

# Digital Audit System

Система цифрового аудита технических систем безопасности (АПС, СОУЭ, СКУД, СОР и др.)



## Возможности

- **Norm Library** - Нормативная библиотека с версионированием (ПК/ПФ/INT)
- **Requirement Sets** - Каталог требований с импортом/экспортом XLSX
- **Pre-Audit Setup** - Мастер предаудита с freeze baseline
- **Field App** - Полевой ввод данных (чек-листы, дефекты, фото, протоколы)
- **Review Console** - Контроль качества главным аудитором
- **QC System** - Автоматический контроль качества (DRAFT/PRELIMINARY/FINAL)
- **Report Generator** - Генерация пакета DOC-01...DOC-08 (PDF/XLSX)



## Пакет отчетов

### Обязательные (6 документов):

- **DOC-01:** Executive Summary (PDF)
- **DOC-02:** Technical Report (PDF)
- **DOC-03:** Defects Register (XLSX + PDF)
- **DOC-04:** Normative Compliance Matrix (XLSX + PDF)
- **DOC-05:** Test Protocols (PDF)
- **DOC-06:** Photo Appendix (PDF)

### Опциональные (2 документа):

- **DOC-07:** CAPA / Action Plan (XLSX + PDF)
- **DOC-08:** ROM Budget / Cost Estimate (XLSX + PDF)



## Быстрый старт

### 1. Установка зависимостей

```
npm install
```

### 2. Настройка Supabase

1. Зарегистрируйтесь на [supabase.com](https://supabase.com)
2. Создайте новый проект
3. Скопируйте Database URL из Settings → Database → Connection string (URI)
4. Создайте файл `.env` :

```
DATABASE_URL="postgresql://postgres:[YOUR-PASSWORD]@db.[YOUR-PROJECT-REF].supabase.co:5432/postgres"
```

```
NEXT_PUBLIC_SUPABASE_URL=https://[YOUR-PROJECT-REF].supabase.co
```

```
NEXT_PUBLIC_SUPABASE_ANON_KEY=your-anon-key
```

```
SUPABASE_SERVICE_ROLE_KEY=your-service-role-key # Для парсера
```

```
NEXT_PUBLIC_APP_URL=http://localhost:3000
```

```
# AI API (для парсинга PDF и анализа)
```

```
OPENAI_API_KEY=sk-proj-... # ОБЯЗАТЕЛЬНО! Используется для всех AI операций
```

```
GEMINI_API_KEY=... # НЕ используется (опционально)
```

**⚠ ВАЖНО:** Система использует **ТОЛЬКО OpenAI API** для парсинга PDF и анализа требований.

Gemini API НЕ используется в текущей версии. См. `IMPORTANT_AI_API_INFO.md` для деталей.

### 3. Инициализация базы данных

```
# Создать таблицы
npm run db:push

# Заполнить справочники
npm run db:seed
```

### 4. Запуск приложения

```
npm run dev
```

Откройте <http://localhost:3000>



## Структура базы данных

#### Справочники (Reference Data)

- `systems` - Системы (APS, SOUE, CCTV, ACS, OS, SCS)
- `defect_types` - Типы дефектов
- `severity_levels` - Уровни критичности (CRITICAL/HIGH/MEDIUM/LOW)
- `na_reasons` - Причины неприменимости
- `evidence_types` - Типы доказательств
- `customer_doc_types` - Типы документов заказчика

#### Norm Library

- `norm_sources` - Нормативные источники
- `norm_files` - Файлы нормативов (PDF)

#### Requirement Sets

- `requirement_sets` - Наборы требований
- `requirements` - Требования (REQ-APS-KZ-0042)

#### Audit

- `audits` - Аудиты
- `audit_baselines` - Freeze snapshots
- `audit_norm_snapshots` - Снапшоты нормативов
- `audit_requirement_snapshots` - Снапшоты требований

#### Field & Review

- `locations` - Локации (иерархия)
- `check_items` - Чек-листы проверки
- `defects` - Дефекты
- `evidence` - Доказательства (фото, видео, логи)
- `protocols` - Протоколы испытаний

#### CAPA & Budget

- `capa_actions` - План корректирующих мероприятий
- `estimates` - Сметы
- `estimate_lines` - Позиции смет

#### Pack Generation

- `audit_pack_issues` - Выпущенные пакеты отчетов

## Роли

- **FE** (Field Engineer) - Полевой инженер
- **LA** (Lead Auditor) - Главный аудитор
- **AN** (Analyst) - Аналитик
- **PM** (Project Manager) - Координатор
- **ADMIN** - Администратор

## Технический стек




- **Frontend:** Next.js 15 (App Router) + React 19
- **Backend:** Next.js Server Actions
- **Database:** PostgreSQL (Supabase)
- **ORM:** Prisma
- **Styling:** Tailwind CSS + shadcn/ui
- **PDF Generation:** @react-pdf/renderer
- **XLSX Generation:** exceljs
- **Language:** TypeScript

## Жизненный цикл аудита


1. **Pre-Audit** - Подготовка (scope, нормы, требования, freeze baseline)
2. **Field** - Полевой сбор данных (чек-листы, дефекты, фото, протоколы)
3. **Review** - Офисная обработка (severity, requirement\_id, финализация)
4. **QC** - Контроль качества (валидация данных)
5. **Pack Generation** - Генерация отчетов (DRAFT/PRELIMINARY/FINAL)

## Документация

### Для пользователей:

-  [Инструкция для Нормативщика](#) - полное руководство по добавлению НД
-  [Быстрая памятка для Нормативщика](#) - краткая шпаргалка
-  [Логика конвертации фрагментов](#) - как работает конвертация

### Для разработчиков:

-  [Информация об AI API](#) - какой API используется
- [Техническое задание \(ТЗ\)](#)
- [Пояснительная записка](#)
- [API Reference](#)

## Полезные команды

```
# Разработка
npm run dev           # Запуск dev сервера
npm run build         # Production build
npm run start         # Production сервер

# База данных
npm run db:push       # Применить schema к БД
npm run db:studio     # Открыть Prisma Studio (GUI)
npm run db:seed       # Заполнить справочники
```

# Линтинг

npm run lint

# ESLint проверка



## Лицензия

ISB Инжиниринг © 2026

---

**Версия:** 1.0.0

**Дата:** 23 января 2026