Процедура установки СОНАТА PMS

RDM

Оглавление

1 Установка docker engine 3
2 Развертывание системы с помощью docker-compose: 4
3 Создание баз данных 7
4 Файл “hosts” - добавление записей 9

1 Установка docker engine

Если docker engine отсутствует, он должен быть установлен. Инструкция может быть найдена здесь: https://dev.to/bowmanjd/install-docker-on-windows-wsl-without-docker-desktop-34m9

2 Развертывание системы с помощью docker-compose:

Для установки создайте временный каталог, далее нам понадобится создать 2 файла:

`rabbitmq-with-plugin.dockerfile`

с содержимым:

```
ARG PLUGIN_VERSION=3.12.0
ARG BASE_VERSION=3.12
FROM ubuntu:20.04 AS builder
ARG PLUGIN_VERSION
RUN apt-get update && DEBIAN_FRONTEND=noninteractive apt-get install -y
curl
RUN mkdir -p /plugins && \
curl -fsSL \
-o "/plugins/rabbitmq_delayed_message_exchange-${PLUGIN_VERSION}.ez" \
https://github.com/rabbitmq/rabbitmq-delayed-message-
exchange/releases/download/v${PLUGIN_VERSION}/rabbitmq_delayed_message_exc
hange-${PLUGIN_VERSION}.ez
FROM rabbitmq:${BASE_VERSION}-management-alpine
ARG PLUGIN_VERSION
COPY --from=builder --chown=rabbitmq:rabbitmq \
/plugins/rabbitmq_delayed_message_exchange-${PLUGIN_VERSION}.ez \
$RABBITMQ_HOME/plugins/rabbitmq_delayed_message_exchange-
${PLUGIN_VERSION}.ez
RUN rabbitmq-plugins enable --offline rabbitmq_delayed_message_exchange
```

This is an alternative to the previous `docker run` commands. Download dockerfile (see previous section). In the same directory where you put the dockerfile create a new file named `tng4-compose.yml` with contents below and run command `docker-compose -f tng4-compose.yml up -d`:

и `tng4-compose.yml`

с содержимым:

```
version: "3.9"

services:
  postgres:
    image: postgres:12.3-alpine
    command: postgres -c 'max_connections=1000'
    container_name: postgres
    environment:
      POSTGRES_PASSWORD: "tng4"
      PGDATA: "/var/lib/postgresql/data/pgdata"
    volumes:
      - pg-data:/var/lib/postgresql/data
    ports:
      - 5432:5432
  rabbitmq:
    container_name: rabbitmq
    ports:
      - 5672:5672
      - 15672:15672
    build:
      dockerfile: rabbitmq-with-plugin.dockerfile
  elasticsearch:
    image: elasticsearch:7.17.4
    container_name: elasticsearch
    environment:
      - "discovery.type=single-node"
      - "ES_JAVA_OPTS=-Xms256m -Xmx256m"
    mem_limit: 512m
    ports:
      - 9200:9200
  redis:
    image: redis:latest
    container_name: redis
    ports:
      - 6379:6379
  cashiering:
    image: rdm-registry.sonatapms.ru/rdm-cashiering:dev
    container_name: cashiering
    ports:
      - "5007:5007"
    environment:
      spring.cloud.config.uri: http://host.docker.internal:8888
      spring.cloud.config.label: local
      spring.profiles.active: prod
      JAVA_TOOL_OPTIONS: -
agentlib:jdwp=transport=dt_socket,address=*:15007,server=y,suspend=n -
Xmx256m -Xms128m
      datasource_url:
jdbc:postgresql://host.docker.internal:5432/cashiering
      rabbit_host: host.docker.internal
  housekeeping:
    image: rdm-registry.sonatapms.ru/rdm-housekeeping:dev
    container_name: housekeeping
    ports:
      - "5008:5008"
    environment:
      spring.cloud.config.uri: http://host.docker.internal:8888
      spring.cloud.config.label: local
      spring.profiles.active: prod
      JAVA_TOOL_OPTIONS: -
agentlib:jdwp=transport=dt_socket,address=*:15008,server=y,suspend=n -
Xmx256m -Xms128m
      datasource_url:
jdbc:postgresql://host.docker.internal:5432/housekeeping
      rabbit_host: host.docker.internal
  api-rdm:
```

```
    image: rdm-registry.sonatapms.ru/rdm-api:dev
    container_name: rdm-api
    ports:
      - "5009:5009"
    extra_hosts:
      - "host.docker.internal:host-gateway"
    environment:
      spring.application.name: api-rdm
      spring.cloud.config.uri: http://host.docker.internal:8888
      spring.cloud.config.label: local
      spring.profiles.active: prod
      JAVA_TOOL_OPTIONS: -
agentlib:jdwp=transport=dt_socket,address=*:15009,server=y,suspend=n -
Xmx256m -Xms128m
      datasource_url: jdbc:postgresql://host.docker.internal:5432
      rabbit_host: host.docker.internal

volumes:
    pg-data:
```

Далее выполните команду `docker-compose -f tng4-compose.yml up -d`

3 Создание баз данных

Скрипт для создания пользователей и схем БД:

```
docker exec -u postgres db psql -c "CREATE USER tng4 WITH PASSWORD
'tng4';";
docker exec -u postgres db psql -c "CREATE USER report WITH PASSWORD
'report' login;";
docker exec -u postgres db psql -c "CREATE DATABASE audit WITH OWNER tng4
ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE dictionary WITH OWNER
tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE email WITH OWNER tng4
ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE employee WITH OWNER
tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE integration WITH OWNER
tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE integrationohip WITH
OWNER tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE \"integrationohip-ru\"
WITH OWNER tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE \"integration-sts\"
WITH OWNER tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE ohipevents WITH OWNER
tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE license WITH OWNER
tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE offer WITH OWNER tng4
ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE licman2 WITH OWNER
tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE \"order\" WITH OWNER
tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE organization WITH
OWNER tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE profile WITH OWNER
tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE \"profile-ru\" WITH
OWNER tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE rbac WITH OWNER tng4
ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE report WITH OWNER tng4
ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE \"report-storage\"
WITH OWNER tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE resource WITH OWNER
tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE scheduler WITH OWNER
tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE sso WITH OWNER tng4
ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE storage WITH OWNER
tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE availability WITH
OWNER tng4 ENCODING 'UTF-8';"

docker exec -u postgres db psql -c "CREATE DATABASE \"property-gw\" WITH
OWNER tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE cashiering WITH OWNER
tng4 ENCODING 'UTF-8';"
docker exec -u postgres db psql -c "CREATE DATABASE housekeeping WITH
OWNER tng4 ENCODING 'UTF-8';"
```

Перед запуском микросервиса выполните команду для соответствующей БД:

```
create extension if not exists pgcrypto;
```

- availability
- storage
- profile
- profile-ru
- profile-eu
- profile-us
- sso

4 Файл “hosts” - добавление записей

Добавьте в файл “hosts” следующие записи

```
#192.168.1.10 host.docker.internal (comment out this docker host alias,
you may have different IPs)
#192.168.1.10 gateway.docker.internal (and insert localhost mapping
instead - this is done for eureka)
127.0.0.1 host.docker.internal
127.0.0.1 gateway.docker.internal

127.0.0.1 db
127.0.0.1 rabbitmq
127.0.0.1 elasticsearch
127.0.0.1 redis

127.0.0.1 api
127.0.0.1 availability
127.0.0.1 audit
127.0.0.1 cashiering
127.0.0.1 housekeeping
127.0.0.1 api-rdm
127.0.0.1 config
127.0.0.1 dictionary
127.0.0.1 email
127.0.0.1 employee
127.0.0.1 gateway
127.0.0.1 image-storage
127.0.0.1 integration
127.0.0.1 integration-ohip
127.0.0.1 integration-ohip-events
127.0.0.1 integration-ohip-events-processor
127.0.0.1 integration-sts
127.0.0.1 integration-webhook
127.0.0.1 license
127.0.0.1 offer
127.0.0.1 order
127.0.0.1 organization
127.0.0.1 profile
127.0.0.1 profile-ru
127.0.0.1 rbac
127.0.0.1 registry
127.0.0.1 report
127.0.0.1 report-storage
127.0.0.1 resource
127.0.0.1 sso
127.0.0.1 storage
```