**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:** Обработка ошибок в веб-приложении 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:**

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

**Ход работы:**

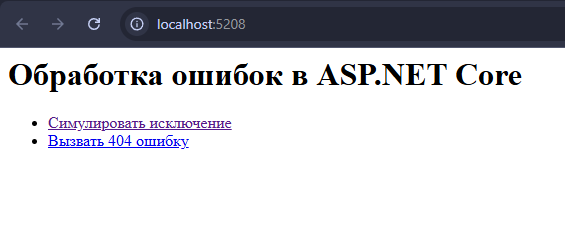
В первую очередь в файле Program.cs была настроена глобальная обработка исключений. Для режима разработки использовался middleware UseDeveloperExceptionPage(), который предоставляет подробные сведения о возникших ошибках, включая трассировку стека и контекст выполнения. Для режима производства была добавлена пользовательская страница обработки ошибок, доступная по маршруту /Home/Error, которая отображает пользователю понятное сообщение без раскрытия внутренней информации о приложении.

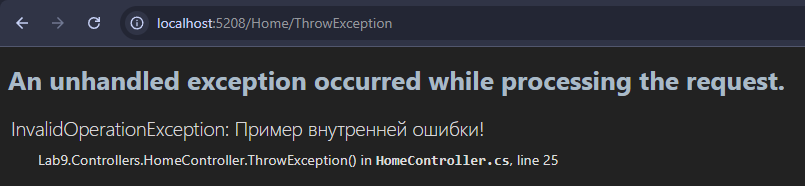
Помимо глобальной обработки исключений, была настроена обработка кодов статуса HTTP с помощью UseStatusCodePagesWithReExecute(). Этот middleware позволяет перенаправлять пользователя на специальный контроллер при возникновении ошибок, таких как 404 (страница не найдена) или 500 (внутренняя ошибка сервера). В контроллере HomeController был реализован метод StatusCodeHandler, который формирует пользовательские HTML-страницы для различных кодов ошибок.

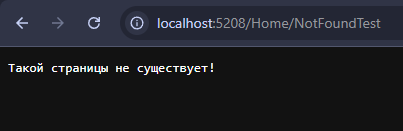
Также для демонстрации работы обработки ошибок были созданы два метода: ThrowException() — искусственно вызывает исключение InvalidOperationException, и NotFoundTest() — возвращает стандартный ответ NotFound(), генерируя ошибку 404. Это позволило проверить работу всех уровней обработки ошибок в приложении.

Пользовательская страница ошибок (Error) оформлена в простом HTML-стиле и содержит информативное сообщение, а также ссылку для возвращения на главную страницу. Для ошибок 404 и других кодов были реализованы отдельные страницы с сообщениями, соответствующими типу ошибки.

**Скриншоты работы приложения:**

****





**Листинг:**

**Program.cs:**

|  |
| --- |
| var builder = WebApplication.CreateBuilder(args);  builder.Services.AddControllersWithViews();  builder.Services.AddEndpointsApiExplorer();  builder.Services.AddSwaggerGen();  var app = builder.Build();  if (app.Environment.IsDevelopment())  {      // *В режиме разработки показываем детальные ошибки*      app.UseDeveloperExceptionPage();      app.UseSwagger();      app.UseSwaggerUI();  }  else  {      // *В продакшене используем пользовательскую страницу ошибок*      app.UseExceptionHandler("/Home/Error");      // *Обработка ошибок статуса, например 404*      app.UseStatusCodePagesWithReExecute("/Home/StatusCode", "?code={0}");  }  app.UseStaticFiles();  app.MapControllerRoute(      name: "default",      pattern: "{controller=Home}/{action=Index}/{id?}"  );  app.Run(); |

**Controllers/HomeController.cs:**

|  |
| --- |
| using Microsoft.AspNetCore.Mvc;  using System;  namespace Lab9.Controllers  {      public class HomeController : Controller      {          public IActionResult Index()          {              return Content(@"                  <html>                      <head><title>Практическое занятие №9</title></head>                      <body>                          <h1>Обработка ошибок в ASP.NET Core</h1>                          <ul>                              <li><a href='/Home/ThrowException'>Симулировать исключение</a></li>                              <li><a href='/Home/NotFoundTest'>Вызвать 404 ошибку</a></li>                          </ul>                      </body>                  </html>", "text/html; charset=utf-8");          }          public IActionResult ThrowException()          {              throw new InvalidOperationException("Пример внутренней ошибки!");          }          public IActionResult NotFoundTest()          {              return NotFound("Такой страницы не существует!");          }          [Route("Home/Error")]          public IActionResult Error()          {              var html = @"                  <html>                      <head><title>Ошибка</title></head>                      <body style='font-family:Arial;'>                          <h1 style='color:red;'>Произошла ошибка!</h1>                          <p>Что-то пошло не так. Пожалуйста, попробуйте позже.</p>                          <a href='/'>Вернуться на главную</a>                      </body>                  </html>";              return Content(html, "text/html; charset=utf-8");          }          [Route("Home/StatusCode")]          public IActionResult StatusCodeHandler(int code)          {              string message = code switch              {                  404 => "Страница не найдена (404)",                  500 => "Внутренняя ошибка сервера (500)",                  \_ => $"Ошибка: {code}"              };              var html = $@"                  <html>                      <head><title>Ошибка {code}</title></head>                      <body style='font-family:Arial;'>                          <h1>Ошибка {code}</h1>                          <p>{message}</p>                          <a href='/'>Вернуться на главную</a>                      </body>                  </html>";              return Content(html, "text/html; charset=utf-8");          }      }  } |

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