Руководство по установке и настройке

Это руководство поможет вам установить и настроить необходимые сервисы с использованием Docker, Docker Compose, а также с помощью скрипта для управления профилями и загрузки образов.

Требования:

1. **Docker** — установленный Docker для запуска контейнеров. Рекомендуемая версия 24.0.6 и выше.
2. **Docker Compose** — для управления мультиконтейнерной средой. Рекомендуемая версия 1.29 и выше.

Описание файлов

1. Папка ***`images`***

- Содержит образы Docker-контейнеров для различных сервисов.
- Эти образы загружаются при выполнении скрипта установки, если в конфигурации установлена соответствующая опция.

2. Папка ***`configs`***

- Содержит конфигурационные файлы для сервисов в формате JSON.
- Эти файлы монтируются в контейнеры во время их запуска и управляют параметрами работы сервисов.

3. Папка ***`mock-channels/mappings`***

- Содержит конфигурационные файлы для ЗАГЛУШЕК каналов в формате JSON.
- Эти файлы необходимы контейнеру **wiremock-channels**, который создан на основе образа wiremock.
- В установочный пакет входит три заглушки, которые всегда имитируют успешную отправку и доставку.
- В файле *configs/communication-api-appsettings.json* в секции *ChannelSettings:Channels:** прописаны настройки к этим заглушкам.

4. Файл ***`.env`***

- В этом файле хранятся переменные окружения, используемые для настройки сервисов. Ниже приведён список основных переменных и их описания:

Переменная	Описание
ASPNETCORE_ENVIRONMENT	Определяет среду, в которой запущены сервисы (`Development`, `Production`).
AUTHORIZATION_DB_CONNECTION	Строка подключения к базе данных сервиса авторизации, используемая сервисом `authorization-api`.

Переменная	Описание
COMMUNICATION_DB_CONNECTION	Строка подключения к базе данных КС, используемая сервисом `communication-api`.
COMMUNICATION_RABBITMQ_CONNECTION	Строка подключения к серверу RabbitMQ для сервиса `communication-api`.
SECURITY_KEY	Ключ безопасности для шифрования паролей и подписи токенов, используемый сервисами `authorization-api` и `communication-ui`. *Минимальный размер ключа должен быть 32 символа.*
NEED_CREATE_ADMIN	Логическое значение (`true`/`false`), определяющее необходимость создания/обновления администратора по умолчанию. У такого адмнистратора login=admin и password=admin. При перезапуске `authorization-api` пароль у существующего администратора *admin* будет обновляться. При необходимости можно создать своего администратора и удалить базового.
COMMUNICATION_API_HOST	URL-адрес API сервиса `communication-api`. Используется сервисом `communication-ui`
AUTHORIZATION_API_HOST	URL-адрес API сервиса `authorization-api`. Используется сервисом `communication-ui`

5. Файл ***`docker-compose.yml`***

- Основной файл, описывающий сервисы и их параметры. Содержит несколько профилей, которые позволяют управлять запуском контейнеров.
- Все контейнеры имеют начальную привязку внутренних портов к внешним, при необходимости привязку можно отключить или изменить port binding.
- Контейнеры баз данных и `RabbitMq` имеют базовые учётные данные для входа, при необходимости их можно изменить.
- Контейнеры баз данных имеют папки для монтирования томов. По умолчанию папки будут созданы на одном уровне с `docker-compose.yml`

  Основные профили:

Профиль	Описание
service	Запускает основные сервисы, такие как `authorization-api`, `communication-api`, и `communication-ui`.
authorization-db	Запускает контейнер базы данных для `authorization-api`.
communication-db	Запускает контейнер базы данных для `communication-api`.
communication-rabbit	Запускает контейнер с RabbitMQ для `communication-api`.
ui	Запускает контейнеры `authorization-api` и `communication-ui`
channels	Запускает контейнер *wiremock-channels*, который содержит API для демонстрационных заглушек
full	Запускает все контейнеры вместе (включая базы данных, сервисы и RabbitMQ).

6. Файл ***`docker-compose.override.yml`***

- Дополняет и переопределяет параметры из `docker-compose.yml` для конкретных окружений.
- Используется для настройки переменных окружения, портов и других специфичных параметров.

Установка

Следуйте этим шагам для установки и запуска сервисов.

1. Настройка параметров установки в ***`setup-config.env`***

- Переменная `LOAD_IMAGES` определяет, нужно ли загружать образы Docker из локальной папки `images`. Образы загружаются в локальный реестр из tar файлов.
- Переменная `PROFILES` указывает, какие профили сервисов будут запущены (например, `service`, `authorization-db`, `communication-db`). В качестве разделителя между профилями используется `запятая` без пробелов.

2. Запуск скрипта установки

- Выполните скрипт `setup.sh` для начала процесса установки и запуска сервисов.
- Скрипт может принимать на вход следующие параметры:
    - `-y (-Y)` указывает, что необходимо загрузить образы из локальной папки `images`. Образы загружаются в локальный реестр из `tar` файлов. Аргумент `-n (-N)` указывает, что образы не требуется загружать. При отсутствии аргументов `(-y/-n)` будет запрошено подтверждение на загрузку после запуска скрипта.
    - `--p` аргумент для формирования профилей запуска. После него необходимо передать название профилей для запуска. Если аргумент не передан, то профили будут взяты из переменной `PROFILES` файла `setup-config.env`

- Скрипт выполняет следующие шаги:
  - Формирует команду для запуска `Docker Compose` с указанными профилями.
  - Загружает образы контейнеров, если передан аргумент загрузки `-y`.
  - Останавливает запущенные сервисы (всегда по профилю `full`).
  - Заново запускает сервисы с указанными профилями

- Примеры запуска:
  - `./setup.sh -n` – не загружает образы, профили берёт из конфига
  - `./setup.sh -y --p service channels` – загружает образы, использует для запуска профили `service, channels`

- Для просмотра состояния сервисов можно использовать команду `docker compose ps -a`. Для просмотра логов в консоли можно использовать команду `docker compose logs -f communication-api` (`communication-api` можно заменить другим сервисом)

- Возможные проблемы:
  - Возникновение ошибки: Error creating overlay mount to /var/lib/docker/overlay2/...-init/merged: too many levels of symbolic links. Проблема связана с файловой системой Docker. Необходимо провести очистку от ненужных образов и контейнеров. Возможно потребуется переустановить Docker.
  - Проблема с добавлением образов в локальный реестр, если такой образ уже существует. Обычно загрузка такого же образа происходит поверх имеющегося без проблем. Но можно удалить все имеющиеся образы по имени. Например, команда (для linux)

    ```
    docker rmi $(docker images | grep 'communication-ui')
    ```

3. Самостоятельная установка всех компонентов

Скрипт `setup.sh` необязателен к использованию. Его задача загрузить образы из `tar` файлов в локальный реестр и запустить сервисы, определённые в `docker-compose.yml` файле с выбранными профилями. Вы можете использовать собственную сборку и расположить образы из папки `images` в разных окружениях с их конфигурацией.

Для запуска сервисов без скрипта, но с текущим `docker-compose` файлом можно использовать команду `docker compose --profile full up -d`. Для нескольких профилей при запуске нужно указать `--profile` перед каждым профилем. Таким же образом можно использовать команду для остановки сервисов `docker compose --profile full stop`. Остановленные контейнеры можно удалить командой `docker compose --profile full rm -f`.