

Google Photos API Integration Guide (PHP)

Autor: Dino Karl & Philipp | Stand: 2026

Diese Anleitung beschreibt, wie man eine PHP-Anwendung (ohne Composer/Bibliotheken) mit der Google Photos API verbindet, um Bilder in ein privates Album hochzuladen.

Teil 1: Google Cloud Console (Der bürokratische Teil)

Bevor eine Zeile Code geschrieben wird, braucht man die "Schlüssel".

1. Projekt erstellen:

- Geh auf console.cloud.google.com.
- Erstelle ein neues Projekt (oder nutze ein vorhandenes).

2. API aktivieren:

- Suche oben nach **"Google Photos Library API"**.
- Klicke auf **"Aktivieren"**.

3. OAuth Consent Screen (Zustimmungsbildschirm):

- Geh zu **"APIs & Dienste"** -> **"OAuth-Zustimmungsbildschirm"**.
- Wähle **User Type: Extern**.
- Fülle App-Name und E-Mail-Adressen aus.
- **WICHTIG:** Füge unter **"Test Users"** deine eigene Google-Mail-Adresse hinzu! (Sonst Error 403: Access Denied).

4. Credentials (Zugangsdaten):

- Geh zu **"Zugangsdaten"** -> **"+ ZUGANGSDATEN ERSTELLEN"** -> **"OAuth-Client-ID"**.
- **Typ:** Webanwendung.
- **Autorisierte Weiterleitungs-URIs:** Die URL, wo dein Skript läuft.
 - Lokal: `http://localhost/mein-projekt/auth.php`
 - Live: `https://deine-domain.de/mein-projekt/auth.php`
- Klicke auf Erstellen und lade die JSON herunter.
- Benenne sie um in: `client_secret.json`.

Teil 2: Die Dateien (Der Code)

Du brauchst 5 Dateien in deinem Projektordner.

1. `client_secret.json`

- Die Datei aus Teil 1. Darf **niemals** öffentlich zugänglich sein!

2. .htaccess (Sicherheit)

Schützt deine sensiblen JSON-Dateien vor Zugriff durch Browser.

```
<FilesMatch "\.(json|txt)$">
    Order Deny,Allow
    Deny from all
</FilesMatch>
```

3. google_helper.php (Die Logik-Klasse)

Eine Klasse, die Auth, Refresh-Tokens und den Upload handhabt. (Code siehe Projektarchiv oder unten).

- **Wichtigste Methoden:** `authenticate($code)` , `uploadImage($binary, $desc)` .
- **Wichtig:** Speichert Tokens in `tokens.json` .

4. auth.php (Der Login)

Einmalig aufzurufen, um die Verbindung herzustellen.

```
<?php
require_once 'google_helper.php';
$google = new GoogleHelper('client_secret.json', '[https://deine-url.de/auth.php]

if (isset($_GET['code'])) {
    if ($google->authenticate($_GET['code'])) echo "Verbunden!";
} else {
    echo '<a href="' . $google->getAuthUrl() . '">Login mit Google</a>';
}
?>
```

5. upload_handler.php (Der Empfänger)

Nimmt Bilder per POST (AJAX) entgegen und ruft den Helper auf.

```
<?php
require_once 'google_helper.php';
// ... JSON Input decoden ...
// ... Base64 Bild decoden ...
$google = new GoogleHelper(...);
$google->uploadImage($binaryData, "Beschreibung");
?>
```

Teil 3: Der Workflow

Einmalige Einrichtung (Der "Handshake")

1. Lade alle Dateien auf den Server.
2. Öffne `auth.php` im Browser.
3. Logge dich mit deinem Google-Konto (dem "Test User") ein.
4. Akzeptiere die Warnung ("App isn't verified") -> Erweitert -> Trotzdem fortfahren.
5. Erlaube Zugriff auf Google Photos.
6. **Ergebnis:** Eine Datei `tokens.json` wird auf dem Server erstellt. Ab jetzt bist du "drin".

Tägliche Nutzung (Automatisch)

- Dein Skript nutzt den `refresh_token` aus der `tokens.json`, um sich immer wieder neue `access_tokens` zu holen.
- Du musst dich nie wieder manuell einloggen, solange die `tokens.json` existiert.

⚠ Häufige Stolpersteine

- **Redirect URI Mismatch:** Die URL in der Google Cloud Console muss **exakt** (inkl. `http/https` und Slash am Ende) mit der in deinem Code übereinstimmen.
- **Error 403 (Access Denied):** Du hast vergessen, deine E-Mail bei "Test Users" in der Cloud Console einzutragen.
- **Bild erscheint nicht:** Prüfe, ob das Album "Dino Wissenskarten" (oder dein App-Name) erstellt wurde. Der Upload landet im Album, nicht immer ganz oben in der Timeline.
- **Token abgelaufen:** Lösche `tokens.json` und rufe `auth.php` erneut auf, um einen frischen Reset zu machen.

Happy Coding, alte Säge! 🦖