



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

Руководство администратора ПО «NAUKA. Phoenix»

Оглавление

1 Общие сведения	3
2 Описание сервера.....	3
2.1 Операционная система:.....	3
2.2 Системные компоненты сервера.....	3
3 Архитектура ПО.....	4
3.1 Системная архитектура продукта	4
3.2 Сторонние компоненты	4
3.3 Собственные компоненты.....	5
4 Порядок первичной установки системы	5
4.1 Предварительные требования.....	5
4.2 Установка ПО.....	6
5 Управление приложением.....	9

1 Общие сведения

ПО «NAUKA.Phoenix» предназначено для для управления проектной деятельностью организации. 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 1.

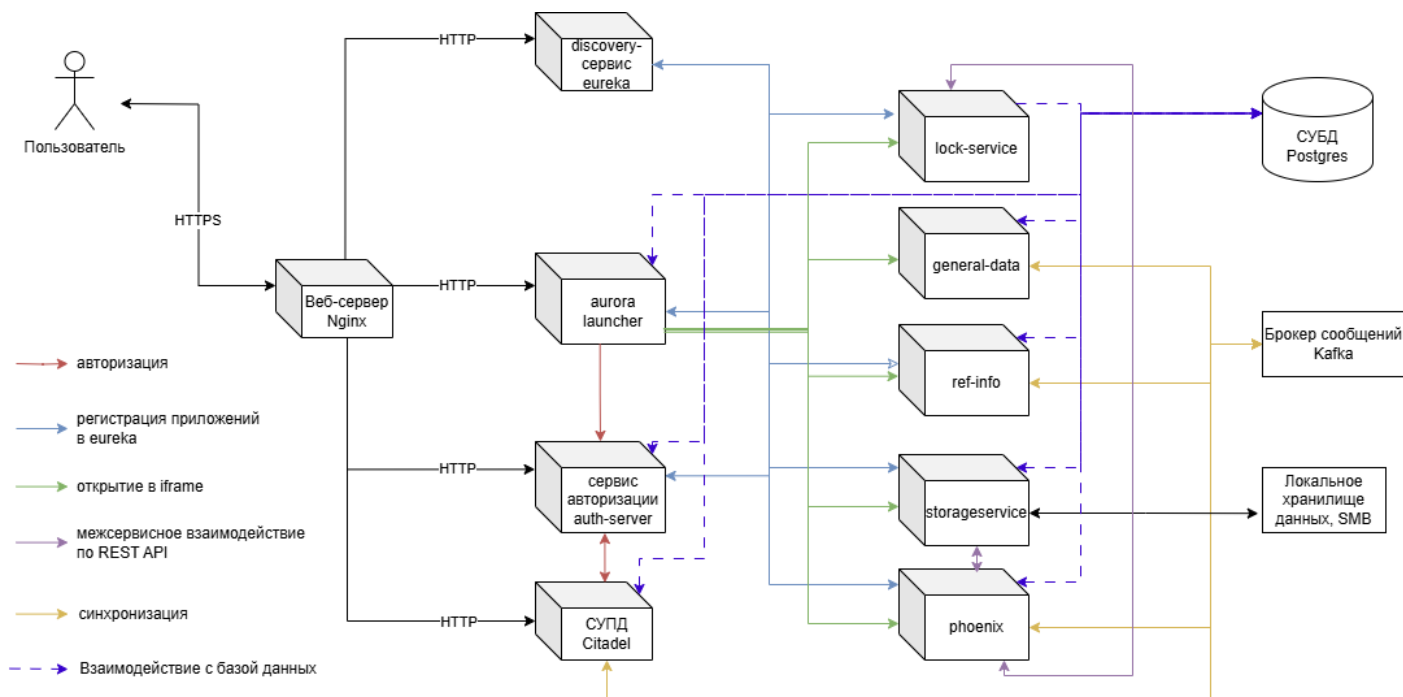


Рисунок 0.1 - Системная архитектура продукта

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

- **Phoenix** – приложение (ПО, продукт) для управления проектами, функционирующее на базе платформы собственной разработки
- 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

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) mother_queen
- 6) general_data
- 7) ref_info

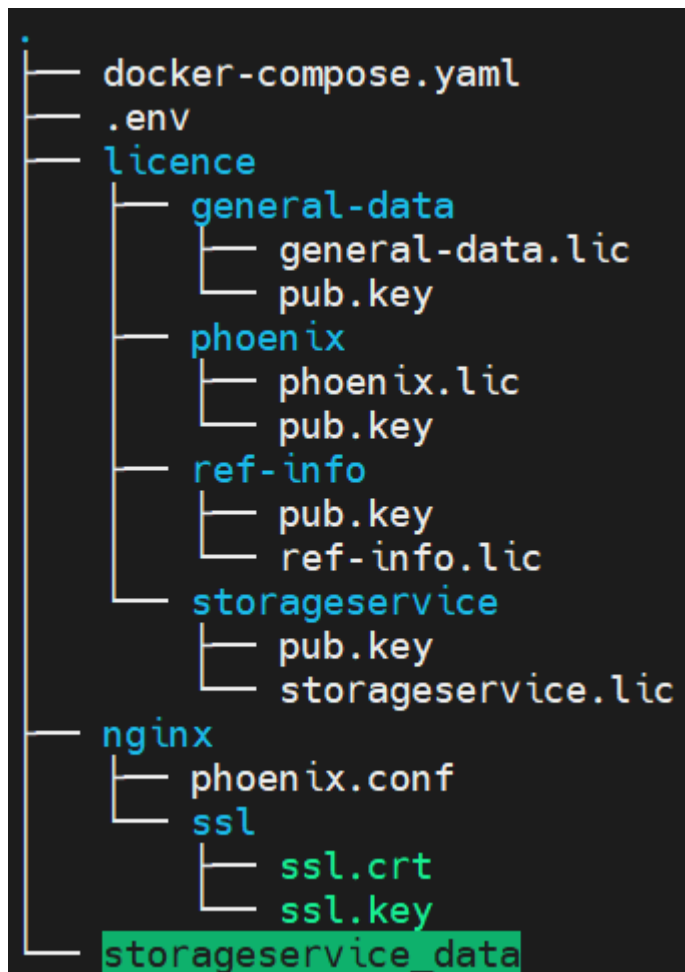
- 8) storageservice
 - 9) phoenix
2. На целевом или другом сервере должна быть установлена Kafka kafka 3.7.0 или выше, настроен доступ к ней.
 3. На целевом сервере должно быть установлено ПО docker и docker-compose.

4.2 Установка ПО

1. Распаковать архив phoenix.tar в любую директорию на сервере. И перейти в каталог.

Пример: **`tar -xvf phoenix.tar -C /opt/ && cd /opt/phoenix`**

Структура файлов и каталогов:



2. Скопируйте ваши tls сертификаты для работы протокола https в директорию `./nginx/ssl` с именами `ssl.crt` и `ssl.key`
3. При необходимости, впишите адрес хоста в конфигурацию веб-сервера nginx в файле `./nginx/phoenix.conf` в строке `server_name _;`, заменив `"_"` на требуемое dns имя.
4. Отредактируйте файл `.env` где укажите параметры, соответствующие вашему окружению:

KAFKA_SERVER=172.17.0.1:9092 # Укажите адрес сервера kafka. Оставьте по умолчанию, если kafka расположена на хостовой машине

DB_SERVER=172.17.0.1 # Укажите адрес СУБД Postgres. Оставьте по умолчанию, если СУБД расположена на хостовой машине.

LAUNCHER_SERVER="srv-fenix-testhl01.nauka.local" # DNS имя по которому будет доступно приложение, на которые выдан tls сертификат

EUREKA_SPRING_SECURITY_USER_NAME=admin # имя пользователя для авторизации в сервис eureka

EUREKA_SPRING_SECURITY_USER_PASSWORD=My_Pass_123! # Пароль для авторизации в сервис eureka

CITADEL_DB_ADDRESS=\${DB_SERVER}/citadel # адрес и имя БД для цитадели

CITADEL_DB_USER=citadel # пользователь db для цитадели

CITADEL_DB_PASS='citadel_subd_password' # пароль для БД цитадели

Далее у каждого сервиса однотипные переменные:

- **<APP>_TAG** # тег docker image приложения.

Пример: `NGINX_TAG=1.28`

- **<APP>_SPRING_DATASOURCE_URL** # url подключения к бд. Укажите имя БД, остальное без изменений.

Пример:

```
TORAGE_SPRING_DATASOURCE_URL="jdbc:postgresql://${DB_SERVER}/storageservice"
```

- **<APP>_SPRING_DATASOURCE_USERNAME** # Укажите имя пользователя БД

Пример: STORAGE_SPRING_DATASOURCE_USERNAME=storage

- **<APP>_SPRING_DATASOURCE_PASSWORD** # Укажите пароль пользователя БД

Пример: STORAGE_SPRING_DATASOURCE_PASSWORD='My_password_123!'

- **<APP>_APP_STARTUP_OPTIONS** # Дополнительные параметры запуска приложения, оставьте без изменений.

5. 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).

Пример:

```
cd /path/with/images && for image in $(ls *.tar); do docker load -i $image; done
```

cd -

6. Если файлы лицензий были переданы отдельным архивом, то переместите их в соответствующие пути:

```
./licence/<APP_NAME>/<app-name.>.lic
```

```
./licence/<APP_NAME>/pub.key
```

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
```