using System;

using System.Windows.Forms;

using Microsoft.VisualBasic;

using System.IO;

using System.Text.RegularExpressions;

namespace InvestMe

{

public partial class investMeApp : Form

{

bool isLoginPage = true;

decimal value;

decimal INTEREST\_RATE\_1\_YEAR = 0m, INTEREST\_RATE\_3\_YEARS = 0m, INTEREST\_RATE\_5\_YEARS = 0m, INTEREST\_RATE\_10\_YEARS = 0m;

const string term\_1 = "1 Year", term\_3 = "3 Years", term\_5 = "5 Years", term\_10 = "10 Years";

decimal totalOfOneYear = 0m, totalOfThreeYears = 0m, totalOfFiveYears = 0m, totalOfTenYears = 0m;

string tenure, customerName="", customerTelephone="", customerEmail="", transactionNumber = "";

string enteredText,line;

const string EMPLOYEE\_PASSWORD = "Employee", CUSTOMER\_PASSWORD = "Customer";

string path = @"D:\\MSc Business Analytics\\Sem 1\\MS806 Business Applications Programming\\Assignment\\Assignment 4\\WindowsFormsApp1\\bin\\Debug\\InvestmentDetails.txt";

decimal interestRate = 0m;

decimal accuredAmount = 0m;

string enteredNumber;

private void customerLoginButton\_Click(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(passwordTextBox.Text)) //to check whether password is empty or not

{

MessageBox.Show("Password cannot be empty", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error); //to display error message if password is empty

passwordTextBox.Focus();

}

else if (passwordTextBox.Text.Equals(CUSTOMER\_PASSWORD)) // if password equals to cutomer password showing related controls and hiding irrelevant ones

{

dashboardPage();

logoutButton.Visible = true;

isLoginPage = false;

passwordTextBox.Visible = false;

customerLoginButton.Visible = false;

employeeLoginButton.Visible = false;

summaryButton.Visible = false;

investmentSummaryGroupBox.Visible = false;

passwordLabel.Visible = false;

}

else

{

MessageBox.Show("Wrong password, Please enter correct customer password.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error); // to display error message if password entered is wrong

}

}

private void clearButton\_Click(object sender, EventArgs e)

{

if (isLoginPage) // if current page is login page clear button click clears text in password text box and set focus on it

{

passwordTextBox.Clear();

passwordTextBox.Focus();

}

else

{

resetAllInputs(); // this method clears inputs and selections by user

}

}

private void employeeLoginButton\_Click(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(passwordTextBox.Text)) ///to check whether password is empty or not

{

MessageBox.Show("Password cannot be empty", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error); //to display error if password is empty

passwordTextBox.Focus();

}

else if (passwordTextBox.Text.Equals(EMPLOYEE\_PASSWORD)) // if password equals to employee password showing related controls and hiding irrelevant ones

{

summaryButton.Visible = true;

searchButton.Visible = true;

searchTextBox.Visible = true;

logoutButton.Visible = true;

isLoginPage = false;

passwordTextBox.Visible = false;

customerLoginButton.Visible = false;

employeeLoginButton.Visible = false;

passwordLabel.Visible = false;

enterKeywordLabel.Visible = true;

}

else

{

MessageBox.Show("Wrong password, Please enter correct employee password.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error); //to diplay error if password entered is wrong

}

}

public static Regex re = new Regex(@"^([a-zA-Z0-9\_\-\.]+)@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.)|(([a-zA-Z0-9\-]+\.)+))([a-zA-Z]{2,4}|[0-9]{1,3})(\]?)$",

RegexOptions.CultureInvariant | RegexOptions.Singleline); // regular expression to check whther entered string is a valid email format or not

public static bool IsValidEmailFormat(string enteredText)

{

return re.IsMatch(enteredText); // to check entered string matches valid email format

}

private void logoutButton\_Click(object sender, EventArgs e) //to show necessary conntrols on the click of logout button and hide irrelevant ones

{

investMeApp\_Load(this, e);

passwordLabel.Visible = true;

passwordTextBox.Visible = true;

passwordTextBox.Clear();

employeeLoginButton.Visible = true;

customerLoginButton.Visible = true;

resetAllInputs();

}

private void searchButton\_Click(object sender, EventArgs e)

{

enteredText = searchTextBox.Text; //stored input provided in search textbox in a string

System.IO.StreamReader file = new System.IO.StreamReader(path); //streamreader object to read data from file

//below three boolean variables are declared to identify transaction start transaction end of one specific transaction, and to check whether input provided in search textbox matches with file content

bool transactionStart = false;

bool transactionEnd = false;

bool textMatched = false;

//below are string variables to store the file content

String currentTransaction = "";

String allMatchedTransactions = "";

if (!string.IsNullOrWhiteSpace(searchTextBox.Text)) //to check if search textbox is empty

{

if (Regex.IsMatch(searchTextBox.Text.Trim(), @"\d")) // to check if input provided by user in search textbox is an integer value

{

int it = int.Parse(searchTextBox.Text); //parsing input value as a integer

enteredNumber = it.ToString(); //storing input provided as a string

}

if (IsValidEmailFormat(enteredText) || enteredNumber.Length == 6) //to check if user has entered input in valid email format or valid 6 digit transaction number

{

while ((line = file.ReadLine()) != null) //to check whether or not the end of file has been reached

{

if (line.Equals("TRANSACTION\_START")) //if line equals TRANSACTION\_START set value of transactionStart boolean variable as true

{

transactionStart = true;

transactionEnd = false;

textMatched = false;

continue;

}

if (line.Equals("TRANSACTION\_END")) //if line equals TRANSACTION\_END set value of transactionEnd boolean variable as true and value of transactionStart boolean variable as false

{

transactionStart = false;

transactionEnd = true;

}

if (transactionStart) //if transactionStart value is true

{

currentTransaction = currentTransaction + line + "\n"; //add current line to a string variable

if (line.Contains("Transaction Number") && line.Contains(enteredText)) // if line contains Transaction Number and is matches with input provided by user in search textbox then seat value of textMatched boolean variable as true

{

textMatched = true;

}

else if (line.Contains("Customer Email") && line.Contains(enteredText)) // if line contains Customer Email and is matches with input provided by user in search textbox then seat value of textMatched boolean variable as true

{

textMatched = true;

}

}

if (transactionEnd) //if transactionEnd value is true

{

if (textMatched) // if textMatched value is true

{

allMatchedTransactions = allMatchedTransactions+"\n" + currentTransaction; // add currentTransaction string variable to allMatchedTransactions string variable

}

currentTransaction = ""; //empty the currentTransaction string variable content

textMatched = false; // set textMatched value as false

}

}

if (!string.IsNullOrWhiteSpace(allMatchedTransactions)) // if allMatchedTransactions is not null display the information to the user

{

MessageBox.Show(allMatchedTransactions, "Details for "+enteredText, MessageBoxButtons.OK, MessageBoxIcon.Information);

searchTextBox.Clear();

searchTextBox.Focus();

}

else //display the error message to user that no record found for input provided

{

MessageBox.Show("No record found for " + enteredText, "Search Unsuccessfull", MessageBoxButtons.OK, MessageBoxIcon.Error); //display the error message to user that no record found for input provided

searchTextBox.Clear();

searchTextBox.Focus();

}

}

else

{

MessageBox.Show("Please Enter Valid email address or Transaction number", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error); //display the input error message to user

}

}

else

{

MessageBox.Show("Search field cannot be empty, please enter keyword to search ", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error); //display error message saying field cannot be empty

searchTextBox.Focus();

}

}

private void summaryButton\_Click(object sender, EventArgs e)

{

investmentSummaryGroupBox.Visible = true; //to make investement summary groupbox visible on the click of summary button

//decalring variables for calculation purpose

string allTransactionNumbers = "", totalTransactionNumbers, totalInvestmentAmount, totalAmountWithInterest, totalYears;

decimal totalInvestmentAmountInDecimal, totalAmountWithInterestInDecimal, totalYearsInDecimal;

decimal totalAmountInvested = 0m;

decimal totalAccuredAmount = 0m;

decimal totalInterest = 0m;

decimal averageTermDuration = 0m;

decimal totalNumberOfYears= 0m;

decimal counter=0m;

System.IO.StreamReader file = new System.IO.StreamReader(path); //streamreader object to read data from file

while ((line = file.ReadLine()) != null) //to check whether or not the end of file has been reached

{

if (line.Contains("Transaction Number")) //to check if line contains Transaction Number

{

totalTransactionNumbers = Regex.Match(line, @"\d+").Value; //regular expression to store only numeric value from line

allTransactionNumbers = allTransactionNumbers + totalTransactionNumbers+"\n"; //adding transaction numbers to a string variable

}

if (line.Contains("Tenure of the deposit")) //to check if line contains Tenure of the deposit

{

totalYears = Regex.Match(line, @"\d+").Value; //regular expression to store only numeric value from line

totalYearsInDecimal = decimal.Parse(totalYears); // to parse string into decimal value

totalNumberOfYears = totalNumberOfYears + totalYearsInDecimal; // adding tenures to a decimal variable

counter++; //increamenting counter for occurrences of tenure

}

if(line.Contains("Amount Invested")) // to check if line contains Amount Invested

{

totalInvestmentAmount = Regex.Match(line, @"\d+").Value; //regular expression to store only numeric value from line

totalInvestmentAmountInDecimal = decimal.Parse(totalInvestmentAmount); //to parse string into decimal value

totalAmountInvested = totalAmountInvested + totalInvestmentAmountInDecimal; //adding amount invested to a decimal variable

}

if (line.Contains("Total Accured Amount")) //to check if line contains Total Accured Amount

{

totalAmountWithInterest = Regex.Match(line, @"\d+.+\d?").Value; //regular expression to store only numeric(deciaml) value from line

totalAmountWithInterestInDecimal = decimal.Parse(totalAmountWithInterest); //to parse string into decimal value

totalAccuredAmount = totalAccuredAmount + totalAmountWithInterestInDecimal; //adding total acccured amount to a decimal variable

}

}

totalInterest = totalAccuredAmount - totalAmountInvested; // to caluclate total interest accuring on the investements

averageTermDuration = totalNumberOfYears / counter; // to caluclate the average duration (term) of an investment

//displaying all transactions summary below

transactionNumbersSummaryTextBox.AppendText(allTransactionNumbers);

totalAmountInvestedSummaryTextBox.Text = totalAmountInvested.ToString("C");

totalInterestSummaryTextBox.Text = totalInterest.ToString("C");

averageDurationSummaryTextBox.Text = averageTermDuration+" years";

}

private void proceedButton\_Click(object sender, EventArgs e)

{

if (termInterestRateTable.SelectedIndex == -1) //to check if user has selected any term to invest

{

MessageBox.Show("Please choose a term you wish to invest for", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error); //to display error message if any term is not selected

termInterestRateTable.Focus();

}

else

{

//displaying relevant controls and hiding irrelevant ones

customerDetailsGroupBox.Visible = true;

amountTextBox.Visible = false;

amountToBeInvestedLabel.Visible = false;

displayButton.Visible = false;

termInterestRateTable.Visible = false;

termLabel.Visible = false;

interestRateLabel.Visible = false;

accuredAmountLabel.Visible = false;

proceedButton.Visible = false;

searchTextBox.Visible = false;

searchButton.Visible = false;

//below is to create a unique 6 digit transaction number for every transaction

Random r = new Random();

int randNum = r.Next(1000000);

transactionNumber = randNum.ToString("D6");

switch (termInterestRateTable.SelectedIndex) // switch case to identify which term user has selected and storing related details into respective variables

{

case 0:

tenure = term\_1;

interestRate = INTEREST\_RATE\_1\_YEAR;

accuredAmount = totalOfOneYear;

break;

case 1:

tenure = term\_3;

interestRate = INTEREST\_RATE\_3\_YEARS;

accuredAmount = totalOfThreeYears;

break;

case 2:

tenure = term\_5;

interestRate = INTEREST\_RATE\_5\_YEARS;

accuredAmount = totalOfFiveYears;

break;

case 3:

tenure = term\_10;

interestRate = INTEREST\_RATE\_10\_YEARS;

accuredAmount = totalOfTenYears;

break;

}

}

}

private void confirmButton\_Click(object sender, EventArgs e)

{

//below are the string varibles to store the customer details

customerName = customerNameTextBox.Text.ToString();

customerTelephone = customerTelephoneTextBox.Text.ToString();

customerEmail = customerEmailTextBox.Text.ToString();

if (!string.IsNullOrWhiteSpace(customerName) && !string.IsNullOrWhiteSpace(customerTelephone) && !string.IsNullOrWhiteSpace(customerEmail) && IsValidEmailFormat(customerEmail)) //to check if any of the customer field is empty and whether customer email is in valid format

{

DialogResult dialogResult = MessageBox.Show(" Transaction Number " + transactionNumber +

"\n Customer Name : " + customerName +

"\n Customer Telephone : " + customerTelephone +

"\n Customer Email : " + customerEmail +

"\n Amount Invested :" + value.ToString("C") +

"\n Tenure of Deposit : " + tenure +

"\n Interest Rate : " + interestRate +

"\n Total Accured Amount : " + accuredAmount.ToString("C") + "\n Would you like to confirm the booking?", "Confirm Booking", MessageBoxButtons.YesNo, MessageBoxIcon.Information); //displaying all information related to the investement and asking for confirmation

if (dialogResult == DialogResult.Yes) //to check if user has selected Yes option

{

MessageBox.Show("Dear " + customerName + ", your investement in 'InvestMe' medium-term investement product is successfull. ", "Success", MessageBoxButtons.OK, MessageBoxIcon.Information); //to display investment successfull message

StreamWriter outputFile; //streamwriter object to write into file

outputFile = File.AppendText(path); //this appends text to the file specified and creates a file if its not there in specified path

//below we are writing all transaction related details into the file

outputFile.WriteLine("TRANSACTION\_START");

outputFile.WriteLine("Transaction Number :" + transactionNumber);

outputFile.WriteLine("Customer Name :" + customerName);

outputFile.WriteLine("Customer Telephone :" + customerTelephone);

outputFile.WriteLine("Customer Email :" + customerEmail);

outputFile.WriteLine("Amount Invested :" + value);

outputFile.WriteLine("Tenure of the deposit :" + tenure);

outputFile.WriteLine("Interest Rate :" + interestRate);

outputFile.WriteLine("Total Accured Amount :" + accuredAmount);

outputFile.WriteLine("TRANSACTION\_END");

outputFile.Close();

//below we are showing related controls and hiding irrelevant ones

customerDetailsGroupBox.Visible = false;

customerNameTextBox.Clear();

customerTelephoneTextBox.Clear();

customerEmailTextBox.Clear();

dashboardPage();

amountTextBox.Clear();

}

else if (dialogResult == DialogResult.No) //if user selects No option take user back to customer details groupbox page

{

customerDetailsGroupBox.Visible = true;

customerEmailTextBox.Clear();

customerTelephoneTextBox.Clear();

customerNameTextBox.Clear();

}

}

else

{

if (!IsValidEmailFormat(customerEmail)) //to check if customer has entered email in valid format

{

MessageBox.Show("Please enter a valid email id", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error); //to display error message if email is not in valid format

}

else

{

MessageBox.Show("Please fill all the details before confirming the booking", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error); //to display error message if any of the customer details field is empty

}

}

}

private void exitButton\_Click(object sender, EventArgs e)

{

this.Close(); // closes the application on Exit button click

}

private void amountTextBox\_KeyPress(object sender, KeyPressEventArgs e)

{

isNumeric(e); //to allow only numeric values in Amount textbox

}

private void customerNameTextBox\_KeyPress(object sender, KeyPressEventArgs e)

{

isCharacter(e); //to allow only characters in Customer Name textbox

}

private void customerTelephoneTextBox\_KeyPress(object sender, KeyPressEventArgs e)

{

isNumeric(e); //to allow only numeric values in Customer Telephone textbox

}

private void investMeApp\_Load(object sender, EventArgs e)

{

//to display necessary controls on forma load and hiding irrelevant ones

termInterestRateTable.Visible = false;

proceedButton.Visible = false;

customerDetailsGroupBox.Visible = false;

termLabel.Visible = false;

interestRateLabel.Visible = false;

accuredAmountLabel.Visible = false;

searchTextBox.Visible = false;

searchButton.Visible = false;

amountTextBox.Visible = false;

amountToBeInvestedLabel.Visible = false;

displayButton.Visible = false;

summaryButton.Visible = false;

investmentSummaryGroupBox.Visible = false;

enterKeywordLabel.Visible = false;

logoutButton.Visible = false;

passwordTextBox.Focus();

}

public investMeApp()

{

InitializeComponent();

}

private void displayButton\_Click(object sender, EventArgs e)

{

termInterestRateTable.ClearSelected();

termInterestRateTable.Items.Clear();

if (!string.IsNullOrWhiteSpace(amountTextBox.Text)) //to check whether amount to be invested has entered or not

{

value = decimal.Parse(amountTextBox.Text); //pasre string into decimal value

if (value <= 250000) //to check if value is less than 250000

{

DialogResult dialogResult = MessageBox.Show("InvestMe offers an additional €25,000 bonus for investements of greater than 1 million for 3 years or more on the term completion.\nWould you like to change your investment amount?", "InvestMe Special Offer", MessageBoxButtons.YesNo, MessageBoxIcon.Information); //to diplay investme offer to user and asking if they want to change the investment amount accordingly

if (dialogResult == DialogResult.Yes) //if user selects yes allow user to enter new investment amount

{

amountTextBox.Clear();

amountTextBox.Focus();

}

else if (dialogResult == DialogResult.No) //if user selects No display term interest table to user

{

INTEREST\_RATE\_1\_YEAR = 0.50000m;

INTEREST\_RATE\_3\_YEARS = 0.62500m;

INTEREST\_RATE\_5\_YEARS = 0.71250m;

INTEREST\_RATE\_10\_YEARS = 1.01250m;

//calculating accured amount according to investement amount entered by user and displaying it in a listbox

totalOfOneYear = Math.Round((value \* (decimal)Math.Pow(1 + (double)INTEREST\_RATE\_1\_YEAR / (12 \* 100), 12)), 2);

totalOfThreeYears = Math.Round((value \* (decimal)Math.Pow(1.00 + (double)INTEREST\_RATE\_3\_YEARS / (36 \* 100), 108)), 2);

totalOfFiveYears = Math.Round((value \* (decimal)Math.Pow(1.00 + (double)INTEREST\_RATE\_5\_YEARS / (60 \* 100), 300)), 2);

totalOfTenYears = Math.Round((value \* (decimal)Math.Pow(1.00 + (double)INTEREST\_RATE\_10\_YEARS / (120 \* 100), 1200)), 2);

termInterestRateTable.Items.Add(term\_1 + "\t\t" + INTEREST\_RATE\_1\_YEAR + "\t\t" + totalOfOneYear.ToString("C"));

termInterestRateTable.Items.Add(term\_3 + "\t\t" + INTEREST\_RATE\_3\_YEARS + "\t\t" + totalOfThreeYears.ToString("C"));

termInterestRateTable.Items.Add(term\_5 + "\t\t" + INTEREST\_RATE\_5\_YEARS + "\t\t" + totalOfFiveYears.ToString("C"));

termInterestRateTable.Items.Add(term\_10 + "\t\t" + INTEREST\_RATE\_10\_YEARS + "\t\t" + totalOfTenYears.ToString("C"));

proceedButton.Visible = true;

termLabel.Visible = true;

interestRateLabel.Visible = true;

accuredAmountLabel.Visible = true;

termInterestRateTable.Visible = true;

//amountTextBox.Clear();

}

}

else if (value > 250000 && value <= 1000000) //to check if investemnt amount entered by user is in between 250000 and 1000000

{

DialogResult dialogResult = MessageBox.Show("InvestMe offers an additional €25,000 bonus for investements of greater than 1 million for 3 years or more on the term completion.\nWould you like to change your investment amount?", "InvestMe Special Offer", MessageBoxButtons.YesNo, MessageBoxIcon.Information); //to diplay investme offer to user and asking if they want to change the investment amount accordingly

if (dialogResult == DialogResult.Yes) //if user selects yes allow user to enter new investment amount

{

amountTextBox.Clear();

amountTextBox.Focus();

}

else if (dialogResult == DialogResult.No) //if user selects No display term interest table to user

{

INTEREST\_RATE\_1\_YEAR = 0.60000m;

INTEREST\_RATE\_3\_YEARS = 0.72500m;

INTEREST\_RATE\_5\_YEARS = 0.81250m;

INTEREST\_RATE\_10\_YEARS = 1.02500m;

//calculating accured amount according to investement amount entered by user and displaying it in a listbox

totalOfOneYear = Math.Round((value \* (decimal)Math.Pow(1 + (double)INTEREST\_RATE\_1\_YEAR / (12 \* 100), 12)), 2);

totalOfThreeYears = Math.Round((value \* (decimal)Math.Pow(1 + (double)INTEREST\_RATE\_3\_YEARS / (36 \* 100), 108)), 2);

totalOfFiveYears = Math.Round((value \* (decimal)Math.Pow(1 + (double)INTEREST\_RATE\_5\_YEARS / (60 \* 100), 300)), 2);

totalOfTenYears = Math.Round((value \* (decimal)Math.Pow(1 + (double)INTEREST\_RATE\_10\_YEARS / (120 \* 100), 1200)), 2);

termInterestRateTable.Items.Add(term\_1 + "\t\t" + INTEREST\_RATE\_1\_YEAR + "\t\t" + totalOfOneYear.ToString("C"));

termInterestRateTable.Items.Add(term\_3 + "\t\t" + INTEREST\_RATE\_3\_YEARS + "\t\t" + totalOfThreeYears.ToString("C"));

termInterestRateTable.Items.Add(term\_5 + "\t\t" + INTEREST\_RATE\_5\_YEARS + "\t\t" + totalOfFiveYears.ToString("C"));

termInterestRateTable.Items.Add(term\_10 + "\t\t" + INTEREST\_RATE\_10\_YEARS + "\t\t" + totalOfTenYears.ToString("C"));

proceedButton.Visible = true;

termLabel.Visible = true;

interestRateLabel.Visible = true;

accuredAmountLabel.Visible = true;

termInterestRateTable.Visible = true;

}

}

else if (value > 1000000)

{

MessageBox.Show("Congratulations! You are eligible to recieve an additional €25,000 bonus for investements of greater than 1 million for 3 years or more on the term completion as a part of InvestMe special offer.", "InvestMe Special Offer", MessageBoxButtons.OK, MessageBoxIcon.Information); //display message to user stating they are eligible for InvestMe offer

INTEREST\_RATE\_1\_YEAR = 0.60000m;

INTEREST\_RATE\_3\_YEARS = 0.72500m;

INTEREST\_RATE\_5\_YEARS = 0.81250m;

INTEREST\_RATE\_10\_YEARS = 1.02500m;

// calculating accured amount according to investement amount entered by user and displaying it in a listbox

totalOfOneYear = Math.Round((value \* (decimal)Math.Pow(1 + (double)INTEREST\_RATE\_1\_YEAR / (12\*100), 12)), 2);

totalOfThreeYears = Math.Round((value \* (decimal)Math.Pow(1 + (double)INTEREST\_RATE\_3\_YEARS / (36\*100), 108)), 2) + 25000;

totalOfFiveYears = Math.Round((value \* (decimal)Math.Pow(1 + (double)INTEREST\_RATE\_5\_YEARS / (60\*100), 300)), 2) + 25000;

totalOfTenYears = Math.Round((value \* (decimal)Math.Pow(1 + (double)INTEREST\_RATE\_10\_YEARS / (120\*100), 1200)), 2) + 25000;

termInterestRateTable.Items.Add(term\_1 + "\t\t" + INTEREST\_RATE\_1\_YEAR + "\t\t" + totalOfOneYear.ToString("C"));

termInterestRateTable.Items.Add(term\_3 + "\t\t" + INTEREST\_RATE\_3\_YEARS + "\t\t" + totalOfThreeYears.ToString("C"));

termInterestRateTable.Items.Add(term\_5 + "\t\t" + INTEREST\_RATE\_5\_YEARS + "\t\t" + totalOfFiveYears.ToString("C"));

termInterestRateTable.Items.Add(term\_10 + "\t\t" + INTEREST\_RATE\_10\_YEARS + "\t\t" + totalOfTenYears.ToString("C"));

proceedButton.Visible = true;

termLabel.Visible = true;

interestRateLabel.Visible = true;

accuredAmountLabel.Visible = true;

termInterestRateTable.Visible = true;

}

}

else

{

MessageBox.Show("Amount field cannot be empty ", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error); //display error message that amount field cannot be empty

amountTextBox.Focus();

}

}

private void dashboardPage()

{

//displaying necessary controls and hiding irrelevant ones

amountTextBox.Visible = true;

amountToBeInvestedLabel.Visible = true;

displayButton.Visible = true;

termInterestRateTable.Visible = true;

termLabel.Visible = true;

interestRateLabel.Visible = true;

accuredAmountLabel.Visible = true;

proceedButton.Visible = true;

searchTextBox.Visible = true;

searchButton.Visible = true;

termInterestRateTable.ClearSelected();

termInterestRateTable.Items.Clear();

termInterestRateTable.Visible = false;

termLabel.Visible = false;

interestRateLabel.Visible = false;

accuredAmountLabel.Visible = false;

proceedButton.Visible = false;

}

public void resetAllInputs()

{

//resetting all inputs and selectios provided by user

amountTextBox.Clear();

searchTextBox.Clear();

termInterestRateTable.ClearSelected();

termInterestRateTable.Items.Clear();

termInterestRateTable.Visible = false;

customerEmailTextBox.Clear();

customerTelephoneTextBox.Clear();

customerNameTextBox.Clear();

termLabel.Visible = false;

interestRateLabel.Visible = false;

accuredAmountLabel.Visible = false;

proceedButton.Visible = false;

}

public void isNumeric(KeyPressEventArgs e) //method to allow only numeric values

{

char ch = e.KeyChar;

if (!char.IsDigit(ch) && ch != 8) //this check is to make sure that user enters only numeric values investment amount textbox

{

e.Handled = true;

}

}

public void isCharacter(KeyPressEventArgs e) //method to allow only character values

{

char le = e.KeyChar;

if (!char.IsLetter(le) && le != 8 && le != 32) //this check is to make sure that user enters only character values in the student name textbox

{

e.Handled = true;

}

}

}

}