Title Page

Software Title: Comazon

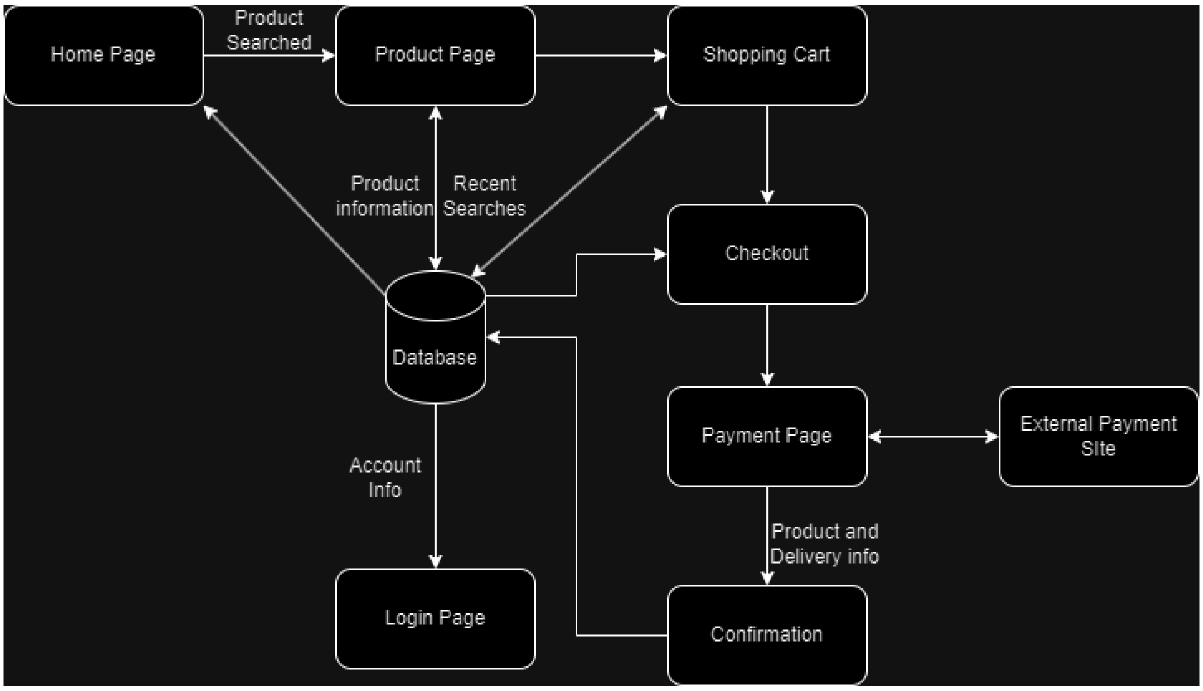
Team Members: Nathan Chau, Jimin Li

System Description

Comazon is an e-commerce system that specializes in the clothing department. Similar to *Amazon*, Comazon will offer a large variety of clothing items. The purpose of Comazon is to provide a quick and easy shopping experience for its customers, allowing them to browse, search, and purchase clothing items easily.

Software Architecture Overview

- **Architectural diagram of all major components**

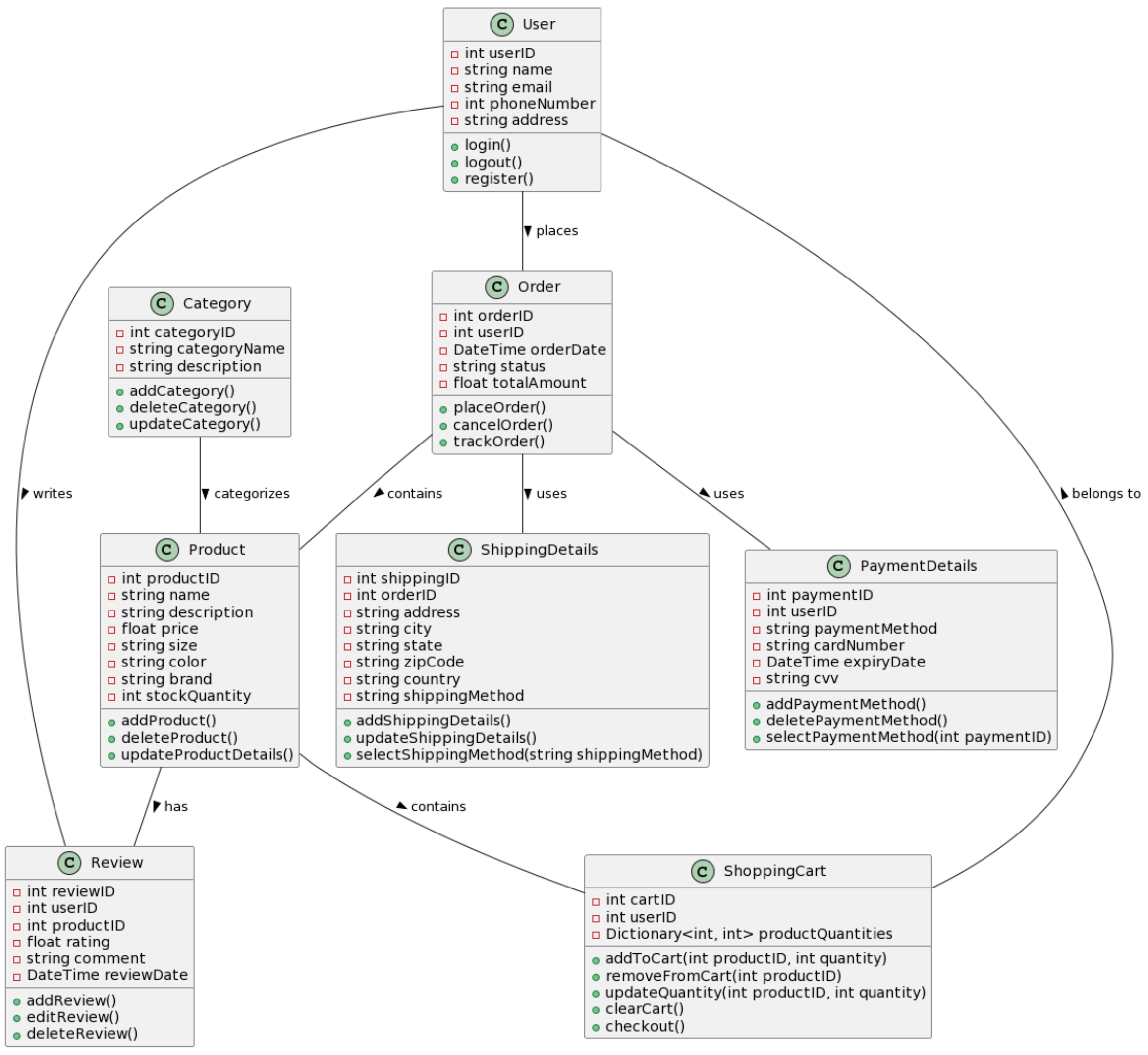


- **Description of SWA diagram**

The SWA diagram for Comazon displays how the customer will interact with the system and how each page will navigate through the system.

- The homepage communicates with the Product Page through featured clothing and categorical banners. The Search function on the home page communicates with the Product page as well. Both options send the "product searched" data to the product page.
- On the Product Page, users can click the "Add to Cart" button to add items to their Shopping Cart. This transfers the product data to be added to the customer's Shopping Cart, such as product, price, and quantity.
- The Shopping Cart allows users to view, manage, and adjust the items they have previously added. The Shopping Cart proceeds to the Checkout Page, transferring the information of the items, such as the product(s) themselves as well as the price.

- The Checkout collects shipping information, and payment methods, and displays an order summary. The Checkout directs users to the Payment Page, taking the total of all the items and charging the customer.
- The Payment page will communicate with a payment gateway to handle the transaction. The payment page will create a payment confirmation and communicate the payment status.
- A Confirmation page will display the order details, order confirmation, as well as a tracking option.
- Database
    - Homepage: Display featured products by fetching data from the Database
        - The Database holds the information needed for the homepage to update its contents
    - Product Page: Interacts with the Database to receive product information
        - The Database updates the product inventory based on purchases
    - Shopping Cart: Takes user cart contents from the Database
        - The Database will update the cart contents as customers add or remove items from their Shopping Cart
    - Checkout: Uses the Database to store shipping information as well as the order details
        - The Database updates the order history of the customer
    - Confirmation: Updates the order history in the Database
        - The Database maintains user information, product data, order history, and inventory.
- **UML Class Diagram**



- **Description of classes,attributes and operations.**
    - 1. User Class
        - Purpose: Represents both customers and sellers on Comazon.
        - Attributes:
            - int userID: A unique identifier for each user.
            - string name: The user's full name.
            - string email: The user's email address.
            - int phoneNumber: The user's contact number.
            - string address: The user's shipping address.
        - Operations:
            - login(): Authenticates the user.
            - logout(): Ends the user session.
            - register(): Registers a new user.
    - 2. Product Class
        - Purpose: Describes the clothing items listed for sale.
        - Attributes:
            - int productID: Unique identifier for each product.
            - string name: Name of the product.

        - string description: Detailed description of the product.
        - float price: Price of the product.
        - string size: Available sizes for the clothing item.
        - string color: Available colors for the clothing item.
        - string brand: Brand of the clothing item.
        - int stockQuantity: Quantity of the item in stock.
    - Operations:
        - addProduct(): Adds a new product to the listing.
        - deleteProduct(): Removes a product from the listing.
        - updateProductDetails(): Updates the details of an existing product.
- 3. Order Class
    - Purpose: Manages purchase orders placed by customers.
    - Attributes:
        - int orderID: Unique identifier for each order.
        - int userID: Identifier for the user who placed the order.
        - DateTime orderDate: The date and time the order was placed.
        - string status: Current status of the order (e.g., pending, shipped).
        - float totalAmount: Total amount of the order.
    - Operations:
        - placeOrder(): Initiates a new order.
        - cancelOrder(): Cancels an existing order.
        - trackOrder(): Provides tracking information for an order.
- 4. Review Class
    - Purpose: Enables users to post reviews and ratings for products.
    - Attributes:
        - int reviewID: Unique identifier for each review.
        - int userID: Identifier for the user who posted the review.
        - int productID: Identifier for the product being reviewed.
        - float rating: Rating given to the product.
        - string comment: Written comment about the product.
        - DateTime reviewDate: The date and time the review was posted.
    - Operations:
        - addReview(): Posts a new review.
        - editReview(): Modifies an existing review.
        - deleteReview(): Removes a review.
- 5. ShoppingCart Class
    - Purpose: Represents a user's shopping cart.
    - Attributes:
        - int cartID: Unique identifier for the shopping cart.

        - int userID: Identifier for the user who owns the cart.
        - Dictionary<int, int> productQuantities: A collection of product IDs and their quantities in the cart.
    - Operations:
        - addToCart(int productID, int quantity): Adds a product to the cart.
        - removeFromCart(int productID): Removes a product from the cart.
    - updateQuantity(int productID, int quantity): Updates the quantity of a product in the cart.
    - clearCart(): Empties the cart.
    - checkout(): Proceeds to checkout.
- 6. Category Class
    - Purpose: Categorizes products for easier browsing.
    - Attributes:
        - int categoryID: Unique identifier for each category.
        - string categoryName: Name of the category.
        - string description: Description of the category.
    - Operations:
        - addCategory(): Creates a new product category.
        - deleteCategory(): Removes an existing category.
        - updateCategory(): Updates the details of an existing category.
- 7. PaymentDetails Class
    - Purpose: Handles payment information for orders.
    - Attributes:
        - int paymentID: Unique identifier for the payment detail.
        - int userID: Identifier for the user making the payment.
        - string paymentMethod: Method of payment (e.g., credit card, PayPal).
        - string cardNumber: Credit card number (if applicable).
        - DateTime expiryDate: Expiry date of the credit card (if applicable).
        - string cvv: CVV number of the credit card (if applicable).
    - Operations:
        - addPaymentMethod(): Adds a new payment method.
        - deletePaymentMethod(): Removes an existing payment method.
        - selectPaymentMethod(int paymentID): Selects a payment method for an order.
- 8. ShippingDetails Class
    - Purpose: Manages shipping information for orders.
    - Attributes:
        - int shippingID: Unique identifier for the shipping detail.
        - int orderID: Identifier for the order being shipped.

        - string address: Shipping address.
        - string city: City of the shipping address.
        - string state: State of the shipping address.
        - string zipCode: Zip code of the shipping address.
        - string country: Country of the shipping address.
        - string shippingMethod: Method of shipping (e.g., standard, express).
      - Operations:
        - addShippingDetails(): Adds new shipping details.
        - updateShippingDetails(): Updates existing shipping details.
        - selectShippingMethod(string shippingMethod): Selects a shipping method for an order.
  - 9. Connection:
      - **User** has a relationship with **Order, Review**, and **ShoppingCart**.
      - **Product** is linked to **Review** and is included in **Order**.
      - **Category** organizes **Product**.
      - **Order** is associated with **PaymentDetails** and **ShippingDetails**.

Development plan and timeline

- **Partitioning of tasks**
  - **System Description:** Nathan Chau
  - **Brief overview of system:** Nathan Chau
  - **Software Architecture Overview:** Nathan Chau
  - **Architectural diagram of all major components:** Nathan Chau
  - **UML Class Diagram:** Jimin Li
  - **Description of classes, attributes, and operations:** Jimin Li
- **Team Member Responsibilities:** Making sure to keep each other updated
- **Estimated timeline:** 8 Hours.