**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 № \_9\_**

**Дисциплина:** \_Backend разработка

**Тема:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Выполнил(а): студент(ка) группы \_\_231-336\_\_**

\_\_\_\_\_\_\_\_Канищев И.М\_\_\_\_\_\_\_\_\_\_\_\_

(Фамилия И.О.)

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

(Дата) (Подпись)

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

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

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

(Дата) (Подпись)

**Замечания:**

**Москва**

**2025**

## **1. Настройка обработки ошибок**

### ****1.1 Глобальная обработка ошибок****

В Program.cs добавляем middleware для обработки ошибок:

var builder = WebApplication.CreateBuilder(args);

builder.Services.AddControllersWithViews();

var app = builder.Build();

// Глобальная обработка ошибок

if (!app.Environment.IsDevelopment())

{

app.UseExceptionHandler("/Error"); // Редирект на пользовательскую страницу ошибок

app.UseHsts();

}

app.UseHttpsRedirection();

app.UseStaticFiles();

app.UseRouting();

app.UseAuthorization();

app.MapControllerRoute(

name: "default",

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

app.Run();

### ****1.2 Обработка HTTP-ошибок (404, 500)****

Создаем контроллер ErrorController:

using Microsoft.AspNetCore.Mvc;

public class ErrorController : Controller

{

[Route("Error")]

public IActionResult Error()

{

var feature = HttpContext.Features.Get<Microsoft.AspNetCore.Diagnostics.IExceptionHandlerPathFeature>();

ViewBag.Path = feature?.Path;

ViewBag.ErrorMessage = feature?.Error.Message;

return View("Error");

}

[Route("Error/404")]

public IActionResult NotFoundError()

{

return View("NotFound");

}

}

### ****1.3 Создание пользовательских страниц ошибок****

**Views/Error/Error.cshtml**:

@{

ViewData["Title"] = "Ошибка";

}

<h1>Произошла ошибка</h1>

<p>Путь: @ViewBag.Path</p>

<p>Сообщение: @ViewBag.ErrorMessage</p>

**Views/Error/NotFound.cshtml**:

@{

ViewData["Title"] = "Страница не найдена";

}

<h1>404 - Страница не найдена</h1>

<p>Запрашиваемая страница не существует.</p>

## **2. Обработка ошибок в контроллерах**

Пример контроллера, генерирующего исключение:

using Microsoft.AspNetCore.Mvc;

public class DemoController : Controller

{

public IActionResult Index()

{

throw new Exception("Тестовое исключение");

}

public IActionResult NotFoundTest()

{

return NotFound(); // Генерация ошибки 404

}

}

## **3. Тестирование обработки ошибок**

### ****3.1 Исключение в контроллере****

* Переходим на /Demo/Index

### В режиме development:

### В режиме production:

### 

### ****3.2 Ошибка 404****

### **При переходе на несуществующую страницу нас переносит на страницу ошибки**

### 

### ****3.3 Исключения базы данных****

* Аналогично можно обернуть вызовы к базе данных в try-catch и редиректить на страницу /Error.