**ПМГ „Академик Боян Петканчин“ – Хасково**

**НП „ИТ-КАРИЕРА“**

**Документация**

**Модул 7: Разработка на софтуер**

**Тема: Хранителен магазин (Shop)**

**Изготвили:**

**Мухамед Ахмед**

**Пламена Николова**

**Хасково 2020г.**

**СЪДЪРЖАНИЕ:**

**I. Въведение**

I.1. [Какво представлява програмата?.........................................................2](#Каквопредставлявапрограмата)

I.2. [Какви функционалности поддържа.....................................................2](#функционалности)

**II. Data**

II.1. [Клас-диаграма за всички класове (Models) в папка Data…………..2](#Models)

II.2. [Клас за животински продукти (AnimalProduct.cs)………………….2](#AnimalProduct)

II.3. [Клас за напитки (Drink.cs)……………………………………………3](#Drink)

II.4. [Клас за плодове и зеленчуци (FruitAndVegetable.cs)……………….3](#FruitAndVegetable)

II.5. [Клас за ядки (Nut.cs)……………………………………………….....4](#Nut)

II.6. [Клас за печени изделия (Patry.cs)…………………………………….4](#Patry)

II.7. [Контекст клас (ShopContext.cs)………………………...……………5](#ShopContext)

**III. Controllers**

III.1. [Клас-диаграма за всички контролери………………………………5](#Controllers)

III.2. [AnimalProductController.cs…………………………………………..6](#AnimalProductController)

III.3. [DrinkController.cs…………………………………………………….7](#DrinkController)

III.4. [FruitAndVegetableController.cs...……………………………………8](#FruitAndVegetableController)

III.5. [PastryController.cs………………………...………………………...11](#PatryController)

**IV. View**

IV.1. [Клас-диаграма за класа Display.cs………………………………...13](#View)

IV.2[. Display.cs……………………………………………………………13](#Display)

**V. Program.cs**

V.1. [Program.cs..…………………………………………………………..27](#Program)

**Въведение**

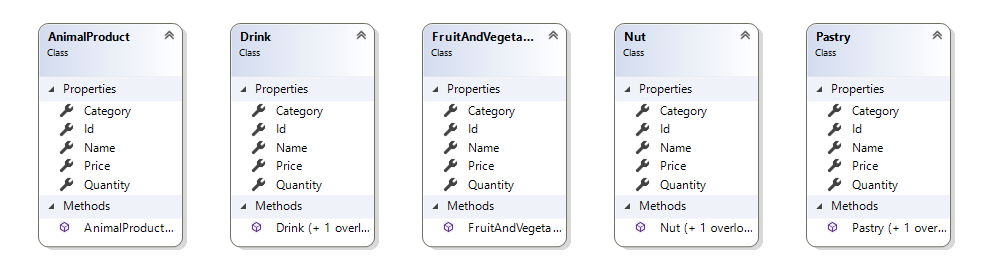
* Картина, която съдържа екранна снимка, компютър, седящ, монитор

  Описанието е генерирано автоматично**Какво представлява програмата?**

Програмата е предназначена за хранителни магазини или за складове, в които се съхраняват хранителни продукти. Тя поддържа пет типа продукти: животински продукти, напитки, ядки, печени изделия и плодове и зеленчуци. За всеки един продукт в базата данни се съхранява информация за ИД номера, категорията, името, цената и количеството.

* **Какви функционалности поддържа?**
* Извеждане на списък с всички налични продукти
* Търсене на продукт по ИД номер
* Добавяне на продукт
* Изтриване на продукт
* Актуализиране на данните за даден продукт

**Код**

* Data
  + Models
    - AnimalProduct.cs

/// <summary>

/// The structure of animal product table in database.

/// </summary>

public class AnimalProduct

   {

       public int Id { get; set; }

       public string Category { get; set; }

       public string Name { get; set; }

       public decimal Price { get; set; }

       public int Quantity { get; set; }

       public AnimalProduct(string category, string name, decimal price, int quantity)

       {

           Category = category;

           Name = name;

           Price = price;

           Quantity = quantity;

       }

       public AnimalProduct()

       {

       }

   }

* + - Drink.cs

/// <summary>

/// The structure of drink table in database.

/// </summary>

public class Drink

   {

       public int Id { get; set; }

       public string Category { get; set; }

       public string Name { get; set; }

       public decimal Price { get; set; }

       public int Quantity { get; set; }

       public Drink(string category, string name, decimal price, int quantity)

       {

           Category = category;

           Name = name;

           Price = price;

           Quantity = quantity;

       }

       public Drink()

       {

       }

   }

* + - FruitAndVegetable.cs

/// <summary>

/// The structure of fruits and vegetable table in database.

/// </summary>

public class FruitAndVegetable

   {

       public int Id { get; set; }

       public string Category { get; set; }

       public string Name { get; set; }

       public decimal Price { get; set; }

       public int Quantity { get; set; }

       public FruitAndVegetable(string category, string name, decimal price, int quantity)

       {

           Category = category;

           Name = name;

           Price = price;

           Quantity = quantity;

       }

       public FruitAndVegetable()

       {

       }

   }

* + - Nut.cs

/// <summary>

/// The structure of the Nut table in the database.

/// </summary>

public class Nut

    {

        public int Id { get; set; }

        public string Category { get; set; }

        public string Name { get; set; }

        public decimal Price { get; set; }

        public int Quantity { get; set; }

        public Nut(string category, string name, decimal price, int quantity)

        {

            Category = category;

            Name = name;

            Price = price;

            Quantity = quantity;

        }

        public Nut()

        {

        }

    }

* + - Patry.cs

/// <summary>

/// The structure of the Pastry table in the database.

/// </summary>

public class Pastry

    {

        public int Id { get; set; }

        public string Category { get; set; }

        public string Name { get; set; }

        public decimal Price { get; set; }

        public int Quantity { get; set; }

        public Pastry(string category, string name, decimal price, int quantity)

        {

            Category = category;

            Name = name;

            Price = price;

            Quantity = quantity;

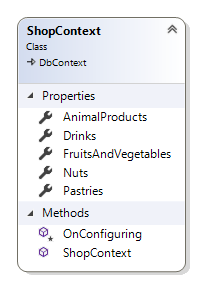
        }

        public Pastry()

        {

        }

    }

* + Context
    - ShopContext.cs

/// <summary>

/// Context to connect to the database.

/// </summary>

public class ShopContext : DbContext

   {

       public ShopContext() : base()

       {

       }

       public virtual DbSet<FruitAndVegetable> FruitsAndVegetables { get; set; }

       public virtual DbSet<Nut> Nuts { get; set; }

       public virtual DbSet<Pastry> Pastries { get; set; }

       public virtual DbSet<Drink> Drinks { get; set; }

       public virtual DbSet<AnimalProduct> AnimalProducts { get; set; }

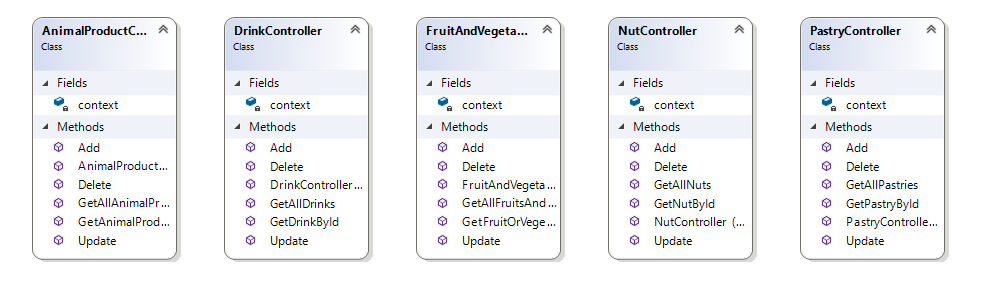
       protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

       {

           optionsBuilder.UseSqlServer("Server = .\\SQLEXPRESS; Database = Shop; Integrated Security = true ");

       }

   }

* Controllers
  + - AnimalProductController.cs

/// <summary>

    /// Providers the link between the database and UI.

    /// </summary>

    public class AnimalProductController

    {

        private ShopContext context;

        /// <summary>

        ///

        /// </summary>

        /// <param name="shopContext"></param>

        public AnimalProductController(ShopContext shopContext)

        {

            context = shopContext;

        }

        /// <summary>

        ///

        /// </summary>

        public AnimalProductController()

        {

            context = new ShopContext();

        }

        /// <summary>

        /// Gives all animal products in the database.

        /// </summary>

        /// <returns>all animal products from the database</returns>

        public List<AnimalProduct> GetAllAnimalProducts()

        {

            return context.AnimalProducts.ToList();

        }

        /// <summary>

        /// Gives an animal product with wanted id.

        /// </summary>

        /// <param name="id">id of the wanted animal product</param>

        /// <returns>a animal product with wanted id</returns>

        public AnimalProduct GetAnimalProductById(int id)

        {

            return context.AnimalProducts.FirstOrDefault(m => m.Id == id);

        }

        /// <summary>

        /// Adds an animal product in database.

        /// </summary>

        /// <param name="animalProduct">the animal product that will be added</param>

        public void Add(AnimalProduct animalProduct)

        {

                context.AnimalProducts.Add(animalProduct);

                context.SaveChanges();

        }

        /// <summary>

        /// Updates an animal product.

        /// </summary>

        /// <param name="animalProduct">the animal product that will be updated</param>

        public void Update(AnimalProduct animalProduct)

        {

                var item = context.AnimalProducts.Find(animalProduct.Id);

                if (item != null)

                {

                    context.Entry(item).CurrentValues.SetValues(animalProduct);

                    context.SaveChanges();

                }

        }

        /// <summary>

        /// Deletes a animal product with wanted id.

        /// </summary>

        /// <param name="id">id of the wanted animal product</param>

        public void Delete(int id)

        {

                var item = context.AnimalProducts.FirstOrDefault(m => m.Id == id);

                if (item != null)

                {

                    context.AnimalProducts.Remove(item);

                    context.SaveChanges();

                }

        }

    }

* + - DrinkController.cs

/// <summary>

  /// Providers the link between the database and UI.

  /// </summary>

  public class DrinkController

  {

      private ShopContext context;

      /// <summary>

      ///

      /// </summary>

      /// <param name="shopContext"></param>

      public DrinkController(ShopContext shopContext)

      {

          context = shopContext;

      }

      /// <summary>

      ///

      /// </summary>

      public DrinkController()

      {

          context = new ShopContext();

      }

      /// <summary>

      /// Gives all Drinks in the database.

      /// </summary>

      /// <returns>all drinks from the database</returns>

      public List<Drink> GetAllDrinks()

      {

         return context.Drinks.ToList();

      }

      /// <summary>

      /// Gives s drink with wanted id.

      /// </summary>

      /// <param name="id">id of the wanted drink</param>

      /// <returns>a drink with wanted id</returns>

      public Drink GetDrinkById(int id)

      {

          return context.Drinks.FirstOrDefault(m => m.Id == id);

      }

      /// <summary>

      /// Adds a drink in database.

      /// </summary>

      /// <param name="drink">the drink that will be added</param>

      public void Add(Drink drink)

      {

              context.Drinks.Add(drink);

              context.SaveChanges();

      }

      /// <summary>

      /// Updates a drink.

      /// </summary>

      /// <param name="drink">the drink that will be updated</param>

      public void Update(Drink drink)

      {

              var item = context.Drinks.Find(drink.Id);

              if (item != null)

              {

                  context.Entry(item).CurrentValues.SetValues(drink);

                  context.SaveChanges();

              }

      }

      /// <summary>

      /// Deletes a dring with wanted id.

      /// </summary>

      /// <param name="id">id of the wanted drink</param>

      public void Delete(int id)

      {

         var item = context.Drinks.FirstOrDefault(m => m.Id == id);

         if (item != null)

         {

             context.Drinks.Remove(item);

              context.SaveChanges();

         }

      }

  }

* + - FruitAndVegetableController.cs

/// <summary>

   /// Providers the link between the database and UI.

   /// </summary>

   public class FruitAndVegetableController

   {

       /// <summary>

       /// Database link.

       /// </summary>

       private ShopContext context;

       /// <summary>

       ///

       /// </summary>

       /// <param name="shopcontext"></param>

       public FruitAndVegetableController(ShopContext shopcontext)

       {

           context = shopcontext;

       }

       /// <summary>

       ///

       /// </summary>

       public FruitAndVegetableController()

       {

           context = new ShopContext();

       }

       /// <summary>

       /// Gives all fruits and vegetables in the database.

       /// </summary>

       /// <returns>all fruits and vegetables from the database</returns>

       public List<FruitAndVegetable> GetAllFruitsAndVegetables()

       {

               return context.FruitsAndVegetables.ToList();

       }

       /// <summary>

       /// Gives a fruit or vegetable with the wanted Id.

       /// </summary>

       /// <param name="id">Id of the wanted fruit or vegetable</param>

       /// <returns>a fruit or vegetable with that id</returns>

       public FruitAndVegetable GetFruitOrVegetableById(int id)

       {

               return context.FruitsAndVegetables.FirstOrDefault(m => m.Id == id);

       }

       /// <summary>

       /// Adds a fruit or vegetable.

       /// </summary>

       /// <param name="fruitORvegetable">the fruit or vegetable that will be added.</param>

       public void Add(FruitAndVegetable fruitORvegetable)

       {

               context.FruitsAndVegetables.Add(fruitORvegetable);

               context.SaveChanges();

       }

       /// <summary>

       /// Updates a fruit or vegetable.

       /// </summary>

       /// <param name="fruitORvegetable">the fruit or vegetable that will be updated.</param>

       public void Update(FruitAndVegetable fruitORvegetable)

       {

               var item = context.FruitsAndVegetables.Find(fruitORvegetable.Id);

               if (item != null)

               {

                   context.Entry(item).CurrentValues.SetValues(fruitORvegetable);

                   context.SaveChanges();

               }

       }

       /// <summary>

       /// Deletes a fruit or vegetable with the wanted Id.

       /// </summary>

       /// <param name="id">Id of the wanted fruit or vegetable.</param>

       public void Delete(int id)

       {

               var item = context.FruitsAndVegetables.FirstOrDefault(m => m.Id == id);

               if (item != null)

               {

                   context.FruitsAndVegetables.Remove(item);

                   context.SaveChanges();

               }

       }

   }

* + - NutController.cs

/// <summary>

    /// Provides the link between the database and the UI.

    /// </summary>

    public class NutController

    {

        /// <summary>

        /// Database link.

        /// </summary>

        private ShopContext context;

        /// <summary>

        ///

        /// </summary>

        /// <param name="shopContext"></param>

        public NutController(ShopContext shopContext)

        {

            context = shopContext;

        }

        /// <summary>

        ///

        /// </summary>

        public NutController()

        {

            context = new ShopContext();

        }

        /// <summary>

        /// Gives all Nuts in the database.

        /// </summary>

        /// <returns>all nuts from the database</returns>

        public List<Nut> GetAllNuts()

        {

                return context.Nuts.ToList();

        }

        /// <summary>

        /// Gives a Nut with the wanted Id.

        /// </summary>

        /// <param name="id">Id of the wanted nut</param>

        /// <returns>a nut with that id</returns>

        public Nut GetNutById(int id)

        {

                return context.Nuts.FirstOrDefault(m => m.Id == id);

        }

        /// <summary>

        /// Adds a Nut.

        /// </summary>

        /// <param name="nut">the nut that will be added</param>

        public void Add(Nut nut)

        {

                context.Nuts.Add(nut);

                context.SaveChanges();

        }

        /// <summary>

        /// Updates a Nut.

        /// </summary>

        /// <param name="nut">the nut that will be updated</param>

        public void Update(Nut nut)

        {

                var item = context.Drinks.Find(nut.Id);

                if (item != null)

                {

                    context.Entry(item).CurrentValues.SetValues(nut);

                    context.SaveChanges();

                }

        }

        /// <summary>

        /// Deletes a Nut with the wanted Id.

        /// </summary>

        /// <param name="id">Id of the wanted nut</param>

        public void Delete(int id)

        {

                var item = context.Nuts.FirstOrDefault(m => m.Id == id);

                if (item != null)

                {

                    context.Nuts.Remove(item);

                    context.SaveChanges();

                }

        }

    }

* + - PatryController.cs

/// <summary>

   /// Provides the link between the database and the UI.

   /// </summary>

   public class PastryController

   {

       /// <summary>

       /// Database link.

       /// </summary>

       private ShopContext context;

       /// <summary>

       ///

       /// </summary>

       /// <param name="shopContext"></param>

       public PastryController(ShopContext shopContext)

       {

           context = shopContext;

       }

       /// <summary>

       ///

       /// </summary>

       public PastryController()

       {

           context = new ShopContext();

       }

       /// <summary>

       /// Gives all Pastry in the database.

       /// </summary>

       /// <returns>all pastries from the database</returns>

       public List<Pastry> GetAllPastries()

       {

               return context.Pastries.ToList();

       }

       /// <summary>

       /// Gives a Pastry with the wanted Id.

       /// </summary>

       /// <param name="id">Id of the wanted pastry</param>

       /// <returns>a pastry with that id</returns>

       public Pastry GetPastryById(int id)

       {

               return context.Pastries.FirstOrDefault(m => m.Id == id);

       }

       /// <summary>

       /// Adds a Pastry.

       /// </summary>

       /// <param name="pastry">the pastry that will be added</param>

       public void Add(Pastry pastry)

       {

               context.Pastries.Add(pastry);

               context.SaveChanges();

       }

       /// <summary>

       /// Updates a Pastry.

       /// </summary>

       /// <param name="pastry">the pastry that will be updated</param>

       public void Update(Pastry pastry)

       {

               var item = context.Pastries.Find(pastry.Id);

               if (item != null)

               {

                   context.Entry(item).CurrentValues.SetValues(pastry);

                   context.SaveChanges();

               }

       }

       /// <summary>

       /// Deletes a Pastry with the wanted Id.

       /// </summary>

       /// <param name="id">Id of the wanted pastry</param>

       public void Delete(int id)

       {

               var item = context.Pastries.FirstOrDefault(m => m.Id == id);

               if (item != null)

               {

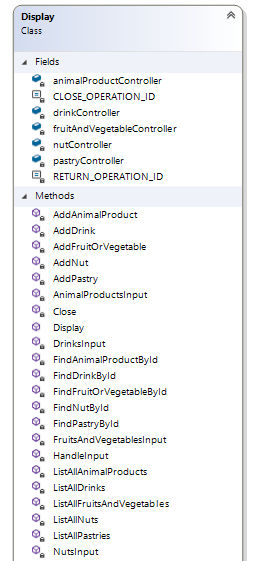
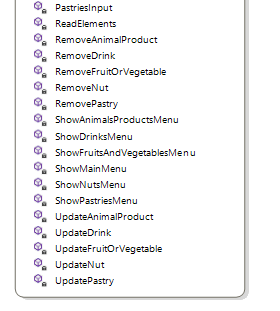
                   context.Pastries.Remove(item);

                   context.SaveChanges();

               }

       }

   }

* View
  + - Display.cs

/// <summary>

    /// User interface.

    /// </summary>

    public class Display

    {

        private const int CLOSE\_OPERATION\_ID = 6;

        private const int RETURN\_OPERATION\_ID = 6;

        private DrinkController drinkController;

        private NutController nutController;

        private FruitAndVegetableController fruitAndVegetableController;

        private PastryController pastryController;

        private AnimalProductController animalProductController;

        /// <summary>

        /// Initialize controllers.

        /// </summary>

        public Display()

        {

            drinkController = new DrinkController(new ShopContext());

            nutController = new NutController(new ShopContext());

            fruitAndVegetableController = new FruitAndVegetableController(new ShopContext());

            pastryController = new PastryController(new ShopContext());

            animalProductController = new AnimalProductController(new ShopContext());

            HandleInput();

        }

        private void ShowMainMenu()

        {

            Console.WriteLine(new string('\*',40));

            Console.Write(new string(' ',15));

            Console.BackgroundColor = ConsoleColor.DarkRed;

            Console.WriteLine(" MAIN MENU ");

            Console.ResetColor();

            Console.WriteLine(new string('\*', 40));

            Console.BackgroundColor = ConsoleColor.DarkYellow;

            Console.WriteLine("1. Go to pastries");

            Console.BackgroundColor = ConsoleColor.DarkGreen;

            Console.WriteLine("2. Go to fruits and vegetables");

            Console.BackgroundColor = ConsoleColor.DarkMagenta;

            Console.WriteLine("3. Go to nuts");

            Console.BackgroundColor = ConsoleColor.DarkCyan;

            Console.WriteLine("4. Go to drinks");

            Console.BackgroundColor = ConsoleColor.Red;

            Console.WriteLine("5. Go to animal products");// we need to add color

            Console.BackgroundColor = ConsoleColor.DarkBlue;

            Console.WriteLine("6. Exit");

            Console.ResetColor();

            Console.WriteLine(new string('\*', 40));

        }

        private void HandleInput()

        {

            var operation = -1;

            do

            {

                ShowMainMenu();

                Console.Write("Enter number: ");

                operation = int.Parse(Console.ReadLine());

                switch (operation)

                {

                    case 1:

                        PastriesInput();

                        break;

                    case 2:

                        FruitsAndVegetablesInput();

                        break;

                    case 3:

                        NutsInput();

                        break;

                    case 4:

                        DrinksInput();

                        break;

                    case 5:

                        AnimalProductsInput();

                        break;

                    default:

                        break;

                }

            } while (operation != CLOSE\_OPERATION\_ID);

        }

        private void AnimalProductsInput()

        {

            Console.Clear();

            var operation = -1;

            do

            {

                ShowAnimalsProductsMenu();

                Console.Write("Enter number: ");

                operation = int.Parse(Console.ReadLine());

                Console.Clear();

                switch (operation)

                {

                    case 1:

                        ListAllAnimalProducts();

                        Close();

                        break;

                    case 2:

                        FindAnimalProductById();

                        Close();

                        break;

                    case 3:

                        AddAnimalProduct();

                        Close();

                        break;

                    case 4:

                        RemoveAnimalProduct();

                        Close();

                        break;

                    case 5:

                        UpdateAnimalProduct();

                        Close();

                        break;

                    default:

                        break;

                }

            } while (operation != RETURN\_OPERATION\_ID);

        }

        private void UpdateAnimalProduct()

        {

            Console.Write("Enter ID: ");

            int id = int.Parse(Console.ReadLine());

            var animalProduct = animalProductController.GetAnimalProductById(id);

            if (animalProduct != null)

            {

                Console.WriteLine($"{animalProduct.Id} {animalProduct.Category} {animalProduct.Name} {animalProduct.Price}lv/kg {animalProduct.Quantity}kg.");

                var elements = ReadElements();

                animalProduct.Category = elements[0];

                animalProduct.Name = elements[1];

                animalProduct.Price = decimal.Parse(elements[2]);

                animalProduct.Quantity = int.Parse(elements[3]);

                animalProductController.Update(animalProduct);

                Console.WriteLine("The product was updated successfully!");

            }

            else

            {

                Console.WriteLine("The product was not found");

            }

        }

        private void RemoveAnimalProduct()

        {

            Console.Write("Enter ID: ");

            int id = int.Parse(Console.ReadLine());

            var animalProduct = animalProductController.GetAnimalProductById(id);

            if (animalProduct != null)

            {

                animalProductController.Delete(animalProduct.Id);

                Console.WriteLine("Тhe product was deleted successfully!");

            }

            else

            {

                Console.WriteLine("The product was not found!");

            }

        }

        private void AddAnimalProduct()

        {

            var elements = ReadElements();

            AnimalProduct animalProduct = new AnimalProduct(elements[0], elements[1], decimal.Parse(elements[2]), int.Parse(elements[3]));

            animalProductController.Add(animalProduct);

            Console.WriteLine("The product was successfully added!");

        }

        private void FindAnimalProductById()

        {

            Console.Write("Enter ID: ");

            var id = int.Parse(Console.ReadLine());

            var animalProduct = animalProductController.GetAnimalProductById(id);

            if (animalProduct != null)

            {

                Console.WriteLine(new string('\*', 40));

                Console.WriteLine("ID: " + animalProduct.Id);

                Console.WriteLine("Category: " + animalProduct.Category);

                Console.WriteLine("Name: " + animalProduct.Name);

                Console.WriteLine("Price: " + animalProduct.Price + "lv/kg");

                Console.WriteLine("Quantity: " + animalProduct.Quantity + "kg.");

                Console.WriteLine(new string('\*', 40));

            }

            else

            {

                Console.WriteLine("The product was not found!");

            }

        }

        private void ListAllAnimalProducts()

        {

            Console.WriteLine(new string('\*', 40));

            Console.Write(new string(' ', 11));

            Console.BackgroundColor = ConsoleColor.Red;

            Console.WriteLine($" ANIMAL PRODUCTS ");

            Console.ResetColor();

            Console.WriteLine(new string('\*', 40));

            var animalProducts = animalProductController.GetAllAnimalProducts();

            foreach (var item in animalProducts)

            {

                Console.BackgroundColor = ConsoleColor.Red;

                Console.Write($"{item.Id}");

                Console.ResetColor();

                Console.WriteLine($" {item.Category} {item.Name} {item.Price}lv/kg {item.Quantity}kg.");

            }

            Console.WriteLine(new string('\*', 40));

        }

        private void ShowAnimalsProductsMenu()

        {

            Console.WriteLine(new string('\*', 40));

            Console.Write(new string(' ', 10));

            Console.BackgroundColor = ConsoleColor.Red;

            Console.WriteLine(" ANIMAL PRODUCTS MENU ");

            Console.ResetColor();

            Console.WriteLine(new string('\*', 40));

            Console.BackgroundColor = ConsoleColor.DarkBlue;

            Console.WriteLine("1. List all animal products");

            Console.BackgroundColor = ConsoleColor.Red;

            Console.WriteLine("2. Found animal product by ID");

            Console.BackgroundColor = ConsoleColor.DarkBlue;

            Console.WriteLine("3. Add animal product");

            Console.BackgroundColor = ConsoleColor.Red;

            Console.WriteLine("4. Remove animal product");

            Console.BackgroundColor = ConsoleColor.DarkBlue;

            Console.WriteLine("5. Update animal product");

            Console.BackgroundColor = ConsoleColor.Red;

            Console.WriteLine("6. Return to main menu");

            Console.ResetColor();

            Console.WriteLine(new string('\*', 40));

        }

        private void PastriesInput()

        {

            Console.Clear();

            var operation = -1;

            do

            {

                ShowPastriesMenu();

                Console.Write("Enter number: ");

                operation = int.Parse(Console.ReadLine());

                Console.Clear();

                switch (operation)

                {

                    case 1:

                        ListAllPastries();

                        Close();

                        break;

                    case 2:

                        FindPastryById();

                        Close();

                        break;

                    case 3:

                        AddPastry();

                        Close();

                        break;

                    case 4:

                        RemovePastry();

                        Close();

                        break;

                    case 5:

                        UpdatePastry();

                        Close();

                        break;

                    default:

                        break;

                }

            } while (operation != RETURN\_OPERATION\_ID);

        }

        private void ListAllPastries()

        {

            Console.WriteLine(new string('\*', 40));

            Console.Write(new string(' ', 16));

            Console.BackgroundColor = ConsoleColor.DarkYellow;

            Console.WriteLine(" PASTRY ");

            Console.ResetColor();

            Console.WriteLine(new string('\*', 40));

            var pastries = pastryController.GetAllPastries();

            foreach (var item in pastries)

            {

                Console.BackgroundColor = ConsoleColor.DarkYellow;

                Console.Write($"{item.Id}");

                Console.ResetColor();

                Console.WriteLine($" {item.Category} {item.Name} {item.Price}lv/pcs {item.Quantity}pcs.");

            }

            Console.WriteLine(new string('\*', 40));

        }

        private void FindPastryById()

        {

            Console.Write("Enter ID: ");

            var id = int.Parse(Console.ReadLine());

            var pastry = pastryController.GetPastryById(id);

            if (pastry != null)

            {

                Console.WriteLine(new string('\*', 40));

                Console.WriteLine("ID: " + pastry.Id);

                Console.WriteLine("Category: " + pastry.Category);

                Console.WriteLine("Name: " + pastry.Name);

                Console.WriteLine("Price: " + pastry.Price + "lv/pcs");

                Console.WriteLine("Quantity: " + pastry.Quantity + "pcs.");

                Console.WriteLine(new string('\*', 40));

            }

            else

            {

                Console.WriteLine("The product was not found!");

            }

        }

        private void AddPastry()

        {

            var elements = ReadElements();

            Pastry pastry = new Pastry(elements[0],elements[1],decimal.Parse(elements[2]),int.Parse(elements[3]));

            pastryController.Add(pastry);

            Console.WriteLine("The product was successfully added!");

        }

        private void RemovePastry()

        {

            Console.Write("Enter ID: ");

            int id = int.Parse(Console.ReadLine());

            var pastry = pastryController.GetPastryById(id);

            if (pastry != null)

            {

                pastryController.Delete(pastry.Id);

                Console.WriteLine("Тhe product was deleted successfully!");

            }

            else

            {

                Console.WriteLine("The product was not found!");

            }

        }

        private void UpdatePastry()

        {

            Console.Write("Enter ID: ");

            int id = int.Parse(Console.ReadLine());

            var pastry = pastryController.GetPastryById(id);

            if (pastry != null)

            {

                Console.WriteLine($"{pastry.Id} {pastry.Category} {pastry.Name} {pastry.Price}lv/pcs {pastry.Quantity}pcs.");

                var elements = ReadElements();

                pastry.Category = elements[0];

                pastry.Name = elements[1];

                pastry.Price = decimal.Parse(elements[2]);

                pastry.Quantity = int.Parse(elements[3]);

                pastryController.Update(pastry);

                Console.WriteLine("The product was updated successfully!");

            }

            else

            {

                Console.WriteLine("The product was not found");

            }

        }

        private void FruitsAndVegetablesInput()

        {

            Console.Clear();

            var operation = -1;

            do

            {

                ShowFruitsAndVegetablesMenu();

                Console.Write("Enter number: ");

                operation = int.Parse(Console.ReadLine());

                Console.Clear();

                switch (operation)

                {

                    case 1:

                        ListAllFruitsAndVegetables();

                        Close();

                        break;

                    case 2:

                        FindFruitOrVegetableById();

                        Close();

                        break;

                    case 3:

                        AddFruitOrVegetable();

                        Close();

                        break;

                    case 4:

                        RemoveFruitOrVegetable();

                        Close();

                        break;

                    case 5:

                        UpdateFruitOrVegetable();

                        Close();

                        break;

                    default:

                        break;

                }

            } while (operation != RETURN\_OPERATION\_ID);

        }

        private void ListAllFruitsAndVegetables()

        {

            Console.WriteLine(new string('\*', 40));

            Console.Write(new string(' ', 10));

            Console.BackgroundColor = ConsoleColor.DarkGreen;

            Console.WriteLine(" FRUITS AND VEGETABLES ");

            Console.ResetColor();

            Console.WriteLine(new string('\*', 40));

            var fruitsAndVegetables = fruitAndVegetableController.GetAllFruitsAndVegetables();

            foreach (var item in fruitsAndVegetables)

            {

                Console.BackgroundColor = ConsoleColor.DarkGreen;

                Console.Write($"{item.Id}");

                Console.ResetColor();

                Console.WriteLine($" {item.Category} {item.Name} {item.Price}lv/kg {item.Quantity}kg.");

            }

            Console.WriteLine(new string('\*', 40));

        }

        private void FindFruitOrVegetableById()

        {

            Console.Write("Enter ID: ");

            var id = int.Parse(Console.ReadLine());

            var fruitOrVegetable = fruitAndVegetableController.GetFruitOrVegetableById(id);

            if (fruitOrVegetable != null)

            {

                Console.WriteLine(new string('\*', 40));

                Console.WriteLine("ID: " + fruitOrVegetable.Id);

                Console.WriteLine("Category: " + fruitOrVegetable.Category);

                Console.WriteLine("Name: " + fruitOrVegetable.Name);

                Console.WriteLine("Price: " + fruitOrVegetable.Price + "lv/kg");

                Console.WriteLine("Quantity: " + fruitOrVegetable.Quantity + "kg.");

                Console.WriteLine(new string('\*', 40));

            }

            else

            {

                Console.WriteLine("The product was not found!");

            }

        }

        private void AddFruitOrVegetable()

        {

            var elements = ReadElements();

            FruitAndVegetable fruitOrVegetable = new FruitAndVegetable(elements[0], elements[1], decimal.Parse(elements[2]), int.Parse(elements[3]));

            fruitAndVegetableController.Add(fruitOrVegetable);

            Console.WriteLine("The product was successfully added!");

        }

        private void RemoveFruitOrVegetable()

        {

            Console.Write("Enter ID: ");

            int id = int.Parse(Console.ReadLine());

            var fruitOrVegetable = fruitAndVegetableController.GetFruitOrVegetableById(id);

            if (fruitOrVegetable != null)

            {

                fruitAndVegetableController.Delete(fruitOrVegetable.Id);

                Console.WriteLine("Тhe product was deleted successfully!");

            }

            else

            {

                Console.WriteLine("The product was not found!");

            }

        }

        private void UpdateFruitOrVegetable()

        {

            Console.Write("Enter ID: ");

            int id = int.Parse(Console.ReadLine());

            var fruitOrVegetable = fruitAndVegetableController.GetFruitOrVegetableById(id);

            if (fruitOrVegetable != null)

            {

                Console.WriteLine($"{fruitOrVegetable.Id} {fruitOrVegetable.Category} {fruitOrVegetable.Name} {fruitOrVegetable.Price}lv/kg {fruitOrVegetable.Quantity}kg.");

                var elements = ReadElements();

                fruitOrVegetable.Category = elements[0];

                fruitOrVegetable.Name = elements[1];

                fruitOrVegetable.Price = decimal.Parse(elements[2]);

                fruitOrVegetable.Quantity = int.Parse(elements[3]);

                fruitAndVegetableController.Update(fruitOrVegetable);

                Console.WriteLine("The product was updated successfully!");

            }

            else

            {

                Console.WriteLine("The product was not found");

            }

        }

        private void NutsInput()

        {

            Console.Clear();

            var operation = -1;

            do

            {

                ShowNutsMenu();

                Console.Write("Enter number: ");

                operation = int.Parse(Console.ReadLine());

                Console.Clear();

                switch (operation)

                {

                    case 1:

                        ListAllNuts();

                        Close();

                        break;

                    case 2:

                        FindNutById();

                        Close();

                        break;

                    case 3:

                        AddNut();

                        Close();

                        break;

                    case 4:

                        RemoveNut();

                        Close();

                        break;

                    case 5:

                        UpdateNut();

                        Close();

                        break;

                    default:

                        break;

                }

            } while (operation != RETURN\_OPERATION\_ID);

        }

        private void ListAllNuts()

        {

            Console.WriteLine(new string('\*', 40));

            Console.Write(new string(' ', 18));

            Console.BackgroundColor = ConsoleColor.DarkMagenta;

            Console.WriteLine(" NUTS ");

            Console.ResetColor();

            Console.WriteLine(new string('\*', 40));

            var nuts = nutController.GetAllNuts();

            foreach (var item in nuts)

            {

                Console.BackgroundColor = ConsoleColor.DarkMagenta;

                Console.Write($"{item.Id}");

                Console.ResetColor();

                Console.WriteLine($" {item.Category} {item.Name} {item.Price}lv/kg {item.Quantity}kg.");

            }

            Console.WriteLine(new string('\*', 40));

        }

        private void FindNutById()

        {

            Console.Write("Enter ID: ");

            var id = int.Parse(Console.ReadLine());

            var nut = nutController.GetNutById(id);

            if (nut != null)

            {

                Console.WriteLine(new string('\*', 40));

                Console.WriteLine("ID: " + nut.Id);

                Console.WriteLine("Category: " + nut.Category);

                Console.WriteLine("Name: " + nut.Name);

                Console.WriteLine("Price: " + nut.Price + "lv/kg");

                Console.WriteLine("Quantity: " + nut.Quantity + "kg.");

                Console.WriteLine(new string('\*', 40));

            }

            else

            {

                Console.WriteLine("The product was not found!");

            }

        }

        private void AddNut()

        {

            var elements = ReadElements();

            Nut nut = new Nut(elements[0], elements[1], decimal.Parse(elements[2]), int.Parse(elements[3]));

            nutController.Add(nut);

            Console.WriteLine("The product was successfully added!");

        }

        private void RemoveNut()

        {

            Console.Write("Enter ID: ");

            int id = int.Parse(Console.ReadLine());

            var nut = nutController.GetNutById(id);

            if (nut!=null)

            {

                nutController.Delete(nut.Id);

                Console.WriteLine("Тhe product was deleted successfully!");

            }

            else

            {

                Console.WriteLine("The product was not found!");

            }

        }

        private void UpdateNut()

        {

            Console.Write("Enter ID: ");

            int id = int.Parse(Console.ReadLine());

            var nut = nutController.GetNutById(id);

            if (nut != null)

            {

                Console.WriteLine($"{nut.Id} {nut.Category} {nut.Name} {nut.Price}lv/kg {nut.Quantity}kg.");

                var elements = ReadElements();

                nut.Category = elements[0];

                nut.Name = elements[1];

                nut.Price = decimal.Parse(elements[2]);

                nut.Quantity = int.Parse(elements[3]);

                nutController.Update(nut);

                Console.WriteLine("The product was updated successfully!");

            }

            else

            {

                Console.WriteLine("The product was not found");

            }

        }

        private void DrinksInput()

        {

            Console.Clear();

            var operation = -1;

            do

            {

                ShowDrinksMenu();

                Console.Write("Enter number: ");

                operation = int.Parse(Console.ReadLine());

                Console.Clear();

                switch (operation)

                {

                    case 1:

                        ListAllDrinks();

                        Close();

                        break;

                    case 2:

                        FindDrinkById();

                        Close();

                        break;

                    case 3:

                        AddDrink();

                        Close();

                        break;

                    case 4:

                        RemoveDrink();

                        Close();

                        break;

                    case 5:

                        UpdateDrink();

                        Close();

                        break;

                    default:

                        break;

                }

            } while (operation != RETURN\_OPERATION\_ID);

        }

        private void ListAllDrinks()

        {

            Console.WriteLine(new string('\*', 40));

            Console.Write(new string(' ', 18));

            Console.BackgroundColor = ConsoleColor.DarkCyan;

            Console.WriteLine(" DRINKS ");

            Console.ResetColor();

            Console.WriteLine(new string('\*', 40));

            var drinks = drinkController.GetAllDrinks();

            foreach (var item in drinks)

            {

                Console.BackgroundColor = ConsoleColor.DarkCyan;

                Console.Write($"{item.Id}");

                Console.ResetColor();

                Console.WriteLine($" {item.Category} {item.Name} {item.Price}lv/pcs {item.Quantity}pcs.");

            }

            Console.WriteLine(new string('\*', 40));

        }

        private void FindDrinkById()

        {

            Console.Write("Enter ID: ");

            var id = int.Parse(Console.ReadLine());

            Drink = drinkController.GetDrinkById(id);

            if (drink!=null)

            {

                Console.WriteLine(new string ('\*',40));

                Console.WriteLine("ID: " + drink.Id);

                Console.WriteLine("Category: " + drink.Category);

                Console.WriteLine("Name: " + drink.Name);

                Console.WriteLine("Price: " + drink.Price + "lv/pcs");

                Console.WriteLine("Quantity: " + drink.Quantity + "pcs.");

                Console.WriteLine(new string('\*', 40));

            }

            else

            {

                Console.WriteLine("The product was not found!");

            }

        }

        private void AddDrink()

        {

            var elements = ReadElements();

            Drink drink = new Drink(elements[0], elements[1], decimal.Parse(elements[2]), int.Parse(elements[3]));

            drinkController.Add(drink);

            Console.WriteLine("The product was successfully added!");

        }

        private void RemoveDrink()

        {

            Console.Write("Enter ID: ");

            int id = int.Parse(Console.ReadLine());

            Drink = drinkController.GetDrinkById(id);

            if (drink!=null)

            {

                drinkController.Delete(drink.Id);

                Console.WriteLine("Тhe product was deleted successfully!");

            }

            else

            {

                Console.WriteLine("The product was not found!");

            }

        }

        private void UpdateDrink()

        {

            Console.Write("Enter ID: ");

            int id = int.Parse(Console.ReadLine());

            Drink = drinkController.GetDrinkById(id);

            if (drink!=null)

            {

                Console.WriteLine($"ID: {drink.Id} Category: {drink.Category} Name: {drink.Name} Price: {drink.Price}lv/pcs {drink.Quantity}pcs.");

                var elements = ReadElements();

                drink.Category = elements[0];

                drink.Name = elements[1];

                drink.Price = decimal.Parse(elements[2]);

                drink.Quantity = int.Parse(elements[3]);

                drinkController.Update(drink);

                Console.WriteLine("The drink was updated successfully!");

            }

            else

            {

                Console.WriteLine("The product was not found!");

            }

        }

        private void ShowPastriesMenu()

        {

            Console.WriteLine(new string('\*', 40));

            Console.Write(new string(' ', 13));

            Console.BackgroundColor = ConsoleColor.DarkYellow;

            Console.WriteLine(" PASTRY MENU ");

            Console.ResetColor();

            Console.WriteLine(new string('\*', 40));

            Console.BackgroundColor = ConsoleColor.DarkBlue;

            Console.WriteLine("1. List all pastries");

            Console.BackgroundColor = ConsoleColor.DarkYellow;

            Console.WriteLine("2. Found pastry by ID");

            Console.BackgroundColor = ConsoleColor.DarkBlue;

            Console.WriteLine("3. Add pastry");

            Console.BackgroundColor = ConsoleColor.DarkYellow;

            Console.WriteLine("4. Remove pastry");

            Console.BackgroundColor = ConsoleColor.DarkBlue;

            Console.WriteLine("5. Update pastry");

            Console.BackgroundColor = ConsoleColor.DarkYellow;

            Console.WriteLine("6. Return to main menu");

            Console.ResetColor();

            Console.WriteLine(new string('\*', 40));

        }

        private void ShowFruitsAndVegetablesMenu()

        {

            Console.WriteLine(new string('\*', 40));

            Console.Write(new string(' ', 6));

            Console.BackgroundColor = ConsoleColor.DarkGreen;

            Console.WriteLine(" FRUITS AND VEGETABLES MENU ");

            Console.ResetColor();

            Console.WriteLine(new string('\*', 40));

            Console.BackgroundColor = ConsoleColor.DarkBlue;

            Console.WriteLine("1. List all fruts and vegetables");

            Console.BackgroundColor = ConsoleColor.DarkGreen;

            Console.WriteLine("2. Found fruit or vegetable by ID");

            Console.BackgroundColor = ConsoleColor.DarkBlue;

            Console.WriteLine("3. Add fruit or vegetable");

            Console.BackgroundColor = ConsoleColor.DarkGreen;

            Console.WriteLine("4. Remove fruit or vegetable");

            Console.BackgroundColor = ConsoleColor.DarkBlue;

            Console.WriteLine("5. Update fruit or vegetable");

            Console.BackgroundColor = ConsoleColor.DarkGreen;

            Console.WriteLine("6. Return to main menu");

            Console.ResetColor();

            Console.WriteLine(new string('\*', 40));

        }

        private void ShowNutsMenu()

        {

            Console.WriteLine(new string('\*', 40));

            Console.Write(new string(' ', 16));

            Console.BackgroundColor = ConsoleColor.DarkMagenta;

            Console.WriteLine(" NUTS MENU ");

            Console.ResetColor();

            Console.WriteLine(new string('\*', 40));

            Console.BackgroundColor = ConsoleColor.DarkBlue;

            Console.WriteLine("1. List all nuts");

            Console.BackgroundColor = ConsoleColor.DarkMagenta;

            Console.WriteLine("2. Found nut by ID");

            Console.BackgroundColor = ConsoleColor.DarkBlue;

            Console.WriteLine("3. Add nut");

            Console.BackgroundColor = ConsoleColor.DarkMagenta;

            Console.WriteLine("4. Remove nut");

            Console.BackgroundColor = ConsoleColor.DarkBlue;

            Console.WriteLine("5. Update nut");

            Console.BackgroundColor = ConsoleColor.DarkMagenta;

            Console.WriteLine("6. Return to main menu");

            Console.ResetColor();

            Console.WriteLine(new string('\*', 40));

        }

        private void ShowDrinksMenu()

        {

                Console.WriteLine(new string('\*', 40));

                Console.Write(new string(' ', 15));

                Console.BackgroundColor = ConsoleColor.DarkCyan;

                Console.WriteLine(" DRINKS MENU ");

                Console.ResetColor();

                Console.WriteLine(new string('\*', 40));

                Console.BackgroundColor = ConsoleColor.DarkBlue;

                Console.WriteLine("1. List all drinks");

                Console.BackgroundColor = ConsoleColor.DarkCyan;

                Console.WriteLine("2. Found drink by ID");

                Console.BackgroundColor = ConsoleColor.DarkBlue;

                Console.WriteLine("3. Add drink");

                Console.BackgroundColor = ConsoleColor.DarkCyan;

                Console.WriteLine("4. Remove drink");

                Console.BackgroundColor = ConsoleColor.DarkBlue;

                Console.WriteLine("5. Update drink");

                Console.BackgroundColor = ConsoleColor.DarkCyan;

                Console.WriteLine("6. Return to main menu");

                Console.ResetColor();

                Console.WriteLine(new string('\*', 40));

        }

        private void Close()

        {

            Console.Write("Press any key to continue... ");

            Console.ReadKey();

            Console.Clear();

        }

        private List<string> ReadElements()

        {

            var list = new List<string>();

            Console.Write("Enter category: ");

            list.Add(Console.ReadLine());

            Console.Write("Enter name: ");

            list.Add(Console.ReadLine());

            Console.Write("Enter price: ");

            list.Add(Console.ReadLine());

            Console.Write("Enter quantity: ");

            list.Add(Console.ReadLine());

            return list;

        }

    }

* Program.cs

class Program

    {

        static void Main(string[] args)

        {

            System.Timers.Timer aTimer = new System.Timers.Timer();

            aTimer.Elapsed += new ElapsedEventHandler(OnTimedEvent);

            aTimer.AutoReset = false;

            aTimer.Start();

            Thread.Sleep(7000);

            Console.CursorVisible = true;

            Display display = new Display();

        }

        /// <summary>

        /// Splash screen with countdown timer.

        /// </summary>

        /// <param name="source"></param>

        /// <param name="e"></param>

        private static void OnTimedEvent(object source, ElapsedEventArgs e)

        {

            for (int a = 6; a > 0; a--)

            {

            Console.CursorVisible = false;

            Console.Write(new string(' ', 2));

            Console.WriteLine(new string('\*', 42));

            Console.Write(new string(' ', 2));

            Console.WriteLine("\* Market Created by Muhamed and Plmanena \*");

            Console.Write(new string(' ', 2));

            Console.WriteLine(new string('\*', 42));

            Console.WriteLine("");

            Console.Write("The program will start after: {0}", a);

            System.Threading.Thread.Sleep(1000);

            Console.Clear();

            }

        }

    }