**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №4**

**по дисциплине «Веб-технологии»**

Тема: Модуль приложения “Участие в аукционе картин”

|  |  |  |
| --- | --- | --- |
| Студент гр. 8383 |  | Киреев К.А. |
| Преподаватель |  | Беляев С.А. |

Санкт-Петербург

2020

**Цель работы**

Изучение возможностей применения jQuery UI для создания интерфейса пользователя, обработки ошибок, ведения журналов ошибок, реализация взаимодействия приложений с использованием web-сокетов, применение статического анализатора Flow, освоение инструмента сборки WebPack и организации модульного тестирования web-приложений с использованием Mocha.

**Основные теоретические сведения**

jQuery UI (https://jqueryui.com/) – библиотека JavaScript с открытым исходным кодом для создания насыщенного пользовательского интерфейса в веб-приложениях. Она построена на основе библиотеки jQuery и предоставляет упрощенный доступ к её функциям взаимодействия, анимации и эффектов, а также набор виджетов для построения интерфейса пользователя.

Механизм «domain» Node.JS не рекомендуется использовать в связи с разработкой новых механизмов обработки ошибок, но именно он позволяет обрабатывать асинхронные ошибки. При возможности используйте традиционный механизм try/catch или функцию catch() у промисов.

Журналы ошибок позволяют контролировать появление ошибок как на этапе разработки, так и при работе пользователей. В качестве журналов ошибок предлагается использовать Rollbar (https://rollbar.com/) или Sentry (https://sentry.io/).

WebSocket – протокол связи, который может передавать и принимать одновременно сообщения поверх TCP-соединения, предназначен для обмена сообщениями между браузером и web-сервером, но может быть использован для любого клиентского или серверного приложения. Для создания webсокетов предлагается использовать модуль Socket.IO (https://socket.io/).

Flow (https://flow.org/) – инструмент статического анализа JavaScript (разработки Facebook), позволяющий контролировать изменение типов переменных.

Mocha (https://mochajs.org/) – это фреймворк для написания тестов серверной части web-приложений, поддерживает разработку, основанную на тестах (TDD – test-driven development) и разработку, основанную на поведении (BDD – behavior-driven development). Может использоваться совместно с другими библиотеками для тестирования, например, Shell и Chai.

WebPack (https://webpack.js.org/) – модуль JavaScript, обеспечивающий сборку статических пакетов («bundle»). На вход он получает «точки входа» (js-файлы), в которых он находит все зависимости, и формирует соответствующие пакеты (по одному пакету на одну «точку входа»). Пакет представляет из себя специально оформленный js-файл, в него входят не только связанные js-файлы, но и ресурсы, например, css-файлы.

**Общая формулировка задачи**

Необходимо создать web-приложение, обеспечивающее участие в аукционе картин. Каждый участник может подключиться к аукциону картин. В заданное время начинается аукцион. Участники могут торговаться и повышать стоимость продажи картины. Информация о ходе торгов и сделанных ставках рассылается всем участникам с учётом заданной конфигурации аукциона. Аукцион заканчивается, когда заканчивается торг по последней картине. Часть картин может остаться не проданными.

Основные требования:

1. Приложение получает исходные данные из модуля администрирования приложения «Аукцион картин» в виде настроек в формате JSON-файла;

2. В качестве сервера используется Node.JS с модулем express;

3. Участники аукциона подключаются к приложению «Участие в аукционе картин»;

4. Предусмотрена HTML-страница администратора, на которой отображается перечень участников аукциона, перечень картин, текущее состояние по каждой картине (минимальная цена, кому продана, за какую цену), окно (область на странице) с сообщениями о ходе торгов;

5. Предусмотрена HTML-страница участника, на которой отображается информация о балансе средств участника, текущей продаваемой картине, времени, которое прошло с начала торгов, окно (область на странице) с сообщениями о ходе торгов, информация о начальной и текущей цене, предусмотрена кнопки «Подать заявку» и «Предложить новую цену», предусмотрен виджет, позволяющий указать сумму повышения цены (по умолчанию – минимальное допустимое значение), ссылка (или кнопка) перехода на страницу с покупками;

6. Обеспечен контроль, что картину купит только один участник, проверяется, что у участника хватает средств на покупку;

7. Окончание торга по картине осуществляется после истечения заданного времени (соответствующий отсчёт ведётся в окне с сообщениями о ходе торгов);

8. Окно с сообщениями о ходе торгов предназначено для предоставления актуальной информации о ходе торгов (время и текст с соответствующей цветовой подсветкой сообщений). В качестве сообщений как минимум вы ступают: сообщение о начале торгов в целом, о начале торгов по картине, о подаче заявке, о повышении цены, обратный отсчёт об окончании торга по картине, об окончании торгов в целом;

9. Предусмотрена HTML-страница, на которой можно ознакомиться со всеми картинами, купленными участником торгов. Преимуществом будет использование звукового сопровождения событий: начало и окончание торгов в целом, обратный отсчёт об окончании торга.

**Ход работы**

1. Используя среду разработки JetBrains WebStorm, были установлены все необходимые расширения.
2. Используя модуль express, был создан и настроен сервер.
3. При помощи jQuery UI был создан насыщенный пользовательский интерфейс.
4. Разработка интерфейса пользователя:
5. Пользовательский интерфейс представлен на рисунке 1

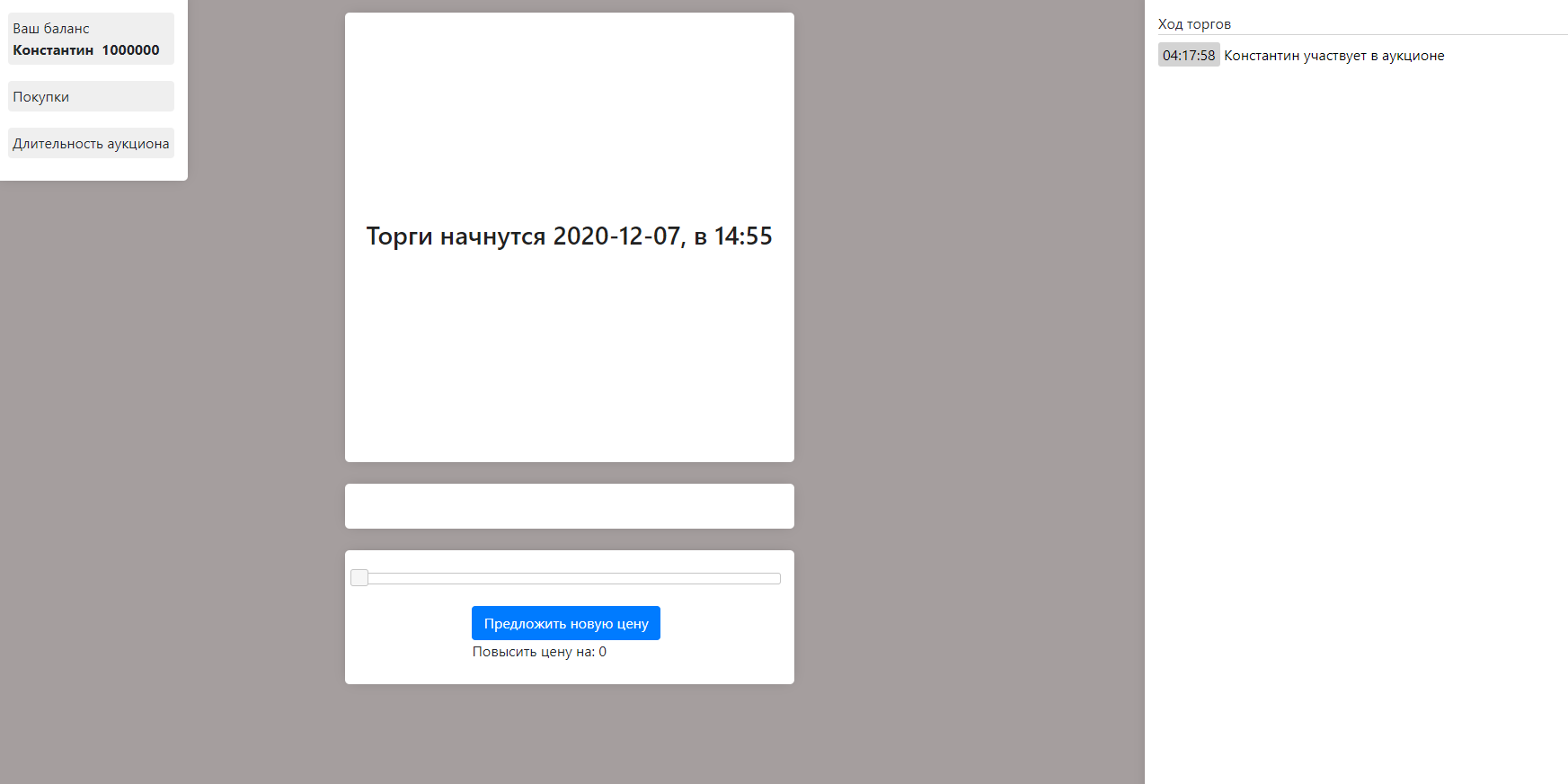


Рисунок 1 – Пользовательский интерфейс

1. Окно администратора представлено на рисунке 2.

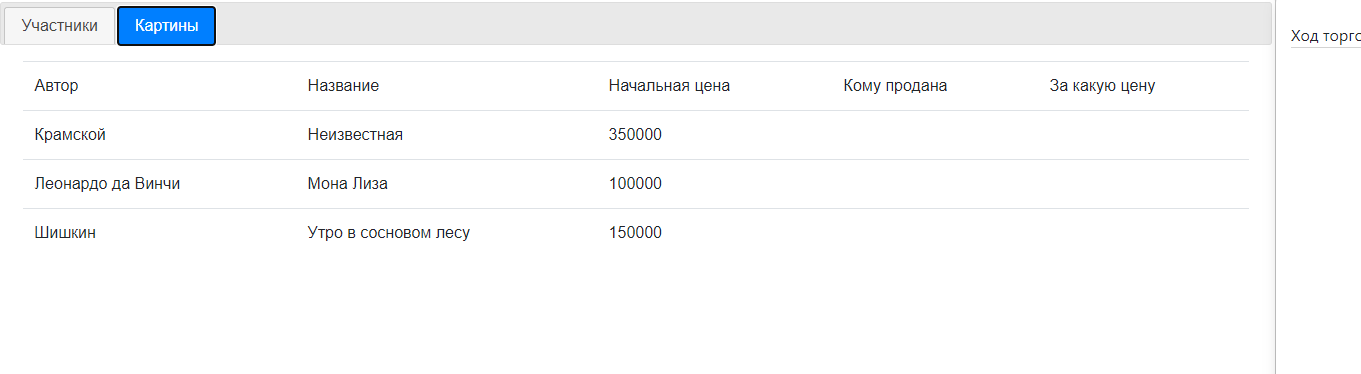


Рисунок 2 – Окно администратора

1. Для проекта были созданы следующие файлы.
2. Был создан набор css файлов для обеспечения стилевого оформления страниц
3. Был создан набор pug файлов, служащие для отображения страниц.
4. autctionInfo.json – начальные настройки аукциона.
5. server.js – содержит функции, обеспечивающие работу сервера и сокетов;
6. sys.js – содержит функции по работе с DOM при помощи jQuery UI, а также функции по работе с сокетами .
7. auctionWait.js – содержит функции, реализующие таймер ожидания аукциона.

**Вывод**

В ходе лабораторной работы был получен опыт работы с jQuery UI, WebPack, Socket.io, Flow, Mocha, на основе реализации модуля приложения “Участие в аукционе картин”