CPSC 501 – Assignment #4

Student Database Application

Follow the instructions in lecture 6 (available in the kiwi site for CPSC 501) to create the StudentDB database with three tables as shown below. The database name should contain your student id e.g., StudentDB_1234567.

Students table has the StudentId as the primary key.

ALPHA2\SQLEXPRES...DB - dbo.Students | ALPHA2\SQLEXPRES...sDB - dbo.C

Column Name	Data Type	Allow Nulls
StudentId	int	☐
FirstName	varchar(50)	☐
LastName	varchar(50)	☐
Address	varchar(100)	☐
City	varchar(50)	☐
State	varchar(20)	☑
TelePhone	varchar(20)	☑

Courses table has the CourseNum as the primary key.

ALPHA2\SQLEXPRES...DB - dbo.Students | ALPHA2\SQLEXPRES...sDB - dbo.Courses

Column Name	Data Type	Allow Nulls
CourseNum	varchar(20)	☐
CourseName	varchar(50)	☐
Description	text	☑
Credits	int	☐

Enrollments table has both the Studentid and the CourseNum as the primary keys.

ALPHA2\SQLEXPRES...- dbo.Enrollments | ALPHA2\SQLEXPRES...DB - dbo.Studen

Column Name	Data Type	Allow Nulls
StudentId	int	☐
CourseNum	varchar(20)	☐

Establish the primary-foreign key relationships as demonstrated in lecture 5 for the above tables so that referential integrity is maintained in the database.

Populate some data in the three tables as shown in lecture 6.

ALPHA2\SQLEXPRES...DB - dbo.Students | ALPHA2\SQLEXPRES...- dbo.Enrollments

StudentId	FirstName	LastName	Address	City	State	TelePhone
123100	Bill	Baker	100 Main Street	Bridgeport	CT	555-1213
123101	Sally	Simpson	50 Pine Street	Milford	CT	557-1214

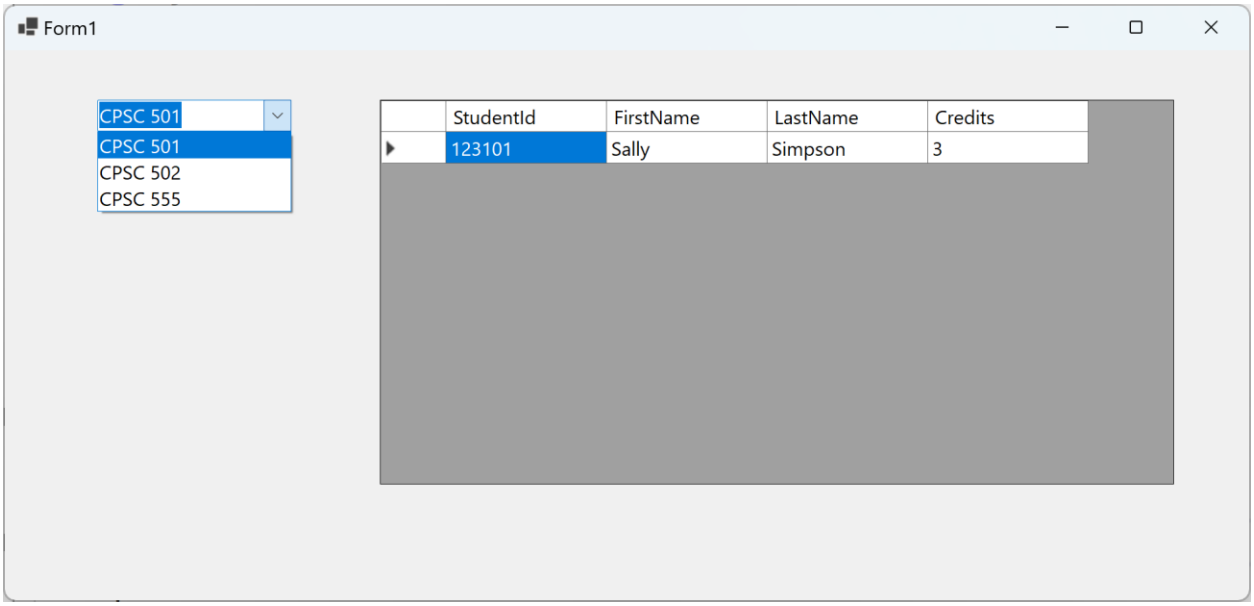
ALPHA2\SQLEXPRES...sDB - dbo.Courses | ALPHA2\SQLEXPR

CourseNum	CourseName	Descripti...	Credits
CPSC 501	OOP with D...	*NULL*	3
CPSC 502	Algorithms	*NULL*	3
CPSC 555	Web App D...	*NULL*	3

ALPHA2\SQLEXPRES...- dbo.Enrollments

StudentId	CourseNum
123100	CPSC 555
123101	CPSC 501
123101	CPSC 502

Create a windows forms application called DBStudentApp_yourId as described in lectures 7 and 8 to display the student enrollment for a course as shown below.



Code for the above application has been described in lectures 7 and 8.

Here is a summary of the steps to create this project.

Here is a summary of the steps to code the application.

From tools menu, choose Nuget Package Manager->Manage Nuget Packages for Solution, , then install the package System.Data.SqlClient.

Add the configuration file to the project called App.config. Then ddd the connection string to the appSettings.json file as shown below (you will need to replace ALPHA2\SQLEXPRESS with name of your database server)

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
      <connectionStrings>
            <add name="STUDENTDBCONN"
connectionString="server=ALPHA2\SQLEXPRESS;integrated
security=true;database=StudentsDB"/>
      </connectionStrings>
</configuration>
```

Add a folder to the project called Models. Then add a class called Course to it with the following code in it.

```
internal class Course
{
    public string CourseNum { get; set; }
    public string CourseName { get;set; }
    public string Description { get; set; }
    public int Credits { get;set; }
}
```

Add a class called Enrollment to the Models folder with the following code in it.

```
public class Enrollment
{
    public int StudentId { get; set; }
    public string FirstName { get; set;}
    public string LastName { get; set;}
    public int Credits { get; set;}
}
```

Add a folder to the project called DataLayer. Then add an interface called IDataAccess to it with the following code in it.

```
internal interface IDataAccess  // Facade for talking to DB
{
    object GetSingleAnswer(string sql);
    DataTable GetManyRowsCols(string sql);
    int InsertUpdateDelete(string sql);
}
```

Add a class called DataAccess to the DataLayer folder with the following code in it.

```
using System;
using System.Collections.Generic;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;
using System.Linq;
using System.Text;
using System.Threading.Tasks;


namespace DBStudentApp.DataLayer
{
    internal class DataAccess : IDataAccess
    {
        string connstr =
ConfigurationManager.ConnectionStrings["STUDENTDBCONN"].ConnectionString;
        public DataTable GetManyRowsCols(string sql)
        {
            DataTable dt= new DataTable();
            SqlConnection conn = new SqlConnection(connstr);
            try
            {
                conn.Open();
                SqlDataAdapter da = new SqlDataAdapter(sql, conn);
                da.Fill(dt);
            }
            catch (Exception ex)
            {
                throw;  // send error back to caller
            }
            finally { conn.Close(); }
            return dt;
        }

        public object GetSingleAnswer(string sql)
        {
            object obj = null;
            SqlConnection conn = new SqlConnection(connstr);
            try
            {
                conn.Open();
                SqlCommand cmd = new SqlCommand(sql, conn);
                obj = cmd.ExecuteScalar();
            }
            catch(Exception ex)
            {
                throw;  // send error back to caller
            }
            finally { conn.Close(); }
            return obj;
        }

        public int InsertUpdateDelete(string sql)
```

```
        {
            int rowsModified = 0;
            SqlConnection conn = new SqlConnection(connstr);
            try
            {
                conn.Open();
                SqlCommand cmd = new SqlCommand(sql, conn);
                rowsModified = cmd.ExecuteNonQuery();
            }
            catch (Exception ex)
            {
                throw;  // send error back to caller
            }
            finally { conn.Close(); }
            return rowsModified;
        }
    }
}
```

Add an interface called IRepository to the DataLayer folder with the following code in it.

```
    internal interface IRepository
    {
        List<Course> GetAllCourses();
        List<Enrollment> GetEnrollment(string courseNum);
    }
```

Add a class called Repository to the DataLayer folder with the following code in it.

```
using DBStudentApp.Models;
using System;
using System.Collections.Generic;
using System.Data;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace DBStudentApp.DataLayer
{
    internal class Repository : IRepository // compose SQL and send it to
DataLayer
    {
        IDataAccess _idac = new DataAccess();

        public List<Course> GetAllCourses()
        {
            List<Course> CList = new List<Course>();
            try
            {
                string sql = "select * from Courses";
                DataTable dt = _idac.GetManyRowsCols(sql);
                foreach (DataRow dr in dt.Rows)
```

```
                {
                    Course c = new Course();
                    c.CourseNum = (string)dr["CourseNum"];
                    c.CourseName = (string)dr["CourseName"];
                    CList.Add(c);
                }
            }
            catch (Exception ex) { throw; }
            return CList;
        }

        public List<Enrollment> GetEnrollment(string courseNum)
        {
            List<Enrollment> EList = new List<Enrollment>();
            try
            {
                string sql = "select
s.StudentId,s.FirstName,s.LastName,c.Credits " +
                    "from Students s " +
                    "join enrollments e on s.StudentId=e.StudentId " +
                    "join courses c on e.coursenum=c.coursenum " +
                    "where c.CourseNum='" + courseNum + "'";
                DataTable dt = _idac.GetManyRowsCols(sql);
                // convert dt to List<Enrollment>
                foreach (DataRow dr in dt.Rows)
                {
                    Enrollment e = new Enrollment();
                    e.StudentId = (int) dr["StudentId"];
                    e.FirstName = (string) dr["FirstName"];
                    e.LastName = (string)dr["LastName"];
                    e.Credits = (int) dr["Credits"];
                    EList.Add(e);
                }
            }
            catch (Exception ex) { throw; }
            return EList;
        }
    }
}
```

Modify the Form1 to have a combobox and a datagridview as shown below. You will drag and drop these controls from the toolbox.

Name the combobox as cmbCourses and the datagridview as dg1.

Double click on the combobox to write the selected index changed event handler for it. Also double click anywhere on the above form where there is no control to write the form1_load event. The final code for the form1.cs appears as:

```
using DBStudentApp.DataLayer;
using DBStudentApp.Models;

namespace DBStudentApp
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            IRepository irep = new Repository();
            List<Course> CList = irep.GetAllCourses();
            cmbCourses.DataSource= CList;
            cmbCourses.DisplayMember = "CourseNum";
            cmbCourses.ValueMember= "CourseNum";
            cmbCourses.Refresh();
        }

        private void cmbCourses_SelectedIndexChanged(object sender, EventArgs e)
        {
            string courseNum = cmbCourses.SelectedValue.ToString();
            IRepository irep = new Repository();
            var EList = irep.GetEnrollment(courseNum);
            dg1.DataSource = EList;
            dg1.Refresh();

        }
    }
}
```

Build and test the application.