Санкт-Петербургский государственный университет

Кафедра системного программирования

Разработка инструмента для работы с индивидуальными графиками

Каргин Глеб Павлович, группа 21.Б10-мм

Научный руководитель: к.т.н. Ю. В. Литвинов, старший преподаватель кафедры системного программирования

Санкт-Петербург
2024

Введение

- Дополнительная попытка для студентов, пропустивших по уважительной причине зачёт или экзамен
- Проблемы преподавателей:
  - Отправка индивидуальных графиков по частям
  - Индивидуальные графики в формате текстовых документов
  - Из-за огромного количества студентов-должников велика вероятность упустить кого-то
- Проблемы учебного отдела:
  - Необходимость вручную проверять расписание преподавателей, студентов и аудиторий с помощью Timetable[1]

[1] `https://timetable.spbu.ru`

Требования к проекту

- Веб-приложение для составления и просмотра расписания индивидуальных графиков
- Реализация должна быть на платформе .NET и языках программирования C# или F#
- Клиентская часть должна разрабатываться на TypeScript, как одностраничное приложение
- При составлении графика должно подгружаться расписание студента, преподавателя, а также аудитории
- Возможность экспорта записей в файл с распоряжением
- Возможность отображать все записи или фильтровать их по преподавателям или студентам

Существующие решения

- Системы для совместной работы и отслеживания задач (например, Jira[2])
  - Большое количество плагинов
  - Возможность экспортировать мероприятия
  - Отсутствие автоматического формирования файла с распоряжением
- Сервисы для планирования мероприятий (например, Яндекс.Календарь[3])
  - Возможность учитывать занятость участников мероприятий
  - Возможность экспортировать мероприятия и проводить видеовстречи
  - Отсутствие пользовательской фильтрации событий
- Разработка плагина не сильно проще реализации системы с нуля

[2] https://www.atlassian.com/ru/software/jira

[3] https://calendar.yandex.ru

Постановка задачи

Цель: проектирование и реализация веб-приложения для составления и просмотра расписания индивидуальных графиков

Задачи:

- Осенний семестр
  1. Провести обзор аналогичных инструментов
  2. Спроектировать общую архитектуру сервиса и базы данных
  3. Спроектировать пользовательский интерфейс
  4. Реализовать инструмент для разбора .docx файлов
  5. Реализовать API сервиса
- Весенний семестр
  1. Реализовать преподавательскую часть
  2. Реализовать часть учебного отдела
  3. Реализовать аутентификацию и авторизацию пользователей
  4. Реализовать пользовательский интерфейс
  5. Провести тестирование реализованного инструмента

Используемые технологии

- C#
- ASP.NET
- Entity Framework — ORM-фреймворк для запросов к базе данных
- PostgreSQL
- Docker
- TypeScript
- React
- Axios — для работы с HTTP-запросами

Проектирование

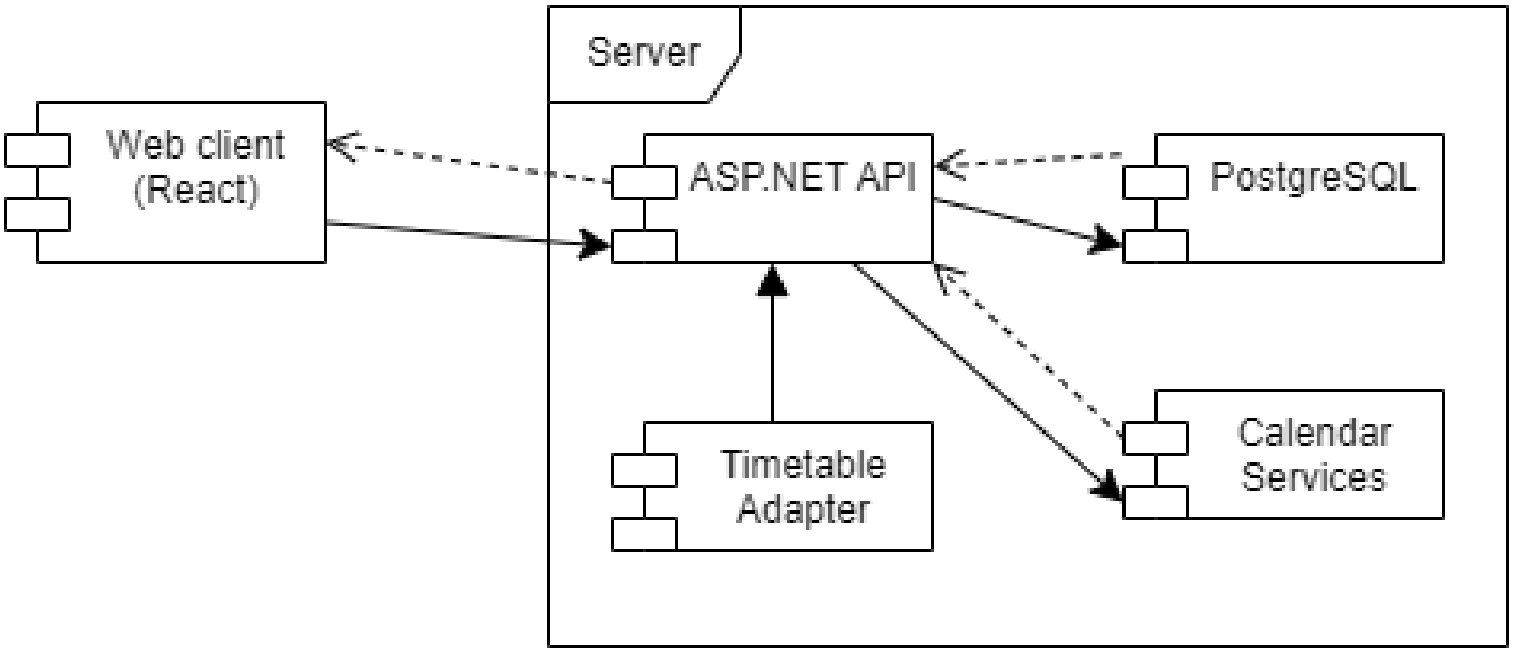

Рис.: Архитектура веб-приложения

Проектирование

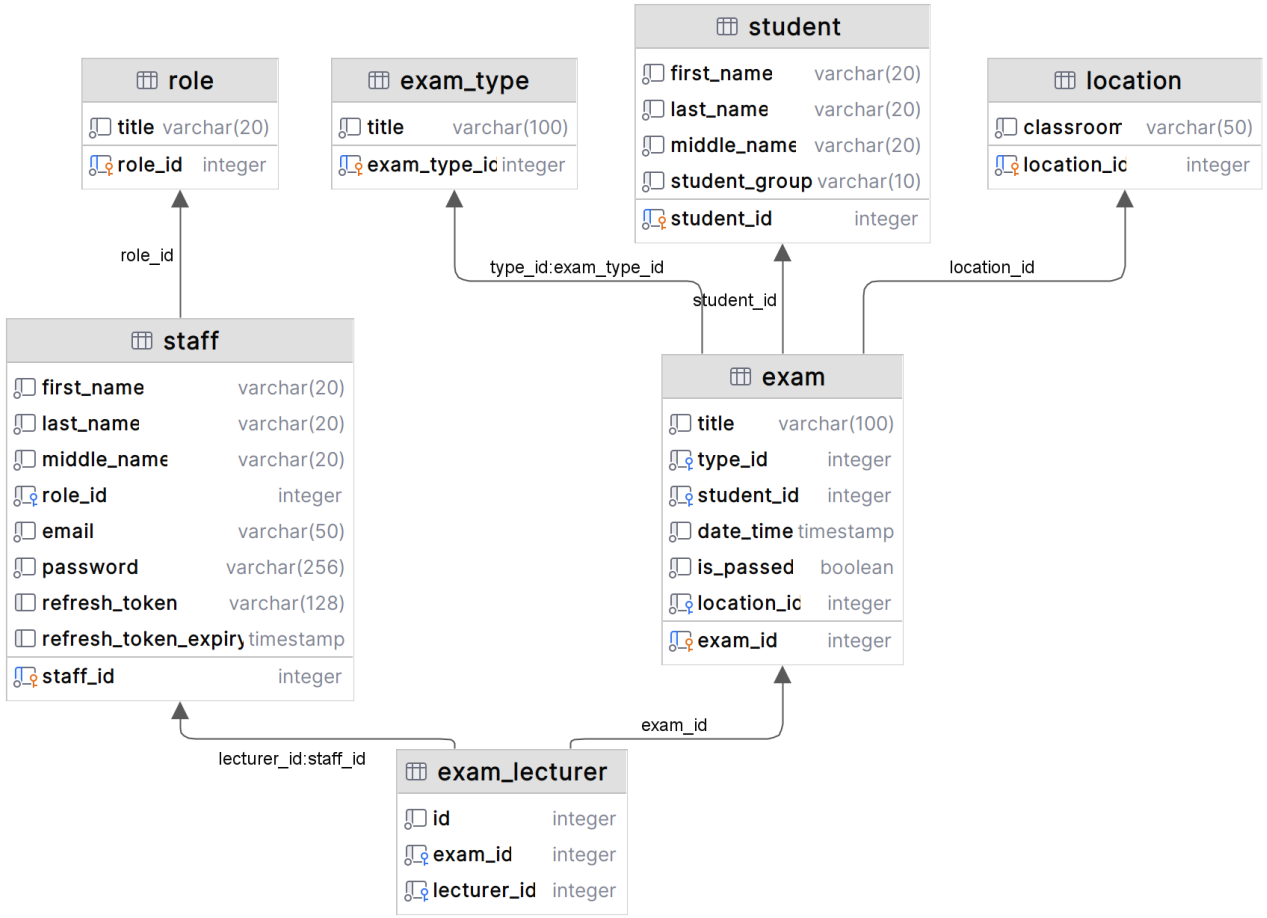

Рис.: Схема базы данных

Проектирование

Рис.: Мокап страницы отображения расписания

Реализация

- Было принято решение начать разработку с преподавательской части
- Консольное приложение для разбора .docx файлов
  - Принимает путь до таблицы, путь до файла формата .docx с индивидуальными графиками
  - Файл формата .docx разбивается на абзацы, содержащие данные студента и таблицу графика
  - Данные из документа извлекаются в объект DataTable
  - Таблица заполняется новыми записями и сохраняется

Реализация

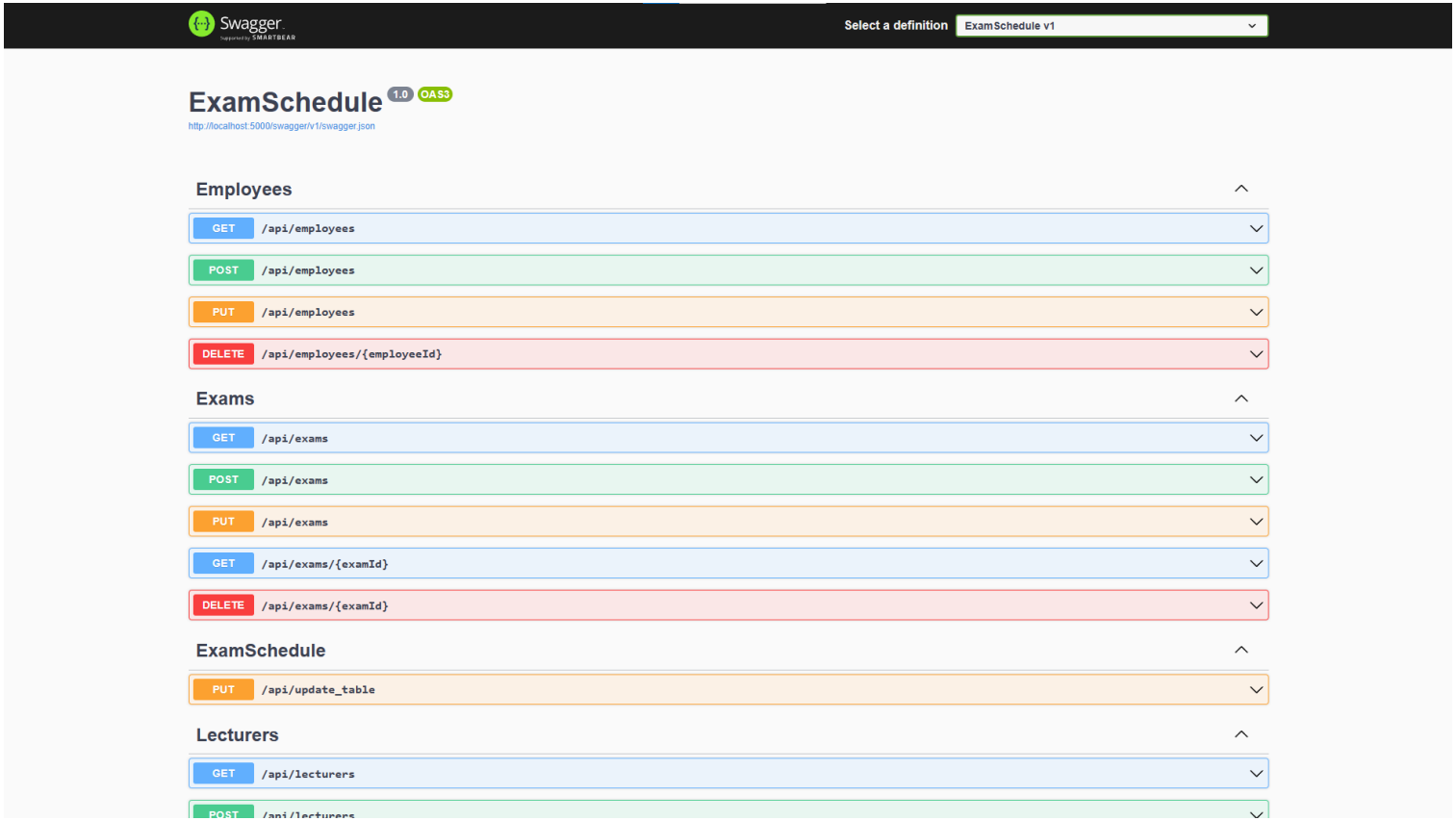

Рис.: Документация API приложения в Swagger[4]

[4] https://docs.swagger.io/spec.html

Реализация

- Авторизация и аутентификация с использованием JWT-токенов
- Access и refresh токены
- Пути «/login» и «/refresh»

Реализация

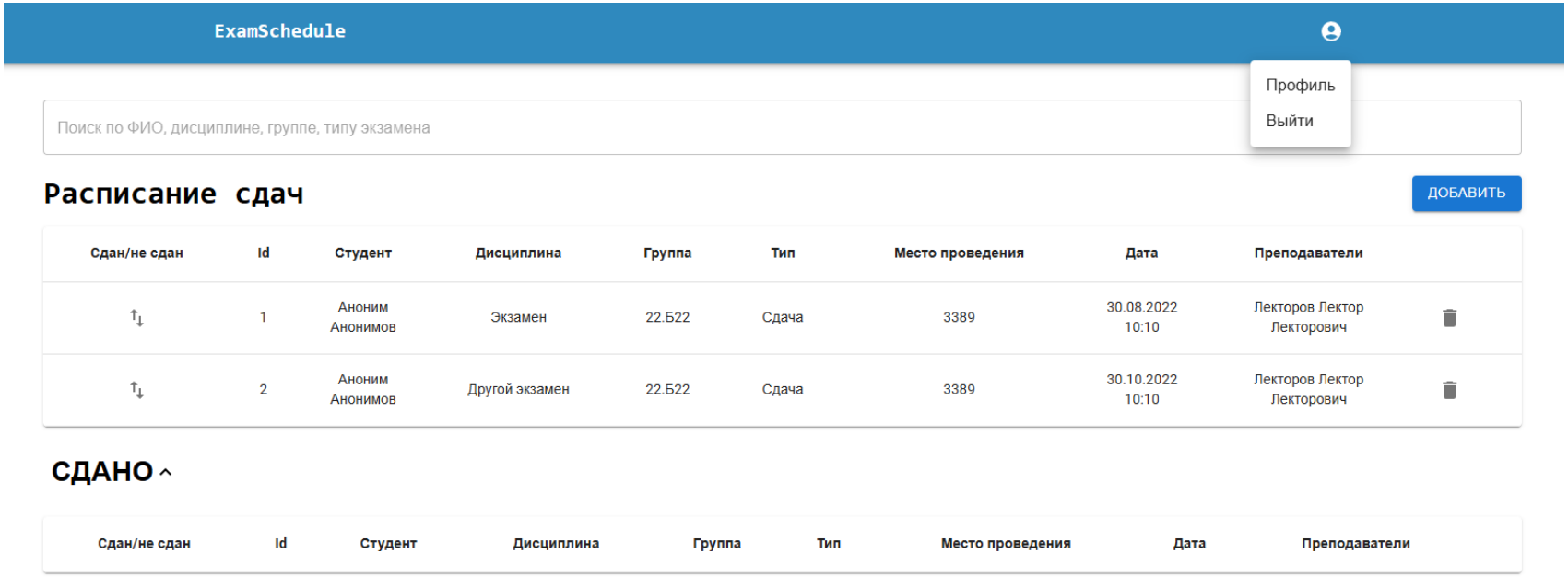

Рис.: Пользовательский интерфейс страницы отображения записей

Реализация

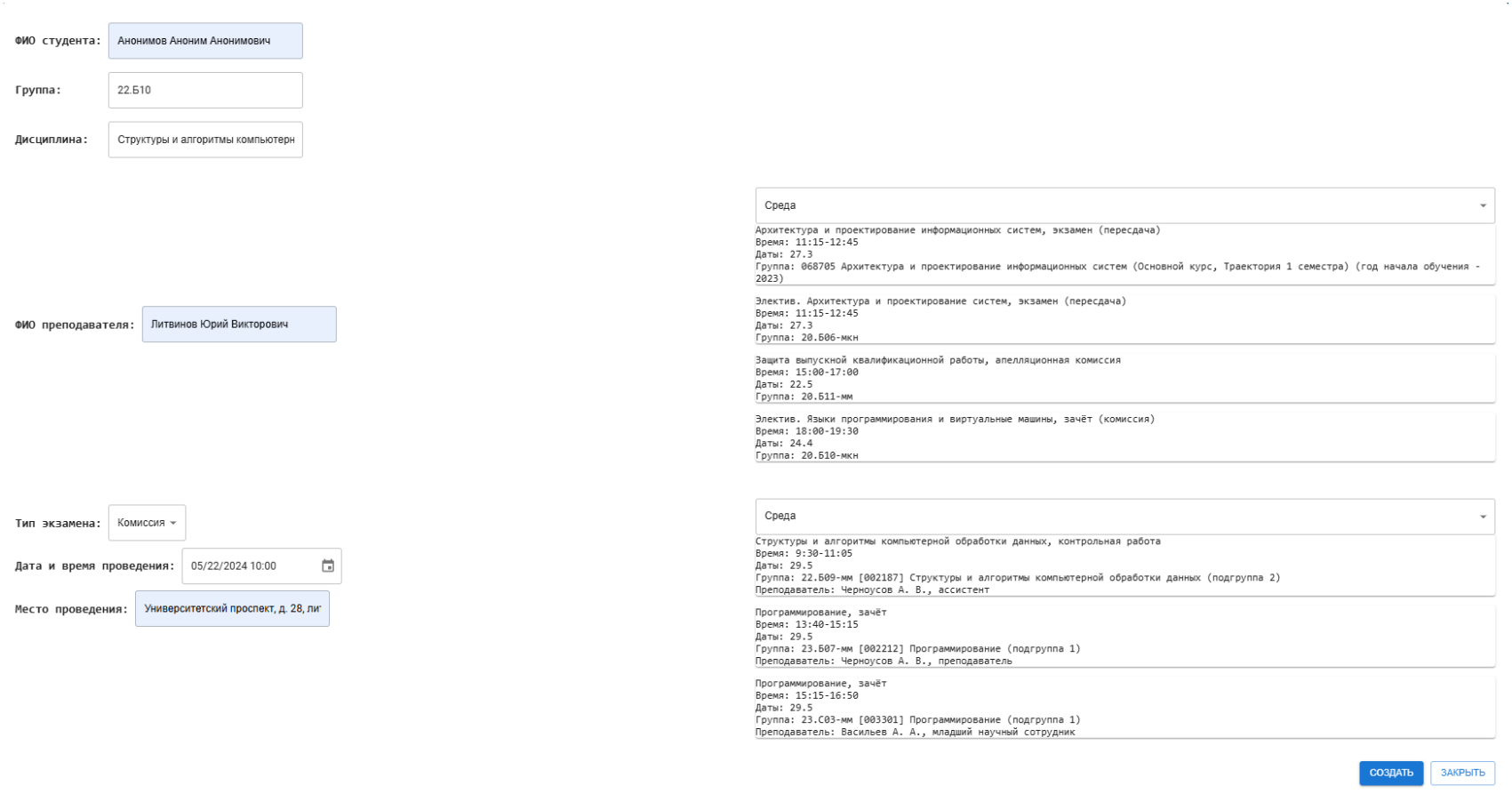

Рис.: Диалоговое окно создания записи

Эксперимент

- Для обеспечения качества были проведены модульное и сквозное тестирование
- Для модульного тестирования использовалась библиотека NUnit[5], для сквозного — Playwright[6]

[5] https://nunit.org
[6] https://playwright.dev

Эксперимент

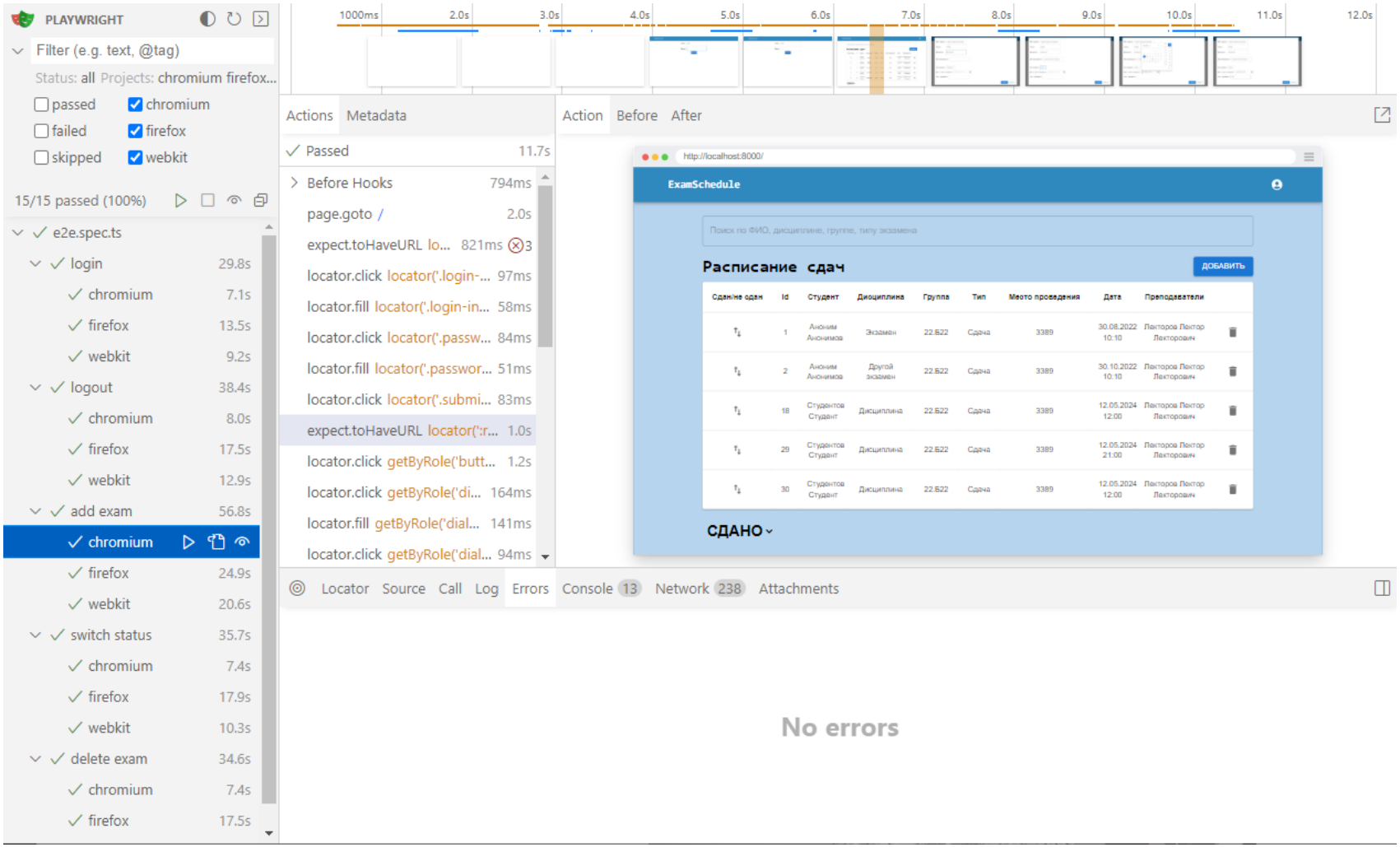

Рис.: Сквозное тестирование веб-приложения

Результаты

- Проведен обзор аналогичных инструментов
- Спроектирована общая архитектура сервиса и базы данных
- Спроектирован пользовательский интерфейс
- Реализован инструмент для разбора .docx файлов
- Реализован API сервиса
- Реализована преподавательская часть
- Реализована часть учебного отдела
- Реализована аутентификация и авторизация пользователей
- Реализован пользовательский интерфейс
- Проведено тестирование реализованного инструмента
- Код доступен в репозитории GitHub[7]

[7] https://github.com/yurii-litvinov/ExamSchedule