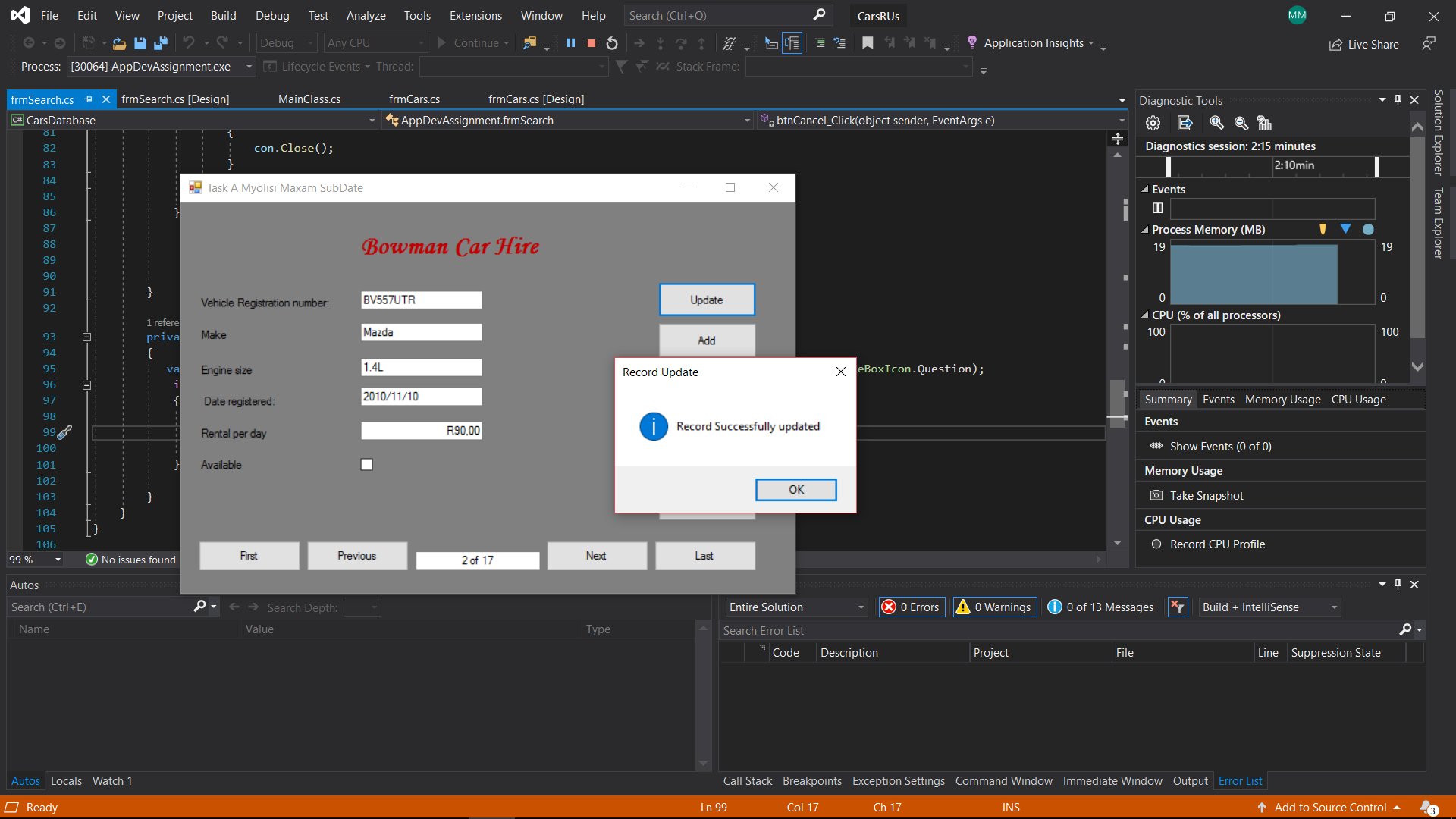
Task B 1

Update:

Expected result: Already existing data successfully updates

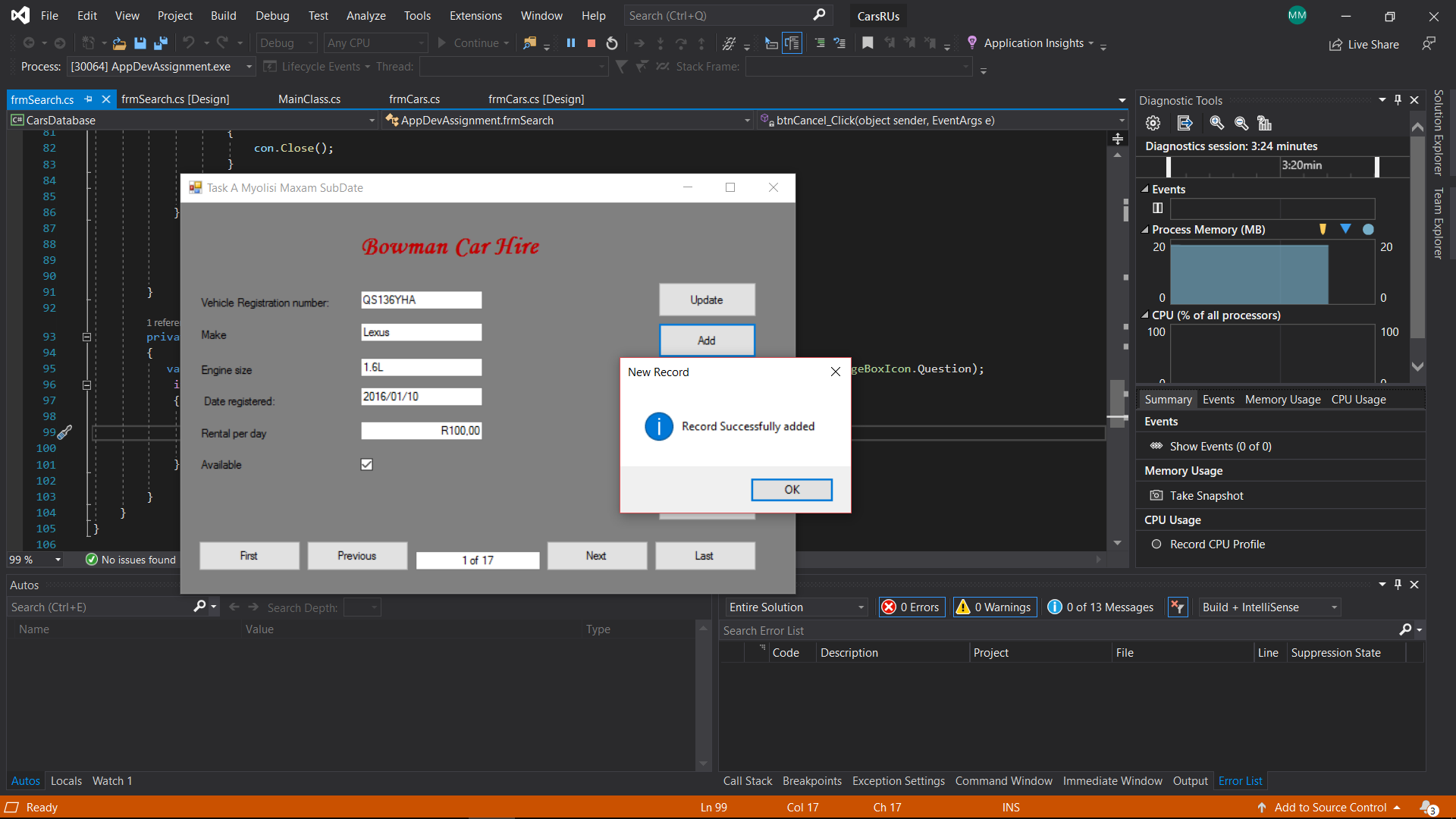
Actual result: Data is able to be updated as expected



Add:

Expected result: Insert new data into the existing Hire database

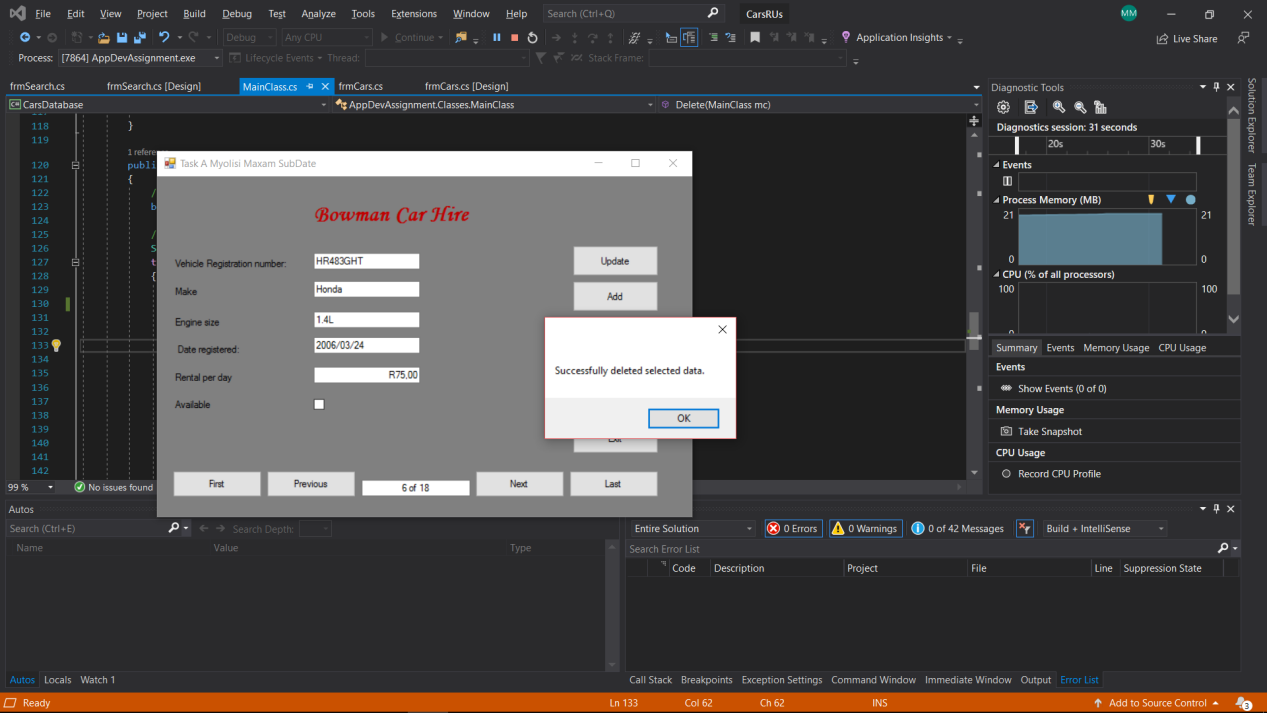
Actual result: New records of data can successfully be added



Delete:

Expected result: Delete current record of data in the database

Actual result: Successfully able to delete data

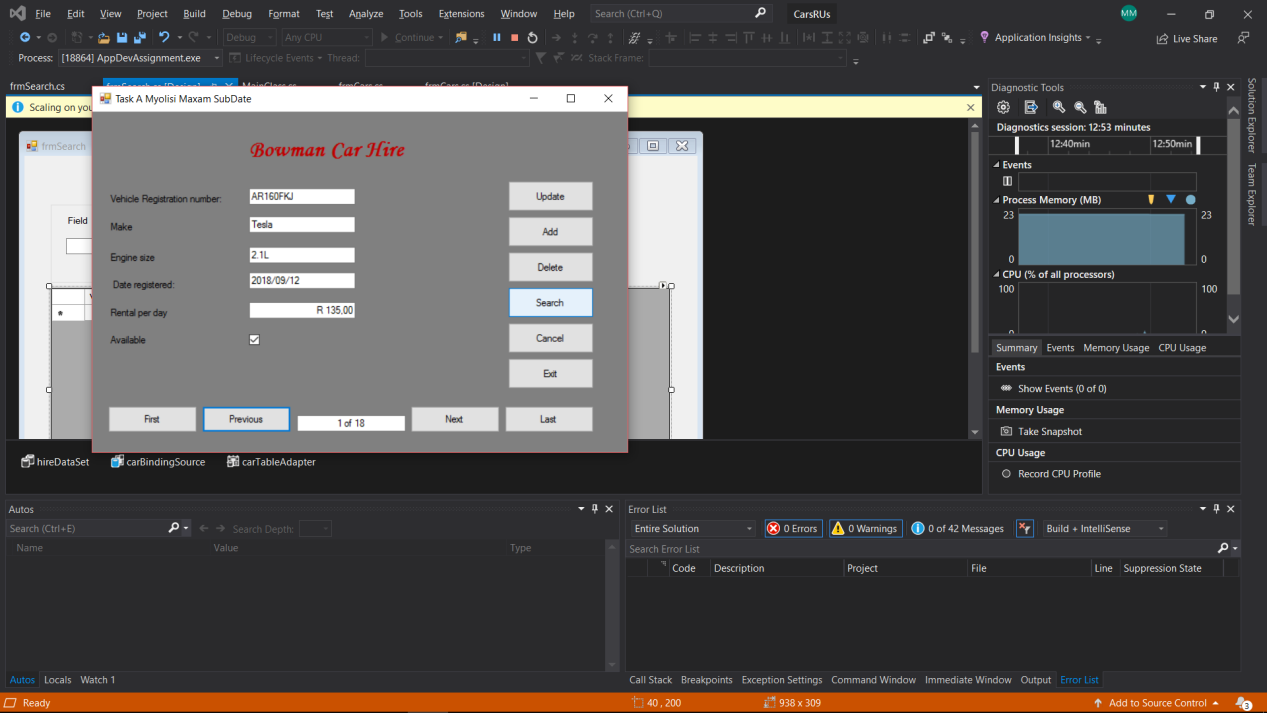


Search:

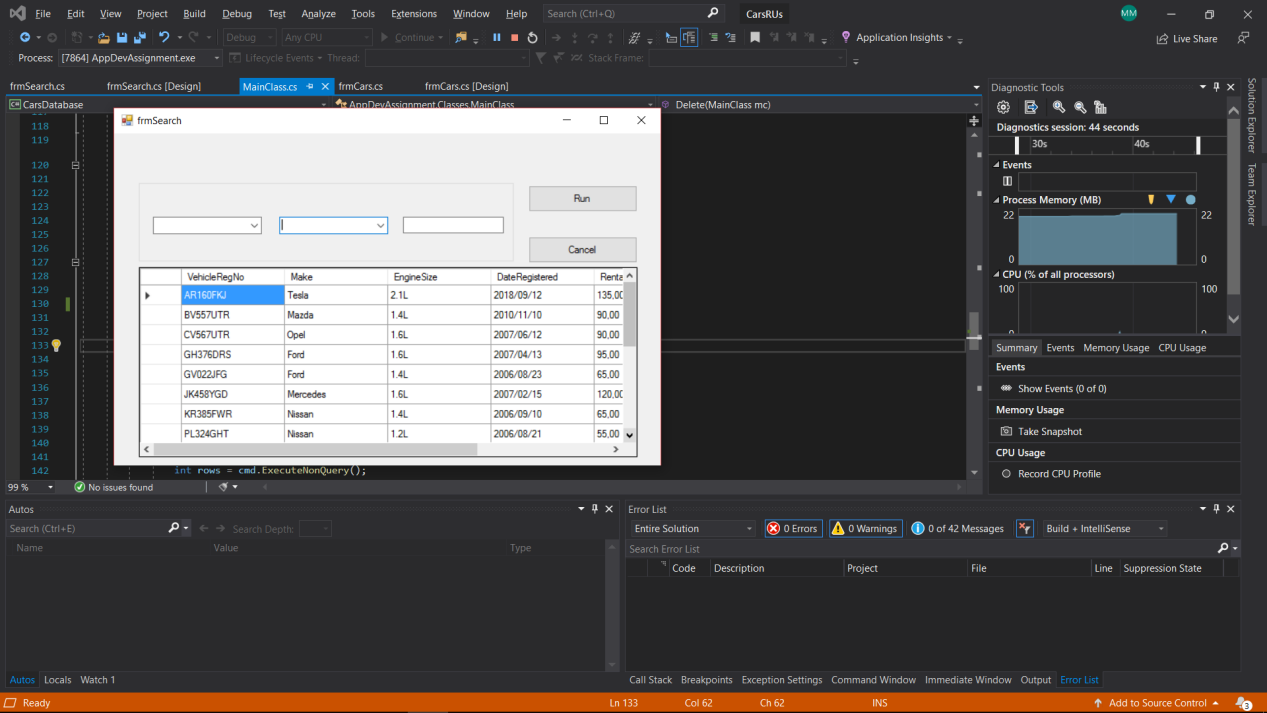
Expected result: Displays frmSearch while hiding frmCar

Actual result: Expected results have been met by actual results

Before:



After:

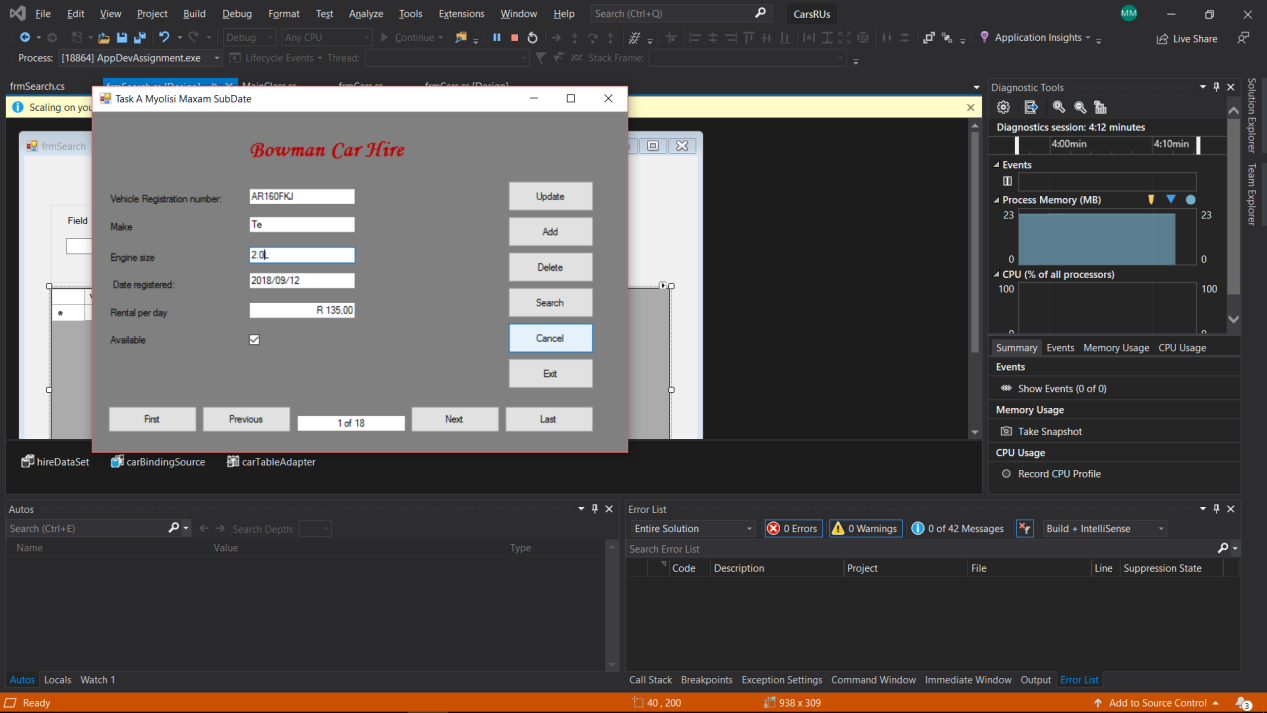


Cancel:

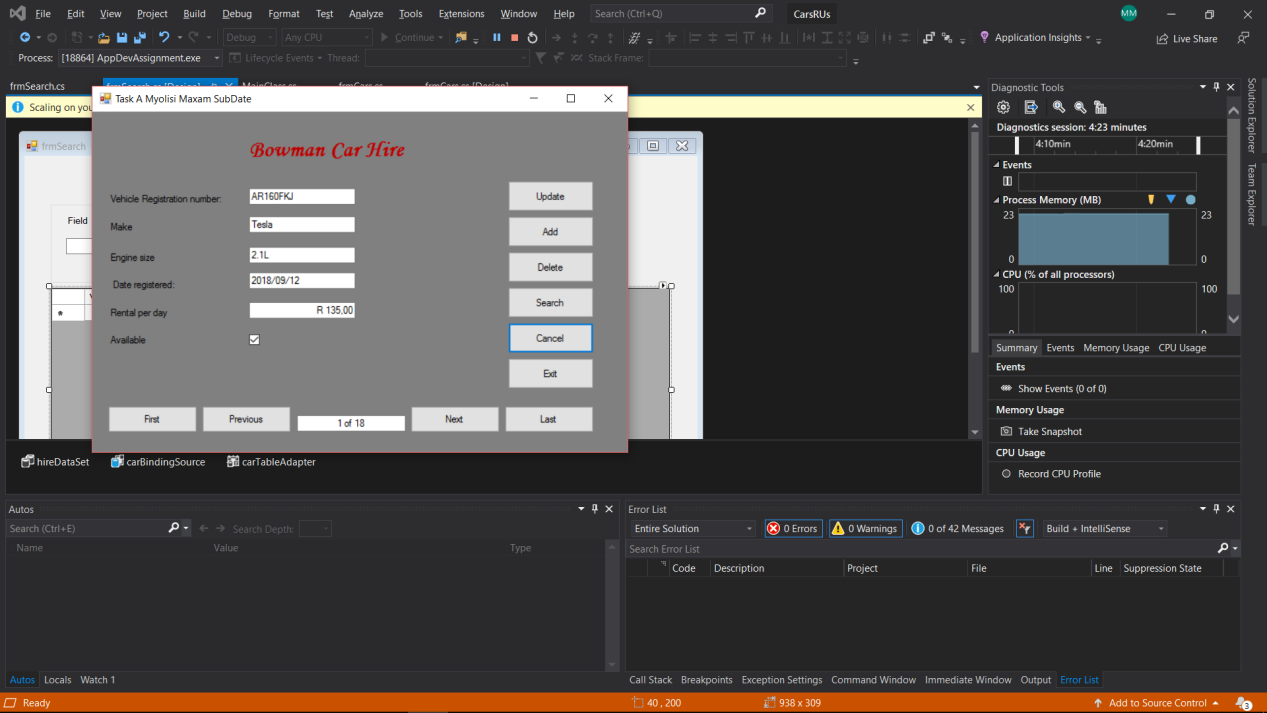
Expected result:

Actual result:

Before:



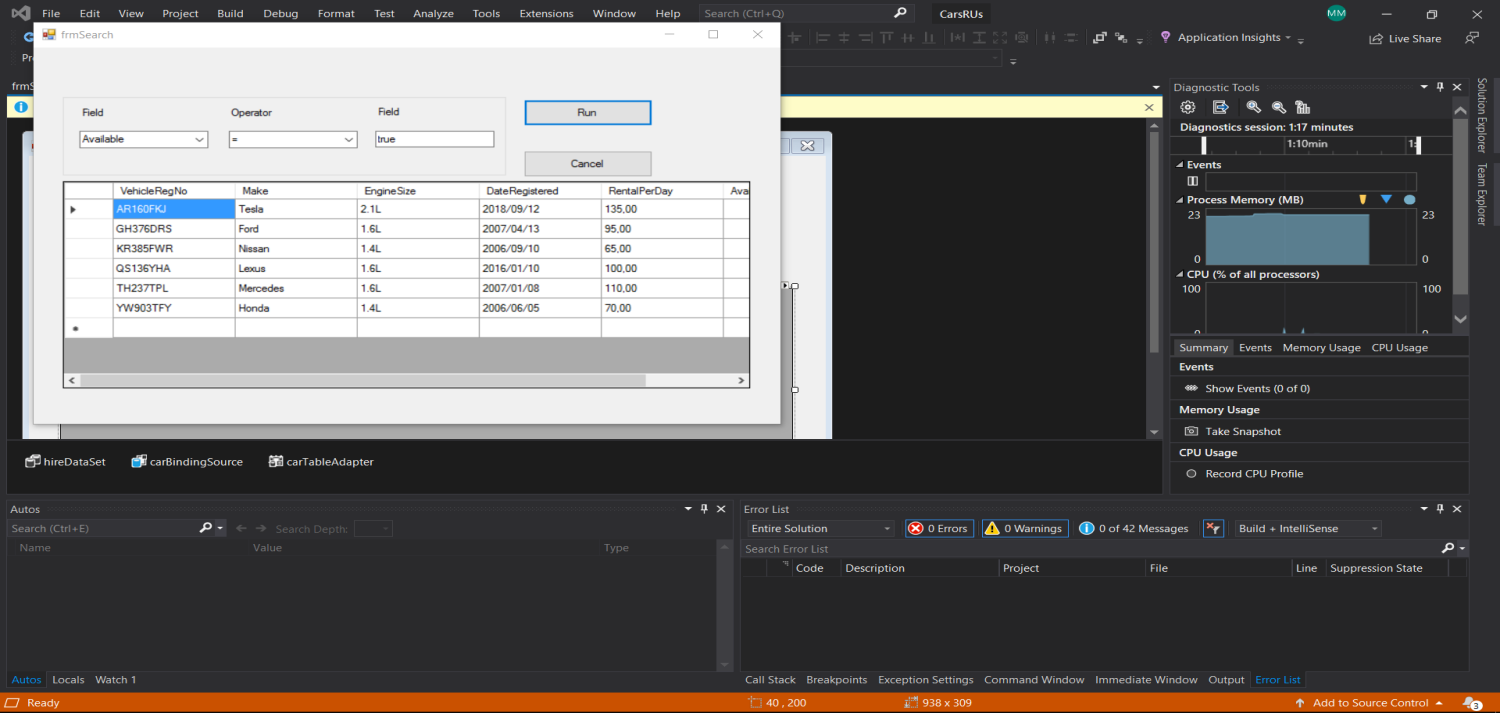
After:



Run:

Expect result: Filters through the Car table database to display data using combo boxes and a text boxes.

Actual result: Expected results have been met by actual result

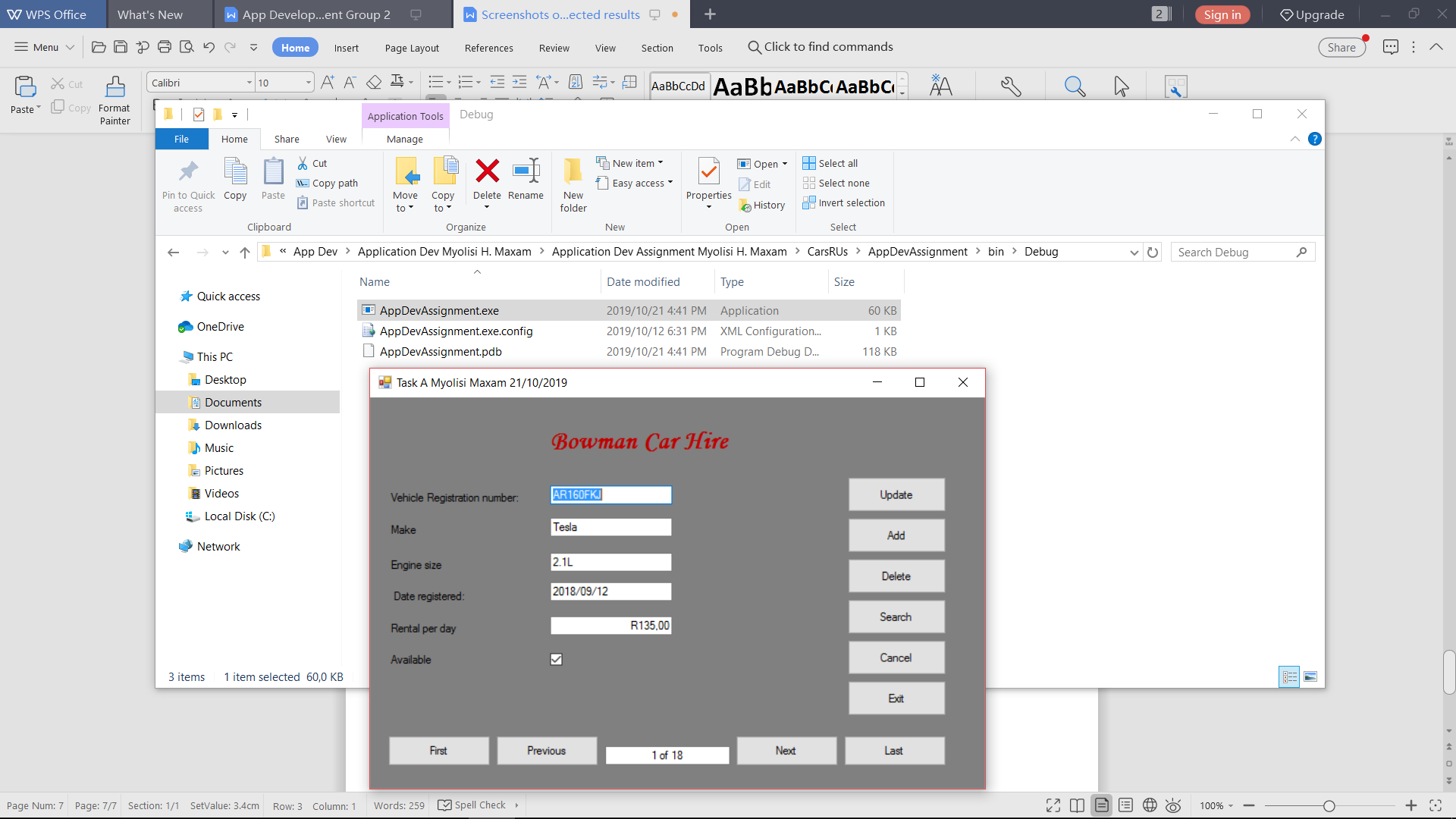


Task B 2

Test plan:

|  |  |  |
| --- | --- | --- |
| Buttons | Expected | Actual |
| Update | Alter/Update record of data that already exist in the database | Expected results ran successfully |
| Add | Insert new data into the existing Hire database | Expected results ran successfully |
| Delete | Delete a row of data in the database | Expected results ran successfully |
| Search | Displays a frmSearch while hiding frmCar | Expected results ran successfully |
| First | Displays the first record in the database | Expected results ran successfully |
| Next | Displays the next record of data in the database | Expected results ran successfully |
| Previous | Displays the previous record of data in the database. | Expected results ran successfully |
| Last | Displays the last record of data in the database. | Expected results ran successfully |
| Cancel | Displays the current record of data | Expected results ran successfully |
| Run | Filters through the Car table database to display data using combo boxes and a text boxes. | Expected results ran successfully |

Task B 3:



Task B 4:

Planning stage

This is a car rental application that can access data in a external database to allow a user to see individual cars that are available or not. The application is able to add, delete and update data in the database and also search data in the records.

The application will be created in C# programming language and the design requirements are stated in assignment.

Creating and Testing

The application is created with a .Net Framework (4.7.2) on Visual Studio 2019 with a User Interface(UI). The database is created on Microsoft SQL Server Management Studio 2017 and is linked to the application externally.

Quality Assurance testing is required for each feature in application during and after creating of the application.

Final product

Application’s exe file will be used for installing the application. The actual of the program will have comments for any other programmer to understand what is actually happening each method and class. The product meets ultimately meets the requirements stated in the assignment only if the connection to database is established.

Task B 5:

**frmCars**

using AppDevAssignment.Classes;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Globalization;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace AppDevAssignment

{

public partial class frmCars : Form

{

MainClass mc = new MainClass();

public frmCars()

{

InitializeComponent();

}

int R = 0;

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

{

//Code used to update record of data in the database

mc.VehicleRegNo = txtVRegNumb.Text;

mc.Make = txtMake.Text;

mc.EngineSize = txtEngineSize.Text;

double rent = double.Parse(txtRentPerDay.Text.Replace("R", ""));

mc.RentalPerDay = rent;

mc.DateRegistered = txtDateReg.Text;

if (cbxAvailable.Checked)

{

mc.Available = true;

}

else

{

mc.Available = false;

}

bool success = mc.Update(mc);

if (success)

{

MessageBox.Show("Record Successfully updated", "Record Update", MessageBoxButtons.OK, MessageBoxIcon.Information);

DataTable dt = mc.Select();

txtRecord.Text = R + " of " + dt.Rows.Count;

}

else

{

MessageBox.Show("Record failed to update","Update failure", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

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

{

//Receiving and sending all data in the textfields to the Insert methods in the MainClass class

mc.VehicleRegNo = txtVRegNumb.Text;

mc.Make = txtMake.Text;

mc.EngineSize = txtEngineSize.Text;

double rent = double.Parse(txtRentPerDay.Text.Replace("R",""));

mc.RentalPerDay = rent;

mc.DateRegistered = txtDateReg.Text;

if (cbxAvailable.Checked)

{

mc.Available = true;

}

else

{

mc.Available = false;

}

bool success = mc.Insert(mc);

if (success)

{

MessageBox.Show("Record Successfully added","New Record", MessageBoxButtons.OK, MessageBoxIcon.Information);

DataTable dt = mc.Select();

R = 0;

txtRecord.Text = (R+1) + " of " + dt.Rows.Count;

}

else

{

MessageBox.Show("Program failed to add new record");

}

}

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

{

//Code deletes a record of data in the database by using the VehicleRegNo as reference

mc.VehicleRegNo = txtVRegNumb.Text;

bool success = mc.Delete(mc);

if (success)

{

MessageBox.Show("Successfully deleted selected data.");

DataTable dt = mc.Select();

txtRecord.Text = 1 + " of " + (dt.Rows.Count + 1);

}

else

{

MessageBox.Show("Failed to Delete selected data.");

}

}

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

{

this.Hide();

//Opens a new windows form called frmSearch

frmSearch search = new frmSearch();

search.Show();

}

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

{

DataTable dt = mc.Select();

//Data is repopulated by the current record

txtVRegNumb.Text = dt.Rows[R]["VehicleRegNo"].ToString();

txtMake.Text = dt.Rows[R]["Make"].ToString();

txtEngineSize.Text = dt.Rows[R]["EngineSize"].ToString();

txtDateReg.Text = timeRemove();

txtRentPerDay.Text = "R " + dt.Rows[R]["RentalPerDay"].ToString();

cbxAvailable.Checked = (bool)dt.Rows[R]["Available"];

}

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

{

//Closes application

var result = MessageBox.Show("Are you sure you want to leave?", "Exit", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (result == DialogResult.Yes)

{

this.Close();

}

}

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

{

DataTable dt = mc.Select();

//When app loads data of the first record in the database is automatically displayed

txtVRegNumb.Text = dt.Rows[R]["VehicleRegNo"].ToString();

txtMake.Text = dt.Rows[R]["Make"].ToString();

txtEngineSize.Text = dt.Rows[R]["EngineSize"].ToString();

txtDateReg.Text = timeRemove();

txtRentPerDay.Text = "R" + dt.Rows[R]["RentalPerDay"].ToString();

cbxAvailable.Checked = (bool)dt.Rows[R]["Available"];

//Displays what record is being displayed

txtRecord.Text = (R + 1) + " of " + dt.Rows.Count;

}

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

{

DataTable dt = mc.Select();

//Code displays the first record in the database

R = 0;

txtVRegNumb.Text = dt.Rows[0]["VehicleRegNo"].ToString();

txtMake.Text = dt.Rows[0]["Make"].ToString();

txtEngineSize.Text = dt.Rows[0]["EngineSize"].ToString();

txtDateReg.Text = timeRemove();

txtRentPerDay.Text = "R" + dt.Rows[0]["RentalPerDay"].ToString();

cbxAvailable.Checked = (bool)dt.Rows[0]["Available"];

//Displays what record is being displayed

txtRecord.Text = (R + 1) + " of " + dt.Rows.Count;

}

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

{ //Loads the previous record of data in the database

DataTable dt = mc.Select();

if(R > 0)

{

R--;

txtVRegNumb.Text = dt.Rows[R]["VehicleRegNo"].ToString();

txtMake.Text = dt.Rows[R]["Make"].ToString();

txtEngineSize.Text = dt.Rows[R]["EngineSize"].ToString();

txtDateReg.Text = timeRemove();

txtRentPerDay.Text = "R" + dt.Rows[R]["RentalPerDay"].ToString();

cbxAvailable.Checked = (bool)dt.Rows[R]["Available"];

}

if (R == 0)

{

txtRecord.Text = (R + 1) + " of " + dt.Rows.Count;

}

else

{

txtRecord.Text = (R + 1) + " of " + dt.Rows.Count;

}

}

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

{ //Loads the next record of data in the database

DataTable dt = mc.Select();

if (R >= 0 && R < dt.Rows.Count-1)

{

R++;

txtVRegNumb.Text = dt.Rows[R]["VehicleRegNo"].ToString();

txtMake.Text = dt.Rows[R]["Make"].ToString();

txtEngineSize.Text = dt.Rows[R]["EngineSize"].ToString();

txtDateReg.Text = timeRemove();

txtRentPerDay.Text = "R" + dt.Rows[R]["RentalPerDay"].ToString();

cbxAvailable.Checked = (bool)dt.Rows[R]["Available"];

txtRecord.Text = (R+1) + " of " + dt.Rows.Count;

}

}

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

{ //Loads the last record of data in the database

DataTable dt = mc.Select();

R = dt.Rows.Count-1;

txtVRegNumb.Text = dt.Rows[R]["VehicleRegNo"].ToString();

txtMake.Text = dt.Rows[R]["Make"].ToString();

txtEngineSize.Text = dt.Rows[R]["EngineSize"].ToString();

txtDateReg.Text = timeRemove();

txtRentPerDay.Text = "R" + dt.Rows[R]["RentalPerDay"].ToString();

cbxAvailable.Checked = (bool)dt.Rows[R]["Available"];

txtRecord.Text = (R+1) + " of " + dt.Rows.Count;

}

public string timeRemove()

{

DataTable dt = mc.Select();

string date = txtDateReg.Text = dt.Rows[R]["DateRegistered"].ToString();

date = date.Substring(0, date.IndexOf(" "));

return date;

}

}

}

**frmSearch:**

using AppDevAssignment.Classes;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace AppDevAssignment

{

public partial class frmSearch : Form

{

public frmSearch()

{

InitializeComponent();

}

static string myConString = ConfigurationManager.ConnectionStrings["Hire"].ConnectionString;

DataTable dt = new DataTable("Hire");

DataSet ds = new DataSet();

MainClass mc = new MainClass();

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

{

//Code populates datagrid table with the Hire database

this.carTableAdapter.Fill(this.hireDataSet.Car);

DataTable dt = mc.Select();

dgvCarList.DataSource = dt;

//Code populates all combo boxes with the appropriate combo box items

cboField.Items.Add("Make");

cboField.Items.Add("EngineSize");

cboField.Items.Add("RentalPerDay");

cboField.Items.Add("Available");

cboOperator.Items.Add("=");

cboOperator.Items.Add("<");

cboOperator.Items.Add(">");

cboOperator.Items.Add("<=");

cboOperator.Items.Add(">=");

}

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

{ //Code uses the options selected in the combo boxes and the text in the text box to filter through the database to display data that the user wants to see

ds.Reset();

if (cboField.Text != "" && cboOperator.Text != "" && txtValue.Text != "")

{

string Field = cboField.Text.ToString();

string Operator = cboOperator.Text.ToString();

string Value = txtValue.Text;

string search = Field + " " + Operator + " '" + Value + "'";

using (SqlConnection con = new SqlConnection(myConString))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand();

cmd.Connection = con;

cmd.CommandType = CommandType.Text;

cmd.CommandText = "SELECT \* FROM Car WHERE " + @search;

SqlDataAdapter sqlData = new SqlDataAdapter(cmd);

sqlData.Fill(ds);

dgvCarList.DataSource = ds.Tables[0];

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString(), "Error updating data", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

con.Close();

}

}

}

}

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

{

var result = MessageBox.Show("Are you sure you want to leave?", "Exit", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (result == DialogResult.Yes)

{

this.Close();

frmCars cars = new frmCars();

cars.Show();

}

}

}

}