



## TH Lap trinh tren moi truong Window 2023

Thực hành lập trình trên môi trường Windows (Trường Đại học Bách khoa - Đại học Quốc gia Thành phố Hồ Chí Minh)



Scan to open on Studocu

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**  
**ĐẠI HỌC CÔNG NGHỆ TP.HCM**



# **THỰC HÀNH LẬP TRÌNH TRÊN MÔI TRƯỜNG WINDOWS**

**Biên soạn:**

ThS. Nguyễn Đình Ánh  
ThS. Dương Thành Phết  
ThS. Trịnh Đồng Thạch Trúc  
ThS. Nguyễn Huy Cường

**THỰC HÀNH LẬP TRÌNH TRÊN MÔI TRƯỜNG WINDOWS**

Ấn bản 2023

# MỤC LỤC

MỤC LỤC.....	I
HƯỚNG DẪN .....	II
BÀI 1: ỨNG DỤNG CONSOLE NGÔN NGỮ C# .....	1
1.1 MỤC TIÊU.....	1
1.2 BÀI TẬP .....	2
BÀI 2: LẬP TRÌNH WINDOWS FORM VỚI CONTROLS CƠ BẢN .....	11
2.1 MỤC TIÊU.....	11
2.2 HƯỚNG DẪN LÀM QUEN VỚI WINDOWS FORM .....	12
2.3 BÀI TẬP .....	18
BÀI 3: LẬP TRÌNH WINDOWS FORM VỚI GIAO DIỆN MDI.....	28
3.1 MỤC TIÊU.....	28
3.2 BÀI TẬP .....	28
BÀI 4: LẬP TRÌNH VỚI CƠ SỞ DỮ LIỆU SỬ DỤNG ENTITY FRAMEWORK.....	38
4.1 MỤC TIÊU.....	38
4.2 BÀI TẬP .....	38
BÀI 5: TỔ CHỨC DỰ ÁN VỚI ENTITY FRAMEWORK .....	55
5.1 MỤC TIÊU.....	55
5.2 BÀI TẬP .....	55
BÀI 6: ÔN TẬP VÀ KIỂM TRA.....	67

# HƯỚNG DẪN

## MÔ TẢ MÔN HỌC

Lập trình C# trên Windows là một trong những môn học nhằm cung cấp kiến thức cơ bản cho những ai muốn xây dựng được ứng dụng trên môi trường trên hệ điều hành Windows. Môn học này cung cấp những kiến thức cơ bản, tổng quan về công nghệ mới của Microsoft: .Net Framework, lập trình hướng đối tượng trên .Net, cũng như giới thiệu bộ thư viện .Net mà cung cấp sẵn (LINQ, Entity framework...), sinh viên có thể tạo giao diện ứng dụng có kết nối các hệ cơ sở dữ liệu (SQL Server). Từ đó sinh viên có thể bước đầu có thể xây dựng những ứng dụng tin học hóa, hỗ trợ các doanh nghiệp trong việc quản lý hàng hóa, nhân sự, mua bán, v.v.

## NỘI DUNG THỰC HÀNH

- Bài 1. Tổng quan ngôn ngữ C# trên ứng dụng Console: Bài này cung cấp cho học viên làm quen với ngôn ngữ C#. Với phương pháp lập trình hướng đối tượng, sử dụng thư viện LINQ để giúp truy vấn nhanh chóng.
- Bài 2. Lập trình với ứng dụng giao diện Windows Form với các control cơ bản. Học viên được thiết kế giao diện, làm quen với các controls như: TextBox, Button, ComboBox, RadioButton, DataGridView...Bên cạnh đó việc viết code cho các event trên giao diện người dùng.
- Bài 3. Lập trình với giao diện MDI: Giúp học viên tìm hiểu thêm 1 số controls nâng cao trong Windows Form như MenuStrip, ToolStrip, Timer...
- Bài 4. Sử dụng EntityFrameWork để kết nối với cơ sở dữ liệu SQL Server: Học viên được hướng dẫn cách sử dụng mô hình Code First trên EntityFrameWork với hướng tiếp cận có sẵn cơ sở dữ liệu.
- Bài 5. Tổ chức dự án: Hướng dẫn cách tạo dự án theo mô hình 3 lớp từ đó cung cấp sự phân chia rõ ràng giữa các thành phần trong ứng dụng, tạo điều kiện thuận lợi cho việc bảo trì, mở rộng và quản lý dự án. Nó giúp tăng tính linh hoạt, đồng nhất và tái sử dụng code, cũng như hỗ trợ việc phát triển đa nền tảng.

- Bài 6: Bài kiểm tra kết thúc thực hành giúp đánh giá quá trình học tập của học viên.

## **KIẾN THỨC TIỀN ĐỀ**

Môn học Lập trình C# trên Windows đòi hỏi sinh viên có nền tảng về Lập trình C, và lập trình hướng đối tượng.

## **YÊU CẦU MÔN HỌC**

Người học phải dự học đầy đủ các buổi lên lớp và làm bài tập đầy đủ ở nhà.

## **CÁCH TIẾP NHẬN NỘI DUNG MÔN HỌC**

Để học tốt môn này, người học cần ôn tập các bài đã học, trả lời các câu hỏi và làm đầy đủ bài tập; đọc trước bài mới và tìm thêm các thông tin liên quan đến bài học.

Đối với mỗi bài học, người học đọc trước mục tiêu và tóm tắt bài học, sau đó đọc nội dung bài học. Kết thúc mỗi ý của bài học, người đọc trả lời câu hỏi ôn tập và kết thúc toàn bộ bài học, người đọc làm các bài tập.

## **PHƯƠNG PHÁP ĐÁNH GIÁ THỰC HÀNH**

Môn học được đánh giá:

- Điểm thực hành: Điểm trung bình các bài thực hành hàng tuần
- Hình thức tham gia và làm bài tập trên các buổi thực hành trên lớp/ ở nhà theo yêu cầu của giáo viên.



# BÀI 1: ỨNG DỤNG CONSOLE NGÔN NGỮ C#

## 1.1 MỤC TIÊU

- Hướng dẫn sinh viên làm quen với ngôn ngữ lập trình C#: qua việc viết các ứng dụng
- Console trong Visual Studio .NET 2022.
- Xây dựng các lớp, tạo đối tượng, truy xuất các phương thức, ...
- Soạn thảo mã nguồn, biên dịch, debug, thực thi chương trình...
- Kế thừa trong lập trình hướng đối tượng trên C#.
- Tìm hiểu về sử dụng thư viện LINQ của .NET
- Khuyến khích sinh viên thực hiện đúng chuẩn coding C# ( C# Coding Convention)

- Kiểu dữ liệu: `bool`, `decimal`, `double`, `float`, `int`, `string`, `DateTime`, `bool?`, `decimal?`, `double?`, `float?`, `int?`, `long?`, `DateTime?`, `object`, `var`, `dynamic`...
- Chuyển đổi/ép kiểu dữ liệu: `as`, `is`, Thư viện `Convert`
- Vòng lặp: `for`, `foreach`, `while`, `do...while` và Lệnh điều khiển `break`, `continue`, `return`
- Một số phương thức của thư viện **Console**

```
Console.Write("giá trị cần xuất ra màn hình");
```

```
Console.WriteLine(); // Xuất ra màn hình có xuống dòng
```

```
Console.ReadLine(); // Đọc chuỗi từ bàn phím cho đến khi gặp ký tự xuống dòng
```

```
Console.ReadKey(); // Dừng màn hình để xem kết quả.
```



- Coding Convention C#: Đưa ra các quy ước khi coding với ngôn ngữ lập trình C#, với các quy tắc này giúp tiết kiệm thời gian rất trong quá trình phát triển và bảo trì phần mềm.

Ví dụ: Đặt tên class dùng danh từ, đối với phương thức dùng động từ ...

Tên biến, tên phương thức thể hiện được ý nghĩa

Nên comment những đoạn code khó hiểu hoặc có chức năng đặc biệt

## 1.2 BÀI TẬP

### Bài tập 1: Viết chương trình trò chơi đoán số được mô tả như sau:

- Máy tính sẽ phát sinh ngẫu nhiên một số có 3 chữ số từ 100 đến 999
- Người chơi sẽ đoán số này bằng cách nhập vào một số có 3 chữ số.
- Sau mỗi lần đoán, máy tính sẽ phản hồi dựa trên số đã đoán của người chơi:
  - ✓ Dấu '+' được sử dụng để chỉ ra rằng một chữ số trong số đoán của người chơi là chính xác và nằm ở đúng vị trí tương ứng.
  - ✓ Dấu '?' được sử dụng để chỉ ra rằng một chữ số trong số đoán của người chơi là chính xác, nhưng nằm ở một vị trí khác so với vị trí tương ứng trong số mà máy tính đã phát sinh.
  - ✓ Các vị trí còn lại sẽ không có phản hồi nào. Trường hợp có các số đoán có chữ số trùng nhau thì máy tính vẫn phản hồi ở tất cả các vị trí số đã đoán.
- Thông báo chiến thắng/thất bại sau 7 lần đoán tối đa ?

**Ví dụ các phản hồi của máy tính:** Máy tính đã phát sinh số 123

- Người chơi đoán: 111. Kết quả phản hồi: +??
- Người chơi đoán: 325. Kết quả phản hồi: ?+
- Người chơi đoán: 249. Kết quả phản hồi: ?
- Người chơi đoán: 490. Kết quả phản hồi:
- Người chơi đoán: 212. Kết quả phản hồi: ???

- Người chơi đoán: 212. Kết quả phản hồi: ???
- Người chơi đoán: 123. Kết quả phản hồi: +++

### Minh họa:

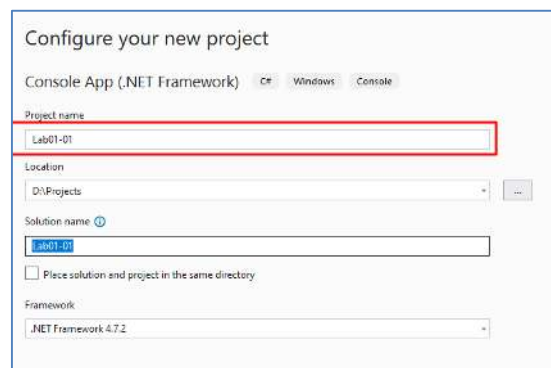
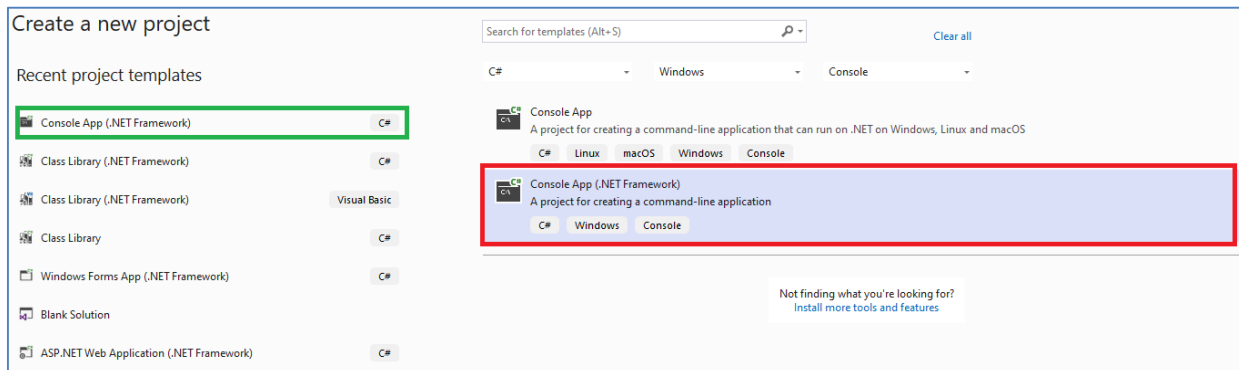
```
=== Chương trình đoán số ===
Lan đoán thử 1: 123
Phản hồi từ máy tính: ??
Lan đoán thử 2: 456
Phản hồi từ máy tính: ?
Lan đoán thử 3: 789
Phản hồi từ máy tính:
Lan đoán thử 4: 230
Phản hồi từ máy tính: +
Lan đoán thử 5: 215
Phản hồi từ máy tính: +?
Lan đoán thử 6: 261
Phản hồi từ máy tính: ++
Lan đoán thử 7: 266
Phản hồi từ máy tính: +
Người chơi đã đoán 7 lần. Trò chơi kết thúc!
Người chơi thua cuộc. Số cần đoán là: 241
```

```
=== Chương trình đoán số ===
Lan đoán thử 1: 123
Phản hồi từ máy tính: +
Lan đoán thử 2: 456
Phản hồi từ máy tính:
Lan đoán thử 3: 789
Phản hồi từ máy tính: ?
Lan đoán thử 4: 107
Phản hồi từ máy tính: +++
Người chơi đã đoán 4 lần. Trò chơi kết thúc!
Người chơi thắng cuộc!
```

### Hướng dẫn:

- Sử dụng một số biến như sau:
  - targetNumber**: Số máy tính đã phát sinh- **targetString**: Biểu diễn chuỗi số từ targetNumber.
  - attempt**: Số lần đoán của người chơi. MAX\_GUESS = 7 là số lần đoán tối đa;
  - guess**: Chuỗi số mà người chơi đoán.
  - feedback**: Chuỗi phản hồi từ máy tính dựa trên đoán của người chơi.
- Sử dụng một vòng lặp while khi chưa đoán đúng hoặc hết lượt đoán
  - Trong vòng lặp, yêu cầu người chơi nhập số đoán và gọi hàm **GetFeedback** để lấy phản hồi từ máy tính dựa trên số đoán và số cần đoán.
- Hàm **GetFeedback** lặp qua từng chữ số **targetString** và so sánh chúng với chữ số tương ứng trong số đoán **guess**. Trả về chuỗi feedback với các ký tự + và ? tương ứng với số đúng vị trí và số sai vị trí.
- Sau khi đoán đúng số hoặc vòng lặp kết thúc và hiển thị thông báo với số lần đoán để kết thúc trò chơi.

**Bước 1:** Tạo project mới, có template Console App (.NET Framework), ngôn ngữ C# với tên Lab01-01, trong Visual Studio 2022 /2019



**Bước 2:** Viết chương trình ở trong hàm Main, trong Program.cs

```

class Program
{
    0 references
    static void Main()
    {
        Console.WriteLine("=== Chương trình đoán số ===");

        Random random = new Random();
        int targetNumber = random.Next(100, 999);
        string targetString = targetNumber.ToString();

        int attempt = 1, MAX_GUESS = 7;
        string guess, feedback = "";
        while (feedback != "+++" && attempt <= MAX_GUESS)
        {
            Console.Write("Lan đoán thu {0}: ", attempt);
            guess = Console.ReadLine();
            feedback = GetFeedback(targetString, guess);
            Console.WriteLine("Phản hồi từ máy tính: {0}", feedback);
            attempt++;
        }
        Console.WriteLine("Người chơi đã đoán {0} Lan. Trò chơi kết thúc!", attempt-1);
        if (attempt > MAX_GUESS)
            Console.WriteLine("Người chơi thua cuộc. Số cần đoán là: {0}", targetNumber);
        else
            Console.WriteLine("Người chơi thắng cuộc!", attempt);
        Console.ReadLine();
    }

    1 reference
    static string GetFeedback(string target, string guess)
    {
        string feedback = "";
        for (int i = 0; i < target.Length; i++)
        {
            if (target[i] == guess[i])
                feedback += "+";
            else if (target.Contains(guess[i].ToString()))
                feedback += "?";
        }
        return feedback;
    }
}

```

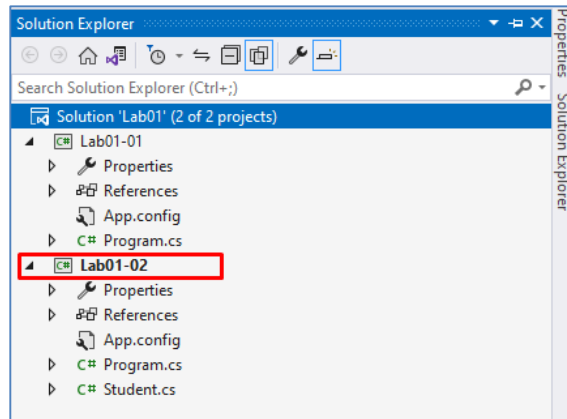
**Bài tập 2A:** Viết chương trình menu cho phép người dùng quản lý sinh viên. Biết rằng mỗi sinh viên có (mã số, họ tên, khoa và điểm trung bình).

Các chức năng menu:

- 1 – Thêm sinh viên
- 2 – Xuất danh sách sinh viên
- 0 - Thoát

## Hướng Dẫn

**Bước 1:** Thêm project mới Lab01-02 trong cùng solution ở bài tập 1. (Visual Studio cho phép có nhiều projects trong cùng 1 solution. Để Run project thì click phải chọn Set as Startup Project)



**Bước 2: Tạo lớp Student:** Khai báo thuộc tính, phương thức (Right Click vào Lab01-02 Project, chọn Add/ Class đặt tên là Student.cs). Các Property và Constructor nên sử dụng tools để sinh tự động (Quick actions and Refactorings...)

```
class Student
{
    //1. Field
    private string studentID;
    private string fullName;
    private float averageScore;
    private string faculty;

    //2. Property
    public string StudentID { get => studentID; set => studentID = value; }
    public string FullName { get => fullName; set => fullName = value; }
    public float AverageScore { get => averageScore; set => averageScore =
value; }
    public string Faculty { get => faculty; set => faculty = value; }

    //3. Constructor
    public Student()
    {
    }
    public Student(string studentID, string fullName, float averageScore,
string faculty)
    {
        this.studentID = studentID;
        this.fullName = fullName;
        this.averageScore = averageScore;
        this.faculty = faculty;
    }

    //4. Methods
    public void Input()
    {
        Console.WriteLine("Nhập MSSV:");
        StudentID = Console.ReadLine();
        Console.WriteLine("Nhập Họ tên Sinh viên:");
        FullName = Console.ReadLine();
        Console.WriteLine("Nhập Điểm TB:");
        AverageScore = float.Parse(Console.ReadLine()); //ép sang kiểu float

        Console.WriteLine("Nhập Khoa:");
    }
}
```

```

        Faculty = Console.ReadLine();
    }
    public void Show()
    {
        Console.WriteLine("MSSV:{0} Họ Tên:{1} Khoa:{2} ĐiểmTB:{3}",
this.StudentID, this.fullName, this.Faculty, this.AverageScore);
    }
}

```

**Bước 3:** Viết code trong hàm **Main()** để tạo menu

```

List<Student> studentList = new List<Student>();

bool exit = false;
while (!exit)
{
    Console.WriteLine("=== MENU ===");
    Console.WriteLine("1. Thêm sinh viên");
    Console.WriteLine("2. Hiển thị danh sách sinh viên");
    Console.WriteLine("0. Thoát");
    Console.Write("Chọn chức năng (0-2): ");

    string choice = Console.ReadLine();

    switch (choice)
    {
        case "1":
            AddStudent(studentList);
            break;
        case "2":
            DisplayStudentList(studentList);
            break;
        case "0":
            exit = true;
            Console.WriteLine("Kết thúc chương trình.");
            break;
        default:
            Console.WriteLine("Tùy chọn không hợp lệ. Vui lòng chọn lại.");
            break;
    }

    Console.WriteLine();
}

```

**Bước 4:** Thực hiện các hàm cho case 1 và case 2

```

static void AddStudent(List<Student> studentList)
{
    Console.WriteLine("=== Nhập thông tin sinh viên ===");
    Student student = new Student();
    student.Input();
    studentList.Add(student);
    Console.WriteLine("Thêm sinh viên thành công!");
}
static void DisplayStudentList(List<Student> studentList)
{
    Console.WriteLine("=== Danh sách chi tiết thông tin sinh viên ===");
}

```

```

        foreach (Student student in studentList)
        {
            student.Show();
        }
    }

```

**Bài tập 2B:** Tiếp tục bài tập 2 thực hiện các menu chức năng sau:

- 3 - Xuất ra thông tin của các SV đều thuộc khoa "CNTT"
- 4 - Xuất ra thông tin sinh viên có điểm TB lớn hơn bằng 5.
- 5 - Xuất ra danh sách sinh viên được sắp xếp theo điểm trung bình tăng dần
- 6 - Xuất ra danh sách sinh viên có điểm TB lớn hơn bằng 5 và thuộc khoa "CNTT"
- 7- Xuất ra danh sách sinh viên có điểm trung bình cao nhất và thuộc khoa "CNTT"
- 8- Hãy cho biết số lượng của từng xếp loại trong danh sách? Biết rằng theo thang điểm 10.  
 Từ 9,0 đến 10,0: Xuất sắc; Từ 8,0 đến cận 9,0: Giỏi; Từ 7,0 đến cận 8,0: Khá;  
 Từ 5,0 đến cận 7,0: Trung bình; Từ 4,0 đến cận 5,0: Yếu; Dưới 4,0: Kém.

*Sử dụng Cú pháp truy vấn (query syntax) hoặc cú pháp phương thức (method syntax) trong LINQ. Danh sách truy vấn trong LINQ*

Loại	Phương thức sử dụng
Lọc dữ liệu	Where
Chọn dữ liệu	Select, SelectMany
Phân vùng dữ liệu	Take, Skip, TakeWhile, SkipWhile
Kết hợp	Join, GroupJoin
Sắp Xếp	OrderBy / ThenBy, Reverse
Gom nhóm	GroupBy
Toán tử tập hợp	Distinct, Union, Intersect, Except
Chuyển đổi kiểu dữ liệu	ToSequence, ToArray, ToList, ToDictionary, ToLookup, OfType, Cast
Phần tử	First, FirstOrDefault, Last, LastOrDefault, Single, SingleOrDefault, ElementAt, ElementAtOrDefault, DefaultIfEmpty
Các hàm tính toán kết hợp	Count, LongCount, Sum, Min, Max, Average, Aggregate
Toán tử phát sinh	Range, Repeat, Empty
Toán tử lượng hóa	Any, All, Contains
Trộn	Concat

Bước 1: Thêm các case tương ứng cho bài tập

```

case "3":
    DisplayStudentsByFaculty(studentList, "CNTT");
    break;
case "4":
    DisplayStudentsWithHighAverageScore(studentList, 5);
    break;
case "5":
    SortStudentsByAverageScore(studentList);
    break;
case "6":
    DisplayStudentsByFacultyAndScore(studentList, "CNTT", 5);
    break;
case "7":
    DisplayStudentsWithHighestAverageScoreByFaculty(studentList, "CNTT");
    break;

```

Bước 2: Thực hiện từng case tương ứng

//case 3- DS Sinh viên khoa CNTT

```

static void DisplayStudentsByFaculty(List<Student> studentList, string faculty)
{
    Console.WriteLine("=== Danh sách sinh viên thuộc khoa {0}", faculty);
    var students = studentList.Where(s => s.Faculty.Equals(faculty,
        StringComparison.OrdinalIgnoreCase));
    DisplayStudentList(studentList);
}

```

//case 4: Xuất ra thông tin sinh viên có điểm TB lớn hơn bằng 5.

```

static void DisplayStudentsWithHighAverageScore(List<Student>
studentList, float minDTB)
{
    Console.WriteLine("=== Danh sách sinh viên có điểm TB >= {0}",
minDTB);
    var students = studentList.Where(s => s.AverageScore >= minDTB);
    DisplayStudentList(studentList);
}

```

//case 5: Xuất ra danh sách sinh viên được sắp xếp theo điểm trung bình tăng dần

```

static void SortStudentsByAverageScore(List<Student> studentList)
{
    Console.WriteLine("=== Danh sách sinh viên được sắp xếp theo điểm
trung bình tăng dần ===");
    var sortedStudents = studentList.OrderBy(s =>
s.AverageScore).ToList();
    DisplayStudentList(sortedStudents);
}

```

//case 6: DS sinh viên có DTB >=5 và thuộc khoa CNTT

```

static void DisplayStudentsByFacultyAndScore(List<Student> studentList,
string faculty, float minDTB)
{
    Console.WriteLine("=== Danh sách sinh viên có điểm TB >= {0} và thuộc
khoa {1}", minDTB, faculty);
    var students = studentList.Where(s => s.AverageScore >= minDTB
        && s.Faculty.Equals(faculty,
StringComparison.OrdinalIgnoreCase)).ToList();
    DisplayStudentList(students);
}

```



```
//case 7,8: SV tự thực hiện
```

**Bài tập 3:** Tạo thêm 1 project ở trong cùng solution Lab01 có tên là "Lab01-03".

- ✓ Viết lại chương trình từ bài tập trên theo cách tạo thêm một lớp là **Person** làm lớp cơ sở cho lớp **Student**. Chọn Mã số, Họ tên làm field để đưa lên lớp **Person**.
- ✓ Thêm thông tin lớp giảng viên **Teacher** kế thừa từ lớp **Person**. Mỗi giảng viên đều có Mã số, Họ Tên và Địa chỉ.
- ✓ Viết chương trình menu cho phép thực hiện các yêu cầu

1- Thêm sinh viên

2- Thêm giáo viên

3- Xuất danh sách sinh viên

4- Xuất danh sách giáo viên

5- Số lượng từng danh sách (tổng số sinh viên, tổng số giáo viên)

6- Xuất danh sách các Sinh Viên thuộc khoa "**CNTT**".

7- Xuất ra danh sách giáo viên có địa chỉ chứa thông tin "Quận 9"

8- Xuất ra danh sách sinh viên có điểm trung bình cao nhất và thuộc khoa "CNTT"

9- Hãy cho biết số lượng của từng xếp loại trong danh sách? Biết rằng theo thang điểm 10.

# BÀI 2: LẬP TRÌNH WINDOWS FORM VỚI CONTROLS CƠ BẢN

## 2.1 MỤC TIÊU

---

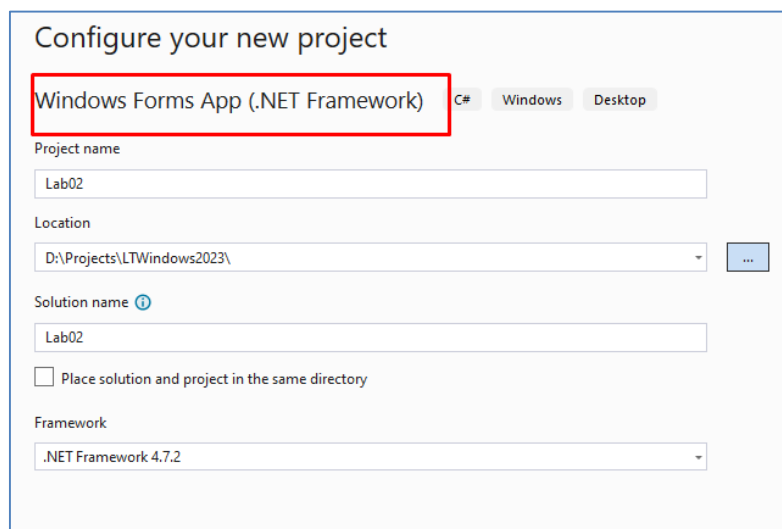
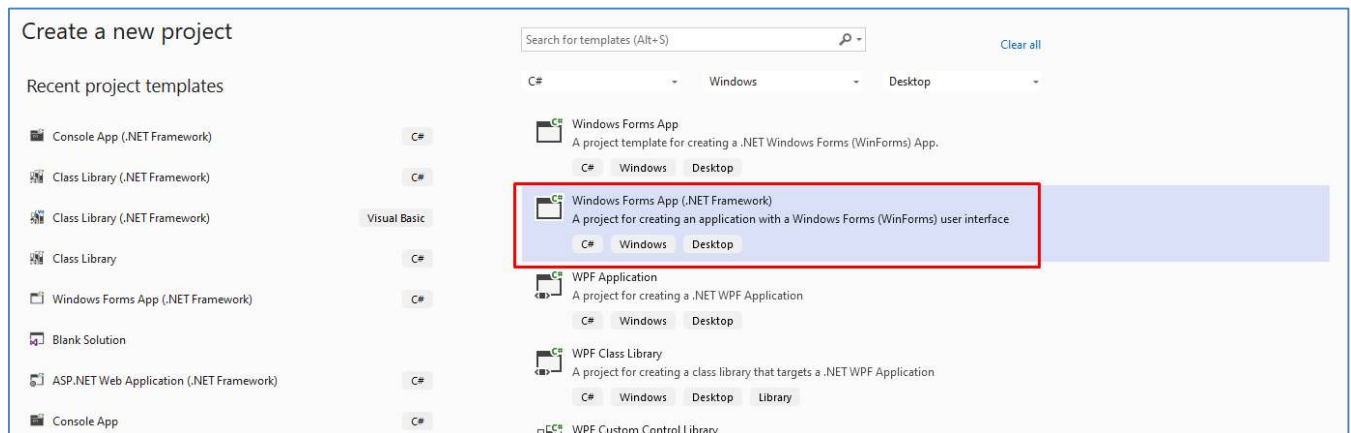
- Sử dụng Visual Studio .NET 2022, tạo ứng dụng dạng Windows Forms App (.NET Framework), ngôn ngữ C# .
- Làm quen với việc sử dụng các control thông dụng trên Forms như:
  - o Label: Hiển thị các thông tin chỉ dẫn
  - o TextBox: Hộp nhập liệu thông tin
  - o Button: Cho phép user click chọn để thực hiện chức năng
  - o CheckBox: Cho phép user chọn một hoặc nhiều option
  - o Radio button: Cho phép user chọn duy nhất một option
  - o MessageBox: Hiển thị thông tin đến user
  - o DataGridView: Hiển thị danh sách thông tin trên bảng
  - o ListView: Hiển thị một danh sách các item với các biểu tượng
  - o ComboBox: Hộp chọn 1 giá trị trong danh sách giá trị
  - o ListBox: Danh sách các mục chọn, cho phép chọn 1 hoặc nhiều mục
  - o GroupBox: Nhóm các đối tượng về cùng nhóm
  - o Panel: Nhóm các đối tượng vào cùng 1 khung
- Tìm hiểu các thuộc tính trên control (Visible, Enable, Name, Text ...) và các phương thức là Event (Click, Text\_Change, Text\_Press...).

- Binding dữ liệu vào controls (Combobox, DataGridView, ListView)

## 2.2 HƯỚNG DẪN LÀM QUEN VỚI WINDOWS FORM

- ✓ Tạo Project Application, Giao diện màn hình thiết kế form
  - Từ màn hình khởi động Microsoft Studio chọn Menu File - New - Project
  - Language : Visual C#
  - Loại ứng dụng: Windows Forms Application (.NET Framework)
  - Name: Tên Project – ví dụ Lab02

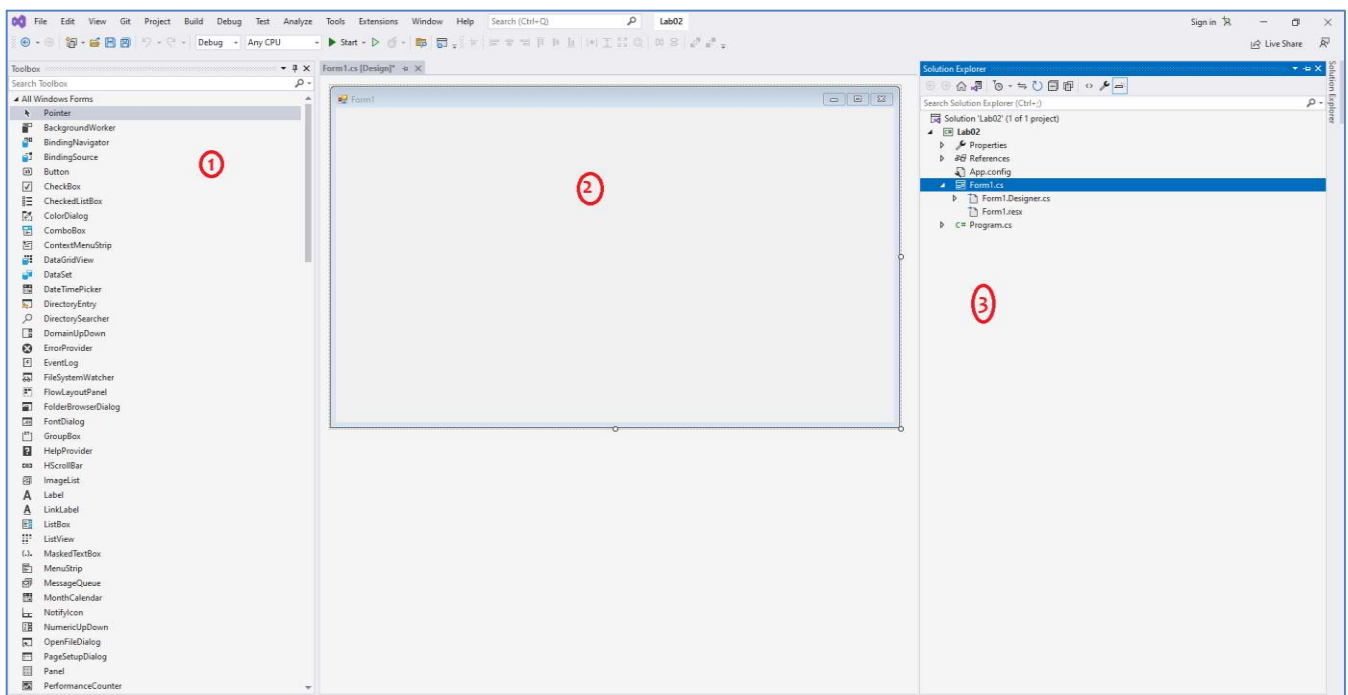
Location: Đường dẫn lưu Project



*Hình 2.1: Tạo mới project Windows Form Application*

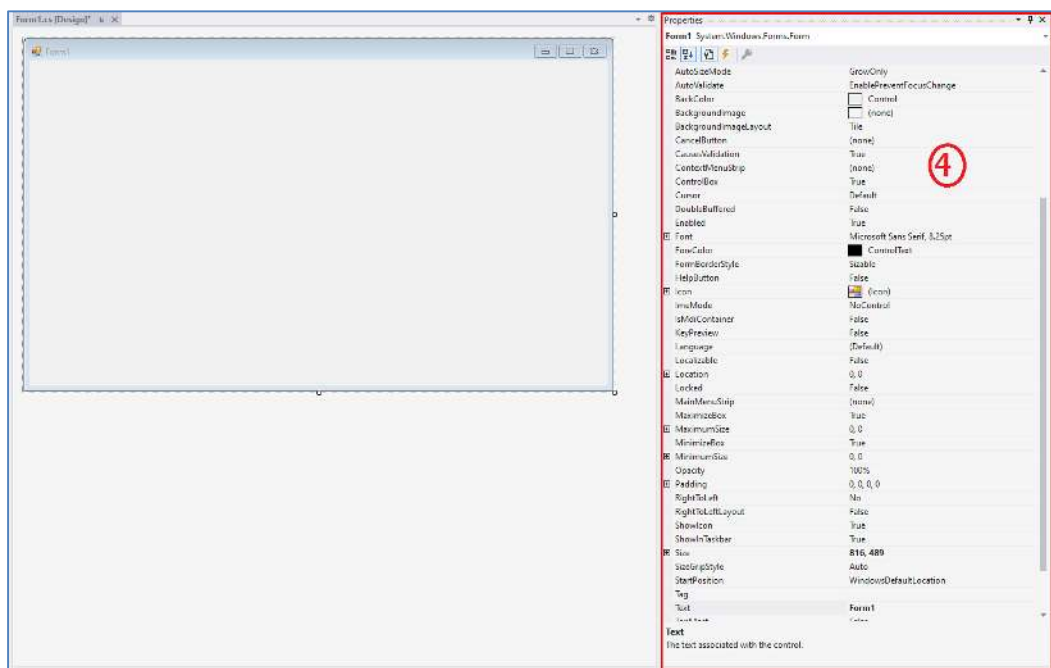
Kết quả màn hình VS.NET cho ứng dụng Windows Form bao gồm các phần cơ bản

- (1): Toolbox: Chứa các control cho phép kéo thả vào Form
- (2): Chứa thiết kế giao diện Form, có thể chuyển sang View Code...
- (3): Cửa sổ Solution Explorer: Cho phép người lập trình có thể quản lý các thành phần trong project, hỗ trợ định vị nhanh chóng đến các file mã nguồn.



*Hình 2: Màn hình VS. NET phục vụ cho việc tạo project Windows Form*

- (4): Cửa sổ property: cho phép user có thể custom lại các thành phần control trên form như:



- Thiết lập các thuộc tính (Property): Trong cửa sổ "Properties" (Thuộc tính), sẽ thấy danh sách các thuộc tính của điều khiển được chọn. 1 số thuộc tính thường hay gặp như:
  - + **Text**: Thuộc tính này cho phép thiết lập hoặc lấy văn bản hiển thị trên một Control. Ví dụ khi chọn Text của 1 Button là "Click me" tương đương ở code `button1.Text = "Click me";`
  - + **Name**: Đặt tên cho một Control. Nó được sử dụng để tham chiếu đến Control
  - + **BackColor**: Thiết lập màu nền cho một Control.
  - + **ForeColor**: Thiết lập màu chữ cho một Control.
  - + **Enabled**: Bật hoặc Tắt một Control. Khi Enabled là False, điều khiển sẽ bị vô hiệu hóa và không thể tương tác được.
  - + **Visible**: Ẩn hoặc hiển thị một điều khiển. Khi Visible là False, điều khiển sẽ được ẩn đi và không được hiển thị trên giao diện.
  - + **Size**: Thiết lập kích thước của một điều khiển.
  - + **Location**: Thiết lập vị trí của một điều khiển trên giao diện
- Khai báo đăng ký xử lý sự kiện: Các điều khiển (controls) có thể kích hoạt các sự kiện (events) khi tương tác xảy ra, chẳng hạn như khi người dùng

nhấp chuột, nhập liệu hoặc thay đổi trạng thái của điều khiển. 1 số event phổ biến

+ **Click**: Sự kiện Click xảy ra khi người dùng nhấp chuột vào một điều khiển, chẳng hạn như Button.

Ví dụ: `button1.Click += new EventHandler(button1_Click);`

+ **TextChanged**: Khi thay đổi giá trị Text

+ **SelectedIndexChanged**: Khi mục được chọn thay đổi

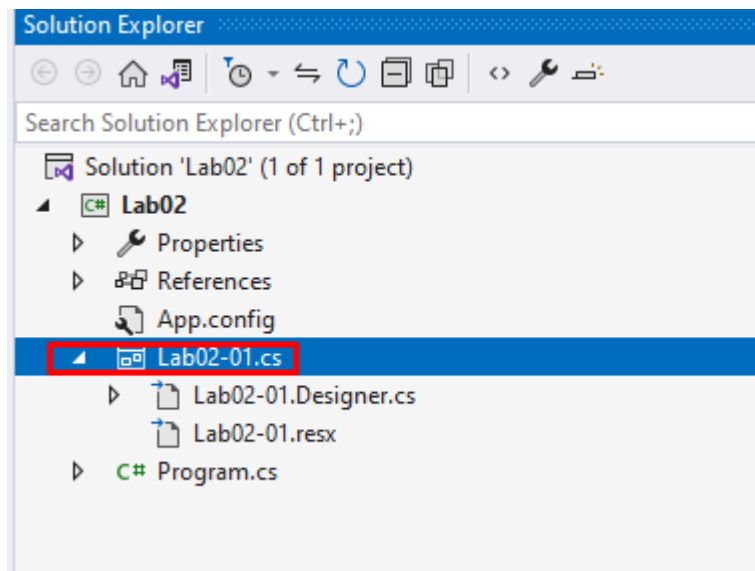
+ **CheckedChanged**: Khi trạng thái được kiểm tra (checked) của Control ( như CheckBox hoặc RadioButton thay đổi).

+ **MouseClick**: Nhấp chuột lên một điều khiển.

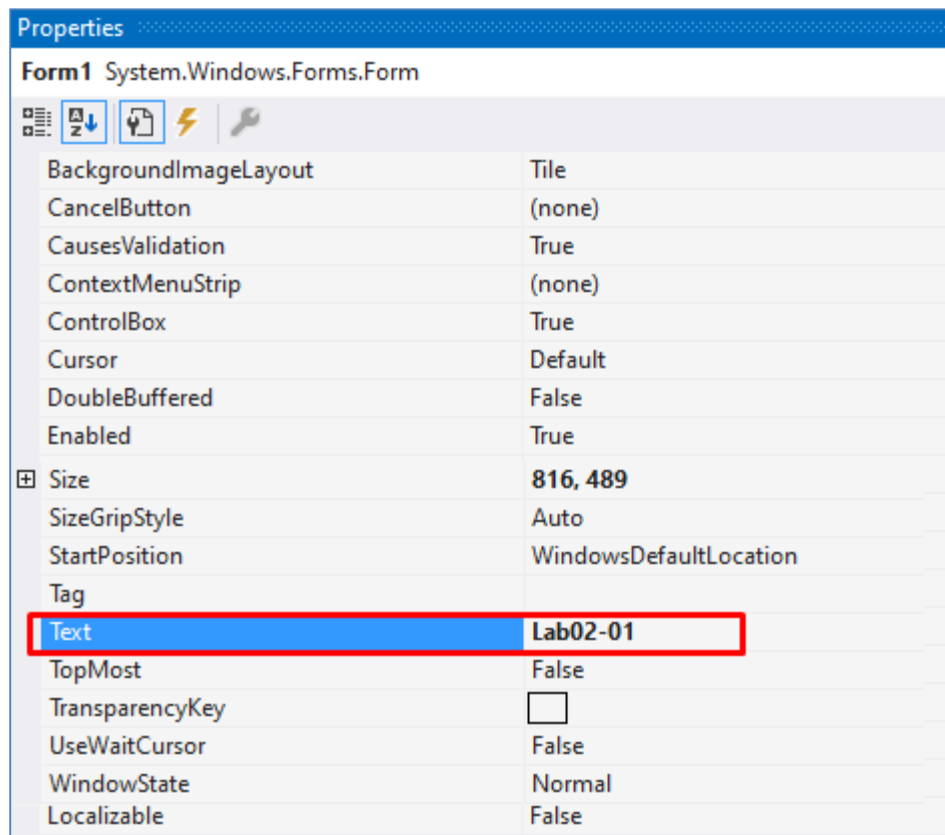
+ **KeyPress**: Nhấn một phím trong khi một Control (như TextBox) đang có trạng thái nhận nhập liệu.

+ **FormClosing**: Khi người dùng đóng một Form hoặc ứng dụng.

- Chú ý cửa sổ Toolbox chứa các công cụ để thiết kế: Nếu không thấy cửa sổ này, ta chọn menu View / **Toolbox**.
- Đổi tên form: Click lên Form1 ở cửa sổ Design, quan sát trên cửa sổ Properties, ta thấy có thuộc tính Text, giá trị mặc định là Form1, ta đổi thành Lab02-01

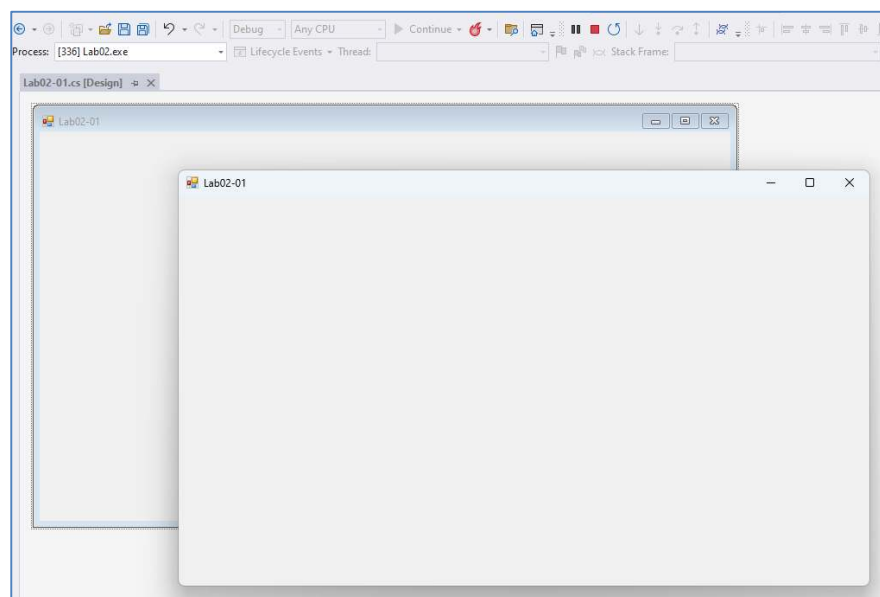


Thay đổi Property Text của Form



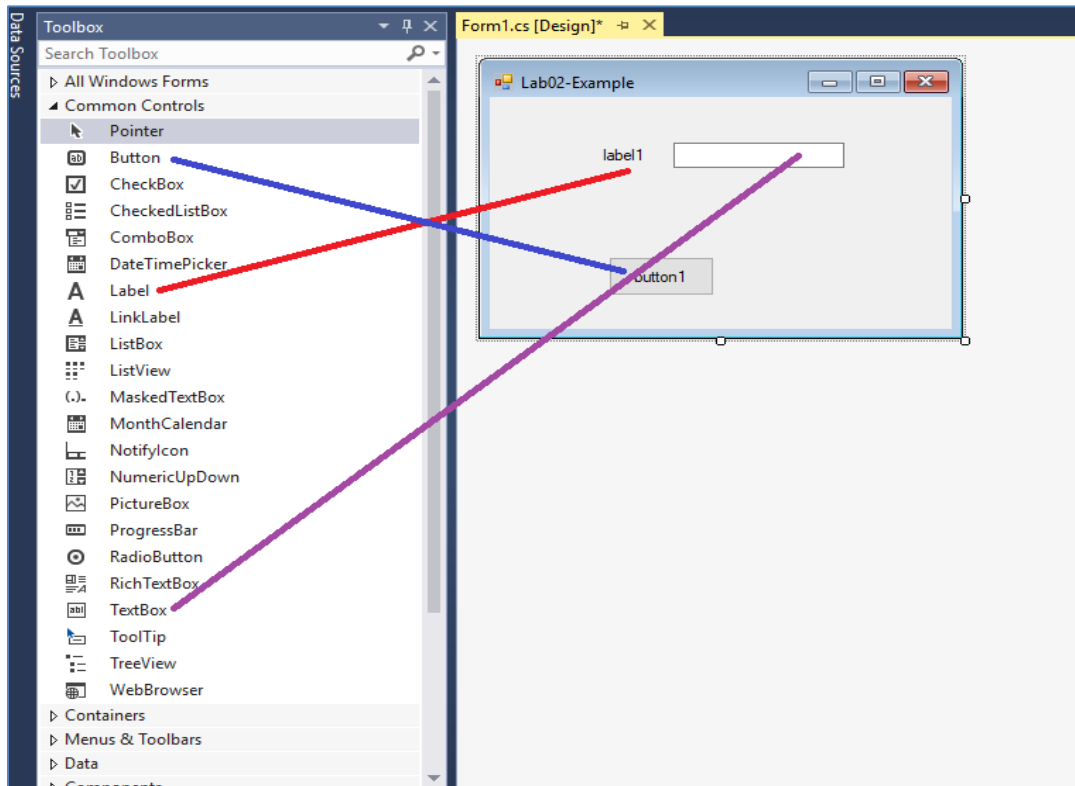
Hình 3: Property của đối tượng Form

- Chạy thử chương trình F5

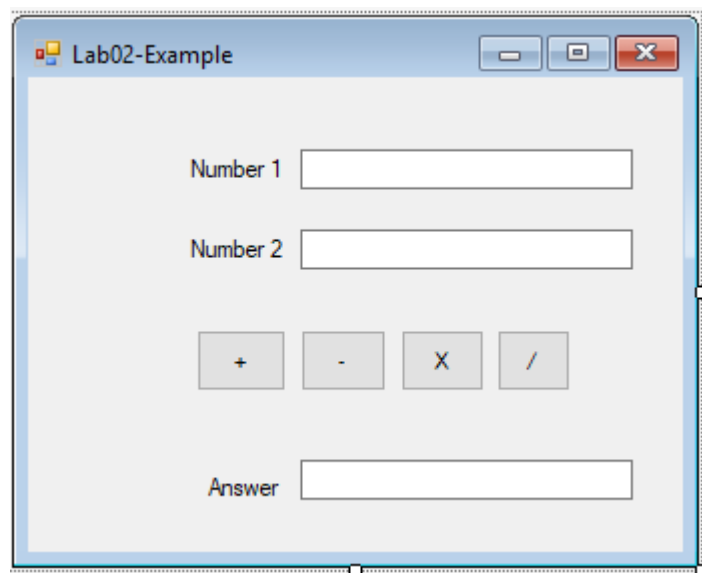


Hình 4: Chạy giao diện Forms đầu tiên

- Kéo thả các control từ **Toolbox** vào Forms: (chương trình tính toán + - \* / )



- Thiết kế lại Form như hình dưới đây



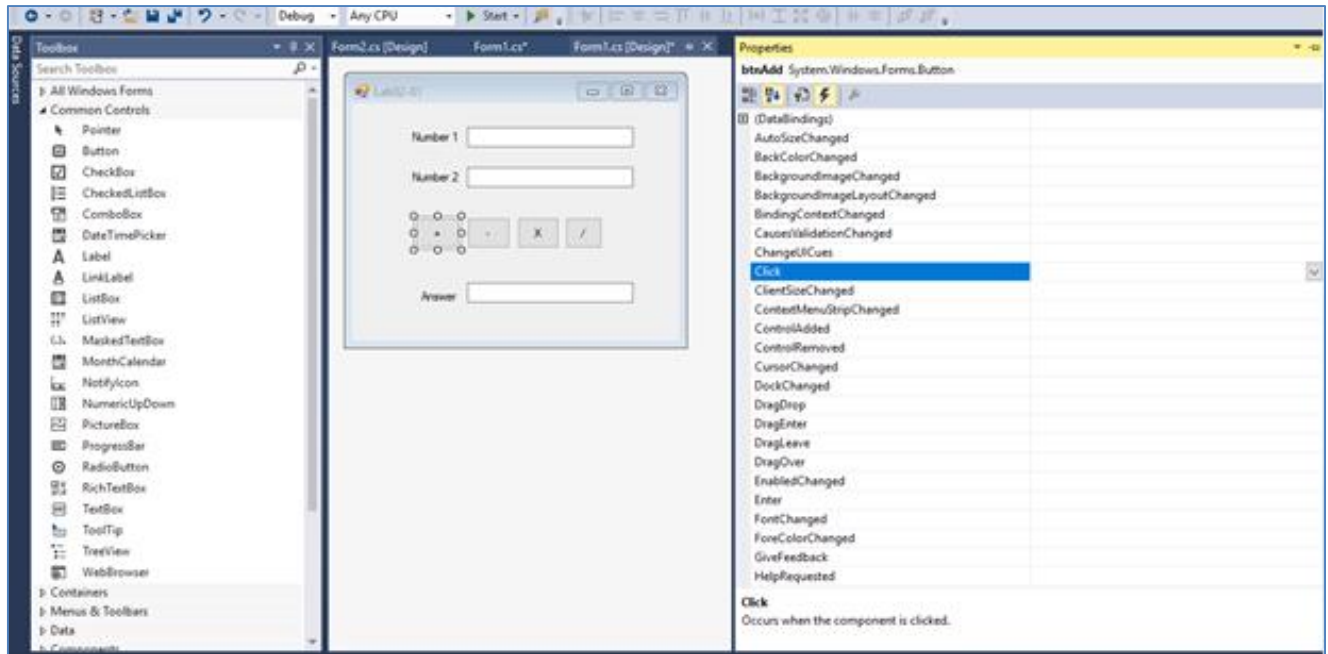
Hình 6: Giao diện sau khi thiết kế, chỉnh sửa Property

Đặt tên các **Controls** (thuộc tính **Name** trong **Property**) để dễ dàng quản lý. Đặt tên như sau:

- Các **Label** ở trong form có tên: lblNumber1, lblNumber2, lblAnswer



- Các **TextBox** ở trong form có tên: txtNumber1, txtNumber2, txtAnswer
- Các **Button** ở trong form có tên: btnAdd, btnSub, btnMul, btnDiv
- Xử lý sự kiện trên Control: Vào **Property** chọn biểu tượng **Events**



Hình 7: Các Events trong buttonAdd

- ✓ Chọn tên Event cần xử lý và double click vào đó.

Đối với các button để chọn nhanh sự kiện click ta có thể double click trực tiếp vào button đó.

Double click vào Button **btnAdd** để tiến hành viết code cho sự kiện Mouse click. VS .NET tự động sinh ra phương thức **btnAdd\_Click**. Sau đó tiến hành viết code tính tổng 2 số vừa nhập liệu

```
private void btnAdd_Click(object sender, EventArgs e)
{
    // Viết xử lý tính tổng trong sự kiện Add_click
    float number1 = float.Parse(txtNumber1.Text);
    float number2 = float.Parse(txtNumber2.Text);
    float result = number1 + number2;
    txtAnswer.Text = result.ToString();
}
```

## 2.3 BÀI TẬP

### Bài tập 1:

- ✓ Viết các sự kiện hoàn thành chương trình ở ví dụ trên ( các phép toán +, -, \*, /)

- ✓ Sử dụng **MessageBox** để đưa thông tin thêm cho người dùng khi gặp tất cả những lỗi ngoài mong muốn
- ✓ Quản lý lỗi bằng cách sử dụng: try... catch....finally

**Hướng Dẫn:** Sử dụng try...catch ở sự kiện, xử lý các sự kiện trong try

```
private void btnAdd_Click(object sender, EventArgs e)
{
    try
    {
        //Xử lí sự kiện cộng 2 số
    }
    catch(Exception ex) //Khi gặp bất kì lỗi nào sẽ vào catch
    {
        MessageBox.Show(ex.Message);
    }
}
```

**Bài tập 2:** Tạo thêm Project “**Lab02-02**” trong cùng Solution Lab02 (Sử dụng CheckBox, ComboBox, DataGridView)

Quản lý thông tin sinh viên cần lưu trữ các thông tin sau: Mã số sinh viên, Họ Tên, Giới Tính, Điểm Trung Bình và Tên Khoa. Thiết kế chương trình quản lý thông tin sinh viên tương tự như sau:

Có 3 khoa được đưa vào comboBox (QTKD, CNTT, NNA).

### 2.1 Khi mới Load Form

- Khoa được chọn mặc định là QTKD. Giới tính Nữ mặc định được checked. Tổng số sinh viên Nam/Nữ đều là 0.

### 2.2 Khi nhấn vào nút "Thêm/Sửa"

- Kiểm tra các thông tin bắt buộc phải nhập liệu cho sinh viên như mã sinh viên, tên, và điểm trung bình. Nếu để trống sẽ xuất hiện thông báo lỗi "**Vui lòng nhập đầy đủ thông tin!**".
- Nếu mã số sinh viên vừa nhập chưa có ở trong DataGridView (bên phải) thì Thêm mới dữ liệu sinh viên vừa nhập vào DataGridView, và thông báo "**Thêm mới dữ liệu thành công!**"

Nếu đã có MSSV trong DataGridView thì Cập nhật dữ liệu sinh viên vào DataGridView, và thông báo "**Cập nhật dữ liệu thành công!**"

### 2.3 Khi nhấn vào nút "Xóa"

- Kiểm tra nếu MSSV cần xóa không tồn tại trong DataGridView thì thông báo lỗi "**Không tìm thấy MSSV cần xóa!**".
- Ngược lại thì xuất hiện cảnh báo YES/NO. Nhấn YES sẽ thực hiện xóa dòng dữ liệu sinh viên trong DataGridView và thông báo "**Xóa sinh viên thành công!**".

2.4 Viết code cho sự kiện ở DataGridView, người dùng chọn 1 dòng thì thể hiện ngược lại thông tin của các sinh viên đã chọn ở phần nhập liệu (bên trái).

## 2.5 Tính toán lại số sinh viên Nam, Nữ phù hợp với các ngữ cảnh khi thay đổi dữ liệu

### **Hướng Dẫn**

- Thiết kế giao diện tương tự như trên. Nên đặt các tên các Control để nhớ như: txtStudentID, txtFullName, optMale, optFemale, txtAverageScore, cmbFaculty, dgvStudent, btnUpdate, btnDelete...
- Ở ComboBox Khoa, để sẵn giá trị vào bằng cách vào Properties / Items rồi nhập các dòng (QTKD, CNTT, NNA)
- Thiết kế DataGridView có 5 cột (Columns) như yêu cầu. Đặt các tên column để nhớ. Điều chỉnh lại kích thước width cho phù hợp. Properties có SelectionMode = FullRowSelect
- Ở các sự kiện event, nên sử dụng try...catch để bắt lỗi.
- Trong sự kiện Form\_Load (double click vào Form để sinh ra). Để chọn mặc định khoa ban đầu

```
private void frmQuanLySinhVien_Load(object sender, EventArgs e)
{
    cmbKhoa.SelectedIndex = 0;
}
```

- Viết sự kiện Thêm/ Sửa. Double click vào button để tạo event btnUpdate\_Click

```
private int GetSelectedRow(string studentID)
{
    for (int i = 0; i < dgvStudent.Rows.Count; i++)
    {
        if( dgvStudent.Rows[i].Cells[0].Value.ToString() == studentID)
        {
            return i;
        }
    }
    return -1;
}

private void InsertUpdate(int selectedRow)
{
    dgvStudent.Rows[selectedRow].Cells[0].Value = txtStudentID.Text;
    dgvStudent.Rows[selectedRow].Cells[1].Value = txtFullName.Text;
```

```

        dgvStudent.Rows[selectedRow].Cells[2].Value = optFemale.Checked ? "Nữ" :
        "Nam";

        dgvStudent.Rows[selectedRow].Cells[3].Value =
        float.Parse(txtAverageScore.Text).ToString();

        dgvStudent.Rows[selectedRow].Cells[4].Value = cmbFaculty.Text;
    }

    private void btnUpdate_Click(object sender, EventArgs e)
    {
        try
        {
            if (txtStudentID.Text == "" || txtFullName.Text == "" || txtAverageScore.Text == "")
                throw new Exception("Vui lòng nhập đầy đủ thông tin sinh viên!");

            int selectedRow = GetSelectedRow(txtStudentID.Text);
            if(selectedRow == -1)    //TH Thêm mới
            {
                selectedRow = dgvStudent.Rows.Add();
                InsertUpdate(selectedRow);
                MessageBox.Show("Thêm mới dữ liệu thành công!", "Thông Báo", MessageBoxButtons.OK);
            }
            else //TH cập nhật
            {
                InsertUpdate(selectedRow);
                MessageBox.Show("Cập nhật dữ liệu thành công!", "Thông Báo", MessageBoxButtons.OK);
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
}

```

- Viết sự kiện Xóa

```

private void btnDelete_Click(object sender, EventArgs e)
{
    try
    {
        int selectedRow = GetSelectedRow(txtStudentID.Text);
        if (selectedRow == -1)
        {
            throw new Exception("Không tìm thấy MSSV cần xóa!");
        }
        else
        {
            DialogResult dr = MessageBox.Show("Bạn có muốn xóa ?", "YES/NO", MessageBoxButtons.YesNo);
            if (dr == DialogResult.Yes)
            {
                dgvStudent.Rows.RemoveAt(selectedRow); //xóa thông tin sinh viên tại dòng tìm thấy
                MessageBox.Show("Xóa sinh viên thành công!", "Thông Báo", MessageBoxButtons.OK);
            }
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

Sinh viên tự thực hiện yêu cầu còn lại 2.4 và 2.5

### Bài tập 3: Tạo thêm Project "Lab02-03" trong cùng Solution Lab02

Chương trình bán vé rạp chiếu phim



Rạp có 4 hàng ghế, mỗi hàng có 5 ghế, các ghế được đánh số từ 1 đến 20 và được phân thành 4 dãy như (hình trên):

Dãy 1 (ghế 1-5), Giá vé 30000đ

Dãy 2 (ghế 6-10), Giá vé 40000đ

Dãy 3 (ghế 11-15), Giá vé 50000đ

Dãy 4 (ghế 16-20), Giá vé 80000đ

Trên Form trình bày một sơ đồ các chỗ ngồi để người sử dụng chọn vị trí muốn mua. Trên sơ đồ này cũng thể hiện những vị trí đã bán vé và những vị trí chưa bán vé bằng cách thể hiện màu khác nhau (ghế chưa bán vé màu trắng, ghế đã bán vé màu vàng).

Khi người sử dụng click chuột tại một vị trí trên sơ đồ thì:

- ✓ Nếu đây là vị trí chưa bán vé thì đổi màu của vị trí này sang màu xanh để cho biết đây là vị trí đang chọn.
- ✓ Nếu đây là vị trí đang chọn (có màu xanh) thì đổi màu của vị trí này trở về màu trắng
- ✓ Nếu đây là một vị trí đã bán vé (có màu vàng) thì xuất hiện một Message box thông báo cho người sử dụng biết vé ở vị trí này đã được bán. Sau khi đã chọn các vị trí người sử dụng có thể click chuột vào nút **CHỌN** hoặc **HỦY BỎ**

Nếu click vào nút **CHỌN** thì:

- Đổi màu các vị trí đã chọn (màu xanh) trên sơ đồ sang màu vàng (cho biết vị trí đã bán vé)
- Xuất lên một Label tổng số tiền phải trả cho số vé đã mua (phụ thuộc vào các vị trí đã chọn)

Nếu click vào nút **HỦY BỎ** thì:

- Đổi màu các vị trí đã chọn (màu xanh) trên sơ đồ sang màu trắng trở lại
- Xuất lên Label giá trị 0

**Hướng Dẫn:**

- Sử dụng GroupBox để gom nhóm (dễ dàng cho việc tìm kiếm, đổi màu, tính tiền...). Mặc định ban đầu các hàng ghế là trống BackColor= White;
- Nên Viết 1 sự kiện dùng chung cho tất cả các nút bấm (1 – 20)

```
private void btnChooseASeat(object sender, EventArgs e)
{
    Button btn = sender as Button;
    if (btn.BackColor == Color.White)
        btn.BackColor = Color.Blue;
    else if (btn.BackColor == Color.Blue)
        btn.BackColor = Color.White;
    else if (btn.BackColor == Color.Yellow)
        MessageBox.Show("Ghế đã được bán!!");
}
```

- Có thể tạo giao diện ban đầu bằng Code (thay vì từ kéo thả)

**Bài tập 4:** Tạo thêm Project “**Lab02-04**” trong cùng Solution Lab02 (Sử dụng ListView)

Quản lý thông tin tài khoản cần lưu trữ các thông tin sau: số tài khoản, tên khách hàng, địa chỉ khách hàng và số tiền trong tài khoản. Thiết kế chương trình quản lý thông tin tài khoản cho một ngân hàng tương tự như sau:



#### 4.1 Khi nhấn vào nút "Thêm/Cập Nhật"

- Kiểm tra các thông tin bắt buộc phải nhập liệu cho số tài khoản, tên, địa chỉ và số tiền. Xuất hiện thông báo lỗi "Vui lòng nhập đầy đủ thông tin!".
- Nếu chưa có dữ liệu **số tài khoản** trong ListView thì Thêm mới dữ liệu nhập vào ListView, tính lại tổng tiền và thông báo "Thêm mới dữ liệu thành công!"

Nếu đã tồn tại số tài khoản trong ListView thì Cập nhật dữ liệu vào ListView và tính lại tổng tiền và thông báo "Cập nhật dữ liệu thành công!"

#### 4.2 Khi nhấn vào nút "Xóa"

- Kiểm tra nếu số tài khoản cần xóa tồn tại trong ListView, thì xuất hiện cảnh báo YES/NO

Nhấn YES sẽ thực hiện xóa dòng dữ liệu tài khoản trong ListView và thông báo "Xóa tài khoản thành công!".

- Nếu số tài khoản cần xóa không tồn tại thì thông báo lỗi "Không tìm thấy số tài khoản cần xóa!".

4.3 Viết code cho sự kiện ở ListView khi người dùng chọn 1 dòng thì thể hiện ngược lại ở phần nhập liệu đúng thông tin trên.

Quản Lý Tài Khoản

## QUẢN LÝ THÔNG TIN TÀI KHOẢN

Số tài khoản: 010292914

Tên khách hàng: Nguyễn Quốc Anh

Địa chỉ khách hàng: 219/21 Trần Thái Tông Q1

Số tiền trong tài khoản: 1000000

Thêm / Cập Nhật Xóa Thoát

STT	Mã tài khoản	Tên khách hàng	Địa chỉ	Số tiền
1	02130131	Nguyễn Thái Công	123 Trương Định P3 TP.HCM	2012012
2	010292914	Nguyễn Quốc Anh	219/21 Trần Thái Tông Q1	1000000

# BÀI 3: LẬP TRÌNH WINDOWS FORM VỚI GIAO DIỆN MDI

## 3.1 MỤC TIÊU ---

- Sử dụng Visual Studio .NET 2022 tạo ứng dụng dạng Windows Forms MDI ( Multiple Document Interface)
- Thực hành với các controls
  - ✓ *MenuStrip*
  - ✓ *ToolStrip*
  - ✓ *StatusStrip*
  - ✓ *Timer*
  - ✓ *Media*

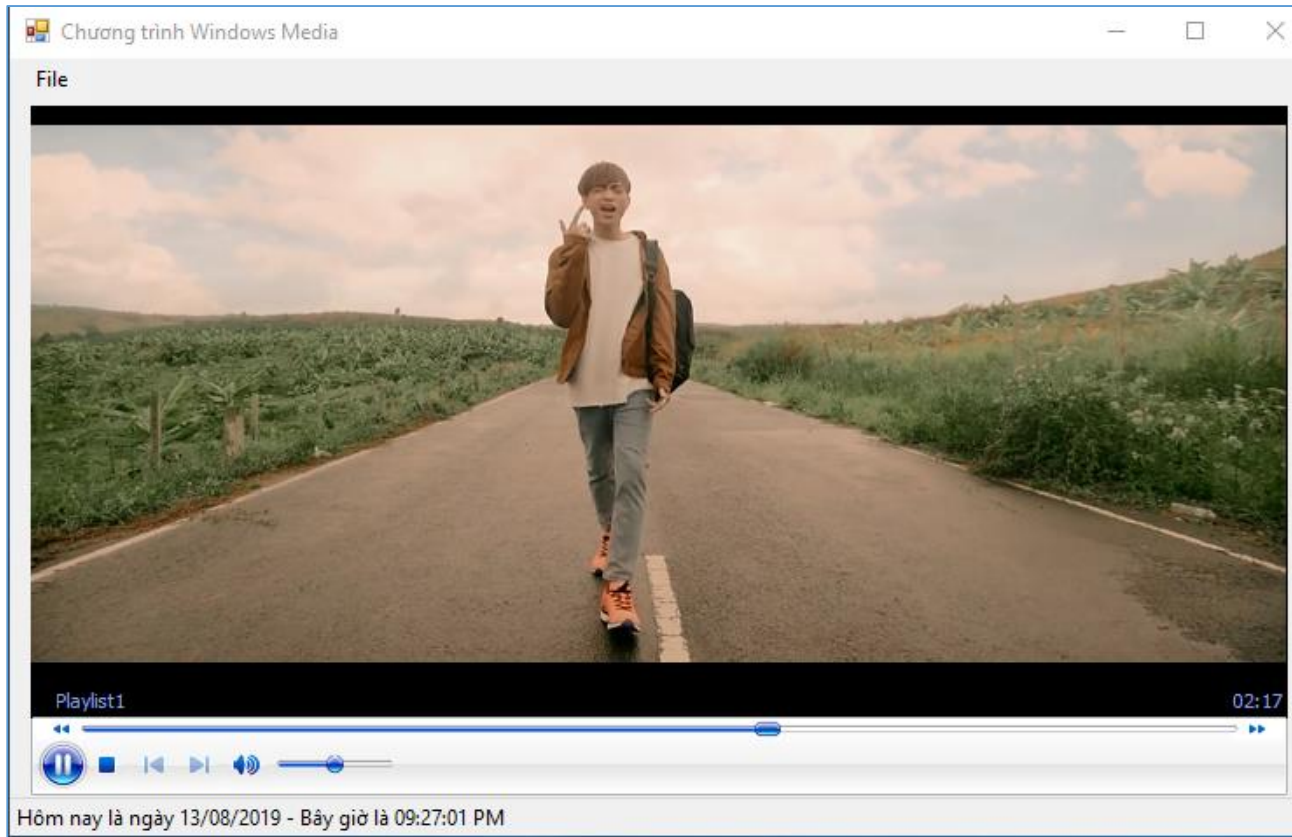
## 3.2 BÀI TẬP ---

**Bài Tập 1:** Tạo Project **Lab03-01** mô phỏng điều khiển Windows media Player để phát các tập tin Audio và Video.

Yêu cầu:

- Ứng dụng chứa Windows Media Player control cho phép Play các file video/sound theo nhiều dạng format ( \*.avi \*.mpeg \*.wav \*.midi \*.mp4 \*.mp3 )
- Menu Bar có 1 Menu File và 2 Sub Menu Open và Exit
  - ✓ Khi chọn Sub Menu Exit sẽ thoát ứng dụng

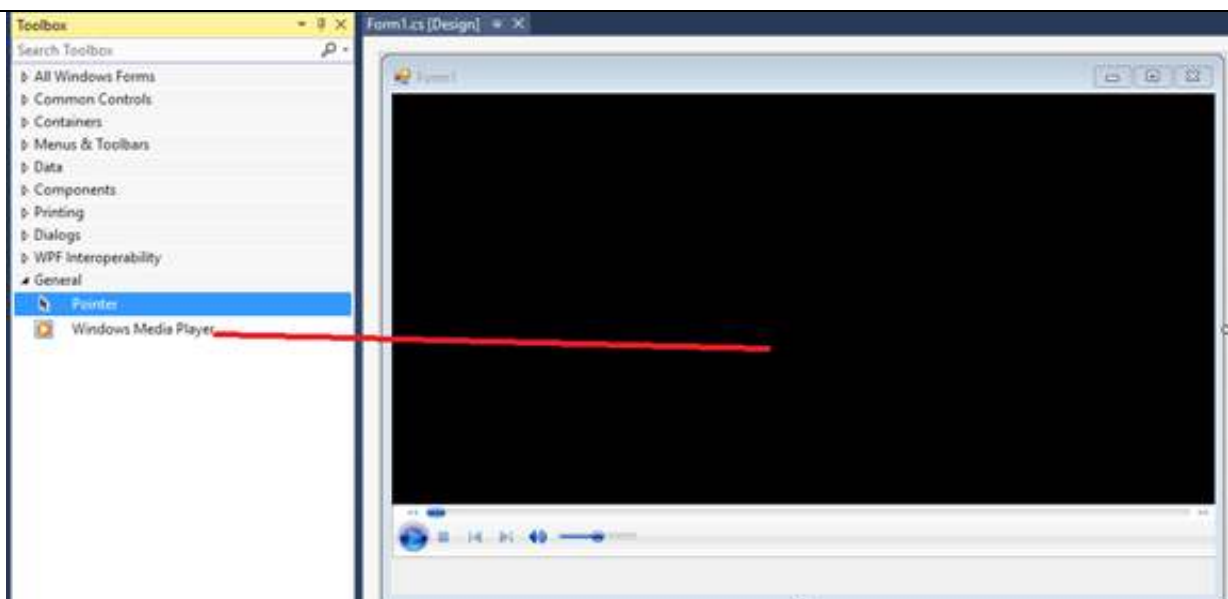
- ✓ Khi chọn Sub Menu Open sẽ mở hộp thoại Open File để chọn mở file Media - Control Multi Media sẽ phát file Media đã chọn.
- StatusStrip hiện thị: Ngày giờ hệ thống và thay đổi giờ theo mỗi giây.



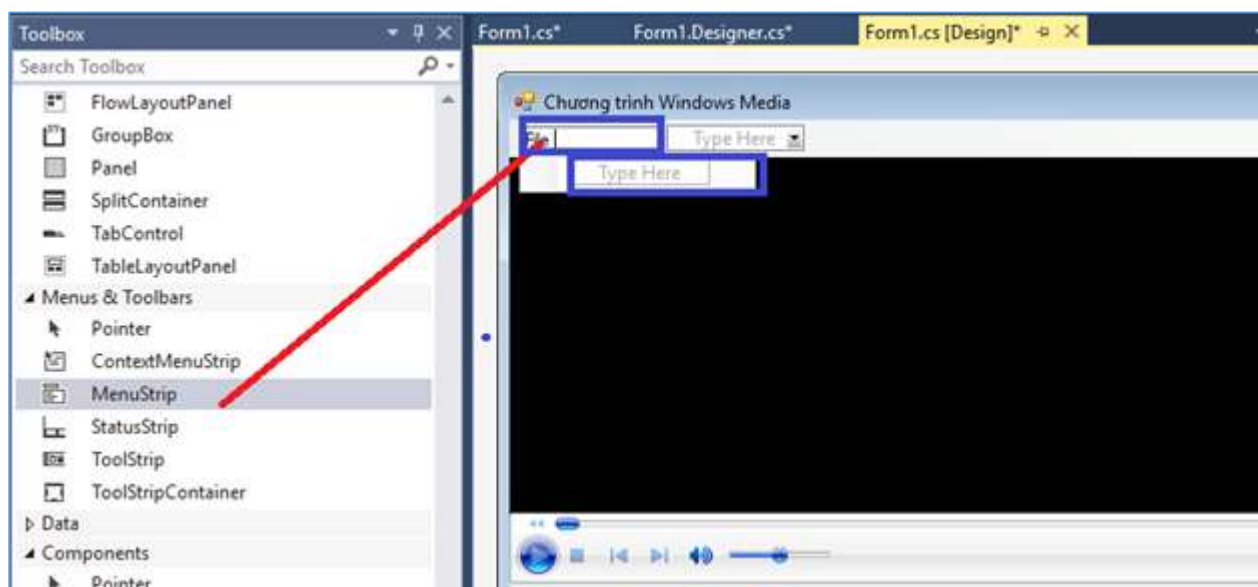
*Hình 1: Thiết kế giao diện có windows media*

**Hướng dẫn:**

**Bước 1: Thiết kế giao diện:** Để sử dụng được điều khiển Windows Media Player click phải lên ToolBox và chọn Choose Items, trong cửa sổ Choose Toolbox Item, chọn thẻ COM Components và chọn thư viện Windows Media Player.

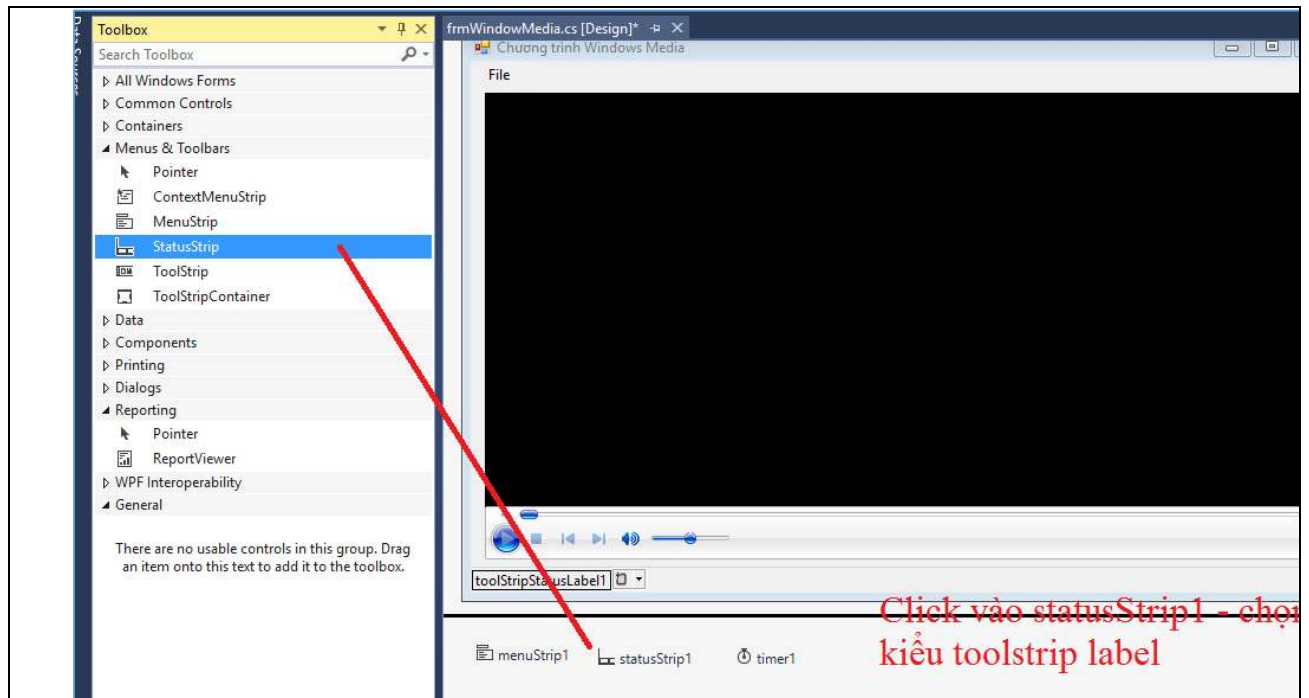


✓ Để tạo Menu **File**: Kéo thả MenuStrip trên toolbar ( Menu & ToolBars)

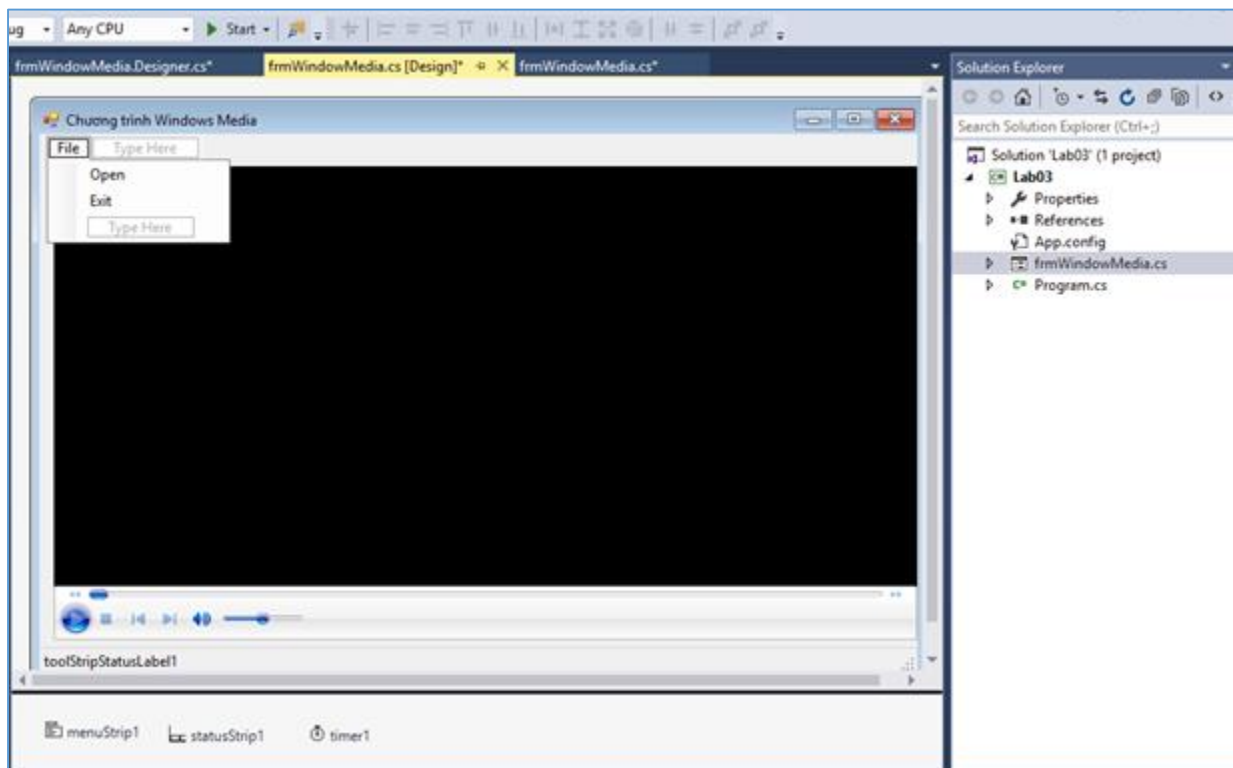


Hình 3: Kéo MenuStrip để tạo Menu

✓ Để tạo Status: Kéo thả StatusStrip



- ✓ Để thực hiện các sự kiện liên quan đến thời gian: Kéo **Timer** vào forms
- ✓ Giao diện sau thiết kế



Hình 5: Giao diện sau khi thiết kế

**Bước 2: Viết các lệnh xử lý**✓ **Viết xử lý cho sự kiện Timer**

- Thiết lập thuộc tính của Timer **Interval:1000** (Đơn vị ms, cứ mỗi 1 giây sự kiện tick sẽ xảy ra)
- Chọn thuộc tính Enable = True để Timer được hoạt động
- Viết lệnh xử lý cho sự kiện **Timer1\_Tick()**

```
private void timer1_Tick(object sender, EventArgs e)
{
    this.toolStripStatusLabel1.Text = string.Format("Hôm nay là ngày {0} - Bây giờ là {1}", DateTime.Now.ToString("dd/MM/yyyy"), DateTime.Now.ToString("hh:mm:ss tt"));
}
```

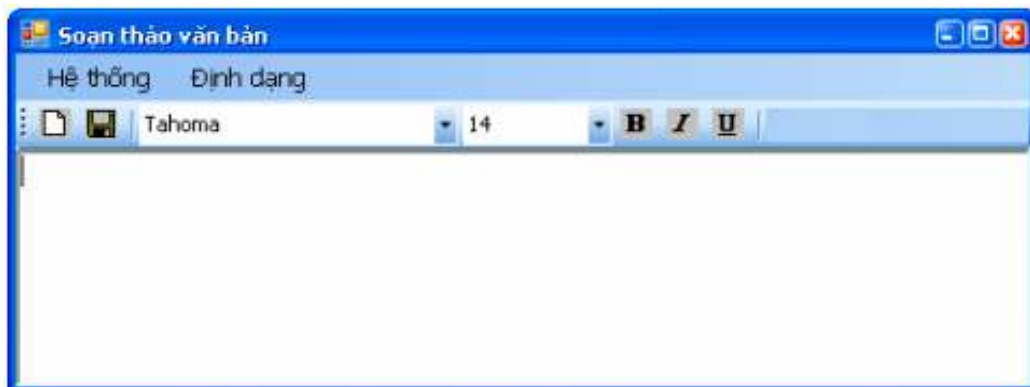
✓ **Viết xử lý cho SubMenu Open**

// Xử lý menu Open Sử dụng OpenFileDialog

```
private void openToolStripMenuItem_Click(object sender, EventArgs e)
{
    //Tạo hộp thoại mở file
    OpenFileDialog dlg = new OpenFileDialog();
    //lọc hiển thị các loại file
    dlg.Filter = "AVI file| *.avi | MPEG File | *.mpeg | Wav File | *.Wav | Midi File | *.midi | Mp4 File | *.mp4 | MP3 | *.mp3";
    //hiển thị openFileDialog
    if (dlg.ShowDialog() == DialogResult.OK)
        axWindowsMediaPlayer1.URL = dlg.FileName; //Lấy tên file cần mở
}
```

**Bài tập 2: Soạn thảo văn bản**

Thêm project **Lab03-02** vào Solution có giao diện như sau

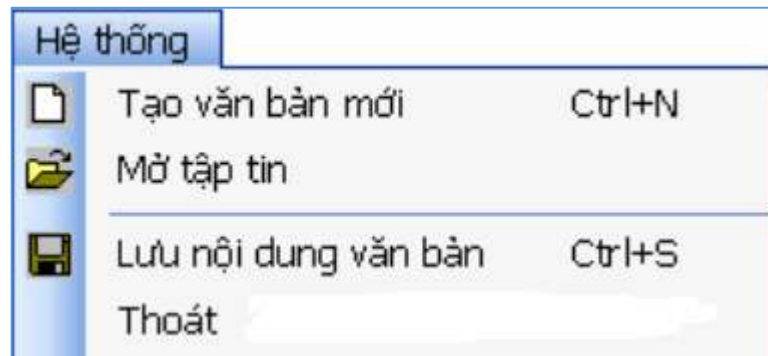


Sử dụng **RichTextBox** để thiết kế điều khiển hiển thị và nhập nội dung văn bản:

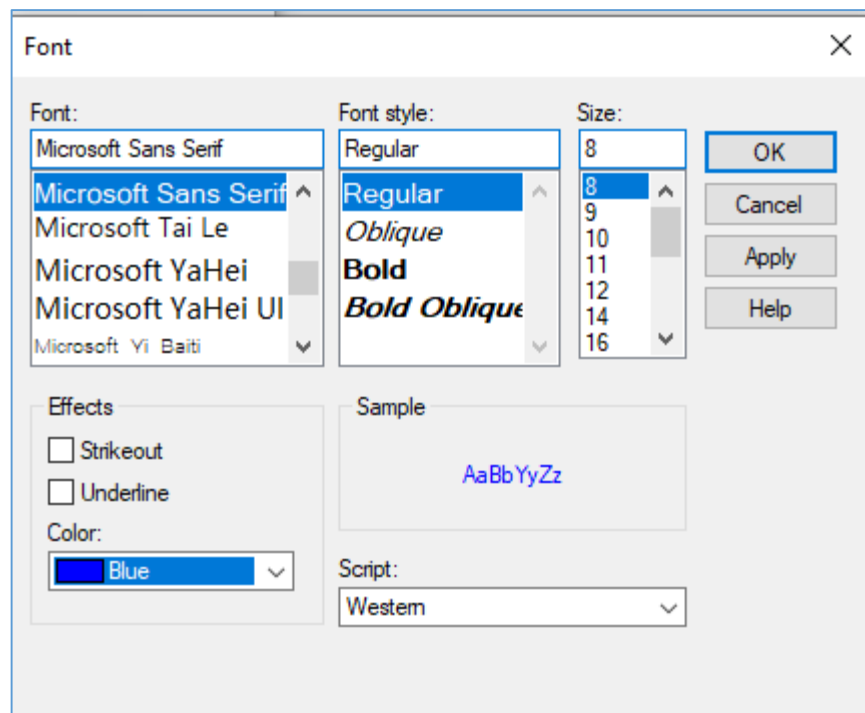
Sử dụng công cụ MenuStrip để tạo Hệ Thống và Định Dạng

Sử dụng công cụ **ToolStrip** để tạo thanh công cụ chứa các button và image như trên

- ✓ Trong menu Hệ thống thiết kế như sau



- ✓ Khi click vào menu định dạng gọi **FontDialog** có sẵn của windows



Hướng Dẫn cách gọi

```
FontDialog fontDlg = new FontDialog();  
fontDlg.ShowColor = true;  
fontDlg.ShowApply = true;  
fontDlg.ShowEffects = true;  
fontDlg.ShowHelp = true;  
if (fontDlg.ShowDialog() != DialogResult.Cancel)
```



```
{  
    richText.ForeColor= fontDlg.Color;  
    richText.Font = fontDlg.Font;  
}
```


- ✓ **Cài đặt xử lý cho các chức năng**


2.1 Khi mới mở Form, thực hiện:


- Tạo dữ liệu cho ComboBox Font: chứa tất cả các Font chữ của hệ thống.
- Tạo dữ liệu cho ComboBox Size: chứa các giá trị từ 8 → 72.

(8, 9, 10, 11, 12, 14, 16, 18, 20, 22, 24, 26, 28, 36, 48, 72)


- Tạo giá trị mặc định ban đầu là Font **Tahoma**, Size **14**


2.2 Khi chọn Tạo văn bản mới (hoặc nhấn nút ): Xóa nội dung hiện có trên RichTextBox và tạo lại các giá trị mặc định như Font, Size, ...


2.3 Khi chọn Mở tập tin (hoặc nhấn nút ): Hiển thị hộp thoại mở tập tin (OpenFileDialog) cho phép người dùng chọn tập tin văn bản (\*.txt hoặc \*.rtf) để mở.

2.4 Khi chọn Lưu nội dung văn bản (hoặc nhấn nút ): Lưu nội dung văn bản trên RichTextBox xuống tập tin.

- ✓ Nếu là văn bản mới và trước đó chưa lưu lần nào thì hiển thị hộp thoại lưu tập tin (SaveFileDialog) cho phép người dùng chọn thư mục cần lưu tập tin với kiểu tập tin cần lưu là \*.rtf.
- ✓ Nếu là văn bản đã được mở trước đó thì thông báo cho người dùng lưu văn bản thành công.

2.5 Nút : Khi chọn, tùy thuộc vào trạng thái của nút để xử lý nội dung của vùng văn bản đang được chọn có được in đậm hay không.

Nút : Khi chọn, tùy thuộc vào trạng thái của nút để xử lý nội dung của vùng văn bản đang được chọn có được in nghiêng hay không.

Nút : Khi chọn, tùy thuộc vào trạng thái của nút để xử lý nội dung của vùng văn bản đang được chọn có được gạch dưới hay không.

### Hướng Dẫn

- Lấy tất cả các fonts từ hệ thống đưa vào combobox Fonts  
`foreach (FontFamily font in new InstalledFontCollection().Families)`  
`{`  
`cmbFonts.Items.Add(font.Name);`  
`}`
- Để load file sử dụng `richText.LoadFile(...);`
- Để Save file sử dụng `richText.SaveFile(...);`

**Bài tập 3: Chương trình truyền dữ liệu giữa các form giao diện**

Thêm project **Lab03-03** vào cùng solution có giao diện như sau

- Form Quản lý sinh viên có Menu Chức năng gồm 2 SubMenu là **Thêm mới** và **Thoát**

Sử dụng MenuStrip để tạo menu

Sử dụng ToolStrip để tạo hình và nút bấm Thêm mới, Tìm kiếm theo tên

Số TT	Mã Số SV	Tên Sinh Viên	Khoa	Điểm TB
-------	----------	---------------	------	---------

- Khi click vào menu thêm mới (Ctrl + N) hoặc ở icon hình thêm mới trên toolStrip thì gọi Form nhập liệu sinh viên như sau

Có 3 khoa được hiện thị trong Combobox (Style là DropDownList) gồm “Công nghệ thông tin”, “Ngôn ngữ Anh” và “Quản trị kinh doanh”

Khi click vào Nút Thêm Mới

- ✓ Thông báo cho người dùng các thông tin bắt buộc phải nhập liệu (Mã số, Tên Sinh Viên, Điểm)
- ✓ Thông báo cho người dùng thông tin mã số sinh viên nếu bị trùng trong DataGridView ở Form trước đó
- ✓ Thông báo cho người dùng dữ liệu nhập điểm trong phạm vi từ (0 -10)
- ✓ Trở về Form chính và đưa dữ liệu vào GridView hiện thị

Ở phần tìm kiếm theo tên sinh viên, khi Textbox tìm kiếm thay đổi thì luôn luôn tìm lại dữ liệu chứa tên tìm kiếm (không phân biệt hoa thường)

	Số TT	Mã Số SV	Tên Sinh Viên	Khoa	Điểm TB
▶	1	1	Nguyễn văn Bảo	Công nghệ thông tin	5.6
	2	BH030343	Phạm Chí Bình	Công nghệ thông tin	8.9

# BÀI 4: LẬP TRÌNH VỚI CƠ SỞ DỮ LIỆU SỬ DỤNG ENTITY FRAMEWORK

## 4.1 MỤC TIÊU

- Hướng dẫn sinh viên làm quen với việc xây dựng ứng dụng Windows Application có kết nối với CSDL SQL Server bằng Entity Framework của .NET.
- Sử dụng trong EntityFramework với hướng tiếp cận Code First (From Database) đã có CSDL
- Thiết kế các Form nhập liệu cho các bảng trong cơ sở dữ liệu (hiện thị, thêm, xóa, sửa)

## 4.2 BÀI TẬP

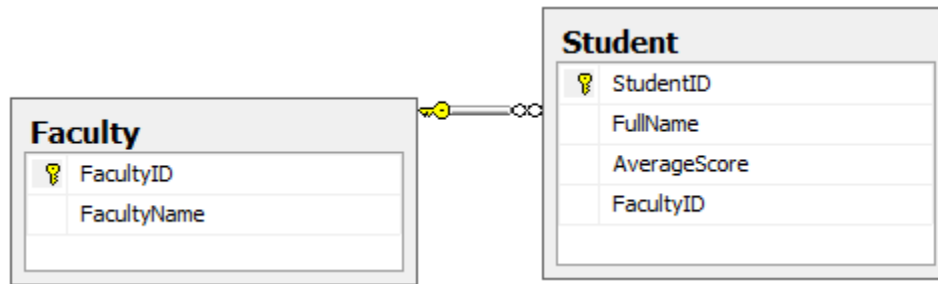
Sử dụng SQL Server tạo cơ sở dữ liệu "QuanLySinhVien" đơn giản với 2 bảng: Sinh viên và Khoa như sau

**Student** (**StudentID**, FullName, AverageScore, FacultyID)

**Faculty**(**FacultyID**, FacultyName)

DESKTOP-AA3CGL7...n - dbo.Student				DESKTOP-AA3CGL7...en - dbo.Faculty			
	Column Name	Data Type	Allow Nulls		Column Name	Data Type	Allow Nulls
PK	StudentID	nvarchar(20)	<input type="checkbox"/>	PK	FacultyID	int	<input type="checkbox"/>
	FullName	nvarchar(200)	<input type="checkbox"/>		FacultyName	nvarchar(200)	<input type="checkbox"/>
	AverageScore	float	<input type="checkbox"/>				<input type="checkbox"/>
	FacultyID	int	<input type="checkbox"/>				

- ✓ Tạo mối quan hệ 2 bảng như sau:



- ✓ Nhập liệu sẵn vào cơ sở dữ liệu một số dòng

DESKTOP-AA3CGL7....en - dbo.Faculty		DESKTOP-AA3CGL7...n - dbo.Student			
FacultyID	FacultyName	StudentID	FullName	AverageScore	FacultyID
1	Công Nghệ Thông Tin	1611061916	Nguyễn Trần Hoàng Lan	4.5	1
2	Ngôn Ngữ Anh	1711060596	Đàm Minh Đức	2.5	1
3	Quản trị kinh doanh	1711061004	Nguyễn Quốc An	10	2

**Bài Tập 1** Sử dụng EntityFrameWork với mô hình Code First để kết nối CSDL

- ✓ Viết chương trình quản lý sinh viên có giao diện tương tự sau đây

Quản Lý Sinh Viên

## Quản lý thông tin Sinh viên

**Thông Tin Sinh Viên**

Mã Số SV:

Họ Tên:

Khoa:

Điểm TB:

	Mã Số SV	Họ Tên	Tên Khoa	Điểm TB
	1611061916	Nguyễn Trần Hoàng Lan	Công Nghệ Thông Tin	4.5
	1711060596	Đàm Minh Đức	Công Nghệ Thông Tin	2.5
▶	1711061004	Nguyễn Quốc An	Ngôn Ngữ Anh	10

### Yêu Cầu Xử Lý

#### 1.1 Sự kiện Form\_load:

- Hiển thị danh sách sinh viên hiện có trong CSDL (Lấy từ bảng sinh viên)
- ComboBox Khoa lấy từ bảng **Faculty** và hiển thị tên khoa

#### 1.2 Khi nhấn vào nút "Thêm" Hoặc "Sửa"

- Kiểm tra các thông tin bắt buộc phải nhập liệu cho sinh viên như mã sinh viên, tên, và điểm trung bình. Nếu để trống sẽ xuất hiện thông báo lỗi "**Vui lòng nhập đầy đủ thông tin!**"
- Kiểm tra mã số sinh viên phải có 10 kí tự. Nếu không sẽ xuất thông báo "Mã số sinh viên phải có 10 kí tự!"
- Nếu trường hợp nhấn vào nút "Thêm" thì Thêm mới dữ liệu sinh viên vừa nhập vào CSDL, Load lại DataGridView, và thông báo "**Thêm mới dữ liệu thành công!**".
- Nếu trường hợp nhấn vào nút "Sửa". Nếu mã sinh viên đã tồn tại thì Cập nhật dữ liệu sinh viên vào CSDL, và thông báo "**Cập nhật dữ liệu thành công!**". Nếu mã sinh viên đó không tồn tại thì xuất thông báo "**Không tìm thấy MSSV cần sửa!**"
- Reset lại dữ liệu về giá trị ban đầu sau khi thêm/ sửa thành công

### 1.3 Khi nhấn vào nút "Xóa"

- Kiểm tra nếu MSSV cần xóa không tồn tại trong CSDL thì thông báo lỗi "**Không tìm thấy MSSV cần xóa!**".
- Ngược lại thì xuất hiện cảnh báo YES/NO. Nhấn YES sẽ thực hiện xóa dòng dữ liệu sinh viên trong DataGridView và thông báo "**Xóa sinh viên thành công!**".
- Reset lại dữ liệu về giá trị ban đầu sau khi xóa thành công

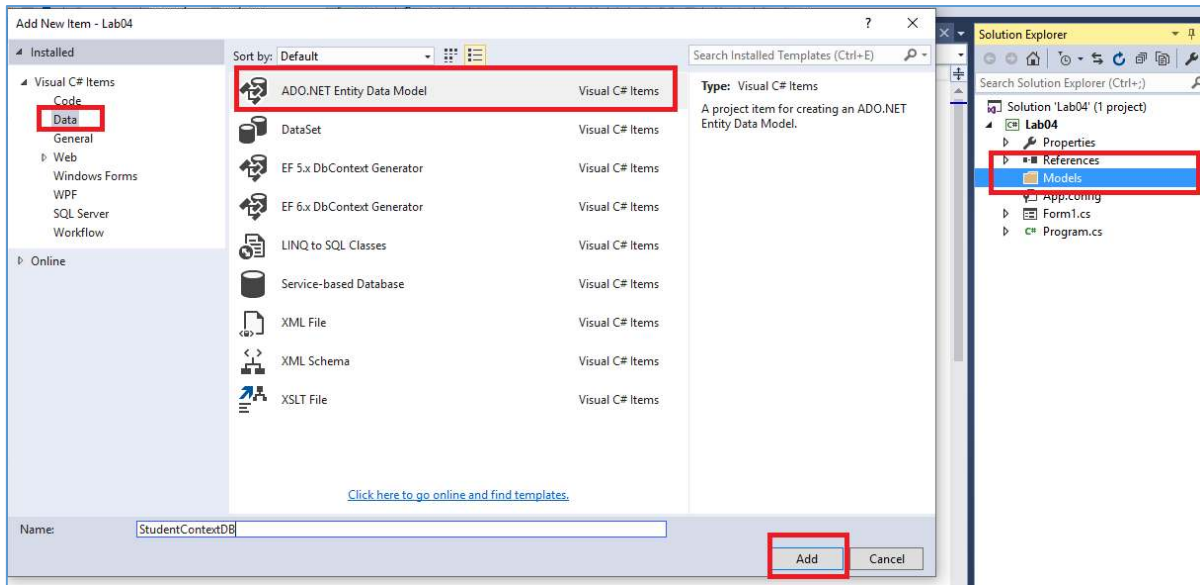
1.4 Viết code cho sự kiện ở DataGridView, người dùng chọn 1 dòng thì thể hiện ngược lại thông tin của các sinh viên đã chọn ở phần nhập liệu (bên trái).

### Hướng Dẫn

**Bước 1:** Entity Framework sinh ra các class chúng ta nên tạo trong 1 thư mục (**Models**) để dễ dàng quản lý.

Click chuột phải vào Models chọn **New Item**. Chọn Loại **Data/ ADO.NET Entity Data Model**

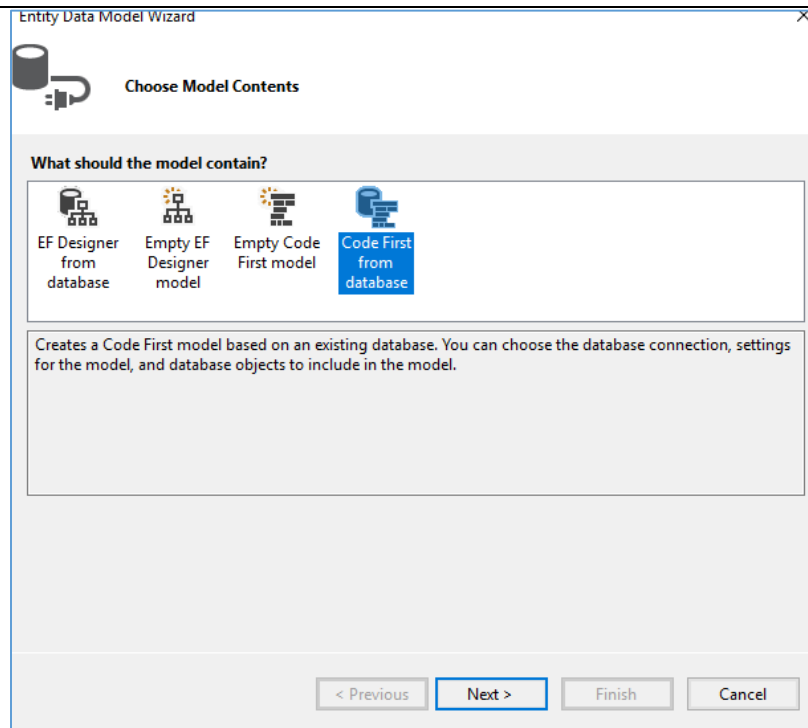
Đặt tên context là "**StudentContextDB**" (mặc định là Model1). Và chọn **Add**



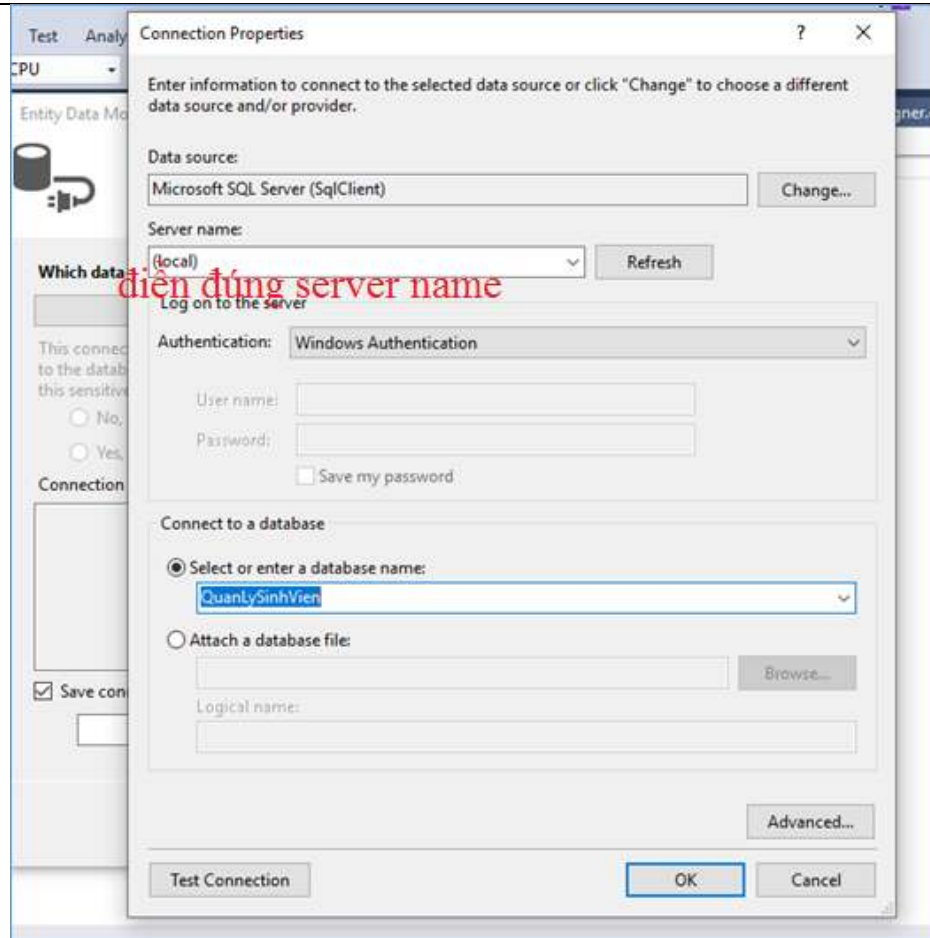
Sau khi chọn Add có 4 hướng loại Entity model để kết nối với cơ sở dữ liệu

- Database First
- Model First
- Code First với hướng tiếp cận tạo ra cơ sở dữ liệu
- Code First với hướng tiếp cận đã có sẵn cơ sở dữ liệu

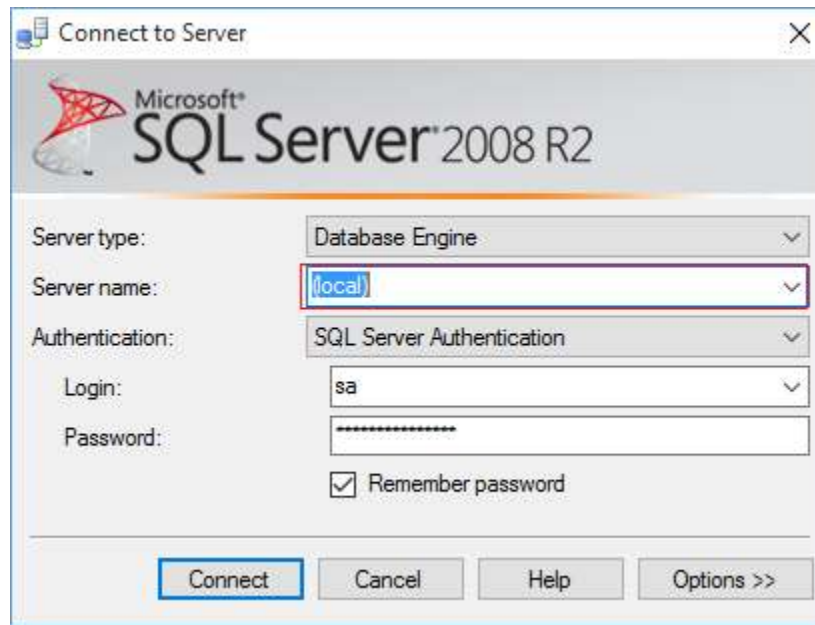




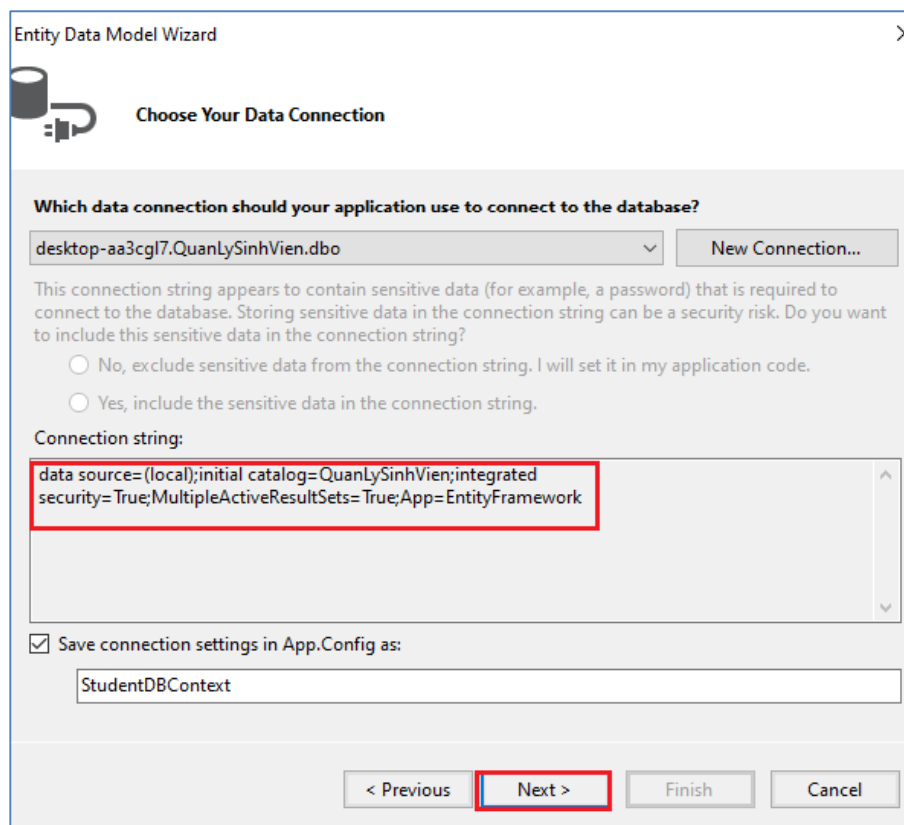
Ta chọn loại model là “**Code first from database**”. Chọn **Next**  
Tìm đúng cơ sở dữ liệu Student ở SQL để trở database name vào



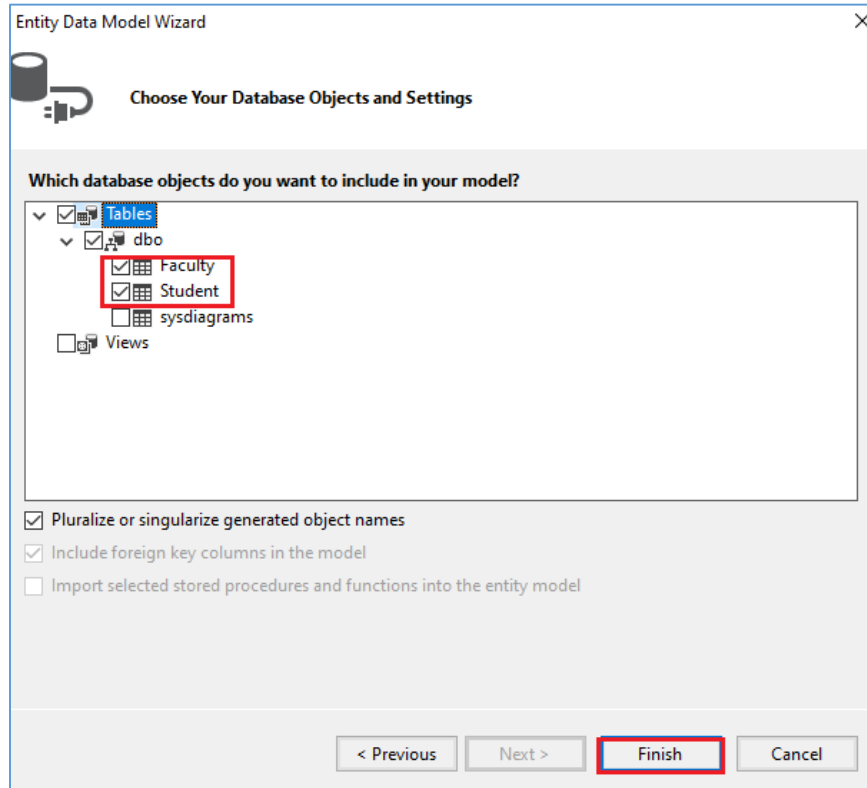
Chọn đúng tên server Name (trên từng máy có thể khác nhau - sinh viên có thể re-connect lại database để xem đúng tên Server Name. Ở ví dụ đây là trên local máy cá nhân)



Chọn Next để tiếp tục tạo. Connect String sẽ được lưu ở file App.Config



Sau đó chọn các bảng muốn tạo object (ở đây chọn 2 bảng từ CSDL)



Sau khi Finish, Entity Framework đã tạo các class tương ứng như trong cơ sở dữ liệu



Một số thông tin cần lưu ý:

- App.Config: Sẽ lưu trữ connection String và tên của Context sử dụng
- StudentContextDB: Lớp chứa tập hợp DataSet cho các table được chọn.

VS.Net tự động sinh ra các file cs tương ứng map với CSDL. Sinh viên kiểm tra từng file (Student, Faculty) để hiểu cách mapping tương ứng với CSDL.

- ✓ Cách Sử dụng Entity để thao tác với cơ sở dữ liệu như: Lấy tất cả, thêm, xóa, sửa với CSDL (Sinh viên đọc kĩ và thử để thực hiện bài tập)

```
//luôn luôn sử dụng context để làm việc với các class
StudentContextDB context = new StudentContextDB();

//1. lấy tất cả các sinh viên từ bảng Student
List<Student> listStudent = context.Students.ToList();
//2. lấy sinh viên đầu tiên có StudentID = ID cho trước
Student db = context.Students.FirstOrDefault(p => p.StudentID == ID);
//3. insert 1 đối tượng sinh viên s vào database
Student s = new Student() { StudentID = "99", FullName = "test insert", AverageScore
= 100 };
context.Students.Add(s);
context.SaveChanges();
//4. Update sinh viên -> lấy item ra và cần update thuộc tính nào thì set thuộc tính
```

đó

```
Student dbUpdate = context.Students.FirstOrDefault(p => p.StudentID == ID);
if(dbUpdate!= null){
    dbUpdate.FullName = "Update FullName"; //....
    context.SaveChanges(); //lưu thay đổi
}

//5. Xóa Student có ID cho trước , tương tự update
Student dbDelete = context.Students.FirstOrDefault(p => p.StudentID == ID);
if (dbDelete != null) {
    context.Students.Remove(db);
    context.SaveChanges(); // lưu thay đổi
}
//6. Lưu ý: Nếu sử dụng using System.Data.Entity.Migrations; có thể
sử dụng hàm AddOrUpdate để thay thế Add và Update từ EntityFramework
6.0.0.0
context.Students.AddOrUpdate(s); //Add or Update sinh viên s
context.SaveChanges();
```

## Bước 2: Thiết kế và lập trình – Viết sự kiện Form-Load

```
private void frmStudentManagement_Load(object sender, EventArgs e)
{
    try
    {
        StudentContextDB context = new StudentContextDB();
        List<Faculty> listFaculty = context.Faculties.ToList(); //lấy các khoa
        List<Student> listStudent = context.Students.ToList(); //lấy sinh viên
        FillFacultyCombobox(listFaculty);
        BindGrid(listStudent);
    }
    catch (Exception ex)
```

```

    {
        MessageBox.Show(ex.Message);
    }
}

//Hàm binding list có tên hiển thị là tên khoa, giá trị là Mã khoa
private void FillFacultyCombobox(List<Faculty> listFaculty)
{
    this.cmbFaculty.DataSource = listFaculty;
    this.cmbFaculty.DisplayMember = "FacultyName";
    this.cmbFaculty.ValueMember = "FacultyID";
}

//Hàm binding gridView từ list sinh viên
private void BindGrid(List<Student> listStudent)
{
    dgvStudent.Rows.Clear();
    foreach (var item in listStudent)
    {
        int index = dgvStudent.Rows.Add();
        dgvStudent.Rows[index].Cells[0].Value = item.StudentID;
        dgvStudent.Rows[index].Cells[1].Value = item.FullName;
        dgvStudent.Rows[index].Cells[2].Value = item.Faculty.FacultyName ;
        dgvStudent.Rows[index].Cells[3].Value = item.AverageScore;
    }
}

// Sinh viên tự viết các sự kiện thêm, xóa, sửa sau khi đọc hướng dẫn sử dụng Entity để
thao tác với CSDL.

```

**Bài tập 2:** Tạo Form quản lý thông tin các khoa ở cùng project trong bài tập 1

- Thêm 1 cột **TotalProfessor** (tổng số giáo sư) kiểu INT cho phép **NULL** vào bảng **Faculty**

DESKTOP-AA3CGL7....en - dbo.Faculty*			
	Column Name	Data Type	Allow Nulls
🔑	FacultyID	int	<input type="checkbox"/>
	FacultyName	nvarchar(200)	<input type="checkbox"/>
▶	TotalProfessor	int	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

- Thêm 1 form mới là **frmFaculty** có đủ các chức năng thêm, xóa, sửa, hiển thị thông tin khoa tương tự như quản lý Sinh viên ở bài tập 1.

Mã Khoa	Tên Khoa	Tổng số GS	
1	Công Nghệ Thông Tin		
2	Ngôn Ngữ Anh		
3	Quản trị kinh doanh		

- Tạo 1 Button ở Form Quản lý sinh viên để khi click đó sẽ gọi sang Form quản lý thông tin các khoa (Hoặc sinh viên có thể dùng MenuStrip – tạo ra Sub menu)
- Thực hiện các yêu cầu trên form quản lý khoa: Lấy dữ liệu vào DataGridView, Thêm/Sửa, Xóa và Đóng form.

**Chú ý:** Khi CSDL có thay đổi => nên cập nhật lại phần models **được thay đổi** bằng cách tương tự như lúc tạo ra ban đầu (xóa đi – tạo lại hoặc đưa phần thay đổi chèn vào models hiện tại). Mục đích để đảm bảo Models phải được mapping đúng với CSDL.

### Bài tập 3: Tìm kiếm sinh viên

**Thiết kế chương trình quản lý sinh viên như giao diện sau** (Thiết kế thêm Chức năng, Quản lý khoa, tìm kiếm và form tìm kiếm sinh viên từ bài tập 1-2)

Quản Lý Sinh Viên

Chức năng

Quản lý khoa Tìm kiếm

Thông Tin Sinh Viên

Mã Số SV

Họ Tên

Khoa

Điểm TB

Thêm Sửa Xóa

	Mã Số SV	Họ Tên	Tên Khoa	Điểm TB
▶	1611061916	Nguyễn Trần Hoàng Lan	Công Nghệ Thông Tin	4.5
	1711060596	Đàm Minh Đức	Công Nghệ Thông Tin	2.5
	1711061004	Nguyễn Quốc An	Ngôn Ngữ Anh	10

- Sử dụng ToolStrip, 2 button Quản lý khoa và Tìm kiếm
- Ở Menu chức năng: thể hiện các chức năng có phím tắt để tới các form: Quản lý khoa (F2) và Tìm Kiếm (Ctrl +F)

Chức năng

Quản lý khoa F2

Tìm Kiếm Ctrl+F

Thoát

- Thiết kế form Tìm kiếm và thực hiện tìm kiếm thông tin sinh viên

Tim kiếm

Thông Tin Tìm Kiếm

Mã Số SV

Họ Tên

Khoa

Tim Kiếm Xóa Trở về

Mã Số SV	Họ Tên	Tên Khoa	Điểm TB
----------	--------	----------	---------

Kết quả tìm kiếm 0

Các khoa được lấy từ CSDL. Mặc định chưa chọn là Empty



- ✓ Khi người dùng click vào button tìm kiếm sẽ tìm kiếm thông tin sinh viên thỏa các điều kiện tìm kiếm ( không nhập có nghĩa là bỏ qua điều kiện tìm kiếm đó)
- ✓ Khi người dùng click vào button xóa: Trả lại giá trị mặc định như khi load form tìm kiếm.

Ví dụ: Tìm các sinh viên thuộc khoa công nghệ thông tin

Mã Số SV	Họ Tên	Tên Khoa	Điểm TB
1611061916	Nguyễn Trần Hoàng Lan	Công Nghệ Thông Tin	4.5
1711060596	Đàm Minh Đức	Công Nghệ Thông Tin	2.5

Kết quả tìm kiếm 2

**Bài tập 4:** Cho cơ sở dữ liệu quản lý sản phẩm và đơn hàng như sau

- ✓ Sử dụng cơ sở dữ liệu SQL server có 3 bảng **Product**, **Order**, **Invoice** lần lượt như sau

**Product:** Lưu trữ thông tin sản phẩm (**Mã sản phẩm**, Tên Sản phẩm, Đơn vị Tính, Giá Mua, Giá Bán)

**Order:** Lưu trữ chi tiết thông tin đơn hàng (**Số HĐ, Số TT**, Mã SP, Tên SP, ĐVT, Đơn giá, Số lượng)

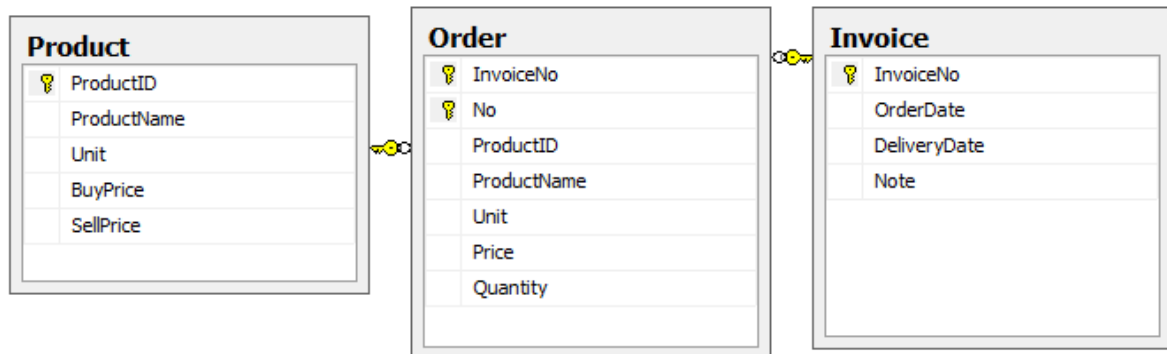
**Invoice:** Lưu trữ thông tin hóa đơn đặt hàng (**Số HĐ**, Ngày đặt hàng, ngày giao hàng, ghi chú)

Column Name	Data Type	Allow Nulls
ProductID	nvarchar(20)	<input type="checkbox"/>
ProductName	nvarchar(100)	<input type="checkbox"/>
Unit	nvarchar(20)	<input type="checkbox"/>
BuyPrice	decimal(18, 0)	<input checked="" type="checkbox"/>
SellPrice	decimal(18, 0)	<input checked="" type="checkbox"/>

Column Name	Data Type	Allow Nulls
InvoiceNo	nvarchar(20)	<input type="checkbox"/>
No	int	<input type="checkbox"/>
ProductID	nvarchar(20)	<input type="checkbox"/>
ProductName	nvarchar(100)	<input checked="" type="checkbox"/>
Unit	nvarchar(20)	<input checked="" type="checkbox"/>
Price	decimal(18, 0)	<input type="checkbox"/>
Quantity	int	<input type="checkbox"/>

Column Name	Data Type	Allow Nulls
InvoiceNo	nvarchar(20)	<input type="checkbox"/>
OrderDate	datetime	<input type="checkbox"/>
DeliveryDate	datetime	<input type="checkbox"/>
Note	nvarchar(255)	<input checked="" type="checkbox"/>

✓ Sơ đồ diagrams



Sử dụng script để tạo ra nhanh CSDL, và dữ liệu tương ứng

```

USE [ProductOrder]
GO
/***** Object: Table [dbo].[Invoice]      Script Date: 07/04/2020 23:13:15
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Invoice] (
    [InvoiceNo] [nvarchar](20) NOT NULL,
    [OrderDate] [datetime] NOT NULL,
    [DeliveryDate] [datetime] NOT NULL,
    [Note] [nvarchar](255) NULL,
    CONSTRAINT [PK_Invoice] PRIMARY KEY CLUSTERED
(
    [InvoiceNo] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/***** Object: Table [dbo].[Product]      Script Date: 07/04/2020 23:13:15
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Product] (
    [ProductID] [nvarchar](20) NOT NULL,
    [ProductName] [nvarchar](100) NOT NULL,
    [Unit] [nvarchar](20) NOT NULL,
    [BuyPrice] [decimal](18, 0) NULL,
    [SellPrice] [decimal](18, 0) NULL,
  
```

```

CONSTRAINT [PK_Product] PRIMARY KEY CLUSTERED
(
    [ProductID] ASC
) WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/***** Object: Table [dbo].[Order]      Script Date: 07/04/2020 23:13:15
*****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Order] (
    [InvoiceNo] [nvarchar](20) NOT NULL,
    [No] [int] NOT NULL,
    [ProductID] [nvarchar](20) NOT NULL,
    [ProductName] [nvarchar](100) NULL,
    [Unit] [nvarchar](20) NULL,
    [Price] [decimal](18, 0) NOT NULL,
    [Quantity] [int] NOT NULL,
    CONSTRAINT [PK_Order] PRIMARY KEY CLUSTERED
(
    [InvoiceNo] ASC,
    [No] ASC
) WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/***** Object: ForeignKey [FK_Order_Invoice]      Script Date: 07/04/2020
23:13:15 *****/
ALTER TABLE [dbo].[Order] WITH CHECK ADD CONSTRAINT [FK_Order_Invoice]
FOREIGN KEY([InvoiceNo])
REFERENCES [dbo].[Invoice] ([InvoiceNo])
GO
ALTER TABLE [dbo].[Order] CHECK CONSTRAINT [FK_Order_Invoice]
GO
/***** Object: ForeignKey [FK_Order_Product]      Script Date: 07/04/2020
23:13:15 *****/
ALTER TABLE [dbo].[Order] WITH CHECK ADD CONSTRAINT [FK_Order_Product]
FOREIGN KEY([ProductID])
REFERENCES [dbo].[Product] ([ProductID])
GO
ALTER TABLE [dbo].[Order] CHECK CONSTRAINT [FK_Order_Product]
GO

```

### Thêm 1 số dữ liệu vào database như sau

```

USE [ProductOrder]
GO
/***** Object: Table [dbo].[Invoice]      Script Date: 07/04/2020 23:14:16
*****/
INSERT [dbo].[Invoice] ([InvoiceNo], [OrderDate], [DeliveryDate], [Note])
VALUES (N'HDX001', CAST(0x0000AAD900000000 AS DateTime),
CAST(0x0000AADA00000000 AS DateTime), N'Giao hàng trước 9h')

```

```
INSERT [dbo].[Invoice] ([InvoiceNo], [OrderDate], [DeliveryDate], [Note])
VALUES (N'HDX002', CAST(0x0000AADA00000000 AS DateTime),
CAST(0x0000AADA00000000 AS DateTime), N'Gọi điện trước khi giao')

INSERT [dbo].[Invoice] ([InvoiceNo], [OrderDate], [DeliveryDate], [Note])
VALUES (N'HDX003', CAST(0x0000AADA00000000 AS DateTime),
CAST(0x0000AADC00000000 AS DateTime), N'giao tu 1-3h')

/***** Object: Table [dbo].[Product]      Script Date: 07/04/2020 23:14:16
*****/

INSERT [dbo].[Product] ([ProductID], [ProductName], [Unit], [BuyPrice],
[SellPrice]) VALUES (N'Product1', N'Sản phẩm 1', N'Cái', CAST(100000 AS
Decimal(18, 0)), CAST(120000 AS Decimal(18, 0)))

INSERT [dbo].[Product] ([ProductID], [ProductName], [Unit], [BuyPrice],
[SellPrice]) VALUES (N'Product2', N'Sản phẩm 2', N'Cái', CAST(90000 AS
Decimal(18, 0)), CAST(120000 AS Decimal(18, 0)))

INSERT [dbo].[Product] ([ProductID], [ProductName], [Unit], [BuyPrice],
[SellPrice]) VALUES (N'Product3', N'Sản phẩm 3', N'Cái', CAST(40000 AS
Decimal(18, 0)), CAST(70000 AS Decimal(18, 0)))

INSERT [dbo].[Product] ([ProductID], [ProductName], [Unit], [BuyPrice],
[SellPrice]) VALUES (N'Product4', N'Sản phẩm 4', N'Hộp', CAST(200000 AS
Decimal(18, 0)), CAST(300000 AS Decimal(18, 0)))

/***** Object: Table [dbo].[Order]      Script Date: 07/04/2020 23:14:16
*****/

INSERT [dbo].[Order] ([InvoiceNo], [No], [ProductID], [ProductName], [Unit],
[Price], [Quantity]) VALUES (N'HDX001', 1, N'Product1', N'Sản phẩm 1', N'Cái',
CAST(120000 AS Decimal(18, 0)), 20)

INSERT [dbo].[Order] ([InvoiceNo], [No], [ProductID], [ProductName], [Unit],
[Price], [Quantity]) VALUES (N'HDX001', 2, N'Product2', N'Sản phẩm 2', N'Cái',
CAST(120000 AS Decimal(18, 0)), 4)

INSERT [dbo].[Order] ([InvoiceNo], [No], [ProductID], [ProductName], [Unit],
[Price], [Quantity]) VALUES (N'HDX001', 3, N'Product4', N'Sản phẩm 4', N'Hộp',
CAST(300000 AS Decimal(18, 0)), 10)

INSERT [dbo].[Order] ([InvoiceNo], [No], [ProductID], [ProductName], [Unit],
[Price], [Quantity]) VALUES (N'HDX002', 1, N'Product4', N'Sản phẩm 1', N'Hộp',
CAST(300000 AS Decimal(18, 0)), 10)

INSERT [dbo].[Order] ([InvoiceNo], [No], [ProductID], [ProductName], [Unit],
[Price], [Quantity]) VALUES (N'HDX002', 2, N'Product2', N'Sản phẩm 3', N'Cái',
CAST(300000 AS Decimal(18, 0)), 12)

INSERT [dbo].[Order] ([InvoiceNo], [No], [ProductID], [ProductName], [Unit],
[Price], [Quantity]) VALUES (N'HDX003', 1, N'Product1', N'Sản phẩm 1', N'Cái',
CAST(120000 AS Decimal(18, 0)), 40)

INSERT [dbo].[Order] ([InvoiceNo], [No], [ProductID], [ProductName], [Unit],
[Price], [Quantity]) VALUES (N'HDX003', 4, N'Product2', N'Sản phẩm 2', N'Cái',
CAST(120000 AS Decimal(18, 0)), 60)
```

**Viết chương trình phần mềm xem thông tin đơn hàng như sau**

Thông Tin Đơn Hàng

☐ Xem tất cả trong tháng

Thời Gian Giao Hàng: 02/10/2019 ~ 02/10/2019

STT	Số HĐ	Ngày Đặt Hàng	Ngày Giao Hàng	Thành Tiền

Tổng Cộng: 0

### Khi load Form

- Thời gian giao hàng được thể hiện trong ngày hiện hành và tự động tìm kiếm dữ liệu có Hóa Đơn phát sinh trong ngày hiện hành này
- Người dùng có thể thay đổi thời gian giao hàng trong 1 khoảng thời gian bất kì, khi đó dữ liệu cũng được tự động thay đổi theo

Thông Tin Đơn Hàng

☐ Xem tất cả trong tháng

Thời Gian Giao Hàng: 02/10/2019 ~ 04/10/2019

STT	Số HĐ	Ngày Đặt Hàng	Ngày Giao Hàng	Thành Tiền
1	HDX001	01/10/2019	02/10/2019	5880000
2	HDX002	02/10/2019	02/10/2019	6600000

- Khi check vào CheckBox Xem tất cả trong tháng, thì thời gian giao hàng sẽ được thể hiện từ **ngày đầu tháng hiện hành** đến **cuối tháng** và hiện thị thông tin giao hàng trong thời gian đó.

# BÀI 5: TỔ CHỨC DỰ ÁN VỚI ENTITY FRAMEWORK

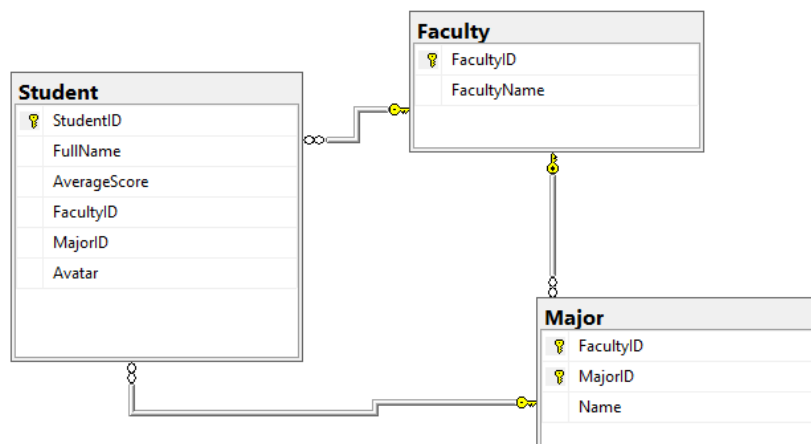
## 5.1 MỤC TIÊU

- Thiết kế ứng dụng với mô hình 3-layer.
- Kết hợp mô hình 3-layer và entity framework
- Truyền dữ liệu giữa các Form
- SV tự tìm hiểu cách triển khai các mẫu: Repository, Unit of work để phát triển các dự án tốt hơn.

## 5.2 BÀI TẬP

Thực hiện quản lý sinh viên và quản lý đăng ký chuyên ngành

Cho CSDL như Sau:



Column Name	Data Type	Allow Nulls
StudentID	nvarchar(10)	<input type="checkbox"/>
FullName	nvarchar(255)	<input type="checkbox"/>
AverageScore	float	<input type="checkbox"/>
FacultyID	int	<input checked="" type="checkbox"/>
MajorID	int	<input checked="" type="checkbox"/>
Avatar	nvarchar(255)	<input checked="" type="checkbox"/>

Column Name	Data Type	Allow Nulls
FacultyID	int	<input type="checkbox"/>
FacultyName	nvarchar(255)	<input type="checkbox"/>
		<input type="checkbox"/>

Column Name	Data Type	Allow Nulls
FacultyID	int	<input type="checkbox"/>
MajorID	int	<input type="checkbox"/>
Name	nvarchar(255)	<input type="checkbox"/>
		<input type="checkbox"/>

**Thêm vào ban đầu 1 số dữ liệu cho bảng Faculty, Major, Student**

FacultyID	FacultyName
1	Cong nghe thong tin
2	Ngon Ngu Anh
3	Quan Tri Kinh Doanh
NULL	NULL

FacultyID	MajorID	Name
1	1	Công nghệ phần mềm
2	1	Tiếng Anh Thương Mại
1	2	Hệ thống thông tin
2	2	Tiếng Anh Truyền Thông
1	3	An toàn thông tin
NULL	NULL	NULL

## 1. Giao diện quản lý sinh viên

Quản Lý Sinh Viên

Chức Năng

Quản lý sinh viên

☐ Chưa ĐK chuyên ngành

MSSV	Họ Tên	Khoa	DTB	Chuyên ngành
1234567890	Nguyen Van B	Cong nghe thong tin	7.5	Công nghệ phần mềm
1234567891	Nguyen Van A	Cong nghe thong tin	4.5	Hệ thống thông tin
2345643213	Nguyen Van C	Ngon Ngu Anh	10	

Thông Tin Sinh Viên

Ma Sinh Viên: 1234567890

Ho Ten: Nguyen Van B

Khoa: Cong nghe thong tin

Diem Trung Binh: 7.5

Anh đại diện:

Add/Update Delete

Các chức năng:

- Check / Uncheck CheckBox "Chưa đăng ký chuyên ngành":
  - ✓ *Checked = False (mặc định)*: Hiển thị tất cả sinh viên, kể ra sinh viên chưa có chuyên ngành và đã có chuyên ngành
  - ✓ *Checked = True*: Hiển thị các sinh viên chưa có chuyên ngành (MajorID = NULL)
- Load Ảnh Avatar / Show avatar
  - ✓ Tạo 1 folder "**Images**" để lưu hình ảnh, trong CSDL "Avatar" chỉ lưu tên hình
  - ✓ Khi thêm mới, nếu có hình ảnh sẽ được lưu vào thư mục Images với tên là {studentID}.{typeFile}.

Ví dụ 1: với mã sv là 1234567890 thì nếu người dùng upload hình file .jpg sẽ được lưu **Images/1234567890.jpg**

Ví dụ 2: với mã sv là 1234567891 thì nếu người dùng upload hình file .png sẽ được lưu **Images/ 1234567891.png**
- Các chức năng Add/Update và Xóa tương tự bài 4.



## 2. Giao diện đăng ký chuyên ngành

The screenshot shows a Windows form titled "frmRegister". At the top, the text "Đăng ký chuyên ngành" (Register Major) is displayed in red. Below this, there are two dropdown menus: "Khoa" (Faculty) with "Ngôn Ngữ Anh" (English Language) selected, and "Chuyên ngành" (Major) with "Tiếng Anh Thương Mại" (Business English) selected. Below the dropdowns is a DataGrid with the following data:

	Chọn	MSSV	Họ Tên	Khoa	DTB
	<input checked="" type="checkbox"/>	2345643213	Nguyễn Văn C	Ngôn Ngữ Anh	10

Below the DataGrid is a large empty rectangular area. At the bottom of the form is a "Register" button.

Khi chọn các DropDownList Khoa sẽ lấy các chuyên ngành tương ứng và hiển thị tất cả các sinh viên của Khoa mà chưa có chuyên ngành.

Có thể check chọn các Sinh viên ở DataGridView, khi click vào "register" thì sẽ chọn chuyên ngành cho sinh viên (được checked)

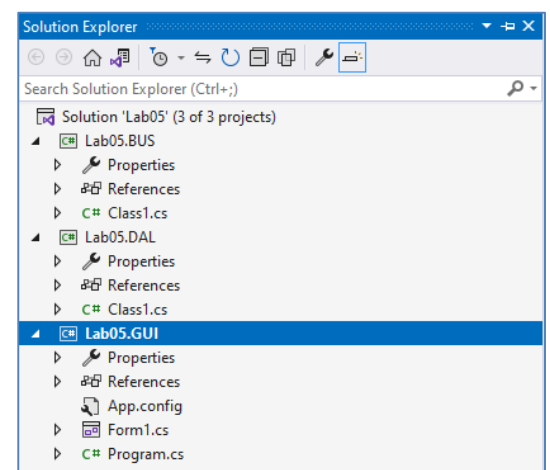
### HƯỚNG DẪN:

**Bước 1:** Tạo dự án theo cấu trúc:

**Lab05.GUI:** là project Windows Form App (.NET Framework)

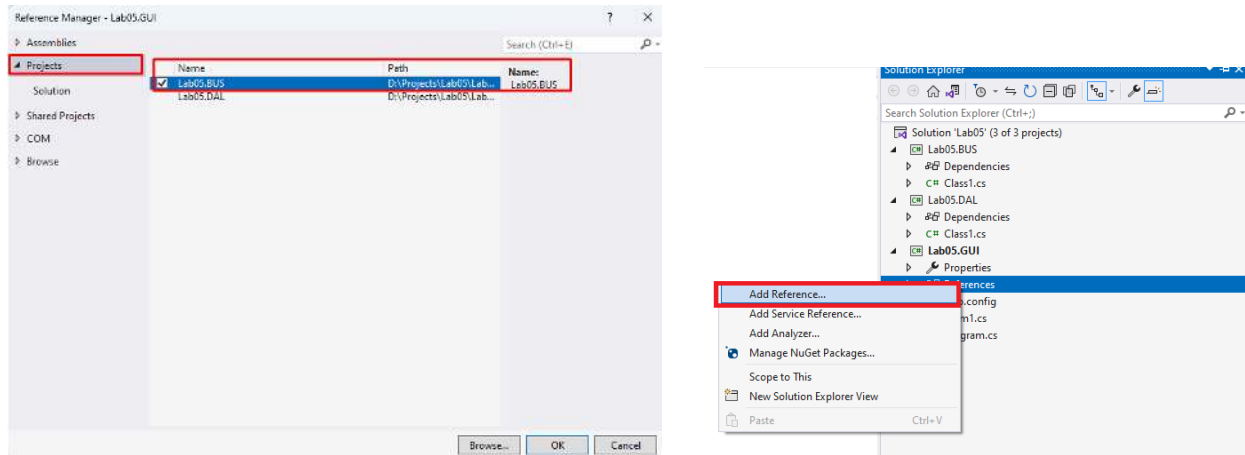
**Lab05.BUS:** là project dạng Class Library (.NET Framework)

**Lab05.DAL:** là project Library (.NET Framework)



Cần liên kết các project bằng cách sử dụng tính năng References => Add reference để liên kết

- ✓ Lab05.GUI liên kết tới Lab05.BUS



Ở project đơn giản này, ta có thể sử dụng DTO là các Entities ở tầng DAL nên liên kết GUI tới DAL (Hoặc có thể tạo thêm tầng DTO để làm rõ mô hình này hơn và có các kiểu đối tượng DTO như: StudentDTO, FacultyDTO, MajorDTO)

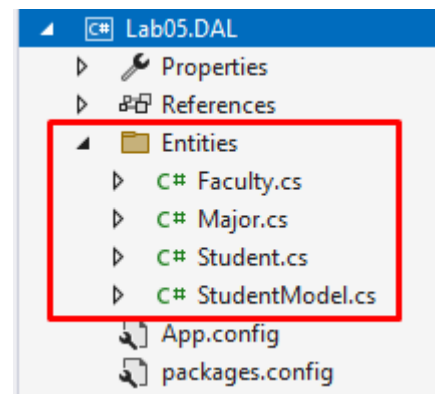
- ✓ Lab05.BUS liên kết tới Lab05.DAL (tự thực hiện)

## **Bước 2:** Thực hiện ở tầng DAL (kết nối CSDL)

Xem lại cách kết nối CSDL sử dụng mô hình

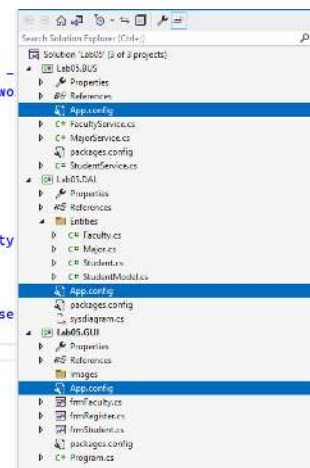
Entity FrameWork Code First theo hướng tiếp cận đã

Có cơ sở dữ liệu



Chú ý: Thông tin kết nối CSDL được lưu ở App.Config trong tầng DAL. Tuy nhiên khi gọi từ các tầng khác chúng ta cũng cần phải có file App.Config tương ứng

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <configSections>
    <!-- For more information on Entity Framework configuration, visit http://go.microsoft.com/fwlink/?LinkID=237468 -->
    <section name="entityFramework" type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework" />
  </configSections>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />
  </startup>
  <entityFramework>
    <defaultConnectionFactory type="System.Data.Entity.Infrastructure.SqlConnectionFactory, EntityFramework" />
    <providers>
      <provider invariantName="System.Data.SqlClient" type="System.Data.Entity.SqlServer.SqlProviderServices, EntityFramework.SqlServer" />
    </providers>
  </entityFramework>
  <connectionStrings>
    <add name="StudentModel" connectionString="data source=HUYCUONG\SQLEXPRESS;initial catalog=StudentLab05;persist security info=True;multiple active result sets=chained;" providerName="System.Data.SqlClient" />
  </connectionStrings>
</configuration>
```



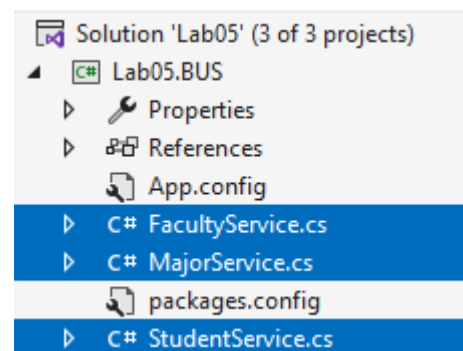
### Bước 3: Thực hiện ở tầng BUS

Implement các hàm cần sử dụng

( có thể đặt tên với các ý nghĩa xử lý logic ở tầng BUS này như: StudentDAO, MajorDAO, FacultyDAO...)

Một số hàm thực hiện theo yêu cầu đề bài:

- ✓ **StudentService**: Thực hiện các xử lý logic với Student như: Lấy danh sách sinh viên, Lấy ds sinh viên chưa đăng ký chuyên ngành, Tìm sinh viên theo ID, Thực hiện update cho cả insert/update...



```

public class StudentService
{
    2 references
    public List<Student> GetAll()
    {
        StudentModel context = new StudentModel();
        return context.Students.ToList();
    }
    1 reference
    public List<Student> GetAllHasNoMajor()
    {
        StudentModel context = new StudentModel();
        return context.Students.Where(p=>p.MajorID == null).ToList();
    }
    1 reference
    public List<Student> GetAllHasNoMajor(int facultyID)
    {
        StudentModel context = new StudentModel();
        return context.Students.Where(p => p.MajorID == null && p.FacultyID == facultyID).ToList();
    }
    1 reference
    public Student FindById(string studentId)
    {
        StudentModel context = new StudentModel();
        return context.Students.FirstOrDefault(p => p.StudentID == studentId);
    }
    1 reference
    public void InsertUpdate(Student s)
    {
        StudentModel context = new StudentModel();
        context.Students.AddOrUpdate(s);
        context.SaveChanges();
    }
}

```

- ✓ **MajorService:** Thực hiện các xử lý logic về chuyên ngành như: Lấy danh sách chuyên ngành từ CSDL,...

```

namespace Lab05.BUS
{
    2 references
    public class MajorService
    {
        1 reference
        public List<Major> GetAllByFaculty(int facultyID)
        {
            StudentModel context = new StudentModel();
            return context.Majors.Where(p=>p.FacultyID == facultyID).ToList();
        }
    }
}

```

- ✓ **FacultyService:** Thực hiện các xử lý logic về Khoa như: Lấy danh sách Khoa từ CSDL, thêm, xóa, sửa...

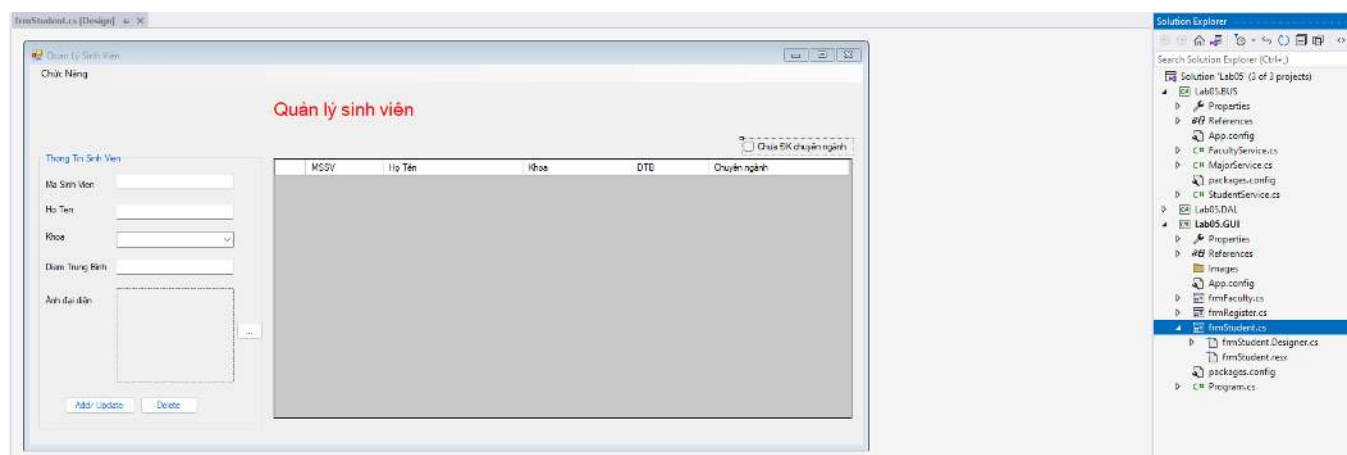
```

namespace Lab05.BUS
{
    4 references
    public class FacultyService
    {
        2 references
        public List<Faculty> GetAll()
        {
            StudentModel context = new StudentModel();
            return context.Faculties.ToList();
        }
    }
}

```

## Bước 4: Thực hiện ở GUI và điều chỉnh cần thiết

### 4.1 Quản lý sinh viên



- ✓ Load Form: Lấy ds sinh viên từ CSDL để fill vào DataGridView, các thông tin Khoa để đưa vào Combobox

```

public partial class frmStudent : Form
{
    private readonly StudentService studentService = new StudentService();
    private readonly FacultyService facultyService = new FacultyService();
    public frmStudent()
    {
        InitializeComponent();
    }

    private void Form1_Load(object sender, EventArgs e)
    {
        try
        {
            setGridViewStyle(dgvStudent);
            var listFacultys = facultyService.GetAll();
            var listStudents = studentService.GetAll();
            FillFacultyCombobox(listFacultys);
            BindGrid(listStudents);
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
}

```

```

    }
    //Hàm binding list dữ liệu khoa vào combobox có tên hiển thị là tên
    khoa, giá trị là Mã khoa
    private void FillFacultyComboBox(List<Faculty> listFacultys)
    {
        listFacultys.Insert(0, new Faculty());
        this.cmbFaculty.DataSource = listFacultys;
        this.cmbFaculty.DisplayMember = "FacultyName";
        this.cmbFaculty.ValueMember = "FacultyID";
    }
    //Hàm binding gridView từ list sinh viên
    private void BindGrid(List<Student> listStudent)
    {
        dgvStudent.Rows.Clear();
        foreach (var item in listStudent)
        {
            int index = dgvStudent.Rows.Add();
            dgvStudent.Rows[index].Cells[0].Value = item.StudentID;
            dgvStudent.Rows[index].Cells[1].Value = item.FullName;
            if (item.Faculty != null)
                dgvStudent.Rows[index].Cells[2].Value =
item.Faculty.FacultyName;
            dgvStudent.Rows[index].Cells[3].Value = item.AverageScore +
"";

            if (item.MajorID != null)
                dgvStudent.Rows[index].Cells[4].Value = item.Major.Name +
"";

            ShowAvatar(item.Avatar);
        }
    }
    private void ShowAvatar(string ImageName)
    {
        if (string.IsNullOrEmpty(ImageName))
        {
            picAvatar.Image = null;
        }
        else
        {
            string parentDirectory =
Directory.GetParent(AppDomain.CurrentDomain.BaseDirectory).Parent.Parent.FullName;
            string imagePath = Path.Combine(parentDirectory, "Images",
ImageName);
            picAvatar.Image = Image.FromFile(imagePath);
            picAvatar.Refresh();
        }
    }

    public void setGridViewStyle(DataGridView dgvview)
    {
        dgvview.BorderStyle = BorderStyle.None;
        dgvview.DefaultCellStyle.SelectionBackColor = Color.DarkTurquoise;
        dgvview.CellBorderStyle =
DataGridViewCellStyle.SingleHorizontal;
        dgvview.BackgroundColor = Color.White;
        dgvview.SelectionMode = DataGridViewSelectionMode.FullRowSelect;
    }

```

- ✓ Đăng ký sự kiện **chkUnregisterMajor\_CheckedChanged** ở checkbox cho việc lọc các sinh viên chưa đăng ký chuyên ngành

```
private void chkUnregisterMajor_CheckedChanged(object sender, EventArgs e)
{
    var listStudents = new List<Student>();
    if (this.chkUnregisterMajor.Checked)
        listStudents = studentService.GetAllHasNoMajor();
    else
        listStudents = studentService.GetAll();
    BindGrid(listStudents);
}
```

## 4.2 Đăng ký chuyên ngành

Sau khi chọn khoa

- Hiển thị ds sinh viên của Khoa mà chưa có chuyên ngành  
( MajorId = NULL trong Student)
- Hiển thị chuyên ngành trong Khoa ( mục đích để đăng ký)
- Việc đăng ký sinh viên khi checked vào danh sách sinh viên ở DataGridView

Hướng dẫn:

- ✓ Form Load: Đổ dữ liệu vào Khoa

```
public partial class frmRegister : Form
{
    private readonly StudentService studentService = new StudentService();
    private readonly FacultyService facultyService = new FacultyService();
    private readonly MajorService majorService = new MajorService();

    public frmRegister()
```

```
{
    InitializeComponent();
}

private void frmRegister_Load(object sender, EventArgs e)
{
    try
    {
        var listFacultys = facultyService.GetAll();
        FillFalculyCombobox(listFacultys);
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}

//Hàm binding list dữ liệu khoa vào combobox có tên hiển thị là tên khoa, giá
trị là Mã khoa
private void FillFalculyCombobox(List<Faculty> listFacultys)
{
    this.cmbFaculty.DataSource = listFacultys;
    this.cmbFaculty.DisplayMember = "FacultyName";
    this.cmbFaculty.ValueMember = "FacultyID";
}
}
```

- ✓ cmbFaculty\_SelectedIndexChanged: Khi thay đổi index lựa chọn cho Khoa



```

private void cmbFaculty_SelectedIndexChanged(object sender, EventArgs e)
{
    Faculty selectedFaculty = cmbFaculty.SelectedItem as Faculty;
    if(selectedFaculty != null)
    {
        var listMajor = majorService.GetAllByFaculty(selectedFaculty.FacultyID);
        FillMajorCombobox(listMajor);
        var listStudents = studentService.GetAllHasNoMajor(selectedFaculty.FacultyID);
        BindGrid(listStudents);
    }
}
1 reference
private void BindGrid(List<Student> listStudent)
{
    dgvStudent.Rows.Clear();
    foreach (var item in listStudent)
    {
        int index = dgvStudent.Rows.Add();
        dgvStudent.Rows[index].Cells[1].Value = item.StudentID;
        dgvStudent.Rows[index].Cells[2].Value = item.FullName;
        if (item.Faculty != null)
            dgvStudent.Rows[index].Cells[3].Value = item.Faculty.FacultyName;
        dgvStudent.Rows[index].Cells[4].Value = item.AverageScore + "";
        if (item.MajorID != null)
            dgvStudent.Rows[index].Cells[5].Value = item.Major.Name + "";
    }
}

```

Còn lại SV tự thực hiện để có thể đăng ký được chuyên ngành có các sinh viên được chọn (checked) ở DataGridView.

# BÀI 6: ÔN TẬP VÀ KIỂM TRA

Sử dụng MS SQLServer tạo CSDL tên *QLSach* với 2 bảng sau:

LoaiSach ( **MaLoai** INT, TenLoai nvarchar(50) )

Sach( **MaSach** char(6) , TenSach nvarchar(150), NamXB INT, MaLoai INT)

DESKTOP-AA3CGL7...h - dbo.LoaiSach			
Column Name	Data Type	Allow Nulls	
MaLoai	int	<input type="checkbox"/>	
TenLoai	nvarchar(50)	<input type="checkbox"/>	

DESKTOP-AA3CGL7.QLSach - dbo.Sach			
Column Name	Data Type	Allow Nulls	
MaSach	char(6)	<input type="checkbox"/>	
TenSach	nvarchar(200)	<input type="checkbox"/>	
NamXB	int	<input type="checkbox"/>	
MaLoai	int	<input type="checkbox"/>	

Mã Loại: FK tới LoaiSach.MaLoai

2.1 Thiết kế CSDL và Nhập 1 số dữ liệu vào sẵn như sau (1đ)

DESKTOP-AA3CGL7...h - dbo.LoaiSach		
MaLoai	TenLoai	
1	Khoa Học	
2	Đời sống	
3	Y học	

DESKTOP-AA3CGL7.QLSach - dbo.Sach				
MaSach	TenSach	NamXB	MaLoai	
KH0001	Khám phá sự sống	2018	1	
KH0002	Hải dương học	2018	1	
YH0001	Chẩn đoán và điều trị	2020	3	

2.2 Thiết kế giao diện quản lý sách tương tự như sau (1đ)

	Mã Sách	Tên Sách	Năm XB	Thể loại
▶	KH0001	Khám phá sự sống	2018	Khoa Học
	KH0002	Hải dương học	2018	Khoa Học
	YH0001	Chẩn đoán và điều trị	2020	Y học
	YH0002	Dược Học Cổ Truyền	2019	Y học

### 2.3 Khi Load Form (1.5 đ)

- Hiển thị các giá trị Thể loại sách vào Combobox (lấy dữ liệu ở bảng LoaiSach và hiển thị TenLoai) (1đ)
- Hiển thị các sách tìm thấy bên phải (Lấy dữ liệu từ bảng Sách và Loại Sách) (0.5đ)

### 2.4 Chọn 1 dòng ở DataGridView, hiển thị lại thông tin sách phía bên trái (1đ)

### 2.5 Khi Click vào nút xóa (1đ)

- Nếu mã sách muốn xóa đã tồn tại trong CSDL, Hiển thị cảnh báo YES/NO “**Bạn có muốn xóa không?**” (0.25đ). Nhấn YES: Xóa dữ liệu sách đã chọn (0.25đ) và cập nhật lại DataGridView (0.25đ)

- Ngược lại: Thông báo “**Sách cần xóa không tồn tại!**” (0.25)

### 2.6 Khi Click vào nút thêm mới / sửa (3đ)

- Kiểm tra tất cả thông tin bắt buộc phải nhập cho sách. Nếu không xuất hiện thông báo “**Vui lòng nhập đầy đủ thông tin sách!**” (0.5đ)
- Kiểm tra số kí tự mã sách vừa nhập phải là 6. Nếu không xuất thông báo “**Mã sách phải có 6 kí tự!**”. (0.5đ)
- Thêm mới / Cập nhật dữ liệu nhập vào CSDL (1đ).

- Xuất thông báo “*thêm mới/ cập nhật thành công*” và cập nhật lại DataGridView (0.5d)

- Reset các thông tin nhập liệu sách. Giá trị mặc định như lúc Load Form ban đầu (0.5đ – các control nhập liệu bằng Empty, Combobox thể hiện thông tin các loại sách)

2.7 Tìm kiếm theo mã sách, tên sách hoặc năm xuất bản nếu có chứa chuỗi tìm kiếm vừa nhập liệu (0.5 đ)

Mã Sách	Tên Sách	Năm XB	Thể loại
KH0002	Hải dương học	2018	Khoa Học
YH0002	Được Học Cổ Truyền	2019	Y học

2.8 Tạo menuStrip thống kê có submenu là “Thống kê sách theo năm” (1đ)

- Tạo thêm form mới có chứa **ReportViewer** để chứa báo cáo thống kê tương tự sau.

- Khi người dùng click vào menu xem báo cáo thống kê hoặc Ctrl + P, sẽ xuất danh sách thông tin danh sách các sách được *sắp xếp giảm dần* năm XB trên **ReportViewer**.

The screenshot shows a ReportViewer window with the title 'Thống kê'. The table displays the following data:

Năm XB	Mã Sách	Tên Sách	Mã Loại
2020	YH0001	Chuẩn đoán và điều trị	3
2019	YH0002	Dược Học Cổ Truyền	3
2018	KH0001	Khám phá sự sống	1
2018	KH0002	Hải dương học	1

Điểm cộng (0.5đ): khi thể hiện được ***Tên Loại*** (thay vì mã loại) trong Report.

-----HẾT-----