

**Общество с ограниченной ответственностью
«РЖД – Цифровые пассажирские решения»**

*ОГРН 5167746236906 ИНН 9717045555
107078, г. Москва, ул. Новорязанская, д. 8А, стр. 2*

**ПРОГРАММА ДЛЯ ЭВМ
«АВТОПРОТОКОЛ»**

Руководство по установке программного обеспечения

На 11 листах

Москва, 2026

АННОТАЦИЯ

Настоящий документ содержит сведения, необходимые для установки программы для ЭВМ «Автопротокол» (далее – ПО, Сервис): требования к среде развертывания, порядок установки «с нуля», описание конфигурационных параметров, порядок проверки работоспособности, процедуры обновления и удаления. Порядок повседневной работы пользователя и администратора, регламентные операции обслуживания и способы устранения типовых неисправностей приведены в отдельном документе «Руководство по эксплуатации программного обеспечения», входящем в комплект документации.

Документ предназначен для системных администраторов и инженеров сопровождения, осуществляющих развертывание и обслуживание ПО. При подготовке документа учтены рекомендации ГОСТ 19.505-79 «ЕСПД. Руководство оператора» и ГОСТ Р ИСО/МЭК 12207-2010.

СОДЕРЖАНИЕ

1. Общие сведения.....	4
2. Требования к среде развертывания	4
2.1. Аппаратные требования	4
2.2. Программные требования	4
3. Установка программного обеспечения	5
3.1. Получение дистрибутива.....	5
3.2. Подготовка конфигурации	6
3.3. Настройка TLS (при необходимости)	6
3.4. Сборка и запуск	7
3.5. Автоматическая первичная инициализация	7
3.6. Офлайн-развертывание (без доступа к внешним репозиториям)	7
3.7. Распределенное развертывание (несколько серверов).....	8
4. Проверка работоспособности	8
5. Конфигурационные параметры	9
6. Обновление программного обеспечения	10
7. Удаление программного обеспечения	10

1. Общие сведения

Программа для ЭВМ «Автопротокол» – сервис автоматической транскрибации и саммаризации аудио- и видеозаписей, эксплуатируемый в закрытом контуре организации. ПО поставляется в виде дистрибутива (полной копии репозитория исходного кода с конфигурацией развертывания) и развертывается средствами контейнеризации Docker: все компоненты собираются из исходного текста в контейнерные образы непосредственно на оборудовании эксплуатирующей организации, что гарантирует соответствие исполняемого кода переданному исходному тексту.

Функциональные характеристики, архитектура и состав компонентов ПО приведены в документах «Описание функциональных характеристик программного обеспечения» и «Описание технической архитектуры программного обеспечения», представляемых в комплекте с настоящим руководством. Настоящее руководство сосредоточено на практических процедурах установки.

2. Требования к среде развертывания

2.1. Аппаратные требования

Ресурс	Требование
Графический ускоритель (GPU)	NVIDIA с поддержкой CUDA; рекомендуется видеопамять 24 ГБ и более (совокупно для конвейера распознавания и языковой модели). Поддерживаются конфигурации с несколькими GPU: тензорный параллелизм языковой модели, отдельный воркер на каждый ускоритель
Оперативная память	от 16 ГБ (брокер сообщений ограничен 2 ГБ; основная нагрузка приходится на GPU)
Дисковое пространство	от 100 ГБ: образы контейнеров (PyTorch/vLLM) и кэш моделей искусственного интеллекта
Скорость обработки	на рекомендованной конфигурации обеспечивается обработка со скоростью не ниже 2× реального времени

2.2. Программные требования

– операционная система: Linux x86-64 (производственная эксплуатация); допускается любой дистрибутив с поддержкой Docker Engine;

- Docker Engine и Docker Compose v2 (команда `docker compose`);
- драйвер NVIDIA версии не ниже 580.65.06 (ветка r580 и новее) и NVIDIA Container Toolkit – компоненты ПО собраны на CUDA 13.x;
- сетевой доступ пользователей к портам веб-интерфейса (80/443 либо другой настроенный порт);
- доступ сервера к корпоративному SMTP-серверу (для отправки писем);
- для первичной загрузки моделей – доступ к репозиторию моделей (напрямую либо через корпоративный прокси-сервер) или заранее подготовленный офлайн-архив моделей (см. п. 3.6).

Проверка корректности проброса GPU в контейнеры выполняется командой:

```
docker run --rm --gpus all nvidia/cuda:12.4.0 -base-ubuntu22.04  
nvidia-smi
```

В выводе команды должны отображаться сведения об установленном драйвере ($\text{Driver Version} \geq 580.65.06$) и о доступных графических ускорителях.

3. Установка программного обеспечения

3.1. Получение дистрибутива

Дистрибутив ПО представляет собой полную копию репозитория исходного кода, включающую исходные тексты всех компонентов, файлы декларативной конфигурации развертывания (`docker-compose.yml`, каталог `deploy/`), файл примера конфигурации `.env.example` и комплект эксплуатационной документации (каталог `docs/`). Дистрибутив передается эксплуатирующей организации в виде архива либо доступом к репозиторию системы контроля версий правообладателя:

```
git clone git@gitlab.int.smarttravel.ru:protocol/protocol.git  
cd protocol
```

При передаче архивом достаточно распаковать архив в целевой каталог (например, /opt/autoprotocol) и перейти в него.

3.2. Подготовка конфигурации

Конфигурация ПО задается файлом переменных окружения .env, создаваемым из поставляемого примера:

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

В файле .env необходимо заполнить обязательные параметры (сервисы намеренно не запускаются без них – защита от небезопасных значений по умолчанию): пароли брокера сообщений, объектного хранилища и СУБД; учетные данные административной панели (ADMIN_USERNAME, ADMIN_PASSWORD); секрет доверенного прокси (SENTINEL_AUTH_SECRET); токен репозитория моделей для первичной загрузки компонент диаризации (HF_TOKEN); параметры SMTP. Полный справочник всех параметров приведен в документе docs/CONFIGURATION.md в составе дистрибутива; основные параметры перечислены в разделе 5 настоящего руководства.

Обязательная переменная COMPOSE_PROFILES определяет схему развертывания (набор запускаемых компонентов). Для развертывания на одном сервере используется полный набор профилей:

```
COMPOSE_PROFILES=core,local-db,local-broker,local-s3,local-llm,worker
```

3.3. Настройка TLS (при необходимости)

Для доступа по HTTPS в каталог frontend/certs/ размещаются файл цепочки сертификатов fullchain.crt и файл закрытого ключа wildcard.key, выпущенные удостоверяющим центром организации. Каталог сертификатов исключен из системы контроля версий. При отсутствии сертификатов Сервис доступен по HTTP на настроенном порту.

3.4. Сборка и запуск

Сборка контейнерных образов из исходного текста и запуск всех компонентов выполняются одной командой из корневого каталога дистрибутива:

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

При первом запуске выполняются загрузка базовых образов, сборка компонентов и первичная инициализация моделей искусственного интеллекта: сервер языковой модели – 10–30 минут (предусмотрен увеличенный стартовый период проверки доступности), воркер транскрибации – до часа (загрузка и локальный ONNX-экспорт моделей распознавания). Ход инициализации отслеживается командой:

```
docker compose logs -f whisper-worker
```

3.5. Автоматическая первичная инициализация

При первом запуске ПО самостоятельно выполняет все подготовительные операции, не требуя ручных действий администратора:

- создаются служебные бакеты объектного хранилища с правилами автоматического удаления временных файлов (по умолчанию – через 1 сутки);
- создаются таблицы базы данных (статистика, справочники, настройки, журналы) и необходимые индексы; схема данных актуализируется идемпотентными миграциями при каждом старте;
- формируется снимок настроек времени выполнения в объектном хранилище.

3.6. Офлайн-развертывание (без доступа к внешним репозиториям)

Для полностью изолированных контуров предусмотрен перенос кэша моделей с ранее развернутого экземпляра: содержимое тома моделей упаковывается в архив, переносится на целевой сервер, распаковывается в одноименный том, после чего в конфигурации включается офлайн-режим (GIGAAM_OFFLINE_MODE=1, HF_HUB_OFFLINE=1). При таком развертывании обращения к внешним ресурсам не выполняются.

3.7. Распределенное развертывание (несколько серверов)

Поддерживается вынос вычислительных воркеров на отдельные GPU-серверы. На главном сервере запускаются профили core, local-db, local-broker, local-s3 с публикацией портов брокера и хранилища для GPU-сервера; на GPU-сервере – профиль worker с указанием эндпоинтов главного сервера. Готовые шаблоны конфигураций приведены в каталоге deploy/ (env.core-host.example, env.gpu-host.example, env.single-host.example); масштабирование воркеров выполняется штатной командой `docker compose up -d --scale whisper-worker=N` либо конфигурацией «воркер на каждый GPU» (deploy/docker-compose.gpu-multi.yml). Требования: синхронизация времени серверов (NTP), взаимная сетевая доступность по портам брокера и хранилища.

4. Проверка работоспособности

После завершения запуска выполняются следующие проверки:

```
docker compose ps                # все сервисы в состоянии
healthy/running
curl http://localhost:8080/api/health # ответ {"status":"ok"}
```

Веб-проверки:

- веб-интерфейс пользователя открывается по адресу `http://<адрес сервера>:8080` (или по настроенному доменному имени по HTTPS);
- административная панель открывается по отдельному административному адресу и запрашивает учетные данные;
- контрольная (смоук) задача: загрузка короткой аудиозаписи через веб-интерфейс проходит полный цикл до статуса «готово», результат отображается в интерфейсе; при указании адреса электронной почты доставляется письмо с вложениями.

Признаки штатной работы: все контейнеры в состоянии `healthy`; в журналах отсутствуют сообщения об ошибках; в административной панели в секции «Состояние системы» отображаются зеленые индикаторы всех сервисов и активный воркер. Порядок работы

с Сервисом после установки приведен в документе «Руководство по эксплуатации программного обеспечения».

5. Конфигурационные параметры

Все параметры задаются в файле .env и подробно описаны в справочнике docs/CONFIGURATION.md в составе дистрибутива. Основные группы параметров:

Группа	Параметры (основные)	Назначение
Схема развертывания	COMPOSE_PROFILES	Набор запускаемых компонентов (обязательный параметр)
Порты	NGINX_PORT (8080), UVICORN_PORT (8000)	Порт веб-интерфейса; внутренний порт API
Брокер сообщений	BROKER_TYPE, RABBITMQ_HOST/PORT/USER/PASS, KAFKA_BOOTSTRAP_SERVERS, имена очередей	Тип брокера (RabbitMQ/Kafka) и параметры подключения
Объектное хранилище	MINIO_ENDPOINT/BUCKET, MINIO_ACCESS_KEY/SECRET_KEY, S3_*	Подключение к S3-хранилищу (встроенному или внешнему)
СУБД	POSTGRES_HOST/DB/USER/PASSWORD	Подключение к PostgreSQL
Транскрибация	GIGAAM_MODEL, GIGAAM_BATCH_SIZE, GIGAAM_DENOISE, GIGAAM_NORM_FILTER, WHISPER_DEVICE_ID, HF_TOKEN	Параметры конвейера распознавания и первичной загрузки моделей
Языковая модель	VLLM_MODEL, VLLM_COMMAND, VLLM_API_URL, LLM_MAX_TOKENS, LLM_TIMEOUT, LLM_SPLIT_LIMIT	Модель, параметры запуска сервера инференса, лимиты
Оркестрация	BATCH_SIZE, BATCH_TIMEOUT	Пакетная обработка саммаризации
Администрирование	ADMIN_USERNAME/PASSWORD, SENTINEL_AUTH_SECRET	Доступ в административную панель; секрет доверенного прокси
Почта	SMTP_SERVER/PORT/USER/PASSWORD, MAIL_FROM	Отправка результатов по электронной почте

Сеть и лимиты	HTTP(S)_PROXY, NO_PROXY, HF_HUB_OFFLINE, MAX_FILE_SIZE	Прокси, офлайн- режим, лимит размера файла
---------------	---	--

Часть эксплуатационных параметров (параметры конвейера распознавания, лимиты языковой модели, параметры SMTP, шаблон письма, лимит размера файла и др.) дополнительно управляется из административной панели в режиме реального времени с журналированием изменений; значения из панели имеют приоритет над значениями из окружения, индикация источника значения отображается в интерфейсе (см. «Руководство по эксплуатации», раздел 4).

6. Обновление программного обеспечения

Обновление выполняется заменой дистрибутива и пересборкой измененных компонентов:

```
git pull                                # либо распаковка нового
дистрибутива
docker compose up -d --build            # пересборка измененных
компонентов
docker compose restart <сервис>         # перезапуск отдельного
компонента
```

Перед обновлением рекомендуется дождаться завершения активных задач (контроль глубины очереди в административной панели) и создать точку восстановления: резервную копию каталога приложения и тегирование текущих контейнерных образов. Подробная процедура обновления с точкой восстановления и процедура восстановления описаны в документе docs/DEPLOYMENT.md в составе дистрибутива. Тома данных (СУБД, хранилище, кэш моделей) при обновлении сохраняются; изменения схемы данных выполняются автоматически и являются аддитивными.

7. Удаление программного обеспечения

Остановка Сервиса выполняется командой `docker compose down` с сохранением томов данных. Полное удаление вместе с данными – `docker compose down -v` (необратимо удаляет содержимое СУБД и хранилища;

предварительно выполнить резервное копирование по регламенту – см. «Руководство по эксплуатации», раздел «Регламентные операции»). Конфиденциальные данные в объектном хранилище к моменту удаления, как правило, уже удалены автоматической политикой хранения; персональные данные в СУБД (адреса электронной почты в статистике) удаляются вместе с томами либо по отдельному регламенту эксплуатирующей организации.

Контактные данные специалистов для обращений, касающихся установки программного обеспечения:

Телефон: +7 (499) 490-25-58 (доб. 440)

Адрес электронной почты: n.korolev@smarttravel.ru

Режим работы: пн.-чт. 8:00-17:00, пт. 8:00-15:45