



**Руководство по
установке
МоиОтчеты
Корпоративный
сервер**

Разворачивание кластера Kubernetes

Для установки и настройки сервера отчётов необходимо развернуть облачную экосистему Kubernetes.

Для корректной работы МоиОтчеты Корпоративный Сервер необходимо минимум три узла - один master узел и два worker узла. В процессе работ количество узлов может быть увеличено при росте нагрузки. Так же количество узлов может быть уменьшено при снижении нагрузки. Динамическое управление количеством узлов в настоящее время не реализовано.

[Видео установки Корпоративного Сервера](#)

Подготовка и установка компонентов Kubernetes

Установка Kubernetes осуществляется последовательным выполнением следующих bash скриптов. Эти примеры подразумевают операционную систему Debian Linux. В случае использования другого дистрибутива некоторые пункты будут отличаться. Например, для разворачивания кластера в системе Alt Linux Server, использовалась документация со страницы www.altlinux.org/Kubernetes, которой будет достаточно для установки Kubernetes, описанной в этой разделе.

1. Отключение раздела swap.

```
#
# Permanatly disable swap
#
sed -e '/swap/s/^\#/g' -i /etc/fstab
swapoff -a
```

2. Загрузка необходимых модулей ядра.

```
#
# Enable kernel modules
#
MODULES=/etc/modules-load.d/k8s.conf
if [ ! -f $MODULES ]; then
    echo "Create $MODULES"
    cat<<EOF | tee $MODULES
overlay
br_netfilter
EOF
fi
```

3. Настройка сетевых свойств ядра системы.

```
#
# Confugre kernel
#
SYSCTL=/etc/sysctl.d/k8s.conf
if [ ! -f $SYSCTL ]; then
    echo "Prepare kernel options"
    cat<<EOF | tee $SYSCTL
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
net.bridge.bridge-nf-call-ip6tables = 1
EOF
fi
sysctl --system
```

4. Установка дополнительного программного обеспечения, которое потребуется для установки оркестратора и сервера отчётов.

```

apt-get update

#
# Install prerequested packages
#
prerequisite=( curl sudo gnupg2 apt-transport-https ca-certificates software-properties-common )

for package in "${prerequisite[@]}"
do
    echo -n "Checking $package: "
    dpkg -s $package > /dev/null 2> /dev/null
    if [ $? -ne 0 ]; then
        echo "Installing"
        apt-get install -y $package
    else
        echo "OK"
    fi
done

```

5. Установка Container Runtime Interface (CRI).

Следующий пример использует CRI containerd. Различные версии Kubernetes и Linux могут потребовать установки CRI-O. Kubernetes использует CRI для загрузки контейнеров, управления контейнерами и для запуска процессов в этих контейнерах.

```

## Add Docker's official GPG key
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | apt-key --keyring /etc/apt/trusted.gpg.d/docker.gpg add -

## Add Docker apt repository.
add-apt-repository \
    "deb [arch=amd64] https://download.docker.com/linux/debian \
    $(lsb_release -cs) \
    stable"

## Install containerd
apt-get update && apt-get install -y containerd.io

# Configure containerd
if [ ! -d /etc/containerd ]; then
    mkdir -p /etc/containerd
fi

# Remove default config to avoid errors
if [ -f /etc/containerd/config.toml ]; then
    rm /etc/containerd/config.toml
fi

# Restart containerd
systemctl restart containerd

```

6. Установка компонентов Kubernetes.

```

curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add -
cat <<EOF | tee /etc/apt/sources.list.d/kubernetes.list
deb https://apt.kubernetes.io/ kubernetes-xenial main
EOF
apt-get update
apt-get install -y kubelet kubeadm kubectl
apt-mark hold kubelet kubeadm kubectl

```

Старт Kubernetes

МоиОтчеты Корпоративный Сервер и импортозамещение

Версии компонентов Kubernetes

Изначально продукт разрабатывался для кластера, развёрнутого на основе дистрибутива Debian GNU/Linux 10 (buster). Для поддержки импортозамещения был протестирован ряд дистрибутивов Linux российских производителей. По итогам тестирования зависимости от версии Kubernetes не выявлено - минимальные номера версий, на которых производилось тестирование показаны в следующей таблице:

КОМПОНЕНТ	ВЕРСИЯ
docker	19.3.8
kubelet	1.18.1
kube-proxy	1.18.1

При тестировании проверялись также более свежие версии Kubernetes. Обновление версии Kubernetes не нарушает работу сервера отчётов. Таким образом тестирование отечественных дистрибутивов свелось к проверке возможности установки Kubernetes.

Провайдеры CNI

Сервер отчётов был протестирован со CNI провайдерами Flannel и Calico. Оба провайдера показали удовлетворительную работу. Конфигурация с Flannel была протестирована на самостоятельно развёрнутом кластере, конфигурация с Calico была протестирована на облачном провайдере Selectel (Россия).

Российские дистрибутивы Linux

Результаты тестирования представлены в следующей таблице:

ПРОИЗВОДИТЕЛЬ	СОВМЕСТИМОСТЬ	ПРИМЕЧАНИЯ
Alt Linux Server 9	Полная	https://www.altlinux.org/Kubernetes
ROSA "Хром" Linux	Заявлена	http://wiki.rosalab.com/ru/index.php/Установка_docker,_kebernates_кластера
ASTRA Linux 1.7	Deckhouse	https://deckhouse.io/ru/
RED OS	Заявлена	https://redos.red-soft.ru/base/server-configuring/container/kubernetes/
AlterOS 7	Deckhouse	https://deckhouse.io/ru/

Примечание: Deckhouse - продукт российских разработчиков для разворачивания кластера.

Установка сервиса nginx-ingress

Все описанные команды выполняются только на master узле!

Добавление в Helm репозитория ingress-nginx:

```
helm repo add ingress-nginx https://kubernetes.github.io/ingress-nginx
helm repo update
```

Установка nginx в Kubernetes

Внимание! Переменная `MAIN_IF` должна указывать IP адрес внешнего интерфейса для доступа к серверу отчётов.

```
export MAIN_IF=192.168.1.191
kubectl create namespace nginx

helm upgrade --install nginx-ingress ingress-nginx/ingress-nginx \
  --create-namespace --namespace nginx \
  --set controller.replicaCount=2 \
  --set controller.service.externalIPs[0]=$MAIN_IF \
  --set controller.extraArgs.v=2
```

Следующая команда конфигурирует `nginx` в кластере. `nginx` будет принимать все входящие запросы и перенаправлять их в `gateway`, который распределяет входящие запросы между компонентами сервера отчётов.

```
HOST=my.server-server.ru
NAMESPACE=fr-corporate-server

cat <<EOF | kubectl apply -n $NAMESPACE -f -
kind: Ingress
apiVersion: networking.k8s.io/v1
metadata:
  name: $HOST-gateway
  namespace: $NAMESPACE
  annotations:
    ingress.kubernetes.io/ssl-redirect: 'true'
    nginx.ingress.kubernetes.io/configuration-snippet: |
      add_header X-Robots-Tag "noindex, nofollow, nosnippet, noarchive";
    nginx.ingress.kubernetes.io/limit-rps: '50'
    nginx.ingress.kubernetes.io/proxy-body-size: '0'
    nginx.ingress.kubernetes.io/proxy-buffering: 'off'
    nginx.ingress.kubernetes.io/proxy-request-buffering: 'off'
spec:
  ingressClassName: nginx
  tls:
  - hosts:
    - $HOST
    secretName: corporate-tls-secret
  rules:
  - host: $HOST
    http:
      paths:
      - path: /
        pathType: ImplementationSpecific
        backend:
          service:
            name: fr-gateway
            port:
              number: 80
EOF
```

Далее понадобится SSL сертификат для настройки защищённого соединения. Обычно его можно купить/получить у регистратора доменного имени или купить у удостоверяющего центра. При использовании сервера отчётов в интранете можно создать самостоятельно подписанный сертификат с помощью следующей команды:

```
export CERT_NAME=my.server-server.ru
openssl req -x509 -nodes -days 3650 -newkey rsa:2048 -keyout del_me_file.key -out del_me_file.cer -subj
"/CN=$CERT_NAME/O=$CERT_NAME"
```

Регистрация сертификата в Kubernetes с именем `fr-corporate-tls`. Далее этот сертификат используется различными компонентами отчёта, в том числе `nginx`.

```
kubectl create secret tls fr-corporate-tls --key del_me_file.key --cert del_me_file.cer
```

Регистрация `nginx-ingress` в кластере Kubernetes.

Необходимо установить переменную `HOST`, соответствующую доменному имени сервера отчётов.

```
cat <<EOF | kubectl apply -n $NAMESPACE -f -
kind: Ingress
apiVersion: networking.k8s.io/v1
metadata:
  name: $HOST-gateway
  namespace: $NAMESPACE
  annotations:
    ingress.kubernetes.io/ssl-redirect: 'true'
    nginx.ingress.kubernetes.io/configuration-snippet: |
      add_header X-Robots-Tag "noindex, nofollow, nosnippet, noarchive";
    nginx.ingress.kubernetes.io/limit-rps: '50'
    nginx.ingress.kubernetes.io/proxy-body-size: '0'
    nginx.ingress.kubernetes.io/proxy-buffering: 'off'
    nginx.ingress.kubernetes.io/proxy-request-buffering: 'off'
spec:
  ingressClassName: nginx
  tls:
    - hosts:
      - $HOST
      secretName: corporate-tls-secret
  rules:
    - host: $HOST
      http:
        paths:
          - path: /
            pathType: ImplementationSpecific
            backend:
              service:
                name: fr-gateway
                port:
                  number: 80
EOF
```


Лог установки helm и пакетов №2

Kubernetes-Dashboard это WEB-интерфейс для управления кластером. Он позволяет с помощью WEB-браузера управлять Kubernetes. Dashboard не является необходимым компонентом для работы сервера отчётов, однако он позволяет упростить настройку и конфигурирование сервера, а также мониторить состояние узлов и контейнеров.

Установка Dashboard

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.0.0-rc7/aio/deploy/recommended.yaml
```

Конфигурирование Dashboard

При конфигурации Dashboard используется переменная `SERVER_DOMAIN_NAME`. Обратите внимание, нижеприведённая конфигурация использует Dashboard на том же самом IP адресе, что и сервер отчётов. Для захода на страницу Dashboard используется правило, определяющее путь к Dashboard.

```
export SERVER_DOMAIN_NAME="my.server-server.ru"

cat <<EOF | kubectl apply -f -
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: fastreport.cloud-dashboard
  namespace: kubernetes-dashboard
  annotations:
    kubernetes.io/ingress.class: "nginx"
    nginx.ingress.kubernetes.io/backend-protocol: HTTPS
    nginx.ingress.kubernetes.io/use-regex: "true"
    nginx.ingress.kubernetes.io/rewrite-target: "/$2"
    nginx.ingress.kubernetes.io/configuration-snippet: |
      rewrite ^(/dashboard)\$ \$1/ redirect;

spec:
  tls:
  - hosts:
    - $SERVER_DOMAIN_NAME
    secretName: fr-corporate-tls
  rules:
  - host: $SERVER_DOMAIN_NAME
    http:
      paths:
      - path: /dashboard(/|/)(.*)
        backend:
          serviceName: kubernetes-dashboard
          servicePort: 443
EOF
```

Проверить созданную конфигурацию nginx сервера можно с помощью следующих команд, используя вместо `nginx-ingress-controller-6674ff5868-t47xk` имя созданного nginx контейнера:

```
kubectl exec -it nginx-ingress-controller-6674ff5868-t47xk -n nginx -- ls /etc/nginx/
kubectl exec -it nginx-ingress-controller-6674ff5868-t47xk -n nginx -- cat /etc/nginx/nginx.conf
```

Далее необходимо получить token, который будет использоваться для административного доступа к Dashboard. Создание административной учётной записи:

```
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: ServiceAccount
metadata:
  name: dashboard-admin-user
  namespace: kube-system
EOF
```

Назначение прав доступа административной учётной записи:

```
cat <<EOF | kubectl apply -f -
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: dashboard-admin-user
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- kind: ServiceAccount
  name: dashboard-admin-user
  namespace: kube-system
EOF
```

Создание токена:

```
kubectl -n kubernetes-dashboard create token dashboard-admin-user -n kube-system
```

Распаковка и вывод на экран токена, используемого для авторизации на Dashboard:

```
kubectl -n kube-system describe secret $(kubectl -n kube-system get secret | grep dashboard-admin-user | awk '{print $1}')
```

Сохраните токен для последующего использования в недоступном для посторонних месте и используйте его для авторизации в Dashboard.

Пример URL для доступа к dashboard:

```
https://my.server-server.ru/dashboard/
```

Лог установки helm и пакетов №3

Grafana это компонент аналитики и интерактивной визуализации для мониторинга состояния кластера. Это необязательный компонент для разворачивания сервера отчётов, но он может быть полезен для контроля состояния сервера.

Loki это агрегатор логов, который используется совместно с Grafana.

Создание пространства имён для компонентов мониторинга

```
kubectl create namespace monitoring
```

Обновление чартов для установки компонентов мониторинга

```
helm repo add loki https://grafana.github.io/loki/charts
helm repo update
```

Установка Loki

```
helm upgrade \
  --install loki loki/loki \
  --namespace=monitoring \
  --set persistence.enabled=true \
  --set persistence.storageClassName=hcloud-volumes
```

```
helm upgrade \
  --install promtail loki/promtail \
  --namespace=monitoring \
  --set loki.serviceName=loki.monitoring
```

Установка Grafana

Не забудьте установить переменную `SERVER_DOMAIN_NAME`.

```
export SERVER_DOMAIN_NAME="<доменное имя сервера>"

helm upgrade \
  --install grafana stable/grafana \
  --namespace=monitoring \
  --set persistence.enabled=true \
  --set persistence.storageClassName=hcloud-volumes \
  --set 'grafana.ini'.server.serve_from_sub_path=true \
  --set 'grafana.ini'.server.root_url=https://$SERVER_DOMAIN_NAME/grafana/ \
  --set 'grafana.ini'.server.domain=$SERVER_DOMAIN_NAME \
  --set readinessProbe.httpGet.path=/grafana/api/health \
  --set livenessProbe.httpGet.path=/grafana/api/health \
```

Получение пароля от Grafana

```
kubectl get secret --namespace monitoring grafana -o jsonpath="{.data.admin-password}" | base64 --decode ; echo
```

```
admin  
*****
```

Добавление ingress

Этот блок конфигурации позволяет расположить Grafana на том же самом сетевом интерфейсе, что и сервер отчётов и обращаться к ней по ссылке вида: <https://my.server-server.ru/grafana>

```
cat <<EOF | kubectl apply -f -  
apiVersion: extensions/v1beta1  
kind: Ingress  
metadata:  
  name: fastreport.cloud-grafana  
  namespace: monitoring  
  annotations:  
    kubernetes.io/ingress.class: "nginx"  
    ingress.kubernetes.io/ssl-redirect: "true"  
spec:  
  tls:  
  - hosts:  
    - $SERVER_DOMAIN_NAME  
    secretName: fr-corporate-tls  
  rules:  
  - host: $SERVER_DOMAIN_NAME  
    http:  
      paths:  
      - path: /grafana  
        backend:  
          serviceName: grafana  
          servicePort: 80  
EOF
```

Для завершения конфигурации Grafana необходимо зайти на её страницу, авторизоваться и добавить ссылку на Loki <http://loki.monitoring:3100>

Установка брокера сообщений RabbitMQ

Для установки RabbitMQ необходимо определить переменную `RABBITMQ_PASSWORD`, содержащую пароль. Также можно настроить размер тома для очереди брокера сообщений.

Добавление чарта и обновление helm репозитория.

```
helm repo add bitnami https://charts.bitnami.com/bitnami
helm repo update
```

Создать пространство имён для RabbitMQ.

```
kubectl create namespace rabbitmq
```

Установка RabbitMQ может варьироваться от типа установки. Ниже приведены два примера - для провайдера Hetzner и для установки на тестовой конфигурации.

Установка на облачном хостинге Hetzner

Нижеприведённый скрипт использует размер очереди 10 Гб.

```
export RABBITMQ_PASSWORD="<пароль для RabbitMQ>"

cat <<EOF > ./rabbitmq.yaml
rabbitmq:
  extraConfiguration: |-
    #disk_free_limit.absolute = 10GB
    management.path_prefix = /rabbit
    #management.load_definitions = /app/load_definition.json
  persistence:
    enabled: true
    storageClass: hcloud-volumes
    size: 10Gi
  livenessProbe:
    commandOverride:
      - sh
      - -c
      - rabbitmq-api-check "http://user:\$RABBITMQ_PASSWORD@127.0.0.1:15672/rabbit/api/healthchecks/node"
      '{"status":"ok"}'
  readinessProbe:
    commandOverride:
      - sh
      - -c
      - rabbitmq-api-check "http://user:\$RABBITMQ_PASSWORD@127.0.0.1:15672/rabbit/api/healthchecks/node"
      '{"status":"ok"}'
EOF

helm upgrade \
  --install rabbitmq bitnami/rabbitmq \
  --namespace=rabbitmq \
  --values ./rabbitmq.yaml

rm ./rabbitmq.yaml
```

Пример минимальной установки RabbitMQ для разворачивания на тестовом кластере

Нижеприведённая конфигурация использует размер очереди 4 Гб.

Обратите внимание - данный пример является минимальным для запуска МоиОтчеты Корпоративный Сервер и описанная конфигурация томов не является отказоустойчивой.

```
apiVersion: v1
kind: PersistentVolume
metadata:
  namespace: "rabbitmq"
  name: pv-for-rmq
  labels:
    name: mongo-volume-1-0-0
spec:
  storageClassName: manual
  capacity:
    storage: 4Gi
  accessModes:
  - ReadWriteOnce
  hostPath:
    path: /devkube/rabbitmq
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: "rabbitmq-pvc"
  namespace: "rabbitmq"
spec:
  accessModes:
  - ReadWriteOnce
  storageClassName: manual
  resources:
    requests:
      storage: 4Gi
```

Выполните `kubectl apply -f <имя вышеприведённого yaml>` и собственно скрипт установки RabbitMQ. Обратите внимание, что используется ранее созданный PersistentVolumeClaim.

```
helm upgrade \
--namespace=rabbitmq \
--install rabbitmq bitnami/rabbitmq \
--set persistence.existingClaim=rabbitmq-pvc \
--set volumePermissions.enabled=true \
--set ingress.tlsSecret="fr-corporate-tls" \
| tee RabbitMQ.txt
```

Добавление конфигурации ingress

Этот пункт необходим если вы хотите получить доступ к контрольной панели сервиса RabbitMQ извне кластера.

```
export SERVER_DOMAIN_NAME="<доменное имя сервера>"
```

```
cat <<EOF | kubectl apply -f -
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: fastreport.cloud-rabbitmq
  namespace: rabbitmq
  annotations:
    kubernetes.io/ingress.class: "nginx"
    nginx.ingress.kubernetes.io/rewrite-target: "\$2"
    nginx.ingress.kubernetes.io/configuration-snippet: |
      rewrite ^(/rabbit)\$ \$1/ redirect;
spec:
  tls:
    - hosts:
      - \$SERVER_DOMAIN_NAME
      secretName: fr-corporate-tls
  rules:
    - host: \$SERVER_DOMAIN_NAME
      http:
        paths:
          - path: /rabbit
            backend:
              serviceName: rabbitmq
              servicePort: 15672
EOF
```

Доступ по .

Установка MongoDB в кластер

Добавление репозитория для helm:

```
helm repo add bitnami https://charts.bitnami.com/bitnami
helm repo update
```

Сервер отчётов использует MongoDB для хранения шаблонов отчётов, подготовленных отчётов и различных документов, экспортированных из подготовленных отчётов.

Подготовка пространства имён:

```
NAMESPACE=mongo
MONGO_USER=any_name
MONGO_PASS=mongo_password
MONGO_DB_SIZE=10Gi
MONGO_SERVICE_NAME=fr-mongo
kubectl create namespace $NAMESPACE
```

Обратите внимание на переменную MONGO_SERVICE_NAME - это имя в дальнейшем будет использоваться для обращения к MongoDB из сервера отчётов.

Дальнейшая настройка будет отличаться от того, используется ли хранилище облачного провайдера или локальное хранилище. Ниже приведено два примера. Используйте только один из них, или модифицируйте настройки для своего облачного провайдера.

Настройка хранилища для облачного провайдера Hetzner

```
helm install $MONGO_SERVICE_NAME bitnami/mongodb \
--namespace $NAMESPACE \
--set persistence.enabled=true \
--set persistence.storageClass=hcloud-volumes \
--set persistence.size=$MONGO_DB_SIZE \
--set auth.username=$MONGO_USER \
--set auth.password=$MONGO_PASS \
| tee MongoDB.txt
```

Настройка локального хранилища

StorageClass предоставляет средства для администраторов для описания "классов" хранилищ, которые они предоставляют.


```

MONGO_DATA_DIR=/devkube/mongoddb
MONGO_DB=reports_db # Можно использовать любое имя

##
## Create StorageClass for MongoDB
##
cat <<EOF | kubectl apply -n $NAMESPACE -f -
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: local-sc-mongoddb
provisioner: kubernetes.io/no-provisioner
volumeBindingMode: WaitForFirstConsumer
EOF

##
## Create Persistent volume for MongoDB
##

cat <<EOF | kubectl apply -n $NAMESPACE -f -
apiVersion: v1
kind: PersistentVolume
metadata:
  name: mongo-storage
  namespace: $NAMESPACE
  labels:
    type: local
spec:
  capacity:
    storage: $MONGO_DB_SIZE
  accessModes:
    - ReadWriteOnce
  storageClassName: local-sc-mongoddb
  local:
    path: $MONGO_DATA_DIR
  nodeAffinity:
    required:
      nodeSelectorTerms:
        - matchExpressions:
            - key: kubernetes.io/hostname
              operator: In
              values:
                - altlinux-10-2
EOF

```

Для установки MongoDB используйте следующий скрипт:

```

helm install $MONGO_SERVICE_NAME bitnami/mongoddb \
  --namespace $NAMESPACE \
  --set persistence.storageClass=local-sc-mongoddb \
  --set auth.username=$MONGO_USER \
  --set auth.password=$MONGO_PASS \
  --set auth.database=$MONGO_DB | tee MongoDB.txt

```

Результатом выполнения этого скрипта, помимо установки MongoDB в кластер, будет создан файл, содержащий информацию о результате установки базы данных. Обратите внимание, что параметры, передаваемые скрипту, были описаны в самом начале этого документа.

Проверить корректность установки MongoDB проще всего при помощи Kubernetes Dashboard, установка которого была описана ранее.

Установка компонентов сервера отчётов МоиОтчеты Корпоративный Сервер на узлы кластера Kubernetes

Архив с образами компонентов сервера отчётов называется `frcs.tar.gz`. Ссылку на скачивание архива предоставляет компания ООО "Быстрые отчеты" через электронную почту.

Для разворачивания сервера отчётов необходимо выполнить следующую последовательность на **каждом** узле кластера:

1. Проверить наличие установленной утилиты `ctr`. Если отсутствует, то установить утилиту `podman` (<https://podman.io/>), и для установки лучше воспользоваться пакетным менеджером используемого дистрибутива.
2. Скопировать на каждый узел кластера архив с продуктом и выполнить декомпрессию архива с помощью команды `gunzip frcs.tar.gz`.
3. Загрузить образы из tar-архива в локальный репозиторий образов. В случае, если `ctr` доступна, команда для загрузки образа - `ctr -n=k8s.io image import frcs.tar`, иначе с воспользуйтесь утилитой `podman`: `podman load -i frcs.tar`.
4. Проверить, что образы доступны в локальном хранилище с помощью команды: `cricctl img`.

В случае, если всё прошло без ошибок, вывод `cricctl img` будет подобен следующему примеру:

IMAGE	TAG	IMAGE ID	SIZE
docker.io/bitnami/bitnami-shell		10-debian-10-r394	45faa1e302205 83.1MB
docker.io/bitnami/mongodb		4.4.13-debian-10-r46	1fc49bc75137e 437MB
docker.io/bitnami/rabbitmq		3.9.14-debian-10-r15	6c871950ea79a 240MB
docker.io/kubernetesui/dashboard		v2.5.0	57446aa2002e1 226MB
docker.io/kubernetesui/metrics-scraper		v1.0.7	7801cfc6d5c07 34.5MB
docker.io/library/nginx	1.14.2	295c7be079025	113MB
docker.io/rancher/mirrored-flannelcni-flannel-cni-plugin		v1.0.1	ac40ce6257406 8.38MB
docker.io/rancher/mirrored-flannelcni-flannel		v0.17.0	9247abf086779 60.8MB
k8s.gcr.io/coredns/coredns	v1.8.4	8d147537fb7d1	47.7MB
k8s.gcr.io/defaultbackend-amd64	1.5	b5af743e59849	5.14MB
k8s.gcr.io/e2e-test-images/jessie-dnsutils	1.3	4db8afc88fa88	261MB
k8s.gcr.io/etcd	3.5.0-0	0048118155842	296MB
k8s.gcr.io/ingress-nginx/controller	<none>	c1695499dda39	288MB
k8s.gcr.io/ingress-nginx/kube-webhook-certgen	<none>	c41e9fcadf5a2	49.1MB
k8s.gcr.io/kube-apiserver	v1.22.5	059e6cd8cf78e	130MB
k8s.gcr.io/kube-controller-manager	v1.22.5	04185bc88e08d	123MB
k8s.gcr.io/kube-proxy	v1.22.8	c1cfbd59f7747	105MB
k8s.gcr.io/kube-scheduler	v1.22.5	935d8fdc2d521	53.9MB
k8s.gcr.io/pause	3.5	ed210e3e4a5ba	690kB
localhost/fastreport-corporate-server-app	2022.2.3	2fc71408d7440	266MB
localhost/fastreport-corporate-server-backend	2022.2.3	154ab29853315	225MB
localhost/fastreport-corporate-server-designer	2022.2.3	fee8cebc69ebb	220MB
localhost/fastreport-corporate-server-gateway	2022.2.3	3ba7f871d665d	223MB
localhost/fastreport-corporate-server-homer	2022.2.3	da03d26d32fc6	264MB
localhost/fastreport-corporate-server-scheduler	2022.2.3	f39731aa92549	217MB
localhost/fastreport-corporate-server-static-preview	2022.2.3	0d3d42c9e1289	235MB
localhost/fastreport-corporate-server-workercore	2022.2.3	7bd7a33418995	898MB
localhost/fastreport-corporate-server-fonts	2022.2.3	b0f5e585a7563	216MB
us.gcr.io/k8s-artifacts-prod/ingress-nginx/controller	v0.34.1	6fb0739a741f4	332MB

На этом этапе подготовка к разворачиванию сервера отчётов закончена, дальнейшие процедуры описаны в документе [Установка МоиОтчеты Корпоративный Сервер](#)

Настройка сертификатов

Для работы сервера отчётов необходимо создать сертификат.

Чтобы создать самостоятельно подписанный сертификат, используйте следующие команды:

```
#!/bin/sh

openssl req -x509 -nodes -new -sha256 -days 1024 -newkey rsa:2048 -keyout RootCA.key -out RootCA.pem -subj
"/C=US/CN=debian-master.fast-report.com"
openssl x509 -outform pem -in RootCA.pem -out RootCA.crt
openssl pkcs12 -export -out ./certificate.pfx -inkey RootCA.key -in RootCA.crt
```

Чтобы создать секрет, используйте следующие команды:

```
NAMESPACE=fr-corporate-server
SECRET_VOLUME_NAME=corporate-volume-secret
kubectl create namespace $NAMESPACE
kubectl create secret generic $SECRET_VOLUME_NAME -n $NAMESPACE --from-file=certificate.pfx
```

Альтернативный способ генерации сертификата, предложенный пользователем продукта

1. Создайте и отредактируйте файл `san.cnf`:

```
[ req ]
default_bits = 2048
default_md = sha256
distinguished_name = req_distinguished_name
req_extensions = v3_req
[ req_distinguished_name ]
countryName = CN # C=
stateOrProvinceName = Shanghai # ST=
localityName = MyCity # L=
#postalCode = 200000 # L/postalcode=
#streetAddress = "My Address" # L/street=
organizationName = My Corporation # O=
organizationalUnitName = My Department # OU=
commonName = myname.mysoftware.mycorporation.com # CN=
emailAddress = myname@example.com # CN/emailAddress=
[ v3_req ]
subjectAltName = @alt_names
[ alt_names ]
DNS.1 = myname.mysoftware.mycorporation.com
#DNS.2 = other2.com
#DNS.3 = other3.coM
```

2. Сгенерируйте сертификат:

```
openssl req -x509 -nodes -days 365 -subj "/C=CN/ST=Shanghai/L=Shanghai/O=My Corporation/OU=My
Department/CN=myname.mysoftware.mycorporation.com/emailAddress=myname@example.com" -keyout privateKey.pem
-out public.crt -config san.cnf -extensions v3_req
openssl pkcs12 -export -out ./certificate.pfx -inkey privateKey.key -in public.crt
```

3. Создайте секрет в облачном сервере:

```
NAMESPACE=fr-corporate-server  
SECRET_VOLUME_NAME=corporate-volume-secret
```

```
kubectl create namespace $NAMESPACE  
kubectl create secret generic $SECRET_VOLUME_NAME -n $NAMESPACE --from-file=certificate.pfx
```

Настройка параметров МоиОтчеты Корпоративный Сервер

Данный раздел описывает процесс конфигурирования сервера отчётов. Здесь описана базовая информация. Для тонкой настройки сервера обратитесь к руководству `admi_config`. Конфигурация сохраняется в защищенном хранилище Kubernetes. Эту операцию оптимально выполнить до старта контейнеров, описанных в секции `fr-corporate-server`.

Базовые параметры сервера отчётов определены ниже. Для удобства они показаны в отдельной секции. Возможные варианты: поместить этот фрагмент в отдельный файл и с помощью `операции точка` внедрить их в конфигурационный скрипт. Или же скопировать этот фрагмент непосредственно в конфигурационный скрипт. Так возможно дописать `export` перед каждой переменной и экспортировать в переменные среды.

Значение переменной `MONGO_HOST` можно проверить в Kubernetes Dashboard - оно соответствует имени сервиса и его пространству имён, разделённых точкой. Для успешного старта сервера отчётов необходима лицензия - специальная строка, описывающая права использования сервера отчётов. Получите её у компании ООО "Быстрые отчеты".

Значения параметров `MONGO_USER` и `MONGO_PASS` должны соответствовать тем же самым параметрам, что указаны при установке MongoDB.

```

NAMESPACE=fr-corporate-server
CLOUD_ENV=prod

## Пароль и имя пользователя могут быть установлены любые, главное, чтобы совпадали с теми, что были заданы
при установке MongoDB.
## MONGO_DB должна быть admin.
## Однако, есть правила установки имени хоста - оно образуется из имени сервиса и пространства имён, в котором
зарегистрирована MongoDB.

MONGO_HOST=fr-mongo-mongodb.mongo
MONGO_USER=root
MONGO_PASS=mxzgrkGvvk
MONGO_DB=admin

## Следующие параметры лучше ЗДЕСЬ не изменять. Хост и имя пользователя устанавливаются автоматически, а
## пароль для RabbitMQ генерируется автоматически на момент установки и сохраняется в хранилище секретов
Kubernetes.
RABBITMQ_HOST=rabbitmq.rabbitmq.svc.cluster.local
# RABBITMQ_HOST=rabbitmq-0.rabbitmq-headless.rabbitmq
RABBITMQ_USER=user
RABBITMQ_PASS=$(kubectl get secret --namespace rabbitmq rabbitmq -o jsonpath="{.data.rabbitmq-password}" | base64 -
-decode)

## Разработчики утверждают, что здесь может быть любая последовательность символов. И длина не определена.
## Для теста были использованы следующие значения, и установка прошла успешно

DESIGNER_SECRET=yXgQDpNzohhN3dAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAHrXiepieQtOaLwdqfRijMaZwy5p27tho7XK4C
WORKER_SECRET=9bYuYu4cvt73oWD3AAAAAAAAAAAAAAAAAAAAAAAAAAAAA5Z6i6pRzkR8mKGL1pTRnns3P0clxzTuVu
GATEWAY_SECRET=R0jEINg6OA4TssyAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAY46ZtsYowl7gB9yteZNMsvPoL6sGgiWC4
SCHEDULER_SECRET=fcfFZNWC3zyBmAAAAAAAAAAAAAAAAAAAAAAAAAAAAA5gD2MeexV3svLdy87qC6bX3bM2W6m
ZjRC

## В эту переменную должна быть вписана лицензия, полученная у компании ООО "Быстрые отчеты". Увы, без
лицензии сервер отчётов работать не будет.
## Не используйте кавычки.

LICENSE=

## Строка подключения к MongoDB формируется автоматически на основе ранее определённых переменных. Тут
лучше ничего не менять.

MONGO_ACCESS="mongodb://$MONGO_USER:$MONGO_PASS@$MONGO_HOST:27017/"
CONNECTION_STRING="$MONGO_ACCESS?
authSource=$MONGO_DB&readPreference=primary&appName=MongoDB%20Compass&ssl=false&maxPoolSize=100&wait
QueueMultiple=100"

```

Регистрация пространства имён сервера отчётов в кластере:

```
kubectl create namespace $NAMESPACE
```

Создание минимальной конфигурации сервера отчётов:

```

cat <<EOF | kubectl apply -n $NAMESPACE -f -
apiVersion: v1
kind: ConfigMap
metadata:
  name: fast-report-config
data:
  appsettings.Production.json: |
    {

```

```

"Auth": {
  "UseOpenId": false,
  "UseLocal": true
},

"MainConfig": {
  "InternalHeaders": {
    "$WORKER_SECRET": "000000000000000000000001",
    "$DESIGNER_SECRET": "000000000000000000000002",
    "$GATEWAY_SECRET": "000000000000000000000003",
    "$SCHEDULER_SECRET": "000000000000000000000004"
  },
  "License": "$LICENSE",
  "Server": {
    "Title": "<Заголовок в приложении>",
    "Copyright": "<Краткое описание авторского права>",
    "LogoLink": "<путь к логотипу, отображаемому в панели пользователя>",
    "LastSlaUpdate": "2022-11-28",
    "CorporateServerMode": false,
    "SlaLink": "<Ссылка на пользовательское соглашение>",
    "FirstStepsVideoLink": "https://rutube.ru/play/embed/f64e80d2fa8e2d73b30c47483fa2b59e",
    "AboutLink": "https://быстрыеотчеты.рф/products/corporate-server",
    "HomePageLink": "быстрыеотчеты.рф/ru",
    "AuthServerName": "ID Быстрые отчеты",
    "UsersPerWorkSpace": 0
  },
  "Rabbit": {
    "Host": "$RABBITMQ_HOST",
    "Port": 5672,
    "UserName": "$RABBITMQ_USER",
    "Password": "$RABBITMQ_PASS",
    "QueueName": "ReportProcessQueue",
    "DirectExchangeName": "DirectEx",
    "AlternateExchangeName": "AExchange",
    "UnroutedQueueName": "Default",
    "Capacity": 1
  },
  "Database": {
    "ConnectionString": "$CONNECTION_STRING",
    "DatabaseName": "$MONGO_DB"
  },
  "Gateway": {
    "BackendUrl": "http://fr-backend.$NAMESPACE:80",
    "InternalKey": "$GATEWAY_SECRET",
    "SignInPagePath": "/account/signin?r={0}",
    "MaxConcurrentRequests": 200,
    "RequestQueueLimit": 5000
  },
  "Serilog": {
    "MinimumLevel": {
      "Default": "Debug"
    }
  },
  "Services": {
    "Items": {
      "OnlineDesigner": {
        "Namespace": "$NAMESPACE"
      },
      "Backend": {
        "Namespace": "$NAMESPACE"
      },
      "FrontendApp": {
        "Namespace": "$NAMESPACE"
      },
      "Fonts": {

```

```

"Namespace": "$NAMESPACE"
},
"StaticPreviewApp": {
  "K8sServiceName": "fr-s-preview",
  "Namespace": "$NAMESPACE",
  "HostType": "WebApp",
  "PathBase": "/staticpreview",
  "Type": "K8s",
  "PingPath": "/staticpreview/",
  "IsSignInRequired": false,
  "Priority": 21
},
"Default": {
  "Namespace": "$NAMESPACE"
},
"HomerApp": {
  "Namespace": "$NAMESPACE",
  "WhiteListClaims": {
    "cloud_service_access": "super_user"
  }
}
}
},
"Designer": {
  "BackendUrl": "http://fr-backend.$NAMESPACE:80",
  "InternalKey": "$DESIGNER_SECRET"
},
"WorkerCore": {
  "BackendUrl": "http://fr-backend.$NAMESPACE:80",
  "InternalKey": "$WORKER_SECRET"
},
"Scheduler": {
  "BackendUrl": "http://fr-backend.$NAMESPACE:80",
  "InternalKey": "$SCHEDULER_SECRET"
}
}
}
EOF

```

Обратите внимание, вышеприведённый фрагмент shell-скрипта содержит базовые настройки, проверенные разработчиками. При первоначальном разворачивании сервера отчётов рекомендуется использовать их, чтобы убедиться в работоспособности сервера. Перед изменением настроек сохраните эту версию, таким образом вы сэкономите значительно время при тонкой настройке сервера, используя параметры, описанные в секции `admin_config`.

Установка МоиОтчеты Корпоративный Сервер

Минимальная конфигурация корпоративного сервера отчётов состоит из девяти компонентов, описанных ниже. Перед установкой необходимо определить переменные, например, в файле `config.sh`. Во избежание конфликтов имён лучше иметь глобальный файл конфигурации для всех скриптов отчёта. Ниже показан пример инициализации переменных, используемых скриптами установки сервера отчётов:

```
IMAGE_STORAGE=docker.io/library
IMAGE_TAG=2022.2.22
NAMESPACE=fr-corporate-server
SECRET_VOLUME_NAME=corporate-volume-secret
IMAGE_REGISTRY_SECRET_NAME=storage_secret
SECRET_VOLUME_NAME=corporate-volume-secret
PULL_POLICY=Never
```

Описание переменных:

`IMAGE_STORAGE` - адрес хранилища образов сервисов сервера отчётов, Корпоративный Сервер использует локальное хранилище, наполнение образами которого описано в разделе [Подготовка образов](#).

`IMAGE_TAG` - версия сервера отчётов.

`NAMESPACE` - пространство имён Kubernetes, в котором будет выполняться сервер отчётов.

`SECRET_VOLUME_NAME` - имя тома, на котором будет храниться конфигурация и информация для авторизации.

`IMAGE_REGISTRY_SECRET_NAME` - имя записи, сохраняющей информацию для авторизации на сервере, хранящем образы сервисов сервере отчётов. `PULL_POLICY` - определяет как Kubernetes будет скачивать образы сервисов. Возможные значения:

- Always - всегда скачивать образы при старте сервиса;
- IfNotPresent - скачивать образ только если он не присутствует локально;
- Never - использовать заранее импортированные образы, никогда не скачивать их с внешнего сервера.

Обратите внимание: описанная конфигурация не скачивает образы из внешнего хранилища (`PULL_POLICY=Never`), эти переменные должны быть заданы во избежание ошибок при применении конфигурации. Перед установкой значения `PULL_POLICY` обратитесь к секции `fr-images` для ознакомления с настройками.

Настройка параметров сервера отчётов описана в документе [Конфигурация сервера отчётов](#). В этом документе описаны базовые параметры, определяющие режимы работы сервера отчётов.

Gateway - сервис обработки входящих запросов

Gateway это сервис, обрабатывающий все соединения к серверу отчётов. Он анализирует запросы и распределяет их между другими сервисами. Если сервис Gateway не работает или работает неправильно, то ни один из компонентов сервера отчётов работать не будет. Ingress конфигурация по умолчанию отправляет все внешние запросы именно этому сервису. В случае каких-либо проблем с сервисом начните анализ проблемы с просмотра логов сервиса `fr-gateway`.

```
#!/bin/sh

HTTPS_NODE_PORT=32222
IMAGE="{IMAGE_STORAGE}/fastreport-corporate-server-gateway:{IMAGE_TAG}"

cat <<EOF | kubectl apply -n $NAMESPACE -f -
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
```

```
kind: ClusterRole
metadata:
  name: fr-gateway
  namespace: $NAMESPACE
  labels:
    app.kubernetes.io/name: fr-gateway
rules:
- apiGroups: [""]
  resources:
    - services
    - endpoints
    - pods
  verbs: ["get", "list", "watch"]
```

```
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: fr-gateway
  namespace: $NAMESPACE
  labels:
    app.kubernetes.io/name: fr-gateway
```

```
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: fr-gateway
  namespace: $NAMESPACE
  labels:
    app.kubernetes.io/name: fr-gateway
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: fr-gateway
subjects:
- kind: ServiceAccount
  name: fr-gateway
  namespace: $NAMESPACE
```

```
---
apiVersion: v1
kind: Service
metadata:
  name: fr-gateway
  namespace: $NAMESPACE
spec:
  type: NodePort
  selector:
    app: fr-gateway
  ports:
    - name: http
      protocol: TCP
      port: 80
      nodePort: 32223
    - name: https
      protocol: TCP
      port: 5005
      nodePort: $HTTPS_NODE_PORT
```

```
---
apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2
kind: Deployment
metadata:
  name: fr-gateway
  namespace: $NAMESPACE
```

```
spec:
  replicas: 1
  selector:
    matchLabels:
      app: fr-gateway
  strategy:
    type: Recreate
  template:
    metadata:
      namespace: $NAMESPACE
      labels:
        app: fr-gateway
    spec:
      serviceAccountName: fr-gateway
      containers:
        - image: $IMAGE
          imagePullPolicy: $PULL_POLICY
          name: fr-gateway
          #env:
          # # Use secret in real usage
          #- name: MYSQL_ROOT_PASSWORD
          # value: password
          ports:
            - containerPort: 80
              name: fr-gateway
          volumeMounts:
            - name: config-volume
              mountPath: /app/appsettings.Production.json
              subPath: appsettings.Production.json
            - name: secret-volume
              mountPath: /etc/cert
          imagePullSecrets:
            - name: $IMAGE_REGISTRY_SECRET_NAME
          volumes:
            - name: config-volume
          configMap:
            name: fast-report-config
            items:
              - key: appsettings.Production.json
                path: appsettings.Production.json
            - name: secret-volume
          secret:
            secretName: $SECRET_VOLUME_NAME
EOF
```

Установка планировщика сервера отчётов

```
IMAGE="${IMAGE_STORAGE}/fastreport-corporate-server-scheduler:${IMAGE_TAG}"
```

```
cat <<EOF | kubectl apply -n $NAMESPACE -f -
apiVersion: v1
kind: Service
metadata:
  name: fr-scheduler
  namespace: $NAMESPACE
spec:
  type: ClusterIP
  ports:
    - port: 80
  selector:
    app: fr-scheduler
EOF
```

```
cat <<EOF | kubectl apply -n $NAMESPACE -f -
apiVersion: apps/v1
kind: Deployment
metadata:
  name: fr-scheduler
  namespace: $NAMESPACE
spec:
  replicas: 1
  selector:
    matchLabels:
      app: fr-scheduler
  strategy:
    type: Recreate
  template:
    metadata:
      namespace: $NAMESPACE
    labels:
      app: fr-scheduler
    spec:
      containers:
        - image: $IMAGE
          imagePullPolicy: $PULL_POLICY
          name: fr-scheduler
          #env:
          # # Use secret in real usage
          #- name: MYSQL_ROOT_PASSWORD
          # value: password
          ports:
            - containerPort: 80
              name: fr-scheduler
          volumeMounts:
            - name: config-volume
              mountPath: /app/appsettings.Production.json
            - name: secret-volume
              mountPath: /etc/cert
          imagePullSecrets:
            - name: $IMAGE_REGISTRY_SECRET_NAME
          volumes:
            - name: config-volume
              configMap:
                name: fast-report-config
            - name: secret-volume
              secret:
                secretName: $SECRET_VOLUME_NAME
EOF
```

Static Preview - сервис просмотра отчётов

```
#!/bin/sh

IMAGE="${IMAGE_STORAGE}/fastreport-corporate-server-static-preview:${IMAGE_TAG}"

cat <<EOF | kubectl apply -n $NAMESPACE -f -
apiVersion: v1
kind: Service
metadata:
  name: fr-s-preview
  namespace: $NAMESPACE
spec:
  type: ClusterIP
  ports:
    - port: 80
  selector:
    app: fr-s-preview
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: fr-s-preview
  namespace: $NAMESPACE
spec:
  replicas: 1
  selector:
    matchLabels:
      app: fr-s-preview
  strategy:
    type: Recreate
  template:
    metadata:
      namespace: $NAMESPACE
      labels:
        app: fr-s-preview
    spec:
      containers:
        - image: $IMAGE
          resources:
            limits:
              memory: 200Mi
            requests:
              memory: 200Mi
          imagePullPolicy: $PULL_POLICY
          name: fr-s-preview
      #env:
      # # Use secret in real usage
      #- name: MYSQL_ROOT_PASSWORD
      # value: password
      ports:
        - containerPort: 80
          name: fr-s-preview
      imagePullSecrets:
        - name: $IMAGE_REGISTRY_SECRET_NAME
EOF
```

Сервис Homer

```
#!/bin/sh

IMAGE="${IMAGE_STORAGE}/fastreport-corporate-server-homer:${IMAGE_TAG}"

SECRET_VOLUME_NAME=corporate-volume-secret

cat <<EOF | kubectl apply -n $NAMESPACE -f -
apiVersion: v1
kind: Service
metadata:
  name: fr-homer
  namespace: $NAMESPACE
spec:
  type: ClusterIP
  ports:
  - port: 80
  selector:
    app: fr-homer
---
apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2
kind: Deployment
metadata:
  name: fr-homer
  namespace: $NAMESPACE
spec:
  replicas: 1
  selector:
    matchLabels:
      app: fr-homer
  strategy:
    type: Recreate
  template:
    metadata:
      namespace: $NAMESPACE
    labels:
      app: fr-homer
    spec:
      containers:
      - image: $IMAGE
        imagePullPolicy: $PULL_POLICY
        name: fr-homer
        #env:
        # # Use secret in real usage
        #- name: MYSQL_ROOT_PASSWORD
        # value: password
        ports:
        - containerPort: 80
          name: fr-homer
      imagePullSecrets:
      - name: $IMAGE_REGISTRY_SECRET_NAME
EOF
```

Сервис Backend

```
#!/bin/sh

IMAGE="${IMAGE_STORAGE}/fastreport-corporate-server-backend:${IMAGE_TAG}"

cat <<EOF | kubectl apply -n $NAMESPACE -f -
apiVersion: v1
kind: Service
metadata:
  name: fr-backend
```

```

namespace: $NAMESPACE
spec:
  type: ClusterIP
  ports:
  - port: 80
  selector:
    app: fr-backend
---
apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2
kind: Deployment
metadata:
  name: fr-backend
  namespace: $NAMESPACE
spec:
  replicas: 3
  selector:
    matchLabels:
      app: fr-backend
  strategy:
    type: Recreate
  template:
    metadata:
      namespace: $NAMESPACE
    labels:
      app: fr-backend
    spec:
      containers:
      - image: $IMAGE
        resources:
          limits:
            memory: 400Mi
          requests:
            memory: 400Mi
        imagePullPolicy: $PULL_POLICY
        name: fr-backend
        env:
        - name: Serilog_Using_0
          value: FastReport.Cloud.Base.Mvc
        ports:
        - containerPort: 80
          name: fr-backend
        volumeMounts:
        - name: config-volume
          mountPath: /app/appsettings.Production.json
          subPath: appsettings.Production.json
        imagePullSecrets:
        - name: $IMAGE_REGISTRY_SECRET_NAME
      volumes:
      - name: config-volume
        configMap:
          name: fast-report-config
          items:
          - key: appsettings.Production.json
            path: appsettings.Production.json
EOF

```

Сервис Designer - онлайн дизайнер отчётов

```

#!/bin/sh

IMAGE="${IMAGE_STORAGE}/fastreport-corporate-server-designer:${IMAGE_TAG}"

cat <<EOF | kubectl apply -n $NAMESPACE -f -
apiVersion: v1

```

```

kind: Service
metadata:
  name: fr-designer
  namespace: $NAMESPACE
spec:
  type: ClusterIP
  ports:
    - port: 80
  selector:
    app: fr-designer
---
apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2
kind: Deployment
metadata:
  name: fr-designer
  namespace: $NAMESPACE
spec:
  replicas: 1
  selector:
    matchLabels:
      app: fr-designer
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: fr-designer
      namespace: $NAMESPACE
    spec:
      containers:
        - image: $IMAGE
          imagePullPolicy: $PULL_POLICY
          name: fr-designer
          #env:
          # # Use secret in real usage
          #- name: MYSQL_ROOT_PASSWORD
          # value: password
          ports:
            - containerPort: 80
              name: fr-designer
          volumeMounts:
            - name: config-volume
              mountPath: /app/appsettings.Production.json
          subPath: appsettings.Production.json
            - name: secret-volume
              mountPath: /etc/cert
          imagePullSecrets:
            - name: $IMAGE_REGISTRY_SECRET_NAME
          volumes:
            - name: config-volume
              configMap:
                name: fast-report-config
                items:
                  - key: appsettings.Production.json
                    path: appsettings.Production.json
            - name: secret-volume
              secret:
                secretName: $SECRET_VOLUME_NAME
EOF

```

Сервис шрифтов

```
#!/bin/sh
```



```
IMAGE="${IMAGE_STORAGE}/fastreport-corporate-server-fonts:${IMAGE_TAG}"
```

```
cat <<EOF | kubectl apply -n $NAMESPACE -f -
```

```
apiVersion: v1
```

```
kind: Service
```

```
metadata:
```

```
  name: fr-fonts
```

```
  namespace: $NAMESPACE
```

```
spec:
```

```
  type: ClusterIP
```

```
  ports:
```

```
  - port: 80
```

```
  selector:
```

```
    app: fr-fonts
```

```
---
```

```
apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2
```

```
kind: Deployment
```

```
metadata:
```

```
  name: fr-fonts
```

```
  namespace: $NAMESPACE
```

```
spec:
```

```
  replicas: 1
```

```
  selector:
```

```
    matchLabels:
```

```
      app: fr-fonts
```

```
  strategy:
```

```
    type: Recreate
```

```
  template:
```

```
    metadata:
```

```
      labels:
```

```
        app: fr-fonts
```

```
      namespace: $NAMESPACE
```

```
    spec:
```

```
      containers:
```

```
      - image: $IMAGE
```

```
        resources:
```

```
          limits:
```

```
            memory: 200Mi
```

```
          requests:
```

```
            memory: 200Mi
```

```
        imagePullPolicy: $PULL_POLICY
```

```
        name: fr-fonts
```

```
        env:
```

```
        - name: Serilog_Using_0
```

```
          value: FastReport.Cloud.MvcExtentions
```

```
        ports:
```

```
        - containerPort: 80
```

```
          name: fr-fonts
```

```
        volumeMounts:
```

```
        - name: config-volume
```

```
          mountPath: /app/appsettings.Production.json
```

```
          subPath: appsettings.Production.json
```

```
    imagePullSecrets:
```

```
    - name: $IMAGE_REGISTRY_SECRET_NAME
```

```
    volumes:
```

```
    - name: config-volume
```

```
      configMap:
```

```
        name: fast-report-config
```

```
        items:
```

```
        - key: appsettings.Production.json
```

```
          path: appsettings.Production.json
```

```
EOF
```

Сервис облачного приложения

```
#!/bin/sh

IMAGE="{IMAGE_STORAGE}/fastreport-corporate-server-app:{IMAGE_TAG}"

cat <<EOF | kubectl apply -n $NAMESPACE -f -
apiVersion: v1
kind: Service
metadata:
  name: fr-app
  namespace: $NAMESPACE
spec:
  type: ClusterIP
  ports:
    - port: 80
  selector:
    app: fr-app
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: fr-app
  namespace: $NAMESPACE
spec:
  replicas: 1
  selector:
    matchLabels:
      app: fr-app
  strategy:
    type: Recreate
  template:
    metadata:
      namespace: $NAMESPACE
      labels:
        app: fr-app
    spec:
      containers:
        - image: $IMAGE
          imagePullPolicy: $PULL_POLICY
          name: fr-app
          #env:
          # # Use secret in real usage
          #- name: MYSQL_ROOT_PASSWORD
          # value: password
          ports:
            - containerPort: 80
              name: fr-app
          imagePullSecrets:
            - name: $IMAGE_REGISTRY_SECRET_NAME
EOF
```

Сервис построения отчётов

```
#!/bin/sh

IMAGE="${IMAGE_STORAGE}/fastreport-corporate-server-workercore:${IMAGE_TAG}"

cat <<EOF | kubectl apply -n $NAMESPACE -f -
apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2
kind: Deployment
metadata:
  name: fr-workercore
  namespace: $NAMESPACE
spec:
  replicas: 4
  selector:
    matchLabels:
      app: fr-workercore
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: fr-workercore
        namespace: $NAMESPACE
    spec:
      containers:
      - image: $IMAGE
        imagePullPolicy: $PULL_POLICY
        name: fr-workercore
        volumeMounts:
        - name: config-volume
          mountPath: /app/appsettings.Production.json
          subPath: appsettings.Production.json
        - name: secret-volume
          mountPath: /etc/cert
      imagePullSecrets:
      - name: $IMAGE_REGISTRY_SECRET_NAME
    volumes:
    - name: config-volume
      configMap:
        name: fast-report-config
        items:
        - key: appsettings.Production.json
          path: appsettings.Production.json
    - name: secret-volume
      secret:
        secretName: $SECRET_VOLUME_NAME
EOF
```

После выполнения всех вышеописанных скриптов установка сервера отчётов закончена. Для того чтобы проверить успешность установки и начать работу, зайдите с помощью браузера на сервер отчётов, используя доменное имя, указанное в переменной HOST при настройке nginx-ingress и в появившемся окне введите следующие данные:

Имя пользователя:

Пароль:

Приятной работы!

Конфигурация

Конфигурация МоиОтчеты Корпоративный Сервер осуществляется через файлы `appsettings.json`, которые лежат в директории приложения. Эти файлы по умолчанию уже имеют ряд свойств, которые можно переопределить одним из трех способов:

1. Через `appsettings.{Environment}.json`, по умолчанию переменная `Environment` имеет значение `Production`, поэтому достаточно изменить содержание файла `appsettings.Production.json`. Регистр имеет значение!

Для смены значения используйте переменные среды `ASPNETCORE_ENVIRONMENT` и `DOTNET_ENVIRONMENT`.

```
export ASPNETCORE_ENVIRONMENT=Production
export DOTNET_ENVIRONMENT=Production
```

2. Через переменные среды. Подробности описаны ниже.

Приоритезация загрузки конфига `appsettings.json` -> `appsettings.{Environment}.json` -> переменные среды. Это означает, что конфигурация будет загружена слева направо, иначе говоря, те файлы что находятся правее перезаписывают конфиг предыдущих.

Описание формата `appsettings.json`

В каждом из разделов будет приведён пример конфигурации и описательная часть: ключ -> значение.

```
{
  "Logging": [
    {
      "Name": "ToEmail"
    }
  ]
}
```

Этот ключ можно использовать как переменную среды:

```
export Logging_0_Name=ToEmail
```

Здесь запись `Logging_0_Name` означает обращение к разделу `Logging`, элементу под индексом `0` и свойство `Name`.

Раздел `Kestrel`

Позволяет конфигурировать http сервер для прослушивания определённого порта или использования сертификата. Пример конфига:

```

{
  "Kestrel":{
    "Endpoints":{
      "Http":{
        "Url":"http://localhost:5000"
      },
      "Https":{
        "Url":"https://localhost:5001",
        "Certificate":{
          "Path":"<путь к файлу .pfx>",
          "Password":"<пароль к сертификату>"
        }
      }
    }
  }
}

```

КЛЮЧ	ТИП	ОПИСАНИЕ
Kestrel_Endpoints_Http_Url	string (uri)	Точка доступа для прослушивания http.
Kestrel_Endpoints_Https_Url	string (uri)	Точка доступа для прослушивания https.
Kestrel_Endpoints_Https_Certificate_Path	string (local path)	Путь к файлу .pfx сертификата для https трафика.
Kestrel_Endpoints_Https_Certificate_Password	string	Пароль для доступа к .pfx файлу.

Раздел Auth

Позволяет конфигурировать процесс аутентификации и авторизации.

```

{
  "Auth":{
    "ClientId":"<клиент OpenID>",
    "Scopes":"OpenID и другие области видимости",
    "Authority":"https://id.быстрыеотчеты.рф",
    "Audience":"https://облако.моиотчеты.рф",
    "Secret":"<секрет для клиента OpenID>",
    "UseApiKeys":true,
    "UseOpenId":true,
    "UseLocal":false,
    "RsaXml":"<XML-кодированный объект>",
    "UserInfoEndpoint":"https://example.com/userinfo",
    "TokenEndpoint":"https://example.com/token",
    "MetadataEndpoint":"https://example.com/.well-known/openid-configuration"
  }
}

```

КЛЮЧ	ТИП	ОПИСАНИЕ
Auth_ClientId	string	Уникальный идентификатор на сервере аутентификации по протоколу openid.
Auth_Scopes	string	Список областей клиента по протоколу openid.

КЛЮЧ	ТИП	ОПИСАНИЕ
Auth_Authority	string (uri)	Точка доступа Authority у которой будет запрашиваться токен доступа по протоколу openid.
Auth_Audience	string (uri)	Точка доступа Audience для которой будет запрашиваться токен доступа по протоколу openid.
Auth_Secret	string	Секретный токен для подтверждения авторизации по протоколу openid.
Auth_UseApiKeys	boolean	Включает или отключает возможность авторизации через ключи доступа.
Auth_UseOpenid	boolean	Включает или отключает возможность авторизации по протоколу openid, обратите внимание, клиент должен быть настроен для авторизации через code flow.
Auth_UseLocal	boolean	[Скоро будет]
Auth_RsaXml	string (xml)	RSA сериализованный в XML. Например, <pre><RSAKeyValue><Modulus> {base64}</Modulus><Exponent> {base64}</Exponent><P> {base64}</P><Q> {base64}</Q><DP> {base64}</DP><DQ> {base64}</DQ><InverseQ> {base64}</InverseQ><D> {base64}</D></RSAKeyValue></pre>
Auth_UserInfoEndpoint	string (uri)	Точка доступа, через которую будут запрашиваться данные пользователя по протоколу openid.
Auth_TokenEndpoint	string (uri)	Точка доступа, через которую будет запрашиваться токен пользователя по протоколу openid.
Auth_MetadataEndpoint	string (uri)	Точка доступа well-known с метаданными для openid.

Раздел MainConfig

Позволяет конфигурировать основную конфигурацию МоиОтчеты Корпоративный Сервер.

```
{
  "MainConfig": {
    "Database": {
      "ConnectionString": "<строка подключения>"
      "DatabaseName": "<имя базы данных>"
      "ExportsCollectionName": "Exports"
      "ReportsCollectionName": "Reports",
      "TemplatesCollectionName": "Templates",
      "TemplateFoldersCollectionName": "TemplateFolders",
      "ReportFoldersCollectionName": "ReportFolders",
      "ExportFoldersCollectionName": "ExportFolders",
      "UsersCollectionName": "Users",
      "SubscriptionPlansCollectionName": "SubscriptionPlans",
      "SubscriptionsCollectionName": "Subscriptions",
      "GroupsCollectionName": "Groups",
      "DataSourceCollectionName": "DataSources",
      "GridFSFilesCollectionName": "fs.files",
      "MigrationsCollectionName": "Migrations",
      "SubscriptionInvitesCollectionName": "SubscriptionInvites"
    },
    "Server": {
```

```

"Title": "<Заголовок в приложении>",
"Copyright": "<Краткое описание авторского права>",
"LogoLink": "<путь к логотипу, отображаемому в панели пользователя>",
"LastSlaUpdate": "2022-11-28",
"CorporateServerMode": false,
"SlaLink": "<Ссылка на пользовательское соглашение>",
"FirstStepsVideoLink": "https://rutube.ru/play/embed/f64e80d2fa8e2d73b30c47483fa2b59e",
>AboutLink": "https://быстрыеотчеты.рф/products/corporate-server",
"HomePageLink": "быстрыеотчеты.рф/ru",
"AuthServerName": "ID Быстрые отчеты",
"UsersPerWorkSpace": 0
},
"Pagination":{
  "MaxEntries":120
},
"Rabbit":{
  "Host":"my-rabbit-server.com",
  "Port":5672,
  "UserName":"<имя пользователя>",
  "Password":"<пароль пользователя>",
  "QueueName":"ReportProcessQueue",
  "DirectExchangeName":"DirectEx",
  "AlternateExchangeName":"AExchange",
  "UnroutedQueueName":"Default",
  "Capacity":1
},
"SecurityAdvisor":{
  "RestrictUnsafe":true,
  "RestrictUnmanaged":true,
  "RestrictExtern":true,
  "RestrictAsync":true,
  "RestrictTypeOf":true,
  "Whitelist":[
    "^\\w+:\\s+FastReport.*$",
    "^\\w+\\.\\.\\.System\\.Math.*$",
    "^\\w+\\.\\.\\.System\\.DateTime.*$",
    "^\\w+\\.\\.\\.System\\.Environment.*$"
  ],
  "Blacklist":[
    "^Method\\.\\.\\.\\.\\.GetType\\(\\|\\)$",
    "^\\w+\\.\\.\\.System\\.IO.*$",
    "^\\w+\\.\\.\\.FastReport\\.Utils\\.Config.*$",
    "^\\w+\\.\\.\\.System\\.Environment.*$",
    "^\\w+\\.\\.\\.System\\.Diagnostics.*$",
    "^\\w+\\.\\.\\.System\\.Reflection.*$",
    "^\\w+\\.\\.\\.System\\.Net.*$",
    "^\\w+\\.\\.\\.System\\.Threading.*$"
  ]
},
"License": "",
"SmtpServer": {
  "EnableSsl": true,
  "Server": "smtp.s.com",
  "Port": 587,
  "Username": "--",
  "From": "--",
  "Password": "--"
},
"Tasks": {
  "Attempts": 3
},
"Frontend": {
  "Mixins": {
    "Head": "",
    "Body": ""
  }
}

```

```

}
},
"Rfc2898CryptorSettings": {
  "Salt": "",
  "Password": "",
  "IV": ""
},
"DataSourcesConfig": {
  "XmlConfig": { "CommandTimeout": 30 },
  "JsonConfig": { "CommandTimeout": 30 },
  "CsvConfig": { "CommandTimeout": 30 },
  "MySqlConfig": { "CommandTimeout": 30 },
  "PostgreSqlConfig": { "CommandTimeout": 30 },
  "MsSqlConfig": { "CommandTimeout": 30 },
  "OracleDbConfig": { "CommandTimeout": 30 },
  "FirebirdConfig": { "CommandTimeout": 30 },
  "MongoDbConfig": { "CommandTimeout": 30 },
  "ClickHouseConfig": { "CommandTimeout": 30 }
}
}
}
}

```

КЛЮЧ	ТИП	ОПИСАНИЕ
MainConfig__Database__ConnectionString	string	Строка для подключения к базе данных MongoDB.
MainConfig__Database__DatabaseName	string	Название базы данных сервера MongoDB.
MainConfig__Database__ExportsCollectionName	string	Название коллекции для хранения списка экспортов.
MainConfig__Database__ReportsCollectionName	string	Название коллекции для хранения списка отчётов.
MainConfig__Database__TemplatesCollectionName	string	Название коллекции для хранения списка шаблонов.
MainConfig__Database__TemplateFoldersCollectionName	string	Название коллекции для хранения древовидной структуры шаблонов.
MainConfig__Database__ReportFoldersCollectionName	string	Название коллекции для хранения древовидной структуры отчётов.
MainConfig__Database__ExportFoldersCollectionName	string	Название коллекции для хранения древовидной структуры экспортов.
MainConfig__Database__UsersCollectionName	string	Название коллекции для хранения списка пользователей.
MainConfig__Database__SubscriptionPlansCollectionName	string	Название коллекции для хранения списка планов подписки.

КЛЮЧ	ТИП	ОПИСАНИЕ
MainConfig__Database__SubscriptionsCollectionName	string	Название коллекции для хранения списка подписок.
MainConfig__Database__GroupsCollectionName	string	Название коллекции для хранения списка групп.
MainConfig__Database__DataSourceCollectionName	string	Название коллекции для хранения списка источников данных.
MainConfig__Database__GridFSFilesCollectionName	string	Название коллекции для хранения файлов.
MainConfig__Database__MigrationsCollectionName	string	Название коллекции для хранения списка применённых миграций.
MainConfig__Database__SubscriptionInvitesCollectionName	string	Название коллекции для хранения списка приглашений.
MainConfig__Server__Title	string	Имя для сервера, которое будет использовать пользовательское приложение в заголовке старницы.
MainConfig__Server__Copyright	string	Текст для информации об авторском праве (или любой другой информации, которая будет отображена внизу страницы пользовательского приложения).
MainConfig__Server__LastSLAUpdate	string	Дата последнего изменения пользовательского соглашения, если изменить эту дату на более новую, при открытии пользовательского приложения откроется диалог, в котором будет ссылка на обновлённые условия (из поля SlaLink).
MainConfig__Server__CorporateServerMode	boolean	Это поле изменяет поведение части API, например, если указано false, то в админской панели нельзя будет удалять рабочие пространства.
MainConfig__Server__FirstStepsVideoLink	string	Ссылка на видеоруководство к корпоративному серверу, которое появится при первом входе в пользовательское приложение или при нажатии на ? внизу страницы.
MainConfig__Server__AboutLink	string	Ссылка на информацию о продукте.

КЛЮЧ	ТИП	ОПИСАНИЕ
MainConfig_Server_HomePageLink	string	Ссылка на домашнюю страницу сайта продукта.
MainConfig_Server_AuthServerName	string	Имя для сервера авторизации, указанного в разделе AuthConfig, используемое в пользовательском приложении.
MainConfig_Server_UsersPerWorkSpace	integer	Максимальное количество пользователей в рабочем пространстве, задаваемое администратором.
MainConfig_Pagination_MaxEntries	integer	Максимальное количество элементов для пагинации при запросе списка по API.
MainConfig_Rabbit_Host	string	Адрес хоста для доступа к RabbitMQ.
MainConfig_Rabbit_Port	string	Порт для доступа к RabbitMQ.
MainConfig_Rabbit_UserName	string	Имя пользователя для доступа к RabbitMQ.
MainConfig_Rabbit_Password	string	Пароль для доступа к RabbitMQ.
MainConfig_Rabbit_QueueName	string	Название очереди, которая используется для базового имени конкретных очередей подписок пользователей.
MainConfig_Rabbit_DirectExchangeName	string	Название обменника RabbitMQ, который используется для направления сообщений на очереди подписок пользователей.
MainConfig_Rabbit_AlternateExchangeName	string	Название обменника RabbitMQ, который будет использоваться по умолчанию, когда для пользователя ещё не создана подписка.
MainConfig_Rabbit_UnroutedQueueName	string	Название очереди RabbitMQ, которая будет обрабатываться по умолчанию, когда для пользователя ещё не создана подписка.
MainConfig_Rabbit_Capacity	1	Не меняйте это значение! Задаёт максимальное количество параллельных построений отчётов на одном рабочем построителе.

КЛЮЧ	ТИП	ОПИСАНИЕ
MainConfig__SecurityAdvisor__RestrictUnsafe	string	Включает или отключает ключевое слово <code>unsafe</code> в скрипте.
MainConfig__SecurityAdvisor__RestrictUnmanaged	string	Включает или отключает ключевое слово <code>unmanaged</code> в скрипте.
MainConfig__SecurityAdvisor__RestrictExtern	string (uri)	Включает или отключает ключевое слово <code>extern</code> в скрипте.
MainConfig__SecurityAdvisor__RestrictAsync	string (uri)	Включает или отключает ключевое слово <code>async</code> в скрипте.
MainConfig__SecurityAdvisor__RestrictTypeOf	string	Включает или отключает ключевое слово <code>typeof</code> в скрипте.
MainConfig__SecurityAdvisor__Whitelist__0	boolean	Задаёт список API, которые можно использовать без предупреждений в скрипте отчёта. Число указывает порядковый номер в списке.
MainConfig__SecurityAdvisor__Blacklist__0	boolean	Задаёт список API, которые нельзя использовать в скрипте отчёта. Число указывает порядковый номер в списке.
MainConfig__License	string	Лицензионный ключ. Предоставляется вместе с продуктом.
MainConfig__SmtpServer__Server	string	Адрес почтового сервера
MainConfig__SmtpServer__Port	integer	Порт SMTP сервера
MainConfig__SmtpServer__Username	string	Имя пользователя SMTP сервера
MainConfig__SmtpServer__Password	string	Пароль пользователя SMTP сервера
MainConfig__SmtpServer__From	string	Почтовый адрес отправителя (отображается в письме)
MainConfig__Tasks__Attempts	integer	Количество попыток запуска задачи воркером
MainConfig__Frontend__Mixins__Head	string	Примесь, встраиваемая в header пользовательской панели (например, код аналитики)

КЛЮЧ	ТИП	ОПИСАНИЕ
MainConfig_Frontend_Mixins_Body	string	Примесь, встраиваемая в body пользовательской панели (например, код аналитики)
MainConfig_Rfc2898CryptorSettings_Salt	string	Соль криптографического алгоритма (используется для шифрования паролей)
MainConfig_Rfc2898CryptorSettings_Password	string	Пароль криптографического алгоритма
MainConfig_Rfc2898CryptorSettings_IV	string	Вектор криптографического алгоритма
MainConfig_InvariantLocale	string	Постоянная локализация. Работает независимо от языка браузера
MainConfig_ErrorHooks_RocketChat_BaseAddress	string	Базовый адрес всех вебхуков RocketChat
MainConfig_ErrorHooks_RocketChat_Path	string	Путь, по которому обращаться чтобы отправить сообщение
MainConfig_DataSourcesConfig_XmlConfig_CommandTimeout	integer	Время ожидания ответа от источника данных XML в секундах. Если не указать значение, будет использоваться стандартный таймаут для этого источника данных.
MainConfig_DataSourcesConfig_JsonConfig_CommandTimeout	integer	Время ожидания ответа от источника данных JSON в секундах. Если не указать значение, будет использоваться стандартный таймаут для этого источника данных.
MainConfig_DataSourcesConfig_CsvConfig_CommandTimeout	integer	Время ожидания ответа от источника данных CSV в секундах. Если не указать значение, будет использоваться стандартный таймаут для этого источника данных.
MainConfig_DataSourcesConfig_MySqlConfig_CommandTimeout	integer	Время ожидания ответа от источника данных MySQL в секундах. Если не указать значение, будет использоваться стандартный таймаут для этого источника данных.

КЛЮЧ	ТИП	ОПИСАНИЕ
MainConfig_DataSourcesConfig_PostgreSqlConfig_CommandTimeout	integer	Время ожидания ответа от источника данных PostgreSQL в секундах. Если не указать значение, будет использоваться стандартный таймаут для этого источника данных.
MainConfig_DataSourcesConfig_MsSqlConfig_CommandTimeout	integer	Время ожидания ответа от источника данных MS SQL в секундах. Если не указать значение, будет использоваться стандартный таймаут для этого источника данных.
MainConfig_DataSourcesConfig_OracleDbConfig_CommandTimeout	integer	Время ожидания ответа от источника данных Oracle DB в секундах. Если не указать значение, будет использоваться стандартный таймаут для этого источника данных.
MainConfig_DataSourcesConfig_FirebirdConfig_CommandTimeout	integer	Время ожидания ответа от источника данных Firebird в секундах. Если не указать значение, будет использоваться стандартный таймаут для этого источника данных.
MainConfig_DataSourcesConfig_MongoDbConfig_CommandTimeout	integer	Время ожидания ответа от источника данных Mongo DB в секундах. Если не указать значение, будет использоваться стандартный таймаут для этого источника данных.
MainConfig_DataSourcesConfig_ClickHouseConfig_CommandTimeout	integer	Время ожидания ответа от источника данных ClickHouse в секундах. Если не указать значение, будет использоваться стандартный таймаут для этого источника данных.

Раздел Gateway

Позволяет конфигурировать шлюз доступа МоиОтчеты Корпоративный Сервер.

```

{
  "Gateway":{
    "WhiteListForDisabled":{
      "<any_claim_name>":"<claim_value>"
    },
    "IsDisabled":false,
    "ExcludePaths":[
      "/account",
      "/disabled"
    ],
    "SignInPagePath":"/account/signin?r={0}",
    "DisabledPath":"/disabled?r={0}",
    "IsSignInRequired":false
  }
}

```

КЛЮЧ	ТИП	ОПИСАНИЕ
Gateway__WhiteListForDisabled__<any_claim_name>	string	Список утверждений пользователя в токене для доступа к выключенному МоиОтчеты Облако.
Gateway__IsDisabled	boolean	Включает или отключает доступ к МоиОтчеты Облако.
Gateway__ExcludePaths__0	string	Список путей, к которым можно получить доступ даже при выключенном доступе к МоиОтчеты Облако. Число указывает порядковый номер в списке.
Gateway__SignInPagePath	string	Строка для перенаправления к странице с формой входа.
Gateway__DisabledPath	string	Строка для перенаправления к странице с сообщением о технических работах или о выключении доступа.
Gateway__IsSignInRequired	boolean	Включает или отключает необходимость входа для пользователя для доступа к МоиОтчеты Облако.

Раздел `Services`

Позволяет конфигурировать список сервисов для маршрутизации шлюза.

```

{
  "Services":{
    "HealthCheckInterval":30,
    "Items":{
      "<name>":{
        "Urls":[
          "http://localhost:5555"
        ],
        "Scheme":"http",
        "Port":5555,
        "K8sServiceName":"fr-rp",
        "HostType":"WebApp",
        "PathBase":"/api/rp/swagger",
        "Namespace":"fr-cloud",
        "Type":"K8s",
        "PingPath":"/api/rp/v1/healthcheck",
        "IsSignInRequired":true,
        "Priority":10,
        "PingResponseCode":200,
        "LoadBalanceMode":"Random",
        "HealthCheckAttemptsNumber":3,
        "WhiteListClaims":{
          "<claim_name>":"<claim_value>"
        }
      }
    }
  }
}

```

КЛЮЧ	ТИП	ОПИСАНИЕ
Services_HealthCheckInterval	integer	Интервал для проверки работоспособности сервисов, задаётся в секундах.
Services_Items_<name>_Urls_0	string (url)	Список url адресов для доступа к статичным сервисам. Число указывает порядковый номер в списке.
Services_Items_<name>_Scheme	string	Схема доступа к сервису: http или https.
Services_Items_<name>_Port	integer	Порт доступа к сервису.
Services_Items_<name>_K8sServiceName	string	Название сервиса в Kubernetes.
Services_Items_<name>_HostType	string	Тип сервиса для обработки перенаправления шлюзом, может принимать значения: WebApp, API, External, Websocket.
Services_Items_<name>_PathBase	string	Базовый путь для сервиса.
Services_Items_<name>_Namespace	string	Пространство имён сервиса в Kubernetes.
Services_Items_<name>_Type	string	Тип сервиса для обработки доступа шлюзом; может принимать значения: Static, K8s.

КЛЮЧ	ТИП	ОПИСАНИЕ
<code>Services__Items__<name>__PingPath</code>	string	Часть строки запроса для получения информации о состоянии сервиса.
<code>Services__Items__<name>__IsSignInRequired</code>	string	Указывает необходимость авторизации перед получением доступа к сервису пользователям.
<code>Services__Items__<name>__Priority</code>	number	Приоритет сервиса перед другими; значение меньше значит сильнее.
<code>Services__Items__<name>__PingResponseCode</code>	integer	Статус код ответа, который будет ожидаться от healthcheck.
<code>Services__Items__<name>__LoadBalanceMode</code>	string	Указывает тип балансировки нагрузки для этого сервиса; может принимать одно из значений: <code>Random</code> , <code>AverageMetric</code> .
<code>Services__Items__<name>__HealthCheckAttemptsNumber</code>	integer	Количество попыток проверить состояния сервиса.
<code>Services__Items__<name>__WhiteListClaims__<claim_name></code>	string	Указывает утверждение у пользователя для получения доступа к сервису.

Вместо `<name>` следует использовать имя сервиса; список сервисов можно посмотреть в файле `appsettings.json`.

Раздел `Constants`

Позволяет устанавливать ограничения на размер тела запроса в приложении.

```
{
  "Constants": {
    "LimitsMaxRequestBodySize": 2097152000
  }
}
```

КЛЮЧ	ТИП	ОПИСАНИЕ
<code>Constants__LimitsMaxRequestBodySize</code>	long	Максимальный размер тела запроса. При превышении данного ограничения запрос будет отклонен.

Удаление МоиОтчеты Корпоративный Сервер из кластера

1. Удаление компонентов сервера отчётов из кластера:

```
kubectl delete -n fr-corporate-server deployment fr-designer
kubectl delete -n fr-corporate-server deployment fr-scheduler
kubectl delete -n fr-corporate-server deployment fr-workercore
kubectl delete -n fr-corporate-server deployment fr-gateway
kubectl delete -n fr-corporate-server deployment fr-app
kubectl delete -n fr-corporate-server deployment fr-backend
kubectl delete -n fr-corporate-server deployment fr-s-preview
kubectl delete -n fr-corporate-server deployment fr-fonts
kubectl delete -n fr-corporate-server deployment fr-homer
kubectl delete -n fr-corporate-server secret corporate-volume-secret
```

2. Удаление MongoDB.

Удаление MongoDB приведёт к удалению пользователей, групп, шаблонов отчётов, подготовленных отчётов и экспортированных отчётов.

```
kubectl delete -n mongo deployment fr-mongo-mongodb
```

Обратите внимание - в зависимости от настроек, вышеприведённая команда может не удалять дисковое пространство, арендованное у провайдеров облачных сервисов. Если вы арендуете кластер у облачного провайдера, воспользуйтесь web-интерфейсом провайдера для удаления Persistent Volumes сервера отчётов - mongo-storage.

3. Удаление остальных компонентов кластера необязательно, а в случае, если сервер был использован в рабочем кластере совместно со сторонними сервисами, то удаление этих компонентов может повредить инфраструктуру. Например, RabbitMQ может быть использован сторонними сервисами кластера.

4. Для полного удаления кластера и его компонентов используйте команду:

```
kubeadm reset
```

Разворачивание Корпоративного Сервера без среды Kubernetes

Помимо разворачивания в Kubernetes Корпоративный Сервер может быть запущен с помощью docker-compose.

Для этого необходимо выполнить следующие действия.

Скачиваем архив с docker-образами Корпоративного Сервера

Демо версию Корпоративного Сервера можно запросить в вашем личном кабинете в разделе поддержки. Если у вас уже есть полная версия Корпоративного Сервера, то в личном кабинете будет доступ к файлу лицензии.

Загружаем в локальное хранилище образы МоиОтчеты Корпоративный Сервер.

```
```sh
docker load -i frcs.tar.gz
```

## Указываем лицензию в файл `appsettings.Production.json`

Создаём файл с названием `appsettings.Production.json` и указываем в нём лицензионный ключ.

```
{
 "Auth": {
 "UseOpenId": false,
 "UseLocal": true
 },
 "MainConfig": {
 "InternalHeaders": {
 "S56nHMSzjQzXYx5KJJsU3cjU": "000000000000000000000001",
 "x2aHtuSsxFeYqE8xPTaxAnbH": "00000000000000000000000002",
 "QNpq2nyzvDNSCLtVhMJ9e8m": "00000000000000000000000003",
 "9MXgeFwLNjeUrjvw7N2aQv9F": "00000000000000000000000004"
 },
 "Frontend": {
 "Mixins": {
 "Head": "",
 "Body": ""
 },
 "InvariantLocale": ""
 },
 "License": "",
 "Server": {
 "Title": "МоиОтчеты Корпоративный Сервер",
 "CorporateServerMode": true
 },
 "Rabbit": {
 "Host": "rabbitmq",
 "Port": 5672,
 "UserName": "fastreports",
 "Password": "Qwerty!23456",
 "QueueName": "ReportProcessQueue",
 "DirectExchangeName": "DirectEx",
 "AlternateExchangeName": "AExchange",
 "UnroutedQueueName": "Default",
 "Capacity": 1
 },
 "Database": {
 "ConnectionString": "mongodb://fastreport:Qwerty!23456@mongo:27017/?
```

```

authSource=ReportStore&readPreference=primary&appName=MongoDB%20Compass&ssl=false&maxPoolSize=100&waitQ
ueueMultiple=100",
 "DatabaseName": "ReportStore"
}
},
"Gateway": {
 "BackendUrl": "http://fr-backend:80",
 "InternalKey": "QNpq2nyzvDNscBLtVhMJ9e8m",
 "SignInPagePath": "/account/signin?r={0}",
 "MaxConcurrentRequests": 200,
 "RequestQueueLimit": 5000
},
"Serilog": {
 "MinimumLevel": {
 "Default": "Debug"
 }
},
"Services": {
 "Items": {
 "OnlineDesigner": {
 "Type": "Static",
 "Urls": [
 "http://fr-onlinedesigner:80"
]
 },
 "Backend": {
 "Type": "Static",
 "Urls": [
 "http://fr-backend:80"
]
 },
 "FrontendApp": {
 "Type": "Static",
 "Urls": [
 "http://fr-app:80"
]
 },
 "Fonts": {
 "Type": "Static",
 "Urls": [
 "http://fr-fonts:80"
]
 },
 "StaticPreviewApp": {
 "Type": "Static",
 "Urls": [
 "http://fr-staticpreview:80"
]
 },
 "HomerApp": {
 "Type": "Static",
 "Urls": [
 "http://fr-homer:80"
]
 }
 }
},
"Designer": {
 "BackendUrl": "http://fr-backend:80",
 "InternalKey": "x2aHtuSsxFeYqE8xPTaxAnbH"
},
"WorkerCore": {
 "BackendUrl": "http://fr-backend:80",
 "InternalKey": "S56nHMSzjQzXYx5KJjsU3cjU"
},
"Cache": {

```

```
 scheduler: {
 "BackendUrl": "http://fr-backend:80",
 "InternalKey": "9MXgeFwLNjeUrjvw7N2aQv9F"
 }
 }
}
```

Находим поле License и указываем ключ:

```
...
"License": "{ключ из файла лицензии}"
...
```

## Настраиваем базу данных

Создаём файл mongo-init.js со следующим содержимым:

```
db = db.getSiblingDB('admin')

db.auth('admin', 'Qwerty!23456')

db = db.getSiblingDB('ReportStore')

db.createUser({
 user: "fastreport",
 pwd: "Qwerty!23456",
 roles: [{ role: "readWrite", db: "ReportStore" }]
});
```

## Файл docker compose

Создаём файл с именем docker-compose.yml со следующим содержимым:

```
version: "3.9"
services:
 mongo:
 image: mongo:5.0
 volumes:
 - mongo:/data/db
 - ./mongo-init.js:/docker-entrypoint-initdb.d/mongo-init.js:ro
 restart: always
 environment:
 - MONGO_INITDB_ROOT_USERNAME=admin
 - MONGO_INITDB_ROOT_PASSWORD=Qwerty!23456
 - MONGO_INITDB_DATABASE=ReportStore
 networks:
 - fr-cs
 rabbitmq:
 image: bitnami/rabbitmq:3.9
 volumes:
 - rabbitmq:/bitnami
 restart: always
 networks:
 - fr-cs
 environment:
 - RABBITMQ_USERNAME=fastreports
 - RABBITMQ_PASSWORD=Qwerty!23456
 fr-backend:
 image: fastreport-corporate-server-backend:2022.2.22
 restart: always
 volumes:
 - ./appsettings.Production.json:/app/appsettings.Production.json:ro
 networks:
```

```
networks:
 - fr-cs
fr-gateway:
 image: fastreport-corporate-server-gateway:2022.2.22
 ports:
 - 8080:80
 restart: always
environment:
- SomeEnviromentSettings=value
volumes:
 - ./appsettings.Production.json:/app/appsettings.Production.json:ro
networks:
 - fr-cs
fr-fonts:
 image: fastreport-corporate-server-fonts:2022.2.22
 restart: always
 volumes:
 - ./appsettings.Production.json:/app/appsettings.Production.json:ro
 networks:
 - fr-cs
fr-app:
 image: fastreport-corporate-server-app:2022.2.22
 restart: always
 volumes:
 - ./appsettings.Production.json:/app/appsettings.Production.json:ro
 networks:
 - fr-cs
fr-staticpreview:
 image: fastreport-corporate-server-static-preview:2022.2.22
 restart: always
 volumes:
 - ./appsettings.Production.json:/app/appsettings.Production.json:ro
 networks:
 - fr-cs
fr-onlinedesigner:
 image: fastreport-corporate-server-designer:2022.2.22
 restart: always
 volumes:
 - ./appsettings.Production.json:/app/appsettings.Production.json:ro
 networks:
 - fr-cs
fr-homer:
 image: fastreport-corporate-server-homer:2022.2.22
 restart: always
 volumes:
 - ./appsettings.Production.json:/app/appsettings.Production.json:ro
 networks:
 - fr-cs
fr-workercore:
 image: fastreport-corporate-server-workercore:2022.2.22
 restart: always
 volumes:
 - ./appsettings.Production.json:/app/appsettings.Production.json:ro
 networks:
 - fr-cs
 deploy:
 mode: replicated
 replicas: 2
 endpoint_mode: vip
fr-scheduler:
 image: fastreport-corporate-server-scheduler:2022.2.22
 restart: always
 volumes:
 - ./appsettings.Production.json:/app/appsettings.Production.json:ro
 networks:
 - fr-cs
```

```
networks:
 fr-cs:
 driver: bridge
volumes:
 mongo:
 rabbitmq:
```

## Запускаем Корпоративный Сервер

По итогу в докере должны быть загружены скачанные образы, а в папке лежать файлы `appsettings.Production.json`, `docker-compose.yml` и `mongo-init.js`.

Запускаем сервер:

```
docker-compose up --build -d
```

## Вход в систему

После выполнения всех вышеописанных скриптов установка сервера отчётов закончена. Для того чтобы проверить успешность установки и начать работу, зайдите с помощью браузера на сервер отчётов по адресу [localhost:8080](http://localhost:8080) и в появившемся окне введите следующие данные:

Имя пользователя:

Пароль:

Чтобы попасть в панель администратора перейдите по ссылке [localhost:8080/homer](http://localhost:8080/homer)

## Выключаем Корпоративный Сервер

```
docker-compose down
```