

MANUALE TECNICO V1.0

MCP Google Workspace Server

Guida completa all'installazione, configurazione e utilizzo standalone da script Python su Windows

Sistema	Python	Versione	Data
Windows 10/11	3.11+	1.0.0	Febbraio 2026

Indice

1. Panoramica del sistema
2. Requisiti e installazione
3. Configurazione credenziali Google
4. Configurazione del file .env
5. Prima esecuzione e autorizzazione OAuth2
6. Come usare i tool da script Python
7. Riferimento completo dei tool
8. Errori comuni e soluzioni
9. Note su rate limit e sicurezza

Uso standalone

Questo manuale descrive l'utilizzo direttamente da script Python, senza dipendenze da Claude Desktop o altre applicazioni AI.

1. Panoramica del Sistema

Il progetto espone una libreria Python che permette di interagire con **Google Drive**, **Google Docs** e **Google Sheets** tramite le API ufficiali Google. Usabile direttamente da qualsiasi script Python.

Struttura cartelle

```
mcp-google-workspace/  
├─ connectors/google/  
│   ├── auth.py      ← gestione credenziali OAuth2 / Service Account  
│   ├── drive.py     ← Google Drive API v3  
│   ├── docs.py      ← Google Docs API v1  
│   └─ sheets.py     ← Google Sheets API v4  
├─ tools/  
│   ├── _base.py     ← helper output ok()/err()  
│   ├── drive_tools.py ← funzioni Drive pronte all'uso  
│   ├── docs_tools.py ← funzioni Docs pronte all'uso  
│   └─ sheets_tools.py ← funzioni Sheets pronte all'uso  
├─ config/  
│   └─ .env.example ← template configurazione  
├─ .env              ← credenziali (crei tu, NON su git)  
└─ .token_cache.json ← token OAuth2 (generato automaticamente)
```

Punto chiave

Una volta configurato, importa i moduli tools/ nel tuo script Python e chiama le funzioni direttamente. Non serve avviare nessun server separato.

2. Requisiti e Installazione

Installazione smart

Il setup_project.py va copiato nella cartella dove vuoi installare il progetto. Configura automaticamente tutti i path — non devi modificare nulla a mano.

Passo 1 — Scegli la cartella di installazione

Crea una cartella dedicata dove preferisci, ad esempio:

```
C:\Users\TUO_NOME\Progetti\mcp-google-workspace\
```

Copia il file `setup_project.py` dentro questa cartella.

Evita il Desktop

Non installare sul Desktop — si riempie, e' visibile a tutti e su alcuni PC viene sincronizzato con OneDrive (rischio di caricare le credenziali sul cloud).

Passo 2 — Esegui il setup

Apri il terminale nella cartella di installazione e digita:

```
python setup_project.py
```

Lo script crea tutta la struttura **nella cartella corrente** e ti chiede di copiare `client_secret.json` prima di completare.

Passo 3 — Copia client_secret.json

Quando lo script si ferma e chiede il file, copia `client_secret.json` (scaricato da Google Cloud Console) **nella stessa cartella di installazione**. Poi premi INVIO.

Lo script completa la configurazione e crea automaticamente:

- **.env** con tutti i path gia' configurati correttamente
- **agente_esempio.py** con i path precompilati, pronto all'uso

Passo 4 — Installa dipendenze

```
pip install -e ".[dev]"
```

Struttura finale nella cartella di installazione

```
mcp-google-workspace\  
├─ connectors\      ← connettori Google APIs  
├─ tools\          ← funzioni pronte all'uso  
├─ server\         ← bootstrap MCP (opzionale)  
├─ tests\          ← test automatici  
├─ config\         ← settings.py  
├─ .env            ← path configurati automaticamente  
├─ client_secret.json ← credenziali OAuth2 (messo da te)  
├─ .token_cache.json ← token OAuth2 (generato al primo avvio)  
├─ agente_esempio.py ← template pronto con path precompilati  
└─ setup_project.py ← questo script
```

Nessun path da modificare

Il file `.env` e `agente_esempio.py` vengono scritti con i percorsi assoluti della cartella corrente. Puoi iniziare subito senza modificare nulla.

Dipendenze installate

Libreria	Scopo
<code>google-api-python-client</code>	Client ufficiale Google APIs
<code>google-auth</code> + <code>oauthlib</code>	Autenticazione OAuth2 e Service Account
<code>tenacity</code>	Retry automatico su errori 429/5xx
<code>python-dotenv</code>	Caricamento file <code>.env</code>

3. Configurazione Credenziali Google

Quale metodo?

Per uso desktop/personale usa OAuth2. Per server automatici usa Service Account.

Metodo OAuth2 — Passo per passo

1. Crea progetto Google Cloud

Vai su **console.cloud.google.com** → menu tendina in alto → **Nuovo progetto** → nome: `mcp-workspace` → Crea.

2. Abilita le 3 API

Vai su **API e servizi** → **Libreria** e abilita: **Google Drive API**, **Google Docs API**, **Google Sheets API**.

3. Configura schermata consenso OAuth

API e servizi → **Credenziali** → **Configura schermata di consenso** → scegli **Esterno** → inserisci nome app e email → vai su **Pubblico** nel menu laterale → aggiungi la tua email come **utente di prova**.

4. Crea credenziali OAuth2

+ **Crea credenziali** → **ID client OAuth 2.0** → **Applicazione desktop** → dai un nome → Crea → clicca **Scarica JSON** → salva come `C:\Users\TUO_NOME\Desktop\client_secret.json`

Importante

Salva subito il JSON — Google non lo mostrerà di nuovo. Non condividere mai questo file.

4. Configurazione del File .env

Apri il file con:

```
notepad .env
```

Configurazione OAuth2 (consigliata)

```
# NON committare .env su git!

# Service Account (lascia commentato se usi OAuth2)
# GOOGLE_APPLICATION_CREDENTIALS=C:\percorso\service_account.json

# OAuth2 — decommentare e compilare con i tuoi percorsi
OAUTH_CLIENT_SECRET_FILE=C:\Users\TUO_NOME\Desktop\client_secret.json
TOKEN_CACHE_PATH=C:\Users\TUO_NOME\Desktop\mcp-google-workspace\.token_cache.json

LOG_LEVEL=INFO
```

Backslash su Windows

I percorsi Windows usano \ (backslash). Nel file .env vanno scritti così:

```
C:\Users\anton\Desktop\file.json
```

Sicurezza

Non caricare mai .env su GitHub, Google Drive o altri servizi. Non condividere client_secret.json o .token_cache.json.

5. Prima Esecuzione e Autorizzazione OAuth2

La prima volta che esegui uno script, il browser si apre automaticamente.

1. "Google non ha verificato questa app"

Clicca **Continua** — sei tu lo sviluppatore, e' normale per app in fase di test.

2. Seleziona i permessi

Clicca **Seleziona tutto** (Drive + Docs + Sheets) poi **Continua**.

3. Token salvato

Il file `.token_cache.json` viene creato automaticamente. Dalle prossime esecuzioni non verra' piu' chiesta l'autorizzazione.

6. Come Usare i Tool da Script Python

Template base — incolla sempre queste righe all'inizio

```
import sys, os
sys.path.insert(0, r"C:\Users\TUO_NOME\Desktop\mcp-google-workspace")

from dotenv import load_dotenv
load_dotenv(r"C:\Users\TUO_NOME\Desktop\mcp-google-workspace\.env")

# Importa i tool che ti servono
from tools.drive_tools import _handle_search_files, _handle_create_doc
from tools.docs_tools import _handle_read_doc, _handle_append_doc
from tools.sheets_tools import _handle_read_range, _handle_update_range
```

Esempio 1 — Cercare file su Drive

```
result = _handle_search_files({
    "name": "Budget",
    "mime_type": "application/vnd.google-apps.spreadsheet",
    "page_size": 10
})
if result["status"] == "ok":
    for f in result["data"]["files"]:
        print(f["name"], f["id"])
```

Esempio 2 — Leggere un Google Doc

```
result = _handle_read_doc({"doc_id": "ID_DEL_DOCUMENTO"})
if result["status"] == "ok":
    print(result["data"]["title"])
    print(result["data"]["text"])
```

Esempio 3 — Scrivere su Google Sheets

```
result = _handle_update_range({
    "spreadsheet_id": "ID_DEL_FOGLIO",
    "range": "Foglio1!A1:C2",
    "values": [
        ["Nome", "Cognome", "Email"],
        ["Mario", "Rossi", "mario@email.it"]
    ]
})
```

Come trovare l'ID di un documento

Apri il documento nel browser. L'URL e': https://docs.google.com/document/d/ID_QUI/edit — copia la stringa lunga tra /d/ e /edit.

Struttura output di ogni tool

```
{"status": "ok", "data": { ... }, "errors": None}    # successo
{"status": "error", "data": None, "errors": "msg"}  # errore
```

7. Riferimento Completo dei Tool

Google Drive

Funzione	Parametri obbligatori	Descrizione
<code>_handle_search_files</code>	nessuno (tutti opzionali)	Cerca file per nome, mimeType, cartella, query
<code>_handle_upload_file</code>	local_path	Carica file locale su Drive
<code>_handle_download_file</code>	file_id, dest_path	Scarica file da Drive
<code>_handle_create_doc</code>	name	Crea nuovo Google Doc
<code>_handle_create_sheet</code>	name	Crea nuovo Google Spreadsheet

Google Docs

Funzione	Parametri obbligatori	Descrizione
----------	-----------------------	-------------

<code>_handle_read_doc</code>	<code>doc_id</code>	Legge titolo e testo completo
<code>_handle_append_doc</code>	<code>doc_id, text</code>	Aggiunge testo in fondo
<code>_handle_replace_doc</code>	<code>doc_id + (search/replacement oppure new_text)</code>	Sostituisce testo o sovrascrive tutto

Google Sheets

Funzione	Parametri obbligatori	Descrizione
<code>_handle_read_range</code>	<code>spreadsheet_id, range</code>	Legge celle. <code>as_dicts=True</code> per header→dict
<code>_handle_update_range</code>	<code>spreadsheet_id, range, values</code>	Scrive valori (sovrascrive)
<code>_handle_append_rows</code>	<code>spreadsheet_id, values</code>	Aggiunge righe dopo l'ultimo dato
<code>_handle_clear_range</code>	<code>spreadsheet_id, range</code>	Svuota range
<code>_handle_list_sheets</code>	<code>spreadsheet_id</code>	Elenca tutti i fogli con ID numerico
<code>_handle_add_sheet</code>	<code>spreadsheet_id, title</code>	Aggiunge nuovo foglio (tab)
<code>_handle_rename_sheet</code>	<code>spreadsheet_id, sheet_id, new_title</code>	Rinomina foglio (usa <code>list_sheets</code> per l'ID)

MIME type per download file Google-native (export_mime)

Tipo	export_mime
Google Doc → testo	<code>text/plain</code>
Google Doc → Word	<code>application/vnd.openxmlformats-officedocument.wordprocessingml.document</code>
Google Sheet → CSV	<code>text/csv</code>
Google Sheet → Excel	<code>application/vnd.openxmlformats-officedocument.spreadsheetml.sheet</code>

8. Errori Comuni e Soluzioni

EnvironmentError: Imposta GOOGLE_APPLICATION_CREDENTIALS oppure OAUTH_CLIENT_SECRET_FILE

Causa: Il file .env non viene trovato.

Soluzione:

Usa il percorso assoluto in load_dotenv: `load_dotenv(r"C:\Users\TUO_NOME\Desktop\mcp-google-workspace\.env")`

NameError: name 'sys' is not defined

Causa: La riga import sys ha il # davanti (e' commentata).

Soluzione:

Rimuovi il # da `#import sys` → deve essere `import sys`

"cp" non e' riconosciuto come comando

Causa: Comando Linux su Windows.

Soluzione:

Usa `copy config\.env.example .env`

Google: "L'app non e' verificata"

Causa: Normale per app in fase di test.

Soluzione:

Clicca Continua — sei tu lo sviluppatore.

HttpError 403: The caller does not have permission

Causa: API non abilitate o file non condiviso con Service Account.

Soluzione:

Verifica che Drive API, Docs API e Sheets API siano abilitate su Google Cloud Console.

HttpError 429: Too Many Requests

Causa: Rate limit API Google superato.

Soluzione:

Il sistema fa retry automatico. Se persiste aggiungi `import time; time.sleep(1)` tra le chiamate.

Token scaduto / Invalid credentials

Causa: `.token_cache.json` corrotto o scaduto.

Soluzione:

Elimina il file `.token_cache.json` e riesegui — il browser si riaprirà per nuova autorizzazione.

%APPDATA% non funziona in PowerShell

Causa: Le variabili % % sono sintassi CMD, non PowerShell.

Soluzione:

In PowerShell usa `$env:APPDATA` invece di `%APPDATA%`

9. Rate Limit e Sicurezza

API	Limite	Gestione
Google Drive	12.000 req/min per utente	Retry automatico con backoff esponenziale
Google Docs	300 req/min per progetto	Raggruppa in batchUpdate
Google Sheets	300 req/min progetto / 60 per utente	Retry + batchUpdate

File da proteggere

File	Contiene	Regola
<code>.env</code>	Percorsi credenziali	Mai su GitHub o cloud
<code>client_secret.json</code>	Client ID e Secret OAuth2	Tratta come password
<code>service_account.json</code>	Chiave privata SA	Tratta come chiave privata
<code>.token_cache.json</code>	Token accesso Google	Mai condividere

Regola d'oro

Il .gitignore del progetto esclude automaticamente tutti questi file se usi Git. Non aggiungere mai eccezioni al .gitignore per questi file.

Da fare

- Raggruppa operazioni in batchUpdate
- Usa page_size adeguato
- Aggiungi sleep() tra operazioni massive
- Riutilizza il token dalla cache

Da evitare

- Loop veloci con centinaia di chiamate
- Creare nuove credenziali ogni volta
- Condividere .env o client_secret.json
- Caricare credenziali su GitHub