Assessment Project Requirement Document

Project Overview

The goal of this project is to develop a simplified stock exchange platform. The assessment is aimed at demonstrating the ability to apply your knowledge of Java, Spring Boot, and frontend development to build a functional application.

Pre-requisite:

Make all the time-bound functionalities configurable so that it's easy for demo purposes.

Functional Requirements

- **User Management:**
    - User registration and login
        - First Name
        - Last Name
        - Email
        - Phone Number
        - PAN
        - Username
    - User profile management
        - All user information can be edited except PAN which will be greyed out post registration
        - Email, Username, Phone Number should have unique checks
    - Role-based access control
        - Investor
            - Password based authentication
            - Screens for Login, Stocks watchlist, place orders, holdings, order history, reports (Get creative)
        - Admin
            - Password based authentication
            - Screens for maintaining the stocks and their prices, reports (Get creative)
- **Stock Management:**
    - Create and manage stocks (name, symbol, price)
        - Provide a feature for the admin to do a file upload of the list of companies/scrips along with their previous day closing share price into the platform which needs to be done before the trading hours begin
        - The values uploaded by the admin will only be considered for the trades being executed on that day.
        - Example: Considering the trading hours to be 9AM to 3PM, the admin needs to upload the share prices list of all the companies by 8.30AM. This list could include the existing companies in the platform and the new ones getting added onto the exchanges. All the trades happening on that day will consider the share prices from the values uploaded by the admin.
    - Display stock information (price, small description of the company, change, volume): Refer screener.in

- The values uploaded by the admin will help in displaying the current share price of the company/scrip, % change in the share price based on the previous day's value, and the volume of trades.
- T-1 values will be considered for trades on the T-day
- Example: Consider the following to be the file uploaded by the admin on 25th and 26th August 2024:

File uploaded on 25th August 2024:

Date	Company	Number of shares	Share Price	Volume
24.8.2024	Stock A	100	Rs. 75	X
24.8.2024	Stock B	100	Rs. 94	Y
24.8.2024	Stock C	100	Rs. 83	Z

File uploaded on 26th August 2024

Date	Company	Number of shares	Share Price	Volume
25.8.2024	Stock A	100	Rs. 78	X
25.8.2024	Stock B	100	Rs. 92	Y
25.8.2024	Stock C	100	Rs. 84	Z

For the trades happening on 26.8.2024 – the 25th August 2024's values should be used. (Always T-1)

- **Trading:**
  - Buy and sell stocks
    - A screen for the investors to place a buy/sell order. Investor needs to provide only the quantity as the price will automatically be fetched from the table above.
  - Order Settlement
    - The scope for the project in terms of settlement will be T-day.
    - Case 1: There needs to be exact match for the order fulfilment to be complete.
    - For example,
      - Investor A places a buy order for Stock A of 28 quantity
      - Investor B places a sell order for Stock A of 28 quantity
      - The re-conciliation happens, and the orders get matched, hence the transaction for both the investors go through.
    - At the end of the day, a complete re-conciliation of the transactions should be performed to ensure that the total number of shares per company/scrip is as per the file upload done by the admin. This process needs to be automated once all the transactions for the day gets completed.
    - Example:
      - Stock A has a total of 100 shares
      - After all the transactions related to Stock A gets completed, at the end of the day, the total shares should remain 100.
      - Another validation would be the reconciliation of the total buy and sell order transactions to ensure no invalid orders had been executed.

        - Case 2: Partial Settlement: If the buy and sell orders do not match, the next best possible match would be for a partial settlement of orders.
        - Example:
            - Investor A places a buy order for 25 shares of Stock A
            - Investor B places a sell order for 20 shares of Stock B
            - Since there are no other orders that perfectly matches these orders, the need for partial order settlement arises.
            - The sell order of Investor B goes through and gets executed, whereas the buy order of Investor A gets split – Transaction 1 of 20 shares buy – gets executed and Transaction 2 of 5 shares buy – gets cancelled.
            - Aim is to minimize the number of unfulfilled transactions.
        - Unfulfilled transactions should get cancelled at the end of the trading hours for the day.
        - Once the transaction is complete, the corresponding values should get reflected in the Investor Portfolio/Holdings screen.
        - Error Scenario – If there any errors in the re-conciliation process, the total of buy and sell quantity does not match or the total number of shares transacted does not match the value that was uploaded on to the platform by the admin, generate a report of the invalid transactions for manual intervention.
    - View order history
        - To display the list of transactions done by the investor along with their status.
    - Maintain user portfolio
        - Show the consolidated list of the investor's holdings and the total portfolio worth including the list of companies/scrips, quantity per company/scrip, invested value per company/scrip, current value per company/scrip and the total value of the portfolio.
        - Based on the current share price of the scrip, the portfolio holdings need to be updated daily.
        - Example:
            - Investor A buys 2 shares of Stock A on 25th August 2024
            - Portfolio holdings as on 25th August 2024 for Investor A is Rs. 150 (Share Price is Rs. 75 on 25th August 2024 – Refer the table above)
            - Investor A buys 2 shares of Stock B on 26th August 2024
            - Portfolio holdings for Investor A will be as follows:

Company	Quantity	Average Share Price	Invested Value	Current Value
Stock A	2	75	150	156
Stock B	2	92	184	184

Total Portfolio Value: 340

- **Reporting:**
    - Generate basic reports (e.g., user portfolio value, trading history)
    - 2 user personas:
        - For Investors, a portfolio report of their holdings and order history

        - For Admin, a consolidated report of order volume, list of transactions per day, list of companies and their corresponding share prices based on the file upload for audit.

Non-Functional Requirements

- **Performance:** The application should be responsive and able to handle a reasonable number of concurrent users (simulate multiple users).
- **Security:** Basic security measures should be implemented (e.g., password hashing, input validation).
- **Usability:** The user interface should be intuitive and easy to navigate.
- **Maintainability:** The code should be well-structured, commented, and easy to understand.

Technology Stack

- **Backend:** Java, Spring Boot, Spring Data JPA, H2 Database (in-memory database)
- **Frontend:** HTML, CSS, JavaScript, and a suitable UI framework (e.g., React, Vue.js, Angular)
    - **Note:** Given the time constraints and focus on backend development, a simpler framework like Bootstrap or Materialize CSS can be considered.

Development Environment

- **IDE:** Any Java IDE (e.g., VSCode, IntelliJ IDEA, Eclipse)
- **Database:** H2 Database (embedded)
- **Build Tool:** Maven or Gradle
- **Deployment:** The application should run locally on the developer's machine.

Assessment Criteria

- **Functionality:** Evaluate the completeness of implemented features and adherence to requirements.
- **Code Quality:** The code should be well-structured, readable, and efficient.
- **Architecture:** Evaluate the overall system design and code structure.
- **Database Design:** The database schema should be properly designed and normalized.
- **Problem-Solving:** The ability to handle unexpected issues and find solutions.

Additional Considerations

- **Data:** Simulate stock price data through a file upload.
- **Error Handling:** Implement proper error handling and user feedback mechanisms.
- **Testing:** Encourage the use of unit tests.

Note: To simplify the project and focus on core concepts, consider limiting the scope of certain features (e.g., basic user authentication, limited stock data)