**Введение**

**Актуальность работы**  
Филателия как область коллекционирования требует систематизации больших объемов данных: учета марок, их характеристик, владельцев и рыночной стоимости. В современных условиях ручное ведение каталогов становится неэффективным из-за риска ошибок, сложности поиска и отсутствия оперативной аналитики а также объема данных. Эта работа посвящена разработке веб приложения для автоматизации учета марок, коллекций и взаимодействия с коллекционерами. Решение позволит не только упростить процессы управления данными, но и предоставит инструменты для анализа редких экземпляров и формирования отчетов.

**Цель работы**  
Разработка информационной системы «Справочник филателиста» на платформе. Курсовой проект нацелен на изучение и понимание работы систем с пользователями. Так же понимание предметной области в технологических решениях.

**Задачи работы**

Задачи работы заключаются в анализе предметной области (учет коллекций) и изучение-проектирование и создание решений для коллекционеров. Так же нужно правильно сформулировать требования к функционалу системы.  
Не мало важным является и выбор технологий реализации программного продукта

**Объект исследования**

Процессы автоматизации учета марок, а также коллекций и взаимодействия с коллекционерами. В приложении реализуется взаимодействие между всеми сущностями

**Предмет исследования**

WEB-приложение для управления данными филателистической коллекции.

**Методы исследования**

Анализ и сравнение существующих решений, моделирование базы данных для хранения и структуризации данных.

**Информационная база исследования**

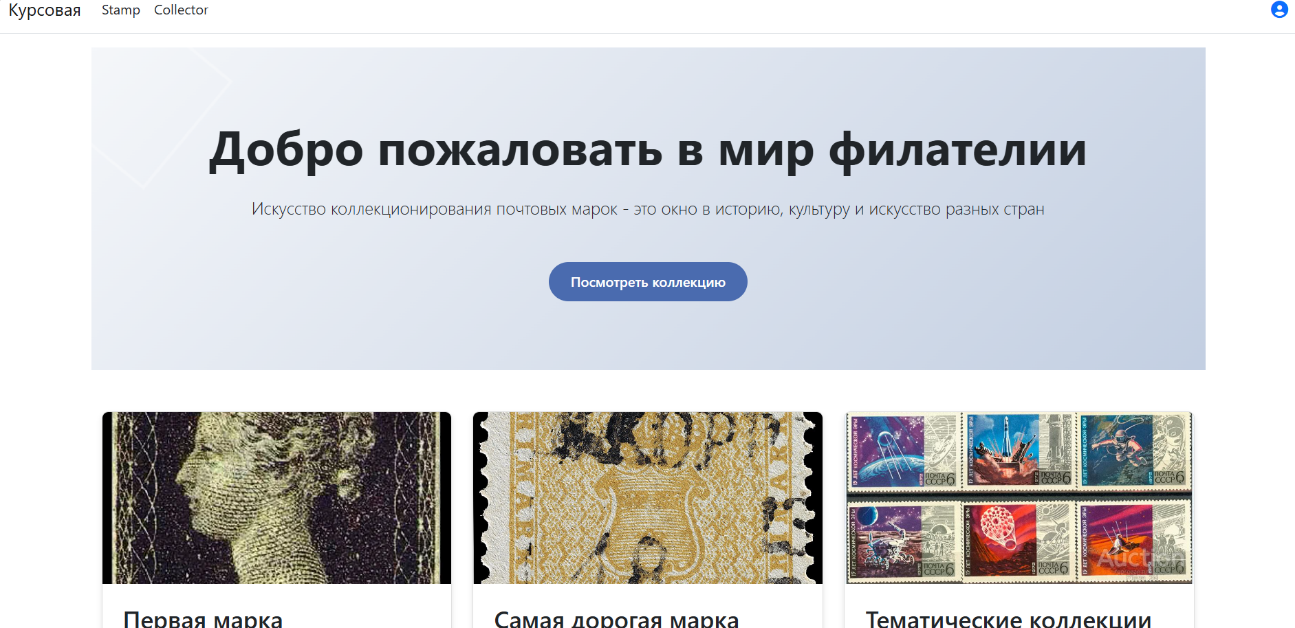
Документация по ASP.NET Core и SQL Server, материалы курса «Создание программного обеспечения».

**1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

**1.1 Анализ предметной области**

*Филателия* — система учета, где ключевыми объектами являются марки, коллекции и коллекционеры. Каждая марка имеет уникальные атрибуты: страну выпуска, номинальную стоимость, год, тираж, тематику и рыночную цену. Коллекционеры формируют личные собрания, включающие редкие экземпляры (стоимостью свыше 1000 руб.), что требует отслеживания принадлежности и ценности.

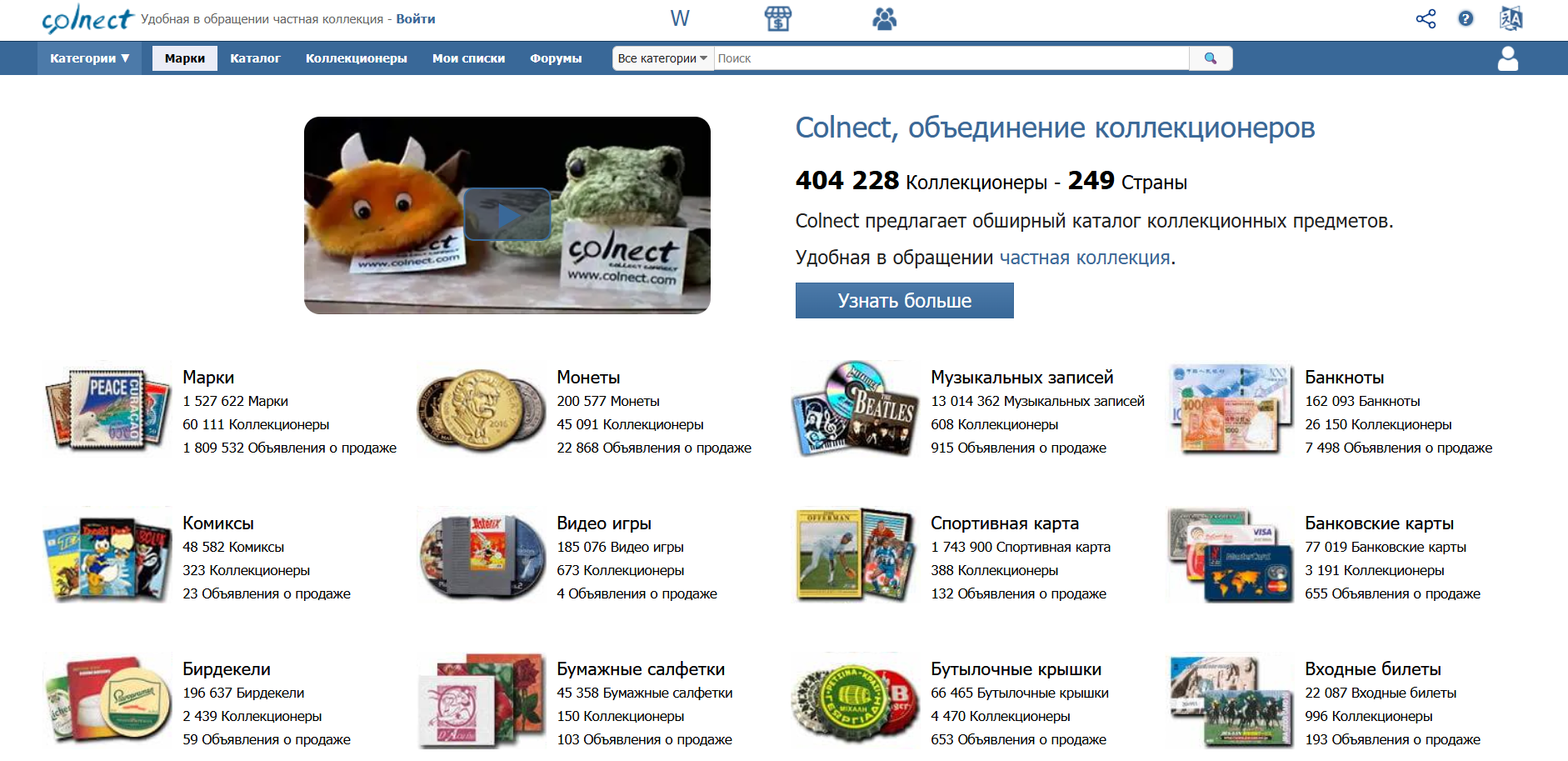
Мы можем столкнуться с проблемами ручного учета. Будут возникать ошибки в данных: ручной ввод характеристик марок (например, зубцовки или года выпуска) приводит к неточностям из-за ошибок человека который будет к примеру переписывать данные из одной коллекции в данные коллекционераТак же не эффективный поиск является существенным минусом, а именно отсутствие фильтров усложняет подбор марок по критериям (например, «марки СССР 1960–1970 гг.») будет занимать много времени.Отсутствие аналитики может привести к невозможности быстро определить владельца самой дорогой марки или коллекционера с максимальным числом редких экземпляров.Очень важным является риск потери данных: хранение информации в бумажных каталогах или Excel-файлах ненадежно.

**Решение** — система, которая будет автоматизировать учет марок и коллекций ,предоставлять инструменты для формирования отчетов (например, группировка редких марок по владельцам). Так же будет обеспечивать безопасное хранение данных в SQL Server.

**1.2 Анализ аналогов разрабатываемого решения**

На рынке уже существуют специализированные сервисы для коллекционеров:

**Colnect:** каталог марок с возможностью обмена, но без функций учета личных коллекций.

****

**StampWorld** позволяет просматривать марки, но не поддерживает сложные запросы (например, группировку по стоимости).



А если мы смотрим на локальные решения то это **Excel и Access.**



Хорошие инструменты, но требуют ручной работы и не обеспечивают многопользовательский доступ (Хотя можно использовать API Гугл таблиц).

Преимущества разрабатываемой системы заключаются в аналитике, то есть запросы для выявления редких марок, топовых коллекционеров и т.д. Интеграция с ASP.NET Core это само по себе масштабируемо, безопасно и кроссплатформенно.

**1.3 Требования к разрабатываемой системе**

**Функциональные требования**

В связи с поставленной задачей, я выделяю три основных функциональных требования к разрабатываемой системе. Они представлены по пунктам в таблице ниже (см. Таблица 1). Представленные требования являются основополагающим направлением в разработке этого WEB-приложения для учета и сортировки марок. Представленные требования уже на этом этапе определяют бизнес-логику взаимодействия элементов системы проекта. По существу – это минимальные требования к приложению для корректной работы, самого смысла и нужды в этом.

|  |  |  |
| --- | --- | --- |
| **Таблица 1** | | |
| *Управление каталогом марок* | *Управление пользовательскими коллекциями:* | *Аналитические возможности и отчетность:* |
| 1. Система должна обеспечивать ведение базы данных почтовых марок с возможностью добавления, редактирования и удаления записей  2. Для каждой марки необходимо хранить набор атрибутов: страну происхождения, номинальную стоимость, год выпуска, тираж, тематическую принадлежность  3. Требуется реализовать механизм пометки редких экземпляров на основе установленного ценового критерия (например, свыше 1000 рублей) | 1. Необходимо предусмотреть возможность создания персональных коллекций для зарегистрированных пользователей  2. Система должна позволять добавлять и удалять марки из коллекций, а также просматривать состав существующих коллекций  3. Требуется реализовать функционал подсчета базовой статистики по коллекциям | 1. Система должна обеспечивать формирование различных отчетов на основе имеющихся данных  2. Необходимо предусмотреть возможность ранжирования коллекционеров по заданным критериям  3. Требуется реализовать механизмы поиска и фильтрации данных по различным параметрам |

Также, нужно понимать, что работа приложения с базой данных – это сложная система с набором логик, которая будет работать с большим количеством данных, которое со временем будут повышаться. Именно поэтому я также выделяю дополнительные требования - производительность и интерфейс

Приложение должно обеспечить стабильную работу при увеличении объема данных. Это зависит как от сервера на котором будет находиться сам проект , так и самое главное, от того как оптимизирован код.

Интерфейс системы должен быть интуитивно понятным и удобным для пользователей. Если вывод будет не структурирован логически, то будет критически сложно разобраться что есть что. Так же требуется обеспечить адаптивность интерфейса для различных типов устройств, по той причине что приложение должно поддерживаться не только персональным компьютером, но и с мобильными устройствами.

**Перспективы развития**

1. Расширение аналитических возможностей системы

2. Добавление функционала для оценки стоимости марок

3. Интеграция с платежными системами для реализации механизмов обмена между коллекционерами

**2. Технологическая часть**

**2.1 Технологический стек**

Учитывая все особенности из пункта 1.3, включая ”Таблица 1” – я выбираю технологический стек представленный ниже.

*Серверная часть выполнена на* **ASP.NET Core (MVC)**. Благодаря высокой производительности платформа позволяет писать код, который будет быстро и корректно выполнять бизнес-логику приложения. А сам язык программирования C# является объектно-ориентированным языком – само использование является оптимизацией процессов. Объектно-ориентированность позволяет без каких-либо проблем, расширять функционал.

*Клиентская часть:*

**Razor Pages** – стандартный инструмент для View в Visual Studio. Основными плюсами которого в данной работе является полная поддержка языка C#. Так же хотелось бы отметить возможности разбиения на частичные представления. Эти перечисленные возможности, делают Razor Pages идеальным решением для выбора разработки фронтенда.

**JavaScript** – является неотъемлемой частью любого хорошего проекта, в основном используют для стилей, анимаций и выполнения элементарной логики

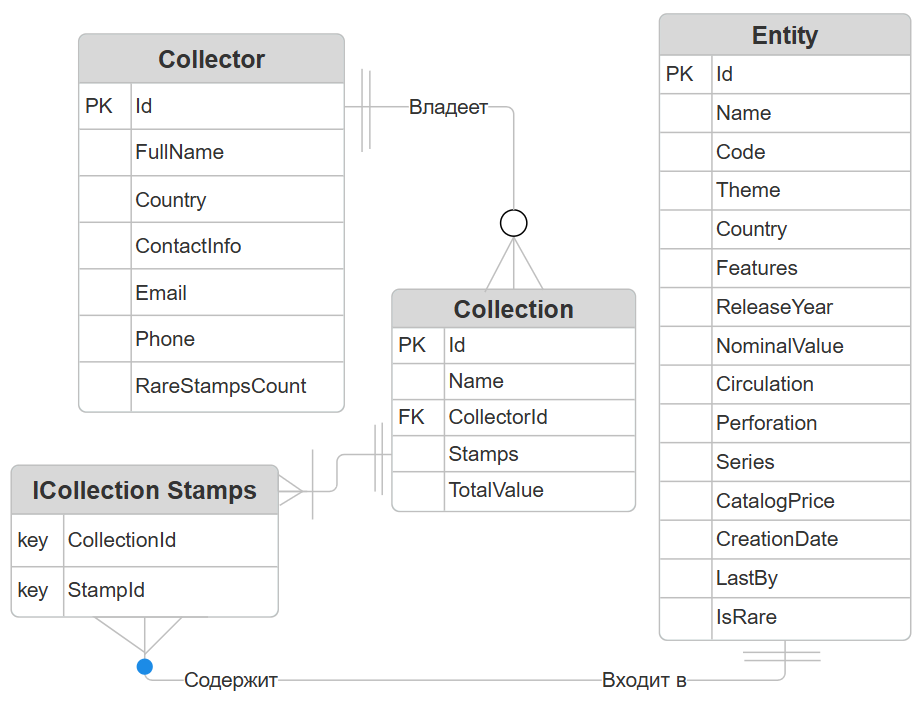
*Хранение данных…………………., ……..:*

**СУБД SQL Server** – главная на рынке система управления баз данных. Официальные библиотеки Microsoft гарантируют корректную работу с данными, а также обеспечиваю безопасность их хранения.

**Основные сущности:** Марки, Коллекции, Пользователи

**2.2 Разработка решения**

**ERD-диаграмма** описывает структуру сущностей и их связи в базе данных (рис.1).

**Рисунок 1**

**Диаграмма последовательности работы** приложения с учетом разделения ролей(рис.2) раскрывает нам логику и структуру программы.

*Как у базового пользователя*, есть возможности просмотра представлений с сущностями из БД. А администратор предопределен программно, так как в данной реализации программы, существование профилей пользователей не предусматривается, у них не будет никакой функциональности, так как все CUD операции выполняет администратор.

*У администратора* спектр операций полный. При входе в программу, далее в контроллеры которые требуют авторизованного пользователя роли Admin, программа проверит наличие и корректность токена. Если все верно, программы нас пропускает далее. Если нет токена или он не совпал, то нас перебросит на страницу входа администратора.

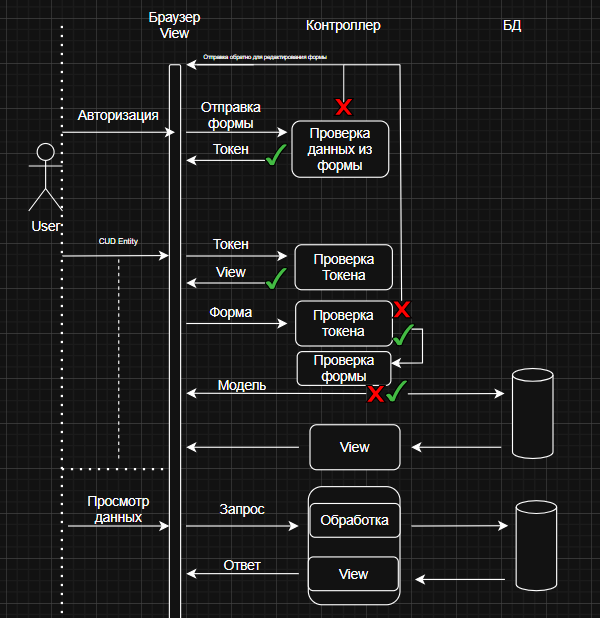
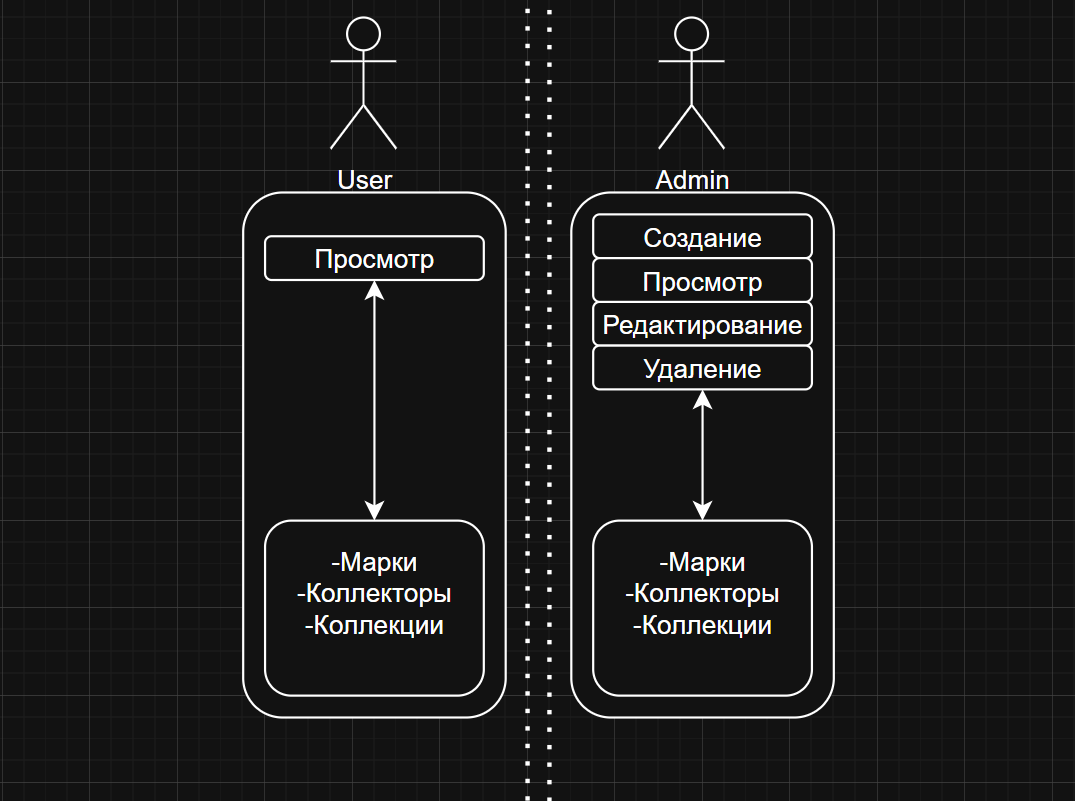
**Рисунок 2**

Диаграмма функциональных возможностей.



**3 Практическая часть**