**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**



**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

***Факультет Информационных технологий***

***Кафедра Информатики и информационных технологий***

**направление подготовки**

**09.03.02 «Информационные системы и технологии»**

**ЛАБОРАТОРНАЯ РАБОТА №7**

**Дисциплина:** BackEnd-разработка

**Тема:** Логирование веб-приложение на основе ASP.NET Core

**Выполнил: студент группы 231-332**

**Чесноков Александр Евгеньевич**

(Фамилия И.О. обучающегося)

**Дата, подпись** 01.10.2025 

(Дата)(Подпись)

**Проверил:** \_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_*

(Фамилия И.О., степень, звание) (Оценка)

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *\_\_\_ \_\_\_\_\_\_\_\_*

(Дата)(Подпись)

**Москва**

**2025**

**Цель:**

Ознакомиться с методами логирования и их применением в веб-приложениях на платформе ASP.NET Core.

**Ход работы:**

Был создан новый проект ASP.NET Core с поддержкой контроллеров и минимальной конфигурацией. После этого в проекте была настроена система логирования, при этом стандартный метод AddFile() оказался недоступным, что потребовало использования стороннего пакета Serilog для записи логов в файл. Были установлены необходимые пакеты: Serilog.AspNetCore и Serilog.Sinks.File.

Далее в Program.cs была проведена конфигурация Serilog, включающая запись логов как в консоль, так и в файл с ежедневной ротацией. В процессе настройки был использован метод UseSerilog() для замены стандартного логгера ASP.NET Core. После этого была подключена конфигурация контроллеров, настроен Swagger для удобного тестирования.

Для проверки работы логирования были добавлены вызовы \_logger.LogInformation(), \_logger.LogWarning() и \_logger.LogError() в разных методах контроллера. После запуска приложения логи успешно отображались в консоли и одновременно записывались в файл Logs/log.txt, при этом проверялась корректность формирования сообщений и ротации файлов по дням.

В завершение работы были проведены тестовые вызовы контроллеров через Swagger, что позволило удостовериться, что все действия приложения корректно фиксируются в логах, и информация о выполнении запросов доступна для последующего анализа. Таким образом, конфигурация логирования была полностью интегрирована в проект, включая запись в файл и консоль, с поддержкой различных уровней логирования и ротации файлов.

**Скриншоты работы приложения:**

**Информационный лог:**В консоли:  


В файле log.txt:  


**Warning лог:**  
В консоли:



В файле log.txt:

****

**Error лог:**

В консоли:



В файле log.txt:

****

**Листинг:**

**Program.cs:**

|  |
| --- |
| using Serilog;  var builder = WebApplication.CreateBuilder(args);  // *Конфигурация Serilog*  Log.Logger = new LoggerConfiguration()      .WriteTo.Console() // *лог в консоль*      .WriteTo.File("Logs/log.txt", rollingInterval: RollingInterval.Day) // *лог в файл*      .CreateLogger();  builder.Host.UseSerilog(); // *Используем Serilog вместо стандартного логгера*  builder.Services.AddControllers();  builder.Services.AddEndpointsApiExplorer();  builder.Services.AddSwaggerGen();  var app = builder.Build();  if (app.Environment.IsDevelopment())  {      app.UseSwagger();      app.UseSwaggerUI();  }  app.UseAuthorization();  app.MapControllers();  app.Run(); |

**Controllers/LogDemoController.cs:**

|  |
| --- |
| using Microsoft.AspNetCore.Mvc;  namespace Lab07.Controllers;  [ApiController]  [Route("api/[controller]")]  public class LogDemoController : ControllerBase  {      private readonly ILogger<LogDemoController> \_logger;      public LogDemoController(ILogger<LogDemoController> logger)      {          \_logger = logger;      }      [HttpGet("info")]      public IActionResult InfoLog()      {          \_logger.LogInformation("Информационное сообщение для InfoLog");          return Ok("Информационный лог записан");      }      [HttpGet("warning")]      public IActionResult WarningLog()      {          \_logger.LogWarning("Предупреждение для WarningLog");          return Ok("Warning лог записан");      }      [HttpGet("error")]      public IActionResult ErrorLog()      {          \_logger.LogError("Ошибка для ErrorLog");          return Ok("Error лог записан");      }  } |

**appsettings.Development.json** (указаны уровни логирования)**:**

|  |
| --- |
| {    "Logging": {      "LogLevel": {        "Default": "Information",        "Microsoft.AspNetCore": "Warning"      }    }  } |

**Ссылка на репозиторий:** <https://github.com/avoryyy/backend_mospoly>