МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА № 42

КУРСОВАЯ РАБОТА  
ЗАЩИЩЕНА С ОЦЕНКОЙ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

РУКОВОДИТЕЛЬ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| канд. техн. наук, доцент |  |  |  | О. И. Красильникова | |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия | |
| ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОЙ РАБОТЕ | | | | |
| WEB-САЙТ ВЕТЕРИНАРНОЙ КЛИНКИ | | | | |
|  | | | | |
| по дисциплине: WEB-ТЕХНОЛОГИИ | | | | |

РАБОТУ ВЫПОЛНИЛ(А)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4228 |  |  |  | А.Л. Хреева |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Современные информационные технологии оказывают значительное влияние на повседневную жизнь общества, способствуя развитию новых форм коммуникации и обмена информацией. Одним из наиболее заметных проявлений этого процесса стало появление и быстрое распространение социальных сетей. Эти цифровые платформы предоставляют пользователям уникальную возможность для самовыражения, создания и распространения контента, а также взаимодействия с широкой аудиторией в режиме реального времени.

Одной из ключевых особенностей современных социальных сетей является использование коротких текстовых сообщений для быстрого обмена информацией и обсуждения актуальных тем. Такие платформы отличаются высокой динамичностью контента, открытостью и возможностью моментального вовлечения в дискуссии. При этом пользователи получают не только инструмент для личного общения, но и возможность следить за событиями в мире, высказывать мнение по различным вопросам и создавать собственные информационные потоки.

Целью данной курсовой работы является разработка и анализ концепции социальной сети, основанной на принципах краткости и лаконичности сообщений, с акцентом на обеспечение удобства взаимодействия и высокой скорости передачи информации. В работе рассматриваются архитектурные решения, функциональные возможности, а также аспекты обеспечения безопасности и приватности данных пользователей.

1. Дизайн сайта
2. Модель сайта

В процессе разработки социальной сети было спроектировано две навигационные модели сайта: одна для зарегистрированных пользователей, вторая — для пользователей, которые не вошли в систему. Эти модели отличаются составом доступных страниц и функциональностью, что позволяет обеспечить удобное взаимодействие с приложением для обеих категорий пользователей.

Для незарегистрированных пользователей предусмотрена упрощённая модель навигации. С главной страницы они могут перейти на страницы регистрации/входа, просмотреть профили пользователей в ограниченном виде, ознакомиться с лентой твитов без возможности взаимодействия (например, лайков или публикации), а также использовать поиск пользователей. Отсутствует доступ к функциям подписки, создания и редактирования публикаций, уведомлениям и закладкам.

Таким образом, модель направлена на предоставление базового доступа к содержимому платформы и мотивацию зарегистрироваться для получения полного функционала.

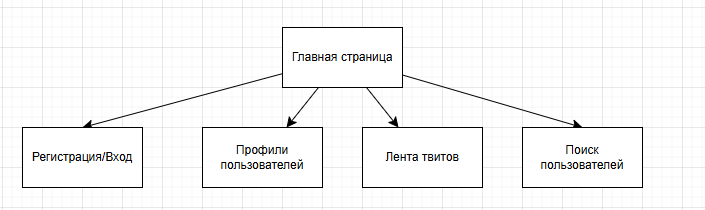


Рисунок 1 – Модель навигации для незарегистрированных пользователей

Для зарегистрированных пользователей сайт предлагает расширенный набор возможностей. С главной страницы они могут попасть на ленту твитов, где доступны функции публикации, редактирования, удаления твитов, добавления закладок и управления подписками. Также доступен раздел уведомлений с возможностью фильтрации событий, профиль пользователя с функцией обновления аватара и биографии, а также поиск пользователей с возможностью подписки.

Такая модель обеспечивает полный функционал социальной сети и удобное взаимодействие между пользователями.

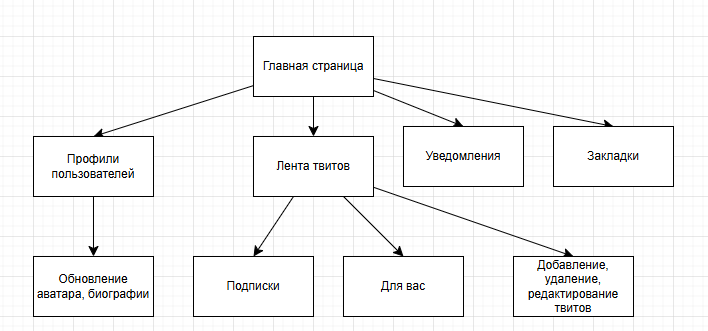


Рисунок 2 – Модель навигации для зарегистрированных пользователей

1. Применение технологии каскадных таблиц стилей

В рамках разработки социальной сети используется технология каскадных таблиц стилей (CSS) для построения современного, удобного и эстетически привлекательного интерфейса. В проекте применяются современные методологии CSS, обеспечивающие модульность, масштабируемость и удобство поддержки кода.

**Основная концепция применения CSS в проекте**

В проекте реализован подход CSS Modules, который широко используется в современных приложениях и позволяет инкапсулировать стили компонентов. Основные преимущества данного подхода:

* Инкапсуляция стилей: каждый компонент имеет собственный CSS-модуль, предотвращающий конфликты имен классов.
* Повышение удобства поддержки: стили логически связаны с компонентами, что упрощает сопровождение кода.
* Увеличение читаемости: разработчик легко определяет, какие стили применяются к конкретному компоненту.
* Оптимизация производительности: загружаются только необходимые стили для текущей страницы.

**Основные элементы стилизации**

В проекте применяются следующие CSS-техники:

* Flexbox для гибкой верстки элементов.
* Фиксированное позиционирование для закрепления элементов интерфейса.
* Медиа-запросы для адаптивного отображения на разных устройствах.
* Переходы и трансформации для создания интерактивных элементов.

**Организация стилей**

Структура стилей организована в соответствии с компонентной архитектурой приложения:

* Глобальные стили.
* Стили макетов.
* Стили основных секций, таких как header, sidebar и другие.
* Стили отдельных элементов интерфейса.

**Преимущества подхода**

Использование CSS Modules обеспечивает:

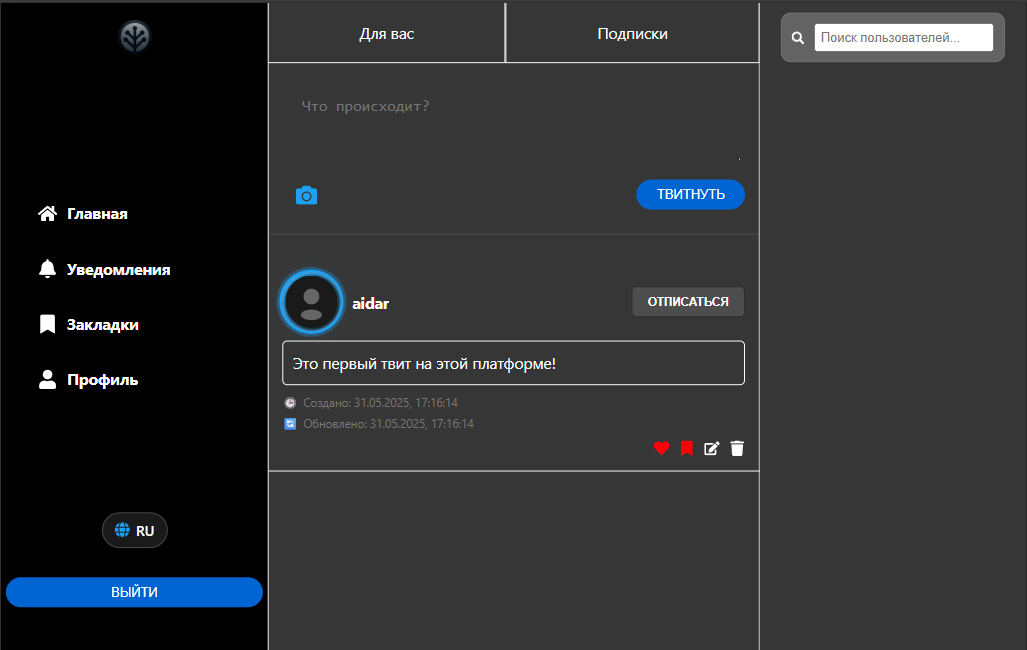
* Масштабируемость: лёгкое добавление новых компонентов без конфликтов стилей.
* Поддерживаемость: логическое разделение упрощает поддержку.
* Производительность: загрузка только нужных стилей повышает быстродействие.
* Удобство разработки: структура кода становится понятной и гибкой.
* Адаптивность: интерфейс корректно отображается на разных устройствах.

Применение технологии CSS в проекте позволяет создавать современный, отзывчивый интерфейс социальной сети. Использование CSS Modules в сочетании с компонентной архитектурой обеспечивает удобство разработки, поддержку масштабируемости и высокую производительность приложения.

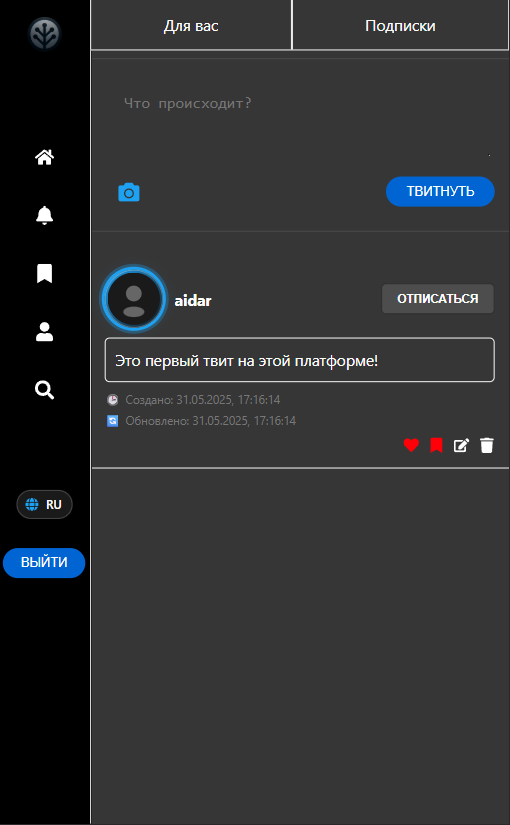
1. Графическое оформление web-страниц

Графическое оформление веб-страниц играет ключевую роль в создании удобного пользовательского интерфейса. Для реализации верстки могут применяться различные подходы, включая фиксированную, резиновую и адаптивную верстку. Ниже приведены примеры каждого типа верстки с подробным описанием расположения элементов на страницах.

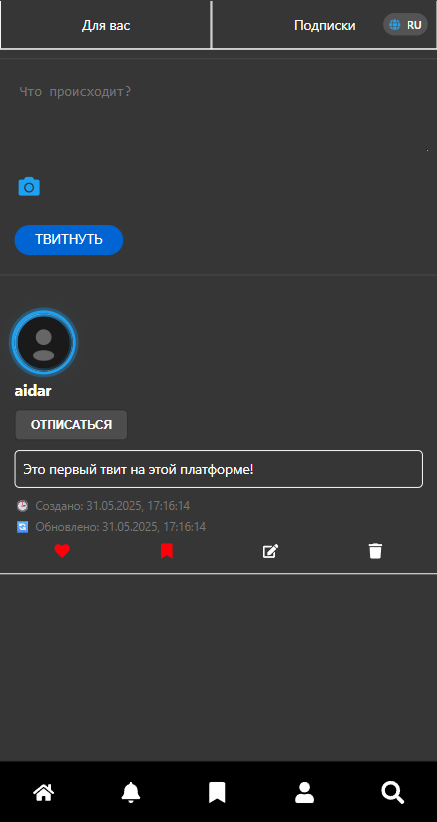
На рисунке 3 изображена главная страница приложения с примером фиксированной верстки. Слева расположено боковое меню с иконками для перехода на разные страницы: главная, уведомления, закладки, профиль. Центральная область содержит основное содержимое – здесь отображается пост пользователя с аватаром, именем и текстом. В правой части экрана находится панель поиска и дополнительной информации. При изменении размера окна браузера элементы интерфейса остаются на своих местах без изменений.

  
Рисунок 3 – Пример фиксированной верстки главной страницы

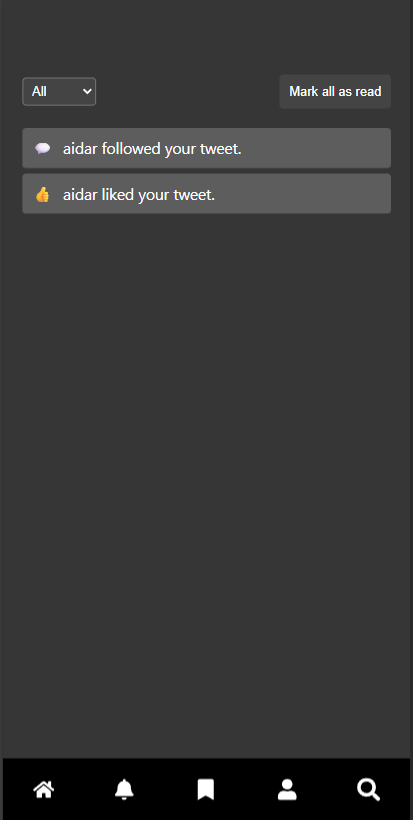
На рисунке 4 представлен пример резиновой верстки. Страница построена так, что ширина элементов изменяется в зависимости от ширины окна браузера. Боковое меню с иконками остаётся слева, но центральная область с постами и кнопками управления (твиты, лайки, комментарии) сжимается или растягивается, сохраняя общую структуру. В правой части отображается панель поиска и дополнительных функций.

  
Рисунок 4 – Пример резиновой верстки

На рисунке 5 показан пример адаптивной верстки. Здесь при изменении ширины окна браузера элементы интерфейса перестраиваются. Боковое меню остаётся слева на широком экране, но может скрываться на мобильных устройствах. Центральная часть с основным контентом (постами) занимает большую часть экрана, а кнопки управления и панель поиска оптимизируются для удобства пользователя.

  
Рисунок 5 – Пример адаптивной верстки

На рисунке 6 продемонстрирован экран уведомлений.

  
Рисунок 6 – Вид страницы с уведомлениями

На рисунке 7 показана страница закладок.

  
Рисунок 7 – Вид страницы с уведомлениями

1. Создание навигационных панелей для сайта

Навигационные панели являются ключевыми элементами интерфейса сайта, обеспечивая удобный доступ пользователей к основным разделам. Реализованы две панели:

Боковая панель (Sidebar) расположена слева и содержит логотип приложения, ссылки на разделы («Главная», «Уведомления», «Закладки», «Профиль», «Поиск»), переключатель языка и кнопку «Выход» для зарегистрированных пользователей. Эта панель зафиксирована, что позволяет пользователю быстро перемещаться между разделами независимо от положения на странице.

Верхняя панель (Header) расположена в верхней части страницы и содержит вкладки для переключения между лентами контента («Для вас», «Подписки») и кнопку для публикации нового сообщения. Эта панель также фиксирована, что обеспечивает удобное переключение между разделами.

1. Web-сценарии сайта на языке Typescript
2. Назначение языка Typescript

Проект реализован на основе React с использованием языка TypeScript, который является расширением JavaScript и обеспечивает строгую типизацию, модульность и высокую надёжность кода. TypeScript применяется для описания типов данных, интерфейсов компонентов и структур данных, что позволяет выявлять ошибки ещё на этапе компиляции и упрощает сопровождение проекта.

Использование TypeScript в сочетании с React даёт следующие преимущества:

* создание масштабируемого и гибкого интерфейса приложения, который легко расширяется новыми компонентами и функционалом;
* предотвращение распространённых ошибок в коде за счёт статической проверки типов;
* повышение читаемости кода и упрощение взаимодействия между членами команды;
* интеграция с современными инструментами и библиотеками React, что ускоряет процесс разработки;
* поддержка новых возможностей ECMAScript с компиляцией в стандартный JavaScript для различных браузеров.

Таким образом, TypeScript делает разработку проекта более структурированной, безопасной и удобной, обеспечивая поддержку современных стандартов и высокое качество пользовательского интерфейса.

1. Организация кода на Typescript в проекте

Проект построен с использованием **React** и **TypeScript**, которые в сочетании обеспечивают надёжную архитектуру приложения, удобство разработки и масштабируемость. Ключевым элементом организации кода являются **TSX-файлы** – файлы, содержащие как разметку компонентов (JSX), так и их логику, при этом с использованием строгой типизации TypeScript.

Каждый компонент приложения хранится в отдельной папке и включает:

* **TSX-файл** с реализацией компонента. Здесь определяется логика, структура JSX-разметки, импорт стилей, хуков и других зависимостей.
* **CSS-модуль** для стилизации компонента, что обеспечивает инкапсуляцию стилей.

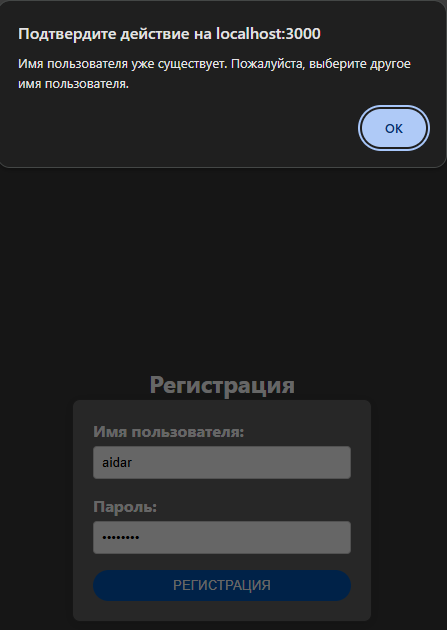
Ключевые особенности устройства TSX-файлов:

* **Инкапсуляция**: компоненты независимы друг от друга, что позволяет переиспользовать их в разных частях приложения.
* **Типизация пропсов и состояний**: используется интерфейсы TypeScript для описания типов данных, которые передаются в компоненты через пропсы или используются в состоянии.
* **Лаконичная структура**: компонент начинается с импорта зависимостей, далее описывается логика (хуки, обработчики событий), затем возвращается JSX-разметка с привязкой классов из CSS-модулей.
* **Поддержка многоязычности (i18n)**: текстовые строки компонентов получают из системы интернационализации, что позволяет легко адаптировать интерфейс под разные языки.

1. Основной функционал веб-приложения

Функционал приложения охватывает все ключевые сценарии взаимодействия пользователя с системой. Ниже представлены описания этих функций и предусмотренные к ним визуализации.

На рисунке 8 изображена ошибка регистрации нового пользователя. Появляется диалоговое окно с сообщением о том, что выбранное имя пользователя уже существует, предлагается выбрать другое имя.

  
Рисунок 8 – Ошибка регистрации: имя пользователя уже существует

На рисунке 9 изображено подтверждение успешной регистрации. Диалоговое окно информирует пользователя, что регистрация прошла успешно и он автоматически вошёл в систему.  
Изображение выглядит как текст, снимок экрана, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 9 – Успешная регистрация пользователя

На рисунке 10 изображён интерфейс профиля пользователя. В нём пользователь изменяет свою личную информацию и готовится сохранить изменения.  
Изображение выглядит как снимок экрана, текст, логотип, круг

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10 – Изменение личной информации профиля

На рисунке 11 изображена форма создания нового твита. Пользователь прикрепил изображения и готовится опубликовать твит.

Изображение выглядит как текст, Мультимедийное программное обеспечение, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 11 – Создание нового твита с прикреплёнными изображениями

На рисунке 12 изображён твит в ленте с текстом и изображениями. В нижней части видны кнопки управления, такие как лайк, закладка, редактирование и удаление.Изображение выглядит как снимок экрана, программное обеспечение, Мультимедийное программное обеспечение, текст

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 12 – Твит с текстом и изображениями в ленте

На рисунке 13 изображено окно редактирования твита. Пользователь может изменить текст и сохранить его.

Изображение выглядит как текст, снимок экрана, мультимедиа, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 13 – Интерфейс редактирования твита

На рисунке 14 изображено диалоговое окно подтверждения удаления твита. Пользователь может подтвердить или отменить удаление.

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 14 – Диалоговое окно удаления твита

На рисунке 15 изображено модальное окно просмотра загруженного изображения.

Изображение выглядит как текст, График, снимок экрана, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 15 – Модальное окно просмотра изображения

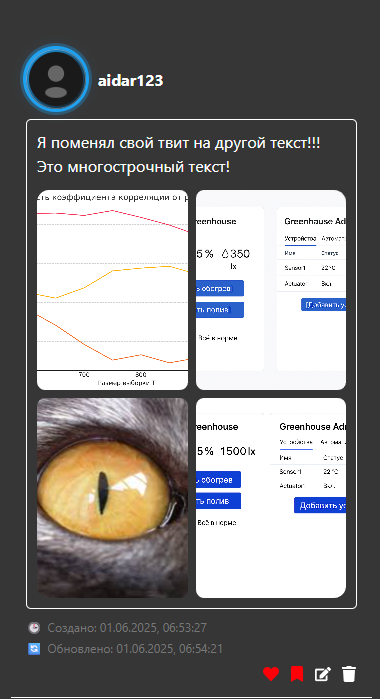


Рисунок 16 – Отредактированный твит

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 17 – Удаленный твит

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 18 – Демонстрация лайка и закладки

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 19 – Демонстрация подписки на пользователя

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 19 – Фильтрация по подпискам

Изображение выглядит как текст, снимок экрана, Шрифт, Графика

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 20 – Страница закладок

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 21 – Страница с уведомлениями

Изображение выглядит как снимок экрана, текст, Шрифт, смартфон

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 22 – Поиск пользователей

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 23 – Поиск пользователей

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 24 – Применение локализации

1. Валидация кода и продвижение сайта
2. Валидация кода сайта

В процессе разработки фронтенд-части приложения активно использовались встроенные инструменты проверки кода, доступные в редакторе Visual Studio Code. Для валидации HTML и CSS файлов применялись расширения **HTMLHint** и **stylelint**, которые автоматически анализируют код и выделяют синтаксические ошибки, предупреждения, а также некорректные или устаревшие конструкции. Эти инструменты позволяют обнаружить ошибки на ранней стадии разработки и повысить качество кода.

Кроме того, важную роль в валидации кода играет использование **TypeScript**, который обеспечивает строгую типизацию данных. Это позволяет не только предотвращать типовые ошибки на этапе написания кода, но и облегчает работу с API и внутренними структурами данных приложения. Благодаря TypeScript была реализована проверка типов, что помогает выявить ошибки ещё до запуска приложения, тем самым повышая его надёжность и безопасность.

Таким образом, сочетание встроенных инструментов Visual Studio Code с системой строгой типизации TypeScript обеспечивает высокое качество кода приложения, соответствие стандартам и улучшает поддержку и масштабируемость проекта.

1. Способы продвижения сайтов

Для успешного продвижения веб-приложения и привлечения новых пользователей применяются различные стратегии, направленные на повышение видимости сайта в поисковых системах, улучшение его репутации и расширение аудитории.

Одним из ключевых инструментов продвижения является поисковая оптимизация (SEO). Для повышения позиции сайта в результатах поисковых систем используются следующие подходы:

* Оптимизация структуры сайта и его навигации;
* Использование корректной семантической разметки HTML;
* Наполнение сайта уникальным и качественным контентом;
* Добавление мета-тегов (title, description, keywords) для каждой страницы;
* Оптимизация изображений (размер, атрибуты alt);
* Повышение скорости загрузки страниц.

Также важным аспектом является техническая оптимизация, включающая обеспечение безопасности соединения (HTTPS), адаптивную верстку для разных устройств и корректную работу на всех современных браузерах.

Для продвижения сайта могут использоваться рекламные кампании (например, в Google Ads или социальных сетях), а также email-маркетинг для привлечения аудитории.

В дополнение к этому, важную роль играет развитие бренда и построение репутации. Регулярное обновление контента, публикация новостей, взаимодействие с аудиторией через соцсети и рассылки помогают удерживать пользователей и увеличивать количество посещений.

Таким образом, комплексный подход, включающий SEO, техническую оптимизацию, рекламу и развитие бренда, позволяет обеспечить эффективное продвижение веб-приложения на рынке.

1. Использованные приемы продвижения сайта

Необходимо разработать программу, которая имитирует покупку товаров или услуг в магазине. Покупатель, у которого есть бонусная карта, кошелек и список покупок,

В рамках SEO-оптимизации социальной платформы был создан комплекс технических и контентных решений для улучшения видимости проекта в поисковых системах и повышения пользовательского опыта.

Базовый HTML-документ был полностью переработан с добавлением необходимых метатегов, включая заголовок, описание и ключевые слова. Документ также обогащен специальными тегами для социальных сетей (Open Graph и Twitter Cards), которые улучшают представление ссылок при публикации их в социальных медиа. Добавлены канонические URL для предотвращения проблем с дублированием контента.

Был разработан файл robots.txt с детальными директивами для поисковых роботов, указывающими какие страницы нужно индексировать, а какие следует исключить из индекса. Создана динамическая карта сайта sitemap.xml, содержащая информацию о всех публичных страницах приложения с указанием приоритета и частоты обновления каждой страницы.

Важным элементом оптимизации стало внедрение структурированных данных Schema.org через JSON-LD. Созданы утилитарные функции для генерации разметки организации, веб-сайта, профилей пользователей и публикаций. Эта разметка позволяет поисковым системам лучше понимать контент сайта и показывать обогащенные сниппеты в результатах поиска.

Для неавторизованных пользователей разработана полноценная посадочная страница с семантически правильной структурой HTML. Страница содержит разделы преимуществ сервиса, отзывы пользователей и призывы к действию. Ее дизайн адаптирован для всех типов устройств и оптимизирован для максимальной конверсии посетителей в зарегистрированных пользователей

Система маршрутизации приложения модифицирована для поддержки SEO-дружественных URL и правильной обработки состояния авторизации. Для авторизованных пользователей показывается основной интерфейс приложения, а для неавторизованных - оптимизированная для SEO страница.

Реализована полная поддержка многоязычности с использованием i18next. Переводы включают контент посадочной страницы и все элементы интерфейса на английском и русском языках. Это не только улучшает пользовательский опыт, но и повышает релевантность сайта для поисковых запросов на разных языках.

Все эти меры в совокупности формируют комплексный подход к SEO-оптимизации, обеспечивающий лучшую видимость проекта в поисковых системах, повышение органического трафика и улучшение пользовательского опыта на всех типах устройств.

Заключение

В рамках данной курсовой работы было разработано приложение для управления и выполнения операций с покупкой товаров в магазине, демонстрирующее применение основных принципов объектно-ориентированного программирования (ООП) с использованием языка программирования C#. Работа включала в себя разработку пользовательского интерфейса на основе Windows Forms, а также реализацию логики для управления корзиной, взвешивания товаров и проведения платежей. Приложение позволяет пользователю выполнять различные операции, такие как добавление и удаление товаров из корзины, взвешивание товаров перед покупкой, а также оплату покупок наличными, дебетовой картой или бонусными баллами.

В дополнение к основной функциональности, приложение включает в себя элементы пользовательского интерфейса, разработанные в Visual Studio, которые позволяют пользователю взаимодействовать с различными классами и их методами напрямую. Это включает в себя не только ввод данных и отображение результатов, но и дополнительные возможности, такие как управление балансом и корзиной покупок, что делает приложение не только функциональным, но и удобным для использования.

Были исследованы и успешно применены ключевые концепции C#, такие как создание классов, наследование, полиморфизм, инкапсуляция, а также работа с пользовательским интерфейсом при помощи Windows Forms. Реализация функционала приложения включала создание и использование различных классов и интерфейсов, демонстрируя возможности языка в работе с абстрактными типами данных и их конкретными реализациями.

Были использованы следующие отношения между классами и интерфейсами: наследование, реализация, ассоциация, агрегация, композиция. Применение принципов SOLID позволило создать четкую и понятную структуру кода.

В результате данная курсовая работа позволила углубленно изучить и применить возможности C# в контексте создания приложения Windows Forms. Реализация интерфейсов и абстрактных классов, а также построение эффективного графического интерфейса может быть полезно для дальнейшего изучения и разработки приложений на платформе .NET, а также демонстрирует глубокое понимание объектно-ориентированного подхода в программировании.

Список использованных источников

1. AppMaster. Архитектурные шаблоны: объяснение MVC, MVP и MVVM. URL: <https://appmaster.io/ru/blog/arkhitekturnye-shablony-mvc-mvp-i-mvvm> (дата обращения 10.05.2024).
2. Citforum. Обзор паттернов проектирования. URL: <http://citforum.ru/SE/project/pattern/> (дата обращения 10.05.2024).
3. Linode. SOLID Principles in Programming: A Comprehensive Guide. URL: <https://www.linode.com/docs/guides/solid-principles-programming/> (дата обращения 12.05.2024).
4. Microsoft. Windows Forms documentation. URL: <https://learn.microsoft.com/en-us/dotnet/desktop/winforms/?view=netdesktop-8.0> (дата обращения 15.05.2024).
5. Vscode. Модульное тестирование в Visual Studio. URL: <https://vscode.ru/prog-lessons/modulnoe-testirovanie-v-visual-studio.html> (дата обращения 15.05.2024).
6. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж. Приемы объектно-ориентированного проектирования. Паттерны проектирования. — СПб: Питер, 2015.

Приложение А

Код формы ShopForm