МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

ДОПУСТИТЬ К ЗАЩИТЕ

Заведующий кафедрой, д.ф.-м.н.,
профессор

_____ Л.Б. Соколинский

«___»_____ 2024 г.

Разработка веб-приложения для энтузиастов пешего горного туризма в парке «Таганай»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 09.03.04.2024.308-330.ВКР

Научный руководитель,
профессор кафедры СП, д.г.н.,
к.ф.-м.н.
_____ С.М. Абдуллаев

Автор работы,
студент группы КЭ-403
_____ И.П. Анин

Ученый секретарь
(нормоконтролер)
_____ И.Д. Володченко
«___»_____ 2024 г.

Челябинск, 2024 г.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

УТВЕРЖДАЮ
Зав. кафедрой СП
_____ Л.Б. Соколинский
29.01.2024 г.

ЗАДАНИЕ
на выполнение выпускной квалификационной работы бакалавра
студенту группы КЭ-403
Анину Ивану Павловичу,
обучающемуся по направлению
09.03.04 «Программная инженерия»

1. **Тема работы** (утверждена приказом ректора от 22.04.2024 г. № 764-13/12) Разработка веб-приложения для энтузиастов пешего горного туризма в парке «Таганай».
2. **Срок сдачи студентом законченной работы:** 03.06.2024 г.
3. **Исходные данные к работе**

3.1. Официальный сайт национального парка «Таганай». [Электронный ресурс] URL: https://www.taganay.org/ (дата обращения: 12.02.2024 г.).

3.2. Авторский проект «Ураловед» Павла Распопова об Урале. [Электронный ресурс] URL: https://uraloved.ru/natsionalnyj-park-taganaj (дата обращения: 12.02.2024 г.).

3.3. Костерин В.В. Разработка сайтов и Web-страниц: учебное пособие. / В.В. Костерин Е.В. Бунова, С.А. Богатенков. // Челябинск: Издательский центр ЮУрГУ, 2016. – 110 с.

4. **Перечень подлежащих разработке вопросов**

4.1. Провести анализ предметной области.

4.2. Сделать обзор существующих приложений по данной тематике.

4.3. Спроектировать и реализовать web-приложение.

4.4. Протестировать работу реализованного приложения.

5. Дата выдачи задания: 29.01.2024 г.

Научный руководитель,
профессор кафедры СП, д.г.н., к.ф.-м.н. С.М. Абдуллаев

Задание принял к исполнению И.П. Анин

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность

Пеший экотуризм – это вид туризма, который направлен на посещение охраняемых природных территорий и позиционируется как элемент общественного движения за сохранение окружающей среды и улучшение благосостояния местного населения.

Цель такого вида туризма может заключаться в экологическом просвещении путешественников, предоставлении финансовых средств для сохранения окружающей среды, в непосредственной помощи экономическому развитию ближайших сельских муниципалитетов и тем самым сохранению традиционного уклада природопользования.

Экотуризм одна из востребованных форм отдыха в России. Об этом свидетельствует недавнее обращение президента к Госдуме о скорейшей разработке закона об экотуризме [6].

Выделяют следующие положительные аспекты экотуризма.

1. Финансовая поддержка. Доходы от экотуризма могут направляться на сохранение и охрану природных заповедников, парков и резерватов. Это позволяет финансировать меры по охране природы, восстановлению экосистем и проведению научных исследований.

2. Поддержка местного населения. Экотуризм может стимулировать экономическое развитие местных сообществ, предоставляя рабочие места и стимулируя рост мелкого и среднего бизнеса, такого как гостиницы, рестораны и ремесленные магазины.

3. Сохранение культурного наследия. Экотуризм способствует сохранению культурного наследия и традиций местных сообществ. Туристы могут участвовать в культурных мероприятиях, посещать местные рынки и общаться с местным населением, что способствует сохранению уникальных культурных и традиционных практик.

4. Стимулирование заинтересованности в сохранении природы: Посещение природных мест может вдохновлять туристов на более ответственное отношение к окружающей среде. Многие из них могут стать сторонниками и активистами в области экологии, поддерживая проекты по сохранению природы и устойчивому развитию.

5. Участие в научных исследованиях: Экотуризм предоставляет возможность для участия в экологических и научных исследованиях, которые помогают лучше понять и сохранять местные экосистемы и виды.

Эти положительные аспекты показывают, что экотуризм может играть важную роль в сохранении окружающей среды, способствуя устойчивому развитию и содействуя местным сообществам. Экотуризм не только уменьшает негативное воздействие на природу, но и стимулирует экономический рост, создавая рабочие места и улучшая инфраструктуру в удаленных и сельских районах. Пеший экотуризм в настоящее время становится все более популярным среди людей различных возрастов и уровней подготовки. Этот вид активного отдыха предлагает уникальную возможность наслаждаться природой, открывать новые места и испытывать свои силы. В связи с ростом спроса как на гостиничный отдых внутри страны, так и на горный туризм, интернет-ресурсы для планирования и изучения местных достопримечательностей становятся все более востребованными. Они полезны как опытным туристам, которые ищут новые маршруты, так и новичкам, которые только начинают знакомиться с этим видом отдыха.

Национальный парк «Таганай», расположенный в Челябинской области, является одним из самых живописных и удобных парков в России. В нем созданы маршруты разной сложности и протяженности, что привлекательно как для профессиональных туристов, предпочитающих долгие маршруты и восхождения на скалы, так и для новичков, желающих совершить простые однодневные прогулки с красивыми видами. Развитая инфраструктура с маркированными тропами, указателями и оборудованными стоянками обеспечивает удобство и безопасность посетителям парка.

Постановка задачи

Целью выпускной квалификационной работы является разработка веб-приложения для энтузиастов пешего горного туризма в парке «Таганай». Для достижения поставленной цели необходимо решить следующие задачи:

1) провести обзор аналогичных решений;
2) спроектировать веб-приложение;
3) разработать дизайн;
4) разработать веб-приложение;
5) протестировать веб-приложение.

Структура и содержание работы

Работа состоит из введения, 4 глав, заключения и списка литературы. Объем работы составляет 40 страниц, объем списка литературы – 16 источников.

В рамках настоящей разработки портала для организации пешего туризма в национальном парке «Таганай» было проведено глубокое исследование существующих аналогичных порталов экотуризма. В первой главе квалификационной работы было подробно оценено содержимое нескольких таких порталов, проведен анализ используемых технологических инструментов, а также проанализированы функциональные и нефункциональные достоинства этих сайтов.

Учитывая особенности парка, его цели и задачи, были четко определены требования к функционалу и дизайну портала, чтобы он максимально соответствовал потребностям посетителей парка и помогал им получить наиболее полезную и качественную информацию о месте и его окружении.

В функционале, позволяющем интерактивно и гибко составить план похода и заключается уникальность разработки. Туристам предлагаются как традиционные, хорошо изученные маршруты, собранные в разделе «Излюбленная тропа», так и другие экотропы.

В разделе «Мое путешествие» туристы могут выбрать сами траекторию движения оценить ее длительность, физиологическую сложность и рассчитать необходимое туристское оборудование, запас питания и медикаменты. Функционал расширяется с помощью простейшей экспертной системы: вводя количество членов группы, их физические данные, ресурсы и длительность возможного похода туристы получат несколько вариантов маршрутов сопровождаемыми подробным описанием преимуществ и рисков маршрута в различных погодных условиях. После утверждения этого маршрута администрацией парка туристы могут совершить «собственное» путешествия с минимизацией рисков для себя и природы.

Во второй главе, используя язык универсального моделирования UML, проведено проектирование системы.

Третья и четвертая главы посвящены соответственно реализации и тестированию системы.

В приложении А содержится код, реализующий добавление интерактивной карты, и ввод желаемых параметров маршрута в экспертную модель, код генерации маршрута.

В приложении Б содержатся скриншоты разработанного приложения.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1. Обзор аналогичных приложений

На данный момент существуют несколько веб-сайтов, которые предоставляют возможность подробно изучить особенности пешего горного туризма в национальном парке «Таганай». Однако, несмотря на обширный контент, функционал таких ресурсов обычно ограничивается лишь описанием местности и достопримечательностей, не предоставляя достаточно информации для туристов-новичков. Такие сайты могут помочь быстро ознакомиться с красотами «Таганая» и узнать о его ключевых достопримечательностях, однако они недостаточно информативны для тех, кто только начинает свой путь в мир горного туризма и ищет руководство к действию или советы для самостоятельных походов, даже самых легких.

Сайт «Таганай» [14]

Наиболее популярный ресурс для ознакомления с национальным парком «Таганай» – это его официальный сайт, который представлен на рисунке 1.

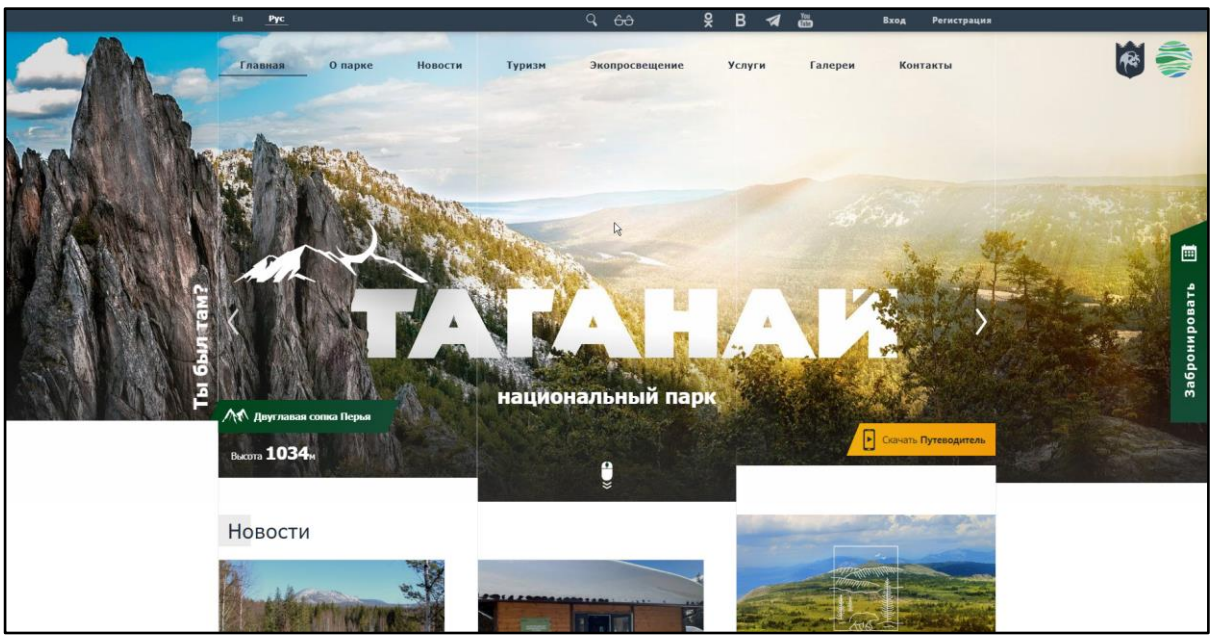


Рисунок 1 – Скриншот главной страницы сайта «Таганай»

Сайт позволяет увидеть фотографии различных мест парка, узнать об истории его создания, получить сведения о количестве видов животных и

растений, обитающих на территории «Таганая», изучить правила посещения и категории посетителей, которым установлены льготы, прочитать новости парка. Недостатком данного сервиса является то, что его информация – обзорная и будет полезна опытным туристам, но не даст ответа на многочисленные вопросы тех, кто хочет сходить в свой первый поход, а также не поможет получить карты маршрутов и списки снаряжения для похода. Основным средством разработки этого сайта является CMS Drupal.

Сайт «Наш Урал» [15]

Популярный ресурс для ознакомления с национальным парком «Таганай» – портал «Наш Урал», который представлен на рисунке 2.

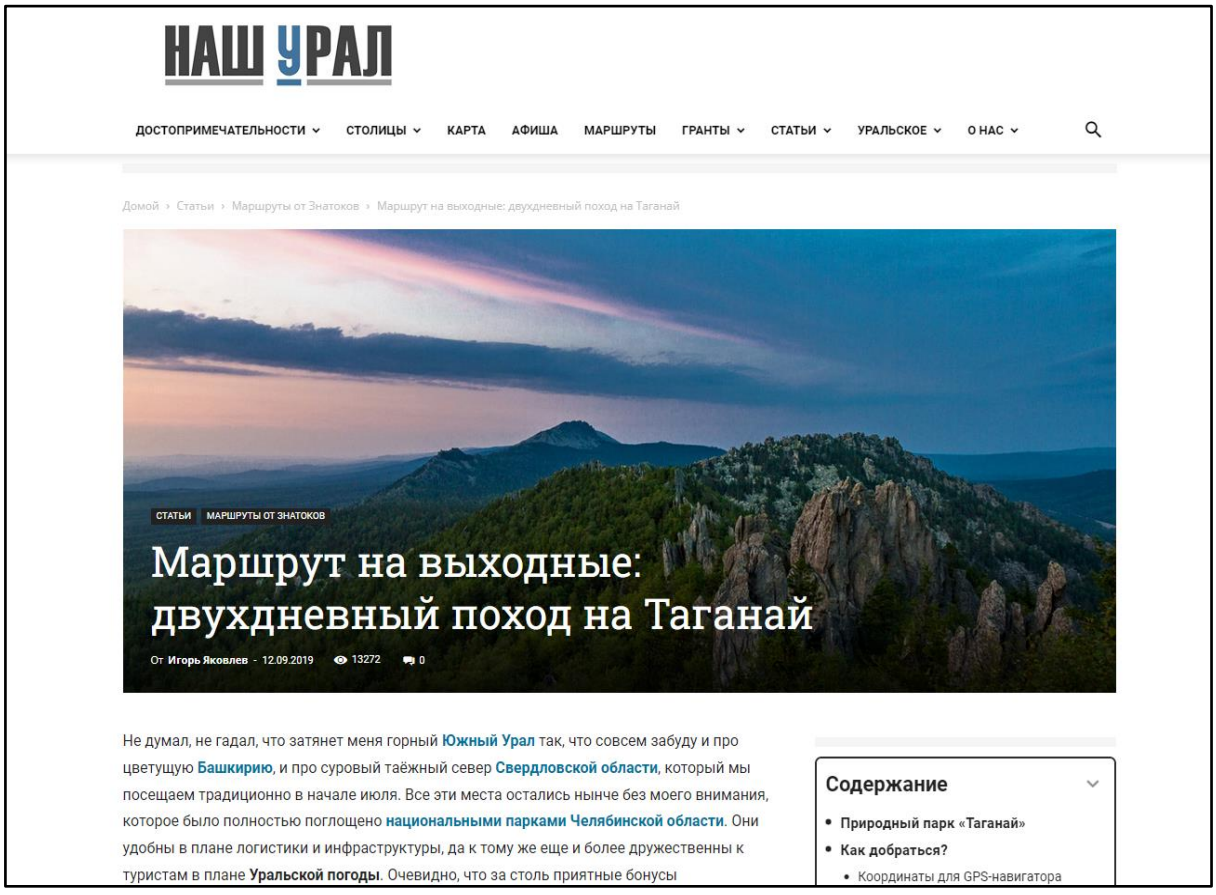


Рисунок 2 – Скриншот раздела «Таганай» на сайте «Наш Урал»

Сайт содержит большое количество авторских фотографий, описание национального парка «Таганай», раздел «Как добраться?», карту с рельефом местности, маршрут, по которому шел автор. Далее следует подробное описание того, с чем пришлось столкнуться автору, преодолевая выбранный им

маршрут. Ресурс отличается красочными фотографиями достопримечательностей «Таганая», подробно составленным дневником прохождения маршрута. К недостаткам относятся: отсутствие советов по необходимому снаряжению и особенностям климата в разное время года. Информации в разделе «Как добраться?» недостаточно. Нет информации о тарифах и видах размещения на территории национального парка «Таганай». Сайт написан с использованием HTML, CSS, JS, WordPress.

Сайт «Заповедники» [1]

Популярный ресурс для ознакомления с национальным парком «Таганай» – портал «Заповедники», который представлен на рисунке 3.

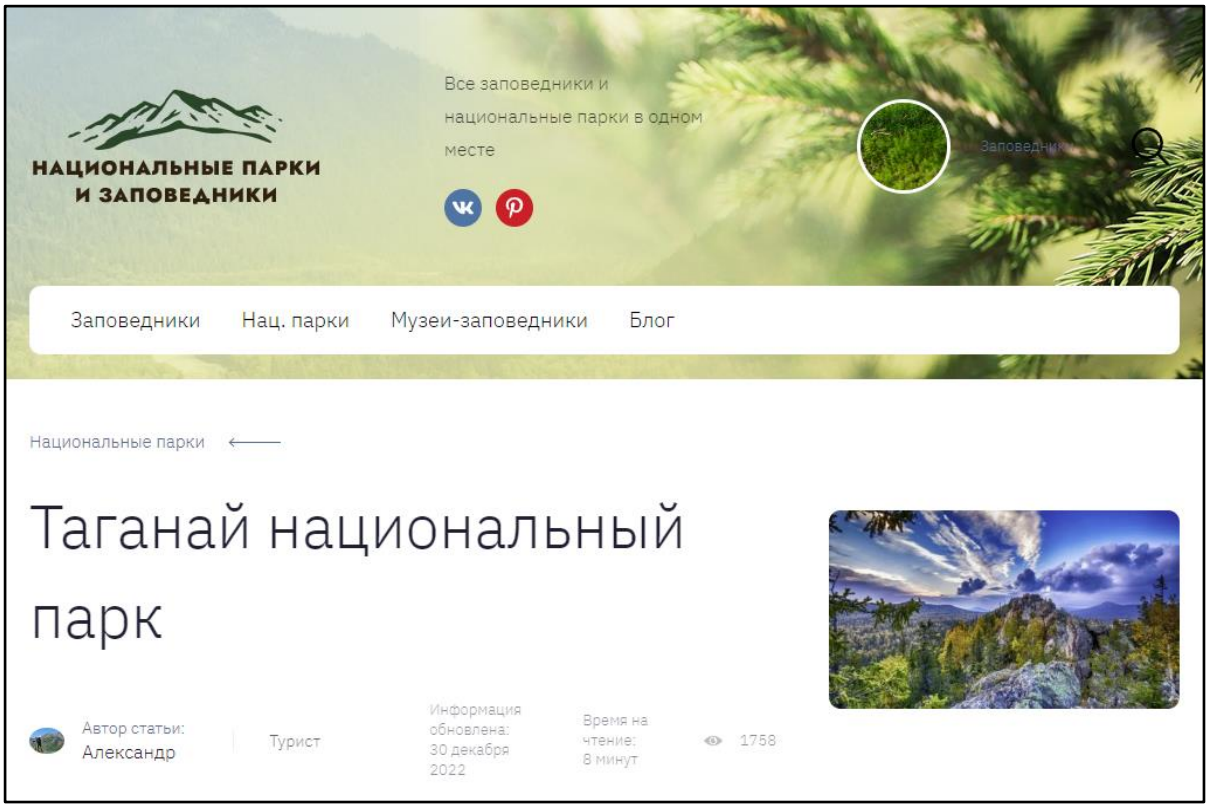


Рисунок 3 – Скриншот раздела «Таганай» на сайте «Заповедники»

Данный содержит большое количество информации о посещении национального парка «Таганай», включая:

1) информацию о том, как добраться до парка, на автомобиле или на общественном транспорте;

2) размер платы за вход, возможные льготы;

3) краткий список снаряжения;
4) фото животного и растительного мира национального парка;
5) список самых популярных маршрутов.

Однако, не смотря на обилье контента, присутствуют существенные недостатки: информация не актуализируется (данные о существующих тропах, тарифах и льготах устарели), рекомендованный список вещей и необходимого снаряжения есть, но является неполным. Сайт создан с использованием HTML, CSS, JS, WordPress.

1.2. Технологии разработки веб-приложений

Разработка веб-приложения включает в себя написание двух модулей: front-end и back-end. Front-end отвечает за визуальную составляющую сайта и взаимодействие пользователя с интерфейсом. Пользовательский интерфейс (UI) в настоящей работе создан с использованием HTML, CSS и JavaScript. Такой подход является традиционным в разработке front-end.

Back-end отвечает за обработку запросов пользователя и управление данными на сервере, исполнение сценариев бизнес-логики, взаимодействие с базой данных, аутентификацию и авторизацию пользователей, обработку файлов и многое другое. Основные технологии, применяющиеся для написания back-end: Python, Ruby, PHP, Java.

Традиционный подход к разработке back-end включает создание серверного приложения с использованием выбранного языка программирования и фреймворка, который облегчает разработку и обеспечивает безопасность и масштабируемость приложения. Back-end также отвечает за обеспечение взаимодействия с front-end сайта посредством API (Application Programming Interface), которое позволяет обмениваться данными между клиентом и сервером.

Сейчас стали набирать популярность JavaScript фреймворки, которые не требуют при разработке продукта использование HTML и CSS. Популярные JS фреймворки: React, создан Джорданом Валке, программистом из Facebook, и Vue.js, созданный Эваном Ю, разработчиком из Google.

React [11]

React – это JavaScript-библиотека с открытым кодом для создания пользовательских интерфейсов. В отличие от других библиотек JavaScript, предоставляющих полноценную платформу приложений, React ориентируется исключительно на создание представлений приложений через инкапсулированные единицы (называются компонентами), которые сохраняют состояние и генерируют элементы пользовательского интерфейса.

Vue.js [12]

Vue.js – это прогрессивный фреймворк для создания пользовательских интерфейсов. В отличие от фреймворков-монолитов, Vue создан пригодным для постепенного внедрения. Его ядро в первую очередь решает задачи уровня представления (view), что упрощает интеграцию с другими библиотеками и существующими проектами. С другой стороны, Vue полностью подходит и для создания сложных одностраничных приложений (SPA – Single-Page Applications), если использовать его совместно с современными инструментами и дополнительными библиотеками.

В ходе анализа технологий было выяснено что, применение JS фреймворков усложнит реализацию разрабатываемой системы, существенно повысив время разработки. Функционал JS фреймворков требуется для больших проектов, подразумевающих реализацию каскадных обновлений. В связи с этим был найден веб-фреймворк Django, функционал которого, наоборот должен помочь ускорить реализацию системы.

Django [4]

Django – это свободный фреймворк высокого уровня для разработки быстрых и безопасных веб-приложений и сайтов на языке Python. Исполь-

зует шаблон проектирования MVC. Django создан опытными разработчиками, берет на себя решение большей части стандартных вопросов по веб-разработке, поэтому разработчик может сосредоточиться на написании своего приложения без необходимости изобретать приемы и средства. Он бесплатный и с открытым исходным кодом. Фреймворк Django спроектирован по принципу «Все включено». Разработчик может с его помощью создать веб-приложение без сторонних компонентов, делая упор на функциональность и удобство пользовательского интерфейса.

Применение фреймворка позволяет:

1) ускорить разработку;
2) упростить поддержку;
3) избежать ошибок в работе приложения.

После изучения существующих вариантов было принято решение реализовать web-приложение используя фреймворк Django, спроектированный на Python высокого уровня, который поощряет быструю разработку и чистый, прагматичный дизайн.

Выводы по первой главе

В результате анализа предметной области и изучения аналогичных решений был выбран фреймворк Django для дальнейшей разработки. Проведенный обзор аналогичных платформ помог определить основной функционал, который типичен для подобных приложений. Далее была осуществлена более глубокая оценка выбранного фреймворка Django, включая его возможности, расширяемость, активное сообщество разработчиков и наличие дополнительных пакетов. В результате оценки подтверждено, что фреймворк Django соответствует требованиям проекта и обладает достаточной гибкостью для реализации необходимого функционала.

2. ПРОЕКТИРОВАНИЕ

2.1. Определение требований

В ходе проектирования приложения были определены функциональные и нефункциональные требования к разрабатываемой системе, сформированы варианты использования такой системы.

К функциональным требованиям относятся те, которые описывают требуемое поведение системы в определенных условиях.

Нефункциональные требования являются ограничениями, которые налагают на систему и определяют атрибуты ее качества.

В результате анализа предметной области и обзора существующих решений были сформированы следующие функциональные требования [16].

1. Система должна предоставлять возможность администратору системы актуализировать данные, удалять не актуальные.
2. Система должна предоставлять возможность сформировать документ на печать, содержащий маршрут похода с выбранными критериями.
3. Система должна предоставлять возможность сформировать документ на печать, содержащий список необходимого снаряжения.
4. Система должна содержать интерактивные формы для ввода пользовательских параметров похода.
5. Система должна предоставлять выбор списка готовых маршрутов.
6. Система должна отображать справочную информацию о каждом маршруте.

Также к системе были сформированы следующие нефункциональные требования.

1. Система должна быть доступна в любое время.
2. Система должна быть работоспособна в любое время.
3. Система должна эффективно использовать ресурсы.
4. Система должна быть отказоустойчивой.
5. Система должна быть защищена от несанкционированного доступа.

6. Система должна быть защищена от атак.

7. Система должны быть совместима с любыми устройствами, имеющими любое разрешение экрана и его ориентацию.

8. Система должна содержать простой и интуитивно понятный интерфейс пользователя.

2.2. Варианты использования системы

Для проектирования приложения был использован язык графического описания для объектного моделирования UML [5]. На рисунке 4 представлена диаграмма вариантов использования приложения для планирования туристического похода по национальному парку «Таганай». С приложением взаимодействуют два актера, пользователь и администратор.

Рисунок 4 – Диаграмма вариантов использования

Для взаимодействия с функциями приложения, пользователю необходимо выбрать нужный раздел на сайте. Разделы сайта представлены на рисунке 5.

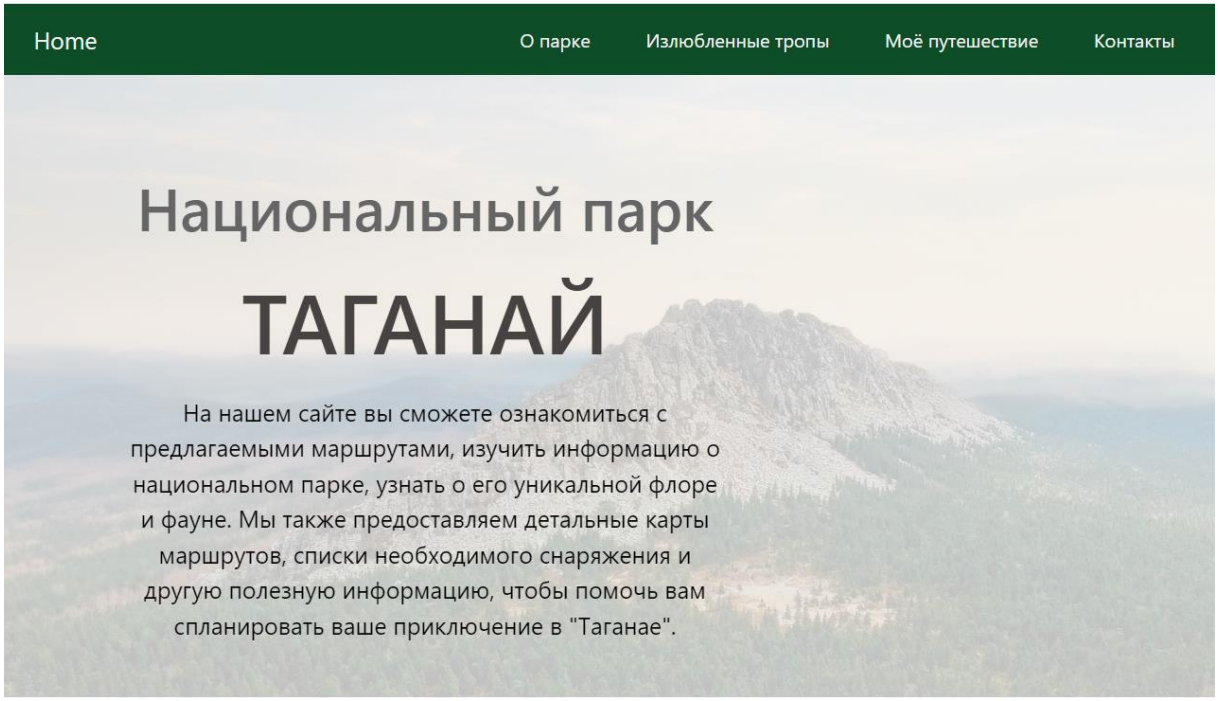


Рисунок 5 – Скриншот навигации сайта

Основные действия, которые пользователь может выполнять с помощью приложения.

1. Пользователь может посмотреть предложенные предустановленные походы (в разделе «Излюбленные тропы»), их описание, узнать их сложность и продолжительность. После ознакомления, пользователь может выбрать маршрут и вывести на свое устройство план выбранного похода и список необходимого снаряжения.

2. Пользователь может самостоятельно спланировать поход, указав желаемые требования по сложности, продолжительности, количеству участников, виду кострового снаряжения и т.п. После планирования, пользователь может загрузить на свое устройство план похода и список необходимого снаряжения.

3. Пользователь может изучить информацию о национальном парке «Таганай», посмотреть фотографии, ознакомиться с картой парка.

Для взаимодействия с функциями редактирования данных, используемых приложением, администратору необходимо авторизоваться в админ-

панели на сайте. После авторизации в которой, администратору становятся доступны следующие возможности.

1. Администратор может редактировать информацию о представленных маршрутах, актуализировать открытые стоянки, существующие доступные источники воды, уведомлять о введение в действие противопожарного режима.

2. Администратор может изменять информацию о тарифах на посещение национального парка в конструкторе похода.

3. Администратор может редактировать ознакомительную информацию о национальном парке «Таганай».

2.3. Проектирование архитектуры приложения

Архитектура приложения на фреймворке Django основана на шаблоне проектирования Model-View-Controller [9], MVC (модель-представление-контроллер). MVC состоит из объектов трех видов. Model – это объект приложения, содержащий данные приложения и «бизнес-логику». View отображает данные на экран пользователя. Controller реагирует на действия пользователя, оповещая модель о необходимости изменений. Общий принцип работы архитектуры MVC изображен на рисунке 6.

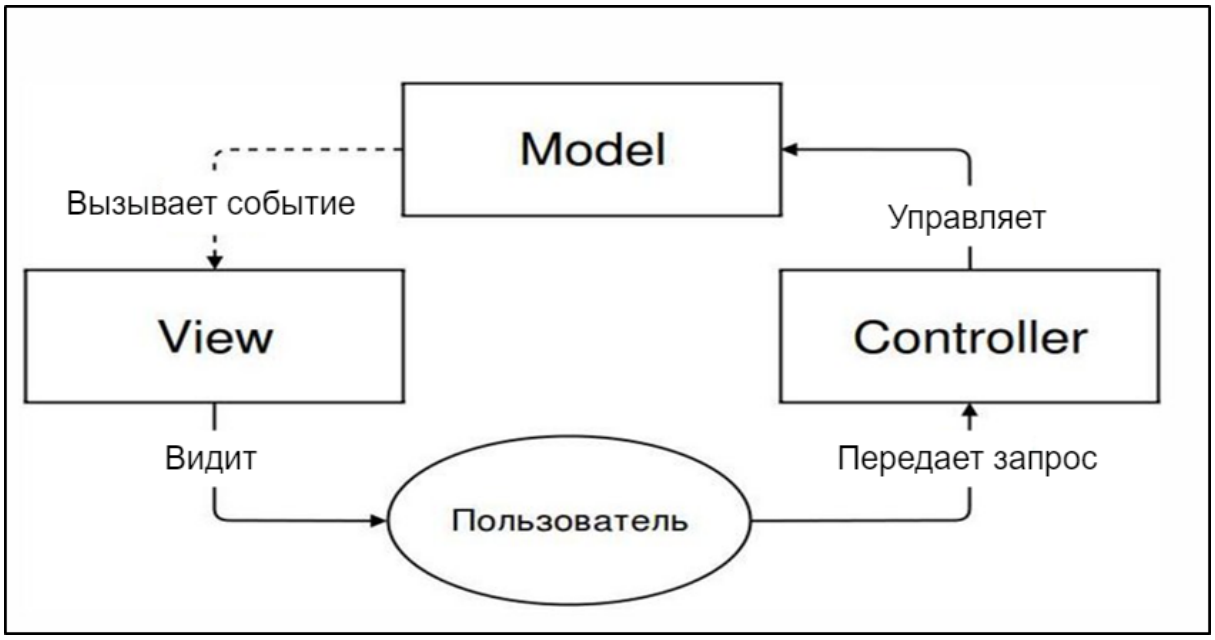


Рисунок 6 – Диаграмма архитектуры MVC

MVC предоставляет преимущества в разработке приложений, упрощая процесс добавления нового функционала [8]. MVC помогает разработчику сосредоточиться на модели данных, отделяя ее от остальных компонентов приложения. Это позволяет лучше организовать код и облегчает переиспользование, так как они становятся модульными и могут быть переиспользованы.

2.4. Компоненты системы

На рисунке 7 представлена диаграмма основных компонентов приложения.

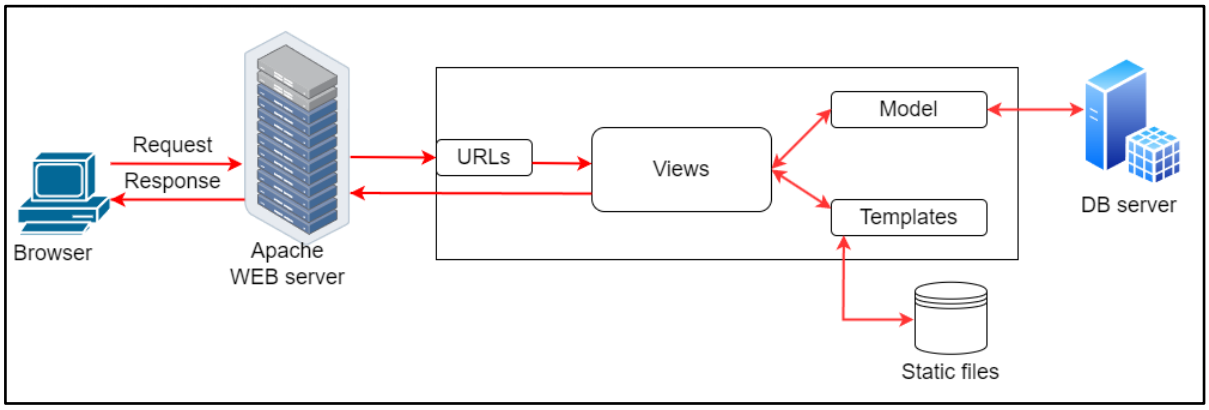


Рисунок 7 – Диаграмма основных компонентов

Пользователь взаимодействует с веб-приложением через браузер. Он отправляет HTTP-запросы к серверу, чтобы получить веб-страницы, отправить данные формы. Apache сервер принимает HTTP-запросы от браузера. Он действует как посредник между браузером и Django приложением. Django приложение выполняет основную бизнес-логику. Вот основные компоненты Django-приложения [7].

1. URLs выполняют маршрутизацию запросов. Определяет, какие запросы будут обрабатываться какими представлениями(views).

2. Views обрабатывают запросы и возвращает ответы (HTML-страницы). Представления взаимодействуют с моделями для получения данных.

3. Models представляют структуру данных. Модели взаимодействуют с базой данных для создания, чтения, обновления и удаления записей.

4. Templates определяют, как отображать данные на HTML-страницах. Шаблоны отрисовываются представлениями и возвращаются браузеру.

2.5. Модель базы данных

Все данные приложения хранятся в базе данных SQLite3. Каждая таблица представляет отдельную сущность. В SQLite3 каждое поле имеет свой тип данных, такой как INTEGER (целое число), TEXT (текстовая строка), REAL (число с плавающей запятой) и другие. Тип данных определяет, какие значения могут быть сохранены в поле и как они будут интерпретироваться. SQLite3 автоматически управляет хранением и индексацией данных в файле базы данных. При выполнении операции чтения или записи данных, SQLite3 обращается к файлу базы данных и возвращает результаты в соответствии с запросом. Для выполнения операций с базой данных, таких как выборка данных, обновление, вставка и удаление используется язык SQL.

На рисунке 8 визуальное представление схемы базы данных приложения для планирования похода.

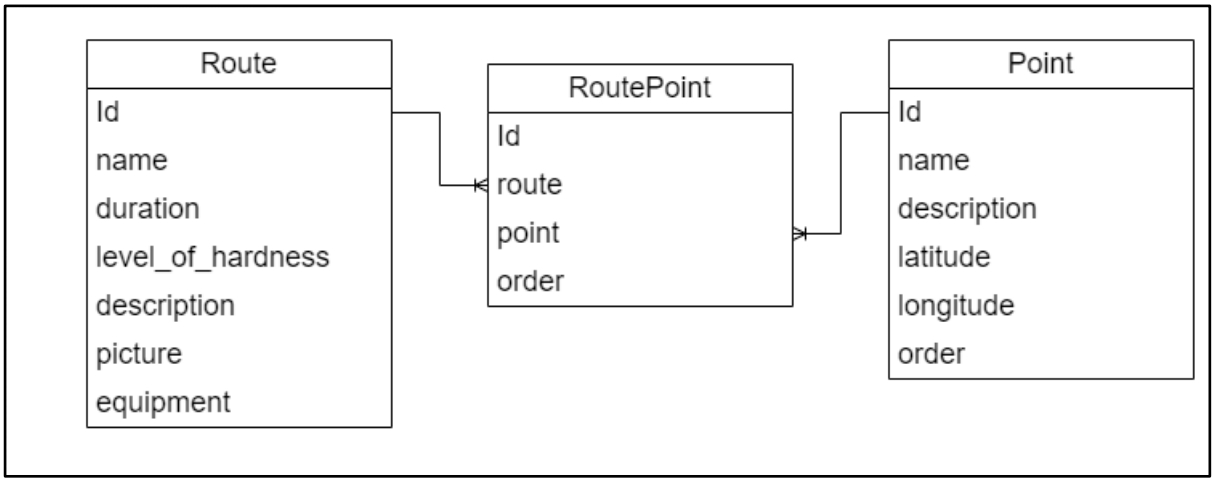


Рисунок 8 – Схема базы данных

В базе данных предусмотрены три основные таблицы для хранения информации о маршрутах и их точках.

Таблица «Point» содержит данные о точках, которые составляют маршруты. Каждая запись представляет собой точку маршрута и включает в себя следующие поля: идентификатор точки (id), название точки (name), широту (latitude) и долготу (longitude) ее координат, а также описание (description).

Таблица «Route» хранит информацию о маршрутах. Каждая запись представляет собой маршрут и содержит в себе следующие поля: идентификатор маршрута (id), название маршрута (name), уровень сложности (level_of_hardness), продолжительность (duration), описание (description), фото главной достопримечательности маршрута (picture), необходимое оборудование (equipment) и точки маршрута (points).

Таблица RoutePoint служит для связи таблиц Route и Point и включает в себя поля: идентификатор (id), идентификатор маршрута (route_id), идентификатор точки (point_id) и порядок точки в маршруте (order).

Выводы по второй главе

В данной главе были описаны функциональные и нефункциональные требования к разрабатываемой системе. Было определено, что система должна обеспечивать следующие функциональные возможности: просмотр пользователем предложенных маршрутов, построение собственных маршрутов (по параметрам, введенным пользователем), просмотр ознакомительной информации, редактирование и актуализация информации в приложении администратором.

3. РЕАЛИЗАЦИЯ

3.1. Выбор IDE для разработки и создание проекта

Для реализации веб-приложения был выбран PyCharm в качестве интегрированной среды разработки (IDE) [13]. PyCharm предоставляет широкий спектр инструментов и функций, специально разработанных для работы с языком Python и фреймворками, такими как Django. Ниже приведены основные причины, по которым был выбран PyCharm.

Поддержка Django: PyCharm обеспечивает полную интеграцию с Django, что позволяет легко создавать, отлаживать и развертывать веб-приложения на основе Django.

Интеллектуальные инструменты: PyCharm предоставляет широкий спектр интеллектуальных инструментов, таких как автозавершение кода, проверка типов, рефакторинг кода и многое другое, что упрощает процесс разработки и увеличивает производительность.

Удобство использования: интерфейс PyCharm интуитивно понятен и удобен в использовании. Он предоставляет множество функций, таких как навигация по проекту, интегрированная консоль, управление зависимостями и многое другое, что делает разработку приложений более эффективной.

Сообщество и поддержка: PyCharm имеет активное сообщество пользователей и разработчиков, что обеспечивает доступ к полезным ресурсам, форумам поддержки и расширениям, способствующим улучшению разработки.

Выбор PyCharm позволит эффективно использовать его возможности для разработки веб-приложения экотуризма на основе Django, обеспечивая высокое качество и производительность проекта.

Создание Django проекта

Для создания проекта нам необходимо установить фреймворк Django. Установку Django можно выполнить с помощью стандартного менеджера

пакетов Python – pip. Pip предоставляет простой способ управления зависимостями Python, позволяя пользователям легко устанавливать, обновлять и удалять пакеты Python из централизованного репозитория Python Package Index (PyPI) и других источников. Pip также поддерживает управление виртуальными средами Python, что позволяет изолировать зависимости различных проектов. Пример установки Django с помощью pip представлен на рисунке 9.

Рисунок 9 – Установка Django

Далее выполним команду «`django-admin startproject mysite`». После выполним «`python manage.py startapp naturetrail`». После выполнения этих команд в проекте будут автоматически созданы все необходимые файлы, включая файлы настроек, представлений, моделей и маршрутов для приложения naturetrail.

3.2. Создание базы данных сайта

В соответствии с разработанной моделью базы данных созданы соответствующие таблицы. Для этого был использован подход «code-first», подразумевающий определение модели данных на языке Python с использованием классов. Затем встроенная в Django ORM (Object-Relational Mapping) автоматически создает соответствующую схему базы данных на основе этих моделей.

Преимущества такого подхода включают:

1) быстрое создание моделей;
2) удобство внесения изменений, простой процесс миграции;

3) понятность и читаемость, код модели более понятен человеку, чем SQL-запросы и схемы базы данных.

Код модели `Route` представлен в листинге 1.

Листинг 1 – Код модели `Route`

```
class Routes(models.Model):
    DIFICULT_CHOICES = (
        ("Легкий", "Легкий"),
        ("Средний", "Средний"),
        ("Сложный", "Сложный"),
        ("Выживание", "Выживание"),
    )
    name = models.CharField(max_length=320, default='generated-route')
    duration = models.PositiveSmallIntegerField()
    level_of_hardness = models.CharField(max_length=20,
                                         choices=DIFICULT_CHOICES, )
    description = models.TextField()
    picture = models.ImageField(upload_to='images')
    equipment = models.FileField(upload_to='equipments')
    water = models.CharField(max_length=320)
    points = models.ManyToManyField('Point', through='RoutePoint')
```

Код модели `Point` представлен в листинге 2.

Листинг 2 – Код модели `Point`

```
class Points(models.Model):
    name = models.CharField(max_length=320, default='generated-route')
    description = models.TextField()
    coordinates = models.PointField()
    latitude = models.FloatField()
    longitude = models.FloatField()
    order = models.PositiveSmallIntegerField()
```

Код модели `RoutePoint`, необходимой для моделирования связи между таблицами `Route` и `Point`, представлен в листинге 3.

Листинг 3 – Код модели `RoutePoint`

```
class RoutePoint(models.Model):
    route = models.ForeignKey(Route, on_delete=models.CASCADE)
    point = models.ForeignKey(Point, on_delete=models.CASCADE)
    order = models.PositiveSmallIntegerField()
```

Созданные модели `Route`, `Point` и `RoutePoint` обеспечивают необходимую функциональность для управления данными о маршрутах и точках, входящих в эти маршруты. Модель `RoutePoint`, является промежуточной таблицей, моделирует связь «многие ко многим» между маршрутами и точками, и позволяет упорядочивать точки в каждом маршруте.

Таким образом, использование Django ORM и подхода «code-first» продемонстрировало свою эффективность и удобство. Определение моделей данных с помощью классов Python позволило быстро создать необходимые таблицы в базе данных.

3.3. Настройка маршрутизации и представлений страниц

Маршрутизация и представления являются ключевыми компонентами Django-приложения. Маршрутизация отвечает за связывание URL-адресов с соответствующими представлениями, которые обрабатывают запросы и возвращают ответы. В Django используется файл urls.py для настройки маршрутизации. Каждая запись в этом файле связывает определенный URL-адрес с функцией представления. Правильная настройка маршрутизации позволяет обеспечить корректное функционирование веб-приложения и его удобство для пользователей. Код файла urls.py представлен в листинге 4.

Листинг 4 – Код файла маршрутизации

```
urlpatterns = [
    path("", views.index, name='index'),
    path("generate/", views.create, name='create'),
    path('route/<int:route_id>/', views.route_detail, name='route_detail'),
    path('admin/', admin.site.urls),
    path('routeq/', views.submit_points, name='submit_points'),
 ]
```

Функция `index` обрабатывает запросы к корневому URL и отвечает за отображение домашней страницы сайта. Код функции представлен в листинге 5.

Листинг 5 – Код функции `index`

```
def index(request):
    return render(request, 'homePage.html')
```

Функция `create` отвечает за отображение страницы генерации маршрута и передает в шаблон `route_generator.html` массив точек, которые доступны для построения маршрута. Код функции представлен в листинге 6.

Листинг 6 – Код функции `create`

```
def create(request):
    queryset = Point.objects.all()
    all_points = [{"name": point.name, "longitude": point.longitude, "lati-
tude":point.latitude, "description":point.description, "order":point.order,
"closest_accomodation": point.closest_accomodation} for point in queryset]
    return    render(request,    'route_generator.html',    {'all_points':
all_points})
```

Функция `route_detail` отвечает за отображение детальной информации о маршруте. Принимая уникальный идентификатор маршрута в URL, она извлекает данные маршрута из базы данных, подготавливает их для отображения и возвращает соответствующую страницу с подробной информацией о маршруте. Код функции представлен в листинге 7.

Листинг 7 – Код функции `route_detail`

```
def route_detail(request, route_id):
    try:
        route = Route.objects.get(pk=route_id)
        data = prepare_data_for_ready_route(route)
    except Route.DoesNotExist:
        return render(request, 'error.html', {'error_message': 'Маршрут не найден'})
    print(data)
    return render(request, 'route_detail.html', data)
```

Функция `submit_points` предназначена для обработки параметров маршрута, отправленных через POST-запрос из формы генератора. Она принимает и обрабатывает JSON данные, создавая новые записи маршрута и связанных с ним точек в базе данных. Также функция отправляет запрос к внешнему API для генерации имени и описания маршрута и возвращает ответ с результатом операции, включая идентификатор созданного маршрута. Код функции представлен в листинге 1 приложения А.

Django предоставляет встроенную административную панель, доступ к которой осуществляется через URL `/admin/`. Это представление позволяет администраторам управлять данными сайта через удобный интерфейс.

3.4. Страница генерации маршрута

В соответствии с функциональными требованиями, страница для генерации маршрута должна содержать интерактивную карту, позволяющую

выбрать интересующие его точки, а также экспертную систему, которая позволяет ввести желаемые параметры маршрута.

Для реализации интерактивной карты была выбрана картографическая система Яндекс Карты [10]. Для работы с API Яндекс Карт был получен доступ на официальном сайте. В шаблон отображения страницы генератора маршрута был добавлен скрипт, реализующий отображение всех доступных точек на карте, обрабатывающий клик мышью по метке на карте. Выбранные точки сохраняются в объект selectedMarks типа Map. В объекте selectedMarks ключом является имя точки (point.name), а значением - сам объект точки. Таким образом, когда пользователь кликает на метку, код проверяет, находится ли метка в состоянии «выбрана» (красная иконка) или нет. Затем код изменяет иконку метки на противоположную (с красной на синюю или наоборот) и обновляет selectedMarks, добавляя или удаляя точку в зависимости от ее состояния. При нажатии пользователем на кнопку «Сгенерировать маршрут», происходит сбор данных из формы, таких как уровень сложности, продолжительность, количество участников, сезон и проживание. Данные полей формы и данные о точках маршрута, сохраненных в объекте selectedMarks, добавляются в объект bodyData, и отправляется POST-запросом на сервер в формате JSON. Это происходит с помощью функции fetch, которая выполняет AJAX-запрос, обрабатывает ответ, перенаправляя пользователя на страницу с детальной информацией о маршруте. Скрипт также содержит функцию getCookie, которая извлекает значение CSRF-токена из куки для обеспечения защиты от CSRF-атак при отправке POST-запроса. Код скрипта представлен в листинге 2 приложения А.

3.5. Сохранение маршрута в формате pdf

Для обеспечения возможности сохранения маршрута в формате PDF, на веб-странице был реализован соответствующий функционал. Используется библиотека html2pdf, которая позволяет преобразовывать содержимое

HTML-элемента в PDF-документ. Код реализующий данную функцию представлен в листинге 8.

Листинг 8 – Код функции сохранения маршрута в pdf-файл

```
<script>
    document.getElementById('save-pdf').addEventListener('click', function () {
        var element = document.getElementById('card-body');
        var opt = {
            margin:   1,
            filename: 'my_route.pdf',
            image: { type: 'jpeg', quality: 0.98 },
            html2canvas: { scale: 2 },
            jsPDF: { unit: 'in', format: 'letter', orientation: 'portrait'}};
        html2pdf().set(opt).from(element).save();});
</script>
```

3.6. Реализация интерфейса пользователя

Интерфейс пользователя (UI) является одним из ключевых компонентов веб-приложения. Он определяет, как пользователи взаимодействуют с системой, и оказывает значительное влияние на общий опыт использования. Перед началом реализации интерфейса был проведен тщательный анализ требований пользователей.

Основные требования включают:

1) интерактивные элементы экспертной модели;
2) визуальную привлекательность, эстетически приятный дизайн;
3) адаптивность.

Для создания интерфейса были использованы HTML5, CSS и JS. Скриншоты пользовательского интерфейса представлены в приложении Б.

Выводы по третьей главе

В данной главе был описан процесс реализации требуемого функционала приложения. Был осуществлён выбор и настройка среды разработки, создание проекта, разработка базы данных, настройка маршрутизации и представлений страниц. Также была создана страница генерации маршрута и реализован интуитивно понятный пользовательский интерфейс. Все этапы разработки были спланированы и последовательно выполнены, что позволило создать надёжное и удобное в использовании веб-приложение, соответствующее заявленным требованиям.

4. ТЕСТИРОВАНИЕ

4.1. Функциональное тестирование

Тестирование является одной из важнейших фаз разработки программного обеспечения, позволяющей выявить и устранить ошибки, а также подтвердить соответствие программы заданным требованиям [3]. В данном разделе рассмотрены процессы и методы, использованные для тестирования разработанного веб-приложения. Функциональное тестирование, обеспечивающему проверку всех основных функций и их корректное выполнение в соответствии с требованиями пользователей. Результаты функционального тестирования веб-приложения представлены в таблице 1.

Таблица 1 – Функциональное тестирование

№	Действие	Результат	Тест пройден?
1	Актуализация данных администратором системы	Администратор может обновлять данные и удалять неактуальные записи. Изменения отображаются корректно.	Да
2	Формирование документа на печать с маршрутом похода в формате pdf	Документ формируется с указанными критериями маршрута и доступен для печати, в формате pdf	Да
3	Формирование документа на печать со списком необходимого снаряжения в формате pdf	Документ формируется в формате pdf с полным списком необходимого снаряжения и доступен для печати.	Да
4	Ввод пользовательских параметров похода через интерактивные формы	Формы корректно принимают и обрабатывают пользовательские данные. Все параметры используются для генерации маршрута.	Да
5	Выбор маршрута из списка предложенных	Пользователь может выбирать из списка готовых маршрутов, информация отображается корректно.	Да
6	Отображение справочной информации о маршрутах	Справочная информация о каждом маршруте отображается корректно и полно.	Да
7	Выбор точек маршрута на интерактивной карте	Пользователь может выбирать и отмечать точки маршрута на карте, все выбранные точки передаются в функцию генерации.	Да
8	Определение параметров похода	Пользователь может устанавливать параметры похода, такие как уровень сложности, продолжительность, количество участников, сезон и проживание.	Да

Окончание таблицы 1

№	Действие	Результат	Тест пройден?
9	Сохранение маршрутов	После генерации маршрут корректно сохраняется в базу данных. Сохраненные данные корректно отображаются при последующем доступе.	Да
10	Переключение иконок меток на карте	Метки на карте корректно изменяют свой статус и иконки при клике, добавляются и удаляются из списка выбранных точек.	Да

4.2. Нефункциональное тестирование

Нефункциональное тестирование направленно на проверку аспектов работы системы, которые не связаны непосредственно с ее функциональностью. Основная цель нефункционального тестирования – удостовериться, что система удовлетворяет всем заявленным нефункциональным требованиям, таким как производительность, надежность, безопасность, удобство использования, совместимость и отказоустойчивость. Результаты нефункционального тестирования веб-приложения представлены в таблице 2.

Таблица 2 – Нефункциональное тестирование

№	Действие	Результат	Тест пройден?
1	Доступность системы в любое время	Система доступна пользователям 24/7 без простоев и технических перерывов.	Да
2	Работоспособность системы в любое время	Система функционирует корректно и стабильно без ошибок и сбоев в любое время суток.	Да
3	Эффективное использование ресурсов	Система оптимизирована для минимального потребления вычислительных и сетевых ресурсов.	Да
4	Отказоустойчивость системы	Система продолжает работу в случае частичного сбоя компонентов, обеспечены механизмы резервирования.	Да
5	Защита от несанкционированного доступа	Внедрены механизмы аутентификации и авторизации, предотвращающие несанкционированный доступ.	Да
6	Совместимость с любыми устройствами	Интерфейс системы адаптивен и корректно отображается на различных устройствах с любыми разрешениями.	Да
7	Простой и интуитивно понятный интерфейс пользователя	Интерфейс разработан таким образом, что пользователи легко и быстро осваиваются с системой.	Да

Выводы по четвертой главе

В данной главе был описан процесс тестирования приложения, включающий как функциональное, так и нефункциональное тестирование. Функциональное тестирование подтвердило корректность работы всех основных функций веб-приложения, таких как актуализация данных, формирование документов в формате PDF, ввод пользовательских параметров, выбор маршрутов и точек на карте, а также сохранение данных. Нефункциональное тестирование обеспечило проверку таких аспектов, как доступность и надежность системы, эффективность использования ресурсов, отказоустойчивость, защита от несанкционированного доступа, совместимость с различными устройствами и удобство пользовательского интерфейса. Результаты обоих видов тестирования подтвердили соответствие приложения заявленным требованиям, что свидетельствует о его готовности к эксплуатации.

ЗАКЛЮЧЕНИЕ

В данной работе было разработано веб-приложение для создания и планирования маршрутов туристических походов, предоставляющее пользователям возможность выбирать точки маршрута на интерактивной карте, определять параметры похода и сохранять созданные маршруты. В ходе выполнения работы были достигнуты все поставленные цели, учитывая как функциональные, так и нефункциональные требования:

1) проведен анализ предметной области;
2) произведен обзор существующих решений;
3) выполнено проектирование веб-приложения;
4) реализовано веб-приложение;
5) выполнено тестирование веб-приложение.

В результате работы получено веб-приложение, которое полностью соответствует потребностям пользователей и обладает потенциалом для дальнейшего расширения и улучшения. Этот проект не только представляет собой значимый вклад в область разработки веб-приложений, но и может стать полезным инструментом для туристического сообщества, облегчив процесс планирования и организации туристических маршрутов.

ЛИТЕРАТУРА

1. Авторский блог «Национальные парки и заповедники». [Электронный ресурс] URL: https://zapovedniki.com/nacionalnyj-park-taganaj (дата обращения: 12.02.2024 г.).

2. Авторский проект «Ураловед» Павла Распопова об Урале. [Электронный ресурс] URL: https://uraloved.ru/natsionalnyj-park-taganaj (дата обращения: 12.02.2024 г.).

3. Аграновский А.В. Тестирование веб-приложений: учебное пособие. // Санкт-Петербург: ГУАП, 2020. – 155 с.

4. Документация Django. [Электронный ресурс] URL: https://www.djangoproject.com (дата обращения: 12.02.2024 г.).

5. Иванов Д. Моделирование на UML. / Д. Иванов, Ф. Новиков. // Санкт-Петербург: НИУ ИТМО, 2010. – 200 с.

6. Информационное агентство ТАСС. [Электронный ресурс] URL: https://tass.ru/obschestvo/17105363 (дата обращения: 12.02.2024 г.).

7. Костерин В.В. Разработка сайтов и Web-страниц: учебное пособие. / В.В. Костерин, Е.В. Бунова, С.А. Богатенков. // Челябинск: Издательский центр ЮУрГУ, 2016. – 110 с.

8. Кряжева Е.В. Общие подходы к проектированию веб-приложений. / Е.В. Кряжева, Т.А. Васина, А.В. Краев. // Заметки ученого, 2021. – № 9–2. –С. 32–36.

9. Меле А. Django 2 в примерах. / А. Меле, перевод с английского Д. В. Плотниковой. // Москва: ДМК Пресс, 2019. – 408 с.

10. Официальная документация «API Яндекс Карт». [Электронный ресурс] URL: https://yandex.ru/maps-api/ (дата обращения: 12.02.2024 г.).

11. Официальный сайт JS фреймворка React. [Электронный ресурс] URL: https://react.dev (дата обращения: 12.02.2024 г.).

12. Официальный сайт JS фреймворка Vue.js. [Электронный ресурс] URL: https://vuejs.org/ (дата обращения: 12.02.2024 г.).

13. Официальный сайт интегрированной среды разработки «PyCharm». [Электронный ресурс] URL: https://www.jetbrains.com/pycharm/?var=1(дата обращения: 12.02.2024 г.).

14. Официальный сайт национального парка «Таганай». [Электронный ресурс] URL: https://www.taganay.org (дата обращения: 12.02.2024 г.).

15. Официальный сайт проекта «Двигаем Урал в России и в мире!». [Электронный ресурс] URL: https://nashural.ru/article/travel/dvuhdnevniy-pohod-na-taganay/ (дата обращения: 12.02.2024 г.).

16. Петрова О.Б. Разработка и анализ требований проектирования программного обеспечения: практикум: учебное пособие. // Санкт-Петербург: СПбГУТ им. М.А. Бонч-Бруевича, 2022. – 37 с.

ПРИЛОЖЕНИЯ

Приложение А.

В листинге 1 представлена реализация функции представления `submit_points`.

Листинг 1 – Реализация функции `submit_points`

```
def submit_points(request):
    if request.method == 'POST':
        try:
            data = json.loads(request.body)
            points = data.get('points', [])
            level_of_hardness = data.get('level', [])
            duration = data.get('duration', [])
            number_participants = data.get('number_participants', [])
            season = data.get('season', [])
            accommodation = data.get('accommodation', [])
            url = 'https://api.proxyapi.ru/openai/v1/chat/completions'
            headers = {
                'Content-Type': 'application/json',
                'Authorization': 'Bearer sk-NcABIisIJT5'
            }
            payload = {
                "model": "gpt-3.5-turbo",
                "messages": [
                    {"role": "system", "content": SYSTEM_PROMPT},
                    {"role": "user", "content": str(data)}
                ],
                "temperature": 0.7
            }
            response = requests.post(url, headers=headers, data=json.dumps(payload))
            if response.status_code == 200:
                api_response = response.json()
                api_json = json.loads(api_response['choices'][0]['message']['content'])
                generated_name = api_json['name']
                generated_description = api_json['description']
                # print(generated_name, generated_description)
            else:
                return JsonResponse({'status': 'error', 'message': 'API re-
quest failed'}, status=response.status_code)
             # Создание маршрута в базе данных
            new_route = Route.objects.create(
                name=generated_name,
                duration=duration,
                level_of_hardness=level_of_hardness,
                description=generated_description,
                price_of_accommodation=1000,
                water='',
                picture=None,
                equipment=None
            )
            # Добавление точек в маршрут
            for idx, point in enumerate(points):
                new_point = Point.objects.create(
                    name=point.get('name', 'Unnamed Point'),
                    description=point.get('description', ''),
                    latitude=point.get('latitude', 0),
                    longitude=point.get('longitude', 0),
                    order=idx + 1,
```

```
                )
                # Создание связи между маршрутом и точкой с указанием порядка
                RoutePoint.objects.create(
                    route=new_route,
                    point=new_point,
                    order=idx + 1
                )
                new_route.points.add(new_point)
            return JsonResponse({'status': 'success', 'message': 'Route created',
'route_id': new_route.id})
        except json.JSONDecodeError:
            return JsonResponse({'status': 'error', 'message': 'Invalid JSON'}, sta-
tus=400)
    else:
        return JsonResponse({'status': 'error', 'message': 'Invalid request method'}, sta-
tus=405)
```

Скрипт страницы генерации маршрута

В листинге 2 представлена реализация добавления интерактивной карты, с возможностью выбора точек, ввода желаемых параметров маршрута в экспертную модель, и отправки этих данных на сервер.

Листинг 2 – Код файла maps.js

```
document.addEventListener("DOMContentLoaded", (event) => {
  var selectedMarks = new Map();
  ymaps.ready(init);
  function init() {
    var myMap = new ymaps.Map("map", {
      center: [55.263577, 59.778038],
      zoom: 10,
      controls: [],
    });
    console.log(points);
    points.forEach(function (point) {
      var placeMark = new ymaps.Placemark([point.latitude, point.longitude], {
        hintContent: point.name,
      });
      // Добавляем флаг для отслеживания текущего состояния иконки
      placeMark.state.set("isRed", false);
      myMap.geoObjects.add(placeMark);
      placeMark.events.add("click", function (e) {
        var target = e.get("target");
        var isRed = target.state.get("isRed");
        console.log(isRed, point);
        if (isRed) {
          target.options.set("preset", "islands#blueIcon");
          selectedMarks.delete(point.name);
        } else {
          target.options.set("preset", "islands#redIcon");
          selectedMarks.set(point.name, point);
        }
        // Переключаем флаг
        target.state.set("isRed", !isRed);
      });
    });
    function getCookie(name) {
```

```
    let cookieValue = null;
    if (document.cookie && document.cookie !== "") {
      const cookies = document.cookie.split(";");
      for (let i = 0; i < cookies.length; i++) {
        const cookie = cookies[i].trim();
        if (cookie.substring(0, name.length + 1) === name + "=") {
          cookieValue = decodeURIComponent(cookie.substring(name.length + 1));
          break;
        }
      }
    }
    return cookieValue;
  }
  document
    .getElementById("create-route-btn")
    .addEventListener("click", function (e) {
      e.preventDefault();
      const level = document.getElementById("level_of_hardness").value;
      const duration = document.getElementById("duration").value;
      const number_participants = document.getElementById(
        "number_participants"
      ).value;
      const season = document.getElementById("season").value;
      const accommodation = document.getElementById("accommodation").value;

      var selectedPoints = Array.from(selectedMarks.values()).map(function (
        point
      ) {
        return {
          name: point.name,
          latitude: point.latitude,
          longitude: point.longitude,
          description: point.description,
          order: point.order,
          closest_accomodation: point.closest_accomodation,
        };
      });
      const bodyData = {
        points: selectedPoints,
        level,
        duration,
        number_participants,
        season,
        accommodation,
      };
      console.log(bodyData);
      fetch("/routeq/", {
        method: "POST",
        headers: {
          "Content-Type": "application/json",
          "X-CSRFToken": getCookie("csrftoken"),
        },
        body: JSON.stringify(bodyData),
      })
        .then((response) => response.json())
        .then((data) => {
          window.location.href = `\http://127.0.0.1:8000/route/${data.route_id}/`;
        });
    });
  }
});
```

Приложение Б.

Скриншоты пользовательского интерфейса

На рисунках 1–6 представлен пользовательский интерфейс приложения.

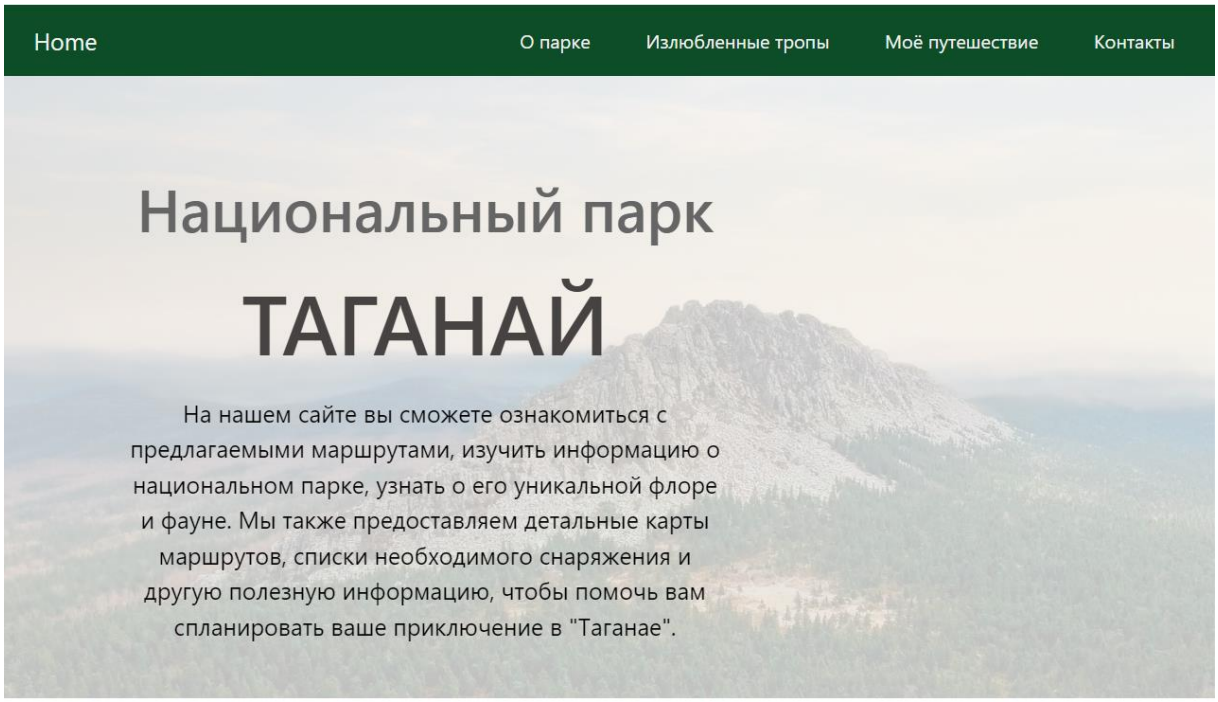


Рисунок 1 – Домашняя страница

Рисунок 2 – Домашняя страница, раздел «О парке»

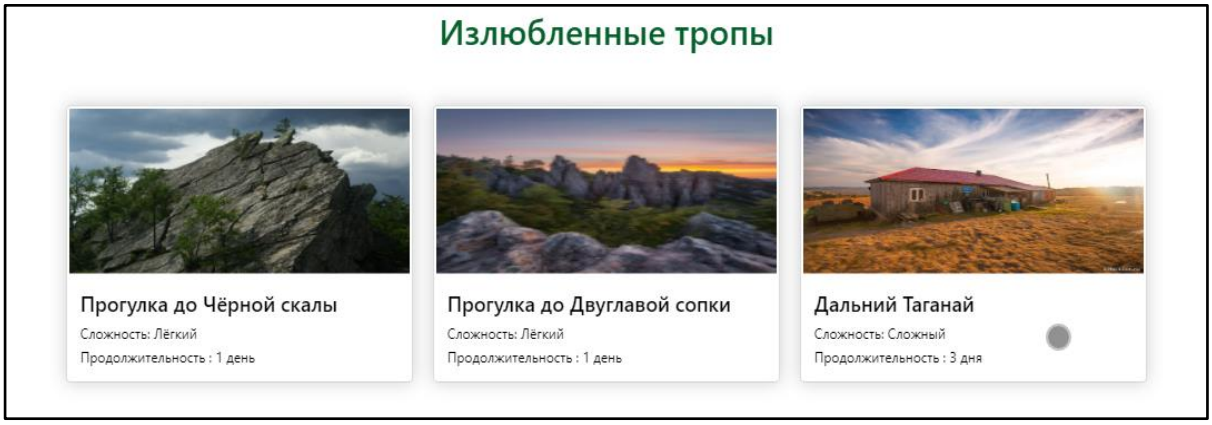


Рисунок 3 – Домашняя страница, раздел «Излюбленные тропы»

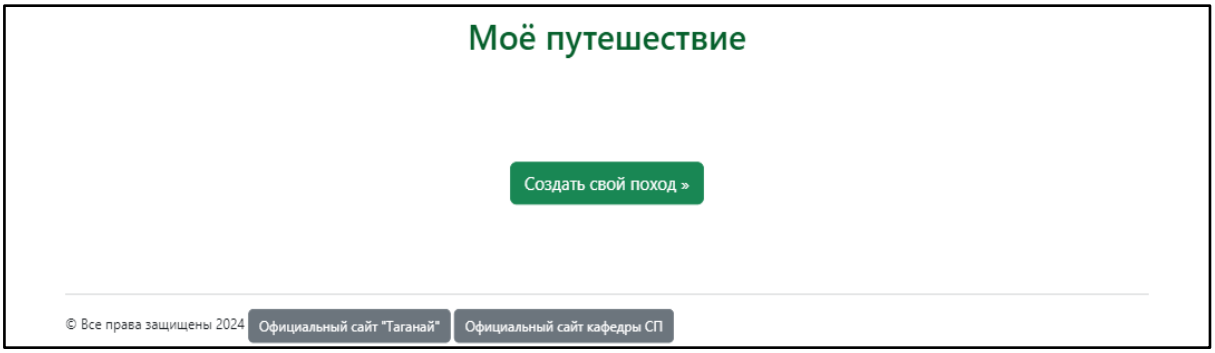


Рисунок 4 – Домашняя страница, раздел «Мое путешествие»

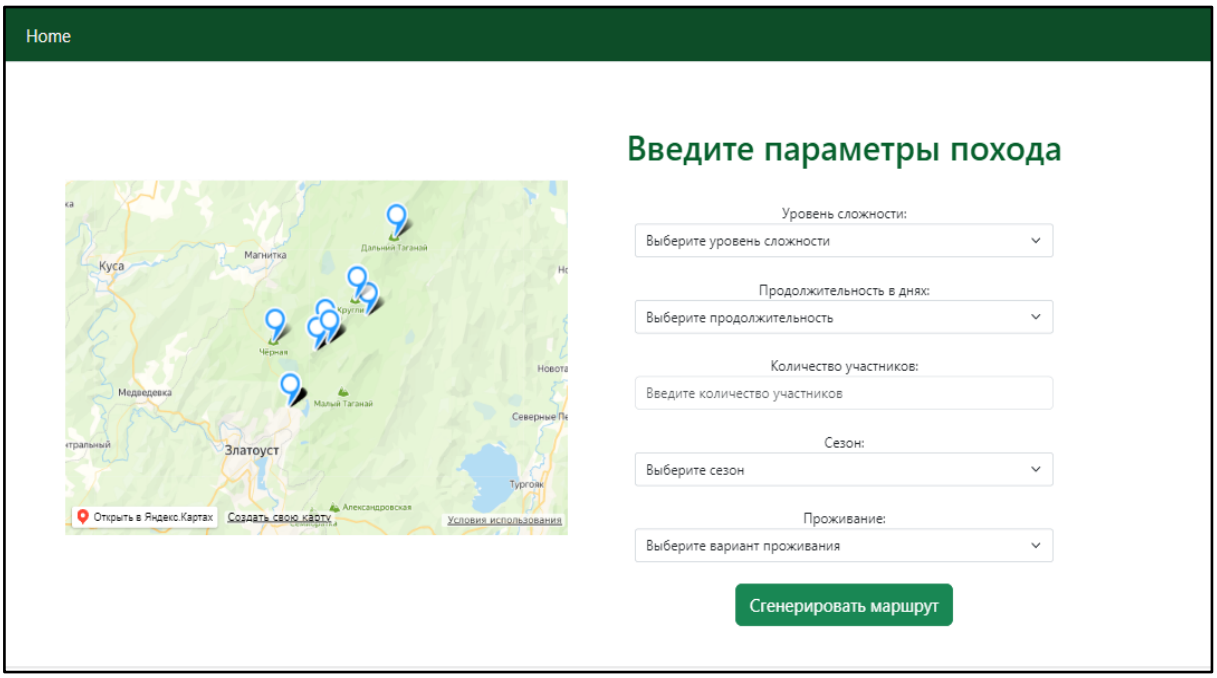


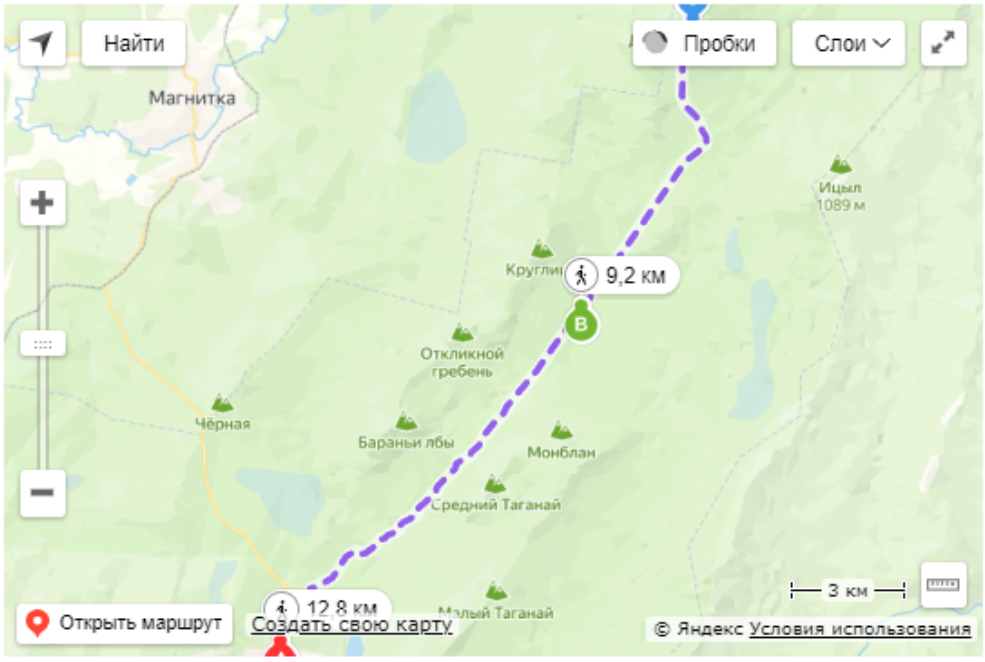
Рисунок 5 – Страница генерации маршрута

Дальний Таганай

Сложность: Сложный

Продолжительность: 3

Общая протяженность маршрута 50 км. Маршрут начинается от Центральной усадьбы Парка и проходит по Нижней тропе – одной из главных туристических троп Парка. Далее проходит вдоль Большой Каменной Реки, дважды пересекая её. В 9 км от начала пути на тропе находится туристическая стоянка «Писаный Камень» с источником питьевой воды, где можно отдохнуть. Полноценный привал можно сделать на приюте «Таганай». Почти на всем протяжении маршрута тропа проходит по Старой Киалимской дороге с остатками старых гатей через заболоченные участки и естественным каменистым покрытием. Пройдя еще 8 км от деревянных изб приюта «Таганай», тропа выводит вас на приют «Киалимский кордон», на котором запланировано разбить лагерь. На другой день запланирован подъем на вершину горы Дальний Таганай. Он несложный, но затяжной — 5 км от приюта. Здесь по мере подъема сосны и березы становятся всё ниже и ниже, перемежаясь с можжевельником и постепенно исчезая совсем. На самой вершине стоит здание бывшей метеостанции «Таганай-гора», на котором вы также можете остаться переночевать, днем исследуя тундру и каменные останцы Три Брата – гранитную летопись 300-миллионолетней давности. На следующей день возвращайтесь тем же маршрутом — спуск к «Киалимскому кордону», прогулка по нижней тропе вдоль каменной реки и выход к «Центральной усадьбе». Маршрут доступен для людей с хорошей физической подготовкой. ! Не рекомендуется брать детей младше 12 лет.



Сохранить в PDF

Рисунок 6 – Страница отображения информации о маршруте