Assessment Task B - Ciarán Maher – Event-driven Programming using C#

Table of Contents

Ctrl + Left click to follow hyperlinks

Test Plan – Back to Top

Date	Version	Description	Expected	Result
21st-Sept	v0.4	Check Update Button	Clicking update should submit any changes to the database.	Ok
21st-Sept	v0.5	Check Add Button	Clicking Add should create a new entry in the database and await inputs.	Fail – See Test Log
21st-Sept	v0.5	Check Delete Button	Clicking Delete should remove the current entry from the database.	Ok
21st-Sept	v0.5	Check Cancel Button	Clicking Cancel should discard any unsaved changes.	Ok
21st-Sept	v0.6	Check Search Button	Clicking the Search button should create a new instance of the frmSearch and display a new form.	Ok
21st-Sept	v0.7	Re-check Add Button	Clicking Add should create a new entry in the database and await inputs.	Ok

Test Log – Back to Top

Date	Version	Description	Result	Reproduction Steps	Action	Status
21st-Sept	v0.5	Check Add Button	Fail	Click 'Add' button, this should create a new entry in the database and await inputs.	Switch to ".AddNew" method, instead of manually adding a new row.	Ok

Test data – Back to Top

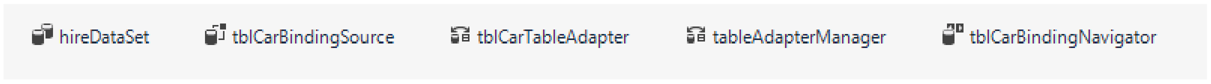
RegistrationNo.	Make	EngineSize	Date Registered	Rental Per Day	Available
14D31499	Nissan Micra	1.0	22/12/2003	670.90	False
14KY6123	Toyota	1.3	22/07/2014	950.00	True
151MO12234	Honda	1.1	19/04/2015	10010.00	True
98OY1003	Nissan	XL 1.5	01/01/2006	1300.00	True

Known Issues – Back to Top

Program throws up a recoverable error, if the user attempts to add a new entry to the database but then tries to navigate away using the control buttons.

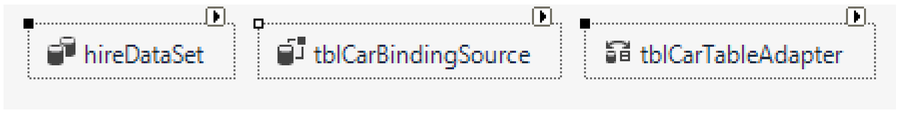
Connections – Back to Top

frmCars



hireDataSet tblCarBindingSource tblCarTableAdapter tableAdapterManager tblCarBindingNavigator

frmSearch



hireDataSet tblCarBindingSource tblCarTableAdapter

hireDataSet

Obtains the relevant data from the DataBase (Hire.mdf), this set of data is edited first and the passed to the original database to make changes via the tblCarTableAdapter.

tableAdapterManager

Uses the “Primary Key” to determine the correct order data is stored in the DataBase (Hire.mdf) and to send the Inserts, Updates, and Deletes whilst keeping the integrity of the DataBase.

tblCarBindingSource

The link between the DataSet (hireDataSet) and the record that is currently on the frmCars. Also keeps control of the Button/Options “First/Previous/Next/Last” record.

tblCarTableAdapter

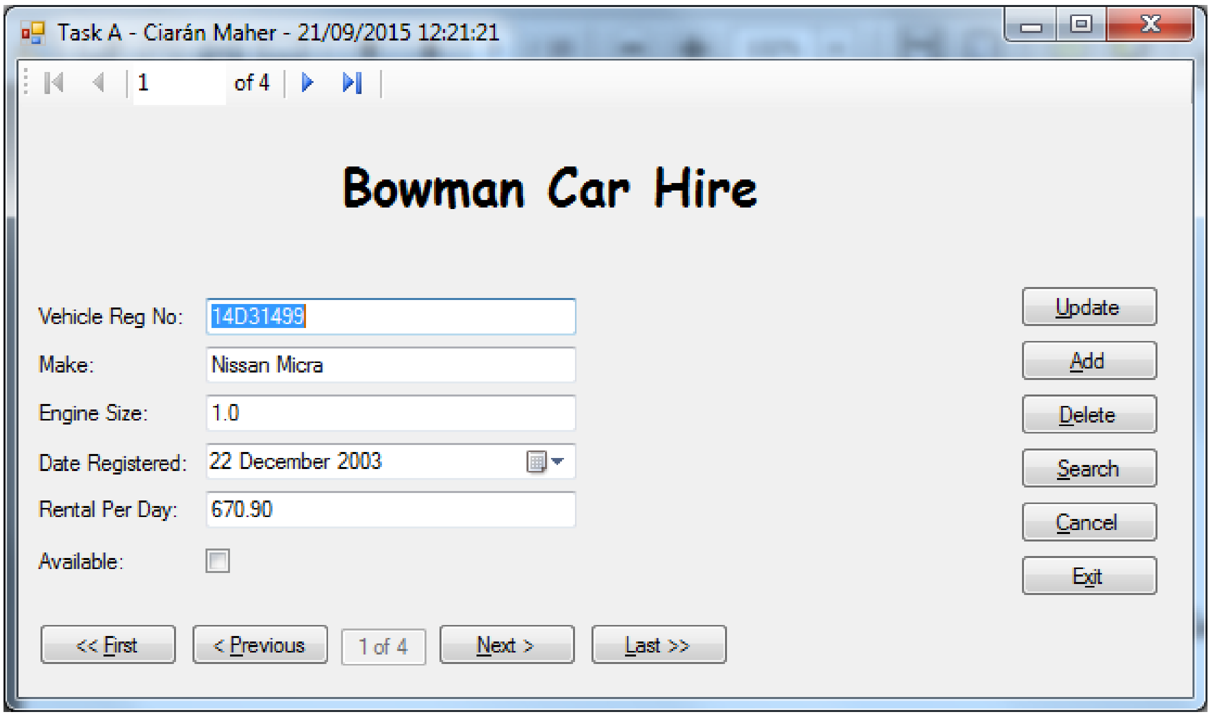
Links to the DataBase (Hire.mdf) and retrieves data from the specified table then returns the appropriate data.

Purpose

The purpose of this software is to allow a user to save, edit, and delete records in a database containing information regarding available cars for rental. This includes whether they are available or not, the rental price per day, as well as all other relevant details about the car; such as ‘Make’ and ‘Engine Size’

Screenshots – Back to Top

frmCars



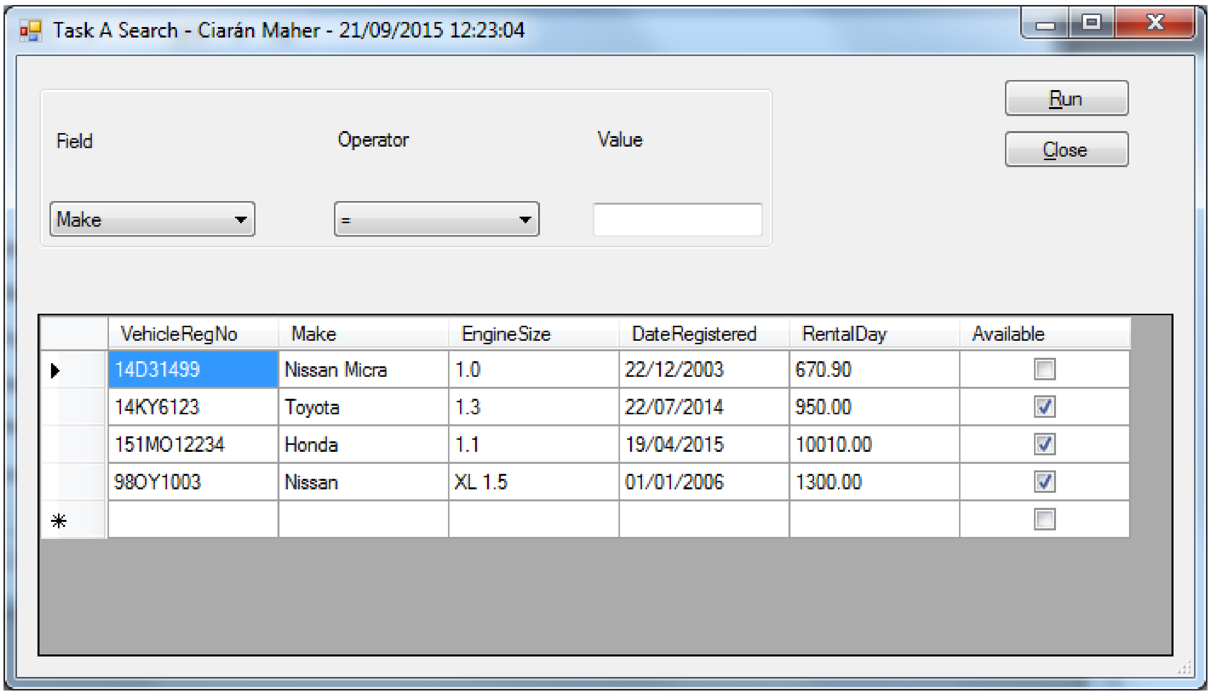
Task A - Ciarán Maher - 21/09/2015 12:21:21

1 of 4

Bowman Car Hire

Vehicle Reg No:	14D31499
Make:	Nissan Micra
Engine Size:	1.0
Date Registered:	22 December 2003
Rental Per Day:	670.90
Available:	☐

frmSearch



Task A Search - Ciarán Maher - 21/09/2015 12:23:04

Field	Operator	Value
Make	=	

VehicleRegNo	Make	EngineSize	DateRegistered	RentalDay	Available
14D31499	Nissan Micra	1.0	22/12/2003	670.90	☐
14KY6123	Toyota	1.3	22/07/2014	950.00	✓
151MO12234	Honda	1.1	19/04/2015	10010.00	✓
98OY1003	Nissan	XL 1.5	01/01/2006	1300.00	✓
					☐

Code – Back to Top

frmCars

```
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 CarsDatabase
{
    public partial class frmCars : Form
    {
        public frmCars()
        {
            InitializeComponent();
        }
        private void tblCarBindingNavigatorSaveItem_Click(object sender, EventArgs e)
        {
            this.Validate();
            this.tblCarBindingSource.EndEdit();
            this.tableAdapterManager.UpdateAll(this.hireDataSet);
        }
        private void Form1_Load(object sender, EventArgs e)
        {
            // TODO: This line of code loads data into the 'hireDataSet.tblCar' table. 
You can move, or remove it, as needed.
            this.tblCarTableAdapter.Fill(this.hireDataSet.tblCar);
            this.Text = "Task A - Ciarán Maher - " + DateTime.Now;
            updatePosition();
        }
        private void btnUpdate_Click(object sender, EventArgs e)
        {
            try
            {
                if (txtVehicleReg.Text != "")
                {
                    this.Validate();
                    this.tblCarBindingSource.EndEdit();
                    this.tblCarTableAdapter.Update(this.hireDataSet.tblCar);
                    MessageBox.Show("Update successful!");
                }
                else
                {
                    MessageBox.Show("Program requires a valid registration 
number!\nPlease try again. ");
                }
            }
                //Constraint specific exception catch, to prevent the user from input 
an invalid entry with tailored feedback to this error.
            catch (ConstraintException exCon)
            {
                MessageBox.Show("That Vehicle Registration Number already exists.\n" + 
"Original Error:\n" + exCon.Message);
            }
            //General exception catch, to prevent the user from input an invalid entry 
with tailored feedback to this error.
```

```
            catch (Exception ex)
            {
                MessageBox.Show("The program encountered an error.\n" + "Original
Error:\n" + ex.Message);
            }
        }
        private void btnAdd_Click(object sender, EventArgs e)
        {
            //Creates a new entry in the database and awaits user input for
confirmation.
            try
            {
                this.tblCarBindingSource.AddNew();
                this.tableAdapterManager.UpdateAll(this.hireDataSet);
                tblCarBindingSource.MoveLast();
                updatePosition();
            }
            catch (ConstraintException)
            {
                MessageBox.Show("Error updating database.\nRegistration number already
exists.");
            }
            catch (Exception ex)
            {
                MessageBox.Show("Unable to add new patient.\n" + "Do you have changes
pending?\n" + "Original Error:" + ex.Message);
            }
        }
        private void updatePosition()
        {
            txtPosition.Text = (tblCarBindingSource.Position + 1) + " of " +
tblCarBindingSource.Count;
        }
        private void btnDelete_Click(object sender, EventArgs e)
        {
            DialogResult dlgDeleteConfirmationResult;
            dlgDeleteConfirmationResult = MessageBox.Show("Are you sure you want to
delete this patient?", "Delete Confirmation", MessageBoxButtons.OKCancel,
MessageBoxIcon.Question);
            if (dlgDeleteConfirmationResult == DialogResult.OK)
            {
                try
                {
                    this.tblCarBindingSource.RemoveCurrent();
                    updatePosition();
                    MessageBox.Show("Delete successful.");
                }
                catch (Exception ex)
                {
                    MessageBox.Show("Delete failed. Is your database empty?\n" +
"Original error:\n" + ex.Message);
                }
            }
            else
            {
                MessageBox.Show("Delete cancelled.");
            }
        }
        private void btnSearch_Click(object sender, EventArgs e)
        {
            frmSearch frmSearch = new frmSearch();
            frmSearch.ShowDialog();
        }
        private void btnCancel_Click(object sender, EventArgs e)
        {
            hireDataSet.RejectChanges();
            tblCarBindingSource.ResetBindings(false);
            updatePosition();
        }
        private void btnExit_Click(object sender, EventArgs e)
        {
            this.Close();
        }
        private void btnNext_Click(object sender, EventArgs e)
        {
            tblCarBindingSource.MoveNext();
            updatePosition();
        }
        private void btnLast_Click(object sender, EventArgs e)
        {
            tblCarBindingSource.MoveLast();
            updatePosition();
        }
        private void btnFirst_Click(object sender, EventArgs e)
        {
            tblCarBindingSource.MoveFirst();
            updatePosition();
        }
        private void btnPrevious_Click(object sender, EventArgs e)
        {
            tblCarBindingSource.MovePrevious();
            updatePosition();
        }
    }

}
```

frmSearch

```
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;
using System.Data.SqlClient;

namespace CarsDatabase
{
    public partial class frmSearch : Form
    {
        public frmSearch()
        {
            InitializeComponent();
        }

        private void frmSearch_Load(object sender, EventArgs e)
        {
            // TODO: This line of code loads data into the 'hireDataSet.tblCar' table.
You can move, or remove it, as needed.
            this.tblCarTableAdapter.Fill(this.hireDataSet.tblCar);
            fillBoxes();
            this.Text = "Task A Search - Ciarán Maher - " + DateTime.Now;
        }

        private void btnSearch_Click(object sender, EventArgs e)
        {
            if (txtValue.Text != "")
            {
                try
                {
                    //Builds a string with the pre-set format to match the table
layout. FillBy querybuilder option was considered but this was found to be more
efficient.
                    string strMyQuery = String.Format("SELECT VehicleRegNo, Make,
EngineSize, DateRegistered, '€' + CAST(RentalDay AS varchar) AS RentalDay, Available
FROM tblCar WHERE {0} {1} @Third", cboField.SelectedItem, cboOperator.SelectedItem);

                    SqlConnection cnMyConnection = new SqlConnection(@"Data
Source=(LocalDB)\v11.0;AttachDbFilename=|DataDirectory|\Hire.mdf;Integrated
Security=True");
                    cnMyConnection.Open();
                    SqlCommand cmMySQLcommand = cnMyConnection.CreateCommand();


                    cmMySQLcommand.CommandType = CommandType.Text;
                    cmMySQLcommand.Parameters.AddWithValue("@Third", txtValue.Text);

                    cmMySQLcommand.CommandText = strMyQuery;
                    cmMySQLcommand.ExecuteNonQuery();

                    DataTable table = new DataTable();
                    SqlDataAdapter dataAdapter = new SqlDataAdapter(cmMySQLcommand);
                    dataAdapter.Fill(table);
                    dgvSearchGrid.DataSource = table;
                    //dgvSearchGrid.DataMember = "tblCar";

                    if (dgvSearchGrid.Rows.Count == 0)
                    {
                        MessageBox.Show("Unable to find a match for your query, please
try again!");
                    }
                }
                //Catch an SQL specific crash and return appropriate feedback to the
user.
                catch (SqlException sqlEx)
                {
                    MessageBox.Show("Program encountered an error in your query!\n" +
"Original Error:\n" + sqlEx.Message);
                }
                //Catch a general crash and return appropriate feedback to the user.
                catch (Exception ex)
                {
                    MessageBox.Show("Program encountered an error.\n" + "Original
Error:\n" + ex.Message);
                }
            }
                //Error Feedback if user doesn't specify a search value.
            else
            {
                MessageBox.Show("You need to enter value text!");
            }
        }
        //Closes the form
        private void btnClose_Click(object sender, EventArgs e)
        {
            this.Close();
        }
        //Populates combo-boxes.
        private void fillBoxes()
        {
            cboField.Items.Add("Make");
            cboField.Items.Add("EngineSize");
            cboField.Items.Add("RentalPerDay");
            cboField.Items.Add("Available");
            cboField.SelectedIndex = 0;

            cboOperator.Items.Add("=");
            cboOperator.Items.Add("<");
            cboOperator.Items.Add(">");
            cboOperator.Items.Add("<=");
            cboOperator.Items.Add(">=");
            cboOperator.SelectedIndex = 0;
        }
    }
}
```