

АВТОМАТИЗИРОВАННАЯ ИНФОРМАЦИОННАЯ СИСТЕМА «ОРБИТА»
(АИС «ОРБИТА»)

ИНСТРУКЦИЯ ПО УСТАНОВКЕ И РАЗВЁРТЫВАНИЮ

Программа для ЭВМ. Свидетельство о государственной регистрации № 2026685490

Версия 1.0.0

Правообладатель: ИП Жарков А. В. · Российская Федерация · 2026

ИНСТРУКЦИЯ ПО УСТАНОВКЕ И РАЗВЁРТЫВАНИЮ

АИС «ОРБИТА». ВЕРСИЯ 1.0.0

ТИТУЛЬНЫЙ ЛИСТ

Поле	Значение
Наименование программы	Автоматизированная информационная система управления деятельностью реабилитационного центра «Орбита» (АИС «Орбита»)
Версия (редакция)	1.0.0
Дата редакции	05.08.2026
Класс ПО по классификатору Минцифры	12.01 «Информационные системы для государственных нужд»
Правообладатель	Жарков Александр (azharkov73@gmail.com , 8-999-971-30-31)
Вид документа	Руководство администратора (по ГОСТ 19.503-79)
Язык документа	Русский
Дата утверждения	05.08.2026

Документ является эксплуатационным программным документом по ГОСТ 19.101-77, оформлен по ГОСТ 19.106-78, предназначен для включения в состав заявочных материалов в единый реестр российского программного обеспечения (Постановление Правительства РФ от 16.11.2015 № 1236, Приказ Минцифры России от 22.09.2020 № 486).

АННОТАЦИЯ

Настоящий документ — «Инструкция по установке и развёртыванию АИС «Орбита», версия 1.0.0» — описывает порядок установки, настройки, администрирования, резервного копирования, восстановления, обновления, мониторинга и обеспечения информационной безопасности автоматизированной информационной системы «Орбита» (далее — Система, ПО, АИС «Орбита»).

Документ предназначен для системных администраторов организаций, эксплуатирующих Систему, а также для специалистов, ответственных за защиту информации и обеспечение бесперебойной работы информационной инфраструктуры.

Документ разработан в соответствии с требованиями:

- ГОСТ 19.503-79 «ЕСПД. Руководство системного программиста. Требования к содержанию и оформлению»;
- ГОСТ 19.106-78 «ЕСПД. Требования к программным документам, выполненным печатным и графическим способами»;
- ГОСТ 34.601-90 «Автоматизированные системы. Стадии создания»;
- Приказ Минцифры России от 22.09.2020 № 486 «Об утверждении методических рекомендаций...»;
- Постановление Правительства РФ от 16.11.2015 № 1236 «Об установлении запрета на допуск программного обеспечения, происходящего из иностранных государств...»;
- Федеральный закон от 27.07.2006 № 149-ФЗ «Об информации, информационных технологиях и о защите информации»;

- Федеральный закон от 27.07.2006 № 152-ФЗ «О персональных данных»;
- Федеральный закон от 06.04.2011 № 63-ФЗ «Об электронной подписи»;
- Федеральный закон от 26.07.2017 № 187-ФЗ «О безопасности критической информационной инфраструктуры Российской Федерации».

Документ является частью комплекта эксплуатационной документации и сопровождается «Руководством пользователя» АИС «Орбита» v1.0.0.

СОДЕРЖАНИЕ

1. Назначение и условия применения
2. Структура программы
3. Требования к техническим и программным средствам
4. Установка программы
5. Настройка программы
6. Управление учётными записями
7. Управление правами доступа (ролевой моделью)
8. Интеграция со средством электронной подписи (63-ФЗ)
9. Настройка канала уведомлений (СМС-шлюз)
10. Резервное копирование
11. Восстановление после сбоев
12. Обновление программы
13. Мониторинг и диагностика
14. Обеспечение защиты персональных данных (152-ФЗ)
15. Обеспечение информационной безопасности (149-ФЗ, 187-ФЗ)
16. Типовые неисправности и способы их устранения
17. Поддержка жизненного цикла программного обеспечения
18. Сведения о соответствии законодательству Российской Федерации
19. Нормативные и технические ссылки
20. Термины, определения и сокращения

Приложение А. Состав контейнеров Docker и публикуемых портов. Приложение Б. Перечень переменных конфигурации `app/.env`. Приложение В. Состав сведений, регистрируемых в журнале аудита.

1. НАЗНАЧЕНИЕ И УСЛОВИЯ ПРИМЕНЕНИЯ

1.1 Назначение программы

АИС «Орбита» предназначена для комплексной автоматизации деятельности реабилитационного центра (в том числе социально-реабилитационного центра, центра реабилитации детей и подростков, иных организаций, оказывающих реабилитационные услуги населению): учёта клиентов (воспитанников / подопечных) и их персональных данных, формирования, редактирования и подписания договоров и актов выполненных работ, ведения маршрутных листов реабилитации, планирования расписания процедур, кабинетов и специалистов, работы врачебных комиссий, ведения стоп-листа и листа ожидания, ведения коммерческой деятельности в части платных услуг, архивного хранения скан-копий документов, журналирования подписаний и аудита действий пользователей.

При использовании Системы в организации, оказывающей медицинскую помощь (при наличии соответствующей лицензии), дополнительно применяются требования Федерального закона от 21.11.2011 № 323-ФЗ «Об основах охраны здоровья граждан в Российской Федерации». В типовой установке (для социально-реабилитационных центров) данный закон не применяется.

1.2 Функциональные возможности

Система обеспечивает выполнение следующих функций:

1. ведение реестра клиентов (персональные данные, документы, контакты, законные представители, сопровождающие);
2. оформление, редактирование и подписание договоров на оказание реабилитационных услуг;
3. оформление, редактирование и подписание актов выполненных работ;
4. формирование маршрутных листов реабилитации;
5. ведение расписания: слоты процедур, кабинеты, специалисты, листы ожидания, отчёты по загрузке;
6. проведение заседаний врачебных комиссий, фиксация решений, блокировка слотов;
7. ведение стоп-листа и листа ожидания;
8. ведение коммерческой деятельности по платным услугам (договоры, акты, тарифы, подписание);
9. хранение скан-копий документов;
10. создание и редактирование шаблонов документов с подстановкой данных клиента;
11. управление учётными записями, отделами, ролями и правами;
12. автоматическое и регламентное резервное копирование;
13. журналирование подписаний (способ, телефон, IP, отпечаток сертификата, время);
14. журналирование действий пользователей (аудит) с историей изменений;
15. встроенное редактирование документов «Орбита Офис» без передачи документа во внешние сервисы;
16. обезличивание персональных данных перед передачей копий Системы третьим лицам.

1.3 Условия применения

Система применяется в государственных и муниципальных организациях, осуществляющих реабилитационную деятельность (социально-реабилитационные центры, центры реабилитации детей и подростков, центры социальной адаптации), в качестве средства автоматизации учёта клиентов, документооборота и планирования расписания. Система разворачивается полностью в контуре заказчика; обращения к внешним облачным сервисам не осуществляются, за исключением СМС-шлюза (по выбору администратора).

2. СТРУКТУРА ПРОГРАММЫ

2.1 Архитектура

Система имеет трёхуровневую клиент-серверную архитектуру и разворачивается в контейнерах Docker:

Уровень	Компонент	Технология
Клиентский	Браузер пользователя	React 18, Vite, MUI, TypeScript (SPA)
Сервер приложений (основной модуль)	Orbita.Api	.NET 8 (C#), ASP.NET Core, EF Core

Уровень	Компонент	Технология
Сервер приложений (модуль «Платные услуги»)	Platno.API	.NET 8 (C#), ASP.NET Core, EF Core
СУБД	PostgreSQL 16	Две базы: <code>orbita_db</code> , <code>platno</code>
Обратный прокси / TLS	nginx	Маршрутизация, завершение HTTPS
Подписание КЭП	КриптоПро CSP 5.0 + КриптоПро ЭЦП Browser plug-in	ГОСТ Р 34.10-2012 (на рабочем месте пользователя)
Контейнеризация	Docker Engine 24+	Воспроизводимое развёртывание

2.2 Состав контейнеров

Полный перечень контейнеров, их назначение и порты приведены в Приложении А.

2.3 Схема баз данных

Система использует две независимые базы данных PostgreSQL 16:

- `orbita_db` — основная БД (учёт клиентов, договоры, акты, расписание, комиссии, скан-копии, шаблоны, аудит, пользователи, отделы, права);
- `platno` — БД модуля «Платные услуги» (договоры и акты платных услуг, тарифы, шаблоны, журнал подписаний).

Каждая база данных обслуживается отдельным контейнером PostgreSQL (`orbita-db`, `orbita-platno-db`). Миграция схемы выполняется автоматически при старте сервера приложений на основе миграций Entity Framework Core.

3. ТРЕБОВАНИЯ К ТЕХНИЧЕСКИМ И ПРОГРАММНЫМ СРЕДСТВАМ

3.1 Требования к серверу (контейнерный хост)

Параметр	Минимум	Рекомендуется
ОС	Windows 10/11, Windows Server 2019+, Linux (любой дистрибутив с Docker Engine 24+)	Linux (Astra Linux, ALT Linux, РЕД ОС, Ubuntu, Debian) / Windows Server
Docker	Docker Engine 24+ с плагином compose v2	Docker Engine 26+
CPU	4 ядра	8 ядер
ОЗУ	8 ГБ	16 ГБ
Диск	40 ГБ свободно	100 ГБ SSD
Сеть	100 Мбит/с	1 Гбит/с

3.2 Требования к рабочему месту пользователя

- браузер: Яндекс Браузер, Chromium GOST, Google Chrome, Microsoft Edge актуальной версии;
- разрешение экрана: 1366×768 и выше;
- доступ к серверу Системы по HTTPS (по умолчанию порт 18443) или HTTP (18090);
- для подписания КЭП — КриптоПро CSP 5.0 (R2+), КриптоПро ЭЦП Browser plug-in, сертификат КЭП в личном хранилище.

3.3 Требования к программным средствам сервера

- Docker Engine 24.0+ (плагин compose v2);
- утилита `openssl` (для генерации самоподписанного сертификата; если отсутствует, используется одноимённая утилита внутри контейнера `orbita-nginx`);
- в ОС Astra Linux SE «Смоленск» при включённом мандатном контроле PARSEC — утилита `pdpl-file` для выставления мандатных меток (см. п. 16).

4. УСТАНОВКА ПРОГРАММЫ

4.1 Состав дистрибутива

Дистрибутив АИС «Орбита» v1.0.0 поставляется в виде архива со следующей структурой:

<code>app/</code>	– исходные коды и конфигурация (<code>compose</code> , <code>backend</code> , <code>frontend</code> , <code>nginx</code>)
<code>data/orbita_db.dump</code>	– дамп основной БД
<code>data/platno.dump</code>	– дам БД модуля «Платные услуги»
<code>data/volumes/*.tar.gz</code>	– файловые тома: шаблоны, договоры, сканы, бэкапы, загрузки
<code>images/local-images.tar.gz</code>	– образы приложения (<code>orbita-api</code> , <code>orbita-frontend</code> , <code>orbita-platno-api</code>)
<code>images/base-images.tar.gz</code>	– базовые образы (<code>postgres:16-alpine</code> , <code>nginx:alpine</code> , <code>alpine</code>)
<code>restore.ps1</code>	– скрипт установки для Windows (PowerShell 5.1+)
<code>restore.sh</code>	– скрипт установки для Linux / macOS (bash)
<code>generate-env.ps1</code>	– генератор секретов конфигурации
<code>anonymize-db.ps1</code>	– обезличивание БД (152-ФЗ)
<code>docs/</code>	– настоящее руководство и руководство пользователя
<code>README.md</code>	– общее описание

4.2 Подготовка к установке

1. Убедиться в выполнении требований п. 3.
2. Установить Docker (для Astra Linux: `sudo apt install docker.io docker-compose-plugin`; для Windows: Docker Desktop с WSL2-backend; для macOS: Docker Desktop).
3. Добавить пользователя, от имени которого выполняется установка, в группу `docker` (на Linux): `sudo usermod -aG docker $USER`, после чего перелогиниться.
4. На Astra Linux SE / hardened-конфигурациях при необходимости снять атрибут неизменяемости со скрипта: `sudo chattr -i restore.sh`; `chmod +x restore.sh`.
5. Распаковать архив в папку **без кириллицы и пробелов** в пути.

4.3 Генерация секретов конфигурации

Перед первым запуском необходимо сгенерировать криптостойкие значения секретов и записать их в `app/.env`:

```
powershell -ExecutionPolicy Bypass -File .\generate-env.ps1
```

Скрипт создаёт/перезаписывает `app/.env` и обновляет `app/.env.example`. Ключевые секреты:

- `POSTGRES_PASSWORD`, `PLATNO_DB_PASSWORD` — пароли СУБД;
- `JWT_KEY`, `PLATNO_JWT_SECRET` — секреты подписи JWT-токенов;
- `SEED_ADMIN_PASSWORD` — пароль учётной записи `admin` при первом входе (используется также в качестве `VITE_PLATNO_ADMIN_PASSWORD` на этапе сборки фронтенда, см. п. 5.2).

Сохраните значение `SEED_ADMIN_PASSWORD` в защищённом месте (KMS, сейф, защищённый парольный менеджер). Передача значения третьим лицам и хранение в открытом виде запрещены.

4.4 Запуск установки

Из корня распакованного архива:

```
# Windows
powershell -ExecutionPolicy Bypass -File .\restore.ps1

# Linux / Astra Linux / macOS
chmod +x restore.sh
./restore.sh
```

Скрипт выполняет последовательность шагов:

1. проверка окружения (Docker, свободные порты, диск);
2. создание сети `orbita_default`;
3. загрузка образов из `images/local-images.tar.gz` и `images/base-images.tar.gz`; при несовпадении архитектуры — `docker compose pull && docker compose build` (при наличии доступа к Docker Hub);
4. генерация самоподписанного TLS-сертификата в `app/nginx/ssl/`;
5. запуск контейнеров `orbita-db` и `orbita-platno-db`, восстановление БД из `data/orbita_db.dump` и `data/platno.dump`;
6. развёртывание файловых томов (шаблоны, договоры, сканы, бэкапы, загрузки);
7. пересборка контейнера `orbita-frontend` с актуальным `VITE_PLATNO_ADMIN_PASSWORD` (см. п. 5.2);
8. запуск всех сервисов;
9. финальный `docker compose restart` для применения всех настроек.

4.5 Проверка работоспособности после установки

1. Проверить состояние контейнеров:

```
docker compose -f app\docker-compose.yml ps
```

Все контейнеры `orbita-*` должны иметь состояние `running` и `healthy`.

2. Проверить доступ к API основного модуля (HTTP 200, тело `{"status":"ok"}` или аналог):

```
https://localhost:18443/api/health
```

3. Проверить доступ к веб-интерфейсу:

```
https://localhost:18443
```

4. Войти под учётной записью `admin` с паролем из `app/.env` (`SEED_ADMIN_PASSWORD`) и немедленно сменить пароль в разделе «Пользователи → Профиль» (см. Руководство пользователя).
5. Открыть раздел «Настройки → Платные услуги» и убедиться, что подразделы «Пользователи», «Права ролей», «Отделения и подписанты», «Шаблоны», «Дашборды», «Архив базы», «Лицензия», «Аудит», «Журнал ЭП» заполнены данными.

4.6 Удаление программы

Действие	Команда	Эффект
Остановка	<code>docker compose -f app\docker-compose.yml down</code>	Контейнеры остановлены, тома сохранены
Полное удаление	<code>docker compose -f app\docker-compose.yml down -v</code>	Контейнеры и тома удалены (данные уничтожены)
Удаление образов	<code>docker image prune -a</code>	Освобождает диск

Перед полным удалением убедитесь, что выполнено актуальное резервное копирование (п. 10).

5. НАСТРОЙКА ПРОГРАММЫ

5.1 Переменные конфигурации

Все секреты и параметры задаются в `app/.env` и подставляются в контейнеры при запуске. Полный перечень переменных приведён в Приложении Б.

Наиболее важные:

Параметр	Назначение
<code>POSTGRES_PASSWORD</code>	Пароль пользователя <code>orbita_user</code> БД <code>orbita_db</code>
<code>PLATNO_DB_PASSWORD</code>	Пароль пользователя <code>platno_user</code> БД <code>platno</code>
<code>SEED_ADMIN_PASSWORD</code>	Пароль учётной записи <code>admin</code> модуля «Платные услуги» при первом запуске (см. п. 5.2)
<code>JWT_KEY</code>	Секрет подписи JWT-токенов основного модуля (минимум 32 байта энтропии)
<code>PLATNO_JWT_SECRET</code>	Секрет подписи JWT-токенов модуля «Платные услуги»
<code>CORS_ALLOWED_ORIGINS</code>	Список разрешённых источников (Origin) веб-клиента (через запятую)
<code>SIGNING_PAGE_URL</code>	Внешний URL страницы подписания, подставляемый в СМС-ссылки
<code>SIGNING_TTL_HOURS</code>	Срок жизни ссылки на подписание (часы, по умолчанию 24)
<code>SMS_PROVIDER</code> , <code>SMS_API_KEY</code> , <code>SMS_SENDER</code> , <code>SMS_GATEWAY_URL</code>	Параметры СМС-шлюза
<code>BACKUP_INTERVAL_MINUTES</code> , <code>BACKUP_RETENTION_DAYS</code>	Интервал и срок хранения автоматических бэкапов

5.2 Согласование `SEED_ADMIN_PASSWORD` с фронтендом

`SEED_ADMIN_PASSWORD` используется одновременно в двух местах:

- (а) как сидовый пароль администратора модуля «Платные услуги» (читается контейнером `orbita-platno-api` из `env` при старте);
- (б) как `VITE_PLATNO_ADMIN_PASSWORD`, зашиваемый в JS-бандл фронтенда на этапе `docker build`.

Для согласования этих значений в `app/frontend/Dockerfile` значение ARG `VITE_PLATNO_ADMIN_PASSWORD` записывается в `.env.production`, откуда Vite автоматически читает его при сборке. Так как Docker инвалидирует кэш слоя `RUN echo` при изменении ARG, пересборка с новым `SEED_ADMIN_PASSWORD` гарантированно даёт свежий бандл. Скрипты `restore.ps1/restore.sh` (сборка 1.0.0+) на шаге 5/6 всегда выполняют пересборку, поэтому при стандартной установке из архива значения согласованы автоматически.

Если администратор вручную изменил `SEED_ADMIN_PASSWORD` в `app/.env` уже после установки, необходимо синхронизировать фронтенд:

```
cd app
docker compose build frontend
docker compose up -d frontend
```

5.3 Редактирование конфигурации

После изменения `app/.env` необходимо перезапустить затронутые контейнеры:

```
cd app
docker compose restart api frontend platno-api
```

Полный перезапуск всего стека: `docker compose restart`.

5.4 Правила обращения с файлом `app/.env`

- Файл содержит секреты (пароли СУБД, ключи JWT, пароль администратора); не включайте его в состав резервных копий, передаваемых третьим лицам.
- Ограничьте права доступа к файлу: `chmod 600 app/.env` (Linux) / соответствующие ACL в Windows.
- Храните за пределами каталогов, доступных веб-серверу.
- При компрометации любого секрета — немедленно сгенерируйте новый (`generate-env.ps1`) и перезапустите контейнеры.

5.5 Самоподписанный TLS-сертификат

При первом запуске в `app/nginx/ssl/` создаётся самоподписанный сертификат (CN = IP-адрес сервера или `localhost`, SAN: IP/DNS). Для промышленной эксплуатации необходимо заменить его на сертификат российского удостоверяющего центра (УЦ Минцифры или аккредитованного УЦ), выданный по ГОСТ Р 34.10-2012. Замена:

1. поместить `cert.pem` и `key.pem` в `app/nginx/ssl/`;
2. перезагрузить nginx: `docker exec orbita-nginx nginx -s reload`.

6. УПРАВЛЕНИЕ УЧЁТНЫМИ ЗАПИСЯМИ

6.1 Учётная запись администратора

Учётная запись `admin` создаётся при первом запуске модуля «Платные услуги» с паролем из `SEED_ADMIN_PASSWORD`. Учётная запись администратора основного модуля создаётся миграцией `Orbita.Api` (логин `admin`, пароль `admin123` — **обязательно сменить при первом входе**, см. п. 6.4).

6.2 Создание учётных записей

1. Войти в Систему с правами администратора.
2. Открыть раздел «Пользователи».
3. Нажать «Создать пользователя», заполнить:
 - ФИО (обязательно);
 - логин (уникальный, латиницей, рекомендуется ≤ 32 символов);
 - пароль (минимум 8 символов, рекомендуется ≥ 12 с буквами разного регистра, цифрами и спецсимволами);
 - отдел (из справочника «Отделы»);
 - роль (из перечня ролей);
 - контактный e-mail (опционально).
4. Сохранить запись.

6.3 Редактирование, блокировка, разблокировка

В карточке пользователя доступны:

- редактирование ФИО, отдела, роли, e-mail;

- блокировка (запрет входа без удаления записи);
- разблокировка ранее заблокированной учётной записи;
- сброс пароля (см. п. 6.4);
- завершение активных сеансов (инвалидация выданных JWT).

Все действия фиксируются в журнале аудита.

6.4 Сброс пароля

Сброс пароля администратором выполняется в карточке пользователя. При сбросе система генерирует временный пароль (или использует введенный администратором), который **обязан** быть передан пользователю по защищённому каналу и сменён при первом входе.

Собственный пароль пользователь может сменить в разделе «Профиль».

6.5 Удаление учётной записи

Удаление записи пользователя невозможно, если в Системе имеются связанные документы (договоры, акты, подписания). В этом случае учётная запись блокируется (см. п. 6.3). Полное удаление с очисткой связанных данных не предусмотрено для обеспечения целостности аудита и архива документов.

7. УПРАВЛЕНИЕ ПРАВАМИ ДОСТУПА (РОЛЕВОЙ МОДЕЛЮ)

7.1 Принципы

Система использует ролевую модель доступа (RBAC). Каждому пользователю назначается одна роль; роль определяет набор разрешений на разделы и действия. Набор разрешений настраивается в разделе «Права доступа» и хранится в таблице `role_permissions` БД `orbita_db`.

7.2 Состав встроенных ролей

Роль	Назначение
Администратор	Полный доступ ко всем разделам, включая настройки, бэкапы, обезличивание
Врач	Карточки клиентов, договоры, акты, маршрутные листы, расписание
Регистратор	Клиенты, расписание, лист ожидания, договоры (без удаления)
Специалист	Расписание, маршрутные листы, процедуры
Бухгалтер / кассир	Договоры, акты, печатные формы, отчёты
Оператор платных услуг	Модуль «Платные услуги» (договоры, акты, тарифы)

Состав ролей может быть расширен администратором.

7.3 Настройка разрешений

1. Открыть раздел «Права доступа».
2. Выбрать роль.
3. Для каждого раздела/действия установить флажок разрешения.
4. Сохранить.

Изменения применяются немедленно для последующих запросов пользователя; ранее выданные JWT остаются действительными до истечения срока.

8. ИНТЕГРАЦИЯ СО СРЕДСТВОМ ЭЛЕКТРОННОЙ ПОДПИСИ (63-ФЗ)

8.1 Правовое основание

Система поддерживает подписание документов простой электронной подписью (ПЭП) в форме подтверждения по СМС-коду и квалифицированной электронной подписью (КЭП) по ГОСТ Р 34.10-2012 в соответствии с Федеральным законом от 06.04.2011 № 63-ФЗ «Об электронной подписи». Средство КЭП — КристоПро CSP — устанавливается на **рабочем месте пользователя** (не на сервере Системы).

8.2 Подписание КЭП

1. Установить КристоПро CSP 5.0 (R2 или новее) с действующей лицензией.
2. Установить КристоПро ЭЦП Browser plug-in, разрешить его в браузере.
3. Установить сертификат КЭП в личное хранилище («КристоПро CSP» → «Сервис» → «Просмотреть сертификаты в контейнере» → «Установить»). Для корневых сертификатов УЦ — «КристоПро CSP» → «Сервис» → «Установить корневой сертификат».
4. В браузере открыть страницу подписания документа и разрешить доступ плагина по запросу.
5. Система фиксирует результат подписания (отпечаток, субъект, издатель сертификата, дата/время) в карточке документа и в журнале подписаний.

8.3 Подписание ПЭП (СМС-код)

1. Настроить СМС-шлюз (п. 9).
2. При подписании Система отправляет клиенту СМС со ссылкой на страницу подписания и одноразовым кодом.
3. Клиент переходит по ссылке, вводит код, подтверждает.
4. Система фиксирует телефон, IP, дату/время, способ подписания.

8.4 Хранение результатов

Сведения о каждом подписании (документ, способ, телефон/IP, отпечаток сертификата КЭП, дата/время, успех/отказ) сохраняются в журнале подписаний и недоступны для изменения. Журнал доступен администратору и предназначен для расследования инцидентов и предъявления в качестве доказательной базы (ст. 6, 9 63-ФЗ).

9. НАСТРОЙКА КАНАЛА УВЕДОМЛЕНИЙ (СМС-ШЛЮЗ)

9.1 Параметры

В `app/.env` задаются:

Параметр	Назначение
<code>SMS_PROVIDER</code>	Идентификатор провайдера (<code>smsru</code> , <code>smsaero</code> , <code>mts</code> , <code>beeline</code> , иной)
<code>SMS_API_KEY</code>	API-ключ провайдера
<code>SMS_SENDER</code>	Имя отправителя (зарегистрированное у провайдера)
<code>SMS_GATEWAY_URL</code>	URL API провайдера (для провайдеров с произвольным API)

9.2 Проверка

1. После изменения параметров: `docker compose -f app\docker-compose.yml restart api`.
2. Открыть карточку любого договора → «Отправить код подписания».
3. Убедиться в получении СМС клиентом.

4. При отсутствии СМС — проверить баланс у оператора и журнал `docker logs orbita-api` (раздел `SmsService`).

При отсутствии СМС-шлюза Система сохраняет работоспособность остальных модулей; подписание ПЭП будет недоступно.

10. РЕЗЕРВНОЕ КОПИРОВАНИЕ

10.1 Встроенное автоматическое копирование

В разделе «Настройки → Резервное копирование» доступны:

- включение/отключение автоматического копирования;
- интервал (по умолчанию 1440 минут = 24 часа);
- срок хранения копий (по умолчанию 30 суток).

Файлы резервных копий (`*.gzbackup`) создаются в томе `orbita_backups_data` (внутри контейнера `orbita-api` — каталог `/app/backups`). Содержимое копии: дампы основной БД `orbita_db` в сжатом виде, дампы БД `platno`, снимок файловых томов (договоры, сканы, шаблоны).

10.2 Регламентное копирование (рекомендация для соответствия требованиям по защите ИСПДн)

Рекомендуется ежедневно сохранять (помимо встроенного):

1. Дампы БД (с хранением вне сервера Системы):

```
docker exec orbita-db sh -c "pg_dump -U orbita_user -d orbita_db -Fc" >
/backup/orbita_$(date +%F).dump
docker exec orbita-platno-db sh -c "pg_dump -U platno_user -d platno -Fc" >
/backup/platno_$(date +%F).dump
```

2. Файловые тома (для каждого из `orbita_templates_data`, `orbita_contracts_data`, `orbita_scans_data`, `orbita_backups_data`, `orbita_platno_uploads`):

```
docker run --rm -v orbita_templates_data:/src -v /backup:/dst alpine \
tar czf /dst/templates_$(date +%F).tar.gz -C /src .
```

3. Конфигурация — `app/.env` (хранить отдельно, в зашифрованном виде, например в архиве с паролем: `zip -e /backup/env_$(date +%F).zip app/.env`).

4. Журнал аудита и журнал подписаний — выгружаются средствами API в CSV (раздел «Аудит», «Журнал ЭП» → «Экспорт»).

Хранение резервных копий — на территориально удалённом носителе (NAS, лента, зашифрованное облако) с разграничением доступа. Минимальный срок хранения — в соответствии с приказами по учёту документов организации (как правило, не менее 5 лет; для ПДн — не менее срока обработки соответствующих ПДн).

11. ВОССТАНОВЛЕНИЕ ПОСЛЕ СБОЕВ

11.1 Восстановление БД

```
# 1. Запустить только БД
docker compose -f app/docker-compose.yml up -d db platno-db

# 2. Дождаться healthy
docker compose -f app/docker-compose.yml ps

# 3. Восстановить дампы
```

```
docker cp /backup/orbita_2026-08-05.dump orbita-db:/tmp/d.dump
docker exec orbita-db pg_restore -U orbita_user -d orbita_db \
  --no-owner --clean --if-exists /tmp/d.dump

docker cp /backup/platno_2026-08-05.dump orbita-platno-db:/tmp/d.dump
docker exec orbita-platno-db pg_restore -U platno_user -d platno \
  --no-owner --clean --if-exists /tmp/d.dump
```

11.2 Восстановление файловых томов

```
for vol in orbita_templates_data orbita_contracts_data orbita_scans_data \
  orbita_backups_data orbita_platno_uploads; do
  docker run --rm -v $vol:/dst -v /backup:/src:ro alpine \
    sh -c "rm -rf /dst/* && tar xzf /src/${vol}_2026-08-05.tar.gz -C /dst"
done
```

11.3 Запуск полного состава

```
docker compose -f app/docker-compose.yml up -d
```

11.4 Контроль целостности

После восстановления:

1. `docker compose ps` — все контейнеры `healthy`.
2. Войти под `admin`, открыть «Клиенты», «Договоры», «Акты» — данные на месте.
3. Открыть «Настройки → Платные услуги» — все подразделы заполнены.
4. Проверить записи в «Аудит» за дату, предшествующую сбою.
5. Проверить журнал подписаний.

12. ОБНОВЛЕНИЕ ПРОГРАММЫ

12.1 Порядок обновления

1. Выполнить резервное копирование (п. 10).
2. Остановить Систему: `docker compose -f app/docker-compose.yml down` (тома сохраняются).
3. Заменить исходные коды в `app\` файлами новой версии (либо распаковать новый архив поверх с сохранением `app/.env`).
4. Пересобрать и запустить:

```
cd app
docker compose build
docker compose up -d
```

5. Структура БД обновляется автоматически при старте API (миграции EF Core).
6. После запуска — выполнить проверки п. 4.5.

12.2 Откат

1. Остановить Систему.
2. Восстановить БД из резервной копии (п. 11.1).
3. Развернуть исходные коды предыдущей версии в `app\`.
4. `docker compose -f app/docker-compose.yml build && docker compose up -d`.

13. МОНИТОРИНГ И ДИАГНОСТИКА

13.1 Команды оперативного мониторинга

```
# Состояние контейнеров
docker compose -f app/docker-compose.yml ps

# Журналы сервиса
docker logs orbita-api -f --tail=200
docker logs orbita-platno-api -f --tail=200
docker logs orbita-db --tail=200
docker logs orbita-nginx --tail=200

# Проверка доступности БД
docker exec orbita-db pg_isready -U orbita_user -d orbita_db
docker exec orbita-platno-db pg_isready -U platno_user -d platno

# Проверка использования ресурсов
docker stats
```

13.2 Журналирование

Серверы приложений используют Serilog (уровень по умолчанию `Information`). Журналы включают временную метку, уровень, идентификатор запроса (`correlation id`), сведения о пользователе и контекст операции.

13.3 Встроенный редактор документов

Редактор документов «Орбита Офис» (редактирование DOCX, конвертация DOCX ↔ HTML и DOCX → PDF) встроен в серверы `orbita-api` и `orbita-platno-api` (библиотека PdfSharpCore, шрифты Liberation). Отдельный контейнер не требуется; при неполадках проверять `docker logs orbita-api` и `docker logs orbita-platno-api`.

13.4 Индикаторы работоспособности

Индикатор	Норма
<code>docker compose ps</code>	<code>orbita-api, orbita-platno-api, orbita-frontend, orbita-nginx — Up (healthy); orbita-db, orbita-platno-db — Up (healthy)</code>
HTTP <code>GET /api/health</code>	200 OK
<code>pg_isready</code>	<code>accepting connections</code>
Доступность веб-интерфейса	<code>https://<адрес-сервера>:18443</code> открывается, форма входа отображается

14. ОБЕСПЕЧЕНИЕ ЗАЩИТЫ ПЕРСОНАЛЬНЫХ ДАННЫХ (152-ФЗ)

14.1 Правовое основание

Система обрабатывает персональные данные клиентов (воспитанников, подопечных) и сотрудников, являясь информационной системой обработки персональных данных (ИСПДн). Защита обеспечивается в соответствии с Федеральным законом от 27.07.2006 № 152-ФЗ «О персональных данных», Постановлением Правительства РФ от 01.11.2012 № 1119 «Об утверждении требований к защите персональных данных при их обработке в информационных системах персональных данных», Приказом ФСТЭК России от 18.02.2013 № 21 «Об утверждении Состав и содержания организационных и технических мер по обеспечению безопасности персональных данных...».

14.2 Организационные меры

- утверждение перечня обрабатываемых ПДн и оснований обработки;

- определение уровня защищённости ИСПДн (УЗ-1/УЗ-2/УЗ-3/УЗ-4) по результатам модели угроз;
- определение угроз безопасности ПДн (актуальные угрозы 2-го или 3-го типа по Приказу ФСТЭК № 21);
- ознакомление пользователей с политикой обработки ПДн;
- ограничение доступа к ПДн ролевой моделью (п. 7).

14.3 Технические меры, реализованные в Системе

- аутентификация пользователей по логину/паролю с блокировкой после заданного числа неуспешных попыток (настраивается);
- JWT-токены с ограниченным сроком действия (access — 15 минут; refresh — 14 суток);
- ролевая модель доступа (п. 7);
- журналирование действий пользователей (Приложение В);
- фиксация результатов подписания документов (телефон, IP, отпечаток сертификата КЭП, дата/ время);
- TLS-шифрование канала между клиентом и сервером (п. 5.5);
- хранение паролей в БД в виде криптографического хеша (PBKDF2, bcrypt или Argon2 в зависимости от модуля);
- раздельное хранение БД основного модуля и БД модуля «Платные услуги» (разные учётные данные, разные контейнеры).

14.4 Технические меры, реализуемые администратором

- настройка межсетевого экрана (iptables/nftables): ограничение доступа к портам 18081 (API), 15433 (БД), 18082 (frontend direct) — эти порты **не должны** быть опубликованы наружу; внешний доступ — только через 18090/18443 (nginx);
- ограничение доступа к Docker socket (/var/run/docker.sock) только для администратора;
- размещение Системы в выделенном сегменте сети (VLAN);
- антивирусная защита узла;
- контроль целостности дистрибутива (sha256-суммы архива — указаны в `VERSION`).

14.5 Обезличивание персональных данных

Перед передачей копий Системы третьим лицам (тестирование, демонстрация, аудит) необходимо выполнить обезличивание БД:

```
powershell -ExecutionPolicy Bypass -File .\anonymize-db.ps1
```

Скрипт во временном изолированном контейнере восстанавливает дампы, заменяет ПДн синтетическими значениями (ФИО, телефоны, паспорта, адреса, IP, данные журналов) и пересоздаёт `data/orbita_db.dump` и `data/platno.dump`. Рабочие контейнеры Системы не затрагиваются.

15. ОБЕСПЕЧЕНИЕ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ (149-ФЗ, 187-ФЗ)

15.1 Соответствие 149-ФЗ

Система соответствует требованиям Федерального закона от 27.07.2006 № 149-ФЗ «Об информации, информационных технологиях и о защите информации» в части:

- использования только сертифицированных средств защиты (КриптоПро CSP — сертификат ФСБ России);

- обеспечения целостности хранимой информации (аудит, журнал подписаний);
- обеспечения конфиденциальности (TLS, ролевая модель, JWT);
- обеспечения доступности (резервирование, п. 10–11).

15.2 Категорирование и КИИ (187-ФЗ)

Если организация, эксплуатирующая Систему, отнесена к субъектам критической информационной инфраструктуры (КИИ) в соответствии с Федеральным законом от 26.07.2017 № 187-ФЗ «О безопасности критической информационной инфраструктуры Российской Федерации», Система подлежит категорированию как объект КИИ. Категорирование выполняется в установленном порядке с учётом Постановления Правительства РФ от 08.02.2018 № 127 и Приказа ФСТЭК России от 21.12.2017 № 235. Состав средств защиты определяется присвоенной категорией значимости.

15.3 Применение СКЗИ

Подписание КЭП по ГОСТ Р 34.10-2012 выполняется сертифицированным СКЗИ «КриптоПро CSP» (сертификат ФСБ России). Ключи КЭП хранятся на защищённых носителях (eToken, Рутокен, JaCarta) в соответствии с требованиями к средствам КЭП.

16. ТИПОВЫЕ НЕИСПРАВНОСТИ И СПОСОБЫ ИХ УСТРАНЕНИЯ

№	Симптом	Вероятная причина	Решение
1	port is already allocated при docker compose up	Порт занят другим стеком	Остановить посторонний стек, либо сменить порты в app/docker-compose.yml; синхронизировать CORS_ALLOWED_ORIGINS, SIGNING_PAGE_URL в app/.env и \$cors_origin в app/nginx/nginx.conf
2	Не открывается https://<адрес>:18443 (Windows)	Не установлен WSL2	wsl --install от имени администратора, перезагрузка, включить WSL2 backend в Docker Desktop
3	Ошибка no such host при docker compose pull	Нет доступа к Docker Hub (офлайн-установка)	Нормально: образы уже загружены из архива. Игнорировать (try/catch в restore.sh / restore.ps1 сборки 1.0.0+)
4	Браузер блокирует соединение по HTTPS (самоподписанный сертификат)	Ожидается для автономного стенда	Использовать http://<адрес>:18090 или добавить исключение в браузер
5	Astra Linux: Permission denied при запуске restore.sh	Атрибут неизменяемости	sudo chattr -i restore.sh; chmod +x restore.sh
6	orbita-api не стартует	БД не готова	Дождаться healthy у orbita-db: docker compose ps; API имеет healthcheck, ожидающий БД
7	PowerShell 5.1: InvalidVariableReferenceWithDrive	Баг парсера PS с кириллицей	Сохранить restore.ps1 в UTF-8 (ВОМ допустим); в сборке 1.0.0 скрипт переведён в латиницу
8	Вкладка «Платные услуги» в Настройках пустая (нет пользователей/шаблонов),	Рассинхронизация VITE_PLATNO_ADMIN_PASSWORD	При стандартной установке из архива не возникает (шаг 5/6 в

№	Симптом	Вероятная причина	Решение
	хотя API возвращает данные	(в бандле) и SEED_ADMIN_PASSWORD (в БД)	restore.sh / restore.ps1 1.0.0+). При ручной смене пароля: cd app && docker compose build frontend && docker compose up -d frontend
9	Astra Linux SE: контейнеры не видят друг друга по DNS	PARSEC / hardened nftables блокируют правила Docker	sudo update-alternatives --set iptables /usr/sbin/iptables-legacy && sudo systemctl restart docker ; либо разрешить подсеть Docker (172.16.0.0/12) в системном nftables
10	Astra Linux SE: тома не монтируются в контейнер	PARSEC блокирует доступ по мандатным меткам	sudo pdpl-file 0:0:0:0 /var/lib/docker/volumes/orbita_* и sudo pdpl-file 0:0:0:0 /var/lib/docker , перезапустить Docker и restore.sh
11	Не доставляются СМС	Неверные параметры шлюза / баланс	Проверить SMS_API_KEY , SMS_GATEWAY_URL , баланс у оператора; docker logs orbita-api (раздел SmsService)
12	Не работает подписание КЭП	Плагин КриптоПро не установлен / заблокирован	Установить КриптоПро CSP 5.0, КриптоПро ЭЦП Browser plug-in; разрешить плагин в браузере; проверить сертификат в личном хранилище

17. ПОДДЕРЖКА ЖИЗНЕННОГО ЦИКЛА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

17.1 Стадии жизненного цикла

Жизненный цикл ПО АИС «Орбита» соответствует ГОСТ Р ИСО/МЭК 12207-2010 и ГОСТ 34.601-90 и включает стадии:

1. формирование требований (ТЗ);
2. проектирование архитектуры;
3. реализация (кодирование, модульное тестирование);
4. интеграция и квалификационные испытания;
5. поставка заказчику (выпуск дистрибутива);
6. эксплуатация (установка, настройка, сопровождение);
7. сопровождение (исправление дефектов, выпуск обновлений);
8. снятие с эксплуатации (при необходимости).

17.2 Гарантийная поддержка

Правообладатель обеспечивает:

- исправление критических дефектов (блокирующих работу Системы) в течение 5 рабочих дней с момента обращения;
- исправление прочих дефектов в срок до 30 календарных дней (в плановых релизах);
- консультационную поддержку по вопросам установки, настройки и эксплуатации (e-mail, телефон, указанные в п. 17.4);

- выпуск обновлений безопасности по мере необходимости (оперативно, в течение 5 рабочих дней);
- выпуск плановых обновлений с новой функциональностью не реже 1 раза в год.

17.3 Совместимость с будущими версиями

Правообладатель обеспечивает совместимость обновлений в пределах одной мажорной версии (1.x). Переход на новую мажорную версию (2.0.0) сопровождается миграционными инструкциями и, при необходимости, миграционными скриптами БД.

17.4 Каналы поддержки

Канал	Значение
E-mail	azharkov73@gmail.com
Телефон	8-999-971-30-31
Документация	Дистрибутив (docs/) + ИНСТРУКЦИЯ.txt

17.5 Лицензионная политика

Код АИС «Орбита» — собственная разработка правообладателя. Используются компоненты с открытым исходным кодом: .NET Runtime (MIT), React (MIT), PostgreSQL (PostgreSQL License), nginx (BSD), Docker-образы alpine/postgres/nginx. КриптоПро CSP — коммерческий продукт ООО «КРИПТО-ПРО», лицензируется пользователем самостоятельно.

18. СВЕДЕНИЯ О СООТВЕТСТВИИ ЗАКОНОДАТЕЛЬСТВУ РОССИЙСКОЙ ФЕДЕРАЦИИ

Нормативный документ	Область соответствия	Реализация в Системе
Федеральный закон от 27.07.2006 № 149-ФЗ «Об информации...»	Защита информации, целостность, конфиденциальность, доступность	TLS, ролевая модель, аудит, резервное копирование
Федеральный закон от 27.07.2006 № 152-ФЗ «О персональных данных»	Обработка и защита ПДн	Согласие, минимизация, ролевая модель, аудит, обезличивание, разделение доступа
Федеральный закон от 06.04.2011 № 63-ФЗ «Об электронной подписи»	ЭП (ПЭП и КЭП)	ПЭП по СМС-коду; КЭП через КриптоПро CSP по ГОСТ Р 34.10-2012; журнал подписаний
Федеральный закон от 21.11.2011 № 323-ФЗ «Об основах охраны здоровья граждан в РФ»	Учёт и отчётность в сфере здравоохранения	Применяется только при использовании Системы в организации, оказывающей медицинскую помощь (при наличии лицензии). В типовой установке для социально-реабилитационного центра не применяется.
Федеральный закон от 26.07.2017 № 187-ФЗ «О безопасности КИИ РФ»	Безопасность КИИ (при категорировании)	Поддержка категорирования, организационные и технические меры
Постановление Правительства РФ от 16.11.2015 № 1236	Запрет на допуск иностранного ПО	Включение в реестр российского ПО
Приказ Минцифры России от 22.09.2020 № 486	Методические рекомендации по описанию ПО для реестра	Настоящий документ и описание ПО подготовлены по рекомендациям

Нормативный документ	Область соответствия	Реализация в Системе
Приказ ФСТЭК России от 18.02.2013 № 21	Состав и содержание мер защиты ПДн	Реализованные меры — п. 14.3
Приказ ФСТЭК России от 11.02.2013 № 17 (для ГИС)	Защита ГИС (при использовании в ГИС)	Реализованные меры — п. 14, 15
Приказ ФСТЭК России от 21.12.2017 № 235	Категорирование КИИ	Процедура категорирования — п. 15.2
ГОСТ 19.503-79	Руководство системного программиста	Настоящий документ
ГОСТ 19.505-79	Руководство оператора (пользователя)	Руководство пользователя (отдельный документ)
ГОСТ 34.601-90	Стадии создания АС	Жизненный цикл — п. 17
ГОСТ Р ИСО/МЭК 12207-2010	Процессы жизненного цикла ПО	п. 17

19. НОРМАТИВНЫЕ И ТЕХНИЧЕСКИЕ ССЫЛКИ

- Федеральный закон от 27.07.2006 № 149-ФЗ «Об информации, информационных технологиях и о защите информации».
- Федеральный закон от 27.07.2006 № 152-ФЗ «О персональных данных».
- Федеральный закон от 06.04.2011 № 63-ФЗ «Об электронной подписи».
- Федеральный закон от 26.07.2017 № 187-ФЗ «О безопасности критической информационной инфраструктуры Российской Федерации».
- Постановление Правительства РФ от 16.11.2015 № 1236 «Об установлении запрета на допуск...».
- Постановление Правительства РФ от 01.11.2012 № 1119 «Об утверждении требований к защите персональных данных...».
- Постановление Правительства РФ от 08.02.2018 № 127 «Об утверждении Правил категорирования объектов КИИ...».
- Приказ Минцифры России от 22.09.2020 № 486 «Об утверждении методических рекомендаций...».
- Приказ ФСТЭК России от 18.02.2013 № 21.
- Приказ ФСТЭК России от 11.02.2013 № 17.
- Приказ ФСТЭК России от 21.12.2017 № 235.
- ГОСТ 19.101-77, ГОСТ 19.106-78, ГОСТ 19.201-78, ГОСТ 19.402-78, ГОСТ 19.503-79, ГОСТ 19.505-79.
- ГОСТ 34.601-90.
- ГОСТ Р ИСО/МЭК 12207-2010.
- ГОСТ Р 34.10-2012 (электронная подпись).
- Документация PostgreSQL 16 — <https://www.postgresql.org/docs/16/>.
- Документация Docker Engine — <https://docs.docker.com/engine/>.

20. ТЕРМИНЫ, ОПРЕДЕЛЕНИЯ И СОКРАЩЕНИЯ

Термин / сокращение	Определение
АИС	Автоматизированная информационная система
API	Application Programming Interface — программный интерфейс приложения
БД	База данных
ГИС	Государственная информационная система
ЕСПД	Единая система программной документации (ГОСТ 19 серии)
ИСПДн	Информационная система персональных данных
КИИ	Критическая информационная инфраструктура
КЭП	Квалифицированная электронная подпись (ГОСТ Р 34.10-2012)
Минцифры	Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации
НПА	Нормативный правовой акт
ПДн	Персональные данные
ПО	Программное обеспечение
ПЭП	Простая электронная подпись (по 63-ФЗ)
СКЗИ	Средство криптографической защиты информации
СМС	Short Message Service — служба коротких сообщений
СУБД	Система управления базами данных
УЦ	Удостоверяющий центр
ФСТЭК	Федеральная служба по техническому и экспортному контролю
CAdES	CMS Advanced Electronic Signatures (формат подписи)
JWT	JSON Web Token
PBKDF2 / bcrypt / Argon2	Алгоритмы хеширования паролей
RBAC	Role-Based Access Control — управление доступом на основе ролей
Serilog	Библиотека структурированного логирования для .NET
TLS	Transport Layer Security — протокол защищённой передачи данных

Приложение А. СОСТАВ КОНТЕЙНЕРОВ DOCKER И ПУБЛИКУЕМЫХ ПОРТОВ

Контейнер	Базовый образ	Назначение	Внутренние порты	Публикуемые порты (по умолчанию)
orbita-db	postgres:16-alpine	БД <code>orbita_db</code>	5432	15433
orbita-platno-db	postgres:16-alpine	БД <code>platno</code>	5432	—
orbita-api	mcr.microsoft.com/dotnet/aspnet:8.0 + Liberation	REST API основного модуля	8080	18081
orbita-platno-api	mcr.microsoft.com/dotnet/aspnet:8.0 + Liberation	REST API модуля «Платные услуги»	5000	— (через nginx)

Контейнер	Базовый образ	Назначение	Внутренние порты	Публикуемые порты (по умолчанию)
orbita-frontend	nginx:alpine	Статический SPA	80	18082
orbita-nginx	nginx:alpine	Обратный прокси, TLS	80, 443	18090 (HTTP), 18443 (HTTPS)

Контейнер `orbita-platno-api` доступен изнутри сети `orbita_default` по имени `platno-api:5000` и публикуется наружу только через `orbita-nginx` (путь `/platno-api/...`).

Приложение Б. ПЕРЕЧЕНЬ ПЕРЕМЕННЫХ КОНФИГУРАЦИИ `app/.env`

Переменная	Назначение	Пример
<code>POSTGRES_USER</code>	Имя пользователя основной БД	<code>orbita_user</code>
<code>POSTGRES_PASSWORD</code>	Пароль основной БД	(генерируется)
<code>POSTGRES_DB</code>	Имя основной БД	<code>orbita_db</code>
<code>PLATNO_DB_USER</code>	Имя пользователя БД модуля «Платные услуги»	<code>platno_user</code>
<code>PLATNO_DB_PASSWORD</code>	Пароль БД модуля «Платные услуги»	(генерируется)
<code>PLATNO_DB_NAME</code>	Имя БД модуля «Платные услуги»	<code>platno</code>
<code>SEED_ADMIN_PASSWORD</code>	Пароль <code>admin</code> модуля «Платные услуги» при первом входе	(генерируется)
<code>JWT_KEY</code>	Секрет подписи JWT основного модуля	(генерируется, ≥32 байт)
<code>JWT_ISSUER</code>	Издатель JWT	<code>orbita</code>
<code>JWT_AUDIENCE</code>	Аудитория JWT	<code>orbita-web</code>
<code>PLATNO_JWT_SECRET</code>	Секрет подписи JWT модуля «Платные услуги»	(генерируется)
<code>CORS_ALLOWED_ORIGINS</code>	Разрешённые источники (через запятую)	<code>https://localhost:18443</code>
<code>SIGNING_PAGE_URL</code>	URL страницы подписания	<code>https://localhost:18443/signing</code>
<code>SIGNING_TTL_HOURS</code>	Срок жизни ссылки на подписание	<code>24</code>
<code>SMS_PROVIDER</code>	Провайдер СМС	<code>smsru</code>
<code>SMS_API_KEY</code>	API-ключ СМС-провайдера	—
<code>SMS_SENDER</code>	Имя отправителя СМС	<code>ORBITA</code>
<code>SMS_GATEWAY_URL</code>	URL API СМС-провайдера	—
<code>BACKUP_INTERVAL_MINUTES</code>	Интервал автоматических бэкапов (минуты)	<code>1440</code>
<code>BACKUP_RETENTION_DAYS</code>	Срок хранения автоматических бэкапов (дни)	<code>30</code>
<code>VITE_PLATNO_ADMIN_PASSWORD</code>	Значение <code>SEED_ADMIN_PASSWORD</code> , передаваемое в <code>docker build</code> для фронтенда	(из <code>SEED_ADMIN_PASSWORD</code>)

Приложение В. СОСТАВ СВЕДЕНИЙ, РЕГИСТРИРУЕМЫХ В ЖУРНАЛЕ АУДИТА

Журнал аудита (`audit_log`) фиксирует:

- дата и время события (UTC, миллисекунды);
- идентификатор пользователя (логин);
- тип операции (CREATE / UPDATE / DELETE / LOGIN / LOGOUT / SIGN / EXPORT / ADMIN);
- идентификатор сущности и её тип (Client, Contract, Act, User, Department, ...);
- набор изменённых полей с предыдущим и новым значением (для UPDATE);
- IP-адрес клиента;
- идентификатор корреляции запроса (correlation id);
- признак успешности операции;
- текст ошибки (при неуспехе).

Журнал подписаний (`signature_journal_entries`) фиксирует:

- дата и время подписания (UTC, миллисекунды);
- идентификатор документа и его тип;
- идентификатор пользователя, инициировавшего подписание;
- способ подписания (PES_SMS / KEP_CRYPTOPRO);
- телефон клиента (для ПЭП);
- IP-адрес клиента (для ПЭП);
- отпечаток сертификата КЭП, субъект и издатель (для КЭП);
- признак успешности;
- текст ошибки (при неуспехе).

Хранение журналов — в БД `orbita_db` и `platno` соответственно. Срок хранения — не менее 5 лет; экспорт в CSV/JSON — через API.

© Жарков Александр, 2026. Документ является частью заявочных материалов для включения АИС «Орбита» в единый реестр российского программного обеспечения.

ПРИЛОЖЕНИЕ А. КРАТКАЯ ИНСТРУКЦИЯ ПО УСТАНОВКЕ ДИСТРИБУТИВА

АИС «ОРБИТА» — ПЕРЕНОС И ЗАПУСК НА НОВОМ ПК (v1.0.0 от 05.08.2026)

Полный автономный комплект для развёртывания системы через Docker на Windows 10/11, Linux (Ubuntu/Debian, в т.ч. Astra Linux) и macOS (Intel и Apple Silicon).

1. СОСТАВ АРХИВА

`app/` — исходники и конфигурация (`docker-compose.yml`, `backend`, `frontend`, платный модуль, `nginx`)
`data/orbita_db.dump`, `data/platno.dump` — дампы баз данных PostgreSQL (основная и «Платные услуги»)
`data/volumes/*.tar.gz` — файловые данные: шаблоны, договоры, сканы, резервные копии, загрузки платного модуля
`images/local-images.tar.gz` — образы приложения: `orbita-api`, `orbita-frontend`, `orbita-platno-api`
`images/base-images.tar.gz` — базовые образы: `postgres:16-alpine`, `nginx:alpine`, `alpine restore.ps1` — автоустановка для Windows (PowerShell) `restore.sh` — автоустановка для Linux / macOS (bash) `VERSION` — сведения о версии релиза

2. ТРЕБОВАНИЯ

Общие для всех систем:

- Docker Engine 24+ с плагином docker compose v2;
- Docker ЗАПУЩЕН до начала установки;
- свободно ~6 ГБ на диске (образы + данные + тома);
- свободные порты (по умолчанию): 18090, 18443, 18081, 18082, 15433.
- если на ПК уже работают другие Docker-стеки (например, gz-* на 8090/8443/8081/8082/5433) — порты можно сдвинуть; см. п. 7.1.

Интернет:

- если архитектура ПК совпадает с архитектурой образов в архиве — интернет НЕ нужен;
- если отличается (например, архив собран на x86_64, а ПК — ARM или наоборот) — скрипт сам соберёт образы приложений из исходников (5–15 минут), а базовые образы postgres/nginx скачает из интернета (нужен доступ к Docker Hub).

Как установить Docker:

Windows 10/11 — Docker Desktop с backend WSL2 (wsl --install при необходимости) Ubuntu / Debian — `sudo apt update && sudo apt install -y docker.io docker-compose-plugin` Astra Linux — `sudo apt install docker.io docker-compose-plugin`; пользователя добавить в группу docker: `sudo usermod -aG docker $USER`, затем перелогиниться macOS — Docker Desktop для Mac (Intel или Apple Silicon)

3. БЫСТРЫЙ ЗАПУСК (РЕКОМЕНДУЕТСЯ)

Распакуйте архив в любую папку БЕЗ КИРИЛЛИЦЫ И ПРОБЕЛОВ в пути.

Windows: откройте PowerShell ИЗ ПАПКИ АРХИВА и выполните:

```
powershell -ExecutionPolicy Bypass -File .\restore.ps1
```

Linux / Astra Linux / macOS: из папки архива в терминале:

```
chmod +x restore.sh
./restore.sh
```

Скрипт выполняет: создание сети -> загрузка образов из архива (или сборка/скачивание при несовпадении архитектуры) -> генерация самоподписанного TLS-сертификата при отсутствии -> запуск БД -> восстановление дампов и файловых томов -> запуск всех сервисов.

По завершении:

- Вход: <http://localhost:18090> (или <https://localhost:18443> — самоподписанный сертификат)
- Логин: admin, пароль: admin123 (по умолчанию; задаётся переменной SEED_ADMIN_PASSWORD в файле app/.env и автоматически синхронизируется при старте контейнеров)

ВАЖНО: при первом запуске порты автоматически сдвинуты на 18090/18443/ 18081/18082/15433, чтобы не конфликтовать с другими Docker-стеками, которые могут уже работать на 8090/8443/8081/8082/5433. Если эти порты тоже свободны, верните прежние значения в app/docker-compose.yml и в app/.env (CORS_ALLOWED_ORIGINS, SIGNING_PAGE_URL).

4. РУЧНАЯ УСТАНОВКА (ЕСЛИ СКРИПТ НЕДОСТУПЕН)

Все команды — из папки app/:

1. сеть

```
docker network inspect orbita_network >/dev/null 2>&1 || docker network create orbita_network
```

2. образы из архива

```
docker load -qi ../images/local-images.tar.gz docker load -qi ../images/base-images.tar.gz
```

если архитектура не совпадает — docker compose pull && docker compose build

3. запуск

```
docker compose up -d
```

4. восстановление БД (после готовности db)

```
docker cp ../data/orbita_db.dump orbita-db:/tmp/d.dump docker exec orbita-db sh -c "pg_restore -U orbita_user -d orbita_db --no-owner --clean --if-exists /tmp/d.dump" docker cp ../data/platno.dump orbita-platno-db:/tmp/d.dump docker exec orbita-platno-db sh -c "pg_restore -U platno_user -d platno --no-owner --clean --if-exists /tmp/d.dump"
```

5. файловые тома (для каждого: backups_data, templates_data,

contracts_data, scans_data, platno_uploads)

если архитектура ПК отличается от архива — сначала скачайте

нативный alpine: docker pull alpine

```
docker volume create orbita_templates_data # пример docker run --rm -v orbita_templates_data:/dst -v ../data/volumes:/src:ro alpine sh -c "tar xzf /src/templates_data.tar.gz -C /dst"
```

Имена томов: orbita_backups_data, orbita_templates_data, orbita_contracts_data, orbita_scans_data, orbita_platno_uploads.

5. АДРЕСА И ПОРТЫ

По умолчанию (сдвинуты относительно стандартных 8090/8443/8081/8082/5433, чтобы не конфликтовать с уже работающими Docker-стеками):

Веб-интерфейс (через nginx) — <http://localhost:18090> и <https://localhost:18443> Фронтенд напрямую — <http://localhost:18082> API основной — <http://localhost:18081> PostgreSQL основной — localhost:15433 (БД orbita_db, пользователь orbita_user)

При смене опубликованных портов обязательно синхронизируйте CORS_ALLOWED_ORIGINS и SIGNING_PAGE_URL в app/.env и карту \$cors_origin в app/nginx/nginx.conf (иначе фронтенд не сможет обращаться к API).

Пароли и ключи задаются в app/.env (SEED_ADMIN_PASSWORD, POSTGRES_PASSWORD, PLATNO_DB_PASSWORD, JWT_KEY и т.д.) — при необходимости измените до первого запуска.

ВАЖНО: значение SEED_ADMIN_PASSWORD используется сразу в двух местах: (а) как сидовый пароль администратора модуля «Платные услуги» (в контейнере orbita-platno-api — читается из env при старте); (б) как VITE_PLATNO_ADMIN_PASSWORD, зашиваемый в JS-бандл фронтенда на этапе docker build. restore.ps1 / restore.sh в сборке 1.0.0+ на шаге 5/6 всегда пересобирает frontend с актуальным SEED_ADMIN_PASSWORD из app/.env (Docker переиспользует загруженный образ как кэш, поэтому шаг

занимает менее минуты). Это значит, что при стандартной установке из архива пароль во фронте всегда будет соответствовать паролю в БД Platno, какое бы значение `SEED_ADMIN_PASSWORD` ни было задано в `app/.env`.

Если же администратор вручную меняет `SEED_ADMIN_PASSWORD` в `.env` после установки (например, поменял пароль админу в БД и хочет синхронизировать), пересоберите фронт вручную: `cd app docker compose build frontend docker compose up -d frontend`

6. ПОВСЕДНЕВНЫЕ КОМАНДЫ

`cd app`

`docker compose ps` # статус контейнеров `docker compose logs -f api` # логи сервиса (`api` / `frontend` / `# platno-api` / `nginx` / `db`) `docker compose restart` # перезапуск `docker compose down` # остановить (данные сохраняются # в томах) `docker compose up -d` # запустить снова `docker compose down -v` # остановить и ПОЛНОСТЬЮ удалить # данные

7. ЧАСТЫЕ ПРОБЛЕМЫ

1. Порт занят (`port is already allocated`) — остановите другое приложение, либо измените порты в `app/docker-compose.yml`. После смены портов обязательно обновите `app/.env` (`CORS_ALLOWED_ORIGINS`, `SIGNING_PAGE_URL`) и карту `$cors_origin` / редирект в `app/nginx/nginx.conf`, иначе CORS-ответы и HTTPS-редирект перестанут работать.
2. WSL2 не установлен (Windows) — выполните `wsl --install` от имени администратора, перезагрузитесь, включите WSL2 backend в настройках Docker Desktop.
3. Нет доступа к Docker Hub при несовпадении архитектуры — перенесите архив, собранный на ПК той же архитектуры, либо заранее выполните `docker compose pull` на ПК с интернетом и передайте образы через `docker save`.
4. Браузер ругается на самоподписанный сертификат (<https://localhost:18443>) — это ожидаемо для автономного стенда; используйте <http://localhost:18090> или добавьте исключение.
5. Astra Linux: при запрете запуска скриптов снимите атрибуты: `sudo chattr -i restore.sh; chmod +x restore.sh`
6. Контейнер `api` не стартует — убедитесь, что БД `healthy` (`docker compose ps`); API ждёт готовности БД по `healthcheck`.
7. PowerShell 5.1 ругается на синтаксис `restore.ps1` ("InvalidVariableReferenceWithDrive" в строках с кириллицей) — это известный баг Windows PowerShell с эм-/ен-дешами и многострочными строками. В сборке 1.0.0 скрипт уже переведён в латиницу и использует `${name}` вместо `$name` рядом с двоеточием. Если ошибка появляется на старых версиях, проверьте кодировку файла: `restore.ps1` должен быть сохранён в UTF-8 (BOM допустим).
8. Если `restore.ps1` прерывается на "`docker compose pull ... no such host`" — это нормально для офлайн-установки; образы уже загружены из архива. В сборке 1.0.0 `pull` обернут в `try/catch` и не валит установку.
9. Вкладка «Платные услуги» в разделе Настройки выглядит пустой (нет пользователей, шаблонов, услуг, подписантов), хотя БД заполнена и API возвращает данные при ручном запросе токена. Причина: в собранном JS-бандле `orbita-frontend` «зашит» пароль авто-логина (`VITE_PLATNO_ADMIN_PASSWORD`), который не совпадает с текущим `SEED_ADMIN_PASSWORD` в `app/.env` — как правило, это случается, если администратор вручную изменил `SEED_ADMIN_PASSWORD` в `.env` после установки и забыл пересобрать фронт. В

результате PlatnoAuthProvider получает 401 от /platno-api/auth/login, user.permissions остаётся пустым, и PlatnoSettingsPage скрывает все табы. Решение — из app/ пересобрать фронт: `cd app`
`docker compose build frontend`
`docker compose up -d frontend`
В restore.ps1 / restore.sh (сборка 1.0.0+) этот шаг выполняется автоматически при установке из архива (шаг 5/6).

10. Astra Linux SE «Смоленск» / hardened-конфигурации. На стандартной Astra Linux CE «Орёл» restore.sh отработывает без дополнительных действий. На Astra SE с включённым мандатным контролем (PARSEC) или на Astra CE с hardened-профилем («усиленный» режим ядра, кастомный nftables) возможны два специфичных сбоя:

(а) Контейнеры не видят друг друга по DNS-именам сервисов (например, orbita-api не резолвит db). Причина: ядро со строгими nftables/iptables блокирует правила, которые Docker добавляет в цепочку DOCKER-USER. Диагностика: `docker network inspect orbita_default | grep Subnet`
`iptables -S | grep -i docker` Если цепочка пуста или правила не применены, временно переключите iptables в легаси-режим: `sudo update-alternatives --set iptables /usr/sbin/iptables-legacy`
`sudo systemctl restart docker` Если в организации iptables-legacy запрещён политикой, добавьте разрешающие правила для подсети Docker (обычно 172.16.0.0/12) в системный nftables и перезапустите Docker.

(б) Тома и файлы, создаваемые Docker, недоступны из контейнера (видны пустыми, не монтируются). Причина: PARSEC блокирует доступ к точке монтирования из-за мандатных меток. Решение: выставить на хостовой папке/томе нулевые метки: `sudo pdpl-file 0:0:0:0 /var/lib/docker/volumes/orbita_*` либо для docker-root: `sudo pdpl-file 0:0:0:0 /var/lib/docker` После перезапустите сервис Docker и повторите ./restore.sh.

(в) restore.sh падает на шаге восстановления файловых томов с «apk: not found» или «exec format error». Причина: alpine из архива собран под другую архитектуру, а эмуляция (binfmt) на hardened-ядре запрещена. Решение: `sudo docker pull alpine` и перезапустите ./restore.sh — он подтянет нативный образ.

В логах установки соответствующие строки помечены как «обновление образа alpine под amd64» и «pdpl-file» — можно проверить, что они действительно выполнились.

8. ОСОБЕННОСТИ ВЕРСИИ 1.0.0

- Редактор документов «Орбита Офис» (собственная разработка, без внешних зависимостей): лента в стиле MS Word (Главная / Вставка / Макет / Таблица / Вид), линейка с маркерами отступов, галерея стилей, форматная кисть, контекстное меню, разбивка по страницам, колонтитулы, точная печать (@page под формат/поля), экспорт DOCX. автономно.

9. ФОРМАЛЬНАЯ ДОКУМЕНТАЦИЯ (ДЛЯ РЕЕСТРА РОССИЙСКОГО ПО)

Документация, оформленная по ГОСТ 19 серии (ЕСПД) и пригодная для включения в состав заявочных материалов в единый реестр российского программного обеспечения (Постановление Правительства РФ от 16.11.2015 № 1236, Приказ Минцифры России от 22.09.2020 № 486), находится в каталоге docs/ дистрибутива:

docs/administrator-guide.md — Руководство администратора (по ГОСТ 19.503-79). Установка, настройка, резервное копирование, восстановление, обновление, мониторинг, защита персональных данных (152-ФЗ), интеграция с КриптоПро CSP (63-ФЗ), поддержка жизненного цикла ПО. Word-версия: docs/administrator-guide.docx

docs/user-guide.md — Руководство пользователя (по ГОСТ 19.505-79). Вход, навигация, работа с модулями, подписание ПЭП/КЭП, аварийные ситуации, обязанности пользователя по 152-ФЗ. Word-версия: docs/user-guide.docx

При внесении правок в .md-файлы пересоберите .docx-версии командой: `python _md2docx.py docs\administrator-guide.md docs\administrator-guide.docx python _md2docx.py docs\user-guide.md docs\user-guide.docx` (требуется Python 3.8+ и пакет `python-docx`; в Windows ставится как: `pip install python-docx`).

В обоих документах приведены: титульный лист с реквизитами ПО и правообладателя, аннотация, содержание, перечень функциональных возможностей, требования к техническим и программным средствам, ссылки на 149-ФЗ, 152-ФЗ, 63-ФЗ, 187-ФЗ, 323-ФЗ, приказы ФСТЭК № 17, 21, 235, постановления Правительства № 1119, 1236, 127, а также сведения о поддержке жизненного цикла по ГОСТ Р ИСО/МЭК 12207-2010 и ГОСТ 34.601-90.