AccountController
    - Description: The AccountController acts as the backend component responsible for handling client requests related to user accounts. It processes these requests by interacting with the database through data access methods. The controller sends queries to the database to retrieve, create, update, or delete account records.
    - Methods:
        - GetAll(): Retrieve all accounts.
        - GetBaseOnId(): Retrieve account details by ID.
        - PostUser(): Create a new account.
        - PutUser(): Update existing account details.
        - Delete(): Delete an account.
        - PostCode(): Handle code-related requests (e.g., verification, password reset).

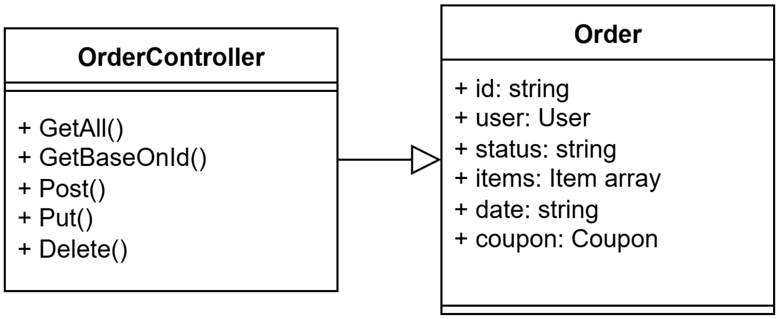
4.6. Package: Selling Management Process

4.6.1. Product Controller



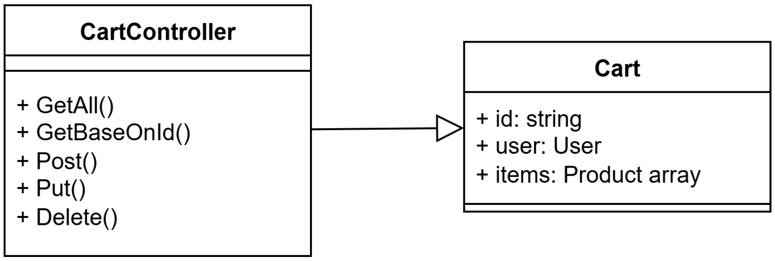
- Class: Product Controller
    - Description: The Product Controller class handles backend operations related to product data in the selling process.
    - Methods:
        - GetAll(): Retrieves all products
        - GetBaseOnId(): Retrieves a product based on its ID
        - Post(): Create a new product to the system
        - Put(): Updates information of an existing product
        - Delete(): Removes a product from the system

4.6.2. Order Controller



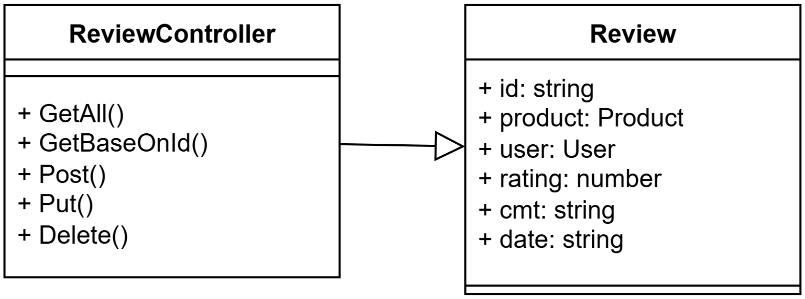
- Class: OrderController
  - Description: The OrderController handles backend operations related to the user's order. It processes client requests to retrieve all orders, get a specific order by ID, add new orders, update existing orders, and delete orders. The controller interacts with the database to manage order's data.
  - Methods:
    - GetAll(): Retrieve all orders.
    - GetBaseOnId(): Retrieve an order by its ID.
    - Post(): Create a new order.
    - Put(): Update an existing order.
    - Delete(): Delete an order.

4.6.3. Cart Controller



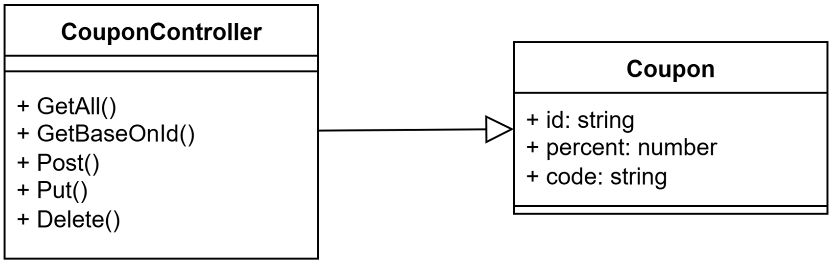
- Class: CartController
  - Description: The CartController handles backend operations related to shopping carts. It processes client requests to retrieve all carts, get a specific cart by ID, add new carts, update existing carts, and delete carts. The controller interacts with the database to manage cart data.
  - Methods:
    - GetAll(): Retrieve all carts.
    - GetBaseOnId(): Retrieve a cart by its ID.
    - Post(): Create a new cart.
    - Put(): Update an existing cart.
    - Delete(): Delete a cart.

4.6.4. Review Controller



- Class: ReviewController
  - Description: The ReviewController handles backend operations related to users' reviews. It processes client requests to retrieve all reviews, get a specific review by ID, add new reviews, update existing reviews, and delete reviews. The controller interacts with the database to manage reviews' data.
  - Methods:
    - GetAll(): Retrieve all reviews.
    - GetBaseOnId(): Retrieve a review by its ID.
    - Post(): Create a new review.
    - Put(): Update an existing review.
    - Delete(): Delete a review

4.6.5. Coupon Controller



- Class: CouponController
  - Description: The CouponController manages backend operations related to coupons. It handles requests to retrieve all coupons, get a coupon by ID, create new coupons, update existing coupons, and delete coupons. It interacts with the database to manage coupon data.
  - Methods:
    - GetAll(): Retrieve all coupons.
    - GetBaseOnId(): Retrieve a coupon by its ID.
    - Post(): Create a new coupon.
    - Put(): Update an existing coupon.
    - Delete(): Delete a coupon.

5. Deployment

None

6. Implementation View

None