

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:**

**Хранителен Магазин**

**Задача № 1**

*Ученици: Божидар Найденов Четинов/*

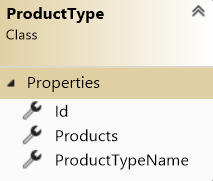
*Николай Николаев Инджов*

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

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

Сопот, 2024 г.

**ТОЧКА 1.**



**Class ProductType**

**Id**: Уникален идентификатор за всеки тип продукт.  
Това е основният ключ (Primary Key).

**ProductTypeName**: Име на типа продукт

(например: "Хранителен продукт", "Напитка").

**Products**: Колекция от продукти,  
които са свързани с този тип продукт.  
Това създава връзка one-to-many с таблицата Product.

**Class Product**

A screenshot of a phone

Description automatically generated**Id**: Уникален идентификатор за всеки продукт.  
Това е основният ключ (Primary Key).

**Properties: Brand, Description, ExpirationDate,  
Price, ProductTypeId, ProductTypes**

**Brand**: Марка на продукта.

**Description**: Описание на продукта.

**Price**: Цена на продукта.

**ExpirationDate**: Дата на изтичане на продукта.

**ProductTypeId**: Външен ключ,  
който сочи към ProductType.Id.

**ProductTypes**: Обект от тип ProductType,  
който представлява връзката към таблицата ProductType.

A screenshot of a computer

Description automatically generated**Class MagazineDbContext**

**MagazineDbContext**: Клас, който наследява  
DbContext.Това е основният клас  
за комуникация с базата данни,  
използвайки **Entity Framework**.

**Products**: **DbSet**, който представлява   
таблицата Product в базата данни.

**ProductTypes**: **DbSet**, който представлява  
таблицата ProductType в базата данни.

**Връзки и Зависимости**

* Базата данни се състои от две основни таблици: Product и ProductType.
* Връзката между тези таблици е от тип one-to-many (един към много), където един тип продукт (ProductType) може да има много продукти (Product).
* ProductType: Таблица, съдържаща типове продукти.
* Product: Таблица, съдържаща информация за продуктите, включително връзка към ProductType чрез външен ключ (ProductTypeId).
* Връзкаone-to-many между ProductType и Product: Един тип продукт може да има много продукти, но всеки продукт е свързан само с един тип продукт.
* Външен ключ ProductTypeId в таблицата Product- външен ключ, който свързва продуктите с техния тип.

**ТОЧКА 2.**

**Описание на програмата, как работи и какво прави и как да се използва.**

---------------------------------------------------------------------------------------------------------------------

**“Brand:” - Label с текстова кутия за въвеждане на име на марка.**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

---------------------------------------------------------------------------------------------------------------------

**“Price:” - Label с текстова кутия за въвеждане на цената за продукта (само с числа).**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated  
  
---------------------------------------------------------------------------------------------------------------------

**“Expiration Date:” - Label с текстова кутия за въвеждане на срока на годност за продукта (dd/mm/year).**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

---------------------------------------------------------------------------------------------------------------------

**“Description:” - Label с текстова кутия за въвеждане описание за продукта.**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

---------------------------------------------------------------------------------------------------------------------

**“Product Type:” - Label с ComboBox за избиране на типа на продукта чрез падащо меню от базата данни.**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

**(1) Бутон  “ADD” - Добавя и записва в базата данни цялата информация от въведените полета.**

**(2) Бутон  “UPDATE” - Променя добавен запис в базата данни, без да се налага да се трие и добавя нов.**

**(3) Бутон  “DELETE” - Изтрива данни от базата чрез въвеждане на избраното от нас ID.**

**(4) Бутон  “CLEAR ALL” - Изтрива всички данни добавени в полетата за въвеждане на информация.**

**(5) Бутон  “FIND” - Намира данни от базата чрез въвеждане на ид и ги поставя в полетата за въвеждане.**

A blurry picture of a building

Description automatically generated**Кутията “ListBox” - Служи за събиране и визуализиране на цялата информация, въведена от потребителя.**

A screenshot of a computer

Description automatically generated

**ТОЧКА 3.**

**Анализ на задачата**

**Контролери методи и CRUD операции**

**Обект от класа DbContext**

private MagazineDbContext \_magazineDbContext = new MagazineDbContext();

* **Добавя нов продукт в базата данни**.

public void Create(Product product)

{

\_magazineDbContext.Products.Add(product);

\_magazineDbContext.SaveChanges();

}

* **Чете и връща продукт по ID или всички продукти от базата данни.**

public Product Get(int id)

{

Product findedProduct = \_magazineDbContext.Products.Find(id);

if (findedProduct != null)

{

\_magazineDbContext.Entry(findedProduct).Reference(x => x.ProductTypes).Load();

}

return findedProduct;

}

public List<Product> GetAll()

{

return \_magazineDbContext.Products.Include("ProductTypes").ToList();

}

* **Актуализира съществуващ продукт с нови данни по дадено ID.**

public void Update(int id, Product product)

{

Product findedProduct = \_magazineDbContext.Products.Find(id);

if (findedProduct == null)

{

return;

}

findedProduct.Brand = product.Brand;

findedProduct.Description = product.Description;

findedProduct.Price = product.Price;

findedProduct.ExpirationDate = product.ExpirationDate;

findedProduct.ProductTypeId = product.ProductTypeId;

\_magazineDbContext.SaveChanges();

}

* **Изтрива продукт по дадено ID от базата данни**

public void Delete(int id)

{

Product findedProduct = \_magazineDbContext.Products.Find(id);

\_magazineDbContext.Products.Remove(findedProduct);

\_magazineDbContext.SaveChanges();

}

* **Кода на формата – всички методи, бутони и код относно създаването и въревенето на задачата**

ProductController productController = new ProductController();   
ProductTypeController productTypeController = new ProductTypeController();

* **LoadRecord(Product product): Зарежда данните на продукт в съответните контроли на формата.**

private void LoadRecord(Product product)

{

txtID.BackColor = Color.White;

txtID.Text = product.Id.ToString();

txtBrand.Text = product.Brand;

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

txtDescription.Text = product.Description;

txtExpirationDate.Text = product.ExpirationDate.ToString();

cmbProductType.Text = product.ProductTypes.ProductTypeName;

}

* **Form1\_Load(object sender, EventArgs e): Метод, който се изпълнява при зареждане на формата.**

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

{

List<ProductType> allProductTypes = productTypeController.GetAllProductTypes();

cmbProductType.DataSource = allProductTypes;

cmbProductType.DisplayMember = "ProductTypeName";

cmbProductType.ValueMember = "Id";

SelectAll();

lblID.BackColor = System.Drawing.Color.Transparent;

lblBrand.BackColor = System.Drawing.Color.Transparent;

lblDescription.BackColor = System.Drawing.Color.Transparent;

lblPrice.BackColor = System.Drawing.Color.Transparent;

lblExpirationDate.BackColor = System.Drawing.Color.Transparent;

lblProductType.BackColor = System.Drawing.Color.Transparent;

}

* **ClearScreen(): Изчиства всички текстови полета и нулира избора в cmbProductType**.

private void ClearScreen()

{

txtBrand.Clear();

txtDescription.Clear();

txtExpirationDate.Clear();

txtPrice.Clear();

txtID.Clear();

cmbProductType.Text = "";

}

* **btnADD\_Click(object sender, EventArgs e): Метод, който се изпълнява при натискане на бутона за добавяне (btnADD).**

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

{

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

{

MessageBox.Show("Въведете данни!");

txtDescription.Focus();

return;

}

Product newProduct = new Product();

newProduct.Brand = txtBrand.Text;

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

newProduct.ExpirationDate = DateTime.Parse(txtExpirationDate.Text);

newProduct.Description = txtDescription.Text;

newProduct.ProductTypeId = (int)cmbProductType.SelectedValue;

productController.Create(newProduct);

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

SelectAll();

}

**btnUPDATE\_Click(object sender, EventArgs e): Метод, който се изпълнява при натискане на бутона за актуализация (btnUPDATE).**

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.BackColor = Color.Red;

txtID.Focus();

return; **}**

else

findId = int.Parse(txtID.Text);

if (string.IsNullOrEmpty(txtBrand.Text))

{

Product findedProduct = productController.Get(findId);

if (findedProduct == null)

{

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

txtID.BackColor = Color.Red;

txtID.Focus();

return;

}

LoadRecord(findedProduct);

}

else

{

Product updatedProduct = new Product

{ updatedProduct.Brand = txtBrand.Text,

updatedProduct.Description = txtDescription.Text,

updatedProduct.ExpirationDate = DateTime.Parse(txtExpirationDate.Text),

updatedProduct.Price = decimal.Parse(txtPrice.Text),

updatedProduct.ProductTypeId = (int)cmbProductType.SelectedValue };

productController.Update(findId, updatedProduct);

}

SelectAll();

}

**btnFIND\_Click(object sender, EventArgs e): Метод, който се изпълнява при натискане на бутона**

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.BackColor = Color.Cyan;

txtID.Focus();

return;

}

else

{

findId = int.Parse(txtID.Text);

}

Product findedProduct = productController.Get(findId);

if (findedProduct == null)

{

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

}

else

{

LoadRecord(findedProduct);

}

}