**ИНСТРУКЦИЯ ПО СКАЧИВАНИЮ И УСТАНОВКЕ СИСТЕМЫ ВЫЗОВА ПОСЕТИТЕЛЕЙ**

**НА 14 ЛИСТАХ**

[Подготовка сервера 3](#_Toc1)

[Установка Docker CE и вспомогательных утилит 3](#_Toc2)

[Подготовка конфигурации 4](#_Toc3)

[Настройка конфигурации http сервера nginx 4](#_Toc4)

[Настройка компонент системы 6](#_Toc5)

[Запуск системы 14](#_Toc6)

# **Подготовка сервера**

## **Установка Docker CE и вспомогательных утилит**

Программное обеспечение (далее ПО) системы работает с использованием технологий контейнеризации Docker. На сервер необходимо установить Docker версии не ниже 19.03.6 в соответствии с официальной инструкцией с сайта <https://docs.docker.com/engine/install>

Для использования команд Docker без sudo добавьте вашего пользователя в группу Docker:

$ sudo usermod -aG docker $USER

Завершите ssh сессию на сервере для применения изменений. При повторном входе на сервер проверьте возможность использования команд docker без sudo.

$ docker ps

Вывод команды должен быть следующим:

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

Для настройки конфигурации всех контейнеров используется единый конфигурационный YAML файл и утилита Docker Compose для запуска служб из этой конфигурации.

Необходимо установить Docker compose версии не ниже 1.17.1 по инструкции с сайта https://docs.docker.com/compose/install

# **Подготовка конфигурации**

## **Настройка конфигурации http сервера nginx**

ПО Nginx запускается внутри Docker контейнера. На этапе настройки необходимо создать следующие директории:

* **/nginx/config/default.conf** – директория для конфигурационного файла admin frontend
* **/nginx/config/tv.conf**

$ sudo mkdir /nginx/config

Создайте конфигурационный файл al.conf в специальной директории и поместите в него приложенный ниже текст.

$ sudo vim /nginx/config/default.conf

**default.conf**

*server {*

*listen 80;*

*server\_name \_\_\*;*

location /session-service {

proxy\_pass http://al-suo.snap.dev.local:8072;

}

location /showcase-service {

proxy\_pass http://al-suo.snap.dev.local:8072;

}

location /televisions-near-the-courtroom {

rewrite /televisions-near-the-courtroom /$1 last;

}

location /televisions-in-the-general-courtroom {

rewrite /televisions-in-the-general-courtroom /$1 last;

}

location /tablets {

rewrite /tablets /$1 last;

}

location /halls {

rewrite /halls /$1 last;

}

location /admin {

alias /usr/share/nginx/html/admin/;

proxy\_redirect off;

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

proxy\_set\_header X-Forwarded-Host $server\_name;

}

location /admin/showcase-service/api {

proxy\_pass http://al-suo.snap.dev.local:8072/showcase-service/api;

}

location /tv {

alias /usr/share/nginx/html/tv/;

proxy\_redirect off;

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

proxy\_set\_header X-Forwarded-Host $server\_name;

}

location /tv/showcase-service/api {

proxy\_pass http://al-suo.snap.dev.local:8072/showcase-service/api;

}

location /showcase-service/swagger-ui/ {

proxy\_pass http://al-suo.snap.dev.local:8072/showcase-service/swagger-ui/;

}

location ~\* /\. {

deny all;

}

}

sudo vim /nginx/config/tv.conf

***tv.conf***

*server {*

*listen 3030;*

*server\_name \_;*

*location / {*

*root /usr/share/nginx/html;*

*}*

*location /schedule-meetings {*

*rewrite /schedule-meetings /$1 last;*

*}*

*location /showcase-service {*

*proxy\_pass http://localhost:8072;*

*}*

*}*

## **Настройка компонент системы**

ПО портала запускается внутри Docker контейнеров. Для его работы используется 15 контейнеров:

1. keycloak– контейнер авторизации Keycloak
2. showcaseservice– контейнер-витрина
3. sessionservice – контейнер домена посещений
4. nsiservice – контейнер домена нормативно-справочной информации
5. gatewayserver – контейнер Spring Cloud
6. postgres— контейнер базы данных (далее БД)
7. hearingservice – контейнер домена посещений из внешнего источника данных
8. eurekaserver – контейнер Spring Cloud
9. configserver – контейнер Spring Cloud
10. integrationservice – контейнер сервиса внешней интеграции с внешней системой-поставщиком данных о посещениях за день
11. al-tv – контейнер с сайтом для информационной панели в общем зале
12. al-admin-portal-frontend – контейнер для сайта модуля управления роли Администратор (административная панель)
13. kafka – контейнер брокера очередей Apache Kafka
14. zookeeper – контейнер для Apache Kafka

Docker образы хранятся в облачной папке https://ncloud.avilex.ru/index.php/s/DWT5WZDKCsgkXX6. Для инициализации образов на виртуальной машины (далее ВМ) необходимо выполнить команду:

$ docker load --input <image\_name>.tar.gz

Полученный образ можно увидеть с помощью команды:

$ docker image ls

Создайте файл docker-compose.yml в директории **/docker\_stack** и поместите приложенный ниже текст.

$ sudo mkdir /docker\_stack

$ sudo vim /docker\_stack/docker-compose.yml

*docker-compose.yml*

version: '3.9'

services:

al-admin-portal-frontend

image: al/al-admin-portal-frontend

container\_name: al-admin-portal-frontend

volumes:

- "./nginx/config/default.conf:/etc/nginx/conf.d/al.conf:ro"

ports:

- "80:80"

restart: always

al-tv

image: al/al-tv

container\_name: al-tv

volumes:

- "./nginx/config/tv.conf:/etc/nginx/conf.d/tv.conf:ro"

ports:

- "3030:3030"

restart: always

configserver:

image: al/configserver

container\_name: configserver

environment:

SPRING\_PROFILES\_ACTIVE: native

volumes:

- ./config:/opt/config

ports:

- "8071:8071"

restart: always

eurekaserver:

image: al/eurekaserver

container\_name: eurekaserver

ports:

- "8070:8070"

environment:

SPRING\_PROFILES\_ACTIVE: dev

depends\_on:

- configserver

restart: always

gatewayserver:

image: al/gatewayserver

container\_name: gatewayserver

ports:

- "8072:8072"

environment:

SPRING\_PROFILES\_ACTIVE: dev

SERVER\_PORT: "8072"

CONFIGSERVER\_URI: "http://configserver:8071"

EUREKASERVER\_URI: "http://eurekaserver:8070/eureka/"

EUREKASERVER\_PORT: "8070"

CONFIGSERVER\_PORT: "8071"

SERVICE\_BINDING\_ROOT: /bindings

volumes:

- ./ssl/public:/bindings/ca-certificates

- ./ssl/keystore:/opt/keystore

depends\_on:

- keycloak

- configserver

- eurekaserver

restart: always

showcaseservice:

image: al/showcaseservice

container\_name: showcaseservice

environment:

SPRING\_PROFILES\_ACTIVE: dev

SERVICE\_BINDING\_ROOT: /bindings

volumes:

- ./ssl/public:/bindings/ca-certificates

depends\_on:

- postgres

- configserver

- kafka

- gatewayserver

- eurekaserver

ports:

- "8099:8099"

restart: always

hearingservice:

image: al/hearingservice

container\_name: hearingservice

environment:

SPRING\_PROFILES\_ACTIVE: dev

SERVICE\_BINDING\_ROOT: /bindings

volumes:

- ./ssl/public:/bindings/ca-certificates

depends\_on:

- postgres

- configserver

- kafka

- gatewayserver

- eurekaserver

ports:

- "8097:8097"

restart: always

nsiservice:

image: al/nsiservice

container\_name: nsiservice

environment:

SPRING\_PROFILES\_ACTIVE: dev

SERVICE\_BINDING\_ROOT: /bindings

volumes:

- ./ssl/public:/bindings/ca-certificates

depends\_on:

- postgres

- configserver

- kafka

- gatewayserver

- eurekaserver

ports:

- "8087:8087"

restart: always

sessionservice:

image: al/sessionservice

container\_name: sessionservice

environment:

SPRING\_PROFILES\_ACTIVE: dev

SERVICE\_BINDING\_ROOT: /bindings

volumes:

- ./ssl/public:/bindings/ca-certificates

depends\_on:

- postgres

- configserver

- kafka

- gatewayserver

- eurekaserver

ports:

- "8094:8094"

restart: always

postgres:

image: postgres:15.2

container\_name: postgres

user: root

ports:

- "5432:5432"

environment:

POSTGRES\_USER: "postgres"

POSTGRES\_PASSWORD: "postgres"

POSTGRES\_DB: "al"

volumes:

- ./data/postgresql/data:/var/lib/postgresql/data

- ./postgres/docker-entrypoint-initdb.d:/docker-entrypoint-initdb.d

command: postgres -c 'max\_connections=1000'

healthcheck:

test: [ "CMD-SHELL", "pg\_isready -U postgres" ]

interval: 10s

timeout: 5s

retries: 5

restart: unless-stopped

keycloak:

build:

dockerfile: ./keycloak/Dockerfile

container\_name: keycloak

environment:

KC\_HEALTH\_ENABLED: true

KC\_METRICS\_ENABLED: true

KC\_DB: postgres

KC\_DB\_URL: 'jdbc:postgresql://postgres/al'

KC\_DB\_USERNAME: kc\_user

KC\_DB\_PASSWORD: kc\_user

KC\_DB\_SCHEMA: keycloak

KEYCLOAK\_ADMIN: admin

KEYCLOAK\_ADMIN\_PASSWORD: admin

KC\_HTTPS\_PORT: 8480

KC\_HTTPS\_KEY\_STORE\_FILE: /opt/keycloak/cert/al.p12

KC\_HTTPS\_KEY\_STORE\_PASSWORD: password

KC\_HTTPS\_KEY\_STORE\_TYPE: PKCS12

KC\_HOSTNAME: al.avilex.ru

KC\_HOSTNAME\_STRICT: 'false'

HTTP\_ADDRESS\_FORWARDING: "true"

PROXY\_ADDRESS\_FORWARDING: 'true'

KC\_HTTP\_RELATIVE\_PATH: "/auth"

KC\_PROXY: 'edge'

KAFKA\_BOOTSTRAP\_SERVERS: kafka:29092

#KEYCLOAK\_FRONTEND\_URL: localhost/keycloak

volumes:

# - ./keycloak/providers/:/opt/keycloak/providers/

# - ./keycloak/theme.jar:/opt/keycloak/providers/

- ./keycloak/al-keycloak-theme/:/opt/keycloak/themes/al-keycloak-theme/

- ./ssl/keystore/:/opt/keycloak/cert/

# - ./keycloak/role-listener-keycloak-extension-jar-with-dependencies.jar:/opt/keycloak/providers/

ports:

- "8480:8480"

depends\_on:

- postgres

command:

# - start-dev

- start

zookeeper:

image: confluentinc/cp-zookeeper:7.3.0

hostname: zookeeper

container\_name: zookeeper

user: root

ports:

- "2181:2181"

volumes:

- ./data/zookeeper:/var/lib/zookeeper

environment:

ZOOKEEPER\_CLIENT\_PORT: 2181

healthcheck:

test: "echo stat | nc localhost $$ZOOKEEPER\_CLIENT\_PORT"

restart: unless-stopped

kafka:

image: confluentinc/cp-kafka:7.3.0

hostname: kafka

container\_name: kafka

user: root

depends\_on:

- zookeeper

ports:

- "29092:29092"

- "9092:9092"

- "9101:9101"

volumes:

- ./data/kafka/data:/var/lib/kafka/data

environment:

KAFKA\_BROKER\_ID: 1

KAFKA\_ZOOKEEPER\_CONNECT: 'zookeeper:2181'

KAFKA\_LISTENER\_SECURITY\_PROTOCOL\_MAP: PLAINTEXT:PLAINTEXT,PLAINTEXT\_HOST:PLAINTEXT

KAFKA\_ADVERTISED\_LISTENERS: PLAINTEXT://kafka:29092,PLAINTEXT\_HOST://127.0.0.1:9092

KAFKA\_LISTENERS: PLAINTEXT://0.0.0.0:29092,PLAINTEXT\_HOST://0.0.0.0:9092

KAFKA\_OFFSETS\_TOPIC\_REPLICATION\_FACTOR: 1

KAFKA\_TRANSACTION\_STATE\_LOG\_MIN\_ISR: 1

KAFKA\_TRANSACTION\_STATE\_LOG\_REPLICATION\_FACTOR: 1

KAFKA\_GROUP\_INITIAL\_REBALANCE\_DELAY\_MS: 0

KAFKA\_JMX\_PORT: 9101

KAFKA\_JMX\_HOSTNAME: kafka

KAFKA\_AUTO\_CREATE\_TOPICS\_ENABLE: 'true'

KAFKA\_JMX\_OPTS: -Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.jmxremote.ssl=false -Djava.rmi.server.hostname=kafka -Dcom.sun.management.jmxremote.rmi.port=1099

JMX\_PORT: 1099

restart: unless-stopped

# **Запуск системы**

Для запуска контейнеров необходимо в директории **/docker\_stack** и выполнить команду:

$ cd /docker\_stack

$ docker-compose up -d

Для проверки работоспособности запущенных сервисов используется следующая команда:

$ docker logs -ft <service\_name>

В случае неуспешного запуска в логе будут перечислены причины такого запуска контейнера.