**Backend Development Plan for Released Email App**

**Phase 1: Core Setup & User Management**

**✅ Task 1.1 – Project Initialization**

* Set up a Python virtual environment
* Create project structure: app/, models/, routes/, utils/, config/
* Install Flask, Flask-CORS, Flask-SQLAlchemy, Flask-Migrate, python-dotenv
* Create a config.py file for environment settings

**✅ Task 1.2 – Database Setup**

* Define User model with fields: id, name, email, auth\_provider, created\_at
* Set up SQLite or PostgreSQL with SQLAlchemy
* Apply migrations with Flask-Migrate

**✅ Task 1.3 – Authentication**

* Implement Google OAuth2 token verification route (using google-auth)
* Store users after successful token verification
* Return JWT or session token for frontend use

**Phase 2: Subscription Handling**

**🔄 Task 2.1 – Store Subscription Scan Results**

* Define Subscription model: id, user\_id (FK), from\_name, from\_email, subject, category, unsub\_link, created\_at
* Create route to accept and store scanned results (bulk insert)
* Link each subscription to its user

**🔄 Task 2.2 – Categorization Logic (Optional)**

* Allow re-categorization or enhancement of classification using backend rules

**🔄 Task 2.3 – Retrieve Dashboard Data**

* Create route to return total subscriptions, grouped by category
* Return category summary and individual subscription previews

**Phase 3: Unsubscribe Workflow**

**🔄 Task 3.1 – Logging Unsubscribe Actions**

* Define UnsubscribeAction model: id, subscription\_id (FK), status, method, timestamp
* Create route to log when an unsubscribe is attempted or succeeded

**🔄 Task 3.2 – Process Gmail Unsubscribe Requests (Optional)**

* Store tokens with refresh support (encrypted)
* Implement server-side Gmail API interaction (for bulk unsubscribe filters)

**Phase 4: User Dashboard & API Polish**

**🔄 Task 4.1 – User-specific API**

* Secure all routes to check current user (JWT/session)
* Add user profile route (get/update basic info)

**🔄 Task 4.2 – Analytics (Optional)**

* Return stats: unsub rate, most common senders, etc.
* Prepare routes to support future frontend charts

**Tools**

* Python 3.10+
* Flask + Flask-CORS
* SQLAlchemy + Alembic
* PostgreSQL or SQLite
* Optional: Redis for caching or task queue later

Let me know when you’re ready to begin implementation for Task 1.1 and I’ll scaffold it for you.