Министерство науки высшего образования Российской Федерации

федеральное государственное автономное образовательное учреждение высшего образования

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

(Университет ИТМО)

Факультет информационных технологий и программирования

Дисциплина: Управление требованиями

Лабораторная работа №5.

Создание документа SRS (Software Requirements Specification)

Тема: Бронирование столиков и формирование предварительного заказа в ресторане

Выполнили студенты группы М3311

Авсюкевич Анастасия

Михалев Никита

1. Введение

1.1 Назначение документа

Документ описывает спецификацию требований (SRS) для системы бронирования столиков и формирования предварительного заказа в ресторане. Цель данного документа – предоставить полное и детализированное описание требований к системе бронирования столиков и формирования предварительного заказа в ресторане. Этот документ является основным источником информации для всех участников проекта: от разработчиков до тестировщиков и заказчиков. Он описывает функциональные, нефункциональные требования и интерфейсы, которые будут использоваться для создания и дальнейшей эксплуатации системы. Документ обеспечивает единое понимание того, как должна функционировать система, и дать четкие указания для ее разработки, тестирования и внедрения.

1.2 Область действия

Система предназначена для автоматизации процесса выбора и резервирования столиков, а также оформления предварительных заказов еды. Пользователи могут выбрать столик в зависимости от своих предпочтений. Система также позволяет делать предварительный заказ блюд, с возможностью учета диетических предпочтений и аллергий. Она обеспечивает автоматическое подтверждение бронирования, напоминания и уведомления о статусе заказа. Вся информация защищена, включая персональные и платежные данные пользователей. Система гарантирует высокую производительность, быструю реакцию на запросы и надежность работы.

1.3 Целевая аудитория

Целевая аудитория системы делится на несколько ключевых групп, каждая из которых имеет свои уникальные потребности и цели при взаимодействии с системой:

1. Конечные пользователи (клиенты ресторана)

Это основная группа пользователей, которая будет активно взаимодействовать с системой через веб-интерфейс или мобильное приложение. Их основные цели и потребности:

- Бронирование столика: Клиенты могут выбирать дату, время и количество гостей для бронирования столика, а также предпочтительное расположение столика (окно, терраса, VIP-зона и т.д.).
- Предварительный заказ блюд: Пользователи могут заказать блюда заранее, с учетом диетических предпочтений (например, вегетарианские или безглютеновые блюда), а также указания аллергенов.
- Уведомления: Получение уведомлений о статусе бронирования и напоминаний за 1-2 часа до посещения.
- Удобство и безопасность: Легкость в использовании интерфейса, возможность изменять или отменять бронирование, а также уверенность в защите персональных и платежных данных.

2. Менеджеры ресторана

Менеджеры и администраторы ресторанов используют систему для управления бронированиями и заказами. Их основные задачи:

- Управление бронированиями: Контроль за статусом всех бронирований, обновление доступных столов в реальном времени.

- Предварительные заказы: Управление заказами, полученными заранее, с учетом предоплаты и ограничений.
- Отчеты и аналитика: Доступ к статистике по бронированиям, популярности столиков и блюд, а также возможность регулирования бронирований в случае изменений или отмен.

3. Администраторы системы
Технические специалисты и администраторы системы отвечают за настройку, обслуживание и безопасность системы. Их основные задачи:
- Техническая поддержка: Обеспечение бесперебойной работы системы, включая тестирование, обновления и решение технических проблем.
- Безопасность данных: Мониторинг и внедрение мер безопасности для защиты данных пользователей и предотвращения утечек информации.
- Интеграция с внешними сервисами: Подключение системы к платёжным шлюзам и другим сервисам для обработки заказов и платежей.

4. Разработчики и тестировщики
Эта группа включает людей, непосредственно работающих над созданием и тестированием системы. Основные задачи:
- Разработка функционала: Создание новых функций и улучшений системы, таких как фильтры для выбора столиков и блюд, реализация предоплаты и автоматических уведомлений.
- Тестирование системы: Проверка работоспособности системы в различных условиях, включая нагрузки, корректность обработки данных и безопасность

5. Платежные провайдеры и партнеры
Эта категория включает компании, с которыми интегрируется система для обработки платежей. Задачи включают:
- Обработка платежей: Платежные провайдеры отвечают за обработку предоплаты и окончательных платежей за бронирование и еду.
- Безопасность транзакций: Обеспечение безопасности финансовых операций через сертифицированные каналы.

6. Маркетологи и бизнес-аналитики
Эти пользователи используют систему для анализа поведения клиентов, управления рекламными акциями и улучшения пользовательского опыта:
- Сбор и анализ данных о бронированиях, предпочтениях клиентов, популярных блюдах и времени посещений.
- Маркетинговые кампании: Настройка и отслеживание эффективности маркетинговых мероприятий для привлечения новых клиентов, таких как акции и скидки на бронирование.

1.3.1 Важные аспекты для целевой аудитории:
- Пользовательский опыт: Интерфейс системы должен быть интуитивно понятным и доступным для разных типов пользователей, от простых клиентов до профессионалов, работающих с системой.
- Безопасность и конфиденциальность: Все группы пользователей требуют надежной защиты данных, особенно личных и платежных.

- Производительность: Система должна обеспечивать быструю реакцию на запросы и стабильную работу при высоких нагрузках.

1.4 Определения, сокращения

- ***SRS (Software Requirements Specification)*** - спецификация требований к программному обеспечению.
- ***Пользователь*** - клиент ресторана, который использует систему для бронирования столиков и оформления предварительного заказа еды.
- ***Администратор*** - сотрудник ресторана, управляющий бронированиями и заказами.
- ***API (Application Programming Interface)*** - программный интерфейс приложения, предоставляющий доступ к функциональности системы.
- ***TLS (Transport Level Security)*** - протокол защиты передачи данных в сети.
- ***XSS (Cross-Site Scripting)*** - уязвимость веб-приложений, позволяющая злоумышленникам внедрять и выполнять вредоносный код в браузере пользователя.
- ***DDoS (Distributed Denial of Service)*** - распределенная атака отказа в обслуживании.
- ***CSRF (Cross-Site Request Forgery)*** - атака, при которой злоумышленник заставляет пользователя выполнить нежелательный запрос на доверенном сайте.
- ***SQL-инъекции*** - атака, при которой злоумышленник вводит вредоносные SQL-запросы с целью изменения или кражи данных в базе.
- ***DOM (Document Object Model)*** - программный интерфейс для работы со структурой HTML и XML-документов.
- ***OAuth2*** - протокол авторизации, позволяющий предоставлять доступ к ресурсам без передачи пароля.
- ***OpenID Connect*** - надстройка над OAuth2, обеспечивающая аутентификацию пользователей.
- ***CDN (Content Delivery Network)*** - сеть серверов, предназначенная для ускорения загрузки контента путем кэширования данных в разных географических точках.

1.5 Используемые стандарты и документы

- ***ISO/IEC 27001:2013*** - стандарт по управлению информационной безопасностью.
- ***OWASP Top 10*** - руководство по безопасности веб-приложений, описывающее наиболее критичные уязвимости.
- ***PCI DSS (Payment Card Industry Data Security Standard)*** - стандарт безопасности данных платежных карт.
- ***IEEE 830 -*** международный стандарт спецификации требований

1.6 Обзор документа

Документ состоит из следующих разделов:

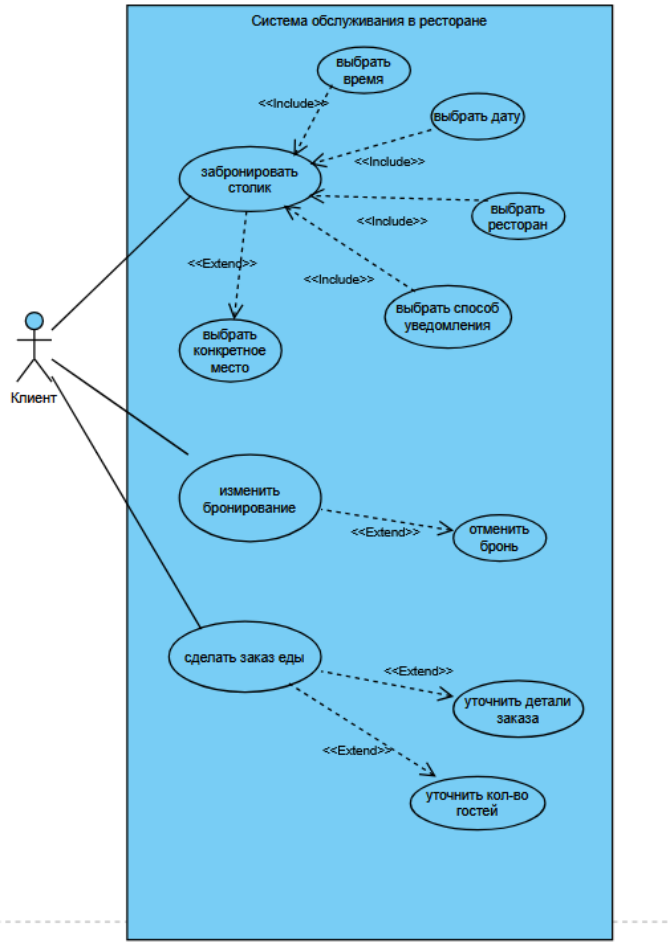
1. Введение
2. Общее описание
3. Интерфейсы
4. Системные требования
5. Дополнительные разделы
6. Аналитическая часть

2. Общее описание

2.1 Описание продукта

Система позволяет пользователям:

- Забронировать столики в ресторане на выбранную дату и время.
- Заказывать блюда заранее с возможностью выбора по фильтрам (например, вегетарианские блюда, аллергии).
- Получать уведомления о статусе бронирования и изменениях.
- Отменять бронирование с соблюдением условий предоплаты.



Юзкейс 1: Бронирование столика

Описание: Пользователь хочет забронировать столик на определенное время в конкретном ресторане.

- Акторы: Пользователь, Система обслуживания ресторана.
- Предусловие: Пользователь зашел в систему и выбрал ресторан.
- Основной сценарий:
    - Пользователь выбирает дату и время бронирования.
    - Система отображает доступные столики.
    - Пользователь выбирает столик и предпочтительный способ уведомлений.
    - Пользователь подтверждает бронирование.
    - Система отправляет подтверждение.
- Альтернативные сценарии:
    - Если выбранный столик занят, система предлагает альтернативные варианты.

    - Если пользователь отменяет бронирование, система отменяет бронь и отправляет уведомление.
    - Если пользователь изменяет бронирование (например, столик или время брони), система корректно изменяет данные и отправляет уведомление об этом.

Юзкейс 2: Формирование предварительного заказа

Описание: Пользователь выбирает блюда из меню и оформляет предварительный заказ.

- Акторы: Пользователь, Система обслуживания ресторана.
- Предусловие: Пользователь выбрал столик и желает заказать еду заранее.
- Основной сценарий:
    - Пользователь просматривает меню ресторана.
    - Пользователь добавляет блюда в корзину.
    - Система позволяет выбрать количество порций и уточнить предпочтения.
    - Пользователь подтверждает заказ.
    - Система подтверждает заказ и отправляет уведомление.
- Альтернативные сценарии:
    - Пользователь может изменить или удалить выбранные блюда до подтверждения заказа.

Юзкейс 3: Изменение бронирования

Описание: Пользователь желает изменить время/дату/место бронирования.

- Акторы: Пользователь, Система обслуживания ресторана.
- Предусловие: Бронирование уже подтверждено.
- Основной сценарий:
    - Пользователь заходит в личный кабинет и выбирает бронирование.
    - Пользователь изменяет то, что ему нужно.
    - Система проверяет доступность нового времени/места и подтверждает изменение.
    - Система отправляет новое подтверждение.
- Альтернативные сценарии:
    - Если выбранное время/место недоступно, система предлагает доступные альтернативы.

Юзкейс 4: Отмена бронирования

Описание: Пользователь хочет отменить бронирование.

- Акторы: Пользователь, Система обслуживания ресторана.
- Предусловие: Бронирование было подтверждено.
- Основной сценарий:
    - Пользователь заходит в личный кабинет и выбирает бронирование для отмены.
    - Пользователь подтверждает отмену бронирования.
    - Система отменяет бронирование и отправляет уведомление.
- Альтернативные сценарии:

- Система предупреждает пользователя о штрафах за позднюю отмену (если пользователь пытается произвести отмену меньше, чем за 3 часа).

2.2 Функциональные требования

1) В процессе создания заказа столик «замораживается» для других клиентов, с целью избежать двойного бронирования. Если клиент не отправляет данные на текущем шагу в течение 5 минут, то заморозка должна быть снята, а текущий процесс заказа аннулируется.
2) При оформлении заказа система должна предоставлять клиенту фильтры для удобного выбора столов и блюд.
3) В зависимости от указанного кол-ва гостей система автоматически предлагает подходящие варианты столиков.
4) Реализация гибких условий отмены бронирования без штрафов: отмена предоплаты
5) Интеграция с государственными сервисами аутентификации.
6) Разработка механизма удаления персональных данных пользователя по запросу.
7) Система фиксирует всю стадию изменения состояний отдельной брони/заказа и уведомляет пользователя об этом.
8) Автоматическое подтверждение бронирования через **выбранный пользователем** способ уведомления.
9) Автоматическая рассылка напоминаний и бронировании за 1-2 часа.
10) Визуальное отображение доступных столов на плане ресторана.
11) Возможность выбора по фильтрам (например, окна, терраса).

2.3 Нефункциональные требования

1) Корректная блокировка возможности бронирования уже занятых мест.
2) Ответ сервера на запрос бронирования не должен превышать 2–3 секунды. (Система должна быть протестирована в штатном и критическом случае, где ответ от сервера будет стабилен и не превышал указанный диапазон времени, которое считается от отправки запроса, до получения ответа, включая в себя обработку и сохранение данных, а также формирование ответа).
3) Все персональные данные пользователей должны быть надежно защищены от несанкционированного доступа и утечек, с использованием шифрования данных (передаваемые данные должны быть защищены TLS, не ниже версии 1.2), все пароли должны быть зашифрованы стойкими алгоритмами (например Argon2d, который используется на backend серверах и позволяет выполнять операции шифрования за достаточно быстрое время).
4) Платежные данные должны шифроваться и передаваться через защищенные каналы, а также их хранение на серверах системы запрещено. Все обработки должны осуществляться через сертифицированные платежные провайдеры.
5) Система должна поддерживать двухфакторную аутентификацию.
6) Система должна быть защищена от всех возможных видов атак (ddos, xss, csrf, sql-инъекции т.п.)
7) Система должна обеспечивать высокую доступность (не менее 99% времени).
8) Должна быть предусмотрена защита от сбоев (например, резервные копии данных).

9) Необходимо контролировать количество одновременных запросов на бронирование столиков, в штатном режиме система обязана поддерживать 200–400 запросов на бронирование одновременно, а в критичных ситуациях необходима возможность горизонтального масштабирования с использованием балансировки нагрузки.
10) Поддержка Chrome начиная с версии 127, Firefox начиная с версии 130, Safari начиная с версии 15.0, Yandex версии 24.5.
11) Поддержка iOS начиная с версии 15, Android начиная с версии 10.
12) Для слабых устройств должна быть поддержка lite версии сайта с минимальным количеством анимаций и более низким расширением изображений.
13) Обеспечение юридической прозрачности обработки персональных данных.
14) Соответствие обновленным требованиям законодательства.

2.4 Пользовательские характеристики

Пользовательские характеристики системы зависят от типа пользователя и его опыта в работе с технологией. Рассмотрим основные категории пользователей и их характеристики:

1. Конечные пользователи (клиенты ресторана)

Опыт и навыки:

- Технические навыки: Средний уровень. Большинство пользователей знакомы с мобильными приложениями и веб-сайтами, однако не все обладают углубленными техническими знаниями.
- Ожидания от интерфейса: Интуитивно понятный интерфейс с минимальным количеством шагов для выполнения бронирования и заказа. Простота и удобство важны для быстрой навигации.
- Устройства: Доступ к системе через мобильные устройства (смартфоны, планшеты) и настольные компьютеры. Интерфейс должен быть адаптирован под различные типы устройств и обеспечивать хороший пользовательский опыт на экранах разных размеров.

Проблемы и потребности:

- Низкая техническая грамотность: Некоторым пользователям может быть трудно ориентироваться в сложных интерфейсах, поэтому система должна быть максимально понятной и простотой в использовании.
- Гибкость выбора: Возможность выбирать столик, заказ еды и предпочтения (например, по аллергиям или вегетарианским блюдам), что требует удобных фильтров и вариантов.
- Безопасность данных: Пользователи ожидают надежности защиты своих персональных и платежных данных.

2. Менеджеры ресторана

Опыт и навыки:

- Технические навыки: Средний и высокий уровень. Менеджеры, как правило, работают с системой на профессиональном уровне, поэтому ожидается, что они будут знакомы с базовыми IT-решениями и интерфейсами управления.
- Ожидания от интерфейса: Требуется наличие подробной и удобной панели управления для отслеживания всех бронирований и заказов, возможности редактирования статусов брони, а также анализа данных о клиентах и заказах.
- Устройства: Система должна быть доступна как на настольных, так и на мобильных устройствах для более гибкой работы.

Проблемы и потребности:

- Эффективное управление: Менеджеры нуждаются в быстром доступе к информации о текущих бронированиях, статусах заказов, а также в легкости настройки уведомлений для клиентов.
- Множество операций: Работа с большим количеством одновременных бронирований, необходимость учета динамических изменений (например, если клиент отменяет или изменяет заказ).

3. Администраторы системы

Опыт и навыки:

- Технические навыки: Высокий уровень. Администраторы должны хорошо разбираться в технических аспектах работы системы, таких как настройка безопасности, баз данных и интеграция с внешними сервисами.
- Ожидания от интерфейса: Нуждается в панели управления системой, которая позволит следить за стабильностью работы, конфигурацией, безопасностью и обновлениями.

Проблемы и потребности:

- Обслуживание системы: Быстрое реагирование на сбои или вопросы пользователей, включая технические проблемы с базой данных или сервером.
- Обновления и безопасность: Потребность в защите системы от атак (например, SQL-инъекций) и в обеспечении бесперебойной работы.

4. Разработчики и тестировщики

Опыт и навыки:

- Технические навыки: Очень высокий уровень. Разработчики должны владеть необходимыми технологиями (например, JavaScript, React, Node.js) для разработки интерфейса и серверной части, а тестировщики — обладать навыками тестирования системы на разных уровнях.
- Ожидания от интерфейса: Должны иметь возможность работать с инструментами для отладки и мониторинга системы, а также поддерживать процесс тестирования с минимальными задержками.

Проблемы и потребности:

- Оптимизация и производительность: Разработчики заинтересованы в быстродействии системы, особенно при высокой нагрузке, а также в удобстве для дальнейшей разработки новых функциональностей.
- Тестирование безопасности: Тестировщики должны убедиться в том, что система защищена от всех возможных уязвимостей и соответствует стандартам безопасности.

5. Платежные провайдеры и партнеры

Опыт и навыки:

- Технические навыки: Высокий уровень. Платежные провайдеры должны иметь глубокие знания в области безопасных онлайн-транзакций и интеграции с платежными шлюзами.
- Ожидания от интерфейса: Ожидается интеграция с внешними системами для обработки платежей, поэтому интерфейсы для передачи и получения данных должны быть четко документированы и защищены.

Проблемы и потребности:

- Безопасность транзакций: Важность обеспечения безопасности всех операций с платежами, включая защиту от фрода и утечек данных.
- Интеграция с системой: Необходимость корректной и безопасной интеграции с внутренними процессами системы для правильной обработки предоплаты и других платежей.

6. Маркетологи и бизнес-аналитики

Опыт и навыки:

- Технические навыки: Средний и высокий уровень. Аналитики, как правило, работают с системой на профессиональном уровне, поэтому ожидается, что они будут знакомы с базовыми IT-решениями и интерфейсами управления.
- Ожидания от интерфейса: Желательно наличие подробной и удобной отдельной панели для сбора статистики и анализа данных по процессам.

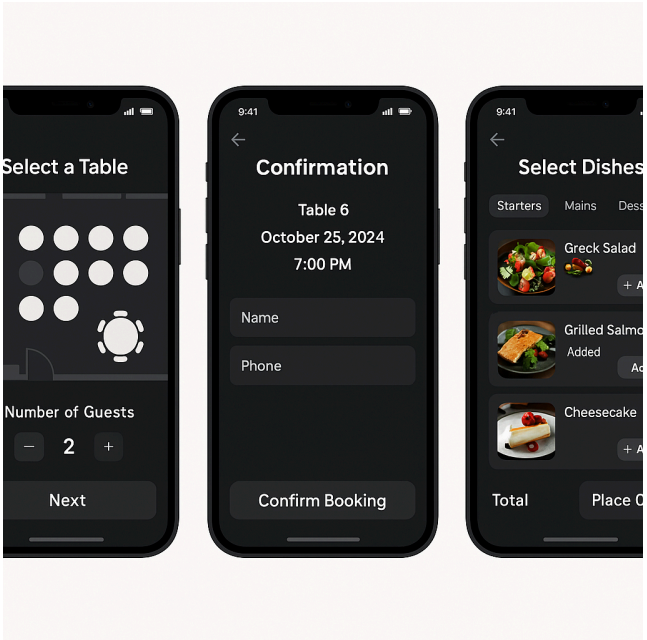
3. Интерфейсы

3.1 Пользовательский интерфейс

Интерфейс должен быть:

- Интуитивно понятным: Простота и логичность взаимодействия.
- Адаптированным под мобильные устройства: Корректное отображение на смартфонах и планшетах.
- Примеры экранов (страниц): Экран выбора стола, экран подтверждения бронирования, экран выбора блюд.

Пример:



3.2 Программные интерфейсы

- API для бронирования: Система предоставляет API для интеграции с другими сервисами, например, с системами управления ресторанами.

3.3 Аппаратные интерфейсы

Не применимо, поскольку система предназначена для работы в веб-среде.

4. Системные требования

4.1 Требования к производительности

- Время отклика на запрос бронирования не должно превышать 2-3 секунд.
- Система должна поддерживать одновременно 200-400 запросов в обычном режиме.

4.2 Требования к безопасности

- Все передаваемые данные должны быть защищены с использованием TLS 1.2 или выше.
- Все пароли и персональные данные должны быть зашифрованы.

4.3 Требования к надежности

- Система должна быть доступна не менее 99% времени.
- Необходимо поддержание резервных копий данных для восстановления после сбоев.

5. Дополнительные разделы

5.1 Ограничения

- Бюджет и сроки: Бюджет проекта ограничен, сроки внедрения – 6 месяцев.
- Технологии: Для разработки используется стек технологий
    1. Языки программирования и фреймворки:
        a. **React -** используется для построения пользовательского интерфейса. Он обеспечивает удобную работу с компонентами, виртуальный DOM и декларативный подход к разработке интерфейсов.
        b. **Node.js -** серверная часть построена на Node.js, что позволяет реализовать асинхронную обработку запросов, масштабируемость и эффективную работу с API.
    2. Базы данных:
        a. **MongoDB -** NoSQL база данных, хранящая данные в формате JSON-подобных документов. Она обеспечивает гибкость схемы и возможность горизонтального масштабирования.
        b. **Redis -** используется как высокопроизводительное хранилище данных в оперативной памяти, выполняя роль кеша и брокера сообщений.
    3. Контейнеризация и развертывание:
        a. **Docker -** применяется для упаковки и изоляции компонентов системы в контейнерах, что упрощает развертывание и управление зависимостями.
        b. **Kubernetes -** отвечает за оркестрацию контейнеров, автоматическое масштабирование и балансировку нагрузки.
    4. Общение между микросервисами:
        a. **Kafka -** обеспечивает асинхронное взаимодействие между микросервисами, гарантируя надежную доставку сообщений и обработку событий в режиме реального времени.

5. Безопасность и защита от атак:
    a. **Keycloak -** используется для управления аутентификацией и авторизацией пользователей, предоставляя поддержку OAuth2 и OpenID Connect.
    b. **Cloudflare -** защищает систему от DDoS-атак, ускоряет загрузку контента через CDN и обеспечивает веб-фильтрацию.
    c. **Argon2 -** применяется для безопасного хеширования паролей и защиты от атак перебора.

5.2 Допущения и зависимости

- Система будет интегрирована с платежными сервисами для обработки предоплаты и платежей.
- Приложение должно поддерживать все версии мобильных операционных систем iOS и Android.

6. Аналитическая часть

6.1 Трудности в разработке

Наибольшие трудности возникли при определении подхода к синхронизации данных и обеспечению безопасности платежных данных. Важно, чтобы система была защищена от атак, таких как SQL-инъекции и DDoS.

6.2 Рекомендации по улучшению документа

- Расширить описание взаимодействия системы с внешними сервисами.
- Подробно описать процессы резервного копирования и восстановления данных.