**МиНобрнауки россии**

федеральное государственное бюджетное образовательное учреждение

высшего образования

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ**

**УНИВЕРСИТЕТ им. В.Г. ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Институт *Энергетики, информационных технологий и управляющих систем*

Кафедра *Программного обеспечения вычислительной техники и автоматизированных систем*

Направление подготовки *09.03.04 Программная инженерия*

(шифр, наименование)

Направленность (профиль) образовательной программы *Разработка программно-информационных систем*

(наименование)

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

на тему:

***Разработка программной системы автоматизации бюджетных процессов в области энергосбережения***

**Студент (ка) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Харитонов Сергей Дмитриевич

**Зав. кафедрой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**канд. техн. наук, доц. Поляков В.М.

**Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**ст.пр. Буханов Д. Г.

**К защите допустить**

**Зав. кафедрой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/** *Поляков В.М .***/**

**«\_\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.**

**Белгород 2020 г.**

**МиНобрнауки россии**

федеральное государственное бюджетное образовательное учреждение

высшего образования

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ**

**УНИВЕРСИТЕТ им. В.Г. ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Институт  *Энергетики, информационных технологий и управляющих систем*

Кафедра *Программного обеспечения вычислительной техники и автоматизированных систем*

Направление подготовки *09.03.04 Программная инженерия*

(шифр, наименование)

Направленность (профиль) образовательной программы *Разработка программно-информационных систем*

(наименование)

|  |  |
| --- | --- |
|  | Утверждаю:  Зав. кафедрой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2020г. |

**ЗАДАНИЕ**

на выпускную квалификационную работу студента (ки)

*Харитонова Сергея Дмитриевича*

(Фамилия Имя Отчество)

1. Вид выпускной квалификационной работы (ВКР) *бакалаврская работа*

2. Тема ВКР *РАЗРАБОТКА ПРОГРАММНОЙ СИСТЕМЫ АВТОМАТИЗАЦИИ БЮДЖЕТНЫХ ПРОЦЕССОВ В ОБЛАСТИ ЭНЕРГОСБЕРЕЖЕНИЯ*

утверждено приказом по университету от *«27» января 2020 г. № 2/47*

3. Срок сдачи студентом законченной ВКР *22.06.2020*

4. Исходные данные *учёт, автоматизация учёта производства, Java Spring, PostgreSQL, Java Persistence API, Web API, REST, SOAP, Data Transfer Object*

5. Содержание ВКР (перечень подлежащих разработке разделов)

*Производственный учёт кондитерских изделий;*

*Проектирование серверной части программного обеспечения учёта производства*

*кондитерских изделий;*

*Разработка серверной части программного обеспечения учёта производства кондитерских*

*изделий.*

6. Перечень графического материала *Титульный слайд, Цель и задачи, Актуальность работы,*

*Учёт производства на АО «КФ «Белогорье», Программные аналоги, Общая архитектура приложения, Средства разработки, Схема взаимодействия сервисов приложения, База данных «Цех», Диаграмма классов (model) «Цех», Диаграмма классов (controller) «Цех», Основные*

*запросы к системе, Основные запросы к системе, Итоги*

Консультанты по работе с указанием относящихся к ним разделов

|  |  |  |  |
| --- | --- | --- | --- |
| Раздел | Консультант | Задание выдал  (подпись, дата) | Задание принял  (подпись, дата) |

Дата выдачи задания «\_\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись руководителя)

Задание принял к исполнению

(подпись студента)

**КАЛЕНДАРНЫЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование этапов работы | Срок выполнения  этапов работы | Примечание |
| **1** | Постановка задачи, описание предметной области, изучение процессов учёта производства АО «Кондитерская фабрика «Белогорье» | *27.01.2020 – 16.03.2020* | *Выполнено* |
| **2** | Прoектирoвaние серверной части программного обеспечения учёта производства кондитерских изделий | *16.03.2020 – 13.04.2020* | *Выполнено* |
| **3** | Разработка и тестирование программного обеспечения учёта производства кондитерских изделий | *14.05.2020 – 15.06.2020* | *Выполнено* |
| **4** | Оформление пояснительной записки, презентации | *16.06.2020 – 19.06.2020* | *Выполнено* |

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Харитонов Сергей Дмитриевич

(подпись) (Ф.И.О.)

Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Буханов Дмитрий Геннадьевич.

(подпись)

**«Результаты проверки ЭВ ВКР на заимствование»**

Кафедра Программного обеспечения вычислительной техники и автоматизированных систем

Студент Харитонов С.Д. ПВ-42

Фамилия Инициалы Группа Дата

Тема ВКР Разработка серверной части программного обеспечения

учёта производства кондитерских изделий

ВКР прошла проверку на объем заимствований.

Итоговая оценка оригинальности:\_\_\_\_\_\_\_\_\_\_\_

Работу проверил Гончарук А.А.

Подпись Дата Фамилия Инициалы

Руководитель

ВКР доцент, канд. техн. наук Буханов Д.Г.

Должность, уч. степень, звание Подпись Дата Фамилия Инициалы

**OПРЕДЕЛЕНИЯ, СOКРAЩЕНИЯ И OБOЗНAЧЕНИЯ**

ПЭС – Программа ЭнергоСбережения

СOДЕРЖAние

[ВВЕДЕНИЕ 8](#_Toc95169016)

[1. Бюджетные процессы в области энергосбережения 9](#_Toc95169017)

[1.1. Программа энергосбережения 10](#_Toc95169018)

[2.1.1. Описание сервиса «Справочник сырья» 10](#_Toc95169019)

[ЗAКЛЮЧЕНИЕ 11](#_Toc95169020)

[СПИСOК ЛИТЕРAТУРЫ 12](#_Toc95169021)

[ПРИЛOЖЕНИЕ A 15](#_Toc95169022)

[ПРИЛOЖЕНИЕ Б 17](#_Toc95169023)

[ПРИЛOЖЕНИЕ В 31](#_Toc95169024)

[ПРИЛOЖЕНИЕ Г 35](#_Toc95169025)

[ПРИЛOЖЕНИЕ Д 44](#_Toc95169026)

[ПРИЛOЖЕНИЕ Е 45](#_Toc95169027)

# ВВЕДЕНИЕ

# 1. Бюджетные процессы в области энергосбережения

В современном мире значительную роль в быту и ежедневной рутине играют энергоресурсы. Трудно представить кухню без газовой плиты, жилую комнату без электрического освещения, а дом без системы отопления. Для доставки энергоресурсов конечным потребителям существует огромная электроэнергетическая система.

Электроэнергетическая система представляет из себя совокупность объектов электроэнергетики и энергопринимающих установок потребителей̆ электрической̆ энергии, связанных общим режимом работы в едином технологическом процессе производства, передачи и потребления электрической̆ энергии в условиях централизованного оперативно-диспетчерского управления в электроэнергетике. [https://www.elec.ru/viewer?url=/files/2021/09/27/GOST-21027-2021.pdf]

Данная система имеет огромное влияние на работу экономики страны и благосостояние ее граждан, именно поэтому нуждается в государственном регулировании. Согласно энергетической стратегии России одной из главных задач государственного регулирования в сфере энергетики является повышение энергетической и экологической эффективности российской экономики и энергетики, в том числе за счет структурных изменений и активизации технологического энергосбережения.[ https://centerpolit.org/national-security/energeticheskaya-strategiya-rossii-na-period-do-2030-goda/]

Особенно пристального внимания заслуживают объекты бюджетного сектора, такие как объекты культуры, медицины, спорта, занятости населения и другие, финансируемые из бюджета страны. Для их контроля в области потребления энергоресурсов используются следующие бюджетные процессы:

* Расчет потенциала энергосбережения и целевых уровней снижения потребления энергоресурсов [https://energy.midural.ru/wp-content/uploads/2020/09/PMER\_15\_07\_2020\_425.pdf]
* Представление деклараций о потреблении энергетических ресурсов [https://set.rk.gov.ru/uploads/txteditor/set/attachments/d4/1d/8c/d98f00b204e9800998ecf8427e/phpjQqoEY\_707.pdf]
* Разработка и исполнение программ энергосбережения[https://www.consultant.ru/document/cons\_doc\_LAW\_93978/]

В данной работе речь пойдет о программах энергосбережения и автоматизации их разработки.

## Программа энергосбережения

Одним из инструментов повышения эффективности использования энергоресурсов является разработка и реализация программ энергосбережения.

Согласно, федеральному закону «Об энергосбережении и о повышении энергетической эффективности и о внесении изменений в отдельные законодательные акты Российской Федерации» каждая бюджетная организация обязана разрабатывать и актуализировать ПЭС. Также помимо бюджетных организаций ПЭС разрабатывают муниципалитеты и субъекты РФ. [https://www.consultant.ru/document/cons\_doc\_LAW\_93978/]

Основной целью ПЭС является сокращение потребления используемых энергоресурсов, а также полное использование потенциала энергосбережения организации. Для этого на основе данных об организации и потребляемых энергоресурсах составляется план внедрения мероприятий, по энергосбережению и повышению энергетической эффективности, на ближайшие 3-5 лет. В ходе исполнения плана проводятся различные мероприятия: от сугубо технических (установка приборов учета) до организационных (проведение инструктажей и обучение персонала), нацеленные на более эффективное использование энергоресурсов.

Так же в ходе разработки ПЭС формируется паспорт программы энергосбережения и ежегодная промежуточная отчетность.

Паспорт ПЭС содержит информацию об организации, ее имуществе (здания, транспорт, приборы учета и другое) и план внедрения энергосберегающих мероприятий. Исходя из плана внедрения мероприятий и потребления энергоресурсов за предшествующие года вычисляется плановое ежегодное потребление, которого организация должна достичь в ходе реализации ПЭС.

Ежегодная промежуточная отчетность должна содержать информацию о реализации энергосберегающих мероприятий и сравнение планового потребление энергоресурсов с фактическим. На основании промежуточной отчетности ПЭС может корректироваться для лучшего достижения энергосберегающего эффекта.

# ЗAКЛЮЧЕНИЕ

# СПИСOК ЛИТЕРAТУРЫ

1. Экономическая теория / ред. Видяпин В.И. Добрынин А.И., Журавлёва Г.П., Тарасевич Л.С. – Москва : ИНФРА-М, 2019. – 714 с. – ISBN 978-5-16-003507-9
2. Бурмистров А.Н. Формирование модернизационной стратегии развития предприятий пищевой промышленности / авт. Бурмистров А.Н. – Саратов, 2011. – 162 с.
3. ГОСТ Р 53041-2008. Изделия кондитерские и полуфабрикаты кондитерского производства [Текст]. – Москва: Стандартинформ, 2019. – 16 с.
4. Кондитерская промышленность. [Электронный ресурс]. Кондитерская промышленность на территории России. – Электрон, дан., 2019. – Режим доступа: <https://fabricators.ru/article/konditerskaya-promyshlennost>
5. Бутейкис Н. Г. Технология приготовления мучных кондитерских изделий / Н. Г. Бутейкис, А.А. Жукова – 3-е изд., стереотипное – Москва: ACADEMA, 2015. – 302 с. – ISBN 978-5-7695-6733-9
6. Толкаченко, А. А. Правовая (судебная) бухгалтерия: курс лекций / А. А. Толкаченко, К. В. Харабет; ред. Н. А. Петухов. – 3-е изд. – Москва: Омега-Л, 2004. – 216 с. – (Библиотека высшей школы). – ISBN 5-98119-281-0
7. Оперативный учёт. [Электронный ресурс]. Библиотека банковских документов. – Электрон, дан., 2020. – Режим доступа: http://www.bbdoc.ru
8. Хозяйственный учёт. [Электронный ресурс] Хозяйственный учёт, его сущность и значение. – Электрон, дан., 2017. – Режим доступа: https://buhbook.net/buhgalterskij-uchet/teoriya-buhgalterskogo-ucheta/hozyaystvennyiy-uchet-ego-sushhnost-i-znachenie/
9. Оперативный учёт: цели, особенности, измерители. [Электронный ресурс] Оперативный учёт. – Электрон, дан., 2017. – Режим доступа: <https://www.audit-it.ru/terms/accounting/operativnyy_uchet.html>
10. Оперативный учёт. [Электронный ресурс]. Оперативный учёт. Оперативный счёт. – Электрон, дан., 2020. ­ – Режим доступа: <https://bankfs.ru/deposits/operativnyi-schet-operativnyi-uch-t-buhgalterskii-uchet-v.html>
11. Кузнецова Л. С Технология приготовления мучных кондитерских изделий: учебн. для студ. учреждений сред. проф. образования. / Л.С. Кузнецова, М. Ю Сиданова – Москва: ACADEMIA, 2013. – 320 с. – ISBN 978-5-7695-9525-7
12. ERP-система. [Электронный ресурс]. Что такое ERP-система. – Электрон, дан., 2017. ­ – Режим доступа: <https://trinion.org/articles/chto-takoe-erp-sistema>
13. Корякин А. О. Разработка клиентской части программного обеспечения учёта производства кондитерских изделий [Отчет]. – Белгород, 2020.
14. Фаулер М. Архитектура коропративных программных приложений / М. Фаулер – Москва : Издательский дом «Вильяме», 2018. – 540 с. – ISBN 978-5-6041394-0-0
15. Язык программирования Java SE 8. Подробное описание / Д. Гослинг, Б. Джой, Г. Стил и др. – 5-е изд. – Москва: «Вильямс», 2015. – 672 с. – ISBN 978-5-8459-1875-8
16. Маншин Т. С. Web-сервисы Java / Т. С. Маншин – Санкт-Петербург: БХВ-Петербург, 2012. – 560 с. – ISBN 978-5-9775-0778-3
17. Webber J. REST in Practice / J. Webber, S. Parastatidis S., I. Robinson – O'Reilly Media, Inc., 2010. – First Edition – 448 c. – ISBN: 978-0-596-80582-1
18. Коннолли Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика / Т. Коннолли, К. Бегг / ред. Н. С. Тригуб – 3-е изд., – Москва: Вильямс, 2017. – 1440 с. – ISBN 978-5-8459-2020-1
19. Дейт К. Введение в системы баз данных / К. Дейт / ред. К.А. Птицына / перев. К.А. Птицына. – Москва: Диалектика, 2019. – 1328 c. – ISBN 978-5-8459-0788-2
20. PostgreSQL. [Электронный ресурс]. Manual Archive. – Электрон, дан. 2020. ­ – Режим доступа: <https://www.postgresql.org/docs/manuals/archive/>
21. Национальная библиотека им. Н. Э. Баумана. [Электронный ресурс]. PostgreSQL. – Электрон, дан., 2019.  – Режим доступа: <https://ru.bmstu.wiki/PostgreSQL>
22. Уоллс К. Spring в действии / К. Уоллс / ред. Д.А. Мовчан. – Москва: ДМК Пресс, 2015. – 752 c. – ISBN 978-5-97060-171-6
23. Consul.io. [Электронный ресурс]. Habr: Consul.io, часть 1. — Электрон, дан., 2016. — Режим доступа:: <https://habr.com/en/post/278085>
24. DTO. [Электронный ресурс]. DTO vs POCO и Value Object . – Электрон, дан., 2016.  – Режим доступа: <https://habr.com/ru/post/268371/>
25. MapStruct. [Электронный ресурс]. MapStruct: Frequently Asked Questions – Электрон, дан., 2020.  – Режим доступа: <https://mapstruct.org/>
26. The Postman API Platform. [Электронный ресурс]. Create API Documentation with Postman. – Электрон, дан., 2020.  – Режим доступа: <https://postman.com/>

# ПРИЛOЖЕНИЕ A

**Сервер конфигураций**

**1. pom.xml – файл конфигурации**

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns="http://maven.apache.org/POM/4.0.0"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<GroupId>ru.tn</groupId> <artifactId>ConfigServer</artifactId>

<version>1.0-SNAPSHOT</version>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.2.2.RELEASE</version>

</parent>

<dependencies>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-consul-discovery</artifactId>

<version>2.2.2.RELEASE</version>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-actuator</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency> <groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-config-server</artifactId>

<version>2.2.2.RELEASE</version> </dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

**2. bootstrap.yaml – настройка сервера конфигураций**

server:

port:

8888

spring:

application:

name: ConfigServer

cloud:

consul:

host: ${DISCOVERY\_SERVER\_HOST:localhost}

port: ${DISCOVERY\_SERVER\_PORT:8500}

discovery:

instanceId: config-server-id

tags: env

port: ${SERVER\_PORT:8888}

serviceName: ${spring.application.name}

config:

server:

git:

uri: https://github.com/IOBratkova/pa-microservices-config

clone-on-start: true

management:

endpoints:

web:

exposure:

include: "\*"

# ПРИЛOЖЕНИЕ Б

**Сервис «Справочник сырья»**

**1. SQL-запросы на создание таблиц базы данных**

CREATE TABLE "public".controlled\_parameters (

id bigint PRIMARY KEY,

name varchar(100) NOT NULL,

"type" smallint NOT NULL );

CREATE TABLE "public".manufacturers (

id bigint PRIMARY KEY,

name varchar(50) NOT NULL );

CREATE TABLE "public".material\_categories (

id bigint PRIMARY KEY,

name varchar(40) NOT NULL );

CREATE TABLE "public".material\_type (

id bigint PRIMARY KEY,

measure varchar NOT NULL,

material\_category\_id bigint NOT NULL,

name varchar(50) NOT NULL );

CREATE TABLE "public".material\_types\_parameters (

id bigint PRIMARY KEY,

parameter\_id bigint NOT NULL,

type\_id bigint NOT NULL,

"source" smallint NOT NULL

CONSTRAINT idx\_material\_types\_parameters UNIQUE (parameter\_id, type\_id) );

CREATE TABLE "public".names (

id bigint PRIMARY KEY,

manufacturer\_id bigint NOT NULL,

material\_type\_id bigint NOT NULL,

name varchar(100),

"one\_c\_name" varchar(100) NOT NULL );

CREATE TABLE "public".norms (

id bigint PRIMARY KEY,

material\_type\_parameter\_id bigint,

"from\_value" decimal,

"to\_value" decimal,

CONSTRAINT idx\_norm UNIQUE (material\_type\_parameter\_id) );

CREATE TABLE "public".storage\_location (

id bigint PRIMARY KEY,

material\_type\_id integer NOT NULL,

name varchar(100) NOT NULL );

ALTER TABLE "public".material\_type

ADD CONSTRAINT fk\_material\_type

FOREIGN KEY (material\_category\_id) REFERENCES "public".material\_categories(id)

ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE "public".material\_types\_parameters

ADD CONSTRAINT fk\_material\_type\_parameter\_controlled\_parameter

FOREIGN KEY (parameter\_id) REFERENCES "public".controlled\_parameters(id)

ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE "public".material\_types\_parameters

ADD CONSTRAINT fk\_material\_type\_parameter\_material\_type

FOREIGN KEY (type\_id) REFERENCES "public".material\_type(id)

ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE "public".names

ADD CONSTRAINT fk\_names\_manufacturer

FOREIGN KEY (manufacturer\_id) REFERENCES "public".manufacturers(id)

ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE "public".names

ADD CONSTRAINT fk\_names\_material\_type

FOREIGN KEY (material\_type\_id) REFERENCES "public".material\_type(id)

ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE "public".norms

ADD CONSTRAINT fk\_norm\_material\_type\_parameter

FOREIGN KEY (material\_type\_parameter\_id)

REFERENCES "public".material\_types\_parameters(id)

ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE "public".storage\_location

ADD CONSTRAINT fk\_storage\_location

FOREIGN KEY (material\_type\_id)

REFERENCES "public".material\_type(id);

**2. material-directory-dev.yaml – настройка справочного сервиса сырья**

spring:

profiles:

active: dev

datasource:

username: postgres

password: root

url: jdbc:postgresql://localhost:5432/pa-material-directory-db

jpa:

database: POSTGRESQL

show-sql: true

liquibase:

changeLog: classpath:/db/changelog/db.changelog-master.xml

url: ${spring.datasource.url}

user: ${spring.datasource.username}

password: ${spring.datasource.password}

drop-first: true

server:

servlet:

session:

cookie:

secure: false

http-only: false

port: 8083

1. **Сущность Поставщик (Manufacturer)**

@Data

@Entity

@Table(name = "controlled\_parameters")

public class ControlledParameter {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name="name")

private String name;

@Column(name="type")

@Enumerated(EnumType.ORDINAL)

private ControlledParameterTypeEnum type;

}

1. **Сущность Поставщик (Manufacturer)**

@Data

@Entity

@Table(name = "manufacturers")

public class Manufacturer {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name="name")

private String name;

}

1. **Сущность Категория материалов (MaterialCategory)**

@Data

@Entity

@Table(name = "material\_categories")

public class MaterialCategory {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name="name")

private String name;

}

1. **Сущность Тип сырья (MaterialType)**

@Data

@Entity

@Table(name = "material\_type")

public class MaterialType {

@Id

@GeneratedValue

(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name="name")

private String name;

@Column(name="measure")

private String measure;

@ManyToOne

@JoinColumn(name = "material\_category\_id", referencedColumnName = "id")

private MaterialCategory materialCategory;

@OneToMany(mappedBy = "materialType")

private List<MaterialTypeParameter> materialTypeParameters;

@OneToMany(mappedBy = "materialType")

private List<StorageLocation> storageLocations;

}

1. **Сущность Параметры типа сырья (MaterialTypeParameter)**

@Data

@Entity

@Table(name = "material\_types\_parameters")

public class MaterialTypeParameter {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@ManyToOne

@JoinColumn(name = "type\_id")

private MaterialType materialType;

@ManyToOne

@JoinColumn(name = "parameter\_id", referencedColumnName = "id")

private ControlledParameter controlledParameter;

@Enumerated(EnumType.ORDINAL)

@Column(name = "source")

private SourceEnum sourceEnum;

@OneToOne(mappedBy = "materialTypeParameter")

private Norm norm;

}

1. **Сущность Норма (Norm)**

@Data

@Entity

@Table(name = "norms")

public class Norm {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name="from\_value")

private BigDecimal valueFrom;

@Column(name="to\_value")

private BigDecimal valueTo;

@OneToOne

@JoinColumn(name = "material\_type\_parameter\_id")

private MaterialTypeParameter materialTypeParameter;

}

1. **Сущность Место хранения (StorageLocation)**

@Data

@Entity

@Table(name = "storage\_location")

public class StorageLocation {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name="name")

private String name;

@ManyToOne

@JoinColumn(name = "material\_type\_id", referencedColumnName = "id")

private MaterialType materialType;

}

1. **Перечисление Тип параметра, Источник параметра**

public enum ControlledParameterTypeEnum { QUANTITATIVE, QUALITATIVE }

public enum SourceEnum { ANALYSES, CERTIFICATE }

1. **NameServiceImpl**

public class NameServiceImpl implements NameService {

private final NameRepository nameRepository;

private final NameMapper nameMapper;

private final ManufacturerService manufacturerService;

private final MaterialTypeService materialTypeService;

@Override

public Long create(NameDTO nameDTO) {

if (nameDTO == null)

throw new InvalidParametersException("Name");

Name name = nameMapper.toEntity(nameDTO);

return nameRepository.save(name).getId();

}

@Override

public Boolean delete(Long id) {

if(id == null)

throw new InvalidParametersException("Name");

if(nameRepository.existsById(id)){

nameRepository.deleteById(id);

return true;

}

throw new ResourceNotFoundException(id, "Name");

}

@Override

public Boolean update(NameDTO nameDTO, Long id) {

if(nameDTO == null || id == null)throw new InvalidParametersException("Name");

Name name = nameRepository.findById(id).orElseThrow(

() -> new ResourceNotFoundException(id, "Name"));

name.setName(nameDTO.getName());

name.setNameOneC(nameDTO.getNameOneC());

name.setManufacturer(manufacturerService.readEntity(

nameDTO.getManufacturer().getId()));

name.setMaterialType(materialTypeService.readEntity(

nameDTO.getMaterialType().getId()));

nameRepository.save(name);

return true;

}

@Override

public NameDTO read(Long id) {

if(id == null)throw new InvalidParametersException("Name");

return nameMapper.toDTO(readEntity(id));

}

@Override

public NameWithParametersDTO readWithParams(Long id) {

if(id == null)throw new InvalidParametersException("Name");

return nameMapper.toDTOwithParams(readEntity(id));

}

@Override

public Name readEntity(Long id) {

if(id == null)throw new InvalidParametersException("Name");

return nameRepository.findById(id).orElseThrow(

() -> new ResourceNotFoundException(id, "Name"));

}

@Override

public List<NameDTO> readAll() {

return nameMapper.toDTOs(nameRepository.findAll());

}

@Override

public List<NameDTO> readAllByMaterialType\_Id(Long id) {

if(id == null)throw new InvalidParametersException("Name");

return nameMapper.toDTOs(nameRepository.findAllByMaterialType\_Id(id));

}

@Override

public List<NameDTO> readAllByManufacturer\_Id(Long id) {

if(id == null)throw new InvalidParametersException("Name");

return nameMapper.toDTOs(nameRepository.findAllByManufacturer\_Id(id));

}

}

1. **MaterialTypeParameterDTO**

@Data

public class MaterialTypeParameterDTO {

private Long id;

private MaterialTypeDTO materialType;

private ControlledParameterDTO controlledParameter;

private SourceEnum sourceEnum;

private NormDTO norm; }

1. **MaterialTypeParameterGetDTO**

@Data

public class MaterialTypeParameterGetDTO {

private Long id;

private ControlledParameterDTO controlledParameter;

private SourceEnum sourceEnum;

private NormDTO norm;

}

1. **MaterialTypeWithParametersDTO**

@Data

public class MaterialTypeWithParametersDTO {

private Long id;

private String name;

private String measure;

private MaterialCategoryDTO materialCategory;

private List<MaterialTypeParameterGetDTO> materialTypeParameters;

}

1. **NameDTO**

@Data

@AllArgsConstructor

public class NameDTO {

private Long id;

private String name;

private String nameOneC;

private ManufacturerDTO manufacturer;

private MaterialTypeDTO materialType;

}

1. **NameWithParametersDTO**

@Data

public class NameWithParametersDTO {

private Long id;

private String name;

private String nameOneC;

private ManufacturerDTO manufacturer;

private MaterialTypeWithParametersDTO materialType;

}

1. **NormDTO**

@Data

public class NormDTO {

private Long id;

private BigDecimal valueFrom;

private BigDecimal valueTo;

}

1. **NameController**

@RestController

@RequiredArgsConstructor

@RequestMapping(Constants.NAME)

public class NameController {

private final NameService nameService;

@PostMapping

public Long create(@RequestBody NameDTO nameDTO) {

return nameService.create(nameDTO); }

@PutMapping

public Boolean update(@RequestBody NameDTO nameDTO, @RequestParam Long id){

return nameService.update(nameDTO, id); }

@GetMapping

public NameDTO read(@RequestParam Long id){

return nameService.read(id); }

@GetMapping(NAME\_WITH\_PARAMS)

public NameWithParametersDTO readWithParameters(@RequestParam Long id){

return nameService.readWithParams(id); }

@DeleteMapping

public Boolean delete(@RequestParam Long id){

return nameService.delete(id); }

@GetMapping(Constants.ALL)

public List<NameDTO> readAll() {

return nameService.readAll(); }

@GetMapping(Constants.NAME\_BY\_MATERIAL\_TYPE\_ID)

public List<NameDTO> readAllByMaterialTypeId(@RequestParam Long id){

return nameService.readAllByMaterialType\_Id(id); }

@GetMapping(Constants.NAME\_BY\_MANUFACTURER\_TYPE\_ID)

public List<NameDTO> readAllByManufacturerId(@RequestParam Long id){

return nameService.readAllByManufacturer\_Id(id); }

}

1. **Constants ­– класс, содержащий пути к контроллерам и их методам**

public class Constants {

public static final String API = "/api/v1";

@AllArgsConstructor

@Data

public static class MethodInfo{

AuthorityEnum autority;

TypeAccessEnum typeAccess;

String methoduri;

}

public static final String CONTROLLED\_PARAMETERS = API

+ "/controlled\_parameters";

public static final MethodInfo LAB\_ADMIN\_CONTROLLED\_PARAMETERS = new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY,

TypeAccessEnum.READ\_WRITE, CONTROLLED\_PARAMETERS);

public static final MethodInfo WORK\_ADMIN\_CONTROLLED\_PARAMETERS = new MethodInfo(

AuthorityEnum.ADMIN\_WORKSHOP,

TypeAccessEnum.READ, CONTROLLED\_PARAMETERS);

public static final MethodInfo LAB\_USER\_CONTROLLED\_PARAMETERS = new MethodInfo(

AuthorityEnum.USER\_LABORATORY,

TypeAccessEnum.READ, CONTROLLED\_PARAMETERS);

public static final String MANUFACTURERS = API + "/manufacturers";

public static final MethodInfo LAB\_ADMIN\_MANUFACTURERS = new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY,

TypeAccessEnum.READ\_WRITE, MANUFACTURERS);

public static final MethodInfo WORK\_ADMIN\_MANUFACTURERS = new MethodInfo(

AuthorityEnum.ADMIN\_WORKSHOP,

TypeAccessEnum.READ, MANUFACTURERS);

public static final MethodInfo LAB\_USER\_MANUFACTURERS = new MethodInfo(

AuthorityEnum.USER\_LABORATORY,

TypeAccessEnum.READ, MANUFACTURERS);

public static final String MATERIAL\_CATEGORY = API + "/material\_category";

public static final MethodInfo LAB\_ADMIN\_MATERIAL\_CATEGORY = new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY,

TypeAccessEnum.READ\_WRITE, MATERIAL\_CATEGORY);

public static final MethodInfo WORK\_ADMIN\_MATERIAL\_CATEGORY = new MethodInfo(

AuthorityEnum.ADMIN\_WORKSHOP,

TypeAccessEnum.READ, MATERIAL\_CATEGORY);

public static final MethodInfo LAB\_USER\_MATERIAL\_CATEGORY = new MethodInfo(

AuthorityEnum.USER\_LABORATORY,

TypeAccessEnum.READ, MATERIAL\_CATEGORY);

public static final String MATERIAL\_TYPE = API + "/material\_type";

public static final MethodInfo LAB\_ADMIN\_MATERIAL\_TYPE = new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY,

TypeAccessEnum.READ\_WRITE, MATERIAL\_TYPE);

public static final MethodInfo WORK\_ADMIN\_MATERIAL\_TYPE = new MethodInfo(

AuthorityEnum.ADMIN\_WORKSHOP,

TypeAccessEnum.READ, MATERIAL\_TYPE);

public static final MethodInfo LAB\_USER\_MATERIAL\_TYPE = new MethodInfo(

AuthorityEnum.USER\_LABORATORY,

TypeAccessEnum.READ, MATERIAL\_TYPE);

public static final String MATERIAL\_TYPE\_BY\_MATERIAL\_CATEGORY\_ID =

"/by\_material\_category\_id";

public static final MethodInfo LAB\_ADMIN\_MATERIAL\_TYPE\_BY\_MATERIAL\_CATEGORY\_ID =

new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY,TypeAccessEnum.READ,

MATERIAL\_TYPE + MATERIAL\_TYPE\_BY\_MATERIAL\_CATEGORY\_ID );

public static final MethodInfo WORK\_ADMIN\_MATERIAL\_TYPE\_BY\_MATERIAL\_CATEGORY\_ID =

new MethodInfo(

AuthorityEnum.ADMIN\_WORKSHOP,TypeAccessEnum.READ,

MATERIAL\_TYPE + MATERIAL\_TYPE\_BY\_MATERIAL\_CATEGORY\_ID );

public static final MethodInfo LAB\_USER\_MATERIAL\_TYPE\_BY\_MATERIAL\_CATEGORY\_ID =

new MethodInfo(

AuthorityEnum.USER\_LABORATORY, TypeAccessEnum.READ,

MATERIAL\_TYPE + MATERIAL\_TYPE\_BY\_MATERIAL\_CATEGORY\_ID );

public static final String STORAGE\_LOCATION = API + "/storage\_location";

public static final MethodInfo LAB\_ADMIN\_STORAGE\_LOCATION = new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY,TypeAccessEnum.READ\_WRITE,

STORAGE\_LOCATION);

public static final MethodInfo WORK\_ADMIN\_STORAGE\_LOCATION = new MethodInfo(

AuthorityEnum.ADMIN\_WORKSHOP,TypeAccessEnum.READ,

STORAGE\_LOCATION);

public static final MethodInfo LAB\_USER\_STORAGE\_LOCATION = new MethodInfo(

AuthorityEnum.USER\_LABORATORY,TypeAccessEnum.READ,

STORAGE\_LOCATION);

public static final String STORAGE\_LOCATION\_BY\_MATERIAL\_TYPE\_ID =

"/by\_material\_type\_id";

public static final MethodInfo LAB\_ADMIN\_STORAGE\_LOCATION\_BY\_MATERIAL\_TYPE\_ID =

new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY,TypeAccessEnum.READ,

STORAGE\_LOCATION +STORAGE\_LOCATION\_BY\_MATERIAL\_TYPE\_ID);

public static final MethodInfo WORK\_ADMIN\_STORAGE\_LOCATION\_BY\_MATERIAL\_TYPE\_ID =

new MethodInfo(

AuthorityEnum.ADMIN\_WORKSHOP,TypeAccessEnum.READ,

STORAGE\_LOCATION +STORAGE\_LOCATION\_BY\_MATERIAL\_TYPE\_ID);

public static final MethodInfo LAB\_USER\_STORAGE\_LOCATION\_BY\_MATERIAL\_TYPE\_ID =

new MethodInfo(

AuthorityEnum.USER\_LABORATORY,TypeAccessEnum.READ,

STORAGE\_LOCATION + STORAGE\_LOCATION\_BY\_MATERIAL\_TYPE\_ID);

public static final String NAME = API + "/name";

public static final MethodInfo LAB\_ADMIN\_NAME = new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY, TypeAccessEnum.READ\_WRITE,

NAME);

public static final MethodInfo WORK\_ADMIN\_NAME = new MethodInfo(

AuthorityEnum.ADMIN\_WORKSHOP,TypeAccessEnum.READ,NAME );

public static final MethodInfo LAB\_USER\_NAME = new MethodInfo(

AuthorityEnum.USER\_LABORATORY,TypeAccessEnum.READ,NAME);

public static final String NAME\_BY\_MATERIAL\_TYPE\_ID = "/by\_material\_type\_id";

public static final MethodInfo LAB\_ADMIN\_NAME\_BY\_MATERIAL\_TYPE\_ID =

new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY,TypeAccessEnum.READ,

NAME + NAME\_BY\_MATERIAL\_TYPE\_ID);

public static final MethodInfo WORK\_ADMIN\_NAME\_BY\_MATERIAL\_TYPE\_ID =

new MethodInfo(

AuthorityEnum.ADMIN\_WORKSHOP,TypeAccessEnum.READ,

NAME + NAME\_BY\_MATERIAL\_TYPE\_ID);

public static final MethodInfo LAB\_USER\_NAME\_BY\_MATERIAL\_TYPE\_ID =

new MethodInfo(

AuthorityEnum.USER\_LABORATORY,TypeAccessEnum.READ,

NAME + NAME\_BY\_MATERIAL\_TYPE\_ID);

public static final String NAME\_BY\_MANUFACTURER\_TYPE\_ID = "/by\_manufacturer\_id";

public static final MethodInfo LAB\_ADMIN\_NAME\_BY\_MANUFACTURER\_TYPE\_ID =

new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY,TypeAccessEnum.READ,

NAME + NAME\_BY\_MANUFACTURER\_TYPE\_ID );

public static final MethodInfo WORK\_ADMIN\_NAME\_BY\_MANUFACTURER\_TYPE\_ID =

new MethodInfo(

AuthorityEnum.ADMIN\_WORKSHOP,TypeAccessEnum.READ,

NAME + NAME\_BY\_MANUFACTURER\_TYPE\_ID);

public static final MethodInfo LAB\_USER\_NAME\_BY\_MANUFACTURER\_TYPE\_ID =

new MethodInfo(

AuthorityEnum.USER\_LABORATORY,TypeAccessEnum.READ,

NAME + NAME\_BY\_MANUFACTURER\_TYPE\_ID);

public static final String NAME\_WITH\_PARAMS = "/withparams";

public static final MethodInfo LAB\_ADMIN\_NAME\_WITH\_PARAMS = new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY,TypeAccessEnum.READ,

NAME + NAME\_WITH\_PARAMS );

public static final MethodInfo WORK\_ADMIN\_NAME\_WITH\_PARAMS = new MethodInfo(

AuthorityEnum.ADMIN\_WORKSHOP,TypeAccessEnum.READ,

NAME + NAME\_WITH\_PARAMS);

public static final MethodInfo LAB\_USER\_NAME\_WITH\_PARAMS = new MethodInfo(

AuthorityEnum.USER\_LABORATORY,TypeAccessEnum.READ,

NAME + NAME\_WITH\_PARAMS);

public static final String NORM = API + "/norm";

public static final MethodInfo LAB\_ADMIN\_NORM = new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY,TypeAccessEnum.READ\_WRITE,

NORM);

public static final MethodInfo WORK\_ADMIN\_NORM = new MethodInfo(

AuthorityEnum.ADMIN\_WORKSHOP,TypeAccessEnum.READ,

NORM);

public static final MethodInfo LAB\_USER\_NORM = new MethodInfo(

AuthorityEnum.USER\_LABORATORY,TypeAccessEnum.READ,

NORM);

public static final String NORM\_BY\_MATERIAL\_TYPE\_ID = "/by\_material\_type\_id";

public static final MethodInfo LAB\_ADMIN\_NORM\_BY\_MATERIAL\_TYPE\_ID =

new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY,TypeAccessEnum.READ,

NORM + NORM\_BY\_MATERIAL\_TYPE\_ID);

public static final MethodInfo WORK\_ADMIN\_NORM\_BY\_MATERIAL\_TYPE\_ID =

new MethodInfo(

AuthorityEnum.ADMIN\_WORKSHOP,TypeAccessEnum.READ,

NORM + NORM\_BY\_MATERIAL\_TYPE\_ID);

public static final MethodInfo LAB\_USER\_NORM\_BY\_MATERIAL\_TYPE\_ID =

new MethodInfo(

AuthorityEnum.USER\_LABORATORY, TypeAccessEnum.READ,

NORM + NORM\_BY\_MATERIAL\_TYPE\_ID);

public static final String MATERIAL\_TYPE\_PARAMETER = API +

"/material\_type\_parameter";

public static final MethodInfo LAB\_ADMIN\_MATERIAL\_TYPE\_PARAMETER =

new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY,TypeAccessEnum.READ\_WRITE,

MATERIAL\_TYPE\_PARAMETER);

public static final MethodInfo WORK\_ADMIN\_MATERIAL\_TYPE\_PARAMETER =

new MethodInfo(

AuthorityEnum.ADMIN\_WORKSHOP,

TypeAccessEnum.READ,MATERIAL\_TYPE\_PARAMETER);

public static final MethodInfo LAB\_USER\_MATERIAL\_TYPE\_PARAMETER = new MethodInfo(

AuthorityEnum.USER\_LABORATORY,TypeAccessEnum.READ,

MATERIAL\_TYPE\_PARAMETER);

public static final String MATERIAL\_TYPE\_PARAMETER\_BY\_MATERIAL\_TYPE\_ID =

"/by\_material\_type\_id";

public static final MethodInfo

LAB\_ADMIN\_MATERIAL\_TYPE\_PARAMETER\_BY\_MATERIAL\_TYPE\_ID = new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY,TypeAccessEnum.READ,

MATERIAL\_TYPE\_PARAMETER + MATERIAL\_TYPE\_PARAMETER\_BY\_MATERIAL\_TYPE\_ID);

public static final MethodInfo

WORK\_ADMIN\_MATERIAL\_TYPE\_PARAMETER\_BY\_MATERIAL\_TYPE\_ID = new MethodInfo(

AuthorityEnum.ADMIN\_WORKSHOP, TypeAccessEnum.READ,

MATERIAL\_TYPE\_PARAMETER + MATERIAL\_TYPE\_PARAMETER\_BY\_MATERIAL\_TYPE\_ID );

public static final MethodInfo

LAB\_USER\_MATERIAL\_TYPE\_PARAMETER\_BY\_MATERIAL\_TYPE\_ID = new MethodInfo(

AuthorityEnum.USER\_LABORATORY, TypeAccessEnum.READ,

MATERIAL\_TYPE\_PARAMETER + MATERIAL\_TYPE\_PARAMETER\_BY\_MATERIAL\_TYPE\_ID );

public static final String ALL = "/all";

public static final MethodInfo LAB\_USER\_CONTROLLED\_PARAMETERS\_ALL =

new MethodInfo(

AuthorityEnum.USER\_LABORATORY, TypeAccessEnum.READ,

CONTROLLED\_PARAMETERS + ALL);

public static final MethodInfo LAB\_USER\_MANUFACTURERS\_ALL = new MethodInfo(

AuthorityEnum.USER\_LABORATORY, TypeAccessEnum.READ, MANUFACTURERS + ALL );

public static final MethodInfo LAB\_USER\_MATERIAL\_CATEGORY\_ALL = new MethodInfo(

AuthorityEnum.USER\_LABORATORY, TypeAccessEnum.READ,

MATERIAL\_CATEGORY + ALL);

public static final MethodInfo LAB\_USER\_MATERIAL\_TYPE\_ALL = new MethodInfo(

AuthorityEnum.USER\_LABORATORY, TypeAccessEnum.READ,

MATERIAL\_TYPE + ALL );

public static final MethodInfo LAB\_USER\_STORAGE\_LOCATION\_ALL = new MethodInfo(

AuthorityEnum.USER\_LABORATORY, TypeAccessEnum.READ,

STORAGE\_LOCATION + ALL );

public static final MethodInfo LAB\_USER\_NAME\_ALL = new MethodInfo(

AuthorityEnum.USER\_LABORATORY, TypeAccessEnum.READ, NAME + ALL );

public static final MethodInfo LAB\_USER\_NORM\_ALL = new MethodInfo(

AuthorityEnum.USER\_LABORATORY, TypeAccessEnum.READ, NORM + ALL );

public static final MethodInfo LAB\_USER\_MATERIAL\_TYPE\_PARAMETER\_ALL =

new MethodInfo(

AuthorityEnum.USER\_LABORATORY, TypeAccessEnum.READ,

MATERIAL\_TYPE\_PARAMETER + ALL );

public static final MethodInfo LAB\_ADMIN\_CONTROLLED\_PARAMETERS\_ALL =

new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY, TypeAccessEnum.READ,

CONTROLLED\_PARAMETERS + ALL );

public static final MethodInfo LAB\_ADMIN\_MANUFACTURERS\_ALL = new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY, TypeAccessEnum.READ, MANUFACTURERS + ALL);

public static final MethodInfo LAB\_ADMIN\_MATERIAL\_CATEGORY\_ALL = new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY, TypeAccessEnum.READ,

MATERIAL\_CATEGORY + ALL );

public static final MethodInfo LAB\_ADMIN\_MATERIAL\_TYPE\_ALL = new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY, TypeAccessEnum.READ, MATERIAL\_TYPE + ALL);

public static final MethodInfo LAB\_ADMIN\_STORAGE\_LOCATION\_ALL = new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY, TypeAccessEnum.READ,

STORAGE\_LOCATION + ALL);

public static final MethodInfo LAB\_ADMIN\_NAME\_ALL = new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY, TypeAccessEnum.READ, NAME + ALL);

public static final MethodInfo LAB\_ADMIN\_NORM\_ALL = new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY, TypeAccessEnum.READ, NORM + ALL);

public static final MethodInfo LAB\_ADMIN\_MATERIAL\_TYPE\_PARAMETER\_ALL =

new MethodInfo(

AuthorityEnum.ADMIN\_LABORATORY, TypeAccessEnum.READ,

MATERIAL\_TYPE\_PARAMETER + ALL);

}

1. **SecurityInterceptor**

public class SecurityInterceptor extends HandlerInterceptorAdapter {

@Autowired

private EmployesClient employesClient;

TypeAccessEnum getTypeAccess(String method){

switch (method){

case "GET" :

return TypeAccessEnum.READ;

case "POST":case "PATCH":case "PUT":case "DELETE":

return TypeAccessEnum.READ\_WRITE;

}

throw new InvalidParametersException("getTypeAccess(String method)");

}

@Override

public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler) throws Exception {

String uri = request.getRequestURI();

if(uri.endsWith("/"))uri = uri.substring(0, uri.length()-1);

if(uri.equals("/error"))return false;

String method = request.getMethod();

TypeAccessEnum typeAccessEnum = getTypeAccess(method);

CheckTokenAnswerDTO checkTokenAnswerDTO = employesClient.checkToken(request.getHeader("ticket"));

String authority = checkTokenAnswerDTO.getAuthorities().get(0);

Field[] fields = Constants.class.getDeclaredFields();

for (Field field: fields) {

if(field.getType() == Constants.MethodInfo.class){

Constants.MethodInfo methodInfo =

(Constants.MethodInfo)field.get(null);

if (methodInfo.getMethoduri().equals(uri)){

if(typeAccessEnum.ordinal() >=

methodInfo.getTypeAccess().ordinal()){

if (methodInfo.getAutority().toString().equals(authority)){

return true;

}

}

}

}

}

throw new ForbiddenFeignException("403");

}

}

# ПРИЛOЖЕНИЕ В

**Сервис «Входной контроль»**

1. **final-product-dev.yaml – настройка справочного сервиса сырья**

spring:

profiles:

active: dev

datasource:

username: postgres

password: root

url: jdbc:postgresql://localhost:5432/pa-incoming-control-db

jpa:

database: POSTGRESQL

show-sql: true

liquibase:

changeLog: classpath:/db/changelog/db.changelog-master.xml

url: ${spring.datasource.url}

user: ${spring.datasource.username}

password: ${spring.datasource.password}

server:

servlet:

session:

cookie:

secure: false

http-only: false

port: 8082

1. **SQL-запросы на создание таблиц базы данных**

CREATE TABLE material (

id bigint PRIMARY KEY GENERATED BY DEFAULT AS IDENTITY,

names\_id bigint NOT NULL ,

manufacturer\_id bigint NOT NULL ,

material\_type\_id bigint NOT NULL ,

employee\_id uuid NOT NULL ,

storage\_location\_id bigint NOT NULL ,

condition bigint NOT NULL ,

batch\_manufacturer\_number varchar(40) NOT NULL ,

quality\_certificate\_number varchar(50) NOT NULL ,

manufacture\_date date NOT NULL ,

shelf\_life integer ,

receipt\_date date NOT NULL ,

count\_material decimal NOT NULL ,

car\_number varchar(15) ,

accompanying\_document\_number varchar(30) );

CREATE TABLE qualitative\_parameters (

id bigint PRIMARY KEY GENERATED BY DEFAULT AS IDENTITY,

material\_id bigint NOT NULL ,

parameter\_value varchar(30) NOT NULL ,

parameter\_id integer );

CREATE TABLE quantitative\_parameters (

id bigint PRIMARY KEY GENERATED BY DEFAULT AS IDENTITY,

material\_id bigint NOT NULL ,

parameter\_value decimal NOT NULL ,

parameter\_id bigint );

CREATE TABLE conclusion (

id bigint PRIMARY KEY GENERATED BY DEFAULT AS IDENTITY,

reason smallint NOT NULL ,

material\_id bigint UNIQUE );

ALTER TABLE conclusion

ADD CONSTRAINT fk\_conclusion\_material

FOREIGN KEY ( material\_id )

REFERENCES material( id ) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE qualitative\_parameters

ADD CONSTRAINT fk\_qualitative\_parameters\_material

FOREIGN KEY ( material\_id )

REFERENCES material( id ) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE quantitative\_parameters

ADD CONSTRAINT fk\_quantitative\_parameters

FOREIGN KEY ( material\_id )

REFERENCES material( id ) ON DELETE CASCADE ON UPDATE CASCADE;

1. **Сущность Сырье (Material)**

@Data

@Entity

@Table(name = "material")

public class Material {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name="accompanying\_document\_number")

private String accompanyingDocumentNumber;

@Column(name="car\_number")

private String carNumber;

@Column(name="count\_material")

private BigDecimal countMaterial;

@Column(name="receipt\_date")

private LocalDate receiptDate;

@Column(name = "shelf\_life")

private Integer shelfLife;

@Column(name="manufacture\_date")

private LocalDate manufactureDate;

@Column(name = "quality\_certificate\_number")

private String qualityCertificateNumber;

@Column(name = "batch\_manufacturer\_number")

private String batchManufacturerNumber;

@Enumerated(EnumType.ORDINAL)

@Column(name = "condition")

private ConditionEnum conditionEnum;

@Column(name="storage\_location\_id")

private Long storageLocationId;

@Column(name="employee\_id")

private UUID employeeId;

@Column(name="names\_id")

private Long namesId;

@Column(name = "manufacturer\_id")

private Long manufacturerId;

@Column(name = "material\_type\_id")

private Long materialTypeId;

@OneToOne(mappedBy = "material")

private Conclusion conclusion;

@Cascade(value = org.hibernate.annotations.CascadeType.ALL)

@OneToMany(mappedBy = "material")

private List<QualitativeParameter> qualitativeParameters;

@Cascade(value = org.hibernate.annotations.CascadeType.ALL)

@OneToMany(mappedBy = "material")

private List<QuantitativeParameter> quantitativeParameters;

}

1. **Сущность Заключение (Conclusion)**

@Data

@Entity

@Table(name = "conclusion")

public class Conclusion {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@OneToOne

@JoinColumn(name = "material\_id")

private Material material;

@Enumerated(EnumType.ORDINAL)

@Column(name = "reason")

private ReasonEnum reason;

}

1. **Сущность Качественный параметер (QualitativeParameter)**

@Data

@Entity

@Table(name = "qualitative\_parameters")

public class QualitativeParameter {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name="parameter\_value")

private String parameterValue;

@ManyToOne

@JoinColumn(name = "material\_id", referencedColumnName = "id")

private Material material;

@Column(name="parameter\_id")

private Long parameterId;

}

1. **Сущность Количественный парамер (QuantitativeParameter)**

@Data

@Entity

@Table(name = "quantitative\_parameters")

public class QuantitativeParameter {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name="parameter\_value")

private BigDecimal parameterValue;

@ManyToOne

@JoinColumn(name = "material\_id", referencedColumnName = "id")

private Material material;

@Column(name="parameter\_id")

private Long parameterId;

}

# ПРИЛOЖЕНИЕ Г

**Сервис «Цех»**

1. **final-product-dev.yaml – настройка справочного сервиса сырья**

spring:

profiles:

active: dev

datasource:

username: postgres

password: root

url: jdbc:postgresql://localhost:5432/pa-final-product-db

jpa:

database: POSTGRESQL

show-sql: true

liquibase:

changeLog: classpath:/db/changelog/db.changelog-master.xml

url: ${spring.datasource.url}

user: ${spring.datasource.username}

password: ${spring.datasource.password}

server:

servlet:

session:

cookie:

secure: false

http-only: false

port: 8085

1. **SQL-запросы на создание таблиц базы данных**

CREATE TABLE additional\_material (

id bigint PRIMARY KEY GENERATED BY DEFAULT AS IDENTITY,

incoming\_control\_id integer NOT NULL,

remains decimal NOT NULL );

CREATE TABLE final\_product (

id bigint PRIMARY KEY GENERATED BY DEFAULT AS IDENTITY,

product\_name\_id integer NOT NULL,

employee\_master\_id uuid NOT NULL,

employee\_tech\_id uuid NOT NULL,

product\_state integer NOT NULL,

change integer NOT NULL,

production\_date date NOT NULL,

product\_comment varchar(50),

all\_weigth decimal NOT NULL );

CREATE TABLE final\_product\_packaging (

id bigint PRIMARY KEY GENERATED BY DEFAULT AS IDENTITY,

final\_product\_id bigint NOT NULL,

packing\_id bigint NOT NULL,

box\_count integer NOT NULL );

CREATE TABLE main\_material (

id bigint NOT NULL PRIMARY KEY GENERATED BY DEFAULT AS IDENTITY,

incoming\_control\_id integer NOT NULL,

leftover decimal NOT NULL );

CREATE TABLE material\_tab (

id bigint PRIMARY KEY GENERATED BY DEFAULT AS IDENTITY,

final\_product\_id bigint NOT NULL,

note varchar(100) );

CREATE TABLE sanitary\_waste (

id bigint PRIMARY KEY GENERATED BY DEFAULT AS IDENTITY,

final\_product\_id bigint NOT NULL UNIQUE,

all\_weigth decimal NOT NULL );

CREATE TABLE specific\_packaging\_components (

id bigint PRIMARY KEY GENERATED BY DEFAULT AS IDENTITY,

additional\_materials\_id integer NOT NULL,

final\_product\_packing\_id integer NOT NULL,

expenditure decimal NOT NULL

CREATE TABLE tab\_ingredients (

id bigint PRIMARY KEY GENERATED BY DEFAULT AS IDENTITY,

material\_tab\_id bigint NOT NULL,

expenditure decimal NOT NULL );

CREATE TABLE material\_ingredients\_tabs (

tab\_ingredients\_id bigint NOT NULL,

main\_material\_id integer NOT NULL,

PRIMARY KEY ( tab\_ingredients\_id, main\_material\_id ) );

ALTER TABLE final\_product\_packaging ADD CONSTRAINT fk\_final\_product\_packaging

FOREIGN KEY ( final\_product\_id )

REFERENCES final\_product( id ) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE material\_ingredients\_tabs ADD CONSTRAINT fk\_material\_ingredients\_tabs

FOREIGN KEY ( main\_material\_id )

REFERENCES main\_material( id ) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE material\_ingredients\_tabs

ADD CONSTRAINT fk\_material\_ingredients\_tabs\_material\_tab

FOREIGN KEY ( tab\_ingredients\_id )

REFERENCES tab\_ingredients( id ) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE material\_tab ADD CONSTRAINT fk\_material\_tab\_final\_product

FOREIGN KEY ( final\_product\_id )

REFERENCES final\_product( id ) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE sanitary\_waste ADD CONSTRAINT fk\_sanitary\_marriage

FOREIGN KEY ( final\_product\_id )

REFERENCES final\_product( id ) ON DELETE SET NULL ON UPDATE CASCADE;

ALTER TABLE specific\_packaging\_components

ADD CONSTRAINT fk\_specific\_packaging\_components\_matreials

FOREIGN KEY ( additional\_materials\_id )

REFERENCES additional\_material( id ) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE specific\_packaging\_components

ADD CONSTRAINT fk\_specific\_packaging\_components\_packing

FOREIGN KEY ( final\_product\_packing\_id )

REFERENCES final\_product\_packaging( id ) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE tab\_ingredients ADD CONSTRAINT fk\_tab\_ingredients

FOREIGN KEY ( material\_tab\_id )

REFERENCES material\_tab( id ) ON DELETE CASCADE ON UPDATE CASCADE;

1. **Сущность Вспомогательные материалы (AdditionalMaterial)**

@Data

@Entity

@Table(name = "additional\_material")

public class AdditionalMaterial {

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Id private Long id;

@Column(name = "incoming\_control\_id")

private Long incomingControlId;

@Column(name = "remains")

private Double remains;

}

1. **Сущность Готовый продукт (FinalProduct)**

@Data

@Entity

@Table(name = "final\_product")

public class FinalProduct {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name = "product\_state")

private ProductStateEnum state;

@Column(name = "change")

private Integer change;

@Column(name = "production\_date")

private LocalDate productionDate;

@Column(name = "product\_comment")

private String productComment;

@Column(name = "all\_weigth")

private Double allWeight;

@Column(name = "employee\_master\_id")

private UUID employeeMasterId;

@Column(name = "employee\_tech\_id")

private UUID employeeTechId;

@Column(name = "product\_name\_id")

private Long productNameId;

@OneToMany(mappedBy = "finalProduct")

private List<FinalProductPackaging> packagings;

@OneToOne(mappedBy = "finalProduct")

private SanitaryWaste sanitaryWaste;

@OneToMany(mappedBy = "finalProduct")

private List<MaterialTab> materialTabs;

}

1. **Сущность Упаковка готового продукта (FinalProductPackaging)**

@Data

@Entity

@Table(name = "final\_product\_packaging")

public class FinalProductPackaging {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name = "packing\_id")

private Long packingId;

@Column(name = "box\_count")

private Integer boxCount;

@ManyToOne

@JoinColumn(name = "final\_product\_id", referencedColumnName = "id")

private FinalProduct finalProduct;

}

1. **Сущность Основные материалы (MainMaterial)**

@Data

@Entity

@Table(name = "main\_material")

public class MainMaterial {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name = "incoming\_control\_id")

private Long incomingControlId;

@Column(name = "leftover")

private Double remains;

@ManyToMany(mappedBy = "mainMaterials")

private List<TabIngredient> tabIngredients;

}

1. **Сущность Основные ингредиенты закладки (MaterialIngredientTab)**

@Data

@Entity

@Table(name = "material\_ingredients\_tabs")

public class MaterialIngredientTab {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@ManyToOne

@JoinColumn(name = "tab\_ingredients\_id", referencedColumnName = "id")

private TabIngredient tabIngredients;

@ManyToOne

@JoinColumn(name = "main\_material\_id", referencedColumnName = "id")

private MainMaterial mainMaterial;

}

1. **Сущность Закладка (MaterialTab)**

@Data

@Entity

@Table(name = "material\_tab")

public class MaterialTab {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name = "note")

private String note;

@ManyToOne

@JoinColumn(name = "final\_product\_id", referencedColumnName = "id")

private FinalProduct finalProduct;

@Cascade(org.hibernate.annotations.CascadeType.ALL)

@OneToMany(mappedBy = "materialTab")

private List<TabIngredient> tabIngredients;

}

1. **Сущность Санитарный брак (SanitaryWaste)**

@Data

@Entity

@Table(name = "sanitary\_waste")

public class SanitaryWaste {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name = "all\_weigth")

private Double weigth;

@OneToOne

@JoinColumn(name = "final\_product\_id", referencedColumnName = "id")

private FinalProduct finalProduct;

}

1. **Сущность Состав конкретной упаковки (SpecificPackagingComponent)**

@Data

@Entity

@Table(name = "specific\_packaging\_components")

public class SpecificPackagingComponent {

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Id private Long id;

@Column(name = "expenditure")

private Double expenditure;

@ManyToOne

@JoinColumn(name = "additional\_materials\_id", referencedColumnName = "id")

private AdditionalMaterial additionalMaterial;

@ManyToOne

@JoinColumn(name = "final\_product\_packing\_id", referencedColumnName = "id")

private FinalProductPackaging finalProductPackaging;

}

1. **Сущность Ингредиенты закладки (TabIngredient)**

@Data

@Entity

@Table(name = "tab\_ingredients")

public class TabIngredient {

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Id private Long id;

@Column(name = "expenditure")

private Double expenditure;

@ManyToOne

@JoinColumn(name = "material\_tab\_id", referencedColumnName = "id")

private MaterialTab materialTab;

@ManyToMany

@Cascade(org.hibernate.annotations.CascadeType.SAVE\_UPDATE)

@JoinTable(name = "material\_ingredients\_tabs",

joinColumns = @JoinColumn(name ="tab\_ingredients\_id"),

inverseJoinColumns = @JoinColumn(name ="main\_material\_id"))

private List<MainMaterial> mainMaterials;

}

1. **CrudService**

public interface CrudService<E, DR, DC, I> {

E findById(I id);

I save(E entity);

void update(E entity);

List<E> readAll();

DR findByIdDTO(I id);

List<DR> readAllDTO();

I saveDTO(DC dto);

void updateDTO(DC dto);

Boolean isExist(I id);

void delete(I id);

}

1. **AbstractService**

@RequiredArgsConstructor

public abstract class AbstractService <E, I, R extends JpaRepository<E, I>, DR, DC, M extends GenericMapper<DR, DC, E>>

implements CrudService<E, DR, DC, I> {

private final R repository;

private final M mapper;

@Override

public Boolean isExist(I id) {

if(id == null) throw new InvalidParametersException("id");

return repository.existsById(id);

}

@Override

public void delete(I id) {

if (id == null)throw new InvalidParametersException("id");

E entity = findById(id);

repository.delete(entity);

}

@Override

public E findById(I id) {

if(id == null)throw new InvalidParametersException("id");

return repository.findById(id).orElseThrow(

() -> new ResourceNotFoundException("id"));

}

@Override

public DR findByIdDTO(I id) {

if(id == null)throw new InvalidParametersException("id");

return mapper.toReadDto(findById(id));

}

@Override

public List<E> readAll() { return repository.findAll(); }

@Override

public List<DR> readAllDTO() { return mapper.toReadDtos(readAll()); }

}

1. **GenericMapper**

public interface GenericMapper<DR, DC, E> {

DR toReadDto(E entity);

List<DR> toReadDtos(List<E> entities);

E toReadEntity(DR dto);

List<E> toReadEntities(List<DR> dtos);

DC toCreateDto(E entity);

List<DC> toCreateDtos(List<E> entities);

E toCreateEntity(DC dto);

List<E> toCreateEntities(List<DC> dtos);

@BeanMapping(nullValuePropertyMappingStrategy = IGNORE)

void updateFromCreateDto(DC d, @MappingTarget E e);

}

1. **ProductDirectoryClient**

@FeignClient(name = "product-directory", configuration = FeignConfiguration.class)

public interface ProductDirectoryClient {

@GetMapping(value = "/api/v1/line")

LineDTO getLine(@RequestHeader("ticket") String ticket, @RequestParam Long id);

@GetMapping(value = "/api/v1/product\_name")

ProductNameDTO getProductName(@RequestHeader("ticket") String ticket, @RequestParam Long id);

@GetMapping(value = "/api/v1/packing/by\_product\_name\_id")

List<PackagingDTO> getPackagingByProductNameId(@RequestHeader("ticket") String ticket, @RequestParam Long id);

@GetMapping(value = "/api/v1/packing")

PackagingDTO getPackaging(@RequestHeader("ticket") String ticket, @RequestParam Long id);

}

1. **Клиентские объекты передачи данных**

@Data

public class LineDTO {

private Long id;

private String name;

}

@Data

public class PackagingDTO {

private Long id;

private String name;

private String productCode;

private Double packWeight;

private Integer countPacksInBox;

}

@Data

public class ProductNameDTO {

private Long id;

private LineDTO line;

private String name;

private String description;

}

@Data

public class UserDTO {

private UUID id;

private String firstName;

private String secondName;

private String patronymic;

}

1. **FinalProductController**

@RestController @RequestMapping(Constants.PRODUCT)

@RequiredArgsConstructor

public class FinalProductController {

private final FinalProductService finalProductService;

private final EmployeesClient employeesClient;

private final ProductDirectoryClient productDirectoryClient;

@PostMapping

public Long create(@RequestBody FinalProductCreateDto dto) {

return finalProductService.saveDTO(dto); }

@GetMapping(Constants.ALL)

public List<FullFinalProductReadDto> readAllFullDtos(@RequestHeader("ticket")

String ticket){

List<FullFinalProductReadDto> fullFinalProductReadDtos =

finalProductService.findAllFullReadDto();

fullFinalProductReadDtos.forEach(f -> fillDTO(f, ticket));

return fullFinalProductReadDtos;

}

@GetMapping(Constants.TECH)

public List<FullFinalProductReadDto> readAllFullDtos(@RequestHeader("ticket")

String ticket, UUID techID){

List<FullFinalProductReadDto> fullFinalProductReadDtos =

finalProductService.findAllFullReadDtoByTech();

fullFinalProductReadDtos.forEach(f -> fillDTO(f, ticket));

return fullFinalProductReadDtos;

}

@GetMapping(Constants.MASTER)

public List<FullFinalProductReadDto> readAllFullDtos(@RequestHeader("ticket")

String ticket, UUID masterID){

List<FullFinalProductReadDto> fullFinalProductReadDtos =

finalProductService.findAllFullReadDtoByMaster();

fullFinalProductReadDtos.forEach(f -> fillDTO(f, ticket));

return fullFinalProductReadDtos;

}

private FullFinalProductReadDto fillDTO(

FullFinalProductReadDto fullFinalProductReadDto, String ticket){

fullFinalProductReadDto.setMaster(

employeesClient.findUserById(

fullFinalProductReadDto.getMaster().getId(), "Bearer " +

ticket

)

);

fullFinalProductReadDto.setTech(

employeesClient.findUserById(

fullFinalProductReadDto.getTech().getId(), "Bearer " + ticket

)

);

fullFinalProductReadDto.setProductName(

productDirectoryClient.getProductName(

ticket, fullFinalProductReadDto.getProductName().getId()

)

);

return fullFinalProductReadDto;

}

}

# ПРИЛOЖЕНИЕ Д

**Серис «Справочник продукта»**

1. **product-directory-dev.yaml – настройка справочного сервиса продукции**

spring:

profiles:

active: dev

datasource:

username: postgres

password: root

url: jdbc:postgresql://localhost:5432/pa-product-directory-db

jpa:

database: POSTGRESQL

show-sql: true

liquibase:

changeLog: classpath:/db/changelog/db.changelog-master.xml

url: ${spring.datasource.url}

user: ${spring.datasource.username}

password: ${spring.datasource.password}

# drop-first: true

server:

servlet:

session:

cookie:

secure: false

http-only: false

port: 8084

# ПРИЛOЖЕНИЕ Е

**Основные запросы к системе**

1. **Результат запроса на получение данных о сырье**

[

{

"id": 3,

"accompanyingDocumentNumber": "555",

"carNumber": "к028му127",

"countMaterial": 555.5,

"receiptDate": "2020-06-10",

"shelfLife": 12,

"manufactureDate": "2020-05-28",

"qualityCertificateNumber": "324-01б/к",

"batchManufacturerNumber": "67",

"conditionEnum": "ACCEPTED",

"name": {

"id": 8,

"name": "Вишня садовая",

"nameOneC": "В-15"

},

"manufacturer": {

"id": 3,

"name": "ОАО АРОМАВКУС"

},

"materialType": {

"id": 5,

"name": "Ароматизаторы",

"measure": "кг",

"materialCategory": {

"id": 1,

"name": "Основное сырье"

}

},

"storageLocation": {

"id": 1,

"name": "Склад цеха №2"

},

"employee": {

"id": "4985da53-56f8-4e25-a030-2fde8b3e49f7",

"firstName": "Оксана",

"secondName": "Браткова",

"patronymic": "Константиновна"

},

"conclusion": null,

"qualitativeParameters": [

{

"id": 3,

"parameterValue": "Свойственный",

"parameter": {

"id": 6,

"controlledParameter": {

"id": 3, "name": "Вкус",

"type": "QUALITATIVE"

},

"sourceEnum": "CERTIFICATE",

"norm": null

}

}

],

"quantitativeParameters": [

{

"id": 3,

"parameterValue": 0.6,

"parameter": {

"id": 7,

"controlledParameter": {

"id": 4, "name": "Влажность",

"type": "QUANTITATIVE"

},

"sourceEnum": "CERTIFICATE",

"norm": null

}

}

]

}

]