

Министерство образования и науки Российской Федерации
Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий
Высшая школа программной инженерии

Курсовой проект

по дисциплине «Объектно-ориентированное программирование»

Выполнил
студент группы 3530904/80001

<подпись>

Проверил
преподаватель

В.В. Гладун

Д.С. Эйзенах

<подпись>

Санкт-Петербург
2020

Содержание

| | |
|------------------------------|----|
| Задание | 2 |
| Описание архитектуры | 3 |
| Структура классов back-end | 4 |
| Схема базы данных | 5 |
| Описание REST API | 6 |
| Security | 15 |
| Назначение классов front-end | 16 |
| Интерфейс | 17 |
| Заключение | 21 |

Задание

Разработать клиент-серверное приложение на заданную тему (автоматизация работы оптовой фирмы). Список обязательных к использованию технологий при выполнении работы:

Клиент:

требования отсутствуют, можно использовать любые языки

Сервер:

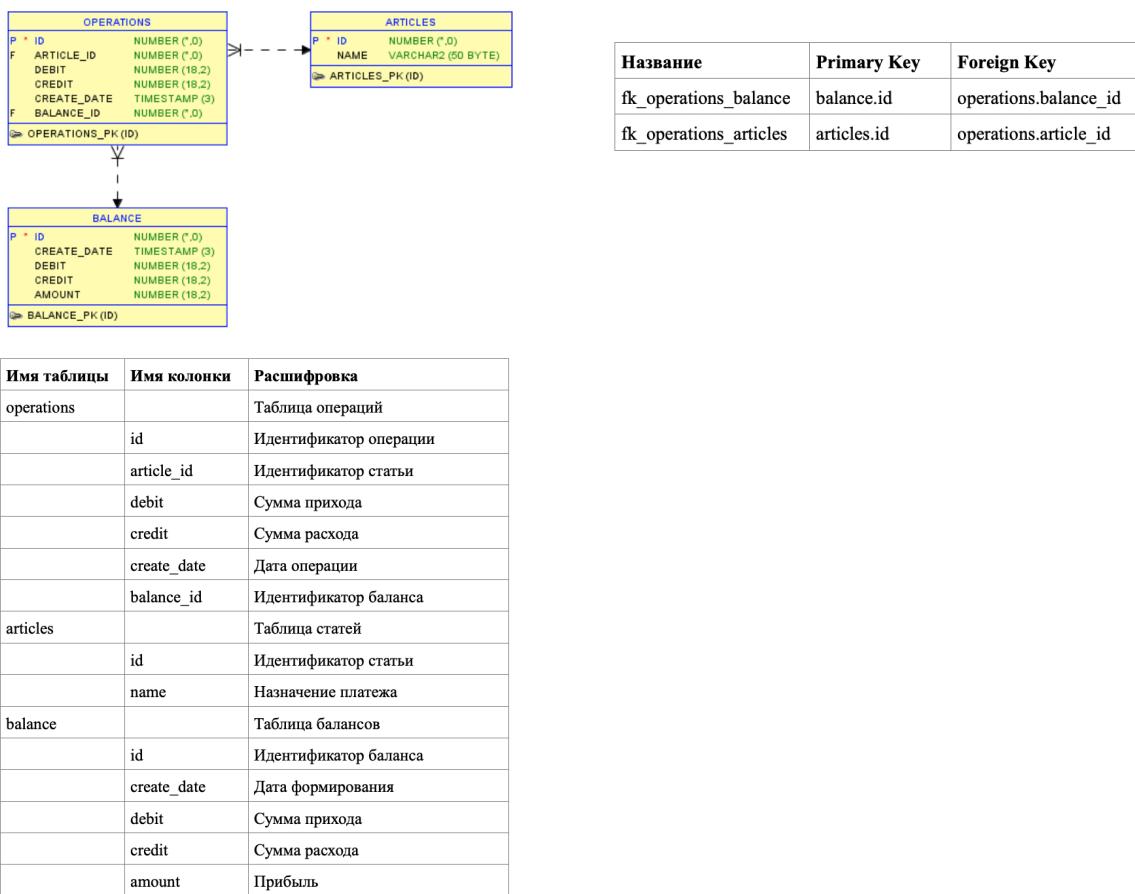
Java 12 и выше

База данных на выбор - SQLite/PostgreSQL/MS SQL/Oracle. Схему БД, соответствующую заданию, см. ниже.

SpringData/Hibernate для работы с БД

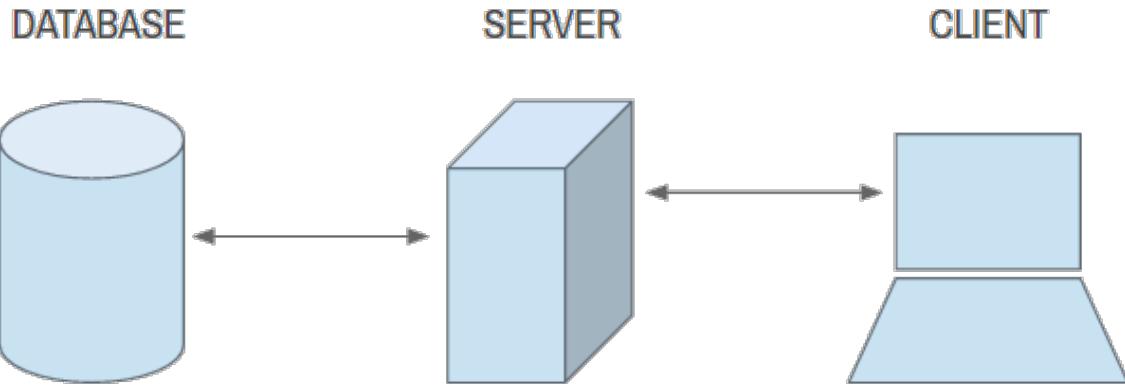
Spring Security

Взаимодействие с клиентом осуществляется посредством REST API.

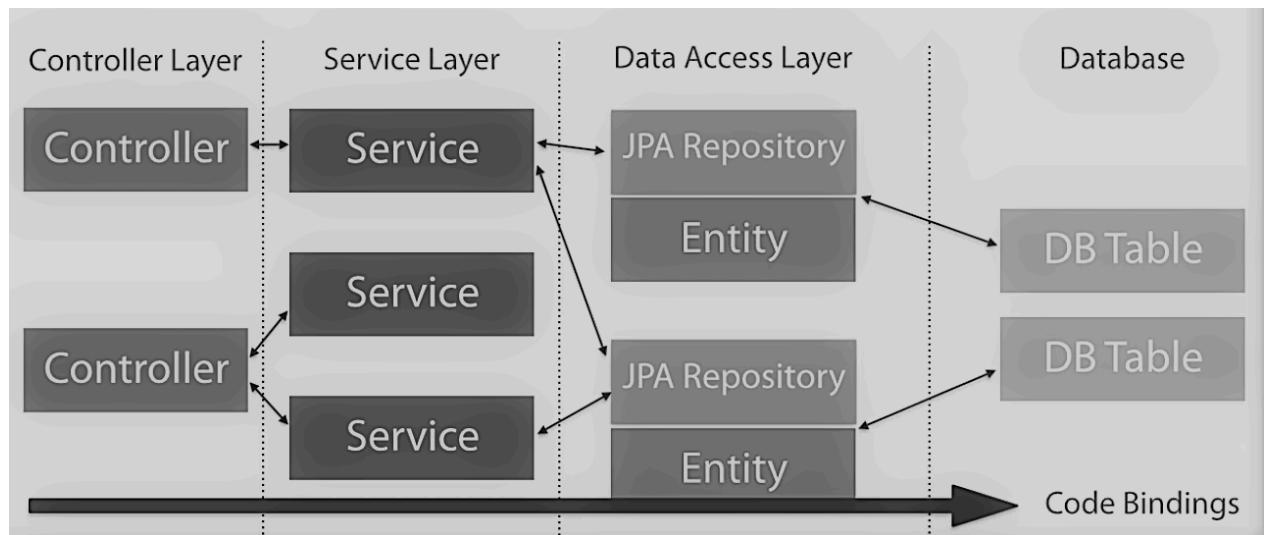


Описание архитектуры

В приложении использована классическая архитектура клиент-серверного приложения.



Слои доступа к базе данных (структура всего back-end) выглядит следующим образом:



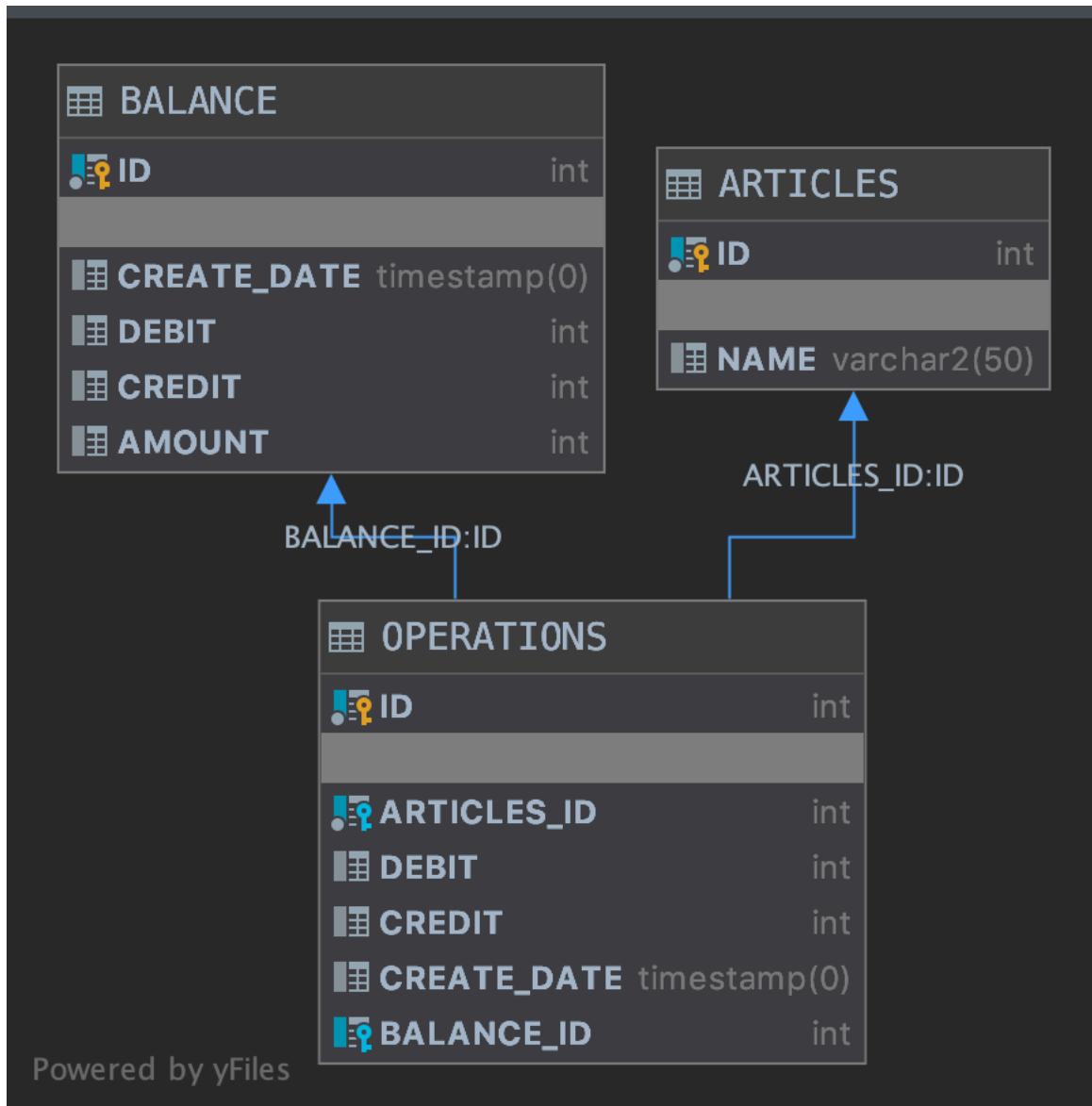
Структура классов back-end



Back-end включает в себя следующие основные пакеты:

- сущности, отражающие таблицы в базе данных
 - репозитории для взаимодействия с базой данных
 - сервисы для обработки данных, полученных из репозиториев и поступающих в них из контроллеров
 - контроллеры для взаимодействия с front-end посредством REST-запросов
 - структуры «response» для некоторых контроллеров чтобы обеспечить правильную интерпретацию данных, поступивших в запросе
 - файлы конфигурации, которые содержат необходимые настройки Spring Security и пути к файлам front-end.

Схема базы данных



Описание REST API

| Endpoint | Method | Input | Output |
|-------------------|--------|--|---|
| /api/article | Get | None | { "id": 0, "name": "string" } |
| /api/article | Post | { "id": 0, "name": "string" } | { "success": true, "error": "" } |
| /api/article/{id} | Post | None | { "success": true, "error": "" } |
| /api/article/{id} | Get | None | { "id": 0, "name": "string" } |

| Endpoint | Method | Input | Output |
|-------------------|--------|---|--|
| /api/article/{id} | Put | { "id": 0, "name": "string" } | { "success": true, "error": "" } |
| /api/balance | Get | None | [{ "amount": 0, "createDate": "2020-05-31T13:18:35.345 Z", "credit": 0, "debit": 0, "id": 0 }] |
| /api/balance | Post | { "amount": 0, "createDate": "2020-05- 31T13:18:35.352Z", "credit": 0, "debit": 0, "id": 0 } | { "success": true, "error": "" } |

| Endpoint | Method | Input | Output |
|-------------------|--------|---|---|
| /api/balance/{id} | Post | None | { "success": true, "error": "" } |
| /api/balance/{id} | Get | None | { "amount": 0, "createDate": "2020-05-31T13:18:35.369Z", "credit": 0, "debit": 0, "id": 0 } |
| /api/balance/{id} | Put | { "amount": 0, "createDate": "2020-05-31T13:18:35.376Z", "credit": 0, "debit": 0, "id": 0 } | { "success": true, "error": "" } |

| Endpoint | Method | Input | Output |
|----------------|--------|--------------|---|
| /api/operation | Get | None | <pre>{ "article": { "id": 0, "name": "string" }, "createDate": "2020-05-31T13:18:35.414Z", "credit": 0, "debit": 0, "id": 0 }</pre> |
| /api/operation | Post | <pre>}</pre> | <pre>{ "article": { "id": 0, "name": "string" }, "createDate": "2020-05-31T13:18:35.417Z", "credit": 0, "debit": 0, "id": 0 }</pre> |

| Endpoint | Method | Input | Output |
|---------------------|--------|--|---|
| /api/operation/{id} | Get | None | <pre>{ "article": { "id": 0, "name": "string" }, "createDate": "2020-05-31T13:18:35.429Z", "credit": 0, "debit": 0, "id": 0 }</pre> |
| /api/operation/{id} | Put | <pre> } "article": { "id": 0, "name": "string" }, "createDate": "2020-05-31T13:18:35.435Z", "credit": 0, "debit": 0, "id": 0 }</pre> | <pre>{ "success": true, "error": "" }</pre> |

| Endpoint | Method | Input | Output |
|----------|--------|-------|---|
| /user | Get | None | <pre> } "accountNonExpired": true, "accountNonLocked": true, "authorities": [{ "authority": "string" }], "credentialsNonExpired": true, "enabled": true, "id": 0, "password": "string", } </pre> |

| Endpoint | Method | Input | Output |
|----------|--------|--|---|
| /user | Post | <pre> } "accountNonExpired": true, "accountNonLocked": true, "authorities": [{ "authority": "string" }, "credentialsNonExpired": true, "enabled": true, "id": 0, "password": "string", } </pre> | <pre> { "success": true, "error": "" } </pre> |

| Endpoint | Method | Input | Output |
|-------------------|--------|-------|--|
| /user/all | Get | None | { "credentialsNonExpired": true, "accountNonExpired": true, "accountNonLocked": true, "authorities": [{ "authority": "string" }], "enabled": true, "id": 0, "password": "string", "} } |
| /api/article/{id} | Delete | None | { "success": true, "error": "" } |

| Endpoint | Method | Input | Output |
|----------------------|--------|-------|---|
| /api/balance/{id} | Delete | None | <pre>{ "success": true, "error": "" }</pre> |
| /api/operation/ {id} | Delete | None | <pre>{ "success": true, "error": "" }</pre> |

Security

Шифрование

Защита осуществляется посредством Basic Authorization. Юзеры с паролями хранятся в зашифрованном виде. Использован bCryptPasswordEncoder.

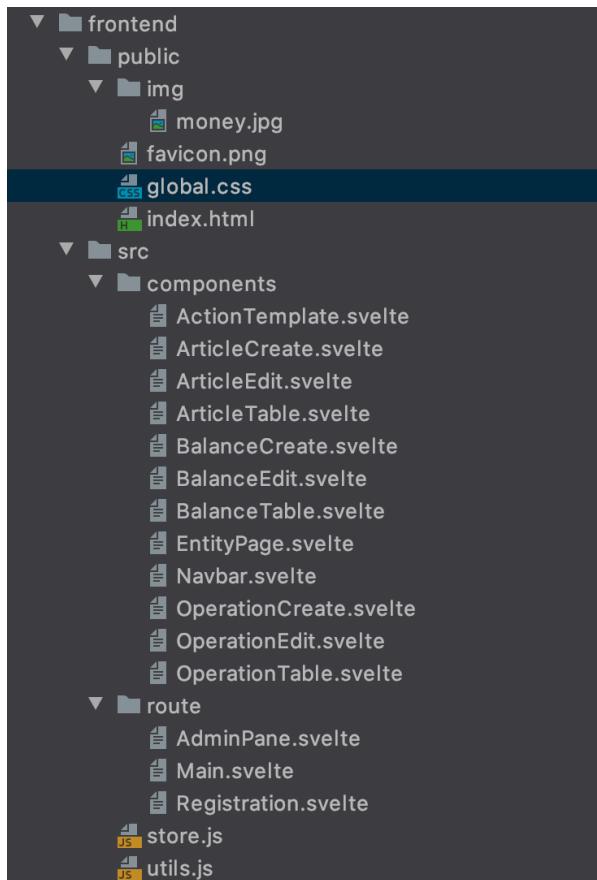
Права

В данном проекте существуют обычные пользователи и администратор. Администратору доступна страница, недоступная для обычных пользователей: список пользователей.

Назначение классов front-end

Структура

Front-end находится в каталоге «frontend».



Файл css содержит стили страниц, написанных на языке CSS.

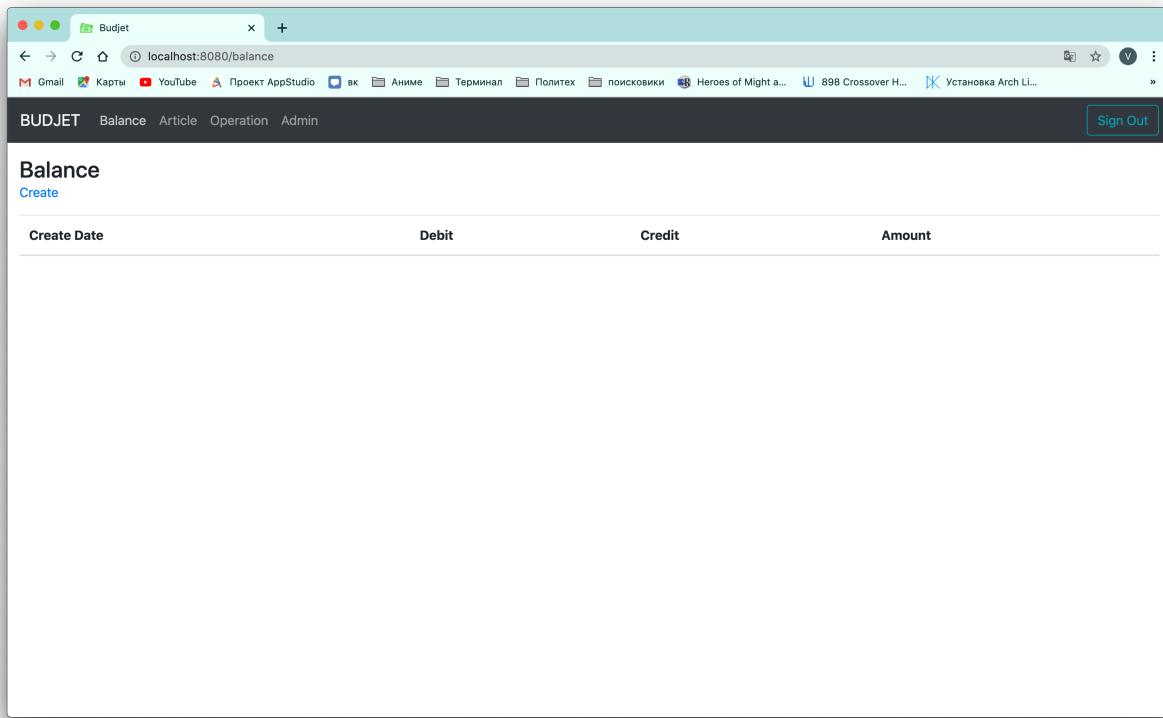
В основном для реализации Front-end был использован Фреймворк Svelte в силу своей простоты.

Интерфейс

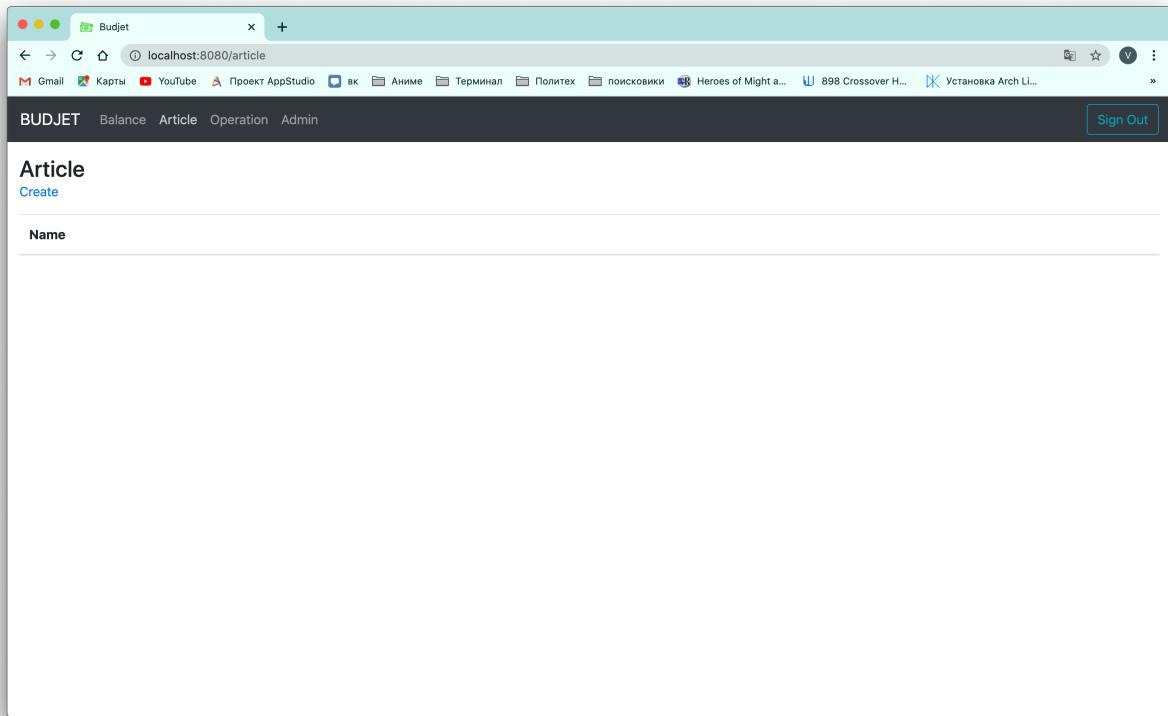
В данном разделе находятся скриншоты интерфейса, реализованных в папке «frontend».

Главная страница

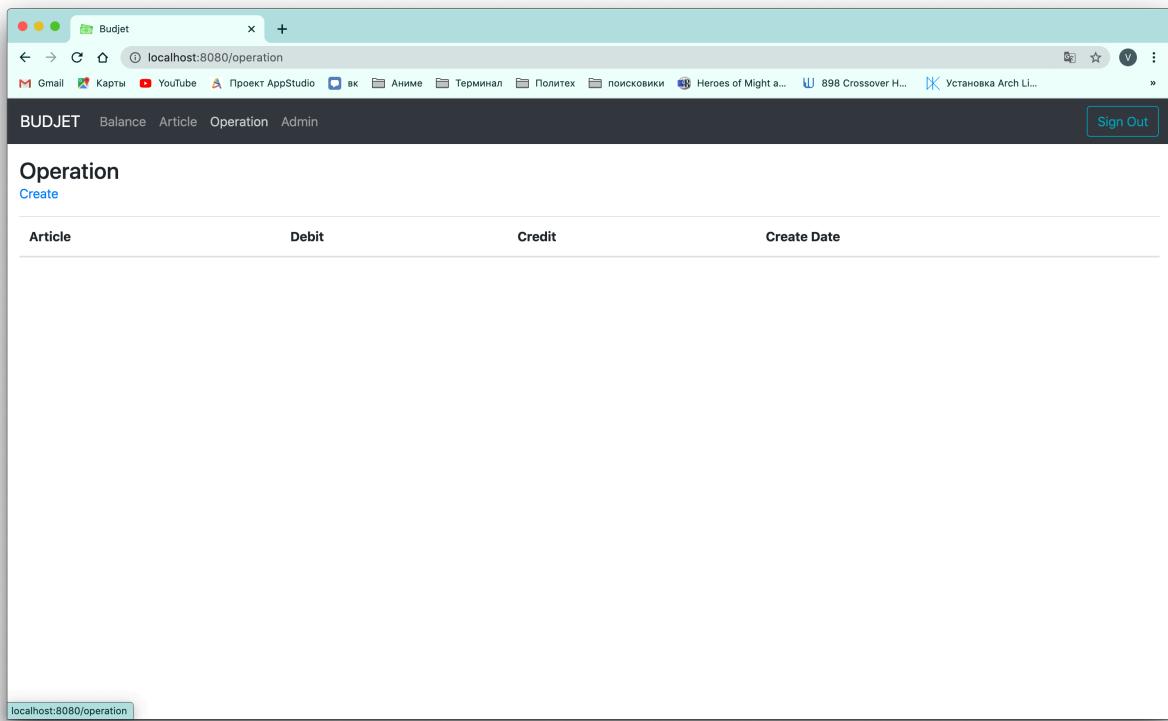
Открыта вкладка «Balance»



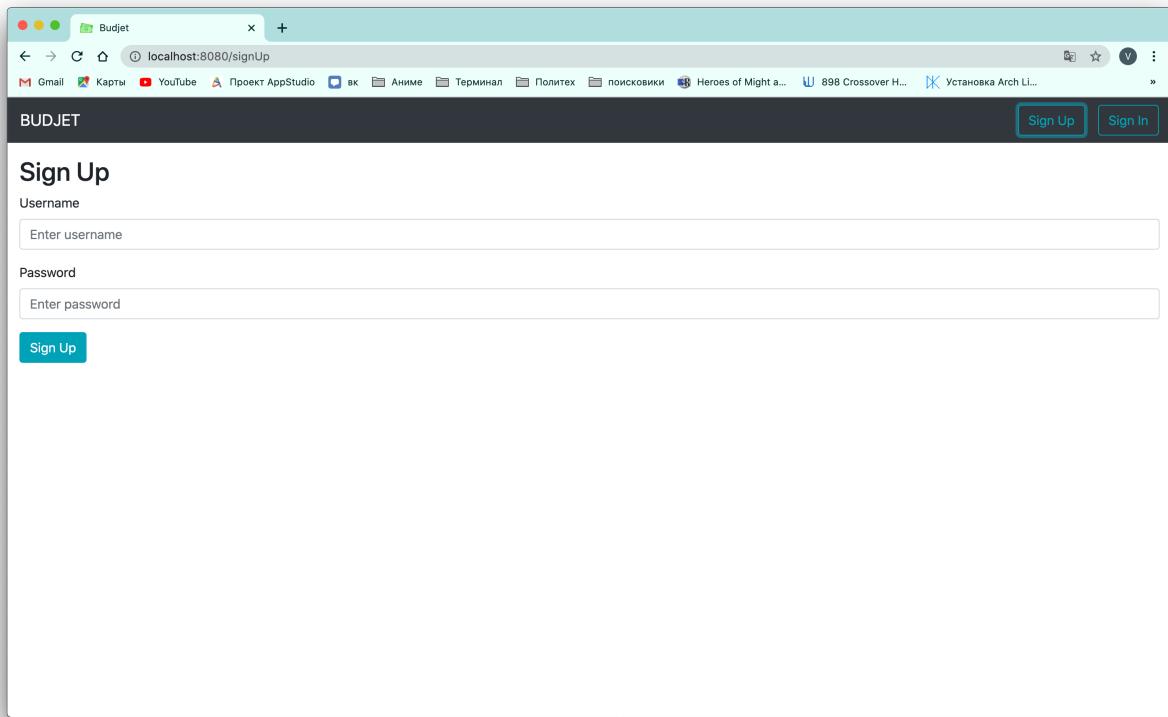
Открыта вкладка «Article»



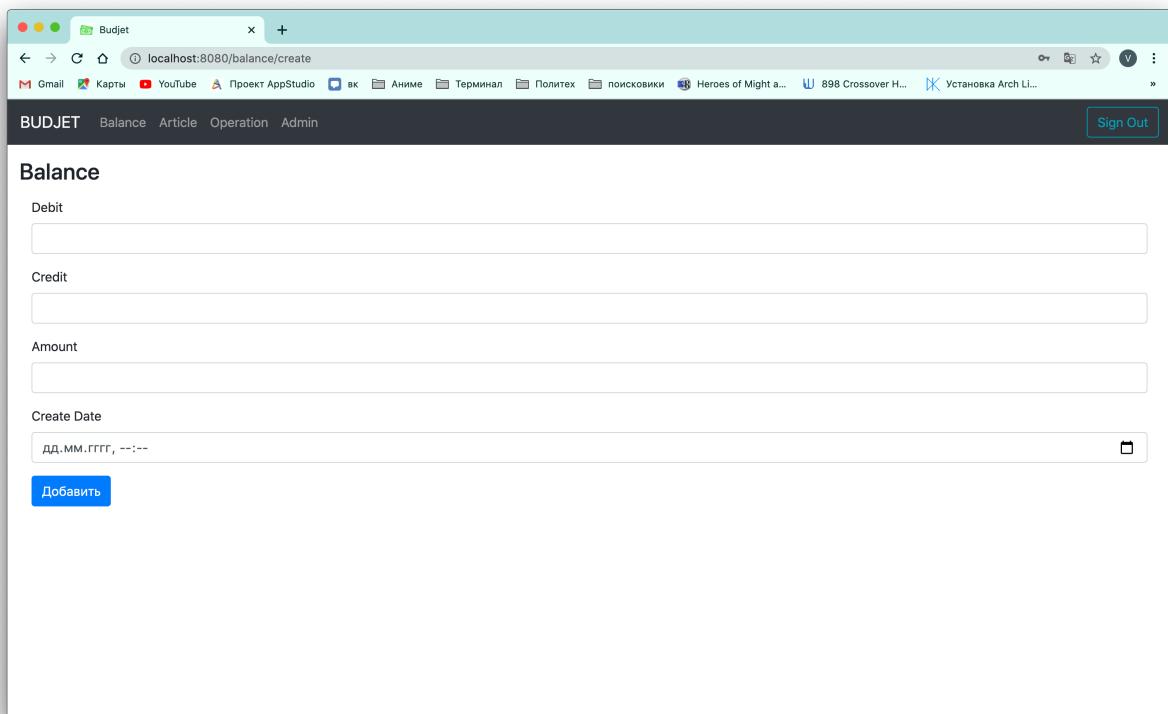
Открыта вкладка «Operation»



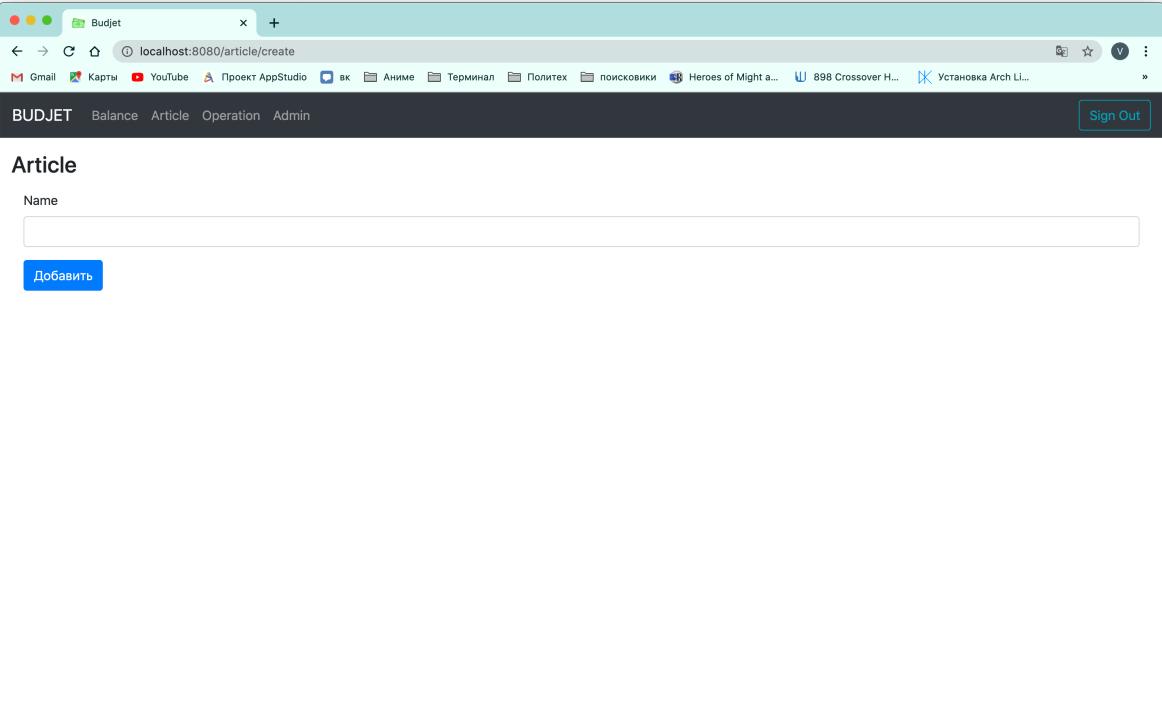
Регистрация



Редактирование «Balance»

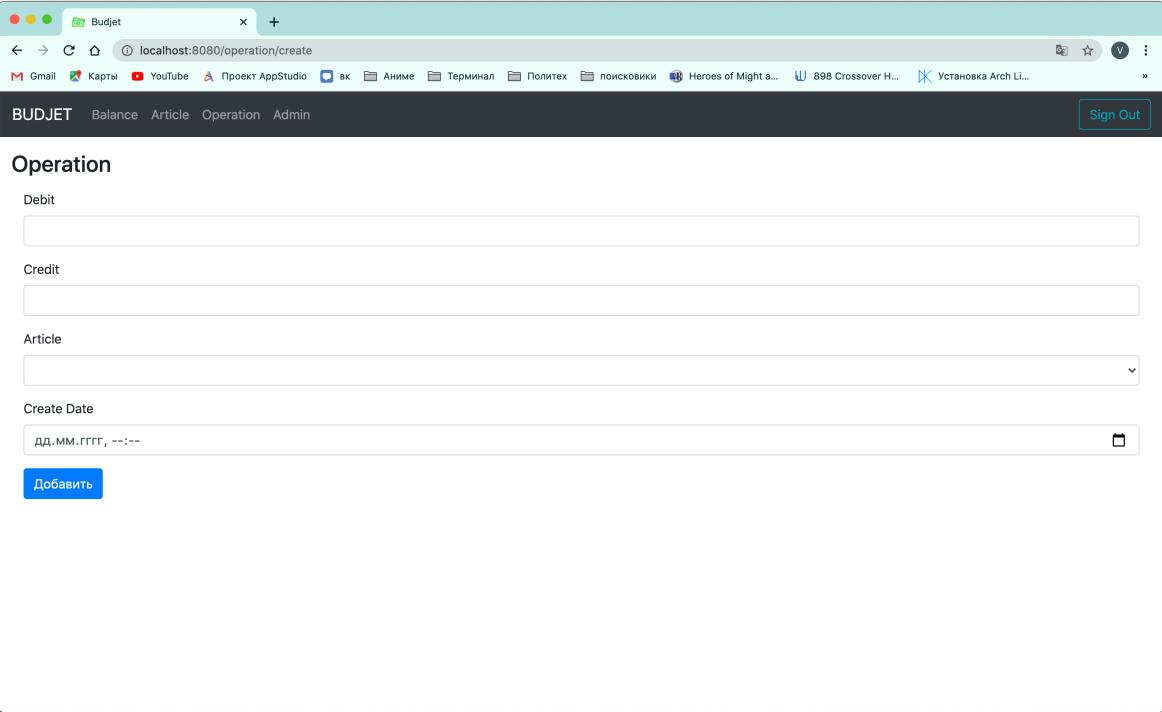


Редактирование «Article»



The screenshot shows a web browser window with the title 'Budjet' and the URL 'localhost:8080/article/create'. The page has a dark header with the text 'BUDJET' and navigation links for 'Balance', 'Article', 'Operation', and 'Admin'. On the right side of the header is a 'Sign Out' button. The main content area is titled 'Article' and contains a single input field labeled 'Name' with a placeholder 'Name'. Below the input field is a blue 'Добавить' (Add) button. The browser's address bar shows the URL 'localhost:8080/article/create'.

Редактирование «Operation»



The screenshot shows a web browser window with the title 'Budjet' and the URL 'localhost:8080/operation/create'. The page has a dark header with the text 'BUDJET' and navigation links for 'Balance', 'Article', 'Operation', and 'Admin'. On the right side of the header is a 'Sign Out' button. The main content area is titled 'Operation' and contains four input fields: 'Debit' (with a placeholder 'Debit'), 'Credit' (with a placeholder 'Credit'), 'Article' (with a dropdown menu), and 'Create Date' (with a date input field showing 'ДД.ММ.ГГГГ, --:--'). Below these fields is a blue 'Добавить' (Add) button. The browser's address bar shows the URL 'localhost:8080/operation/create'.

Заключение

В ходе работы было выполнено поставленное задание с использованием всех требуемых технологий.

Приложение

Исходный код доступен в репозитории: <https://github.com/GladunVladimir/KursJava>