MLM COMMUNITY

ROADMAP РАЗРАБОТКИ

Пошаговый порядок реализации проекта

v5.1 | 4 фазы | 24 шага | для Sonnet 4.6 / Opus

Часть комплекта:
1. MLM_TZ_v5.1.pdf — полное ТЗ (31 стр.)
2. MLM_SUPABASE.pdf — БД: 44 таблицы, 19 Edge Functions (16 стр.)
3. MLM_ADMIN.pdf — админка: 22 экрана, 4 роли (7 стр.)
4. MLM_PLATFORM.pdf — мультиплатформа (ожидается)
5. MLM_ROADMAP.pdf — ЭТОТ ДОКУМЕНТ

ДИАГНОСТИКА ТЕКУЩЕГО СОСТОЯНИЯ:

Готово (~12/51 экранов): Лендинг, ДНК-тест, Онбординг (6 экранов), Лента (посты, лайки, комментарии, stories), Компании (частично), Supabase Auth (email), Админка (8 разделов), SPA-роутер

Критические проблемы (7):
1. Схема БД (profiles) не соответствует ТЗ v5.1 (users, 44 таблицы)
2. Триггер handle_new_user сломан — костыль в коде
3. Кодировка файлов — UTF-8 mojibake (русский = мусор)
4. Нет атомарных операций (race conditions в лайках/XP)
5. Нет Edge Functions — бизнес-логика на клиенте (XP подделать через DevTools)
6. Auth только email — нет Telegram Mini App / OAuth
7. Нет RLS-политик — любой может читать/писать любую таблицу

ВЫВОД: Фронтенд-скелет (дизайн, навигация, UI) — хорошая база. Бэкенд (БД, auth, безопасность, бизнес-логика) — переделывать с нуля под v5.1.

ПРИНЦИП: ФУНДАМЕНТ → ЯДРО → ЭКРАНЫ → ПЛАТФОРМА

Сначала бэкенд (БД + безопасность + Edge Functions), потом адаптация существующего кода, потом новые экраны, потом мультиплатформа. Функционал → затем дизайн-полировка.

ОБЗОР: 4 ФАЗЫ, 24 ШАГА

Фаза	Шаги	Что делаем	Результат
0. Фундамент	1-4	БД, RLS, Edge Functions, кодировка	Безопасный бэкенд v5.1
1. Ядро	5-8	Рефакторинг существующего кода	Существующие экраны работают на новой БД
2. Экраны	9-18	Новый функционал (профиль, чат, сделки...)	Полный MVP (51 экран)
3. Платформа	19-24	PWA, Capacitor, Push, Админка v2	Готов к App Store / Google Play

ПРАВИЛО ЗАВИСИМОСТЕЙ: Каждый шаг зависит от предыдущих. Нельзя делать Шаг 9 (Профиль) до завершения Шагов 1-8. Нельзя делать Шаг 19 (PWA) до завершения всех экранов. Внутри фазы шаги можно делать параллельно, если нет прямой зависимости.

Общая структура проекта (целевая):

```
mlm-community/
  index.html              -- точка входа (SPA)
  admin.html              -- админка (отдельное приложение)
  manifest.json           -- PWA манифест
  sw.js                   -- Service Worker
  capacitor.config.json   -- iOS/Android конфиг
  vite.config.js
  package.json
  templates/              -- HTML-шаблоны экранов (~30 файлов)
  js/                     -- JavaScript модули (~40 файлов)
    core/                 -- ядро (router, auth, supabase, state)
    screens/              -- логика экранов
    utils/                -- утилиты (format, validate, animate)
    admin/                -- админка
  css/                    -- стили (~12 файлов)
  assets/                 -- иконки, изображения
  supabase/
    migrations/           -- SQL-миграции (CREATE TABLE ...)
    functions/            -- Edge Functions (Deno/TypeScript)
    seed.sql              -- тестовые данные
```

ФАЗА 0: ФУНДАМЕНТ (Бэкенд)

Цель: создать безопасную серверную инфраструктуру по ТЗ v5.1. Без этого нельзя писать фронтенд — он будет ложиться на несуществующие таблицы.

ШАГ 1 ПЕРЕСОЗДАТЬ СХЕМУ БД В SUPABASE

Удалить все старые таблицы (profiles, post_likes, post_comments, post_bookmarks, friends, applications, wisdom_cards, daily_quests, quest_templates, user_streaks, user_stats, user_settings, user_xp_log, dna_results, notifications). Создать 44 новых таблицы по MLM_SUPABASE.pdf.

Порядок создания (с учётом REFERENCES):

- Группа А (без зависимостей): platform_settings, xp_rates, admin_users, admin_sessions, admin_audit_log
- Группа Б (companies): companies → alliances → alliance_promotions
- Группа В (users): users → user_settings, user_stats, social_links, achievements, push_subscriptions
- Группа Г (контент): posts → comments, reactions
- Группа Д (эксперты): expert_cards → orders → order_responses
- Группа Е (магазин): products → purchases, reviews
- Группа Ж (чат): conversations → conversation_members, messages
- Группа З (форум): forum_topics → forum_replies, friends
- Группа И (сделки): deals → deal_milestones
- Группа К (финансы): transactions, referrals, subscriptions, withdrawals
- Группа Л (задания): ad_campaigns → tasks → task_completions
- Группа М (конкурсы): contests → contest_entries → contest_winners
- Группа Н (обучение): courses → lessons → user_courses → certificates
- Группа О (матчинг): matches, mentorships
- Группа П (вебинары): webinars → webinar_registrations
- Группа Р (система): notifications, strikes, reports, moderation_queue, verification_requests

Действие: Создать SQL-миграцию supabase/migrations/001_schema.sql. Выполнить в Supabase SQL Editor.

Проверка: SELECT count(*) FROM information_schema.tables WHERE table_schema='public' → должно быть 44.

ШАГ 2 НАСТРОИТЬ RLS-ПОЛИТИКИ

Включить RLS на КАЖДОЙ таблице (ALTER TABLE ... ENABLE ROW LEVEL SECURITY). Затем создать политики по MLM_SUPABASE.pdf, раздел RLS-политики.

Ключевые правила:

- users: SELECT всем, UPDATE только свой (auth.uid() = id)
- messages: SELECT/INSERT только участники conversation
- deals: SELECT только client_id и executor_id
- transactions: SELECT только свои
- admin-таблицы: БЕЗ RLS (доступ только через service_role из Edge Functions)

Действие: supabase/migrations/002_rls.sql. Выполнить.

Проверка: Попробовать с anon key прочитать чужие transactions → должен вернуть пустой массив.

ШАГ 3 НАПИСАТЬ КЛЮЧЕВЫЕ EDGE FUNCTIONS

Первая очередь — 6 функций, без которых нельзя запускать фронтенд:

#	Функция	Зачем	Приоритет
1	auth-telegram	Авторизация в Telegram Mini App (HMAC-SHA256 initData)	КРИТИЧНО
2	auth-email	Регистрация/вход через email (заменяет сломанный триггер)	КРИТИЧНО
3	complete-task	Начисление XP на сервере (не на клиенте!)	КРИТИЧНО
4	send-push	Уведомления: Telegram > FCM > Web Push	КРИТИЧНО
5	process-deal-payment	Оплата сделки: списание, эскроу, предоплата	ВАЖНО
6	purchase-product	Покупка в магазине: 80/20 split	ВАЖНО

Вторая очередь (CRON-функции, можно позже): update-streaks, weekly-report, process-referral-monthly, check-frozen-referrals, expire-subscriptions, return-chain, auto-accept-deal, draw-contest.

Третья очередь (v5.1 мультиплатформа): auth-telegram-oauth, register-push-sub.

Действие: Создать supabase/functions/auth-email/index.ts и т.д. Деплоить через supabase functions deploy.

ШАГ 4 ПОЧИНИТЬ КОДИРОВКУ + STORAGE BUCKETS

4а. Кодировка:

- Открыть index.html, supabase-api.js в редакторе → File → Save with Encoding → UTF-8 without BOM
- Проверить все файлы: file -i js/*.js templates/*.html → должно быть utf-8

4б. Создать 9 Storage Buckets в Supabase:

- avatars (public), task-screenshots (private), verification-docs (private), portfolio (public), products (private), product-covers (public), company-logos (public), post-media (public), chat-files (private)

4в. Настроить Storage Policies:

- avatars: любой authenticated может upload в свою папку (user_id/), public read
- chat-files: upload/read только участники conversation
- verification-docs: upload через Edge Function, read только admin

Проверка фазы 0: 44 таблицы, RLS на всех, 6+ Edge Functions работают, 9 buckets, кодировка UTF-8.

ФАЗА 1: ЯДРО (Рефакторинг существующего кода)

Цель: существующие экраны (лендинг, ДНК, онбординг, лента) работают на новой БД v5.1. Не добавляем новый функционал — только переключаем на правильную схему.

ШАГ 5 РЕФАКТОРИНГ СТРУКТУРЫ ПРОЕКТА

Реорганизовать файлы в целевую структуру. Перенести JS в подпапки:

```
js/
  core/
    router.js             -- SPA-навигация (существующий, доработать)
    state.js              -- НОВЫЙ: глобальное состояние (currentUser, settings)
    events.js             -- НОВЫЙ: шина событий (pub/sub)
  api/
    supabase-client.js    -- НОВЫЙ: только createClient + хелперы
    auth.js               -- рефакторинг auth-core.js + auth-ui.js
    users.js              -- НОВЫЙ: CRUD users, user_settings, user_stats
    posts.js              -- рефакторинг supabase-api.js (только посты)
    companies.js          -- рефакторинг supabase-api.js (только компании)
    tasks.js              -- НОВЫЙ: задания
    deals.js              -- НОВЫЙ: сделки
    chat.js               -- НОВЫЙ: чат API
    notifications.js      -- НОВЫЙ: уведомления
  screens/
    landing.js            -- существующий
    dna-test.js           -- существующий
    onboarding.js         -- существующий
    feed.js + feed-*.js   -- существующие
    profile.js            -- НОВЫЙ
    chat.js               -- НОВЫЙ
    ...                   -- новые экраны
  utils/
    format.js             -- форматирование дат, чисел, XP
    validate.js           -- валидация форм
    animate.js            -- анимации (из animations.js)
```

Важно: Обновить index.html — поменять пути скриптов. Проверить что всё загружается.

ШАГ 6 РЕФАКТОРИНГ AUTH (3 метода)

Заменить supabase-config.js + auth-core.js + auth-ui.js → js/api/auth.js:

- Telegram Mini App: window.Telegram.WebApp.initData → POST в auth-telegram Edge Function → JWT
- Telegram OAuth: redirect на oauth.telegram.org → callback → auth-telegram-oauth EF → JWT
- Email: форма email+пароль → auth-email Edge Function → Supabase Auth сессия

Автоопределение метода:

```
function detectPlatform() {
  if (window.Telegram?.WebApp?.initData) return 'telegram_mini_app';
  if (window.Capacitor?.isNativePlatform()) return 'native_app';
  return 'web';
}

// Для telegram_mini_app → автологин через initData
// Для native_app → Telegram OAuth или Email
// Для web → Telegram OAuth или Email
```

Удалить: костыль sbSignUpManual, sbInsertProfile. Профиль создаётся в Edge Function.

ШАГ 7 РЕФАКТОРИНГ SUPABASE API

Заменить supabase-api.js (406 строк, всё в одном файле) на модули по домену:

- js/api/posts.js: loadPosts, createPost, likePost (через rpc!), addComment, loadComments, toggleBookmark
- js/api/users.js: loadProfile, updateProfile, saveDnaResult, saveInterests, completeOnboarding
- js/api/companies.js: loadCompanies, loadCompanyDetail, sendApplication
- js/api/notifications.js: loadNotifications, markRead, subscribeRealtime

Критичные исправления:

- Лайки: заменить SELECT+UPDATE на sb.rpc('toggle_like', {post_id, user_id}) — атомарная операция
- XP: убрать sbAddXp с клиента. XP начисляется ТОЛЬКО через Edge Function complete-task
- Счётчики: заменить ручной increment на sb.rpc('increment_counter', {table, id, field})
- Таблицы: profiles → users, post_likes → reactions, post_comments → comments

ШАГ 8 АДАПТИРОВАТЬ СУЩЕСТВУЮЩИЕ ЭКРАНЫ

Пройтись по каждому работающему экрану и переключить на новые API:

Экран	Файлы	Что поменять
Лендинг	landing.js, landing.html	Регистрация → новый auth.js. Убрать mojibake текст.

Экран	Файлы	Что поменять
ДНК-тест	dna-test.js	sbSaveDnaResult → api/users.js. dna_results → users.dna_profile
Онбординг	onboarding.js	sbSaveProfileName/Interests/Goal → api/users.js. sbCompleteOnboarding → EF
Лента	feed.js, feed-data.js	sbLoadPosts → api/posts.js. Лайки через rpc. Новая схема comments.
Компании	companies.js	sbLoadCompanies → api/companies.js. Новые поля companies.

Проверка фазы 1: Лендинг → ДНК → Регистрация → Онбординг → Лента → всё работает на новой БД. XP начисляется через Edge Function. Auth через 2+ метода.

ФАЗА 2: ЭКРАНЫ (Новый функционал)

Цель: реализовать все оставшиеся ~39 экранов. Каждый шаг — один логический модуль. Для каждого: HTML-шаблон + JS-логика + CSS + обновление роутера.

ПАТТЕРН ДЛЯ КАЖДОГО НОВОГО ЭКРАНА:
1. Создать templates/[name].html (HTML-разметка)
2. Создать js/screens/[name].js (логика + init[Name]() + window.init[Name])
3. Создать/обновить css/[name].css (стили)
4. Обновить router.js: TEMPLATES + ensureTemplate + init-вызов
5. Обновить index.html: подключить JS и CSS
6. Лимиты: HTML <500, JS <500, CSS <800, функция <50 строк

ШАГ 9 ПРОФИЛЬ ПОЛЬЗОВАТЕЛЯ

Ключевой экран — показывает hero-карточку, шахматную фигуру с ДНК-цветом, XP-прогрессбар, достижения, статистику.

Экраны: scrProfile, scrProfileEdit, scrProfileSettings

Файлы:

- templates/profile.html, profile-edit.html, profile-settings.html
- js/screens/profile.js, profile-edit.js, profile-settings.js
- css/profile.css

API (js/api/users.js):

- loadUserProfile(userId) → users JOIN user_stats JOIN social_links JOIN achievements
- updateProfile(fields) → users.update
- uploadAvatar(file) → storage.avatars.upload

Данные из таблиц: users, user_stats, social_links, achievements, subscriptions

Зависит от: Шаги 1-8

ШАГ 10 ЧАТ И СООБЩЕНИЯ

Экраны: scrChatList, scrChat, scrChatInfo

Файлы:

- templates/chat-list.html, chat.html, chat-info.html
- js/screens/chat-list.js, chat.js, chat-info.js
- js/api/chat.js (loadConversations, loadMessages, sendMessage, createConversation)
- css/chat.css

Realtime: подписка на messages INSERT для текущего conversation_id

Данные: conversations, conversation_members, messages

Фичи: текст, файлы (chat-files bucket), reply_to, системные сообщения, unread_count

Зависит от: Шаг 9 (профиль — для отображения собеседника)

ШАГ 11 СДЕЛКИ (DEALS)

Экраны: scrDealCreate, scrDealDetail, scrDealList

- Создание: заказчик выбирает исполнителя, сумму, описание, дедлайн
- Оплата: process-deal-payment EF (50% предоплата + 50% эскроу + 20% комиссия)
- Статусы: pending → accepted → paid → in_progress → submitted → completed/revision/disputed
- Автоприёмка: auto-accept-deal EF (>72ч без ответа)
- Чат сделки: автоматически создаётся conversation (type='deal')

Данные: deals, deal_milestones, conversations, transactions

Зависит от: Шаг 10 (чат), Шаг 3 (process-deal-payment EF)

ШАГ 12 МАГАЗИН

Экраны: scrShop, scrProductDetail, scrProductCreate, scrMyProducts

- Каталог: фильтры по типу (guide/template/course/service/tool), dna_match, рейтинг
- Покупка: purchase-product EF (80% продавцу, 20% платформе)
- Отзывы: reviews (target_type='product')

Данные: products, purchases, reviews, transactions

ШАГ 13 ФОРУМ

Экраны: scrForum, scrForumTopic, scrForumCreate

- 7 категорий: business, marketing, tools, education, newbies, cases, offtopic
- Темы: forum_topics (pinned, solved, best_reply_id)
- Ответы: forum_replies (вложенные, лайки, is_best)

ШАГ 14 ЗАДАНИЯ v2 (3 типа)

Экраны: scrTasks, scrTaskDetail

- Платформенные: ежедневные квесты под ДНК-тип (заменяют старые daily_quests)
- Рекламные: от рекламодателей (ad_campaigns), оплачиваемые
- Бизнес: от компаний (reward_money в копейках)
- Проверка: скриншот → модератор (SLA 30мин) или auto-approve
- XP: через complete-task EF (не на клиенте!)

Данные: tasks, task_completions, ad_campaigns

ШАГ 15 КОНКУРСЫ (РОЗЫГРЫШИ)

Экраны: scrContests, scrContestDetail

- 3 категории: bronze/silver/gold (разные entry_fee)
- Еженедельно: draw-contest EF (CRON вс 20:00) — seed, генерация, начисление

Данные: contests, contest_entries, contest_winners

ШАГ 16 ЭКСПЕРТЫ И ЗАКАЗЫ

Экраны: scrExperts, scrExpertCard, scrExpertEdit, scrOrders, scrOrderDetail

- Карточка эксперта: specializations, portfolio, pricing, rating, skill_tree
- Заказы: клиент создаёт, эксперты откликаются

Данные: expert_cards, orders, order_responses

ШАГ 17 МАТЧ-СИСТЕМА + НАСТАВНИЧЕСТВО

Экраны: scrMatch, scrMatchResult, scrMentors

- Свайп-карточки: compatibility_score (ДНК + навыки + need/offer)
- like/pass/superlike → match → автосоздание conversation
- Менторство: mentor_id + mentee_id, оплата через platform (80/20)

Данные: matches, mentorships, conversations

ШАГ 18 ОБУЧЕНИЕ + ВЕБИНАРЫ + АЛЬЯНСЫ

Экраны: scrAcademy, scrCourse, scrLesson, scrWebinars, scrWebinarDetail, scrAlliances

- Курсы: modules → lessons (text/video/quiz/practice), progress tracking
- Сертификаты: certificate_hash для QR-верификации
- Вебинары: расписание, стрим, запись, рейтинг
- Альянсы: 2 компании объединяют ресурсы

Данные: courses, lessons, user_courses, certificates, webinars, webinar_registrations, alliances, alliance_promotions

Проверка фазы 2: Все 51 экран работают. Навигация по всем разделам. Данные сохраняются в Supabase. XP начисляется корректно.

ФАЗА 3: ПЛАТФОРМА (PWA + Native + Push + Админка)

ШАГ 19 PWA (Progressive Web App)

- manifest.json: name, icons (192px, 512px), theme_color=#8b5cf6, background_color=#06060b
- sw.js: кэширование статики (CSS, JS, HTML, шрифты), network-first для API
- Install prompt: beforeinstallprompt → красивая кнопка 'Установить'
- Offline mode: показывать кэшированную ленту, очередь запросов

ШАГ 20 CAPACITOR (iOS / Android)

- npm install @capacitor/core @capacitor/cli
- capacitor.config.json: appId='com.mlmcommunity.app', server.url
- iOS: npx cap add ios → Xcode → signing → App Store Connect
- Android: npx cap add android → Android Studio → signing → Google Play
- Плагины: @capacitor/push-notifications, @capacitor/browser, @capacitor/camera
- App Store: требование Apple — In-App Purchase для подписок (30% комиссия). Оставить Tribute для Telegram Mini App, IAP для iOS.

ШАГ 21 PUSH-УВЕДОМЛЕНИЯ (3 канала)

Канал 1: Telegram Bot API

- Для пользователей Telegram Mini App. Через Edge Function send-push.

Канал 2: Web Push (VAPID)

- Для PWA. Service Worker получает push, показывает notification.
- Подписка: register-push-sub EF сохраняет endpoint/p256dh/auth в push_subscriptions.

Канал 3: FCM Native

- Для Capacitor (iOS/Android). Firebase Cloud Messaging через @capacitor/push-notifications.
- Токен сохраняется в push_subscriptions (platform='android'/'ios', fcm_token).

Логика приоритетов (send-push EF):

- Обычное: один канал (TG > FCM > Web Push)
- Критичное (деньги, сделки): все каналы

ШАГ 22 CRON EDGE FUNCTIONS

Функция	Расписание	Что
update-streaks	Ежедневно 00:05	Streak ++/reset, бейджи
process-referral-monthly	1-е число	15% с подписок рефералов
check-frozen-referrals	Ежедневно	7-дн заморозка: разморозка/сгорание
expire-subscriptions	Ежедневно	Деактивация истёкших, даунгрейд, push
weekly-report	Вс 10:00	Статистика, формирование отчёта
return-chain	Ежедневно	Возвратные сообщения (7 ступеней)
auto-accept-deal	Каждый час	Автоприёмка >72ч
draw-contest	Вс 20:00	Seed, победители, начисление

ШАГ 23 АДМИНКА v2 (22 экрана)

Полная пересборка admin.html по MLM_ADMIN.pdf. 22 экрана, 4 роли (суперадмин, админ, модератор, аналитик). Отдельное веб-приложение admin.mlm-community.ru.

Порядок разработки (по приоритету):

- Критично (1 очередь): A1 Дашборд, A4 Верификация, A7 Проверка заданий
- Важно (2 очередь): A2-A3 Пользователи, A5 Модерация, A9 Выводы
- Нужно (3 очередь): A6 Задания, A8 Финансы, A20 Настройки, A21 Push
- Позже (4 очередь): A10-A19 (остальные), A22 Аналитика

ШАГ 24 ТЕСТИРОВАНИЕ + ДЕПЛОЙ

- Тестирование всех платформ: Telegram Mini App, Web PWA, iOS Simulator, Android Emulator
- Нагрузочное: 1000 одновременных пользователей (Supabase Pro plan)
- Безопасность: проверить все RLS (попытки обхода), Edge Functions (rate limiting)
- App Store: подготовить скриншоты, описание, privacy policy, review
- Google Play: аналогично + APK signing
- Деплой web: Vercel / Netlify (статика) + Supabase Cloud

ПРАВИЛА ДЛЯ РАЗРАБОТКИ (передать Sonnet 4.6)

Правило 1: Файловая дисциплина

Параметр	Лимит	Действие при превышении
HTML файл	< 500 строк	Разбить на компоненты
JS файл	< 500 строк	Вынести в отдельный модуль
CSS файл	< 800 строк	Разбить по секциям
Функция	< 50 строк	Декомпозиция
Файл в проекте	Конкретная задача	Один файл = один модуль

Правило 2: Именование

- Экраны: scr + PascalCase (scrFeed, scrChat, scrProfile)
- HTML: kebab-case (feed.html, chat-list.html)
- JS: kebab-case (feed.js, feed-data.js)
- Init-функции: init + PascalCase (initFeed, initChat)
- API-функции: действие + существительное (loadPosts, createDeal, toggleLike)
- CSS-классы: kebab-case с префиксом секции (feed-card, chat-bubble, profile-hero)

Правило 3: Безопасность

- НИКОГДА не начислять XP/деньги на клиенте. Только через Edge Functions.
- НИКОГДА не хранить секреты в JS (только Supabase Vault)
- ВСЕГДА проверять auth.uid() в RLS для пользовательских данных
- ВСЕГДА валидировать input на сервере (Edge Function), даже если валидация есть на клиенте
- ВСЕГДА использовать sb.rpc() для операций со счётчиками (атомарность)

Правило 4: Паттерн нового экрана

```
// 1. templates/chat.html
<div class='scr hidden' id='scrChat'>
  <div class='scr-head'>...</div>
  <div class='scr-body'>...</div>
</div>

// 2. js/screens/chat.js
function initChat() {
  // Инициализация после загрузки шаблона в DOM
  // Привязка обработчиков, загрузка данных
}
window.initChat = initChat;

// 3. router.js:
TEMPLATES.scrChat = '/templates/chat.html';
// В ensureTemplate:
if (id === 'scrChat') { if (window.initChat) window.initChat(); }

// 4. index.html:
<link rel='stylesheet' href='css/chat.css'>
<script src='js/screens/chat.js'></script>
```

Правило 5: Дизайн-система (НЕ МЕНЯТЬ)

- Шрифт: Outfit (300-900). Фон: #06060b. Primary: #8b5cf6
- Glass: rgba(255,255,255,0.035), border rgba(255,255,255,0.06)
- Радиусы: 14px карточки, 10px кнопки, 50% аватары
- Анимации: 200-400ms, cubic-bezier(0.16, 1, 0.3, 1)
- ДНК-цвета: Стратег #3b82f6, Коммуникатор #22c55e, Креатор #f59e0b, Аналитик #a78bfa
- Иконки: только SVG inline. НИКОГДА emoji в UI.
- Mobile-first: max-width 420px

Правило 6: Мультиплатформа

- Код ОДИН для всех платформ. Адаптация через detectPlatform()
- Telegram-специфичные API (MainButton, BackButton, HapticFeedback) — только если платформа = telegram
- Native-специфичные (Camera, PushNotifications) — только если Capacitor доступен
- CSS: safe-area-inset-* для iOS notch. env(safe-area-inset-top) и т.д.

Правило 7: Запрещено

- React, Vue, jQuery — только vanilla JS
- console.log в продакшене (только console.error в catch)
- Дублирование функций между файлами
- TODO/FIXME без номера задачи
- localStorage для пользовательских данных (только Supabase)
- Inline styles в HTML (только CSS-классы)
- Бэкап-файлы (.backup.*) в проекте

ЧЕКЛИСТ: ПЕРЕД КАЖДЫМ КОММИТОМ

Перед каждым изменением проверить:

[ ] Файл < лимита строк (HTML <500, JS <500, CSS <800)?
[ ] Функции < 50 строк?
[ ] Нет console.log (только console.error)?
[ ] Нет дублирования с другими файлами?
[ ] window.initXxx = initXxx; — экспорт есть?
[ ] router.js обновлён (TEMPLATES + ensureTemplate)?
[ ] index.html обновлён (подключение JS и CSS)?
[ ] Кодировка UTF-8 без BOM?
[ ] Нет inline styles?
[ ] Нет emoji в UI (только SVG)?
[ ] XP/деньги начисляются на сервере (не клиенте)?
[ ] Все данные идут через Supabase (не localStorage)?
[ ] RLS покрывает новую таблицу?
[ ] Работает на: Telegram Mini App, Web, мобильный?

СВОДНАЯ ТАБЛИЦА: ВСЕ 24 ШАГА

#	Шаг	Файлы	Зависит от	Приоритет
1	Схема БД (44 таблицы)	supabase/migrations/001_schema.sql	—	P0
2	RLS-политики	supabase/migrations/002_rls.sql	1	P0
3	Edge Functions (6 шт.)	supabase/functions/*	1,2	P0
4	Кодировка + Storage	все файлы + Supabase buckets	—	P0
5	Структура проекта	js/core/*, js/api/*, js/screens/*	4	P1
6	Рефакторинг Auth	js/api/auth.js	3,5	P1
7	Рефакторинг API	js/api/posts.js, users.js, ...	5,6	P1
8	Адаптация экранов	landing.js, feed.js, dna-test.js, ...	6,7	P1
9	Профиль	templates/profile.html, js/screens/profile.js	8	P2
10	Чат	templates/chat*.html, js/screens/chat*.js	9	P2
11	Сделки	templates/deal*.html, js/screens/deal*.js	10	P2
12	Магазин	templates/shop*.html, js/screens/shop*.js	8	P2
13	Форум	templates/forum*.html, js/screens/forum*.js	8	P2
14	Задания v2	templates/tasks*.html, js/screens/tasks*.js	8	P2
15	Конкурсы	templates/contest*.html	14	P2
16	Эксперты	templates/expert*.html	8	P2
17	Матч + Менторство	templates/match*.html	9,10	P2
18	Обучение + Вебинары	templates/academy*.html	8	P2
19	PWA	manifest.json, sw.js	все экраны	P3
20	Capacitor	capacitor.config.json, ios/, android/	19	P3
21	Push (3 канала)	js/core/push.js	20	P3
22	CRON Functions	supabase/functions/*	3	P3
23	Админка v2	admin.html, js/admin/*	все экраны	P3

#	Шаг	Файлы	Зависит от	Приоритет
24	Тест + Деплой	—	всё	P3

MLM Community | Roadmap разработки v5.1 FINAL | 4 фазы | 24 шага | 20 февраля 2026

Документ для передачи в Claude Sonnet 4.6 вместе с MLM_TZ_v5.1.pdf, MLM_SUPABASE.pdf и MLM_ADMIN.pdf. Выполнять строго по порядку: Фаза 0 → 1 → 2 → 3. Внутри фазы — по номерам шагов.