

Joyo AI · Google Ads API Design Document

Company: Joyo AI

Website: <https://joyo.marketing>

Application URL: <https://app.joyo.marketing>

Contact Email: danieldand61@gmail.com

Manager Account (MCC) ID: 870-793-3264

Date: February 2026

1. Company Overview

Joyo AI is a SaaS marketing automation platform designed for small and medium businesses. The platform helps businesses manage their social media presence, generate AI-powered marketing content, and manage Google Ads campaigns · all from a single dashboard.

Our target users are small business owners and marketing managers who need an easy-to-use tool to run and monitor their Google Ads campaigns without requiring deep advertising expertise.

2. Purpose of Google Ads API Integration

We integrate with the Google Ads API to provide our users with the following capabilities:

- 1 **Campaign Reporting & Analytics** · Display campaign performance metrics (impressions, clicks, CTR, conversions, cost, average CPC) in a unified dashboard alongside social media analytics.
 - 2 **Responsive Search Ad (RSA) Creation** · Our AI generates optimized ad copy (headlines and descriptions) based on the user's business profile and website content. These are then pushed to Google Ads as Responsive Search Ads via the API.
 - 3 **Sitelink Extensions Management** · Users can add sitelink extensions to their campaigns through our platform interface.
 - 4 **Campaign & Ad Group Browsing** · Users can view their campaign structure (campaigns, ad groups) to select where to create new ads.
-

3. User Authentication Flow

We use the standard OAuth 2.0 authorization code flow:

Step 1: User clicks "Connect Google Ads" in Joyo AI Settings page

Step 2: User is redirected to Google's OAuth consent screen

URL: <https://accounts.google.com/o/oauth2/auth>

Scope: <https://www.googleapis.com/auth/adwords>

Access type: offline (to obtain refresh_token)

Prompt: consent

Step 3: User grants access and is redirected back to our callback

URL

Callback:

<https://app.joyo.marketing/auth/google-ads/callback>

Step 4: Our backend exchanges the authorization code for a refresh token

Endpoint: <https://oauth2.googleapis.com/token>

Step 5: User enters their Google Ads Customer ID (e.g., 871-729-2842)

Step 6: We verify the connection by fetching campaigns via the API
Step 7: Refresh token and Customer ID are stored securely in our database

Token Storage: Refresh tokens are stored in Supabase (PostgreSQL) with encryption at rest. Access tokens are obtained on-demand using the refresh token and are never persisted.

4. API Services & Methods Used

4.1 Read Operations (Reporting)

| Service | Method | Purpose | |-----|-----|-----| | GoogleAdsService | SearchStream | Fetch campaign list with performance metrics | | GoogleAdsService | SearchStream | Fetch ad groups for a specific campaign |

Example GAQL Query • Campaign Performance:

```
SELECT
    campaign.id,
    campaign.name,
    campaign.status,
    campaign.advertising_channel_type,
    metrics.impressions,
    metrics.clicks,
    metrics.cost_micros,
    metrics.conversions,
    metrics.ctr,
    metrics.average_cpc
FROM campaign
WHERE campaign.status != 'REMOVED'
    AND segments.date DURING LAST_30_DAYS
ORDER BY campaign.name
```

4.2 Write Operations (Ad Management)

| Service | Method | Purpose | |-----|-----|-----| | AdGroupAdService | MutateAdGroupAds | Create Responsive Search Ads | | ExtensionFeedItemService | MutateExtensionFeedItems | Add sitelink extensions to campaigns |

RSA Creation Details:

- Headlines: 3-15 per ad (max 30 characters each)
- Descriptions: 2-4 per ad (max 90 characters each)
- New ads are created in PAUSED status for user review before activation
- Input validation is performed before API calls

5. System Architecture

```
.....
.                               End User (Browser)                               .
.                               https://app.joyo.marketing                         .
.....
. HTTPS
```

<div> <div>Frontend (React / Vite)</div> <div>Hosted on Vercel</div> </div>	
<div> <div>Dashboard</div> <div>(metrics)</div> </div>	<div> <div>Ad Generator</div> <div>(RSA create)</div> </div>
<div>Google Ads Settings</div> <div>(OAuth connect/status)</div>	
REST API (JSON)	
<div> <div>Backend (Python / FastAPI)</div> <div>Hosted on Railway</div> </div>	
<div>Auth Middleware</div> <div>(JWT validation)</div>	<div>Google Ads Router</div> <div>/api/google-ads/*</div>
	GET /oauth/authorize
	GET /oauth/callback
	POST /oauth/complete
	GET /campaigns
	GET /campaigns/{id}/ad-groups
	POST /create-rsa
	POST /add-sitelinks
	GET /status
	DELETE /disconnect
<div>GoogleAdsService</div> <div>(API Client)</div>	
google-ads-python client library	
<div>Google Ads API (v19)</div> <div>ads.googleapis.com</div>	
Data Store:	
Supabase (PostgreSQL)	
Table: google_ads_connections	
...	user_id (FK)
...	customer_id (Google Ads account ID)
...	refresh_token (encrypted at rest)
...	status (active / inactive)

6. Rate Limiting & Error Handling

- All API calls are made **server-side only** from our backend; the frontend never contacts Google Ads API directly.
 - We implement exponential backoff for transient errors (HTTP 429, 500, 503).
 - API quota usage is monitored via logging.
 - With Basic Access (15,000 operations/day), our current user base is well within limits.
 - Each user action (view campaigns, create ad) triggers a minimal number of API calls (typically 1-2).
-

7. Data Handling & Privacy

- **No raw Google Ads data is shared with third parties.**
 - Refresh tokens are stored encrypted at rest in Supabase.
 - Access tokens are ephemeral · requested on-demand and never persisted.
 - Users can disconnect their Google Ads account at any time, which deactivates the stored credentials.
 - Campaign performance data is fetched on-demand and displayed in real-time; we do not bulk-export or cache campaign data long-term.
 - Our platform complies with Google's API Terms of Service and Required Minimum Functionality policies.
-

8. User Access & Permissions

- **External users (clients)** access Google Ads features through our platform at app.joyo.marketing.
 - Each user authenticates with their own Google account via OAuth 2.0.
 - Users can only access their own Google Ads data · multi-tenancy is enforced via `user_id` in all database queries.
 - Admin users of Joyo AI do not have access to individual client Google Ads data.
-

9. Campaign Types Supported

Currently, our platform supports **Search campaigns** only:

- Viewing Search campaign performance metrics
- Creating Responsive Search Ads within existing Search campaigns
- Adding sitelink extensions to Search campaigns

We plan to expand support to additional campaign types (Performance Max, Display) in future releases.

10. Technology Stack

| Component | Technology | |-----|-----| | Frontend | React, TypeScript, Vite, Tailwind CSS | | Backend | Python 3.11, FastAPI | | Google Ads Client | google-ads-python (official client library) | | Database | Supabase (PostgreSQL) | | Hosting (Frontend) | Vercel | | Hosting (Backend) | Railway | | Authentication | Supabase Auth (JWT) + Google OAuth 2.0 |