# 🔐 Platform-Level Meta App Integration (OAuth & Token Proxy Layer)

## ✅ Module Overview

This module enables all connected brands to leverage a single platform-level Meta (Facebook/Instagram) app for authenticating and accessing social features (Pages, Messenger, IG DMs, etc.) without needing to register their own Meta Developer accounts. The system uses a secure proxy token and permission layer to route access and usage based on tenant configurations.

## 🎯 Goals

* Eliminate the need for each brand to register their own Facebook app
* Centralize and secure token storage, refresh, and API call routing
* Handle permissions, scopes, and reauthentication logic across tenants
* Enable fine-grained access control per brand
* Ensure GDPR and Meta Platform policy compliance

## ⚙️ System Architecture

### Key Components:

1. **Meta App (Platform-Level):**
   * Single approved app with permissions for: pages\_messaging, instagram\_manage\_messages, pages\_read\_engagement, etc.
   * Admin-controlled by the SaaS owner
2. **OAuth Proxy Service:**
   * Handles Meta login redirects and auth callback
   * Stores and encrypts access tokens securely per brand
   * Maps token to tenant-specific brand ID
3. **Token Vault:**
   * Encrypted storage layer (e.g., using AWS KMS, HashiCorp Vault)
   * Handles refresh tokens automatically via scheduled background workers
4. **Tenant Permission Router:**
   * Validates if the tenant has enabled Facebook/IG modules
   * Checks user roles and active scopes before allowing access
   * Maintains audit logs for all token usage
5. **Usage Throttler + Tracker:**
   * Monitors daily API limits and alerts superadmin if nearing quota
   * Reports usage per brand for fair billing and throttling

## 🧩 Feature Capabilities

* Connect Facebook Page → triggers OAuth flow
* Admins see which Page(s) are connected
* Brand admins can disconnect/re-auth when needed
* Token expiry reminders + auto-refresh flow
* Option to manually force re-authentication per brand
* Logs all access tokens, scopes, last-used timestamp, and expiry
* FB/IG API calls are made using token matched to brand’s assigned profile

## 🔐 Security Controls

* Token encryption at rest using platform-wide secure keys
* API calls routed through internal proxy service (not directly from frontend)
* Tenant isolation by verifying brand\_id on each API access
* Superadmin controls access toggles per brand:
  + Enable Facebook integration
  + Enable IG DM access
  + Enable Inbox automation
* Logs per token usage: method, response, latency, user

## 🔄 Integration Points

* Inbox Module → Meta token for FB/IG replies
* Campaign Module → FB Post auto-generation
* Gamification → Tracks comment-based campaigns
* Blog Engine → Auto-publishing via Facebook Page
* Auto Poster Scheduler → Uses Page/IG token to post at set times

## 📊 Admin Dashboard View

* List of all brands using Meta App
* Pages/IG profiles connected
* Token health (valid, expired, expiring soon)
* Last used, last refreshed, next refresh
* Actions: Revoke / Reauth / Resync Scopes

## 🔧 Developer Notes

### Database Tables:

* brand\_meta\_tokens: brand\_id, page\_id, access\_token, scopes, expires\_at, last\_used, token\_status
* meta\_token\_logs: brand\_id, method, timestamp, response\_code, duration

### Services:

* MetaOAuthService
* MetaTokenRefreshWorker
* FBApiProxyHandler

## 📘 Sample Prompt for Codex Integration:

“Build a secure OAuth handler for Facebook that saves access tokens for a multi-tenant app. Encrypt tokens, map them to brand\_id, store scope and expiry. Allow secure token refresh and proxy all FB API calls through a backend handler. Add admin dashboard to manage all tokens by tenant.”

✅ Module finalized. Let me know if you’d like this integrated with the Inbox module next or proceed with the next pending module.