Software Requirements Specification (SRS)

SaveAndSound –облачное Java-приложение для управления звуковыми ресурсами

1. Введение

1.1 Назначение

Цель данного документа — формализовать требования к серверному приложению SaveAndSound, которое предоставляет REST API для управления звуковыми файлами, пользователями и создателями контента. Приложение будет развернуто в облаке с использованием Docker и CI/CD, и в будущем дополнено фронтендом.

1.2 Область применения

SaveAndSound — это Java-приложение, взаимодействующее с PostgreSQL и контейнеризованное через Docker. Оно предназначено для интеграции с внешними клиентами (веб-интерфейс, мобильные приложения) и может быть развернуто на платформах типа Railway, Render, Heroku.

Приложение реализует ролевую модель доступа, CRUD-операции и логирование действий.

1.3 Определения, акронимы и сокращения

- **REST API** — интерфейс взаимодействия по HTTP
- **CI/CD** — непрерывная интеграция и доставка
- **Docker** — платформа контейнеризации
- **DTO** — Data Transfer Object
- **CRUD** — Create, Read, Update, Delete
- **SaaS / PaaS / IaaS** — модели облачного размещения

1.4 Ссылки

- PostgreSQL Documentation
- Docker Documentation
- GitHub Actions
- OpenAPI Specification

2. Требования пользователя

2.1 Программные интерфейсы

Приложение должно взаимодействовать с:

- PostgreSQL (через JDBC)
- Docker и Docker Compose
- CI/CD-системами (GitHub Actions, Railway)

- Внешними REST-клиентами
- В будущем — с фронтендом через HTTP API

2.2 Интерфейс пользователя

На текущем этапе взаимодействие осуществляется через REST API. В будущем будет реализован веб-интерфейс.

Действие пользователя	Реакция системы
GET /users	Возвращает список пользователей
POST /sounds	Добавляет звуковой файл
PUT /users/{id}	Обновляет данные пользователя
DELETE /sounds/{id}	Удаляет звуковой файл

2.3 Характеристики пользователей

- **Администраторы** — полный доступ к API, управление системой
- **Создатели контента** — добавление и редактирование звуков
- **Пользователи** — просмотр и взаимодействие с контентом

Все группы предполагаются как технически грамотные, работающие через API или веб-интерфейс.

2.4 Предположения и зависимости

- Docker и PostgreSQL доступны в окружении
- CI/CD настроен для автоматического тестирования и деплоя
- API будет использоваться внешними клиентами
- Приложение развёртывается в облаке (Railway, Render, Heroku)
- Фронтенд будет реализован отдельно и подключён к API

3. Системные требования

3.1 Функциональные требования

1. Система должна предоставлять REST API для управления пользователями
2. Система должна предоставлять REST API для управления звуковыми файлами
3. Система должна поддерживать ролевую модель доступа (администратор, создатель, пользователь)
4. Система должна сохранять и извлекать данные из PostgreSQL
5. Система должна быть контейнеризирована с использованием Docker
6. Система должна поддерживать автоматическую сборку и деплой через CI/CD
7. Система должна логировать действия и ошибки
8. Система должна валидировать входные данные
9. Система должна быть готова к интеграции с фронтендом
10. Система должна обеспечивать безопасность доступа к API

3.2 Нефункциональные требования

3.2.1 Атрибуты качества

Атрибут	Почему важен	Как измеряется
Надёжность	Стабильная работа API и базы данных	Uptime > 99%, корректные ответы
Масштабируемость	Возможность облачного развертывания	Поддержка Docker и облачных платформ
Безопасность	Защита данных и контроль доступа	Ролевые ограничения, валидация, шифрование
Портативность	Работа на разных платформах	Docker-образ совместим с Linux/Windows
Поддерживаемость	Лёгкость обновления и тестирования	CI/CD пайплайн, модульная архитектура
Расширяемость	Возможность добавления новых функций	Чёткая структура API, DTO, слоёв приложения
Производительность	Быстрая обработка запросов	Время ответа < 500 мс при стандартной нагрузке