**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Сибирский государственный университет науки и технологий**

**имени академика М.Ф. Решетнева»**

|  |
| --- |
| Институт инженерной экономики |
| институт |
| Кафедра информационных экономических систем |
| кафедра |

**КУРСОВАЯ РАБОТА**

по дисциплине «Компьютерные технологии в экономике»

|  |
| --- |
| Разработка информационно-справочной системы «Справочник грибов» |
| Тема |

Руководитель Тихоненко Д.В.

подпись, дата инициалы, фамилия

Обучающийся БПЦ21-01, 211519021 Шуляк Е. . номер группы, зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2024

## **СОДЕРЖАНИЕ**

[СОДЕРЖАНИЕ 2](#_Toc167832747)

[ВВЕДЕНИЕ 3](#_Toc167832748)

[ОСНОВНАЯ ЧАСТЬ 4](#_Toc167832749)

[1. Системный анализ предметной области 4](#_Toc167832750)

[2. Описание программы 4](#_Toc167832751)

[3. Руководство пользователя 5](#_Toc167832752)

[3.1. Область применения 5](#_Toc167832753)

[3.2. Уровень подготовки пользователя 6](#_Toc167832754)

[3.3. Назначение программы 7](#_Toc167832755)

[3.4. Программные и аппаратные требования к системе 7](#_Toc167832756)

[3.5. Пуск программы 8](#_Toc167832757)

[3.6 Состав и структура программы 9](#_Toc167832758)

[3.7 Описание операций 12](#_Toc167832759)

[4. Руководство программиста 13](#_Toc167832760)

[4.1. Код программы: 13](#_Toc167832761)

[5. Отладка и тестирование программы 41](#_Toc167832762)

[ЗАКЛЮЧЕНИЕ 44](#_Toc167832763)

[СПИСОК ЛИТЕРАТУРЫ 45](#_Toc167832764)

## **ВВЕДЕНИЕ**

В современном мире, в условиях стремительного развития информационных технологий, создание и использование информационно-справочных систем становится не только актуальной, но и необходимой задачей. Справочники, в том числе и справочники о грибах, играют важную роль в обеспечении доступа к полезной и актуальной информации для широкого круга пользователей. В данном контексте разработка информационно-справочной системы "Справочник грибов" приобретает особую актуальность.

Развитие информационных технологий и интернета приводит к тому, что доступ к информации становится все более мгновенным и удобным. В контексте справочников о грибах, это означает возможность быстрого определения вида гриба непосредственно на месте его обнаружения с помощью мобильных устройств, что является особенно важным с точки зрения безопасности и предотвращения отравлений.

Важным аспектом разработки информационно-справочной системы является также ее удобство использования и доступность для широкого круга пользователей. В связи с этим, система "Справочник грибов" разрабатывается с учетом современных принципов дизайна пользовательского интерфейса и интерактивности, что позволит пользователям легко ориентироваться в приложении и быстро находить необходимую информацию.

Кроме того, следует отметить важность разработки подобных справочных систем с точки зрения образования и экологического просвещения. Предоставление пользователю надежной и проверенной информации о грибах способствует формированию правильного подхода к сбору и использованию этого уникального природного ресурса, а также помогает предотвращать случаи отравлений и сохранять биоразнообразие.

В рамках данной курсовой работы разрабатывается информационно-справочная система "Справочник грибов", предназначенная для ознакомления пользователей с различными видами грибов, их съедобностью, а также мерами предосторожности при сборе. Система реализована на языке программирования C# с использованием JavaScript для интерактивных элементов.

Объем информации, представленной в системе, охватывает не только описание различных видов грибов, но и предоставляет пользователю возможность быстро и удобно получить информацию о том, можно ли употреблять данный вид гриба в пищу, а также какие меры предосторожности следует соблюдать при его сборе и употреблении.

Таким образом, разработка информационно-справочной системы "Справочник грибов" не только актуальна с точки зрения обеспечения доступа к полезной информации о грибах, но и имеет важное значение с практической, образовательной и экологической точек зрения.

**ОСНОВНАЯ ЧАСТЬ**

## **Системный анализ предметной области**

Системный анализ предметной области является важным этапом при разработке информационно-справочной системы "Справочник грибов". Для полноценного понимания требований к системе необходимо провести анализ основных аспектов предметной области, включая характеристики грибов, их классификацию, особенности сезонности и мест обитания, а также информацию о безопасности при сборе и употреблении грибов.

1.1. Характеристики грибов:

Различные виды грибов имеют уникальные характеристики, такие как форма, цвет, размер, текстура и аромат.

Для каждого вида грибов необходимо предоставить информацию о его биологических особенностях, сезоне сбора, местах обитания и распространения.

1.2. Классификация грибов:

Грибы классифицируются по различным признакам, таким как структура, способ образования спор, морфологические особенности и химический состав.

Классификация грибов на съедобные, несъедобные и ядовитые является важной для безопасного использования в пищу.

1.3. Сезонность и места обитания:

Грибы имеют определенные сезоны сбора, которые зависят от климатических и природных условий.

Отображение информации о сезонности и местах обитания грибов позволит пользователям оптимально выбирать время и место для сбора.

1.4. Безопасность при сборе и употреблении:

Важно предоставить пользователям информацию о безопасных методах сбора грибов, правилах их обработки и хранения.

Также необходимо описать признаки отличия съедобных грибов от ядовитых и меры предосторожности при обнаружении неизвестных видов.

Анализ предметной области позволит определить не только функциональные требования к системе, но и основные потребности пользователей, что будет полезно для разработки удобного и информативного интерфейса.

# **Описание программы**

Информационно-справочная система "Справочник грибов" представляет собой веб-приложение, разработанное для запуска в современных веб-браузерах. Программа размещается на сервере и доступна пользователям через интернет, обеспечивая широкий доступ к полезной информации о грибах в любое время и из любой точки мира.

Основные функции программы включают в себя:

1. Поиск и просмотр информации о различных видах грибов: пользователи могут получить доступ к подробным описаниям различных видов грибов, включая их характеристики, фотографии, сезон сбора и места обитания.
2. Интерактивные функции: пользователи могут взаимодействовать с системой с помощью интерактивных элементов, таких как фильтры по характеристикам грибов, голосования за полезность информации и комментирование.

Дополнительно, система также может предоставлять следующие возможности:

1. Мобильное приложение: для удобства использования в полевых условиях пользователи могут загрузить мобильное приложение, которое позволит им получать доступ к справочнику грибов прямо с мобильного устройства.

Таким образом, информационно-справочная система "Справочник грибов" предназначена для широкого круга пользователей, интересующихся грибами, и обеспечивает удобный доступ к полезной информации о грибах как в домашних условиях, так и в полевых условиях при сборе грибов.

В качестве выходных данных является информация о книгах в Web-странице.

Запуск Информационно- справочной системы

Закрытие системы

Ядовитые грибы

Несъедобные грибы

Отображение списка представлений

Съедобные грибы

О нас

Меры предосторожности

Рисунок 1 - Функционирование программы «Информационно-справочная система»

# **Руководство пользователя**

## Область применения

Область применения информационно-справочной системы "Справочник грибов" охватывает несколько ключевых аспектов, связанных с безопасным сбором и использованием грибов. Основными пользователями данной системы могут быть грибники, ботаники, любители природы, кулинары, а также любые лица, интересующиеся грибами. Система может использоваться в следующих ситуациях:

Безопасный сбор грибов:

Грибники могут оперативно определять вид грибов, чтобы избежать сбора ядовитых или несъедобных экземпляров.

Возможность сравнения внешнего вида грибов с картинками и описаниями помогает предотвратить ошибки при сборе.

Образовательные цели:

Система может служить учебным пособием для школ, университетов и специализированных курсов по ботанике.

Помощь в изучении грибов, их классификации и особенностей.

Кулинарные цели:

Кулинары могут использовать справочник для определения съедобных грибов и выбора подходящих для приготовления блюд.

Описание вкусовых качеств и рекомендаций по приготовлению различных грибов.

Меры предосторожности:

Вкладка "Меры предосторожности" предоставляет информацию о первых признаках отравления грибами и рекомендуемые действия при подозрении на отравление.

Повышение осведомленности пользователей о потенциальных опасностях, связанных с неправильным употреблением грибов.

Информационно-просветительская деятельность:

Вкладка "О нас" может включать информацию о создателях системы, партнёрах и источниках информации, что повысит доверие пользователей к предоставляемым данным.

Распространение полезной информации среди широкой аудитории, популяризация безопасного и ответственного отношения к сбору и употреблению грибов.

## Уровень подготовки пользователя

Для эффективного использования информационно-справочной системы "Справочник грибов" пользователям необходимо обладать базовым уровнем подготовки. Это подразумевает следующие навыки и знания:

1. **Базовые компьютерные навыки:**
   1. Умение пользоваться интернет-браузером для доступа к онлайн-ресурсам.
   2. Навыки навигации по вкладкам и страницам веб-сайтов.
   3. Способность использовать поисковую строку для быстрого нахождения нужной информации.
2. **Основы работы с мобильными приложениями:**
   1. Умение скачивать и устанавливать приложения на мобильные устройства.
   2. Навыки использования сенсорного экрана для навигации по приложению.
3. **Базовые знания о грибах:**
   1. Понимание основных терминов, связанных с грибами (например, шляпка, ножка, споры).
   2. Знание различий между съедобными, несъедобными и ядовитыми грибами (общие признаки и предосторожности).
4. Чтение и понимание текстовой информации:
   1. Способность читать и понимать краткие описания и инструкции.
   2. Умение интерпретировать визуальные данные, такие как изображения грибов, для их идентификации.

## Назначение программы

Программа "Справочник грибов" предназначена для предоставления пользователям достоверной и легко доступной информации о различных видах грибов. Она помогает безопасно определять и различать съедобные, несъедобные и ядовитые грибы, способствует повышению знаний о грибах и обеспечивает меры предосторожности при их сборе и употреблении.

## Программные и аппаратные требования к системе

Аппаратные требования:

Для настольных компьютеров и ноутбуков:

Процессор: Двухъядерный процессор с тактовой частотой 1.6 ГГц или выше.

Оперативная память: Минимум 2 ГБ.

Жесткий диск: Минимум 100 МБ свободного места для установки программы.

Монитор: Разрешение экрана 1024x768 или выше.

Интернет-соединение: Для обновления базы данных и доступа к онлайн-функциям.

Для мобильных устройств (смартфоны и планшеты):

Процессор: Двухъядерный процессор с тактовой частотой 1.2 ГГц или выше.

Оперативная память: Минимум 1 ГБ.

Хранилище: Минимум 50 МБ свободного места для установки приложения.

Экран: Разрешение экрана 800x600 или выше.

Интернет-соединение: Для обновления базы данных и доступа к онлайн-функциям.

Программные требования:

Для настольных компьютеров и ноутбуков:

Операционная система:

Windows 7 или выше.

macOS 10.10 или выше.

Linux (современные дистрибутивы).

Браузер: Современный веб-браузер (Google Chrome, Mozilla Firefox, Microsoft Edge, Safari) с включенной поддержкой JavaScript.

Дополнительные компоненты: Adobe Reader или другой PDF-ридер для чтения документации.

Для мобильных устройств (смартфоны и планшеты):

Операционная система:

Android 5.0 или выше.

iOS 10.0 или выше.

Приложение: Установка из официальных магазинов приложений (Google Play Store, Apple App Store).

Дополнительные требования:

Интернет-соединение: Для обновления базы данных грибов, доступа к онлайн-ресурсам и получения актуальной информации.

Камера (опционально для мобильных устройств): Для функции распознавания грибов по фотографии (если такая функция предусмотрена).

Эти требования обеспечивают корректную работу программы "Справочник грибов" на различных устройствах и платформах, позволяя пользователям эффективно использовать все её возможности.

## Пуск программы

Открытие браузера:

Запустите веб-браузер на вашем устройстве. Поддерживаются современные браузеры, такие как Google Chrome, Mozilla Firefox, Microsoft Edge, Safari и другие.

Ввод URL-адреса:

В адресной строке браузера введите URL-адрес веб-приложения "Справочник грибов" (например, https://www.spravochnik-gribov.com) и нажмите Enter.

Загрузка страницы:

Дождитесь полной загрузки главной страницы сайта "Справочник грибов".

Начальная настройка (при первом запуске):

При первом посещении сайта может потребоваться начальная настройка, такая как выбор языка интерфейса. Следуйте инструкциям на экране.

Основной интерфейс:

На главной странице сайта вы увидите основные вкладки: "Съедобные грибы", "Несъедобные грибы", "Ядовитые грибы", а также "О нас" и "Меры предосторожности".

Выберите нужную вкладку для перехода к соответствующей категории грибов.

Дополнительные функции:.

Навигация по разделам:

Переключайтесь между вкладками для изучения различных категорий грибов и дополнительной информации.

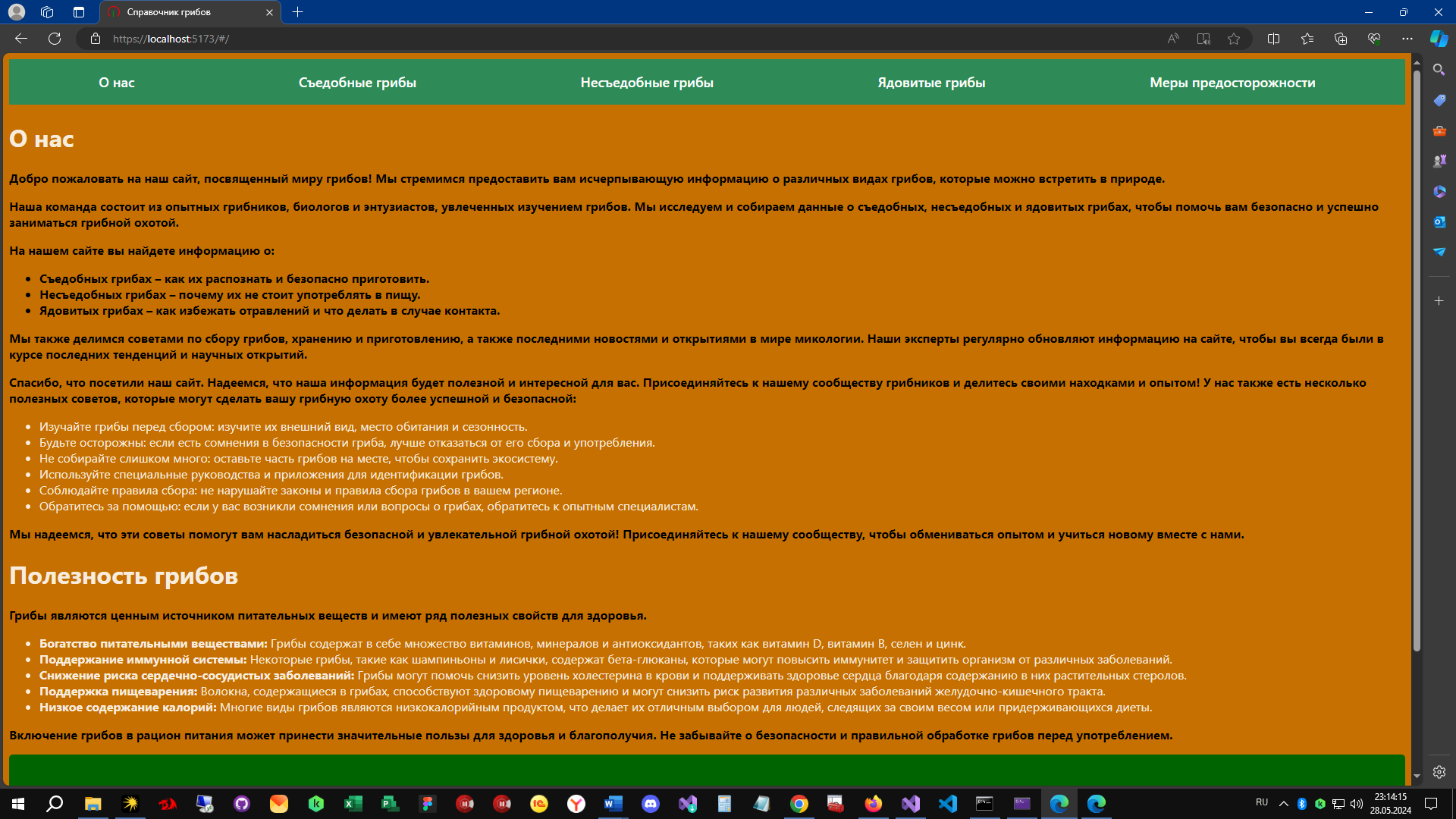


Рисунок 2 - Окно главного представления

## Состав и структура программы

Главная страница

Описание: Вводная страница, предоставляющая краткую информацию о программе и навигацию по основным разделам.

Элементы:

1. Заголовок сайта.
2. Меню навигации с основными вкладками.
3. Строка поиска.
4. Краткое приветственное сообщение или описание.
5. Вкладки с категориями грибов
   1. Съедобные грибы

Описание: Раздел, содержащий информацию о съедобных грибах.

Элементы:

Список съедобных грибов.

Каждый гриб имеет:

1. Название.
2. Картинку.
3. Краткое описание.

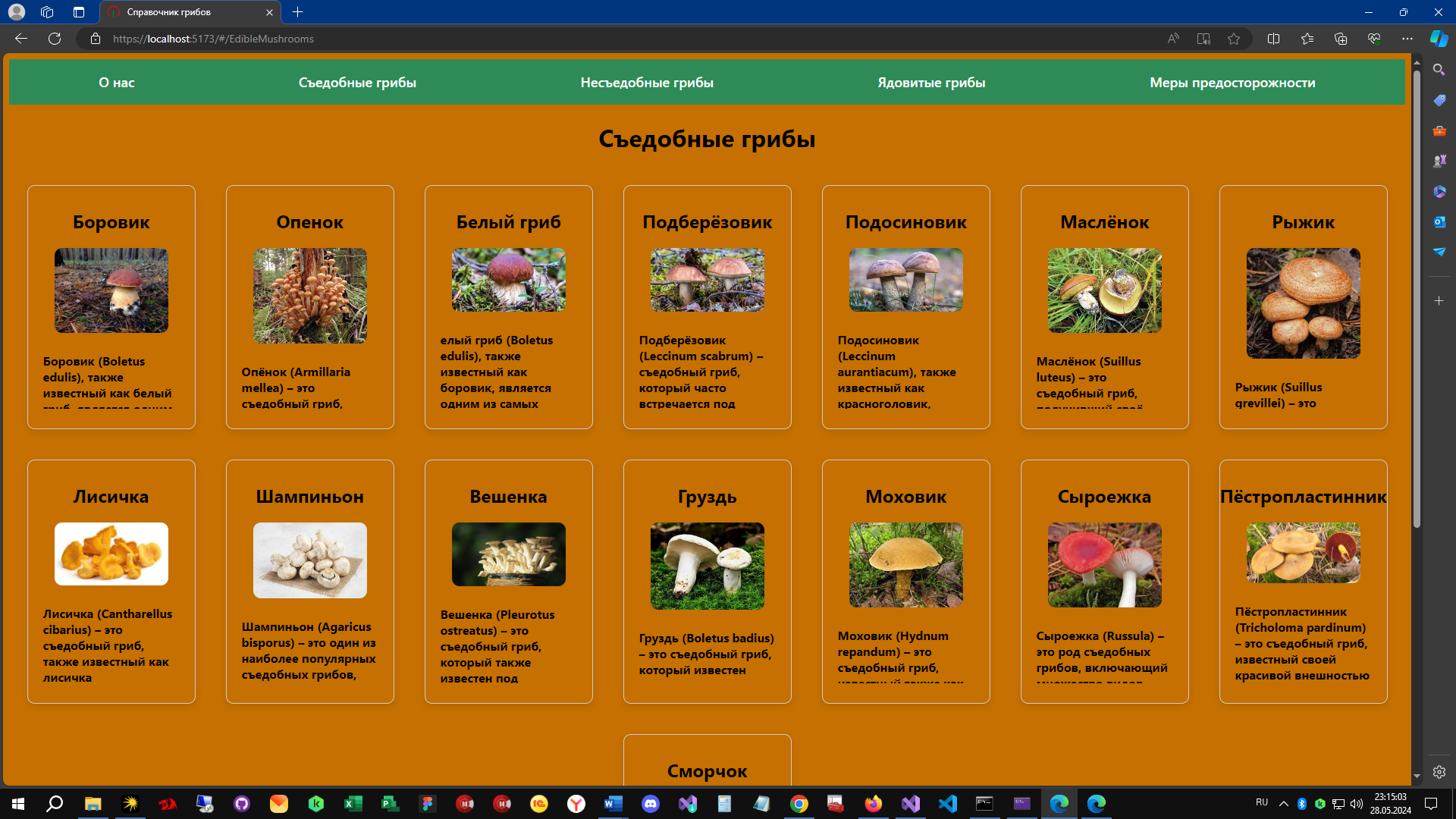


Рисунок 3 - Окно съедобные грибы

Несъедобные грибы

Описание: Раздел, содержащий информацию о несъедобных грибах.

Элементы:

Список несъедобных грибов.

Каждый гриб имеет:

* + 1. Название.
    2. Картинку.
    3. Краткое описание.

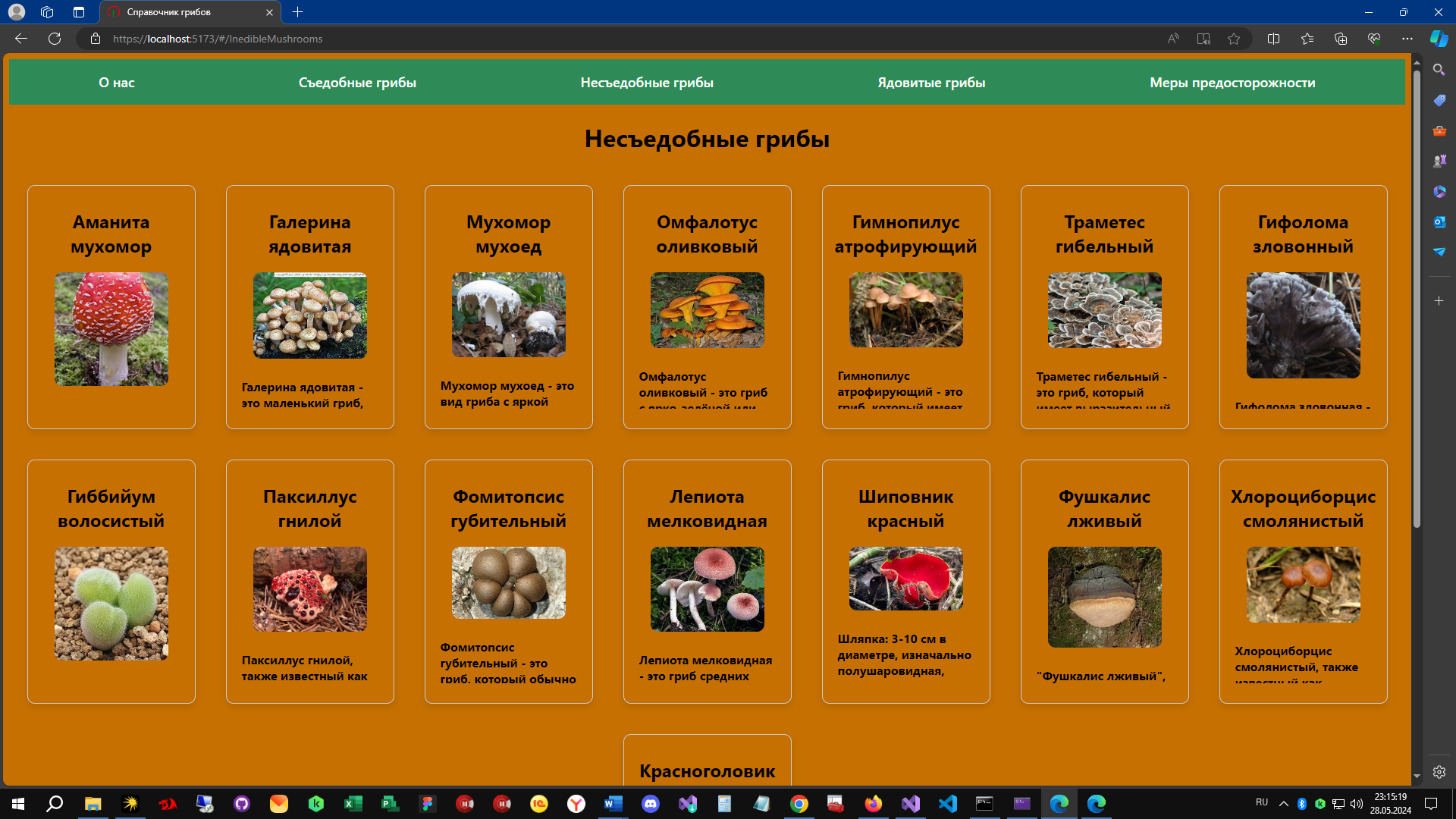


Рисунок 4 - Окно несъедобные грибы

Ядовитые грибы

Описание: Раздел, содержащий информацию о ядовитых грибах.

Элементы:

Список ядовитых грибов.

Каждый гриб имеет:

1. Название.
2. Картинку.
3. Краткое описание, включая симптомы отравления и меры первой помощи.

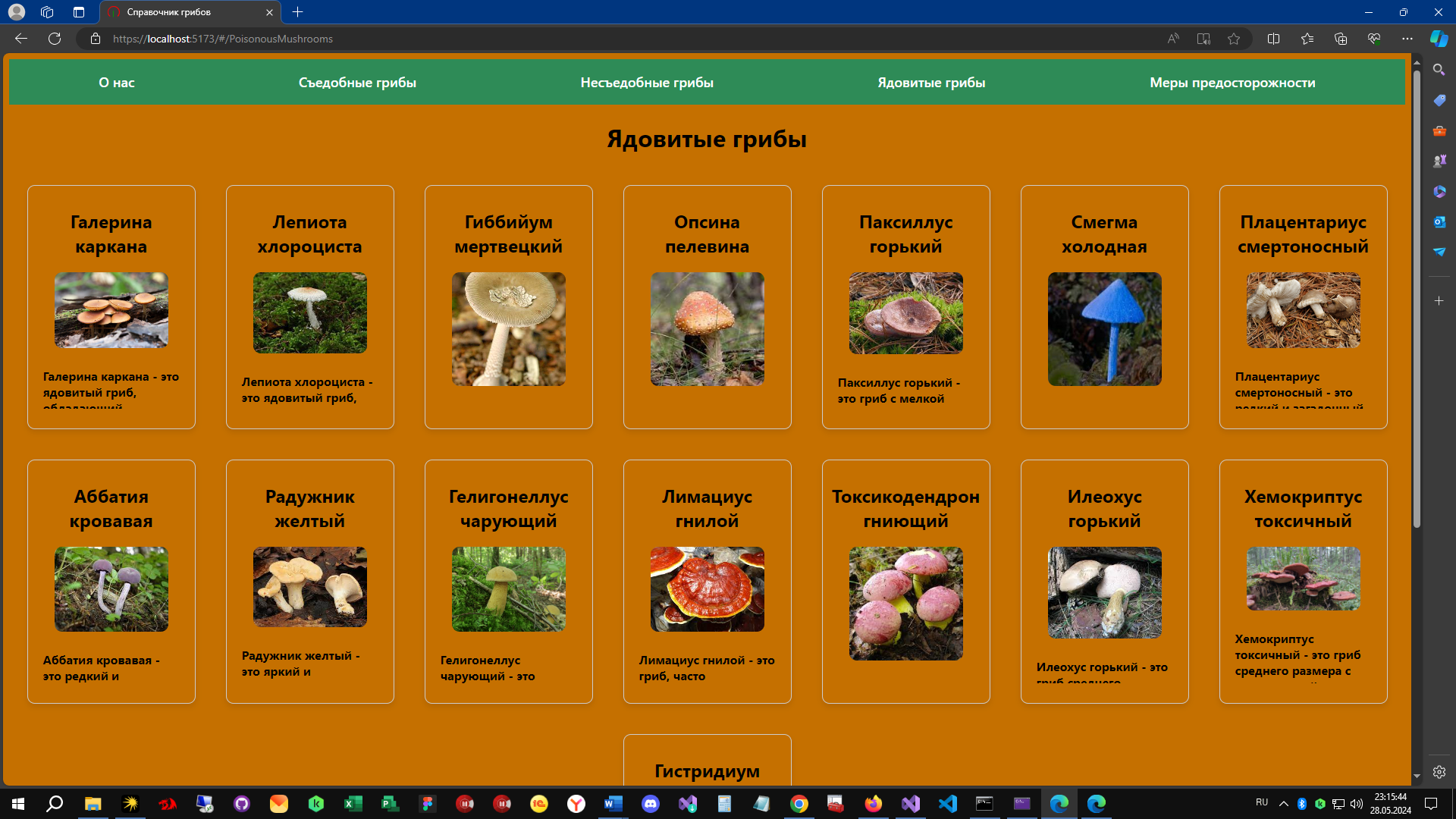


Рисунок 5 - Окно ядовитые грибы

Дополнительные вкладки

О нас

Описание: Информация о создателях программы, партнёрах и целях проекта.

Элементы:

Описание команды разработчиков.

История создания проекта.

Контактная информация.

Меры предосторожности

Описание: Рекомендации и инструкции по безопасному сбору и употреблению грибов.

Элементы:

Советы по сбору грибов.

Информация о первых признаках отравления.

Рекомендации по первой помощи при отравлении грибами.

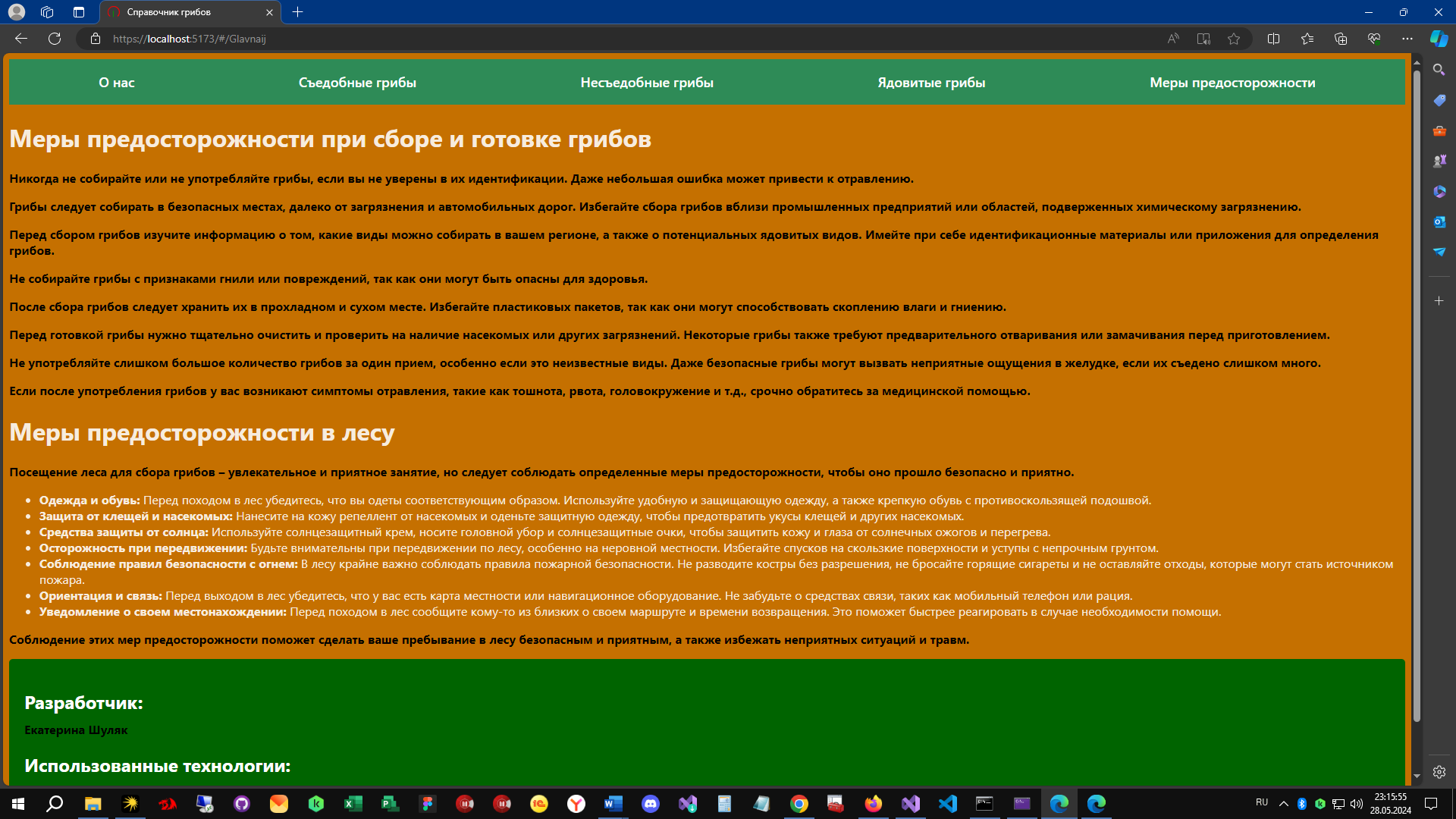


Рисунок 6 - Окно меры предосторожности

## Описание операций

1. Запуск программы

Операция: Открытие веб-приложения через браузер.

Действия пользователя:

Запустить веб-браузер.

Ввести URL-адрес веб-приложения "Справочник грибов".

Нажать Enter.

Результат: Загружается главная страница веб-приложения.

1. Навигация по разделам

Операция: Переход к различным категориям грибов и информационным страницам.

Действия пользователя:

Нажать на вкладку "Съедобные грибы", "Несъедобные грибы", "Ядовитые грибы", "О нас" или "Меры предосторожности".

Результат: Открывается соответствующая страница с информацией.

1. Просмотр информации о грибах

Операция: Просмотр детальной информации о выбранном грибе.

Действия пользователя:

Выбрать гриб из списка на странице категории или из результатов поиска.

Нажать на название гриба или изображение.

Результат: Открывается страница с подробным описанием гриба, включая название, картинку и текстовое описание.

1. Просмотр мер предосторожности

Операция: Изучение информации о мерах предосторожности при сборе и употреблении грибов.

Действия пользователя:

Нажать на вкладку "Меры предосторожности".

Результат: Открывается страница с рекомендациями по безопасному сбору и употреблению грибов, а также инструкциями по оказанию первой помощи при отравлении.

# **Руководство программиста**

Описание проекта

Программа "Справочник грибов" разработана для предоставления пользователям информации о различных видах грибов. Приложение разделено на три основные категории: съедобные грибы, несъедобные грибы и ядовитые грибы. Оно также содержит разделы "О нас" и "Меры предосторожности". Программа разработана с использованием C#, ASP.NET, JavaScript, HTML, CSS и базы данных SQL Express.

1. Технологический стек
2. Backend: C#, ASP.NET
3. Frontend: HTML, CSS, JavaScript
4. База данных: SQL Express
5. Структура проекта
6. Backend (C#, ASP.NET)
7. Controllers
8. Models
9. Views
10. Services
11. Data Access
12. Frontend (HTML, CSS, JavaScript)
13. HTML файлы
14. CSS файлы
15. JavaScript файлы
16. База данных (SQL Express)
17. Таблицы
18. Хранимые процедуры

Начальная настройка

Установите необходимые инструменты:

1. Visual Studio
2. SQL Server Express
3. Node.js (если необходимы дополнительные фронтенд-инструменты)

Настройте базу данных:

1. Установите SQL Express.
2. Создайте базу данных для хранения информации о грибах.
3. Настройте таблицы и хранимые процедуры.

## Код программы:

1. using HunterAndFisherman.Server;
2. using Microsoft.EntityFrameworkCore;
3. using System;
4. var builder = WebApplication.CreateBuilder(args);
5. // Add services to the container.
6. builder.Services.AddControllers();
7. // Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle
8. builder.Services.AddEndpointsApiExplorer();
9. builder.Services.AddSwaggerGen();
10. builder.Services.AddDbContext<AppDbContext>(options =>
11. options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));
12. var app = builder.Build();
13. app.UseDefaultFiles();
14. app.UseStaticFiles();
15. app.UseCors(options =>
16. {
17. options.AllowAnyOrigin();
18. options.AllowAnyMethod();
19. options.AllowAnyHeader();
20. });
21. // Configure the HTTP request pipeline.
22. if (app.Environment.IsDevelopment())
23. {
24. app.UseSwagger();
25. app.UseSwaggerUI();
26. }
27. app.UseHttpsRedirection();
28. app.UseAuthorization();
29. app.MapControllers();
30. app.MapFallbackToFile("/index.html");
31. app.Run();
32. namespace HunterAndFisherman.Server
33. {
34. public class WeatherForecast
35. {
36. public DateOnly Date { get; set; }
37. public int TemperatureC { get; set; }
38. public int TemperatureF => 32 + (int)(TemperatureC / 0.5556);
39. public string? Summary { get; set; }
40. }
41. }
42. using HunterAndFisherman.Server.Models;
43. using Microsoft.EntityFrameworkCore;
44. namespace HunterAndFisherman.Server
45. {
46. public class AppDbContext : DbContext
47. {
48. public AppDbContext(DbContextOptions<AppDbContext> options) : base(options) { }
49. public DbSet<SiedGrib> EdibleMushrooms { get; set; }
50. public DbSet<Nesied> Nesied { get; set; }
51. public DbSet<Iadov> Iadov { get; set; }
53. }
54. }
55. using HunterAndFisherman.Server.Models;
56. using Microsoft.AspNetCore.Mvc;
57. using Microsoft.EntityFrameworkCore;
58. using System;
59. using System.Threading.Tasks;
60. namespace HunterAndFisherman.Server.Controllers
61. {
62. [ApiController]
63. [Route("[controller]")]
64. [Produces("application/json")] // Указываем, что контроллер производит JSON-данные
65. public class IadovGribController : ControllerBase
66. {
67. private readonly AppDbContext \_context;
68. public IadovGribController(AppDbContext context)
69. {
70. \_context = context;
71. }
72. [HttpGet]
73. public async Task<IActionResult> GetSiedGrib()
74. {
75. try
76. {
77. var entities = await \_context.Iadov.ToListAsync();
78. return Ok(entities);
79. }
80. catch (Exception ex)
81. {
82. return StatusCode(500, $"Internal server error: {ex.Message}");
83. }
84. }
85. }
86. }

using Microsoft.AspNetCore.Hosting;

using Microsoft.AspNetCore.Mvc;

using System.IO;

namespace HunterAndFisherman.Server.Controllers

{

[ApiController]

[Route("[controller]")]

public class ImagesController : ControllerBase

{

private readonly IWebHostEnvironment \_env;

public ImagesController(IWebHostEnvironment env)

{

\_env = env;

}

[HttpGet("{imageName}")]

public IActionResult GetImage(string imageName)

{

var imageFilePath = Path.Combine(\_env.ContentRootPath, "Images", imageName);

if (System.IO.File.Exists(imageFilePath))

{

var imageBytes = System.IO.File.ReadAllBytes(imageFilePath);

return File(imageBytes, "image/jpeg"); // Укажите соответствующий MIME-тип вашим изображениям

}

else

{

return NotFound();

}

}

}

1. }

using HunterAndFisherman.Server.Models;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using System;

using System.Threading.Tasks;

namespace HunterAndFisherman.Server.Controllers

{

[ApiController]

[Route("[controller]")]

[Produces("application/json")] // Указываем, что контроллер производит JSON-данные

public class NesiedGribController : ControllerBase

{

private readonly AppDbContext \_context;

public NesiedGribController(AppDbContext context)

{

\_context = context;

}

[HttpGet]

public async Task<IActionResult> GetSiedGrib()

{

try

{

var entities = await \_context.Nesied.ToListAsync();

return Ok(entities);

}

catch (Exception ex)

{

return StatusCode(500, $"Internal server error: {ex.Message}");

}

}

}

1. }

using HunterAndFisherman.Server.Models;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using System;

using System.Threading.Tasks;

namespace HunterAndFisherman.Server.Controllers

{

[ApiController]

[Route("[controller]")]

[Produces("application/json")] // Указываем, что контроллер производит JSON-данные

public class SiedGribController : ControllerBase

{

private readonly AppDbContext \_context;

public SiedGribController(AppDbContext context)

{

\_context = context;

}

[HttpGet]

public async Task<IActionResult> GetSiedGrib()

{

try

{

var entities = await \_context.EdibleMushrooms.ToListAsync();

return Ok(entities);

}

catch (Exception ex)

{

return StatusCode(500, $"Internal server error: {ex.Message}");

}

}

}

1. }

using Microsoft.AspNetCore.Mvc;

namespace HunterAndFisherman.Server.Controllers

{

[ApiController]

[Route("[controller]")]

public class WeatherForecastController : ControllerBase

{

private static readonly string[] Summaries = new[]

{

"Freezing", "Bracing", "Chilly", "Cool", "Mild", "Warm", "Balmy", "Hot", "Sweltering", "Scorching"

};

private readonly ILogger<WeatherForecastController> \_logger;

public WeatherForecastController(ILogger<WeatherForecastController> logger)

{

\_logger = logger;

}

[HttpGet(Name = "GetWeatherForecast")]

public IEnumerable<WeatherForecast> Get()

{

return Enumerable.Range(1, 5).Select(index => new WeatherForecast

{

Date = DateOnly.FromDateTime(DateTime.Now.AddDays(index)),

TemperatureC = Random.Shared.Next(-20, 55),

Summary = Summaries[Random.Shared.Next(Summaries.Length)]

})

.ToArray();

}

}

1. } import React from 'react';
2. import { HashRouter, Routes, Route, NavLink } from 'react-router-dom';
3. class About extends React.Component {
4. constructor(props){
5. super(props);
6. }
7. render() {
8. return (
9. <div className='About'>
10. <h1>О нас</h1>
11. <p>
12. Добро пожаловать на наш сайт, посвященный миру грибов! Мы стремимся предоставить вам
13. исчерпывающую информацию о различных видах грибов, которые можно встретить в природе.
14. </p>
15. <p>
16. Наша команда состоит из опытных грибников, биологов и энтузиастов, увлеченных изучением
17. грибов. Мы исследуем и собираем данные о съедобных, несъедобных и ядовитых грибах, чтобы
18. помочь вам безопасно и успешно заниматься грибной охотой.
19. </p>
20. <p>
21. На нашем сайте вы найдете информацию о:
22. <ul>
23. <li>Съедобных грибах – как их распознать и безопасно приготовить.</li>
24. <li>Несъедобных грибах – почему их не стоит употреблять в пищу.</li>
25. <li>Ядовитых грибах – как избежать отравлений и что делать в случае контакта.</li>
26. </ul>
27. </p>
28. <p>
29. Мы также делимся советами по сбору грибов, хранению и приготовлению, а также последними
30. новостями и открытиями в мире микологии. Наши эксперты регулярно обновляют информацию на сайте,
31. чтобы вы всегда были в курсе последних тенденций и научных открытий.
32. </p>
33. <p>
34. Спасибо, что посетили наш сайт. Надеемся, что наша информация будет полезной и
35. интересной для вас. Присоединяйтесь к нашему сообществу грибников и делитесь своими
36. находками и опытом! У нас также есть несколько полезных советов, которые могут
37. сделать вашу грибную охоту более успешной и безопасной:
38. </p>
39. <ul>
40. <li>Изучайте грибы перед сбором: изучите их внешний вид, место обитания и сезонность.</li>
41. <li>Будьте осторожны: если есть сомнения в безопасности гриба, лучше отказаться от его сбора и употребления.</li>
42. <li>Не собирайте слишком много: оставьте часть грибов на месте, чтобы сохранить экосистему.</li>
43. <li>Используйте специальные руководства и приложения для идентификации грибов.</li>
44. <li>Соблюдайте правила сбора: не нарушайте законы и правила сбора грибов в вашем регионе.</li>
45. <li>Обратитесь за помощью: если у вас возникли сомнения или вопросы о грибах, обратитесь к опытным специалистам.</li>
46. </ul>
47. <p>
48. Мы надеемся, что эти советы помогут вам насладиться безопасной и увлекательной грибной охотой!
49. Присоединяйтесь к нашему сообществу, чтобы обмениваться опытом и учиться новому вместе с нами.
50. </p>
51. <h1>Полезность грибов</h1>
52. <p>Грибы являются ценным источником питательных веществ и имеют ряд полезных свойств для здоровья.</p>
53. <ul>
54. <li><strong>Богатство питательными веществами:</strong> Грибы содержат в себе множество витаминов, минералов и антиоксидантов, таких как витамин D, витамин В, селен и цинк.</li>
55. <li><strong>Поддержание иммунной системы:</strong> Некоторые грибы, такие как шампиньоны и лисички, содержат бета-глюканы, которые могут повысить иммунитет и защитить организм от различных заболеваний.</li>
56. <li><strong>Снижение риска сердечно-сосудистых заболеваний:</strong> Грибы могут помочь снизить уровень холестерина в крови и поддерживать здоровье сердца благодаря содержанию в них растительных стеролов.</li>
57. <li><strong>Поддержка пищеварения:</strong> Волокна, содержащиеся в грибах, способствуют здоровому пищеварению и могут снизить риск развития различных заболеваний желудочно-кишечного тракта.</li>
58. <li><strong>Низкое содержание калорий:</strong> Многие виды грибов являются низкокалорийным продуктом, что делает их отличным выбором для людей, следящих за своим весом или придерживающихся диеты.</li>
59. </ul>
60. <p>Включение грибов в рацион питания может принести значительные пользы для здоровья и благополучия. Не забывайте о безопасности и правильной обработке грибов перед употреблением.</p>
61. </div>
62. );
63. }
64. }
65. export default About;
66. import React from 'react';
67. import axios from 'axios';
68. import './EdibleMushrooms.css';
69. class EdibleMushrooms extends React.Component {
70. constructor(props) {
71. super(props);
72. this.state = {
73. mushrooms: [],
74. loading: true,
75. error: null,
76. expandedMushroomId: null // Добавляем состояние для отслеживания раскрытого гриба
77. };
78. }
79. componentDidMount() {
80. axios.get('https://localhost:7199/SiedGrib')
81. .then(response => {
82. if (Array.isArray(response.data)) {
83. this.setState({ mushrooms: response.data, loading: false });
84. } else {
85. console.error('Expected array but got:', typeof response.data, response.data);
86. this.setState({ error: 'Invalid data format', loading: false });
87. }
88. })
89. .catch(error => {
90. console.error('Error fetching data:', error);
91. this.setState({ error: error.message, loading: false });
92. });
93. }
94. toggleExpand = (id) => {
95. this.setState(prevState => ({
96. expandedMushroomId: prevState.expandedMushroomId === id ? null : id
97. }));
98. };
99. render() {
100. const { mushrooms, loading, error, expandedMushroomId } = this.state;
101. if (loading) {
102. return <div className='EdibleMushrooms'>Loading...</div>;
103. }
104. if (error) {
105. return <div className='EdibleMushrooms'>Error: {error}</div>;
106. }
107. return (
108. <div className='EdibleMushrooms'>
109. <h1>Съедобные грибы</h1>
110. <div className='mushroom-list'>
111. {mushrooms.map(mushroom => (
112. <div
113. key={mushroom.id}
114. className={`mushroom-item ${expandedMushroomId === mushroom.id ? 'expanded' : ''}`}
115. onClick={() => this.toggleExpand(mushroom.id)}
116. >
117. <h2>{mushroom.name}</h2>
119. {mushroom.img && (
120. <img
121. src={`https://localhost:7199/Images/${mushroom.img}.jpg`}
122. alt={mushroom.name}
123. className='mushroom-image'
124. />
125. )}
126. <p className='mushroom-discripsion'>{mushroom.description}</p>
127. </div>
128. ))}
129. </div>
130. </div>
131. );
132. }
133. }
134. export default EdibleMushrooms;
135. import React from 'react';
136. class Glavnaij extends React.Component {
137. render() {
138. return (
139. <div className='Glavnaij'>
140. <h1>Меры предосторожности при сборе и готовке грибов</h1>
141. <p>
142. Никогда не собирайте или не употребляйте грибы, если вы не уверены в их идентификации.
143. Даже небольшая ошибка может привести к отравлению.
144. </p>
145. <p>
146. Грибы следует собирать в безопасных местах, далеко от загрязнения и автомобильных дорог.
147. Избегайте сбора грибов вблизи промышленных предприятий или областей, подверженных химическому загрязнению.
148. </p>
149. <p>
150. Перед сбором грибов изучите информацию о том, какие виды можно собирать в вашем регионе,
151. а также о потенциальных ядовитых видов. Имейте при себе идентификационные материалы или
152. приложения для определения грибов.
153. </p>
154. <p>
155. Не собирайте грибы с признаками гнили или повреждений, так как они могут быть опасны для здоровья.
156. </p>
157. <p>
158. После сбора грибов следует хранить их в прохладном и сухом месте. Избегайте пластиковых
159. пакетов, так как они могут способствовать скоплению влаги и гниению.
160. </p>
161. <p>
162. Перед готовкой грибы нужно тщательно очистить и проверить на наличие насекомых или других загрязнений.
163. Некоторые грибы также требуют предварительного отваривания или замачивания перед приготовлением.
164. </p>
165. <p>
166. Не употребляйте слишком большое количество грибов за один прием, особенно если это неизвестные виды.
167. Даже безопасные грибы могут вызвать неприятные ощущения в желудке, если их съедено слишком много.
168. </p>
169. <p>
170. Если после употребления грибов у вас возникают симптомы отравления, такие как тошнота, рвота, головокружение и т.д.,
171. срочно обратитесь за медицинской помощью.
172. </p>
173. <h1>Меры предосторожности в лесу</h1>
174. <p>Посещение леса для сбора грибов – увлекательное и приятное занятие, но следует соблюдать определенные меры предосторожности, чтобы оно прошло безопасно и приятно.</p>
175. <ul>
176. <li><strong>Одежда и обувь:</strong> Перед походом в лес убедитесь, что вы одеты соответствующим образом. Используйте удобную и защищающую одежду, а также крепкую обувь с противоскользящей подошвой.</li>
177. <li><strong>Защита от клещей и насекомых:</strong> Нанесите на кожу репеллент от насекомых и оденьте защитную одежду, чтобы предотвратить укусы клещей и других насекомых.</li>
178. <li><strong>Средства защиты от солнца:</strong> Используйте солнцезащитный крем, носите головной убор и солнцезащитные очки, чтобы защитить кожу и глаза от солнечных ожогов и перегрева.</li>
179. <li><strong>Осторожность при передвижении:</strong> Будьте внимательны при передвижении по лесу, особенно на неровной местности. Избегайте спусков на скользкие поверхности и уступы с непрочным грунтом.</li>
180. <li><strong>Соблюдение правил безопасности с огнем:</strong> В лесу крайне важно соблюдать правила пожарной безопасности. Не разводите костры без разрешения, не бросайте горящие сигареты и не оставляйте отходы, которые могут стать источником пожара.</li>
181. <li><strong>Ориентация и связь:</strong> Перед выходом в лес убедитесь, что у вас есть карта местности или навигационное оборудование. Не забудьте о средствах связи, таких как мобильный телефон или рация.</li>
182. <li><strong>Уведомление о своем местонахождении:</strong> Перед походом в лес сообщите кому-то из близких о своем маршруте и времени возвращения. Это поможет быстрее реагировать в случае необходимости помощи.</li>
183. </ul>
184. <p>Соблюдение этих мер предосторожности поможет сделать ваше пребывание в лесу безопасным и приятным, а также избежать неприятных ситуаций и травм.</p>
185. </div>
186. );
187. }
188. }
189. export default Glavnaij;
190. import React from 'react';
191. import axios from 'axios';
192. import '../EdibleMushrooms/EdibleMushrooms.css';
193. class InedibleMushrooms extends React.Component {
194. constructor(props) {
195. super(props);
196. this.state = {
197. mushrooms: [],
198. loading: true,
199. error: null,
200. expandedMushroomId: null // Добавляем состояние для отслеживания раскрытого гриба
201. };
202. }
203. componentDidMount() {
204. axios.get('https://localhost:7199/NesiedGrib')
205. .then(response => {
206. if (Array.isArray(response.data)) {
207. this.setState({ mushrooms: response.data, loading: false });
208. } else {
209. console.error('Expected array but got:', typeof response.data, response.data);
210. this.setState({ error: 'Invalid data format', loading: false });
211. }
212. })
213. .catch(error => {
214. console.error('Error fetching data:', error);
215. this.setState({ error: error.message, loading: false });
216. });
217. }
218. toggleExpand = (id) => {
219. this.setState(prevState => ({
220. expandedMushroomId: prevState.expandedMushroomId === id ? null : id
221. }));
222. };
223. render() {
224. const { mushrooms, loading, error, expandedMushroomId } = this.state;
225. if (loading) {
226. return <div className='EdibleMushrooms'>Loading...</div>;
227. }
228. if (error) {
229. return <div className='EdibleMushrooms'>Error: {error}</div>;
230. }
231. return (
232. <div className='EdibleMushrooms'>
233. <h1>Несъедобные грибы</h1>
234. <div className='mushroom-list'>
235. {mushrooms.map(mushroom => (
236. <div
237. key={mushroom.id}
238. className={`mushroom-item ${expandedMushroomId === mushroom.id ? 'expanded' : ''}`}
239. onClick={() => this.toggleExpand(mushroom.id)}
240. >
241. <h2>{mushroom.name}</h2>
243. {mushroom.img && (
244. <img
245. src={`https://localhost:7199/Images/${mushroom.img}.jpg`}
246. alt={mushroom.name}
247. className='mushroom-image'
248. />
249. )}
250. <p className='mushroom-discripsion'>{mushroom.description}</p>
251. </div>
252. ))}
253. </div>
254. </div>
255. );
256. }
257. }
258. export default InedibleMushrooms;
259. import React from 'react';
260. import axios from 'axios';
261. import '../EdibleMushrooms/EdibleMushrooms.css';
262. class PoisonousMushrooms extends React.Component {
263. constructor(props) {
264. super(props);
265. this.state = {
266. mushrooms: [],
267. loading: true,
268. error: null,
269. expandedMushroomId: null // Добавляем состояние для отслеживания раскрытого гриба
270. };
271. }
272. componentDidMount() {
273. axios.get('https://localhost:7199/IadovGrib')
274. .then(response => {
275. if (Array.isArray(response.data)) {
276. this.setState({ mushrooms: response.data, loading: false });
277. } else {
278. console.error('Expected array but got:', typeof response.data, response.data);
279. this.setState({ error: 'Invalid data format', loading: false });
280. }
281. })
282. .catch(error => {
283. console.error('Error fetching data:', error);
284. this.setState({ error: error.message, loading: false });
285. });
286. }
287. toggleExpand = (id) => {
288. this.setState(prevState => ({
289. expandedMushroomId: prevState.expandedMushroomId === id ? null : id
290. }));
291. };
292. render() {
293. const { mushrooms, loading, error, expandedMushroomId } = this.state;
294. if (loading) {
295. return <div className='EdibleMushrooms'>Loading...</div>;
296. }
297. if (error) {
298. return <div className='EdibleMushrooms'>Error: {error}</div>;
299. }
300. return (
301. <div className='EdibleMushrooms'>
302. <h1>Ядовитые грибы</h1>
303. <div className='mushroom-list'>
304. {mushrooms.map(mushroom => (
305. <div
306. key={mushroom.id}
307. className={`mushroom-item ${expandedMushroomId === mushroom.id ? 'expanded' : ''}`}
308. onClick={() => this.toggleExpand(mushroom.id)}
309. >
310. <h2>{mushroom.name}</h2>
312. {mushroom.img && (
313. <img
314. src={`https://localhost:7199/Images/${mushroom.img}.jpg`}
315. alt={mushroom.name}
316. className='mushroom-image'
317. />
318. )}
319. <p className='mushroom-discripsion'>{mushroom.description}</p>
320. </div>
321. ))}
322. </div>
323. </div>
324. );
325. }
326. }
327. export default PoisonousMushrooms;
328. import React from 'react';
329. import { HashRouter, Routes, Route, NavLink } from 'react-router-dom';
330. class Header extends React.Component {
331. constructor(props){
332. super(props);
333. }
334. render() {
335. return (
336. <div className='Header'>
337. <div className='Header-panel-block'>
338. <NavLink className="Header-panel-block-link" to='/'>О нас</NavLink>
339. </div>
340. <div className='Header-panel-block'>
341. <NavLink className="Header-panel-block-link" to='/EdibleMushrooms'>Съедобные грибы</NavLink>
342. </div>
343. <div className='Header-panel-block'>
344. <NavLink className="Header-panel-block-link" to='/InedibleMushrooms'>Несъедобные грибы</NavLink>
345. </div>
346. <div className='Header-panel-block'>
347. <NavLink className="Header-panel-block-link" to='/PoisonousMushrooms'>Ядовитые грибы</NavLink>
348. </div>
349. <div className='Header-panel-block'>
350. <NavLink className="Header-panel-block-link" to='/Glavnaij'>Меры предосторожности</NavLink>
351. </div>
352. </div>
353. );
354. }
355. }
356. export default Header;
357. #root {
359. }
360. .Header{
361. background-color: green;
362. height: 60px; /\* стандартная высота для хедера \*/
363. width: 100%; /\* хедер занимает всю ширину экрана \*/
364. display: flex;
365. align-items: center; /\* выравнивание по вертикали \*/
366. justify-content: space-between; /\* выравнивание элементов внутри по горизонтали \*/
367. padding: 0 20px; /\* внутренние отступы слева и справа \*/
368. box-sizing: border-box; /\* учитываем паддинги в общей ширине \*/
369. }
370. .Mine{
371. padding: 20px;
372. box-sizing: border-box;
373. background-color: #f0f0f0; /\* светлый фон для области под хедером \*/
374. min-height: calc(100vh - 60px);
375. color: black;
376. }
377. .futher{
378. background-color: #b8860b; /\* золотисто-коричневый цвет \*/
379. color: white;
380. text-align: center;
381. padding: 50px;
382. position: relative;
383. bottom: 0;
384. width: 100%;
385. box-sizing: border-box;
386. }
387. /\*Далее хедер\*/
388. /\* styles.css \*/
389. .Header {
390. display: flex;
391. justify-content: space-around;
392. background-color: #2E8B57; /\* Цвета лесной зелени \*/
393. padding: 10px;
394. }
396. .Header-panel-block {
397. padding: 10px;
398. }
400. .Header-panel-block-link {
401. text-decoration: none;
402. color: white; /\* Цвет текста \*/
403. font-size: 18px;
404. transition: all 0.3s ease;
405. }
407. .Header-panel-block-link:hover {
408. color: #FFD700; /\* Цвет свечения при наведении \*/
409. text-shadow: 0 0 10px #FFD700; /\* Эффект свечения \*/
410. }
411. p{
412. font-weight: 700;
413. color: black;
414. }
415. .Futher {
416. background-color: #006400; /\* Темно-зеленый цвет фона \*/
417. color: #fff; /\* Белый цвет текста \*/
418. padding: 20px; /\* Отступы внутри блока \*/
419. border-radius: 5px; /\* Закругление углов \*/
420. }
421. .Futher h2 {
422. margin-bottom: 10px; /\* Отступ снизу для заголовков \*/
423. }
424. .Futher p {
425. margin: 0; /\* Убираем внешние отступы у абзацев \*/
426. }
427. import './App.css';
428. import React from 'react';
429. import { HashRouter, Routes, Route, NavLink } from 'react-router-dom';
430. import Header from './Component/Header';
431. import About from './Component/About/About';
432. import EdibleMushrooms from './Component/EdibleMushrooms/EdibleMushrooms';
433. import Glavnaij from './Component/Glavnaij/Glavnaij';
434. import InedibleMushrooms from './Component/InedibleMushrooms/InedibleMushrooms';
435. import PoisonousMushrooms from './Component/PoisonousMushrooms/PoisonousMushrooms';
436. import Futher from './Component/Futher';
437. class Content extends React.Component {
438. constructor(props){
439. super(props);
440. }
441. render() {
442. return (
443. <div className='Content'>
444. <HashRouter>
445. <Header></Header>
446. <Routes>
447. <Route path="/" element={<About />} />
448. <Route path="/EdibleMushrooms" element={<EdibleMushrooms />} />
449. <Route path="/Glavnaij" element={<Glavnaij />} />
450. <Route path="/InedibleMushrooms" element={<InedibleMushrooms />} />
451. <Route path="/PoisonousMushrooms" element={<PoisonousMushrooms />} />
452. </Routes>
453. <Futher></Futher>
454. </HashRouter>
455. </div>
456. );
457. }
458. }
459. export default Content;
460. componentDidMount() {
461. axios.get('https://localhost:7199/IadovGrib')
462. .then(response => {
463. if (Array.isArray(response.data)) {
464. this.setState({ mushrooms: response.data, loading: false });
465. } else {
466. console.error('Expected array but got:', typeof response.data, response.data);
467. this.setState({ error: 'Invalid data format', loading: false });
468. }
469. })
470. .catch(error => {
471. console.error('Error fetching data:', error);
472. this.setState({ error: error.message, loading: false });
473. });
474. }
475. toggleExpand = (id) => {
476. this.setState(prevState => ({
477. expandedMushroomId: prevState.expandedMushroomId === id ? null : id
478. }));
479. };
480. render() {
481. const { mushrooms, loading, error, expandedMushroomId } = this.state;
482. if (loading) {
483. return <div className='EdibleMushrooms'>Loading...</div>;
484. }
485. if (error) {
486. return <div className='EdibleMushrooms'>Error: {error}</div>;
487. }
488. return (
489. <div className='EdibleMushrooms'>
490. <h1>Ядовитые грибы</h1>
491. <div className='mushroom-list'>
492. {mushrooms.map(mushroom => (
493. <div
494. key={mushroom.id}
495. className={`mushroom-item ${expandedMushroomId === mushroom.id ? 'expanded' : ''}`}
496. onClick={() => this.toggleExpand(mushroom.id)}
497. >
498. <h2>{mushroom.name}</h2>
500. {mushroom.img && (
501. <img
502. src={`https://localhost:7199/Images/${mushroom.img}.jpg`}
503. alt={mushroom.name}
504. className='mushroom-image'
505. />
506. )}
507. <p className='mushroom-discripsion'>{mushroom.description}</p>
508. </div>
509. ))}
510. </div>
511. </div>
512. );
513. }
514. }
515. export default PoisonousMushrooms;
516. import React from 'react';
517. import { HashRouter, Routes, Route, NavLink } from 'react-router-dom';
518. class Header extends React.Component {
519. constructor(props){
520. super(props);
521. }
522. render() {
523. return (
524. <div className='Header'>
525. <div className='Header-panel-block'>
526. <NavLink className="Header-panel-block-link" to='/'>О нас</NavLink>
527. </div>
528. <div className='Header-panel-block'>
529. <NavLink className="Header-panel-block-link" to='/EdibleMushrooms'>Съедобные грибы</NavLink>
530. </div>
531. <div className='Header-panel-block'>
532. <NavLink className="Header-panel-block-link" to='/InedibleMushrooms'>Несъедобные грибы</NavLink>
533. </div>
534. <div className='Header-panel-block'>
535. <NavLink className="Header-panel-block-link" to='/PoisonousMushrooms'>Ядовитые грибы</NavLink>
536. </div>
537. <div className='Header-panel-block'>
538. <NavLink className="Header-panel-block-link" to='/Glavnaij'>Меры предосторожности</NavLink>
539. </div>
540. </div>
541. );
542. }
543. }
544. export default Header;
545. #root {
547. }
548. .Header{
549. background-color: green;
550. height: 60px; /\* стандартная высота для хедера \*/
551. width: 100%; /\* хедер занимает всю ширину экрана \*/
552. display: flex;
553. align-items: center; /\* выравнивание по вертикали \*/
554. justify-content: space-between; /\* выравнивание элементов внутри по горизонтали \*/
555. padding: 0 20px; /\* внутренние отступы слева и справа \*/
556. box-sizing: border-box; /\* учитываем паддинги в общей ширине \*/
557. }
558. .Mine{
559. padding: 20px;
560. box-sizing: border-box;
561. background-color: #f0f0f0; /\* светлый фон для области под хедером \*/
562. min-height: calc(100vh - 60px);
563. color: black;
564. }
565. .futher{
566. background-color: #b8860b; /\* золотисто-коричневый цвет \*/
567. color: white;
568. text-align: center;
569. padding: 50px;
570. position: relative;
571. bottom: 0;
572. width: 100%;
573. box-sizing: border-box;
574. }
575. /\*Далее хедер\*/
576. /\* styles.css \*/
577. .Header {
578. display: flex;
579. justify-content: space-around;
580. background-color: #2E8B57; /\* Цвета лесной зелени \*/
581. padding: 10px;
582. }
584. .Header-panel-block {
585. padding: 10px;
586. }
588. .Header-panel-block-link {
589. text-decoration: none;
590. color: white; /\* Цвет текста \*/
591. font-size: 18px;
592. transition: all 0.3s ease;
593. }
595. .Header-panel-block-link:hover {
596. color: #FFD700; /\* Цвет свечения при наведении \*/
597. text-shadow: 0 0 10px #FFD700; /\* Эффект свечения \*/
598. }
599. p{
600. font-weight: 700;
601. color: black;
602. }
603. .Futher {
604. background-color: #006400; /\* Темно-зеленый цвет фона \*/
605. color: #fff; /\* Белый цвет текста \*/
606. padding: 20px; /\* Отступы внутри блока \*/
607. border-radius: 5px; /\* Закругление углов \*/
608. }
609. .Futher h2 {
610. margin-bottom: 10px; /\* Отступ снизу для заголовков \*/
611. }
612. .Futher p {
613. margin: 0; /\* Убираем внешние отступы у абзацев \*/
614. }
615. import './App.css';
616. import React from 'react';
617. import { HashRouter, Routes, Route, NavLink } from 'react-router-dom';
618. import Header from './Component/Header';
619. import About from './Component/About/About';
620. import EdibleMushrooms from './Component/EdibleMushrooms/EdibleMushrooms';
621. import Glavnaij from './Component/Glavnaij/Glavnaij';
622. import InedibleMushrooms from './Component/InedibleMushrooms/InedibleMushrooms';
623. import PoisonousMushrooms from './Component/PoisonousMushrooms/PoisonousMushrooms';
624. import Futher from './Component/Futher';
625. class Content extends React.Component {
626. constructor(props){
627. super(props);
628. }
629. render() {
630. return (
631. <div className='Content'>
632. <HashRouter>
633. <Header></Header>
634. <Routes>
635. <Route path="/" element={<About />} />
636. <Route path="/EdibleMushrooms" element={<EdibleMushrooms />} />
637. <Route path="/Glavnaij" element={<Glavnaij />} />
638. <Route path="/InedibleMushrooms" element={<InedibleMushrooms />} />
639. <Route path="/PoisonousMushrooms" element={<PoisonousMushrooms />} />
640. </Routes>
641. <Futher></Futher>
642. </HashRouter>
643. </div>
644. );
645. }
646. }
647. export default Content;
648. componentDidMount() {
649. axios.get('https://localhost:7199/IadovGrib')
650. .then(response => {
651. if (Array.isArray(response.data)) {
652. this.setState({ mushrooms: response.data, loading: false });
653. } else {
654. console.error('Expected array but got:', typeof response.data, response.data);
655. this.setState({ error: 'Invalid data format', loading: false });
656. }
657. })
658. .catch(error => {
659. console.error('Error fetching data:', error);
660. this.setState({ error: error.message, loading: false });
661. });
662. }
663. toggleExpand = (id) => {
664. this.setState(prevState => ({
665. expandedMushroomId: prevState.expandedMushroomId === id ? null : id
666. }));
667. };
668. render() {
669. const { mushrooms, loading, error, expandedMushroomId } = this.state;
670. if (loading) {
671. return <div className='EdibleMushrooms'>Loading...</div>;
672. }
673. if (error) {
674. return <div className='EdibleMushrooms'>Error: {error}</div>;
675. }
676. return (
677. <div className='EdibleMushrooms'>
678. <h1>Ядовитые грибы</h1>
679. <div className='mushroom-list'>
680. {mushrooms.map(mushroom => (
681. <div
682. key={mushroom.id}
683. className={`mushroom-item ${expandedMushroomId === mushroom.id ? 'expanded' : ''}`}
684. onClick={() => this.toggleExpand(mushroom.id)}
685. >
686. <h2>{mushroom.name}</h2>
688. {mushroom.img && (
689. <img
690. src={`https://localhost:7199/Images/${mushroom.img}.jpg`}
691. alt={mushroom.name}
692. className='mushroom-image'
693. />
694. )}
695. <p className='mushroom-discripsion'>{mushroom.description}</p>
696. </div>
697. ))}
698. </div>
699. </div>
700. );
701. }
702. }
703. export default PoisonousMushrooms;
704. import React from 'react';
705. import { HashRouter, Routes, Route, NavLink } from 'react-router-dom';
706. class Header extends React.Component {
707. constructor(props){
708. super(props);
709. }
710. render() {
711. return (
712. <div className='Header'>
713. <div className='Header-panel-block'>
714. <NavLink className="Header-panel-block-link" to='/'>О нас</NavLink>
715. </div>
716. <div className='Header-panel-block'>
717. <NavLink className="Header-panel-block-link" to='/EdibleMushrooms'>Съедобные грибы</NavLink>
718. </div>
719. <div className='Header-panel-block'>
720. <NavLink className="Header-panel-block-link" to='/InedibleMushrooms'>Несъедобные грибы</NavLink>
721. </div>
722. <div className='Header-panel-block'>
723. <NavLink className="Header-panel-block-link" to='/PoisonousMushrooms'>Ядовитые грибы</NavLink>
724. </div>
725. <div className='Header-panel-block'>
726. <NavLink className="Header-panel-block-link" to='/Glavnaij'>Меры предосторожности</NavLink>
727. </div>
728. </div>
729. );
730. }
731. }
732. export default Header;
733. #root {
735. }
736. .Header{
737. background-color: green;
738. height: 60px; /\* стандартная высота для хедера \*/
739. width: 100%; /\* хедер занимает всю ширину экрана \*/
740. display: flex;
741. align-items: center; /\* выравнивание по вертикали \*/
742. justify-content: space-between; /\* выравнивание элементов внутри по горизонтали \*/
743. padding: 0 20px; /\* внутренние отступы слева и справа \*/
744. box-sizing: border-box; /\* учитываем паддинги в общей ширине \*/
745. }
746. .Mine{
747. padding: 20px;
748. box-sizing: border-box;
749. background-color: #f0f0f0; /\* светлый фон для области под хедером \*/
750. min-height: calc(100vh - 60px);
751. color: black;
752. }
753. .futher{
754. background-color: #b8860b; /\* золотисто-коричневый цвет \*/
755. color: white;
756. text-align: center;
757. padding: 50px;
758. position: relative;
759. bottom: 0;
760. width: 100%;
761. box-sizing: border-box;
762. }
763. /\*Далее хедер\*/
764. /\* styles.css \*/
765. .Header {
766. display: flex;
767. justify-content: space-around;
768. background-color: #2E8B57; /\* Цвета лесной зелени \*/
769. padding: 10px;
770. }
772. .Header-panel-block {
773. padding: 10px;
774. }
776. .Header-panel-block-link {
777. text-decoration: none;
778. color: white; /\* Цвет текста \*/
779. font-size: 18px;
780. transition: all 0.3s ease;
781. }
783. .Header-panel-block-link:hover {
784. color: #FFD700; /\* Цвет свечения при наведении \*/
785. text-shadow: 0 0 10px #FFD700; /\* Эффект свечения \*/
786. }
787. p{
788. font-weight: 700;
789. color: black;
790. }
791. .Futher {
792. background-color: #006400; /\* Темно-зеленый цвет фона \*/
793. color: #fff; /\* Белый цвет текста \*/
794. padding: 20px; /\* Отступы внутри блока \*/
795. border-radius: 5px; /\* Закругление углов \*/
796. }
797. .Futher h2 {
798. margin-bottom: 10px; /\* Отступ снизу для заголовков \*/
799. }
800. .Futher p {
801. margin: 0; /\* Убираем внешние отступы у абзацев \*/
802. }
803. import './App.css';
804. import React from 'react';
805. import { HashRouter, Routes, Route, NavLink } from 'react-router-dom';
806. import Header from './Component/Header';
807. import About from './Component/About/About';
808. import EdibleMushrooms from './Component/EdibleMushrooms/EdibleMushrooms';
809. import Glavnaij from './Component/Glavnaij/Glavnaij';
810. import InedibleMushrooms from './Component/InedibleMushrooms/InedibleMushrooms';
811. import PoisonousMushrooms from './Component/PoisonousMushrooms/PoisonousMushrooms';
812. import Futher from './Component/Futher';
813. class Content extends React.Component {
814. constructor(props){
815. super(props);
816. }
817. render() {
818. return (
819. <div className='Content'>
820. <HashRouter>
821. <Header></Header>
822. <Routes>
823. <Route path="/" element={<About />} />
824. <Route path="/EdibleMushrooms" element={<EdibleMushrooms />} />
825. <Route path="/Glavnaij" element={<Glavnaij />} />
826. <Route path="/InedibleMushrooms" element={<InedibleMushrooms />} />
827. <Route path="/PoisonousMushrooms" element={<PoisonousMushrooms />} />
828. </Routes>
829. <Futher></Futher>
830. </HashRouter>
831. </div>
832. );
833. }
834. }
835. export default Content;
836. componentDidMount() {
837. axios.get('https://localhost:7199/IadovGrib')
838. .then(response => {
839. if (Array.isArray(response.data)) {
840. this.setState({ mushrooms: response.data, loading: false });
841. } else {
842. console.error('Expected array but got:', typeof response.data, response.data);
843. this.setState({ error: 'Invalid data format', loading: false });
844. }
845. })
846. .catch(error => {
847. console.error('Error fetching data:', error);
848. this.setState({ error: error.message, loading: false });
849. });
850. }
851. toggleExpand = (id) => {
852. this.setState(prevState => ({
853. expandedMushroomId: prevState.expandedMushroomId === id ? null : id
854. }));
855. };
856. render() {
857. const { mushrooms, loading, error, expandedMushroomId } = this.state;
858. if (loading) {
859. return <div className='EdibleMushrooms'>Loading...</div>;
860. }
861. if (error) {
862. return <div className='EdibleMushrooms'>Error: {error}</div>;
863. }
864. return (
865. <div className='EdibleMushrooms'>
866. <h1>Ядовитые грибы</h1>
867. <div className='mushroom-list'>
868. {mushrooms.map(mushroom => (
869. <div
870. key={mushroom.id}
871. className={`mushroom-item ${expandedMushroomId === mushroom.id ? 'expanded' : ''}`}
872. onClick={() => this.toggleExpand(mushroom.id)}
873. >
874. <h2>{mushroom.name}</h2>
876. {mushroom.img && (
877. <img
878. src={`https://localhost:7199/Images/${mushroom.img}.jpg`}
879. alt={mushroom.name}
880. className='mushroom-image'
881. />
882. )}
883. <p className='mushroom-discripsion'>{mushroom.description}</p>
884. </div>
885. ))}
886. </div>
887. </div>
888. );
889. }
890. }
891. export default PoisonousMushrooms;
892. import React from 'react';
893. import { HashRouter, Routes, Route, NavLink } from 'react-router-dom';
894. class Header extends React.Component {
895. constructor(props){
896. super(props);
897. }
898. render() {
899. return (
900. <div className='Header'>
901. <div className='Header-panel-block'>
902. <NavLink className="Header-panel-block-link" to='/'>О нас</NavLink>
903. </div>
904. <div className='Header-panel-block'>
905. <NavLink className="Header-panel-block-link" to='/EdibleMushrooms'>Съедобные грибы</NavLink>
906. </div>
907. <div className='Header-panel-block'>
908. <NavLink className="Header-panel-block-link" to='/InedibleMushrooms'>Несъедобные грибы</NavLink>
909. </div>
910. <div className='Header-panel-block'>
911. <NavLink className="Header-panel-block-link" to='/PoisonousMushrooms'>Ядовитые грибы</NavLink>
912. </div>
913. <div className='Header-panel-block'>
914. <NavLink className="Header-panel-block-link" to='/Glavnaij'>Меры предосторожности</NavLink>
915. </div>
916. </div>
917. );
918. }
919. }
920. export default Header;
921. #root {
923. }
924. .Header{
925. background-color: green;
926. height: 60px; /\* стандартная высота для хедера \*/
927. width: 100%; /\* хедер занимает всю ширину экрана \*/
928. display: flex;
929. align-items: center; /\* выравнивание по вертикали \*/
930. justify-content: space-between; /\* выравнивание элементов внутри по горизонтали \*/
931. padding: 0 20px; /\* внутренние отступы слева и справа \*/
932. box-sizing: border-box; /\* учитываем паддинги в общей ширине \*/
933. }
934. .Mine{
935. padding: 20px;
936. box-sizing: border-box;
937. background-color: #f0f0f0; /\* светлый фон для области под хедером \*/
938. min-height: calc(100vh - 60px);
939. color: black;
940. }
941. .futher{
942. background-color: #b8860b; /\* золотисто-коричневый цвет \*/
943. color: white;
944. text-align: center;
945. padding: 50px;
946. position: relative;
947. bottom: 0;
948. width: 100%;
949. box-sizing: border-box;
950. }
951. /\*Далее хедер\*/
952. /\* styles.css \*/
953. .Header {
954. display: flex;
955. justify-content: space-around;
956. background-color: #2E8B57; /\* Цвета лесной зелени \*/
957. padding: 10px;
958. }
960. .Header-panel-block {
961. padding: 10px;
962. }
964. .Header-panel-block-link {
965. text-decoration: none;
966. color: white; /\* Цвет текста \*/
967. font-size: 18px;
968. transition: all 0.3s ease;
969. }
971. .Header-panel-block-link:hover {
972. color: #FFD700; /\* Цвет свечения при наведении \*/
973. text-shadow: 0 0 10px #FFD700; /\* Эффект свечения \*/
974. }
975. p{
976. font-weight: 700;
977. color: black;
978. }
979. .Futher {
980. background-color: #006400; /\* Темно-зеленый цвет фона \*/
981. color: #fff; /\* Белый цвет текста \*/
982. padding: 20px; /\* Отступы внутри блока \*/
983. border-radius: 5px; /\* Закругление углов \*/
984. }
985. .Futher h2 {
986. margin-bottom: 10px; /\* Отступ снизу для заголовков \*/
987. }
988. .Futher p {
989. margin: 0; /\* Убираем внешние отступы у абзацев \*/
990. }
991. import './App.css';
992. import React from 'react';
993. import { HashRouter, Routes, Route, NavLink } from 'react-router-dom';
994. import Header from './Component/Header';
995. import About from './Component/About/About';
996. import EdibleMushrooms from './Component/EdibleMushrooms/EdibleMushrooms';
997. import Glavnaij from './Component/Glavnaij/Glavnaij';
998. import InedibleMushrooms from './Component/InedibleMushrooms/InedibleMushrooms';
999. import PoisonousMushrooms from './Component/PoisonousMushrooms/PoisonousMushrooms';
1000. import Futher from './Component/Futher';
1001. class Content extends React.Component {
1002. constructor(props){
1003. super(props);
1004. }
1005. render() {
1006. return (
1007. <div className='Content'>
1008. <HashRouter>
1009. <Header></Header>
1010. <Routes>
1011. <Route path="/" element={<About />} />
1012. <Route path="/EdibleMushrooms" element={<EdibleMushrooms />} />
1013. <Route path="/Glavnaij" element={<Glavnaij />} />
1014. <Route path="/InedibleMushrooms" element={<InedibleMushrooms />} />
1015. <Route path="/PoisonousMushrooms" element={<PoisonousMushrooms />} />
1016. </Routes>
1017. <Futher></Futher>
1018. </HashRouter>
1019. </div>
1020. );
1021. }
1022. }
1023. export default Content;

# **Отладка и тестирование программы**

Отладка

Использование отладчика:

В Visual Studio откройте проект и запустите отладку, нажав на кнопку "Отладка" (Debug) или нажав клавишу F5.

Используйте отладчик для пошагового выполнения кода, проверки значений переменных и отслеживания ошибок.

Добавление точек останова (breakpoints):

Выберите строку кода, в которой хотите поставить точку останова, и щелкните левой кнопкой мыши слева от номера строки.

Запустите отладку, и выполнение программы остановится на выбранной точке останова, позволяя вам анализировать текущее состояние программы.

Использование логов:

Вставьте в код инструкции для записи в лог информации о ходе выполнения программы, значениях переменных и прочих деталях, которые могут быть полезны при отладке.

Тестирование

Модульное тестирование:

Напишите модульные тесты для каждой функции или метода вашего приложения.

Используйте библиотеки тестирования, такие как NUnit или MSTest, для написания и запуска тестов.

Интеграционное тестирование:

Проведите тестирование взаимодействия между различными компонентами вашего приложения, такими как база данных, бэкенд и фронтенд.

Проверьте корректность передачи данных и выполнение запросов между различными частями приложения.

UI тестирование:

Проведите тестирование пользовательского интерфейса, чтобы убедиться, что он работает корректно на разных разрешениях экранов и в различных браузерах.

Используйте инструменты для автоматизации тестирования пользовательского интерфейса, такие как Selenium WebDriver.

Тестирование производительности:

Проверьте производительность вашего приложения, запуская тесты на больших объемах данных или при максимальной нагрузке.

Используйте профилировщики производительности для выявления узких мест в вашем коде.

Тестирование безопасности

Тестирование уязвимостей:

Проведите аудит кода для выявления потенциальных уязвимостей в вашем приложении, таких как SQL инъекции, кросс-сайтовый скриптинг и другие.

Тестирование аутентификации и авторизации:

Проверьте корректность работы механизмов аутентификации и авторизации в вашем приложении.

Убедитесь, что доступ к защищенным ресурсам предоставляется только авторизованным пользователям.

Тестирование в производственной среде

Бета-тестирование:

Разверните приложение в бета-среде и предложите его использовать небольшой группе тестеров.

Получите обратную связь от тестеров о работе приложения и обнаруженных ошибках.

Мониторинг и отзывы пользователей:

Мониторьте работу приложения в производственной среде с помощью инструментов мониторинга.

Анализируйте отзывы и жалобы пользователей, чтобы быстро реагировать на обнаруженные проблемы и улучшить работу приложения.

Документация и отчеты

Составление отчетов:

Документируйте результаты тестирования, включая обнаруженные ошибки, их приоритеты и статус исправления.

Обновление документации:

Обновите документацию по проекту с учетом внесенных изменений и обнаруженных проблем.

# **ЗАКЛЮЧЕНИЕ**

В ходе курсовой работы была разработана информационно-справочная система "Справочник грибов", предоставляющая пользователям удобный доступ к информации о различных видах грибов. Программа обеспечивает пользователей актуальной информацией о съедобных, несъедобных и ядовитых грибах, а также содержит разделы с рекомендациями по безопасному сбору и употреблению грибов.

Достижения

Разработано веб-приложение с использованием технологий C#, ASP.NET, JavaScript, HTML, CSS и базы данных SQL Express.

Создана структура программы, включающая разделение на категории грибов, систему поиска, а также разделы "О нас" и "Меры предосторожности".

Реализована функциональность по добавлению и редактированию информации о грибах, а также обновлению базы данных.

Проведено тестирование программы, включая модульное, интеграционное, UI и тестирование безопасности.

Перспективы развития

Дальнейшее развитие программы может включать следующие аспекты:

Расширение базы данных грибов с добавлением новых видов и более подробной информации о каждом грибе.

Улучшение пользовательского интерфейса для повышения удобства использования и привлекательности приложения.

Внедрение дополнительных функций, таких как возможность сохранения избранных грибов или советов по приготовлению.

Расширение тестового покрытия для повышения надежности и стабильности приложения.

Проведение дополнительных исследований по безопасности и защите данных пользователей.

Разработка программы "Справочник грибов" представляет собой значимый шаг в обеспечении доступа к полезной информации о грибах для широкого круга пользователей. Проект демонстрирует не только технические навыки разработки программного обеспечения, но и стремление к созданию полезного и функционального продукта, способного улучшить качество жизни людей.

# **СПИСОК ЛИТЕРАТУРЫ**

1. Общие сведения о Windows Forms / Microsoft. URL: <https://docs.microsoft.com/ru-ru/dotnet/framework/winforms/windows-forms-overview> (дата обращения 21.04.2023)
2. Введение в язык C# и .NET Framework / Microsoft. URL: https://docs.microsoft.com/ru-ru/dotnet/csharp/getting-started/introduction-to-the-csharp-language-and-the-net-framework (дата обращения 21.04.2023)
3. Руководство по языку программирования VB.NET URL: https://metanit.com/visualbasic/tutorial/ (дата обращения 21.04.2023)
4. Работа с базами данных на языке C#. Технология ADO.NET: Учебное пособие / сост. О. Н. Евсеева, А. Б. Шамшев. – Ульяновск: УлГТУ, 2009. – 170 с.
5. Компьютерные технологии в экономике. Методические указания по выполнению курсовой работы. / Сост.: Тихоненко Д.В. – Красноярск: СибГАУ, 2015 – 18 с.
6. Введение в язык C# и NET. Framework. URL: <https://www.youtube.com/@SimpleCodeIT> (дата обращения 15.04.2023)