Faculté des technologies de l'information et de la communication

Département informatique appliquée

UNIVERSITÉ DES
MASCAREIGNES

SAVOIR, C'EST POUVOIR

Travaux pratique

TP6

Réalisé par : Nathan Carlinot RANDRIAMIHAJA

Année scolaire : 2023 - 2024

Lab session 6: Integrating Machine Learning Model in a Flutter application

Task-1 Tutoriel TensorFlow Lite pour Flutter : classification d'images

Dans cette tâche, nous apprendrons à utiliser TensorFlow Lite dans Flutter. Cela implique de former le modèle d'apprentissage automatique avec Teachable Machine et d'intégrer le résultat dans une application mobile Flutter.

Nous développerons une application appelée "Reconnaissance de Plantes" qui utilise l'apprentissage automatique pour reconnaître les plantes simplement en regardant des photos d'elles. Nous réaliserons cela en utilisant la plateforme Teachable Machine, TensorFlow Lite, et un package Flutter nommé tflite_flutter.

TensorFlow est une bibliothèque d'apprentissage automatique populaire pour les développeurs qui souhaitent créer des modèles d'apprentissage pour leurs applications. TensorFlow Lite est une version mobile de TensorFlow pour le déploiement de modèles sur des appareils mobiles. Et Teachable Machine est une plateforme conviviale pour la formation de modèles d'apprentissage automatique.

A- Commencer

Nous utiliserons un projet de départ fourni par le tutoriel comme base. Ce projet permet déjà aux utilisateurs de choisir des images ou de les glisser-déposer directement dans l'application, mais l'application ne reconnaît pas les images. Nous utiliserons TensorFlow Lite pour résoudre ce problème dans les prochaines sections.

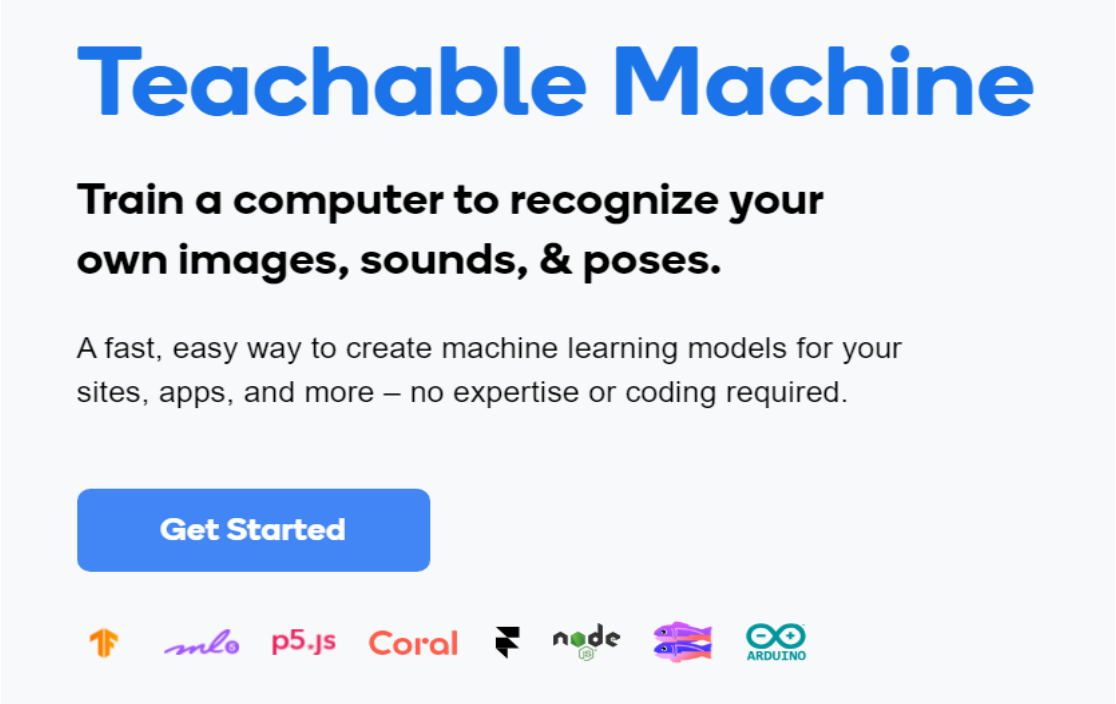
B- Construire un modèle avec une machine enseignable

Le projet de départ contient déjà un modèle entraîné model_unquant.tflite et des étiquettes de classification dans le fichier labels.txt, mais nous partirons du processus d'entraînement pour comprendre comment il est mis en œuvre.

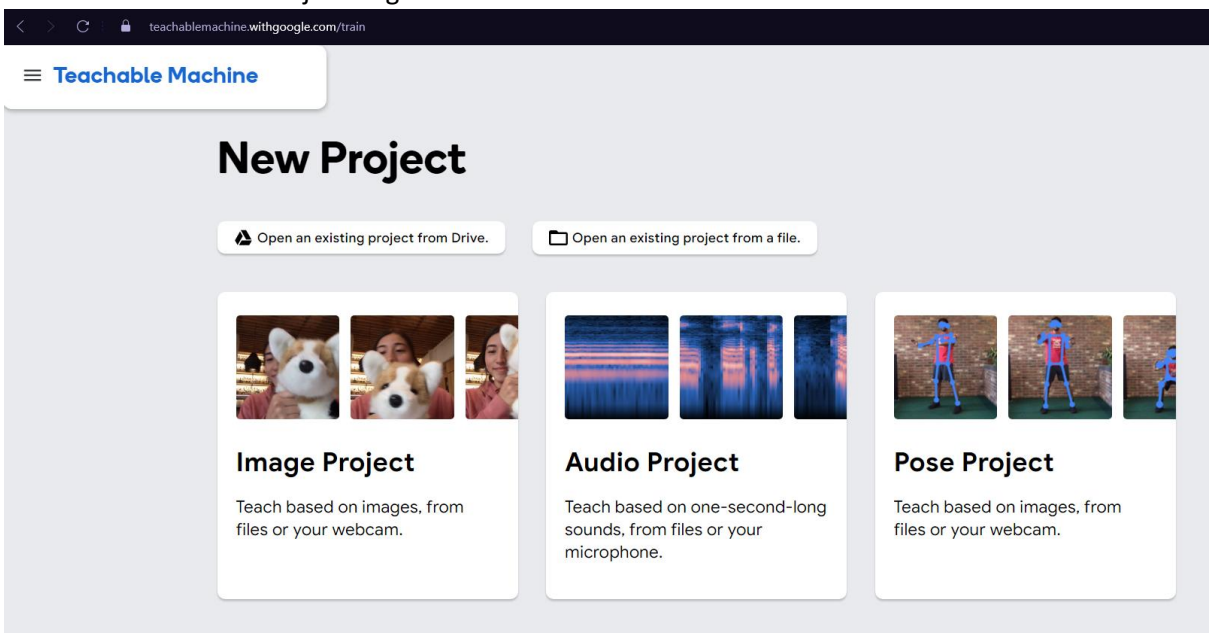
C- Préparation de l'ensemble de données- Entraînement du modèle

La formation est le processus par lequel l'ordinateur apprend des données et en dérive des règles.

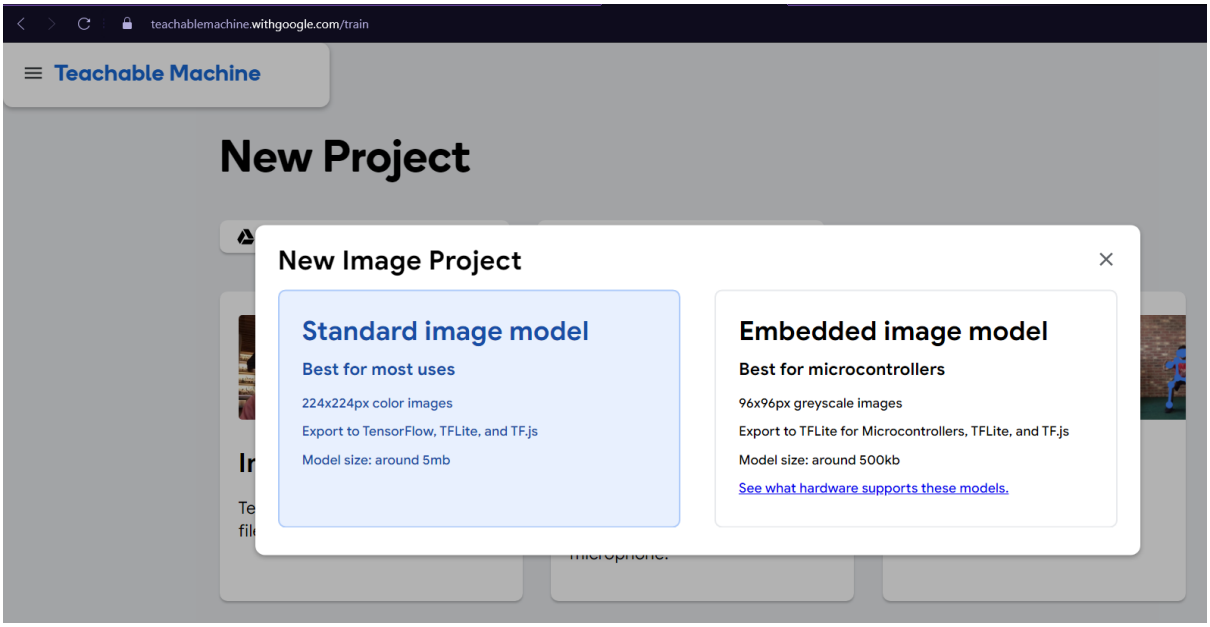
1. Tout d'abord, accédez à https://teachablemachine.withgoogle.com et cliquez sur Commencer pour ouvrir l'outil de formation



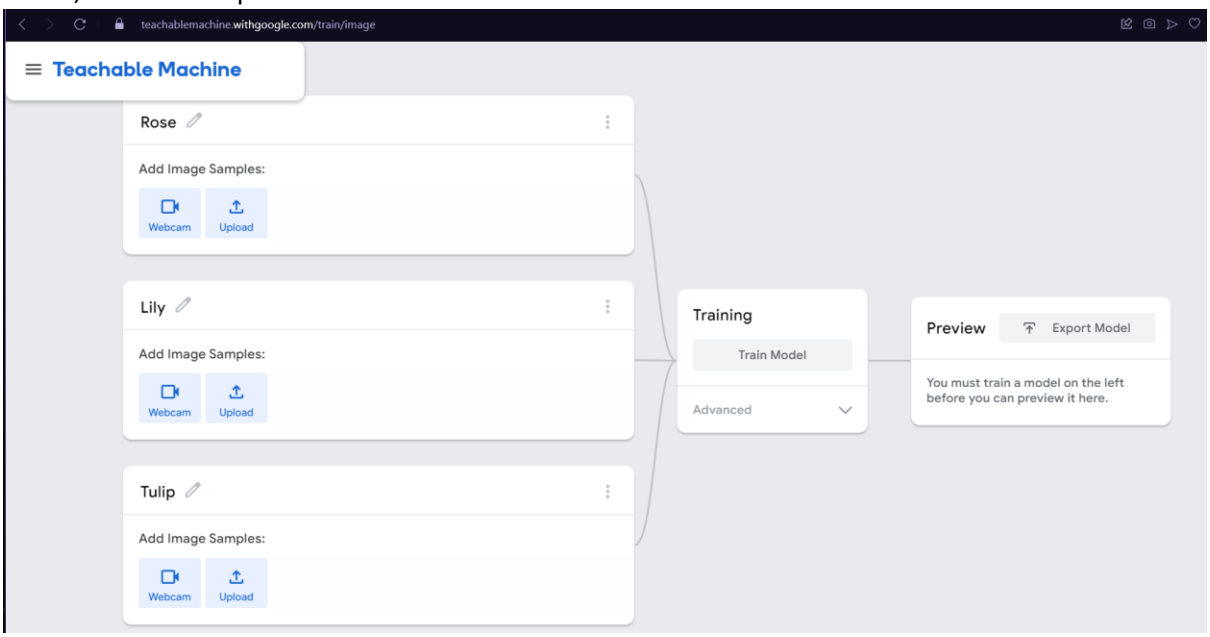
2. Sélectionnez ensuite Projet Image :



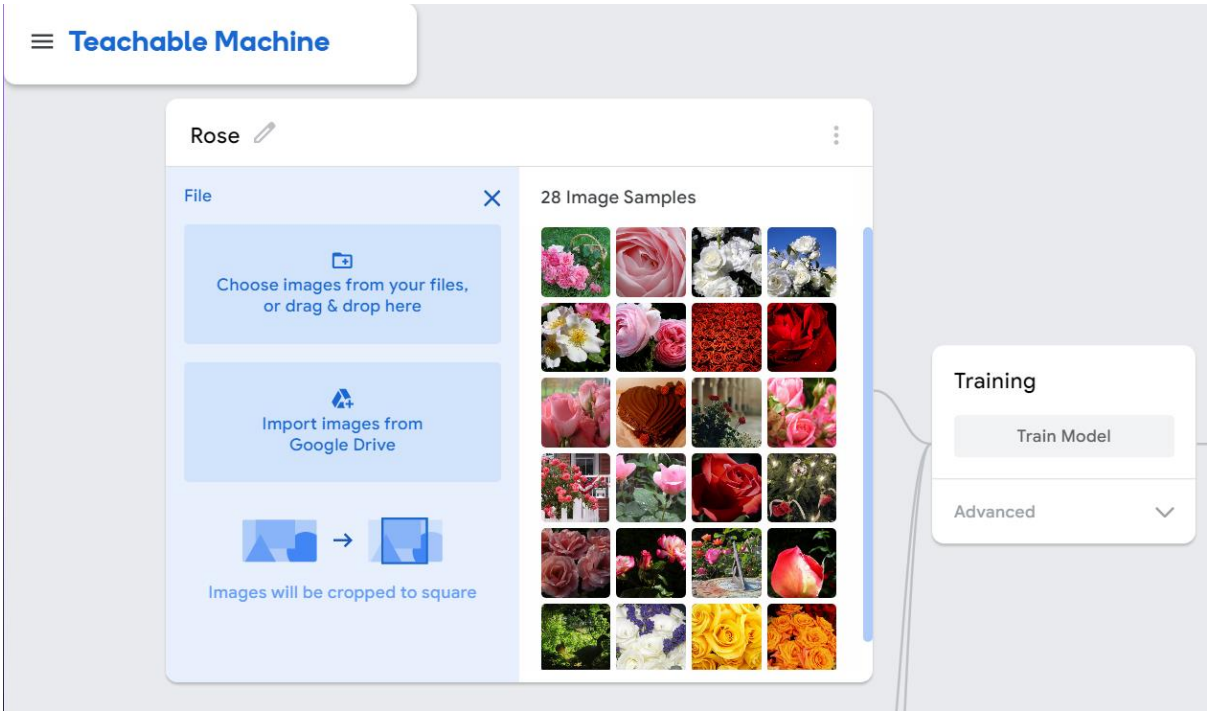
3. Choisissez Modèle d'image standard, car nous n'entraînons pas un modèle pour qu'il s'exécute sur un microcontrôleur :



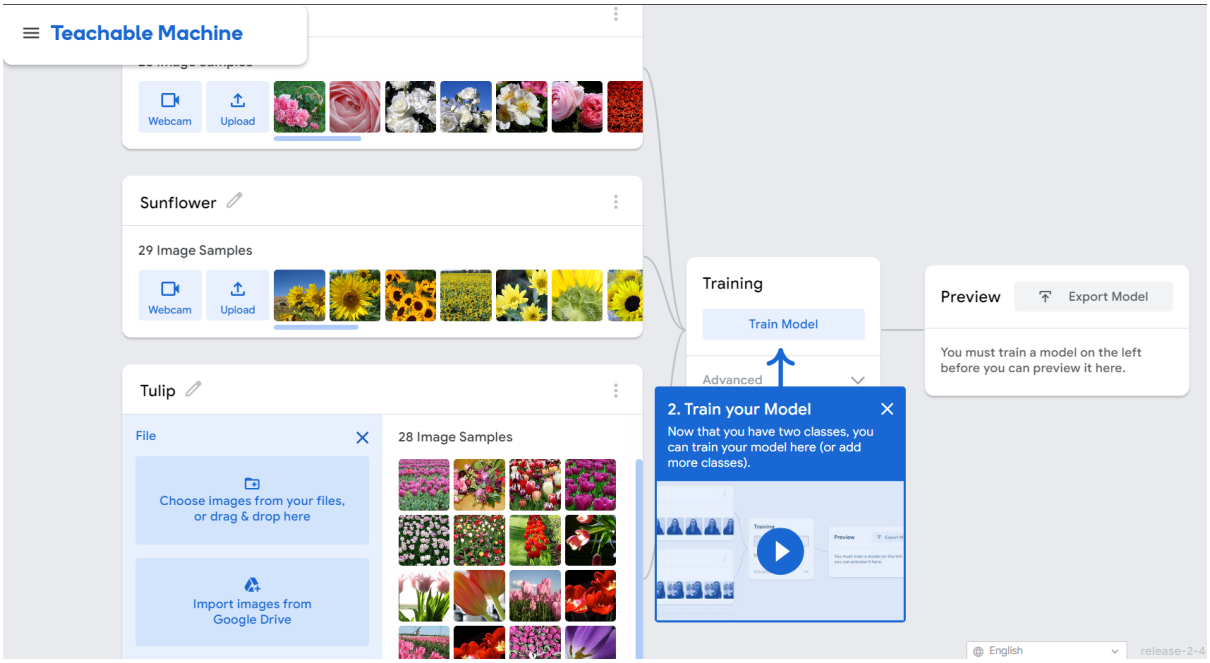
4. Une fois dans l'outil de formation, ajoutez les classes et modifiez les étiquettes de chaque classe, comme indiqué ci-dessous :



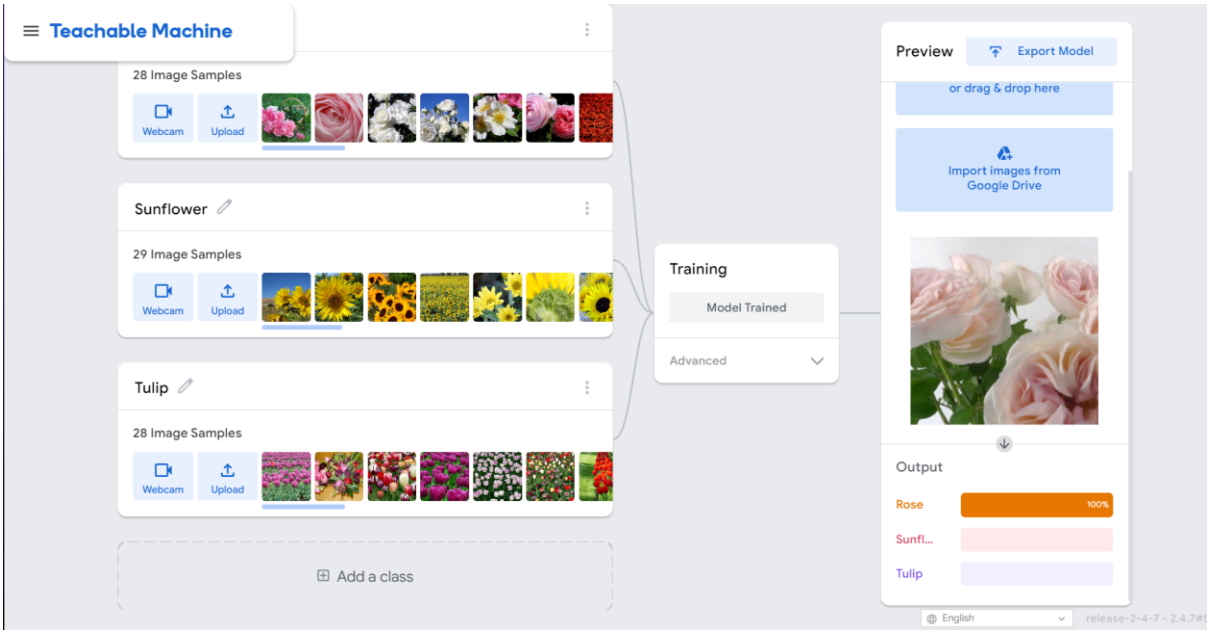
5. Ensuite, ajoutez les exemples de formation en cliquant sur Télécharger sous chaque classe. Ensuite, faites glisser le dossier du type de plante approprié du dossier d'échantillons vers le panneau Choisir des images à partir de vos fichiers….



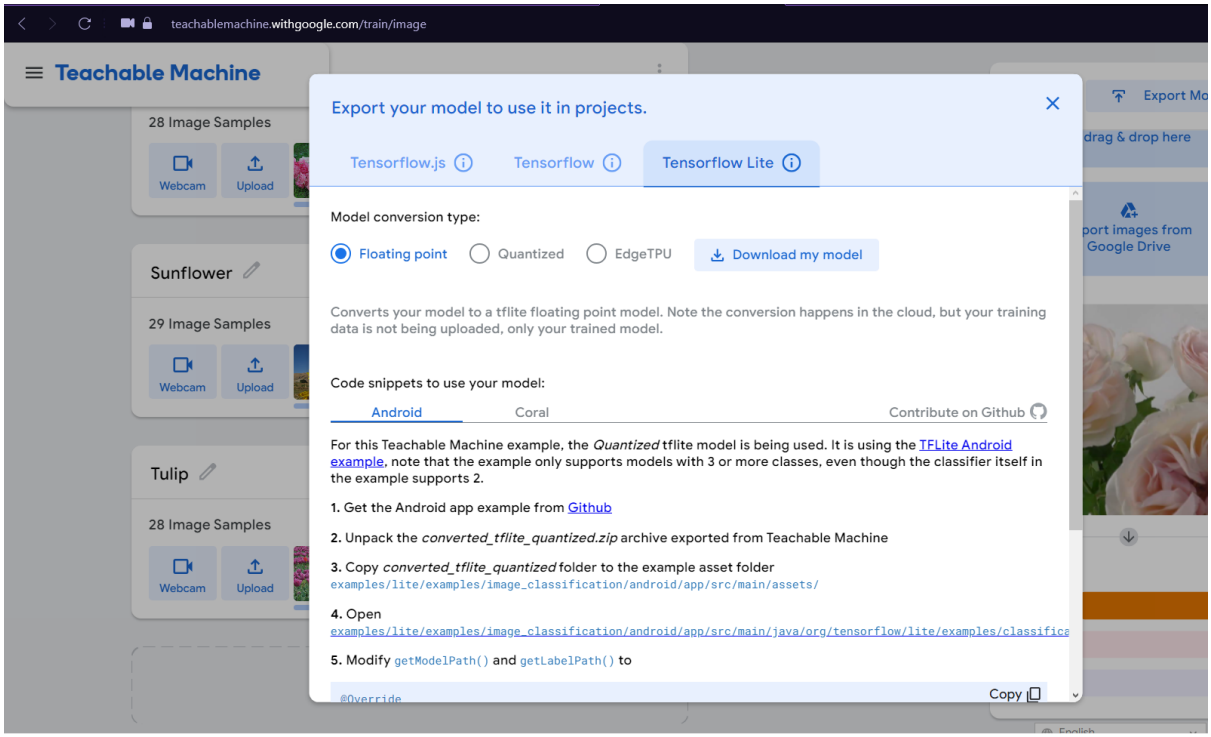
6. Après avoir ajouté tous les exemples de formation, cliquez sur Entraîner le modèle pour entraîner le modèle :



7. Une fois la formation terminée, testez le modèle avec d'autres images d'usine. Utilisez les images du dossier samples-test, comme ceci :

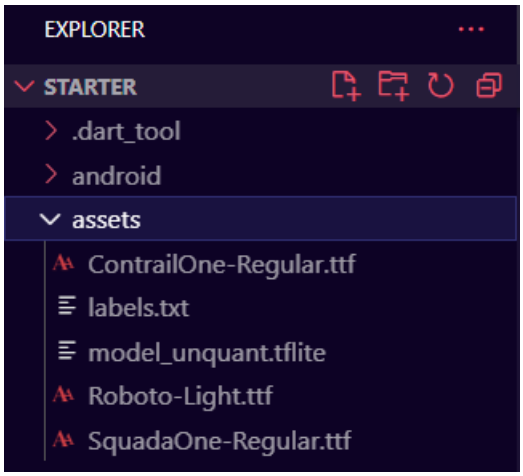


8. Enfin, exportez le modèle en cliquant sur Exporter le modèle dans le panneau Aperçu. Dans la boîte de dialogue, choisissez TensorFlow Lite. C'est parce que la plateforme cible est mobile.
9. Ensuite, sélectionnez le type de conversion à virgule flottante pour obtenir les meilleures performances prédictives. Cliquez ensuite sur Télécharger mon modèle pour convertir et télécharger le modèle.



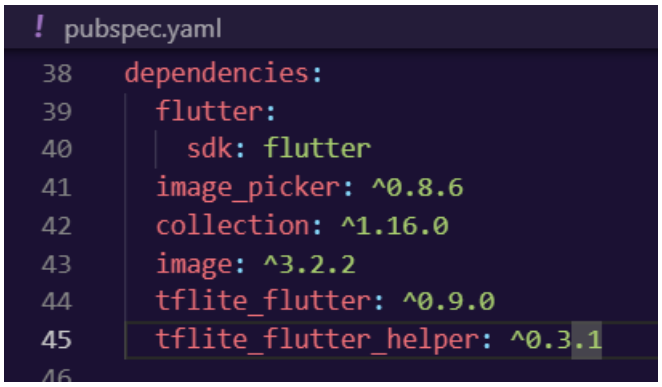
Le processus de conversion du modèle peut prendre plusieurs minutes. Une fois cela fait, le fichier modèle sera automatiquement téléchargé sur votre système.

10. Une fois que vous avez le fichier modèle converti_tflite.zip en main, décompressez-le et copiez labels.txt et model_unquant.tflite dans le dossier ./assets du projet de démarrage.



D- Installer TensorFlow Lite dans Flutter

Pour utiliser TensorFlow dans votre application Flutter, vous devez installer les packages suivants :



- tflite_flutter : vous permet d'accéder à la bibliothèque native TensorFlow Lite. Lorsque vous invoquez les méthodes de tflite_flutter, cela appelle la méthode correspondante du SDK natif TensorFlow Lite.
- tflite_flutter_helper : vous permet de manipuler les entrées et les sorties de TensorFlow. Par exemple, il convertit les données d'image en structure tensorielle. Cela réduit l'effort nécessaire pour créer la logique de prétraitement et de post-traitement pour votre modèle.

E- Création d'un classificateur d'images

En apprentissage automatique, la classification consiste à prédire la classe d'un objet parmi un nombre fini de classes, en fonction de certaines entrées.
Le projet de départ implémente déjà les widgets et l'utilisation de l'instance du classifieur.

F- Importer le modèle dans Flutter

Nous allons charger deux éléments de données dans le programme : le modèle d'apprentissage automatique – model_unquant.tflite et les étiquettes de classification – labels.txt, que nous avons obtenues de la plateforme Teachable Machine.

Pour commencer, assurez-vous d'inclure le dossier Assets dans pubspec.yaml :

```
! pubspec.yaml
67    assets:
68      - assets/
69
```

L'enregistrement des actifs est responsable de la copie des fichiers de ressources dans le bundle d'application final.

G- Chargement des étiquettes de classification

1. Ouvrir lib/classifier/classifier.dart et import tflite_flutter_helper:

```
lib > classifier > classifier.dart > ...
27  // THE SOFTWARE.
28
29  import 'package:flutter/foundation.dart';
30  import 'package:image/image.dart';
31
32  import 'classifier_category.dart';
33  import 'classifier_model.dart';
34  import 'package:tflite_flutter_helper/tflite_flutter_helper.dart';
```

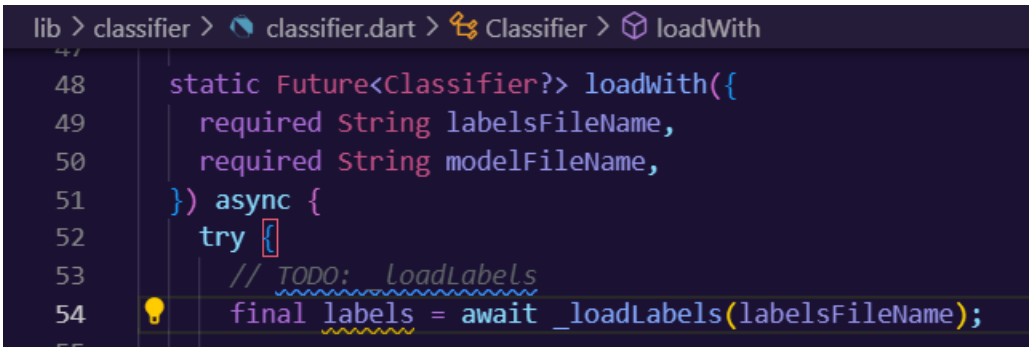
2. Ajoutez ensuite le code suivant après la prédiction :

```
lib > classifier > classifier.dart > ...
67
68    ClassifierCategory predict(Image image) {
69      debugPrint(
70        'Image: ${image.width}x${image.height}, '
71        'size: ${image.length} bytes',
72      );
73
74      // TODO: _preProcessInput
75      // TODO: run TF Lite
76      // TODO: _postProcessOutput
77
78      return ClassifierCategory('Unknown', 0);
79    }
80
81    static Future<ClassifierLabels> _loadLabels(String labelsFileName) async {
82      // #1
83      final rawLabels = await FileUtil.loadLabels(labelsFileName);
84
85      // #2
86      final labels = rawLabels
87          .map((label) => label.substring(label.indexOf(' ')).trim())
88          .toList();
89
90      debugPrint('Labels: $labels');
91      return labels;
92    }
```

Voici ce que fait le code ci-dessus :

- Supprime le préfixe du numéro d'index des étiquettes que vous avez précédemment téléchargées. Par exemple, il transforme "0 Rose" en "Rose".
- Charge les étiquettes en utilisant l'utilitaire de fichier de tflit_flutter_helper

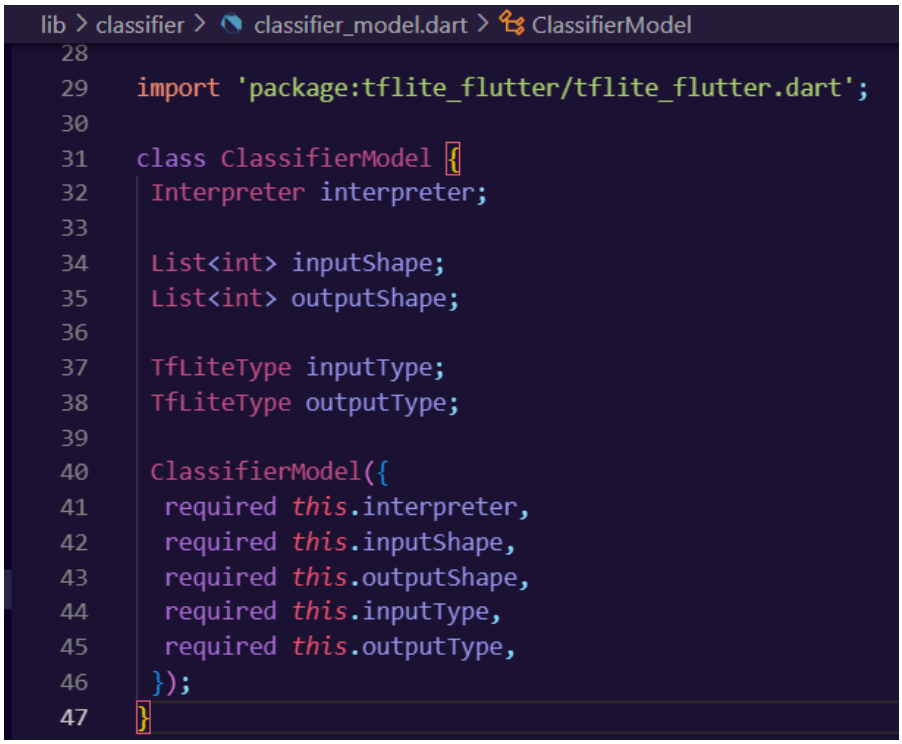
3. Ensuite, remplacez // TODO: _loadLabels dans loadWith en appelant _loadLabels comme ceci :



Ce code charge le fichier d'étiquette.

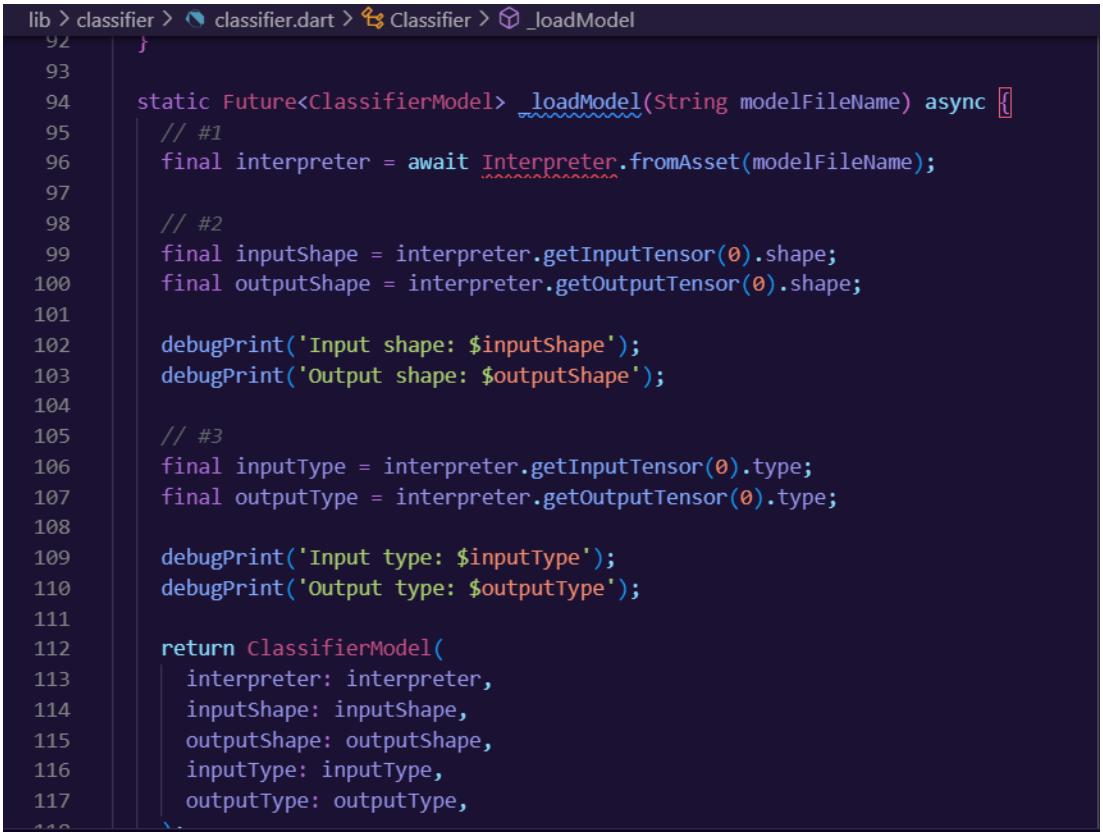
H- Importer TensorFlow Lite Model

1. 1. Accédez à lib/classifier/classifier_model.dart et remplacez le contenu par le code suivant :



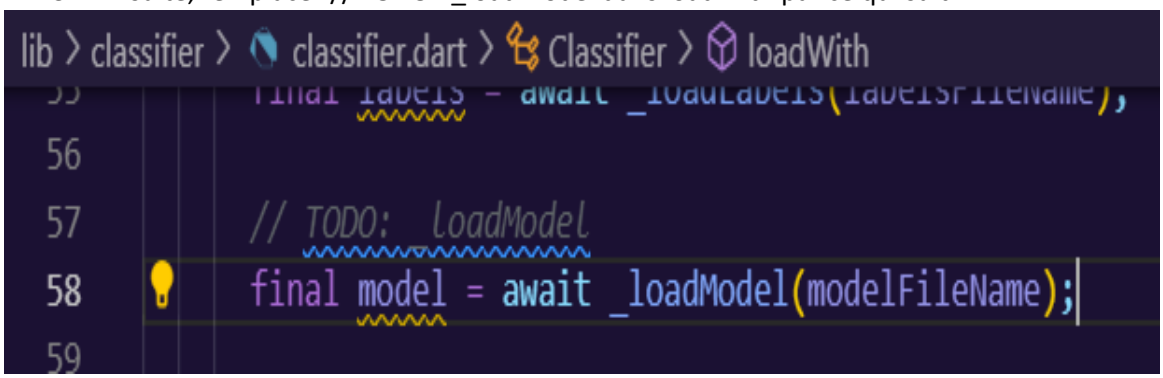
ClassifierModel stocke toutes les données liées au modèle pour votre classificateur. Vous utiliserez l'interprète pour prédire les résultats. inputShape et outputShape sont respectivement des formes pour les données d'entrée et de sortie, tandis que inputType et outputType sont les types de données des tenseurs d'entrée et de sortie.

2. Maintenant, importez le modèle à partir du fichier. Accédez à lib/classifier/classifier.dart et ajoutez le code suivant après _loadLabels :



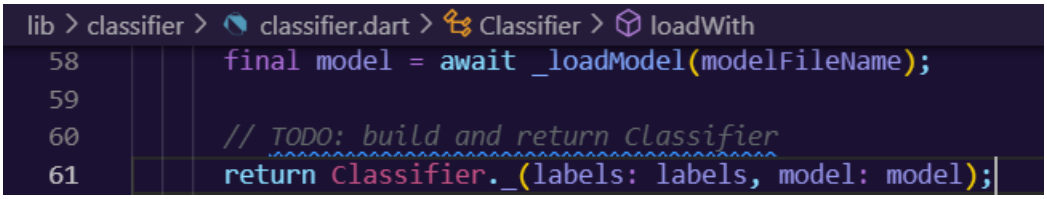
N'oubliez pas d'ajouter l'import import 'package:tflite_flutter/tflite_flutter.dart' ; au sommet.

3. Ensuite, remplacez // TODO : _loadModel dans loadWith par ce qui suit :



Le code ci-dessus charge le fichier modèle.

4. Enfin, pour l'initialisation, remplacez // TODO: build et return Classifier in loadWith par ce qui suit :



Cela crée l'instance Classifier, que PlantRecogniser utilise pour reconnaître les images fournies par l'utilisateur.

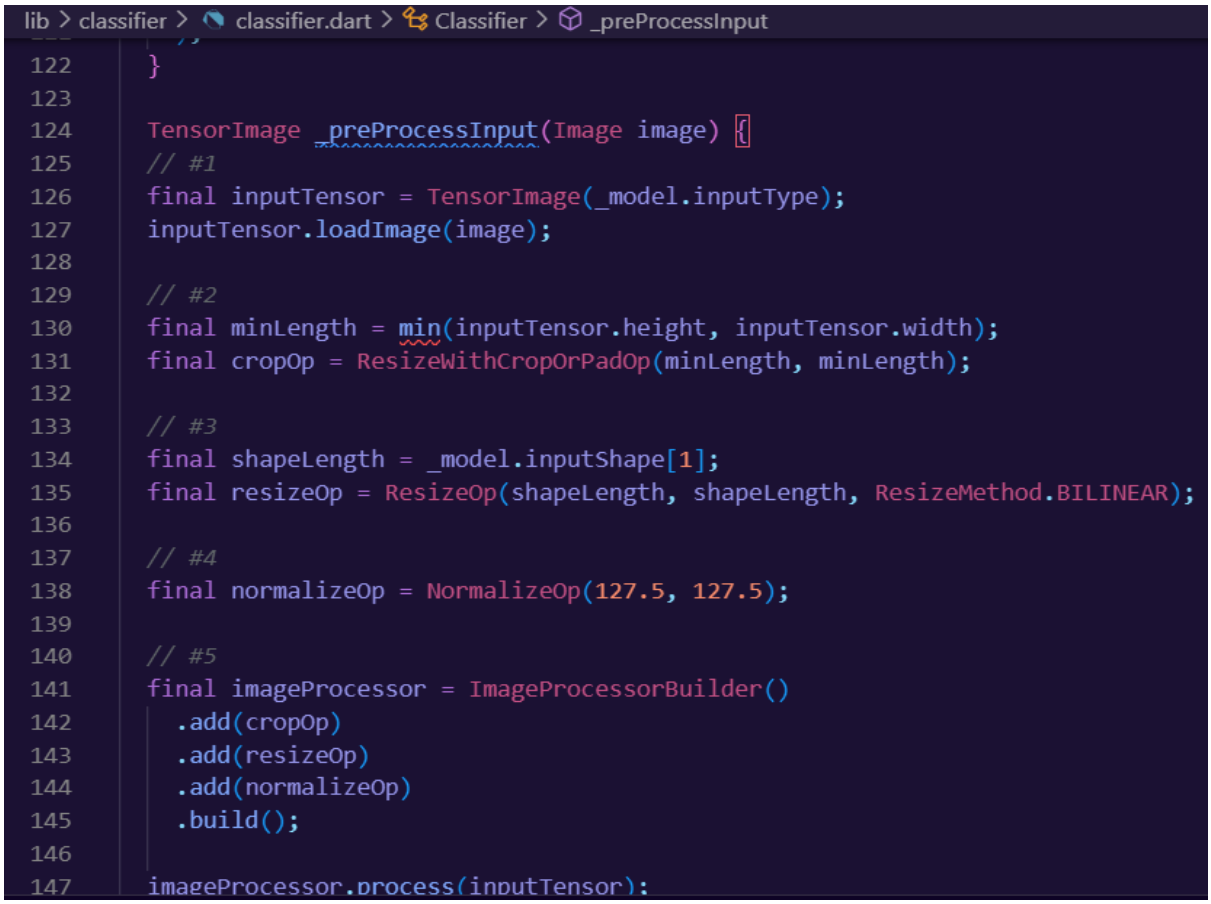
I-Implémentation de la prédiction TensorFlow

Avant de faire une prédiction, nous devons préparer les entrées.
Nous allons écrire une méthode pour convertir l'objet Flutter Image en TensorImage, la structure tensorielle utilisée par TensorFlow pour les images. Nous devons également modifier l'image pour l'adapter à la forme requise du modèle.

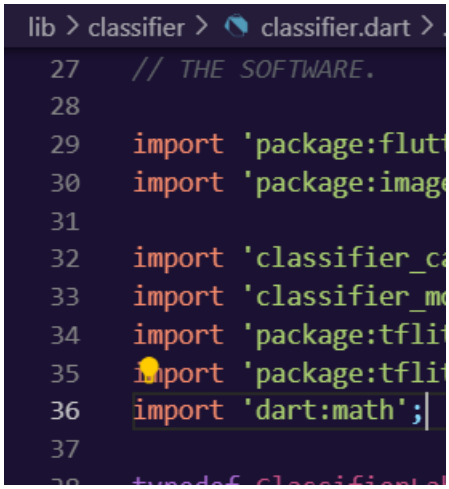
Données d'image de pré-traitement

Avec l'aide de tflite_flutter_helper, le traitement des images est simple car la bibliothèque fournit plusieurs fonctions que vous pouvez intégrer pour gérer le remodelage des images.

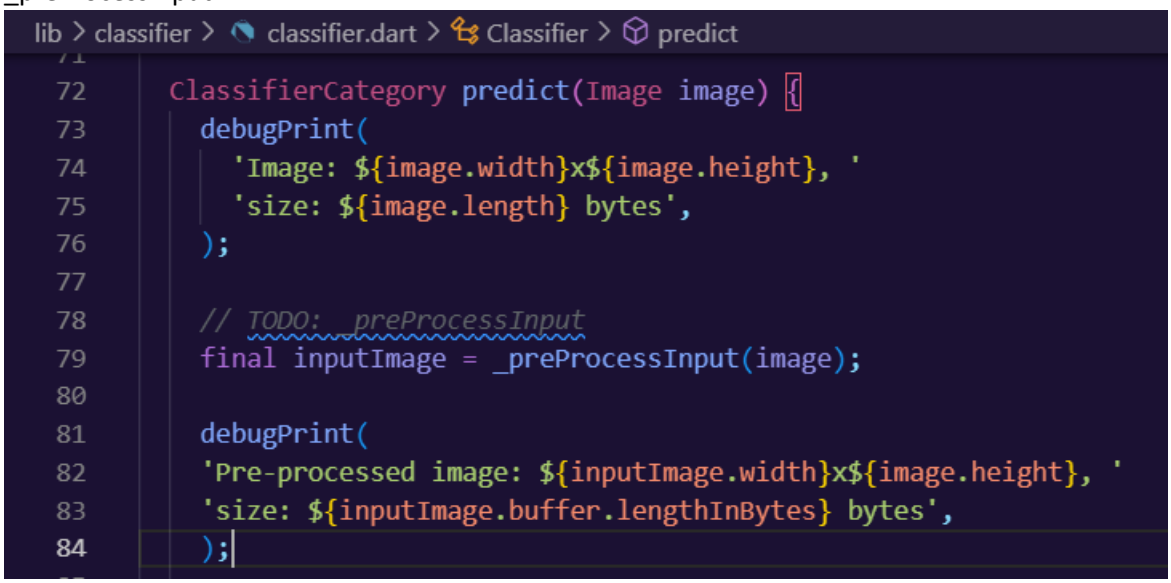
1. Ajoutez la méthode _preProcessInput à lib/classifier/classifier.dart :



Vous devez importer dart:math en haut pour utiliser la fonction min.

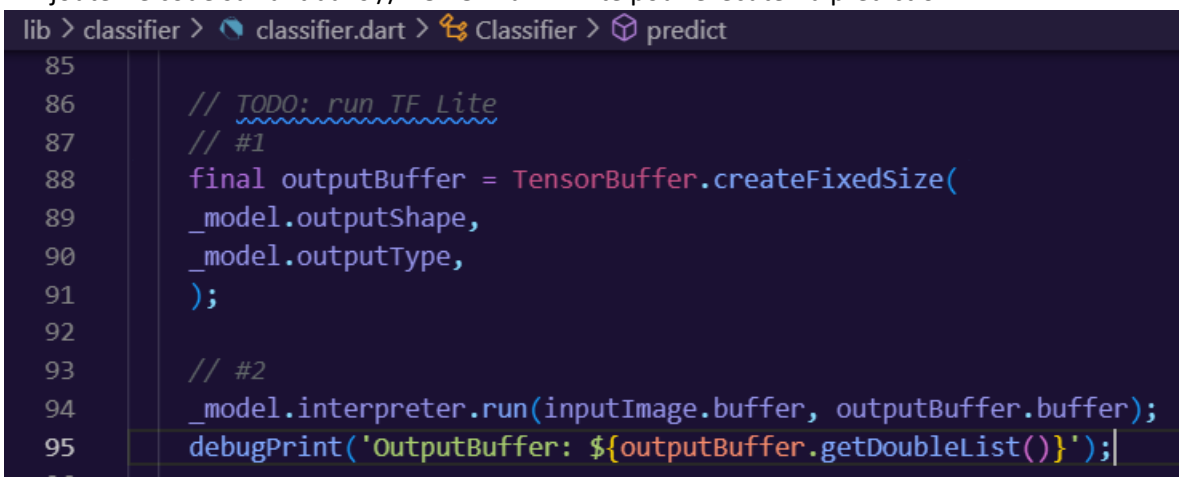


2. Ensuite, appelez la méthode à l'intérieur de prédire(...) à // TODO : _preProcessInput :



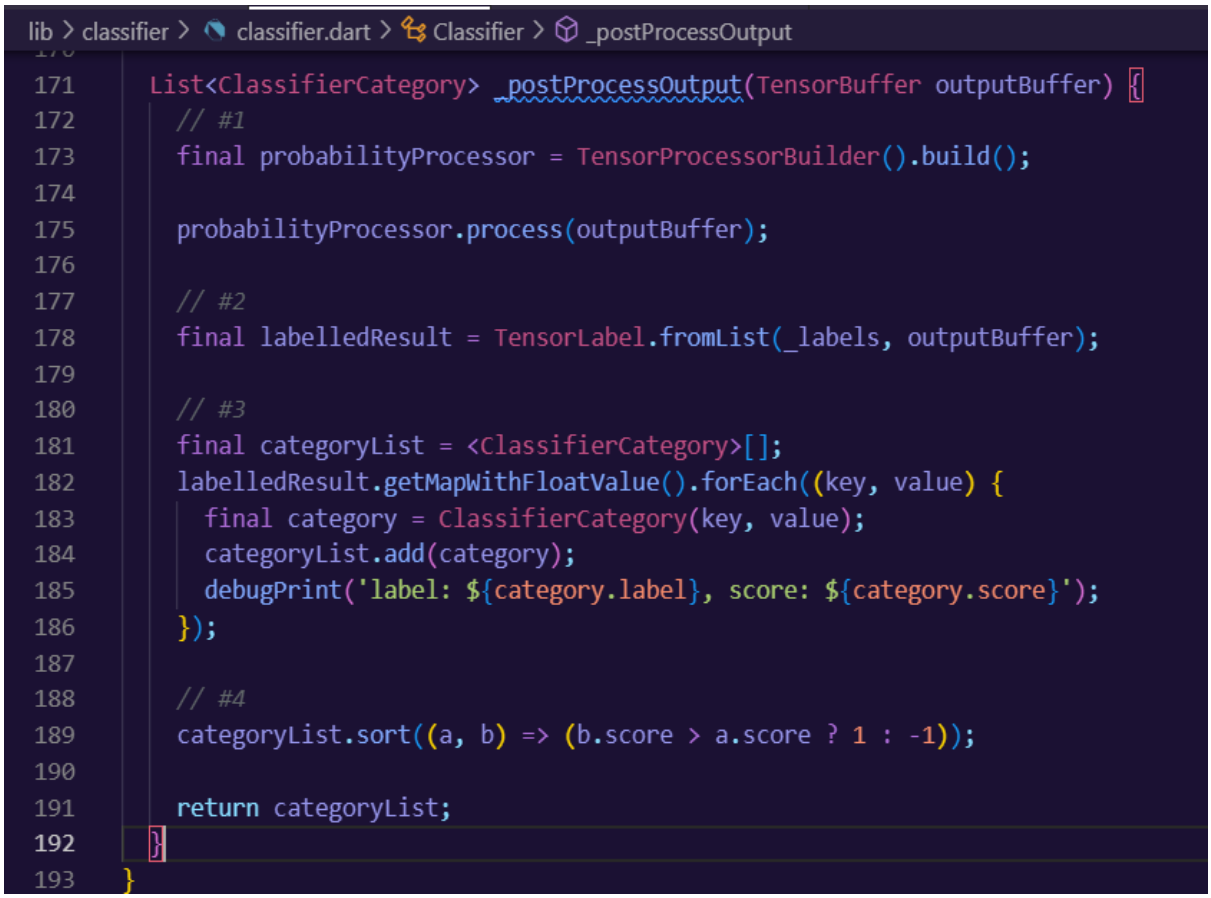
Exécuter la prediction

1. Ajoutez le code suivant dans // TODO : run TF Lite pour exécuter la prédiction :

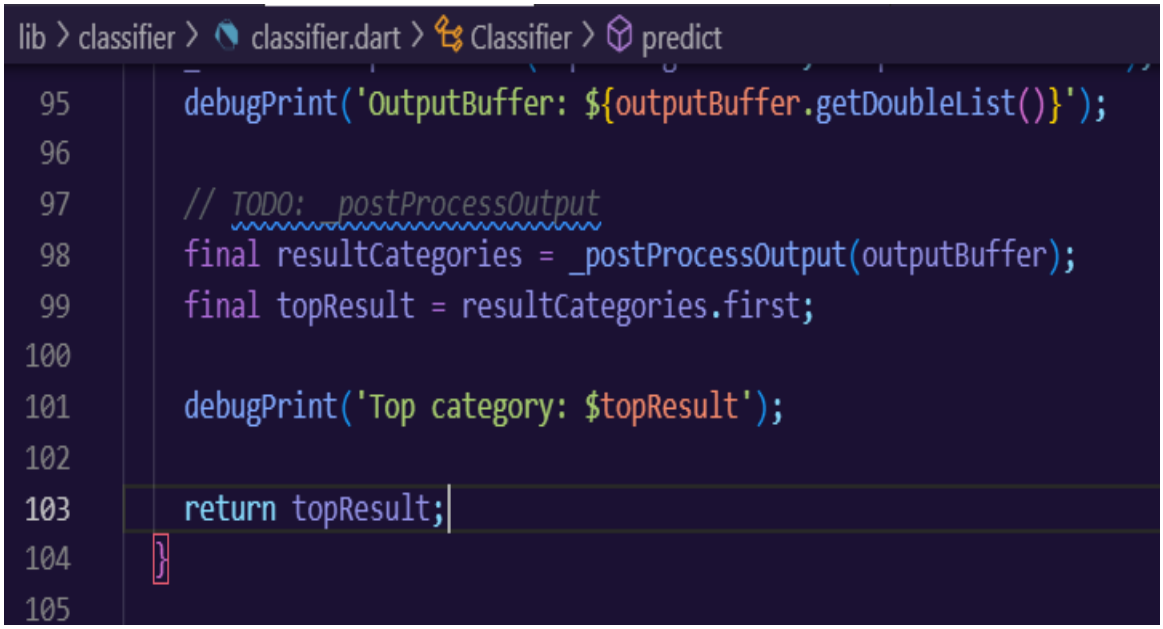


Post-traitement du résultat de sortie

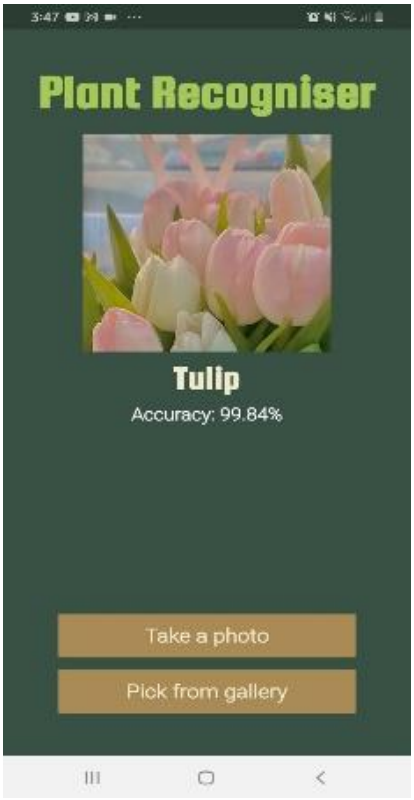
1. Ajoutez la méthode suivante à lib/classifier/classifier.dart :



2. Il ne vous reste plus qu'à invoquer _postProcessOutput() pour la prédiction. Mettez à jour prédire (...) pour qu'il ressemble à ce qui suit :



3. Créez et exécutez. Téléchargez une image et voyez-la prédire correctement la plante :



Conclusion

Pour conclure, ce tutoriel s'est avéré être un point de départ exceptionnel pour nous qui débutons dans l'implémentation de l'apprentissage automatique. Nous avons assimilé plusieurs concepts clés grâce à des explications à la fois précises et accessibles. En définitive, cette expérience nous a permis d'acquérir des compétences essentielles telles que :

- L'application de l'apprentissage automatique au sein d'une application mobile, ouvrant ainsi la voie à des innovations technologiques personnelles et professionnelles.
- L'entraînement d'un modèle de manière intuitive avec Teachable Machine, démystifiant le processus souvent perçu comme complexe de la modélisation.
- L'intégration et l'exploitation efficace de TensorFlow Lite en conjonction avec le package tflite_flutter, illustrant la puissance et la flexibilité de ces outils.
- Le développement d'une application mobile innovante capable de reconnaître les plantes à partir d'images, témoignant de l'impact pratique et tangible de l'apprentissage automatique dans notre quotidien.

Cette aventure éducative nous a non seulement équipés de connaissances techniques fondamentales mais nous a également inspirés à poursuivre l'exploration du potentiel illimité de l'apprentissage automatique.

Table de Matière