Workflow draft:

1. Have the workflow authenticate on a Google Drive (assume for the moment any batches will be using 10-12 GB at most and any pictures more than that will either queue for subsequent batch processing or maybe use a bigger Google Drive, etc.)

Using HEIF ISO 400 3024x4032 26mm f1.8, it is about 12 MP for 2 Mb per photo

10 GB can fit 5,000 photos. However, if Cloud Storage (rather than Google Drive) is used, it is limited to 5GB free, which is 2,500 images only.

1. Create a webhook event that triggers OCR scanning.
   1. Ala-carte: Any picture uploaded to call an OCR processor

or

* 1. A manual trigger so requests can be batched efficiently at the end of the day or according to the number of inventoried items (for the MVP, use a manual trigger inside the workflow or receive a chat event message in Gmail or WhatsApp, etc.)

1. Call the OCR API, possibly Google’s Document AI processing if there is a free tier. Still, it is best to use more straightforward services, like the pretrained computer vision ML **Cloud Vision API**, just to extract insights from images, documents, and videos.

Specify the desired data fields (brand, model, serial number, description) in the API request:

* 1. Pricing of Cloud Vision is well known.

<https://cloud.google.com/vision/pricing>

|  |  |
| --- | --- |
| [LABEL\_DETECTION](https://cloud.google.com/vision/docs/labels) | Add labels based on image content |
| [TEXT\_DETECTION](https://cloud.google.com/vision/docs/ocr) | Perform Optical Character Recognition (OCR) on text within the image. Text detection is optimized for areas of sparse text within a larger image. If the image is a document (PDF/TIFF), has dense text, or contains handwriting, use DOCUMENT\_TEXT\_DETECTION instead. |

The final 1000-unit block is prorated out of 1000 units.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | **Price per 1000 units** | | | |
| **First 1000 units/month** | **Units 1001 - 5,000,000 / month** | **Units 5,000,001 and higher / month** |  |
| Label Detection | Free | $1.50 | $1.00 |  |
| Text Detection | Free | $1.50 | $0.60 |  |

* 1. Pricing of Document AI is tricky. We do not need to extract structures and entities because the hardware label is a very well-known document. The cost of digitizing the test for the Enterprise Document OCR processor can be limited to $1.50 per 1000 pages.

*What constitutes a page depends on the file format.*

* + 1. *Images (JPEG/JPG, PNG, BMP, HEIF): Each image = 1 page*
    2. *PDF: Each page in the PDF = 1 page*
    3. *TIFF: Each image in the TIFF = 1 page*
    4. *Word (DOCX): Up to 3,000 characters = 1 page*
    5. *Excel (XLSX): Each tab = 1 page*
    6. *Powerpoint (PPTX): Each slide = 1 page*
    7. *HTML: Up to 3,000 characters = 1 page*
    8. *Parsed Documents: Up to 3,000 characters = 1 page*

1. Transform the extracted JSON fields and write back to Google Sheets (or AirTable whatever).
2. Use Error Handler to monitor.
3. This should be much cheaper than the GKE/google cloud version.

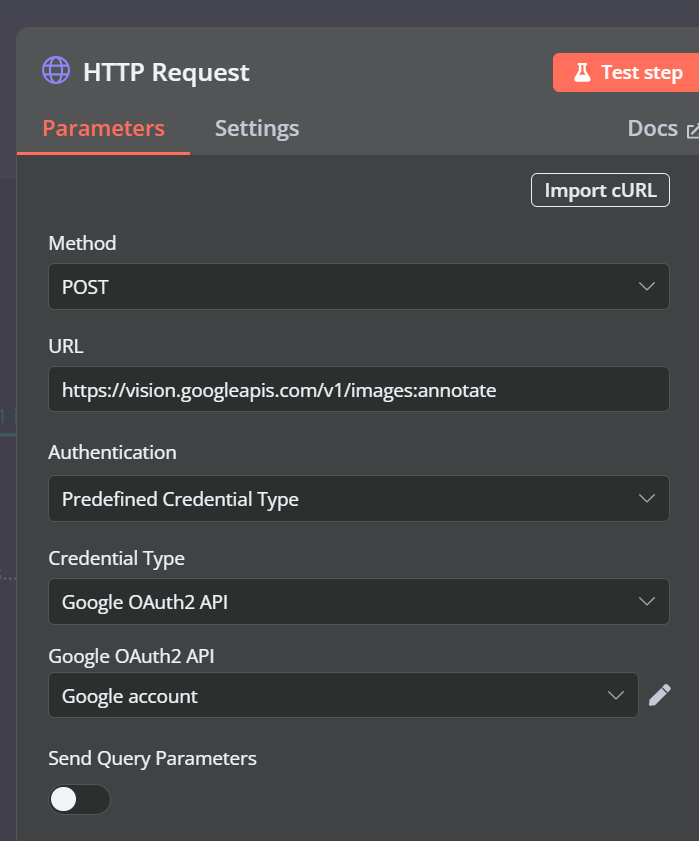
Make a custom HTTP Node request for google vision as there are no predefined HTTP integration in n8n

<https://cloud.google.com/vision/docs/ocr>

From original sample curl reference:

(there is an option to import cURL into HTTP node in n8n but this will overwrite existing definitions)

*curl -X POST -H "Authorization: Bearer $(gcloud auth application-default print-access-token)" -H "x-goog-user-project: play-452100" -H "Content-Type: application/json; charset=utf-8" https://vision.googleapis.com/v1/images:annotate -d @req2.json -o res2.json*



Notes:

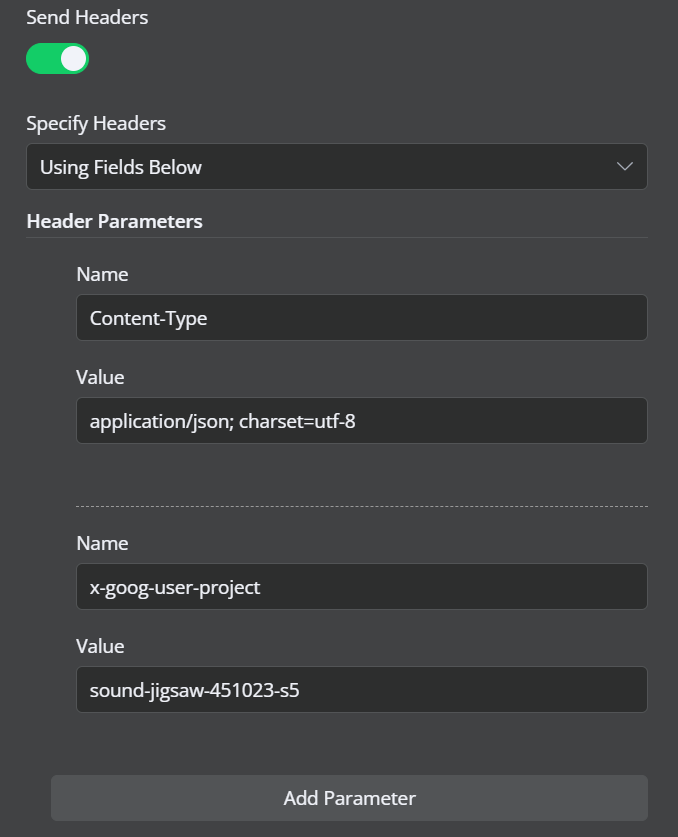
1. Use Predefined Credential Type and is usually simpler to use Google OAuth2 API unless the API service wants something else.
2. It is important when you validate the credential to specify the correct Scopes in the HTTP Credentials interface in n8n as needed by the particular API service e.g. for Google Vision these scopes:

<https://www.googleapis.com/auth/cloud-platform>

https://www.googleapis.com/auth/ cloud-vision

Refer to cloud.google.com/vision/docs/reference/rest/v1/images/annotate

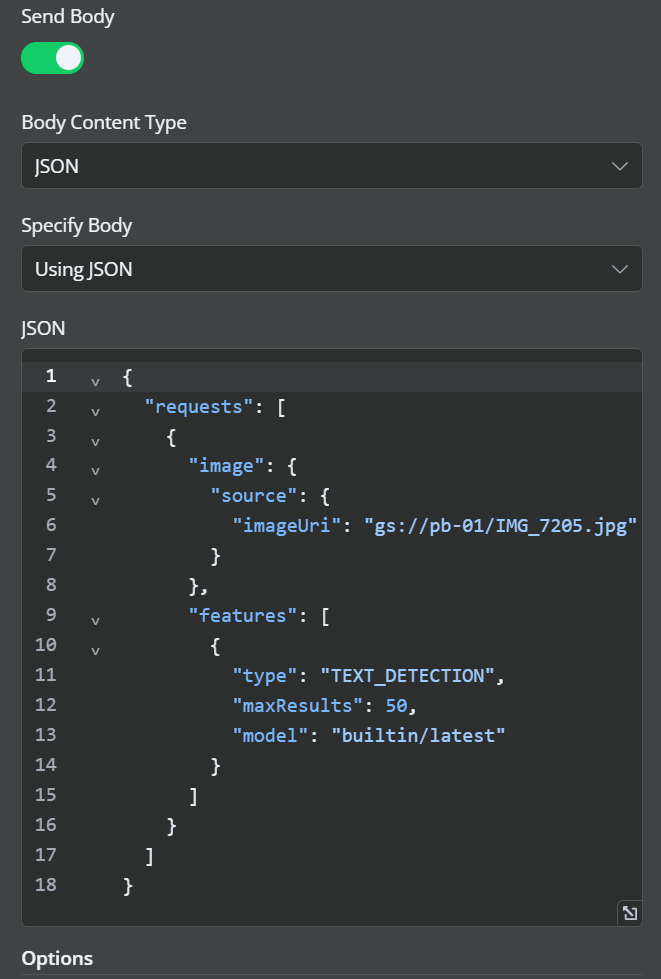
\*No method-level parameters, so do not send Query

****

**Header Parameters**

Project ID parameter i.e. x-goog-user-project is not really required because the OAuth credential is specific to the project so it will default to what that credential can access.

Content-Type has to specify the format as utf-8 as JSON really supports text and not binary.



**BODY**

In JSON, the imageUri is specifically the gs(google storage)/gsutil representation of the bucket and the file image (in base64) to analyze. An external URL is unreliable and is not recommended since the external storage may refuse googlevision or kill the access. So for production it is best to use cloud storage buckets within your control and administration.

**features** could be more complex but this sample use case is just looking for text OCR and then a post processing of the text

**Next Steps**

* Investigate one image annotation at a time

The sample above is online or synchronous request and immediately returns inline annotations to the user. Request with images:annotate limits images to <= 16,

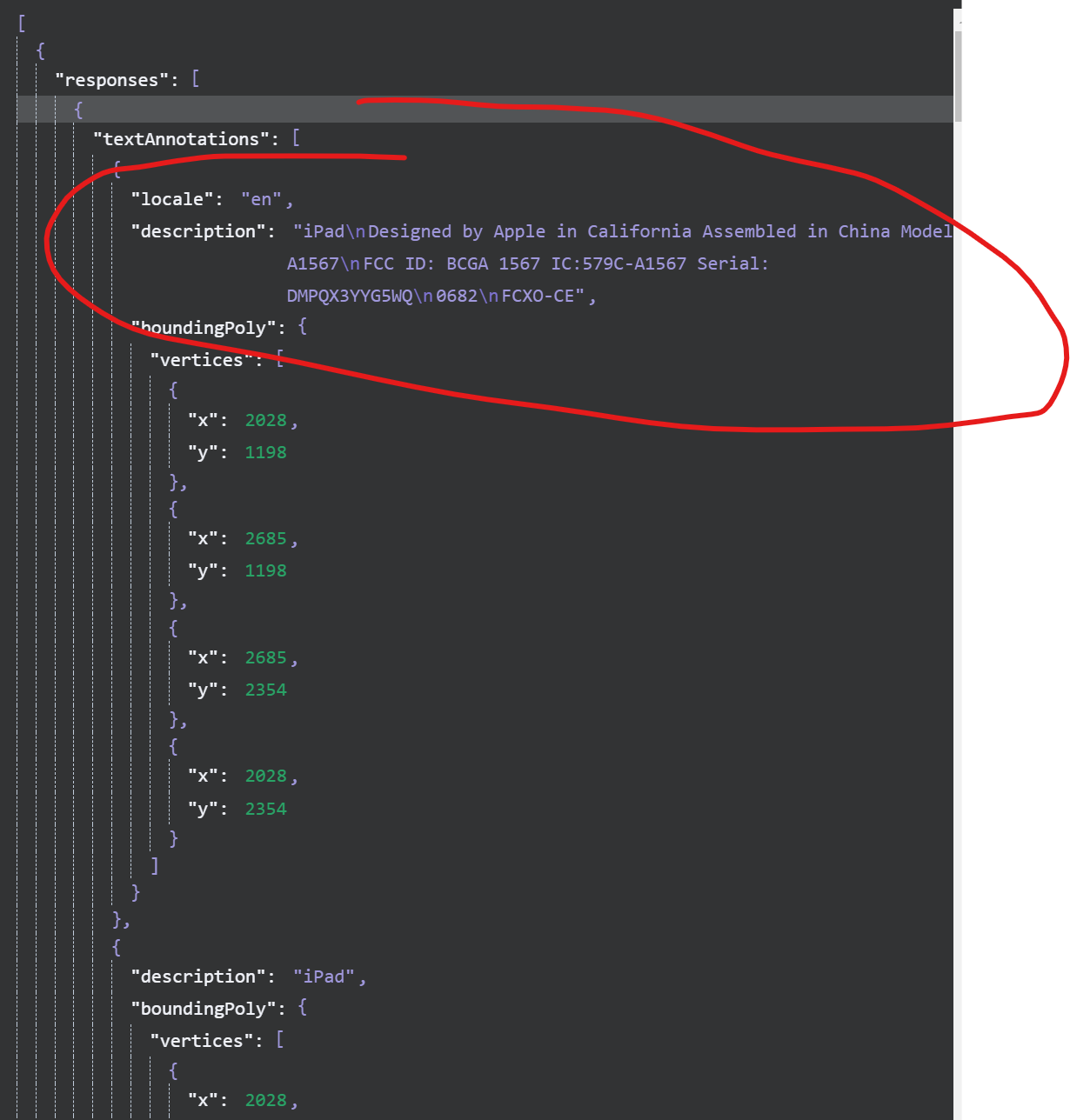
and files:annotate limits to a single file and small number of pages <=5.

* Investigate offline, batched images annotation and how to deal with response JSON

[*https://vision.googleapis.com/v1/images:asyncBatchAnnotate*](https://vision.googleapis.com/v1/images:asyncBatchAnnotate) *or files:astncBatchAnnotate*

and the resulting JSON is written in the same bucket/folder as the source images

for more details, see *cloud.google.com/vision/docs/batch*

**

The response can be a long JSON, with the summarized/consolidated description at the first item, to be parsed by yet another process for component information like brand, model, serial number using typical clues and pattern seen in most hardware and using regex thereafter.

But if the image is blurry or the area taken has bad quality and OCR fails, you will get this empty reply (although it is OK 200):

[

{

"responses": [

{}

]

}

]