**FORM 1**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace \_2175\_MGIS\_350\_01\_Final

{

public partial class Form1 : Form

{

CarDatabaseClass myCD = new CarDatabaseClass();

public Form1()

{

InitializeComponent();

// Setting up combobox

cboCarMakeModel.DropDownStyle = ComboBoxStyle.DropDownList;

cboCarMakeModel.Items.Add("Chevy Cruz");

cboCarMakeModel.Items.Add("Ford Focus");

cboCarMakeModel.SelectedIndex = 0;

}

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

{

// Read in input

double input = 0.0;

string input2;

if (double.TryParse(txtPurchasePrice.Text, out input))

{

DialogResult drAnswer = DialogResult.Yes;

// Test to confirm the user intended to enter a negative number.

if (input < 0.0)

{

drAnswer = MessageBox.Show("Transaction Amount invalid. Please correct.", "Input error.", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else

{

myCD.AddPurchases(input, txtCustomerName.Text, cboCarMakeModel.Text);

// Update accounts

//myCD.PopulateModels(cboCarMakeModel);

// Display transactions for the current account.

myCD.DisplayPurchases(cboCarMakeModel.Text, lstPurchases);

// Output balance for current account.

lblAveragePrice.Text = myCD.GetBalance(cboCarMakeModel.Text).ToString("c");

// Clearing input.

txtPurchasePrice.Text = string.Empty;

}

}

}

private void cboCarMakeModel\_SelectedIndexChanged(object sender, EventArgs e)

{

// Display transactions for the current model

myCD.DisplayPurchases(cboCarMakeModel.Text, lstPurchases);

// Output balance for current account.

lblAveragePrice.Text = myCD.GetBalance(cboCarMakeModel.Text).ToString("c");

}

}

}

**CAR DATABASE**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.Odbc; // Database Libraries

using System.Diagnostics; // Debugging Libraries

namespace \_2175\_MGIS\_350\_01\_Final

{

class CarDatabaseClass

{

// Database Objects

OdbcConnection dbConn;

OdbcCommand dbCmd;

OdbcDataReader dbReader;

List<int> ids;

public CarDatabaseClass()

{

dbConn = new OdbcConnection("Driver={SQL Server};"

+ "Server=scb-sv-mgis-4.main.ad.rit.edu\\mgismssql;"

+ "DataBase=MGIS350\_2175\_29;"

+ "Uid=vv3210;"

+ "Pwd=duyuxljj$&4949CNVY");

ids = new List<int>();

}

private void ExecuteQuery(string Query)

{

try

{

Debug.WriteLine("");

Debug.WriteLine("SQL Query: " + Query);

Debug.WriteLine("");

// Building command to database and executing the query string

dbCmd = new OdbcCommand(Query, dbConn);

// Opening the connection

dbConn.Open();

// Executing the query and closing the connection

dbReader = dbCmd.ExecuteReader();

}

catch (Exception ex)

{

// Determine if database is open. If so close it.

if (dbConn.State.ToString() == "Open")

{

this.CloseDatabase();

}

// Display Error

System.Windows.Forms.MessageBox.Show("Error:\n\n" + ex.ToString() + "\n");

}

}

private void CloseDatabase()

{

// Determine if database is open. If so close it.

if (dbConn.State.ToString() == "Open")

{

// Closing connection

dbConn.Close();

}

}

public void DisplayPurchases(string carMakeModel, System.Windows.Forms.ListBox List)

{

// Clearing output and ids list

List.Items.Clear();

ids.Clear();

this.ExecuteQuery("SELECT \* FROM finalExam WHERE carMakeModel = '" + carMakeModel + "';");

// Loop through the models

while (dbReader.Read())

{

// Ensure the results are not null

if (dbReader.IsDBNull(0) == false)

{

// Adding each transaction to the output.

List.Items.Add(Convert.ToDouble(dbReader["purchasePrice"]).ToString("c"));

// Adding the id for each transaction to the list of ids.

ids.Add(Convert.ToInt32(dbReader["id"]));

}

}

// Closing the database connection.

this.CloseDatabase();

}

public void AddPurchases(double purchasePrice, string customerName, string carMakeModel)

{

// Setting up query to insert new transaction.

this.ExecuteQuery("INSERT INTO finalExam (customerName, carMakeModel, purchasePrice) VALUES ('" + customerName + "','" + carMakeModel + "','" + purchasePrice + "');");

this.CloseDatabase();

}

public double GetBalance(string carMakeModel)

{

// Setting default return value

double balance = 0.0;

// Querying balance for selected account

this.ExecuteQuery("SELECT AVG(purchasePrice) AS Balance FROM finalExam WHERE carMakeModel = '" + carMakeModel + "';");

while (dbReader.Read())

{

// Ensure the results are not null

if (dbReader.IsDBNull(0) == false)

{

balance += Convert.ToDouble(dbReader["Balance"]);

}

}

this.CloseDatabase();

// Returning balance

return balance;

}

public double GetIndividualTransactionAmount(int index)

{

// Return value

double purchasePrice = 0;

// Query for finding transaction

this.ExecuteQuery("SELECT purchasePrice FROM finalExam WHERE id = " + ids[index] + ";");

// Loop through accounts and add to combobox.

while (dbReader.Read())

{

// Ensure the results are not null

if (dbReader.IsDBNull(0) == false)

{

purchasePrice = Convert.ToDouble(dbReader["purchasePrice"]);

}

}

// Closing database.

this.CloseDatabase();

// Returning value

return purchasePrice;

}

/\*public void PopulateModels(System.Windows.Forms.ComboBox cboCarMakeModel)

{

// Capturing current account name

string current = cboCarMakeModel.Text;

// Clearing accounts

cboCarMakeModel.Items.Clear();

// Query for finding unique accounts

this.ExecuteQuery("SELECT DISTINCT carMakeModel FROM finalExam ORDER BY carMakeModel ASC;");

// Loop through accounts and add to combobox.

while (dbReader.Read())

{

// Ensure the results are not null

if (dbReader.IsDBNull(0) == false)

{

cboCarMakeModel.Items.Add(dbReader["carMakeModel"]);

}

}

// Resetting current account name.

this.CloseDatabase();

cboCarMakeModel.Text = current;

}\*/

}

}