Vikram Vinodh 2347263

Kalpana N 2347229

Tushar Sakhuja 2347262

CASE STUDY

stock trading simulator

**Introduction:**

In the world of finance, understanding how stock trading works is essential for investors and traders. A stock trading simulator provides a safe environment for individuals to learn and practice trading strategies without risking real money. In this case study, we will explore the development of a stock trading simulator using C. The simulator allows users to buy and sell stocks, view stock prices, and track their portfolio's performance.

**Problem statement: Stock Trading Simulator**

In this case study, we are tasked with developing a stock trading simulator that offers users the experience of buying and selling stocks, monitoring stock prices, and tracking their portfolio's performance. The goal is to create an interactive and educational platform where users can virtually invest in stocks and gain insights into the dynamics of the stock market.

**Features to be Implemented:**

1. **Stock Information:** A mechanism to provide information about available stocks. Each stock should have a symbol, name, and a pre-defined price.
2. **User Management:** Designing a system where user’s account with a username and an initial balance which are pre-defined. This balance will be used to purchase stocks.
3. **View Stock Prices:** Users will be able to view the current prices of all available stocks. Display the list of stocks along with their respective prices and symbols.
4. **Buy Stocks:** Allowing users to buy stocks using their account balance. Users will be able to select a stock and specify the quantity they want to purchase. The cost of the total purchased shares will be deducted from their balance and their stock holdings and balances will be updated accordingly.
5. **Sell Stocks:** Implementing the functionality for the users to sell the stocks they own. Users can select a stock from their portfolio and specify the quantity they want to sell. Their account balance and remaining shares will be updated accordingly.
6. **Portfolio Tracking:** Providing a mechanism for users to view their portfolio, which will include the stocks they own, the quantity of each stock, and the current value of their investments.
7. **Interactive Interface:** Developing a user-friendly interface which will guide the users through different actions, such as buying, selling, and viewing their portfolio.

**Constraints:**

* Implementing the solution using the C programming language.
* We are going to use arrays and structures to manage stocks and user accounts.
* The simulation will be text-based and will run in the console.

The goal of making this stock trading simulator is to create an engaging and educational tool that simulates the real-world stock trading experience without involving real money.

**Objectives:**

The primary objectives of this project are:

1. Developing a simulated stock trading platform to educate users about stock trading and investment strategies.
2. Implementing user account management to enable users to track their balances.
3. Allowing users to buy and sell stocks, ensuring proper balance validation and portfolio management.
4. Providing a portfolio tracking system to display users owned stocks and overall portfolio value.

**Challenges:**

Developing a stock trading simulator presents several challenges:

1. Implementing efficient data structures to store stock and user information.
2. Ensuring user account management and preventing loss of stocks and funds.
3. Handling buy and sell transactions accurately while updating user balances and portfolios.
4. Displaying stock and portfolio information in a user-friendly and informative manner.

**Implementation:**

The implementation of the stock trading simulator involves the following steps:

1. Defining data structures to store stock and user information using arrays and structures.
2. Implementing user account management functionalities.
3. Develop buying and selling mechanisms with proper balance validation.
4. Design a portfolio tracking system to display user-owned stocks and portfolio values.
5. Create a user-friendly menu system for interaction.

**Outcomes:**

Upon completing the implementation, users will be able to:

1. Manage accounts and their balances.
2. Buy and sell stocks based on their preferences.
3. Monitor their portfolio's performance and evaluate their trading strategies.
4. Gain practical experience in trading without real financial risk.

**Here's a breakdown of the code and its functionality:**

1. **Struct Definitions**: The code defines two structures, **struct Stock** and **struct User**, to store information about stocks and users. **struct Stock** contains the symbol, name, and price of a stock, while **struct User** contains the username, balance, owned stocks, stock quantities, and stock count of a user.
2. **Array Declarations**: The program declares arrays **stocks** and **users** to store stock and user data. It also initializes counters **userCount** and **stockCount** to keep track of the number of users and stocks.
3. **Functions**:
   * **createStock()**: Adds a new stock to the **stocks** array.
   * **createUser()**: Creates a new user with an initial balance and initializes their stock-related data.
   * **findUserIndex()**: Searches for a user index based on the given username.
   * **findStockIndex()**: Searches for a stock index based on the given stock symbol.
   * **buyStock()**: Allows a user to buy a specific quantity of a stock, deducting the cost from their balance and updating their stock portfolio.
   * **sellStock()**: Allows a user to sell a specific quantity of a stock, updating their balance and stock portfolio accordingly.
   * **printPortfolio()**: Displays the user's portfolio, including their balance and owned stocks.
4. **Main Function**: The main part of the program consists of a loop that continuously presents the user with options to buy stocks, sell stocks, view their portfolio, or exit the system. The loop reads the user's choice and processes it accordingly using the functions mentioned above.
5. **Initialization**: The program initializes with a few stocks (**APPLE**, **GOOGLE**, **WIPRO**, **INFOSYS**) and users (**Tushar**, **Kalpana**, **Vikram**, **Jerry**) along with their initial balances.
6. **User Interaction**: Within the loop, users can select options to buy stocks, sell stocks, view their portfolios, or exit the program. The user is prompted to provide their username and the stock symbol they want to trade. Depending on the selected option, the user may also need to input a quantity.
7. **Printing Stock Information**: The program displays available stock information and the corresponding prices before prompting the user to make a choice.
8. **Exiting the Program**: When the user chooses to exit, the loop breaks, and the program terminates.

**Future Enhancements:**

To further enhance the stock trading simulator, several improvements can be considered:

1. Integration with real-time stock price data to provide a more authentic trading experience.
2. Advanced trading features such as limit orders and stop-loss orders.
3. Incorporation of more sophisticated stock price models to better simulate market behaviour.
4. Multi-user capabilities with networking for virtual trading competitions.
5. Allowing the users to create account in real time.
6. Error handling for invalid inputs.
7. Bounds checking to prevent exceeding the maximum number of stocks a user can own or the maximum number of users and stocks in the system.

**Conclusion:**

Overall, the program simulates a basic stock trading system where users can interact with stocks, buy and sell them, and monitor their portfolios.

The stock trading simulator project provides an engaging and educational experience for users interested in understanding stock trading and allowing users to experiment with different investment strategies, the simulator bridges the gap between theoretical knowledge and practical trading. It not only offers a risk-free learning environment but also sets the foundation for more advanced trading simulations and strategies in the future.