

## Инструкция по установке системы ДОРИС Платформа - Единая Платформа Управления Транспортными Системами

### 1. Базовое системное программное обеспечение

Сервера приложений:

- Операционная система Ubuntu версия 18.04 или более поздняя
- Система контейнеризации Docker 19.03 или более поздняя
- Сборщик контейнеров Docker-compose 1.17 или более поздняя

Сервера баз данных:

- Операционная система FreeBSD 12.1 или более поздняя
- СУБД Postgres 12.0 или более поздняя
- Расширение PostGIS 3.0 или более поздняя
- Расширение postgresql-contrib версии соответствующей Postgres
- СУБД Redis 5.0 или более поздняя

Рабочие станции необходимо следующее программное обеспечение:

- браузер Google Chrome 51.0 и выше или Mozilla FireFox версии 14 и выше;

### 2. Техническое обеспечение

Используемые для эксплуатации платформы технические средства (персональные компьютеры, сервера, системы виртуализации, периферийные устройства) должны быть совместимы между собой и поддерживать сетевой протокол TCP/IP.

Для работы платформы используются компьютеры 64-разрядной архитектуры AMD/Intel:

- с операционной системой FreeBSD для серверов баз данных
- с операционной системой Ubuntu для серверов приложений
- операционной системой Windows/Linux/MacOS для рабочих станций

Минимальные технические характеристики клиентских компьютеров:

- тактовая частота процессора не менее 2 GHz;
- оперативная память не менее 4 GB;
- свободное дисковое пространство не менее 20 GB.

Рекомендуется использовать клиентские компьютеры с объемом оперативной памяти (RAM) от 8 GB.

Минимальные технические характеристики серверного оборудования:

- система виртуализации vmware ESXi 6.5 или Microsoft Hyper-V Server 2019
- каждый сервер баз данных
  - виртуальных ядер не менее 8
  - оперативная память не менее 32 GB
  - дисковое пространство не менее 240 GB
- каждый сервер приложений
  - виртуальных ядер не менее 4
  - оперативная память не менее 12 GB
  - дисковое пространство не менее 160 GB

Рекомендуется иметь полутора кратный запас количества ядер и размера оперативной памяти для серверов.

### 3. Установка сервера баз данных

- Установка СУБД Postgres, выполняется от пользователя root (или через команду sudo)
  - `pkg install postgresql12-server`
  - `pkg install redis`
  - `vi /usr/local/db/data12/pg_hba.conf`  
указать IP-адреса клиента - серверов приложений, пользователей и режим авторизации
  - `vi /usr/local/db/data12/postgresql.conf`  
`max_connections = 300`  
`listen_addresses = '*'`
  - `vi /etc/rc.conf`  
`redis_enable="YES"`  
`postgresql_class="postgres"`  
`postgresql_enable="YES"`
  - `initdb`
  - `service postgresql start`
  - `service redis start`

### 4. Установка сервера приложений

4.1. Установить Docker по официальной инструкции <https://docs.docker.com/engine/install/ubuntu/>

4.2. Установить docker-compose

```
sudo apt-get install docker-compose
```

4.3. создать директорию для настройки сервера приложения

```
mkdir docker && cd docker
```

4.4. Создать файл docker-compose.yml

```
nano docker-compose.yml
```

```
version: '3'
```

```
services:
```

```
front:
```

```
image: you-registry.local/neoteh/rtcc.website2:${PROJECT}-latest
```

```
restart: unless-stopped
```

```
tty: true
```

```
ports:
```

```
- 80:80
```

```
networks:
```

```
- app-bridge
```

```
back-main:
```

```
image: you-registry.local/its-backend/main-${PROJECT}:latest
```

```
restart: unless-stopped
```

```
tty: true
```

```
ports:
```

```
- 10100:8080
```

```
- 6001:6001
```

```
volumes:
```

```
- ${DATA_PATH_STORAGE}/app-main/storage:/var/www/storage
```

```
links:
```

```
- rabbitmq
```

networks:

- app-bridge

back-transport-passenger:

image: you-registry.local/its-backend/transport-core-kursk-passager

restart: unless-stopped

tty: true

ports:

- 10082:8080

volumes:

- \${DATA\_PATH\_STORAGE}/app-transport-passenger/storage:/var/www/storage

links:

- rabbitmq

networks:

- app-bridge

back-transport-service:

image: you-registry.local/its-backend/transport-core-kursk-special

restart: unless-stopped

tty: true

ports:

- 10083:8080

volumes:

- \${DATA\_PATH\_STORAGE}/app-transport-service/storage:/var/www/storage

links:

- rabbitmq

networks:

- app-bridge

back-eco:

image: you-registry.local/its-backend/eco

#you-registry.local/its-backend/eco-\${PROJECT}:latest

restart: unless-stopped

tty: true

ports:

- 10081:8080

volumes:

- \${DATA\_PATH\_STORAGE}/app-eco/storage:/var/www/storage

links:

- rabbitmq

networks:

- app-bridge

back-video:

image: you-registry.local/its-backend/video-\${PROJECT}:latest

restart: unless-stopped

tty: true

ports:

- 10084:8080

volumes:

- \${DATA\_PATH\_STORAGE}/app-video/storage:/var/www/storage

links:

- rabbitmq

networks:

- app-bridge

back-telemetry:

image: you-registry.local/its-backend/telemetry/kursk-telemetry-transport-`{PROJECT}`:latest  
restart: unless-stopped  
tty: true  
ports:

- 10085:8080

volumes:

- `{DATA_PATH_STORAGE}/app-kursk-telemetry-transport/storage:/var/www/storage`

links:

- rabbitmq

networks:

- app-bridge

back-storage:

image: you-registry.local/its-backend/storage  
restart: unless-stopped  
tty: true  
ports:

- 10087:8080

volumes:

- `{DATA_PATH_STORAGE}/app-storage/storage:/var/www/storage`

links:

- rabbitmq

networks:

- app-bridge

back-road-works:

image: you-registry.local/its-backend/road-works-`{PROJECT}`:latest  
restart: unless-stopped  
tty: true  
ports:

- 10089:8080

volumes:

- `{DATA_PATH_STORAGE}/app-road-works/storage:/var/www/storage`

links:

- rabbitmq

networks:

- app-bridge

rabbitmq:

image: rabbitmq:3.8.9-management  
volumes:

- /etc/localtime:/etc/localtime:ro
- `{DATA_PATH_HOST}/rabbitmq:/var/lib/rabbitmq`

ports:

- `{RABBITMQ_NODE_HOST_PORT}:5672"`
- `{RABBITMQ_MANAGEMENT_HTTP_HOST_PORT}:15672"`
- `{RABBITMQ_MANAGEMENT_HTTPS_HOST_PORT}:15671"`

privileged: true  
environment:  
- RABBITMQ\_DEFAULT\_USER=\${RABBITMQ\_DEFAULT\_USER}  
- RABBITMQ\_DEFAULT\_PASS=\${RABBITMQ\_DEFAULT\_PASS}  
networks:  
- app-bridge

clickhouse:  
container\_name: clickhouse  
image: yandex/clickhouse-server:20  
ports:  
- 8123:8123  
volumes:  
- '\${DATA\_PATH\_HOST}/clickhouse:/var/lib/clickhouse'  
privileged: true  
restart: unless-stopped  
networks:  
- app-bridge

proxykafka:  
image: you-registry.local/its-backend/proxy-from-kafka-to-rabbitmq-kursk:latest  
restart: unless-stopped  
tty: true  
links:  
- rabbitmq  
networks:  
- app-bridge

egts-spec:  
image: you-registry.local/go/base-egts-formattwo:v2  
restart: unless-stopped  
tty: true  
ports:  
- 6000:6000  
links:  
- rabbitmq  
networks:  
- app-bridge

egts-pass:  
image: you-registry.local/go/base-egts-formattwo:v2  
restart: unless-stopped  
tty: true  
ports:  
- 6300:6300  
links:  
- rabbitmq  
networks:  
- app-bridge

networks:  
app-bridge:

#### 4.5. Создать файл настройки .env

```
nano .env  
DATA_PATH_HOST=./data-docker  
DATA_PATH_STORAGE=./storage-all  
VOLUMES_DRIVER=local
```

```
PROJECT=YouNameProject
```

```
### RABBITMQ ###  
RABBITMQ_ERLANG_COOKIE=its  
RABBITMQ_NODE_HOST_PORT=5672  
RABBITMQ_MANAGEMENT_HTTP_HOST_PORT=15672  
RABBITMQ_MANAGEMENT_HTTPS_HOST_PORT=15671  
RABBITMQ_DEFAULT_USER=its  
RABBITMQ_DEFAULT_PASS=password
```

```
CONFIG_PATH=./config
```

#### 4.6. создать вспомогательные директории для каждого микросервиса

```
mkdir -p storage-all/app-main/storage/app/  
mkdir -p storage-all/app-main/storage/logs/  
mkdir -p storage-all/app-main/storage/framework/cache/data/  
mkdir -p storage-all/app-main/storage/framework/sessions/  
mkdir -p storage-all/app-main/storage/framework/testing/  
mkdir -p storage-all/app-main/storage/framework/views/
```

```
mkdir -p storage-all/app-road-works/storage/app/  
mkdir -p storage-all/app-road-works/storage/logs/  
mkdir -p storage-all/app-road-works/storage/framework/cache/data/  
mkdir -p storage-all/app-road-works/storage/framework/sessions/  
mkdir -p storage-all/app-road-works/storage/framework/testing/  
mkdir -p storage-all/app-road-works/storage/framework/views/
```

```
mkdir -p storage-all/app-transport-passenger/storage/app/  
mkdir -p storage-all/app-transport-passenger/storage/logs/  
mkdir -p storage-all/app-transport-passenger/storage/framework/cache/data/  
mkdir -p storage-all/app-transport-passenger/storage/framework/sessions/  
mkdir -p storage-all/app-transport-passenger/storage/framework/testing/  
mkdir -p storage-all/app-transport-passenger/storage/framework/views/
```

```
mkdir -p storage-all/app-transport-service/storage/app/  
mkdir -p storage-all/app-transport-service/storage/logs/  
mkdir -p storage-all/app-transport-service/storage/framework/cache/data/  
mkdir -p storage-all/app-transport-service/storage/framework/sessions/  
mkdir -p storage-all/app-transport-service/storage/framework/testing/  
mkdir -p storage-all/app-transport-service/storage/framework/views/
```

```
mkdir -p storage-all/app-storage/storage/app/  
mkdir -p storage-all/app-storage/storage/logs/  
mkdir -p storage-all/app-storage/storage/framework/cache/data/  
mkdir -p storage-all/app-storage/storage/framework/sessions/  
mkdir -p storage-all/app-storage/storage/framework/testing/
```

```
mkdir -p storage-all/app-storage/storage/framework/views/
```

```
mkdir -p storage-all/app-video/storage/app/
```

```
mkdir -p storage-all/app-video/storage/logs/
```

```
mkdir -p storage-all/app-video/storage/framework/cache/data/
```

```
mkdir -p storage-all/app-video/storage/framework/sessions/
```

```
mkdir -p storage-all/app-video/storage/framework/testing/
```

```
mkdir -p storage-all/app-video/storage/framework/views/
```

```
mkdir -p storage-all/app-eco/storage/app/
```

```
mkdir -p storage-all/app-eco/storage/logs/
```

```
mkdir -p storage-all/app-eco/storage/framework/cache/data/
```

```
mkdir -p storage-all/app-eco/storage/framework/sessions/
```

```
mkdir -p storage-all/app-eco/storage/framework/testing/
```

```
mkdir -p storage-all/app-eco/storage/framework/views/
```

#### 4.7. Запустить сервер приложений

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