**SDLC**

1. **Requirements Gathering**

* User Registration and Login: Users should be able to create accounts and log into the system.
* Account Management: Users should be able to view and manage their account information.
* Market Data: Real-time display of stock market data.
* Trading Functionality: Users should be able to buy and sell stocks.
* Trade History: Users should be able to view their trading history.
* Notification System: Users should be able to set price alerts and receive notifications.
* Security: Ensure the security of the system and user data.

1. **System Design**

**User Management Module:**

* User registration and login
* User information management

**Market Data Module:**

* Fetch real-time market data
* Display stock prices and charts

**Trading Module:**

* Buy and sell stocks
* Handle trading requests

**Trade History Module:**

Store and display users' trading history

**Notification Module:**

* Set and send price alerts

**Security Module:**

* Data encryption
* Authentication and authorization

**Database design:**

* Users Table: Stores user information (user ID, username, password, email, etc.).
* Stocks Table: Stores stock information (stock ID, name, symbol, current price, etc.).
* Trades Table: Stores trade records (trade ID, user ID, stock ID, trade type, quantity, price, time, etc.).
* Notifications Table: Stores user-set notifications (notification ID, user ID, stock ID, price, notification type, etc.).

1. **Implementation**

Based on the design, we can start implementing the system. The key modules to implement include:

* User Management: Handling user registration, login, and account management.
* Market Data: Fetching and displaying real-time stock market data.
* Trading: Allowing users to place buy and sell orders and processing these transactions.
* Trade History: Recording and displaying the user's trade history.
* Notifications: Allowing users to set up and receive price alerts.
* Security: Implementing data encryption, authentication, and authorization mechanisms.

1. **Testing**

After implementing the system, we need to test it to ensure all modules and functionalities work correctly. Testing can be divided into unit testing, integration testing, and system testing.

1. **Deployment**

Once the testing is successful, we can deploy the system to a production environment. Cloud services like AWS, Azure, or other hosting providers can be used for deployment.

1. **Maintenance**

After the system is live, continuous maintenance and updates are necessary. This includes fixing any bugs, addressing security vulnerabilities, and adding new features as needed.