Лабораторная работа №13
Разработка web-API для доступа к данным

1 **Цель работы**

1.1 Научиться выполнять разработку web-API для доступа к БД.

2Литература

2.1 ASP.NET Core MVC | Полное руководство. metanit.com – Текст : электронный // metanit.com, 2025. – URL: https://metanit.com/sharp/aspnet6/ – гл.11-12.

3Подготовка к работе

3.1 Повторить теоретический материал (см.п.2).

3.2 Изучить описание практической работы.

4Основное оборудование

4.1 Персональный компьютер.

5Задание

5.1 Создание структуры проектов

5.1.1 Создать проект «Веб-API ASP.NET Core (Майкрософт)». В окне «Дополнительные сведения» выбрать платформу .NET 9 и поставить флажки «Включить поддержку OpenAPI» и «Использовать контроллеры». Добавить в проект поддержку Scalar для тестирования web-API.

5.1.2 Реализовать запуск браузера и открытие страницы со Scalar при старте приложения.

5.1.3 Создать проект «Библиотека классов (Майкрософт)» выбрать платформу .NET 9. Добавить в проект пакеты для работы с БД, создать контекст данных и модели данных для таблиц предметной области «Кинотеатр».

Чтобы не загружать данные из модели по навигационным свойствам и необязательные данные, над свойствами в модели можно поставить аннотацию [JsonIgnore], например, над постером у фильма и над списком фильмов в жанрах.

5.1.4 Добавить в приложение веб-API ссылку на проект библиотеки классов и строку:

builder.Services.AddDbContext<ИмяКонтекста>();

5.1.5 Сгенерировать контроллер для класса Фильм.

5.2 Создание GET-методов с параметрами запроса

Добавить в приложение метод:

- /films/pages?page={page}&sortBy={sortBy} для постраничного вывода информации о фильмах (размер страницы 3). Включить в метод возможность сортировки по названию, году выхода по возрастанию и году выхода по убыванию,
- /films/filter?year={year}&title={title} для получения списка фильмов указанного года и/или содержащих указанный текст в названии.

Проверить работу созданных методов в браузере.

5.3 Создание GET-методов с параметрами пути

Добавить в приложение метод:

- /films/{id}/genres для получения списка жанров фильма с указанным в пути id,
- /films/{id}/sessions для получения списка будущих сеансов фильма с указанным в пути id.

Проверить работу созданных методов в браузере.

5.4 Создание GET-методов с составными параметрами запроса

Добавить в приложение метод /films/search

Добавить в созданный метод параметры:

- year={minYear}-{maxYear} для получения списка фильмов с годом из указанного диапазона (включительно), данные разделены дефисом,
- genres={genre1},{genre2},… для получения списка фильмов, у которых жанр входит в указанный в списке через запятую.

Проверить работу созданного метода браузере.

5.5 Создание GET-методов, возвращающих DTO

Добавить в приложение класс FilmDto {Id, Title, SalesProfit, TicketsCount}. SalesProfit – сумма, полученная за продажу билетов на фильм.

Добавить в приложение метод:

- /films/statistics для получения информации о количестве проданных билетов и прибыли, полученной за продажу билетов (данные вернуть в формате FilmDto).
- /films/statistics/{id} для получения информации о количестве проданных билетов и прибыли, полученной за продажу билетов с указанным в пути id (данные вернуть в формате FilmDto).

Проверить работу созданных методов в браузере.

6Порядок выполнения работы

6.1 Выполнить все задания из п.5.

6.2 Ответить на контрольные вопросы.

7Содержание отчета

7.1 Титульный лист

7.2 Цель работы

7.3 Ответы на контрольные вопросы

7.4 Вывод

8Контрольные вопросы

8.1 Что такое REST-запрос?

8.2 Что такое RESTful?

8.3 Для чего используется метод GET?

8.4 Для чего используется метод POST?

8.5 Для чего используется метод PUT? 8.6 Для чего используется метод DELETE?