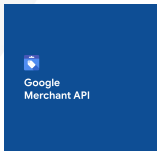
magmodules



Google Cloud Service Account Setup

Google Shopping - Merchant API for Magento 2

This guide explains how to create a Google Cloud service account and generate the JSON key required for the Google Shopping API extension to authenticate with Google Merchant Center.

What is Google Merchant Center?

Google Merchant Center is Google's platform for managing your product data. When you want your products to appear in Google Shopping, Google Search, or other Google services, you upload your product information to Merchant Center.

Traditionally, merchants upload product data via XML feeds (scheduled fetch or SFTP upload). The Google Shopping API extension takes a different approach: it syncs your products directly via the **Google Merchant API**, enabling real-time updates whenever your product data changes in Magento.

Why Do You Need a Service Account?

There are two ways to authenticate with Google APIs:

1. **OAuth 2.0 (User Authentication)** - Requires a user to log in and grant permissions. Tokens expire and need manual refresh. Not suitable for automated server-to-server communication.

2. **Service Account (Application Authentication)** - A special Google account that represents your application, not a person. It authenticates automatically using a private key, making it perfect for background processes like cron jobs and queue workers.

The Google Shopping API extension runs synchronization in the background without user interaction. A service account allows the extension to authenticate with Google 24/7, automatically refreshing tokens as needed.

In short: Service accounts enable your Magento store to communicate with Google Merchant Center autonomously, without requiring anyone to log in.

What You'll Create

By the end of this guide, you'll have:

1. **A Google Cloud project** - A container for your Google Cloud resources
2. **The Merchant API enabled** - Permission to use Google's product sync API
3. **A service account** - An identity for your Magento application
4. **A JSON key file** - The credentials your extension uses to authenticate

5. **Merchant Center access** - The service account linked to your products

Prerequisites

Before you begin, make sure you have:

- A Google account with access to Google Cloud Console
- A Google Merchant Center account with products
- Admin access to the Merchant Center
- Admin access to your Magento installation

Step 1: Create a Google Cloud Project

Google Cloud projects organize your resources and APIs. If you already have a project you'd like to use, you can skip to Step 2.

1. Go to Google Cloud Console
2. You'll see a Welcome screen with "Create or select a project" link
3. Click **Select a project** (top left) or the "Create or select a project" link
4. In the dialog that opens:
    - Select an existing project from the list, or
    - Click **New Project** in the top right of the dialog
5. If creating a new project:
    - Enter a project name (e.g., "Magento Google Shopping")
    - The **Project ID** is auto-generated (you can click Edit to customize it)
    - **Parent resource**: Leave as "No organization" - this is fine for most setups
    - Click **Create**
6. A notification appears in the top right - click **Select Project** to switch to your new project
7. Note your **Project ID** - you'll need this later

Step 2: Enable the Merchant API

Google Cloud projects don't have access to APIs by default. You need to explicitly enable each API you want to use.

The **Merchant API** is Google's current API for managing products in Merchant Center. It replaced the older "Content API for Shopping" which is being deprecated.

1. Use the search bar at the top of Google Cloud Console
2. Type **"Merchant API"**
3. Click **Merchant API** under "Marketplace" in the results
4. Click **Enable**

After enabling, you'll see the API details page with Status: **Enabled**.

Note: Don't use "Content API for Shopping" - that's the deprecated older API. The correct one is **Merchant API** (service name: `merchantapi.googleapis.com`).

Step 3: Create a Service Account

A service account is like a robot user - it has an email address and credentials, but it's controlled by your application rather than a person.

1. From the API details page, click **Create credentials** (top right)
2. On the "Credential Type" screen:
   - **Which API are you using?** → Merchant API (pre-selected)
   - **What data will you be accessing?** → Select **Application data**
3. Click **Next**
4. Fill in the service account details:

Field	Value
Service account name	`magento-google-shopping` (or your preference)
Service account ID	Auto-generated from name
Description	`Magento Google Shopping sync` (optional)

5. Click **Create and Continue**
6. **Permissions (optional)**: Skip this step - click **Continue**
7. **Principals with access (optional)**: Skip this step - click **Done**

The service account is now created. Its email address will look like: `magento-google-shopping@your-project-id.iam.gserviceaccount.com`

Step 4: Generate the JSON Key

The JSON key file contains your service account's private key. The extension uses this to prove its identity to Google.

1. Go to **APIs & Services** → **Credentials** in the left sidebar
2. Scroll down to the **Service Accounts** section
3. Click on your service account email (e.g., `magento-google-shopping@your-project.iam.gserviceaccount.com`)
4. Click the **Keys** tab at the top
5. Click **Add key** → **Create new key**
6. Select **JSON** as the key type
7. Click **Create**

The JSON key file will automatically download to your computer (check your Downloads folder).

After creation, you'll see the key listed with Status: **Active**.

Important:

- Store this file securely - it provides full access to your service account
- You can only download the key once - if you lose it, you'll need to create a new one
- If the key is compromised, delete it immediately and create a new one

Step 5: Link Service Account to Merchant Center

The service account exists in Google Cloud, but it doesn't automatically have access to your Merchant Center. You need to explicitly grant it permission.

Think of it like adding a new employee to your team - they need to be given access to the systems they'll work with.

1. Go to Google Merchant Center
2. Click **Settings** (gear icon) → **Account access**
3. Click **+ Add user**
4. Enter the service account email address:
   - Format: `your-service-account-name@your-project-id.iam.gserviceaccount.com`
   - Example: `magento-google-shopping@vast-cogency-486308-k7.iam.gserviceaccount.com`
5. Set access level to **Admin**
6. Click **Add user**

After adding, the service account will show as **Verified** in the user list.

Why Admin access? The extension needs to create, update, and delete products. Standard access only allows viewing products, which isn't sufficient for synchronization.

Step 6: Create an API Data Source

Before the extension can sync products, you need to create a data source in Merchant Center that will receive the API data.

1. Go to Google Merchant Center
2. Navigate to **Settings** → **Data Sources**
3. Click **Add product source**
4. Select **Merchant API** as the source type
5. Configure the data source:
   - **Target country**: Select your target market (e.g., Netherlands, United States)
   - **Language**: Select the content language (e.g., Dutch, English)
   - **Name**: Give it a recognizable name (e.g., "Magento API Sync")
6. Click **Create**

After creation, note the **Data Source ID**. You'll find it on the data source details page. You'll need this for the Magento configuration:

- Example: `10610267509`

Step 7: Configure the Magento Extension

Now you have everything needed to configure the extension.

1. Open the JSON key file in a text editor
2. Copy the entire contents (including the curly braces)
3. In Magento Admin, go to **Stores** → **Configuration** → **Magmodules** → **Google Shopping API**
4. Configure the settings:

Field	Value
Enable Module	Yes
Merchant ID	Your Merchant Center ID (e.g., `5544859096`)
API Data Source	The Data Source ID from Step 6 (e.g., `10610267509`)
Feed Label (Target Country)	Country code (e.g., `NL`, `US`, `DE`)
Content Language	Language code (e.g., `nl`, `en`, `de`)
Service Account JSON	Paste the entire JSON key contents

5. Click **Save Config**
6. Click **Test Connection** to verify everything works

If successful, you'll see:

- A green "Successfully connected to Google Merchant Center" message
- Your Merchant ID, Data Source ID, and Feed Label confirmed

If there's an error, check the Troubleshooting section below.

How Authentication Works

Understanding the authentication flow helps with troubleshooting:

1. **Extension reads JSON key** - Extracts the private key and service account email
2. **Creates a JWT token** - A signed token requesting access to the Merchant API
3. **Exchanges JWT for access token** - Google verifies the signature and returns an access token
4. **Uses access token for API calls** - The token is valid for about 1 hour
5. **Automatic refresh** - The extension automatically creates new tokens as needed

This all happens in the background - you don't need to do anything after initial setup.

JSON Key File Contents

For reference, here's what the downloaded JSON file contains:

```
{
  "type": "service_account",
  "project_id": "your-project-id",
  "private_key_id": "abc123...",
  "private_key": "-----BEGIN PRIVATE KEY-----\n...\n-----END PRIVATE KEY-----\n",
  "client_email": "your-service-account@your-project-id.iam.gserviceaccount.com",
  "client_id": "123456789",
  "auth_uri": "https://accounts.google.com/o/oauth2/auth",
  "token_uri": "https://oauth2.googleapis.com/token",
  "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
  "client_x509_cert_url": "https://www.googleapis.com/robot/v1/metadata/x509/..."
}
```

The extension primarily uses:

- `client_email` - The service account's identity

- `private_key` - Used to sign authentication requests

Troubleshooting

"Data Source is not configured"

- Create an API data source in Merchant Center (see Step 6)
- Enter the Data Source ID in the extension configuration (e.g., `10610267509`)

"Permission denied" or 403 errors

- Verify the service account email is added to Merchant Center
- Ensure the access level is **Admin**
- Wait a few minutes after adding - permissions can take time to propagate

"API not enabled" errors

- Go to Google Cloud Console → APIs & Services → Library
- Search for "Merchant API" and ensure it's enabled
- Check you're in the correct project

"Invalid credentials" or authentication errors

- Verify you pasted the complete JSON (including curly braces)
- Check for extra whitespace or line breaks
- Re-download the key and try again

"Merchant ID not found"

- Verify your Merchant ID in Merchant Center (Settings → Account info)
- Ensure the service account has access to the correct Merchant Center

Test Connection fails

Run the CLI test command for more details:

```
bin/magento googleshopping:api:test-connection --store=1
```

Security Best Practices

The JSON key file is sensitive - treat it like a password.

1. **Never commit the JSON key to version control** - Add it to `.gitignore`
2. **Restrict key access** - Only share with team members who need it
3. **Rotate keys periodically** - Delete old keys and create new ones every few months
4. **Use separate keys per environment** - Don't use production keys in development
5. **Monitor usage** - Check Google Cloud Console for unusual activity

Managing Keys

Deleting a Key

If a key is compromised or no longer needed:

1. Go to **IAM & Admin → Service Accounts**
2. Click on the service account
3. Go to **Keys** tab
4. Click the trash icon next to the key
5. Confirm deletion

The key is immediately invalidated - any application using it will stop working.

Creating Additional Keys

You can have multiple keys for the same service account. This is useful for:

- Rotating keys without downtime
- Different keys for different environments
- Temporary access for debugging

Multiple Merchant Centers

If you manage multiple Merchant Centers (e.g., for different countries or brands):

1. Create one service account
2. Add the service account email to each Merchant Center
3. Use the same JSON key in Magento
4. Configure different Merchant IDs per store view

The service account can access any Merchant Center it's been added to.

Further Reading

- Google Cloud IAM Documentation - Service accounts overview
- Creating Service Account Keys - Official Google documentation
- Merchant API Authentication - API auth guide
- Merchant API Overview - Complete API documentation

Need More Help?

Documentation:

- All Help Articles - Complete documentation overview

Support:

- Contact Support - Get help from our team

For a complete overview of features and configuration options, see the Google Shopping - Merchant API extension on magmodules.eu

All articles for Google Shopping - Merchant API

Installation

1	Installation using Composer (recommended)

Configuration

1	**Google Cloud Service Account Setup (current)**

Background

1	Google Shopping Merchant API Sync Explained
2	Using Feed and API Together

