📘 ТЕХНИЧЕСКОЕ ЗАДАНИЕ — FUCENT v3.0

ЧАСТЬ 1 / 5

1. Концепция платформы

FUCENT — торговая платформа нового поколения для Кыргызстана и СНГ, объединяющая:

- маркетплейс товаров
- систему ленты как Instagram/TikTok
- чаты как Instagram DM
- карту магазинов
- POS-интеграцию
- систему рекомендаций
- аналитику для продавцов
- мобильное приложение (Flutter) и веб (Angular)

Платформа рассчитана на города (MVP — Бишкек), затем масштабирование по регионам и странам.

Цель: создать маркетплейс с сильной социальной составляющей, упором на видео-публикации, сторис и прямые эфиры магазинов.

---

2. Пользовательские роли

2.1 Покупатель (User)

Может:

- регистрироваться/логиниться

- просматривать ленту: тренды / рекомендации / подписки
- подписываться на магазины
- искать магазины и товары
- просматривать каталог
- добавлять товары в корзину
- оформлять заказы (если магазин поддерживает POS-инвентарь)
- переписываться в чатах
- получать уведомления
- оставлять отзывы на товары (только после покупки)
- оценивать магазины (агрегируется из оценок товаров)
- шарить товары и публикации

2.2 Магазин (Merchant)

Один merchant = юридическое лицо / владелец бренда.

Может:

- создавать аккаунт магазина
- создавать точки (Shop)
- назначать продавцов (Seller accounts)
- управлять товарами и публикациями
- вести прямые эфиры
- публиковать сторис
- общаться с клиентами
- смотреть аналитику

- включать режимы POS / HYBRID
- подавать жалобы на фейковые отзывы

2.3 Продавец (Seller / Employee)

Продавец привязан к **одной точке** (Shop).

Права продавца:

- вести чат от лица магазина
- создавать товары, варианты, публиковать посты
- обновлять остатки
- запускать прямой эфир
- загружать сторис
- получать уведомления по заказам
- видеть аналитику своей точки

Продавец **не видит** данные других филиалов магазина.

2.4 Админ (Platform Admin)

Права:

- модерировать товары, публикации, сторис
- модерировать отзывы (удалять/одобрять)
- рассматривать жалобы магазинов
- модерировать магазины и точки
- блокировать пользователей/магазины
- управлять категориями товаров

- видеть аналитику платформы
- управлять рекламой
- модерировать live-стримы

---

3. Основные сущности платформы

3.1 Merchant (магазин/бренд)

Основные поля:

- id
- owner_user_id
- name
- description
- payout_account_number
- payout_bank_name
- payout_status
- buy_eligibility (manual_contact | online_purchase)
- settings_json
- rating (float, агрегируется из ревью товаров)

3.2 Shop (филиал магазина)

Поля:

- id
- merchant_id

- name
- phone
- address
- lat/lon
- pos_status (inactive | active)
- working_hours (optional, future)

3.3 Seller Account

Привязан к Shop.

Поля:

- id
- user_id
- shop_id
- role: seller
- permissions: {manage_products, manage_posts, chat, livestream}

3.4 Product

Поля:

- id
- shop_id
- category_id
- name
- description

- gallery_json (список картинок)
- is_published
- average_rating (float)

3.5 ProductVariant

Поля:

- id
- product_id
- sku
- barcode
- price
- stockQty
- attributes_json (size, color…)

3.6 Post (публикация)

Аналог Instagram Reels / TikTok:

Поля:

- id
- shop_id
- product_id (nullable — привязка к товару)
- media_json (до 10 фото/видео)
- caption
- created_at

- likes_count
- views_count
- comments_count
- trending_score (для трендов)

3.7 Story (сторис)

Поля:

- id
- shop_id
- media (1 фото/видео)
- expires_at (24 часа)

3.8 Review (отзыв)

Поля:

- id
- user_id
- product_id
- rating (1–5)
- comment
- photo_json
- created_at
- status (active / removed_by_admin / under_review)

Отзыв создаётся **только после покупки**.

3.9 Complaint (жалоба магазина на отзыв)

Поля:

- id
- merchant_id
- review_id
- reason
- created_at
- status (pending / accepted / rejected)

3.10 Chat / Message

Одна структура для всех типов:

Chat:

- id
- type (user-user / user-shop / shop-user)
- participants_json

Message:

- id
- chat_id
- sender_id
- text
- attachments
- created_at

3.11 Order

Поля:

- id
- user_id
- shop_id
- total_amount
- status
- created_at

---

4. Основные экраны (Frontend navigation)

Глобальное меню (и веб, и мобильное):

- Лента
- Каталог
- Чат
- Корзина
- Профиль

(В мобильном — снизу; на веб — сверху.)

---

5. Поиск

Глобальный поиск на страницах:

- рекомендации (в ленте)
- чаты
- каталог

Ищет:

- магазины
- товары
- пользователей (для отправки в чат)

Результат включает:

- аватар
- имя
- username
- тип (user/shop)

📘 ТЕХНИЧЕСКОЕ ЗАДАНИЕ — FUCENT v3.0

ЧАСТЬ 2 / 5 — ЛЕНТА (REELS / TIKTOK), ПУБЛИКАЦИИ, СТОРИС, СТРИМЫ, РЕКОМЕНДАЦИИ, АЛГОРИТМ ТРЕНДОВ

---

6. Лента (Feed System)

Лента является ключевым элементом платформы и делится на **3 основные страницы**:

1) Подписки

Показываются **публикации только тех магазинов**, на которых пользователь подписан.

2) Тренды (Trends)

Работает как TikTok / YouTube Shorts — вертикальная лента, свайп вверх/вниз.

Показывает:

- самые популярные публикации за последние дни
- новые публикации хорошо-показывающих магазинов
- публикации с высоким engagement score

3) Рекомендации (Explore)

Полная аналогия Instagram Explore:

- много превью (сетка / вафелька)
- фото и видео
- теги категории товаров
- рекомендация на основе контента, похожего на интерес пользователя
- в этой вкладке есть **поиск**, который ищет магазины по нику и имени

---

7. Типы контента в ленте

7.1 Публикации (Posts)

Общие правила:

- создавать публикации могут **только магазины и продавцы** (не покупатели)
- публикация содержит **до 10 фото или видео** (карусель)
- отображается как:

- Reels-режим (полный экран)
- или как плитка в “Рекомендациях”

Виды публикации:

- **фото-пост**
- **видео-пост**
- **карусель** (свайп влево/вправо)
- **Reels-пост** (вертикальное видео)

7.2 Свайпы (UX логика)

Reels / TikTok формат:

- свайп вверх → следующее видео
- свайп вниз → предыдущее видео
- автоматическое проигрывание
- вертикальная ориентация

Карусель:

- в самой публикации свайп влево-вправо
- всегда одна медиа-единица на экране

7.3 Поддержка привязки публикаций к товару

Публикация может быть привязана к товару:

`post.product_id (nullable)`

Если привязка есть → в ленте появляется кнопка:

"Перейти к товару" → открывает карточку товара.

Если нет → кнопки нет.

---

8. Сторис (24 часа)

Функциональность аналогична Instagram:

- магазины и продавцы могут загружать сторис
- фото или видео до 15 секунд
- живёт 24 часа
- видно сверху ленты (аватарки магазинов)

Будет храниться таблица:

```
story: id, shop_id, media, created_at, expires_at
```

---

9. Прямые эфиры (Live Streams)

Магазины и продавцы могут запускать стрим.

Функции:

- запуск эфира магазином
- уведомления подписчикам
- список активных эфиров наверху ленты
- после завершения стрим не сохраняется (MVP)
- далее можно добавить:
    - сохранение записи

    - покупки прямо во время стрима

---

10. Алгоритм рекомендаций (Explore)

Алгоритм учитывает:

10.1 Engagement Metrics

- лайки
- сохранения
- комментарии
- время просмотра
- досмотры до конца
- свайпы “вверх” (быстрый skip снижает рейтинг)

10.2 Категории

Публикации показываются тем, кто интересовался товарами тех же категорий.

10.3 Геолокация

Преимущественно показываются магазины, которые находятся в том же районе города.

10.4 Новые магазины

Платформа повышает видимость новых магазинов (boost).

10.5 Личный интерес пользователя

Учитывается:

- на что он нажимает
- какие магазины открывает
- какие товары смотрит

- что ищет
- на кого подписан

---

11. Алгоритм Трендов (TikTok-модель)

Отдельный алгоритм.
Механика:

11.1 Расчёт рейтинга публикации

Каждые 10 минут обновляется поле:

```
post.trending_score
```

Формула (упрощённый вариант):

```
TrendingScore =
  (Likes * 2.0) +
  (Comments * 3.0) +
  (Saves * 5.0) +
  (Views * 0.1) +
  (ViewRetention * 4.0) +
  (CTR * 1.5)
  - (SkipRate * 2.0)
  + BoostNewShop
```

Где:

BoostNewShop = +10% для публикаций магазинов < 14 дней.

11.2 Распределение показов

Тренды — это:

1. Пул самых популярных публикаций последних **48 часов**
2. Пул новых публикаций с высоким potential score

3. Пул магазинов в твоём регионе

Выбор осуществляется случайно-взвешенным алгоритмом.

11.3 Порог входа в тренды

Пост попадает в тренды, если:

- engagement_rate ≥ 3%
- view_rate ≥ 60%
- skip_rate ≤ 40%

---

12. Поиск внутри вкладки "Рекомендации"

В вкладке "Рекомендации" есть собственный поиск.

Ищет:

- магазины (по нику, имени, категории)
- товары
- публикации

Результаты:

- сортируются по релевантности
- показываются плиткой

---

13. Лайки, сохранения, комментарии

Лайки:

- таблица likes (user_id, post_id)

Сохранения:

- таблица saves (user_id, post_id)

Комментарии:

- таблица comments (user_id, post_id, text, created_at)

Комментарии:

- открыты для всех
- но если покупатель реально купил товар из поста → появляется бейдж **"Verified Purchase"**

---

14. Шаринг публикации

У публикации есть кнопка:

"Поделиться"

При нажатии:

- открывается окно (30–70% высоты)
- пользователь может искать людей/магазины
- и отправлять ссылку в чат

Работает как Instagram.

---

15. UX-анимации (обязательные)

- плавный свайп между публикациями
- быстрое переключение при лёгком свайпе
- автозагрузка следующего видео заранее
- карусель с индикаторами (точками)
- видео играет только если публикация на экране

---

16. Ограничения и требования

16.1 Форматы медиа

- фото: JPG, PNG, WEBP
- видео: MP4, HEVC
- максимум: 10 штук в публикации
- максимальный размер файла: до 100MB

16.2 Скорость загрузки

Лента должна работать за **≤ 2 секунды**.

16.3 Бэкенд требования

- кэширование Reels (Redis)
- prefetch следующей публикации
- API должен возвращать:
    - media_urls
    - type (image/video)
    - aspect_ratio
    - product_id (если есть)

📘 ТЕХНИЧЕСКОЕ ЗАДАНИЕ — FUCENT v3.0

ЧАСТЬ 3 / 5 — КАТАЛОГ, ТОВАРЫ, ВАРИАНТЫ, КАТЕГОРИИ, КАРТА, МАРШРУТЫ, ОЦЕНКИ И ОТЗЫВЫ

Этот раздел описывает структуру каталога, правила публикации товаров, валидацию атрибутов, работу с остатками (POS / manual), экспорт/импорт, карту магазинов и систему отзывов/рейтингов.

---

17. Общие принципы каталога

- Вся торговая информация (товары, варианты, цены, остатки) принадлежит **магазинам (Shop)** и управляется владельцем Merchant / назначенными Seller`ами.
- Публикации (Feed posts) — отдельный контент-тип; публикация может быть **привязана** к товару (`post.product_id`), но товар и публикация — разные сущности.
- Для возможности онлайн-продажи (кнопка «Купить») **обязательно** иметь валидный `sku` или `barcode` (EAN/UPC) либо уникальный QR.
- Для MVP схему миграций можно не применять (hibernate `ddl-auto=update`), но в продакшне — обязательно Flyway/Liquibase.

---

18. Категории (Category)

Цель: централизованная иерархия категорий, похожая на Wildberries.

Основные требования:

- Дерево категорий `parent_id` → `id`.

- Мультиязычные поля: `name_ru`, `name_kg`, `name_en`, `name_zh`, `name_kz`, `name_uzb`.
- Поля: `id`, `parent_id`, `slug`, `names_json` (или отдельные колонки), `sort_order`.
- Управление категориями через админку (создание/редактирование/удаление).
- Поддержка атрибутов категории (обязательные атрибуты для одежды: size/color/gender/season).

---

19. Сущность продукта (Product) и валидация атрибутов

19.1 Product

Поля (ключевые):

- `id UUID`
- `shop_id FK→shop`
- `category_id FK→category`
- `title`
- `description`
- `is_published boolean`
- `search_tags` (array / json)
- `created_at`, `updated_at`
- `visibility` (public / private / archived)
- `metadata_json` (доп. данные)

19.2 ProductVariant

Каждый товар должен иметь хотя бы один вариант.

Поля:

- `id UUID`
- `product_id FK→product`
- `sku` (обязательный для online_purchase) — уникален в пределах shop / product (настраиваемо)
- `barcode` (EAN/UPC) — обязателен для онлайн-продажи
- `attributes_json` — пример: `{"size":"M","color":"red"}`
- `price` (decimal)
- `msrp` (optional)
- `stock_qty` (int)
- `is_active` (boolean)
- `images` (список)
- `weight`, `dimensions` (опционально для доставки)
- `created_at`, `updated_at`

Правила:

- Для `buy_eligibility = online_purchase` обязательны: `sku` и `barcode` (или QR/уникальный идентификатор).
- SKU/Barcode валифицируются при создании; если не совпадает формат EAN/UPC — отклоняется.
- Поддержка составных/bundled товаров: `bundle` — коллекция `variant_id` с количеством. Bundles реализуются как отдельный `Product` с флагом `is_bundle=true` и `bundle_items` в JSON.

---

20. Импорт / экспорт и API-интеграции

20.1 Массовая загрузка

- Поддержка CSV / XLSX / XML для массового импорта товаров и вариантов.
- Валидация — **dry-run**: перед импортом запускается проверка и возвращается отчёт ошибок (формат: строка, колонка, причина).
- Отчёт импорта доступен для скачивания (CSV/XLSX).
- Поддержка маппинга колонок (user maps columns → fields) и сохранение схемы импорта для повторного использования.

20.2 1C / ERP интеграция

- Планируется API для импорта/синхронизации (REST).
- Частота обновлений — задаётся в настройках; в MVP — ручной импорт.

---

21. Запасы, резерв и взаимодействие с POS

21.1 Модели наличия

- **Manual mode** (default) — продавец вручную обновляет остатки; покупка онлайн требует обращения к продавцу (кнопка «Связаться»).
- **POS mode (smart-pos)** — терминал посылает продажи/возвраты → платформа синхронизирует `pos_sale` → `product_variant.stock_qty` обновляется автоматически.
- **Hybrid** — магазин использует POS, но некоторые SKU выключены из онлайн-продажи (например, редкие/нестатичные товары). Это задаётся в `variant.is_active_online`.

21.2 Heartbeat и SLA

- POS присылает heartbeat (каждые 5 мин рекомендовано).
- Если `last_heartbeat` > 10 минут → магазин считается offline для онлайн-продажи: **кнопка «Купить» скрывается** на карточках товаров у этого

магазина.

- В панели продавца виден статус POS: `active`, `degraded`, `offline`.
- При `degraded` — включается предупреждение о возможной некорректности остатков (пользователь уведомлён).

21.3 Резервирование при заказе

- При создании заказа система делает **резерв** на `variant.stock_qty`:
  - Hold до оплаты — **30 минут** (резерв снимается, если не оплачен).
  - После оплаты — резерв действует **24 часа** для самовывоза (если выбран самовывоз), либо списывается при отгрузке.
- Авто-отмена заказа: scheduler проверяет `order.hold_until`, отменяет и возвращает `stock_qty` (уведомления продавцу/покупателю).

---

22. Поведение «Купить» и Buy Eligibility

- `buy_eligibility` для магазина/варианта влияет на UI:
  - `manual_contact` — **нет** кнопки «Купить», показываются CTA: чат, звонок, маршрут.
  - `online_purchase` — отображается кнопка «Купить» (если POS активен или магазин подтвердил наличие через POS/Payout on-boarding).
  - `hybrid` — для некоторых SKU может быть `is_active_online=false`.

---

23. Карта магазинов и маршруты

23.1 Карта

- В каталоге — кнопка **«Карта»**, открывающая карту со всеми магазинами (clustering при большом кол-ве).

- Для каждой точки: `name`, `rating`, `address`, `phone`, `open_now`, `has_pos` (да/нет).
- KPI по витрине на карте: просмотры точки, запрос построить маршрут.

23.2 Построение маршрута

- На карточке магазина возможность «Построить маршрут» → запускает внешнюю навигацию (Google Maps / 2GIS / встроенный маршрутизатор).
- В API/Frontend — кнопка генерирует deeplink вида `geo:lat,lon?q=name` или `https://www.google.com/maps/dir/?api=1&destination=lat,lon`.

---

24. Поиск и фильтры в каталоге

- Фильтры: категория, цена (min/max), наличие (в наличии / под заказ), рейтинг магазина, доставка (есть/нет), расстояние (по карте), бренд, скидки.
- Сортировка: релевантность, цена, рейтинг, свежесть, популярность.
- Поддержка A → B фильтров (например — размер + цвет + цена).
- Пагинация и cursor-based pagination для больших списков.

---

25. Резерв и конкурентные покупки

- При попытке купить товар, у которого осталось `stock_qty` меньше запрошенного — пользователь получает сообщение об остатке и предлагается оформить запрос наличия.
- Для POS-магазинов, при одновременных попытках покупки, транзакции должны обрабатываться в DB с `SELECT ... FOR UPDATE` или эквивалентом, чтобы избежать race conditions.

---

26. Рейтинги, отзывы и модерация

26.1 Бизнес-правила отзывов

- **Отзыв на товар** доступен **только** если пользователь действительно купил этот товар (проверяется по заказам).
- Отзыв содержит: `rating` (1–5 stars), `comment` (текст), `photos` (опционально), `created_at`.
- Отзыв привязан к `product_id` и к конкретному `order_id` (гарантия, что покупка была).
- Отзыв помечается `verified_purchase=true` если покупка подтверждена. Бейдж отображается на UI.

26.2 Агрегация рейтинга магазина

- `shop.rating = AVG( product_variant_reviews.rating )` — агрегируется при создании / обновлении отзывов.
- В карточке магазина показывается:
  - средняя оценка (1…5)
  - количество отзывов
  - последние N отзывов (с возможностью фильтрации по товару)

26.3 Жалобы магазина на отзывы

- Магазин подаёт `complaint` на конкретный `review_id` с описанием и доказательствами.
- Процесс:
  1. Магазин отправляет жалобу → статус `pending`.
  2. Админ видит тикет, просматривает доказательства (фото/заказы/лог транзакций).
  3. Админ решает: `accept` (удалить отзыв / пометить как мошеннический) или `reject` (оставить отзыв).

4. Если удалён — автору отзыва отправляется уведомление с объяснением.

- Логи действий и доказательства хранятся в `complaint_history`.

26.4 Модерация контента отзывов

- Автофильтр стоп-слов блокирует публикацию откровенно запрещённого контента.
- Пользователь может приложить фото как доказательство брака — фото доступны модератору.
- Админ имеет возможность восстановить удалённый отзыв (например, после проверки).

---

27. Витрина товара и карточка товара (UI)

Карточка товара должна показывать:

- название, цену, скидку, галерею (carousel)
- available_variants (размер/цвет)
- кнопка «Купить» (если `online_purchase` и POS active) или CTA «Уточнить наличие»
- рейтинг товара + общее количество отзывов
- блок «Похожие товары» (в той же категории)
- кнопка «Поделиться» (в чат / как ссылка)
- лог уровня доступности (последний heartbeat POS, последний синк остатков)

---

28. Экспорт отчётов для продавцов

- Продавец может выгрузить отчёты по товарам и кампаниям в CSV / XLSX:

    - остатки по SKU, продажи за период, возвраты, просмотры карточек, переходы в чат.

- Экспорт доступен из интерфейса Seller → Analytics → Export.

---

29. Защита от фрода и валидация

- Проверка целостности barcode/sku — reject на этапе импорта/создания.
- Защита от клик-фрода для рекламных кампаний — rate limits и анализ аномалий (в дальнейшем ML).
- Лог действий продавцов и админов (audit log) хранится и доступен для расследований.

---

30. Производительность и кэширование

- Кэширование карточек товаров и Reels meta — Redis.
- Tile-кэш карты (при использовании сторонних карт) — опционально (локальный кэш на CDN).
- Индексы в PostgreSQL: `product.sku`, `product_variant.sku`, `product_variant.barcode`, `shop.lat_lon` (GiST), `post.trending_score`.

---

31. Тестовые случаи (основные)

- Создание товара → обязательные поля sku/barcode при online_purchase.
- Создание bundle → проверка целостности элементов.
- Импорт CSV → dry-run с отчётом ошибок.
- POS sale → уменьшение `stock_qty` и появление записи в `pos_sale`.
- Order create → резервирование stock, автоснятие по таймауту.

- Review create → только после покупки; проверка verified_purchase badge.
- Complaint flow → merchant files complaint → admin deletes review.

---

32. UX-заметки для фронтенда (важно для реализации)

- При отображении карточки товара указывай текстовое сообщение, если магазин offline: “Онлайн-покупка временно недоступна — уточните наличие”.
- В каталоге при фильре по «Только онлайн» исключать магазины с `pos_status=offline` или SKU с `is_active_online=false`.
- Для магазинов с нескольких точками — на карте показывать отдельную точку для каждой `Shop`. Карточка магазина должна предлагать выбрать точку (если у магазина несколько).
- При шаринге товара в чат — отображать preview: фото, название, цену и ссылку.

📘 ТЕХНИЧЕСКОЕ ЗАДАНИЕ — FUCENT v3.0

ЧАСТЬ 4 / 5 — ЧАТЫ, ПОИСК, ШАРИНГ, ПОДПИСКИ, КОРЗИНА, ЗАКАЗЫ, ОПЛАТЫ, ДИАЛОГИ С МАГАЗИНОМ

Эта часть описывает всю социально-коммуникационную часть платформы:
 чат, подписки, рекомендации, прямые эфиры, отправку товаров/публикаций, корзину, покупки, workflow заказов, push-уведомления.

---

33. Чат (Messaging System)

33.1 Общая концепция

Чат — универсальный, единый для всех:

- пользователь ↔ пользователь
- пользователь ↔ магазин
- продавец/менеджер ↔ покупатель
- магазин ↔ магазин (опционально, фазой позже)

Одна модель `chat` + `participants` + `messages`.
Нет отдельных сущностей "чат магазина" и "чат пользователя".

Типы чатов:

- `private` — 1:1 между пользователями/магазином
- `group` — не MVP, можно добавить позже
- `support` — чат с админом / поддержкой

33.2 Структура БД (высокоуровневая)

Таблица `chat`:

- `id` (UUID)
- `type` (`private` / `support`)
- `created_at`
- `last_message_at`
- `is_archived`

Таблица `chat_participant`:

- `chat_id`
- `user_id` **или** `shop_id` (универсальный `subject_id + subject_type`)
- `role` (`owner`, `member`)
- `joined_at`

Таблица `message`:

- `id`
- `chat_id`
- `sender_type` (`user` / `shop`)
- `sender_id`
- `text` (nullable)
- `media_url` (nullable)
- `payload_json` (для шаринга товара/ленты)
- `created_at`
- `is_read`
- `read_at`

Таблица `message_attachment` (до 10 файлов как у Instagram Direct)

- `message_id`
- `url`
- `type` (`image`, `video`)
- порядок для carousel

---

34. Поиск внутри чата

Поиск похож на Instagram Direct:

Параметры поиска:

- по username
- по имени и фамилии
- по названию магазина
- поиск сразу показывает:
    - людей, на которых подписан
    - магазины (возможно с галкой “проверенный”)
    - недавние чаты
    - рекомендованных людей/магазины на основе активности

Поиск осуществляется по общему индексу:

- `users.username`
- `users.name`, `surname`
- `shop.name`
- `merchant.name`

Используется PostgreSQL full-text search + trigram search.

---

35. Шаринг товаров и публикаций (как Instagram Send)

35.1 Отправка товара

Пользователь нажимает кнопку “Поделиться” в карточке товара → всплывает **нижнее модальное окно 30–70%**:

- поиск пользователей/магазинов
- список избранных контактов
- последний чаты
- возможность отправить нескольким людям

Отправляется:

```
{
  "type": "product_share",
  "product_id": "...",
  "variant_id": null | "...",
  "title": "Nike Air Force",
  "image": "first_image_url",
  "price": 12990
}
```

35.2 Отправка публикации (ленты)

Аналогично, но тип:

```
{
  "type": "post_share",
  "post_id": "...",
  "preview": "...",
  "merchant_id": "...",
  "shop_id": "..."
}
```

В чате появляется **кликабельная карточка** с переходом к товару/публикации.

---

36. Подписки

36.1 Типы подписок

Пользователь может подписываться на:

- Магазины (основное)
- Продавцов (опционально)
- Других пользователей (фаза 2)

В MVP: подписки **только на магазины**.

Таблица `subscription`

- `follower_user_id`
- `shop_id`
- `created_at`

---

37. Лента (Feed): подписки, тренды, рекомендации

В ленте три вкладки:

37.1 «Подписки»

Тут показываются только публикации тех магазинов, на которые подписан пользователь.

Сортировка:

- сначала новые (по времени)
- затем отфильтровать скрытые, заблокированные магазины
- пагинация infinite scroll + swipe-наверх (как TikTok/Instagram Reels)

UI → одна публикация на весь экран, свайп вверх → следующая.

37.2 «Тренды» (как TikTok)

Публикации, которые получили:

- много просмотров
- много лайков
- репосты
- длительное удержание (watch_time)
- активные переходы на товар

Алгоритм (простая версия):

```
score = views * 0.1 + likes * 1 + comments * 1.5 + shares * 2 +
watch_time_sec * 0.05
```

Раз в 5 минут recalculation + обновление `post.trending_score`.

Публикации сортируются по `trending_score DESC`.

Механика — TikTok-style вертикальный свайп.

37.3 «Рекомендации» (как Instagram — сетка/вафелька)

Здесь многоплиточное отображение публикаций.

Презентация:

- маленькие плитки 1:1
- нажатие открывает публикацию в формате Reels
- сортировка по контентной релевантности, а не только по тренду

Факторы ранжирования:

- предпочтения пользователя (категории товаров, магазины)
- похожие пользователи
- взаимодействие с товарами/лентой
- персональный ML позже

---

38. Сторис и прямые эфиры магазинов

38.1 Сторис

Как в IG:

- магазин может загрузить 10 фото/видео
- хранятся 24 часа
- пользователь видит аватарки магазинов сверху

38.2 Прямой эфир

Вертикальная видеотрансляция:

- статус: `live`
- в ленте у магазина появляется красная подсветка «LIVE»
- можно перейти в чат и задавать вопросы

В MVP можно отложить live, оставить только сторис.

---

39. Корзина (Cart)

Корзина хранится:

- локально (если пользователь не авторизован)
- в БД (если авторизован)

Таблица `cart_item`

- `user_id`
- `variant_id`
- `qty`
- `added_at`

Правило: один товар → один магазин в заказе (как у Wildberries, Ozon).
Если корзина содержит товары разных магазинов → создаются **отдельные заказы по каждому магазину**.

---

40. Заказы (Orders)

40.1 Структура заказа

order

- `id`
- `user_id`
- `shop_id`
- `status` (`pending`, `awaiting_payment`, `paid`, `processing`, `ready`, `shipped`, `delivered`, `canceled`)
- `total_amount`
- `payment_method`

- `delivery_method`
- `address_json`
- `created_at`

order_item

- `order_id`
- `variant_id`
- `qty`
- `price`
- `subtotal`

order_payment

- `order_id`
- `transaction_id`
- `status`
- `provider_response_json`

40.2 Workflow заказа

1. **Сбор корзины**
2. **Создание заказа** (резерв на складе)
3. **Оплата**
4. Если оплата успешна → статус `paid`

5. Продавец подтверждает → `processing`
6. Выдача/отправка → `ready`/`shipped`
7. После получения → пользователь может оставить отзыв
8. Если время истекло → заказ автоматически `delivered`

---

41. Отзывы после покупки

Пользователь может оставить отзыв ТОЛЬКО после доставки.

Схема:

- оценка товара → на товар
- комментарий + фото → на товар
- рейтинг магазина = avg всех отзывов по товарам магазина

Магазин может пожаловаться → админ решает → отзыв остаётся/удаляется.

---

42. Уведомления (push + in-app)

Типы:

- новое сообщение в чате
- магазин ответил
- товар отправлен / готов к выдаче
- новый подписчик
- лайк на публикацию
- комментарий

- жалоба на отзыв решена

Push API → Firebase Cloud Messaging.

---

43. Приватность, блокировки, модерирование

- Пользователь может заблокировать магазин/пользователя
- Магазин может скрыть пользователя (если спам)
- Админ может блокировать магазин до разбирательства
- Жалобы на пользователей/магазины
- Любой контент (посты, сторис, отзывы) может быть удалён модератором

---

44. UX механики свайпа (как Instagram/TikTok)

Для ленты:

- один пост на весь экран
- свайп вверх → следующая
- свайп вниз → предыдущая

Для галереи в постах:

- до 10 фото/видео
- свайп влево/вправо

Для “вафельки” (рекомендации):

- grid 3×N
- открытие на полный экран

ЧАСТЬ 5 / 5 — АДМИНКА, РОЛИ, ДОСТУПЫ, МОДЕРАЦИЯ, АНАЛИТИКА, БЕЗОПАСНОСТЬ, ИНФРАСТРУКТУРА

Эта часть описывает административную панель, роли (покупатель, магазин, продавец, админ), модерацию контента, аналитику магазинов, безопасность, логирование, серверную архитектуру и DevOps-концепцию.

---

45. Роли и уровни доступа (RBAC)

45.1 Список ролей

Платформа поддерживает 4 роли:

1. Покупатель (CUSTOMER)

- CRUD на свой профиль
- просмотр ленты
- подписки на магазины
- создание заказов
- чаты
- оценки/отзывы после покупки

2. Магазин (MERCHANT)

Это “владелец бизнеса” (Nike, Adidas…).
Права:

- создание магазинов/точек (shop)
- управление продавцами (seller)

- управление товарами
- публикация лент
- просмотр заказов
- чаты от имени магазина
- аналитика магазина

3. Продавец (SELLER)

Работает на конкретной торговой точке (shop).
Права:

- управлять товарами только своего магазина
- отвечать в чатах
- подтверждать и выдавать заказы
- просматривать аналитику своей точки

4. Администратор (ADMIN)

Полный доступ:

- модерация публикаций
- модерация магазинов
- управление категориями
- управление пользователями
- просмотр всех заказов
- удаление отзывов / решение жалоб
- просмотр логов
- бан/разбан аккаунтов

---

46. Привязка продавцов к магазинам

46.1 Таблица `seller_assignment`

- seller_user_id
- shop_id
- role (`seller`, `manager`)
- created_at

Продавец работает строго в одном магазине (или нескольких — future).

---

47. Админ-панель (Admin Dashboard)

Admin Dashboard — это отдельное Angular-приложение или модуль внутри основного фронта.

47.1 Разделы админки

1. Пользователи

- список всех пользователей
- поиск по e-mail, имени, телефону
- бан/разбан
- просмотр профиля
- просмотры активности

2. Магазины

- заявки на создание магазинов
- проверка документов

- статус “pending → approved → suspended”
- данные магазина
- отчёты о мошенничестве
- удаление магазина

3. Публикации (лента / посты)

- жалобы
- отчёт по нарушению
- удаление контента

4. Товары

- управление категориями
- нарушение карточки товаров
- массовое удаление

5. Отзывы / Жалобы

- список жалоб магазинов на отзывы
- просмотр медиа-доказательств
- решение: удалить / оставить
- комментарий администратора

6. Аналитика платформы

- количество пользователей
- количество активных магазинов
- заказы
- конверсия

- популярные категории
- ошибки / баги / отчёты

7. Логи и аудит

- audit-log действий админов
- лог авторизаций
- лог ошибок
- мониторинг API

---

48. Модерация

48.1 Жалобы магазинов на отзывы

Workflow:

1. Магазин → “Пожаловаться на отзыв”
2. Заполняет описание
3. Админ видит жалобу
4. Решение:
    - удалить отзыв
    - оставить
    - запросить больше данных
5. Уведомление клиенту и магазину

48.2 Жалобы на публикации

- публикации с нарушениями правил

- публикации с мошенничеством
- недостоверные лоты
- админ может удалить и забанить автора

---

49. Аналитика и метрики

49.1 Метрики магазинов (для MERCHANT/SELLER)

Просмотры:

- просмотра товаров
- просмотра публикаций
- переходы на товар из ленты
- переходы из чата
- переходы из подписок

Продажи:

- количество заказов
- количество отмен
- средний чек
- конверсия магазина

Чаты:

- количество сообщений
- скорость ответа

Подписки:

- количество новых
- динамика роста

49.2 Метрики платформы (для ADMIN)

- DAU / WAU / MAU
- конверсия в покупку
- количество активных магазинов
- retention покупателей
- жалобы
- отчёты по магазинам
- тренды публикаций
- эффективность рекомендаций

---

50. Алгоритм рекомендаций (дополнение)

Посты в “Рекомендациях” ранжируются по:

- ER (лайки/комменты/шер/просмотры)
- CTR переходов в товар
- watch_time
- похожесть на любимые товары пользователя
- географическую близость магазинов
- “freshness boost” — бонус новым публикациям

Планируется ввод ML-модели — фаза 3.

---

51. Система логирования

51.1 Audit Log (обязателен)

Хранит:

- кто
- когда
- что сделал
- старое значение → новое значение
- IP-адрес

Логируются:

- создание/изменение магазинов
- блокировки
- удаление отзывов
- обновление товаров
- действия в админке

51.2 Application Log

Стандартные логи Spring Boot:

- INFO, WARN, ERROR
- ошибки API

- исключения

Сохраняются 90 дней.

---

52. Безопасность

52.1 Paranoid-модель

- JWT токены: 15–30 дней
- Optional 2FA для магазинов и админов
- Ограничение числа запросов (rate limit)
- Анти-DDoS на уровне NGINX
- Пароли: bcrypt

52.2 Фильтрация текстов

- анти-мат фильтр
- анти-спам
- лимиты на частоту сообщений

---

53. Сервер и DevOps

53.1 Архитектура backend

`Spring Boot → PostgreSQL → Minio (медиа) → Redis (кэш) → Firebase Push`

53.2 Развёртывание

MVP план:

- ручной деплой: `git pull` + `docker compose up`
- база — отдельный контейнер или отдельный сервер

Дальше:

- GitLab CI/CD
- Docker Registry
- Kubernetes (future)

53.3 Мониторинг (future)

- Prometheus
- Grafana
- Алерты по ошибкам выплат, API, POS-heartbeat

---

54. Хранение данных

- медиа (фото, видео, сторис, аватары, публикации) → Minio
- логи → 90 дней
- сырые события (просмотры, лайки) → 13 месяцев
- отзывы → навсегда
- чаты → не удаляем (по закону можно удалить по запросу пользователя)

---

55. Приватность и безопасность пользователей

- пользователь может скрыть профиль
- пользователь может удалить историю поиска
- пользователь может заблокировать магазин или другого клиента
- магазины не видят адреса покупателя до оплаты
- после завершения заказа адрес автоматически скрывается

---

56. Push-уведомления

Типы:

- новый заказ
- статус заказа изменился
- новое сообщение
- магазин ответил
- live у подписанного магазина
- акция магазина
- отзыв одобрен/удален админом

---

57. Страница профиля

57.1 Покупатель:

- аватар
- имя, фамилия
- username
- сохранённые товары
- сохранённые публикации
- настройки приватности
- чаты
- заказы
- подписки

57.2 Магазин:

- название
- описание
- логотип
- оценка (avg)
- список отзывов
- товары
- публикации
- аналитика
- продавцы