

МИНИСТЕРСТВО НА ОБРАЗОВАНИЕТО И НАУКАТА

**ПРОФЕСИОНАЛНА ГИМНАЗИЯ “ГЕН. ВЛАДИМИР ЗАИМОВ” гр. СОПОТ**

4330 гр. Сопот, ул. ”Иван Вазов” №1, тел./факс: /03134/ 83-31, 83-32, e-mail: [pgzaimov@yahoo.com](mailto:pgzaimov@yahoo.com)

**ПРОЕКТ**

**Заглавие на проекта: Доставка на храна**

**Задача № 4**

*Ученик: Габриела Василева*

***Професия:*** *„Системен програмист“*

***Специалност:*** *„Системно програмиране“*

Сопот, 2024 г.

**ТОЧКА 1.**

**/Описание на БД/**

Създадена е технологията ORM /CODE FIRST/ в MS SQL Server.

В сървъра има база данни - Delivery с таблици: Dishes и DishType за доставка на храна.

1. Всяка таблица съдържа следните полета:

* Таблица Dishes:
* Номер
* Име на ястието
* Описание
* Цена
* Грамаж
* Вид /от долната таблица/



public class Dish

{

public int Id { get; set; }

public string Name { get; set; }

public string Description { get; set; }

public decimal Price { get; set; }

public int Weight { get; set; }

//M:1

public int DishTypeId { get; set; } //FK

public DishType DishTypes { get; set; } //таблицата, с която се осъществява връзката

}

Фигура 1 /Диаграма на клас Dish/

* Таблица DishType
* Номер на вида ястие
* Име на типа ястие

****

public class DishType

{

public int Id { get; set; }

public string Name { get; set; }

//1:M

public ICollection<Dish> Dishes { get; set; }

}

Фигура 2 /Диаграма на клас DishType/

1. Създаване на DbContext:

* DishesContext е клас, който наследява DbContext и представлява връзката с базата данни.
* Като наследим DbContext се предизвиква инсталиране на пакета „Entity Framework“.
* Пакетът съдържа всички необходими инструменти за работа с база от данни: описание на таблици и релации, изпълнение на заявки, менежиране на данни и тяхната сигурност.
* Може да се извърши от Tools>NuGet Package Manager>Managе NuGet Packages for Solution…

public class DishesContext:DbContext

{

public DishesContext() : base("DishesContext")

{

}

public DbSet<Dish> Dishes { get; set; } //Таблица Dishes

public DbSet<DishType> DishTypes { get; set; } //Таблица DishTypes

}

1. Конфигуриране на базата данни:

* Добавяме селекцията за конфигуриране на връзката във файла „App.config“ от структурата приложението

<connectionStrings>

<add name="DishesContext"

connectionString = "Data source=(localdb)\MSSQLLocalDB;

Initial Catalog=Delivery;

Integrated security=True;"

providerName="System.Data.SqlClient"/>

</connectionStrings>

1. Миграции – изпълняваме последователно командите:

PM>Enable Migrations

PM> Add-migration имеНаМиграция

PM> Update-database

**ТОЧКА 2.**

**/ Отделните екрани с кратко описание за работа с приложението./**

1. Обяснение на класа DishController:

* Създаване на инстанция на контекста за базата данни:
* Този ред създава екземпляр на контекста на базата данни DishesContext, който ще се използва за извършване на операции с базата данни.

private DishesContext \_dishesDbContext = new DishesContext();

* Получаване на ястие по идентификатор:
* Този метод получава ястие по даден идентификатор id. Първо, се търси ястието в базата данни с Find(id). Ако ястието е намерено, се зарежда свързаният DishType с помощта на Reference(x => x.DishTypes).Load(). Накрая, намереното ястие се връща.

public Dish Get(int id)

{

Dish findedDish = \_dishesDbContext.Dishes.Find(id);

if (findedDish != null)

{

\_dishesDbContext.Entry(findedDish).Reference(x => x.DishTypes).Load();

}

return findedDish;

}

* Получаване на всички ястия:
* Този метод връща списък с всички ястия, като включва и свързаните DishType обекти. Използва се методът Include("DishTypes") за да се включат свързаните типове ястия, след което резултатът се конвертира в списък с ToList().

public List<Dish> GetAll()

{

return \_dishesDbContext.Dishes.Include("DishTypes").ToList();

}

* Създаване на ново ястие:
* Този метод добавя ново ястие към базата данни. Първо, новото ястие се добавя към контекста с Add(dish), след което промените се записват в базата данни с SaveChanges().

public void Create(Dish dish)

{

\_dishesDbContext.Dishes.Add(dish);

\_dishesDbContext.SaveChanges();

}

* Актуализиране на съществуващо ястие:
* Този метод актуализира съществуващо ястие. Първо, се търси ястието по идентификатор id. Ако ястието е намерено, се актуализират неговите свойства с новите стойности от dish. Накрая, промените се записват в базата данни с SaveChanges().

public void Update(int id, Dish dish)

{

Dish findedDish = \_dishesDbContext.Dishes.Find(id);

if (findedDish == null)

{

return;

}

findedDish.Name = dish.Name;

findedDish.Description = dish.Description;

findedDish.Price = dish.Price;

findedDish.Weight = dish.Weight;

findedDish.DishTypeId = dish.DishTypeId;

\_dishesDbContext.SaveChanges();

}

* Изтриване на съществуващо ястие:
* Този метод изтрива съществуващо ястие. Първо, се търси ястието по идентификатор id. Ако ястието е намерено, се премахва от контекста с Remove(findedDish), след което промените се записват в базата данни с SaveChanges().

public void Delete(int id)

{

Dish findedDish = \_dishesDbContext.Dishes.Find(id);

\_dishesDbContext.Dishes.Remove(findedDish);

\_dishesDbContext.SaveChanges();

}

1. Обяснение на класа DishTypeController:

* Създаване на инстанция на контекста за базата данни:
* Този ред създава екземпляр на контекста на базата данни DishesContext, който ще се използва за извършване на операции с базата данни.

private DishesContext \_dishesDbContext = new DishesContext();

* Получаване на всички типове ястия:
* Този метод връща списък с всички типове ястия от базата данни. Използва се методът ToList(), за да се конвертира резултатът в списък.

public List<DishType> GetAllDishTypes()

{

return \_dishesDbContext.DishTypes.ToList();

}

* Получаване на име на тип ястие по идентификатор:
* Този метод връща името на тип ястие по даден идентификатор id. Първо, се търси типът ястие в базата данни с Find(id). След като типът е намерен, се връща неговото име Name.

public string GetDishTypeById(int id)

{

return \_dishesDbContext.DishTypes.Find(id).Name;

}

**ТОЧКА 3.**

**/Анализ на задачата/**

1. Интерфейс:

Картина, която съдържа текст, екранна снимка, софтуер, номер

Описанието е генерирано автоматично

Фигура 3 /Интерфейс/

1.1.Контроли:

* ComboBox: Използван за избор на тип ястие от предварително въведени типове, което улеснява потребителите при въвеждането на нови данни.
* TextBox: За въвеждане на текстови данни като номер, име, описание, цена и грамаж.
* Buttons: За изпълнение на различни действия като добавяне, редактиране, намиране, изтриване, изчистване и извеждане на всички на записи:
* Add: Добавяне на ново ястие към базата данни.
* Update: Редактиране на избраното ястие.
* Delete: Изтриване на избраното ястие.
* Find: Намиране на ястие по Id.
* Clear: Изчистване на полетата за въвеждане.
* Select All: Извеждане на всички записи от базата данни.
* Label: За етикетиране на полетата
* ListBox: Показва списък с всички ястия, налични в базата данни. Всяко ястие е представено с номер и име.

1. Описание на Form.cs кода:

* Създаване на инстанция на контролерите:
* DishController и DishTypeController са инстанции на контролерите за работа с ястия и типове ястия.

DishController dishesController = new DishController();

DishTypeController dishTypesController = new DishTypeController();

* Събитие за зареждане на формата:
* Form1\_Load събитието се изпълнява при зареждане на формата. То извлича всички типове ястия и ги задава като източник на данни (DataSource) за ComboBox (cmbDishType).

private void Form1\_Load(object sender, EventArgs e)

{

List<DishType> allDishTypes = dishTypesController.GetAllDishTypes();

cmbDishType.DataSource = allDishTypes;

cmbDishType.DisplayMember = "Name";

cmbDishType.ValueMember = "Id";

}

* Зареждане на записи:
* LoadRecord(Dish dish) зарежда данните от дадено ястие в текстовите полета и комбинирания списък.

private void LoadRecord(Dish dish)

{

txtId.Text = dish.Id.ToString();

txtName.Text = dish.Name;

txtDescription.Text = dish.Description;

txtPrice.Text = dish.Price.ToString();

txtWeight.Text = dish.Weight.ToString();

cmbDishType.Text = dish.DishTypes.Name;

}

* Изчистване на записи:
* ClearScreen() изчиства всички текстови полета и комбинирания списък.

private void ClearScreen()

{

txtId.Clear();

txtName.Clear();

txtPrice.Clear();

txtWeight.Clear();

txtDescription.Clear();

cmbDishType.Text = "";

}

* Изчистване на формата:
* Изчиства всички полета на формата.

private void btnClear\_Click(object sender, EventArgs e)

{

ClearScreen();

}

2.1. CRUD операции:

* Добавяне на ново ястие:
* Проверява се дали полето за име на ястие не е празно.
* Създава се нов обект Dish и се попълва с данните от формата.
* Извиква се методът Create() на контролера DishController за добавяне на новото ястие в базата данни.
* След добавяне се показва съобщение за успешен запис и се изчиства формата.

private void btnAdd\_Click(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(txtName.Text) || txtName.Text == "")

{

MessageBox.Show("Въведете данни!");

txtName.Focus();

return;

}

Dish newDish = new Dish();

newDish.Name = txtName.Text;

newDish.Description = txtDescription.Text;

newDish.Price = decimal.Parse(txtPrice.Text);

newDish.Weight = int.Parse(txtWeight.Text);

newDish.DishTypeId = (int)cmbDishType.SelectedValue;

dishesController.Create(newDish);

MessageBox.Show("Записът е успешно добавен!");

ClearScreen();

btnSelectAll\_Click(sender, e);

}

* Показване на всички записи:
* Извличат се всички записи за ястията.
* Показват се в ListBox, където всяко ястие е представено с номер и име.

private void btnSelectAll\_Click(object sender, EventArgs e)

{

List<Dish> allDishes = dishesController.GetAll();

listBox.Items.Clear();

foreach (var item in allDishes)

{

listBox.Items.Add($"{item.Id}, {item.Name}");

}

}

* Намиране на ястие:
* Проверява се дали полето за Id е валидно и дали съществува такъв запис.
* Ако записът съществува, данните за ястието се зареждат.

private void btnFind\_Click(object sender, EventArgs e)

{

int findId = 0;

if (string.IsNullOrEmpty(txtId.Text) || !txtId.Text.All(char.IsDigit))

{

MessageBox.Show("Въведете Id за търсене!");

txtId.Focus();

return;

}

else

{

findId = int.Parse(txtId.Text);

}

Dish findedDish = dishesController.Get(findId);

if (findedDish == null)

{

MessageBox.Show("НЯМА ТАКЪВ ЗАПИС в БД! \nВъведете Id за търсене!");

txtId.Focus();

return;

}

LoadRecord(findedDish);

}

* Редактиране на съществуващо ястие:
* Проверява се дали полето за Id е валидно и дали съществува такъв запис.
* Ако полето за име на ястие е празно, зарежда данните за ястието.
* Ако не е празно, създава се нов обект Dish, попълва се с данните от формата и се извиква методът Update() на контролера за обновяване на ястието.

private void btnUpdate\_Click(object sender, EventArgs e)

{

int findId = 0;

if (string.IsNullOrEmpty(txtId.Text) || !txtId.Text.All(char.IsDigit))

{

MessageBox.Show("Въведете Id за търсене!");

txtId.Focus();

return;

}

else

{

findId = int.Parse(txtId.Text);

}

if (string.IsNullOrEmpty(txtName.Text))

{

Dish findedDish = dishesController.Get(findId);

if (findedDish == null)

{

MessageBox.Show("НЯМА ТАКЪВ ЗАПИС в БД! \nВъведете Id за търсене!");

txtId.Focus();

return;

}

LoadRecord(findedDish);

}

else

{

Dish updatedDish = new Dish();

updatedDish.Name = txtName.Text;

updatedDish.Description = txtDescription.Text;

updatedDish.Price = decimal.Parse(txtPrice.Text);

updatedDish.Weight = int.Parse(txtWeight.Text);

updatedDish.DishTypeId = (int)cmbDishType.SelectedValue;

dishesController.Update(findId, updatedDish);

}

MessageBox.Show("Записът е успешно update-нат!");

ClearScreen();

btnSelectAll\_Click(sender, e);

}

* Изтриване на ястие:
* Проверява се дали полето за Id е валидно и дали съществува такъв запис.
* Ако записът съществува, данните за ястието се зареждат и се показва диалогов прозорец за потвърждение.
* Ако потребителят потвърди, се извиква методът Delete() на контролера за изтриване на ястието.

private void btnDelete\_Click(object sender, EventArgs e)

{

int findId = 0;

if (string.IsNullOrEmpty(txtId.Text) || !txtId.Text.All(char.IsDigit))

{

MessageBox.Show("Въведете Id за търсене!");

txtId.Focus();

return;

}

else

{

findId = int.Parse(txtId.Text);

}

Dish findedDish = dishesController.Get(findId);

if (findedDish == null)

{

MessageBox.Show("НЯМА ТАКЪВ ЗАПИС в БД! \nВъведете Id за търсене!");

txtId.Focus();

return;

}

LoadRecord(findedDish);

DialogResult answer = MessageBox.Show("Наистина ли искате да изтриете запис No " + findId + "?",

"?",

MessageBoxButtons.YesNo,

MessageBoxIcon.Question);

if (answer == DialogResult.Yes)

{

dishesController.Delete(findId);

}

MessageBox.Show("Записът е изтрит!");

ClearScreen();

btnSelectAll\_Click(sender, e);

}