Released Email App - Backend Implementation Plan

Overview:

This document outlines the backend architecture and features necessary to support the Released Email App.

It assumes prior backend experience and proposes a step-by-step development plan across core areas:

authentication, subscription storage, activity logs, user management, and infrastructure.

1. Technology Stack

- Framework: Flask (Python)
- Database: PostgreSQL or SQLite (dev)
- ORM: SQLAlchemy
- Auth: OAuth2 (Google), Flask-JWT
- Email APIs: Gmail API integration
- Deployment: Railway / Render / Fly.io

2. Core Backend Features

User Authentication

- OAuth with Google: Store minimal user info (email, sub ID)
- JWT issuance: For session state / frontend auth
- Protected routes: For user-specific data like subscriptions

Subscription Management

- Store scanned subscriptions (sender, subject, unsubscribe link, category)

- CRUD APIs: Read/delete subscriptions, update tags

- Categorization support: Optional field for ML/manual tag

Activity Logging

- Scanned inbox: Logged

- Unsubscribed from: Logged

- Manual action required: Logged

3. API Endpoints Sketch

Endpoint	Method	Auth	Description
/auth/google/callback	GET	No	OAuth redirect endpoint
/auth/token	POST	No	Exchange for JWT
/subscriptions	GET	Yes	List subscriptions
/subscriptions	POST	Yes	Store scanned subs
/subscriptions/:id	DELETE	Yes	Remove one
/logs	GET	Yes	Fetch recent activity

4. Suggested Daily Development Plan

Day 1: Project setup + Auth (Flask + JWT auth routes)

Day 2: Subscription model & API (DB model + endpoints)

Day 3: Activity logs (Model + attach to sub actions)

Day 4: Frontend integration (Save subs, fetch for dashboard)

Day 5: Error handling & filtering (Add filtering + categories)

Day 6: Manual Unsub log + retry (Add status updates)

Day 7: Deployment + DB migration (Production readiness)

5. Database Models (simplified sketch)

User(id, email, google_sub)

Subscription(id, user_id, from_name, from_email, subject, category, unsub_link, status)

ActivityLog(id, user_id, action, target, timestamp)

Extras:

- Celery or background jobs for async unsub requests

- Rate limiting

- Admin panel (Flask-Admin or custom dashboard)