

# МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ  
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)



ИНСТИТУТ №8  
«ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И ПРИКЛАДНАЯ  
МАТЕМАТИКА»

КАФЕДРА 813  
«КОМПЬЮТЕРНАЯ МАТЕМАТИКА»

**Курсовой проект по дисциплине «Базы данных»**

**Тема: «Веб-приложение для тестирования»**

Студент: Василийев Дмитрий Олегович

Группа: М8О-310Б-18

Преподаватель: Романенков Александр Михайлович

Дата: 13 октября 2020 г.

Оценка: \_\_\_\_\_

Подпись преподавателя: \_\_\_\_\_

Подпись студента: \_\_\_\_\_

Москва 2020

# Содержание

<b>Аббревиатуры</b>	<b>3</b>
<b>1 Введение</b>	<b>4</b>
1.1 Формальные требования . . . . .	4
1.2 Клиентские приложения . . . . .	5
1.2.1 Тестирующая система . . . . .	5
1.2.2 Система управления . . . . .	5
1.3 Предметная область . . . . .	5
1.4 Стэк технологий . . . . .	5
1.5 Инструменты . . . . .	6
<b>2 Инфраструктура проекта</b>	<b>8</b>
2.1 Архитектура . . . . .	8
2.1.1 Связь логических компонент и применяемых технологий . . . . .	9
2.2 Сущности . . . . .	9
2.3 Сборка и запуск . . . . .	10
2.3.1 Development . . . . .	10
2.3.2 Production . . . . .	11
2.4 Деплоинг . . . . .	11
2.5 Организация работы с <i>Git</i> [5] . . . . .	11
2.5.1 Git Workflow . . . . .	11
2.5.2 Git hooks . . . . .	11
<b>3 Описание проекта</b>	<b>13</b>
3.1 Разработка дизайна . . . . .	13
3.2 Авторизации через JSON Web Token (JWT) . . . . .	13
3.2.1 Авторизация . . . . .	13
3.2.2 Аутентификация . . . . .	13
3.2.3 JSON Web Token (JWT) . . . . .	13
3.2.4 Реализация . . . . .	14
3.3 Схема базы данных . . . . .	15
<b>4 Заключение</b>	<b>16</b>
4.1 Недостатки . . . . .	16

А Визуализации структуры проекта	17
В Код проекта	21

## Аббревиатуры

**API** Applicatoin Programming Interface. 5, 6

**BEM** Block Element Modifier. 16

**JSON** JavaScript Object Notaion. 13, 16

**JWT** JSON Web Token. 1, 13–15

**MVC** Model View Controller. 8

**NoSQL** Not only SQL. 16

**ORM** Object-Relational Mapping. 6

**REST** Representational State Transfer. 5

**SPA** Single Page Application. 5, 16

# 1 Введение

## 1.1 Формальные требования

1. Необходимо выбрать предметную область для создания базы данных. Выбранная предметная область должна быть уникальной для всего потока, а не только в рамках учебной группы.
2. Необходимо описать таблицы и их назначение. Выполнить проектирование логической структуры базы данных. Описать схему базы данных. Все реальные таблицы должны иметь 3 нормальную форму или выше. База данных должна иметь минимум 5 таблиц.
3. Необходимо разработать два клиентских приложения для доступа к базе данных. Данные приложения должны быть написаны на двух разных языках программирования и иметь разный интерфейс (например, классическое оконное приложение и web-приложение). Выбор языков программирования произволен.
4. Необходимо организовать различные роли пользователей и права доступа к данным. Далее, необходимо реализовать возможность создания архивных копий и восстановления данных из клиентского приложения.
5. При разработке базы данных следует организовать логику обработки данных не на стороне клиента, а, например, на стороне сервера, базы данных, клиентские приложения служат только для представления данных и тривиальной обработки данных.
6. Ваша база данных должна иметь представления, триггеры и хранимые процедуры, причем все эти объекты должны быть осмысленны, а их использование оправдано.
7. При показе вашего проекта необходимо уметь демонстрировать таблицы, представления, триггеры и хранимые процедуры базы данных, внешние ключи, ограничения целостности и др. В клиентских приложениях уметь демонстрировать подключение к базе данных, основные режимы работы с данными (просмотр, редактирование, обновление ...)
8. Необходимо реализовать корректную обработку различного рода ошибок, которые могут возникать при работе с базой данных.

## 1.2 Клиентские приложения

Оба клиента будут SPA приложениями, которые общаются с сервером посредством REST API.

### 1.2.1 Тестирующая система

Данное приложение даёт возможность пользователю:

- создавать, редактировать, комбинировать, удалять тесты.
- рассылать приглашения на прохождения тестов.

### 1.2.2 Система управления

Данное приложение доступно только для администратора. Оно даёт ему следующие возможности:

- рассылать email-рассылку.
- банить тесты, пользователей.
- удалять тесты, пользователей.

## 1.3 Предметная область

Область применения данного приложения универсальна. Можно использовать тестирование на сотрудниках, школьниках, студентах и так далее.

## 1.4 Стэк технологий

- *JavaScript* [8] — мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией стандарта *ECMAScript* [2].
- *React* [14] — *JavaScript* [8] библиотека для создания пользовательских интерфейсов.
- *Redux* [16] — контейнер состояния для *JavaScript* [8] приложения.
- *React Router* [15] — набор навигационных компонентов.
- *SCSS* [17] — препроцессор, который расширяет *CSS* [1].

- *CSS* [1] — формальный язык описания внешнего вида документа, написанного с использованием языка разметки.
- *HTML* [7] — гипертекстовый язык разметки.
- *Node.js* [11] — среда выполнения JavaScript, созданная на основе движка Chrome V8 JavaScript.
- *Express* [4] — минимальный и гибкий *Node.js* [11] фреймворк для создания веб-приложений.
- *Python* [13] — язык программирования, который позволяет быстро работать и более эффективно интегрировать системы.
- *PostgreSQL* [12] — объектно-реляционная база данных с открытым исходным кодом.
- *Sequelize* [19] — *Node.js* [11] ORM на основе обещаний для Postgres *PostgreSQL* [12].

## 1.5 Инструменты

- *Git* [5] — система контроля версий.
- Postman — платформа совместной разработки API
- IDEs: — интегрированная среда разработки.
  - WebStorm
  - DataGrip
- Линтеры — программы, которые следят за качеством кода.
  - *ESLint* [3] — проверяет качество кода на *JavaScript* [8].
  - *Stylelint* [20] — проверяет качество кода на *SCSS* [17], *CSS* [1].
- Тестирующие фреймворки:
  - *Jest* [9] — среда тестирования *JavaScript* [8] с упором на простоту.
  - *Selenium with Python* [18] — предоставляют простой API для написания тестов с использованием Selenium WebDriver.

*Webpack* [21] — сборщик статических модулей для современных *JavaScript* [8] приложений.



## 2 Инфраструктура проекта

### 2.1 Архитектура

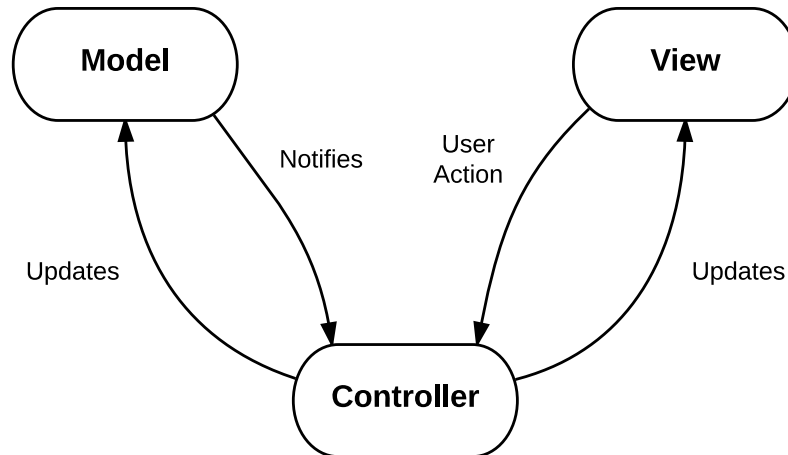


Рис. 1: Визуализация архитектуры MVC

За основу берётся архитектурный паттерн MVC. Он предполагает разделение данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: Модель, Представление и Контроллер – таким образом, что модификация каждого компонента может осуществляться независимо.

- **Модель** — предоставляет собой объектную модель некой предметной области, включает в себя данные и методы работы с этими данными, реагирует на запросы из контроллера, возвращая данные и/или изменяя своё состояние. При этом модель не содержит в себе информации о способах визуализации данных или форматах их представления, а также не взаимодействует с пользователем напрямую.
- **Представление** — отвечает за отображение информации. Одни и те же данные могут представляться различными способами и в различных форматах. Например, коллекцию объектов при помощи разных представлений можно представить на уровне пользовательского интерфейса как в табличном виде, так и списком.
- **Контроллер** — обеспечивает связь между пользователем и системой, использует модель и представление для реализации необходимой реакции на действия пользователя. Как правило, на уровне контроллера осуществляется фильтрация полученных данных и авторизация — проверяются права пользователя на выполнение действий или получение информации.

### 2.1.1 Связь логических компонентов и применяемых технологий

- **Model** — *Sequelize* [19] для сервера и *Redux* [16] для клиента.
- **View** — *React* [14]
- **Controller** — *Express* [4]

## 2.2 Сущности

Всегда перед проектированием проекта описывают сущности и их атрибуты. На основе данной информации будет строиться база данных. В моём случае они следующие:

Сущность	Атрибуты
Пользователь	<ul style="list-style-type: none"><li>• ID пользователя</li><li>• Роли</li><li>• Логин</li><li>• Пароль</li><li>• Email</li><li>• Дата создания</li></ul>
Тест	<ul style="list-style-type: none"><li>• ID теста</li><li>• Название</li><li>• Описание</li><li>• Теги</li><li>• Контент</li><li>• Ответы</li><li>• Дата создания</li><li>• Дата последнего изменения</li></ul>
Тег	<ul style="list-style-type: none"><li>• ID тега</li><li>• Название</li></ul>

Попытка	<ul style="list-style-type: none"> <li>• ID попытки</li> <li>• ID пользователя</li> <li>• ID теста</li> <li>• Результат</li> <li>• Ответы пользователя</li> <li>• Дата прохождения</li> <li>• Длительность прохождения попытки</li> </ul>
Роль	<ul style="list-style-type: none"> <li>• ID роли</li> <li>• Название</li> </ul>

Таблица 1: Описание сущностей и атрибутов

## 2.3 Сборка и запуск

Все процессы отвечающие за сборку и запуск приложения я разделил на подзадачи. Каждая такая подзадача является npm скриптом. Они все описываются в файле package.json. Также среди данных скриптов можно выделить две группы – Development и Production.

### 2.3.1 Development

Скрипты из данной группы отвечают за то, чтобы приложение можно было запускать в режиме разработки, а именно:

1. сборка клиентской части не занимало слишком много времени
2. клиент пересобирался при изменении какого-либо файла
3. сервер перезапускался при изменении кода серверверной части
4. в браузере были доступны source map

### 2.3.2 Production

Скрипты из данной группы отвечают за то, чтобы приложение можно было запускать в режиме с максимальными оптимизациями, а именно:

1. минификация статических файлов
2. оптимизация работы библиотек
3. сборка серверной части

## 2.4 Деплоинг

Приложение разворачивается в системе *Heroku* [6]. Там же работает СУБД.

## 2.5 Организация работы с *Git* [5]

### 2.5.1 Git Workflow

Для организации работы с системой контроля версий в проекте используется подход Git Workflow. Он нужен для согласованного и продуктивного выполнения работы.

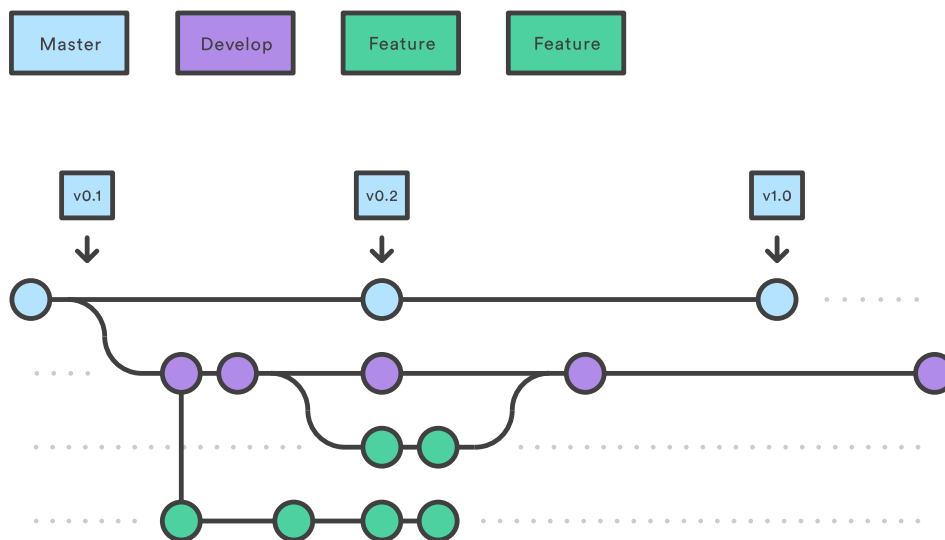


Рис. 2: Пример использования подхода Git Workflow

### 2.5.2 Git hooks

Чтобы в репозитории хранился код, который проходит проверки линтеров и тестовых фреймворков, нужно использовать Git Hooks. Они позволяют обработать события pre-commit, pre-push, post-commit и так далее.

Есть удобный пакет в npm – husky. Он позволяет определить в package.json обработку событий. В моём проекте нужно, чтобы на событие pre-commit выполняли проверки линтеры, а потом при успешном результате исполнялись unit-тесты. Также необходимо запускать selenium-тесты при событии pre-push.

---

```
1 {  
2   "hooks": {  
3     "pre-commit": "yarn es-lint && yarn style-lint && yarn test",  
4     "pre-push": "./venv/bin/pytest tests"  
5   }  
6 }
```

---

Listing 1: Настройки для Git Hooks

## 3 Описание проекта

### 3.1 Разработка дизайна

Так как я не дизайнер, то мне нужно оперировать концептами и эскизами интерфейса. Поэтому вначале я сделал макет страниц и связь между ними.

### 3.2 Авторизации через JSON Web Token (JWT)

#### 3.2.1 Авторизация

**Авторизация** — это процесс предоставления определённому лицу или группе лиц прав на выполнение определённых действий. Также сюда входит проверка данных, прав при попытке выполнения этих действий.

#### 3.2.2 Аутентификация

**Аутентификация** — процедура проверки подлинности данных.

#### 3.2.3 JSON Web Token (JWT)

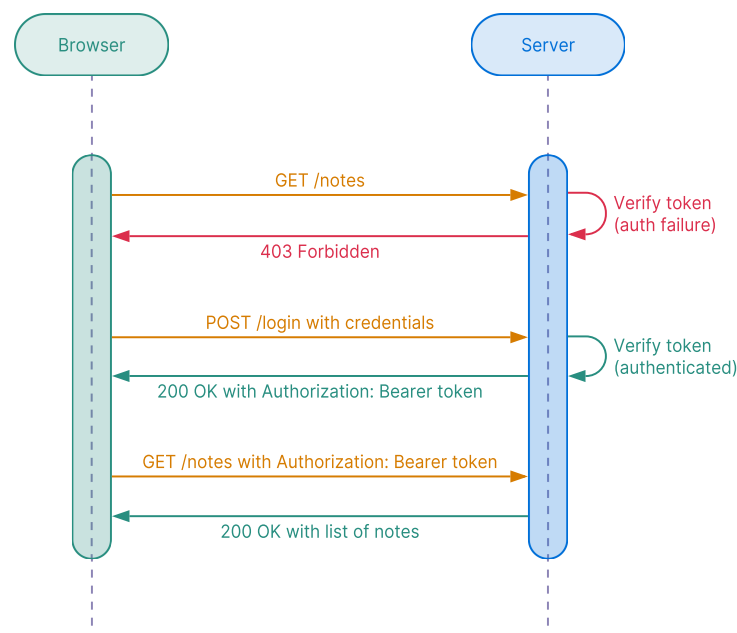


Рис. 3: Демонстрация работы JWT

**JSON Web Token (JWT)** — это открытый стандарт (RFC 7519), который определяет способ для безопасной передачи информации между сторонами с помощью JSON объектов. Эту информацию можно проверить, потому что она имеет цифровую подпись.

Вот несколько сценариев, в которых полезен JWT:

- **Авторизация** — это наиболее распространенный сценарий использования JWT. После того, как пользователь вошел в систему, каждый последующий запрос будет включать JWT, позволяя пользователю получать доступ к маршрутам, службам и ресурсам, разрешенным с помощью этого токена.
- **Обмен информацией** — JWT хороший способ безопасной передачи информации между сторонами. Поскольку JWT могут быть подписаны, например, с использованием пар открытого и закрытого ключей, вы можете быть уверены, что отправители являются теми, кем они себя называют. Кроме того, поскольку подпись рассчитывается с использованием **заголовка** и **полезных данных**, вы также можете убедиться, что содержимое не было изменено.

JWT состоит из следующих частей:

- **Заголовок** — содержит информацию о том, как должна вычисляться подпись. Обычно состоит из двух частей: типа токена, которым является JWT, и используемого алгоритма подписи, такого как HMAC SHA256 или RSA.
- **Полезные данные** — это данные, которые хранятся внутри JWT. Они также называют JWT-claims (заявки). Список доступных полей для JWT доступен на Wiki.
- **Подпись** — используется для проверки того, что сообщение не было изменено в процессе. В компактной форме JWT является строкой, которая состоит из трех частей, разделенных точками. Псевдокод вычисления подписи:

---

```
1 SECRET_KEY = 'some string';
2 unsignedToken = encodeBase64Url(header) + '.' + encodeBase64Url(payload)
3 signature = SHA256(unsignedToken, SECRET_KEY);
4
5 // собираем всё вместе
6 jwt = encodeBase64Url(header) + '.' + encodeBase64Url(payload) + '.' + encodeBase64Url(signature);
```

---

### 3.2.4 Реализация

Авторизация проходит следующим образом:

1. Пользователь делает `POST /api/signup` запрос на регистрацию. Если всё нормально, то в базе данных создаётся запись с данными пользователя.
2. Пользователь делает `POST /api/signin` запрос на аутентификацию. Если данные

верные, то высылается JWT вместе с состоянием пользователя (логин, почта). Когда ответ с сервера получен, то JWT сохраняется в `localStorage` (долговременное хранилище), а состояние передается в глобальное *Redux* [16] хранилище.

3. После того, как состояние глобального хранилища обновилось, приложение обновляет интерфейс.

Если перезагрузить веб-страницу, то *Redux* [16] хранилище обнуляется. Поэтому нам нужно сделать следующее: при запуске приложения проверять на валидность JWT, который лежит в `localStorage`. Это делается через `post /api/init` запрос. Если токен валидный, то переавторизовываем пользователя. Иначе перенаправляем на главную страницу.

Далее этот токен будет использоваться для доступа к защищённым ресурсам.

### 3.3 Схема базы данных

Схема базы данных



## 4 Заключение

Благодаря данному курсовому проекту, я поверхностно освоил разработку SPA приложений с помощью библиотеки *React* [14] и фреймворка *Express* [4].

### 4.1 Недостатки

Подводя итоги, мне бы хотелось перечислить вещи, на которые я буду обращать внимание при разработке следующих проектов:

- Использование CI/CD.
- Использование методологий в вёрстке. Например, Block Element Modifier (BEM). Её разработали внутри компании Яндекс. У них есть свой стек технологий под данную методологию, который облегчает разработку клиентской части.
- Использование NoSQL баз данных вместе с реляционными. Хранить JSON в таблице плохо, поэтому для этой задачи подходит *MongoDB* [10].
- Разделение клиентского кода на чанки.
- Микросервисная архитектура.
- Использование TypeScript или Flow. Во время разработки я отказался от TypeScript из-за того, что надо очень много времени тратить на type-hinting.

## A Визуализации структуры проекта

```
/db-course-project-app
--- .gitignore
--- package.json
--- .babelrc
--- jest.config.js
--- requirements.txt
--- .eslintignore
--- yarn.lock
--- .env
--- README.md
--- LICENSE
--- /util
----- nodemon.json
--- /tests
----- test_smoke.py
--- /src
----- .stylelintrc
----- config.js
----- index.js
----- webpack.config.js
----- .eslintrc.js
----- /client
----- main.jsx
----- admin.jsx
----- /tests
----- smoke.test.js
----- /apps
----- /main
----- App.jsx
----- /pages
----- /ProfileSettings
----- style.scss
```

```
----- ProfileSettings.jsx
----- index.js
----- /SignUp
----- style.scss
----- index.js
----- SignUp.jsx
----- /TestEditor
----- /Profile
----- style.scss
----- Profile.jsx
----- index.js
----- /ProfileAttempts
----- style.scss
----- ProfileAttempts.jsx
----- index.js
----- /Test
----- /Home
----- style.scss
----- index.js
----- Home.jsx
----- /Login
----- style.scss
----- Login.jsx
----- index.js
----- /components
----- /Header
----- style.scss
----- Header.jsx
----- index.js
----- /TestCard
----- style.scss
----- index.js
----- TestCard.jsx
----- /ListTestCards
```

```
----- index.js
----- ListTestCards.jsx
----- /Footer
----- style.scss
----- Footer.jsx
----- index.js
----- /services
----- /admin
----- Login.jsx
----- App.jsx
----- /routes
----- main.js
----- auth.js
----- /tests
----- smoke.test.js
----- /controllers
----- auth.js
----- /helpers
----- FormListErrors.js
----- /templates
----- admin.handlebars
----- index.handlebars
----- /layouts
----- main.handlebars
----- /partials
----- favicon.handlebars
----- meta.handlebars
----- /middlewares
----- checkToken.js
----- /models
----- UserRole.js
----- Role.js
----- /User
----- constraints.js
```

```
----- User.js
----- index.js
----- /services
--- /db
----- init_db.sql
```

## В Код проекта

db-course-project-app/package.json

```
1 {
2   "name": "db-course-project-app",
3   "version": "1.0.0",
4   "description": "Web-application for course project by Database.",
5   "main": "build/index.js",
6   "author": "Dmitry Vasiliev",
7   "scripts": {
8     "test": "jest",
9     "start-dev": "nodemon --config \"/util/nodemon.json"/",
10    "build": "rm -rf ./build/* && babel src -d build && webpack --config src/webpack.config.js
11    ↪ --mode=\"production\"",
12    "start": "node -r dotenv/config build/index.js",
13    "es-lint": "eslint . -c src/.eslintrc.js --ext \".jsx,js\"",
14    "style-lint": "stylelint --ignore-pattern src/client/tests --config src/.stylelintrc src/client/*",
15    "watch": "webpack --config src/webpack.config.js --watch"
16  },
17  "dependencies": {
18    "@fortawesome/fontawesome-svg-core": "^1.2.32",
19    "@fortawesome/free-brands-svg-icons": "^5.15.1",
20    "@fortawesome/free-regular-svg-icons": "^5.15.1",
21    "@fortawesome/free-solid-svg-icons": "^5.15.1",
22    "@fortawesome/react-fontawesome": "^0.1.11",
23    "bcrypt": "^5.0.0",
24    "bootstrap": "^4.5.0",
25    "compression": "^1.7.4",
26    "dotenv": "^8.2.0",
27    "express": "^4.17.1",
28    "express-handlebars": "^5.0.0",
29    "http-status-codes": "^1.4.0",
30    "jsonwebtoken": "^8.5.1",
31    "lodash": "^4.17.19",
32    "morgan": "^1.10.0",
33    "multer": "^1.4.2",
34    "pg": "^8.3.0",
35    "pg-hstore": "^2.3.3",
36    "prop-types": "^15.7.2",
37    "pug": "^3.0.0",
38    "react": "^16.13.1",
39    "react-bootstrap": "^1.2.2",
40    "react-dom": "^16.13.1",
41    "react-redux": "^7.2.1",
42    "react-router-bootstrap": "^0.25.0",
43    "react-router-dom": "^5.2.0",
44    "react-router-prop-types": "^1.0.5",
45    "redux": "^4.0.5",
46    "sequelize": "^6.3.3",
47    "serve-favicon": "^2.5.0"
48  },
49  "devDependencies": {
50    "@babel/cli": "^7.11.6",
51    "@babel/core": "^7.11.6",
52    "@babel/node": "^7.10.5",
53    "@babel/plugin-transform-runtime": "^7.11.5",
54    "@babel/preset-env": "^7.11.5",
55    "@babel/preset-react": "^7.10.4",
56    "@types/compression": "^1.7.0",
57    "@types/dotenv": "^8.2.0",
58    "@types/express": "^4.17.7",
59    "@types/express-handlebars": "^3.1.0",
60    "@types/jest": "^26.0.4",
61    "@types/jsonwebtoken": "^8.5.0",
62    "@types/lodash": "^4.14.161",
63    "@types/morgan": "^1.9.1",
64    "@types/multer": "^1.4.4",
65    "@types/node": "^14.0.22",
66    "@types/react": "^16.9.43",
67    "@types/react-dom": "^16.9.8",
68    "@types/react-redux": "^7.1.9",
69    "@types/react-router-bootstrap": "^0.24.5",
70    "@types/react-router-dom": "^5.1.5",
```

```

70     "@types/redux": "^3.6.0",
71     "@types/sequelize": "^4.28.9",
72     "@types/serve-favicon": "^2.5.0",
73     "babel-loader": "^8.1.0",
74     "css-loader": "^3.6.0",
75     "eslint": "^7.9.0",
76     "eslint-plugin-jest": "^24.0.1",
77     "eslint-plugin-react": "^7.20.6",
78     "husky": "^4.2.5",
79     "jest": "^26.1.0",
80     "jest-fetch-mock": "^3.0.3",
81     "lorem-ipsu": "^2.0.3",
82     "nodemon": "^2.0.4",
83     "sass": "^1.26.10",
84     "sass-loader": "^9.0.2",
85     "style-loader": "^1.2.1",
86     "stylelint": "^13.6.1",
87     "stylelint-config-sass-guidelines": "^7.0.0",
88     "webpack": "^4.43.0",
89     "webpack-cli": "^3.3.12"
90   },
91   "husky": {
92     "hooks": {
93       "pre-commit": "yarn es-lint && yarn style-lint && yarn test",
94       "pre-push": "./venv/bin/pytest tests"
95     }
96   }
97 }

```

---

## db-course-project-app/.babelrc

```

1 {
2   "presets": ["@babel/preset-env", "@babel/preset-react"],
3   "plugins": [
4     "@babel/plugin-transform-runtime"
5   ]
6 }

```

---

## db-course-project-app/jest.config.js

```

1 module.exports = {
2   setupFiles: ["./src/setupTests.js"],
3   testEnvironment: 'node',
4   testRegex: '(/src/tests|/src/client/tests/).*\\. (test|spec)?\\. (js|jsx)$',
5   moduleFileExtensions: ['js', 'jsx', 'json', 'node']
6 };

```

---

## db-course-project-app/requirements.txt

```

1 attrs==19.3.0
2 iniconfig==1.0.1
3 more-itertools==8.4.0
4 packaging==20.4
5 pluggy==0.13.1
6 py==1.9.0
7 pyparsing==2.4.7
8 pytest==6.0.1
9 selenium==3.141.0
10 six==1.15.0
11 toml==0.10.1
12 urllib3==1.25.10

```

---

## db-course-project-app/README.md

```

1 # db-course-project-app
2 :book: Web-application for course project by Database

```

---

```
3
4 Link to description about project:
  ↳ [db-course-project-report](https://github.com/swimmwatch/db-course-project-report)
```

---

## db-course-project-app/util/nodemon.json

```
1 {
2   "watch": ["src"],
3   "ext": "js",
4   "ignore": ["src/public"],
5   "exec": "babel-node -r dotenv/config src/index.js"
6 }
```

---

## db-course-project-app/tests/test\_smoke.py

```
1 def test_add():
2     assert 2 + 2 == 4
```

---

## db-course-project-app/src/.stylelintrc

```
1 {
2   "extends": "stylelint-config-sass-guidelines"
3 }
```

---

## db-course-project-app/src/index.js

```
1 import * as path from "path";
2 import express from "express";
3 import exphbs from "express-handlebars";
4 import compression from "compression";
5 import morgan from "morgan";
6 import serveFavicon from "serve-favicon";
7 import { Sequelize } from "sequelize";
8
9 import config from "./config";
10
11 import mainRouter from "./routes/main";
12 import authRouter from "./routes/auth";
13 import testEditorRouter from "./routes/testEditor";
14 import profileModify from "./routes/profileModify";
15 import errorHandler from "./middlewares/errorHandler";
16
17 import * as models from "./models";
18
19 const app = express();
20
21 // set static path
22 app.use("/static", express.static("src/public"));
23
24 // set template engine
25 app.set('views', path.join(process.cwd(), 'src', 'templates'));
26 app.engine('handlebars', exphbs());
27 app.set('view engine', 'handlebars');
28
29 // set response compression
30 app.use(compression());
31 // set logger
32 app.use(morgan("common"));
33 // serve json requests
34 app.use(express.json());
35 // serve form requests
36 app.use(express.urlencoded({ extended: true }));
37 // serve favicon
38 app.use(serveFavicon(path.join(process.cwd(), 'src', 'public', 'favicon.ico')))
39
40 const PORT = process.env.PORT || 3000;
41
```



```

42 const sequelize = new Sequelize(process.env.DATABASE_URL, config.db.options);
43
44 // app.get("/admin", (req, res) => {
45 //     res.render("admin");
46 // });
47
48 app.use("/api", authRouter);
49 app.use("/api/profile", profileModify);
50 app.use("/api/test", testEditorRouter);
51
52 app.use("*", mainRouter);
53
54 app.use(errorHandler);
55
56 app.listen(PORT, async () => {
57     try {
58         await sequelize.authenticate();
59         await sequelize.sync({ force: true });
60
61         for (let modelName in models) {
62             await models[modelName].sync();
63         }
64
65         console.log('Connection has been established successfully.');


---



```

## db-course-project-app/src/webpack.config.js

---

```

1 const path = require('path');
2
3 module.exports = {
4     mode: 'development',
5     entry: {
6         main: './src/client/main.jsx',
7         admin: './src/client/admin.jsx'
8     },
9     devtool: 'source-map',
10    module: {
11        rules: [
12            {
13                test: /\.jsx?$/,
14                use: 'babel-loader',
15                exclude: /node_modules/,
16            },
17            {
18                test: /\.s[ac]ss$/i,
19                use: [
20                    // Creates `style` nodes from JS strings
21                    'style-loader',
22                    // Translates CSS into CommonJS
23                    'css-loader',
24                    // Compiles Sass to CSS
25                    'sass-loader',
26                ],
27            },
28        ],
29    },
30    resolve: {
31        extensions: [ '.jsx', '.js' ],
32    },
33    output: {
34        filename: '[name].bundle.js',
35        path: path.resolve(__dirname, 'public'),
36    },
37 };

```

---

## db-course-project-app/src/.eslintrc.js

---

```
1 module.exports = {
2   "env": {
3     "browser": true,
4     "es2020": true,
5     "node": true,
6     "jest": true
7   },
8   "extends": [
9     "eslint:recommended",
10    "plugin:react/recommended",
11    "plugin:jest/recommended"
12  ],
13  "parserOptions": {
14    "ecmaFeatures": {
15      "jsx": true
16    },
17    "ecmaVersion": 11,
18    "sourceType": "module"
19  },
20  "plugins": [
21    "react",
22    "jest"
23  ],
24  "rules": {
25  }
26 };
```

---

#### db-course-project-app/src/client/main.jsx

---

```
1 import * as React from "react";
2 import * as ReactDOM from "react-dom";
3 import { Provider } from "react-redux";
4 import { BrowserRouter } from "react-router-dom";
5 import { toggleLoader } from "../helpers/loader";
6
7 import App from "../apps/main/App";
8 import { store, initAuthStore } from "../store";
9
10 import "bootstrap/scss/bootstrap.scss";
11
12 initAuthStore(store).then(() => {
13   toggleLoader();
14
15   ReactDOM.render(
16     <Provider store={store}>
17       <BrowserRouter>
18         <App />
19       </BrowserRouter>
20     </Provider>,
21     document.getElementById("root")
22   );
23 });
```

---

#### db-course-project-app/src/client/admin.jsx

---

```
1 import * as React from "react";
2 import * as ReactDOM from "react-dom";
3 import { BrowserRouter } from "react-router-dom";
4
5 import "bootstrap/scss/bootstrap.scss";
6
7 import App from "../apps/admin/App";
8
9 ReactDOM.render(
10   <BrowserRouter>
11     <App />
12   </BrowserRouter>,
13   document.getElementById("root")
14 );
```

---

db-course-project-app/src/client/tests/smoke.test.js

```
1 describe("Sum", () => {
2   it("addition", () => {
3     expect(6).toBe(6);
4   });
5 });
```

db-course-project-app/src/client/apps/main/App.jsx

```
1 import * as React from 'react';
2 import { Switch, Route } from 'react-router-dom';
3 import PrivateRoute from "../../hoc/PrivateRoute";
4 import NotIsLoggedInRoute from "../../hoc/NotIsLoggedInRoute";
5
6 import Container from "react-bootstrap/Container";
7
8 import Login from "./pages/Login";
9 import Home from "./pages/Home"
10 import SignUp from "./pages/SignUp";
11 import TestEditor from "./pages/TestEditor";
12
13 import Footer from "./components/Footer";
14 import Header from "./components/Header";
15 import Profile from "./pages/Profile";
16 import HttpErrorInfo from "./components/HttpErrorInfo";
17 import {NOT_FOUND} from "http-status-codes";
18
19 class App extends React.Component {
20   render() {
21     return (
22       <div>
23         <Header />
24
25         <Switch>
26           <Route exact path="/" component={Home}/>
27           <NotIsLoggedInRoute path="/login" component={Login}/>
28           <NotIsLoggedInRoute path="/signup" component={SignUp}/>
29           <PrivateRoute path="/profile" component={Profile}/>
30           <PrivateRoute exact path="/test/edit" component={TestEditor} />
31           <Route component={() => <HttpErrorInfo status={NOT_FOUND} />} />
32         </Switch>
33
34         <Container className="p-3">
35           <Footer/>
36         </Container>
37       </div>
38     );
39   }
40 }
41
42 export default App;
```

db-course-project-app/src/client/apps/main/pages/ProfileSettings/style.scss

```
1
```

db-course-project-app/src/client/apps/main/pages/ProfileSettings/ProfileSettings.js

```
1 import * as React from "react";
2 import Container from "react-bootstrap/Container";
3 import Row from "react-bootstrap/Row";
4 import Col from "react-bootstrap/Col";
5 import ErrorFormAlert from "../../components/ErrorFormAlert";
6 import UpdatePasswordForm from "../../components/UpdatePasswordForm";
7 import DeleteProfileForm from "../../components/DeleteProfileForm";
8
9 import "../style.scss";
```

```

10
11 class ProfileSettings extends React.Component {
12   constructor(props) {
13     super(props);
14
15     this.state = {
16       listErrors: []
17     }
18
19     this.updateListErrors = this.updateListErrors.bind(this);
20     this.hideErrorAlert = this.hideErrorAlert.bind(this);
21   }
22
23   updateListErrors(errors) {
24     this.setState({ listErrors: errors })
25   }
26
27   hideErrorAlert() {
28     this.setState({ listErrors: [] });
29   }
30
31   render() {
32     const { listErrors } = this.state;
33
34     return (
35       <Container className="p-3">
36         <Row>
37           <ErrorFormAlert listErrors={listErrors}
38                         show={listErrors.length !== 0}
39                         onHide={this.hideErrorAlert} />
40         </Row>
41         <Row>
42           <Col lg={6}>
43             <UpdatePasswordForm onSubmitError={this.updateListErrors} />
44           </Col>
45         </Row>
46         <Row>
47           <Col lg={6}>
48             <DeleteProfileForm />
49           </Col>
50         </Row>
51       </Container>
52     );
53   }
54 }
55
56 export default ProfileSettings;

```

---

db-course-project-app/src/client/apps/main/pages/ProfileSettings/index.js

---

```

1 import ProfileSettings from "../ProfileSettings";
2
3 export default ProfileSettings;

```

---

db-course-project-app/src/client/apps/main/pages/SignUp/style.scss

---

```

1 .main-signup-form {
2   &__title {
3     text-align: center;
4   }
5
6   &__label {
7     font-weight: bold;
8   }
9 }

```

---

db-course-project-app/src/client/apps/main/pages/SignUp/index.js

---

```

1 import SignUp from "../SignUp";

```

---

```
2
3 export default SignUp;
```

---

db-course-project-app/src/client/apps/main/pages/SignUp/SignUp.jsx

---

```
1 import * as React from "react";
2 import ReactRouterPropTypes from "react-router-prop-types";
3 import { withRouter } from "react-router-dom";
4
5 import Form from "react-bootstrap/Form";
6 import Button from "react-bootstrap/Button";
7 import Container from "react-bootstrap/Container";
8 import Col from "react-bootstrap/Col";
9 import authService from "../../../../../services/auth";
10
11 import ErrorFormAlert from "../../components/ErrorFormAlert";
12
13 import userConstraints from "../../../../../models/User/constraints";
14
15 import "./style.scss";
16
17 const {
18   MIN_PASSWORD_LENGTH,
19   MAX_PASSWORD_LENGTH,
20   MIN_LOGIN_LENGTH,
21   MAX_LOGIN_LENGTH,
22   MAX_EMAIL_LENGTH
23 } = userConstraints;
24
25 class SignUp extends React.Component {
26   constructor(props) {
27     super(props);
28
29     this.state = {
30       email: '',
31       login: '',
32       password: '',
33       repeatPassword: '',
34
35       listErrors: [],
36
37       isLoading: false
38     };
39
40     this.handleInputChange = this.handleInputChange.bind(this);
41     this.handleFormSubmit = this.handleFormSubmit.bind(this);
42     this.hideErrorAlert = this.hideErrorAlert.bind(this);
43     this.toggleLoadingState = this.toggleLoadingState.bind(this);
44   }
45
46   handleInputChange(event) {
47     const { name, value } = event.target;
48
49     this.setState({ [name]: value });
50   }
51
52   toggleLoadingState() {
53     this.setState(prev => {
54       return {
55         isLoading: !prev.isLoading
56       }
57     });
58   }
59
60   _generateFormData() {
61     const formData = new FormData();
62
63     formData.append('login', this.state.login);
64     formData.append('password', this.state.password);
65     formData.append('email', this.state.email);
66     formData.append('repeatPassword', this.state.repeatPassword);
67
68     return formData;
69   }
```

```

70
71   async handleFormSubmit(event) {
72     event.preventDefault();
73
74     const { history } = this.props;
75     const formData = this._generateFormData();
76
77     this.toggleLoadingState();
78
79     try {
80       await authService.signUp(formData);
81
82       history.push('/login');
83     } catch ({ errors }) {
84       this.setState({ listErrors: errors });
85     }
86
87     this.toggleLoadingState();
88   }
89
90   hideErrorAlert() {
91     this.setState({ listErrors: [] });
92   }
93
94   render() {
95     const { listErrors, isLoading } = this.state;
96
97     return (
98       <Container className="p-3">
99         <Col lg={{ offset: 3, span: 6 }}>
100           <h2 className="main-signup-form__title">Sign Up form</h2>
101           <Form>
102             <ErrorFormAlert listErrors={listErrors}
103                           show={listErrors.length !== 0}
104                           onHide={this.hideErrorAlert} />
105             <Form.Group controlId="main-signup-form__email">
106               <Form.Control type="email"
107                             placeholder="Enter email"
108                             name="email"
109                             maxLength={MAX_EMAIL_LENGTH}
110                             required
111                             onChange={this.handleInputChange} />
112             </Form.Group>
113             <Form.Group controlId="main-signup-form__login">
114               <Form.Control type="text"
115                             placeholder="Enter login"
116                             name="login"
117                             minLength={MIN_LOGIN_LENGTH}
118                             maxLength={MAX_LOGIN_LENGTH}
119                             required
120                             onChange={this.handleInputChange} />
121             </Form.Group>
122             <Form.Group controlId="main-signup-form__password">
123               <Form.Control type="password"
124                             placeholder="Enter password"
125                             name="password"
126                             minLength={MIN_PASSWORD_LENGTH}
127                             maxLength={MAX_PASSWORD_LENGTH}
128                             required
129                             onChange={this.handleInputChange} />
130             </Form.Group>
131             <Form.Group controlId="main-signup-form__repeat-password">
132               <Form.Control type="password"
133                             placeholder="Enter password"
134                             name="repeatPassword"
135                             minLength={MIN_PASSWORD_LENGTH}
136                             maxLength={MAX_PASSWORD_LENGTH}
137                             required
138                             onChange={this.handleInputChange} />
139             </Form.Group>
140             <Button variant="primary"
141                     type="submit"
142                     block
143                     disabled={isLoading}
144                     onClick={this.handleFormSubmit}>
145               { isLoading ? 'Loading...' : 'Submit' }

```

```

146             </Button>
147         </Form>
148     </Col>
149 </Container>
150 );
151 }
152 }
153
154 SignUp.propTypes = {
155     history: ReactRouterPropTypes.history
156 };
157
158 export default withRouter(SignUp);

```

---

## db-course-project-app/src/client/apps/main/pages/Profile/style.scss

---

```

1 .profile {
2     &__username {
3         font-weight: bold;
4         text-align: center;
5     }
6
7     &__avatar {
8         border: 6px solid #f0f8ff;
9         border-radius: 100%;
10        display: block;
11        margin: 0 auto;
12        width: 70%;
13    }
14 }

```

---

## db-course-project-app/src/client/apps/main/pages/Profile/Profile.jsx

---

```

1 import * as React from "react";
2 import PropTypes from "prop-types";
3 import { connect } from "react-redux";
4 import {NOT_FOUND} from "http-status-codes";
5
6 import { Switch, Route } from 'react-router-dom';
7 import { LinkContainer } from "react-router-bootstrap";
8 import Container from "react-bootstrap/Container";
9 import Col from "react-bootstrap/Col";
10 import Row from "react-bootstrap/Row";
11 import Nav from "react-bootstrap/Nav";
12 import ProfileSettings from "../ProfileSettings";
13 import ProfileAttempts from "../ProfileAttempts";
14 import HttpErrorInfo from "../../components/HttpErrorInfo";
15 import ProfileTests from "../ProfileTests";
16
17 import "../style.scss";
18
19 const Profile = ({ user }) => {
20     return (
21         <Container className="p-3">
22             <Row>
23                 <Col lg={3}>
24                     
26                     <p className="profile__username">{ user.login }</p>
27                 </Col>
28                 <Col lg={9}>
29                     <Nav variant="tabs" defaultActiveKey="tests">
30                         <Nav.Item>
31                             <LinkContainer to="/profile/tests">
32                                 <Nav.Link eventKey="tests">My tests</Nav.Link>
33                             </LinkContainer>
34                         </Nav.Item>
35                         <Nav.Item>
36                             <LinkContainer to="/profile/attempts">
37                                 <Nav.Link eventKey="attempts">My attempts</Nav.Link>
38                             </LinkContainer>
39                         </Nav.Item>

```

```

40         <Nav.Item>
41             <LinkContainer to="/profile/settings">
42                 <Nav.Link eventKey="settings">Settings</Nav.Link>
43             </LinkContainer>
44         </Nav.Item>
45     </Nav>
46
47     <Switch>
48         <Route exact path="/profile" render={() => <ProfileTests />} />
49         <Route path="/profile/tests" render={() => <ProfileTests />} />
50         <Route path="/profile/attempts" component={ProfileAttempts} />
51         <Route path="/profile/settings" component={ProfileSettings} />
52         <Route component={() => <HttpErrorInfo status={NOT_FOUND} />} />
53     </Switch>
54 </Col>
55 </Row>
56 </Container>
57 );
58 }
59
60 Profile.propTypes = {
61     user: PropTypes.shape({
62         login: PropTypes.string
63     })
64 };
65
66 function mapStateToProps(state) {
67     const { user } = state.auth;
68
69     return { user };
70 }
71
72 const connectedProfile = connect(mapStateToProps)(Profile);
73
74 export { connectedProfile as Profile };

```

---

db-course-project-app/src/client/apps/main/pages/Profile/index.js

---

```

1 import { Profile } from "../Profile";
2
3 export default Profile;

```

---

db-course-project-app/src/client/apps/main/pages/ProfileAttempts/style.scss

---

```

1

```

---

db-course-project-app/src/client/apps/main/pages/ProfileAttempts/ProfileAttempts.js

---

```

1 import * as React from "react";
2
3 import "../style.scss";
4
5 const ProfileAttempts = () => {
6     return (
7         <h2>Attempts</h2>
8     );
9 };
10
11 export default ProfileAttempts;

```

---

db-course-project-app/src/client/apps/main/pages/ProfileAttempts/index.js

---

```

1 import ProfileAttempts from "../ProfileAttempts";
2
3 export default ProfileAttempts;

```

---



db-course-project-app/src/client/apps/main/pages/Home/style.scss

1

db-course-project-app/src/client/apps/main/pages/Home/index.js

```
1 import Home from "../Home";
2
3 export default Home;
```

db-course-project-app/src/client/apps/main/pages/Home/Home.jsx

```
1 import * as React from "react";
2
3 import Jumbotron from "react-bootstrap/Jumbotron";
4 import Container from "react-bootstrap/Container";
5 import Row from "react-bootstrap/Row";
6 import Col from "react-bootstrap/Col";
7
8 import "../style.scss";
9
10 const Home = () => {
11   return (
12     <Container className="p-3">
13       <Jumbotron>
14         <h1>Hello!</h1>
15         <p>
16           This is a simple hero unit, a simple jumbotron-style component for calling
17           extra attention to featured content or information.
18         </p>
19       </Jumbotron>
20       <Row>
21         <Col lg={4}>
22           <h2>Create</h2>
23           <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ab blanditiis delectus
24             ↳ dolorem eius expedita nemo neque pariatur perferendis quasi! Accusantium ad atque
25             ↳ corporis deleniti eius, eligendi qui rem sunt vitae!</p>
26         </Col>
27         <Col lg={4}>
28           <h2>Share</h2>
29           <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ab blanditiis delectus
30             ↳ dolorem eius expedita nemo neque pariatur perferendis quasi! Accusantium ad atque
31             ↳ corporis deleniti eius, eligendi qui rem sunt vitae!</p>
32         </Col>
33       </Row>
34     </Container>
35   );
36 }
37
38 export default Home;
```

db-course-project-app/src/client/apps/main/pages/Login/style.scss

```
1 .main-login-form {
2   &__forgot-password {
3     display: block;
4     margin: 10px 0;
5     text-align: center;
6   }
7
8   &__title {
9     text-align: center;
```

```

10   }
11
12   &__label {
13     font-weight: bold;
14   }
15 }

```

---

## db-course-project-app/src/client/apps/main/pages/Login/Login.jsx

---

```

1  import * as React from "react";
2  import ReactRouterPropTypes from "react-router-prop-types";
3  import PropTypes from "prop-types";
4  import { connect } from "react-redux";
5
6  import Form from "react-bootstrap/Form";
7  import Button from "react-bootstrap/Button";
8  import Container from "react-bootstrap/Container";
9  import Col from "react-bootstrap/Col";
10 import LinkContainer from "react-router-bootstrap/lib/LinkContainer";
11 import ErrorFormAlert from "../../components/ErrorFormAlert";
12
13 import authService from "../../../../../services/auth";
14 import * as authActions from "../../../../../actions/auth";
15 import userConstraints from "../../../../../models/User/constraints";
16
17 import "./style.scss";
18
19 const {
20   MIN_PASSWORD_LENGTH,
21   MAX_PASSWORD_LENGTH,
22   MIN_LOGIN_LENGTH,
23   MAX_LOGIN_LENGTH
24 } = userConstraints;
25
26 class Login extends React.Component {
27   constructor(props) {
28     super(props);
29
30     this.state = {
31       login: '',
32       password: '',
33       readMeChecked: false,
34
35       listErrors: [],
36
37       isLoading: false
38     }
39
40     this.handleChange = this.handleChange.bind(this);
41     this.handleCheckboxChange = this.handleCheckboxChange.bind(this);
42     this.handleSubmit = this.handleSubmit.bind(this);
43     this.hideErrorAlert = this.hideErrorAlert.bind(this);
44   }
45
46   handleChange(event) {
47     const { name, value } = event.target;
48
49     this.setState({ [name]: value });
50   }
51
52   handleCheckboxChange(event) {
53     const { name, checked } = event.target;
54
55     this.setState({ [name]: checked });
56   }
57
58   _generateFormData() {
59     const formData = new FormData();
60
61     formData.append('login', this.state.login);
62     formData.append('password', this.state.password);
63
64     return formData;
65   }

```

```

66
67   async handleFormSubmit(event) {
68     event.preventDefault();
69
70     const { history, dispatch } = this.props;
71     const formData = this._generateFormData();
72
73     this.toggleLoadingState();
74
75     try {
76       const { token, user } = await authService.signIn(formData);
77
78       localStorage.setItem('TOKEN', token);
79
80       dispatch(authActions.success(user));
81
82       history.push('/');
83     } catch ({ errors }) {
84       this.setState({ listErrors: errors });
85     }
86
87     this.toggleLoadingState();
88   }
89
90   hideErrorAlert() {
91     this.setState({
92       listErrors: []
93     });
94   }
95
96   toggleLoadingState() {
97     this.setState(prev => {
98       return {
99         isLoading: !prev.isLoading
100       }
101     });
102   }
103
104   render() {
105     const { listErrors, isLoading } = this.state;
106
107     return (
108       <Container className="p-3">
109         <Col lg={{offset: 3, span: 6}}>
110           <h2 className="main-login-form__title">Login form</h2>
111           <Form>
112             <ErrorFormAlert listErrors={listErrors}
113               show={listErrors.length !== 0}
114               onHide={this.hideErrorAlert} />
115             <Form.Group controlId="main-login-form__login">
116               <Form.Control type="text"
117                 placeholder="Enter login"
118                 name="login"
119                 minLength={MIN_LOGIN_LENGTH}
120                 maxLength={MAX_LOGIN_LENGTH}
121                 required
122                 onChange={this.handleInputChange} />
123             </Form.Group>
124             <Form.Group controlId="main-login-form__password">
125               <Form.Control type="password"
126                 placeholder="Enter password"
127                 name="password"
128                 minLength={MIN_PASSWORD_LENGTH}
129                 maxLength={MAX_PASSWORD_LENGTH}
130                 required
131                 onChange={this.handleInputChange} />
132             </Form.Group>
133             <Form.Group controlId="main-login-form__checkbox">
134               <Form.Check type="checkbox"
135                 label="Remember me"
136                 name="readMeChecked"
137                 onChange={this.handleCheckboxChange} />
138             </Form.Group>
139             <Button variant="primary"
140               type="submit"
141               block

```

```

142             disabled={isLoading}
143             onClick={this.handleFormSubmit}>
144             { isLoading ? 'Loading...' : 'Submit' }
145         </Button>
146         <LinkContainer to="#">
147             <a className="main-login-form__forgot-password">Forgot password?</a>
148         </LinkContainer>
149     </Form>
150 </Col>
151 </Container>
152 );
153 }
154 }
155
156 Login.propTypes = {
157     location: ReactRouterPropTypes.location,
158     history: ReactRouterPropTypes.history,
159     dispatch: PropTypes.func
160 };
161
162 const connectedLogin = connect()(Login);
163
164 export { connectedLogin as Login };

```

---

db-course-project-app/src/client/apps/main/pages/Login/index.js

---

```

1 import { Login } from "../Login";
2
3 export default Login;

```

---

db-course-project-app/src/client/apps/main/components/Header/style.scss

---

```

1 // .header {
2 //   @_user-dropdown-menu {
3 //     float: right;
4 //   }
5 // }

```

---

db-course-project-app/src/client/apps/main/components/Header/Header.jsx

---

```

1 import * as React from "react";
2 import PropTypes from "prop-types";
3 import { withRouter } from "react-router-dom";
4 import ReactRouterPropTypes from "react-router-prop-types";
5 import { connect } from "react-redux";
6 import * as authActions from "../../actions/auth";
7
8 import Navbar from "react-bootstrap/Navbar";
9 import Nav from "react-bootstrap/Nav";
10 import NavDropdown from "react-bootstrap/NavDropdown";
11 import Modal from "react-bootstrap/Modal";
12 import Button from "react-bootstrap/Button";
13
14 import { LinkContainer } from "react-router-bootstrap";
15
16 import "../style.scss";
17
18 const Header = ({ isLoggedIn, user, dispatch, history }) => {
19     const [modalShow, setModalShow] = React.useState(false);
20
21     const showModal = () => setModalShow(true);
22
23     const hideModal = () => setModalShow(false);
24
25     const onLogout = () => {
26         dispatch(authActions.logout());
27
28         setModalShow(false);
29     }

```

```

30     history.push("/");
31 };
32
33 return (
34     <Navbar bg="dark" variant="dark">
35         <LinkContainer to="/">
36             <Navbar.Brand>PassQuiz</Navbar.Brand>
37         </LinkContainer>
38         <Nav className="mr-auto">
39             { !isLoggedIn ? (
40                 <>
41                     <LinkContainer to="/signup">
42                         <Nav.Link>Sign Up</Nav.Link>
43                     </LinkContainer>
44                     <LinkContainer to="/login">
45                         <Nav.Link>Login</Nav.Link>
46                     </LinkContainer>
47                 </>
48             ) : (
49                 <NavDropdown title={ user.login } id="user-nav-dropdown">
50                     <LinkContainer to="/profile">
51                         <NavDropdown.Item>My profile</NavDropdown.Item>
52                     </LinkContainer>
53                     <NavDropdown.Divider />
54                     <LinkContainer to="#">
55                         <NavDropdown.Item onClick={showModal}>Logout</NavDropdown.Item>
56                     </LinkContainer>
57                 </NavDropdown>
58             ) }
59         </Nav>
60         {
61             isLoggedIn && (
62                 <Nav>
63                     <LinkContainer to="/test/edit">
64                         <Button type="primary">Create test</Button>
65                     </LinkContainer>
66                 </Nav>
67             )
68         }
69
70         <Modal
71             size="md"
72             aria-labelledby="contained-modal-title-vcenter"
73             centered
74             onHide={hideModal}
75             show={modalShow}
76         >
77             <Modal.Header closeButton>
78                 <Modal.Title id="contained-modal-title-vcenter">Log out</Modal.Title>
79             </Modal.Header>
80             <Modal.Body>
81                 <p>Are you sure you want to log-off?</p>
82             </Modal.Body>
83             <Modal.Footer>
84                 <Button onClick={onLogOut}>Yes</Button>
85             </Modal.Footer>
86         </Modal>
87     </Navbar>
88 );
89 }
90
91 Header.propTypes = {
92     history: ReactRouterPropTypes.history,
93     dispatch: PropTypes.func,
94     isLoggedIn: PropTypes.bool,
95     user: PropTypes.shape({
96         login: PropTypes.string
97     })
98 };
99
100 function mapStateToProps(state) {
101     const { isLoggedIn, user } = state.auth;
102
103     return { isLoggedIn, user };
104 }
105

```

```

106 const headerWithRouter = withRouter(Header);
107 const connectedHeaderWithRouter = connect(mapStateToProps)(headerWithRouter);
108
109 export { connectedHeaderWithRouter as Header };

```

---

db-course-project-app/src/client/apps/main/components/Header/index.js

---

```

1 import { Header } from "../Header";
2
3 export default Header;

```

---

db-course-project-app/src/client/apps/main/components/TestCard/style.scss

---

```

1 .test-card {
2   margin: 0 0 15px;
3   width: 100%;
4
5   &__title {
6     font-size: 1.1rem;
7   }
8
9   &__label-info {
10    font-weight: bold;
11    padding-right: 5px;
12  }
13
14  &__control {
15    display: flex;
16    justify-content: space-between;
17  }
18 }

```

---

db-course-project-app/src/client/apps/main/components/TestCard/index.js

---

```

1 import TestCard from "../TestCard";
2
3 export default TestCard;

```

---

db-course-project-app/src/client/apps/main/components/TestCard/TestCard.jsx

---

```

1 import * as React from "react";
2 import PropTypes from "prop-types";
3 import { useHistory } from "react-router-dom";
4 import Card from "react-bootstrap/Card";
5 import Button from "react-bootstrap/Button";
6 import Dropdown from "react-bootstrap/Dropdown";
7 import TagList from "../TagList";
8
9 import "../style.scss";
10
11 const TestCard = ({ title, description, author, tags, onDeleteTestCard, testId }) => {
12   const history = useHistory();
13
14   return (
15     <Card className="test-card">
16       <Card.Body>
17         <Card.Title className="test-card__title">{title}</Card.Title>
18
19         <div className="test-card__author">
20           <span className="test-card__label-info">Author:</span> {author}
21         </div>
22
23         <div className="test-card__tags">
24           <span className="test-card__label-info">Tags:</span>
25           {
26             tags.length ? (
27               <TagList tags={tags} canDelete={false} />

```

```

28         ) : (
29             "None"
30         )
31     }
32 </div>
33
34 <Card.Text className="test-card__description">{description}</Card.Text>
35
36 <div className="test-card__control">
37     <Button className="test-card__pass-btn"
38         variant="primary">Pass test</Button>
39
40     <Dropdown className="test-card__dropdown-menu">
41         <Dropdown.Toggle variant="primary">Menu</Dropdown.Toggle>
42         <Dropdown.Menu>
43             <Dropdown.Item as="button" onClick={() => {
44                 history.push(`/test/edit?id=${testId}`);
45             }}>Edit</Dropdown.Item>
46             <Dropdown.Item as="button"
47                 onClick={() => {
48                     onDeleteTestCard(testId);
49                 }}>
50                 Delete
51             </Dropdown.Item>
52             <Dropdown.Item as="button">Share</Dropdown.Item>
53         </Dropdown.Menu>
54     </Dropdown>
55 </div>
56 </Card.Body>
57 </Card>
58 );
59 };
60
61 TestCard.propTypes = {
62     title: PropTypes.string.isRequired,
63     description: PropTypes.string.isRequired,
64     author: PropTypes.string.isRequired,
65     tags: PropTypes.arrayOf(PropTypes.string).isRequired,
66     testId: PropTypes.number.isRequired,
67     onDeleteTestCard: PropTypes.func
68 };
69
70 export default TestCard;

```

---

db-course-project-app/src/client/apps/main/components/ListTestCards/index.js

---

```

1 import ListTestCards from "../ListTestCards";
2
3 export default ListTestCards;

```

---

db-course-project-app/src/client/apps/main/components/ListTestCards/ListTestCards.js

---

```

1 import * as React from "react";
2 import PropTypes from "prop-types";
3
4 import Row from "react-bootstrap/Row";
5 import TestCard from "../TestCard";
6
7 import "./style.scss";
8
9 const ListTestCards = ({ tests, onDeleteTestCard }) => {
10     return (
11         <Row>
12             {
13                 tests.map(test => {
14                     const {
15                         title,
16                         description,
17                         author,
18                         tags,
19                         testId
20                     } = test;

```

```

21
22         return (
23             <TestCard title={title}
24                 description={description}
25                 author={author}
26                 tags={tags}
27                 key={testId}
28                 testId={testId}
29                 onDeleteTestCard={onDeleteTestCard} />
30         );
31     })
32 }
33 </Row>
34 );
35 };
36
37 ListTestCards.propTypes = {
38     tests: PropTypes.array,
39     onDeleteTestCard: PropTypes.func
40 };
41
42 export default ListTestCards;

```

---

db-course-project-app/src/client/apps/main/components/Footer/style.scss

---

```

1 .footer {
2     &__copyright {
3         font-weight: bold;
4     }
5 }

```

---

db-course-project-app/src/client/apps/main/components/Footer/Footer.jsx

---

```

1 import * as React from "react";
2
3 import "./style.scss"
4
5 const Footer = () => {
6     const currDate = new Date();
7
8     return (
9         <footer className="footer">
10             <hr/>
11             <p className="footer__copyright">&#169; Copyright {currDate.getFullYear()}</p>
12         </footer>
13     );
14 };
15
16 export default Footer;

```

---

db-course-project-app/src/client/apps/main/components/Footer/index.js

---

```

1 import Footer from './Footer';
2
3 export default Footer;

```

---

db-course-project-app/src/client/apps/admin/Login.jsx

---

```

1 import * as React from "react";
2
3 import Form from "react-bootstrap/Form";
4 import Button from "react-bootstrap/Button";
5
6 const Login = () => {
7     return (
8         <Form>
9             <Form.Group controlId="formBasicEmail">

```



```

10         <Form.Label>Email address</Form.Label>
11         <Form.Control type="email" placeholder="Enter email" />
12     </Form.Group>
13     <Form.Group controlId="formBasicPassword">
14         <Form.Label>Password</Form.Label>
15         <Form.Control type="password" placeholder="Password" />
16     </Form.Group>
17     <Form.Group controlId="formBasicCheckbox">
18         <Form.Check type="checkbox" label="Check me out" />
19     </Form.Group>
20     <Button variant="primary" type="submit">Submit</Button>
21 </Form>
22     );
23 };
24
25 export default Login;

```

---

## db-course-project-app/src/client/apps/admin/App.jsx

---

```

1 import * as React from 'react';
2 import { Switch, Route } from 'react-router-dom';
3
4 import Container from 'react-bootstrap/Container';
5
6 import Login from "../Login";
7
8 class App extends React.Component {
9     render() {
10         const App = () => (
11             <div>
12                 <Container className="p-3">
13                     <Switch>
14                         <Route path='/admin/login' component={Login}/>
15                     </Switch>
16                 </Container>
17             </div>
18         );
19
20         return (
21             <Switch>
22                 <App/>
23             </Switch>
24         );
25     }
26 }
27
28 export default App;

```

---

## db-course-project-app/src/routes/main.js

---

```

1 import { Router } from "express";
2
3 const router = new Router();
4
5 router.get("/*", (req, res) => {
6     res.render("index");
7 });
8
9 export default router;

```

---

## db-course-project-app/src/routes/auth.js

---

```

1 import multer from "multer";
2 import { Router } from "express";
3 import checkToken from "../middlewares/checkToken";
4 import * as authController from "../controllers/auth";
5
6 const upload = multer();
7

```

```

8 const router = new Router();
9
10 router.post('/signup', upload.none(), authController.signUp);
11 router.post('/signin', upload.none(), authController.signIn);
12 router.post('/init', checkToken, authController.initAuth);
13
14 export default router;

```

---

## db-course-project-app/src/tests/smoke.test.js

---

```

1 describe("Sum", () => {
2   it("addiction", () => {
3     expect(6).toBe(6);
4   });
5 });

```

---

## db-course-project-app/src/controllers/auth.js

---

```

1 import {
2   INTERNAL_SERVER_ERROR,
3   BAD_REQUEST,
4   OK
5 } from "http-status-codes";
6 import * as jwt from "jsonwebtoken";
7 import FormListErrors from "../helpers/FormListErrors";
8 import User from "../models/User";
9
10 export const signUp = async (req, res, next) => {
11   const {
12     login,
13     repeatPassword,
14     password,
15     email,
16   } = req.body;
17   const formListErrors = new FormListErrors();
18
19   if (repeatPassword !== password) {
20     formListErrors.add("passwords doesn't equal.");
21
22     next({
23       status: BAD_REQUEST,
24       errors: formListErrors.data.errors
25     });
26   }
27
28   try {
29     await User.create({ email, login, password });
30   } catch ({ errors }) {
31     formListErrors.addFromModelErrors(errors);
32
33     next({
34       status: BAD_REQUEST,
35       errors: formListErrors.data.errors
36     });
37   }
38
39   res.sendStatus(OK);
40 };
41
42 export const signIn = async (req, res, next) => {
43   const {
44     login,
45     password,
46   } = req.body;
47   const formListErrors = new FormListErrors();
48
49   let user;
50   try {
51     user = await User.findOne({ where: { login } });
52   } catch (error) {
53     formListErrors.addDefault();
54   }

```

```

55     next({
56       status: INTERNAL_SERVER_ERROR,
57       errors: formListErrors.data.errors
58     });
59   }
60
61   if (!user) {
62     formListErrors.add("user with such name not found.");
63
64     next({
65       status: BAD_REQUEST,
66       errors: formListErrors.data.errors
67     });
68   } else {
69     const isRightPassword = await user.comparePasswords(password);
70
71     if (isRightPassword) {
72       const token = jwt.sign({ sub: user.id }, process.env.JWT_SECRET);
73
74       res.json({
75         ...user.initState(),
76         token
77       });
78     } else {
79       formListErrors.add("password is invalid");
80
81       next({
82         status: BAD_REQUEST,
83         errors: formListErrors.data.errors
84       });
85     }
86   }
87 };
88
89 export const initAuth = async (req, res) => {
90   const user = await User.findByPk(req.userId);
91
92   res.json({ ...user.initState() });
93 };

```

---

## db-course-project-app/src/helpers/FormListErrors.js

---

```

1  // TODO: add documentation
2
3  export default class FormListErrors {
4    constructor() {
5      this.data = { errors: [] };
6    }
7
8    addFromModelErrors(errors) {
9      this.data.errors.push(
10        ...errors.map(err => {
11          let { message } = err;
12
13          return { message };
14        })
15      );
16    }
17
18    addDefault() {
19      this.data.errors.push({
20        message: "Oops, something went wrong."
21      });
22    }
23
24    add(message) {
25      this.data.errors.push({ message });
26    }
27
28    isEmpty() {
29      return this.data.errors.length;
30    }
31  }

```

---

## db-course-project-app/src/templates/admin.handlebars

```
1 <main id="root">
2   <!-- React entry point -->
3 </main>
4
5 <script src="/static/admin.bundle.js"></script>
```

## db-course-project-app/src/templates/index.handlebars

```
1 <main id="root">
2   <!-- React entry point -->
3 </main>
4
5 <script src="/static/main.bundle.js"></script>
```

## db-course-project-app/src/templates/layouts/main.handlebars

```
1 <html lang="en">
2 <head>
3   {{> meta }}
4   {{> favicon }}
5   <title>passquiz</title>
6 </head>
7 <body>
8   {{> loader }}
9   {{{ body }}}
10 </body>
11 </html>
```

## db-course-project-app/src/templates/partials/favicon.handlebars

```
1 <link rel="apple-touch-icon" sizes="180x180" href="/static/apple-touch-icon.png">
2 <link rel="icon" type="image/png" sizes="32x32" href="/static/favicon-32x32.png">
3 <link rel="icon" type="image/png" sizes="192x192" href="/static/android-chrome-192x192.png">
4 <link rel="icon" type="image/png" sizes="16x16" href="/static/favicon-16x16.png">
5 <link rel="manifest" href="/static/site.webmanifest">
6 <meta name="apple-mobile-web-app-title" content="lms.labchecker.ru">
7 <meta name="application-name" content="lms.labchecker.ru">
8 <meta name="msapplication-TileColor" content="#00aba9">
9 <meta name="theme-color" content="#ffffff">
```

## db-course-project-app/src/templates/partials/meta.handlebars

```
1 <meta charset="UTF-8">
2 <meta name="viewport"
3   content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0,
4   <=> minimum-scale=1.0">
5 <meta http-equiv="X-UA-Compatible" content="ie=edge">
```

## db-course-project-app/src/middlewares/checkToken.js

```
1 import * as jwt from "jsonwebtoken";
2
3 export default async (req, res, next) => {
4   const token = req.headers["authorization"];
5
6   let tokenObj = null;
7   try {
8     tokenObj = await jwt.verify(token, process.env.JWT_SECRET);
9   } catch (err) {
10     console.error(err);
11   }
12 }
```

```
13     req.userId = tokenObj.sub;
14
15     next();
16 };
```

---

#### db-course-project-app/src/models/UserRole.js

---

```
1 import { Sequelize } from "sequelize";
2
3 import config from "../config";
4
5 import User from "./User";
6 import Role from "./Role";
7
8 const sequelize = new Sequelize(process.env.DATABASE_URL, config.db.options);
9
10 // setup many to many
11 const UserRole = sequelize.define("user_role", {});
12
13 User.belongsToMany(Role, { through: UserRole });
14 Role.belongsToMany(User, { through: UserRole });
15
16 export default UserRole;
```

---

#### db-course-project-app/src/models/Role.js

---

```
1 import { Sequelize } from "sequelize";
2
3 import config from "../config";
4
5 const sequelize = new Sequelize(process.env.DATABASE_URL, config.db.options);
6
7 const Role = sequelize.define("role", {
8   id: {
9     type: Sequelize.INTEGER.UNSIGNED,
10    autoIncrement: true,
11    primaryKey: true
12  },
13   role_name: {
14     type: Sequelize.STRING(64),
15     allowNull: false,
16     unique: true
17   }
18 });
19
20 export default Role;
```

---

#### db-course-project-app/src/models/User/constraints.js

---

```
1 export default {
2   // Password
3   MIN_PASSWORD_LENGTH: 10,
4   MAX_PASSWORD_LENGTH: 128,
5
6   // Login
7   MIN_LOGIN_LENGTH: 6,
8   MAX_LOGIN_LENGTH: 128,
9
10  // Email
11  MAX_EMAIL_LENGTH: 320
12 };
```

---

#### db-course-project-app/src/models/User/User.js

---

```
1 import { Sequelize } from "sequelize";
2 import * as bcrypt from "bcrypt";
3
```

```

4 import config from "../config";
5 import userConstraints from "../constraints";
6 import Test from "../Test";
7
8 const {
9   MIN_PASSWORD_LENGTH,
10  MAX_PASSWORD_LENGTH,
11  MIN_LOGIN_LENGTH,
12  MAX_LOGIN_LENGTH,
13  MAX_EMAIL_LENGTH
14 } = userConstraints;
15
16 const sequelize = new Sequelize(process.env.DATABASE_URL, config.db.options);
17
18 const User = sequelize.define("user", {
19   id: {
20     type: Sequelize.INTEGER,
21     autoIncrement: true,
22     primaryKey: true,
23   },
24   login: {
25     type: Sequelize.STRING(MAX_LOGIN_LENGTH),
26     allowNull: false,
27     unique: true,
28     validate: {
29       len: {
30         msg: `login must has length between ${MIN_LOGIN_LENGTH} and ${MAX_LOGIN_LENGTH}.`,
31         args: [
32           MIN_LOGIN_LENGTH,
33           MAX_LOGIN_LENGTH
34         ]
35       }
36     }
37   },
38   password: {
39     type: Sequelize.STRING(MAX_PASSWORD_LENGTH),
40     allowNull: false,
41     validate: {
42       len: {
43         msg: `password must has length between ${MIN_PASSWORD_LENGTH} and
44 ↪ ${MAX_PASSWORD_LENGTH}.`,
45         args: [
46           MIN_PASSWORD_LENGTH,
47           MAX_PASSWORD_LENGTH
48         ]
49       }
50     }
51   },
52   email: {
53     type: Sequelize.STRING(MAX_EMAIL_LENGTH),
54     unique: true,
55     allowNull: false,
56     validate: {
57       isEmail: {
58         msg: 'invalid email address.'
59       }
60     }
61   },
62 });
63
64 User.hashPassword = async (value) => {
65   const salt = await bcrypt.genSalt(10);
66   return await bcrypt.hash(value, salt);
67 };
68
69 User.prototype.comparePasswords = async function(password) {
70   return await bcrypt.compare(password, this.password);
71 };
72
73 User.prototype.initState = function() {
74   const { login } = this;
75   return {
76     user: { login }
77   };
78

```

```
79     }
80   };
81
82   User.beforeCreate(async user => {
83     user.password = await User.hashPassword(user.password);
84   });
85
86   User.beforeUpdate(async user => {
87     user.password = await User.hashPassword(user.password);
88   });
89
90   User.hasMany(Test);
91   Test.belongsTo(User);
92
93   export default User;
```

---

db-course-project-app/src/models/User/index.js

---

```
1 import User from "../User";
2
3 export default User;
```

---

db-course-project-app/db/init\_db.sql

---

```
1 CREATE TABLE IF NOT EXISTS users (
2   user_id SERIAL NOT NULL,
3   login VARCHAR(255) NOT NULL UNIQUE,
4   password VARCHAR(255) NOT NULL,
5   email VARCHAR(255) NOT NULL,
6   PRIMARY KEY (user_id)
7 );
```

---

## Список использованных источников

- [1] *CSS*. URL: <https://ru.wikipedia.org/?oldid=107701928>.
- [2] *ECMAScript*. URL: <https://ru.wikipedia.org/?oldid=108101383>.
- [3] *ESLint*. URL: <https://eslint.org>.
- [4] *Express*. URL: <https://expressjs.com>.
- [5] *Git*. URL: <https://git-scm.com>.
- [6] *Heroku*. URL: <https://heroku.com>.
- [7] *HTML*. URL: <https://developer.mozilla.org/en-US/docs/Web/HTML>.
- [8] *JavaScript*. URL: <https://ru.wikipedia.org/?oldid=107293496>.
- [9] *Jest*. URL: <https://jestjs.io>.
- [10] *MongoDB*. URL: <https://www.mongodb.com/>.
- [11] *Node.js*. URL: <https://nodejs.org>.
- [12] *PostgreSQL*. URL: <https://www.postgresql.org>.
- [13] *Python*. URL: <https://www.python.org>.
- [14] *React*. URL: <https://reactjs.org>.
- [15] *React Router*. URL: <https://reactrouter.com>.
- [16] *Redux*. URL: <https://redux.js.org>.
- [17] *SCSS*. URL: <https://sass-lang.com>.
- [18] *Selenium with Python*. URL: <https://selenium-python.readthedocs.io>.
- [19] *Sequelize*. URL: <https://sequelize.org>.
- [20] *Stylelint*. URL: <https://stylelint.io>.
- [21] *Webpack*. URL: <https://webpack.js.org>.