ООО «Клиенткор»

ИИ-платформа ClientCore (ClientCore AI Platform)

УТВЕРЖДАЮ

Генеральный директор ООО «Клиенткор»

_____ / Вишняков А. А.

«___» _____ 2026 г.

ИНСТРУКЦИЯ ПО УСТАНОВКЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

ИИ-платформа ClientCore

Москва, 2026

Содержание

1. Общие сведения

Настоящая инструкция описывает порядок установки и развёртывания программного обеспечения «ИИ-платформа ClientCore» (далее — Платформа), а также порядок установки экземпляра Платформы, предоставляемого для проведения экспертной проверки.

Платформа построена по микросервисной архитектуре и развёртывается средствами контейнеризации Docker и оркестратора Docker Compose. Все компоненты поставляются в виде набора контейнеров и файла-манифеста docker-compose.yml.

2. Системные требования

Параметр	Минимальное значение
Операционная система	Linux (Ubuntu 22.04 LTS и выше) либо иная ОС с поддержкой Docker
Docker Engine	версия 24 и выше
Docker Compose	v2 (плагин docker compose)
Оперативная память	8 ГБ (рекомендуется 16 ГБ)
Процессор	4 vCPU и выше
Дисковое пространство	не менее 40 ГБ свободного места
Сеть	исходящий HTTPS-доступ к API российского провайдера LLM (YandexGPT / GigaChat)

3. Состав поставки (контейнеры)

Развёртываемый комплекс включает следующие контейнеры:

Контейнер	Образ	Порт	Назначение
backend	сборка из ./backend (python:3.12)	8000	Основной REST API
chatbot-service	сборка (python:3.12)	8012	Чат-сервис, агентный цикл
chatbot-manager	сборка (python:3.12)	8001	Оркестрация LLM
summary-service	сборка (python:3.12)	8010	Суммаризация
search-service	сборка (python:3.12)	8011	Поиск по знаниям
mcp-server	сборка (python:3.12)	8013	MCP-сервер инструментов
mindbox-connector	сборка (python:3.12)	8014	Коннектор Mindbox
db	pgvector/pgvector:pg15	5432	PostgreSQL + pgvector
mongodb	mongo:7.0	27017	MongoDB
redis	redis:alpine	6379	Кэш / потоки
minio	minio/minio	9000, 9001	Объектное хранилище S3
rabbitmq	rabbitmq:3-management	5672, 15672	Брокер сообщений
frontend	сборка Next.js (node:22)	3000	Веб-интерфейс

Контейнер	Образ	Порт	Назначение
nginx	nginx	80, 443	Обратный прокси, TLS

4. Подготовка к установке

4.1. Получение дистрибутива

Исходный код и манифесты получаются из системы управления версиями правообладателя: https://gitflic.ru/project/clientcore/platform (серверная часть) и https://gitflic.ru/project/clientcore/platform-frontend (клиентская часть). Репозитории являются приватными; доступ предоставляется правообладателем.

```
git clone https://gitflic.ru/project/clientcore/platform.git platform
git clone https://gitflic.ru/project/clientcore/platform-frontend.git platform-
frontend
cd platform
```

4.2. Настройка переменных окружения

В корне каталога platform создаётся файл .env. В поставку входит шаблон .env.example со всеми параметрами и значениями по умолчанию — достаточно скопировать его и заполнить обязательные поля:

- параметры PostgreSQL (DB_NAME, DB_USER, DB_PASSWORD, DB_HOST, DB_PORT, DB_POOL_SIZE, DB_MAX_OVERFLOW);
- параметры MongoDB (MONGO_ROOT_USER, MONGO_ROOT_PASSWORD, MONGODB_URL);
- параметры Redis и RabbitMQ (REDIS_PASSWORD, RABBITMQ_DEFAULT_USER, RABBITMQ_DEFAULT_PASS);
- параметры объектного хранилища S3 (S3_ENDPOINT_URL, S3_ACCESS_KEY, S3_SECRET_KEY, S3_BUCKET_NAME, S3_EXTERNAL_URL) — в поставку входит локальный MinIO, может быть заменён внешним S3-хранилищем;
- параметры доступа к языковой модели (YANDEX_API_KEY, YANDEX_FOLDER_ID, REQUEST_TIMEOUT, MAX_CONTEXT_CHARS);
- параметры аутентификации (JWT_SECRET_KEY, JWT_ALGORITHM);
- адреса домена и сервисов (DOMAIN, FRONTEND_URL, BACKEND_URL);
- параметры интеграций (Mindbox, корпоративный мессенджер, система задач) и ключ внешнего API (EXTERNAL_API_KEY).

Обязательными являются: YANDEX_API_KEY и YANDEX_FOLDER_ID (доступ к языковой модели), JWT_SECRET_KEY (ключ подписи токенов доступа), пароли PostgreSQL, MongoDB, Redis и RabbitMQ. Значения по умолчанию для паролей и ключа подписи использовать нельзя: при них серверная часть завершает запуск с ошибкой. Остальные параметры имеют рабочие значения по умолчанию и подходят для развёртывания на одном сервере.

5. Установка и запуск

5.1. Сборка и запуск контейнеров

```
# скопировать шаблон окружения и заполнить обязательные значения
cp .env.example .env

# сборка образов из исходного текста
docker compose build

# запуск хранилищ (схема БД приводится к актуальной версии до старта сервисов)
docker compose up -d db mongodb redis rabbitmq minio
```

5.2. Инициализация схемы базы данных (миграции Alembic)

Схема PostgreSQL управляется инструментом Alembic. Миграции применяются до начала эксплуатации; при несоответствии схемы последней ревизии серверная часть завершает запуск с ошибкой.

```
# применить все миграции до последней ревизии
docker compose run --rm --no-deps -T backend alembic upgrade head

# проверить отсутствие расхождений схемы
docker compose run --rm --no-deps -T backend alembic check

# запустить остальные сервисы
docker compose up -d

# проверка статуса контейнеров
docker compose ps
```

При первичной инициализации существующей базы, уже содержащей полную схему, выполняется отметка базовой ревизии перед применением форвард-миграций:

```
docker compose run --rm --no-deps -T backend alembic stamp f079af0c6e65
docker compose run --rm --no-deps -T backend alembic upgrade head
```

5.3. Создание учётной записи администратора

Открытой регистрации в Платформе нет: учётные записи создаются по приглашению от администратора. Первая учётная запись администратора создаётся командой инициализации. Без параметра --password команда запрашивает пароль вводом и не сохраняет его в истории командной оболочки.

```
docker compose run --rm --no-deps -T backend \
    python -m scripts.create_admin --email admin@example.org \
    --password 'ПарольНеМенееДвенадцатиСимволов'
```

6. Проверка работоспособности

После запуска выполняется проверка доступности компонентов:

```
# статус всех контейнеров
docker compose ps

# доступность API (ожидается код 200)
curl -sk -o /dev/null -w "%{http_code}\n" https://localhost:8000/docs

# журнал запуска backend (ожидается 'Application startup complete')
docker compose logs --tail 30 backend
```

```
# доступность веб-интерфейса
curl http://localhost:3000
```

Признаком успешной установки является: статус running у всех контейнеров; код ответа 200 от /docs; запись «Application startup complete» и успешное прохождение валидации конфигурации в журнале backend; открытие веб-интерфейса по адресу, указанному в параметре DOMAIN.

7. Установка экземпляра ПО для экспертной проверки

Для проведения экспертной проверки правообладатель предоставляет работающий экземпляр Платформы, развёрнутый по описанной выше процедуре.

7.1. Способ предоставления

Экземпляр программного обеспечения предоставляется в виде исходного текста и манифестов развёртывания, размещённых в репозиториях правообладателя на российском сервисе управления репозиториями GitFlic (gitflic.ru): серверная часть — https://gitflic.ru/project/clientcore/platform, клиентская часть — https://gitflic.ru/project/clientcore/platform-frontend. Репозитории являются приватными; учётные данные для доступа предоставляются экспертной организации на период проведения проверки. Развёртывание выполняется по процедуре, описанной в разделах 4–6 настоящей инструкции.

7.2. Доступ для проверки

Эксперту предоставляются: адрес экземпляра; учётная запись с необходимыми правами; настоящая инструкция по установке; документация по эксплуатации и описание функциональных характеристик экземпляра. При необходимости развёртывания «с нуля» эксперту передаётся архив с исходным кодом и файлом docker-compose.yml; установка выполняется по разделам 4–6 настоящей инструкции.

Требования к среде для самостоятельного развёртывания экземпляра соответствуют разделу 2 настоящей инструкции.