UCS2313-OBJECT ORIENTED PROGRAMMING LAB

EXAMEASE-ONLINE EXAMINATION SYSTEM

Report By

Thivyatharshan (3122235001150)

Vigneshwar k (3122235001156)

Thirumoorthy j (3122235001148)

1.Problem Statement

Design an **Online Shopping System** where users can browse, search, and purchase products from a virtual store. The system should allow customers to create accounts, view product details, and add items to a shopping cart. It should support secure checkout, order management, and payment processing. The system should also have an admin interface for managing products, viewing sales reports, and handling customer inquiries.

2. Motivation for the Problem

The motivation for choosing an **Online Shopping System** stems from the growing trend of digital shopping and its convenience in today's fast-paced world. This system allows users to shop anytime and from anywhere, making it highly accessible. By creating this project, I can explore various aspects of e-commerce, like product management, payment security, and user experience. Additionally, it's an excellent opportunity to work on real-world requirements such as database management, user authentication, and transaction handling, which are relevant to modern applications. This project also provides insight into handling large data and creating a seamless user interface.

3. Scope and Limitations

Scope:

 Users can create accounts, log in, and manage profiles, with separate roles for customers and admins.

 Provide a product catalog where users can browse, search, and filter products with detailed information for each item.

 Implement a shopping cart and checkout process, allowing secure payment and address management.

 Include features for order tracking, history, and management, with tools for admins to monitor and update orders.

Limitations:

Yet to be done

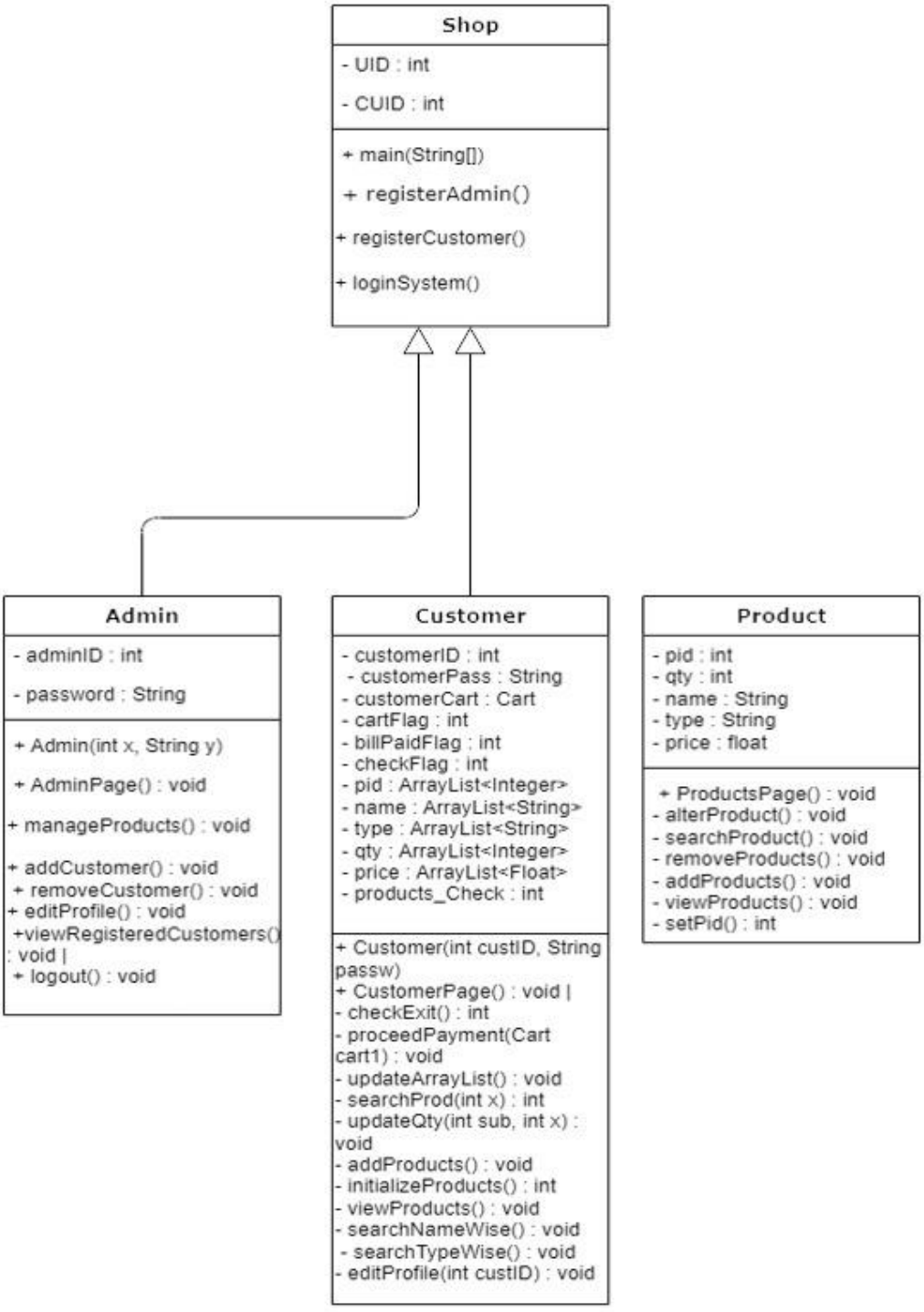
4.Design of the Solution (Class Diagram)

Payment
- customerBill : Bill - billPaid_f : int - customerCart : Cart
+ Payment(cart1 : Cart, c_name : String, b_add : String, c_phn : String) + paymentPage() : void

Bill
- bill_id : int - cust_name : String - bill_addr : String - cust_phone : String - total_amount : float - pid : ArrayList<Integer> - pname : ArrayList<String> - qty : ArrayList<Integer> - price : ArrayList<Float>
+ Bill(cname : String, badd : String, cphn : String, p_id : ArrayList<Integer>, p_name : ArrayList<String>, quant : ArrayList<Integer>, prc : ArrayList<Float>) - generateBill() : void - setBillId() : int + displayBill() : void + addToDatabase() : void

DatabaseConnection
- root : String
+ makeDatabase() : void

Cart
- pid : ArrayList<Integer> - pname : ArrayList<String> - ptype : ArrayList<String> - qpur : ArrayList<Integer> - qprice : ArrayList<Float>
+ getpid() : ArrayList<Integer> + getpname() : ArrayList<String> + getpqty() : ArrayList<Integer> + getprice() : ArrayList<Float> + addToCart(p_id : int, p_name : String, p_type : String, q_pur : int, q_price : float) : void + viewCart() : void + removeFromCart(p_id : int) : void + cancelCart() : void


Shop
- UID : int
- CUID : int
+ main(String[])
+ registerAdmin()
+ registerCustomer()
+ loginSystem()
Admin
- adminID : int
- password : String
+ Admin(int x, String y)
+ AdminPage() : void
+ manageProducts() : void
+ addCustomer() : void
+ removeCustomer() : void
+ editProfile() : void
+viewRegisteredCustomers()
: void |
+ logout() : void
Customer
- customerID : int
- customerPass : String
- customerCart : Cart
- cartFlag : int
- billPaidFlag : int
- checkFlag : int
- pid : ArrayList<Integer>
- name : ArrayList<String>
- type : ArrayList<String>
- qty : ArrayList<Integer>
- price : ArrayList<Float>
- products_Check : int
+ Customer(int custID, String
passw)
+ CustomerPage() : void |
- checkExit() : int
- proceedPayment(Cart
cart1) : void
- updateArrayList() : void
- searchProd(int x) : int
- updateQty(int sub, int x) :
void
- addProducts() : void
- initializeProducts() : int
- viewProducts() : void
- searchNameWise() : void
- searchTypeWise() : void
- editProfile(int custID) : void
Product
- pid : int
- qty : int
- name : String
- type : String
- price : float
+ ProductsPage() : void
- alterProduct() : void
- searchProduct() : void
- removeProducts() : void
- addProducts() : void
- viewProducts() : void
- setPid() : int

5. Modules Split-Up

1)Admin module

The Admin class provides essential admin functionalities:

- **Admin Access**: Allows login with adminID and password.
- **Admin Menu**: Offers options to manage products, add/remove customers, edit profile, view customers, and logout.
- **Customer Management**: Adds, views, and removes customers by updating the database.
- **Profile Editing**: Lets admins update their personal information in the database.
- **Database Operations**: Executes SQL queries to handle admin and customer data securely.

2)Bill module

The Bill class handles customer billing:

- **Initialization**: Takes customer and product details as input.
- **Generate Bill**: Calculates total amount and sets a unique bill_id.
- **Display Bill**: Prints bill details, including customer info and products.
- **Database Storage**: Saves the bill in the database with key details.

3)Cart module

The Cart class manages a customer's shopping cart:

- **Attributes**: Stores product IDs, names, types, quantities, and prices.
- **Add to Cart**: Adds a product with details to the cart.
- **View Cart**: Displays all items with their details in a structured format.
- **Remove from Cart**: Deletes a specific product from the cart and updates the database stock.
- **Cancel Cart**: Empties the cart, restoring quantities back to stock in the database.

4)Customer module

Customer Profile Management:

- Create and view personal details
- Edit contact information, address, and password

Product Viewing:

- View available products in the store

Product Search:

- Search by product name
- Search by product type

Cart Management:

- Add products to the cart
- Remove products from the cart

Cart Viewing:

- View items in the cart

Payment:

- Proceed with payment for items in the cart

Order History:

- View past orders (if implemented)

Logout:

- Logout from the customer account

5)Database connection

Connects to MySQL database.

Prompts for root password.

Creates the "onlineshop" database.

Creates tables: admininfo, custinfo, logininfo, products, bills.

Uses JDBC for database connection and operations.

6) Payment module

- Manages customer payment process.
- Initializes bill details using Cart data.
- Displays a payment page with options to:

- Pay the bill.
- View the bill.
- Exit.
- Verifies payment amount and ensures it's correct.
- Updates the database with payment details.
- Marks the bill as paid when payment is successful.

7) Products

- Manages product operations: add, remove, update, view, search.
- Allows updating product details (name, type, quantity, price).
- Provides product search by ID.
- Removes products by ID and adds new products with unique IDs.
- Displays all products in a tabular format.

8) shop

- Main Menu: Displays options for registering as admin/customer, logging in, or exiting.
- Login System: Verifies user credentials from logininfo database table and directs to either AdminPage or CustomerPage.
- Admin/Customer Registration: Collects user details, assigns unique IDs, and stores information in adminInfo, custInfo, and loginInfo tables.

6. Implementation Specifics

Programming Language: Java, with JDBC for database handling.

Data Storage: MySQL database for user data and login credentials**.**

Front-End: Command-line interface with menu-based navigation for admins and customers.

Core Logic:

- Authentication: Login validates credentials and directs users based on role (Admin or Customer).
- Registration: Separate registration for admins and customers with auto-generated unique IDs.
- Role-Based Access: Different functionalities for admins and customers after login.

Data Management:

- Database Connection: Managed by DatabaseConnection class.
- Prepared Statements: For secure data handling.
- ID Generation: setUID() and setCUID() ensure unique IDs by checking existing records.

Output Screenshots

Yet to be done

7.Object-Oriented Features Used

Inheritance:

- The User class is the parent class, with Admin and Customer as child classes. Each inherits common attributes and methods from User but provides specific functionality based on user roles.

Encapsulation:

- Private fields such as name, password, contact number, and email are used in Admin and Customer classes, with public getters and setters to control access to these properties.

Polymorphism:

- The loginSystem() method demonstrates polymorphism, where the same method call behaves differently depending on whether the user is an admin or a customer. Methods in Admin and Customer classes override the functionality of the User class.

Abstraction:

- The User class is abstract, defining essential attributes like userID, password, and userType. Specific implementation details for admins and customers are abstracted into their respective classes, leaving only the necessary structure in User.

File I/O Operations:

- Data storage and retrieval are managed through JDBC, with queries for inserting and retrieving user and login information from the MySQL database, ensuring data persistence.

8. Inference and Future Extensions

Inference: The implemented system successfully allows teachers to track students' performance, generate rank lists, and provide focused feedback on weak areas based on exam results. The use of keywords helps identify topics that require additional focus, enhancing the overall educational process.

Future Extensions:

Database Integration:

- Transition to an SQL or NoSQL database for better scalability and data handling.

Real-Time Analytics:

- Incorporate data analytics to give deeper insights into performance trends and learning patterns.

Enhanced Feedback:

- Implement AI-based analysis to offer automatic suggestions and study materials for weak topics.

Additional Exam Formats:

- Support for more question types, such as short answer and coding exercises.

Mobile App Support:

- Develop mobile-friendly applications for increased accessibility.