

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: Плодова борса**

**Задача № 10**

*Ученик: Мартин Маринов и Александър Димов*

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

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

Сопот, 2024 г.

**ТОЧКА 1.**

**Описание на класовете-таблици и DbContext**

**Клас: Fruit**

|  |
| --- |
| * **namespace Fruit\_Storage.Models** * **{** * **public class Fruit** * **{** * **public int Id { get; set; }** * **public string Name { get; set; }** * **public string Description { get; set; }** * **public float Price { get; set; }** * **public int FruitTypeId { get; set; }** * **public FruitType FruitType { get; set; }** * **public override string ToString()** * **{** * **return Name;**   **}**   * **}** * **}** |

 **Описание: Id: Целочислен идентификатор на плода (примарен ключ). Name: Име на плода. Description: Описание на плода. Price: Цена на плода. FruitTypeId: Целочислен външен ключ, свързващ плода с определен вид плод. FruitType: Навигационно свойство към обект от клас FruitType, представляващ вида плод**

**Клас: FruitType**

|  |
| --- |
| * **namespace Fruit\_Storage.Models** * **{** * **public class FruitType** * **{** * **public int Id { get; set; }** * **public string Name { get; set; }** * **public ICollection<Fruit> Fruits { get; set; }** * **public override string ToString()** * **{** * **return Name;** * **}** * **}** * **}** |

 **Описание: Id: Целочислен идентификатор на вида плод (примарен ключ). Name: Име на вида плод. Fruits: Колекция от плодове, свързани с този вид.**

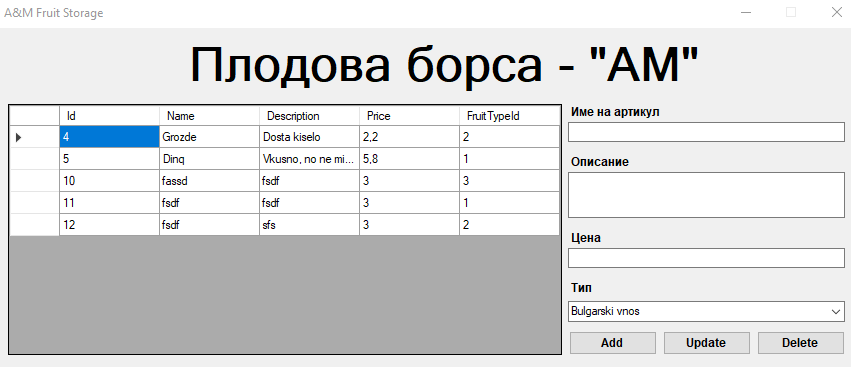
**Клас: FruitsContext**

|  |
| --- |
| * **using Microsoft.EntityFrameworkCore;** * **namespace Fruit\_Storage** * **{** * **public class FruitsContext : DbContext** * **{** * **public DbSet<Fruit> Fruits { get; set; }** * **public DbSet<FruitType> FruitTypes { get; set; }**   **protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)**  **{**  **if (!optionsBuilder.IsConfigured)**  **{ optionsBuilder.UseSqlServer(@"Server=(localdb)\MSSQLLocalDB;Database=Fruit\_Storage;Integrated Security=true");**  **}**  **}**  **}** |

 **Описание: FruitsContext: Клас, който представлява контекста на базата данни и управлява взаимодействието с нея чрез Entity Framework Core. DbSet<Fruit>: Таблица за плодовете в базата данни. DbSet<FruitType>: Таблица за видовете плодове в базата данни. OnConfiguring: Метод, който конфигурира връзката към базата данни.**

**ГЛАВА 2.**

**Функционално описание: Отделните екрани с кратко описание за работа с приложението.**

****

 **Функционално описание Екрани и тяхното функционалностно описание Екран: Главен прозорец (Form1)**

 **Преглед на списък с плодове: В този прозорец се зарежда списък от всички плодове, запазени в базата данни.**

 **Плодовете са представени в DataGridView контрола. При зареждане на прозореца се извикват методи за зареждане на плодовете и типовете плодове (LoadData() и LoadFruitTypes()).**

 **Добавяне на нов плод: Потребителят може да въведе ново име, описание, цена и да избере тип на плода чрез комбо бокс контрол.**

 **След въвеждане на данните, натискането на бутон за добавяне (button1\_Click) води до добавяне на нов плод в базата данни. Редактиране на плод: Избира се плод от списъка в DataGridView.**

 **Информацията за избрания плод се показва в текстовите полета и комбо бокса за редактиране (button2\_Click). След въвеждане на нова информация и натискане на бутон за редактиране, данните на плода се обновяват в базата данни.**

 **Изтриване на плод: Избира се плод от списъка в DataGridView. Чрез натискане на бутон за изтриване (button3\_Click), се премахва избраният плод от базата данни.**

**ГЛАВА 3.**

**Анализ на задачата**

#### Технология за връзка с базата данни

**За връзка с базата данни в този проект използвахме Entity Framework (EF) и ADO.NET. EF е ORM (Object-Relational Mapping) технология, която улеснява работата с базата данни, като позволява работа с обекти, вместо директно с таблици. За някои операции, като извличане на данни, използвахме и ADO.NET за по-добър контрол върху изпълнението на SQL заявки.**

#### Обяснение на методите и кода

##### Клас: FruitLogic

1. **GetFruits**:

**public DataTable GetFruits()**

**{**

**string query = "SELECT \* FROM Fruit";**

**using (SqlConnection conn = new SqlConnection(connString))**

**{**

**conn.Open();**

**SqlDataAdapter adapter = new SqlDataAdapter(query, conn);**

**DataTable table = new DataTable();**

**adapter.Fill(table);**

**return table;**

**}**

**}**

 **Описание**: Извлича всички плодове от базата данни и ги връща като DataTable.

 **Използване**: Този метод се използва за зареждане на плодовете в DataGridView при стартиране на приложението или след CRUD операции.

**AddFruit**:

public void AddFruit(Fruit fruit)

{

using (SqlConnection conn = new SqlConnection(connString)) { conn.Open(); string insertQuery = "INSERT INTO Fruit (Name, Description, Price, FruitTypeId) VALUES (@Name, @Description, @Price, @FruitTypeId)";

using (SqlCommand cmd = new SqlCommand(insertQuery, conn))

{

cmd.Parameters.AddWithValue("@Name", fruit.Name);

cmd.Parameters.AddWithValue("@Description", fruit.Description);

cmd.Parameters.AddWithValue("@Price", fruit.Price);

cmd.Parameters.AddWithValue("@FruitTypeId", fruit.FruitTypeId);

cmd.ExecuteNonQuery();

}

}

}

* **Описание**: Добавя нов запис за плод в базата данни.
* **Използване**: Този метод се извиква, когато потребителят натисне бутона за добавяне на нов плод

.

**UpdateFruit**:

public void UpdateFruit(Fruit fruit)

{

using (SqlConnection conn = new SqlConnection(connString))

{

conn.Open();

string updateQuery = "UPDATE Fruit SET Name = @Name, Description = @Description, Price = @Price, FruitTypeId = @FruitTypeId WHERE Id = @Id";

using (SqlCommand cmd = new SqlCommand(updateQuery, conn))

{

cmd.Parameters.AddWithValue("@Name", fruit.Name);

cmd.Parameters.AddWithValue("@Description", fruit.Description);

cmd.Parameters.AddWithValue("@Price", fruit.Price);

cmd.Parameters.AddWithValue("@FruitTypeId", fruit.FruitTypeId);

cmd.Parameters.AddWithValue("@Id", fruit.Id);

cmd.ExecuteNonQuery();

}

}

}

 **Описание**: Обновява съществуващ запис за плод в базата данни.

 **Използване**: Този метод се извиква, когато потребителят натисне бутона за редактиране на избран плод.

**DeleteFruit**:

public void DeleteFruit(int fruitId)

{

using (SqlConnection conn = new SqlConnection(connString))

{

conn.Open();

string deleteQuery = "DELETE FROM Fruit WHERE Id = @Id";

using (SqlCommand cmd = new SqlCommand(deleteQuery, conn))

{

cmd.Parameters.AddWithValue("@Id", fruitId);

cmd.ExecuteNonQuery();

}

}

}

 **Описание**: Изтрива запис за плод от базата данни по зададен идентификатор.

 **Използване**: Този метод се извиква, когато потребителят натисне бутона за изтриване на избран плод.

#### Избор на контроли и интерфейс

1. **DataGridView**:
   * Използва се за визуализация на данните за плодовете. Позволява лесно преглеждане и избор на записи за редактиране или изтриване.
2. **TextBox**:
   * Използва се за въвеждане на информация за името, описанието и цената на плода.
3. **ComboBox**:
   * Използва се за избор на типа на плода. Включва списък от предварително заредени типове плодове от базата данни.
4. **Button**:
   * Три бутона за добавяне, редактиране и изтриване на плодове.

#### Описание на CRUD операциите

##### Добавяне на плод (Create)

private void button1\_Click(object sender, EventArgs e)

{

string name = textBox1.Text;

string description = textBox2.Text;

float price;

int typeId;

if (string.IsNullOrWhiteSpace(name) || string.IsNullOrWhiteSpace(description) ||

!float.TryParse(textBox3.Text, out price) || comboBox1.SelectedValue == null ||

!int.TryParse(comboBox1.SelectedValue.ToString(), out typeId))

{

MessageBox.Show("Невалиден вход");

return;

}

try

{

var fruit = new Fruit

{

Name = name,

Description = description,

Price = price,

FruitTypeId = typeId

};

fruitController.AddFruit(fruit);

LoadData();

textBox1.Clear();

textBox2.Clear();

textBox3.Clear();

comboBox1.SelectedIndex = -1;

}

catch (Exception ex)

{

MessageBox.Show("Проблем с вкарването на запис: " + ex.Message);

}

}

**Описание**: Валидира въвеждането, създава нов обект Fruit, добавя го в базата данни чрез fruitController.AddFruit и презарежда данните.

Обновяване на плод (Update):

private void button2\_Click(object sender, EventArgs e)

{

if (dataGridView1.CurrentCell == null)

{

MessageBox.Show("Избери ред");

return;

}

int index = dataGridView1.CurrentCell.RowIndex;

if (index < 0)

{

MessageBox.Show("Невалиден ред");

return;

}

DataGridViewRow selectedRow = dataGridView1.Rows[index];

var idCell = selectedRow.Cells["Id"];

if (idCell == null || idCell.Value == DBNull.Value)

{

MessageBox.Show("Невалиден Primary key");

return;

}

int id = Convert.ToInt32(idCell.Value);

string name = string.IsNullOrWhiteSpace(textBox1.Text) ? selectedRow.Cells["Name"].Value.ToString() : textBox1.Text;

string description = string.IsNullOrWhiteSpace(textBox2.Text) ? selectedRow.Cells["Description"].Value.ToString() : textBox2.Text;

float price;

if (!float.TryParse(textBox3.Text, out price))

{

price = Convert.ToSingle(selectedRow.Cells["Price"].Value);

}

int typeId;

if (comboBox1.SelectedValue == null || !int.TryParse(comboBox1.SelectedValue.ToString(), out typeId))

{

typeId = Convert.ToInt32(selectedRow.Cells["FruitTypeId"].Value);

}

try

{

var fruit = new Fruit

{

Id = id,

Name = name,

Description = description,

Price = price,

FruitTypeId = typeId

};

fruitController.UpdateFruit(fruit);

LoadData();

textBox1.Clear();

textBox2.Clear();

textBox3.Clear();

comboBox1.SelectedIndex = -1;

}

catch (Exception ex)

{

MessageBox.Show("Проблем с обновяването на записа: " + ex.Message);

}

}

**Описание**: Проверява избран ред, актуализира полетата и обновява записа в базата данни чрез fruitController.UpdateFruit.

Изтриване на плод (Delete)

private void button3\_Click(object sender, EventArgs e)

{

if (dataGridView1.CurrentCell == null)

{

MessageBox.Show("Избери ред");

return;

}

int index = dataGridView1.CurrentCell.RowIndex;

if (index < 0)

{

MessageBox.Show("Невалиден ред");

return;

}

DataGridViewRow selectedRow = dataGridView1.Rows[index];

var idCell = selectedRow.Cells["Id"];

if (idCell == null || idCell.Value == DBNull.Value)

{

MessageBox.Show("Невалиден Primary key");

return;

}

int id = Convert.ToInt32(idCell.Value);

try

{

fruitController.DeleteFruit(id);

LoadData();

}

catch (Exception ex)

{

MessageBox.Show("Проблем с изтриването на записа: " + ex.Message);

}

}

**Описание**: Проверява избран ред и изтрива записа от базата данни чрез fruitController.DeleteFruit.