Assignment 5

Stock Trading System using Python

Programming Languages_SWE3006_41

Deadline : May 21th, 11:59pm

Overview

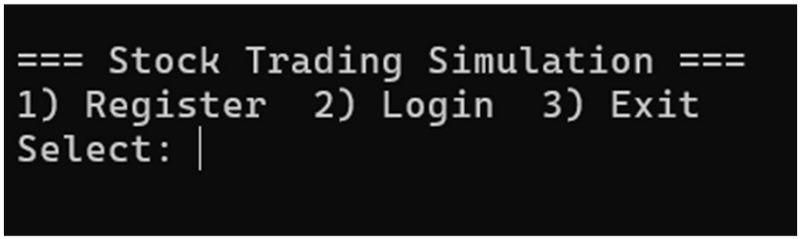
- In this assignment, you will create a multi-stock trading simulation system in **Python**. The project models a real-time stock market and user trading behavior, incorporating multiple stock tickers, automatic trading strategies, real-time price updates, and portfolio management.

- You are required to be able to use json to store and retrieve informations during the interactions.
  - You can use a .json file with python to store and retrieve the usernames, passwords and the interactions. For more detail, you can visit https://codeinstitute.net/global/blog/working-with-json-in-python/
- You must also implement a background thread using Python's threading module.
  - https://docs.python.org/ko/3.9/library/threading.html

- All transactions, market info, and user details should be stored and retrieved from json file(s).

Initial User Interface

- When you first run the program, you will have three options
  1) Register 2) Login 3) Exit



1) Register

You will input your **username**, **login password**, and **strategy**.

1. **Username**
   - The username must be unique.
   - If the username already exists in the system (users.json), display the message
     "Username already exists. Please choose another. "
2. **Password**
   - The password must satisfy the following conditions:
   - At least 8 characters
   - At least one uppercase letter
   - At least one special character from !@#$%^&*()
   - Error Message.
     Print an error message of what requirements were violated (If all three requirements are violated, print all three)
     - Password must be at least 8 characters.
     - Password must include at least one uppercase letter.
     - Password must include at least one special character (!@#$%^&*()).
   - Prompt the user to re-enter a valid password until it passes all checks

Once a valid username and password are provided, the user information must be saved to users.json

1) Register

3. Strategy Selection

- After username and password are valid, the user must choose one trading strategy (See details about strategy pg. 16)
  1) RandomStrategy
  2) MovingAverageStrategy
  3) MomentumStrategy

- Only the values 1, 2, or 3 are valid.
- If an invalid option is entered, display the message
  “Invalid selection. Try again.”

1) Register

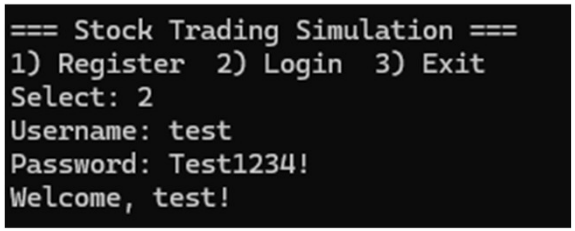
4. **On Success (Save the user information to users.json)**
   - A new user profile is created with:
     - Username
     - Password (stored in plaintext unless you choose to hash)
     - Strategy
     - Initial balance of $10,000.0
     - Empty portfolio (e.g., {})
   - Display: User '<username>' registered successfully.

*The auto field is optional and can be omitted.

2) Login

When a user selects the Login option from the initial menu, the system should

1. Prompt for Username and Password
   - Input fields: Username:, Password:
2. Validation
   - If the username does not exist or the password is incorrect:
     - Display: Invalid username or password.
     - Return to the main menu
3. On Success
   - Display: Welcome, <username>!
   - Transition to the user trading interface



3) Exit

Display the message : “Goodbye!”

```
=== Stock Trading Simulation ===
1) Register  2) Login  3) Exit
Select: 3
Goodbye!
```

Multi-Ticker Stock Market

- Simulate at least 3 stock tickers (e.g., AAPL, TSLA, GOOG)
- Each ticker has its own:
  - Price history
  - Volume history
  - Real-time price update
- Prices change every 10 seconds using random fluctuations
- All updates run on a background thread using threading
- All stock price and volume data are maintained in a shared data structure and persistently stored in **market.json**, which is updated every 10 seconds in a background thread.

Main screen

- Once a user logs in, they are presented with seven available options:
  1. View : View current price and volume for all tickers
  2. Buy : Purchase a quantity of a specific stock
  3. Sell : Sell a quantity of a specific stock
  4. Portfolio : Show the user's cash and stock holdings
  5. History : View transaction history
  6. Auto on/off : Enable or disable auto-trading using the selected strategy
  7. Logout : Return to the initial login/register screen
- If one of the seven options (view, buy, sell, portfolio, history, auto, logout) is selected, the system will immediately switch to the corresponding functionality.
- Note that choosing the "logout" option brings the user back to the initial screen.

1) View

- Command: view AAPL (or any ticker)
- Shows:
  - Current price
  - Lastest 5 price
  - Lastest 5 volumes

```
===== Select Option =====
1. view
2. buy TICKER QTY
3. sell TICKER QTY
4. portfolio
5. history
6. auto on/off
7. logout
test> 1

[AAPL] $66.93 Vol:136088
Last 5 prices: [65.73, 65.26, 64.55, 65.74, 66.93]
Last 5 volumes: [155143, 333813, 350594, 130222, 136088]

[TSLA] $106.93 Vol:387822
Last 5 prices: [104.53, 105.2, 105.74, 105.76, 106.93]
Last 5 volumes: [198570, 206873, 207752, 281521, 387822]

[GOOG] $130.18 Vol:301849
Last 5 prices: [132.84, 132.38, 130.98, 129.43, 130.18]
Last 5 volumes: [217150, 194172, 317962, 152056, 301849]
```

2) Buy

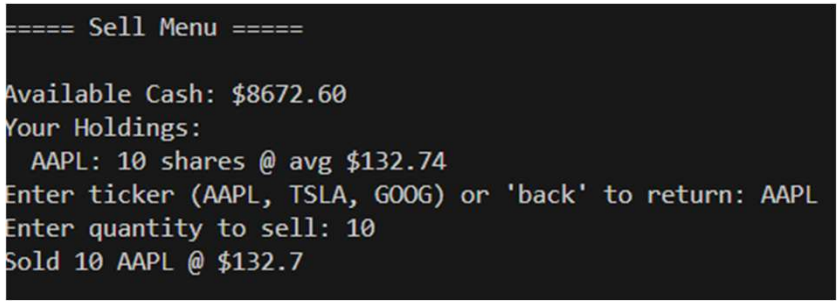
1. Display user's current balance and portfolio.
2. Prompt: "Enter stock ticker and quantity to buy”
3. Validate input:
   - Ticker must be valid (AAPL, TSLA, GOOG).
   - Quantity must be a positive integer.
4. Check sufficient balance.
5. If valid, execute trade:
   - Deduct cost from balance.
   - Add shares to portfolio.
6. Log transaction in **transactions.json**

```
===== Buy Menu =====

Available Cash: $2955.18
Your Holdings:
  TSLA: 15 shares @ avg $82.45
  GOOG: 29 shares @ avg $124.92
  AAPL: 32 shares @ avg $64.2
Enter ticker (AAPL, TSLA, GOOG) or 'back' to return: TSLA
Enter quantity to buy: 3
Bought 3 TSLA @ $38.33
```

3) Sell

1. Display user's current balance and portfolio.
2. Prompt: "Enter stock ticker and quantity to sell”
3. Validate input:
   - Ticker must be valid (AAPL, TSLA, GOOG).
   - Quantity must be a positive integer.
4. Check sufficient shares.
5. If valid, execute trade:
   - Calculates the proceeds = current price × quantity.
   - Adds the proceeds to the user's balance.
   - Deducts the shares from the user's portfolio.
   - If the number of shares reaches zero, the stock is removed from the portfolio.
   - Log transaction in **transactions.json**



4) Portfolio

- Users start with $10,000 balance and empty portfolio
- Portfolio tracks per-ticker share counts
- Evaluation function computes:
  - Current value of holdings
  - Total account value (cash + holdings)
  - Average purchase price per stock
  - Profit/Loss percentage per stock

```
===== Select Option =====
1. view
2. buy TICKER QTY
3. sell TICKER QTY
4. portfolio
5. history
6. auto on/off
7. logout
test> 4

Cash: $2954.49
TSLA: 15 @avg$75.10 -> $550.80 (-51.11%)
GOOG: 29 @avg$124.92 -> $7711.10 (+112.86%)
AAPL: 32 @avg$64.20 -> $1609.92 (-21.64%)
Total: $12826.31
```

5) History

- Lists recent transactions (most recent first)
- Displayed Information per Record:
  - Time of transaction
  - Action: BUY or SELL
  - Quantity
  - Stock ticker
  - Trade price

```
===== Select Option =====
1. view
2. buy TICKER QTY
3. sell TICKER QTY
4. portfolio
5. history
6. auto on/off
7. logout
test> 5

--- Transactions ---
2025-05-05 20:20:13 - SELL 3 TSLA @$38.1
2025-05-05 20:19:34 - BUY 3 TSLA @$38.33
2025-05-05 20:10:02 - SELL 3 TSLA @$38.8
2025-05-05 19:57:10 - BUY 3 AAPL @$53.11
2025-05-05 19:53:58 - BUY 3 AAPL @$50.82
2025-05-03 21:31:02 - BUY 4 AAPL @$66.96
2025-05-03 21:30:52 - BUY 2 TSLA @$80.3
2025-05-03 21:30:42 - BUY 3 GOOG @$121.84
2025-05-03 21:30:42 - BUY 2 AAPL @$66.98
2025-05-03 21:30:32 - BUY 1 GOOG @$121.05
2025-05-03 21:30:27 - BUY 4 GOOG @$119.97
2025-05-03 21:30:17 - BUY 1 AAPL @$65.63
2025-05-03 21:30:07 - BUY 2 TSLA @$82.12
2025-05-03 21:30:07 - BUY 2 AAPL @$66.89
2025-05-03 21:29:57 - BUY 5 AAPL @$67.64
2025-05-03 21:29:47 - BUY 5 GOOG @$123.75
2025-05-03 21:29:47 - BUY 4 AAPL @$66.53
2025-05-03 21:29:37 - BUY 1 GOOG @$126.08
```

6) Automated Trading Strategies

- Optional toggle per user: Enable/Disable auto-strategy
- Run every market tick (10 seconds)
- The strategies fetch historical and current prices for each stock from the **market.json**, which maintains a list of recent price updates used in moving average and momentum calculations.

6) Automated Trading Strategies

- Strategies:
  1. **RandomStrategy**
     - On each tick, the user has a 50% chance of making a purchase.
     - If selected, the system buys a random number of shares (1–5) of a random stock.
     - Example**:** Buy 3 shares of TSLA

  2. **MovingAverageStrategy**:
     - Calculates a **3-tick short-term average** and a **7-tick long-term average** for each stock.
     - **Buy 1 share:** If short-term average > long-term average
     - **Sell 1 share:** If short-term average < long-term average
     - Example : prices = [100, 101, 102, 103, 104, 105, 106]
       short-term = avg(104, 105, 106) = 105, long-term = avg(100~106) = 103
       short-term > long-term : Buy 1 share

  3. **MomentumStrategy**:
     - Compares the **current price** with the price **6 ticks (60 seconds)** ago.
     - **Buy:** if the current price is higher.
     - **Sell:** if the current price is lower.
     - Example : prices = [110, 109, 107, 105, 104, 103, 101]
       current price = 101, old price = 110
       current price < old price : sell 1 share

Evaluation

- You are only allowed to use **Python** for this assignment.

- You should submit a zip file contains the python code(s),storage files and a report in i-campus and name the zip file with {student_id}_Assignment5.zip (If any one of the three submissions are missing, you will be given 0 points for the assignment):
  - Code: {student_id)_Trading.py file containing the python code for the game. We will be running this file to assess.
    - You are free to use more python files to code. If so, submit these files as well and include it in the report.
  - Storage File: The actual result of the json files used for writing the report.
  - Report: You should submit 1 pdf report {student_id}_ Trading _report.pdf with:
    1. The explanation of your code.
    2. The architecture of how you stored and retrieved the information from the json file(s).
    3. The screen shots of the results from your coded program.

- **Design (10 pts)** : If you make the interface neat enough and easy to understand, you will get full points.

- **Code (80 pts)**: All pages should work as intended and have the exceptions (if there are) handled. Your points will be deducted for each mishandled exceptions.
  - Initial Screen (3 pts)
  - Register (12 pts)
  - Login (4 pts)
  - Main screen (3 pts)
  - View (4 pts)
  - Buy (8 pts)
  - Sell (8 pts)
  - Portfolio (9 pts)
  - History (9 pts)
  - Auto-Trading (12 pts)
  - Thread & JSON Usage (8 pts)

- **Report (10 pts)**

Any type of plagiarism, code sharing and usage of chat models like Chat-GPT will result in your final grade being F.

QnAs

Please use the Discussion session in I-campus for any questions about the assignment.