Дорожная карта архитектуры: Telegram-бот для цветочного магазина

Цель документа: единое живое руководство для разработки, тестирования и запуска Telegram-бота + дальнейшего перехода в Web App. Документ служит источником правды для задач, приоритетов, рисков и ответственных.

---

0. Исходные установки

- Платформа: Telegram (первичная). В будущем Web App с reuse API.
- Язык интерфейса: Русский, Узбекский (латиница).
- География старта: Бухара.
- Технологии: Python (ядро/бот), PostgreSQL (основная БД), Redis (кеш/rezervation), Storage: Telegram file_id + S3/MinIO для WebApp.
- Платёжные шлюзы: Click, Payme.
- Роли: Client, Florist, Courier, Owner, Manager/Support, Accountant (опционально), Marketing (опционально).
- Текущий статус: бот создан, есть Telegram bot API.

---

1. Уровни работ и фазы (milestones)

Фаза A — Подготовка и планирование (1-2 недели)

Цель: спроектировать архитектуру, ER-диаграмму, конечные API-энды и CI-пайплайн. **Доставляемое:** дорожная карта (этот документ), ER-диаграмма, UX-flows (3 сценария), список API-эндпойнтов, план миграции данных. **Задачи:** - Уточнить перечисление полей для каталогов/товаров/склада. - Определить формат хранения медиа (Telegram file_id vs S3). - Согласовать слоты доставки и временные зоны.

Фаза B — Базовая платформа и ядро (2-4 недели)

Цель: поднять backend-сервис и интеграцию с Telegram, настроить БД и Redis. **Доставляемое:** backend (API), подключённый Telegram bot, базовые CRUD для товаров/категорий, механика корзины и резервации. **Задачи:** - Настройка PostgreSQL, Redis, миграции. - User model + auth по tg_id, роли. - CRUD для categories/subcategories/products. - Cart + reservation в Redis (TTL 10–15 min). - Basic logging и audit.

Фаза C — Оформление заказа и канал флористов (2-3 недели)

Цель: обеспечить клиенту полный поток покупки и флористу — каналы приёма. **Доставляемое:** конструктор заказа (штучные, композиции, доп.товары), корзина, оформление (адрес/слоты), публикация заказа в канал флористов с inline-кнопками. **Задачи:** - Логика заказа: мультипозиционный, упаковка, открытка. - Публикация в закрытый канал заказов. - Принятие заказа флористом → транзакционное списание со склада.

Фаза D — Платежи и статусы (1-2 недели)

Цель: интегрировать Click/Payme, реализовать webhooks и статусы платежей. **Доставляемое:** онлайн-платежи, обработка webhook, обновление payment_status, обработка ошибок и отката. **Задачи:** - Подключение sandbox Click и Payme. - Обработка успешных/проваленных платежей. - Встраивание в юзер-флоу и UI статусов.

Фаза E — Склад и инвентаризация (2 недели)

Цель: автоматизация учета, админ-инструменты и пороги. **Доставляемое:** stock management, inventory log, threshold alerts, приемка товара (GUI для флориста), операции loss/correction. **Задачи:** - Разработка операций incoming/sale/loss/correction. - Внедрение пороговых уведомлений. - Импорт/экспорт остатков.

Фаза F — Канал магазина и маркетинг (1-2 недели)

Цель: автоматизация публикаций в публичный канал и deep-linking для быстрого заказа. **Доставляемое:** постинг готовых работ, шаблоны постов, кнопки "Заказать" с deep links. **Задачи:** - Формат captions и кнопок. - Генерация deep link для product_id.

Фаза G — Рекомендации и персонализация (2-4 недели)

Цель: внедрить простую рекомендательную логику и персональные подсказки. **Доставляемое:** rule-based recommendations + simple collaborative filtering prototype. **Задачи:** - Теги продуктов и rules engine. - «К клиенту подходят» рекомендации в карточке.

Фаза H — WebApp + масштабирование (4-8 недель)

Цель: поднять WebApp (React/Tailwind) с тем же API, dashboard для владельца. **Доставляемое:** WebApp, авторизация через Telegram, панель администратора. **Задачи:** - Адаптация медиа хранения, CDN. - Расширенные отчёты, графики.

---

2. Компоненты архитектуры (высокоуровнево)

1. **Telegram Bot Layer** — обработка update, команды, inline callbacks, deep links.
2. **Backend API (Python)** — бизнес-логика, авторизация, очереди задач (Celery/Task runner опционально).
3. **DB Layer (Postgres)** — данные каталога, заказы, клиенты, inventory_log.
4. **Cache/Reservation (Redis)** — cart reservations, rate limiting, locks.
5. **Media Storage** — Telegram file_id + S3/MinIO для WebApp backups.
6. **Payment Gateways** — Click, Payme (webhooks + confirmation flows).
7. **Notification System** — Telegram messages, scheduled reminders (APScheduler/Celery Beat).
8. **Channels & Groups** — публикация заказов, магазинный канал, группа доставщиков.
9. **Admin/WebApp** — панель управления каталогом, report export.
10. **Monitoring & Logging** — error tracking, audit logs.

---

3. Требования к API (high-level)

- **Auth**: идентификация по tg_id; role-based access token для WebApp.
- **Catalog API**: list categories, products, product detail, search & filter (by color/tag/event/VIP).
- **Cart API**: add/remove/list, reserve, confirm order.
- **Order API**: create, update status, assign florist/courier, attach media.
- **Payment API**: create payment, webhook handler, check status.
- **Inventory API**: view stock, adjust, import/export.
- **Channel API**: post to shop channel, post to florist channel (with actions).

---

4. Контроль качества и тестирование

- Unit tests для критичных бизнес-логик (pricing, stock reservation, order flow).
- Integration tests: Telegram update → backend → DB → publication in channel (emulated).
- End-to-end tests: заказ от клиента до отметки Delivered (sandbox payments).
- Manual QA checklists: UX, multilanguage checks (RU/UZ-latin), edge cases.

---

5. Развёртывание и окружение

- **Dev**: локально / Docker compose (Postgres, Redis, backend, ngrok для webhook). Owner тестирует.
- **Staging**: отдельный Telegram bot token + staging DB. CI-run tests.
- **Production**: managed Postgres, Redis, object storage, HTTPS endpoints, прод-секреты, мониторинг.
- **Backups**: nightly DB dump + media backup.

---

6. Риски и mitigations

- **Риск:** конфликт с несколькими флористами одновременно → **Mitigation:** transactional locks, first-accept wins.
- **Риск:** ошибка списания stock из-за человеческой ошибки → **Mitigation:** audit log, двухступенчатые large corrections, threshold alerts.
- **Риск:** платежные сбои → **Mitigation:** retry logic, ручная сверка платежей в админке.

---

7. Метрики успеха (KPI)

- Время от заказа до подтверждения флористом (целевой < 15 min).
- Процент успешных списаний stock без корректировок (> 98%).
- Уровень отказов в оплате (< 2%).
- NPS / feedback (через 7 дней после доставки).

---

8. План работы на ближайшие 14 дней (конкретные задачи)

День 1–3: - Уточнить поля для продуктов и склада. - Подготовить ER-диаграмму (deliverable). - Сверка с Owner по Telegram BOT token и test environment.

День 4–7: - Запуск окружения Dev (Postgres, Redis). Проверка webhook (ngrok). - Реализация user model + регистрация (tg_id) и roles. - CRUD для categories/products (админ-интерфейс minimal через бот или простая web-страница).

День 8–14: - Cart + reservation flow (Redis TTL). - Order creation → publish to florist channel with inline buttons. - Implement payment sandbox endpoints (Click/Payme) stubbed.

---

9. Owner actions (что нужно от тебя сейчас)

1. Предоставить доступ к Telegram bot token (sandbox/staging if available).
2. Список приоритетных товаров/3–5 примеров карточек продуктов (фото/цены) для импорта тестовых данных.
3. Указать контакт-флористов / группу для канала заказов (можно позже).

---

10. Формат контроля и коммуникации

- Вести задачи в трекере (Trello/Jira/GitHub Projects). Каждая задача — owner/assignee, время оценки, статус.
- Еженедельный статус-репорт (короткий): выполнено / в процессе / блокирует.
- Важные изменения — фиксировать в changelog внутри репозитория.

---

11. Следующие шаги (я готов выполнить)

- Подготовить ER-диаграмму и UX-flows (3 сценария) — выбрать приоритет.
- Сгенерировать детализованный список API эндпойнтов (без кода).
- Составить checklist для QA для первых e2e тестов.

Документ живой — предлагай правки, добавляй задачи, будем обновлять roadmap по мере продвижения.