Инструкция по установке и настройке платформы Релевантер.

1. Назначение документа

Настоящий документ описывает общий порядок установки, первичной настройки, проверки и обновления платформы Релевантер в инфраструктурном контуре заказчика.

Инструкция предназначена для системных администраторов, DevOps-инженеров и специалистов ИТ-службы заказчика, отвечающих за эксплуатацию серверной инфраструктуры, сетевых доступов, резервного копирования и мониторинга.

Документ описывает типовой процесс разворачивания платформы на базе Docker Compose. Для Kubernetes-инфраструктур порядок установки согласуется отдельно.

2. Состав платформы

Платформа Релевантер состоит из нескольких сервисов:

Компонент	Назначение
`recruiter-front`	Веб-интерфейс рекрутера
`candidate-front`	Веб-интерфейс кандидата
`business-back`	Основной backend платформы, бизнес-логика, аккаунты, вакансии, интервью
`relevanter-back`	Сервис релевантного поиска и интеграций с внешними HR-источниками
`ai-recruiter-back`	Сервис AI-рекрутера, диалогов и кампаний
`messaging-back`	Сервис коммуникаций и интеграций с мессенджерами
`hr-proxy`	Прокси-сервис для интеграций с внешними HR-системами
`postgres`	Основная база данных
`redis`	Кэш, очереди и служебное хранилище
`kafka`	Брокер событий между сервисами
`nginx` или иной reverse proxy	Единая точка входа, TLS-терминация и маршрутизация запросов

Состав конкретного релиза может отличаться. Актуальный перечень сервисов указывается в файле `docker-compose.yml`, входящем в релизный комплект.

3. Требования к инфраструктуре

3.1. Минимальные требования для тестового контура

Ресурс	Минимальное значение
CPU	4 vCPU
RAM	16 GB
Диск	100 GB SSD
ОС	Linux x86_64
Docker	24.x или выше
Docker Compose	v2.x

3.2. Рекомендуемые требования для промышленного контура

Ресурс	Рекомендуемое значение
CPU	8 vCPU и выше
RAM	32 GB и выше
Диск	300 GB SSD и выше
ОС	Linux x86_64, актуальная LTS-версия
Docker	24.x или выше
Docker Compose	v2.x

Итоговые требования зависят от количества пользователей, объема базы кандидатов, интенсивности AI-операций, количества коммуникационных кампаний и политики хранения аудио/логов.

4. Сетевые требования

4.1. Входящие соединения

Для пользователей платформы должен быть доступен HTTPS-адрес, например:

```
https://relevanter.customer.example
```

Рекомендуемый публичный входящий порт:

Порт	Протокол	Назначение
443	HTTPS	Доступ пользователей к платформе

Порт `80` может использоваться только для перенаправления на HTTPS или для выпуска TLS-сертификатов.

4.2. Внутренние соединения

Внутри Docker-сети сервисы взаимодействуют друг с другом по внутренним DNS-именам контейнеров. Публиковать наружу внутренние порты backend-сервисов, PostgreSQL, Redis и Kafka не рекомендуется.

4.3. Исходящие соединения

В зависимости от включенных интеграций серверу могут потребоваться исходящие соединения к следующим внешним системам:

Система	Назначение
HeadHunter	Работа с вакансиями, резюме и откликами
E-Staff / Skillaz	Интеграции с HR-системами заказчика
AI-провайдеры	Генерация, анализ и обработка AI-запросов
ElevenLabs или аналогичный сервис	Голосовые сценарии, если включены
SMTP-сервер	Отправка email-уведомлений
Telegram / WhatsApp / MAX	Коммуникации с кандидатами, если включены
Google OAuth / Calendar	Авторизация и календарные интеграции, если включены

Если контур полностью закрытый, необходимо заранее согласовать с поставщиком список функций, которые должны работать без внешнего доступа, и список интеграций, которые будут отключены или заменены локальными аналогами.

5. Подготовка сервера

1. Установите Docker и Docker Compose.
2. Создайте рабочий каталог:

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

3. Распакуйте релизный комплект и перейдите в каталог, где находится файл `docker-compose.yml`:

```
tar -xzf <release-archive>.tar.gz -C /opt/relevanter
cd /opt/relevanter/<release-directory>
```

4. Создайте постоянные каталоги для данных и резервных копий:

```
mkdir -p data/postgres data/redis data/kafka data/audio data/audio-answers data/logo backups logs
```

6. Получение Docker-образов

6.1. Вариант A: загрузка из private registry

Если сервер имеет доступ к registry поставщика:

```
docker login registry.relevanter.example
docker compose pull
```

Адрес registry, логин и пароль передаются заказчику отдельно по защищенному каналу.

6.2. Вариант B: установка в закрытом контуре

Если сервер не имеет доступа к внешним registry, загрузите образы из архива:

```
docker load -i relevanter-images-1.0.0.tar
```

После загрузки проверьте наличие образов:

```
docker images | grep relevanter
```

7. Настройка переменных окружения

1. Создайте рабочий `.env` на основе шаблона:

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

2. Отредактируйте значения:

```
nano .env
```

Основные параметры:

```
# Версия поставки
RELEVANTER_VERSION=1.0.0
```

```
# PostgreSQL
POSTGRES_DB=hr_recruiter
POSTGRES_USER=hr_user
POSTGRES_PASSWORD=<strong-password>

# SMTP / email
EMAIL_HOST=<smtp-host>
EMAIL_PORT=587
EMAIL_USERNAME=<smtp-user>
EMAIL_PASSWORD=<smtp-password>
EMAIL_FROM=no-reply@relevanter.customer.example

# HeadHunter
HH_ACCESS_TOKEN=<hh-token>
HH_CLIENT_ID=<hh-client-id>
HH_CLIENT_SECRET=<hh-client-secret>
HH_REDIRECT_URI=https://relevanter.customer.example/relevanter/api/auth/hh/callback

# HR-интеграции
ESTAFF_API_URL=<estaff-url>
ESTAFF_BEARER_TOKEN_API=<estaff-token>
SKILLAZ_API_URL=<skillaz-url>
SKILLAZ_API_TOKEN=<skillaz-token>

# AI-провайдеры
GIGACHAT_AUTH_KEY=<gigachat-key>

# Голосовые сценарии
ELEVEN_LABS_API_KEY=<elevenlabs-key>
ELEVENLABS_DEFAULT_VOICE_ID=<elevenlabs-voice-id>

# Messaging / мессенджеры
TELEGRAM_API_ID=<telegram-api-id>
TELEGRAM_API_HASH=<telegram-api-hash>
TELEGRAM_NOTIFICATION_BOT_TOKEN=<telegram-bot-token>
TELEGRAM_NOTIFICATION_BOT_USERNAME=<telegram-bot-username>
WHATSAPP_PHONE_NUMBER_ID=<whatsapp-phone-number-id>
WHATSAPP_ACCESS_TOKEN=<whatsapp-access-token>
WHATSAPP_WEBHOOK_VERIFY_TOKEN=<whatsapp-webhook-verify-token>

# Google OAuth / Calendar
GOOGLE_CLIENT_ID=<google-client-id>
GOOGLE_CLIENT_SECRET=<google-client-secret>
GOOGLE_REDIRECT_URI=https://relevanter.customer.example/relevanter/ai-recruiter/integrations/google-calendar/callback
```

Полный перечень поддерживаемых параметров окружения указывается в файле `.env.example`, входящем в релизный комплект. При расхождении между инструкцией и `.env.example` приоритет имеет `.env.example` для конкретной версии поставки.

3. Ограничьте права на файл с секретами:

```
chmod 600 .env
```

Секреты, токены, пароли и приватные ключи не должны храниться в системе контроля версий, передаваться в открытых каналах связи или попадать в диагностические архивы без маскирования.

8. Настройка TLS и домена

Рекомендуется использовать единый домен:

```
https://relevanter.customer.example
```

Типовая маршрутизация:

URL	Сервис
`/`	`recruiter-front`
`/candidate/`	`candidate-front`
`/api/`	`business-back`
`/relevanter/`	`relevanter-back`
`/hr-proxy/`	`hr-proxy`, если требуется внешний доступ к интеграционному API
`/admin/`	Административный интерфейс `hr-proxy`, если включен

Сервисы `ai-recruiter-back`, `messaging-back`, `postgres`, `redis` и `kafka` обычно не публикуются наружу и используются только для внутреннего взаимодействия между компонентами платформы.

TLS-сертификат может быть:

1. Выпущен ИТ-службой заказчика.
2. Выпущен через публичный CA, если контур имеет доступ в интернет.
3. Предоставлен в виде корпоративного сертификата и ключа.

Файлы сертификатов обычно размещаются в каталоге:

```
nginx/certs/
```

Пример:

```
nginx/certs/fullchain.pem
nginx/certs/privkey.pem
```

9. Запуск платформы

1. Проверьте итоговую конфигурацию:

```
docker compose config
```

2. Запустите инфраструктурные сервисы:

```
docker compose up -d postgres redis kafka
```

3. Дождитесь готовности PostgreSQL, Redis и Kafka:

```
docker compose ps
```

4. Запустите прикладные сервисы:

```
docker compose up -d
```

5. Проверьте статус контейнеров:

```
docker compose ps
```

Все критичные сервисы должны находиться в состоянии `running` или `healthy`.

10. Проверка установки

10.1. Проверка контейнеров

```
docker compose ps
```

10.2. Проверка логов

```
docker compose logs --tail=200 business-back
docker compose logs --tail=200 relevanter-back
docker compose logs --tail=200 ai-recruiter-back
docker compose logs --tail=200 messaging-back
docker compose logs --tail=200 hr-proxy
```

10.3. Проверка HTTP-доступности

Проверьте основной адрес платформы:

```
curl -k https://relevanter.customer.example
```

Проверьте health endpoints, если они опубликованы через reverse proxy:

```
curl -k https://relevanter.customer.example/api/actuator/health
curl -k https://relevanter.customer.example/relevanter/health
```

Фактические URL health endpoints указываются в релизной документации конкретной версии.

11. Первичная настройка приложения

После технического запуска необходимо выполнить прикладную настройку:

1. Создать или проверить административную учетную запись.
2. Настроить домен и публичные URL платформы.
3. Настроить SMTP для отправки писем.
4. Настроить интеграции с HH, E-Staff, Skillaz или другими HR-системами.
5. Настроить AI-провайдеров и лимиты использования.
6. Настроить мессенджеры и ботов, если используются коммуникационные сценарии.
7. Проверить создание вакансии, кандидата, интервью и тестовой коммуникационной кампании.

Доступы к административной учетной записи передаются заказчику отдельно или создаются на этапе первичной настройки.

12. Резервное копирование

Рекомендуется настроить регулярное резервное копирование следующих данных:

Данные	Что копировать
PostgreSQL	Дамп базы данных
Redis	Снимки данных, если Redis используется как персистентное хранилище
Kafka	Данные брокера, если требуется восстановление очередей
Файловое хранилище	Аудио, ответы, логотипы, вложения
Конфигурация	`.env`, nginx-конфигурация, сертификаты

Пример ручного backup PostgreSQL:

```
mkdir -p backups
docker compose exec postgres pg_dump -U "$POSTGRES_USER" "$POSTGRES_DB" > backups/relevanter-$(date +%F-%H%M%S).sql
```

Пример архивации файлового хранилища:

```
tar -czf backups/relevanter-files-$(date +%F-%H%M%S).tar.gz data/
```

Рекомендуемый график:

Тип backup	Частота	Хранение
Полный backup БД	1 раз в сутки	14-30 дней
Backup файлов	1 раз в сутки	14-30 дней
Backup перед обновлением	Перед каждым обновлением	До подтверждения успешного обновления

13. Восстановление из резервной копии

Перед восстановлением остановите прикладные сервисы:

```
docker compose stop business-back relevanter-back ai-recruiter-back messaging-back hr-proxy recruiter-front candidate-front
```

Восстановите базу данных согласно процедуре, согласованной с поставщиком. Типовой пример:

```
docker compose exec -T postgres psql -U "$POSTGRES_USER" "$POSTGRES_DB" < backups/relevanter-YYYY-MM-DD-HHMMSS.sql
```

После восстановления запустите сервисы:

```
docker compose up -d
```

Проверьте логи и health endpoints.

14. Обновление платформы

Перед обновлением:

1. Ознакомьтесь с release notes новой версии.
2. Сделайте резервную копию PostgreSQL и файлового хранилища.
3. Сохраните текущий `.env`.
4. Проверьте наличие свободного места на диске.

14.1. Обновление через registry

```
docker compose pull
docker compose up -d
docker compose ps
```

14.2. Обновление в закрытом контуре

```
docker load -i relevanter-images-1.0.1.tar
```

Затем измените версию в `.env`:

```
RELEVANTER_VERSION=1.0.1
```

Примените обновление:

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

Миграции базы данных выполняются сервисами автоматически при старте, если это предусмотрено конкретным релизом. Если релиз требует ручного шага миграции, он должен быть отдельно описан в release notes.

15. Откат версии

Откат допускается только в рамках процедур, описанных в release notes конкретного релиза.

Типовой порядок:

1. Остановить сервисы.
2. Вернуть предыдущую версию образов в `.env`.
3. Восстановить backup базы данных, если новая версия применяла необратимые миграции.
4. Запустить сервисы.
5. Проверить логи и health endpoints.

Пример:

```
docker compose stop
cp .env.backup-before-upgrade .env
docker compose up -d
```

16. Мониторинг и эксплуатация

Рекомендуется контролировать:

1. Доступность HTTPS-адреса платформы.
2. Состояние контейнеров.
3. Свободное место на диске.
4. Использование CPU и RAM.
5. Размер PostgreSQL.
6. Ошибки в логах backend-сервисов.
7. Состояние очередей Kafka.
8. Доступность внешних интеграций.

Базовые команды:

```
docker compose ps
docker stats
docker compose logs --tail=200
df -h
```

При промышленной эксплуатации рекомендуется подключить централизованный сбор логов и метрик средствами заказчика.

17. Диагностика типовых проблем

17.1. Контейнер не запускается

Проверьте логи:

```
docker compose logs --tail=200 <service-name>
```

Частые причины:

1. Ошибка в `.env`.
2. Недоступна база данных.
3. Недоступен Kafka или Redis.
4. Неверный токен внешней интеграции.
5. Недостаточно прав на volume или каталог с данными.

17.2. Веб-интерфейс открывается, но API недоступен

Проверьте:

1. Конфигурацию reverse proxy.
2. Значения `PUBLIC_BASE_URL`, `API_URL`, `FRONTEND_URL`.
3. Логи `business-back` и `relevanter-back`.
4. CORS-настройки, если используются разные домены.

17.3. Не работает интеграция с внешней HR-системой

Проверьте:

1. Доступность внешнего URL с сервера.
2. Корректность токена.
3. Ограничения firewall/proxy.
4. Логи соответствующего сервиса.

17.4. Не отправляются письма

Проверьте:

1. SMTP host, port, username, password.
2. Доступность SMTP-сервера из контейнера.
3. Требования к TLS/STARTTLS.
4. Политики отправителя на стороне почтового сервера.

18. Ограничение ответственности

Заказчик отвечает за подготовку и эксплуатацию инфраструктурного контура: серверы или виртуальные машины, операционную систему, сетевые доступы, DNS, TLS-сертификаты, firewall, резервное копирование, мониторинг, хранение секретов и соблюдение внутренних требований информационной безопасности.

Поставщик отвечает за предоставление релизного комплекта платформы, Docker-образов или иного согласованного формата поставки, инструкции по установке и обновлению, описание обязательных параметров конфигурации, а также консультационную поддержку при установке, диагностике и обновлении платформы.

Интеграции с внешними системами, такими как HR-системы, AI-провайдеры, SMTP-серверы, мессенджеры и сервисы авторизации, требуют совместной настройки. Заказчик обеспечивает доступность внешних систем и корректность выданных учетных данных, поставщик обеспечивает корректную работу платформы с поддерживаемыми интеграционными интерфейсами.

19. Требования к безопасности

1. Используйте только HTTPS для пользовательского доступа.
2. Не публикуйте наружу PostgreSQL, Redis, Kafka и внутренние backend-порты.
3. Храните `.env` с правами `600`.
4. Используйте уникальные секреты для каждого контура.
5. Не используйте тестовые пароли в промышленном контуре.
6. Регулярно обновляйте TLS-сертификаты.
7. Ограничьте SSH-доступ к серверу.
8. Настройте резервное копирование и проверку восстановления.
9. Передавайте диагностические архивы поставщику только после удаления или маскирования персональных данных и секретов.

20. Информация для обращения в поддержку

При обращении в поддержку подготовьте:

1. Версию платформы.
2. Версию релизного комплекта.
3. Описание проблемы и время возникновения.
4. Вывод команды:

```
docker compose ps
```

5. Логи проблемного сервиса:

```
docker compose logs --tail=500 <service-name>
```

6. Информацию о последних изменениях: обновление, изменение `.env`, замена сертификата, изменение firewall, изменение внешней интеграции.

Не передавайте пароли, токены, приватные ключи и персональные данные без предварительного согласования защищенного канала передачи.

21. История изменений документа

Версия	Дата	Изменения
1.0.0	2026-05-25	Первичная версия общей инструкции по установке и настройке платформы в контуре заказчика