BÀI THỰC HÀNH SỐ 9 (8 tiết)

ENTITY FRAMEWORK

I. Mục tiêu:

Bài thực hành này giúp sinh viên tìm hiểu cách sử dụng Entity Framework để kết nối tới cơ sở dữ liệu và thực hiện các truy vấn đơn giản:

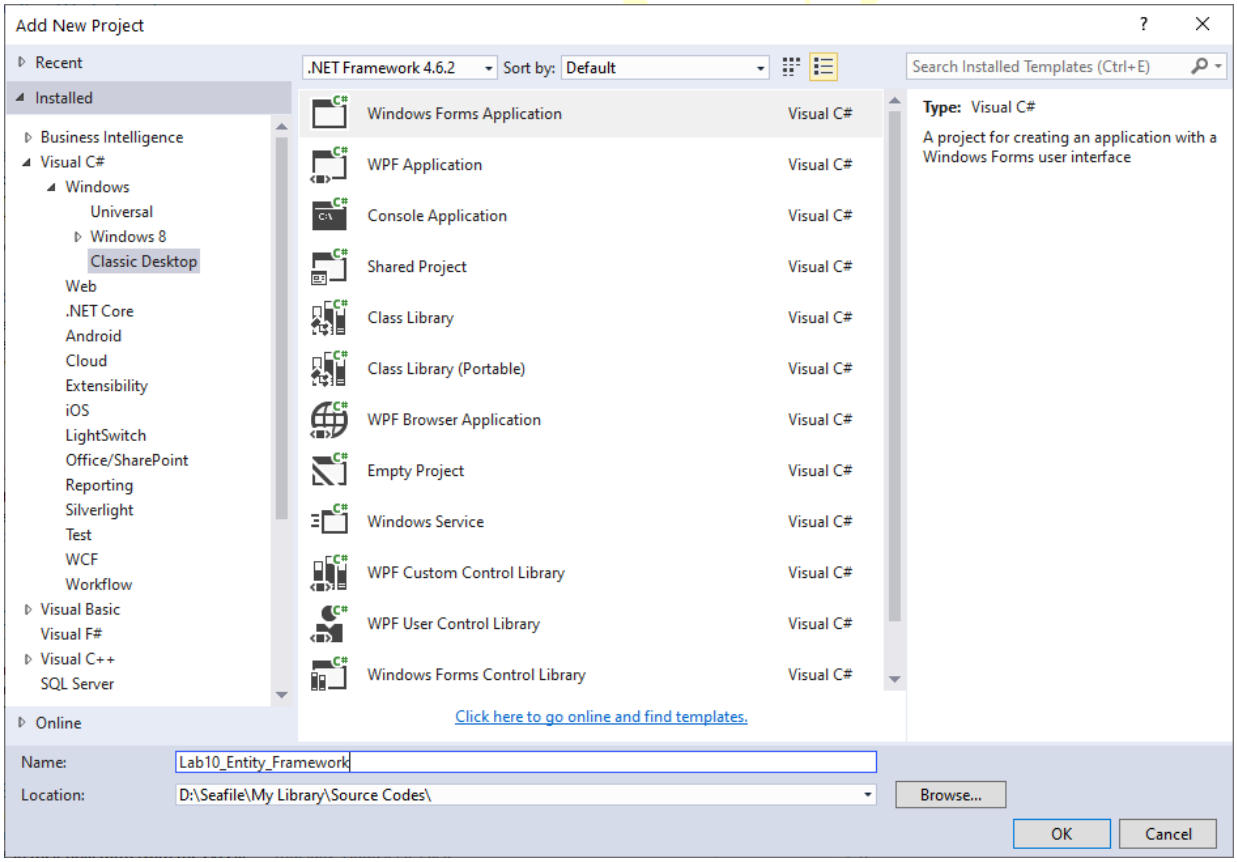
- SELECT: Lấy các mẫu tin từ một bảng.
- INSERT: Thêm một mẫu tin mới vào một bảng.
- UPDATE: Cập nhật một mẫu tin có sẵn trong bảng.
- DELETE: Xóa một mẫu tin khỏi bảng.

Sau bài thực hành này, sinh viên cần nắm rõ những vấn đề sau:

- Các thành phần của chuỗi kết nối, ý nghĩa của chúng và cách tạo chuỗi kết nối.
- Cách định nghĩa các lớp thực thể và lớp ngữ cảnh.
- Cách sử dụng đối tượng Context để thực thi truy vấn, thêm, cập nhật dữ liệu vào cơ sở dữ liệu.
- Cách sử dụng lớp DTO để lấy thông tin từ nhiều bảng.
- Cách xây dựng ứng dụng trên nền Windows Form với Entity Framework.

II. Thực hành:

Tạo một dự án Windows Application mới, đặt tên là Lab09_Entity_Framework



Đặt lại tên Form1 thành MainForm. Kéo một điều khiển ToolTip vào MainForm. Thiết kế MainForm như sau:

MainForm:
- Text: Quản lý nhà hàng
- StartPosition: CenterScreen

Button R:
- Name: btnReloadCategory
- ToolTip on toolTip1: Tải lại danh mục
- Text: R

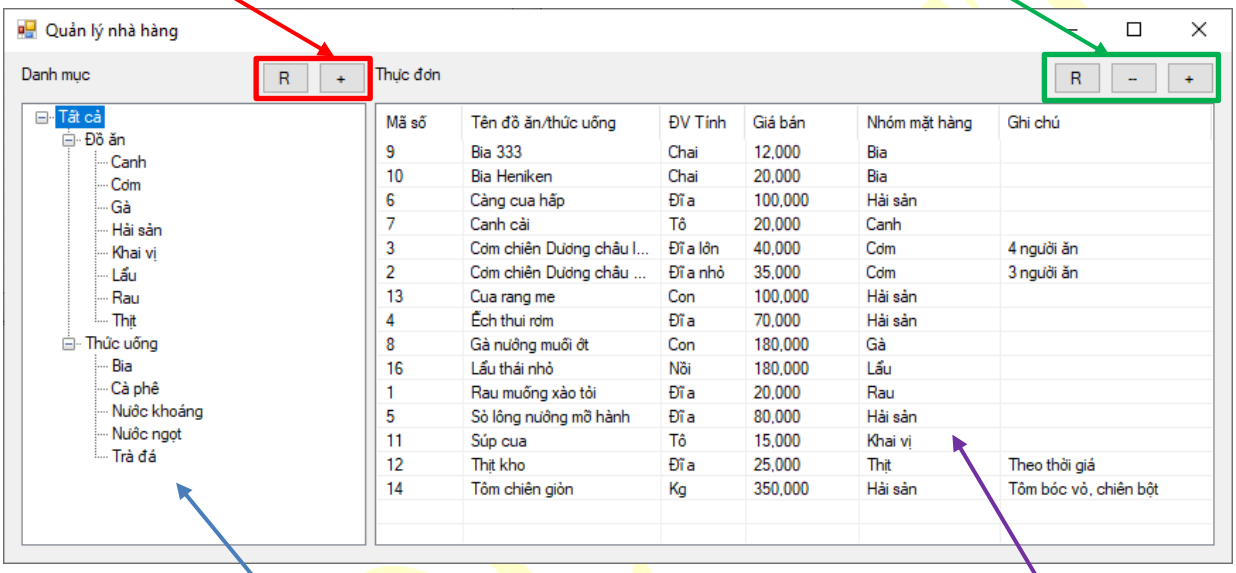
Button +:
- Name: btnAddCategory
- ToolTip on toolTip1: Thêm danh mục mới
- Text: +

Button R:
- Name: btnReloadFood
- ToolTip: Tải lại danh sách món ăn

Button --:
- Name: btnDelete
- ToolTip: Xóa món ăn được chọn

Button +:
- Name: btnAddFood
- ToolTip: Thêm món ăn mới

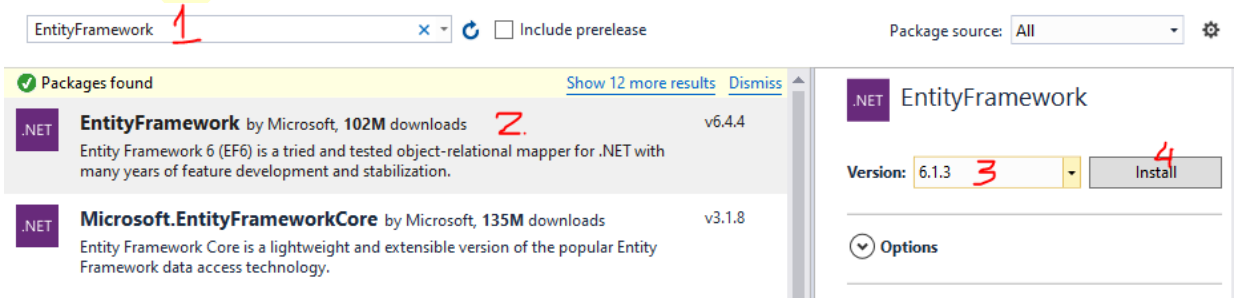
Cả 3 buttons: Anchor: Top, Right



TreeView:
- Name: tvwCategory
- Anchor: Top, Bottom, Left

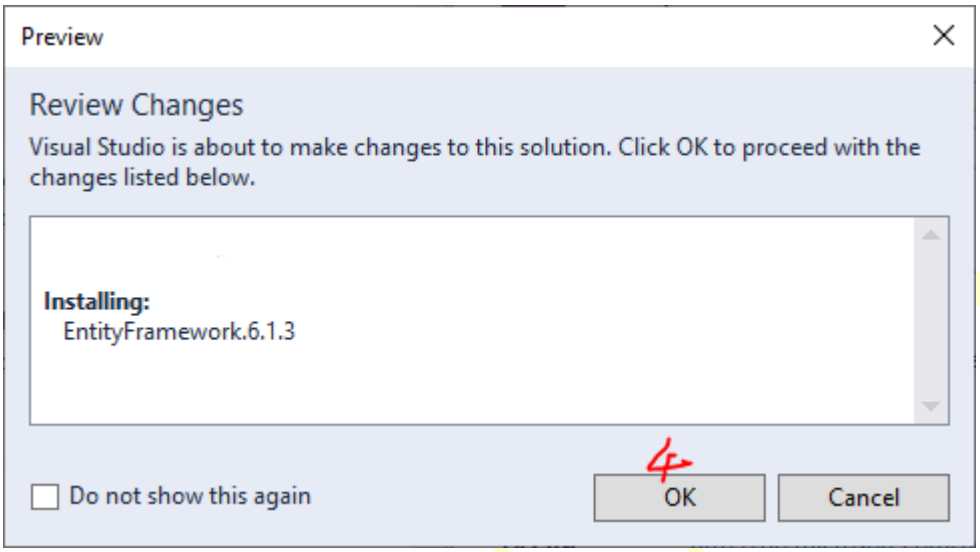
ListView:
- Name: lvwFood
- Anchor: Top, Bottom, Left, Right
- FullRowSelect: True
- GridLines: True
- MultiSelect: False
- View: Details

1. *Cài đặt gói thư viện EntityFramework*



Nhấp phải chuột vào dự án, chọn Manage Nuget Packages …

- Trong cửa sổ mới "NuGet: Lab09_Entity_Framework", chọn tab Browse. Trong mục Package source, chọn All. Nhập từ khóa EntityFramework vào ô tìm kiếm.
- Trong cửa sổ hiển thị kết quả tìm kiếm, chọn EntityFramework.
- Ở khung kế bên phải, chọn phiên bản muốn cài đặt là 6.1.3. Sau đó nhấn nút Install.
- Trong cửa sổ popup Preview, nhấn nút OK để xác nhận và bắt đầu cài đặt thư viện



2. *Định nghĩa các lớp thực thể và lớp ngữ cảnh (DbContext)*

Nhấp phải chuột vào tên dự án, chọn Add > New Folder. Đặt tên thư mục là Models. Trong thư mục Models, tạo ra các lớp thực thể sau đây:

- Lớp Category.cs: Biểu diễn thông tin nhóm đồ ăn, thức uống.

```
8 references | 0 changes | 0 authors, 0 changes
public class Category
{
    5 references | 0 changes | 0 authors, 0 changes
    public int Id { get; set; }

    9 references | 0 changes | 0 authors, 0 changes
    public string Name { get; set; }

    6 references | 0 changes | 0 authors, 0 changes
    public CategoryType Type { get; set; }
}

7 references | 0 changes | 0 authors, 0 changes
public enum CategoryType
{
    Drink,
    Food
}
```

- Lớp Food.cs: Biểu diễn thông tin về một món ăn, đồ uống.

```
5 references | 0 changes | 0 authors, 0 changes
public class Food
{
    4 references | 0 changes | 0 authors, 0 changes
    public int Id { get; set; }

    8 references | 0 changes | 0 authors, 0 changes
    public string Name { get; set; }

    6 references | 0 changes | 0 authors, 0 changes
    public string Unit { get; set; }

    6 references | 0 changes | 0 authors, 0 changes
    public int FoodCategoryId { get; set; }

    6 references | 0 changes | 0 authors, 0 changes
    public int Price { get; set; }

    6 references | 0 changes | 0 authors, 0 changes
    public string Notes { get; set; }


    4 references | 0 changes | 0 authors, 0 changes
    public Category Category { get; set; }
}
```

Tiếp theo, để có thể truy xuất tới cơ sở dữ liệu, ta cần định nghĩa lớp ngữ cảnh RestaurantContext.cs trong thư mục Models như sau:

```
8 references | 0 changes | 0 authors, 0 changes
public class RestaurantContext : DbContext
{
    // Tham chiếu tới các nhóm món ăn trong bảng Category
    4 references | 0 changes | 0 authors, 0 changes
    public DbSet<Category> Categories { get; set; }

    // Tham chiếu tới các món ăn, đồ uống trong bảng Food
    6 references | 0 changes | 0 authors, 0 changes
    public DbSet<Food> Foods { get; set; }

    0 references | 0 changes | 0 authors, 0 changes
    protected override void OnModelCreating(DbModelBuilder modelBuilder)
    {
        // Xóa bỏ quy tắc sử dụng danh từ số nhiều cho tên bảng
        // Lúc này, thuộc tính Categories ánh xạ tới bảng Category trong db
        // Và thuộc tính Foods tương ứng với bảng Food trong cơ sở dữ liệu
        modelBuilder.Conventions.Remove<PluralizingTableNameConvention>();

        // Định nghĩa mối quan hệ một nhiều giữa hai bảng Category và Food
        modelBuilder.Entity<Food>()
            .HasRequired(x => x.Category)
            .WithMany()
            .HasForeignKey(x => x.FoodCategoryId)
            .WillCascadeOnDelete(true);
    }
}
```

Và cuối cùng, để lấy thông tin chi tiết món ăn từ cả hai bảng Food và Category, ta tạo một lớp DTO (Data Transfer Object) làm trung gian và chứa những thuộc tính cần thiết.

Nhấp phải chuột vào thư mục Models, chọn Add > Class … Đặt tên là FoodModel. Định nghĩa các thuộc tính cho lớp FoodModel như sau:

```
6 references | 0 changes | 0 authors, 0 changes
public class FoodModel
{
    3 references | 0 changes | 0 authors, 0 changes
    public int Id { get; set; }

    3 references | 0 changes | 0 authors, 0 changes
    public string Name { get; set; }

    3 references | 0 changes | 0 authors, 0 changes
    public string Unit { get; set; }

    3 references | 0 changes | 0 authors, 0 changes
    public string CategoryName { get; set; }

    3 references | 0 changes | 0 authors, 0 changes
    public int Price { get; set; }

    3 references | 0 changes | 0 authors, 0 changes
    public string Notes { get; set; }
}
```

3. *Định nghĩa chuỗi thông tin kết nối tới cơ sở dữ liệu*

Nhấp đôi chuột vào tập tin App.config để mở nó trong cửa sổ soạn thảo. Nếu chưa có, nhấp phải chuột vào tên dự án, chọn Add > New Item. Trong cửa sổ Add New Item, chọn Visual C# Items > General > Application Configuration File rồi nhấn nút Add.
Trong tập tin App.config, bổ sung thêm thẻ <connectionStrings> để thêm chuỗi kết nối tới cơ sở dữ liệu như trong hình sau:

```
<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>
    <connectionStrings>
        <add name="RestaurantContext"
             connectionString="Data Source=(local)\SQL16E;Initial Catalog=RestaurantManagement;User ID=sa;Password=123456;"
             providerName="System.Data.SqlClient"/>
    </connectionStrings>
    <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.6.2" />
    </startup>
```

Có thể, bạn cần phải thay đổi thông tin trong chuỗi kết nối cho phù hợp với thông tin mà bạn đã cài đặt MS SQL Server và tạo cơ sở dữ liệu.

4. *Nạp danh mục các nhóm món ăn lên TreeView*

Mở MainForm, nhấp đôi chuột vào form để tạo hàm xử lý sự kiện Load trên MainForm. Tiếp đến, nhấp đôi chuột vào nút R bên trái (btnReloadCategory) để tạo hàm xử lý sự kiện Click trên nút btnReloadCategory.

```
1 reference | 0 changes | 0 authors, 0 changes
private void MainForm_Load(object sender, EventArgs e)
{

}

1 reference | 0 changes | 0 authors, 0 changes
private void btnReloadCategory_Click(object sender, EventArgs e)
{

}
```

Cũng trong lớp MainForm.cs, định nghĩa các phương thức sau:

```
1 reference | 0 changes | 0 authors, 0 changes
private List<Category> GetCategories()
{
    // Khởi tạo đối tượng context
    var dbContext = new RestaurantContext();

    // Lấy danh sách tất cả nhóm thức ăn, sắp xếp theo tên
    return dbContext.Categories.OrderBy(x => x.Name).ToList();
}
```

```
4 references | 0 changes | 0 authors, 0 changes
private void ShowCategories()
{
    // Xóa tất cả các nút hiện có trên cây
    tvwCategory.Nodes.Clear();

    // Tạo danh sách loại nhóm thức ăn, đồ uống
    // Tên của các loại này được hiển thị trên các nút mức 2
    var cateMap = new Dictionary<CategoryType, string>()
    {
        [CategoryType.Food] = "Đồ ăn",
        [CategoryType.Drink] = "Thức uống"
    };

    // Tạo nút gốc của cây
    var rootNode = tvwCategory.Nodes.Add("Tất cả");

    // Lấy danh sách nhóm đồ ăn, thức uống
    var categories = GetCategories();

    // Duyệt qua các loại nhóm thức ăn
    foreach (var cateType in cateMap)
    {
        // Tạo các nút tương ứng với loại nhóm thức ăn
        var childNode = rootNode.Nodes.Add(cateType.Key.ToString(), cateType.Value);
        childNode.Tag = cateType.Key;

        // Duyệt qua các nhóm thức ăn
        foreach (var category in categories)
        {
            // Nếu nhóm đang xét không cùng loại thì bỏ qua
            if (category.Type != cateType.Key) continue;

            // Ngược lại, tạo các nút tương ứng trên cây
            var grantChildNode = childNode.Nodes.Add(category.Id.ToString(), category.Name);
            grantChildNode.Tag = category;
```

```
        }
    }

    // Mở rộng các nhánh của cây để thấy hết tất cả các nhóm thức ăn
    tvwCategory.ExpandAll();

    // Đánh dấu nút gốc đang được chọn
    tvwCategory.SelectedNode = rootNode;
}
```

Sau đó, gọi hàm ShowCategories() trong trong hai hàm xử lý sự kiện đã tạo lúc nãy:

```
1 reference | 0 changes | 0 authors, 0 changes
private void MainForm_Load(object sender, EventArgs e)
{
    ShowCategories();
}

1 reference | 0 changes | 0 authors, 0 changes
private void btnReloadCategory_Click(object sender, EventArgs e)
{
    ShowCategories();
}
```

Nhấn F5 để chạy chương trình và xem kết quả.
Nhấn nút "R" ở phía bên trái để tải lại danh mục nhóm món ăn.

5. Hiển thị danh sách món ăn, đồ uống khi chọn một danh mục

Trong phần này, ta sẽ định nghĩa các phương thức và xử lý sự kiện để hiển thị danh sách các món ăn, đồ uống vào ListView phía bên phải khi người dùng chọn một nút trong TreeView bên trái.

- Nếu người dùng chọn nút Tất cả, ListView bên phải sẽ hiển thị tất cả các món ăn, thức uống.
- Nếu người dùng chọn nút Đồ ăn, tất cả các món ăn trong các danh mục đồ ăn sẽ được hiển thị trong ListView.
- Nếu người dùng chọn nút Thức uống, ListView sẽ cho thấy tất cả các đồ uống trong thực đơn.
- Nếu người dùng chọn một danh mục, các món ăn hoặc đồ uống trong danh mục (nhóm) đó sẽ được liệt kê trong ListView.

Như vậy, ta cần phải phân biệt các nút được chọn để có thể truy vấn danh sách các món ăn, đồ uống thích hợp. Ở đây, ta dùng mức (level) của các nút trên cây để phân biệt.
Trước tiên, ta định nghĩa các phương thức sau:

- GetFoodByCategory(int? categoryId): Lấy danh sách món ăn theo mã danh mục. Nếu mã này bằng null thì lấy tất cả các món ăn. Sắp xếp món ăn tăng dần theo tên.
- GetFoodByCategoryType(CategoryType cateType): Lấy danh sách món ăn hoặc đồ uống, tùy theo loại danh mục.

```
1 reference | 0 changes | 0 authors, 0 changes
private List<FoodModel> GetFoodByCategory(int? categoryId)
{
    // Khởi tạo đối tượng context
    var dbContext = new RestaurantContext();

    // Tạo truy vấn lấy danh sách món ăn
    var foods = dbContext.Foods.AsQueryable();

    // Nếu mã nhóm món ăn khác null và hợp lệ
    if (categoryId != null && categoryId > 0)
    {
        // Thì tìm theo mã số nhóm thức ăn
        foods = foods.Where(x => x.FoodCategoryId == categoryId);
    }

    // Sắp xếp đồ ăn, thức uống theo tên và trả về
    // danh sách chứa đầy đủ thông tin về món ăn.
    return foods
        .OrderBy(x => x.Name)
        .Select(x => new FoodModel()
        {
            Id = x.Id,
            Name = x.Name,
            Unit = x.Unit,
            Price = x.Price,
            Notes = x.Notes,
            CategoryName = x.Category.Name
        })
        .ToList();
}
1 reference | 0 changes | 0 authors, 0 changes
private List<FoodModel> GetFoodByCategoryType(CategoryType cateType)
{
    var dbContext = new RestaurantContext();

    // Tìm các món ăn theo loại nhóm thức ăn (Category Type).
    // Sắp xếp đồ ăn, thức uống theo tên và trả về
    // danh sách chứa đầy đủ thông tin về món ăn.
    return dbContext.Foods
        .Where(x => x.Category.Type == cateType)
        .OrderBy(x => x.Name)
        .Select(x => new FoodModel()
        {
            Id = x.Id,
            Name = x.Name,
            Unit = x.Unit,
            Price = x.Price,
            Notes = x.Notes,
            CategoryName = x.Category.Name
        })
        .ToList();
}
```

Tiếp đến, ta định nghĩa hàm ShowFoodsForNode nhận đầu vào là một nút của cây. Kiểm tra xem nút đó ở mức nào để gọi các hàm tương ứng đã định nghĩa ở trên. Lưu ý rằng, trong phương thức ShowCategories(), ta đã lưu các thông tin cần thiết vào nút của cây.

```
4 references | 0 changes | 0 authors, 0 changes
private void ShowFoodsForNode(TreeNode node)
{
    // Xóa danh sách thực đơn hiện tại khỏi listview
    lvwFood.Items.Clear();

    // Nếu node = null, không cần xử lý gì thêm
    if (node == null) return;

    // Tạo danh sách để chứa danh sách các món ăn tìm được
    List<FoodModel> foods = null;

    // Nếu nút được chọn trên TreeView tương ứng với
    // loại nhóm thức ăn (Category Type) (mức thứ 2 trên cây)
    if (node.Level == 1)
    {
        // Thì lấy danh sách món ăn theo loại nhóm
        var categoryType = (CategoryType)node.Tag;
        foods = GetFoodByCategoryType(categoryType);
    }
    else
    {
        // Ngược lại, lấy danh sách món ăn theo thể loại
        // Nếu nút được chọn là 'Tất cả' thì lấy hết
        var category = node.Tag as Category;
        foods = GetFoodByCategory(category?.Id);
    }

    // Gọi hàm để hiển thị các món ăn lên ListView
    ShowFoodsOnListView(foods);
}

1 reference | 0 changes | 0 authors, 0 changes
private void ShowFoodsOnListView(List<FoodModel> foods)
{
    // Duyệt qua từng phần tử của danh sách food
    foreach (var foodItem in foods)
    {
        // Tạo item tương ứng trên ListView
        var item = lvwFood.Items.Add(foodItem.Id.ToString());

        // Và hiển thị các thông tin của món ăn
        item.SubItems.Add(foodItem.Name);
        item.SubItems.Add(foodItem.Unit);
        item.SubItems.Add(foodItem.Price.ToString("##,###"));
        item.SubItems.Add(foodItem.CategoryName);
        item.SubItems.Add(foodItem.Notes);
    }
}
```

Để xử lý sự kiện chọn một nút của cây, ta làm như sau:

- Nhấp phải chuột vào TreeView, chọn Properties.
- Trong cửa sổ Properties, click chọn nút Events (hình tia sét)
- Sau đó nhấp đôi chuột vào sự kiện AfterSelect để tạo hàm xử lý sự kiện đó

Sau đó, gọi hàm ShowFoodsForNode() trong hàm xử lý sự kiện và truyền nút được chọn vào đối số của hàm. Nút được chọn được lưu trong thuộc tính Node của tham số e.

```
1 reference | 0 changes | 0 authors, 0 changes
private void tvwCategory_AfterSelect(object sender, TreeViewEventArgs e)
{
    ShowFoodsForNode(e.Node);
}
```

Nhấn phím F5 để chạy chương trình và kiểm tra kết quả.

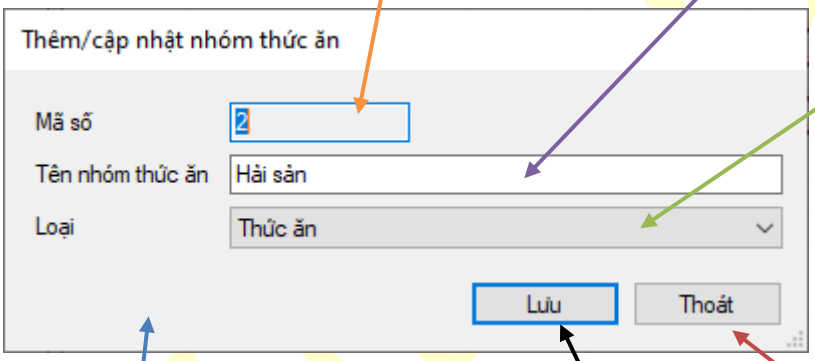
6. *Xây dựng Form cập nhật thông tin danh mục (nhóm) món ăn, đồ uống*

Phần này sẽ hướng dẫn bạn cách xây dựng một form mới – có tên UpdateCategoryForm – dùng để thêm mới hoặc cập nhật thông tin một nhóm thức ăn, đồ uống.

- Nhấp phải chuột vào tên dự án, chọn Add > Windows Form.
- Đặt tên cho form mới là UpdateCategoryForm.
- Thiết kế form có giao diện như sau:

TextBox
- Name: txtCategoryId
- ReadOnly: true

TextBox:
- Name: txtCategoryName



ComboBox:
- Name: cbbCategoryType
- DropDownStyle: DropDownList
- Items: Đồ uống, Thức ăn

Form:
- Name: UpdateCategoryForm
- AcceptButton: btnSave
- CancelButton: btnCancel
- MaximizeBox: false
- MinimizeBox: false
- StartPosition: CenterParent
- Text: Thêm/cập nhật nhóm thức ăn

Button:
- Name: btnCancel
- Anchor: Bottom, Right
- DialogResult: Cancel
- Text: &Thoát

Button
- Name: btnSave
- Anchor: Bottom, Right
- Text: &Lưu

Tiếp theo, trong lớp UpdateCategoryForm, khai báo thêm 2 biến:

```
private RestaurantContext _dbContext;
private int _categoryId;
```

Và cập nhật lại hàm khởi tạo của form như sau:

```
2 references | 0 changes | 0 authors, 0 changes
public UpdateCategoryForm(int? categoryId = null)
{
    InitializeComponent();

    _dbContext = new RestaurantContext();
    _categoryId = categoryId ?? 0;
}
```

Tham số categoryId có kiểu Nullable<int> và nhận giá trị mặc định là null. Điều này cho phép ta khởi tạo form mới mà không cần phải truyền vào mã danh mục món ăn. Điều này được áp dụng khi muốn tạo mới một danh mục.

Trong trường hợp tham số này nhận một giá trị dương, form sẽ ở chế độ cập nhật thông tin của một danh mục có sẵn. Lúc này, khi form được mở, ta cần hiển thị thông tin của danh mục đã được chọn.

Để truy vấn và hiển thị thông tin của một danh mục có mã cho trước, ta định nghĩa các phương thức sau đây:

```
2 references | 0 changes | 0 authors, 0 changes
private Category GetCategoryById(int categoryId)
{
    // Nếu ID được truyền vào là hợp lệ, ta tìm thông tin theo ID
    // Ngược lại, chỉ đơn giản trả về null, cho biết không thấy.
    return categoryId > 0 ? _dbContext.Categories.Find(categoryId) : null;
}

1 reference | 0 changes | 0 authors, 0 changes
private void ShowCategory()
{
    // Lấy thông tin của nhóm thức ăn
    var category = GetCategoryById(_categoryId);

    // Nếu không tìm thấy thông tin, không cần làm gì cả
    if (category == null) return;

    // Ngược lại, nếu tìm thấy, hiển thị lên form
    txtCategoryId.Text = category.Id.ToString();
    txtCategoryName.Text = category.Name;
    cbbCategoryType.SelectedIndex = (int)category.Type;
}
```

Muốn hiển thị thông tin của danh mục lên màn hình khi form được mở, ta xử lý sự kiện Load của UpdateCategoryForm như sau:

```
1 reference | 0 changes | 0 authors, 0 changes
private void UpdateCategoryForm_Load(object sender, EventArgs e)
{
    // Hiển thị thông tin nhóm thức ăn lên form
    ShowCategory();
}
```

Khi người dùng nhấn nút Lưu, ta cần phải kiểm tra dữ liệu nhập có hợp lệ hay không. Nếu dữ liệu không hợp lệ, chương trình cần hiển thị hộp thoại thông báo cho người dùng

biết. Ngược lại, ta lấy dữ liệu nhập và thực thi truy vấn để lưu chúng vào cơ sở dữ liệu. Trong form này, ta cần phải xử lý cả hai tình huống:

- Biến _categoryId > 0: nghĩa là khi form bật lên, bạn đã chọn một danh mục để cập nhật. Trong trường hợp này, chương trình cần cập nhật thông tin mới cho danh mục đó.
- Biến _categoryId = 0: nghĩa là form được mở ra để tạo mới một danh mục.

Ta định nghĩa thêm hai phương thức để kiểm tra dữ liệu nhập và lấy thông tin đã nhập như sau:

```
1 reference | 0 changes | 0 authors, 0 changes
private Category GetUpdatedCategory()
{
    // Tạo đối tượng Category với thông tin đã nhập
    var category = new Category()
    {
        Name = txtCategoryName.Text.Trim(),
        Type = (CategoryType)cbbCategoryType.SelectedIndex
    };

    // Gán giá trị của ID ban đầu (nếu đang cập nhật)
    if (_categoryId > 0)
    {
        category.Id = _categoryId;
    }

    return category;
}

1 reference | 0 changes | 0 authors, 0 changes
private bool ValidateUserInput()
{
    // Kiểm tra tên nhóm thức ăn đã được nhập hay chưa
    if (string.IsNullOrWhiteSpace(txtCategoryName.Text))
    {
        MessageBox.Show("Tên nhóm thức ăn không được để trống", "Thông báo");
        return false;
    }

    // Kiểm tra loại nhóm thức ăn đã được chọn hay chưa
    if (cbbCategoryType.SelectedIndex < 0)
    {
        MessageBox.Show("Bạn chưa chọn loại nhóm thức ăn", "Thông báo");
        return false;
    }

    return true;
}
```

Tiếp đến, nhấp đôi chuột vào nút Lưu (btnSave) để tạo hàm xử lý sự kiện Click và định nghĩa hàm đó như sau:

```
1 reference | 0 changes | 0 authors, 0 changes
private void btnSave_Click(object sender, EventArgs e)
{
    // Kiểm tra nếu dữ liệu nhập vào là hợp lệ
    if (ValidateUserInput())
    {
        // Thì lấy thông tin người dùng nhập vào
        var newCategory = GetUpdatedCategory();

        // Và thử tìm xem đã có nhóm thức ăn trong CSDL chưa
        var oldCategory = GetCategoryById(_categoryId);

        // Nếu chưa có (chưa tồn tại)
        if (oldCategory == null)
        {
            // Thì thêm nhóm thức ăn mới
            _dbContext.Categories.Add(newCategory);
        }
        else
        {
            // Ngược lại, ta chỉ cần cập nhật thông tin cần thiết
            oldCategory.Name = newCategory.Name;
            oldCategory.Type = newCategory.Type;
        }

        // Lưu các thay đổi xuống CSDL
        _dbContext.SaveChanges();

        // Đóng hộp thoại
        DialogResult = DialogResult.OK;
    }
}
```

7. *Thêm mới, cập nhật một danh mục món ăn*

Phần này sẽ hướng dẫn bạn cách sử dụng UpdateCategoryForm để thêm mới và cập nhật một danh mục món ăn.

- Để thêm mới một danh mục, ta xử lý sự kiện Click nút dấu + (btnAddCategory) trên MainForm và gọi hàm để hiển thị hộp thoại UpdateCategoryForm.
- Để cập nhật một danh mục, người dùng nhấp đôi chuột vào một nút trên cây. Chương trình cần xử lý sự kiện NodeMouseDoubleClick để mở form cập nhật.
- Trong cả 2 trường hợp, sau khi Lưu dữ liệu thành công, chương trình cần tải và hiện thị lại danh mục thức ăn trên TreeView.

Nhấp đôi chuột vào nút btnAddCategory và định nghĩa hàm xử lý sự kiện Click như sau:

```
1 reference | 0 changes | 0 authors, 0 changes
private void btnAddCategory_Click(object sender, EventArgs e)
{
    var dialog = new UpdateCategoryForm();
    if (dialog.ShowDialog(this) == DialogResult.OK)
    {
        ShowCategories();
    }
}
```

Nhấp phải chuột vào TreeView, chọn Properties > Events. Nhấp đôi chuột vào sự kiện NodeMouseDoubleClick để tạo hàm xử lý sự kiện. Định nghĩa hàm đó như sau:

```
1 reference | 0 changes | 0 authors, 0 changes
private void tvwCategory_NodeMouseDoubleClick(object sender, TreeNodeMouseClickEventArgs e)
{
    if (e.Node == null || e.Node.Level < 2 || e.Node.Tag == null) return;

    var category = e.Node.Tag as Category;
    var dialog = new UpdateCategoryForm(category?.Id);
    if (dialog.ShowDialog(this) == DialogResult.OK)
    {
        ShowCategories();
    }
}
```

Nhấn phím F5 để chạy chương trình. Click vào nút dấu + hoặc nhấp đôi chuột vào 1 nút trên cây để xem cách hoạt động của chương trình.

8. Xử lý việc xóa một món ăn và nạp lại danh sách món ăn, đồ uống

Việc nạp lại danh sách đồ ăn, thức uống tương đối đơn giản vì ta đã có hàm ShowFoodsForNode được định nghĩa ở phần trước. Trong màn hình thiết kế MainForm, nhấp đôi chuột vào nút R (btnReloadFood) để tạo hàm xử lý sự kiện Click. Gọi hàm ShowFoodsForNode với tham số là nút hiện đang được chọn trong TreeView.

```
1 reference | 0 changes | 0 authors, 0 changes
private void btnReloadFood_Click(object sender, EventArgs e)
{
    ShowFoodsForNode(tvwCategory.SelectedNode);
}
```

Để xóa một món ăn, đồ uống đang được chọn trong ListView, ta cần xử lý sự kiện Click cho nút "--" (btnDeleteFood) trong MainForm.

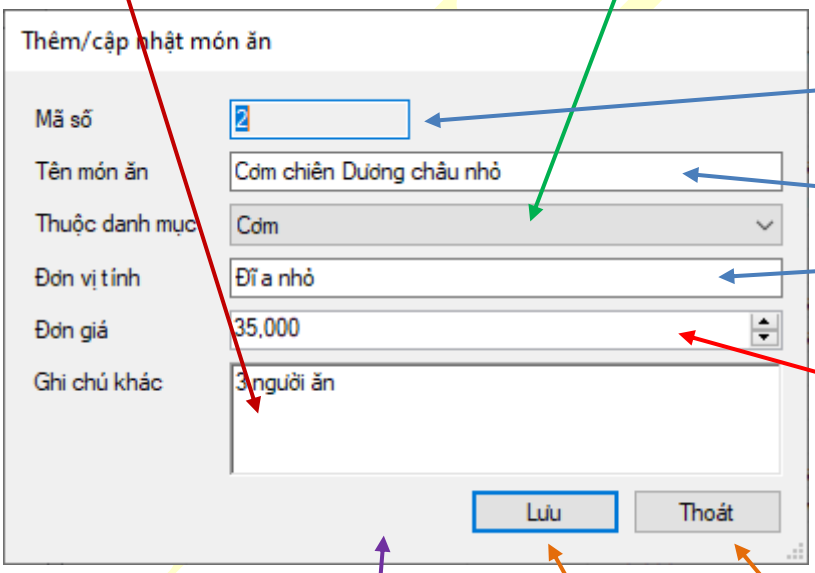
```
1 reference | 0 changes | 0 authors, 0 changes
private void btnDeleteFood_Click(object sender, EventArgs e)
{
    // Nếu không có món ăn nào được chọn, không cần làm gì cả
    if (lvwFood.SelectedItems.Count == 0) return;

    // Ngược lại, lấy mã số của món ăn được chọn
    var dbContext = new RestaurantContext();
    var selectedFoodId = int.Parse(lvwFood.SelectedItems[0].Text);
```

```
        // Truy vấn để lấy thông tin của món ăn đó
        var selectedFood = dbContext.Foods.Find(selectedFoodId);

        // Nếu tìm thấy thông tin món ăn
        if (selectedFood != null)
        {
            // Thì xóa nó khỏi cơ sở dữ liệu
            dbContext.Foods.Remove(selectedFood);
            dbContext.SaveChanges();

            // Và đồng thời xóa khỏi ListView
            lvwFood.Items.Remove(lvwFood.SelectedItems[0]);
        }
    }
```

Chạy chương trình và thử nhấn vào nút R và nút "--" ở bên phải để xem hoạt động của chương trình.

9. *Xây dựng form cập nhật thông tin món ăn, đồ uống*

Để thêm và cập nhật thông tin món ăn, đồ uống, ta xây dựng form mới như hình dưới đây



RichTextBox:
- Name: txtFoodNotes

ComboBox
- Name: cbbFoodCategory
- DropDownStyle: DropDownList

TextBox:
- Name: txtFoodId
- ReadOnly: True
- Name: txtFoodName
- Name: txtFoodUnit

NumericUpDown
- Name: nudFoodPrice
- Minimum: 0
- Maximum: 100 000 000
- Increment: 100
- ThousandsSeparator: true

Form: (UpdateFoodForm)
- AcceptButton: btnSave
- CancelButton: btnCancel
- ControlBox: False
- MaximizeBox, MinimizeBox: False
- StartPosition: CenterParent
- Text: Thêm/cập nhật món ăn

Button:
- Name: btnCancel, Text: &Thoát
- Anchor: Bottom, Right
- DialogResult: Cancel
- Name: btnSave, Text: &Lưu
- Anchor: Bottom, Right

- Nhấp phải chuột vào tên dự án, chọn Add > Windows Form.
- Đặt tên cho form mới là UpdateFoodForm.
- Thiết kế form có giao diện như hình trên
- Trong lớp UpdateFoodForm, khai báo thêm 2 biến

```
private RestaurantContext _dbContext;
private int _foodId;
```

- Khởi tạo hai biến đó trong hàm khởi tạo của form như sau

```
2 references | 0 changes | 0 authors, 0 changes
public UpdateFoodForm(int? foodId = null)
{
    InitializeComponent();

    _dbContext = new RestaurantContext();
    _foodId = foodId ?? 0;
}
```

Tiếp theo, ta cần lấy danh sách các nhóm đồ ăn, thức uống và nạp chúng vào ComboBox khi form được mở lên. Ngoài ra, nếu người dùng mở form để cập nhật thông tin một món ăn đã chọn thì chương trình cũng cần phải truy vấn và hiển thị thông tin món ăn đó.
Để nạp danh mục đồ ăn, thức uống vào ComboBox, ta định nghĩa hàm sau:

```
1 reference | 0 changes | 0 authors, 0 changes
private void LoadCategoriesToCombobox()
{
    // Lấy tất cả danh mục thức ăn, sắp tăng theo tên
    var categories = _dbContext.Categories
        .OrderBy(x => x.Name).ToList();

    // Nạp danh mục vào combobox, hiển thị tên cho người
    // dùng xem nhưng khi được chọn thì lấy giá trị là ID
    cbbFoodCategory.DisplayMember = "Name";
    cbbFoodCategory.ValueMember = "Id";
    cbbFoodCategory.DataSource = categories;
}
```

Sau đó gọi tới phương thức vừa tạo trong hàm xử lý sự kiện Load của UpdateFoodForm. Bạn cần nhấp đôi chuột vào form để tạo hàm xử lý sự kiện này.

```
1 reference | 0 changes | 0 authors, 0 changes
private void UpdateFoodForm_Load(object sender, EventArgs e)
{
    // Nạp danh sách nhóm thức ăn vào combobox
    LoadCategoriesToCombobox();
}
```

Việc truy vấn để lấy thông tin món ăn dựa vào mã số cho trước sẽ được thực hiện nhiều lần. Vì vậy, ta có thể định nghĩa phương thức dưới đây để có thể sử dụng lại nó.

```
2 references | 0 changes | 0 authors, 0 changes
private Food GetFoodById(int foodId)
{
    // Tìm món ăn theo mã số
    return foodId > 0 ? _dbContext.Foods.Find(foodId) : null;
}
```

Hàm này kiểm tra nếu mã số là hợp lệ thì lấy thông tin món ăn từ cơ sở dữ liệu. Ngược lại, chỉ đơn giản là trả về null.
Để hiển thị thông tin món ăn người dùng đã chọn lên form để cập nhật, ta định nghĩa thêm một phương thức mới:

```
1 reference | 0 changes | 0 authors, 0 changes
private void ShowFoodInformation()
{
    // Tìm món ăn theo mã số đã được truyền vào form
    var food = GetFoodById(_foodId);

    // Nếu không tìm thấy, không cần làm gì cả
    if (food == null) return;

    // Ngược lại, hiển thị thông tin món ăn lên form
    txtFoodId.Text = food.Id.ToString();
    txtFoodName.Text = food.Name;
    cbbFoodCategory.SelectedValue = food.FoodCategoryId;
    txtFoodUnit.Text = food.Unit;
    nudFoodPrice.Value = food.Price;
    txtFoodNotes.Text = food.Notes;
}
```

Sau đó, bổ sung lời gọi hàm tới phương thức này trong hàm xử lý sự kiện form load.

```
1 reference | 0 changes | 0 authors, 0 changes
private void UpdateFoodForm_Load(object sender, EventArgs e)
{
    // Nạp danh sách nhóm thức ăn vào combobox
    LoadCategoriesToCombobox();

    // Hiển thị thông tin món ăn lên form
    ShowFoodInformation();
}
```

Việc tiếp theo cần xử lý là kiểm tra dữ liệu nhập khi người dùng nhấn nút Lưu và hiển thị thông báo nếu dữ liệu nhập chưa đúng. Nếu mọi thông tin đều hợp lệ, ta cần tái tạo lại đối tượng Food từ dữ liệu nhập và thêm hoặc cập nhật vào cơ sở dữ liệu.

```
—references | 0 changes | 0 authors, 0 changes
private bool ValidateUserInput()
{
    // Kiểm tra tên món ăn đã được nhập hay chưa
    if (string.IsNullOrWhiteSpace(txtFoodName.Text))
    {
        MessageBox.Show("Tên món ăn, đồ uống không được để trống", "Thông báo");
        return false;
    }

    // Kiểm tra đơn vị tính đã được nhập hay chưa
    if (string.IsNullOrWhiteSpace(txtFoodUnit.Text))
    {
        MessageBox.Show("Đơn vị tính không được để trống", "Thông báo");
        return false;
    }
```

```
    // Kiểm tra giá món ăn đã được nhập hay chưa
    if (nudFoodPrice.Value.Equals(0))
    {
        MessageBox.Show("Giá của thức ăn phải lớn hơn 0", "Thông báo");
        return false;
    }

    // Kiểm tra nhóm món ăn đã được chọn hay chưa
    if (cbbFoodCategory.SelectedIndex < 0)
    {
        MessageBox.Show("Bạn chưa chọn nhóm thức ăn", "Thông báo");
        return false;
    }

    return true;
}
```

```
1 reference | 0 changes | 0 authors, 0 changes
private Food GetUpdatedFood()
{
    // Tạo đối tượng food với thông tin được lấy từ
    // các điều khiển trên form
    var food = new Food()
    {
        Name = txtFoodName.Text.Trim(),
        FoodCategoryId = (int)cbbFoodCategory.SelectedValue,
        Unit = txtFoodUnit.Text,
        Price = (int)nudFoodPrice.Value,
        Notes = txtFoodNotes.Text
    };

    // Gán giá trị của ID ban đầu (nếu đang cập nhật)
    if (_foodId > 0)
    {
        food.Id = _foodId;
    }

    return food;
}
```

Cuối cùng, để hoàn thành form cập nhật thông tin món ăn, bạn cần viết mã cho hàm xử lý khi người dùng nhấn nút Lưu. Nhấp đôi chuột vào nút Lưu (btnSave) để tạo hàm xử lý sự kiện Click và bổ sung đoạn mã sau:

```
1 reference | 0 changes | 0 authors, 0 changes
private void btnSave_Click(object sender, EventArgs e)
{
    // Kiểm tra nếu dữ liệu nhập vào là hợp lệ
    if (ValidateUserInput())
    {
        // Thì lấy thông tin người dùng nhập vào
        var newFood = GetUpdatedFood();

        // Và thử tìm xem đã có món ăn trong CSDL chưa
        var oldFood = GetFoodById(_foodId);

        // Nếu chưa có (chưa tồn tại)
```

```
            if (oldFood == null)
            {
                // Thì thêm món ăn mới
                _dbContext.Foods.Add(newFood);
            }
            else
            {
                // Ngược lại, cập nhật thông tin món ăn
                oldFood.Name = newFood.Name;
                oldFood.FoodCategoryId = newFood.FoodCategoryId;
                oldFood.Unit = newFood.Unit;
                oldFood.Price = newFood.Price;
                oldFood.Notes = newFood.Notes;
            }

            // Lưu các thay đổi xuống CSDL
            _dbContext.SaveChanges();

            // Đóng hộp thoại
            DialogResult = DialogResult.OK;
        }
    }
```

10. Gọi form cập nhật thông tin món ăn, đồ uống

Trong phần này, ta bổ sung hai phương thức để xử lý sự kiện nhấn nút + (btnAddFood) để thêm món ăn mới và sự kiện nhấn đôi chuột vào một item của ListView để cập nhật thông tin món ăn, đồ uống.

```
1 reference | 0 changes | 0 authors, 0 changes
private void btnAddFood_Click(object sender, EventArgs e)
{
    var dialog = new UpdateFoodForm();
    if (dialog.ShowDialog(this) == DialogResult.OK)
    {
        ShowFoodsForNode(tvwCategory.SelectedNode);
    }
}

1 reference | 0 changes | 0 authors, 0 changes
private void lvwFood_DoubleClick(object sender, EventArgs e)
{
    if (lvwFood.SelectedItems.Count == 0) return;

    var foodId = int.Parse(lvwFood.SelectedItems[0].Text);
    var dialog = new UpdateFoodForm(foodId);
    if (dialog.ShowDialog(this) == DialogResult.OK)
    {
        ShowFoodsForNode(tvwCategory.SelectedNode);
    }
}
```

Chạy chương trình và kiểm tra kết quả.

III. Bài tập

1. Thiết kế Form: RoleForm và viết hàm xử lý để
   a. Hiển thị danh sách các nhóm (hay vai trò) người dùng
   b. Thêm role mới, cập nhật một role đã có.
   c. Xem danh sách người dùng thuộc role được chọn

2. Thiết kế Form: AccountForm và viết hàm xử lý để
   a. Xem danh sách tài khoản, có cho phép tìm theo role và tên người dùng
   b. Thêm một tài khoản mới, trong đó cho phép chọn role cho người dùng
   c. Cập nhật thông tin của một tài khoản
   d. Reset mật khẩu cho tài khoản
   e. Thay đổi mật khẩu của tài khoản
   f. Click chuột phải vào một tài khoản hiển thị menu sau:

Xóa tài khoản
Xem danh sách vai trò

   Trong đó:
   - Nếu chọn Xóa tài khoản thì xóa Password, gán về null. Lúc này, người dùng không thể đăng nhập được, tài khoản ở trạng thái Inactive.
   - Xem danh sách vai trò: Mở một Form mới để hiển thị các vai trò (role) được gán cho tài khoản này

3. Thiết kế Form: TableForm và viết các hàm xử lý để
   a. Hiển thị danh sách các bàn
   b. Xem hóa đơn hiện tại của một bàn
   c. Thêm một bàn mới
   d. Cập nhật thông tin của bàn
   e. Xóa một bàn.
   f. Khi nhắp phải chuột vào một bàn, hiển thị menu sau

Xóa bàn
Xem danh mục hóa đơn
Xem nhật ký hóa đơn

   Trong đó:
   - Nếu chọn Xóa bàn thì dữ liệu về bàn đó sẽ bị xóa khỏi cơ sở dữ liệu
   - Xem danh mục hóa đơn: Mở một Form mới, phần bên trái của Form là một ListBox chứa ngày lập của các hóa đơn. Khi nhấn chuột vào ngày nào thì hiển thị thông tin chi tiết (cả danh mục sản phẩm được mua) của hóa đơn ở phần bên phải Form.
   - Xem nhật ký mua hàng: Liệt kê số lượng hóa đơn, tổng số tiền, tổng thuế, tổng giảm giá của tất cả các hóa đơn, thông tin liên quan đến từng hóa đơn như ngày lập, tên nhân viên lập hóa đơn. (sử dụng ListView hoặc DataGridView)

4. Thiết kế Form: ActivityForm và viết các hàm xử lý để xem trạng thái hoạt động của các bàn, quản lý việc đặt món cho các bàn, thanh toán tiền lập hóa đơn, in hóa đơn.

5. Thiết kế Form: MenuForm để xem và in thực đơn.