МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Кафедра «Прикладная математика и информатика»
(наименование)

09.03.03 Прикладная информатика
(код и наименование направления подготовки / специальности)

Разработка программного обеспечения
(направленность (профиль)/специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка программного обеспечения веб-представительства турагентства»

Обучающийся Е.С. Дёмин
(Инициалы Фамилия) (личная подпись)

Руководитель д.т.н., доцент, С.В. Мкртычев
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2024

Аннотация

Тема выпускной квалификационной работы: «Разработка программного обеспечения веб-представительства турагентства».

Объектом исследования является веб-представительство турагентства.

Предметом исследования является программное обеспечение веб-представительства турагентства.

Цель работы состоит в том, чтобы создать функциональный веб-сайт для турагентства, который соответствует требованиям и тенденциям отрасли.

Данная работа состоит из введения, трех глав, заключения и списка используемой литературы.

В первой главе представлено обоснование разработки сайта, а также задачи, решаемые в процессе разработки.

Во второй главе представлено проектирование веб-сайта компании. Была разработаны логическая модель, а также внешние и внутренние структурные схемы сайта.

В третьей главе представлена разработка веб-представительства, выбор и описание программных средств, описание страниц и модулей веб-представительства, а также тестирование и публикация сайта в Интернете.

Бакалаврская работа состоит 67 страниц текста, 50 рисунков, 3 таблиц и 25 источников.

Оглавление

Введение

В наши дни стремительное развитие информационных технологий неизбежно влияет на многие области жизни людей. Одной из таких областей является туризм.

Присутствие турагентств в Интернете становится жизненно важным, поскольку оно не только предоставляет информацию о путешествиях, но и упрощает бронирование и организует идеальное путешествие за несколько кликов мыши [3].

Объектом исследования является веб-представительство турагентства.

Предметом исследования является программное обеспечение веб-представительства турагентства.

Цель работы состоит в том, чтобы создать функциональный веб-сайт для турагентства, который соответствует требованиям и тенденциям отрасли.

Для достижения поставленной цели необходимо решить следующие задачи:

- выполнить постановку задачи на разработку программного обеспечения веб-представительства турагентства;
- спроектировать программное обеспечение веб-представительства турагентства;
- реализовать и выполнить тестирование программного обеспечения веб-представительства турагентства [11].

Методы исследования: методы и технологии проектирования программного обеспечения [11].

Практическая значимость работы заключается в том, что результаты работы могут быть использованы турагентством для создания собственного веб-сайта.

Данная работа состоит из введения, трех глав, заключения и списка используемой литературы и используемых источников.

Во введении подтверждена актуальность темы работы, представлены объект, предмет, цель и задачи исследования.

В первой главе представлены обоснование разработки сайта, а также задачи, решаемые в процессе разработки.

Во второй главе представлено проектирование веб-сайта компании. Была разработаны логическая модель, а также внешние и внутренние структурные схемы сайта.

В третьей главе представлена разработка веб-представительства, выбор и описание программных средств, описание страниц и модулей веб-представительства, а также тестирование и публикация сайта в Интернете.

В заключении описываются результаты выполнения выпускной квалификационной работы.

Бакалаврская работа состоит 67 страниц текста, 50 рисунков, 3 таблиц и 25 источников.

Глава 1 Постановка задачи на разработку программного обеспечения информационной системы

1.1 Функциональные и архитектурные особенности веб-представительств организаций сферы услуг

В современном цифровом мире наличие веб-представительства является ключевым фактором для успеха бизнеса, особенно для компаний, оказывающих услуги, таких как туристические агентства. Хорошо спроектированный и функциональный веб-сайт представляя бренд, демонстрируя услуги и взаимодействуя с потенциальными клиентами. Рассмотрим основные функциональные и архитектурные особенности, которые характеризуют успешные веб-сайты компаний, оказывающих услуги, с основным акцентом на туристических агентствах.

Пользовательский опыт является ключевым при проектировании веб-сайтов для компаний, оказывающих услуги. Веб-сайты туристических агентств должны придерживаться принципов дизайна, ориентированного на пользователя, обеспечивая интуитивную навигацию, четкие призывы к действию и увлекательные интерактивные функции. Внедрение интерактивных элементов, таких как формы бронирования повышает участие пользователей и облегчает процесс бронирования.

Полное представление информации является ключевым аспектом для веб-сайтов туристических агентств. Путешественники часто ищут подробную информацию перед принятием решения о бронировании. Поэтому веб-сайты туристических агентств должны представлять полную информацию о турах. Четкое и краткое содержание, сопровождаемое высококачественными изображениями и мультимедийным контентом, обогащает пользовательский опыт и помогает пользователям принять решение.

Визуальный дизайн играет важную роль в привлечении внимания пользователей и передаче идентификации бренда туристического агентства. Использование привлекательных макетов, изображений и стилей бренда усиливает узнаваемость бренда и способствует доверию потенциальных клиентов. Согласованная визуальная идентичность на всем сайте подчеркивает профессионализм и доверие к агентству [1].

С увеличением числа мобильных устройств веб-сайты туристических агентств должны быть оптимизированы для беспрепятственной работы на различных размерах экранов и устройствах. Адаптивный дизайн гарантирует, что веб-сайт легко адаптируется под различные устройства, обеспечивая пользователям одинаковый и приятный опыт просмотра, независимо от того, используют ли они компьютер, планшет или смартфон [10].

Видимость в результатах поиска является ключевым аспектом привлечения органического трафика на веб-сайт. Туристические агентства должны внедрить эффективные стратегии SEO для улучшения рейтинга своего веб-сайта в результатах поиска. Это включает в себя оптимизацию структуры веб-сайта, использование ключевых слов, создание мета-описаний и регулярное обновление контента для обеспечения его актуальности [10].

Поддержка клиентов необходима для решения вопросов и выстраивания доверительных отношений с клиентами. Веб-сайты туристических агентств должны включать различные инструменты поддержки и взаимодействия с клиентами, такие как функционал онлайн-чата, контактные формы, разделы часто задаваемых вопросов, чтобы облегчить коммуникацию и улучшить общий опыт клиента.

Постоянное совершенствование необходимо для оптимизации производительности веб-сайта и достижения бизнес-целей. Туристические агентства должны внедрять надежные аналитические инструменты для отслеживания ключевых показателей производительности, поведения пользователей, конверсий и других соответствующих данных. Анализ аналитики и метрик производительности позволяет выявлять сильные и

слабые стороны, а также области для улучшения функциональности и пользовательского опыта веб-сайта.

Поддержка успешного веб-сайта туристического агентства требует постоянного внимания и обновлений для обеспечения актуальности, безопасности и производительности. Регулярное обновление контента, исправление ошибок, внедрение патчей безопасности и оптимизация скорости работы веб-сайта являются ключевыми задачами для сохранения его эффективности и удобства использования. Кроме того, следование тенденциям отрасли, технологическим инновациям и обратной связи от клиентов позволяет туристическим агентствам адаптировать и совершенствовать свои стратегии веб-сайтов для удовлетворения меняющихся требований рынка и ожиданий пользователей.

1.2 Разработка требований к веб-представительству турагентства

При разработке веб-сайта туристического агентства критически важно установить четкие требования, которые соответствуют бизнес-целям и удовлетворяют потребностям целевой аудитории.

Для разработки требований к веб-представительству турагентства мы используем методологию FURPS+, которая является классификацией требований к программным системам. В данной классификации выделяются 5 требований:

- функциональные требования (Functionality) - это спецификации и описания функциональности системы или программного обеспечения, которые определяют, как система должна взаимодействовать с пользователями, другими системами или средой окружения. Эти требования описывают конкретные действия или функции, которые система должна выполнять, чтобы удовлетворить потребности пользователей или решить определенные бизнес-задачи;

– удобство использования (Usability) определяет внешний вид, структуру и пользовательский интерфейс системы или продукта. Эти требования описывают, как система должна быть оформлена и организована, чтобы обеспечить удобство использования, привлекательный внешний вид и соответствие бренду или стилю компании;
– надежность (Reliability) обеспечивает устойчивость системы и возможность ее быстрого восстановления в случае выхода из строя;
– производительность (Performance) оценивается временем отклика и скоростью работы программы;
– поддерживаемость (Supportability) определяет адаптивность программы, ее удобство обслуживания, возможность расширения программы и ее настройки [8].

В таблице 1 представлены требования к веб-представительству турагентства по методологии FURPS+.

Таблица 1 – Требования к веб-представительству турагентства в методологии FURPS+

Требование	Статус	Полезность	Риск	Стабильность
Функциональные требования				
Главная страница является входной точкой веб-сайта	Обязательное	Важная	Низкий	Высокая
Наличие системы бронирования туров	Обязательное	Критическая	Высокий	Высокая
Возможность просматривать все туры	Обязательное	Критическая	Средний	Высокая
Каждое направление имеет собственную страницу с подробной информацией и ценах	Обязательное	Критическая	Средний	Высокая

Продолжение таблицы 1

Требование	Статус	Полезность	Риск	Стабильность
Требования к юзабилити				
Адаптивность и удобство использования на различных устройствах	Обязательное	Важная	Средний	Средняя
Интуитивная навигация и дружественный интерфейс	Обязательное	Критическая	Средний	Средняя
Требования к надёжности				
Среднее время сбоев: до 1 минуты 1 в день	Обязательное	Критическая	Средний	Средняя
Доступность: 24/7	Обязательное	Критическая	Средний	Высокая
Требования к производительности				
Время загрузки сайта: до 2 секунд	Обязательное	Важная	Средний	Высокая
Требования к поддержке				
Время устранения критических проблем: в течение 1 часа	Обязательное	Критическая	Высокий	Средняя

Разработанные требования являются основой для проектирования веб-представительства турагентства.

1.3 Анализ аналогов веб-представительства турагентства

Современная индустрия туризма сталкивается с быстрыми изменениями и возрастающими требованиями со стороны клиентов. В условиях растущей конкуренции и постоянно меняющихся тенденций в поведении потребителей, турагентствам необходимо эффективно адаптироваться и использовать передовые технологии для обеспечения качественного обслуживания и удовлетворения потребностей клиентов.

Перед тем как приступить к разработке сайта, нам необходимо провести анализ сайтов конкурентов и основываясь на результатах выявить особенности, которые дадут нашему веб-представительству конкурентное преимущество.

Рассмотрим туристическую компанию Joy Tour, которая занимается авторскими турами в Италию. К сожалению, у данного агентства нет собственного сайта. Вместо этого у них есть аккаунт в Instagram, где можно ознакомиться с услугами компании [17]. Скриншот аккаунта представлен на рисунке 1.

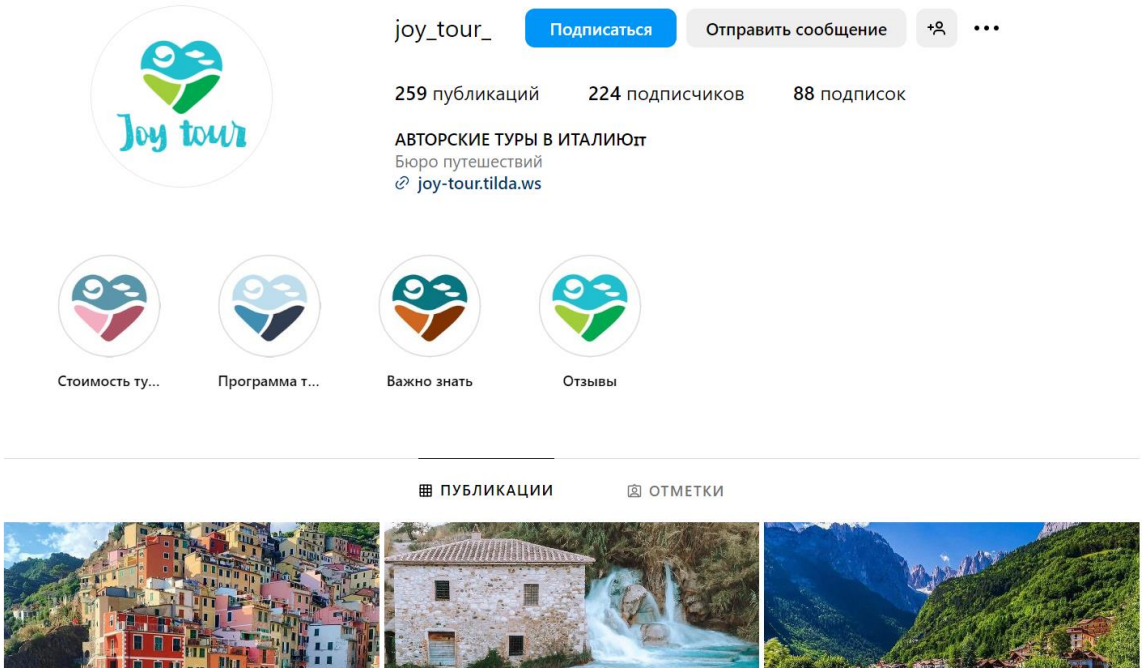

Рисунок 1 – Аккаунт компании Joy Tour в Instagram

Практически каждая компания, помимо собственного сайта, имеет аккаунт в популярных соцсетях, но в качестве дополнительного канала рекламы и связи с аудиторий, что не скажешь об агентстве Joy Tour, которая использует соцсети в качестве основного средства продажи своих услуг. У пользователя нет возможности быстро найти и ознакомиться с предлагаемыми турами. Посты служат обычной галереей с фотографиями из Италии, когда

обычный пользователь ожидает, что посты с фотографиями относятся к определенному туру, а в описании под фотографиями можно найти детали и стоимость тура.

С программами и ценами туров можно ознакомиться в сторис, пример которого представлен на рисунке 2.

Рисунок 2 – Сторис с описанием пакета тура

Также турагентство не предоставляет возможность бронирования тура онлайн. Вместо этого необходимо связаться с менеджерами в Instagram или по другим каналам связи, например, телефон или электронная почта. Таким образом, отсутствие собственного сайта означает для турагентства потерю клиентов, так как нет удобной возможности ознакомиться со всеми турами, а также детальной информацией к ним. Для этого пользователю придется связаться с менеджером и узнать всю необходимую информацию. Если появятся какие-то дополнительные вопросы, то это означает, что нужно снова

обращаться в турагентство и ждать ответа. Даже если потенциальный клиент решит забронировать тур, то придется снова писать менеджеру. Также отсутствие сайта создает недоверие к данному турагентству, создавая впечатление, что это мошенники или молодая компания, которая совсем не имеет опыта.

У другого турагентства DIANA ESPOSITO TRAVEL, которое также занимается авторскими турами в Италию, основным каналом продаж и связи с клиентами является Instagram [15]. На рисунке 3 представлен скриншот аккаунта.

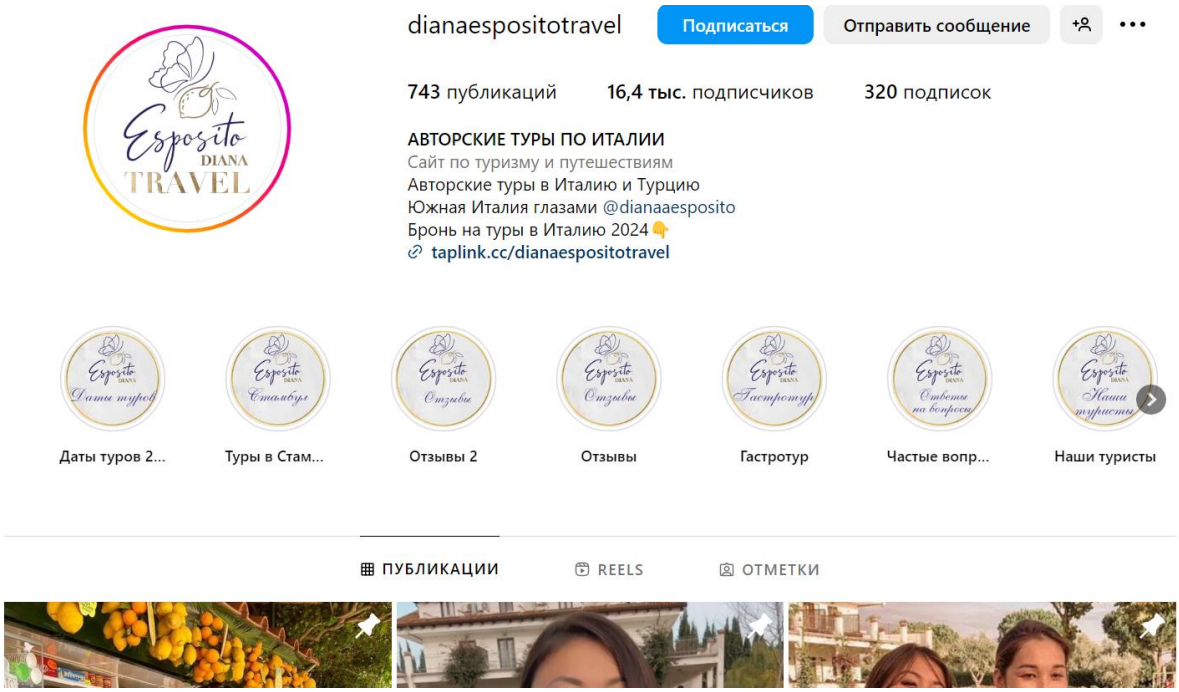

Рисунок 3 – Аккаунт компании DIANA ESPOSITO TRAVEL в Instagram

В шапке профиля есть ссылка, которая, как ожидалось, ведет на сайт компании, но, перейдя по ссылке, мы видим, что страница, скриншот которой представлен на рисунке 4, не содержит никакой информации кроме ссылок на мессенджеры, где можно связаться с менеджерами для консультации по турам.

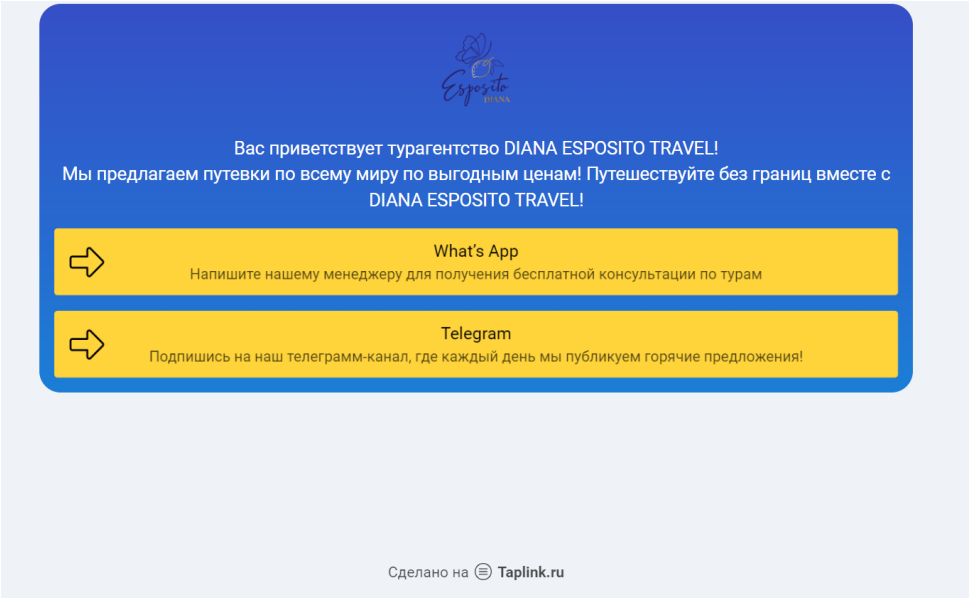

Рисунок 4 – Ссылки на мессенджеры

Таким образом, у клиентом снова нет удобного способа ознакомиться со всеми турами детально, а также забронировать тур. Вместо этого снова придется выходить на связь с менеджерами.

Рассмотрим другого конкурента ItaliaMania, который позиционирует себя как турагентство, которое предоставляет многочисленные туры в Италию с разной программой – от туров выходного дня до учебных и бизнес-туров [16]. У данной компании есть собственный сайт, скриншот главной страницы которого представлен на рисунке 5.

Рисунок 5 – Скриншот главной страницы компании ItaliaMania

В боковом меню есть хорошая возможно просмотреть сразу все разделы сайта, а именно услуги, которые предоставляет данная компания, но список слишком длинный и при прокручивании страницы вниз приходится напрягать зрение, чтобы не пропустить тот или иной пункт списка.

Также на главной странице в главном блоке мы видим снова список всех видов туров, которые есть и в боковом меню. Такое решение излишне. В центральной части, особенно главной страницы, необходимо разместить наиболее важный контент, который призван заинтересовать внимание клиента, например, блок со специальными предложениями.

Выберем раздел «Экскурсионные туры в Италию», скриншот которого представлен на рисунке 6.

Рисунок 6 – Скриншот страницы «Экскурсионные туры в Италию»

На странице экскурсионных туров не совсем понятна необходимость располагать кнопку «Оставить заявку на тур» на странице со списком туров вместо страницы конкретного тура. Если нажать на эту кнопку, то появится всплывающее окно, которое представлено на рисунке 7.

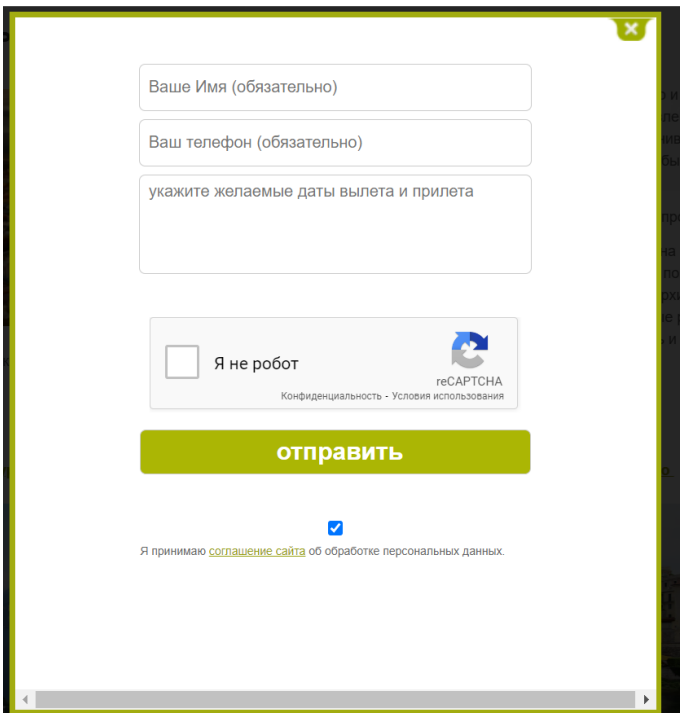

Рисунок 7 – Всплывающее окно «Оставить заявку»

Во всплывающем окне находится форма, где необходимо заполнить имя, номер телефона и желаемые даты вылета и прилета. Но сразу возникает вопросы: «Даты вылета куда? На какой тур мы оставляем заявку?». По сути это обычная форма для связи.

Нажмем на любой из туров, например, «Экскурсионные туры в Лигурию». На рисунке 8 представлен скриншот детальной страницы тура.

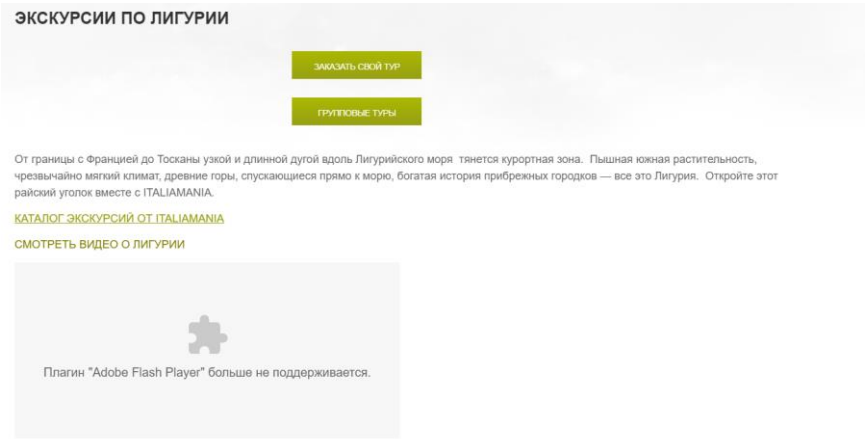

Рисунок 8 – Детальная страница экскурсионного тура

На детальной странице тура мы видим, что вместо ожидаемой кнопки «Забронировать тур» или хотя бы «Заказать тур» мы видим кнопку «Заказать свой тур», из-за которой возникает непонимание предназначения кнопки – предлагают ли нам забронировать данный тур или заказать свой? Если нажать на эту кнопку, то снова видим всплывающее окно, представленное на рисунке 9, которое предлагает нам заполнить контактную информацию и дату вылета и прилета.

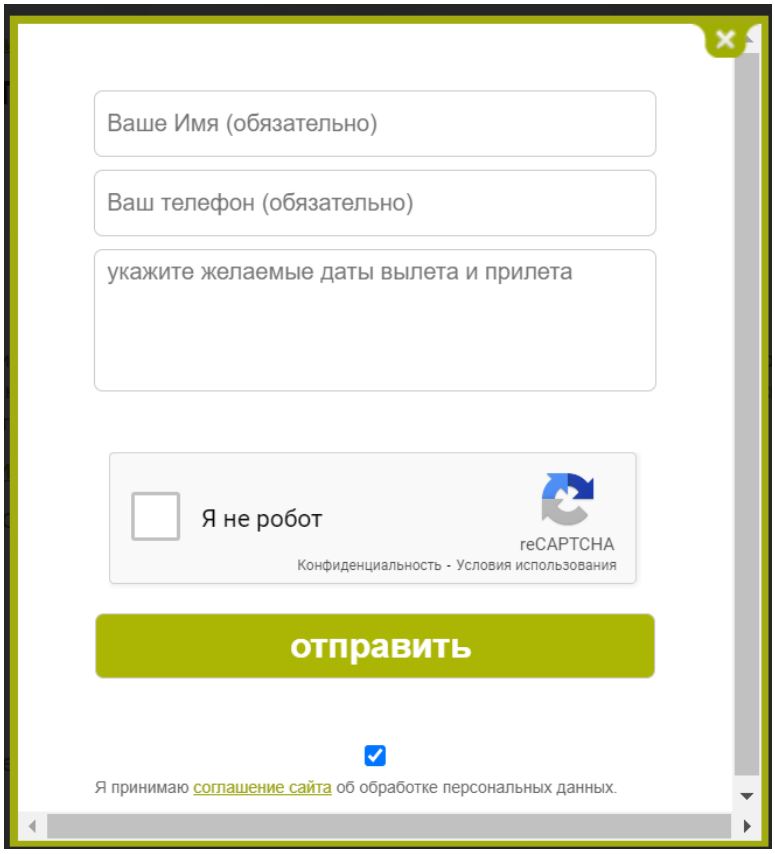

Рисунок 9 – Всплывающее окно «Оставить заявку»

Кнопка «Групповые туры» ведет на страницу с совершенно другими программами, которые отвечают за туры в группах.

Видео о туре просмотреть невозможно, так как на сайте используется устаревшая технология Adobe Flash Player, которая больше не поддерживается современными браузерами.

Ссылка «Каталог экскурсий от ItaliaMania» открывает PDF-файл, который содержит 8 страниц со справочной информацией о Лигурии. На рисунке 10 представлен скриншот PDF-файла со справочной информацией.

Рисунок 10 – Справочная информация о Лигурии

Для определения соответствия требованиям к реализации веб-представительства турагентства используем таблицу 2 и следующие критерии оценивания:

- полное несоответствие требованиям - 0;
- значительное несоответствие требованиям - 1;
- незначительное несоответствие требованиям - 2;
- полное соответствие требованиям – 3.

Таблица 2 – Сравнительный анализ аналогов веб-представительств турагентств

Характеристика	Joy Tour	DIANA ESPOSITO TRAVEL	ItaliaMania
Наличие системы бронирования туров	0	0	1
Возможность просматривать все туры	0	2	3
Каждое направление имеет собственную страницу с подробной информацией и ценах	0	1	3
Адаптивность и удобство использования на различных устройствах	3	3	2
Интуитивная навигация и дружественный интерфейс	0	0	1
Итого	3	6	10

Проведя сравнительный анализ трех конкурентов, мы выявили, что их решения частично или полностью не соответствуют критериям. Компании Joy Tour и DIANA ESPOSITO TRAVEL вообще не имеют собственных сайтов, где можно было бы ознакомиться с компанией и турами. Несмотря на то, что агентство ItaliaMania имеет собственный сайт, он содержит множество недостатков – визуальное оформление не соответствует современным трендам, ., некоторые аспекты оставлены без внимания, например, поддержка Adobe Flash Player приостановлена, в результате чего пользователи не могут просмотреть видео. Также клиенты ожидают краткое описание тура на его детальной странице, а не в отдельном PDF-файле на несколько страниц. Что касается формы бронирования, то ни у одного из рассмотренных конкурентов

его нет. Необходимо напрямую связывать с менеджерами либо, как у ItaliaMania, необходимо заполнить форму для связи с датами вылета и прилета в обычном текстовом формате.

Таким образом, можно сделать вывод, что наше веб-представительство должно иметь то, чего нет у конкурентов. Для этого разрабатываемое веб-представительство должно отвечать следующим требованиям:

- иметь современный дизайн;
- быть удобным в использовании;
- быть понятным для пользователей;
- иметь детальное описание туров с самой необходимой информацией
- иметь форму для связи;
- иметь форму бронирования тура;
- форма бронирования должна иметь возможность выбора дат при помощи календаря;
- пользователи имеют возможность увидеть итоговую стоимость тура.

В результате реализации данных требований веб-представительство турагентства станет мощным инструментом, который позволит достичь поставленных целей, таких как увеличение количества клиентов, повышение узнаваемости бренда и, как следствие, рост прибыли.

Выводы по главе 1

В 1 главе выпускной квалификационной работы были рассмотрены функциональные и архитектурные особенности веб-представительств туристических агентств, а также был проведен анализ аналогов и были сформулированы требования к нашему веб-представительству. На основании проведенного анализа, мы сделали вывод, что наше веб-представительство должно иметь ряд преимуществ перед сайтами конкурентов.

Для этого веб-представительство турагентства должно отвечать следующим требованиям:

- сайт должен иметь привлекательный и актуальный дизайн, соответствующий современным трендам веб-разработки;
- навигация по сайту должна быть простой и интуитивно понятной, чтобы пользователи могли легко найти необходимую информацию;
- информация на сайте должна быть представлена в простой и понятной форме;
- на сайте должны быть представлены подробные описания всех туров, включая информацию о программе тура, стоимости, датах вылета и прилета, условиях проживания и т.д.;
- на сайте должна быть удобная форма для связи с менеджерами турагентства, чтобы пользователи могли задать вопросы или забронировать тур;
- на сайте должна быть возможность онлайн-бронирования туров;
- должна быть возможность выбора дат при помощи календаря;
- должна отображаться итоговая стоимость тура.

Соблюдение этих требований позволит создать веб-представительство турагентства, которое будет не только привлекательным и удобным для пользователей, но и эффективным инструментом для достижения бизнес-целей компании.

Глава 2 Проектирование программного обеспечения веб-представительства турагентства

2.1 Логическое моделирование веб-представительства

Для концептуального моделирования используется унифицированный язык моделирования UML, который обеспечивает единый визуальный язык для проектирования и внедрения программных средств [2].

Диаграмма прецедентов является одним из графических инструментов для описания вариантов использования системы, а также ее взаимодействия с внешними сущностями [18].

В разрабатываемом нами веб-представительстве мы выделим следующие акторы:

– клиенты;

– менеджеры по работе с клиентами.

Так как веб-представительство используется для обратной связи с клиентом, а также предоставляет возможность забронировать тур, то исходя из этого, мы выделили варианты использования и создали диаграмму прецедентов, которая представлена на рисунке 11 [23].

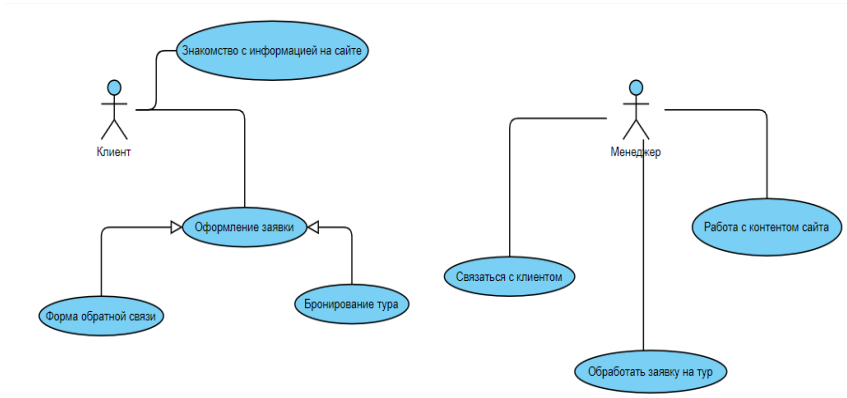

Рисунок 11 – Диаграмма прецедентов веб-представительства

На сайте клиент может получиться информацию о турагентстве, контактной и адресной информацией, а также предлагаемыми турами. Кроме того, клиент имеет возможность связаться с агентством, заполнив контактную форму. Сайт также предлагает возможность забронировать тур.

Менеджеры обрабатывают заявки клиентов на туры, а также отвечают клиентам на их обращения через контактную форму или иные виды связи. Также менеджеры занимаются контентом сайтом и заботятся, чтобы он всегда был в актуальном состоянии.

На диаграмме деятельности, которая представлена на рисунке 12, мы детализировали прецедент «бронирование тура» [23].

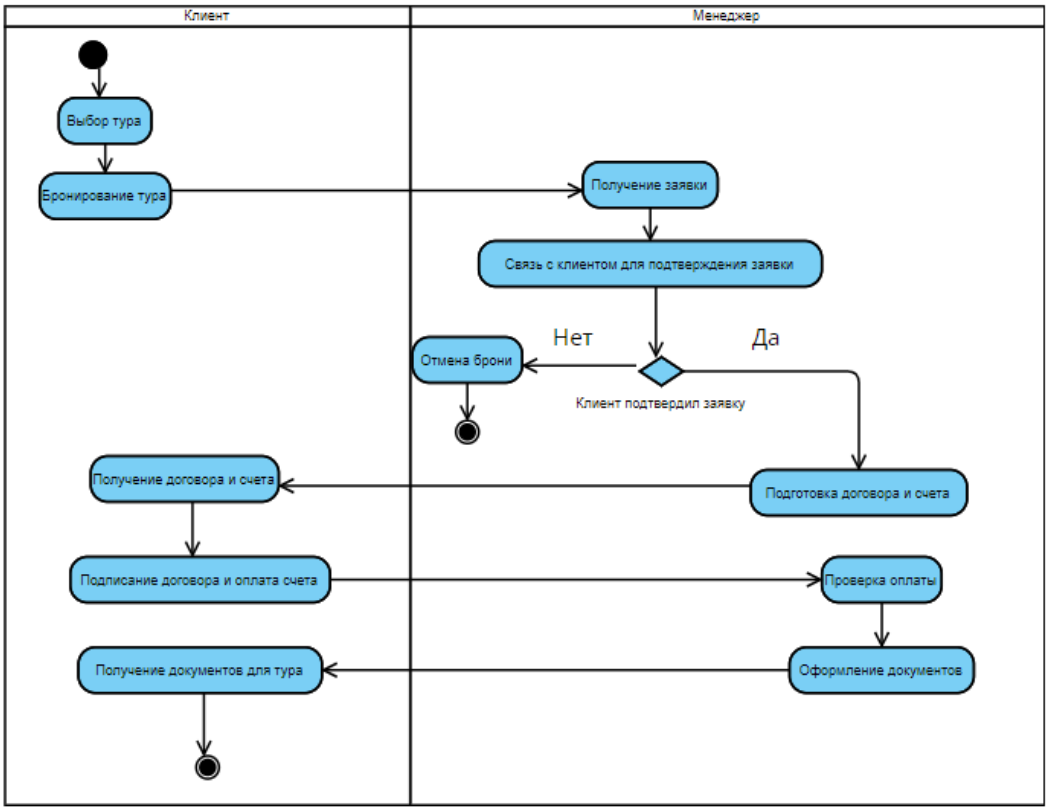

Рисунок 12 – Диаграмма деятельности процесса бронирования тура

На диаграмме мы видим, что клиент выбирает тур на сайте и оформляет бронирование. Менеджер связывается с клиентом, чтобы подтвердить бронирование. Если клиент подтверждает бронирование, менеджер

подготавливает договор и счет на оплату, затем отправляет их клиенту, который в свою очередь подписывает договор и оплачивает счет. Затем менеджер проверяет оплату, оформляет тур и отправляет необходимые документы клиенту.

Диаграмма классов является одной из ключевых моделей в языке моделирования UML (Unified Modeling Language) [9]. Она используется для представления статической структуры системы, показывая ее классы, атрибуты, методы и отношения между объектами. Диаграммы классов помогают разработчикам и архитекторам понять, как компоненты системы взаимодействуют друг с другом, и служат основой для последующей реализации кода.

На основе анализа требований и функций веб-представительства туристического агентства, мы выделили основные сущности и их взаимосвязи, которые представлены на рисунке 13 [23].

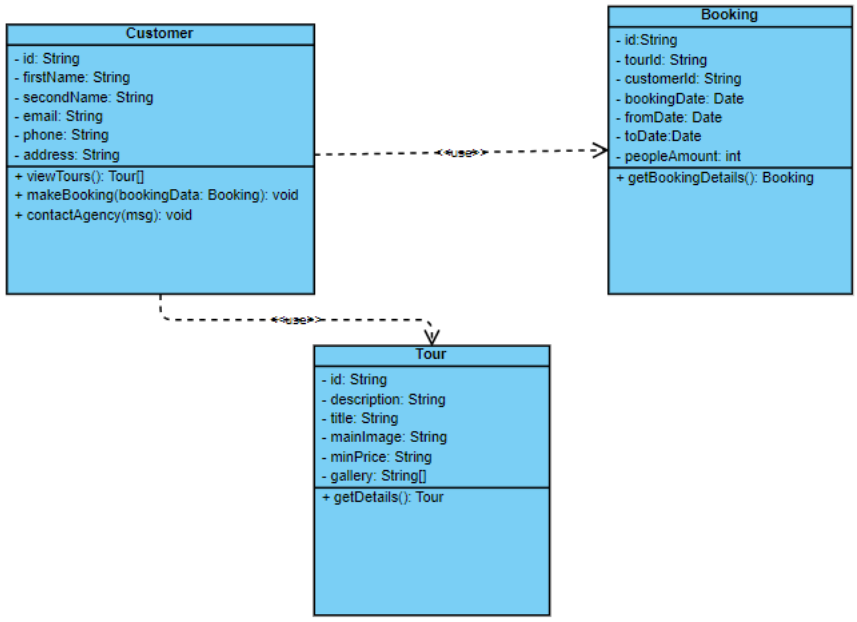

Рисунок 13 – Диаграмма классов веб-представительства

В таблице 3 представлена спецификация классов веб-представительства.

Таблица 3 – Спецификация диаграммы классов веб-представительства

Класс	Описание
Customer	Класс объектов, моделирующих на логическом уровне пользователей
Booking	Класс объектов, моделирующих на логическом уровне бронирований
Tour	Класс объектов, моделирующих на логическом уровне туров

Диаграмма классов является важным инструментом для проектирования и разработки веб-представительства туристического агентства. Она помогает визуализировать структуру системы, определить основные сущности и их взаимодействия, а также служит основой для написания кода [18]. В результате, разработка веб-сайта становится более структурированной и понятной, что способствует созданию качественного и функционального продукта.

2.2 Логическое моделирование данных веб-представительства

Существуют две популярные модели баз данных – SQL и NoSQL. Так как каждая из них обладает своими преимуществами и недостатками, нам необходимо поняться, в чем состоит их различие.

Язык структурированных запросов SQL (Structured Query Language) используется для управления и манипулирования реляционными базами данных. SQL подходит для того, чтобы хранить и управлять данными, которые обладают структурированной природой [12].

Нереляционные базы данных NoSQL используют различные структуры для хранения данных, например, документы или ключ-значение. Нереляционные базы данных подходят для данных, которые не обладают структурированной природой [14].

Для нашего проекта мы выбрали NoSQL-базы данных по ряду причин.

У NoSQL отсутствует фиксированное количество столбцов, за счет чего легко адаптируется к новым схемам данных. Кроме того, разработчику не нужно заранее продумывать и создавать схему базы данных, что делает NoSQL гибким и ускоряет процесс разработки, что не скажешь об SQL, для работы с которым необходимо тщательно спроектировать схему.

Несмотря на то, что SQL-базы данных могут применяться для широкого круга задач, часто они подходят для работы с данными и транзакциями, которые нуждаются в строгом контроле, например, для хранения и обработок транзакций в финансовом секторе. NoSQL тоже является универсальным и может применяться в приложениях, работающих с разными источниками данных и имеющих различную структуру. Также NoSQL имеет простой язык запросов, которого достаточно для наших задач.

Базы данных NoSQL бывают четырех видов:

- хранилище типа «ключ-значение»;
- документные хранилища;
- столбцовые хранилища;
- графовые хранилища [12].

В хранилищах типа «ключ-значение» можно хранить значения, которые связаны с ключом. Этот вариант хорошо подходит для быстрого поиска информации. Такие хранилища часто используются для кэширования данных.

В документных хранилищах информация хранится в файлах формата JSON, XML или BSON. Документы одного типа могут быть объединены в коллекции. Данный тип подходит для хранения и управления контентом, каталожной информации и др.

Столбцововые хранилища представляют собой столбцы, каждый из которых храниться в виде логического массива значений. Данный тип подходит для аналитических операций [19].

Графовые хранилища состоят из структурных единиц, которые называются узлами и представляют собой изолированный документ, хранящий данные произвольной формы. В то же время, узлы связаны между

собой ребрами [19]. Данный типа хранилища подходит для таких случаев, когда необходимо создавать взаимосвязи между данными, например, в сервисах рекомендаций, оптимизации маршрутов и др.

В нашем веб-представительстве мы будем использовать документно-ориентированные хранилища. В документах мы можем хранить данные о турах, бронированиях и клиентах, объединяя их в соответствующие коллекции.

База данных состоит из трех базовых коллекций:

- tours, которая хранит в себе документы, отвечающие за информацию о турах;
- booking, в которой хранится информация о бронировании;
- customers, которая хранит информацию о клиенте.

Все документы имеют уникальный идентификатор (id).

В документе tour хранится описание тура (description), галерея изображений (gallery), которая представляет из себя массив строк, в которых хранится путь к изображениям на файловом хранилище, также путь к главному изображению (mainImage), минимальная цена за одного человека (minPrice) и название тура (title).

В документе booking хранится уникальный идентификатор клиента (customerId), дата отправления (fromDate), дата окончания тура (toDate), количество туристов (peopleAmount) и уникальный идентификатор тура (tourId).

В документе customer хранится имя клиента (firstName), фамилия клиента (secondName), номер телефона (phone) и адрес электронной почты (email).

2.3 Разработка внутренней структуры веб-представительства

Разработка сайта или любого другого ресурса начинается с создания структуры, которая представляет собой логическую схему расположения

страниц сайта и его разделов относительно друг друга. Хорошо продуманная структурная схема позволяет избежать ошибки на последующих этапах разработки, исправление которых будет стоить дорого и затронет значительную часть ресурса.

Требования к структуре сайта могут различаться в зависимости от решаемых им бизнес-задач, например, схема онлайн-магазина будет намного сложнее и состоять из множества страниц, разделов и подразделов, когда для сайта-визитки достаточно будет одной-двух страниц. Но несмотря на это, все же существуют единые требования к схеме, а именно:

- любая схема должна учитывать аудиторию сайта, а именно ее возраст, поведение и интересы;
- схема должна сделать поиск информации легче. Навигация должна быть интуитивно понятной, а страницы и разделы должны иметь логическую связь. Это гарантирует, что пользователь легко и быстро доберется до нужной информации, а не покинет ресурс из-за его непонятности [13].

Разобравшись с тем, что необходимо учитывать при создании схемы сайта, рассмотрим виды структур и выберем подходящую для нашего проекта. Разные источники выделяют различное количество схем, но они представляют собой лишь модификации двух основных схем: линейной и древовидной.

В линейной структуре, представленной на рисунке 14, все страницы сайта взаимосвязаны последовательно друг за другом по цепочке, которая берет начало от главной страницы. Пользователь должен пройти через все цепочку страниц от главной к самой последней, что является в то же время недостатком, так как посетитель не имеет быстрого доступа к информации. Подобные структуры удобны для простых проектов, например, сайтов-портфолио или визиток [13].

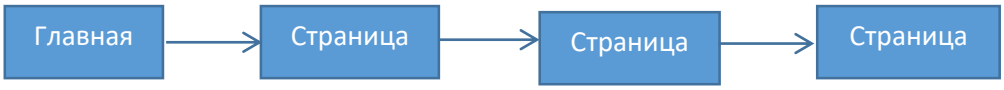

Рисунок 14 – Линейная структурная схема

Иерархическая, или древовидная, структура, которая представлена на рисунке 15, является универсальным вариантом, который используется подавляющей частью ресурсов. Древовидная схема представляет собой несколько иерархически расположенных уровней. На вершине находится главная страница, от которой ответвляются остальные разделы сайта, в которые могут быть вложены другие подразделы. Такая схема является наиболее удобной и наглядной для пользователей, а так как она позволяет структурировать большое количество информации, то иерархическая схема удобна для сайтов, которые состоят из большого количества страниц с разной смысловой нагрузкой. Именно этот вид структурных схем подходит для нашего сайта туристического агентства.

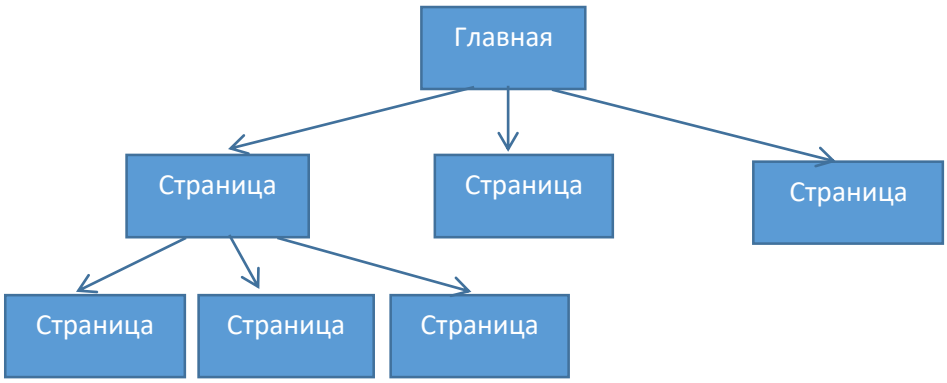

Рисунок 15 – Иерархическая структурная схема

Выбрав нужный вид структурной схемы, создадим структуру нашего сайта, которая представлена на рисунке 16.

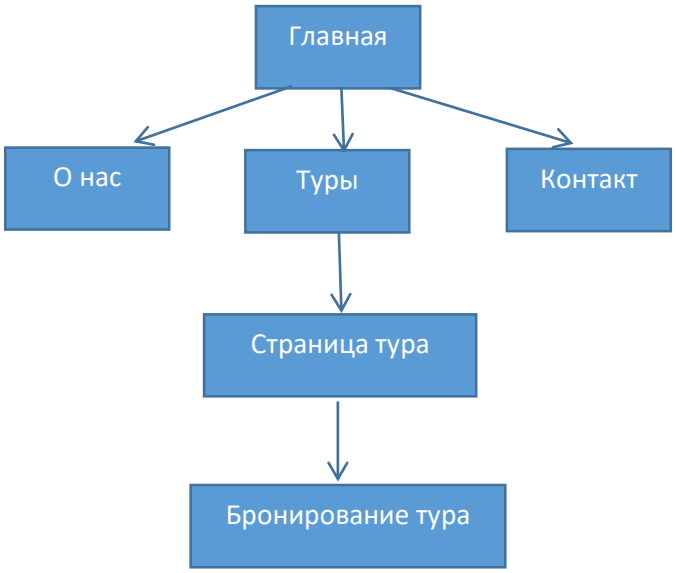

Рисунок 16 – Структурная схема сайта турагентства

Когда пользователь открывает сайт, то он сразу попадает на главную страницу. Перемещаться по сайту посетитель может с помощью меню, расположенного в шапке и подвале сайта.

На главной странице размещается основная информация, которая необходима для привлечения внимания пользователя, например, акции и специальные предложения.

На странице «Туры» содержится список туров, которые предоставляются туристическим агентством. При нажатии на отдельный тур, пользователь переходит на отдельную страницу этого тура, где расположена более подробная о нем информация.

На странице «О нас» можно ознакомиться с самой фирмой, ее историей, преимуществами и ценностями.

Раздел «Контакты» содержит адрес, контактные данные, а также форму связи.

На странице тура содержится детальная информация о туре, фотогалерея, а также кнопка «Забронировать», при нажатии на которую, пользователь перенаправляется на страницу бронирования тура.

Структура нашего сайта является простой и интуитивно понятной для пользователей, что является важным фактором для удобства пользователей, а это в первую очередь является гарантией того, что пользователь захочет остаться на странице и заинтересуется туристическими предложениями данной турфирмы.

2.4 Разработка внешней структуры веб-представительства

В то время как внутренняя структура сайта описывает логические связи между страницами и разделами сайта, внешняя схема является макетом страницы, которая описывает расположение элементов в виде блоков относительно друг друга.

Существуют четыре стандартных блока, которые присутствуют практически на каждом сайте:

- шапка (header), в которой обычно размещаются логотип, главное меню, а также кнопки регистрации и логина;
- боковая панель (sidebar), которая может располагаться справа, слева или с двух сторон. В этом блоке обычно размещается дополнительная информация, меню сайта или раздела, а также реклама;
- основной блок (content), который содержит основную информацию, которая прежде всего интересует посетителя сайта;
- подвал (footer), в котором располагаются контактная и адресная информация, ссылки на соцсети и дополнительные ссылки [21].

Так как внешняя структура определяет то, как сайт выглядит для пользователя, ее разработка является немаловажным этапом, которому стоит уделить должное внимание. Выше приведенные элементы являются шаблонной организацией страниц сайта, привычной пользователям, поэтому стоит придерживаться этому шаблону. Например, размещение главного меню только в футере вместо хедера приведет к тому, что пользователю будет

трудно найти навигацию, а без нее и нужную информацию, что обрекает сайт на неудачу.

Внешняя структура нашего сайта будет выглядеть как на рисунке 17.

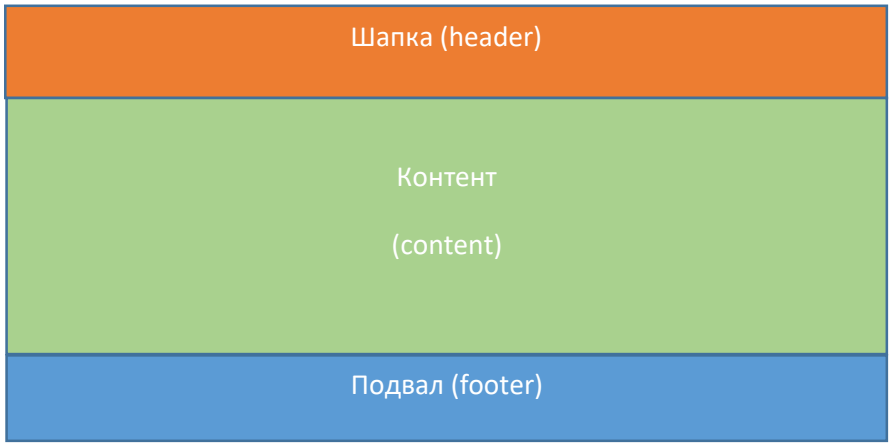

Рисунок 17 – Внешняя структурная схема сайта турагентства

Внешняя структура нашего сайта состоит из трех структурных блоков:

– шапка, в которой мы разместим логотип агентства и меню;
– основной блок, в котором будет располагаться главное содержимое;
– подвал, которой будет содержать меню, адрес агентства, контактную информацию и ссылки на соцсети.

Таким образом организация структурных блоков и контента нашего сайта соответствует общепринятому шаблону, что делает пользование нашим ресурсом удобным, логичным и понятным пользователю.

2.5 Разработка графического дизайна веб-представительства

Под дизайном сайта подразумевается его визуальное оформление. Так как пользователи сайта первым делом обращают внимание на его интерфейс, разработка качественного дизайна во многом определяет успешность сайта, который должен быть не только удобным, но и уникальным.

Дизайн любого сайта состоит из 4 элементов:

– цвет;
– графические элементы;
– шрифты;
– иконки.

Выбор этих элементов должен соответствовать правилам и трендам дизайна, целевой аудитории сайта, а также самому бренду. Например, цвета не должны быть кричащими и кислотными, а соответствовать цвету бренда.

Сайт должен обладать единой цветовой гаммой, что улучшает восприятие информации. Фон и цвет текста должны гармонировать друг с другом и контрастировать, так чтобы текст был читабельным. Цвет текста основного содержания должен быть выполнен в спокойных и приглушенных тонах. Более яркие цвета можно использовать, например, для заголовков.

Стиль текста должен быть на всем сайте единым, а шрифт соответствовать фирменному стилю. Обычно для сайта достаточно одного-двух шрифтов, которые необходимо подобрать так, чтобы они соответствовали содержанию сайта или их назначению, например, один шрифт для основной информации, а другой для заголовков.

Графические элементы должны быть качественными и гармонировать с цветовой гаммой сайта. Не стоит использовать слишком много графических элементов, так как их избыток отвлекает внимание пользователя. Кроме того, большое количество графики замедляет загрузку страницы.

Дизайн сайта должен в первую очередь соответствовать тематике. Например, сайт коммерческой фирмы должен обладать фирменным стилем и привлекать внимание клиентов, а также быть удобным и содержать информацию, которая полезна пользователям. Вместе с тем сайт должен обладать всем необходимым функционалом таким как формы связи, оформления и оплаты заказов и другие функции, которые помогают в достижении задач.

Иногда дизайнеры в погоне за уникальным дизайном уделяют большое внимание красоте интерфейса, делая его вычурным и сложным. Пользователю

зачастую не интересны нестандартные решения дизайнера, например, при разработке главного меню, в котором сложно разобраться. Необходимо помнить, что клиент приходит на сайт за информацией, которая должна быть на поверхности. Поэтому дизайн - это не только о красоте, но и о функциональности и задачах, ради которых создается сайт.

Основные принципы и правила веб-дизайна, рассмотренные в данной главе, а также представленный дизайн страниц для будущего сайта турагентства, являются фундаментом для разработки сайта и призваны обеспечить удобство и привлекательность для пользователей.

Выводы по главе 2

В этой главе мы подробно рассмотрели этапы проектирования веб-представительства турагентства, включая:

- логическое моделирование: были созданы диаграмма прецедентов, диаграмма деятельности, диаграмма классов и логическая модель данных;
- разработка внутренней структуры сайта: выбран тип иерархической структуры сайта как наиболее подходящий для данного проекта и представлена структурная схема сайта, отображающая логическую связь между страницами;
- разработка внешней структуры сайта: описаны стандартные блоки, используемые в структуре веб-сайтов и представлена внешняя структурная схема сайта, демонстрирующая расположение блоков на странице.

Разработан графический дизайн сайта: определены 4 ключевых элемента дизайна и описаны принципы подбора этих элементов в соответствии с тематикой сайта, целевой аудиторией и фирменным стилем.

Глава 3 Реализация и тестирование

3.1 Выбор и описание программных средств разработки

Выбор инструментальных средств для разработки веб-сайта является одним из ключевых этапов процесса создания информационной системы. Современные технологии предлагают широкий спектр инструментов и платформ для разработки, каждый из которых имеет свои преимущества и особенности. От правильно выбранных технологий зависит производительность, надежность и функциональность будущего сайта. Поэтому необходимо провести тщательный анализ и оценку доступных опций, чтобы выбрать наиболее подходящие инструменты, учитывая специфику задачи, требования заказчика и возможности команды разработчиков.

Разработка любого веб-сайта осуществляется в веб-редакторе, которые позволяют облегчить верстку сайтов, а также автоматизировать этот процесс. На рынке существует множество редакторов, которые предоставляют различный функционал, например, подсветка синтаксиса, автодополнение кода, установка расширений и множество других полезных функциональностей.

Все веб-редакторы можно разделить на две большие категории, а именно визуальные и текстовые. В текстовых редакторах код сайта пишется вручную, а результат виден только если запустить этот код в браузере. Визуальные редакторы, которые называются WYSIWYG (What You See Is What You Get)-редакторами, позволяют составлять страницу из готовых элементов, например, кнопок, блоков и т.п. Редактор сам при этом пишет соответствующий код, а разработчик видит результат работы сразу же. При этом зачастую для разработки сайта в визуальных редакторах не требуются знания HTML, CSS и языков программирования.

WSYSIWYG-редакторы позволяют легко и быстро создать сайт, не обладая для этого особыми навыками, в то время как тестовые редакторы

подходят больше для профессионалов, которые обладают всеми необходимыми знаниями для разработки ПО с самого нуля вручную. Кроме того текстовые редакторы дают разработчику больше свободы и возможностей оптимизация кода.

Visual Studio Code, скриншот которого представлен на рисунке 18, был разработан компанией Microsoft и является не только текстовым редактором кода, но также обладает частью функционала IDE, а также многими другими механизмами, например, автозаполнение, расширения и дополнения, интеграция с Git, встроенные отладчик для кода JavaScript, TypeScript, Node.js и распространяется бесплатно.

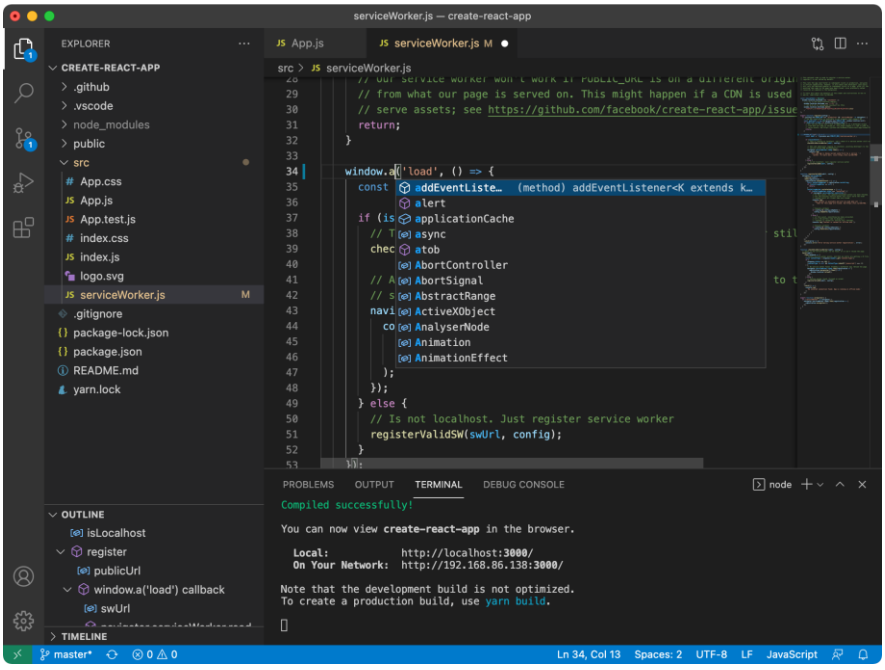

Рисунок 18 – Среда Visual Studio Code

Sublime Text, представленный на рисунке 19, является еще одним популярным редактором для HTML, который распространяется бесплатно, но с ограничениями. Чтобы пользоваться всем функционалом, необходимо приобрести лицензию. Есть возможность установки дополнительных готовых плагинов, а также создания своих собственных.

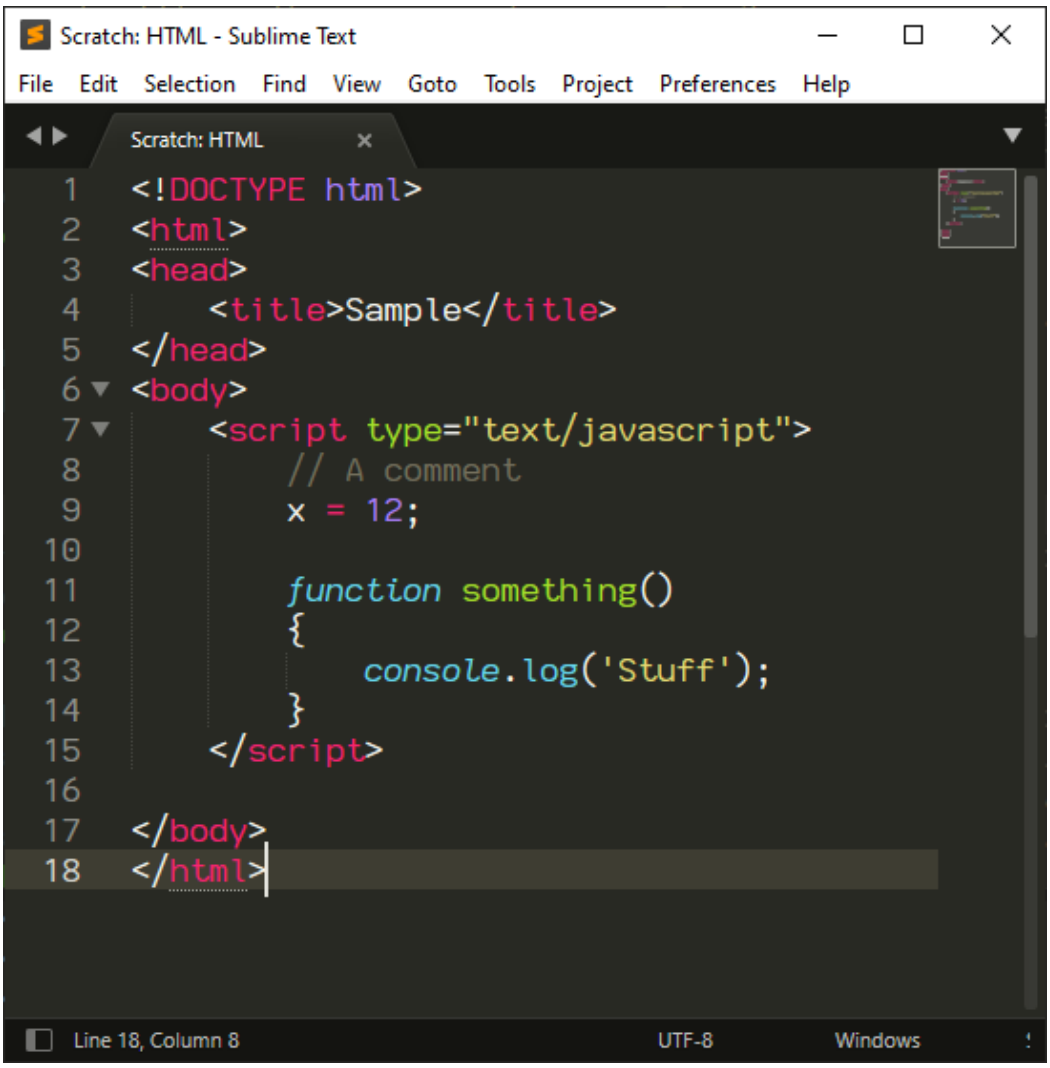

Рисунок 19 – Sublime Text

WebStorm, скриншот которого представлен на рисунке 20, разработан компанией JetBrains и также является популярной средой разработки, имеющей автодополнение кода, проверку на наличие ошибок, удобную отладку кода, интеграцию с GitHub, а также большое количество плагинов.

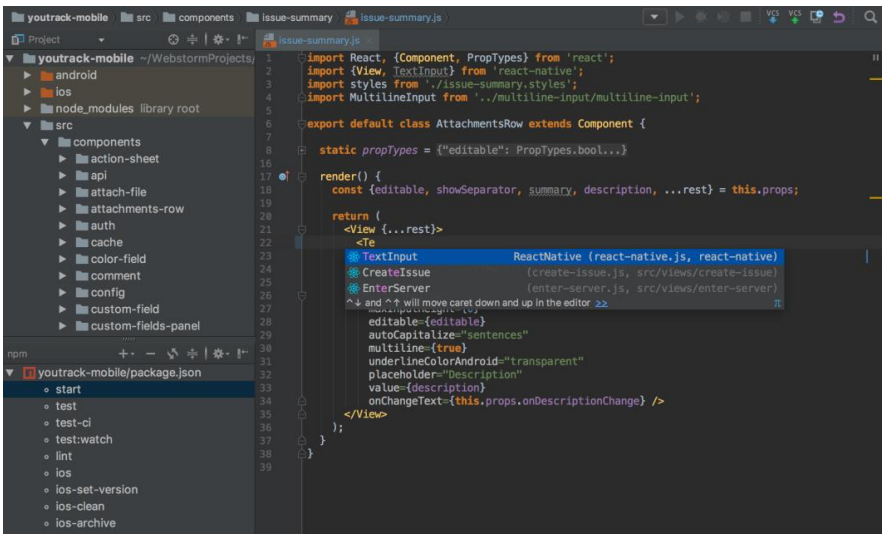

Рисунок 20 – WebStorm

Adobe Dreamweaver CC, представленная на рисунке 21, разработана Adobe Inc и является одним из редакторов, который поддерживает как текстовые, так и WYSIWYG редакторы кода. Программа поддерживает множество языков программирования, а также удобный предпросмотр с возможностью увидеть, как выглядит тэг. Большим недостатком является то, что Dreamweaver распространяется по подписке, цена за которую достаточно высокая.

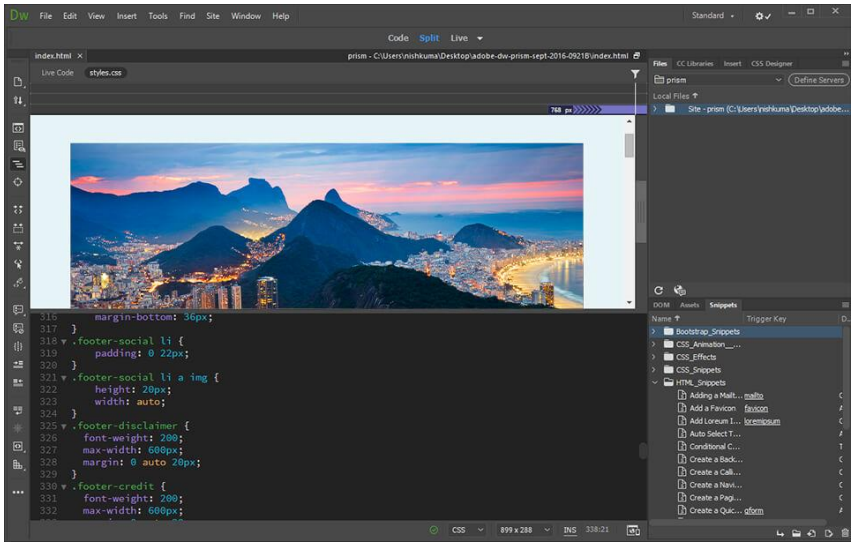

Рисунок 21 – Adobe Dreamweaver CC

Так как мы будем создавать сайт туристического агентства вручную, то у нас нет необходимости в графическом редакторе, поэтому разрабатывать наш проект мы будем в Visual Studio Code, так как этот редактор достаточно прост, распространяется бесплатно и обладает достаточным функционалом, который делает процесс разработки достаточно быстрым и удобным.

3.2 Выбор технологии реализации проекта веб-представительства

Для разработки web-сайтов и приложений существуют множество технологий, которые можно разделить на две большие группы, а именно серверные и интерфейсные технологии. В нашей работе мы рассмотрим интерфейсные технологии.

Интерфейсные технологии, или Front-end, используются для разработки клиентской части сайта, с которой непосредственно взаимодействуют пользователи.

В нашем проекте мы использовали:

- HTML;
- CSS;
- JavaScript;
- Vue.js;
- Vite;
- Quasar;
- Firebase;
- Node.js.

Фундаментальной технологией веб-разработки является язык гипертекстовой разметки HTML, с помощью которого описывается базовая структура страницы. Когда пользователь посещает сайт, то браузер загружает HTML-файл, который описывает структуру и контент страницы [4].

Язык HTML обладает простым синтаксисом и состоит из тегов, являющихся указаниями для браузера, как отображать контент, который помещен в них.

Так как HTML создает только каркас, то сайты, написанные чисто на нем, не обладают необходимыми эстетическими качествами, которые необходимы в современных сайтах. Чтобы придать нашей странице красивый внешний вид, необходимы каскадные таблицы стилей, известные как CSS, которые позволяют отформатировать веб-страницу и сделать ее более привлекательной для пользователя [4].

Несмотря на то, что зная только CSS и HTML, уже можно создавать сайты различной сложности, HTML и CSS не являются языками программирования и поэтому не могут обрабатывать логику, например, отправлять и получать данные с сервера, производить вычисления, а также реагировать на действия пользователя. Чтобы исправить это, на помощь приходят языки программирования. Самым популярным языком программирования, который применяется в веб-разработке, является JavaScript, который позволяет сделать сайт интерактивным [24].

JavaScript является интерпретируемым языком программирования, то есть код, написанный на интерпретируемым языке, не нуждается в компиляции, а сразу выполняется программой-интерпретатором, что ускоряет разработку. Так интерпретатор JavaScript встроен во все современные браузеры, то код может запускаться на любом компьютере [20]. В современных браузерах JavaScript может реагировать на действия пользователя, например, на щелчок мыши или нажатие на клавишу клавиатуры, также может отправлять запросы на сервер или получать данные с сервера, скачивать или загружать файлы, изменять HTML-код или менять стили, сохранять данные пользователя в браузере и многое другое.

С развитием языка JavaScript, появилась необходимость в инструментах, которые повысили бы эффективность разработки, а также помогли управлять сложностью кода. Такие инструменты называются фреймворками, которые

представляют собой набор инструментов, библиотек и функций, которые делают разработку приложений легче и быстрее, предоставляя готовые решения для общих задач. На рынке существует множество JS-фреймворков, каждый из которых обладает своими преимуществами и недостатками. Одним из популярных фреймворков является Vue.js, который привлекает своей простотой, легкостью интеграции и простой синтаксической моделью.

Vue.js основан на реактивном подходе, что позволяет автоматически отслеживать изменения данных и обновлять соответствующие части пользовательского интерфейса. Удобные инструменты фреймворка позволяют легко создавать компоненты для разнообразных задач. Благодаря своему чистому и интуитивно понятному синтаксису Vue.js ускоряет процесс разработки и облегчает поддержку приложений [7].

Основой Vue.js являются понятия компонента и состояния. Каждый компонент это отдельный файл, который имеет расширение .vue и состоит из трех секций: template (представление), script (модель) и style (стили) [7]. На рисунке 22 представлен пример однофайлового компонента:

```
<script setup>
import { ref } from 'vue'
const count = ref(0)
</script>

<template>
  <button @click="count++">Count is: {{ count }}</button>
</template>

<style scoped>
button {
  font-weight: bold;
}
</style>
```

Рисунок 22 – Пример однофайлового компонента

Благодаря декларативной отрисовке, разработчики могут описывать состояние интерфейса в HTML-подобной форме с использованием шаблонов, указывая, как и что должно быть отображено, а Vue автоматически реагирует на изменение данных и таким образом заботится о том, чтобы DOM всегда соответствовал описанию.

Кроме того Vue.js обладает своей экосистемой, которая представлена различными библиотеками и инструментами, дополняющими функционал фреймворка. Приведем некоторые из них:

- Router – отвечает за маршрутизацию между компонентами;
- Pinia – библиотека для управления состоянием;
- Server-renderer – отвечает за серверный рендеринг компонентов в HTML-код с отправкой в браузер;
- Devtools – плагин для отладки в браузере.

Quasar является относительно новым фреймворком, в основе которого лежит Vue.js, позволяет веб-разработчикам быстро создавать адаптивные сайты и приложения во многих вариантах:

- одностраничные приложения (SPA);
- приложения с рендерингом на стороне сервера (SSR);
- прогрессивные веб-приложения (PWA);
- расширения браузера;
- мобильные приложения;
- многоплатформенные настольные приложения [6].

Кроме того Quasar предоставляет современный пользовательский интерфейс для веб-сайтов и приложений сразу из коробки, он легко настраивается и расширяется.

Для начала работы с данными фреймворками, необходимо установить Node.js, который является средой выполнения JavaScript, которая позволяет запускать код на JavaScript на сервере, но в контексте Vue.js он необходим для установки зависимостей и управления пакетами, для чего используется

инструмент npm (Node Package Manager). Кроме того Node.js выполняет сборку и упаковку JavaScript-кода, стилей и других ресурсов в оптимизированные файлы для развертывания на сервере, для чего используются инструменты, например webpack или Vite. Также Node.js используется для запуска локального сервера и автоматической пересборки проекта при внесении изменений в код.

В нашем проекте мы также используем Firebase от компании Google. Эта платформа используется для создания мобильных и веб-приложений и предоставляет большое количество инструментов. Выбор в пользу Firebase был сделан по ряду преимуществ, таких как:

- встроенные инструменты для хранилища файлов;
- возможность отслеживать статистику сайта;
- сбор информации о сбоях;
- быстрая разработка и развертывание приложений;
- встроенные инструменты для базы данных;
- хостинг приложений;
- аутентификация пользователей [5].

Все эти преимущества и инструменты входят в бесплатную версию, которой достаточно для требований нашего сайта. Если появится необходимость в дополнительных платных инструментах, то можно просто расширить функционал, перейдя на платный тарифный план.

3.3 Выбор и описание средств работы с базой данных

В мире документно-ориентированных баз данных выделяются три основных игрока: CouchDB, MongoDB и Firestore. Каждая из них имеет свои особенности и преимущества, подходящие для различных типов приложений.

CouchDB, с его распределенной архитектурой и гибкой схемой данных, отлично подходит для приложений, требующих высокой

доступности и масштабируемости. Он использует формат JSON для хранения данных и позволяет легко добавлять и изменять поля документов.

MongoDB, пожалуй, самая популярная из трех, обеспечивает гибкость, производительность и простоту в использовании. Ее широкое применение и активное сообщество пользователей делают ее привлекательным выбором для различных приложений. MongoDB также поддерживает масштабирование как вертикально, так и горизонтально.

Наконец, Firestore, входящий в экосистему Firebase, предлагает документно-ориентированную модель данных с поддержкой реального времени. Его основное преимущество - надежная синхронизация данных между клиентами и сервером, что делает его идеальным для веб-приложений, где требуется мгновенное обновление данных. Удобный интерфейс Firestore SDK и интеграция с Firebase делают его привлекательным выбором для проектов, особенно если уже используется Firebase [5].

Для проекта турагентства, где важны мгновенные обновления данных и простота использования, Firestore представляется наиболее подходящим вариантом. Его возможности реального времени, простота интеграции с Firebase и автоматическое масштабирование обеспечивают надежное и удобное хранилище данных для туров, клиентов и бронирований.

3.4 Реализация веб-представительства

Определившись с необходимыми инструментами, можно приступить к разработке. После инициализации Vue.js сразу создает базовую структуру, пример которой представлен на рисунке 23.

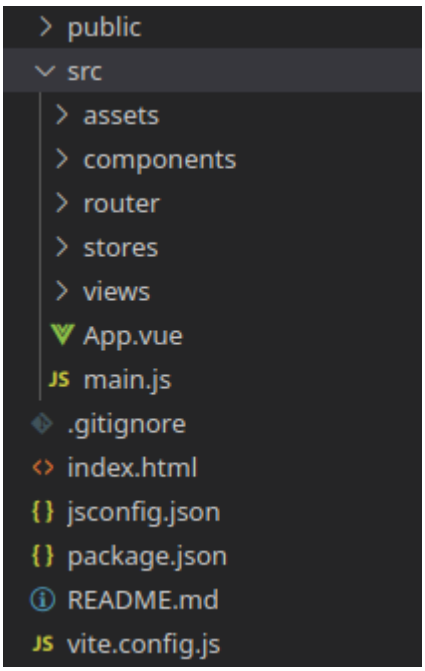

Рисунок 23 - Структура проекта

В папке assets хранятся файлы CSS, шрифты и изображения. В components располагаются элементы, из которых строятся страницы сайты. Разбиение сайта на компоненты удобно тем, что разные страницы могут использовать одни и те же компоненты без изменений или с некоторыми модификациями, поэтому создав компонент единожды, его можно использовать множество раз, что уменьшает количество кода и не усложняет структуру и логику приложения. Папка router содержит в себе настройки роутинга. Stores хранит конфигурацию и данные хранилища, в нашем случае это Pinia. Код самих страниц хранится в папке views.

Файл App.vue представляет собой главный компонент приложения, который содержит в себе HTML-разметку, стили и логику приложения, являясь корневой точкой приложения, в которой объединяются различные компоненты, стили и логика всего приложения.

Main.js – это точка входа приложения, в которой происходит конфигурация и инициализация приложения.

Конфигурационный файл jsconfig.json используется для настройки среды разработки JavaScript, например, настройка пути к модулям или указания корня проекта.

Package.json содержит информацию о зависимостях, скриптах и настройках проектах на Node.js.

Vite.config.js – это конфигурационный файл инструмента сборки Vite, в котором можно настроить процесс сборки, различные аспекты среды разработки, такие как порт сервера разработки, прокси-сервер и т.д. Также здесь можно оптимизировать и минимизировать код и ресурсы для улучшения производительности и уменьшения размера бандла. Кроме того, в этом файле можно подключать плагины и расширения, а также определять пользовательские скрипты сборки, например, для генерации дополнительных ресурсов.

В папке views мы создали файла с расширением *.vue, отвечающие за базовые страницы сайта и представленные на рисунке 24.

Рисунок 24 – Файлы страниц сайта

Каждую страницу мы разбили на компоненты, например, главную страницу, код которой представлен на рисунке 25, мы разбили на 4 блока:

- главный;
- популярные направления;
- преимущества;
- отзывы клиентов.

```vue
<template>
    <main>
        <MainBlock/>
        <DestinationsBlock blockTitle="Исследуйте наши популярные направления" :tours="popularTours"/>
        <AdvantagesBlock/>
        <FeedbacksBlock/>
    </main>
</template>
<script setup>
import MainBlock from "../components/HomeView/MainBlock.vue"
import DestinationsBlock from "../components/HomeView/DestinationsBlock.vue"
import AdvantagesBlock from "../components/HomeView/AdvantagesBlock.vue"
import FeedbacksBlock from "../components/FeedbacksBlock.vue"
import {useStateStore} from "../stores/state"
import { onMounted } from "vue";
import { getTours } from "../api/tour";

const stateStore = useStateStore();
onMounted(()=>{
    stateStore.setBackground("dark")
})
const popularTours = await getTours();
</script>
```

Рисунок 25 – Код главной страницы

Туры мы получаем из базы данных в виде отдельных объектов, хранящихся в массиве. Данные из родительских компонентов дочерним можно передавать через пропсы. Например, так мы делаем со списком туров, которые мы получаем из базы данных при переходе на страницу «Пакеты туров» и передаем через пропсы в DestinationsBlock, код которого представлен на рисунке 26, в котором размещен список с карточками самих туров. В качестве списка мы использовали не стандартный HTML-тэг, а компонент QList, который поставляется Quasar и является удобным, так как нет необходимости с нуля писать стили и функционал. Вместо этого мы можем передать в готовый элемент необходимые данные, такие как стили, текст, изображения, события, а также динамически генерировать список. Через параметр class мы можем передать наши собственные классы, а также использовать уже готовые, поставляемые Quasar, например, text-weight-bold

задает тексту жирное начертание, при этом нам не нужно для этого прописывать отдельный css-стиль.

```
<template>
    <section class="row destinations-block justify-center">
            <h1 class="text-left fit text-grey-14 text-weight-bold text-h4
q-mb-xl">{{blockTitle}}</h1>
            <q-list class="row wrap fit">
                <q-item v-for="tour in tours" :key="tour.id" class="list-
item col-3">
                    <TourCard
                        :tour="tour"/>
                </q-item>
            </q-list>
   </section>
</template>
```

Рисунок 26 – Код блока туров

Использование такого инструмента, как vue-router, избавляет нас от необходимости прописывать функционал навигации вручную, так как он обладает множеством готовых решений:

- вложенные маршруты;
- динамическая маршрутизация;
- эффекты перехода;
- настраиваемое поведение прокрутки и многие другие.

Для наших потребностей достаточно просто создать файл роутера, код которого представлен на рисунке 27, импортировать компоненты сайта, в нашем случае страницы, создать объект роутера и экспортировать его. Весь этот процесс занимает мало времени, а новички могут быстро освоить этот инструмент благодаря понятной документации.

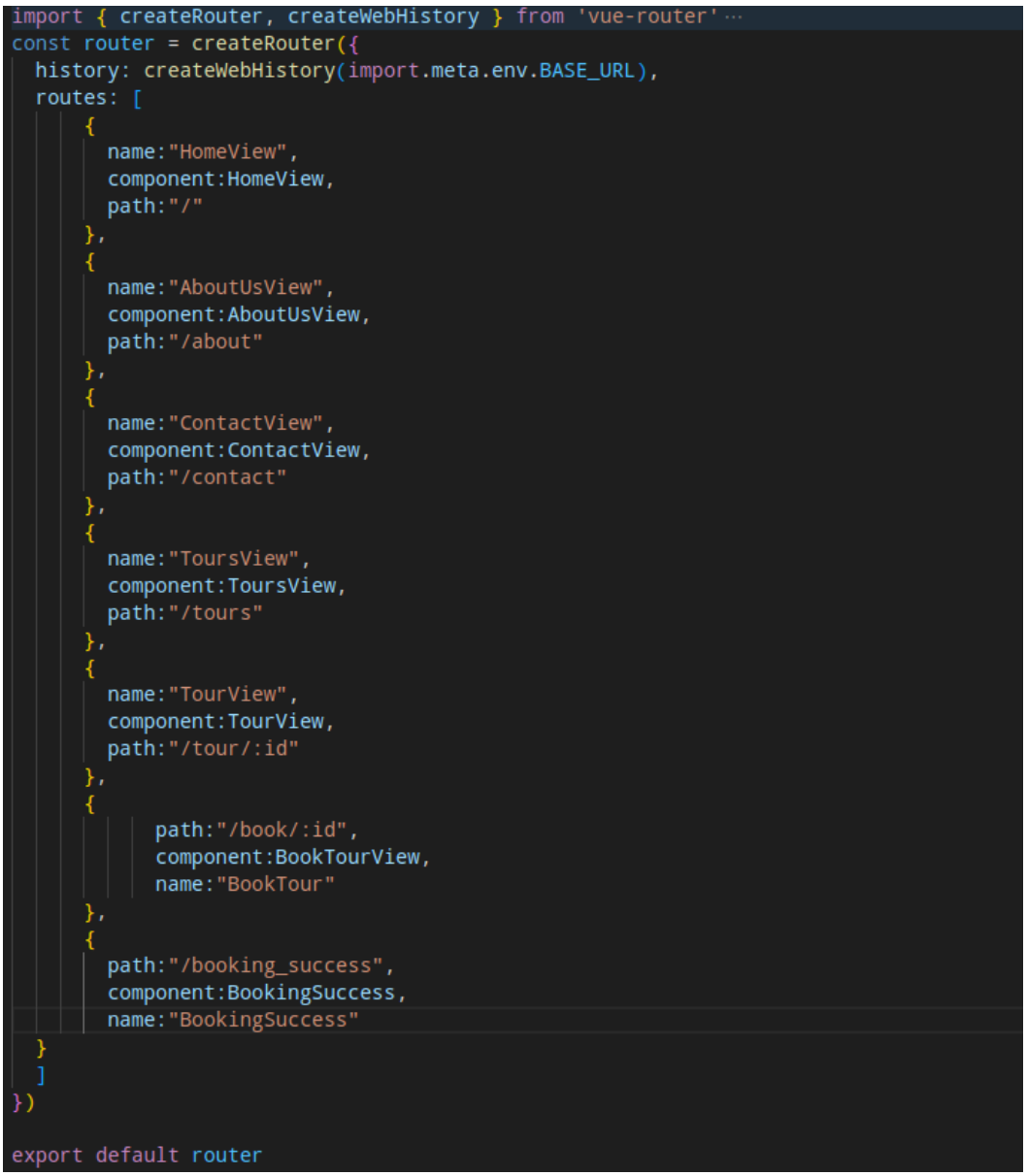

Рисунок 27 - Код роутера

Благодаря набору готовых компонентов Quasar, нам также не пришлось разрабатывать степпер для разбиения процесса бронирования на несколько этапов. Для этого достаточно было использовать QStepper, код которого представлен на рисунке 28 и который предоставляет весь необходимый функционал и набор стилей. Компонент QStepper может иметь один и больше вложенных элементов QStep, который отвечает за отдельный шаг. Достаточно разместить в этом элементе необходимый HTML-код и через атрибуты этого

элемента передать необходимые параметры, такие как название шага, номер шага и т.д.

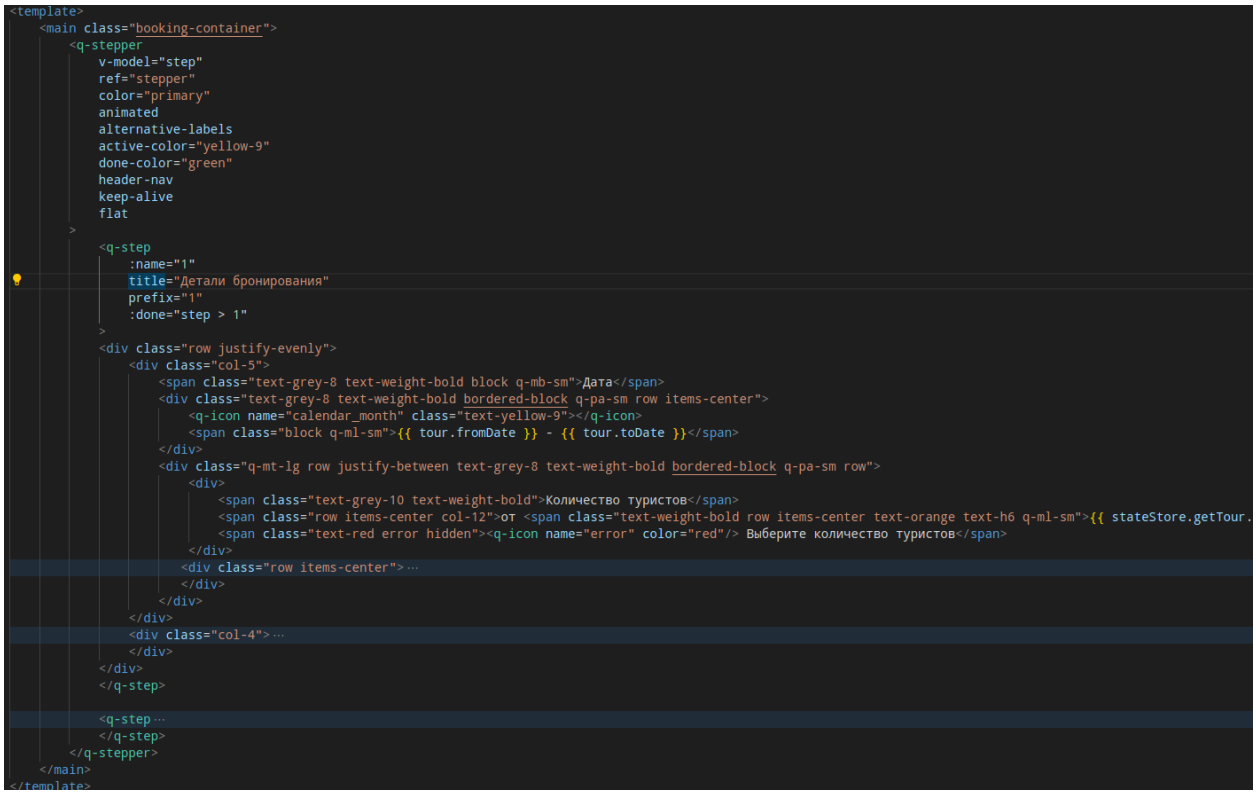

Рисунок 28 – Код степпера бронирования тура

Для работы с базой данных Cloud Firestore мы используем готовые инструменты, которые достаточно импортировать и написать необходимую логику, пример которой представлен на рисунке 29. Например, мы можем получить все туры из базы данных, обратившись к нужной коллекции по ее имени, в результате чего мы получаем все документы этой коллекции, в нашем случае коллекции tours.

```
import { collection, getDocs, getDoc, doc} from "firebase/firestore";
import { ref,getDownloadURL} from "firebase/storage";
import {firestore,storage} from "../firebase"
export async function getTours(){
    const tours = []
    const querySnapshot = await getDocs(collection(firestore, "tours"));
    querySnapshot.forEach( (document)=> {
        tours.push(document.data())
    })
   return tours
}
```

Рисунок 29 – Код обращения к базе данных

На рисунке 30 приведен скриншот документа одного из туров из базы данных.

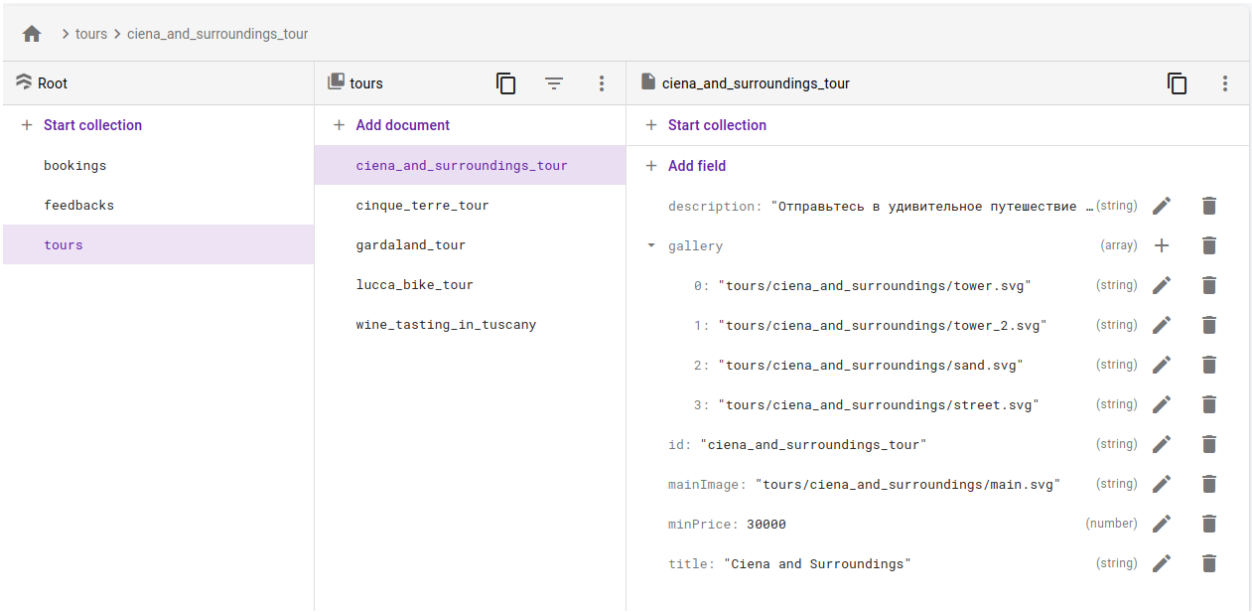

Рисунок 30 – Документ тура в базе данных

В Cloud Storage будем хранить изображения для каждого тура. Для этого мы создали в хранилище папку tours, в которой находятся отдельные папки

для каждого тура, в которые мы загружаем изображения. На рисунке 31 представлен скриншот хранилища файлов.

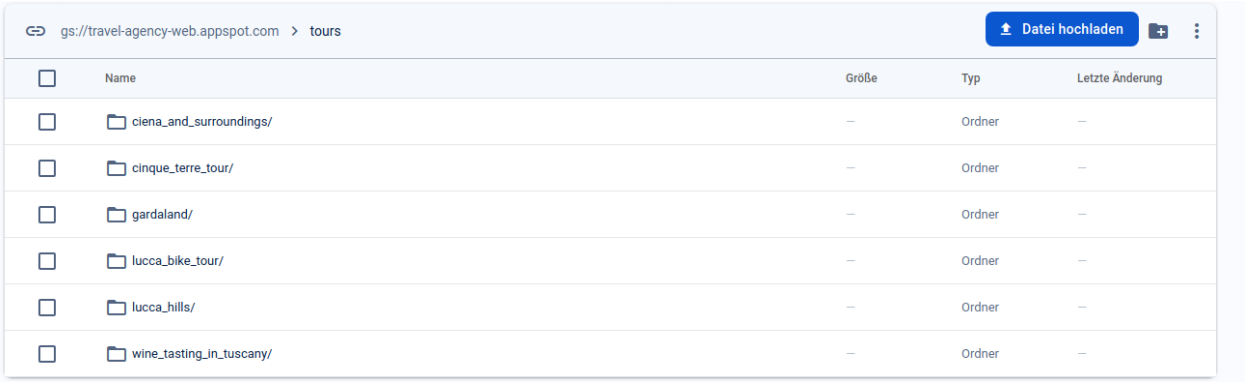

Рисунок 31 - Хранилище файлов

Чтобы преобразовать пути изображений в ссылку для скачивания этого изображения из хранилища, также воспользуемся готовой функцией getDownloadURL(), в результате чего мы получаем ссылку, которую достаточно будет использоваться в качестве источника изображения. На рисунке 32 представлен пример кода для получения ссылки на изображение из хранилища файлов.

```
export async function getURL(str){
    const url =  await getDownloadURL(ref(storage, str))
    return url
}
```

Рисунок 32 – Код для получения изображения из хранилища файлов

Таким образом, используя фреймворки Vue.js и Quasar, а также инструменты, предоставляемые Firebase, нам удалось приступить к разработке, выполнив несколько шагов, не тратя время на дополнительные конфигурации или нахождение решений для наших требований, а готовый

компоненты пользовательского интерфейса и стили, предоставляемых Quasar из коробки, позволили нам сосредоточиться на разработке логики и функционала нашего сайта.

3.5 Публикация и тестирование сайта

Тестирование сайта предназначено для проверки его соответствия заявленным требованиям и характеристикам. От направленности тестирования зависит то, какая особенность сайта подвергается проверке. Весь процесс тестирования документируется и оформляется в виде плана тестирования, в котором описываются стратегия тестирования, средства и методы, а также порядок тестирования.

Задачи тестирования заключаются в том, чтобы подтвердить:

- соответствие сайта функциональным требованиям;
- отсутствие ошибок в коде;
- корректность отображения и функционирования сайта на всех браузерах;
- отсутствие проблем с навигацией по сайту.

Процесс тестирования нашего сайта состоит из нескольких этапов:

- функциональное тестирование, на котором проверяются функции сайта на их соответствие требованиям заказчика;
- конфигурационное тестирование, которое проверяет корректное отображение сайта на различных экранах и браузерах;
- нагрузочное тестирование, заключающееся в проверке уровня критических нагрузок на сервер;
- тестирование пользователем, в процессе которого продукт тестируется конечным пользователем на удобство использования.

Проведя тестирование на локальном сервере, мы провели тестирование на сервере в сети Интернет. Для публикации сайта в сети Интернет,

необходимы домен и сервер. На сервере, или хостинге, хранятся все файлы, необходимые для корректной работы сайта, а домен – это уникальный адрес, по которому происходит обращение к сайту.

В качестве хостинга мы используем Firebase Hosting, который обеспечивает быстрый и безопасный хостинг для веб-приложений.

Ключевыми возможностями являются:

- передача контента через безопасное соединение;
- поддержка всех видов контента для хостинга;
- быстрая доставка контента за счет кэширования файлов на твердотельных накопителях по всему миру, благодаря чему контент доставляется быстро, независимого от того, где находятся пользователи;
- быстрое развертывание приложения на хостинге, используя интерфейс командной строки Firebase [25].

Стоимость хостинга Firebase зависит от:

- объёма дискового пространства, необходимого для хранения содержимого сайта(статических файлов и файлов конфигурации). Бесплатно до 10 Гб;
- объема данных, передаваемых конечным пользователям. Бесплатно до 10 Гб в месяц.

Рекомендуется сразу перейти на план Blaze, так как если мы превысим лимит бесплатного хостинга, то наш сайт не будет отключен. Вместо этого будет автоматически выставлен счет - 0,026 долларов за каждый дополнительный гигабайт хранилища хостинга и 0,15 доллара за каждый дополнительный гигабайт данных, переданных в этом месяце. Таким образом хостинг на Firebase довольно экономный вариант особенно для небольших сайтов с небольшой аудиторией.

Чтобы развернуть наш сайт на хостинге, понадобится пара шагов:

Шаг. 1. Установка интерфейса командной строки Firebase, а также подключение проекта к Firebase, что мы сделали при инициализации и конфигурации проекта;

Шаг. 2. Запуск команды firebase deploy - –only hosting, чтобы загрузить сайт на сервер.

После окончания загрузки, в консоли Firebase мы видим дату и идентификатор последнего релиза, скриншот которого представлен на рисунке 33.

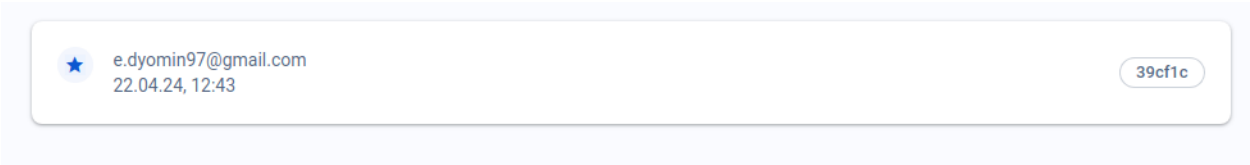

Рисунок 33 – Данные о последнем релизе в консоли Firebase

Далее мы приведем результаты нашей работы. На рисунке 34 представлен скриншот главного блока домашней страницы и меню.

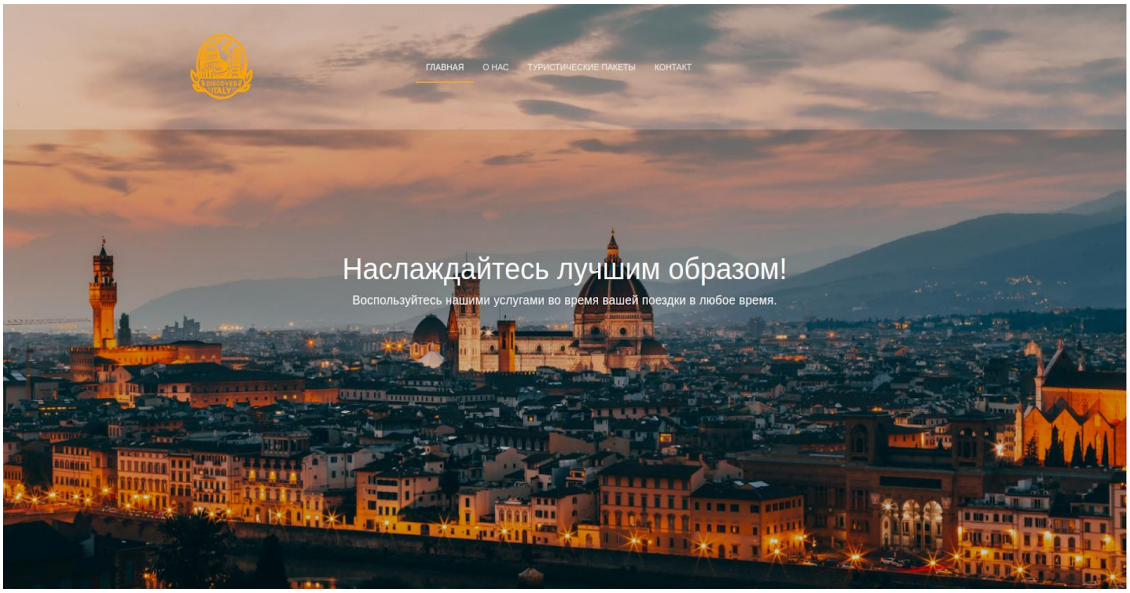

Рисунок 34 – Скриншот главного блока домашней страницы и меню

На главной странице мы также разместили блок с популярными турами, скриншот которого представлен на рисунке 35.

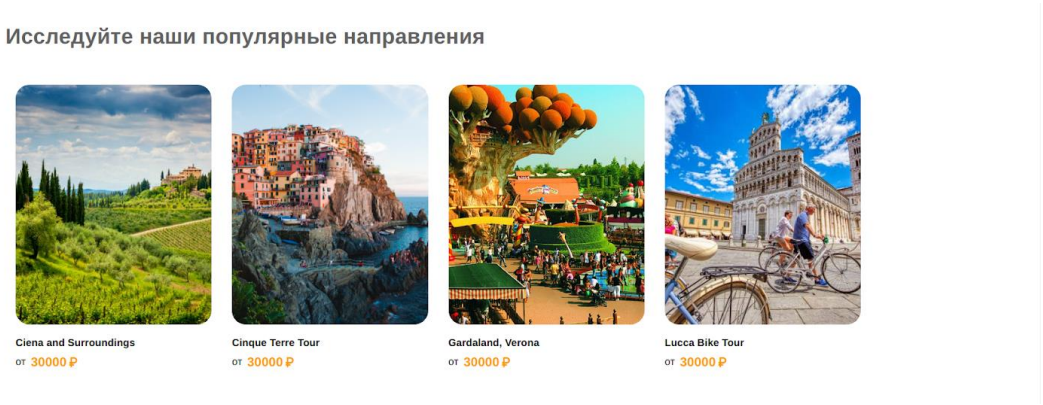

Рисунок 35 – Популярные туры

На рисунке 36 представлен скриншот подвал сайта.

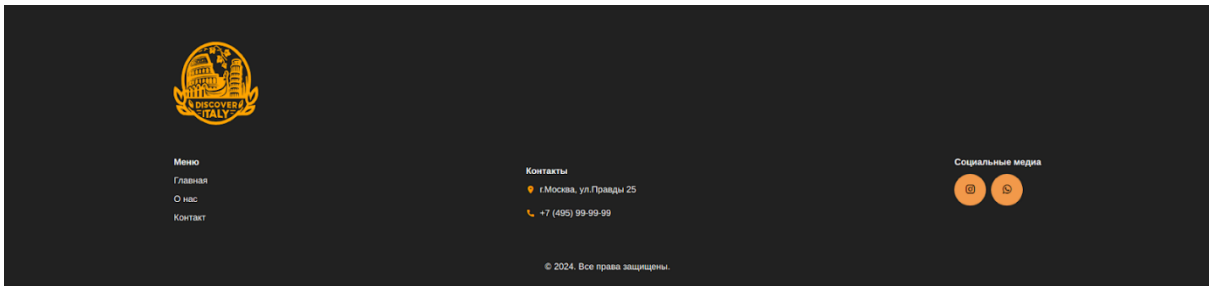

Рисунок 36 – Подвал сайта

На странице «Туристические пакеты» располагается список всех доступных туров, который приведен на рисунке 37.

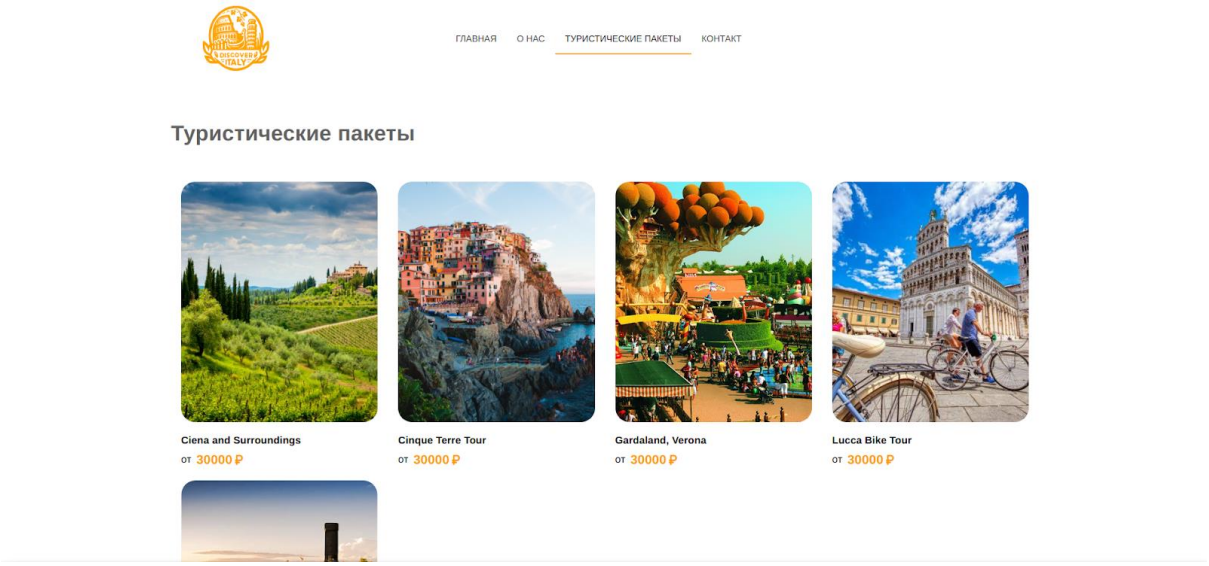

Рисунок 37 – Список доступных туров

При нажатии на любой из туров, пользователь попадает на страницу с детальным описанием тура, а также календарем для выбора даты и галереей.

На рисунке 38 представлен скриншот главного блока детальной страницы тура.

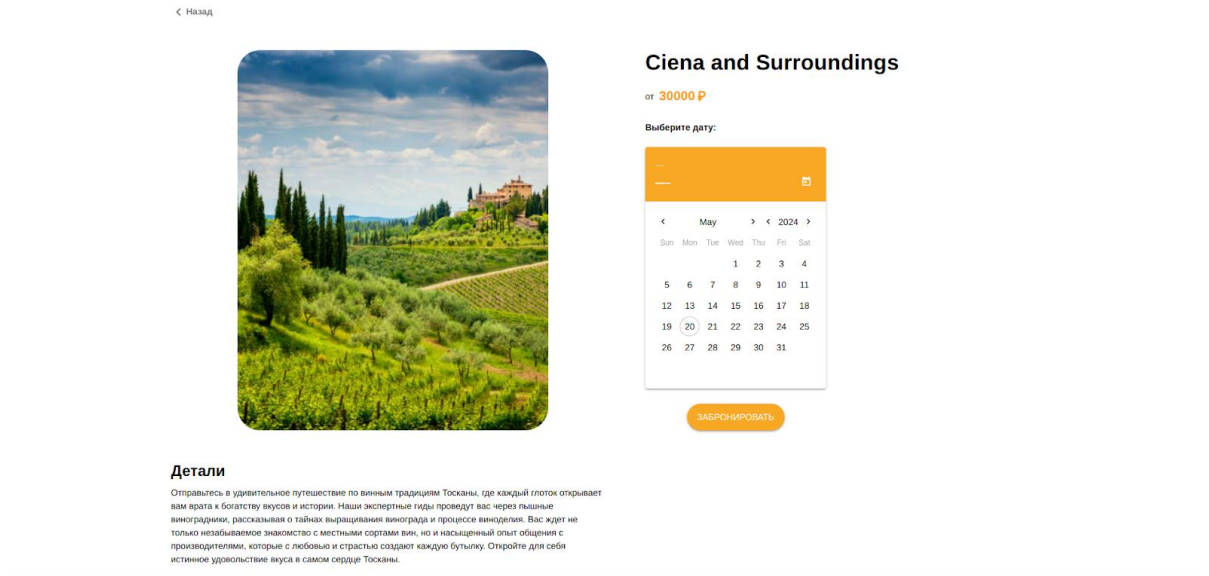

Рисунок 38 – Детальная страница тура

На рисунке 39 представлена галерея изображений тура.

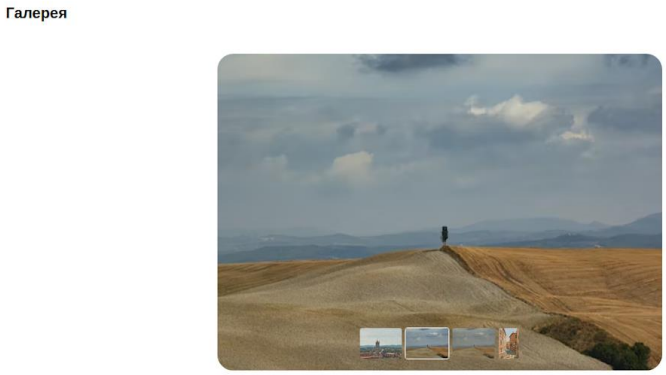

Рисунок 39 – Галерея изображений тура

Во время функционального тестирования мы провели проверку функциональностей веб-представительства на соответствие требованиям и отсутствие ошибок. Приведем пример с функцией бронирования тура. Пользователь не может перейти к бронированию, если он не выбрал даты тура, и в таком случае пользователь увидит ошибку. На рисунке 40 представлена ошибка валидации даты.

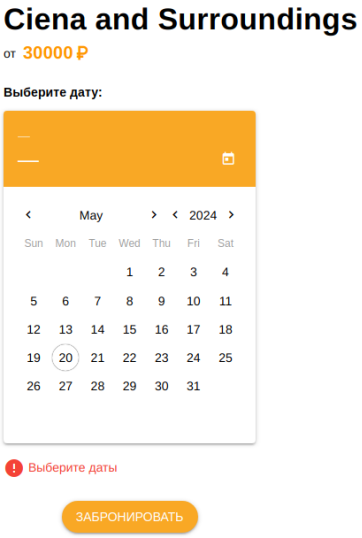

Рисунок 40 – Ошибка валидации даты

Также календарь должен работать корректно, а именно позволить пользователю выбрать дату началу и дату окончания тура. На рисунке 41 представлен скриншот корректной работы календаря.

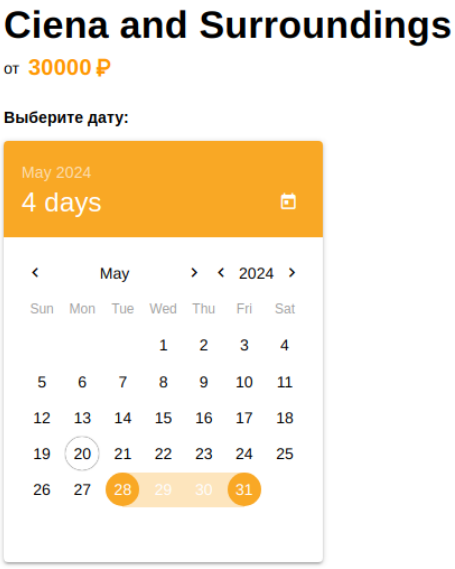

Рисунок 41 – Корректная работа календаря

После выбора дат тура, пользователь будет перенаправлен на страницу бронирования, где на первом шаге необходимо выбрать количество путешествующих. На рисунке 42 представлена форма выбора количества путешествующих.

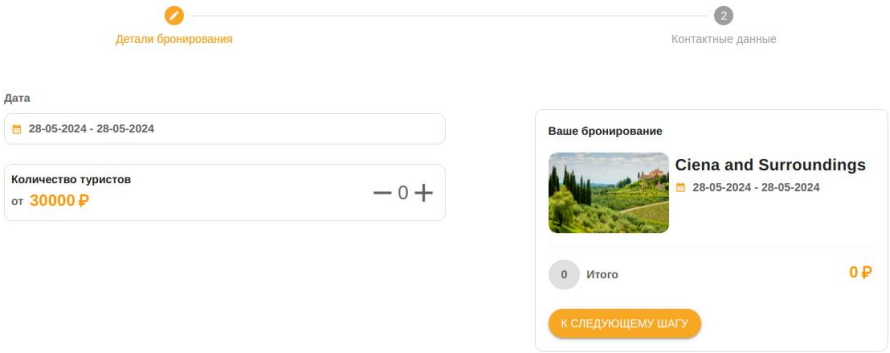

Рисунок 42 – Форма выбора количества путешествующих

При увеличении количества путешествующих в блоке справа подсчитывается итоговая стоимость тура с учетом количества туристов. На рисунке 43 представлен пример подсчитанной итоговой стоимости на одного туриста.

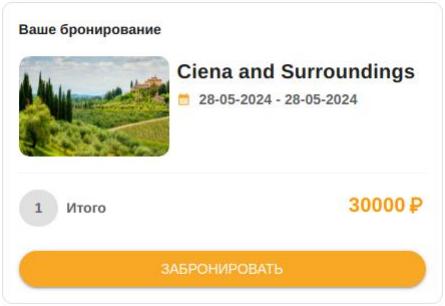

Рисунок 43 – Счетчик итоговой стоимости

При попытке перейти к следующему шагу бронирования, не выбрав количество туриста, то произойдет ошибка валидации формы с просьбой выбрать количетство туристов. На рисунке 44 представлен скриншот ошибки валидации формы выбора количества туристов.

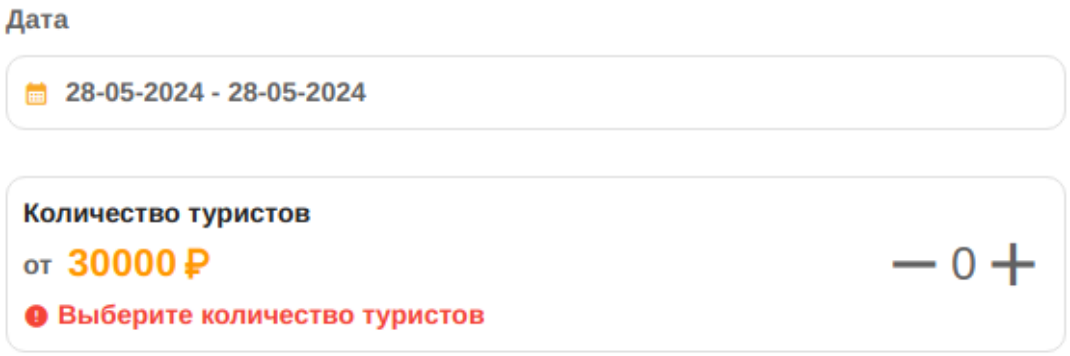

Рисунок 44 – Ошибка валидации формы выбора количества туристов

В противном случае пользователь переходит к следующему шагу, а именно заполнению формы с персональной информацией. На рисунке 45 представлена форма для личных данных.

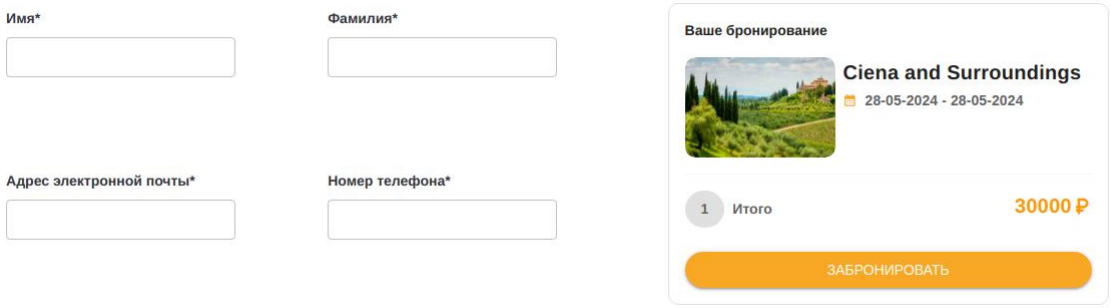

Рисунок 45 – Форма для личных данных

Все поля обязательны для заполнения, поэтому, если пользователь не заполнит поля, он увидит предупреждение о том, что поле обязательно к заполнению. На рисунке 46 представлена ошибка валидации формы для личных данных.

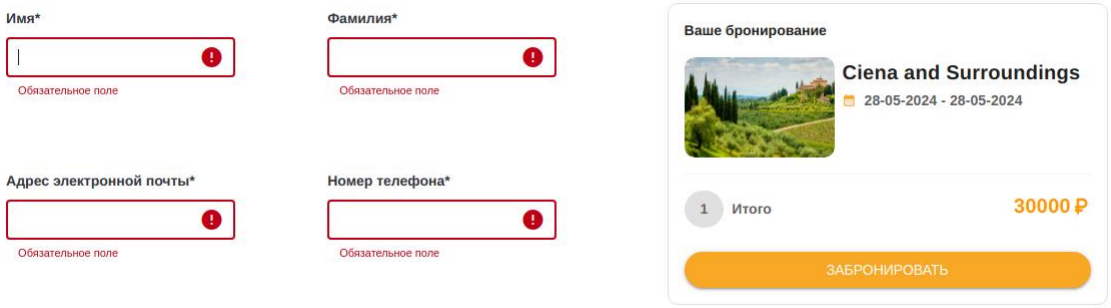

Рисунок 46 – Ошибка валидации формы личных данных

Если пользователь заполнил все верно и нажал кнопку «Забронировать», то появится сообщение об успешном бронировании, скриншот которого представлен на рисунке 47.

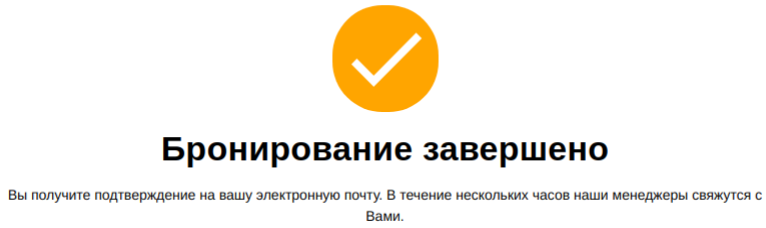

Рисунок 47 – Сообщение об успешном бронировании

Помимо корректной работы функциональностей веб-представительства, нам важно протестировать корректную работу и отображение сайта на экранах мобильных устройств, так как большинство пользователей в наше современное время выходят в сеть Интернет именно с мобильных устройств [22]. Приведем пару примеров того, как выглядит наш сайт на экране мобильного устройства.

На рисунке 48 представлен скриншот списка всех туров.

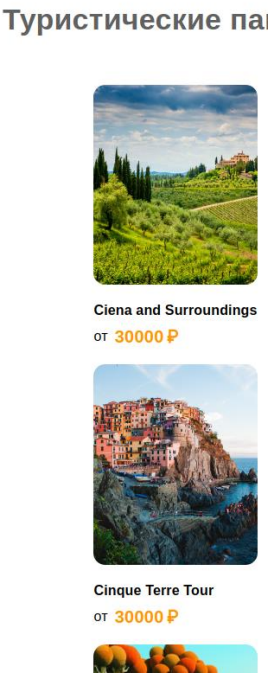

Рисунок 48 – Список туров на экране мобильного устройства

На рисунке 49 представлен скриншот детальной страницы тура с календарем.

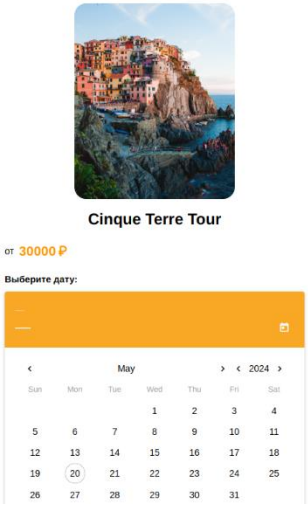

Рисунок 49 – Детальная страница тура на экране мобильного устройства

На рисунке 50 представлен скриншот формы для личной информации пользователя.

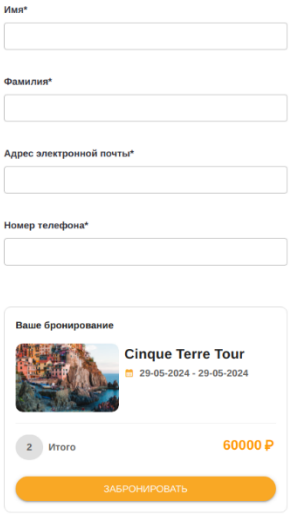

Рисунок 50 – Форма личной информации на экране мобильного устройства

Таким образом, всего за пару шагов нам удалось опубликовать сайт быстро и без дополнительных настроек сервера. Результаты тестирования показали, что функционал сайта соответствует заявленным требованиям, корректно отображается на различных экранах и браузерах, а также выдерживает высокие нагрузки на сервер.

Выводы по главе 3

В этой главе мы рассмотрели реализацию и тестирование нашего веб-приложения. Мы выбрали Visual Studio Code в качестве нашего текстового редактора, потому что он прост, бесплатен и обладает достаточной функциональностью, чтобы сделать процесс разработки быстрым и удобным.

Для разработки веб-представительства мы использовали HTML, CSS, JavaScript, Vue.js, Quasar, Firebase и Node.js.

Также мы протестировали веб-сайт на функциональность, производительность и адаптивность и на разных браузерах и устройствах.

Затем мы развернули веб-сайт на Firebase Hosting.

Таким образом, успешно реализовано и протестировано разработанное веб-приложение.

Приложение соответствует всем требованиям и готово к использованию.

Заключение

В данной бакалаврской работе была разработано веб-представительство для туристического агентства. В ходе работы был проведен анализ предметной области, выявлены потребности целевой аудитории и особенности веб-представительств туристических агентств. На основе полученных данных были разработаны функциональные и нефункциональные требования к системе.

В процессе проектирования было проведено логическое моделирование данных и функциональности системы, разработаны диаграммы прецедентов, деятельности и классов. Также была создана логическая модель данных и разработана внутренняя и внешняя структуры сайта.

Для реализации проекта были выбраны современные и эффективные технологии, такие как Vue.js, Quasar, Firebase и Node.js, а также документно-ориентированная база данных Firestore. Выбранные инструменты позволили реализовать весь необходимый функционал, включая отображение информации о турах, онлайн-бронирование, форму обратной связи, а также адаптивный дизайн сайта для различных устройств.

Тщательное тестирование системы подтвердило ее соответствие заявленным требованиям. Сайт успешно опубликован на Firebase Hosting, обеспечивая быструю и безопасную работу.

Разработанная система веб-представительства является современным и эффективным инструментом для привлечения клиентов, повышения узнаваемости бренда и, как следствие, роста прибыли туристического агентства.

Список используемой литературы

1. Баканов, А. С. Проектирование пользовательского интерфейса: эргономический подход / А. С. Баканов, А. А. Обознов. – 2-е изд. –Москва: Издательство «Институт психологии РАН», 2019. – 184 с.
2. Белов, В. В. Проектирование информационных систем [Текст] / В. В. Белов, В. И. Чистякова – М.: Академия, 2013. – 352 с.
3. Варзунов, А. В. Анализ и управление бизнес-процессами : учебное пособие. Санкт-Петербург : Университет ИТМО. 114 с. [Электронный ресурс]. URL: https://www.iprbookshop.ru/65772.html (дата обращения: 10.05.2024).
4. Дакетт, Д. HTML и CSS. Разработка и дизайн веб-сайтов / Д. Дакетт. - М.: Эксмо, 2018. - 208 с.
5. Документация Firebase [Электронный ресурс] – URL: https://firebase.google.com/docs?hl=en_(дата обращения: 20.05.2024).
6. Документация Quasar [Электронный ресурс] – URL: https://quasar.dev/docs (дата обращения: 20.05.2024).
7. Документация Vue.js [Электронный ресурс] – URL: https://vuejs.org/guide/introduction.html (дата обращения: 20.05.2024).
8. Документация Firebase Hosting [Электронный ресурс] – URL: https://firebase.google.com/docs/hosting?hl=ru (дата обращения: 20.05.2024).
9. ИНТУИТ [Электронный ресурс] – URL: https://intuit.ru/studies/courses/2188/174/lecture/4714?page=2 (дата обращения: 05.05.2024).
10. Котлинский, С. В. Разработка моделей предметной области автоматизации – СПб.: Лань, 2021. – 412 с.
11. Круг С. Не заставляйте меня думать. Веб–юзабилити и здравый смысл. – Эксмо, 2014. 250 с.
12. Лаврищева, Е. М. Программная инженерия. Парадигмы, Технологии и CASE–средства. Учебник для вузов [Текст] / Е. М. Лаврищева М.: Юрайт, 2017. – 280 с.

13. Новиков, Б. А. Основы технологий баз данных. Учебное пособие [Текст] / Б. А. Новиков, Е. А. Горшкова – М.: ДМК Пресс, 2019. – 240 с.

14. Правильная структура сайта: классические составляющие и фишки // Envybox блог. – [Б. м.], 2015–2023. URL: https://envybox.io/blog/strukturasajta/ (дата обращения: 25.04.2024).

15. Советов, Б. Я., Базы данных. Учебник. 3–изд. [Текст] / Б. Я. Советов, В. В. Цехановский, В. Д. Чертовской – М.: Юрайт, 2018. – 420 с.

16. Турагентство DIANA ESPOSITO TRAVEL [Электронный ресурс] – URL: https://www.instagram.com/dianaespositotravel/ (дата обращения: 29.04.2024).

17. Турагентство ItaliaMania [Электронный ресурс] – URL: https://italiamania.ru/ (дата обращения: 29.04.2024).

18. Турагентство Joy Tour [Электронный ресурс] – URL: https://www.instagram.com/joy_tour_/ (дата обращения: 29.04.2024).

19. Флегонтов, А. В. Моделирование информационных систем [Текст] / А. В. Флегонтов, И. Ю. Матюшичев – М.: Лань, 2018. – 112 с.

20. Introduction to Database Systems, 8th Edition. / C.J. Date – Boston : Pearson/Addison Wesley 2005. – 1328 p.

21. Learning PHP, MySQL, JavaScript, CSS & HTML5, Third Edition. / Robin Nixon – O'Reilly Media, 2014 – 729 p.

22. Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics 5th Edition / Jennifer Robbins – O'Reilly Media, 2018. – 811p.

23. Responsive Web Design with HTML5 and CSS3 Illustrated Edition. /Ben Frain – Packt Publishing, 2012. – 324 p.

24. Visual Paradigm [Электронный ресурс]. URL: https://online.visual-paradigm.com/ (дата обращения: 20.05.2024).

25. You Don't Know JS Yet (book series) - 2nd Edition. / Kyle Simpson – GetiPub & Leanpub, 2020 – 145 p.