

Содержание

Реферат.....	2
Условные обозначения и сокращения.....	3
Введение.....	5
Глава 1. Анализ предметной области и постановка задачи.....	6
1.1 Общая характеристика онлайн-торгов.....	6
1.2 Особенности и проблемы проведения торгов по монетам.....	8
1.3 Требования к системе.....	10
1.4 Постановка задачи.....	12
Глава 2. Проектирование информационной системы.....	14
2.1 Архитектура системы.....	14
2.2 Модель данных.....	16
2.3 Диаграмма компонентов.....	18
2.4 Бизнес-логика и пользовательские сценарии.....	21
2.5 Безопасность и авторизация.....	23
1. Хранение учетных данных.....	23
2. Защищённый обмен данными.....	23
3. Аутентификация и управление сессией.....	23
4. Ограничение прав доступа (авторизация).....	24
5. Защита от атак.....	24
Глава 3. Реализация системы.....	25
3.1 Используемые технологии.....	25
3.2 Основные модули системы.....	27
3.3 Примеры REST-запросов.....	30
3.4 Скриншоты пользовательского интерфейса.....	32
3.5 Тестирование и отладка.....	40
Глава 4. Экономическое обоснование.....	42
4.1 Расчёт затрат на разработку.....	42
4.2 Оценка стоимости системы.....	43
4.3 Сравнение с альтернативами.....	44
4.4 Риски и способы их минимизации.....	45
Глава 5. Защита данных и безопасность.....	47
5.1 Актуальные угрозы.....	47
5.2 Средства защиты.....	48
5.3 Регламент хранения и обработки данных.....	49
Заключение.....	51
Список использованных источников.....	53
Приложение А.....	54

Реферат

Бакалаврская работа состоит из 43 страниц, содержащих 10 источников, 12 рисунков, и 12 таблиц. Объект исследования: Разработка информационной системы торгов. Дипломная работа выполнена в текстовом редакторе LibreOffice Writer, представлена в программе Adobe Acrobat в формате PDF.

В настоящей работе рассматривается вопрос разработки специализированной информационной системы для онлайн-торговли коллекционными монетами. Описываются актуальность поставленной проблемы и существующие недостатки традиционных и интернет-платформ в сфере нумизматики. Целью исследования является создание веб-приложения, обеспечивающего безопасное, функциональное и удобное размещение и торговлю коллекционными монетами.

Для достижения указанной цели выполнен анализ потребностей пользователей и сравнительный обзор существующих информационных систем (таких как eBay, Catawiki и др.), предоставляющих функциональность аукционов и торговых площадок для коллекционеров. На основании полученных данных сформулирован набор ключевых требований к разрабатываемой системе и определены основные механизмы её работы.

В технической части работы проанализированы критерии выбора инструментов и технологий (производительность, совместимость, безопасность, масштабируемость) и обоснован выбор конкретных решений для реализации серверной и клиентской частей, а также базы данных. Разработан прототип веб-приложения, который демонстрирует возможности указанной концепции и соответствует сформулированной цели исследования. Результаты работы могут быть использованы при дальнейшем создании полнофункциональных платформ для торговли редкими монетами в сети Интернет.

Условные обозначения и сокращения

1. **ИС (Информационная система)** — совокупность программных и аппаратных средств, обеспечивающих сбор, хранение, обработку и передачу информации.
2. **БД (База данных)** — организованная структура для хранения данных. В проекте используется документно-ориентированная база MongoDB.
3. **API (Application Programming Interface)** — интерфейс взаимодействия между различными компонентами системы, чаще всего реализованный через HTTP-запросы.
4. **REST (Representational State Transfer)** — архитектурный стиль, определяющий принципы построения API.
5. **HTTP (Hypertext Transfer Protocol)** — протокол передачи гипертекста, используемый для взаимодействия клиента и сервера.
6. **HTTPS** — защищённая версия HTTP с использованием TLS/SSL для шифрования данных при передаче.
7. **CRUD (Create, Read, Update, Delete)** — базовые операции при работе с данными в системе: создание, чтение, обновление и удаление.
8. **JWT (JSON Web Token)** — формат токена для аутентификации и авторизации пользователей в API.
9. **SMTP (Simple Mail Transfer Protocol)** — протокол, используемый для отправки электронной почты.
10. **TLS (Transport Layer Security)** — криптографический протокол, обеспечивающий защищённую передачу данных.
11. **XSS (Cross-Site Scripting)** — тип атаки, связанный с внедрением вредоносных скриптов на веб-страницы.
12. **CSRF (Cross-Site Request Forgery)** — атака, при которой злоумышленник заставляет пользователя выполнить нежелательное действие от имени авторизованной сессии.
13. **IP (Internet Protocol)** — адрес, используемый для идентификации устройств в сети.
14. **UI (User Interface)** — пользовательский интерфейс, то, с чем взаимодействует пользователь при работе с системой.

15. **UX (User Experience)** — пользовательский опыт, связанный с восприятием и удобством использования интерфейса.
16. **OAuth2** — современный протокол авторизации, позволяющий пользователю предоставлять доступ к своим данным стороннему сервису без передачи пароля.
17. **FastAPI** — высокопроизводительный Python-фреймворк для создания REST API.
18. **Frontend** — клиентская часть приложения, отображаемая в браузере пользователя.
19. **Backend** — серверная часть, отвечающая за бизнес-логику и обработку запросов от клиента.
20. **VPS (Virtual Private Server)** — виртуальный частный сервер, на котором размещается серверная часть проекта.
21. **Docker** — технология контейнеризации, позволяющая упаковать приложение и его зависимости в изолированную среду.
22. **ORM (Object-Relational Mapping)** — метод работы с базой данных, при котором объекты программы сопоставляются с документами или таблицами. В MongoDB используется схожая логика через ODM (Object-Document Mapping).
23. **Swagger UI** — графический интерфейс для просмотра и тестирования REST API, автоматически создаваемый FastAPI.
24. **JSON (JavaScript Object Notation)** — формат структурированных данных, используемый в запросах и ответах API.
25. **Base64** — способ кодирования бинарных данных (например, изображений) в текстовый вид.
26. **Лот** — объект аукциона, представляющий собой монету или набор монет, выставленных на торги.
27. **Ставка** — денежное предложение, которое пользователь делает на лот в процессе аукциона.

Введение

Актуальность темы

Развитие онлайн-аукционов в последние годы стало глобальным трендом, однако в России подобные площадки, специализирующиеся на нумизматике, остаются малоразвитыми. Большинство торгов монетами происходит через форумы, соцсети или локальные клубы, что не обеспечивает прозрачности, безопасности и удобства для участников. Создание специализированной информационной системы для онлайн-торгов позволит структурировать этот рынок, предоставив коллекционерам и инвесторам надежный инструмент для покупки и продажи монет.

Кроме того, такая система может стать важным ценовым ориентиром для розничных продавцов. Поскольку аукционные цены отражают реальный спрос, магазины и частные продавцы смогут использовать данные о завершенных торгах для формирования справедливой стоимости монет. Это особенно актуально для редких экземпляров, где цена часто определяется аукционной конкуренцией. Также система даст нумизматическому сообществу возможность приобретать монеты до их попадания в массовую розницу, что особенно ценно для коллекционеров, ищущих раритеты.

Глава 1. Анализ предметной области и постановка задачи

1.1 Общая характеристика онлайн-торгов

Онлайн-торги представляют собой процесс купли-продажи товаров или услуг в формате аукциона с использованием интернет-технологий. Данный формат становится все более популярным благодаря высокой доступности, прозрачности и возможности привлечения широкой аудитории участников вне зависимости от их географического положения.

Современные онлайн-аукционы подразделяются на несколько видов:

- **английский аукцион:** классическая форма, при которой участники поочередно повышают ставки до тех пор, пока не останется один победитель;
- **голландский аукцион:** цена начинается с высокой и постепенно снижается до тех пор, пока кто-то не согласится на покупку;
- **обратный аукцион:** используется чаще в сфере услуг, когда заказчики публикуют запрос, а исполнители соревнуются, снижая стоимость выполнения;
- **закрытый аукцион:** участники делают ставки вслепую, не видя предложений конкурентов.

Обзор существующих информационных систем

В настоящее время существуют различные информационные системы и платформы, которые можно рассматривать как аналоги или прототипы для решения задачи онлайн-торговли нумизматикой:

- **eBay:** крупнейшая в мире интернет-торговая площадка, предоставляющая продавцам и покупателям простой и интуитивный сервис для проведения аукционов и сделок по фиксированной цене. Платформа поддерживает аукционный формат и продажу по фиксированной цене, позволяя продавцам выставлять лоты на торги или устанавливать «Buy It Now» цену. eBay объединяет международные рынки (сайт доступен на множестве языков) и выступает в роли посредника при заключении сделок: компания не участвует напрямую в передаче това-

ра от продавца покупателю, а лишь обеспечивает инфраструктуру для размещения объявлений и обработки транзакций;

- **Catawiki:** европейская онлайн-аукционная платформа, специализирующаяся на редких и уникальных предметах коллекционирования. Она позиционирует себя как одна из самых посещаемых в Европе площадок с курируемыми (отобранными экспертами) еженедельными аукционами «особых» товаров wine-searcher.com. На Catawiki имеется категория «Монеты и марки», где специалисты вручную отбирают редкие и ценные лоты для включения в аукцион. Благодаря экспертному отбору и строгому проверочному процессу Catawiki обеспечивает повышенное качество и аутентичность представленных объектов;

- **AuctionSoftware:** коммерческий SaaS-провайдер решений для создания аукционных и торговых сайтов. По данным разработчиков платформы, на базе AuctionSoftware создано более 500 онлайн-аукционов, на которых за четыре года обработано свыше 1 миллиарда транзакций. Платформа предоставляет широкий набор инструментов для реализации различных форматов торгов и позволяет быстро запустить специализированный торговый сайт. Использование подобных сервисов сокращает время и затраты на разработку проекта, но требует оплаты лицензий и адаптации под конкретные требования;

- **другие платформы:** Существуют и другие примеры, которые можно адаптировать под нумизматическую тематику. Так, на российском рынке действуют специализированные аукционы (например, Auction.ru, [Conros](http://Conros.ru) и др.), где предусматривается экспертная оценка монет. Также многие международные маркетплейсы ([Amazon](http://Amazon.com), [AliExpress](http://AliExpress.com)) имеют разделы для коллекционных товаров, но не предлагают узкоспециализированных функций для коллекционеров нумизматики. Анализ этих систем помогает выявить лучшие практики и типичные ограничения, что важно при формулировании требований к новой системе.

Онлайн-аукционы обладают рядом преимуществ:

- упрощение логистики и документооборота;
- автоматизация всех этапов торгов;

- возможность точной фиксации ставок и времени;
- обширный выбор лотов для покупателей и широкая аудитория для продавцов.

Тем не менее, у данного подхода существуют и определённые недостатки:

- вероятность мошенничества при отсутствии верификации участников;
- проблемы с подлинностью товаров;
- зависимость от технической стабильности платформы;
- необходимость обеспечения кибербезопасности.

В сфере нумизматики аукционы особенно востребованы, так как они позволяют формировать справедливую рыночную цену за счёт конкурентных ставок. Однако при отсутствии специализированной платформы торги часто происходят стихийно — на форумах, в социальных сетях или через личные сообщения, что снижает уровень доверия к сделкам и затрудняет привлечение новых участников.

Разработка специализированной онлайн-системы торгов, учитывающей особенности нумизматического рынка, становится актуальной задачей для цифровизации данной ниши.

1.2 Особенности и проблемы проведения торгов по монетам

Торги нумизматическими объектами — это узкоспециализированный сегмент аукционного рынка, который имеет ряд уникальных характеристик, отличающих его от других видов онлайн-торгов. Монеты являются не только предметом коллекционирования, но и потенциальным объектом инвестиций, что требует от участников рынка высокой точности, прозрачности и доверия.

Особенности торгов по монетам:

1. **Уникальность лотов:** каждая монета обладает определённой степенью уникальности, зависящей от года выпуска, номинала, состояния (грейда), наличия браков, редкости и исторической ценности. Это делает ценообразование особенно чувствительным к качеству описания и визуальному представлению.

2. **Необходимость экспертной оценки:** в отличие от массовых товаров, определить стоимость и подлинность монеты может только специалист. Поэтому доверие к платформе сильно зависит от возможности привлечения экспертов или предоставления сертификатов.

3. **Высокая вероятность подделок:** на рынке нумизматики распространены фальшивки, реплики и копии, которые могут быть визуально неотличимы от оригиналов без глубокого анализа. Это создаёт дополнительные риски для покупателей.

4. **Часто — ограниченный круг участников:** торги монетами, особенно редкими, проводятся в узком кругу коллекционеров и заинтересованных лиц. Такие торги требуют тонкой настройки интерфейса и коммуникаций для профессиональной аудитории.

5. **Ценность истории объекта:** происхождение монеты, её владельцы, участие в предыдущих аукционах могут значительно влиять на её стоимость.

Проблемы, возникающие при существующих подходах:

1. **Отсутствие централизованной платформы:** большинство торгов в России происходит через форумы, Telegram-чаты, социальные сети, что не позволяет обеспечить контроль, безопасность и единые стандарты.

2. **Недостаточная прозрачность:** часто ставки размещаются вручную, без фиксации времени, что создаёт возможность манипуляций.

3. **Отсутствие автоматизации:** закрытие торгов, уведомления участникам, определение победителя — всё это зачастую делается вручную, что снижает эффективность и вызывает споры.

4. **Нет гарантий безопасности сделки:** в отсутствие механизмов верификации и защиты интересов сторон, сделки могут срываться, а участники — сталкиваться с мошенничеством.

5. **Слабая правовая защищенность:** так как большинство торгов проводится вне регулируемых площадок, отсутствует юридическая база для решения споров.

Таким образом, несмотря на высокий интерес к нумизматическим торгам, текущие методы их проведения не соответствуют современным требованиям безопасности, автоматизации и пользовательского удобства. Эти проблемы могут быть решены за счёт внедрения специализированной информационной системы, учитывающей особенности предметной области

1.3 Требования к системе

Для эффективного функционирования информационной системы онлайн-торгов нумизматическими объектами необходимо определить функциональные и нефункциональные требования. Они формируют основу для проектирования архитектуры, интерфейсов и бизнес-логики системы.

Функциональные требования:

1. Регистрация и авторизация пользователей:

- возможность регистрации с подтверждением электронной почты;
- вход в систему с использованием логина и пароля;
- хранение паролей в зашифрованном виде;
- восстановление доступа.

2. Работа с лотами:

- создание лота с возможностью добавления изображений, описания, стартовой цены и срока завершения торгов;
- просмотр всех активных лотов;
- поиск и фильтрация по параметрам (категория, цена, дата окончания);
- редактирование и удаление лота до начала торгов (только владельцем или администратором).

3. Система ставок:

- возможность делать ставки зарегистрированным пользователям;
- автоматическое повышение текущей цены и сохранение истории ставок;
- уведомление пользователей при перебитии ставки и завершении аукциона;

- определение победителя по истечении времени торгов.

4. Личный кабинет:

- просмотр истории ставок и побед;
- управление собственными лотами;
- настройки профиля.

5. Административная панель:

- управление пользователями (блокировка, удаление);
- модерация лотов (в том числе подозрительных или фальшивых);
- управление категориями монет.

6. История торгов и аналитика:

- доступ к завершённым торгам и их результатам;
- возможность отслеживания рыночной динамики цен на конкретные монеты;
- статистика по активности пользователей и категориям лотов.

Нефункциональные требования

7. Безопасность:

- использование защищённого протокола HTTPS;
- токены для авторизации;
- защита от SQL-инъекций и XSS-атак;
- ограничение частоты запросов (rate limiting).

8. Надёжность и устойчивость:

- обработка ошибок на серверной и клиентской части;
- сохранение данных в случае сбоя;
- резервное копирование БД.

9. Масштабируемость:

- возможность дальнейшего увеличения нагрузки;
- возможность интеграции с платёжными системами и сервисами оценки.

10. Удобство использования:

- удобный и интуитивно понятный интерфейс;
- адаптация под мобильные устройства;
- быстрая навигация по категориям и лотам.

11. Локализация:

- русский язык по умолчанию;
- возможность дальнейшей реализации мультиязычного интерфейса.

Таким образом, требования к системе охватывают все основные аспекты её функционирования — от пользовательского взаимодействия до обеспечения безопасности и устойчивости работы. На основе этих требований в следующей главе будет разработана архитектура будущей платформы.

1.4 Постановка задачи

Постановка задачи разработки системы онлайн-торговли нумизматикой включает определение набора функциональных требований и входных/выходных данных.

Требуется реализовать веб-приложение, содержащее следующие основные компоненты:

- **управление пользователями:** регистрация и авторизация участников, хранение их профилей и контактных данных;
- **управление лотами:** добавление и редактирование информации о монетах (название, описание, характеристики, изображения), установка начальной цены или способа проведения торгов;
- **проведение торгов:** организация аукционной продажи, приём ставок пользователей, автоматическое определение победителя и итоговой цены по завершении аукциона;
- **коммуникация с участниками:** возможность получить контактные данные участника для отправки сообщений от продавца покупателю, уведомления о ходе торгов.

Входными данными для системы являются: сведения о выставляемых монетах (описание, фотографии, стартовая цена, параметры аукциона: длительность, шаг ставки), а также информация о пользователях. Выходными данными системы служат: актуальные списки лотов с информацией о текущих ставках, результаты завершённых торгов (победители аукционов, конечная цена сделки), а также отчёты о совершённых продажах и историю ставок. Таким образом, строгая постановка задачи заключается в создании информационной системы, которая по полученным на вход данным организует процесс онлайн-аукциона монет и представляет результаты пользователям в понятном виде. Система должна обеспечивать надёжность и бесперебойность, а также удобный и интуитивно понятный интерфейс для участников торгов.

Глава 2. Проектирование информационной системы

2.1 Архитектура системы

Информационная система онлайн-торгов монетами реализована по принципу клиент-серверной архитектуры с чётким разделением backend- и frontend-частей. Это обеспечивает гибкость, масштабируемость и возможность дальнейшего развития системы. Для обеспечения более высокой степени информационной безопасности панель администратора вынесена в отдельный сервис, который никак не связан с основным кодом. В таком случае, злоумышленнику потребуется как минимум знать, где расположен адрес админ панели, но это можно предотвратить, если настроить сервер с обработкой админ запросов обрабатывать данные только от знакомых ip адресов.

Более подробно о реализации системы написано в Главе 3.

Общие принципы архитектуры:

- **backend:** реализован на языке программирования Python с использованием современного фреймворка FastAPI, который обеспечивает высокую производительность (до 10000 запросов в секунду) и поддержку асинхронных запросов;
- **frontend:** представляет собой веб-клиент, написанный с использованием HTML/CSS/JavaScript;
- **база данных:** MongoDB используется в качестве СУБД для хранения информации о пользователях, лотах, ставках и результатах торгов;
- **хранение изображений:** изображения лотов загружаются и сохраняются в базе данных в виде закодированной строки в формате b64;
- **аутентификация и безопасность:** применяется Bearer токен для авторизации пользователей, с поддержкой ролей (пользователь, администратор).

Основные компоненты системы:

1. Клиентская часть (Frontend):

- форма регистрации и входа;
- интерфейс просмотра лотов;

- страница лота с возможностью сделать ставку;
- личный кабинет пользователя;
- административная панель (доступна только админам);
- страница с информацией о компании.

2. Серверная часть (Backend):

- авторизация/аутентификация;
- управление пользователями;
- работа с лотами;
- обработка ставок;
- логика завершения торгов и определения победителя;
- middleware для обработки ошибок и логирования;
- планировщик, для автоматического завершения торгов в нужное вре-

мя.

3. База данных:

- таблицы: users, admins, auctions, tokens;
- индексы на ключевые поля для повышения производительности.

4. Механизм уведомлений:

- отправка email-уведомлений при перебитии ставки или завершении аукциона.

Взаимодействие компонентов (в виде последовательности):

- пользователь отправляет HTTP-запрос (например, POST /login);
- FastAPI обрабатывает запрос, проверяет данные, возвращает Bearer-токен;
- клиент использует токен для последующих действий (например, POST /bid);
- сервер проверяет валидность токена, обновляет данные в БД;
- в случае события (завершение торгов) система уведомляет участников и записывает результат.

Преимущества выбранной архитектуры:

- высокая производительность благодаря FastAPI и асинхронной обработке;
- гибкость расширения функционала (возможность добавления платёжных модулей, модерации, чата и пр.);
- удобство сопровождения и отладки за счёт модульной структуры;
- совместимость с различными клиентскими приложениями (в том числе мобильными).

2.2 Модель данных

В системе используется MongoDB — документно-ориентированная нереляционная база данных. Это позволяет гибко хранить и масштабировать данные без строгой схемы, что особенно удобно в системах с активным пользовательским взаимодействием, как в случае онлайн-аукционов.

В системе определены следующие основные коллекции:

1. clients — Пользователи:

Содержит данные зарегистрированных клиентов, участвующих в торгах.

Таблица 1 — Пользователи

Поле	Тип	Описание
_id	ObjectId	Уникальный ID MongoDB
id	int	Внутренний числовой ID
nickname	string	Отображаемое имя
email	string	Адрес электронной почты
phone_number	string	Телефон
password	string (SHA256)	Захешированный пароль
avito_url	string	Ссылка на профиль/портфолио
email_verified	boolean	Подтверждение email
get_mails, mail_receive_	boolean	Настройки уведомлений

Связь: один пользователь может участвовать в неограниченном числе торгов.

2. auctions — Аукционы / Лоты:

Представляет монету или набор монет, выставленных на торги.

Таблица 2 — Аукционы

Поле	Тип	Описание
_id	ObjectId	Уникальный ID
a_id	int	Числовой ID аукциона
short_name	string	Название лота
description	string	Подробности
start_price	float (string)	Стартовая цена
min_bid_step	float (string)	Минимальный шаг ставки
start_datetime	datetime	Время начала
end_datetime	datetime	Время окончания
bets	array	Вложенные документы ставок
bank	string (URL)	Ссылка на ЦБ/инфу о выпуске
photo	string (base64/path)	Фото монеты
is_active	boolean	Идут ли торги
deleted	boolean	Удалён ли лот

Связь: один аукцион содержит много ставок. Каждая ставка — это объект с nickname, bet_cost, clients_id, created_at.

3. admins — Администраторы:

Таблица 3 — Администраторы

Поле	Тип	Описание
_id	ObjectId	ID администратора
email	string	Email администратора

Используется для разграничения доступа, например, к модерации лотов.

4. tokens — Токены сброса пароля / подтверждения:

Таблица 4 — Токены

Поле	Тип	Описание
_id	ObjectId	ID документа
email	string	Email, к которому привязан токен
token	string	Сам токен
expire_at	datetime	Время истечения срока действия

Используется для восстановления доступа.

5. i_counters — Счетчики идентификаторов

MongoDB не использует автоинкременты, поэтому в системе реализован ручной счетчик.

Таблица 5 — Счетчики

Поле	Тип	Описание
_id	ObjectId	ID документа
a_id	int	Последний ID для аукционов
clients_id	int	Последний ID для пользователей
c_id	int	Дополнительный счётчик (например, ставок)

Связи между коллекциями (логические):

- clients.id → используется в auctions.bets.clients_id — чтобы связать пользователя и его ставку;
- auctions.a_id — можно использовать как внешний ID для ссылок, истории, аналитики;
- admins.email ↔ clients.email — пересечение возможно для прав администратора.

2.3 Диаграмма компонентов

Информационная система состоит из трех основных компонентов:

1. Веб-клиент (Frontend)

Назначение: предоставляет пользовательский интерфейс для участия в торгах, регистрации, авторизации, просмотра лотов и ставок. Интерфейс реализован с помощью инструмента текстовой разметки HTML, каскадной таблицы стилей CSS и языка программирования JavaScript

Основной функционал:

- регистрация и авторизация;
- просмотр текущих и завершённых аукционов;
- участие в торгах (ставки);
- уведомления о победе или перебитой ставке.

2. Backend-сервер (FastAPI)

Назначение: обработка бизнес-логики, маршрутизация запросов, управление пользователями, лотами и ставками. Реализован при помощи языка Python с интеграцией фреймворка FastAPI

REST API эндпоинты:

- /auth/register, /auth/login;
- /auctions/ — список активных аукционов;
- /auctions/{id} — информация о лоте;
- /bets/ — создание ставок;
- /account — личный кабинет пользователя.

Дополнительные функции:

- проверка ролей (админ/пользователь);
- расчёт победителя по окончании аукциона;
- уведомления на email.

3. База данных (MongoDB)

Назначение: хранение информации о пользователях, лотах, ставках, сессиях, настройках уведомлений и ID-счетчиках.

Коллекции:

- clients — пользователи;

- auctions — аукционы (с вложенными ставками);
- admins — список администраторов;
- tokens — временные токены (например, сброс пароля);
- id_counters — ручное управление ID.

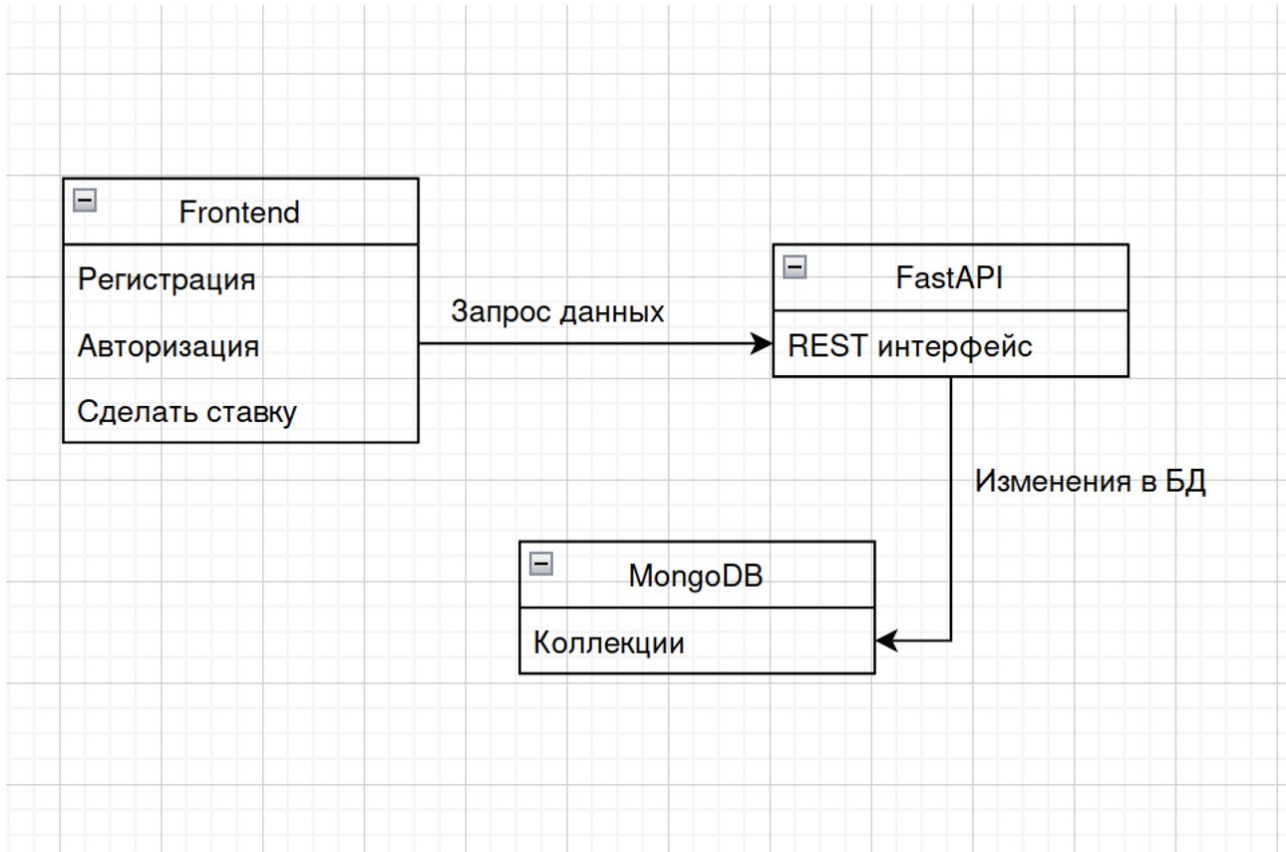


Рисунок 1 – Диаграмма компонентов

2.4 Бизнес-логика и пользовательские сценарии

Бизнес-логика системы онлайн-торгов монетами реализует ключевые процессы, обеспечивающие корректное проведение аукционов, взаимодействие пользователей и безопасность сделок. Основные компоненты:

1. Управление аукционами:

- создание лота: продавец заполняет данные (название, описание, стартовая цена, сроки торгов), загружает фото. Система проверяет обязательные поля и присваивает уникальный ID;
- автоматическое завершение торгов: по истечении времени `end_datetime` система определяет победителя (участник с максимальной ставкой) и уведомляет всех заинтересованных.

2. Система ставок:

- проверка минимального шага ставки: новая ставка должна быть выше текущей на `min_bid_step`;
- обновление цены лота в реальном времени через WebSocket или polling;
- фиксация истории ставок для аудита (сохранение `bet_cost`, `clients_id`, `created_at`).

3. Безопасность и валидация:

- только авторизованные пользователи могут делать ставки;
- только администраторы могут редактировать и создавать лоты;
- панель администратора не связана с главным сайтом.

4. Email уведомления:

- перебитии ставки;
- победе в аукционе;
- начале/завершении торгов.

Пользовательские сценарии:

1. Регистрация и вход (действия пользователя):

- переходит на страницу регистрации;

- заполняет форму (email, пароль, никнейм);
- получает письмо с подтверждением;
- входит в систему через логин/пароль.

Ответ системы:

- создаёт запись в коллекции clients;
- генерирует токен для авторизации.

2. Создание лота (действия пользователя):

- администратор в админ панели нажимает "Создать лот";
- загружает фото монеты, указывает стартовую цену и сроки;
- подтверждает размещение.

Ответ системы:

- проверяет данные (например, `start_price > 0`);
- сохраняет лот в коллекции auctions с `is_active=true`.

3. Участие в торгах (действия пользователя)

- просматривает активные лоты;
- выбирает монету, вводит ставку (например, 10 000 руб.).

Ответ системы:

- проверяет, что ставка $\geq \text{current_price} + \text{min_bid_step}$;
- обновляет цену лота и добавляет ставку в массив bets;
- отправляет уведомление предыдущему лидеру.

4. Завершение аукциона

Действия системы:

- при достижении `end_datetime` выбирает победителя;
- помечает лот как `is_active=false`;
- отправляет победителю письмо с инструкциями по оплате.

5. Администрирование

Действия администратора:

- просматривает список лотов через `/admin/auctions`;
- удаляет лот по усмотрению.

Ответ системы:

- меняет статус лота на `deleted=true`;
- уведомляет владельца о причине удаления.

2.5 Безопасность и авторизация

Система предусматривает многоуровневую защиту данных пользователей и реализует безопасные методы аутентификации и авторизации. Все ключевые аспекты безопасности были учтены на этапах проектирования и разработки:

1. Хранение учетных данных

Когда пользователь регистрируется в системе, его данные — такие как номер телефона, электронная почта, имя и пароль — сохраняются в базе данных. Пароли пользователей не хранятся в открытом виде: перед сохранением они проходят необратимое хэширование с использованием криптографической функции SHA-256. Это означает, что даже в случае компрометации базы данных злоумышленник не сможет восстановить оригинальный пароль. Хеш-функция применяется в совокупности с солью, что дополнительно защищает от атак по словарю и радужных таблиц.

2. Защищённый обмен данными

Вся передача данных между клиентским приложением (например, веб-браузером) и сервером осуществляется по протоколу HTTPS с использованием TLS-сертификатов. Это исключает возможность перехвата чувствительной информации (например, токенов, логинов и паролей) при передаче данных в открытых сетях.

3. Аутентификация и управление сессией

Для управления доступом используется реализация системы токенов. После успешного входа пользователь получает `access token`, который используется при совершении защищённых операций. Доступ к определённым маршрутам ограничен с помощью `middleware`-слоя, проверяющего валидность

токена. В случае отсутствия или истечения срока действия токена запрос блокируется, что предотвращает несанкционированный доступ.

4. Ограничение прав доступа (авторизация)

Система различает несколько уровней пользователей: обычный пользователь и администратор. Определённые маршруты доступны только администраторам (например, удаление клиентов, просмотр всех пользователей, доступ к административной панели). Это разграничение реализовано с помощью проверки ролей в access token. Также функционал разработанный для администраторов площадки разворачивается на отдельном от клиентов сервисе.

5. Защита от атак

Предусмотрены базовые меры защиты от наиболее распространённых угроз, включая:

- Защиту от SQL-инъекций путём использования ORM (в данном случае, доступ к базе MongoDB с использованием безопасных методов);
- Ограничение числа запросов с одного IP-адреса;
- Защиту от XSS и CSRF в клиентской части административной панели.

Глава 3. Реализация системы

3.1 Используемые технологии

Выбор инструментов и технологий для реализации системы выполнялся на основе следующих основных критериев:

- **производительность:** способность системы обрабатывать большое число запросов и проводить расчёты (аукционные торги) без значительных задержек;
- **безопасность:** поддержка надёжных механизмов хранения и передачи данных, защита от веб-уязвимостей (SQL-инъекции, XSS, CSRF и др.);
- **совместимость:** использование кроссплатформенных стандартов и библиотек, возможность интеграции с внешними сервисами;
- **масштабируемость:** способность системы расширяться при возрастании нагрузки, включая возможность горизонтального увеличения вычислительных ресурсов.

Приложение поделено на две части: клиентская и администраторская, которая работает отдельно от системы торгов. На основании вышеперечисленных критериев был выполнен выбор стек-технологий:

- **серверная часть:** выбран язык программирования Python с веб-фреймворком FastAPI. Этот стек обеспечивает быстрое прототипирование и включает встроенные механизмы безопасности: система аутентификации пользователей, защита от CSRF и XSS атак, а также ORM для удобной работы с базой данных. FastAPI легко масштабируется и поддерживает множество сторонних библиотек. В этом наборе инструментов реализованы практически все необходимые сценарии, а недостающие компоненты можно импортировать благодаря полной совместимости пакетов. Также на серверной части используется JinjaTemplates — пакет для более комфортной индексации шаблонных файлов и используется для отправки статического контента на веб клиент. В качестве ORM для базы данных выбран асинхронный движок Motor для

MongoDB из пакета Pymongo. Для поддержки токенов авторизации используется FastAPILogin;

- **база данных:** в качестве базы данных выбрана документоориентированная MongoDB. Ключевым решением для выбора этой СУБД стал формат хранения данных в базе. Традиционно в MongoDB используется BSON, который при ответе на запросы преобразовывается в JSON, родной формат для браузера. Вторым поводом выбрать эту СУБД стал тот, факт, что архитектура базы данных подразумевает в себе отношения между документами только в единичных случаях, поэтому возможность SQL-like баз в данном случае избыточны;

- **клиентская часть (frontend):** в качестве фронтенд реализации предпочтение отдано классическому набору – HTML, CSS, JavaScript. Фреймворки хоть и дают эффективность при разработке, но всегда проигрывают в скорости обработки браузером, ведь для того, чтобы фреймворк отработал, на клиент нужно также отправить и пакеты с фреймворком, которые часто могут в объеме превосходить написанный код в десятки раз. Отдельное внимание уделено безопасности, так как невнимательность при разработке клиентской части зачастую становится лазейкой для угрозы кибер атаки. Все поля ввода и запросы экранируются как на сервере, так и на клиенте, чтобы избежать атаки. На клиентскую часть не поступает никакой информации, которая могла бы описать размеры базы данных и данные пользователей;

- **инфраструктура и развертывание:** для хостинга серверного приложения и базы данных выбран облачный провайдер VDS/VPS. Облачная инфраструктура позволяет динамически масштабировать ресурсы (виртуальные машины, базы данных) в зависимости от нагрузки. Также выбранный подход написания дает возможность распределить ключевые части по Docker контейнерам и впоследствии подключить метрики. Такое развертывание является эталонным в современной разработке, а при смене сервера позволяет перенести все части приложения на новые машины в несколько шагов.

Данные технологические решения удовлетворяют заявленным критериям: они обеспечивают высокую производительность и отказоустойчивость сервиса, совместимы с современными стандартами веб-разработки, а также предоставляют встроенные средства безопасности и возможности масштабирования при расширении системы.

3.2 Основные модули системы

Серверная часть системы построена по модульному принципу, где каждый компонент выполняет строго определённую функцию. Такой подход упрощает масштабирование, отладку и тестирование системы. Ниже приведено описание ключевых модулей, из которых состоит серверное приложение.

- **точка входа приложения:**

Это основной скрипт `__main__.py`, с которого начинается выполнение всей серверной логики. Он инициализирует веб-сервер, регистрирует маршруты, подключает модули и запускает приложение. Является точкой запуска как в локальной среде разработки, так и при деплое на сервер;

- **инициализация параметров запуска:**

Модуль конфигурации отвечает за чтение и применение параметров запуска сервера. В нём устанавливаются ключевые параметры, такие как порт, на котором запускается приложение, пути к ресурсам, ключи авторизации, параметры подключения к базе данных и настройки SMTP. Обычно реализуется через конфигурационный файл (`.env`, `config.py`) или систему переменных окружения;

- **подключение маршрутов (роутов):**

Этот модуль включает в себя все сценарии обработки запросов от клиента. Он регистрирует API-эндпойнты и связывает их с соответствующими обработчиками, например: `/login`, `/register`, `/auctions`, `/admin/users` и т. д. Обеспечивает маршрутизацию запросов по REST-принципам, а также назначение middleware-функций для проверки авторизации и логирования;

- **модуль управления пользователями в БД:**

Содержит бизнес-логику, связанную с пользователями: регистрация, вход, получение информации о пользователе, изменение профиля, удаление. Работает напрямую с базой данных и реализует безопасные методы CRUD-операций (создание, чтение, обновление, удаление);

– **модуль управления аукционами в БД:**

Осуществляет всю работу с сущностью "аукцион". Включает создание новых аукционов, редактирование, отображение всех или одного, фильтрацию по категориям и статусу (активный, завершённый). Также может включать механизмы ставок, таймеров и завершения лотов;

– **модуль отправки email по протоколу SMTP**

Отвечает за отправку электронных писем — например, писем подтверждения регистрации, уведомлений о завершении аукциона или действий администратора. Работает с SMTP-сервером (например, Gmail или корпоративным сервером), формирует письма в формате HTML и Plain Text;

– **модуль авторизации:**

Реализует механизм аутентификации с использованием токенов. Обрабатывает вход, проверку пароля, генерацию access token и проверку их действительности при доступе к защищённым маршрутам. Поддерживает разграничение прав доступа (например, администратор и обычный пользователь);

– **модуль отправки статического контента:**

Этот компонент отвечает за обработку и раздачу файлов, таких как изображения товаров, скрипты интерфейса, стили CSS и медиа-контент. Использует FastAPI StaticFiles, механизм для обслуживания frontend-ресурсов;

– **модуль SEO-оптимизации:**

Этот модуль занимается генерацией динамического HTML с необходимыми мета-тегами, чтобы страницы корректно индексировались поисковыми системами. Может поддерживать Open Graph, корректные заголовки, описания и canonical-ссылки для повышения видимости сайта;

– **модуль панели администратора:**

В этом компоненте реализована внутренняя логика для административного интерфейса. Он обеспечивает доступ к управлению пользователями, аукционами, логами, сообщениями и прочими ключевыми

элементами системы. Доступ к этому модулю строго ограничен ролью администратора и защищён дополнительными механизмами авторизации.

3.3 Примеры REST-запросов

Для того, чтобы клиент мог взаимодействовать с сервером, необходимо определить, по каким адресам клиент сможет запрашивать информацию и какие данные возвращать. Для удобной документации API используется стандарт OpenAPI. В таблице 6 приведу все эндпоинты сервера.

Таблица 6 — Эндпоинты приложения

Метод	Путь	Требует токен?	Описание	Параметры / Тело запроса
POST	/auth/token	Нет	Логин (получение токена)	grant_type, username, password (form)
GET	/auth/get_data	Да	Получить данные пользователя	—
POST	/auth/logout	Да	Выход из аккаунта	—
POST	/auth/register	Нет	Регистрация	phone_number, password, nickname, email, avito_url
POST	/auth/check_mail	Нет	Проверить почту	mail (query param)
GET	/mongo/auction/get_all	Нет	Получить все аукционы	—
GET	/mongo/auction/get	Нет	Получить один аукцион	a_id (query param)
POST	/mongo/auction/add	Да	Добавить аукцион	JSON тело

Метод	Путь	Требует токен?	Описание	Параметры / Тело запроса
POST	/mongo/auction/ add_bet_to_auction	Да	Добавить ставку в аукцион	a_id, bet_cost (query params)
POST	/mongo/auction/ update_auction	Да	Обновить аукцион	JSON тело
DELETE	/mongo/auction/delete	Да	Удалить аукцион	a_id (query param)
GET	/mongo/auction/ get_bets	Нет	Получить ставки по аукциону	a_id (query param)
GET	/mongo/auction/ get_time	Нет	Получить оставшееся время аукциона	a_id (query param)
GET	/mongo/clients/get_all	Да	Получить всех клиентов	—
GET	/mongo/clients/get	Нет	Получить одного клиента	clients_id (query param)
POST	/mongo/clients/add	Нет	Добавить клиента	phone_number, password, email, nickname, avito_url
POST	/mongo/clients/update	Да	Обновить клиента	JSON тело
POST	/mongo/clients/ change_password	Да	Сменить пароль клиента	JSON тело
POST	/mongo/clients/ active_clients	Да	Сделать клиентов активными	—
DELETE	/mongo/clients/ban	Да	Забанить	id (query param)

Метод	Путь	Требуется токен?	Описание	Параметры / Тело запроса
			клиента	
POST	/mongo/clients/unban	Да	Разбанить клиента	id (query param)
POST	/mongo/clients/edit	Да	Редактировать клиента	JSON тело
GET	/robots.txt	Нет	robots.txt для SEO	—
GET	/sitemap.xml	Нет	sitemap.xml для SEO	—
GET	/mailservice/verify	Нет	Подтвердить email	token (query param)

3.4 Скриншоты пользовательского интерфейса

Для того, чтобы пользователю было приятно пользоваться информационной системой, скорости обработки данных недостаточно, также необходимо чтобы навигация по приложению была удобной, а UI/UX дизайн продуманным.

Чтобы решить проблему удобства, решено опираться на решения в других системах, которые с торгами могут быть даже не связанными. В итоге было выбрано решение следующего типа: малый набор цветовой палитры, разделение активной области на две части, первая из которых вертикальное навигационное меню в левой части области, и вторая, которая занимает 80% пространства и включает в себя область, в которую от контекста добавляется информация актуальная для конкретного отдела. Выбор навигационного меню вертикального типа обоснован тем, что в процессе жизни системы может потребоваться дополнение новыми элементами управления. Так как

вертикальное меню легко пролистывается, его удобство использование с увеличением количества вложенных элементов не изменяется, в отличии от горизонтального меню на верху или внизу рабочего пространства.

По ГОСТ 9241-210–2014 "Эргономика взаимодействия человек-система. Принципы диалогового интерфейса" Приведу критерии, которым интерфейс должен следовать:

- **Соответствие задачам пользователя:** Интерфейс должен поддерживать цели и задачи пользователя;
- **Самоописуемость:** Пользователь должен понимать, что происходит, без обращения к справке.
- **Управляемость:** Пользователь должен контролировать действия системы.
- **Устойчивость к ошибкам:** Интерфейс должен предотвращать ошибки и помогать восстанавливаться.
- **Эстетика и минимизация нагрузки:** Интерфейс не должен быть перегружен, должен быть визуально чистым.

Главная страница (Рисунок 2) приложения содержит в себе информацию об аукционах, серым цветом обозначены прошедшие, а активные имеют полную цветовую палитру.

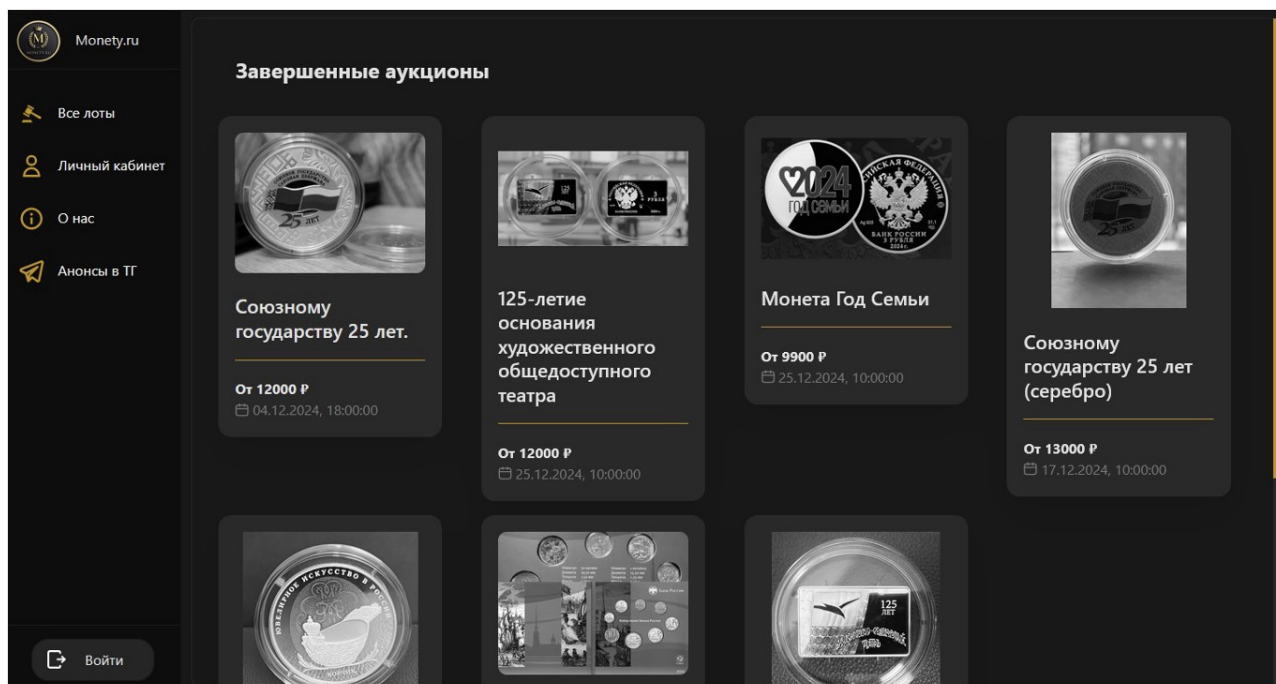


Рисунок 2 – Главная страница приложения

Страница лота (Рисунок 3) описывает всю необходимую информацию о монете, с фотографиями, характеристиками и ссылкой на подробности на ресурсе Банка России, актуальными ставками и интерфейсом для участия в торгах.

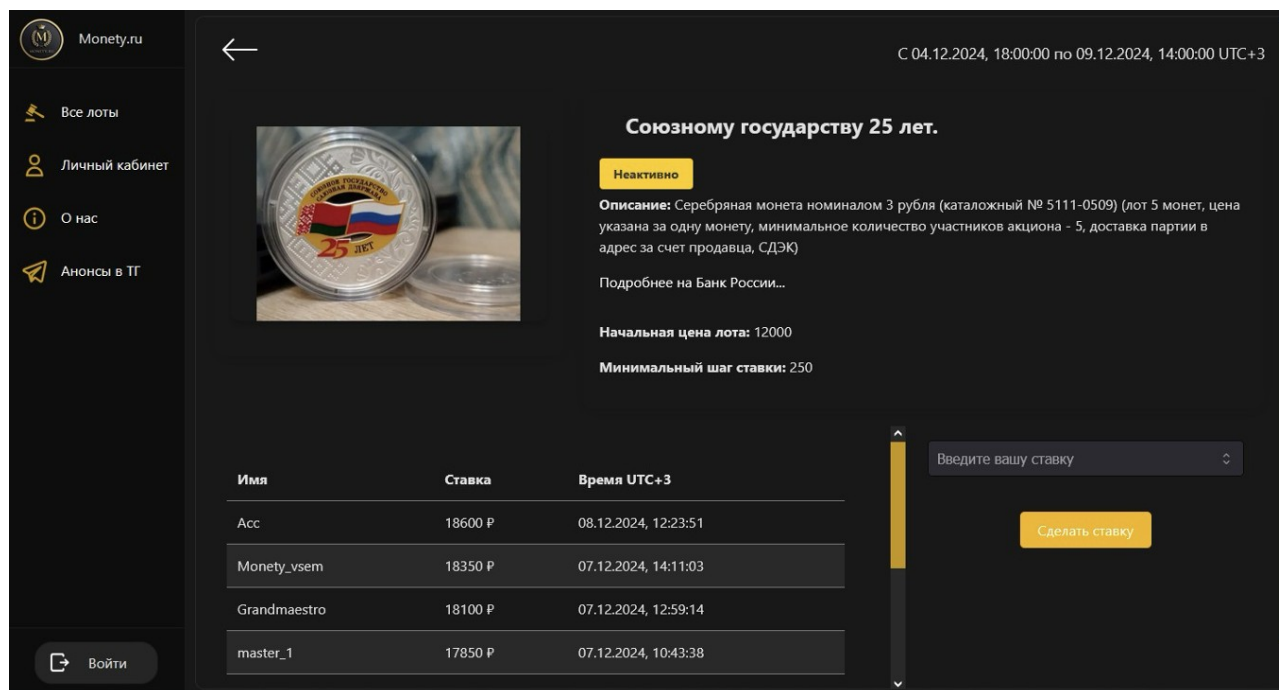


Рисунок 3 – Страница ставок

Личный кабинет (Рисунок 4) позволяет пользователю актуализировать некоторые данные и при желании произвести сброс пароля.

Рисунок 4 – Личный кабинет

Страница «О нас» (Рисунок 5) описывает информацию о компании, которая проводит аукционы. Здесь пользователь может подробно ознакомиться с организацией.

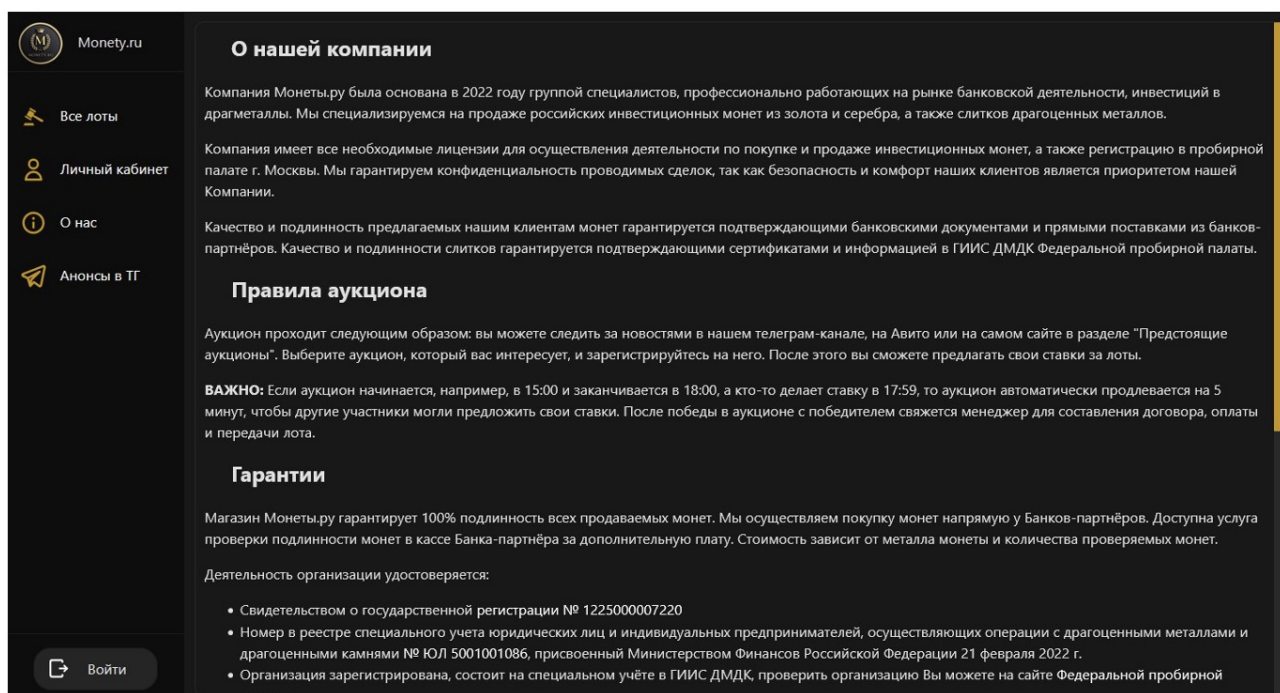


Рисунок 5 – Страница с информацией о компании

Панель Администратора (Рисунок 6) содержит инструменты для модерации всех сущностей в системе торгов.

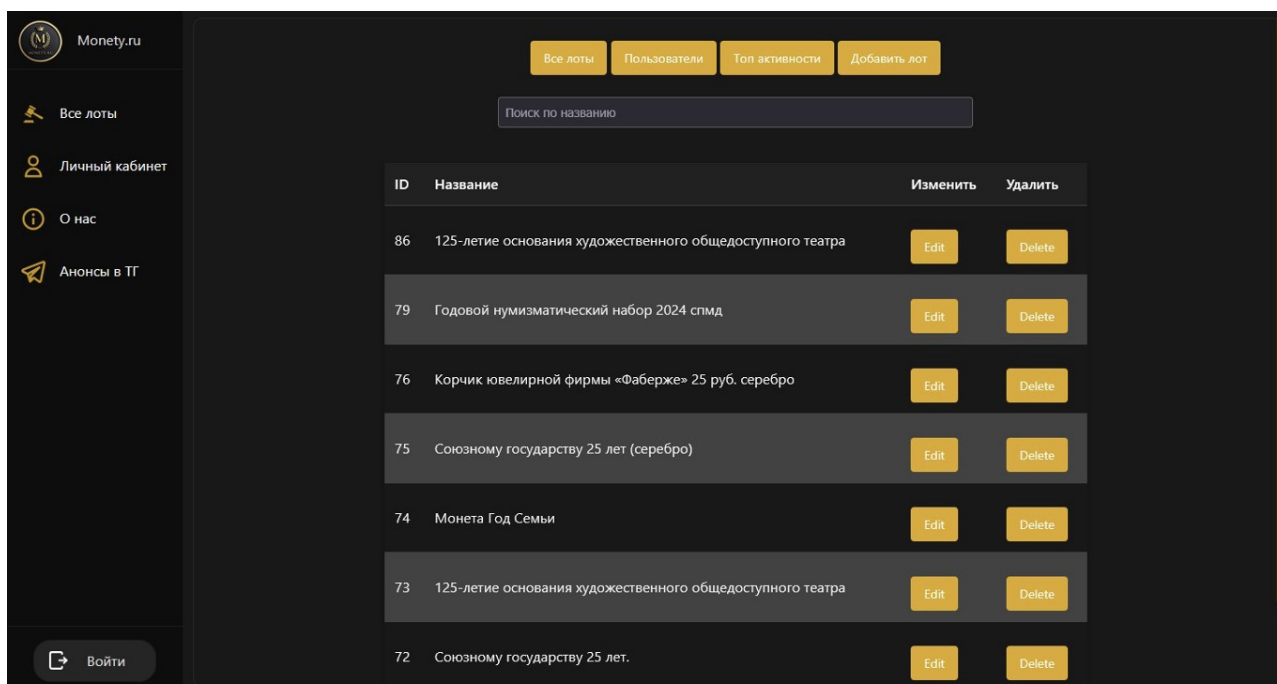


Рисунок 6 – Панель администратора

Окно для редактирования аукциона (Рисунок 7) позволяет администратору указать актуальные свойства аукциона, изменить название, фотографии, начальную цену, шаг ставки и другие параметры лота.

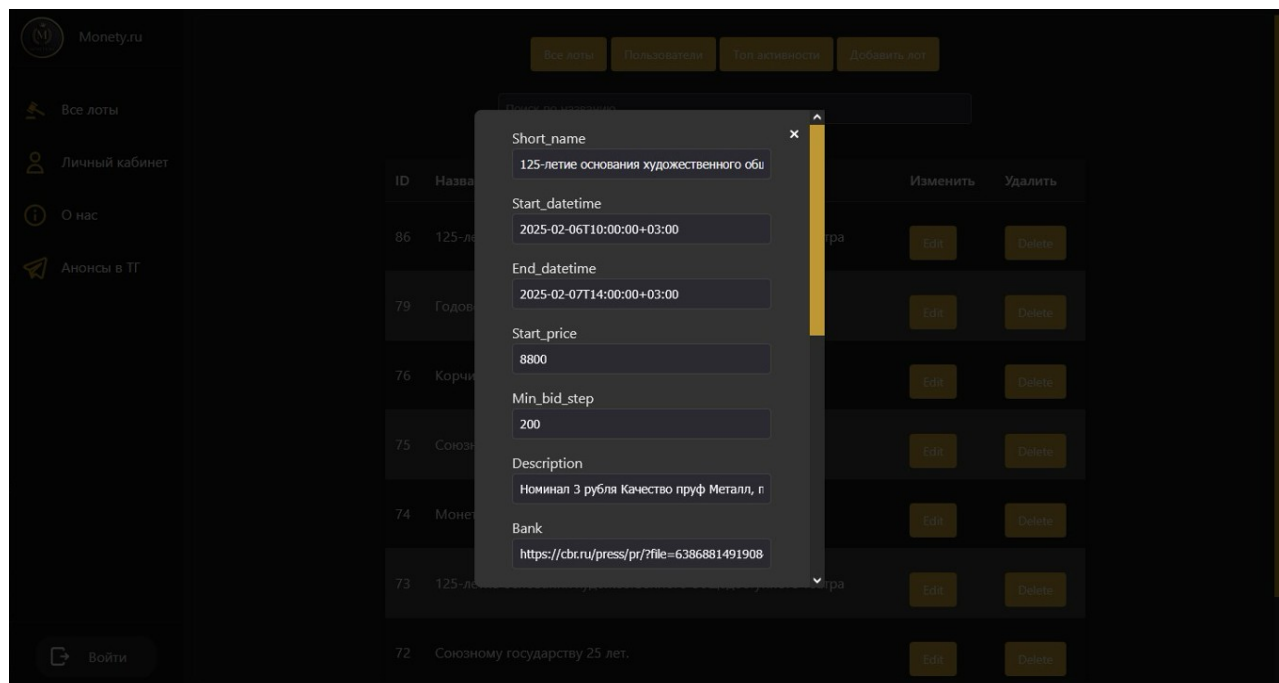


Рисунок 7 – Окно для редактирования аукциона

Использование поиска по пользователям (Рисунок 8) — в этом меню администратор может осуществить быстрый поиск пользователей, написав в поле ввода ключевые символы. Поиск происходит по поиску подстроки в данных почты, имени и номера телефона в базе данных.

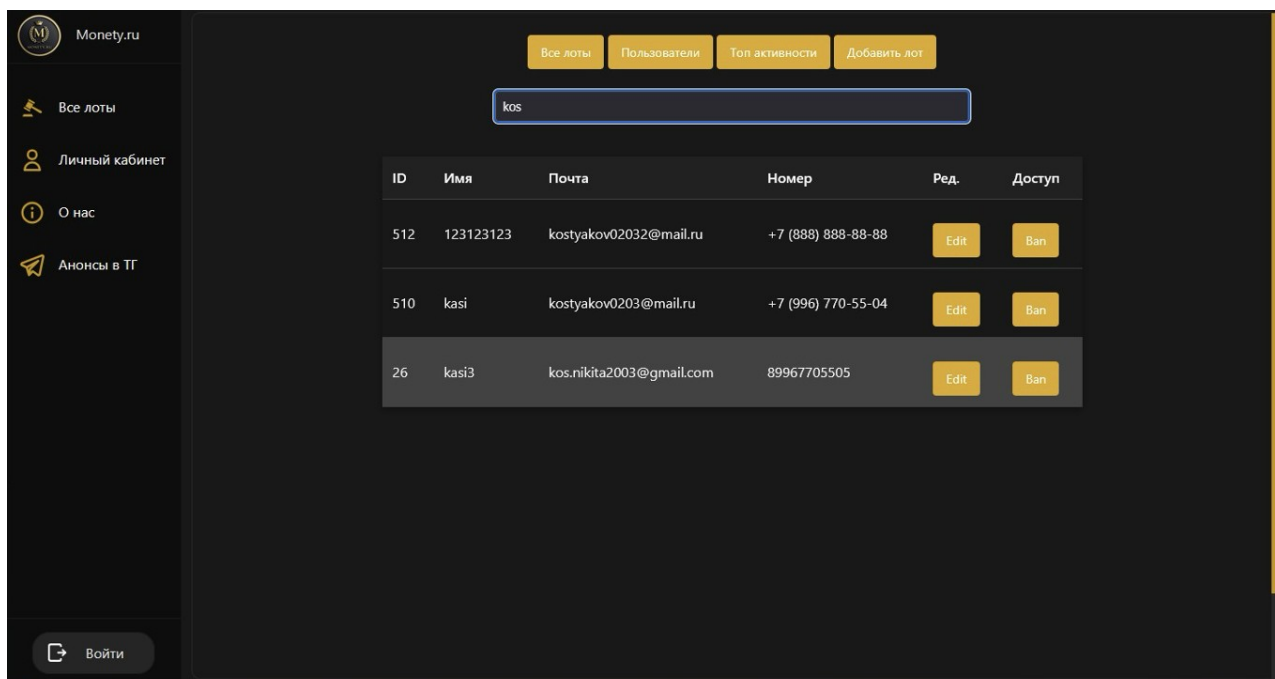


Рисунок 8 – Использование поиска по пользователям

Таблица всех пользователей (Рисунок 9) – меню для навигации по базе данных пользователей

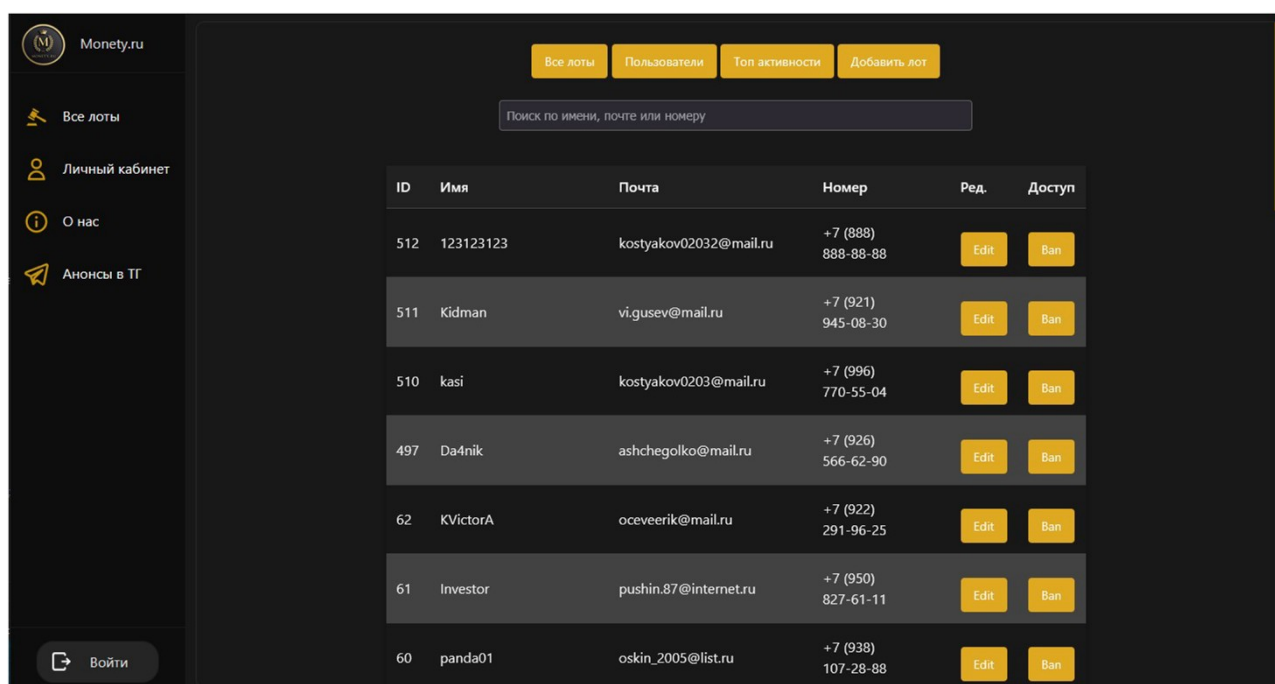
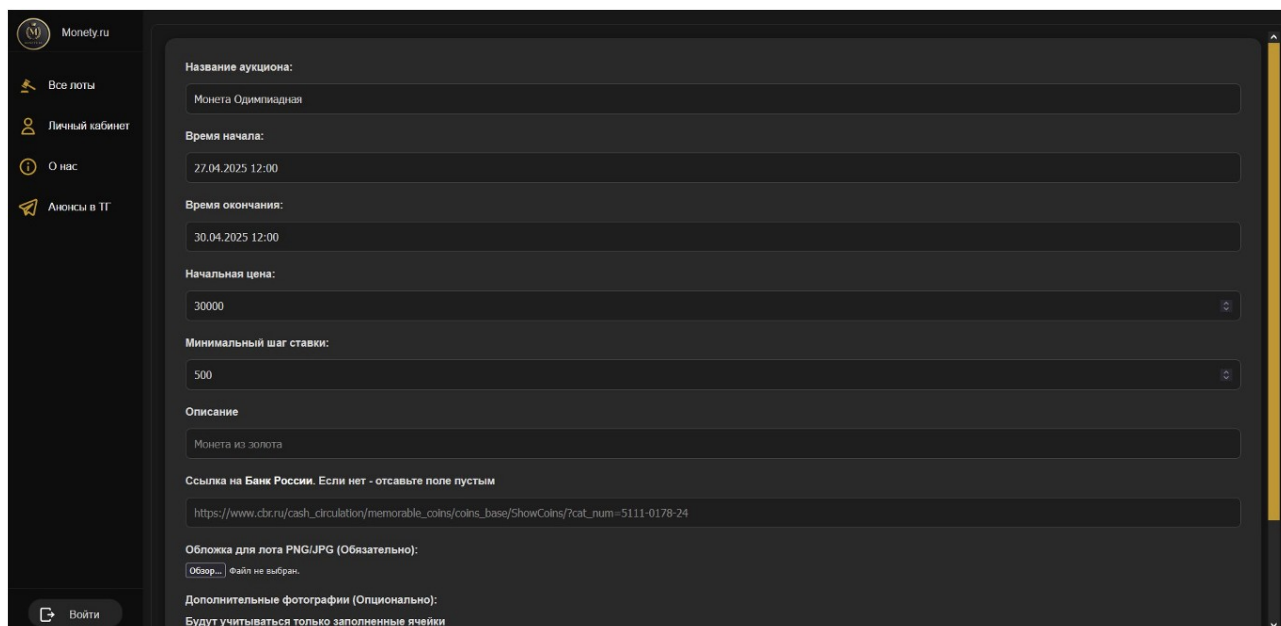


Рисунок 9 – Таблица всех пользователей

Добавление аукциона (Рисунок 10) — в этом окне Администратор определяет новый лот, чтобы на главной странице системы пользователи могли принять участие в торгах. В этом меню необходимо описать название лота, сроки проведения торгов, начальную цену, шаг ставки, привести описание лота, и при наличии ссылку на официальный ресурс монеты на Банк России и фотографии монеты.



The screenshot shows a web interface for adding an auction. On the left is a dark sidebar with the 'Monet.ru' logo and navigation links: 'Все лоты', 'Личный кабинет', 'О нас', and 'Лоты в ТГ'. The main area is a dark-themed form with the following fields:

- Название аукциона:** Text input with 'Монета Олимпийская'.
- Время начала:** Date and time picker set to '27.04.2025 12:00'.
- Время окончания:** Date and time picker set to '30.04.2025 12:00'.
- Начальная цена:** Text input with '30000' and a spinner icon.
- Минимальный шаг ставки:** Text input with '500' and a spinner icon.
- Описание:** Text input with 'Монета из золота'.
- Ссылка на Банк России. Если нет - оставьте поле пустым:** Text input with a long URL.
- Обложка для лота PNG/JPG (Обязательно):** File upload button labeled 'Обзор...' with the text 'Файл не выбран'.
- Дополнительные фотографии (Опционально):** Section header.
- Будут учитываться только заполненные ячейки** (Note below the photo section).

At the bottom left of the sidebar is a 'Войти' button with a key icon.

Рисунок 10 – Добавление аукциона

Страница входа в систему (Рисунок 11) — так выглядит страница авторизации пользователя в системе. Гостю необходимо ввести пару логин-пароль для доступа к ресурсу. Если пользователь до этого в системе зарегистрирован не был, получит уведомление о необходимости первичной регистрации на этой же странице во вкладке регистрации.

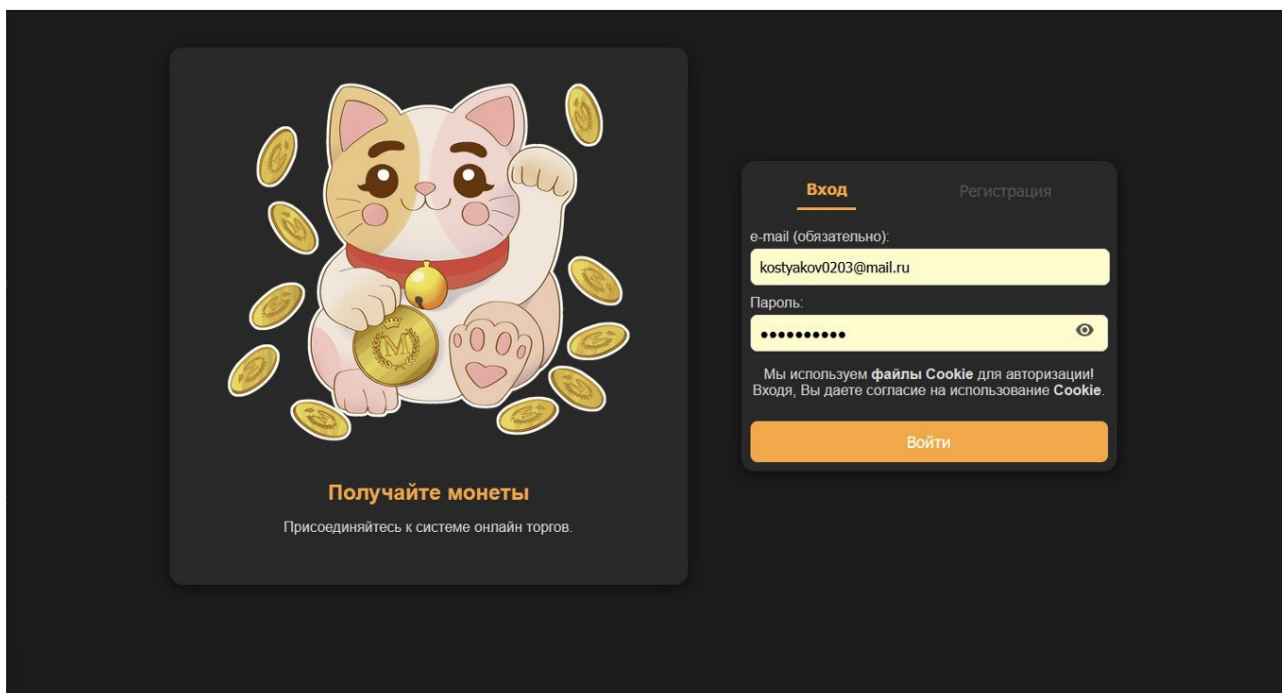


Рисунок 11 — Страница входа в систему

3.5 Тестирование и отладка

Для проверки корректности работы API предусмотрены несколько способов тестирования.

1. Использование Swagger UI

FastAPI автоматически генерирует **интерактивную документацию** по адресу: `http://localhost:8000/docs`

В Swagger UI можно:

- отправлять тестовые запросы к API;
- проверять ответы и коды статусов;
- передавать токен авторизации через кнопку **Authorize**.

2. Postman

Для более глубокой отладки рекомендуется использовать **Postman**:

- импортировать коллекцию запросов;
- устанавливать токены авторизации;
- сохранять сценарии тестирования.

3. curl или httpie

Тестировать отдельные запросы из консоли:

Пример с помощью curl:

```
curl -X POST "http://localhost:8000/auth/token" \  
-H "Content-Type: application/x-www-form-urlencoded" \  
-d "username=user@example.com&password=string"
```

Пример с помощью httpie:

```
http POST http://localhost:8000/auth/token username=user@example.com  
password=string
```

4. Логи сервера

Для отладки ошибок следует:

- проверять вывод консоли, где запущен FastAPI-сервер;
- логи содержат информацию об ошибках валидации, неправильных маршрутах и статусах ответов.

Глава 4. Экономическое обоснование

4.1 Расчёт затрат на разработку

Расчет затрат на разработку основывается на следующих параметрах

Таблица 7 — Затраты на разработку

Этап работы	Описание этапа	Оценка времени	Ставка (руб/час)	Стоимость (руб)
Проектирование API	Разработка структуры маршрутов, авторизация, безопасности	8 ч	1500	12 000
Реализация функционала	Написание эндпоинтов, логики работы аукционов и клиентов	60 ч	1500	90 000
Разработка административной панели	Интерфейс управления аукционами и пользователями	15 ч	1500	22 500
Тестирование и отладка	Unit-тесты, интеграционное тестирование, исправление багов	10 ч	1500	15 000
Подготовка документации	Руководство пользователя, описание API	5 ч	1500	7 500
Развёртывание на сервере	Настройка сервера, деплой проекта	5 ч	1500	7 500

Итого:

- Общее время: 103 часа;
- Общая стоимость: 154 500 руб.

4.2 Оценка стоимости системы

Стоимость системы складывается из нескольких ключевых факторов:

Таблица 8 — Оценка стоимости системы

Компонент системы	Описание	Ориентировочная стоимость (руб)
Разработка программного обеспечения	Стоимость создания серверной части и админ-панели	154 500
Серверное оборудование / хостинг	Аренда VPS-сервера (12 мес. по 1000 руб/мес)	12 000
Системное и сервисное ПО	Подписки на сторонние сервисы (почтовые API, домен)	3 000
Техническая поддержка и сопровождение	Ежемесячная поддержка и обновления (12 мес. по 15000 руб/мес)	180 000

Итого:

- начальная стоимость разработки и запуска: 158 000 руб;
- дополнительные расходы на 12 месяцев эксплуатации: 180 000 руб;
- полная стоимость системы за первый годовой цикл - 338 000 руб.

Примечания:

- в дальнейшем стоимость эксплуатации может быть снижена за счёт оптимизации серверных мощностей;

- расходы на поддержку могут варьироваться в зависимости от количества доработок и пользовательской активности;
- при расширении функционала (например, интеграция платёжных систем) стоимость возрастёт.

4.3 Сравнение с альтернативами

При выборе решения для реализации системы управления аукционами были рассмотрены следующие альтернативные варианты:

Таблица 9 — Сравнение альтернативных вариантов разработки

Вариант	Преимущества	Недостатки	Оценка стоимости (руб)
Разработка на заказ (мой проект)	Полная адаптация под бизнес-процессы, гибкость, расширяемость	Требуется время на разработку и тестирование	154 000
Готовые SaaS-решения (например, AuctionSoftware, WeAuction)	Быстрый запуск, техподдержка включена	Высокая абонентская плата, ограниченные возможности кастомизации	от 15 000 руб/мес → 180 000 за 12 мес
Опенсорс-платформы (например, Sharetribe, OpenAuction)	Бесплатное ПО, доступ к коду	Требуется значительные доработки и сопровождение	70 000 (доработки + внедрение)
Разработка на конструкторах сайтов (Tilda, Wix + доп.	Быстрая разработка интерфейса,	Ограничения по функциональности, сложная интеграция	40 000

Вариант	Преимущества	Недостатки	Оценка стоимости (руб)
скрипты)	низкий порог входа	со сторонними системами	

Вывод:

- разработка **собственной системы** является наиболее целесообразным вариантом при необходимости высокой гибкости, масштабируемости и контролируемых затрат на долгосрочную перспективу;
- **готовые решения** целесообразны только для очень маленьких проектов с минимальными требованиями;
- **опенсорс** — хорош для старта, но требует ресурсов для поддержки и развития;
- **конструкторы** удобны для витринных сайтов, но не подходят для сложной бизнес-логики (аукционы, ставки, лоты).

4.4 Риски и способы их минимизации

Таблица 10 — Риски и способы их минимизации

Риск	Возможные последствия	Способы минимизации
Технические ошибки при разработке	Сбои в работе системы, потеря данных	Многоуровневое тестирование (юнит-, интеграционное), ревью кода
Недостаточная защита данных пользователей	Утечка персональной информации, штрафы	Использование SSL/TLS, шифрование данных, аудит безопасности

Риск	Возможные последствия	Способы минимизации
Невозможность масштабирования системы	Ограничение роста бизнеса	Проектирование масштабируемой архитектуры, использование облачных решений
Рост затрат на поддержку	Увеличение эксплуатационных расходов	Документирование системы, автоматизация мониторинга и развертывания
Низкая вовлеченность пользователей	Малая активность на платформе	Упрощение пользовательского интерфейса, маркетинговые кампании
Проблемы с интеграцией сторонних сервисов	Ограничение функциональности	Выбор стабильных API-провайдеров, создание резервных планов
Задержки в сроках разработки	Срыв запуска проекта	Agile-методология, регулярные планерки, контроль выполнения задач

Основная стратегия минимизации рисков:

- использование **гибких методов управления проектами** (Scrum, Kanban);
- проведение **регулярных внутренних проверок качества**;
- постоянная **обратная связь с тестовой группой пользователей**;
- создание **резервных копий базы данных** и планов аварийного восстановления.

Глава 5. Защита данных и безопасность

5.1 Актуальные угрозы

Таблица 11 — актуальные угрозы

Угроза	Возможные последствия	Способы защиты
Несанкционированный доступ	Кража данных, управление аккаунтами	Аутентификация, авторизация, логирование
Атаки на отказ в обслуживании (DDoS)	Недоступность сервиса	Использование анти-DDoS сервисов, фильтрация трафика
Утечка персональных данных	Нарушение закона о защите данных, штрафы	Шифрование хранения и передачи данных
Вредоносное ПО на стороне клиента	Кража данных пользователей	Защита от XSS и CSRF атак
Взлом API-интерфейсов	Неавторизованный доступ к функциям системы	Ограничение доступа, использование токенов
Фишинговые атаки на пользователей	Потеря данных для входа	Обучение пользователей, двухфакторная аутентификация
Ошибки конфигурации серверов и баз данных	Потеря целостности или доступности системы	Регулярный аудит конфигураций, автоматизированные проверки
Уязвимости сторонних библиотек и фреймворков	Компрометация всей системы	Постоянное обновление зависимостей,

Угроза	Возможные последствия	Способы защиты
		мониторинг CVE

5.2 Средства защиты

Важные направления защиты:

— **Принцип минимально необходимого доступа:** каждому пользователю и компоненту системы предоставляются только те права, которые необходимы для выполнения его задач. Это снижает вероятность несанкционированного доступа и ограничивает масштаб возможных повреждений при компрометации учётной записи.

— **Мониторинг событий безопасности в реальном времени:** серверная часть системы журналирует критически важные события: входы в систему, ошибки авторизации, обращения к защищённым маршрутам. Логи могут анализироваться вручную либо автоматически, с целью обнаружения подозрительной активности.

— **Регулярное обновление программного обеспечения и зависимостей:** все сторонние библиотеки (включая FastAPI, PyMongo, Motor и др.) периодически проверяются на наличие уязвимостей. При необходимости производится обновление до стабильных безопасных версий. Это особенно важно с учётом постоянного появления новых CVE (обнаруженных уязвимостей).

— **Резервное копирование данных и план восстановления после инцидентов:** ежедневно создаются зашифрованные резервные копии базы данных, которые хранятся в отдельном изолированном хранилище. В случае аппаратного сбоя, атаки или логической ошибки возможен полный откат до работоспособного состояния.

— **Шифрование передаваемых данных:** вся коммуникация между клиентом и сервером происходит по протоколу HTTPS с использованием TLS. Это

гарантирует защиту от перехвата данных (например, логинов, токенов или электронной почты) при передаче по сети.

— **Защита от автоматизированных атак и брутфорса:** система может быть дополнена ограничением количества запросов (rate limiting) и задержками после неудачных попыток входа, что снижает риск взлома паролей подбором.

— **Аудит и логирование действий администратора:** все действия, совершённые через административную панель (изменения данных, удаление аукционов, управление пользователями), фиксируются в журнале. Это позволяет отслеживать, кто и когда выполнял критические операции.

— **Использование безопасных токенов авторизации:** токены имеют ограниченный срок действия, подписаны секретным ключом и передаются только через защищённые каналы. Это снижает вероятность их подделки и несанкционированного использования.

— **Изоляция среды выполнения:** серверная часть может быть запущена в изолированном контейнере (Docker), что уменьшает влияние ошибок внутри приложения на другие части системы.

5.3 Регламент хранения и обработки данных

Для обеспечения безопасности, целостности и законности хранения данных в системе вводятся следующие правила:

Таблица 12 — Регламенты обработки данных

Параметр	Регламент
Категории обрабатываемых данных	Персональные данные (ФИО, email, телефон), данные о ставках и активности пользователей, служебные данные системы
Срок хранения персональных данных	Минимально необходимый для целей обработки. Обычно 1 год после удаления аккаунта, если иное не требуется законодательством

Параметр	Регламент
Хранилище данных	Защищённые базы данных на сервере с шифрованием на уровне хранения (например, MongoDB с включённым шифрованием)
Резервное копирование	Ежедневное автоматическое резервное копирование с хранением копий в зашифрованном виде на отдельном сервере в течение 30 дней
Передача данных	Только через защищённые каналы (HTTPS, SSL/TLS)
Доступ к данным	Ограниченный: только авторизованные сотрудники с необходимым уровнем доступа
Удаление данных	По запросу пользователя или по истечении срока хранения, с использованием безопасных методов удаления
Обработка данных третьими лицами	Только на основании договоров, гарантирующих соблюдение требований законодательства о защите данных
Журналирование событий	Ведение логов доступа и изменения данных с сохранением в течение 6 месяцев для анализа инцидентов безопасности
Соответствие законодательству	GDPR (при наличии пользователей из ЕС), ФЗ-152 (о персональных данных), другие применимые нормы

Дополнительные положения:

- пользователи имеют право на доступ к своим данным, их исправление и удаление;
- регулярные проверки и аудит процедур обработки данных;
- обработка специальных категорий данных (например, биометрических) запрещена без явного согласия пользователя.

Заключение

В ходе выполнения дипломной работы была разработана современная веб-система для управления аукционами и клиентскими данными, оснащённая административной панелью. Основные итоги работы можно сформулировать следующим образом:

- реализован полный цикл REST API для управления аукционами и пользователями, включая создание, обновление, удаление и получение данных;
- разработана безопасная система аутентификации и авторизации с использованием протокола OAuth2 и менеджера сессий;
- создана удобная и функциональная административная панель для управления ресурсами системы;
- проведено тестирование API-запросов для проверки корректности всех реализованных функций;
- оценена стоимость разработки и эксплуатации системы, произведено сравнение с альтернативными решениями;
- проанализированы риски и разработаны мероприятия по их минимизации;
- установлены регламенты хранения и обработки персональных данных в соответствии с современными стандартами безопасности и требованиями законодательства.

В результате система отвечает требованиям надёжности, безопасности и масштабируемости, а её архитектура позволяет быстро расширять функциональность при необходимости.

В дальнейшем проект можно развивать не только в сторону улучшения аукционной составляющей, но и добавлять новые способы торговли монетами, например разработать раздел для розничной продажи. Также помимо реализации английской системы, можно добавить и альтернативные способы. Когда клиентская база достигнет достаточного количества пользователей и коммуникация с ними будет достаточно объемной, можно подключить шлюз

для работы с CRM системами, а также разработать внутриплатформенный чат или единый инструмент для общения с клиентами через популярные мессенджеры.

Список использованных источников

1. [Документация по FastAPI. Sebastián Ramírez. *FastAPI Documentation*.](#)
2. [Python Software Foundation. *Python 3.12 Documentation*.](#)
3. [Armin Ronacher. *Jinja2 Documentation*.](#)
4. [MongoDB, Inc. *MongoDB Manual*.](#)
5. [Mark J. Bostic, David Beasley. *Motor Documentation*.](#)
6. [Docker, Inc. *Docker Documentation*.](#)
7. [Python Software Foundation. *PyMongo Documentation*.](#)
8. [*HTML, CSS, and JavaScript Guide*](#)
9. [Kristina Chodorow. *MongoDB: The Definitive Guide*. O'Reilly Media, 2013.](#)
10. [ISO. ISO/IEC 9241-210:2010 – Ergonomics of human-system interaction](#)
11. [Mozilla. Content Security Policy \(CSP\)](#)
12. [OWASP Foundation. OWASP Top 10: Web Application Security Risks.](#)
13. [SQL Injection Prevention Cheat Sheet. OWASP.](#)
14. [Cross-Site Scripting \(XSS\). OWASP.](#)
15. [JetBrains. PyCharm Documentation](#)
16. [Microsoft. Visual Studio Code Docs.](#)
17. [GitHub. Git Documentation.](#)
18. [GitHub REST API Docs.](#)
19. [Postman. API Platform Documentation.](#)
20. [Gunicorn. Python WSGI HTTP Server for UNIX.](#)
21. [Uvicorn. ASGI server for Python.](#)
22. [OpenAPI Initiative. OpenAPI Specification 3.1.0](#)

Приложение А

[Исходный код программы](#)