МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ «ЛЭТИ»

ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра вычислительной техники

ОТЧЕТ

по лабораторной работе №4

по дисциплине «Программирование в среде .NET»

Тема: РАЗРАБОТКА ПРОГРАММНОГО ИНТЕРФЕЙСА ДЛЯ ВЗАИМОДЕЙСТВИЯ С ПРИЛОЖЕНИЕМ

Студент гр. 6306 _____ Жукаускайте А.В.

Преподаватель _____ Пешехонов К.А.

Санкт-Петербург

2020

Цель работы

Реализовать программный интерфейс для взаимодействия с приложением.

Задание

Реализовать WebAPI-слой для приложения (для проверки можно использовать SoapUI, Postman и т.д.)

Код программы

Примеры кода программы даны для сущности «сеанс», другие сущности («кинотеатр» и «фильм») реализованы похожим образом.

1) Пример объекта для передачи данных между слоями (ScreeningDTO)

```
public class ScreeningDTO
{
    public int Id { get; set; }
    public string Time { get; set; }
    public string Date { get; set; }
    public CinemaDTO Cinema { get; set; }
    public MovieDTO Movie{ get; set; }
}
```

2) Пример модели DTO для запроса (ScreeningCreateDTO)

```
public class ScreeningCreateDTO
{
    public int? MovieId{ get; set; }
    [Required(ErrorMessage = "Time is required")]
    public string Time { get; set; }
    [Required(ErrorMessage = "Date is required")]
    public string Date { get; set; }
    public int? CinemaId { get; set; }
}
```

3) Пример контроллера WebAPI (ScreeningController)

```
public class ScreeningController
{
    private ILogger<ScreeningController> Logger { get; }
    private IScreeningCreateService ScreeningCreateService { get; }
    private IScreeningGetService ScreeningGetService { get; }
    private IScreeningUpdateService ScreeningUpdateService { get; }
    private IMapper Mapper { get; }

    public ScreeningController(ILogger<ScreeningController> logger, IMapper mapper,
                               IScreeningCreateService screeningCreateService,
                               IScreeningGetService screeningGetService,
                               IScreeningUpdateService screeningUpdateService)
    {
        this.Logger = logger;
        this.ScreeningCreateService = screeningCreateService;
        this.ScreeningGetService = screeningGetService;
```

```
        this.ScreeningUpdateService = screeningUpdateService;
        this.Mapper = mapper;
    }

    [HttpPut]
    [Route("")]
    public async Task<ScreeningDTO> PutAsync(ScreeningCreateDTO screening)
    {
        this.Logger.LogTrace($"{nameof(this.PutAsync)} called");

        var result = await
this.ScreeningCreateService.CreateAsync(this.Mapper.Map<ScreeningUpdateModel>(screening));

        return this.Mapper.Map<ScreeningDTO>(result);
    }

    [HttpPatch]
    [Route("")]
    public async Task<ScreeningDTO> PatchAsync(ScreeningUpdateDTO screening)
    {
        this.Logger.LogTrace($"{nameof(this.PutAsync)} called");

        var result = await
this.ScreeningUpdateService.UpdateAsync(this.Mapper.Map<ScreeningUpdateModel>(screening));

        return this.Mapper.Map<ScreeningDTO>(result);
    }

    [HttpGet]
    [Route("")]
    public async Task<IEnumerable<ScreeningDTO>> GetAsync()
    {
        this.Logger.LogTrace($"{nameof(this.GetAsync)} called");

        return this.Mapper.Map<IEnumerable<ScreeningDTO>>(await
this.ScreeningGetService.GetAsync());
    }

    [HttpGet]
    [Route("{screeningId}")]
    public async Task<ScreeningDTO> GetAsync(int screeningId)
    {
        this.Logger.LogTrace($"{nameof(this.GetAsync)} called for {screeningId}");

        return this.Mapper.Map<ScreeningDTO>(await this.ScreeningGetService.GetAsync(new
ScreeningIdentityModel(screeningId)));
    }
}
```

Результаты работы приложения

Приложение тестировалось с помощью Postman. Результаты работы некоторых запросов представлены на рисунках ниже.

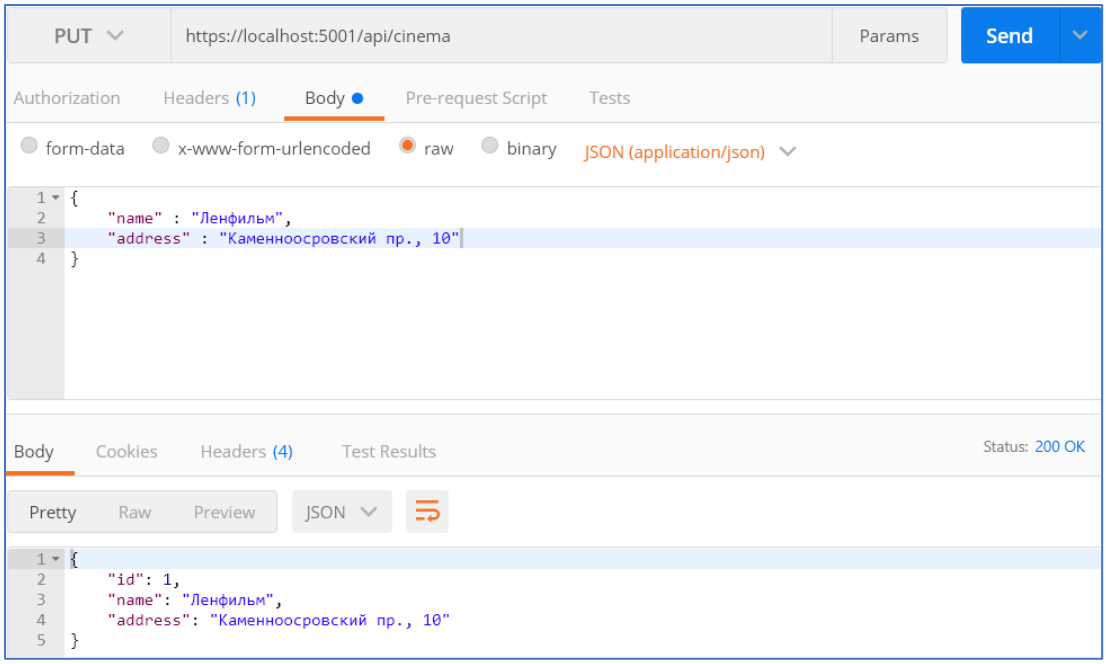

Рисунок 1. Put-запрос для добавления кинотеатра.

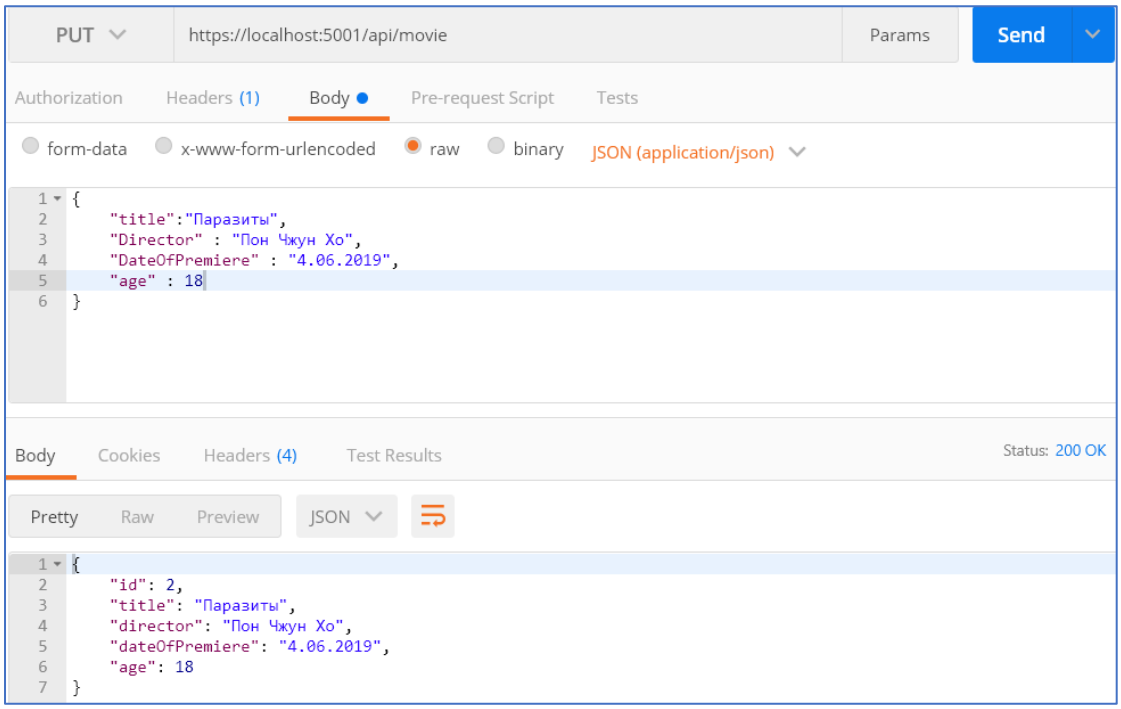

Рисунок 2. Put-запрос для добавления фильма.

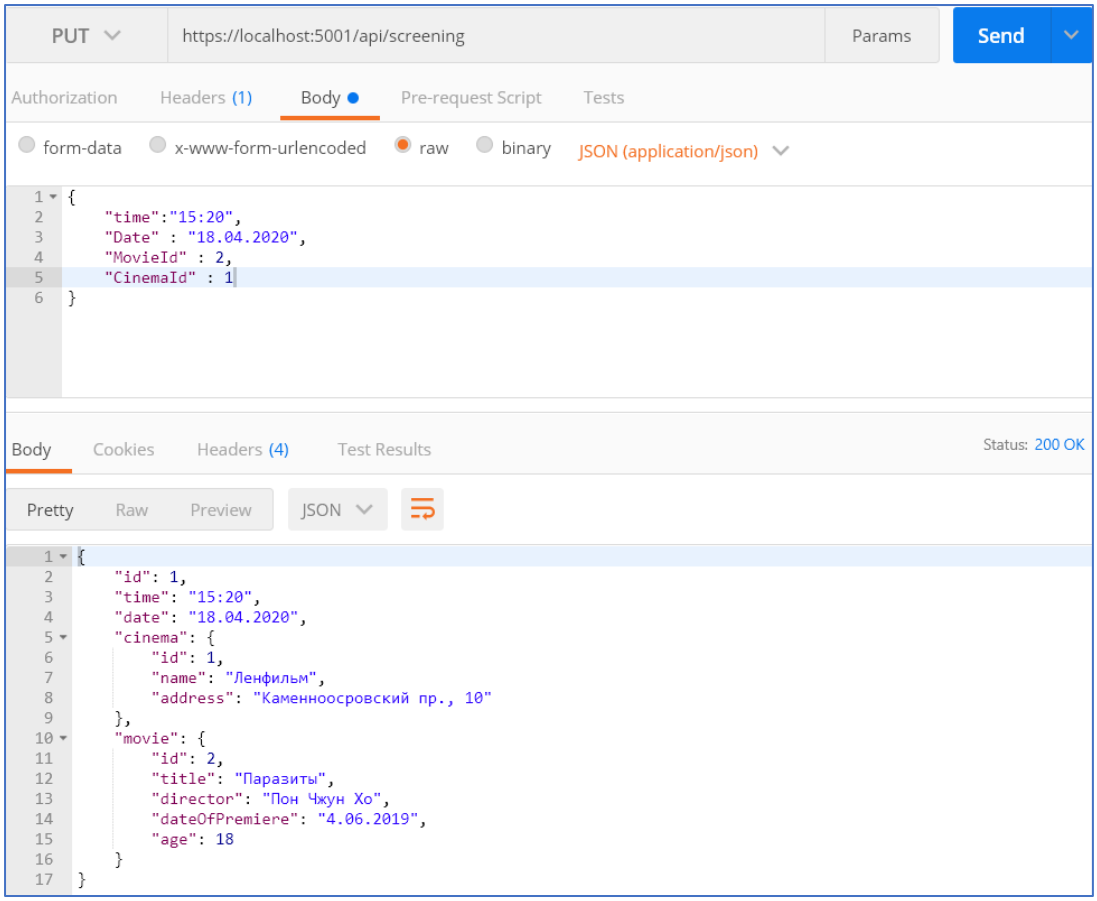

Рисунок 3. Put-запрос для добавления сеанса.

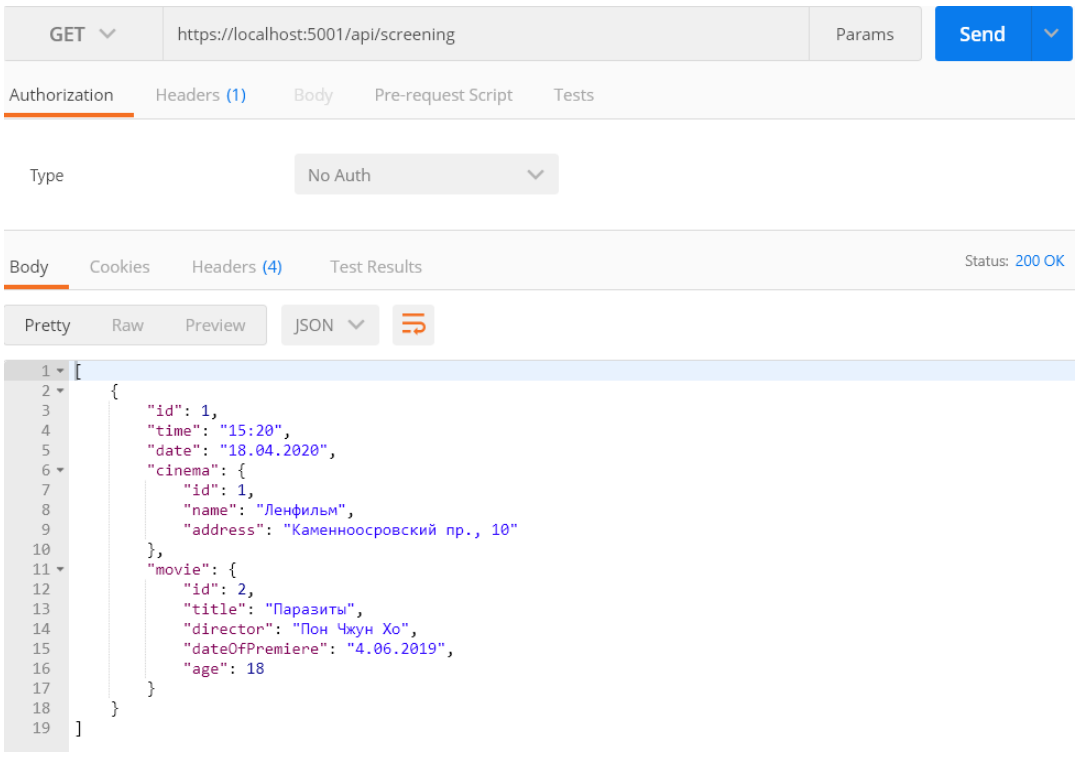

Рисунок 4. Get-запрос для получения списка сеансов

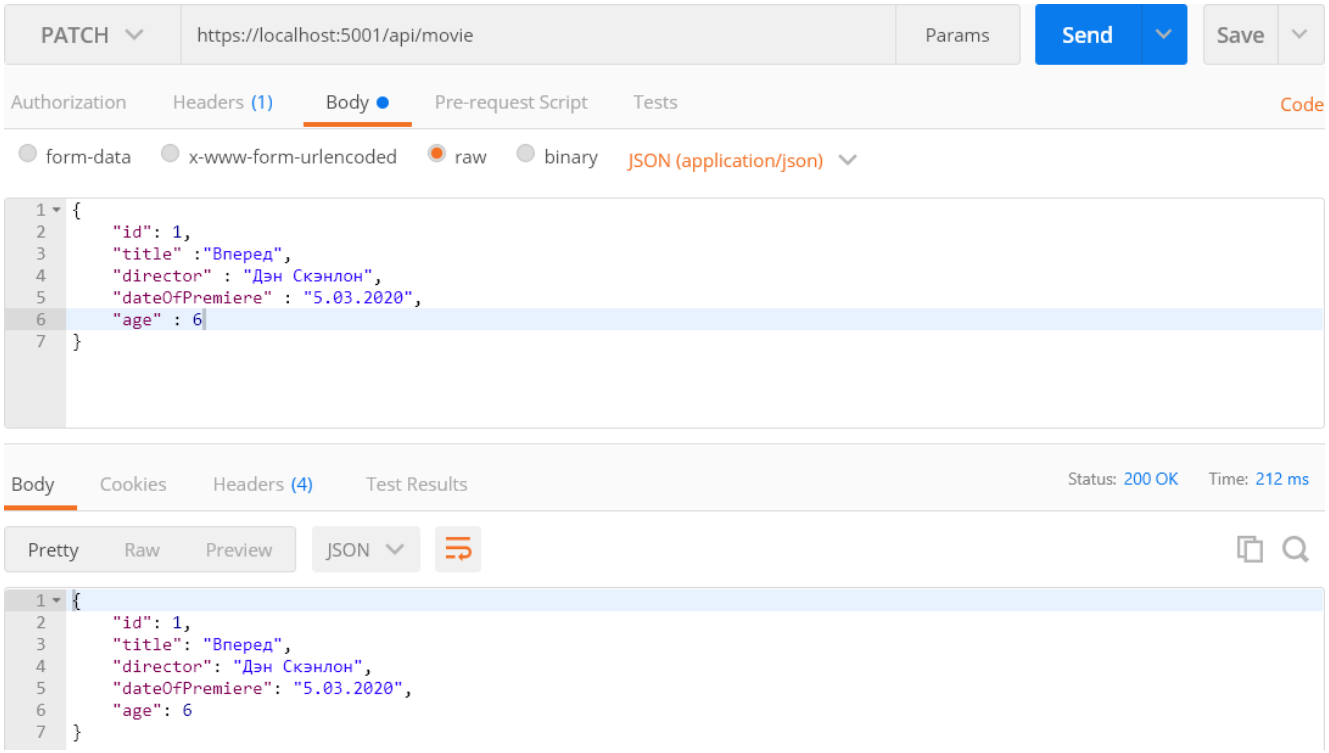

Рисунок 5. Patch-запрос для изменения данных о фильме с id 1

Выводы:

В процессе выполнения лабораторной работы был реализован слой WebAPI приложения в среде .NET. Были получены навыки по работе с WebAPI и реализации передачи данных с помощью DTO.