Руководство администратора

GRAVITONUM PLATFORMUM PLATFORM

ООО «ПЛАТФОРМА ГРАВИТОН»

Оглавление

Оглавление

Введение

Данное руководство предназначено для администраторов, которые самостоятельно развертывают платформу. Раздел содержит технические требования к платформе, инструкция по развертыванию платформы в Kubernetes, детали по настройки Keycloak и Git, и другие технические моменты.

Установка Gravitonum platform на Kubernetes

Требования к среде

Инструкция была протестирована на Kubernetes версии 1.18. Развертывание осуществляется с использованием helm версии 3.x.

Убедитесь, что в кластере создан Namespace и выделены ресурсы: минимальное количество памяти для развертывания платформы - 8ГБ.

Установка в Kubernetes

Перечень компонентов приложения представлен ниже:

- application
- generator
- frontend
- security-agent
- authenticator
- otp-mail
- otp-sms-over-http-qt
- notification
- notification-mail
- notification-telegram
- notification-sms-qt-over-http
- template
- address
- address-dadata

Порядок первого развертывания компонентов

1. Перейти в директорию с Helm скриптами развертывания

2. В файле ./{имя развертываемого компонента}/Chart.yaml в переменной appVersion указать актуальную версию разворачиваемого компонента. Актуальный список компонент на данный момент:

- application
- generator
- frontend
- security-agent
- authenticator
- otp-mail
- otp-sms-over-http-qt
- notification
- notification-mail
- notification-telegram
- notification-sms-qt-over-http
- template
- address
- address-dadata

3. выполнить следующую команду команду:

helm install --name-template {имя развертываемого компонента} --namespace {имя namespace приложения} ./{имя развертываемого компонента}

Порядок обновления версий компонентов

1. перейти в директорию с Helm скриптами развертывания и провести следующие шаги для каждого требуемого компонента платфомы

2. в файле ./{имя развертываемого компонента}/Chart.yaml в переменной appVersion указать актуальную версию разворачиваемого компонента

3. выполнить следующую команду команду:

helm upgrade {имя развертываемого компонента} --namespace {имя namespace приложения} ./{имя развертываемого компонента}

Проверка работоспособности

Проверить логи установки платформы можно командой:

kubectl --namespace {имя namespace приложения} get pods

Все pod’ы должны иметь Status: Running

Откройте заглавную страницу http://{имя домена} в браузере и авторизуйтесь в системе.

Описание docker-контейнеров платформы Gravitonum platform

Данный документ содержит описание компонент Gravitonum platform. Архитектурно все компоненты находятся в одной внутренней сети, могут обращаться друг к другу по доменным именам(именам контейнеров) и видят все порты друг друга.
В качестве примера будут предоставлены docker-compose файлы с настройкой non-prod и prod окружений.

application

Основной Backend компонент low-code платформы.

Для non-prod окружений так же необходимо примонтировать «persistent» папку в корневую папку проекта(указана в переменной PROJECT_ROOT) для хранения кода приложения. Папка должна быть единой для application и generator контейнеров и примонтирована с RW правами. Пример в синтаксисе docker-compose:

volumes:
- ./storage/app:/application

Список переменных окружения необходимых для запуска контейнера:
- PROJECT_NAME: наименование артефакта (artifactId) разрабатываемого приложения. Например: lk
- PROJECT_PACKAGE - наименование java пакета (groupId) разрабатываемого приложения. Например: com.acme.service
- PROJECT_ROOT - корневая папка для проекта, должно быть установлено: “/application”
- PORT - порт на котором будет запущено приложение. По умолчанию 8080
- DATASOURCE_HOST: URL сервера базы данных. Например: localhost
- DATASOURCE_PORT: порт сервера базы данных, по умолчанию 5432
- DATASOURCE_NAME: имя базы данных приложения. Например: local
- DATASOURCE_USER: имя пользователя для базы данных. Например: user
- DATASOURCE_PASSWORD: пароль пользователя базы данных. Например: Pa55w0rD
- OIDC_URL: URL сервера OpenID Connect. Например: https://auth.acme.org/auth/realms/ACME
- OIDC_CLIENT_ID: идентификатор клиента на OpenID Connect сервере. Например: integration
- ADMINISTRATOR_ROLE: наименование роли имеющей права администратора рилма OpenID Connect сервера. Например: business-admin
- MAVEN_OPTS: Только для non-prod окружений. Дополнительные опции для запуска Maven, необходимо установить следующее значение: Xms512m -Xmx1024m -XX:MetaspaceSize=256m -XX:MaxMetaspaceSize=256m -XshowSettings:vm -Dquarkus.package.type=mutable-jar
- JAVA_OPTIONS: Только для production окружений. Дополнительные опции для запуска JAVA, необходимо установить следующее значение: Xms512m -Xmx1024m -XX:MetaspaceSize=256m -XX:MaxMetaspaceSize=256m -Ddebug=false -XshowSettings:vm -Djava.util.logging.manager=org.jboss.logmanager.LogManager

Production версия приложения является stateless и может масштабироваться как вертикально, так и горизонтально.
Non-prod приложение может масштабироваться только вертикально путем выделения большего количество памяти в MAVEN_OPTS .

generator

Генератор исходного кода для разрабатываемого приложения. Должен быть запущен только на non-prod окружениях.

Так же необходимо примонтировать «persistent» папку в корневую папку проекта для хранения кода приложения. Папка должна быть единой для application и generator контейнеров и примонтирована с RW правами. Пример в синтаксисе docker-compose:

volumes:

- ./storage/app:/application

Список переменных окружения необходимых для запуска контейнера:

- PROJECT_NAME: наименование артефакта (artifactId) разрабатываемого приложения. Например: lk
- PROJECT_PACKAGE: наименование java пакета (groupId) разрабатываемого приложения. Например: com.acme.service
- PROJECT_ROOT: корневая папка для проекта, должно быть установлено: “/application”
- PROJECT_URL: application URL, должен быть установлен: *http://application:8080/application*
- DATASOURCE_HOST: URL сервера базы данных. Например: localhost
- DATASOURCE_PORT: порт сервера базы данных, по умолчанию 5432
- DATASOURCE_NAME: имя базы данных приложения. Например: local
- DATASOURCE_USER: имя пользователя для базы данных. Например: user
- DATASOURCE_PASSWORD: пароль пользователя базы данных. Например: Pa55w0rD
- CHANGESET_PRECONDITIONS_GENERATION_ENABLED: признак необходимости генерации предусловий в скриптах миграции, по умолчанию: true
- OIDC_URL: URL сервера OpenID Connect. Например: https://auth.acme.org/auth/realms/ACME
- OIDC_CLIENT_ID: идентификатор клиента на OpenID Connect сервере. Например: integration
- VCS_URL: URL репозитория для хранения данных low-code платформы. Например: https://gitlab.template.com/lk/projects/lk
- VCS_USER: имя пользователя для доступа к репозиторию для хранения данных low-code платформы. Например: user
- VCS_EMAIL: email адрес пользователя для доступа к репозиторию low-code платформы. Например: *user@acme.com*
- VCS_PASSWORD: пароль пользователя для доступа к репозиторию low-code платформы. Например: Pa55w0rD
- VCS_PLATFORM_BRANCH: наименование ветки репозитория используемой в данном окружении, по умолчанию platform
- SOURCE_PACKAGE_DOMAIN: постфикс используемый для именования пакета содержащего классы модели данных, по умолчанию domain

- SOURCE_PACKAGE_REPOSITORY: постфикс используемый для именования пакета содержащего классы репозиториев данных, по умолчанию repository
- SOURCE_PACKAGE_SERVICE: постфикс используемый для именования пакета содержащего классы сервисов, по умолчанию service
- SOURCE_PACKAGE_GRAPHQL: постфикс используемый для именования пакета содержащего классы обработчиков GraphQL API, по умолчанию graphql
- SOURCE_PACKAGE_CONTROLLER: постфикс используемый для именования пакета содержащего классы контроллеров REST ресурсов, по умолчанию controller
- SOURCE_MARKER_REPOSITORY: постфикс используемый для именования классов репозиториев данных, по умолчанию Repository
- SOURCE_MARKER_SERVICE: постфикс используемый для именования классов сервисов, по умолчанию Service
- SOURCE_MARKER_GRAPHQL: постфикс используемый для именования классов обработчиков GraphQL API, по умолчанию Endpoint
- SOURCE_MARKER_CONTROLLER: постфикс используемый для именования классов контроллеров REST ресурсов, по умолчанию Controller
- SOURCE_MARKER_QUERY: наименование класса содержащего пользовательские методы запросов на HQL, по умолчанию CustomQueryEndpoint
- SOURCE_MARKER_METHOD: наименование класса содержащего пользовательские методы GraphQL API, по умолчанию CustomMethodEndpoint
- SOURCE_REST_GENERATION_ENABLED: признак необходимости генерации REST API, по умолчанию false
- REST_BASE: базовый путь REST API, по умолчанию /api/v1
- GENERATOR_PORT: HTTP порт на котором будет запущен генератор, по умолчанию 9090
- JAVA_OPTIONS: дополнительные параметры для запуска JAVA. Значение: "-Xms256m -Xmx512m -XX:MetaspaceSize=256m -XX:MaxMetaspaceSize=256m -Ddebug=false -XshowSettings:vm -Djava.util.logging.manager=org.jboss.logmanager.LogManager"

Приложение генератора может масштабироваться только вертикально.

security-agent

Контейнер служит для коммуникации между application и Keycloak как сервера OpenID Connect.

Список переменных окружения необходимых для запуска контейнера:

- PORT - порт на котором будет запущено приложение. Должен быть выставлен как 8090
- JAVA_OPTIONS - дополнительные JAVA опции для запуска приложения, переменная должна иметь следующее значение: "-Xms128m -Xmx256m -XX:MetaspaceSize=128m -XX:MaxMetaspaceSize=256m -Ddebug=false -XshowSettings:vm -Djava.util.logging.manager=org.jboss.logmanager.LogManager"

Приложение агента stateless и может масштабироваться как вертикально, так и горизонтально.

authenticator

Сервис-посредник для реализации гибридной аутентификации в Keycloak.
Список переменных окружения необходимых для запуска контейнера:

- PORT: HTTP порт, который обслуживается сервисом
  По умолчанию: *8080*
- KC_URL: URL сервера Keycloak
  Например: *https://keycloak.acme.org/auth*
- KC_REALM: Наименование realm сервера Keycloak
  Например: *demo*
- KC_CLIENT_ID: Идентификатор приватного клиента в realm сервера Keycloak, обладающего правами редактирования данных пользователей realm
  Например: *backend-service*
- KC_CLIENT_SECRET: Секрет приватного клиента в realm сервера Keycloak
  Например: *sEcReT*
- KC_RSA_PUBLIC_KEY: публичный ключ RSA из Keycloak realm привязанного к окружению. Узнать его можно во вкладке Keys раздела Real Setting в Keycloak, необходим RSA ключ от провайдера rsa-generated

template-service

Сервис хранения и разработки шаблонов.

Список переменных окружения необходимых для запуска контейнера:

- PORT: HTTP порт, который обслуживается сервисом
  По умолчанию: *8080*
- DATASOURCE_HOST - URL базы данных
  Например: *localhost*
- DATASOURCE_PORT - Порт базы данных
  По умолчанию: *5432*
- DATASOURCE_NAME - Наименование базы данных
  Например: *local*
- DATASOURCE_USER - Имя пользователя базы данных
  Например: *user*
- DATASOURCE_PASSWORD - Пароль пользователя базы данных
  Например: *Pa55w0rD*
- OIDC_URL - URL сервера OpenID Connect
  Например: *https://auth.acme.org/auth/realms/ACME*
- OIDC_CLIENT_ID - Идентификатор публичного клиента зарегистрированного на сервере OpenID Connect
  Например: *frontend-app*
- LOG_LEVEL: Уровень логирования сервиса
  Значение из перечня TRACE|DEBUG|INFO|WARN|ERROR
  По умолчанию: *DEBUG*

notification-service

Сервис отправки уведомлений.

Список переменных окружения необходимых для запуска контейнера:

- PORT – Порт на котором будет запущен сервис. *8080* по умолчанию

- OIDC_URL - URL сервера Keycloak. Например: *https://auth.acme.org/auth/realms/ACME*
- OIDC_CLIENT_ID - Идентификатор публичного клиента в realm сервера Keycloak. Например: *frontend-app*
- TEMPLATE_SERVICE_URL – URL сервиса шаблонов. Например: *http://localhost:9001*
- MAIL_SERVICE_URL – URL сервиса email уведомлений. Например: *http://localhost:9002*. Выключено если переменная не указана.
- SMS_SERVICE_URL – URL сервиса sms уведомлений. Например: *http://localhost:9004*. Выключено если переменная не указана.

notification-mail

Сервис отправки уведомлений по e-mail.

Список переменных окружения необходимых для запуска контейнера:

- PORT – Порт на котором будет запущен сервис. *8080* по умолчанию
- OIDC_URL - URL сервера Keycloak. Например: *https://auth.acme.org/auth/realms/ACME*
- OIDC_CLIENT_ID - Идентификатор публичного клиента в realm сервера Keycloak. Например: *frontend-app*
- MAIL_HOST - URL SMTP сервера. Например: *smtp.gmail.com*
- MAIL_PORT – Порт SMTP сервера. Например: *465*
- MAIL_SSL – Статус SSL SMTP сервера. Например: *true*
- MAIL_USERNAME – Имя пользователя SMTP сервера. Например: *acme@gmail.com*
- MAIL_PASSWORD – Пароль SMTP сервера. Например: *Pa55w0rD*
- MAIL_FROM – Адрес электронной почты отправителя. Например: *noreply@acme.org*

notification-sms-qt-over-http

Сервис отправки уведомлений по протоколу HTTP через sms-шлюз «Quick Telecom».

Список переменных окружения необходимых для запуска контейнера:

- PORT: HTTP порт для обслуживания сервисом
  По умолчанию: *8080*
- OIDC_URL - OpenID Connect server URL
  Например: *https://auth.acme.org/auth/realms/ACME*
- OIDC_CLIENT_ID - OpenID Connect server client id
  Например: *dummy*
- SERVICE_URL: URL сервиса для обращения
  По умолчанию: *https://service.qtelecom.ru/public/http*
- SERVICE_USERNAME: Логин для входа в систему на сайте go.qtelecom.ru
  Например: *13579*
- SERVICE_PASSWORD: Пароль для входа в систему на сайте go.qtelecom.ru
  Например: *02468*

- MESSAGE_FROM: Наименование отправителя, зарегистрированного для вас, в системе на сайте go.qtelecom.ru
  Перед установкой данного значения требуется зарегистрировать его в личном кабинете на сайте go.qtelecom.ru или оставить по умолчанию
  Например: *Acme*
  По умолчанию: *системное имя отправителя на сайте go.qtelecom.ru по умолчанию*
- LOG_LEVEL: Уровень логгирования приложения
  Значение из перечня TRACE|DEBUG|INFO|WARN|ERROR
  По умолчанию: *DEBUG*

address

Сервис адресов.

Список переменных окружения необходимых для запуска контейнера:

- PORT: **HTTP порт, который обслуживается сервисом**
  По умолчанию: *8080*
- ROOT_PATH: **Базовый путь сервиса**
  По умолчанию: */address*
- OIDC_URL - **URL сервера OpenID Connect**
  Например: *https://auth.acme.org/auth/realms/ACME*
- OIDC_CLIENT_ID - **Идентификатор клиента зарегистрированного на сервере OpenID Connect**
  Например: *frontend*
- DATA_SERVICE_URL - **URL сервиса источника данных**
  Например: *http://localhost:9004/api/v1*
  По умолчанию: *отсутствует*
- LOG_LEVEL: **Уровень логирования сервиса**
  Значение из перечня: TRACE|DEBUG|INFO|WARN|ERROR
  По умолчанию: *DEBUG*

address-dadata

Сервис-адаптер DaData.ru.

Список переменных окружения необходимых для запуска контейнера:

- PORT: **HTTP порт, который обслуживается сервисом**
  По умолчанию: *8080*
- OIDC_URL - **URL сервера OpenID Connect**
  Например: *https://auth.acme.org/auth/realms/ACME*
- OIDC_CLIENT_ID - **Идентификатор клиента зарегистрированного на сервере OpenID Connect**
  Например: *frontend*

- DADATA_PARSE_SERVICE_URL - **URL сервиса 'Разбор адреса из строки'**
  Например: *http://localhost:9004/api/v1*
  По умолчанию: *https://cleaner.dadata.ru/api/v1/clean/address*
- DADATA_SUGGEST_SERVICE_URL - **URL сервиса 'Автодополнение при вводе'**
  Например: *http://localhost:9004/api/v1*
  По умолчанию: *https://suggestions.dadata.ru/suggestions/api/4_1/rs/suggest/address*
- DADATA_FIND_SERVICE_URL - **URL сервиса 'Адрес по коду'**
  Например: *http://localhost:9004/api/v1*
  По умолчанию: *https://suggestions.dadata.ru/suggestions/api/4_1/rs/findById/address*
- DADATA_API_KEY - **API-ключ**
  Например: *h9120b43f2fae4838a14485e43c2bfb2*
  По умолчанию: *отсутствует*
- DADATA_SECRET_KEY - **Секретный ключ**
  Например: *h9120b43f2fae4838a14485e43c2bfb2*
  По умолчанию: *отсутствует*
- LOG_LEVEL: **Уровень логирования сервиса**
  Значение из перечня: TRACE|DEBUG|INFO|WARN|ERROR
  По умолчанию: *DEBUG*

notification-telegram

Сервис отправки уведомлений в Telegram.

Список переменных окружения необходимых для запуска контейнера:

- PORT - **service HTTP port**. *8080* by default
- OIDC_URL - **OpenID Connect server URL**. ex: *https://auth.acme.org/auth/realms/ACME*
- OIDC_CLIENT_ID - **OpenID Connect server client id**. ex: *dummy*
- OIDC_CLIENT_SECRET - **OpenID Connect server client secret**. ex: *secret*
- SECURITY_AGENT_URL - **security-agent service URL**. ex: *https://security.auth.acme.org*
- BOT_NAME - **The name of a registered bot** ex.: *acmeBot*
- BOT_TOKEN - **The token of the registered bot** ex.: *0123456789:ABCDGkxHcSUVZRLc42tDtXorvio-abc0123*

otp-mail

Сервис передачи одноразового пароля по e-mail.

Список переменных окружения необходимых для запуска контейнера:

- PORT: **HTTP порт для обслуживания сервисом**
  По умолчанию: *8080*

- SERVICE_HOST: **Адрес SMTP сервера**
  Например: *smtp.acme.org*
- SERVICE_PORT: **Порт SMTP сервера**
  Например: *465*
- SERVICE_USE_SSL: **Признак использования SMTP сервером протокола SSL**
  По умолчанию: *true*
- SERVICE_USERNAME - **Имя пользователя SMTP сервера**
  Например: *user@acme.org*
- SERVICE_PASSWORD - **Пароль пользователя SMTP сервера**
  Например: *Pa55w0rD*
- MESSAGE_FROM - **Адрес электронной почты отправителя**
  Например: *noreply@acme.org*
- MESSAGE_SUBJECT - **Тема сообщения содержащего одноразовый пароль**
  Например: *"Одноразовый пароль для доступа"* По умолчанию: *""*
- MESSAGE_TEMPLATE - **Шаблон тела сообщения содержащего одноразовый пароль**
  Строка, содержащая специальные заполнители:%s - для подстановки значения одноразового пароля, %n - для переноса строки
  Например: *"Одноразовый пароль доступа:%n%n %s%n%nЭто автоматическое сообщение, созданное почтовой системой. Пожалуйста, не отвечайте на него, ваш запрос останется без ответа."*
  По умолчанию: *"%s"*
- LOG_LEVEL: **Уровень логгирования приложения**
  Значение из перечня: TRACE|DEBUG|INFO|WARN|ERROR
  По умолчанию: *DEBUG*

otp-sms-over-http-qt

Сервис передачи одноразового пароля по протоколу HTTP через sms-шлюз «Quick Telecom».

Список переменных окружения необходимых для запуска контейнера:

- PORT: **HTTP порт для обслуживания сервисом**
  По умолчанию: *8080*
- SERVICE_URL: **URL сервиса для обращения**
  По умолчанию: *https://service.qtelecom.ru/public/http*
- SERVICE_USERNAME: **Логин для входа в систему на сайте go.qtelecom.ru**
  Например: *13579*
- SERVICE_PASSWORD: **Пароль для входа в систему на сайте go.qtelecom.ru**
  Например: *02468*
- MESSAGE_TO_SOURCE: **Наименование атрибута содержащего номер телефона для отправки**

Например: *phone*
По умолчанию: *username*

- MESSAGE_FROM: **Наименование отправителя, зарегистрированного для вас, в системе на сайте go.qtelecom.ru**
  Перед установкой данного значения требуется зарегистрировать его в личном кабинете на сайте go.qtelecom.ru или оставить по умолчанию
  Например: *Acme*
  По умолчанию: *системное имя отправителя на сайте go.qtelecom.ru по умолчанию*
- MESSAGE_TEMPLATE: **Шаблон тела сообщения содержащего одноразовый пароль**
  Строка, содержащая специальный заполнитель%s - для подстановки значения одноразового пароля
  Например: *"Одноразовый пароль доступа: %s"*
  По умолчанию: *"%s"*
- DEBUG_LEVEL: **Уровень логгирования приложения**
  Значение из перечня TRACE|DEBUG|INFO|WARN
  По умолчанию: *INFO*

frontend-app

Контейнер содержащий в себе nginx и версию бизнес приложения.

Для запуска требуется примонтировать директорию с конфигурацией в /etc/nginx/conf.d/. Пример в синтаксисе docker-compose:

volumes:

- ./conf/app:/etc/nginx/conf.d/

Конфигурация будет передана отдельным файлом(nginx-app.conf), в исходной конфигурации nginx приложение работает на 80 порту(может быть изменен в файле конфигурации) и данный порт должен быть доступен пользователям.

frontend-platform

Контейнер содержащий в себе nginx и версию приложения платформы. Должен быть запущен только на non-prod окружениях.

Контейнер не имеет внешних портов, доступ к ресурсам nginx осуществляется через проксирование frontend-app контейнера.

Для запуска требуется примонтировать директорию с конфигурацией в /etc/nginx/conf.d/. Пример в синтаксисе docker-compose:

volumes:

- ./conf/platform:/etc/nginx/conf.d/

Конфигурация будет передана отдельным файлом(nginx-platform.conf).

Конфигурация Keycloak и Git сервисов

Данный документ содержит описание требований Gravitonum platform к Keycloak и Git сервисам.

Git

В конфигурации платформы указывается 5 переменных окружения связанных с git сервисом:

- VCS_URL
- VCS_USER
- VCS_EMAIL
- VCS_PASSWORD
- VCS_PLATFORM_BRANCH

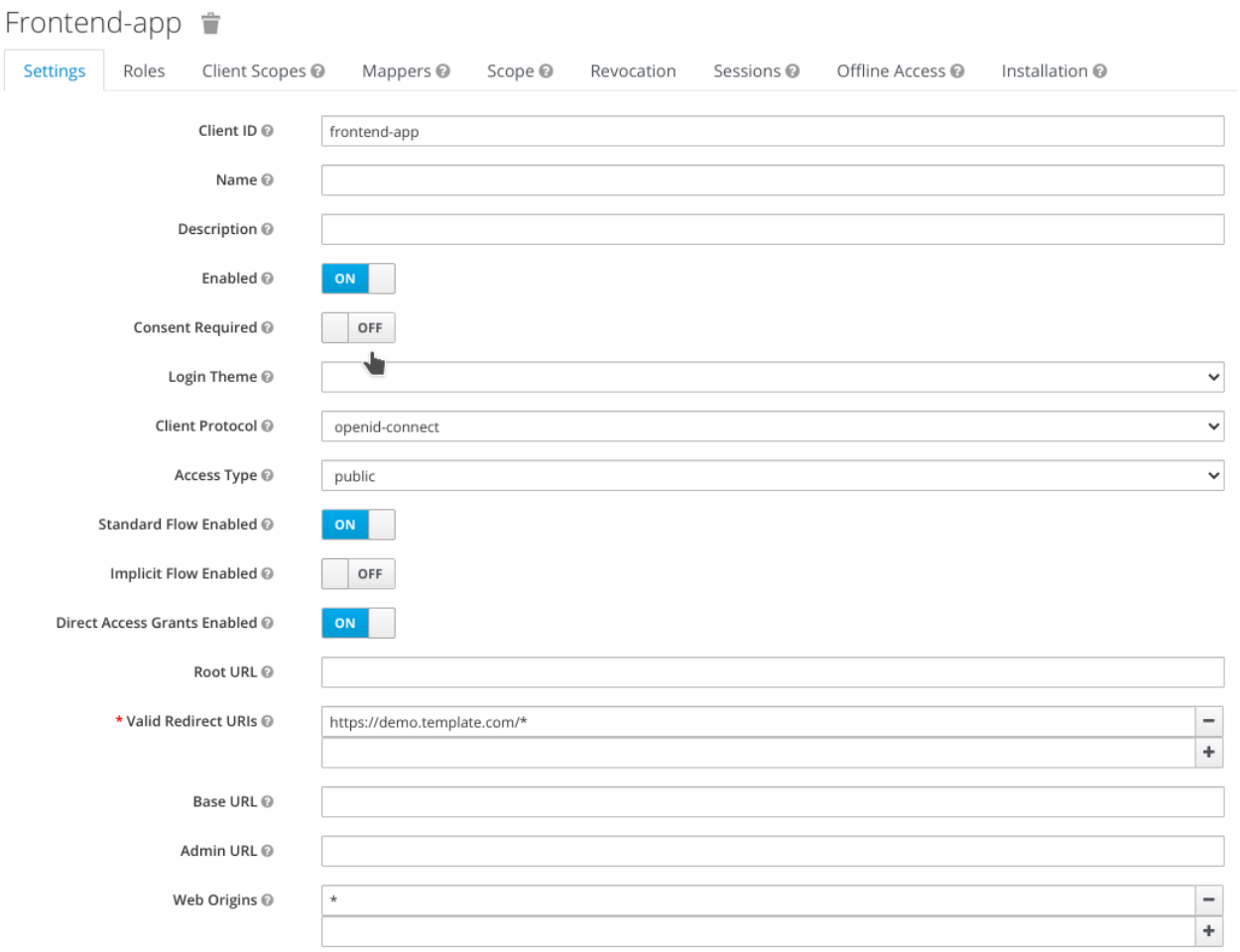
Более подробное описание переменных можно найти в документе “Описание компонент системы”.

Пользователь указанный в переменной VCS_USER должен иметь права на создание репозитория, либо права не ниже Maintainer внутри уже созданного репозитория, указанного в переменной VCS_URL

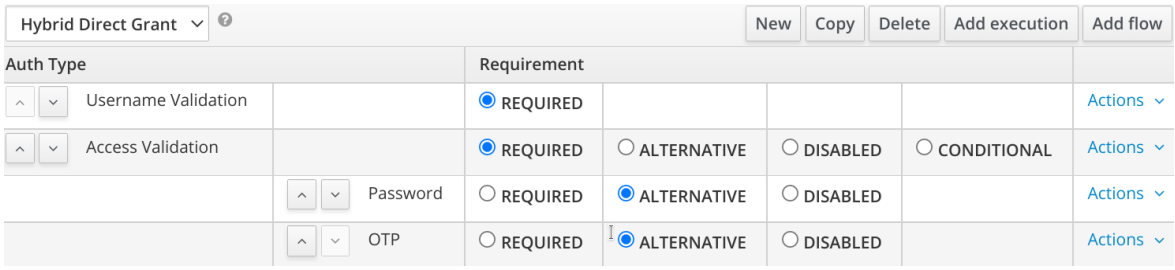
Keycloak

Требования к Keycloak realm’у указанному в переменной окружения OIDC_URL:

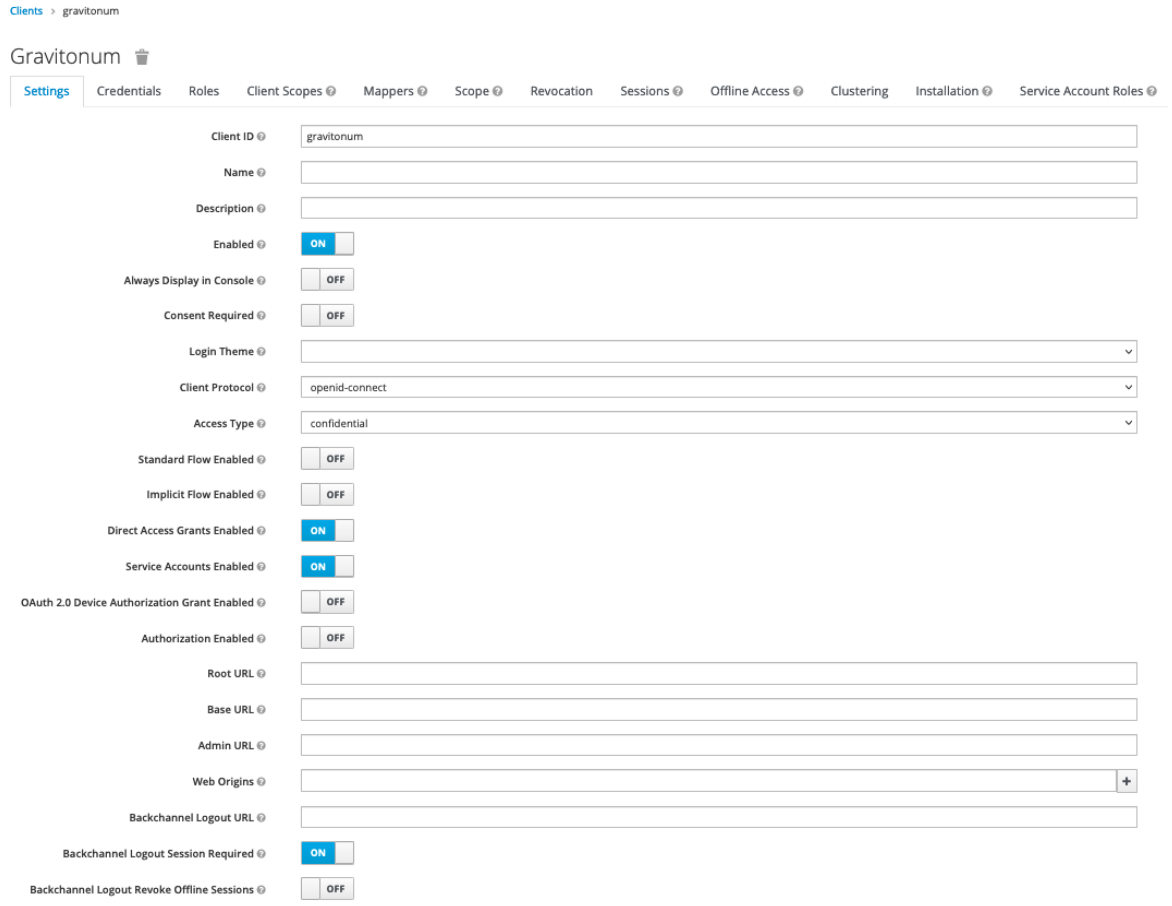
1. В realm’е должен быть создан Client указываемый в переменной окружения OIDC_CLIENT_ID со следующими параметрами:
    a. Client Protocol - openid-connect
    b. Access Type - public
    c. В Valid Redirect URIs должен быть указан адрес платформы по следующему шаблону: (http|https)://<DOMAIN_NAME>/*



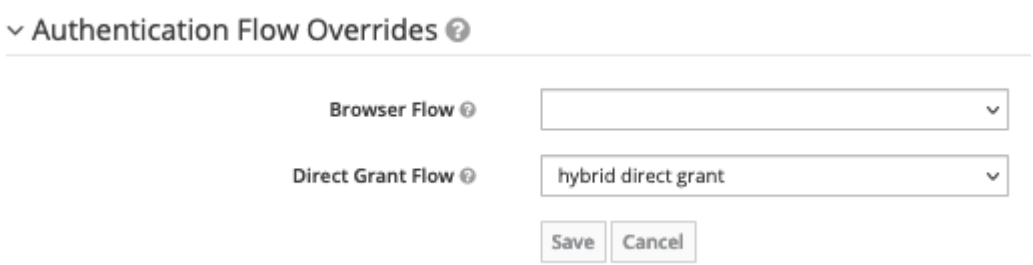
2. Для возможности использования one-time-password должен быть созданы Сценарий аутентификации(Authentication Flow)



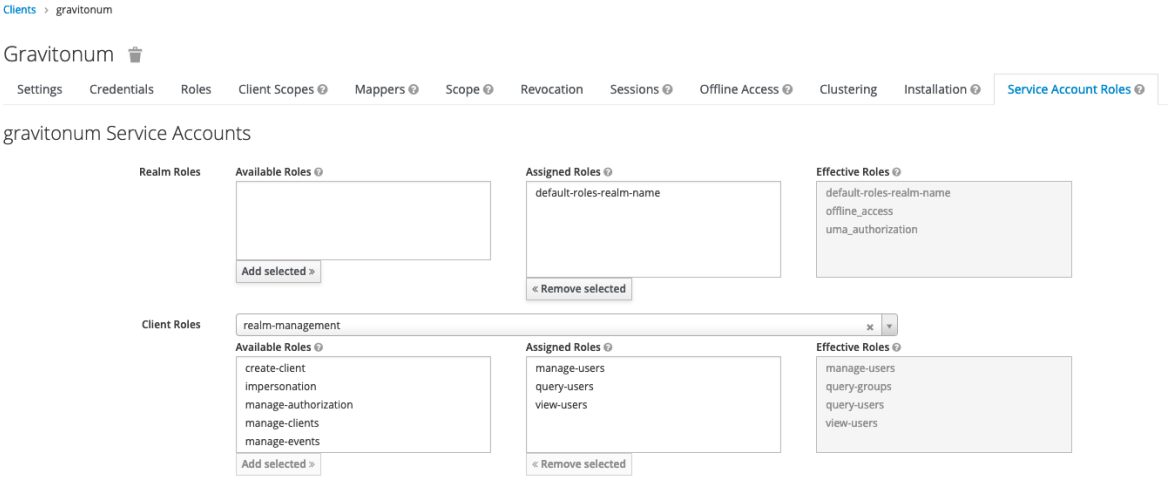
Защищённый секретом (confidential) клиент:



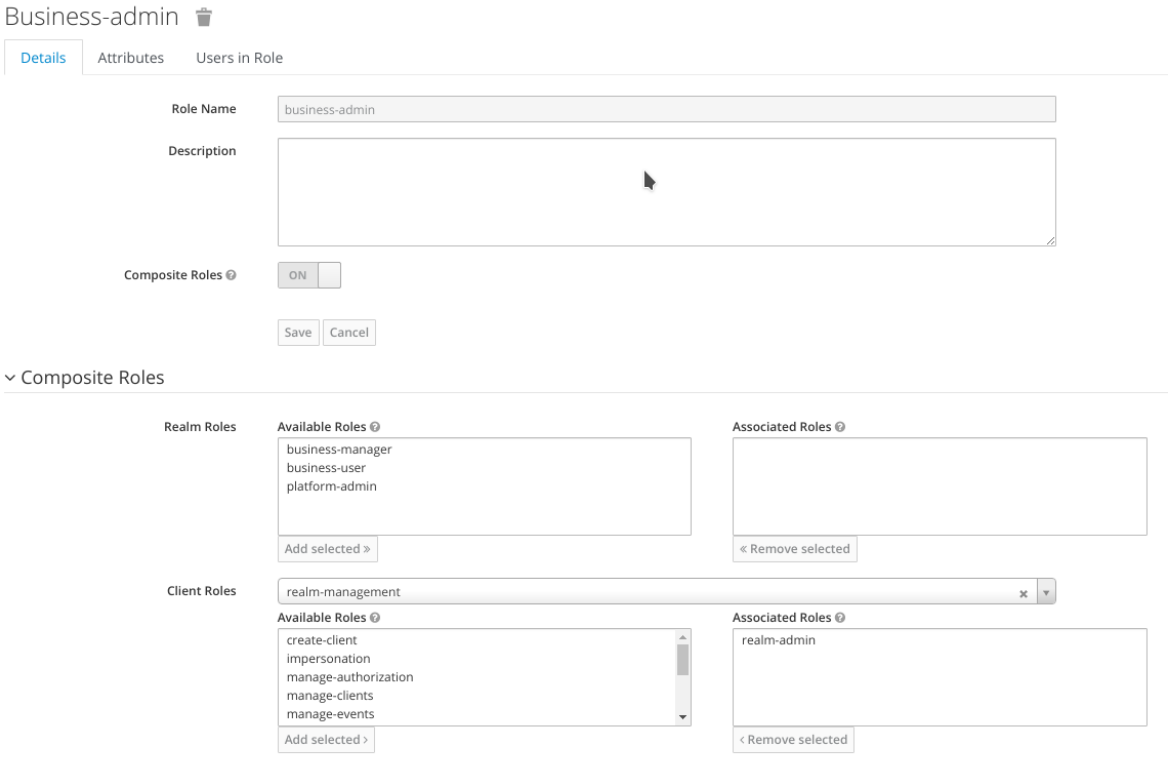
связанный с созданным сценарием аутентификации:



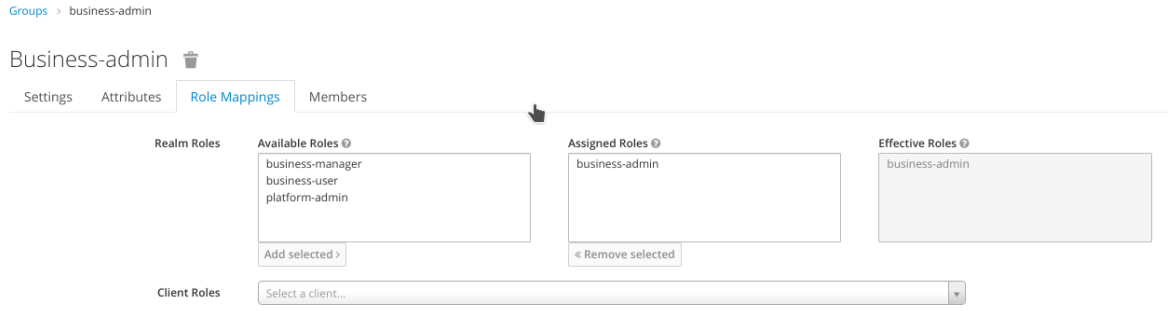
И наделённый следующими сервисными ролями(Service Account Roles):



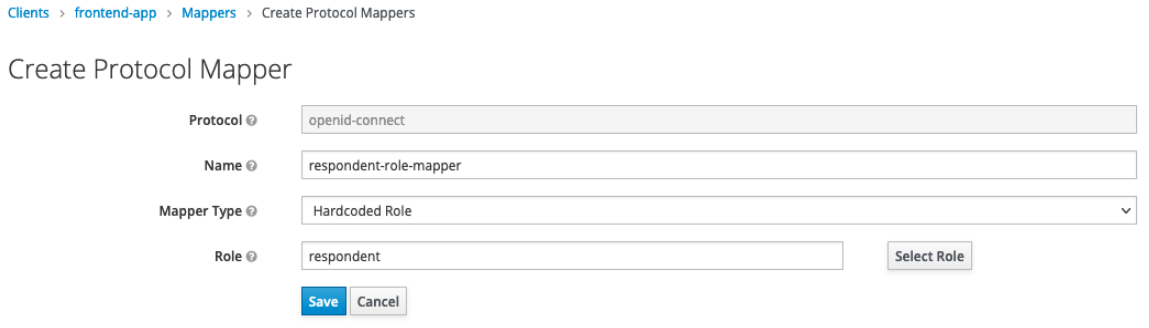
3. На nonprod окружениях должны быть созданы роли platform-admin и business-admin (на prod окружении только роль business-admin), для ролей должна быть добавлена Composite role Realm-admin как показано на скриншоте



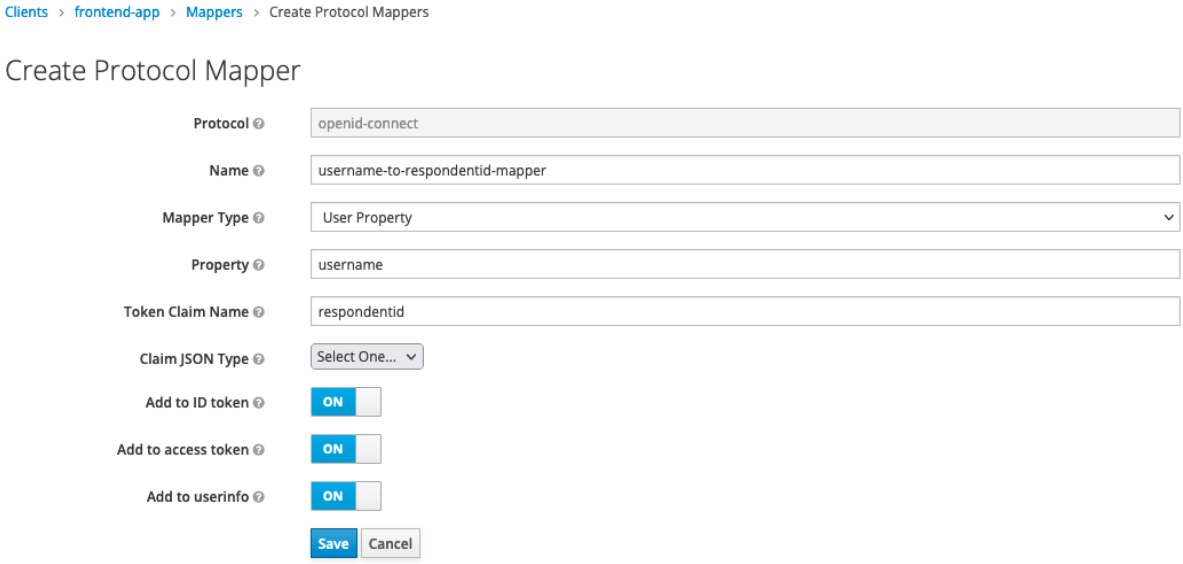
На nonprod окружениях должны быть созданы группы platform-admin и business-admin(на prod окружении только роль business-admin) с Role mapping к одноименным группам



4. Должен быть создан пользователь Administrator состоящий в группах и ролях созданных в предыдущих пунктах
5. Должна быть добавлена роль respondent
6. Для Client'а созданного в пункте 2 данного раздела создан mapper роли respondent:



7. Для Client’а созданного в пункте 2 данного раздела создан mapper username в token-claim respondentid:



Системные требования

Данный документ содержит минимальные системные требования для установки и корректной работы Gravitonum platform Enterprise. Данный документ не является руководством по конфигурированию серверов компании, а служит для приблизительной оценки совместимости оборудования компании и ПО Gravitonum platform.

Реальные требования к оборудованию зависят от практики использования системы и сильно разнятся от организации к организации.

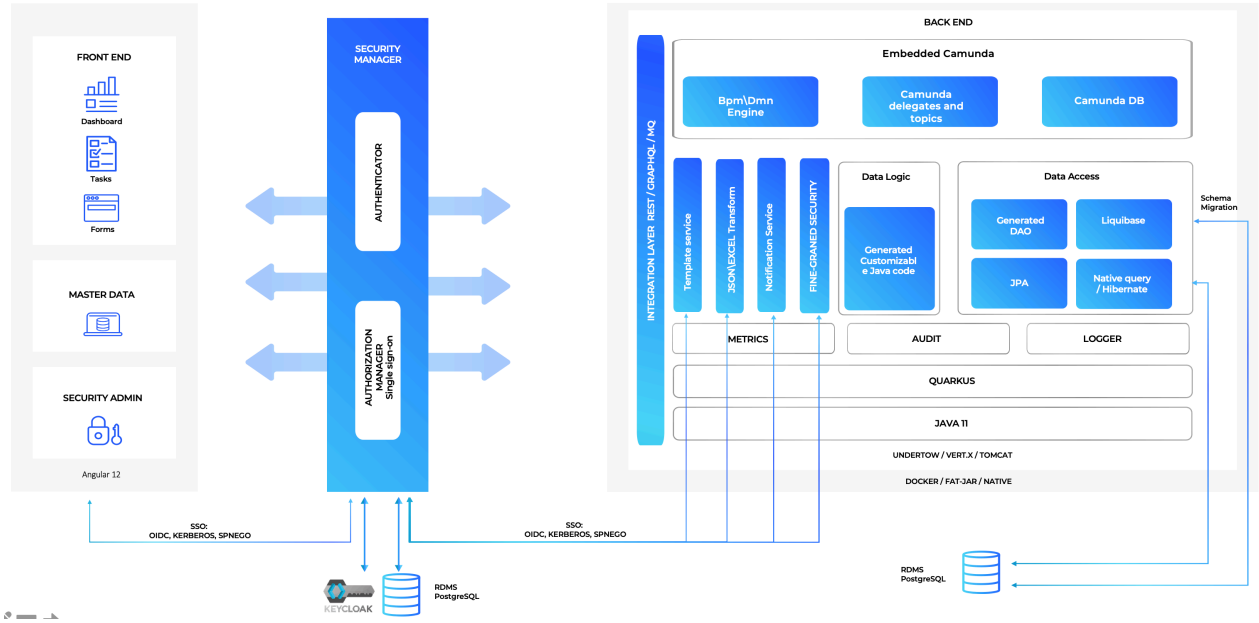


Рисунок 1. Архитектура системы

Системные требования для серверной части платформы Gravitonum platform

В таблице указаны требования к программному и аппаратному обеспечению для установки low-code платформы Gravitonum platform.

Операционная система	Минимальные аппаратные требования * в расчете на: - 100 одновременных пользователей; - 100 Экранов; - 100 Сущностей - 10 Бизнес-процессов
Centos7 и выше или RHEL 7 и выше, при условии обратной совместимости	При размещении на Kubernetes, OpenShift(OKD) CPU: 8 виртуальных ядер, не менее 2.8 ГГц на каждое ядро; • RAM: от 32 ГБ; Сетевое подключение: LAN 1 Gbit/sec и выше

PostgreSQL

Минимальные аппаратные требования * в расчете на: • 100 одновременных пользователей; • 100 Экранов; • 100 Сущностей • 10 Бизнес-процессов
• CPU: 4 ядра, тактовая частота не менее 2.8 ГГц на каждое ядро; • RAM: от 16 ГБ; • SDD: от 100 ГБ; Для промышленной среды рекомендуется использование территориально распределенная СУБД (по умолчанию Postgres) для хранения данных приложения

- Система резервного копирования должна располагаться на независимом сервере, объем доступной памяти должен соответствовать частоте резервного копирования

Системные требования для РАБОЧЕГО МЕСТА Gravitonum platform

В таблице указаны требования к программному и аппаратному обеспечению для организации рабочего места сотрудника компании. Система предоставляет доступ к данным через веб-интерфейс.

Операционная система	Поддерживаемые Веб-браузеры	Минимальные аппаратные требования
Microsoft Windows (10 и выше), Linux(Ubuntu, Debian, RedHat, SuSe), OS X	Chrome 90 и более поздние обратно-совместимые версии FireFox 88 и более поздние обратно-совместимые версии	RAM 4 Гбайт; CPU 4 ядра 2.2 Гц; Монитор FullHD, для удобной работы рекомендуется использование экрана 1920x1080.

Версионирование и поддержка

Gravitonum platform придерживается системы семантического версионирования.

Сервисный релиз

Сервисный релиз гарантирует полную обратную совместимость и предоставляет в основном исправление известных дефектов, приводящих к потере данных, значительным проблемам с производительностью, а также критических дефектов заявленной базовой функциональности. В некоторых случаях сервисный релиз может также включать улучшения API и новую функциональность при условии полной совместимости с предыдущей версией.

Минорный релиз

Минорный релиз в основном совместим с предыдущими версиями, однако может привносить существенные изменения на уровне основных возможностей и API. Все существенные изменения и инструкции к ним публикуются в списке обновлений. Зачастую необходимые изменения в исходном коде и настройках приложений автоматически применяются GRAVITONUM PLATFORM после обновления версии. Основная задача минорного релиза - внедрение новой функциональности и API при условии простой (часто автоматической) процедуры миграции с предыдущей версии.

Мажорный релиз

Задача мажорного обновления состоит в том, чтобы внедрять новые популярные концепции разработки, фреймворки, а также кардинально улучшать архитектуру платформы вместе с ее функциональными возможностями и API. Также в рамках мажорного релиза производится обновление используемых библиотек, в случае если это невозможно сделать с сохранением обратной совместимости. Такие обновления не гарантируют обратной совместимости. При этом API с предыдущих версий может быть объявлен устаревшим без его удаления.

Нестабильные и предварительные версии

Версии, которые находятся на стадии разработки:

SNAPSHOT

обозначение server.major.minor-SNAPSHOT.
Это ночная сборка ветки, находящейся в разработке. Эти версии актуальны для раннего доступа к новым возможностям.

ALPHA

Обозначение server.major.minor[.maintenance].ALPHAx

Служит предварительной версией ближайшего релиза, необходим для обкатки нового API и функциональных возможностей, при этом не гарантирует их полного сохранения в релизной версии.

BETA

Обозначение server.major.minor[.maintenance].BETAx
Служит финальной полнофункциональной версией ближайшего релиза. Вносятся только исправления дефектов. В целом представленный API и функционал сохраняются в таком же виде в релизной версии.

Используемые библиотеки в составе бизнес приложений, созданных с помощью low-code платформы Gravitonum platform

Front-end

Открытые библиотеки

Наименование библиотеки	Лицензия
angular	
@angular-devkit/build-angular	MIT
@angular/animations	MIT
@angular/cdk	MIT
@angular/common	MIT
@angular/core	MIT
@angular/forms	MIT
@angular/material	MIT
@angular/platform-browser	MIT
@angular/router	MIT
@angular/flex-layout	MIT
@angular-material-components/datetime-picker	MIT
angular-calendar	MIT
Apollo	
Apollo-angular	MIT
Apollo-angular-link-http	MIT
Apollo-angular-link-http-common	MIT
Apollo-cache	MIT
Apollo-cache-inmemory	MIT
Apollo-client	MIT

Apollo-link	MIT
Apollo-link-error	MIT
Apollo-utilities	MIT
Graphql	
Graphql	MIT
Graphql-tag	MIT
Camunda	
Base64-js	MIT
Bpmn-moddle	MIT
Camunda-bpmn-moddle	MIT
Core-js	MIT
Is-retina	MIT
Js-sha256	MIT
Keyclock	
Keycloak-angular	MIT
Keycloak-js	Apache License 2.0
Other	
Luxon	MIT
Min-dash	MIT
Moddle	MIT
Moddle-xml	MIT
Ngx-avatar	MIT
ts-input-mask	MIT
Regenerator-runtime	MIT
Rxjs	Apache License 2.0
Saxen	MIT
Ts-md5	MIT

Tslib	BSD Zero Clause License
Webpack	MIT
Zone.js	MIT
@ngneat/until-destroy	MIT
@ngx-loading-bar/core	MIT
@ngx-loading-bar/router	MIT
json5	MIT
apexcharts	MIT
ng-apexcharts	MIT
@swimlane/ngx-datatable	MIT
highlight.js	BSD 3-Clause "New" or "Revised" License
angular-code-input	MIT
swiper	MIT
xml-beautifier	MIT
quill	BSD 3-Clause "New" or "Revised" License
ngx-quill	MIT
ngx-mat-select-search	MIT
ngx-monaco-editor	MIT
monaco-editor	MIT
simplebar	MIT
convert-layout	MIT
echarts	Apache License 2.0
ngx-echarts	MIT
ng-zorro-antd	MIT
File-saver	MIT
ngx-img-cropper	MIT
date-fns	MIT
memoizee	ISC

@babel/standalone	MIT
ngx-material-timepicker	MIT
angular-imask	MIT
cytoscape	MIT
ngx-markdown	MIT
@ngrx/component-store	MIT
@ngrx/effects	MIT
@ngrx/entity	MIT
@ngrx/router-store	MIT
@ngrx/store	MIT
quill-delta-to-html	MIT
phone-formatter	MIT
mime	MIT
libphonenumber-js	MIT
tailwindcss	MIT

Проприетарные библиотеки

@gravitonum/ng-sdk - лицензия Gravitonum platform
@gravitonum/platform-sdk - лицензия Gravitonum platform
@gravitonum/renderer - лицензия Gravitonum platform
@gravitonum/validation-module - лицензия Gravitonum platform
@gravitonum/renderer-components - лицензия Gravitonum platform
gravitonum-business-app - лицензия Gravitonum platform

Back-end

Открытые библиотеки

Наименование библиотеки	Лицензия
quarkus	Apache-2.0
jackson	Apache-2.0
camunda	Apache-2.0
camel-quarkus	Apache-2.0
junit	EPL-2.0
quarkus-jgit	Apache-2.0
quarkus-logging-manager	Apache-2.0
jnats	Apache-2.0
json-logic-java	MIT
hibernate-types	Apache-2.0
spoon-core	MIT
commons-exec	Apache-2.0
poi	Apache-2.0
spqr	Apache-2.0
quarkus-minio	Apache-2.0
quarkus-jackson-jq	Apache-2.0
jxls	Apache-2.0
zt-zip	Apache-2.0
telegrambots	MIT
reflections	Apache-2.0

Проприетарные библиотеки

Проприетарные библиотеки - внешние бинарные библиотеки, от которых зависит исходный код итогового разработанного бизнес-приложения.

- ✓ com.gravitonum.platform.vendor
- ✓ com.gravitonum.platform.storage
- ✓ com.gravitonum.platform.security

- ✓ com.gravitonum.platform.health
- ✓ com.gravitonum.platform.metrics
- ✓ com.gravitonum.platform.storage-db
- ✓ com.gravitonum.platform.camunda
- ✓ com.gravitonum.platform.audit
- ✓ com.gravitonum.platform.exchange
- ✓ com.gravitonum.platform.excel-transform
- ✓ com.gravitonum.platform.logging
- ✓ com.gravitonum.platform.storage-object
- ✓ com.gravitonum.platform.json-transform
- ✓ com.gravitonum.platform.validation

Проприетарные сервисы

Проприетарные внешние сервисы платформы.
Код итогового разработанного бизнес-приложения может использовать API, который предоставляют данные сервисы.

- ✓ security-agent
- ✓ authenticator
- ✓ otp-mail
- ✓ otp-sms-over-http-qt
- ✓ notification
- ✓ notification-mail
- ✓ notification-telegram
- ✓ notification-sms-qt-over-http
- ✓ template
- ✓ address
- ✓ address-dadata

Требования к настройке логов

Команде разработки Gravitonum platform должен быть обеспечен доступ к логам окружения одним из следующих методов:

1. SSH доступ на сервер с окружением с правами достаточными для чтения логов
2. Доступ к ELK, EFK, Loki-Grafana или любому другому технологическому стеку заказчика предоставляющему историю логов и поиск по ним
3. Предоставление возможности разворачивания Filebeat на сервере для использования EFK стека на нашей стороне.