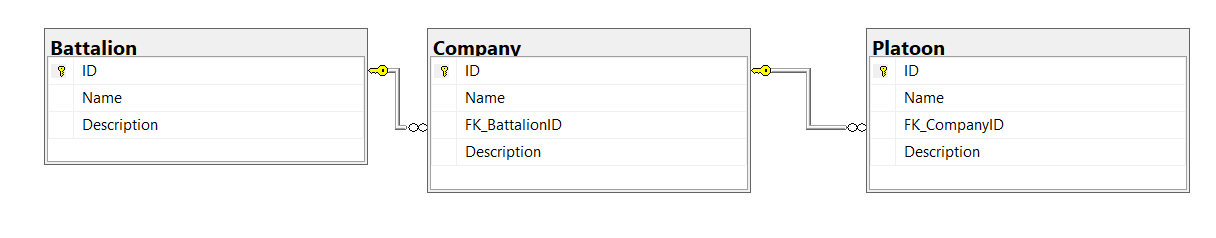
Quản lý đơn vị quân đội

# 1. Models

## 1.1. Mô hình quan hệ



## 1.2. Mã nguồn SQL

| CREATE DATABASE MilitaryUnitManagement;  GO  USE MilitaryUnitManagement;  GO  CREATE TABLE Battalion (  ID INT PRIMARY KEY IDENTITY(1,1),  Name NVARCHAR(255) NOT NULL,  Description NTEXT  );  GO  CREATE TABLE Company (  ID INT PRIMARY KEY IDENTITY(1,1),  Name NVARCHAR(255) NOT NULL,  FK\_BattalionID INT,  Description NTEXT,  FOREIGN KEY (FK\_BattalionID) REFERENCES Battalion(ID) ON DELETE CASCADE  );  GO  CREATE TABLE Platoon (  ID INT PRIMARY KEY IDENTITY(1,1),  Name NVARCHAR(255) NOT NULL,  FK\_CompanyID INT,  Description NTEXT,  FOREIGN KEY (FK\_CompanyID) REFERENCES Company(ID) ON DELETE CASCADE  );  GO  INSERT INTO Battalion (Name, Description) VALUES  (N'Tiểu đoàn 1', N'Mô tả Tiểu đoàn 1'),  (N'Tiểu đoàn 2', N'Mô tả Tiểu đoàn 2'),  (N'Tiểu đoàn 3', N'Mô tả Tiểu đoàn 3');  GO  INSERT INTO Company (Name, FK\_BattalionID, Description) VALUES  (N'Đại đội A', 1, N'Đại đội đầu tiên của Tiểu đoàn 1'),  (N'Đại đội B', 1, N'Đại đội thứ hai của Tiểu đoàn 1'),  (N'Đại đội C', 2, N'Đại đội đầu tiên của Tiểu đoàn 2'),  (N'Đại đội D', 3, N'Đại đội đầu tiên của Tiểu đoàn 3');  GO  INSERT INTO Platoon (Name, FK\_CompanyID, Description) VALUES  (N'Trung đội Alpha', 1, N'Trung đội đầu tiên của Đại đội A'),  (N'Trung đội Beta', 1, N'Trung đội thứ hai của Đại đội A'),  (N'Trung đội Gamma', 2, N'Trung đội đầu tiên của Đại đội B'),  (N'Trung đội Delta', 3, N'Trung đội đầu tiên của Đại đội C'),  (N'Trung đội Epsilon', 4, N'Trung đội đầu tiên của Đại đội D');  GO |
| --- |

## 1.3. Mã nguồn C#

### 1.3.1. Battalion

| namespace MilitaryUnitManagement.Models  {  public class Battalion  {  public int ID { get; set; }  public string Name { get; set; }  public string Description { get; set; }  public Battalion()  {  }  public Battalion(int id)  {  ID = id;  }  public Battalion(int id, string name, string description)  {  ID = id;  Name = name;  Description = description;  }  }  } |
| --- |

### 1.3.2. Company

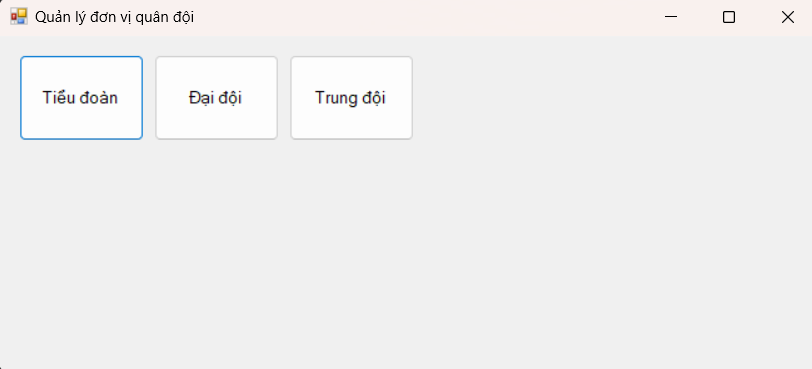
| namespace MilitaryUnitManagement.Models  {  public class Company  {  public int ID { get; set; }  public string Name { get; set; }  public int FK\_BattalionID { get; set; }  public string Description { get; set; }  public Company() { }  public Company(int id)  {  ID = id;  }  public Company(int id, string name, string description, int fkBattalionID)  {  ID = id;  Name = name;  Description = description;  FK\_BattalionID = fkBattalionID;  }  }  } |
| --- |

### 1.3.3. Platoon

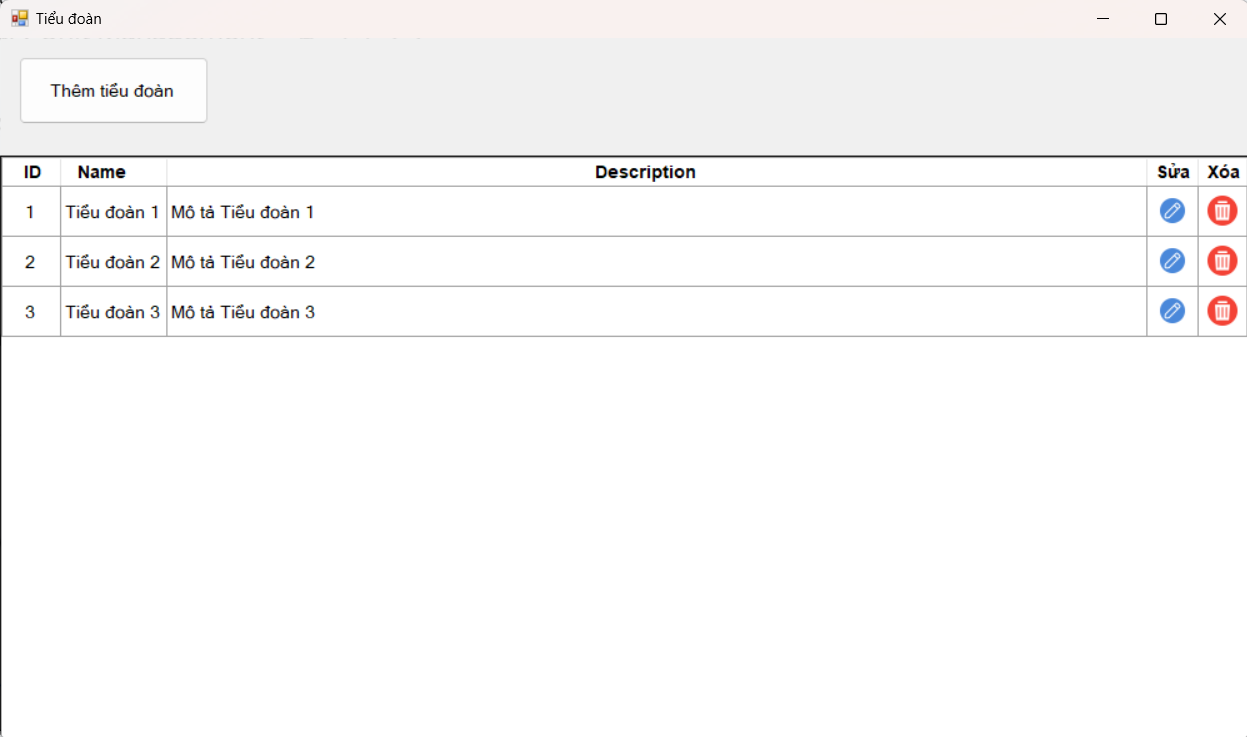
| namespace MilitaryUnitManagement.Models  {  public class Platoon  {  public int ID { get; set; }  public string Name { get; set; }  public int FK\_CompanyID { get; set; }  public string Description { get; set; }  public Platoon() { }  public Platoon(int id)  {  ID = id;  }  public Platoon(int id, string name, string description, int fkCompanyID)  {  ID = id;  Name = name;  Description = description;  FK\_CompanyID = fkCompanyID;  }  }  } |
| --- |

### 

# 2. Views

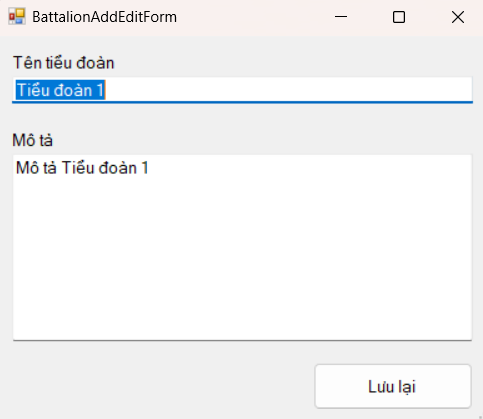


## 2.1. BattalionForm



| using MilitaryUnitManagement.Models;  using MilitaryUnitManagement.Controllers;  using MilitaryUnitManagement.Services;  using System;  using System.Windows.Forms;  namespace MilitaryUnitManagement.Views  {  public partial class BattalionForm : Form  {  private ConfigView configView = new ConfigView(); // Khai báo biến static  public BattalionForm()  {  InitializeComponent();  dtgv.RowTemplate.Height = 40;  }  // Load dữ liệu  private void BattalionForm\_Load(object sender, EventArgs e)  {  LoadData();  }  public void LoadData()  {  BattalionController.Instance().ShowData(dtgv);  configView.ConfigureDataGridViewReadOnly(dtgv, "ID");  configView.ConfigureColumnHeaders(dtgv);  configView.ConfigureColumnAlignment(dtgv, new string[] { "ID", "Name" });  configView.AddActionColumns(dtgv);  configView.ConfigureSelectionMode(dtgv);  }  #region EVENT HANDLER  private void btnAdd\_Click(object sender, EventArgs e)  {  BattalionAddEditForm f = new BattalionAddEditForm();  f.ShowDialog();  LoadData();  }  private void dtgv\_CellClick(object sender, DataGridViewCellEventArgs e)  {  if (e.RowIndex < 0) return; // Bỏ qua khi click vào tiêu đề  var dgv = sender as DataGridView;  var headerText = dgv.Columns[e.ColumnIndex].HeaderText;  if (headerText == "Xóa")  {  HandleDelete(dgv, e.RowIndex);  }  else if (headerText == "Sửa")  {  HandleEdit(dgv, e.RowIndex);  }  }  private void HandleDelete(DataGridView dgv, int rowIndex)  {  int Id = int.Parse(dgv.Rows[rowIndex].Cells[0].Value.ToString());  var Name = dgv.Rows[rowIndex].Cells[1].Value.ToString();  // Xác nhận trước khi xóa  var confirmation = MessageBox.Show(  $"Bạn có chắc chắn muốn xóa \"{Name}\"? Thao tác này sẽ xóa toàn bộ các Đại đội của của tiểu đoàn này (nếu có).",  "Xác nhận",  MessageBoxButtons.YesNo,  MessageBoxIcon.Question  );  if (confirmation == DialogResult.Yes)  {  if (BattalionController.Instance().RemoveData(new Battalion(Id)))  {  LoadData(); // Tải lại dữ liệu  MessageBox.Show("Xóa tiểu đoàn thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  else  {  MessageBox.Show("Xóa tiểu đoàn thất bại.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Error);  }  }  }  private void HandleEdit(DataGridView dgv, int rowIndex)  {  var Id = int.Parse(dgv.Rows[rowIndex].Cells[0].Value.ToString());  var Name = dgv.Rows[rowIndex].Cells[1].Value.ToString();  var Description = dgv.Rows[rowIndex].Cells[2].Value.ToString();  //if (SubjectController.Instance().checkDuplicate(subjectSymbol, subjectId, subjectName) > 0)  //{  // MessageBox.Show("Ký hiệu hoặc tên môn học đã tồn tại!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Error);  // dtgvEduProgram[columnIndex, rowIndex].Value = previousValue; // Khôi phục giá trị trước đó  // return;  //}  BattalionAddEditForm f = new BattalionAddEditForm(new Battalion(Id, Name, Description));  f.ShowDialog();  LoadData();  }  #endregion  }  } |
| --- |

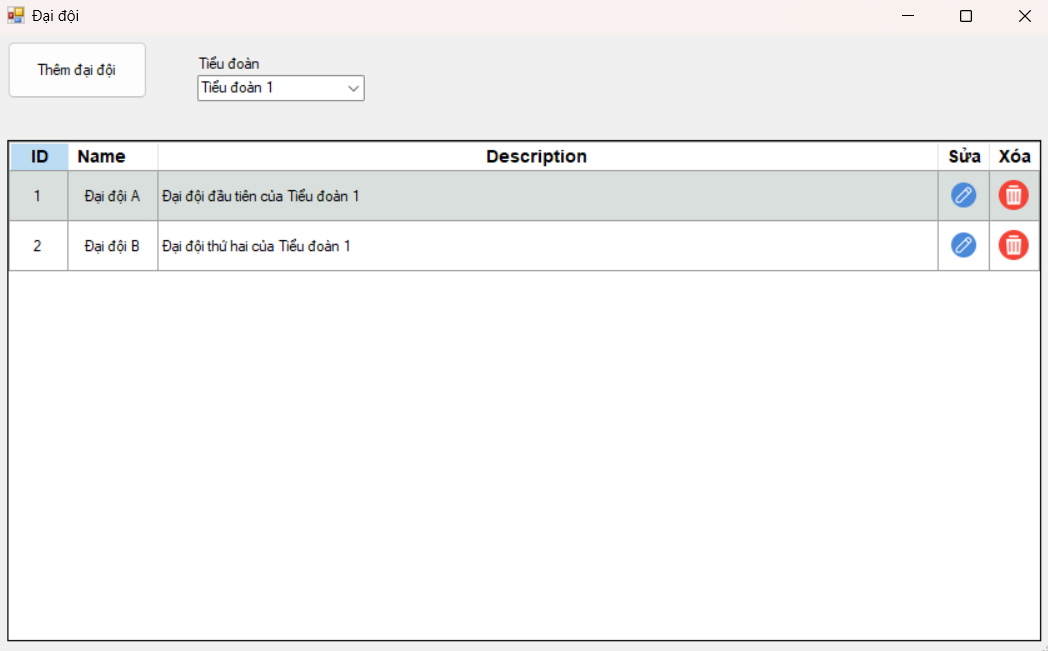
## 2.2. BattalionAddEditForm



| using MilitaryUnitManagement.Controllers;  using MilitaryUnitManagement.Models;  using System;  using System.Windows.Forms;  namespace MilitaryUnitManagement.Views  {  public partial class BattalionAddEditForm : Form  {  private bool isEdit = false;  private Battalion battalionEdit = new Battalion();  public BattalionAddEditForm()  {  InitializeComponent();  }  public BattalionAddEditForm(Battalion battalion)  {  InitializeComponent();  isEdit = true;  battalionEdit.ID = battalion.ID;  txbNameBattalion.Text = battalion.Name;  txbDescription.Text = battalion.Description;  }  private void btnSave\_Click(object sender, EventArgs e)  {  if (!isEdit)  {  Battalion battalion = new Battalion();  battalion.Name = txbNameBattalion.Text;  battalion.Description = txbDescription.Text;  if (BattalionController.Instance().AddData(battalion))  {  MessageBox.Show("Thêm tiểu đoàn thành công!");  this.Close();  }  else  {  MessageBox.Show("Thêm tiểu đoàn thất bại!");  }  }  else  {  battalionEdit.Name = txbNameBattalion.Text;  battalionEdit.Description = txbDescription.Text;  if (BattalionController.Instance().EditData(battalionEdit))  {  MessageBox.Show("Sửa tiểu đoàn thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);  this.Close();  }  else  {  MessageBox.Show("Sửa tiểu đoàn thất bại.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Error);  }  }  }  }  } |
| --- |

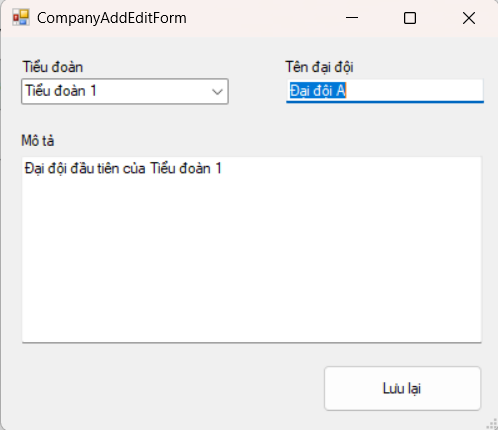
## 

## 2.3. CompanyForm



| using MilitaryUnitManagement.Controllers;  using MilitaryUnitManagement.Models;  using MilitaryUnitManagement.Services;  using System;  using System.Data;  using System.Windows.Forms;  namespace MilitaryUnitManagement.Views  {  public partial class CompanyForm : Form  {  private ConfigView configView;  private int selectedValue;  private string selectedText;  public CompanyForm()  {  InitializeComponent();  configView = new ConfigView(); // Initialize ConfigView instance  dtgv.RowTemplate.Height = 40;  }  private void CompanyForm\_Load(object sender, EventArgs e)  {  CompanyController.Instance().LoadComboBoxData(cb);  }  private void loadData(int ID)  {  CompanyController.Instance().ShowData(dtgv, ID);  // Use the ConfigView instance to configure the DataGridView  configView.ConfigureDataGridViewReadOnly(dtgv, "ID");  configView.ConfigureColumnHeaders(dtgv);  configView.ConfigureColumnAlignment(dtgv, new string[] { "ID", "Name" });  configView.AddActionColumns(dtgv);  configView.ConfigureSelectionMode(dtgv);  }  private void cb\_SelectedIndexChanged(object sender, EventArgs e)  {  // Kiểm tra xem selectedItem có phải là DataRowView hay không  DataRowView selectedItem = cb.SelectedItem as DataRowView;  if (selectedItem != null)  {  // Trích xuất giá trị ID từ DataRowView  if (int.TryParse(selectedItem["ID"].ToString(), out selectedValue))  {  // Lấy tên từ DataRowView  selectedText = selectedItem["Name"].ToString();  // Load dữ liệu dựa trên selectedValue  loadData(selectedValue);  }  else  {  // Xử lý trường hợp không thể chuyển đổi giá trị ID thành số nguyên  MessageBox.Show("Vui lòng chọn ID hợp lệ.");  }  }  else  {  // Xử lý khi không có item được chọn hoặc item không phải là DataRowView  MessageBox.Show("Vui lòng chọn ID hợp lệ.");  }  }  private void btnAdd\_Click(object sender, EventArgs e)  {  CompanyAddEditForm f = new CompanyAddEditForm();  f.ShowDialog();  loadData(selectedValue);  }  private void dtgv\_CellClick(object sender, DataGridViewCellEventArgs e)  {  if (e.RowIndex < 0) return; // Bỏ qua khi click vào tiêu đề  var dgv = sender as DataGridView;  var headerText = dgv.Columns[e.ColumnIndex].HeaderText;  if (headerText == "Xóa")  {  HandleDelete(dgv, e.RowIndex);  }  else if (headerText == "Sửa")  {  HandleEdit(dgv, e.RowIndex);  }  }  private void HandleDelete(DataGridView dgv, int rowIndex)  {  int Id = int.Parse(dgv.Rows[rowIndex].Cells[0].Value.ToString());  var Name = dgv.Rows[rowIndex].Cells[1].Value.ToString();  // Xác nhận trước khi xóa  var confirmation = MessageBox.Show(  $"Bạn có chắc chắn muốn xóa \"{Name}\"? Thao tác này sẽ xóa toàn bộ các Trung đội của của Đại đội này (nếu có).",  "Xác nhận",  MessageBoxButtons.YesNo,  MessageBoxIcon.Question  );  if (confirmation == DialogResult.Yes)  {  if (CompanyController.Instance().RemoveData(new Company(Id)))  {  loadData(selectedValue); // Tải lại dữ liệu  MessageBox.Show("Xóa Đại đội thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  else  {  MessageBox.Show("Xóa Đại đội thất bại.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Error);  }  }  }  private void HandleEdit(DataGridView dgv, int rowIndex)  {  var Id = int.Parse(dgv.Rows[rowIndex].Cells[0].Value.ToString());  var Name = dgv.Rows[rowIndex].Cells[1].Value.ToString();  var Description = dgv.Rows[rowIndex].Cells[2].Value.ToString();  var FK\_BattalionID = Convert.ToInt32(cb.SelectedValue);  //if (SubjectController.Instance().checkDuplicate(subjectSymbol, subjectId, subjectName) > 0)  //{  // MessageBox.Show("Ký hiệu hoặc tên môn học đã tồn tại!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Error);  // dtgvEduProgram[columnIndex, rowIndex].Value = previousValue; // Khôi phục giá trị trước đó  // return;  //}  CompanyAddEditForm f = new CompanyAddEditForm(new Company(Id, Name, Description, FK\_BattalionID));  f.ShowDialog();  loadData(selectedValue);  }  }  } |
| --- |

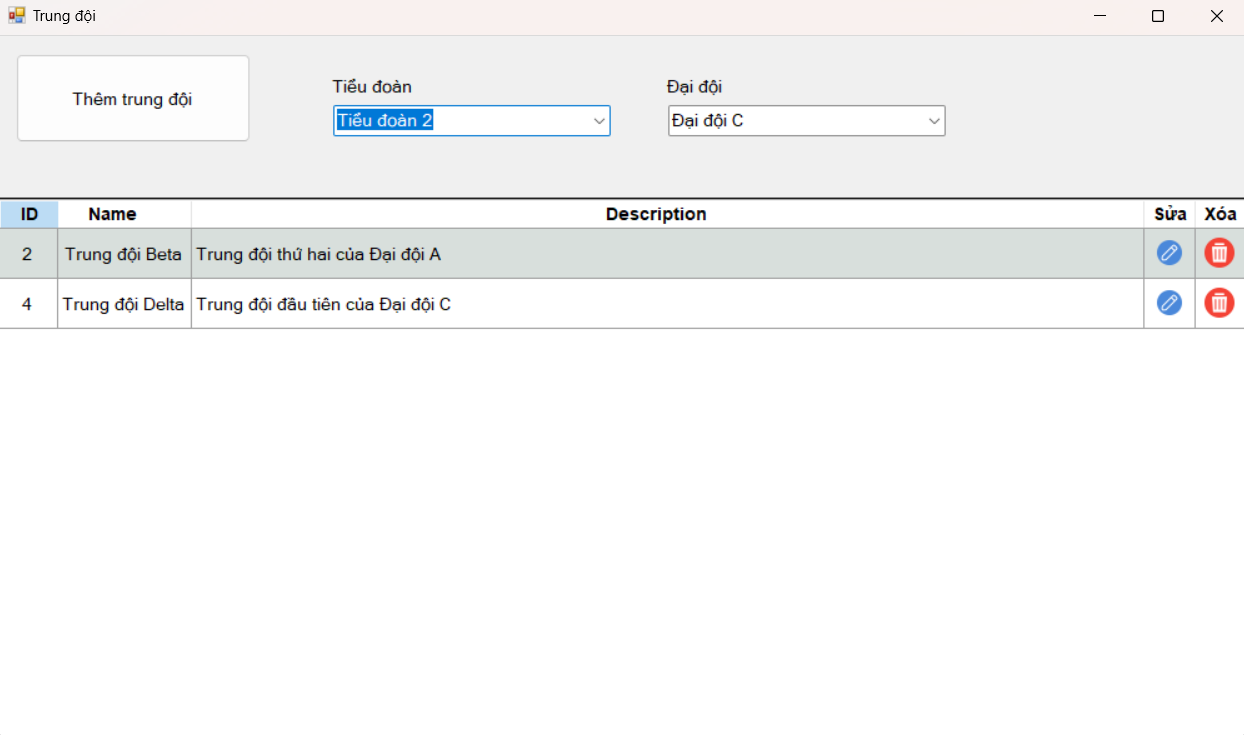
## 2.4. CompanyAddEditForm



| using MilitaryUnitManagement.Controllers;  using MilitaryUnitManagement.Models;  using System;  using System.Windows.Forms;  namespace MilitaryUnitManagement.Views  {  public partial class CompanyAddEditForm : Form  {  private bool isEdit = false;  private Company companyEdit = new Company();  public CompanyAddEditForm()  {  InitializeComponent();  CompanyController.Instance().LoadComboBoxData(cbBattalion);  }  public CompanyAddEditForm(Company company)  {  InitializeComponent();  CompanyController.Instance().LoadComboBoxData(cbBattalion);  isEdit = true;  companyEdit.ID = company.ID;  txbNameBattalion.Text = company.Name;  txbDescription.Text = company.Description;  cbBattalion.SelectedValue = company.FK\_BattalionID;  }  private void btnSave\_Click(object sender, EventArgs e)  {  if (!isEdit)  {  Company company = new Company();  company.Name = txbNameBattalion.Text;  company.Description = txbDescription.Text;  company.FK\_BattalionID = Convert.ToInt32(cbBattalion.SelectedValue);  if (CompanyController.Instance().AddData(company))  {  MessageBox.Show("Thêm tiểu đoàn thành công!");  this.Close();  }  else  {  MessageBox.Show("Thêm tiểu đoàn thất bại!");  }  }  else  {  companyEdit.Name = txbNameBattalion.Text;  companyEdit.Description = txbDescription.Text;  companyEdit.FK\_BattalionID = Convert.ToInt32(cbBattalion.SelectedValue);  if (CompanyController.Instance().EditData(companyEdit))  {  MessageBox.Show("Sửa tiểu đoàn thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);  this.Close();  }  else  {  MessageBox.Show("Sửa tiểu đoàn thất bại.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Error);  }  }  }  }  } |
| --- |

## 

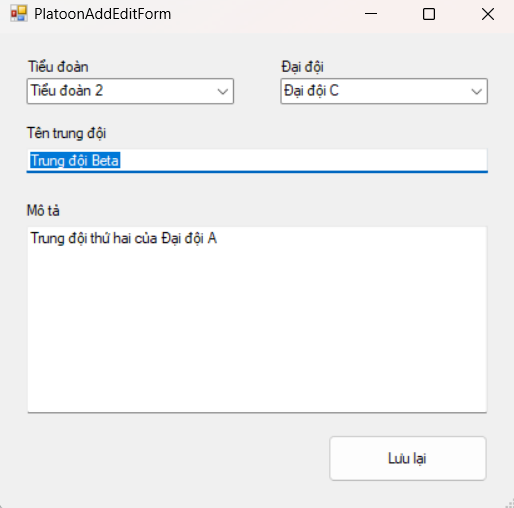
## 2.5. PlatoonForm



| using MilitaryUnitManagement.Controllers;  using MilitaryUnitManagement.Models;  using MilitaryUnitManagement.Services;  using System;  using System.Data;  using System.Windows.Forms;  namespace MilitaryUnitManagement.Views  {  public partial class PlatoonForm : Form  {  private ConfigView configView;  private int selectedBattalionID;  private int selectedCompanyID;  public PlatoonForm()  {  InitializeComponent();  configView = new ConfigView();  dtgv.RowTemplate.Height = 40;  }  private void PlatoonForm\_Load(object sender, EventArgs e)  {  PlatoonController.Instance().LoadComboBoxData\_Battalion(cbBattalion);  }  private void loadData(int FK\_BattalionID, int FK\_CompanyID)  {  PlatoonController.Instance().ShowData(dtgv, FK\_BattalionID, FK\_CompanyID);  configView.ConfigureDataGridViewReadOnly(dtgv, "ID");  configView.ConfigureColumnHeaders(dtgv);  configView.ConfigureColumnAlignment(dtgv, new string[] { "ID", "Name" });  configView.AddActionColumns(dtgv);  configView.ConfigureSelectionMode(dtgv);  }  private void cbBattalion\_SelectedIndexChanged(object sender, EventArgs e)  {  // Kiểm tra xem selectedItem có phải là DataRowView hay không  DataRowView selectedItem = cbBattalion.SelectedItem as DataRowView;  if (selectedItem != null)  {  if (int.TryParse(selectedItem["ID"].ToString(), out selectedBattalionID))  {  string selectedText = selectedItem["Name"].ToString();  PlatoonController.Instance().LoadComboBoxData\_Company(cbCompany, selectedBattalionID);  loadData(selectedBattalionID, selectedCompanyID);  }  }  }  private void cbCompany\_SelectedIndexChanged(object sender, EventArgs e)  {  // Kiểm tra xem selectedItem có phải là DataRowView hay không  DataRowView selectedItem = cbCompany.SelectedItem as DataRowView;  if (selectedItem != null)  {  if (int.TryParse(selectedItem["ID"].ToString(), out selectedCompanyID))  {  string selectedText = selectedItem["Name"].ToString();  // Tải dữ liệu với ID của cbCompany đang được chọn  loadData(selectedBattalionID, selectedCompanyID);  }  else  {  MessageBox.Show("Vui lòng chọn ID hợp lệ.");  }  }  else  {  MessageBox.Show("Vui lòng chọn ID hợp lệ.");  }  }  private void btnAdd\_Click(object sender, EventArgs e)  {  PlatoonAddEditForm f = new PlatoonAddEditForm();  f.ShowDialog();  loadData(selectedBattalionID, selectedCompanyID);  }  private void dtgv\_CellClick(object sender, DataGridViewCellEventArgs e)  {  if (e.RowIndex < 0) return; // Bỏ qua khi click vào tiêu đề  var dgv = sender as DataGridView;  var headerText = dgv.Columns[e.ColumnIndex].HeaderText;  if (headerText == "Xóa")  {  HandleDelete(dgv, e.RowIndex);  }  else if (headerText == "Sửa")  {  HandleEdit(dgv, e.RowIndex);  }  }  private void HandleDelete(DataGridView dgv, int rowIndex)  {  int Id = int.Parse(dgv.Rows[rowIndex].Cells[0].Value.ToString());  var Name = dgv.Rows[rowIndex].Cells[1].Value.ToString();  // Xác nhận trước khi xóa  var confirmation = MessageBox.Show(  $"Bạn có chắc chắn muốn xóa \"{Name}\"?",  "Xác nhận",  MessageBoxButtons.YesNo,  MessageBoxIcon.Question  );  if (confirmation == DialogResult.Yes)  {  if (PlatoonController.Instance().RemoveData(new Platoon(Id)))  {  loadData(selectedBattalionID, selectedCompanyID);  MessageBox.Show("Xóa Đại đội thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);  }  else  {  MessageBox.Show("Xóa Đại đội thất bại.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Error);  }  }  }  private void HandleEdit(DataGridView dgv, int rowIndex)  {  var Id = int.Parse(dgv.Rows[rowIndex].Cells[0].Value.ToString());  var Name = dgv.Rows[rowIndex].Cells[1].Value.ToString();  var Description = dgv.Rows[rowIndex].Cells[2].Value.ToString();  var FK\_CompanyID = Convert.ToInt32(cbCompany.SelectedValue);  //if (SubjectController.Instance().checkDuplicate(subjectSymbol, subjectId, subjectName) > 0)  //{  // MessageBox.Show("Ký hiệu hoặc tên môn học đã tồn tại!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Error);  // dtgvEduProgram[columnIndex, rowIndex].Value = previousValue; // Khôi phục giá trị trước đó  // return;  //}  PlatoonAddEditForm f = new PlatoonAddEditForm(new Platoon(Id, Name, Description, FK\_CompanyID), selectedBattalionID);  f.ShowDialog();  loadData(selectedBattalionID, selectedCompanyID);  }  }  } |
| --- |

## 

## 2.6. PlatoonAddEditForm



| using MilitaryUnitManagement.Controllers;  using MilitaryUnitManagement.Models;  using System;  using System.Data;  using System.Windows.Forms;  namespace MilitaryUnitManagement.Views  {  public partial class PlatoonAddEditForm : Form  {  private bool isEdit = false;  private Platoon platoonEdit = new Platoon();  private int selectedBattalionID;  private int selectedCompanyID;  public PlatoonAddEditForm()  {  InitializeComponent();  CompanyController.Instance().LoadComboBoxData(cbBattalion);  PlatoonController.Instance().LoadComboBoxData\_Battalion(cbBattalion);  }  public PlatoonAddEditForm(Platoon platoon, int selectedBattalionID)  {  InitializeComponent();  CompanyController.Instance().LoadComboBoxData(cbBattalion);  PlatoonController.Instance().LoadComboBoxData\_Battalion(cbBattalion);  isEdit = true;  platoonEdit.ID = platoon.ID;  txbNamePlatoon.Text = platoon.Name;  txbDescription.Text = platoon.Description;  cbCompany.SelectedValue = platoon.FK\_CompanyID;  cbBattalion.SelectedValue = selectedBattalionID;  }  private void cbBattalion\_SelectedIndexChanged(object sender, EventArgs e)  {  // Kiểm tra xem selectedItem có phải là DataRowView hay không  DataRowView selectedItem = cbBattalion.SelectedItem as DataRowView;  if (selectedItem != null)  {  if (int.TryParse(selectedItem["ID"].ToString(), out selectedBattalionID))  {  string selectedText = selectedItem["Name"].ToString();  PlatoonController.Instance().LoadComboBoxData\_Company(cbCompany, selectedBattalionID);  selectedCompanyID = Convert.ToInt32(cbCompany.SelectedValue);  }  }  }  private void cbCompany\_SelectedIndexChanged(object sender, EventArgs e)  {  DataRowView selectedItem = cbCompany.SelectedItem as DataRowView;  if (selectedItem != null)  {  selectedCompanyID = Convert.ToInt32(selectedItem["ID"]);  }  }  private void btnSave\_Click(object sender, EventArgs e)  {  if (!isEdit)  {  Platoon platoon = new Platoon();  platoon.Name = txbNamePlatoon.Text;  platoon.Description = txbDescription.Text;  platoon.FK\_CompanyID = selectedCompanyID;  if (cbCompany.SelectedItem == null)  {  MessageBox.Show($"{cbBattalion.Text} chưa có Đại đội nào. Hãy thêm Đại đội rồi quay lại nhé!");  return;  }  if (PlatoonController.Instance().AddData(platoon))  {  MessageBox.Show("Thêm Trung đội thành công!");  this.Close();  }  else  {  MessageBox.Show("Thêm Trung đội thất bại!");  }  }  else  {  platoonEdit.Name = txbNamePlatoon.Text;  platoonEdit.Description = txbDescription.Text;  platoonEdit.FK\_CompanyID = selectedCompanyID;  MessageBox.Show($"Mã Trung đội: {platoonEdit.ID}\nMã đại đội: {selectedCompanyID}");  if (PlatoonController.Instance().EditData(platoonEdit))  {  MessageBox.Show("Sửa Trung đội thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);  this.Close();  }  else  {  MessageBox.Show("Sửa Trung đội thất bại.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Error);  }  }  }  }  } |
| --- |

## 

# 3. Controllers

## 3.1. BattalionController

| using MilitaryUnitManagement.Models;  using MilitaryUnitManagement.Services;  using System.Data.SqlClient;  using System.Windows.Forms;  namespace MilitaryUnitManagement.Controllers  {  internal class BattalionController  {  private static BattalionController instance;  private ConnectDatabase connectDatabase = ConnectDatabase.getInstance();  private BattalionController() { }  // Khởi tạo một instance mới nếu chưa tồn tại, ngược lại trả về instance hiện tại  public static BattalionController Instance()  {  if (instance == null)  {  instance = new BattalionController();  }  return instance;  }  // Hiển thị dữ liệu  public void ShowData(DataGridView dgv)  {  string query = "SELECT \* FROM Battalion";  dgv.Columns.Clear();  dgv.DataSource = ConnectDatabase.getInstance().ExecuteQuery(query);  }  // Thêm dữ liệu  public bool AddData(Battalion b)  {  string query = "INSERT INTO Battalion VALUES (@Name, @Description)";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("Name", b.Name),  new SqlParameter("Description", b.Description)  };  return connectDatabase.ExecuteNonQuery(query, para) > 0;  }  // Sửa dữ liệu  public bool EditData(Battalion b)  {  string query = "UPDATE Battalion " +  "SET Name = @Name, " +  "Description = @Description " +  "WHERE ID = @ID";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("ID", b.ID),  new SqlParameter("Name", b.Name),  new SqlParameter("Description", b.Description)  };  int rowAffect = connectDatabase.ExecuteNonQuery(query, para);  return rowAffect > 0 ? true : false;  }  // Xóa dữ liệu  public bool RemoveData(Battalion b)  {  string query = "DELETE FROM Battalion WHERE ID = @ID";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("ID", b.ID)  };  int rowAffect = connectDatabase.ExecuteNonQuery(query, para);  return rowAffect > 0 ? true : false;  }  }  } |
| --- |

## 3.2. CompanyController

| using System.Data;  using System.Data.SqlClient;  using System.Windows.Forms;  using MilitaryUnitManagement.Models;  using MilitaryUnitManagement.Services;  namespace MilitaryUnitManagement.Controllers  {  internal class CompanyController  {  private static CompanyController instance;  private ConnectDatabase connectDatabase = ConnectDatabase.getInstance();  private CompanyController() { }  public static CompanyController Instance()  {  if (instance == null)  {  instance = new CompanyController();  }  return instance;  }  public void ShowData(DataGridView dgv, int ID)  {  string query = $"SELECT ID, Name, Description FROM Company WHERE FK\_BattalionID = @FK\_BattalionID";  SqlParameter parameter = new SqlParameter("@FK\_BattalionID", ID);  dgv.Columns.Clear();  dgv.DataSource = connectDatabase.ExecuteQuery(query, new SqlParameter[] { parameter });  }  public void LoadComboBoxData(ComboBox comboBox)  {  string query = "SELECT ID, Name FROM Battalion";  DataTable dataTable = connectDatabase.ExecuteQuery(query);  comboBox.DataSource = dataTable;  comboBox.DisplayMember = "Name"; // Column to display  comboBox.ValueMember = "Id"; // Column for value  }  // Thêm dữ liệu  public bool AddData(Company c)  {  string query = "INSERT INTO Company VALUES (@Name, @Description, @FK\_BattalionID)";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("Name", c.Name),  new SqlParameter("FK\_BattalionID", c.Description),  new SqlParameter("Description", c.FK\_BattalionID)  };  return connectDatabase.ExecuteNonQuery(query, para) > 0;  }  // Sửa dữ liệu  public bool EditData(Company c)  {  string query = "UPDATE Company " +  "SET Name = @Name, " +  "Description = @Description, " +  "FK\_BattalionID = @FK\_BattalionID " +  "WHERE ID = @ID";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("ID", c.ID),  new SqlParameter("Name", c.Name),  new SqlParameter("Description", c.Description),  new SqlParameter("FK\_BattalionID", c.FK\_BattalionID)  };  int rowAffect = connectDatabase.ExecuteNonQuery(query, para);  return rowAffect > 0 ? true : false;  }  // Xóa dữ liệu  public bool RemoveData(Company c)  {  string query = "DELETE FROM Company WHERE ID = @ID";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("ID", c.ID)  };  int rowAffect = connectDatabase.ExecuteNonQuery(query, para);  return rowAffect > 0 ? true : false;  }  }  } |
| --- |

## 3.3. PlatoonController

| using MilitaryUnitManagement.Services;  using System.Data.SqlClient;  using System.Data;  using System.Windows.Forms;  using MilitaryUnitManagement.Models;  namespace MilitaryUnitManagement.Controllers  {  internal class PlatoonController  {  private static PlatoonController instance;  private ConnectDatabase connectDatabase = ConnectDatabase.getInstance();  private PlatoonController() { }  public static PlatoonController Instance()  {  if (instance == null)  {  instance = new PlatoonController();  }  return instance;  }  // Hiển thị dữ liệu  public void ShowData(DataGridView dgv, int FK\_BattalionID, int FK\_CompanyID)  {  string query = "SELECT Platoon.ID, Platoon.Name, Platoon.Description FROM Platoon INNER JOIN Company ON Platoon.FK\_CompanyID = Company.ID " +  "WHERE Company.FK\_BattalionID = @FK\_BattalionID AND Platoon.FK\_CompanyID = @FK\_CompanyID";  // Tạo các tham số SqlParameter  SqlParameter parameter1 = new SqlParameter("@FK\_BattalionID", FK\_BattalionID);  SqlParameter parameter2 = new SqlParameter("@FK\_CompanyID", FK\_CompanyID);  // Gửi tham số vào hàm ExecuteQuery và nhận lại DataTable  dgv.Columns.Clear();  dgv.DataSource = connectDatabase.ExecuteQuery(query, new SqlParameter[] { parameter1, parameter2 });  }  // Load data into ComboBox  public void LoadComboBoxData\_Battalion(ComboBox comboBox)  {  string query = "SELECT ID, Name FROM Battalion";  DataTable dataTable = connectDatabase.ExecuteQuery(query);  comboBox.DataSource = dataTable;  comboBox.DisplayMember = "Name"; // Column to display  comboBox.ValueMember = "Id"; // Column for value  }  public void LoadComboBoxData\_Company(ComboBox comboBox, int selectedBattalion)  {  string query = "SELECT ID, Name FROM Company WHERE FK\_BattalionID = @FK\_BattalionID";  SqlParameter parameter = new SqlParameter("@FK\_BattalionID", selectedBattalion);  DataTable dataTable = connectDatabase.ExecuteQuery(query, new SqlParameter[] { parameter });  if (dataTable.Rows.Count != 0)  {  comboBox.DataSource = dataTable;  comboBox.DisplayMember = "Name"; // Column to display  comboBox.ValueMember = "Id"; // Column for value  }  }    // Thêm dữ liệu  public bool AddData(Platoon p)  {  string query = "INSERT INTO Platoon(Name, Description, FK\_CompanyID) VALUES (@Name, @Description, @FK\_CompanyID)";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("Name", p.Name),  new SqlParameter("Description", p.Description),  new SqlParameter("FK\_CompanyID", p.FK\_CompanyID)  };  return connectDatabase.ExecuteNonQuery(query, para) > 0;  }  // Sửa dữ liệu  public bool EditData(Platoon p)  {  string query = "UPDATE Platoon " +  "SET Name = @Name, " +  "Description = @Description, " +  "FK\_CompanyID = @FK\_CompanyID " +  "WHERE ID = @ID";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("ID", p.ID),  new SqlParameter("Name", p.Name),  new SqlParameter("Description", p.Description),  new SqlParameter("FK\_CompanyID", p.FK\_CompanyID)  };  int rowAffect = connectDatabase.ExecuteNonQuery(query, para);  return rowAffect > 0 ? true : false;  }  // Xóa dữ liệu  public bool RemoveData(Platoon p)  {  string query = "DELETE FROM Platoon WHERE ID = @ID";  SqlParameter[] para = new SqlParameter[]  {  new SqlParameter("ID", p.ID)  };  int rowAffect = connectDatabase.ExecuteNonQuery(query, para);  return rowAffect > 0 ? true : false;  }  }  } |
| --- |

# 4. Services

## 4.1. ConnectDatabase

| using System;  using System.Collections.Generic;  using System.Data;  using System.Data.SqlClient;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace MilitaryUnitManagement.Services  {  internal class ConnectDatabase  {  // laptop = "DUNGHAQUANG";  // desktop = "DESKTOP-001";  private static ConnectDatabase instance;  private string \_connectionString = $"Data Source=DUNGHAQUANG;Initial Catalog=MilitaryUnitManagement;Integrated Security=True";  private ConnectDatabase() { }  public static ConnectDatabase getInstance()  {  if (instance == null)  {  instance = new ConnectDatabase();  }  return instance;  }  public DataTable ExecuteQuery(string query, SqlParameter[] parameter = null)  {  DataTable data = new DataTable();  using (SqlConnection sqlConnection = new SqlConnection(\_connectionString))  {  sqlConnection.Open();  SqlCommand cmd = new SqlCommand(query, sqlConnection);  if (parameter != null)  cmd.Parameters.AddRange(parameter);  SqlDataAdapter adapter = new SqlDataAdapter(cmd);  data.Clear();  adapter.Fill(data);  sqlConnection.Close();  }  return data;  }  // Sử dụng để thêm sửa xóa  public int ExecuteNonQuery(string query, SqlParameter[] parameter = null)  {  int rowsAffected = 0;  using (SqlConnection sqlConnection = new SqlConnection(\_connectionString))  {  sqlConnection.Open();  SqlCommand cmd = new SqlCommand(query, sqlConnection);  if (parameter != null)  cmd.Parameters.AddRange(parameter);  rowsAffected = cmd.ExecuteNonQuery();  sqlConnection.Close();  }  return rowsAffected;  }  public string GetValue(string query)  {  string value = null;  using (SqlConnection sqlConnection = new SqlConnection(\_connectionString))  {  sqlConnection.Open();  SqlCommand cmd = new SqlCommand(query, sqlConnection);  SqlDataReader dataReader = cmd.ExecuteReader();  if (dataReader.Read())  {  value = dataReader.GetValue(0).ToString();  }  sqlConnection.Close();  }  return value;  }  public SqlDataReader ExecuteReader(string query)  {  SqlConnection sqlConnection = new SqlConnection(\_connectionString);  sqlConnection.Open();  SqlCommand cmd = new SqlCommand(query, sqlConnection);  SqlDataReader reader = cmd.ExecuteReader();  return reader;  }  }  } |
| --- |

## 4.2. ConfigView

| using System;  using System.Drawing;  using System.IO;  using System.Windows.Forms;  namespace MilitaryUnitManagement.Services  {  internal class ConfigView  {  public void AddActionColumns(DataGridView dgv)  {  if (dgv.Columns["Edit"] == null) // Add if not exists  {  string editImagePath = Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "..", "..", "Resources", "ICON\_EDIT-24x24.png");  var editColumn = new DataGridViewImageColumn  {  Name = "Edit",  HeaderText = "Sửa",  Image = Image.FromFile(editImagePath),  AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCells  };  dgv.Columns.Add(editColumn);  }  if (dgv.Columns["Delete"] == null) // Add if not exists  {  string deleteImagePath = Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "..", "..", "Resources", "ICON\_TRASH-24x24.png");  var deleteColumn = new DataGridViewImageColumn  {  Name = "Delete",  HeaderText = "Xóa",  Image = Image.FromFile(deleteImagePath),  AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCells  };  dgv.Columns.Add(deleteColumn);  //deleteColumn.DefaultCellStyle.BackColor = ColorTranslator.FromHtml("#f44336");  }  }  public void ConfigureDataGridViewReadOnly(DataGridView dgv, string columnName)  {  var column = dgv.Columns[columnName];  if (column != null)  {  column.ReadOnly = true; // Set column to read-only  }  }  public void ConfigureColumnHeaders(DataGridView dgv)  {  var headerStyle = new DataGridViewCellStyle  {  Alignment = DataGridViewContentAlignment.MiddleCenter,  BackColor = SystemColors.GrayText,  ForeColor = SystemColors.WindowText,  Font = new Font("Arial", 10, FontStyle.Bold)  };  dgv.ColumnHeadersDefaultCellStyle = headerStyle;  }  public void ConfigureColumnAlignment(DataGridView dgv, string[] columnNames)  {  foreach (var columnName in columnNames)  {  var column = dgv.Columns[columnName];  if (column != null)  {  column.DefaultCellStyle.Alignment = DataGridViewContentAlignment.MiddleCenter;  column.HeaderCell.Style.Alignment = DataGridViewContentAlignment.MiddleCenter;  column.AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCells;  }  }  }  public void ConfigureSelectionMode(DataGridView dgv)  {  dgv.SelectionMode = DataGridViewSelectionMode.FullRowSelect;  dgv.DefaultCellStyle.SelectionBackColor = ColorTranslator.FromHtml("#d8dfdc");  dgv.DefaultCellStyle.SelectionForeColor = ColorTranslator.FromHtml("#000");  }  }  } |
| --- |