Введение

CollectiblesVault - Это десктопное приложение, предназначенное для коллекционеров, которым важно эффективно управлять своей коллекцией предметов.

Актуальность проекта обусловлена тем, что коллекционирование требует системного подхода к учету экспонатов, их классификации и сохранению данных. Традиционные способы, такие как ведение записей вручную или использование стандартных таблиц, часто оказываются неудобными и не соответствуют потребностям пользователей. CollectiblesVault решает эти проблемы, предлагая удобное решение для хранения, поиска и анализа коллекции.

Цель разработки — создание удобного и интуитивно понятного приложения, позволяющего коллекционерам регистрироваться в системе, добавлять и редактировать записи о предметах коллекции, загружать изображения, управлять категориями и формировать отчёты.

Целевая аудитория включает как начинающих коллекционеров, так и опытных пользователей, которым важно иметь доступ к инструменту для ведения систематизированного учета своих предметов.

Основные функции приложения включают регистрацию и авторизацию пользователей, ввод данных о коллекционных предметах, настройку категорий, создание отчетов по периодам и категориям, а также экспорт данных в различных форматах.

Структура отчета включает несколько ключевых разделов, которые подробно раскрывают все этапы разработки. В разделе, посвященном методам разработки, рассматриваются выбор технологий, процесс проектирования интерфейса и реализация основных функций. Далее в отчете приводится анализ полученных результатов, включающий описание функциональных возможностей приложения, удобства его использования и оценки производительности. В заключительном разделе отчета проводится итоговая оценка эффективности приложения, рассматриваются возможные направления для его дальнейшего развития, включая возможные улучшения интерфейса, расширение функциональности и внедрение дополнительных технологий для повышения удобства работы пользователей.

1. Проектирование ПП

Проектирование программного продукта — это процесс создания архитектуры, структуры и детальных спецификаций будущей программы перед её непосредственной разработкой. Проектирование позволяет обеспечить качество, масштабируемость и поддержку программного продукта в будущем.

* 1. Диаграмма связей

Диаграмма связей (рисунок 1) является удобным инструментом визуализации, который отображает взаимосвязи между компонентами системы. С её помощью можно проследить логику взаимодействия отдельных элементов приложения, выявить ключевые зависимости и понять общие принципы функционирования системы.

Работа с приложением начинается с главного окна, где пользователь проходит регистрацию или авторизацию (если у него уже есть аккаунт). После успешного входа открывается главное меню, выступающее в роли центрального узла навигации. Отсюда пользователь получает доступ ко всем основным функциям программы.

Подобная структура делает систему интуитивно понятной: пользователь последовательно переходит от общего интерфейса к конкретным возможностям приложения. Диаграмма чётко отражает эту логику, демонстрируя взаимосвязь между аутентификацией, главным меню и остальными модулями системы.

A diagram of a diagram

AI-generated content may be incorrect.

Рисунок 1 – диаграмма структуры приложения

1.2 Диаграмма сценариев

Диаграмма сценариев, представленная на рисунке 2, является графическим отображением взаимодействия между пользователями и системой, демонстрирующим последовательность действий для выполнения различных задач.

На схеме отражены две основные категории пользователей. Гости, не прошедшие авторизацию, имеют ограниченный функционал и могут только приступить к процедуре регистрации. После успешного создания учетной записи пользователь получает доступ ко всем возможностям приложения.

Для авторизованных пользователей предусмотрен расширенный функционал. Через раздел «Ввод данных» осуществляется добавление новой информации в систему. Функция «Просмотр данных» позволяет работать с существующими записями. Дополнительно доступен раздел «Настройки», где можно персонализировать параметры системы в соответствии с индивидуальными предпочтениями.

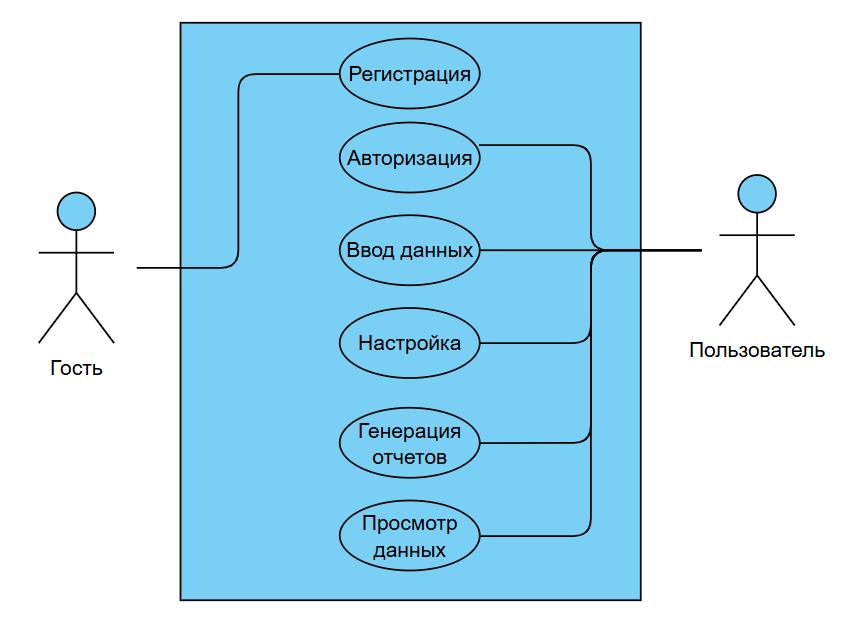


Рисунок 2 – Диаграмма сценариев приложения

1.3 Проектирование базы данных

Каждый пользователь получает возможность добавлять предметы в свою коллекцию, назначать им цену и дату добавления. Эти предметы должны быть классифицированы, поэтому предусмотрена система категорий, которая помогает структурировать коллекцию. Таблица Users связана с Items, поскольку каждый пользователь может добавлять несколько предметов, но один предмет принадлежит только одному пользователю. Таблица Categories связывается с Items, позволяя каждому предмету принадлежать к одной категории, обеспечивая четкую классификацию коллекции. Дополнительно включена таблица WishList, позволяющая пользователю сохранять информацию о желаемых предметах.

На рисунке 3 представлена ER-диаграмма, со всеми необходимыми таблицами для реализации функционала проекта.

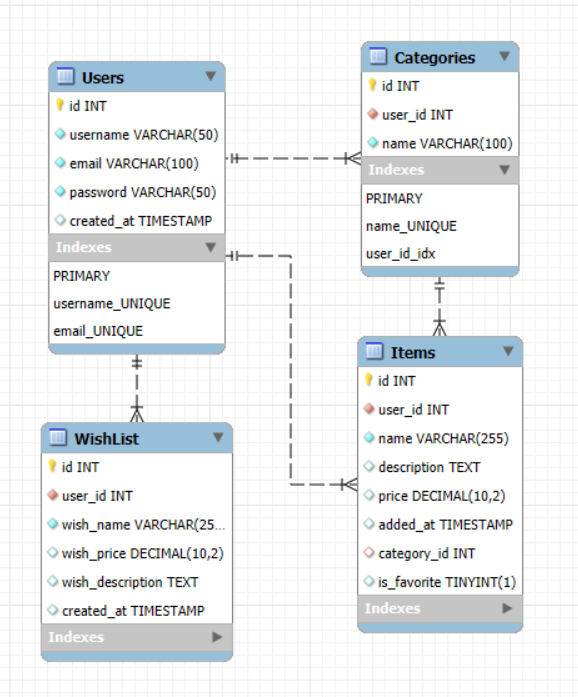


Рисунок 3 – ER-диаграмма приложения

2.1 Описание средств разработки

MySQL Workbench был выбран в качестве основного инструмента для работы с базой данных благодаря своей комплексной функциональности. Ключевым фактором стало наличие визуальных средств проектирования ER-диаграмм, что существенно упрощает процесс моделирования структуры данных. Интегрированный SQL-редактор с автодополнением сокращает время написания запросов, а инструменты администрирования позволяют эффективно управлять сервером без необходимости использования дополнительного ПО. Возможность миграции данных между различными СУБД обеспечивает гибкость при изменении инфраструктуры.

MySQL был выбран в качестве СУБД по нескольким объективным причинам. Система предоставляет полную поддержку SQL-стандартов с дополнительными оптимизациями для сложных запросов. Встроенные механизмы репликации позволяют масштабировать систему при увеличении нагрузки. Широкая распространенность MySQL обеспечивает доступность квалифицированных специалистов для поддержки и развития системы. Выбор MySQL был обусловлен оптимальным сочетанием характеристик для требований проекта, где важны надежность хранения данных, предсказуемая производительность и возможность масштабирования по мере роста нагрузки.

Совместное использование MySQL и MySQL Workbench создает замкнутую экосистему разработки, где каждый этап — от проектирования схемы до эксплуатации рабочей системы — покрывается специализированными инструментами. Такая интеграция минимизирует риски несовместимости и обеспечивает согласованность между различными стадиями жизненного цикла базы данных. Выбор именно этой технологической связки был обусловлен требованиями проекта к надежности хранения данных, производительности выполнения запросов и необходимости комплексного инструментария для администрирования. MySQL предоставляет оптимальный баланс между функциональностью и ресурсоемкостью, а MySQL Workbench значительно упрощает процессы разработки и сопровождения базы данных.

Приложение Б

В основе кода для базы данных приложения лежит структура, обеспечивающая удобное добавление, систематизацию и отслеживание предметов. Каждый пользователь, зарегистрировавшись в системе, получает возможность создавать коллекцию, добавлять предметы с описанием, ценой и датой добавления. Для лучшей организации предусмотрена система категорий, позволяющая группировать предметы по различным параметрам. Все таблицы связаны между собой посредством внешних ключей, обеспечивающих целостность данных.

Листинг 1 – Код базы данных MySQL

CREATE TABLE IF NOT EXISTS Users (

    id INT PRIMARY KEY AUTO\_INCREMENT,

    username VARCHAR(50) UNIQUE NOT NULL,

    email VARCHAR(100) UNIQUE NOT NULL,

    password\_hash VARCHAR(255) NOT NULL,

    created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE IF NOT EXISTS Categories (

    id INT PRIMARY KEY AUTO\_INCREMENT,

    user\_id INT NOT NULL,

    name VARCHAR(100) UNIQUE NOT NULL,

    FOREIGN KEY (user\_id) REFERENCES Users(id)

);

CREATE TABLE IF NOT EXISTS Items (

    id INT PRIMARY KEY AUTO\_INCREMENT,

    user\_id INT NOT NULL,

    name VARCHAR(255) NOT NULL,

    description TEXT,

    price DECIMAL(10,2),

    added\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

    category\_id INT,

    is\_favorite BOOLEAN DEFAULT FALSE,

    FOREIGN KEY (user\_id) REFERENCES Users(id),

    FOREIGN KEY (category\_id) REFERENCES Categories(id)

);

CREATE TABLE IF NOT EXISTS WishList (

    id INT PRIMARY KEY AUTO\_INCREMENT,

    user\_id INT NOT NULL,

    wish\_name VARCHAR(255) NOT NULL,

    wish\_price DECIMAL(10,2),

    wish\_description TEXT,

    created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

    FOREIGN KEY (user\_id) REFERENCES Users(id)

);