SKAI-Desk.

Руководство по развертыванию.

ООО «Проектная среда»

Автор — Цибряева Екатерина Дмитриевна

Санкт-Петербург, 2025

197972, Санкт-Петербург, ул. Оптиков 4, корп. 3, стр. 1
8 800 250 60 77

office@skai.online
skai.online

История версий документа

Версия документа	Дата внесения изменений	Содержание/Причина изменений	Автор
0.0	03.03.2025	Первоначальная версия документа	Цибряева Е.Д

ОГЛАВЛЕНИЕ

ОПРЕДЕЛЕНИЯ И СОКРАЩЕНИЯ

Карточка проекта	– Инструмент управления проектами, реализованный на SD
Компания-клиент	– Компания, которой Организация предоставляет свои услуги. При создании карточки проекта указывается в поле «Компания»
Организация	– Компания, использующая SKAI-Desk в качестве системы управления проектами
Оснащение ТС	– Процесс монтажа и последующей проверки корректности работы оборудования на ТС. Включает в себя согласование работ, выезд монтажника и последующие проверки выполнения им работ
ПО	– Программное обеспечение
Проект	– Процесс массового оснащения ТС в рамках конкретного договора, под который собирается Проектная команда. Проекты делятся на 4 типа: • «Пилот». Позволяет компании-клиенту протестировать продукт и принять решение о продолжении сотрудничества • «Оснащение». Тип проекта, в рамках которого реализуется основная масса оснащений для компании-клиента (или его первая партия). После завершения оснащения компания становится полноценным клиентом Организации • «Дооснащение». Последующие партии оснащения ТС компании-клиента, если "Оснащение" уже реализовано • «Проект перевода». Проект по переводу уже оснащенных ТС в Организации. Не требует проведения массовых монтажных работ.
Тикет	– Запись в SKAI-Desk, содержащая информацию о проекте, его описании, статусе, ответственных лицах и др.
Тикетная система	– Инструмент для регистрации и управления тикетами
ТС	– Транспортное средство
API	– Application Programming Interface — интерфейс прикладного программирования- описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой
CI/CD	– Continuous Integration, Continuous Delivery — технология автоматической доставки новых версий ПО пользователю на протяжении всего жизненного цикла разработки
CRM	– Customer Relationship Management — это система для управления взаимоотношениями с пользователями, которая позволяет работать с базой, собирать лиды, отслеживать действия пользователей и сотрудников и автоматизировать рутинные операции
ERP	– Enterprise Resource Planning — это комплексное программное обеспечение для упрощения, автоматизации и эффективного управления бизнес-процессами в организации
SaaS	– Software as a Service — модель предоставления программного обеспечения, при которой пользователи получают доступ к софту онлайн. Такие решения ещё называют облачными: вендор сервиса размещает его на своих серверах, берёт на себя обслуживание и развитие, а клиент подключается к уже готовому продукту через интернет

SKAI-Desk (SD)	– Веб-приложение, представляющее собой специализированную тикетную систему для повышения эффективности сервисных процессов
SLA	– Соглашение об уровне услуг (Service-level agreement). Определяет срок, за который проект должен быть реализован (согласно внутренним регламентам и договором с Компанией-клиентом)
SQL	– Structured query language — «язык структурированных запросов». Декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей *СУБД*
S3	– Simple Storage Service — сервис для хранения цифровых данных большого объёма

1 ОБЩИЕ СВЕДЕНИЯ

1.1 Наименование системы

- Полное наименование — Информационная система SKAI-Desk.
- Краткое наименование — SKAI-Desk.

1.2 Сведения о системе

Информационная система **SKAI-Desk** – веб-приложение, представляющее собой специализированную тикетную систему, которая используется для повышения эффективности сервисных процессов, связанных с обслуживанием пользователей.

SKAI-Desk позволяет управлять ходом реализации проектов оснащения ТС для Компании-клиента с помощью регистрируемых тикетов (Карточка Проекта).

Модель предоставления ПО пользователю – SaaS.

Перед эксплуатацией системы пользователь должен ознакомиться с полным пакетом документации на систему, предоставленную разработчиком.

2 ОСНОВНЫЕ ХАРАКТЕРИСТИКИ

2.1 Системные требования

Для корректной работы системы рабочее место пользователя должно соответствовать требованиям:

- **Аппаратные:**
  - Персональный компьютер или ноутбук с установленной операционной системой и подключением к сети Интернет или локальной сети.
  - Процессор: 2,3 ГГц или выше (рекомендуется многопоточный процессор с поддержкой 64-разрядных вычислений).
  - Оперативная память: минимум 4 ГБ (рекомендуется 8 ГБ и выше для комфортной работы).
  - Свободное дисковое пространство: минимум 2 ГБ для хранения временных файлов и кэширования.
  - Разрешение экрана: 1366×768 пикселей или выше (рекомендуется 1920×1080 для лучшего отображения интерфейса).
  - Глубина цвета: 24 бит или выше.
  - Сетевая карта для проводного или беспроводного подключения к Интернету или локальной сети.
  - В случае работы на мобильных устройствах, рекомендуется использование планшетов с экраном от 10 дюймов и выше.
  - Постоянное подключение к сети Интернет или локальной сети:
    - Скорость соединения: не менее 10 Мбит/с (рекомендуется 50 Мбит/с и выше для комфортной работы с веб-приложением).
    - Доступ к протоколам: HTTP(S) (порт 80/443).
- **Программные:**
  - Операционная система:
    - **Windows**: Windows 10 (версии 20H2 и выше), Windows 11 (все актуальные версии), 64-битные версии.
    - **Linux**: Ubuntu (версии 20.04 и выше), Debian (версии 11 и выше), Fedora (версии 38 и выше), Alt Linux (версии 9 и выше), RED OS (версии 7 и выше), Astra Linux (версии 1.9 и выше).
    - **MacOs**: macOS 12 (Monterey) и выше.
  - Веб-браузер:
    - Яндекс Браузер – версия 20 и выше.

- Спутник – версия 2.0 и выше (на базе Chromium).
- Google Chrome – версия 100 и выше.
- Mozilla Firefox – версия 90 и выше.
- Microsoft Edge – версия 100 и выше.
- Safari – версия 14 и выше (для macOS).

- Обязательные настройки веб-браузера:
  - Включенная поддержка JavaScript (требуется для работы интерактивных элементов и динамического обновления контента).
  - Включенная поддержка CSS3 (используется для корректного отображения интерфейса).
  - Включенные cookie (необходимы для хранения пользовательских сессий и персонализации).
  - Разрешены всплывающие окна (требуется для работы модальных окон и уведомлений системы).
  - Разрешены запросы AJAX (используются для динамической загрузки данных без перезагрузки страницы).
  - Поддержка WebSockets (в системе есть функции, работающие в реальном времени, например, обновление данных без обновления страницы).
  - В системе есть загрузка и обработка файлов, убедитесь, что браузер разрешает скачивание файлов.
- Дополнительное программное обеспечение:
  - ПО для чтения PDF-файлов.
  - Docker (если пользователь разрабатывает или тестирует приложение локально).

Важно: Использование устаревших версий браузеров и операционных систем может привести к некорректному отображению интерфейса и сбоям в работе системы.

2.2 Назначение системы и основные функции

SKAI-Desk – веб-приложение для повышения эффективности процессов, связанных с обслуживанием пользователей.

SKAI-Desk позволяет управлять ходом реализации проектов оснащения ТС для Компании-клиента с помощью регистрируемых тикетов (Карточка Проекта).

Задачи системы:

1. Учет и систематизация тикетов

- Фиксация всех тикетов пользователей.
- Присвоение уникального номера (тикета) для отслеживания.
- Классификация тикетов по категориям, приоритетам и типам.

2. Распределение тикетов

- Автоматическое или ручное назначение тикетов на соответствующих специалистов или отделы.
- Распределение нагрузки между сотрудниками.

3. Контроль сроков выполнения

- Установление сроков обработки тикетов.
- Отслеживание времени выполнения задач.

4. Повышение прозрачности процессов

- Отслеживание статуса тикета для пользователей.
- Просмотр истории всех действий по тикету (кто и когда работал над ним).

5. Аналитика и отчетность

- Сбор статистики по количеству тикетов, времени их обработки, загруженности сотрудников.
- Формирование отчетов для анализа эффективности работы над тикетами.

6. Улучшение качества обслуживания

- Стандартизация процессов обработки тикетов.
- Снижение вероятности потери или игнорирования тикетов.

7. Интеграция с другими системами

- Возможность интеграции с CRM, ERP, системами мониторинга и другими инструментами.
- Автоматизация рутинных задач.

8. Сбор обратной связи

- Сбор отзывов от пользователей после закрытия тикета.
- Оценка удовлетворенности пользователей.

2.3 Количественные и качественные характеристики

2.3.1 Состав системы

Состав системы в общем виде приведен в таблице 1.

Таблица 1. Список компонентов системы

Категория	Компонент	Описание
Database SKAI-DESK	PostgreSQL 15.5	Реляционная база данных.
Backend SKAI-DESK	PHP 8.3	Среда выполнения для Laravel
	Laravel Framework 11.9	Основной фреймворк для бэкенда
Frontend SKAI-DESK	Vue.js 3.5	Фреймворк для фронтэнда
	Nuxt 3.15	
External systems (интеграция по API)	Redis	Брокер сообщений для Laravel
	MinIO	Объектное хранилище, совместимое с S3.
	Typesense	Поисковая система для полнотекстового поиска.
	Mailpit	Локальный SMTP-сервер для отладки почты.
	Docker Compose	Управление многоконтейнерным окружением проекта.

Взаимодействие компонентов:

- Laravel работает с PostgreSQL для хранения данных.
- Redis используется Laravel для кэширования и очередей.
- MinIO применяется как S3-хранилище.
- Typesense обеспечивает полнотекстовый поиск.
- Mailpit отлаживает почтовые уведомления.
- Docker Compose управляет всей инфраструктурой, связывая компоненты.

2.3.2 **Структура и функциональность**

Взаимодействие компонентов SKAI-Desk изображена на рисунке 1.

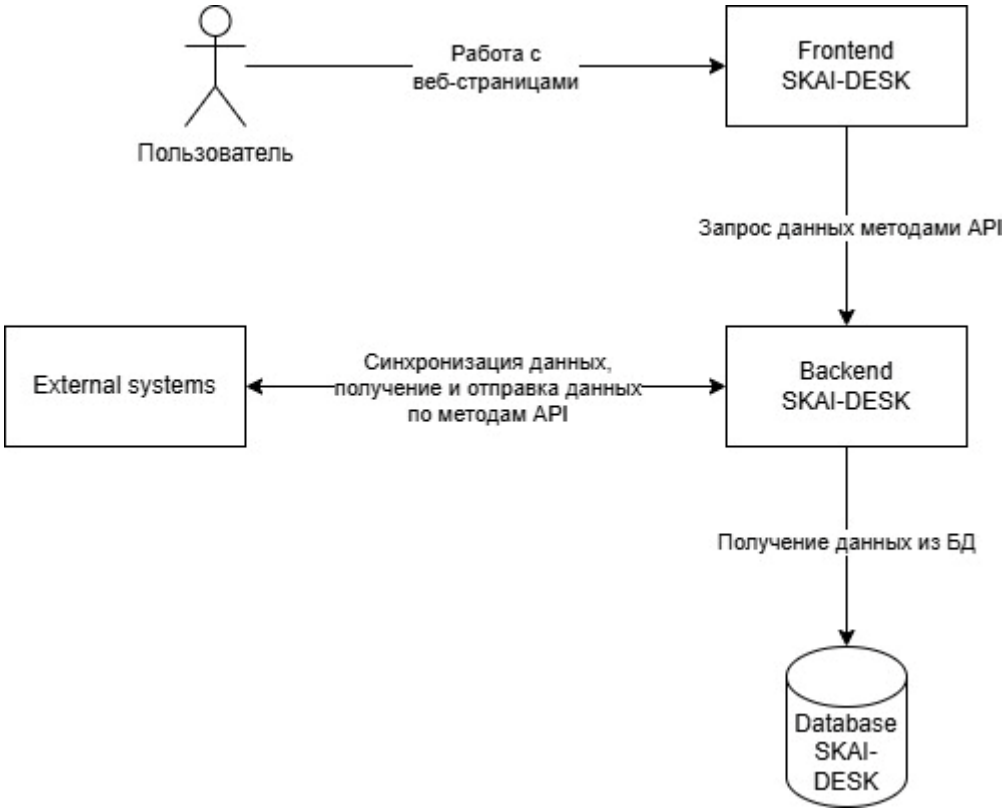

Рисунок 1. Взаимодействие компонентов SKAI-Desk

Система состоит из:

- Database SKAI-DESK: PostgreSQL 15.5.
- Backend SKAI-DESK: PHP 8.3, Laravel Framework 11.9.
- Frontend SKAI-DESK: Vue.js 3.5, Nuxt 3.15.
- External systems: любая система, которая интегрируется со SKAI-Desk.

Пользователь взаимодействует с системой через веб-интерфейс Frontend SKAI-DESK. Веб-интерфейс позволяет пользователю самостоятельно создавать, редактировать и удалять сущности в рамках установленной функциональности.

Для получения информации Frontend SKAI-DESK обращается к Backend SKAI-DESK по средствам методов API.

Backend SKAI-DESK хранит и взаимодействует с Database PostgreSQL, где хранится и изменяется информация.

Backend SKAI-DESK может быть интегрирован с другими системами для синхронизации, получения или передачи информации с помощью API-методов и других типов интеграций в рамках необходимой функциональности.

Интеграции не являются обязательными для функционирования SKAI-Desk.

Система позволяет:

- создавать и редактировать тикет на реализацию проекта (Карточку Проекта);
- клонировать проекты;
- отслеживать текущий статус проекта;
- назначать Проектную команду;
- назначать контактных лиц со стороны Компании-клиента;
- работать со списком проектов, фильтровать и выгружать его;
- настраивать собственный профиль фильтрации списка проектов и сохранять его;
- фиксировать длительность нахождения проекта в каждом статусе для последующего сравнения с SLA.

2.4 Сведения о взаимодействии с другими/внешними системами

Получение и отправка почты в системе происходит с помощью почтового сервера.

Внешние системы по отношению к системе и их описание приведены в таблице 4.

Таблица 1. Список внешних систем

№	Система	Описание
1	S3 Object Storage/Selcloud	Облачное хранилище для файлов и резервных копий
2	GitLab	Хостинг репозитория кода, управление CI/CD
3	SMTP-сервер	Отправка email-уведомлений
4	Grafana/Prometheus	Мониторинг производительности системы
5	Sentry	Мониторинг и логирование ошибок

3 ИНСТРУКЦИЯ ПО РАЗВЕРТЫВАНИЮ ПРОЕКТА

Развертывание системы производится в следующем порядке:

1. Установка Docker и Docker Compose
2. Настройка окружения
3. Развертывание Backend (Laravel)
4. Развертывание Frontend (Nuxt.js)
5. Настройка PostgreSQL
6. Настройка других сервисов
7. Тестирование с использованием Pest
8. Настройка локального SMTP-сервера Mailpit
9. Завершение развертывания проекта

3.1 Установка Docker и Docker Compose

Для установки Docker и Docker Compose на сервер необходимо:

1. Выполнить следующие команды:

- **Для Docker:**

```
curl -fsSL https://get.docker.com -o get-docker.sh
sudo sh get-docker.sh
sudo usermod -aG docker $USER
```

- **Для Docker Compose:**

```
 sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
```

2. Проверьте успешную установку:

- **Для Docker:**

```
docker –version
```

- **Для Docker Compose:**

```
docker-compose –version
```

3.2 Настройка окружения

Для настройки окружения необходимо:

1. Создать файл ***.env*** в корневой директории проекта и добавить необходимые переменные окружения:

- **Для Backend:**

```
APP_DEV_DOMAIN=your-dev-domain
APP_DOMAIN=your-prod-domain
APP_ENV=local
APP_KEY=base64:your-app-key
DB_CONNECTION_PEST=pgsql
DB_DATABASE=your-database
DB_HOST=your-db-host
DB_USERNAME=your-db-username
DB_PASSWORD=your-db-password
DB_PORT=5432
AWS_ACCESS_KEY_ID=your-aws-access-key-id
AWS_SECRET_ACCESS_KEY=your-aws-secret-access-key
```

- **Для Frontend:**

```
APP_LOCALE=en
NUXT_PUBLIC_API_BASE=your-api-url
```

2. Проверить, что все переменные окружения корректно настроены в соответствии с вашими потребностями.

3.3 Развертывание Backend (Laravel)

Для развертывания Backend необходимо:

1. Перейти в директорию с Backend проекта:

```
cd /path/to/your/backend
```

2. Создать Docker контейнер для бэкэнда. Docker Compose автоматически создаст все необходимые сервисы:

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

3. Проверить, что контейнеры запущены:

```
docker-compose ps
```

4. Применить миграции базы данных:

```
docker exec -it your-backend-container-name php artisan migrate –force
```

5. Запустить сервер Laravel Octane;

```
docker exec -it your-backend-container-name php artisan octane:start --host=0.0.0.0 --port=80
```

3.4 Развертывание Frontend (Nuxt.js)

Для развертывания Frontend необходимо:

1. Перейти в директорию с Frontend проекта:

```
cd /path/to/your/frontend
```

2. Собрать проект для разработки:

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

3. Открыть веб-браузер и перейти по адресу, настроенному в переменной `APP_DEV_DOMAIN`

3.5 Настройка PostgreSQL

Для настройки PostgreSQL необходимо:

1. Убедиться, что контейнер с PostgreSQL запущен:

```
docker ps
```

2. Выполнить команду для подключения к базе данных:

```
docker exec -it your-pg-container-name psql -U your-db-username your-database
```

3. Создать базу данных для тестов или разработки, если необходимо:

```
CREATE DATABASE your_database_name;
```

Важно: Для развертывания системы рекомендуется использовать PostgreSQL версии 15.5 и выше.

3.6 Настройка других сервисов

- Для работы с Redis:
  - Контейнер с Redis должен быть запущен автоматически через Docker Compose.
- Если необходимо подключиться к Redis для тестирования или администрирования, используйте:

```
docker exec -it your-redis-container-name redis-cli
```

- Для работы с Typesense:
  - Контейнер с Typesense будет доступен на порту, который вы настроили в переменных окружения.

http://localhost:8108

- Для работы с MinIO:
  - Контейнер с MinIO должен быть доступен через настройки в .env

```
http://localhost:9001
```

3.7 Тестирование с использованием Pest

Для проведения тестирования необходимо:

1. Установить зависимость Pest:

```
composer require pestphp/pest –dev
```

2. Запустить тесты с помощью команды:

```
docker exec -it your-backend-container-name vendor/bin/pest
```

3.8 Настройка локального SMTP-сервера Mailpit

Mailpit используется для отладки почты и доступен по ссылке:

- ***http://localhost:8025*** — веб-интерфейс для просмотра писем.
- ***http://localhost:1025*** — SMTP-сервер, через который приложения отправляют письма.

3.9 Завершение развертывания проекта

Для завершения развертывания проекта необходимо:

Проверить логи контейнеров для выявления возможных ошибок:

```
docker-compose logs -f
```

- Для остановки и очистки контейнеров использовать команду:

```
docker-compose down
```

- **Если необходимо, для повторного запуска использовать команду:**

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

Проект с Laravel, PostgreSQL, Redis, Typesense, MinIO и фронтэнд на Nuxt.js развернут и готов к использованию.

ЭКСПЛУАТАЦИОННАЯ ДОКУМЕНТАЦИЯ

Таблица 1.

Название	Краткое описание
SKAI-Desk. Паспорт	Документ описывает общие сведения о системе, ее характеристиках, структуре, назначении.
SKAI-Desk. Руководство по развертыванию.	Документ описывает порядок получения своей версии SKAI-Desk.
SKAI-Desk. Руководство пользователя.	Данный документ, содержит общую информацию о функциональности и базовых характеристиках системы.
SKAI-Desk. Руководство по Жизненному циклу ПО	Документ описывает набор периодических работ по системе, позволяющих обеспечивать качество функционирования систем.