v1.18_Johnny

Eltex-johnny - , API Wi-Fi. , SoftWLC. Wi-Fi , SoftWLC.

```
apt-get install eltex-johnny
```

/

	service eltex-johnny status	Active: active (running) C Active: failed
	service eltex-johnny start	Active: active (running)

	service eltex-johnny stop	Active: failed
	service eltex-johnny restart	Active: active (running)

/etc/eltex-johnny/application.conf

.

application.conf

```
pcrf {
  // connection host
  host = localhost
  // connection port (7070 is default)
  port = 7070
  // timeout (you can use ns, us, ms, s, m and h letters)
  // consult with HOCON duration format for more information
  timeout = 10s

  // pool configuration
  pool {
    // minimum idle objects in pool
    min = 1
    // maximum pool size
    max = 20
    // timeout to retrieve an object from pool
    waitTimeout = 5s
  }
}


mercury {
   // connection host
   host = localhost
   // connection port (6565 is default)
   port = 6565

   // pool configuration
   pool {
     // minimum idle objects in pool
     min = 1
     // maximum pool size
     max = 20
     // timeout to retrieve an object from pool
     waitTimeout = 5s
   }
}

nbi {
  // connection host
  host = localhost
  // connection port (8080 is default)
  port = 8080
```

```
  // timeout (you can use ns, us, ms, s, m and h letters)
  // consult with HOCON duration format for more information
  timeout = 80s
  // NBI login
  login = admin
  // NBI password
  password = password

  // pool configuration
  pool {
    // minimum idle objects in pool
    min = 1
    // maximum pool size
    max = 20
    // timeout to retrieve an object from pool
    waitTimeout = 5s
  }
}

portal {
  scheme = http
  host = localhost
  port = 9000
}

doors {
  host = localhost
  port = 9097
  path = /api
  username = user
  password = password
}

http {
  connectionTimeout = 20s
  connectionTotal = 100
}

// JWT validation. You need a key from Eltex Doors.
// Or you could generate it yourself.
validation {
  public_key = /etc/eltex-doors/keys/public.pem
}
```

- **pcrf** - eltex-pcrf, , .
- **mercury** - eltex-mercury, , .
- **nbi** - , nbi, , .
- **portal** - , eltex-portal.
- **doors** - , , API eltex-doors.
- **http** - API.
- **validation** - eltex-doors, .

/etc/eltex-johnny/log4j2.xml

.

log4j2.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<Configuration monitorInterval="10">
    <Properties>
        <Property name="rootLevel">${env:LOG_LEVEL:-ERROR}</Property>

        <Property name="baseDir">/var/log/eltex-johnny</Property>
        <Property name="maxFileSize">20 MB</Property>
        <Property name="accumulatedFileSize">100 MB</Property>
        <Property name="lastModified">7d</Property>
        <Property name="maxCount">20</Property>
        <Property name="logPattern">%d{ISO8601} [%t] %-5p %logger{12} %C{1}.%M(line:%L). %m%n</Property>

        <Property name="gelfLevel">${env:GELF_LEVEL:-OFF}</Property>
        <Property name="gelfHost">${env:GELF_HOST:-udp:lab3-test.eltex.loc}</Property>
        <Property name="gelfPort">${env:GELF_PORT:-12201}</Property>
    </Properties>
    <Appenders>

        <RollingFile name="RollingFile"
                     fileName="${baseDir}/johnny.log"
                     filePattern="${baseDir}/log/johnny-%i.log.gz">
            <PatternLayout pattern="${logPattern}"/>
            <Policies>
                <SizeBasedTriggeringPolicy size="${maxFileSize}"/>
            </Policies>
            <DefaultRolloverStrategy max="${maxCount}">
                <Delete basePath="${baseDir}" maxDepth="3">
                    <IfFileName glob="*/johnny-*.log.gz">
                        <IfLastModified age="${lastModified}"/>
                        <IfAny>
                            <IfAccumulatedFileSize exceeds="${accumulatedFileSize}"/>
                            <IfAccumulatedFileCount exceeds="${maxCount}"/>
                        </IfAny>
                    </IfFileName>
                </Delete>
            </DefaultRolloverStrategy>
        </RollingFile>

        <Gelf name="Gelf" host="${gelfHost}" port="${gelfPort}" version="1.1" facility="eltex-johnny"
              extractStackTrace="true" originHost="%host{fqdn}" maximumMessageSize="8192">
            <Field name="thread" pattern="%t"/>
            <Field name="level" pattern="%level"/>
            <Field name="severity" pattern="%-5level"/>
            <Field name="logger" pattern="%logger{12}"/>
            <Field name="location" pattern="%C{1}.%M(line:%L)"/>
        </Gelf>
    </Appenders>

    <Loggers>
        <Root level="${rootLevel}">
            <AppenderRef ref="RollingFile"/>
            <AppenderRef ref="Gelf" level="${gelfLevel}"/>
        </Root>

        <Logger name="org.springframework" level="ERROR"/>

        <Logger name="org.apache" level="OFF"/>

    </Loggers>

</Configuration>
```

- :

```
<Configuration monitorInterval="10">
```

- :

```
<Property name="rootLevel">${env:LOG_LEVEL:-ERROR}</Property>
```

- :

```
 <Property name="baseDir">/var/log/eltex-johnny</Property>
```

- (.):

```
<Property name="maxFileSize">20 MB</Property>
```

- (+). :

```
<Property name="accumulatedFileSize">10 GB</Property>
```

- , :

```
<Property name="lastModified">4d</Property>
```

- , :

```
<Property name="maxCount">20</Property>
```

- Graylog (, ,):

```
<Property name="gelfLevel">${env:GELF_LEVEL:-OFF}</Property>
<Property name="gelfHost">${env:GELF_HOST:-udp:lab3-test.eltex.loc}</Property>
<Property name="gelfPort">${env:GELF_PORT:-12201}</Property>
```

/etc/default/eltex-johnny

.

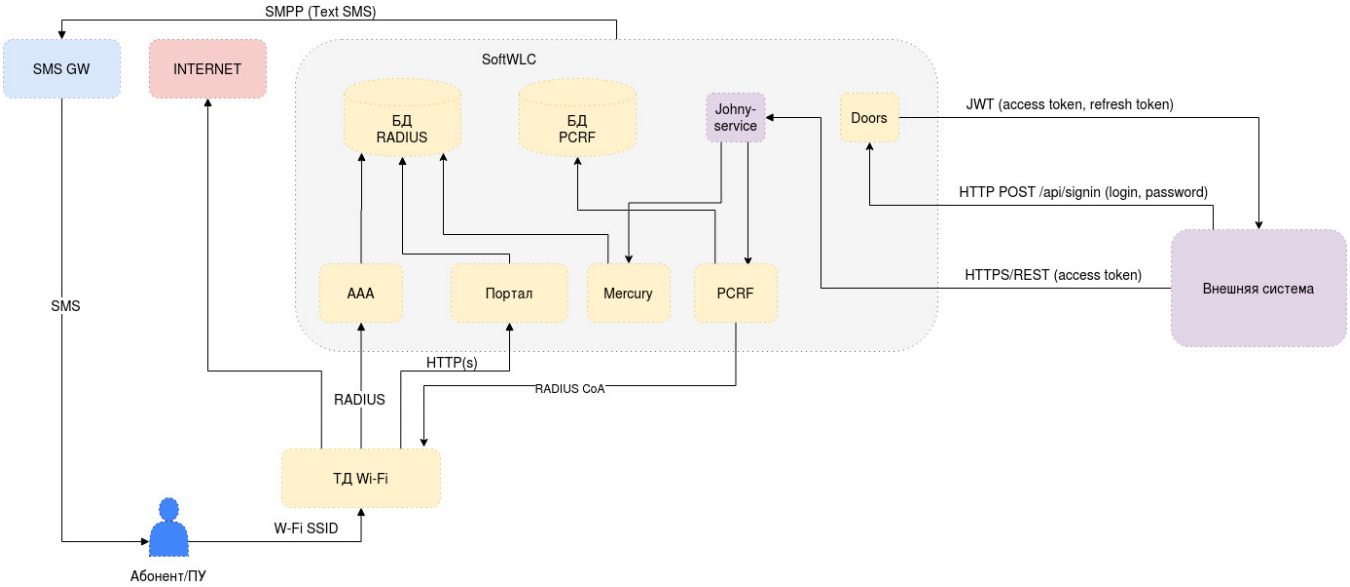
eltex-johnny

```
PORT=9100

# Initial size of Java heap
JAVA_INIT_HEAP=64m
# Maximum size of Java heap
JAVA_MAX_HEAP=256m

# Additional arguments to pass to java
JAVA_OPTS="-XX:+HeapDumpOnOutOfMemoryError -XX:HeapDumpPath=/var/log/eltex-johnny"
```

PORT	,

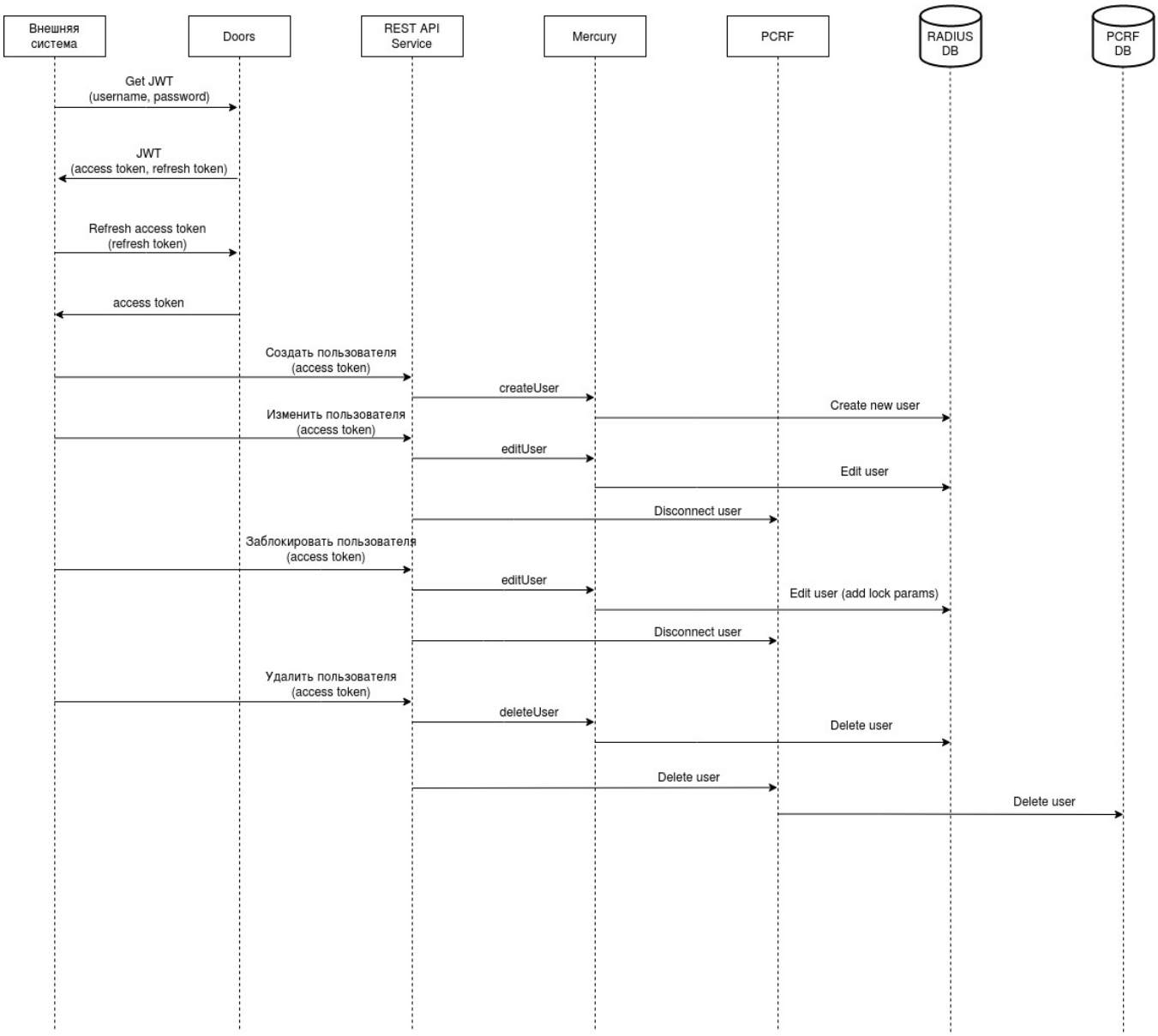
JAVA_INIT_HEAP	, . JAVA_MAX_HEAP.
JAVA_MAX_HEAP	,
JAVA_OPTS	jvm

API



B2C

(B2C)



API Open API (B2C)

API

```
openapi: "3.0.1"
info:
  title: OSS Interaction Service API
  version: "1.0"

components:
  securitySchemes:
    JwtAuth:
      type: http
      scheme: bearer
      bearerFormat: JWT

  schemas:
    username:
      type: integer
      description: '  ( ).     7  15 '
      example: '79123456789'
```

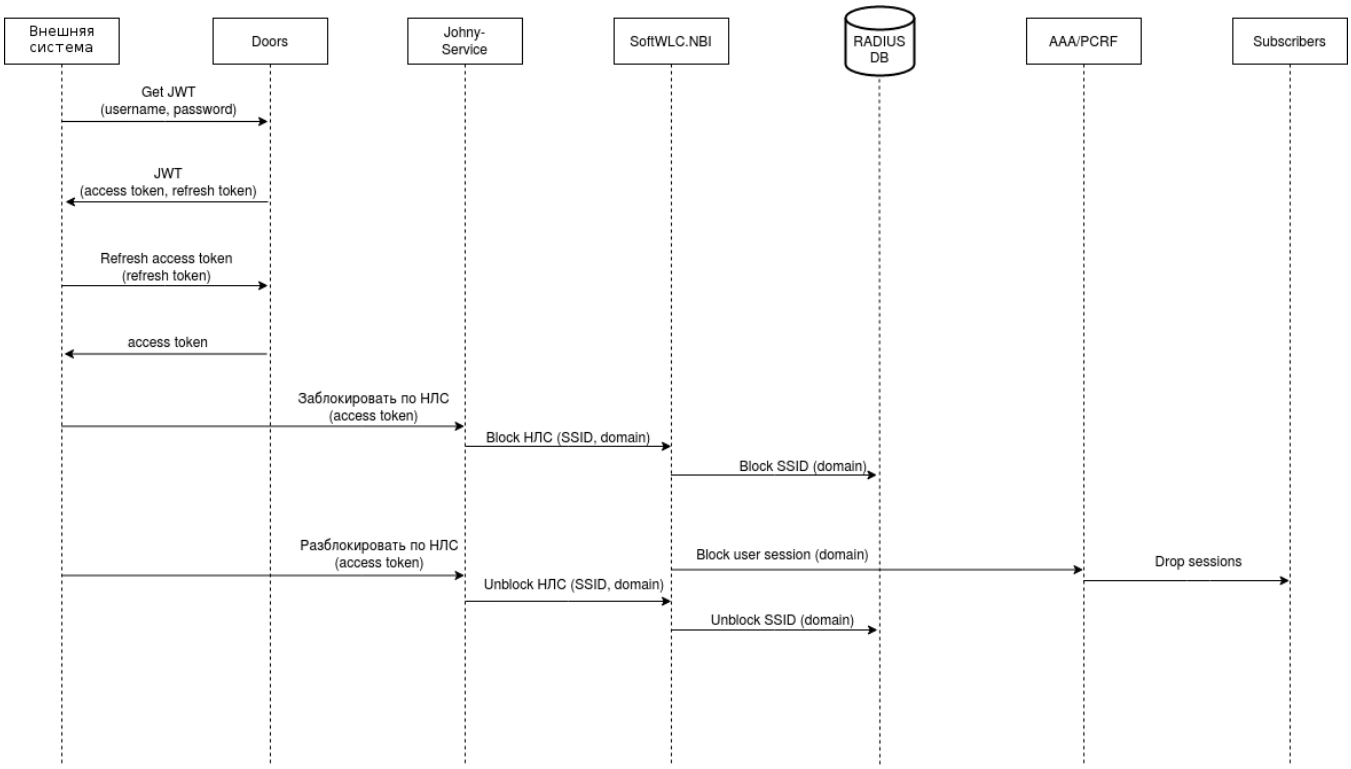
```
      maxLength: 15
      minLength: 7

    domain:
      type: string
      description: ' ( ),    .       1  63       / ,   : _  -.     235 .'
      example: "root.hello.world"
      maxLength: 235
      minLength: 1
      pattern: '(?=^.{1,235}$)(^((?!-|_|.*(__|--).*)[a-zA-Z0-9_-]{1,63}.)*((?!-|_|.*(__|--).*)[a-zA-Z0-9_-]
{1,63})$)'

    User:
      type: object
      properties:
        username:
          $ref: '#/components/schemas/username'

        domain:
          $ref: '#/components/schemas/domain'

    User_attrs:
      type: object
      properties:
        tariff_code:
          type: string
          description: '  .      .      /   64- .'
          maxLength: 64
          minLength: 1
          pattern: '[a-zA-Z0-9]\{1,64}'
        locked:
          type: boolean
          description: '  . true – , false – .'

    User_full:
      type: object
      allOf:
        - $ref: '#/components/schemas/User'
        - $ref: '#/components/schemas/User_attrs'
        - type: object

    Error:
      type: object
      properties:
        description:
          type: string
          description:  .

  requestBodies:
    # user without attrs (only username and domain)
    User:
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/User'
      description:    .

    # full user information, including username, domain, lock and tariff code
    User_full:
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/User_full'
      description:   .
      required: true

    # user attributes only (tariff code and lock value)
    User_attrs:
      content:
        application/json:
          schema:
```

```
            $ref: '#/components/schemas/User_attrs'
      description:   .
      required: true

  responses:
    200:
      description:  .       .
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/User_full'

    200_full:
      description:  .    .
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/User_full'

    400:
      description:     .
      content:
        'application/json':
          schema:
            $ref: '#/components/schemas/Error'

    403:
      description:
      content:
        'application/json':
          schema:
            $ref: '#/components/schemas/Error'

    404:
      description:     /     URL .

    500:
      description:  .
      content:
        'application/json':
          schema:
            $ref: '#/components/schemas/Error'



security:
  - JwtAuth: []

tags:
  - name: users
    description: API  .


paths:
  /api/users/new:
    post:
      tags:
        - users
      summary:
      requestBody:
        $ref: '#/components/requestBodies/User_full'

      responses:
        200:
          $ref: "#/components/responses/200_full"
        400:
          $ref: "#/components/responses/400"
        403:
          $ref: "#/components/responses/403"
        404:
          $ref: "#/components/responses/404"
```

```
/api/users/{username}@{domain}:
  parameters:
    - name: username
      in: path
      required: true
      schema:
        $ref: '#/components/schemas/username'

    - name: domain
      in: path
      required: true
      schema:
        $ref: '#/components/schemas/domain'


  put:
    tags:
      - users
    summary:   .
    requestBody:
      $ref: '#/components/requestBodies/User_attrs'

    responses:
      200:
        $ref: "#/components/responses/200_full"
      400:
        $ref: "#/components/responses/400"
      403:
        $ref: "#/components/responses/403"
      404:
        $ref: "#/components/responses/404"

  delete:
    tags:
      - users
    summary:  .  URL   username  .
    responses:
      200:
        $ref: "#/components/responses/200"
      400:
        $ref: "#/components/responses/400"
      403:
        $ref: "#/components/responses/403"
      404:
        $ref: "#/components/responses/404"
```

B2B

(B2B)



API Open API (B2B)

API

```
openapi: "3.0.1"
info:
  title: OSS Interaction Service API
  version: "1.0"

components:
  securitySchemes:
    JwtAuth:
      type: http
      scheme: bearer
      bearerFormat: JWT

  schemas:
    account_id:
      type: integer
      description:     ().
      maxLength: 12
      minLength: 12

    locked:
      type: boolean
      description:     ().

    account_properties:
      type: object
      properties:
        locked:
          $ref: '#/components/schemas/locked'

    Response:
      type: object
      properties:
```

```
        locked:
          $ref: '#/components/schemas/locked'

        account_id:
          $ref: '#/components/schemas/account_id'

    Error:
      type: object
      properties:
        description:
          type: string
          description:  .

  requestBodies:
    # user without attrs (only username and domain)
    account_properties:
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/account_properties'
      description:  .


  responses:

    200:
      description:  .   .
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/Response'

    400:
      description:   .
      content:
        'application/json':
          schema:
            $ref: '#/components/schemas/Error'

    403:
      description:  .
      content:
        'application/json':
          schema:
            $ref: '#/components/schemas/Error'

    404:
      description:    .

    500:
      description:  .
      content:
        'application/json':
          schema:
            $ref: '#/components/schemas/Error'



security:
  - JwtAuth: []

tags:
  - name: clients
    description: API   .

paths:
  /api/clients/{account_id}:
    put:
      tags:
        - clients
      summary:      .
```

```
    parameters:
      - in: path
        name: account_id
        required: true
        schema:
          $ref: '#/components/schemas/account_id'

    requestBody:
      $ref: '#/components/requestBodies/account_properties'

    responses:
      200:
        $ref: "#/components/responses/200"
      400:
        $ref: "#/components/responses/400"
      403:
        $ref: "#/components/responses/403"
      404:
        $ref: "#/components/responses/404"
```

---

docker-. .env docker-compose.yml

docker-compose.yml

```
version: "3"
services:
  eltex-johnny:
    container_name: eltex-johnny
    image: hub.eltex-co.ru/softwlc/eltex-johnny:1.18-95
    ports:
      - 9100:${JOHNNY_PORT}
    environment:
      - server.port=${JOHNNY_PORT}
      - pcrf.host=${JOHNNY_PCRF_HOST}
      - pcrf.port=${JOHNNY_PCRF_PORT}
      - pcrf.timeout=${JOHNNY_PCRF_TIMEOUT}
      - mercury.host=${JOHNNY_MERCURY_HOST}
      - mercury.port=${JOHNNY_MERCURY_PORT}
      - nbi.host=${JOHNNY_NBI_HOST}
      - nbi.port=${JOHNNY_NBI_PORT}
      - nbi.timeout=${JOHNNY_NBI_TIMEOUT}
      - nbi.login=${JOHNNY_NBI_LOGIN}
      - nbi.password=${JOHNNY_NBI_PASSWORD}
      - portal.scheme=${JOHNNY_PORTAL_SCHEME}
      - portal.host=${JOHNNY_PORTAL_HOST}
      - portal.port=${JOHNNY_PORTAL_PORT}
      - doors.host=${JOHNNY_DOORS_HOST}
      - doors.port=${JOHNNY_DOORS_PORT}
      - doors.path=${JOHNNY_DOORS_PATH}
      - doors.username=${JOHNNY_DOORS_USERNAME}
      - doors.password=${JOHNNY_DOORS_PASSWORD}
      - http.connectionTimeout=${JOHNNY_HTTP_CONNECTION_TIMEOUT}
      - http.connectionTotal=${JOHNNY_HTTP_CONNECTION_TOTAL}
      - validation.public_key=${JOHNNY_PUBLIC_KEY}
      - LOG_LEVEL=DEBUG
    volumes:
      - /etc/eltex-doors/keys/public.pem:${JOHNNY_PUBLIC_KEY}:ro
```

, , Graylog.

.env

```
JOHNNY_PORT=9100

JOHNNY_PCRF_HOST=<IP-address>
JOHNNY_PCRF_PORT=7070
JOHNNY_PCRF_TIMEOUT=10s

JOHNNY_MERCURY_HOST=<IP-address>
JOHNNY_MERCURY_PORT=6565

JOHNNY_NBI_HOST=<IP-address>
JOHNNY_NBI_PORT=8080
JOHNNY_NBI_TIMEOUT=80s
JOHNNY_NBI_LOGIN=admin
JOHNNY_NBI_PASSWORD=password

JOHNNY_PORTAL_SCHEME=http
JOHNNY_PORTAL_HOST=<IP-address>
JOHNNY_PORTAL_PORT=9000

JOHNNY_DOORS_HOST=<IP-address>
JOHNNY_DOORS_PORT=9097
JOHNNY_DOORS_PATH=/api
JOHNNY_DOORS_USERNAME=user
JOHNNY_DOORS_PASSWORD=password

JOHNNY_HTTP_CONNECTION_TIMEOUT=20s
JOHNNY_HTTP_CONNECTION_TOTAL=100

JOHNNY_PUBLIC_KEY=/etc/eltex-doors/keys/public.pem

LOG_LEVEL=DEBUG
```

.env docler-compose.yml . :

```
docker-compose up
```

- **JOHNNY_PORT** - , ;
- **JAVA_INIT_HEAP** - *JVM* `-Xms;`
- **JAVA_MAX_HEAP** - *JVM* `-Xmx;`
- **JAVA_OPTS** – *JVM* ( ).

eltex-pcrf

- **JOHNNY_PCRF_HOST** – eltex-pcrf;
- **JOHNNY_PCRF_PORT** – eltex-pcrf;
- **JOHNNY_PCRF_TIMEOUT** – pcrf ;
- **pcrf.pool.min** – idle pcrf ;
- **pcrf.pool.max** – pcrf ;
- **pcrf.pool.waitTimeout** – pcrf .

eltex-mercury

- **JOHNNY_MERCURY_HOST** – eltex-mercury;
- **JOHNNY_MERCURY_PORT** – eltex-mercury;
- **mercury.pool.min** – idle mercury ;
- **mercury.pool.max** – mercury ;
- **mercury.pool.waitTimeout** – mercury .

eltex-nbi

- **JOHNNY_NBI_HOST** – eltex-nbi;

- **JOHNNY_NBI_PORT** – eltex-nbi;
- **JOHNNY_NBI_TIMEOUT** – nbi ;
- **JOHNNY_NBI_LOGIN** – eltex-nbi;
- **JOHNNY_NBI_PASSWORD** – eltex-nbi;
- **nbi.pool.min** – idle nbi ;
- **nbi.pool.max** – nbi ;
- **nbi.pool.waitTimeout** – nbi .

eltex-portal

- **JOHNNY_PORTAL_SCHEME** – eltex-portal;
- **JOHNNY_PORTAL_HOST** – eltex-portal;
- **JOHNNY_PORTAL_PORT** – eltex-portal.

eltex-doors

- **JOHNNY_DOORS_HOST** – eltex-doors;
- **JOHNNY_DOORS_PORT** – eltex-doors;
- **JOHNNY_DOORS_PATH** – API eltex-doors ;
- **JOHNNY_DOORS_USERNAME** – eltex-doors;
- **JOHNNY_DOORS_PASSWORD** – eltex-doors.

HTTP

- **JOHNNY_HTTP_CONNECTION_TIMEOUT** – *http* ;
- **JOHNNY_HTTP_CONNECTION_TOTAL** – *http* ;

JWT

- **JOHNNY_PUBLIC_KEY** – *JWT (public.pem, eltex-doors, /etc/eltex-doors/keys/public.pem).*