Инструкция по установке и запуску

Платформа контроля складских операций (WES)

Версия 1.0.0 от 04.03.2026

---

1. Общие сведения

1.1. Назначение документа

Настоящая инструкция содержит сведения, необходимые для установки и первичного запуска программного обеспечения «Платформа контроля складских операций» (далее — WES, Платформа) на объекте заказчика. Документ адресован системным администраторам и инженерам, выполняющим развёртывание Платформы.

1.2. Принцип развёртывания

WES поставляется в виде набора Docker-контейнеров (OCI-совместимых образов) и развёртывается средствами Docker Engine и Docker Compose. Контейнерная модель обеспечивает воспроизводимость окружения, изоляцию зависимостей и переносимость между поддерживаемыми операционными системами. Дополнительная компиляция или ручная сборка компонентов на стороне заказчика не требуется.

1.3. Состав поставки

В состав поставки Платформы входят: — контейнерный образ WES Server (серверная часть); — контейнерный образ Map Editor (веб-клиент); — контейнерный образ СУБД PostgreSQL либо инструкции по подключению существующей внешней СУБД; — файл `docker-compose.yml` с конфигурацией развёртывания; — шаблон файла переменных окружения `.env.example`; — настоящая инструкция, руководство пользователя и иные эксплуатационные документы.

---

2. Требования к системе

2.1. Аппаратные требования (хост Docker)

Минимальные требования: — процессор — 2 ядра; — оперативная память — 4 ГБ; — дисковое пространство — 20 ГБ SSD; — сеть — 100 Мбит/с; — архитектура — x86-64 (linux/amd64).

Рекомендуемые требования: — процессор — 4 и более ядер; — оперативная память — 8 ГБ и более; — дисковое пространство — 50 ГБ и более SSD; — сеть — 1 Гбит/с.

Конкретный объём дискового пространства зависит от планируемой длительности хранения журналов событий и истории заданий. Для длительной эксплуатации с большим объёмом операций рекомендуется выделить дополнительное пространство под базу данных.

2.2. Программное обеспечение хоста

Для развёртывания Платформы на сервере должны быть установлены: — операционная система Linux (см. перечень совместимых ОС в п. 2.3); — Docker Engine версии 24.0 или выше; — Docker Compose версии 2.20 или выше (как Compose Plugin либо отдельный пакет).

Дополнительно для администрирования удобно иметь: — клиент `ssh` (для удалённого доступа); — текстовый редактор (`nano`, `vim` и т. п.); — утилиту `curl` или `wget` для проверки сетевых интерфейсов.

2.3. Совместимость с операционными системами

WES совместим со всеми операционными системами, обеспечивающими работу Docker Engine. В первую очередь поддерживаются: — Astra Linux Special Edition 1.7 и выше; — РЕД ОС 7.3 и 8.0; — Альт (ALT Linux) 10 и 11; — Роса СО и SX; — ОСь 2.0 и выше; — Ubuntu Server 20.04 и выше; — Debian 11 и выше; — РОСА «Хром» Сервер 12 и выше.

Контейнеры WES запускаются на перечисленных ОС без модификаций.

2.4. Требования к клиентским рабочим станциям

На рабочих местах пользователей требуется только современный веб-браузер: — Chromium (Google Chrome, Microsoft Edge, Яндекс.Браузер, Atom и др.) версии 90 и выше; — Mozilla Firefox версии 90 и выше.

Должна быть включена поддержка JavaScript и WebSocket. Рекомендуемое разрешение экрана — 1920×1080. Дополнительная установка программ на клиентские рабочие станции не требуется.

2.5. Сетевые требования

Между компонентами Платформы и внешними системами должны быть открыты следующие сетевые каналы: — TCP 8081 — программный интерфейс WES Server (HTTP API и WebSocket); доступ требуется клиентскому браузеру, WMS и (при необходимости) сторонним RCS; — TCP 8080 — веб-интерфейс Map Editor; доступ требуется клиентскому браузеру; — TCP 5432 — соединение с СУБД PostgreSQL; открывается только при использовании внешней (неконтейнерной) СУБД. При работе с контейнером PostgreSQL соединение проходит по внутренней сети Docker и наружу не публикуется.

Конкретные номера портов могут быть изменены через файл переменных окружения (см. раздел 4).

---

3. Подготовка среды

3.1. Установка Docker Engine

Для установки Docker Engine следует руководствоваться официальной документацией для соответствующей операционной системы. Ниже приведены общие команды для типовых случаев.

Для Ubuntu и Debian:

```
sudo apt update
sudo apt install -y ca-certificates curl gnupg
```

```
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | \
    sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] \
    https://download.docker.com/linux/ubuntu $(. /etc/os-release; echo "$VERSION_CODENAME")
stable" | \
    sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt update
sudo apt install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-
plugin
```

Для Astra Linux, РЕД ОС, Альт и других российских дистрибутивов следует использовать рекомендации поставщика дистрибутива по установке Docker Engine. В большинстве случаев Docker доступен в штатных репозиториях.

После установки необходимо включить и запустить службу Docker:

```
sudo systemctl enable docker
sudo systemctl start docker
```

Проверка корректности установки:

```
docker --version
docker compose version
docker run --rm hello-world
```

3.2. Настройка пользователя

Для удобства администрирования рекомендуется добавить эксплуатирующего пользователя в группу `docker`, чтобы выполнять команды Docker без `sudo`:

```
sudo usermod -aG docker $USER
```

После выполнения команды необходимо завершить и заново начать сеанс пользователя.

3.3. Подготовка каталога установки

Создайте каталог для размещения файлов Платформы и перейдите в него:

```
sudo mkdir -p /opt/wes
sudo chown $USER:$USER /opt/wes
cd /opt/wes
```

В дальнейшем все файлы конфигурации, данные и журналы Платформы будут размещаться в этом каталоге.

---

4. Развёртывание Платформы

4.1. Получение поставки

Скопируйте предоставленный поставщиком архив с конфигурацией Платформы в каталог `/opt/wes` и распакуйте его. В каталоге должны появиться: — файл `docker-compose.yml`; — файл

`.env.example`; — подкаталоги для размещения томов данных и конфигурации (создаются автоматически при первом запуске).

Контейнерные образы могут поставляться двумя способами: — загрузка с указанного поставщиком реестра контейнеров; — импорт из локальных архивов формата `.tar`, поставленных вместе с дистрибутивом.

В случае импорта из архивов выполните для каждого образа:

```
docker load -i <имя-архива>.tar
```

4.2. Настройка переменных окружения

Создайте рабочий файл переменных окружения на основе шаблона:

```
cp .env.example .env
```

Откройте файл `.env` текстовым редактором и заполните параметры. Ниже перечислены ключевые группы параметров.

Параметры подключения к базе данных: — строка подключения к PostgreSQL (с указанием пользователя, пароля, адреса, порта и имени базы); — имя пользователя СУБД, пароль, имя базы (для первичного создания базы средствами Платформы).

Параметры сетевых интерфейсов: — порт HTTP API и WebSocket (по умолчанию 8081); — порт веб-интерфейса (по умолчанию 8080); — разрешённые источники CORS (адреса фронтенда, WMS и иных интеграций).

Параметры интеграции с WMS: — включение и адрес внешней WMS; — реквизиты доступа для двустороннего обмена.

Параметры взаимодействия с парком роботов: — режим работы со встроенной RCS WES либо со сторонней RCS вендора; — адреса и параметры соединения с RCS вендора (при использовании).

Прочие параметры: — уровень детализации журнала событий; — часовой пояс системы.

После сохранения файла `.env` убедитесь, что доступ к нему ограничен:

```
chmod 600 .env
```

4.3. Запуск Платформы

Запустите все компоненты Платформы единой командой:

```
docker compose up -d
```

Команда автоматически: — загрузит необходимые образы (если они ещё не были импортированы); — создаст внутреннюю изолированную сеть для компонентов; — создаст постоянные тома для данных СУБД и файлов конфигурации; — запустит контейнеры WES Server, Map Editor и (при использовании) PostgreSQL.

Проверка состояния:

```
docker compose ps
docker compose logs -f
```

Все контейнеры должны находиться в состоянии `running` или `healthy`. Для выхода из режима просмотра журналов нажмите Ctrl+C.

---

5. Инициализация базы данных

5.1. Автоматическая инициализация

При первом запуске WES Server автоматически выполняет инициализацию схемы базы данных и применение всех актуальных миграций. Никаких дополнительных действий со стороны администратора, как правило, не требуется.

Признаком успешной инициализации является переход WES Server в состояние `healthy` (виден в выводе `docker compose ps`). При наличии ошибок их описание выводится в журнале контейнера WES Server.

5.2. Использование внешней СУБД

При использовании внешней (не контейнерной) СУБД PostgreSQL администратор должен предварительно:

1. Создать базу данных и пользователя с правами на чтение и запись:

   ```
   CREATE DATABASE wes;
   CREATE USER wes_user WITH ENCRYPTED PASSWORD '<пароль>';
   GRANT ALL PRIVILEGES ON DATABASE wes TO wes_user;
   ```

2. Указать строку подключения в файле `.env` и убедиться, что контейнер WES Server имеет сетевой доступ к серверу СУБД.

3. Выполнить инициализацию: WES Server применит миграции при первом запуске.

5.3. Проверка состояния миграций

Текущее состояние миграций можно посмотреть в журнале контейнера WES Server при запуске. При корректной инициализации в журнале будет отмечено успешное применение всех миграций до целевой версии.

---

6. Первичная настройка

6.1. Загрузка карты склада

При первой эксплуатации Платформы необходимо подготовить карту склада. Доступны два варианта: — создание карты средствами интерфейса Map Editor (рекомендуется для новых установок); — импорт карты из ранее сохранённого файла (используется при переносе с предыдущей установки или при подготовке карты на отдельном стенде).

Подробное описание работы с картой приведено в руководстве пользователя.

6.2. Регистрация роботов

После загрузки карты администратор регистрирует в Платформе все роботы парка. Для каждого робота указываются: уникальный идентификатор, исходное положение (при необходимости) и параметры взаимодействия с системой управления роботами (встроенной либо сторонней).

6.3. Настройка интеграции с WMS

Интеграция с внешней системой управления складом (WMS) является обязательным шагом первичной настройки: WMS является источником производственных заданий, и без подключённой WMS WES не получает задания для исполнения.

Администратор настраивает: — параметры подключения WMS к программному интерфейсу WES (адрес, реквизиты доступа); — правила преобразования заказов WMS в задания Платформы (при необходимости); — проверяет двусторонний обмен (приём заданий, отправка статусов выполнения).

Подробное описание программного интерфейса для интеграции с WMS приведено в отдельном документе технической документации. Если на момент развёртывания WMS ещё не готова к интеграции, для предварительной проверки работоспособности Платформы допускается создание тестовых заданий через веб-интерфейс администратором.

6.4. Настройка взаимодействия с парком роботов

В зависимости от выбранного режима работы: — при использовании встроенной RCS WES — настраиваются параметры подключения к роботам в файле `.env` либо через интерфейс администратора; — при использовании сторонней RCS вендора — настраивается адрес и реквизиты доступа к программному интерфейсу RCS вендора.

При смешанной эксплуатации (часть парка обслуживается встроенной RCS, часть — сторонней) обе группы параметров настраиваются одновременно.

---

7. Запуск, остановка и перезапуск

7.1. Запуск

Запуск всех компонентов Платформы выполняется командой:

```
docker compose up -d
```

7.2. Остановка

Корректная остановка с сохранением состояния:

```
docker compose stop
```

Полная остановка с удалением контейнеров (тома данных при этом сохраняются):

```
docker compose down
```

7.3. Перезапуск

Перезапуск всех компонентов:

```
docker compose restart
```

Перезапуск отдельного компонента (на примере WES Server):

```
docker compose restart wes-server
```

7.4. Просмотр журналов работы

Просмотр сводных журналов всех контейнеров:

```
docker compose logs -f
```

Просмотр журнала отдельного компонента:

```
docker compose logs -f wes-server
docker compose logs -f map-editor
```

Журналы Платформы дополнительно сохраняются в стандартном механизме журналирования Docker и могут быть переданы в централизованную систему сбора журналов организации.

---

8. Проверка работоспособности

8.1. Проверка состояния контейнеров

```
docker compose ps
```

Все компоненты Платформы должны находиться в состоянии `running` или `healthy`. При наличии состояния `restarting` или `exited` следует обратиться к журналу соответствующего контейнера.

8.2. Проверка программного интерфейса

```
curl http://localhost:8081/api/health
```

Ожидаемый ответ: `{"status":"ok"}`. При получении иного ответа или таймауте — проверьте журнал WES Server и доступность СУБД.

8.3. Проверка веб-интерфейса

Откройте в браузере адрес `http://<адрес-сервера>:8080`. Должна загрузиться страница входа Map Editor. Войдите с учётной записью администратора и убедитесь, что: — карта склада загружается (при наличии загруженной карты); — индикатор соединения в верхней панели окрашен в зелёный цвет; — список роботов и список заданий отображаются корректно.

8.4. Проверка интеграции с внешними системами

При настроенной интеграции с WMS — выполните пробное создание тестового задания со стороны WMS и убедитесь, что оно появилось в Платформе. При настроенной интеграции с RCS — убедитесь, что зарегистрированные роботы отображаются в Платформе в статусе IDLE или CHARGING.

---

9. Обновление Платформы

9.1. Подготовка к обновлению

Перед обновлением:

1. Создайте резервную копию базы данных и файлов конфигурации (см. раздел 10).
2. Уведомите пользователей о плановом простое.
3. Дождитесь завершения активных заданий либо корректно их остановите.

9.2. Установка новых образов

При получении обновлённой поставки:

1. Замените файл `docker-compose.yml` (при необходимости).
2. Импортируйте новые образы (если они поставлены в виде архивов) или обновите ссылки на образы в `docker-compose.yml`.
3. Перезапустите Платформу:

```
docker compose pull   # при использовании реестра контейнеров
docker compose up -d
```

WES Server при запуске автоматически применит миграции базы данных, требуемые для новой версии.

9.3. Откат к предыдущей версии

В случае проблем после обновления допустим откат к предыдущей версии. Порядок отката включает:

1. Остановку Платформы.
2. Восстановление резервной копии базы данных (если миграции изменили схему несовместимым образом).
3. Возврат `docker-compose.yml` и образов к предыдущей версии.
4. Запуск Платформы.

Перед каждым откатом необходимо проконсультироваться с поставщиком ПО относительно совместимости версий и порядка отката для конкретного обновления.

---

10. Резервное копирование и восстановление

10.1. Резервное копирование базы данных

Создание резервной копии базы данных (при использовании контейнерной PostgreSQL):

```
docker compose exec -T postgres pg_dump -U <пользователь> <база> > /opt/wes/backups/wes-$(date +%F).sql
```

Рекомендуемая периодичность резервного копирования — не реже одного раза в сутки. Резервные копии следует хранить на отдельном носителе или удалённом хранилище.

10.2. Резервное копирование конфигурации

Файл `.env`, `docker-compose.yml` и иные файлы конфигурации следует копировать вместе с резервными копиями базы данных. Конфигурационные файлы содержат секретные сведения (пароли, ключи) — резервные копии должны храниться с надлежащим уровнем защиты.

10.3. Восстановление

Порядок восстановления:

1. Остановите Платформу: `docker compose down`.
2. Восстановите файлы конфигурации и тома данных.
3. Восстановите базу данных:

   ```
   docker compose up -d postgres
   cat /opt/wes/backups/wes-<дата>.sql | docker compose exec -T postgres psql -U <пользователь> <база>
   ```

4. Запустите остальные компоненты: `docker compose up -d`.
5. Проверьте работоспособность по разделу 8.

---

11. Устранение неполадок

11.1. Контейнер не запускается

Признаки: в выводе `docker compose ps` контейнер находится в состоянии `restarting` или `exited`.

Действия: посмотреть журнал контейнера командой `docker compose logs <имя контейнера>`. Типовые причины — занятость порта, недоступность СУБД, ошибки в файле `.env`.

11.2. Веб-интерфейс не открывается

Признаки: при обращении к `http://<адрес-сервера>:8080` страница не загружается или возвращается ошибка соединения.

Действия: убедиться, что контейнер Map Editor запущен; проверить настройку межсетевого экрана (`firewall-cmd`, `ufw`, `iptables`); проверить корректность переменных окружения, связанных с адресом WES Server.

11.3. Веб-клиент загружается, но соединение с сервером отсутствует

Признаки: индикатор соединения окрашен в красный цвет, обновления не приходят.

Действия: убедиться, что WES Server запущен и отвечает на запросы (см. п. 8.2); проверить правильность адреса WES Server в настройках веб-клиента; проверить настройки межсетевого экрана для порта 8081.

11.4. Ошибки подключения к базе данных

Признаки: в журнале WES Server присутствуют сообщения о невозможности подключения к СУБД.

Действия: проверить, что контейнер PostgreSQL (или внешняя СУБД) запущен и доступен; сверить строку подключения в файле `.env`; убедиться, что пароль и имя пользователя соответствуют действительности.

11.5. Высокая нагрузка

Признаки: замедление работы Платформы, длительная обработка действий пользователя.

Действия: посмотреть нагрузку командой `docker stats`; при необходимости — увеличить выделенные ресурсы хоста; обратиться к поставщику ПО для рекомендаций по тонкой настройке.

11.6. Сброс к начальному состоянию

При необходимости полного сброса (только для случая тестового развёртывания, без сохранения данных!) выполните:

```
docker compose down -v
```

Команда удалит все контейнеры и тома данных. Использовать с особой осторожностью; на промышленной установке применять не следует.

---

12. Приложения

Приложение А. Пример файла docker-compose.yml

Конкретный состав файла зависит от версии поставки и может включать дополнительные параметры (тома, сети, политики перезапуска и т. п.). Ниже приведена упрощённая схема, иллюстрирующая принцип:

```
services:
  postgres:
    image: postgres:18-alpine
    environment:
      POSTGRES_DB: ${DB_NAME}
      POSTGRES_USER: ${DB_USER}
      POSTGRES_PASSWORD: ${DB_PASSWORD}
    volumes:
      - postgres_data:/var/lib/postgresql/data
    restart: unless-stopped
```

```
  wes-server:
    image: wes-server:1.0.0
    environment:
      DB_DSN: ${DB_DSN}
      API_PORT: ${API_PORT}
    ports:
      - "${API_PORT}:8081"
    depends_on:
      - postgres
    restart: unless-stopped

  map-editor:
    image: wes-map-editor:1.0.0
    environment:
      API_URL: ${API_URL}
    ports:
      - "${WEB_PORT}:80"
    depends_on:
      - wes-server
    restart: unless-stopped

volumes:
  postgres_data:
```

Приложение Б. Пример файла .env

```
# База данных
DB_NAME=wes
DB_USER=wes_user
DB_PASSWORD=ЗАМЕНИТЬ_НА_СИЛЬНЫЙ_ПАРОЛЬ
DB_DSN=postgres://wes_user:ЗАМЕНИТЬ_НА_СИЛЬНЫЙ_ПАРОЛЬ@postgres:5432/wes?sslmode=disable

# Сетевые порты
API_PORT=8081
WEB_PORT=8080
API_URL=http://адрес-сервера:8081

# Часовой пояс
TZ=Europe/Moscow

# Уровень журналирования
LOG_LEVEL=info

# Интеграция с WMS (опционально)
WMS_ENABLED=false
WMS_URL=
WMS_TOKEN=

# Взаимодействие с RCS
RCS_BUILTIN_ENABLED=true
RCS_VENDOR_ENABLED=false
RCS_VENDOR_URL=
```

Приложение В. Пример настройки обратного прокси (Nginx)

При размещении Платформы за обратным прокси Nginx рекомендуется следующая конфигурация:

```
server {
    listen 80;
    server_name ваш-домен;

    location / {
        proxy_pass http://localhost:8080;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
    }

    location /api {
        proxy_pass http://localhost:8081;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
    }

    location /ws {
        proxy_pass http://localhost:8081;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_read_timeout 86400;
    }
}
```

При организации HTTPS-доступа дополнительно настраиваются сертификаты (например, средствами Let's Encrypt) и блок `listen 443 ssl` с соответствующими параметрами.

Приложение Г. Контакты технической поддержки

— Техническая поддержка: [ЗАПОЛНИТЬ: email и телефон]; — Контакты по вопросам интеграции: [ЗАПОЛНИТЬ]; — Часы работы поддержки: [ЗАПОЛНИТЬ].

Подготовил: _____ / _____ **Должность:** _____ **Дата:** «» _____ **20** г.

Утверждаю: _____ / _____ **Должность:** _____ **Дата:** «» _____ **20** г.