

API Gateways для открытия OpenAPI наружу

В данной статье приведен список шлюзов API для публикации OpenAPI наружу и инструкция по созданию «ручки» (URL метода API) наружу.

- Для QA: <link>
- Для DEV: <link> <link> <link>
- Список API Gateways
 - Stage API Gateways
 - Prod API Gateways
 - ECOM Stage API Gateways
 - ECOM Prod API Gateways
- Конфигурация
 - Открытие ручек на шлюз (gateway)
 - Пример:
 - Описание полей apiGateways
 - Открытие ручки на admin-gw для Admin UI
 - Конфигурация сервиса для shp-gw
 - Получение JWT для запроса к сервису через опубликованную ручку на api-gateway закрытую авторизацией из AD через Postman
 - Получить JWT для запроса к сервису через опубликованную ручку на api-gateway закрытую авторизацией из AD через bash скрипт
- Как добавить внутренний и внешний API Gateway (инструкция для администраторов)

Для QA: <link>

Для DEV: <link> <link> <link>

Список API Gateways



Если вы не знаете, какой gateway использовать для вашего сервиса, обратитесь к архитектурному комитету. См. [Архитектурное ревью в PaaS](#).

Stage API Gateways

- stf-gw.k-stage.sbermarket.tech
- shp-gw.k-stage.sbermarket.tech - **auth enabled**
- admin-gw.k-stage.sbermarket.tech
- admin-gw.k-stage.sbmt.io
- chat-bo-gw.k-stage.sbmt.io
- chat-widget-gw.k-stage.sbermarket.tech
- exponea-gw.k-stage.sbermarket.tech
- exponea-gw.k-stage.sbmt.io
- retailers-gw.k-stage.sbermarket.tech
- retailers-gw.k-stage.sbmt.io
- retailers-admin-gw.k-stage.sbmt.io
- api-gw.k-stage.sbermarket.tech
- api-gw.k-stage.sbmt.io
- paas-gw.k-stage.sbmt.io - НЕ ИСПОЛЬЗОВАТЬ
- crm-gw.k-stage.sbermarket.tech - **auth enabled (partner_crm_authentication)**
- job-gw.k-stage.sbermarket.tech - **auth enabled**
- [merchant-api](#).k-stage.sbermarket.tech
- analytics-gw.k-stage.sbmt.io - **ТОЛЬКО ПРИВАТНЫЕ РУЧКИ. Обязателен флаг: private: true**
- dam-gw.k-stage.sbermarket.tech
- dam-gw.k-stage.sbmt.io
- integrations-gw.k-stage.sbermarket.tech
- integrations-open-gw.k-stage.sbermarket.tech
- pims-gw.k-stage.sbermarket.tech
- daas-gw.k-stage.sbermarket.tech
- daas-gw.k-stage.sbmt.io
- [hrau-gw.k-stage.sbmt.io](#)
- [retailers-x5-gw.k-stage.sbermarket.tech](#) (включен mTLS, необходим клиентский сертификат)
- [sber-integration-gw.k-stage.sbermarket.tech](#) (включен mTLS, необходим клиентский сертификат)
- ris-gw.k-stage.sbmt.io - [INFRADEV-70616](#) - Получение подробных данных проблемы... СТАТУС
- ads-gw.k-stage.sbmt.io - [INFRADEV-80509](#) - Получение подробных данных проблемы... СТАТУС

Prod API Gateways

- stf-gw.sbermarket.ru
- shp-gw.sbermarket.ru - **auth enabled**
- admin-gw.sbermarket.ru
- admin-gw.sbmt.io
- chat-bo-gw.sbmt.io
- exponea-gw.sbermarket.ru
- exponea-gw.sbmt.io
- retailers-gw.sbermarket.ru
- retailers-gw.sbmt.io
- retailers-admin-gw.sbmt.io
- api.sbermarket.ru
- api.sbmt.io
- chat-widget-gw.sbermarket.ru
- sm-api.sbermarket.ru - [PROJ-404](#) - Получение подробных данных проблемы... СТАТУС
- paas-gw.sbmt.io - НЕ ИСПОЛЬЗОВАТЬ
- crm-gw.sbermarket.ru - **auth enabled**
- job-gw.sbermarket.ru - **auth enabled**
- [merchant-api.sbermarket.ru](#)
- [analytics-gw.sbmt.io](#) - ТОЛЬКО ПРИВАТНЫЕ РУЧКИ. Обязателен флаг: `private: true`
- dam-gw.sbermarket.ru
- dam-gw.sbmt.io
- integrations-gw.sbermarket.ru
- integrations-open-gw.kuper.ru
- daas-gw.sbermarket.ru
- daas-gw.sbmt.io
- [retailers-x5-gw.sbermarket.ru](#) (включен mTLS, необходим клиентский сертификат)
- [sber-integration-gw.sbermarket.ru](#) (включен mTLS, необходим клиентский сертификат)
- ris-gw.sbmt.io - [INFRADEV-70616](#) - Получение подробных данных проблемы... СТАТУС
- ads-gw.sbmt.io - [INFRADEV-80509](#) - Получение подробных данных проблемы... СТАТУС

ECOM Stage API Gateways

- paas-gw.gw-stage-ecom-tech.sbmt.io - НЕ ИСПОЛЬЗОВАТЬ
- hrau-gw.gw-stage-ecom-tech.sbmt.io
- integrations-gw.gw-stage-ecom-tech.sbmt.io
- admin-gw.gw-stage-ecom-tech.sbmt.io

ECOM Prod API Gateways

- paas-gw.ecom-tech.sbmt.io - НЕ ИСПОЛЬЗОВАТЬ
- hrau-gw.ecom-tech.sbmt.io
- integrations-gw.ecom-tech.sbmt.io
- admin-gw.ecom-tech.sbmt.io

Конфигурация



Конфиг работает и для фича, и для стейбл стейджа, но для начала его работы нужно обязательно раскатить стейбл

Открытие ручек на шлюз (gateway)

Для открытия ручки на API GW нужно в `./configs/values/app.yaml` описать блок `apiGateways`.

Пример:

```
issuer: &issuer
issuer:
  stage: https://paas-keycloak-stage.gw-stage.sbmt.io/auth/realms/sbermarket.ru
  prod: https://paas-keycloak.sbmt.io/auth/realms/sbermarket.ru
```

```

enableApiGwAuth: &enableApiGwAuth
  auth:
    enabled: true
    issuer:
      stage: https://paas-keycloak-stage.gw-stage.sgmt.io/auth/realms/sbermarket.ru
      prod: https://paas-keycloak.sgmt.io/auth/realms/sbermarket.ru

apiGateways:
- name: shp-gw
  targetApp: app
  rateLimiter:
    alertChannel:
      _default: custom-alert-chanel-for-all-locations
    _default:
      - unit: second
        requestsPerUnit: 20
        limitByIP: true
        method: GET
  locations:
    - name: partners-login
      path: /auth/v1/partners/login
      pathType: exact
      rewrite: /api/v1/auth/partners/login
      matchMethods:
        _default:
          exact: "GET"
      prod:
        regex: "^GET|POST$"
    - path: /auth/v1/partners/code
      pathType: prefix
      rewrite: /api/v1/auth/partners/code
      rateLimiter:
        alertChannel:
          _default: custom-alert-chanel-for-location
        _default:
          - unit: minute
            requestsPerUnit: 20
    - path: /auth/v1/strict_path
      pathType: exact
      rewrite: /api/v1/strict_path
      auth:
        enabled: true
        allowOptions: true
        issuer: " ", auth-back"
    - path: /auth/v1/ip-allow-list-path
      pathType: exact
      rewrite: /api/v1/ip-allow-list-path
      ipAllowList:
        _default:
          - 192.168.1.1/24
          - 169.254.1.1/32
      auth:
        enabled: true
        byToken:
          token:
            _default: PAAS__VAULT_VAR__MOCK_DUMMY_TOKEN
    - path: /auth/v1/cors-path
      pathType: exact
      rewrite: /api/v1/cors-path
      corsPolicy:
        _default:
          allowOrigins:
            - exact: https://example.com
          allowMethods:
            - POST
            - GET
          allowCredentials: false
          allowHeaders:
            - X-Foo-Bar
          maxAge: "24h"
    - path: /auth/v1/groups-protected-handle

```



```



pathType: exact
rewrite: /v1/groups-protected-handle
auth:
  enabled: true
  allowOptions: true
  allowedRoles:
    _default: ["SberMarket Team", "SBMT Team", "res.huggin.ext_users"]
  <<: *issuer
- path: /odin/api/v1/services
  pathType: exact
  rewrite: /api/v1/services
  private: true
  <<: *enableApiGwAuth
  methods:
    - GET
- path: /odin/api/v1/services
  pathType: exact
  rewrite: /api/v1/services
  private: true
  methods:
    - POST
- path: /odin/api/v1/services/([^/]+)/ci_jobs/([^/]+)/cancel
  pathType: regex
  rewrite: /api/v1/services/\1/ci_jobs/\2/cancel
  additionalRouteHeaders:
    stage:
      - name: sbm-forward-tenant
        value: metro
  tests:
    - candidatePath: /odin/api/v1/services/2348/ci_jobs/1123211/cancel
      expectedPath: /api/v1/services/2348/ci_jobs/1123211/cancel
    - candidatePath: /odin/api/v1/services/N2348/ci_jobs/1123211 #This test will be failed for example
      expectedPath: /api/v1/services/2348/ci_jobs/1123211/cancel
- path: /superapp/directresponse1
  pathType: exact
  directResponse:
    status: 503
  body:
    string: "unknown error"


```

Описание полей apiGateways


Название поля	Тип	Обязательность	Описание
name	string	Обязательно	Определяет на каком из gw будет открыта ручка. Возможные значения: <i>shp-gw, stf-gw, admin-gw, exponea-gw, retailers-gw, retailers-admin-gw, api-gw, paas-gw, job-gw, crm-gw</i>
targetApp	string	Необязательно	Имя деплоймента приложения, на который будет направлен трафик по описанным ниже ручкам (см. locations)
private	boolean	Необязательно	Возможные значения: <ul style="list-style-type: none"> • true — открыть ручки только на *.sbmt.io • false — ручки открыты без ограничений
rateLimiter		Необязательно	<u>Список</u> конфигураций rate limiter на все ручки (locations) сервиса
• unit	string	Обязательно	Доступные значения: day, hour, minute, second'
• request sPerUnit	integer	Обязательно	Количество запросов в период, определенный в unit

<ul style="list-style-type: none"> • limitByIP 	boolean	Необязательно	<p>Возможные значения:</p> <ul style="list-style-type: none"> • true — выставленное ограничение будет применено для каждого уникального IP адреса с которого приходит запрос(используются заголовки: x-real-ip для публичных и x-forwarded-for для sbmt.io) • false — ограничение не применяется
<ul style="list-style-type: none"> • method 	string	Необязательно	HTTP-метод
locations		Обязательно	Список ручек
<ul style="list-style-type: none"> ▪ name 	string	Необязательно	Имя маршрута, без этого поля имя будет генерироваться по индексу
<ul style="list-style-type: none"> • path 	string	Обязательно	<p>Путь, по которому ручка будет доступна на api-gw. Важно валидировать путь, чтобы не перекрыть своим эндпойнтом другие сервисы. Список опубликованных ручек смотрите на https://paas-huginn.sbmt.io/map?filterLink=declared&filterTypeNode=gateway.</p> <div>  Сортировка приоритета роутинга с одним и тем же pathType происходит по количеству символов в path. </div>
<ul style="list-style-type: none"> • pathType 	string	Обязательно	<p>Доступные значения:</p> <ul style="list-style-type: none"> • exact - точное совпадение(рекомендуется к использованию), например /api соответствует /api • prefix - совпадение по префиксу, например /api соответствует /api/* • regex - точное совпадение по регулярному выражению. <div>  Роутинг происходит по наиболее точному совпадению. По pathType сортировка происходит так: <ol style="list-style-type: none"> 1. Exact - высший приоритет 2. Далее regex 3. И в конце prefix </div> <p>На текущий момент для regex действуют ограничения, проверяемые линтером OpenAPI спек. А именно:</p> <ol style="list-style-type: none"> 1. Все сегменты пути, для которых указаны группы в регулярном выражении, имеют значение "([^\s/]+)". Также допускается использовать именованные группы, вида: "(?P<client_id>[^\s/]+)" (правило sm-gateway-location-regex-generic-groups). 2. Все группы, объявленные в location.path используются для замены в location.rewrite (правило sm-gateway-location-regex-all-groups-used) 3. Если разбить path на подстроки по символу '/', то полученные строки содержат хотя бы одну подстроку, которая начинается и заканчивается круглыми скобками. (sm-gateway-location-regex-has-groups) 4. Если разбить path на подстроки по символу '/', то полученные строки либо будут начинаться и заканчиваться круглыми скобками, либо содержать только символы, подходящие под регулярное выражение "[a-zA-Z0-9_\-\.]+\$" (правило sm-gateway-location-regex-segment-chars)
<ul style="list-style-type: none"> • matchMethods 		Необязательно	<p>С помощью двух ручек с разным значением поля matchMethods вы можете задать разное поведение для двух HTTP методов с одинаковым path, например на GET запросы делать redirect, а POST запросы отправлять на сервис.</p> <p>Если HTTP метод не совпал со значением в matchMethods, то API Gateway перебирает другие ручки, пока не будет достигнуто совпадение. Если совпадение не достигнуто, API Gateway ответит на запрос кодом HTTP 403 Forbidden.</p> <p>Доступные значения:</p> <ul style="list-style-type: none"> ◦ exact - точное совпадение (рекомендуется к использованию), например POST / GET. В поле exact можно указать только один HTTP метод, если вам нужно задать список, используйте regex. ◦ regex - совпадение по регулярному выражению, например "^POST GET\$"
<ul style="list-style-type: none"> • methods 		Необязательно	<p><u>Опциональный белый список</u> HTTP методов, лимитирующих правило. Если не задан, применяется к любым методам.</p> <p>С помощью поля methods вы можете запретить обращение к ручке по всем HTTP методам. кроме заданных. Если HTTP метод не совпал с заданными значениями, то API Gateway прекратит перебор ручек и ответит на запрос кодом HTTP 403 Forbidden. Так, например, вы можете запретить обращение к ручке по всем HTTP методам кроме GET и POST.</p>
<ul style="list-style-type: none"> • rewrite 		Обязательно	Путь который будет получать сервис

• private	boolean	Необязательно	Возможные значения: <ul style="list-style-type: none"> • true, то открыть ручку только на *.sbmt.io • false, то ручка открыта всем
• auth		Необязательно	Блок настроек авторизации для ручек
◦ enabled	boolean	Обязательно	Возможные значения: <ul style="list-style-type: none"> • true — включить авторизацию через auth service (заголовок: Authorization: Bearer <TOKEN>) • false — авторизация отключена <div>  Если auth.enabled: true, но опция byToken не включена, необходимо также заполнить поле auth.issuer (istio на api-gw будет проверять, что токен выдан определенным issuer). Если не заполнить auth.issuer, то автоматически поставится issuer authentication_service@sbermarket/* (auth-back для shp-gw). См. пример с enableApiGwAuth. Это значит, что при настройке только enabled: true авторизация возможна только с помощью JWT токена с указанием issuer. Если вы хотите настроить авторизацию с "рандомными" токенами, полученными где-то в другом месте, к примеру, токены сессии stf - необходимо отключить авторизацию вовсе. </div>
◦ issuer	string	обязательно*	Обязательно, если auth.enabled:true Заполняется, если включена авторизация, но не заполнена опция byToken Если значение пустое, то автоматически поставится issuer authentication_service@sbermarket/* (auth-back для shp-gw)
◦ allowOptions	boolean	необязательно	Возможные значения: <ul style="list-style-type: none"> • true - разрешить OPTIONS запросы без авторизации, использовать, когда браузер отправляет preflight запросы • false - OPTIONS запросы запрещены
◦ byToken		необязательно	Блок настроек авторизации по application token (решение пока нет поддержки application токенов в auth-сервисе. Заголовок Authorization: <TOKEN>), <u>невозможно использовать эту опцию одновременно с опцией allowedGroups</u> <ul style="list-style-type: none"> ◦ token - строка токена (есть возможность использовать секрет из волта). Пример: PAAS__VAULT_VAR__RETAILERS_DEPZDRAV_API_GW_TOKEN, где PAAS__VAULT_VAR__ — префикс, который используется для вызова функции поиска секрета в волте, RETAILERS_DEPZDRAV_API_GW_TOKEN — это имя секрета в волте.
◦ allowedGroups		необязательно	Список ActiveDirectory групп, разрешенных для ручки (включает авторизацию по Active Directory группам, <u>невозможно использовать эту опцию одновременно с опцией byToken</u>). Также необходимо задать поле issuer (также есть в примерах). <div>  Опция доступна пока только на paas-gw. </div>
◦ allowedRoles		необязательно	Список keycloak ролей, разрешенных для ручки
• ipAllowList		необязательно	Список IP-адресов или блоков IP-адресов, которые имеют доступ к ручке(по умолчанию разрешено: 0.0.0.0/0). Описывать по окружениям(stage, prod) или в блоке _default
• corsPolicy		необязательно	Настройки CORS. https://istio.io/latest/docs/reference/config/networking/virtual-service/#CorsPolicy . Описывать по окружениям(stage, prod) или в блоке _default
• rateLimiter		необязательно	<u>Список</u> конфигураций rate limiter на ручку. Для prefix и regex действует общий счетчик rate limiter.

◦ unit	string	Обязательно	Доступные значения: day, hour, minute, second'
◦ requestsPerUnit	integer	Обязательно	Количество запросов на период времени, указанный в параметре unit
◦ limitByIP	boolean	необязательно	Возможные значения: <ul style="list-style-type: none"> • true — выставленное ограничение будет применено для каждого уникального IP адреса с которого приходит запрос • false — ограничение не применяется
◦ limitByUser	boolean	необязательно	Возможные значения: <ul style="list-style-type: none"> • true — выставленное ограничение будет применено для каждого user_uid из JWT токена (сейчас работает только на shp-gw) • false — ограничение не применяется
◦ limitByHeader	string	необязательно	Строковое значение, если заполнено, то будет работать ограничение по заголовку(пример: limitByHeader: Sbm-Auth-Email)
◦ method	string	необязательно	Если указан, то применяется лимит только для него
◦ overwriteDefaults	boolean	необязательно	Возможные значения: <ul style="list-style-type: none"> • true — значения, указанные для rateLimiter для данной ручки (location), переопределяют общие значения rateLimiter для всех ручек • false — используются общие значения rateLimiter для всех ручек
• action	string	необязательно	Возможные значения: <ul style="list-style-type: none"> • allow. По умолчанию allow. • deny. С помощью deny можно запретить доступ к ручке  - вопрос от пользователей, для чего это может быть нужно в реальной среде?
• additionalRouteHeaders		необязательно	Дополнительные заголовки, при наличии которых трафик будет направляться на данную ручку. Может быть полезно, если, например, необходимо, чтобы одна и та же ручка могла вести на разные деплойменты в зависимости от значения какого-либо заголовка (sbm-forward-tenant в STF активно используется в качестве такого заголовка). <ul style="list-style-type: none"> ◦ env (stage prod) <ul style="list-style-type: none"> ▪ <u>список</u> name (имя заголовка) + value (значение заголовка)
• directResponse		необязательно	Блок настроек, чтобы на определенные пути ответ на запрос отдавал Gateway без отправки в сервис
◦ status	string	обязательно	Код ответа
◦ body	string	обязательно	Тело ответа
▪ tests	list	необязательно	Список проверок для указанных path, который будет запущен в jobs linter: api-gw-routes в пайплайне

<ul style="list-style-type: none"> o candidatePath 	string	обязательно	Request_uri приходящее на gateway. В kibana эквивалентно полю path в ingressgateway
<ul style="list-style-type: none"> o expectedPath 	string	обязательно	Request_uri отправленное в приложение. В kibana эквивалентно полю rewrite_path в ingressgateway

 Настройки rate limiter (в `apiGateways/rateLimiter`) будут добавлены в список лимитов из `apiGateways.rateLimiter` (если такие заданы на весь гейтвей)

Если необходимо переписать глобальные настройки для конкретной ручки, в раздел `apiGateways/locations/.../rateLimiter` добавьте опцию **`overwriteDefaults`**: [true](#).

```
apiGateways:
- name: some-gw
  rateLimiter:
    _default:
      - unit: second
        requestsPerUnit: 1
        limitByIP: true
  locations:
    - path: /some/v1/example
      pathType: exact
      rewrite: /new/v1/example
      rateLimiter:
        overwriteDefaults: true
        _default:
          - unit: second
            requestsPerUnit: 60
            limitByIP: true
```

Открытие ручки на admin-gw для Admin UI

Этот пример работает для:

- Admin UI
- Есть аутентификация по JWT-токену (в том числе при использовании сервиса авторизации. См. [Сервис авторизации: быстрый старт](#))


```

apiGwCorsPolicy: &apiGwCorsPolicy
  stage:
    allowOrigins:
      - exact: https://paas-content-admin-front-admin-platform.gw-stage.sbmt.io //      stage
    allowMethods:
      - POST
      - GET
      - OPTIONS
      - PUT
      - PATCH
      - DELETE
    maxAge: "24h"
    allowHeaders:
      - Authorization
      - Content-Type
  prod:
    allowOrigins:
      - exact: https://admin.sbmt.io //      prod
    allowMethods:
      - POST
      - GET
      - OPTIONS
      - PUT
      - PATCH
      - DELETE
    maxAge: "2h"
    allowHeaders:
      - Authorization
      - Content-Type

apiGateways:
- name: admin-gw
  private: true
  targetApp: app
  locations:
    - path: /randomservice //
      corsPolicy:
        <<: *apiGwCorsPolicy
      pathType: prefix
      rewrite: /
      auth:
        enabled: true
        allowOptions: true
        issuer:
          stage: https://keycloak-stage.gw-stage.sbmt.io/auth/realms/admin
          prod: https://keycloak.sbmt.io/auth/realms/admin

```

Конфигурация сервиса для shp-gw


Для роутинга на shp-gw, используйте следующие настройки в values.yaml:

```

apiGateways:
- name: shp-gw
  locations:
    - path: < >
      pathType: exact
      rewrite: < >
      auth:
        enabled: true

```

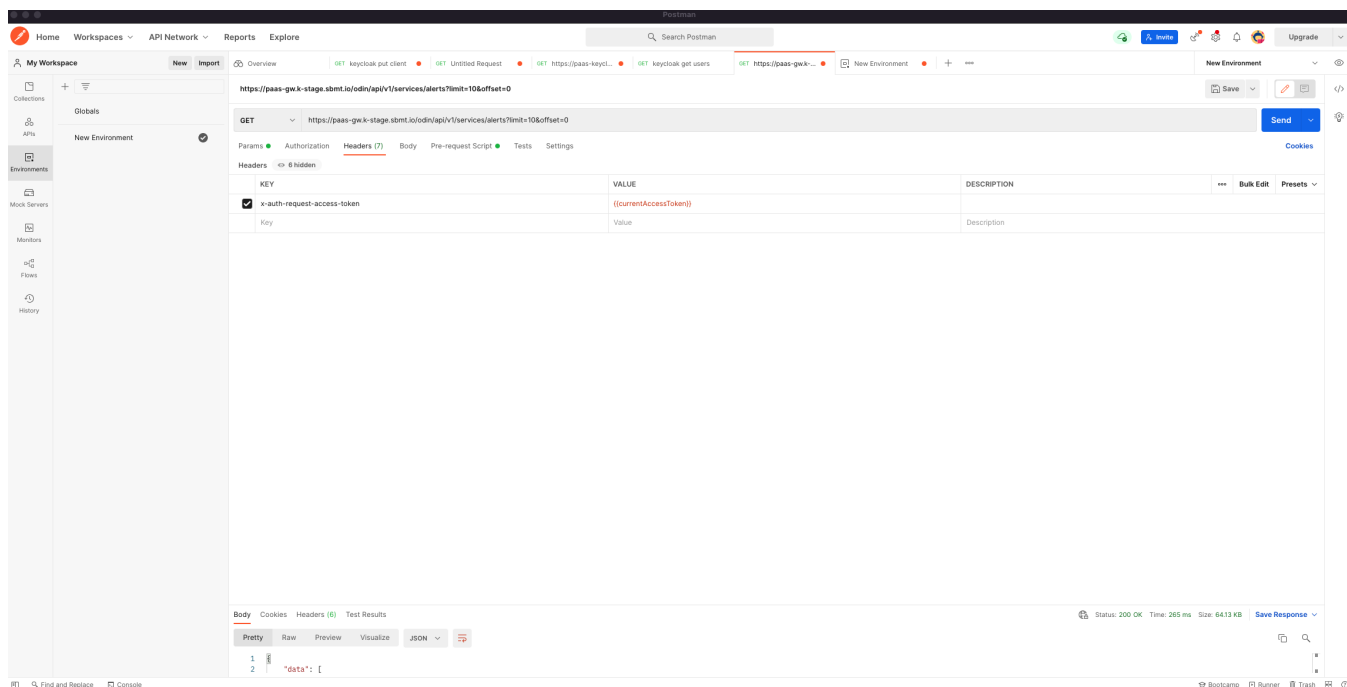
Получение JWT для запроса к сервису через опубликованную ручку на api-gateway закрытую авторизацией из AD через Postman

 Данный способ работает только на stage.

Чтобы получить JWT для запроса к сервису, сделайте следующее:

1. Добавьте в Headers следующую строку:

```
x-auth-request-access-token: {{currentAccessToken}}
```



2. В Pre-request Script Postman вставьте следующий код:

```

const echoPostRequest = {
  url: 'https://paas-keycloak-stage.gw-stage.sgmt.io/auth/realms/sbermarket.ru/protocol/openid-connect/token',
  method: 'POST',
  header: 'Content-Type:application/x-www-form-urlencoded',
  body: {
    mode: 'urlencoded',
    urlencoded: [
      { key: 'username', value: pm.environment.get('adUsername') },
      { key: 'password', value: pm.environment.get('adPassword') },
      { key: 'grant_type', value: 'password' },
      { key: 'client_id', value: 'paasapp-test' }
    ]
  }
}

var getToken = true;

if (!pm.environment.get('accessTokenExpiry') ||
    !pm.environment.get('currentAccessToken')) {
  console.log('Token or expiry date are missing')
} else if (pm.environment.get('accessTokenExpiry') <= (new Date()).getTime()) {
  console.log('Token is expired')
} else {
  getToken = false;
  console.log('Token and expiry date are all good');
}

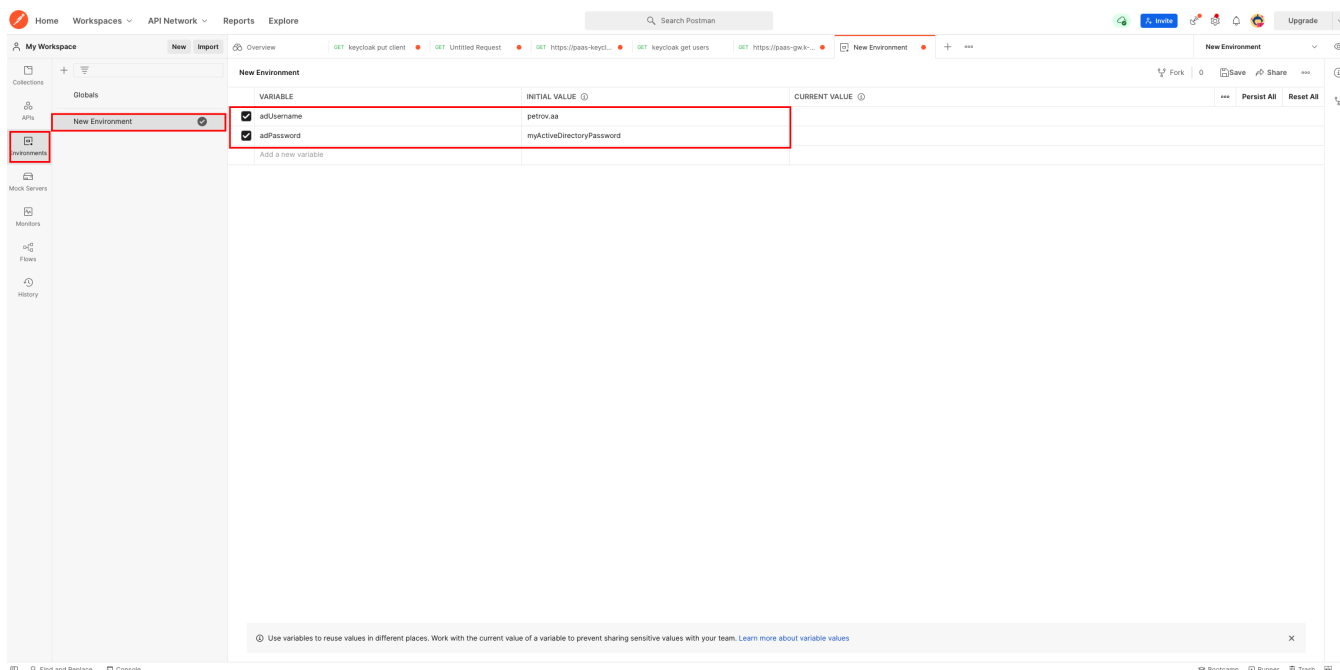
if (getToken === true) {
  pm.sendRequest(echoPostRequest, function (err, res) {
    console.log(err ? err : res.json());
    if (err === null) {
      console.log('Saving the token and expiry date')
      var responseJson = res.json();
      pm.environment.set('currentAccessToken', responseJson.access_token)

      var expiryDate = new Date();
      expiryDate.setSeconds(expiryDate.getSeconds() + responseJson.expires_in);
      pm.environment.set('accessTokenExpiry', expiryDate.getTime());
    }
  });
}

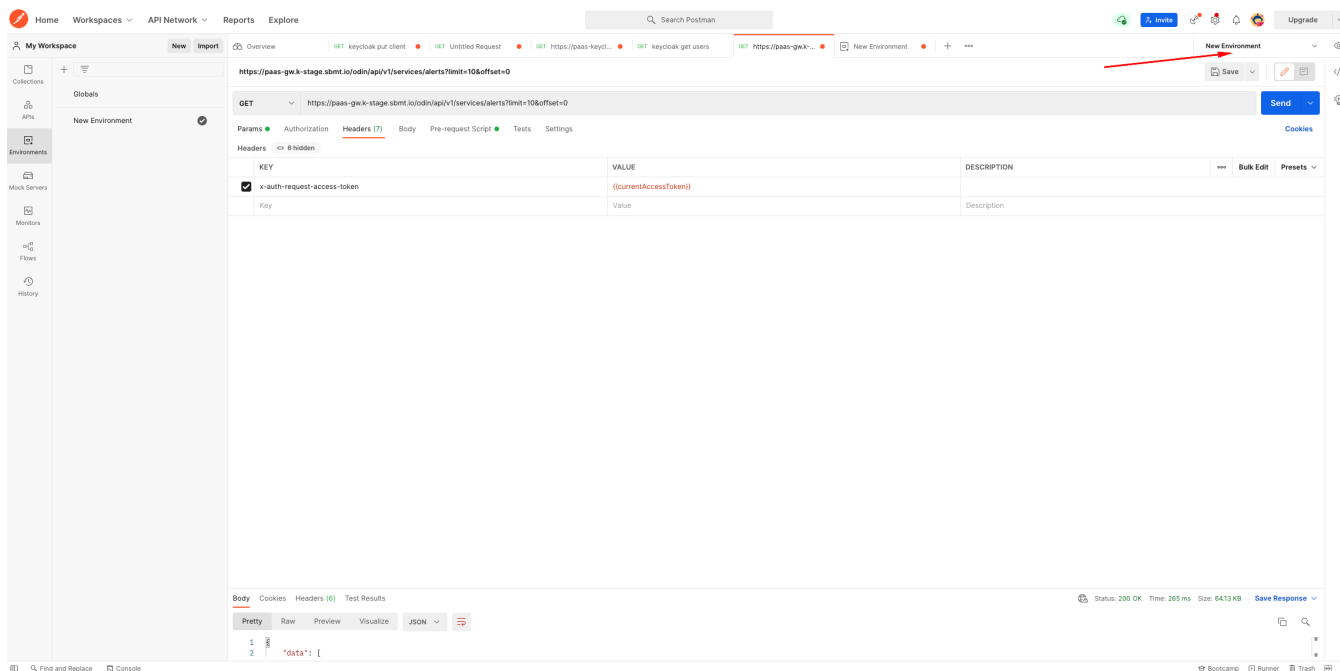
```

The screenshot shows the Postman application interface. On the left, the 'My Workspace' sidebar is visible with various tool categories. The main workspace displays a REST client request for the endpoint `https://paas-gw-k-stage.sgmt.io/odn/api/v1/services/alerts?limit=10&offset=0`. The 'Pre-request Script' tab is selected, showing the JavaScript code that manages the token and expiry date. The 'Body' tab shows the request body in JSON format. The status bar at the bottom indicates a successful status of 200 OK, with a response time of 265 ms and a size of 64.13 KB.

3. Создайте environment и добавьте в него логин/пароль Active Directory (adUsername, adPassword):



4. При выполнении запроса выберите созданный Environment:



Получить JWT для запроса к сервису через опубликованную ручку на api-gateway закрытую авторизацией из AD через bash скрипт



keycloak-curl.sh

Получить через bash скрипт

```
./keycloak-curl.sh paas-keycloak-stage.gw-stage.sbmt.io sbermarket.ru surname.ip paasapp-test y
```

Как добавить внутренний и внешний API Gateway (инструкция для администраторов)

Чтобы добавить внутренний *api-gw (internal)*, сделайте следующее:

1. Добавьте gateway в [helmfiles](#) для каждого environment без указания **loadBalancerIP** (будет создан в момент деплоя *api-gw*) (например <https://gitlab.sbmt.io/infra/k8s/helmfiles/-/blob/master/environments/stage/istiooperatorgateways.yaml#L810>)
 - Получить список используемых **nodePort**, а затем выбрать свободный можно командой:

```
kubectl --context yc-stage get svc --all-namespaces --sort-by='.spec.ports[0].nodePort' -o go-template="{{range .items}}{{range .spec.ports}}{{if .nodePort}}{{.nodePort}}{{printf \"\\n\\n\"}}{{end}}{{end}}{{end}}"
```

2. Проверьте логи *istio-operator* на наличие ошибок, а именно создался ли целевой namespace для *api-gw*

```
kubectl --context yc-stage -n istio-operator logs deployments/istio-operator-1-17-2
```

3. Далее переходите к инструкции по добавлению внешнего *api-gw* и выполните все шаги.
4. После выполнения деплоя в репозитории [api-gateways](#) будет создан балансировщик нагрузки. Добавьте его адрес в [helmfiles](#) строка с **loadBalancerIP: ip-адрес, полученный балансировщиком нагрузки**
5. Создайте **DNS запись** для *api-gw* в репозитории [yc-infra](#).

Чтобы добавить внешний *api-gw*, сделайте следующее:

1. Зарезервируйте IP-адрес для gateway в облаке, например [для stage](#) (кнопка зарезервировать адрес)
2. Добавьте gateway в [helmfiles](#) для каждого environment (например <https://gitlab.sbmt.io/infra/k8s/helmfiles/-/blob/master/environments/stage/istiooperatorgateways.yaml#L810>) копируем какой-нибудь предыдущий gateway, в зависимости от потребности:
 - с внутренним ip или внешним
 - стоящим за *nginx* (на таком будет аннотация **"proxy.istio.io/config": '{ "gatewayTopology" : { "numTrustedProxies": 2 } }**) и *nodePort* (*http2*, *https*, *healthchecks*) или без *nginx*в **loadBalancerIP** вписываем зарезервированный в облаке ip адрес
3. Прокатите *istio_install:stage / istio_install:prod*. В неймспейсе, указанном в *yaml*, появится *deployment* (или *daemonset* в зависимости от настройки в *yaml*) и *service*
4. Добавьте файлы настроек в репозиторий [api-gateways](#) (например <https://gitlab.sbmt.io/paas/api-gateways/-/blob/master/values-m-gw-stage.yaml>).
5. Добавьте джобы в <https://gitlab.sbmt.io/paas/api-gateways/-/blob/master/.gitlab-ci.yml>
6. Посмотрите диффы и прокатите джобы, должны появиться сами gateways (и в зависимости от настроек *envoyfilters* и *custom resource ratelimitservice*)
7. Если *api gateway* будет стоять за *nginx*, то создайте заявку для безопасников (<https://jira.sbmt.io/servicedesk/customer/portal/25/group/179>), вписав туда gateway domain и ip адрес, например

```
    ngenix      api gateways
m.k-stage.sbermarket.tech
ip 51.250.34.74

m.sbermarket.ru
ip 51.250.43.115
```

8. Добавьте gateway в деплой paas https://gitlab.sbmt.io/paas/cicd/deploy/-/blob/0.4/helm-charts/common/templates/_api_gateways.tpl и <https://gitlab.sbmt.io/paas/cicd/deploy/-/blob/0.4/helm-charts/values.yaml#L68>
9. Добавьте в blackbox и ssl exporters и создайте заявку в <https://jira.sbmt.io/servicedesk/customer/portal/5/> на доступ для экспортеров в стейдж
10. Добавьте gateway в список API Gateways (в начале этой страницы)
11. В на бонус не забываем про сберклауд (на момент написания этого комментария деплой в сбер находится в разработке, так что первый кто доберется должен будет дописать доку. https://gitlab.sbmt.io/infra/terraform/sbercloud-advanced/sbc-stage-staging/-/merge_requests/65)