Технические риски

1 Координация разработки и контроль версий

Сложность: конфликты кода при параллельной работе над frontend частью (React) и backend частью (Spring), также между двумя React приложениями (PWA и веб-приложения для администрации).

Решение:

— релизация GitFlow с ветками для feature/bugfix/release. Использование GitHub Actions для автоматических тестов и деплоя;

— разделение PWA и админ-панели на отдельные репозитории с четким определением API-контрактов между frontend и backend (например, через Swagger-документацию).

2 Защита данных и предотвращение уязвимостей

Сложность: риски утечек персональных данных, XSS/CSRF-атак, несанкционированного доступа к API и админ-панели.

Решение:

— настройка Spring Security с активацией CORS-политик, защитой от CSRF и санитизацией пользовательского ввода;

— шифрование чувствительных данных в PostgreSQL.

3 Интеграция AI для генерации описания

Сложность: некорректные или нерелевантные описания из-за неправильно сформулированных промптов, а также зависимость от скорости работы внешнего AI-сервиса.

Решение: проведение тестов с разными вариантами промптов и шаблонами входных данных.

4 Производительность сервера и базы данных при высокой нагрузке

Сложность: замедление обработки запросов при увеличении числа пользователей и объема данных.

Решение:

— контейнеризация серверной части с Docker для упрощения развертывания и ручного масштабирования;

— оптимизация PostgreSQL: создание индексов для часто используемых полей.

5 Согласованность взаимодействия клиентских приложений и сервера

Сложность: обеспечение стабильного обмена данными между PWA и веб-приложением для администрации на React и сервером на Spring Boot.

Решение:

— использование единого REST API с документацией в Swagger для минимизации несоответствий;

— применение JWT-токенов с четкими сроками жизни и refresh-механизмов для аутентификации без сохранения состояния сервера.