MTender Pilot

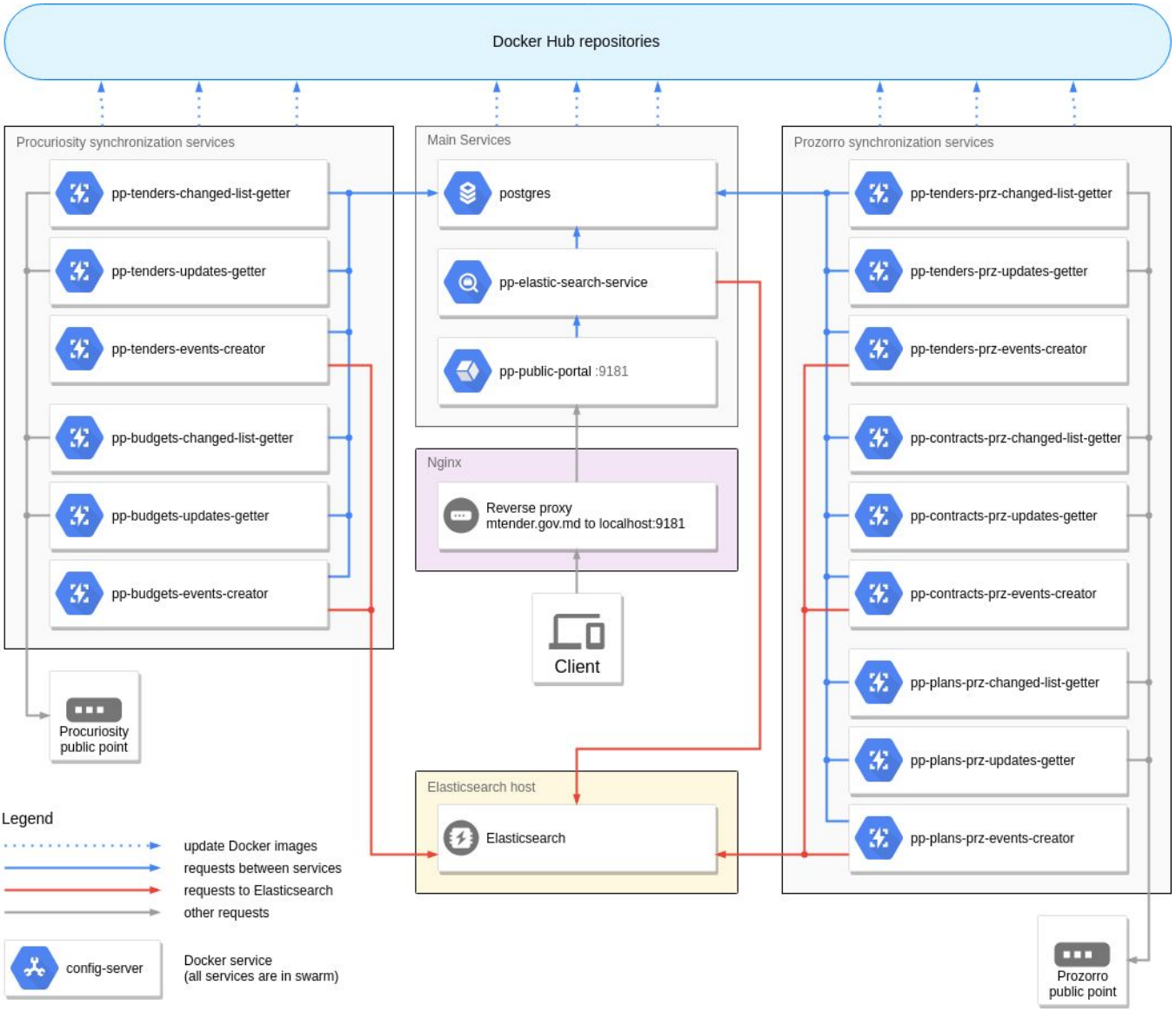
Public Portal Deployment Guide

Ver 1.0

Prepared under technical cooperation for EBRD UNCITRAL Public Procurement Initiative
by uStudio LLC Kiev, Ukraine

Table of contents

MTender components scheme



Docker services

Order	*Service name*	*External port*	*Description*
1	postgres	-	PostgreSQL database
2	pp-tenders-changed-list-getter	-	Get changed Procuriosity tenders id list
3	pp-tenders-updates-getter	-	Get Procuriosity tender JSON data
4	pp-tenders-events-creator	-	Refresh Procuriosity tender into DB and index to Elasticsearch
2	pp-budgets-changed-list-getter	-	Get changed Procuriosity budgets id list
3	pp-budgets-updates-getter	-	Get Procuriosity budget JSON data
4	pp-budgets-events-creator	-	Refresh Procuriosity budget into DB and index to Elasticsearch
2	pp-tenders-prz-changed-list-getter	-	Get changed Prozorro tenders id list
3	pp-tenders-prz-updates-getter	-	Get Prozorro tender JSON data
4	pp-tenders-prz-events-creator	-	Refresh Prozorro tender into DB and index to Elasticsearch
2	pp-contracts-prz-changed-list-getter	-	Get changed Prozorro contracts id list
3	pp-contracts-prz-updates-getter	-	Get Prozorro contract JSON data
4	pp-contracts-prz-events-creator	-	Refresh Prozorro contract into DB and index to Elasticsearch
2	pp-plans-prz-changed-list-getter	-	Get changed Prozorro plans id list
3	pp-plans-prz-updates-getter	-	Get Prozorro plan JSON data
4	pp-plans-prz-events-creator	-	Refresh Prozorro plan into DB and index to Elasticsearch
5	pp-elastic-search-service	-	Elasticsearch JSON API
6	pp-public-portal	9181	MTender website

***** - deployment order for synchronization services groups does not matter

Deployment scripts

```
#create file example
touch deploy.config

#grant execution permission example
chmod +x deploy.config

#edit file example
nano deploy.config

#execute file example
./deploy.sh
```

The following steps describe the files you need to create and grant permissions to execute:

1. Create config file [deploy.config]

```
ADMIN_USER=******
ADMIN_PASSWORD=******
ADMIN_EMAIL=*****@mt*****.***.*d

PUBLIC_PORTAL_PORT=9181

DB_TYPE=pgsql
DB_PORT=5432
DB_PUB_PORT=5555
DB_USERNAME=postgres
DB_PASSWORD=******
DB_HOST=postgres
DB_TENDERS_NAME=pp_tenders
DB_BUDGETS_NAME=pp_budgets
DB_PLANS_NAME=pp_plans
DB_CONTRACTS_NAME=pp_contracts
DB_PP_NAME=pp_public_portal

TENDERS_URL=https://public.mtender.gov.md/tenders
BUDGETS_URL=https://public.mtender.gov.md/budgets
TENDERS_PRZ_URL=https://public.api.mepps.openprocurement.net/api/2.3/tenders
PLANS_PRZ_URL=https://public.api.mepps.openprocurement.net/api/2.3/plans
CONTRACTS_PRZ_URL=https://public.api.mepps.openprocurement.net/api/2.3/contracts

ELASTIC_INDEXING=true
ELASTIC_TENDERS_INDEX=pp_tenders
ELASTIC_TENDERS_TYPE=docs
ELASTIC_BUDGETS_INDEX=pp_budgets
ELASTIC_BUDGETS_TYPE=docs
ELASTIC_PLANS_INDEX=pp_plans
ELASTIC_PLANS_TYPE=docs
ELASTIC_CONTRACTS_INDEX=pp_contracts
ELASTIC_CONTRACTS_TYPE=docs
ELASTIC_CPV_INDEX=pp_cpv
ELASTIC_CPV_TYPE=docs
ELASTIC_PAGE_SIZE=25
ELASTIC_URL=http://10.41.10.3:9200

SYNCHRONIZATION_SERVICES="pp-tenders-prz-changed-list-getter
                          pp-tenders-prz-updates-getter
                          pp-tenders-prz-events-creator
                          pp-tenders-changed-list-getter
                          pp-tenders-updates-getter
                          pp-tenders-events-creator
                          pp-budgets-changed-list-getter
                          pp-budgets-updates-getter
                          pp-budgets-events-creator
                          pp-plans-prz-changed-list-getter
                          pp-plans-prz-updates-getter
                          pp-plans-prz-events-creator
                          pp-contracts-prz-changed-list-getter
                          pp-contracts-prz-updates-getter
                          pp-contracts-prz-events-creator"

SLEEP_DELAY_INTERVAL=120
SLEEP_ERROR_INTERVAL=300
NET_NAME=ocds-network
```

2. Create Docker network script [network.sh]

```
#### include config ####
. deploy.config

#### swarm init ####
docker swarm init

#### network init ####
docker network create \
--driver=overlay \
--attachable \
${NET_NAME}
```

3. Create PostgreSQL service and databases script [postgresql_service.sh]

```
#### include config ####
. deploy.config

#### postgres ####
SERVICE_NAME=postgres;
echo "Creating service: ${SERVICE_NAME}";
docker volume create pgdata
docker service create  \
--mode=replicated \
--label inner-ports=${DB_PORT} \
--limit-memory=200m \
--name ${SERVICE_NAME} \
--hostname=${SERVICE_NAME} \
--network=${NET_NAME} \
--mount type=bind,source=/etc/localtime,target=/etc/localtime,readonly \
--mount source=pgdata,destination=/var/lib/postgresql/data \
--env POSTGRES_PASSWORD=${DB_PASSWORD} \
--with-registry-auth \
--update-order="start-first" \
--publish published=${DB_PUB_PORT},target=${DB_PORT} \
postgres:10

#### wait service ####
sleep 10

#### create databases ####
docker exec $(docker ps -q -f name=${SERVICE_NAME}) createdb -h localhost -p ${DB_PORT} -U postgres
${DB_TENDERS_NAME}
docker exec $(docker ps -q -f name=${SERVICE_NAME}) createdb -h localhost -p ${DB_PORT} -U postgres
${DB_BUDGETS_NAME}
docker exec $(docker ps -q -f name=${SERVICE_NAME}) createdb -h localhost -p ${DB_PORT} -U postgres
${DB_PLANS_NAME}
docker exec $(docker ps -q -f name=${SERVICE_NAME}) createdb -h localhost -p ${DB_PORT} -U postgres
${DB_CONTRACTS_NAME}
docker exec $(docker ps -q -f name=${SERVICE_NAME}) createdb -h localhost -p ${DB_PORT} -U postgres
${DB_PP_NAME}
```

4. Create synchronization services script [sync_service.sh]

```
#### include config ####
. deploy.config

#### create synchronization services ####
for SERVICE_NAME in ${SYNCHRONIZATION_SERVICES}
do
    echo "Creating service: ${SERVICE_NAME}"
    docker service create \
        --limit-cpu=0.5 \
        --limit-memory=100m \
        --hostname=${SERVICE_NAME} \
        --name=${SERVICE_NAME} \
        --network=${NET_NAME} \
        --update-order="start-first" \
        --mount type=bind,source=/etc/localtime,target=/etc/localtime,readonly \
        --mount type=bind,source=/etc/hosts,destination=/etc/hosts \
        --env SERVICENAME=${SERVICE_NAME} \
        --env environments_DB_TYPE=${DB_TYPE} \
        --env environments_DB_PORT=${DB_PORT} \
        --env environments_DB_USERNAME=${DB_USERNAME} \
        --env environments_DB_PASSWORD=${DB_PASSWORD} \
        --env environments_DB_HOST=${DB_HOST} \
        --env environments_DB_BUDGETS_NAME=${DB_BUDGETS_NAME} \
        --env environments_DB_TENDERS_NAME=${DB_TENDERS_NAME} \
        --env environments_DB_PLANS_NAME=${DB_PLANS_NAME} \
        --env environments_DB_CONTRACTS_NAME=${DB_CONTRACTS_NAME} \
        --env environments_ELASTIC_INDEXING=${ELASTIC_INDEXING} \
        --env environments_ELASTIC_BUDGETS_INDEX=${ELASTIC_BUDGETS_INDEX} \
        --env environments_ELASTIC_BUDGETS_TYPE=${ELASTIC_BUDGETS_TYPE} \
        --env environments_ELASTIC_TENDERS_INDEX=${ELASTIC_TENDERS_INDEX} \
        --env environments_ELASTIC_TENDERS_TYPE=${ELASTIC_TENDERS_TYPE} \
        --env environments_ELASTIC_PLANS_INDEX=${ELASTIC_PLANS_INDEX} \
        --env environments_ELASTIC_PLANS_TYPE=${ELASTIC_PLANS_TYPE} \
        --env environments_ELASTIC_CONTRACTS_INDEX=${ELASTIC_CONTRACTS_INDEX} \
        --env environments_ELASTIC_CONTRACTS_TYPE=${ELASTIC_PLANS_TYPE} \
        --env environments_ELASTIC_CPV_INDEX=${ELASTIC_CPV_INDEX} \
        --env environments_ELASTIC_CPV_TYPE=${ELASTIC_CPV_TYPE} \
        --env environments_ELASTIC_URL=${ELASTIC_URL} \
        --env environments_TENDERS_URL=${TENDERS_URL} \
        --env environments_BUDGETS_URL=${BUDGETS_URL} \
        --env environments_TENDERS_PRZ_URL=${TENDERS_PRZ_URL} \
        --env environments_PLANS_PRZ_URL=${PLANS_PRZ_URL} \
        --env environments_CONTRACTS_PRZ_URL=${CONTRACTS_PRZ_URL} \
        --env environments_SLEEP_DELAY_INTERVAL=${SLEEP_DELAY_INTERVAL} \
            --env environments_SLEEP_ERROR_INTERVAL=${SLEEP_ERROR_INTERVAL} \
        eprocurementsystems/pp-sync-cdb-service
done
```

5. Set mapping in Elasticsearch and import CPV script [elasticsearch.sh]

```
#### include config ####
. deploy.config

#### elasticsearch mapping and CPV ####
services=($SYNCHRONIZATION_SERVICES)
echo "Prepare mapping";
docker exec -w /var/www/service $(docker ps -q -f name=${services[0]}) php yii mapping-elastic/all
echo "Import CPV";
docker exec -w /var/www/service $(docker ps -q -f name=${services[0]}) php yii cpv/import
```

6. Create Elasticsearch search service script [search_service.sh]

```
#### include config ####
. deploy.config

##==== elastic-search-service ======
SERVICE_NAME=pp-elastic-search-service;
echo "Creating service: ${SERVICE_NAME}";
docker service create \
--limit-cpu=0.5 \
--limit-memory=100m \
--hostname=${SERVICE_NAME} \
--name=${SERVICE_NAME} \
--network=${NET_NAME} \
--env SERVICENAME=${SERVICE_NAME} \
--env environments_DB_HOST=${DB_HOST} \
--env environments_DB_PORT=${DB_PORT} \
--env environments_DB_USERNAME=${DB_USERNAME} \
--env environments_DB_PASSWORD=${DB_PASSWORD} \
--env environments_DB_BUDGETS_NAME=${DB_BUDGETS_NAME} \
--env environments_DB_TENDERS_NAME=${DB_TENDERS_NAME} \
--env environments_DB_PLANS_NAME=${DB_PLANS_NAME} \
--env environments_ELASTIC_BUDGETS_INDEX=${ELASTIC_BUDGETS_INDEX} \
--env environments_ELASTIC_BUDGETS_TYPE=${ELASTIC_BUDGETS_TYPE} \
--env environments_ELASTIC_TENDERS_INDEX=${ELASTIC_TENDERS_INDEX} \
--env environments_ELASTIC_TENDERS_TYPE=${ELASTIC_TENDERS_TYPE} \
--env environments_ELASTIC_PLANS_INDEX=${ELASTIC_PLANS_INDEX} \
--env environments_ELASTIC_PLANS_TYPE=${ELASTIC_PLANS_TYPE} \
--env environments_ELASTIC_CONTRACTS_INDEX=${ELASTIC_CONTRACTS_INDEX} \
--env environments_ELASTIC_CONTRACTS_TYPE=${ELASTIC_PLANS_TYPE} \
--env environments_ELASTIC_CPV_INDEX=${ELASTIC_CPV_INDEX} \
--env environments_ELASTIC_CPV_TYPE=${ELASTIC_CPV_TYPE} \
--env environments_ELASTIC_URL=${ELASTIC_URL} \
--env environments_ELASTIC_PAGE_SIZE=${ELASTIC_PAGE_SIZE} \
--env environments_SLEEP_ERROR_INTERVAL=${SLEEP_ERROR_INTERVAL} \
--update-order="start-first" \
--mount type=bind,source=/etc/localtime,target=/etc/localtime,readonly \
--mount type=bind,source=/etc/hosts,destination=/etc/hosts \
eprocurementsystems/pp-elastic-search-service
```

7. Create website service script [portal_service.sh]

```
#### include config ####
. deploy.config

##==== public-portal ======
SERVICE_NAME=pp-public-portal;
echo "Creating service: ${SERVICE_NAME}";
docker volume create uploads
docker service create \
--limit-cpu=0.5 \
--limit-memory=100m \
--hostname=${SERVICE_NAME} \
--name=${SERVICE_NAME} \
--network=${NET_NAME} \
--env environments_ELASTIC_SERVICE_URL=http://pp-elastic-search-service \
--env environments_DB_TYPE=${DB_TYPE} \
--env environments_DB_PORT=${DB_PORT} \
--env environments_DB_USERNAME=${DB_USERNAME} \
--env environments_DB_PASSWORD=${DB_PASSWORD} \
--env environments_DB_HOST=${DB_HOST} \
--env environments_DB_NAME=${DB_PP_NAME} \
--env environments_ADMIN_USER=${ADMIN_USER} \
--env environments_ADMIN_PASSWORD=${ADMIN_PASSWORD} \
--env environments_ADMIN_EMAIL=${ADMIN_EMAIL} \
--update-order="start-first" \
--publish published=${PUBLIC_PORTAL_PORT},target=80 \
--mount type=bind,source=/etc/localtime,target=/etc/localtime,readonly \
--mount type=bind,source=/etc/hosts,destination=/etc/hosts \
--mount source=uploads,destination=/var/www/service/frontend/web/uploads \
eprocurementsystems/pp-portal
```

Service update scripts

1. Update pp-public-portal

```
docker pull eprocurementsystems/pp-portal:latest
docker service update --image eprocurementsystems/pp-portal:latest pp-public-portal
```

2. Update pp-elastic-search-service

```
docker pull eprocurementsystems/pp-elastic-search-service:latest
docker service update --image eprocurementsystems/pp-elastic-search-service:latest
pp-elastic-search-service
```

3. Update synchronization services

```
docker pull eprocurementsystems/pp-sync-cdb-service:latest

docker service update --image eprocurementsystems/pp-sync-cdb-service:latest pp-elastic-search-settings

docker service update --image eprocurementsystems/pp-sync-cdb-service:latest
pp-tenders-prz-changed-list-getter
docker service update --image eprocurementsystems/pp-sync-cdb-service:latest
pp-tenders-prz-updates-getter
docker service update --image eprocurementsystems/pp-sync-cdb-service:latest
pp-tenders-prz-events-creator

docker service update --image eprocurementsystems/pp-sync-cdb-service:latest
pp-tenders-changed-list-getter
docker service update --image eprocurementsystems/pp-sync-cdb-service:latest pp-tenders-updates-getter
docker service update --image eprocurementsystems/pp-sync-cdb-service:latest pp-tenders-events-creator

docker service update --image eprocurementsystems/pp-sync-cdb-service:latest
pp-budgets-changed-list-getter
docker service update --image eprocurementsystems/pp-sync-cdb-service:latest pp-budgets-updates-getter
docker service update --image eprocurementsystems/pp-sync-cdb-service:latest pp-budgets-events-creator

docker service update --image eprocurementsystems/pp-sync-cdb-service:latest
pp-plans-prz-changed-list-getter
docker service update --image eprocurementsystems/pp-sync-cdb-service:latest
pp-plans-prz-updates-getter
docker service update --image eprocurementsystems/pp-sync-cdb-service:latest
pp-plans-prz-events-creator

docker service update --image eprocurementsystems/pp-sync-cdb-service:latest
pp-contracts-prz-changed-list-getter
docker service update --image eprocurementsystems/pp-sync-cdb-service:latest
pp-contracts-prz-updates-getter
docker service update --image eprocurementsystems/pp-sync-cdb-service:latest
pp-contracts-prz-events-creator
```