Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(МОСКОВСКИЙ ПОЛИТЕХ)

ДОПУЩЕНА К ЗАЩИТЕ

Заведующий кафедрой

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Е.А. Пухова/

Руководитель образовательной программы

\_\_\_\_\_\_\_\_\_\_\_\_\_ / М.В. Даньшина /

**«Проектирование и разработка веб-сервиса для публикации и обмена коллекционных предметов»**

Выпускная квалификационная работа

по направлению 09.03.01 Информатика и вычислительная техника

Образовательная программа (профиль) «Веб-технологии»

Студент: \_\_\_\_\_\_\_\_\_\_\_\_ / Сурайкин И. С., 191-322 /

*подпись*  *ФИО, группа*

Руководитель ВКР: \_\_\_\_\_\_\_\_\_\_\_ / Васильев Д. Б., \_\_\_\_\_\_\_\_\_\_\_ /

*подпись*  *ФИО, уч. звание и степень*

Москва, 2023

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное автономное образовательное учреждение высшего образования

«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

ЗАДАНИЕ НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ

по направлению 09.03.01 Информатика и вычислительная техника

Образовательная программа (профиль) «Веб-технологии»

|  |  |
| --- | --- |
| **ТЕМА ВКР** | Проектирование и разработка веб-сервиса для публикации и обмена коллекционных предметов |
| **ПРАКТИЧЕСКИЙ РЕЗУЛЬТАТ** | |
| Назначение | Автоматизация процесса обмена и просмотра коллекционных предметов. |
| Основные функции | * авторизация пользователей; * просмотр опубликованных предметов; * создание коллекции предметов; * редактирование коллекции предметов; * удаление коллекции предметов; * создание предложений обмена; * просмотр предложений обмена; * добавление коллекционных предметов; * редактирование коллекции предметов; * удаление коллекции предметов; * чат с другими пользователями. |
| Используемые технологии и платформы | Vue.js, Django, JavaScript, Python, SQL, Figma. |
| **ВЫПОЛНЕНИЕ РАБОТЫ** | |
| Решаемые задачи | * провести анализ предметной области; * провести анализ существующих решений; * рассмотреть и проанализировать технологии, требуемые для решения; * спроектировать бизнес-процессы; * составить техническое задание; * разработать графический интерфейс; * разработать серверную часть сервиса; * разработать базу данных; * провести тестирование веб-сервиса; * разработать план дальнейшего продвижения веб-сервиса. |
| Состав технической документации | * техническое задание; |
| Состав графической части | Таблицы, диаграммы бизнес-процессов, схемы базы данных, эскизы и макеты экранных форм. |

**ПЛАН РАБОТЫ НАД ВКР**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ЗАДАЧИ** | **НЕДЕЛИ** | | | | | | | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| Провести анализ предметной области |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Провести анализ существующих решений |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Рассмотреть и проанализировать технологии, требуемые для решения |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Спроектировать бизнес-процессы |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Составить техническое задание |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Разработать базу данных |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Разработать графический интерфейс |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Разработать серверную часть сервиса |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Провести тестирование веб-сервиса |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Разработать план дальнейшего продвижения веб-сервиса |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

РУКОВОДИТЕЛЬ ОП:

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2023, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / \_ Даньшина Марина Владимировна, старший преподаватель /  
 *подпись ФИО, уч. звание и степень*

РУКОВОДИТЕЛЬ ВКР:

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2023, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /       Васильев Денис Борисович, старший преподаватель /  
 *подпись ФИО, уч. звание и степень*

СТУДЕНТ:

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2023, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /     Сурайкин Илья Сергеевич, 191-322     /  
 *подпись*  *ФИО, группа*

**АННОТАЦИЯ**

Наименование работы: проектирование и разработка веб-сервиса для публикации и обмена коллекционных предметов.

Цель работы: спроектировать и разработать веб-сервис для автоматизации процесса обмена коллекционными предметами между пользователями сервиса, а также публикации личной коллекции.

Задачи для достижения цели: провести анализ предметной области, провести анализ существующих решений, изучить современные тенденции в области веб-разработки и выбрать подходящие технологии, спроектировать базу данных, спроектировать интерфейсы и дизайн сервиса, разработать серверную и пользовательскую части сервиса, протестировать созданный веб-сервис, разработать план дальнейшего продвижения веб-сервиса.

Объект и предмет исследования: объект – процесс обмена коллекционными предметами между пользователями веб-сервиса, а также процесс публикации личной коллекции, предмет – проектирование и разработка веб-сервиса.

Работа состоит из 3 глав, Введения, Заключения и Приложений. Общий объем работы \* страниц, из них … приложений на … листах. В работу включены 19 рисунков, 2 таблицы. Библиографический список состоит из …. источников, из них …. печатных источников, …. интернет-источников.

Во Введении описаны цель и задачи работы, объект и предмет исследования, его актуальность, новизна и практическая значимость.

В Первой главе описывается предметная область и технологии.

Во Второй главе представлена разработка техническая реализация.

В Третьей главе раскрыты технические аспекты внедрения, тестирования и надежности системы.

В Заключении представлены выводы по поставленным задачам

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 7](#_Toc134744529)

[1. ПРЕДМЕТНАЯ ОБЛАСТЬ И ТЕХНОЛОГИИ. 9](#_Toc134744530)

[1.1 Описание предметной области 9](#_Toc134744531)

[1.2 Обоснование выбора языков программирования 10](#_Toc134744532)

[1.3 Анализ аналогов 12](#_Toc134744533)

[1.4 Бизнес-процессы 20](#_Toc134744534)

[1.5 Техническое задание 26](#_Toc134744535)

[1.6 Выводы 28](#_Toc134744536)

[2. ТЕХНИЧЕСКАЯ РЕАЛИЗАЦИЯ 29](#_Toc134744537)

[2.1 Функции сайта. 29](#_Toc134744539)

[2.2 Создание графического пользовательского интерфейса. 31](#_Toc134744540)

[2.3 База данных сайта 45](#_Toc134744541)

[2.4 Основные алгоритмы работы программы 51](#_Toc134744542)

[2.5 Выводы 54](#_Toc134744543)

[3. ВВЕДЕНИЕ В ЭКСПЛУАТАЦИЮ 55](#_Toc134744544)

[3.1 Тестирование 55](#_Toc134744546)

[3.2 Аспекты отказоустойчивости и надежности. 60](#_Toc134744547)

[3.3 Анализ внедрения продукта и оценка результатов 62](#_Toc134744548)

[3.4 Планы развития 63](#_Toc134744549)

[3.5 Выводы 66](#_Toc134744550)

[СПИСОК ЛИТЕРАТУРЫ 67](#_Toc134744551)

[ПРИЛОЖЕНИЯ 72](#_Toc134744552)

[Приложение А 72](#_Toc134744553)

ВВЕДЕНИЕ

В современном мире все большее количество людей увлекаются коллекционированием различных предметов, будь то монеты, книги, марки или игрушки. Однако, в процессе коллекционирования возникает необходимость обмена предметами между коллекционерами, что может быть затруднительным из-за географических, временных и иных ограничений.

Данная работа актуальна, так как существующие сервисы для коллекционеров не всегда позволяют обеспечить удобный и быстрый обмен предметами, а также не всегда удобны для поиска и публикации коллекций. Разработка нового веб-сервиса, который учитывает все требования и пожелания пользователей, будет способствовать развитию коллекционирования и обмену предметами, а также увеличит число увлеченных этим хобби людей в интернете.

Существующие сервисы для коллекционеров, как правило, не предоставляют полный спектр возможностей для удобного обмена и публикации коллекций. Некоторые из них имеют ограничения по количеству предметов или по возможностям обмена, другие не имеют достаточно развитого сообщества пользователей. Веб-сервис, разрабатываемый в данной работе, позволит решить эти проблемы.

Для решения данной проблемы возникает необходимость создания веб-сервиса, который позволит коллекционерам обмениваться предметами через интернет, а также публиковать свои коллекции и находить единомышленников.

Целью данной работы является создание удобного и функционального веб-сервиса для обмена и публикации коллекционных предметов, который сможет удовлетворить потребности и пожелания пользователей. Разработка данного сервиса позволит увеличить число людей, занимающихся коллекционированием и обменом предметами, а также облегчит процесс обмена и публикации коллекций.

Для этого необходимо решить следующие задачи:

* провести анализ предметной области;
* провести анализ существующих решений;
* рассмотреть и проанализировать технологии, требуемые для решения;
* спроектировать бизнес-процессы
* составить техническое задание;
* разработать графический интерфейс;
* разработать серверную часть сервиса;
* разработать базу данных;
* провести тестирование веб-сервиса;
* разработать план дальнейшего продвижения веб-сервиса.

1. ПРЕДМЕТНАЯ ОБЛАСТЬ И ТЕХНОЛОГИИ.

Описание предметной области

Предметная область выпускной квалификационной работы - коллекционирование и обмен коллекционными предметами. Коллекционирование является одним из наиболее популярных хобби, которое занимаются многие люди во всем мире [8]. Коллекционирование может быть связано с различными предметами, например, монетами, банкнотами, почтовыми марками, значками, журналами и т.д.

Одним из главных аспектов коллекционирования является возможность обмена предметами между коллекционерами. Это позволяет получить редкие или уникальные предметы, которые не всегда доступны для покупки или находятся за пределами региона, где живет коллекционер.

В последнее время коллекционирование стало более доступным благодаря развитию Интернета и возможности публикации и обмена предметами через специальные веб-сервисы. Однако существующие сервисы часто имеют ограничения по количеству предметов или по возможностям обмена, что ограничивает возможности коллекционеров.

Веб-сервис будет очень удобен для коллекционеров, которые ведут учет своих предметов и хотят поделиться своими коллекциями с другими участниками сервиса. Благодаря сервису, коллекционеры смогут создавать профиль, на котором будут отображаться их коллекции, а также информация обо всех предметах в этих коллекциях.

Для удобства использования сервиса пользователям предоставлен широкий выбор категорий для различных типов коллекционных предметов, таких как монеты, марки, значки, видеоигры, журналы и т.д. Каждый предмет можно будет описать с помощью текста и изображений, а также других характеристик, это позволит пользователям получить полную информацию о каждом предмете.

Обоснование выбора языков программирования

Vue.js — это популярная среда JavaScript с открытым исходным кодом для создания пользовательских интерфейсов (UI) и одностраничных приложений [40]. Vue.js, как и другие языки программирования, имеет свои уникальные особенности и преимущества, которые делают его популярным среди разработчиков [21]. В этом сравнительном анализе мы рассмотрим некоторые популярные языки программирования, их особенности и сравним их с Vue.js [19].

Хотя у каждого фреймворка есть свои уникальные преимущества, у Vue.js есть несколько конкретных причин, по которым его можно считать лучше других фреймворков в определенных аспектах:

Простота в освоении и использовании: Vue.js — один из самых простых фреймворков для изучения и использования. Его синтаксис похож на HTML, что упрощает его понимание и начало работы. Vue.js также имеет отличную документацию, которая помогает разработчикам быстро освоиться. По сравнению с Angular, начать работу с Vue.js намного проще.

Гибкость: Vue.js — очень гибкий фреймворк, что делает его отличным выбором для небольших и крупных приложений. Его можно использовать как облегченную библиотеку для создания небольших приложений или как полнофункциональную платформу для создания крупномасштабных приложений. Vue.js также позволяет разработчикам легко интегрироваться с существующими проектами, что делает его отличным выбором для создания новых функций поверх существующих приложений.

Производительность: Vue.js разработан с учетом производительности. Он использует виртуальный DOM, что позволяет быстро и эффективно обновлять пользовательский интерфейс. Это приводит к более высокой производительности приложения по сравнению с другими фреймворками.

Интеграция: Vue.js можно легко интегрировать с другими библиотеками и фреймворками. Это означает, что разработчики могут использовать свои любимые библиотеки и инструменты с Vue.js, что делает его отличным выбором для создания пользовательских приложений. Он также имеет официальную поддержку TypeScript, который является популярным статически типизированным языком для создания веб-приложений [24].

Сообщество:

Vue.js имеет растущее сообщество разработчиков, а это означает, что у разработчиков есть множество ресурсов, на которых они могут учиться. Он также имеет растущую экосистему сторонних библиотек и инструментов, что упрощает разработчикам создание пользовательских приложений [22].

В заключение, Vue.js — это мощный и гибкий фреймворк, который имеет ряд преимуществ перед другими фреймворками. Его простота использования, гибкость, производительность, интеграция и сообщество делают его отличным для создания клиентской части веб-сервиса.

Для разработки серверной части веб-сервиса выбран фреймворк Django [23]. Список некоторых причин, по которым он считается лучше других фреймворков:

* Удобный ORM (Object-Relational Mapping) для работы с базами данных, который позволяет использовать базы данных различных типов, в том числе PostgreSQL, MySQL, Oracle и другие. Django ORM обеспечивает удобный интерфейс для работы с данными, а также инструменты для создания миграций и работы с индексами [25, 26, 27].
* Встроенная система аутентификации и авторизации, которая обеспечивает безопасность приложения, позволяя разработчикам легко контролировать доступ к ресурсам и управлять пользователями.
* Удобная работа с формами и валидацией данных, благодаря чему можно легко создавать и обрабатывать HTML-формы [28].
* Широкий набор готовых пакетов и библиотек, таких как Django REST framework для создания RESTful API, Django Debug Toolbar для отладки и оптимизации приложения, Django Celery для асинхронной обработки задач и другие.
* Встроенная система администрирования, которая позволяет быстро создавать и управлять административными интерфейсами для приложения, без необходимости писать дополнительный код.
* Широкий выбор плагинов и расширений, доступных для Django, позволяет создавать приложения любой сложности и масштаба.
* Обширное сообщество разработчиков, которое обеспечивает поддержку и развитие фреймворка, а также активное обсуждение новых идей и возможностей.
* Легко настраиваемый и гибкий фреймворк, который позволяет быстро создавать и изменять приложения в соответствии с потребностями клиента.

Таким образом, на основе проведенного анализа преимуществ фреймворка Django и сравнения его с другими решениями, можно сделать вывод о том, что данный фреймворк является оптимальным выбором для разработки серверной части веб-сервиса для публикации и обмена коллекционных предметов.

Анализ аналогов

Анализ аналогов является важным шагом в процессе разработки любого нового продукта или системы, поскольку он позволяет тщательно изучить существующие решения и дает информацию, которая может помочь в разработке новых подходов. В контексте дизайна веб-сайтов анализ аналогов включает изучение других веб-сайтов в той же или аналогичных отраслях для выявления передового опыта, общих тенденций и потенциальных областей для улучшения.

Изучив ряд веб-сайтов, имеющих отношение к целям проекта, мы определим ключевые функции, элементы дизайна и функциональные возможности, которые являются эффективными и широко распространенными. Кроме того, критически оценим эти аналоги, чтобы определить области, которые можно улучшить, и предложить идеи, которые помогут разработать более эффективный веб-сайт.

При анализе аналогов большое внимание следует уделить дизайну [19]. Дизайн сайта является очень важным аспектом веб-разработки, потому что он оказывает огромное влияние на первое впечатление пользователя и на его дальнейшее поведение на сайте.

Для анализа рассмотрим три сайта со схожим функционалом и тематикой: mycollection.site, mycoll.ru, Avito. Для начала кратко опишем рассматриваемые сайты.

Mycollection.site:

Сайт mycollection.site представляет собой онлайн-магазин, специализирующийся на продаже товаров для коллекционирования. На сайте можно найти широкий ассортимент коллекционных предметов, таких как монеты, банкноты, марки, магниты, значки, футболки с логотипами и многое другое. Кроме того, на сайте представлены материалы для коллекционеров, такие как каталоги, статьи и новости из мира коллекционирования.

Для удобства посетителей сайт разделен на категории, которые можно выбирать из меню. Каждый товар снабжен фотографиями и описанием, что помогает сделать выбор и совершить покупку.

На сайте представлен широкий выбор коллекционных предметов, что может удовлетворить потребности разных коллекционеров и помочь найти нужный товар.

Онлайн-консультации. Покупатели могут связаться с администрацией сайта через онлайн-форму или по электронной почте для получения помощи и консультаций.

Администрация сайта старается предоставить качественное обслуживание, быстро реагировать на запросы покупателей и помогать решать возможные проблемы.

Сайт регулярно обновляется новыми товарами и материалами для коллекционеров, что позволяет посетителям быть в курсе последних новинок в мире коллекционирования.

Фрагмент дизайна главной страницы сайта представлен на 0. Рассмотрим дизайн сайта подробнее:

* Неэффективное использование пространства: главная страница сайта занимает только небольшую часть экрана, в то время как большая часть экрана остается незаполненной.
* Неудобное меню: меню на сайте находится в верхней части страницы и не всегда является удобным для использования.
* Отсутствие современных визуальных элементов: сайт не использует современные визуальные элементы, такие как анимация или параллакс-эффекты.
* Не интуитивный дизайн: некоторые элементы дизайна на сайте могут быть не интуитивными для пользователя. Например, на странице с предметом коллекции, кнопка для добавления этого предмета в свою коллекцию находится вверху страницы, что может быть неочевидным для пользователя.

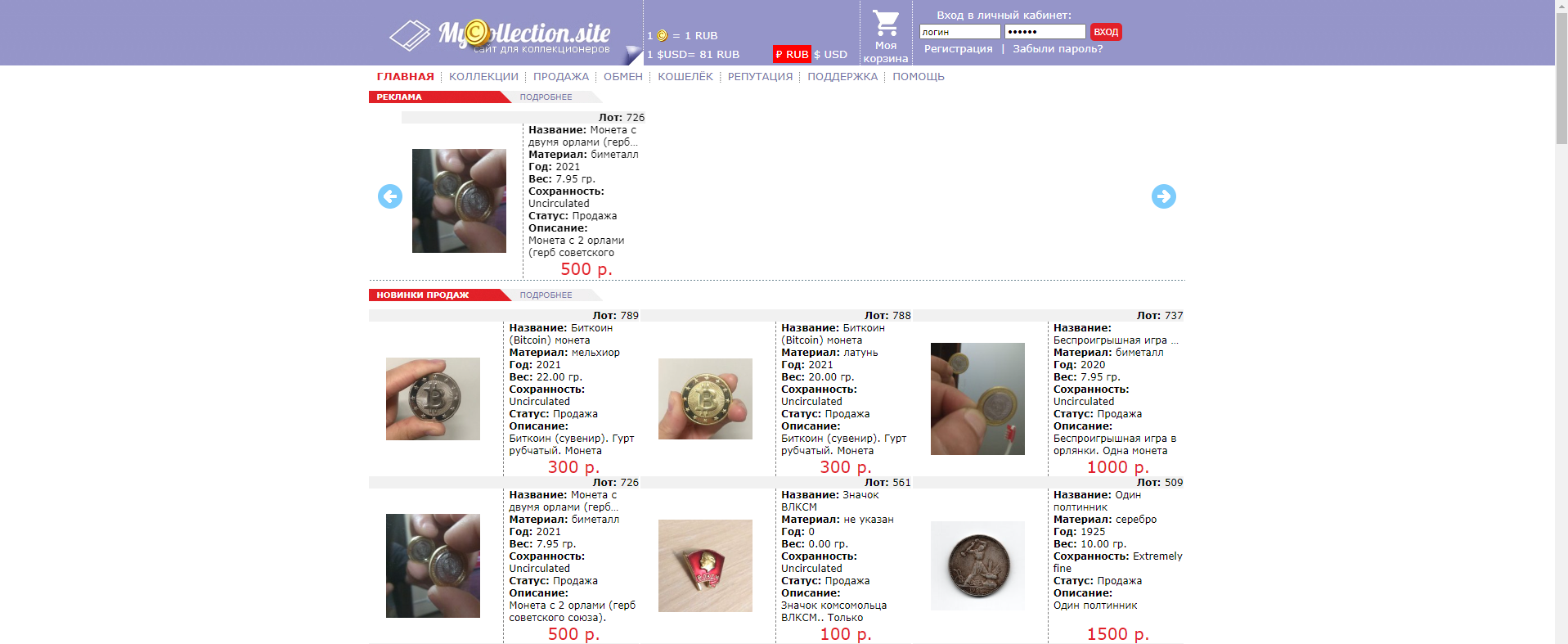


Рисунок 1 - Главная страница сайта Mycollection.site

Можно выделить следующие недостатки сайта:

Ограниченный ассортимент. Хотя сайт предлагает широкий ассортимент коллекционных предметов, он может не иметь всего, что ищет конкретный покупатель.

Не всегда доступны актуальные товары. Товары на сайте могут быть уже проданы, но не обновлены в онлайн-магазине, что может привести к разочарованию у покупателей.

Ограниченный интерфейс. Возможно, сайт может показаться не очень удобным для некоторых пользователей, особенно если они не знакомы с онлайн-магазинами.

Нет подробных описаний. Некоторые товары на сайте могут иметь неполные или недостаточно подробные описания, что может затруднить выбор и понимание покупателем.

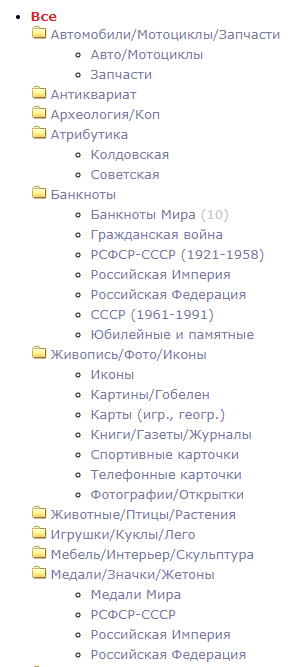


Рисунок 2 - Разделение предметов по категориям.

К другим недостаткам можно отнести, то что сайт не имеет функциональных возможностей, кроме просмотра контента.

Сайта mycoll.ru:

Сайт mycoll.ru представляет собой онлайн-коллекцию, на которой пользователи могут просматривать и наслаждаться коллекцией банкнот, монет и других артефактов.

К преимуществам данного сайта можно отнести интуитивно понятный дизайн. Предметы на сайте разделены на категории, на главной странице можно легко найти интересующую категорию. Фрагмент главной страницы показан на рисунке 3.

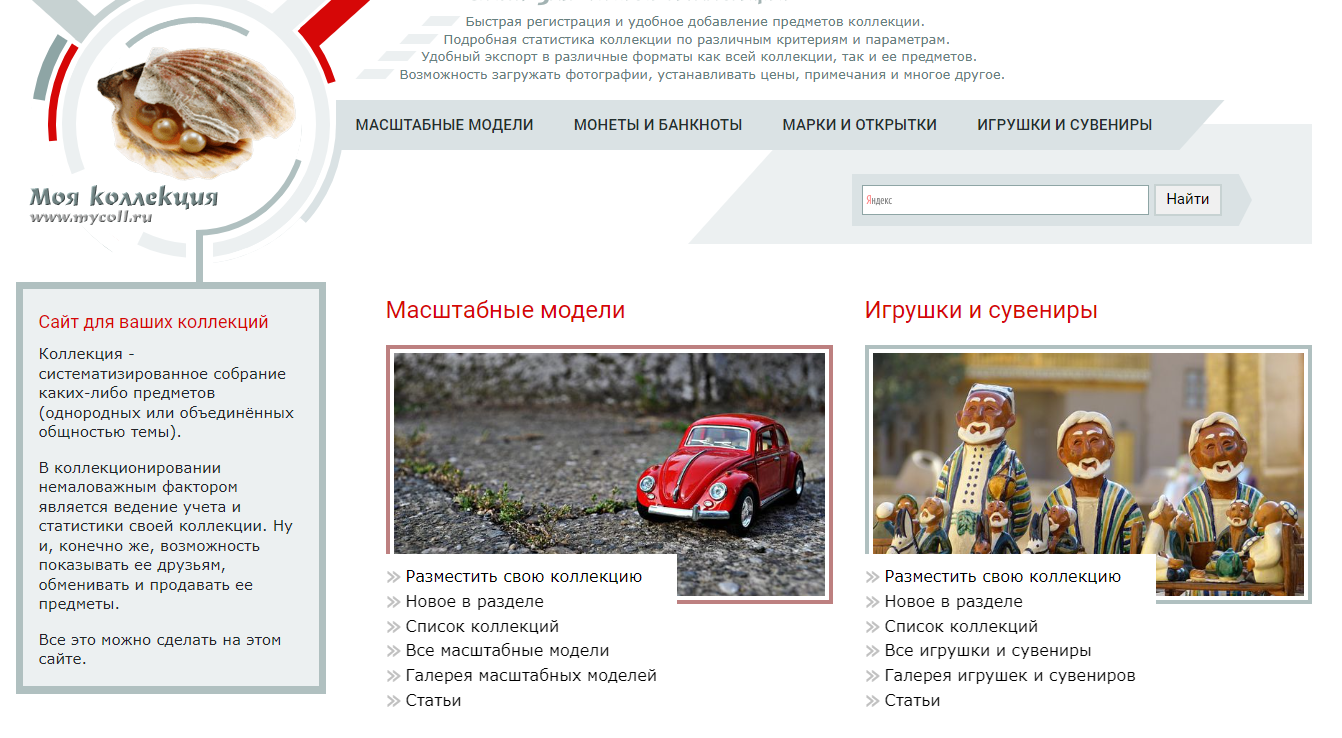


Рисунок 3 - Главная страница сайта mycoll.ru

Еще одним преимуществом является возможность создать коллекцию и добавить в неё свои предметы. При этом поиск необходимых предметов облегчается за счёт наличия фильтров (Рисунок 4).

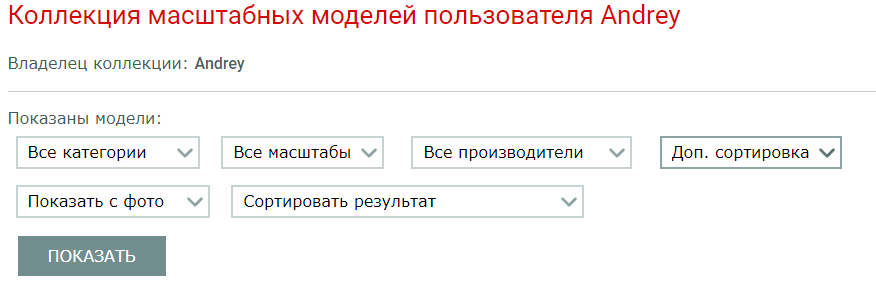


Рисунок 4 - Фильтры на странице просмотра коллекций.

Общение происходит прямо на сайте, но нет указать цену предмета.

К другим недостаткам можно отнести, то что сайт не имеет функциональных возможностей, кроме просмотра контента.

Avito.ru:

Avito - это один из самых популярных российских онлайн-сервисов, позволяющий продавать и покупать различные товары и услуги.

На сайте Avito можно найти объявления о продаже многих категорий товаров. Каждое объявление содержит описание товара, фотографии, контактную информацию продавца, а также возможность связаться с продавцом через встроенную систему сообщений.

В целом Avito это удобный и надежный сервис для продажи и покупки товаров.

Хотя Avito позволяет продавать и покупать широкий спектр товаров и услуг, в том числе коллекционные предметы, это не является основной специализацией сайта. На Avito можно найти объявления о продаже коллекционных предметов, таких как монеты, открытки, почтовые марки и другие, но поиск и продажа коллекционных предметов на этом сайте не всегда является удобным и эффективным.

Avito не обладает такой же гибкостью, как специализированные сайты для коллекционеров. Например, на Avito нельзя создавать свои собственные коллекции, где можно хранить информацию о коллекционных предметах, которыми вы владеете или которые ищете.

Таблица 1 – Сравнение функционала

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Mycollection | Mycoll | Avito | Разрабатываемый сайт |
| Создание коллекции | Есть | Есть | Нет | Есть |
| Добавление предмета | Есть | Есть | Есть | Есть |
| Разделение предметов по категориям | Есть | Есть | Есть | Есть |
| Поиск предметов по сайту | Нет | Есть | Есть | Есть |
| Разграничение доступа к предметам и коллекциям | Нет | Нет | Нет | Есть |
| Отправление обмена | Нет | Нет | Нет | Есть |
| Возможность указать номинальную цену предмета | Есть | Нет | Есть | Есть |

Чтобы лучше понимать критерии сравнения в таблице, приведем подробное описание каждого пункта отдельно.

1. Создание коллекции необходимо для более удобного взаимодействия с сайтов и более удобного добавления предметов.
2. Разделение предметов по категориям и поиск предметов по сайту позволяет более быстро и намного удобнее искать необходимые предметы.
3. Разграничение доступа необходимо для того чтобы скрывать из публичного доступа, либо делать доступ по ссылке для необходимых предметов и коллекций.
4. Отправление обмена подразумевает под собой функцию отправки пользователю предложения обмена, не через чат в письменной форме, а специальным инструментом.

Таблица сравнивает функционал трех различных сайтов для коллекционеров: Mycollection, Mycoll и Avito, а также включает в себя пункт о разрабатываемом сайте.

Из таблицы видно, что Mycollection и Mycoll предоставляют больший функционал для коллекционеров, чем Avito. Например, на Mycollection и Mycoll можно создавать свои коллекции, разделять предметы по категориям и указывать номинальную цену на предметы. В то же время на Avito этот функционал ограничен, и сайт не нацелен на коллекционные предметы.

Однако, ни один из существующих сайтов не позволяет разграничивать доступ к коллекциям и предметам. Это может быть важным функционалом для тех, кто хочет дать доступ только определенным пользователям. Разграничение доступа означает возможность установки определенных прав на доступ к коллекции или конкретному предмету в коллекции. Это добавляет управлять тем, кто видит коллекции или предметы, и кто может иметь к ним доступ.

Различные уровни доступа могут быть установлены для различных пользователей, например, чтобы предмет был видим только владельцу коллекции или, был доступен только по ссылке. Также возможность установить доступ к предмету или коллекции только для определенных пользователей, которые будут авторизованы на сайте.

Таким образом, разграничение доступа может быть полезным для тех, кто хочет сохранить приватность своей коллекции или предмета, и предоставить доступ только определенным пользователям. Это также может быть полезно для тех, кто хочет создать коллекцию для обмена и хочет, чтобы только определенные пользователи имели доступ к этим предметам.

Также, отправка обмена может быть важной функцией для коллекционеров, но ни один из сравниваемых сайтов не предоставляет такую возможность.

В этом отношении разрабатываемый сайт будет лучшим выбором. Однако, для того чтобы разрабатываемый сайт был лучше, чем существующие, необходимо учесть все потребности и пожелания пользователей, а также обеспечить качественную и удобную работу с сайтом.

* 1. Бизнес-процессы

Схемы и диаграммы играют важную роль в проектировании и разработке веб-сервиса. Они позволяют упростить сложную информацию и обеспечить лучшее понимание функциональных и архитектурных особенностей системы [9]. Причины, почему необходимо строить схемы и диаграммы при проектировании веб-сервиса:

* с помощью схем и диаграмм можно представить архитектуру веб-сервиса в виде графического изображения, что поможет лучше понимать ее структуру и функциональность;
* диаграммы используются для общения между разработчиками, дизайнерами, заказчиками и другими заинтересованными сторонами, это позволит уточнять требования и согласовывать решения на ранних стадиях проекта;
* диаграммы позволяют лучше понимать пользовательский опыт и потребности пользователей, это помогает определить, какие функции и интерфейсы следует включить в веб-сервис;
* использование диаграмм и схем в процессе разработки веб-сервиса может помочь выявить ошибки и улучшить процессы работы системы, это может уменьшить количество ошибок и сбоев в работе сервиса и повысить его эффективность;
* схемы и диаграммы являются важным компонентом документации веб-сервиса. Они помогают описать систему и ее функциональность для технических и не технических пользователей. Это может быть полезно при поддержке и развитии веб-сервиса в будущем.

BPMN (Business Process Model and Notation) - это нотация, используемая для моделирования бизнес-процессов [29]. BPMN диаграммы используются для визуализации бизнес-процессов и позволяют описать, как различные задачи, ресурсы и участники связаны между собой [45].

BPMN диаграммы помогают визуализировать бизнес-процессы и описать их в виде графического изображения. Это делает процессы более понятными и доступными [20].

Кроме того, диаграммы позволяют идентифицировать узкие места в процессах и выявить проблемы. Это помогает улучшить процессы и повысить эффективность бизнеса. Также они помогают описать процессы и их структуру для технических и не технических пользователей.

В целом, BPMN диаграммы являются полезным инструментом для моделирования бизнес-процессов. Они позволяют улучшить понимание процессов, улучшить процессы и повысить эффективность.

Процесс добавления новой коллекции представлен на рисунке 5.

На данной диаграмме представлено два пула с дорожками «Пользователь» и «Система». Первым шагом пользователю необходимо авторизоваться на сайте, то есть ввести email и пароль в соответствующие поля. После чего система открывает страницу главную страницу. Следующим шагом пользователь заходит в раздел «Мои коллекции». Система открывает ему страницу со всеми его коллекциями. Далее пользователь нажимает на кнопку создать коллекцию, и система открывает ему форму с шаблоном для создания новой коллекции. После этого пользователь вручную заполняет данные и нажимает на кнопку «Создать». Система сохраняет коллекцию в базе данных «Коллекции», далее размещает её на сайте.

Процесс создания предложения обмена представлен на рисунке 6.

Данная диаграмма содержит два пула с разделением на «Пользователя» и «Систему». Чтобы начать использовать сайт, пользователь должен сначала пройти авторизацию, введя свой email и пароль в соответствующие поля. После успешной авторизации система открывает главную страницу сайта. После успешной авторизации пользователь переходит в раздел каталога, где система предоставляет страницу с предметами, добавленными другими пользователями на сайт. Затем пользователь заполняет поле поиска и применяет необходимые фильтры для сортировки предметов по своим потребностям. Далее он нажимает на кнопку "обмен" на карточке нужного товара, и система открывает страницу с информацией об обмене. Пользователь заполняет форму обмена и нажимает кнопку "обмен". Система сохраняет данные о сделке в базе данных "Обмены" и публикует ее на сайте.

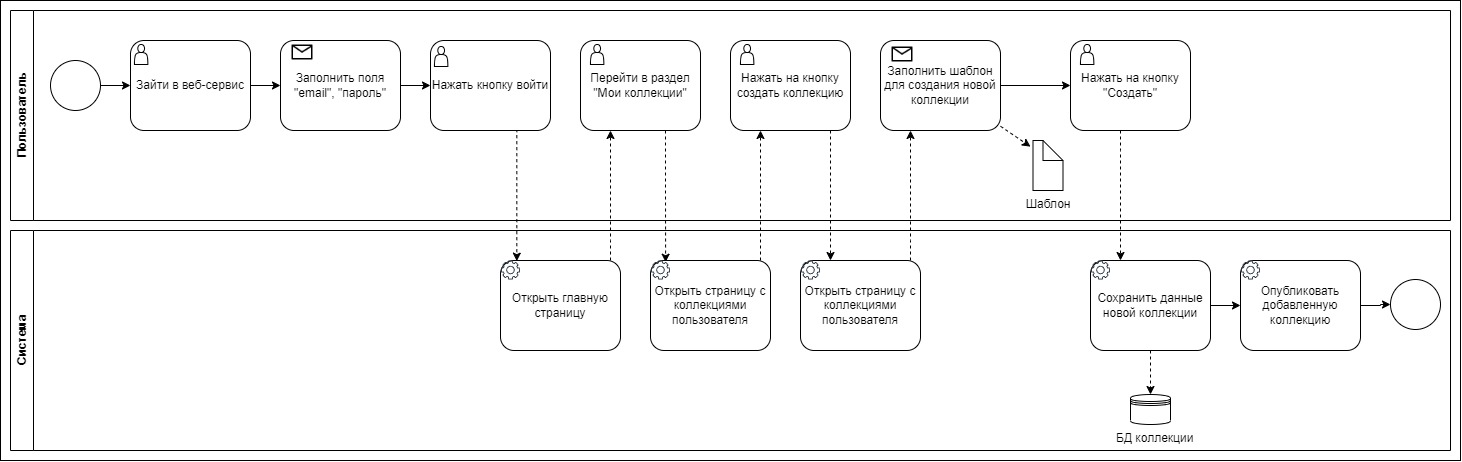


Рисунок 5 - BPMN диаграмма для процесса добавления коллекции

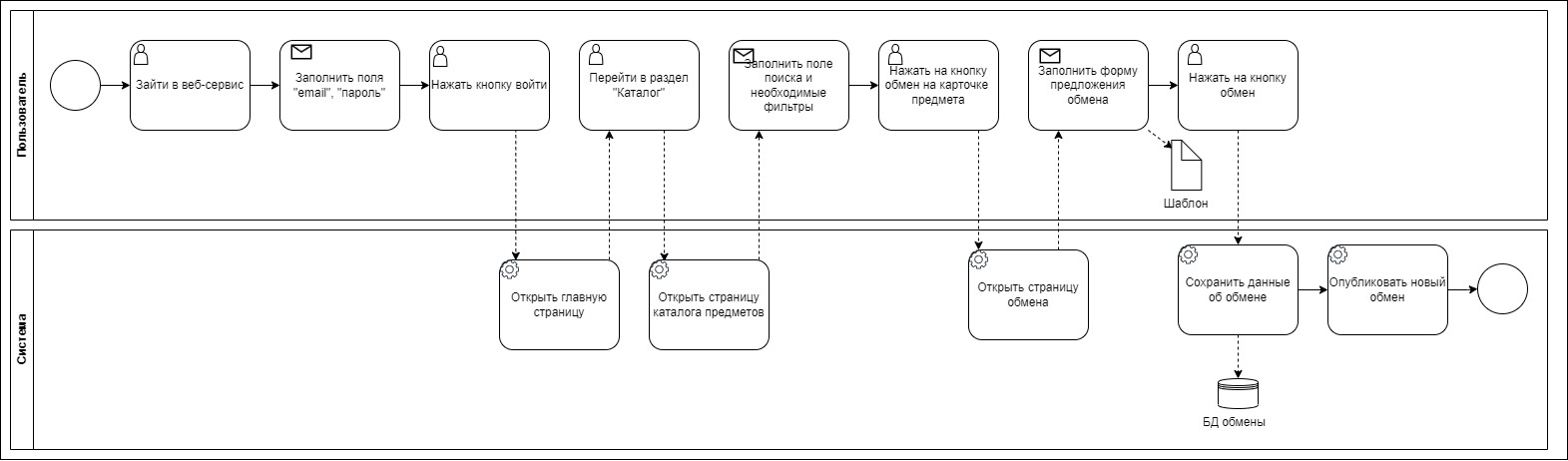


Рисунок 6 - BPMN диаграмма для процесса создания обмена

Диаграмма IDEF0 to-be позволяет изобразить процессы и функции системы на блок-схеме с использованием стандартных символов и конвенций, что позволяет легче понимать структуру и логику работы системы.

Диаграмма IDEF0 (Integration Definition for Function Modeling) to-be - это графический инструмент, используемый для моделирования бизнес-процессов и системных функций с целью улучшения их эффективности [10].

При разработке веб-сервисов диаграмма IDEF0 to-be может использоваться для определения оптимальной последовательности действий, которые должна выполнять система для достижения заданных целей. Также она может быть полезна для выявления возможных проблем или узких мест в процессе работы системы, что позволит улучшить ее эффективность и надежность [33].

На рисунке 7 представлена контекстная диаграмма TO-BE в нотации IDEF. На вход диаграммы поступают информация о пользователе (I1), информация о доступных предметах в базе данных (I2). Элементами управления являются политика сайта (C1) и инструкция пользователя (C2). Механизмами являются информационная система (M1) и пользователь (M2). На выходе диаграммы получается договоренность о сделке (O1) и подтвержденный обмен (O2).

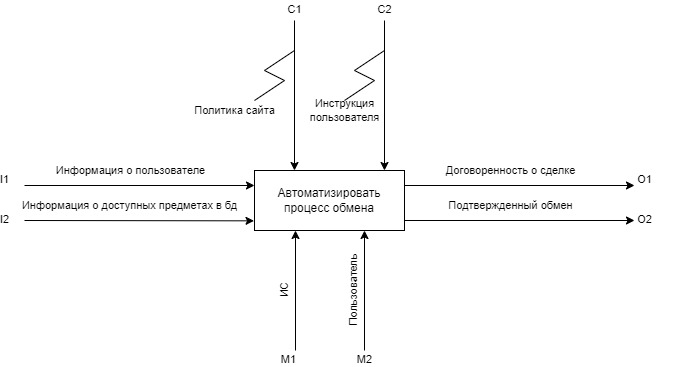


Рисунок 7 - Диаграмма IDEF0 TO-BE

На рисунке 8 представлена декомпозиция диаграммы IDEF0 TO-BE. Всего в диаграмме 4 функциональных блока. На вход поступают информация о пользователе и информация о доступных предметах в базе данных. На выходе получается подтвержденный обмен и договоренность о сделке.

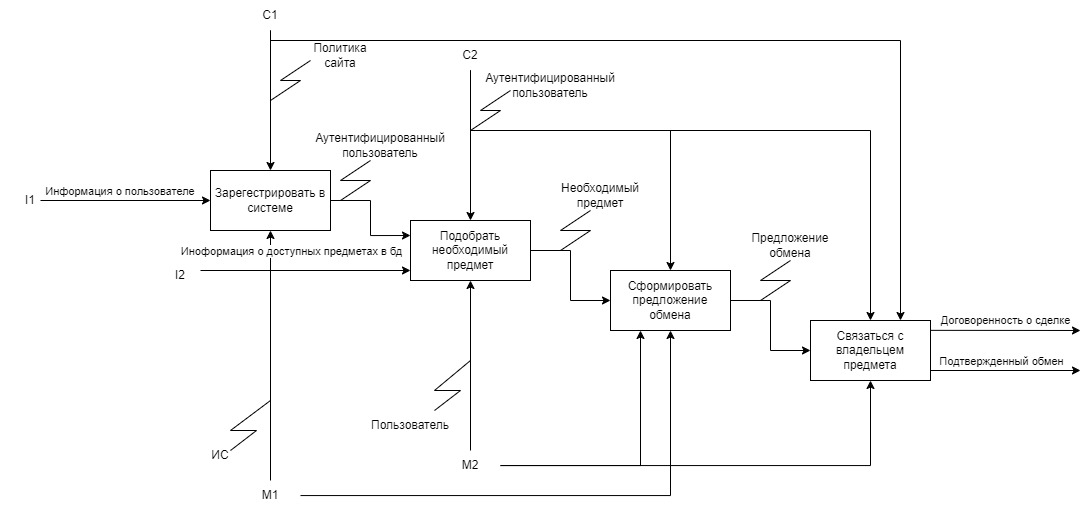


Рисунок 8 - Декомпозиция диаграммы IDEF0 TO-BE

* 1. Техническое задание

Техническое задание (ТЗ) является важным документом при разработке сайта, так как оно определяет все основные требования и условия, которые необходимо выполнить при его создании. Техническое задание содержит в себе технические детали проекта, цели и требования к функциональности сайта, а также определяет ожидаемые результаты и сроки выполнения проекта [18].

Кроме того, техническое задание является основой для взаимодействия между заказчиком и разработчиком сайта. Оно помогает избежать возможных недоразумений и конфликтов, а также сокращает количество возможных правок в будущем. В ТЗ указываются все требования заказчика к сайту, поэтому этот документ является обязательным элементом при заключении контракта на разработку сайта.

Кроме того, техническое задание определяет требования к дизайну, содержанию, функционалу и взаимодействию с пользователем. Это позволяет создать сайт, который будет соответствовать потребностям заказчика и обеспечит высокий уровень удобства использования для пользователей.

В целом, техническое задание является ключевым элементом при разработке сайта и его правильное выполнение гарантирует успешную реализацию проекта.

Техническое задание (ТЗ) играет очень важную роль при ведении разработки веб-сайта. Его преимущества включают:

* Определение целей и требований проекта: ТЗ содержит детальное описание всех требований и условий проекта, что позволяет убедиться в том, что все цели и требования клиента будут учтены и выполнены в процессе разработки.
* Уменьшение возможных конфликтов: ТЗ помогает избежать конфликтов между клиентом и разработчиком, так как все условия и требования проекта определены заранее.
* Ускорение процесса разработки: Техническое задание ускоряет процесс разработки, так как в нем четко определены все требования и условия проекта.
* Легкий контроль за процессом работы: Техническое задание является руководством для разработчика, что позволяет контролировать процесс работы и убедиться, что все условия и требования проекта выполнены.

Для разрабатываемого сайта было составлено техническое задание для, которое представлено в Приложение А. В этом документе были четко определены требования.

* 1. Выводы

В данном разделе была проведена работа по описанию предметной области, поставлены задачи для выполнения в рамках дипломной работы.

Для написания серверной части разрабатываемого сайта будет использоваться Django. При написании фронтенд части будет применяться Vue.

Затем был развернут анализ трех частично совпадающих по функционалу сервисов с приведенными выводами.

Разработан и проведен обзор основных бизнес-процессов.

В результате анализа предметной области, было составлено техническое

задание, которое приводится в приложении А.

1. ТЕХНИЧЕСКАЯ РЕАЛИЗАЦИЯ
2. 1. Функции сайта.

На начальных этапах жизненного цикла был выполнен проект системы, предполагающий построение модели для будущего использования на этапе разработки. Одной из моделей, созданных в ходе этого процесса, была диаграмма прецедентов.

Диаграмма прецедентов - это один из ключевых элементов в процессе проектирования программного обеспечения, которая помогает определить функциональные требования к системе и ее пользовательскому интерфейсу.

Диаграмма прецедентов представляет собой графическое изображение взаимодействия между пользователями и системой, показывая какие действия пользователи могут выполнять в рамках системы и как система будет на них реагировать. Она содержит актеров, прецеденты и связи между ними [11].

Преимущества использования диаграммы прецедентов включают [12]:

* Понимание функциональных требований к системе: Диаграмма прецедентов помогает определить, какие функции системы должны выполняться и как пользователи будут взаимодействовать с системой.
* Улучшение UX: Диаграмма прецедентов позволяет определить, какие действия пользователи могут выполнять в рамках системы и как система будет на них реагировать, что помогает улучшить пользовательский интерфейс [13].
* Уменьшение рисков и ошибок: Диаграмма прецедентов помогает идентифицировать возможные проблемы и риски, которые могут возникнуть в ходе разработки и использования системы.
* Упрощение коммуникации между разработчиками и заказчиками: Диаграмма прецедентов позволяет заказчикам и разработчикам понимать, как система будет использоваться и какие функции будут выполняться, что упрощает коммуникацию между ними.

Использование диаграммы прецедентов является важным элементом процесса разработки программного обеспечения, что позволяет определить функциональные требования к системе, улучшить UX, уменьшить риски и ошибки, а также упростить коммуникацию между заказчиками и разработчиками.

На рисунке 9 изображена диаграмма вариантов использования, построенная для сайта просмотра и обмена коллекционных предметов collectxchange.

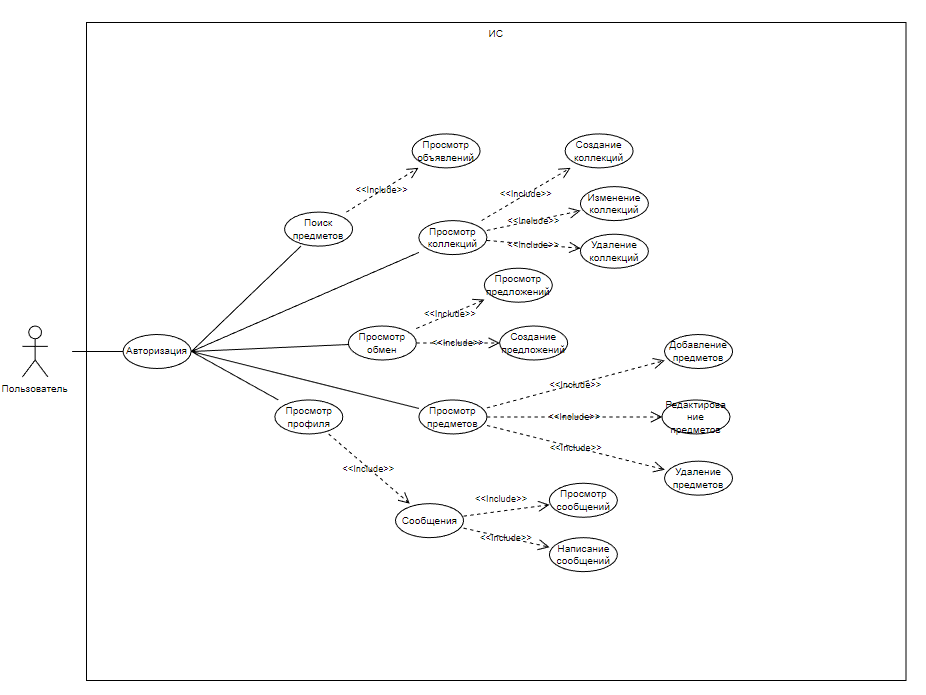


Рисунок 9 - Диаграмма прецедентов сайта collectxchange.

На диаграмме мы можем наблюдать роль пользователя, представленную актером «Пользователь», который выступает в роли типичного пользователя разрабатываемого веб-сайта. После авторизации в системе этот пользователь получает доступ к определенным функциям: просмотр профиля, просмотр предметов, просмотр обменов, просмотр коллекций, поиск предметов.

Просмотр профиля включает в себя возможность просмотра сообщений. Пользователь может обмениваться сообщениями с другими пользователями сайта для обсуждения коллекционных предметов, обменов или для решения других вопросов.

Поиск товаров - пользователь вводит ключевые слова в поисковую строку на сайте, и система отображает соответствующие результаты.

Прецеденты просмотра предметов и коллекций, каждый включает в себя следующие возможности:

* Добавление
* Редактирование
* Удаление

Прецедент обмен позволяет просматривать предложения обмена полученные от других пользователей сайта, а также пользователь может сам отправить предложения обмена.

* 1. Создание графического пользовательского интерфейса.

Графический пользовательский интерфейс (GUI) сайта необходим для облегчения взаимодействия между пользователем и сайтом. GUI обеспечивает более удобный и интуитивно понятный интерфейс, который позволяет пользователям легко осуществлять навигацию по сайту, выполнять различные действия, ознакомиться с предоставляемым контентом и взаимодействовать с другими пользователями [17, 34].

Графический пользовательский интерфейс сайта также способствует повышению удобства использования сайта и снижению порога входа для новых пользователей, что увеличивает вероятность их повторного посещения и улучшения UX [14]. Кроме того, GUI позволяет упростить процесс взаимодействия с различными функциями сайта, сократить время на поиск необходимой информации и повысить производительность пользователей в использовании сайта [39].

Графический пользовательский интерфейс (GUI) состоит из различных элементов, таких как:

* Элементы управления, такие как кнопки, переключатели, выпадающие списки, чекбоксы, радиокнопки и другие элементы, которые позволяют пользователю взаимодействовать с сайтом и выполнить нужные действия.
* Графические элементы, такие как иллюстрации, фотографии, диаграммы, графики и другие элементы, которые используются для визуализации информации и облегчения понимания контента сайта.
* Меню и навигационные элементы, которые помогают пользователям легко перемещаться по сайту и найти нужную информацию.
* Текстовые элементы, такие как заголовки, тексты, описания, подписи и другие элементы, которые используются для передачи информации и описания контента сайта.
* Различные анимации и эффекты, которые используются для улучшения визуального восприятия сайта и создания более динамичного UX.

Все эти элементы в совокупности обеспечивают более удобный, привлекательный и интуитивно понятный интерфейс для пользователей, что повышает эффективность использования сайта и улучшает UX [30].

При разработке сайта важно учитывать не только функциональные требования, но и дизайн, в том числе создание эффективного и интуитивно понятного пользовательского интерфейса.

Для того, чтобы создать оптимальный GUI, необходимо учитывать множество факторов, таких как потребности целевой аудитории, особенности предлагаемого продукта или услуги, актуальные тренды и стандарты дизайна, а также множество других аспектов. В процессе разработки GUI, часто используются методы тестирования, проводятся опросы пользователей и анализируются результаты, чтобы определить наиболее эффективные решения для улучшения UX [47].

Рассмотрим основные компоненты сайта и подробно изучим каждый элемент сайта, его функционал и визуальное оформление [15,16,17].

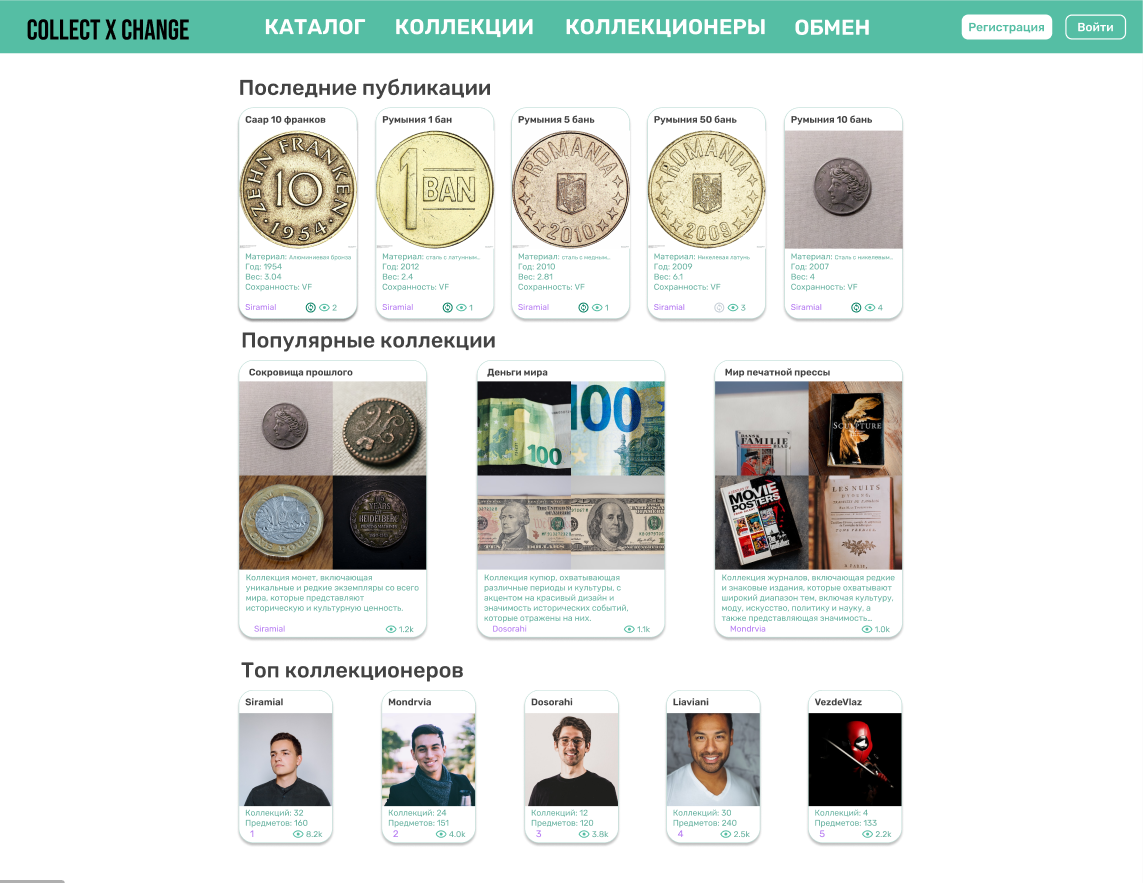


Рисунок 10 - Главная страница сайта collectxchange.

Главная страница (0100) на сайте является первым местом, которое видят пользователи при посещении сайта. Она предоставляет пользователям доступ к различным функциям и сервисам сайта, а также позволяет им быстро найти интересующие их предметы.

Интерфейс имеет верхнее меню навигации с размещенными в правой части кнопками "Войти" и "Регистрация". Пользователи могут использовать эти кнопки для авторизации в системе или создания нового аккаунта.

Расположение меню навигации с ссылками на основные страницы сайта в верхней части экрана имеет несколько преимуществ:

* Удобство использования: пользователи могут легко найти ссылки на основные страницы сайта в верхней части экрана. Это позволяет им быстро получить доступ к нужным страницам и осуществлять навигацию по сайту с минимальными усилиями.
* Повышение узнаваемости бренда: расположение меню навигации в верхней части экрана является стандартным для большинства сайтов, поэтому пользователи быстро узнают, где оно находится и как им пользоваться. Это может способствовать узнаваемости бренда и повышению уровня доверия к сайту.

На главной странице размещены три основных заголовка: "Последние публикации", "Популярные коллекции" и "Топ коллекционеров". Под каждым заголовком размещены соответствующие контентные блоки.

В блоке "Последние публикации" находятся карточки предметов и их основные характеристики. Карточки включают в себя фотографию предмета, его название, описание, и другие характеристики.

В блоке "Популярные коллекции" пользователи могут увидеть коллекции, отсортированные по количеству просмотров. Каждая коллекция содержит изображения предметов, которые входят в нее, название коллекции, краткое описание и информацию об авторе.

В блоке "Топ коллекционеров" пользователи могут увидеть коллекционеров с наибольшим количеством просмотров. Каждый профиль коллекционера содержит информацию о количестве коллекций, а также количество просмотров.

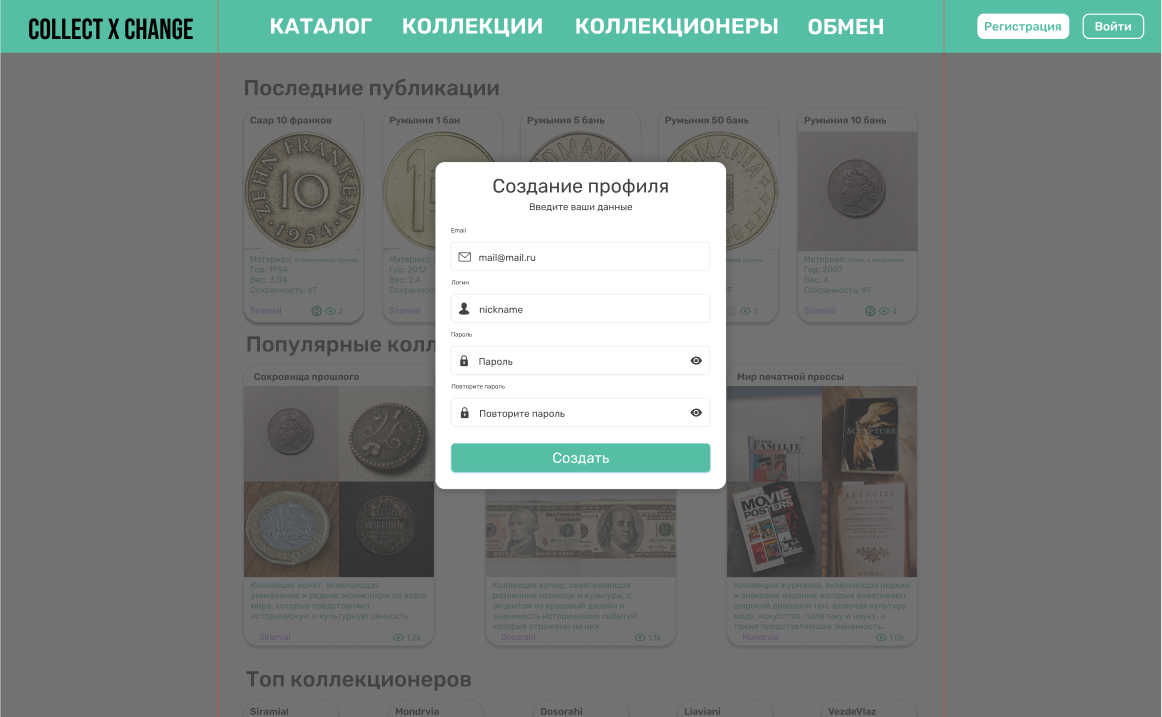


Рисунок 11 - Форма создания нового профиля.

При нажатии на кнопку регистрация на экране пользователя всплывающее окно (pop-up) (Рис. 11). В этом окне пользователю необходимо ввести его данные, в случае ошибок, таких как дублирующийся nickname либо не совпадение паролей, пользователь видит соответствующую ошибку на экране.

Размещение формы регистрации во всплывающем окне - это распространенная практика на веб-сайтах, которая имеет несколько преимуществ:

* Улучшение UX: когда форма регистрации размещена во всплывающем окне, пользователи могут быстро и удобно зарегистрироваться на сайте без необходимости покидать текущую страницу или искать форму на другой странице. Это уменьшает количество шагов, необходимых для выполнения регистрации, что может повысить удовлетворенность пользователей и сократить вероятность отказа от регистрации.
* Повышение конверсии: форма регистрации во всплывающем окне может быть более заметной и привлекательной для пользователей, чем форма на обычной странице. Это может увеличить количество зарегистрированных пользователей и повысить конверсию.

После прохождения регистрации и авторизации пользователь может добавлять свои коллекционные предметы на сайт для этого создана удобная и простая форма (рис. 12).

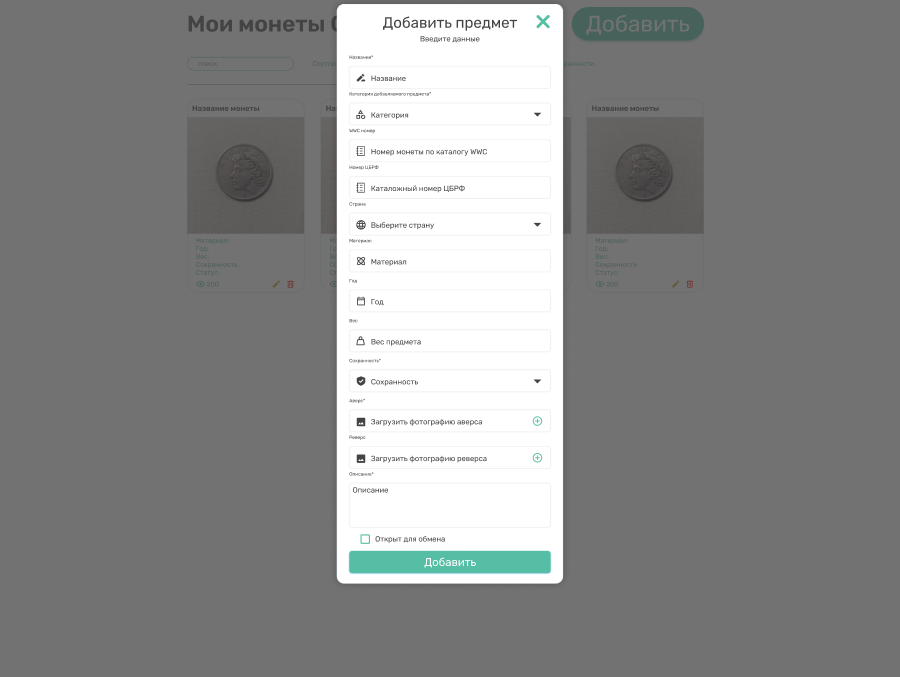


Рисунок 12 - Окно добавления предмета на сайт

Каждая категория имеет свои свойства соответственно. На сайте на данном этапе разработки представлено 8 основных, самых популярных категорий среди коллекционеров:

* Монеты
* Марки
* Купюры
* Журналы
* Видеоигры
* Значки
* Набор монет

В случае отсутствия необходимой категории пользователь может выбрать категорию «Другое». Характеристики в свою очередь делятся на 2 вида обязательные для заполнения и необязательные. Для удобства пользователя обязательные поля помечены символом “\*”.

Далее заполняются характеристики, предложенные пользователю согласно категории.

Также у каждого предмета имеется свойство «Открыт для обмена», при необходимости пользователь может отключить возможность обменивать необходимый предмет с другими пользователями.

В данной форме неоднократно используется поле input с выпадающим списком - это элемент интерфейса, который позволяет пользователю выбирать один вариант из предопределенного списка. Например, поля «Выберите страну» и «Категория» (рис. 13). Преимущества такого элемента интерфейса:

* + - Уменьшение количества ошибок ввода: с помощью поля input с выпадающим списком пользователь может выбрать один из предопределенных вариантов, что снижает вероятность ошибки при вводе данных. Это особенно полезно, когда список вариантов достаточно большой или вводимые данные имеют строгий формат.
    - Упрощение процесса выбора: с помощью поля input с выпадающим списком пользователи могут быстро и удобно выбирать нужный вариант, не нужно вводить данные вручную или искать нужную информацию на странице. Это может уменьшить время, затрачиваемое на выполнение определенных задач и улучшить UX.
    - Снижение затрат на обработку данных: с помощью поля input с выпадающим списком данные, которые пользователь вводит в форму, уже находятся в правильном формате и не нуждаются в дополнительной обработке или валидации. Это может уменьшить затраты на обработку данных и ускорить выполнение операций.

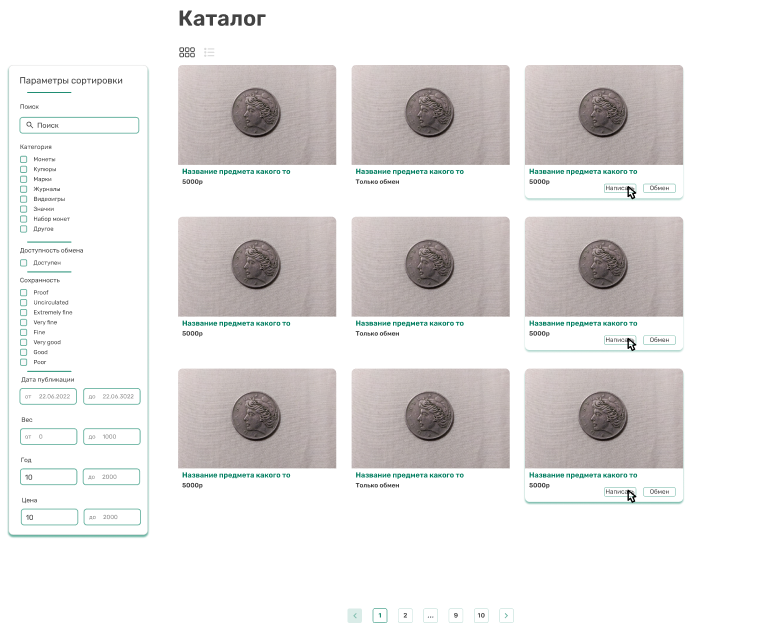


Рисунок 13 - Режим отображения каталога ячейками

Страница каталога на сайте - это страница, на которой можно просмотреть все доступные категории товаров на сайте. Эта страница позволяет пользователям быстро найти нужный предмет. (рис. 14)

Пользователь может ввести запрос в строку поиска и получает список соответствующих объявлений.

Для удобства пользователей реализовано два типа отображения ячейки (рис. 13) и список (рис. 14)

Каждое объявление на странице содержит заголовок, фотографии описание и цену предмета.

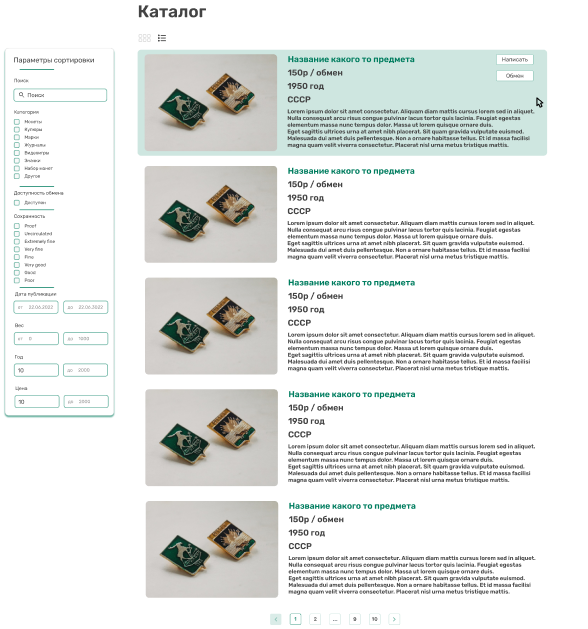


Рисунок 14 - Режим отображения каталога списком

Пользователи могут сортировать объявления по различным критериям, таким как цена, дата публикации, год и вес. Они также могут фильтровать результаты поиска по категориям, чтобы показать только те объявления, которые соответствуют необходимой категории.

Помимо этого, пользователь может отфильтровать только предметы доступные для обмена.

Параметры сортировки - это элемент интерфейса, который позволяет сократить количество отображаемых товаров и сузить поиск до конкретных товаров, которые соответствуют критериям пользователя.

Преимущества для пользователей:

* Экономия времени: с помощью фильтров пользователь может быстро отсортировать и выбрать нужный товар, не тратя время на поиск по всему каталогу.
* Улучшенный UX: благодаря упрощенному поиску, пользователь может легко и быстро найти то, что ищет, и это положительно сказывается на его впечатлении от магазина.
* Улучшение качества поиска: с помощью фильтров, пользователи могут уточнять свои запросы и уменьшать количество результатов поиска, что может сократить количество ненужной информации.
* Лучший контроль над поиском: фильтры дают пользователям полный контроль над поиском и позволяют им выбирать критерии, которые им важны.
* Уменьшение вероятности ошибок: благодаря уточненному поиску, пользователи могут избежать ошибок при выборе товара, которые могут возникнуть при неправильном или неточном запросе.
* Упрощение выбора: с помощью фильтров, пользователи могут выбирать товары по нескольким критериям одновременно, что упрощает выбор и делает его более точным.
* Повышение удобства использования: фильтры обеспечивают удобный и простой интерфейс для выбора нужных товаров, что может сделать покупку более удобной и приятной для пользователя.

Пользователи могут кликнуть на любой объявление, чтобы просмотреть подробную информацию о предмете. На странице объявления пользователи могут увидеть дополнительные фотографии, описание, способы связи с продавцом.

Если пользователь заинтересовался товаром, он может связаться с продавцом, используя контактные данные на странице объявления. Пользователи также могут добавлять объявления в избранное, чтобы вернуться к ним позже, или поделиться ссылкой на объявление с другими пользователями.

После того, как пользователь нашел товар, который он хотел бы обменять, он может отправить запрос на обмен владельцу товара, разместившему объявление. Для этого он может нажать на кнопку, размещённую в объявлении, либо зайти в профиль человека, разместившего объявление. В запросе он может указать, что он предлагает взамен на свой товар и задать дополнительные вопросы, если нужно.

В нижней части страницы расположена пагинация. Пагинация - это элемент интерфейса веб-сайта, который используется для разбиения большого количества контента на несколько страниц. Страница каталога содержит определенное количество предметов. Когда пользователь достигает конца страницы, он может перейти на следующую страницу, нажав на соответствующую ссылку, или на предыдущую страницу, если ему нужно вернуться к ранее просмотренному контенту.

Преимущества пагинации для пользователей сайта:

* Улучшенная скорость загрузки страницы: когда на странице отображается большое количество контента, это может замедлить скорость ее загрузки. Разбиение на страницы позволяет загружать только необходимый контент, ускоряя время загрузки страницы.
* Улучшенная навигация по сайту: пользователи могут легко перемещаться по страницам и находить нужный контент, используя ссылки на страницы пагинации.
* Удобство использования: пагинация является удобным и интуитивно понятным элементом интерфейса, который позволяет пользователям быстро переходить к нужной странице.
* Уменьшение загрузки сервера: разбиение на страницы помогает снизить нагрузку на сервер и улучшить производительность сайта.
* Улучшенный SEO: использование пагинации может улучшить SEO-оптимизацию сайта, поскольку поисковые системы лучше индексируют страницы с пагинацией.
* Улучшенный UX: разбиение на страницы улучшает UX, поскольку пользователи могут быстро находить нужную информацию и легко перемещаться по сайту.

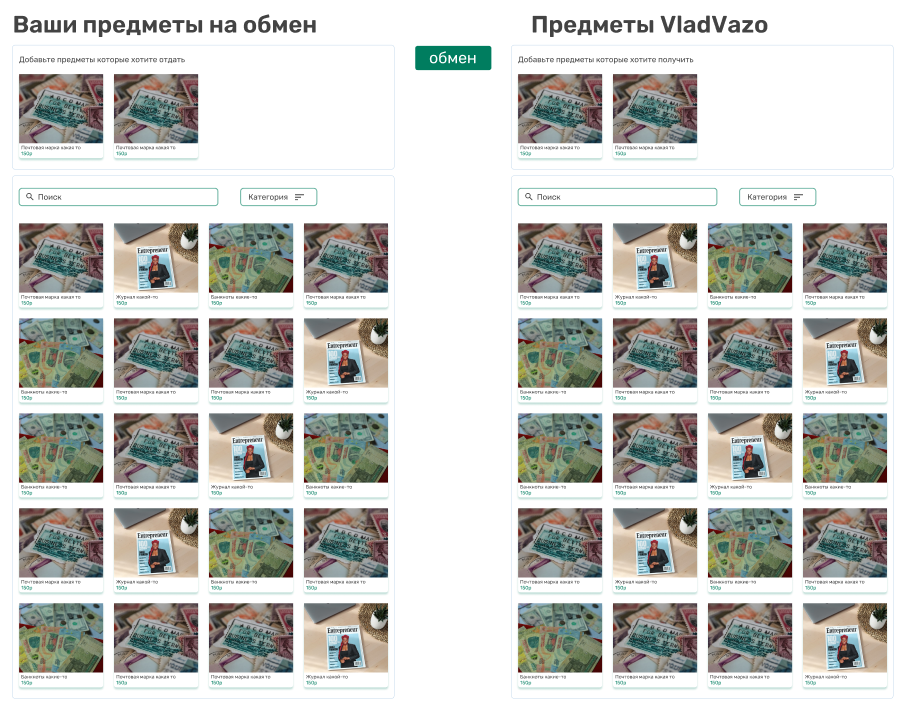


Рисунок 15 - Страница предложения обмена

При нажатии на кнопку обмена в каталоге пользователь попадает на страницу обмена (Рис. 15).

Страница обмена на сайте CS.MONEY предоставляет пользователям возможность быстро и удобно обменивать предметы. На этой странице пользователь может найти все доступные для обмена предметы, которые были добавлены в учетную запись.

На этой странице вы увидите список доступных предметов, которые можно обменять, а также их стоимость. Для обмена пользователю необходимо выбрать предметы, которые он хочет обменять, а затем выбрать предметы, которые хочет получить взамен.

Кроме того, страница обмена на предоставляет возможность сортировки по категориям. Это позволяет быстро и удобно находить нужные предметы и выбирать оптимальный вариант для обмена.

Также на странице обмена доступна функция поиска, которая позволяет быстро находить нужные предметы по ключевым словам. Все это делает процесс обмена на сайте еще более удобным и простым.

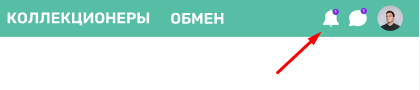


Рисунок 16 - Уведомление о получении предложения обмена.

Пользователям, получившим предложение обменяться приходит уведомление, которая отображается в верхней части страницы (Рис. 16).

При нажатии на уведомления открывается страница со всеми доступными предложениями (Рис. 17)

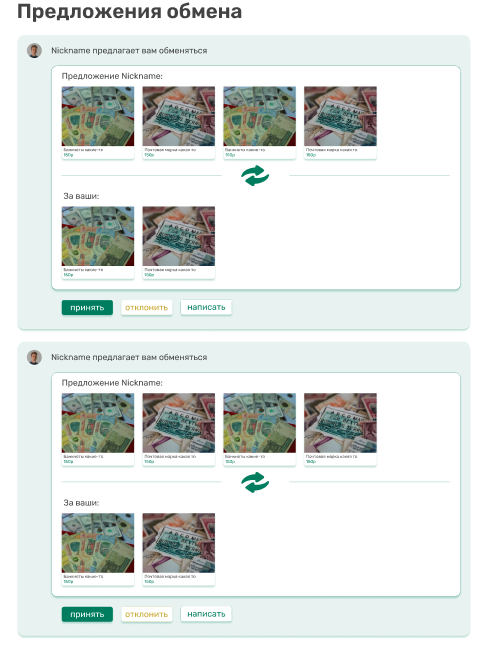


Рисунок 17 - Предложения обмена.

Общий вид интерфейса имеет следующие основные элементы:

* Шапка окна: в верхней части окна расположены заголовок "Предложение обмена" кнопка "Закрыть" (Close) и информация о том, кто отправил вам предложение обмена (имя пользователя).
* Область полученных предметов: в этой области отображаются предметы, которые вам предлагают в обмен. Вы можете увидеть название, изображение и стоимость каждого предмета.
* Область ваших предметов: в этой области отображаются предметы, на которые вам предлагают обмен.
* Кнопки управления: внизу окна расположены кнопки управления, которые позволяют вам принять или отклонить предложение обмена, а также написать пользователю.
  1. База данных сайта

ER-диаграмма базы данных для сайта collectxchange приведена на рисунке 18

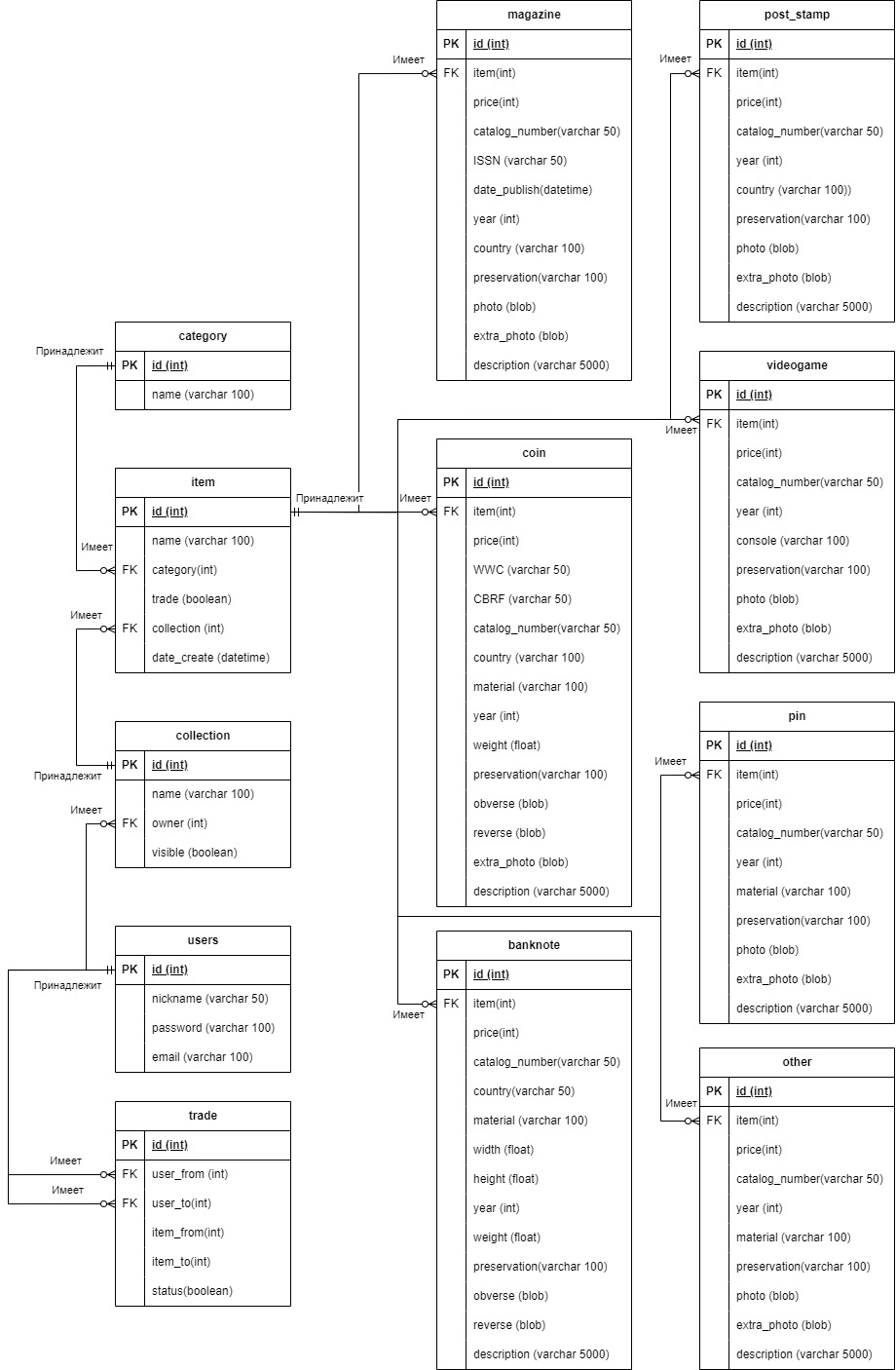


Рисунок 18 - ER-диаграмма базы данных сайта collectxchange

ER-диаграмма (Entity-Relationship diagram) - это графическое представление структуры базы данных, которое позволяет описать связи между объектами в базе данных. Она используется для моделирования и проектирования баз данных перед их созданием, а также для документирования уже существующих баз данных [32].

ER-диаграмма представляет собой набор связанных между собой сущностей, атрибутов и связей между ними. Сущности - это объекты в базе данных, такие как пользователи, заказы, продукты, и т.д. Атрибуты - это свойства сущностей, например, имя пользователя, номер заказа, название продукта, и т.д. Связи между сущностями показывают, как они связаны друг с другом и как взаимодействуют в базе данных [35].

Преимущества ER-диаграммы:

* Ясность и понятность: ER-диаграмма обеспечивает ясное и понятное представление структуры базы данных, что облегчает ее проектирование и документирование.
* Удобство в использовании: ER-диаграмма представляет собой графическое представление базы данных, что делает ее более удобной в использовании и понимании.
* Визуальное моделирование: ER-диаграмма позволяет визуально моделировать базу данных, что позволяет разработчикам быстро и эффективно изменять структуру базы данных.
* Обнаружение ошибок: ER-диаграмма позволяет обнаруживать ошибки в структуре базы данных, такие как несогласованность между связанными сущностями или атрибутами.
* Сокращение времени разработки: ER-диаграмма позволяет сократить время разработки базы данных, так как позволяет разработчикам быстро и эффективно проектировать базу данных до ее создания.
* Облегчение поддержки: ER-диаграмма позволяет легко понимать структуру базы данных, что делает ее поддержку более эффективной и облегченной.

База данных будет содержать 14 таблиц:

* users – таблица с данными пользователей;
* collection - таблица с данными о коллекциях предметов, принадлежащих определенным пользователям;
* item - таблица с данными о предметах добавленных пользователем в коллекцию;
* category - таблица с данными о категории принадлежащей определенному предмету;
* preservation – таблица с данными о категории сохранности принадлежащей определенному предмету;
* country - таблица с данными о стране происхождения, принадлежащей определенному предмету;
* trade – таблица с данными о предложениях обмена, от одного пользователя другому.
* magazine, post\_stamp, coin, banknote, videogame, pin, other – таблицы с перечнем характеристик, принадлежащих определённому предмету.

Таблица users содержит 4 атрибута, а именно

* id – тип данных Integer, содержит уникальный идентификатор записи в таблице, является первичным ключом таблицы;
* nickname – тип данных Varchar с ограничением 50 символов, содержит ник пользователя;
* password - тип данных Varchar с ограничением 100 символов, содержит пароль пользователя;
* email - тип данных Varchar с ограничением 100 символов, содержит электронную почту пользователя

Таблица collection содержит 4 атрибута, а именно

* id – тип данных Integer, содержит уникальный идентификатор записи в таблице, является первичным ключом таблицы;
* name - тип данных Varchar с ограничением 100 символов, содержит название коллекции;
* owner – тип данных Integer, содержит ID пользователя, является внешним ключом к таблице users;
* visible – тип данных Boolean, содержит логическое значение (True или False), показывающее видна ли коллекция другим пользователям .

Таблица item содержит 5 атрибутов, а именно

* id – тип данных Integer, содержит уникальный идентификатор записи в таблице, является первичным ключом таблицы;
* name - тип данных Varchar с ограничением 100 символов, содержит название предмета;
* category - тип данных Integer, содержит ID категории, является внешним ключом к таблице category;
* trade – тип данных Boolean, содержит логическое значение (True или False), показывающее открыт ли предмет для обмена с другими пользователями;
* date\_create – тип данных Date, содержит в себе дату публикации предмета на сайте.

Таблица category содержит 2 атрибута, а именно

* id – тип данных Integer, содержит уникальный идентификатор записи в таблице, является первичным ключом таблицы;
* category\_name - тип данных Varchar с ограничением 100 символов, содержит название категории;

Таблица preservation содержит 2 атрибута, а именно

* id – тип данных Integer, содержит уникальный идентификатор записи в таблице, является первичным ключом таблицы;
* preservation\_name - тип данных Varchar с ограничением 100 символов, содержит название категории сохранности предмета;

Таблицы magazine, post\_stamp, coin, banknote, videogame, pin, other содержат в общей сложности 20 атрибутов, а именно

* id – тип данных Integer, содержит уникальный идентификатор записи в таблице, является первичным ключом таблицы;
* item - тип данных Integer, содержит ID предмета, является внешним ключом к таблице item;
* price – тип данных Integer, содержит информацию о стоимости предмета;
* WWC – тип данных Varchar с ограничением 50 символов, содержит номер предмета по каталогу WWC;
* CBRF - тип данных Varchar с ограничением 50 символов, содержит номер предмета по каталогу CBRF;
* catalog\_number - тип данных Varchar с ограничением 50 символов, содержит номер предмета по какому-либо каталогу;
* country - тип данных Varchar с ограничением 50 символов, содержит название страны;
* material - тип данных Varchar с ограничением 100 символов, содержит в себе названия материала;
* year – тип данных Integer, содержит в себе год.
* weight – тип данных Float, содержит в себе вес предмета;
* preservation – тип данных Integer, содержит ID категории сохранности, является внешним ключом к таблице preservation;
* country – тип данных Integer, содержит ID страны, является внешним ключом к таблице country;
* obverse – тип данных Blob, содержит в себе изображение лицевой стороны предмета;
* reverse – тип данных Blob, содержит в себе изображение обратной стороны предмета;
* extra\_photo – тип данных Blob, содержит в себе дополнительное изображение предмета;
* description - тип данных Varchar с ограничением 5000 символов, содержит в себе описание предмета;
* width - тип данных Float, содержит в себе ширину предмета;
* height - тип данных Float, содержит в себе ширину предмета;
* ISSN - тип данных Varchar с ограничением 50 символов, содержит номер предмета по каталогу ISSN;
* date\_publish – тип данных Date, содержит в себе дату публикации журнала.

Таблицы users и collection связаны отношением 1 к 0..n – одной записи из таблицы users может соответствовать ни одной, одна или множество записей из таблицы collection, а одной записи из таблицы collection может соответствовать одна и только одна запись в таблице user [31].

Таблицы collection и item связаны отношением 1 к 0..n – одной записи из таблицы collection может соответствовать ни одной, одна или множество записей из таблицы item, а одной записи из таблицы item может соответствовать одна и только одна запись в таблице collection.

* 1. Основные алгоритмы работы программы

Веб-сервис создан на основе клиент-серверной архитектуры с использованием REST API [37]. Клиентская часть написана на Vue, фреймворке для разработки пользовательских интерфейсов, который используется для отображения данных и взаимодействия с сервером через API [38]. Серверная часть проекта написана на Django, фреймворке для разработки веб-приложений на языке Python, который используется для обработки запросов, обработки данных и предоставления доступа к данным через API.

REST API на серверной стороне реализован с использованием библиотеки Django REST Framework [36]. Она обеспечивает удобное взаимодействие между клиентской и серверной частями проекта. Клиент отправляет запросы на сервер с использованием методов HTTP, таких как GET, POST, PUT, DELETE и других. Сервер обрабатывает запросы, выполняет соответствующие действия и отправляет ответы клиенту в формате JSON.

Основное преимущество данного подхода заключается в том, что он позволяет разделить приложение на две логические части, обеспечить эффективное взаимодействие между ними и обеспечить доступ к данным через API. Такой подход облегчает сопровождение и масштабирование проекта.

Для регистрации пользователь указывает необходимые данные в соответствующей форме далее на фронтенде определена функция submit, которая отправляет запрос на сервер по адресу http://localhost:8000/api/register с методом POST и телом запроса в формате JSON, содержащим данные пользователя, которые были введены в форму регистрации на фронтенде [48]. Запрос отправляется при помощи функции fetch. Код выполняющий данную операцию представлен в листинге 1.

Листинг 1 – Фрагмент кода отправления запроса на сервер для регистрации.

const data = reactive({

nickname: '',

email: '',

password: ''

});

const submit = async () => {

await fetch('http://localhost:8000/api/register', {

method: 'POST',

headers: {'Content-Type': 'application/json'},

body: JSON.stringify(data)

});

context.emit('close');

}

На сервере, в Django, определен класс RegisterView, который наследуется от класса APIView. Метод post этого класса вызывается при получении POST-запроса на адрес http://localhost:8000/api/register. В методе post создается экземпляр сериализатора UserSerializer с данными, полученными из запроса. Если данные валидны, т.е. соответствуют правилам, указанным в UserSerializer, то сериализатор сохраняет нового пользователя в базе данных и возвращает ответ клиенту, содержащий данные пользователя в формате JSON [49].

Код выполняющий данную операцию представлен в листинге 2.

Листинг 2 – Код класса RegisterView

class RegisterView(APIView):

def post(self, request):

serializer = UserSerializer(data=request.data)

serializer.is\_valid(raise\_exception=True)

serializer.save()

return Response(serializer.data)

Сериализатор UserSerializer определяет, каким образом объект пользователя будет преобразован из формата Python в формат JSON и наоборот. Он также определяет, какие поля будут включены в ответ на запрос, и какие поля могут быть изменены в запросе. Сериализатор также переопределяет метод create, чтобы корректно обработать пароль пользователя, перед сохранением в базу данных [50].

Код выполняющий данную операцию представлен в листинге 2.

Листинг 2 – Сериализатор UserSerializer

class UserSerializer(serializers.ModelSerializer):

class Meta:

model = User

fields = ['id', 'nickname', 'email', 'password']

extra\_kwargs = {

'password': {'write\_only': True}

}

def create(self, validated\_data):

password = validated\_data.pop('password', None)

instance = self.Meta.model(\*\*validated\_data)

if password is not None:

instance.set\_password(password)

instance.save()

return instance

* 1. Выводы

Во второй главе дипломной работы был описан пользовательский интерфейс программного продукта, а также его функционал.

Была приведена и описана разработанная модель данных программного продукта. Модель данных были проиллюстрирована ER-диаграммой.

Была представлена диаграмма прецедентов программного продукта.

Помимо этого, были представлены основные алгоритмы работы.

Концепт

1. ВВЕДЕНИЕ В ЭКСПЛУАТАЦИЮ
2. 1. Тестирование

Одним из наиважнейших этапов ведения проекта является проведение тестирования. Тестирование после разработки веб-сайта является неотъемлемой частью процесса его развертывания и играет важную роль в обеспечении качества и функциональности сайта перед его запуском [42].

Цель проведения тестирования состоит в проверке различных аспектов веб-сайта, включая его функциональность, надежность, производительность, безопасность и удобство использования.

Тестирование позволяет убедиться, что все функции и возможности веб-сайта работают правильно и соответствуют требованиям заказчика. Следует проверить, что все ссылки, формы, кнопки и другие элементы сайта функционируют должным образом и не вызывают ошибок.

Помимо этого, необходимо проверить, что содержимое отображается корректно, что страницы загружаются быстро, а также исправить проблемы совместимости с различными браузерами и устройствами.

Перед каждым испытанием проводилась тщательная оценка соответствия. Эта оценка включала в себя оценку разработанной функциональности и конкретных условий задачи, для решения которых предназначалась функциональность. Как только проверка дала положительный результат, был разработан подробный план испытаний. Этот исчерпывающий документ послужил основой для работы по обеспечению качества, предоставляя подробные сведения о процессе тестирования.

План испытаний включал несколько ключевых компонентов, в том числе подробное описание тестируемого объекта. Это описание дало четкое представление о тонкостях, связанных с тестированием конкретной функциональности. Кроме того, в плане указаны необходимые стратегии тестирования, которые могут варьироваться в зависимости от характера тестируемой функциональности. В нем также указаны критерии начала и завершения этапа тестирования, обеспечивающие структурированный подход на протяжении всего процесса.

Более того, в плане тестирования учитывались любые специальные знания или опыт, которые могли потребоваться для эффективного выполнения тестов. В нем перечислены основное оборудование и программное обеспечение, необходимые для облегчения процедур тестирования. Предоставляя эту исчерпывающую информацию, документ помог создать прочную основу для последующих этапов тестирования [44].

Когда план тестирования был готов, основное внимание было уделено этапу разработки тестового примера. Тестовые случаи играли ключевую роль в процессе тестирования, поскольку они описывали точные шаги и условия, необходимые для проведения тщательной оценки. По сути, каждый тестовый пример представлял собой сценарий, посредством которого будет тщательно проверяться функциональность, с подробными сведениями об условиях и параметрах, необходимых для оценки качества реализации.

Когда начался этап тестирования, по результатам тестов были собраны ценные данные. Эти данные послужили основой для создания отчетов об ошибках, в которых документировались любые ошибки, обнаруженные во время тестирования. Эти отчеты не только описывали конкретные ошибки, но и давали представление о последовательности действий, которые привели к таким неверным результатам. Кроме того, в отчетах об ошибках указывались ожидаемые результаты, что еще больше повышало ясность и точность документации.

Стоит отметить, что процесс тестирования всех элементов проекта выполнялся вручную, подчеркивая дотошность и внимание к деталям. Это включало строгие меры контроля качества, гарантирующие, что макет соответствует установленным правилам проектирования и общей структуре. Проверки включали проверку правильности отображения информации и проверку правильности назначения функций кнопкам [43].

Для обеспечения качества REST API осуществлялся мониторинг через визуальный интерфейс Swagger.

Swagger - это инструмент для разработки, документирования и тестирования API (интерфейса программного обеспечения). Он предоставляет набор инструментов и стандартов, которые помогают создавать, описывать и визуализировать API. Swagger обеспечивает автоматическую генерацию интерактивной документации API, что упрощает понимание и использование API [40].

Swagger имеет несколько преимуществ, которые делают его полезным инструментом:

* Автоматическая генерация документации
* Интерактивность и простота использования
* Удобное тестирование
* Однородность и стандартизация
* Возможность генерации кода
* Поддержка разных языков и фреймворков

Основная цель Swagger состоит в обеспечении единообразной спецификации API, которая позволяет разработчикам легко описывать, визуализировать и тестировать функции, параметры, запросы и ответы API. С помощью Swagger можно создать подробное описание API, включающее информацию о доступных маршрутах, параметрах, типах данных и примерах запросов и ответов. Кроме того, он обеспечивает возможность тестирования API прямо из его документации.

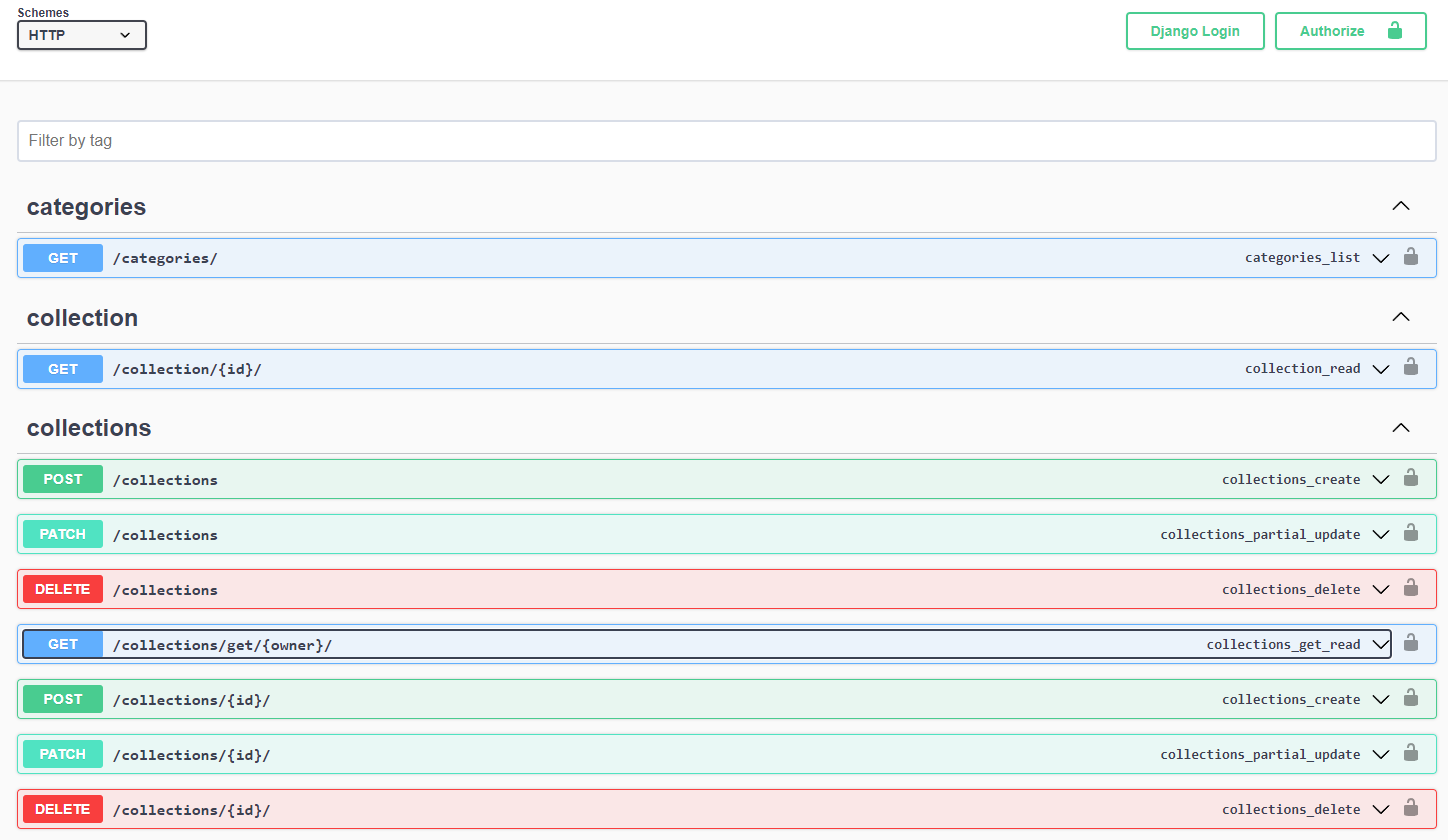


Рисунок 19 - Интерфейс Swagger UI

Для описания процесса тестирования важно упомянуть связь между OpenAPI и Swagger UI (Рисунок 19).

OpenAPI и Swagger UI являются тесно связанными компонентами, используемыми для документирования и визуализации API.

OpenAPI (ранее известный как Swagger) - это стандарт спецификации, который позволяет разработчикам описывать API в структурированном формате. Он определяет ресурсы, методы, параметры, типы данных и другие детали, необходимые для полного описания API.

Swagger UI, в свою очередь, представляет собой интерфейс пользователя, который автоматически создает документацию API на основе спецификации OpenAPI. Он позволяет разработчикам взаимодействовать с API, просматривать доступные ресурсы, выполнять запросы и получать ответы прямо в браузере.

Swagger UI предоставляет возможности для работы с различными методами HTTP, такими как GET, POST, PUT и DELETE и т.д.

Помимо этого, он предоставляет возможность разбивать запросы на различные части, такие как пути (path), параметры запроса (query parameters), заголовки (headers) и тело запроса (request body). Это позволяет пользователю точно определить, какие данные должны быть отправлены с запросом и какие ожидать в ответе. Разделение запросов улучшает читабельность, понятность и гибкость документации API.

Для проведения проверки необходимо использовать следующий алгоритм:

* Нажать кнопку «Try it out» («Попробовать»), чтобы открыть форму для заполнения параметров запроса. Ввести значения параметров, используя предоставленные поля ввода.
* После заполнения параметров нажать кнопку «Execute» («Выполнить»), чтобы отправить запрос к выбранному эндпоинту API с заданными параметрами.
* Swagger UI отобразит ответ от сервера, который включает информацию о статусе запроса, заголовках и теле ответа.
* После выполнения запроса можно вносить изменения в параметры запроса и повторять операцию для тестирования различных сценариев и проверки поведения API.

Статус ответа: Swagger UI отображает HTTP-статус ответа, такой как 200 OK (успешный запрос), 404 Not Found (ресурс не найден) или 500 Internal Server Error (внутренняя ошибка сервера). Этот статус указывает на результат выполнения операции.

Заголовки ответа: Swagger UI показывает заголовки, отправленные сервером в ответ на запрос. Заголовки могут содержать информацию, такую как тип содержимого (Content-Type), дата последней модификации, кэш и другие дополнительные метаданные.

Тело ответа: Swagger UI отображает тело ответа, которое содержит данные или ресурсы, возвращенные сервером. Это может быть JSON, XML, текстовая строка или другой формат данных, в зависимости от спецификации API и запроса, который был выполнен. Вы можете просмотреть и проанализировать эти данные, чтобы убедиться, что операция выполнилась правильно и вернула ожидаемый результат.

Полученные ответы сравниваются с ожидаемыми и на основании этого делаются выводы о наличии ошибок, корректности работы. В случае выявления ошибок предпринимаются меры для их исправления.

Тестирование верстки веб-сайта проводилось с целью проверки и обеспечения качества отображения и взаимодействия элементов на странице.

Важно убедиться, что все элементы на веб-странице правильно расположены, выровнены и отображаются согласно предполагаемому дизайну. Тестирование верстки позволяет проверить, что текст, изображения, кнопки, ссылки и другие элементы выглядят и функционируют должным образом.

Веб-сайт должен корректно отображаться в различных веб-браузерах, таких как Google Chrome, Mozilla Firefox, Safari, Internet Explorer и других. Необходимо проверить, как элементы и стили отображаются на разных браузерах и убедиться в их совместимости.

Помимо этого, тестирование верстки включает проверку правильности ссылок, переходов и навигации по веб-сайту. Это важно для обеспечения плавной и безошибочной навигации пользователей по различным разделам и страницам сайта.

* 1. Аспекты отказоустойчивости и надежности.

Отказоустойчивость (resilience) и uptime являются понятиями, связанными с надежностью и доступностью систем и услуг.

Отказоустойчивость относится к способности системы или сервиса продолжать нормальное функционирование, даже если происходят сбои или отказы в отдельных компонентах. Она предусматривает меры и механизмы, которые позволяют системе пережить сбои или отказы, минимизировать их влияние на работоспособность и оперативно восстанавливаться после них. Цель отказоустойчивости состоит в обеспечении непрерывной и надежной работы системы, даже в случае непредвиденных событий или ошибок.

Uptime (время работы или доступности) - это мера, указывающая на время, в течение которого система или сервис остаются доступными и функционируют без существенных перерывов. Он измеряется в процентах или временных единицах, указывая, сколько времени система была доступна и работала в сравнении с общим доступным временем. Чем выше процент uptime, тем более надежной и доступной считается система или сервис.

Уровень отказоустойчивости и uptime важны для многих критически важных систем и сервисов, таких как серверы, сети, облачные платформы, финансовые системы и др. Они гарантируют непрерывность бизнес-процессов, минимизацию потерь и удовлетворение требований пользователей к доступности и надежности.

Предположим, что в течение года (365 дней) было следующее время, когда приложение было недоступно:

* Плановое обслуживание: плановое обслуживание сервера, которое требовало его выключения на 2 часа каждый месяц. За год это составляет 2 часа \* 12 месяцев = 24 часа.
* Аппаратные сбои: Из-за проблем с аппаратным обеспечением сервера были сбои, которые занимали 1 час в месяц. За год это составляет 1 часа \* 12 месяцев = 12 часа.

Итак, общее время, когда ваше приложение было недоступно, составляет:

24 часа + 12 часов = 36 часов.

Теперь мы можем рассчитать uptime, используя формулу:   
Uptime (%) = (Время работы приложения / Общее доступное время) \* 100

Uptime (%) = (8724 / 8760) \* 100 = 99,6 %

99,6% uptime является довольно хорошим показателем и в большинстве случаев считается приемлемым. Это означает, что система может быть недоступна до 14,9 часов в году или примерно 1 час 15 минут в месяц. Для данной информационной системы такой уровень доступности является достаточным.

Для достижения более высокого уровня отказоустойчивости следует применять различные методы, такие как резервные компоненты или системы, распределение нагрузки между несколькими серверами или узлами, создание реплики данных на различных узлах или серверах, регулярне создание резервных копий данных и их восстановление и т.д.

* 1. Анализ внедрения продукта и оценка результатов

Цели, поставленные перед разработкой данного приложения, были ясно определены и направлены на достижение определенных результатов. Основной целью было создание программного продукта, обладающего определенным функционалом, который будет полезен и эффективен для конечных пользователей.

Для оценки результатов внедрения приложения была проведена детальная аналитика, включающая анализ основных характеристик и функциональности продукта. В результате этого анализа была составлена таблица, отражающая степень достижения поставленных целей.

Таблица 2 – Функциональный бенчмаркинг

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Mycollection | Mycoll | Avito | Разрабатываемый сайт |
| 1 | 2 | 3 | 4 | 5 |
| Создание коллекции | Есть | Есть | Нет | Есть |
| Добавление предмета | Есть | Есть | Есть | Есть |
| Разделение предметов по категориям | Есть | Есть | Есть | Есть |
| Поиск предметов по сайту | Нет | Есть | Есть | Есть |
| Разграничение доступа к предметам и коллекциям | Нет | Нет | Нет | Есть |
| Отправление обмена | Нет | Нет | Нет | Есть |
| Возможность указать номинальную цену предмета | Есть | Нет | Есть | Есть |

Из анализа следует, что разработанный программный продукт содержит в себе весь задуманный функционал. Каждая цель, поставленная перед проектом, была успешно достигнута, и продукт полностью соответствует требованиям, установленным в начальном этапе разработки.

Благодаря функциональности и эффективности продукта, можно сделать вывод, что он является высокоэффективным решением для задач, которые предполагалось решать с его помощью. Внедрение приложения позволяет пользователю выполнять требуемые задачи более эффективно и быстро, что существенно повышает их производительность и удовлетворенность результатами.

В целом, результаты анализа подтверждают успешное внедрение приложения и его соответствие поставленным целям. Пользователи получают доступ к полному функционалу продукта, что позволяет им эффективно выполнять свои задачи и достигать поставленных целей.

* 1. Планы развития

В текущей фазе разработки, программный продукт представляет себя в виде минимально жизнеспособного продукта (MVP), что означает, что у него есть потенциал для дальнейшего развития. Важной составляющей этого потенциала является микросервисная архитектура продукта. Такая архитектура обеспечивает гибкость при обновлении и улучшении веб-приложения, позволяя легко интегрировать новые микросервисы с дополнительной функциональностью или обновить уже существующие микросервисы без прерывания работы всего приложения.

Среди планов для развития программного продукта можно выделить следующие:

* Сравнение предметов - функция сравнения предметов, добавляемая на сайт для обмена коллекционными предметами, позволит пользователям легко сравнивать характеристики разных предметов и принимать информированные решения об обмене.
* Рейтинг и отзывы – функция позволит пользователям оставлять рейтинги и отзывы о предметах и других пользователях, с которыми они имели дело. Это поможет другим пользователям принять информированное решение при обмене предметами и создаст доверие в сообществе коллекционеров.
* Аукционы - система аукционов, где пользователи смогут участвовать в торгах на интересующие их предметы и предлагать свои предметы на аукцион. Это добавит элемент соперничества и возможность получить более ценные предметы.
* Комментарии и обсуждения - возможность оставлять комментарии и участвовать в обсуждениях на страницах предметов. Пользователи смогут делиться своими мнениями, опытом и знаниями о конкретных предметах, что будет ценной информацией для других коллекционеров.
* История изменений предметов - история изменений каждого предмета, чтобы пользователи могли отслеживать его происхождение, владельцев и изменения в характеристиках. Это поможет подтвердить подлинность предметов и улучшит доверие в процессе обмена.
* Интеграция с социальными сетями - возможность пользователям легко делиться своими коллекциями и предложениями обмена через популярные социальные сети. Это способствует распространению информации о сайте и привлечению новых пользователей.
* Подписки и уведомления о новых предметах - возможность подписываться на определенные категории предметов или конкретные поисковые запросы, чтобы получать уведомления о новых предметах, соответствующих их интересам. Это упростит процесс поиска и повысит удобство пользования сайтом.
* Доставка - функция интеграции сервиса доставки на сайте позволит пользователям удобно и безопасно осуществлять доставку своих предметов в процессе обмена.

Выше приведенный список также стоит дополнить пунктами для улучшения пользовательского опыта от работы сайтом:

* Адаптивная верстка - разработка адаптивной верстки позволит сайту автоматически адаптироваться к различным размерам экранов. Это обеспечит удобную навигацию и читаемость контента на планшетах и мобильных устройствах, улучшая пользовательский опыт.
* Интерактивные элементы касания - Добавление интерактивных элементов касания, таких как свайпы, жесты и увеличение/уменьшение масштаба прикосновением, позволит пользователям более естественным образом взаимодействовать с сайтом на мобильных устройствах. Это улучшит удобство пользования и повысит пользовательский опыт.
* Упрощенное меню - при адаптации сайта для мобильных устройств предусмотрите упрощенное и удобное меню навигации. Использование выпадающих списков, иконок или скрытых панелей обеспечит легкий доступ к основным разделам сайта, минимизируя занимаемое пространство.
* Оптимизация – Быстрая загрузка страниц, особенно для мобильных устройств, где скорость интернета может быть медленнее. Сжатие изображений, минимизация кода и использование кэширования ускорят загрузку страниц и улучшат общую производительность сайта.
* Уведомления на мобильных устройствах - возможность пользователей получать уведомления на своих мобильных устройствах о новых предметах, сообщения
  1. Выводы

В данной главе описан процесс тестирования приложения.

Изучены аспекты отказоустойчивости приложения, а также произведены примерные расчеты соответствующих показателей.

Приведен анализ внедрения продукта и оценка результатов.

Приведено описание функций, которые планируется в будущем внедрить в приложение.

СПИСОК ЛИТЕРАТУРЫ

Список нормативно-правовых документов

1. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению
2. ГОСТ 19.301-79 Программа и методика испытаний. Требования к содержанию и оформлению
3. ГОСТ 34.602-89 Информационная технология (ИТ). Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы
4. ГОСТ Р 53633.3-2009 Информационные технологии. Сеть управления электросвязью
5. Федеральный закон от 27.07.2006 N 149-ФЗ (ред. от 09.03.2021) "Об информации, информационных технологиях и о защите информации" (с изм. и доп., вступ. в силу с 20.03.2021)
6. Трудовой кодекс Российской Федерации от 30.12.2001 N 197-ФЗ (ред. от 25.02.2022) (с изм. и доп., вступ. в силу с 01.03.2022)
7. Федеральный закон РФ от 27 июля 2006 г. No 152-ФЗ «О персональных данных» — федеральный закон, регулирующий деятельность по обработке (использованию) персональных данных. Обязательно должен быть учтен при создании приложений, использующих формы регистрации или таблицы с информацией о сотрудниках

Книжные издания и издания в периодике

1. Петухов, А. Коллекционер: Психологические аспекты увлечения: Монография / А. Петухов. - СПб.: Когито-Центр, 2018. - 192 с.
2. Мартин, Р. Чистая архитектура. Книга о том, как вести разработку программного обеспечения. Москва: ДМК Пресс, 2018. - 432 с.
3. Функциональное моделирование на базе стандарта IDEF0: метод. указания / сост. Д.Ю. Киселев, Ю.В. Киселев, А.В. Вавилин. – Самара: Изд-во СГАУ, 2019. – 20 с.
4. Яковлев, И. В. Проектирование программного обеспечения. М.: Финансы и статистика, 2018. 352 с.
5. Разработка диаграммы прецедентов web-сайта / Н.В. Картечина, Р.Н. Абалуев, В.А. Шацкий, А.М. Дорохова // Наука и Образование. – 2021. – Т. 4. – No 1.
6. Литвиненко, Ю. С. Проектирование пользовательского интерфейса / Ю. С. Литвиненко. – М.: Бином, 2018. – 408 с.
7. Борисова, И. В. Графический пользовательский интерфейс веб-приложений / И. В. Борисова. – М.: Издательский дом "Лори", 2020. – 192 с.
8. Николаев, И. А. Дизайн веб-сайта: от идеи до результата / И. А. Николаев. – М.: БХВ-Петербург, 2021. – 368 с.
9. Голованов, А. В. Оптимизация пользовательского интерфейса веб-сайта / А. В. Голованов. – М.: Финансы и статистика, 2020. – 224 с.
10. Белкин, А. В. Пользовательский интерфейс веб-приложений / А. В. Белкин. – Москва : ДМК Пресс, 2018. – 288 с.
11. Л.К. Белезяк, Р.А. Карелова Бриф как основа технического задания на разработку программного обеспечения // Молодежь и наука: материалы международной научно-практической конференции старшеклассников, студентов и аспирантов, Нижний Тагил, 28 мая 2021 года / Уральский федеральный университет имени первого Президента России Б.Н. Ельцина. – Нижний Тагил: Уральский федеральный университет имени первого Президента России Б.Н. Ельцина, 2021. – С. 9-12.
12. Захаров, В.Б. Проблемы выбора языков программирования при разработке кроссплатформенных приложений / В.Б. Захаров, М.Г. Мальковский, А.И. Мостяев // International Journal of Open Information Technologies. – 2018. – Т. 5. – No 7. – С. 29-37.
13. Назарова, О. Моделирование бизнес-процессов: учебно- методическое пособие / О. Назарова, О.Е. Масленникова. – Магнитогорск: Изд-во Магнитогорск. гос. техн. ун-та им. Г.И. Носова, 2015. – 230 с.

Интернет-ресурсы

1. Документация Vue.js [Электронный ресурс] / Vue.js. - URL: https://vuejs.org/. – (дата обращения: 01.04.2023).
2. Vue.js Reddit Community [Электронный ресурс]. - URL: https://www.reddit.com/r/vuejs/. - (дата обращения: 01.04.2023)
3. Django Documentation [Электронный ресурс] / Django Software Foundation. - Режим доступа: URL: https://docs.djangoproject.com/. – (дата обращения: 01.04.2023).
4. TypeScript Documentation [Электронный ресурс] / Microsoft Corporation. - Режим доступа: URL: https://www.typescriptlang.org/docs/. - (дата обращения: 02.04.2023).
5. PostgreSQL Documentation [Электронный ресурс] / PostgreSQL Global Development Group. - Режим доступа: URL: https://www.postgresql.org/docs/. - (дата обращения: 01.04.2023).
6. MySQL Documentation [Электронный ресурс] / Oracle Corporation. - Режим доступа: URL: https://dev.mysql.com/doc/. - (дата обращения: 02.04.2023).
7. Oracle Database Documentation [Электронный ресурс] / Oracle Corporation. - Режим доступа: URL: https://docs.oracle.com/en/database/. - (дата обращения: 02.04.2023).
8. HTML Forms Guide [Электронный ресурс] / Mozilla Developer Network. - Режим доступа: URL: https://developer.mozilla.org/en-US/docs/Learn/Forms. - (дата обращения: 02.04.2023).
9. BPMN.org. [Электронный ресурс] / BPMN Specification. Режим доступа URL: http://www.bpmn.org/. (дата обращения: 12.04.2023).
10. UX Design [Электронный ресурс] / Interaction Design Foundation. - Режим доступа: URL: https://www.interaction-design.org/literature/topics/ux-design. - (дата обращения: 19.04.2023).
11. Модель сущность-связь [Электронный ресурс]-Режим доступа: http://www.mstu.edu.ru/study/materials/zelenkov/ch\_2\_4.html(дата обращения: 30.04.2023)
12. Er-диаграммы: [Электронный ресурс] / Studfiles, файловый архив студентов. https://studfiles.net/preview/400285/page:3/. (Дата обращения: 23.04.2023)
13. Нотация IDEF0 [Электронный ресурс] — Режим доступа: URL: https://www.businessstudio.ru/wiki/docs/v4/doku.php/ru/csdesign/bpmodeling/idef0 (дата обращения: 10.05.2023)
14. Проектирование графического интерфейса пользователя [Электронный ресурс]-Режим доступа: https://habr.com/ru/post/208966/ (дата обращения: 10.05.2023)
15. Database Management Systems (DBMS) - GeeksforGeeks [Электронный ресурс] - Режим доступа: URL: https://www.geeksforgeeks.org/dbms-entity-relationship-model/. (дата обращения: 10.05.2022)
16. Документация по Django REST Framework [Электронный ресурс]. Режим доступа: URL: https://www.django-rest-framework.org/. (дата обращения: 11.05.2023)
17. Richardson, L. RESTful Web Services [Электронный ресурс]. O'Reilly Media, Режим доступа: URL: https://www.oreilly.com/library/view/restful-web-services/9780596529260/. (дата обращения: 11.05.2023)
18. Байдыбеков А. А. Современные фреймворки для разработки web- приложений / А. А. Байдыбеков, Р. Г. Гильванов, И. А. Молодкин // Интеллектуальные технологии на транспорте. — 2020. — No 4 (24). URL: https://cyberleninka.ru/article/n/sovremennye-freymvorki-dlya-razrabotki-web- prilozheniy (дата обращения: 11.05.2023)
19. Данилин Д. А. Анализ технологий веб-программирования для создания модулей визуализации и выгрузки данных информационных систем / Д. А. Данилин, Я. В. Зиновьев, К. М. Кузьмин // Вестник Пензенского государственного университета. — 2019. — No 3 (27). URL: https://cyberleninka.ru/article/n/analiz-tehnologiy-veb-programmirovaniya-dlya- sozdaniya-moduley-vizualizatsii-i-vygruzki-dannyh-informatsionnyh-sistem (дата обращения: 11.05.2023).
20. Топ-5 JS-фреймворков для фронтенд-разработки в 2020 году программирования [Электронный ресурс] – Режим доступа: https://habr.com/ru/company/ruvds/blog/476286/ (дата обращения: 12.05.2023).
21. Документация Swagger [Электронный ресурс]. Режим доступа: URL: https://swagger.io/docs//. (дата обращения: 12.05.2023)
22. Тестирование приложений [Электронный ресурс]. Режим доступа: URL: https://quality-lab.ru/blog/key-principles-of-web-testing/ (дата обращения 12.05.2023) – Загл. с экрана.
23. Особенности тестирования веб-приложений [Электронный ресурс]. — Режим доступа: URL: https://quality-lab.ru/blog/key-principles-of-web-testing/, свободный (дата обращения: 12.05.2023). — Загл. с экрана.
24. Ручное тестирование [Электронный ресурс] – Режим доступа: URL: https://www.appline.ru/services/testing/functionalnoe-testirovanie/ruchnoe-testirovanie (дата обращения 12.05.2023).
25. Нотация BPMN [Электронный ресурс] – Режим доступа: URL: https://www.businessstudio.ru/wiki/docs/current/doku.php/ru/csdesign/bpmodeling/bpmn\_notation (дата обращения 12.05.2023) – Загл. с экрана.
26. Документация Vuex / [Электронный ресурс]. – Режим доступа: URL: https://vuex.vuejs.org/ (дата обращения: 12.05.2023). \*

Иностранные издания

1. Web Design: Introductory / Jennifer T. Campbell. - Boston: Cengage Learning, 2018. - 528 p.
2. Guillaume C., Vue.js 2 Web Development Projects" by Guillaume Chau. - Packt Publishing, 2018. - 380 с.
3. William S. Vincent Django for Beginners. - Independently published, 2018. - 290 с.
4. Gastón C. Hillar Django RESTful Web Services. - Packt Publishing, 2018. - 374 с.

ПРИЛОЖЕНИЯ

Приложение А

**Техническое задание**

**1 ОБЩИЕ ПОЛОЖЕНИЯ**

**1.1 Полное наименование системы и ее условное обозначение**

Веб-сервис для обмена коллекционными предметами

**1.2 Перечень документов, на основании которых создаётся система**

Основой для разработки служат основная профессиональная образовательная программа по направлению 09.03.01 «Информатика и вычислительная техника», профиль «Веб-технологии», учебный план и приказ на утверждение темы ВКР и научного руководителя.

**1.3 Плановые сроки начала и окончания работы по созданию системы**

Плановый срок начала разработки информационной системы «Веб-сервис для обмена коллекционными предметами» для студенческого городка Московского Политеха» – 28.03.2021.

Плановый срок окончания работ по разработке информационной системе «Веб-сервис для обмена коллекционными предметами» – 01.06.2021.

**1.4 Порядок оформления и предъявления заказчику результатов работ по созданию системы**

Система передается в виде функционирующего комплекса на базе средств Заказчика и Исполнителя в сроки, установленные Договором. Приемка системы осуществляется заказчиком.

Результаты передаются заказчику частями по завершении каждой стадии работы по созданию системы.

Документация – в электронном виде в формате .doc, .docx и на бумажных носителях.

**1.5 Состав используемой нормативно-технической документации**

При разработке автоматизированной системы и создании проектно-эксплуатационной документации Исполнитель должен руководствоваться требованиями следующих нормативных документов:

ГОСТ 34.601-90. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания;

ГОСТ 34.201-89. Информационная технология. Комплекс стандартов на автоматизированные системы. Виды, комплексность и обозначение документов при создании автоматизированных систем;

РД 50-34. 698-90. Методические указания. Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Требования к содержанию документов;

ГОСТ 7.32-2017 СИБИД. Отчет о научно-исследовательской работе. Структура и правила оформления.

**2 НАЗНАЧЕНИЕ И ЦЕЛИ СОЗДАНИЯ СИСТЕМЫ**

**2.1 Назначение системы**

Информационная система «Веб-сервис для обмена коллекционными предметами» автоматизирует процессы обмена и поиска коллекционных предметов.

**2.2 Цели создания системы**

Цели создания системы:

* автоматизация процесса поиска коллекционных предметов;
* автоматизация процесса просмотра коллекционных предметов;
* автоматизация процесса обмена коллекционными предметами;
* автоматизация процесса просмотра коллекций предметов.

**3 ХАРАКТЕРИСТИКА ОБЪЕКТА АВТОМАТИЗАЦИИ**

**3.1 Объект автоматизации**

Разработка информационной системы «Веб-сервис для обмена коллекционными предметами».

Для автоматизации разработки базы данных необходимы следующие специалисты:

* программист;
* администратор БД;
* системный администратор.

Процесс разработки системы также включает в себя следующие этапы:

* проектирование системы;
* создание технического задания;
* программирование.

**4 ТРЕБОВАНИЯ К СИСТЕМЕ**

**4.1 Требования к системе в целом**

**4.1.1 Перспективы развития, модернизации системы**

Информационная система «Веб-сервис для обмена коллекционными предметами» будет иметь дальнейшие перспективы развития, путем добавления новых сервисов, позволяющих использовать дополнительные функции для автоматизации процессов.

**4.1.2 Требования к численности и квалификации персонала**

Для эксплуатации научно-справочной системы были выявлены следующие роли:

* системный администратор;
* администратор информационной безопасности;
* оператор БД.

В обязанности системного администратора будет входить следующее:

* обеспечение безошибочной работы системы;
* обеспечение работоспособности системы и базового программного
* обеспечения;
* обеспечение резервного копирования данных.

В обязанности оператора баз данных входит следующее:

* установка, модернизация, настройка параметров программного
* обеспечения СУБД;
* оптимизация прикладных баз данных по времени отклика, скорости
* доступа к данным;
* разработка, управление и реализация эффективной политики доступа к информации, хранящейся в прикладных базах данных.

В обязанности администратора информационной безопасности входит следующее:

* поддержание системы в рамках выбранной политики безопасности;
* обеспечение должного уровня конфиденциальности и целостности данных;
* периодическое практическое тестирование защищенности системы;
* документирование проделанной работы;
* устранение неполадок в системе.

**4.1.3 Требования к показателям назначения**

Информационная система «Веб-сервис для обмена коллекционными предметами» должна обеспечивать возможность исторического хранения данных с глубиной не менее 10 лет, если иного не указано в распорядительных документах.

Система должна предусматривать возможность масштабирования по производительности и объему обрабатываемой информации без модификации ее программного обеспечения путем модернизации используемого комплекса технических средств. Возможности масштабирования должны обеспечиваться средствами используемого базового программного обеспечения.

**4.1.4 Требования к надежности**

Система должна сохранять работоспособность и обеспечивать восстановление своих функций при возникновении следующих внештатных ситуаций:

* система должна предусматривать базовую защиту от основных видов атак: SQL-инъекций;
* система может быть недоступна не более чем 24 часа в год.
* У администратора сайта должна быть возможность выгрузить и загрузить копию справочника.
* Для защиты аппаратуры от бросков напряжения и коммутационных помех должны применяться сетевые фильтры.

**4.1.5 Требования к безопасности**

Все внешние элементы технического оборудования, находящиеся под напряжением, должны быть защищены от случайного прикосновения, а само техническое оборудование имеет нейтральную или защитную основу по ГОСТ 12.1.030-81 и ПУЭ.

Система электроснабжения должна обеспечивать профилактическое отключение при перегрузке и коротком замыкании цепей нагрузки, а также отключение вручную в случае аварии.

Общие требования пожарной безопасности соответствуют нормам для бытового электрооборудования. В случае пожара нельзя выделять токсичные газы и пары. После отключения источника питания допускается использование средств пожаротушения.

Факторы, оказывающие вредные воздействия на здоровье со стороны всех элементов системы (в том числе инфракрасное, ультрафиолетовое, рентгеновское и электромагнитное излучения, вибрация, шум, электростатические поля, ультразвук строчной частоты и т.д.), не должны превышать действующих норм (СанПиН 2.2.2./2.4.1340-03 от 03.06.2003 г.).

**4.1.6 Требования к эксплуатации, техническому обслуживанию, ремонту и хранению компонентов**

Система должна быть спроектирована для работы в составе программно-аппаратного комплекса заказчика и учитывать разделение ИТ-инфраструктуры заказчика на внутреннюю и внешнюю. Техническая и физическая защита аппаратных компонентов системы, носителей данных, источников бесперебойного питания, резервирование ресурсов и текущее обслуживание осуществляется техническими и организационными средствами, предусмотренными в ИТ-инфраструктуре заказчика.

Для нормальной работы разрабатываемой системы требуется источник бесперебойного питания для ПК. Во время работы система должна иметь температуру и влажность, соответствующие стандартам хранения носителей и работы ПК.

Заказчик обязан следить за техническим состоянием оборудования, в случае возникновения технических неисправностей - своевременно проводить техническое обслуживание, а также проводить периодическое техническое обслуживание в соответствии с рекомендациями производителя оборудования, обеспечивать постоянную чистоту. сервера, а также обеспечить соблюдение всех условий эксплуатации, предусмотренных производителем.

Подрядчик не несет ответственности за убытки, понесенные в процессе работы и / или бездействие заказчика при поддержании и поддержании условий эксплуатации.

Регулярное обслуживание и тестирование технического оборудования должно включать обслуживание и тестирование всего используемого оборудования, включая рабочие станции, серверы, кабели, сетевое оборудование и источники бесперебойного питания.

В процессе проведения периодического технического обслуживания следует проводить внешний и внутренний осмотр и чистку технического оборудования, проверять контакты контактов, проверять параметры исправности настроек технического оборудования и тестировать их взаимодействие.

Восстановление работоспособности технических средств осуществляется в соответствии с инструкциями разработчика и поставщика технических средств и документации по восстановлению работоспособности технических средств и должно завершаться их испытанием. Расположение зданий и их оборудования должно исключать возможность неконтролируемого проникновения посторонних лиц и обеспечивать сохранность конфиденциальных документов и технических средств, содержащихся в этих зданиях.

Размещение оборудования и технических средств должно соответствовать требованиям безопасности, санитарной и пожарной безопасности.

Все пользователи системы должны соблюдать правила работы с электронными вычислительными машинами.

Квалификация и подготовка персонала должны соответствовать технической документации.

**4.1.8 Требования к защите информации от несанкционированного доступа**

Система должна обеспечивать защиту от несанкционированного доступа (НСД) на уровне не ниже уровня, установленного требованиями класса 1Д по классификации действующего методического документа ГТК России «Автоматизированные системы. Защита от несанкционированного доступа. к информации.

Компоненты подсистемы защиты от несанкционированного доступа должны обеспечивать:

1. идентификация пользователя;
2. проверка прав пользователя при работе с системой;
3. разграничение доступа пользователей на уровне задач и информационных матриц.

Журналы аудита системы и приложений должны быть защищены от несанкционированного доступа локально и в архиве.

Уровень защиты от несанкционированного доступа к вычислительным средствам, обрабатывающим конфиденциальную информацию, должен соответствовать требованиям класса безопасности 6 в соответствии с требованиями действующего руководящего документа Гостехкомиссии России «Средства вычислительной техники. Защита от несанкционированного доступа к информации. Показатели защищенности от несанкционированного доступа к информации».

Защищенная часть системы должна использовать «скрытые» пароли (при наборе пароля его символы не отображаются на экране или заменяются символом одного типа; количество символов не соответствует длине пароля).

Защищенная часть системы должна автоматически блокировать пользовательские сеансы и сеансы приложений на предустановленное время простоя для пользователей и приложений.

Защищаемая часть системы должна использовать многоуровневую систему безопасности. Безопасная часть системы должна быть отделена от незащищенной части системы брандмауэром.

**4.1.9 Требование по сохранности информации при авариях**

Программа должна восстановить свою работоспособность после безопасного перезапуска устройства. Должна быть предусмотрена возможность организации автоматического резервного копирования системных данных системным и базовым программным обеспечением (ОС, СУБД), входящим в программно-аппаратный комплекс заказчика.

Вышеуказанные требования не распространяются на компоненты системы, разработанные третьими сторонами, и действуют только при соблюдении правил использования этих компонентов.

**4.1.10 Требования по патентной чистоте**

Установка системы в целом, а также установка отдельных частей системы не должны предъявлять дополнительных требований к покупке сторонних лицензий на программное обеспечение, за исключением программного обеспечения.

**4.2 Требования к функциям, выполняемым системой**

**4.2.1 Администратор информационной безопасности**

Администратор информационной безопасности поддерживает систему в рамках выбранной политики безопасности, обеспечивает должный уровень конфиденциальности и целостности данных.

**4.2.2 Системный администратор**

Администратор отвечает за работоспособность системы в целом.

**4.2.3 Оператор БД**

Данная роль относится к категории обслуживания. Он отвечает за достоверность и актуальность базы данных. Оператор так же выполняет функции, связанные с обработкой запросов на предоставление БПД.

**4.2.2 Описание процессов и функций работы с системой**

Процессы и функции, выполняемые при эксплуатации системы, приведены в разбивке по подсистемам.

* подсистема администрирования БД;
* подсистема администрирования.

Процессы, выполняемые под управлением различных подсистем ИС узла, реализуются на основе системных процедур, которые являются составной частью функций пользователей системы.

Администрирование БД предполагает реализацию процессов, связанных с управлением БД РИПД, и включает следующие функции:

1. создание базы данных;
2. создание резервных копий;
3. проверка целостности БД;
4. проверка работоспособности БД;
5. разграничение доступа к БД.

Подсистема администрирования доступа.

Функции подсистемы администрирования доступа объединяют функционал системы по обеспечению функций ограничения доступа и сохранности информационных ресурсов системы.

**4.3 Требования к видам обеспечения**

**4.3.1 Требования к информационному обеспечению системы**

Состав, структура и методы организации данных в системе должны быть определены на этапе технического проектирования.

Хранение данных должно быть основано на современных системах управления реляционными базами данных. Для обеспечения целостности данных необходимо использовать механизмы, встроенные в СУБД.

Средства системы управления базой данных, а также средства используемых операционных систем должны обеспечивать документирование и регистрацию информации, которая обрабатывается в системе.

Структура базы данных должна поддерживать кодирование хранимой информации и обработку по общероссийским классификаторам (где применимо).

Доступ к данным предоставляется только авторизованным пользователям в соответствии с их официальными полномочиями, а также в соответствии с категорией требуемой информации.

Структура базы данных должна быть организована рационально, за исключением единовременной полной загрузки информации, содержащейся в базе данных системы.

В системе должна быть специализированная подсистема резервного копирования и восстановления данных.

При проектировании и развертывании системы крайне важно учитывать потенциальное использование информации, накопленной из уже действующих информационных систем.

**4.3.2 Требования к лингвистическому обеспечению**

Все прикладное программное обеспечение системы для организации взаимодействия с пользователем должно использовать русский язык.

**4.3.3 Требования к программному обеспечению системы**

Используемое при разработке программное обеспечение и библиотеки программных кодов должны иметь широкое распространение.

Дополнительных требований к базовой программной платформе не предъявляется.

**4.3.4 Требование к техническому обеспечению**

Техническое обеспечение системы должно максимально и наиболее эффективным образом использовать существующие технические средства.

**4.3.5 Требования к организационному обеспечению**

 Организационное обеспечение системы должно быть достаточным для эффективного выполнения персоналом возложенных на него обязанностей при осуществлении автоматизированных и связанных с ними неавтоматизированных функций системы. Заказчиком должны быть определены должностные лица, ответственные за:

* обработку информации;
* администрирование системы;
* обеспечение безопасности информации.

К работе с системой должны допускаться сотрудники, имеющие навыки работы на персональном компьютере, ознакомленные с правилами эксплуатации и прошедшие обучение работе с системой.

**5 СОСТАВ И СОДЕРЖАНИЕ РАБОТ ПО СОЗДАНИЮ МОДУЛЯ**

Осуществление всего комплекса работ должно осуществляться в несколько очередей. Спецификация работ по созданию «Веб-сервис для обмена коллекционными предметами» в объеме требований настоящего ТЗ приведена в [таблице А.1](https://docs.google.com/document/d/1ZrUS3gketExfT6CtNIAiB4CNrdrYMy3F/edit#bookmark=id.1opuj5n).

Таблица А.1 – Спецификация работ

|  |  |  |  |
| --- | --- | --- | --- |
| Стадия работ | Выполняемые работы | Сроки | Итоги выполнения работы |
| Формирование требований | Обследование объекта автоматизации | С 28.03.2021 по 30.03.2021 | Отчет о результатах проделанной работы. |
| Проектирование | Разработка технического задания | С 01.04.2021 по 04.04.2021 | Документация на разрабатываемую систему |
| Разработка модели программы | Разработка модели программы | С 05.04.2021 по 20.04.2021 | Описание спецификаций данных, определение связей между сущностями, построение концептуальной модели БД, построение логической модели БД. |
| Разработка проекта | Написание программного продукта | С 21.04.2021 по 18.05.2021 | Программный продукт на этапе разработки. |
| Тестирование | Проведение тестирования и доработка программного продукта | С 20.05.2021 по 01.06.2021 | Отчет о проведении тестирования |
| Сдача готового продукта | Сдача разработанной системы | С 02.06.2021 по 10.08.2021 | Итоговый комплект наработанных материалов, включающий в себя бумажный оригинал документов и электронный носитель. |

Типовой состав работ по развертыванию, внедрению и опытной эксплуатации, который должен быть предусмотрен в ЧТЗ на развертывании, приведен в [таблице А.2.](https://docs.google.com/document/d/1ZrUS3gketExfT6CtNIAiB4CNrdrYMy3F/edit#bookmark=id.48pi1tg)

Таблица А.2 – Типовой состав работ

| Виды работ | Состав работ |
| --- | --- |
| Подготовка регламентов применения | Регламент применения пользовательской подсистемы |
| Регламент применения системы |
| Обучение | Обучение пользователей |
| Развертывание системы | Установка серверного ПО |
| Настройка процедур резервного копирования |
| Внедрение системы | Разворачивание сервисов на сервере |
| Настройка процессов |
| Проведение испытаний на объекте, передача в опытную эксплуатацию |
| Опытная эксплуатация | Техническая поддержка в течении опытной эксплуатации |
|  | Устранение ошибок |

**6 ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ СИСТЕМЫ**

Испытания системы должны проводиться в соответствии с требованиями ГОСТ 34.603-92 «Информационная технология. Виды испытаний автоматизированных систем». При реализации Системы в рамках настоящего ТЗ устанавливаются предварительные испытания на стенде Исполнителя по созданию Системы.

Испытания Системы должны осуществляться в соответствии с документом «Программа и методика испытаний», который должен устанавливать необходимый и достаточный объем испытаний, обеспечивающий требуемый уровень достоверности получаемых результатов. Программа и методика испытаний утверждается Заказчиком.

Приемку работ должна осуществлять приемочная комиссия, в состав которой входят:

1. представители Заказчика;
2. представители Исполнителя.

При проведении испытаний приемочной комиссии предъявляются разработанные Исполнителем материалы. Комплектность предоставляемой документации определяется требованиями настоящего ТЗ.

Предварительные испытания заканчиваются подписанием приемочной комиссией протокола испытания с указанием в нем перечня необходимых доработок программного обеспечения, программной и эксплуатационной документации и сроков их выполнения.

После устранения замечаний осуществляются повторные предварительные испытания Подсистемы. Испытания завершаются оформлением Акта готовности системы к эксплуатации.

Отдельные пункты ТЗ могут изменяться и уточняться по согласованию сторон.

**7 ТРЕБОВАНИЯ К ДОКУМЕНТИРОВАНИЮ**

**7.1 Общие требования к документированию**

Документы должны быть предоставлены в бумажном виде и на электронном носителе. Исходные тексты программ – только на электронном носителе.

Все документы должны быть оформлены на русском языке.

Для системы на различных стадиях создания должны быть выпущены документы, соответствующие ГОСТ 34.201- 89.

**7.2 Перечень подлежащих разработке документов**

При создании системы необходимо подготовить и доставить заказчику проектную документацию и проектные материалы для разработки системы;

Подготовительные документы должны соответствовать требованиям государственных стандартов и рекомендаций по оформлению, содержанию, согласованию, определений, надписей, обозначений программ и программных документов.

**8** **ТРЕБОВАНИЯ К СОСТАВУ И СОДЕРЖАНИЮ РАБОТ ПО ПОДГОТОВКЕ ОБЪЕКТА АВТОМАТИЗАЦИИ К ВВОДУ СИСТЕМЫ В ДЕЙСТВИЕ**

При реализации проекта на объекте автоматизации требуются работы по подготовке системы к работе. При подготовке к эксплуатации системы заказчик должен обеспечить выполнение следующих действий:

* определить подразделение и ответственных должностных лиц, ответственных за внедрение и проведение опытной эксплуатации системы;
* обеспечить присутствие пользователей на обучении работе с системой, проводимом Исполнителем;
* обеспечить соответствие помещений и рабочих мест пользователей системы в соответствии с требованиями, изложенными в настоящем ЧТЗ;
* обеспечить выполнение требований, предъявляемых к программно-техническим средствам;
* совместно с Исполнителем подготовить план развертывания системы на технических средствах Заказчика;
* провести опытную эксплуатацию системы.

Требования к составу и содержанию работ по подготовке объекта автоматизации к работе системы, включая перечень основных мероприятий и их исполнителей, следует уточнить на этапе подготовки рабочих документов и по результатам тестового запуска.

**9 ПОРЯДОК ВНЕСЕНИЯ ИЗМЕНЕНИЙ**

Настоящее ТЗ может дополняться и изменяться в процессе разработки и приемочных испытаний в установленном порядке по взаимному соглашению.