**Title:** TDA-7 – Google Places API Integration Service (Nearby & Text) — Completed

**Summary**  
Implemented a robust, quota-efficient Google Places integration as a reusable service (services/google\_service.py). The service wraps **Places API v1** (Text Search & Nearby Search), uses **field masks** to minimize payloads/quota, implements **exponential backoff with jitter** for transient failures (429/5xx/network), and provides **structured logging** for auditing and cost tracking. A small dev router exposes two GET endpoints for manual verification.

**What we built**

* **Core service**: GooglePlacesService with:
  + search\_text(textQuery, max\_results, language, region)
  + search\_nearby(lat, lng, radius, included\_types?, max\_results, language)
* **Field mask** via X-Goog-FieldMask (minimal set: id, displayName, formattedAddress, location, types, rating, userRatingCount, businessStatus, websiteUri).
* **Error handling & retries**:
  + Retries on **429/503** (honors Retry-After) and **5xx/network** with exponential backoff + jitter.
  + Client 4xx (non-429) are terminal and logged via \_log\_error.
* **Structured logging** (structlog):
  + Per call: endpoint, status\_code, payload\_summary (PII-safe), (optional) request\_id header.
  + Errors: status\_code, Google error message/status, short response snippet.
* **Configuration** (.env → Pydantic v2 settings):
  + GOOGLE\_API\_KEY, GOOGLE\_PLACES\_LANGUAGE, GOOGLE\_PLACES\_REGION.
  + extra="ignore" to tolerate unrelated env keys; case-insensitive matching.

**Dev test endpoints** (FastAPI – dev only)

* GET /dev/google/text?q=Turkse bakkerij Rotterdam&max\_results=10
  + **Observed:** 10 items returned (names/addresses/types/ratings/coords).
* GET /dev/google/nearby?lat=51.9244&lng=4.4777&radius=2000&max\_results=20
  + **Observed:** 20 items returned around Rotterdam Centraal.

**Acceptance Criteria — Evidence**

* **Given lat/lng/radius → list of places**: ✅ Verified via /dev/google/nearby (20 results).
* **Given text query → list of places**: ✅ Verified via /dev/google/text (10 results).
* **Exponential backoff**: ✅ Implemented (429/5xx/network; honors Retry-After).
* **Audit logging**: ✅ Every API call logged with status + summarized payload; errors captured with details.

**Definition of Done — Status**

* **Integration working**: ✅
* **Error handling & retry logic**: ✅
* **Comprehensive logging**: ✅
* **Field masks for quota efficiency**: ✅

**Security & quota notes**

* API key is read from server-side .env (not committed).
* Recommend API restriction to **Places API** only; for local dev, avoid HTTP referrer restrictions (use None/IP restriction).
* Field masks and capped max\_results keep responses lean.

**Next steps (out of scope for TDA-7)**

* Add a small **in-process rate gate** (token bucket) to smooth bursts.
* **Unit tests** with pytest + respx for success/429/5xx branches.
* Extend /nearby to accept included\_types (filter categories).
* Wire the service into the **Discovery pipeline** (grid scan + dedup + DB persist as CANDIDATE).

**Conclusion**  
TDA-7 meets all Acceptance Criteria and DoD. The service is stable, efficient, and ready to be consumed by the Discovery jobs in the next epic step.

Als je wilt, maak ik er direct 2 subtasks bij (unit tests + rate gate) met concrete todo’s en schattingen.