## Содержание

# Введение

Веб-приложение – это программа или набор программ, которые выполняются на сервере и доступны пользователям через браузер. Веб-приложения обычно работают с данными, хранящимися на сервере, и могут предоставлять различные функции, такие как обработка информации, управление данными, взаимодействие с пользователем и т.д. Одним из преимуществ веб-приложений является то, что они доступны с любого устройства, имеющего доступ в интернет.

Веб-приложения могут выполнять различные функции, в зависимости от их назначения. Некоторые из наиболее распространенных функций включают:

– обработка информации: веб-приложения могут использоваться для сбора, обработки и анализа данных. например, интернет-магазины используют веб-приложения для обработки информации о заказах, платежах и доставке;

– управление данными: веб-приложения также могут использоваться для управления и хранения данных. например, системы управления контентом (cms) используют веб-приложения для хранения и редактирования контента на сайте;

– взаимодействие с пользователем: веб-приложения могут предоставлять пользователю различные возможности для взаимодействия, такие как формы обратной связи, чаты, форумы и т. д.;

– безопасность: веб-приложения должны обеспечивать безопасность данных пользователей и защиту от возможных угроз, таких как хакерские атаки.

Функции веб-приложения, которые должны быть реализованы в ходе данного проекта:

– регистрация и авторизация пользователей: эта функция позволит пользователям создавать аккаунт в вашем интернет-магазине и входить в него для совершения покупок. Регистрация должна быть простой и быстрой, с возможностью использования различных видов учетных записей (например, через социальные сети);

– управление пользователями и ролями: функция управления пользователями и ролями позволит управлять аккаунтами пользователей, изменять их данные и роли в системе;

– отображение информации о товарах в виде карточек товара с фотографиями и ценами: информация о товарах должна быть представлена в виде карточек товаров с фотографиями, ценами, описанием и другими характеристиками; карточки товаров должны быть оптимизированы для мобильных устройств и иметь адаптивный дизайн;

– возможность добавления товаров в корзину и оформление заказа: пользователи должны иметь возможность добавлять товары в свою корзину для дальнейшего оформления заказа; количество товаров в корзине должно отображаться в реальном времени; после того, как пользователь добавил все необходимые товары в корзину, он может перейти к оформлению заказа; здесь

он должен указать свои контактные данные, выбрать способ доставки и оплаты, а также подтвердить заказ.

Тема курсовой работы «Разработка веб-приложения для продажи чая и кофе» является актуальной по нескольким причинам:

– рост рынка интернет-торговли: в последние годы наблюдается значительный рост интернет-торговли, особенно в сфере продуктов питания и напитков;

– удобство и доступность: веб-сайт позволяет покупателям удобно и быстро получить доступ к ассортименту чая и кофе, сравнить цены и ознакомиться с отзывами других покупателей. Это повышает вероятность того, что покупатель сделает выбор в пользу данного сайта при покупке данных товаров;

– конкурентность: рынок чая и кофе насыщен различными брендами и продуктами, поэтому разработка уникального веб-сайта, который будет выгодно отличаться от конкурентов, может стать ключом к успеху;

– персонализация и сегментация: веб-технологии позволяют создавать персонализированные предложения для каждого покупателя, учитывая его предпочтения и историю покупок. Это позволит увеличить лояльность клиентов и их готовность совершать покупки на данном веб-сайте.

Целью данной работы является создание веб-приложения для продажи чая и кофе.

Задачи:

– исследовать предметную область;
– провести анализ предприятия;
– составить техническое задание;
– проанализировать литературные и интернет-источники, посвященные аналогичным продуктам;
– разработать дизайн веб-приложения и его структуру;
– спроектировать веб-приложение;
– провести тестирование веб-приложения;
– реализовать веб-приложение.

Объектом является веб-приложение для продажи чая и кофе.

Предметом является процесс создания веб-приложения для продажи чая и кофе.

Для реализации проекта потребуются следующие ресурсы:

– фреймворк react для создания пользовательского интерфейса;
– node.js для работы серверного кода;
– postgresql для управления базой данных;
– express.js для создания веб-сервера;
– html и css;
– VS Code.

HTML и CSS являются основными технологиями для создания веб-страниц и управления их внешним видом. HTML используется для разметки и структурирования контента на веб-странице, а CSS - для определения стилей и

внешнего вида этой разметки. Вместе они позволяют создавать красивые и функциональные веб-страницы с текстом, изображениями, ссылками, таблицами и многим другим.

JavaScript позволяет добавлять различные элементы на веб-страницы, такие как анимация, всплывающие окна, формы обратной связи и многое другое. JavaScript также используется для работы с API, такими как геолокация, уведомления и обработка ошибок.

React: открытый JavaScript-фреймворк для создания пользовательских интерфейсов. Он был разработан компанией Facebook и стал одним из самых популярных инструментов для создания веб-приложений и мобильных приложений. React используется для создания интерактивных пользовательских интерфейсов, которые могут быстро обновляться и реагировать на действия пользователя. Он также позволяет создавать более быстрые и эффективные приложения, так как позволяет разбивать интерфейс на отдельные компоненты и управлять ими более эффективно.

PostgreSQL – это система управления базами данных (СУБД), которая используется для хранения и управления данными на сервере. Она поддерживает множество функций, таких как транзакции, индексы, триггеры и многое другое. PostgreSQL также является открытым исходным кодом, что означает, что его можно использовать бесплатно и изменять по своему усмотрению.

Node.js – это платформа для разработки серверных приложений на JavaScript. Она использует модель событий и позволяет обрабатывать множество запросов одновременно. Node.js используется для создания различных веб-приложений, таких как веб-сайты, API, мессенджеры и многое другое.

Express.js – это веб-фреймворк для Node.js, который используется для создания серверной части веб-приложений. Он предоставляет набор инструментов для обработки запросов, маршрутизации, обработки ошибок и многого другого. Express.js является простым в использовании и гибким, что делает его популярным выбором для многих разработчиков.

Visual Studio Code – это редактор кода с открытым исходным кодом, разработанный компанией Microsoft. Он имеет множество функций, которые упрощают процесс разработки, таких как подсветка синтаксиса, IntelliSense, отладка и поддержка множества языков программирования. Visual Studio Code также имеет множество плагинов, которые расширяют его функциональность.

# 1 Анализ предметной области

Интернет-магазин является удобным и доступным способом приобретения средств от насекомых и грызунов. Для успешной работы интернет-магазина необходимо провести анализ предметной области, чтобы определить целевую аудиторию, конкурентов, преимущества и недостатки бизнеса.

Чай играет важную роль в современной культуре. Он является одним из самых популярных напитков в мире и употребляется в различных формах, включая черный, зеленый, улун и травяной чай. Чай также имеет множество полезных свойств, включая антиоксидантные, противовоспалительные и антиканцерогенные. Он также может помочь улучшить здоровье сердца, снизить риск развития диабета и улучшить общее самочувствие.

Интернет-магазин чая и кофе должен включать в себя следуюшие функции:

– удобный каталог товаров: интернет-магазин должен иметь удобный каталог товаров, который позволяет покупателям легко находить нужную продукцию. Каталог должен содержать подробную информацию о каждом товаре, включая его название, цену и фотографию;

– фильтрация: интернет-магазин должен предоставлять возможность поиска товаров по типу и категории. Фильтрация должна быть быстрой и точной, чтобы покупатели могли легко найти то, что им нужно;

– корзина покупок: интернет-магазин должен иметь функциональную корзину покупок, которая позволяет покупателям добавлять товары в корзину и оформлять заказ. Корзина должна содержать информацию о количестве товаров, общей стоимости и статусе заказа;

– оформление заказа: интернет-магазин должен предлагать различные способы оплаты и доставки товаров. Покупатели должны иметь возможность выбрать наиболее подходящий для них способ оплаты и доставки;

– регистрация и авторизация: интернет-магазин должен предоставить покупателям возможность регистрироваться и авторизовываться на сайте;

– личный кабинет: интернет-магазин должен предоставить покупателям возможность создать личный кабинет, где они могут управлять своими заказами, просматривать историю покупок.

Целевая аудитория интернет-магазина чая и кофе включает в себя широкий круг потребителей. Это могут быть как любители чая и кофе для домашнего использования, так и профессиональные бариста, ищущие новые вкусы и сорта для своих кофеен. Также к целевой аудитории можно отнести корпоративных клиентов, которые заказывают чай и кофе для офисов и мероприятий. Возраст потребителей может варьироваться от 18 до 60 лет, с различным уровнем дохода и социальным статусом.

Конкуренты интернет-магазина могут быть представлены другими онлайн-магазинами, торгующими чаем и кофе, а также оффлайн-магазинами в городах, где есть спрос на данную продукцию.

Преимущества интернет-магазина включают в себя:

– большой выбор товаров: в интернет-магазине можно найти множество сортов чая и кофе от различных производителей;

– удобство: покупатели могут легко сравнивать товары, читать отзывы и делать заказы в любое время и из любого места;

– быстрая доставка: большинство заказов доставляются в течение нескольких дней;

– скидки и акции: интернет-магазины часто предлагают скидки и акции на товары, что может привлечь новых покупателей;

– экологичность: использование интернет-технологий для продаж уменьшает количество отходов и экономит ресурсы;

– возможность масштабирования: интернет-магазин можно легко масштабировать, добавляя новые товары и услуги, а также расширяя географию продаж.

Недостатки интернет-магазина связаны с возможными проблемами доставки и возврата товара, а также с необходимостью проверки качества продукции перед покупкой. Кроме того, некоторые покупатели могут предпочитать покупать товары в оффлайн-магазинах, так как это дает возможность лично проверить качество продукции.

В целом, интернет-магазин чайных и кофейных напитков является перспективным бизнесом, который может привлечь широкую аудиторию потребителей.

# 2 Проектирование приложения

Проектирование веб-приложения магазина чая и кофе можно разделить на следующие этапы:

– разработка дизайна веб-приложения;
– проектирование структуры веб-приложения (разделы, страницы);
– разработка функционала веб-приложения.

Для начала рассмотрим дизайн нашего веб-приложения (см. Приложение Г).

На главной странице сайта в разделе «УТП» представлены следующие элементы:

– логотип;
– меню;
– кнопка (иконка) перехода на страницу «Авторизация»;
– кнопка (иконка) перехода на страницу «Корзина и оформление заказа»;
– картинки;
– заголовок с названием магазина;
– подзаголовок с кратким описанием;
– кнопка перехода на страницу «Каталог».

На главной странице сайта в разделе «О нас» представлены следующие элементы:

– заголовок;
– краткое описание компании.

На главной странице сайта в разделе «Категории» представлены следующие элементы:

– заголовок;
– картинки с названиями категорий (при наведении);
– кнопка перехода на страницу «Каталог».

На главной странице сайта в разделе «Доставка и оплата» представлены следующие элементы:

– заголовок;
– блок с информацией о доставке и оплате.

На главной странице сайта в разделе «Контакты» представлены следующие элементы:

– заголовок;
– интерактивная Яндекс-карта;
– блок с контактной информацией.

На главной странице сайта в разделе «footer» представлены следующие элементы:

– логотип;
– меню;
– ссылки на социальные сети в виде иконок.

На странице сайта «Каталог» представлены следующие элементы:

– меню;

– заголовок;

– фильтрация;

– карточки товаров.

На странице сайта «Корзина и оформление заказа» представлены следующие элементы:

– меню;

– блок с товарами, добавленными в корзину;

– блок с оформлением заказа;

– кнопка «Оформить».

На странице сайта «Личный кабинет» представлены следующие элементы:

– меню;

– блок с личной информацией пользователя;

– кнопка «Выйти».

На странице сайта «Авторизация» представлены следующие элементы:

– меню;

– блок с формой ввода данных;

– кнопка «Войти»;

– ссылка на страницу «Регистрация», если пользователь еще не зарегистрирован.

На странице сайта «Регистрация» представлены следующие элементы:

– меню;

– блок с формой ввода данных;

– кнопка «Зарегистрироваться»;

– ссылка на страницу «Авторизация», если пользователь уже зарегистрирован.

Теперь перейдем к рассмотрению структуры веб-приложения.

Создание шаблонов страниц веб-приложения необходимо для:

– повышения производительности: шаблоны страниц веб-приложения позволяют быстрее создавать новые страницы, так как они могут использовать уже готовые блоки и элементы дизайна, что снижает затраты времени на разработку и улучшает общую производительность;

– повторное использование кода: шаблоны предоставляют возможность повторного использования кода, что уменьшает количество ошибок и упрощает процесс обновления веб-приложения;

– поддержка масштабируемости: с помощью шаблонов можно легко добавлять новые страницы и разделы в веб-приложение, не нарушая его структуру и дизайн, это обеспечивает большую масштабируемость и гибкость веб-приложения;

– удобный поиск по странице: за счет свой структурированности шаблоны упрощают поиск определенного элемента сайта;

– легкость обновления: если нужно внести изменения в дизайн или функционал веб-приложения, достаточно обновить определенный элемент, и все страницы с этим элементом будут обновлены.

Теперь рассмотрим общий шаблон нашего веб-приложения (см. рисунок 13).

```
<Header />
<div className="wrapper">…
</div>
<Footer />
</>
```

Рисунок 1 – Общая структура

Также в этом пункте следует рассмотреть такие общие элементы сайта, как шапка сайта (header) (см. Рисунок 14) и подвал сайта (footer) (см. Рисунок 15), которые используются для всех страниц сайта.

```
<div className="wrapper">
    <div className="header">
    <ul className="menu">
      <Link to={'/App'}>
        <li><a href='#home'>Главная</a></li>
      </Link>
      <Link to={'/App'}>
        <li><a href='#about'>О нас</a></li>
      </Link>

            <li><a href='#categories'>Каталог</a></li>
    </ul>
    <div className="logo">
      <a href='#home'><img src={Logo} /></a>
      <a href='#home'><h1>Tea History</h1></a>
    </div>
    <ul className="menu">
        <li><a href='#delivery'>Доставка</a></li>
        <li><a href='#contacts'>Контакты</a></li>
    </ul>
    <div className="header_icons">
      <Link to={'/Login'}>
        <img src={User} />
      </Link>
      <Link to={'/Cart'}>
        <img src={Card} />
      </Link>
    </div>
</div>
</div>
```

Рисунок 2 – Шапка сайта (header)

```
<div className="footer">
        <div className="footer_logo">
            <img src={Logo} />
            <h1>Tea History</h1>
        </div>
            <ul className='footer_menu'>
              <li><a href="">Главная</a></li>
              <li><a href="">О нас</a></li>
              <li><a href="">Каталог</a></li>
              <li><a href="">Доставка</a></li>
              <li><a href="">Контакты</a></li>
            </ul>
        <div className="footer_icons">
                <img src={Inst} />
                <img src={Tg} />
        </div>
    </div>
```

Рисунок 3 – Подвал сайта (footer)

Теперь рассмотрим функциональные возможности нашего веб-приложения. На сайте представлены следующие функциональные возможности:

– авторизация и регистрация пользователей: предоставление возможности новым пользователям зарегистрироваться и войти в систему, а также сохранение учетных данных для уже авторизованных пользователей;

– добавление товаров в корзину: возможность добавления товаров в виртуальную корзину для дальнейшего оформления заказа;

– оформление заказа: предоставление формы для указания адреса доставки, контактных данных, выбора способа оплаты и подтверждения заказа;

– ведение личного кабинета пользователя: предоставление пользователю возможности редактирования личной информации профиля.

Для наглядности работы данных функций в приложении А на рисунке А.1 представлена «Диаграмма прецедентов».

# Разработка программного обеспечения

## 3.1 Описание технологического стека разработки

В качестве технологического стека разработки веб-приложения был выбран Pern stack. Pern stack состоит из PostgreSQL, Express, React и Node. Помогает в создании полнофункционального веб-приложения.

React – это популярная JavaScript-библиотека для создания пользовательских интерфейсов (UI), разработанная компанией Facebook. Она позволяет создавать интерактивные и реактивные интерфейсы, которые могут обновляться в реальном времени при изменении данных или состоянии приложения.

Выбор React в качестве основного стека для разработки веб-приложения продиктован несколькими причинами:

– реактивность: react обеспечивает реактивный подход к созданию пользовательских интерфейсов, что позволяет приложениям обновлять свои элементы автоматически при изменении состояния приложения. это делает процесс разработки более эффективным и удобным;

– производительность: react оптимизирован для производительности, что делает его идеальным выбором для больших и сложных приложений. он также поддерживает асинхронное обновление состояния, что снижает нагрузку на браузер;

– компонентный подход: react использует концепцию компонентов, что позволяет разработчикам разбивать приложения на небольшие, независимые и повторно используемые блоки. это ускоряет разработку и улучшает масштабируемость приложений;

– эффективное использование ресурсов: react позволяет управлять зависимостями между компонентами, что предотвращает перерисовку ненужных элементов и оптимизирует использование ресурсов;

– поддержка сообщества: react имеет активное и большое сообщество разработчиков, что обеспечивает доступ к большому количеству ресурсов, таких как библиотеки, инструменты и примеры кода.

Node.js - это среда выполнения JavaScript, которая позволяет выполнять JavaScript-код на сервере. Node.js использует модель событий, которая позволяет разрабатывать масштабируемые и высокопроизводительные приложения.

Преимущества Node.js:

– высокая производительность: node.js использует асинхронное выполнение кода, что позволяет снизить нагрузку на процессор и увеличить производительность приложения.

– масштабируемость: node.js позволяет легко масштабировать приложения, так как они могут быть распределены на множество серверов без потери производительности.

– веб-сокеты: node.js поддерживает веб-сокеты, что позволяет создавать приложения с постоянной связью между клиентом и сервером.

– большое сообщество: node.js имеет большое и активное сообщество разработчиков, которое помогает новым разработчикам быстрее освоиться в разработке на node.js.

– использование javascript: node.js позволяет использовать тот же язык программирования (javascript) на сервере, что и на клиенте, что упрощает обучение и разработку.

PostgreSQL - это объектно-реляционная система управления базами данных с открытым исходным кодом.

Преимущества PostgreSQL:

– открытость и доступность исходного кода: postgresql имеет открытый исходный код, что позволяет разработчикам создавать свои приложения и инструменты, а также изменять код для улучшения производительности или добавления новых функций;

– высокая производительность и масштабируемость: postgresql может обрабатывать большие объемы данных и масштабироваться для работы с еще большими объемами;

– гибкость в работе с данными: postgresql поддерживает множество типов данных и позволяет создавать сложные запросы с использованием языка запросов sql;

– поддержка транзакций: postgresql обеспечивает безопасную работу с данными и гарантирует их целостность благодаря поддержке транзакций;

– совместимость с другими субд: postgresql совместим с другими системами управления базами данных, что позволяет легко интегрировать его с существующими системами;

– поддержка различных платформ: postgresql работает на многих операционных системах, включая linux, macos и windows, что делает его доступным для широкого круга пользователей;

– безопасность: postgresql имеет встроенные механизмы безопасности, которые помогают защитить данные от несанкционированного доступа и других угроз.

Express – это минималистичный и гибкий веб-фреймворк для приложений Node.js, предоставляющий обширный набор функций для мобильных и веб-приложений.

Выбор Express, как фреймворка React, обусловлен следующими преимуществами:

– минимализм: express.js имеет минимальный набор функций, что упрощает ее использование и понимание;

– производительность: express.js оптимизирована для высокой производительности, она быстро обрабатывает запросы и отвечает на них;

– модульность: express.js построена на модульной архитектуре, что позволяет разработчикам выбирать только те функции, которые им нужны;

– поддержка сообщества: express.js имеет активное сообщество разработчиков и пользователей, которые предоставляют поддержку и советы по использованию фреймворка;

– гибкость: express.js предоставляет гибкость в настройке и расширении, что позволяет разработчикам адаптировать фреймворк под свои нужды.

## 3.2 Описание алгоритма работы

Веб-приложение - это программа, которая работает на компьютере пользователя через интернет. Она может быть написана на любом языке программирования и использовать различные технологии.

Алгоритм работы веб-приложения – это последовательность действий, которые выполняются при запуске приложения и при взаимодействии с пользователем. Алгоритм может включать в себя обработку ввода пользователя, работу с базой данных, отправку запросов на сервер и получение ответов, а также обработку ошибок и сбоев.

Принцип функционирования веб-приложения заключается в следующем:

– пользователь открывает веб-браузер и вводит адрес веб-сайта в адресной строке;

– веб-браузер отправляет запрос на сервер, где находится веб-приложение;

– сервер обрабатывает запрос и отправляет ответ обратно пользователю;

– веб-приложение отправляет html-код и другие ресурсы (например, css, javascript, изображения) на устройство пользователя;

– веб-браузер отображает полученную веб-страницу на экране пользователя;

– пользователь может взаимодействовать с веб-приложением, нажимая на кнопки, вводя информацию в формы и т.д.;

– веб-приложение обрабатывает полученные данные и отправляет их на сервер для дальнейшей обработки;

– сервер выполняет необходимые операции с полученными данными и отправляет ответ обратно веб-приложению;

– веб-приложение отображает результат на экране пользователя.

Теперь рассмотрим принцип функционирования нашего веб-приложения.

Веб-приложение интернет-магазина чая и кофе использует базу данных PostgreSQL для хранения данных. PostgreSQL - это объектно-реляционная система управления базами данных с открытым исходным кодом. Она поддерживает множество языков программирования, включая JavaScript (через библиотеку pg). Ознакомиться с моделью базы данных можно в Приложении В.

При регистрации пользователя данные, такие как имя, адрес электронной почты и пароль, сохраняются в базе данных. При оформлении заказа информация о выбранном товаре, адресе доставки и платежном методе также записывается в базу данных.

В приложении используются события, такие как клики на кнопки, выбор товара или изменение количества в корзине. Эти события запускают различные функции JavaScript, которые обрабатывают данные действия и обновляют

состояние приложения. Например, при клике на кнопку «Купить» выполняется функция, которая добавляет выбранный товар в корзину и обновляет количество товаров в корзине.

События также могут инициировать отправку данных на сервер. Например, когда пользователь нажимает кнопку «Оформить заказ», отправляется запрос на сервер, который обрабатывает заказ и выполняет необходимые операции в базе данных, такие как добавление товаров в список заказов или обновление количества товаров на складе.

Для наглядного представления функционирования веб-приложения в Приложении Б на рисунке Б.1 представлена «Диаграмма деятельности».

### 3.3 Описание интерфейса пользователя

Интерфейс веб-приложения - это внешний вид и функциональность сайта или приложения, который пользователи видят и используют для взаимодействия с ним. Он включает в себя все элементы сайта, такие как кнопки, формы, изображения, тексты и другие компоненты, которые помогают пользователям выполнять задачи и достигать своих целей.

Принципы разработки интерфейса веб-приложения:

– простота использования: интерфейс должен быть простым и интуитивным, чтобы пользователь мог легко найти нужную информацию и выполнить необходимые действия;

– гибкость: интерфейс должен адаптироваться под различные устройства и разрешения экрана, чтобы пользователи чувствовали себя комфортно на любом устройстве;

– эффективность: интерфейс должен быть эффективным и быстрым, чтобы пользователи не тратили много времени на поиск нужной информации;

– безопасность: интерфейс должен обеспечивать защиту данных пользователя и безопасность его транзакций;

– доступность: интерфейс должен быть доступен для всех пользователей, включая людей с ограниченными возможностями;

– эстетика: интерфейс должен выглядеть привлекательно и соответствовать корпоративному стилю компании.

На рисунке 1 изображена схема взаимодействия пользователя с сайтом. При входе на сайт пользователь попадает на главную страницу сайта, где может ознакомиться с такими разделами сайта, как «УТП», «О нас», «Категории», «Доставка и оплата», «Контакты».

В меню, а также в разделах сайта «УТП» и «Категории», пользователь может перейти по кнопке на страницу «Каталог».

На странице «Каталог» пользователь может ознакомиться с товарами магазина, если пользователь заинтересовался в каком-либо товаре, то он может перейти по кнопке «Купить». После нажатия на кнопку товар добавляется в корзину.

На странице «Корзина» пользователь может смотреть и редактировать свои товары. Если пользователь захочет оформить заказ, то ему нужно заполнить форму, введя свое имя, номер телефона, email и адрес. При успешной отправки данных пользователь переместится на страницу «Успешно», а в случае ошибки – на страницу «Ошибка».

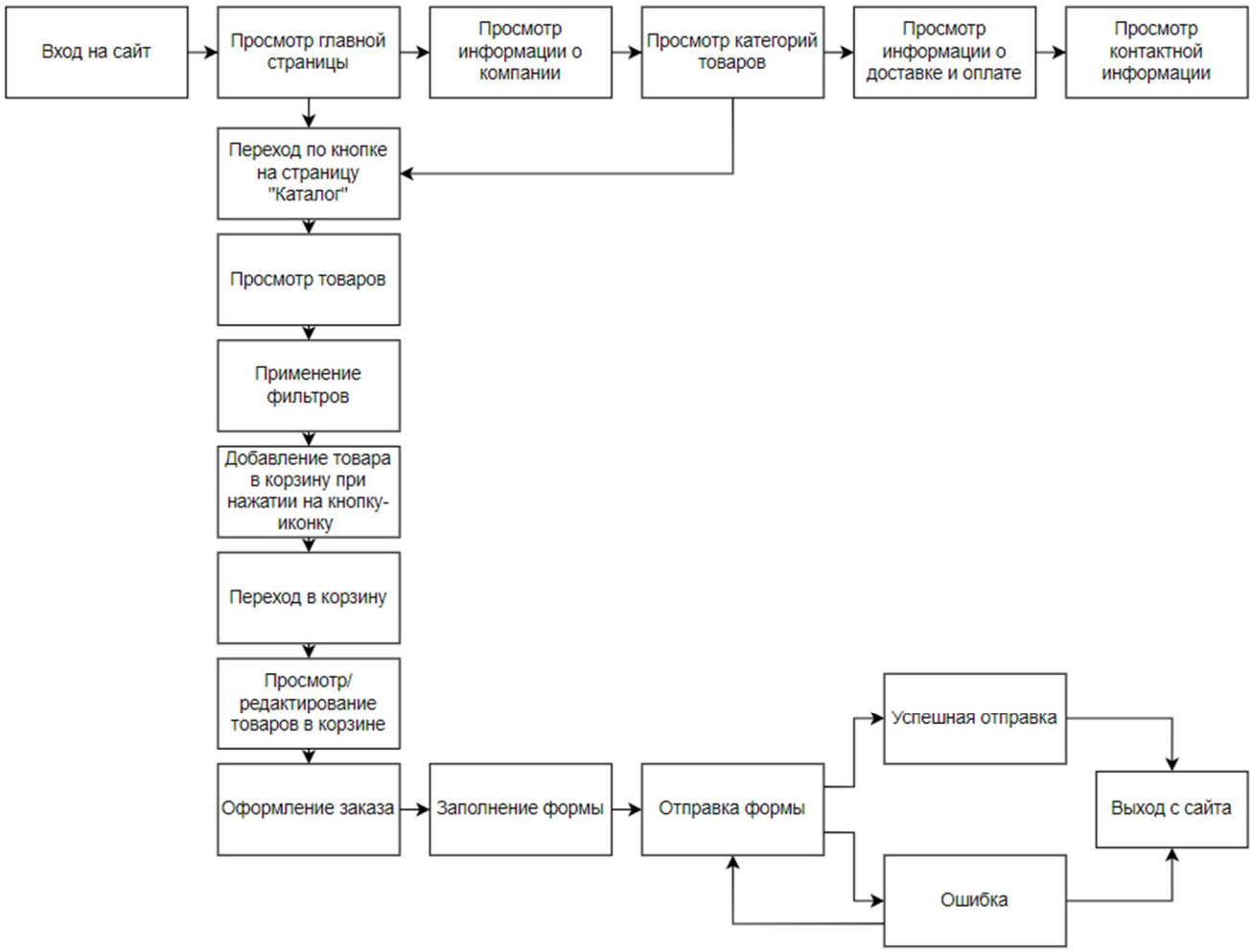


Рисунок 4 – Схема интерфейса

# 4 Тестирование приложения

## 4.1 План тестирования

Интеграционное тестирование - это процесс тестирования, который проверяет насколько хорошо различные компоненты системы работают вместе.

Цель интеграционного тестирования - убедиться, что компоненты системы взаимодействуют друг с другом корректно и не вызывают ошибок или сбоев в работе всей системы.

Некоторые задачи интеграционного тестирования:

- проверка взаимодействия модулей;
- обеспечение совместимости;
- раннее обнаружение проблем;
- повышение общей надёжности системы;
- повышение качества ПО за счёт выявления и устранения ошибок до того, как это станет более сложным и дорогостоящим процессом.

Преимущества интеграционного тестирования:

– позволяет выявить проблемы на более раннем этапе разработки, что снижает вероятность возникновения серьезных ошибок в будущем;

– улучшает качество кода, так как позволяет обнаружить и исправить ошибки, которые могли бы остаться незамеченными при модульном тестировании;

– уменьшает количество регрессионных тестов, так как интеграционное тестирование проверяет взаимодействие между компонентами, которое может измениться при внесении изменений в один из компонентов.

Недостатки интеграционного тестирования:

– может занимать больше времени, чем модульное тестирование, так как требует тестирования всех комбинаций компонентов;

– может быть сложнее в организации и управлении, так как включает в себя больше людей и процессов;

– может требовать больше ресурсов, таких как время программистов и тестовых специалистов.

План проведения интеграционного тестирования:

Определение целей и задач интеграционного тестирования:

– проверить корректность работы всех функций и возможностей веб-приложения;

– определить эффективность взаимодействия между различными компонентами веб-приложения и его окружением;

– выявить возможные ошибки и недочеты в процессе интеграции отдельных частей приложения.

Выбор тестовых сценариев и методов:

– разработать тестовые сценарии, которые будут проверять все основные функции веб-приложения, а также его интеграцию с другими системами;

– выбрать подходящие методы тестирования, такие как функциональное тестирование, интеграционное тестирование, регрессионное тестирование и т. д.

Подготовка тестовой среды:

– создать тестовую среду, которая будет имитировать реальную среду функционирования веб-приложения (например, использовать виртуальные машины);

– настроить тестовую среду для проведения интеграционного тестирования, включая настройку серверов, баз данных, сетей и т.п.

Выполнение тестирования:

– провести интеграционное тестирование веб-приложения в соответствии с разработанными тестовыми сценариями;

– зафиксировать результаты тестирования, включая обнаруженные ошибки, проблемы и несоответствия.

Анализ результатов тестирования:

– проанализировать полученные результаты тестирования и определить, соответствуют ли они требованиям и ожиданиям;

– подготовить отчет о проведенном интеграционном тестировании, указав в нем все обнаруженные проблемы и рекомендации по их устранению.

Устранение выявленных проблем:

– осуществить корректировку и исправление обнаруженных проблем и ошибок в веб-приложении;

– повторно провести интеграционное тестирование после исправления ошибок для подтверждения их устранения.

Завершение тестирования:

– завершить процесс интеграционного тестирования и оформить все необходимые документы, подтверждающие успешное завершение тестирования;

– передать веб-приложение в эксплуатацию после успешного завершения тестирования.

## 4.2 Оценка результатов проведения тестирования

После составления плана тестирования было проведено само интеграционное тестирование следующих функций:

– регистрация;

– авторизация;

– добавление товаров в корзину;

– оформление заказа.

Регистрация

При тестировании регистрации были подтверждены функции:

– регистрация нового пользователя происходит корректно;

– подтверждение регистрации создает новую учетную запись;

– если данные введены некорректно, система выдает ошибку и просит повторить ввод данных.

Авторизация

Тестирование авторизации прошло успешно. Были использованы различные учетные записи для входа в систему. В результате тестирования было подтверждено, что:

– учетные записи создаются при регистрации пользователей;

– пользователи могут успешно войти в систему, использовав правильные учетные данные;

– некорректный ввод учетных данных вызывает сообщение об ошибке;

– после нескольких неудачных попыток входа учетная запись блокируется.

Добавление товаров в корзину:

– тестирование прошло успешно, товары были добавлены в корзину с использованием разных учетных записей;

– количество товаров в корзине соответствовало количеству добавленных товаров;

– товары были успешно удалены из корзины, и количество товаров соответствовало новому состоянию корзины.

Оформление заказа:

– заказ был успешно оформлен с использованием данных учетной записи;

– данные были корректно отправлены на сервер;

– система подтвердила успешное оформление заказа, и пользователь был перенаправлен на страницу подтверждения заказа.

Интеграционное тестирование веб-приложения интернет-магазина было успешно проведено. Все функции, включая регистрацию и авторизацию пользователей, добавление товаров в корзину и оформление заказов, работают корректно. Система правильно обрабатывает различные сценарии, такие как ввод некорректных учетных данных или добавление товаров в корзину. Все ошибки и исключения обрабатываются должным образом, и система предоставляет пользователю понятные сообщения об ошибках. Также было подтверждено, что система корректно взаимодействует с другими системами, такими как платежные системы и службы доставки. В целом, веб-приложение интернет-магазина функционирует должным образом и готово к запуску и использованию пользователями.

# Заключение

В заключении проекта по созданию веб-приложения для продажи чая и кофе, можно сделать вывод, что данный проект является актуальным и перспективным. Рынок чайных и кофейных напитков постоянно растет, и спрос на эти товары не уменьшается. Интернет-магазин позволит расширить рынок сбыта для производителей и продавцов этих товаров, а также предоставит покупателям удобный способ покупки.

В ходе работы были изучены основные аспекты создания интернет-магазина, такие как выбор платформы для разработки, определение ассортимента товаров, разработка дизайна и структуры сайта.

Проект имеет хорошие перспективы развития, так как рынок чайных и кофейных напитков продолжает расти, а использование интернет-магазинов становится все более популярным. Благодаря использованию современных технологий и маркетинговых инструментов, интернет-магазин сможет успешно конкурировать на рынке и приносить прибыль своим создателям.

В ходе выполнения данного проекта был спроектирован, реализован и внедрен программный продукт «Веб-приложение для продажи чая и кофе».

В процессе достижения вышеуказанной цели была исследована предметная область, а также проведен анализ целевой аудитории и конкурентов. Также были проанализированы литературные и интернет-источники, посвященные программным продуктам, аналогичным разрабатываемому, и сами аналогичные программные продукты, вследствие чего были сформулированы требования к разрабатываемому веб-приложению. Определены характеристики, которые делают веб-приложение понятным, удобным и функциональным:

– простой и понятный интерфейс: веб-приложение должно иметь простой и интуитивно понятный интерфейс, который позволяет пользователям легко находить нужные товары и оформлять заказы;

– быстрая загрузка страниц: веб-приложение должно быстро загружаться, чтобы пользователи не теряли терпение и не уходили на другие сайты;

– наличие поиска: веб-приложение должно иметь функцию поиска, чтобы пользователи могли быстро найти нужный товар;

– удобное меню навигации: веб-приложение должно иметь удобное меню навигации, которое позволяет пользователям быстро переходить на нужные страницы;

– наличие фильтров и сортировки товаров: веб-приложение должно предоставлять возможность фильтрации и сортировки товаров по различным параметрам, таким как цена, бренд, категория и т.д.;

– возможность добавления товаров в избранное или в список сравнения: пользователи должны иметь возможность добавлять товары в избранное или сравнивать их между собой;

– поддержка разных способов оплаты и доставки: веб-приложение должно поддерживать различные способы оплаты и доставки товаров, чтобы удовлетворить потребности разных пользователей;

– наличие отзывов и рейтингов товаров: пользователи часто обращают внимание на отзывы и рейтинги товаров перед покупкой, поэтому веб-приложение должно предоставлять эту информацию.

Функции, которые были реализованы в ходе данного проекта:

– авторизация;

– регистрация;

– фильтрация по категориям;

– добавление товаров в корзину;

– оформление заказа.

Сложности, с которыми мы столкнулись в ходе данного проекта:

– разработка адаптивного дизайна для мобильных устройств;

– интеграция с системами оплаты и доставки;

– функция поиска;

– организация системы отзывов и рейтингов товаров;

– обеспечение безопасности личных данных пользователей и защита от мошенничества.

Возможности для развития веб-приложения:

– расширение ассортимента товаров;

– внедрение системы лояльности для постоянных клиентов;

– разработка мобильного приложения для удобства пользователей;

– интеграция с социальными сетями для быстрого входа и обмена отзывами;

– оптимизация работы приложения для повышения скорости загрузки и улучшения пользовательского опыта;

– проведение маркетинговых кампаний и акций для привлечения новых клиентов.

После исследования предметной области и анализа информационных источников, были разработаны структура и дизайн сайта.

После проверки на соответствие требованиям и тестирования, сайт был внедрен для автоматизации бизнес-процессов.

Таким образом, следует считать, что задачи курсовой работы полностью выполнены и цель достигнута.

# Список использованных источников


1 Building a RESTful API with Express and PostgreSQL. - [Электронный ресурс]. - Режим доступа: https://scotch.io/tutorials/getting-started-with-node-express-and-postgres-using-sequelize (дата обращения: 15.12.2023)

2 React.js Tutorial - Full Course for Beginners. - [Электронный ресурс]. - Режим доступа: https://www.freecodecamp.org/news/react-js-tutorial-for-beginners/ (дата обращения: 14.12.2023)

3 Node.js Tutorial - Learn Node.js in 1 Hour. - [Электронный ресурс]. - Режим доступа: https://www.youtube.com/watch?v=TlB_eWDSMt4 (дата обращения: 20.12.2023)

4 Единая система документации. - [Электронный ресурс]. - Режим доступа: https://docs.cntd.ru/document/1200007627?ysclid=l4b3gsfkta201134956 (дата обращения: 22.12.2023)

5 Пользовательские сценарии. - [Электронный ресурс]. - Режим доступа: https://pcnews.ru/top/articles/mat-drawer/polzovatelskie_scenarii_cto_eto_takoe_kak_i_dla_cego_ih_nuzno_stroit-753459.html?ysclid=l4b3qqrz5g980601158 (дата обращения: 9.12.2023)

6 Пользовательские сценарии. - [Электронный ресурс]. - Режим доступа: https://netology.ru/blog/users-scenarios (дата обращения: 10.12.2023)

7 Резервное копирование. - [Электронный ресурс]. - Режим доступа: https://backupsolution.ru/backup-plan/?ysclid=l4b3itm4ha196679905 (дата обращения: 11.12.2023)

8 Информационная система. - [Электронный ресурс]. - Режим доступа: https://docs.cntd.ru/document/0900006974?ysclid=l4b3lgvmgj187665433 (дата обращения: 16.12.2023)

9 Резервная копия сайта. - [Электронный ресурс]. - Режим доступа: https://1zaicev.ru/kak-sozdat-rezervnuyu-kopiyu-sajta/?ysclid=l4igv9jj2h33094655 (дата обращения: 13.12.2023)

10 Средства автоматизации. - [Электронный ресурс]. - Режим доступа: https://docs.cntd.ru/document/0900083083?ysclid=l4igwrig24723096681 (дата обращения: 20.12.2023)

11 Этапы и виды тестирования сайтов. - [Электронный ресурс]. - Режим доступа: https://polyarix.com/blog/testirovanie-sajta/?ysclid=l4ih2vzzrt25295302 (дата обращения: 17.12.2023)

12 Руководство программиста. - [Электронный ресурс]. - Режим доступа: https://studbooks.net/2236932/informatika/rukovodstvo_programmista?ysclid=l4ih6t76gp393362569 (дата обращения: 1.12.2023)

13 Обратная связь на сайте. - [Электронный ресурс]. - Режим доступа: https://www.imagecms.net/blog/obzory/zachem-nuzhna-obratnaia-sviaz-na-saite (дата обращения: 10.12.2023)

14 Документация React (официальный сайт). - [Электронный ресурс]. - Режим доступа: https://ru.reactjs.org/ (дата обращения: 11.12.2023)

15 Документация Node.js (официальный сайт). - [Электронный ресурс]. - Режим доступа: https://nodejs.org/ru/docs/ (дата обращения: 16.12.2023)

16 Документация PostgreSQL (официальный сайт). - [Электронный ресурс]. - Режим доступа: https://www.postgresql.org/docs/ (дата обращения: 13.12.2023)

17 Джойс Д. Безупречный JavaScript: идеи, стили и лучшие практики. - Санкт-Петербург: Питер, 2015.

18 Официальная документация Express.js. - [Электронный ресурс]. - Режим доступа: https://expressjs.com/ (дата обращения: 17.12.2023)

19 Stack Overflow. - [Электронный ресурс]. - Режим доступа: https://stackoverflow.com/ (дата обращения: 1.12.2023)

20 Tutorial v6.21.0 | React Router. - [Электронный ресурс]. - Режим доступа: https://reactrouter.com/en/main/start/tutorial (дата обращения: 10.12.2023)

# Приложение А
# (обязательное)

## Диаграмма прецедентов (use case)

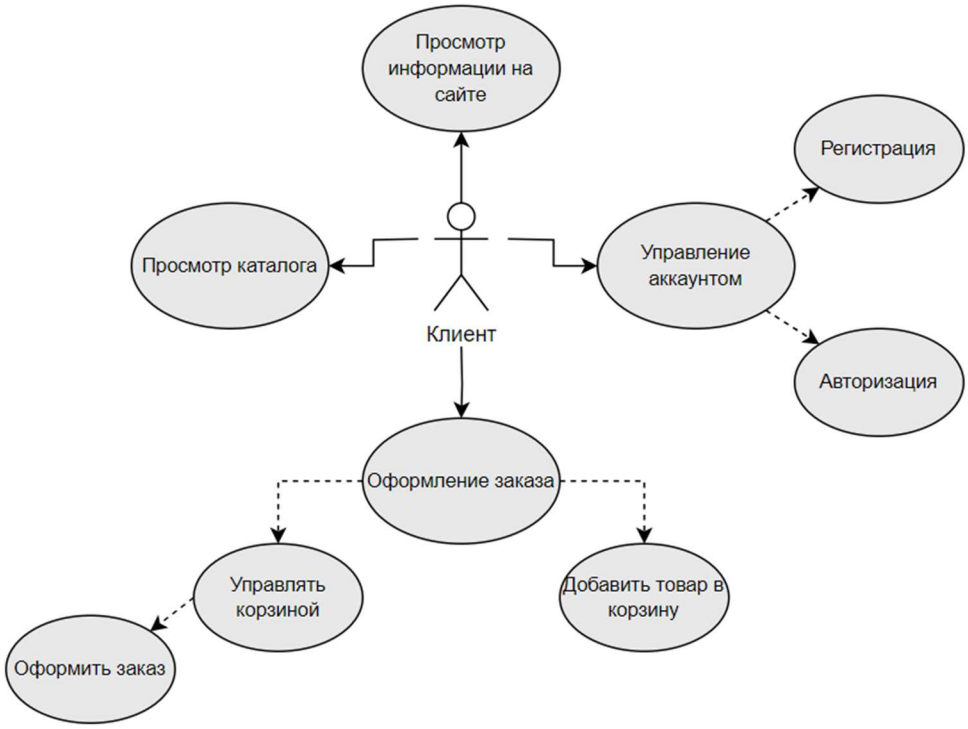


Рисунок А.1 – Диаграмма прецедентов

# Приложение Б
# (обязательное)

## Диаграмма деятельности

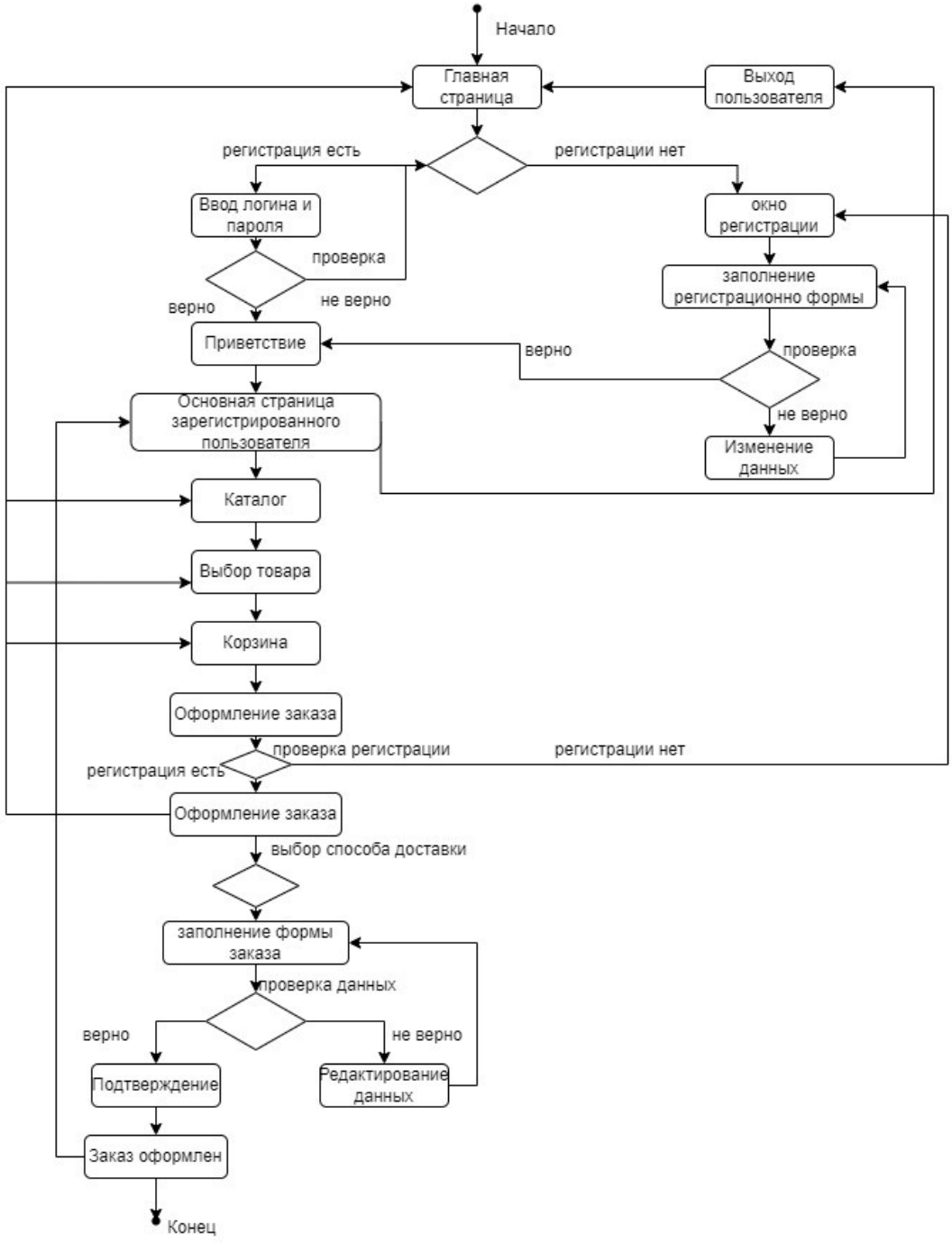


Рисунок Б.1 – Диаграмма деятельности

# Приложение В
# (обязательное)

## Диаграмма классов

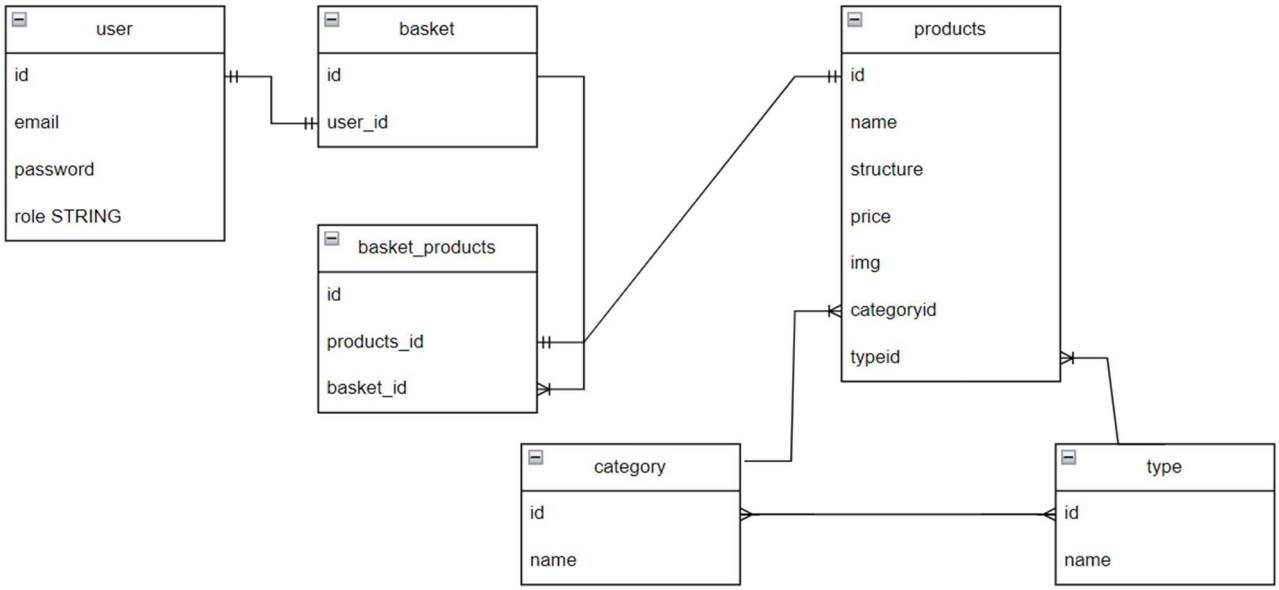


Рисунок В.1 – Диаграмма классов

# Приложение Г
# (обязательное)

## Дизайн веб-приложения


Рисунок Г.1 – Раздел «УТП»


Рисунок Г.2 – Раздел «О нас»

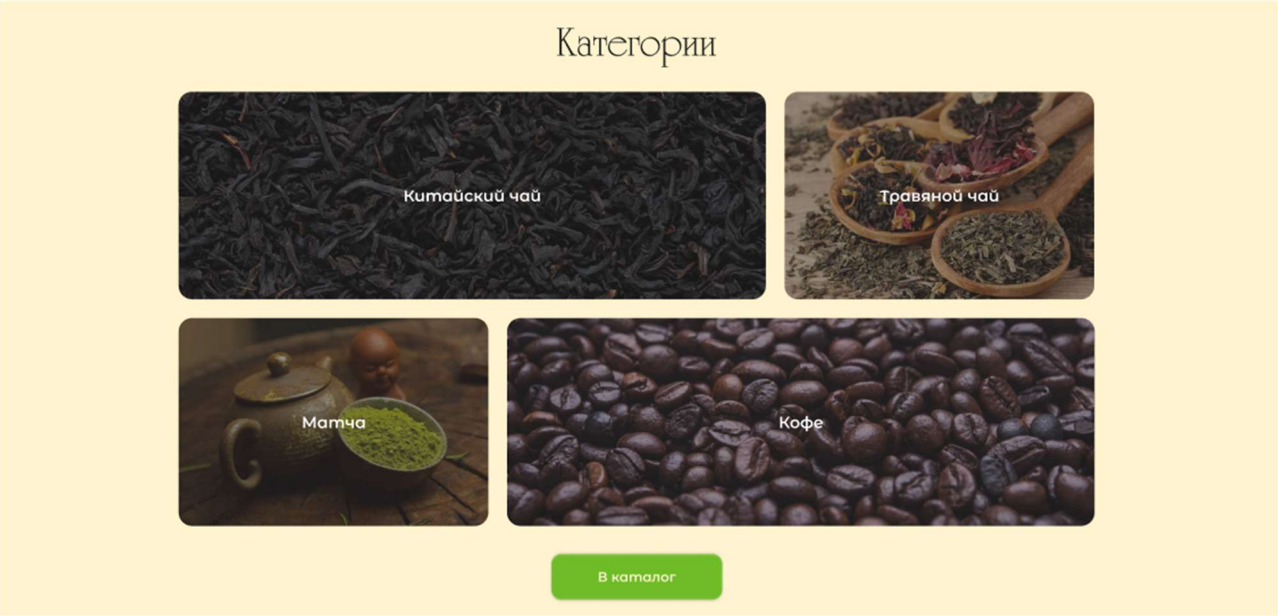


Рисунок Г.3 – Раздел «Категории»


Рисунок Г.4 – Раздел «Доставка и оплата»


Рисунок Г.5 – Раздел «Контакты»

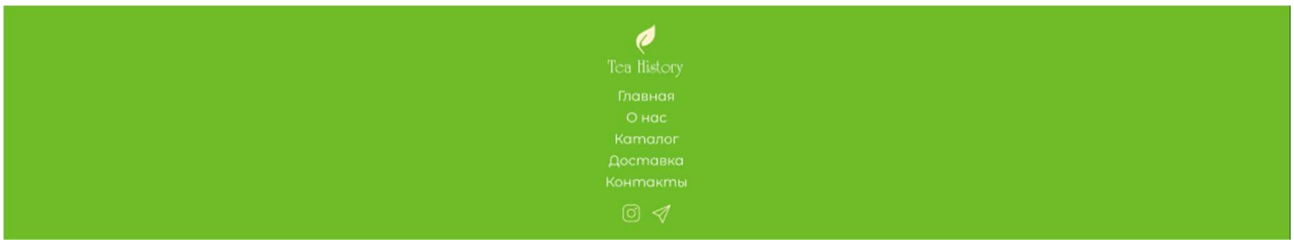


Рисунок Г.6 – Раздел «footer»

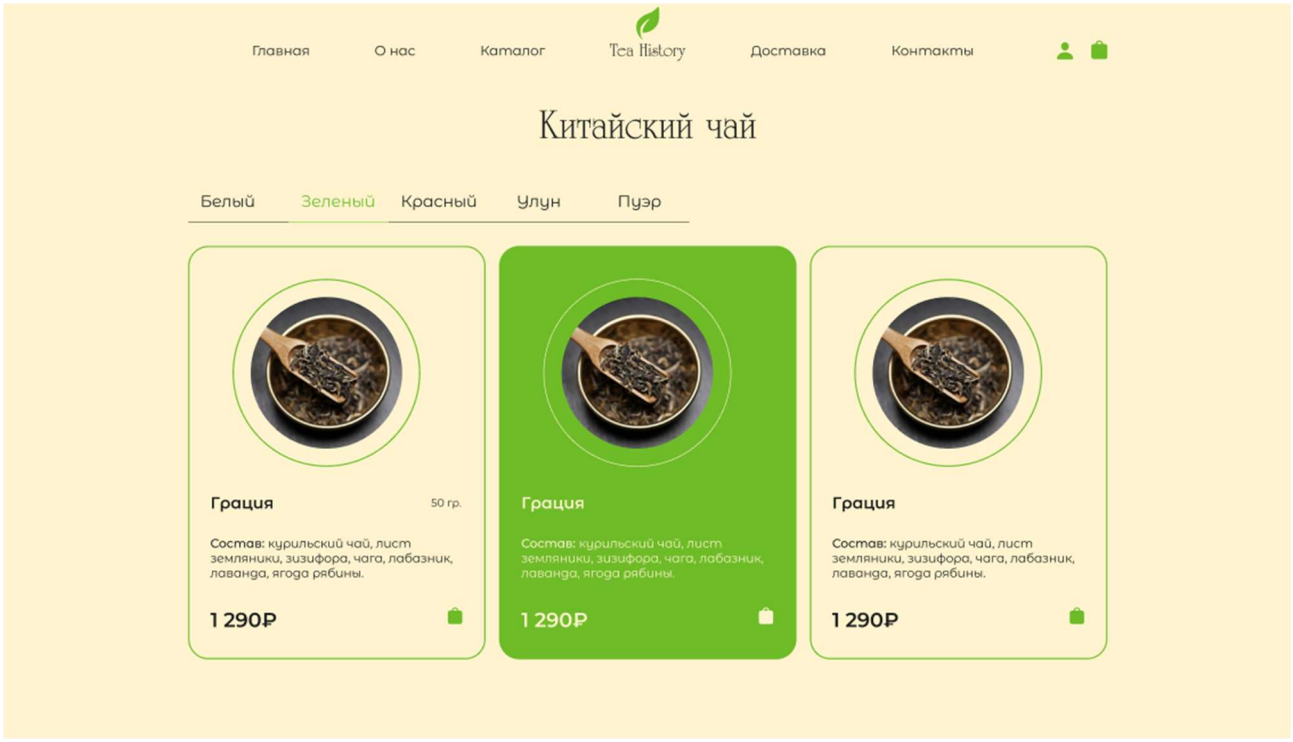


Рисунок Г.7 – Страница «Каталог»

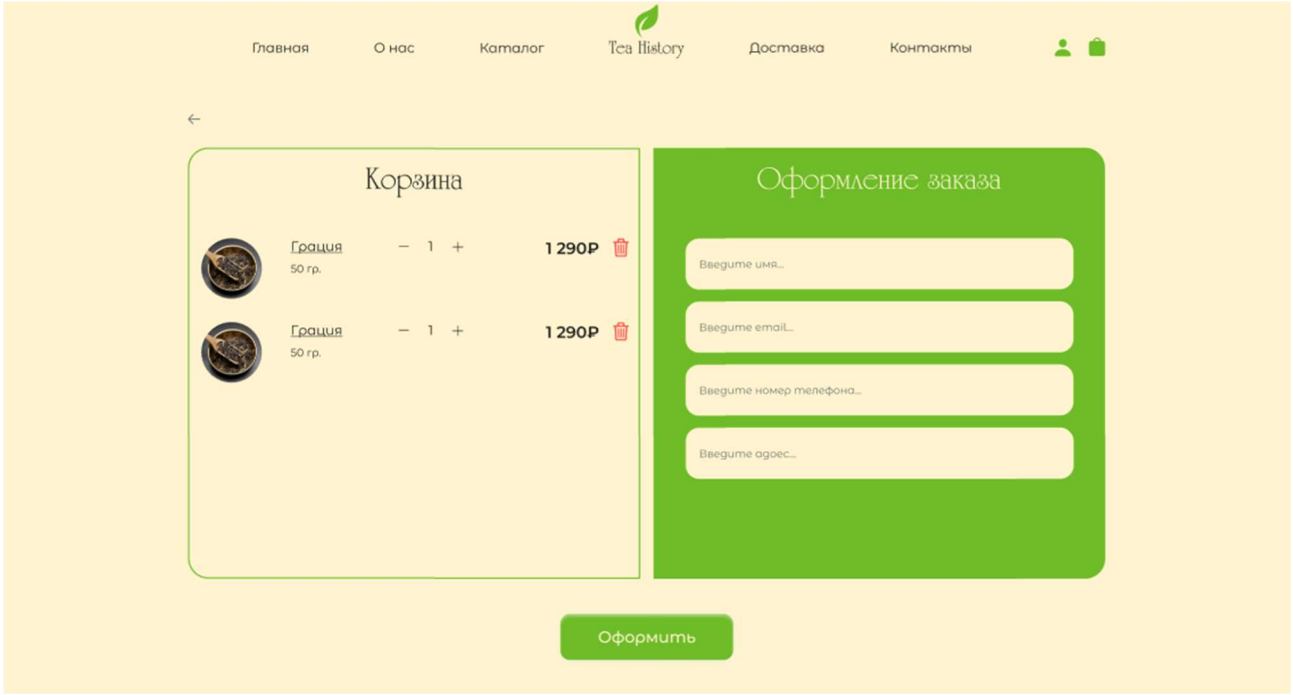


Рисунок Г.8 – Страница «Корзина и оформление заказа»

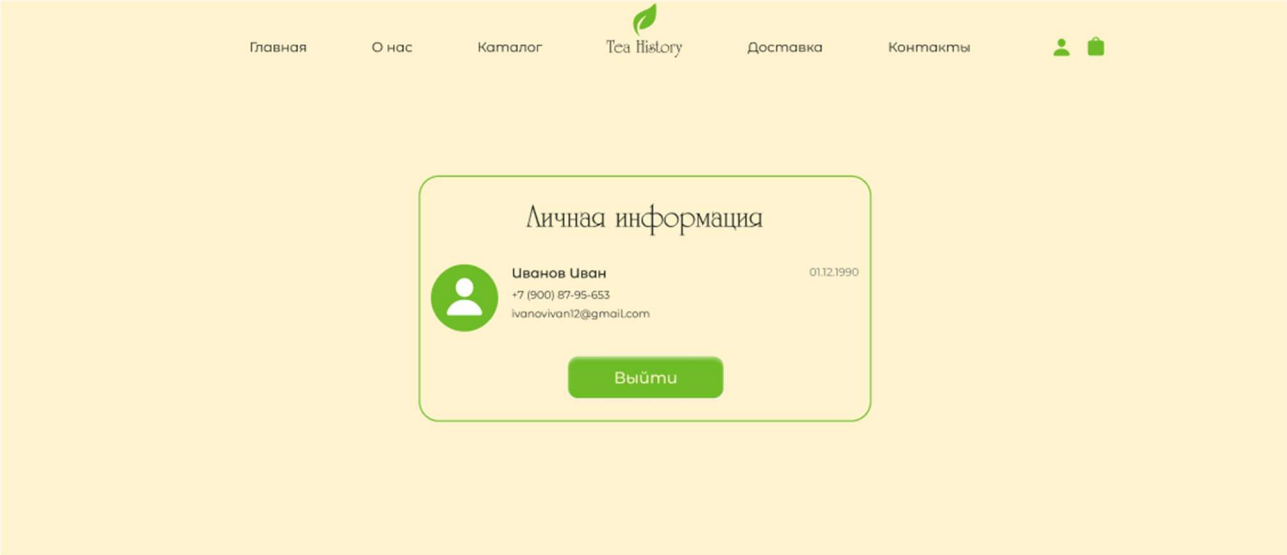


Рисунок Г.9 – Страница «Личный кабинет»


Рисунок Г.10 – Страница «Авторизация»

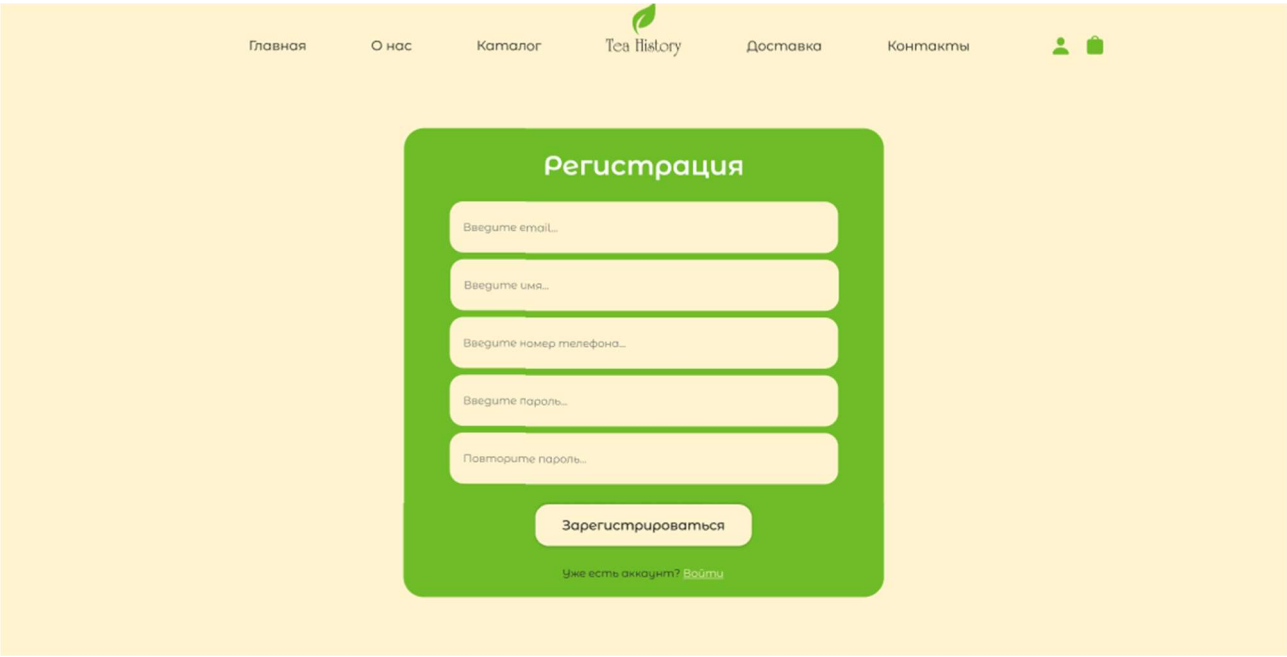


Рисунок Г.11 – Страница «Регистрация»