Xây dựng phân phối chương trình

# 1. Cơ sở dữ liệu

## 1.1. Bảng

Tạo thêm bảng LessonSub

| CREATE TABLE LessonSub (  Id\_Les INT PRIMARY KEY IDENTITY(1,1),  Les\_Unit NVARCHAR(10),  Les\_Name NVARCHAR(200),  FK\_Id\_Sub INT,  FK\_Id\_LS INT,  NumHour INT DEFAULT 0,  FOREIGN KEY (FK\_Id\_Sub) REFERENCES Subjects(Id\_Sub),  FOREIGN KEY (FK\_Id\_LS) REFERENCES LearningStyle(Id\_LS)  ); |
| --- |

## 1.2. View

### 1.2.1. Quản lý phân phối chương trình học

| CREATE VIEW LessonSubjectView AS  WITH SourceData AS (  SELECT  Subjects.Sym\_Sub,  Les\_Unit,  Les\_Name,  Name\_LS,  NumHour  FROM  LessonSub  JOIN  Subjects ON LessonSub.FK\_Id\_Sub = Subjects.Id\_Sub  JOIN  LearningStyle ON LessonSub.FK\_Id\_LS = LearningStyle.Id\_LS  )  SELECT  Sym\_Sub AS "Ký hiệu môn học",  Les\_Unit AS "Bài học",  Les\_Name AS "Tên bài học",  [Lý thuyết / lý luận] AS "Lý thuyết",  [Bài tập / Thảo luận] AS "Bài tập",  [Thực hành] AS "Thực hành"  FROM  SourceData  PIVOT (  MAX(NumHour)  FOR Name\_LS IN ([Lý thuyết / lý luận], [Bài tập / Thảo luận], [Thực hành])  ) AS PivotTable;  GO |
| --- |

## 

### 1.2.2. Liệt kê những môn học chưa được phân phối chương trình

| CREATE VIEW UnassignedSubjectsView AS  SELECT s.Id\_Sub, s.Name\_Sub  FROM Subjects s LEFT JOIN LessonSub ls ON s.Id\_Sub = ls.FK\_Id\_Sub  WHERE ls.Id\_Les IS NULL;  GO |
| --- |

### 

## 1.3. Thủ tục

# 2. Mã nguồn chương trình

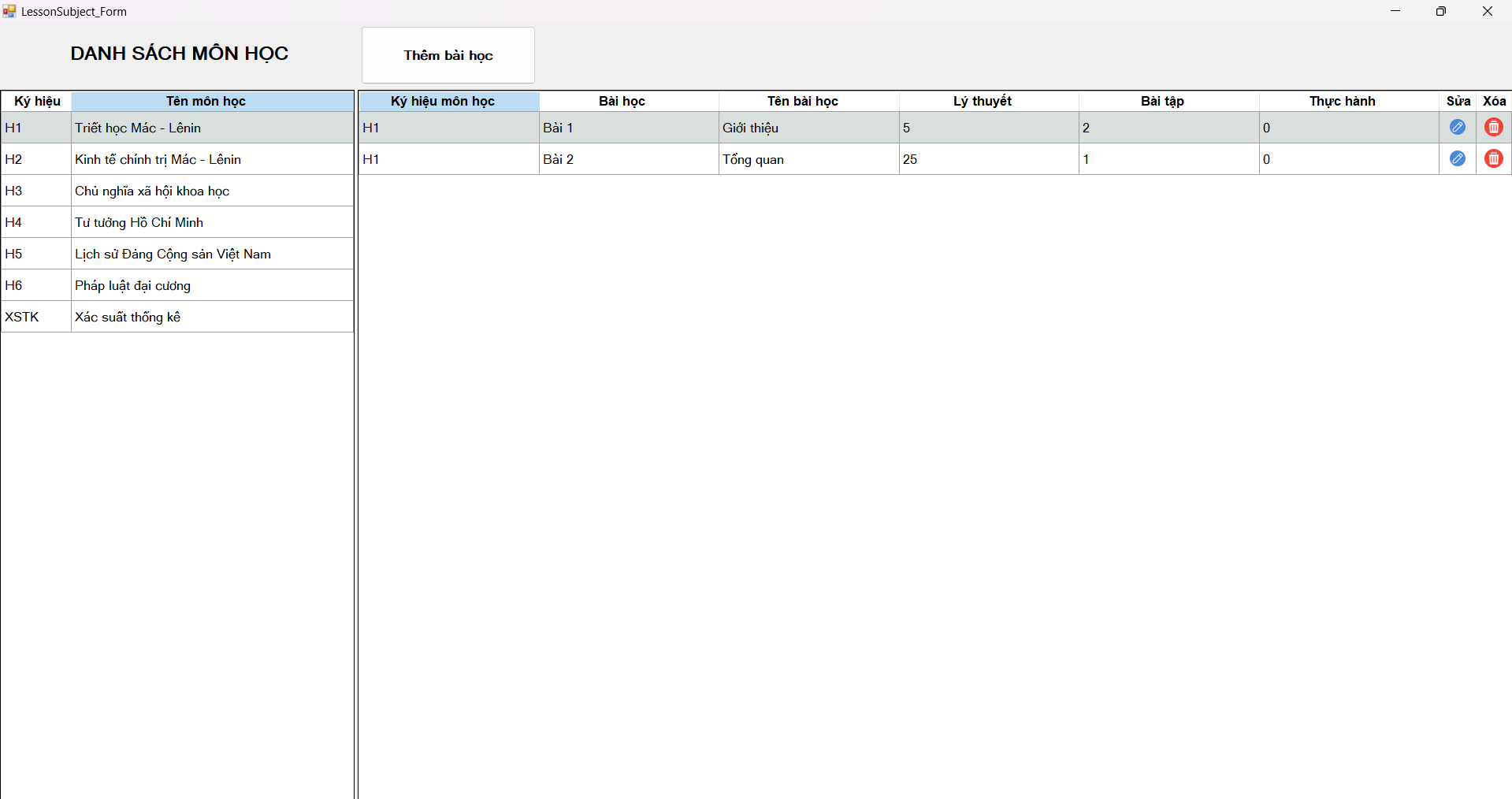
## 2.1. Model

Tạo mới Model **LessonSubject**

| namespace EduManager.Models  {  internal class LessonSubject  {  public int Id\_Les { get; set; }  public string Les\_Unit { get; set; }  public string Les\_Name { get; set; }  public int FK\_Id\_Sub { get; set; }  public int FK\_Id\_LS { get; set; }  public int NumHour { get; set; }  public LessonSubject() { }  public LessonSubject(string les\_Unit, string les\_Name, int fK\_Id\_Sub, int fK\_Id\_LS, int numHour)  {  Les\_Unit = les\_Unit;  Les\_Name = les\_Name;  FK\_Id\_Sub = fK\_Id\_Sub;  FK\_Id\_LS = fK\_Id\_LS;  NumHour = numHour;  }  }  } |
| --- |

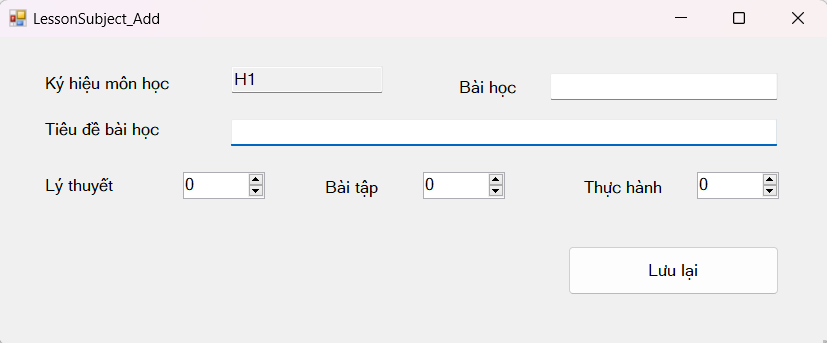
## 2.2. View

### 2.2.1. LessonSubject\_Form



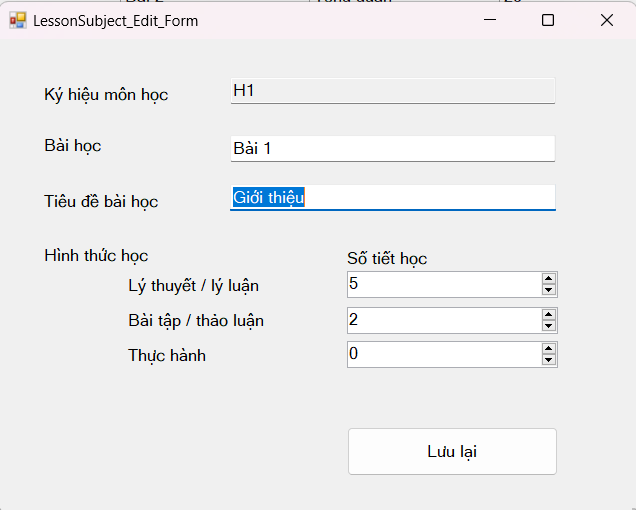
| using EduManager.Controllers;  using System;  using System.ComponentModel;  using System.Drawing;  using System.Windows.Forms;  namespace EduManager.Views  {  public partial class LessonSubject\_Form : Form  {  **// Khai báo dữ liệu**  private string **selectedSubjectSymbol** = string.Empty;  **// Khởi tạo form**  public **LessonSubject\_Form**()  {  InitializeComponent();  SetupDataGridViewDefaults();  }  **// Sự kiện khởi tạo form**  private void **LessonSubject\_Form\_Load**(object sender, EventArgs e)  {  LoadData();  ConfigureDataGridViewStyles();  }  **// Cài đặt mặc định cho DataGridView**  private void **SetupDataGridViewDefaults**()  {  dtgvLessonList.RowTemplate.Height = 40;  dtgvSubjectsList.RowTemplate.Height = 40;  dtgvLessonList.SelectionMode = DataGridViewSelectionMode.FullRowSelect;  dtgvSubjectsList.SelectionMode = DataGridViewSelectionMode.FullRowSelect;  }  **// Nhóm hàm thiết lập cấu hình cho DataGridView**  private void **ConfigureDataGridViewStyles**()  {  ConfigureDataGridView(dtgvLessonList, "#d8dfdc", "#000");  ConfigureDataGridView(dtgvSubjectsList, "#d8dfdc", "#000");  ConfigureColumnWeights();  ConfigureColumnHeaders();  }  private void **ConfigureDataGridView**(DataGridView dataGridView, string selectionBackColor, string selectionForeColor)  {  dataGridView.DefaultCellStyle.SelectionBackColor = ColorTranslator.FromHtml(selectionBackColor);  dataGridView.DefaultCellStyle.SelectionForeColor = ColorTranslator.FromHtml(selectionForeColor);  }  private void **ConfigureColumnWeights**()  {  dtgvSubjectsList.Columns["Ký hiệu"].FillWeight = 20;  dtgvSubjectsList.Columns["Tên môn học"].FillWeight = 80;  dtgvLessonList.Columns["Ký hiệu môn học"].FillWeight = 10;  dtgvLessonList.Columns["Bài học"].FillWeight = 10;  dtgvLessonList.Columns["Tên bài học"].FillWeight = 50;  dtgvLessonList.Columns["Lý thuyết"].FillWeight = 10;  dtgvLessonList.Columns["Bài tập"].FillWeight = 10;  dtgvLessonList.Columns["Thực hành"].FillWeight = 10;  dtgvLessonList.Sort(dtgvLessonList.Columns["Bài học"], ListSortDirection.Ascending);  }  private void **ConfigureColumnHeaders**()  {  var headerStyle = new DataGridViewCellStyle  {  Alignment = DataGridViewContentAlignment.MiddleCenter,  BackColor = SystemColors.GrayText,  ForeColor = SystemColors.WindowText,  Font = new Font("Arial", 10, FontStyle.Bold)  };  dtgvSubjectsList.ColumnHeadersDefaultCellStyle = headerStyle;  dtgvLessonList.ColumnHeadersDefaultCellStyle = headerStyle;  }  **// Quản lý dữ liệu**  public void **LoadData**()  {  var controller = LessonSubjectController.Instance();  controller.ShowSubjectsList(dtgvSubjectsList);  controller.ShowLessonList(dtgvLessonList, selectedSubjectSymbol);  }  **// Nhóm hàm xử lý sự kiện**  private void **dtgvSubjectsList\_CellClick**(object sender, DataGridViewCellEventArgs e)  {  if (e.RowIndex >= 0)  {  selectedSubjectSymbol = dtgvSubjectsList.Rows[e.RowIndex].Cells[0].Value?.ToString();  LessonSubjectController.Instance().ShowLessonList(dtgvLessonList, selectedSubjectSymbol);  }  }  private void **btnAddLessonSubject\_Click**(object sender, EventArgs e)  {  if (string.IsNullOrWhiteSpace(selectedSubjectSymbol))  {  MessageBox.Show("Vui lòng chọn một môn học", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);  return;  }  var addForm = new LessonSubject\_Add(this);  addForm.ShowDialog();  }  private void **dtgvLessonList\_CellClick**(object sender, DataGridViewCellEventArgs e)  {  if (e.RowIndex < 0) return;  var headerText = dtgvLessonList.Columns[e.ColumnIndex].HeaderText;  switch (headerText)  {  case "Xóa":  HandleDelete(e.RowIndex);  break;  case "Sửa":  HandleEdit(e.RowIndex);  break;  }  }  private void **HandleDelete**(int rowIndex)  {  var dgvRow = dtgvLessonList.Rows[rowIndex];  var lessonSymbol = dgvRow.Cells[1].Value?.ToString();  var lessonName = dgvRow.Cells[2].Value?.ToString();  var confirmation = MessageBox.Show(  $"Bạn có chắc chắn muốn xóa \"{lessonSymbol} - {lessonName}\"?",  "Xác nhận",  MessageBoxButtons.YesNo,  MessageBoxIcon.Question  );  if (confirmation == DialogResult.Yes)  {  var isDeleted = LessonSubjectController.Instance().DeleteLessonSubject(lessonSymbol);  if (isDeleted)  {  LoadData();  MessageBox.Show("Xóa bài học thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  else  {  MessageBox.Show("Xóa bài học thất bại.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Error);  }  }  }  private void **HandleEdit**(int rowIndex)  {  var dgvRow = dtgvLessonList.Rows[rowIndex];  var lessonSymbol = dgvRow.Cells[1].Value?.ToString();  var lessonName = dgvRow.Cells[2].Value?.ToString();  var lt = Convert.ToInt32(dgvRow.Cells[3].Value);  var bt = Convert.ToInt32(dgvRow.Cells[4].Value);  var th = Convert.ToInt32(dgvRow.Cells[5].Value);  var editForm = new LessonSubject\_Edit\_Form(this, lessonSymbol, lessonName, lt, bt, th);  editForm.ShowDialog();  }  **// Nhóm hàm hỗ trợ**  public string **GetSym\_Sub**()  {  return selectedSubjectSymbol;  }  }  } |
| --- |

### 2.2.2. LessonSubject-Add\_Form



| using System;  using System.Data;  using System.Linq;  using System.Windows.Forms;  using EduManager.Controllers;  using EduManager.Models;  namespace EduManager.Views  {  public partial class LessonSubject\_Add : Form  {  private readonly LessonSubject\_Form \_lessonSubjectForm;  public LessonSubject\_Add(LessonSubject\_Form lessonSubjectForm)  {  InitializeComponent();  \_lessonSubjectForm = lessonSubjectForm;  }  **// ==== Sự kiện (Event Handlers) ====**  private void **LessonSubject\_Add\_Load**(object sender, EventArgs e)  {  txbSym\_Sub.Text = \_lessonSubjectForm.GetSym\_Sub();  }  private void **btnSave\_Click**(object sender, EventArgs e)  {  string subjectSymbol = txbSym\_Sub.Text;  int subjectId = LessonSubjectController.Instance().GetIDSub(subjectSymbol);  if (IsOverLimit(subjectSymbol))  {  return; // Dừng lại nếu quá giới hạn  }  string lesson = txbLesson.Text;  string lessonName = txbLessonName.Text;  AddLessonSubjects(subjectId, lesson, lessonName);  \_lessonSubjectForm.LoadData(); // Cập nhật dữ liệu giao diện  MessageBox.Show("Thêm thành công", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  **// ==== Xử lý dữ liệu (Data Processing) ====**  private bool **IsOverLimit**(string subjectSymbol)  {  DataTable lessonData = LessonSubjectController.Instance().ShowLessonList(subjectSymbol);  int totalLyThuyet = CalculateColumnSum(lessonData, "Lý thuyết");  int totalBaiTap = CalculateColumnSum(lessonData, "Bài tập");  int totalThucHanh = CalculateColumnSum(lessonData, "Thực hành");  int allowedLyThuyet = LessonSubjectController.Instance().GetNumHour(subjectSymbol, "Lý thuyết");  int allowedBaiTap = LessonSubjectController.Instance().GetNumHour(subjectSymbol, "Bài tập");  int allowedThucHanh = LessonSubjectController.Instance().GetNumHour(subjectSymbol, "Thực hành");  if (totalLyThuyet + nmLT.Value > allowedLyThuyet)  {  DisplayWarning("Lý thuyết", allowedLyThuyet);  return true;  }  if (totalBaiTap + nmBT.Value > allowedBaiTap)  {  DisplayWarning("Bài tập", allowedBaiTap);  return true;  }  if (totalThucHanh + nmTH.Value > allowedThucHanh)  {  DisplayWarning("Thực hành", allowedThucHanh);  return true;  }  return false; // Không vượt quá giới hạn  }  private void **AddLessonSubjects**(int subjectId, string lesson, string lessonName)  {  var lessonTypes = new[] { "Lý thuyết", "Bài tập", "Thực hành" };  var numHours = new[] { nmLT.Value, nmBT.Value, nmTH.Value };  for (int i = 0; i < lessonTypes.Length; i++)  {  int numHour = Convert.ToInt32(numHours[i]);  LessonSubject newLessonSubject = new LessonSubject(lesson, lessonName, subjectId, i + 1, numHour);  LessonSubjectController.Instance().AddLessonSubject(newLessonSubject);  }  }  **// ==== Giao diện người dùng (UI) ====**  private void **DisplayWarning**(string subjectType, int limit)  {  MessageBox.Show(  $"Số giờ {subjectType} đã vượt quá {limit} tiết trong Chương trình đào tạo",  "Thông báo",  MessageBoxButtons.OK,  MessageBoxIcon.Warning  );  }  **// ==== Hàm tiện ích (Utility Functions) ====**  public static int **CalculateColumnSum**(DataTable dataTable, string columnName)  {  if (!dataTable.Columns.Contains(columnName))  {  throw new ArgumentException($"Cột '{columnName}' không tồn tại trong bảng dữ liệu.");  }  return dataTable.AsEnumerable()  .Where(row => row[columnName] != DBNull.Value)  .Sum(row => Convert.ToInt32(row[columnName]));  }  }  } |
| --- |

### 2.2.3. LessonSubject-Edit\_Form



| using EduManager.Controllers;  using EduManager.Models;  using System;  using System.Collections.Generic;  using System.Data;  using System.Linq;  using System.Windows.Forms;  namespace EduManager.Views  {  public partial class LessonSubject\_Edit\_Form : Form  {  private readonly LessonSubject\_Form \_lessonSubjectForm;  private int \_ltOldValue, \_btOldValue, \_thOldValue;  **// === Nhóm hàm khởi tạo và thiết lập ===**  public **LessonSubject\_Edit\_Form**(LessonSubject\_Form lessonSubjectForm, string lesson, string lessonName, int lt, int bt, int th)  {  InitializeComponent();  \_lessonSubjectForm = lessonSubjectForm;  SetInitialValues(lesson, lessonName, lt, bt, th);  }  private void **SetInitialValues**(string lesson, string lessonName, int lt, int bt, int th)  {  txbSym\_Sub.Text = \_lessonSubjectForm.GetSym\_Sub();  txbLesson.Text = lesson;  txbLessonName.Text = lessonName;  nmLT.Value = lt;  nmBT.Value = bt;  nmTH.Value = th;  }  private void **LessonSubject\_Edit\_Form\_Load**(object sender, EventArgs e)  {  \_ltOldValue = (int)nmLT.Value;  \_btOldValue = (int)nmBT.Value;  \_thOldValue = (int)nmTH.Value;  }  **// === Nhóm hàm xử lý sự kiện ===**  private void **btnSave\_Click**(object sender, EventArgs e)  {  var lessonUpdates = CreateLessonUpdates();  if (ValidateHours(lessonUpdates))  {  bool success = lessonUpdates.All(LessonSubjectController.Instance().EditLessonSubject);  if (success)  {  \_lessonSubjectForm.LoadData();  ShowMessage("Sửa thành công", "Thông báo", MessageBoxIcon.Information);  }  }  }  **// === Nhóm hàm tiện ích ===**  private List<LessonSubject> **CreateLessonUpdates**()  {  string lessonName = txbLessonName.Text;  string unit = txbLesson.Text;  string symSub = txbSym\_Sub.Text;  int subId = LessonSubjectController.Instance().GetIDSub(symSub);  return new List<LessonSubject>  {  new LessonSubject { Les\_Name = lessonName, Les\_Unit = unit, FK\_Id\_Sub = subId, FK\_Id\_LS = 1, NumHour = (int)nmLT.Value },  new LessonSubject { Les\_Name = lessonName, Les\_Unit = unit, FK\_Id\_Sub = subId, FK\_Id\_LS = 2, NumHour = (int)nmBT.Value },  new LessonSubject { Les\_Name = lessonName, Les\_Unit = unit, FK\_Id\_Sub = subId, FK\_Id\_LS = 3, NumHour = (int)nmTH.Value }  };  }  private bool **ValidateHours**(List<LessonSubject> lessonUpdates)  {  string symSub = txbSym\_Sub.Text;  DataTable lessonData = LessonSubjectController.Instance().ShowLessonList(symSub);  int maxLyThuyet = LessonSubjectController.Instance().GetNumHour(symSub, "Lý thuyết");  int maxBaiTap = LessonSubjectController.Instance().GetNumHour(symSub, "Bài tập");  int maxThucHanh = LessonSubjectController.Instance().GetNumHour(symSub, "Thực hành");  int currentLyThuyet = CalculateColumnSum(lessonData, "Lý thuyết");  int currentBaiTap = CalculateColumnSum(lessonData, "Bài tập");  int currentThucHanh = CalculateColumnSum(lessonData, "Thực hành");  bool isLTValid = (int)nmLT.Value - \_ltOldValue + currentLyThuyet <= maxLyThuyet;  bool isBTValid = (int)nmBT.Value - \_btOldValue + currentBaiTap <= maxBaiTap;  bool isTHValid = (int)nmTH.Value - \_thOldValue + currentThucHanh <= maxThucHanh;  return ValidateNewHours(isLTValid, isBTValid, isTHValid);  }  private static bool **ValidateNewHours**(bool isLTValid, bool isBTValid, bool isTHValid)  {  if (!isLTValid)  {  ShowWarning("Số giờ lý thuyết đã vượt quá giới hạn trong Chương trình đào tạo");  return false;  }  if (!isBTValid)  {  ShowWarning("Số giờ bài tập đã vượt quá giới hạn trong Chương trình đào tạo");  return false;  }  if (!isTHValid)  {  ShowWarning("Số giờ thực hành đã vượt quá giới hạn trong Chương trình đào tạo");  return false;  }  return true;  }  private static void **ShowWarning**(string message)  {  MessageBox.Show(message, "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);  }  private static void **ShowMessage**(string message, string title, MessageBoxIcon icon)  {  MessageBox.Show(message, title, MessageBoxButtons.OK, icon);  }  public static int **CalculateColumnSum**(DataTable dataTable, string columnName)  {  if (!dataTable.Columns.Contains(columnName))  {  throw new ArgumentException("Tên cột không hợp lệ");  }  return dataTable.AsEnumerable()  .Where(row => row[columnName] != DBNull.Value)  .Sum(row => Convert.ToInt32(row[columnName]));  }  }  } |
| --- |

## 2.3. Controller

| using EduManager.Models;  using EduManager.Services;  using System;  using System.Data;  using System.Data.SqlClient;  using System.Drawing;  using System.IO;  using System.Windows.Forms;  namespace EduManager.Controllers  {  internal class LessonSubjectController  {  private static LessonSubjectController instance;  private ConnectDatabase connectDatabase = ConnectDatabase.getInstance();  private LessonSubjectController() { }  **// Khởi tạo một instance mới nếu chưa tồn tại, ngược lại trả về instance hiện tại**  public static LessonSubjectController **Instance**()  {  if (instance == null)  {  instance = new LessonSubjectController();  }  return instance;  }  **// == Nhóm hàm thao tác CRUD (CREATE, READ, UPDATE, DELETE) ==**  public bool **AddLessonSubject**(LessonSubject leSub)  {  //les\_Unit, int les\_Name, int fK\_Id\_Sub, int fK\_Id\_LS, int numHour  string query = "INSERT INTO LessonSub(Les\_Unit, Les\_Name, FK\_Id\_Sub, FK\_Id\_LS, NumHour) VALUES (@Les\_Unit, @Les\_Name, @FK\_Id\_Sub, @FK\_Id\_LS, @NumHour)";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("Les\_Unit", leSub.Les\_Unit),  new SqlParameter("Les\_Name", leSub.Les\_Name),  new SqlParameter("FK\_Id\_Sub", leSub.FK\_Id\_Sub),  new SqlParameter("FK\_Id\_LS", leSub.FK\_Id\_LS),  new SqlParameter("NumHour", leSub.NumHour)  };  return connectDatabase.ExecuteNonQuery(query, para) > 0;  }  public bool **EditLessonSubject**(LessonSubject leSub)  {  string query = "UPDATE LessonSub SET Les\_Name = @Les\_Name, FK\_Id\_Sub = @FK\_Id\_Sub, FK\_Id\_LS = @FK\_Id\_LS, NumHour = @NumHour WHERE Les\_Unit = @Les\_Unit AND FK\_Id\_LS = @FK\_Id\_LS";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("Les\_Name", leSub.Les\_Name),  new SqlParameter("FK\_Id\_Sub", leSub.FK\_Id\_Sub),  new SqlParameter("FK\_Id\_LS", leSub.FK\_Id\_LS),  new SqlParameter("NumHour", leSub.NumHour),  new SqlParameter("Les\_Unit", leSub.Les\_Unit)  };  return connectDatabase.ExecuteNonQuery(query, para) > 0;  }  public bool **DeleteLessonSubject**(string lesson)  {  string query = $"DELETE FROM LessonSub WHERE Les\_Unit = '{lesson}'";  return connectDatabase.ExecuteNonQuery(query) > 0;  }  **// == Nhóm hàm đọc dữ liệu ==**  public void **ShowSubjectsList**(DataGridView dgv)  {  string query = "SELECT Sym\_Sub AS [Ký hiệu], Name\_Sub AS [Tên môn học] FROM Subjects ";  dgv.Columns.Clear();  dgv.DataSource = ConnectDatabase.getInstance().ExecuteQuery(query);  }  public void **ShowLessonList**(DataGridView dgv, String sym\_sub)  {  dgv.Columns.Clear();  string query = $"SELECT \* FROM LessonSubjectView WHERE \"Ký hiệu môn học\" = '{sym\_sub}'";  DataTable dataTable = ConnectDatabase.getInstance().ExecuteQuery(query);  dgv.DataSource = dataTable;  if (dgv.Columns["Edit"] == null) // Chỉ thêm nếu chưa tồn tại  {  string imagePath = Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "..", "..", "Resources", "ICON\_EDIT-24x24.png");  var deleteColumn = new DataGridViewImageColumn  {  Name = "Edit",  HeaderText = "Sửa",  Image = Image.FromFile(imagePath),  AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCells  };  dgv.Columns.Add(deleteColumn);  }  if (dgv.Columns["Delete"] == null) // Chỉ thêm nếu chưa tồn tại  {  string imagePath = Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "..", "..", "Resources", "ICON\_TRASH-24x24.png");  var deleteColumn = new DataGridViewImageColumn  {  Name = "Delete",  HeaderText = "Xóa",  Image = Image.FromFile(imagePath),  AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCells  };  dgv.Columns.Add(deleteColumn);  }  }  public DataTable **ShowLessonList**(String sym\_sub)  {  string query = $"SELECT \* FROM LessonSubjectView WHERE \"Ký hiệu môn học\" = '{sym\_sub}'";  DataTable dataTable = ConnectDatabase.getInstance().ExecuteQuery(query);  return dataTable;  }  **// == Nhóm hàm khác ==**  public int **GetIDSub**(string sym\_sub)  {  string query = $"SELECT Id\_Sub FROM Subjects WHERE Sym\_Sub = '{sym\_sub}'";  string result = ConnectDatabase.getInstance().GetValue(query);  return Convert.ToInt32(result);  }  public int **GetNumHour**(String sym\_sub, String learningStyle)  {  string query = $"SELECT [{learningStyle}] FROM EduProgramView WHERE [Ký hiệu] = '{sym\_sub}'";  string result = ConnectDatabase.getInstance().GetValue(query);  return Convert.ToInt32(result);  }  }  } |
| --- |

## 