МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

ОТЧЁТ

по лабораторной работе №3

по дисциплине

Сервис-ориентированные системы

РУКОВОДИТЕЛЬ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Жевнерчук В.Д.

СТУДЕНТ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Уткин А.О

21-ПО

Работа защищена «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_

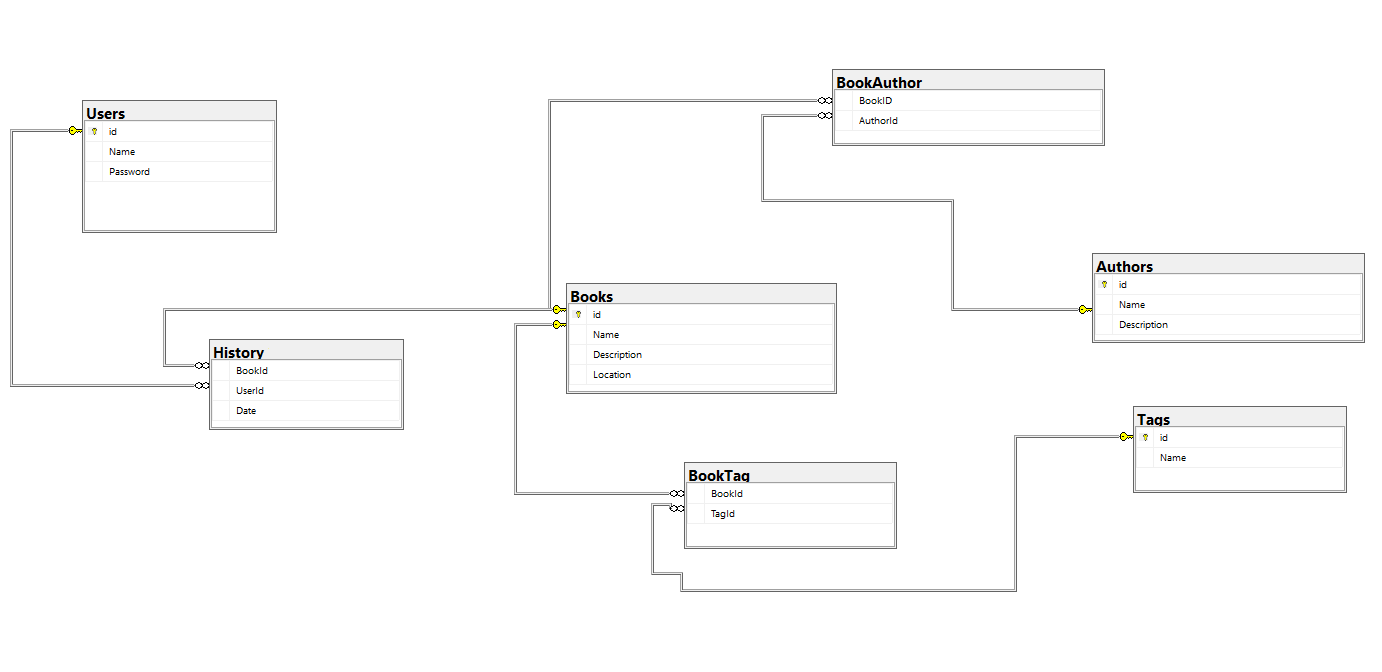
С оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нижний Новгород 2024

# Описание сервиса

Тема проекта: «Электронная библиотека вуза». В электронной библиотеке есть книги, а у книг есть авторы и категории. По категориям можно найти книги с такой же категорией. Также можно найти книги того же автора. Есть возможность скачать и загрузить книгу, но только авторизовавшимся пользователям. Каждый пользователь имеет личную историю скачиваний.

# Диаграмма БД



# Описание слоев разделения данных- как реализуется слой и его предназначение

В сервисе присутствуют следующие слои:

* БД- база данных, отвечающая за хранение данных.
* DAO- класс api отвечающий за взаимодействие с бд посредством SQL запросов.
* Business- класс api отвечающий за преобразование данных с БД в экземпляры классов
* Модели – классы предназначенные для передачи данных
* Контроллеры- классы отвечающие за преобразование экземпляров классов моделей в json формат и передачу их до клиента
* Клиент – отвечает за запрос и получение данных

# Описание разработанных точек доступа к сервису

**https://localhost:7022/Author**

GET- получить всех авторов

GET – с параметром id получить элемент по id

POST – с параметрами id и name –добавляет книге с id автора с именем name

POST – с параметром name –добавляет нового автора

DELETE – с параметрами bookid и authorid удаляет связь между книгой с id = bookid и автора с id=authored

DELETE – с параметром id удаляет автора с таким id

UPDATE – с параметрами id и name – меняет имя автора с id

**https://localhost:7022/Tag**

GET- получить все категории

GET – с параметром id получить элемент по id

POST – с параметрами id и name –добавляет книге с id категорию с именем name

POST – с параметром name –добавляет новую категорию

DELETE – с параметрами bookid и ефпid удаляет связь между книгой с id = bookid и тега с id=tagid

DELETE – с параметром id удаляет тег с таким id

UPDATE – с параметрами id и name – меняет имя тега с id

**https://localhost:7022/Book**

GET- получить все книги

GET – с параметром id получить элемент по id

POST – с параметром name –добавляет новую книгу

DELETE – с параметром id удаляет книгу с таким id

UPDATE – с параметрами id и name – меняет имя книги с id

[**https://localhost:7022/User**](https://localhost:7022/User)

GET – с параметром name и password получить id по имени и паролю, если такого нет возвращает 401

POST – с параметром name и password –добавляет нового пользователя

GET – с параметром id возвращает историю пользователя

[**https://localhost:7022/Files**](https://localhost:7022/Files)

**GET- скачать файл с сервера**

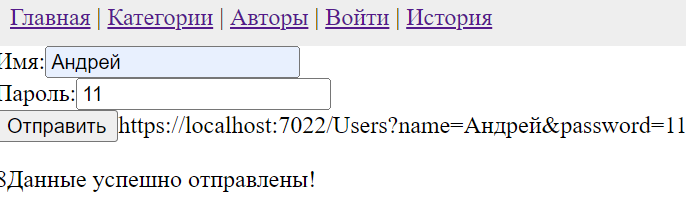
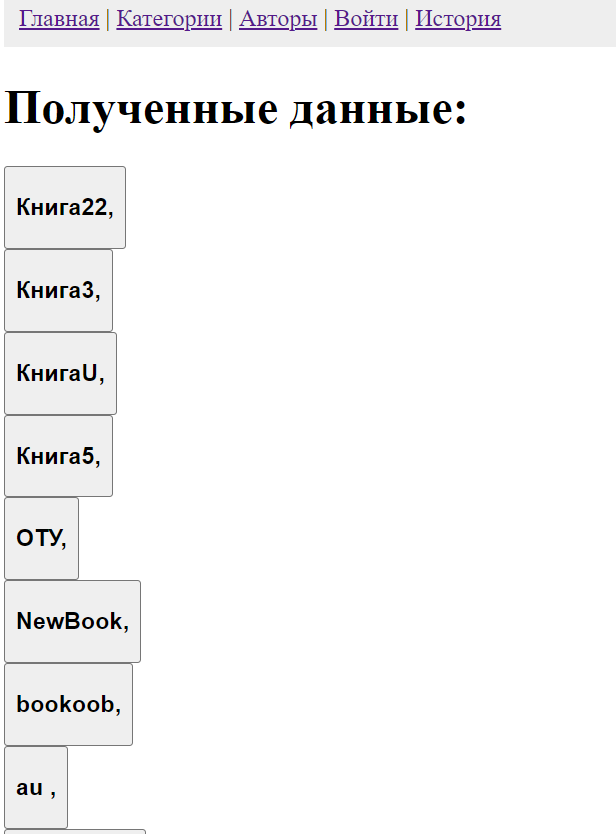
**POST – загрузить файл на сервер**

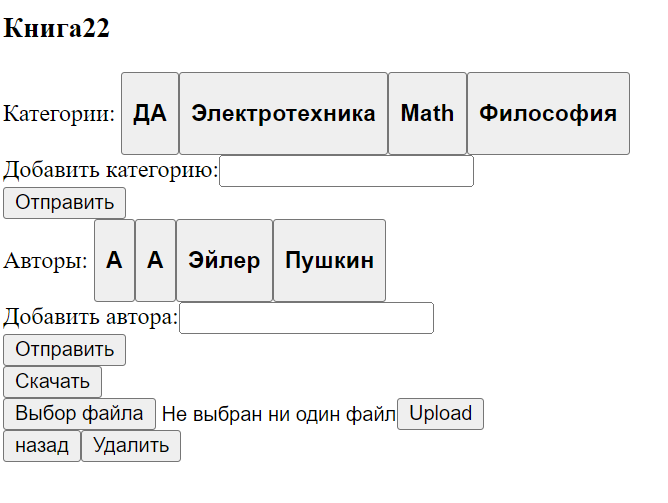
# Фронтенд

Фреймворком для создания фронтенда был выбран реакт. React — это популярная библиотека для создания пользовательских интерфейсов, разработанная компанией Facebook и впервые представлена в 2013 году. Она предназначена для упрощения разработки сложных приложений, особенно тех, которые требуют частого обновления данных и динамического взаимодействия с пользователем.

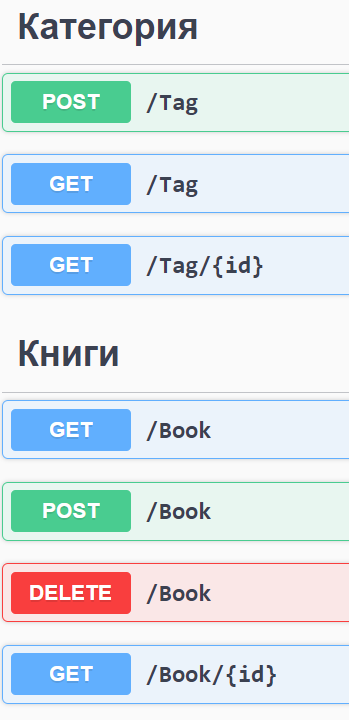
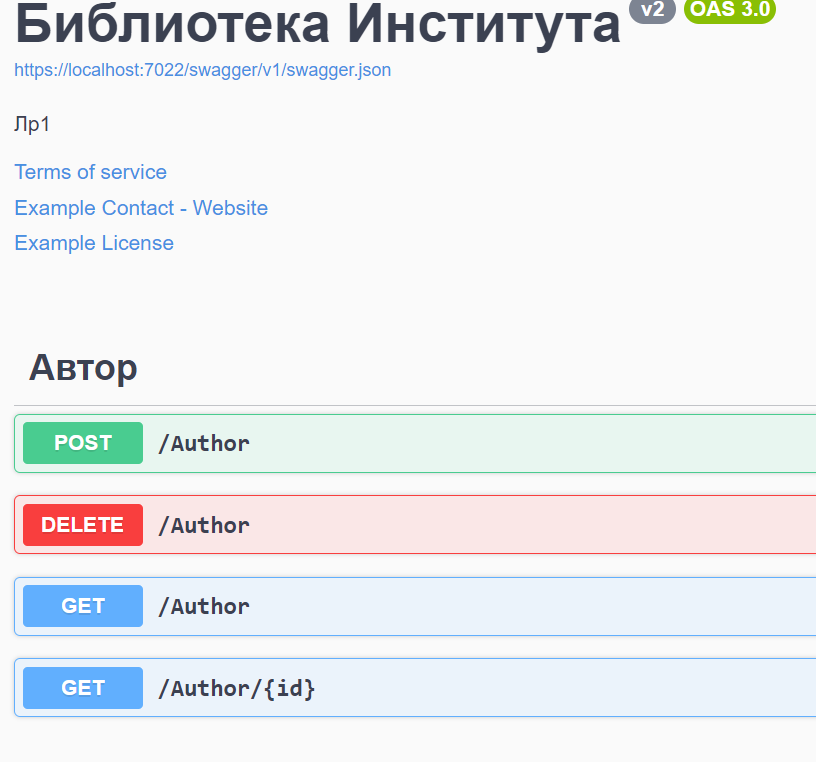
Для получения данных от бекенда использовалась функция fetch.fetch — это встроенный метод JavaScript, который используется для выполнения HTTP-запросов. В контексте React он часто применяется для взаимодействия с API и получения данных, которые затем можно использовать в компонентах.

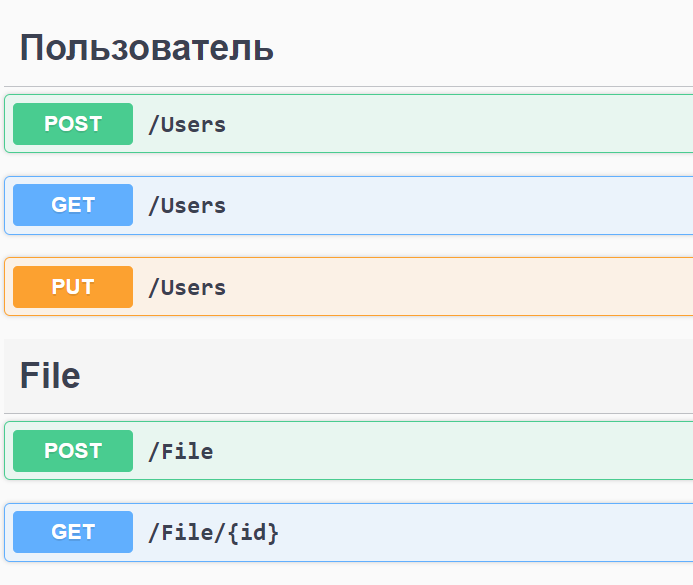
Вот некоторые скриншоты фронтенда.

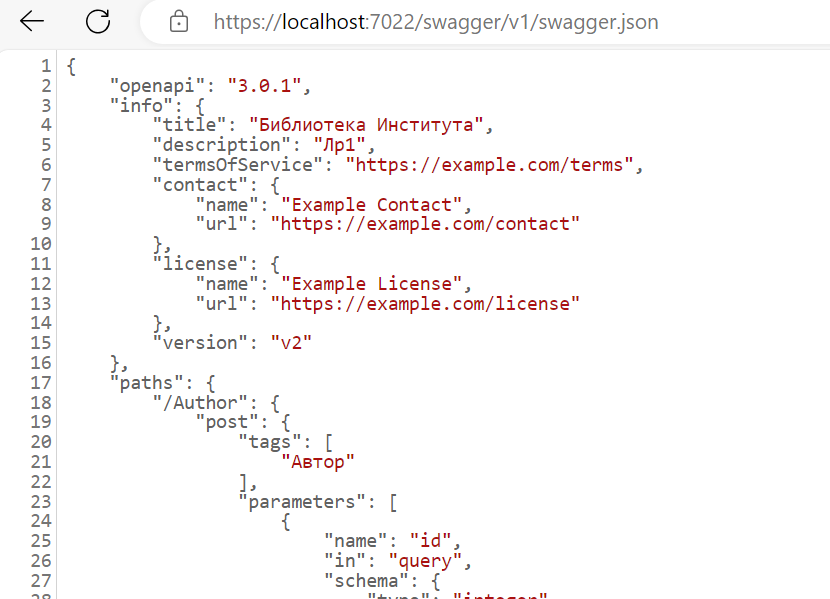




# API







# Фрагменты кода

## Контроллер Author

using Microsoft.AspNetCore.Mvc;

namespace СОСapi.Controllers

{

[ApiController]

[Route("[controller]")]

public class Author : ControllerBase

{

[Tags("Автор")]

[HttpPost]

public IActionResult Post(int id, string name)

{

Business bn = new Business();

bn.BookAuthor(id, name);

return Ok();

}

[Tags("Автор")]

[HttpDelete]

public IActionResult Delete(int id, string name)

{

Business bn = new Business();

bn.DeliteBookAuthor(id, name);

return Ok();

}

[Tags("Автор")]

[HttpGet]

[ProducesResponseType(typeof(List<global::Author\_short>), 200)]

public IActionResult GetAuthors()

{

Business bn = new Business();

List<Author\_short> authors = bn.AllAuthors();

return Ok(authors);

}

[Tags("Автор")]

[HttpGet("{id}")]

public IActionResult AllAboutAu(int id)

{

Business bn = new Business();

var myModel = bn.AuthorInf(id);

return Ok(myModel); ;

}

}

}

## Контроллер Tag

using Microsoft.AspNetCore.Mvc;

namespace СОСapi.Controllers

{

[ApiController]

[Route("[controller]")]

public class TagController : ControllerBase

{

[Tags("Категория")]

[HttpPost]

public IActionResult Post(int id, string name)

{

Business bn = new Business();

bn.BookTag(id, name);

return Ok();

}

[Tags("Категория")]

[HttpGet]

[ProducesResponseType(typeof(List<global::Tag>), 200)]

public IActionResult GetTags()

{

Business bn = new Business();

List<Tag> authors = bn.Tags();

return Ok(authors);

}

[Tags("Категория")]

[HttpGet("{id}")]

public IActionResult AllAboutAu(int id)

{

Business bn = new Business();

List<int> booksId = bn.Tagged(id);

List<global::Book> books = new List<global::Book>();

foreach (var book in booksId)

{

Book bk = bn.BookInf(book);

books.Add(bk);

}

var v = books.ToArray();

return Ok(v);

}

}

}

## Program.cs

using Microsoft.OpenApi.Models;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.

builder.Services.AddControllers();

// Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen();

builder.Services.AddSwaggerGen(options =>

{

options.SwaggerDoc("v1", new OpenApiInfo

{

Version = "v2",

Title = "Библиотека Института",

Description = "Лр1",

TermsOfService = new Uri("https://example.com/terms"),

Contact = new OpenApiContact

{

Name = "Example Contact",

Url = new Uri("https://example.com/contact")

},

License = new OpenApiLicense

{

Name = "Example License",

Url = new Uri("https://example.com/license")

}

});

});

var app = builder.Build();

// Configure the HTTP request pipeline.

if (app.Environment.IsDevelopment())

{

app.UseSwagger();

app.UseSwaggerUI();

}

if (builder.Environment.IsDevelopment())

{

app.UseSwaggerUI(options => // UseSwaggerUI is called only in Development.

{

options.SwaggerEndpoint("/swagger/v1/swagger.json", "v1");

options.RoutePrefix = string.Empty;

});

}

app.UseSwagger(options =>

{

options.SerializeAsV2 = true;

});

app.UseHttpsRedirection();

app.UseAuthorization();

app.UseCors(builder => builder

.AllowAnyOrigin()

.AllowAnyMethod()

.AllowAnyHeader());

app.MapControllers();

app.Run();

# Класс Busines

//Файл для бизнес процессов

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//\*\*\*\*\*\*\*\*\*\*\*

//\*\*\*\*\*\*\*

//\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

//using System.Windows.Forms;

using System.Data.SqlClient;

using static System.Net.Mime.MediaTypeNames;

using System.Threading.Tasks;

using static System.Reflection.Metadata.BlobBuilder;

using System.Net;

public class Visit

{

public Book\_short book { get; set; }

public DateTime date { get; set; }

public Visit(Book\_short book, DateTime date)

{

this.book = book; this.date=date;

}

}

public class User

{

public int id { get; set; }

public string name { get; set; }

public List<Visit> visits { get; set; }

public User(int id, string name, List<Visit> visit)

{

this.id = id;

this.name = name; this.visits = visit;

}

}

public class Tag

{

public int id { get; set; }

public string name { get; set; }

public Tag(int id, string name)

{

this.id = id;

this.name = name;

}

}

/// <summary>

/// Книга

/// </summary>

public class Book

{

/// <summary>

/// Книга

/// </summary>

[System.ComponentModel.DataAnnotations.Required]

public int id { get; set; }

/// <summary>

/// Книга

/// </summary>

[System.ComponentModel.DataAnnotations.Required]

public string name { get; set; }

/// <summary>

/// Книга

/// </summary>

[System.ComponentModel.DataAnnotations.Required]

public List<Author\_short> Authors { get; set; }

/// <summary>

/// Книга

/// </summary>

[System.ComponentModel.DataAnnotations.Required]

public List<Tag> Tags { get; set; }

/// <summary>

/// Книга

/// </summary>

[System.ComponentModel.DataAnnotations.Required]

public string description { get; set; }

/// <summary>

/// Книга

/// </summary>

public string Location { get; set; }

public Book(int id, string name, List<Author\_short> authors, List<Tag> tags, string description,string location)

{

this.id = id;

this.name = name;

Authors = authors;

Tags = tags;

this.description = description;

this.Location = location;

}

}

public class Book\_short

{

public int id { get; set; }

public string name { get; set; }

public Book\_short(int id, string name)

{

this.id = id;

this.name = name;

}

}

public class Author

{

public int id { get; set; }

public string name { get; set; }

public List<Book\_short> Books { get; set; }

public string description { get; set; }

public Author(int id, string name, List<Book\_short> Books, string description)

{

this.id = id;

this.name = name;

this.Books = Books;

this.description = description;

}

}

public class Author\_short

{

public int id { get; set; }

public string name { get; set; }

public Author\_short(int id, string name)

{

this.id = id;

this.name = name;

}

}

public class Business

{

private string connectionString = @"Data Source=.\ANDREU;Initial Catalog=Books;Integrated Security=false;User Id=AUE; Password =777;";

public User? checkUsr(string name, string pwd)

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

string commandString = "SELECT id FROM Users Where name='"+name+"' AND password='"+pwd+"';";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

int n = ds\_DataSet\_Edit.Tables[0].Rows.Count;

if(n==0)

return null;

int id=ds\_DataSet\_Edit.Tables[0].Rows[0].Field<int>("id");

List<Visit> visits = History(id);

return new User(id, name, visits);

}

public List<Visit> History(int UsrId)

{

int id = UsrId;

int n = countRows("History", "UserId", id.ToString());

string commandString = "SELECT BookID, Date FROM History WHERE UserId = '" + id.ToString() + "';";// создаем запрос

List<int> BookId = new List<int>();

List<DateTime> Dates = new List<DateTime>();

DataSet ds\_DataSet\_Edit = new DataSet();

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

for (int i = 0; i < n; i++)

{

BookId.Add(ds\_DataSet\_Edit.Tables[0].Rows[i].Field<int>("BookID"));

Dates.Add((ds\_DataSet\_Edit.Tables[0].Rows[i].Field<DateTime>("Date")));

}

ds\_DataSet\_Edit = new DataSet();

List<Book\_short> books = new List<Book\_short>();

for (int i = 0; i < n; i++)

{

ds\_DataSet\_Edit = new DataSet();

commandString = "SELECT Name FROM Books WHERE id = '" + BookId[i] + "';";// создаем запрос

Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

books.Add(new Book\_short(BookId[i], ds\_DataSet\_Edit.Tables[0].Rows[0].Field<string>("Name")));

}

List<Visit> vis=new List<Visit>();

for (int i = 0; i < n; i++)

{

vis.Add(new Visit(books[i], Dates[i]));

}

return vis;

}

public int countRows(string table, string row, string token)

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

//string connectionString = @"Data Source=.\ANDREU;Initial Catalog=Books;Integrated Security=false;User Id=AUE; Password =777;";

string commandString = "SELECT 1 FROM " + table + " WHERE " + row + " = '" + token + "';";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

//int rows\_ = 0;

int rows\_ = ds\_DataSet\_Edit.Tables[0].Rows.Count;

return rows\_;

}

public List<Author\_short> AllAuthors()

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

string commandString = "SELECT id FROM Authors;";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

int n = ds\_DataSet\_Edit.Tables[0].Rows.Count;

List<int> books = new List<int>();

for (int i = 0; i < n; i++)

{

books.Add(ds\_DataSet\_Edit.Tables[0].Rows[i].Field<int>("id"));

}

List<Author\_short> authors = new List<Author\_short>();

foreach (int book in books)

{

authors.Add(Author\_short(book));

}

return authors;

}

public List<Tag> Tags()

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

string commandString = "SELECT id FROM Tags;";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

int n = ds\_DataSet\_Edit.Tables[0].Rows.Count;

List<int> books = new List<int>();

for (int i = 0; i < n; i++)

{

books.Add(ds\_DataSet\_Edit.Tables[0].Rows[i].Field<int>("id"));

}

List<Tag> authors = new List<Tag>();

foreach (int book in books)

{

authors.Add(Tag(book));

}

return authors;

}

public Author\_short Author\_short(int id)

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

//string connectionString = @"Data Source=.\ANDREU;Initial Catalog=Books;Integrated Security=false;User Id=AUE; Password =777;";

string commandString = "SELECT Name FROM Authors WHERE id = '" + id + "';";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

string name = ds\_DataSet\_Edit.Tables[0].Rows[0].Field<string>("Name");

return new Author\_short(id, name);

}

public Book\_short Book\_short(int id)

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

//string connectionString = @"Data Source=.\ANDREU;Initial Catalog=Books;Integrated Security=false;User Id=AUE; Password =777;";

string commandString = "SELECT Name FROM Books WHERE id = '" + id + "';";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

string name = ds\_DataSet\_Edit.Tables[0].Rows[0].Field<string>("Name");

return new Book\_short(id, name);

}

public Tag Tag(int id)

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

//string connectionString = @"Data Source=.\ANDREU;Initial Catalog=Books;Integrated Security=false;User Id=AUE; Password =777;";

string commandString = "SELECT Name FROM Tags WHERE id = '" + id + "';";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

string name = ds\_DataSet\_Edit.Tables[0].Rows[0].Field<string>("Name");

return new Tag(id, name);

}

public Author AuthorInf(int id)

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

//string connectionString = @"Data Source=.\ANDREU;Initial Catalog=Books;Integrated Security=false;User Id=AUE; Password =777;";

string commandString = "SELECT Name FROM Authors WHERE id = '" + id + "';";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

string name = ds\_DataSet\_Edit.Tables[0].Rows[0].Field<string>("Name");

ds\_DataSet\_Edit = new DataSet();

commandString = "SELECT Description FROM Authors WHERE id = '" + id + "';";// создаем запрос

Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

string descr = ds\_DataSet\_Edit.Tables[0].Rows[0].Field<string>("Description");

List<int> books = BooksOf(id);

List<Book\_short> books\_Shorts = new List<Book\_short>();

foreach (int book in books)

{

books\_Shorts.Add(Book\_short(book));

}

return new Author(id, name, books\_Shorts, descr);

}

public Book BookInf(int id)

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

//string connectionString = @"Data Source=.\ANDREU;Initial Catalog=Books;Integrated Security=false;User Id=AUE; Password =777;";

string commandString = "SELECT Name FROM Books WHERE id = '" + id + "';";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

string name = ds\_DataSet\_Edit.Tables[0].Rows[0].Field<string>("Name");

ds\_DataSet\_Edit = new DataSet();

commandString = "SELECT Description FROM Books WHERE id = '" + id + "';";// создаем запрос

Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

string descr = ds\_DataSet\_Edit.Tables[0].Rows[0].Field<string>("Description");

ds\_DataSet\_Edit = new DataSet();

commandString = "SELECT Location FROM Books WHERE id = '" + id + "';";// создаем запрос

Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

string loc = ds\_DataSet\_Edit.Tables[0].Rows[0].Field<string>("Location");

List<int> auth = GetAuthors(id);

List<Author\_short> books\_Shorts = new List<Author\_short>();

foreach (int book in auth)

{

books\_Shorts.Add(Author\_short(book));

}

List<int> tags = GetTags(id);

List<Tag> tag = new List<Tag>();

foreach (int book in tags)

{

tag.Add(Tag(book));

}

return new Book(id, name, books\_Shorts, tag, descr,loc);

}

public List<int> AllTags()

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

string commandString = "SELECT id FROM Tags;";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

int n = ds\_DataSet\_Edit.Tables[0].Rows.Count;

List<int> books = new List<int>();

for (int i = 0; i < n; i++)

{

books.Add(ds\_DataSet\_Edit.Tables[0].Rows[i].Field<int>("id"));

}

return books;

}

public List<Book> Catalog()

{

List<int> ints = new List<int>();

ints = AllBooks();

List<Book> books = new List<Book>();

foreach (int book in ints)

{

books.Add(BookInf(book));

}

return books;

}

public List<int> AllBooks()

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

string commandString = "SELECT id FROM Books;";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

int n = ds\_DataSet\_Edit.Tables[0].Rows.Count;

List<int> books = new List<int>();

for (int i = 0; i < n; i++)

{

books.Add(ds\_DataSet\_Edit.Tables[0].Rows[i].Field<int>("id"));

}

return books;

}

public List<int> GetTags(int id)

{

int n = countRows("BookTag", "BookID", id.ToString());

string commandString = "SELECT TagID FROM BookTag WHERE BookID = '" + id.ToString() + "';";// создаем запрос

List<int> authorID = new List<int>();

DataSet ds\_DataSet\_Edit = new DataSet();

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

for (int i = 0; i < n; i++)

{

authorID.Add(ds\_DataSet\_Edit.Tables[0].Rows[i].Field<int>("TagId"));

}

return authorID;

}

public List<int> GetAuthors(int id)

{

int n = countRows("BookAuthor", "BookID", id.ToString());

string commandString = "SELECT AuthorID FROM BookAuthor WHERE BookID = '" + id.ToString() + "';";// создаем запрос

List<int> authorID = new List<int>();

DataSet ds\_DataSet\_Edit = new DataSet();

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

for (int i = 0; i < n; i++)

{

authorID.Add(ds\_DataSet\_Edit.Tables[0].Rows[i].Field<int>("AuthorId"));

}

return authorID;

}

public List<int> BooksOf(int id)

{

int n = countRows("BookAuthor", "AuthorId", id.ToString());

string commandString = "SELECT BookID FROM BookAuthor WHERE AuthorId = '" + id.ToString() + "';";// создаем запрос

List<int> authorID = new List<int>();

DataSet ds\_DataSet\_Edit = new DataSet();

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

for (int i = 0; i < n; i++)

{

authorID.Add(ds\_DataSet\_Edit.Tables[0].Rows[i].Field<int>("BookID"));

}

return authorID;

}

public string GetPwd(string Name)

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

string commandString = "SELECT Password FROM Users WHERE Name = '" + Name + "';";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

string pwd = ds\_DataSet\_Edit.Tables[0].Rows[0].Field<string>("Password");

return pwd;

}

public async Task BookUser(int UserId, int BookId)

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

string sqlExpression = "INSERT INTO History (BookID, UserID, Date) VALUES (" + BookId + ", " + UserId + ", '" + curDateTime() + "')";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

int number = command.ExecuteNonQuery();

//Console.WriteLine("Добавлено объектов: {0}", number);

}

//Console.Read();

}

bool check(string table, string row, string token)

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

string connectionString = @"Data Source=.\ANDREU;Initial Catalog=Books;Integrated Security=false;User Id=AUE; Password =777;";

string commandString = "SELECT " + row + " FROM " + table + " WHERE " + row + " = '" + token + "';";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

//int rows\_ = 0;

int rows\_ = ds\_DataSet\_Edit.Tables[0].Rows.Count;

if (rows\_ == 0)

{ return false; }

else { return true; };

}

bool check2(string table, string row1, string token, string row2, string token2)

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

string connectionString = @"Data Source=.\ANDREU;Initial Catalog=Books;Integrated Security=false;User Id=AUE; Password =777;";

string commandString = "SELECT " + row1 + " FROM " + table + " WHERE " + row1 + " = '" + token + "' AND " +row2 + " = '" + token2+ "';";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

//int rows\_ = 0;

int rows\_ = ds\_DataSet\_Edit.Tables[0].Rows.Count;

if (rows\_ == 0)

{ return false; }

else { return true; };

}

public async Task AddBook(string Name, string Description)

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

string commandString = "SELECT id FROM Books;";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

List<int> ids = new List<int>();

int n = ds\_DataSet\_Edit.Tables[0].Rows.Count;

for (int i = 0; i < n; i++)

{

ids.Add(ds\_DataSet\_Edit.Tables[0].Rows[i].Field<int>("id"));

}

int newId;

bool flag = false;

int lastid = ids[0];

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

{

if (ids[i] > lastid + 1)

{

flag = true;

newId = lastid;

break;

}

lastid = ids[i];

}

if (!flag) { newId = n; }

string sqlExpression = "INSERT INTO Books (id, Name, Description, Location) VALUES (" + (lastid + 1).ToString() + ", '" + Name + "', '" + Description + "',NULL)";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

int number = command.ExecuteNonQuery();

//Console.WriteLine("Добавлено объектов: {0}", number);

}

// Console.Read();

}

public async Task AddTag(string Name)

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

string commandString = "SELECT id FROM Tags;";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

List<int> ids = new List<int>();

int n = ds\_DataSet\_Edit.Tables[0].Rows.Count;

for (int i = 0; i < n; i++)

{

ids.Add(ds\_DataSet\_Edit.Tables[0].Rows[i].Field<int>("id"));

}

int newId;

bool flag = false;

int lastid = ids[0];

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

{

if (ids[i] > lastid + 1)

{

flag = true;

newId = lastid;

break;

}

lastid = ids[i];

}

if (!flag) { newId = n; }

string sqlExpression = "INSERT INTO Tags (id, Name) VALUES (" + (lastid + 1).ToString() + ", '" + Name + "')";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

int number = command.ExecuteNonQuery();

//Console.WriteLine("Добавлено объектов: {0}", number);

}

// Console.Read();

}

public async Task AddAuthor(string Name, string Description)

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

string commandString = "SELECT id FROM Authors;";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

List<int> ids = new List<int>();

int n = ds\_DataSet\_Edit.Tables[0].Rows.Count;

for (int i = 0; i < n; i++)

{

ids.Add(ds\_DataSet\_Edit.Tables[0].Rows[i].Field<int>("id"));

}

int newId;

bool flag = false;

int lastid = ids[0];

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

{

if (ids[i] > lastid + 1)

{

flag = true;

newId = lastid;

break;

}

lastid = ids[i];

}

if (!flag) { newId = n; }

string sqlExpression = "INSERT INTO Authors (id, Name, Description) VALUES (" + (lastid + 1).ToString() + ", '" + Name + "', '" + Description + "')"; ;

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

int number = command.ExecuteNonQuery();

//Console.WriteLine("Добавлено объектов: {0}", number);

}

// Console.Read();

}

public async Task BookAuthor(int BookId, string author)

{

int n=countRows("Authors","Name",author);

if(n==0)

{

AddAuthor(author,"");

}

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

//string connectionString = @"Data Source=.\ANDREU;Initial Catalog=Books;Integrated Security=false;User Id=AUE; Password =777;";

string commandString = "SELECT id FROM Authors WHERE Name = '" + author + "';";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

int AuthId = ds\_DataSet\_Edit.Tables[0].Rows[0].Field<int>("id");

if(!check2("BookAuthor", "AuthorId", AuthId.ToString(),"BookId",BookId.ToString()))

{

ds\_DataSet\_Edit = new DataSet();

ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

string sqlExpression = "INSERT INTO BookAuthor (BookID, AuthorID) VALUES (" + BookId + ", " + AuthId + ")";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

int number = command.ExecuteNonQuery();

//Console.WriteLine("Добавлено объектов: {0}", number);

}

}

//Console.Read();

}

public async Task BookTag(int bookId, string tag)

{

int n = countRows("Tags", "Name", tag);

if (n == 0)

{

AddTag(tag);

}

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

//string connectionString = @"Data Source=.\ANDREU;Initial Catalog=Books;Integrated Security=false;User Id=AUE; Password =777;";

string commandString = "SELECT id FROM Tags WHERE Name = '" + tag + "';";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

int TagId = ds\_DataSet\_Edit.Tables[0].Rows[0].Field<int>("id");

if (!(check2("BookTag", "TagId", TagId.ToString(),"BookId",bookId.ToString())))

{

ds\_DataSet\_Edit = new DataSet();

ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

string sqlExpression = "INSERT INTO BookTag (BookID, TagID) VALUES (" + bookId + ", " + TagId + ")";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

int number = command.ExecuteNonQuery();

//Console.WriteLine("Добавлено объектов: {0}", number);

}

}

}

public void DeliteBook(int bookId)

{

string sqlExpression = "DELETE FROM BookAuthor WHERE BookId=" + bookId;

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

int number = command.ExecuteNonQuery();

//Console.WriteLine("Удалено объектов: {0}", number);

}

sqlExpression = "DELETE FROM BookTag WHERE BookId=" + bookId;

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

int number = command.ExecuteNonQuery();

//Console.WriteLine("Удалено объектов: {0}", number);

}

sqlExpression = "DELETE FROM Books WHERE id=" + bookId;

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

int number = command.ExecuteNonQuery();

//Console.WriteLine("Удалено объектов: {0}", number);

}

}

public void DeliteBookAuthor(int id, string author)

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

//string connectionString = @"Data Source=.\ANDREU;Initial Catalog=Books;Integrated Security=false;User Id=AUE; Password =777;";

string commandString = "SELECT id FROM Authors WHERE Name = '" + author + "';";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

int AuthId = ds\_DataSet\_Edit.Tables[0].Rows[0].Field<int>("id");

string sqlExpression = "DELETE FROM BookAuthor WHERE BookId=" + id + " AND AuthorId=" + AuthId;

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

int number = command.ExecuteNonQuery();

//Console.WriteLine("Удалено объектов: {0}", number);

}

}

public void renameBook(string book, string newName)

{

string sqlExpression = "UPDATE Books SET Name='" + newName + "' WHERE Name='" + book + "'";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

int number = command.ExecuteNonQuery();

//Console.WriteLine("Обновлено объектов: {0}", number);

}

}

public void AddLok(int id, string newLoc)

{

string sqlExpression = "UPDATE Books SET Location='" + newLoc + "' WHERE id='" + id + "'";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

int number = command.ExecuteNonQuery();

//Console.WriteLine("Обновлено объектов: {0}", number);

}

}

//public Book AllAbout(string book)

//{

// List<string> Authors = this.AllAuthors();

// List<string> Tags = this.AllTags();

// return new Book(book, Authors, Tags);

//}

public string curDateTime()

{

DateTime myDateTime = DateTime.Now;

return myDateTime.ToString("yyyy-MM-dd HH:mm:ss.fff");

}

public List<int> Tagged(int id)

{

int n = countRows("BookTag", "TagId", id.ToString());

string commandString = "SELECT BookID FROM BookTag WHERE TagID = '" + id.ToString() + "';";// создаем запрос

List<int> authorID = new List<int>();

DataSet ds\_DataSet\_Edit = new DataSet();

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

for (int i = 0; i < n; i++)

{

authorID.Add(ds\_DataSet\_Edit.Tables[0].Rows[i].Field<int>("BookId"));

}

return authorID;

}

public async Task AddUser(string Name, string PWD)

{

DataSet ds\_DataSet\_Edit = new DataSet();// создаем объект DataSet

string commandString = "SELECT id FROM Users;";// создаем запрос

SqlDataAdapter Adapter\_EDIT = new SqlDataAdapter(commandString, connectionString);// создаем SqlDataAdapter adapter

Adapter\_EDIT.Fill(ds\_DataSet\_Edit); // заполнение DataSet данными с помощью DataAdapter

List<int> ids = new List<int>();

int n = ds\_DataSet\_Edit.Tables[0].Rows.Count;

for (int i = 0; i < n; i++)

{

ids.Add(ds\_DataSet\_Edit.Tables[0].Rows[i].Field<int>("id"));

}

int newId;

bool flag = false;

int lastid = ids[0];

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

{

if (ids[i] > lastid + 1)

{

flag = true;

newId = lastid;

break;

}

lastid = ids[i];

}

if (!flag) { newId = n; }

string sqlExpression = "INSERT INTO Users (id, Name, Password) VALUES (" + (lastid + 1).ToString() + ", '" + Name + "', '" + PWD + "')";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(sqlExpression, connection);

int number = command.ExecuteNonQuery();

//Console.WriteLine("Добавлено объектов: {0}", number);

}

// Console.Read();

}

}