

Claw4Growth — PRD (Current State)

Last update: 2026-02-17 (UTC)

Owner: Luca

Execution: Tony + Pieter

Status: Build phase (MVP infra wiring)

1) Product Goal

Claw4Growth is a **managed AI marketing operator** for non-technical marketers.

Core promise:

- Buy in minutes
- Get an operator deployed automatically
- Start working via Telegram + connected marketing apps

2) Non-Negotiable Decisions

1. **Onboarding order (final):**

Login → Name → Brand → Tone → Payment → Apps → Deploy → Telegram setup

2. **Deploy should be automatic after payment** (no manual deploy step for users).

3. **Telegram bot flow (launch): BYOB**

- User creates bot on BotFather
- User adds token in dashboard
- Pairing flow links Telegram account

4. **Integrations use full redirect OAuth** (no popups).

5. **Meta Ads is covered by Facebook integration** (no separate Meta Ads app card).

6. **Single launch offer:** €34.90/mo (Early Bird).

7. **Wrapper is used for reverse-engineering only**; source of truth must be C4G repo.

3) Tech Architecture (Target)

Frontend

- Landing + onboarding UX: `claw4growth` repo
- Dashboard: wrapper-inspired, C4G-branded

Backend

- API routes for deploy/provisioning, Telegram, integrations, Stripe webhook handling
- Supabase for auth + persistence
- Docker provisioning on dedicated VPS

Runtime

- Per-customer OpenClaw container
- Per-customer workspace/memory/context
- Composio integrations linked per customer

4) Repository Strategy

Primary working repo (source of truth)

`/root/clawd/claw4growth`

Imported wrapper modules (already copied)

`/root/clawd/claw4growth/platform-integration/`

Contains:

- Docker lifecycle module
- Instance actions + DB access modules
- Telegram pairing modules + API routes
- Telegram dashboard components
- Supabase migration SQL files
- Transfer map doc

Wrapper repos

- `ClawWrapper-custom`: research/lab (can diverge)
- `ClawWrapper`: untouched template backup

5) Current State (As-Is)

■ Done

UX / Onboarding / Landing

- Onboarding flow order aligned
- Mobile fixes + CTA/layout fixes
- Deploy animation fixed (including power mascot sequence)
- Paid-state skip logic introduced
- OAuth redirect behavior standardized
- EN/IT landing improvements shipped

C4G Integration Direction

- Deploy API scaffold added: `api/deploy/route.ts`
- Wrapper critical modules imported into C4G repo under `platform-integration/`
- Migration drafts present in C4G repo

Ops

- Cron model emergency switch to GPT done (temporary)
- Backup/restore plan for cron model settings created

■■ Partially Done / Needs Wiring

- Deploy API exists but provisioning is still partially mocked
- Instance lifecycle code imported but not fully wired into active dashboard routes
- Telegram pages imported but not connected to active C4G runtime
- Supabase schema SQL present but needs apply/verify against production project

■ Missing for true MVP completion

1. Stripe webhook → automatic deploy trigger (robust, idempotent)
2. Docker provisioning live on dedicated VPS
3. Dashboard integration in active app (not just imported files)
4. Telegram BYOB flow end-to-end in active dashboard
5. Instance status polling and start/stop/restart actions in live UI
6. Production env setup + secrets management + health checks

6) MVP Scope (What must work)

A paying user must be able to:

1. Complete onboarding + payment
2. Be auto-provisioned to a running instance
3. Open dashboard URL
4. Add personal Telegram bot token (BYOB)
5. Pair Telegram account successfully
6. Connect at least 2 OAuth apps (e.g. Google + Facebook)
7. Send first Telegram command and receive valid response from own isolated instance

7) Delivery Plan (Execution Order)

Phase A — Infra Backbone (Priority 0)

1. Provision dedicated VPS (Hetzner)
2. Install Docker + Caddy/reverse proxy + TLS
3. Wire provisioning module from `platform-integration/lib/docker/containers.ts`
4. Add health endpoints + logs for provisioning status

Phase B — Data + Deploy Reliability (Priority 0)

1. Apply Supabase migrations
2. Add idempotent Stripe webhook handling
3. Store deployment state transitions ('provisioning/running/error') cleanly
4. Retry policy for transient failures

Phase C — Dashboard Live Wiring (Priority 1)

1. Move Telegram pages/components from `platform-integration` to active dashboard routes
2. Wire `instance-actions` to active UI
3. Add status cards + controls + errors

Phase D — Telegram + Integrations E2E (Priority 1)

1. BYOB token save/restart flow
2. Pairing code generation + webhook approval
3. OAuth integration state persistence in DB

Phase E — Hardening (Priority 2)

1. Metrics + alerts (deployment failures, queue delay)
2. Resource limits per container
3. Support playbook + runbook

8) Brand/UI Requirements for Dashboard

Must be fully C4G, not wrapper-generic:

- Color system aligned to C4G brand
- Copy in C4G tone (marketer-friendly, non-technical)
- KPI-first layout (campaign utility, not infra jargon)
- Clear state labels: setup complete / action needed
- Mobile-first dashboard behavior

9) Environment Variables (Minimum)

- Supabase:
 - `NEXT_PUBLIC_SUPABASE_URL`
 - `NEXT_PUBLIC_SUPABASE_PUBLISHABLE_KEY`
 - `SUPABASE_SERVICE_ROLE_KEY`
- Stripe:
 - `STRIPE_SECRET_KEY`
 - `STRIPE_WEBHOOK_SECRET`
 - `C4G_STRIPE_PRICE_ID`
- Runtime:
 - `ENCRYPTION_KEY`
 - `DEPLOYED_PRODUCT_IMAGE`
 - `DEPLOYED_PRODUCT_PORT`
 - `PLATFORM_TELEGRAM_BOT_TOKEN` (optional fallback)

- Telegram (BYOB flow pages/webhook):
 - `TELEGRAM_BOT_USERNAME` (if platform bot used for pairing)
 - `TELEGRAM_WEBHOOK_SECRET`

10) Immediate Next Actions (for PC continuation)

1. Start from repo: `claw4growth`
2. Read: `platform-integration/docs/TRANSFER-MAP.md`
3. Wire imported modules into active app routes
4. Apply migrations to Supabase
5. Implement Stripe webhook auto-deploy idempotency
6. Deploy backend runtime on dedicated VPS
7. Test full path with one real test customer

11) Definition of Done (MVP)

MVP is done only when:

- Payment triggers auto-deploy reliably
- Instance is reachable and healthy
- Telegram BYOB setup works without manual engineering intervention
- User can use operator from Telegram on first day
- No cross-user data leakage (isolated containers + isolated memory)

Single source of execution now: `claw4growth` repo.