/\*

## MS806 - Business Application Programming

## Assignment No : 4

## Name : Jayakarthi Boovendran | ## StudentID : 19230487

## Date of Submission : November 13, 2019

## Assignment Description : To Build a Window Form 'InvestMe', to enable employees to make investment bookings for Customers

and to compute Company's overall Investment summary.

## The Form takes in user's info like Investment Amount,Investment Plan and Customer Info as Input,

and the completed Transaction details are stored in a text file. The same is read and reported as Summary. \*/

using System;

using System.Data;

using System.Windows.Forms;

using System.Configuration;

using System.Net.Mail;

using System.IO;

namespace InvestMe

{

public partial class InvestMeForm : Form

{

// Holds the Selected Investment Plan Info like Terms and Interest Rate

DataGridViewRow SelectedInvestmentPlan = new DataGridViewRow();

// Holds Summary Object

DataTable SummaryTable = new DataTable();

public InvestMeForm()

{

InitializeComponent();

// Loads Home Page

InitializeHomePage();

}

#region Load Home Page

// Loads the Home Page - Contains options for 'New Investment','Investment Summary' and 'Exit'

public void InitializeHomePage()

{

HomePageGroupBox.Visible = true;

NewInvestmentButton.Visible = true;

SummaryButton.Visible = true;

ExitButton.Visible = true;

InvestmentGroupBox.Visible = false;

InvestmentConfirmationGroupBox.Visible = false;

SummaryGroupBox.Visible = false;

}

#endregion

#region Home Page Functionalities

#region New Investment

// Loads Investment page

// Allows employers to select investment details such as, Capital amount, rate of interest, terms and ROI

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

{

InitializeInvestmentPage();

}

#endregion

#region Load Investment Page

// Loads Investment Page on click of 'New Investment' from Home Page

public void InitializeInvestmentPage()

{

InvestmentGroupBox.Visible = true;

HomePageGroupBox.Visible = false;

InvestmentConfirmationGroupBox.Visible = false;

SummaryGroupBox.Visible = false;

PrincipleInvestmentAmountTextBox.Text = string.Empty;

InvestmentDetailsGrid.Visible = false;

ProceedButton.Visible = false;

EditSelectionButton.Visible = false;

CustomerInfoGroupBox.Visible = false;

HomePageButton.Visible = true;

ExitButton2.Visible = true;

}

#endregion

#region Exit

// Closes the Application

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

{

Close();

}

#endregion

#endregion

#region Investment Page Functionalities

#region Display Investment Plans

// Displays Investment Plans in a Data Grid

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

{

decimal InvestmentAmount = 0.0m;

try

{

InvestmentAmount = Convert.ToDecimal(PrincipleInvestmentAmountTextBox.Text);

// Investment Amount should atleast be 1

if (InvestmentAmount >= 1)

{

// Loads Meta Investment Data from App.Config into the Data Grid

GetMetaData(InvestmentAmount);

InvestmentDetailsGrid.Visible = true;

InvestmentDetailsGrid.Enabled = true;

InvestmentDetailsGrid.Cursor = Cursors.Default;

ProceedButton.Visible = true;

ProceedButton.Enabled = true;

CustomerNameTextBox.Text = string.Empty;

EmailIDTextBox.Text = string.Empty;

TelephoneNumberTextBox.Text = string.Empty;

}

else

{

MessageBox.Show("Invalid Investment Amount", "Input Validation Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

PrincipleInvestmentAmountTextBox.Text = string.Empty;

PrincipleInvestmentAmountTextBox.Focus();

}

}

catch (Exception)

{

MessageBox.Show("Enter Numeric Data for Investment Amount", "Input Validation Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

PrincipleInvestmentAmountTextBox.Text = string.Empty;

PrincipleInvestmentAmountTextBox.Focus();

}

}

#endregion

#region Interest Calculations

// Computes Monthly Compound Interest

public decimal CalculateCompoundedInterest(int \_terms, decimal \_rate, decimal InvestmentAmount)

{

int months = \_terms \* 12, count = 1;

\_rate = \_rate / 100;

decimal \_balance = InvestmentAmount;

while (count <= months)

{

\_balance = \_balance + (\_balance \* \_rate);

count++;

}

return Math.Round(\_balance, 2);

}

#endregion

#region Load Grid Data

//Populates Investment Details into Data Grid

public void GetMetaData(decimal InvestmentAmount)

{

decimal BonusAmount = 0.0m, \_rate = 0.0m;

int \_terms = 0;

DataTable table = new DataTable();

//Index position of the Column 'Term' in the App.Config 'ColumnHeader' String

int TermIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Term\_Index").ToString());

//Columns added from App.Config into the DataTable

string ColumnHeaders = ConfigurationManager.AppSettings.Get("ColumnHeader").ToString();

for (int i = 0; i < ColumnHeaders.Split('|').Length; i++)

{

if (i == TermIndex)

{

// Term - (int)

table.Columns.Add(ColumnHeaders.Split('|')[i], typeof(int));

}

else

{

// Rate,Invested Amount,Interest,Bonus and ROI (Accured Interest) - (Decimal)

table.Columns.Add(ColumnHeaders.Split('|')[i], typeof(decimal));

}

}

// Total No.of.Terms = 4 (1,3,5,10)

int TotalTerms = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Total\_Terms").ToString());

//Rows are fetched from App.Config and Inserted into the Data Table

if (InvestmentAmount <= 250000 && InvestmentAmount > 0)

{

string Category = "Category\_1\_Record\_";

BonusAmount = 0.0m;

for (int i = 1; i <= TotalTerms; i++)

{

string \_value = ConfigurationManager.AppSettings.Get(Category + i.ToString()).ToString();

\_terms = Convert.ToInt32(\_value.Split(',')[0]);

\_rate = Convert.ToDecimal(\_value.Split(',')[1]);

table.Rows.Add(\_terms, \_rate, InvestmentAmount, CalculateCompoundedInterest(\_terms, \_rate, InvestmentAmount) - InvestmentAmount, BonusAmount, CalculateCompoundedInterest(\_terms, \_rate, InvestmentAmount) + BonusAmount);

}

}

else if (InvestmentAmount > 250000 && InvestmentAmount <= 1000000)

{

string Category = "Category\_2\_Record\_";

BonusAmount = 0.0m;

for (int i = 1; i <= TotalTerms; i++)

{

string \_value = ConfigurationManager.AppSettings.Get(Category + i.ToString()).ToString();

\_terms = Convert.ToInt32(\_value.Split(',')[0]);

\_rate = Convert.ToDecimal(\_value.Split(',')[1]);

table.Rows.Add(\_terms, \_rate, InvestmentAmount, CalculateCompoundedInterest(\_terms, \_rate, InvestmentAmount) - InvestmentAmount, BonusAmount, CalculateCompoundedInterest(\_terms, \_rate, InvestmentAmount) + BonusAmount);

}

}

else if (InvestmentAmount > 1000000)

{

string Category = "Category\_2\_Record\_";

BonusAmount = 0.0m;

for (int i = 1; i <= TotalTerms; i++)

{

string \_value = ConfigurationManager.AppSettings.Get(Category + i.ToString()).ToString();

\_terms = Convert.ToInt32(\_value.Split(',')[0]);

\_rate = Convert.ToDecimal(\_value.Split(',')[1]);

table.Rows.Add(\_terms, \_rate, InvestmentAmount, CalculateCompoundedInterest(\_terms, \_rate, InvestmentAmount) - InvestmentAmount, BonusAmount, CalculateCompoundedInterest(\_terms, \_rate, InvestmentAmount) + BonusAmount);

// Bonus=250000 (for years 3,5,and 10)

BonusAmount = 250000.00m;

}

}

InvestmentDetailsGrid.DataSource = table;

}

#endregion

#region Proceed Invesment with User Details

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

{

string SelectedPlan = string.Empty;

// Investment Plan Validations

if (InvestmentDetailsGrid.SelectedRows.Count == 0)

{

MessageBox.Show("Please Select an Investment Plan. To Select an Investment Plan, Select the Entire Row.", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else if (InvestmentDetailsGrid.SelectedRows.Count > 1)

{

MessageBox.Show("Selection of more than one Investment plan is not allowed", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

InvestmentDetailsGrid.ClearSelection();

}

else if (InvestmentDetailsGrid.SelectedRows.Count == 1)

{

SelectedInvestmentPlan = InvestmentDetailsGrid.SelectedRows[0];

//Transaction ID generation

Random random = new Random();

TransactionIDTextBox.Text = random.Next(100000, 999999).ToString();

CustomerInfoGroupBox.Visible = true;

InvestmentDetailsGrid.Enabled = false;

ProceedButton.Enabled = false;

InvestmentDetailsGrid.Cursor = Cursors.No;

EditSelectionButton.Visible = true;

EditSelectionButton.Enabled = true;

}

}

#endregion

#region Edit Investment Plan

// Allows Users to Change the Investment Plan

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

{

InvestmentDetailsGrid.Enabled = true;

ProceedButton.Enabled = true;

InvestmentDetailsGrid.Cursor = Cursors.Default;

EditSelectionButton.Enabled = false;

}

#endregion

#region Customer Info Validation

public bool CustomerInfoValidation()

{

bool \_result = false;

long \_tempTelephoneNumber;

try

{

// Customer Name - Required Field Validation

if (string.IsNullOrWhiteSpace(CustomerNameTextBox.Text))

{

MessageBox.Show("Pleae Enter Customer Name", "Input Validation Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

CustomerNameTextBox.Focus();

return \_result;

}

// Customer Telephone - Numeric Data Validation

// Assume that only the phone number is entered without Country Code (+353)

if (!string.IsNullOrWhiteSpace(TelephoneNumberTextBox.Text))

{

\_tempTelephoneNumber = long.Parse(TelephoneNumberTextBox.Text);

if(TelephoneNumberTextBox.Text.Length!=9)

{

MessageBox.Show("Invalid Telephone Number! Please Enter your 9 digit Telephone Number excluding the Country code(+353-)","Input Validation Error",MessageBoxButtons.OK,MessageBoxIcon.Error);

TelephoneNumberTextBox.Focus();

return \_result;

}

}

else

{

// Required Field Validation

MessageBox.Show("Pleae Enter Telephone Number", "Input Validation Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

TelephoneNumberTextBox.Focus();

return \_result;

}

//Customer Email - Format Validation

if (!string.IsNullOrWhiteSpace(EmailIDTextBox.Text))

{

\_result = IsEmailValid(EmailIDTextBox.Text);

}

else

{

// Required Field Validation

MessageBox.Show("Pleae Enter Email ID", "Input Validation Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

EmailIDTextBox.Focus();

}

return \_result;

}

catch (Exception)

{

MessageBox.Show("Enter Numeric Data for Phone Number", "Input Validation Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

TelephoneNumberTextBox.Text = string.Empty;

TelephoneNumberTextBox.Focus();

return \_result;

}

}

public bool IsEmailValid(string EmailID)

{

try

{

MailAddress Email = new MailAddress(EmailID);

return true;

}

catch (FormatException)

{

MessageBox.Show("Pleae Enter a Valid Email ID", "Input Validation Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

EmailIDTextBox.Clear();

EmailIDTextBox.Focus();

return false;

}

}

#endregion

#region Clear Customer Info

// Clears Customer Info like Name, Email and Telephone

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

{

CustomerNameTextBox.Text = string.Empty;

TelephoneNumberTextBox.Text = string.Empty;

EmailIDTextBox.Text = string.Empty;

CustomerNameTextBox.Focus();

}

#endregion

#region Redirect to HomePage from Investment Details Page

//Redirects to Home Page

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

{

DialogResult result = MessageBox.Show("Transaction will not be saved! Continue Redirection?", "Confirm Redirection", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

InitializeHomePage();

}

}

#endregion

#region Exit

//Exits from InvestmentDetails Page

private void ExitButton2\_Click(object sender, EventArgs e)

{

Close();

}

#endregion

#endregion

#region Transaction Confirmation Page Functionalities

//Confirm Transaction Page

//User Confirms the Investment and Customer Details

#region Confirm Investment and User Details

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

{

if (CustomerInfoValidation())

{

// Ordinality of Columns in Investment Plan DataTable

int TermIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Term\_Index").ToString());

int InterestRateIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Interest\_Rate\_Index").ToString());

int InvestmentPrincipleIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Investment\_Principle\_Index").ToString());

int InterestAmountIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Interest\_Amount\_Index").ToString());

int ROIIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("ROI\_Index").ToString());

InvestmentConfirmationGroupBox.Visible = true;

HomePageGroupBox.Visible = false;

InvestmentGroupBox.Visible = false;

// Customer Info

NameConfirmationTextBox.Text = CustomerNameTextBox.Text;

EmailIDConfirmationTextBox.Text = EmailIDTextBox.Text;

ContactNumberConfirmationTextBox.Text = TelephoneNumberTextBox.Text;

TransactionIDConfirmationTextBox.Text = TransactionIDTextBox.Text;

// Investment Details

InvestmentAmountConfirmationTextBox.Text = "€ " + SelectedInvestmentPlan.Cells[InvestmentPrincipleIndex].Value.ToString();

InterestRateConfirmationTextBox.Text = SelectedInvestmentPlan.Cells[InterestRateIndex].Value.ToString();

InvestmentTermConfirmationTextBox.Text = SelectedInvestmentPlan.Cells[TermIndex].Value.ToString();

InterestAmountConfirmationTextBox.Text = "€ " + SelectedInvestmentPlan.Cells[InterestAmountIndex].Value.ToString();

ROIConfirmationTextBox.Text = "€ " + SelectedInvestmentPlan.Cells[ROIIndex].Value.ToString();

}

}

#endregion

#region Edit Transaction

// Allows User to Edit the Transaction which includes modifying Customer details and Investment Plans

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

{

InvestmentConfirmationGroupBox.Visible = false;

HomePageGroupBox.Visible = false;

SummaryGroupBox.Visible = false;

InvestmentGroupBox.Visible = true;

ProceedButton.Enabled = false;

EditSelectionButton.Enabled = true;

InvestmentDetailsGrid.Enabled = true;

ConfirmPlanButton.Focus();

}

#endregion

#region Confirm Transaction - Write to File

//On Confirmation – Transaction Details are written in a text file in a Specified format

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

{

string DataString = ConfigurationManager.AppSettings.Get("DataString").ToString();

DataString = DataString. Replace("{Transaction\_ID}", TransactionIDConfirmationTextBox.Text).Replace("{Customer\_Name}", NameConfirmationTextBox.Text).Replace("{Email\_ID}", EmailIDConfirmationTextBox.Text).Replace("{Contact\_Number}", ContactNumberConfirmationTextBox.Text).Replace("{Investment\_Amount}", InvestmentAmountConfirmationTextBox.Text).Replace("{Investment\_Term}", InvestmentTermConfirmationTextBox.Text).Replace("{Interest\_Rate}", InterestRateConfirmationTextBox.Text).Replace("{Bonus\_Amount}", "€ " + SelectedInvestmentPlan.Cells[4].Value.ToString()).Replace("{Created\_On}", DateTime.Now.ToString()).Replace("{Interest\_Amount}", InterestAmountConfirmationTextBox.Text).Replace("{Accured\_Interest}", ROIConfirmationTextBox.Text);

// DataString is saved in a text file

StreamWriter streamWriter;

if (!File.Exists("InvestMe\_DataFile.txt"))

{

var \_file=File.CreateText("InvestMe\_DataFile.txt");

\_file.Dispose();

}

streamWriter = File.AppendText("InvestMe\_DataFile.txt");

streamWriter.WriteLine(DataString);

streamWriter.Close();

MessageBox.Show("Transaction Saved!", "Transaction Successful", MessageBoxButtons.OK, MessageBoxIcon.Information);

// Redirects to home page once Transaction is saved.

InitializeHomePage();

}

#endregion

#endregion

#region Summary Page Functionalities

//Summary Page

//Allows users to View Transaction Summary, Total Ivestment,Total Transactions and Average Term

//Allows users to Search the summary using TransactionID or Email

#region Summary

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

{

// Hides other group boxes

SummaryGroupBox.Visible = true;

HomePageGroupBox.Visible = false;

InvestmentGroupBox.Visible = false;

InvestmentConfirmationGroupBox.Visible = false;

// Order of Data points in File

int TransactionIDIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("TransactionID").ToString());

int NameIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Name").ToString());

int EmailIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Email").ToString());

int ContactNumberIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Contact Number").ToString());

int InvestedAmountIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Invested Amount").ToString());

int InvestedOnIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Invested On").ToString());

int InvestmentTermIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Investment Term").ToString());

int InvestmentRateIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Interest Rate").ToString());

int InvestmentBonusIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Bonus").ToString());

int CompundedInterestIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Interest Amount").ToString());

int AccuredInterestIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Accured Interest").ToString());

SummaryTable.Columns.Clear();

SummaryTable.Rows.Clear();

//Columns Added from App.Config

string ColumnHeaders = ConfigurationManager.AppSettings.Get("SummaryColumnHeader").ToString();

for (int i = 0; i < ColumnHeaders.Split('|').Length; i++)

{

SummaryTable.Columns.Add(ColumnHeaders.Split('|')[i], typeof(string));

}

//File Data is read and Inserted as Rows

foreach (string data in File.ReadAllLines("InvestMe\_DataFile.txt"))

{

SummaryTable.Rows.Add(data.Split(',')[TransactionIDIndex].Split(':')[1], data.Split(',')[NameIndex].Split(':')[1], data.Split(',')[EmailIndex].Split(':')[1], data.Split(',')[ContactNumberIndex].Split(':')[1], data.Split(',')[InvestedAmountIndex].Split(':')[1], data.Split(',')[InvestedOnIndex].Split(':')[1].Split(' ')[0] + " " + data.Split(',')[InvestedOnIndex].Split(' ')[2], data.Split(',')[InvestmentTermIndex].Split(':')[1], data.Split(',')[InvestmentRateIndex].Split(':')[1], data.Split(',')[InvestmentBonusIndex].Split(':')[1], data.Split(',')[CompundedInterestIndex].Split(':')[1], data.Split(',')[AccuredInterestIndex].Split(':')[1]);

}

SummaryGrid.DataSource = SummaryTable;

SearchKeyTextBox.Focus();

//Computes Calculated Fields

ComputeCalcuatedSummaryFields(SummaryTable);

}

#endregion

#region Summary Calculated Fields

//Computes Total Transactions, Total Investment, Total Interest and Average Terms

public void ComputeCalcuatedSummaryFields(DataTable dataTable)

{

decimal TotalInvestment = 0.0m, TotalInterest = 0.0m, TotalTerm = 0.0m;

// Datatable Column Ordinality index of Calculation fields

int SummaryInvestmentIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Summary\_Investment\_Index"));

int SummaryInvestmentTermIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Summary\_Investment\_Term\_Index"));

int SummaryTotalInterest = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Summary\_Total\_Interest"));

int TotalTransactions = dataTable.Rows.Count;

for (int i = 0; i < TotalTransactions; i++)

{

TotalInvestment = TotalInvestment + Convert.ToInt32(dataTable.Rows[i].ItemArray[SummaryInvestmentIndex].ToString().Replace("€", string.Empty).Trim());

TotalTerm = TotalTerm + Convert.ToInt32(dataTable.Rows[i].ItemArray[SummaryInvestmentTermIndex].ToString().Trim());

TotalInterest = TotalInterest + Convert.ToDecimal(dataTable.Rows[i].ItemArray[SummaryTotalInterest].ToString().Replace("€", string.Empty).Trim());

}

TotalTransactionTextBox.Text = TotalTransactions.ToString();

TotalInvestmentTextBox.Text = "€" + TotalInvestment.ToString("f2");

TotalInterestTextBox.Text = "€" + TotalInterest.ToString("f2");

AverageTermTextBox.Text = Convert.ToDecimal((TotalTerm / TotalTransactions)).ToString("f2");

}

#endregion

#region Search from Summary

//Check if the search key is Transaction ID or Email

public bool IsSearchKeyTrasanctionID()

{

int \_tempTransactionID = 0;

try

{

\_tempTransactionID = int.Parse(SearchKeyTextBox.Text);

//TransactionID

return true;

}

catch (Exception)

{

//EmailID

return false;

}

}

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

{

string SearchKey = SearchKeyTextBox.Text.Trim();

int SummaryTransactionIDIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Summary\_TransactionID\_Index"));

int SummaryEmailIDIndex = Convert.ToInt32(ConfigurationManager.AppSettings.Get("Summary\_EmailID\_Index"));

DataTable SearchResults = new DataTable();

if (!string.IsNullOrWhiteSpace(SearchKey))

{

bool flag = IsSearchKeyTrasanctionID();

string ColumnHeaders = ConfigurationManager.AppSettings.Get("SummaryColumnHeader").ToString();

for (int i = 0; i < ColumnHeaders.Split('|').Length; i++)

{

SearchResults.Columns.Add(ColumnHeaders.Split('|')[i], typeof(string));

}

if (flag)

{

foreach (DataRow viewRow in SummaryTable.Rows)

{

if (viewRow.ItemArray[SummaryTransactionIDIndex].ToString().Contains(SearchKey))

{

SearchResults.Rows.Add(viewRow.ItemArray);

}

}

}

else

{

foreach (DataRow viewRow in SummaryTable.Rows)

{

if (viewRow.ItemArray[SummaryEmailIDIndex].ToString().Contains(SearchKey))

{

SearchResults.Rows.Add(viewRow.ItemArray);

}

}

}

SummaryGrid.DataSource = SearchResults;

if (SearchResults.Rows.Count > 0)

{

ComputeCalcuatedSummaryFields(SearchResults);

}

else

{

MessageBox.Show("No Matching Transactions Found!", "No Results", MessageBoxButtons.OK, MessageBoxIcon.Warning);

SummaryGrid.DataSource = SummaryTable;

ComputeCalcuatedSummaryFields(SummaryTable);

SearchKeyTextBox.Text = string.Empty;

SearchKeyTextBox.Focus();

}

}

else

{

MessageBox.Show("Enter a valid Search Key", "Invalid Search Key", MessageBoxButtons.OK, MessageBoxIcon.Error);

SearchKeyTextBox.Text = string.Empty;

SearchKeyTextBox.Focus();

}

}

#endregion

#region Reset Summary

//Reset Summary to the Original State

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

{

SearchKeyTextBox.Text = string.Empty;

SearchKeyTextBox.Focus();

//Reset Grid

SummaryGrid.DataSource = SummaryTable; // Global Variable

ComputeCalcuatedSummaryFields(SummaryTable);

}

#endregion

#region Redirect to Home Page

// Redirects to Home Page from Summary Page

private void HomePageButton2\_Click(object sender, EventArgs e)

{

SearchKeyTextBox.Text = string.Empty;

InitializeHomePage();

}

#endregion

#region Exit

// Exits from Summary Page

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

{

Close();

}

#endregion

#endregion

**App.Config**

<appSettings>

<!--Investment Plan Details-->

<add key="Category\_1\_Record\_1" value="1,0.50000"/>

<add key="Category\_1\_Record\_2" value="3,0.62500"/>

<add key="Category\_1\_Record\_3" value="5,0.71250"/>

<add key="Category\_1\_Record\_4" value="10,1.01500"/>

<add key="Category\_2\_Record\_1" value="1,0.60000"/>

<add key="Category\_2\_Record\_2" value="3,0.72500"/>

<add key="Category\_2\_Record\_3" value="5,0.81250"/>

<add key="Category\_2\_Record\_4" value="10,1.02500"/>

<add key="Total\_Terms" value="4"/>

<!--Investment Details Grid - Column Headers-->

<add key="ColumnHeader" value="Terms|Interest\_Rate|Investment\_Amount(€)|Interest\_Amount(€)|Bonus(€)|Return\_on\_Investment-ROI(€)"/>

<!--Investment Plan Grid - Column Ordinality-->

<add key="Term\_Index" value="0"/>

<add key="Interest\_Rate\_Index" value="1"/>

<add key="Investment\_Principle\_Index" value="2"/>

<add key="Interest\_Amount\_Index" value="3"/>

<add key="ROI\_Index" value="5"/>

<!--File String Format-->

<add key="DataString" value="TransactionID:{Transaction\_ID},Name:{Customer\_Name},Email:{Email\_ID},Contact Number:{Contact\_Number},Invested Amount:{Investment\_Amount},Invested On:{Created\_On},Investment Term:{Investment\_Term},Interest Rate:{Interest\_Rate},Bonus:{Bonus\_Amount},Interest Amount:{Interest\_Amount},Accured Interest:{Accured\_Interest}"></add>

<!--Ordinality of Columns in File-->

<add key="TransactionID" value="0"/>

<add key="Name" value="1"/>

<add key="Email" value="2"/>

<add key="Contact Number" value="3"/>

<add key="Invested Amount" value="4"/>

<add key="Invested On" value="5"/>

<add key="Investment Term" value="6"/>

<add key="Interest Rate" value="7"/>

<add key="Bonus" value="8"/>

<add key="Interest Amount" value="9"/>

<add key="Accured Interest" value="10"/>

<!--Summary Grid Column Headers-->

<add key="SummaryColumnHeader" value="Transaction\_ID|Name|Email\_ID|Contact\_Number|Principal\_Investment\_Amount|Invested\_On|Investment\_Term|Interest\_Rate|Bonus|Interest\_Amount\_By\_EOT|Return\_on\_Investment\_ROI"/>

<!--Summary Table Column Ordinality-->

<!--Summary Columns Used for Calculation Fields-->

<add key="Summary\_Investment\_Index" value="4"/>

<add key ="Summary\_Investment\_Term\_Index" value="6"/>

<add key="Summary\_Total\_Interest" value="9"/>

<!--Summary Columns Used for Search-->

<add key="Summary\_TransactionID\_Index" value="0"/>

<add key="Summary\_EmailID\_Index" value="2"/>

</appSettings>