Project: Stock Monitoring System:

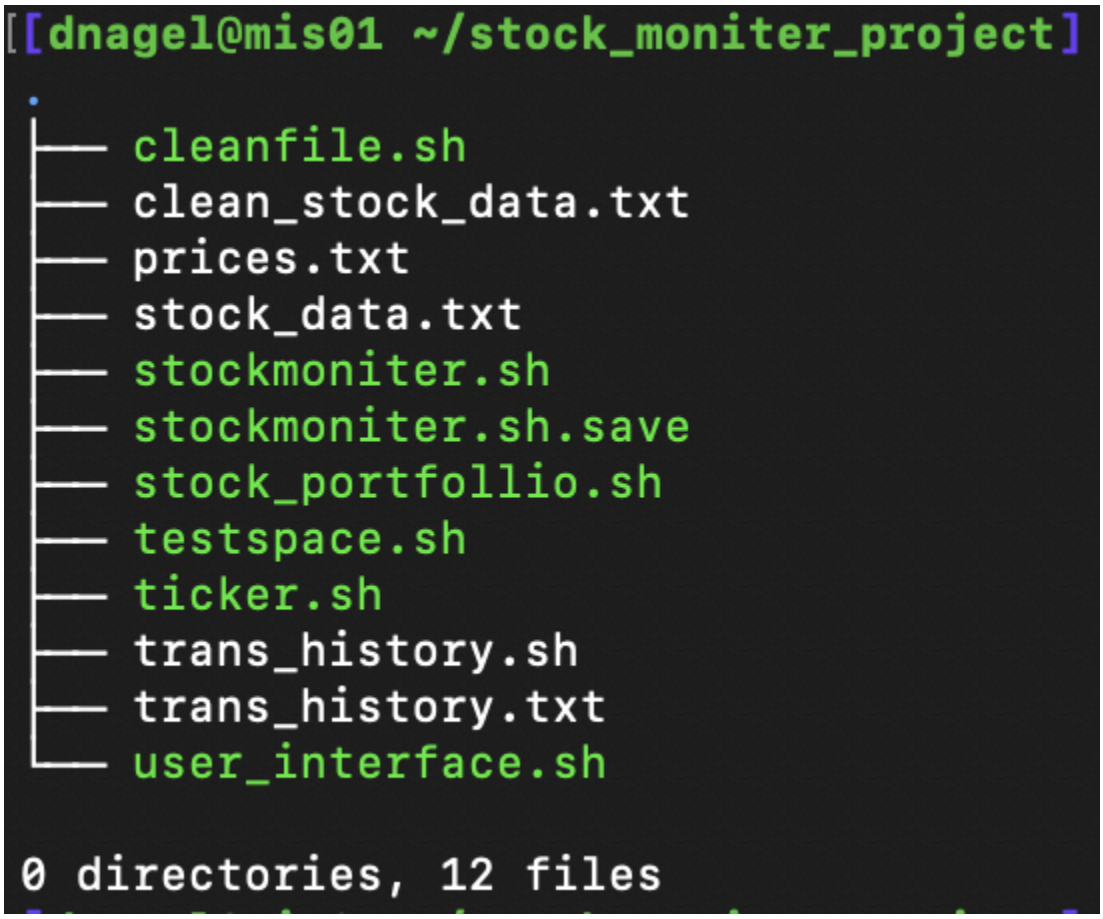
By Declan Nagel and Albert Lai

Features:

- User Log In
- Gives the user multiple options to choose from and accepts various user input for each.
- Option 1: Calculates optimal time to buy and sell based off of percentage change of prices
- Option 2: Calculates optimal time to buy and sell based on minimum and maximum values.
- Option 3: Outputs transaction history of the user.
- Option 4: outputs total value of bought stock, sold stock, and value of profit.

Code:

Directory Tree:



User Interface:

GNU nano 2.9.8 user_interface.sh

```
#!/bin/bash

load() {
        for i in $(seq 1 100); do
        echo -ne "\r$i%"
        sleep 0.01
        done
        echo " "
}
#user interface entry
echo "Welcome to the Declan and Albert Stock Monitoring System"
sleep 0.02
echo "Please enter username to log in"
read user_entry

while true; do
        if [[ "$user_entry" = "$(whoami)" ]]; then
                echo "Log In Succcessful"
                break
        else
                echo "error. Incorrect Username"
                echo "try again"
                read user_entry
        fi
done

echo "-------------------------------------------------"
echo "stock monitoring preferences:"
sleep 0.02
echo "-------------------------------------------------"
echo "select stock monitoring method (enter number of option): "
echo "1:percentage change    2:min and max     3:stock transaction history"     "4:Current stock portfollio"
read usr_option
sleep 0.02
# based off of user input, one of the .sh files in the if statements will run an option.
if [[ "$usr_option" -eq 1 ]]; then
        # gathering data from user for option 1:
        echo "enter stock name: "
        read stock
        sleep 0.02
        echo "loading stock data"
        load
        echo "At what price percentage change to buy stock: "
        echo "recommended values: 3%    5%    7%"
        read user_perc_change
        echo "At what price percentage change to sell stock: "
        echo "recommended valued: 1%    2%      3%"
        read user_perc_change_sell
        load
        echo "How often do you want the systen to check for stock price change (in seconds): "
        read eval_freq
        load
        echo "Enter the max amount of runs: "
        read maxruns
        load
        ./stockmoniter.sh "$stock" "$user_perc_change" "$eval_freq" "$maxruns" "$user_perc_change_sell"
elif [[ "$usr_option" -eq 2 ]]; then
        # gathering data from user for option 2:
```

```
elif [[ "$usr_option" -eq 2 ]]; then
        # gathering data from user for option 2:
        echo "enter stock name: "
        read stock
        sleep 0.02
        echo "loading stock data"
        load
        echo "enter the number of runs: "
        read max
        sleep 0.02
        echo "enter the refresh time between runs: "
        read interval
        sleep 0.02
        echo "enter the minimum current price where the program starts buying: "
        read minbprice
        sleep 0.02
        echo "enter the minimum current price where the program starts selling: "
        read minsprice
        sleep 0.02
        ./testspace.sh "$stock" "$max" "$interval" "$minbprice" "$minsprice"
elif [[ "$usr_option" -eq 3 ]]; then
        # runs option 3
        echo "loading stock transaction history"
        load
        echo "$(cat trans_history.txt)"
        echo "-------------------------------------"
        echo "Enter 1 to exit"
        read usr_input
        if [ "$usr_input" -eq 1 ]; then
                ./user_interface.sh
```

```
        fi
elif [[ "$usr_option" -eq 4 ]]; then
        # runs option 4
        echo "loading Current Stock Portfollio"
        load
        ./stock_portfollio.sh
fi
```

Option 1:

```
#!/bin/bash

progress() {
        for i in $(seq 1 100); do
                echo -ne "\r$i%"
                sleep 0.01
        done
}
# gathers user input values from user_interface.sh
stock=$1
user_perc_change=$2
eval_freq=$3
maxruns=$4
user_perc_change_sell=$5
echo "stock value: $stock"
echo "user perc change value: $user_perc_change"
sleep 0.02


count=0

while true; do
        # getting and cleaning data
        ./ticker.sh $stock >> stock_data.txt 2>/dev/null
        ./cleanfile.sh < stock_data.txt > clean_stock_data.txt

        # retrieving current price and time
        add_price=$(cat clean_stock_data.txt | grep -o '\$[0-9]*\.[0-9]*' | head -1 | tr -d '$')
        add_time=$(date '+%Y-%m-%d %H:%M:%S')

        # appending current price and time to prices.txt
        echo "$add_price $add_time" >> prices.txt

        mapfile -t prices < prices.txt
        # gathers previous price under the condition below and current price
        if [[ $count -gt 0 ]]; then
                prev_price=$(echo "${prices[count-1]}" | awk '{print $1}')
                prev_time=$(echo "${prices[count-1]}" | awk '{print $2, $3}')
                echo "prev price: $prev_price prev time: $prev_time"
        else
                prev_price=0
                prev_time=0
        fi

        cur_price=$add_price
        cur_time=$add_time
        echo "cur price: $cur_price cur time: $cur_time"
        holding=0
        # checks if user should buy
        if [[ $prev_price != 0 ]]; then
                perc_change=$(echo "scale=2; (($cur_price - $prev_price)/$prev_price) * 100" | bc)
                echo "percentage change: $perc_change%"
                if (( $(echo "$perc_change > $user_perc_change" | bc -l) )) && [[ $holding -eq 0 ]]; then
                        holding=1
                        holding=1
                        purc_price=$cur_price
                        purc_time=$cur_time
                        echo "$purc_price $purc_time B" >> trans_history.txt
```

```
 GNU nano 2.9.8                                                    stockmoniter.sh

                fi
        else
                echo "no value for prev price so no percetage value"
        fi
        # under the condition below, sells the stock the user bought
        if [[ $(cat trans_history.txt | wc -l) -gt 0 && $holding -eq 1 ]]; then
                perc_changeS=$(echo "scale=2; (($cur_price - $purc_price)/$purc_price) * 100" | bc)
                echo "percentage change: $perc_change%"
                if (( $(echo "$perc_changeS <= $perc_change_sell" | bc -l) )); then
                        sell_price=$cur_price
                        purc_time=$cur_time
                        holding=0
                        echo "$sell_price $sell_time S" >> trans_history.txt
                fi
        fi


        count=$((count+1))

        if [ "$count" -eq "$maxruns" ]
        then
                break
        fi
        sleep $eval_freq
done
echo "----------------------------------------"
echo "Enter 1 to exit"
read usr_input
if [ "$usr_input" -eq 1 [; then
        echo "exiting..."
        progress
        ./user_interface.sh
fi

```

Option 2:

```
#!/bin/bash

runs=1
tokens=0
revenue=0
cost=0
profit=0

read -p "Enter the stock to test: " stock
read -p "Enter the number of runs: " max
read -p "Enter the refresh time between runs: " interval
read -p "Enter the price threshold the current price should be below for the program starts buying: " minbprice
read -p "Enter the price threshold the current price should be larger than for the program starts selling: " minsprice

while [ "$runs" -le "$max" ]; do
    ./ticker.sh "$stock" >> stockdata.txt 2>/dev/null
    ./cleanfile.sh < stockdata.txt > cleandata.txt
    sed -n '6,6p' cleandata.txt >> storedata.txt
    runprice=$(cat storedata.txt | awk '{print $5}' | sed 's/[^0-9.]//g')
    echo " "
    if [[ $(echo "$minbprice > $runprice" | bc -l) -eq 1 ]]
    then
            tokens=$((tokens + 1))
            cost=$(echo "$cost + $runprice" | bc -l)
            echo "Buying stock, token count: $tokens, transaction cost: $runprice"
    fi
    sell_check=$(echo "$minsprice < $runprice" | bc -l)
    if [ "$sell_check" -eq 1 ] && [ "$tokens" -ge 1 ]
    then
        tokens=$((tokens - 1))
        revenue=$(echo "$revenue + $runprice" | bc -l)
        echo "Selling stock, token count: $tokens, transaction gain: $runprice"
    elif [ "$tokens" -lt 1 ]
    then
        echo "No more stock to sell"
    fi
    echo "add stock data, $runs times"
    runs=$((runs + 1))
    if [ "$runs" -le "$max" ]
    then
            sleep "$interval"
    fi

    > stockdata.txt
    > cleandata.txt
    > storedata.txt
done

profit=$(echo "$revenue - $cost" | bc -l)
echo "The result in this time period is: $profit"
echo "The remaining shares of stock you have is: $tokens"

> stockdata.txt
> cleandata.txt
> storedata.txt

~
~
~
~
~
~
~
~
~
~
"tester2.sh" 56L, 1722C
```

Option 3:

```
elif [[ "$usr_option" -eq 3 ]]; then
        # runs option 3
        echo "loading stock transaction history"
        load
        echo "$(cat trans_history.txt)"
        echo "-------------------------------------"
        echo "Enter 1 to exit"
        read usr_input
        if [ "$usr_input" -eq 1 ]; then
                ./user_interface.sh
        fi
```

Option 4:

```
 GNU nano 2.9.8                                                    stoc

#!/bin/bash

load() {
        for i in $(seq 1 100); do
                echo -ne "\r$i%"
                sleep 0.01
        done
        echo " "
}

echo "-------------------------------------"
echo "My Stock Portfollio"
echo " "
#initialize profit and get bought and sold stock values
profit=0
boughtStock=$(awk 'BEGIN{sumB=0} {if ($4=="B") sumB+=$1} END{print sumB}' trans_history.txt)
soldStock=$(awk 'BEGIN{sumS=0} {if ($4=="S") sumS+=$1} END{print sumS}' trans_history.txt)
profit=$(echo "$boughtStock - $soldStock" | bc)

echo "Total value of bought stock: $boughtStock"
echo "Total value of sold stock: $soldStock"
echo "Profit: $profit"
echo "-------------------------------------"
echo "Enter 1 to exit"
read usr_entry

if [ "$usr_entry" -eq 1 ]; then
        echo "exiting..."
        load
        ./user_interface.sh
fi


```

Example data:

```
[dnagel@mis01 ~/stock_moniter_project] cat prices.txt | head -10
239.07 2025-03-09 14:00:13
239.07 2025-03-09 14:00:15
239.07 2025-03-09 14:00:17
239.07 2025-03-10 16:15:33
239.07 2025-03-10 16:15:35
239.07 2025-03-10 16:15:37
239.07 2025-03-10 16:27:00
239.07 2025-03-10 16:27:02
239.07 2025-03-10 16:27:04
239.07 2025-03-10 16:33:22
```

```
[[dnagel@mis01 ~/stock_moniter_project] cat trans_history.txt | head -10
319.85 2025-03-12 09:38 B
319.84 2025-03-12 09:48 S
319.37 2025-03-12 09:57 B
319.1 2025-03-12 09:59 S
319.0 2025-03-12 10:00 B
318.58 2025-03-12 10:08 S
318.6 2025-03-12 10:18 B
318.88 2025-03-12 10:19 S
318.55 2025-03-12 10:27 B
318.85 2025-03-12 10:37 S
```