Updated Project Plan: LoyalNest App for Shopify

Objective

Develop a Shopify app delivering a customizable, user-friendly loyalty and rewards program to boost customer retention, repeat purchases, and brand loyalty, competing with Smile.io, Yotpo, and LoyaltyLion. Key differentiators include RFM segmentation in all plans, affordable pricing ($29/month for 500 orders), SMS-driven referrals via Klaviyo/Postscript, lightweight gamification, multilingual support, GDPR/CCPA compliance, and broad POS support. Deliver a production-grade TVP in 7–8 months, focusing on Must Have features (points, SMS/email referrals, basic RFM analytics, Shopify POS with offline mode, checkout extensions, GDPR request form, referral status with progress bar, notification templates with live preview, customer import, campaign discounts, rate limit monitoring with alerts), with Should Have and Could Have features phased in later. Use a microservices architecture with Docker for modularity, scalability, and independent deployments, leveraging AI-driven development (GitHub Copilot, Cursor, Grok) for efficiency. Handle UI/UX design and QA in-house to reduce costs, and incorporate enhancements like a freemium-to-Plus funnel, Shopify Flow integration, centralized logging, disaster recovery, and a merchant community to enhance adoption and reliability.

Phase 1: Research and Planning (4 Weeks)

Goals

- Understand Shopify’s app ecosystem and competitors’ gaps.
- Define USPs, prioritizing Must Have features for TVP, including Shopify Plus requirements.
- Establish technical and business requirements for a scalable microservices-based monorepo.

Tasks

- **Market Research**:
  - Analyze competitors (Smile.io, Yotpo, LoyaltyLion, BON Loyalty, Rivo, Gameball) for features, pricing, and reviews.
  - Target small (100–1,000 customers, AOV $20–$50), medium-sized (1,000–10,000 customers, AOV $50–$200), and Shopify Plus merchants (10,000+ customers, multi-store setups).
  - Identify gaps: SMS referrals (Smile.io/Yotpo weakness), affordable RFM analytics (BON/Gameball limitation), non-Shopify POS (Rivo/Gameball gap), lightweight gamification (Smile.io/BON gap), multilingual support (Smile.io/Rivo limitation), checkout extensions (LoyaltyLion strength).

- Validate demand for Must Have features: points (purchases, signups, reviews, birthdays), SMS/email referrals, basic RFM analytics, Shopify POS with offline mode, checkout extensions, automated loyalty email flows, data import from Smile.io/LoyaltyLion, GDPR request form, referral status display with progress bar, notification templates with live preview, campaign discounts with RFM conditions, rate limit monitoring with alerts.

- **Technical Requirements**:
  - Restructure monorepo into microservices (auth, points, referrals, rfm_analytics, event_tracking, admin) using Nx for monorepo management, NestJS (TypeScript) for APIs, and Rust/Wasm for Shopify Functions.
  - Use Shopify's Polaris and App Bridge for UI consistency in merchant dashboard, customer widget, and admin module.
  - Ensure Shopify and Shopify Plus compatibility (e.g., higher API rate limits, Checkout Extensibility APIs, Theme App Extensions).
  - Adopt API-first microservices architecture:
    - **Auth Service**: Handles Shopify OAuth, JWT, RBAC for admin module.
    - **Points Service**: Manages points earning/redemption, Shopify POS with offline mode, checkout extensions, campaign discounts.
    - **Referrals Service**: Handles SMS/email referrals via Klaviyo/Postscript, referral tracking, referral status with progress bar, merchant referrals.
    - **RFM Analytics Service**: Provides basic RFM analytics, segment previews, stored in `customers.rfm_score` (JSONB).
    - **Event Tracking Service**: Manages PostHog event tracking (e.g., `points_earned`, `referral_clicked`).
    - **Admin Service**: Manages merchant accounts, logs, RBAC, GDPR retention tracking, rate limit monitoring, notification templates, customer import.
  - Implement API versioning (`/v1/api/*`) for backward compatibility, using REST for UI-facing endpoints and gRPC for inter-service communication (e.g., RFMAnalyticsService, AdminService) with circuit breakers (`node-circuitbreaker`).
  - Use Redis Cluster for caching points, referrals, RFM scores, webhook idempotency keys, rate limits, and Shopify API rate limit tracking (`shopify_api_rate_limit:{merchant_id}`).
  - Use Kafka for event-driven architecture (`points.earned`, `referral.created`, `customer.imported`) to decouple services.
  - Consolidate database to PostgreSQL with JSONB for RFM/tier configs (e.g., `program_settings.rfm_thresholds`, `vip_tiers.rfm_criteria`, `email_templates.body`) and range partitioning on `created_at` for `points_transactions`, `referrals`, `reward_redemptions`.
  - Apply `loyalnest_full_schema.sql` with indexes on `customers(email, merchant_id, rfm_score)`, `points_transactions(customer_id)`, `referrals(merchant_id, referral_link_id, merchant_referral_id)`, `vip_tiers(merchant_id)`, `reward_redemptions(campaign_id)`, `gdpr_requests(retention_expires_at)`.
  - Implement incremental refresh for `rfm_segment_counts` materialized view using a delta table (`rfm_segment_deltas`).
  - Deploy on a VPS (Ubuntu with Docker) using Docker Compose for microservices and GitHub Actions for CI/CD with change detection.

    - Implement disaster recovery with `pg_dump` for PostgreSQL, Redis snapshotting, and Backblaze B2 for offsite backups.
    - Leverage AI (GitHub Copilot, Cursor, Grok) for NestJS APIs, Rust Functions, React components, Jest/Cypress tests, PostgreSQL index recommendations, VPS setup scripts, and backup scripts, with manual review.
- **USPs**:
    - Free plan: 300 orders/month, basic RFM analytics (churn risk segments), 50 SMS referrals/month, undercutting Smile.io ($49/month).
    - SMS-driven referrals via Klaviyo/Postscript post-purchase popups, surpassing Smile.io/Yotpo's email-only referrals.
    - Affordable $29/month plan with full RFM configuration, campaign discounts, checkout extensions, rate limit monitoring, competing with LoyaltyLion's $399/month.
- **Must Have Features for TVP**: Points (purchases, signups, reviews, birthdays), SMS/email referrals, basic RFM analytics (static thresholds), Shopify POS (offline mode), checkout extensions (points redemption), automated loyalty email flows, data import from Smile.io/LoyaltyLion, GDPR request form, referral status display with progress bar, notification templates with live preview, campaign discounts with RFM conditions, rate limit monitoring with alerts.
- **Should Have Features (Phase 4)**: VIP tiers (spending-based), advanced RFM configuration, exit-intent popups (referral-focused), Klaviyo/Mailchimp integration, behavioral segmentation (cart abandoners, frequent browsers), multi-store point sharing, Shopify Flow templates.
- **Could Have Features (Phase 6)**: Gamification (badges, leaderboards), multilingual widget (10+ languages: Spanish, French, German, etc.), multi-currency discounts, non-Shopify POS (Square, Lightspeed), advanced analytics (25+ reports), developer toolkit for Shopify metafields, AI-powered reward suggestions.
- **Team Formation**:
    - Solo developer handling NestJS, React, Rust, in-house UI/UX design, and QA, using AI tools (GitHub Copilot, Cursor, Grok) for code, tests, Polaris-compliant mockups, and VPS setup.
    - Prioritize Must Have features to manage workload; use AI for rapid prototyping, testing, and documentation.
    - Engage Shopify Partners program for feedback from 5–10 merchants (including 2–3 Shopify Plus) on TVP features.
- **Budget and Timeline**:
    - Development: $74,750 (TVP, AI tools, microservices setup, Shopify Plus features, in-house UI/UX and QA, backups, community).
    - Marketing: $3,000 (website, Shopify community ads, social media).
    - Support Infrastructure: $4,000 (VPS hosting, PostHog, Loki, Backblaze, support tools).
    - Contingency (15%): $11,362.50.
    - Total: $91,912.50.
    - Timeline: 4 weeks.

- **Enhancements & Best Practices**:
  - Document API contracts using OpenAPI/Swagger for REST endpoints (`/v1/api/*`).
  - Implement field-level encryption for PII (e.g., `customers.email` , `rfm_score`) using `pgcrypto` with quarterly key rotation via AWS KMS.
  - Implement GDPR webhook handlers (`customers/data_request` , `customers/redact`) with retry logic using Redis-based dead-letter queue and `retention_expires_at` tracking.
  - Set up CI pipeline (GitHub Actions) with change detection for microservices and Lighthouse CI for accessibility/performance.
  - Pin Shopify API versions (e.g., 2025-01) and monitor changelog RSS feed for deprecations.
  - Use centralized logging with Loki + Grafana, tagging logs with `shop_domain` , `merchant_id` , `service_name` .
  - Set up Slack community for beta merchant feedback and organic referrals.
- **Additional Tasks**:
  - Develop notification system with Klaviyo/Postscript integration, multilingual support (JSONB `email_templates.body`).
  - Design GDPR request form for customer widget (request data, redact options, tied to `gdpr_requests`).
  - Implement referral status display with progress bar (`referral_link_id` , status: `pending` / `completed` / `expired`).
  - Add `referral_link_id` and `merchant_referral_id` to `referrals` table, `campaign_id` to `reward_redemptions` .
  - Define GDPR retention tracking with `retention_expires_at` (90 days) in `gdpr_requests` .
  - Design merchant referral program (`/v1/api/referrals/merchant`).
  - Develop `dev.sh` script to start Docker containers, seed mock data with Faker, and simulate RFM scores.
- **Deliverables**:
  - Competitive analysis report highlighting gaps in SMS referrals, RFM analytics, POS support, checkout extensions, GDPR compliance.
  - Feature list prioritizing Must Have for TVP, with Should Have and Could Have planned for Phases 4–6.
  - Pricing model: Free (300 orders, basic RFM, 50 SMS referrals), $29/month (500 orders, full RFM, checkout extensions, campaign discounts), $99/month (1,500 orders, multi-store), Enterprise (custom).
  - Technical architecture diagram (microservices: NestJS, Rust, PostgreSQL, Redis Cluster, Kafka, Loki, VPS).
  - Solo developer plan with AI-assisted UI/UX, QA, and community setup.

Phase 2: Design and Prototyping (6.5 Weeks)

Goals

- Create intuitive, Polaris-compliant UI for merchants and customers, focusing on Must Have features (checkout extensions, GDPR request form, referral status with progress bar, RFM segment preview, notification templates with live preview, rate limit monitoring, usage thresholds, upgrade nudges).
- Design scalable microservices architecture for 5,000+ merchants and 100,000+ users (50,000+ for Plus).

Tasks

- **UI/UX Design (In-House)**:
  - Develop wireframes/prototypes using Polaris and App Bridge with AI assistance (Grok, Figma plugins) for:
    - Merchant dashboard: Points, referrals (SMS/email), basic RFM analytics, checkout extension config, notification templates with live preview, rate limit monitoring, settings, usage thresholds (e.g., SMS referral limit), upgrade nudges (e.g., Polaris `Banner` for $29/month plan).
    - Customer widget: Points balance, redemption, SMS/email referral popup, GDPR request form, referral status with progress bar, RFM nudges.
    - Admin module: Merchant management, RFM segment views, logs, rate limits, multi-user roles (Shopify Plus).
  - Design no-code RFM setup wizard with static thresholds (e.g., Recency 5: <30 days, Frequency 5: 5+ orders, Monetary 5: $250+ for AOV $50) and segment preview (Chart.js bar chart).
  - Include Must Have on-site content: SEO-friendly loyalty page, rewards panel, launcher button, points display on product/checkout pages, post-purchase/email capture popups, GDPR request form, referral status with progress bar.
  - Design notification template configuration UI with live preview for points earned, referrals, GDPR requests (multilingual, JSONB).
  - Design gamified onboarding checklist (e.g., “Complete RFM setup: +100 points”) for dashboard.
  - Plan Should Have UI elements (Phase 4): VIP tier display, exit-intent popups, discount banners, Theme App Extensions.
  - Use AI (Grok, Figma plugins) to generate Polaris-compliant mockups and validate manually for Shopify UX guidelines and accessibility (ARIA, keyboard nav).
- **Technical Architecture**:
  - Tech stack: Vite + React with Polaris, Tailwind CSS, App Bridge; NestJS for microservices (auth, points, referrals, rfm_analytics, event_tracking, admin); Rust/Wasm for Shopify Functions; PostgreSQL (JSONB, range partitioning); Redis Cluster; Kafka; Loki + Grafana.
  - Microservices:
    - **Auth Service**: `/v1/api/auth` (Shopify OAuth, JWT, RBAC).

    - **Points Service**: `/v1/api/points` (earn/redeem, POS, checkout extensions, campaign discounts).
    - **Referrals Service**: `/v1/api/referral` (SMS/email, referral status, merchant referrals).
    - **RFM Analytics Service**: `/v1/api/rfm/segments` (basic RFM, segment preview), `/v1/api/rfm/config` (Phase 4).
    - **Event Tracking Service**: `/v1/api/events` (PostHog integration).
    - **Admin Service**: `/admin/*` (merchant management, RBAC, GDPR, rate limits, notification templates, customer import).
  - Schema: Use `loyalnest_full_schema.sql` with `customers.rfm_score` (JSONB), `program_settings.rfm_thresholds` (JSONB), `merchants.staff_roles` (JSONB for RBAC), `email_templates.body` (JSONB), `rfm_segment_counts` materialized view (incremental refresh via `rfm_segment_deltas`), `gdpr_requests(retention_expires_at)`, `referrals(referral_link_id, merchant_referral_id)`, `reward_redemptions(campaign_id)`.
  - Shopify integration: OAuth via `@shopify/shopify-app-express`, `orders/create` webhook (batched processing) for points and RFM updates, POS with offline mode (SQLite queue), Checkout UI Extensions with Checkout Extensibility APIs.
  - Use Docker Compose for local and production microservices orchestration.
  - Implement async CSV import system for customer data with validation (`email`, `shopify_customer_id`) and GDPR compliance (AES-256 encryption via `pgcrypto`).
  - Plan disaster recovery with `pg_dump`, Redis snapshotting, and Backblaze B2 backups.
  - Use AI for NestJS microservice boilerplate, React components, Jest/Cypress tests, PostgreSQL index optimization, and `dev.sh` script.
- **Feature Prioritization**:
  - TVP (Must Have): Points, SMS/email referrals, basic RFM analytics, Shopify POS (offline mode), checkout extensions, automated email flows, data import, GDPR request form, referral status with progress bar, notification templates with live preview, campaign discounts, rate limit monitoring with alerts, usage thresholds, upgrade nudges.
  - Plan Should Have for Phase 4: VIP tiers, advanced RFM configuration, exit-intent popups, Klaviyo/Mailchimp integration, behavioral segmentation, multi-store point sharing, Shopify Flow templates.
  - Plan Could Have for Phase 6: Gamification, multilingual widget, multi-currency discounts, non-Shopify POS, advanced analytics, developer toolkit, AI-powered reward suggestions.
- **Prototyping**:
  - Build clickable prototype (Figma) for dashboard, widget, admin module, checkout extensions, GDPR request form, referral status with progress bar, RFM segment preview, notification templates with live preview, rate limit monitoring, usage thresholds, upgrade nudges.
  - Validate with 5–10 Shopify merchants (including 2–3 Shopify Plus) via Shopify Partners program, testing RFM usability, SMS referral popup effectiveness, POS integration, checkout extension adoption, GDPR form, referral status engagement, notification template usability, rate limit monitoring, upgrade funnel effectiveness.
- **Enhancements & Best Practices**:

    - Integrate PostHog for feature usage tracking (e.g., `rfm_wizard_completed`, `gdpr_request_submitted`, `referral_progress_viewed`, `notification_template_edited`, `rate_limit_viewed`, `plan_limit_warning`).
    - Include tooltips, gamified onboarding checklist, and upgrade nudges in wireframes.
    - Document Docker Compose, Nginx, and `dev.sh` configs for VPS setup.
    - Conduct early user testing on RFM wizard, SMS referral popup, GDPR form, referral status, notification templates, rate limit monitoring, checkout extensions, usage thresholds.
- **Additional Tasks**:
    - Implement `rfm_segment_counts` materialized view (PostgreSQL, incremental refresh via `rfm_segment_deltas`, daily at `0 1 * * *`).
    - Design RFM segment preview UI and API (`/v1/api/rfm/segments/preview`).
    - Develop notification template configuration UI and API (`/v1/api/notifications/template`) with live preview and multilingual support.
    - Build rate limit monitoring UI and API (`/v1/api/rate-limits`) with alerts (Slack/email) at 80% Shopify API limit.
    - Design usage thresholds UI (e.g., SMS referral limit progress bar) and API (`/v1/api/plan/usage`).
- **Deliverables**:
    - Wireframes/mockups for dashboard, widget, RFM analytics, admin module, checkout extensions, GDPR form, referral status with progress bar, notification templates, rate limit monitoring, usage thresholds, upgrade nudges.
    - Microservices architecture diagram (NestJS, Rust, PostgreSQL, Redis Cluster, Kafka, Loki, VPS).
    - Clickable prototype (Figma) with Must Have features.
    - Merchant feedback report on RFM, SMS referrals, POS, checkout extensions, GDPR form, referral status, notification templates, rate limits, upgrade funnel.

Phase 3: Development (19 Weeks)

Goals

- Build production-grade TVP with Must Have features across microservices, including Shopify Plus checkout extensions, GDPR request form, referral status with progress bar, notification templates with live preview, customer import, campaign discounts, rate limit monitoring with alerts, usage thresholds, upgrade nudges.
- Develop admin module with RBAC for multi-user support, campaign management, and rate limit monitoring.
- Ensure Shopify compliance and performance for 5,000+ customers (50,000+ for Plus).

Tasks

- **Backend Development (Microservices)**:
  - **Auth Service (NestJS)**:
    - APIs: `/v1/api/auth/login`, `/v1/api/auth/refresh`, `/v1/api/auth/roles`.
    - Shopify: OAuth via `@shopify/shopify-app-express`, JWT (15-minute expiry) for admin module RBAC (roles: `admin:full`, `admin:analytics`, `admin:support`).
    - Cache tokens in Redis Cluster.
    - Implement circuit breakers for gRPC calls to RFM Analytics and Admin services.
  - **Points Service (NestJS)**:
    - APIs: `/v1/api/points/earn`, `/v1/api/points/redeem`, `/v1/api/points/adjust`, `/v1/api/rewards` (campaign discounts).
    - Shopify: `orders/create` webhook (batched) for points (1 point/$), POS with offline mode (SQLite queue, sync via `/v1/api/points/sync`), Checkout UI Extensions with Checkout Extensibility APIs.
    - Implement campaign discounts with RFM conditions (e.g., `bonus_campaigns.conditions` JSONB: `{"rfm_score": {"recency": ">=4"}}`).
    - Add `campaign_id` to `reward_redemptions` table.
    - Cache points balances and campaign discounts in Redis Cluster.
    - Publish events (`points.earned`) to Kafka.
  - **Referrals Service (NestJS)**:
    - APIs: `/v1/api/referrals/create`, `/v1/api/referrals/complete`, `/v1/api/referrals/status`, `/v1/api/referrals/progress`, `/v1/api/referrals/merchant`.
    - Klaviyo/Postscript: SMS/email referrals, merchant referrals, storing codes in `referrals` table (`referral_link_id`, `merchant_referral_id`) via Bull queues.
    - Cache referral codes and statuses in Redis Cluster.
    - Publish events (`referral.created`, `merchant.referral.created`) to Kafka.
  - **RFM Analytics Service (NestJS)**:
    - APIs: `/v1/api/rfm/segments` (basic RFM, churn risk), `/v1/api/rfm/segments/preview`.
    - gRPC: `/analytics.v1/RFMAnalyticsService/GetSegments`, `/analytics.v1/RFMAnalyticsService/PreviewRFMSegments`.

    - RFM: Static calculations (e.g., Recency 5: <30 days, Frequency 5: 5+ orders, Monetary 5: $250+), stored in `customers.rfm_score` (JSONB).
    - Use `rfm_segment_counts` materialized view with incremental refresh via `rfm_segment_deltas`.
- **Event Tracking Service (NestJS)**:
    - APIs: `/v1/api/events` (PostHog integration).
    - Track events: `points_earned`, `referral_clicked`, `rfm_preview_viewed`, `gdpr_request_submitted`, `referral_progress_viewed`, `notification_template_edited`, `campaign_discount_redeemed`, `rate_limit_viewed`, `plan_limit_warning`, `campaign_rfm_conversion`.
    - Publish events to Kafka for async ingestion.
- **Admin Service (NestJS)**:
    - APIs: `/admin/merchants`, `/admin/points/adjust`, `/admin/referrals`, `/admin/rfm-segments`, `/admin/logs`, `/admin/notifications/template`, `/admin/rate-limits`, `/admin/customers/import`, `/v1/api/plan/usage` with JWT and RBAC.
    - gRPC: `/admin.v1/AdminService/UpdateNotificationTemplate`, `/admin.v1/AdminService/GetRateLimits`, `/admin.v1/AdminService/ImportCustomers`.
    - GDPR: Implement `/webhooks/customers/data_request`, `/webhooks/customers/redact` with retry logic (Redis dead-letter queue) and `retention_expires_at` tracking.
    - Build async CSV import for customers with validation (`email`, `shopify_customer_id`) and GDPR compliance (AES-256 encryption).
    - Cache logs, merchant data, rate limits, and notification templates in Redis Cluster.
    - Implement rate limit alerts (Slack/email) at 80% Shopify API limit and centralized rate limit tracking (`shopify_api_rate_limit:{merchant_id}`).
- **Inter-Service Communication**: Use gRPC with circuit breakers for RFM Analytics and Admin services; Kafka for async events (`points.earned`, `referral.created`, `customer.imported`).
- **Rust/Wasm**:
    - Shopify Functions: Discount logic, checkout extensions, basic RFM score updates, campaign discounts with RFM conditions.
    - Use Shopify CLI for testing; generate Rust code and tests with AI.
- **Disaster Recovery**:
    - Implement `pg_dump` for PostgreSQL and Redis snapshotting, storing backups in Backblaze B2.
- Use AI for microservice boilerplate, error handlers, Jest tests, and `dev.sh` script; manually review for Shopify compliance.
- **Frontend Development**:

- Build dashboard with Vite + React, Polaris, Tailwind CSS, App Bridge:
  - `WelcomePage.tsx` : Gamified onboarding checklist, RFM guide, upgrade nudges.
  - `PointsPage.tsx` : Configure purchases, signups, reviews, birthdays, campaign discounts.
  - `ReferralsPage.tsx` : SMS/email config, referral status, merchant referrals.
  - `AnalyticsPage.tsx` : Basic RFM segments, segment preview (Chart.js bar chart).
  - `SettingsPage.tsx` : Store, billing, rewards panel, checkout extension customization, notification templates with live preview, rate limit monitoring, usage thresholds (e.g., SMS referral limit).
- Customer widget: Embeddable React component (`Widget.tsx`) for points balance, redemption, SMS/email referral popup, GDPR request form, referral status with progress bar (`ReferralProgress.tsx`), RFM nudges.
- On-Site Content (Must Have): SEO-friendly loyalty page, rewards panel, launcher button, points display on product/checkout pages, post-purchase/email capture popups, GDPR request form, referral status with progress bar.
- Admin module: Frontend for merchant management, RFM segments, analytics, logs, rate limits, multi-user roles, customer import, notification templates.
- Run Lighthouse CI in GitHub Actions for accessibility (ARIA, keyboard nav) and performance (LCP, FID, CLS), targeting 90+ scores.
- Use AI (Grok, Cursor) for React components and Cypress tests; manually review for Polaris compliance.

- **Integrations**:
  - Shopify: APIs for orders, customers, discounts; POS with offline mode; Checkout UI Extensions with Checkout Extensibility APIs.
  - Klaviyo/Postscript: SMS/email referrals, notification templates for points, referrals, GDPR requests.
  - Reviews: Yotpo or Judge.me for points-for-reviews.
  - Email: Basic Klaviyo/Mailchimp for automated loyalty email flows.
  - Data Import: Async CSV import for Smile.io/LoyaltyLion migration with RBAC enforcement (`admin:full`).
  - Multi-store point sharing: Enable points sharing across Shopify Plus multi-store setups, storing shared points in Redis Cluster with `merchant_group_id` .
- **RBAC Enforcement**:
  - Enforce RBAC for customer import (`admin:full`), campaign management (`admin:full` , `admin:points`), and admin user management (`admin:full`) in Admin Service.
- **Testing (In-House)**:
  - Unit tests: Jest for NestJS APIs, `cargo test` for Rust Functions, Jest for RFM logic and campaign discounts.

- Integration tests: Shopify/Klaviyo/Postscript/RFM/data-import/checkout extension/GDPR/referral status flows (Jest).
- E2E tests: Dashboard, widget, RFM UI, popups, GDPR form, referral status, notification templates, rate limit monitoring, checkout extensions, usage thresholds (Cypress).
- Load test: 5,000 customers (Shopify) and 50,000 customers (Plus) with Redis Cluster caching and PostgreSQL partitioning.
- Optimize PostgreSQL with range partitioning for `points_transactions`, `referrals`, `reward_redemptions`, `rfm_segment_counts` using `created_at`.
- Develop test data factory (`test/factories/merchant.ts`, `test/factories/customer.ts`, `fixtures.rs`) for merchants, customers, referrals, and RFM scores, integrated with `dev.sh`.
- Use AI (Grok, Cursor) to generate Jest/Cypress tests and optimize PostgreSQL indexes; manually validate for coverage.

- **Deployment**:
  - Deploy on VPS (Ubuntu with Docker) using Docker Compose for microservices (auth, points, referrals, rfm_analytics, event_tracking, admin), PostgreSQL, Redis Cluster, Kafka, Loki + Grafana, and Vite + React frontend.
  - Use Nginx for frontend assets and reverse proxy to NestJS APIs.
  - Set up GitHub Actions for CI/CD with change detection, Lighthouse CI, and daily backups (`pg_dump`, Redis snapshots to Backblaze B2).
  - Provide `dev.sh` script to start Docker containers, seed mock data with Faker, and simulate RFM scores.
- **Enhancements & Best Practices**:
  - Integrate Loki + Grafana for centralized logging, tagging with `shop_domain`, `merchant_id`, `service_name`.
  - Implement rate-limiting middleware for Shopify API (2 req/s for Shopify, 40 req/s for Plus) with centralized tracking in Redis Cluster.
  - Use short-lived JWTs (15 minutes) with refresh tokens for admin APIs.
  - Add webhook idempotency using Redis Cluster and dead-letter queue (Kafka) for failures.
  - Use k6 for load testing in CI pipeline, including Plus-scale scenarios (50,000 customers, 1,000 orders/hour).
  - Monitor Shopify API deprecations via changelog RSS feed in CI pipeline.
- **Additional Tasks**:
  - Implement campaign discounts with RFM conditions in Points Service, using Shopify Functions (Rust/Wasm).
  - Build async CSV import system with RBAC enforcement (`admin:full`).
  - Implement referral progress bar in `ReferralProgress.tsx` and API (`/v1/api/referrals/progress`).
  - Add notification template live preview in `SettingsPage.tsx` and API (`/v1/api/notifications/template`).
  - Set up rate limit alerts (Slack/email) and centralized tracking in Admin Service.

  - Implement usage thresholds API (`/v1/api/plan/usage`) and UI (progress bar, nudges).
  - Set up `pg_dump` , Redis snapshotting, and Backblaze B2 backups.
- **Deliverables**:
  - TVP with Must Have features across microservices: Points, SMS/email referrals, basic RFM analytics, Shopify POS (offline mode), checkout extensions, email flows, data import, GDPR request form, referral status with progress bar, notification templates with live preview, campaign discounts, rate limit monitoring with alerts, usage thresholds, upgrade nudges.
  - Admin module with merchant management, RFM segments, analytics, logs, RBAC, customer import, rate limits, notification templates.
  - Shopify/Klaviyo/Postscript/Yotpo/Klaviyo integrations.
  - Test reports and bug fixes (in-house QA).
  - VPS deployment with Docker Compose, Nginx, Loki, and `dev.sh` script.

Phase 4: Beta Testing and Refinement (6 Weeks)

Goals

- Validate TVP with merchants, focusing on Must Have features across microservices, including checkout extensions, GDPR request form, referral status with progress bar, notification templates with live preview, customer import, campaign discounts, rate limit monitoring with alerts, usage thresholds, upgrade nudges.
- Refine based on feedback for Shopify App Store launch.

Tasks

- **Beta Testing**:
  - Recruit 10–15 merchants (including 2–3 Shopify Plus) via Shopify Reddit/Discord, Partners program, and Slack community (free 300-order plan).
  - Test Must Have features: Points, SMS/email referrals (7%+ SMS conversion, 3%+ email conversion), basic RFM analytics (80%+ wizard completion), Shopify POS (offline mode), checkout extensions (20%+ redemption rate), data import (90%+ success rate), GDPR request form (50%+ usage), referral status with progress bar (60%+ engagement), notification templates with live preview (80%+ usage), campaign discounts (10%+ redemption for Plus), rate limit monitoring, usage thresholds, upgrade nudges.
  - Collect feedback on RFM usability, referral conversion rates, POS syncing, checkout extension adoption, GDPR form usability, referral status engagement, notification template effectiveness, customer import accuracy, campaign discount performance, rate limit monitoring utility, upgrade funnel effectiveness.

    - Add PostHog events for Plus-specific features (e.g., `checkout_extension_redeemed`, `multi_store_points_shared`, `gdpr_request_submitted`, `campaign_discount_redeemed`, `rate_limit_viewed`, `campaign_rfm_conversion`, `plan_limit_warning`).
    - Set up Slack community for beta merchants to share feedback and encourage referrals.
- **Refinement**:
    - Fix bugs in RFM calculations, referral popups, POS offline mode, checkout extensions, GDPR form, referral status, notification templates, customer import, campaign discounts, rate limit monitoring, usage thresholds across microservices.
    - Enhance RFM analytics with segment engagement metrics (e.g., 10%+ repeat purchase uplift for RFM-targeted campaigns) and segment preview accuracy.
    - Optimize Redis Cluster caching, PostgreSQL partitioning, and Loki logging for performance.
    - Test disaster recovery (restore from Backblaze B2 backups).
    - Begin Should Have development: VIP tiers, exit-intent popups (referral-focused), behavioral segmentation (cart abandoners, frequent browsers via PostHog), multi-store point sharing, Shopify Flow templates.
    - Implement Theme App Extensions for customer widget (points balance, referral popup, GDPR form).
- **Documentation and Support**:
    - Create guides/YouTube tutorials for no-code setup (RFM wizard, points, referrals, checkout extensions, GDPR form, notification templates, customer import, campaign discounts, rate limit monitoring, usage thresholds, VPS deployment) using AI for script generation.
    - Develop Shopify Plus onboarding guide for multi-user setup, checkout extensions, multi-store point sharing, campaign discounts, rate limit monitoring, Theme App Extensions, Shopify Flow templates.
    - Provide Docker Compose and `dev.sh` scripts for microservices and VPS setup.
    - Set up email support and Slack community; plan live chat for paid plans.
- **Enhancements & Best Practices**:
    - Use PostHog to track feature adoption (e.g., `sms_referral_clicked`, `gdpr_request_submitted`, `referral_progress_viewed`, `notification_template_edited`, `customer_import_completed`, `campaign_discount_redeemed`, `rate_limit_viewed`, `plan_limit_warning`).
    - Conduct surveys and interviews via Slack community for structured feedback.
    - Maintain a public changelog for transparency.
- **Additional Tasks**:
    - Implement GDPR/CCPA webhook handling with `gdpr_requests` table for `customers/data_request` and `customers/redact`, ensuring `retention_expires_at` compliance.

- Add behavioral segmentation (cart abandoners, frequent browsers) using PostHog events (`cart_abandoned` , `product_viewed`) and expose via `/v1/api/segments/behavioral` .
- Build exit-intent popups for referrals (`ReferralPopup.tsx` , triggered by PostHog `page_exit` event).
- Implement Theme App Extensions for widget using Shopify CLI.
- Test backup restore process with Backblaze B2.

- **Deliverables**:
  - Beta test report (RFM usability, referral rates, POS performance, checkout extension adoption, GDPR form, referral status, notification templates, customer import, campaign discounts, rate limits, upgrade funnel).
  - Refined TVP with bug fixes, RFM enhancements, and Theme App Extensions.
  - Documentation, tutorials, support portal, VPS deployment guide, Shopify Plus onboarding guide, Slack community.

Phase 5: Launch and Marketing (7 Weeks)

Goals

- Launch on Shopify App Store with Must Have features and select Should Have features (VIP tiers, behavioral segmentation, exit-intent popups, Theme App Extensions, Shopify Flow templates).
- Attract 100+ merchants (including 5–10 Shopify Plus) in 3 months.

Tasks

- **Shopify App Store Submission**:
  - Ensure Polaris UI, App Bridge, GDPR/CCPA compliance (encrypt `customers.email` , `rfm_score` ; handle GDPR webhooks with `gdpr_requests` table; `retention_expires_at` tracking).
  - Optimize listing with 30-second demo videos (RFM setup, SMS referrals, checkout extensions, GDPR compliance, Theme App Extensions) and keywords (“loyalty program,” “RFM analytics,” “SMS referrals,” “Shopify POS”).
  - Highlight $29/month pricing, data import, and USPs (RFM, SMS referrals, Theme App Extensions) in description, using AI for copywriting and A/B testing (SensorTower).
- **Marketing Strategy**:
  - Launch website with pricing: Free (300 orders, basic RFM, 50 SMS referrals), $29/month (500 orders, full RFM, checkout extensions, campaign discounts), $99/month (1,500 orders, multi-store), Enterprise (custom).

    - Promote via Shopify Reddit/Discord, Slack community, social media (Facebook, Instagram), Shopify Plus agencies (e.g., Eastside Co, Pixel Union).
    - Publish mini-case studies during beta (e.g., “Merchant X: 10% referral conversion, 20% checkout extension adoption”) on Reddit/Discord/Slack.
    - Offer merchant referral program ($50 credit per referral) to drive organic growth.
- **Post-Launch Support**:
    - Monitor performance via admin module (`audit_logs` , `api_logs`) and Loki + Grafana on VPS.
    - Offer 24/7 email support, Slack community, and live chat for paid plans; white-glove onboarding for Plus.
    - Add Should Have features: Multi-store point sharing, Klaviyo/Mailchimp integration, discount banners, Shopify Flow templates.
- **Enhancements & Best Practices**:
    - Use PostHog to monitor onboarding drop-off and feature adoption (e.g., `checkout_extension_redeemed` , `gdpr_request_submitted` , `campaign_discount_redeemed` , `rate_limit_viewed` , `plan_limit_warning`).
    - Include screenshots and 30-second videos in merchant documentation, generated with AI.
    - Optimize VPS (Nginx, Docker, Loki) for low latency across microservices.
- **Additional Tasks**:
    - Implement Shopify Flow templates (e.g., “RFM Churn Risk → Tag Customer”, “Referral Completed → Add Points”) and document in App Store listing.
    - Implement merchant referral program in Referrals Service (`/v1/api/referrals/merchant`).
    - A/B test App Store listing copy using SensorTower.
- **Deliverables**:
    - Approved Shopify App Store listing.
    - Marketing website, promotional materials (mini-case studies, videos).
    - Support system with onboarding guides, Slack community, and VPS maintenance docs.

Phase 6: Post-Launch and Scaling (Ongoing)

Goals

- Grow to 100+ merchants (including 5–10 Shopify Plus) in 3 months.
- Achieve Built for Shopify certification.
- Implement Should Have and Could Have features.

Tasks

- **User Acquisition**:
  - Partner with Shopify Plus agencies (e.g., Eastside Co, Pixel Union) and offer white-label API (`/v1/api/whitelabel`) for Enterprise plans.
  - Run ads emphasizing RFM analytics, SMS referrals, checkout extensions, campaign discounts, rate limit monitoring, Theme App Extensions, and $29/month pricing.
  - Publish case studies (e.g., 20% repeat purchase increase via RFM, 50% multi-store point sharing adoption, 15% campaign discount redemption).
  - Engage Slack community for feedback and organic referrals.
- **Feature Expansion**:
  - **Should Have**:
    - Full RFM configuration (thresholds, tiers, adjustments) with Rust/Wasm for real-time updates.
    - VIP tiers (engagement-based perks: early access, birthday gifts).
    - Advanced Klaviyo events, Postscript integration.
    - Point calculators, checkout extensions, behavioral segmentation.
  - **Could Have**:
    - Gamification (badges, leaderboards).
    - Multilingual widget (Spanish, French, German, etc., using `i18next`).
    - Multi-currency discounts.
    - Non-Shopify POS (Square, Lightspeed).
    - Advanced analytics (25+ reports, ROI dashboard).
    - Developer toolkit with webhook-based integrations for Shopify Plus (ERP/CRM).
    - AI-powered reward suggestions via xAI API (`/v1/api/rewards/recommend` , see https://x.ai/api).
    - Custom webhook APIs (`/v1/api/webhooks/custom`) for automation triggers.

- **Maintenance**:
  - Release quarterly updates (e.g., RFM tiers, gamification).
  - Monitor Shopify API changes via changelog RSS feed and webhooks.
  - Optimize PostgreSQL with partitioning for `points_transactions`, `referrals`, `reward_redemptions`, `vip_tiers`, `rfm_segment_counts`.
  - Maintain Docker containers, Nginx, Redis Cluster, Kafka, and Loki on VPS.
  - Run daily backups (`pg_dump`, Redis snapshots) to Backblaze B2.
- **Certification**:
  - Apply for Built for Shopify certification (4.5+ star rating after 3–6 months).
- **Enhancements & Best Practices**:
  - Scale VPS to managed DB/Redis or Kubernetes for 5,000+ merchants (50,000+ for Plus).
  - Regularly update developer/merchant documentation with AI-generated content.
  - Monitor RFM engagement, referral conversion rates (SMS: 7%+, email: 3%+), campaign discount redemption (15%+ for Plus), and checkout extension adoption (20%+ redemption rate).
- **Deliverables**:
  - Monthly performance reports (merchant growth, RFM engagement, referral rates, campaign discount redemption, checkout extension adoption).
  - Feature updates (Should Have and Could Have).
  - Built for Shopify certification application.
  - VPS maintenance and optimization guide for microservices.

Updated Timeline

- Phase 1: 4 weeks
- Phase 2: 6.5 weeks (extended for usage thresholds/nudges)
- Phase 3: 19 weeks (extended for logging, rate limits, backups, dev script, data factory, Lighthouse)
- Phase 4: 6 weeks (extended for Slack community, backup testing; Theme App Extensions deferred to Phase 5)
- Phase 5: 7 weeks (extended for Theme App Extensions, Shopify Flow templates)

- Phase 6: Ongoing (includes custom webhooks)
- **Total**: 39.5 weeks (7–8 months for TVP, 12–14 months for full implementation)

Adjustments:

- Extended Phase 2 by 0.5 weeks for usage thresholds and upgrade nudges.
- Extended Phase 3 by 3 weeks for Loki + Grafana, centralized rate limits, backups, `dev.sh` script, data factory, and Lighthouse CI, but reduced by moving Theme App Extensions to Phase 5.
- Extended Phase 4 by 0.5 weeks for Slack community setup and 0.5 weeks for backup testing.
- Extended Phase 5 by 1 week for Theme App Extensions and Shopify Flow templates.
- Added 0.5 weeks in Phase 6 for custom webhooks.
- Maintained 39.5-week total by prioritizing Must Have features and deferring Theme App Extensions and Shopify Flow to Phase 5.

Budget Estimate

- Development: $74,750 (TVP, AI tools, microservices setup, Shopify Plus features, in-house UI/UX and QA, backups, community, enhanced features).
- Marketing: $3,000 (website, Shopify community ads, social media).
- Support Infrastructure: $4,000 (VPS hosting, PostHog, Loki, Backblaze, support tools).
- Contingency (15%): $11,362.50.
- **Total**: $91,912.50.

Adjustments:

- Reduced budget by $5,000 by moving UI/UX and QA in-house.
- Increased budget by $7,000 for new features (usage thresholds, Shopify Flow, Theme App Extensions, Loki, backups, community, data factory, Lighthouse).
- Reallocated $2,000 from contingency to development for microservices enhancements.

Risks and Mitigation

- **Risk**: Shopify API changes.
  - **Mitigation**: Use `@shopify/shopify-app-express`, pin API versions (2025-01), monitor changelog RSS feed, test webhooks in CI pipeline.
- **Risk**: High competition.
  - **Mitigation**: Emphasize free RFM analytics, SMS referrals, checkout extensions, campaign discounts, rate limit monitoring, Theme App Extensions, $29/month pricing, and data import in marketing.
- **Risk**: Slow adoption.
  - **Mitigation**: Free plan (300 orders, 50 SMS referrals), 14-day trial, Shopify Reddit/Discord/Slack engagement, mini-case studies, merchant referral program, usage thresholds, upgrade nudges.
- **Risk**: Onboarding complexity (especially for Shopify Plus).
  - **Mitigation**: RFM setup wizard, gamified onboarding checklist, GDPR form, notification templates, tooltips, YouTube tutorials, Plus-specific onboarding guide, Theme App Extensions, Shopify Flow templates, white-glove onboarding.
- **Risk**: Scalability for 5,000+ merchants (50,000+ for Plus).
  - **Mitigation**: Microservices with Redis Cluster, PostgreSQL partitioning, Rust for RFM/campaign discounts, Dockerized VPS, Kafka for events, Loki for logging, Plus-scale load testing.
- **Risk**: AI code quality.
  - **Mitigation**: Manual review, Jest/Cypress tests, in-house QA with AI, data factory, load testing.
- **Risk**: Solo developer bandwidth.
  - **Mitigation**: Leverage AI for code/tests/UI/UX/QA, prioritize Must Have features, defer Should Have to Phase 4, use Nx for monorepo management, `dev.sh` for local setup.
- **Risk**: Microservices complexity.
  - **Mitigation**: Use Nx for monorepo management, gRPC with circuit breakers, Kafka for async events, Docker Compose, CI/CD with change detection, Loki for logging.

Success Metrics

- **TVP (Phase 3–4)**: 90%+ merchant satisfaction, 80%+ RFM wizard completion, 7%+ SMS referral conversion, 3%+ email referral conversion, 85%+ checkout extension adoption, 20%+ checkout extension redemption rate, 50%+ GDPR form usage, 60%+ referral status engagement,

80%+ notification template usage, 90%+ customer import success rate, 10%+ campaign discount redemption for Plus merchants, 10%+ RFM campaign repeat purchase uplift.

- **Launch (Phase 5)**: 100+ merchants (including 5–10 Plus) in 3 months, 4.5+ star rating in 6 months.
- **Post-Launch (Phase 6)**: 20% repeat purchase increase, 10%+ RFM tier engagement, 50%+ multi-store point sharing adoption, 15%+ campaign discount redemption for Plus, Built for Shopify certification in 12 months.

Key Improvements

- **Architecture**: Transitioned to microservices (auth, points, referrals, rfm_analytics, event_tracking, admin) with Nx, Docker, REST for UI, gRPC with circuit breakers, Redis Cluster, Kafka, and Loki + Grafana for logging.
- **In-House Work**: Moved UI/UX and QA in-house, leveraging AI (Grok, Figma plugins, Cursor) to reduce costs by $5,000.
- **Feature Clarity**: Integrated Must Have (points, SMS/email referrals, basic RFM, Shopify POS with offline mode, checkout extensions, GDPR form, referral status with progress bar, notification templates with live preview, customer import, campaign discounts, rate limit monitoring with alerts, usage thresholds, upgrade nudges), Should Have (VIP tiers, advanced RFM, Klaviyo, popups, behavioral segmentation, multi-store point sharing, Shopify Flow templates, Theme App Extensions), and Could Have (gamification, multilingual, non-Shopify POS, advanced analytics, AI reward suggestions, custom webhooks) into phases.
- **Task Optimization**: Prioritized Must Have features for TVP, added referral progress bar, notification template preview, incremental RFM refresh, behavioral segmentation, exit-intent popups, Theme App Extensions, Shopify Flow templates, and merchant community.
- **API Design**: Added versioning (`/v1/api/*`), modular endpoints (`/rfm/segments` , `/rfm/segments/preview` , `/notifications/template` , `/rate-limits` , `/plan/usage`), rate limit alerts, centralized rate limit tracking, webhook idempotency, RBAC, gRPC for RFM Analytics and Admin services.
- **Shopify Plus**: Added checkout extensions with Checkout Extensibility, RBAC, multi-store point sharing, campaign discounts, rate limit monitoring with alerts, Theme App Extensions, Shopify Flow templates, Plus-scale load testing, and dedicated onboarding guide.
- **Risk Mitigation**: Included GDPR webhooks with retries, PostHog for Plus feature tracking, in-house QA with AI, Docker Compose, `dev.sh` , data factory, CI/CD with change detection and Lighthouse CI, `rfm_segment_counts` incremental refresh, `referral_link_id` , `merchant_referral_id` , `campaign_id` tracking, Loki logging, Backblaze backups.
- **Timeline/Budget**: Adjusted for in-house UI/UX and QA (-$5,000), new features (+$7,000), extended Phases 2–5 to maintain 39.5-week TVP timeline.