Содержание

Введение…… 3
1 Анализ предметной области……..... 6
2 Проектирование приложения ……..... 8
3 Разработка программного обеспечения……..……..... 13
3.1 Описание технологического стека разработки ……..... 13
3.2 Описание алгоритма работы ……..... 15
3.3 Описание интерфейса пользователя …… 18
4 Тестирование веб-приложения……..... 21
4.1 План тестирования ……..…….... 21
4.2 Оценка результатов проведения тестирования …… 22
Заключение…… 28
Список использованных источников…….. 30
Приложение А(обязательное) Информационная модель…… 31
Приложение Б(обязательное) Диаграмма прецедентов…… 32
Приложение В(обязательное) Диаграмма последовательности…… 33
Приложение Г(обязательное) Диаграмма классов…… 34

Введение

Интернет и веб-технологии прошли долгий путь с момента их изобретения. От первых электронных писем и простых веб-страниц мы пришли к тому, что имеем сегодня – к сложным веб-приложениям, мобильным приложениям и социальным сетям.

Изначально интернет использовался только для обмена данными между учеными и исследовательскими центрами. С течением времени он стал доступен для всех, и сейчас мы используем его для общения, поиска информации, покупки товаров и многого другого.

Веб-сайты – это ресурсы, расположенные в интернете и доступные для просмотра через браузер. Они могут содержать различные элементы, такие как текст, изображения, видео, аудио и другие. Веб-сайты используются для различных целей, таких как предоставление информации, продажа товаров, общение и другие.

В современном динамичном мире ресторанной индустрии эффективная коммуникация между компаниями-поставщиками (продукты, оборудование, услуги) и ресторанами играет ключевую роль в успехе обеих сторон. Традиционные методы коммуникации, такие как телефонные звонки и электронная почта, часто оказываются недостаточно эффективными, приводя к задержкам, недопониманиям и потере времени. В условиях растущей конкуренции и необходимости оперативного реагирования на изменения рынка, потребность в оптимизированной и прозрачной системе коммуникации становится особенно актуальной. Данная курсовая работа посвящена разработке информационного веб-портала, призванного улучшить взаимодействие между компаниями и ресторанами, обеспечивая своевременный обмен информацией, повышение эффективности сотрудничества и укрепление деловых отношений.

Портал может значительно упростить процесс коммуникации между компанией и ресторанами. Это особенно актуально в условиях, когда скорость и точность передачи информации критически важны для принятия решений.

На сегодняшний день наблюдается значительный рост числа ресторанов и связанных с ними предприятий, что, в свою очередь, стимулирует конкуренцию на рынке. Успех ресторана все больше зависит не только от качества кухни и обслуживания, но и от эффективного управления ресурсами, включая закупку продуктов, оборудования и привлечение новых клиентов. Для компаний в свою очередь, крайне важно поддерживать тесные контакты с ресторанами, обеспечивая своевременную доставку товаров, предоставление качественного сервиса и гибкое реагирование на запросы клиентов. Отсутствие эффективной коммуникации приводит к следующим негативным последствиям:

– потеря времени и ресурсов: запросы и ответы, проходящие через множество посредников, затягивают процесс и увеличивают затраты на коммуникацию;

– недопонимание и ошибки: неясная или неполная информация может привести к ошибкам в заказах, поставках и других аспектах сотрудничества;

– уменьшение прибыли: задержки и недоразумения приводят к снижению эффективности работы ресторанов и уменьшению прибыли как для них самих, так и для компаний-поставщиков;

– ухудшение деловых отношений: негативный опыт коммуникации может нанести вред долгосрочным отношениям между компаниями и ресторанами.

Предлагаемый информационный веб-портал направлен на решение вышеупомянутых проблем, предоставляя комплексную платформу для эффективного взаимодействия между компаниями и ресторанами

Внедрение такого портала позволит компаниям и ресторанам значительно сократить время, затрачиваемое на коммуникацию, снизить вероятность ошибок, повысить эффективность сотрудничества и укрепить деловые отношения. В конечном итоге, это приведет к росту прибыли и конкурентоспособности как ресторанов, так и компаний-поставщиков. Данная курсовая работа представляет собой исследование и разработку такого портала, анализ его потенциальной эффективности и описание архитектуры и функциональных возможностей. В ходе работы будут рассмотрены вопросы выбора технологий, разработки пользовательского интерфейса, обеспечения безопасности и масштабируемости.

Разработка информационного веб-портала как средства оптимизации коммуникации между компаниями и ресторанами – это актуальная задача, направленная на повышение эффективности работы обеих сторон и создание благоприятной среды для развития бизнеса в ресторанной индустрии.

Одним из основных средств создания web-страниц являются HTML-теги. HTML (HyperText Markup Language) – это язык разметки гипертекста, который позволяет структурировать текст и элементы на web-странице. С помощью HTML-тегов можно определить, как будет выглядеть текст, какие элементы будут отображаться, как будут располагаться элементы на странице и т.д.

Объектом исследования является web-программирование.

Предметом исследования является разработка информационного портала для коммуникации между компанией и ресторанами.

Целью данной курсовой работы является разработка концепции информационного портала, который будет служить связующим звеном между компаниями и ресторанами. Такой портал может включать в себя функционал для обмена информацией, упрощения процессов заказа, предоставления актуальных новостей и специальных предложений, а также возможность обратной связи.

Для достижения поставленной цели необходимо решение следующих основных задач:

– проведение анализа исходных данных;

– составить техническое задание на разработку сайта;

– разработать дизайн сайта с учетом удобства навигации, информативности, привлекательности и простоты использования;

– описание хода разработки веб-сайта;

– создание веб-сайта;

- разработка сайта с использованием современных программных средств;
- создание базы данных для сайта;
- проведение тестирование веб-сайта;
- оценить эффективность разработанного сайта.

Выбор темы «Информационный портал для коммуникации между компанией и ресторанами» обусловлен ее практической значимостью и актуальностью в условиях современного рынка. Разработка подобного портала представляет собой сложную, но высоко востребованную задачу, решение которой может существенно улучшить бизнес-процессы в ресторанной индустрии. Данная работа позволит углубить знания в области веб-разработки, проектирования баз данных и управления проектами, а также получить практический опыт в создании реального веб-приложения.

Таким образом, данный проект позволит не только установить эффективные каналы коммуникации между компаниями и ресторанами, но и создать платформу для развития бизнеса, улучшения качества предоставляемых услуг и удовлетворенности клиентов.

1 Анализ предметной области

С развитием технологий и цифровых платформ веб-сайты становятся неотъемлемой частью бизнеса, особенно в сфере общепита. Одной из актуальных задач в данной области является создание эффективного информационного портала, который будет служить связующим звеном между компанией (например, поставщиком продуктов или услуг) и ресторанами. Такой портал должен обеспечивать удобный доступ к информации о новостях, изменениях в правилах и стандартах, а также предоставлять другие важные данные, способствующие успешной интеграции бизнес-процессов.

Предметная область включает двух ключевых участников: компанию и рестораны. Их потребности и цели во взаимодействии формируют основу для функциональных требований портала.

Компания: компания нуждается в эффективном инструменте для распространения информации среди ресторанов, контроля соблюдения стандартов и оперативного реагирования на запросы. Ключевые потребности включают:

– централизованное управление информацией: возможность легко публиковать и обновлять новости, правила, стандарты и другую документацию;

– контроль доступа: управление правами доступа разных пользователей внутри Компании (разделение на администратора, директоров и пользователей);

– эффективная обратная связь: возможность получать вопросы и отзывы от ресторанов, оперативно на них реагировать;

– интеграция с существующими системами[12].

Рестораны: рестораны нуждаются в удобном и оперативном доступе к актуальной информации, необходимой для эффективной работы. Ключевые потребности включают:

– быстрый и легкий доступ к информации: удобный интерфейс для поиска и просмотра новостей, правил, стандартов и другой документации;

– архив документов: доступ к истории публикаций для поиска необходимой информации в любое время;

– простота использования: интуитивно понятный интерфейс, не требующий специальной подготовки.

Предметная область включает следующие ключевые сущности и их взаимосвязи:

– два типа пользователей: администратор (Компания) и пользователи ресторанов;

– новости: текстовые сообщения с возможностью добавления изображений, содержащие актуальную информацию для ресторанов;

– правила: документы, регламентирующие взаимоотношения между Компанией и Ресторанами (например, контрактные условия, требования к качеству);

– стандарты: документы, описывающие требования к качеству продукции, услугам или процессам;

– документы: общий термин, охватывающий новости, правила и стандарты. Может включать различные типы файлов (PDF, DOCX, JPG и т.д.);

– категории: для структурирования новостей, правил и стандартов по тематическим рубрикам;

– роли и права доступа: система управления правами доступа пользователей к различным функциям портала.

Функциональные требования для компании, администратора и директоров ресторанов:

– добавление, редактирование и удаление новостей (с возможностью загрузки изображений);

– добавление, редактирование и удаление правил и стандартов (с возможностью загрузки файлов);

– управление пользователями (добавление, редактирование, удаление, назначение ролей);

– просмотр статистики использования портала (количество посещений, просмотров новостей и т.д.).

Для авторизированных пользователей:

– просмотр новостей, правил и стандартов;

– возможность поиска и фильтрации информации по ключевым словам, дате публикации и типу документа.

Нефункциональные требования:

– надежность: портал должен быть устойчив к отказам и обеспечивать бесперебойную работу;

– производительность: портал должен быстро загружаться и обрабатывать запросы пользователей;

– масштабируемость: портал должен быть способен обрабатывать большое количество пользователей и данных;

– безопасность: портал должен защищать информацию от несанкциониро ванного доступа;

– удобство использования: UI должен быть интуитивно понятным и удобным для пользователей;

– соответствие стандартам: портал должен соответствовать современным стандартам веб-разработки.

2 Проектирование приложения

Разработчик: Исаева Д.С.

Заказчиком является организация ООО «ОренРест», которая занимается деятельностью ресторанов и услуг по доставке продуктов питания.

Почта: daraisaeva713@gmail.com.

Телефон: +79991099762.

Начало работы 16.12.2024, окончание работы 25.01.2025.

Цель работы заключается в разработки веб-портала для эффективной коммуникации между главной компанией и сетью ресторанов, обеспечивающего оперативный обмен информацией о новостях, изменениях правил, стандартов, а также другой важной документацией.

Задачи:

- разработка архитектуры веб-портала, включающей в себя клиентскую и серверную части;
- проектирование базы данных для хранения информации о новостях, правилах, стандартах, ресторанах и пользователях;
- разработка пользовательского интерфейса (UI) для удобного доступа к информации и управления личным кабинетом;
- реализация функционала публикации новостей, правил и стандартов главной компанией;
- реализация функционала доступа к информации для ресторанов, включая поиск, фильтрацию и сортировку;
- реализация системы авторизации и управления доступом пользователей (компания и рестораны);
- тестирование и отладка разработанного портала.

Объектом разработки является процесс коммуникации между компанией-поставщиком и сетью ресторанов.

Предметом разработки является информационный веб-портал, предназначенный для автоматизации и оптимизации этого процесса коммуникации, обеспечивая оперативный обмен новостями, правилами, стандартами и другой важной информацией.

Функциональные требования для администратора:

- добавление новостей. Администратор должен иметь возможность создавать новую новость с указанием заголовка, текста, категории и даты публикации. Для каждой новости предусмотрена функция загрузки изображения. Поддерживаемые форматы должны включать JPEG, PNG и GIF. Администратор должен иметь возможность перемещения новостей в архив, а также полного удаления, с подтверждением действия, чтобы избежать случайного удаления;
- добавление коммуникаций. Возможность создания новых документов, содержащих правила и стандарты с указанием заголовка, текста и даты публикации. Функция загрузки файлов (например, PDF, DOCX) для дополнительных материалов, связанных с правилами и стандартами. Возможность

перемещения документов в архив или полного удаления с подтверждением, чтобы предотвратить случайные действия;

– просмотр обратной связи от пользователей;

– добавление пользователей. Администратор может создать нового пользователя, указав необходимые данные: имя, фамилию, email, пароль и роль (например, пользователь, модератор, редактор).

Для авторизированных пользователей:

– просмотр новостей, правил и стандартов. Пользователи смогут просматривать актуальные новости, а также ознакомиться с действующими правилами и стандартами. Это поможет им быть в курсе важных обновлений и изменений;

– авторизация на сайте;

– обратная связь;

Ниже представлены различные требования к дизайну такого сайта.

Требования к пользовательскому интерфейсу (UI):

– простой и интуитивно понятный интерфейс: дизайн должен быть легким для восприятия, чтобы пользователи могли быстро находить нужную информацию;

– адаптивный дизайн: сайт должен корректно отображаться на различных устройствах (ПК, планшеты, мобильные телефоны);

– чистый и современный дизайн: планируются использовать минималистичные элементы, чтобы не перегружать пользователей;

– логотип и брендинг: наличие яркого логотипа и соблюдение цветовой схемы, соответствующей идентичности компании.

Требования к навигации:

– удобная навигационная панель: ясная и логичная структура меню с выделением главных разделов (Новости, Правила, Стандарты, Контакты и т.д.);

– поиск по сайту: функция поиска по ключевым словам для быстрого доступа к нужной информации.

Требования к контенту:

– актуальная информация: регулярные обновления новостей и материалов;

– подробные разделы: четкое и структурированное представление информации о нововведениях, правилах и стандартах;

– возможность комментирования и обсуждения: опции для пользователей оставлять свои мнения и обмениваться опытом.

Требования к графическим элементам:

– использование изображений и видео: визуальные элементы должны поддерживать текстовой контент и сделать его более интересным;

– иконки и инфографика: они помогут быстро передавать информацию и визуализировать данные[3].

Требование к доступности:

– соблюдение стандартов доступности: этот аспект должен учитывать пользователей с ограниченными возможностями (например, наличие текстов для экранных считывателей);

– контрастность и шрифты: используйте четкие и читаемые шрифты с хорошим контрастом.

Требования к безопасности:

– защита данных: обеспечение конфиденциальности и безопасности пользовательских данных;

– SSL-сертификат: наличие сертификата для защиты данных пользователей.

Требования к SEO-оптимизации:

– адаптация под поисковые системы: оптимизация заголовков, метатегов и контента для повышения видимости в поисковых системах.

Требования к тестированию и обратной связи:

– пользовательское тестирование: регулярное тестирование с реальными пользователями для выявления проблем и улучшения UX;

– сбор обратной связи: опции для пользователей оставлять отзывы о сайте и его функционале[2].

Технические требования:

– серверная часть: Node.js;
– база данных: SQLite;
– фронтенд: HTML, CSS, JavaScript;
– система контроля версий: Git.

Ожидаемый результат:

Работоспособный веб-портал, соответствующий всем функциональным и нефункциональным требованиям, с полной документацией и отчетом о проделанной работе. В отчете должна быть описана архитектура приложения, диаграммы баз данных, описание использованных технологий, результаты тестирования и рекомендации по дальнейшему развитию портала.

Требования к техническому обеспечению:

– сеть с подключение к Интернет процессор Intel Xeon E3 шестого поколение или новее, или AMD Ryzen 1950x или новее; 32 Gb и более оперативной памяти;
– 2 Tb – жесткий диск SSD или HDD с поддержкой SATA 3;
– монитор – FullHD или 4K;
– клавиатура;
– манипулятор типа «мышь».

Требования, предъявляемые к конфигурации клиентских станций:

– процессор, с тактовой частотой не менее 2,4 GHz, не менее 2 ядер;
– 8 Gb оперативной памяти;
– монитор – FullHD;
– клавиатура;
– манипулятор типа «мышь»;
– доступ к подключению – Интернет со скоростью не менее 100 МБ/с.

Требования к программному обеспечению.

На рабочей станции пользователя необходимо установить:

– операционная система: Windows версии не ниже 8.1;

– доступ к широполосному подключению к сети Интернет;
– браузер для просмотра веб-страниц последней версии.

Общие требования к контенту сайта

– высококачественные изображения и видео;
– уникальный и информативный контент;
– правильная грамматика и орфография;
– оптимизация для поисковых систем (SEO);
– регулярное обновление для обеспечения актуальности и новизны.

Требования к сопровождению:

– регулярные обновления программного обеспечения: обновление программного обеспечения сайта для обеспечения безопасности, производительности и функциональности;
– усовершенствование функциональности: изучение новых тенденций и технологий веб-разработки и внедрение улучшений для повышения пользовательского опыта и функциональности сайта;
– улучшение производительности: отслеживание производительности сайта и внедрение оптимизации для повышения скорости загрузки страниц и общей отзывчивости;
– адаптация к новым устройствам и технологиям: проверка совместимости сайта с новыми устройствами и технологиями, такими как мобильные устройства и голосовые помощники;
– постоянная поддержка и обслуживание: обеспечение постоянной поддержки и обслуживания сайта, чтобы он был постоянно доступным и работал без сбоев[2].

В процессе разработки программного обеспечения ключевое значение имеет этап проектирования, на котором определяется архитектура, функциональные и нефункциональные требования к создаваемому приложению. Грамотно выполненное проектирование заложит прочный фундамент для дальнейшего этапа реализации и обеспечит соответствие продукта ожиданиям пользователей и бизнес-требованиям.

Важным аспектом является описание функциональных требований, которые можно визуализировать с помощью диаграммы прецедентов (use case diagram). Эта диаграмма позволяет наглядно представить взаимодействие пользователей с системой и выявить ключевые сценарии использования приложения (Приложение Б). Прецеденты используются для описания конкретных функций или задач, которые система должна выполнять. Они представляют собой сценарии взаимодействия акторов с приложением и помогают понять, какие функции являются критически важными для пользователей[10].

Нефункциональные требования:

– надежность: портал должен быть способен выдерживать различные виды сбоев, таких как отключение сервера или проблемы с сетью, без потери доступности для пользователей. Важно реализовать механизмы резервного копирования и восстановления данных, а также обеспечить безотказную работу

ключевых функций. Проведение регулярного мониторинга состояния системы для быстрого реагирования на возможные проблемы;

– производительность: время загрузки страниц должно составлять не более 2-3 секунд для оптимального пользовательского опыта. Портал должен эффективно обрабатывать запрашиваемые данные, минимизируя задержки и время отклика на действия пользователя. Наличие механизма кэширования для ускорения загрузки и обработки часто запрашиваемой информации;

– масштабируемость: портал должен быть способен обработать увеличенное количество пользователей и транзакций без ухудшения производительности. Необходимо предусмотреть возможность расширения инфраструктуры (например, добавление серверов или увеличение ресурсов) при росте нагрузки. Изоляция компонентов системы для упрощения обновлений и интеграции новых функций;

– безопасность: все данные, передаваемые между пользователями и порталом, должны быть защищены с помощью шифрования (например, использование HTTPS). Реализация многоуровневой аутентификации и авторизации для защиты от несанкционированного доступа. Проведение регулярных аудитов безопасности и обновления системы для защиты от новых уязвимостей;

– удобство использования: интерфейс должен быть интуитивно понятным, позволяя пользователям легко находить необходимую информацию и выполнять действия. Должны быть предусмотрены адаптивные элементы интерфейса для обеспечения возможности работы с порталом на различных устройствах (мобильных, планшетах, ПК). Регулярные пользовательские тестирования для выявления и устранения проблем с удобством использования интерфейса;

– соответствие стандартам: портал должен быть разработан с соблюдением современных стандартов веб-разработки, включая семантическую разметку, доступность (например, WCAG), и лучшие практики безопасности. Необходима кроссбраузерная совместимость, чтобы все пользователи могли получить доступ к функциональности портала независимо от используемого браузера. Следует учитывать требования к локализации и интернационализации, чтобы обеспечить поддержку различных языков и культур.

3 Разработка программного обеспечения

3.1 Описание технологического стека разработки

Технологический стек разработки, состоит из HTML, CSS, JavaScript, SQLite, VisualStudioCode и Figma. Они представляет собой мощный набор инструментов для создания веб-приложений.

HTML — это стандартный язык разметки для создания веб-страниц. Он позволяет структурировать содержание, используя теги для определения заголовков, параграфов, списков, изображений, ссылок и других элементов. HTML — это основа любого веб-документа и обеспечивает семантическую структуру.

Преимущества HTML:

– структурирование контента: HTML позволяет организовать текст, изображения, видео и другие элементы на веб-странице, создавая логическую структуру;

– простота изучения: HTML обладает простой и понятной синтаксической структурой, что делает его доступным для новичков;

– совместимость: HTML поддерживается всеми современными веб-браузерами и устройствами, что обеспечивает высокую степень совместимости.

– доступность: HTML позволяет создавать доступные веб-страницы, которые легко читаются программами для людей с ограниченными возможностями;

– интерактивность: HTML поддерживает использование различных типов медиа и форм, что делает страницу интерактивной.

CSS — это язык стилей, который используется для оформления и визуального представления HTML-документов. С помощью CSS можно задавать цвета, шрифты, отступы, размеры и многие другие стилистические параметры. CSS также позволяет адаптировать дизайн под различные устройства и экраны, используя медиа-запросы.

Преимущества CSS:

– разделение содержания и оформления: CSS позволяет отделить внешний вид веб-страницы от её структуры, что упрощает обновление и поддержку сайта;

– управление стилями: с помощью CSS можно легко изменять шрифты, цвета, отступы, размеры и другие стили элементов на странице;

– адаптивный дизайн: CSS позволяет создавать адаптивные и отзывчивые дизайны, что обеспечивает корректное отображение сайта на разных устройствах (мобильных, планшетах, десктопах);

– кросс-браузерная совместимость: CSS стили хорошо поддерживаются в большинстве современных браузеров, что позволяет создавать единый стиль на разных платформах;

– переиспользование стилей: CSS дает возможность создавать стили один раз и применять их к нескольким элементам или страницам, что сокращает объем кода и упрощает его поддержку;

– анимации и переходы: CSS поддерживает создание анимаций и эффектов перехода, что может значительно улучшить пользовательский интерфейс и восприятие сайта.

SQLite — это легковесная СУБД (система управления базами данных), которая используется для хранения данных в виде файлов. Она не требует установки на сервер и подходит для разработки небольших приложений и прототипов. SQLite идеально вписывается в проекты, где необходима простая и быстрая работа с данными без сложной архитектуры.

Преимущества SQLite:

– легковесность: SQLite – это встроенная база данных, которая не требует отдельного серверного процесса. Вся база данных хранится в одном файле, что делает ее лёгкой в развертывании и использовании;

– простота использования: SQLite имеет простой и понятный интерфейс. Пользователи могут начать работать с ней без необходимости в сложной настройке;

– поддержка ACID: SQLite поддерживает свойства ACID (Атомарность, Согласованность, Изолированность, Долговечность), что гарантирует надежную работу с данными;

– кросс-платформенность: SQLite доступна на различных платформах и может работать на любом устройстве, которое поддерживает файловую систему;

– отсутствие необходимости в установке: так как SQLite является встроенной БД, нет необходимости в установке и управлении сервером баз данных;

– хорошая производительность: для небольших и средних приложений SQLite обеспечивает хорошую производительность при обработке запросов;

– поддержка SQL: SQLite использует стандартный язык SQL для взаимодействия с данными, что облегчает переход от других СУБД.

JavaScript — это язык программирования, который позволяет добавлять интерактивность на веб-страницы. С его помощью можно обрабатывать события, управлять элементами DOM (Document Object Model), делать асинхронные запросы к серверу (например, через AJAX) и многое другое. JavaScript широко используется для создания динамических и отзывчивых интерфейсов[13].

Преимущества JavaScript:

– кросс-платформенность: JavaScript работает на всех современных браузерах и может быть использован как на клиентской, так и на серверной стороне (с помощью Node.js);

– асинхронное программирование: возможность выполнения асинхронных операций позволяет улучшать отзывчивость приложений, что делает его идеальным для разработки веб-приложений;

– обширная экосистема: существует множество библиотек и фреймворков, таких как React, Angular и Vue.js, которые упрощают разработку сложных интерфейсов;

– удобство работы с данными: JavaScript хорошо подходит для работы с JSON, что упрощает взаимодействие с RESTful API и другими формами обмена данными;

– динамическая типизация: JavaScript использует динамическую типизацию, что позволяет разрабатывать код быстрее, хотя и требует внимательности к типам.

– поддержка событий: JavaScript отлично справляется с обработкой событий, что позволяет создавать интерактивные и отзывчивые пользовательские интерфейсы;

– сообщество и поддержка: огромное сообщество разработчиков и регулярные обновления означают, что у вас всегда будет доступ к последним технологиям и поддержке.

VS Code — это популярный редактор кода, который поддерживает множество языков программирования и предоставляет многофункциональные инструменты для разработчиков. Он включает в себя поддержку расширений, подсветку синтаксиса, автозаполнение, отладку и интеграцию с системами контроля версий, такими как Git. VS Code является удобной средой для написания кода на HTML, CSS и JavaScript.

Figma — это облачный инструмент для дизайна интерфейсов и прототипирования. С его помощью дизайнеры могут создавать макеты веб-страниц, мобильных приложений и других продуктов. Figma позволяет командам совместно работать над проектами в реальном времени и предоставляет мощные инструменты для создания интерактивных прототипов, что облегчает процесс разработки[11].

Совместное использование этих технологий позволяет создать полноценное веб-приложение:

– HTML используется для создания структуры страницы;

– CSS оформляет страницу, обеспечивая привлекательный вид;

– JavaScript добавляет интерактивность, позволяя пользователям взаимодействовать с приложением;

– SQLite обеспечивает хранение данных, которые могут быть использованы в приложении;

– VS Code служит инструментом для разработки и написания кода;

– Figma помогает в проектировании пользовательского интерфейса и создании прототипов.

3.2 Описание алгоритма работы

Разработка программного обеспечения (ПО) — это процесс создания и поддержки программных приложений, который включает несколько этапов. Один из основных аспектов разработки ПО — это грамотное проектирование алгоритмов.

Алгоритм работы для информационного портала коммуникации между сотрудниками и ресторанами может быть следующим:

– пользователь авторизуется на сайте и попадает на главную страницу;

– пользователь может просматривать новости как своей так и другой компании;

– пользователь может просматривать общие коммуникации для всех ресторанов;

– пользователь может посмотреть доступные или прошедшие мероприятия;

– пользователь сам может заполнить форму и предложить новость.

Хранение данных в приложении будет реализовано с использованием базы данных для хранения информации о мероприятиях, пользователях, коммуникации, новостей. Для этого планируется использовать реляционную базу данных – SQLite.

Так как на сайте и на странице администратора представлены формы для отправки данных в бд, они реализованы следующим образом:

– создание пользовательского интерфейса: на первой стадии мы разработали HTML-форму, которая содержит поля для ввода данных и кнопку для отправки данных в бд;

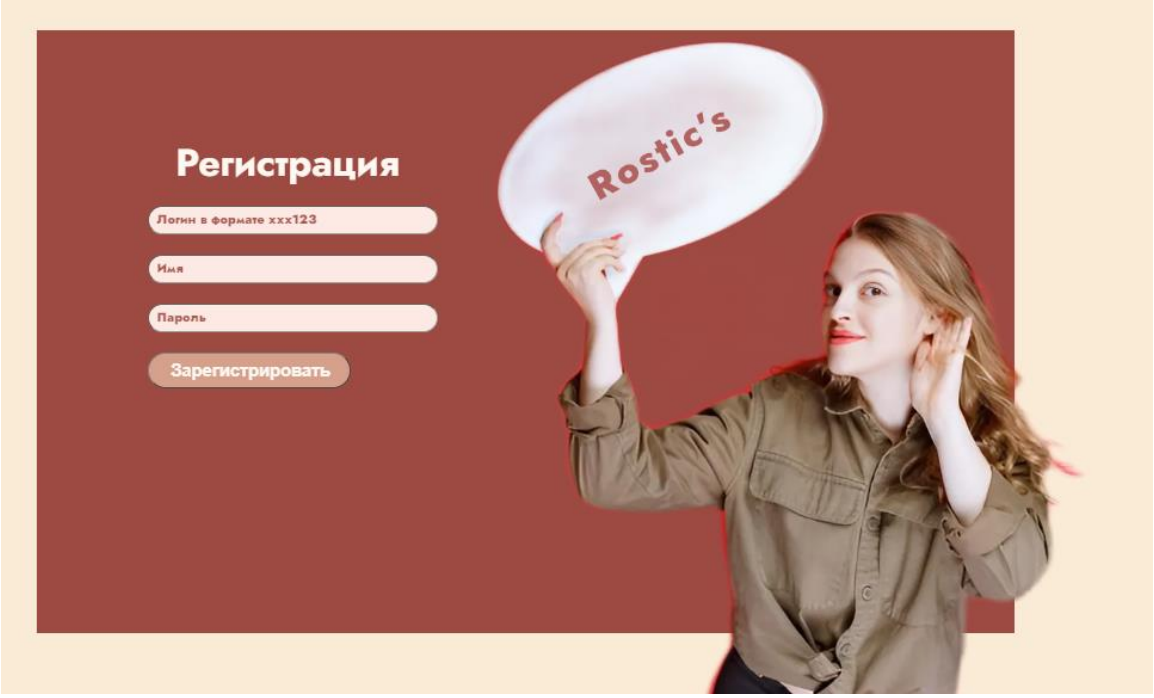

Рисунок 1 – Форма для добавление пользователей

– сбор данных из формы: после того как пользователь заполнит форму и нажмет кнопку отправки, JavaScript используется для захвата данных, введенных в полях формы[14]. Этот процесс включает в себя проверку корректности данных перед отправкой;

– отправка данных на сервер: для передачи собранных данных на сервер используется AJAX-запрос (обычно через fetch или XMLHttpRequest). Данные отправляются в формате JSON, который удобен для обработки на сервере;

– обработка данных на сервере: на серверной стороне создается обработчик, который принимает данные от клиента. Этот обработчик может быть реализован на Node.js[15], Python или другом языке программирования, поддерживающем работу с SQLite. Здесь осуществляется подключение к базе данных и выполнение SQL-команды для вставки полученных данных в соответствующие таблицы;

– работа с базой данных SQLite: после выполнения SQL-команды данные сохраняются в базе данных SQLite. Обработчик возвращает ответ клиенту, подтверждая успешное добавление данных или сообщая об ошибке;

```
const express = require("express");
const bodyParser = require("body-parser");
const session = require("express-session");
const { Sequelize, DataTypes } = require("sequelize");
const bcrypt = require("bcrypt");
const path = require("path");

const app = express();
const port = 3000;

// Настройка базы данных SQLite и подключенние к ней
const sequelize = new Sequelize({
  dialect: "sqlite",
  storage: "db.sqlite",
});
```

Рисунок 2 – Подключение к БД

– получение и отображение данных для администратора: для вывода данных из базы данных на страницу администратора создается еще один AJAX-запрос, который запрашивает данные из таблиц[6]. Сервер обрабатывает этот запрос, извлекает нужные данные из базы и отправляет их обратно клиенту;

```
    <script>
        fetch('/users')
            .then(response => response.json())
            .then(users => {
                const tbody = document.querySelector('#usersTable tbody');
                users.forEach(user => {
                    const row = document.createElement('tr');
                    row.innerHTML = `
                        <td>${user.id}</td>
                        <td>${user.login}</td>
                        <td>${user.name}</td>
                        <td>${user.password}</td>
                        <td>${user.roleId}</td>
                    `;
                    tbody.appendChild(row);
                });
            })
            .catch(error => console.error('Ошибка при загрузке пользователей:', error));
    </script>
</body>
</html>
```

Рисунок 3 – Код отображения данных на странице

– отрисовка данных на странице (рисунок 4): полученные данные отображаются на странице администратора с помощью JavaScript. Это может быть сделано путем динамического создания HTML-элементов или использования библиотек для работы с таблицами (например, DataTables), что позволяет отображать и форматировать данные в удобном для восприятия виде.

Рисунок 4 – Отображение таблицы на странице администратора

Диаграмма классов (Приложение Г) представляет собой статическую модель проектируемого информационного портала для коммуникации между компанией-поставщиком и сетью ресторанов. Она визуализирует структуру системы, отображая ключевые классы, их атрибуты, методы и взаимосвязи. Диаграмма служит основой для дальнейшей разработки программного обеспечения, обеспечивая ясное понимание архитектуры и структуры данных. В ней представлены основные сущности предметной области и их взаимоотношения, что позволяет оценить целостность и корректность модели.

3.3 Описание интерфейса пользователя

Приложение предоставляет авторизованным пользователям доступ к новостям, важной информации (коммуникациям), форме обратной связи, расписанию мероприятий, информации о компании и её контактным данным.

Первое что видит пользователь заходя на сайт, это страница авторизации, где пользователь должен ввести свои данные, которые предоставляет компания для входа на сайт.

Форма авторизации представляет поле «Логин» – текстовое поле для ввода личного неповторяющегося имени пользователя. Валидация на стороне клиента: обязательное поле. Поле «Имя» – текстовое поле для ввода обычного имени пользователя. Поле «Пароль» – поле для ввода пароля. Тип поля: password для скрытия вводимых символов. Валидация на стороне клиента: обязательное поле. Кнопка «Войти» Активируется после заполнения обоих полей. При нажатии отправляет данные формы на сервер для проверки.

После авторизации пользователь попадает на главную страницу, на которой представлены: шапка сайта (рисунок 5) – логотип компании: при клике перенаправляет на главную страницу; ссылки на остальные страницы сайта ;кнопка «Выход»: расположена справа. При нажатии завершает сеанс пользователя и перенаправляет на страницу авторизации.

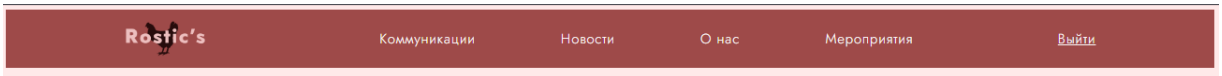

Рисунок 5 – Шапка сайта

Основной контент главной страницы это главная новость или коммуникация компании (рисунок 6). Так же на нем присутствует кнопка «перейти к ценностям», которая перенаправит пользователя на страницу самой новости.

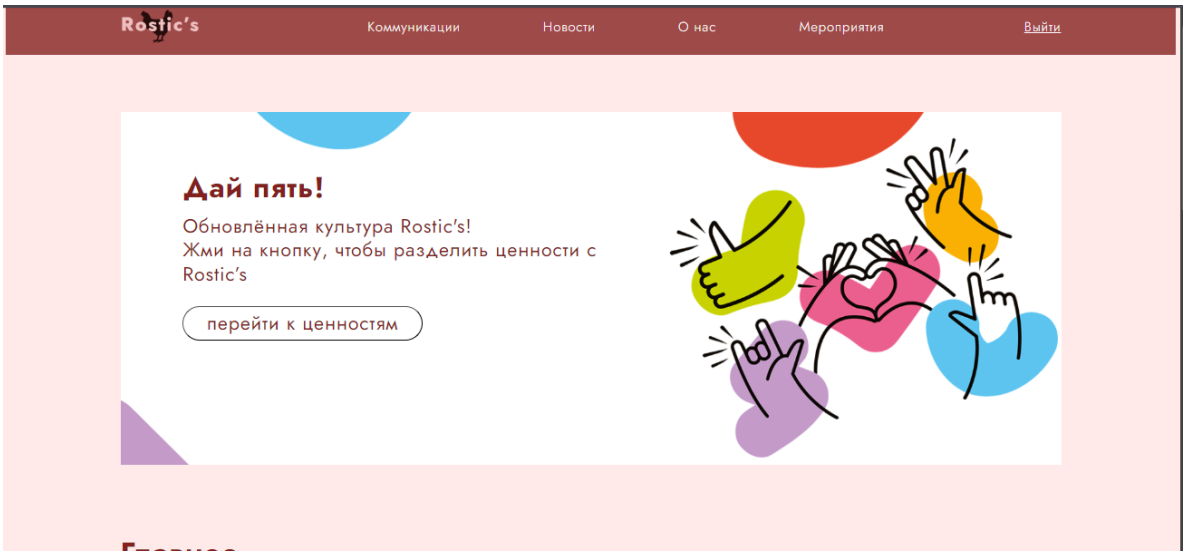

Рисунок 6 – Первый блок главной страницы

Блок «Главное», на котором представлены 3 последние новости компании, каждая новость включает заголовок новости и картинку (Приложение Д).

Блок «О нас» включает краткое описание и историю компании. Контактная информация или подвал сайта предоставляет контактную информацию и реквизиты компании (рисунок 7).

Рисунок 7 – Подвал сайта

Краткая карта сайта представлена на рисунке 8.

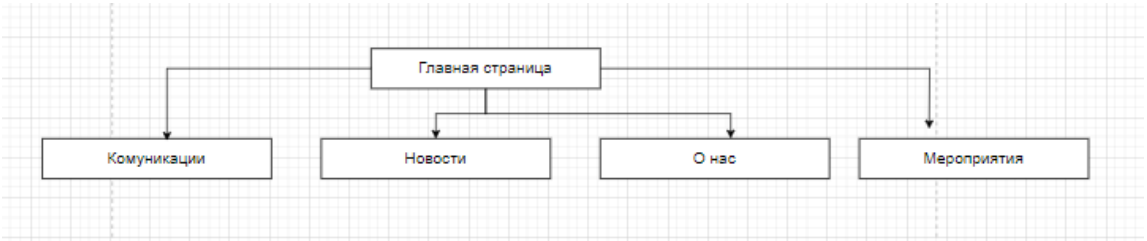

Рисунок 8 – Карта сайта

Также пользователь может оставить обратную связь или задать интересующий вопрос (рисунок 9).

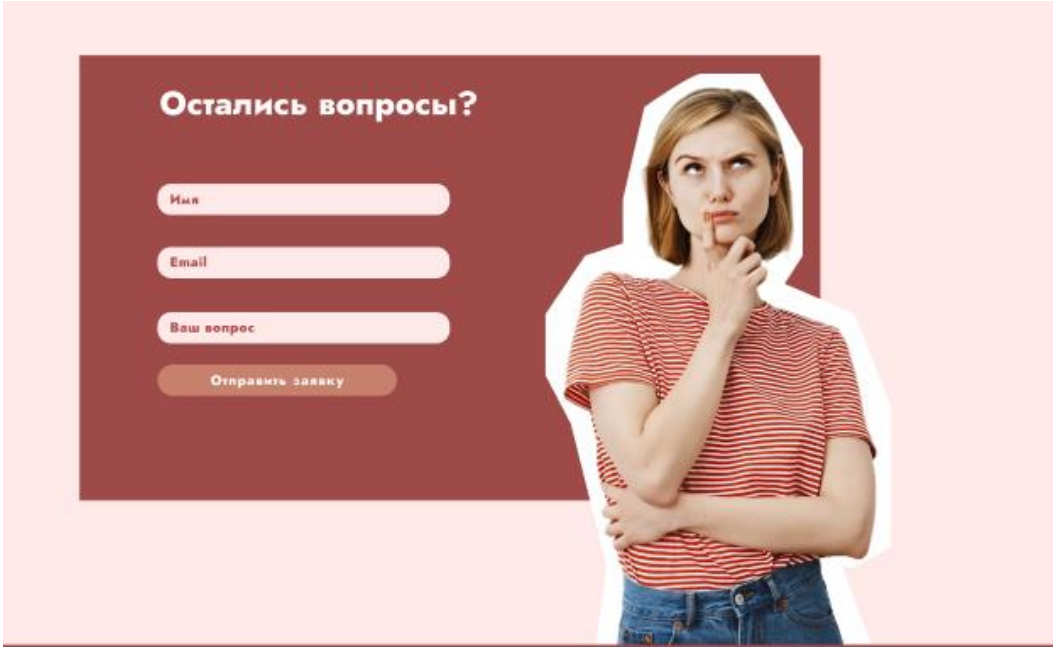

Рисунок 9 – Форма обратной связи

4 Тестирование веб-приложения

4.1 План тестирования

Тестирование веб-приложения — это процесс проверки приложения для выявления ошибок и обеспечения его правильной работы.

Область тестирования: форма авторизации.

Тестирование функции авторизации имеет целью выявление возможных уязвимостей и ошибок в механизме аутентификации, который позволяет пользователям входить в систему, а также контроля доступа к различным уровням функционала. Эффективное тестирование авторизации включает в себя не только проверку корректности реализованных функций, но и оценку их устойчивости к потенциальным угрозам.

Цель данного плана тестирования — определить подходы и стратегии для тестирования функциональности авторизации на сайте. Это включает в себя разработку unit тестов, тест-кейсов для определенной логики работы системы, а также описание порядка интеграционного тестирования.

Тестирование будет охватывать следующие аспекты функциональности авторизации:

- вход с корректными данными;
- вход с некорректными данным;
- обработка ошибок;
- безопасность (например, защита от брутфорс-атак).

Unit тестирование будет сосредоточено на небольших, изолированных частях кода, таких как функции и методы, которые отвечают за авторизацию. Каждый unit тест должен проверять конкретный сценарий.

Unit-тесты: проверка успешной авторизации пользователя:

- входные данные: корректный логин и пароль;
- ожидаемый результат: возвращаемое значение true.

Тест функции обработки некорректного логина или пароля:

- входные данные: некорректный логин или пароль (например, неправильный пароль);
- ожидаемый результат: возвращаемое значение false.

Проверка авторизации с пустыми полями:

- входные данные: пустые поля логин и пароль;
- ожидаемый результат: возвращаемое значение false.

Так же будет проведено интеграционное тестирование. Интеграционное тестирование будет направлено на проверку взаимодействия различных компонентов системы, связанных с авторизацией, таких как база данных, API и интерфейс пользователя.

Порядок интеграционного тестирования:

1. Подготовка среды тестирования:
   - настройка тестовой базы данных с тестовыми пользователями;
   - подготовка окружения с доступом к API авторизации.

2. Тестирование работы с базой данных:

– проверить корректность сохранения данных пользователя при регистрации;

– проверить получение данных пользователя при авторизации.

3. Тестирование взаимодействия с API:

– проверить, что API возвращает корректный ответ при входе с верными данными;

– проверить, что API возвращает правильные ошибки для некорректных данных.

4. Тестирование пользовательского интерфейса:

– проверить, что элементы интерфейса отображаются и функционируют

– правильно, включая поля ввода и кнопки;

– проверить, что сообщения об ошибках отображаются на интерфейсе в случае неудачной авторизации[5].

Данный план тестирования предоставляет структурированный подход к тестированию функциональности авторизации на сайте. Выполнение всех тестов и их мониторинг помогут обеспечит.

4.2 Оценка результатов проведения тестирования

Тестирование веб-приложения — это процесс проверки приложения для выявления ошибок и обеспечения его правильной работы.

Область тестирования: форма авторизации.

Тестирование функции авторизации имеет целью выявление возможных уязвимостей и ошибок в механизме аутентификации, который позволяет пользователям входить в систему, а также контроля доступа к различным уровням функционала. Эффективное тестирование авторизации включает в себя не только проверку корректности реализованных функций, но и оценку их устойчивости к потенциальным угрозам.

Для тестирования данной функции используем тест-кейсы, которык представлены в таблицах 1-3.

Администратор, отвечающий за систему управления пользователями, играет ключевую роль в обеспечении функциональности процедуры регистрации. Его задача включает в себя контроль за корректностью собираемых данных, защиту от несанкционированного доступа и предотвращение злоупотреблений. Эффективное тестирование регистрации пользователей позволяет выявить возможные уязвимости системы, ошибки в логике работы, а также улучшить пользовательский опыт.

Для тестирования данной функции используем тест-кейсы, которык представлены в таблицах 4-5.

Протестируем успешную авторизацию пользователя со всеми верными данными, которая представлена в тест-кейсе в таблице 1.

Таблица 1 – Проверка успешной авторизации пользователя

№1	Проверка заполнения поля		
Ссылка на ТЗ	Исаева Д.С.	Высокий	Заявка
1. У пользователя есть зарегистрированная учетная запись. 2. Пользователь знает свои учетные данные (логин и пароль).		1. Открыть страницу сайта. 2. Ввести корректный логин в поле «Логин». 3. Ввести корректный пароль в поле «Пароль». 4. Нажать на кнопку «Войти».	
Пользователь успешно авторизуется и перенаправляется на главную страницу или личный кабинет. На странице отображается сообщение приветствия с именем пользователя.			

На рисунке 10 изображен корректный ввод данных в форму авторизации.

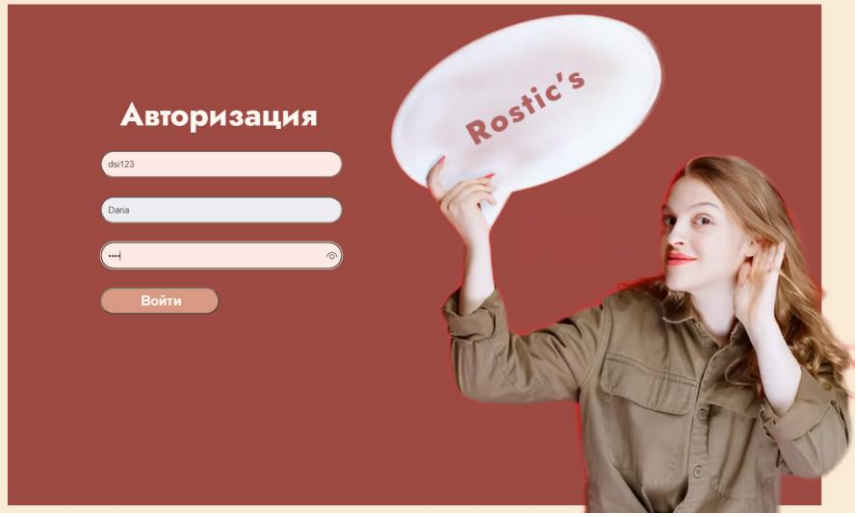

Рисунок 10 – Ввод данных в форму авторизации

После введения данных пользователь переходит на главную страницу сайта. Тестируем авторизицию с неверным логином пользователя. (Таблица 2).

Таблица 2 – Проверка авторизации с некорректным паролем

№2	Проверка заполнения поля		
Ссылка на ТЗ	Исаева Д.С.	Высокий	Заявка
1. У пользователя есть зарегистрированная учетная запись.		1. Открыть страницу сайта. 2. Ввести корректный логин в поле «Логин». 3. Ввести некорректный пароль в поле «Пароль». 4. Нажать кнопку «Войти».	
Появляется сообщение об ошибке, указывающее на некорректные учетные данные. Пользователь остается на странице авторизации.			

На рисунке 11 изображен ввод неверного логина но верного пароля.

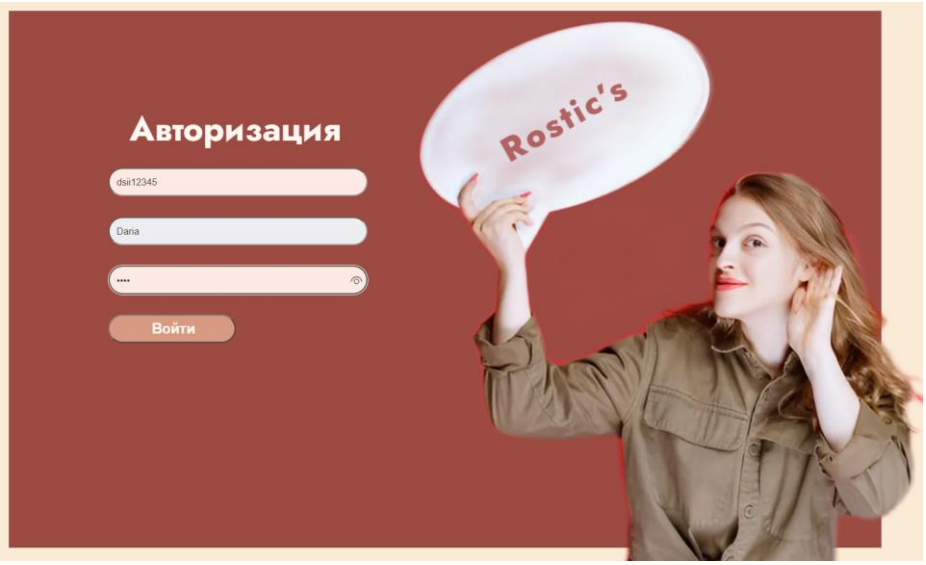

Рисунок 11 – Ввод некорректных данных в форму авторизации

После ввода неверных данных пользователь видит ошибку (рисунок 12).

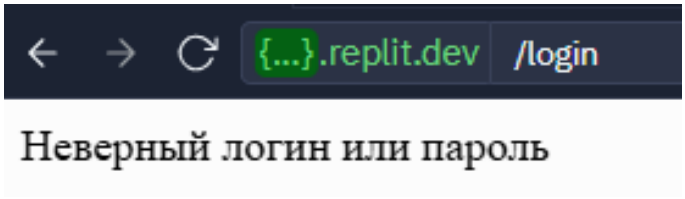

Рисунок 12 – Ошибка

Протестируем функцию авторизации с пустыми полями. В таблице 3 представлен тест-кейс, который подробно описывает процесс проверки данной функции. Он включает в себя описание тестируемых полей, ожидаемые результаты и фактические выводы системы при попытке авторизации без ввода данных.

Таблица 3 – Проверка авторизации с пустыми полями

№3	Проверка заполнения поля		
Ссылка на ТЗ	Исаева Д.С.	Высокий	Заявка
1. Открытый сайт		1. Открыть страницу сайта. 2. Оставить поле «Логин» пустым. 3. Оставить поле «Пароль» пустым. 4. Нажать кнопку «Войти».	
Появляется сообщение об ошибке, указывающее на необходимость заполнения полей логина и пароля. Пользователь остается на странице авторизации.			

На рисунке 13 показан процесс работы с формой авторизации, где

пользователю предоставляется возможность оставить поля для ввода пустыми. Этот тест служит для проверки, как система обрабатывает отсутствие данных, что является критическим аспектом обеспечения безопасной работы приложения.

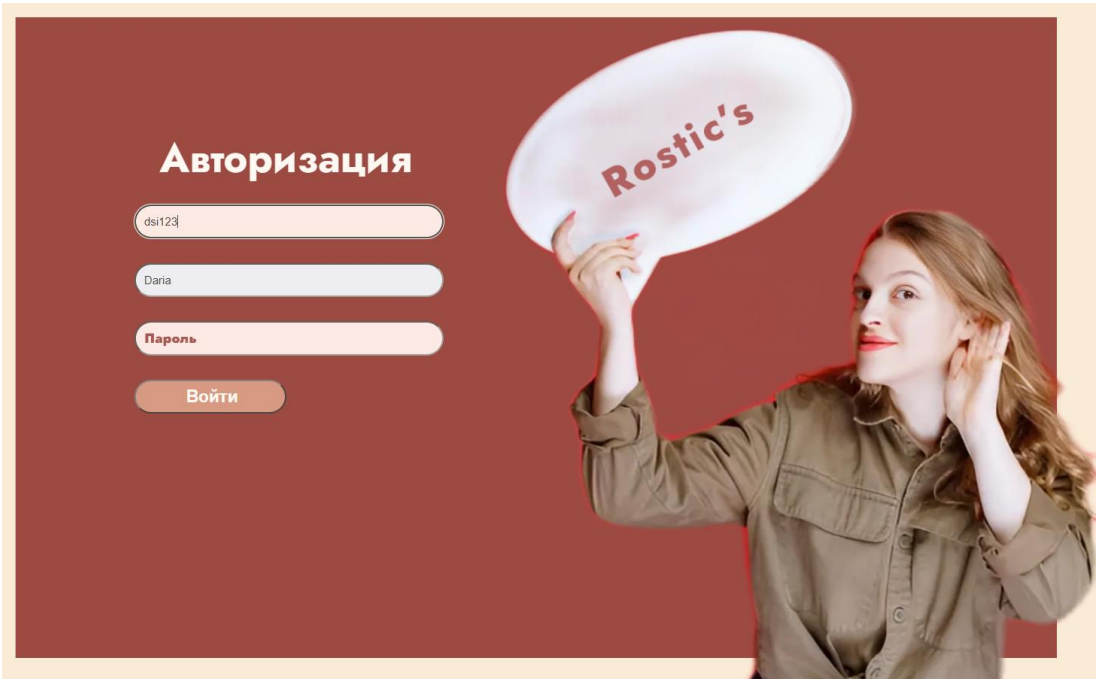

Рисунок 13 – Ввод некорректных данных в форму авторизации

На рисунке 14 мы видим, что пользователю не удалось войти так как он пропустил поле с паролем.

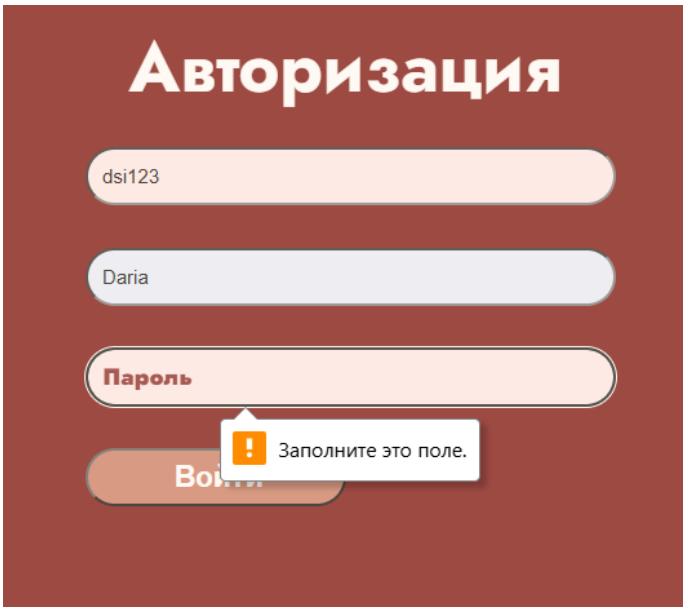

Рисунок 14 – Всплывающая подсказка о пустом поле

Регистрация пользователя является одним из ключевых компонентов любой веб-приложения или мобильного приложения. Эта функция не только позволяет пользователям создавать учетные записи, но и обеспечивает безопасность и управление их данными. Успешная реализация функции регистрации напрямую

влияет на пользовательский опыт, поэтому важно тщательно протестировать ее на наличие ошибок и уязвимостей.

Протестируем функцию регистрации пользователя (Таблица 4). Для успешной регистрации введем валидные данные.

Таблица 4 – Проверка успешной регистрации нового пользователя

№4	Проверка заполнения поля		
Ссылка на ТЗ	Исаева Д.С.	Высокий	Заявка
1. Администратор находится на странице регистрации сайта. 2. Администратор имеет доступ к интернету.		1. Авторизироваться от администратора, войти на страницу регистрации 2. Ввести валидное имя в поле «Имя пользователя». 3. Ввести валидный адрес электронной почты в поле «Логин». 4. Ввести валидный пароль в поле «Пароль». 5. Нажать кнопку «Зарегистрировать».	
Пользователь успешно зарегистрирован. В базе данных добавляется новая запись пользователя с корректными данными.			

На рисунке 15 вводим корректные новые данные.

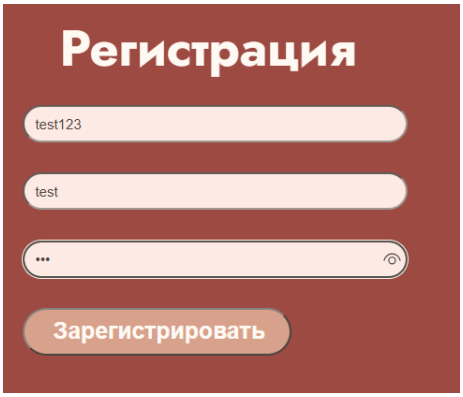

Рисунок 15 – Введение данных для регистрации

На рисунке 16 представлена визуализация, иллюстрирующая успешную регистрацию пользователя в системе. Мы видим подтверждение того, что данные нового пользователя были корректно сохранены в базе данных. Этот этап является критически важным, так как он свидетельствует о том, что функция регистрации работает должным образом и обеспечивает надежность хранения информации.

Список пользователей

ID	Логин	Имя	Пароль	Роль
1	admin	admind	$2b$10$lDnNbICjD2zB55FCpPJfhu3SUgUAQUBxEg5jVMV8UzZTvhIzhckla	2
2	dsi123	Daria	$2b$10$GgbfTbwCc0qD9PvnyIF38ul5Vhpyz6EmjLPxbePONgBbo1q8MkhAW	1
3	test123	test	$2b$10$4/79dC.jjgYAg54wcITvTuPBB2WxZRbMnUZtZvoo1sfD1nszRZf22	1

Рисунок 16 – Данные в таблице «Пользователи»

В таблице 5 представлен тест-кейс, который описывает процесс проверки функции регистрации при попытке использования существующего логина. Тест-кейс содержит описание условий тестирования, шаги, ожидаемый результат и фактический результат, что позволяет четко оценить корректность работы функции.

Таблица 5 – Проверка регистрации с уже существующим Логином

№5	Проверка заполнения поля		
Ссылка на ТЗ	Исаева Д.С.	Высокий	Заявка
1. Пользователь находится на странице регистрации сайта. 2. Пользователь имеет доступ к интернету. 3. В базе данных должна быть создана новая запись с данными, введенными пользователем. 4. Пользователь может войти в систему с использованием зарегистрированных данных.		1. Авторизироваться от администратора, войти на страницу регистрации 2. Ввести валидное имя в поле «Имя пользователя». 3. Ввести существубщий адрес электронной почты в поле «Логин». 4. Ввести валидный пароль в поле «Пароль». 5. Нажать кнопку «Зарегистрировать».	
Ошибка регистрация.			

На рисунке 17 показано, что после попытки регистрации пользователя возникает ошибка. Это может свидетельствовать о том, что введенные данные не соответствуют требованиям системы или уже существует учетная запись с такими же данными.

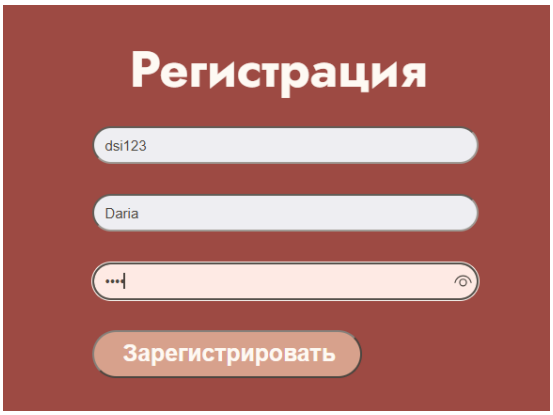

Рисунок 17 – Вводим уже существующего пользователя

Заключение

Данная курсовая работа была посвящена разработке сайта для коммуникации между компанией и ресторанами. В ходе работы была достигнута поставленная цель – разработана концепция и создан прототип информационного портала, обеспечивающего эффективное взаимодействие между двумя целевыми группами: компаниями, заинтересованными в своевременной информированности сотрудников, и самими ресторанами, стремящимися работать по всем правилам и стандартам работы компании. Разработанный сайт призван стать удобным инструментом для обмена информацией, предоставления актуальных новостей и специальных предложений, а также обеспечения эффективной обратной связи.

Работа над проектом началась с проведения анализа предметной области. Были изучены существующие решения в сфере онлайн-коммуникаций между компаниями и ресторанами, выявлены их преимущества и недостатки. Особое внимание уделялось анализу потребностей целевых аудиторий, определению функциональных требований к разрабатываемому сайту. Этот этап позволил сформировать четкое представление о задачах, которые должен решать будущий портал.

На основе проведенного анализа было составлено техническое задание, которое детально описывало функциональные и нефункциональные требования к сайту. В техническом задании были определены основные модули сайта, их функциональность, требования к дизайну, безопасности и производительности. Техническое задание стало основополагающим документом для всех последующих этапов разработки.

Следующим этапом стала разработка дизайна сайта. При этом особое внимание уделялось принципам удобства навигации, информативности, привлекательности и простоты использования. Были разработаны макеты основных страниц сайта, продумана структура меню, выбрана цветовая палитра и шрифты. Дизайн сайта был разработан с учетом современных трендов веб-дизайна и требований целевой аудитории.

Описание хода разработки веб-сайта включало в себя выбор технологий и инструментов разработки, описание архитектуры сайта, а также поэтапное описание процесса создания отдельных модулей и функциональности. В качестве основной технологии разработки был выбран SQlite, HTML, CSS, Node.js, которые обеспечивают высокую производительность, гибкость и масштабируемость.

Процесс создания веб-сайта включал в себя верстку страниц, разработку frontend и backend функциональности, интеграцию с базой данных. На этом этапе были реализованы все основные функции сайта, определенные в техническом задании. Были созданы формы для регистрации пользователей, публикации новостей и специальных предложений, а также модуль обратной связи.

Разработка сайта осуществлялась с использованием современных программных средств, что позволило обеспечить высокое качество кода, оптимизировать производительность и обеспечить безопасность.

Для хранения данных сайта была создана база данных SQlite. Структура базы данных была спроектирована таким образом, чтобы обеспечить эффективное хранение и обработку информации о ресторанах, компаниях, заказах, новостях и других данных.

После завершения разработки был проведен этап тестирования веб-сайта. Тестирование проводилось по различным сценариям использования, с целью выявления и исправления ошибок, а также оценки производительности и безопасности сайта. Было проведено функциональное тестирование. Результаты тестирования показали, что разработанный сайт соответствует требованиям технического задания и готов к использованию.

В заключение следует отметить, что в рамках данной курсовой работы был разработан функциональный прототип сайта для коммуникации между компаниями и ресторанами. Разработанный сайт обладает рядом преимуществ, таких как удобный интерфейс, широкий функционал, высокая производительность и безопасность. В дальнейшем планируется добавить функционал личных кабинетов пользователей. Реализация этой задачи позволит сделать сайт еще более эффективным инструментом для коммуникации между компаниями и ресторанами.

Список использованных источников

1 Бейзер Б. «DAO. Шаблоны проектирования доступа к данным». — СПб.: Питер, 2012. — 368 с.[Дата обращения: 17.12.2024]

2 Бойков, П. Н. (2020). Основы проектирования пользовательских интерфейсов. — Москва: Издательство «Питер».[Дата обращения: 17.12.2024]

3 Вигерс К. «Разработка требований к программному обеспечению» — М.: Питер, 2017. — 768 с.[Дата обращения: 17.12.2024]

4 Гарретт Дж. Дж. «Элементы опыта пользователя. Веб-дизайн для души» — СПб.: Символ-Плюс, 2016. — 320 с.[Дата обращения: 20.12.2024]

5 Закас Н., Брэдбери Д. «CSS. Каскадные таблицы стилей. Подробное руководство». — СПб.: Символ-Плюс, 2018. — 1152 с. [Дата обращения: 21.12.2024]

6 Канер С., Фолк Б., Нгуен Т. «Тестирование программного обеспечения» — М.: Диалектика, 2018. — 544 с.[Дата обращения: 21.12.2024]

7 Кроуфорд С., Дибров М. «JavaScript. Шаблоны». — СПб.: Питер, 2017. — 256 с.[Дата обращения: 31.12.2024]

8 Круг С. «Не заставляйте меня думать. Веб-юзабилити и здравый смысл». — СПб.: Символ-Плюс, 2014. — 288 с.[Дата обращения: 25.12.2024]

9 Кузнецов М. В., Симдянов И. В. «Разработка веб-приложений» — СПб.: БХВ-Петербург, 2021. — 608 с.[Дата обращения: 20.12.2024]

10 Остервальдер А., Пинье И. «Построение бизнес-моделей». — М.: Альпина Паблишер, 2014. — 288 с.[Дата обращения: 17.12.2024]

11 Торнтон К., Уилсон Л. «Дизайн веб-форм» — СПб.: Питер, 2015 — 352 с.[Дата обращения: 16.12.2024]

12 Фаулер М., Райс Д. «Шаблоны корпоративных приложений». — М.: Вильямс, 2015. — 528 с.[Дата обращения: 28.12.2024]

13 Фленган Д. «JavaScript. Подробное руководство». — СПб.: Символ-Плюс, 2019. — 1088 с.[Дата обращения: 5.01.2025]

14 Хейлз Д., Браун К. «Изучаем программирование на JavaScript». — СПб.: Питер, 2020. — 320 с.[Дата обращения: 10.01.2025]

15 Хоган Дж. «Разработка веб-приложений с помощью Node.js». — М.: Вильямс, 2018. — 480 с.[Дата обращения: 17.12.2024]