Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное

образовательное учреждение высшего образования

«Самарский национальный исследовательский университет

имени академика С.П. Королева»

Институт информатики и кибернетики

Кафедра технической кибернетики

Отчет по лабораторной работе №4

Дисциплина: «Технологии сетевого программирования»

# Тема «Разработка пользовательского интерфейса и взаимодействие с API»

Выполнили:   
Прохоров С.А.  
Жиляев М.И.

Группа: 6301-010302D

Самара, 2025

1. Задание на лабораторную работу

Цель работы: Создать клиентское веб-приложение, подключить его к серверу через API и реализовать основные пользовательские сценарии.

1. **Разработка пользовательского интерфейса:**
   * Создать макеты страниц с помощью шаблонизатора Jinja2 (Flask/Django) или Thymeleaf (Spring);
   * Разработать страницы регистрации, авторизации и работы с данными.
2. **Настройка взаимодействия с сервером:**
   * Использовать формы и серверный рендеринг страниц;
   * Передавать данные в шаблоны и динамически обновлять контент.
3. **Работа с аутентификацией:**
   * Подключить поддержку JWT для авторизации пользователей;
   * Реализовать отображение данных в зависимости от авторизации.
4. **Тестирование интерфейса:**
   * Проверка корректности отображения данных;
   * Отладка передачи данных между сервером и клиентом.
5. **Оформление отчета:**
   * Описание структуры интерфейса и логики работы с API;
   * Примеры рендеринга данных и передачи параметров;
   * Презентация с демонстрацией работы приложения.
6. Выполнение работы
   1. Разработка пользовательского интерфейса:

В рамках данной лабораторной работы были создан сайт с помощью JavaScript React, разработаны основные страницы, настроена связь с сервером для аутентификации и получения основных данных. Страницы представлены на рисунках 1-9, в приложениях А, Б. В приведен пример кода проекта.

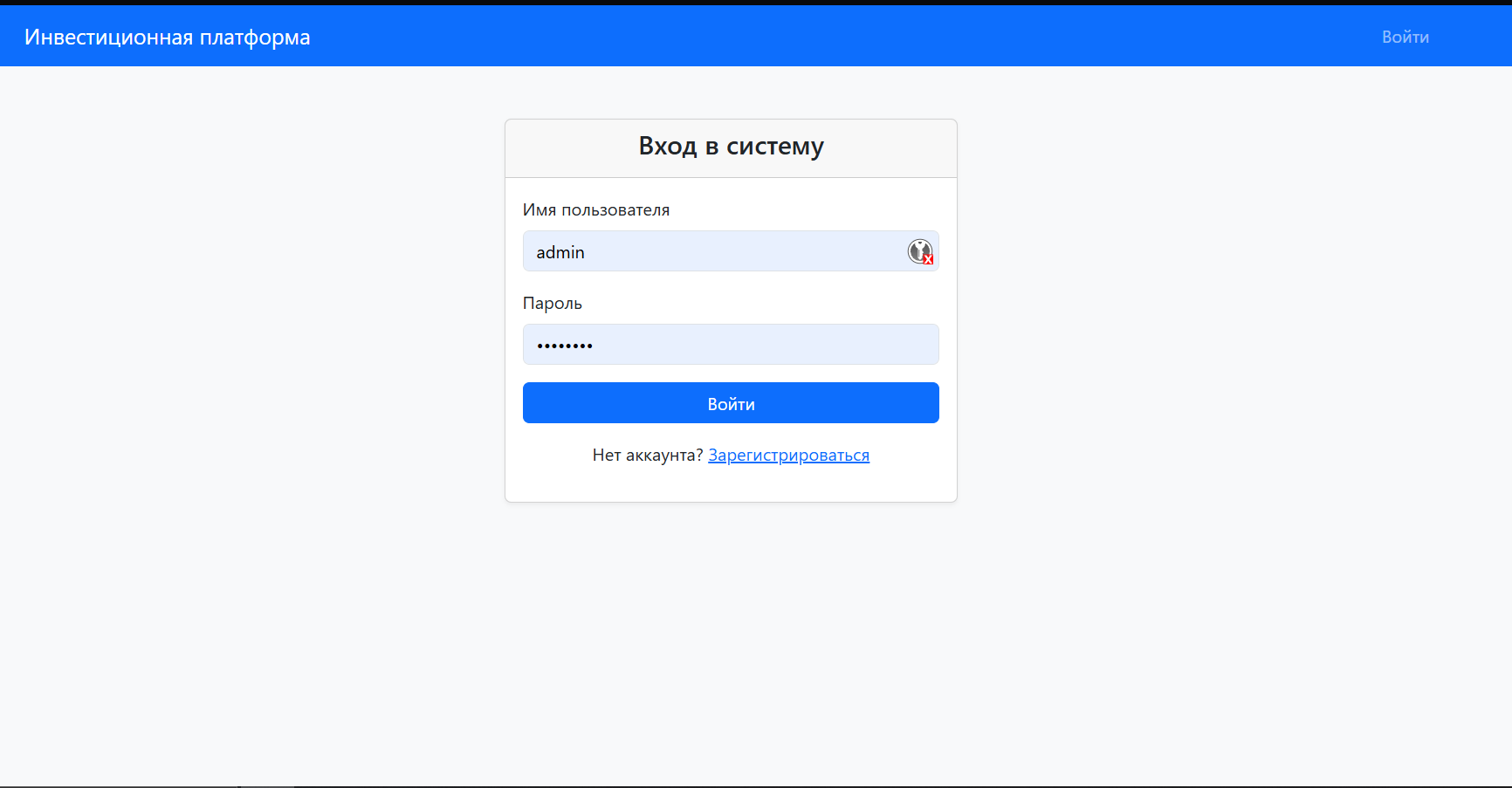


Рисунок 1 - Страница авторизации пользователя

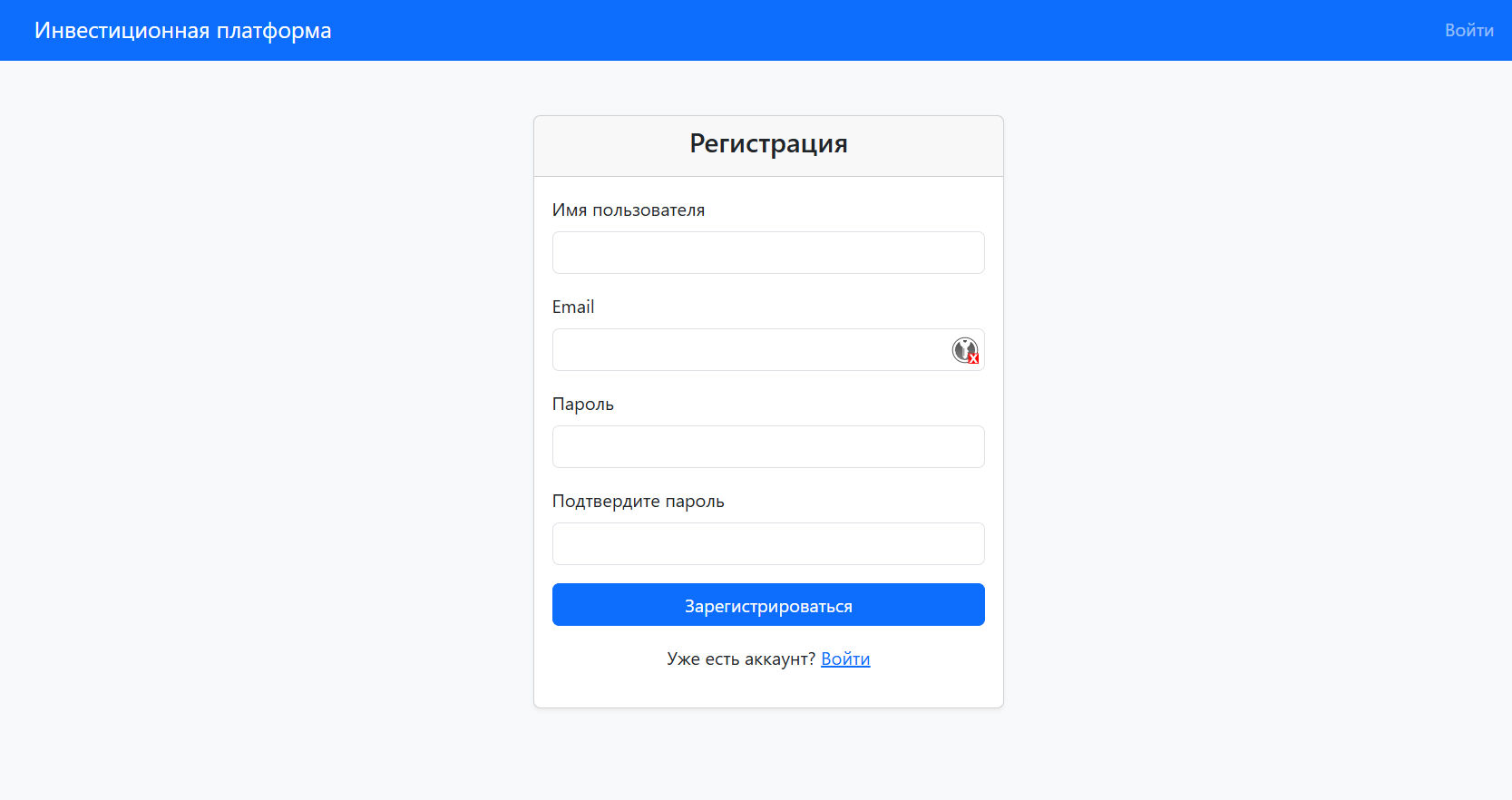


Рисунок 2 - Страница регистрация пользователя

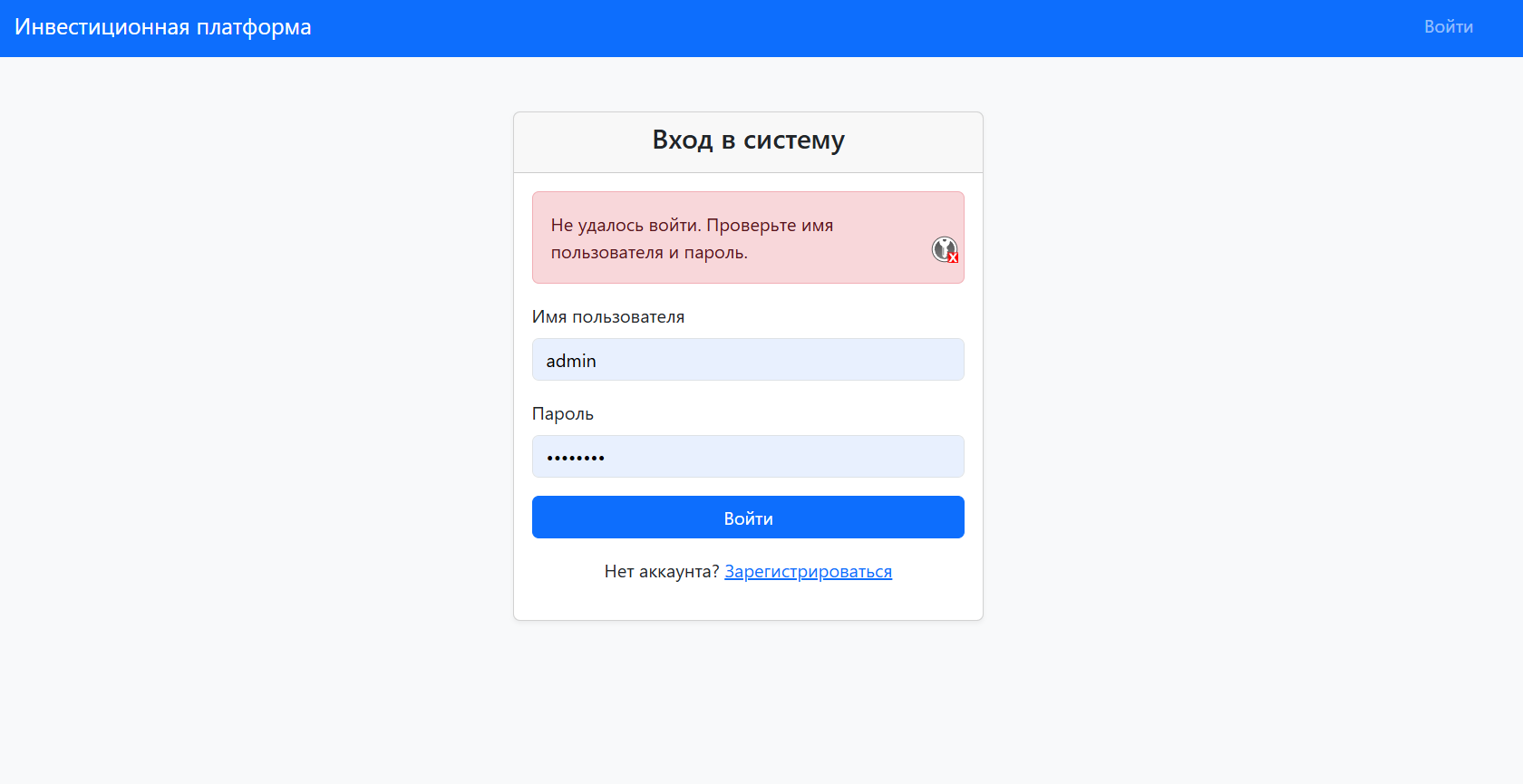


Рисунок 3 – Ошибка при авторизации пользователя с неправильными данными

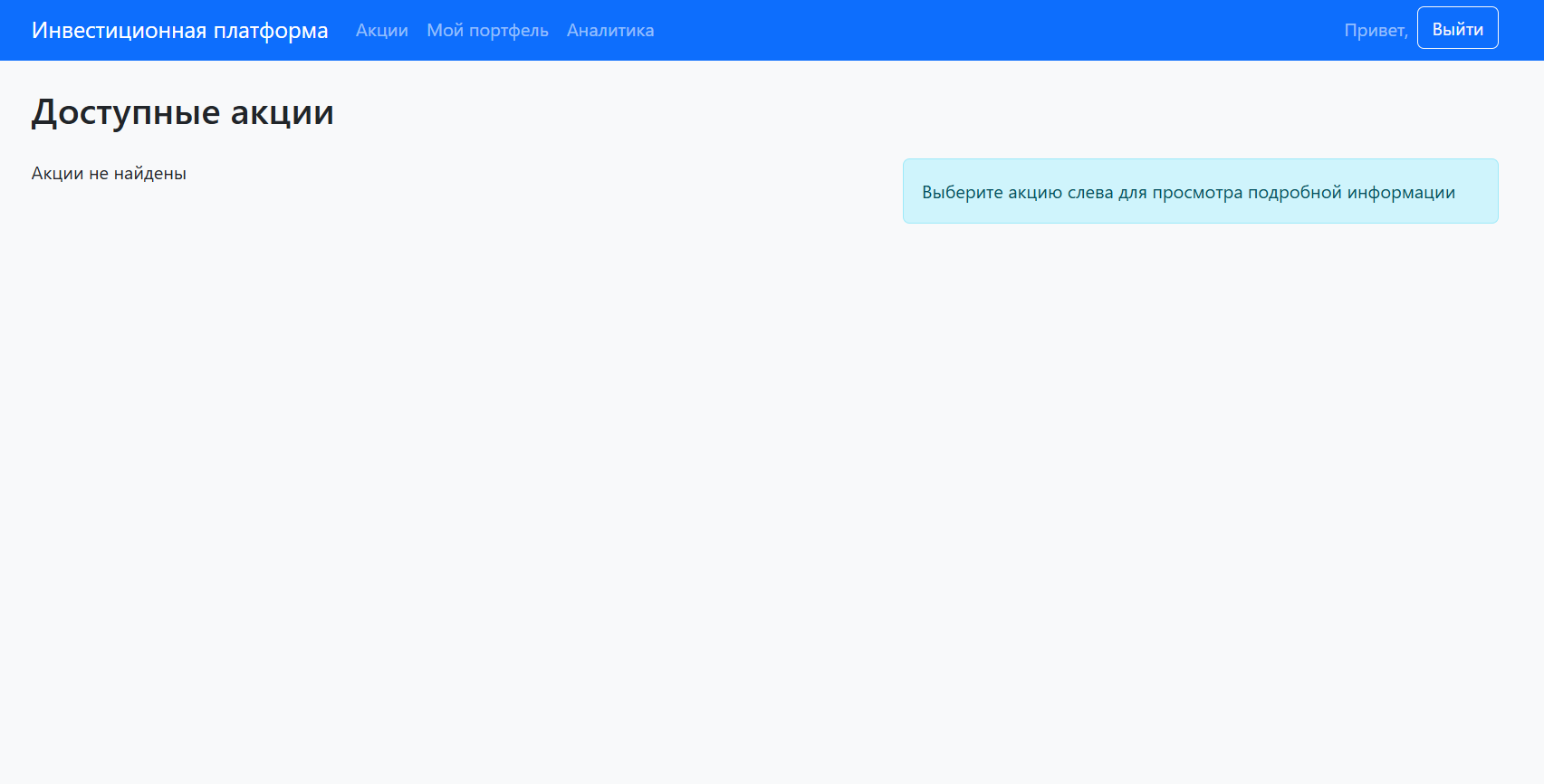


Рисунок 4 – Страница акций при отсутствии записей в базе данных

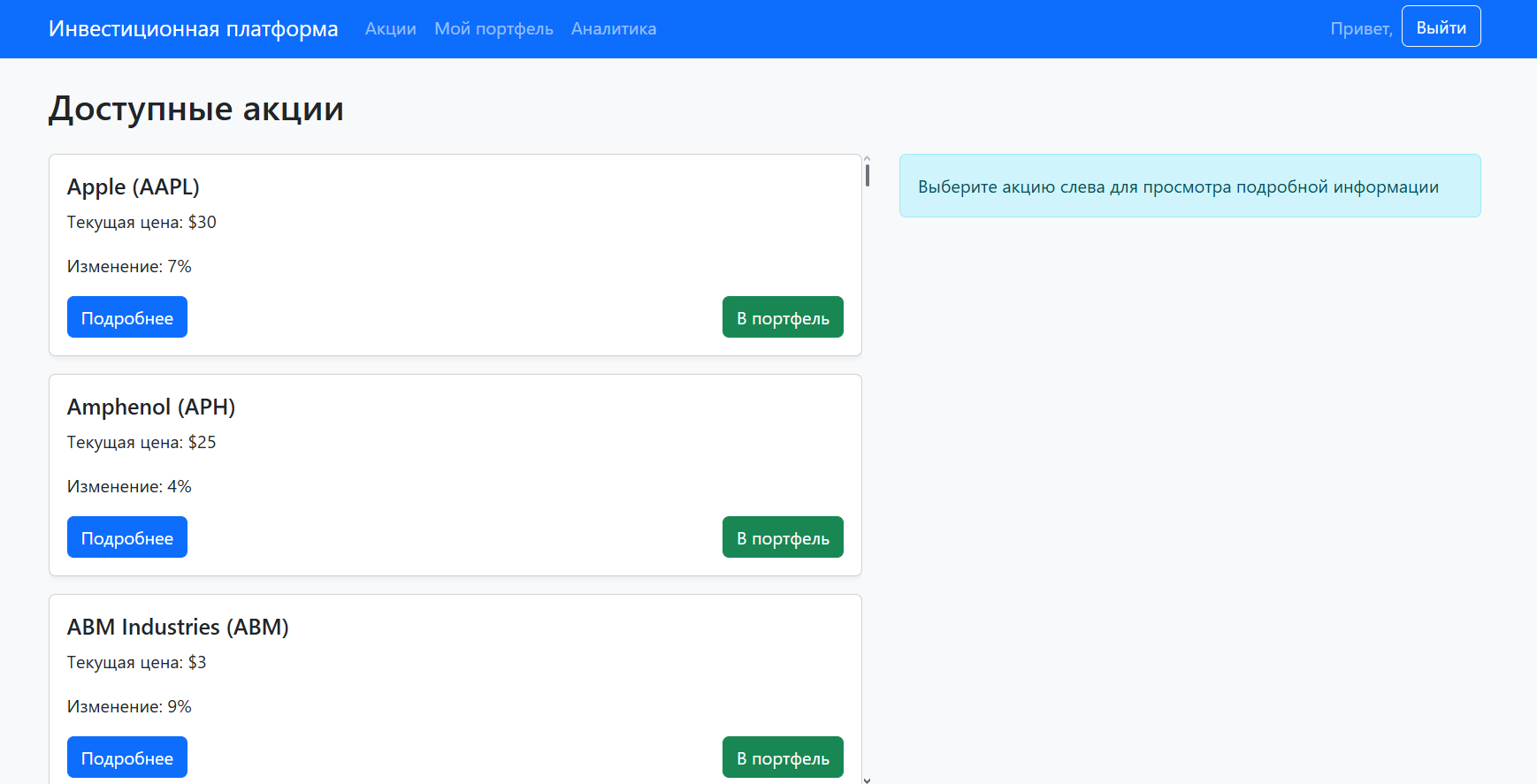


Рисунок 5 – Страница акций

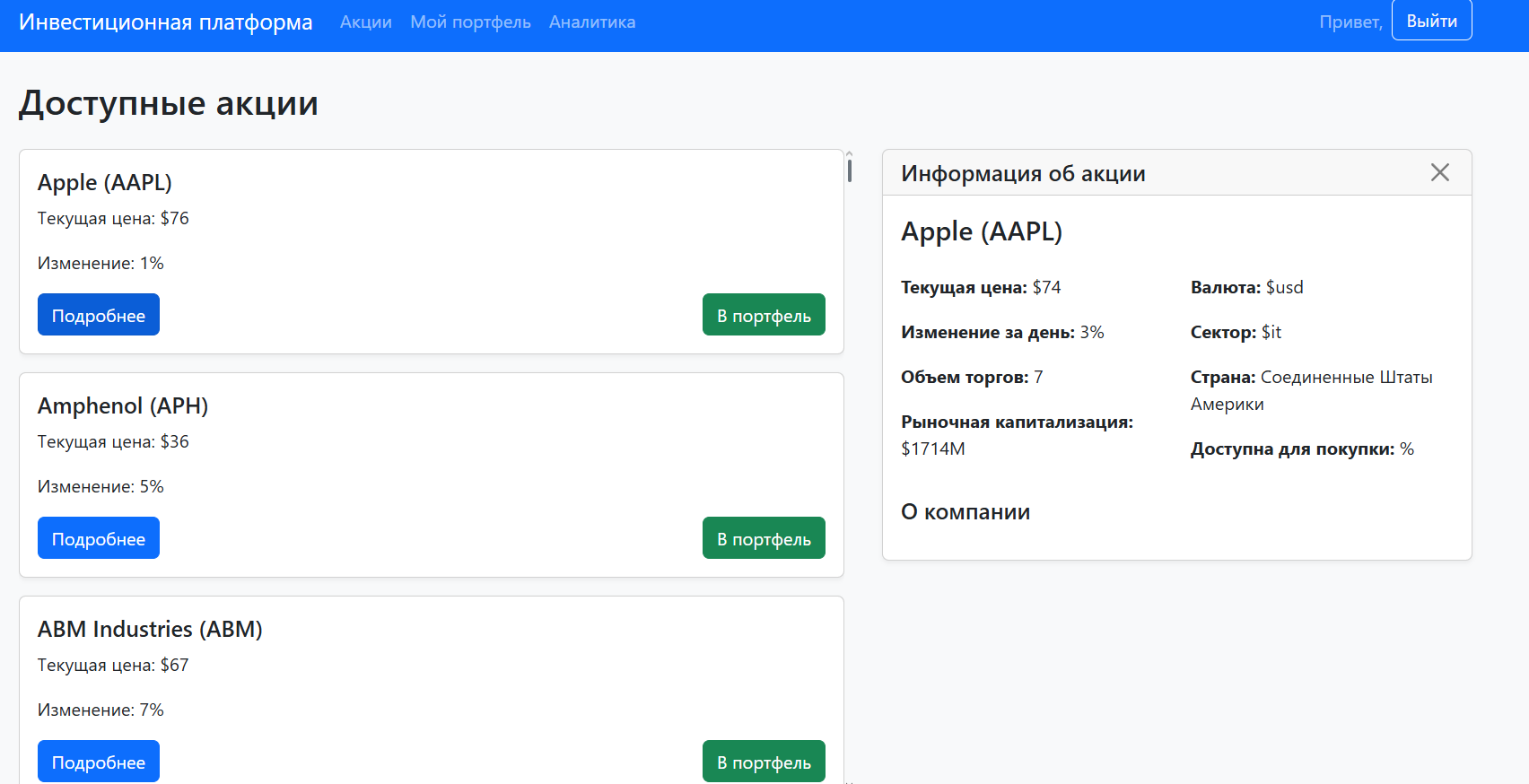


Рисунок 6 – Страница с подробной информацией об акции

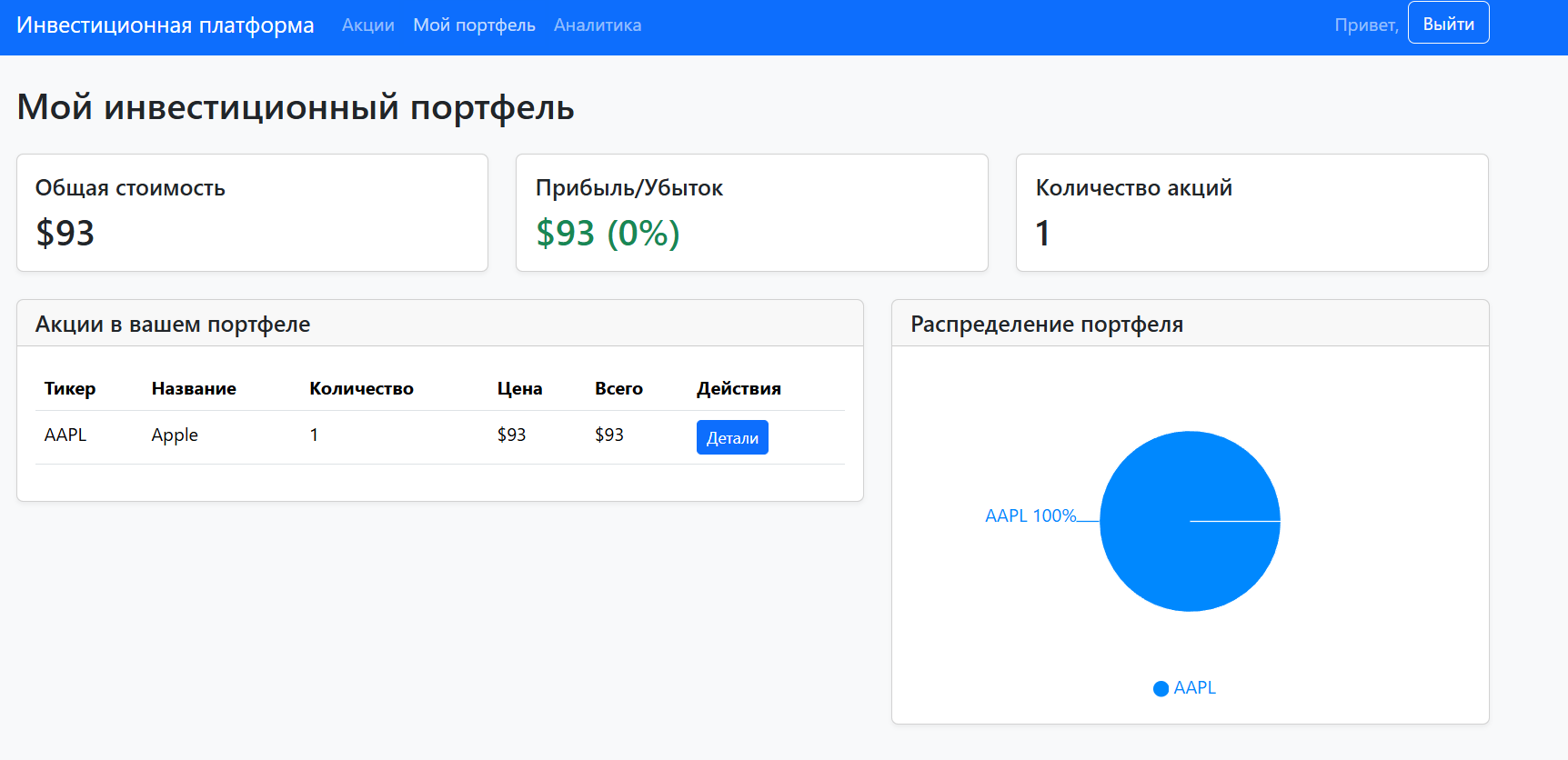


Рисунок 7 – Страница портфолио пользователя

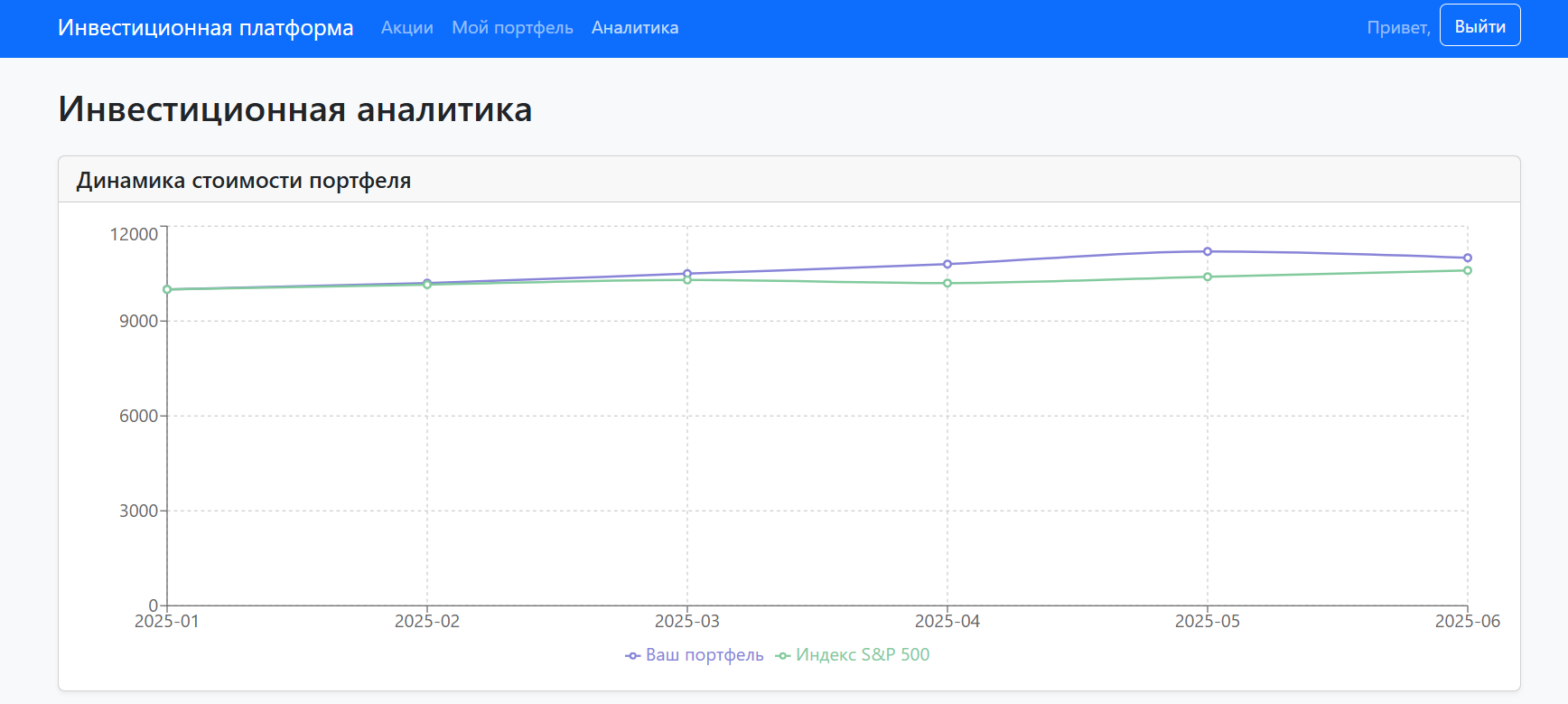


Рисунок 8 - Страница с аналитикой (1)

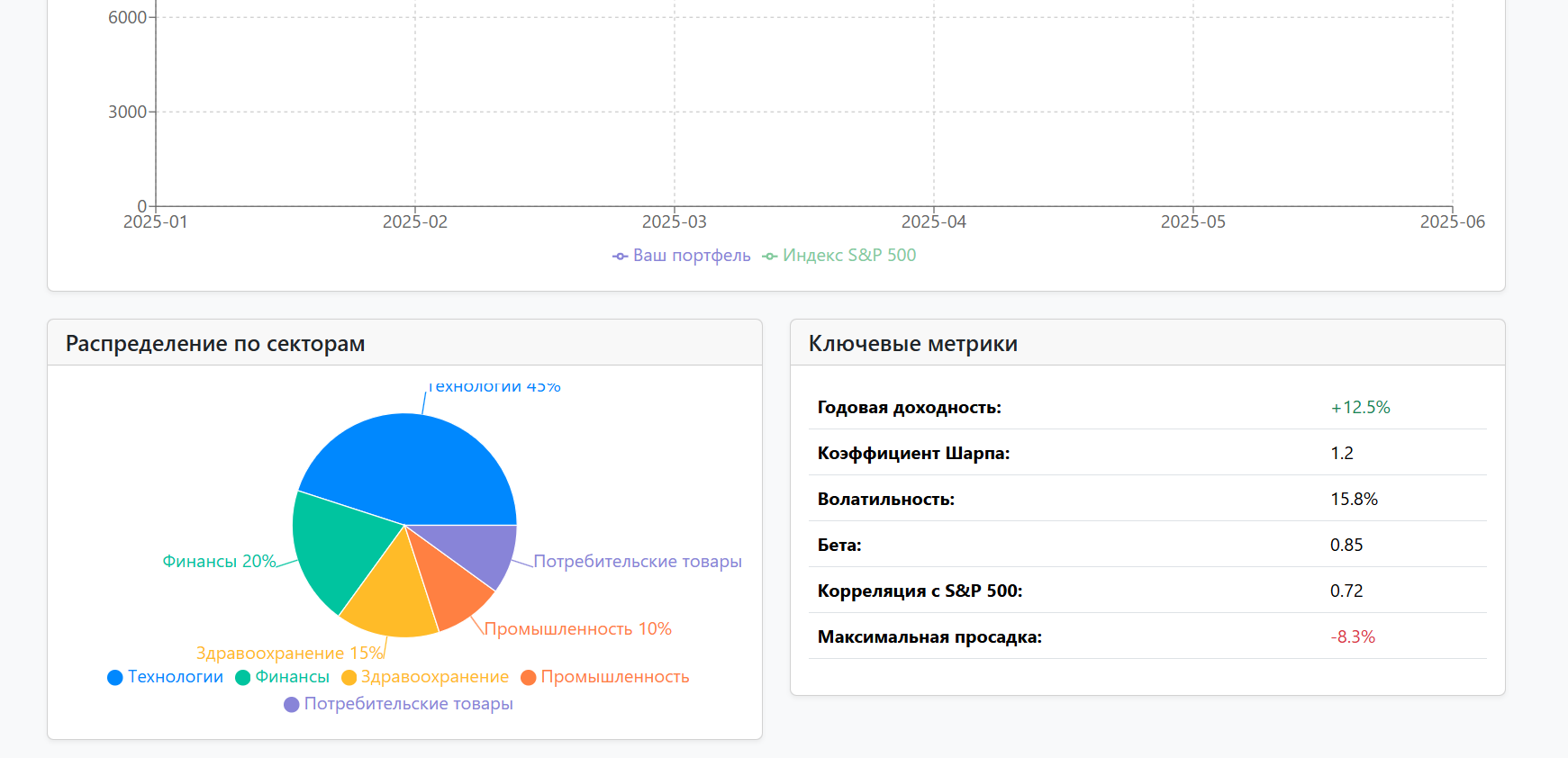


Рисунок 9 - Страница с аналитикой (2)

* 1. Настройка взаимодействия с сервером:

Для создания сайта использовался JavaScript React. Для запросов на сервис использовалась библиотека axios. При переходе на основные странички (/shares, /portfolio) отправляется запрос get на сервис, в ответ сайт получает данные (по всем акциям и акциям пользователя соответственно), записи отправляются в формате json.

К кнопкам на сайте так же привязаны запросы (“Подробнее” отправляет запрос /shares/{uuid}). Для работы сервиса необходима аутентификация, JWT хранится в local storage, его наличие проверяется при запросах.

* 1. Работа с аутентификацией:

При заходе на сайт пользователь должен войти в свой аккаунт. Если аккаунта нет, то пользователь сначала проходит процесс регистрации, а уже затем заходит в аккаунт. Только после успешной аутентификации пользователю становятся доступны остальные страницы сайта, если пользователь не аутентифицирован, то попытки перейти на другие странички даже через адресную строку приведут к перенаправлению на страницу /login. Запросы аутентификации и регистрации отправляются на сервис, в ответ получаем JWT.

* 1. Тестирование интерфейса:

На данном этапе мы столкнулись с проблемой, что при нажатии на кнопку “Выйти” вместо перенаправления на страничку /login для аутентификации сайт зависал.

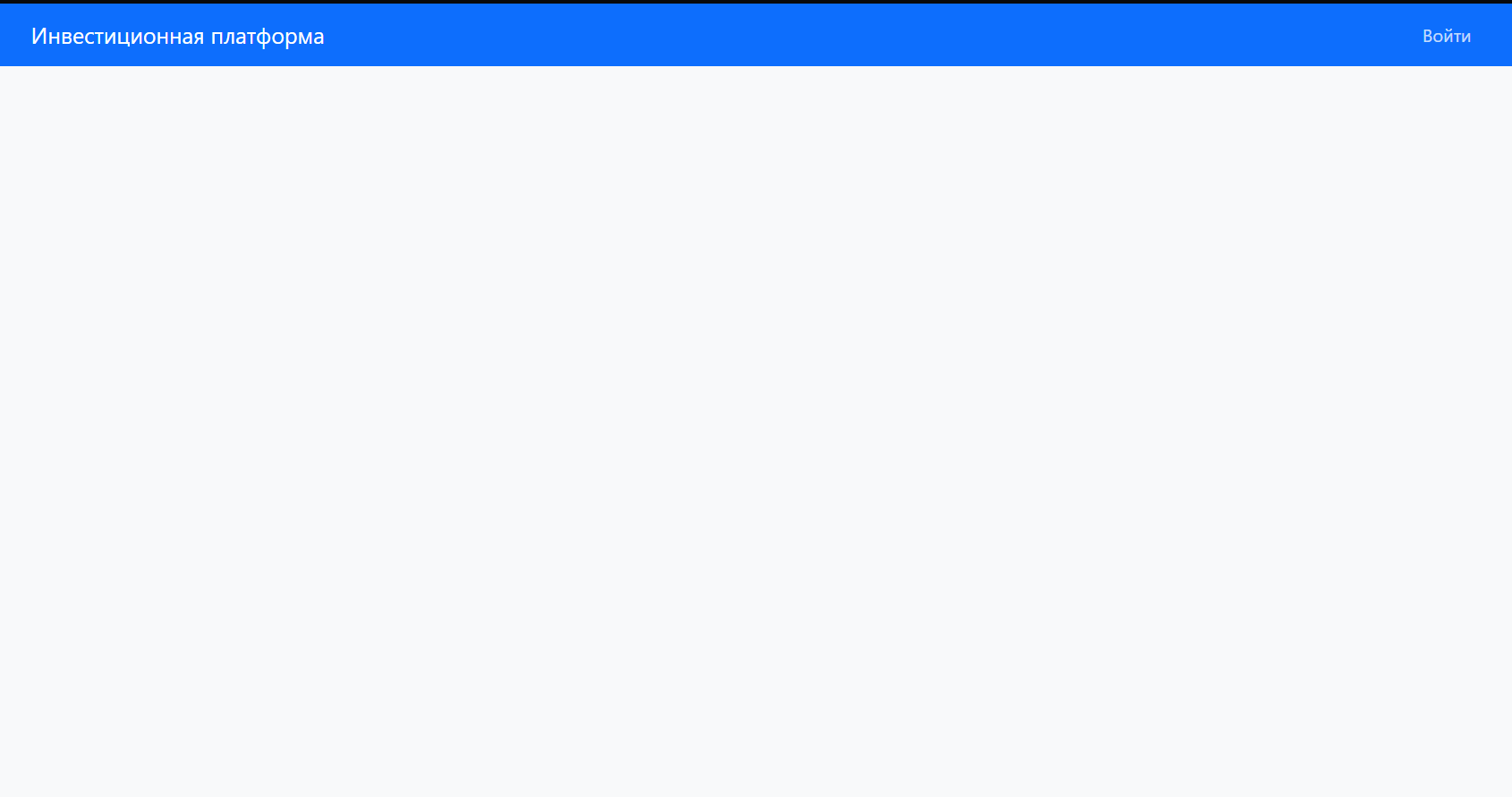


Рисунок 10 – Страница не загружается при нажатии на кнопку “Выйти”

ПРИЛОЖЕНИЕ А

Основной файл App.js:

import React from 'react';

import { BrowserRouter as Router, Routes, Route, Navigate } from 'react-router-dom';

import 'bootstrap/dist/css/bootstrap.min.css';

import './App.css';

// Компоненты

import Navbar from './components/Navbar';

import PrivateRoute from './components/PrivateRoute';

// Страницы

import Login from './pages/Login';

import SharesList from './pages/SharesList';

import Portfolio from './pages/Portfolio';

import Analytics from './pages/Analytics';

import NotFound from './pages/NotFound';

import Register from './pages/Register';

// Сервисы

import authService from './api/authService';

function App() {

  return (

    <Router>

      <div className="app">

        <Navbar />

        <main className="main-content">

          <Routes>

            <Route path="/login" element={

              authService.isAuthenticated() ?

                <Navigate to="/shares" /> :

                <Login />

            } />

            <Route path="/register" element={

              authService.isAuthenticated() ?

                <Navigate to="/shares" /> :

                <Register />

            } />

            <Route path="/shares" element={

              <PrivateRoute>

                <SharesList />

              </PrivateRoute>

            } />

            <Route path="/portfolio" element={

              <PrivateRoute>

                <Portfolio />

              </PrivateRoute>

            } />

            <Route path="/analytics" element={

              <PrivateRoute>

                <Analytics />

              </PrivateRoute>

            } />

            <Route path="/" element={<Navigate to="/shares" />} />

            <Route path="\*" element={<NotFound />} />

          </Routes>

        </main>

      </div>

    </Router>

  );

}

export default App;

ПРИЛОЖЕНИЕ Б

Файл authService.js для аутентификации.

import axios from './axiosConfig';

import { jwtDecode } from 'jwt-decode';

const login = async (username, password) => {

  try {

    const response = await axios.post('/auth/login', {

      login: username,

      password: password

    });

    const decoded = jwtDecode(response.data.accessToken);

    console.log("user: ", decoded.userId, "token: ", response.data.accessToken);

    if (response.data.accessToken) {

      localStorage.setItem('token', response.data.accessToken);

      localStorage.setItem('user', decoded.userId);//JSON.stringify(decoded.userId));

    }

    console.log("successful auth");

    return { user: decoded.userId, token: response.data.accessToken};

  } catch (error) {

    console.log(error.response);

    throw error;

  }

};

const logout = () => {

  //console.log("logout");

  localStorage.removeItem('token');

  localStorage.removeItem('user');

  //Navigate()

};

const getCurrentUser = () => {

  return localStorage.getItem('user');

};

const isAuthenticated = () => {

  //console.log("isauth ", localStorage.getItem('token'));

  return !!localStorage.getItem('token');

};

const register = async (login, email, password) => {

  try {

    const response = await axios.post('/auth/register', {

      email: email,

      login: login,

      password: password

    });

    return response.data;

  } catch (error) {

    console.log(error.response);

    throw error;

  }

};

const authService = {

  login,

  logout,

  register,

  getCurrentUser,

  isAuthenticated

};

export default authService;

ПРИЛОЖЕНИЕ В

Код для страницы портфолио Portfolio.js:

// src/pages/Portfolio.js

import React, { useState, useEffect } from 'react';

import shareService from '../api/shareService';

import ShareDetails from '../components/ShareDetails';

import { PieChart, Pie, Cell, ResponsiveContainer, Legend, Tooltip } from 'recharts';

const COLORS = ['#0088FE', '#00C49F', '#FFBB28', '#FF8042', '#8884d8', '#82ca9d', '#ffc658'];

const Portfolio = () => {

  const [portfolio, setPortfolio] = useState(null);

  const [loading, setLoading] = useState(true);

  const [error, setError] = useState('');

  const [selectedShare, setSelectedShare] = useState(null);

  useEffect(() => {

    const fetchPortfolio = async () => {

      try {

        setLoading(true);

        const data = await shareService.getUserPortfolio();

        console.log("portfolio data: ", data);

        setPortfolio(data);

        setError('');

      } catch (error) {

        console.error('Ошибка при загрузке портфеля:', error);

        setError('Не удалось загрузить ваш инвестиционный портфель. Попробуйте позже.');

      } finally {

        setLoading(false);

      }

    };

    fetchPortfolio();

  }, []);

  const handleShowDetails = (share) => {

    setSelectedShare(share);

  };

  const handleCloseDetails = () => {

    setSelectedShare(null);

  };

  const prepareChartData = () => {

    if (!portfolio || !portfolio.length) return [];

    return portfolio.map(share => ({

      name: share.ticker,

      value: share.nominal \* 1//share.quantity

    }));

  };

  if (portfolio) {

      portfolio.forEach(share => {

      share.nominal = Math.floor(Math.random() \* 100);

    });

    portfolio.nominal = portfolio.reduce((sum, share) => sum + share.nominal, 0);

  }

  if (loading) {

    return (

      <div className="container mt-5 text-center">

        <div className="spinner-border" role="status">

          <span className="visually-hidden">Загрузка...</span>

        </div>

        <p className="mt-2">Загрузка вашего портфеля...</p>

      </div>

    );

  }

  if (error) {

    return (

      <div className="container mt-5">

        <div className="alert alert-danger" role="alert">

          {error}

        </div>

      </div>

    );

  }

  if (!portfolio) {

    return (

      <div className="container mt-5">

        <div className="alert alert-info" role="alert">

          В вашем портфеле пока нет акций. Перейдите на страницу "Акции", чтобы добавить акции в портфель.

        </div>

      </div>

    );

  }

  console.log("portfolio: ", portfolio);

  return (

    <div className="container mt-4">

      <h2 className="mb-4">Мой инвестиционный портфель</h2>

      <div className="row mb-4">

        <div className="col-md-4">

          <div className="card">

            <div className="card-body">

              <h5 className="card-title">Общая стоимость</h5>

              <h2 className="card-text">${portfolio.nominal}</h2>

            </div>

          </div>

        </div>

        <div className="col-md-4">

          <div className="card">

            <div className="card-body">

              <h5 className="card-title">Прибыль/Убыток</h5>

              <h2 className={`card-text ${(portfolio.nominal - 0) >= 0 ? 'text-success' : 'text-danger'}`}>

                ${portfolio.nominal} ({0}%)

              </h2>

            </div>

          </div>

        </div>

        <div className="col-md-4">

          <div className="card">

            <div className="card-body">

              <h5 className="card-title">Количество акций</h5>

              <h2 className="card-text">{portfolio.length}</h2>

            </div>

          </div>

        </div>

      </div>

      <div className="row">

        <div className="col-md-7">

          <div className="card">

            <div className="card-header">

              <h5 className="mb-0">Акции в вашем портфеле</h5>

            </div>

            <div className="card-body">

              <div className="table-responsive">

                <table className="table table-hover">

                  <thead>

                    <tr>

                      <th>Тикер</th>

                      <th>Название</th>

                      <th>Количество</th>

                      <th>Цена</th>

                      <th>Всего</th>

                      <th>Действия</th>

                    </tr>

                  </thead>

                  <tbody>

                    {portfolio.map(share => (

                      <tr key={share.id}>

                        <td>{share.ticker}</td>

                        <td>{share.name}</td>

                        <td>{1}</td>

                        <td>${share.nominal}</td>

                        <td>${share.nominal\*1}</td>

                        <td>

                          <button

                            className="btn btn-sm btn-primary"

                            onClick={() => handleShowDetails(share)}

                          >

                            Детали

                          </button>

                        </td>

                      </tr>

                    ))}

                  </tbody>

                </table>

              </div>

            </div>

          </div>

        </div>

        <div className="col-md-5">

          {selectedShare ? (

            <ShareDetails

              share={selectedShare}

              onClose={handleCloseDetails}

            />

          ) : (

            <div className="card">

              <div className="card-header">

                <h5 className="mb-0">Распределение портфеля</h5>

              </div>

              <div className="card-body">

                <ResponsiveContainer width="100%" height={300}>

                  <PieChart>

                    <Pie

                      data={prepareChartData()}

                      cx="50%"

                      cy="50%"

                      outerRadius={80}

                      fill="#8884d8"

                      dataKey="value"

                      label={({name, percent}) => `${name} ${(percent \* 100).toFixed(0)}%`}

                    >

                      {prepareChartData().map((entry, index) => (

                        <Cell key={`cell-${index}`} fill={COLORS[index % COLORS.length]} />

                      ))}

                    </Pie>

                    <Tooltip formatter={(value) => [`$${value.toFixed(2)}`, 'Стоимость']} />

                    <Legend />

                  </PieChart>

                </ResponsiveContainer>

              </div>

            </div>

          )}

        </div>

      </div>

    </div>

  );

};

export default Portfolio;