**TDA – DiscoveryBot (Grid-Based Search Scheduler) – Final Comment**

**What was delivered**

* Implemented workers/discovery\_bot.py (async) that:
  + Generates a configurable grid around **Rotterdam** and iterates **categories**.
  + Calls **Google Places v1 – Nearby** via app/services/google\_service.py (async httpx, field masks, retry/backoff).
  + Maps results to our locations schema (no website column; aligns with our table).
  + Inserts **idempotently** using ON CONFLICT (place\_id) DO NOTHING and sets state='CANDIDATE'.

**How to run (manual / CLI)**

cd Backend

source .venv/bin/activate

export GOOGLE\_API\_KEY="\*\*\*" # in .env of shell

python workers/discovery\_bot.py \

--city rotterdam \

--categories bakery,restaurant,supermarket \

--nearby-radius-m 1000 \

--grid-span-km 12 \

--max-per-cell-per-category 20 \

--inter-call-sleep-s 0.15

* Grid size at these params: **1056 cells**; 3 cats ≈ **3168 API calls**.

**Result snapshot (this run)**

* Console: Klaar. Totaal nieuw ingevoegd (idempotent): ~3166.
* DB checks during/after run (examples):
  + SELECT COUNT(\*) FROM locations WHERE state='CANDIDATE'; → **~2k+** (keeps rising while run progresses).
  + Duplicates:
  + SELECT place\_id, COUNT(\*)
  + FROM locations GROUP BY place\_id HAVING COUNT(\*) > 1;

→ **0 rows** (dedup OK).

* + Sample Rotterdam bounding box shows steady inflow of bakery/restaurant/supermarket candidates.

**Acceptance Criteria**

* ✅ Bot can be run manually from command line (see command above).
* ✅ Processes at least 3 different business categories for Rotterdam.
* ✅ Successfully deduplicates results based on place\_id before insertion (set + ON CONFLICT).
* ✅ After run, locations contains **500+** new records with state='CANDIDATE' (exceeded; multiple thousands observed).

**Definition of Done**

* ✅ DiscoveryBot executable & functional.
* ✅ Grid-based search for Rotterdam operational.
* ✅ Deduplication implemented and verified (0 dupes).
* ✅ Database populated with 500+ candidates (well over target).

**Hardening added**

* **Run caps:** --max-total-inserts (stop after N inserts), --max-cells-per-category (process only first M cells per category).
* **Config readiness:** structure supports external config/profile (city, categories, radius, span, pacing) — future multi-city runs are trivial.
* **Idempotency & safety:** DB-side uniqueness + small inter-call sleep + retry/backoff on 429/5xx.

**Notes / deviations**

* Our table does not include a website column; mapping/INSERT align with current schema (no write to website).
* GOOGLE\_API\_KEY used from .env/environment (documented in README section).

**Next steps (follow-up tickets, optional)**

* Schedule periodic runs (cron/Render job) with caps (e.g., weekends).
* Add run\_stats table (started/ended, cells processed, categories, insert count, errors).
* Optional: polygon clip to strict Rotterdam municipality bounds (reduce spillover).

— End of comment —