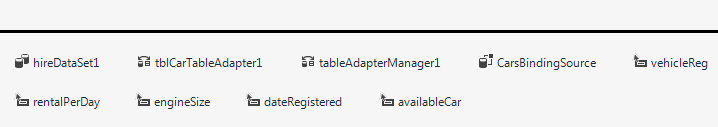
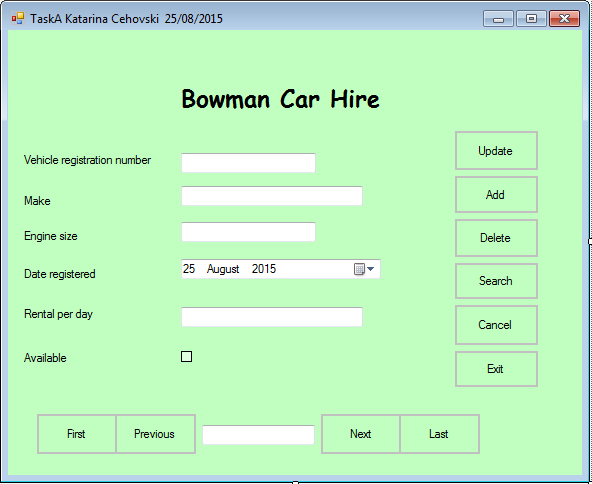
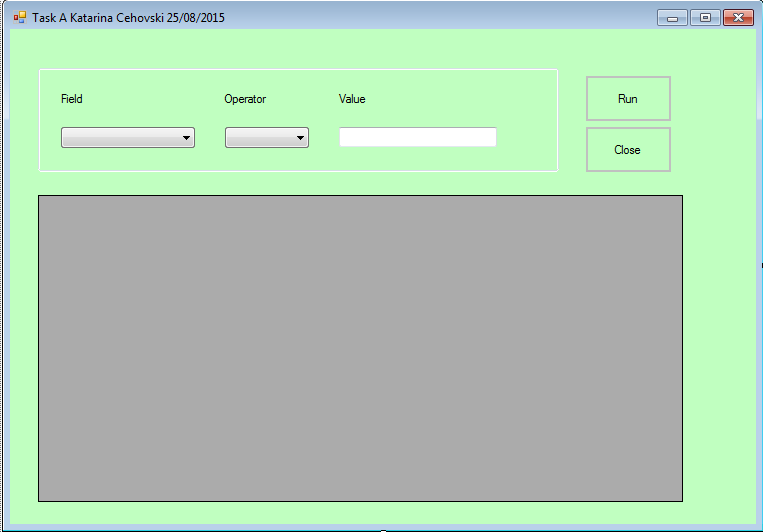
KATARINA CEHOVSKI  
ASSIGNMENT TASK B

PART 1: TEST PLAN AND TEST LOG  
  
**Type of Test:**  
Validation of Update button.  
**Expected Result:**  
When you click the Update button should assign any changes of a record to the table.  
**Actual Result:**  
It passes the test.  
  
**Type of Test:**  
Validation of Add button.  
**Expected Result:**  
When you click the Add button should add a new row/record.   
**Actual Result:**  
It passes the test.  
  
**Type of Test:**Validation of Delete button.  
**Expected Result:**When you click the Delete button should delete the record.   
**Actual Result:**  
It passes the test.  
  
**Type of Test:**Validation of Cancel button.  
**Expected Result:**  
When you click the Cancel button should disregard any changes made to a record.  
**Actual Result:**  
It passes the test.  
  
**Type of Test:**  
Validation of Search button.  
**Expected Result:**  
When you click the Search button should open the form frmSearch.  
**Actual Result:**It passes the test.  
  
**Type of Test:**Validation of Run button.  
**Expected Result:**  
When you click the Run button should execute query that match criteria entered using combo boxes and the value in the data entry text box. All fields from table tblCar for all the records which match the criteria should be displayed in the data grid. The search should be run only if data exists in all three controls. A criteria string that is not matched by any record should return nothing.  
Error messages should show when:  
a) there is no match,  
b) customer didn’t enter value in text box,  
c) there is SQL Error  
**Actual Result:**It passes the test.  
  
  
PART 2:   
  
  
  
hireDataSet1   
Datasets are objects that contain data tables where you can temporarily store the data for use in your application.  
  
tblCarTableAdapter1  
TableAdapters provide communication between your application and a database.  
  
tableAdapterManager1  
The TableAdapterManager is a component that provides the functionality to save data in related data tables.   
  
CarsBindingSource  
It supports the binding of control elements in a form, and can be seen as a link between a *data source* and a control element.   
  
Purpose of the software is to allow clients to access an external database. Client can:  
- display individual records  
- add a new record  
- delete a record  
- edit a record  
- update a record  
- cancel amendments for a record  
- search records

PART 3:



  
  
  
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.Globalization;

namespace CarsDatabase

{

public partial class frmCars : Form

{

public frmCars()

{

InitializeComponent();

}

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

{

tblCarTableAdapter1.Fill(hireDataSet1.tblCar);

updatePosition();

}

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

{

try

{

if (RegTextBox.Text != "")

{

this.Validate();

this.CarsBindingSource.EndEdit();

this.tblCarTableAdapter1.Update(this.hireDataSet1.tblCar);

MessageBox.Show("Update successful");

}

else

{

MessageBox.Show("Vehicle Registration Number can't be empty");

}

}

catch (ConstraintException)

{

MessageBox.Show("That Vehicle Registration Number already exists");

}

}

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

{

CarsBindingSource.MoveNext();

updatePosition();

}

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

{

CarsBindingSource.MovePrevious();

updatePosition();

}

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

{

CarsBindingSource.MoveFirst();

updatePosition();

}

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

{

CarsBindingSource.MoveLast();

updatePosition();

}

private void updatePosition()

{

pageTextBox.Text = CarsBindingSource.Position +1 + " of " + CarsBindingSource.Count;

}

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

{

this.Close();

}

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

{

frmSearch fr = new frmSearch();

fr.ShowDialog();

}

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

{

try

{

DataRow row = hireDataSet1.tblCar.NewRow();

row["VehicleRegNo"] = "REG";

row["Available "] = 0;

hireDataSet1.tblCar.Rows.Add(row);

CarsBindingSource.MoveLast();

updatePosition();

}

catch (ConstraintException)

{

MessageBox.Show("Vehicle Registration Number REG already exist");

}

}

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

{

hireDataSet1.tblCar[CarsBindingSource.Position].Delete();

updatePosition();

}

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

{

hireDataSet1.RejectChanges();

CarsBindingSource.ResetBindings(false);

updatePosition();

}

}

}

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)

{

fieldComboBox.Items.Add("Make");

fieldComboBox.Items.Add("EngineSize");

fieldComboBox.Items.Add("RentalPerDay");

fieldComboBox.Items.Add("Available");

fieldComboBox.SelectedIndex = 0;

operatorComboBox.Items.Add("=");

operatorComboBox.Items.Add("<");

operatorComboBox.Items.Add(">");

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

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

operatorComboBox.SelectedIndex = 0;

}

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

{

this.Close();

}

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

{

if (valueTextBox.Text != "")

{

try

{

string sql = String.Format("SELECT VehicleRegNo, Make, EngineSize, DateRegistered, '€' + CAST(RentalPerDay AS varchar) AS RentalPerDay, Available FROM tblCar WHERE {0} {1} @Third", fieldComboBox.SelectedItem, operatorComboBox.SelectedItem);

SqlConnection connection = new SqlConnection(@"Data Source=(LocalDB)\v11.0;AttachDbFilename=|DataDirectory|\Hire.mdf;Integrated Security=True");

connection.Open();

SqlCommand command = connection.CreateCommand();

command.CommandType = CommandType.Text;

command.Parameters.AddWithValue("@Third", valueTextBox.Text);

command.CommandText = sql;

command.ExecuteNonQuery();

DataTable table = new DataTable();

SqlDataAdapter dataAdapter = new SqlDataAdapter(command);

dataAdapter.Fill(table);

tblCarDataGridView.DataSource = table;

if (tblCarDataGridView.Rows.Count == 0)

{

MessageBox.Show("There is no match!");

}

}

catch (SqlException ex)

{

MessageBox.Show("Error in your query!");

}

catch (Exception)

{

MessageBox.Show("Somthing is wrong, try again");

}

}

else

{

MessageBox.Show("You need to enter value text!");

}

}

}

}