# Phase 1 — Solid foundation

1. **Secrets & config**
   * Move hardcoded secrets to env vars: JWT\_SECRET, ALGORITHM, CORS origins.
   * Add .env.example and load via python-dotenv (backend) and Vite env (.env.local) for the frontend.
2. **Database**
   * Switch from SQLite → **Postgres** (SQLite is fine for dev, not for prod concurrency).
   * Add migration setup (Alembic) and create tables for users, mentions, baselines, scores, sessions.
3. **API shape finalization**
   * Freeze schemas (Token, SentimentItem) and add pagination & filters: ?tickers=AAPL,TSLA&limit=50&sort=interest\_score.

# Phase 2 — Real data pipeline

1. **Cashtag collector**
   * Build a small service that collects tweets with cashtags ($TSLA, $NVDA, …).
   * Because X/Twitter API access can be tricky: consider **official API** (paid tiers) or a vendor (e.g., Gnip-like providers) or a private dataset source. If you want, we can also prototype with **Reddit + StockTwits** as a stopgap.
2. **Baseline & scoring**
   * For each ticker, compute baseline (rolling 7/30-day mean & std).
   * Compute: interest = mentions / baseline\_mean, z = (mentions - mean)/std, plus sentiment (rule-based or a quick model).
   * Materialize yesterday’s daily rollup for the **Free list**; keep a minute-by-minute (or 5-min) rollup for **Pro realtime**.
3. **Streaming**
   * Replace mock generator with a Redis pub/sub or Postgres NOTIFY → broadcast updates to WebSocket clients.

# Phase 3 — Productization

1. **Auth & payments**
   * Add Stripe Checkout (Test mode) → create pro flag on users after successful webhook.
   * Gate /api/pro/\* and /ws/\* by pro status.
2. **Rate limiting & abuse controls**
   * Basic request throttling (e.g., slowapi or a simple token bucket).
   * CORS tighten to your domain only.
3. **Logging & monitoring**
   * Structured logs (JSON), request IDs, error capture (Sentry).
   * Health checks, /api/health shows DB + queue connectivity.

# Phase 4 — Deployment

1. **Containerization**
   * Add Dockerfile (backend) and docker-compose.yml (backend + Postgres + Redis + frontend).
2. **CI/CD**
   * GitHub Actions:
     + Lint/test on PRs.
     + Build & push Docker image on main tags.
3. **Hosting**
   * Minimal: one VM with Docker Compose (front+back+db+redis, HTTPS via Caddy or Nginx + Certbot).
   * Or managed: Render/Fly/Heroku + Neon/ElephantSQL (Postgres) + Upstash (Redis).

# Phase 5 — UX polish

1. **Frontend**
   * Add search & filters (by ticker, sector), sorting, and a simple sparkline per ticker.
   * “Market Index” card: sum of abnormal interest across tickers.
2. **Docs & pricing**
   * Simple pricing page, ToS/Privacy stub, support email, status page link.
3. **Onboarding**
   * Email verification, password reset, “Getting Started” modal for Pro.