Установка локальной копии

RIGHTWAY

Exported on Mar 01, 2021

Table of Contents

1 Cron и supervisor в проекте

После развёртывания локальной копии проекта, по умолчанию в ней не установлен cron и supervisor.

Примечание от Михаила:

> *Крона или супервизора в сборке контейнеров нет*
>
> *Фоновые процессы запускам ручками*
>
> *Чтобы было понимание как это работает внутри*

Полезная информация:

```
###Конфиги nginx, fpm пулов, crond, supervisor для приложений

Хранятся в репозитории вместе с приложением в виде jinja темплейтов
Например,
```
```
loyalty-processing/config/fpm/loyalty-processing.stage.conf.j2
loyalty-processing/config/nginx/loyalty-processing.stage.conf.j2
loyalty-processing/config/cron/stage.j2
loyalty-processing/config/supervisor/loyalty-processing.stage.conf.j2
```
```
```

2 Базовая установка Linux (Debian, Ubuntu, Mint)

> *Статья дополняется*
>
> *Установка в linux на основе Ubuntu 20.04.1 LTS*

1. **Hosts**

Дописываем файл host(/etc/hosts):

127.0.0.1 loyalty.local

127.0.0.1 auth.local

127.0.0.1 customer-ui.local

127.0.0.1 external-api.local

127.0.0.1 internal-api.local

127.0.0.1 catalog.local

127.0.0.1 filestorage-api.local

127.0.0.1 processing.local

2.**Генерация ключей для git.cardsmile.ru**

Генерация RSA ключа:

```
$ ssh-keygen -t rsa -C "your_email@example.com"
```

Проверить сгенерированные ключи(приватный и публичный):

> *$ ls /home/<user>/.ssh*

Скопировать полученный публичный ключ на сайт:

> *$ cat /home/<user>/.ssh/id_rsa.pub*

http://git.cardsmile.ru:7990/projects → менеджер аккаунтов → SSH keys → Add key

3. **Правка файлов перед сборкой проекта.**

В папке: /deployment/apps-deployment/docker скопировать файл .env.dist и вставить сюда же с именем .env. Поправить пути до папок сервисов, до ключей, что сгенерированы для доступа к git.cardsmile.ru и указать USER_ID и GROUP_ID текующего хоста(Команда консоли ~$: id):

> *COMPOSE_PROJECT_NAME=rightway*
> *WEB_SERVER_IP=192.168.1.105*
> *DOCKER_SUBNET=192.168.1.0/24*
>
> *XDEBUG_REMOTE_HOST=docker.for.mac.localhost*
>
> *HOST_USER_ID=1002*
> *HOST_USERGROUP_ID=1002*
> *SSH_PRIVATE_KEY=/home/omni3/.ssh/id_rsa*
> *SSH_PUBLIC_KEY=/home/omni3/.ssh/id_rsa.pub*
>
> *LOYALTY_APP_PATH=/home/omni3/PhpstormProjects/norbit/loyalty-2-0*
> *CUSTOMER_UI_APP_PATH=/home/omni3/PhpstormProjects/norbit/loyalty-customer-ui*
> *AUTH_APP_PATH=/home/omni3/PhpstormProjects/norbit/rightway-auth*
> *FILESTORAGE_APP_PATH=/home/omni3/PhpstormProjects/norbit/rightway-filestorage-api*

```
EXTERNAL_API_APP_PATH=/home/omni3/PhpstormProjects/norbit/rightway-external-api
CATALOG_APP_PATH=/home/omni3/PhpstormProjects/norbit/rightway-catalog
PROCESSING_APP_PATH=/home/omni3/PhpstormProjects/norbit/loyalty-processing
FRONTOL_PROXY_APP_PATH=/home/omni3/PhpstormProjects/norbit/rightway-frontol-proxy
```

Открыть файл **/deployment/apps-deployment/docker/php/Dockerfile** Строки9-11 заменить на:

```
apt-get -t stretch-backports install -y librdkafka-dev && \
apt-get install -y php-pear php7.1-cli php7.1-fpm php7.1-xml php7.1-dev && \
pecl channel-update pecl.php.net && \
```

между 11 и 12 строкой добавить:

```
pecl install mongodb-1.5.3 && \
```

между и 53 - 54 строки:

```
librdkafka1 \
unzip
```

Открыть файл **/deployment/apps-deployment/docker/fluentd/Dockerfile** и 5-7 строки:

```
RUN fluent-gem install elasticsearch -v 7.5
RUN fluent-gem install elasticsearch-transport -v 7.5.0
RUN fluent-gem install fluent-plugin-elasticsearch -v 1.13.0 --no-rdoc --no-ri
```

4. **Сборка и запуск контейнеров.**

Установить **docker** и **docker-compose**.

Из папки **/deployment/apps-deployment/docker** выполнить команду сборки контейнеров и их поднятие:

```
~/norbit$ cd /deployment/apps-deployment/docker

~/norbit/deployment/apps-deployment/docker$ docker-compose build

~/norbit/deployment/apps-deployment/docker$ docker-compose up -d
```

5. **Утилита ansible**

Установить **ansible**.

Открыть файл: deployment/apps-deployment/ansible/common/variables.yaml и добавить переменные(**auth_database, username, password**), необходимые для доступа приложению loyalty-2-0 к mongo:

```
mongodb:
  dsn: "mongodb://mongo"
  host: "mongo"
  port: 27017
  auth_database: "admin"
  username: "user"
```

```
password: "passw0rd"
cardsmile_database: "cardsmile"
filestorage_database: "filestorage"
catalog_database: "catalog"
```

Открыть файл: deployment/apps-deployment/ansible/templates/customer_ui.config.j2, добавить:

```
gelf_host: {{ gelf.host }}
gelf_port: {{ gelf.port }}
gelf_chunk_size: {{ gelf.chunk_size }}
```

Открыть файл : deployment/apps-deployment/ansible/templates/loyalty.config.j2 и раскомментировать строки 45 и 46:

```
'username' => '%mongo_username%',
'password' => '%mongo_password%',
```

Тут же добавить значения переменных на строках 157-159:

```
'%mongo_username_catalog%' => '{{ mongodb.username }}',
'%mongo_password_catalog%' => '{{ mongodb.password }}',
'%mongo_auth_database_catalog%' => '{{ mongodb.auth_database }}',
```

Строки 209-210:

```
'%params.use_mongo_card_index%' => false,
'%params.elastic_search_host%' => '{{ elasticsearch.host }}',
```

Строки 233, 238 - 239(Номера строк взял из файл конфигурации loyalty-2-0 **parametrs.php.dist)**:

```
'%params.exponea_local_reports_path_card_events%' =>
'/tmp/exponea/card_events/reports',
'%params.elastic_search_host_catalog%' => '',
'%params.elastic_search_port_catalog%' => '',
```

Открыть файл deployment/apps-deployment/ansible/templates/processing.config.j2. 4 и 6 строки добавить версию postsql, в конце файла добавить переменные:

```
database_main_server_version: {{ postgres.main_version }}
database_personal_server_version: {{ postgres.personal_version }}
# Url до приложения internal-api
internal_api_uri: '{{ service.internal_api.service_url }}'

# Разрешение асинхронной обработки запросов applyPurchase
enable_async_apply_purchase: false

# Время жизни кэша в секундах для модуля настроек clp-settings
clp_settings_cache_life_time: 86400
clp_settings_prefix: 'clp'
```

Открыть файл /deployment/apps-deployment/ansible/templates/cheque-recognition.config.j2 7 и 9 строки заменить на:

> *APP_SECRET={{ service.cheque_recognition.app_secret }}*
>
> *MONGODB_DATABASE={{ service.cheque_recognition.db_name }}*

Открыть файл /deployment/apps-deployment/ansible/common/variables.yaml в раздел service/cheque_recognition добавить переменную в (~84 строка):

> *app_secret: "ThisTokenIsNotSoSecretChangeIt"*

Из папки docker запустить утилиту ansible:

> *../ansible/run_playbook_with_env.sh ./.env ../ansible/app_config.yaml*

6. **Composer install**

Для каждого из необходимых приложений необходимо:

- Выполнить:

docker exec -ti **container-name** bash или docker-compose exec **container-name** bash

- Перейти в корень директории с приложением

cd /home/apps/rightway/**app_name**

- Выполнить команду

su apps

composer install

- После установки выйти из приложения командой exit exit или CTRL+D дважды

Список приложений, для которых нужны вышеуказанные действия (на примере команды docker exec)

docker exec -ti rightway_auth_1 bash

docker exec -ti rightway_catalog_1 bash

docker exec -ti rightway_filestorage_1 bash

docker exec -ti rightway_customer_ui_1 bash

docker exec -ti rightway_external_api_1 bash

docker exec -ti rightway_processing_1 bash

docker exec -ti rightway_loyalty_1 bash

Вне пункта на "подумать":

1. Влить ветку по настройке локального проекта в мастер(проект deployment, ветка notask/yimaykin.easyDeplomentEnv.mark_I), изменения касаются только прав доступа и развертывания проекта локально;

2. Перейти на докер контейнеры, основанные на дистрибутиве alpine;
3. Уменьшить размер базы для среды разработки

3 Базовая установка Win10

> *Для развертывания окружения в Windows, требуется Windows 10 с обновлением до версии 2004, сборкой 19041*

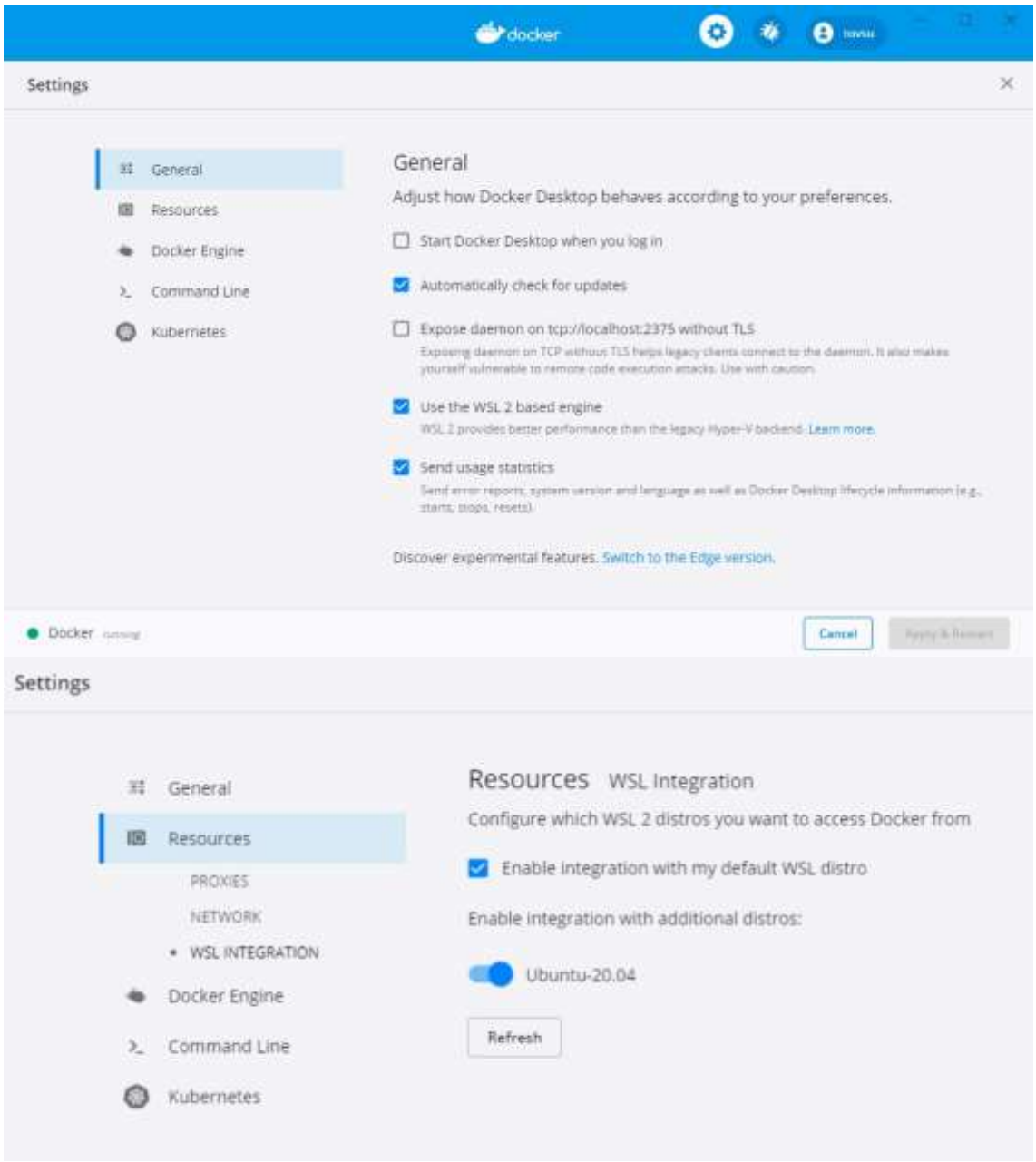
1. Установить подсистему Windows для Linux в Windows10 (WLS2) и дистрибутив Ubuntu

Инструкцию по установке находится по ссылке: https://docs.microsoft.com/ru-ru/windows/wsl/install-win10

---

2. Установить Git и Docker для Windows

Настройки Docker Desktop для Windows

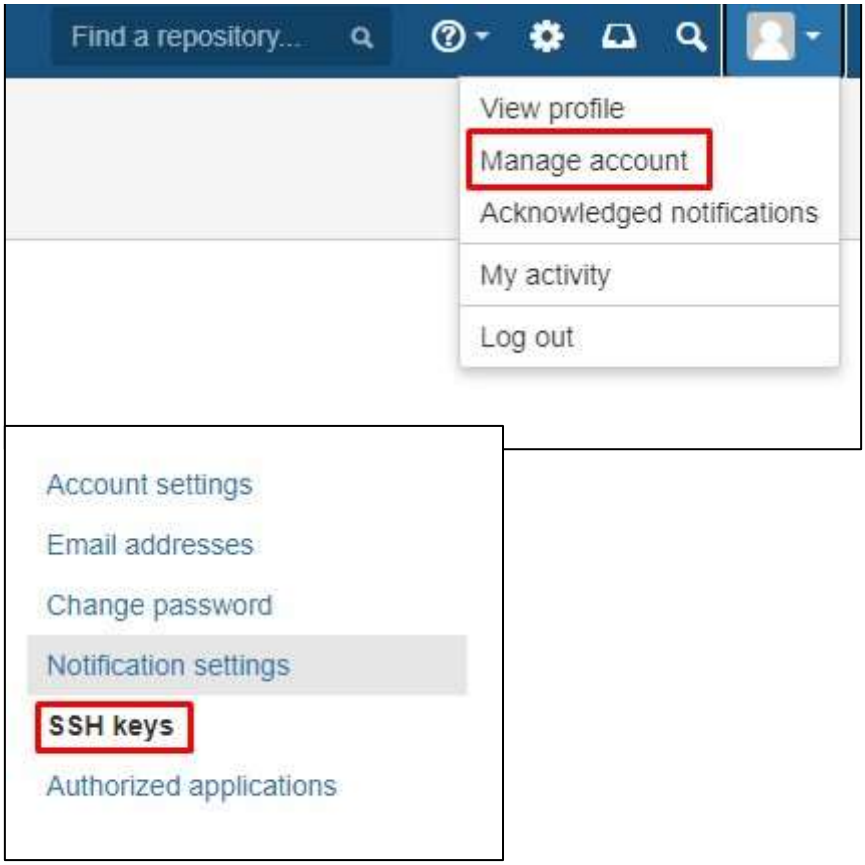


В дальнейшем Docker Desktop должен быть запущен в системе.

---

3. Сгенерировать и добавить свой ssh-ключ в профиль в Stash

Меню в правом верхнем углу – Manage account – SSH keys



---

4. Склонировать в Git дистрибутив:

При клонировании модулей в Git в Windows, нужно **предварительно** выполнить команду (запуск от имени администратора):

git config --system core.longpaths true

Репозитории для клонирования:

- loyalty-2-0
- loyalty-customer-ui
- rightway-auth
- rightway-filestorage-api
- rightway-external-api
- rightway-catalog
- loyalty-processing
- deployment
- rightway-cheque-recognition-api
- rightway-frontol-proxy

Найти каждую папку из этого списка ту:

http://git.cardsmile.ru:7990/projects/LOYAL

> *Удобно искать репозитории на веб-странице командой Ctrl + f*

Зайти в репозиторий, слева в верхнем углу кнопка Clone.

5.Скопировать настройки окружения

Файл .env.dist скопировать и копию переименовать в .env

Хранится в директории: deployment\apps-deployment\docker

---

6. Поправить переменные окружения в файле *.env*

Правка согласно расположению проектов (локальный путь до ssh-ключа, который добавлен в Git).

Если Docker запускается через Windows, то путь должен быть примерно такой:

C:\Users\user_name\...\loyalty-2-0

Если запускается через wls (linux), то:

/mnt/c/users/…/loyalty-2-0

Пример строк для обновления (путь может отличаться):

SSH_PRIVATE_KEY=/mnt/c/Users/user_name/git/Omni/.ssh/id_rsa
SSH_PUBLIC_KEY=/mnt/c/Users/ user_name/git/Omni/.ssh/id_rsa.pub

LOYALTY_APP_PATH=/mnt/c/Users/user_name/git/Omni/loyalty-2-0
CUSTOMER_UI_APP_PATH=/mnt/c/Users/user_name/git/Omni/loyalty-customer-ui
AUTH_APP_PATH=/mnt/c/Users/user_name/git/Omni/rightway-auth
FILESTORAGE_APP_PATH=/mnt/c/Users/user_name/git/Omni/rightway-filestorage-api
EXTERNAL_API_APP_PATH=/mnt/c/Users/ user_name/git/Omni/rightway-external-api
CATALOG_APP_PATH=/mnt/c/Users/user_name/git/Omni/rightway-catalog
PROCESSING_APP_PATH=/mnt/c/Users/user_name/git/Omni/loyalty-processing

---

7. Изменить в \deployment\apps-deployment\docker\config\ssh-config строчку для подключения к Stash по SSH.

Содержимое файла должно быть таким:

Host git.cardsmile.ru

IdentityFile /home/apps/.ssh/

---

8. В файл apps-deployment/docker/fluentd/Dockerfile нужно добавить после 3-й строки:

RUN fluent-gem install elasticsearch -v 7.5

RUN fluent-gem install elasticsearch-transport -v 7.5.0

RUN fluent-gem install fluent-plugin-elasticsearch -v 1.13.0 --no-rdoc --no-ri

---

9. В Ubuntu нужно установить docker, docker-compose, ansible

sudo apt-get update

sudo apt install docker

sudo apt install docker-compose

sudo apt install ansible

---

10. Поднять контейнеры в docker

Из папки \deployment\apps-deployment\docker\ выполнить команду

docker-compose up --build -d

Потом их запускать можно будет просто командой:

docker-compose up -d

---

11. Изменить формат строки файлов для ansible

Открыть в Notepad++ файлы:

apps-deployment/ansible/run_playbook_with_env.sh

apps-deployment/ansible/app_config.yaml

apps-deployment/docker/.env

В меню выбрать Edit(Правка)->EOA Conversion(Формат конца строк)->Unix

---

12. В Ubuntu, находясь в папке /deployment/apps-deployment/docker выполнить:

../ansible/run_playbook_with_env.sh ./.env ../ansible/app_config.yaml

---

13. Установить зависимости Symfony

Для каждого из необходимых приложений необходимо:

- Выполнить:

docker exec -ti **container-name** bash или docker-compose exec **container-name** bash

- Авторизоваться под apps

su apps

- Перейти в корень директории с приложением

cd /home/apps/rightway/**app_name**

- Выполнить команду

composer install

- После установки выйти из приложения командой exit

При установке пакетов, если будет спрашивать, все параметры указываем стандартные.

При установке, может быть указано о необходимости установить unzip. В этом случае можно прервать операцию (ctrl + x), выполнить:

sudo apt install unzip

Затем повторить composer install .

Список приложений, для которых нужны вышеуказанные действия (на примере команды docker exec)

docker exec -ti rightway_auth_1 bash

docker exec -ti rightway_catalog_1 bash

docker exec -ti rightway_filestorage_1 bash

docker exec -ti rightway_customer_ui_1 bash

docker exec -ti rightway_external_api_1 bash

```
docker exec -ti rightway_processing_1 bash
docker exec -ti rightway_loyalty_1 bash
docker exec -ti rightway_cheque_recognition_1 bash
docker exec -ti rightway_frontol_proxy_1 bash
```

---

14. Находясь в папке docker выполнить:

```
ansible-playbook ../ansible/db_deploy.yaml
```

После выполнения всех задач в файле ansible/dev_creds.txt сохранятся все необходимые учетные записи с паролями. Пример содержимого файла:

```
NOTICE: Создано 1 брендов
NOTICE: apiProfileKey = apiPKdeer4_thgh1Lg5fQ1P,
apiContactCenterKey = apiCC4_thghjgh1Lg5fQ1P,
processingKey = processingKey4_thgh1Lg5fQ1P,
user Password = 4_thgh1L,
seller Login = sony_seller
operator Login = sony_operator
admin Login = sony_admin
root Login = sony_root
```

Пароли в будущем вводиться без запятой в конце, она используется в файле просто как перечисление.

---

15. Добавить запись в файл hosts на локальной машине

Необходимо добавить строку в файл hosts на текущем компьютере адрес:

```
127.0.0.1 loyalty.local
127.0.0.1 auth.local
127.0.0.1 customer-ui.local
127.0.0.1 external-api.local
127.0.0.1 internal-api.local
127.0.0.1 catalog.local
127.0.0.1 filestorage-api.local
127.0.0.1 processing.local
```

Путь к файлу hosts

C:\windows\system32\drivers\etc**hosts**

---

16. Добавить папку runtime по адресу loyalty-2-0\applications\web

docker exec -ti rightway_loyalty_1 bash

cd rightway/loyalty/applications/web

mkdir runtime

---

17. Изменить файл настроек приложения loyalty-processing

В файле \loyalty-processing\config\packages\parameters.yml заменить 3-ю и 4-ю строчки:

database_main_dsn: pgsql://postgres:root@postgres:5432/cardsmile
database_main_server_version: 9.5
database_personal_dsn: pgsql://postgres:root@postgres_personal:5432/cardsmile_personal
database_personal_server_version: 9.3

И добавить в конце файла:

Время жизни кэша в секундах для модуля настроек clp-settings
clp_settings_cache_life_time: 86400
clp_settings_prefix: 'clp'

---

18. Добавить пропущенные таблицы

Из папки \deployment\apps-deployment\docker\ выполнить команды:

docker exec -ti rightway_loyalty_1 bash

cd /home/apps/rightway/loyalty

php ./applications/common/yiic.php CreateTablesForExistingBrands

---

19. Исправить - на _ в файле docker-compose.yaml

В файле: deployment\apps-deployment\docker\docker-compose.yaml

На строке 172 прописать customer_ui.local за место customer-ui.local

---

20. Работу системы можно проверить в браузере.

Локальный адрес платформы: loyalty.local

Доступы указаны в файле \deployment\apps-deployment\ansible\dev_creds.txt

4 Возможные ошибки

1. При выполнении команды docker-compose up --build -d

Ошибка:

ERROR: Error installing fluent-plugin-elasticsearch:

elasticsearch-transport requires Ruby version >= 2.4.

Решение:

Нужно в файл apps-deployment/docker/fluentd/Dockerfile после 3й строки вставить эти строки:

RUN fluent-gem install elasticsearch -v 7.5
RUN fluent-gem install elasticsearch-transport -v 7.5.0
RUN fluent-gem install fluent-plugin-elasticsearch -v 1.13.0 --no-rdoc --no-ri

2. При выполнении команды docker-compose up --build -d

Ошибка:

Couldn't connect to Docker daemon at http+docker://localhost - is it running?

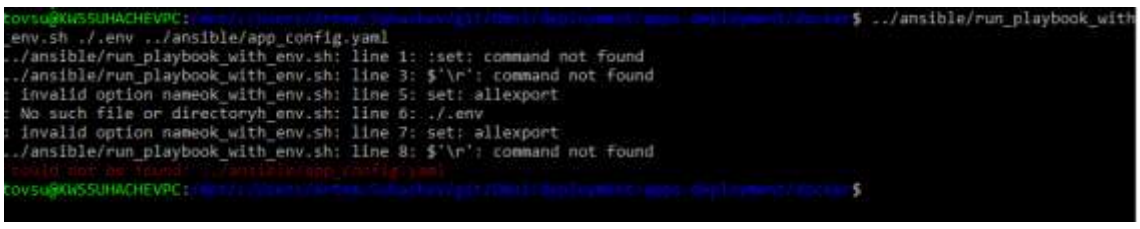
Решение:

Запустите docker desktop в windows

Проверить установлен ли wsl 2 в windows. И правильно ли он работает. В PowerShell запустить команду wsl -l -v Должен выдать список приложений и версии

3. При выполнении команды ../ansible/run_playbook_with_env.sh ./.env ../ansible/app_config.yaml

Ошибка:

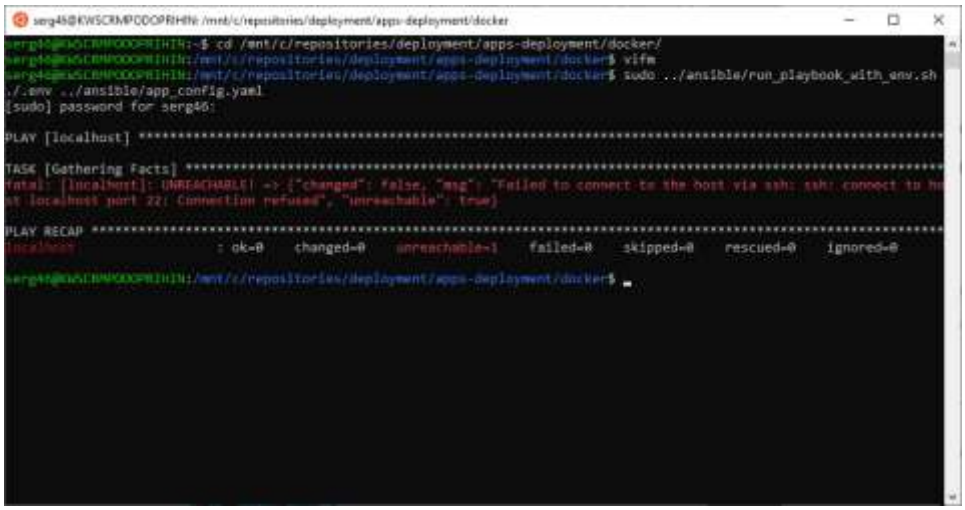


Решение:

- Проверить установлен ли wsl 2 в windows. И правильно ли он работает. В PowerShell запустить команду wsl -l -v Должен выдать список приложений и версии
- Проверить установлен ли ansible

4. При выполнении команды ../ansible/run_playbook_with_env.sh ./.env ../ansible/app_config.yaml

Ошибка:



Описание:

Неправильные настройки ansible.

Решение:

Проверить установлен ли wsl 2 в windows. И правильно ли он работает. В PowerPoint запустить команду wsl -l -v Должен выдать список приложений и версии

В файле /etc/ansible/ansible.cfg должно быть всё закомментировано.

5. При выполнении команды ../ansible/run_playbook_with_env.sh ./.env ../ansible/app_config.yaml

Ошибка:

/bin/sh^M: bad interpreter: No such file or directory

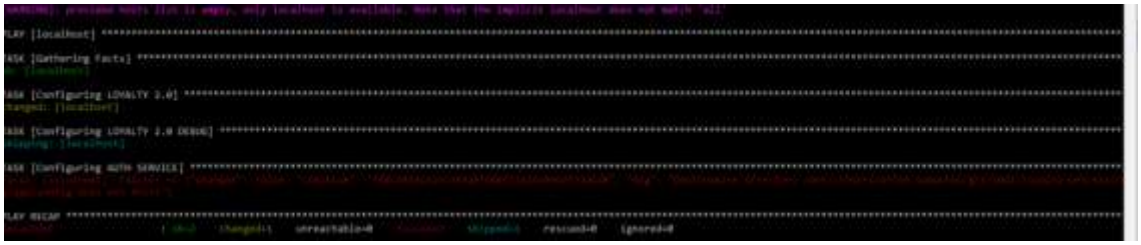
Решение:

Открыть файл apps-deployment/ansible/run_playbook_with_env.sh в Notepad++.

В меню выбрать Edit(Правка)->EOA Conversion(Формат конца строк)→Unix

6. При выполнении команды ../ansible/run_playbook_with_env.sh ./.env ../ansible/app_config.yaml

Ошибка:



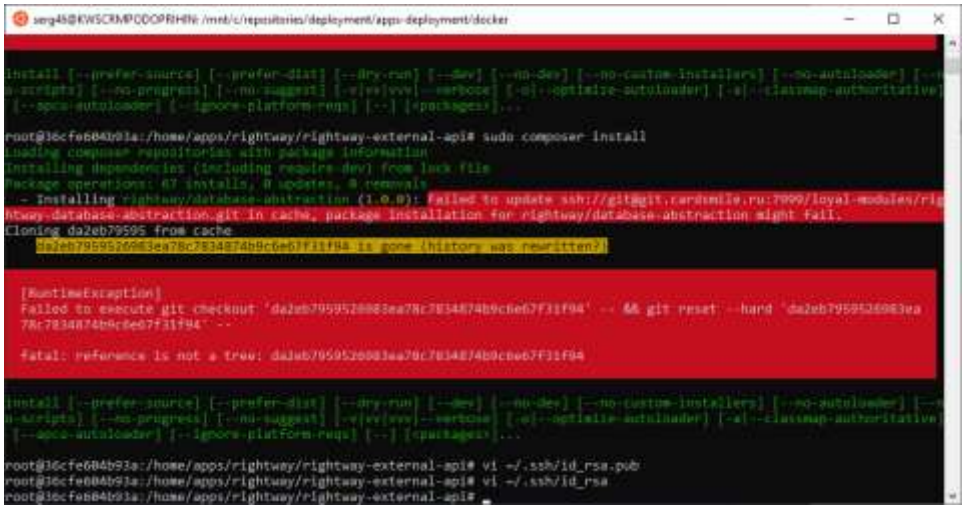
Решение:

Проверить правильно ли введены пути в файле deployment\apps-deployment\docker\.env

7. При выполнении команды composer install

Ошибка:

или



Описание:

Не находит файл ssh ключа, или ssh ключ не добавлен в Stash.

Решение:

Нужно скопировать файл id_rsa в ручную:

1. Внутри контейнера создаём папку mkdir /root/.ssh

2. Узнаём код контейнера, набрав docker ps находясь в папке deployment\apps-deployment\docker.

3. Выполняем команду docker cp {путь до файла ключа} {id контейнера}:root/.ssh/id_rsa

8. При выполнении команды composer install

Ошибка:

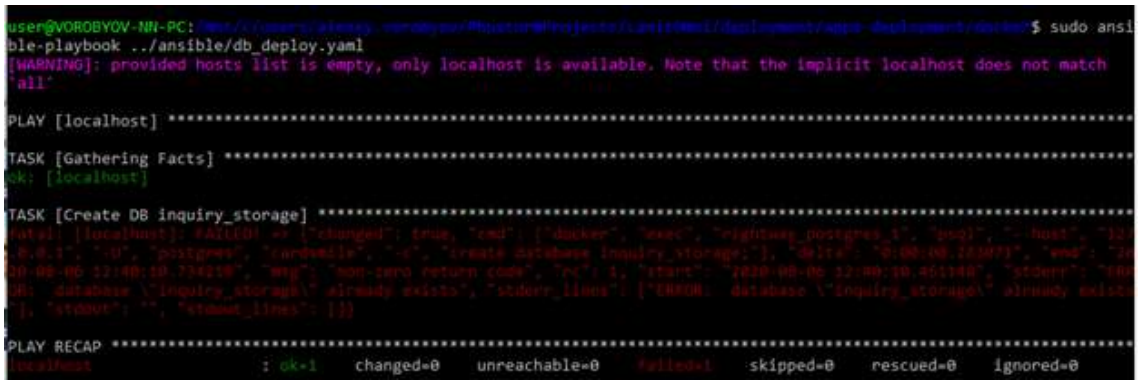
Решение:

В файле \loyalty-processing\config\packages\parameters.yml заменить 3-ю и 4-ю строчки:

database_main_dsn: pgsql://postgres:root@postgres:5432/cardsmile
database_main_server_version: 9.5
database_personal_dsn:
pgsql://postgres:root@postgres_personal:5432/cardsmile_personal
database_personal_server_version: 9.3

9. При выполнении команды ansible-playbook ../ansible/db_deploy.yaml

Ошибка:



Описание:

Не может создать таблицу, потому что она уже создана.

Решение:

1. Из папки deployment\apps-deployment\docker переходим в контейнер docker exec -ti rightway_postgres_1 bash

2. Напираем в консоли:

psql

DROP DATABASE ‘имя базы данных’

В данном случае ругается на базу “inquiry_storage”. Командой /list можно проверить наличие базы данных

10. При переходе в браузере на адрес 127.0.0.1

Ошибка:

Решение:

1. Проверить указаны ли в файле hosts адреса сайта:

   127.0.0.1 loyalty.local

2. Переходить по адресу http://loyalty.local/

11. При выполнении команды docker exec -ti rightway_loyalty_1 bash

Ошибка:

Внутри контейнера нет папок.

Решение:

1. Выйти из контейнера: exit
2. Остановить докер: docker-compose stop
3. Запустить докер: docker-compose -up -d

12. При создании новой акции

Ошибка:

Решение:

Из папки \deployment\apps-deployment\docker\ выполнить команды:

```
docker exec -ti rightway_loyalty_1 bash
cd /home/apps/rightway/loyalty
php ./applications/common/yiic.php CreateTablesForExistingBrands
```

13. При выполнении запросов к processing

Ошибка:

Решение:

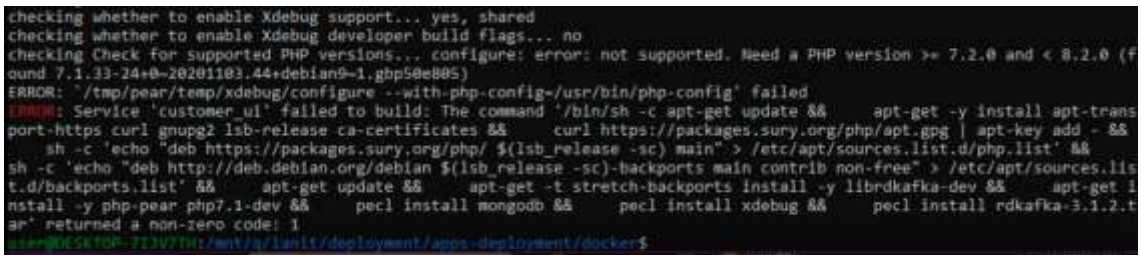
Нужно проверить все адреса в файле loyalty-processing/config/packages/parameters.yml

Например где адрес http://auth/ нужно заменить на http://auth.local/

Файл для примера, как должно выглядеть parameters.yml можно его скачать и заменить

14. При выполнении команды docker-compose build

Ошибка:



Решение:

В файле deployment/apps-deployment/docker/php/Dockerfile ~12 заменить на:

pecl install xdebug-2.9.8 && \

15. При авторизации ошибка timeout Curl

Ошибка:

Решение:

В файле loyalty-2-0\vendor\zendframework\zend-http\src\Client\Adapter\Curl.php 225 и 233 строке вставить значения больше 1000. Я ставил 2000 и 3000.

5 Индексация Elasticsearch

5.1 Документация по индексации Построение индекса карт

При выполнении ниже указанных действий будет происходить индексация таблицы card и customer в PostgreSQL.

Если брать тестовую базу, то там есть бренд с company_id = 60, у которого ~2 200 000 записей в таблице card. Соответственно он будет долго индексироваться. У меня ушло на него порядка 6-7 часов.

Если у карты будет установлен status = 2, то тогда индексация пропустит эту карту. Соответственно, если время вам важно, нужно либо установить у этих карт статус 2. Либо как-то не включать этот бренд в индексацию (как не включать, пока не известно 11.09.2020)

1. Для начала переходим в контейнер с Loyalty, все команды запускать в нём

```
docker exec -ti rightway_loyalty_1 bash
```

2. Авторизируемся под пользователем apps

```
su apps
```

3. Переходим в папку с модулем.

```
cd rightway/loyalty
```

4. Для проверки работы, откроем RebbitMQ.

переходим по ссылке:
http://localhost:15672/#/queues

5. Так же для проверки можно открыть Kibana с смотреть логи rightway-logs

http://localhost:5601/app/kibana#/discover

6. Если не запущены консьюмеры индекса, то нужно запустить.

Нам нужен вот такой список консьюмеров:

BatchIndexBuilder.Card	classic	D	idle	0
BatchIndexBuilder.Card.Elastic	classic	D	idle	0
IndexBuilder.Card	classic	D DLX	idle	0
IndexBuilder.Card.Elastic	classic	D DLX	idle	0
IndexBuilder.Online	classic	D DLX	idle	0
IndexBuilder.Online.Elastic	classic	D DLX	idle	0
IndexBuilder.delayed	classic	D TTL DLX	idle	0

Если их нет, то запускаем по очереди:

Далее запускаем все команды по очереди. Они запустят консьюмер, и будут ждать сообщений в очереди. Но поскольку сообщений нет, их нужно будет останавливать нажатием Ctrl+C.

Вы уведите как в RebiitMQ они будут появляться в списке, после этого их можно останавливать.

```
php /home/apps/rightway/loyalty/applications/common/yiic rabbitmq_consumer --name=IndexBuilder.Card --messages=1
```

```
php /home/apps/rightway/loyalty/applications/common/yiic rabbitmq_consumer --name=IndexBuilder.Card.Elastic --messages=1
```

```
php /home/apps/rightway/loyalty/applications/common/yiic rabbitmq_consumer --name=IndexBuilder.Online --messages=1
```

```
php /home/apps/rightway/loyalty/applications/common/yiic rabbitmq_consumer --name=IndexBuilder.Online.Elastic --messages=1
```

```
php /home/apps/rightway/loyalty/applications/common/yiic rabbitmq_consumer --name=batch_card_index_builder --messages=1
```

```
php /home/apps/rightway/loyalty/applications/common/yiic rabbitmq_consumer --name=batch_card_index_elastic_builder --messages=1
```

После запуска этих консьюмеров, можно будет сравнить их список, со скрином ранее. Должны быть все.

7. Если все консьюмеры на месте, проверяем конфиг parameters.php

Адрес файла loyalty-2-0/applications/parameters.php

Нам нужны строчки: 39,40,41,45,46.

```
'%mongo_auth_database%' => 'admin',
'%mongo_username%' => 'user',
'%mongo_password%' => 'passw0rd',

'username' => '%mongo_username%',
'password' => '%mongo_password%'
```

Там должна быть заполнена информация для авторизации в БД монго. Все комментарии должны быть сняты.

Далее нужны строки: 157, 158, 159, 162, 163.

```
'%mongo_auth_database_catalog%' => 'admin',
'%mongo_username_catalog%' => 'user',
'%mongo_password_catalog%' => 'passw0rd',

'username' => '%mongo_username_catalog%',
'password' => '%mongo_password_catalog%',
```

Там должна быть заполнена информация для авторизации в БД монго. Все комментарии должны быть сняты.

Далее нужны строки: 209, 210.

```
'%params.use_mongo_card_index%' => false,
'%params.elastic_search_host%' => 'elasticsearch',
```

False - означает что информация будет храниться не в монге а в эластике.
А в строке с elastic_search_host нужно за место 127.0.0.1 прописать elasticsearch

8. Запускаем сообщения в очередь для индексации card

Запускаем скрипт в контейнере loyalty в папке /home/apps/rightway/loyalty/:

```
applications/common/yiic indexbuilder --key=card
```

После выполнения скрипта, мы увидим как в RebbitMQ появятся сообщения на против консьюмера IndexBuilder.Card.Elastic.

9. Запускаем обработку сообщений в консьюмере

Цифра в конце строки означает сколько сообщений будет обработано. Лучше указать точное количество, сколько написано на против консьюмера IndexBuilder.Card.Elastic в RebbitMQ.

Тогда мы сможем понять, когда закончится обработка.

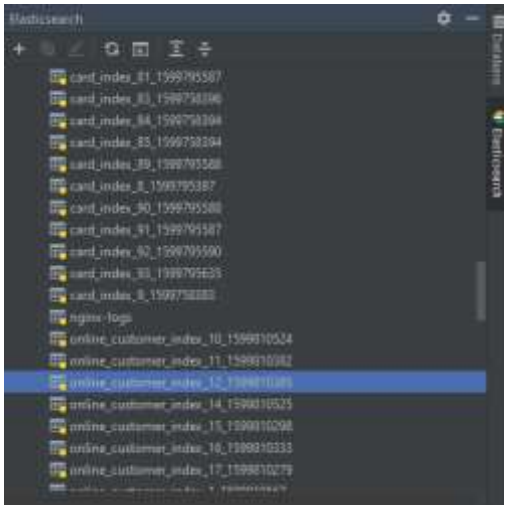
```
php /home/apps/rightway/loyalty/applications/common/yiic rabbitmq_consumer --
name=IndexBuilder.Card.Elastic --messages=58
```

После запуска, в ElasticSearch должны появляться новые кластеры (таблицы) и узлы (записи)

Если разобрались с брендом №60, то индексация должна идти менее часа, если нет, то ложитесь спать, это будет длиться около 7 часов или больше.

10. Проверяем правильно ли всё проиндексировалось

Отличный способ проверить elasticsearch - установить плагин в PHP Storm "Elasticsearch". В котором указать адрес нашего эластика (127.0.0.1) с портом 9200.
И там будет вывод всех кластеров:



Там их будет много, можно их открыть и увидеть в каких есть записи, а какие пустые.

11. Запускаем сообщения в очередь для индексации customers

Запускаем скрипт в контейнере loyalty в папке /home/apps/rightway/loyalty/:

```
applications/common/yiic indexbuilder --key=online
```

После выполнения скрипта, мы увидим как в RebbitMQ появятся сообщения на против консьюмера IndexBuilder.Online.Elastic.

12. Запускаем обработку сообщений в консьюмере

Цифра в конце строки означает сколько сообщений будет обработано. Лучше указать точное количество, сколько написано на против консьюмера IndexBuilder.Online.Elastic в RebbitMQ.

Тогда мы сможем понять, когда закончится обработка.

```
php /home/apps/rightway/loyalty/applications/common/yiic rabbitmq_consumer --name=IndexBuilder.Online.Elastic --messages=58
```

После запуска, в ElasticSearch должны появляться новые кластеры (таблицы) и узлы (записи)

Эта индексация должна пройти быстрее, несколько минут, т.к. записей намного меньше.

13. Проверяем индексацию в плагине Elasticsearch из 10 пункта.

14. По скольку мы скопировали базу с теста, то можно удалить коллекции из mongo связанные с индексацией.

В mongo в БД cardsmile есть коллекции с названием card_index_XXX и online_customer_index_XXX.
Их можно удалить, так как они нужны были для хранения индексаций в монге. Но от этого отказались.
Описание причин отказа можно почитать тут Перенос card_index из MongoDB в ElasticSearch

5.2 Индексация каталога из mongoDB

При создании локальной версии проекта, в elasticsearch не приписаны индексы.

Чтобы их заполнить существует скрипт **fos:elastica:populate**

Нужно прописывать индексы в elasticsearch, когда уже будет скопирована тестовая база mongo

Чтобы его выполнить нужно:

1. Перейти в контейнер rightway_catalog

```
docker exec -ti rightway_catalog_1 bash
```

2. Авторизоваться под пользователем apps

```
su apps
```

3. Перейти в папку rightway/rightway-catalog

```
cd ~/rightway/rightway-catalog
```

4. Выполнить команду fos:elastica:populate

```
./bin/console fos:elastica:populate
```

> *Может быть ошибка:*
> */usr/bin/env: 'php\r': No such file or directory*
>
> *То тогда у файла rightway-catalog\bin\console нужно поменять окончание строк на Unix версию. Это делается через Nodepad++*
>
> *Открыть файл rightway-catalog\bin\console в Notepad++.*
>
> *В меню выбрать Edit(Правка)->EOA Conversion(Формат конца строк)→Unix*

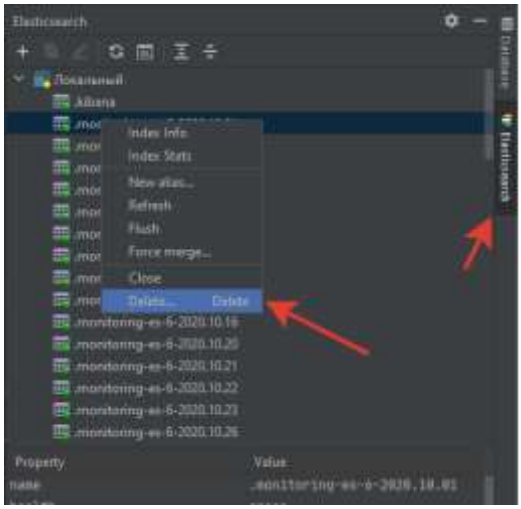
Индексировать должно долго. Примерно от 2х часов и более

5.3 Если тормозит Elasticsearch

Если Elasticsearch на локальной копии тормозит и выключается контейнер, то нужно почистить индексы.

Он просто не может все их обработать и выключает контейнер.

Сделать это можно через плагин Elasticsearch в phpStorm. Нажимаем на каждый кластер правой клавишей и жмём DELETE


Elasticsearch
Локальный
Kibana
Index Info
Index Stats
New alias...
Refresh
Flush
Force merge...
Close
Delete...
Delete
Property
Value

6 Создание docker-контейнера

Контейнер

В файле `docker/.env.dist` добавить новую переменную с шаблонизированным локальным путем до папки с новым приложением. При наличии файла `docker/.env` добавить реальный путь до приложения в него.

Добавить конфигурационные файлы для nginx и php-fpm. Расположить их следует в папке `docker/config`, примеры содержимого файлов можно посмотреть в структуре развертываемого приложения. Обработчик php-fpm следует разворачивать на свободном порту > 9010

Если приложение будет смотреть наружу то nginx нужно знать где оно находится чтобы проксировать туда запросы. Сервису с приложению нужны только путь до папки с кодом и конфиги php-fpm

В файле `docker/docker-compose.yaml` добавить настройки для контейнера. Для этого:

Добавить необходимые переменные в секцию `x-volumes`, которые будут содержать информацию в какие папки маунтить конфигурационные файлы nginx, php-fpm и куда маунтить папку с самим приложением

В настройки сервиса nginx добавить `volumes` переменные с путями до приложения и до конфигураций nginx для этого приложения. Также необходимо добавить предполагаемый url нового приложения в секцию `networks.default.aliases`

Добавить секцию с настройками сервиса. Если в приложении необходима авторизация через единую шину, необходимо добавить пункт `depends_on: auth` Пример:

```
app_name:
build:
<<: *php-context
image: rightway-php:7.1
volumes:
- *app_name-fpm
- *app_name-app
- *ssh-sec
- *ssh-pub
- *ssh-conf
- *fpm-logs
environment:
XDEBUG_CONFIG: remote_host=${XDEBUG_REMOTE_HOST}
PHP_IDE_CONFIG: serverName=app_name
```

Ansible

Добавить шаблон ansible для формирования конфигурационного файла приложения в папку `ansible/templates`. Для этого необходимо взять конфигурационный файл приложения (.env ИЛИ parameters.yaml ИЛИ что-то подобное) и заменить параметры на переменные из `ansible/common/variables.yaml`. При необходимости в файл `ansible/common/variables.yaml` можно добавить новые переменные, для использования в шаблонах.

Речь идет о файлах параметров в приложении. Когда разворачивали окружение на этапе выполнения скриптов ансибла был шаг "Настройка параметров". Нужно в папке ansible/templates добавить новый файл с названием приложения. В этом файле должна быть копия конфиг файла приложения, и параметры которые связаны с урлами, паролями, названиями баз и тд, нужно заменить на переменные которые

есть в ansible/common/variables.yaml, при необходимости можно добавить свои переменные и использовать в шаблонах. Т.е. прямой копипаст из .env .yaml как минимум с заменой переменных. Конфиг файл обычно всегда один, это либо parameters.yaml либо .env.

Далее в файле `ansible/app_config.yaml` необходимо добавить шаг конфигурации нового приложения, по аналогии с уже имеющимися, указав в виде шаблона файлы созданные на предыдущем шаге

Из гита клонировать дистрибутив приложения в папку, которую в конфиге указали. После клона, из диста скопировать env и заполнить его базовыми урлами и ключами (по аналогии с ansible). Прописать в hosts адрес приложения.

Запуск и проверка

После всех шагов необходимо проверить работоспособность приложения. Для этого нужно пересобрать локальное окружение выполнив в терминале `docker-compose up -d --build` находясь в папке `docker`. После этого в контейнере с nginx необходимо перечитать конфигурационные файлы выполнив `nginx -s reload`

В контейнере, в папке с приложением нужно выполнить composer install

7 Установка тестовых баз данных

7.1 Архивированные копии баз данных тут

dump.tar.gz (dump.zip) - тестовая копия базы cardsmile из контейнера rightway_postgres

pers.tar.gz (pers.zip) - тестовая копия базы cardsmile_personal из контейнера rightway_postgres_personal

mongo.tar.gz (mongo.zip)- тестовая копия базы mongo

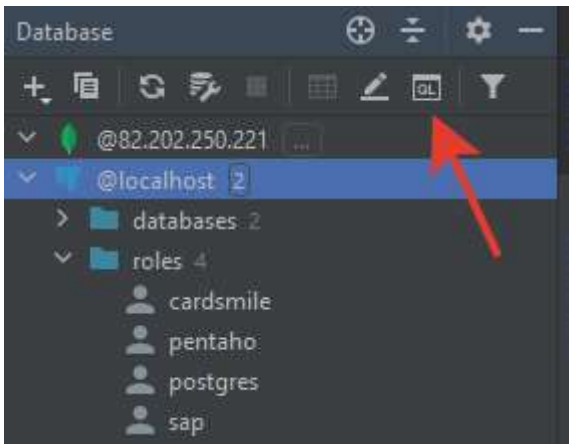
Советую работать с архивами tar.gz. Если будут проблемы с архиватором "zip", лучше архивы перенести в tar.gz.

7.2 Копирование баз Postgre.

1. Перед началом нужно создать роли

На данный момент (31.08.2020) точные настройки прав ролей не известны, нужно будет их поправить, посмотреть как на тестовом сервере

Создать роли для базы cardsmile_personal. Открыть консоль прописать по очереди.




```
create user sap
superuser
createdb
createrole
replication;


create user cardsmile
superuser
createdb
createrole
replication;


create user pentaho
superuser
```

```
createdb
createrole
replication;
```

Создать роли для базы cardsmile. Открыть консоль прописать по очереди.

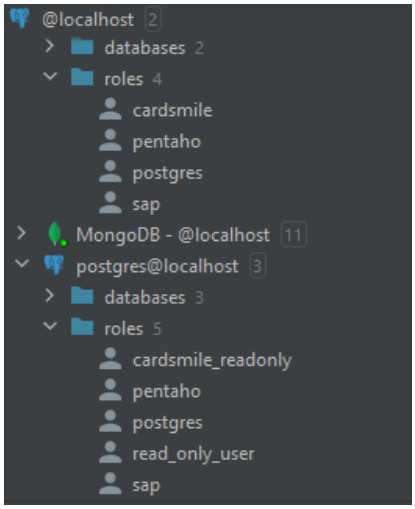
```
create user cardsmile_readonly
superuser
createdb
createrole
replication;
```

```
create user read_only_user
superuser
createdb
createrole
replication;
```

```
create user pentaho
superuser
createdb
createrole
replication;
```

```
create user sap
superuser
createdb
createrole
replication;
```

В итоге должно быть так:



2. Копируем архив в контейнер.

dump.tar.gz - контейнер rightway_postgres

pers.tar.gz - контейнера rightway_postgres_personal

Предлагаю начать с postgres_personal, чтобы было для начала попроще, объём базы меньше.

2.1. Узнаём код контейнера, набрав docker ps находясь в папке deployment\apps-deployment\docker.

2.2. Выполняем команду docker cp {путь до файла базы} {id контейнера}:{путь для нового файла в контейнере}

Копировать в контейнер лучше сначала архивы, они значительно меньше размера. А внутри уже распакуем.

Удобно загружать в контейнер в папку /tmp для избежания проблем правами на запись.

3. Заходим контейнер с нужной БД postgres.

docker exec -ti rightway_postgres_personal_1 bash

Проверить размер файла и убедиться, что он загружен полностью можно с помощью команды:

wc -c /path/to/file

4. Разархивируем базу.

В контейнере есть архиватор tar. Команда для разархивации:

tar -xvf <имя файла>

5. Авторизуемся под пользователем postgres

su postgres

6. Заходим в консоль psql.

Прописать в командной строке:

psql

6. Удаляем старую и создаём новую пустую базу

В консоли psql пишем по очереди (напомню что имя базы для postgres_personal → cardsmile_personal, а для postgres → cardsmile):

DROP DATABASE "имя базы";

CREATE DATABASE "имя базы";

В запросах обязательно в конце нужно ставить точку с запятой ";"

7. Выходим из консоли psql.

Командой \q или ctrl+z

8. Начинаем копирование базы.

Выполняем:

psql "имя базы" < "путь к файлу"

например получится (psql cardsmile_personal < pers.sql)

9. После выполнения все действий не должно быть ошибок.

Ошибки могут быть с ролями, если вы их не создали, либо с уже существующими таблицами, если вы не правильно удалили и создали базу.

Никаких строк начинающихся с ERROR: не должно быть. Но есть исключение:

ERROR: unrecognized configuration parameter "row_security".

Как написано в интернете, это связано с различающимися версиями PostgreSQL. На это не стоит обращать пока то внимание.

Если есть проверьте все пункты, и потом уже обратитесь с вопросом.

Для проверки результатов импорта можно открыть соодержимое таблиц в PHPStorm или PGAdmin.

Так же это возможно сделать прямо в консоли контейнера. Для этого можно выполнить команды

1. Заходим в консооль postgres: psql
2. Выбираем нужную БД: \c cardsmile_personal
3. Открываем список таблиц: \dt
4. Делаем выборку по заполненной таблице: SELECT * FROM public.personal_data;

Не забываем удалить архив с базой из контейнера после разворачивания бэкапа.

Как с копируется первая база postgres, переходите ко второй, начиная со 2-го пункта инструкции.

7.3 Копирование базы MongoDB.

1. Копируем архив в контейнер.

mongo.tar.gz - тестовая копия базы mongo

1.1. Узнаём код контейнера, набрав docker ps находясь в папке deployment\apps-deployment\docker.

1.2. Выполняем команду docker cp {путь до файла базы} {id контейнера}:{путь для нового файла в контейнере}

Копировать в контейнер архив. А внутри уже распакуем.

2. Заходим контейнер.

docker exec -ti rightway_mongo_1 bash

Проверить размер файла и убедиться, что он загружен полностью можно с помощью команды:

wc -c /path/to/file

3. Разархивируем базу.

В контейнере есть архиватор tar. Команда для разархивации:

tar -xvf <имя файла>

4. Заходим контейнер куда разархивировалась база.

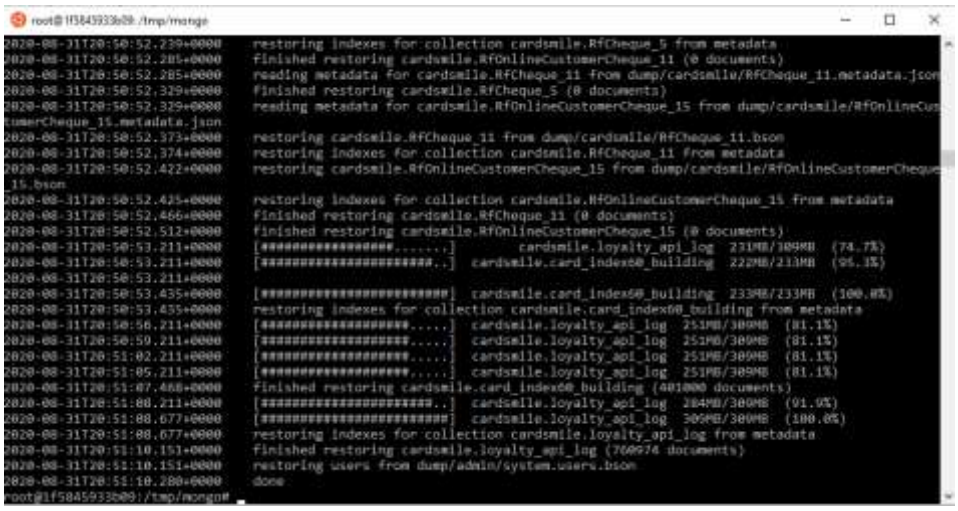
Заходим в папку mongo.

Выполняем команду:

mongorestore

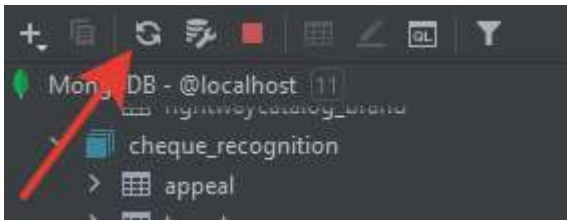
Должна пойти копирование базы.

5. После надписи "done" база должна быть скопирована.



6. После копирование базы обновите соединение в PHP Storm.

Нужно нажать на значок обновления.



Не забываем удалить архив с базой из контейнера после разворачивания бэкапа.

7. Почистить кэш в контейнере auth.

php bin/console cache:clear --env=prodType a message

7.4 Обязательно после копирования нужно сделать ещё 2 пункта: 1. Настройка после копирования тестовой БД 2. Индексация Elasticsearch

7.5 Настройка после копирования тестовой БД

7.5.1 Меняем название БД в Mongo.

Каталог брендов и товаров в Mongo находиться в БД не с тем названием, как настроено по умолчанию у нас на проекте. Нужно переименовать БД.
Для этого делаем следующее:

1. Заходим в контейнер с mongo:

```
docker exec -ti rightway_mongo_1 bash
```

2. Заходим в консоль mongo, прописав:

```
mongo
```

3. Заходим в БД rightway_catalog

```
use rightway_catalog
```

4. Меняем название БД с rightway_catalog на catalog, пишем в консоли mongo:

```
db.adminCommand({renameCollection: "rightway_catalog.rightwaycatalog_category", to: "catalog.rightwaycatalog_category"})
db.adminCommand({renameCollection: "rightway_catalog.rightwaycatalog_brand", to: "catalog.rightwaycatalog_brand"})
db.adminCommand({renameCollection: "rightway_catalog.rightwaycatalog_category_item", to: "catalog.rightwaycatalog_category_item"})
db.adminCommand({renameCollection: "rightway_catalog.rightwaycatalog_file_log", to: "catalog.rightwaycatalog_file_log"})
```

Но остаётся один вопрос, скорее всего он нам потом встретиться. В этих коллекциях есть ссылки друг на друга. И в этих ссылках указана БД как **rightway_catalog.** Скорее всего потом нужно будет их как-то поменять, но пока не нашёл способ.

7.5.2 Меняем параметр в Postgres для отображения каталога.

В БД **cardsmile** в таблице **clinent_settings** для всех строк в поле **json_settings** в конец json-массива (до фигурных скобок, после запятой) добавить параметр:

```
useClp: false
```

После этого нужно, почистить коллекции **auth** в **Redis**. Делается это внутри контейнера **auth** командой:

```
php bin/console clear_auth_cache --processingPath=/auth/processing --
webUserPath=/auth/webUser --apiPath=/auth/api --env=prod
```

Или очистить весь кэш в контейнере **Redis** командой:

```
redis-cli FLUSHALL
```

В конце требуется перелогиниться в системе.