FOODMART

система электронной коммерции

для продуктового ритейла

Дипломный проект

в рамках курса «Системный аналитик»

Автор: Васильев А.Г.

Контакты: [email@email.ru/+799977711555]

июнь, 2025

Оглавление

Глоссарий

	Понятие	Сокращение	Определение
Основные системы			
	ERP-система	ERP	Система для автоматизации бизнес-процессов (учет товаров, ценообразование, закупки, бухгалтерия).
	WMS система	WMS	Warehouse Management System – система управления складом (приемка, комплектация, отгрузка).
	CRM система	CRM	Customer Relationship Management – система управления клиентскими отношениями.
	e-commerce платформа	–	Онлайн-платформа для продажи товаров (каталог, корзина, оплата, доставка).
Технические термины			
	API Gateway	–	Единая точка входа для API-запросов, маршрутизация к микросервисам.
	Микросервис	–	Независимый сервис, выполняющий одну бизнес-функцию (например, CartService).
	REST API	–	Архитектурный стиль для создания веб-сервисов (HTTP/JSON).
	gRPC	–	Высокопроизводительный RPC-фреймворк для взаимодействия сервисов.
	JWT	JWT	JWT (JSON Web Token) - стандартный токен для аутентификации и авторизации, используемый во всех API-запросах, включая случаи аутентификации через внешние системы (VK ID и др.)
	HTTPS	–	Защищенная версия HTTP (шифрование TLS/SSL).
	Kafka	–	Платформа для обработки событий и потоковой передачи данных.
	Redis	–	In-memory база данных для кэширования (например, корзин пользователей).
	PostgreSQL	–	Реляционная СУБД для хранения структурированных данных.
Бизнес-сущности			
	Корзина (Cart)	–	Временное хранилище товаров, выбранных пользователем для покупки.
	Элемент корзины (CartItem)	–	Конкретный товар в корзине с количеством, весом и ценой.
	Каталог товаров	–	Система представления товаров с фильтрами, сортировкой и поиском.

	Пользователь (User)	–	Клиент, взаимодействующий с системой (регистрация, заказы).
	Гость (Guest)	–	Неавторизованный пользователь, который может формировать корзину.
	Администратор (Admin)	–	Роль с правами управления товарами, заказами и пользователями.
	SKU	SKU	Stock Keeping Unit – уникальный идентификатор товара (артикул).
Процессы			
	Оформление заказа	–	Процесс перевода корзины в статус заказа (оплата, доставка).
	Резервирование товаров	–	Блокировка товара на складе после добавления в корзину.
	Синхронизация с ERP/WMS	–	Обмен данными о товарах (остатки, цены) между системами.
	Интеграция с платежной системой	–	Подключение к внешним сервисам (например, банковским эквайрингам).
Метрики и SLA			
	RPS	RPS	Requests Per Second – количество запросов в секунду.
	Uptime	–	Время доступности системы (например, 99.5%).
	MTTR	MTTR	Mean Time To Repair – среднее время восстановления после сбоя.
	SLA	SLA	Service Level Agreement – соглашение об уровне сервиса.
Безопасность			
	TLS	TLS	Transport Layer Security – протокол шифрования данных.
	PCI DSS	PCI DSS	Стандарт безопасности для обработки платежных данных.
	Аудит действий	–	Логирование операций для отслеживания изменений.
Технологии разработки			
	Agile	–	Гибкая методология разработки (Scrum, Kanban).
	MVP	MVP	Minimum Viable Product – минимально жизнеспособный продукт.
	CI/CD	CI/CD	Continuous Integration / Continuous Delivery – автоматизация сборки и развертывания.
Окружения			
	DEV	DEV	Среда разработки.
	TEST	TEST	Среда для тестирования.
	STAGING	STAGING	Предпродакшен-среда (копия PROD).
	PROD	PROD	Боевая среда.

Дополнительно			
	Личный кабинет	–	Раздел для пользователя с историей заказов и настройками.
	Программа лояльности	–	Система скидок и бонусов для постоянных клиентов.
	Воронка продаж	–	Этапы пути клиента от выбора товара до оплаты.
	Фрикции	–	Препятствия, из-за которых пользователи отказываются от покупки.

Описание задачи

Общие сведения

E-commerce платформа FoodMart – это комплексное решение для региональной торговой сети продуктов питания, объединяющее современные технологии и удобный пользовательский опыт. В условиях растущего спроса на онлайн-покупки продуктов компания Заказчика стремится создать надежный, безопасный и интуитивно понятный сервис, который не только удовлетворит потребности клиентов, но и оптимизирует внутренние бизнес-процессы.

Данный документ описывает ключевые аспекты разработки системы: от архитектуры и функциональности до интеграции с существующей инфраструктурой. Решение будет включать гибкие механизмы продаж, автоматизацию логистики и аналитику для повышения эффективности работы компании.

Проект направлен на создание платформы, способной масштабироваться вместе с бизнесом, обеспечивая стабильную работу даже в периоды высокой нагрузки, например, во время сезонного спроса или маркетинговых акций.

Решение должно позволять клиентам осуществлять ряд стандартных действий для совершения покупки:

- просмотреть каталог товаров;
- детали выбранных товаров;
- добавить товар в корзину;
- выбрать вариант доставки и рассчитать стоимость заказа;
- оплатить заказ.

Дополнительный функционал может включать в себя личный кабинет клиента, отслеживание истории покупок, отображение данных программы лояльности и др.

Клиенты должны иметь доступ к каталогу и покупке с помощью web-интерфейса, а также мобильного приложения. Сценарии использования должны совпадать на уровне бизнес-логики, однако необходимо учесть особенности реализации мобильного и web интерфейса.

Текущая ситуация

В компании Заказчика уже внедрен ряд решений, поддерживающих процессы продажи в розничных магазинах. Заказчик ведет работу в ERP системе стороннего разработчика ПО, в которой уже реализованы следующие бизнес-процессы:

- Управление номенклатурой
- Ценообразование
- Управление закупками
- Бухгалтерский учет

ERP система выступает как мастер-система для ведения данных по товарам, ценам и различных справочников. Данные по товарам могут быть обогащены дополнительными атрибутами перед публикацией на уровне проектируемого e-commerce решения.

Также Заказчик ведет свою работу в WMS системе (также стороннего разработчика), которая автоматизирует следующие процессы:

- Приемка товаров
- Комплектация товаров
- Резервирование товаров
- Отгрузка товаров

На данный момент в компании CRM система не внедрена, а для клиентов нет программы лояльности.

Необходимо интегрировать e-commerce решение FoodMart в существующий IT-ландшафт заказчика.

Цели

В рамках создания высоконагруженной e-commerce платформы для продажи продуктов питания FoodMart необходимо разработать сервис по управлению Корзиной товаров. Данный сервис должен быть высоконагруженным, отказоустойчивым микросервисом. Должен позволять осуществлять формирование корзины как авторизованным, так и неавторизованным пользователям, интегрироваться с другими системами FoodMart и внешними системами Заказчика, в случаях, когда это необходимо. Данные, которыми оперирует сервис, должны быть актуальны (цены, остатки на складе и т.д.). Быть удобным в использовании.

Конечные цели проекта, это, во-первых, увеличить число завершённых покупок (конверсию). Во-вторых, исключить препятствия (фрикции), из-за которых покупатели отказываются от заказа. И в-третьих, оптимизировать ключевой этап воронки продаж – переход от выбора товара к оплате.

Это особенно важно в e-commerce, где высокий процент брошенных корзин (иногда до 70%). Хороший сервис корзины помогает вернуть часть этих потерянных клиентов.

Верхнеуровневое описание решения

Для создания **концептуальной модели** интернет-платформы по продаже продуктов питания FoodMart можно выделить следующие основные сущности:

1. Пользователь, клиент (User): Человек, который регистрируется в системе FoodMart, просматривает и выбирает товары и совершает покупки.
2. Товар (Product): Продукты питания, которые предлагаются для продажи.
3. Корзина (Cart): Содержит товары, которые покупатель добавил для покупки.
4. Заказ (Order): Содержит информацию о заказе, такой как товары, сумма и статус.
5. Оплата (Payment): Информация о процессе оплаты.
6. Доставка (Delivery): Данные о доставке, включая адрес и способ доставки.
7. Администратор (Admin): Управляет продуктами, заказами и пользователями.
8. Каталог товаров (Product catalog): Система, в которой представлена информация о товарах для удобства просмотра и выбора.
9. Склад (WMS): Место хранения товаров.
10. Система учета (ERP): Учет товаров и финансовая отчетность.
11. Уведомления (Notifications): Сообщения для пользователей об изменении статуса заказа, а также об акциях и скидках.
12. Отзывы (Review): Рейтинги и отзывы пользователей о продуктах.
13. Акции и скидки (Promo): Система акций и скидок

Все указанные сущности являются кандидатами в классы, поскольку они представляют ключевые элементы, участвующие в процессе работы системы.

Пользователь (User). Пользователь является центральной сущностью, взаимодействующей с системой. Ему необходимы атрибуты (например, имя, электронная почта, пароль) и методы (например, регистрация, авторизация, просмотр заказов).

Товар (Product). Товар является основным объектом торговли. Для него требуются атрибуты (например, название, описание, цена, наличие на складе) и методы (например, добавление в корзину, обновление информации товаре).

Каталог товаров (Product catalog). Каталог предназначен для удобства просмотра и выбора товара. Основной объект здесь карточка товара. Для работы с ней нужны методы добавления/удаления и т.п. Каталог должен обладать структурой. Информация о товарах должна быть сгруппирована с учетом разных признаков. Поэтому должны быть доступны методы группировки, сортировки, фильтрации, пагинации. Учитывая то, что данные о товарах Заказчика содержаться во внешней ERP-системе, каталог должен быть синхронизироваться с ней.

Корзина (Cart). Корзина является временным хранилищем для товаров, которые пользователь собирается приобрести. Она требует атрибутов (например, список продуктов, общая стоимость) и методов (например, добавление/удаление товаров, очистка корзины).

Заказ (Order). Заказ фиксирует данные о покупках пользователя, включая товары, суммы и статус выполнения. Атрибуты могут включать дату, номер заказа и связь с пользователем.

Оплата (Payment). Оплата управляет информацией о платежах и их статусах (например, успешный или неудачный платеж). Может включать атрибуты, такие как способ оплаты и номер транзакции.

Склад (WMS): Предназначен для хранения товаров. Может включать атрибуты название/идентификатор, адрес и методы: приемка, отгрузка товара. Основной объект здесь товар, который может иметь атрибуты (в контексте данного проекта): зарезервирован передан в доставку, и соответствующие методы.

Система учета (ERP): Учет товаров и финансовая отчетность.

Доставка (Delivery). Доставка представляет способ передачи заказов пользователю. Необходимы атрибуты, такие как адрес доставки, способ доставки, статус отправки.

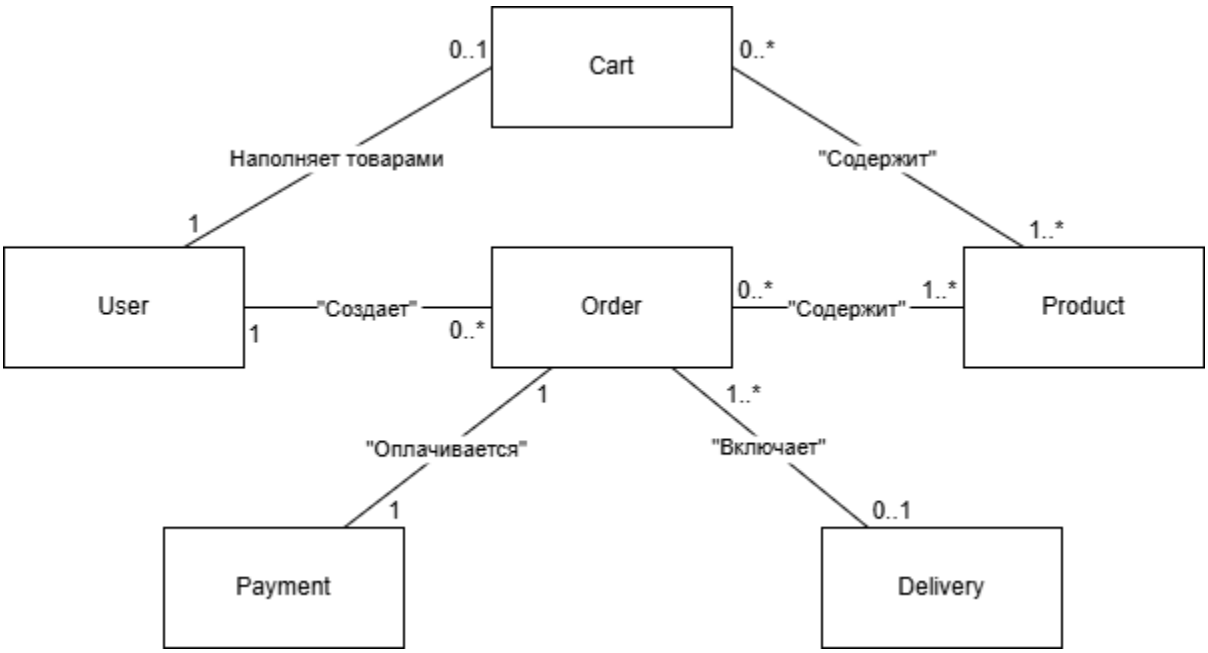
Уведомления (Notifications): Сообщения могут иметь обязательный и необязательный характер. Имеют атрибуты: текст, отправлено, получено, время отправки/получения и соответствующие методы.

Администратор (Admin). Администратор управляет всей системой, включая добавление товаров, управление заказами и пользователями. Это отдельная роль с расширенными правами.

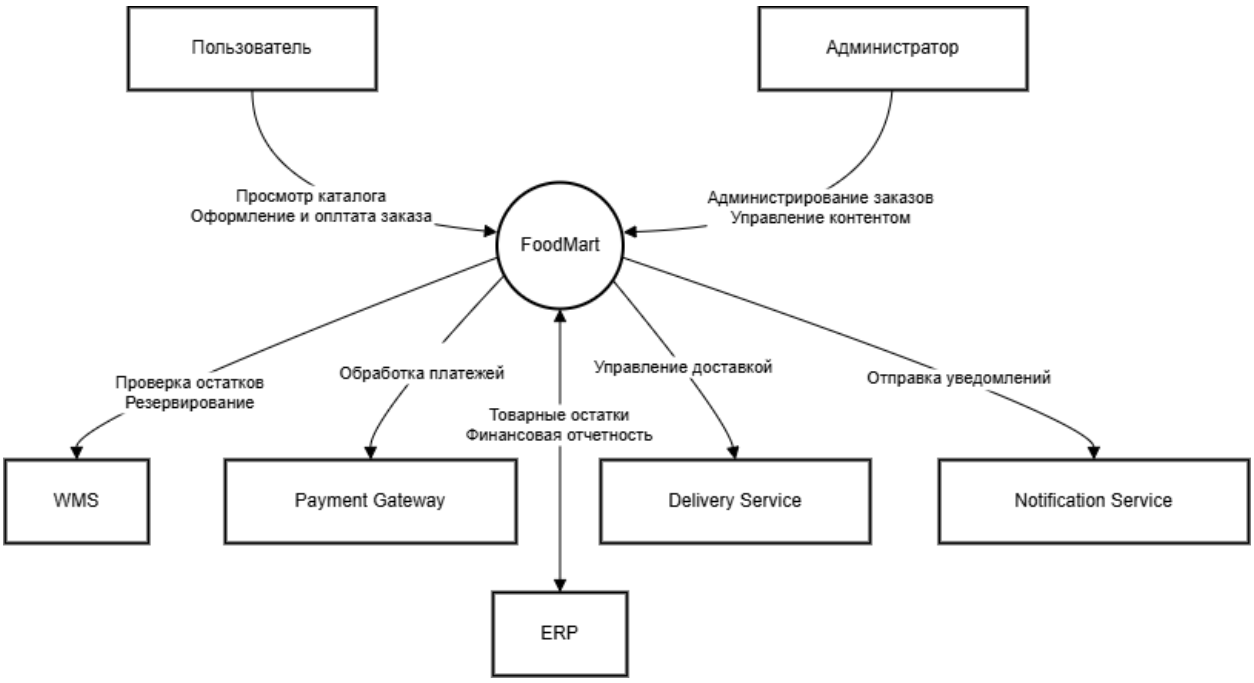
Отзывы (Review). Отзывы содержат пользовательские оценки продуктов. Они требуют атрибутов (например, текст отзыва, рейтинг, дата публикации) и связи с продуктом и пользователем.

Акции и скидки (Promo): Система, в которой представлена информация о скидках на отдельные товары, категории товаров, информация об скидочных промо акциях.

Концептуальная модель данных, поддерживающих процесс покупки товара, оформления и доставки заказа, представлена на диаграмме ниже. Атрибуты данных на схеме не приводятся. Они будут подробно описаны в разделе «Модель данных» настоящей документации.



Системы склада, учета товаров уже есть у Заказчика. Для оплаты, доставки, уведомлений Заказчик планирует использовать системы сторонних разработчиков. В связи с этим необходима интеграция с разрабатываемой системой FoodMart.



Функциональный объем проекта

В рамках планирования работ по проекту определен его основной функционал и сформулированы User Stories (US). Вот основные из них:

1. Выбор продуктов
    - Как пользователь, я хочу увидеть список продуктов, чтобы найти нужные товары.
    - Как пользователь, я хочу фильтровать продукты по категориям (молочные, мясные, овощи, бакалея, и т.д.), чтобы быстрее найти нужное.
    - Как пользователь, я хочу сортировать товары по цене или популярности, чтобы найти лучшие предложения.
    - Как пользователь, я хочу просматривать детальные описания продуктов, чтобы понять, что я покупаю.
    - Как пользователь, я хочу видеть рекомендации для покупки на основе своих предыдущих покупок для того, чтобы выбрать продукт.
    - Как пользователь, я хочу видеть рекомендации для покупки на основе рейтинга покупок других пользователей для того, чтобы выбрать продукт.
    - Как пользователь, я хочу видеть рекомендации для покупки акционных товаров для того, чтобы выбрать продукт.

- Как пользователь, я хочу видеть рекомендации для покупки товаров со скидкой для того, чтобы выбрать продукт.
- Как пользователь, я хочу видеть акции и скидки на товары (при наличии), чтобы выбрать продукт.

2. Управление корзиной
   - Как пользователь, я хочу добавить товар в корзину, чтобы формировать заказ.
   - Как пользователь, я хочу видеть обновленную информацию о количестве товара и цену (общую и за ед. товара) при изменении списка товаров в корзине, чтобы контролировать заказ и его стоимость.
   - Как пользователь, я хочу видеть стоимость отдельных товаров, и общую цену в корзине с учетом акций и скидок на товары, чтобы контролировать стоимость.
   - Как пользователь, я хочу быстро удалить товар из корзины, если изменил решение о покупке.
   - Как пользователь, я хочу начать оформление заказа, если сформировал окончательно список товаров в корзине
3. Оформление заказа
   - Как пользователь, я хочу видеть итоговую сумму заказа, чтобы знать, сколько мне предстоит заплатить.
   - Как пользователь, я хочу видеть сумму отдельных позиций товаров в заказе, чтобы скорректировать общую сумму заказа, если это мне стало необходимо.
   - Как пользователь, я хочу ввести данные доставки (адрес, контактный телефон), чтобы получить заказ по нужному адресу.
   - Как пользователь, я хочу выбрать метод доставки (самовывоз, курьером), чтобы выбрать удобный способ получения.
   - Как пользователь, я хочу выбрать способ оплаты (по предоплате по карте, картой при получении, наличными при получении, биткоинами), чтобы выбрать удобный способ расчета.
4. Подтверждение и оплата
   - Как пользователь, я хочу видеть сводку всех товаров, адрес доставки и итоговую сумму перед оплатой, чтобы убедиться в правильности данных.
   - Как пользователь, я хочу получить подтверждение успешной оплаты, чтобы быть уверенным, что заказ принят.
5. Получение заказа
   - Как пользователь, я хочу получить уведомление о том, что заказ отправлен, чтобы отслеживать процесс.
   - Как пользователь, я хочу отслеживать статус доставки (в пути, доставлено), чтобы быть в курсе, когда ожидать товар.
   - Как пользователь, я хочу иметь возможность оставить отзыв о товаре, чтобы поделиться впечатлениями с другими покупателями.

*Для реализации необходимы **дополнительные** функциональные блоки и сущности:*

- Поиск: Возможность искать продукты по названию, категориям, брендам, составу и т.д.

- Личный кабинет: Регистрация, вход в систему, управление персональными данными, история заказов.
- Рекомендации товаров: Система рекомендаций на основе предыдущих покупок или предпочтений.
- Отзывы и рейтинги: Возможность оставлять отзывы и читать мнения других пользователей о товарах.
- Акции и скидки: Поддержка специальных предложений, скидок, акций и купонов.

Для планирования этапов работы по проекты US сгруппированы по этапам работ и ключевым функциональным блокам.

Табл. User Story Map для роли Пользователь

	Выбор продуктов	**Управление корзиной**	**Оформление заказа**	**Подтверждение и оплата**	**Получение заказа**
MVP	Просмотр списка продуктов	Добавление товара в корзину	Просмотр итоговой суммы заказа	Просмотр сводки товаров, адреса и суммы	Уведомление об отправке заказа
	Фильтрация по категориям	Обновление информации о количестве и цене	Ввод данных доставки	Подтверждение успешной оплаты	Отслеживание статуса доставки
	Сортировка по цене или популярности	Удаление товара из корзины	Выбор метода доставки		
	Просмотр детальных описаний продуктов	Начало оформления заказа	Выбор способа оплаты		
Этап 1	Рекомендации на основе предыдущих покупок	Отображение стоимости с учетом скидок	Корректировка суммы заказа		Возможность оставить отзыв о товаре
	Рекомендации на основе рейтинга				
	Просмотр акций и скидок				
Этап 2	Поиск продуктов по названию, категориям				
	Рекомендации акционных товаров				
Этап 3	Личный кабинет (регистрация, история заказов)				
	Управление персональными данными				

Примечание:

- MVP (Minimum Viable Product). Включает базовые функции, необходимые для работы сервиса.
- Этап 1. Добавляет рекомендации, отображение скидок и возможность оставлять отзывы.
- Этап 2. Включает расширенный поиск и рекомендации акционных товаров.
- Этап 3. Добавляет личный кабинет и управление персональными данными.

Дополнительные US: Функции, которые могут быть реализованы позже для повышения удобства пользователей

Методология разработки ПО

Для создания проекта интернет-магазина e-commerce по продаже продуктов питания для компании с уже внедренными ERP и WMS, наиболее подходящей методологией будет Agile (с использованием фреймворка Scrum или Kanban). Причины выбора следующие.

1. Гибкость и адаптивность

Причина: Проект e-commerce часто включает изменения в требованиях, так как компания может корректировать бизнес-процессы на основе обратной связи клиентов или внутреннего опыта.

Преимущество Agile: Позволяет итеративно развивать продукт и адаптироваться к изменениям благодаря коротким спринтам или циклам.

2. Интеграция с существующими в компании системами (ERP, WMS)

Причина: Важно интегрировать e-commerce платформу с ERP и WMS для синхронизации данных о продуктах, заказах, складе и доставке.

Преимущество Agile: Постоянное тестирование на каждом этапе разработки гарантирует успешную интеграцию с существующими системами.

3. Короткие сроки вывода на рынок

Причина: Интернет-магазины должны быстро запускаться, чтобы компания могла оценить эффективность канала продаж и адаптироваться к рыночным условиям.

Преимущество Agile: Функциональный минимально жизнеспособный продукт (MVP) может быть запущен в короткие сроки, после чего продолжается его доработка.

4. Вовлеченность всех заинтересованных сторон

Причина: Для успешного проекта требуется тесное взаимодействие между IT-командой, отделами логистики, маркетинга, продаж и клиентами.

Преимущество Agile: Регулярные встречи (например, в Scrum — ежедневные стендапы, демо, ретроспективы) поддерживают постоянную связь между участниками.

5. Фокус на клиенте

Причина: E-commerce ориентирован на конечного пользователя, и важно учитывать его потребности и предпочтения.

Преимущество Agile: Позволяет собирать обратную связь от пользователей на ранних этапах и улучшать продукт в процессе разработки.

Дополнение: Возможные адаптации. Если проект требует более строгой структуры из-за интеграции с ERP/WMS, может быть полезно применить гибридный подход Agile + Waterfall: Waterfall для начальных этапов — анализа, планирования архитектуры и интеграции с ERP/WMS. Agile – для разработки клиентского интерфейса, функциональности e-commerce и улучшений на основе обратной связи.

Таким образом, для данного e-commerce проекта с учетом существующих решений предлагается начинать с Agile, но по мере реализации проводить оценку необходимости возможного применения гибридного подхода.

Формирование пула задач и бэклога для разработки ПО

Для учета доработок необходимо добиваться гибкости в управлении проектом и поддерживать бэклог в актуальном состоянии. Для этого можно применить следующие практики:

1. Актуализация бэклога

Пересмотр приоритетов: новые доработки могут вводить изменения в бэклог, включая новые задачи или модификацию существующих. Бэклог должен обновляться регулярно.

Разделение задач: отделять доработки, связанные с текущей версией, от долгосрочных улучшений. Это обеспечит ясность для команды.

2. Влияние на планирование

Гибкость в спринтах: при использовании методология Agile, необходимо оставлять место в спринтах для доработок, которые могут возникнуть в процессе тестирования или ОС.

Оценка влияния: каждую новую доработку нужно оценивать на предмет ее приоритета и влияния на другие задачи.

3. Добавление технического долга

Рефакторинг: при внедрении доработок нужно учитывать возможный технический долг. Для будущих версий МП должен быть запланирован рефакторинг и время на оптимизацию кода.

4. Использование трекинговых систем для управления задачами на проекте и бэклогом, например Jira. В настоящее время внедряются отечественные решения. Цель: упрощение отслеживания задач, обновление их статуса и обеспечение прозрачности между членами команды, а если необходимо, то и др. участниками проекта. Чтобы улучшить управление задачами, в бэклоге можно использовать категории (например, "Баги", "Новые фичи", "Доработки" и т.п.).

5. Обратная связь с пользователями

Интеграция пользовательских запросов: Многие доработки могут быть результатом отзывов пользователей. Можно создать процесс сбора и анализа отзывов для включения наиболее востребованных изменений в бэклог. Пользователей необходимо информировать о доработках через описание к очередному релизу.

6. Мониторинг времени и ресурсов

Ведение доработок влечет за собой дополнительные затраты времени и ресурсов. Важно следить за тем, чтобы изменения не приводили к срыву сроков или превышению бюджета.

Мобильный **бэклог** с основным бэклогом проекта будет **синхронизироваться** по паттерну Proxy Backlog (Прокси-бэклог). Здесь мобильный бэклог — это «прокси» (представитель) основного, содержащий только релевантные задачи. Среди плюсов – четкое разделение с сохранением связи. Среди минусов – требует настройки фильтров в трекинговой системе. Рекомендации по реализации:

- В основном бэклоге задачи помечаются для мобильной разработки, а затем автоматически/вручную копируются в мобильный.
- Использовать фильтры и автоматические правила в трекинговой системе.

В контексте проекта по разработке интернет-магазина для продажи продуктов питания можно выделить следующие ключевые задачи и распределить **ответственность** между аналитиком (BA/SA) и PO.

Зоны ответственности:

- Анализ требований
- Определение бизнес-целей
- Разработка функциональных требований
- Проектирование интерфейса
- Разработка технических спецификаций
- Приоритезация фич и требований
- Коммуникация с командой разработки
- Тестирование продукта
- Релиз и внедрение
- Анализ пользовательского опыта и обратной связи

Тогда RACI матрица может выглядеть:

Задачи / Роли	BA/SA	PO
Анализ требований	R	A
Определение бизнес-целей	C	A
Разработка функциональных требований	R	A
Проектирование интерфейса	C	A
Разработка технических спецификаций	R	C
Приоритезация фич и требований	C	A
Коммуникация с командой разработки	R	A
Тестирование продукта	C	I
Релиз и внедрение	I	A
Анализ пользовательского опыта и ОС	C	A

Примечание: R – ответственный, A – окончательно утверждает, C – консультирует (обладает экспертным мнением), I – информируемый

Ограничения, допущения и риски

- ## Ограничения

Факторы, которые могут ограничить реализацию проекта:

Технические:

1. Интеграции с внешними сервисами (платежные системы, службы доставки, CRM) могут требовать дополнительного времени и согласований.
2. Масштабируемость – если база товаров или трафик вырастут, текущая архитектура может не справиться.
3. Поддержка мобильных устройств – не все функции могут одинаково хорошо работать на всех устройствах.

Бюджетные и временные:

1. Ограниченный бюджет может исключить некоторые функции (например, сложную систему рекомендаций на ИИ).
2. Жесткие сроки запуска MVP могут привести к упрощению некоторых процессов (например, ручная обработка заказов вместо автоматизации).

Бизнес-процессы:

1. Ограниченный ассортимент на старте.
2. География доставки – FoodMart работает только в пределах одного региона (с охватом 500K населения).

- ## Допущения

Утверждения, которые принимаются за истину, но могут измениться:

О пользователях:

1. Пользователи будут регистрироваться, а не оформлять заказы как гости.
2. Основная аудитория – люди 25-45 лет, готовые совершать покупки онлайн.

О бизнес-модели:

1. Доставка будет осуществляться через партнерские службы (Яндекс.Доставка).
2. Оплата картой будет основным способом (электронные карты) другие методы менее востребованы.

О технической реализации:

1. Система рекомендаций будет работать на основе простых правил (например, "похожие товары"), а не машинного обучения.
2. Интеграция с ERP системой (1С или другой) будет возможна без серьезных доработок.

- **Риски**

Потенциальные проблемы, которые могут повлиять на проект:

Технические риски:

1. При сбоях в платежных системах пользователи не смогут оплатить заказ.
2. Низкая производительность при высокой нагрузке (например, в период акций).
3. Потеря данных из-за ошибок в бэкапах.

Бизнес-риски:

1. Низкая конверсия из-за сложного UX или высокой стоимости доставки.
2. Юридические ограничения (например, запрет на продажу определенных товаров).
3. Конкуренция – если аналогичные магазины предлагают лучшие условия.

Организационные риски:

1. Недостаток экспертизы в команде (например, нет специалиста по интеграциям).
2. Задержки поставок товаров от поставщиков.

Бизнес-процессы

Подпроцессы и функциональные требования оказывают влияние на основной бизнес-процесс, определяя его эффективность, гибкость. Для процесса управления корзиной товаров сформированы следующие подпроцессы и функциональные требования:

1. Добавление товаров в корзину. Пользователь может добавлять товары в корзину со страницы каталога или страницы товара.

2. Изменение количества товаров. В корзине пользователь может изменять количество выбранных товаров. Здесь необходимо проверить остаток выбранного товара на складе и сверить с указанным пользователем количеством товара.

3. Удаление товаров из корзины. Пользователь может удалить один или несколько товаров из корзины.

4. Динамическое обновление итоговой стоимости. При добавлении, изменении количества или удалении товаров итоговая стоимость корзины должна пересчитываться автоматически.

5. Просмотр деталей корзины. В корзине должны отображаться: Название товара, изображение, цена за единицу. Общая стоимость для каждого товара (количество × цена).

6. Учет скидок и акций. Корзина должна учитывать скидки, акции и промокоды, примененные к товарам, и отображать обновленную итоговую стоимость.

7. Учет стоимости доставки. Система должна рассчитывать стоимость доставки на основе выбранного способа доставки и отображать ее в итоговой сумме.

8. Сохранение состояния корзины. Корзина должна сохранять добавленные товары даже после закрытия Web браузера или Мобильного приложения.

9. Подсказка о минимальной сумме для бесплатной доставки. Если пользователь близок к минимальной сумме для бесплатной доставки, система должна отображать сообщение с рекомендацией добавить товары.

10. Проверка наличия товаров при оформлении заказа. Перед завершением заказа система должна проверять актуальность наличия товаров на складе. Если чего-то нет в наличии, пользователь получает уведомление.

Альтернативные сценарии:

1. Альтернативный сценарий №1

1.1. Пользователь выходит из раздела каталог.

1.2. Конец сценария.

2. Альтернативный сценарий №2:

2.1. Пользователь нажимает кнопку "Добавить в корзину" на странице каталога

2.2. Система проверяет доступность товаров на Складе. Товар в указанном количестве отсутствует.

2.3. Система выводит сообщение об ошибке.

2.4. Система отображает уведомление об уменьшении количества по данной позиции.

3. Альтернативный сценарий №3:

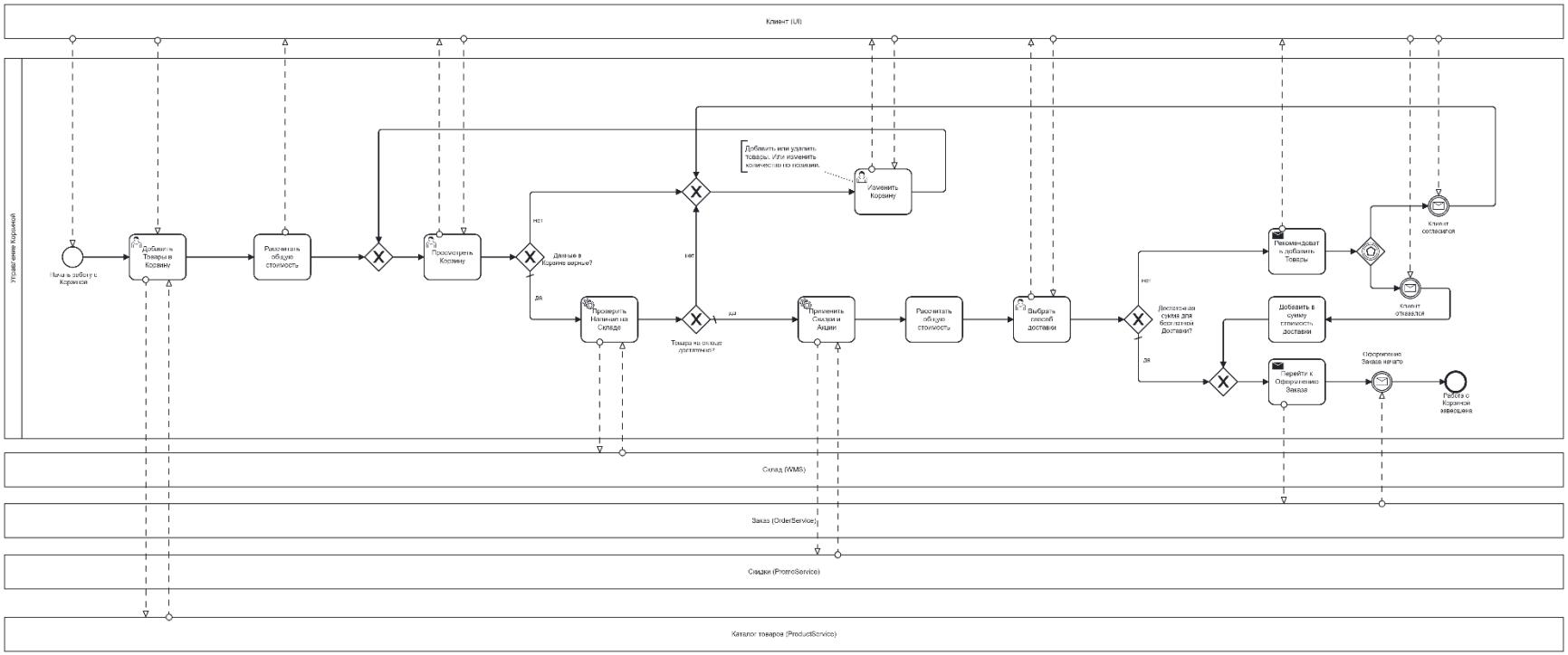
3.1. Пользователь нажимает кнопку "Добавить в корзину" на странице каталога.

3.2. Система не доступна.

3.3. Вывести сообщение об ошибке.

3.4. Вывести уведомление с предложением сделать добавить товар позже.

Результаты анализа и моделирования процесса управления Корзиной товаров пользователем представлены на BPMN-диаграмме. Диаграмма детально описывает процесс, начиная от добавления товаров и заканчивая передачей данных для оформления заказа, с учетом взаимодействия между различными участниками и системами. Это обеспечивает четкое понимание *workflow* и точек принятия решений.



Участники и взаимодействия:

- Клиент (UI): Инициирует процесс, просматривает и изменяет корзину, выбирает способ доставки.
- Склад (WMS): Проверяет наличие товаров.
- Сервис заказов (OrderService): Обрабатывает оформление заказа.
- Сервис скидок (PromoService): Применяет скидки и акции.
- Сервис каталога товаров (ProductService): Предоставляет информацию о товарах.

Особенности:

- Используются шлюзы для принятия решений (например, проверка данных корзины, наличие товаров, условия доставки).
- Включены пользовательские задачи (например, изменение корзины, выбор доставки) и сервисные задачи (например, проверка наличия товаров, применение скидок).
- Процесс включает циклы (например, возврат к изменению корзины при неверных данных или недостаточном количестве товаров).

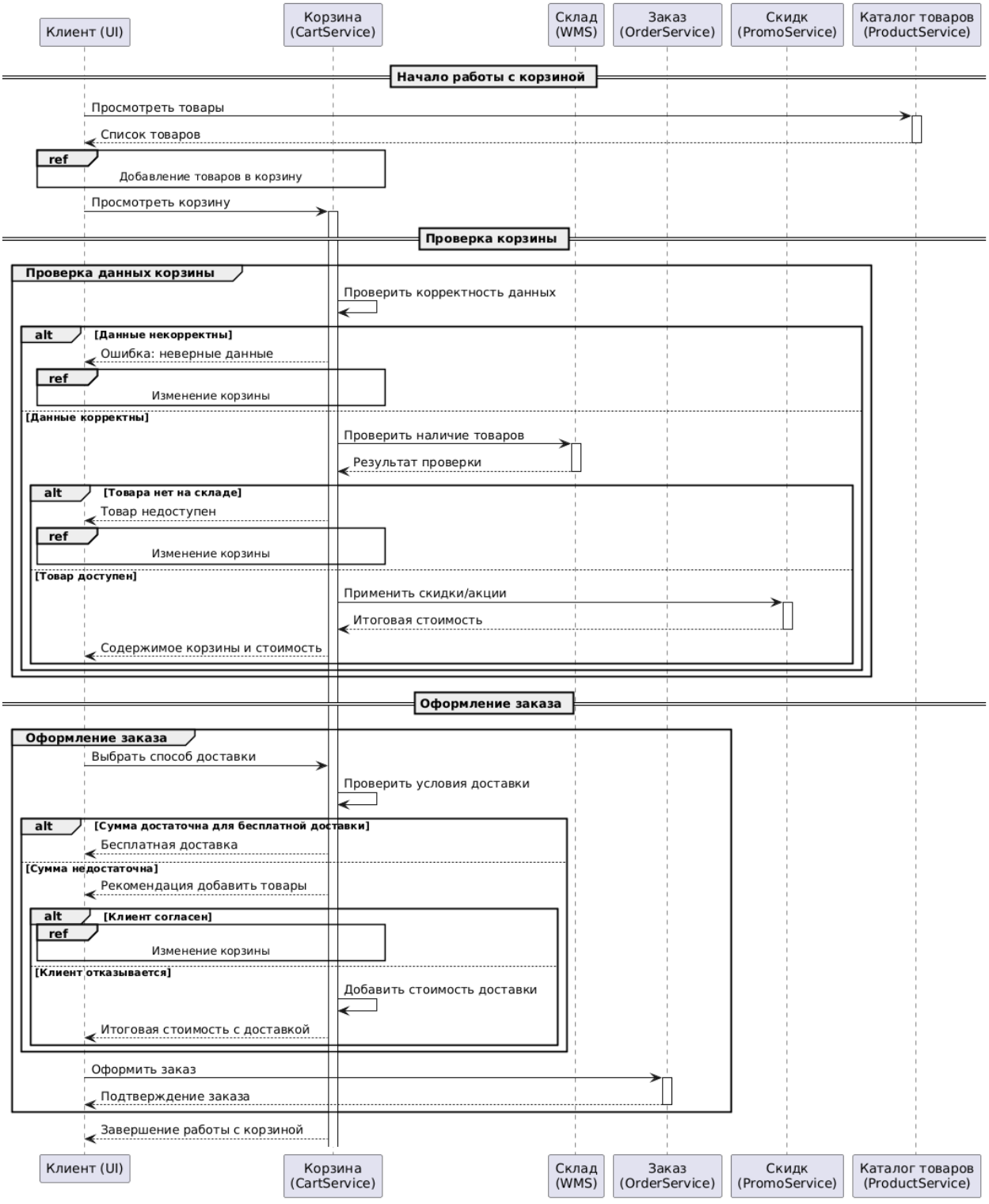


Диаграмма последовательности описывает взаимодействие между участниками (акторами и компонентами системы) в процессе работы с корзиной:

- Добавление товаров.
- Проверка корректности данных.
- Изменение состава корзины.
- Оформление заказа.

Н диаграмме визуализированы:

- Поток сообщений между компонентами.
- Условия и альтернативные сценарии (например, недостаток товара на складе).
- Взаимодействие с внешними сервисами (склад, система скидок).

Основные участники:

Участник	**Роль**
Клиент	Пользователь, взаимодействующий с корзиной через UI.
Корзина (CartService)	Основной сервис, отвечающий за хранение и обработку товаров.
Склад (WMS)	Проверяет наличие товаров и их количество.
Заказ (OrderService)	Обрабатывает оформление заказа после подтверждения.
Скидки (PromoService)	Применяет акции и рассчитывает итоговую стоимость.
Каталог товаров (ProductService)	Предоставляет информацию о товарах при добавлении в корзину.

Ключевые сценарии

- Добавление товаров в корзину
  1. Клиент запрашивает список товаров из Каталога.
  2. Выбирает товар и отправляет запрос в Корзину.
  3. Корзина обновляет состав и возвращает подтверждение.
- Проверка корзины
  1. Клиент запрашивает просмотр корзины.
  2. Корзина проверяет корректность данных (наличие, актуальность цен).
  3. Если данные некорректны — возвращается ошибка.
  4. Если корректны — отправляется запрос на Склад для проверки наличия.
     - Если товара нет: Клиент получает уведомление, предлагается изменить корзину.
     - Если товар доступен: применяются скидки, рассчитывается итоговая стоимость.
- Изменение корзины

1. Клиент запрашивает изменение (удаление/изменение количества).
2. Корзина проверяет новое количество через Склад.
3. При успехе — обновляет данные, иначе уведомляет клиента.
4. Возвращает обновленный состав корзины.

- Оформление заказа
  1. Клиент выбирает способ доставки.
  2. Корзина проверяет условия доставки:
     - Если сумма достаточна для бесплатной доставки — заказ оформляется.
     - Если нет — предлагается добавить товары или оплатить доставку.
  3. После подтверждения данные передаются в сервис Заказа.

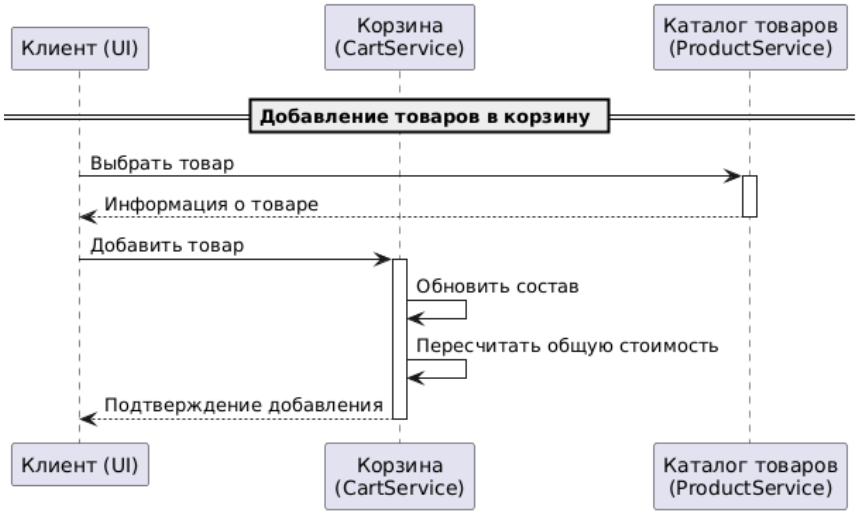
Альтернативные потоки

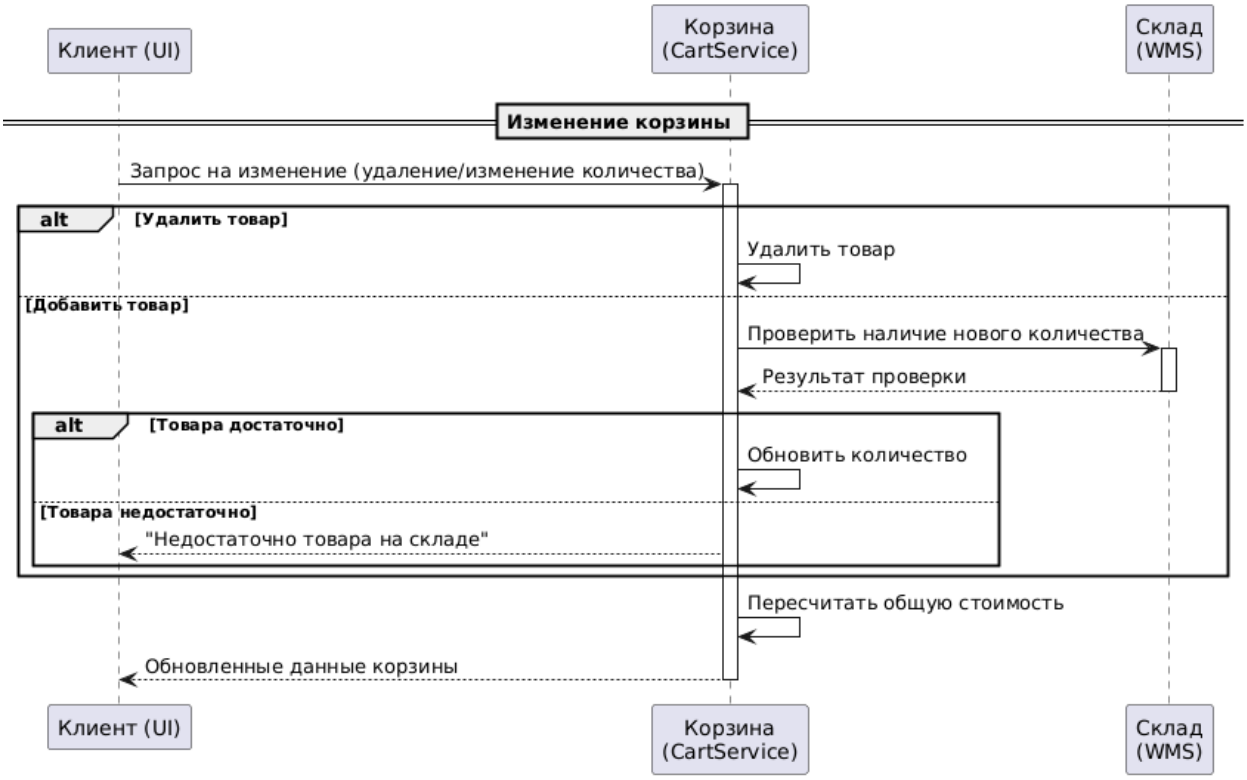
- Товар отсутствует на складе: Корзина уведомляет клиента, предлагает изменить состав.
- Недостаточная сумма для бесплатной доставки: Клиент может добавить товары или согласиться на платную доставку.
- Некорректные данные в корзине: требуется повторная проверка или очистка корзины.

Диаграмма последовательности детализирует шаги из BPMN:

- Проверка наличия на складе – Взаимодействие с Складом.
- Применение скидок – Обращение к PromoService.
- Оформление заказа – Передача данных в сервис Заказа.

Ниже приведены Ref-фреймы, которые декомпозируют сложные сценарии (изменение корзины, добавление товаров).

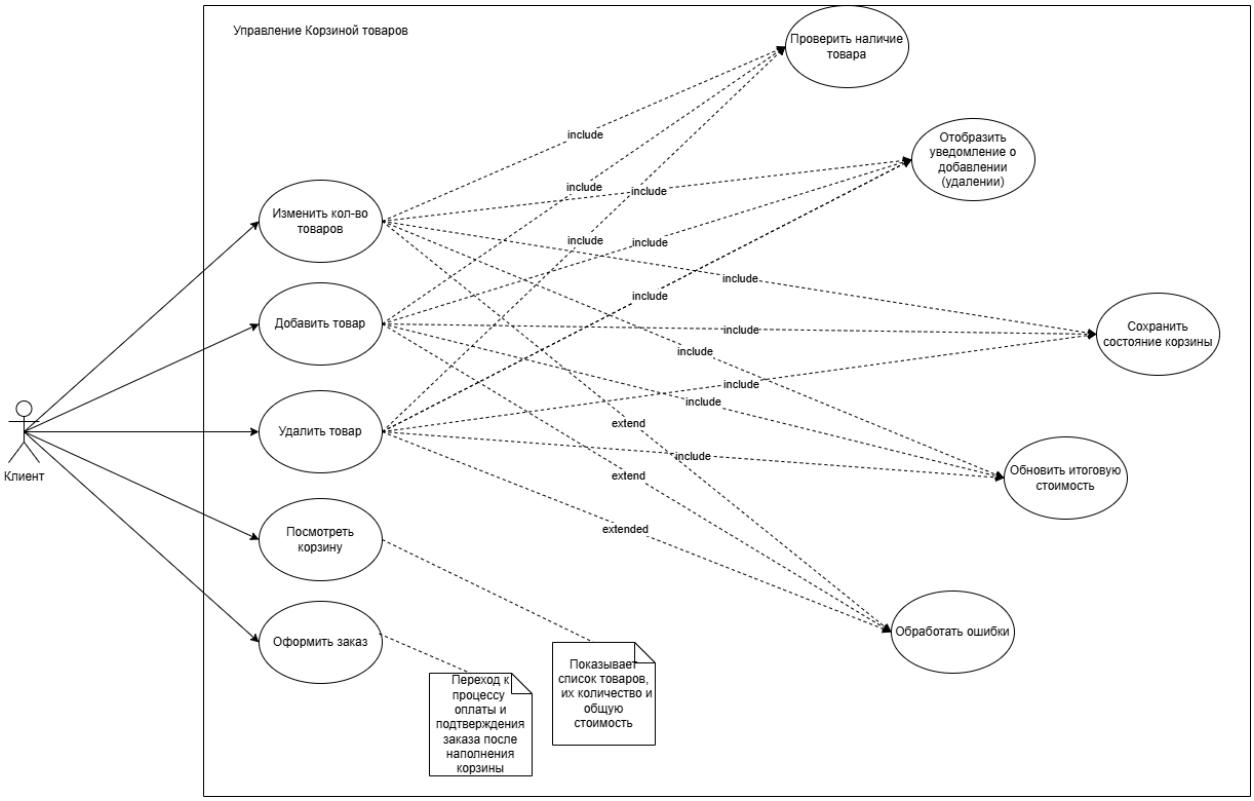



Клиент (UI)
Корзина (CartService)
Склад (WMS)
Изменение корзины
Запрос на изменение (удаление/изменение количества)
alt
[Удалить товар]
Удалить товар
[Добавить товар]
Проверить наличие нового количества
Результат проверки
alt
[Товара достаточно]
Обновить количество
[Товара недостаточно]
"Недостаточно товара на складе"
Пересчитать общую стоимость
Обновленные данные корзины
Клиент (UI)
Корзина (CartService)
Склад (WMS)

Сценарии использования

Сценарии использования для бизнес-кейса «Планирование»

Сценарии взаимодействия Пользователя с Корзиной товаров представлены на Use Case диаграмме.



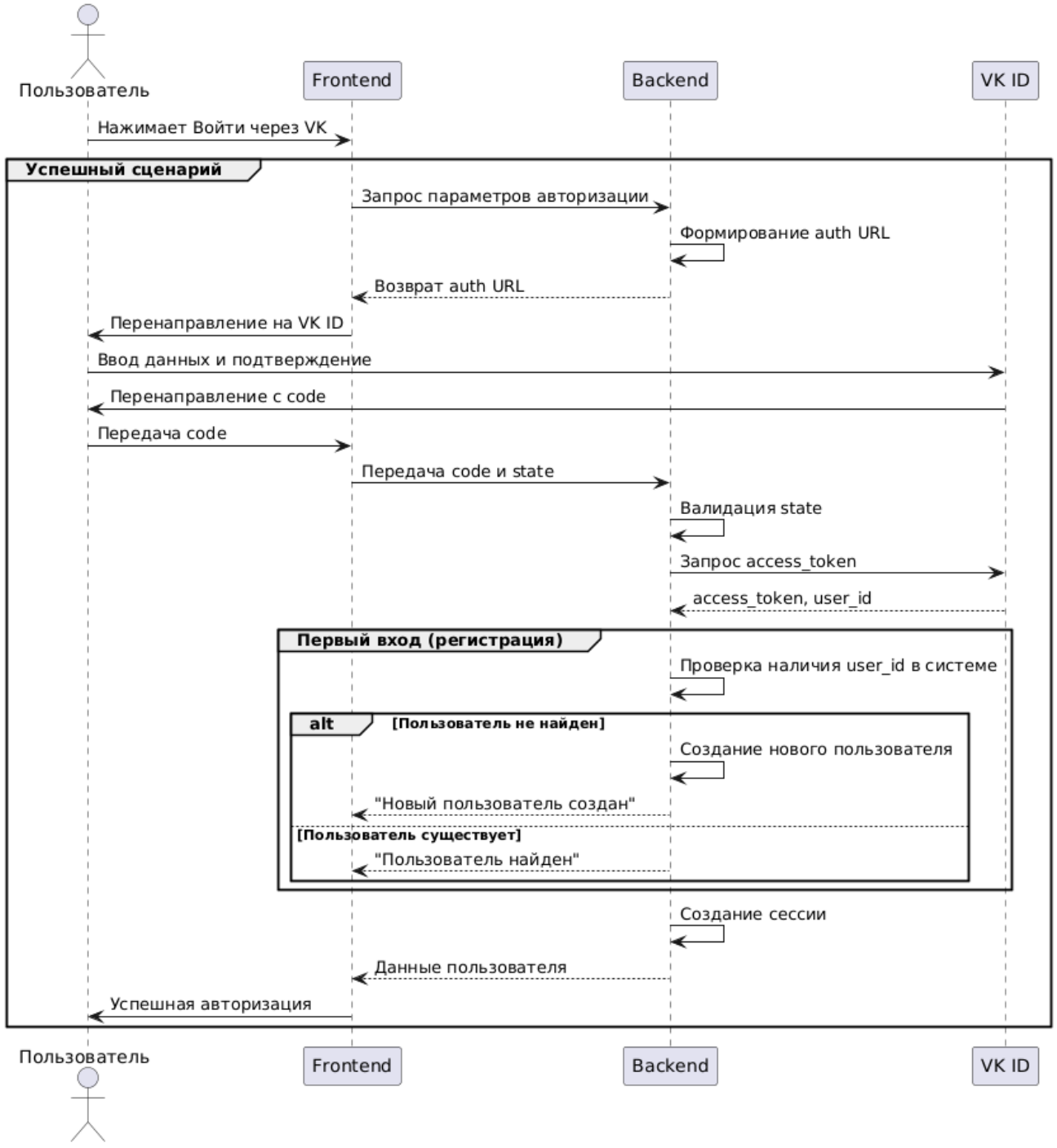
Сценарии использования для управления Корзиной товаров.

Название UC	**Управление корзиной товаров**
Роли, участвующие в UC	Покупатель
Предусловие	Покупатель авторизован в системе и имеет доступ к каталогу товаров.
Основной сценарий	**Добавление товара в корзину:**
	1. Покупатель выбирает товар из каталога.
	2. Покупатель нажимает кнопку "Добавить в корзину".
	3. Система проверяет доступность товара: • Если товар доступен, система добавляет его в корзину и обновляет общую сумму. • Если товар недоступен, система уведомляет Покупателя.
	4. Покупатель может просматривать содержимое корзины, изменять количество товаров или удалять их.
	5. Для оформления заказа Покупатель нажимает кнопку "Перейти к оплате".
Дополнительный сценарий	**Просмотр корзины:**
	1. Покупатель открывает страницу корзины.
	2. Система отображает список товаров в корзине, их количество, стоимость и общую сумму
Дополнительный сценарий	**Изменение количества товара:**
	1. Покупатель изменяет количество товара в корзине.
	2. Система проверяет доступность нового количества:
	• Если товар доступен в нужном количестве, система обновляет сумму. • Если товар недоступен, система уведомляет Покупателя.
Дополнительный сценарий	**Удаление товара из корзины:**
	1. Покупатель выбирает товар в корзине.
	2. Покупатель нажимает кнопку "Удалить".
	3. Система удаляет товар из корзины и обновляет общую сумму.
Дополнительный сценарий	**Покупатель приступает к оформлению заказа:**
	1. Покупатель нажимает кнопку "Оформить заказ".
	2. Система проверяет: • Наличие товаров в корзине. • Доступность всех товаров в нужном количестве.
	3. Если проверка пройдена, система: • Передает сформированную корзину для оформления заказа. • Перенаправляет Покупателя на страницу оплаты.
Постусловие	Товары добавлены в корзину, и Покупатель может перейти к оформлению заказа.

Сценарии использования общесистемных функций

Сценарий использования «Авторизация пользователя»

Авторизация и аутентификация пользователя осуществляется с помощью VK ID и представлена на Sequence диаграмме ниже.



Ссылку для перехода формирует Backend. URL должен содержать:

- client_id: ID приложения, полученный при регистрации приложения в VK,
- redirect_uri: обратный URL приложения,
- scope: список разрешений,
- response_type: code (для получения кода авторизации),
- state: рекомендуется для защиты от CSRF-атак.

В целях безопасности state должен быть одноразовым. Макс. время жизни code – 10 мин. Все ошибки логируются. Ответ VK ID содержит:

- access_token
- expires_in
- user_id
- email (если запрошен)

Авторизация через VK ID используется только как внешний провайдер аутентификации. После успешной аутентификации система выдает стандартный JWT-токен для доступа к API.

Сценарий использования «Настройка ролевой модели пользователей»

Формирование прав доступа для ролей в системе FoodMart определяется на основе следующих требований:

Пользователь системы (клиент). Клиент видит только активные товары, чтобы избежать путаницы с недоступными или удаленными позициями. Максимальное количество товаров в корзине ограничено 50, чтобы предотвратить массовое резервирование. Корзина сохраняется между сессиями для удобства, но не позволяет ему резервировать товары без оформления заказа. Отмена заказа возможна только до оплаты, чтобы избежать проблем с логистикой и возвратами. Клиент не может видеть или изменять чужие заказы. Оплата только банковскими картами. Возвраты возможны только через поддержку, чтобы предотвратить неконтролируемые списания средств. Получает уведомления о статусе заказа. Имеет возможность получать рекламные рассылки и имеет возможность отключить эту опцию.

Администратор системы. Имеет полный CRUD-функционал (создание, чтение, обновление, удаление) для управления Каталогом товаров. Нет ограничений на категории товаров. Работа с ценами доступна только через акционные механизмы.

Базовые цены редактируются через отдельный (недоступный) интерфейс. Имеет видимость всех складов системы. Может резервировать только подтвержденные заказы. Может проводить пассивный мониторинг Доставки без возможности изменять параметры доставки. Ему доступны исторические данные. Может формировать отчеты с настраиваемыми периодами.

Роль	Сущность	Операции	Ограничения
Клиент	Каталог товаров	- Просмотр списка товаров - Просмотр деталей товара (описание, цена, фото)	Только активные товары (не удаленные)

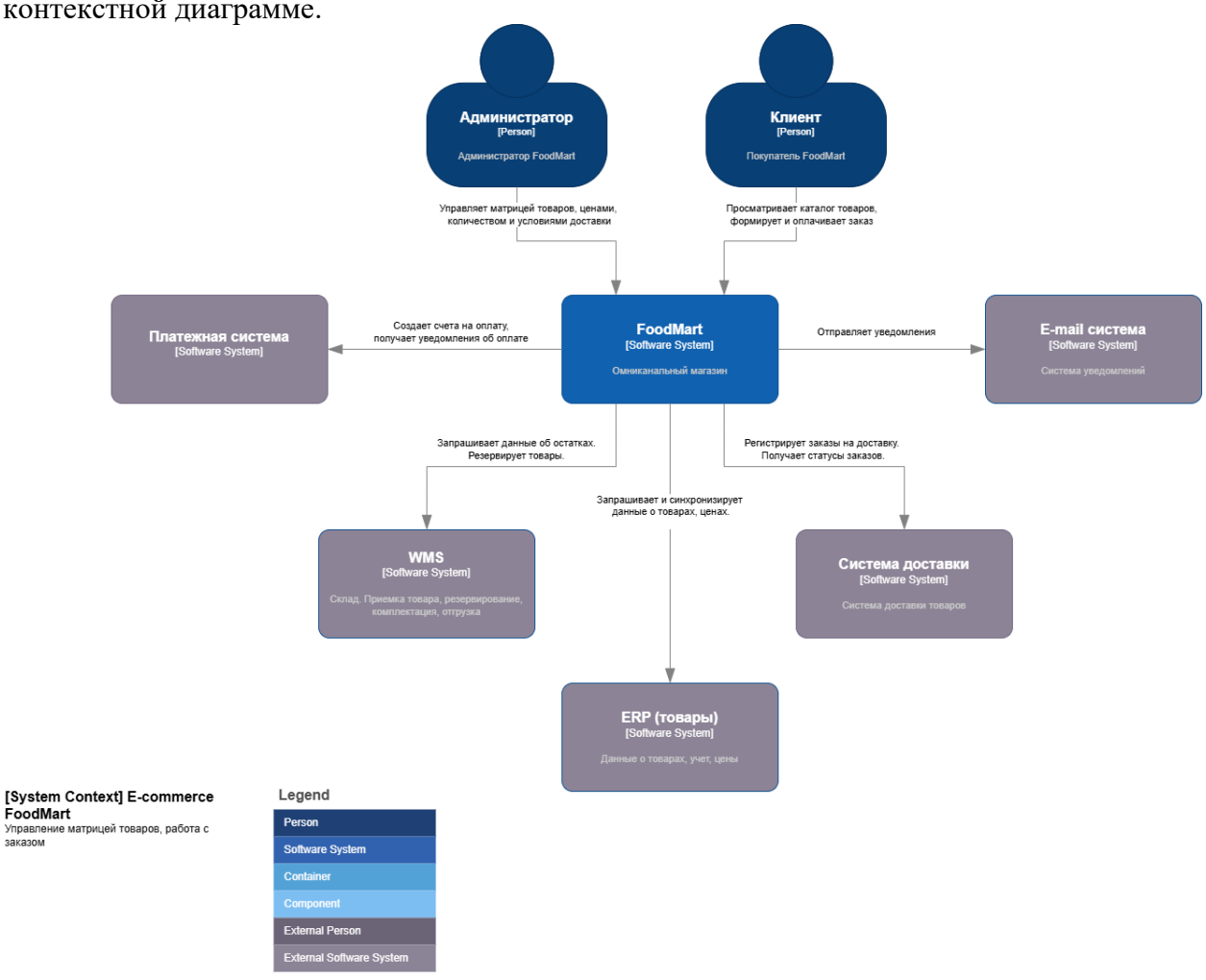
Роль	Сущность	Операции	Ограничения
	Корзина товаров	- Добавление товара в корзину - Изменение количества - Удаление товара	Максимальное количество товаров: 50
	Заказы	- Создание заказа - Просмотр истории заказов - Отмена заказа (до оплаты)	Доступ только к своим заказам
	Платежная система	- Оплата заказа (карта)	Оплата только своих заказов
	Уведомления	- Получение уведомлений (e-mail) о статусе заказа, акциях	Невозможность отключить уведомления о заказах
Администратор	Каталог товаров	- Добавление нового товара - Редактирование атрибутов (название, описание, цена) - Удаление/архивация товаров	Нет
	Скидки и акции	- Создание и управление акциями (скидки, промокоды) - Изменение цен через акции	Запрет на прямое редактирование базовых цен
	Склад (WMS)	- Просмотр текущих остатков - Резервирование товаров под заказы	Нет доступа к управлению поставками
	Заказы	- Просмотр всех заказов системы - Изменение статусов заказов - Отмена заказов	Нет
	Доставка	- Мониторинг статусов доставок - Просмотр маршрутов	Нет доступа к настройке зон/тарифов
	Платежная система	- Просмотр всех транзакций - Формирование платежных отчетов	Запрет на редактирование транзакций
	Уведомления	- Настройка шаблонов email-уведомлений - Ручная отправка уведомлений	Нет
	Отчеты	- Генерация отчетов (продажи, остатки, выручка) - Экспорт данных в XLS/PDF	Нет

Архитектура решения

FoodMart является центральной системой, координирующей процессы покупки товаров (продуктов питания) пользователями на электронной площадке, а также оформления доставки товаров покупателям. FoodMart, как интернет-магазин, предоставляет следующие функции:

- Управление каталогом товаров (цены, наличие, условия доставки).
- Оформление и оплата заказов.
- Резервирование товаров и контроль остатков.
- Взаимодействие с системами складирования, учета и доставки.
- Отправка уведомлений пользователям.

Система в целом и ее интеграционные потоки с внешними системами рассмотрены на контекстной диаграмме.



Ключевые участники (Actors):

1. Администратор FoodMart
    - Управляет товарами, ценами и условиями доставки.
    - Создает счета, получает уведомления об оплатах.

- Контролирует выполнение заказов.

2. Клиент (Покупатель)
    - Просматривает каталог товаров.
    - Формирует и оплачивает заказы.
    - Получает уведомления о статусе заказа.

Внешние системы:

1. E-mail система. Отправляет уведомления клиентам и администраторам (о заказах, оплатах, доставке).
2. WMS. Управляет складскими процессами: приемка, размещение, комплектация, отгрузка товаров.
3. ERP. Содержит данные о товарах, их учете и ценах.
4. Платежная система. Осуществляет функции приема и обработки платежей, а также отвечает за отправку статусов оплат в FoodMart.
5. Система доставки. Обеспечивает логистику: расчет сроков, маршрутов, отслеживание статусов доставки.

Взаимодействия:

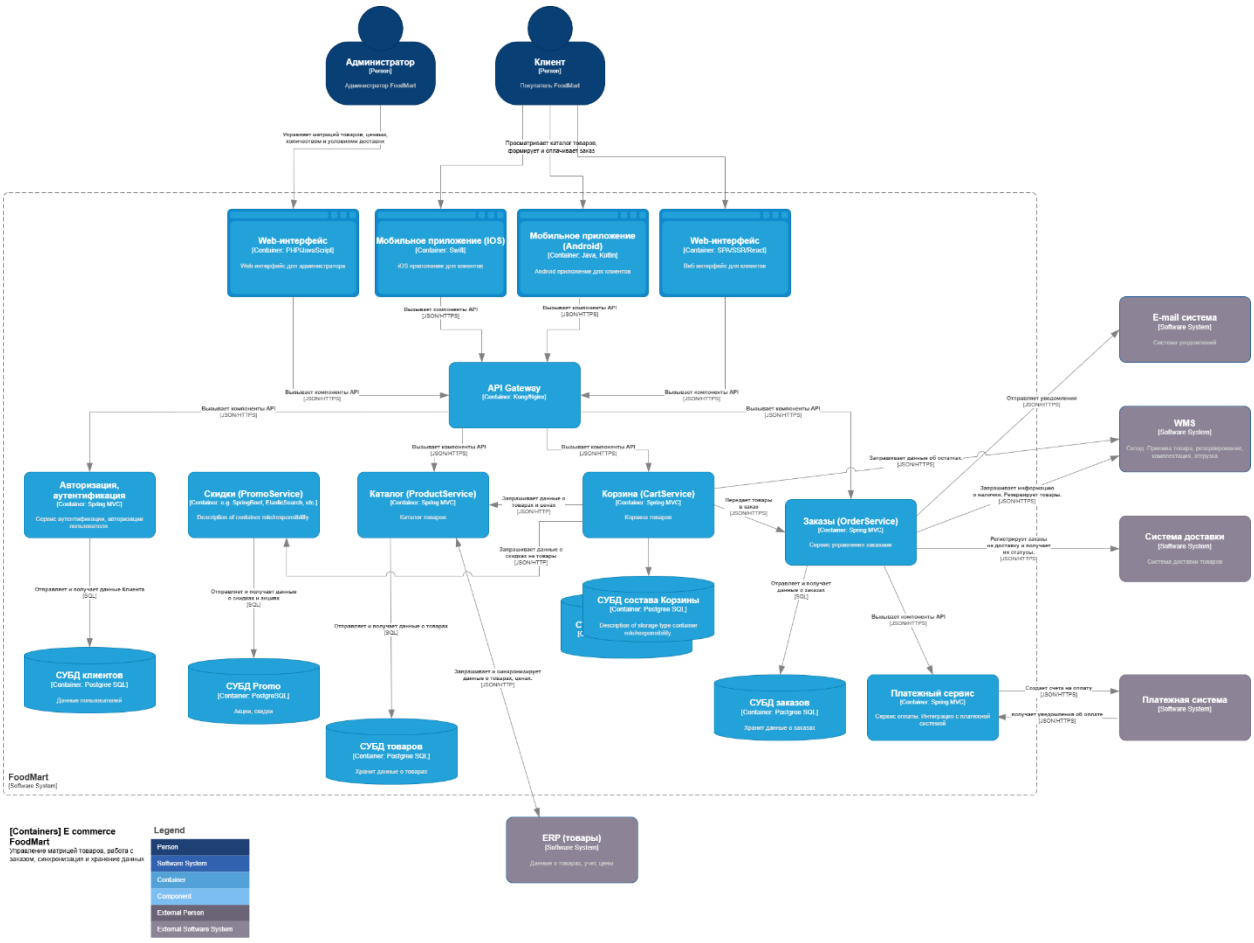
1. Клиент взаимодействует с FoodMart для:
    - Просмотра каталога товаров.
    - Оформления и оплаты заказов.
    - Получения уведомлений о статусе заказа.
2. Администратор управляет системой через FoodMart, который:
    - Синхронизирует данные о товарах с ERP.
    - Обновляет информацию о наличии через WMS.
3. FoodMart взаимодействует с внешними системами:
    - Передает платежные данные в Платежную систему и получает статус оплаты.
    - Запрашивает данные о доставке у Системы доставки.
    - Отправляет уведомления клиентам и администраторам через E-mail систему.
4. WMS и ERP обеспечивают FoodMart актуальной информацией:
    - WMS – о складских остатках и резервировании товаров.
    - ERP – о ценах, ассортименте и учетных данных.

Границы системы:

- FoodMart является центральной системой, координирующей процессы.
- Внешние системы (Платежная система, WMS, ERP, E-mail, Система доставки) взаимодействуют с FoodMart, но не между собой (на данном уровне абстракции).

Приведенная ниже контейнерная диаграмма детализирует архитектурные компоненты (контейнеры) проектируемой системы FoodMart, их технологии и взаимодействия. Она раскрывает:

- Фронтенд- и бэкенд-сервисы.
- Способы хранения данных.
- Интеграции с внешними системами



Ключевые контейнеры и их функции:

1. Фронтенд-приложения:
   - Web-интерфейс для администратора: управление товарами, заказами, ценами.
   - Мобильные приложения для клиентов (iOS и Android): просмотр каталога, оформление заказов, оплата.
   - Web-интерфейс для клиентов: аналогичные функции, доступные через браузер.
2. API Gateway:
   - Единая точка входа для всех API-запросов.
   - Маршрутизация запросов к микросервисам.
3. Сервисы аутентификации и авторизации:
   - Управление учетными данными пользователей.
   - Контроль доступа к ресурсам.
   - Сервис аутентификации поддерживает:
     - Локальную аутентификацию
     - Внешних OAuth-провайдеров (VK ID и други
     - Во всех случаях выдается JWT-токен стандартного формата
4. Базы данных:
   - СУБД клиентов: хранение профилей пользователей.

- СУБД товаров: каталог, цены, остатки.
- СУБД корзин товаров: состав, количество.
- СУБД заказов: история заказов, статусы.
- СУБД акций и скидок.

5. Внешние интеграции:
    - WMS (складская система): Резервирование и отгрузка товаров.
    - Платежная система: Обработка транзакций.
    - Система доставки: Логистика и трекинг.
    - E-mail система (уведомления): Отправка писем клиентам.

Взаимодействия между контейнерами

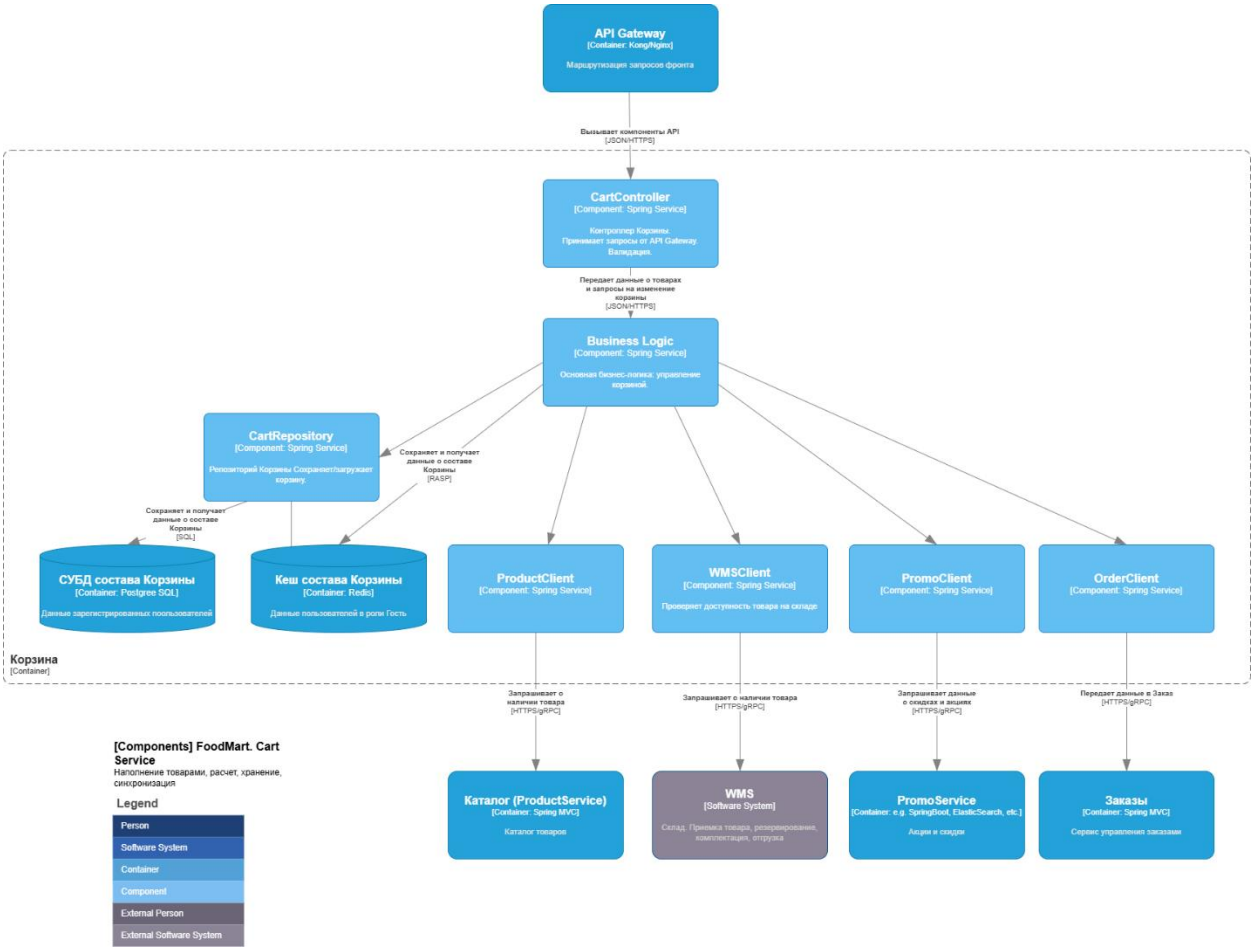
1. Клиентские приложения – API Gateway:
    - Отправка запросов на создание заказов, оплату, аутентификацию.
2. API Gateway – Сервис аутентификации:
    - Проверка токенов доступа.
3. Сервисы – СУБД:
    - Запросы к базам данных (товары, заказы).
4. Backend* – WMS/ERP/Платежная система:
    - Синхронизация данных о товарах и платежах.

Примечание*:
ERP: частота 1 раз в сутки (пакетная синхронизация), протокол REST API, данные – полный дамп товаров (цены, остатки).
WMS: Проверка остатков через gRPC в реальном времени (при операциях с корзиной).

Технологический стек

Контейнер	**Технологии**
Web-интерфейс (админ)	JavaScript
Мобильные приложения	Swift (iOS), Java/Kotlin (Android)
API Gateway	Kong/Nginx
Сервисы: Каталог товаров, Корзина товаров, Заказы, Скидки	Spring MVC
Аутентификация	Spring MVC
Базы данных	PostgreSQL

Детализация сервиса Корзина товаров (CartService), классы и компоненты, взаимодействие с другими системами представлены на компонентной диаграмме.



1. Входные точки
   - API Gateway
     Принимает HTTP-запросы от фронтенда и перенаправляет их в CartController.
   - CartController
     Обрабатывает входящие запросы, валидирует данные, передает их в Business Logic.
2. Бизнес-логика
   - Business Logic
     Ядро сервиса, отвечает за:
     - Формирование корзины
     - Расчет итоговой стоимости (с учетом скидок через PromoClient)
     - Проверку наличия товаров (через WMSClient)
     - Подтверждение заказа (через OrderClient).
3. Работа с данными

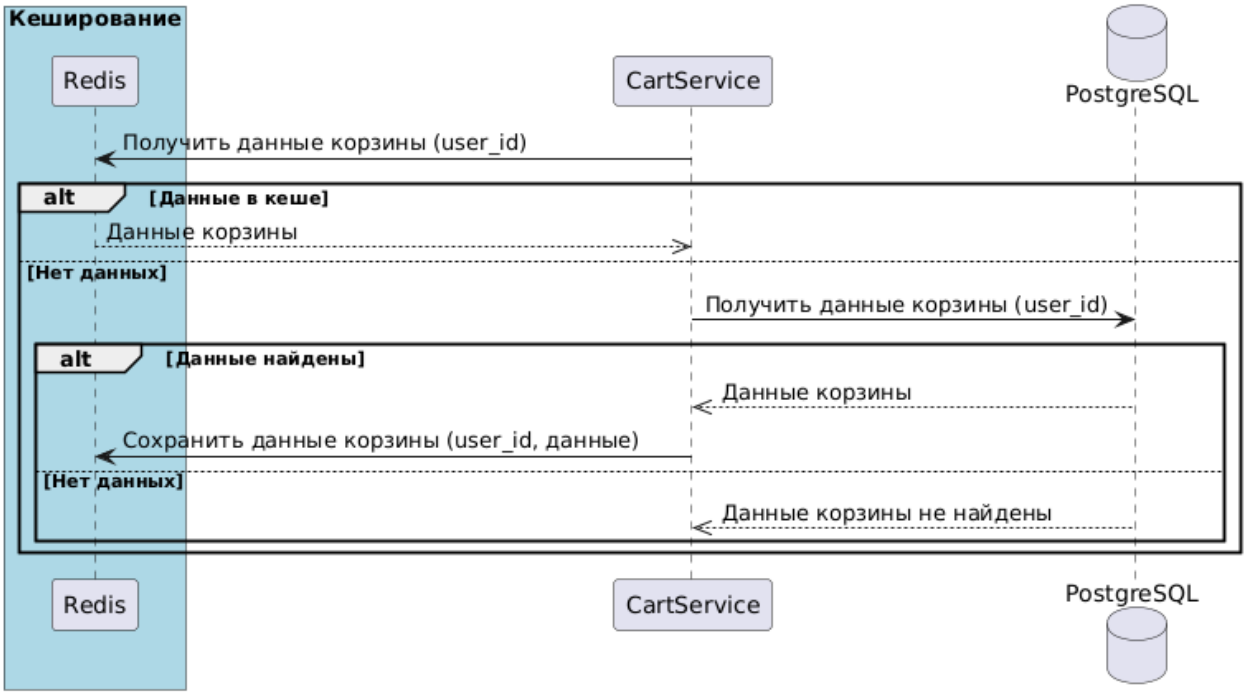
- CartRepository. Абстракция для доступа к данным корзины в PostgreSQL:
- Redis. Кэширует часто запрашиваемые данные (содержимое корзины).

4. Интеграции с внешними сервисами

Компонент	Назначение	Протокол
ProductClient	Получение данных о товарах	HTTP/gRPC
WMSClient	Проверка наличия на складе	HTTP/gRPC
PromoClient	Расчет скидок и акций	HTTP/gRPC
OrderClient	Создание заказа после оформления	HTTP/gRPC

5. Внешние сервисы
- ProductService — каталог товаров (цена, описание).
- WMS — система управления складом.
- PromoService — правила скидок.
- OrderService — обработка подтвержденных заказов.

6. Для оптимизации работы с данными корзин пользователей применяется кэширование.



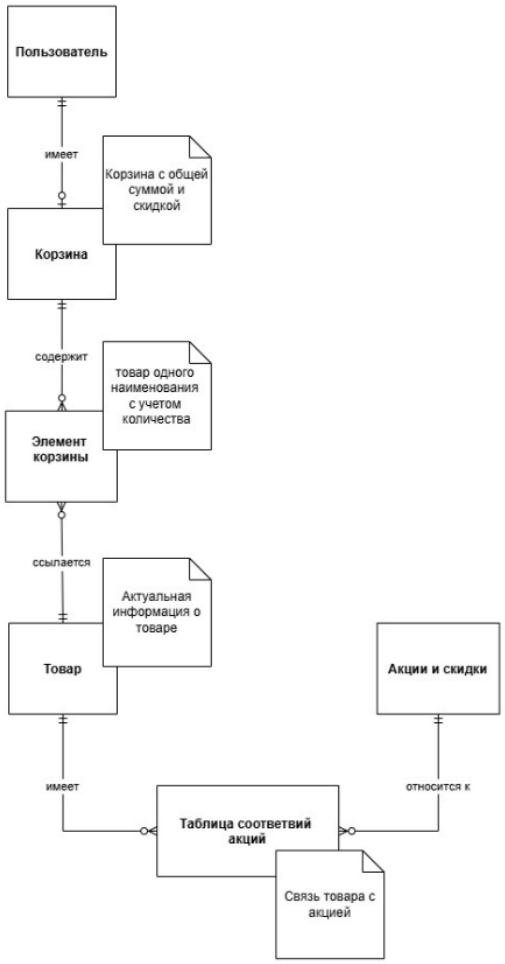
Модель данных

Логическая модель данных
Модель описывает процесс работы корзины интернет-магазина с поддержкой скидок.

- **Пользователь** может иметь не более одной активной корзины.
- **Корзина** содержит элементы с товарами, которые могут быть весовыми или штучными.
- **Товары** связаны с акциями через промежуточную таблицу, что позволяет применять разные скидки к одному товару.
- **Акции** имеют временные рамки и могут быть архивными.

Модель оптимальна для реляционных СУБД, обеспечивает четкую структуру данных и предсказуемость запросов. Предлагается к использованию СУБД PostgreSQL.

Процесс набора и расчета корзины представлен на диаграмме. Авторизованный пользователь добавляет товар в корзину, которая отображает общую сумму и возможные скидки. Корзина содержит элементы с учетом количества и актуальной информации о товаре.

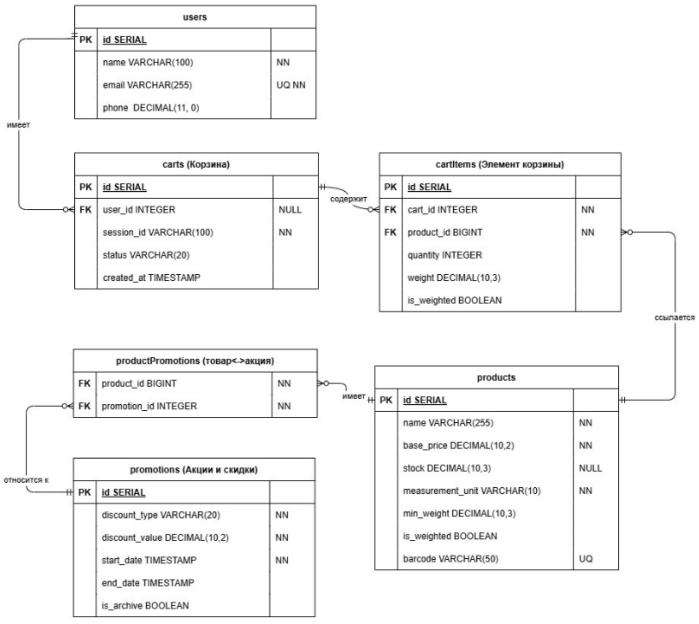


С учетом этого можно выделить основные сущности и их связи:

1. **Пользователь (User)**
    - Хранит данные пользователя.
    - Связан с корзиной (carts) через user_id.
2. **Корзина (Carts)**
    - Содержит информацию о корзине: статус, дата создания, связь с пользователем или сессией.
    - Связана с элементами корзины (cartitems) через cart_id.
3. **Элемент корзины (CartItems)**
    - Содержит данные о товарах в корзине: количество, вес, связь с товаром и корзиной.
    - Связана с товарами (products) через product_id.
4. **Товар (Products)**
    - Хранит информацию о товарах: название, цена, единица измерения, штрих-код и т.д.
    - Связана с акциями (productPromotions) через product_id.
5. **Акции (Promotions)**
    - Содержит данные о скидках: тип, значение, сроки действия.
    - Связана с товарами через промежуточную таблицу productPromotions.

Логическая модель базы данных в виде ER-диаграммы, приведенная ниже, описывает структуру таблиц и связи между ними для реализации функционала корзины с товарами и скидками. Модель предназначена для организации хранения данных в СУБД (PostgreSQL) и обеспечивает:

1. Учет товаров в корзине (включая весовые).
2. Применение скидок через акции.
3. Разделение данных пользователей, корзин и товаров.



Описание таблиц СУБД и их атрибутов

Таблица	Атрибут	Тип данных	Ограничения	Описание
users	id	SERIAL	PK	Уникальный идентификатор пользователя.
	name	VARCHAR(100)	NOT NULL	Имя пользователя.
	email	VARCHAR(255)	NOT NULL, UNIQUE	Электронная почта пользователя.
	phone	DECIMAL(11, 0)	-	Номер телефона пользователя.
carts	id	SERIAL	PK	Уникальный идентификатор корзины.
	user_id	INTEGER	-	ID пользователя (может быть NULL для гостей).
	session_id	VARCHAR(100)	NOT NULL	Идентификатор сессии.
	status	VARCHAR(20)	-	Статус корзины (например, "активная", "оформленная").
	created_at	TIMESTAMP	-	Дата и время создания корзины.
cartitems	id	SERIAL	PK	Уникальный идентификатор элемента корзины.
	cart_id	INTEGER	NOT NULL	ID корзины, к которой относится элемент.
	product_id	BIGINT	NOT NULL	ID товара.
	quantity	INTEGER	-	Количество товара.
	weight	DECIMAL(10, 3)	-	Вес товара (если товар весовой).
	is_weighted	BOOLEAN	-	Флаг, указывающий, является ли товар весовым.
products	id	SERIAL	PK	Уникальный идентификатор товара.
	name	VARCHAR(255)	NOT NULL	Название товара.
	base_price	DECIMAL(10, 2)	NOT NULL	Базовая цена товара.
	stock	DECIMAL(10, 3)	-	Количество товара на складе.
	measurement_unit	VARCHAR(10)	NOT NULL	Единица измерения (например, "кг", "шт").
	min_weight	DECIMAL(10, 3)	-	Минимальный вес для весовых товаров.
	is_weighted	BOOLEAN	-	Флаг, указывающий, является ли товар весовым.
	barcode	VARCHAR(50)	UNIQUE	Штрих-код товара.
promotions	id	SERIAL	PK	Уникальный идентификатор акции.
	discount_type	VARCHAR(20)	NOT NULL	Тип скидки (например, "процент", "фиксированная").
	discount_value	DECIMAL(10, 2)	NOT NULL	Значение скидки.
	start_date	TIMESTAMP	NOT NULL	Дата начала действия акции.
	end_date	TIMESTAMP	-	Дата окончания действия акции.
	is_archive	BOOLEAN	-	Флаг, указывающий, является ли акция архивной.

product_promotions	product_id	BIGINT	PK, NOT NULL	ID товара, связанного с акцией.
	promotion_id	INTEGER	PK, NOT NULL	ID акции, связанной с товар

Описание API

Интеграция Корзины товаров FoodMart с такими компонентами системы как мобильные приложения (Android, iOS), Web-интерфейс пользователя и внутренние модули системы (Backend) необходима для управления содержимым корзины пользователя в реальном времени со стороны систем магазина и для взаимодействия с ней.

Все взаимодействия осуществляются через REST API с использованием стандартных HTTP-методов и формата данных JSON. Для обеспечения безопасности и контроля доступа применяется авторизация по Bearer-токенам, которые передаются в заголовках запросов.

Все API-запросы требуют JWT-токена в заголовке Authorization.

Токен может быть получен:

- Через стандартный процесс аутентификации (логин/пароль)
- Через OAuth-провайдеров (например, VK ID) с последующей выдачей JWT

API поддерживает основные операции: добавление товара в корзину, удаление товаров, обновление количества, получение текущего содержимого корзины и стоимости. Каждая операция сопровождается проверками и обработкой ошибок, что обеспечивает надежность взаимодействия.

Все вызовы осуществляются по защищенному протоколу HTTPS, что гарантирует безопасность передаваемых данных. В случае ошибок или некорректных запросов API возвращает стандартные коды ошибок (например, 400 – неправильный запрос, 401 – неавторизованный, 404 – не найден).

Документация включает описание всех эндпоинтов, параметры, форматы данных, примеры запросов и ответов, а также инструкции по тестированию и мониторингу. Такой подход обеспечивает стабильную и безопасную интеграцию с внешними системами и ускоряет гибкость для внедрения новых решений.

Выделены основные сущности: Корзина (Cart) и Товар в корзине (CartItem), и приведено их описание.

Табл.

Поле	**Тип**	**Обязательное**	**Описание**	**Пример**
Корзина (Cart)				
id	integer (int64)	Да	Уникальный ID корзины	12345
items	Массив CartItem	Да	Список товаров в корзине	(см. ниже)
total	number (float)	Да	Общая стоимость корзины	1500.75
Товар в корзине (CartItem)*				
id	integer (int64)	Да	Уникальный ID позиции в корзине	101
productId	integer (int64)	Да	ID товара в каталоге	500
name	string	Да	Название товара	"Молоко 2,5%"
quantity	number (float)	Да	Количество (мин. 0.001)	2.5 (кг/л), 1 (шт)

Примечание:* Необходимость поля price отсутствует, т.к. цена берется из каталога товаров в реальном времени.

Ниже приведены некоторые запросы, их описание.

Табл.

Поле	Тип	Обязательное	Описание	Пример
Запрос на добавление товара (CartItemRequest)				
productId	integer (int64)	Да	Должен существовать в БД	500
quantity	number (float)	Нет (default: 1)	minimum: 0.001	1.5 (2 пакета)
Запрос на изменение количества (UpdateQuantity)				
itemId	integer (int64)	Да	Должен существовать в корзине	101
quantity	number (float)	Да	minimum: 0.001	0.5 (500 г)

Ниже приведены атрибуты ошибок, их описание.

Табл.

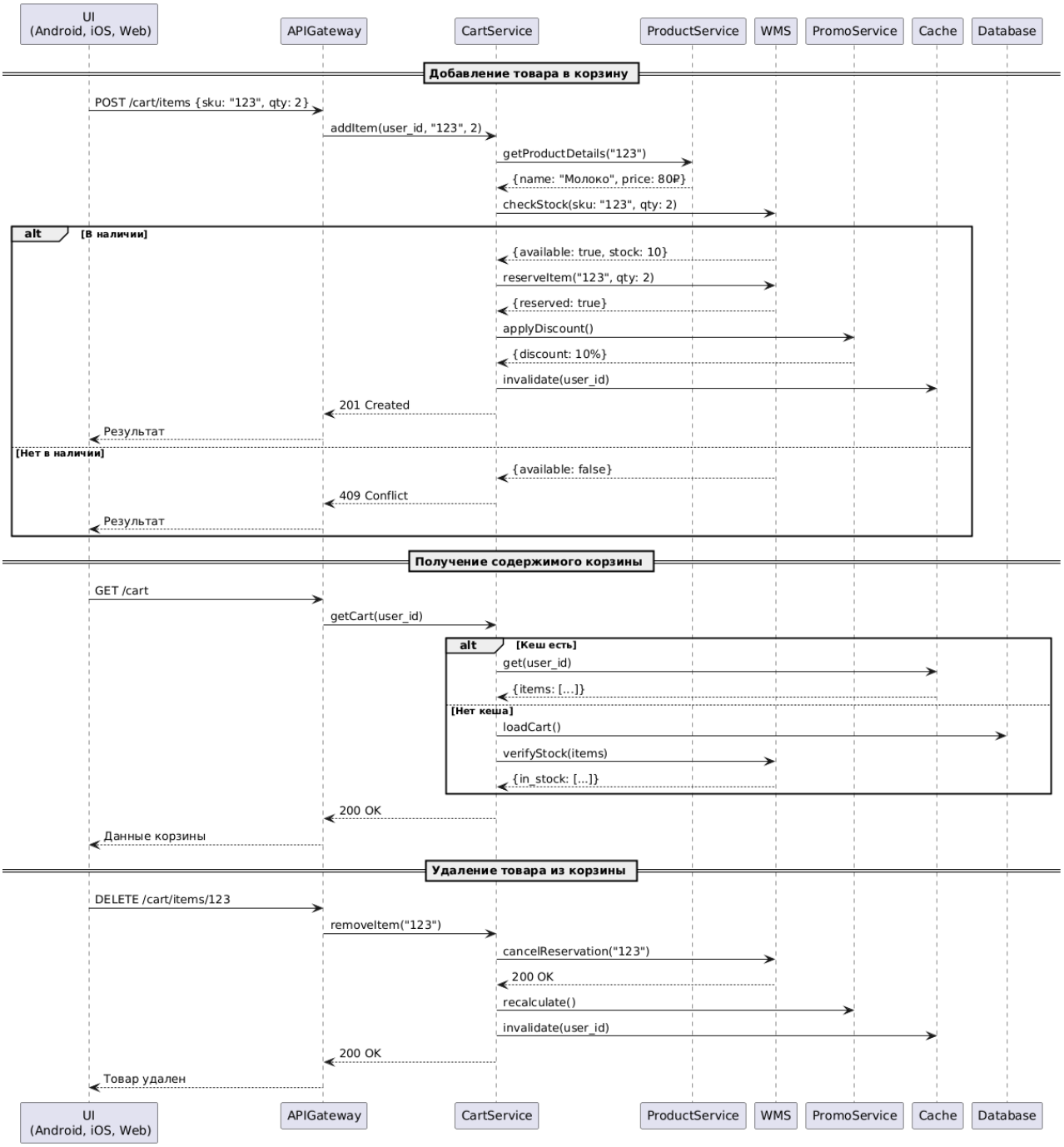
Поле	Тип	Обязательное	Описание	Пример
Ответ с ошибкой (Error)				
error	string	Да	Код ошибки (категория)	"NOT_FOUND"
message	string	Да	Человекочитаемое описание	"Товар не найден в корзине"
details	Массив string	Нет	Дополнительные детали (например, валидация)	["Количество не может быть 0"]

Для взаимодействия систем с Корзиной товаров разработаны следующие методы API.

Табл.

Метод	Эндпоинт	Описание	Аутентификация
POST	/cart	Добавить товар в корзину (в шт., кг, л и т.д.)	Да (JWT)
DELETE	/cart/items/{itemId}	Удалить товар из корзины	Да (JWT)
GET	/cart	Получить текущее содержимое корзины (список товаров, общую стоимость)	Да (JWT)
PUT	/cart/items/{itemId}	Изменить количество товара в корзине	Да (JWT)
POST	/cart/clear	Очистить корзину (удалить все товары)	Да (JWT)
GET	/cart/total	Получить итоговую стоимость корзины	Да (JWT)
POST	/cart/checkout	Оформить заказ (перевод корзины в статус "заказ")	

На диаграмме последовательности представлен жизненный цикл корзины товаров и показано взаимодействие между сервисами в рамках управления корзиной, включая добавление, проверку наличия, применение скидок, удаление товаров и обновление данных.



В связи с внедрением интеграций и рамках наблюдения за процессом взаимодействия и состоянием систем подготовлены рекомендации по постановке интеграционных процессов на мониторинг. Предложены значения метрик для триггеров уведомлений службы поддержки.

Табл. Интеграционные процессы для мониторинга

Процесс	Метрика	Порог
Добавление товара в корзину	Время выполнения end-to-end	Целевое время выполнения ≤500 мс (стандарт). Порог >2 сек допустим только при высокой нагрузке или задержках интеграций (WMS, PromoService). Превышение требует анализа. Технический порог: > 2 с (предупреждение), > 5 с (критично)
	Ошибки проверки наличия	> 5% от запросов за 5 мин
	Сбои резервирования	> 3% запросов
Проверка наличия (WMS)	Время ответа WMS	> 1 с
	Таймауты запросов	1 запрос/мин
Обновление Каталога		
Работа с кешем	Соотношение попаданий в кеш	< 90% (для часто запрашиваемых корзин)
	Ошибки инвалидации	> 10 ошибок/час
Промо-сервис (скидки)	Время расчета скидок	800 мс (p99)
	Ошибки применения скидки	> 2% запросов

Определены потоки данных, где критичны асинхронность, масштабируемость и отказоустойчивость:

1. События изменений корзины. Добавление, удаление товаров, изменение количества, применение скидок.
2. Резервирование товаров (WMS). Запросы на резервирование, отмену резервации.
3. Инвалидация кеша. События изменения данных (цена товара, остатки, скидки).
4. Обработка промо-акций. Применение или отмена скидок, персональные предложения.

Для указанных потоков данных рекомендуется использовать интеграцию через Kafka, где:

- Поддерживаемый рабочий режим: 1К сообщений/сек (средняя нагрузка)

- Пропускная способность: 10К сообщений/сек (пиковая нагрузка)

Рекомендации по мониторингу Kafka:

- Превышение 1К сообщений/сек более 5 минут – триггер для предупреждения (возможный рост нагрузки).
- Превышение 10К сообщений/сек – критический инцидент (требует немедленного реагирования).

Сценарий реагирования: автомасштабирование кластера Kafka при нагрузке >70% CPU.

Приложение. Листинг. Спецификация API

```yaml
openapi: 3.0.0
info:
  title: ServiceCart API
  version: 1.0.0
  description: API для управления Корзиной товаров FoodMart
servers:
  - url: https://api.foodmart.ru/v1
    description: FoodMart server

paths:
  /cart:
    post:
      tags: [Cart]
      summary: Добавить товар в корзину
      security:
        - bearerAuth: []
      requestBody:
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/CartItemRequest'
      responses:
        '200':
          description: Товар успешно добавлен
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Cart'
        '400':
          description: Неверный запрос
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Error'
              example:
                error: "BAD_REQUEST"
                message: "Некорректные данные запроса"
        '401':
          description: Требуется авторизация
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Error'
              example:
                error: "UNAUTHORIZED"
                message: "Требуется аутентификация"
        '404':
          description: Товар не найден
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Error'
              example:
```

```
                error: "NOT_FOUND"
                message: "Товар не найден"
        '500':
          description: Ошибка сервера
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Error'
              example:
                error: "SERVER_ERROR"
                message: "Внутренняя ошибка сервера"

  /cart/items/{itemId}:
    delete:
      tags: [Cart]
      summary: Удалить товар из корзины
      security:
        - bearerAuth: []
      parameters:
        - name: itemId
          in: path
          required: true
          schema:
            type: integer
            format: int64
      responses:
        '200':
          description: Товар успешно удалён
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Cart'
        '404':
          description: Товар не найден
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Error'
              example:
                error: "NOT_FOUND"
                message: "Элемент корзины не найден"
        '500':
          description: Ошибка сервера
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Error'
              example:
                error: "SERVER_ERROR"
                message: "Внутренняя ошибка сервера"

components:
  securitySchemes:
    bearerAuth:
      type: http
      scheme: bearer
      bearerFormat: JWT
```

```yaml
schemas:
  Cart:
    type: object
    properties:
      id:
        type: integer
        format: int64
      items:
        type: array
        items:
          $ref: '#/components/schemas/CartItem'
      total:
        type: number
        format: float

  CartItem:
    type: object
    properties:
      id:
        type: integer
        format: int64
      productId:
        type: integer
        format: int64
      name:
        type: string
      quantity:
        type: number
        format: float
        minimum: 0.001

  CartItemRequest:
    type: object
    required:
      - productId
    properties:
      productId:
        type: integer
        format: int64
      quantity:
        type: number
        format: float
        default: 1
        minimum: 0.001

  UpdateQuantity:
    type: object
    required:
      - itemId
      - quantity
    properties:
      itemId:
        type: integer
        format: int64
      quantity:
        type: number
```

```
        format: float
        minimum: 0.001

    Error:
      type: object
      properties:
        error:
          type: string
        message:
          type: string
        details:
          type: array
          items:
            type: string
      required:
        - error
        - message
```

Нефункциональные требования

Для обеспечения надежности, производительности, безопасности и других аспектов, которые напрямую не связаны с бизнес-логикой, но критичны для успешной работы системы, приведем качественные характеристики системы, ограничения и условия работы.

Табл. Атрибуты и метрики качества работы системы

Атрибут качества	**Сценарий**	**Метрика, показатели**	**Обоснование**
Производительность	Обработка операций с корзиной при пиковой нагрузке	Время отклика: - Добавление товара: ≤500 мс (в нормальных условиях) - Получение корзины: ≤ 500 мс. - Допустимый максимум при пиковой нагрузке: ≤2 с Пропускная способность: 1000 RPS. Задержка обновления кеша: ≤ 1 сек. Задержка обновления СУБД: ≤ 24 ч.	Основано на UX-исследованиях (статистически 95-й перцентиль). Задержки >1 сек раздражают пользователей. ≤2 с допустимо при высокой нагрузке или сложных интеграциях (WMS, PromoService) Учтен запас: +25% к историческим данным.
Доступность	Работа системы 24/7	Общая доступность систем (Uptime): 99.5%, Время восстановления (MTTR): ≤ 15 мин.	Стандарт SLA для коммерческих моделей SaaS. 15 мин MTTR — баланс между затратами и потерями (статистические данные)
Надежность	Корректная обработка данных при сбоях	Потеря данных: 0% Ошибки резервирования: ≤ 0.1% Время синхронизации после сбоя: ≤ 5 мин	0% потерь — стандартное требование бизнеса. 0.1% ошибок — порог для ручного разбора аномалий.
Защищенность	Защита данных пользователей и транзакций	Аутентификация запросов: 100% JWT Шифрование данных: по требованиям ИБ Инциденты безопасности: ≤ 1/год Аудит действий: % логирования по требованиям ИБ	На основе данных о допустимых рисках для e-commerce.
Масштабируемость	Рост нагрузки в x3 раз	Пропускная способность Kafka: - Средняя нагрузка: 1К сообщ./сек - Пиковая нагрузка: 10К сообщ./сек (кратковременно)	Запас для пиков нагрузки (предпраздничные дни, дни проведения акций). На основе статистики:

Атрибут качества	Сценарий	Метрика, показатели	Обоснование
		Автомасштабирование: Правило добавления нод: нагрузка CPU > 70%, то добавить ноды за ≤ 2 мин.	– 1К/сек – базовая нагрузка – 10К/сек — кратковременные всплески

FoodMart это региональный продуктовый сервис. Расширение бизнеса в другие регионы не планируется. С точки зрения масштабов и планов развития бизнеса принимаются следующие допущения, позволяющие составить требования к качеству системы.

Табл. Список допущений, принятых на проекте

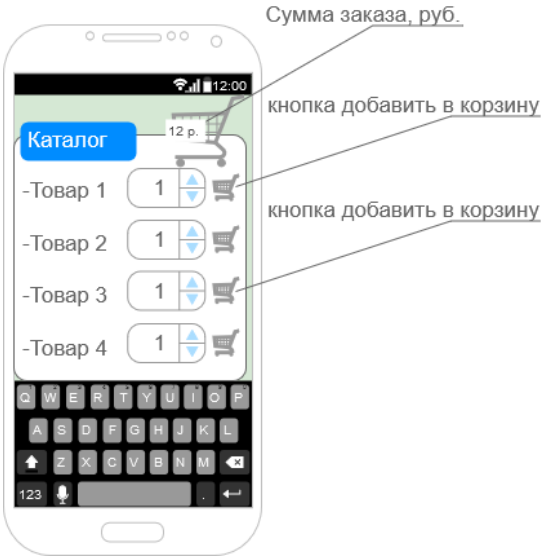
Категория	Допущение	Влияние на систему	Показатель
География	Работа только в одном регионе (без расширения)	Нет необходимости в мультирегиональной инфраструктуре	Все сервисы размещаются в одном дата-центре
Аудитория	Рост пользователей с 50K до 75K в течение 2 лет (+50%)	Умеренное увеличение нагрузки	Пропускная способность: 1000 RPS (с запасом 25%)
Ассортимент	Увеличение с 5K до 8K SKU (+60%)	Частота обновлений цен/остатков 1 раз в день	Инвалидация кеша: ≤5 минут
Пиковая нагрузка	Кратковременный рост в x3 раз (предпраздничные дни, дни проведения акций).	Временная потребность в дополнительных ресурсах	Возможность увеличения мощности на 30% за 15 мин.
Партнеры	Локальные интеграции (курьеры, поставщики)	Ограниченное число внешних зависимостей	SLA партнеров: ≥99,5% uptime
Данные	Аналитика для внутреннего использования (без монетизации)	Скромные требования к обработке событий	Пропускная способность очередей: 1K событий/сек
Пиковая нагрузка	Пиковая нагрузка Kafka (10К сообщ./сек) длится не более 2 часов в сутки (во время акций)	Временная потребность в дополнительных ресурсах	Автомасштабирование: +30% за 15 мин

Табл. Требования к качеству проектируемой системы.

Атрибут	Требование	Обоснование
Производительность	1000 RPS, время отклика ≤500 мс	Покрывает рост аудитории + запас 25%
Доступность	99.5%	Достаточно для регионального сервиса без критичных последствий
Масштабируемость	Поддержка нагрузок: - 1К сообщ./сек (средняя) - 10К сообщ./сек (пиковая, ≤1 час)	Покрывает акционные периоды и рост аудитории
Надежность	Потеря данных ≤0.01%	Гарантия сохранности заказов
Безопасность	TLS 1.2+, базовый аудит действий	Соответствие минимальным требованиям PCI DSS
Кеширование	Актуальность данных ≤5 минут	Баланс между нагрузкой и бизнес-потребностями

Экранные формы

Добавление товара в корзину



Блок экранной формы	Поле (значение по умолчанию)	Видимость/ Редактируемость (если отличается от критериев видимости)	Обязательность/ Валидация	Тип редактора
Каталог	Цена товаров в корзине (сумма заказа), руб.	Видим, нередактируемое (рассчитывается автоматически)	Необязательное (информационное поле)	Текстовое поле (read-only)
Элемент каталога (товар)	Поле ввода кол-ва товара (значение по умолчанию =1)	Видим, редактируемое	Обязательное, валидация: число > 0	Числовое поле

Добавление товара в корзину происходит при нажатии кнопки «Добавить в корзину»

Требования к миграции данных

Для системы FoodMart, в т.ч. работы с каталогом товаров и корзиной товаров необходимо предусмотреть следующие требования к миграции данных:

1. Интеграция с внешней ERP
    - Обеспечение синхронизации данных о товарах (актуальные цены, остатки, атрибуты) между ERP и новой системой.
    - Синхронизация с ERP должна происходить 1 раз в сутки (по расписанию, например, в 02:00 ночи).
        1. Метод: полный выгруз/загруз данных через REST API.
        2. Данные: цены, остатки, атрибуты товаров.
        3. Конфликты: приоритет данных из ERP.
        4. Мониторинг: логирование времени выполнения и ошибок.
2. Тестирование и откат
    - Проверка целостности данных после миграции.
    - Возможность отката в случае критических ошибок. Откат данных должен быть выполнен в течение ≤30 минут с момента обнаружения критической ошибки.
3. Документирование и логирование
    - Фиксация всех этапов миграции, включая ошибки и ручные корректировки.
    - Все этапы миграции должны логироваться в централизованную систему (например, ELK) с обязательными полями: тип операции, статус, идентификатор сущности, timestamp
4. Валидация данных после миграции. Проверка целостности (например, что все товары из корзин имеют корректные ссылки на product_id в ERP и WMS).