Capgemini

Technical Analysis Stock Screener

Low Level Design – High Level Design

History:

Technical Analysis Stock Screener				
Date	**Version**	**Author**	**Brief Description of Changes**	**Approver**
28-09-2022	1.0	Group-3		

Index

1. Introduction

The Technical Analysis Stock Screener is a system that allows the user to enter into the system and check the available listed Stocks and their recommendations. Here the recommendations are done based on Technical Analysis. The user can see the recommendation based on those suggestions the user can do any of these like Buy, Sell and Hold the Stocks.

1.1 Intended Audience: -

The target audience set for this project can be identified as Equity Stock traders / Investors.

1.2 Project Purpose: -

The Technical Analysis Stock Screener is a project that helps us understand the basic concepts of functions like file handling, and data structure. Here this project mainly displays the list of stocks to the user and give recommendations based on the averages calculated between the latest 10 days average stock to 50 days average stock. Based on these averages It provides the recommendations like Buy, Sell and Hold of the stocks.

1.3 Key Project Objectives: -

a. Allow the User to enter the System

b. Can see the list of available stocks

c. updating Functionality upto latest 20 stocks

d. Displays all the records of The stock

e. calculating the Average close Stocks.

f. Suggest the recommendations like Buy, Sell or Hold.

1.4 Project scope : -

This project aims to create the development of an Technical Analysis Stock Screener ,Which takes the Stock information from various Stocks, adds it to the database and processes the Average of latest Stock records, And give recommendations based on the technical Analysis.

2. Design Overview: -

- **Technical Analysis Stock Screener Comprises of the following modules:**

Name of the Module	List of Stocks Module
Handled by	
Description	It display list of stocks available

Name of the Module	Main Menu Module
Handled by	
Description	Consist of List of choices and exit option

Name of the Module	Sub Menu Module
Handled by	
Description	It consist the information about selected choice

Name of the Module	Exit Module
Handled by	
Description	It returns to the main menu from sub menu

2.1 Design Objectives:

- Add No of stocks into the data.
- User can select which stock data want to see.
- It will Calculate averages of the stock records.
- Based on average it show recommendations to the user.
- Recommendations are buy, sell and hold.
- Then user can take decision based on technical Analysis of stocks.

2.2 Design Alternative: -

We have used a linked list structure to store data i.e.. Stock data files, to find averages and for recommendations

2.3 User Interface Paradigms: -

This project Directly perform the Technical analysis on stocks and give recommendations to the user based on the analysis.

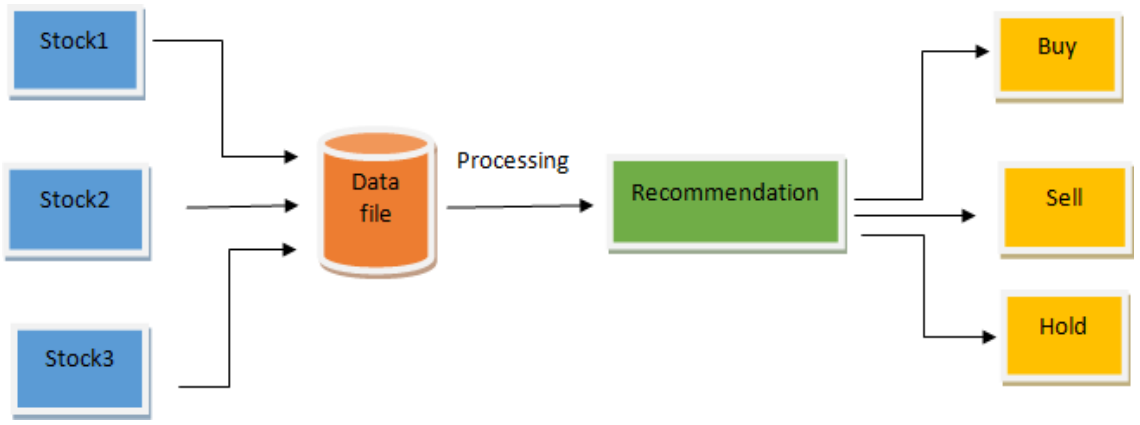
2.4 Validation: -

- User cannot enter more than one choice at a time
- User cannot perform any operations in the stocks data.

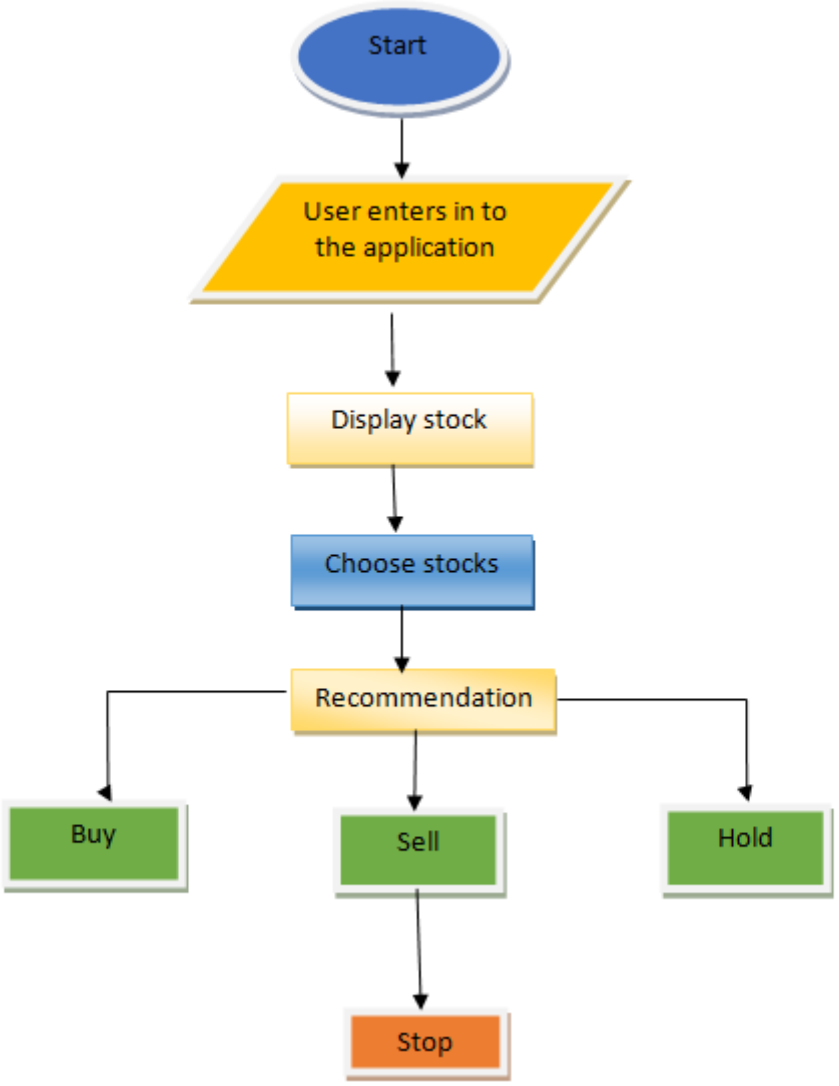
2.5 System Requirements:-

- Language Used: C language
- Tools Used : Make,Valgrind
- Compiler : Gcc
- Environment : Linux

3.DETAILED SYSTEM DESIGN:



3.1 Flow Chart of the application



4.Functional overview:

4.1 Following Header files are include:

- #include<stdio.h>
- #include<stdlib.h>
- #include<string.h>
- #include<main.h>
- #include<struct.h>

4.2 Following Functions are included:

A. selectChoice( ) :

This Function mainly used to give the choices to the user so that based on the list of choices the user can select from it.

B. loadStockData( ) :

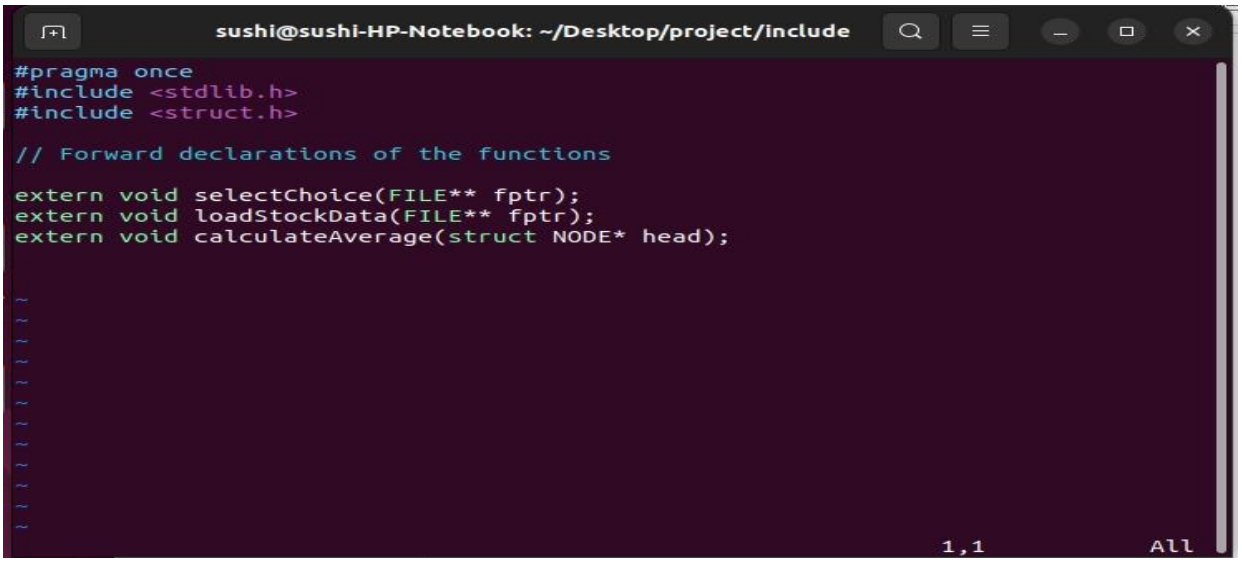
This function is used to load the CSV file into the linked list. By using the file pointer we read the CSV file into the linked list.

C. showRecommendations( ) :

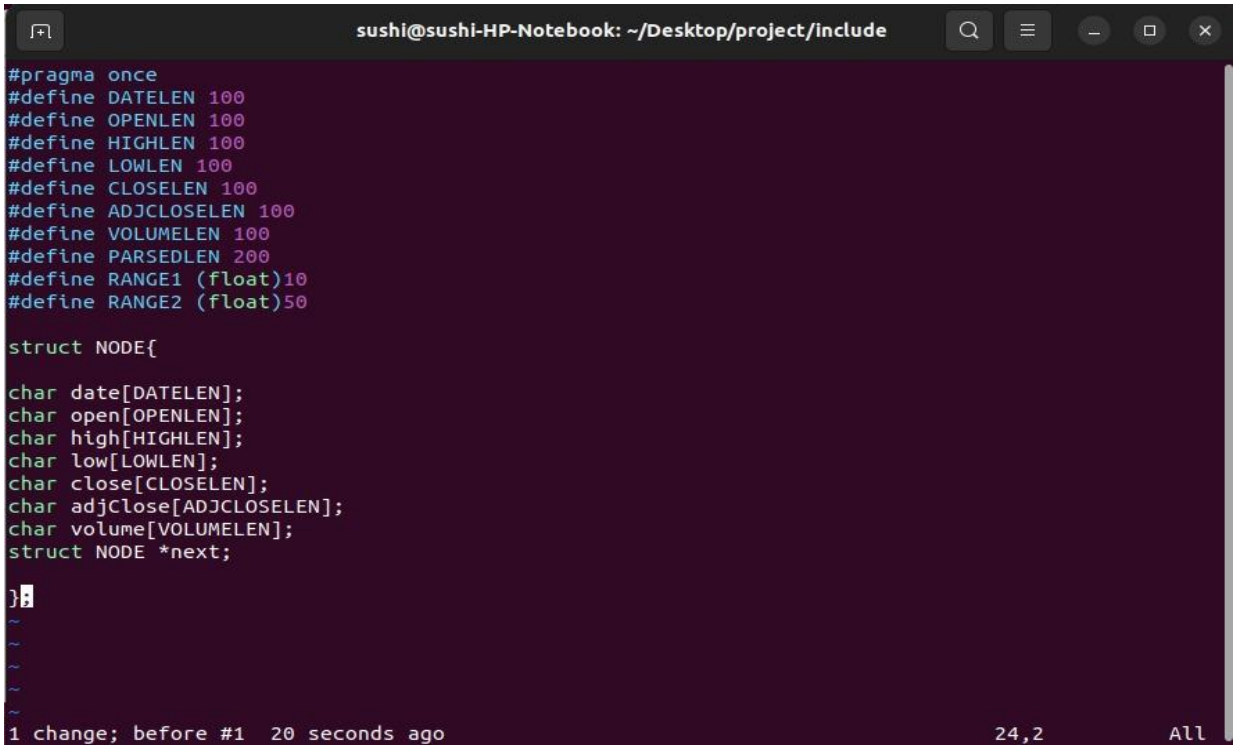
This Function is used to give recommendations to the stock holders based on the average calculated. It give recommendations like buy, sell or hold.

5.Coding:

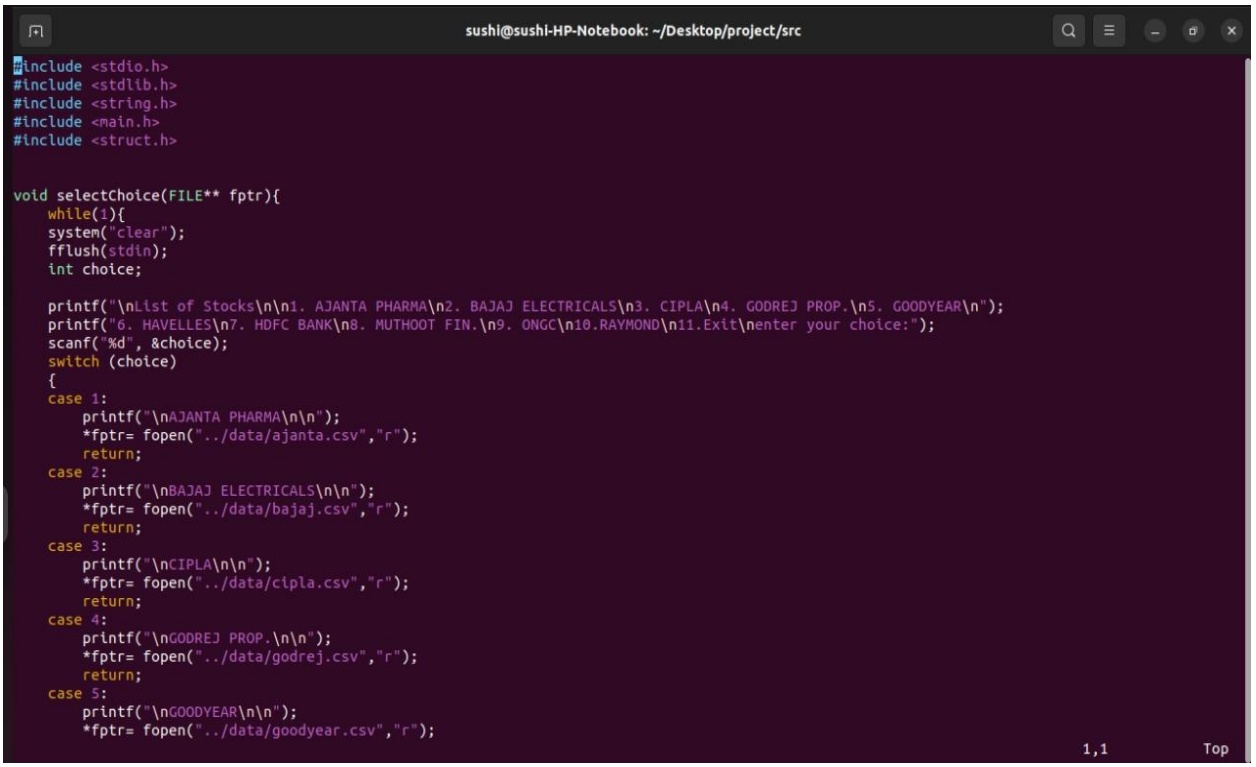
Main.h:

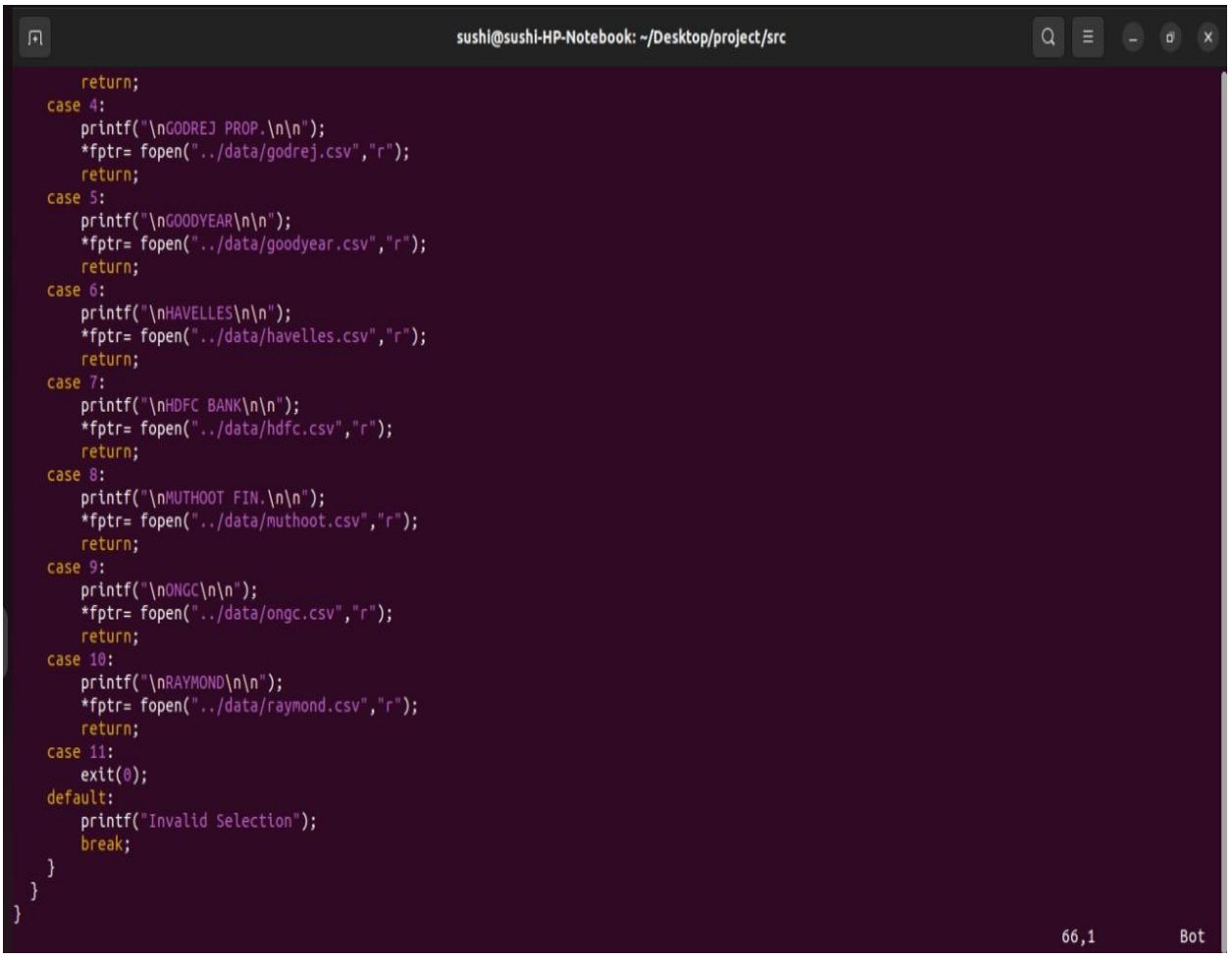


Struct.h:

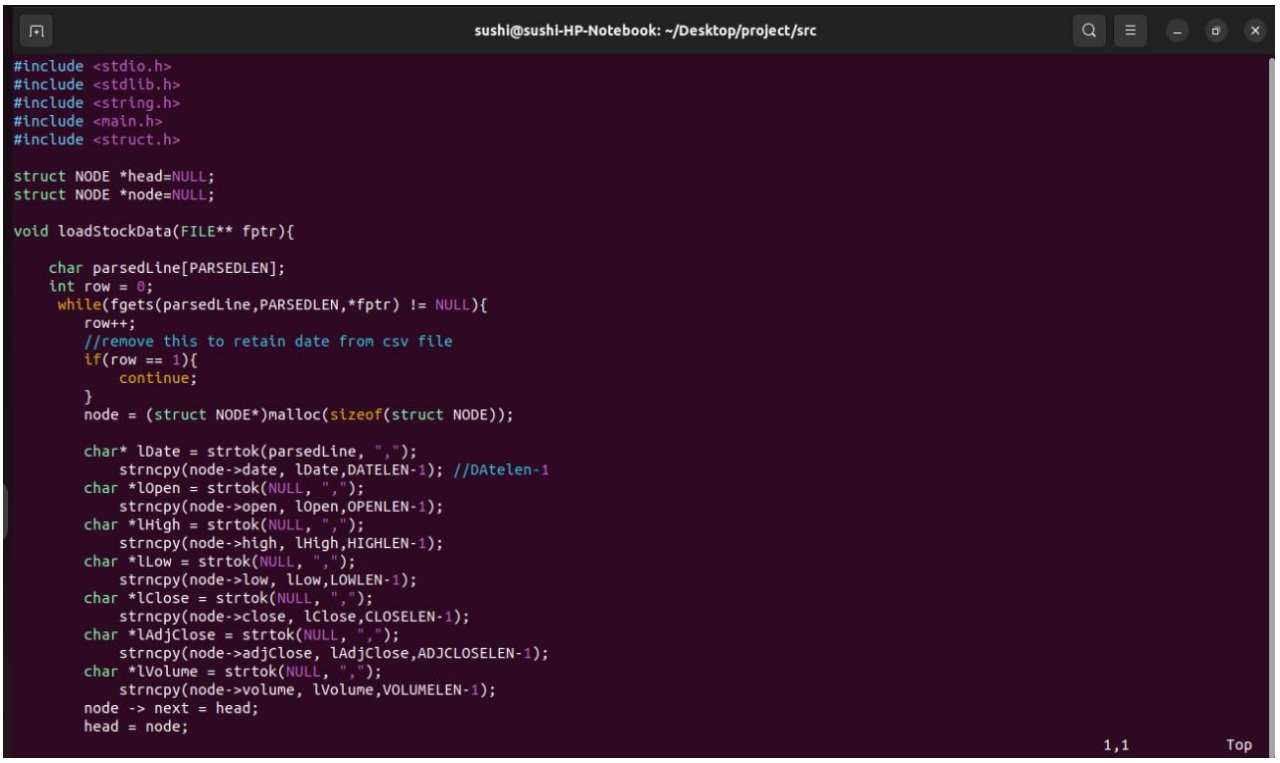


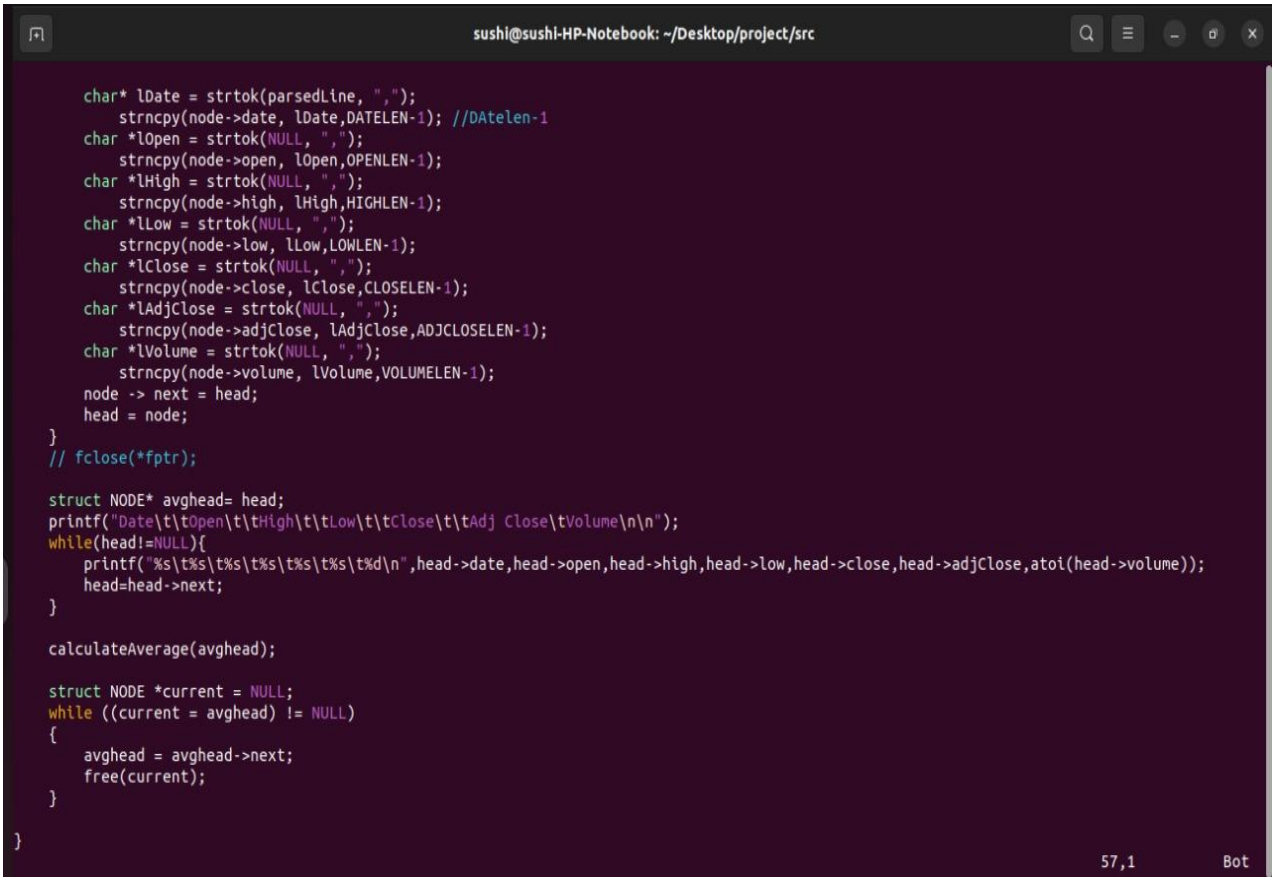
SelectChoice ( ):



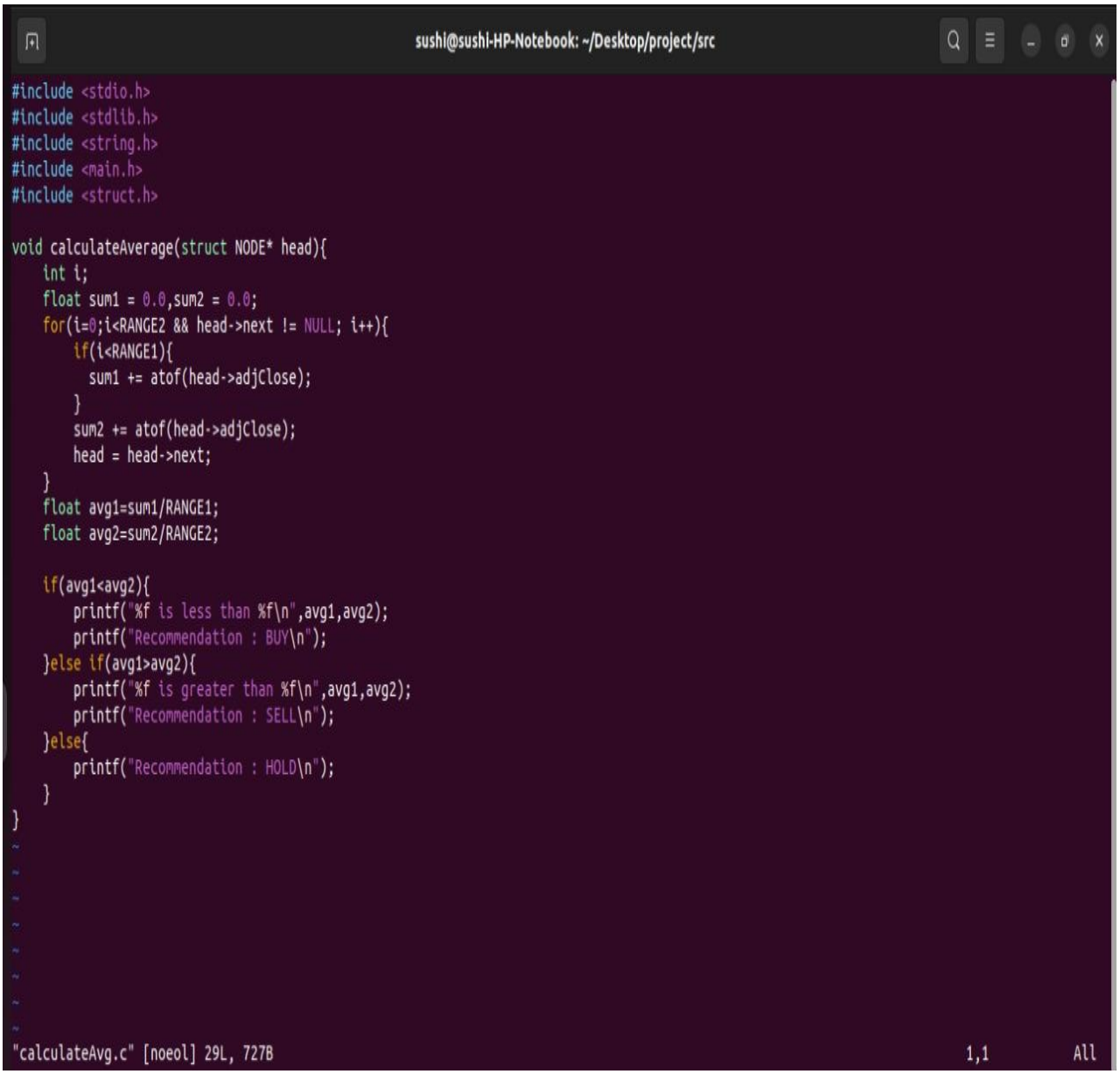


loadStockData ( ):

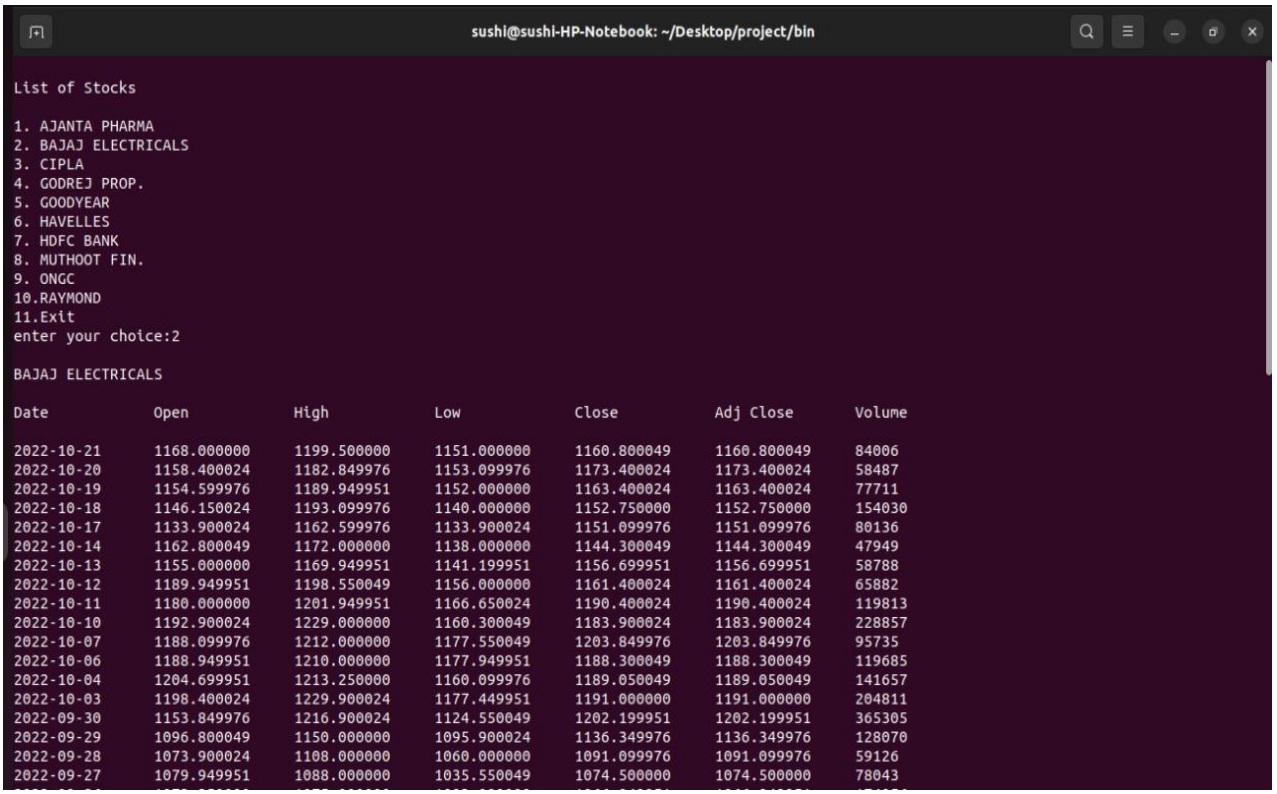




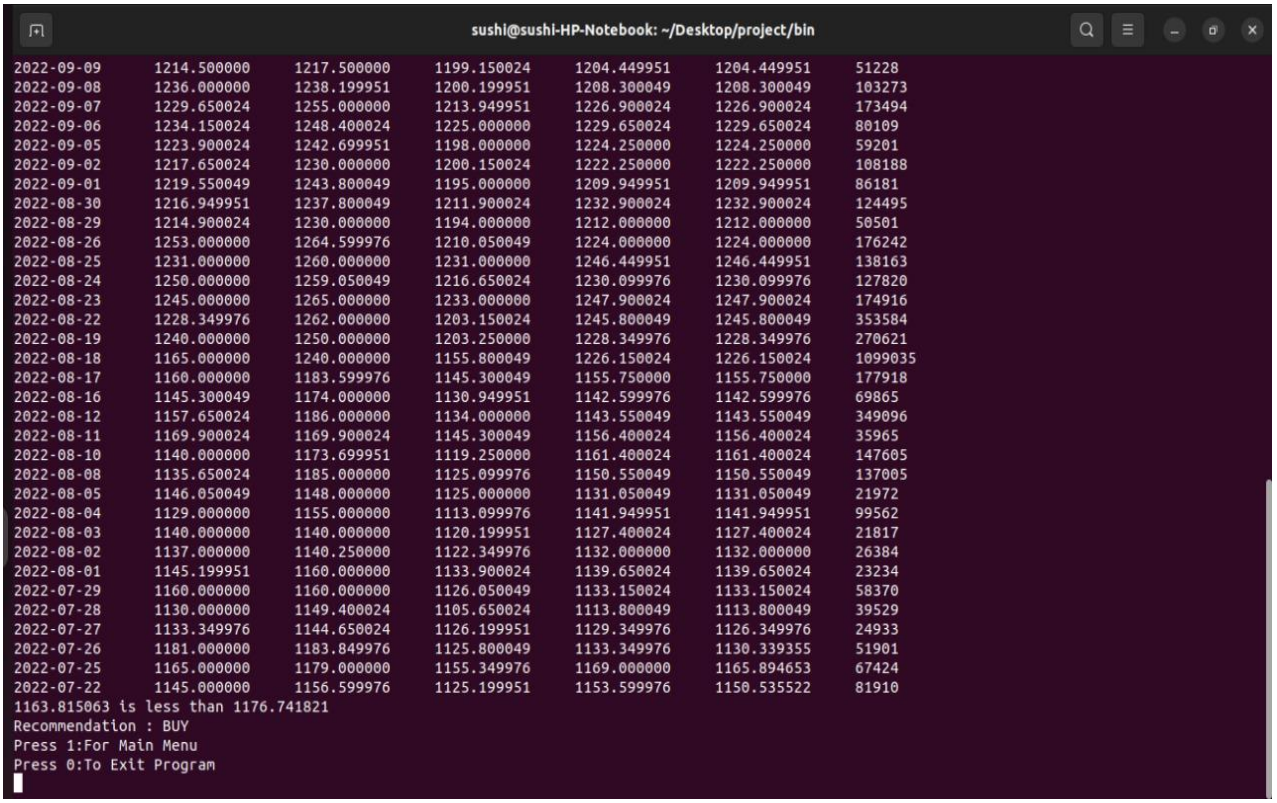
showRecommendations ( ):



List of stocks:



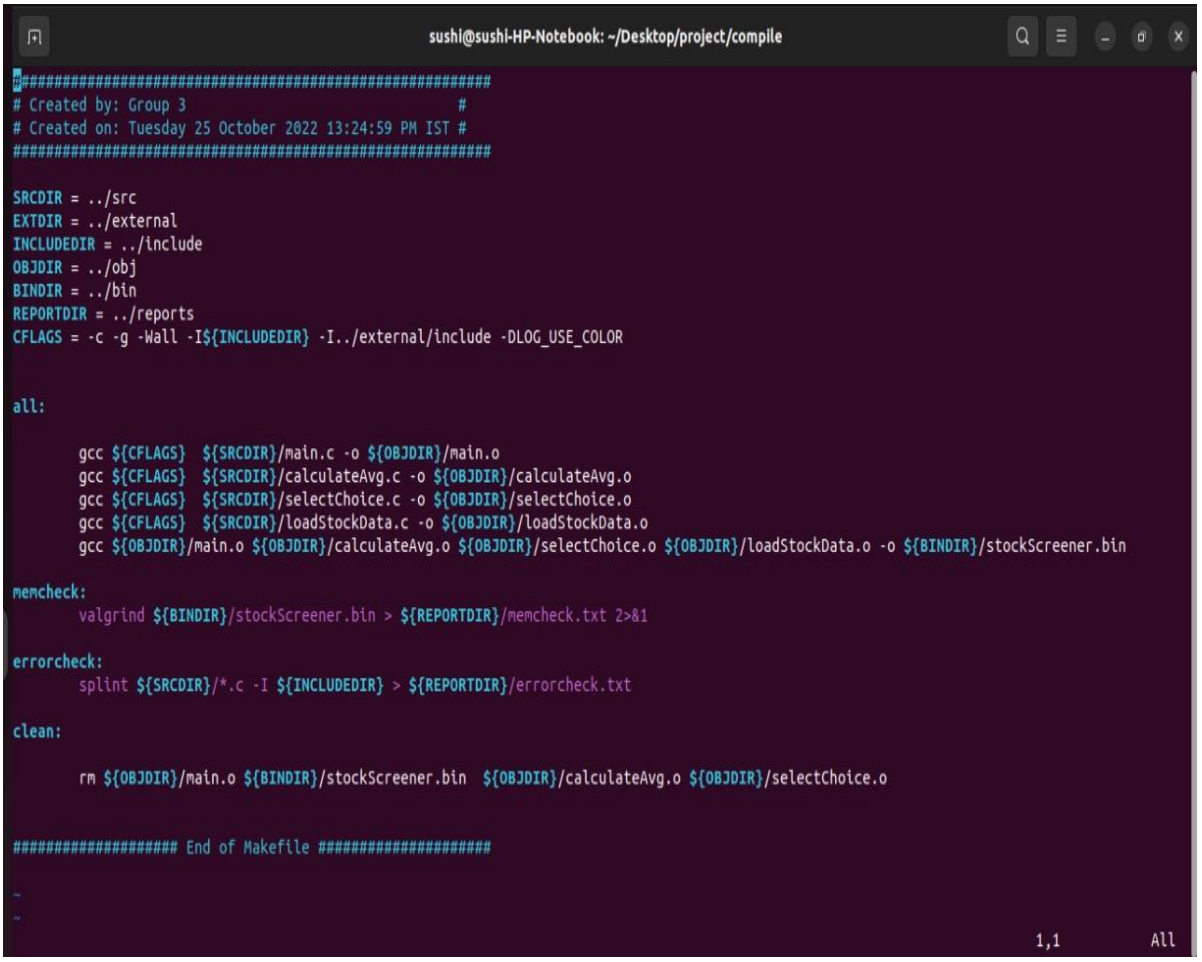
Output:



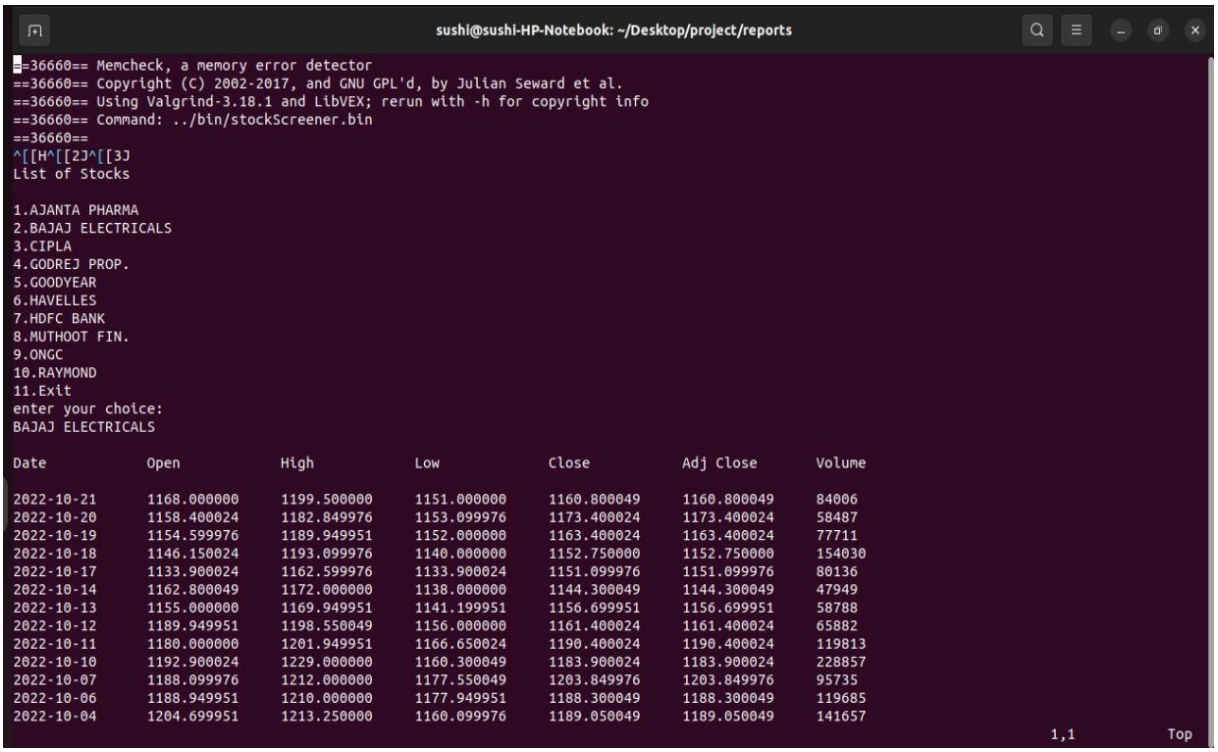
6.Tools Used:

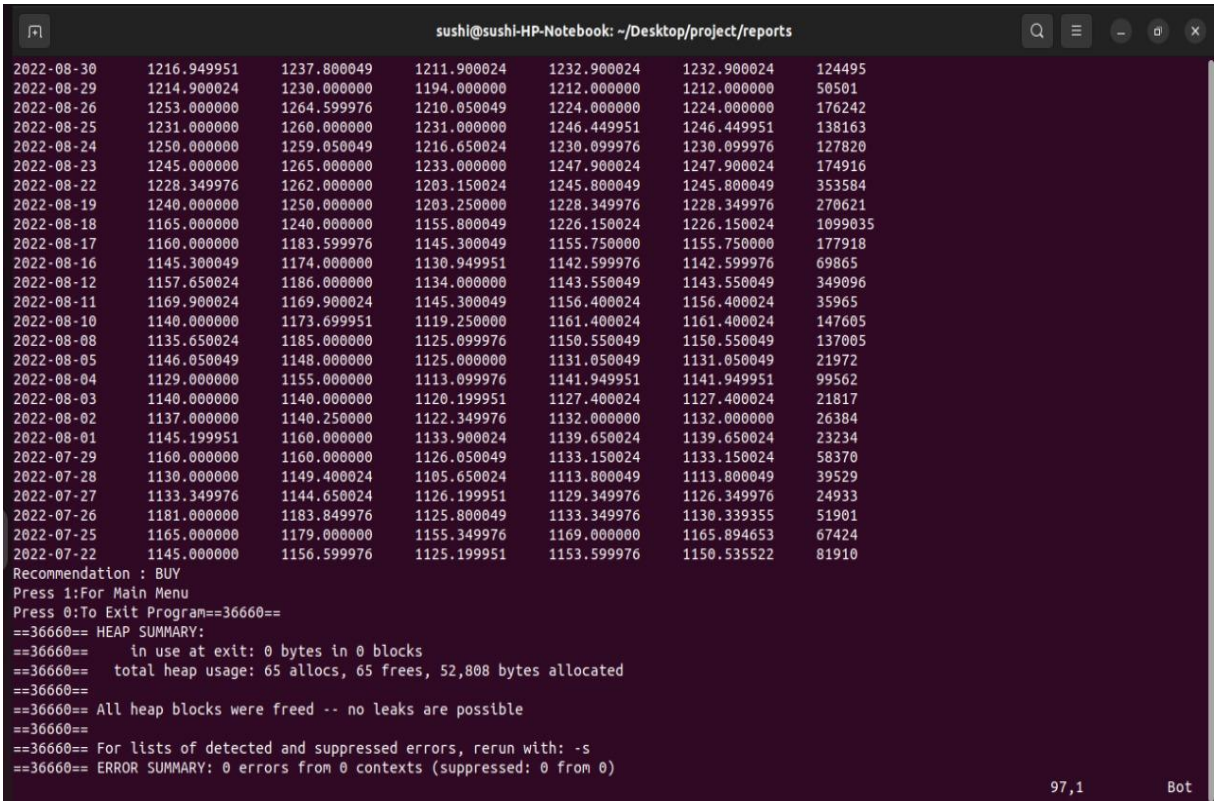
1.Make file:

Make file is a set of commands with variable names and targets to create object files and to remove them. In a single makefile we can create multiple targets to compile and to remove object, binary files. We can compile our project any number of times using make file.

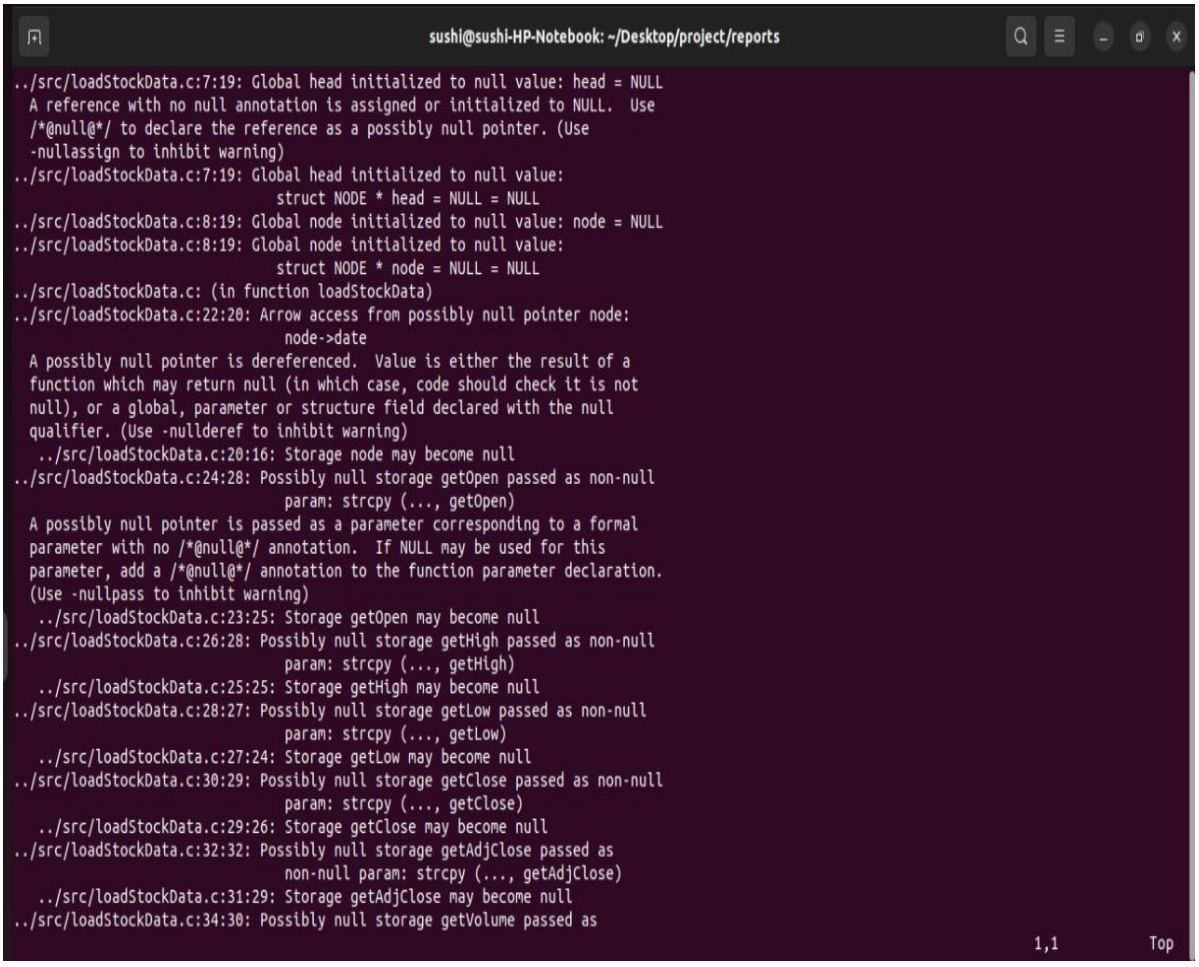


2.Valgrind:


sushi@sushi-HP-Notebook: ~/Desktop/project/reports
==36660== Memcheck, a memory error detector
==36660== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==36660== Using Valgrind-3.18.1 and LibVEX; rerun with -h for copyright info
==36660== Command: ../bin/stockScreener.bin
==36660==
^[[H^[[2J^[[3J
List of Stocks
1.AJANTA PHARMA
2.BAJAJ ELECTRICALS
3.CIPLA
4.GODREJ PROP.
5.GOODYEAR
6.HAVELLES
7.HDFC BANK
8.MUTHOOT FIN.
9.ONGC
10.RAYMOND
11.Exit
enter your choice:
BAJAJ ELECTRICALS
Date Open High Low Close Adj Close Volume
2022-10-21 1168.000000 1199.500000 1151.000000 1160.800049 1160.800049 84006
2022-10-20 1158.400024 1182.849976 1153.099976 1173.400024 1173.400024 58487
2022-10-19 1154.599976 1189.949951 1152.000000 1163.400024 1163.400024 77711
2022-10-18 1146.150024 1193.099976 1140.000000 1152.750000 1152.750000 154030
2022-10-17 1133.900024 1162.599976 1133.900024 1151.099976 1151.099976 80136
2022-10-14 1162.800049 1172.000000 1138.000000 1144.300049 1144.300049 47949
2022-10-13 1155.000000 1169.949951 1141.199951 1156.699951 1156.699951 58788
2022-10-12 1189.949951 1198.550049 1156.000000 1161.400024 1161.400024 65882
2022-10-11 1180.000000 1201.949951 1166.650024 1190.400024 1190.400024 119813
2022-10-10 1192.900024 1229.000000 1160.300049 1183.900024 1183.900024 228857
2022-10-07 1188.099976 1212.000000 1177.550049 1203.849976 1203.849976 95735
2022-10-06 1188.949951 1210.000000 1177.949951 1188.300049 1188.300049 119685
2022-10-04 1204.699951 1213.250000 1160.099976 1189.050049 1189.050049 141657
1,1 Top


sushi@sushi-HP-Notebook: ~/Desktop/project/reports
2022-08-30 1216.949951 1237.800049 1211.900024 1232.900024 1232.900024 124495
2022-08-29 1214.900024 1230.000000 1194.000000 1212.000000 1212.000000 50501
2022-08-26 1253.000000 1264.599976 1210.050049 1224.000000 1224.000000 176242
2022-08-25 1231.000000 1260.000000 1231.000000 1246.449951 1246.449951 138163
2022-08-24 1250.000000 1259.050049 1216.650024 1230.099976 1230.099976 127820
2022-08-23 1245.000000 1265.000000 1233.000000 1247.900024 1247.900024 174916
2022-08-22 1228.349976 1262.000000 1203.150024 1245.800049 1245.800049 353584
2022-08-19 1240.000000 1250.000000 1203.250000 1228.349976 1228.349976 270621
2022-08-18 1165.000000 1240.000000 1155.800049 1226.150024 1226.150024 1099035
2022-08-17 1160.000000 1183.599976 1145.300049 1155.750000 1155.750000 177918
2022-08-16 1145.300049 1174.000000 1130.949951 1142.599976 1142.599976 69865
2022-08-12 1157.650024 1186.000000 1134.000000 1143.550049 1143.550049 349096
2022-08-11 1169.900024 1169.900024 1145.300049 1156.400024 1156.400024 35965
2022-08-10 1140.000000 1173.699951 1119.250000 1161.400024 1161.400024 147605
2022-08-08 1135.650024 1185.000000 1125.099976 1150.550049 1150.550049 137005
2022-08-05 1146.050049 1148.000000 1125.000000 1131.050049 1131.050049 21972
2022-08-04 1129.000000 1155.000000 1113.099976 1141.949951 1141.949951 99562
2022-08-03 1140.000000 1140.000000 1120.199951 1127.400024 1127.400024 21817
2022-08-02 1137.000000 1140.250000 1122.349976 1132.000000 1132.000000 26384
2022-08-01 1145.199951 1160.000000 1133.900024 1139.650024 1139.650024 23234
2022-07-29 1160.000000 1160.000000 1126.050049 1133.150024 1133.150024 58370
2022-07-28 1130.000000 1149.400024 1105.650024 1113.800049 1113.800049 39529
2022-07-27 1133.349976 1144.650024 1126.199951 1129.349976 1126.349976 24933
2022-07-26 1181.000000 1183.849976 1125.800049 1133.349976 1130.339355 51901
2022-07-25 1165.000000 1179.000000 1155.349976 1169.000000 1165.894653 67424
2022-07-22 1145.000000 1156.599976 1125.199951 1153.599976 1150.535522 81910
Recommendation : BUY
Press 1:For Main Menu
Press 0:To Exit Program==36660==
==36660== HEAP SUMMARY:
==36660== in use at exit: 0 bytes in 0 blocks
==36660== total heap usage: 65 allocs, 65 frees, 52,808 bytes allocated
==36660==
==36660== All heap blocks were freed -- no leaks are possible
==36660==
==36660== For lists of detected and suppressed errors, rerun with: -s
==36660== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
97,1 Bot

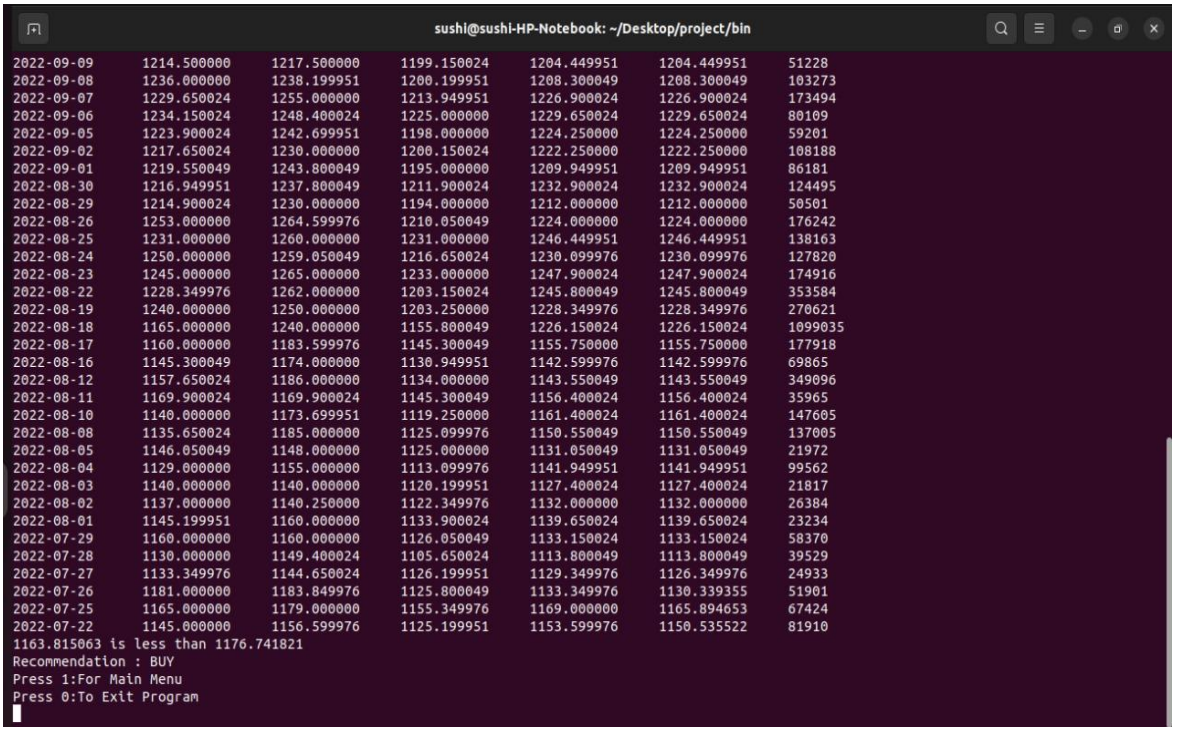
3.Splint：


sushi@sushi-HP-Notebook: ~/Desktop/project/reports
../src/loadStockData.c:7:19: Global head initialized to null value: head = NULL
A reference with no null annotation is assigned or initialized to NULL. Use
/*@null@*/ to declare the reference as a possibly null pointer. (Use
-nullassign to inhibit warning)
../src/loadStockData.c:7:19: Global head initialized to null value:
struct NODE * head = NULL = NULL
../src/loadStockData.c:8:19: Global node initialized to null value: node = NULL
../src/loadStockData.c:8:19: Global node initialized to null value:
struct NODE * node = NULL = NULL
../src/loadStockData.c: (in function loadStockData)
../src/loadStockData.c:22:20: Arrow access from possibly null pointer node:
node->date
A possibly null pointer is dereferenced. Value is either the result of a
function which may return null (in which case, code should check it is not
null), or a global, parameter or structure field declared with the null
qualifier. (Use -nullderef to inhibit warning)
../src/loadStockData.c:20:16: Storage node may become null
../src/loadStockData.c:24:28: Possibly null storage getOpen passed as non-null
param: strcpy (..., getOpen)
A possibly null pointer is passed as a parameter corresponding to a formal
parameter with no /*@null@*/ annotation. If NULL may be used for this
parameter, add a /*@null@*/ annotation to the function parameter declaration.
(Use -nullpass to inhibit warning)
../src/loadStockData.c:23:25: Storage getOpen may become null
../src/loadStockData.c:26:28: Possibly null storage getHigh passed as non-null
param: strcpy (..., getHigh)
../src/loadStockData.c:25:25: Storage getHigh may become null
../src/loadStockData.c:28:27: Possibly null storage getLow passed as non-null
param: strcpy (..., getLow)
../src/loadStockData.c:27:24: Storage getLow may become null
../src/loadStockData.c:30:29: Possibly null storage getClose passed as non-null
param: strcpy (..., getClose)
../src/loadStockData.c:29:26: Storage getClose may become null
../src/loadStockData.c:32:32: Possibly null storage getAdjClose passed as
non-null param: strcpy (..., getAdjClose)
../src/loadStockData.c:31:29: Storage getAdjClose may become null
../src/loadStockData.c:34:30: Possibly null storage getVolume passed as
1,1
Top

7.Testing:

7.1 Integration testing:

Test_case 1 (Recommendation Buy):



Test case-2(Recommendation Sell):