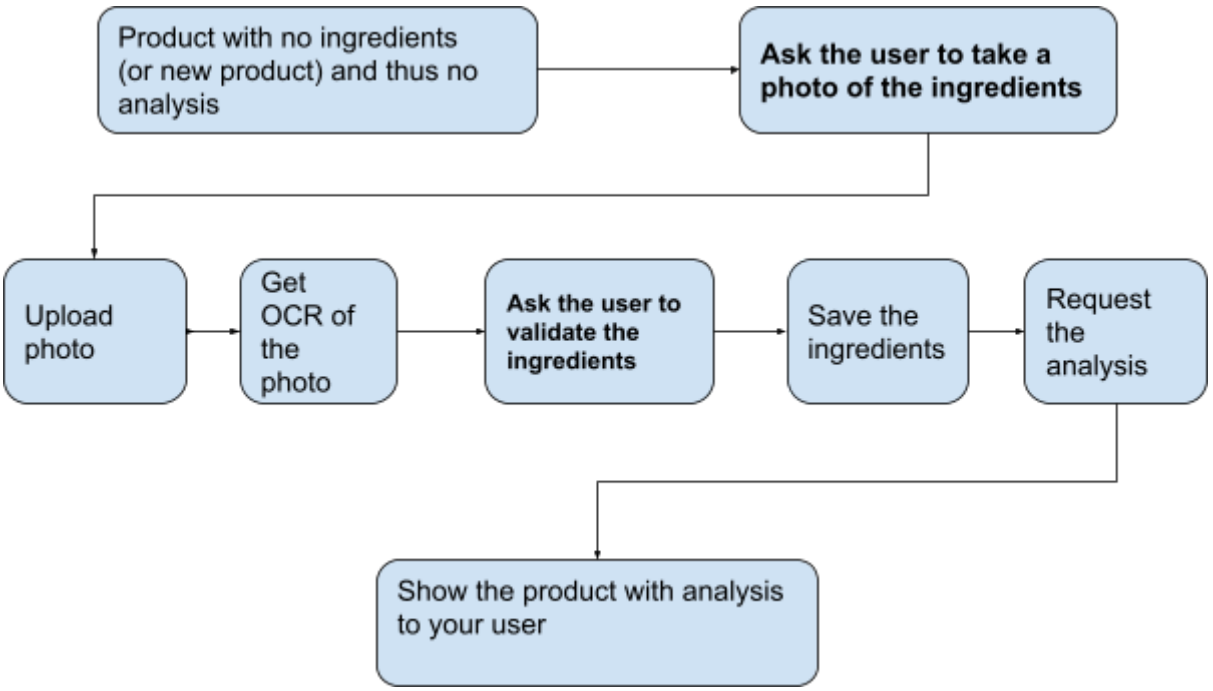
Ingredient related features for 3rd party apps

THIS IS A DRAFT

Introduction

- If you can't get the information on a specific product, you can get your user to send photos and data, that will then be processed by Open Food Facts AI and contributors to get the computed result you want to show them.
- You can implement the complete flow so that they get immediately the result with some effort on their side.
- That will ensure user satisfaction
- Most of the operations described below are implemented in the openfoodfacts-dart plugin, but as individual operations, not as a coherent pipe



Dart/Flutter package

- Get the status of the product
  - https://openfoodfacts.github.io/openfoodfacts-dart/utils_ImageHelper/ImageHelper-class.html
  - https://openfoodfacts.github.io/openfoodfacts-dart/model_ProductImage/ImageField-class.html
- Upload ingredient photo
  -
- Get OCR of the photo
  - https://openfoodfacts.github.io/openfoodfacts-dart/model_OcrIngredientsResult/OcrIngredientsResult-class.html
  - https://openfoodfacts.github.io/openfoodfacts-dart/utils_OcrField/OcrField-class.html
  - https://openfoodfacts.github.io/openfoodfacts-dart/utils_OcrField/OcrFieldExtension.html
- Send the ingredients
- Refresh product

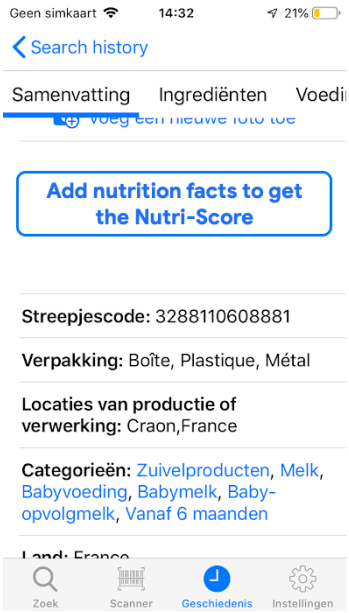
Incomplete products

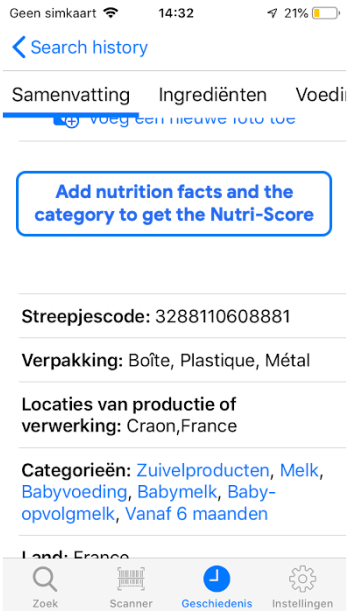
```
if (
status= category-to-be-completed &&
status = ingredients-to-be-completed
)
```

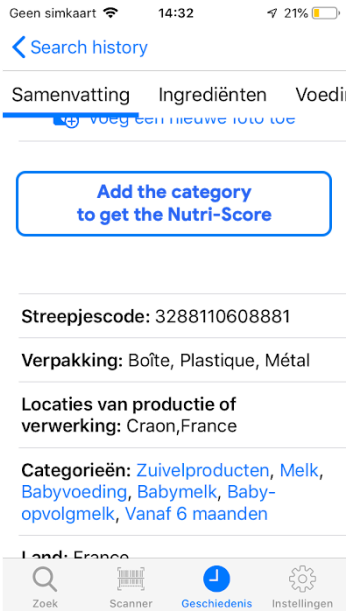
```
then "Add ingredients and a category to see the level of food processing
and potential additives"

if (
status= category-to-be-completed
)
then "Add a category to see the level of food processing and potential
additives"

if (
status = ingredients-to-be-completed
)
then "Add ingredients to see the level of food processing and potential
additives"
```







POST Photos - Uploading

https://us.openfoodfacts.org/cgi/product_jqm2.pl?code=0074570036004&imgupload_front=cheeriosfrontphoto.jpg

Uploading Photos

Photos are source and proof of data. Read this topic to learn how to make calls to upload them to the database.

When you upload an image to Open Food Facts, the image is stored as is.

The first photo uploaded is autoselected as "front" photo.

Read the following before uploading photos:

- **Image Quality:** Uploading quality photos of a product, its ingredients and nutrition table is very important, since it allows the Open Food Facts OCR system to retrieve important data to analyze the product. The minimal allowed size for photos is 640 x 160 px.
- **Upload Behavior:** In case you upload more than one photo of the front, the ingredients and the nutrition facts, beware that only the first photo of each category will be displayed. (You might want to take additional images of labels, recycling instructions, and so on). All photos will be saved.
- **Label Languages:** Multilingual products have several photos based on languages present on the packaging. You can specify the language by adding a lang code suffix to the request.

POST Photo Requests

The API requests to upload photos is very straightforward.

POST https://us.openfoodfacts.org/cgi/product_image_upload.pl

---

Image Upload

Then, add the parameter imagefield to the call and specify from which perspective the photo was taken:

POST
https://us.openfoodfacts.org/cgi/product_jqm2.pl?code=0074570036004&product_image_upload.pl/imguplo
ad_front=cheeriosfrontphoto.jpg

Parameters

- code: the barcode of the product
- imagefield: (can be either: front | ingredients | nutrition | packaging) + '_' and a 2 letter language code. (e.g "front_en" for the front of the product in English, "ingredients_fr" for the list of ingredients in French)
- imgupload_front_fr : your image file if imagefield=front_fr

PARAMS

code

0074570036004

imgupload_front

cheeriosfrontphoto.jpg

Example Request

Photos - Uploading

```
curl --location --request POST
'https://us.openfoodfacts.org/cgi/product_jqm2.pl?code=0074570036004&imgup
load_front=cheeriosfrontphoto.jpg'
```

POSTPhotos - Selecting, Cropping, Rotating

https://world.openfoodfacts.org/cgi/product_image_crop.pl?code=3266110700910&id=nutrition_fr&imgid=1&angle=90

This topic contains the following information:

Selecting, cropping and rotating photos are non-destructive actions. That means, the original version of the image uploaded to the system is kept as is. The subsequent changes made to the image are also stored as versions of the original image.

The actions described in this topic do not modify the image, but provide metadata on how to use it (the data of the corners in the case of selection and the data of the rotation). That is, you send an image to the API, provide an id, you define, for example, the cropping and rotation parameters and as a response, the server generates a new image as requested and you can call this new version of the image.

Selecting and Cropping Photos

Note: Cropping is only relevant for editing existing products. You cannot crop an image the first time you upload it to the system.

Parameters

To select and crop photos, you need to define:

- a barcode
- an incremental id (Similar to a version)
- Cropping parameters (x1, y1, x2, y2). These coordinates define a rectangle in the image and the area that should be kept. Example: 0,0,200,200 px.
- (optional) additional operations:
  - angle= Angle of the rotation
-

Example:

POST
https://world.openfoodfacts.org/cgi/product_image_crop.pl?code=3266110700910&id=nutrition_fr&imgid=1&angle=90

Test server

https://world.openfoodfacts.net/cgi/product_image_crop.org

POST Photos - Performing OCR

https://world.openfoodfacts.net/cgi/ingredients.pl?code=13333560&id=ingredients_en&process_image=1&ocr_engine=tesseract
This topic contains the following information:

Open Food Facts uses optical character recognition (OCR) to retrieve nutritional data and other information from the product labels.

Process

1. Capture the barcode of the product where you want to perform the OCR.
2. The Product Opener server software opens the image (process_image=1)
3. Product Opener returns a JSON response. Processing is done using Tesseract or Google Cloud Vision (recommended). The result is often cripped with errors with Tesseract, less with Google Cloud Vision.

Notes:

- The OCR may contain errors. Encourage your users to correct the output using the ingredients WRITE API.
- You can also use your own OCR, especially if to plan to send a high number of queries.

OCR with Google Cloud Vision

We recommend Google's Vision API to detect and extract text from the images. For more information about this product, see: https://cloud.google.com/vision/docs/ocr?hl=en

Parameters

- Test server: https://world.openfoodfacts.org/cgi/ingredients.pl
- code=code
- id=imagefield
- process_image=1

WRITE Scenario - Adding New products

Meet Dave. He is a developer and an active Open Food Facts contributor that regularly adds new products to the database and completes missing information via API calls. He has described the process below to show other developers how easy it is to contribute.

Structure of the Call

Authentication and Header

If you have an app that makes POST calls and you don't want your users to authenticate in Open Food Facts, you can create a global account. Dave has created a global account for the app he is developing with the following credentials:

- user_id: myappname
- password: 123456

Subdomain

Dave wants to define the subdomain for the query as us. The subdomain automatically defines the country code (cc) and language of the interface (lc).

The country code determines that only the products sold in the US are displayed. The language of the interface for the country code US is English.

In this case:

https://us.openfoodfacts.org/cgi/product_jqm2.pl?

Product Barcode

After the version number, the word code, followed by its barcode must be added:

```
POST https://us.openfoodfacts.org/cgi/product_jqm2.pl?code=0074570036004
```

Credentials

Dave adds his user credentials to the call as follows:

```
POST
https://us.openfoodfacts.org/cgi/product_jqm2.pl?code=0074570036004&user_id=myappname&password=******
```

Add & to concatenate the parameters.

Parameters

You can define one or more parameters to add, for example, the brand and the Kosher label:

- brands: Häagen-Dazs
- labels: kosher

The call looks like this:

POST
https://us.openfoodfacts.org/cgi/product_jqm2.pl?code=0074570036004&user_id=myappname&password=******&brands=Häagen-Dazs&labels=kosher

---

Adding a Comment to your WRITE request.

Use the comment parameter to add the id of the user editing the product. The id should not contain any personal data.

Important! The user id is not the identifier of an Open Food facts user, but the id generated by your system.

It should be structured as: user-agent + user-id.

Example

```
comment=Edit by a Healthy Choices 1.2 iOS user - SxGFRZkFwdytsK2NYaDg4MzRVenNvUEI4LzU2a2JWK05LZkFRSWc9PQ
```
