****

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«РОССИЙСКАЯ АКАДЕМИЯ НАРОДНОГО ХОЗЯЙСТВА И ГОСУДАРСТВЕННОЙ СЛУЖБЫ ПРИ ПРЕЗИДЕНТЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»**

КОЛЛЕДЖ МНОГОУРОВНЕВОГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

ОТЧЕТ

По учебной практике УП.01.01 Разработка программных модулей

специальность 09.02.07 Информационные системы и программирование

Выполнил студент:

Капитонов Р.В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(подпись)

Гусятинер Л. Б.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(оценка)

Москва, 2023

**Оглавление**

[Разработка проекта с графическим интерфейсом 3](#_Toc153120834)

[1.1. Техническое задание 3](#_Toc153120835)

[1.1.2. Структура базы данных 3](#_Toc153120836)

[1.2. Построение диаграммы вариантов использования. 6](#_Toc153120837)

[1.3. Разработка сценария проекта 6](#_Toc153120838)

[2. Разработка приложения 8](#_Toc153120839)

[2.2. Разработка главного модуля 8](#_Toc153120840)

[2.3. Разработка входящих модулей 10](#_Toc153120841)

[2.4. Тестирование и отладка 13](#_Toc153120842)

[Приложение 1. Руководство программиста 16](#_Toc153120843)

[Приложение 2. Руководство пользователя 17](#_Toc153120844)

[Приложение 3. Листинги 18](#_Toc153120845)

Добавить ТЗ и починить титульник

# Разработка проекта с графическим интерфейсом

В данном проекте мы работаем в основе с двумя библиотеками “MAUI” и “EntityFramework”.

Для проектов использующих MAUI необходимо скачать и установить расширение для Visual Studio.

Для использования EntityFramework нужно установить пакет NuGet.

Использование через using:

using Microsoft.EntityFrameworkCore;

* 1. Техническое задание
     1. Описание бизнес процесса

• Клиент через внешний сервис создаёт запрос в базе данных.

• Сотрудник обрабатывает запрос, а также вносит изменения в книги при необходимости.

• Администратор имеет право просматривать изменения, которые внесли другие пользователи.

* + 1. Структура базы данных

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описаниеНаполнение таблиц и их структура видны на Рисунках с 1 по 6.

Рис. 1. Таблица “Authors”

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Рис. 2. Таблица “Users”

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рис. 3. Таблица “Books”

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рис. 4. Таблица “Genres”

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, число, снимок экрана, Шрифт

Автоматически созданное описаниеРис. 5. Таблица “Logs”

Рис. 6. Таблица “Orders ”

* + 1. Запросы к базе данных

Создаются на основе требуемых данных автоматически через EntityFramework через методы.

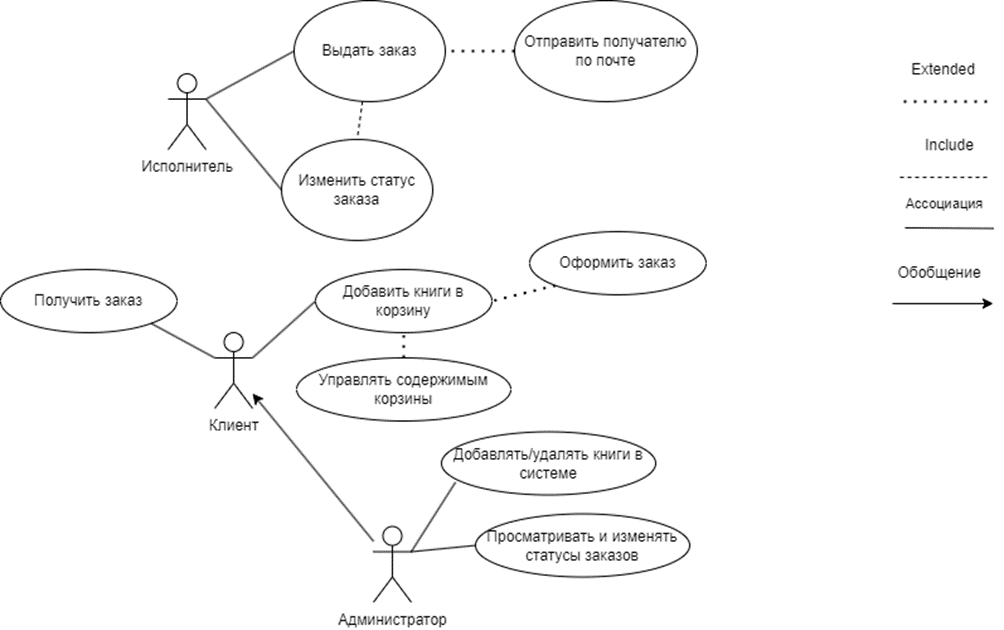
* 1.  Построение диаграммы вариантов использования.

Диаграмма вариантов использования предсавлена на Рисунке 7.

Рис. 7. Диаграмма вариантов использования

* 1. Разработка сценария проекта

В данном разделе приведен сценарий использования программы пользователем (Рисунок 8).

Изображение выглядит как диаграмма, План, Технический чертеж, текст

Автоматически созданное описание

Рис. 8. Сценарий проекта

* 1. Построение диаграммы классов

В данном разделе находится диаграмма классов приложения.(Рисунок 9)

Изображение выглядит как текст, диаграмма, План, схематичный

Автоматически созданное описание

Рис. 9. Диаграмма классов

1. **Разработка приложения**
   1. Состав проекта

Два модуля:

* Classes
* BookDesktop
  1. Разработка главного модуля

Главный модуль состоит из файла Index. Перенаправление на него выполняется сразу при запуске программы. Его также можно назвать главным меню.

Листинг 1. Главный модуль Index

@page "/"

@inject AuthService authService

@using BookDesktop.Models

@using BookDesktop.Service

<div class="w-50 m-auto">

<h3>Авторизация</h3>

<EditForm Model="@Person" OnValidSubmit="@Submit">

<DataAnnotationsValidator/>

<ValidationSummary/>

<div class="mb-3">

<label for="exampleInputEmail1" class="form-label">Логин</label>

<InputText class="form-control" @bind-Value="Person!.Login" id="exampleInputEmail1"/>

</div>

<div class="mb-3">

<label for="exampleInputPassword1" class="form-label">Пароль</label>

<InputText type="password" class="form-control" @bind-Value="Person!.Password" id="exampleInputPassword1"/>

</div>

<button type="submit" class="btn btn-primary">Войти</button>

</EditForm>

</div>

@code

{

Person Person = default!;

private void Submit()

{

if (Person.Login != "" && Person.Password != "")

authService.Login(Person.Login, Person.Password);

}

protected override void OnInitialized()

{

#if DEBUG

authService.Login("root", "root");

#else

base.OnInitialized();

Person = new();

#endif

}

}

* 1. Разработка входящих модулей

Описание спецификаций к модулям

Всего разработано 3 основных модуля:

* Окно каталога книг (Books)
* Окно просмотра логов(Logs)
* Окно просмотра заказов(Orders)

Описание модулей:

Окно каталога книг – на данном окне находится список книг, которые можно изменять и добавлять. Для удобства добавлена фильтрация по жанрам.

Рассмотрим функцию добавления новой книги.

Листинг 2. Функция добавления новой книги.

private void ChangeBook(Book book)

{

bookChangeService.ChangeBook(book);

}

После перенаправления на другую страницу через сервис используй функцию ChangeBook

Листинг 3. Функция сервиса для изменения книги

public void ChangeBook(Book bookLocal)

{

navigation.NavigateTo($"book\_redact/{bookLocal.Id}");

}

Пользователь попадает на вспомогательную страницу, где через интерфейс выбирает необходимые ему параметры. После нажатия кнопки подтверждения изменений вызывается функция BookCommit\_change.

Листинг 4. Функция подтверждения изменений.

public async void BookCommit\_change()

{

bookLocal.Genres.Clear();

foreach (var id in checkedGenres.Where(x => x.Value).Select(x => x.Key))

{

bookLocal.Genres.Add(genres.First(x => x.Id == id));

}

bookLocal.LastRedactor = authService.CurrentUser!;

dbcontext.Update(bookLocal);

dbcontext.Logs.Add(new Log

{

Operation = $"Внесены изменения в книгу с Id = {bookLocal.Id}",

UserId = authService.CurrentUser!.Id,

});

await dbcontext.SaveChangesAsync();

navigation.NavigateTo("books");

}

Рассмотрим блок-схему этой функции на Рисунке 10.

Изображение выглядит как текст, диаграмма, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 10. Блок-схема функции BookCommit\_change.

* 1. Тестирование и отладка

В ходе написания проекта при попытке запустить программу были получены ошибки (Рисунок 11):

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, снимок экрана

Автоматически созданное описание

Рис. 11. Ошибки

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Автоматически созданное описаниеПри проверке кода были исправлены найденные ошибки, в результате при запуске программы ошибок не было (Рисунок 12):

Рис. 12. Успешная сборка

* 1. Дневник

|  |  |  |
| --- | --- | --- |
| Дата | Содержание работ | Отметка о прохождении |
| 11.12 | 36. Промежуточная аттестация: дифференцированный зачет |  |
| 11.12 | 35. Разработать руководство пользователя: Рекомендации по освоению. |  |
| 04.12 | 34. Разработать руководство пользователя: Аварийные ситуации. |  |
| 04.12 | 33. Разработать руководство пользователя: Описание операций. |  |
| 27.11 | 32. Разработать руководство пользователя: Подготовка к работе. |  |
| 27.11 | 31. Разработать руководство пользователя: Введение. Назначение и условия применения. |  |
| 27.11 | 30. Разработать руководство программиста: Сообщения |  |
| 20.11 | 29. Разработать руководство программиста: Входные и выходные данные |  |
| 20.11 | 28. Разработать руководство программиста: Обращение к программе |  |
| 20.11 | 27. Разработать руководство программиста: Характеристика программы |  |
| 06.11 | 26. Разработать руководство программиста: Назначение и условия применения программ |  |
| 06.11 | 25. По результатам тестирования провести рефакторинг программного продукта |  |
| 30.10 | 24. Провести тестирование программного продукта |  |
| 30.10 | 23. Провести тестирование модулей |  |
| 30.10 | 22. Провести тестирование главного модуля |  |
| 23.10 | 21. Разработать тестовые наборыx |  |
| 23.10 | 20. Разработка классов подсистемы отчётов |  |
| 16.10 | 19. Разработка классов подсистемы связи с внешними данными |  |
| 16.10 | 18. Разработка классов подсистемы вычислений |  |
| 16.10 | 17. Разработка методов классов и функций главного модуля |  |
| 09.10 | 16. Разработка классов главного модуля |  |
| 09.10 | 15. Оформление раздела "Атрибуты качества" |  |
| 02.10 | 14. Оформление раздела "Требования к внешним интерфейсам" |  |
| 02.10 | 13. Оформление раздела "Требования к внешним данным" |  |
| 02.10 | 12. Оформление раздела "Функции системы" |  |
| 26.09 | 11. Оформление спецификации программного продукта. Оформление введения и раздела "Общее описание" |  |
| 26.09 | 10. Разработка простой спецификации учебной задачи: Атрибуты качества |  |
| 19.09 | 9. Разработка простой спецификации учебной задачи: Требования к внешним интерфейсам |  |
| 19.09 | 8. Разработка простой спецификации учебной задачи: Требования к данным |  |
| 16.09 | 7. Разработка простой спецификации учебной задачи: Функции системы |  |
| 15.09 | 6. Разработка простой спецификации учебной задачи: Введение. Общее описание |  |
| 12.09 | 5. Оформление математического алгоритма при помощи программных средств |  |
| 12.09 | 4. Изучение средств визуализации алгоритмов |  |
| 09.09 | 3. Формирование алгоритма решения задачи по обработке данных |  |
| 05.09 | 2. Формирование алгоритма решения вычислительной задачи |  |
| 05.09 | 1. Вводный инструктаж. |  |

# Использованные источники

1. Сайт документации языка C#

https://learn.microsoft.com/ru-ru/dotnet/csharp/

1. Сайт примеров оформления на фреймворк .NET MAUI Blazor

https://getbootstrap.com/

# Приложение 1. Руководство программиста

Программа предназначена для обработки заказов и учёта книг в системе.

Условия эксплуатации программы:

* Подключение к серверу базу данных;
* На персональных компьютерах с процессором не ниже 2ГГц, минимум оперативной памяти 2 ГБ и операционная система не ниже Windows 10.

Рассматриваются общие сведения о программе, структура программы, настройка программы, проверка программы, дополнительные возможности программы.

**1 Назначение и условия применения программы**

1.1 Назначение программы

Описываемая программа предназначена для обработки заказов, поступающих из другого сервиса и учёта книг в системе.

1.2 Функции программы

* Авторизация пользователя
* Просмотр книг в базе данных
* Изменение существующих и добавление новых книг
* Просмотр действий пользователей
* Просмотр и изменение статуса заказов

1.3 Использование оперативной памяти

Количество необходимой оперативной памяти равно около 500 МБ, не учитывая объём данных в БД, который принимает клиент.

1.4 Требования к программному обеспечению

Операционная система Windows 10 и выше.

**2 Характеристики программы**

2.1 Средства проверки правильности выполнения программы

Правильность работы программы можно проверить только лишь экспериментально. Наилучший способ, моделирование реальной работы, с мониторингом изменений данных в базе данных.

2.2 Функционирование программы после сбоев

После сбоя достаточно перезапустить программный продукт. Никаких дополнительных действий производить не требуется.

**3 Обращение к программе**

3.1 Способы вызова программы с различных носителей данных

Программа вызывается со статических дисков, на которые программу можно установить.

3.2 Входные точки в программу

Исполняемым файлом программы является файл "BookDesktop.exe".

3.3 Способы передачи управления и параметров данных

Передача данных производится в режиме диалога программы с пользователем.

3.4 Обращение к программе – приложению

С помощью программ файл - менеджеров зайти в каталог, в который установлена программа и запустить файл " BookDesktop.exe".

**4 Входные и выходные данные**

Отсутствуют в связи с работой приложения с базой данных и наличии интерфейса.

**5 Сообщения**

5.1 Сообщения программисту и действия по ним

Сообщений не предусмотрено.

# Приложение 2. Руководство пользователя

Название приложения: BookDesktop

Версия: 1.2

Дата: 10.12.2023

**1. Введение. Назначение и условия применения**

1.1 Назначение

BookDesktop – приложение для работы с книгами и заказами, которые поступают от внешнего сервиса.

1.2. Условия применения

Для работы с приложением необходимо следующее программное и аппаратное обеспечение:

Операционная система:Windows 10 и выше]

Процессор: 2ГГц

Оперативная память: от 500 МБ свободной оперативной памяти

Свободное место на диске: 500 МБ свободного места на диске

**2. Подготовка к работе**

2.1. Установка

Для установки приложения необходимо выполнить следующие действия:

* Загрузить установочный файл приложения.
* Запустить установочный файл.
* Следовать инструкциям на экране.

2.2. Регистрация

Для использования приложения необходима учётная запись

Регистрация возможна только через администратора системы.

**3. Описание операций**

В данном разделе описаны основные операции, которые можно выполнять с помощью приложения.

3.1. Добавление книг в Каталог

Чтобы добавить книгу в каталог, необходимо выполнить следующие действия:

* В верхней строке приложения выбрать пункт "Каталог".
* Нажать кнопку «Добавить».
* Внести необходимые данные
* Нажмите кнопку "Сохранить".

3.2. Авторизация

Чтобы авторизоваться в приложении, необходимо выполнить следующие действия:

* В стартовом меню приложения введите имя пользователя и пароль.
* Нажмите кнопку "Войти".

3.1. Изменение книг в Каталоге

Чтобы изменить книгу в каталоге, необходимо выполнить следующие действия:

* В верхней строке приложения выбрать пункт "Каталог".
* Нажать кнопку «Изменить» в строке с интересующей вас книгой.
* Изменить необходимые данные
* Нажмите кнопку "Сохранить".

3.1. Изменение статуса заказов

Чтобы изменить статус заказа в системе, необходимо выполнить следующие действия:

* В верхней строке приложения выбрать пункт "Заказы".
* Нажать кнопку «Изменить статус» в окне с интересующим вас заказом.
* Выбрать необходимый статус.

3.1. Просмотр действий пользователей

Чтобы просмотреть действия других пользователей в системе, необходимо выполнить следующие действия:

* В верхней строке приложения выбрать пункт "Логи".

Руководство пользователя

Название приложения: [Название приложения]

Версия: [Версия приложения]

Дата: [Дата создания руководства]

1. Введение. Назначение и условия применения

1.1. Назначение

[Название приложения] - это десктоп-приложение, предназначенное для [целевое назначение приложения].

1.2. Условия применения

Для работы с приложением необходимо следующее программное и аппаратное обеспечение:

Операционная система: [список поддерживаемых операционных систем]

Процессор: [минимальные требования к процессору]

Оперативная память: [минимальные требования к оперативной памяти]

Свободное место на диске: [минимальные требования к свободному месту на диске]

2. Подготовка к работе

2.1. Установка

Для установки приложения необходимо выполнить следующие действия:

Загрузите установочный файл приложения с официального сайта.

Запустите установочный файл.

Следуйте инструкциям на экране.

2.2. Регистрация

Для использования некоторых функций приложения необходимо зарегистрироваться. Для этого необходимо выполнить следующие действия:

В главном меню приложения выберите пункт "Регистрация".

Введите необходимую информацию.

Нажмите кнопку "Зарегистрироваться".

3. Описание операций

В данном разделе описаны основные операции, которые можно выполнять с помощью приложения.

3.1. Создание учетной записи

Чтобы создать учетную запись, необходимо выполнить следующие действия:

В главном меню приложения выберите пункт "Создать учетную запись".

Введите необходимую информацию.

Нажмите кнопку "Создать".

3.2. Авторизация

Чтобы авторизоваться в приложении, необходимо выполнить следующие действия:

В главном меню приложения выберите пункт "Авторизация".

Введите имя пользователя и пароль.

Нажмите кнопку "Войти".

3.3. Загрузка файлов

Чтобы загрузить файл в приложение, необходимо выполнить следующие действия:

В главном меню приложения выберите пункт "Загрузить".

Выберите файл, который хотите загрузить.

Нажмите кнопку "Загрузить".

3.4. Просмотр файлов

Чтобы просмотреть файл в приложении, необходимо выполнить следующие действия:

В главном меню приложения выберите пункт "Файлы".

В списке файлов выберите файл, который хотите просмотреть.

Нажмите кнопку "Просмотреть".

3.5. Редактирование файлов

Чтобы отредактировать файл в приложении, необходимо выполнить следующие действия:

В главном меню приложения выберите пункт "Файлы".

В списке файлов выберите файл, который хотите отредактировать.

Нажмите кнопку "Редактировать".

3.6. Удаление файлов

Чтобы удалить файл из приложения, необходимо выполнить следующие действия:

В главном меню приложения выберите пункт "Файлы".

В списке файлов выберите файл, который хотите удалить.

Нажмите кнопку "Удалить".

**4. Аварийные ситуации**

В данном разделе описаны действия, которые необходимо предпринять в случае возникновения аварийной ситуации.

4.1. Ошибка запуска приложения

Если приложение не запускается, необходимо проверить следующие пункты:

* Убедитесь, что на компьютере установлены все необходимые системные требования.
* Попробуйте запустить приложение от имени администратора.
* Переустановите приложение.

4.2. Ошибка подключения к серверу

Если приложение не может подключиться к серверу, необходимо проверить следующие пункты:

Убедитесь, что на компьютере подключено к Интернету.

Убедитесь, что сервер доступен.

Попробуйте перезапустить приложение.

**5. Рекомендации по освоению**

Для быстрого освоения приложения рекомендуется выполнить следующие действия:

* Прочитать руководство пользователя.
* Потренироваться в использовании приложения на простых задачах.

# Приложение 3. Листинги

## Листинг 5. Класс AppDbContext. Хранит списки объектов БД и позволяет к ним обращаться.

using Microsoft.EntityFrameworkCore;

namespace Classes;

public class AppDbContext : DbContext

{

public AppDbContext(DbContextOptions options) : base(options)

{

}

protected AppDbContext()

{

}

public DbSet<User> Users { get; set; }

public DbSet<Book> Books { get; set; }

public DbSet<Author> Authors { get; set; }

public DbSet<Genre> Genres { get; set; }

public DbSet<Order> Orders { get; set; }

public DbSet<Log> Logs { get; set; }

//public string databasepath = $"Data Source={@"C:\Projects\CourseWork\MyDatabase.db"}";

//"Host=127.0.0.1;Port=5432;Database=KRVBooks;Username=TestGroupLocalhost;Password=postgres;";

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

// Путь к базе данных SQLite. Файл будет создан в текущей директории приложения. (bin/Debug/)

//var databasePath = System.IO.Path.Combine(System.Environment.CurrentDirectory, "MyDatabase.db");

//optionsBuilder.UseSQlite($"Data Source={databasePath}");

//optionsBuilder.UseNpgsql(databasepath);

//optionsBuilder.UseSqlite(databasepath);

}

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

modelBuilder.Entity<Book>()

.HasMany(b => b.Genres)

.WithMany(bg => bg.Books);

modelBuilder.Entity<Order>()

.HasMany(o => o.Books)

.WithMany(b => b.Orders);

modelBuilder.Entity<User>()

.HasMany(x => x.Logs)

.WithOne(b => b.User);

}

}

## Листинг 6. Класс MauiProgram. Является стартовым для программы.

using ABI.Windows.UI.Popups;

using BookDesktop.Pages;

using BookDesktop.Service;

using Classes;

using Microsoft.EntityFrameworkCore;

using Microsoft.Extensions.Logging;

namespace BookDesktop;

public static class MauiProgram

{

public static MauiApp CreateMauiApp()

{

Environment.CurrentDirectory =

AppDomain.CurrentDomain.BaseDirectory.Substring(0,

AppDomain.CurrentDomain.BaseDirectory.IndexOf("BookDesktop"));

var builder = MauiApp.CreateBuilder();

builder

.UseMauiApp<App>()

.ConfigureFonts(fonts => { fonts.AddFont("OpenSans-Regular.ttf", "OpenSansRegular"); });

builder.Services.AddMauiBlazorWebView();

#if DEBUG

builder.Services.AddBlazorWebViewDeveloperTools();

builder.Logging.AddDebug();

#endif

builder.Services.AddSingleton<AuthService>();

builder.Services.AddScoped<BookChangeService>();

builder.Services.AddDbContext<AppDbContext>((options) =>

{

options.UseSqlite($"Data Source={Environment.CurrentDirectory + @"\MyDatabase.db"}");

});

builder.Services.AddSingleton<BinService>();

var app = builder.Build();

var dbContext = app.Services.GetRequiredService<AppDbContext>();

dbContext.Database.EnsureCreated();

#if DEBUG

DbDataFill(dbContext);

#endif

return app;

}

private async static void DbDataFill(AppDbContext dbContext)

{

if (dbContext.Users.ToList().Count < 4) //root

{

dbContext.Users.Add(new User

{

Id = 1,

Email = "RootEmail@gmail.com",

Login = "root",

Password = "root",

PermissionLvl = 0

});

dbContext.Users.Add(new User //admin

{

Id = 2,

Email = "AdminEmail@gmail.com",

Login = "admin",

Password = "admin",

PermissionLvl = 1

});

dbContext.Users.Add(new User //employeer1

{

Id = 3,

Email = "Emp1Email@gmail.com",

Login = "employeer1",

Password = "employeer1",

PermissionLvl = 2

});

dbContext.Users.Add(new User //employeer2

{

Id = 4,

Email = "Emp2Email@gmail.com",

Login = "employeer2",

Password = "employeer2",

PermissionLvl = 2

});

dbContext.SaveChanges();

}

if (dbContext.Authors.Where(author => author.Name == "AuthorTestName").ToList().Count == 0) //Author

{

dbContext.Authors.Add(new Author { Id = 1, Name = "AuthorTestName" });

dbContext.SaveChanges();

}

if (dbContext.Books.Where(book => book.Name.Contains("testbookName")).ToList().Count == 0) //Book

{

for (int i = 1; i <= 10; i++)

{

var lastGenre = new Genre();

if (dbContext.Genres.Where(x => x.Id == i).FirstOrDefault() == null)

{

lastGenre = new Genre { Id = i, Name = $"{i}TestGenreName" };

dbContext.Genres.Add(lastGenre);

}

var book = new Book(

i, $"testbookName{i}", dbContext.Authors.First(), i \* 100, i \* 10);

book.Genres.Add(lastGenre);

dbContext.Books.Add(book);

}

dbContext.SaveChanges();

}

if (await dbContext.Orders.CountAsync() == 0)

{

foreach (var book in await dbContext.Books.ToListAsync())

{

dbContext.Orders.Add(new Order(Enum.GetValues<OrderStatus>().RandomElement(), dbContext.Users.RandomElement().Id, book, dbContext.Books.RandomElement()));

}

dbContext.SaveChanges();

}

}

public static T RandomElement<T>(this IEnumerable<T> enumerable)

{

return enumerable.RandomElementUsing<T>(new Random());

}

public static T RandomElementUsing<T>(this IEnumerable<T> enumerable, Random rand)

{

int index = rand.Next(0, enumerable.Count());

return enumerable.ElementAt(index);

}

}