

Инструкция по установке экземпляра программного обеспечения

Требования

Для развертывания в боевом окружении могут быть подключены разные реализации OIDC/OAuth2 сервиса. Смотрите документацию [Micronaut](<https://micronaut-projects.github.io/micronaut-security/latest/guide/#oauth>)

Так же рекомендуется использовать TLS соединения, для чего можно использовать сторонние свободно распространяемые сервисы такие как Nginx, Caddy и т.п.

В качестве ОС может использоваться любая, которая поддерживается выбранной JVM версии не ниже 17. Минимальные требования для одного инстанса FiSys: 1000 MHz CPU, 512 MB RAM,

Демонстрация

В качестве демонстрации установки экземпляра программного обеспечения поставляется архив с минимально необходимым набором инструментов, сервисов, настроечных файлов. Сервисы предоставлены в виде Docker образов. При этом используются следующие компоненты:

- Assist FiSys - разработка компании Assist. Docker образ создается на базе [OpenJDK 17 с Eclipse Temurin](#).
- Oracle MYSQL (Community Edition) - бесплатная версия Базы данных с открытым кодом. Используется под GPL лицензией
- В качестве сервиса Авторизации и аутентификации в данном пакете используется Ory Hydra + небольшой инструмент `hydra-user-and-consent-provider-node`. Ory Hydra - это API-only, "headless," OAuth 2.0 и OpenID Connect provider, который может использовать другие системы интерфейсов и менеджмента пользователей, таких как Ory Kratos, Firebase, LDAP, SAML, и т.п. Распространяется по Apache License 2.0. [Hydra-user-and-consent-provider-node](#) - это простой пример реализации интерфейса для Логина пользователей и "Consent flow" для OAuth2 сервиса (Ory Hydra) написанный на NodeJS, он дополнен небольшим кодом для реализации DEMO учетных записей. Распространяется по Apache License 2.0

Минимальные требования к машине для установки демонстрационной версии FiSys: 2000 MHz CPU, 1024 MB RAM (лучше 2048 MB), 1 GB HDD.

- 1 Для запуска необходимо установить последние версии [Docker](#) и [Docker Compose](#).
- 2 Убедиться в наличии доступа к [dockerhub](#) репозиторию с образами дополнительных сервисов.
- 3 Распаковать полученный архив в выделенную директорию
- 4 Сервисы разворачиваемые при запуске демонстрационного пакета по умолчанию будут использовать следующие порты: TCP:8086, TCP:4444, TCP:3000. Перед запуском необходимо убедиться что они не заняты на текущей машине. Иначе можно изменить их в `docker-compose.yml` файле.
- 5 Прописать хост `hydra` как `127.0.0.1` для доступа с локальной машины.
- 6 Для разворачивания и запусков сервисов необходимо из директории с распакованными файлами запустить скрипт (пример в `build_and_run.sh`): `docker-compose -f docker-compose.yml -p assist-fisys up -d`
- 7 После успешного разворачивания и запуска будут созданы директории `mysql` и `logs`. В `logs/fisys/fisys.log` появится примерно такая запись: `[main] INFO io.micronaut.runtime.Micronaut []- Startup completed in 10574ms. Server Running: http://fisys:8080`
- 8 По умолчанию сервис поднимется и будет доступен по адресу <http://localhost:8086> для работы с Личным кабинетом
- 9 Описание протокола REST API для взаимодействия с системой представлено в виде OpenAPI документации и будет доступно по адресу <http://localhost:8086/swagger-ui>

В системе также по умолчанию будут созданы:

- компания TEST DEMO с идентификатором 555555 и ККМ с идентификатором 555 в TEST режиме, позволяющим эмулировать работу взаимодействия с реальными кассами KZ типа
- компания FOR STARRUS DEMO с идентификатором 666666 и ККМ с идентификатором 666 в ONLINE режиме, позволяющим эмулировать работу взаимодействия с реальными кассами СтарРус
- внешняя система `A5` с возможностью использовать от ее имени REST API используя учетную запись (`client_id/client_secret`): `8ee1c8c9-af83-4452-a643-35abe3693ccf/65MZec6B1VUMuZ2kRytINL.CCu` (scope `fisys/OutSystemAPI`)

Для Личного кабинета можно использовать тестовые учетные записи (роль: логин/пароль):

COMPANY: `admin@bar.com/admin`
INSPECTOR: `inspector@bar.com/inspector`
ADMIN: `admin@bar.com/admin`

Для остановки сервиса можно воспользоваться командой: `docker-compose -f docker-compose.yml -p assist-fisys stop`

Для удаления и очистки всего, что было создано в docker (images, containers, volumes, networks) использовать: `docker-compose -f docker-compose.yml -p assist-fisys down --rmi all -v --remove-orphans`

Настройки

Assist FiSys сервис реализован на базе [Micronaut](#) фреймворка, который имеет множество возможностей [по конфигурированию запускаемого инстанса](#). В представленной демонстрационной версии, данные настройки можно использовать в `conf/fisys/cert.env`. В этом же файле в параметре `JAVA_TOOL_OPTIONS` можно указывать специфические настройки для JVM ([OpenJDK 17](#)). Например одна из настроек (`-Dlogback.configurationFile=/var/conf/logback.xml`) указывает откуда брать настройки для логирования. Файл `conf/fisys/logback.xml` используется для задания необходимого уровня логирования, фильтрации логов, принципов формирования файлов и других приемников лог сообщений. Про его настройку можно узнать в [Logback](#) документации.

Настройки [Ory Hydra](#) находятся в файле `conf/hydra/hydra.yml`. Так же можно обратиться к поднятому контейнеру `assist-fisys/hydra` для выполнения доступных [CLI](#) команд например через `docker exec`.

В `docker-compose.yml` можно добавлять свои сервисы, менять конфигурации.

Основные настройки для `conf/fisys/cert.env`:

`micronaut.server.port` - основной порт для развертывания WEB сервиса, по умолчанию 8080 и HTTP. Есть возможность настроить на базе Micronaut HTTPS

`micronaut.router.static-resources.swagger.enabled` и `micronaut.router.static-resources.swagger-ui` или `SWAGGER_ENABLED` - (true/false) включение endpoint (/swagger-ui) для OpenAPI документации в разрабатываемом WEB сервисе

`datasources.default.url` или `MYSQL_JDBC` - URL для JDBC ресурса Базы данных

`datasources.default.driverClassName` - Java class реализации JDBC драйвера для подключаемой БД можно использовать: `com.mysql.cj.jdbc.Driver` для MySQL или `org.mariadb.jdbc.Driver` для MariaDB

`datasources.default.username` или `MYSQL_USER` - имя пользователя для подключения к БД, от имени которого будет работать сервис

`datasources.default.password` или `MYSQL_PASSWORD` - пароль для указанного пользователя БД

`flyway.datasources.default.enabled` - (true/false) включить или выключить проверку миграции схемы БД при старте инстанса сервиса

`flyway.datasources.default.locations` - список источников для схемы миграции, позволяет добавить свои скрипты например с инициализационными данными, правила именования файлов можно найти в документации [Flyway](#)

`fisys.jobs.cleaner.interval` - интервал запуска внутренней задачи по очистке очереди от устаревших задач, по умолчанию 10m - 10 минут

`fisys.jobs.cleaner.item-ttl` - минимальное время, после которого задача считается устаревшей и убирается задачей очистки из очереди, по умолчанию 10m - 10 минут

`kz-kaztelecom.dev-tools` - (true/false) включить или выключить специальный REST API сервисы для разработчиков или тестировщиков, позволяющие вмешаться в основной процесс работы системы для проверки специфических действий над ККМ в процессе интеграционных тестов, по умолчанию false - отключено

`kz-kaztelecom.ignore-server-block-msg` - (true/false) включение игнорирования команды BLOCKED от ОФД Казактелеком, использоваться может только в тестовой среде для упрощения процесса проверки, по умолчанию false - отключено

`kz-kaztelecom.socket-timeout` - таймаут на чтение данных по открытому сокету при работе с ОФД Казактелеком, указывается в миллисекундах, по умолчанию 7000 - 7 секунд

`kz-kaztelecom.retry-delay` - время в миллисекундах ожидания для повтора запроса к ОФД Казактелеком в случае провала первого, по умолчанию 2500

`kz-kaztelecom.processing-timeout` - таймаут в миллисекундах на всю операцию обмена между FiSys и ОФД Казактелеком, включая повторы, после достижения заданного времени, ККМ переходит в OFFLINE режим

`kz-kaztelecom.offline-disabled` - (true/false) включение возможности функционировать ККМ от ОФД Казактелеком в OFFLINE режиме, по умолчанию false - OFFLINE режим доступен

`kz-kaztelecom.validate-checksums-on-start` - (true/false) включение проверки состояний ККМ при старте инстанса сервиса, по умолчанию true - включено

`kz-kaztelecom.block-kkm-on-counter-mismatch` - (true/false) включение блокировки ККМ если данные ККМ в ОФД при закрытие смены не совпадает с локальными данными, по умолчанию true - блокировка включена

`kz-kaztelecom.ticket-ad-info-placeholder` - дополнительный текст, включаемый в описание состояния ККМ при его блокировке, не обязательное

`nca-trust.keystore` или `NCA_CA_CERTS` - KeyStore пакет со списком CA доверительных сертификатов в PKCS12 формате закодированный в Base64, для проверки подлинности личных сертификатов инспекторов ОФД Казактелеком, которые входят в систему используя [NCALayer](#) продукт

`nca-trust.keystore-password` или `NCA_CA_CERTS_PASS` - пароль от KeyStore `NCA_CA_CERTS`

`endpoints.monitorhealth.enabled` или `MONITOR_HEALTH_ENDPOINT_ENABLED` - (true/false) включение endpoint (/monitorhealth) с выводом текущего флага работоспособности инстанса сервиса, может использоваться системами мониторинга состояния, по умолчанию true - включено

`endpoints.health.enabled` или `HEALTH_ENDPOINT_ENABLED` - (true/false) включение endpoint (/health) с запуском и выводом подробной информации проверок работоспособности инстанса сервиса, может использоваться системами мониторинга состояния, по умолчанию false - выключено

`endpoints.monitorhealth.sensitive` - (true/false) отключение авторизации при вызове monitorhealth, по умолчанию false - авторизация не требуется

`star-rus.fake.enabled` - (true/false) поднимать ли эмулятор ККМ СтарРус, по умолчанию false - не поднимается эмулятор

`star-rus.fake.port` - номер порта на котором будет развернут эмулятор ККМ СтарРус, по умолчанию 7779