Social Army (Discord-first, auto-post + leaderboard)

Concept Overview

- Goal: Gamify brand amplification. Users connect socials, 1-click post brand missions, auto-track engagement, monthly leaderboard + rewards.
- Platforms (Phase 1 → Phase 2):
  - Phase 1: LinkedIn, Instagram Business (Meta), TikTok (+ unique short links).
  - Phase 2: X/Twitter (if API tier/cost ok), Threads (when approved).
-
- Auth & Permissions: OAuth per platform; store access/refresh tokens + expiry.
  - LinkedIn → w_member_social
  - Meta → pages_manage_posts, instagram_content_publish
  - TikTok → content posting
  - X/Twitter → user-write (optional)
-
- Discord Bot Commands:
  - /connect, /missions, /post, /score, /leaderboard, /rolesync
  - Admin: /mission new|edit|push|close, /audit @user
-
- Mission Flow:
  - Admin creates mission (copy, media, platforms, start/end, UTM).
  - Bot posts mission card in #social-army.
  - User clicks Connect & Post → preview → confirm.
  - Backend publishes; saves post IDs/URLs; awards points.
-
- Tracking & Scoring:
  - Unique short link per user/mission.
  - Poll APIs for engagement; shortener for clicks.
  - Example points: +10 publish, +5 on-time, +1 per 10 clicks, +5/10/20 engagement tiers.
-
- Roles/Tiers: Recruit / Soldier / General / Warlord (auto-assigned by score).

Social Army — Tech Spec (v0.1)

Architecture

- Discord Bot (Node/TS or Python)
- API Backend (Express/FastAPI)
- Database: Postgres (+ Redis cache/leaderboard)
- Queue/Workers: BullMQ (Node) or Celery (Python)
- Link Shortener: Bitly/Rebrandly (or self-hosted)

- Media storage: S3/Cloudflare R2
- Cron workers for metrics

OAuth & Scopes

- LinkedIn: w_member_social, r_liteprofile
- Meta (IG Business/FB): pages_manage_posts, pages_read_engagement, pages_show_list, instagram_basic, instagram_content_publish
- TikTok: content posting scopes
- (Phase 2) X/Twitter: user-write

Data Model (Postgres tables)

- users
- social_accounts
- missions
- mission_assignments
- posts (with metrics json)
- scores

Discord Bot Commands

- /connect, /missions, /post, /score, /leaderboard, /rolesync
- Admin: /mission new|edit|push|close, /audit @user

REST Endpoints

- POST /oauth/:platform/start → begin OAuth
- GET /oauth/:platform/callback → save tokens
- POST /missions → create mission
- PATCH /missions/:id → edit/close mission
- GET /missions?status=live → list active
- POST /missions/:id/assign → assign mission to user
- POST /missions/:id/post → publish for user
- GET /users/:id/score → get score
- GET /leaderboard?month=YYYY-MM → top N

Mission Workflow

1. Admin creates mission → bot posts mission card.
2. User clicks Connect & Post → OAuth (once) → confirm.
3. Backend publishes → saves post IDs/URLs → awards base points.
4. Workers poll metrics + clicks → update scores.
5. Leaderboard updates in Redis; monthly rollover resets & announces winners.

Scoring (configurable)

- Publish success: +10

- On-time (≤24h): +5
- Clicks: +1 per 10 unique clicks
- Engagement tiers:
  - LinkedIn: +5 at 10 reactions, +10 at 30, +20 at 100
  - IG/TikTok: +5 at 100 views, +10 at 500, +20 at 2000
-
- Bonus: top 3 clickers → +25 / +15 / +10

Security & Ops

- Encrypt tokens, rotate refresh tokens
- /disconnect command to revoke
- Rate limiting + retries with exponential backoff
- Content safeguards (min caption length, banned words, cooldowns)
- Audit log channel (every post/metric/score update echoed to mods)

Roles & Cron

- Roles: Recruit, Soldier, General, Warlord
- Cron schedule:
  - Click sync: every 5 minutes
  - Engagement polling: every 60 minutes
  - Monthly rollover: 1st of month, 00:05
-