Final Report
(PROJECT FARMHOUSE)

Course Code: CS110 **Course Title:** Computer Programming

Semester: B. Tech 2nd Sem **Section:** S1

Academic Year: 2019-20 **Course Instructor:** B. R. Chandavarkar

Team Members:

1. Dhruv Banerjee, 191CH013, 9428418165, dbanerjee.191ch013@nitk.edu.in

2. Pranshu Shukla, 191ME260, 7385925943, pranshushukla.191me260@nitk.edu.in

1 Abstract

or a farmer getting crops harvested after many months long process is just half of the task, getting crops sold to market-place can be tiring, hard and extremely disadvantageous to a farmer if he is not completely acquainted with the process. This is where Farm-House steps in; it is a simple all-in-one application for farmers, retailers and corporations alike that enables an easy flow of information and communication between the users. Farm-House allows retailers to raise quotes on purchase of crops from farmers which not only makes it easier for farmers to contact their buyers directly but also allow other customers to see their competitions. This process removes the need of a middleman for the purchasing process which tend to take their own proportions of money from farmers. All a farmer has to do is choose retailers from a list of retailers who is offering the best quote for the crop and contact him directly instead of scrambling in a market searching and asking around for the best buyer. Lastly, it also allows retailers to see which crops are being more produced, which competitor is getting better customers and how the market is changing. With just a click away, all these features can really come to aid for anyone associated with the agriculture business. But mainly, from this project, we hope to create an easier world for farmers who are the driving force of our nations growth.

2 Introduction

armhouse is an user friendly application that allows easy communication between farmers and crop buyers during the purchasing phase. It allows buyers or cooperatives to raise quotes on certain crops they wish to buy and these are made visible to farmers making finding the right buyer a tireless process. The purpose of the project is to ease the purchasing and selling process of farmers and buyers alike for it is the most critical stage of their entire harvesting process as it determines the profits and losses that are thereby generated.

It is completely a C-programming based application that is mainly dependent on File Handling of various files and structures. The application utilizes 4 data (.dat) for storing which stores information regarding user details, details of farmers and buyers and finally, the quotes that are issued. The project allows new users to register as 2 type of users: Farmer or buyer. Upon the selection, each user is transferred to their corresponding portal where they can access a variety of facilities and options. Some of the key features of our application include: Login and Sign-Up facility, buyer search using User-ID, best search based on quote and location for farmers, adding and removing quotes for buyers along with visibility change feature and all quote issued history.

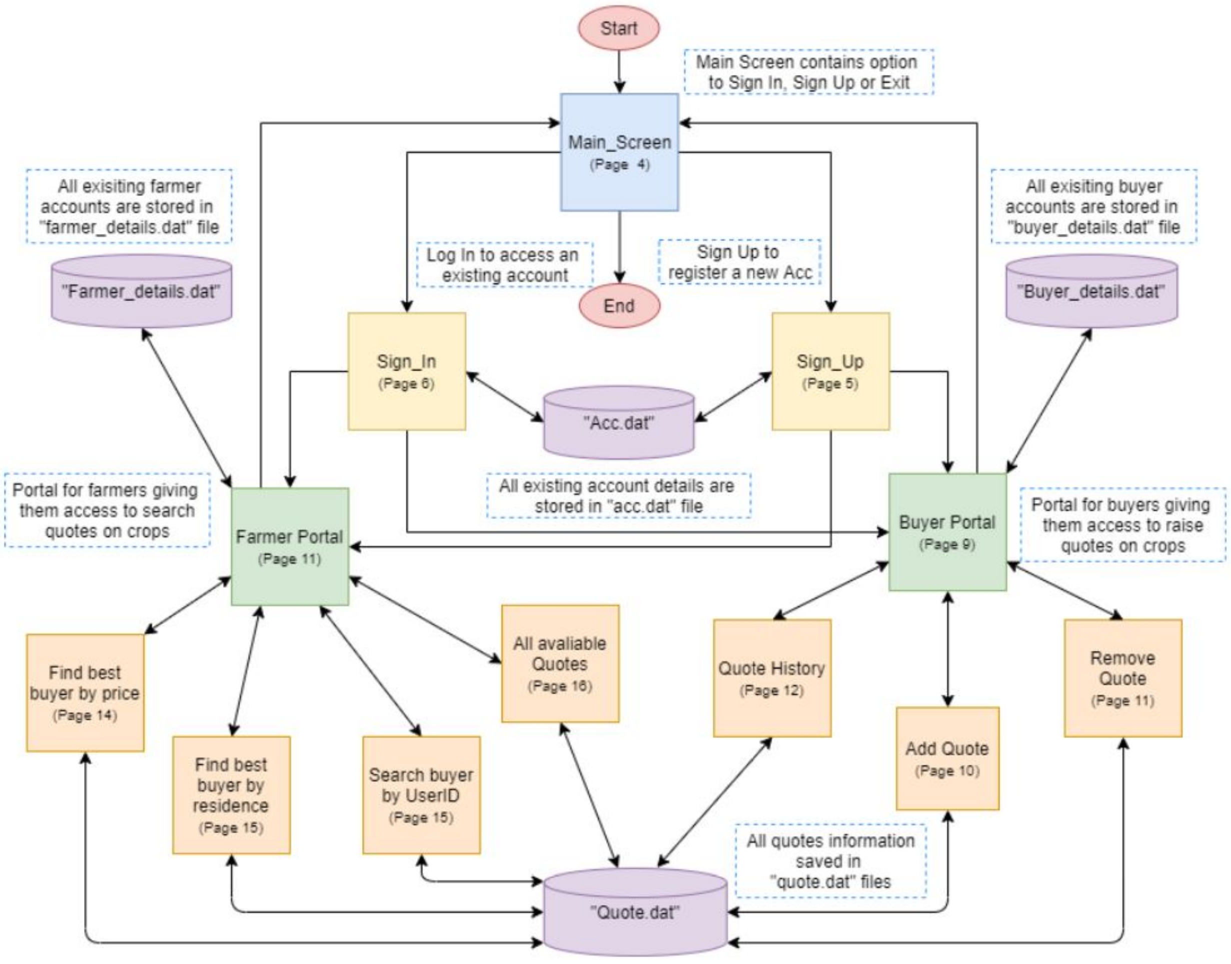
To prevent confusion and to increase appeal, a creative user interface was used on the application. Furthermore, to keep privacy and information safe of each user, a user ID-password system was implemented. Each user has the freedom to choose his or her own User ID with a condition that it is not already in use with another account, on which case the application will notify the same. The users can also Sign-out of their accounts and allow other users to Log-In without the need to close the application.

3 Flowchart or Algorithm

3.1 Program Structure

he program uses 4 different structures for storing information. These act as identification and authenticating points as well when creating or logging into an existing user. They are as follows:

1. "basic_details" - for storing UserID and Password of an user account along with other essential details like phone number, address etc
2. "farmer_details" - for storing farmer related information of the account
3. "buyer_details" - for storing buyer related information of the account
4. "buyer_quotes" - for storing information on each quote issued by a buyer on the application



Start
Main Screen contains option to Sign In, Sign Up or Exit
Main_Screen (Page 4)
All exisiting farmer accounts are stored in "farmer_details.dat" file
"Farmer_details.dat"
Log In to access an existing account
End
Sign Up to register a new Acc
All exisiting buyer accounts are stored in "buyer_details.dat" file
"Buyer_details.dat"
Sign_In (Page 6)
"Acc.dat"
Sign_Up (Page 5)
Portal for farmers giving them access to search quotes on crops
Farmer Portal (Page 11)
All existing account details are stored in "acc.dat" file
Buyer Portal (Page 9)
Portal for buyers giving them access to raise quotes on crops
Find best buyer by price (Page 14)
All avaliable Quotes (Page 16)
Quote History (Page 12)
Remove Quote (Page 11)
Find best buyer by residence (Page 15)
Search buyer by UserID (Page 15)
Add Quote (Page 10)
All quotes information saved in "quote.dat" files
"Quote.dat"

Figure 1: Flow Overview and Index

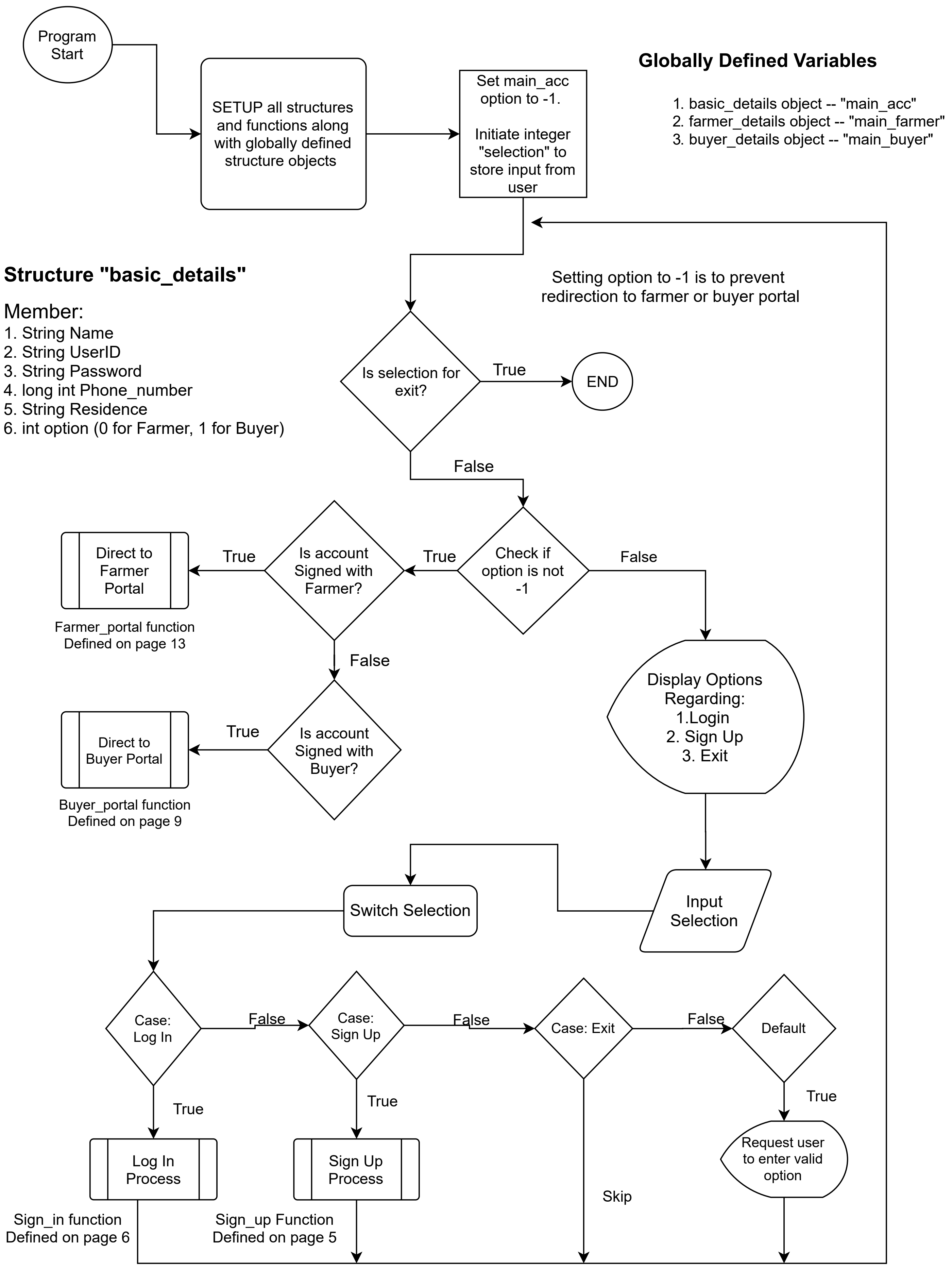
Function-Wise Flowchart

Globally Defined Variables

1. basic_details object -- "main_acc"
2. farmer_details object -- "main_farmer"
3. buyer_details object -- "main_buyer"

Structure "basic_details"

Member:
1. String Name
2. String UserID
3. String Password
4. long int Phone_number
5. String Residence
6. int option (0 for Farmer, 1 for Buyer)



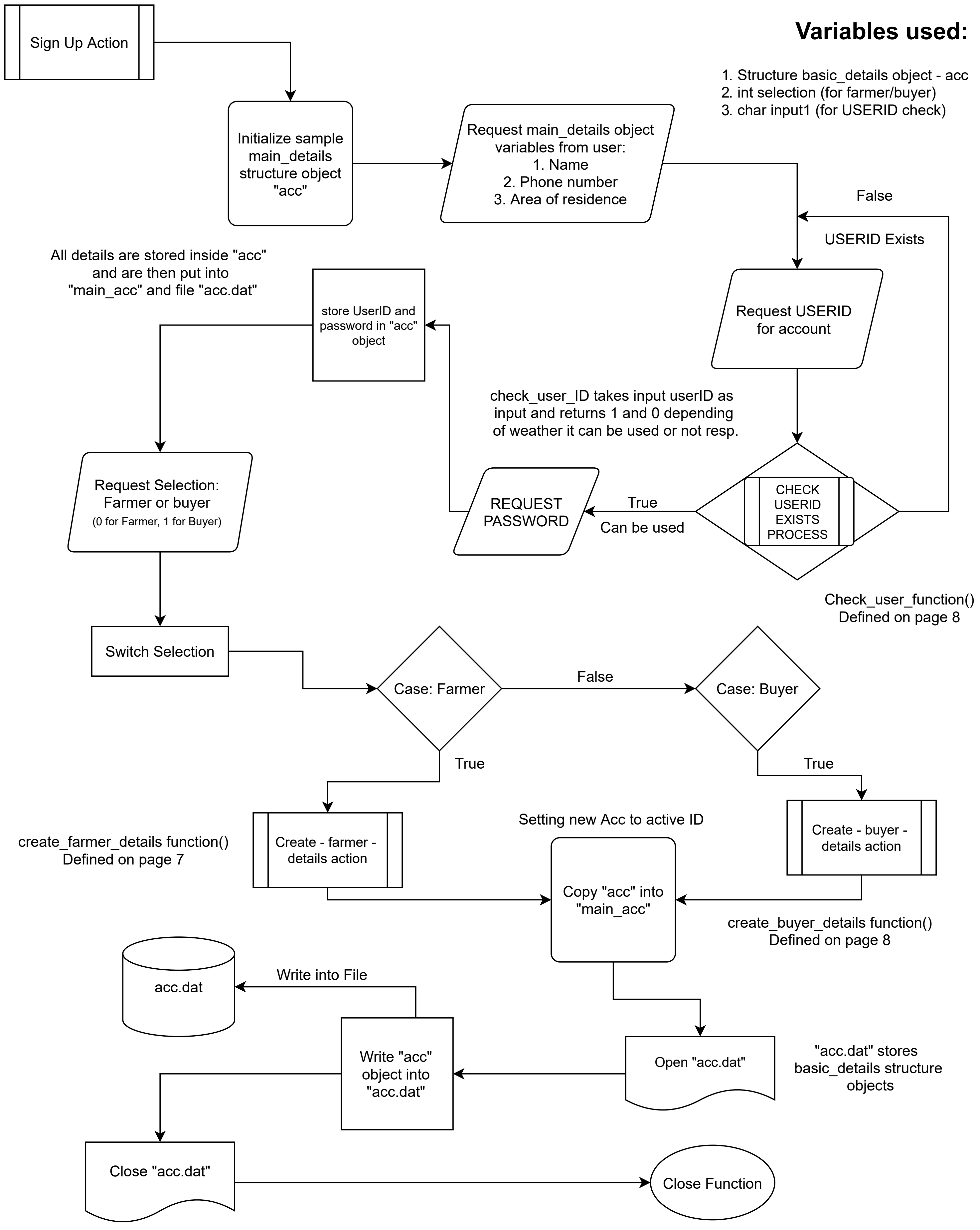
Program Start
SETUP all structures and functions along with globally defined structure objects
Set main_acc option to -1.
Initiate integer "selection" to store input from user
Setting option to -1 is to prevent redirection to farmer or buyer portal
Is selection for exit?
True
END
False
Check if option is not -1
True
Is account Signed with Farmer?
True
Direct to Farmer Portal
Farmer_portal function Defined on page 13
False
Is account Signed with Buyer?
True
Direct to Buyer Portal
Buyer_portal function Defined on page 9
False
Display Options Regarding: 1.Login 2. Sign Up 3. Exit
Input Selection
Switch Selection
Case: Log In
False
Case: Sign Up
False
Case: Exit
False
Default
True
Log In Process
Sign_in function Defined on page 6
True
Sign Up Process
Sign_up Function Defined on page 5
Skip
True
Request user to enter valid option

SIGN UP FUNCTION

Used to Add a new account to the database

Variables used:

1. Structure basic_details object - acc
2. int selection (for farmer/buyer)
3. char input1 (for USERID check)


Sign Up Action
Initialize sample main_details structure object "acc"
Request main_details object variables from user: 1. Name 2. Phone number 3. Area of residence
False
USERID Exists
Request USERID for account
check_user_ID takes input userID as input and returns 1 and 0 depending of weather it can be used or not resp.
CHECK USERID EXISTS PROCESS
Check_user_function() Defined on page 8
True
Can be used
REQUEST PASSWORD
store UserID and password in "acc" object
All details are stored inside "acc" and are then put into "main_acc" and file "acc.dat"
Request Selection: Farmer or buyer (0 for Farmer, 1 for Buyer)
Switch Selection
Case: Farmer
False
Case: Buyer
True
True
Create - farmer - details action
create_farmer_details function() Defined on page 7
Setting new Acc to active ID
Copy "acc" into "main_acc"
Create - buyer - details action
create_buyer_details function() Defined on page 8
acc.dat
Write into File
Write "acc" object into "acc.dat"
Open "acc.dat"
"acc.dat" stores basic_details structure objects
Close "acc.dat"
Close Function

Sign In function

Used to Log-In into a new acc

struct farmer_details

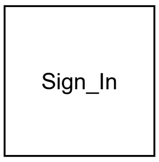
Member

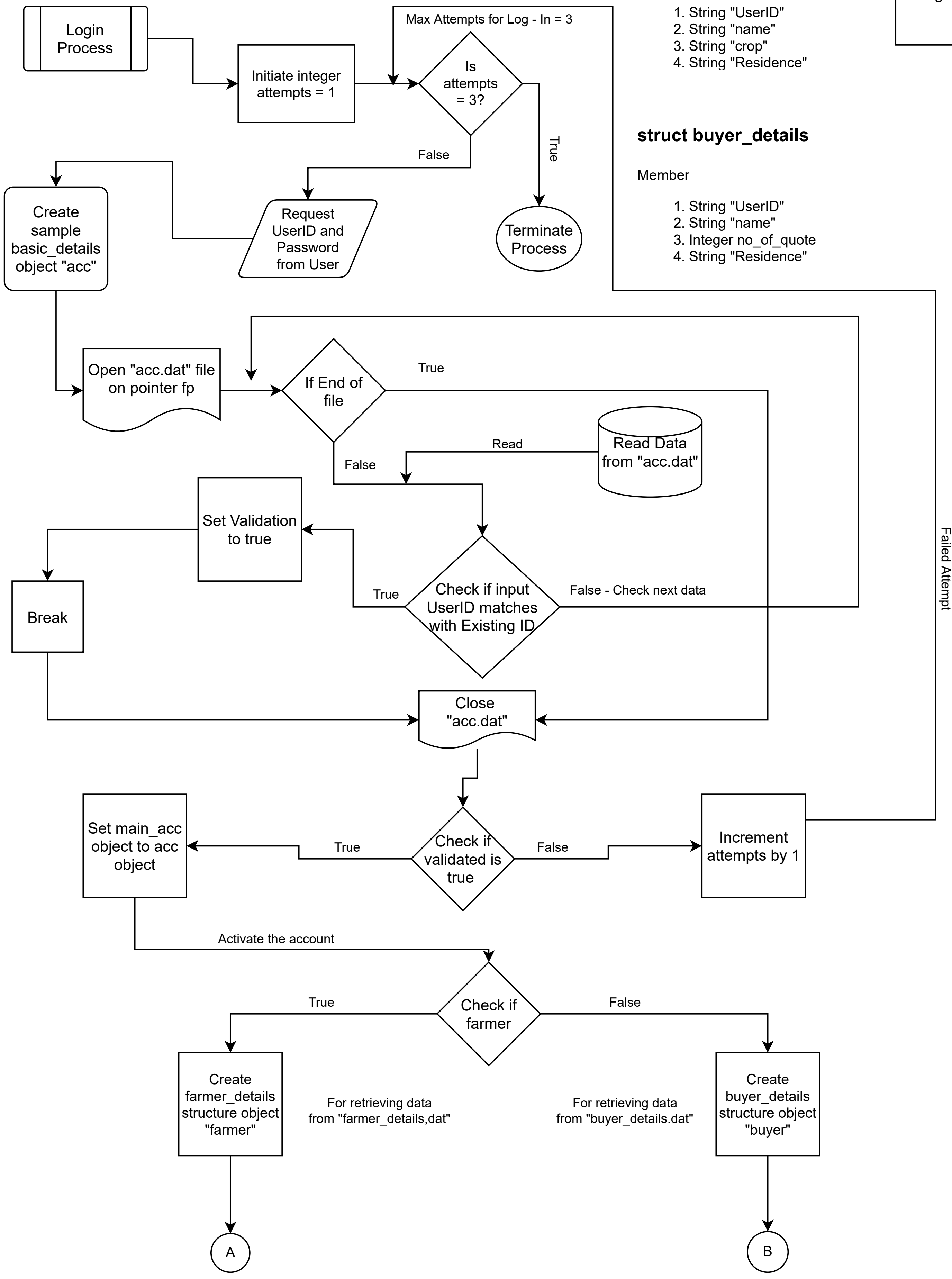
1. String "UserID"
2. String "name"
3. String "crop"
4. String "Residence"

struct buyer_details

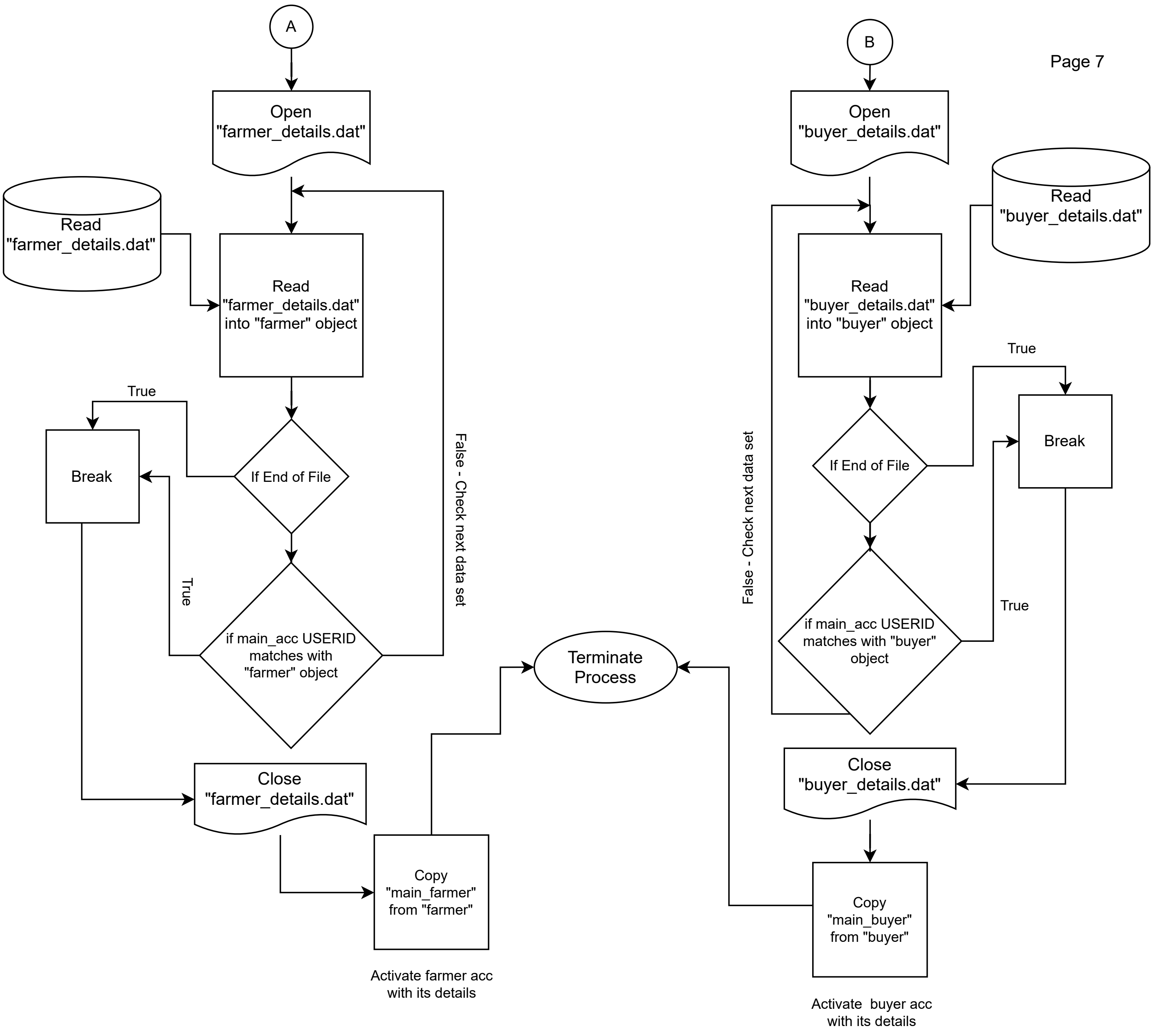
Member

1. String "UserID"
2. String "name"
3. Integer no_of_quote
4. String "Residence"



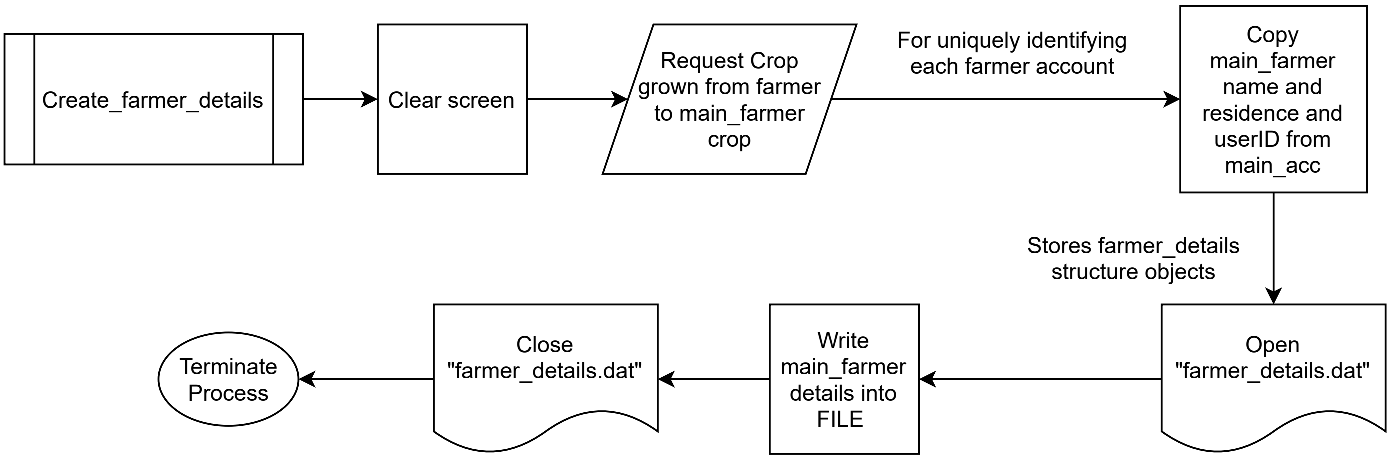


Login Process
Initiate integer attempts = 1
Max Attempts for Log - In = 3
Is attempts = 3?
True
Terminate Process
False
Request UserID and Password from User
Create sample basic_details object "acc"
Open "acc.dat" file on pointer fp
If End of file
True
False
Read
Read Data from "acc.dat"
Check if input UserID matches with Existing ID
True
Set Validation to true
Break
False - Check next data
Close "acc.dat"
Check if validated is true
True
Set main_acc object to acc object
False
Increment attempts by 1
Failed Attempt
Activate the account
Check if farmer
True
Create farmer_details structure object "farmer"
For retrieving data from "farmer_details,dat"
False
Create buyer_details structure object "buyer"
For retrieving data from "buyer_details.dat"
A
B


A
Open "farmer_details.dat"
Read "farmer_details.dat"
Read "farmer_details.dat" into "farmer" object
If End of File
True
Break
if main_acc USERID matches with "farmer" object
True
False - Check next data set
Close "farmer_details.dat"
Copy "main_farmer" from "farmer"
Activate farmer acc with its details
Terminate Process
B
Open "buyer_details.dat"
Read "buyer_details.dat"
Read "buyer_details.dat" into "buyer" object
If End of File
True
Break
if main_acc USERID matches with "buyer" object
True
False - Check next data set
Close "buyer_details.dat"
Copy "main_buyer" from "buyer"
Activate buyer acc with its details

Create_farmer_details

Creates farmer details entry in file and sets to active account

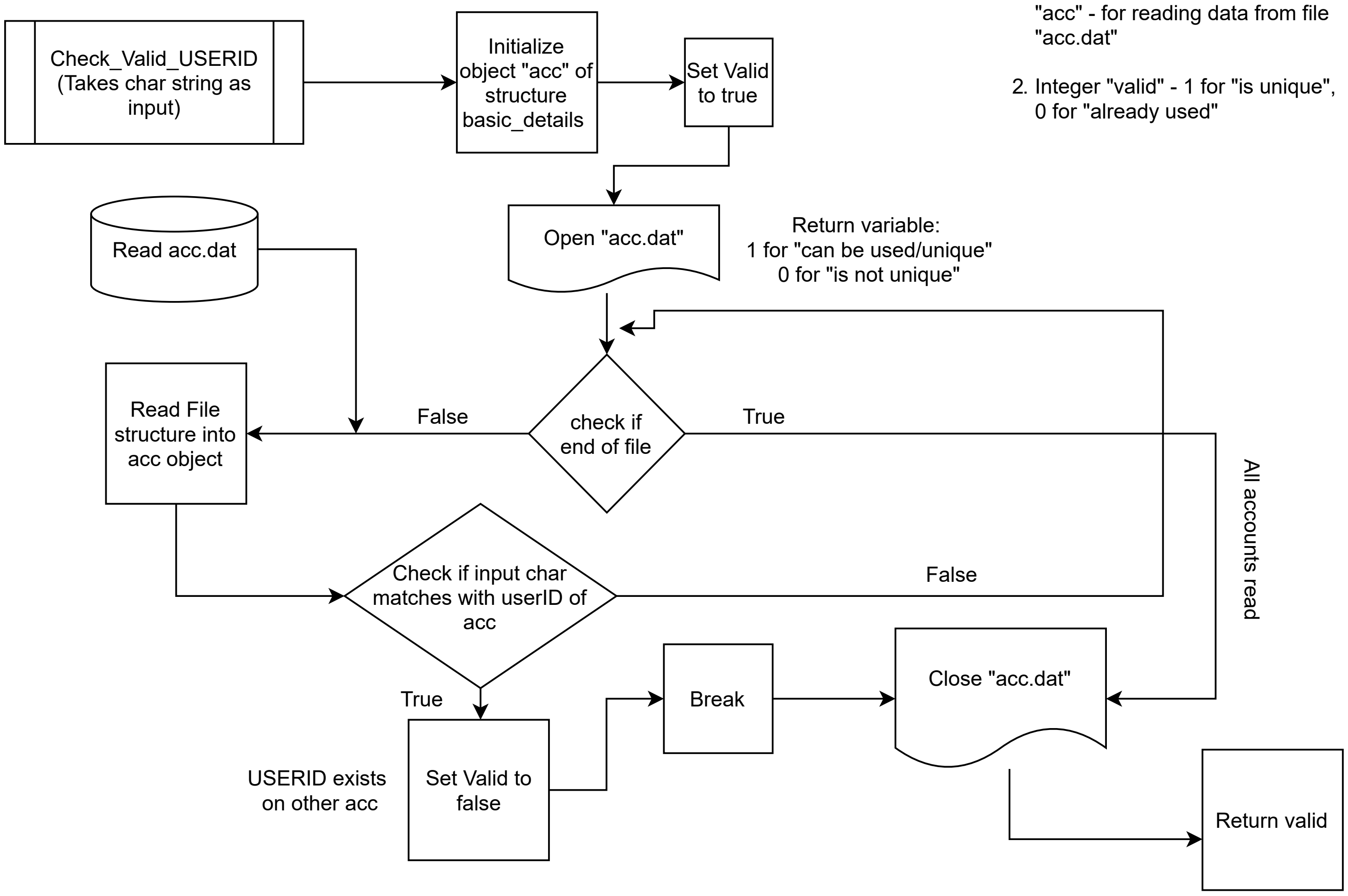

Create_farmer_details
Clear screen
Request Crop grown from farmer to main_farmer crop
For uniquely identifying each farmer account
Copy main_farmer name and residence and userID from main_acc
Stores farmer_details structure objects
Open "farmer_details.dat"
Write main_farmer details into FILE
Close "farmer_details.dat"
Terminate Process

check_valid_userID

Used to check if the string the function is called is unique compared to other IDs

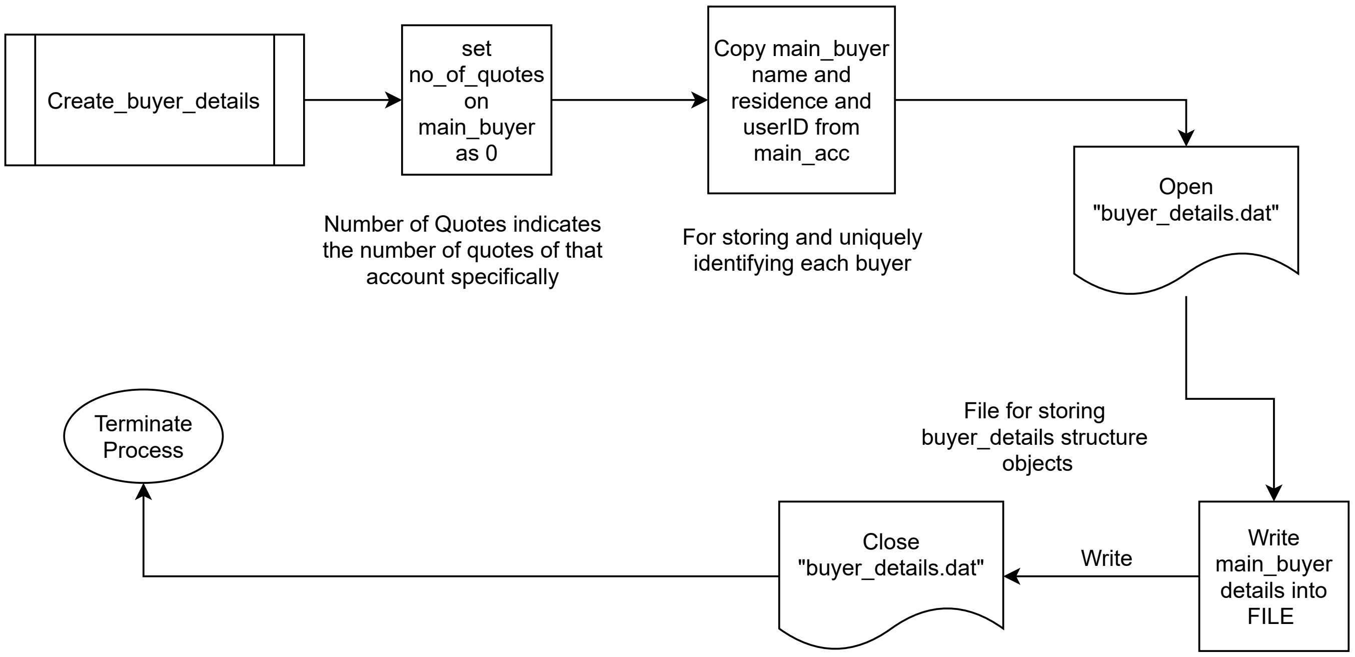
Variables used

1. Basic_details structure object "acc" - for reading data from file "acc.dat"
2. Integer "valid" - 1 for "is unique", 0 for "already used"


Check_Valid_USERID (Takes char string as input)
Initialize object "acc" of structure basic_details
Set Valid to true
Open "acc.dat"
Return variable: 1 for "can be used/unique" 0 for "is not unique"
Read acc.dat
Read File structure into acc object
False
check if end of file
True
All accounts read
Check if input char matches with userID of acc
False
True
USERID exists on other acc
Set Valid to false
Break
Close "acc.dat"
Return valid

create_buyer_details

Creates new buyer account and makes it active.

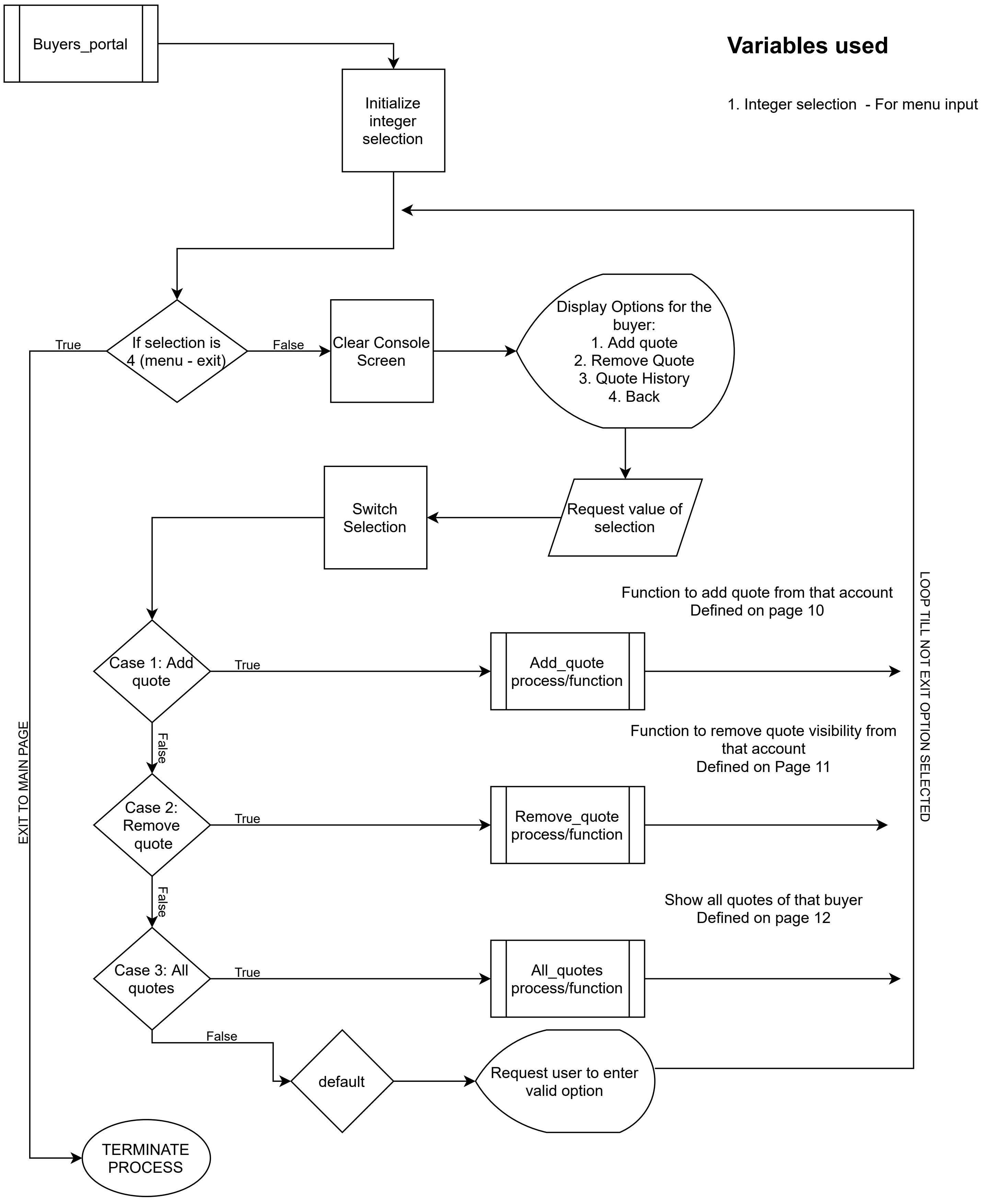

Create_buyer_details
set no_of_quotes on main_buyer as 0
Copy main_buyer name and residence and userID from main_acc
Open "buyer_details.dat"
Number of Quotes indicates the number of quotes of that account specifically
For storing and uniquely identifying each buyer
File for storing buyer_details structure objects
Terminate Process
Close "buyer_details.dat"
Write
Write main_buyer details into FILE

Buyer Portal

Displays Options for user if Buyer

Variables used

1. Integer selection - For menu input



Buyers_portal
Initialize integer selection
If selection is 4 (menu - exit)
True
False
Clear Console Screen
Display Options for the buyer:
1. Add quote
2. Remove Quote
3. Quote History
4. Back
Request value of selection
Switch Selection
Function to add quote from that account
Defined on page 10
Case 1: Add quote
True
Add_quote process/function
False
Function to remove quote visibility from that account
Defined on Page 11
Case 2: Remove quote
True
Remove_quote process/function
False
Show all quotes of that buyer
Defined on page 12
Case 3: All quotes
True
All_quotes process/function
False
default
Request user to enter valid option
LOOP TILL NOT EXIT OPTION SELECTED
EXIT TO MAIN PAGE
TERMINATE PROCESS

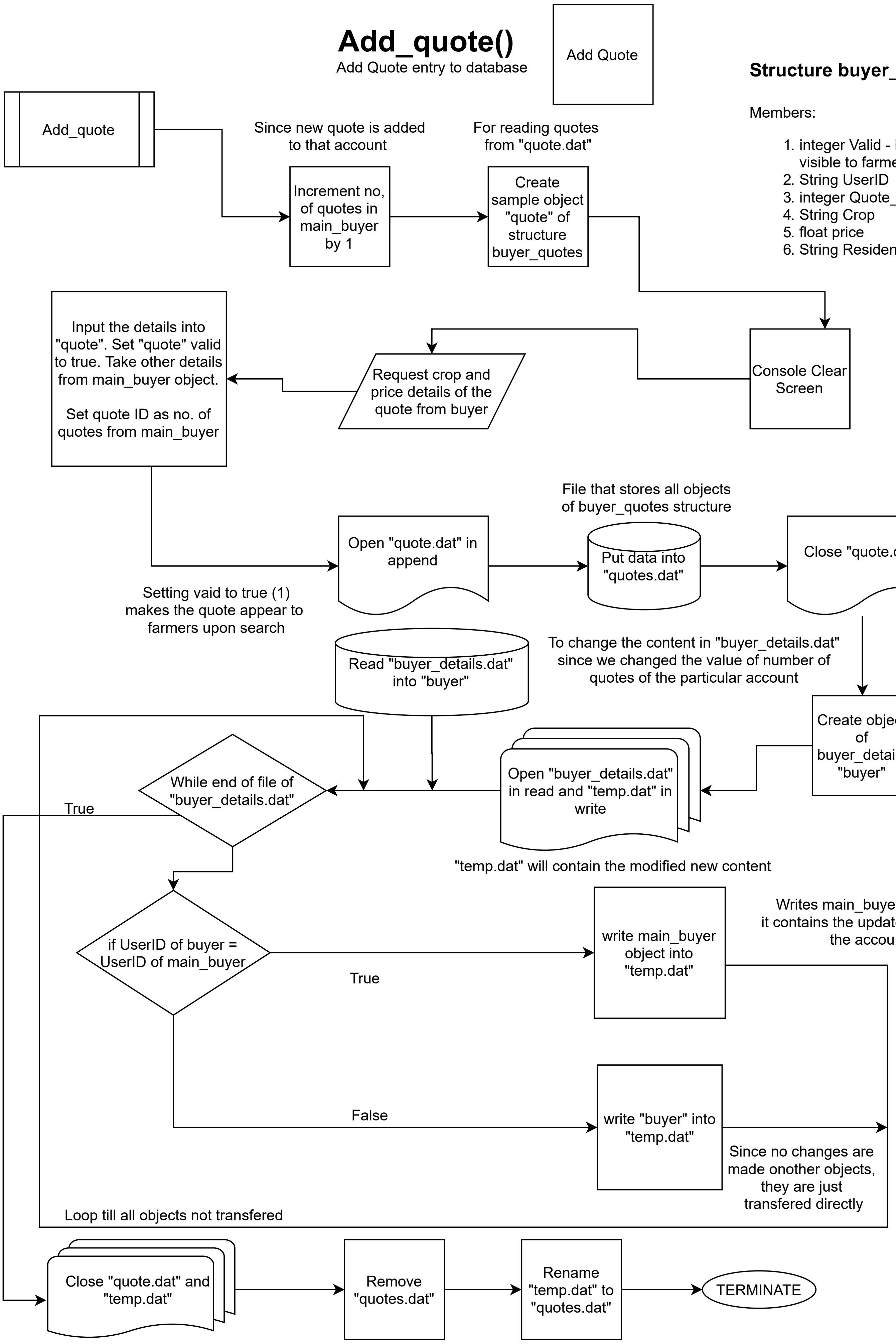
Add_quote()

Add Quote entry to database

Structure buyer_quotes

Members:

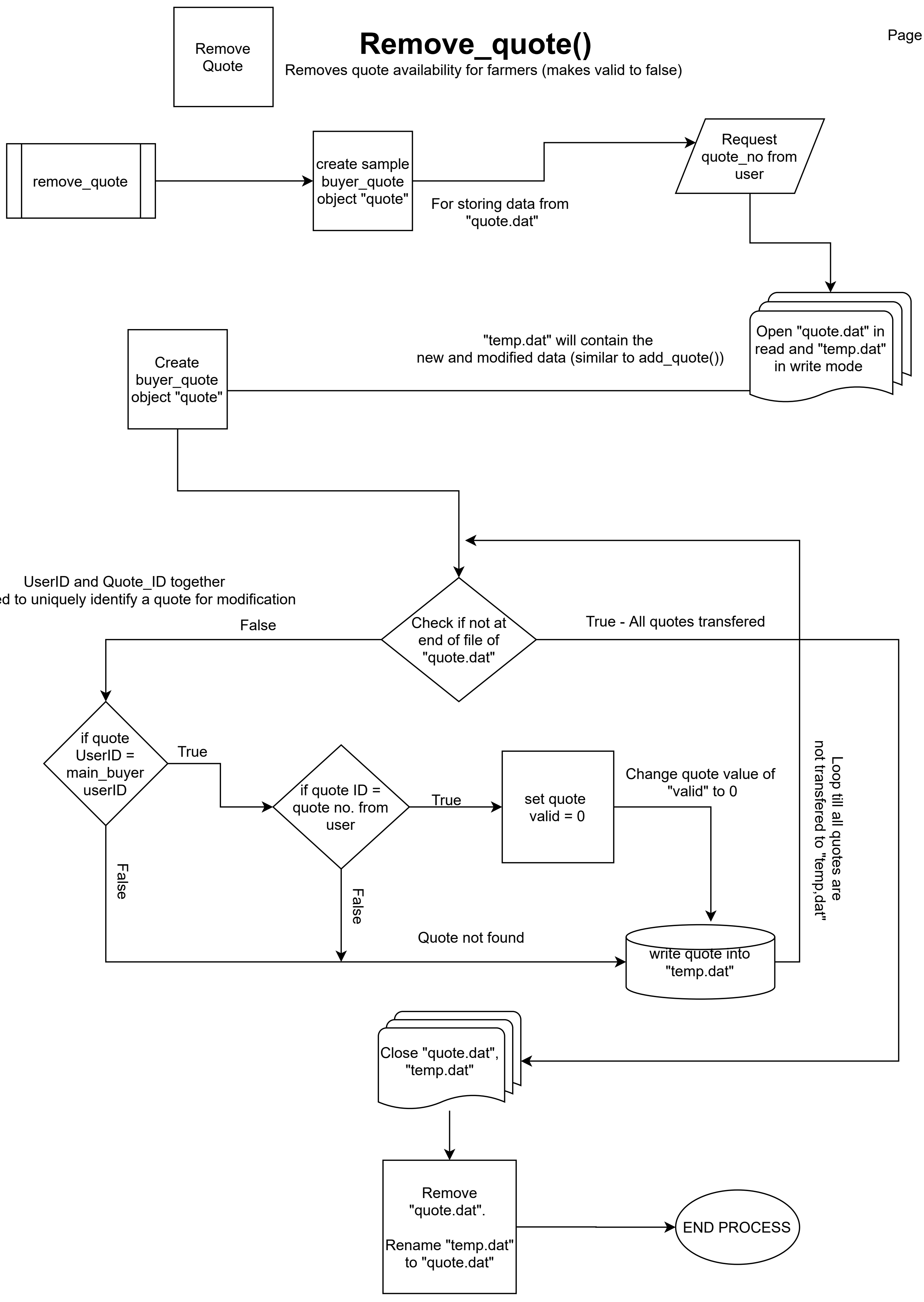
1. integer Valid - if valid then visible to farmers
2. String UserID
3. integer Quote_no
4. String Crop
5. float price
6. String Residence



Add Quote
Add_quote
Since new quote is added to that account
Increment no, of quotes in main_buyer by 1
For reading quotes from "quote.dat"
Create sample object "quote" of structure buyer_quotes
Console Clear Screen
Request crop and price details of the quote from buyer
Input the details into "quote". Set "quote" valid to true. Take other details from main_buyer object.
Set quote ID as no. of quotes from main_buyer
Open "quote.dat" in append
File that stores all objects of buyer_quotes structure
Put data into "quotes.dat"
Close "quote.dat"
Setting vaid to true (1) makes the quote appear to farmers upon search
Read "buyer_details.dat" into "buyer"
To change the content in "buyer_details.dat" since we changed the value of number of quotes of the particular account
Create object of buyer_details "buyer"
Open "buyer_details.dat" in read and "temp.dat" in write
"temp.dat" will contain the modified new content
While end of file of "buyer_details.dat"
True
if UserID of buyer = UserID of main_buyer
True
write main_buyer object into "temp.dat"
Writes main_buyer because it contains the updated details of the account
False
write "buyer" into "temp.dat"
Since no changes are made onother objects, they are just transfered directly
Loop till all objects not transfered
Exit from loop
Close "quote.dat" and "temp.dat"
Remove "quotes.dat"
Rename "temp.dat" to "quotes.dat"
TERMINATE

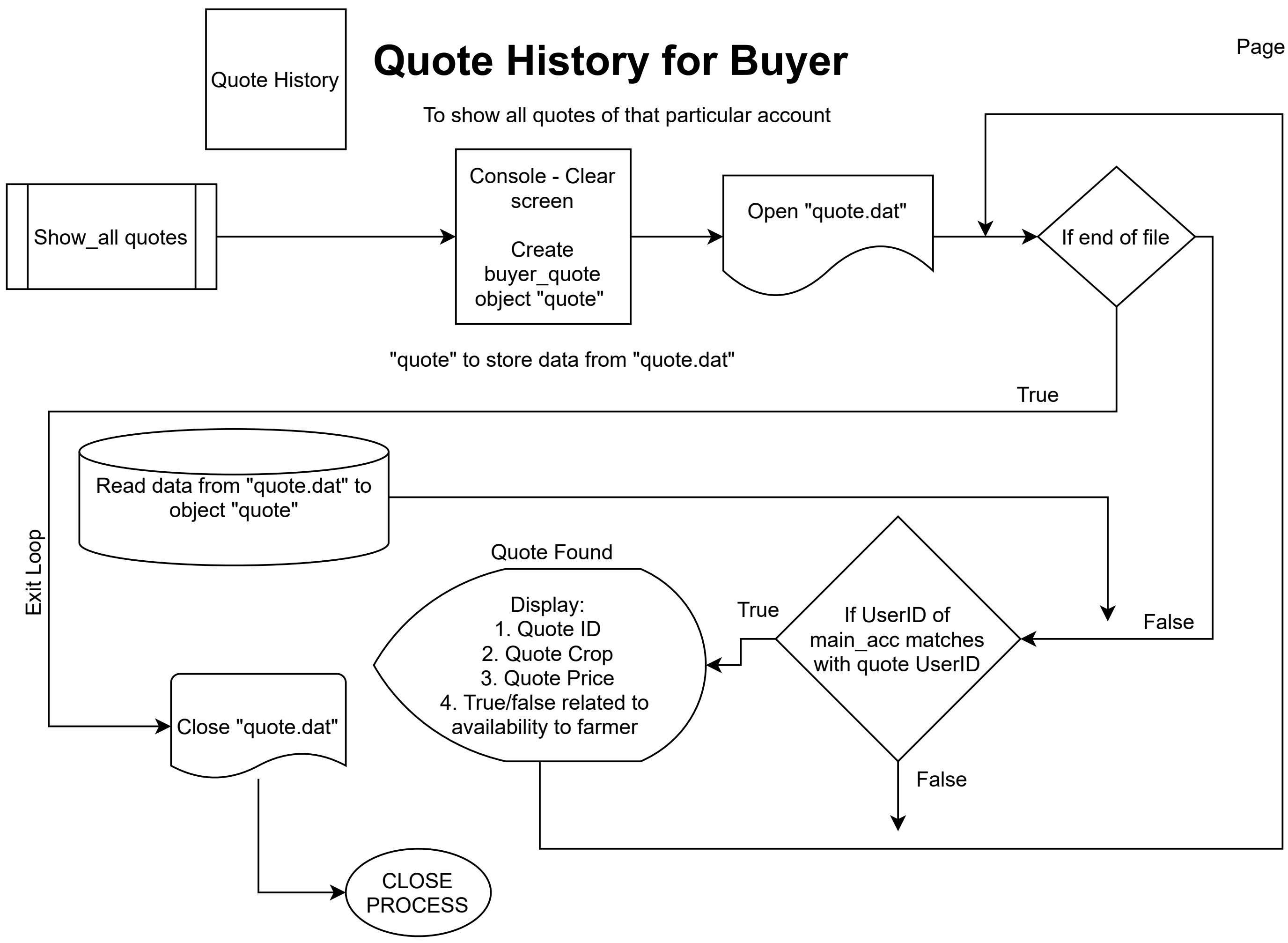
Remove_quote()

Removes quote availability for farmers (makes valid to false)


Remove Quote
remove_quote
create sample buyer_quote object "quote"
For storing data from "quote.dat"
Request quote_no from user
Open "quote.dat" in read and "temp.dat" in write mode
"temp.dat" will contain the new and modified data (similar to add_quote())
Create buyer_quote object "quote"
UserID and Quote_ID together are used to uniquely identify a quote for modification
False
Check if not at end of file of "quote.dat"
True - All quotes transfered
if quote UserID = main_buyer userID
True
if quote ID = quote no. from user
True
set quote valid = 0
Change quote value of "valid" to 0
Loop till all quotes are not transfered to "temp.dat"
False
False
Quote not found
write quote into "temp.dat"
Close "quote.dat", "temp.dat"
Remove "quote.dat". Rename "temp.dat" to "quote.dat"
END PROCESS

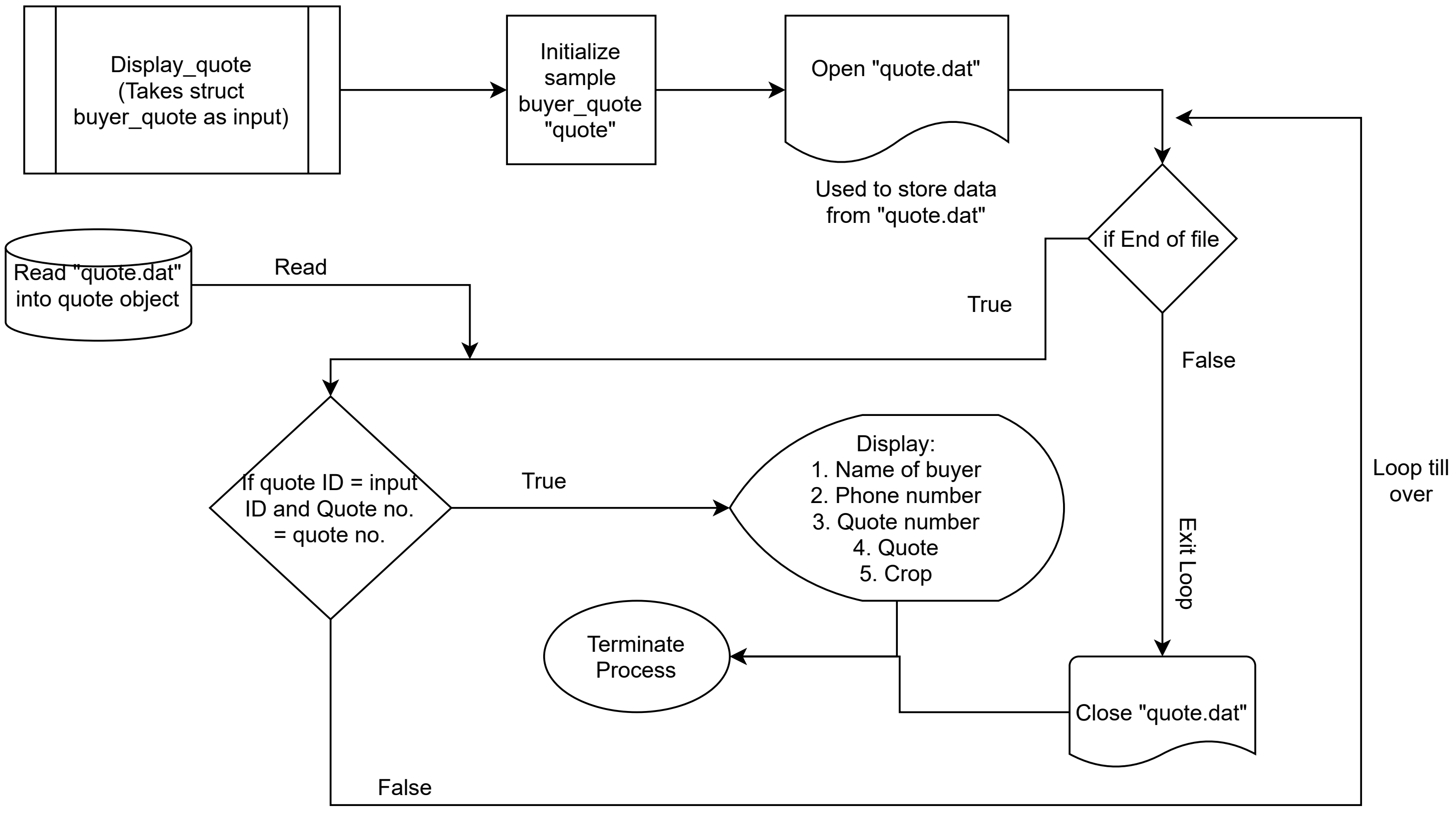
Quote History for Buyer

To show all quotes of that particular account


Quote History
Show_all quotes
Console - Clear screen
Create buyer_quote object "quote"
"quote" to store data from "quote.dat"
Open "quote.dat"
If end of file
True
Exit Loop
Read data from "quote.dat" to object "quote"
Quote Found
Display:
1. Quote ID
2. Quote Crop
3. Quote Price
4. True/false related to availability to farmer
True
If UserID of main_acc matches with quote UserID
False
False
Close "quote.dat"
CLOSE PROCESS

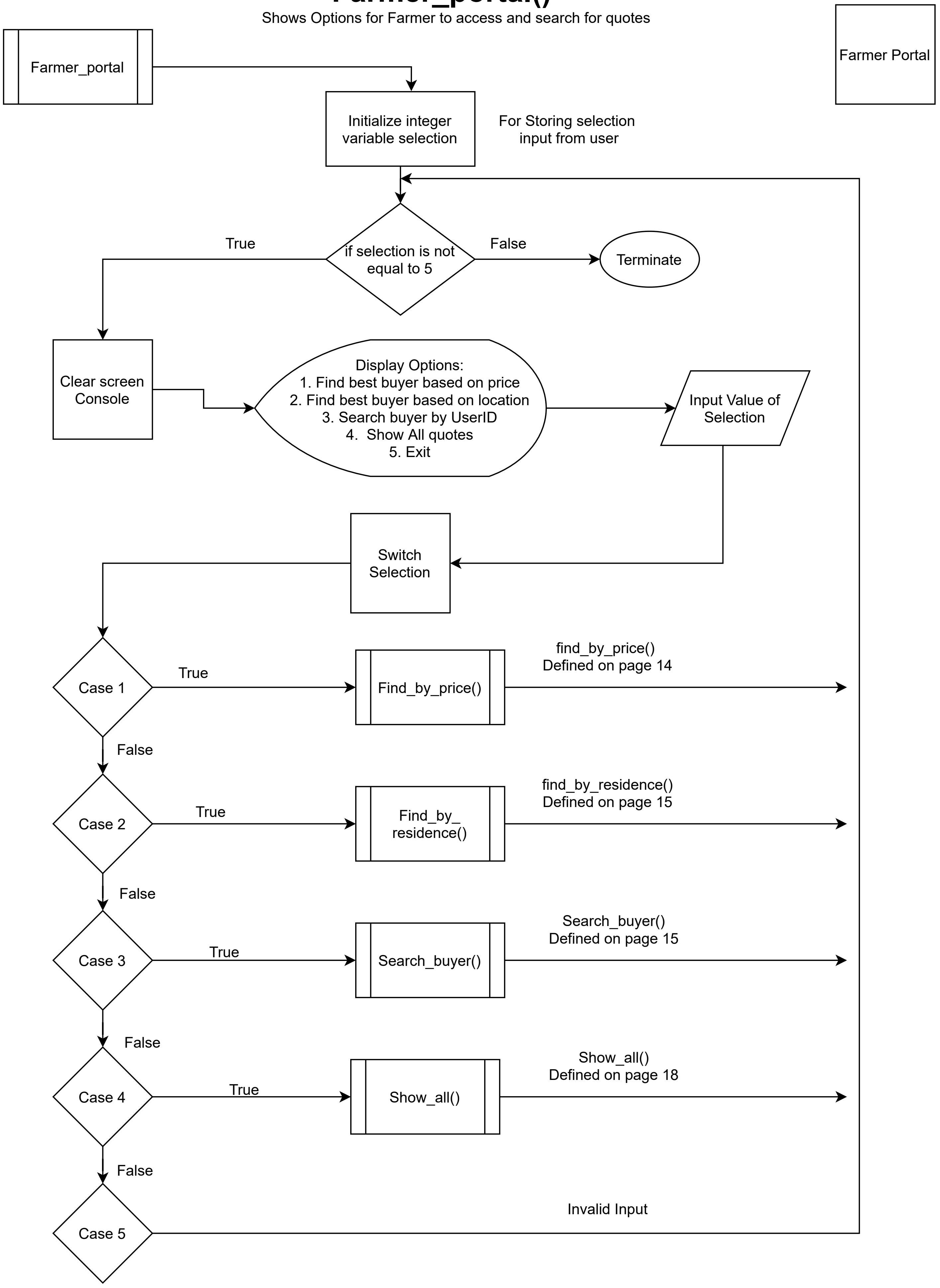
Display_quote()

Takes in structure buyer_quotes object as input and displays its corresponding details


Display_quote (Takes struct buyer_quote as input)
Initialize sample buyer_quote "quote"
Open "quote.dat"
Used to store data from "quote.dat"
if End of file
Read "quote.dat" into quote object
Read
True
False
Loop till over
If quote ID = input ID and Quote no. = quote no.
True
Display:
1. Name of buyer
2. Phone number
3. Quote number
4. Quote
5. Crop
Exit Loop
Terminate Process
Close "quote.dat"
False

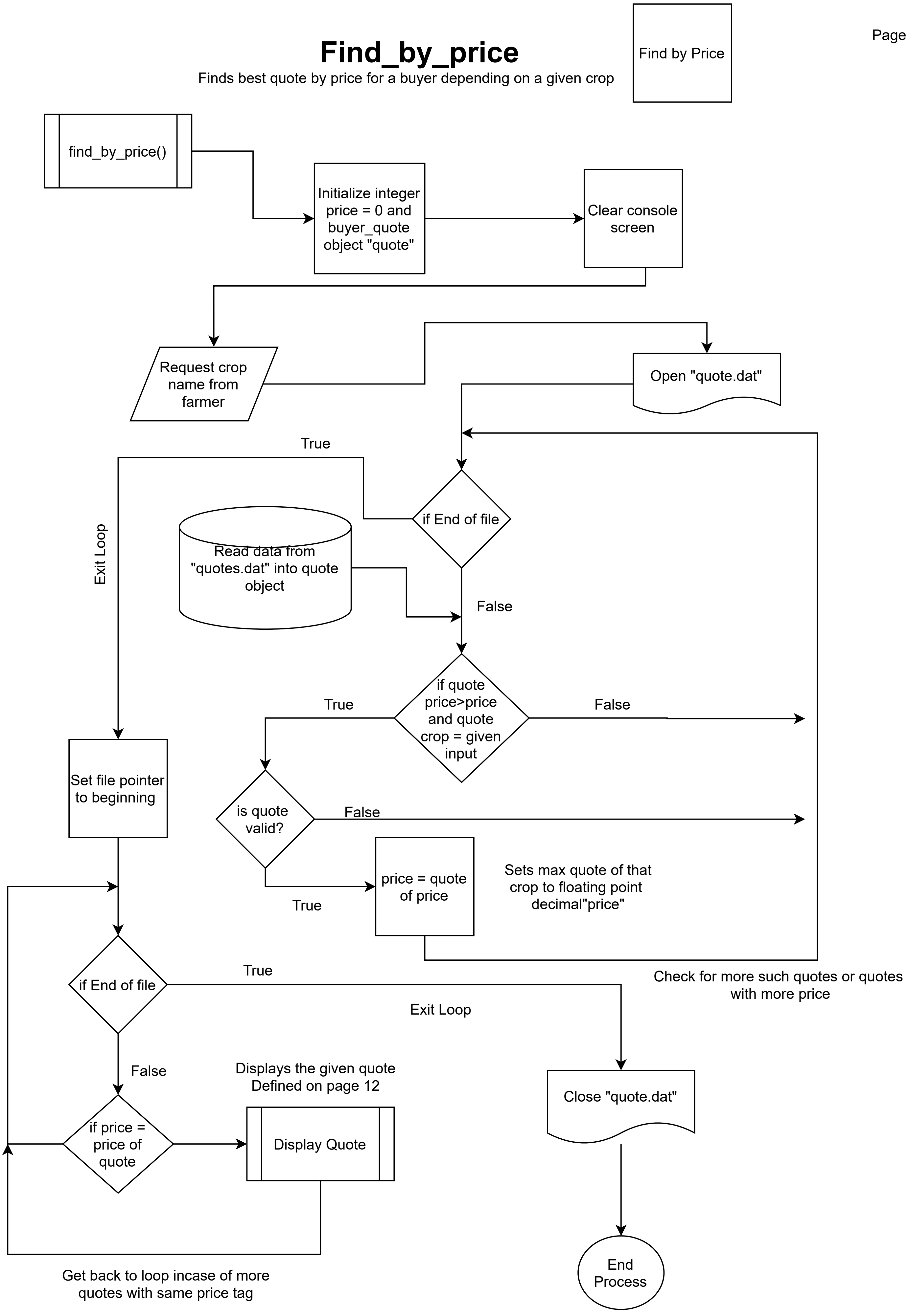
Farmer_portal()

Shows Options for Farmer to access and search for quotes



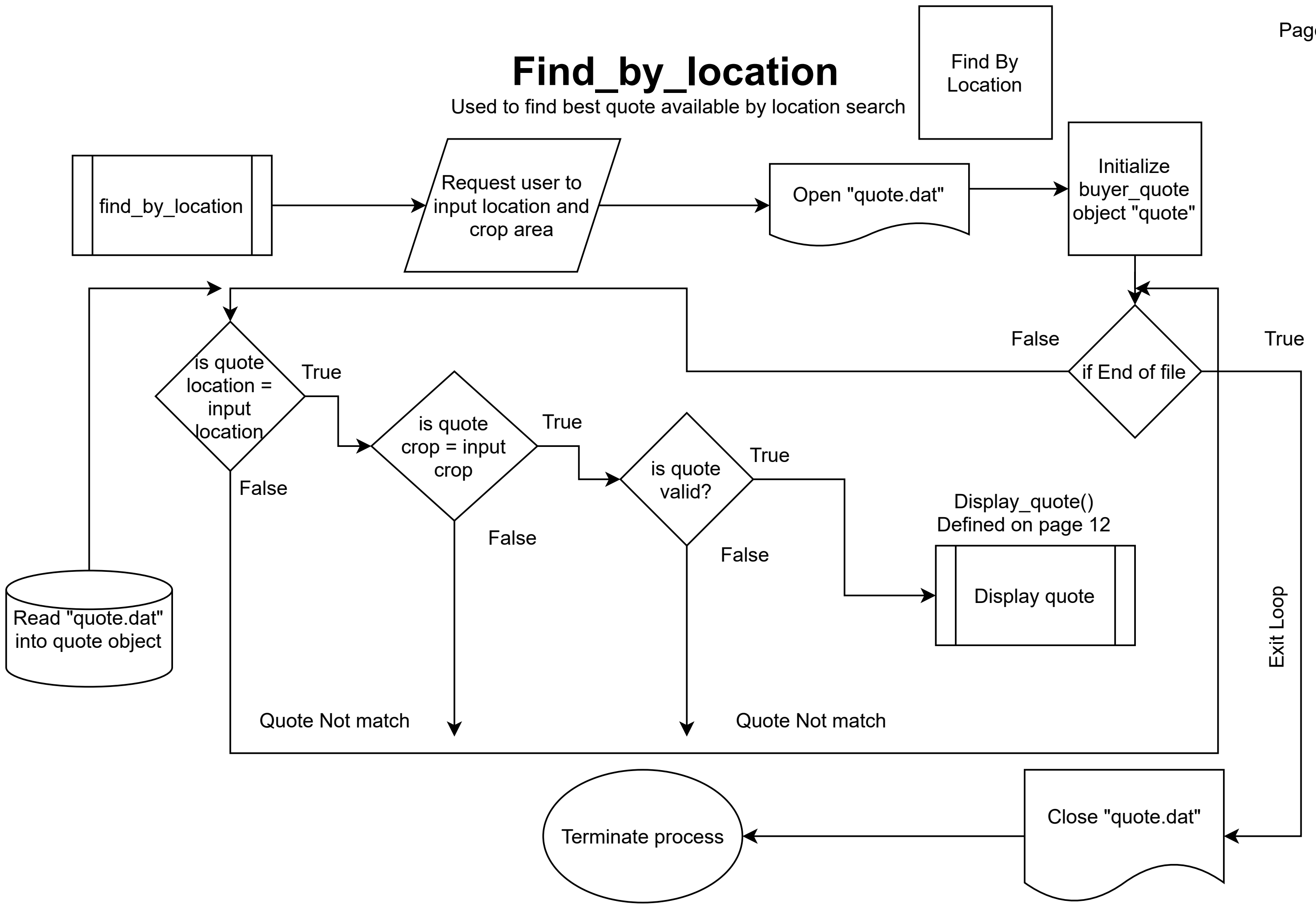
Find_by_price

Finds best quote by price for a buyer depending on a given crop


Find by Price
find_by_price()
Initialize integer price = 0 and buyer_quote object "quote"
Clear console screen
Request crop name from farmer
Open "quote.dat"
True
Exit Loop
if End of file
Read data from "quotes.dat" into quote object
False
if quote price>price and quote crop = given input
True
False
Set file pointer to beginning
is quote valid?
False
price = quote of price
True
Sets max quote of that crop to floating point decimal"price"
if End of file
True
Check for more such quotes or quotes with more price
Exit Loop
False
Displays the given quote Defined on page 12
Close "quote.dat"
if price = price of quote
Display Quote
End Process
Get back to loop incase of more quotes with same price tag

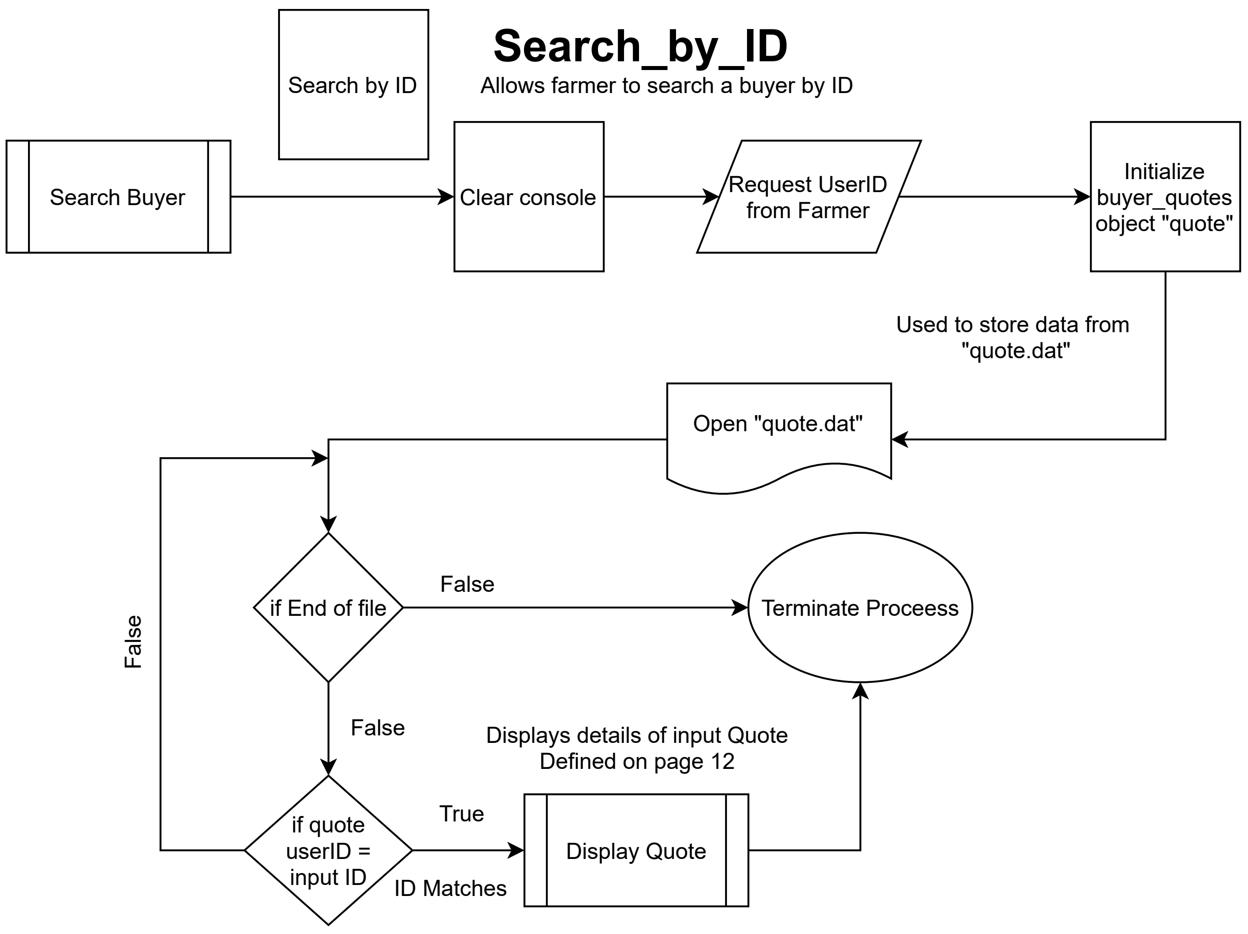
Find_by_location

Used to find best quote available by location search


Find By Location
find_by_location
Request user to input location and crop area
Open "quote.dat"
Initialize buyer_quote object "quote"
False
if End of file
True
is quote location = input location
True
False
is quote crop = input crop
True
False
is quote valid?
True
False
Display_quote() Defined on page 12
Display quote
Read "quote.dat" into quote object
Exit Loop
Quote Not match
Quote Not match
Terminate process
Close "quote.dat"

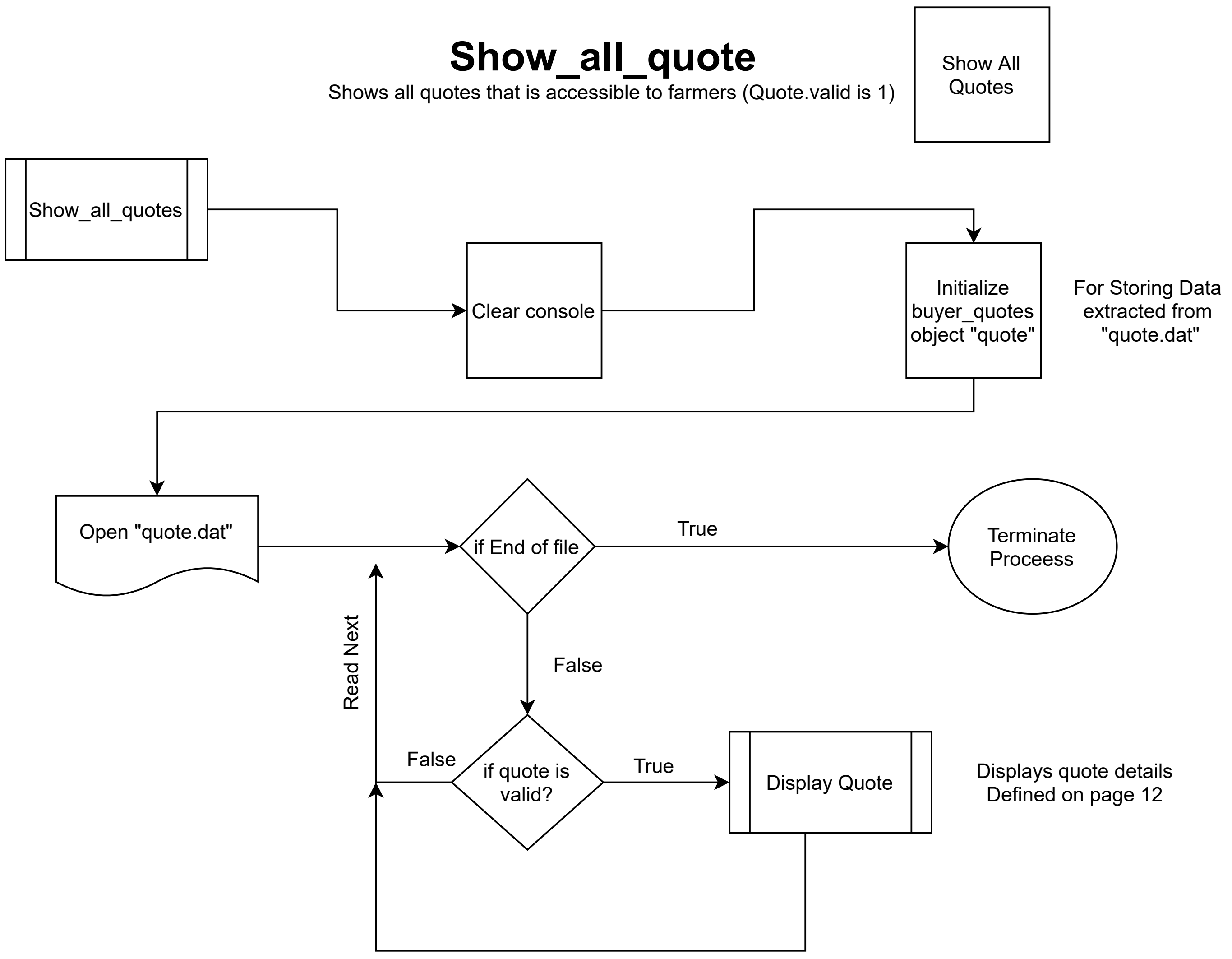
Search_by_ID

Allows farmer to search a buyer by ID


Search by ID
Search Buyer
Clear console
Request UserID from Farmer
Initialize buyer_quotes object "quote"
Used to store data from "quote.dat"
Open "quote.dat"
if End of file
False
Terminate Proceess
False
False
Displays details of input Quote Defined on page 12
if quote userID = input ID
True
Display Quote
ID Matches

Show_all_quote

Shows all quotes that is accessible to farmers (Quote.valid is 1)



4 Source Code

This section of the report presents the source code for project - FARMHOUSE

farmhouse.c

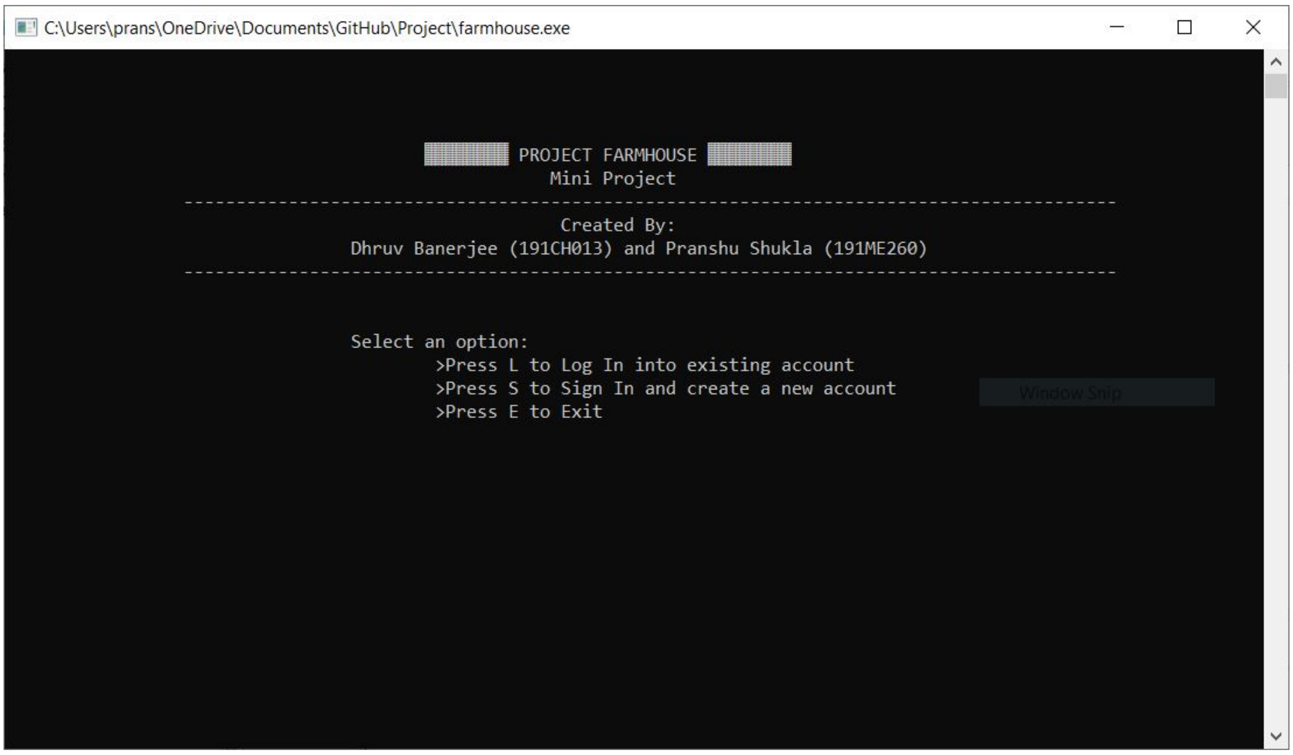
```
/*
Project - FARMHOUSE
CS110 Mini Project
Team: 14
Team Members:
    1. Dhruv Banerjee, 191CH013, 9428418165, dbanerjee.191ch013@nitk.edu.in
    2. Pranshu Shukla, 191ME260, 7385925943, pranshushukla.191me260@nitk.edu.in
*/

#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>



//---------------- STRUCTURE DECLARATION--------------
struct basic_details //Structure to store Basic Details of any user like UserID, Password etc
{
    char userID[20]; //Unique UserID of each user, used to uniquely identify each user
    char password[20]; //Password set by each user to safeguard his account
    char name[20]; //Name of user
    long int phno; //Phone number
    char residence[20]; //Residence
    int option; //0 for farmer, 1 for buyer for identification
};

struct farmer_details //Structure to store details of Farmers if user is a farmer
{
    char userID[20]; //Links this structure to basic_details structure since UserID is unique
    char name[20];
    char crop[20]; //Crop grown by farmer
    char residence[20];
};

struct buyer_details //Structure to store details of Farmers if user is a farmer
{
    char userID[20]; //Links this structure to basic_details structure since UserID is unique
    char name[20];
    char residence[20];
    int no_of_quotes; //Number of quotes raised or issued by the farmer throughout the existence of his/her account.
                      //Is used for quote_no in buyer_quote strucute
};

struct buyer_quotes //Structure for storing all quotes that are raised by farmers
{
    int valid; //if quote is valid (=1), it can be viewed by farmers upon search
    char userID[20]; //Links buyer_quotes to respective User
    int quote_no; //Is the value of the current "no_of_quotes" value from buyer_details. UserID together with quote_no uniquely identifies each quote.
    char crop[20];
    float price;
    char residence[20];
};

struct basic_details main_acc; //For storing details of active User
struct farmer_details main_farmer; //For storing details of active user if Farmer
struct buyer_details main_buyer; //For Storing details of active user if Buyer


//------------------FUNCTIONS DECLARATION-------------------------
```

```
void create_farmer_details(); //Function to create farmer object for new acc
void create_buyer_details(); //Function to create farmer object for new acc
void farmer_portal(); //Farmer Portal containing all features for farmer account
void search_buyer(); //Search By ID
void buyer_portal(); //Buyers Portal containing all features for buyer account
void find_by_location(); //Find Closest Buyers
void add_quote(); //Adding quote to a buyer
void display_buyer( struct buyer_quotes quotes); //Displaying Quotes
void remove_quote(); //Remove Existing Quote from Visibility
void show_all_buyers(); //Show all visible Quotes
void all_quotes(); //Showing all Quotes of the buyer (Quote History)
int check_valid_userID(char check[20]);//Function to check if given userID is unique or not during registeration process
void find_by_price(); //Finding best quote price for a crop
void SignIn(); //Log In into existing account
void SignUp(); //Function to Create new users

//Main screen - First Screen User Visits when program is run
int main()
{
    char ch = 'a';
    main_acc.option = -1; //To prevent redirection into Farmer or buyer Portal
    while(ch!='e')
    {
        if(main_acc.option==0) //If farmer
        {
            farmer_portal(); //Send to Farmer portal
        }
        if(main_acc.option==1) // If Buyer
        {
            buyer_portal(); //Send to Buyer Portal
        }
        system("cls");
        printf("\n\n\n\n");
        printf("\t\t\t\t\t\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2 PROJECT FARMHOUSE \xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\n");
        printf("\t\t\t\t\t\t    Mini Project\n");
        printf("\t\t ------------------------------------------------------------------------------------------\n");
        printf("\t\t\t\t\t\t     Created By:\n");
        printf("\t\t\t\t Dhruv Banerjee (191CH013) and Pranshu Shukla (191ME260)\n");
        printf("\t\t ------------------------------------------------------------------------------------------\n");
        printf("\n\n");
        printf("\t\t\t\t Select an option: \n");
        printf("\t\t\t\t\t >Press L to Log In into existing account \n");
        printf("\t\t\t\t\t >Press S to Sign In and create a new account\n");
        printf("\t\t\t\t\t >Press E to Exit \n");
        ch = getch();
        ch = tolower(ch); //To convert to lower Case
        switch(ch)
        {
            case 108: SignIn(); //ASCII VALUE OF 'l' is 108
            break;

            case 115: SignUp(); //ASCII Value of 's' is 115
            break;

            case 101: break; //ASCI Value of 'e' is 101

            default: printf("\n\n\t\t\t\t\t *Please Enter valid input*\n");
            printf("\t\t\t\t\t Press any key to continue...");
            getch();
        }
    }
}

//---------------------------- Function to Create new users ----------------------------------
void SignUp()
{

    struct basic_details acc; //temporary structure object for storing details
    char ch = 'a'; //To prevent mis-direction
```

```
int nxt = 0; //To continue in the registration process
while(ch!=’e’ && !nxt) //looping and exit condition e-->exit
{
    system("cls");
    printf("\n\n\n\n");
    printf("\t\t\t\t\t\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2 PROJECT FARMHOUSE \xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\n");
    printf("\t\t\t\t\t\t    Mini Project\n");
    printf("\t\t -------------------------------------------------------------------------------------------\n");
    printf("\t\t\t\t\t\tNew Account Registeration\n");
    printf("\t\t -------------------------------------------------------------------------------------------\n");
    printf("\t\t\t\t\t  Enter the kind of User to be Created: \n");
    printf("\t\t\t\t\t\t >Press F for Farmer \n");
    printf("\t\t\t\t\t\t >Press B for Buyer \n");
    printf("\t\t\t\t\t\t >Press E to return to previous screen \n");
    ch = getch();
    ch = tolower(ch);
    printf("%c",ch);
    switch(ch)
    {
        case 102: acc.option=0; //For Farmer
        nxt = 1;
        break;

        case 98: acc.option=1; //For Buyer
        nxt = 1;
        break;

        case 101: break; //Exit condition

        default: printf("\n\n\t\t\t\t\t\t *Please Enter valid input*\n");
        printf("\t\t\t\t\t\t Press any key to continue...");
        getch();
    }
}

if(ch!=’e’)
{
    system("cls");
    printf("\n\n\n\n");
    printf("\t\t\t\t\t\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2 PROJECT FARMHOUSE \xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\n");
    printf("\t\t\t\t\t\t    Mini Project\n");
    printf("\t\t -------------------------------------------------------------------------------------------\n");
    printf("\t\t\t\t\t\tNew Account Registeration\n");
    printf("\t\t -------------------------------------------------------------------------------------------\n");
    FILE *fp;
    fp = fopen("acc.dat","ab"); //File storing "Basic_details" structure objects
    printf("\t\t\t\t\t Enter Name: ");
    fflush(stdin); //to clear buffer which might interfere input of string
    gets(acc.name);
    printf("\t\t\t\t\t Enter Phone number: ");
    scanf("%ld",&acc.phno);
    printf("\t\t\t\t\t Enter area of residence: ");
    fflush(stdin);
    gets(acc.residence);

    while(1)
    {
        system("cls");
        printf("\n\n\n\n");
        printf("\t\t\t\t\t\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2 PROJECT FARMHOUSE \xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\n");
        printf("\t\t\t\t\t\t    Mini Project\n");
        printf("\t\t -------------------------------------------------------------------------------------------\n");
        printf("\t\t\t\t\t\tNew Account Registeration\n");
        printf("\t\t -------------------------------------------------------------------------------------------\n");
        printf("\t\t\t\t Enter USER-ID for your Account: ");
        char input[20];
        gets(input);
        if(check_valid_userID(input)) //check_valid_userID returns 1 if input userID is unique and 0 for not unique
        {
                strcpy(acc.userID,input); //If input UserID is unique, it is put into temp object
                char input2[20],ch1;
                int i =0; //index of password string
                printf("\n");
                printf("\t\t\t\t Enter Password: ");
                while(1)
                {
                    ch1 = getch();
                    if(ch1 == '\r') //return character as loop exit condition
                    {
                        break;
                    }
                    else
                    {
                        input2[i] = ch1;
                        i++;
                        printf("*");
                    }
                }
                input2[i] = '\0';
                strcpy(acc.password,input2); //Copy the password to temp object
                break;
            }
            else
            {
                printf("\n\n");
                printf("\t\t\t\t *User ID already Exists. Try another ID.*\n"); //If UserID is not unique i.e. some other User exists with given ID
                printf("\t\t\t\t Press any key to continue...");
                getch();
            }
        }

        fwrite(&acc,sizeof(acc),1,fp); //Writing the structure to file for storing
        fclose(fp);

        main_acc = acc; //Making new account as active account

        if(acc.option == 0) //If farmer
        {
            create_farmer_details(); //Go to Farmer Details function which will create farmer acc object
        }
        else
        {
            create_buyer_details(); //Else if Buyer,go to Buyer Details function which will create Buyer acc object
        }

    }
}


//---------------------------- Log In into existing account ----------------------------------
void SignIn()
{
    int tries = 1; //Max Tries to log-In incorrectly = 3
    while(tries!=4)
    {
        FILE *fp;
        fp = fopen("acc.dat","rb");
        struct basic_details acc;
        char input1[20];
        system("cls");
        printf("\n\n\n\n");
        printf("\t\t\t\t\t\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2 PROJECT FARMHOUSE \xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\n");
        printf("\t\t\t\t\t\t    Mini Project\n");
        printf("\t\t -----------------------------------------------------------------------------------------\n");
        printf("\t\t\t\t\t\t   Account Log In\n");
        printf("\t\t -----------------------------------------------------------------------------------------\n");
        printf("\t\t\t\t\t Enter USER ID: ");
        fflush(stdin);
        gets(input1);
        printf("\t\t\t\t\t Enter Password: ");
        char input2[20], ch1;
        int i = 0;
        while(1) //Password entry
        {
            ch1 = getch();
            if(ch1 == '\r')
            {
                input2[i] = '\0';
                break;
            }
            else
            {
                input2[i] = ch1;
                i++;
                printf("*");
            }
        }
        int validated = 0; //Validated = 1 means account USERID and Password matches with any existing
        while(fread(&acc, sizeof(acc),1,fp)==1)
        {
            if(!strcmp(acc.userID,input1) && !strcmp(acc.password,input2))
            {
                validated = 1;
                break;
            }
        }
        fclose(fp);

        if(validated)
        {
            main_acc = acc; //set to acc to active account, as it matches with the given data
            if(main_acc.option == 0) //if Farmer
            {
                fp = fopen("farmer_details.dat","rb"); //get active Farmer Data
                while(fread(&main_farmer, sizeof(main_farmer),1,fp)==1)
                    {
                        if(!strcmp(main_farmer.userID,input1))
                        {
                            break;
                        }
                    }
                    fclose(fp);
            }
            else
            {
                fp = fopen("buyer_details.dat","rb"); //get active Buyer data
                while(fread(&main_buyer, sizeof(main_buyer),1,fp)==1)
                    {
                        if(!strcmp(main_buyer.userID,input1))
                        {
                            break;
                        }
                    }
                    fclose(fp);
            }
            break;
        }
        else
        {
            printf("\n\n");
            printf("\n\t\t\t\t\t *INVALID ID or PASSWORD. Try Again.* \n");
            printf("\n\t\t\t\t\t Number of Tries Remaining: %d\n",3-tries);
            printf("\t\t\t\t\t Press any key to continue...");
            tries++;
            getch();
        }
    }
```

```
}


//---------------------------- Function to create farmer object for new acc ----------------------------------
void create_farmer_details()
{
    system("cls");
    printf("\n\n\n\n");
    printf("\t\t\t\t\t\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2 PROJECT FARMHOUSE \xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\n");
    printf("\t\t\t\t\t\t    Mini Project\n");
    printf("\t\t -----------------------------------------------------------------------------------------\n");
    printf("\t\t\t\t\t      New Farmer Registeration\n");
    printf("\t\t -----------------------------------------------------------------------------------------\n");
    printf("\t\t\t\t\t Enter crop grown: ");
    strcpy(main_farmer.name,main_acc.name);
    strcpy(main_farmer.residence,main_acc.residence);
    strcpy(main_farmer.userID,main_acc.userID);
    gets(main_farmer.crop);
    FILE *fp;
    fp = fopen("farmer_details.dat","ab"); //write account details farmer file
    fwrite(&main_farmer,sizeof(main_farmer),1,fp);
    fclose(fp);
}


//---------------------------- Function to create farmer object for new acc ----------------------------------
void create_buyer_details()
{
    strcpy(main_buyer.name, main_acc.name);
    strcpy(main_buyer.residence,main_acc.residence);
    strcpy(main_buyer.userID,main_acc.userID);
    main_buyer.no_of_quotes=0;
    FILE *fp;
    fp = fopen("buyer_details.dat","ab"); //write into buyers file
    fwrite(&main_buyer,sizeof(main_buyer),1,fp);
    fclose(fp);
}


//---------------------------- Function to check if given userID is unique or not during registeration process ----------------------------------
int check_valid_userID(char check[20]) //Takes char as input while it loops through all existing Users and returns 0 or 1
{
    struct basic_details acc;
    FILE *fp;
    fp = fopen("acc.dat","rb");
    int valid = 1;
    while(fread(&acc,sizeof(acc),1,fp))
    {
        if(!strcmp(check,acc.userID))
        {
            valid = 0; //valid = 0, not unique
            break;
        }
    }
    fclose(fp);
    return valid; //returns valid
}


//---------------------------- Farmer Portal ----------------------------------
void farmer_portal()
{
    int selection;
    char ch = 'b'; // to prevent miss direction to wrong menu
    while(ch!='e')
    {
        system("cls");
        printf("\n\n\n\n");
        printf("\t\t\t\t\t\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2 PROJECT FARMHOUSE \xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\n");
        printf("\t\t\t\t\t\t    Mini Project\n");
        printf("\t\t ------------------------------------------------------------------------------------------\n");
        printf("\t\t\t\t\t      Farmer Portal\n");
        printf("\t\t ------------------------------------------------------------------------------------------\n");
        printf("\n\n");
        printf("\t\t\t\t Welcome %s, Select An option: \n",main_acc.name);
        printf("\t\t\t\t\t >Press P to find best quote based on price \n");
        printf("\t\t\t\t\t >Press N to find best buyer based on residence \n");
        printf("\t\t\t\t\t >Press S to search for Buyer based on UserID \n");
        printf("\t\t\t\t\t >Press A to display all visible quotes \n");
        printf("\t\t\t\t\t >Press E to log out \n");
        ch = getch();
        ch = tolower(ch);
        switch(ch)
        {
            case 112: find_by_price(); //to find by price
            break;

            case 110: find_by_location(); //find by residence
            break;

            case 115: search_buyer(); //search buyer
            break;

            case 97: show_all_buyers(); //show all buyers
            break;

            case 101: main_acc.option = -1; // log out
            break;

            default: printf("\n\n\t\t\t\t\t *Enter Valid Option.* \n");
            printf("\t\t\t\t\tPress any key to continue...");
            getch();
        }
    }
}


//----------------------------- Finding best quote price for a crop -----------------------------------
void find_by_price()
{
    system("cls");
    printf("\n\n\n\n");
    printf("\t\t\t\t\t\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2 PROJECT FARMHOUSE \xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\n");
    printf("\t\t\t\t\t\t    Mini Project\n");
    printf("\t\t ------------------------------------------------------------------------------------------\n");
    printf("\t\t\t\t\t      Find Best Quote by Price\n");
    printf("\t\t ------------------------------------------------------------------------------------------\n");
    printf("\n\n");
    int price = 0;
    FILE *fp;
    fp = fopen("quotes.dat","rb");
    struct buyer_quotes quote;
    printf("\t\t\t\t Enter Crop to be sold: ");

    char input[20];
    gets(input);
    int found = 0;

    while(fread(&quote,sizeof(quote),1,fp)) //Finding highest quote value for the input crop
    {
        if(quote.price>=price && quote.valid && !strcmpi(quote.crop,input))
        {
            found = 1; //Crop with given details found
            price = quote.price;
        }
    }
    printf("\n\n");
    fseek(fp,0,SEEK_SET);//setting pointer to beginning of file to read contents again
    if(found)
    {
        printf("\t\t\t\t Results found for search: \n",found);
        printf(" ---------------------------------------------------------------------------------------------------------------------------------------------------- \n");
        printf("|\t  Name    \t|\tPhone Number\t|     Residence Area \t| Quote Number \t|\tCrop \t|     Quote\t|\n");
        printf(" ---------------------------------------------------------------------------------------------------------------------------------------------------- \n");
        while(fread(&quote,sizeof(quote),1,fp))
        {
            if(quote.price==price && quote.valid && !strcmp(quote.crop,input))
            {
                display_buyer(quote);
                printf("\n");
            }
        }
    }
    else
    {
        printf("\t\t\t\t No results found for given search. \n");
    }

    fclose(fp);
    printf("\t\t\t\t Press any key to continue...");
    getch();
}


//------------------------------------- Displaying Quotes ---------------------------------------------
void display_buyer(struct buyer_quotes quotes) //Displays structure data of quotes
{
    FILE *fp_buyer, *fp_main;
    fp_main = fopen("acc.dat","rb");
    fp_buyer = fopen("buyer_details.dat","rb");
    struct basic_details acc;

    while(fread(&acc,sizeof(acc),1,fp_main))
    {
        if(!strcmp(acc.userID,quotes.userID)) //opens corresponding structure object of quote
        {
            printf("|\t%s    \t|\t %ld \t|\t   %s   \t|\t %d \t|\t %s \t| %0.2f Rs/kg\t|\n",acc.name,acc.phno,acc.residence,quotes.quote_no,quotes.crop,quotes.price);
        }
    }
    fclose(fp_main);
    fclose(fp_buyer);
}


//------------------------------------- Find Closest Buyers ---------------------------------------------
void find_by_location()
{
    system("cls");
    printf("\n\n\n\n");
    printf("\t\t\t\t\t\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2 PROJECT FARMHOUSE \xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\n");
    printf("\t\t\t\t\t\t    Mini Project\n");
    printf("\t\t --------------------------------------------------------------------------------------------------------------------\n");
    printf("\t\t\t\t\t      Find Best Quote by Residence\n");
    printf("\t\t --------------------------------------------------------------------------------------------------------------------\n");
    printf("\n\n");
    FILE *fp;
    fp = fopen("quotes.dat","rb");
    printf("\t\t\t\t Enter Residence region: ");
    char input[20];
    gets(input);

    struct buyer_quotes quote;

    printf("\t\t\t\t Results found for search: \n");
    printf(" ---------------------------------------------------------------------------------------------------------------------------------------------------- \n");
    printf("|\t  Name    \t|\tPhone Number\t|     Residence Area \t| Quote Number \t|\tCrop \t|     Quote\t|\n");
    printf(" ---------------------------------------------------------------------------------------------------------------------- \n");
    int found = 0;
    while(fread(&quote,sizeof(quote),1,fp))
    {
        if(!strcmpi(quote.residence,input) && quote.valid)
        {
            found++;
            display_buyer(quote);
        }
    }

    if(found)
    {
        printf("\t\t\t\t %d Results found\n",found);
    }
    else{
        printf("\t\t\t\t No results found\n");
    }
    fclose(fp);
    printf("\t\t\t\t\t Press any Key to continue...\n");
    getch();
}


//----------------------------- Search By ID ------------------------------------
void search_buyer()
{
    system("cls");
    printf("\n\n\n\n");
    printf("\t\t\t\t\t\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2 PROJECT FARMHOUSE \xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\n");
    printf("\t\t\t\t\t\t    Mini Project\n");
    printf("\t\t --------------------------------------------------------------------------------------\n");
    printf("\t\t\t\t\t      Search for a Buyer by USERID\n");
    printf("\t\t --------------------------------------------------------------------------------------\n");
    printf("\n\n");
    printf("\t\t\t\t\t Enter USERID of Buyer: ");
    char input[20];
    gets(input);
    FILE *fp;
    int found = 0;
    struct basic_details acc;
    fp = fopen("acc.dat","rb");
    FILE *fp2;
    fp2 = fopen("buyer_details.dat","rb");
    struct buyer_details buyer_acc;
    printf("\n\n");
    int is_buyer=0;
    while(fread(&buyer_acc,sizeof(buyer_acc),1,fp2))
    {
        if(!strcmp(buyer_acc.userID,input))
        {
            is_buyer = 1; //to prevent a farmer ID to be shown
            break; //userID found
        }

    }
    while(fread(&acc,sizeof(acc),1,fp))
    {
        if(!strcmp(acc.userID,input) && is_buyer) //userID found, Display
        {
            found = 1;
            printf("\t\t\t\t\t\t\t > USERID FOUND \n");
            printf("\t\t\t\t\t\t\t   \xB2UserID: %s \n",acc.userID);
            printf("\t\t\t\t\t\t\t   \xB2Name: %s \n",acc.name);
            printf("\t\t\t\t\t\t\t   \xB2Phone number: %ld \n",acc.phno);
            printf("\t\t\t\t\t\t\t   \xB2Residence: %s \n",acc.residence);
            printf("\t\t\t\t\t\t\t   \xB2No. of Quotes Issued: %d \n",buyer_acc.no_of_quotes);
            break;
        }
    }

    if(!found)
    {
        printf("\t\t\t\t\t *UserID Not Found* \n");
    }

    printf("\t\t\t\t\t Press any key to continue... \n");
    fclose(fp);
    fclose(fp2);
    getch();
}


//----------------------------- Show All Buyers -----------------------------------
void show_all_buyers()
{
    system("cls");
    printf("\n\n\n\n");
    printf("\t\t\t\t\t\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2 PROJECT FARMHOUSE \xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\n");
    printf("\t\t\t\t\t\t    Mini Project\n");
    printf("\t\t ----------------------------------------------------------------------------------------\n");
    printf("\t\t\t\t\t      All Quotes\n");
    printf("\t\t ----------------------------------------------------------------------------------------\n");
    printf("\n\n");
    printf("\t\t\t\t\t All accessible Quotes: \n");
    printf("\n");
    printf("\t\t\t\t Results found for search: \n");
    printf(" ----------------------------------------------------------------------------------------------------------------------- \n");
    printf("|\t  Name    \t|\tPhone Number\t|     Residence Area \t| Quote Number \t|\tCrop \t|     Quote\t|\n");
    printf(" ----------------------------------------------------------------------------------------------------------------------- \n");
    FILE *fp;
    int found = 0;
    struct buyer_quotes quote;
    fp = fopen("quotes.dat","rb");
    while(fread(&quote,sizeof(quote),1,fp))
    {
        if(quote.valid)
        {
            found++;
            display_buyer(quote);
        }
    }
    if(found)
    {
        printf("\t\t\t\t %d Results found \n",found);
    }
    else
    {
        printf("\t\t\t\t No results found \n");
    }
    fclose(fp);
    printf("\t\t\t\t Press any key to continue...\n");
    getch();
}


//----------------------------- Buyers Portal -----------------------------------
void buyer_portal()
{
    char ch = 'b'; // To prevent miss-direction to any option
    while(ch!='e')
    {
        system("cls");
        printf("\n\n\n\n");
        printf("\t\t\t\t\t\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2 PROJECT FARMHOUSE \xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\n");
        printf("\t\t\t\t\t\t    Mini Project\n");
        printf("\t\t ----------------------------------------------------------------------------------------\n");
        printf("\t\t\t\t\t\t    Buyer Portal\n");
        printf("\t\t -----------------------------------------------------------------------------------------\n");
        printf("\n\n");
        printf("\t\t\t\t Welcome %s, Select An option: \n",main_acc.name);
        printf("\t\t\t\t\t >Press A to Add quote \n");
        printf("\t\t\t\t\t >Press R to Remove Quote \n");
        printf("\t\t\t\t\t >Press H for Quote history of all quote raised to date \n");
        printf("\t\t\t\t\t >Press E to log out \n");
        ch = getch();
        ch = tolower(ch);
        switch(ch)
        {
            case 97: add_quote(); //adding quote
            break;

            case 114: remove_quote(); //removing quote
            break;

            case 104: all_quotes(); //quote history
            break;

            case 101: main_acc.option = -1; //sets main_acc option to -1, to redirect back to main screen and not again get redirected to any portal
            break;

            default: printf("\n\n\t\t\t\t\t *Enter Valid Option.*\n");
            printf("\t\t\t\t\t Press any key to continue...");
            getch();
        }
    }

}



//---------------------------- Adding quote to a buyer ----------------------------------
void add_quote()
{
    main_buyer.no_of_quotes++; //increment no of quotes for a buyer
    struct buyer_quotes quote; //creating sample object variable for storing values from file
    quote.quote_no=main_buyer.no_of_quotes;
    system("cls");
    printf("\n\n\n\n");
    printf("\t\t\t\t\t\t\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2 PROJECT FARMHOUSE \xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\n");
    printf("\t\t\t\t\t\t    Mini Project\n");
    printf("\t\t -----------------------------------------------------------------------------------------\n");
    printf("\t\t\t\t\t\t    Add New Quote\n");
    printf("\t\t -----------------------------------------------------------------------------------------\n");
    printf("\n\n");
    printf("\t\t\t\t\t\t Enter Crop: ");
    fflush(stdin);
    gets(quote.crop);
    printf("\t\t\t\t\t\t Quote (Rs per kg): ");
    scanf("%f",&quote.price);
    quote.valid=1; //Setting Quote to valid allows it to be visible to farmers
    strcpy(quote.userID,main_acc.userID); //for uniquely identifying a quote and linking it to a specific user
    strcpy(quote.residence,main_acc.residence);
    FILE *fp;
    fp = fopen("quotes.dat","ab");
    fwrite(&quote,sizeof(quote),1,fp); //writing quote into "quote.dat" file which stores all quotes
    fclose(fp);

    printf("\n\n\t\t\t\t\t\t\t Quote Added Successfully \n");
    printf("\t\t\t\t\t\tPress any key to continue...");

    struct buyer_details acc;
    fp = fopen("buyer_details.dat","rb");
    FILE *fp2;
    fp2 = fopen("temp.dat","wb");
    while(fread(&acc,sizeof(acc),1,fp)) //changing no_of_quotes value of the particular user
    {
        if(!strcmp(acc.userID,main_buyer.userID))
        {
            fwrite(&main_buyer,sizeof(main_buyer),1,fp2); //add changed values to "temp.dat"
        }
        else
        {
            fwrite(&acc,sizeof(acc),1,fp2); //pass unchanged values to "temp.dat"
        }
    }
    fclose(fp);
    fclose(fp2);

    remove("buyer_details.dat");
    rename("temp.dat","buyer_details.dat");
    getch();
}


//----------------------------- Remove Existing Quote from Visibility -----------------------------------
void remove_quote()
{
    system("cls");
    printf("\n\n\n\n");
    printf("\t\t\t\t\t\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2 PROJECT FARMHOUSE \xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\n");
    printf("\t\t\t\t\t\t    Mini Project\n");
    printf("\t\t ------------------------------------------------------------------------------------------\n");
    printf("\t\t\t\t\t\t    Remove Quote\n");
    printf("\t\t ------------------------------------------------------------------------------------------\n");
    printf("\n\n");
    printf("\t\t\t\t List of all visible quotes: \n");
    printf("\t\t -------------------------------------------------------------------------------------------------\n");
    printf("\t\t |\tQuote ID\t|\t  Crop\t\t|\t Quote \t\t|\t Is Valid? \t| \n");
    printf("\t\t -------------------------------------------------------------------------------------------------\n");
    FILE *fp;
    fp = fopen("quotes.dat","rb");
    struct buyer_quotes quote;
    while(fread(&quote,sizeof(quote),1,fp)) //show all quotes which are visibility
        {
            if(!strcmp(quote.userID, main_acc.userID) && quote.valid )
            {
                printf("\t\t |\t     %d   \t|\t  %s\t\t|\t%.2f Rs/kg\t|\t",quote.quote_no,quote.crop,quote.price);
                if(quote.valid)
                {
                    printf("  Yes \t\t| \n");
                }
                else
                {
                    printf("   No \t\t| \n");
                }
            }
        }
    fclose(fp);
    printf("\t\t\t\t Enter Quote number of quote to be deleted: ");
    int input;
    scanf("%d",&input);
    FILE *fp2;
    fp2=fopen("temp.dat","wb");
    fp = fopen("quotes.dat","rb");
    int found = 0; //if quote visibility is removed, found is changed to 1
    while(fread(&quote,sizeof(quote),1,fp))
    {
        if(!strcmp(main_acc.userID,quote.userID) && quote.quote_no==input && quote.valid)
        {
            found = 1;
            quote.valid = 0;
            fwrite(&quote,sizeof(quote),1,fp2);
        }
        else
        {
            fwrite(&quote,sizeof(quote),1,fp2);
        }

    }
    fclose(fp2);
    fclose(fp);

    remove("quotes.dat");
    rename("temp.dat","quotes.dat");

    printf("\n\n");
    if(found)
    {
        printf("\t\t\t\t\t Quote visibility removed successfully\n");
    }
    else
    {
        printf("\t\t\t\t\t *Quote not found*\n");
    }
    printf("\t\t\t\t\t Press any key to continue...");
    getch();
}


//----------------------------- Showing all Quotes of the buyer (Quote History)  ------------------------------------
void all_quotes()
{
    system("cls");
    printf("\n\n\n\n");
    printf("\t\t\t\t\t\t\xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2 PROJECT FARMHOUSE \xB2\xB2\xB2\xB2\xB2\xB2\xB2\xB2\n");
    printf("\t\t\t\t\t\t\t    Mini Project\n");
    printf("\t\t -------------------------------------------------------------------------------------------\n");
    printf("\t\t\t\t\t\t\t    Quote History\n");
    printf("\t\t -------------------------------------------------------------------------------------------\n");
    printf("\n\n");
    printf("\t\t --------------------------------------------------------------------------------------------------\n");
    printf("\t\t |\tQuote ID\t|\t  Crop\t\t|\t Quote \t\t|\t Is Valid? \t| \n");
    printf("\t\t --------------------------------------------------------------------------------------------------\n");
    FILE *fp;
    fp = fopen("quotes.dat","rb");
    struct buyer_quotes quote;
    while(fread(&quote,sizeof(quote),1,fp))
    {
        if(!strcmp(quote.userID, main_acc.userID))
        {
            printf("\t\t |\t     %d   \t|\t  %s\t\t|\t%.2f Rs/kg\t|\t",quote.quote_no,quote.crop,quote.price);
            if(quote.valid)
            {
                printf("  Yes \t\t| \n");
            }
            else
            {
                printf("   No \t\t| \n");
            }
        }
    }
    fclose(fp);
    printf("\t\t --------------------------------------------------------------------------------------------------\n");
    getch();
}
```

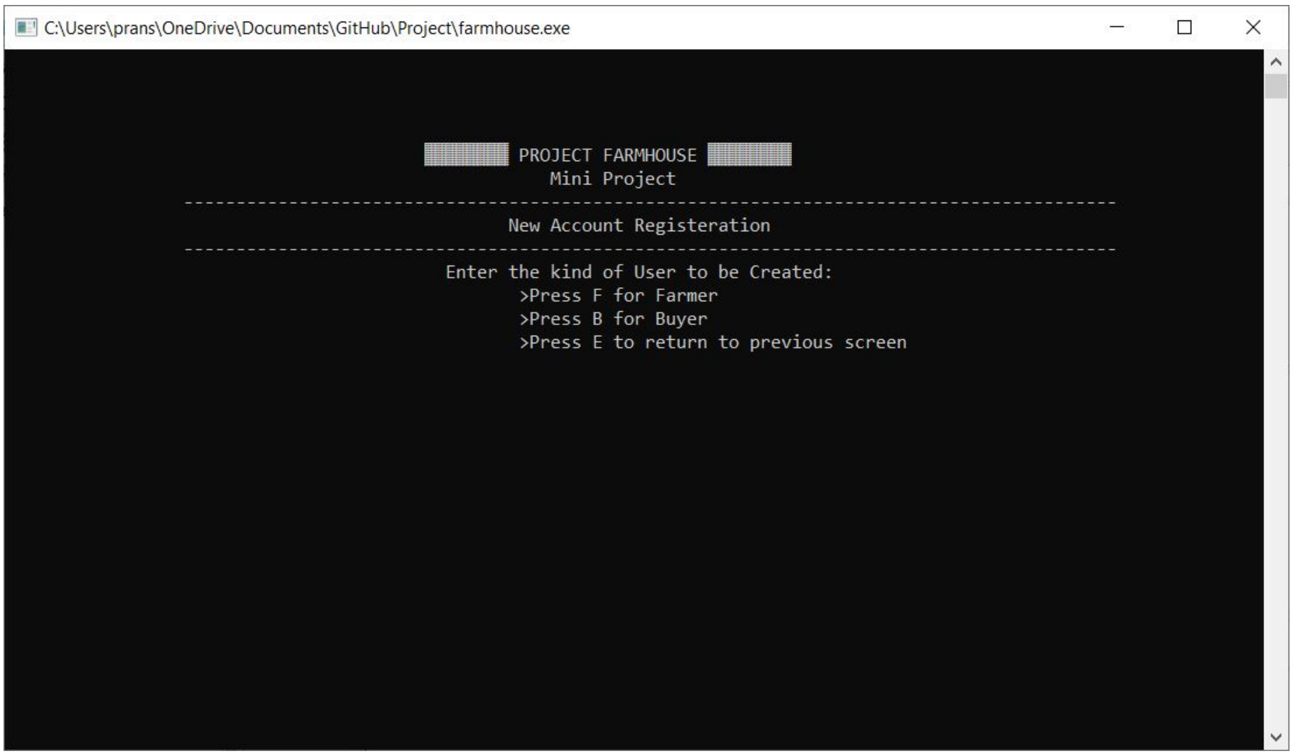
5 Results



C:\Users\prans\OneDrive\Documents\GitHub\Project\farmhouse.exe

```
                         ▓▓▓▓▓▓▓▓ PROJECT FARMHOUSE ▓▓▓▓▓▓▓▓
                                     Mini Project
 ------------------------------------------------------------------------------------------
                                      Created By:
                Dhruv Banerjee (191CH013) and Pranshu Shukla (191ME260)
 ------------------------------------------------------------------------------------------


                Select an option:
                        >Press L to Log In into existing account
                        >Press S to Sign In and create a new account
                        >Press E to Exit
```

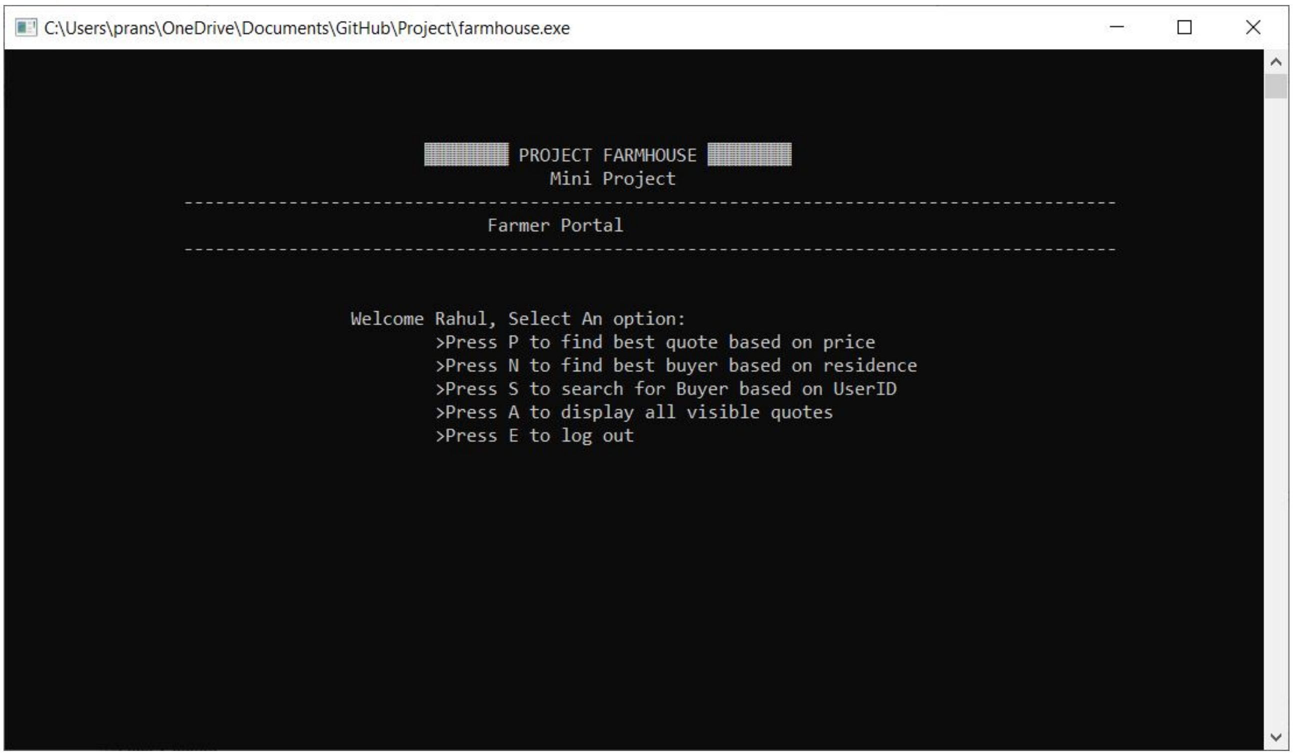
Figure 2: Main Screen



C:\Users\prans\OneDrive\Documents\GitHub\Project\farmhouse.exe

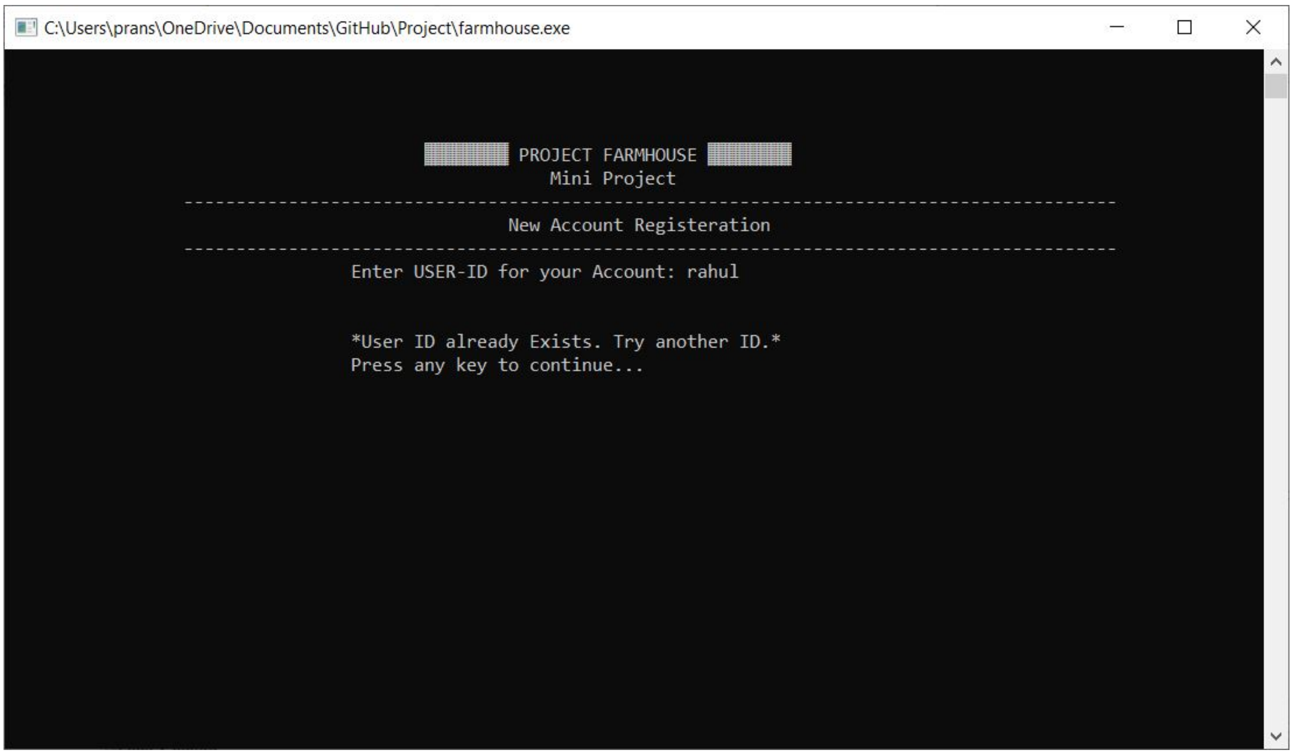
```
                         ▓▓▓▓▓▓▓▓ PROJECT FARMHOUSE ▓▓▓▓▓▓▓▓
                                     Mini Project
 ------------------------------------------------------------------------------------------
                                New Account Registeration
 ------------------------------------------------------------------------------------------
                         Enter the kind of User to be Created:
                                 >Press F for Farmer
                                 >Press B for Buyer
                                 >Press E to return to previous screen
```

Figure 3: Sign Up page - For new account registration



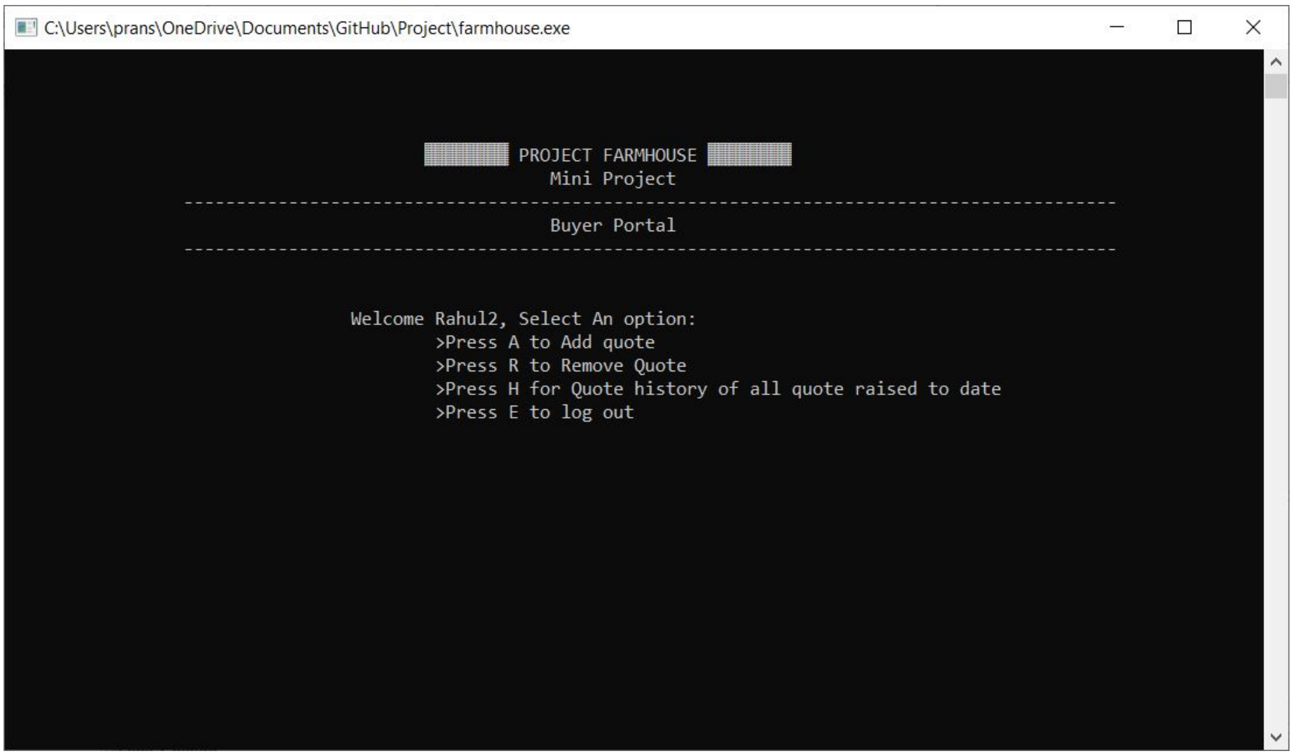
```
C:\Users\prans\OneDrive\Documents\GitHub\Project\farmhouse.exe

                              ▒▒▒▒▒▒▒▒ PROJECT FARMHOUSE ▒▒▒▒▒▒▒▒
                                       Mini Project
                 ------------------------------------------------------------------------------------------
                                    Farmer Portal
                 ------------------------------------------------------------------------------------------


                 Welcome Rahul, Select An option:
                         >Press P to find best quote based on price
                         >Press N to find best buyer based on residence
                         >Press S to search for Buyer based on UserID
                         >Press A to display all visible quotes
                         >Press E to log out
```

Figure 4: Farmer Portal - Facilities and features of Farmer Portal



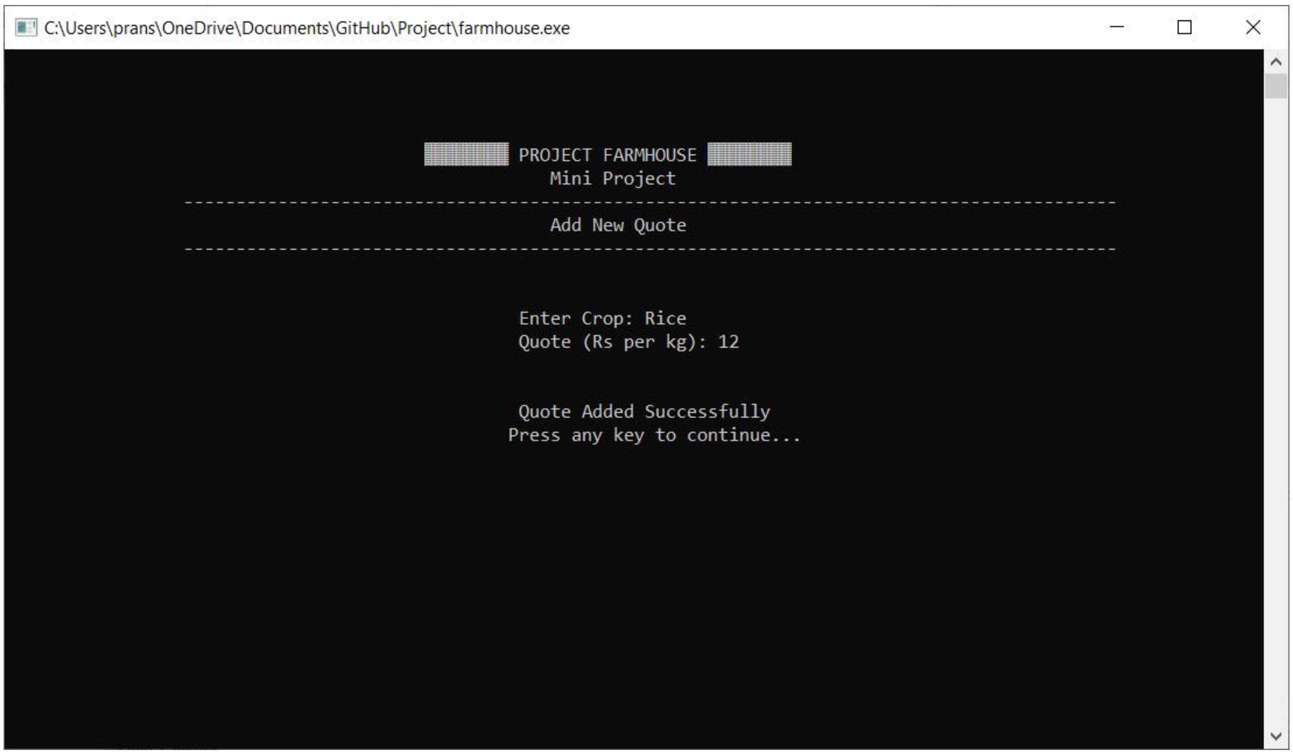
```
C:\Users\prans\OneDrive\Documents\GitHub\Project\farmhouse.exe

                              ▒▒▒▒▒▒▒▒ PROJECT FARMHOUSE ▒▒▒▒▒▒▒▒
                                       Mini Project
                 ------------------------------------------------------------------------------------------
                                    New Account Registeration
                 ------------------------------------------------------------------------------------------
                 Enter USER-ID for your Account: rahul


                 *User ID already Exists. Try another ID.*
                 Press any key to continue...
```

Figure 5: Making New account with existing User ID



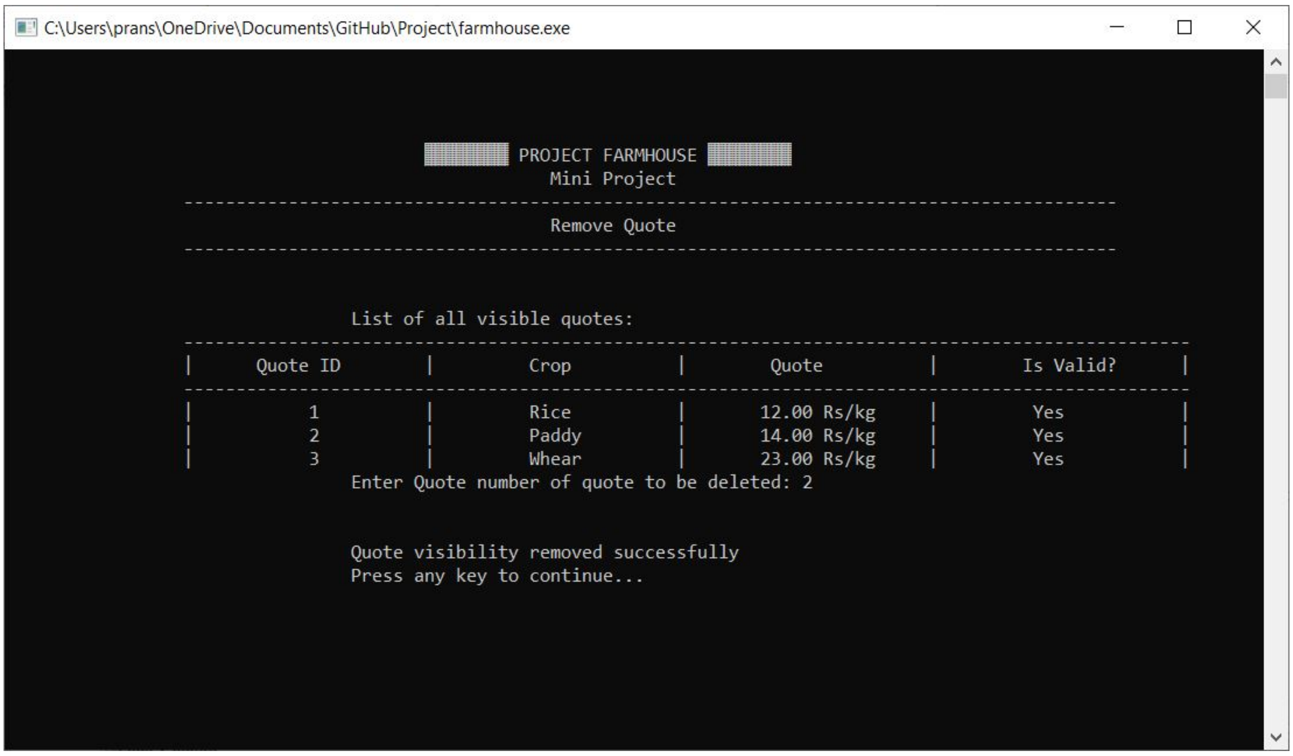
```
C:\Users\prans\OneDrive\Documents\GitHub\Project\farmhouse.exe

                          ░░░░░░░░ PROJECT FARMHOUSE ░░░░░░░░
                                   Mini Project
                 ------------------------------------------------------------------------------------------
                                   Buyer Portal
                 ------------------------------------------------------------------------------------------


                 Welcome Rahul2, Select An option:
                         >Press A to Add quote
                         >Press R to Remove Quote
                         >Press H for Quote history of all quote raised to date
                         >Press E to log out
```

Figure 6: Buyer Portal



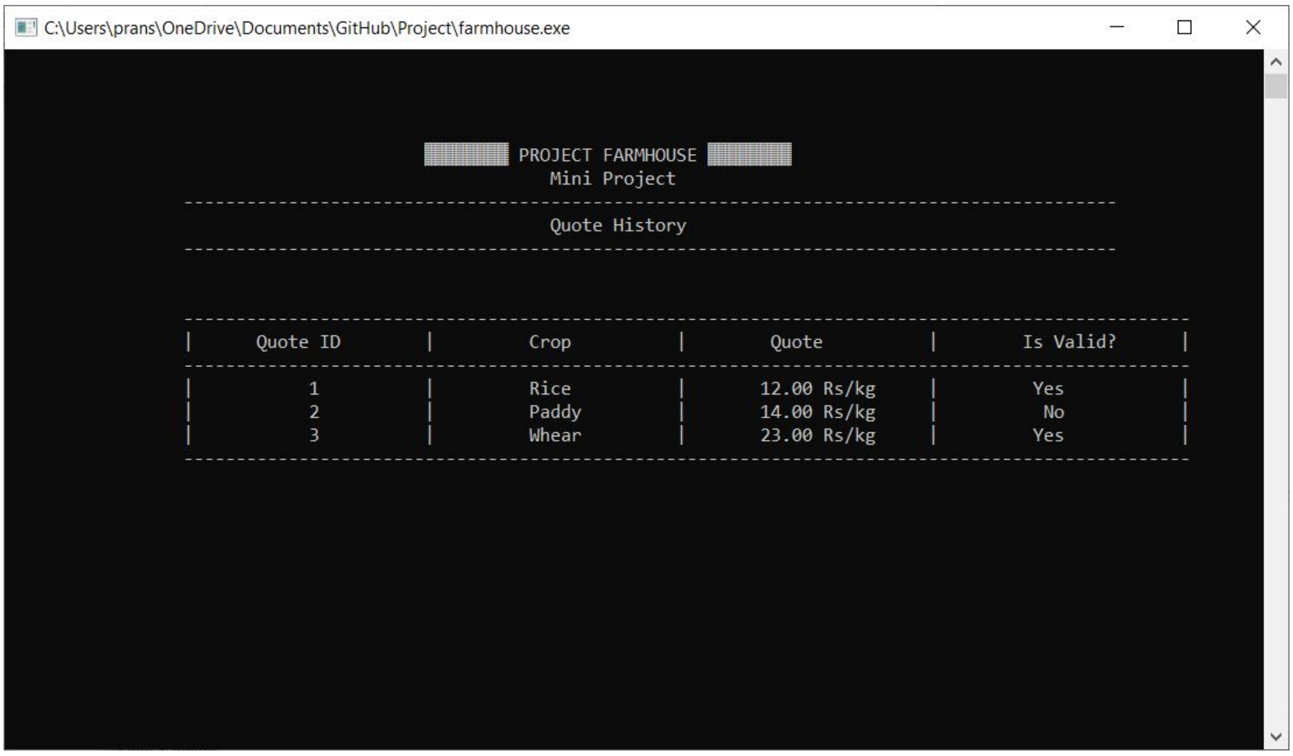
```
C:\Users\prans\OneDrive\Documents\GitHub\Project\farmhouse.exe

                          ░░░░░░░░ PROJECT FARMHOUSE ░░░░░░░░
                                   Mini Project
                 ------------------------------------------------------------------------------------------
                                   Add New Quote
                 ------------------------------------------------------------------------------------------


                                Enter Crop: Rice
                                Quote (Rs per kg): 12


                                Quote Added Successfully
                               Press any key to continue...
```

Figure 7: Adding new Quote



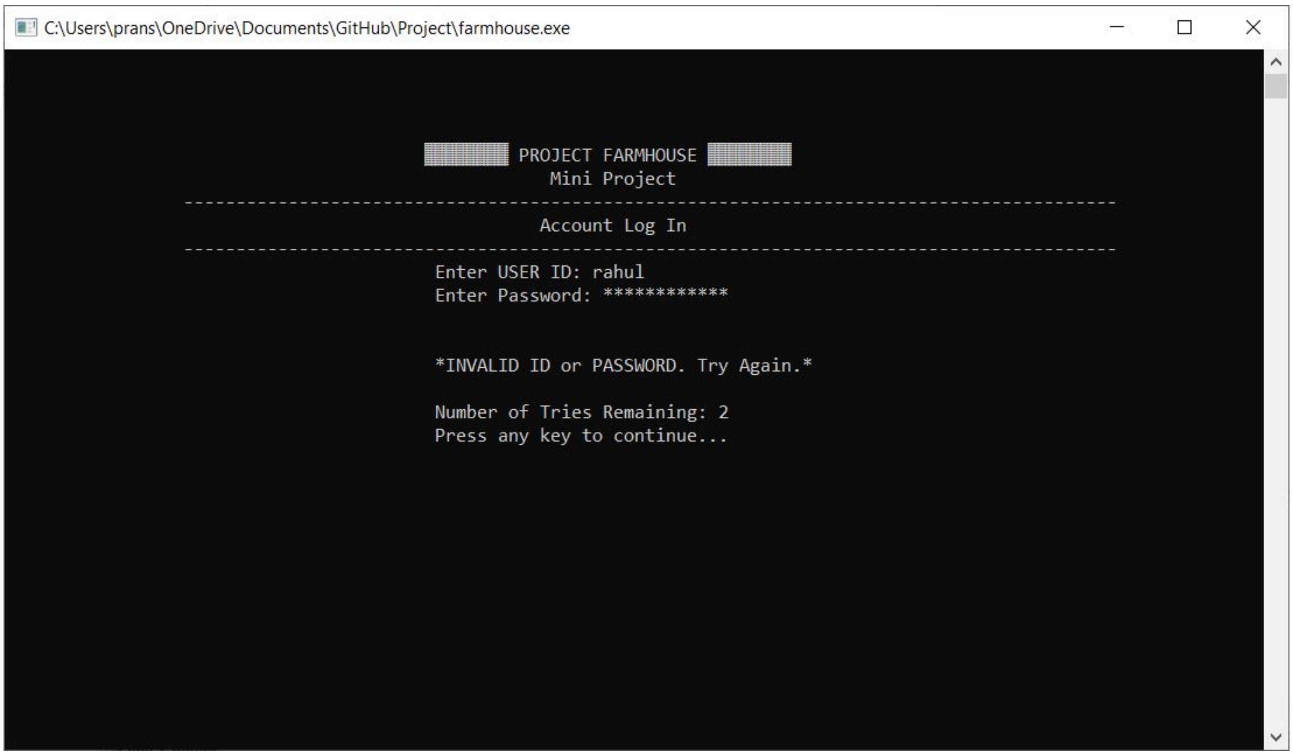
```
C:\Users\prans\OneDrive\Documents\GitHub\Project\farmhouse.exe

                    ▒▒▒▒▒▒▒▒ PROJECT FARMHOUSE ▒▒▒▒▒▒▒▒
                               Mini Project
    --------------------------------------------------------------------------
                               Remove Quote
    --------------------------------------------------------------------------


             List of all visible quotes:
    --------------------------------------------------------------------------------
    |     Quote ID       |       Crop         |       Quote        |      Is Valid?     |
    --------------------------------------------------------------------------------
    |         1          |       Rice         |    12.00 Rs/kg     |        Yes         |
    |         2          |       Paddy        |    14.00 Rs/kg     |        Yes         |
    |         3          |       Whear        |    23.00 Rs/kg     |        Yes         |
             Enter Quote number of quote to be deleted: 2


             Quote visibility removed successfully
             Press any key to continue...
```

Figure 8: Removing Quote



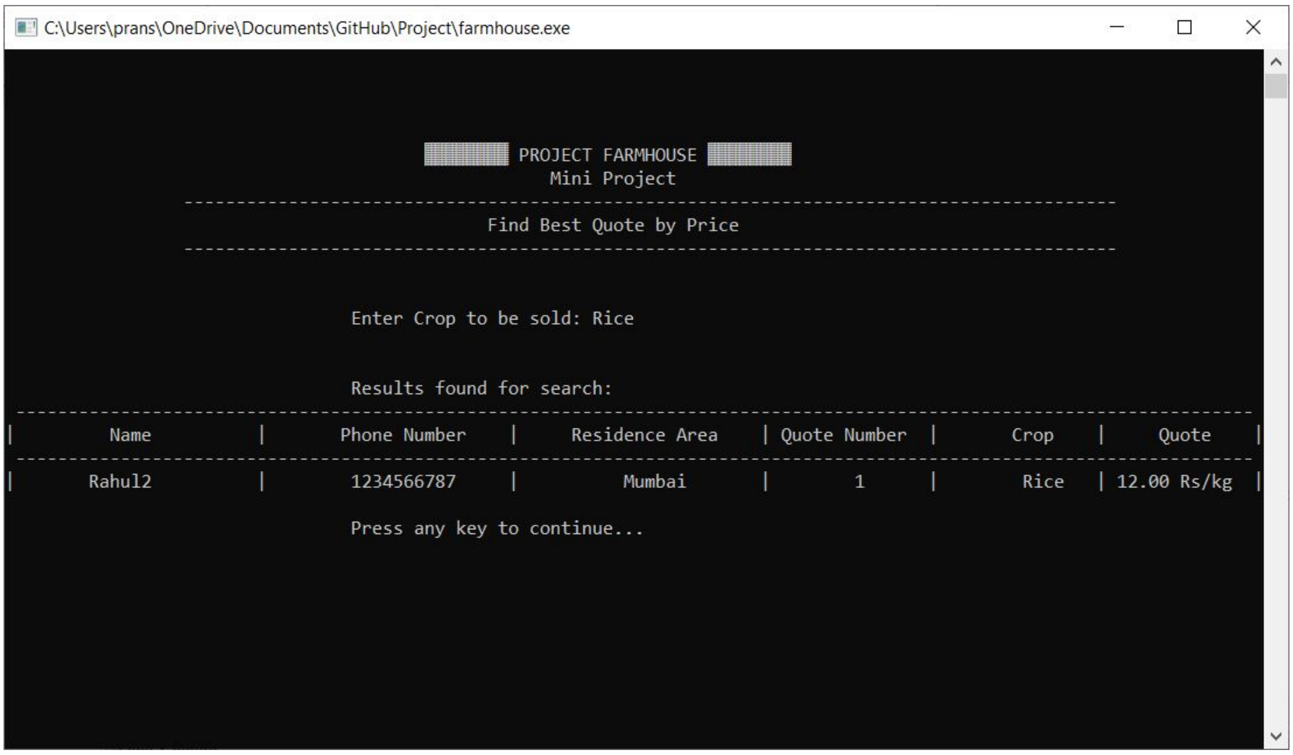
```
C:\Users\prans\OneDrive\Documents\GitHub\Project\farmhouse.exe

                    ▒▒▒▒▒▒▒▒ PROJECT FARMHOUSE ▒▒▒▒▒▒▒▒
                               Mini Project
    --------------------------------------------------------------------------
                               Quote History
    --------------------------------------------------------------------------


    --------------------------------------------------------------------------------
    |     Quote ID       |       Crop         |       Quote        |      Is Valid?     |
    --------------------------------------------------------------------------------
    |         1          |       Rice         |    12.00 Rs/kg     |        Yes         |
    |         2          |       Paddy        |    14.00 Rs/kg     |         No         |
    |         3          |       Whear        |    23.00 Rs/kg     |        Yes         |
    --------------------------------------------------------------------------------
```

Figure 9: Quote History



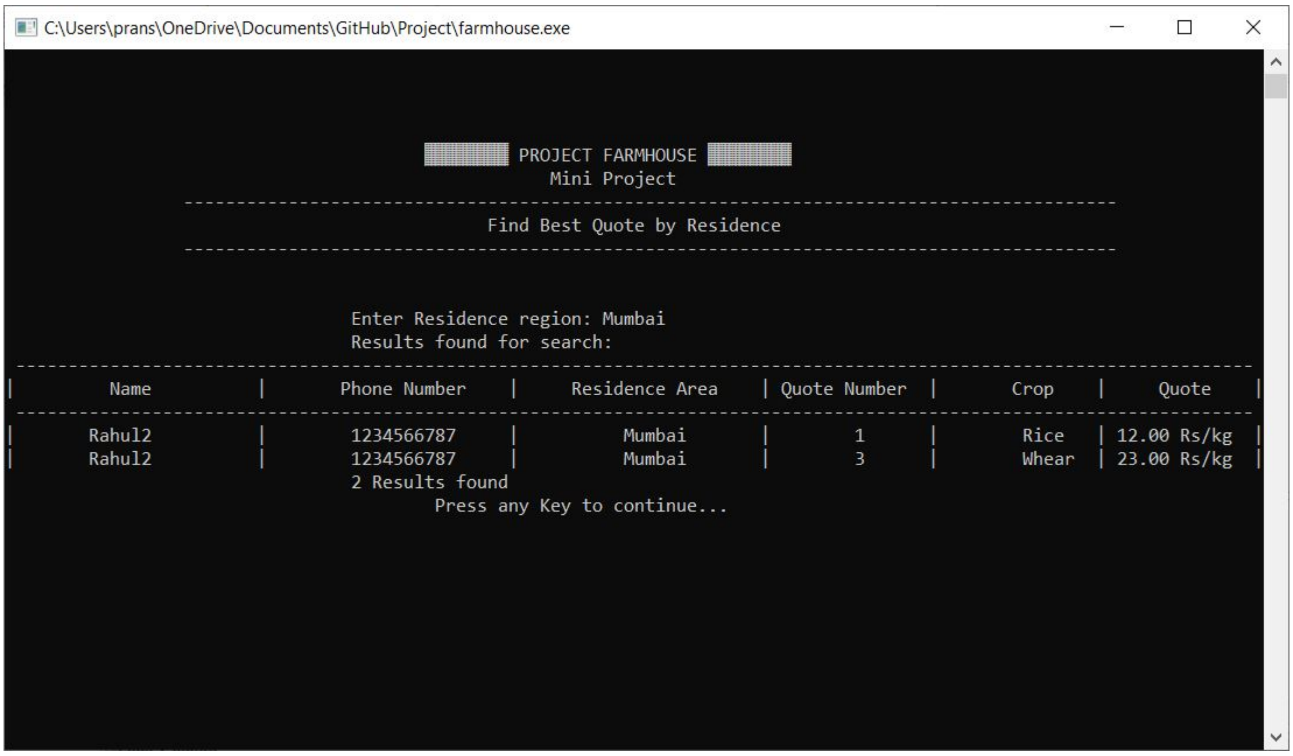
```
C:\Users\prans\OneDrive\Documents\GitHub\Project\farmhouse.exe

                    ░░░░░░░░ PROJECT FARMHOUSE ░░░░░░░░
                              Mini Project
        ---------------------------------------------------------------------------------------
                              Account Log In
        ---------------------------------------------------------------------------------------
                    Enter USER ID: rahul
                    Enter Password: ************


                    *INVALID ID or PASSWORD. Try Again.*

                    Number of Tries Remaining: 2
                    Press any key to continue...
```

Figure 10: Log-In Page - Invalid User-ID



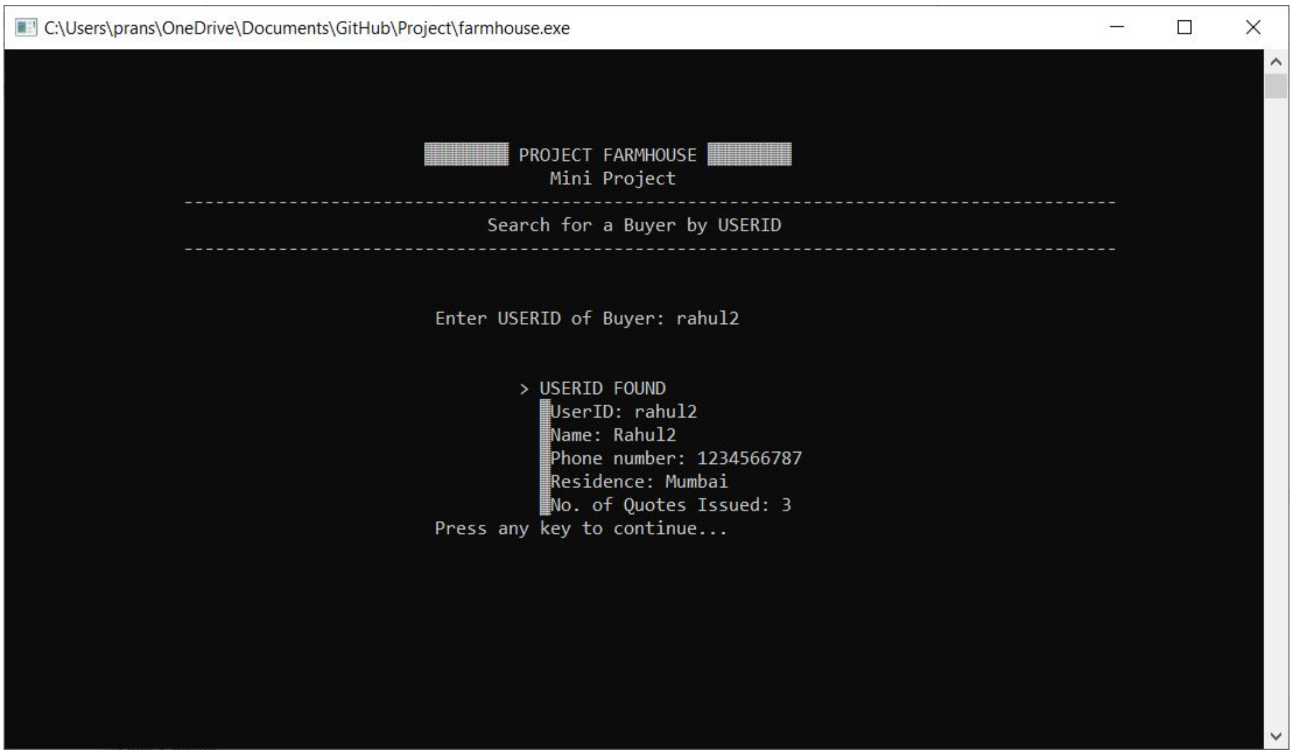
```
C:\Users\prans\OneDrive\Documents\GitHub\Project\farmhouse.exe

                    ░░░░░░░░ PROJECT FARMHOUSE ░░░░░░░░
                              Mini Project
        ---------------------------------------------------------------------------------------
                          Find Best Quote by Price
        ---------------------------------------------------------------------------------------


                Enter Crop to be sold: Rice


                Results found for search:
 --------------------------------------------------------------------------------------------------------------------
|        Name          |    Phone Number     |   Residence Area    | Quote Number |      Crop      |     Quote     |
 --------------------------------------------------------------------------------------------------------------------
|       Rahul2         |     1234566787      |       Mumbai        |      1       |      Rice      | 12.00 Rs/kg   |

                Press any key to continue...
```

Figure 11: Finding quotes in order of Price and Crop



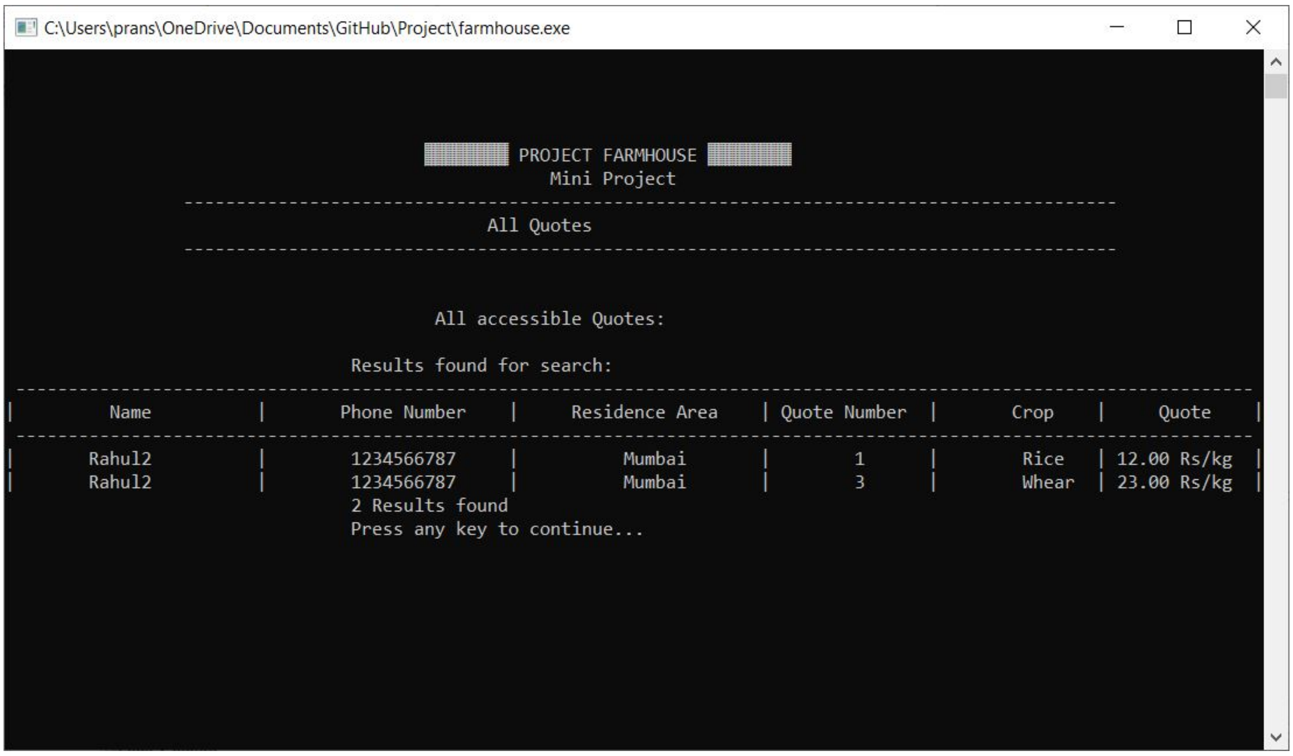
```
C:\Users\prans\OneDrive\Documents\GitHub\Project\farmhouse.exe

                          ▒▒▒▒▒▒▒▒ PROJECT FARMHOUSE ▒▒▒▒▒▒▒▒
                                   Mini Project
        ------------------------------------------------------------------------------------------
                              Find Best Quote by Residence
        ------------------------------------------------------------------------------------------

                          Enter Residence region: Mumbai
                          Results found for search:
---------------------------------------------------------------------------------------------------------------------
|         Name           |     Phone Number       |   Residence Area    | Quote Number  |     Crop     |    Quote     |
---------------------------------------------------------------------------------------------------------------------
|        Rahul2          |      1234566787        |       Mumbai        |       1       |     Rice     | 12.00 Rs/kg  |
|        Rahul2          |      1234566787        |       Mumbai        |       3       |     Whear    | 23.00 Rs/kg  |
                          2 Results found
                                  Press any Key to continue...
```

Figure 12: Finding Quotes based on Residence



```
C:\Users\prans\OneDrive\Documents\GitHub\Project\farmhouse.exe

                          ▒▒▒▒▒▒▒▒ PROJECT FARMHOUSE ▒▒▒▒▒▒▒▒
                                   Mini Project
        ------------------------------------------------------------------------------------------
                              Search for a Buyer by USERID
        ------------------------------------------------------------------------------------------

                                  Enter USERID of Buyer: rahul2


                                          > USERID FOUND
                                            ▒UserID: rahul2
                                            ▒Name: Rahul2
                                            ▒Phone number: 1234566787
                                            ▒Residence: Mumbai
                                            ▒No. of Quotes Issued: 3
                                  Press any key to continue...
```

Figure 13: Search for buyer using USER ID


C:\Users\prans\OneDrive\Documents\GitHub\Project\farmhouse.exe

```
                                ▒▒▒▒▒▒▒▒ PROJECT FARMHOUSE ▒▒▒▒▒▒▒▒
                                          Mini Project
                 -----------------------------------------------------------------------------------------
                                         All Quotes
                 -----------------------------------------------------------------------------------------


                                    All accessible Quotes:

                            Results found for search:
 ------------------------------------------------------------------------------------------------------------------------
|          Name            |      Phone Number      |     Residence Area     | Quote Number  |       Crop      |     Quote     |
 ------------------------------------------------------------------------------------------------------------------------
|        Rahul2            |      1234566787        |          Mumbai        |       1       |       Rice      | 12.00 Rs/kg   |
|        Rahul2            |      1234566787        |          Mumbai        |       3       |       Whear     | 23.00 Rs/kg   |
                            2 Results found
                            Press any key to continue...
```

Figure 14: Displaying All available Quotes

6 References:

1. https://farmityourself.com/how-do-small-farmers-sell-their-crops/

2. https://www.farmerslink.org.uk/where-and-how-farmers-can-sell-their-produce/

3. https://www.thebetterindia.com/101983/farmer-friend-online-vegetables-direct-from-farmers/

****** END ******