Инструкция по установке экземпляра программного обеспечения, предоставленного для проведения экспертной проверки «Программа для ЭВМ «Платформа разметки»»

Содержание

О продукте

Платформа представляет собой веб-приложение для разметки данных. Экземпляр программного обеспечения поставляется в виде исходного кода и конфигурационных файлов для локального развертывания.

В состав локального экземпляра входят:

- backend на Python/Django;
- frontend на React, уже собранный в виде статических файлов в составе поставляемого экземпляра;
- контейнеры инфраструктуры для локального запуска: PostgreSQL, PgBouncer, Redis, MinIO;
- фоновые сервисы обработки задач dramatiq и scheduler;
- конфигурация контейнерного запуска на базе docker compose.

Локальное развертывание выполняется с использованием docker compose.

Состав экземпляра программного обеспечения

После распаковки архива или получения рабочей копии репозитория в корневой папке проекта находятся основные каталоги и файлы:

- backend — серверная часть программного обеспечения;
- frontend — исходный код клиентской части программного обеспечения;
- docs — проектная документация;
- compose.yml — файл локального развертывания контейнеров;
- compose.public.yml — дополнительный файл конфигурации для публичной сборки;
- Dockerfile — инструкция сборки образа backend;
- public.Dockerfile — инструкция сборки образа backend для публичного сценария установки;
- backend/app/static — статические файлы frontend, достаточные для запуска поставляемого экземпляра;
- backend/app/templates/index.html — HTML-шаблон точки входа веб-интерфейса;
- .env.example, .env.compose.example, .env.secrets.example — шаблоны переменных окружения.

Требования к аппаратному и программному обеспечению

Требования к аппаратному обеспечению

Для локальной проверки работоспособности рекомендуется использовать вычислительную среду со следующими характеристиками:

- не менее 4 логических ядер CPU;
- не менее 8 ГБ оперативной памяти;
- не менее 20 ГБ свободного дискового пространства;

• доступ к сети Интернет, если установка выполняется из исходных кодов с загрузкой зависимостей.

Примечание: точные минимальные аппаратные требования в репозитории не зафиксированы. Указанная конфигурация является рекомендуемой для локального развертывания всех сервисов экземпляра.

Требования к программному обеспечению

На рабочем месте должны быть установлены:

• Docker Desktop или Docker Engine с поддержкой docker compose;

• терминал командной строки.

Подготовка к установке

1. Подготовка экземпляра

1. Получите архив с экземпляром программного обеспечения или рабочую копию репозитория.

2. Распакуйте архив в отдельный каталог или перейдите в корневую директорию проекта labelx.

2. Подготовка файлов окружения

В проекте используются три файла переменных окружения:

• .env — базовые настройки приложения;

• .env.compose — настройки контейнерного запуска;

• .env.secrets — секреты и внешние интеграции.

При первом запуске эти файлы подготавливаются вручную:

```
cp -n .env.example .env
cp -n .env.compose.example .env.compose
cp -n .env.secrets.example .env.secrets
```

Базовые значения в шаблонах достаточны для локального запуска. Заполнение .env.secrets требуется только при использовании внешних интеграций, таких как SSO или Sentry.

3. Выбор сценария сборки

Для поставляемого экземпляра доступны два варианта контейнерной сборки:

• стандартный — через compose.yml и Dockerfile;

• публичный — через compose.yml, compose.public.yml и public.Dockerfile.

Для установки из общедоступных источников применяйте публичный сценарий сборки.

Установка экземпляра программного обеспечения

Вариант 1. Установка одной командой

Рекомендуемый способ установки после выполнения шагов из раздела «Подготовка к установке»:

```
docker compose -f compose.yml -f compose.public.yml up --build
```

Команда выполняет следующие действия:

- собирает Docker-образ backend;
- загружает служебные Docker-образы инфраструктуры при их отсутствии;
- запускает контейнеры postgres, pgbouncer, redis, minio, web, dramatiq, scheduler;
- в контейнере web автоматически применяет миграции базы данных;
- в контейнере web автоматически создает локального администратора и базовые группы доступа;
- публикует веб-интерфейс по адресу http://localhost:8080.

Вариант 2. Пошаговая установка

Если требуется выполнить установку по шагам, используйте следующую последовательность команд из корня проекта:

```
cp -n .env.example .env
cp -n .env.compose.example .env.compose
cp -n .env.secrets.example .env.secrets

docker compose pull postgres pgbouncer redis minio
docker compose -f compose.yml -f compose.public.yml build web
docker compose -f compose.yml -f compose.public.yml up postgres pgbouncer
redis minio web dramatiq scheduler
```

При необходимости контейнеры можно запустить в фоновом режиме:

```
docker compose -f compose.yml -f compose.public.yml up -d postgres
pgbouncer redis minio web dramatiq scheduler
```

В процессе запуска будут подняты следующие контейнеры и сервисы:

- postgres — основная база данных;
- pgbouncer — прокси для подключений к PostgreSQL;
- redis — очередь и кэш;
- minio — S3-совместимое файловое хранилище;
- web — основной backend-сервис с веб-интерфейсом;
- dramatiq — обработчик фоновых задач;
- scheduler — планировщик задач.

Вариант 3. Загрузка демонстрационных данных

Если после запуска требуется наполнить систему тестовыми проектами и датасетами, выполните:

```
docker compose -f compose.yml -f compose.public.yml run --rm web python
manage.py fill_db
```

Данный шаг является дополнительным и нужен только для демонстрации работы системы на предзагруженных данных.

Проверка корректности установки

После завершения установки откройте в браузере основной интерфейс системы:

```
http://localhost:8080
```

Дополнительно могут использоваться следующие локальные адреса:

- http://localhost:9001 — консоль MinIO;
- http://localhost:8080/api/schema/ — OpenAPI-схема backend.

Признаками успешной установки являются:

- страница приложения доступна по адресу http://localhost:8080;
- контейнеры web, dramatiq, scheduler, postgres, pgbouncer, redis, minio находятся в состоянии запуска;
- при первом старте автоматически применяются миграции базы данных;
- в локальном режиме создается пользователь-администратор admin, а вход в интерфейс выполняется автоматически.

Для дополнительной проверки состояния контейнеров можно выполнить:

```
docker compose -f compose.yml -f compose.public.yml ps
```

Остановка и повторный запуск

Для остановки локального экземпляра выполните:

```
docker compose -f compose.yml -f compose.public.yml down
```

Для повторного запуска после остановки выполните:

```
docker compose -f compose.yml -f compose.public.yml up
```

Если требуется полностью пересоздать базу данных локального экземпляра, используйте:

```
docker compose -f compose.yml -f compose.public.yml down -v
docker compose -f compose.yml -f compose.public.yml up --build
```

Возможные проблемы при установке

Ошибка загрузки зависимостей из сети Интернет

Причина:

- отсутствует доступ в Интернет;
- временно недоступен публичный репозиторий пакетов;
- локальные сетевые ограничения блокируют загрузку зависимостей.

Действия:

- проверьте сетевое подключение;

• повторите команду установки позднее;

• при необходимости настройте доступ к публичным репозиториям пакетов в соответствии с политиками вашей организации.

Ошибка запуска контейнеров из-за занятых портов

Локальное развертывание использует порты:

• 8080 — веб-приложение;

• 5432 — PostgreSQL;

• 6432 — PgBouncer;

• 6379 — Redis;

• 9000 и 9001 — MinIO.

Если один из портов уже занят, освободите его или скорректируйте проброс портов в compose.yml.

Ошибка сборки Docker-образа при использовании стандартного сценария сборки

Причина:

• выполняется сборка через стандартный сценарий, ориентированный на внутреннюю среду разработки;

• используемая среда установки не соответствует требованиям стандартного сценария.

Действия:

• для внешней установки используйте compose.public.yml и public.Dockerfile;

• не применяйте стандартный сценарий сборки для публичного экземпляра.

Длительная первая установка

При первом запуске установка может занимать продолжительное время, так как выполняются:

• загрузка Docker-образов;

• сборка Docker-образа backend;

• установка Python-зависимостей внутри Docker-образа.

Это штатное поведение для первого развертывания экземпляра.