

## Реферат

Пояснительная записка дипломного проекта содержит 68 страниц пояснительной записки, 1 таблиц, 1 формул, 1 иллюстрации, 1 источник литературы, 1 приложений.

DOCKER 4.39.0, ASP .NET CORE 8, C# 12.0, EF 8.0.2, MARIADB 11.5.2, REACT.JS 18.3.1, REDUX 9.2.0, GEMINI 2.0-FLESH, OPENSTREETMAP 0.7, STRIPE 17.4.0

Основной целью дипломного проекта является разработка веб-приложения для турагентства с возможностью гибкого подбора туров.

В рассматриваемом веб-приложении пользователю доступен просмотр туров, их бронирование и оплата. Так же пользователю доступна возможность подбора туров по географическим признакам при помощи фильтров, опоросу и ИИ.

В результате выполненных задач обоснована экономическая целесообразность разработки рассматриваемого веб-приложения. Данные наработки позволят повысить качество и конкурентоспособность выпускаемого программного обеспечения.

В первом разделе проведен аналитический обзор литературы и описана цель дипломного проекта, обзор аналогов.

Во втором разделе представлены используемые средства разработки, архитектура приложения и проектирование веб-приложения.

Третий раздел посвящен разработке программной реализации веб-приложения.

Четвертый раздел посвящен тестированию веб-приложения.

В пятом разделе приведено руководство пользователя.

В шестом разделе приводится расчет экономических параметров и себестоимость программного продукта.

В заключении приведены результаты проделанной работы.

					ДП 00.00.ПЗ		
		ФИО	Подпись	Дата	Реферат		
Разраб.		Дмитрук И.И.					
Пров.		Белодед Н.И.					
Н. контр.		Белодед Н.И.					
Утв.		Смелов В.В.					
					Лит.	Лист	Листов
						1	1
					у БГТУ 1-40 01 01, 2025		

## Abstract

The explanatory note of the graduation project contains 60 pages of explanatory notes, 1 table, 1 formula, 1 illustration, 1 literature source, 1 appendix.

DOCKER 4.39.0, ASP .NET CORE 8, C# 12.0, EF 8.0.2, MARIADB 11.5.2, REACT.JS 18.3.1, REDUX 9.2.0, GEMINI 2.0-FLESH, OPENSTREETMAP 0.7, STRIPE 17.4.0

The main goal of the diploma project is to develop a web application for a travel agency with the ability to flexibly select tours.

In the web application under consideration, the user can view tours, book them and pay for them. The user can also select tours by geographic features using filters, farrowing and AI.

As a result of the completed tasks, the economic feasibility of developing the web application in question was substantiated. These developments will improve the quality and competitiveness of the software being released.

The first section provides an analytical review of the literature and describes the purpose of the thesis project, an overview of analogues.

The second section presents the development tools used, the architecture of the application, and the design of the web application.

The third section is devoted to the development of a software implementation of a web application.

The fourth section is devoted to testing the web application.

The fifth section contains the user's guide.

The sixth section provides the calculation of the economic parameters and the cost of the software product.

In conclusion, the results of the work done are presented.

					ДП 00.00.ПЗ								
		ФИО	Подпись	Дата									
Разраб.	Дмитрук И.И.				Abstract				Лит.	Лист	Листов		
Пров.	Белодед Н.И.										1	1	
									у БГТУ 1-40 01 01, 2025				
Н. контр.	Белодед Н.И.												
Утв.	Смелов В.В.												

## Содержание

Реферат .....	1
Abstract .....	2
Содержание .....	3
Введение .....	7
1 Постановка задачи и аналитический обзор аналогичных решений .....	8
1.1 Постановка задачи.....	8
1.2 Обзор аналогичных решений.....	8
1.2.1 Веб-приложение «People Travel».....	8
1.2.2 Веб-приложение «Coral Travel».....	9
1.2.3 Веб-приложение «Tez Tour» .....	10
1.3 Выводы по разделу.....	11
2 Проектирование веб-приложения.....	12
2.1 Функциональность веб-приложения .....	12
2.2 Логическая схема базы данных .....	15
2.3 Архитектура веб-приложения.....	15
2.4 Блок-схема алгоритма бронирования тура .....	16
2.5 Выводы по разделу.....	17
3 Разработка веб-приложения .....	18
3.1 Разработка серверной части веб-приложения.....	18
3.1.1 Используемые библиотеки.....	18
3.1.2 Структура серверной части.....	19
3.1.3 Разработка уровня доступа к данным .....	20
3.1.4 Разработка уровня бизнес-логики .....	21
3.1.5 Разработка уровня представления .....	21
3.1.6 Настройка безопасности.....	22
3.1.7 Используемые паттерны.....	23
3.2 Клиентская часть .....	24
3.2.1 Используемые библиотеки.....	24
3.2.2 Структура клиентской части.....	24
3.2.3 Конфигурация проекта .....	25

					ДП 00.00.ПЗ		
		ФИО	Подпись	Дата	Содержание		
Разраб.	Дмитрук И.И.						
Пров.	Белодед Н.И.						
Н. контр.	Белодед Н.И.						
Утв.	Смелов В.В.				БГТУ 1-40 01 01, 2025		
					Лит.	Лист	Листов
						1	4

3.3 Реализация функций .....	25
3.3.1 Регистрация.....	25
3.3.2 Авторизация.....	25
3.3.3 Просмотр маршрутов.....	26
3.3.4 Просмотр отелей .....	26
3.3.5 Просмотр стран .....	26
3.3.6 Фильтрация маршрутов.....	26
3.3.7 Подбор маршрутов по опросу.....	26
3.3.8 Подбор маршрутов с помощью ИИ.....	27
3.3.9 Оставление отзывов к турам .....	27
3.3.10 Подача заявки на бронирование тура .....	27
3.3.11 Оплата забронированного тура.....	27
3.3.12 Просмотр истории бронирований .....	28
3.3.13 Отмена заявки на бронирование тура .....	28
3.3.14 Редактирование профиля.....	28
3.3.15 Добавление туров .....	28
3.3.16 Редактирование туров.....	28
3.3.17 Удаление туров.....	28
3.3.18 Добавление отелей .....	28
3.3.19 Редактирование отелей.....	29
3.3.20 Удаление отелей.....	29
3.3.21 Добавление пунктов отправлений.....	29
3.3.22 Редактирование пунктов отправлений.....	29
3.3.23 Удаление пунктов отправлений.....	29
3.3.24 Добавление стран .....	29
3.3.25 Редактирование стран .....	29
3.3.26 Удаление стран.....	30
3.3.27 Смена цены забронированного тура .....	30
3.3.28 Удаление отзывов к турам .....	30
3.3.29 Смена роли пользователей .....	30
3.3.30 Разблокировка пользователей.....	30
3.3.31 Удаление пользователей.....	30
3.3.32 Блокировка пользователей .....	30

3.4 Маршруты для контролеров.....	31
3.5 Выводы по разделу.....	31
4 Тестирование веб-приложения .....	32
4.1 Функциональное тестирование.....	32
4.2 Нагрузочное тестирования .....	36
4.3 Выводы по разделу.....	36
5 Руководство пользователя (руководство по эксплуатации) .....	37
5.1 Регистрация.....	37
5.2 Авторизация.....	38
5.3 Просмотр маршрутов.....	38
5.4 Просмотр отелей .....	40
5.5 Просмотр стран .....	41
5.6 Фильтрация маршрутов .....	42
5.7 Подбор маршрутов по опросу.....	44
5.8 Подбор маршрутов с помощью ИИ.....	44
5.9 Оставление отзывов к турам .....	45
5.10 Подача заявки на бронирование тура .....	45
5.11 Оплата забронированного тура.....	46
5.12 Просмотр истории бронирований .....	47
5.13 Отмена заявки на бронирование тура .....	48
5.14 Редактирование профиля.....	48
5.15 Добавление туров .....	48
5.16 Редактирование туров.....	50
5.17 Удаление туров.....	50
5.18 Добавление отелей .....	50
5.19 Редактирование отелей .....	51
5.20 Удаление отелей .....	51
5.21 Добавление пунктов отправлений.....	52
5.22 Редактирование пунктов отправлений.....	52
5.23 Удаление пунктов отправлений.....	53
5.24 Добавление стран .....	53
5.25 Редактирование стран .....	54
5.26 Удаление стран .....	54
5.27 Смена цены забронированного тура .....	54
5.28 Удаление отзывов к турам .....	55
5.29 Смена роли пользователей .....	55
5.30 Разблокировка пользователей.....	56
5.31 Удаление пользователей.....	56
5.32 Блокировка пользователей .....	56

5.33 Вывод по разделу .....	56
6 Технико-экономическое обоснования проекта .....	57
6.1 Общая характеристика разрабатываемого программного средства .....	57
6.2 Исходные данные для проведения расчётов и маркетинговый анализ .....	57
6.3 Обоснование цены программного средства .....	58
6.3.1 Расчёт затрат рабочего времени на разработку программного средства .....	58
6.3.2 Расчет основной заработной платы.....	59
6.3.3 Расчет дополнительной заработной платы.....	60
6.3.4 Расчет отчислений в Фонд социальной защиты населения и по обязательному страхованию .....	60
6.3.5 Расчет суммы прочих прямых затрат.....	61
6.3.6 Расчет суммы накладных расходов.....	61
6.3.7 Сумма расходов на разработку программного средства.....	62
6.3.8 Расходы на сопровождение и адаптацию .....	62
6.3.9 Расчет полной себестоимости.....	62
6.4 Вывод по разделу .....	63
Заключение .....	66
Список используемых источников.....	67
Диаграмма вариантов использования (ДП 01.00.ГЧ) .....	68
Логическая схема базы данных (ДП 02.00.ГЧ) .....	69
Диаграмма развертывания (ДП 03.00.ГЧ) .....	70
Блок-схема подключения к совместному просмотру (ДП 04.00.ГЧ) .....	71
Диаграмма компонентов (ДП 05.00.ГЧ) .....	72
Скриншот работы приложения (ДП 06.00.ГЧ) .....	73
Приложение А .....	74
Приложение Б.....	77
Приложение В.....	79
Приложение Г .....	84
Приложение Д.....	87

## Введение

Путешествия всегда привлекали людей — это способ узнать для себя что-то новое, получить впечатления или просто отдохнуть. В современном мире появляется всё больше возможностей для путешествий. Это связано с увеличением количества и качества представления туристических услуг. Многие сталкиваются с необходимостью в помощи подбора направлений, которые соответствуют личным интересам и предпочтениям, без необходимости самостоятельного изучения географических характеристик различных мест.

Цель дипломного проекта — разработать веб-приложение, позволяющее гибко подбирать туры по географическим признакам и их бронировать.

Серверная часть реализована на платформе ASP.NET Core 8.0 [1], клиентская часть — с использованием библиотеки React [2]. В качестве системы управления базами данных используется MariaDB [3], с подключением через Entity Framework Core [4]. Для обеспечения возможности оплаты используется Stripe [5]. Для отображения месторасположения объектов на карте используется OpenStreetMap [6]. Для подборов туров с помощью ИИ используется Gemini [7]. Для передачи данных используется REST API [8]. Приложение развёрнуто в Docker [9].

Основные этапы работы:

- постановка задачи и аналитический обзор аналогичных решений;
- проектирование веб-приложения;
- разработка веб-приложения;
- тестирования веб-приложения;
- руководство пользователя (руководство по эксплуатации);
- технико-экономическое обоснование проекта.

Для обеспечения безопасности пользователей используется валидация данных, управление ролями и политики авторизации.

Приложение предназначено для широкого круга пользователей, включая:

- любителей путешествовать, которые хотят найти предпочитаемый тур и комфортно его забронировать;
- владельцев отелей транспортных компаний, которые могут продвигать свои услуги через приложение;
- менеджеров, обеспечивающие предоставление туристических услуг и организацию туров.

Приложение будет доступно в сети интернет и будет работать через браузер.

					ДП 00.00.ПЗ		
		ФИО	Подпись	Дата			
Разраб.	Дмитрук И.И.				Введение		
Пров.	Белодед Н.И.						
Н. контр.	Белодед Н.И.						
Утв.	Смелов В.В.						
					Лит.	Лист	Листов
					У	1	1
					БГТУ 1-40 01 01, 2025		

# 1 Постановка задачи и аналитический обзор аналогичных решений

## 1.1 Постановка задачи

Необходимо разработать веб-приложение, которое поможет людям в организации путешествий. Веб-приложение должно предоставлять пользователям туры для бронирования. В бронь тура будет включена бронь номеров в отеле и транспорта. Пользователи должны иметь возможность удобно подбирать туры по различным критериям и географическим признакам. Необходимо предоставить возможность менеджерам в добавлении, удалении и редактировании туров, отелей, пунктов отправлений, городов и стран. Так же они должны подтверждать заявки на бронирование туров. Необходима модерация, чтобы следить за, тем, чтобы отзывы пользователей не были оскорбительными и неприличными, а также следить за тем, чтобы пользователи не спамили бронями. Для этого необходимы администраторы, которые смогут удалять неприемлемые отзывы, а также блокировать или удалять учётные записи пользователей.

## 1.2 Обзор аналогичных решений

### 1.2.1 Веб-приложение «People Travel»

Веб-приложение People Travel [10] предоставляет пользователям широкий спектр возможностей для подбора нужного тура. В первую очередь, подбор туров происходит по странам, которые предоставляются пользователям на главной странице в виде флажков и названий, затем уже пользователи более подробно настраивают себе отображение туров в фильтрах. Есть несколько видов туров. При просмотре отдельных туров, пользователю предоставляется очень подробная информация о нём.

Но при этом у дизайна сайта есть ряд проблем. При заходе на сайт, сразу почти на весь экран отображается реклама. Для взаимодействия с основным функционалом, необходимо совершить ряд действий, такие как прокрутка вниз и выбор страны. Так же интерфейс не единообразный, то есть у разных стран, разные виды туров, и при их выборе, настройки отображения туров будут разными, что создаёт путаницу.

Кроме того, при бронировании тура, нельзя указать какую-либо информацию, кроме своих контактных данных. Это всё создаёт ряд сложностей тем, что менеджерам приходится самостоятельно узнавать у пользователей эту информацию.

					ДП 01.00.ПЗ		
		ФИО	Подпись	Дата			
Разраб.	Дмитрук И.И.				1 Постановка задачи и аналитический обзор аналогичных решений		
Пров.	Белодед Н.И.						
Н. контр.	Белодед Н.И.						
Утв.	Смелов В.В.						
					Лит.	Лист	Листов
					У	1	4
					БГТУ 1-40 01 01, 2025		



Интерфейс веб-приложения представлен на рисунке 1.1.

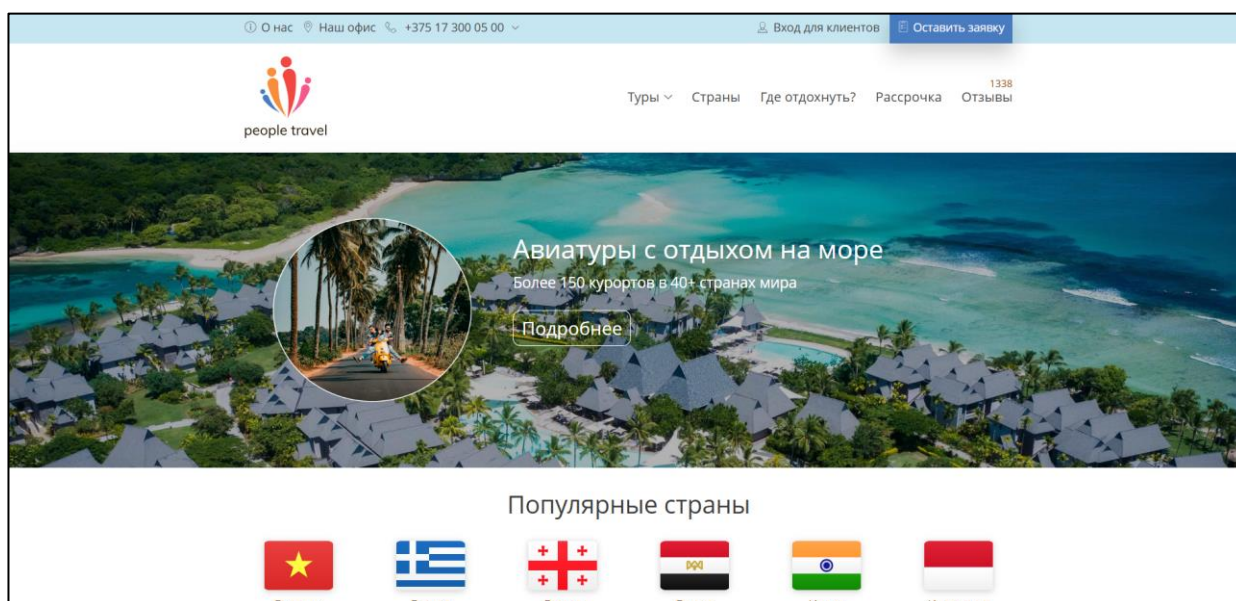


Рисунок 1.1 – Веб-приложение «People Travel»

Проанализировав веб-приложение, было решено сделать похожим дизайн интерфейса подбора стран.

### 1.2.2 Веб-приложение «Coral Travel»

Coral Traver [11] предоставляет возможность подбора туров, рейсов и отелей. Заказать можно как пакетный тур, в который входит бронь транспорта и отеля, так и просто отель. Фильтрация технически реализована удобно – есть множество возможностей для подбора туров, к примеру, можно указать количество детей и их возраст для подбора туров с отелями, в которых есть возможность размещения таких детей, а также предоставляют для них скидки. Есть история подборов туров. Однако у фильтров есть проблемы в дизайне, и заключаются они в том, что в большинстве случаев это сплошной текст. Например, при выборе направлений, города и страны предоставляются в виде списка, а большое их количество, приводит к тому, что их неудобно выбирать.

При просмотре отдельного тура, можно увидеть подробную информацию о нём. При чём, описание многих характеристик представлено визуально в виде значков или фотографий с отеля. При бронировании тура, можно выбрать конкретный тип номера. Однако нельзя заказать несколько номеров. Таким образом, забронировать мест в туре можно столько, сколько людей помещается в номер, что означает, что, если будет больше людей, значит придётся делать несколько заказов.

Важной особенностью является возможность сравнивать отели. После добавления отелей в сравнение, можно просмотреть, в чём их отличия по конкретным характеристикам.

Интерфейс веб-приложения представлен на рисунке 1.2.

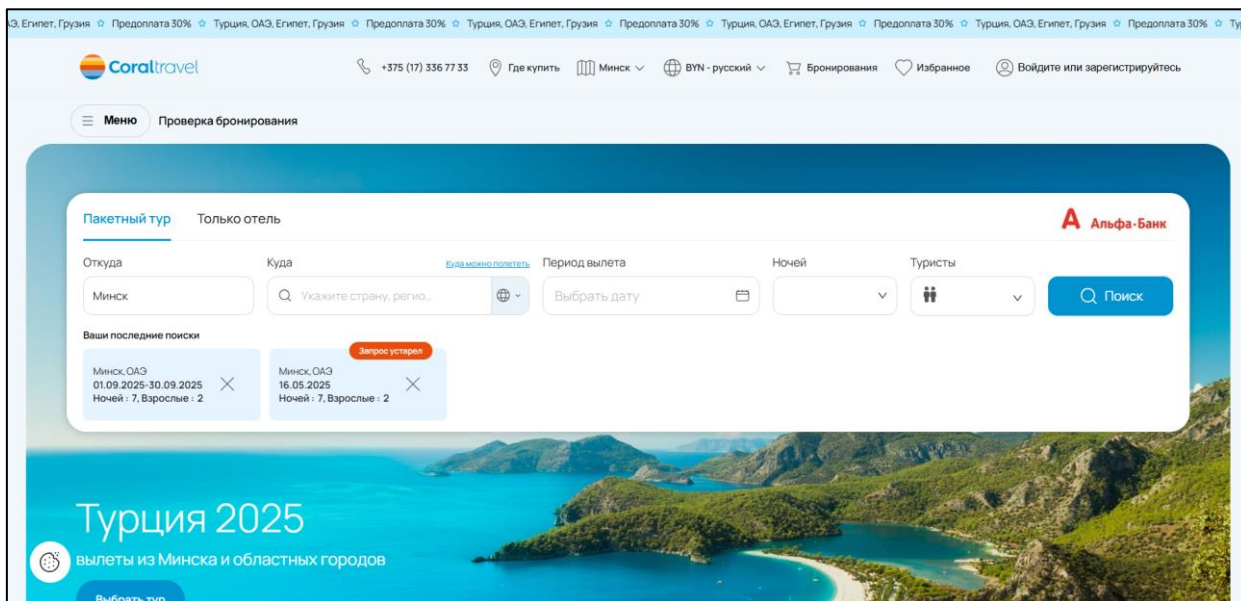


Рисунок 1.2 – Веб-приложение «Coral Travel»

Проанализировав, веб-приложение было решено делать схожую страницу для тура.

### 1.2.3 Веб-приложение «Tez Tour»

Веб-приложение «Tez Tour» [12] на главной странице сайта сразу предоставляет возможность удобно настраивать отображение туров по основным критериям. На сайте можно посмотреть информацию о деталях туров, таких как отели, транспорт, документы т.д. Очень подробную информацию можно узнать о странах, такую как экскурсии в ней, советы по получению виз, достопримечательности, климат и т.д. Самые популярные страны представлены на главной странице в виде фотографий их достопримечательностей. Ниже стран представлены популярные туры. Туры разделены на типы. Эти типы удобно отображаются в виде значков и текстов.

При просмотре отдельного тура, можно увидеть довольно подробную информацию о нём, однако она расположена не удобно и представляет с собой набор хаотично разбросанных характеристик. При бронировании необходимо выбрать тип номера и рейс. Рейсы отображаются на странице тура под каждым типом номера, что не очень удобно, ведь ко всем типам номеров и так одинаковые рейсы, тем самым они просто дублируются и заполняют место ненужной информацией.

Данное веб-приложение повторяет проблему предыдущего, что в нём нельзя заказать несколько номеров в отеле.

В целом подбор туров довольно удобный визуально и технически, однако сама страница тура переполнена ненужной информацией, которая при этом плохо визуально воспринимается.

Интерфейс веб-приложения представлен на рисунке 1.3.

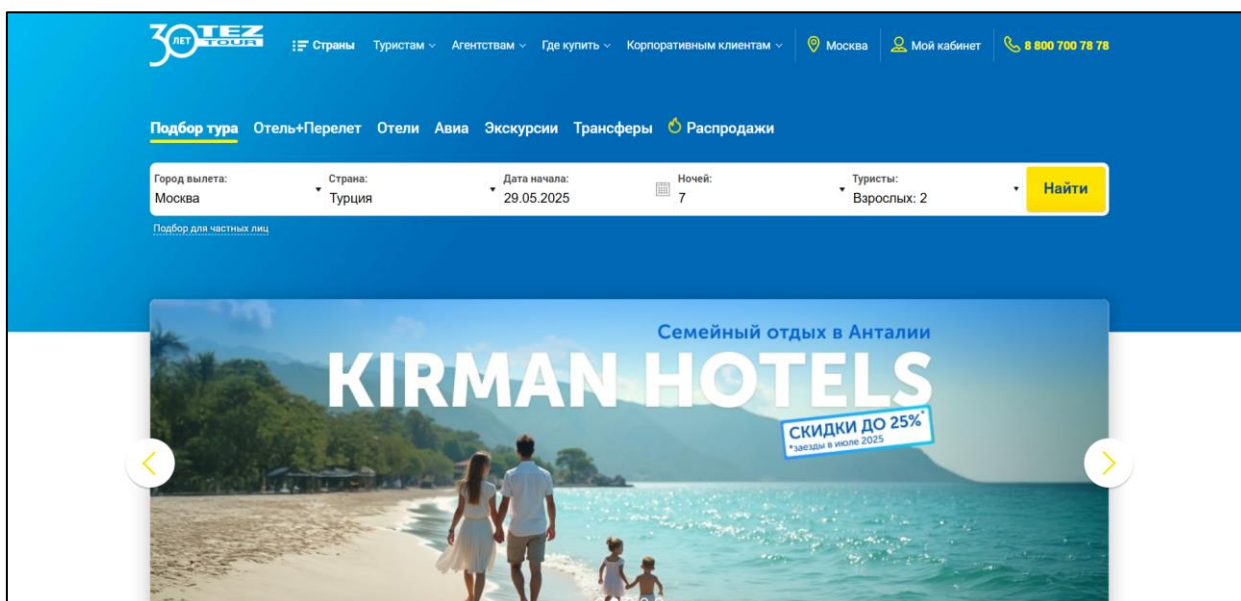


Рисунок 1.3 – Веб-приложение «Tez Tour»

Проанализировав веб-приложение, было решено взять дизайн разделения туров на типы.

### 1.3 Выводы по разделу

В рамках данной главы было выполнено:

1. Поставлены необходимые задачи для реализации веб-приложения;
2. Разобраны три аналогичных решения: «People Travel», «Coral Travel», «Tez Tour». Из «People Travel» взят интерфейс подбора туров по странам, из «Coral Travel» взят дизайн страницы тура, а из «Tez Tour» взято разделение туров на типы.
3. Было отмечено, что ни в одном из аналогов не имеется возможность подбора туров по географическим признакам, а также выбора нескольких номеров в отеле при бронировании.

## 2 Проектирование веб-приложения

### 2.1 Функциональность веб-приложения

Функциональные возможности веб-приложения представлены в диаграмме вариантов использования, которая приведена в ДП 01.00.ГЧ.

Описание ролей пользователей веб-приложения представлены в таблице 2.1.

Таблица 2.1 – Описание ролей

Роль	Описание
Гость	Пользователь, не прошедший авторизацию. Имеет доступ только к просмотру туров.
Клиент	Зарегистрированный пользователь, имеющий возможность взаимодействовать с контентом.
Менеджер	Может добавлять, удалять, редактировать туры, отели, пункты отправлений, города и страна, а также подтверждать заявки на бронирование туров.
Администратор	Может удалять отзывы к турам, а также удалять, блокировать пользователей и изменять им роль. Так же может выполнять функции менеджера.

Роли в системе разделены таким образом, чтобы разграничить доступ к функционалу приложения и обеспечить безопасное использование платформы. Каждая роль наделена строго определённым набором возможностей, что упрощает управление и поддержку системы.

Описание функций предоставлена на таблице 2.2.

Таблица 2.2 – описание функций

№	Функция	Описание	Доступно ролям
1	Регистрация	Возможность создания нового аккаунта для получения доступа к расширенному функционалу.	Гость
2	Авторизация	Вход в систему для использования персонализированного функционала.	Гость
3	Просмотр маршрутов	Возможность просмотра маршрутов.	Гость, Пользователь, Менеджер, Администратор

					ДП 02.00.ПЗ		
		ФИО	Подпись	Дата	2 Проектирование веб-приложения		
Разраб.	Дмитрук И.И.						
Пров.	Белодед Н.И.						
Н. контр.	Белодед Н.И.						
Утв.	Смелов В.В.						
					Лит.	Лист	Листов
					У	1	6
					БГТУ 1-40 01 01, 2025		

Продолжение таблицы 2.2

№	Функция	Описание	Доступно ролям
4	Просмотр отелей	Возможность просмотра отелей.	Гость, Пользователь, Менеджер, Администратор
5	Фильтрация маршрутов	Возможность просмотра списка маршрутов с выбранными характеристиками	Гость, Пользователь, Менеджер, Администратор
6	Просмотр стран	Возможность просмотра стран.	Гость, Пользователь, Менеджер, Администратор
8	Подбор маршрутов по опросу	Возможность просмотра списка маршрутов, соответствующих результатам опроса	Гость, Пользователь, Менеджер, Администратор
9	Подбор маршрутов с помощью ИИ	Возможность просмотра списка маршрутов, соответствующих введённому описанию	Гость, Пользователь, Менеджер, Администратор
10	Оставление отзывов к турам	Возможность оставить отзыв к выбранному туру	Пользователь
11	Подача заявки на бронирование тура	Возможность отправить заявку на бронирование выбранного тура	Пользователь
12	Оплата забронированного тура	Возможность оплатить подтверждённый забронированный тур	Пользователь
13	Просмотр истории бронирований	Возможность просмотра списка броней	Пользователь
14	Отмена заявки на бронирование тура	Возможность отменить бронь тура	Пользователь, Менеджер, Администратор
15	Редактирование профиля	Возможность изменения контактных данных своей учётной записи	Пользователь, Менеджер, Администратор
16	Добавление туров	Возможность создания новых туров	Менеджер, Администратор
17	Редактирование туров	Возможность изменения данные существующих туров.	Менеджер, Администратор
18	Удаление туров	Возможность удаления существующих туров	Менеджер, Администратор
19	Добавление отелей	Возможность создания новых отелей	Менеджер, Администратор
20	Редактирование отелей	Возможность изменение данных существующих отелей.	Менеджер, Администратор
21	Удаление отелей	Возможность удаления существующих отелей	Менеджер, Администратор

Продолжение таблицы 2.2

№	Функция	Описание	Доступно ролям
22	Добавление пунктов отправлений	Возможность создания новых пунктов отправления	Менеджер, Администратор
23	Редактирование пунктов отправлений	Возможность изменение данных существующих пунктов отправлений.	Менеджер, Администратор
24	Удаление пунктов отправлений	Возможность удаления существующих пунктов отправлений	Менеджер, Администратор
25	Добавление стран	Возможность изменение данных существующих стран.	Менеджер, Администратор
26	Редактирование стран	Возможность удалять любые комментарии	Менеджер, Администратор
27	Удаление стран	Возможность удаления существующих стран	Менеджер, Администратор
28	Смена цены забронированного тура	Возможность изменения цены у забронированного тура	Менеджер, Администратор
29	Удаление отзывов к турам	Возможность удаления у туров отзывов, нарушающих правила	Администратор
30	Смена роли пользователей	Возможность блокировки у пользователя всех возможностей, кроме	Администратор
31	Разблокировка пользователей	Возможность восстановления прав пользователям	Администратор
32	Удаление пользователей	Возможность удаление существующих учётных записей	Администратор
33	Блокировка пользователей	Блокировка у пользователя всех возможностей, кроме просмотра туров	Администратор

Функции приложения разделены между ролями таким образом, чтобы обеспечить удобство использования для каждой категории пользователей. Например, гости могут только просматривать и искать туры, что снижает нагрузку на систему, так как им не нужно хранить данные о своих действиях. Пользователи, в свою очередь, обладают более широким спектром возможностей, включая бронирование туров. Менеджеры предоставляют пользователям туры и принимают заявки. Администраторы сосредоточены на обеспечении адекватного поведения пользователей

Диаграмма вариантов использования позволяет визуализировать все основные функции и роли веб-приложения. Разделение на роли обеспечивает структурированный доступ к функционалу и улучшает безопасность приложения. Таблицы с описанием ролей и функций дополняют диаграмму, предоставляя детальное представление о возможностях каждой категории пользователей.

## 2.2 Логическая схема базы данных

Логическая схема базы данных приведена в ДП 02.00.ГЧ. База данных содержит 8 таблиц, описание таблиц базы данных предоставлено в таблице 2.3.

Таблица 2.3 – Описание таблиц базы данных

Таблица	Описание
Roles	Хранит данные о ролях пользователей.
Users	Хранит данные о пользователях.
Tours	Хранит данные о турах.
Routes	Хранит данные о маршрутах.
Bookings	Хранит данные о бронях
TourTypes	Хранит данные о типах туров.
TourCharacteristics	Хранит данные о характеристиках туров.
TourDescriptions	Описывает связь между характеристиками и турами.
NutritionTypes	Хранит данные о типах питания.
Hotels	Хранит данные о отелях.
Cities	Хранит данные о городах
Countries	Хранит данные о странах.
Regions	Хранит данные о регионах мира.
DepartmentDepartures	Хранит данные о пунктах отправления.
Reviews	Хранит данные о отзывах к турам
TransportTypes	Хранит данные о типах транспортов
RoomTypes	Хранит данные о типах номеров отелей.
RoomTypeCharacteristics	Хранит данные о характеристиках типов номеров.
RoomTypeDescriptions	Описывает связь между характеристиками и типов номеров.
HotelCharacteristics	Хранит данные о характеристиках отелей.
HotelDescriptions	Описывает связь между характеристиками и отелями.
BookedRoomTypes	Хранит данные о забронированных типах номеров отелей.
Landmarks	Хранит данные о достопримечательностях.
Climates	Хранит данные о климатах.

Все таблицы разработаны для максимального эффективного хранения данных веб-приложения.

## 2.3 Архитектура веб-приложения

Диаграмма развертывания веб-приложения приведена в ДП 03.00.ГЧ.

Веб-приложение состоит из клиентской и серверной части. Пояснение назначения каждого элемента веб-приложения представлено в таблице 2.4.

Таблица 2.4 – Назначение элементов архитектурной схемы веб-приложения

Элемент	Назначение
Client Browser/Edge	Браузер пользователя, через который он взаимодействует с веб-приложением. Обрабатывает и отображает фронтенд, отправляет HTTP-запросы к Backend Server
Database Server	Сервер с базой данных. Отвечает за хранение данных
Web API Server	Сервер, который обрабатывает запросы от фронтенда, управляет бизнес-логикой приложения, взаимодействует с базой данных

Описание протоколов, используемых при работе веб-приложений, представлено в таблице 2.5.

Таблица 2.5 – Описание используемых протоколов

Протокол	Назначение
HTTPS [13]	Обмен данными между серверной и клиентской части веб-приложения. Обеспечивает безопасную передачу данных путём использования криптографического протокола TLS.
TCP [14]	Обмен данными между Database Server и Backend Server.

Данная архитектура обеспечивает максимально эффективную работу веб-приложения.

## 2.4 Блок-схема алгоритма бронирования тура

Блок-схема алгоритма бронирования тура приведена в ДП 04.00.ГЧ.

Данная блок-схема описывает полный процесс бронирования тура от выбора тура до его оплаты. После выбора тура, необходимо указать конкретный маршрут и заполнить форму оформления бронирования, в которой необходимо указать количество заказываемых номеров в отеле, количество людей, отправляющихся, в тур, при необходимости, указать наличие детей и выбрать приоритет рассадки в транспорте. При необходимости, можно оставить свои пожелания в поле комментария. После чего, отправить заявку на бронирование тура. Заявка будет отправлена в том случае, если пользователь авторизован, имеет роль «Пользователь», не заблокирован и данные введены корректно. В случае несоответствия одному из этих условий, выведется сообщение об ошибке. Если всё прошло успешно, то пользователю необходимо дождаться подтверждения брони менеджером. После подтверждения, необходимо выбрать данную бронь и оплатить её. После оплаты, тур будет считаться полностью забронированным.



## 2.5 Выводы по разделу

Таким образом в рамках проектирования веб-приложения было выполнено.

1. Разработана диаграмма вариантов использования, благодаря которой был определён какой функционал требуется приложению, а также какие роли будут иметь к нему доступ.

2. Спроектирована архитектура веб-приложения. Было определено каким образом будут размещены компоненты веб-приложения и какие протоколы будут использоваться.

3. Спроектирована база данных веб-приложения. Были определены таблицы, а также связи между ими.

4. Спроектирована блок схема алгоритма бронирования тура.

### 3 Разработка веб-приложения

Программное обеспечение – это больше, чем просто программный код. Программа представляет собой исполняемый код, который выполняет некоторые вычислительные задачи. Программное обеспечение считается коллекцией исполняемого программного кода, связанных с кодом библиотек и документации. Программное обеспечение, если оно изготовлено для конкретного требования, называется программным продуктом.

Разработка любого программного продукта – сложный и трудоемкий процесс, в ходе которого необходимо концентрироваться на большом количестве деталей, чтобы не допустить какую-либо критическую ошибку. Важно также следовать тем правилам, которые были заложены в ходе проектирования системы, ведь приложение, реализованное в соответствии с грамотно спроектированным решением – залог успеха всего проекта.

В этом разделе будет рассмотрен процесс разработки системы, в соответствии с проектными требованиями продукта, описанными в разделе 2 данного дипломного проектирования.

#### 3.1 Разработка серверной части веб-приложения

##### 3.1.1 Используемые библиотеки

Описание библиотек предоставлено в таблице 3.1.

Таблица 3.1 – Серверные зависимости

Библиотека	Описание
Microsoft.AspNetCore	Базовые компоненты фреймворка .NET Core.
Microsoft.AspNetCore.Session,	Библиотека, обеспечивающая работу сессий.
Microsoft.AspNetCore.StaticFiles	Библиотека, обеспечивающая работу с файлами и основные веб-функции.
Microsoft.AspNetCore.Authentication.JwtBearer	Библиотека для аутентификации пользователей с использованием JWT.
System.IdentityModel.Tokens.Jwt	Библиотека для работы с JWT-токенами.
Pomelo.EntityFrameworkCore.MySql	Библиотека ORM Entity Framework Core для взаимодействия с базой данных.
Newtonsoft.Json	Библиотека для работы с данными в JSON формате.
Stripe.net	Библиотека для взаимодействия с платёжной системой.

					ДП 03.00.ПЗ		
		ФИО	Подпись	Дата	3 Разработка веб-приложения		
Разраб.	Дмитрук И.И.						
Пров.	Белодед Н.И.						
Н. контр.	Белодед Н.И.						
Утв.	Смелов В.В.						
					Лит.	Лист	Листов
					У	1	14
					БГТУ 1-40 01 01, 2025		

Выбранные библиотеки содержат весь необходимый инструментарий для разработки веб-приложения.

3.1.2 Структура серверной части

Для разработки серверной части использовалась монолитная архитектура, где сервер состоит из одного приложения, которое отвечает за обработку со всей серверной части.

Для обеспечения упорядоченности серверного кода, было решено отказаться от простейшей монолитной архитектуры в пользу довольно простой N-layer [15] архитектуры, с выделением трех основных слоев: уровень доступа к данным, уровень бизнес-логики, уровень представления.

Структура проекта предоставлена на рисунке 3.1.

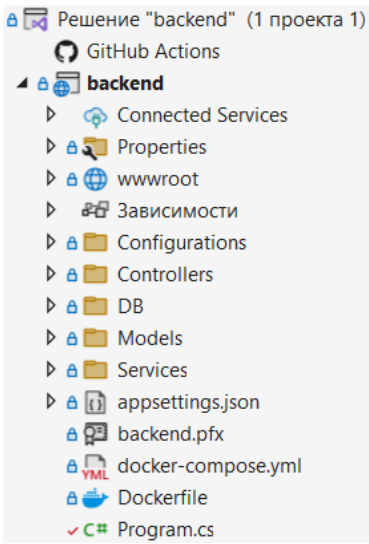


Рисунок 3.1 – Структура проекта

Описание папок входящих в структуру проекта предоставлено в таблице 3.2.

Таблица 3.2 – Папки входящие в проект

Папка	Описание
Configurations	Содержит файлы с конфигурациями, например, для настройки аутентификации с помощью JWT токенов.
Controllers	Содержит контроллеры, которые обрабатывают HTTP-запросы и возвращают ответы клиенту.
DB	Содержит контекст базы данных для Entity Framework.
Models	Содержит все виды моделей: DTO, данные из форм и модели Entity Framework.
Services	Содержит файлы, которые реализуют бизнес-логику для вспомогательных действий в контролерах.
wwwroot	Содержит статические файлы, необходимые для работы клиентской части (bundle.js)

За уровень доступа к данным отвечает папка. За уровень бизнес-логики отвечают папки «Services» и «Controllers». За уровень представления отвечает папка «Controllers».

### 3.1.3 Разработка уровня доступа к данным

На уровне доступа к данным система должна обеспечивать удобный и безопасный способ взаимодействия с данными, гарантируя целостность, производительность и масштабируемость.

Модели данных (или сущности) представляют собой абстракции, которые отражают структуру данных в базе данных. Каждая сущность обычно соответствует таблице в реляционной базе данных, а её свойства – столбцам этой таблицы. В листинге 3.1 предоставлен пример сущности.

```
public class Tour
{
    [Key]
    public int Id { get; set; }
    [Required]
    [MaxLength(255)]
    public string Name { get; set; }
    [MaxLength(255)]
    public string MainDescription { get; set; }
    public int? TourTypeId { get; set; }
    public int? HotelId { get; set; }
    [ForeignKey(nameof(TourTypeId))]
    public TourType TourType { get; set; }
    [ForeignKey(nameof(HotelId))]
    public Hotel Hotel { get; set; }
    public ICollection<TourCharacteristic> Characteristics {
get; set; }
    public ICollection<Route> Routes { get; set; }
    public ICollection<Review> Reviews { get; set; }
}
```

Листинг 3.1 – Сущность Tour

EF автоматически генерирует SQL-запросы на основе моделей данных, обеспечивая их выполнение в базе данных MariaDB. В листинге 3.2 приведён пример получения объектов при помощи Entity Framework Core.

```
List<Tour> tours = await db.Tours
    .Include(t => t.TourType)
    .Include(t => t.Hotel)
    .ThenInclude(h => h.City)
    .ThenInclude(c => c.Country)
    .ToListAsync();
```

Листинг 3.2 – Пример выполнения запроса в базу данных

Таблица с описанием всех сущностей, их типами данных и соответствующими типами данных в SQL представлены в приложении А. В приложении Б – скрипт для создания таблиц.

### 3.1.4 Разработка уровня бизнес-логики

При разработке бизнес-логики для WebAPI на C# одним из ключевых аспектов является правильное использование сервисов и паттернов внедрения зависимостей.

В основном бизнес-логика реализована в контроллерах. Реализация выполнения некоторых задач, которые выполняются во многих местах, были вынесены в сервисы. Данные сервисы внедряются в контролеры при помощи механизма DI [16], который автоматически управляет созданием объектов и их зависимостями. Контроллеры могут получать сервисы через конструктора. Пример внедрения сервисов для определения погоды в туре, взаимодействия с ИИ и взаимодействия с контекстом базы данных, предоставлен в листинге 3.4.

```
private AppDbContext db;
private IWeatherService _weatherService;
private IAIService _aiService;

public TourController(AppDbContext context, IWeatherService
weatherService, IAIService aiService)
{
    db = context;
    _weatherService = weatherService;
    _aiService = aiService;
}
```

Листинг 3.4 – пример передачи зависимостей через конструктор

Таким образом было организовано использование бизнес-логики.

### 3.1.5 Разработка уровня представления

Разработка уровня представления является важным этапом в создании веб-приложений, так как именно через этот слой взаимодействуют пользователи с системой.

Контроллеры в контексте разработки веб-приложений с использованием ASP.NET Core являются классами, которые обрабатывают HTTP-запросы и отвечают на них соответствующими данными. Эти классы обычно наследуются от базового класса Controller, который предоставляет доступ к набору стандартных методов для обработки запросов. Каждый контроллер связан с определённым маршрутом, который указывает на URL, к которому привязан данный контроллер.

Пример контроллера предоставлен в листинге 3.5.

```
[Route("api/[controller]")]
[Route("review")]
public class ReviewController : Controller
{
    private AppDbContext db;

    public ReviewController(AppDbContext context)
    {
        db = context;
    }

    [HttpGet("reviews")]
    public async Task<IActionResult> GetReviews([FromQuery] int?
tourId)
    {...}

    [Authorize(Roles = "User")]
    [HttpPost("add")]
    public async Task<IActionResult> AddReview([FromBody]
ReviewForm review)
    {...}

    [Authorize(Roles = "Admin")]
    [HttpDelete("delete")]
    public async Task<IActionResult> DeleteReview([FromQuery]
int? tourId)
    {...}
}
```

Листинг 3.5 – Пример контроллера

Контроллеры могут выполнять бизнес-логику, однако реализация выполнения частых задач, должна быть вынесена в сервисы.

### 3.1.6 Настройка безопасности

Одной из распространенных задач в рамках реализации веб-приложения, в котором поддерживаются пользовательские учетные записи – реализация механизма аутентификации и авторизации, чаще всего используется механизм на основе токенов.

Создание токенов обычно осуществляется с использованием библиотеки System.IdentityModel.Tokens.Jwt, которая предоставляет все необходимые инструменты для работы с JSON Web Token (JWT) [17]. Access token представляет собой короткоживущий токен, который используется для предоставления доступа к защищённым ресурсам.

Основными параметрами при создании JWT-токенов являются issuer (издатель токена), audience (аудитория токена), claims (утверждения,

содержащие информацию о пользователе), и `expiration` (время истечения токена). Например, `issuer` используется для указания сервера, который выдал токен, а `audience` определяет, для какого ресурса предназначен токен. `Claims` включают такие данные, как идентификатор пользователя, его роль или другие атрибуты. `Expiration` задаёт ограничение по времени, что делает токен недействительным после периода.

Код создание токенов предоставлен в листинге 3.6.

```
public static string GenerateToken(string email, UserRole role)
{
    List<Claim> claims = new List<Claim> { new
Claim(ClaimTypes.Email, email), new Claim(ClaimTypes.Role,
role.ToString()) };

    JwtSecurityToken token = new JwtSecurityToken(
        issuer: AuthOptions.ISSUER,
        audience: AuthOptions.AUDIENCE,
        claims: claims,
        expires: DateTime.UtcNow.Add(TimeSpan.FromHours(8)),
        signingCredentials: new
SigningCredentials(AuthOptions.GetSymmetricSecurityKey(),
SecurityAlgorithms.HmacSha256)
    );

    return new JwtSecurityTokenHandler().WriteToken(token);
}
```

Листинг 3.6 – Код создание токенов

Для проверки авторизации пользователей, используется атрибут для методов `[Authorize]`, который проверяет пользователя на авторизацию, а также если указать параметр `Role`, можно проверить принадлежность пользователю роли.

Таким образом, настройка безопасности через токены позволяет создать эффективную и гибкую систему управления доступом в ASP.NET Core. Это решение обеспечивает высокий уровень защиты, удобство для пользователей и возможность масштабирования приложения.

### 3.1.7 Используемые паттерны

Использование ASP .NET Core в качестве фреймворка для разработки приложений влечет за собой необходимость применения паттернов проектирования.

В данном проекте использовался паттерн `dependency Injection (DI)` — паттерн, который помогает разделить создание зависимостей и их использование. В ASP.NET Core встроена мощная система DI, которая позволяет инжектировать сервисы, такие как репозитории, логеры или контексты баз данных, прямо в контроллеры и другие компоненты через

конструктор. Это способствует лучшей модульности, тестируемости и снижению связности между компонентами системы

## 3.2 Клиентская часть

### 3.2.1 Используемые библиотеки

Описание библиотек предоставлено в таблице 3.3.

Таблица 3.3 – Клиентские библиотеки

Библиотеки	Описание
react, react-dom, react-router-dom	Базовые библиотеки для построения SPA-интерфейса.
@mui/material, @mui/icons-material, @mui/lab,	Визуальные компоненты и иконки для создания интерфейса.
axios	HTTPS-клиент для общения с сервером
reduxjs/toolkit, react-redux, redux	Библиотеки для управления глобальным состоянием приложения.
stripe/react-stripe-js, stripe/stripe-js	Библиотеки для отображения UI элементов и работы с данными платёжной системы Stripe
dayjs	Библиотека для упрощения работы с датами.
react-leaflet	Библиотека, которая позволяет удобно использовать интерактивные карты

Данные библиотеки полностью закрывают потребности клиентской части.

### 3.2.2 Структура клиентской части

Структура проекта организованна таким образом, чтобы все файлы были сгруппированы внутри корневой папки src в директории по их назначению. Структура клиентской части предоставлена на рисунке 3.2.

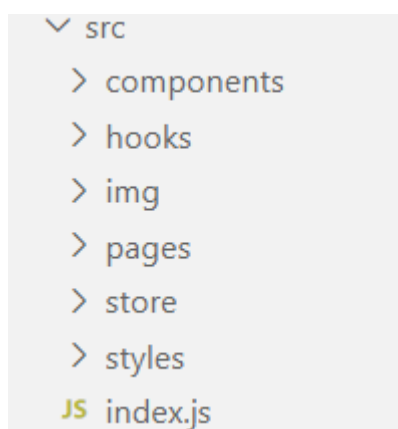


Рисунок 3.2 – Структура клиентской части



В таблице 3.4 приведен список основных директорий проекта и их назначение.

Таблица 3.4 – Директории входящие в проект

Директория	Описание
components	Содержит компоненты элементов страниц.
hooks	Содержит дополнительные хуки
img	Содержит изображения элементов, используемых на страницах
pages	Содержит компоненты страниц.
store	Содержит глобальные данные, которые могут использоваться во всех компонентах
styles	Содержит файлы стилей

Данная структура обеспечивает удобный доступ ко всем файлам проекта.

### 3.2.3 Конфигурация проекта

Для настройки и сборки клиентской части веб-приложения был выбран сборщик WebPack, который является одним из самых популярных и мощных сборщиков для JavaScript-приложений. WebPack позволяет настроить процесс сборки таким образом, чтобы эти ресурсы правильно компилировались, объединялись в пакеты и оптимизировались для быстрого загрузки в браузере.

## 3.3 Реализация функций

### 3.3.1 Регистрация

Метод «Register» отвечает за регистрацию нового пользователя. Сначала он проверяет, существует ли пользователь с переданным в метод email, если существует, то возвращает ответ с 409 статусом. Если нет, то хеширует переданный в метод пароль сохраняет учётные данные пользователя в базу данных. Затем возвращает клиенту токен.

### 3.3.2 Авторизация

Если у пользователя нет токена, ему необходимо авторизоваться. При авторизации, запрос будет отправлен к методу «Login», которые будет искать учётную запись в базе данных, соответствующих переданных в него логину и паролю, и в случае, если запись будет найдена, то пользователю вернётся токен, иначе ответ со статусом 401.

Если у пользователя есть токен, то запрос будет отправлен к методу «Auth», предназначенного для предоставления пользователю его учётных данных по токену. У метода установлен атрибут Authorize, который проверяет корректность полученного токена. Метод получает из базы данных данные

пользователя по его email, полученного из токена и возвращает эти данные клиенту. Если пользователь не будет найден, или токен не корректен, то возвращает ответ с 401 статусом.

Листинг реализации метода предоставлен в приложении В.

### **3.3.3 Просмотр маршрутов**

Метод «GetTours» возвращает список всех туров с маршрутами. Он получает из базы данных все туры с их маршрутами, у которых ещё не закончились места, и дата отправления больше сегодняшней, фильтрует их по количеству оставшихся мест и возвращает их. Листинг реализации методов «Get» и «StreamVideo» предоставлен в приложении В.

### **3.3.4 Просмотр отелей**

Метод «GetHotels» возвращает список всех отелей. Он получает из базы данных все отели и возвращает их. Метод «GetHotel» возвращает данные отеля по ID. Он получает из базы данных данные конкретного отеля и возвращает их. Листинг с реализацией метода предоставлен в приложении В.

### **3.3.5 Просмотр стран**

Метод «GetCountries» возвращает список всех стран. Он получает из базы данных все страны и возвращает их. Метод «GetCountry» возвращает данные страны по ID. Он получает из базы данных данные конкретной страны и возвращает их. Листинг с реализацией метода предоставлен в приложении В.

### **3.3.6 Фильтрация маршрутов**

Метод «GetFiltredTours» возвращает список отфильтрованных туров с их маршрутами. Он получает из базы данных все туры с маршрутами, затем выбирает те туры, данные которых соответствуют данным фильтра, переданных в метод, и возвращает эти туры. Реализация метода предоставлена в листинге 3.8.

### **3.3.7 Подбор маршрутов по опросу**

Метод «GetToursBySurvey» получает объект с данными результата опроса. Затем получает из базы данных все маршруты, отбирает из них те, характеристики и данные которых соответствуют результатам опроса и возвращает их клиенту. Листинг с реализацией метода предоставлен в приложении В.

### 3.3.8 Подбор маршрутов с помощью ИИ

Метод «GetToursByPromptToAi» получает текстовое описание городов или местностей. Затем отправляет это описание сервису Gemini, вместе с промптом, указывающий ИИ, что бы он подобрал города, подходящие под описание. ИИ подбирает города, и Gemini высылает их обратно на сервер. Сервер ищет в базе данных туры с полученными города и высылает их пользователю. Подбор туров с помощью ИИ представлен в диаграмме последовательности, которая приведена в ДП 05.00.ГЧ. Листинг с реализацией метода предоставлен в приложении В.

### 3.3.9 Оставление отзывов к турам

Метод «AddReview» добавляет новый отзыв к определённому туру. Метод получает текст отзыва, ID пользователя и ID тура и добавляет эти в базу данных. Листинг с реализацией метода «Delete» предоставлена в приложении В.

### 3.3.10 Подача заявки на бронирование тура

Метод «AddBooking» создаёт новую бронь. Метод принимает id пользователя и маршрута, и проверяет, есть ли в базе данных бронь тура по данному маршруту у данного пользователя, если есть, то метод возвращает ответ со статусом 409. Затем проверяет есть ли данный маршрут, если нет, то возвращает ответ со статусом 400, затем проверяет есть ли данный пользователь, если нет, то возвращает ответ со статусом 401. Если все проверки пройдут успешно, то метод получает из базы данных маршрут и отнимает у него от количества мест количество заказанных мест, переданных в метод. Если результат меньше нуля, то возвращает ответ со статусом 409, если нет, то бронь успешно добавляется в базу данных вместе с бронями номеров в отеле.

### 3.3.11 Оплата забронированного тура

Метод «CreatePaymentIntent» получает данные для оплаты. Затем отправляет эти данные платёжной системе Stripe. Stripe возвращает секретный ключ для клиента. Сервер возвращает этот ключ клиенту. Клиент, при помощи секретного ключа, отправляет данные банковской карты платёжной системе Stripe. Он выполняет банковские операции и результат возвращает клиенту. В случае успеха, клиент отправляет запрос методу «ConfirmPayment» с ID брони. Метод изменяет у брони с данным ID статус на оплачено. Листинг с реализацией метода предоставлен в приложении В.

### **3.3.12 Просмотр истории бронирований**

Метод «GetBookings» возвращает список броней. Метод принимает id пользователя и получает из базы данных брони пользователя и возвращает их. Так же метод может принимать параметр isHistory. В данном параметре установлено значение по умолчанию false. Если пользователь передаст значение true, то метод вернёт брони туров, в которые пользователь уже отправлялся, то есть у которых дата и время отправления меньше текущего и который был оплачен. Реализация метода для загрузки видео на сервер предоставлен в листинге 3.12.

### **3.3.13 Отмена заявки на бронирование тура**

Метод «DeleteBooking» удаляет существующую бронь. Метод принимает id брони и удаляет бронь с соответствующим id в базе данных. Так же восстанавливает количество доступных мест в маршруте.

### **3.3.14 Редактирование профиля**

Метод «EditUser» изменяет данные профиля. Метод принимает профиля и изменяет их у пользователя в базе данных, id которого соответствует переданному в метод id.

### **3.3.15 Добавление туров**

Метод «AddTour» создаёт новый тур. Метод принимает данные тура и сохраняет их в базу данных.

### **3.3.16 Редактирование туров**

Метод «EditTour» изменяет существующий тур. Метод изменяет данные у тура в базе данных на данные, переданные в метод. Изменения данных происходит у того тура, id которого соответствует id, переданного в метод.

### **3.3.17 Удаление туров**

Метод «DeleteTour» удаляет существующий тур. Метод принимает id тура и удаляет тур с соответствующим id в базе данных вместе со всеми сущностями, соединённых с туром.

### **3.3.18 Добавление отелей**

Метод «AddHotel» создаёт новый отель. Метод принимает данные отеля и сохраняет их в базу данных.

### **3.3.19 Редактирование отелей**

Метод «EditHotel» изменяет существующий отель. Метод изменяет данные у отеля в базе данных на данные, переданные в метод. Изменения данных происходит у того отеля, id которого соответствует id, переданного в метод.

### **3.3.20 Удаление отелей**

Метод «DeleteHotel» удаляет существующий отель. Метод принимает id отеля и удаляет отель с соответствующим id в базе данных вместе со всеми сущностями, соединённых с отелем.

### **3.3.21 Добавление пунктов отправлений**

Метод «AddDepartmentDeparture» создаёт новый пункт отправления. Метод принимает данные пункта отправления и сохраняет их в базу данных.

### **3.3.22 Редактирование пунктов отправлений**

Метод «EditDepartmentDeparture» изменяет существующий пункт отправления. Метод изменяет данные у пункта отправления в базе данных на данные, переданные в метод. Изменения данных происходит у того пункта отправления, id которого соответствует id, переданного в метод.

### **3.3.23 Удаление пунктов отправлений**

Метод «DeleteDepartmentDeparture» удаляет существующий пункт отправления. Метод принимает id пункта отправления и удаляет пункт отправления с соответствующим id в базе данных вместе со всеми сущностями, соединённых с пунктом отправления.

### **3.3.24 Добавление стран**

Метод «AddCountry» создаёт новую страну. Метод принимает данные страны и сохраняет их в базу данных.

### **3.3.25 Редактирование стран**

Метод «EditCountry» изменяет существующий страну. Метод изменяет данные у страны в базе данных на данные, переданные в метод. Изменения данных происходит у той страны, id которой соответствует id, переданного в метод.

### **3.3.26 Удаление стран**

Метод «DeleteCountry» удаляет существующую стран. Метод принимает id страны и удаляет страну с соответствующим id в базе данных вместе со всеми сущностями, соединённых со страной.

### **3.3.27 Смена цены забронированного тура**

Метод «ChangePrice» изменяет итоговую цену забронированного тура. Метод принимает id брони и цену, затем изменяет в базе данных цену у брони, id которой соответствует переданному id, на переданную цену.

### **3.3.28 Удаление отзывов к турам**

Метод «DeleteReview» удаляет существующий отзыв к туру. Метод принимает id отзыва и удаляет отзыв с соответствующим id из базы данных.

### **3.3.29 Смена роли пользователей**

Метод «ChangeRole» изменяет роль у пользователя. Метод принимает id пользователя и id роли, затем в базе данных изменяет у пользователя, id которого соответствует переданному id, id роли на переданную id роли.

### **3.3.30 Разблокировка пользователей**

Метод «BlockUser» изменяет статус блокировки пользователя на противоположный. Метод принимает id пользователя и изменяет статус блокировки у соответствующего пользователя. Если статус блоки true, то после выполнения метода, станет false.

### **3.3.31 Удаление пользователей**

Метод «DeleteUser» удаляет существующего пользователя. Метод принимает id пользователя и удаляет пользователя с соответствующим id в базе данных.

### **3.3.32 Блокировка пользователей**

Метод «BlockUser» изменяет статус блокировки пользователя на противоположный. Метод принимает id пользователя и изменяет статус блокировки у соответствующего пользователя. Если статус блоки false, то после выполнения метода, станет true.

### 3.4 Маршруты для контролеров

Все маршруты контроллеров указаны в таблице, которая предоставлена в приложении Г.

### 3.5 Выводы по разделу

Таким образом в рамках реализации веб-приложения было выполнено.

1. Реализована база данных с использованием СУБД MariaDB 15, обеспечивающая хранение и эффективную обработку данных всех сущностей приложения, включая роли, пользователей, туры, маршруты, отзывы, отели, пункты отправлений, города, страны, характеристики и другие связанные объекты;

2. Использован необходимый стек технологий, который состоит из 8 библиотек серверной части и 14 библиотек клиентской части, обеспечивающих надёжную архитектуру, масштабируемость и высокую производительность приложения;

3. Реализован весь основной функционал приложения, такой как: просмотр маршрутов и туров, бронирование туров, регистрация и авторизация пользователей, работа с турами, отелями, пунктами отправлений, городами с странами (добавление, редактирование, удаление). Также был внедрён функционал для совершения оплаты забронированных туров и подбора маршрутов с помощью ИИ;

4. Реализованы все необходимые компоненты на стороне клиента, навигационные элементы, модальные окна, формы и страницы для работы с контентом, а также компоненты для реализации функций взаимодействия с сервером через REST API.

## 4 Тестирование веб-приложения

### 4.1 Функциональное тестирование

Для тестирования функционала были разработаны специальные тесты, которые предоставлены в таблице 4.1.

Таблица 4.1 – Функциональные тесты

№	Название функции	Описание теста	Ожидаемый результат	Результат
1	Регистрация	Действие: отправить POST запрос на адрес /auth/register/, указав в теле запроса email «ilya@gmail.com», имя (name) «ilya», фамилию (surname) «dmitruk», пароль (password) «1234», номер телефона (phonePassword) «+336461866».	В базе данных появится учётные данные нового пользователя. Сервер вернёт токен.	Успешно
2	Авторизация	Действие: отправить POST запрос на адрес /auth/login/, указав в теле запроса email «ilya@gmail.com», пароль (password) «1234».	Сервер должен вернуть токен.	Успешно
3	Просмотр маршрутов	Действие: необходимо отправить GET-запрос на адрес tour/tours/	Сервер должен вернуть все доступные маршруты	Успешно
4	Просмотр отелей	Действие: необходимо отправить GET-запрос на адрес hotel/hotels/	Сервер должен вернуть все отели	Успешно
5	Просмотр стран	Действие: необходимо отправить GET-запрос на адрес direction/countries/	Сервер должен вернуть все страны	Успешно
6	Фильтрация маршрутов	Отправить POST-запрос на адрес /tour/filtred_tours, указав в теле запроса countryId «1».	Сервер должен вернуть маршруты, отель которого находится в стране с id, равного 1	Успешно
7	Подбор маршрутов по опросу	Отправить POST-запрос на адрес /tour/tours_by_survey, указав в теле запроса Biom «Пустынная местность».	Сервер должен вернуть маршруты с характеристикой «Пустынная местность»	Успешно

					ДП 04.00.ПЗ		
		ФИО	Подпись	Дата	4 Тестирование веб-приложения		
Разраб.	Дмитрук И.И.						
Пров.	Белодед Н.И.						
Н. контр.	Белодед Н.И.						
Утв.	Смелов В.В.				БГТУ 1-40 01 01, 2025		
					Лит.	Лист	Листов
					У	1	5



Продолжение таблиц 4.1

№	Название функции	Описание теста	Ожидаемый результат	Результат
8	Подбор маршрутов с помощью ИИ	Отправить POST-запрос на адрес /tour/tours_by_prompt_to_ai, указав в теле запроса text «Море, жара в январе».	Сервер должен вернуть маршруты с турами туда, где жарко в январе и есть море	Успешно
11	Оставление отзывов к турам	Отправить POST-запрос на адрес review/add, указав в теле userId «1», tourId «1», текст отзыва (ReviewText) «хороший тур».	В базе данных появится отзыв с текстом «хороший тур», у маршрута с id 1, связанный с пользователем с id 1	Успешно
12	Подача заявки на бронирование тура	Отправить POST-запрос на адрес booking/add, указав в теле количество заказываемых мест (orderSeatsNumber) «2», routeId «1», userId «1», Price «1200»,	В базе данных появится новая бронь тура, с количеством заказанных мест равного 2, и связанный с маршрутом с id 1 и пользователем с id 1 и ценой в 1200 byn	Успешно
13	Оплата забронированного тура	Перейти в "Студию". Выбрать видео. Нажать «Редактировать». Изменить название/описание/теги. Нажать «Сохранить».	Изменения вступили в силу.	Успешно
14	Просмотр истории бронирований	Действие: необходимо отправить GET-запрос на адрес booking/delete?и=1	Сервер должен вернуть брони пользователя, id которого равен 1	Успешно
15	Отмена заявки на бронирование тура	Действие: необходимо отправить DELETE-запрос на адрес tour/delete?bookingId=1	В базе данных удалиться бронь с id 1.	Успешно
16	Редактирование профиля	Отправить PUT-запрос на адрес user/edit, указав в теле id «1» name «ilya», surname «dmitruk», phoneNumber «336461866».	В базе данных у пользователя, с id 1 изменится имя на «ilya», фамилия на «dmitruk», номер телефона на «336461866»	Успешно

Продолжение таблиц 4.1

№	Название функции	Описание теста	Ожидаемый результат	Результат
19	Добавление туров	Отправить POST-запрос на адрес tour/add, указав в теле name «mmm», MainDescription «nnn», hotelId «1», tourTypeId «1», nutritionTypeId «1».	В базе данных появится тур с названием «mmm», основным описанием «nnn», с отелем, типом тура, типом питания, id которых 1.	Успешно
20	Редактирование туров	Отправить PUT-запрос на адрес tour/edit, указав в теле id «1», name «mmm», MainDescription «nnn», hotelId «1», tourTypeId «1», nutritionTypeId «1».	В базе данных у тура, с id 1 изменится название на «mmm», основное описание на «nnn», id отеля, типа тура, типа питания на 1.	Успешно
21	Удаление туров	Отправить DELETE-запрос на адрес tour/delete?tourId=1	В базе данных удалиться тур с id равного 1.	Успешно
22	Добавление отелей	Отправить POST-запрос на адрес hotel/add, указав в теле name «mmm», MainDescription «nnn», cityId «1»,	В базе данных появится отель с названием «mmm», основным описанием «nnn», с городом, id которого равен 1.	Успешно
23	Редактирование отелей	Отправить PUT-запрос на адрес hotel/edit, указав в теле id «1», name «mmm», MainDescription «nnn»	В базе данных у отеля, с id 1, изменится название на «mmm», основное описание на «nnn»	Успешно
24	Удаление отелей	Отправить DELETE-запрос на адрес hotel/delete?hotelId=1	В базе данных удалиться отель с id равного 1.	Успешно
25	Добавление пунктов отправлений	Отправить POST-запрос на адрес department_departure/add, указав в теле name «mmm», address «nnn», lat «50», lng «50», cityId «1», TransportTypeId «1»	В базе данных появится пункт отправления с названием «mmm», адресом «nnn», координатами 50 50, городом и	Успешно

			типом транспорта, id которых 1	
26	Редактирование пунктов отправлений	Отправить PUT-запрос на адрес department_departure/edit, указав в теле id «1», name «mmm», MainDescription «nnn»	Комментарий удалён.	Успешно
27	Редактирование пунктов отправлений	Отправить PUT-запрос на адрес department_departure/edit, указав в теле id «1», name «mmm»	В базе данных у пункта отправления, с id 1, изменится название на «mmm»	Успешно
28	Удаление пунктов отправлений	Отправить DELETE-запрос на адрес department_departure/delete? departmentDepartureId=1	В базе данных удалиться пункт отправления с id равным 1.	Успешно
29	Добавление стран	Отправить POST-запрос на адрес direction/ add_country, указав в теле name «Казахстан», lat «50», lng «50», RegionId «1», levelOfDevelopment «1»	В базе данных появится страна с названием «Казахстан», координатами столицы 50 50, id региона равного 1 и уровнем развития 1	Успешно
30	Редактирование стран	Отправить PUT-запрос на адрес direction/edit_country, указав в теле id «1», name «ЮАР»	В базе данных у страны, с id 1, изменится название на «ЮАР»	Успешно
31	Удаление стран	Отправить DELETE-запрос на адрес direction/delete_country? countryId=1	В базе данных удалиться страна с id равным 1.	Успешно
	Смена цены забронированного тура	Отправить PATCH-запрос на адрес booking/change_price? price=500&bookingId=1	В базе данных, у брони с id 1, сменится цена на 500	change_price
	Удаление отзывов к турам	Отправить DELETE-запрос на адрес review/delete?reviewId=1	В базе данных удалиться отзыв с id, равного 1.	
	Смена роли пользователей	Отправить PATCH-запрос на адрес users/change_role?userId=1&roleId=2	В базе данных, у пользователя с id 1, смениться id роли на 2 (Менеджер).	
	Разблокировка пользователей	Отправить PATCH-запрос на адрес users/block?userId=1	В базе данных, у пользователя 1 с id 1, сменится статус блокировки на	

			противоположны й	
	Удаление пользователей	Отправить DELETE-запрос на адрес users/delete?userId=1	В базе данных удалиться пользователь с id, равного 1.	
	Блокировка пользователей	Отправить PATCH-запрос на адрес users/block?userId=1	В базе данных, у пользователя 1 с id 1, сменится статус блокировки на противоположны й	

## 4.2 Нагрузочное тестирования

Для проверки стабильности сервера были разработаны специальные нагрузочные тесты, которые предоставлены в таблице 4.3.

Таблица 4.3 – Нагрузочные тесты

№	Описание	Результат
1	Массовое бронирование туров. Проверить производительность системы при одновременном бронировании разных туров большим числом пользователей.	Никаких сбоев
2	Массовое добавление отзывов к турам. Проверить устойчивость при большом количестве отзывов.	Никаких сбоев
3	Массовая оплата забронированных туров. Проверить производительность системы при одновременном оплате множества брони множества пользователей.	Никаких сбоев

Таким образом веб-приложение было протестировано на стрессоустойчивость.

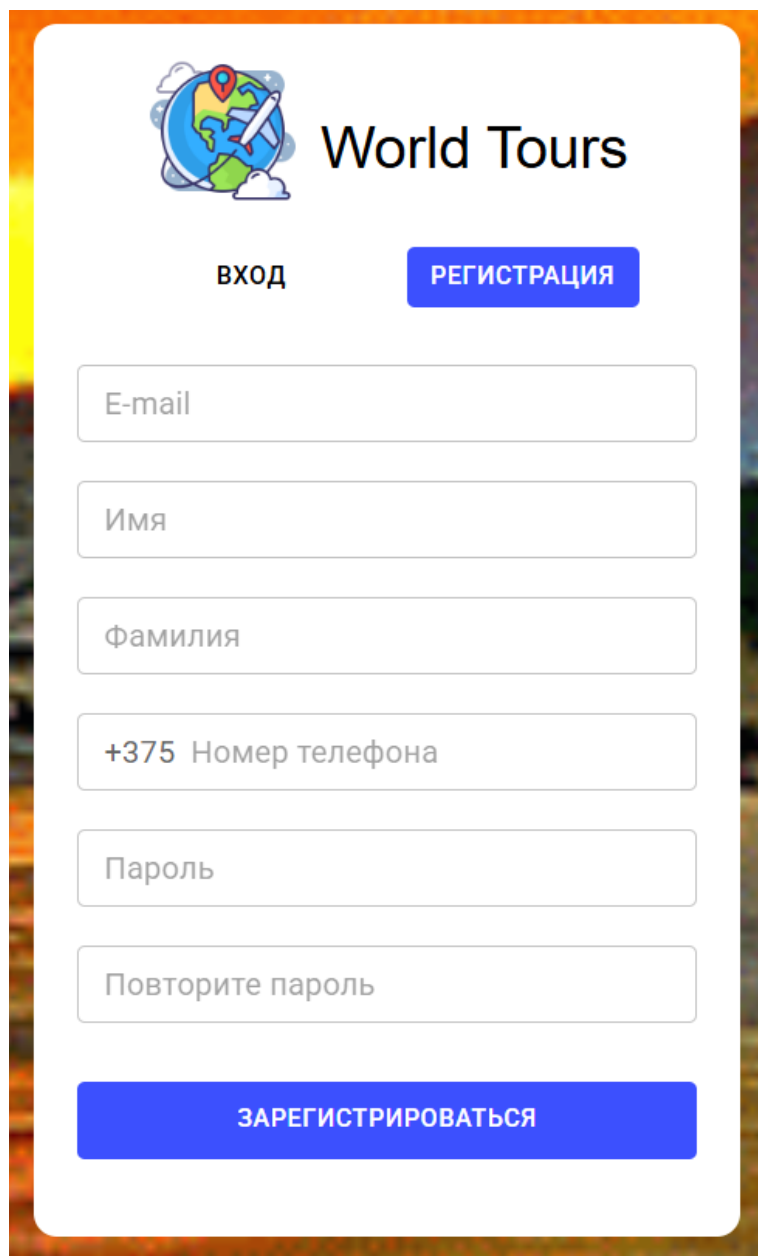
## 4.3 Выводы по разделу

В результате тестирования, веб-приложение для турагентства показало высокий уровень надежности, устойчивости и готовности к работе в реальных условиях эксплуатации. Выявленные дефекты были оперативно устранены, а результаты нагрузочного тестирования доказали способность системы справляться с большим числом пользователей и операций одновременно. Таким образом покрытость тестами составила около 80%. Это обеспечивает уверенность в том, что приложение удовлетворит ожидания пользователей, обеспечивая стабильность и удобство работы.

## 5 Руководство пользователя (руководство по эксплуатации)

### 5.1 Регистрация

Для регистрации необходимо нажать кнопку войти в правом верхнем углу главной страницы, выбрать регистрацию и в поля формы ввести email, имя, фамилию, номер телефона, пароль и повторный пароль. Форма регистрации представлена на рисунке 5.1.



The registration form for 'World Tours' is displayed within a white rounded rectangle with an orange border. At the top left is a logo featuring a globe with a red location pin and a white airplane. To the right of the logo is the text 'World Tours'. Below the logo and text are two buttons: 'ВХОД' (Login) and 'РЕГИСТРАЦИЯ' (Registration). The registration section contains six input fields: 'E-mail', 'Имя' (Name), 'Фамилия' (Surname), '+375 Номер телефона' (Phone number), 'Пароль' (Password), and 'Повторите пароль' (Repeat password). At the bottom of the form is a large blue button labeled 'ЗАРЕГИСТРИРОВАТЬСЯ' (Register).

Рисунок 5.1 – Форма регистрации

					ДП 05.00.ПЗ				
		ФИО	Подпись	Дата					
Разраб.	Дмитрук И.И.				5 Руководство пользователя (руководство по эксплуатации)		Лит.	Лист	Листов
Пров.	Белодед Н.И.						У	1	20
							БГТУ 1-40 01 01, 2025		
Н. контр.	Белодед Н.И.								
Утв.	Смелов В.В.								

После ввода данных, необходимо нажать кнопку «Зарегистрироваться», которая находится под формой. Затем выполнится валидация, и если она пройдет успешно, то в базе данных появятся ваши учетные данные, а вас перенаправит на главную страницу.

## 5.2 Авторизация

Если вы уже авторизовывались, то вам ничего делать не надо. Вы просто заходите на сайт, токен сам отправится на сервер и ваши учетные данные сами подгрузятся. Если же вы не авторизовывались, то вам необходимо нажать кнопку войти в правом верхнем углу главной страницы, и ввести в форму входа свои аутентификационные данные, такие как email и пароль. Форма входа с представлена на рисунке 5.2.

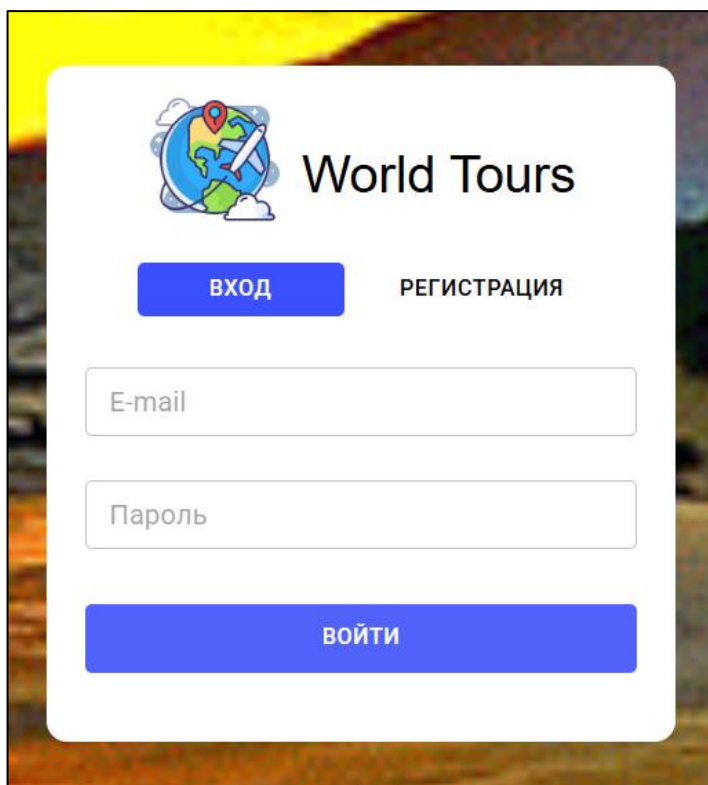


Рисунок 5.2 – Форма входа

После ввода данных, необходимо нажать кнопку «Войти», которая находится под формой. Затем на сервере выполнится поиск пользователя с ведённым email и паролем, и данные найденного пользователя отправляются вам, и вас перенаправит на главную страницу.

## 5.3 Просмотр маршрутов.

Для просмотра маршрутов вам достаточно просто зайти на главную страницу сайта, даже не обязательно авторизовываться. Перед вами

отобразиться список маршрутов, относящихся к разным турам. Главная страница с списком маршрутов представлена на рисунке 5.3.

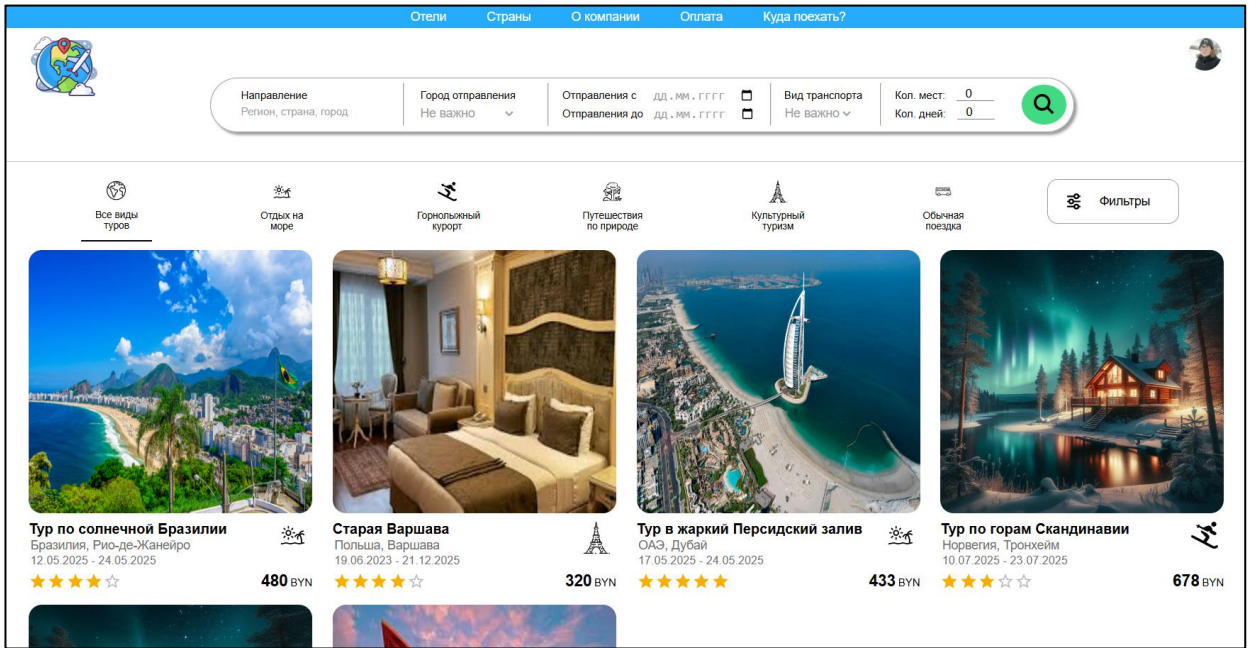


Рисунок 5.3 – Главная страница со списком маршрутов

Что бы просмотреть информацию о конкретном туре, вам необходимо нажать на маршрут, и вас перекинет на страницу тура с выбранным маршрутом. Страница тура представлена на рисунке 5.4.

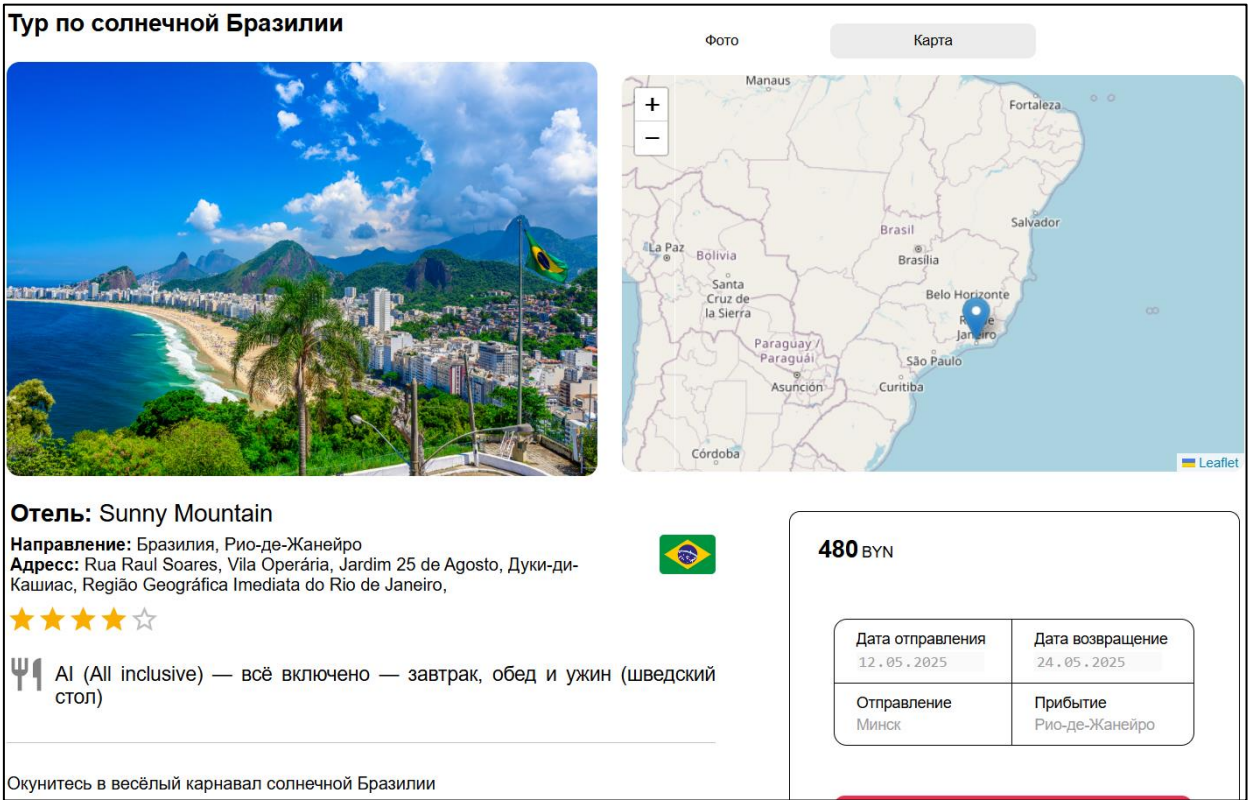


Рисунок 5.4 – Страница тура



Таким образом, можно узнать подробную информацию о туре и маршруте. Здесь можно увидеть фотографии с тура, просмотреть месторасположение отеля на карте, узнать, какие достопримечательности есть в данном туре и просто просмотреть характеристики тура. Так же, на этой странице можно нажать по названию отеля, названию города, названию страны или по флагу этой страны, и перейти на страницы с подробной информацией о них. С этой страницы необходимо совершать бронирование тура.

## 5.4 Просмотр отелей

Для того что бы увидеть список отелей, необходимо в верхнем навигационном меню нажать на «Отели». Вы перейдёте страницу с отелями. На ней можно отфильтровать список отелей по месторасположению или по звёздам. Страница со списком отелей представлена на рисунке 5.5.

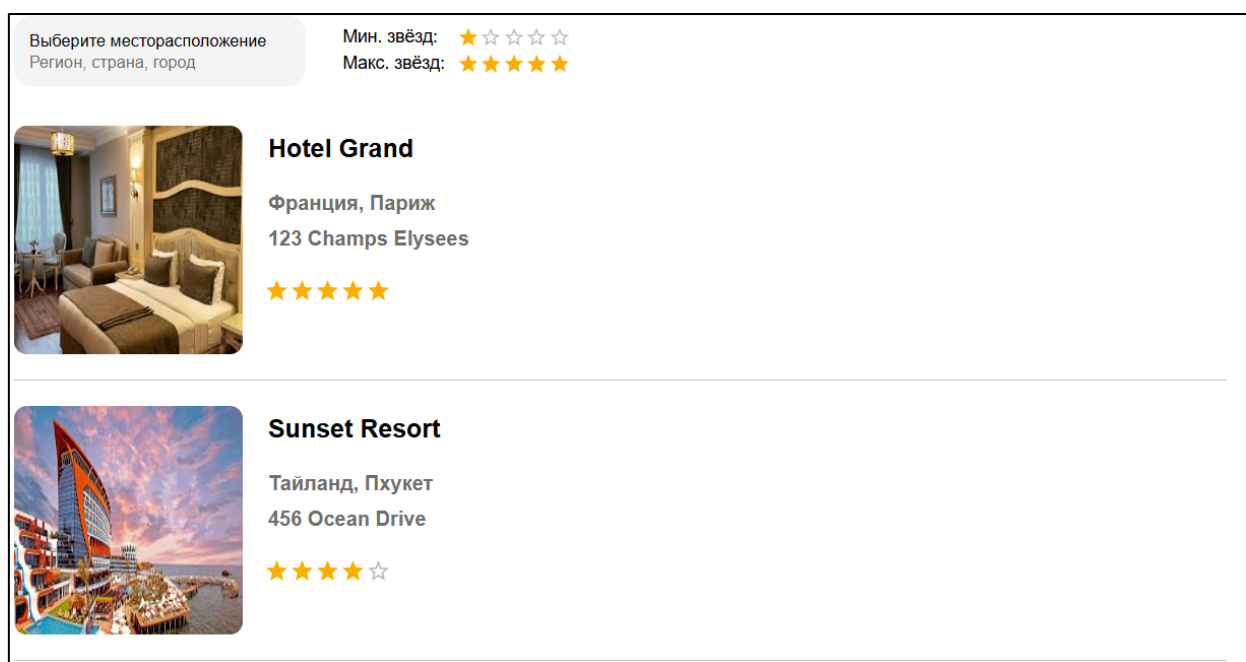


Рисунок 5.5 – Страница со списком отелей

Что бы увидеть информацию о конкретном отеле, необходимо нажать на отель в списке, либо же нажать на название отеля на странице тура. На данной странице можно увидеть фотографии отеля, просмотреть месторасположение его на карте и узнать его характеристики и характеристики его типов номеров. Страница отеля представлена на рисунке 5.6.



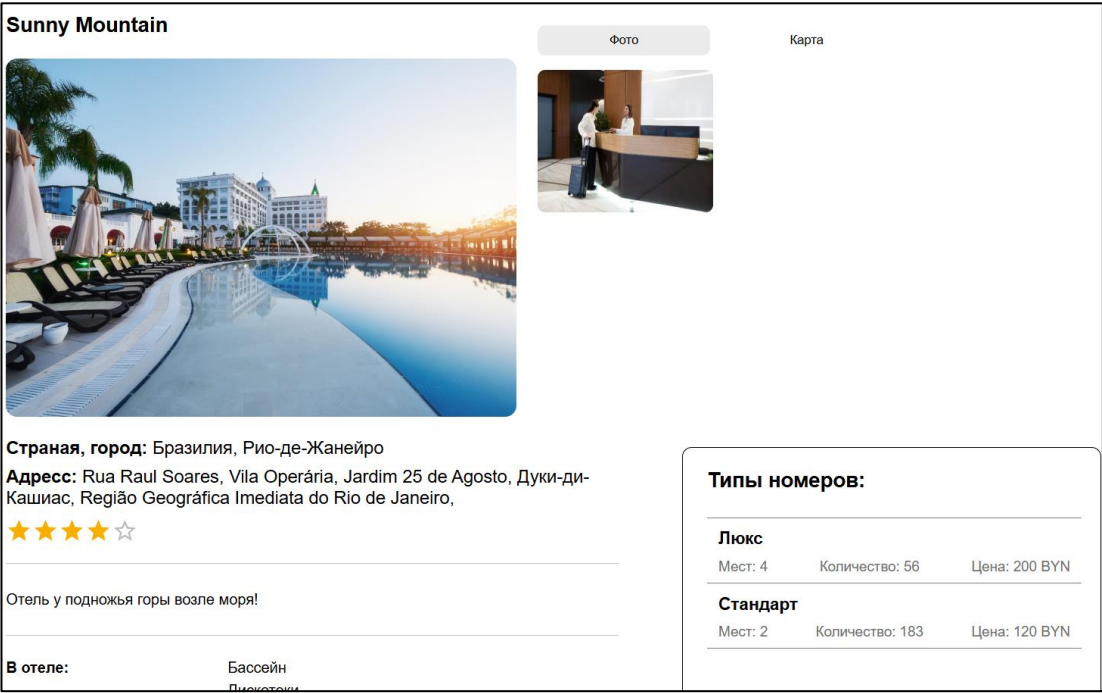


Рисунок 5.6 – Страница отеля

Внизу, под типами номеров, есть кнопка «Просмотреть туры с этим отелем». При нажатии на неё, вас перебросит на главную страницу, где будут отображаться только маршруты тех туров, которые есть с этим отелем.

5.5 Просмотр стран

Для того что бы увидеть список стран, необходимо в верхнем навигационном меню нажать на «Страны», затем выбрать регион. Вы перейдёте страницу со странами определённого региона. На ней можно отфильтровать список стран по месторасположению. Страница со странами представлена на рисунке 5.7.

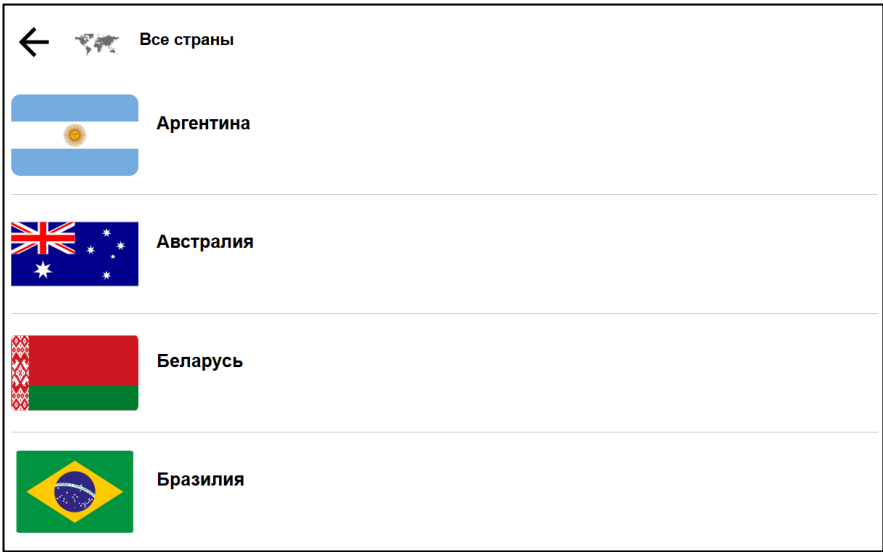


Рисунок 5.7 – Страница со странами

Что бы увидеть информацию о конкретной стране, необходимо нажать на страну в списке, либо же нажать на название страны на странице тура, или там же на её флаг. Страница страны представлена на рисунке 5.8.

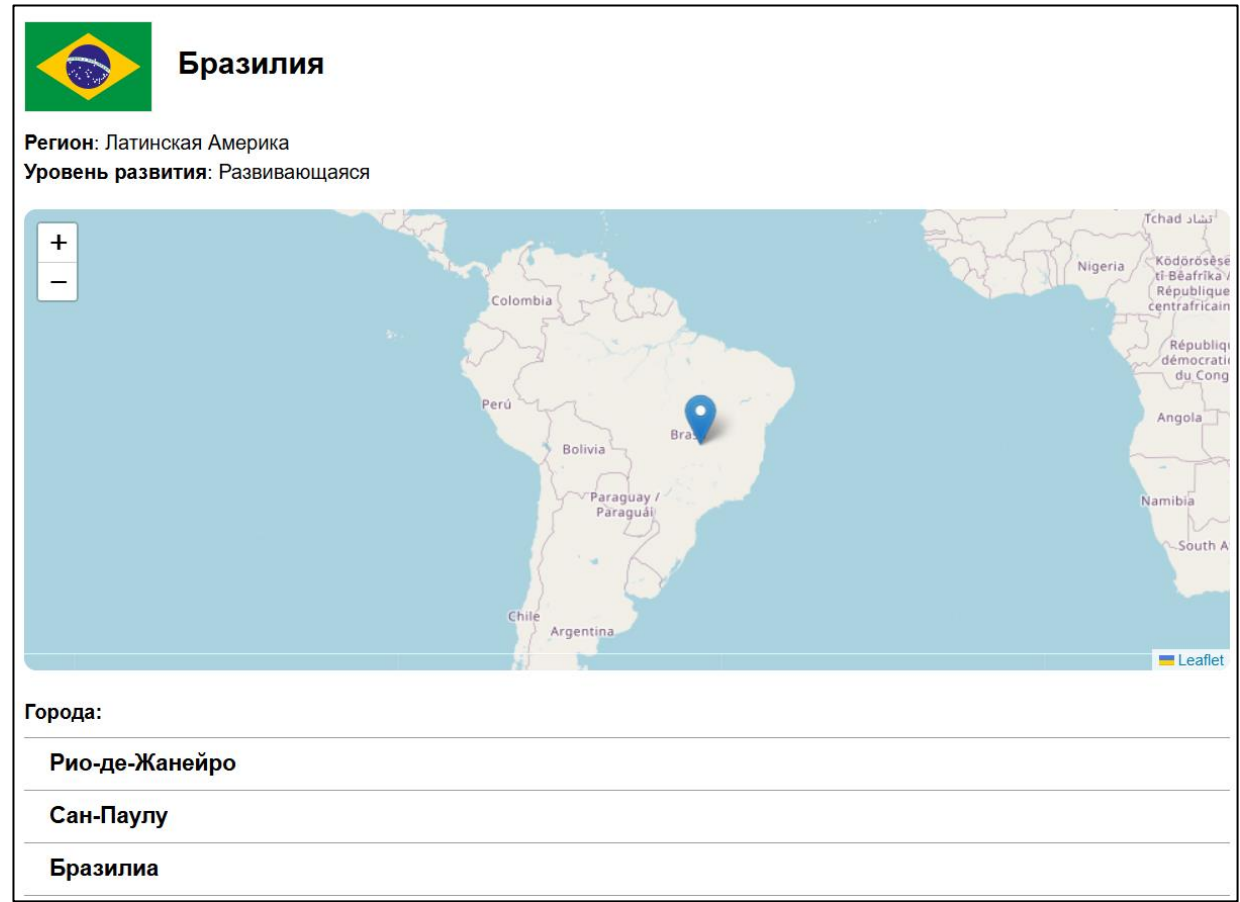


Рисунок 5.8 – Страница страны

На данной странице можно увидеть уровень развития страны, чтобы ориентироваться, какие там цены, и список городов этой страны. Нажав на город, вы перейдёте на страницу города, где вы можете узнать, какой климат в этом городе и какие там есть достопримечательности.

5.6 Фильтрация маршрутов

Для фильтрации маршрутов на главной странице расположены ряд инструментов. В верху страницы расположен главный фильтр. Он представлен на рисунке 5.9.

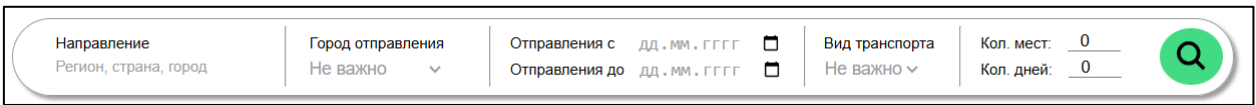


Рисунок 5.9 – Главный фильтр маршрутов

После выбора своих параметров фильтрации, необходимо нажать на кнопку с лупой.

Ниже главного фильтра расположилось навигационное меню с типами туров. Что бы сделать так, чтобы отображались маршруты туров с определённым типом, необходимо нажать на этот тип. Фильтр по типу туров представлен на рисунке 5.10.



Рисунок 5.10 – Фильтр маршрутов по типу туров

Справа от типов туров есть кнопка «Фильтр». Нажав на неё, откроется модальное окно с более подробными фильтрами. Оно представлено на рисунке 5.11.

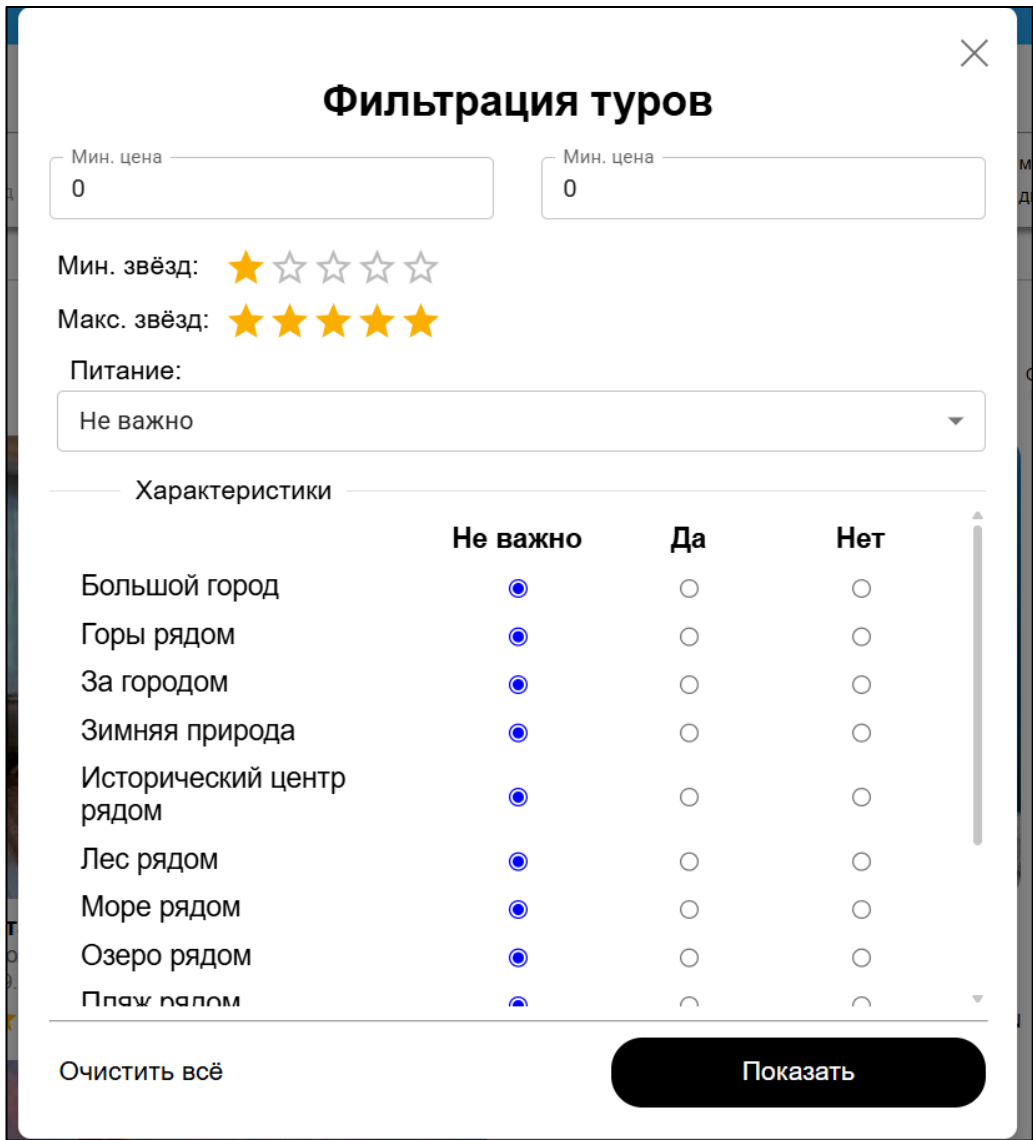


Рисунок 5.11 – Модальное окно с более подробными фильтрами

Здесь можно отфильтровать маршруты по цене и звёздам отелей, а также по географическим признакам.

## 5.7 Подбор маршрутов по опросу

Что бы подобрать маршруты с помощью опроса, необходимо в верхнем навигационном меню нажать на «Куда поехать?». Вы перейдёте на страницу с опросом. Далее вам необходимо отвечать на вопросы. Страница с опросом представлена на рисунке 5.12.

Рисунок 5.12 – Страница с опросом

После того, как вы ответите на все вопросы, вам предоставиться список маршрутов, которые подходят под ваши предпочтения. Результат подбора маршрутов по опросу представлен в ГЧ.

## 5.8 Подбор маршрутов с помощью ИИ

Для того, что перейти на страницу чата с ИИ, необходимо на странице опроса нажать «Спросить у ИИ». После чего у вас откроется страница, на которой вы можете в текстовое поле ввести описание городов или местности. Поле ввода запроса к ИИ представлено на рисунке 5.13.

Рисунок 5.13 – Поле ввода запроса к ИИ

После того, как вы отправите описание городов или местности, сервер получит соответствующие города из сервиса Gemini и вернёт маршруты с

турами в данные города. Вам отобразиться список с данными маршрутами. Данный список приведён в ГЧ. Принцип работы подбора маршрутов через ИИ представлен в ГЧ.

### 5.9 Оставление отзывов к турам

Что бы оставить отзыв, вам необходимо перейти на страницу тура, и пролистать вниз. В специальное поле можно вписать свой отзыв, и, нажав кнопку отправить, оставить отзыв. Оставленный отзыв представлен на рисунке 5.14.

Рисунок 5.14 – Оставленный отзыв

Там можно увидеть отзывы других пользователей. Как мы видим, наш отзыв отобразился.

### 5.10 Подача заявки на бронирование тура

Для того, чтобы забронировать тур, необходимо выбрать маршрут, зайти на страницу тура, в правом меню затем нажать кнопку «Заявка на тур». После появиться модальное окно с формой оформления брони. Модальное окно оформления брони тура представлено на рисунке 5.15.

Хар-ки	Тип номера	Кол. брон. номеров	Цена
▼	Люкс	<span style="color: green;">+</span> <span style="color: red;">-</span> 56	+2000
▼	Стандарт	<span style="color: green;">+</span> <span style="color: red;">-</span> 183	+1200

Всего бронируемых мест:  Осталось: 31

☐ Есть дети (менеджер уточнит о возможных скидках и вам сообщит)

☐ Места в приоритете в транспорте рядом/возле окна

Оставьте комментарий

\* Итоговая цена и некоторые детали тура могут поменяться. Менеджер в случае чего, обо всё вам сообщит

Цена: **480 BYN**

Рисунок 5.15 – Модальное окно оформления брони


В данной форме необходимо указать количество бронируемых номеров разных типов в отеле, количество людей, отправляющихся в тур, наличие детей, если есть и желания по рассадке в транспорте (по стандарту, приоритет рассадки будет друг возле друга). После заполнения формы, необходимо нажать «Отправить заявку». Далее нужно ждать подтверждения менеджером.

### 5.11 Оплата забронированного тура

Для того что бы оплатить забронированный тур, необходимо перейти на страницу брони и выбрать подтверждённую бронь. Пример подтверждённой брони представлен на рисунке 5.16.

×

## Бронь тура





### Тур по солнечной Бразилии

**Отправление с:** Национальный аэропорт «Минск»

**Транспорт:** Самолёт  
Беларусь, Минск, Минск, Минская область

**Отель:** Sunny Mountain  
Бразилия, Рио-де-Жанейро, Rua Raul Soares, Vila Operária, Jardim 25 de Agosto, Дуки-ди-Кашиас, Região Geográfica Imediata do Rio de Janeiro,

**Путь до отеля:** 12.05.2025, 08:40 → 12.05.2025, 21:39  
**Путь обратно:** 23.05.2025, 13:41 ← 24.05.2025, 05:32

Хар-ки	Тип номера	Кол. брон. номеров	Цена
▼	Люкс	1	2000
▼	Стандарт	2	1200

**Заказано мест:** 8  
**Есть дети!**  
**Приоритет рассадки:** рядом

Необходимо оплатить тур



Цена: **8240 BYN**

Отменить
Оплатить


Рисунок 5.16 – Подтверждённая бронь

Далее необходимо нажать «Оплатить». Вас перебросит на страницу для заполнения данных банковской карты. Здесь необходимо ввести номер карты, дату окончания срока действия карты и CVC. Форма оплаты представлена на рисунке 5.17.

Безопасное оформление покупок одним щелчком с помощью Link

Номер карты  
1234 1234 1234 1234  

Окончание срока действия  
MM / ГГ

Код безопасности  
CVC 

Страна  
Беларусь

**ОПЛАТИТЬ**


Рисунок 5.17 – Форма оплаты

После ввода данных, необходимо нажать «Оплатить». В случае успешной оплаты, забронированный тур получит статус оплачен.

## 5.12 Просмотр истории бронирований


Для просмотра истории бронирований, необходимо нажать на иконку профиля в верхнем правом углу страницы, затем выбрать либо «Мои брони», либо «История бронирований». При нажатии на «Мои брони», вас направит на страницу с текущими бронями, а если нажать на «История бронирований», то вас направит на страницу, где будут отображены предыдущие брони туров, в которых вы уже побывали. Страница с историей бронирований представлена на рисунке 5.18.

Направление  
Регион, страна, город
Всего поездок: 2
Всего потрачено: 22280 BYN
Ещё фильтры



**Старая Варшава**  
Беларусь, Минск, 19.06.2023, 08:55 → Польша, Варшава, 14.12.2025, 21:10  
Беларусь, Минск, 21.12.2025, 21:04 ← Польша, Варшава, 21.12.2025, 08:56  
Отправление с: Автовокзал «Центральный»  
Отель: Old Caste  
★★★★★  
Оплачено

**14040 BYN**



**Тур по солнечной Бразилии**  
Беларусь, Минск, 12.05.2025, 08:40 → Бразилия, Рио-де-Жанейро, 12.05.2025, 21:39  
Беларусь, Минск, 24.05.2025, 05:32 ← Бразилия, Рио-де-Жанейро, 23.05.2025, 13:41  
Отправление с: Национальный аэропорт «Минск»  
Отель: Sunny Mountain  
★★★★★  
Оплачено

**8240 BYN**

Рисунок 5.18 – Страница с историей бронирований

На данной странице отображаются брони туров, дата которых меньше текущей и которые были оплачены. Если же тур бы забронирован, подтверждён и пользователь его не оплатил, то данный тур не будет нигде отображаться. Однако администратор будет видеть количество таких броней у каждого пользователя, и в случае чего, будет блокировать или даже удалять таких пользователей.



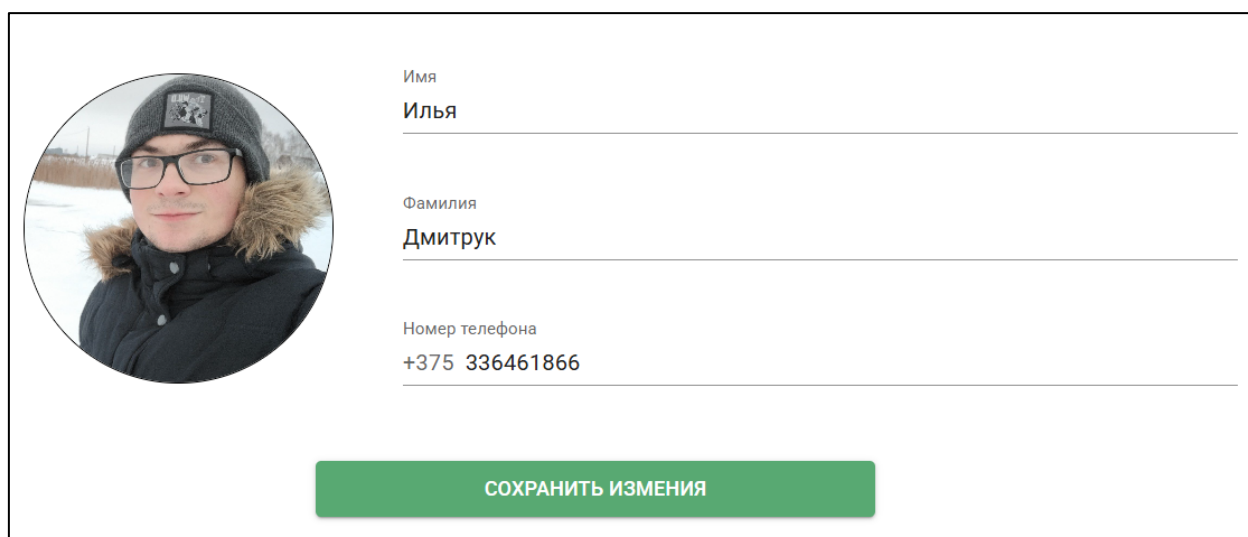
### 5.13 Отмена заявки на бронирование тура

Что бы отменить бронь тура, необходимо либо нажать на крестик в карточке брони на странице бронирований, либо выбрать бронь и в модальном окне нажать «Отменить».

Пользователи могут отменять свои заявки на бронирования туров, а менеджеры могут отменять все заявки.

### 5.14 Редактирование профиля

Для того, чтобы поменять данные профиля, необходимо в верхнем правом углу нажать на иконку профиля и выбрать пункт «Редактировать профиль». У вас откроется страница редактирования профиля. Данная страница представлена на рисунке 5.19.



Имя  
Илья

Фамилия  
Дмитрук

Номер телефона  
+375 336461866

СОХРАНИТЬ ИЗМЕНЕНИЯ

Рисунок 5.19 – Страница редактирования профиля

Здесь можно изменить аватарку, имя, фамилию и номер телефона пользователя. Чтобы подтвердить изменения, необходимо нажать на кнопку «Сохранить изменения».

### 5.15 Добавление туров

Для того, чтобы добавить тур, вам необходимо нажать на иконку профиля в правом верхнем углу и выбрать пункт «Редактор туров». Вас перебросит на страницу со списком туров. Сверху страницы будет кнопка «Добавить тур». Нажав на неё, вы перейдёте в редактор тура. Данная страница будет с незаполненными данными. Данная страница добавления тура представлена на рисунке 5.20.



Рисунок 5.20 – Страница добавления тура

Вам необходимо ввести название тура, загрузить его изображения, выбрать отель, ввести описание тура, выбрать тип питания, выбрать тип тура и выделить те характеристики, что присущи этому туру. В правой части страницы находится меню маршрутов. Что бы добавить новый маршрут, необходимо нажать на кнопку с плюсиком в данном меню, затем ввести все данные маршрута. Модальное окно ввода данных маршрута представлено на рисунке 5.21.

Рисунок 5.20 – Модальное окно ввод данных маршрута

Затем нажать кнопку сохранить. В списке маршрутов появится новый маршрут. Его можно удалить, наведясь на него и нажав на крестик в правом верху маршрута, и редактировать, просто нажав на него. Маршрутов можно добавить несколько.

Ну и чтобы добавить сам тур, необходимо нажать кнопку «Сохранить тур»

## 5.16 Редактирование туров

Процесс редактирования туров в целом схож с добавлением, только у вас при открытии редактора туров, будут подгружены данные тура. Что бы открыть страницу для редактирования тура, необходимо нажать на тур на странице «Редактор туров».

## 5.17 Удаление туров

Что бы удалить тур, необходимо навестись на него на странице «Редактор туров» и нажать на крестик в правом верхнем углу.

## 5.18 Добавление отелей

Для того, чтобы добавить отель, вам необходимо нажать на иконку профиля в правом верхнем углу и выбрать пункт «Редактор отелей». Вас перебросит на страницу со списком отелей. Сверху страницы будет кнопка «Добавить отель». Нажав на неё, вы перейдёте в редактор отеля. Данная страница будет с незаполненными данными. Страница добавления отеля представлена на рисунке 5.21.

Название отеля

Название Загрузить изображения

Страна, город: Выбрать страну, город

Адрес:

Количество звёзд: ★ ☆ ☆ ☆ ☆

Общее описание

Тип питания

Бар ☐

Типы номеров: +

Рисунок 5.21 – Страница добавления отеля

Вам необходимо ввести название отеля, загрузить его изображения, выбрать месторасположение, ввести адрес, или выбрать его на карте, указать количество звёзд, ввести описание отеля, выбрать тип питания и выделить те характеристики, что присущи этому отелю. В правой части страницы находится меню типов номеров. Что бы добавить новый тип номера, необходимо нажать на кнопку с плюсиком в данном меню, затем ввести все данные типа номера. Модальное окно ввода данных типа номера представлено на рисунке 5.22.

Рисунок 5.22 – Модальное окно ввода данных типа номера

Затем нажать кнопку сохранить. В списке типов номеров появится новый тип номера. Его можно удалить, наведясь на него и нажав на крестик в правом верху типа номера, и редактировать, просто нажав на него. Типов номеров можно добавить несколько.

Ну и чтобы добавить сам отель, необходимо нажать кнопку «Сохранить отель»

### 5.19 Редактирование отелей

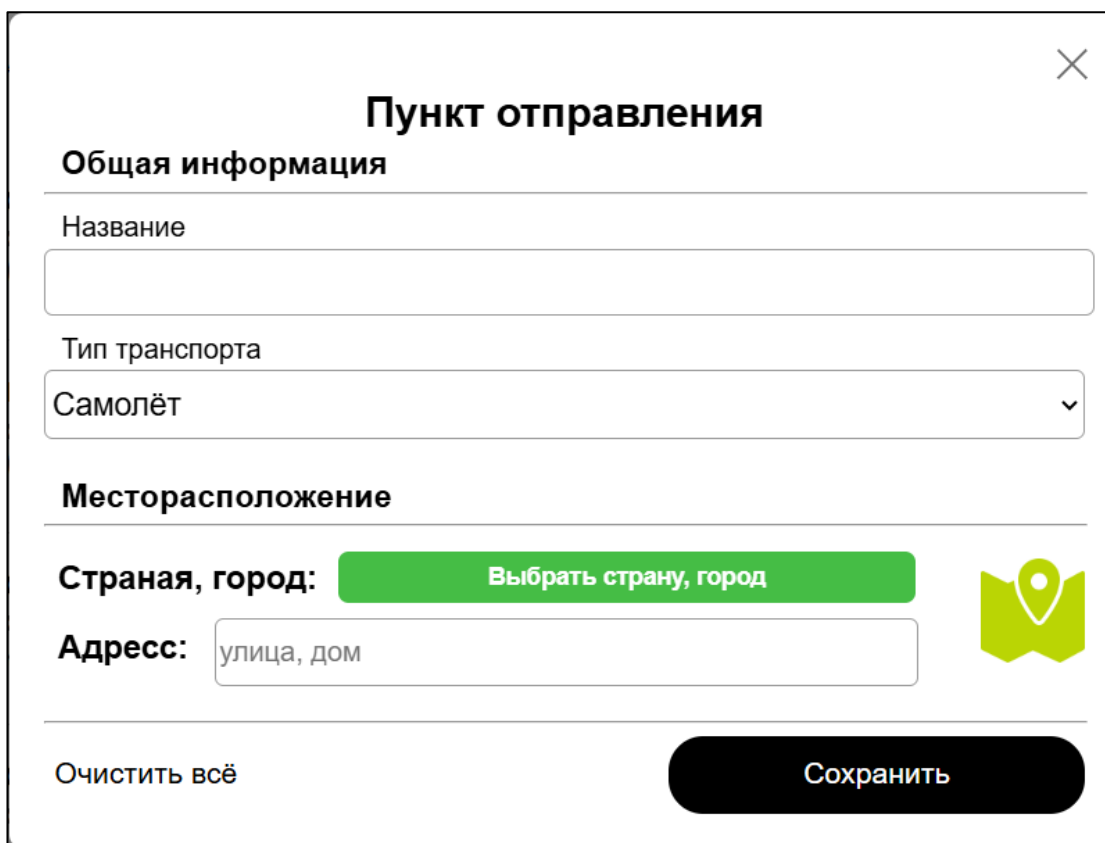
Процесс редактирования отелей в целом схож с добавлением, только у вас при открытии редактора отелей, будут подгружены данные отеля. Что бы открыть страницу для редактирования отеля, необходимо нажать на отель на странице «Редактор отелей».

### 5.20 Удаление отелей

Что бы удалить отель, необходимо навестись на него на странице «Редактор отелей» и нажать на крестик в правом верхнем углу.

### 5.21 Добавление пунктов отправлений

Для того, чтобы добавить пункт отправления, вам необходимо нажать на иконку профиля в правом верхнем углу и выбрать пункт «Редактор транспорта». Вас перебросит на страницу со списком пунктов отправлений. Сверху страницы будет кнопка «Добавить пункт отправления». Нажав на неё, у вас откроется модальное окно с редактором пункта отправления. Данное модальное окно будет с незаполненными данными. Модальное окно добавления пункта отправлений представлена на рисунке 5.23.



**Пункт отправления**

**Общая информация**

Название

Тип транспорта

Самолёт

**Месторасположение**

Страная, город: **Выбрать страну, город**

Адресс:

Очистить всё **Сохранить**

Рисунок 5.23 – Модальное окно добавления пункта отправления

Вам необходимо ввести название пункта отправления, выбрать тип транспорта, выбрать месторасположение и указать адрес. Ну и чтобы добавить пункт отправления, необходимо нажать кнопку «Сохранить».

### 5.22 Редактирование пунктов отправлений

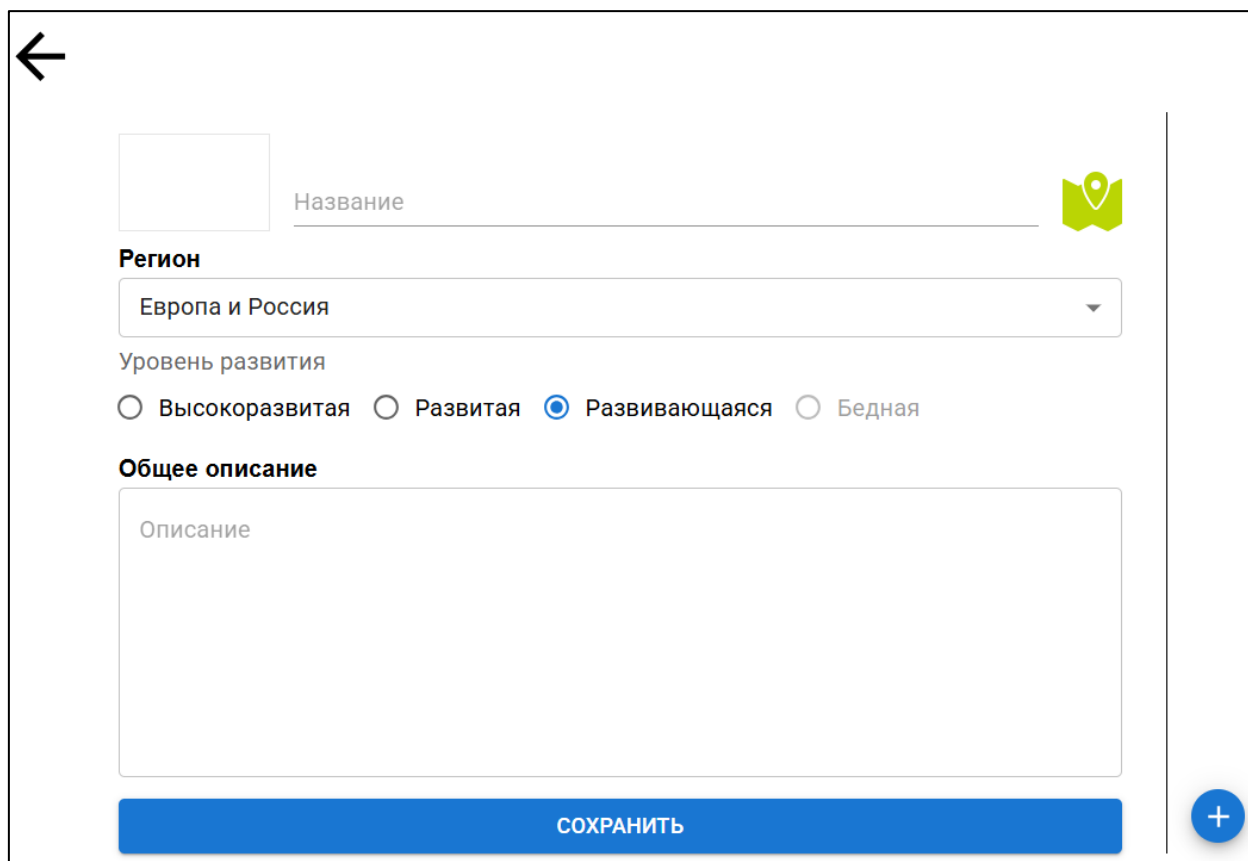
Процесс редактирования пунктов отправлений в целом схож с добавлением, только у вас при открытии редактора пункта отправления, будут подгружены данные пункта отправления. Что бы открыть модальное окно для редактирования пункта отправления, необходимо нажать на пункт отправления на странице «Редактор транспорта».

### 5.23 Удаление пунктов отправлений

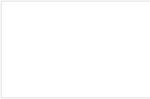
Что бы удалить пункт отправления, необходимо навестись на него на странице «Редактор транспорта» и нажать на крестик в правом верхнем углу.

### 5.24 Добавление стран

Для того, чтобы добавить страну, вам необходимо нажать на иконку профиля в правом верхнем углу и выбрать пункт «Редактор гео-объектов». Вас перебросит на страницу со списком регионов. Сверху страницы будет кнопка «Добавить страну». Нажав на неё, у вас откроется редактор стран с незаполненными данными. Страница добавления страны представлена на рисунке 5.24.



←



**Регион**

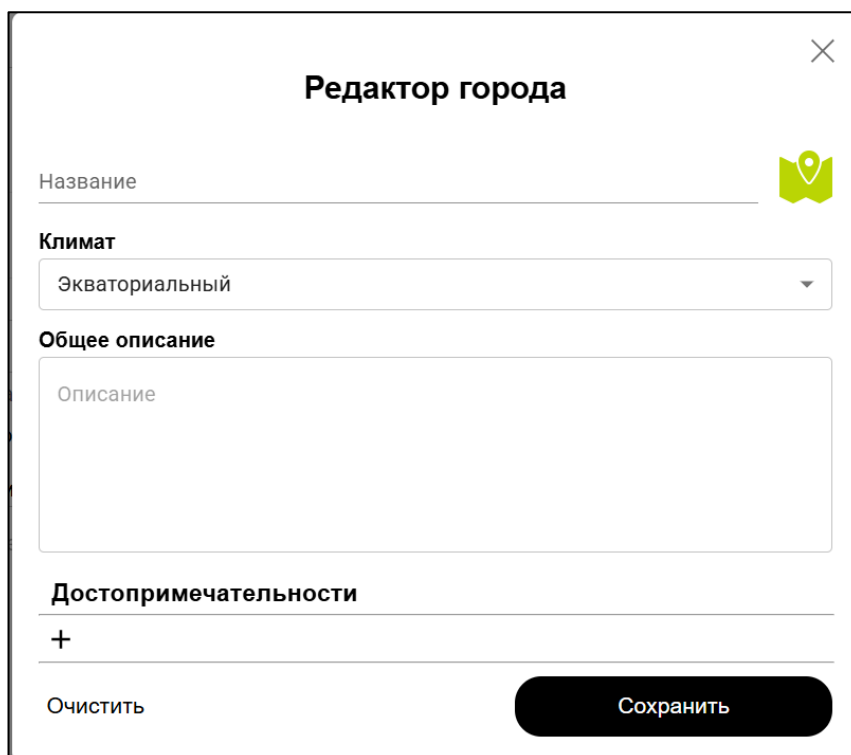
**Уровень развития**

☐ Высокоразвитая ☐ Развитая ☒ Развивающаяся ☐ Бедная

**Общее описание**

Рисунок 5.24 – Страница добавления страны

Вам необходимо ввести название страны, загрузить изображение флага, выбрать регион, где располагается страна, выбрать уровень развития (для подбора по цене по результатам опроса), ввести описание. В правой части страницы находится список городов. Что бы добавить новый город, необходимо нажать на кнопку с плюсиком внизу списка, затем ввести все данные города. Модальное окно добавления города представлено на рисунке 5.25.



**Редактор города**

Название

Климат

Экваториальный

Общее описание

Описание

Достопримечательности

+

Очистить Сохранить

Рисунок 5.25 – Модальное окно добавления города

Затем нажать кнопку сохранить. В списке городов появится новый город. Его можно удалить, наведясь на него и нажав на крестик в правом углу, и редактировать, просто нажав на него.

Ну и чтобы добавить саму страну, необходимо нажать кнопку «Сохранить».

## 5.25 Редактирование стран

Процесс редактирования стран в целом схож с добавлением, только у вас при открытии редактора страны, будут подгружены данные страны. Что бы открыть страницу для редактирования страны, необходимо выбрать регион и нажать на страну на странице «Редактор гео-объектов».

## 5.26 Удаление стран

Что бы удалить страну, необходимо навестись на неё на странице «Редактор гео-объектов» и нажать на крестик в правом верхнем углу.

## 5.27 Смена цены забронированного тура

Для того что бы изменить итоговую цену у уже забронированного тура, необходимо выбрать нужную бронь и снизу возле цены нажать на значок карандаша. У вас откроется модальное окно, где вы можете ввести новую цену. Модальное окно для изменения цены представлено на рисунке 5.26.

Рисунок 5.26 – Модальное окно изменения цены у брони

Это может быть необходимо, когда выясняется, что у данной брони есть скидки или другие причины для изменения цены.

## 5.28 Удаление отзывов к турам

Для того что бы удалить отзыв к туру, необходимо сначала открыть тур, затем найти необходимый отзыв, навестись на него и нажать на крестик справа.

## 5.29 Смена роли пользователей

Что бы изменить роли пользователю, вам необходимо перейти на страницу с пользователями. Для этого нажмите на значок профиля в правом верхнем углу и в всплывающем меню нажмите на «Пользователи». Затем у нужного пользователя нажмите «Сменить роль». У вас раскроются кнопки с ролями, выберите нужную. Смена роли пользователя представлена на рисунке 5.27.

Рисунок 5.27 – Смена роли пользователю

При смене роли, у пользователя поменяются его права. Таким образом, можно делать из обычных пользователей менеджеров и администраторов.

### **5.30 Разблокировка пользователей**

Для того что разблокировать пользователя, необходимо нажать на соответствующую кнопку у нужного пользователя. После разблокировки, пользователь вновь сможет выполнять свои функции.

### **5.31 Удаление пользователей**

Что бы удалить пользователя, необходимо навестись на него на странице «Пользователи» и нажать на крестик в правом верхнем углу.

### **5.32 Блокировка пользователей**

Для того что заблокировать пользователя, необходимо нажать на соответствующую кнопку у нужного пользователя. После блокировки, пользователь не сможет выполнять свои функции. Он сможет только просматривать туры и соответствующую им информацию, и редактировать профиль.

### **5.33 Вывод по разделу**

Раздел охватывает описание основных функций веб-приложения, все действия описаны структурировано, что облегчает пользование, даже неподготовленным пользователям.

В разделе уделено внимание в том числе обработке ошибок, возникающих при использовании системы, в основном это ошибки, касающиеся валидации полей ввода в формах.

Описана каждая вкладка, используемая в приложении, что позволит понять по отдельности каждую часть приложения.

Включены инструкции по началу работы с веб-приложением, что помогает новым пользователям быстро освоиться с системой.

Приведены примеры использования веб-приложения в различных сценариях, чтобы показать, как разные функции могут быть применены в реальных ситуациях.



## 6 Технико-экономическое обоснования проекта

### 6.1 Общая характеристика разрабатываемого программного средства

При выполнении данного проекта было разработано веб-приложение YUTUBE, предназначенное для размещения и просмотра видеоконтента. Целью приложения было создание удобной и открытой платформы, где пользователи могут взаимодействовать с видео, каналами и другими участниками, в зависимости от своих ролей.

Пользователи могут сортировать видео, искать их по названию или тегам, а также просматривать публичные ролики и комментарии под ними. Пользователям предлагается гибкая система ролей, каждая из которых открывает дополнительные возможности. Гости могут зарегистрироваться или авторизоваться, чтобы получить доступ к расширенному функционалу. Клиенты управляют своими каналами, загружают и редактируют видео, создают плейлисты, оставляют комментарии и жалобы, а также используют функцию совместного просмотра. Администраторы следят за порядком на платформе, скрывая или удаляя нежелательный контент, блокируя каналы и обрабатывая жалобы.

Во время разработки дипломного проекта использовались технологии ASP.NET Core для backend-части, SignalR для реализации совместного просмотра, EF Core для работы с базой данных, React и MobX для создания динамичного интерфейса, а также PostgreSQL в качестве надежной СУБД.

Разработанное программное решение имеет следующие преимущества перед рассмотренными в главе 1 аналогичными образцами:

- простота использования приложения;
- открытость платформы для всех категорий пользователей;
- совместный просмотр видео в реальном времени.

Стратегия монетизации не предполагается, так как приложение разрабатывается в социальных целях и предоставляется пользователям полностью бесплатно. Предполагается продвижение приложения через социальные сети, тематические сообщества и форумы, ориентированные на просмотр и создание видеоконтента.

### 6.2 Исходные данные для проведения расчётов и маркетинговый анализ

Источниками исходных данных для данных расчетов выступают действующие нормативные правовые акты. Исходные данные для расчета приведены в таблице 6.1.

					ДП 06.00 ПЗ									
		Ф.И.О	Подпись	Дата										
Разраб	Окулич Д.Ю.				6 Технико-экономическое обоснование проекта				Лит.		Лист		Листов	
Пров.	Белодед Н.И.								У		1		8	
Консульт.	Познякова Л.С.								БГТУ 1-40 01 01, 2025					
Н. контр.	Белодед Н.И.													
Утв.	Смелов В.В.													

Таблица 6.1 – Исходные данные для расчета

Наименование показателя	Условные обозначения	Норматив
Норматив дополнительной заработной платы, %	$H_{дз}$	9
Ставка отчислений в Фонд социальной защиты населения, %	$H_{фсзн}$	34
Ставка отчислений по обязательному страхованию в БРУСП «Белгосстрах», %	$H_{бгс}$	0,6
Норматив прочих прямых затрат, %	$H_{пз}$	25
Норматив накладных расходов, %	$H_{обп,обх}$	50
Норматив расходов на сопровождение и адаптацию, %	$H_{рса}$	10
Ставка НДС, %	$H_{ндс}$	20
Налог на прибыль, %	$H_{п}$	20

В ходе маркетингового анализа была определена стоимость разработки аналогичных веб-приложений, ориентированных на видео контент. Результаты анализа представлены в таблице 6.2.

Таблица 6.2 – Анализ стоимости разработки

Продукты-аналоги	Источник	Стоимость, руб	Примечание
YouTube	<a href="https://youtube.com">https://youtube.com</a>	40 000	Информация за 2006 год.
Vk Video	<a href="https://vkvideo.ru">https://vkvideo.ru</a>	22 000	Примерная стоимость на основе стартового функционала и инфраструктуры что уже была.
RUTUBE	<a href="https://rutube.ru">https://rutube.ru</a>	27 000	Примерная стоимость на основе стартового функционала.

В ходе проведения маркетингового анализа, была определена стоимость разработки аналогичного программного продукта видео-хостинга. Средняя цена разработки аналогичного продукта составляет 25000-33000 рублей. Так же стоит учитывать модуль совместного просмотра, который отдельно стоит 2000 руб. Таким образом, общая стоимость разработки данного программного средства, выбранного в качестве базы сравнения ставит 26000 рублей без НДС. С НДС цена составляет 31000 рублей.

### 6.3 Обоснование цены программного средства

#### 6.3.1 Расчёт затрат рабочего времени на разработку программного средства

В таблице 6.3 в укрупнённом виде указаны работы, которые необходимо выполнить для создания указанного в дипломной работе программного средства, исполнители по данным работам и трудозатраты по каждой работе.

Таблица 6.3 – Затраты рабочего времени на разработку ПС

Содержание работ	Исполнитель	Трудозатраты, чел-часов
Руководство проектом	Проектный менеджер	68
Проектирование архитектуры приложения и структуры базы данных	Бизнес-аналитик	56
Разработка серверной части приложения	Бэкенд-разработчик	160
Вёрстка интерфейса на основе макетов	Фронтенд-разработчик	112
Функциональное тестирование	Тестировщик	96
Сопровождение приложения	Системный администратор	48
Проектирование UI/UX и создание макетов	Дизайнер	64
Всего		604

Таким образом суммарно на разработку будет затрачено 604 часа.

### 6.3.2 Расчет основной заработной платы

Для определения величины основной заработной платы, было проведено исследование величин заработных плат для специалистов в сфере разработки и определение их часовых ставок. Источником данных служили открытые веб-порталы, различные форумы, официальная отчетность, а также общий средний уровень заработка в сфере информационных технологий в Республике Беларусь.

После определения часовых ставок и трудозатрат исполнителей определяются заработные платы всех исполнителей. Заработная плата отдельного специалиста рассчитывается по формуле 6.1. Результаты подсчетов представлены в таблице 6.4.

$$C_{\text{оз}} = T_{\text{раз}} \cdot C_{\text{зп}} \quad (6.1)$$

где  $C_{\text{оз}}$  – основная заработная плата, руб.;

$T_{\text{раз}}$  – трудоемкость, чел./час.;

$C_{\text{зп}}$  – средняя часовая ставка, руб./час.

Таблица 6.4 – Расчет основной заработной платы специалистов

Исполнитель	Затраты рабочего времени, часов	Средняя часовая ставка, руб./час	Основная заработная плата, руб.
Проектный менеджер	68	20	1360
Бизнес-аналитик	56	18	1008
Бэкенд-разработчик	160	16	2560
Фронтенд-разработчик	112	15	1680
Тестировщик	96	13	1248
Системный администратор	48	15	720
Дизайнер	64	12	768

Всего	604		9344
-------	-----	--	------

Суммарная основная заработная плата всех специалистов веб-приложения составит 9344 рублей.

### 6.3.3 Расчет дополнительной заработной платы

Дополнительная заработная плата на конкретное программное средство включает выплаты, предусмотренные законодательством о труде, и определяется по нормативу в процентах к основной заработной плате по формуле 6.2.

$$C_{\text{дз}} = \frac{C_{\text{оз}} \cdot H_{\text{дз}}}{100}, \quad (6.2)$$

где  $C_{\text{оз}}$  – основная заработная плата, руб.;

$H_{\text{дз}}$  – норматив дополнительной заработной платы, %.

$$C_{\text{дз}} = 9344 \cdot 9 / 100 = 840,96 \text{ руб.}$$

Таким образом дополнительная заработная плата составила 840,96 рублей.

### 6.3.4 Расчет отчислений в Фонд социальной защиты населения и по обязательному страхованию

Отчисления в Фонд социальной защиты населения (ФСЗН) и по обязательному страхованию от несчастных случаев на производстве, и профессиональных заболеваний в БРУСП «Белгосстрах» определяются в соответствии с действующими законодательными актами по нормативу в процентном отношении к фонду основной и дополнительной зарплаты исполнителей и вычисляются по формуле 6.3.

$$C_{\text{фсзн}} = \frac{(C_{\text{оз}} + C_{\text{дз}}) \cdot H_{\text{фсзн}}}{100}, \quad (6.3)$$

где  $C_{\text{оз}}$  – основная заработная плата, руб.;

$C_{\text{дз}}$  – дополнительная заработная плата на конкретное ПС, руб.;

$H_{\text{фсзн}}$  – норматив отчислений в Фонд социальной защиты, %.

Отчисления в БРУСП «Белгосстрах» вычисляются по формуле 6.4.

$$C_{\text{бгс}} = \frac{(C_{\text{оз}} + C_{\text{дз}}) \cdot H_{\text{бгс}}}{100}, \quad (6.4)$$

$$C_{\text{фсзн}} = \frac{(9344 + 840,96) \cdot 34}{100} = 3\,462,89 \text{ руб.}$$

$$C_{\text{бгс}} = \frac{(9344 + 840,96) \cdot 0,6}{100} = 61,11 \text{ руб.}$$

Таким образом, общие отчисления в БРУСП «Белгосстрах» составили 61,11 руб., а в фонд социальной защиты населения – 3 462,89 руб.

### 6.3.5 Расчет суммы прочих прямых затрат

Расходы на конкретное программное средство  $C_{\text{пз}}$  включают расходы на приобретение и подготовку специальной технической информации, платных сервисов тестирования и прочие операционные издержки, прямо относимые на проект, и рассчитываются по формуле 6.5.

$$C_{\text{пз}} = \frac{C_{\text{оз}} \cdot H_{\text{пз}}}{100}, \quad (6.5)$$

где  $H_{\text{пз}}$  – норматив прочих затрат в целом по организации, %.

$$C_{\text{пз}} = 9344 \cdot 25 / 100 = 2\,336 \text{ руб.}$$

Таким образом, сумма прочих прямых затрат при разработке веб-приложения составила 2 336 рублей.

### 6.3.6 Расчет суммы накладных расходов

Сумма накладных расходов  $C_{\text{обп,обх}}$  – произведение основной заработной платы исполнителей на конкретное программное средство  $C_{\text{оз}}$  на норматив накладных расходов в целом по организации  $H_{\text{обп,обх}}$ , по формуле 6.6.

$$C_{\text{обп,обх}} = \frac{C_{\text{оз}} \cdot H_{\text{обп,обх}}}{100} \quad (6.6)$$

Сумма накладных расходов составит:

$$C_{\text{обп,обх}} = 9344 \cdot 50 / 100 = 4\,672 \text{ руб.}$$

Таким образом, сумма накладных расходов составила 4 672 руб.

### 6.3.7 Сумма расходов на разработку программного средства

Сумма расходов на разработку программного средства  $C_p$  определяется как сумма основной и дополнительной заработных плат исполнителей на конкретное программное средство, отчислений на социальные нужды, суммы прочих прямых затрат и суммы накладных расходов, по формуле 6.7.

$$C_p = C_{оз} + C_{дз} + C_{фсзн} + C_{бгс} + C_{пз} + C_{обп,обх} \quad (6.7)$$

Все данные необходимые для вычисления есть, поэтому можно определить сумму расходов на разработку программного средства.

$$C_p = 9\,344 + 840,96 + 3\,462,89 + 61,11 + 2\,336 + 4\,672 = 20\,716,96 \text{ руб.}$$

Сумма расходов на разработку программного средства была вычислена на основе данных, рассчитанных ранее в данном разделе, и составила 20 716,96 рублей.

### 6.3.8 Расходы на сопровождение и адаптацию

Сумма расходов на сопровождение и адаптацию программного средства  $C_{рса}$  определяется как произведение суммы расходов на разработку на норматив расходов на сопровождение и адаптацию  $H_{рса}$ , и находится по формуле 6.8.

$$C_{рса} = \frac{C_p \cdot H_{рса}}{100} \quad (6.8)$$

$$C_{рса} = 20\,716,96 \cdot 10 / 100 = 2\,071,7 \text{ руб.}$$

Получим, что сумма расходов на сопровождение и адаптацию программного средства, определенная по формуле 6.8, составляет 2 071,7 рубля.

### 6.3.9 Расчет полной себестоимости

Полная себестоимость  $C_{п}$  определяется как сумма двух элементов: суммы расходов на разработку  $C_p$  и суммы расходов на сопровождение и адаптацию  $C_{рса}$ .

Полная себестоимость  $C_{п}$  вычисляется по формуле 6.9

$$C_{п} = C_p + C_{рса} \quad (6.9)$$

$$C_{п} = 20\,716,96 + 2\,071,7 = 22\,778,65 \text{ руб.}$$

Получим, что полная себестоимость мобильного приложения равна 22 778,65 рубля.

#### **6.4 Вывод по разделу**

В рамках данного раздела были проведены экономические расчеты, на основе которых была определена себестоимость разрабатываемого программного средства, а также прогнозируемая отпускная цена всего продукта. Анализ такого вида позволяет определить целесообразность разработки приложения.

В таблице 6.5 представлены результаты расчетов для основных показателей данной главы в краткой форме.

Таблица 6.5 – Результаты расчетов

Наименование показателя	Значение
Время разработки, ч.	604
Основная заработная плата, руб.	9344
Дополнительная заработная плата, руб.	840,96
Отчисления в Фонд социальной защиты населения, руб.	3462,89
Отчисления в БРУСП «Белгосстрах», руб.	61,11
Прочие прямые затраты, руб.	2336
Накладные расходы, руб.	4976,13
Себестоимость разработки программного средства, руб.	4672
Расходы на сопровождение и адаптацию, руб.	2071,7
Полная себестоимость, руб.	22 778,65

Современный видеохостинг сталкивается с высокой конкуренцией и растущими требованиями пользователей к качеству контента, скорости загрузки и персонализации рекомендаций. Существующие платформы часто не учитывают региональные особенности, испытывают проблемы с модерацией и предлагают шаблонные алгоритмы, которые не всегда соответствуют интересам аудитории. BYTUBE призван решить эти проблемы, предлагая гибкую, адаптивную платформу с акцентом на локальный контент и сообщество.

Необходимость разработки программного средства, обусловлена большим количеством рекламы у конкурентов которая пагубно влияет на развитие сообщества, а также отсутствие некоторые функций.

BYTUBE создаёт уникальный социальный эффект, устраняя раздражающую рекламу и делая просмотр видео более комфортным. Пользователи могут полностью сосредоточиться на контенте, не отвлекаясь на навязчивые ролики или баннеры. Это формирует лояльное сообщество, где ценность контента стоит на первом месте.

Для поддержания работоспособности программного продукта, потребности, затраты на содержание серверов, администрирование, возмещение которых предполагается за счет добровольных пожертвований пользователей.

Совместный просмотр в реальном времени превращает BYTUBE в платформу для живого общения. Друзья, родственники или даже незнакомцы могут смотреть видео одновременно, комментировать и обсуждать происходящее в чате. Это создаёт эффект кинотеатра или телевизионного вечера, но в цифровом пространстве.

Доступность распространения контента на BYTUBE стимулирует творчество среди обычных пользователей. Нет жёстких алгоритмических ограничений, как в крупных коммерческих платформах, поэтому даже начинающие авторы могут найти свою аудиторию. Это делает платформу более демократичной и открытой для экспериментов.



Отсутствие монетизации через рекламу снижает конкуренцию за клики, что уменьшает количество "мусорного" контента. Это формирует здоровую экосистему, где творчество важнее заработка на просмотрах.

## Заключение

В результате выполнения дипломного проектирования было разработано веб-приложение для турагентства с возможностью просмотра и подбора туров по географическим признакам, и бронирования их.

В рамках работы над проектом был проведен обзор аналогичных решений, выбрана платформа для разработки серверной и клиентской частей программного продукта, спроектирована архитектура приложения.

База данных приложения состоит из 24 таблиц, а объем пользовательского кода превышает 12 000 строк.

Для обеспечения качества разработанного продукта было создано 31 тест-кейсов, которые покрыли 90% функционала веб-приложения. Пользователи имеют возможность регистрироваться и авторизовываться в системе, а также использовать широкий спектр функций для подбора и бронирования туров, предоставляемых приложением. Менеджеры имеют возможность предоставлять пользователям туры и управлять заявками на бронирования. Администраторы, в свою очередь, получают доступ ко всем возможностям менеджеров, а также управление пользователями с целью обеспечения порядка.

Для эффективной работы веб-приложения и улучшения качества предоставляемых туристических услуг, использовались следующие внешние сервисы:

- Stripe – сервис платёжной системы, для реализации оплаты;
- Gemini – сервис ИИ, для улучшения качества подбора туров;
- OpenStreetMap – сервис интерактивной карты, для отображения месторасположения объектов на карте.

Веб-приложение является завершенным продуктом с возможностью дальнейшего расширения, что позволяет масштабировать его функционал. Интерфейс приложения разработан с учетом удобства пользователей, что делает его доступным для людей любого возраста. Программное средство полностью соответствует поставленным задачам и требованиям.

					ДП 00.00.ПЗ		
		ФИО	Подпись	Дата	Заключение		
Разраб.		Дмитрук И.И.					
Пров.		Белодед Н.И.					
Н. контр.		Белодед Н.И.					
Утв.		Смелов В.В.			БГТУ 1-40 01 01, 2025		
		Лит.	Лист	Листов			
		У	1	1			

## Список используемых источников

- 1 ASP.NET Core 8.0 [Электронный ресурс]. – Режим доступа: <https://learn.microsoft.com/ru-ru/aspnet/core/release-notes/aspnetcore-8.0> – Дата доступа: 28.05.2025
- 2 React [Электронный ресурс]. – Режим доступа: <https://react.dev/blog/2024/12/05/react-19> – Дата доступа: 28.05.2025
- 3 MOBX [Электронный ресурс]. – Режим доступа: <https://mobx.js.org/README.html> – Дата доступа: 28.05.2025
- 4 PostgreSQL [Электронный ресурс]. – Режим доступа: <https://www.postgresql.org/docs/> – Дата доступа: 28.05.2025
- 5 REST API [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/articles/483202/> – Дата доступа: 28.05.2025
- 6 SignalR [Электронный ресурс]. – Режим доступа: <https://learn.microsoft.com/ru-ru/aspnet/signalr/> – Дата доступа: 28.05.2025
- 7 Docker [Электронный ресурс]. – Режим доступа: <https://www.docker.com> – Дата доступа: 28.05.2025
- 8 YouTube [Электронный ресурс]. – Режим доступа: [www.youtube.com](http://www.youtube.com) – Дата доступа: 28.05.2025
- 9 RUTUBE [Электронный ресурс]. – Режим доступа: <https://rutube.ru> – Дата доступа: 28.05.2025
- 10 VK Video [Электронный ресурс]. – Режим доступа: <https://vkvideo.ru> – Дата доступа: 28.05.2025
- 11 HTTPS RFC [Электронный ресурс]. – Режим доступа: <https://datatracker.ietf.org/doc/html/rfc2616> – Дата доступа: 28.05.2025
- 12 TCP RFC [Электронный ресурс]. – Режим доступа: <https://datatracker.ietf.org/doc/html/rfc793> – Дата доступа: 28.05.2025
- 13 WebSocket RFC [Электронный ресурс]. – Режим доступа: <https://datatracker.ietf.org/doc/html/rfc6455> – Дата доступа: 28.05.2025
- 14 N-Layer [Электронный ресурс]. – Режим доступа: <https://medium.com/design-microservices-architecture-with-patterns/layered-n-layer-architecture-e15ffdb7fa42> – дата доступа: 28.05.2025
- 15 Dependency Injection (DI) [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/articles/350068/> – дата доступа: 28.05.2025
- 16 JSON Web Token (JWT) RFC [Электронный ресурс]. – Режим доступа: <https://datatracker.ietf.org/doc/html/rfc7519> – дата доступа: 28.05.2025
- 17 Паттерн «Repository» [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/articles/248505/> – дата доступа: 28.05.2025
- 18 JSON RFC [Электронный ресурс]. – Режим доступа: <https://datatracker.ietf.org/doc/html/rfc7159.html> – дата доступа: 28.05.2025

					ДП 00.00 ПЗ								
		Ф.И.О	Подпись	Дата									
Разраб	Окулич Д.Ю.				Список используемых источников				Лит.	Лист	Листов		
Пров.	Белодед Н.И.									у		1	1
									БГТУ 1-40 01 01, 2025				
Н. контр.	Белодед Н.И.												
УТВ	Смелов В В												

**Диаграмма вариантов использована (ДП 01.00.ГЧ)**

**Логическая схема базы данных (ДП 02.00.ГЧ)**

**Диаграмма развертывания (ДП 03.00.ГЧ)**

**Блок-схема подключения к совместному просмотру (ДП 04.00.ГЧ)**

**Диаграмма компонентов (ДП 05.00.ГЧ)**



**Скриншот работы приложения (ДП 06.00.ГЧ)**

## Приложение А

Таблица Users

Название	Тип данных	Описание
ID	UUID	Уникальный идентификатор пользователя (PK)
name	TEXT	Имя пользователя
email	TEXT	Электронная почта
password	TEXT	Хешированный пароль
birthday	DATE	Дата рождения
role	INTEGER	Роль пользователя
token	TEXT	Токен авторизации

Таблица Channels

Название	Тип данных	Описание
ID	UUID	Уникальный идентификатор канала (PK)
name	TEXT	Название канала
description	TEXT	Описание канала
created	TIMESTAMP	Дата создания канала
status	INTEGER	Статус канала
userId	UUID	Владелец канала (FK на Users)

Таблица Videos

Название	Тип данных	Описание
ID	UUID	Уникальный идентификатор видео (PK)
channelId	UUID	Канал-владелец (FK на Channels)
title	TEXT	Название видео
description	TEXT	Описание видео
duration	TEXT	Продолжительность видео
forAdults	BOOLEAN	Контент 18+
tags	JSONB	Список тегов
created	TIMESTAMP	Дата загрузки
videoAccess	INTEGER	Уровень доступа
videoStatus	INTEGER	Статус видео

Таблица Comments

Название	Тип данных	Описание
ID	UUID	Уникальный идентификатор комментария (PK)
message	TEXT	Текст комментария
likes	JSONB	Лайки к комментарию
created	TIMESTAMP	Дата создания
ownerId	UUID	Пользователь, оставивший комментарий (FK на Users)

videoId	UUID	Видео, к которому оставлен комментарий (FK на Videos)
---------	------	---

Таблица Playlists

Название	Тип данных	Описание
ID	UUID	Уникальный идентификатор плейлиста (PK)
name	TEXT	Название плейлиста
access	INTEGER	Доступ
userId	UUID	Владелец плейлиста (FK на Users)

Таблица PlaylistItems

Название	Тип данных	Описание
ID	UUID	Уникальный идентификатор элемента (PK)
playlistId	UUID	Идентификатор плейлиста (FK на Playlists)
videoId	UUID	Идентификатор видео (FK на Videos)
order	INTEGER	Порядок отображения в плейлисте

Таблица Subscriptions

Название	Тип данных	Описание
ID	UUID	Уникальный идентификатор подписки (PK)
userId	UUID	Подписавшийся пользователь (FK на Users)
channelId	UUID	Канал, на который подписка (FK на Channels)

Таблица Reports

Название	Тип данных	Описание
ID	UUID	Уникальный идентификатор жалобы (PK)
type	INTEGER	Тип жалобы
message	TEXT	Текст жалобы
created	TIMESTAMP	Дата подачи жалобы
videoId	UUID	Видео, на которое жалоба (FK на Videos)

Таблица VideoMarks

Название	Тип данных	Описание
ID	UUID	Уникальный идентификатор метки (PK)
userId	UUID	Пользователь, поставивший метку (FK на Users)
videoId	UUID	Видео, к которому метка (FK на Videos)

isLike	BOOLEAN	Лайк
isDisLike	BOOLEAN	Дизлайк
updated	TIMESTAMP	Дата последнего обновления метки

Таблица VideoViews

Название	Тип данных	Описание
ID	UUID	Уникальный идентификатор просмотра (PK)
userId	UUID	Пользователь, просмотревший видео (FK на Users)
videoId	UUID	Видео, которое было просмотрено (FK на Videos)
created	TIMESTAMP	Время просмотра

## Приложение Б

```
CREATE EXTENSION IF NOT EXISTS "uuid-oss";

CREATE TABLE Users (
    ID UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    name TEXT,
    email TEXT,
    password TEXT,
    birthday DATE,
    role INTEGER,
    token TEXT
);

CREATE TABLE Channels (
    ID UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    name TEXT,
    description TEXT,
    created TIMESTAMP,
    status INTEGER,
    user_id UUID REFERENCES Users(ID)
);

CREATE TABLE Videos (
    ID UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    channel_id UUID REFERENCES Channels(ID),
    title TEXT,
    description TEXT,
    duration TEXT,
    forAdults BOOLEAN,
    tags JSONB,
    created TIMESTAMP,
    videoAccess INTEGER,
    videoStatus INTEGER
);

CREATE TABLE Comments (
    ID UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    message TEXT,
    likes JSONB,
    created TIMESTAMP,
    owner_id UUID REFERENCES Users(ID),
    video_id UUID REFERENCES Videos(ID)
);

CREATE TABLE Playlists (
    ID UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    name TEXT,
    access INTEGER,
    user_id UUID REFERENCES Users(ID)
);
```

```
CREATE TABLE PlaylistItems (
```

```
  ID UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  playlistId UUID REFERENCES Playlists(ID),
  videoId UUID REFERENCES Videos(ID),
  order INTEGER
);
```

```
CREATE TABLE Subscriptions (
  ID UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  user_id UUID REFERENCES Users(ID),
  channel_id UUID REFERENCES Channels(ID)
);
```

```
CREATE TABLE Reports (
  ID UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  type INTEGER,
  message TEXT,
  created TIMESTAMP,
  video_id UUID REFERENCES Videos(ID)
);
```

```
CREATE TABLE VideoMarks (
  ID UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  user_id UUID REFERENCES Users(ID),
  video_id UUID REFERENCES Videos(ID),
  isLike BOOLEAN,
  isDisLike BOOLEAN,
  updated TIMESTAMP
);
```

```
CREATE TABLE VideoViews (
  ID UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  user_id UUID REFERENCES Users(ID),
  video_id UUID REFERENCES Videos(ID),
  created TIMESTAMP
);
```

## Приложение В

```
[HttpPost("signin")]
public async Task<IResult> SignIn([FromBody] SigninModel model)
{
    try
    {
        var user = await _db.Users.FirstAsync(i => i.Email ==
model.Email);

        if (!_passwordHasher.Verify(model.Password,
user.Password))
        {
            throw new ServerException("Пароли не совпадают");
        }
        HttpContext.Response.Cookies.Append(
            "AccessToken",
            JwtService.GenerateJwtToken(_jwtManager.AccessToken,
new() { Id = user.Id, Role = user.Role}),
            _jwtManager.JwtCookieOptions
        );
        string token = JwtService.GenerateJwtToken(
_jwtManager.RefreshToken, new() { Id = user.Id, Role = user.Role
});

        HttpContext.Response.Cookies.Append(
            "RefreshToken",
            token,
            _jwtManager.JwtCookieOptions
        );

        user.Token = token;
        _db.Users.Update(user);
        await _db.SaveChangesAsync();
        return Results.Ok();
    }
    catch (ServerException apperr)
    {
        return Results.Json(apperr.GetModel(), statusCode:
apperr.Code);
    }
}
```

## Реализация авторизации

```

[HttpGet]
public async Task<IResult> Get([FromQuery] Guid id)
{
    try
    {
        var authData = AuthorizeData.FromContext(HttpContext);

        Video video = await _dbContext.Videos.FindAsync(id)
            ?? throw new ServerException("Видео не найдено", 404);

        Channel channel = await _dbContext.Channels
            .Include(i => i.Subscribes)
            .FirstOrDefault(i => i.Id == video.ChannelId);

        if (video.VideoAccess == Video.Access.Private)
        {
            if (!authData.IsAuthorize || channel.UserId !=
authData.Id)
                throw new ServerException("Видео вам не доступно",
403);
        }

        if (video.VideoStatus == Video.Status.Blocked ||
channel.Status == Channel.ActiveStatus.Blocked)
            throw new ServerException("Видео более не доступно",
403);

        var videoLocalData = _localData.GetVideoData(id);
        var channelLocalData = _localData.GetChannelData(channel.Id);

        var views = await _dbContext.VideoViews.Where(v =>
v.VideoId == id).CountAsync();

        return Results.Json(new VideoFullModel());
    }
    catch (ServerException srverr)
    {
        return Results.Json(srverr.GetModel(), statusCode:
srverr.Code);
    }
}

```

### Реализация метода «Get» в «VideoController»



```

public async Task<IResult> StreamVideo([FromRoute] Guid id) {
    try {
        var authData = AuthorizeData.FromContext(HttpContext);
        string path = $"../Data/videos/{id}/video.mp4";

        if (!System.IO.File.Exists(path))
            throw new ServerException("Файла больше не существует!",
404);

        Video video = await _dbContext.Videos
            .Include(i => i.Channel)
            .FirstAsync(i => i.Id == id);

        if (video.VideoAccess == Video.Access.Private)
        {
            if ((authData.IsAuthorize && authData.Id !=
video.Channel.UserId) || !authData.IsAuthorize)
                throw new ServerException("Видео файл не доступен!",
403);
        }

        if (video.VideoStatus == Video.Status.Blocked)
            throw new ServerException("Видео файл не доступен!",
403);

        HttpContext.Response.Headers.Append("Accept-Ranges",
"bytes");
        var fileStream = new FileStream(path, FileMode.Open,
FileAccess.Read, FileShare.Read | FileShare.Delete);
        var fileLength = fileStream.Length;
        if (Request.Headers.ContainsKey("Range"))
        {
            var rangeHeader =
Request.Headers["Range"].ToString();
            var range = rangeHeader.Replace("bytes=",
"").Split('-');
            var start = long.Parse(range[0]);
            var end = range.Length > 1 &&
!string.IsNullOrEmpty(range[1])
                ? long.Parse(range[1])
                : fileLength - 1;

            var chunkSize = end - start + 1;
            fileStream.Seek(start, SeekOrigin.Begin);
            return Results.File(
                fileStream,
                "video/mp4",
                enableRangeProcessing: true);
        }
        return Results.File(fileStream, "video/mp4");
    }
    catch (ServerException srvErr)

```

```

        {
            return Results.Json(srvErr.GetModel(), statusCode:
srvErr.Code);
        }
    }
}

```

### Реализация метода «SteamVideo» в «VideoController»

```

[HttpGet("video")]
public async Task<IResult> GetVideoComments([FromQuery] string
vid)
{
    try
    {
        var authData = AuthorizeData.FromContext(HttpContext);

        if (!Guid.TryParse(vid, out Guid vguid))
            throw new ServerException("vId is not correct!");

        Comment[] comments = await
_commentRepository.GetVideoComments(vguid);

        return Results.Json(comments.Select(comment =>
        {
            var usrData =
_localData.GetUserData(comment.User.Id);

            bool userIsLikeIt = false;
            bool isVideoOwner = false;

            if (authData.IsAuthorize)
            {
                userIsLikeIt =
comment.Likes.Contains(authData.Id);
                isVideoOwner = comment.Video!.Channel!.UserId ==
authData.Id;
            }

            return new CommentModel()
            { ... };
        })
    }
    catch (ServerException err)
    {
        return Results.Json(err.GetModel(), statusCode:
err.Code);
    }
}

```

### Реализация метода «GetVideoComments» в «CommentController»

```

[HttpDelete, Authorize]
public async Task<IResult> Delete([FromQuery] Guid id)
{
    try
    {
        var authData = AuthorizeData.FromContext(HttpContext);

        Channel channel = await _db.Channels
            .Include(channel => channel.Videos)
            .FirstOrDefaultAsync(c => c.Id == id)
            ?? throw new ServerException("Канал не найден!", 404);

        if (channel.UserId != authData.Id)
            throw new ServerException("Канал вам не принадлежит!",
403);

        foreach (var video in _db.Videos.Where(i => i.ChannelId ==
channel.Id))
        {
            Directory.Delete($"{LocalDataService.VideosPath}/{video.Id}",
true);

            _db.Videos.Remove(video);
        }

        Directory.Delete($"{LocalDataService.ChannelsPath}/{id}", true);

        _db.Channels.Remove(channel);

        await _db.SaveChangesAsync();

        return Results.Ok();
    }

    catch (ServerException err)
    {
        return Results.Json(err.GetModel(), statusCode:
err.Code);
    }
}

```

### Реализация метода «Delete» в «ChannelController»

## Приложение Г

### Маршруты контроллеров

Путь	HTTP-метод	Контроллер	Метод	Описание
api/auth/signin	POST	AuthController	SignIn	Аутентификация пользователя.
api/auth/signout	GET	AuthController	Logout	Выход из системы.
api/auth/register	POST	AuthController	Register	Регистрация нового пользователя.
api/video	GET	VideoController	Get	Получить видео по идентификатору.
api/video/channel	GET	VideoController	GetChannelVideos	Получить список видео канала по channelId.
api/video/playlist	GET	VideoController	GetPlaylistVideos	Получить список видео из плейлиста по playlistId.
api/video/select	GET	VideoController	Select	Получить видео по параметрам SelectOptions.
api/video	POST	VideoController	Post	Создание нового видео
api/video	PUT	VideoController	Put	Редактирование информации о видео с указанным id и channelId.
api/video	DELETE	VideoController	Delete	Удаление видео по id и channelId.
/data/videos/{id}/video.mp4	GET	VideoController	StreamVideo	Потоковая передача видео по id.
api/video/mark	GET	VideoController	GetMark	Получить оценку видео по id.
api/video/mark	POST	VideoController	MarkVideo	Оценить видео по id.
api/video/view	POST	VideoController	AddView	Добавить просмотр видео по id.
api/video/view	GET	VideoController	GetViews	Получить видео по id.
api/video/delete	DELETE	VideoController	DeleteByAdmin	Удаление видео администратором по id.

api/video/status	PUT	VideoController	ChangeStatus ByAdmin	Изменить статус видео администратором по id.
api/comment	GET	CommentController	Get	Получить комментарий по его id.
api/comment/video	GET	CommentController	GetVideoComments	Получить список комментариев для видео с vid.
api/comment	POST	CommentController	Post	Создать новый комментарий
api/comment/like	POST	CommentController	LikeComment	Поставить лайк комментария по id
api/comment	PUT	CommentController	Put	Редактировать комментарий по id
api/comment	DELETE	CommentController	Delete	Удалить комментарий по id
api/channel	GET	ChannelController	Get	Получить данные о канале по id.
api/channel/check	GET	ChannelController	CheckChannel	Проверить, принадлежит ли текущему пользователю канал с указанным id.
api/channel	POST	ChannelController	Post	Создать новый канал.
api/channel	PUT	ChannelController	Put	Изменить параметры канала по id.
api/channel	DELETE	ChannelController	Delete	Удалить канал по id.
api/channel/user	GET	ChannelController	GetUserChannels	Получить список каналов, принадлежащих текущему пользователю.
api/channel/subscribe	POST	ChannelController	Subscribe	Подписаться на канал по id.
api/channel/subscribe	DELETE	ChannelController	Unsubscribe	Отписаться от канала по id.
api/channel/ getchannelsbyadmin	GET	ChannelController	GetAdmin ControllChannels	Получить список каналов для админ-контроля с фильтрацией по имени.

api/channel/statusbyadmin	PUT	ChannelController	SetStatusByAdmin	Изменить статус активности канала.
api/channel/deletebyadmin	DELETE	ChannelController	DeleteByAdmin	Удалить канал администратором по id.
api/playlist	GET	PlaylistController	Get	Получить плейлист по его id.
api/playlist/user	GET	PlaylistController	GetByUser	Получить все плейлисты текущего пользователя.
api/playlist	POST	PlaylistController	Post	Создать новый плейлист.
api/playlist/add	POST	PlaylistController	AddVideoToPlaylist	Добавить видео vid в плейлист id.
api/playlist/remove	DELETE	PlaylistController	RemoveVideoFromPlaylist	Удалить видео vid из плейлиста id.
api/playlist	PUT	PlaylistController	Put	Обновить данные плейлиста по id.
api/playlist	DELETE	PlaylistController	Delete	Удалить плейлист по id.
api/report	GET	ReportController	Get	Получить жалобу по id.
api/report	POST	ReportController	Post	Отправить жалобу на видео
api/report	DELETE	ReportController	Delete	Удалить жалобу по id.
api/report/video	GET	ReportController	GetVideoReports	Получить список жалоб на конкретное видео по vid.

## Приложение Д

```

public class Tests
{
    private readonly JwtManager jwtManager = new JwtManager(new
JwtSettings()
    {
        Audience = "http://localhost:8081/api",
        Issuer = "http://localhost:8081",
        SecretKey = "N0ek5pYWMgnq-
iaCqz811YNSxNFkhTb8oNEFooIyHBg",
        ExpiryMinutes = 1,
    },
    new JwtSettings()
    {
        Audience = "http://localhost:8081/api",
        Issuer = "http://localhost:8081",
        SecretKey = "N0ek5pYWMgnq-
iaCqz811YNSxNFkhTb8oNEFooIyHBf",
        ExpiryMinutes = 2,
    });

    [Fact]
    public void PasswordHasherTest0()
    {
        PasswordHasher hasher = new PasswordHasher("salt");

        string passwordHashed = hasher.Hash("pravoda01");

        Assert.True(hasher.Hash("pravoda01") == passwordHashed);
    }

    [Fact]
    public void PasswordHasherTest1()
    {
        PasswordHasher hasher = new PasswordHasher("salt");

        string passwordHashed = hasher.Hash("pravoda01");

        Assert.False(hasher.Hash("Pravoda01") == passwordHashed);
    }

    [Fact]
    public void JwtManagerTest0()
    {
        string token =
JwtManager.GenerateJwtToken(jwtManager.AccessToken, new User()
        {
            Id = 12,
            Name = "Test",
            Role = User.RoleType.User
        });
    }
}

```

```

        ClaimsPrincipal claims = JwtManager.ValidateToken(token,
JwtManager.GetParameters(jwtManager.AccessToken));

        Assert.Equal("12", claims.Claims.First().Value);
    }

    [Fact]
    public void JwtManagerTest1()
    {
        string token =
JwtManager.GenerateJwtToken(jwtManager.RefreshToken, new User()
        {
            Id = 13,
            Name = "Testsad",
            Role = User.RoleType.User
        });

        ClaimsPrincipal claims = JwtManager.ValidateToken(token,
JwtManager.GetParameters(jwtManager.RefreshToken));

        Assert.Equal("13", claims.Claims.First().Value);
    }

    [Fact]
    public void VideoMediaServiceTest()
    {
        VideoMediaService videoMedia = new
VideoMediaService("C:\\ffmpeg");

        var data =
videoMedia.GetMediaInfo("D:\\BYData\\GAMBLECORE.mp4");

        Assert.NotNull(data);
    }
}

```