Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждение высшего образования
«**СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ**»

Гуманитарный институт
Кафедра информационных технологий
в креативных и культурных индустриях

УТВЕРЖДАЮ
И. о. заведующего кафедрой

_____ М. А. Лаптева

«_____» _____ 2023 г.

БАКАЛАВРСКАЯ РАБОТА

Интернет-аукцион Дома искусств Красноярского края.

Направление подготовки: 09.03.03 Прикладная информатика

Наименование программы: 09.03.03.30 Прикладная информатика

Руководитель доц., канд. техн. наук А. В. Усачёв

Выпускник А. В. Алла

Нормоконтролер И.Р. Нигматуллин

Красноярск 2023

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

С развитием технологий и интернета все больше организаций переходят на онлайн-формат работы, что в свою очередь требует разработки новых программных продуктов и сервисов. В этой связи актуальным является создание интернет-аукциона для Дома искусства Красноярского края, который позволит продавцам выставлять на продажу произведения искусства в онлайн-формате, а покупателям – приобретать их без необходимости посещения офлайн-аукциона.

Разработка интернет-аукциона для Дома искусства Красноярского края имеет большое значение для укрепления позиций организации на рынке искусства и расширения возможностей взаимодействия с клиентами и коллекционерами.

Проблемой, которую ставит перед собой данная работа, является отсутствие у Дома искусства Красноярского края собственного интернет-аукциона, что ограничивает возможности привлечения новых клиентов и взаимодействия с уже существующими.

Объектом исследования данной работы является разработка интернет-аукциона для Дома искусства Красноярского края.

Предметом исследования является создание электронной площадки для проведения аукционов, которая будет удобна и доступна для использования коллекционерами и клиентами Дома искусства.

Тема данного дипломного проекта имеет некоторую научную проработанность, однако большинство исследований, посвященных интернет-аукционам, фокусируются на анализе рынка и прогнозировании его развития. Разработка конкретного интернет-аукциона для организации изучена гораздо меньше.

Цель данного дипломного проекта – разработать интернет-аукцион для Дома искусства Красноярского края, который позволит расширить возможности организации взаимодействия с коллекционерами и клиентами, а также привлечь новых пользователей.

Для достижения поставленной цели были сформулированы следующие задачи:

– изучение существующих интернет-аукционов и анализ их особенностей;

– исследование предметной обрасти;

– проектирование интерфейса front-end части приложения;

– разработка электронной площадки для проведения интернет-аукционов.

Структура дипломной работы включает в себя введение, теоретическую часть, описание процесса разработки.

ГЛАВА 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

§1. Дом искусств

Дом искусств - это учреждение, предназначенное для продвижения, выставки и сохранения искусства. Оно может включать в себя галереи, музеи, выставочные залы, концертные залы, театры и другие подобные учреждения. Дом искусств является местом, где искусство становится доступным для общественности и может быть использовано для образовательных и культурных целей. Во многих случаях Дом искусств является культурным центром, который содействует развитию искусства в регионе и в стране в целом. Он может также организовывать мероприятия, выставки и концерты, а также предоставлять пространство для проведения мероприятий и занятий, связанных с искусством.

Первые Дома искусств появились в конце XIX века в Европе и Северной Америке в связи с растущим интересом к искусству и его распространению среди населения. Одним из первых Домов искусств был созданный в 1877 году в Лондоне "Дом искусств", основной целью которого было увеличение интереса к современному искусству и расширение его аудитории.

Затем подобные учреждения начали появляться в других городах мира, включая Париж, Нью-Йорк, Москву и другие крупные города. Например, в 1900 году в Париже был создан "Дом искусств", в котором были выставлены произведения искусства разных эпох и культур, а также проводились лекции, концерты и другие мероприятия.

В России первый Дом искусств был открыт в Москве в 1898 году при поддержке Третьяковской галереи и Московского художественного театра. В нем проводились выставки работ российских художников и зарубежных мастеров, а также лекции и мастер-классы.

Во второй половине XX века Дома искусств стали появляться в разных странах мира, часто в качестве крупных культурных комплексов,

включающих в себя не только музеи, но и театры, концертные залы и другие учреждения. Сегодня Дома искусств продолжают существовать и служить важными центрами культуры и искусства во всем мире.

Коллекции и экспозиции Дома искусств могут включать в себя произведения искусства разных эпох и культур, представляющие разные формы и жанры, такие как живопись, скульптура, фотография, дизайн, архитектура, театральное и музыкальное искусство и другие.

В зависимости от специализации и ориентации конкретного Дома искусств, его коллекции могут быть узкоспециализированными или охватывать широкий диапазон тематик и стилей. Они могут содержать как классические, известные произведения искусства, так и современные и экспериментальные работы.

Экспозиции Дома искусств могут быть как постоянными, так и временными. Постоянные экспозиции представляют собой постоянно действующие выставки произведений искусства из коллекции Дома искусств. Такие экспозиции могут быть устроены по хронологическому или тематическому принципу, и охватывать определенный период или стиль в истории искусства.

Временные экспозиции, в свою очередь, являются временными выставками, которые организуются для демонстрации конкретных тематик, стилей или авторов, и проводятся на определенный период времени. Они могут содержать произведения искусства из других музеев и коллекций, а также специально созданные произведения.

В целом, коллекции и экспозиции Дома искусств являются важным компонентом его работы, позволяющим представлять искусство широкой аудитории и сохранять его на протяжении длительного времени.

Дом искусств, как крупный культурный центр, играет важную роль в развитии культуры и искусства. Он предоставляет возможности для творческого самовыражения, обучения и развития, а также популяризации различных видов искусства.

Во-первых, Дом искусств является местом проведения различных мероприятий, таких как выставки, концерты, театральные представления и другие события. Они позволяют широкой аудитории познакомиться с различными видами искусства, расширить свой кругозор и получить эстетическое удовольствие.

Во-вторых, Дом искусств становится площадкой для творческой деятельности молодых талантов и профессионалов. Он предоставляет возможности для обучения и совершенствования мастерства в различных областях искусства. Также Дом искусств часто является организатором конкурсов и фестивалей, которые способствуют развитию творческого потенциала молодых людей и созданию новых произведений искусства.

Наконец, Дом искусств играет важную роль в сохранении культурного наследия и истории. Он является местом, где хранятся и экспонируются ценные коллекции произведений искусства и исторические артефакты. Благодаря этому они становятся доступными для широкой публики, что позволяет сохранить и передать наследие прошлого поколениям.

Таким образом, Дом искусств играет важную роль в развитии культуры и искусства, оказывая влияние на формирование культурных ценностей и творческого потенциала общества.

Примеры известных Домов искусств по всему миру:

Эрмитаж - национальный музей России в Санкт-Петербурге, включающий в себя пять зданий, в том числе Зимний дворец. В коллекции музея более 3 миллионов произведений искусства и культуры, включая работы изобразительного искусства, античные скульптуры, предметы декоративно-прикладного искусства, книги, манускрипты и документы.

Лувр - национальный музей Франции в Париже, один из крупнейших и самых посещаемых музеев мира. Коллекция Лувра включает более 460 000 произведений искусства, начиная от древности до XIX века, включая изобразительное искусство, скульптуру, предметы декоративно-прикладного искусства и археологические находки.

Национальный музей Токио - музей в Токио, Япония, основанный в 1872 году. Это один из крупнейших и наиболее значимых музеев в Японии. Его коллекция включает в себя более 110 000 произведений искусства, включая японские искусства, азиатские искусства, археологические находки, предметы декоративно-прикладного искусства и многое другое.

Музей искусств Метрополитен - музей в Нью-Йорке, США, один из самых крупных музеев в мире. Его коллекция включает более 2 миллионов произведений искусства, включая изобразительное искусство, скульптуру, предметы декоративно-прикладного искусства, костюмы, музыкальные инструменты и многое другое.

Галерея Уффици - музей во Флоренции, Италия, один из наиболее известных музеев мира. Коллекция галереи включает произведения искусства от эпохи Ренессанса до XVIII века, включая работы таких художников, как Леонардо да Винчи.

Дом искусств Красноярского края - это крупнейший культурный центр в Сибири, находящийся в городе Красноярске. Он был основан в 1967 году и с тех пор стал важным местом для проведения культурных мероприятий, выставок и концертов.

Коллекции и экспозиции Дома искусств Красноярского края включают в себя произведения изобразительного искусства, фотографии, скульптуры, предметы декоративно-прикладного искусства, а также исторические и археологические экспонаты. На территории Дома искусств располагается несколько выставочных залов, концертных залов, кинозалов и музыкальных студий.

Одной из самых известных экспозиций Дома искусств Красноярского края является коллекция картин известного сибирского художника Виктора Бутко. Также в Доме искусств регулярно проводятся выставки и концерты различных художников и музыкантов из Красноярска и других регионов России.

Будущее Дома искусств зависит от многих факторов, таких как экономические условия, технологические достижения, социокультурные тенденции и другие. Однако, несмотря на изменения внешних условий, Дом искусств остается важным центром культуры и искусства, который будет развиваться и адаптироваться к новым условиям.

Одним из направлений развития Дома искусств является использование новых технологий. В настоящее время все большее количество выставок и мероприятий проводится в онлайн-формате, что позволяет расширить аудиторию и сделать искусство более доступным. Также развитие виртуальной реальности, интерактивных установок и других инновационных технологий может привести к созданию новых форм искусства и разнообразить экспозиции Дома искусств.

§2. Аукцион

2.1 Определение понятия аукцион

Аукцион - это процесс продажи товаров, услуг или имущества, в котором участвуют покупатели, которые конкурируют между собой, предлагая наивысшую цену за лот. Аукцион может проводиться в режиме реального времени, когда все участники находятся в одном месте, или в формате онлайн-аукциона, когда участники участвуют удаленно через интернет. Цель проведения аукциона - получить наибольшую возможную цену за продаваемый лот. Аукцион может быть использован для продажи различных товаров, включая антиквариат, искусство, недвижимость, автомобили и многие другие. Кроме того, аукционы могут использоваться для благотворительных целей, чтобы собрать средства на различные благотворительные проекты.

2.2 Виды аукционов

Существует несколько видов аукционов, которые могут использоваться для продажи товаров, услуг или имущества:

– Английский аукцион (English Auction) – это наиболее распространенный вид аукциона, при котором продавец начинает торги с низкой цены и постепенно повышает ее. Участники торгуются друг с другом, предлагая все более высокие цены до тех пор, пока не будет достигнута окончательная цена, и лот не будет продан высшему предложившему;

– Голландский аукцион (Dutch Auction) – это процесс продажи, при котором продавец начинает с высокой цены и постепенно ее снижает до тех пор, пока кто-то не примет предложенную цену. Этот тип аукциона чаще используется для продажи группы товаров, а не для продажи отдельного лота;

– аукцион с секретным подачей предложений (Sealed Bid Auction) – это процесс продажи, при котором участники торгов подают свои предложения в запечатанных конвертах, и продавец выбирает победителя на основе наивысшей предложенной цены. Этот тип аукциона часто используется для продажи недвижимости или компаний;

– аукцион с обратным отсчетом (Reverse Auction) – это процесс продажи, при котором продавец начинает с высокой цены, а покупатели постепенно снижают цену, которую они готовы заплатить. В конечном итоге, продавец принимает наивысшее предложение, которое он получил. Этот тип аукциона часто используется для продажи товаров и услуг в бизнес-секторе;

– аукцион с динамической ценой (Dynamic Pricing Auction) – это процесс продажи, при котором цена меняется со временем в зависимости от спроса на товар или услугу. Например, на онлайн-аукционе, цена на лот может повышаться, когда количество желающих его купить увеличивается, и наоборот, цена может снижаться, если на лот нет спроса.

Каждый вид аукциона имеет свои особенности, которые могут быть использованы продавцами в зависимости от специфики продаваемого товара, услуги или имущества.

2.3 Особенности интернет-аукционов

Интернет-аукционы – это аукционы, которые проводятся онлайн через интернет. Они имеют ряд особенностей, которые отличают их от традиционных аукционов:

Глобальность: Интернет-аукционы позволяют продавцам продавать свои товары по всему миру, а покупателям приобретать товары из любой точки мира. Это значительно расширяет рынок, на котором продавцы могут предлагать свои товары, и дает покупателям больше возможностей для покупки интересующих их товаров.

Удобство: Интернет-аукционы доступны 24 часа в сутки, 7 дней в неделю, что позволяет участникам торгов участвовать в них в любое время и из любого места. Также нет необходимости лично присутствовать на аукционе, что экономит время и снижает затраты на поездки и проживание.

Прозрачность: Интернет-аукционы обычно обеспечивают высокий уровень прозрачности, поскольку они открыты для всех желающих. Участники могут легко увидеть текущую цену товара, количество ставок и другие данные, что обеспечивает честную конкуренцию между участниками торгов.

Большое количество лотов: Интернет-аукционы могут предложить на продажу большое количество лотов за короткое время, что дает продавцам возможность продать свои товары быстрее и снизить затраты на хранение и управление запасами.

Низкие затраты: Интернет-аукционы требуют намного меньше затрат на организацию, чем традиционные аукционы. Например, нет необходимости арендовать зал, нанимать персонал, оплачивать коммунальные услуги и

другие затраты, что снижает стоимость проведения аукциона и может снизить цену продаваемого товара.

Однако, необходимо учитывать, что у интернет-аукционов есть и риски, такие как возможность мошенничества, несоответствие товара описанию, задержка в доставке товара и другие. Поэтому, необходимо быть осторожны.

§3. Понятие web-сайта

Web-сайт – это совокупность веб-страниц, связанных друг с другом гиперссылками и обычно размещенных на одном домене. Каждая веб-страница содержит текст, изображения, аудио и/или видео-контент, который может быть доступен пользователям интернета через браузер. Web-сайты используются для предоставления информации, продажи товаров и услуг, обмена информацией и коммуникации с пользователем. Сайты могут быть созданы как для личных, так и для коммерческих целей и могут быть доступны как для публичного, так и для частного использования.

Web-сайт является основой для онлайн-присутствия большинства компаний, организаций и частных лиц. Он представляет собой цифровую платформу, которая позволяет пользователям обмениваться информацией и взаимодействовать друг с другом через Интернет.

Web-сайты могут быть статическими или динамическими. Статический сайт состоит из HTML-страниц, которые могут быть предварительно созданы и сохранены на сервере. Они не содержат динамического контента и не имеют возможности для обратной связи с пользователем. Динамические сайты, с другой стороны, генерируют контент при запросе пользователя. Они обычно используют базы данных, чтобы хранить информацию и скрипты для обработки пользовательского ввода.

Web-сайты могут служить различным целям, включая предоставление информации, продажу товаров и услуг, коммуникацию и социализацию.

Сайты-блоги, например, могут использоваться для публикации статей и новостей, в то время как сайты электронной коммерции могут предоставлять пользователю возможность покупки товаров и услуг онлайн. Web-сайты также могут быть использованы для обмена информацией, включая научные исследования, новости, медиа-контент и другие формы информации.

Web-сайты могут быть доступны как для общественного использования, так и для ограниченного круга лиц. Например, корпоративный сайт может быть доступен только для сотрудников компании, в то время как сайт-блог может быть доступен для всех пользователей Интернета. Web-сайты также могут использоваться для коммуникации с пользователем, такой как обратная связь и формы обратной связи, которые позволяют пользователям связаться с владельцами сайта.

§4. Анализ существующих решений

Существует множество Интернет-аукционов на российском рынке, каждый из которых имеет свои особенности и преимущества. Однако, несмотря на большое количество предложений, не все Интернет-аукционы предоставляют высококачественный сервис и соответствуют ожиданиям пользователей. В связи с этим возникает необходимость проанализировать существующие решения Интернет-аукционов в России, выявить их достоинства и недостатки, а также предложить рекомендации по улучшению их работы.

4.1 Artinvestment.ru

Artinvestment.ru: это интернет-аукцион, специализирующийся на продаже произведений русского искусства XIX-XX веков. На сайте можно найти работы художников таких как Илья Репин, Василий Суриков, Валентин Серов и других.

Плюсы:

– специализация на русском искусстве XIX-XX веков, что может привлечь коллекционеров и любителей этого периода;

– хорошая навигация по сайту, что делает его удобным в использовании;

– наличие профессиональных экспертов и оценщиков, что гарантирует подлинность продаваемых на аукционе произведений искусства.

Минусы:

– высокие цены на произведения искусства, что может отпугнуть многих покупателей;

– небольшой выбор лотов в сравнении с другими аукционами;

– сложный, перегруженный интерфейс.

Сайт показан на рисунке 1.

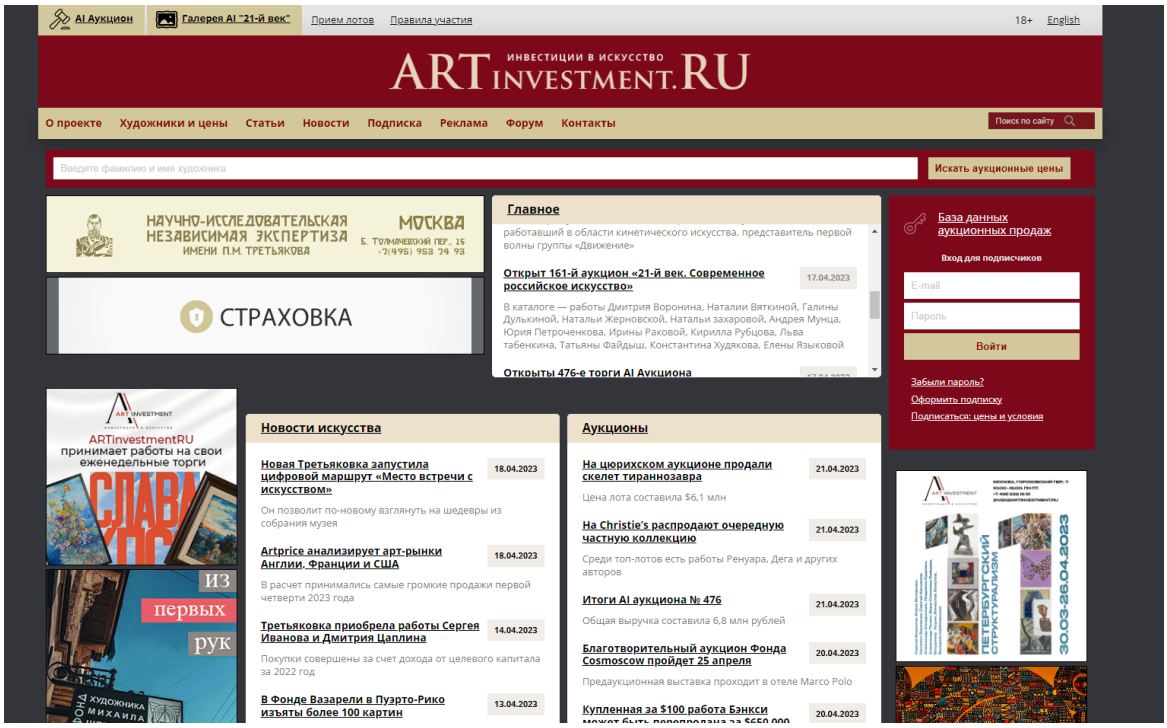

Рисунок 1. Главная страница сайта artinvestment.ru

4.2 artzip.ru

Еще один web-сайт с похожими критериями – “Artzip”. Он также как и первый сайт предоставляет услуги продажи предметов искусства. Рассмотрим плюсы и минусы этого сайта:

Плюсы:

– широкий ассортимент произведений искусства и антиквариата, включая живопись, графику, скульптуру, предметы декора и мебель;

– высокое качество представленных лотов, проверка подлинности и аутентичности произведений перед продажей;

– удобный и интуитивно понятный интерфейс сайта, легкий доступ к информации о лотах и процедуре покупки;

– наличие раздела с информацией о прошлых аукционах и проданных лотах, что позволяет ознакомиться с предыдущим опытом продаж на сайте;

– возможность онлайн-участия в торгах, удобный вариант для покупателей, находящихся в другом регионе или стране;

– онлайн поддержка для покупателя

Минусы:

– высокая конкуренция за лоты, что может привести к завышению цен и ограничению доступности некоторых произведений для покупки.

Сайт показан на рисунке 2.

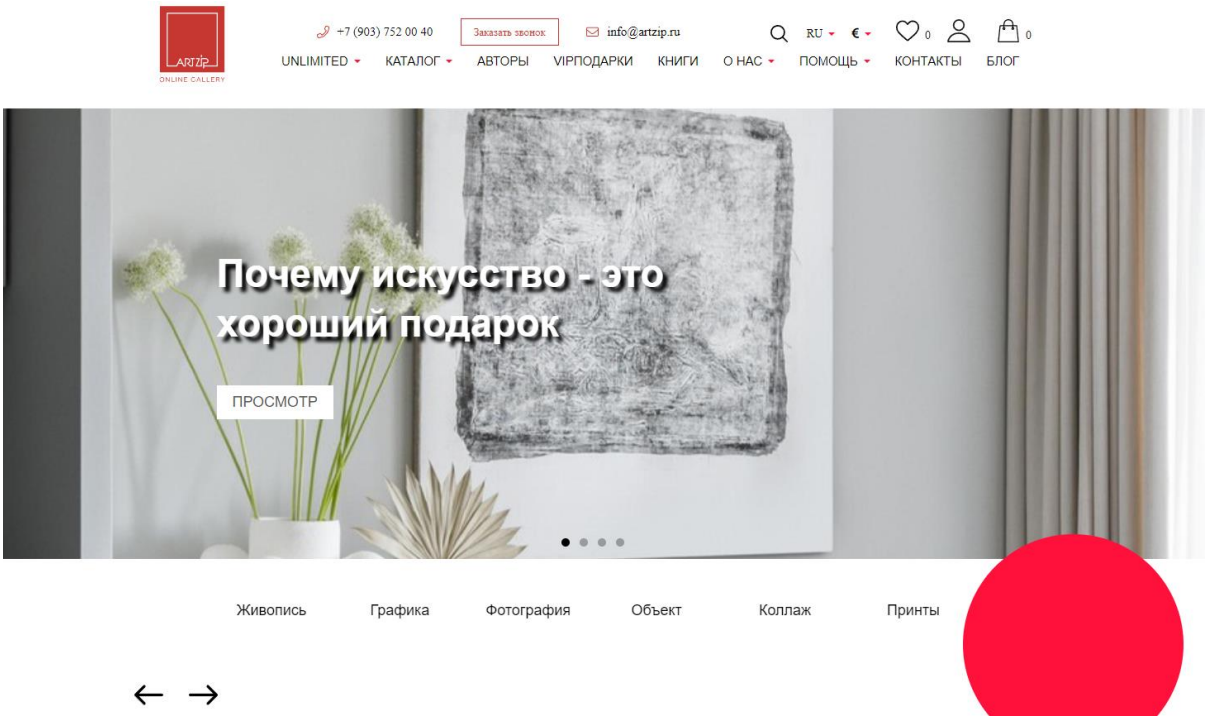

Рисунок 2. Главная страница сайта artzip.ru

Как таковых минусов я больше не увидел, это хороший сайт, который может послужить примером для создания собственного.

Исходя из всего вышесказанного можно сделать вывод, что для создания хорошего интернет-аукциона, нужен продуманный функционал, с большим количеством возможностей и при этом интуитивно понятный интерфейс, чтобы каждый посетитель мог в нем разобраться.

Итоги главы

Целью раздела являлось изучение предметной области, в результате которого были собраны основные требования к разрабатываемому веб-приложению.

В этой главе были рассмотрены основные аспекты, которые необходимо учесть при создании Интернет-аукциона. Анализ проведенных исследований показал, что создание Интернет-аукциона является актуальной и перспективной задачей, поскольку он позволяет осуществлять покупку и продажу товаров и услуг в Интернете, что является удобным и эффективным способом коммерческой деятельности.

В главе были проанализированы различные типы аукционов, преимущества и недостатки каждого типа, а также основные требования, которые должны быть учтены при создании Интернет-аукциона.

Был проведен сравнительный анализ существующих Интернет-аукционов, оценив их преимущества и недостатки. Это позволило выделить наиболее важные функции и особенности, которые должны быть учтены при создании нового Интернет-аукциона.

ГЛАВА 2. ПРОЕКТИРОВАНИЕ

§1. Архитектура информационной системы

Архитектура информационной системы Веб-приложения - это структура и организация компонентов, которые составляют приложение и обеспечивают его функциональность. В общем случае, архитектура информационной системы Веб-приложения включает в себя следующие компоненты:

– клиентская часть (Front-end) — это интерфейс пользователя, который представляет веб-страницы, графические элементы, визуальные эффекты и другие элементы, которые пользователь видит в своем браузере. Клиентская часть обычно разработана с использованием языков HTML, CSS и JavaScript;

– серверная часть (Back-end) — это компонент, который обрабатывает запросы пользователя, обеспечивает функциональность приложения и взаимодействует с базой данных. Серверная часть обычно разработана на одном из языков программирования, таких как PHP, Java, Python, Ruby и др.;

– база данных (Database) — это хранилище данных, которое содержит информацию, необходимую для работы приложения. База данных обеспечивает хранение и доступ к данным, которые используются приложением;

– протоколы и технологии — это набор стандартов и протоколов, которые используются для обеспечения взаимодействия между клиентской и серверной частями приложения. Например, для взаимодействия между клиентской и серверной частями может использоваться протокол HTTP;

– среда выполнения (Runtime Environment) — это программное обеспечение, которое используется для запуска приложения. Например, приложение может запускаться на сервере с использованием веб-сервера Apache или Nginx.

Все компоненты архитектуры информационной системы Веб-приложения взаимодействуют друг с другом, чтобы обеспечить функциональность приложения и удовлетворить потребности пользователей. Правильно разработанная архитектура позволяет обеспечить высокую производительность, масштабируемость, безопасность и удобство использования приложения.

§2. Типы архитектуры веб-приложений

Для правильного распределения логики приложения между серверной частью и клиентской, можно выбрать подходящую архитектуру. Наиболее распространенные и часто используемые:

– одностраничные приложения (SPA);

– многостраничные приложения (MPA);

– прогрессивные веб-приложения (PWA);

– архитектура микросервисов.

Разберем каждый вид в деталях.

2.1 Одностраничные приложения

Одностраничные приложения (Single-Page Applications, SPA) — это веб-приложения, которые загружаются и выполняются на одной странице в браузере пользователя, обновляя содержимое страницы динамически без перезагрузки всей страницы.

SPA строятся на основе клиент-серверной архитектуры, где весь функционал приложения работает на стороне клиента, а сервер предоставляет только API для обмена данными. В SPA используется множество технологий, включая JavaScript-фреймворки и библиотеки (например, Angular, React, Vue), AJAX и JSON для обмена данными между клиентом и сервером, HTML5 и CSS3 для разметки и стилизации

интерфейса, а также множество других инструментов для обеспечения высокой производительности и удобства использования.

Одним из главных преимуществ SPA является повышение пользовательского опыта и скорости работы приложения за счет сокращения времени на загрузку страницы и перерисовку содержимого. Также SPA обеспечивают более простую разработку и поддержку, упрощая процесс обновления и расширения функционала приложения.

Один из примеров SPA-приложений — это Gmail от Google. Пользователь может отправлять и получать письма, создавать документы, работать с календарем и другими приложениями, не покидая одну страницу. Приложение загружается только один раз, а дальнейшая навигация в приложении происходит без перезагрузки страницы.

2.2 Многостраничные приложения

Многостраничные приложения (Multi-Page Applications, MPA) — это веб-приложения, в которых каждая страница является отдельным документом HTML, и приложение обычно работает по принципу запрос-ответ.

В MPA каждая страница загружается с сервера целиком при переходе пользователя на новую страницу, и все элементы на странице обновляются вместе с ней. В MPA сервер обычно отвечает за формирование и отправку HTML-страниц на клиентскую часть приложения, а клиентская часть может содержать скрипты на JavaScript, которые обрабатывают действия пользователя на странице и отправляют запросы на сервер для получения новых данных.

MPA обладают рядом преимуществ, таких как более простую разработку и поддержку, лучшую индексируемость поисковыми системами, а также более высокую безопасность в связи с тем, что клиент не получает

доступа к всей функциональности приложения, а только к запрашиваемым страницам.

Однако, MPA также имеют свои недостатки, такие как более медленную загрузку страницы из-за большего количества запросов на сервер, менее интерактивный пользовательский интерфейс, и более сложную навигацию пользователей по приложению, что может привести к ухудшению пользовательского опыта.

Примеры MPA включают в себя такие сайты, как Amazon, Wikipedia, и многие другие сайты, где каждая страница является отдельным документом HTML и обновляется при переходе пользователя на новую страницу.

2.3 Прогрессивные веб-приложения

Прогрессивные веб-приложения (Progressive Web Applications, PWA) — это веб-приложения, которые обладают функциональностью и возможностями, обычно свойственными нативным приложениям для мобильных устройств. PWA используют множество технологий, таких как service workers, Web App Manifest и push-уведомления, чтобы создать более быстрое, надежное и удобное в использовании приложение на веб-технологиях.

Service workers являются скриптами на JavaScript, которые работают отдельно от основного потока браузера и могут кэшировать данные и предоставлять функциональность офлайн. Web App Manifest позволяет определить значок приложения, заголовок и цвет фона приложения, а также другие характеристики, обычно связанные с нативными приложениями. Push-уведомления позволяют отправлять уведомления пользователям, даже когда приложение не активно.

Преимущества PWA включают в себя более высокую производительность и скорость работы, возможность работы в офлайн-режиме, более простую установку и более гладкую работу на мобильных

устройствах. PWA могут запускаться на любом устройстве, включая компьютеры, планшеты и мобильные телефоны, и обладают широкой совместимостью с различными браузерами.

Примеры PWA включают в себя такие приложения, как Twitter Lite, Pinterest, Alibaba, и многие другие.

2.4 Архитектура микросервисов

Архитектура микросервисов (Microservices Architecture) — это подход к разработке программного обеспечения, в котором приложение разбивается на небольшие и независимые сервисы, которые могут взаимодействовать друг с другом через API. Каждый сервис выполняет свою функцию и может быть разработан и масштабирован отдельно от других сервисов в системе.

В микросервисной архитектуре каждый сервис запущен в своем собственном контейнере, что обеспечивает независимость их работы. Каждый сервис может быть написан на разных языках программирования и использовать разные технологии, что позволяет использовать лучшие подходы для каждого конкретного сервиса.

Эта архитектура позволяет упростить процесс разработки и поддержки приложения, так как каждый сервис может быть разработан и тестирован независимо от других сервисов, а также позволяет быстрее масштабировать отдельные сервисы в случае необходимости.

Однако, микросервисная архитектура также имеет свои недостатки, такие как более сложную интеграцию и управление сервисами, увеличение нагрузки на сеть, необходимость более сложной настройки инфраструктуры, и увеличение сложности тестирования системы в целом.

Примерами компаний, использующих микросервисную архитектуру, являются Netflix, Amazon, Uber, и многие другие компании, где масштабирование и гибкость являются ключевыми требованиями к разрабатываемому приложению.

§3. API

3.1 Понятие API

API (Application Programming Interface) – это интерфейс программирования приложений, который позволяет программам и приложениям взаимодействовать друг с другом. API определяет набор протоколов, правил и инструментов, которые используются для создания приложений и программных компонентов.

API определяет способы, как программа может обращаться к другой программе или компоненту, получать от них данные и передавать им данные. Это может быть выполнено с помощью стандартных протоколов, таких как HTTP, HTTPS, XML, JSON и др.

API могут быть разработаны для взаимодействия между компьютерными системами, мобильными приложениями, веб-сервисами, базами данных и др.

API могут быть открытыми или закрытыми. Открытые API могут быть доступны для общего использования, и разработчики могут использовать их для создания своих собственных приложений. Закрытые API доступны только для определенных пользователей или организаций, которые могут использовать их для своих конкретных нужд.

API являются ключевым элементом взаимодействия между различными компонентами программного обеспечения и позволяют программам взаимодействовать друг с другом, обмениваться данными и выполнять различные задачи.

3.2 Взаимодействие с API

Взаимодействие с API может происходить разными способами в зависимости от используемой технологии и протокола. В общем случае процесс взаимодействия с API выглядит следующим образом:

– Получение доступа к API: для начала работы с API необходимо получить доступ к его документации и ключу API, который будет использоваться для аутентификации при запросах к API;

– Формирование запроса: после получения ключа API необходимо сформировать запрос к API, указав необходимые параметры, методы и форматы передачи данных;

– Отправка запроса: после формирования запроса его необходимо отправить на сервер, на котором расположен API;

– Обработка ответа: после отправки запроса сервер возвращает ответ, который необходимо обработать. Обычно ответ в формате JSON или XML и содержит запрошенные данные или информацию об ошибке;

– Использование полученных данных: после обработки ответа данные могут быть использованы в приложении или программе, которые работают с API.

Взаимодействие с API может происходить через различные протоколы, такие как HTTP, REST, SOAP, GraphQL и др. Различные API могут предоставлять различный функционал и методы для работы с данными, например, получение данных, создание новых объектов, обновление или удаление данных.

§4. Выбор средств front-end разработки для интернет-аукциона

4.1 HTML

HTML (HyperText Markup Language) – это язык разметки гипертекста, который используется для создания веб-страниц. Он представляет собой набор элементов, которые позволяют описывать содержимое веб-страницы, такое как текст, изображения, ссылки, таблицы, формы и другие элементы.

HTML-код представляет собой текстовый документ, который содержит теги и атрибуты для определения структуры и содержимого веб-страницы.

Теги определяют различные элементы, такие как заголовки, абзацы, изображения, таблицы и др., а атрибуты предоставляют дополнительные сведения о том, как отображать содержимое.

HTML-документы можно создавать с помощью текстовых редакторов, таких как Notepad++, Sublime Text, Visual Studio Code и других, а также с помощью специализированных программ, таких как Adobe Dreamweaver.

HTML-код может включать другие технологии, такие как CSS (Cascading Style Sheets) для определения стиля и внешнего вида веб-страниц, а также JavaScript для добавления интерактивности и динамического поведения веб-страниц.[10]

4.2 CSS

CSS (Cascading Style Sheets) — это язык описания стилей, который используется для оформления веб-страниц. Он позволяет определять внешний вид элементов HTML-документа, таких как цвет фона, шрифты, размеры, положение, анимации и др.

CSS работает в сочетании с HTML и определяет стиль элементов, используя селекторы, свойства и значения. Селекторы определяют элементы, для которых задаются стили, свойства определяют конкретные стили, а значения определяют значение свойства для конкретного элемента.

CSS-код может быть определен в отдельных файлах, которые загружаются вместе с HTML-документом, или внедрен непосредственно в HTML-документ с помощью тега <style>. CSS-код может быть также определен в стилевых таблицах, которые применяются к нескольким HTML-документам, чтобы обеспечить единый стиль для всех веб-страниц.

CSS-код может быть создан с помощью текстовых редакторов, таких как Notepad++, Sublime Text, Visual Studio Code и других. Он может включать различные конструкции, такие как селекторы, комбинаторы,

псевдоклассы и псевдоэлементы, которые позволяют создавать разнообразные стили для элементов веб-страницы.[11]

4.3 Tailwind CSS

Tailwind CSS – это CSS-фреймворк, который предоставляет набор готовых компонентов и утилит, которые упрощают и ускоряют процесс разработки пользовательских интерфейсов для веб-приложений.

Основным принципом Tailwind CSS является подход utility-first, который заключается в том, чтобы использовать маленькие классы для определения стилей элементов, вместо создания больших CSS-файлов. Таким образом, разработчик может быстро и легко настроить внешний вид элементов и компонентов на странице, используя лишь несколько классов.

В Tailwind CSS предоставляется большой набор утилит, которые можно использовать для установки размеров, цветов, отступов, границ и многого другого. Также фреймворк предлагает множество готовых компонентов, таких как кнопки, формы, таблицы и т.д.

В целом, использование Tailwind CSS позволяет ускорить и упростить разработку пользовательских интерфейсов, а также обеспечить их согласованность и доступность.

4.4 ReactJs

React (React.js) – это библиотека JavaScript для разработки пользовательских интерфейсов (UI). Она была разработана компанией Facebook и стала одной из наиболее популярных библиотек для создания веб-приложений.

React использует компонентный подход для создания пользовательского интерфейса. Компоненты React – это многократно используемые кирпичики, которые можно объединять в более крупные

компоненты для создания интерфейса. Каждый компонент имеет свою логику и свойства, которые позволяют ему работать с другими компонентами и обрабатывать события.

React использует Virtual DOM (виртуальный DOM), чтобы обновлять только те элементы, которые действительно изменились. Virtual DOM – это виртуальное представление реального DOM, которое React использует для сравнения текущего состояния страницы с предыдущим. Это позволяет ускорить процесс обновления страницы и сделать его более эффективным.

React поддерживает множество инструментов и библиотек, которые помогают разработчикам создавать более сложные приложения. Например, React Router используется для управления маршрутизацией, а Redux – для управления состоянием приложения.

React является открытым исходным кодом, и его исходный код доступен на GitHub. Он также имеет большое сообщество разработчиков, которые создают и поддерживают множество библиотек и инструментов для улучшения работы с React.[5]

4.5 TypeScript

TypeScript – это язык программирования, который является надстройкой над JavaScript. Он добавляет статическую типизацию, классы, интерфейсы, перечисления, модули и другие функции, которые не были доступны в JavaScript до появления стандарта ES6.

TypeScript разработан компанией Microsoft и имеет целью улучшить разработку крупных приложений JavaScript, сделать их более поддерживаемыми, улучшить производительность и уменьшить количество ошибок.

Основным преимуществом TypeScript является статическая типизация. Это означает, что типы данных определяются во время написания кода и проверяются компилятором TypeScript перед запуском приложения. Это

позволяет обнаруживать ошибки до того, как код будет запущен, а также упрощает чтение и понимание кода.

TypeScript также позволяет использовать классы и интерфейсы, которые упрощают создание сложных объектов и обеспечивают более четкую организацию кода. Он также поддерживает модули, которые помогают организовывать код в более мелкие блоки и упрощают повторное использование кода.

TypeScript может быть скомпилирован в обычный JavaScript, что позволяет использовать его в любом современном браузере и на любой платформе. Он также поддерживается множеством инструментов разработки, включая Visual Studio Code, WebStorm и другие.

TypeScript стал очень популярным среди разработчиков, которые работают над крупными проектами JavaScript, так как он позволяет создавать более надежный и поддерживаемый код, который легче масштабировать и изменять.

4.6 FireBase

Firebase – это платформа облачных сервисов, разработанная компанией Google, которая позволяет разработчикам быстро и легко создавать мобильные и веб-приложения. Она включает в себя различные сервисы и инструменты, которые помогают ускорить разработку, улучшить производительность и обеспечить безопасность приложений.

Firebase предоставляет множество сервисов для создания и управления приложениями, включая базы данных, аутентификацию, хранение файлов, отправку уведомлений и другие. Один из самых популярных сервисов Firebase – это Realtime Database, которая позволяет быстро создавать приложения с реальным временем обновления данных.

Firebase также имеет мощные инструменты для аналитики приложений, которые позволяют отслеживать использование приложения и поведение

пользователей, а также принимать меры по улучшению опыта пользователей и увеличению доходов.

Firebase позволяет легко интегрироваться с другими сервисами Google, такими как Google Cloud Platform, Google Ads и Google Analytics. Это делает его идеальным выбором для разработчиков, которые хотят быстро создавать и управлять приложениями в облаке.

Firebase имеет мощное API и SDK, которые позволяют разработчикам легко интегрировать сервисы Firebase в свои приложения, а также обеспечивают хорошую документацию и поддержку сообщества. Firebase поддерживается на всех основных платформах, включая iOS, Android, WEB и Unity.

4.7 Visual Studio Code

Visual Studio Code – это бесплатный редактор кода, разработанный компанией Microsoft. Он предоставляет множество инструментов для разработки приложений, включая подсветку синтаксиса, автодополнение, отладку кода, контроль версий, интеграцию с Git и многие другие.

Visual Studio Code предлагает множество расширений, которые позволяют улучшить функциональность редактора и добавить поддержку для различных языков программирования и фреймворков. Он также интегрируется с другими инструментами и сервисами разработки, такими как Azure, Docker, TypeScript и другие.

Visual Studio Code является платформонезависимым и работает на Windows, macOS и Linux. Он имеет простой и интуитивно понятный интерфейс, что делает его доступным для широкого круга пользователей. Кроме того, он поддерживается большим сообществом разработчиков, что гарантирует быструю и качественную поддержку и обновления.

§5. Методология разработки БЭМ

БЭМ (Блок, Элемент, Модификатор) – это методология и набор инструментов для разработки пользовательских интерфейсов веб-приложений, которые позволяют упростить процесс разработки, улучшить согласованность и уменьшить время на поддержку проекта.

Основной принцип БЭМ заключается в том, что каждый элемент на странице разбивается на отдельные блоки, которые имеют определенные свойства и методы взаимодействия. Эти блоки могут включать в себя дополнительные элементы и модификаторы, которые могут менять их внешний вид и поведение.

Каждый блок имеет уникальное имя, которое обычно отображается в соответствующем классе в HTML-коде. Элементы блока отображаются внутри соответствующего блока и обычно имеют иерархические имена, которые отражают их отношения с блоком. Модификаторы позволяют изменять внешний вид и поведение блока и элементов.

Использование БЭМ позволяет уменьшить количество CSS-кода, улучшить согласованность и упростить поддержку веб-приложения. Также БЭМ является открытым проектом с большим и активным сообществом разработчиков, которые создают новые инструменты и решения для улучшения процесса разработки интерфейсов.

Методология БЭМ была разработана командой Яндекса, российской компании, занимающейся разработкой интернет-сервисов и продуктов. Она была создана в ответ на проблемы, связанные с разработкой крупных и сложных веб-приложений, когда множество разработчиков работают над одним проектом и необходимо обеспечить единообразие в коде и интерфейсе. Официальный сайт БЭМ сообщает, что идея БЭМ пришла из другой компании - Яндекс.Маркет, где в 2005 году Андрей Ярончик и Сергей Бережной внедрили первые принципы методологии в свои проекты. В

дальнейшем, в 2008 году, команда разработчиков в Яндексе систематизировала и уточнила принципы методологии и назвала ее БЭМ.

§6. Интерфейс веб-приложений

Сейчас успех веб-сайтов в значительной степени зависит от качества его интерфейса. Хорошо спроектированный интерфейс должен быть простым для использования, с ключевыми функциями, доступными пользователю, а несущественные должны быть скрыты. Эффективное размещение элементов управления повышает удобство работы с веб-сайтом, а внешний вид интерфейса играет решающую роль в первом впечатлении и удовольствии от использования. При отсутствии проблем с определением целевой аудитории можно выделить несколько ключевых требований к интерфейсу:

– лаконичность;

– простота;

– отсутствие рекламы;

– возможность адаптации под различные экраны.

В ходе разработки интерфейса веб-приложения создаются макеты основных страниц, которые затем используются для создания самого интерфейса. Так как проект будет масштабным, с большим количеством страниц, приведу макеты интерфейса только некоторых их них. Макеты страниц представлены на рисунках 3-5.

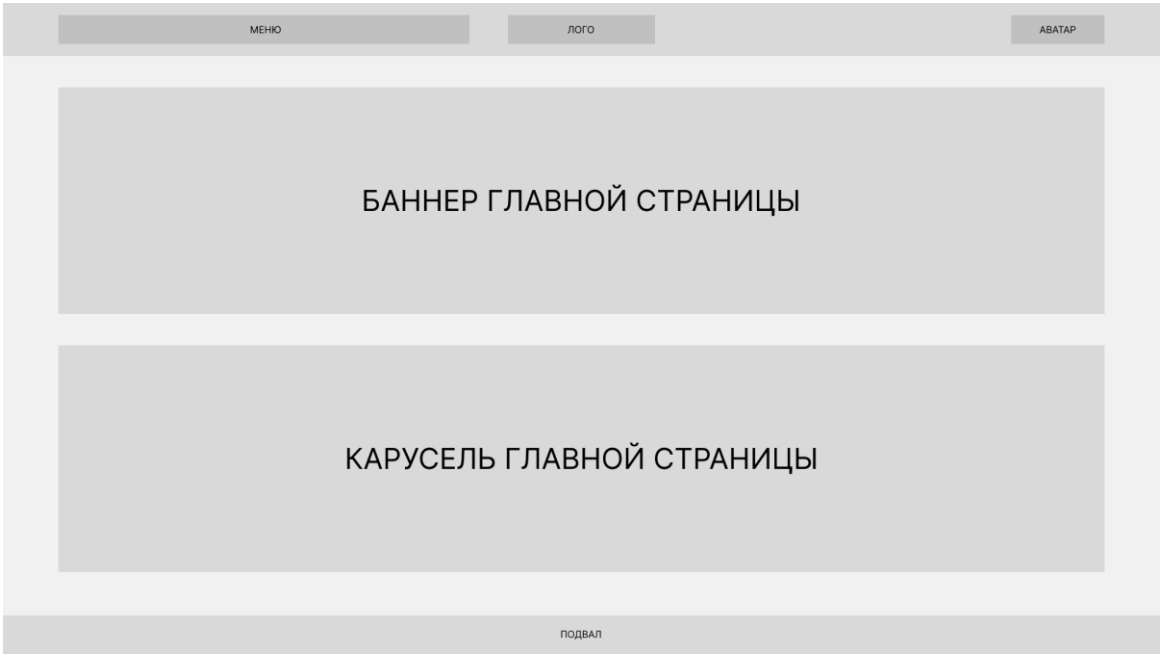

Рисунок 3 – Макет главной страницы

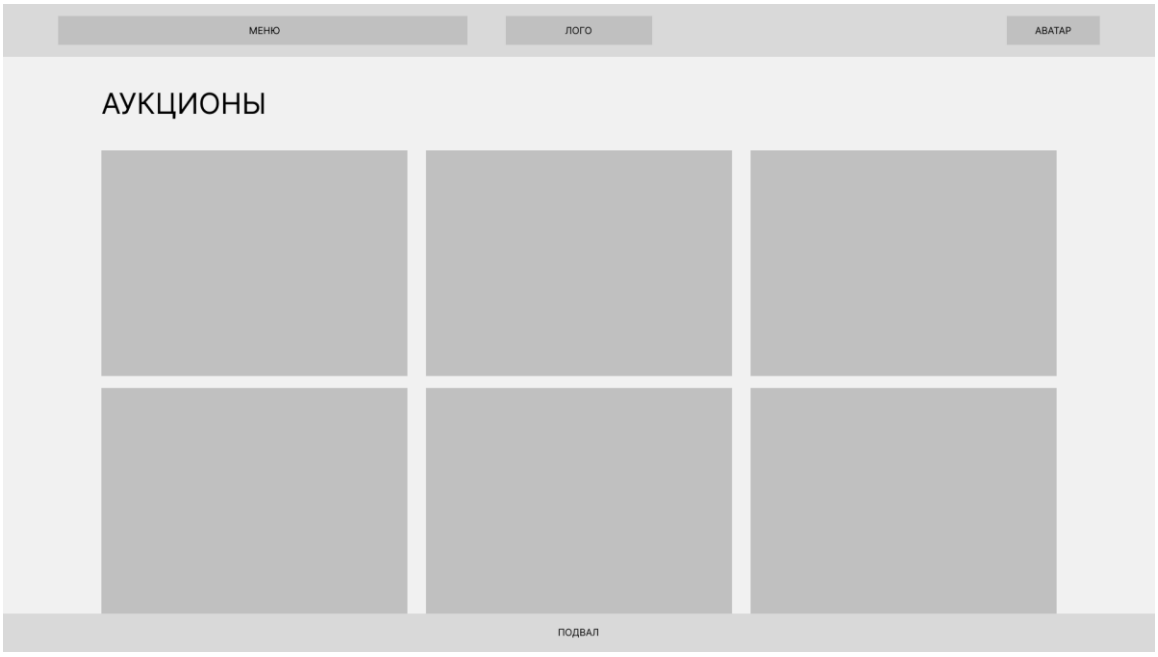

Рисунок 4 – Макет страницы с аукционами

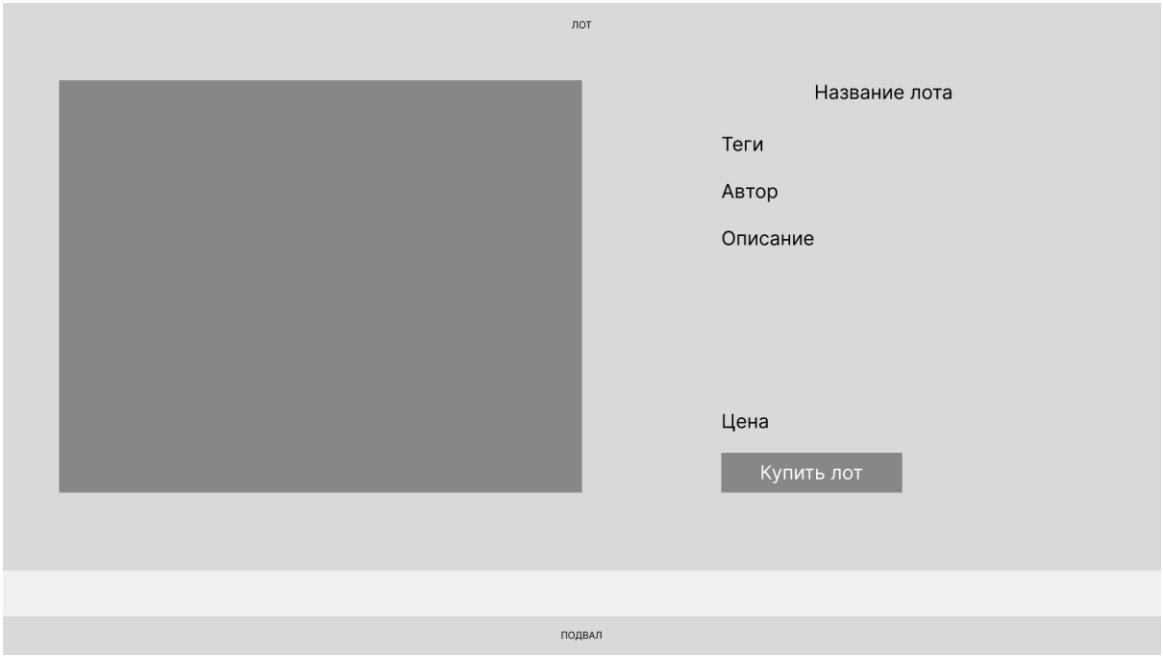

Рисунок 5 – Макет страницы “Просмотр лота”

Итоги главы

Целью раздела являлось описание проектирования разрабатываемой веб-платформы. Был разработан макет интерфейса, а также описаны архитектуры веб-приложений, представлены из особенности.

Были выбраны средства разработки для успешной реализации проекта.

– Сервер и база данных: FireBase;

– Языки программирования: JavaScript, ReactJS, TypeScript;

– Язык стилей: CSS, TailwindCSS;

– Язык разметки: HTML;

– Редактор кода: Visual Studio Code.

ГЛАВА 3. РЕАЛИЗАЦИЯ РАЗРАБОТАННОГО WEB-ПРИЛОЖЕНИЯ

§1. Front-end

Реализацию проекта необходимо начать с верстки главной страницы и наполнения её различными элементами интерфейса. После чего создать страницу регистрации/авторизации и так далее. Всё это нужно сделать в соответствии с разработанным интерфейсом и структурой.

Началом этапа разработки является определение файловой структуры проекта, представленной на рисунке 6.

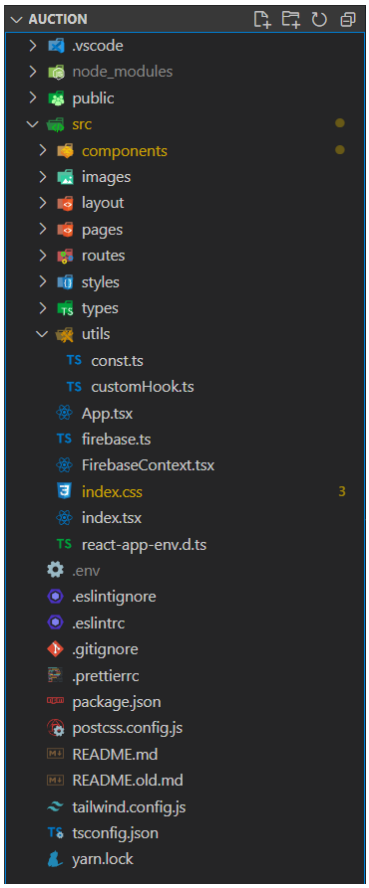

Рисунок 6 – Файловая структура проекта

Файлы с настройками проекта находятся в корневом каталоге.

Корневой каталог включает в себя:

– node_modules – папка, для хранения всех npm пакетов;

– public – содержит html-файл приложения;

– src – содержит все React компоненты и файлы со стилями.

Начальным этапом разработки front-end части проекта являлось создание корневых React компонентов для каждого раздела. В директории src была создана директория pages, содержащая главные компоненты всех разделов (рисунок 7):

– AuctionLots (страница с лотами);

– Auctions (страница с аукционами);

– CreateAuctions (страница создания аукциона);

– FavoritePage (страница с добавленными в избранное лотами);

– Home (главная страница);

– InfoBlock (страница с информацией);

– Login (страница логина);

– LotOverview (страница развернутого лота);

– PersonalArea (страница личного кабинета);

– Registration (страница регистрации).

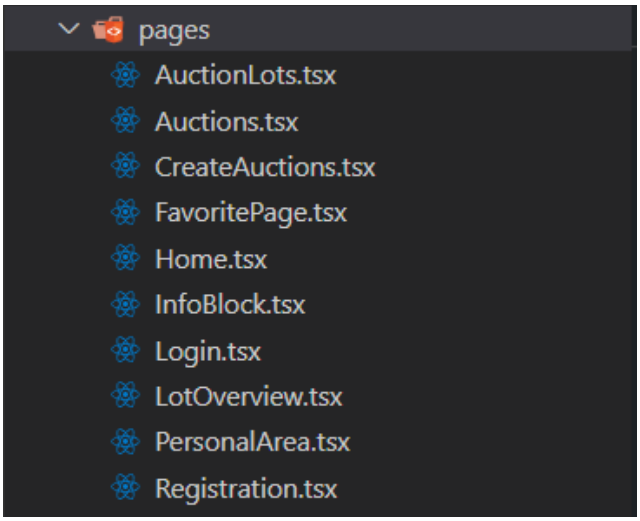

Рисунок 7 – Директория pages, с главными компонентами проекта

Промежуточные компоненты, используемые в главных, хранятся в директории components.

Пример React компонента представлен на рисунке 8. Компоненты CardProduct является карточкой лота.

```
const CardProduct: FC<ICardProductProps> = ({ image, title, descr, price, id, changeLot, digit, toogleFavorite }) => {
  const user = auth.currentUser;

  const [favoriteStyle, setFavotiteStyle] = useState({ color: '#363636' });
  const [favoriteBoolean, setFavoriteBoolean] = useState(false);

  const { data, isLoading, error } = useQuery('favorites', fetchDataFavoritesLots)
  async function fetchDataFavoritesLots() {
    const querySnapshot = await getDoc(doc(db, 'users', user!.uid))
    return querySnapshot.data()!.favoriteLots
  }

  useEffect(() => {
    if (data) {
      if (data.find((i: number) => i === id)) {
        setFavotiteStyle({ color: '#fb332c' })
        setFavoriteBoolean(true)
      } else {
        setFavotiteStyle({ color: '#363636' })
        setFavoriteBoolean(false)
      }
    }
  }, [data]);

  if (isLoading) return <div>Loading</div>

  if (error) return <div>Something went wrong...</div>

  if (data === undefined) return <div>Undefined</div>

  const handleClick = (id: number) => {
    if (toogleFavorite) {
      toogleFavorite(id)

      if (favoriteBoolean) {
        setFavotiteStyle({ color: '#363636' });
        setFavoriteBoolean(false)
      } else {
        setFavotiteStyle({ color: '#fb332c' });
        setFavoriteBoolean(true)
      }
    }
  }

  return (
    <div>
      <div className='auction__lots-cart'>
        <div id='triangle' className='triangle'></div>
        <div className="auction__lots-img-wrapper">
          <img className='auction__lots-img' src={image} alt="image" />
        </div>
        <div className='auction__lots-info'>
          <p className="auction__lots-number">{digit}</p>
          <p className="auction__lots-title">{title}</p>
          <p className="auction__lots-descr">{descr}</p>
          <p className="auction__lots-price">Цена: {price} руб.</p>
          <div className="product-links">
            {
              changeLot ? <span className=' block mt-[3px]'></span> :
                <i id='like' onClick={() => handleClick(id)} style={favoriteStyle} className="fa fa-heart pr-[10px]"></i>
            }
          </div>
        </div>
      </div>
    </div>
  )
}

export default CardProduct
```

Рисунок 8 – Компонент CardProduct

При разработке компонентов использовались React Hooks, которые появились в обновлении библиотеки 16.8. Разработка и с помощью «хуков» позволяет создавать проекты, состоящие только из функциональных компонентов, в отличии от прошлых версий React, где приходилось создавать классовые и функциональные компоненты, что делало код загруженным и менее лаконичным. В проекте использовались такие «хуки», как: useState (для управления состоянием компонента) и useEffect (для

управления побочными эффектами, к примеру обращение к API или обновление DOM-дерева). Пример данных «хуков» показан на рисунке 9.

```
const [favoriteStyle, setFavotiteStyle] = useState({ color: '#363636' });
const [favoriteBoolean, setFavoriteBoolean] = useState(false);

const { data, isLoading, error } = useQuery('favorites', fetchDataFavoritesLots)
async function fetchDataFavoritesLots() {
    const querySnapshot = await getDoc(doc(db, 'users', user!.uid))
    return querySnapshot.data()!.favoriteLots
}

useEffect(() => {
    if (data) {
        if (data.find((i: number) => i === id)) {
            setFavotiteStyle({ color: '#fb332c' })
            setFavoriteBoolean(true)
        } else {
            setFavotiteStyle({ color: '#363636' })
            setFavoriteBoolean(false)
        }
    }
}, [data]);
```

Рисунок 9 – Пример «хуков» useState и useEffect

Здесь useState используется для изменения состояния переменных favoriteStyle и favoriteBoolean. При получении данных (data) с сервера, при первом рендере компонента, useEffect ищет в полученных данных, является ли этот лот добавленным в избранное, при положительном результате переменная favoriteStyle становится равна {color: “#fb332c”}, а переменная favoriteBoolean равна true. В противном же случае favoriteStyle становится равна {color: “#363636”}, а переменная favoriteBoolean равна false.[12]

Нужно это для визуального отображения избранного лота, пример показан на рисунке 10.

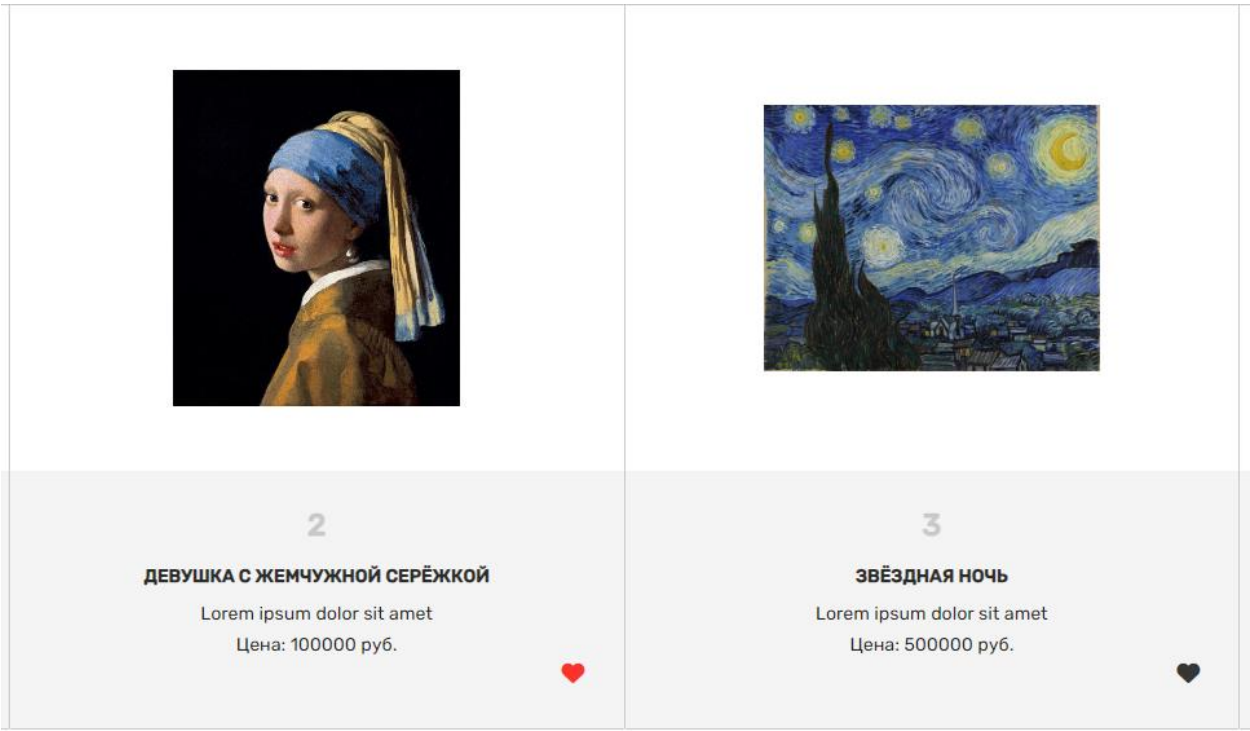

Рисунок 9 – Пример лота добавленного в избранное

Для работы с состоянием всего приложения не использовалась библиотека Redux, но использовался аналог этого стейт менеджера, под названием Jotai.

Jotai — это библиотека состояния для React, предназначенная для управления состоянием в приложениях, использующих React. Она предоставляет простой и интуитивно понятный способ управления состоянием компонентов без необходимости использования классов или хуков.

Основная идея Jotai заключается в использовании атомов (atoms) - маленьких, независимых и неизменяемых единиц состояния. Атомы могут содержать любые данные, такие как числа, строки, объекты или массивы. Каждый атом представлен в виде экземпляра класса Atom, и их значения могут быть прочитаны или изменены.

Для использования Jotai в приложении сначала необходимо создать атомы, определяющие состояние, и затем использовать эти атомы в компонентах.

Для отправки запросов к серверу использовалась библиотека axios и библиотека React Query. Пример использования этих библиотек представлен на рисунке 10.[22]

```
const { data, isLoading, error } = useQuery('auctions', fetchData)

async function fetchData() {
  const querySnapshot = await getDocs(collection(db, 'auctions'))
  return querySnapshot.docs
}

if (isLoading) return <div className='auction__isloading'><Loader size="md" content="Loading..." /></div>

if (error) return <div>Something went wrong...</div>

if (data === undefined) return <div>Undefined</div>
```

Рисунок 9 – Пример запроса на сервер и обработка ответа от него

Здесь функция fetchData() выполняет запрос на сервер по средствам библиотеки React Query, которая в свою очередь возвращает ответ от сервера (data), состояние ожидания ответа от сервера (isLoading), и ошибку (error), если такая случается. Так как ответ от сервера занимает какое-то время, обработка этих состояний является хорошей практикой во front-end разработке.

Итоги главы

Целью раздела являлось описание ключевых моментов разработки клиентской части приложения.

В разделе 3.1 была описана работа с библиотеками React, Jotai, axios, React Query для разработки веб-приложения, учитывая их особенности и нововведения.

ГЛАВА 4. ОПИСАНИЕ РЕЗУЛЬТАТОВ РАЗРАБОТКИ

§1. Front-end

Результатом разработки клиентской части являются страницы:

– главная страница;

– страница аукционов;

– страница создания аукционов;

– страница лотов;

– страница открытого лота;

– поиск по аукционам и лотам;

– страница регистрации;

– страница авторизации;

– страница избранных лотов;

– страница личного кабинета.

Вышеприведенные страницы представлены на рисунках 10-18. За основу контента страниц взято содержимое из разных музеев мира.

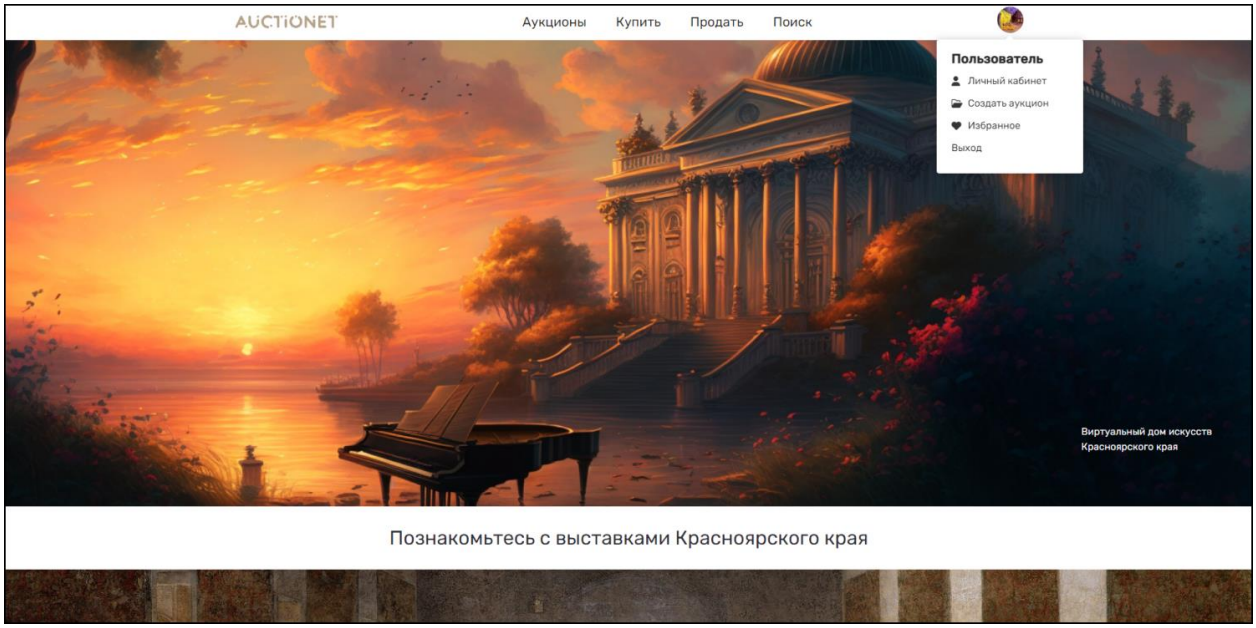

Рисунок 10 – Главная страница Интернет-аукциона

На главной странице сайта расположена основная информация о Доме искусств Красноярского края. Стоит отметить навигационную панель и футер, присутствующие на всех основных страницах системы.

Навигационная панель отвечает за переход между страницами сайта. Содержание навигационной панели сайта представлена на рисунке 11.

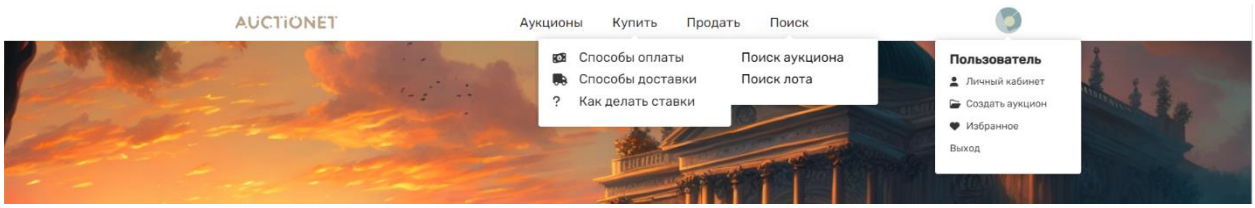

Рисунок 11 – Содержание навигационной панели сайта

Страницы «Способы оплаты», «Способы доставки», «Как делать ставки», «Как продать», «Преимущества» ведут на отдельные страницы с информацией для пользователя сайта.

Также на навигационной панели располагаются кнопка «вход», которая перенаправляет пользователя на страницу регистрации или логина. Если пользователь уже авторизован, то в навигационной панели появляется его аватар, по клику на который открывается меню, в котором можно перейти в личный кабинет, создать аукцион, посмотреть список избранных лотов, который добавил этот пользователь, и кнопка выхода.

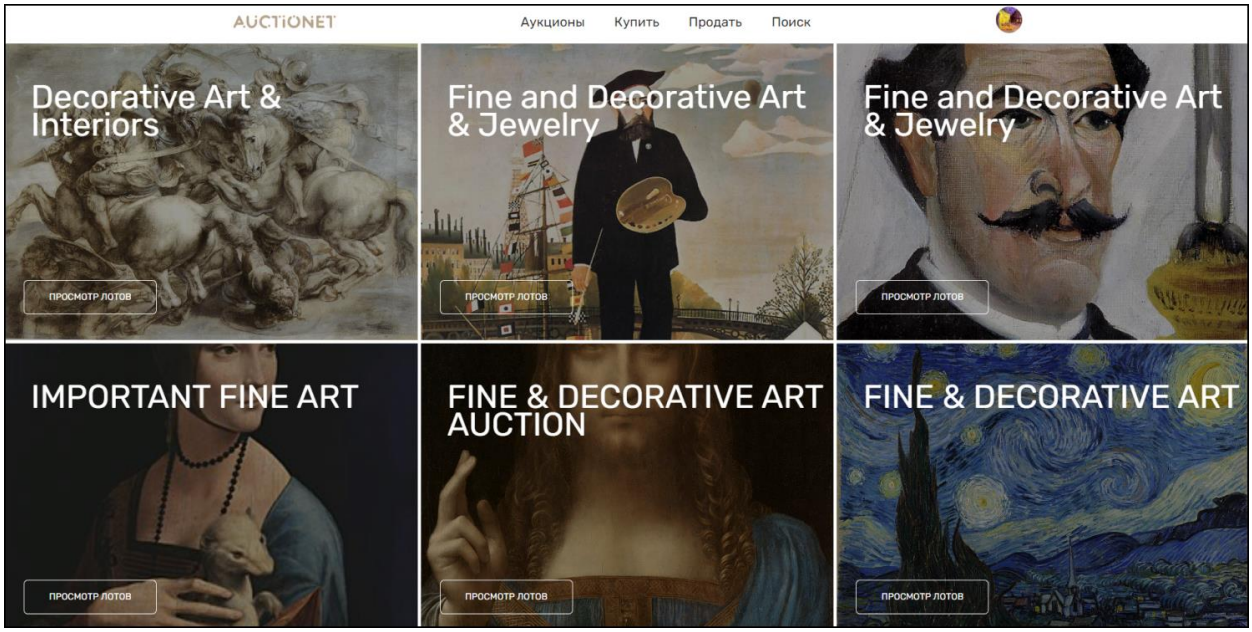

Рисунок 12 – Страница аукционов

На странице аукционов расположен основной функционал сайта, тут находятся аукционы, добавленные пользователями сайта, после прохождения модерации. О процессе создания аукциона будет сказано далее. Каждый аукцион можно открыть для получения лотов, содержащиеся в этом аукционе, а также получить информацию об авторе работы и т.д.

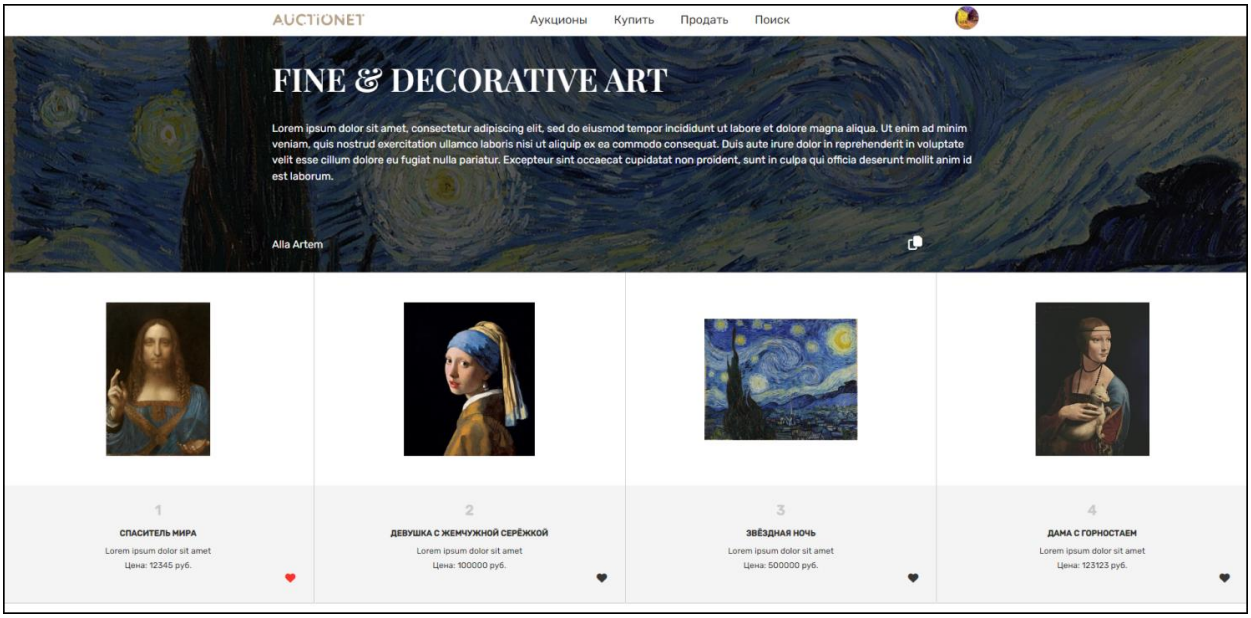

Рисунок 13 – Страница лотов

После перехода на определенный аукцион, мы попадаем на страницу лотов этого аукциона, на данной странице можно узнать описание аукциона, автора, получить ссылку для того, чтобы поделиться этим аукционом со своими знакомыми, а также сами лоты. Количество лотов в аукционе определяется автором и никак не ограничивается платформой. Каждый лот имеет название, краткую информацию, цену, а также возможность добавления его в избранные. Для получения более подробной информации о лоте, можно нажать на него.

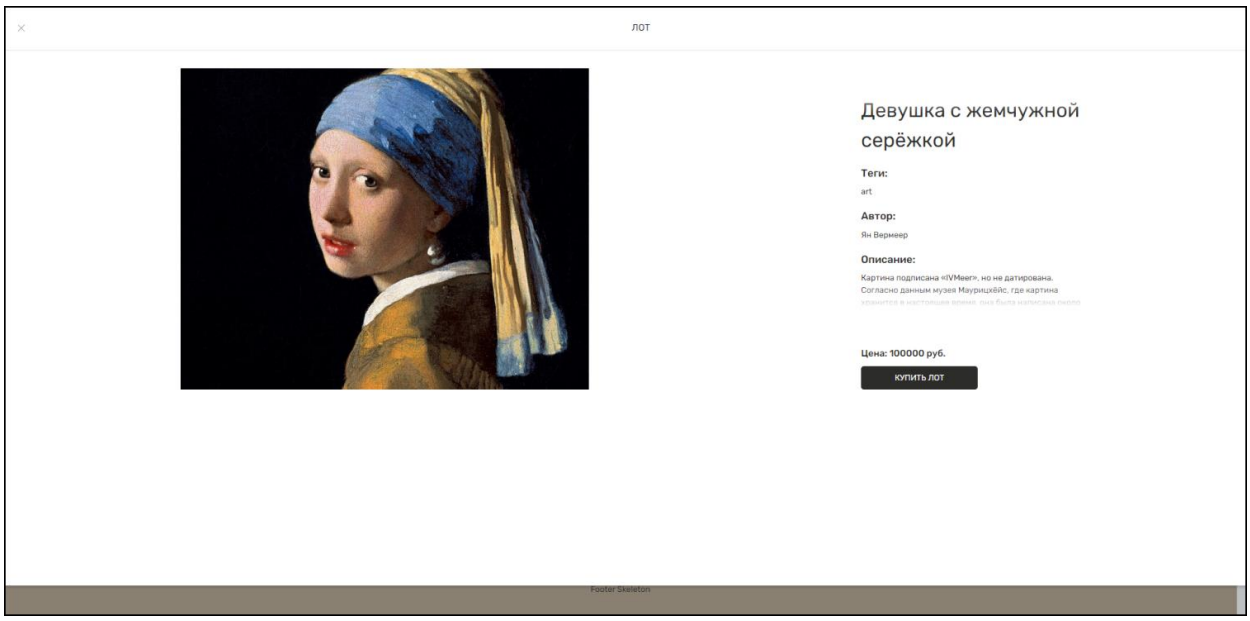

Рисунок 14 – Страница открытого лота

После клика на лот, открывается полная информация которую предоставил автор об этом лоте: теги, описание, цену. Так же с этой страницы можно сразу перейти к оплате лота.

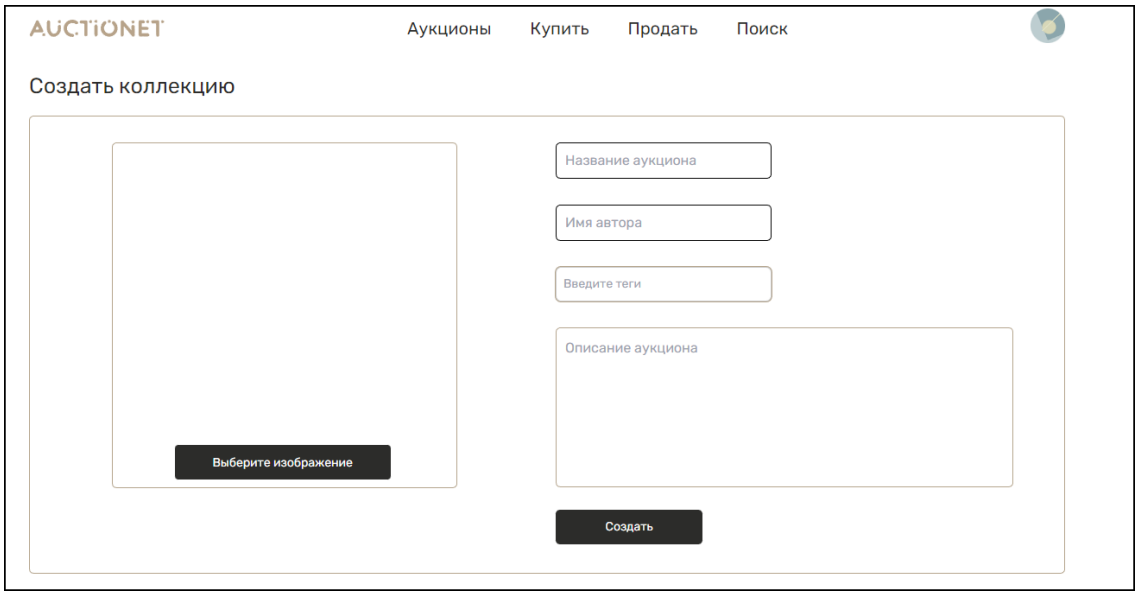

Рисунок 15 – Страница создания коллекции

Страница создания коллекции содержит следующие поля:

– название аукциона;

– имя автора;

– теги;

– описание;

– возможность выбрать изображение.

После того как коллекция создана, автор может наполнять свою коллекцию лотами, страница создания лотов содержит следующие поля для заполнения:

– название лота;

– теги;

– описание;

– возможность выбрать изображение;

– цена.

Рисунок 16 – Поиск

Вызвать поиск можно из любого места на сайте, так как вызов его происходит из навигационной панели в пункте поиска.

Рисунок 17 – Личный кабинет

На странице личного кабинета можно редактировать информацию о себе, задать имя, фамилию, а также изменить почту и аватар.

В левой части имеется навигационное меню, которое перенаправляет пользователя на страницу избранных товаров, так же можно посмотреть уведомления, полученные от сайта, посмотреть корзину и историю покупок.

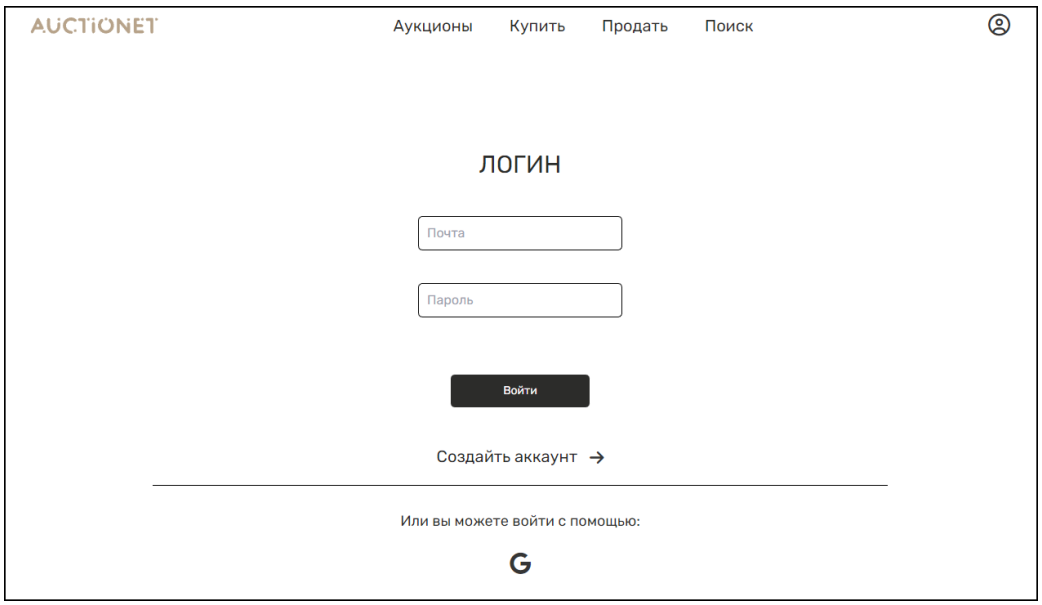

Рисунок 18 – Страница авторизации

На странице авторизации имеется возможность создать аккаунт, если такового нет, или же войти одним из доступных способов, через e-mail и пароль или с помощью google аккаунта.

Итоги главы

Целью данного раздела являлось описание результатов разработки веб-платформы. Были представлены скриншоты интерфейса клиентской части, а также описан функционал разработанных страниц.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы был проведён анализ существующих аналогов разработанной веб-платформы, была выявлена её актуальность, были составлены требования, описаны этапы проектирования и разработки клиентской стороны.

Результатом ВКР является веб-платформа, предлагаемая к использованию пользователям, которые хотят купить или продать предметы искусства. Разработанная платформа позволяет:

– регистрироваться и авторизоваться в ней;

– создание аукционов и добавление лотов;

– возможность добавление лотов в избранное;

– возможность посмотреть информацию об аукционе/лоте;

– возможность изменять данные в личном кабинете;

– возможность поиска аукциона/лота.

Несмотря на то, что все поставленные задачи были выполнены, существуют дальнейшие способы улучшения системы, например:

– добавление онлайн оплаты товара;

– добавление ролей пользователей (пользователь, покупатель, администратор, модератор);

– реализация «умного» поиска по тексту статей, который бы учитывал синонимичные слова и выражения;

– добавление комментариев для обсуждения аукционов;

– добавление истории просмотренных аукционов/лотов;

– добавление рекомендаций на основе просмотренных работ.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. В.В. Алексеева, «Что такое искусство?». – 1973

2. Дмитриева Н.А. Краткая история искусства. Вып. 2. – 1985г.

3. Шорохов В. В. Основы композиции. М., – 1979.

4. Результаты опроса разработчиков в 2019 году [Электронный ресурс] // Система вопросов и ответов о программировании Stack Overflow. – Режим доступа: https://insights.stackoverflow.com/survey/2019. (дата обращения 17.03.2023).

5. Основные концепции React.js, о которых стоит знать [Электронный ресурс] // Библиотека программиста. – Режим доступа: https://proglib.io/p/react-jsconcepts/. (дата обращения 13.03.2023).

6. SEO против React: Веб-краулеры умнее, чем вы думаете [Электронный ресурс] // Сайт организации freeCodeCamp. Режим доступа: https://www.freecodecamp.org/news/seo-vs-react-is-it-neccessary-to-render-reactpages-in-the-backend-74ce5015c0c9/. (дата обращения 17.03.2023).

7. Веб-рендер [Электронный ресурс] // Портал веб-разработки Google. – Режим доступа: https://developers.google.com/web/updates/2019/02/rendering-on-theweb. (дата обращения 17.03.2023).

8. Многоуровневая архитектура [Электронный ресурс] // Сайт сибирского отделения Российской академии наук. – Режим доступа: http://wwwsbras.nsc.ru/Report2006/Report321/node30.html. (дата обращения 17.03.2023).

9. Учебник языка JavaScript [Электронный ресурс]: Современный учебник JavaScript. URL: https://learn.javascript.ru/. (дата обращения 17.03.2023).

10. Learn HTML [электронный ресурс]: обучающий курс. URL: https://www.codecademy.com/learn/learn-html. (дата обращения 17.03.2023).

11. Learn CSS [электронный ресурс]: обучающий курс. URL: https://www.codecademy.com/learn/learn-css. (дата обращения 13.03.2023).

12. Введение в хуки [Электронный ресурс] // Документация React. – Режим доступа: https://reactjs.org/docs/hooks-intro.html. (дата обращения 17.03.2023).

13. Государственный русский музей / ред. А.Н. Савинов. - М.: Изогиз, 2018. - 171 с.

14. Алекс, Бэнкс React и Redux. Функциональная веб-разработка. Руководство / Бэнкс Алекс. - М.: Питер, 2018. - 458 с.

15. Что такое архитектура программного обеспечения [Электронный ресурс] // Сайт компании IBM. – Режим доступа: https://www.ibm.com/developerworks/ru/library/eeles/index.html#notes. (дата обращения 13.03.2023).

16. Многоуровневые системы клиент-сервер [Электронный ресурс] // Сайт издательства «Открытые системы». – Режим доступа: https://www.osp.ru/nets/1997/06/142618/. (дата обращения 17.03.2023).

17. API [Электронный ресурс]. – Режим доступа: https://tproger.ru/articles/web-api/; (дата обращения 17.03.2023).

18. Дэвид Флэнаган, «JavaScript. подробное руководство, 6-е издание», 2012 г.;

19. Руководство по React Router [Электронный ресурс]. – Режим доступа: https://reacttraining.com/react-router/core/guides/quick-star. (дата обращения 13.03.2023).

20. Дунаев, Вадим JavaScript. Самоучитель / Вадим Дунаев. - М.: Питер, 2003. - 400 с.

21. Симпсон, Кайл ES6 и не только / Кайл Симпсон. - М.: Питер, 2017. - 336 с.

22. Документация Axios [Электронный ресурс]// Сайт библиотеки Axios. – Режим доступа: https://axios-http.com/docs/intro. (дата обращения 13.03.2023).

СПИСОК СОКРАЩЕНИЙ

SPA - одностраничные приложения

MPA - многостраничные приложения

PWA - прогрессивные веб-приложения

API - Application Programming Interface

HTML - HyperText Markup Language

CSS - Cascading Style Sheets

HTTPS - HyperText Transfer Protocol Secure

HTTP - HyperText Transfer Protocol

XML - eXtensible Markup Language

JSON - JavaScript Object Notation

REST - Representational State Transfer

SOAP - Simple Object Access Protocol

Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждение высшего образования
«**СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ**»

Гуманитарный институт
Кафедра информационных технологий
в креативных и культурных индустриях

УТВЕРЖДАЮ
И. о. заведующего кафедрой
_____ М. А. Лаптева
«__28__» ____июня____ 2023 г.

БАКАЛАВРСКАЯ РАБОТА

Интернет-аукцион Дома искусств Красноярского края.

Направление подготовки: 09.03.03 Прикладная информатика

Наименование программы: 09.03.03.30 Прикладная информатика

Руководитель доц., канд. тех. наук А. В. Усачёв

Выпускник А. В. Алла

Нормоконтролер И. Р. Нигматуллин

Красноярск 2023