**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МО ЭВМ**

**ОТЧЕТ**

**по лабораторной работе №5**

**по дисциплине «Web-технологии»**

**Тема: Модуль администрирования приложения «Биржа акций»**

| Студент гр. 1303 |  | Чубан Д.В. |
| --- | --- | --- |
| Преподаватель |  | Беляев С.А. |

Санкт-Петербург

2023

## Цель работы.

Целью работы является изучение возможностей применения библиотеки React (https://reactjs.org/) для разработки интерфейсов пользователя web- приложений и использование фреймворка NestJS (https://nestjs.com/) для разработки серверных приложений.

## Задание.

Необходимо создать web-приложение, обеспечивающее настройку биржи брокера, в которой есть возможность задать перечень участников, перечень акций, правила изменения акций во времени. Основные требования следующие:

1. Информация о брокерах (участниках) и параметрах акций сохраняется в файле в формате JSON.

2. В качестве сервера используется NestJS с использованием языка TypeScript.

3. Предусмотрена HTML-страница с перечнем потенциальных брокеров. Брокеров можно добавлять и удалять, можно изменить начальный объем денежных средств.

4. Предусмотрена HTML-страница для перечня акций. Есть возможность просмотреть перечень доступных акций (обозначение, название компании) и исторические данные по изменению курса не менее чем за текущий и предыдущий год. Есть возможность выбрать какие акции будут участвовать в торгах. Минимально должны поддерживаться следующие компании (в скобках – обозначение): Apple, Inc. (AAPL), Starbucks, Inc. (SBUX), Microsoft, Inc. (MSFT), Cisco Systems, Inc. (CSCO), QUALCOMM Incorporated (QCOM), Am- azon.com, Inc. (AMZN), Tesla, Inc. (TSLA), Advanced Micro Devices, Inc. (AMD).

Реальные исторические данные по изменению курса доступны по адресу: https://www.nasdaq.com/market-activity/quotes/historical.

Фрагмент данных для AAPL за три дня (переведён в формат json, оставлены только два столбца: дата и стоимость на время начала торгов): [{"date": "11/5/2021", "open": "$151.89"},

{"date": "11/4/2021", "open": "$151.58"},

{"date": "11/3/2021", "open": "$150.39"}]

5. Предусмотрена HTML-страница для настроек биржи (дата начала торгов, скорость смены дат в секундах при имитации торгов). На этой же странице должна быть кнопка «Начало торгов», которая запускает процесс имитации торгов и предоставление информации об изменении курсов акций всем брокерам по web-сокетам с учётом заданных настроек биржи, здесь же должна отображаться текущая имитируемая дата торгов и текущая стоимость каждой акции.

6. Все элементы в клиентском приложении реализованы с использованием компонентов React. Маршрутизация реализована с использованием «react- router-dom».

7. Для хранения общих данных используется Redux.

8. На сервере спроектированы компоненты и сервисы NestJS для имитации торгов и обработки запросов клиентского приложения.

9. Исторические данные по котировкам представляются как в виде таблиц, так и в виде графиков (например, с использованием Chart.js).

10. Приложение должно реализовывать responsive-интерфейс и корректно работать в том числе при просмотре с мобильного телефона.

11. Для всех страниц web-приложения разработан макет интерфейса с использованием Figma (https://www.figma.com/).

Преимуществом будет создание и использование аутентификации на основе passport.js (http://www.passportjs.org/).

Преимуществом будет использование Material UI React.

**Основные теоретические положения.**

React – библиотека на JavaScript для построения интерфейса пользователя. React представляется удобным инструментом для создания масштабируемых web-приложений (в данном случае речь идет о клиентской части), особенно в тех ситуациях, когда приложение является одностраничным.

В основу React заложены принципы Redux, предлагающее предсказуемый контейнер хранения состояния web-приложения.

Вся структура веб-страницы может быть представлена с помощью DOM. Для решения проблемы производительности предложена концепция виртуального DOM, который представляет собой облегченную версию DOM. React работает именно с виртуальным DOM. Реализован механизм, который периодически сравнивает виртуальный DOM с реальным и вычисляет минимальный набор манипуляций для приведения реального DOM к состоянию, которое хранится в виртуальном DOM.

NestJS – фреймворк для разработки серверных приложений на языках JavaScript и TypeScript. Фреймворк построен на основе компонентного подхода и предлагает стандартизованную структуру приложения по аналогии с Angular.

## Выполнение работы.

Для всех страниц модуля пользователя были разработаны макеты интерфейса с использованием Figma. В приложении А к отчету они представлены.

Разработка серверной части приложения

Для хранения данных об акциях, исторических данных о ценах акций и информации о брокерах была развернута база данных *Mongo.*

Были разработаны следующие сервисы с соответствующими контроллерами и гейтвеями:

1. Broker
2. Controller
3. Stock

*Сервис брокеров*

Информация о брокерах хранится в коллекции *brokers* в *MongoDB*. Схема данной коллекции представлена на рисунке 1.



Рисунок 1 – коллекция брокеров в базе данных

Для взаимодействия с базой данных используется *mongoose* и для сущности брокера реализована соответствующая схема.

Сервис реализует все необходимые методы *CRUD,* а именно создание, удаление, обновление, получение информации о брокере по *ID* и всех брокеров*.*

Данный сервис используется в соответствующем контроллере. Он реализует методы HTTP-запросов. Для описания данных методов использовались декораторы, предоставляемые фреймворком *Nest,* например, *@Post('/create'), @Get('all'),* и т.д.

*Сервис Контроллера*

Реализован сервис, в котором описаны методы старта отправки данных, остановки отправки данных, установки даты начала торгов и задержки отправки.

Сетевое взаимодействие реализовано на основе socket.io. В соответствии с приходящими сообщениями вызываются обработчики.

*Сервис Акций*

Информация об акциях хранится в коллекции *stocks.* Коллекция представлена на рисунке 2.



Рисунок 2 – коллекция акций в базе данных.

Реализован соответствующий интерфейс акций и соответствующие *DTO*. Сервис реализует методы *CRUD,* метод покупки и продажи акции, отправки исторических данных о стоимости акций.

Сетевое взаимодействие основано на использовании *socket.io.* Отправка данных об акциях на определенный день осуществляется при получении соответствующего сообщения от контроллера.

Интерфейс администрирования биржи

Использование библиотеки React (https://reactjs.org/) для разработки интерфейсов пользователя.

*Файл App.jsx*

Основной компонент React-приложения, который использует React Router для навигации между разными страницами.

* function App() { ... }: Объявление функционального компонента App, который будет являться основным компонентом приложения.
* <Router> ... </Router>: оборачивает все компоненты в Router, что обеспечивает поддержку маршрутизации в вашем приложении.
* <Routes> ... </Routes>: определяет контейнер для всех маршрутов.
* <Route exact path="/" element={<StartPage />} />: Определяет маршрут для корневого URL ("/") и указывает, что компонент <StartPage /> должен быть отображен, когда пользователь находится на этом маршруте. Опция exact гарантирует точное соответствие URL.
* Аналогично, определены маршруты и связанные с ними компоненты для страниц "brokers", "stocks", и "settings".
* const root = ReactDOM.createRoot(document.getElementById('root'));: Создание корневого элемента React с использованием createRoot.
* root.render( ... );: Рендеринг всего приложения. Здесь применяется строгий режим React и предоставляется Redux-провайдер, обернутый в провайдер для обеспечения доступности хранилища.
* reportWebVitals();: Отчет о производительности веб-приложения.

Основные компоненты приложения

*BrokerComponent*

Данный компонент реализует страницу брокеров, включает в себя *BrokerContent* и *CreateBroker.*

*BrokerContent*

Данный компонент отвечает за изменение информации о брокере (имени, баланса), обрабатывает действия при нажатии на сохранение и удаление брокера.

*CreateBroker*

Данный компонент представляет собой модальное окно с возможностью ввода данных для создания брокера, в нем же происходит валидация перед отправкой данных на сервер.

*RouterComponent*

Данный компонент реализует маршрутизацию в приложении.

*NavigationComponent*

Данный компонент представляет собой панель навигации в приложении

*TableComponent*

Настраиваемый компонент таблицы, в качестве пропсов передается хедер, контент и название

*StockPageComponent*

Компонент описывающий страницу, содержащую информацию об акциях, включает в себя компонент описанный выше, с переданной информацией об акциях.

*Graph*

Компонент построения графика.

*TradingStocksComponent*

Старница торгов акции, включает в себя список выбранных к торгам акций, а также панель управления торгами.

Изображения с основными страницами приложения представлены в приложении A.

## Выводы.

В ходе лабораторной работы был разработан модуль администратора приложения «Биржа акций».

Были созданы макеты и дизайн интерфейса web-приложения в *Figma.* Созданы страницы веб-приложения, включающие в себя следующие компоненты. Страница брокеров: содержатся данные брокеров, возможность удалить конкретного и добавить нового. Страница со списком акций: просмотр доступных акций компаний, просмотр таблиц и графиков изменения стоимости акций в течение года, выбор акций для имитации торгов. Страница с настройкой биржи: запускает имитацию торгов от даты и обновляет данные по заданному времени.

# Приложение А

Макеты интерфейса.

На рисунке 1 – макет стартовой страницы.

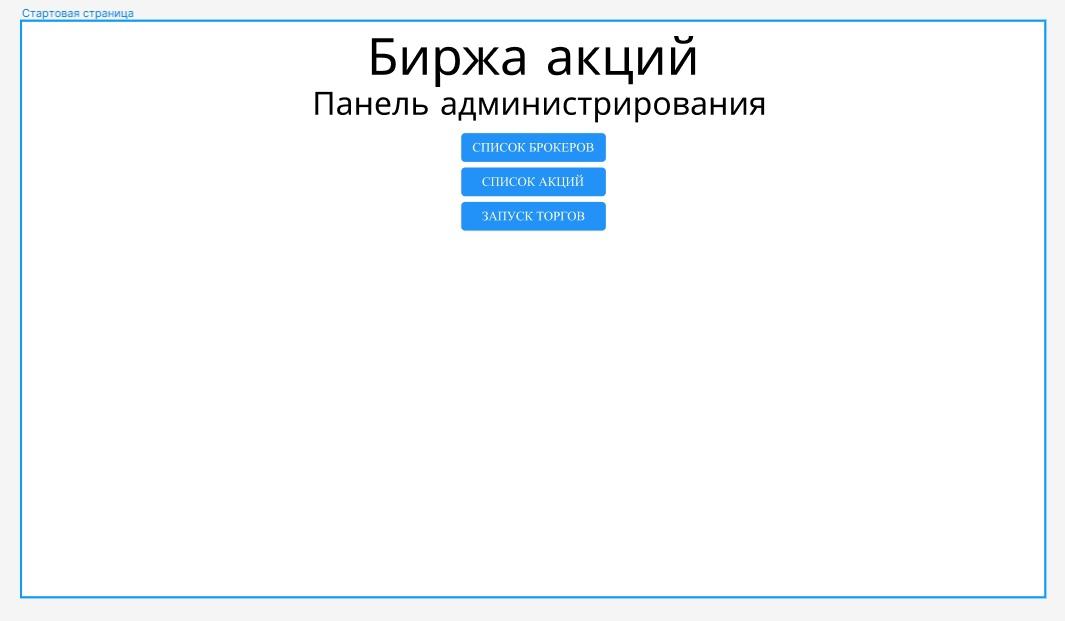


Рисунок 1 – макет стартовой страницы.

На рисунке 2 – макет страницы брокеров.

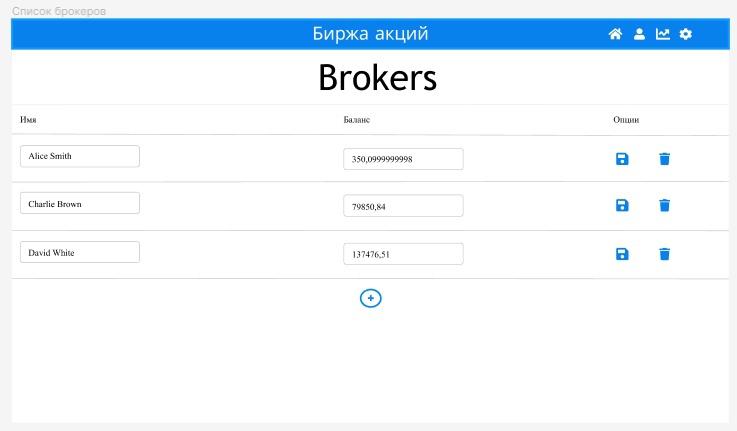


Рисунок 2 – макет страницы брокеров.

На рисунке 3 – макет страницы списка акций.

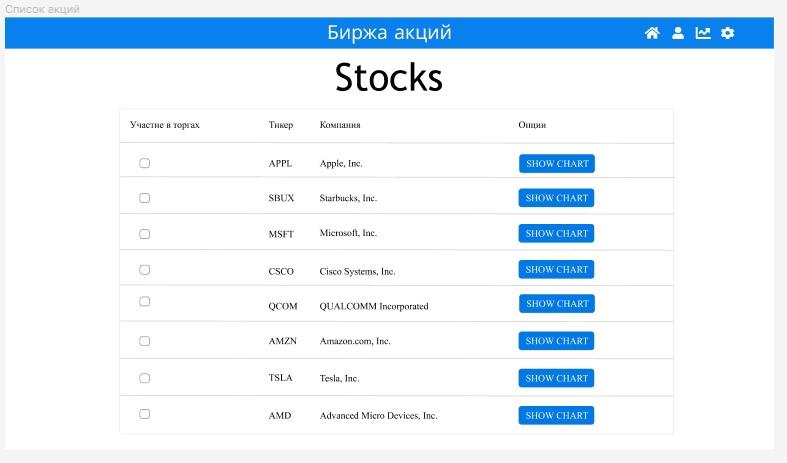


Рисунок 3 – макет страницы списка акций.

На рисунке 4 – макет страницы настройки биржи.

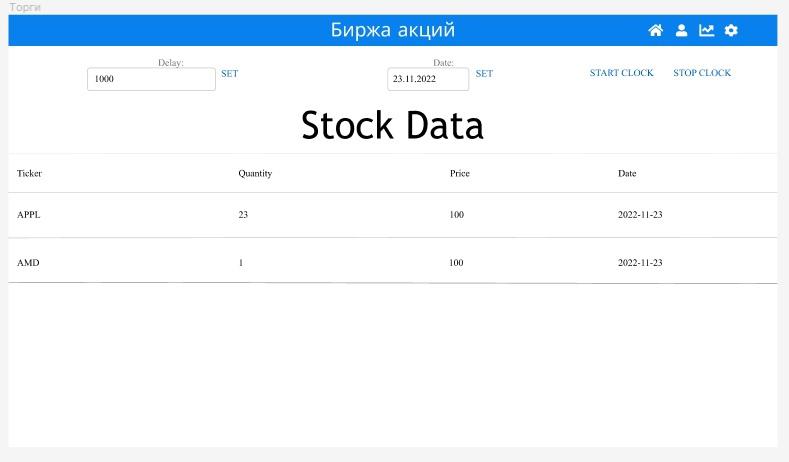


Рисунок 4 – макет страницы настройки биржи.

Модуль администрирования «Биржа акций».

На рисунке 5 – стартовая страница.

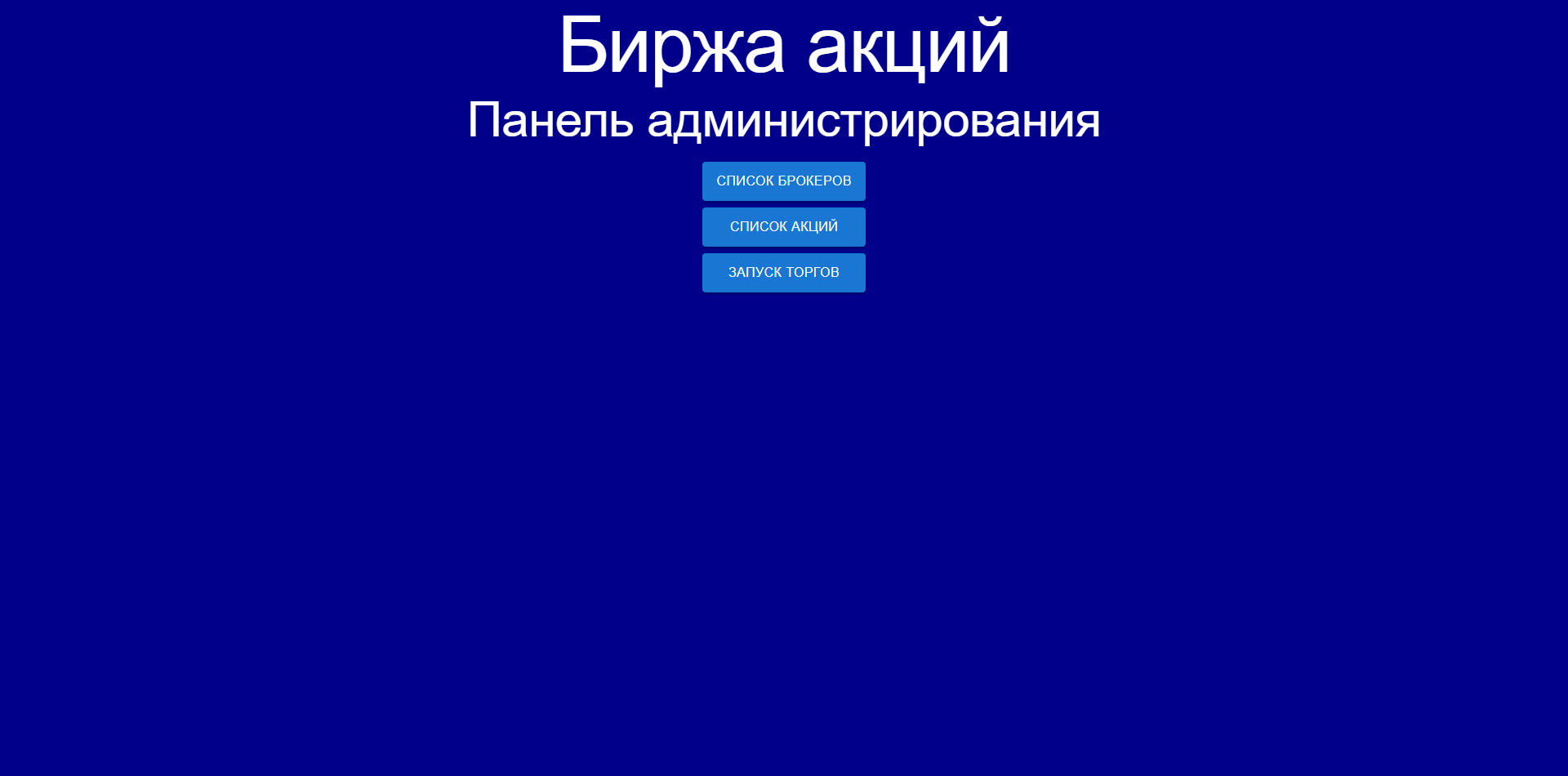


Рисунок 5 – стартовая страница.

На рисунке 6 – страница брокеров.

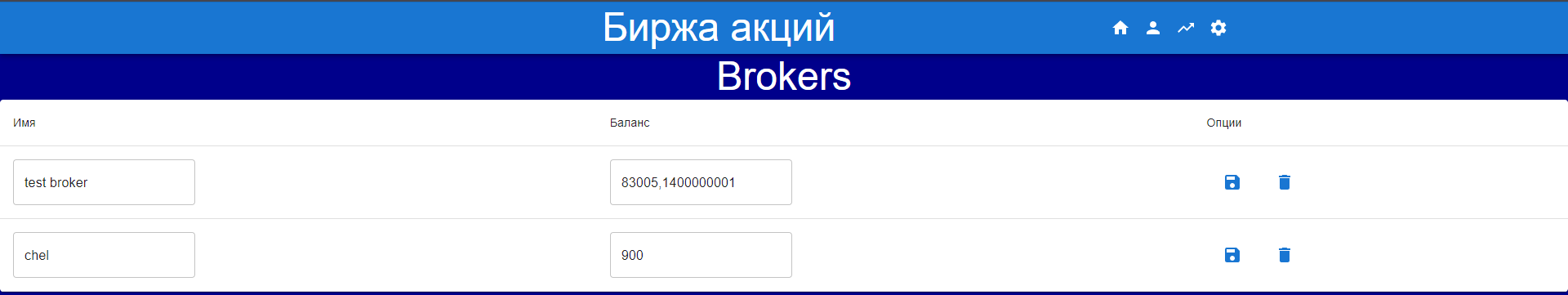


Рисунок 6 – страница брокеров.

На рисунке 7 – страница со списком акций.

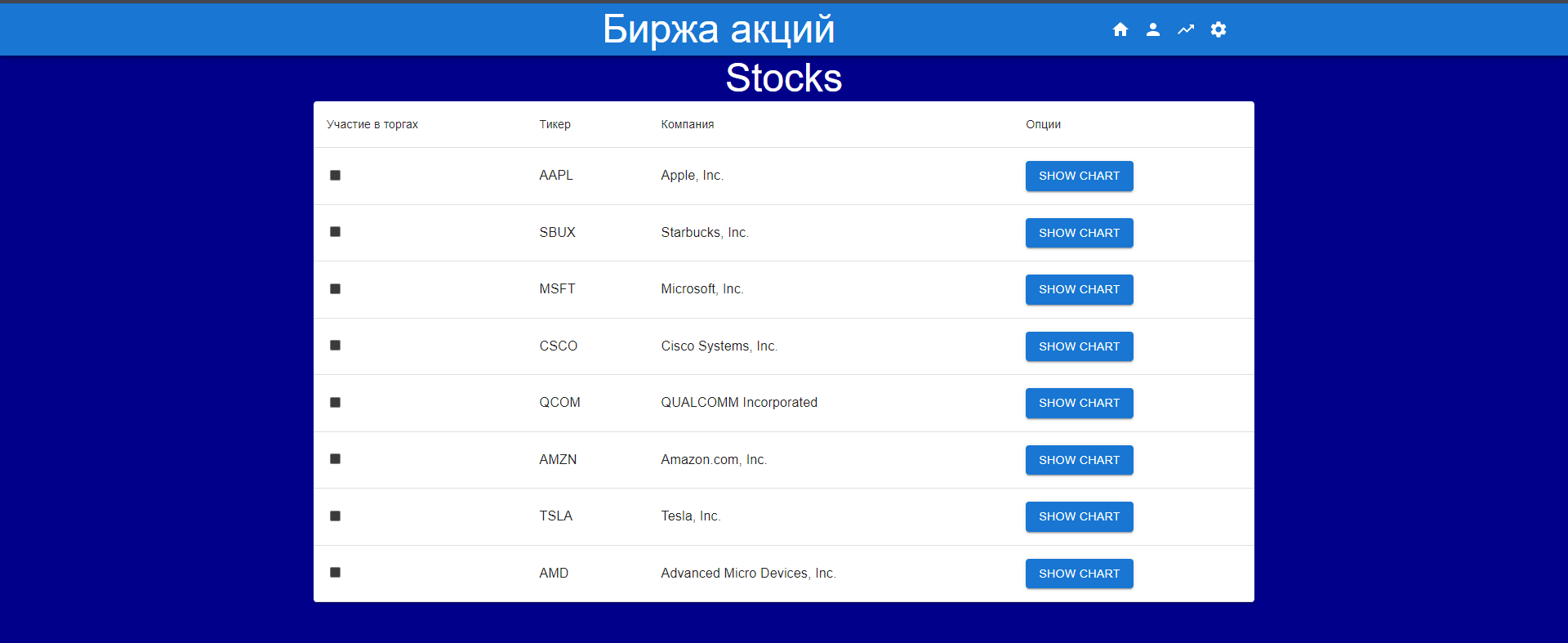


Рисунок 7 – страница со списком акций.

На рисунке 8 – страница с настройками биржи.

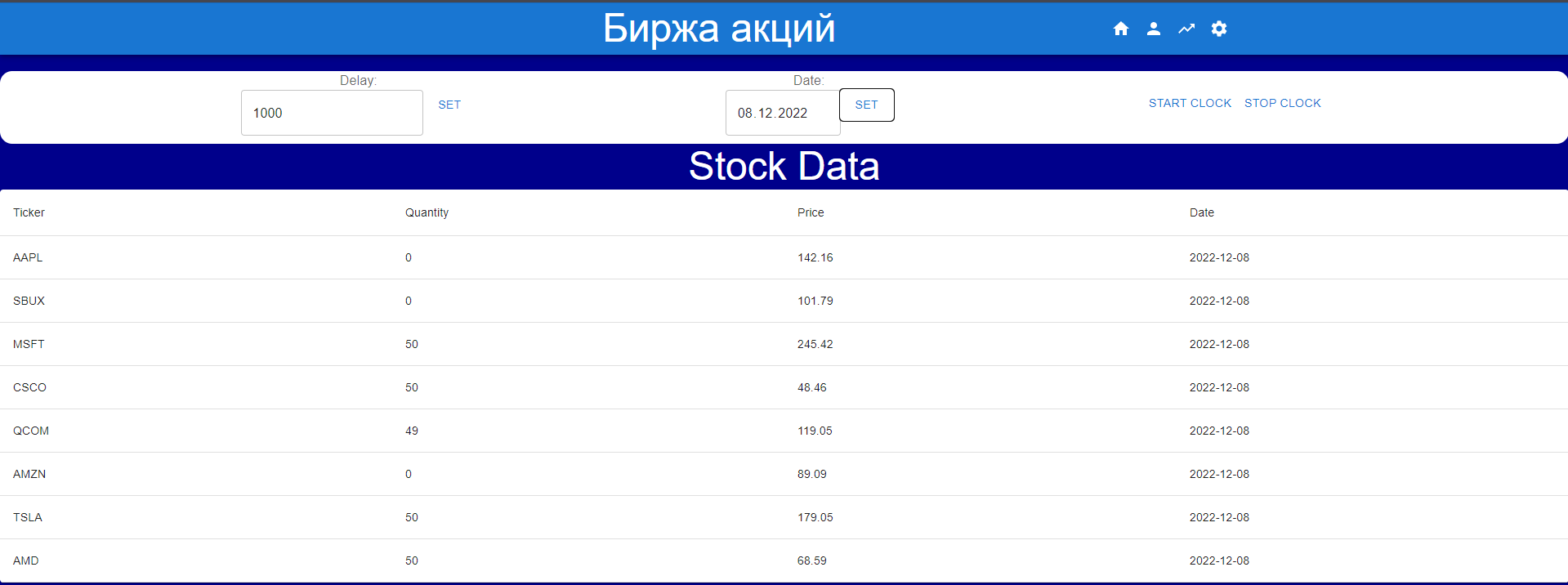


Рисунок 8 – страница с настройками биржи.