PROJECT REPORT

ON

MUTUAL FUNDS MANAGEMENT SYSYTEM USING C

By:

THIRUMALESH M R

PG DIMPLOMA IN EMBEDDED SYSTEM AND DEVELOPMENT

INDIAN INSTITUTE OF EMBEDDED SYSTEM

BANGALURU – 560095

TABLE OF CONTENT

INTRODUCTION

The C program presented here serves as a comprehensive system for mutual fund investment management, focusing on user-friendly interaction with mutual fund transactions. By integrating core functionalities such as user account management, mutual fund record keeping, and transaction processing, it offers an efficient solution to track and manage financial investments. With the capability to calculate return on investment (ROI) and maintain persistent data across sessions, the program mimics real-world financial operations. Users can securely store their data in CSV files, manage their portfolios, and perform financial operations like buying and selling mutual fund stocks.

Mutual funds are a popular investment avenue, where investors can invest in professionally managed portfolios of stocks, bonds, or other securities. The return on investment in mutual funds can be assessed over different time horizons, such as one-year, three-year, and five-year returns, making this program a useful tool for personal or educational use in financial analysis.

The program is structured into distinct functional modules, including user management, mutual fund transactions, portfolio management, and ROI calculations, providing users with an intuitive experience. Through the use of file handling techniques in C, user data and mutual fund details are securely stored and can be retrieved or updated at any time.

OBJECTIVE

The primary objective of this C program is to simulate a basic mutual fund investment system, offering several key functionalities**:**

1. **User Management:**
   - Users can be added, removed, and viewed in the system.
   - The program ensures each user has a unique username and ID, which are stored securely in a CSV file.
   - A CSV file is created for each user to record their mutual fund transactions, ensuring a personalized portfolio management system.
   - It supports up to 50 users, making it scalable for small investment groups or educational purposes

2. **Mutual Funds Management**
   - The program displays a list of predefined mutual funds, along with their tickers and return percentages for 1-year, 3-year, and 5-year periods.
   - Users can view this information to make informed decisions about their investments.
   - Mutual fund details are stored in a CSV file, making it easy to update or add more funds in the future.

3. **Return on Investment (ROI) Calculation:**
   - The program includes a feature to calculate the ROI based on the user's investment in different mutual funds.
   - Users can select a mutual fund and enter an investment amount, and the program calculates the returns based on 1-year, 3-year, and 5-year rates of return.

- This feature helps users assess the potential profitability of their investments, contributing to informed decision-making.

4. Data Persistence and File Management:

- All user and mutual fund data are stored in CSV files, which allows for easy data storage, retrieval, and updates.
- A main CSV file stores the list of all users, including their usernames and passwords.
- Separate CSV files are created for each user, where their investment details are recorded.
- This file-based system ensures the persistence of data between different sessions, allowing users to maintain their investment records over time.

5. Security and Uniqueness:

- The program ensures that each user has a unique username, preventing any duplication and ensuring data integrity.
- Usernames and passwords are stored securely, and each user has a unique file to manage their own transactions, reducing the chances of data mix-ups.
- The generation of a unique user ID for each user adds another layer of uniqueness and helps in managing user records efficiently.

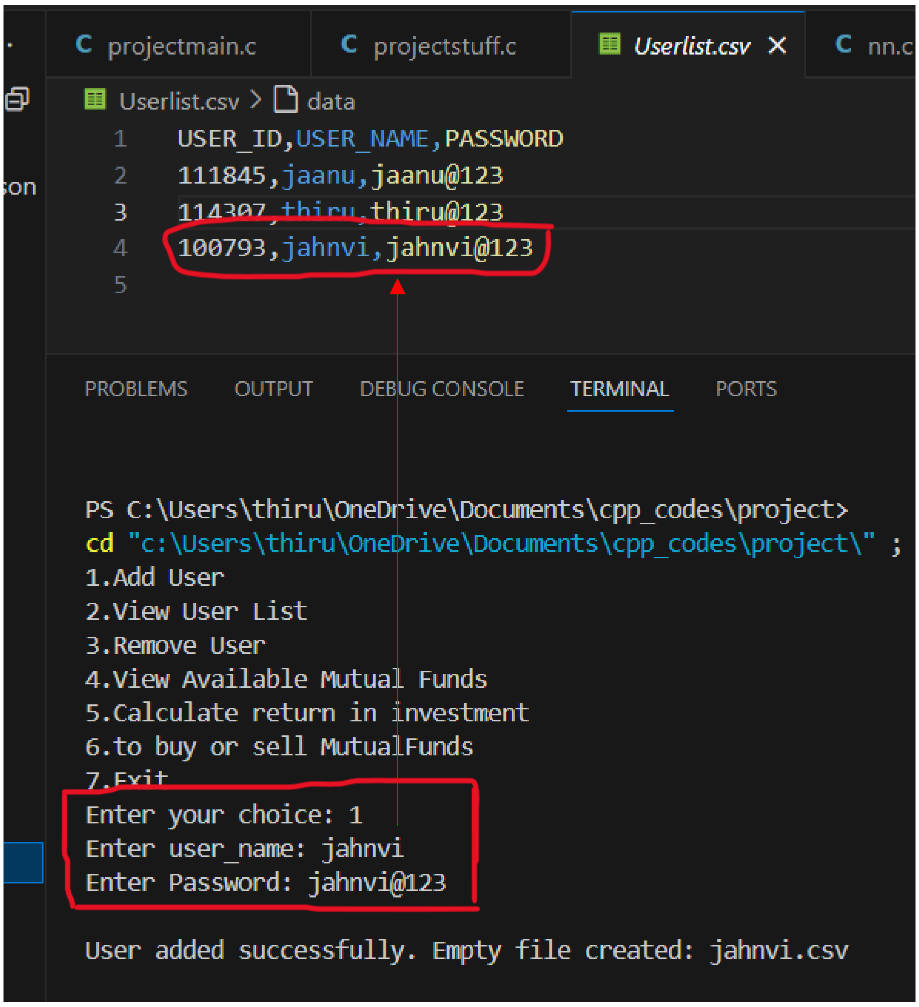
WORK FLOW

The mutual fund investment system is designed with a logical and easy-to-follow workflow that simulates the real-world process of mutual fund management. Below is a step-by-step description of the entire workflow:

1. User registration and Authentication

Manage user accounts by allowing new user registration and ensuring secure access to the system.

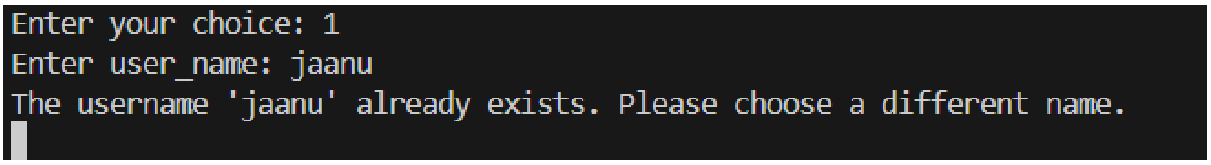
- To Add User
  Add user part generates the six-digit unique ID for each user and ask the user to enter their name (that is user name) and set password.



```
USER_ID,USER_NAME,PASSWORD
111845,jaanu,jaanu@123
114307,thiru,thiru@123
100793,jahnvi,jahnvi@123
```

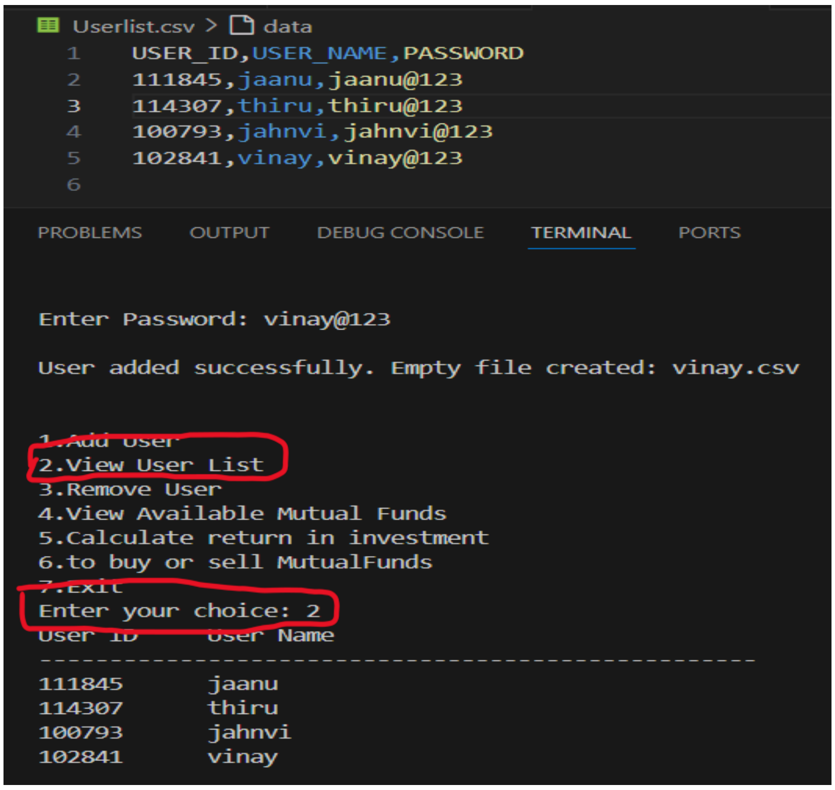
```
PS C:\Users\thiru\OneDrive\Documents\cpp_codes\project>
cd "c:\Users\thiru\OneDrive\Documents\cpp_codes\project\" ;
1.Add User
2.View User List
3.Remove User
4.View Available Mutual Funds
5.Calculate return in investment
6.to buy or sell MutualFunds
7.Exit
Enter your choice: 1
Enter user_name: jahnvi
Enter Password: jahnvi@123

User added successfully. Empty file created: jahnvi.csv
```

If the user name already exist it ask to enter the new user name as it is already exist.



```
Enter your choice: 1
Enter user_name: jaanu
The username 'jaanu' already exists. Please choose a different name.
```

- To View User List



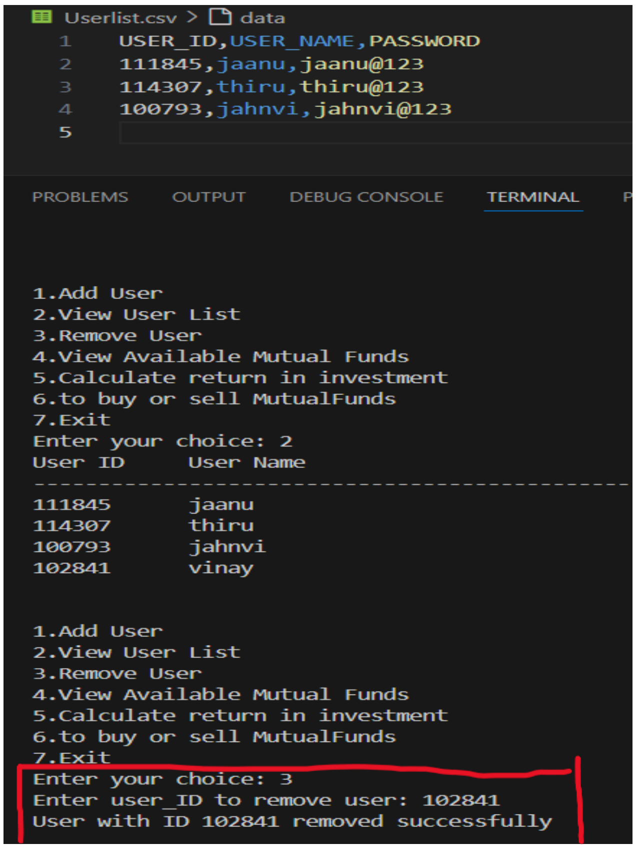
```
USER_ID,USER_NAME,PASSWORD
111845,jaanu,jaanu@123
114307,thiru,thiru@123
100793,jahnvi,jahnvi@123
102841,vinay,vinay@123

```

```
Enter Password: vinay@123

User added successfully. Empty file created: vinay.csv


1.Add User
2.View User List
3.Remove User
4.View Available Mutual Funds
5.Calculate return in investment
6.to buy or sell MutualFunds
7.Exit
Enter your choice: 2
User ID         User Name
---------------------------------------------------
111845          jaanu
114307          thiru
100793          jahnvi
102841          vinay
```

- To remove user from list

  With the user ID the user will be removed form the Userlist.csv file and the rest user will re-written to file Userlist.csv

  If the entered ID doesn't found in Userlist.csv file ,it prints no user found to terminal window.



```
USER_ID,USER_NAME,PASSWORD
111845,jaanu,jaanu@123
114307,thiru,thiru@123
100793,jahnvi,jahnvi@123
```

```
1.Add User
2.View User List
3.Remove User
4.View Available Mutual Funds
5.Calculate return in investment
6.to buy or sell MutualFunds
7.Exit
Enter your choice: 2
User ID      User Name
----------------------------------------------
111845       jaanu
114307       thiru
100793       jahnvi
102841       vinay

1.Add User
2.View User List
3.Remove User
4.View Available Mutual Funds
5.Calculate return in investment
6.to buy or sell MutualFunds
7.Exit
Enter your choice: 3
Enter user_ID to remove user: 102841
User with ID 102841 removed successfully
```

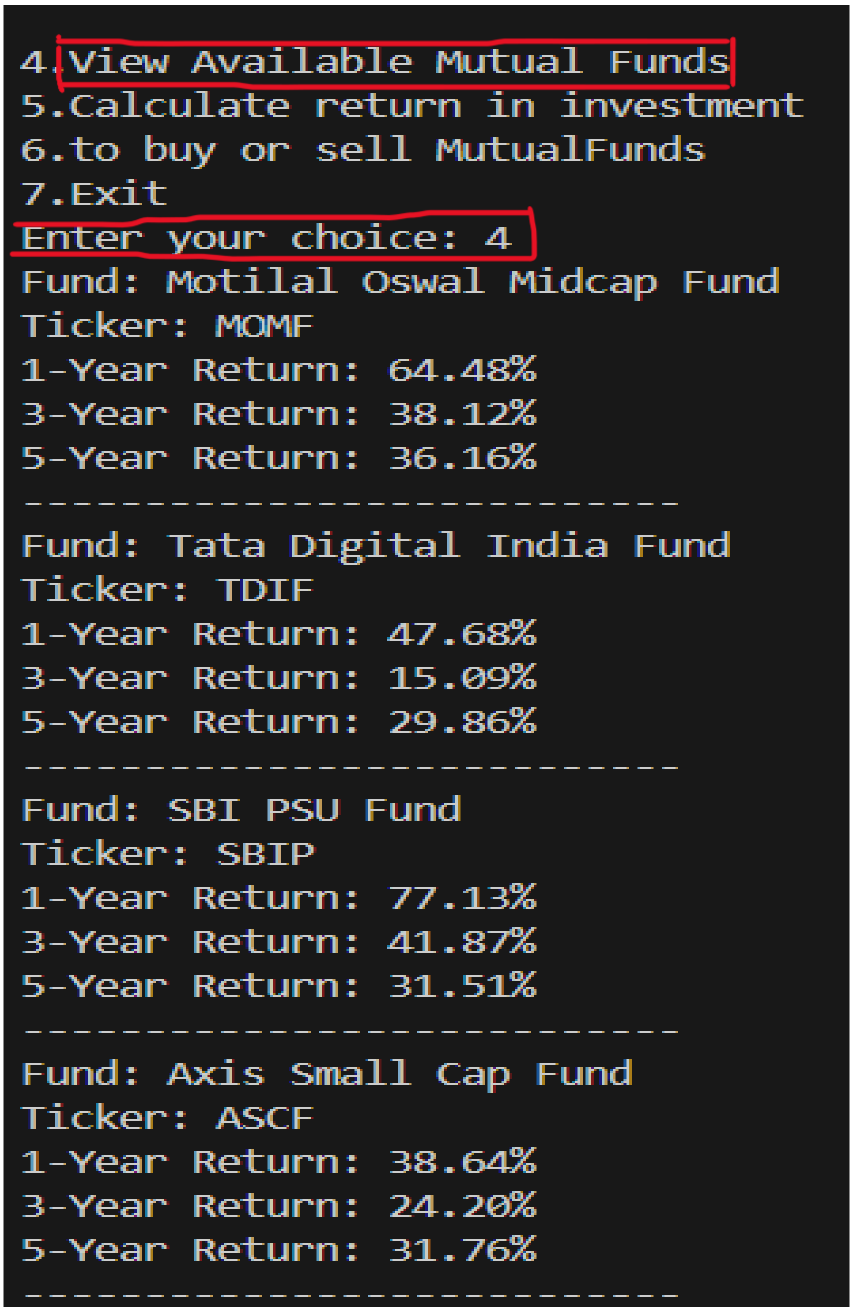
2. Displaying Mutual Funds Information

- To Explore Mutual Funds in List

  The mutual funds are loaded to file mutualfunds.csv file.

  Once the user choice the option to view or explore the mutual funds the funds present in that file will be displayed along with one, three and five years returns.

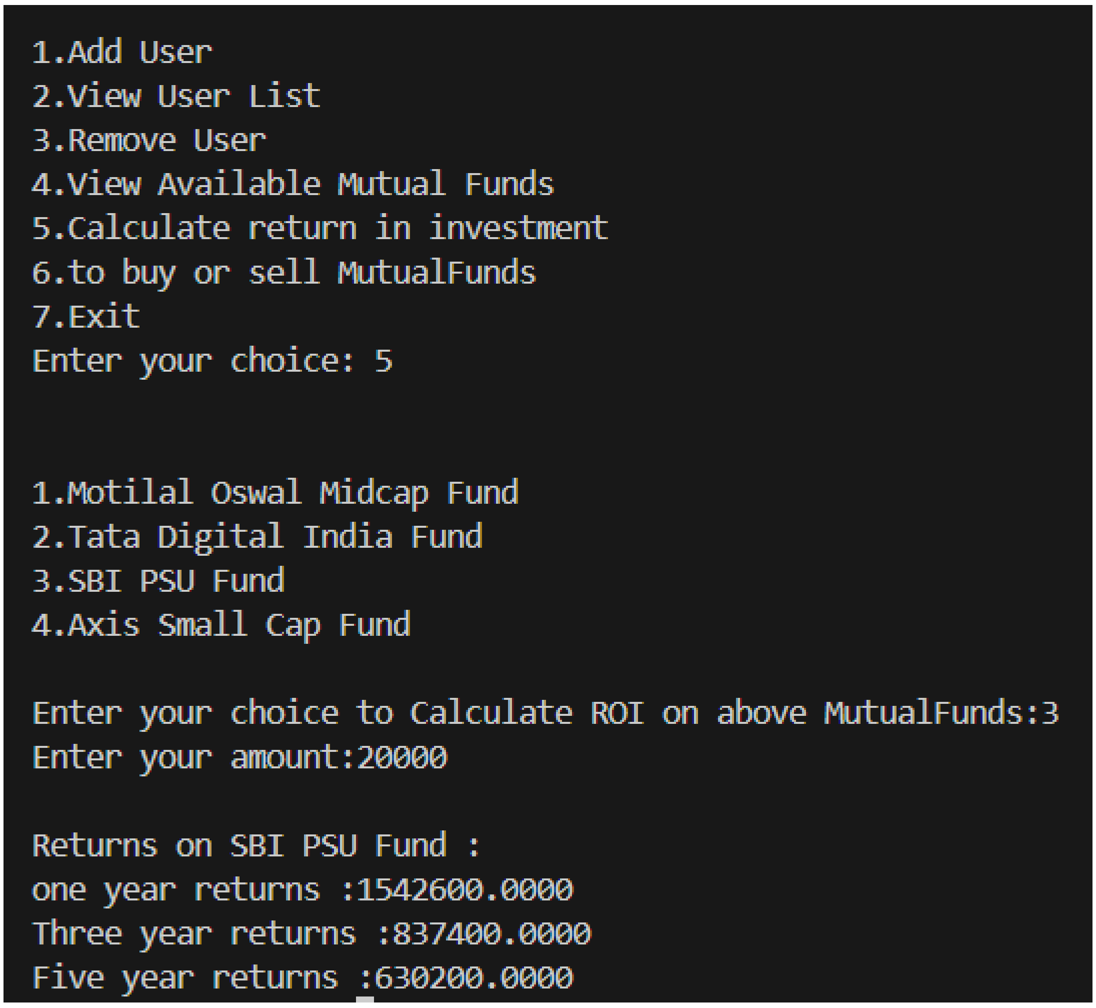
  For now there are few stocks present in that file, those a displayed in below picture.



```
4.View Available Mutual Funds
5.Calculate return in investment
6.to buy or sell MutualFunds
7.Exit
Enter your choice: 4
Fund: Motilal Oswal Midcap Fund
Ticker: MOMF
1-Year Return: 64.48%
3-Year Return: 38.12%
5-Year Return: 36.16%
------------------------------
Fund: Tata Digital India Fund
Ticker: TDIF
1-Year Return: 47.68%
3-Year Return: 15.09%
5-Year Return: 29.86%
------------------------------
Fund: SBI PSU Fund
Ticker: SBIP
1-Year Return: 77.13%
3-Year Return: 41.87%
5-Year Return: 31.51%
------------------------------
Fund: Axis Small Cap Fund
Ticker: ASCF
1-Year Return: 38.64%
3-Year Return: 24.20%
5-Year Return: 31.76%
------------------------------
```

- To calculate ROI (return on investment):
  Based on the number of years return percentage, the ROI (returns on investment) is calculated.

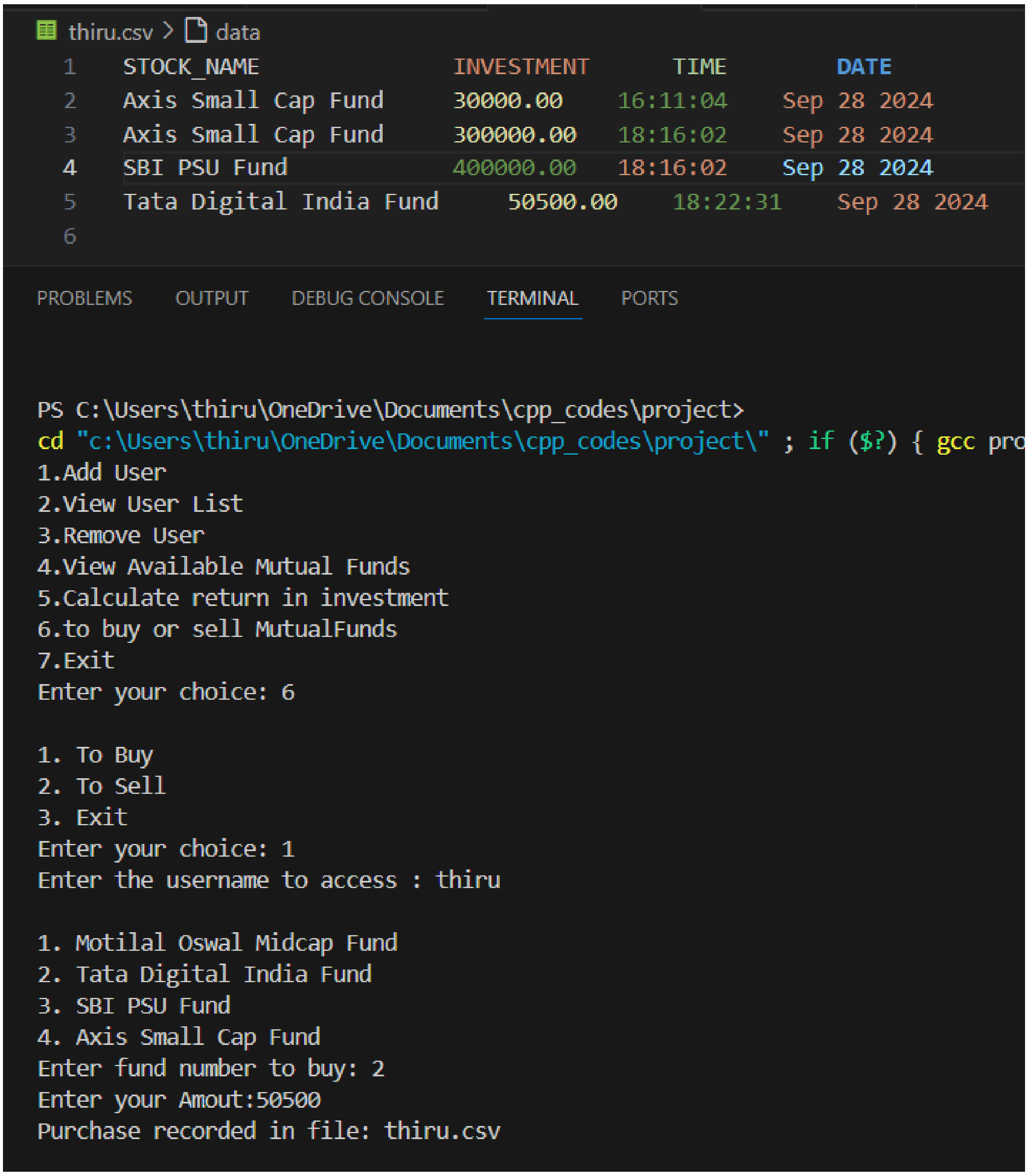
  Depends on the user choice which funds ROI need to seen that returns will be calculated and displayed on terminal window as displayed in below image.



```
1.Add User
2.View User List
3.Remove User
4.View Available Mutual Funds
5.Calculate return in investment
6.to buy or sell MutualFunds
7.Exit
Enter your choice: 5


1.Motilal Oswal Midcap Fund
2.Tata Digital India Fund
3.SBI PSU Fund
4.Axis Small Cap Fund

Enter your choice to Calculate ROI on above MutualFunds:3
Enter your amount:20000

Returns on SBI PSU Fund :
one year returns :1542600.0000
Three year returns :837400.0000
Five year returns :630200.0000
```

- To Buy Mutual Funds

  After the user calculate the ROI and if he/she is interested in buying and chooses the option "To Buy" and the says the amount of investment he/she is investing that mutual fund and load that fund name, his/her investment, time and date to his/her dash board i.e, to their .csv file as displayed in below image

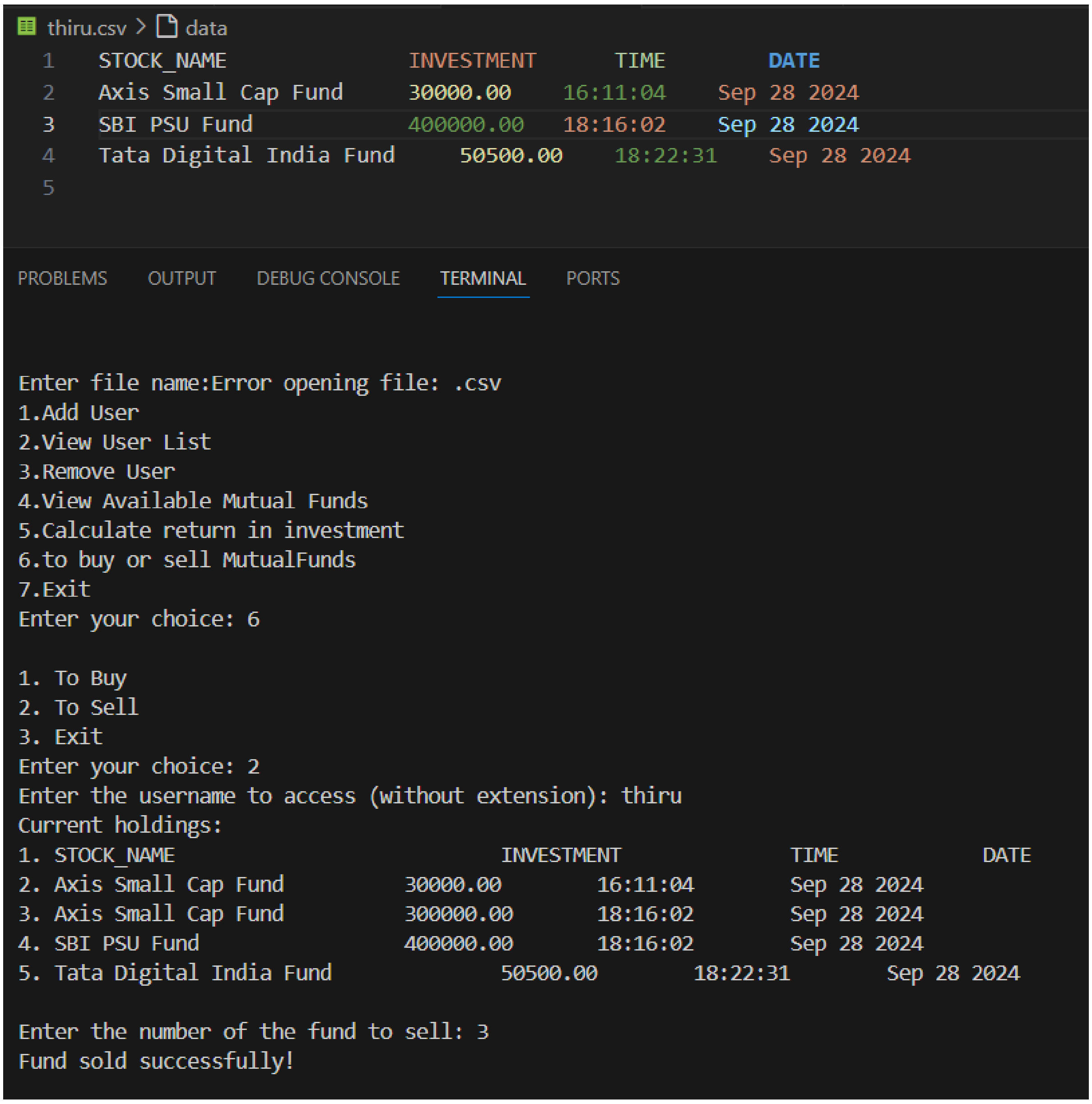


```
STOCK_NAME              INVESTMENT      TIME        DATE
Axis Small Cap Fund     30000.00     16:11:04     Sep 28 2024
Axis Small Cap Fund     300000.00    18:16:02     Sep 28 2024
SBI PSU Fund            400000.00    18:16:02     Sep 28 2024
Tata Digital India Fund     50500.00     18:22:31     Sep 28 2024
```

```
PS C:\Users\thiru\OneDrive\Documents\cpp_codes\project>
cd "c:\Users\thiru\OneDrive\Documents\cpp_codes\project\" ; if ($?) { gcc pro
1.Add User
2.View User List
3.Remove User
4.View Available Mutual Funds
5.Calculate return in investment
6.to buy or sell MutualFunds
7.Exit
Enter your choice: 6

1. To Buy
2. To Sell
3. Exit
Enter your choice: 1
Enter the username to access : thiru

1. Motilal Oswal Midcap Fund
2. Tata Digital India Fund
3. SBI PSU Fund
4. Axis Small Cap Fund
Enter fund number to buy: 2
Enter your Amout:50500
Purchase recorded in file: thiru.csv
```

○ To sell Mutual funds

Once user select the option “To sell”, the user dashboard will be displayed on the on the terminal window. If user select which stock to sell that will be remover form the user dashboard.



thiru.csv > data

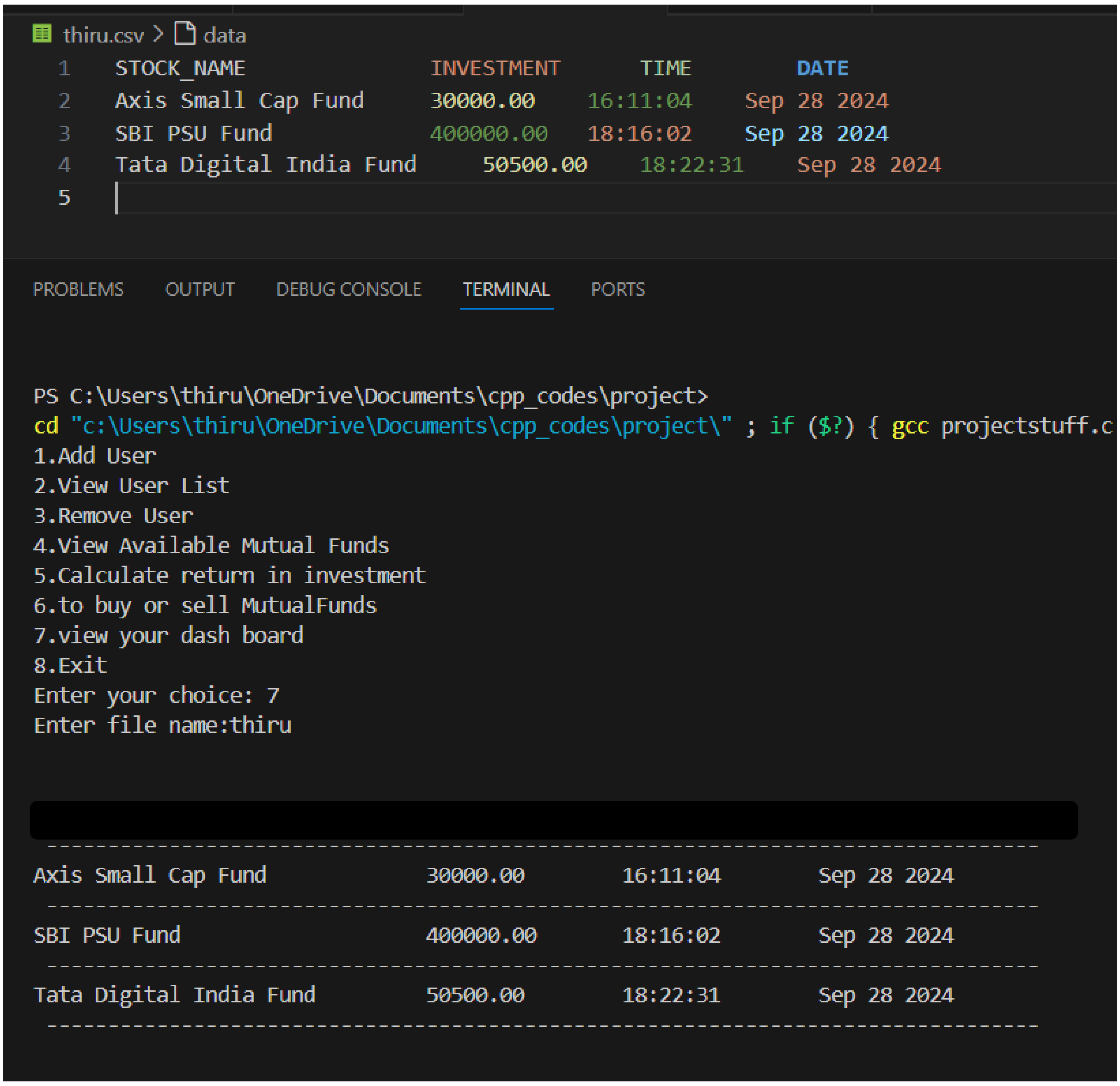
```
STOCK_NAME               INVESTMENT       TIME         DATE
Axis Small Cap Fund      30000.00     16:11:04     Sep 28 2024
SBI PSU Fund             400000.00    18:16:02     Sep 28 2024
Tata Digital India Fund      50500.00     18:22:31     Sep 28 2024
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Enter file name:Error opening file: .csv
1.Add User
2.View User List
3.Remove User
4.View Available Mutual Funds
5.Calculate return in investment
6.to buy or sell MutualFunds
7.Exit
Enter your choice: 6

1. To Buy
2. To Sell
3. Exit
Enter your choice: 2
Enter the username to access (without extension): thiru
Current holdings:
1. STOCK_NAME                       INVESTMENT           TIME           DATE
2. Axis Small Cap Fund         30000.00        16:11:04        Sep 28 2024
3. Axis Small Cap Fund         300000.00       18:16:02        Sep 28 2024
4. SBI PSU Fund                400000.00       18:16:02        Sep 28 2024
5. Tata Digital India Fund             50500.00        18:22:31        Sep 28 2024

Enter the number of the fund to sell: 3
Fund sold successfully!
```

- To View Dashboard

  Enter the option of view your dash board i.e, option 7, all your mutual funds will be displayed on terminal window, those will be the mutual funds which all you bought.



```
STOCK_NAME              INVESTMENT       TIME         DATE
Axis Small Cap Fund     30000.00     16:11:04     Sep 28 2024
SBI PSU Fund            400000.00    18:16:02     Sep 28 2024
Tata Digital India Fund      50500.00     18:22:31     Sep 28 2024
```

```
PS C:\Users\thiru\OneDrive\Documents\cpp_codes\project>
cd "c:\Users\thiru\OneDrive\Documents\cpp_codes\project\" ; if ($?) { gcc projectstuff.c
1.Add User
2.View User List
3.Remove User
4.View Available Mutual Funds
5.Calculate return in investment
6.to buy or sell MutualFunds
7.view your dash board
8.Exit
Enter your choice: 7
Enter file name:thiru


[redacted]
 --------------------------------------------------------------------------------
Axis Small Cap Fund             30000.00        16:11:04        Sep 28 2024
 --------------------------------------------------------------------------------
SBI PSU Fund                    400000.00       18:16:02        Sep 28 2024
 --------------------------------------------------------------------------------
Tata Digital India Fund         50500.00        18:22:31        Sep 28 2024
 --------------------------------------------------------------------------------
```

Conclusion:

In conclusion, the Mutual Fund Investment System program provides a comprehensive and user-friendly simulation of the mutual fund investment process. By integrating features such as user registration, secure login, the ability to view, buy, and sell mutual funds, and calculating potential returns, the system offers a realistic environment for users to manage their investments. Data persistence through CSV files ensures that user information and transaction history are securely stored and accessible across sessions. This program not only helps users familiarize themselves with the workings of mutual funds but also enhances their understanding of long-term investment strategies and the potential returns they can yield. The system serves as a powerful tool for learning about mutual fund investments, making informed decisions, and tracking investment performance over time.

Appendix

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <time.h>

#define max_users 50
#define name_len 50
#define password_len 50
#define NUM_FUNDS 4
#define MAX_LINE_LENGTH 256

typedef struct
{
    int user_id;
    char name[name_len];
    char password[password_len];
} user;

typedef struct
{
    char name[50];
    char ticker[5];
    float oneYearReturn;
    float threeYearReturn;
    float fiveYearReturn;
} MutualFund;

// Add user

void add_user(user *s, int *num_user)
{
    if (*num_user >= max_users)
    {
        printf("User limit reached.\n");
        return;
    }

    // Load the existing users from file to check for duplicate names
    FILE *file = fopen("Userlist.csv", "r");
    if (file == NULL)
    {
        printf("Error opening user list file.\n");
        return;
    }

    char line[MAX_LINE_LENGTH];
    fgets(line, sizeof(line), file); // Skip header line

    // Temporary array to hold existing user names
    char existingNames[max_users][name_len];
    int existingUserCount = 0;

    // Read the existing users from the file
    while (fgets(line, sizeof(line), file))
    {
        sscanf(line, "%*d,%49[^,],%*s",
existingNames[existingUserCount]);
        existingUserCount++;
    }
    fclose(file);

    int uniqueName = 0;
    while (!uniqueName)
    {
        uniqueName = 1;  // Assume the name is unique unless found
otherwise

        getchar(); // Consume any stray newline character
        printf("Enter user_name: ");
        fgets(s[*num_user].name, sizeof(s[*num_user].name), stdin);
        s[*num_user].name[strcspn(s[*num_user].name, "\n")] = 0; //
Remove newline

        // Check if the entered name already exists
        for (int i = 0; i < existingUserCount; i++)
        {
            if (strcmp(s[*num_user].name, existingNames[i]) == 0)
            {
                printf("The username '%s' already exists. Please choose a
different name.\n", s[*num_user].name);
                uniqueName = 0;
                break;
            }
        }
    }

    // Get user password
    printf("Enter Password: ");
    fgets(s[*num_user].password, sizeof(s[*num_user].password),
stdin);
    s[*num_user].password[strcspn(s[*num_user].password, "\n")] = 0;
// Remove newline

    // Generate a unique user ID
    int num;
    srand(time(0));
    num = rand() % 90000 + 100000;
    s[*num_user].user_id = num;

    (*num_user)++;

    // Write to the main user list file
    FILE *ptr = fopen("Userlist.csv", "w");
    if (ptr == NULL)
    {
        printf("Error opening file");
        return;
    }
    fprintf(ptr, "USER_ID,USER_NAME,PASSWORD\n"); // Write
header
    for (int i = 0; i < *num_user; i++)
    {
        fprintf(ptr, "%d,%s,%s\n", s[i].user_id, s[i].name, s[i].password);
    }
    fclose(ptr);

    // Create a separate empty CSV file for the new user
    char filename[100];
    snprintf(filename, sizeof(filename), "%s.csv", s[*num_user -
1].name);
```

```
    // Replace any invalid characters in filename with underscores
    for (int i = 0; filename[i] != '\0'; i++)
    {
        if (filename[i] == ' ' || filename[i] == '/' || filename[i] == '\\')
        {
            filename[i] = '_';
        }
    }

    FILE *userFile = fopen(filename, "w"); // Create an empty file
    if (userFile == NULL)
    {
        printf("Error creating user file: %s\n", filename);
        return;
    }
    fprintf(userFile,
"STOCK_NAME\t\t\t\tINVESTMENT\t\tTIME\t\tDATE\n");

    fclose(userFile); // Close the file immediately to keep it empty

    printf("\nUser added successfully. Empty file created: %s\n",
filename);
    printf("\n\n");
}
// View user list

void user_list()
{
    FILE *ptr = fopen("Userlist.csv", "r");
    if (ptr == NULL)
    {
        perror("Error opening file");
        return;
    }

    char line[MAX_LINE_LENGTH];

    // Read and skip the header line
    if (fgets(line, sizeof(line), ptr) == NULL)
    {
        perror("Error reading header");
        fclose(ptr);
        return;
    }

    // Print header for user list
    printf("User ID     User Name         \n");
    printf("--------------------------------------------------\n");

    // Read each line and print user details
    while (fgets(line, sizeof(line), ptr))
    {
        int user_id;
        char name[50];
        char password[50];

        if (sscanf(line, "%d,%49[^,],%49[^\n]", &user_id, name,
password) == 3)
        {
            printf("%-10d  %-25s\n", user_id, name);
        }
        else
        {
            fprintf(stderr, "Error parsing line: %s", line);
        }
    }

    fclose(ptr);
    printf("\n\n");
}

// Remove user

int removeUser(user *s, int *num_user, int user_id)
{
    int i, found = 0;

    // Find the user with the given ID
    for (i = 0; i < *num_user; i++)
    {
        if (s[i].user_id == user_id)
        {
            found = 1;
            break;
        }
    }

    if (!found)
    {
        printf("User with ID %d not found\n", user_id);
        return -1;
    }

    // Shift elements to remove the user
    for (int j = i; j < *num_user - 1; j++)
    {
        s[j] = s[j + 1];
    }
    (*num_user)--;

    // Rewrite the file after removing the user
    FILE *ptr = fopen("Userlist.csv", "w"); // Use "w" to overwrite the
file
    if (ptr == NULL)
    {
        perror("Error opening file for writing");
        return -1;
    }

    fprintf(ptr, "USER_ID,USER_NAME,PASSWORD\n"); // Write
header once
    for (i = 0; i < *num_user; i++)
    {
        fprintf(ptr, "%d,%s,%s\n", s[i].user_id, s[i].name, s[i].password);
    }

    fclose(ptr);

    printf("User with ID %d removed successfully\n", user_id);

    printf("\n\n");
}

// View Available Mutual Funds

int ToviewMF(MutualFund *funds)
{
    FILE *ptr = fopen("mutualfunds.csv", "r");
    if (ptr == NULL)
    {
        perror("Error opening file");
        return 1;
    }

    char line[MAX_LINE_LENGTH];

    // Skip the header line
    if (fgets(line, sizeof(line), ptr) == NULL)
    {
        perror("Error reading header");
        fclose(ptr);
        return 1;
    }

    // Read each line of the CSV file
    for (int i = 0; i < NUM_FUNDS; i++)
    {
        if (fgets(line, sizeof(line), ptr) == NULL)
        {
            perror("Error reading data line");
            fclose(ptr);
            return 1;
        }
        // Parse the line into the MutualFund structure
        if (sscanf(line, "%49[^,],%4[^,],%f,%f,%f",
                   funds[i].name,
                   funds[i].ticker,
                   &funds[i].oneYearReturn,
                   &funds[i].threeYearReturn,
                   &funds[i].fiveYearReturn) != 5)
        {
            fprintf(stderr, "Error parsing line: %s", line);
            fclose(ptr);
            return 1;
        }
    }

    fclose(ptr);

    // Print the data on console window
    for (int i = 0; i < NUM_FUNDS; i++)
    {
        printf("Fund: %s\n", funds[i].name);
        printf("Ticker: %s\n", funds[i].ticker);
        printf("1-Year Return: %.2f%%\n", funds[i].oneYearReturn);
        printf("3-Year Return: %.2f%%\n", funds[i].threeYearReturn);
        printf("5-Year Return: %.2f%%\n", funds[i].fiveYearReturn);
        printf("--------------------------\n");
    }

    printf("\n\n");
}

// To calculate ROI

void calclulate_ROI()
{
    printf("\n\n1.Motilal Oswal Midcap Fund\n2.Tata Digital India 
Fund\n3.SBI PSU Fund\n4.Axis Small Cap Fund\n");
    printf("\nEnter your choice to Calculate ROI on above 
MutualFunds:");
    int choice;
    scanf("%d", &choice);
    while (1)
    {
        float amount;
        printf("Enter your amount:");
        scanf("%f", &amount);
        switch (choice)
        {
        case 1:
            printf("\nReturns on Motilal Oswal Midcap Fund :\n");
            printf("one year returns :%.4f\n", (amount * 64.48));
            printf("Three year returns :%.4f\n", (amount * 38.12));
            printf("Five year returns :%.4f\n", (amount * 36.16));
            break;
        case 2:
            printf("\nReturns on Tata Digital India Fund :\n");
            printf("one year returns :%.4f\n", (amount * 47.68));
            printf("Three year returns :%.4f\n", (amount * 15.09));
            printf("Five year returns :%.4f\n", (amount * 29.86));
            break;
        case 3:
            printf("\nReturns on SBI PSU Fund :\n");
            printf("one year returns :%.4f\n", (amount * 77.13));
            printf("Three year returns :%.4f\n", (amount * 41.87));
            printf("Five year returns :%.4f\n", (amount * 31.51));
            break;
        case 4:
            printf("\nReturns on Axis Small Cap Fund :\n");
            printf("one year returns :%.4f\n", (amount * 38.64));
            printf("Three year returns :%.4f\n", (amount * 24.20));
            printf("Five year returns :%.4f\n", (amount * 31.76));
            break;
        }
    }
    printf("\n\n");
}
```

```
//buy socks

void buy(user *s, int num_user)
{
    char filename[100];

    // Ask the user to enter an existing filename
    printf("Enter the username to access : ");
    fgets(filename, sizeof(filename), stdin);
    filename[strcspn(filename, "\n")] = 0; // Remove newline character
from the filename input

    // Add ".csv" extension to the filename
    snprintf(filename, sizeof(filename), "%s.csv", filename);

    // Try to open the existing file
    FILE *userFile = fopen(filename, "a");
    if (userFile == NULL)
    {
        printf("Error opening file: %s\n", filename);
        return;
    }

    // Prompt for mutual fund options
    printf("\n1. Motilal Oswal Midcap Fund\n2. Tata Digital India
Fund\n3. SBI PSU Fund\n4. Axis Small Cap Fund\n");
    printf("Enter fund number to buy: ");
    int num;
    scanf("%d", &num);
    float amount;
    printf("Enter your Amout:");
    scanf("%f",&amount);
    switch (num)
    {
        case 1:
            fprintf(userFile, "Motilal Oswal
Midcapfund\t\t%.2f\t%s\t%s\n", amount, __TIME__, __DATE__);
            break;
        case 2:
            fprintf(userFile, "Tata Digital India Fund\t\t%.2f\t%s\t%s\n",
amount,__TIME__, __DATE__);
            break;
        case 3:
            fprintf(userFile, "SBI PSU Fund\t\t\t%.2f\t%s\t%s\n",
amount,__TIME__, __DATE__);
            break;
        case 4:
            fprintf(userFile, "Axis Small Cap Fund\t\t%.2f\t%s\t%s\n",
amount, __TIME__, __DATE__);
            break;
        default:
            printf("Invalid choice!\n");
            fclose(userFile);
    }

    printf("Purchase recorded in file: %s\n", filename);
    fclose(userFile);
    printf("\n\n");
}

//sell stock and display current holdings

void sell(user *s, int num_user)
{
    char filename[100];
    // Ask the user to enter an existing filename
    printf("Enter the username to access (without extension): ");
    fgets(filename, sizeof(filename), stdin);
    filename[strcspn(filename, "\n")] = 0; // Remove newline character
from the filename input

    // Add ".csv" extension to the filename
    snprintf(filename, sizeof(filename), "%s.csv", filename);

    // Open the file for reading to check current holdings
    FILE *userFile = fopen(filename, "r");
    if (userFile == NULL)
    {
        printf("Error opening file: %s\n", filename);
        return;
    }

    // Temporary buffer to hold lines, and array for the user's current
holdings
    char lines[MAX_LINE_LENGTH][MAX_LINE_LENGTH];
    int lineCount = 0;

    // Read the current holdings from the file and store them in the
array
    while (fgets(lines[lineCount], sizeof(lines[lineCount]), userFile) !=
NULL)
    {
        lineCount++;
    }
    fclose(userFile);

    // Display the current holdings to the user
    printf("Current holdings:\n");
    for (int i = 0; i < lineCount; i++)
    {
        printf("%d. %s", i + 1, lines[i]);  // Display the list of holdings
    }

    // Ask the user which stock to sell
    int sellChoice;
    printf("\nEnter the number of the fund to sell: ");
    scanf("%d", &sellChoice);
    getchar();  // To clear newline character from input buffer

    // Validate the choice
    if (sellChoice < 1 || sellChoice > lineCount)
    {
        printf("Invalid choice!\n");
        return;
    }

    // Remove the selected stock from the holdings
    for (int i = sellChoice - 1; i < lineCount - 1; i++)
    {
        strcpy(lines[i], lines[i + 1]);
    }
    lineCount--;  // Decrease the line count as one item has been
removed

    // Open the file for writing (overwrite)
    userFile = fopen(filename, "w");
    if (userFile == NULL)
    {
        printf("Error opening file: %s\n", filename);
        return;
    }

    // Write the updated holdings back to the file
    for (int i = 0; i < lineCount; i++)
    {
        fprintf(userFile, "%s", lines[i]);  // Write each line back to the
file
    }

    fclose(userFile);  // Close the file after updating
    printf("Fund sold successfully!\n");

    printf("\n\n");
}

//to buy or sell stocks

void buysell(user *s, int num_user)
{
    int bs;
    printf("\n1. To Buy\n2. To Sell\n3. Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &bs);
    getchar();

    switch (bs)
    {
        case 1:
            buy(s, num_user);
            break;
        case 2:
            sell(s, num_user);
            break;
        case 3:
            printf("Exiting...\n");
            return;
        default:
            printf("Invalid choice! Try again.\n");
            break;
    }

    printf("\n\n");
}

//To view your mutual funds

void viewyourMF()
{
    char filename[20];
    char line[256];
    getchar();
    printf("Enter file name:");
    fgets(filename,sizeof(filename),stdin);
    filename[strcspn(filename,"\n")]=0;

    snprintf(filename,sizeof(filename),"%s.csv",filename);
     FILE *userFile = fopen(filename, "r");
    if (userFile == NULL)
    {
        printf("Error opening file: %s\n", filename);
        return;
    }
    printf("\n\n");
    while(fgets(line,sizeof(line),userFile))
    {
        printf("%s ",line);
        printf("-----------------------------------------------------------------
------------\n");

    }

    printf("\n\n");
}

int main()
{
    int num_user = 0;
    user s[max_users];
    MutualFund funds[NUM_FUNDS];
    int ch, n;

    // Load existing users from file
    FILE *file = fopen("Userlist.csv", "r");
    if (file != NULL)
    {
        char line[MAX_LINE_LENGTH];
        fgets(line, sizeof(line), file); // Skip header
        while (fgets(line, sizeof(line), file))
        {
            sscanf(line, "%d,%49[^,],%49[^\n]", &s[num_user].user_id,
s[num_user].name, s[num_user].password);
            num_user++;
        }
        fclose(file);
    }

    while (1)
    {
        printf("1.Add User\n2.View User List\n3.Remove User\n4.View Available Mutual Funds\n5.Calculate return in investment\n");
        printf("6.to buy or sell MutualFunds\n7.view your dash board\n8.Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &ch);
        switch (ch)
        {
        case 1:
            add_user(s, &num_user);
            break;
        case 2:
            user_list();
            break;
        case 3:
            printf("Enter user_ID to remove user: ");
            scanf("%d", &n);
            removeUser(s, &num_user, n);
            break;
        case 4:
            ToviewMF(funds);
            break;
        case 5:
            calclulate_ROI();
            break;
        case 6:
            buysell(s, num_user);
        case 7:
            viewyourMF();
            break;
        case 8:
            return 0;
        }}}
```

```
//code to load mutual funds

#include <stdio.h>
#include <string.h>
#define NUM_FUNDS 10

// structure for mutual fund data
typedef struct {
    char name[50];
    char ticker[10];
    float oneYearReturn;
    float threeYearReturn;
    float fiveYearReturn;
} MutualFund;

int main() {
    // Create an array of MutualFund structures
    MutualFund funds[NUM_FUNDS] = {
        {"Motilal Oswal Midcap Fund", "MOMF", 64.48, 38.12, 36.16},
        {"Tata Digital India Fund", "TDIF", 47.68, 15.09, 29.86},
        {"SBI PSU Fund", "SBIP", 77.13, 41.87, 31.51},
        {"Axis Small Cap Fund", "ASCF", 38.64, 24.20, 31.76},
    };

    // Open a CSV file for writing
    FILE *file = fopen("mutualfunds.csv", "w+");
    if (file == NULL) {
        perror("Failed to open file");
        return 1;
    }

    // Write the header line
    fprintf(file, "Name,Ticker,1-Year Return,3-Year Return,5-Year
Return\n");

    // Write data to the CSV file
    for (int i = 0; i < NUM_FUNDS; i++) {
        fprintf(file, "%s,%s,%.2f,%.2f,%.2f\n",
                funds[i].name,
                funds[i].ticker,
                funds[i].oneYearReturn,
                funds[i].threeYearReturn,
                funds[i].fiveYearReturn);
    }

    // Close the file
    fclose(file);
    printf("%d",NUM_FUNDS);
    return 0;
}
```

```
//code to create a file to store user list
#include<stdio.h>
int main()
{
    char line[256];
    FILE *ptr=fopen("Userlist.csv","w");
    if(ptr==NULL)
    {
        printf("error");
    }
    fprintf(ptr,"USER_ID\t\tUSER_NAME\t\tPASSWORD\n");
    fclose(ptr);
}
```