ООО «Наука»
194223, г. Санкт-Петербург,
ул. Курчатова, д. 6, корп. 4, пом. 2Н лит. А

Тел: +7(812)346-61-49
Факс: +7(812)346-61-45
office@ntik.ru
www.ntik.ru

Система для повышения энергоэффективности предприятия «NAUKA.Energy»

Руководство администратора

2025 г.

Оглавление

1 Общие сведения

Система для повышения энергоэффективности предприятия «NAUKA.Energy» – автоматизированная система, предназначенная для формирования оптимальных норм потребления топливно-энергетических ресурсов (ТЭР) для различных режимов работы технологических объектов и выявления основных факторов, влияющих на их потребление.

Система передается заказчику настроенной в docker-контейнерах, разворачивается заказчиком самостоятельно.

При работе пользователя с системой предполагается наличие начальных знаний и навыков работы с персональным компьютером.

2 Описание сервера

2.1 Операционная система:

- ALT Server 10.XX
- любая другая операционная система с установленным ПО docker и docker-compose

2.2 Системные компоненты сервера

Kafka

- kafka 3.7.0

Postgresql

- Версия СУБД: psql (PostgreSQL) 14

Docker

- docker-engine-27.1.1-alt1.x86_64 или выше
- docker-compose-v2-2.29.7-alt2.x86_64 или выше

3 Архитектура ПО

3.1 Системная архитектура продукта

Системная архитектура продукта представлена на рисунке.

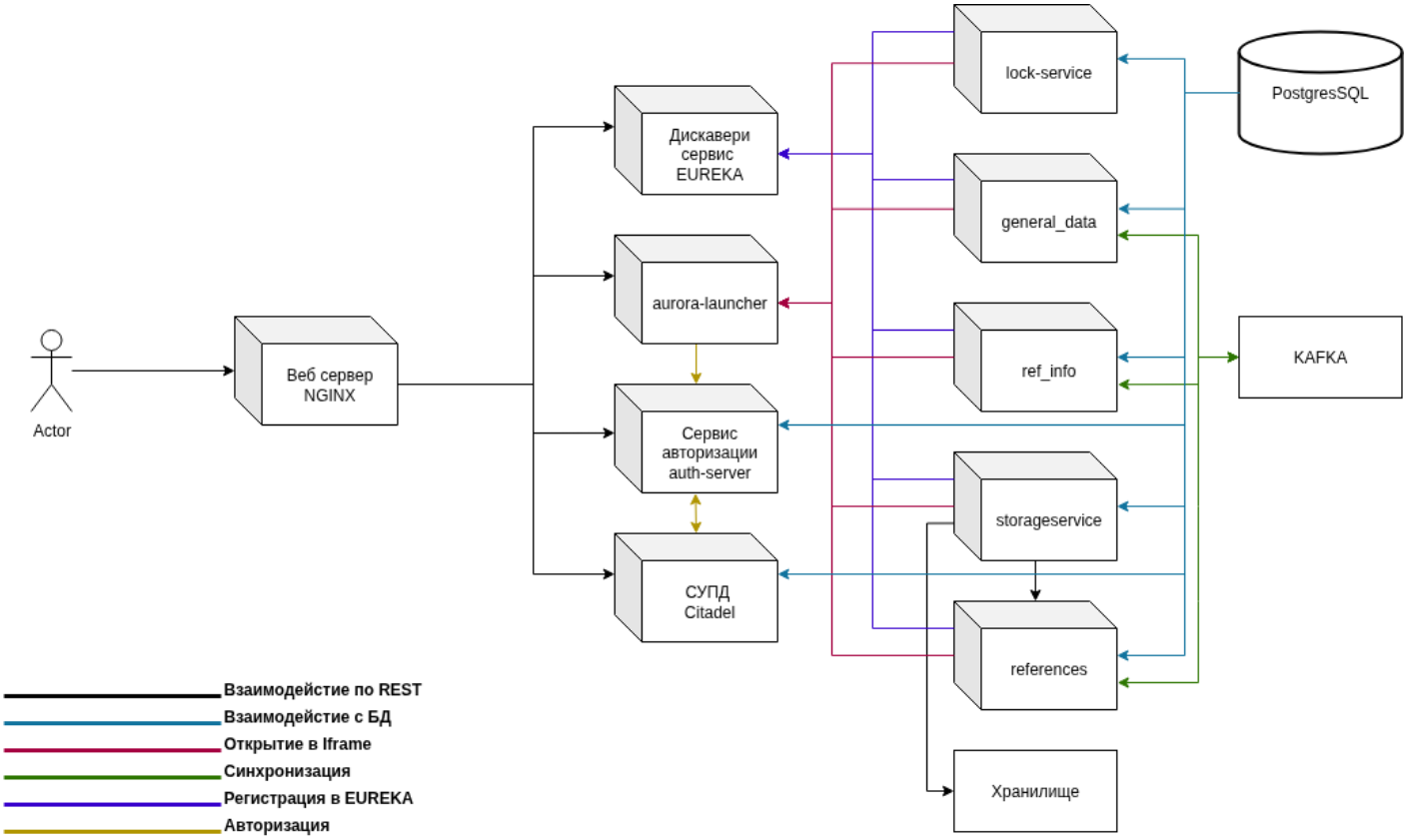

Рисунок - Системная архитектура продукта

3.2 Сторонние компоненты

Alt-linux - семейство дистрибутивов Linux, являющихся отдельной ветвью развития русскоязычного Linux.

PostgreSQL - свободная объектно-реляционная система управления базами данных. Распространяется согласно разрешительной лицензии на свободное ПО. Используется для управления\хранения\обмена данными с ПО. В данном решении используется одна инсталляция PostgreSQL, в рамках которой используются 2 отдельные БД.

NGINX - Веб-сервер с открытым исходным кодом, разработанный на языке программирования C, которое позволяет создавать и управлять HTTP-сервером и прокси-сервером. Используется для проксирования запросов от клиента-веб-браузера к компонентам ПО, а также раздачи статических файлов.

Kafka – распределённый программный брокер сообщений

Eureka – (Eureka Server for Spring), реестр служб, позволяющий микрослужбам регистрировать себя и обнаруживать другие службы

Docker – программное обеспечение для автоматизации развёртывания и управления приложениями в средах с поддержкой контейнеризации

Apache Tomcat - комплект серверных программ от Apache Software Foundation, предназначенный для тестирования, отладки и исполнения веб-приложений на основе Java.

Java - строго типизированный объектно-ориентированный язык программирования общего назначения, разработанный компанией Sun Microsystems (в последующем приобретённой компанией Oracle).
Java Development Kit (сокращенно JDK) — OpenJDK, бесплатно распространяемый компанией Oracle Corporation (ранее Sun Microsystems) комплект разработчика приложений на языке Java, включающий в себя компилятор Java, стандартные библиотеки классов Java, примеры, документацию, различные утилиты и исполнительную систему Java (JRE). Используется для исполнения компонентов ПО в виде .jar файлов.

3.3 Собственные компоненты

Energy – приложение (ПО, продукт) для формирования оптимальных норм потребления топливно-энергетических ресурсов (ТЭР) для различных режимов работы технологических объектов и выявления основных факторов, влияющих на их потребление.

Компоненты платформы собственной разработки:

- **Lock-service** – сервис блокировок для обеспечения невозможности одновременного редактирования данных разными пользователями
- **General-data** – сервис общесистемных классификаторов бизнес объектов (пользователи, основные средства и т.п.)
- **Ref-info** – сервис публичных классификаторов (языки, единицы измерения и т.п.)
- **Storageservice** – сервис, обеспечивающий взаимодействие с файловыми хранилищами
- **Auth-server** – сервер авторизации
- **СУПД Citadel** – система управления правами доступа

4 Порядок первичной установки системы

Система устанавливается на операционную систему Alt-linux двумя подготовленными разработчиком пакетами:

- references-X.X.X.tar, где XXX – номер текущей версии ПО;
- nauka-java17-X.X.X.x86_64.rpm, где XXX – номер текущей версии ПО.

4.1 Предварительные требования

1. На целевом или любом другом сервере должна быть установлена СУБД Postgresql-14, настроен доступ для подключения, созданы базы и пользователи:

   Имя БД и имя пользователя в СУБД могут быть одинаковыми:
   1) citadel
   2) auth_server
   3) launcher
   4) lock_service
   5) general_data
   6) ref_info
   7) storageservice

2. На целевом или другом сервере должна быть установлена Kafka kafka 3.7.0 или выше, настроен доступ к ней.

3. На целевом сервере должно быть установлено ПО docker и docker-compose.

4.2 Установка ПО

1. Установите пакет nauka-java17-X.X.X.x86_64.rpm , после его установки на сервере создадутся каталоги /opt/nauka и пользователь nauka
2. Распакуйте архив на сервер references-X.X.X.tar в каталог /opt/nauka

На сервере появятся следующие каталоги

С файлами лицензий для приложений

3. Настройте NGINX:

   a. Скопируйте ваши tls сертификаты для работы протокола https в директорию ./nginx/ssl с именами ssl.crt и ssl.key

b. При необходимости, впишите адрес хоста в конфигурацию веб-сервера nginx в файле ./nginx/r.conf в строке server_name _;, заменив "_" на требуемое dns имя.

4. Настройте переменные среды (отредактируйте файл .env где укажите параметры, соответствующие вашему окружению):

a. **KAFKA_SERVER**=172.17.0.1:9092 # Укажите адрес сервера kafka. Оставьте по умолчанию, если кафка расположена на хостовой машине
b. **DB_SERVER**=172.17.0.1 # Укажите адрес СУБД Postgres. Оставьте по умолчанию, если СУБД расположена на хостовой машине.
c. **LAUNCHER_SERVER**="srv-fenix-testhl01.nauka.local" # DNS имя по которому будет доступно приложение, на которые выдан tls сертификат
d. **EUREKA_SPRING_SECURITY_USER_NAME**=admin # имя пользователя для авторизации в сервис eureka
e. **EUREKA_SPRING_SECURITY_USER_PASSWORD**=My_Pass_123! # Пароль для авторизации в сервис eureka
f. **CITADEL_DB_ADDRESS**=${DB_SERVER}/citadel # адрес и имя БД для цитадели
g. **CITADEL_DB_USER**=citadel # пользователь дб для цитадели
h. **CITADEL_DB_PASS**='citadel_subd_password' # пароль для БД цитадели

5. Типовые переменные для сервисов
a. **<APP>_TAG** # тег docker image приложения.
Пример: NGINX_TAG=1.28
b. **<APP>_SPRING_DATASOURCE_URL** # url подключения к бд. Укажите имя БД, остальное без изменений.
Пример:
_STORAGE_SPRING_DATASOURCE_URL="jdbc:postgresql://${DB_SERVER}/storageservice"
c. **<APP>_SPRING_DATASOURCE_USERNAME** # Укажите имя пользователя БД.
Пример: STORAGE_SPRING_DATASOURCE_USERNAME=storage
d. **<APP>_SPRING_DATASOURCE_PASSWORD** # Укажите пароль пользователя БД.
Пример: STORAGE_SPRING_DATASOURCE_PASSWORD='My_password_123!'
e. **<APP>_APP_STARTUP_OPTIONS** # Дополнительные параметры запуска приложения, оставьте без изменений. Или если передан файл с дополнительными конфигурациями подключаем тут
Пример:
-Dspring.import.config=path_to_file

6. Кроме архива, у вас должны быть файлы докер-образов приложений в формате tar. Загрузите все образы в локальный docker-registry (Или ваш корпоративный, если планируете загружать образы на хост с него).

7. Произведите запуск стека приложений:

```
docker compose up -d
```

8. Через пару минут система будет готова. Попробуйте открыть в браузере:

https://<ваш_url> - лаунчер, с прикладными приложениями, включая феникс

https://<ваш_url>/citadel – Система Управления Правами Доступа (логин и пароль пользователя admin описаны в блоке настройки приложения.)

https://<ваш_url>/eureka - discovery сервис для внутреннего взаимодействия компонентов системы.

5 Управление приложением

Запуск и остановка выполняется средствами Docker.

Пример:
docker compose up -d
docker compose down

Также можно управлять конкретным сервисом.

Пример:
docker compose up eureka -d