**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 №** 17

**Дисциплина: «Backend»**

**Тема:** *Обработка ошибок в веб-приложении на основе ASP.NET Core*

**Выполнил: студент группы: 231-339**

Карапетян Нвер Каренович\_\_\_\_\_\_\_\_\_\_\_

(Фамилия И.О.)



**Дата, подпись:** 07.05.25 \_\_\_\_\_\_\_\_\_\_\_\_

(Дата)(Подпись)

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

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

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

(Дата)(Подпись)

**Москва**

**2025**

Цель:

Освоить различные методы обработки ошибок в веб-приложениях на платформе ASP.NET Core для повышения устойчивости и безопасности приложений.

Задачи:

* Настроить обработку и отображение исключений, возникающих в приложении.
* Реализовать обработку ошибок для различных типов исключений (например, ошибки HTTP 404, исключения базы данных и др.).
* Создать пользовательские страницы для отображения ошибок для повышения понятности и информативности.
* Протестировать работу обработки ошибок при возникновении различных исключительных ситуаций.

Ход работы

Обработка ошибок — ключевой элемент надежности и безопасности веб-приложений. В ASP.NET Core предусмотрены механизмы для централизованного перехвата исключений, трансляции кодов состояния в дружелюбные ответы и возможности настройки пользовательских страниц ошибок.

Настройка обработки исключений

В файле **Program.cs** вместо чистого API подключена поддержка MVC-представлений методом AddControllersWithViews(). Это позволило возвращать Razor-страницы из контроллеров. Сразу после включения маршрутизации добавлены два middleware. Первый, UseExceptionHandler("/Error"), перехватывает все необработанные исключения в приложении и перенаправляет их на маршрут /Error. Второй, UseStatusCodePagesWithReExecute("/Error/{0}"), обрабатывает HTTP-коды состояния, такие как 404 и 403, перенаправляя их на /Error/404 или /Error/403 соответственно. При этом в коде устанавливаются корректные заголовки ответа, а сам конвейер продолжает управление дальше, передавая управление контроллерам.

Листинг 1. Фрагмент из Program.cs, отвечающий за настройку исключений.

builder.Services.AddControllersWithViews();

var app = builder.Build();

app.UseStaticFiles();

app.UseRouting();

app.UseExceptionHandler("/Error");

app.UseStatusCodePagesWithReExecute("/Error/{0}");

app.UseAuthorization();

app.MapControllerRoute(

    name: "default",

    pattern: "{controller=Home}/{action=Index}/{id?}"

);

app.Run();

Реализация контроллера ошибок

Контроллер **ErrorController**, унаследованный от базового Controller, содержит два метода. Первый метод Error обрабатывает маршрут /Error, получает через HttpContext.Features.Get<IExceptionHandlerPathFeature>() информацию об исключении, устанавливает статус ответа 500 и передает номер кода в ViewData. Второй метод StatusCode, помеченный маршрутом /Error/{code:int}, принимает целочисленный аргумент, устанавливает соответствующий статус ответа и также сохраняет код во ViewData. Оба метода возвращают представление StatusCode.cshtml.

Листинг 2. ErrorController.cs.

public class ErrorController : Controller

{

    [HttpGet("/Error")]

    public IActionResult HandleException()

    {

        var feature = HttpContext.Features.Get<IExceptionHandlerFeature>();

        Response.StatusCode = 500;

        ViewData["Code"] = 500;

        return View("StatusCode");

    }

    [HttpGet("/Error/{code:int}")]

    public IActionResult HandleStatusCode(int code)

    {

        Response.StatusCode = code;

        ViewData["Code"] = code;

        return View("StatusCode");

    }

}

Создание представления ошибок

Предварительно созданный файл **StatusCode.cshtml** отвечает за визуализацию ошибок. В нем выводится номер ошибки из ViewData["Code"].

Листинг 3. StatusCode.cshtml.

@{

    Layout = null;

    var code = ViewData["Code"];

}

<!DOCTYPE html>

<html>

<head>

    <meta charset="utf-8" />

    <title>Ошибка @code</title>

</head>

<body>

    <h1>Произошла ошибка @code</h1>

    <p>Извините за неудобства.</p>

    <a href="/">Вернуться на главную</a>

</body>

</html>

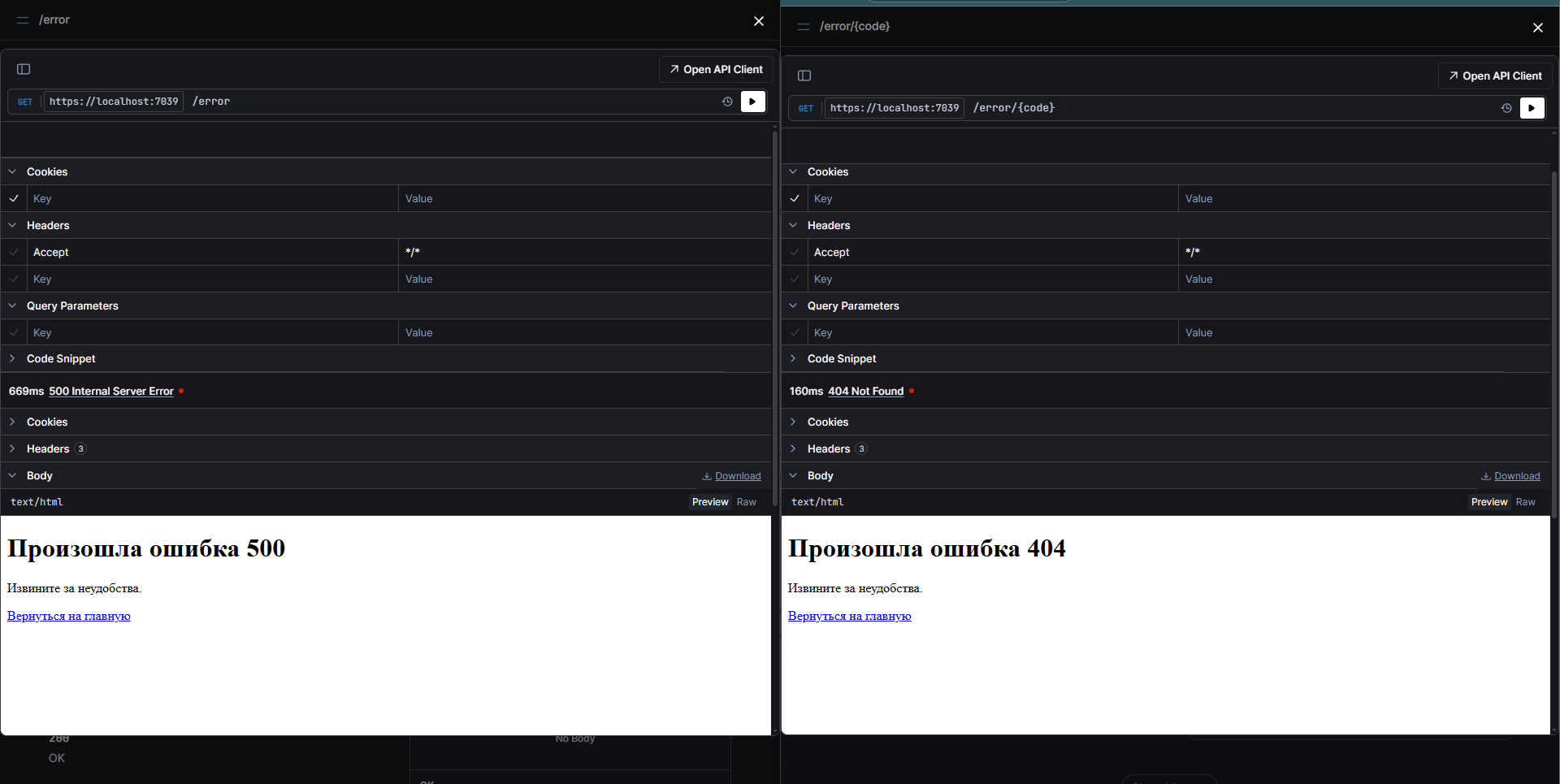


Рисунок . При запросах в браузере отображаются соответствующие страницы.