МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| к.т.н., профессор |  |  |  |  | С.Г.Фомичева |
| должность, уч. степень |  | подпись, дата |  |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №5 |
| **Разработка Web-клиента мониторинга исключений в распределенной информационной системе**  по дисциплине: РАСПРЕДЕЛЕННЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. | 3843 |  |  |  | А.П. Конева |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2021

1. **Цель работы**

Изучить принципы построения MVC-решений, позволяющих распределить бизнес-логику в распределенных информационных системах, научиться использовать методы проектирования приложений доступа к данным, базируясь на принципах Model-First. Освоить механизмы Entity Framework для проектирования Web-клиентов РИС.

2. **Задачи**

1) В соответствии с заявленной в лекциях функциональностью разработать Web-клиент, использующий ASP.NET Core MVC подходы разработки распределенных систем.

2)Web-клиент должен обеспечивать возможность удаленного мониторинга таблицы UserException разработанной ранее архитектуры БД, а также поддержку вызова CRUD-операций (create, read, update, delete) над данной таблицей.

4) Локализовать интерфейс Web-клиента

5) Разработать дополнительный функционал проекта в соответствии с индивидуальным вариантом 7:

Добавить MVC-контроллер, позволяющий на новой странице разработанного Web-клиента выполнять (CRUD – create, read, update, delete) операции с сущностью «AnalisReport» разработанной модели данных.

**3. Ход работы**

В процессе выполнения лабораторной работы с помощью использования механизмов Entity Framework для проектирования Web-клиентов был создан «Тонкий» ASP.NET Core клиент. Был использован подход Model-First, при котором сначала создается графическая модель данных EDMX, а затем создаются объекты в базе данных.

Использовали шаблон проекта «Веб-приложение ASP.Net Core (MVC)». Добавили необходимые для функционирования проекта пакеты NuGet, перечисленные на рис. 1.

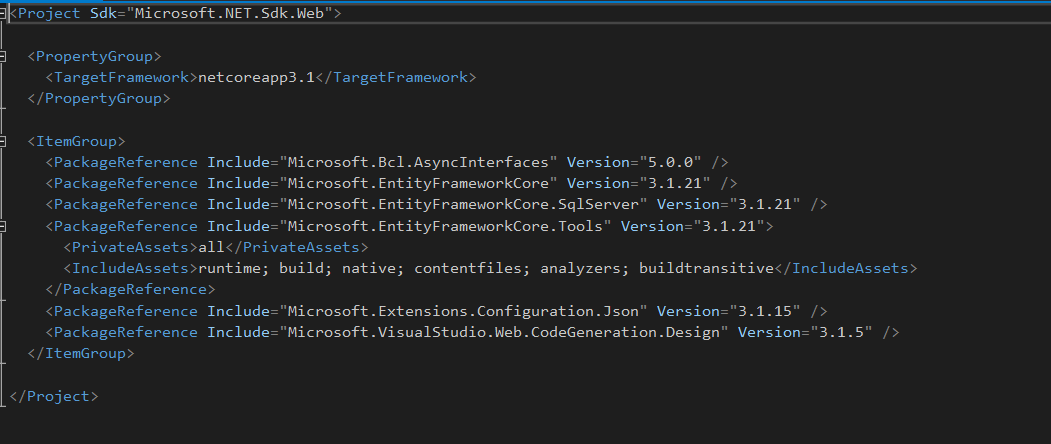


Рис.1 Установленные пакеты NuGet

Добавили в папку Models класс **UserException.cs**:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace WebASPTest.Models

{

[Table ("UserException")]

public class UserException

{

public int Id { get; set; }

public string Message { get; set; }

public string TargetSite { get; set; }

[DisplayFormat(DataFormatString = "{0:dd.MM.yyyy hh:mm:ss}", ApplyFormatInEditMode = true)]

public DateTime DateTimeExc { get; set; }

public string IndexForm { get; set; }

}

}

Туда же добавили класс **AnalisReport.cs**:

using System.Linq;

using System.Threading.Tasks;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace WebASPTest.Models

{

[Table("AnalisReport")]

public class AnalisReport

{

[Key]

public int id\_AnReport { get; set; }

public string titleAnReport { get; set; }

public string typeAnReport { get; set; }

public string contextAnReport { get; set; }

}

}

Создали папку Data. Добавили туда **MyDBContext.cs** со следующим кодом для создания класса контекста для БД:

using Microsoft.EntityFrameworkCore;

using WebASPTest.Models;

namespace MyDB.Data

{

public class MyDBContext : DbContext

{

public MyDBContext(DbContextOptions<MyDBContext> options)

: base(options)

{

}

public DbSet<UserException> UserException { get; set; }

public DbSet<AnalisReport> AnalisReport { get; set; }

}

}

Файл **appsettings.json** теперь выглядит следующим образом:

{

"Logging": {

"LogLevel": {

"Default": "Information",

"Microsoft": "Warning",

"Microsoft.Hosting.Lifetime": "Information"

}

},

"AllowedHosts": "\*",

"ConnectionStrings": {

"MyDBContext": "Server=LAPTOP-5EFL0IAT;Database=AirLogger;Trusted\_Connection=True;MultipleActiveResultSets=true"

}

}

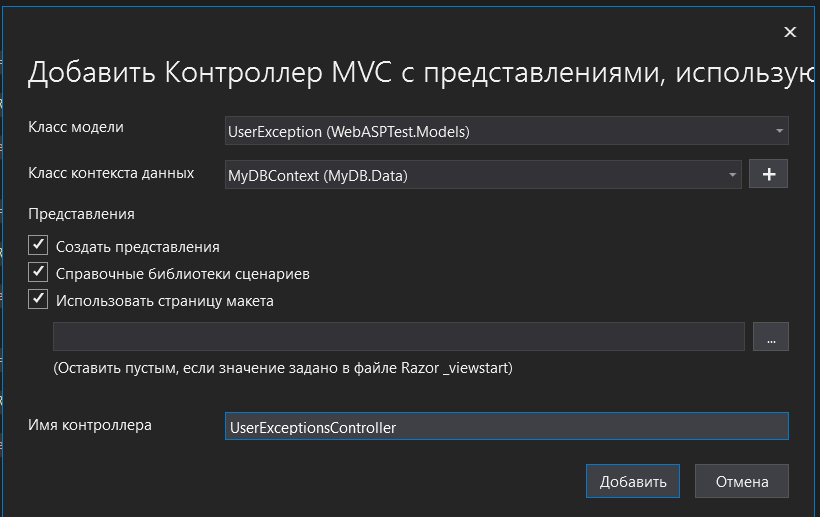
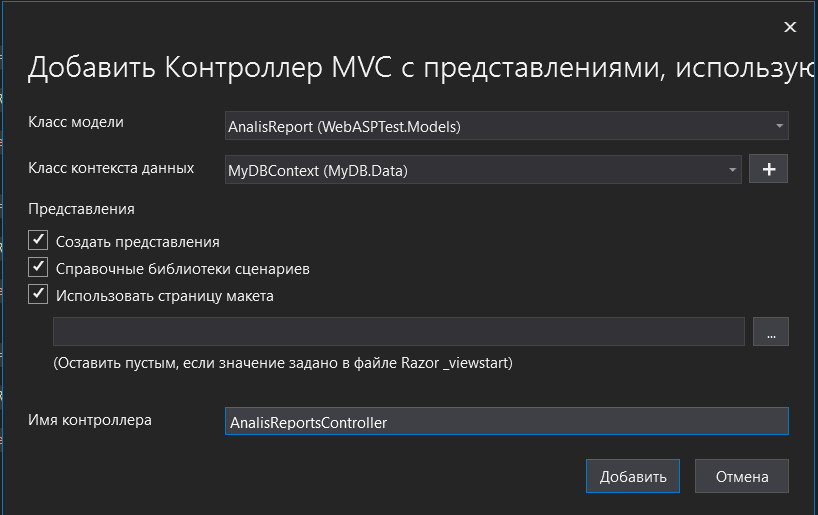
Создали Контроллер MVC с представлениями, использующий Entity Framework, для класса модели UserException (рис. 2), затем для класса модели AnalisReport (рис.3).

Рис.2 Контроллер MVC для UserException

Рис.3 Контроллер MVC для AnalisReport

В файле **Startup.cs** в карте маршрутизации указали имена контроллеров UserExceptions и AnalisReports :

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Hosting;

using Microsoft.AspNetCore.HttpsPolicy;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.Extensions.Hosting;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using MyDB.Data;

using Microsoft.EntityFrameworkCore;

namespace WebASPTest

{

public class Startup

{

public Startup(IConfiguration configuration)

{

Configuration = configuration;

}

public IConfiguration Configuration { get; }

// This method gets called by the runtime. Use this method to add services to the container.

public void ConfigureServices(IServiceCollection services)

{

services.AddControllersWithViews();

services.AddDbContext<MyDBContext>(options =>

options.UseSqlServer(Configuration.GetConnectionString("MyDBContext")));

}

// This method gets called by the runtime. Use this method to configure the HTTP request pipeline.

public void Configure(IApplicationBuilder app, IWebHostEnvironment env)

{

if (env.IsDevelopment())

{

app.UseDeveloperExceptionPage();

}

else

{

app.UseExceptionHandler("/Home/Error");

// The default HSTS value is 30 days. You may want to change this for production scenarios, see https://aka.ms/aspnetcore-hsts.

app.UseHsts();

}

app.UseHttpsRedirection();

app.UseStaticFiles();

app.UseRouting();

app.UseAuthorization();

app.UseEndpoints(endpoints =>

{

endpoints.MapControllerRoute(

name: "default",

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

});

app.UseEndpoints(endpoints =>

{

endpoints.MapControllerRoute(

name: "default",

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

});

app.UseEndpoints(endpoints =>

{

endpoints.MapControllerRoute(

name: "default",

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

});

}

}

}

Добавили в Views\Shared\\_Layout.cshtml кнопки «Пользовательские исключения» и «Аналитические отчеты» для перехода на страницу с пользовательскими исключениями и аналитическими отчетами соответственно:

<header>

<nav class="navbar navbar-expand-sm navbar-toggleable-sm navbar-light bg-white border-bottom box-shadow mb-3">

<div class="container">

<a class="navbar-brand" **asp-area**="" **asp-controller**="Home" **asp-action**="Index">Логгер исключений</a>

<button class="navbar-toggler" type="button" data-toggle="collapse" data-target=".navbar-collapse" aria-controls="navbarSupportedContent"

aria-expanded="false" aria-label="Toggle navigation">

<span class="navbar-toggler-icon"></span>

</button>

<div class="navbar-collapse collapse d-sm-inline-flex flex-sm-row-reverse">

<ul class="navbar-nav flex-grow-1">

<li class="nav-item">

<a class="nav-link text-dark" **asp-area**="" **asp-controller**="Home" **asp-action**="Index">Домой</a>

</li>

<li class="nav-item">

<a class="nav-link text-dark" **asp-area**="" **asp-controller**="Home" **asp-action**="Privacy">Политика конфиденциальности</a>

</li>

<li class="nav-item">

<a class="nav-link text-dark" **asp-area**="" **asp-controller**="UserExceptions" **asp-action**="Index">Пользовательские исключения</a>

</li>

<li class="nav-item">

<a class="nav-link text-dark" **asp-area**="" **asp-controller**="AnalisReports" **asp-action**="Index">Аналитические отчеты</a>

</li>

</ul>

</div>

</div>

</nav>

</header>

Локализовали интерфейс Web-клиента, что продемонстрировано на рис. 4 - 10.

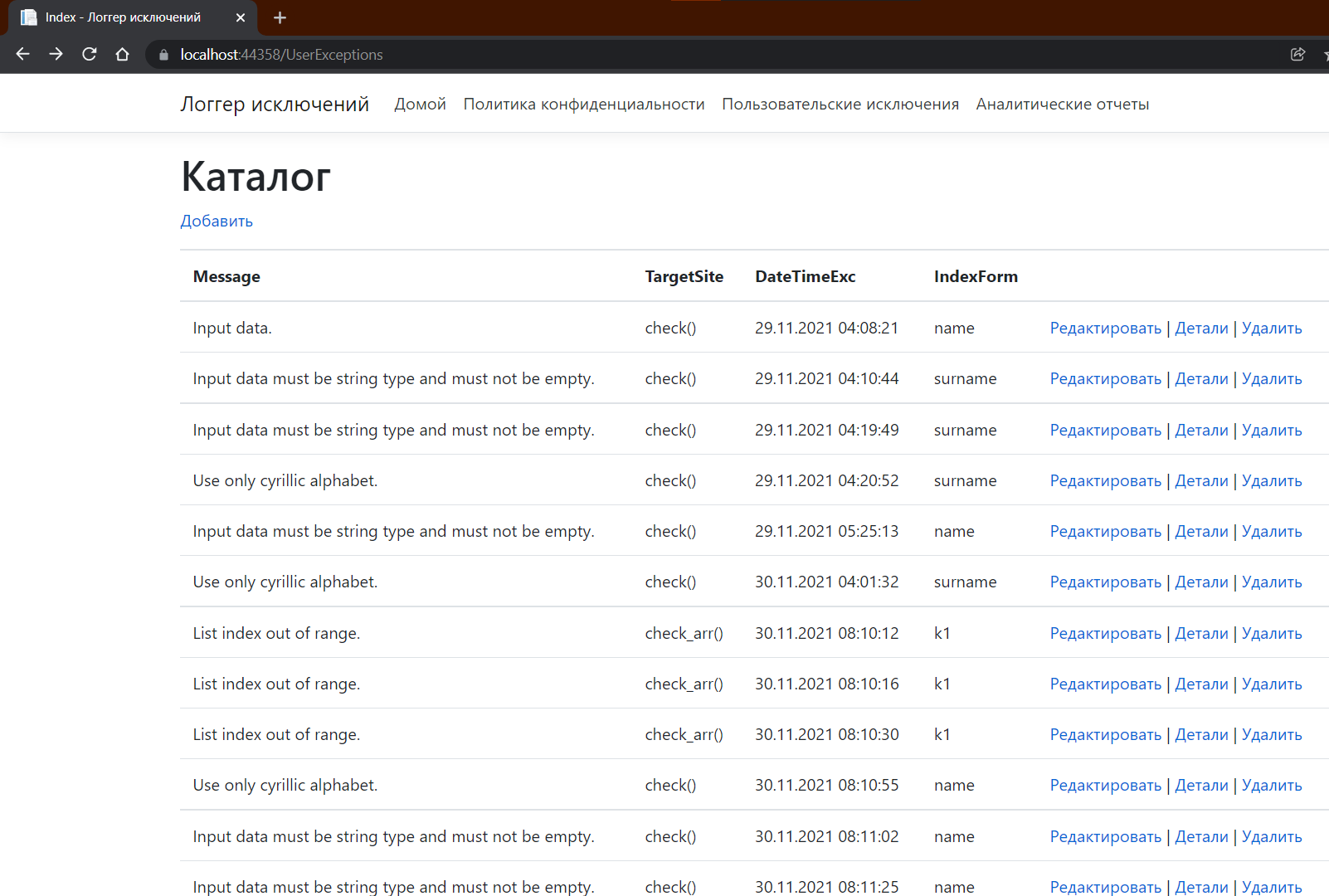


Рис.4 - Локализация интерфейса Web-клиента

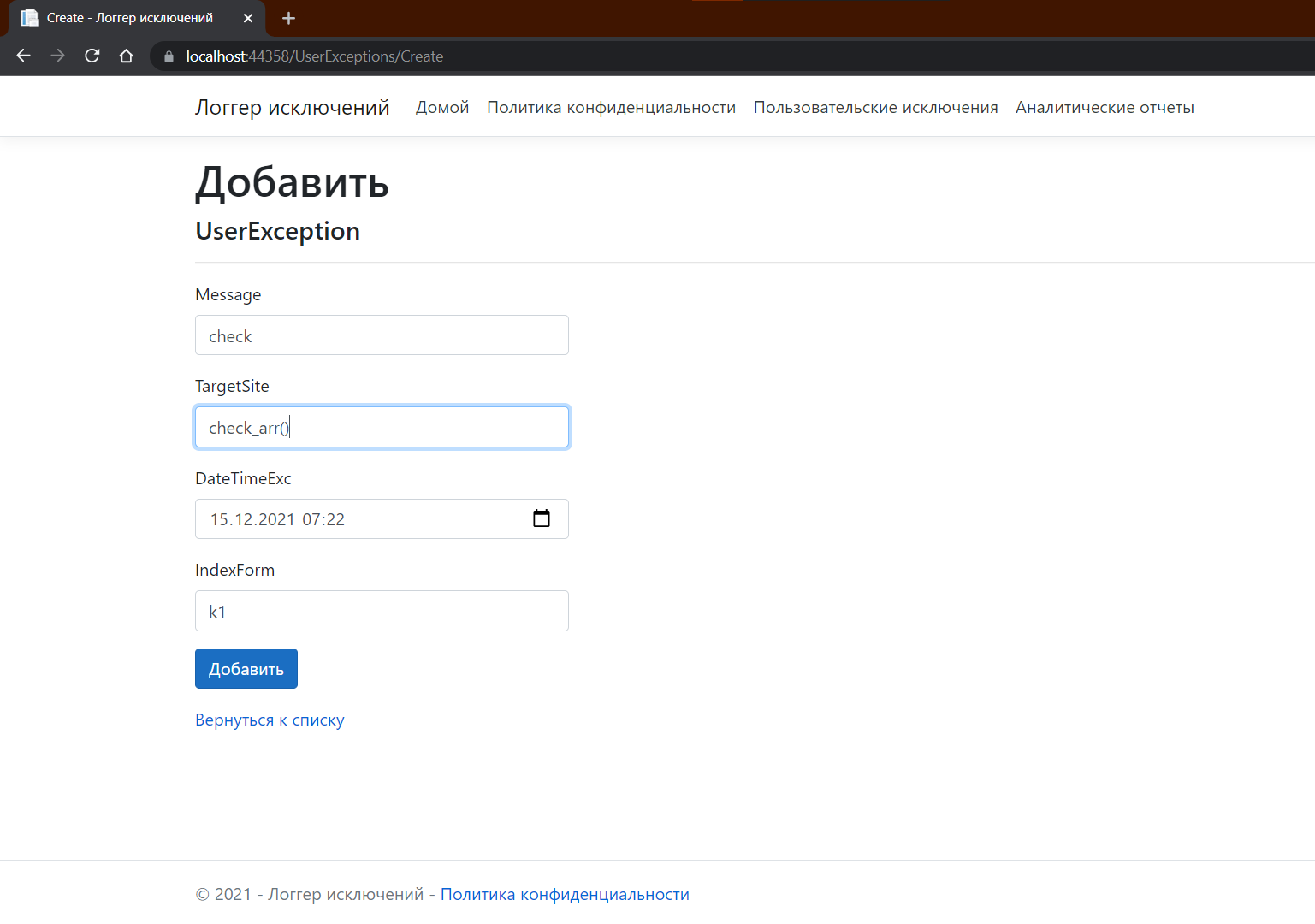


Рис.5 - Локализация интерфейса Web-клиента

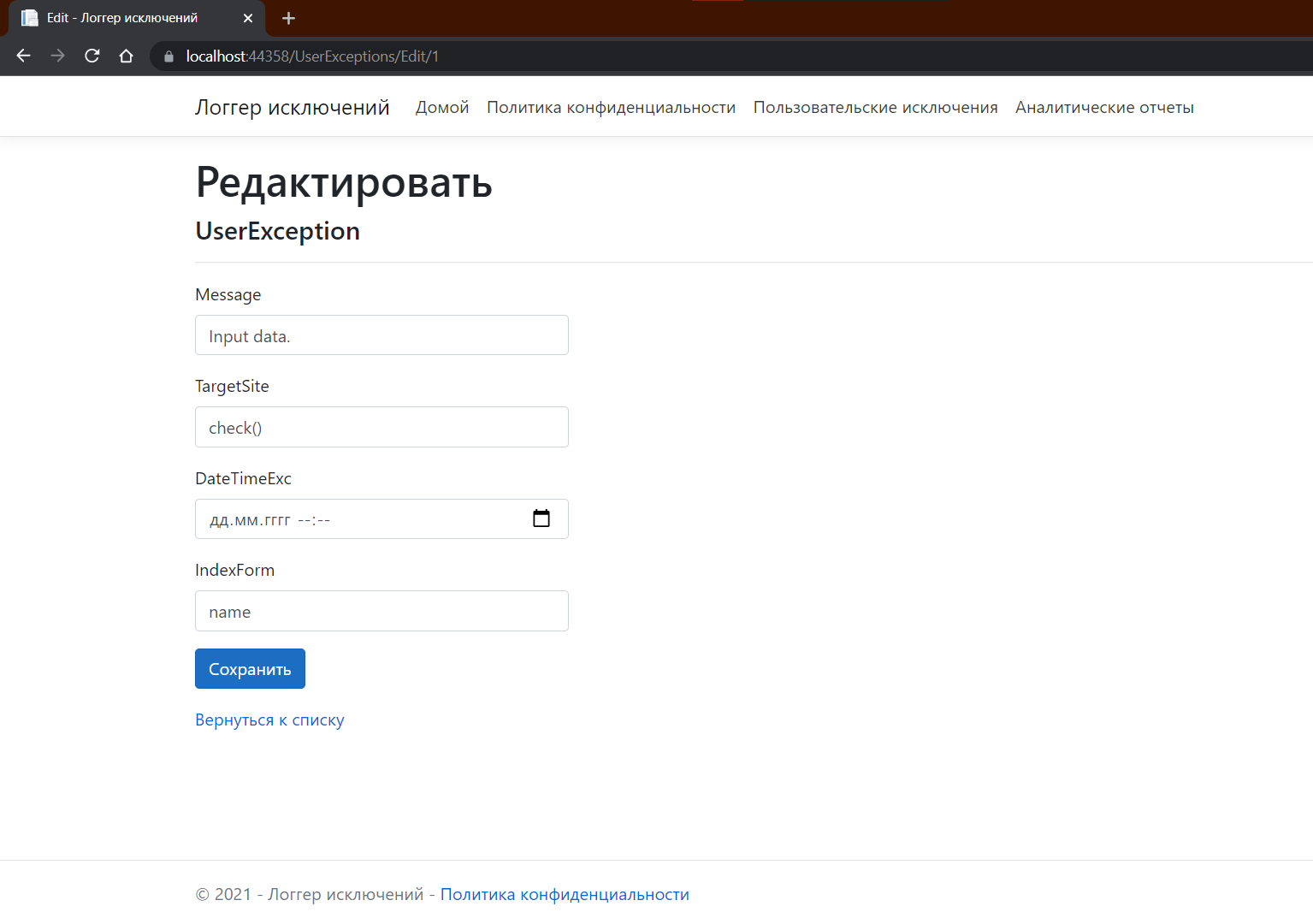


Рис.6 - Локализация интерфейса Web-клиента

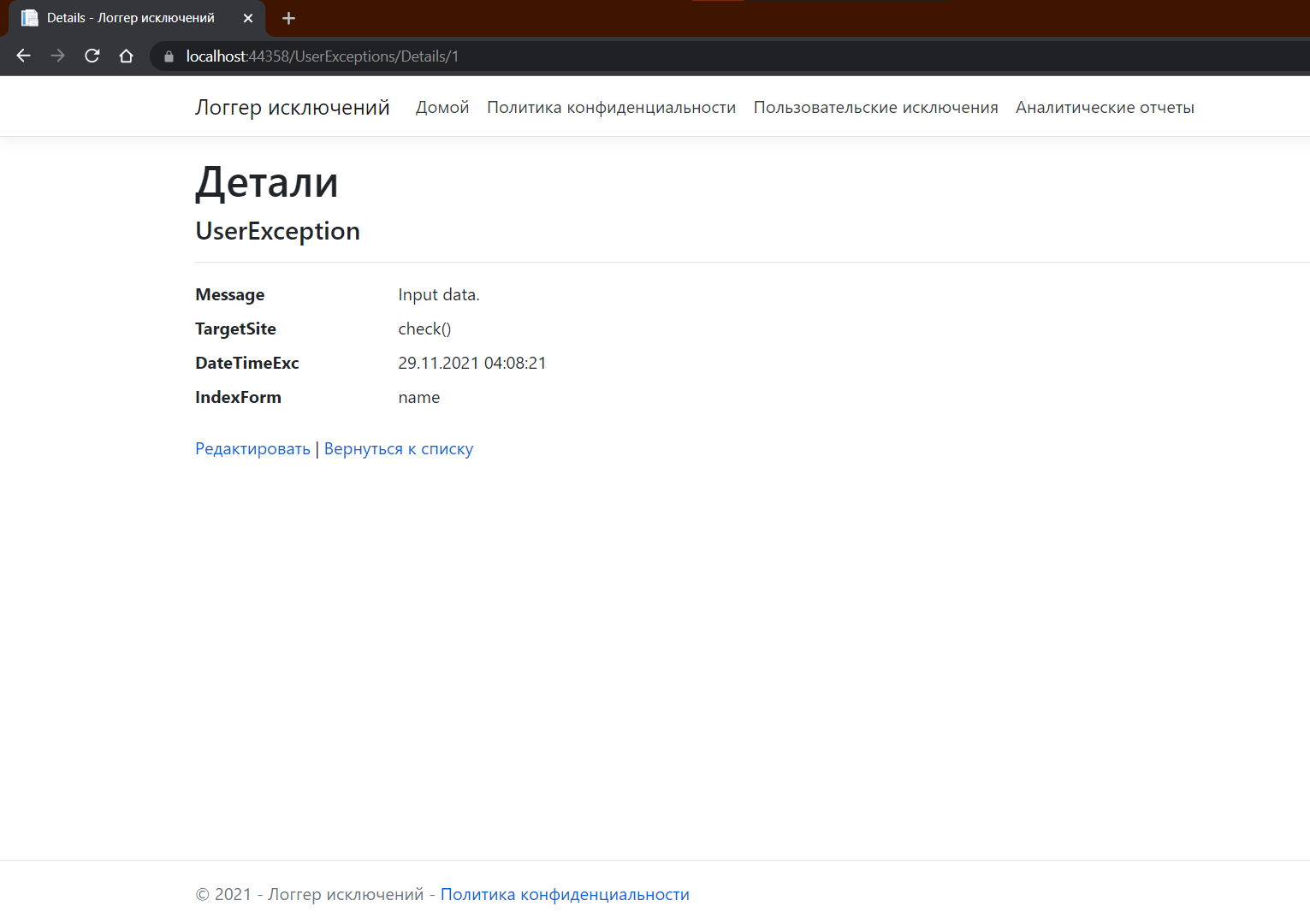


Рис.7 - Локализация интерфейса Web-клиента

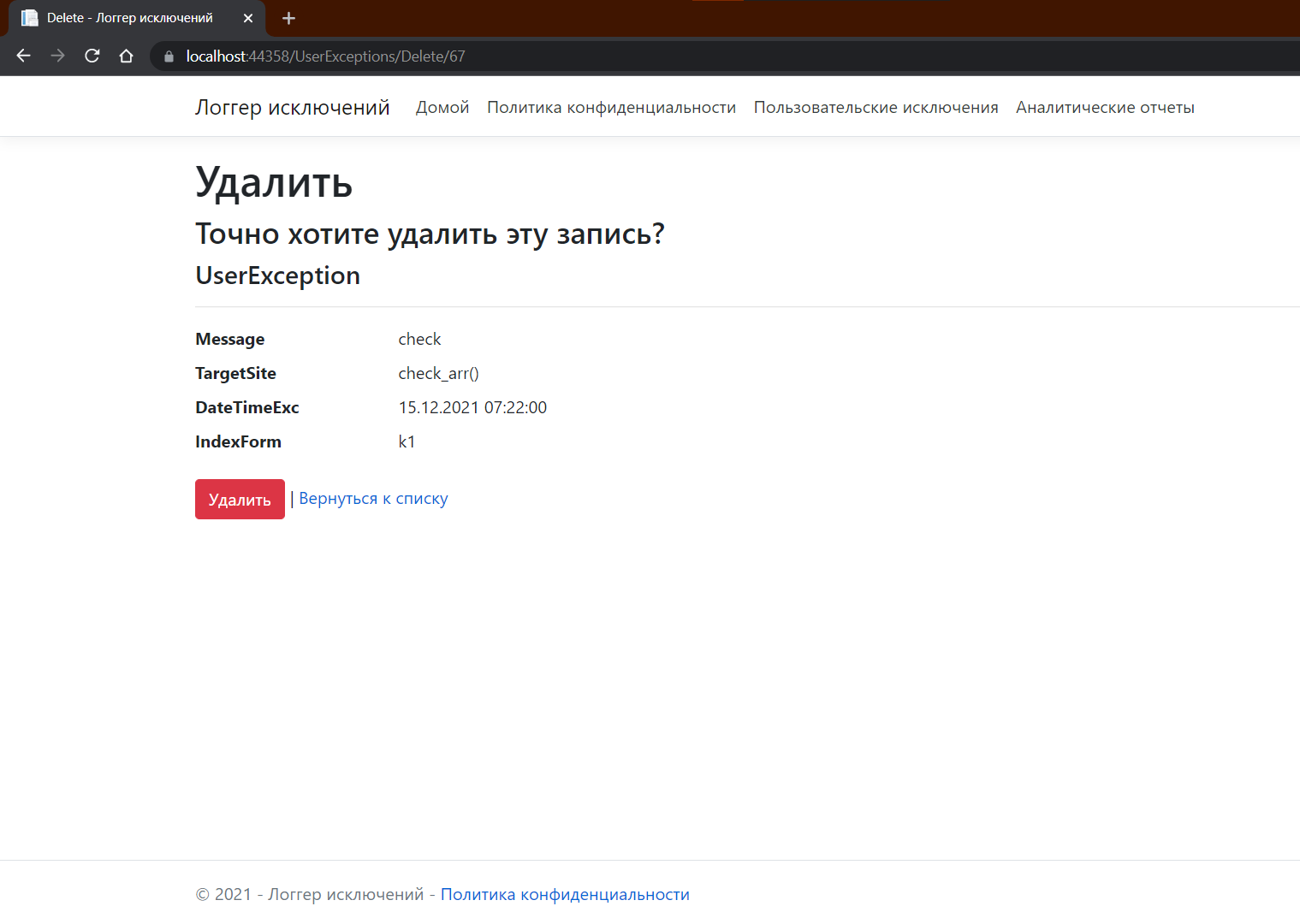
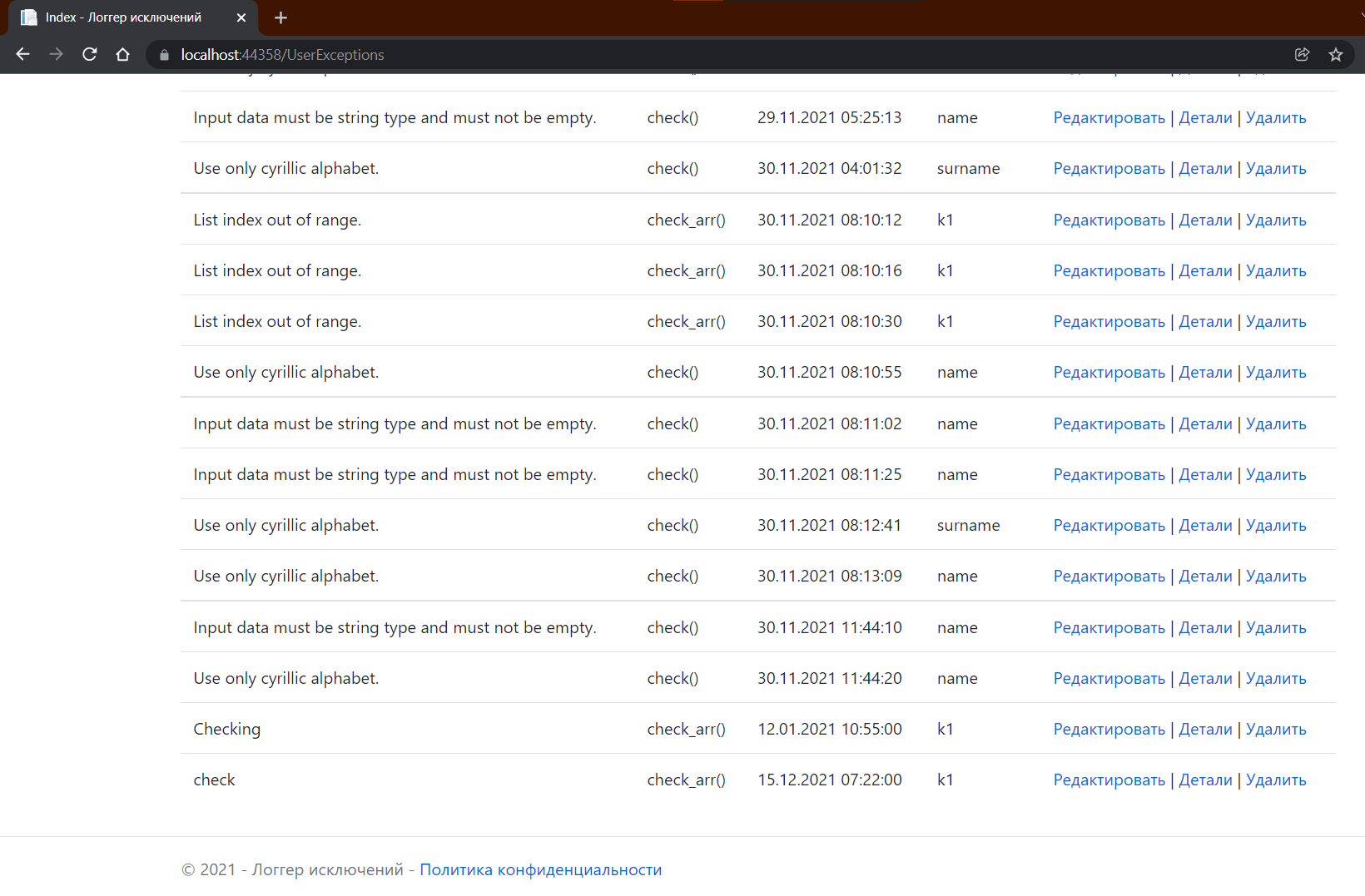


Рис.8 - Локализация интерфейса Web-клиента



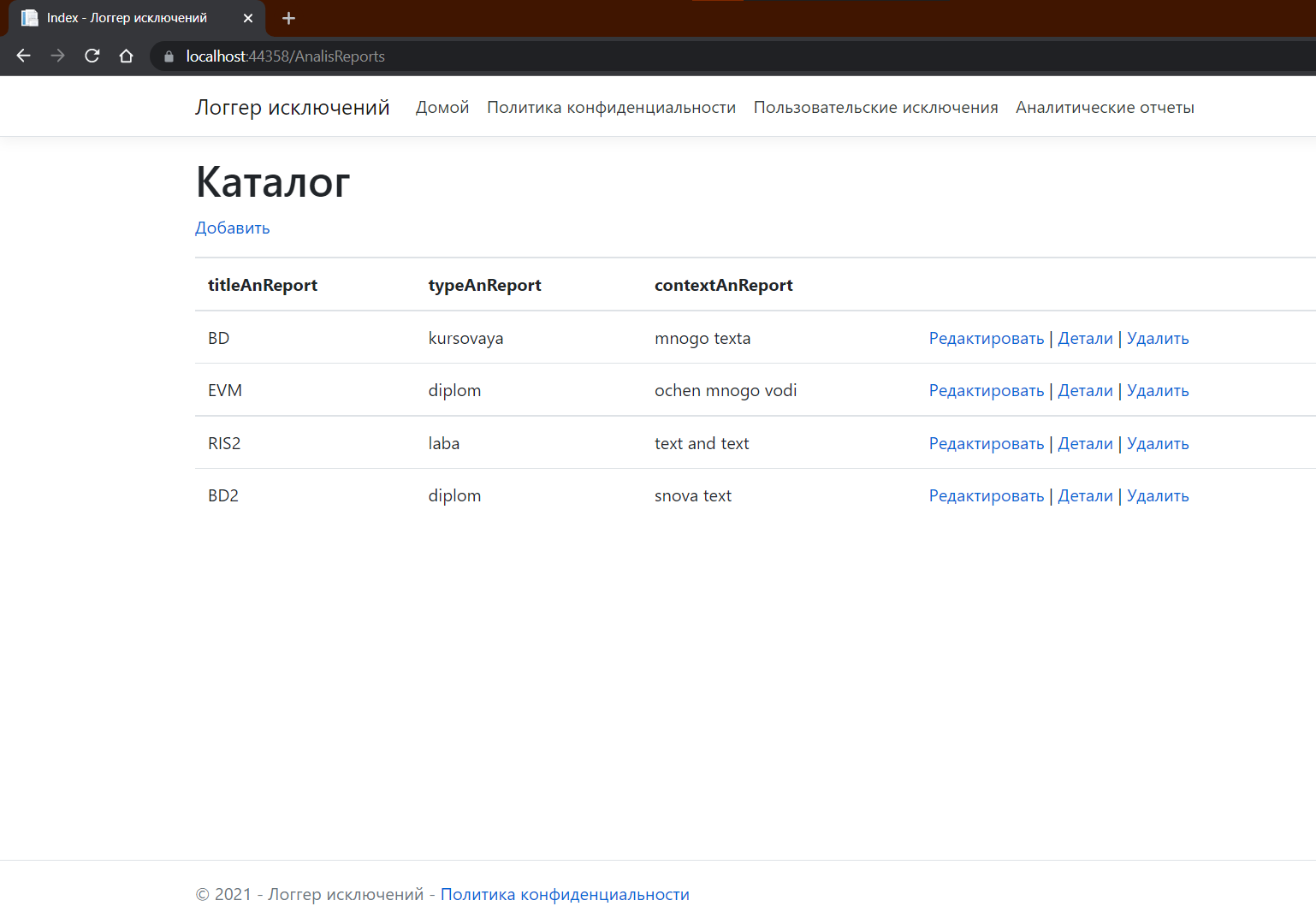
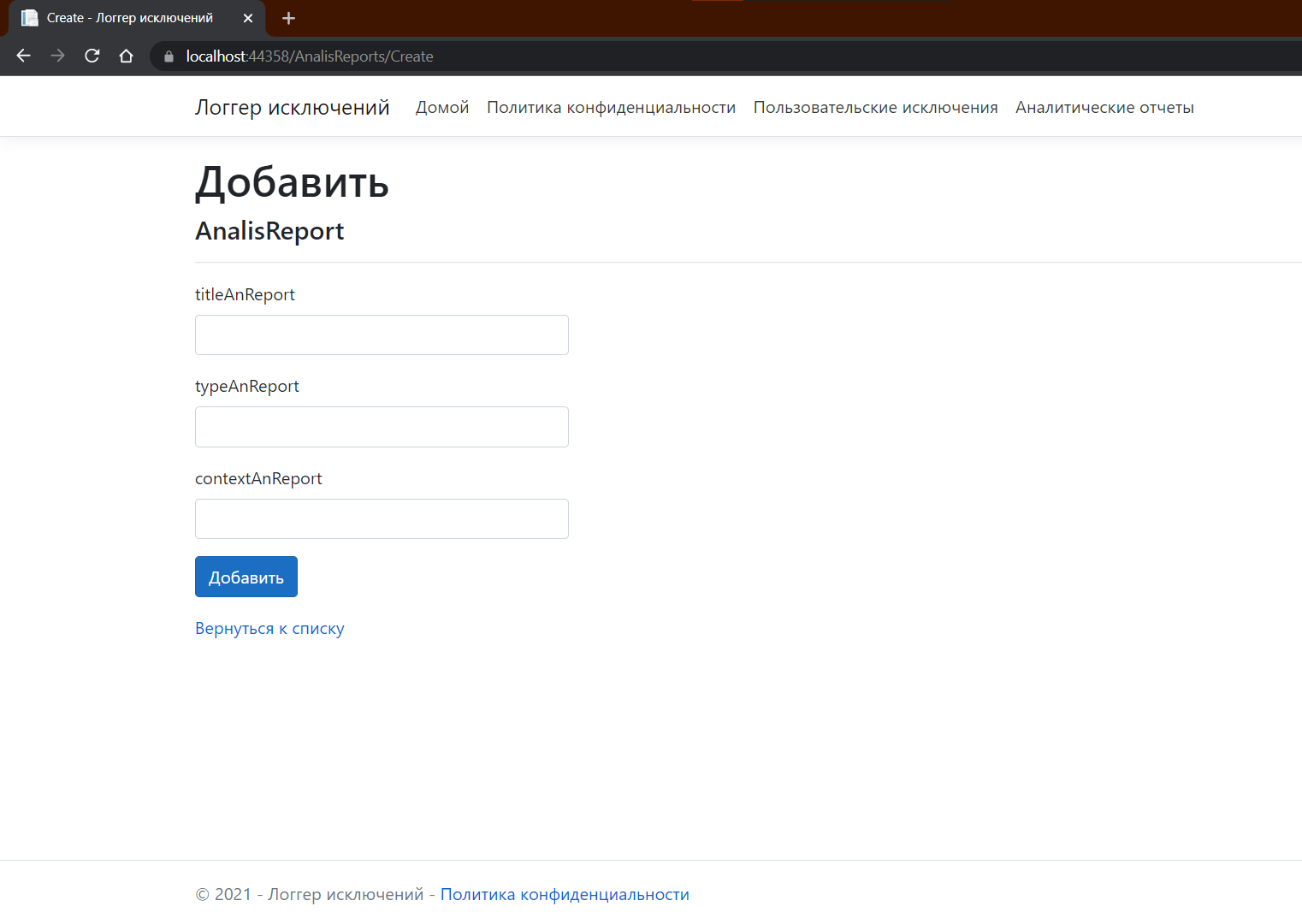
Рис. 9 - Тестирование кнопки «Добавить» для сущности UserException

Рис.10 - Локализация интерфейса Web-клиента



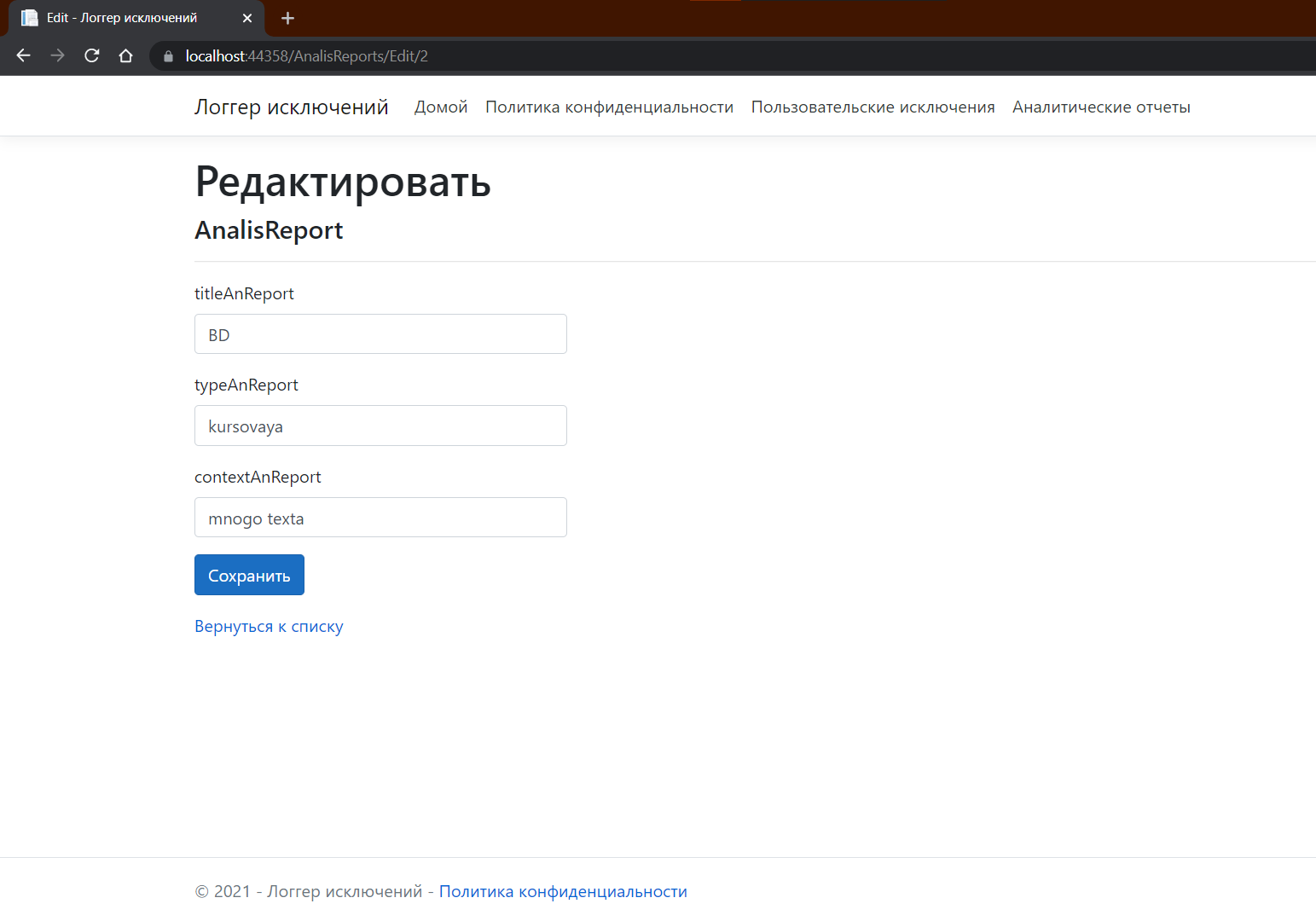
Рис.11 - Локализация интерфейса Web-клиента

Рис.12 - Локализация интерфейса Web-клиента

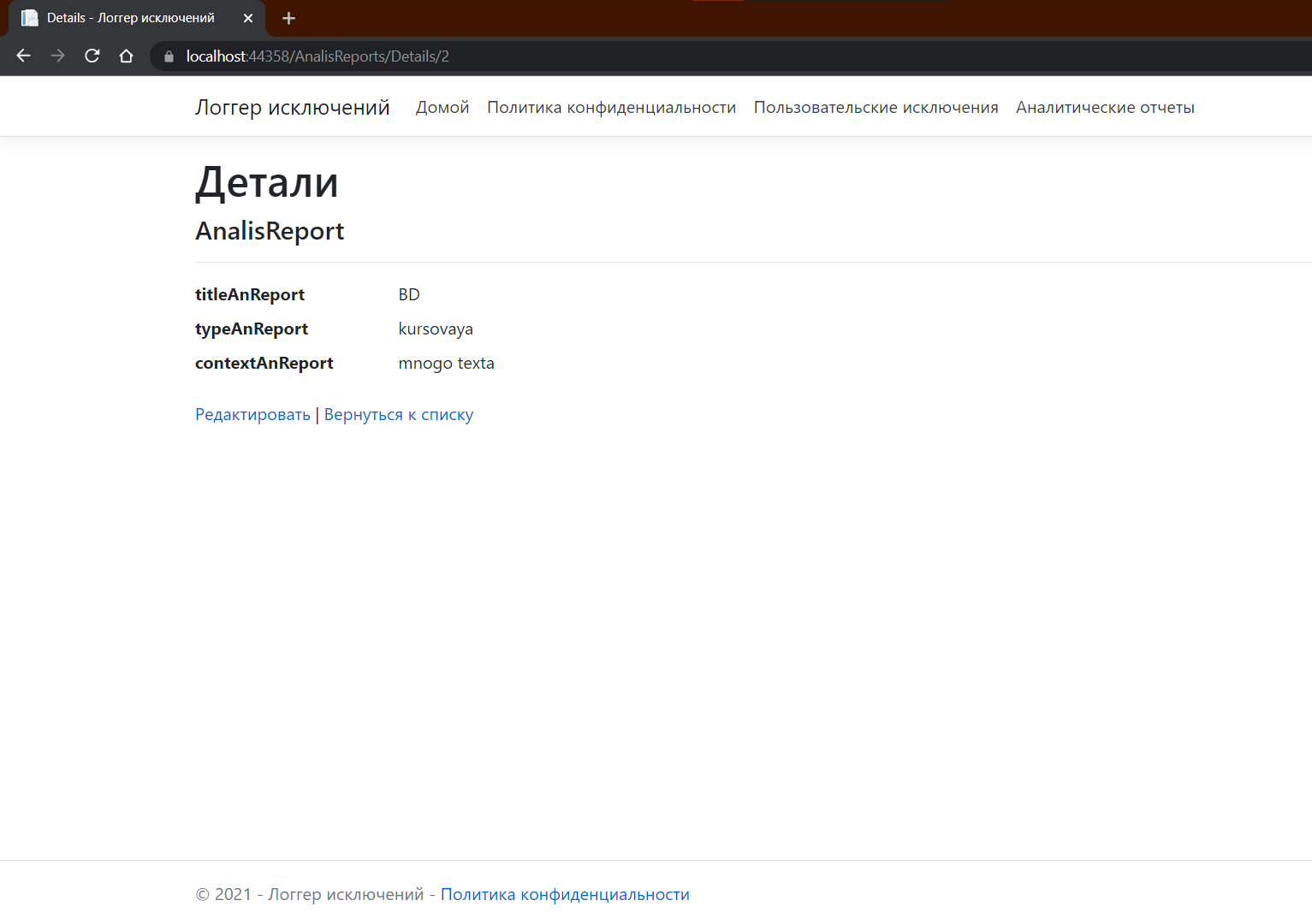


Рис.13 - Локализация интерфейса Web-клиента

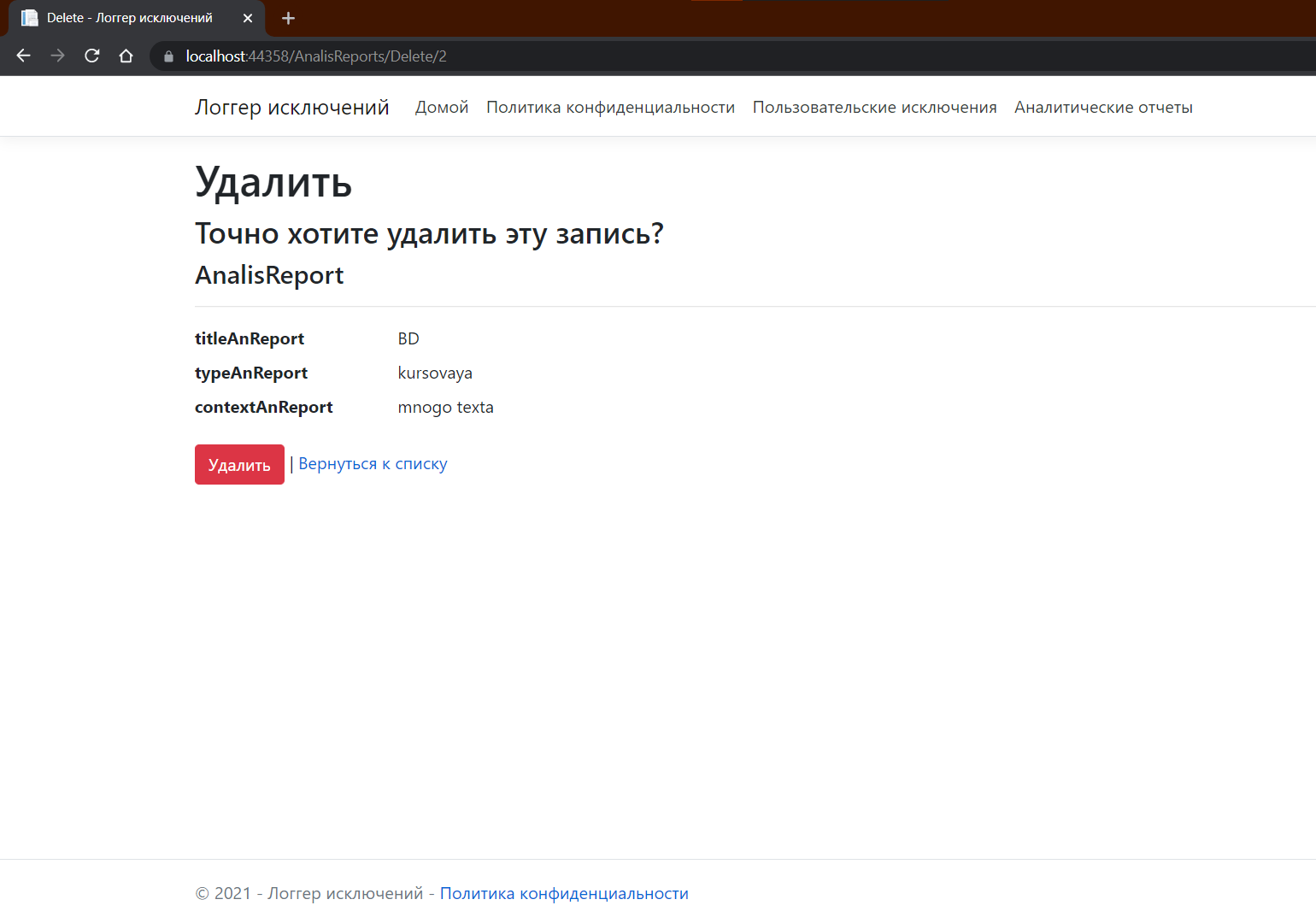


Рис.14 - Локализация интерфейса Web-клиента

**Дополнительное задание**

Локализовать весь интерфейс. Создать исключение, заносящееся в БД при попытке добавления пустых строк в таблицу.

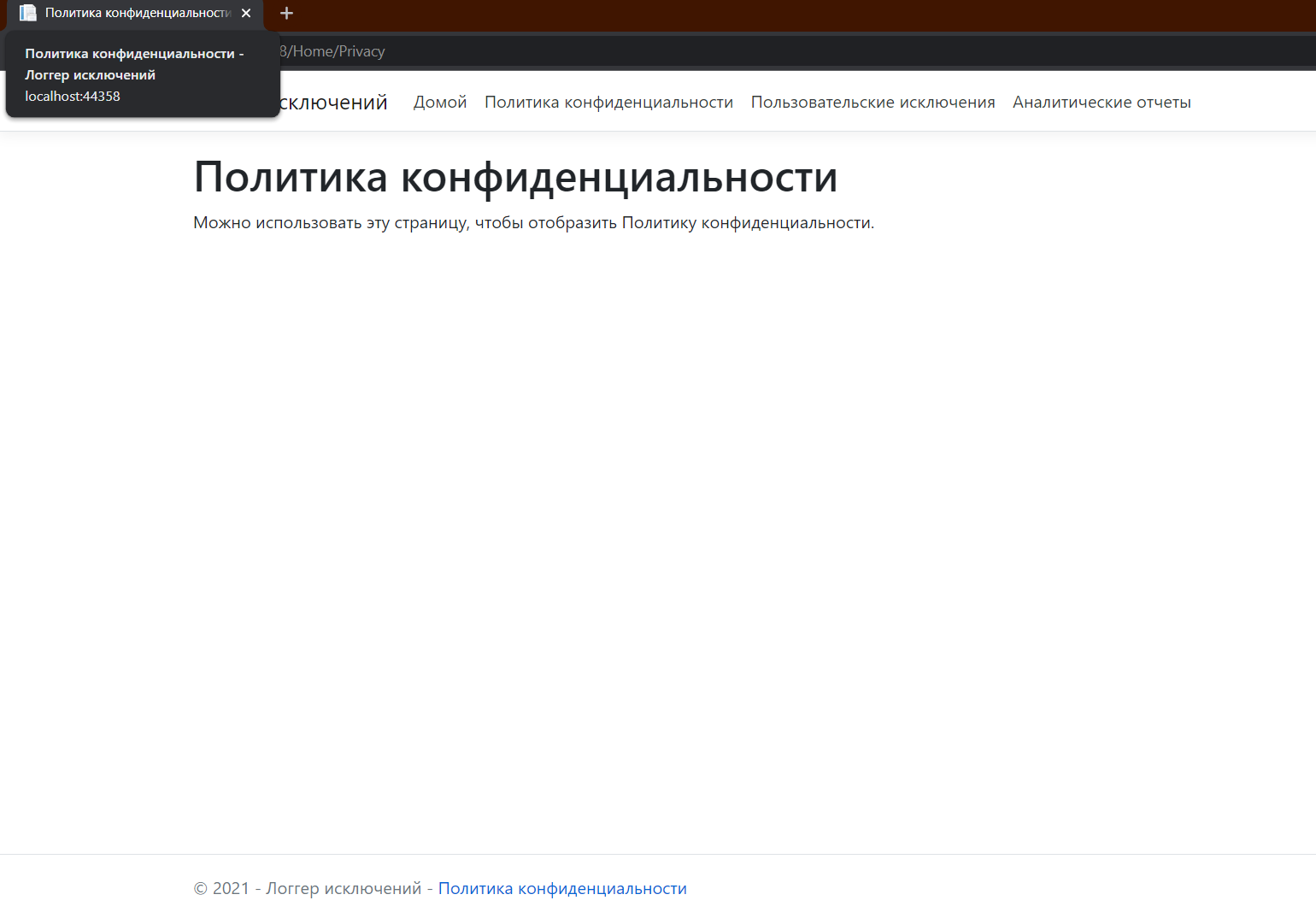
****

Рис.15 - Локализация интерфейса Web-клиента

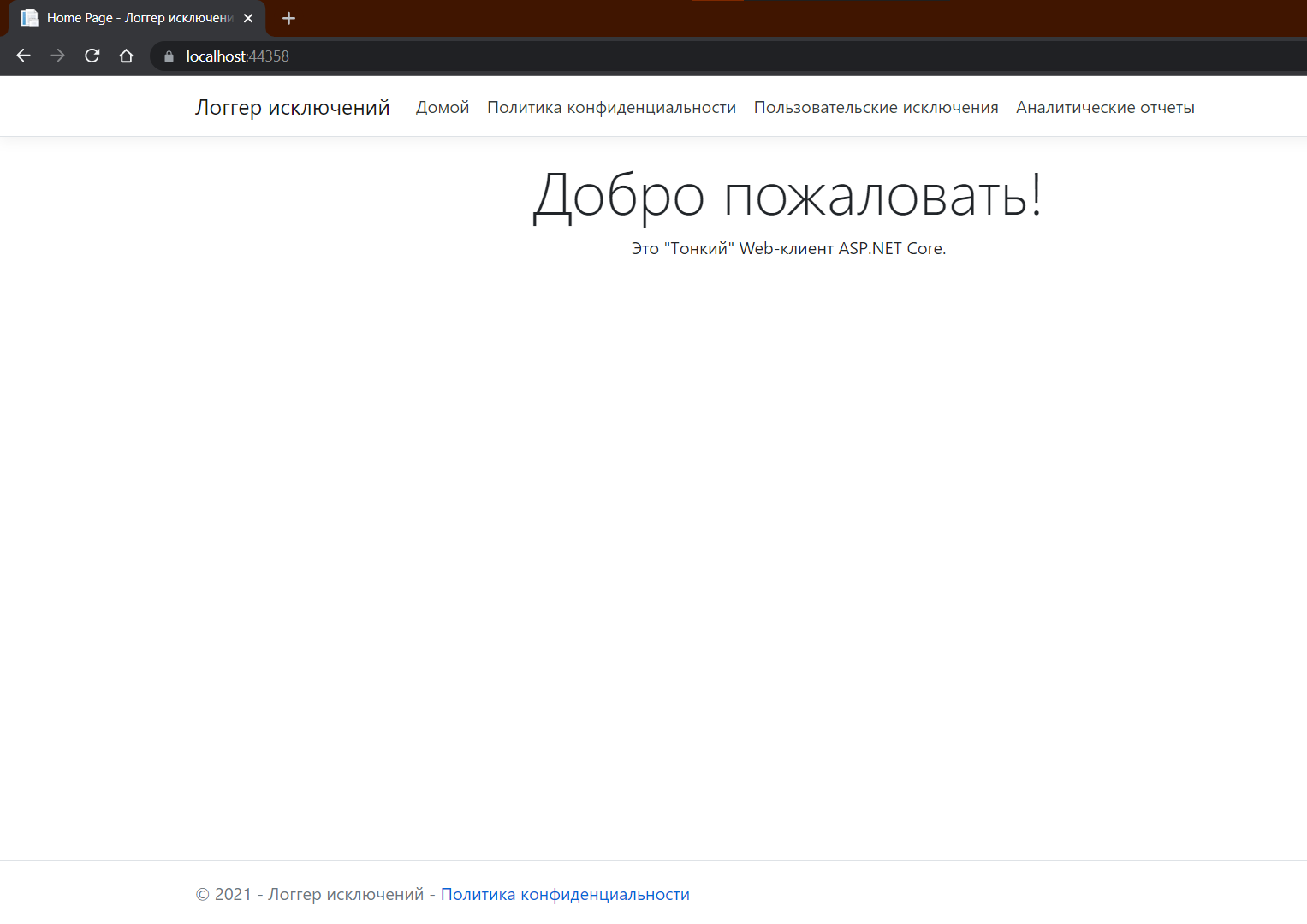


Рис.16 - Локализация интерфейса Web-клиента

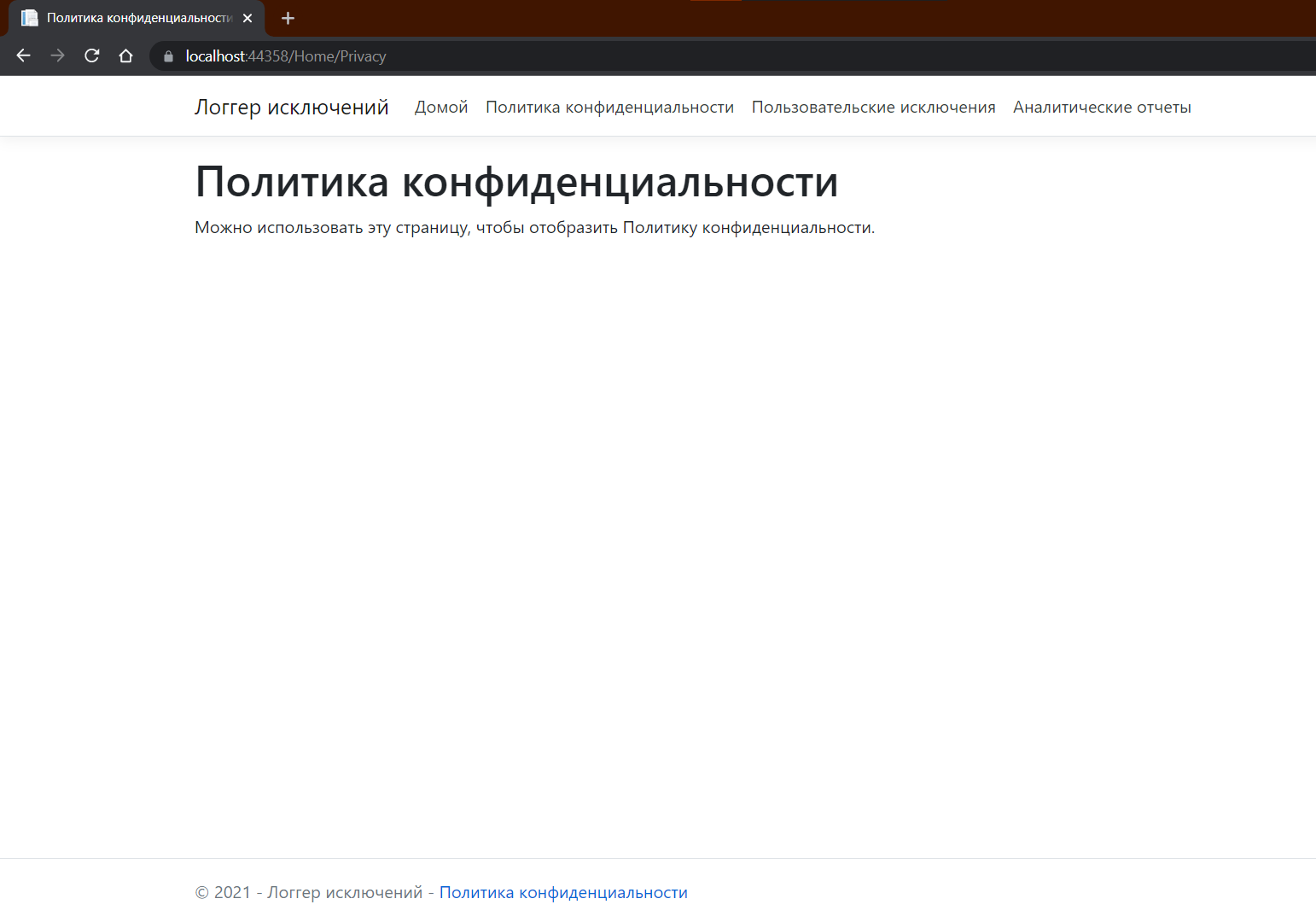


Рис.17 - Локализация интерфейса Web-клиента

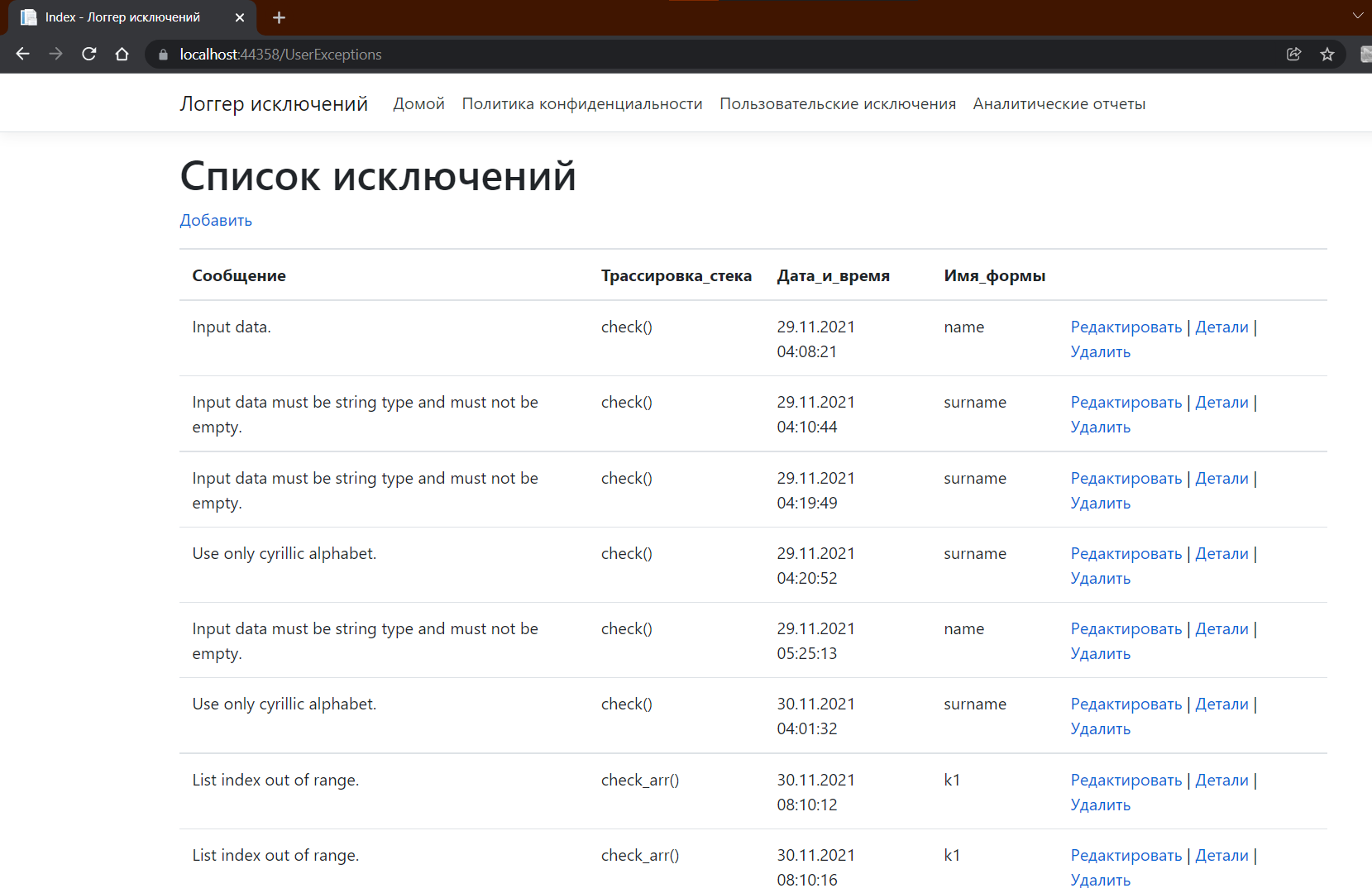


Рис.18 - Локализация интерфейса Web-клиента

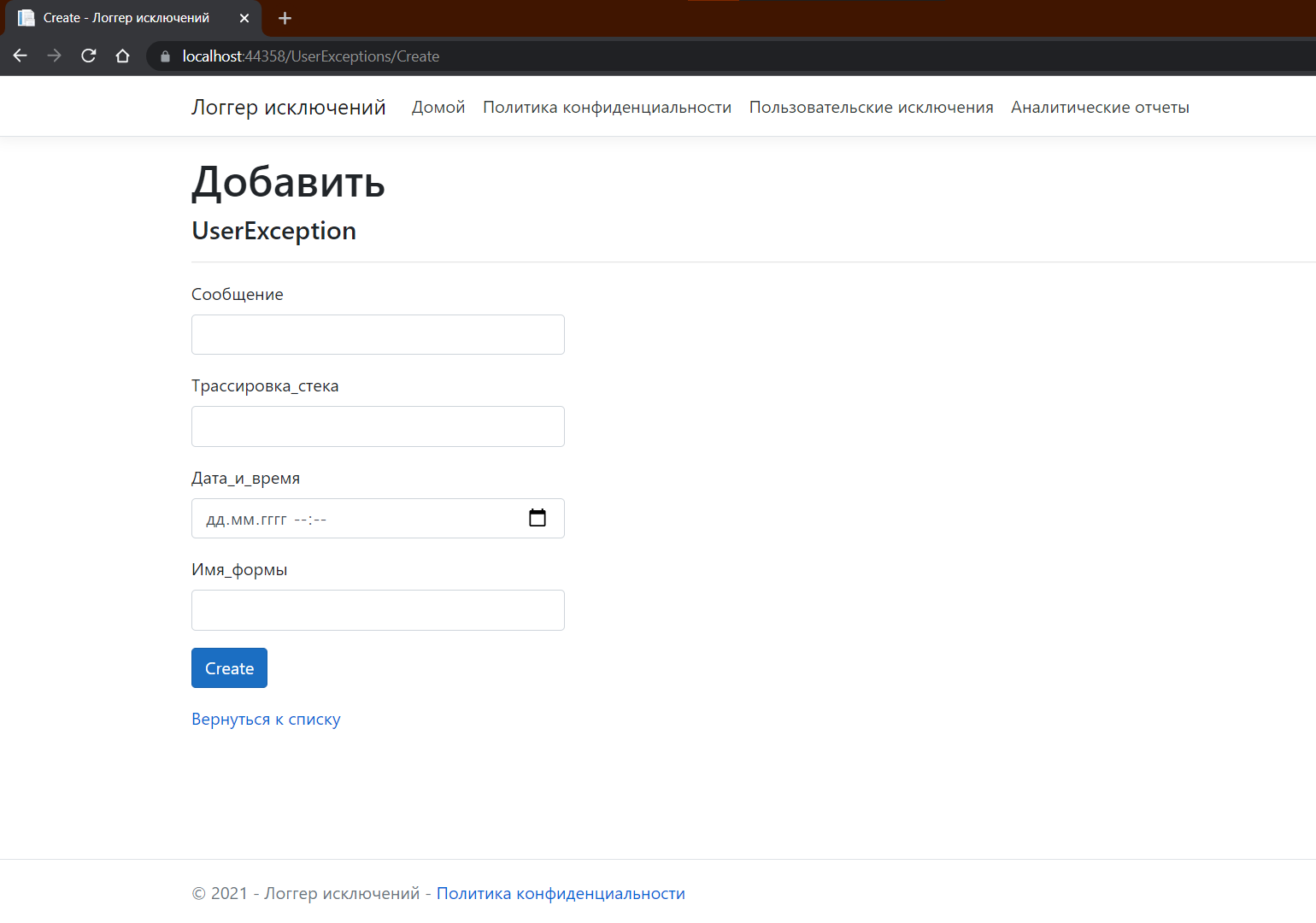


Рис.19 - Локализация интерфейса Web-клиента

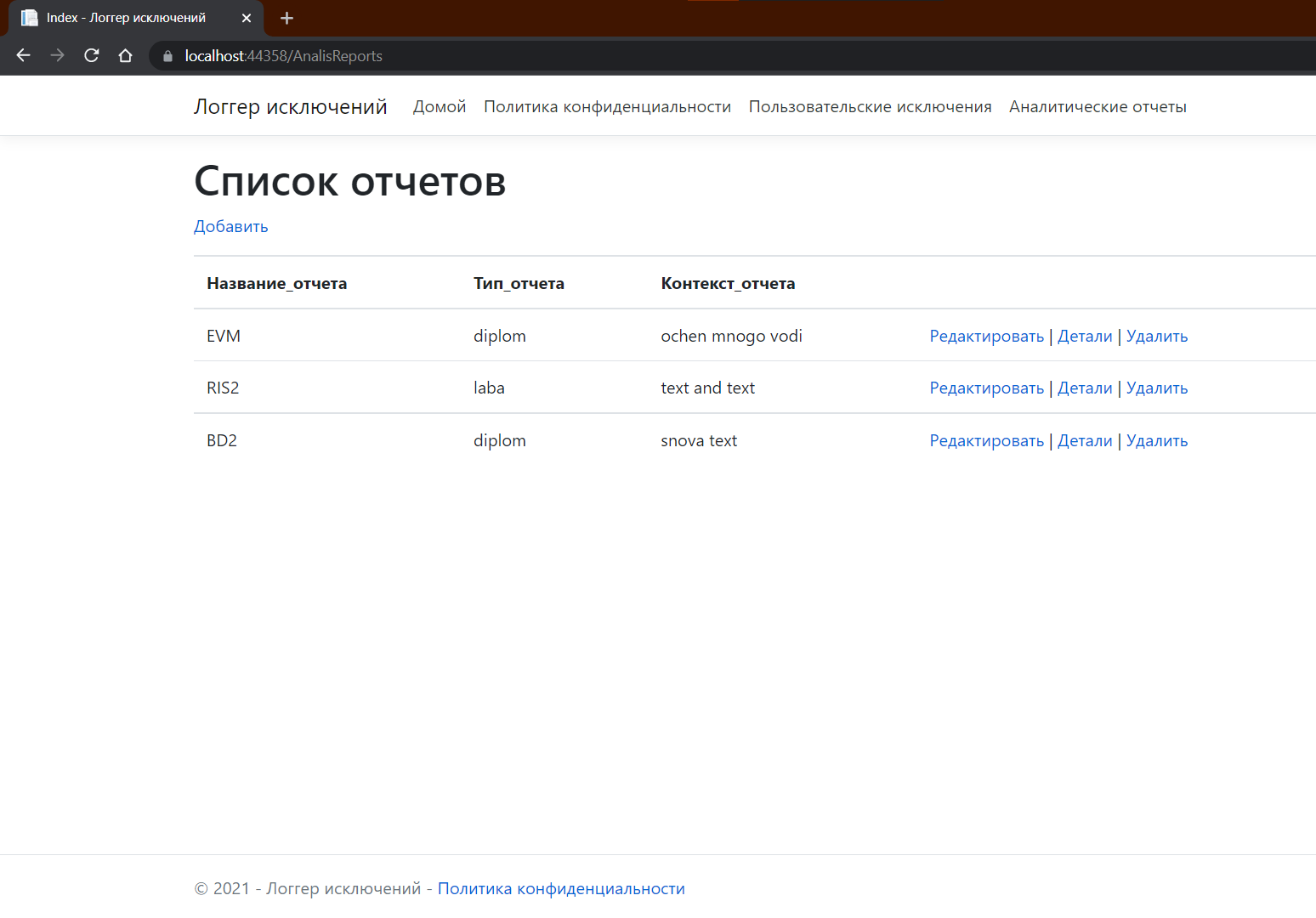


Рис.20 - Локализация интерфейса Web-клиента

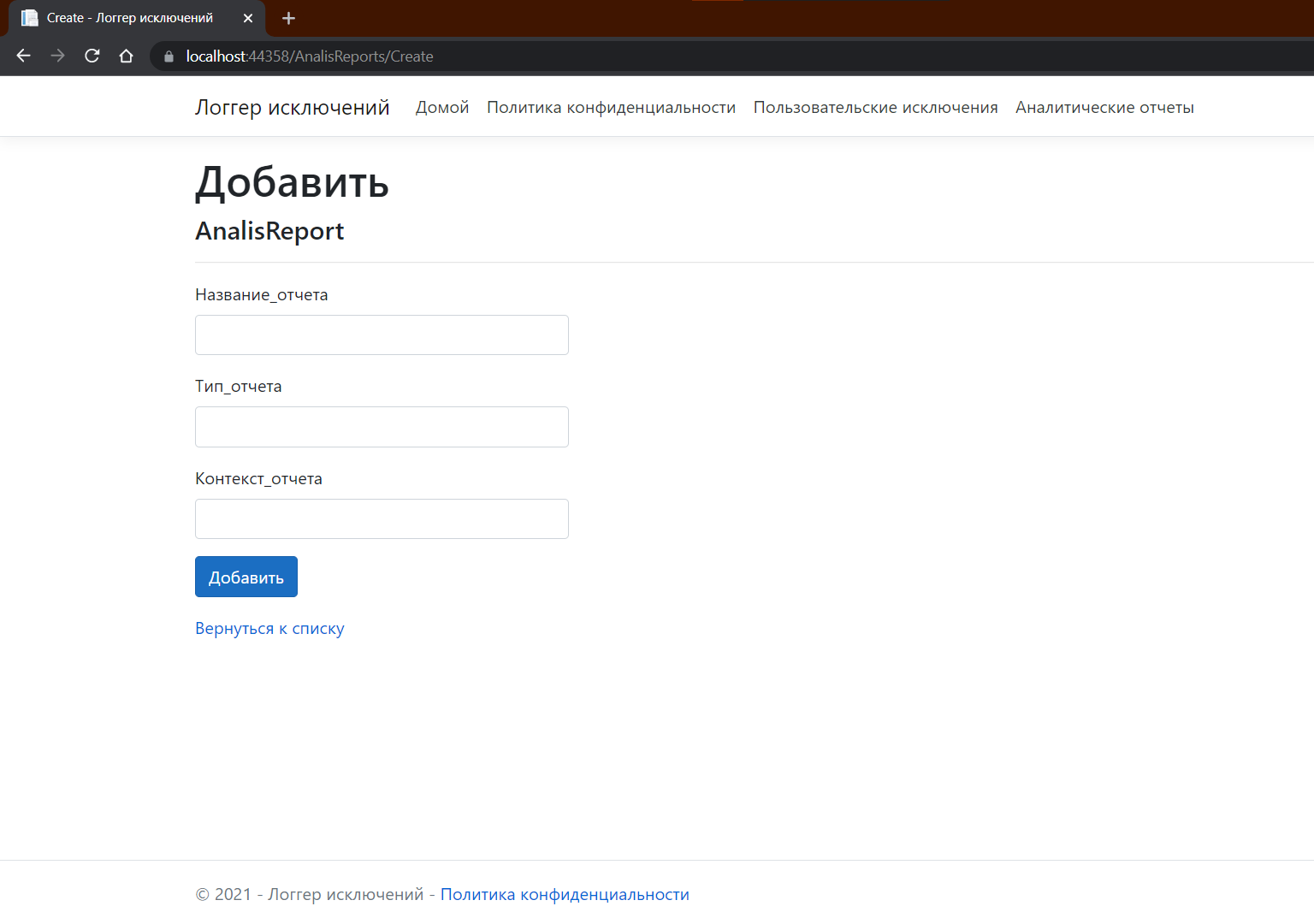


Рис.21 - Локализация интерфейса Web-клиента

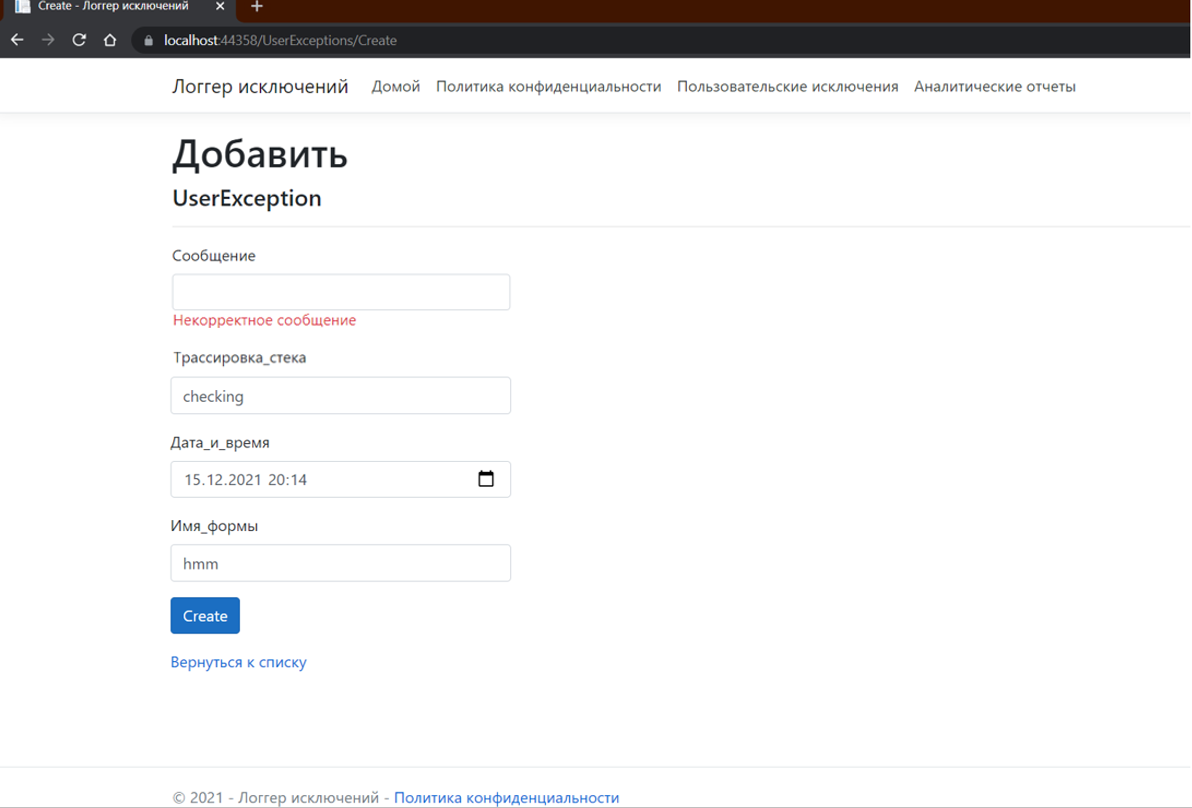


Рис.22 - Ошибка при добавлении пустой формы

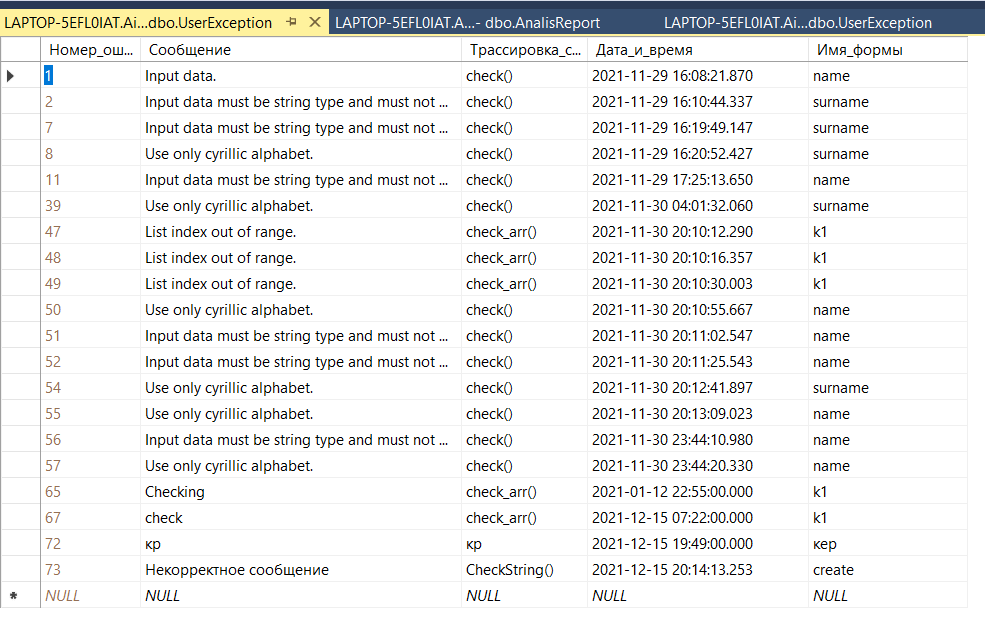


Рис.23 - Добавление исключения в UserException

**Вывод**

В соответствии с заявленной функциональностью разработали «Тонкий» Web-клиент, использующий ASP.NET Core MVC подходы разработки распределенных систем, обеспечивающий возможность удаленного мониторинга таблиц UserException и AnalisReport разработанной ранее архитектуры БД, а также поддержку вызова CRUD-операций (create, read, update, delete) над данными таблицами. Локализовали интерфейс Web-клиента.