

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:**

**Заключителен проект**

**Задача № 7**

*Ученик: Христо Стефанов Фънев 11д № 24*

***Професия:*** *„Приложен програмист“*

***Специалност:*** *„Приложно програмиране“*

Сопот, 2022 г.

**ТОЧКА 1.**

**Описание на БД**

Parcels

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Id** | **Name** | **Description** | **Price** | **Weight** | **ParcelTypeId** |
| 1 | Мишка | Компютърен аксесоар | 30 | 1,2 | 1 |
| 2 | Диван | Мебел | 250 | 50 | 5 |
| 3 | Натурален акт | Документ за притежание на имот | 25 | 0,5 | 2 |

ParcelTypes

|  |  |
| --- | --- |
| **Id** | **Name** |
| 1 | Малък пакет |
| 2 | Документи |
| 3 | Едрогабаритен |
| 4 | Писмо |
| 5 | Непалитизиран |

|  |  |
| --- | --- |
| Parcels |  |
| Id | int |
| Name | string |
| Description | string |
| Price | decimal |
| Weight | decimal |
| ParcelTypeId  ParcelType | Int  ParcelType |

|  |  |
| --- | --- |
| ParcelTypes |  |
| Id | int |
| Name | string |

Таблица (class) Parcel:

В класа се съхранява главната информация за инстанцията която добавяме към таблицата

class Parcel

{

[Key]

public int Id { get; set; }

public string Name { get; set; }

public string Description { get; set; }

public decimal Price { get; set; }

public decimal Weight { get; set; }

public int ParcelTypeId { get; set; }

public ParcelType ParcelType { get; set; }

}

Таблица (class) ParcelType:

В тази таблица се съдържат типовете на инстанцията, които се исползват в първата таблица.

class ParcelType

{

[Key]

public int Id { get; set; }

public string Name { get; set; }

public ICollection<Parcel> Parcels { get; set; }

}

Клас ParcelContext:

Класът служи за свързване на двете таблици и по нататъчната им употреба.

class ParcelContext : DbContext

{

public ParcelContext()

: base("name=ParcelContext")

{

}

public DbSet<Parcel> Parcels { get; set; }

public DbSet<ParcelType> ParcelTypes { get; set; }

}

App.Config файл:

Файла служи за създаването на базата данни чрез класа ParcelContext.

<connectionStrings>

<add name="ParcelContext"

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

Initial Catalog=Courier; Integrated Security=true"

providerName="System.Data.SqlClient" />

</connectionStrings>

**ГЛАВА 2.**

Класът ParcelBusiness:

Класът се използва, за да може да се извършат всички CRUD операции върху таблицата.

internal class ParcelBusiness

{

private ParcelContext parcelContext = new ParcelContext();

public List<Parcel> GetAll()

public Parcel Get(int id)

public void Add(Parcel parcel)

public void Update(int id, Parcel parcel)

public void Delete(int id)

}

Метод GetAll():

Метода се използва за взимане на данните на елементите в таблицата.

public List<Parcel> GetAll()

{

return parcelContext.Parcels.Include("ParcelType").ToList();

}

Метод Get():

Метода се използва за взимане на данните на един елемент от таблицата.

public Parcel Get(int id)

{

Parcel fndParcel = parcelContext.Parcels.Find(id);

if (fndParcel != null)

{

parcelContext.Entry(fndParcel).Reference(x => x.ParcelType).Load();

}

return fndParcel;

}

Метод Add():

Метода се използва за вкарване на данни за нов елемент в таблицата.

public void Add(Parcel parcel)

{

parcelContext.Parcels.Add(parcel);

parcelContext.SaveChanges();

}

Метод Update():

Метода се използва за промяна на данните за даден елемент в таблицата.

public void Update(int id, Parcel parcel)

{

Parcel updParcel = parcelContext.Parcels.Find(id);

if (updParcel == null)

{

return;

}

updParcel.Name = parcel.Name;

updParcel.Description = parcel.Description;

updParcel.Price = parcel.Price;

updParcel.Weight = parcel.Weight;

updParcel.ParcelTypeId = parcel.ParcelTypeId;

parcelContext.SaveChanges();

}

Метод Delete():

Метода се използва за изтриване на данните на даден елемент от таблицата.

public void Delete(int id)

{

Parcel delParcel = parcelContext.Parcels.Find(id);

parcelContext.Parcels.Remove(delParcel);

parcelContext.SaveChanges();

}

Класът ParcelTypeBusiness:

Класът се използва за употребата на comboBox и извличане на данните от таблицата с видове.

internal class ParcelTypeBusiness

{

private ParcelContext parcelTypeContext = new ParcelContext();

public List<ParcelType> GetAllTypes()

public string GetType(int id)

}

Метод GetAllTypes():

Метода се използва за взимане за предварително вкараните типове.

public List<ParcelType> GetAllTypes()

{

return parcelTypeContext.ParcelTypes.ToList();

}

Метод GetType():

Метода се използва за взимане на номера на типа.

public string GetType(int id)

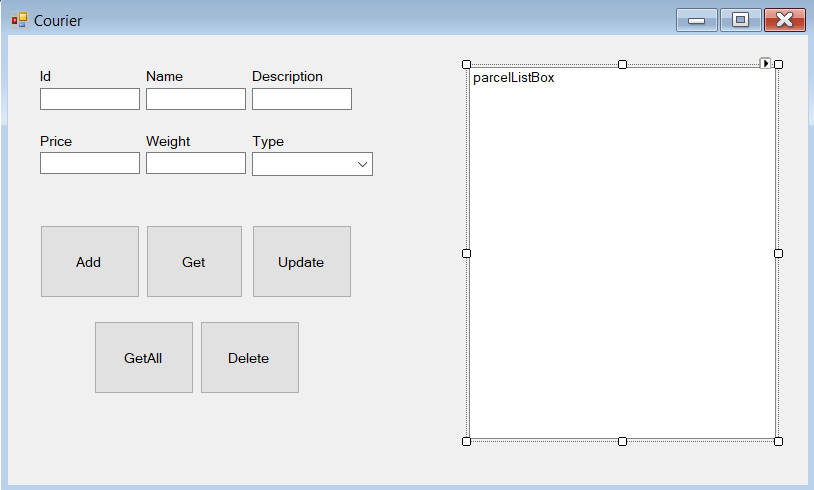
{

return parcelTypeContext.ParcelTypes.Find(id).Name;

}

**ГЛАВА 3.**

**Анализ на задачата**



Създаване на инстанции от Business класовете, за употреба на методите за CRUD операции в тях:

ParcelBusiness parcelBusiness = new ParcelBusiness();

ParcelTypeBusiness parcelTypeBusiness = new ParcelTypeBusiness();

Метод LoadRecord():

Метод за свързване на формата с бизнеса (контролера) на задачата:

private void LoadRecord(Parcel parcel)

{

idTextBox.BackColor = Color.White;

idTextBox.Text = parcel.Id.ToString();

nameTextBox.Text = parcel.Name;

descriptionTextBox.Text = parcel.Description;

priceTextBox.Text = parcel.Price.ToString();

weightTextBox.Text = parcel.Weight.ToString();

parcelTypeComboBox.Text = parcel.ParcelType.Name;

}

Метод ClearScreen():

Метод за изчистване на формата след извършване на някоя от операциите:

private void ClearScreen()

{

idTextBox.BackColor = Color.White;

idTextBox.Clear();

nameTextBox.Clear();

descriptionTextBox.Clear();

priceTextBox.Clear();

weightTextBox.Clear();

parcelTypeComboBox.Text = "";

}

Метод Form1\_Load() (Главен метод):

Начин на зареждане на формата при стартиране.

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

{

List<ParcelType> allTypes = parcelTypeBusiness.GetAllTypes();

parcelTypeComboBox.DataSource = allTypes;

parcelTypeComboBox.DisplayMember = "Name";

parcelTypeComboBox.ValueMember = "Id";

}

Метод за събитето addButton\_Click():

Метод на бутон за добавяне на елемент в базата данни. Метода проверява дали сме въвели данни в текстовото поле и ако не сме ни връща грешка, но ако сме вкарали ни добавя елемента чрез инстанцията, която създадохме.

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

{

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

{

MessageBox.Show("Въведете данни моля!");

return;

}

Parcel parcel = new Parcel();

parcel.Name = nameTextBox.Text;

parcel.Description = descriptionTextBox.Text;

parcel.Price = decimal.Parse(priceTextBox.Text);

parcel.Weight = decimal.Parse(weightTextBox.Text);

parcel.ParcelTypeId = (int)parcelTypeComboBox.SelectedValue;

parcelBusiness.Add(parcel);

ClearScreen();

}

Метод за събитето getButton\_Click():

Метод за бутон за намиране на даден елемент в базата данни по дадено ид (Номер). Метода проверява дали сме вкарали число като номер в текстовоте поле, ако не сме ни връща грешка, след това проверява дали има елемент с такъв номер и ако не съществува също ни връща грешка. При никакви върнати грешки метода връща данните за дадения елемент.

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

{

int id = 0;

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

{

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

idTextBox.BackColor = Color.Red;

return;

}

else

{

id = int.Parse(idTextBox.Text);

}

Parcel fndParcel = parcelBusiness.Get(id);

if (fndParcel == null)

{

MessageBox.Show("Няма такъв запис!");

idTextBox.BackColor = Color.Red;

return;

}

LoadRecord(fndParcel);

}

Метод за събитето updateButton\_Click():

Метод за бутон за променяне на стойностите на вече създаден елемент в базата данни. Метода проверява дали сме вкарали число като номер в текстовоте поле, ако не сме ни връща грешка, след това проверява дали има елемент с такъв номер и ако не съществува също ни връща грешка. При никакви върнати грешки се променят данните на избрания от нас елемент.

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

{

int id = 0;

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

{

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

idTextBox.BackColor = Color.Red;

return;

}

else

{

id = int.Parse(idTextBox.Text);

}

if (string.IsNullOrEmpty(nameTextBox.Text))

{

Parcel updParcel = parcelBusiness.Get(id);

if (updParcel == null)

{

MessageBox.Show("Няма такъв запис!");

idTextBox.BackColor = Color.Red;

return;

}

LoadRecord(updParcel);

}

else

{

Parcel parcel = new Parcel();

parcel.Name = nameTextBox.Text;

parcel.Description = descriptionTextBox.Text;

parcel.Price = decimal.Parse(priceTextBox.Text);

parcel.Weight = decimal.Parse(weightTextBox.Text);

parcel.ParcelTypeId = (int)parcelTypeComboBox.SelectedValue;

parcelBusiness.Update(id, parcel);

}

}

Метод за събитето getAllButton\_Click():

Метод за бутон за изкарване на всички вкарани елементи в базата данни. Метода използва listBox и чрез него изкарва данните за всички вкарани елементи в базата данни.

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

{

List<Parcel> parcels = parcelBusiness.GetAll();

parcelListBox.Items.Clear();

foreach (var item in parcels)

{

parcelListBox.Items.Add($"{item.Id} {item.Name} {item.Description} {item.Price}лв. {item.Weight}кг. Вид пратка: {item.ParcelType.Name}");

}

}

Метод за събитето deleteButton\_Click():

Метод за бутон за изтриване на даден елемент от базата данни по дадено ид (Номер). Метода проверява дали сме вкарали число като номер в текстовоте поле, ако не сме ни връща грешка, след това проверява дали има елемент с такъв номер и ако не съществува също ни връща грешка. След никакви грешки ни изкарва запитване дали със сигурност искаме да изтрием този запис в таблицата и ако отговорим с да, се изтрива записа по даден от нас номер.

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

{

int id = 0;

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

{

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

idTextBox.BackColor = Color.Red;

return;

}

else

{

id = int.Parse(idTextBox.Text);

}

Parcel delParcel = parcelBusiness.Get(id);

if (delParcel == null)

{

MessageBox.Show("Няма такъв запис!");

idTextBox.BackColor = Color.Red;

return;

}

LoadRecord(delParcel);

DialogResult question = MessageBox.Show("Искате ли да изтриете записа?", "Изтриване", MessageBoxButtons.YesNo);

if (question == DialogResult.Yes)

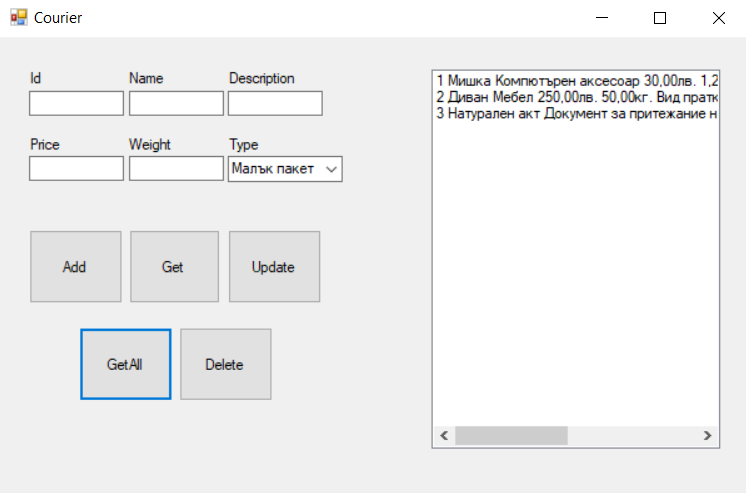
{

parcelBusiness.Delete(id);

}

}

Външен вид на приложениете след стартирането му:



В изпълнението на Задача №7 съм използвал CRUD (Create, Read, Update, Delete) операции, както и технологията ORM (Code First). Също се използва и пакета Entity Framework и миграции за работа с базата данни и поддръжката им. Както и за крайния външен вид съм използвал и Windows Forms App (.NET Framework), за по лесна употреба и по представителен вид.