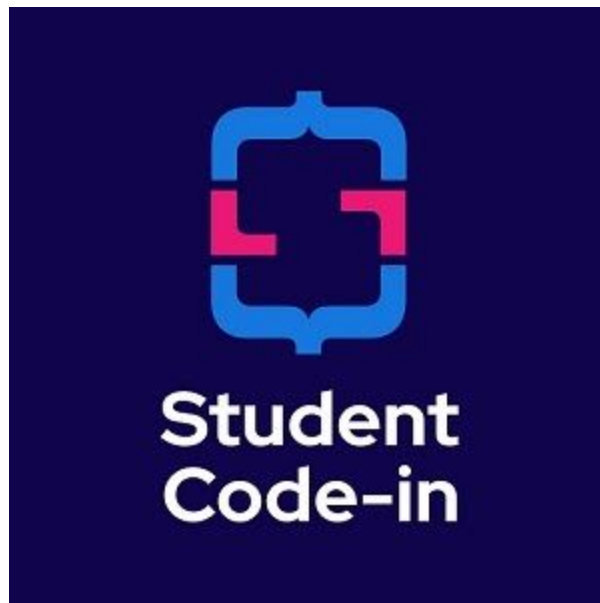


RESEARCH WORK
ON

IMAGE CLASSIFICATION APP FOR VISUALLY IMPAIRED

Submitted in partial fulfillment of the requirement for the
Open Source Program

STUDENT CODE-IN



Submitted By
ASHWINI JHA

Under the supervision of
AKHILDAS KS (Project Administrator)

Preface

There is no greater impediment to the advancement of knowledge than the ambiguity of words.

-Thomas Reid

Image or Object classification based digital aid is quite helpful for visually impaired people to get description of nearby objects. With the advancement in computer vision computing technologies we can afford to develop a system for visually impaired people, which can give audio feedback of surrounding images, objects and context. This project explores image classification methods to assist blind & visually impaired people. We proposed an Image Classification App for Visually Impaired as an assistive system which is very useful for their safety, quality of life and freedom from other people all the time.

The Image Classification module recognizes **13 objects such as car, day, dog, door_open, door_closed, human, human_with_mask, night, shop_closed, stairs, tree, tap, zebra_crossing** and generates an audio feedback to the user by converting the classified text to speech using python[] . Android Studio development environment has been used for Android development. Work on the layout of the App has also been done using Android Studio.

Online available dataset has been used for the evaluation of the Image Classification model (Algorithm). To increase the accuracy, clear, precise & more number of images in the data set have been used .

To teach the model to classify images using files or your webcam Teachable Machine [] is used. Teachable Machine is a web-based tool that makes creating machine learning models fast, easy, and accessible to everyone. The models made with Teachable Machine are real **TensorFlow.js** models that work anywhere javascript runs, so they play nice with tools like Glitch, P5.js, Node.js & more. Plus, export to different formats to use the models elsewhere, like **Coral**, Arduino & more.

Machine Learning under the hood - the models use a technique called transfer learning. There's a pre-trained neural network, and when we create our own classes, we can sort of picture that our classes are becoming the last layer or step of the neural net. Specifically, the image models are learning off of **pre-trained mobilenet models**.

CONTENTS

1. Chapter 1

1.1. Introduction

1.2. Image Classification with Machine Learning

1.2.1. Example

1.2.2. Teaching a computer how to see & recognize different objects

1.3. Assistive Technologies

1.4. Motivation

1.5. Objective

2. Chapter 2

2.1. Developer Tools & Technologies used for building the App

2.2. Teachable Machine

2.2.1. Works With

2.3. Steps for Building the TensorFlow Lite Android App

2.4. Results

3. Chapter 3

3.1. Convert text into natural sounding speech using Python

3.1.1. Speech Synthesis

3.1.2. Text-to-Speech

3.1.3. Let's bring Python into the picture

3.1.4. Let's get started with pip

3.2. gTTS

3.3. pyttsx3

3.4. playsound

3.5. Online Synthesis

- 3.6. Offline Synthesis**
- 3.7. Run the python scripts using the respective commands**
 - 3.7.1. Online Synthesis**
 - 3.7.2. Offline Synthesis**
- 3.8. Result**
- 4. Chapter 4**
 - 4.1. Adding voice to the App**
- 5. Chapter 5**
 - 5.1. Changing the Layout of the App using Android Studio**
 - 5.1.1. Certain changes that have been made in the layout of the App**
 - 5.2. Modifications**
 - 5.3. Files that I have changed**
- 6. Chapter 6**
 - 6.1. Future Work**

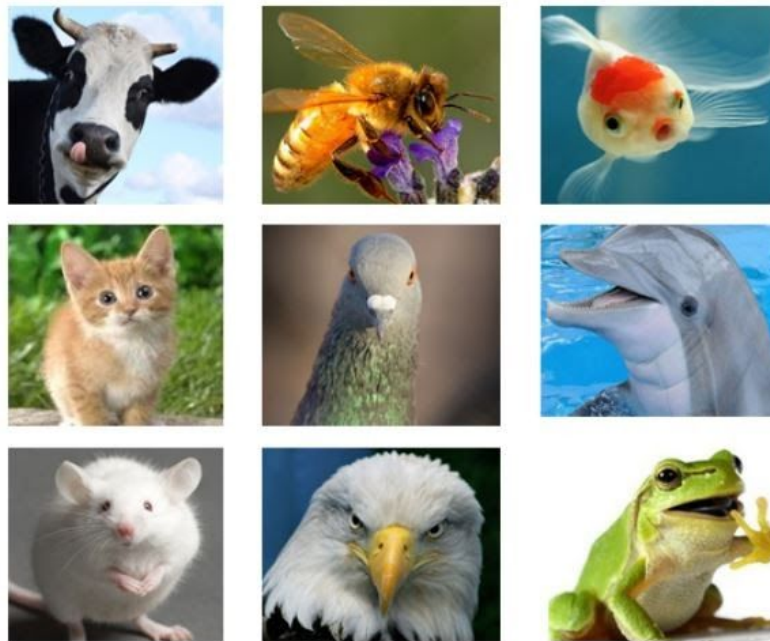
Chapter 1

1.1 Introduction

The environment around us is highly complex so we need several sensors such as vision, touch, smell etc. to survive in this world. All the creatures on Earth have a set of such sensors which help them search for food, water, and safety etc. Among all these sensors vision is a critically important sensor because it gives accurate and complex representation of the environment which can be processed to get valuable information.

The main advantages of a vision sensor as compared to others are its large and wide range, ability to provide complex data which can be processed to extract

information such as object color, shape etc. In the figure some creatures are depicted, note that all of these are having a pair of eyes irrespective of environment(air, land, water), it can be concluded from the figure that vision is the primary sensor to survive not only on land but in water and air also.



Human beings are highly dependent on vision sensors for daily tasks such as walking, eating, finding food, searching, driving vehicles, reading books etc., object recognition is the core algorithm in most vision related tasks. We do object classification all the time for example while you are reading this work you are recognizing characters and hence words, object classification is needed in navigation, tracking, automation and so on, from above discussion we can convince ourselves that object classification is highly important. But what if someone is born without vision capability or in some accident one loses sight?

Without eyes it is hard to survive even in his/her own environment, the person without sight has to remember everything in his environment and get irritated if someone misplaces objects. Navigation and or searching objects in unfamiliar environments is surprisingly difficult for visually impaired people. In the following sections we introduced Face recognition, Image Classification and we cover work which have been done so far and related to our research. Motivation and problem statements are also given in this chapter.

1.2 Image Classification with Machine Learning

Image Classification is inherent part of our vision system to survive in this world, human and other creatures can perform this task instantly and effortlessly, but this is a hard problem for machine because each object in 3D world can cast infinite number of 2D projections due to affine transforms, illumination change and camera viewpoint.

The more recent approach to solve object recognition problems is using Deep learning based methods where a deep neural network is designed and then train it with millions of samples in a supervised manner, though it takes days or even weeks to train but it outperforms all hand designed methods developed so far. Deep neural network architecture can be trained in two ways, one is using supervised learning methods with millions of samples and second is using unsupervised learning methods. In unsupervised learning the network is trained with unlabeled data and it takes very less training time and labeled data is not required.

In our object recognition method, we are using a Teachable machine the working of which is explained in [1].

1.2.1 Example

Consider you are creating a game of **Rock, Paper, Scissors**, when you play this with a human it is very basic a child can learn it in just a few minutes. Now let's take a look at the most basic part of the game that the human brain is really good at & that is recognizing or classifying what it is looking at. Most people can just look at the images (shown in the figure) & recognize which one is a rock, a paper, or a scissor. But the question is How would you program a computer to recognize them ? Think of all of the diversity of hand types, skin colour & all the people who do scissors with their thumbs sticking out & the people who do scissors with their thumb in. If you have ever written any kind of code you will instantly realize that this is a really really difficult task. And might take you thousands or tens of thousands of lines of code & that's just to play rock, paper or scissors.



So what if there was a different way to teach a computer to recognize what it sees? What if you would have a computer learn in the same way a human does. That's the core of Machine Learning & the path to artificial intelligence. Traditional Programming looks like this you have for example a feed from the webcam & you have rules that act on this data. These rules are expressed in a programming language & are the bulk of any code that you write. Ultimately these rules will act on the data & give you an answer . Maybe it sees a rock, maybe it sees a paper & maybe it sees scissors.

But what if you turn this diagram around, & instead of you as the programmer figuring out the rules, you instead give it answers with the data & have the computer figure out what the rules are. That's Machine Learning. So now I can have lots of pictures of rocks & tell a computer this is what a rock looks like, this is what paper looks like, & this is what scissors look like. And I can have a computer figure out patterns that match them to each other. Then my computer will have learnt to recognize a rock, paper & scissors. That's the core of building something that uses Machine Learning . You get a set of data that has patterns inherent in it, & you have a computer learn what those patterns are.

1.2.2 Teaching a computer how to see & recognize different objects

For example, Look at these pictures. How many shoes do you see? You might say two, right? But how do you know they are shoes? Imagine if somebody had never seen shoes before. How would you tell them that despite the great difference between the high heel & sports shoe, they're still both shoes. Maybe they would think if it's red, it's a shoe. Because all they've seen are these two, & they're both red. But, of course, it's not that simple



But how do you know that these two are shoes? Because in your life, you've seen lots of shoes, & you've learned to understand what makes a shoe a shoe. So it follows logically that if we show a computer lots of shoes, it will be able to recognize what a shoe is. And less the data the faster is for a computer to process it. The power of TensorFlow allows you to design neural networks for a variety of tasks with consistent programming API.

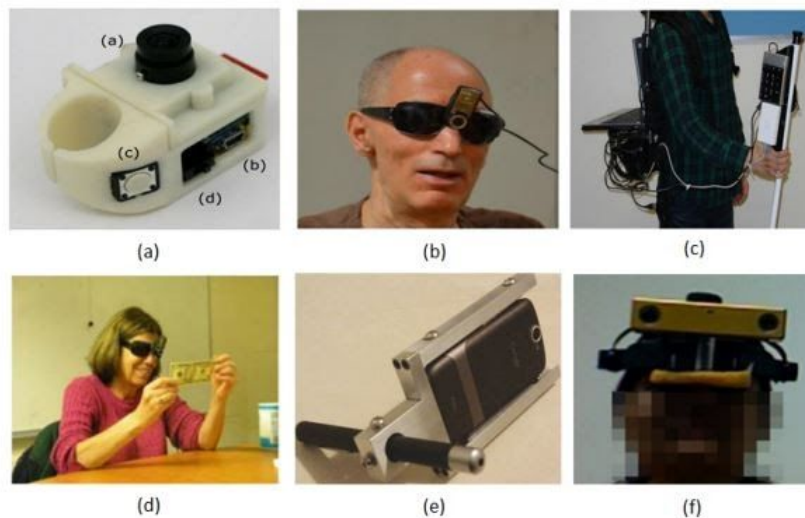


1.3 Assistive technologies

Several assistive methods (vision substitution) have been developed so far, these methods can be categorized as: rfid based methods, sonar based method, image processing based method and computer vision based methods. Among them computer vision based methods are most promising for object recognition application. Visual impairment can be categorized into two parts as below:

- Low vision is the case when visual acuity is less than 6/18 but more than or equal to 6/60.
- A person is considered as blind when visual acuity is less than 6/120.

We are all witness to technological advancement in recent years, which motivates us to develop a system for less privileged groups of people. Several electronic assistive systems have been developed so far, but there are very few which are using computer vision, but vision based systems are gaining momentum in recent research. In figure 1.3 already developed assistive systems using computer vision is portrayed, figure 1.3(a) is a camera device which is developed to wear on finger and can be pointed in desired direction [17], in image figure 1.3(b) and (d) a camera mounted on glasses system is designed which works like an eye in between two eyes, so a person can rotate his head in potential object direction. In figure 1.3(c) and (f) a stereo vision cane is shown which employs 3D imaging to get depth data. Figure 1.3(e) depicted a virtual smart cane [22] which contains a laser and a smart phone with vibration mechanism. Some of these systems are developed for navigation and some are for particular applications like banknote recognition as in 1.3(d).



1.4 Motivation

In a recent survey it is estimated that 285 million people worldwide are visually impaired, among them 246 million have low vision and 39 million are blind. About 65% of all people who are visually impaired are aged 50 and older, with an increasing elderly population in many countries, more people will be at risk of visual impairment due to chronic eye diseases and ageing processes. More surprisingly, 19 million children are estimated as visually impaired and among them 1.4 million are irreversibly blind. It is estimated that approximately 90% of the visually impaired people live in developing countries like China, India and so on. The facts about visual impairment is surprising on one hand, because despite enormous medical efforts to wipe out the visual impairment, the number is still breathtaking. On the other hand these facts encourage us to develop object and color recognition based assistance for visually impaired people.

1.5 Objective

Image classification is a challenging problem and needed to perform almost every task in this world, hence in this work we emphasize on object recognition based assistive methods. Apart from image classification, the classified text is converted to speech using python & the environment used for Android Development is Android Studio.

The objective of the project is:

- To develop an image classification app by training pretrained models using an online available dataset through a teachable machine.
- To increase the accuracy by using clear & precise images in the dataset & also increase the number of classes.
- To convert text to speech using python, so that description of classified text can be heard by the visually impaired.
- To build an app with all these features, the Android development environment used is Android Studio.
- To change the layout of the app using Android Studio.

Chapter 2

2.1 Developer Tools & Technologies used for building the App

- Teachable Machine (<https://teachablemachine.withgoogle.com/>)
- TensorFlow Lite
- Android Studio (Version used 4.0.0)
- Python (To convert text to speech)

2.2 Teachable Machine

Train a computer to recognize your own images, sounds, & poses.

A fast, easy way to create machine learning models for your sites, apps, and more – no expertise or coding required.

2.2.1 Works With...

- TensorFlow.js
- ml5.js
- p5.js
- Coral
- Framer
- Node.js
- Glitch

1. TensorFlow.js

TensorFlow.js is a JavaScript Library for training and deploying machine learning models in the browser and in Node.js. Comfortable with concepts like Tensors, Layers, Optimizers and Loss Functions (or willing to get comfortable with them)? TensorFlow.js provides flexible building blocks for neural network programming in JavaScript. Want to get started with Machine Learning but not worry about any low level details like Tensors or Optimizers? Built on top of TensorFlow.js, the ml5.js library provides access to machine learning algorithms and models in the browser with a concise, approachable API.

There are two main ways to get TensorFlow.js in your browser based projects:

- Using script tags.
- Installation from NPM and using a build tool like Parcel, WebPack, or Rollup..

1.1 How it works

- **Run existing models**
 - Use official TensorFlow.js models
 - Convert Python models
- **Retrain existing models**
 - Retrain pre-existing ML models using your own data (use transfer learning to customize models)
- **Develop ML with JavaScript**
 - Build & train models directly in JavaScript using flexible & intuitive APIs.

2. ml5.js

ml5.js is machine learning *for the web* in your web browser. Through some clever and exciting advancements, the folks building TensorFlow.js figured out that it is possible to use the web browser's built in graphics processing unit (GPU) to do calculations that would otherwise run very slowly using a central processing unit (CPU). ml5 strives to make all these new developments in machine learning on the web more approachable for everyone. In this project I have used a pre-trained

model called MobileNet -- a machine learning model trained to recognize the content of certain images -- in ml5.js.

The fastest way to get started exploring the creative possibilities of ml5.js are to either:

- Download a ml5.js project boilerplate. You can download a zip file here: ml5 project boilerplate.. Or...
- ml5.js has been designed to play very nicely with p5. Open the p5 web editor sketch with ml5.js added.
- You can also copy and paste the cdn link to the ml5 library here

```
<script src="https://unpkg.com/ml5@0.4.3/dist/ml5.min.js"></script>
```

3. p5.js

p5.js is a JavaScript library for creative coding, with a focus on making coding accessible and inclusive for artists, designers, educators, beginners, and anyone else! p5.js is free and open-source and is accessible to everyone. Using the metaphor of a sketch, p5.js has a full set of drawing functionality. However, you're not limited to your drawing canvas. You can think of your whole browser page as your sketch, including HTML5 objects for text, input, video, webcam, and sound.

4. Coral

Coral is a complete toolkit to build products with local AI. One of the projects of coral is the Embedded Teachable MachineA machine that can quickly learn to recognize new objects by re-training a vision classification model directly on your device.

5. Framer

It's the prototyping tool that gives your work a competitive edge. Without sacrificing on speed or quality.

6. Node.js

Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine. As an Asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications. Many connections can be handled concurrently. Upon each connection, the callback is fired, but if there is no work to be done, Node.js will sleep.

7. Glitch

Glitch is the friendly community where everyone codes together. Used to Build fast, full-stack web apps in your browser.

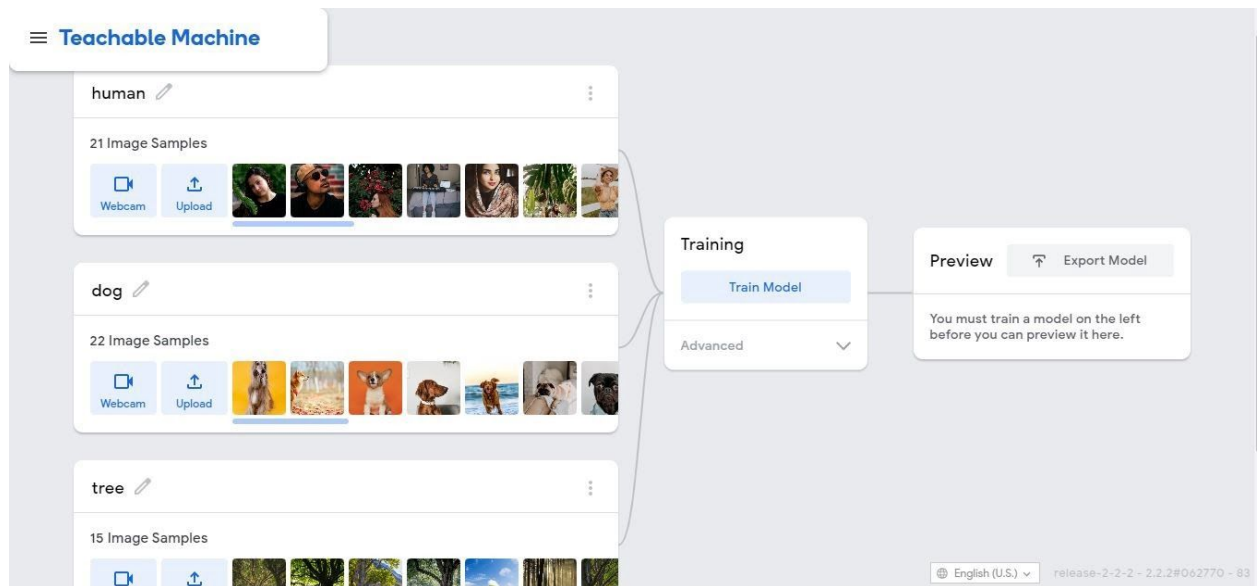
2.3 Steps for Building the TensorFlow Lite Android App

Step 1

- **Gather**

Gather and group the examples into classes, or categories, that you want the computer to learn. Here I have used 13 classes such as car, day, dog, door_open, door_closed, human, human_with_mask, night, shop_closed, stairs, tree, tap, zebra_crossing . Just gather the dataset and click on upload to upload the images. A webcam can also be used.

Note: The TFLite Android App example supports models with 3 or more classes.



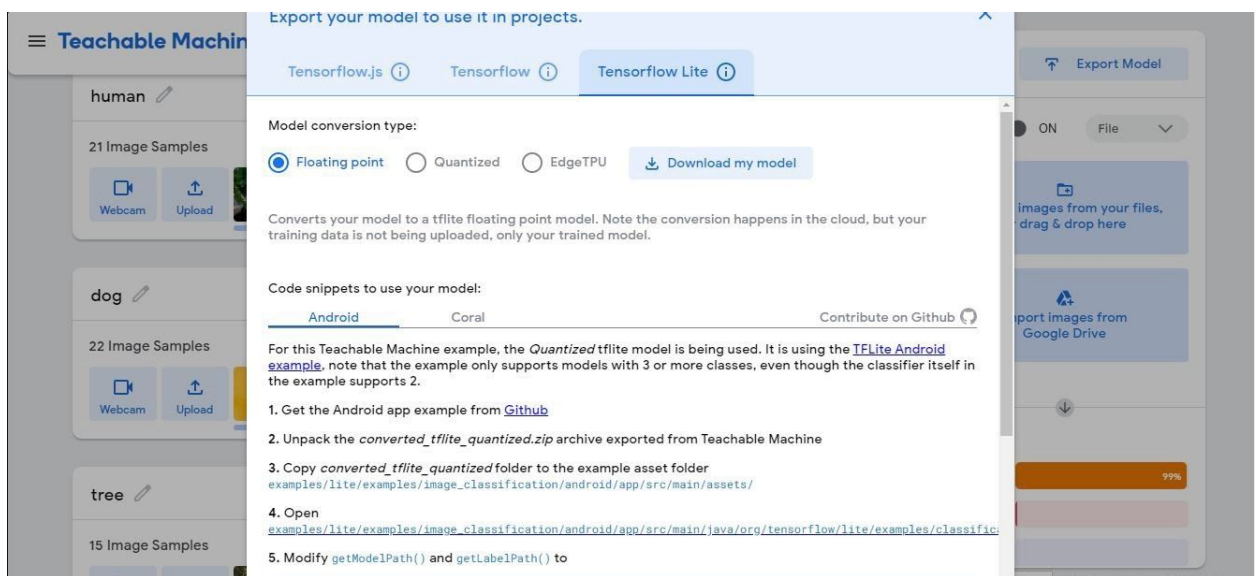
- **Train the model**

Train the model, then instantly test it out to see whether it can correctly classify new examples. After training the model go to the preview option to check the accuracy you can either use a webcam or drop files from any folder.



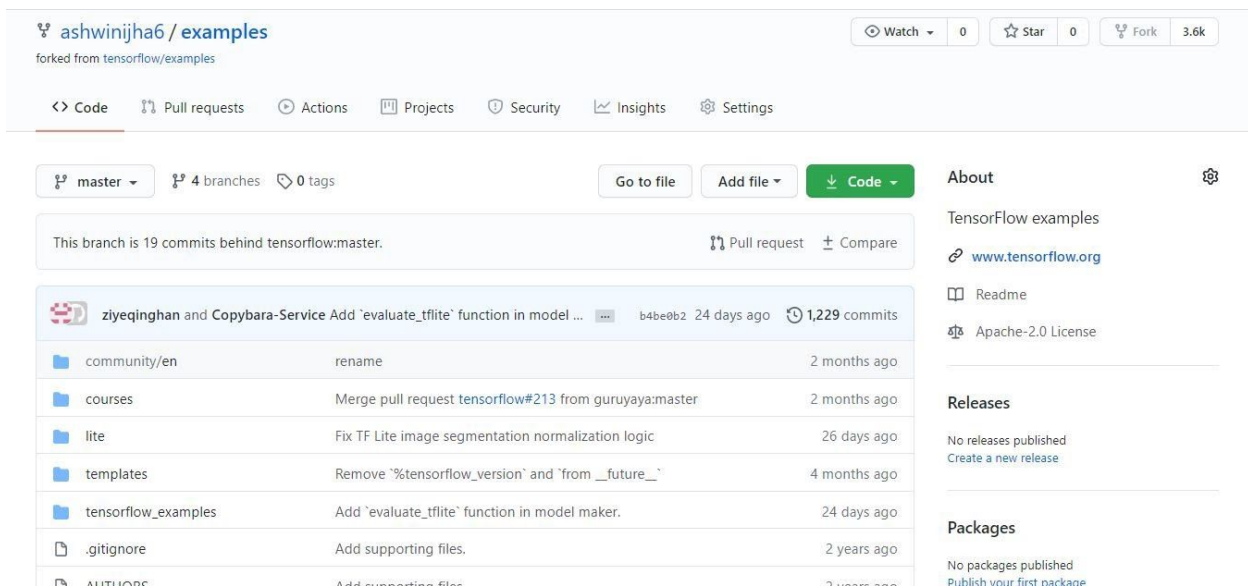
- **Export the model**

Export the model for projects: sites, apps, and more. We can download your model or host it online for free. First select the Floating point option & then click on Download my model to download the model then repeat the same with the Quantized model.



Step 2

TensorFlow Github Repository has been used for this project. Fork this Github repository (<https://github.com/tensorflow/examples.git>) & clone it using the command **git clone** <https://github.com/your-username/examples.git>



Step 3

After cloning the repository move to this location
[examples/lite/examples/image_classification/android/app/src/main/assets](#)
 in the assets directory :-

- i) You can find a **labels.txt** & **labels_without_background.txt** file just delete the **labels_without_background.txt** & update the **labels.txt** with the class names.
- ii) Paste the two model tflite files that you downloaded earlier **model_unquant.tflite** & **model.tflite**.

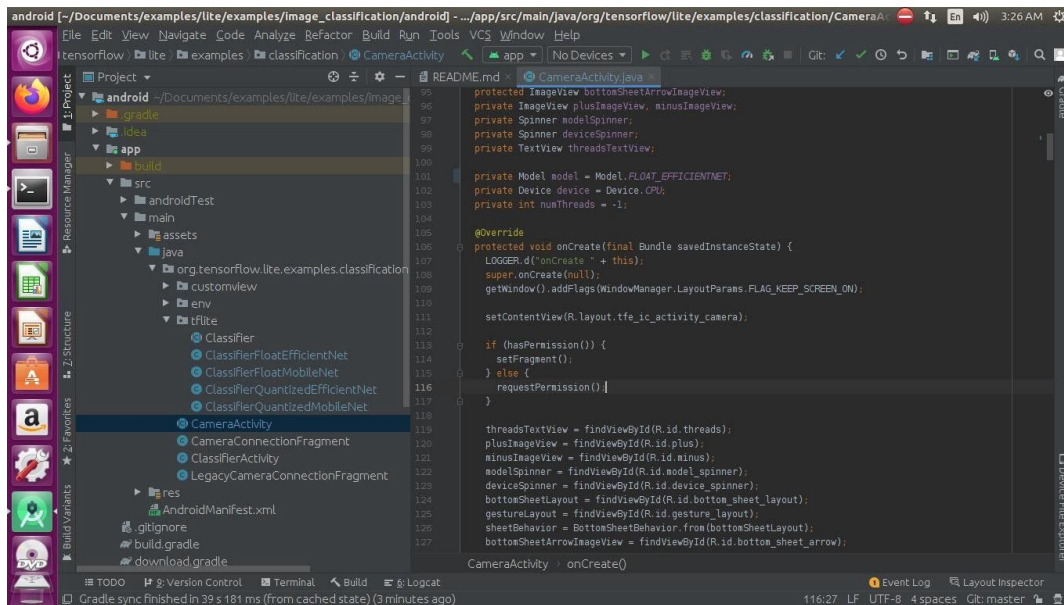
Step 4

- i) Open Android Studio & click on **Open an existing project** move to the path
[examples/lite/examples/image_classification/android](#) & click **OK**. This opens up the code in android studio.
- ii) Let the Gradle Sync Start & when it shows the message Daemon started successfully & [examples/lite/examples/image_classification/android](#) Sync Finished, Make the following changes in the code.

Step 5

Go to this location using the project tab in Android Studio
[android/app/src/main/java/org/tensorflow/lite/examples/classification/](#) &
 make the following changes:-

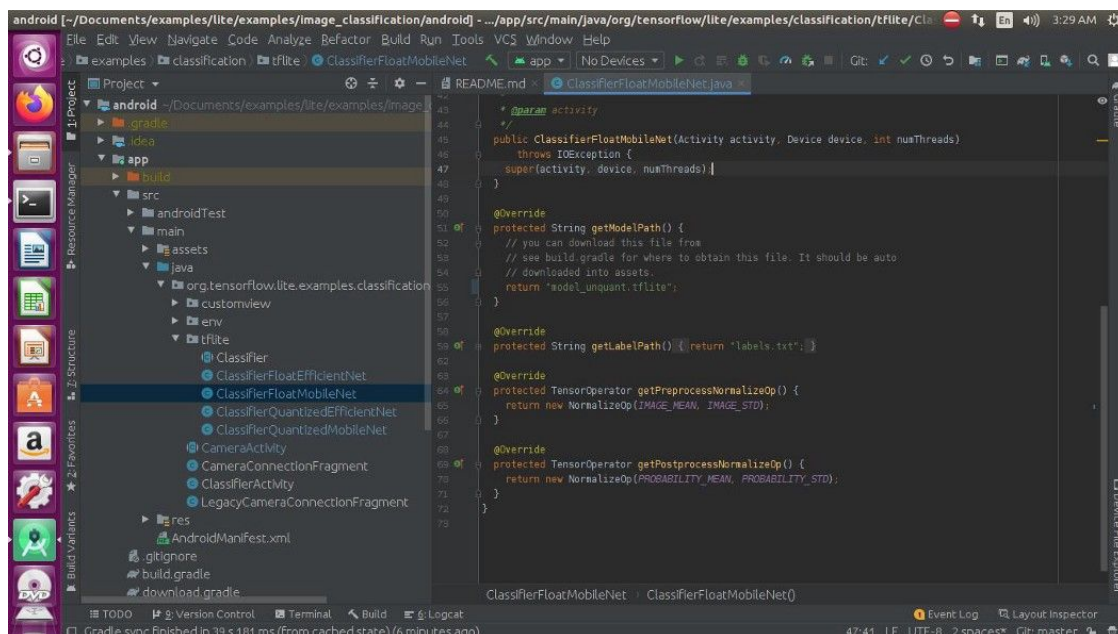
i) There you will find the **CameraActivity.java** change **Model.QUANTIZED_EFFICIENTNET** to **Model.FLOAT_EFFICIENTNET** in line **101**.



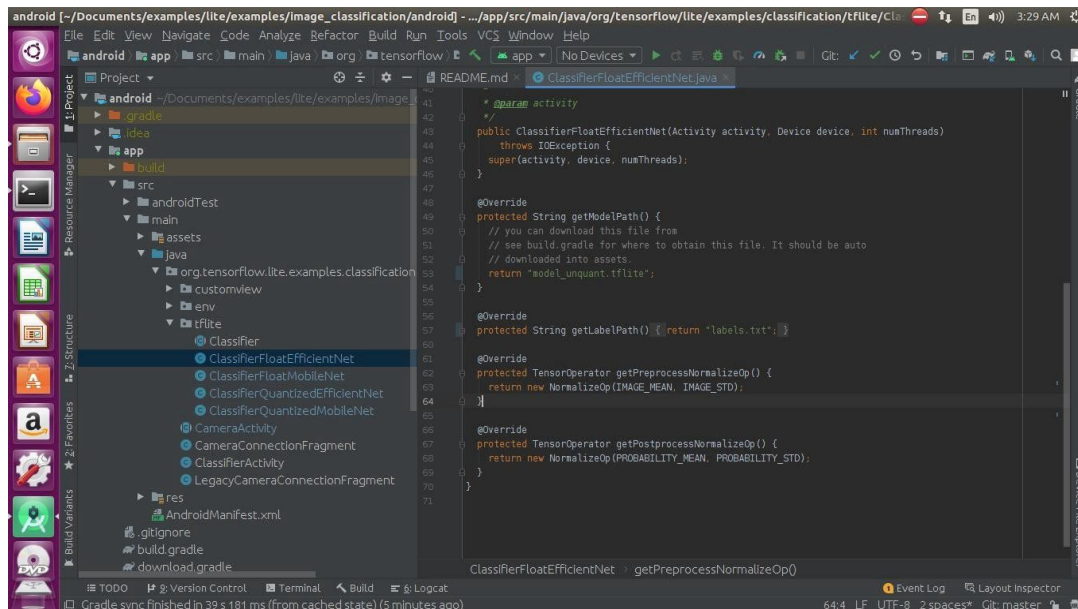
Move to this location

android/app/src/main/java/org/tensorflow/lite/examples/classification/tflite/ & make the following changes:-

i) In the **ClassifierFloatMobileNet.java** file change the return model to **'model_unquant.tflite'** as shown in **line 55**.

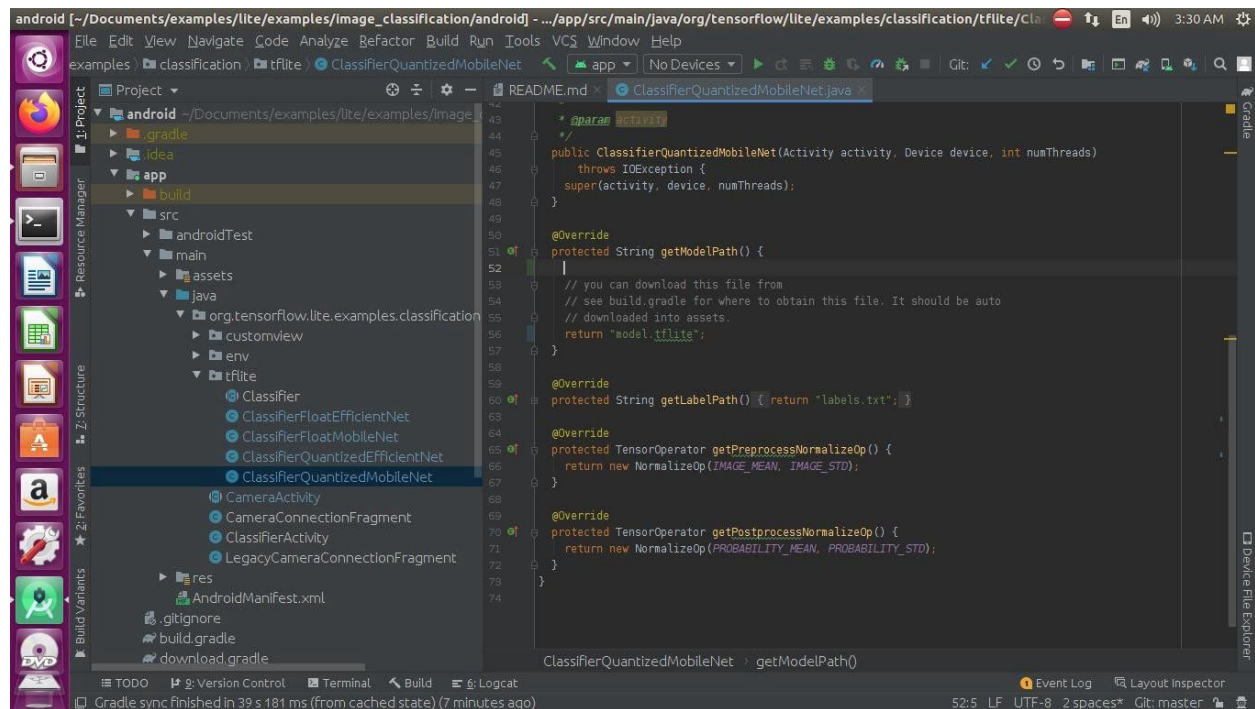


ii) In the **ClassifierFloatEfficientNet.java** file change the return model to **'model_unquant.tflite'** as shown in **line 53** & return **labels.txt** in **line 57**.



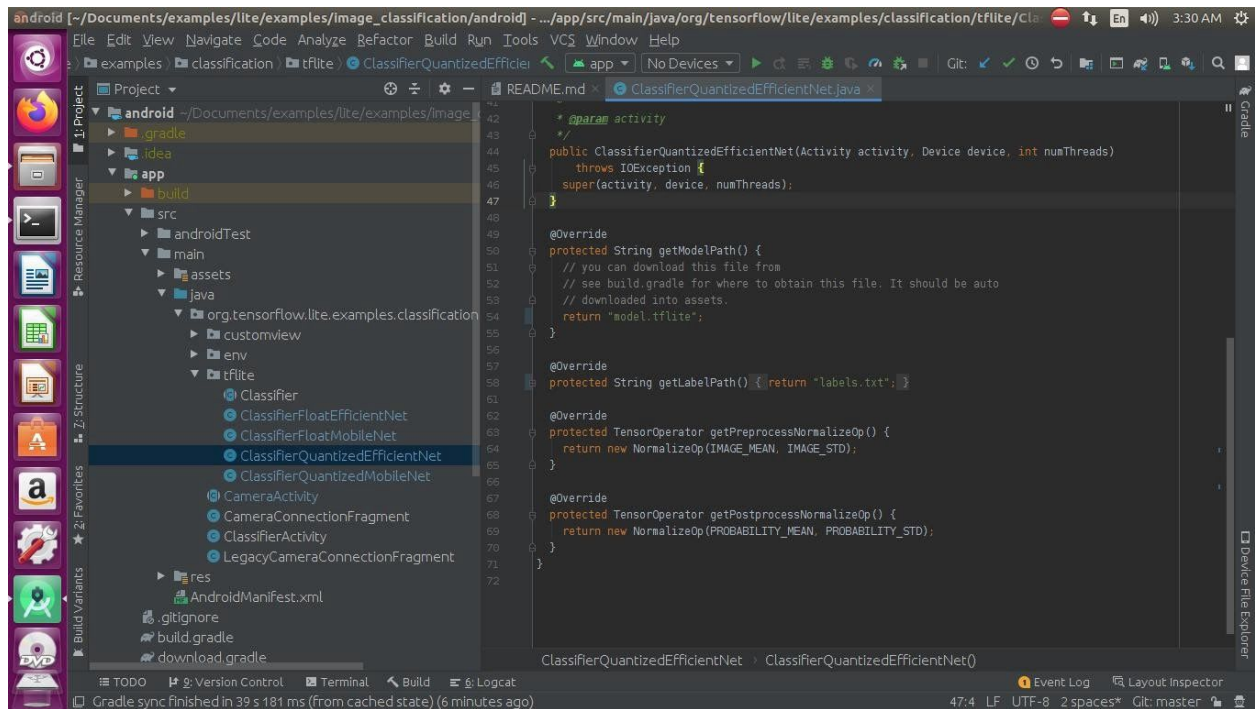
```
41  * @param activity
42  */
43  public ClassifierFloatEfficientNet(Activity activity, Device device, int numThreads)
44      throws IOException {
45      super(activity, device, numThreads);
46  }
47
48  @Override
49  protected String getModelPath() {
50      // you can download this file from
51      // see build.gradle for where to obtain this file. It should be auto
52      // downloaded into assets.
53      return "model_unquant.tflite";
54  }
55
56  @Override
57  protected String getLabelPath() {return "labels.txt";}
58
59  @Override
60  protected TensorOperator getPreprocessNormalizeOp() {
61      return new NormalizeOp(IMAGE_MEAN, IMAGE_STD);
62  }
63
64  @Override
65  protected TensorOperator getPostprocessNormalizeOp() {
66      return new NormalizeOp(PROBABILITY_MEAN, PROBABILITY_STD);
67  }
68
69  }
70
71  ClassifierFloatEfficientNet > getPreprocessNormalizeOp()
```

iii) In the **ClassifierQuantizedMobileNet.java** file change the return model to **'model.tflite'** as shown in **line 56**.



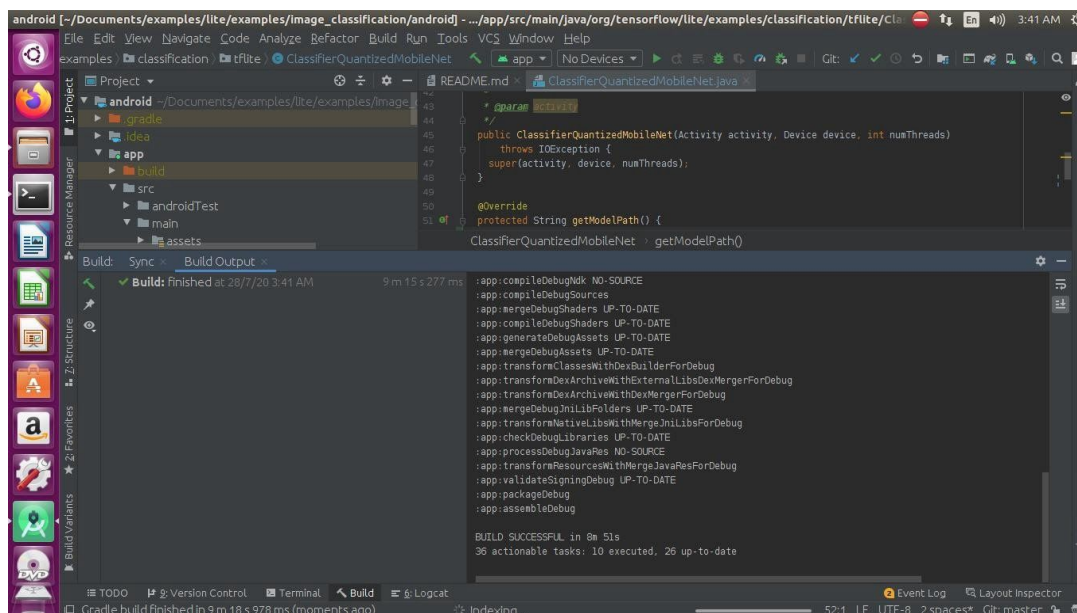
```
43  * @param activity
44  */
45  public ClassifierQuantizedMobileNet(Activity activity, Device device, int numThreads)
46      throws IOException {
47      super(activity, device, numThreads);
48  }
49
50  @Override
51  protected String getModelPath() {
52      // you can download this file from
53      // see build.gradle for where to obtain this file. It should be auto
54      // downloaded into assets.
55      return "model.tflite";
56  }
57
58  @Override
59  protected String getLabelPath() {return "labels.txt";}
60
61  @Override
62  protected TensorOperator getPreprocessNormalizeOp() {
63      return new NormalizeOp(IMAGE_MEAN, IMAGE_STD);
64  }
65
66  @Override
67  protected TensorOperator getPostprocessNormalizeOp() {
68      return new NormalizeOp(PROBABILITY_MEAN, PROBABILITY_STD);
69  }
70
71  }
72
73  ClassifierQuantizedMobileNet > getModelPath()
```

iv) In the **ClassifierQuantizedEfficientNet.java** file change the return model to 'model.tflite' as shown in **line 54** & return **labels.txt** in **line 58**.



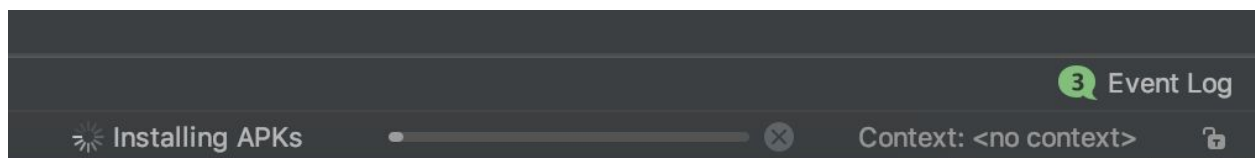
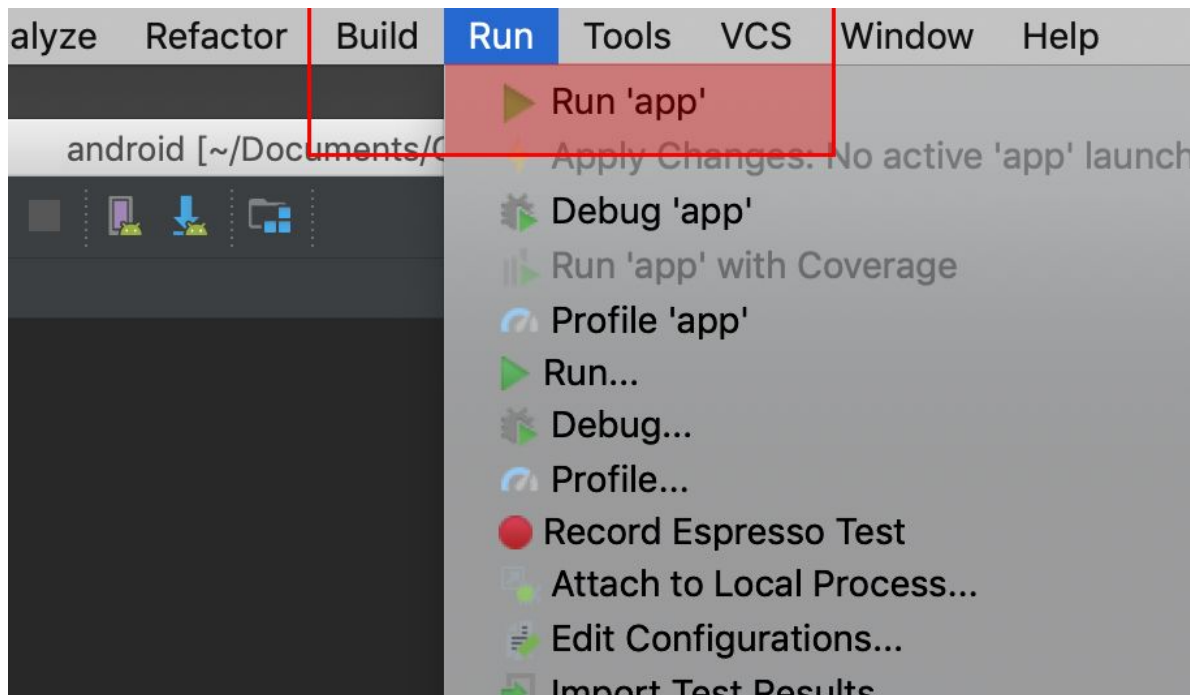
Step 6

Now click on **Build** & then **Make project** it shows **Gradle Build is Running**. Let the **Project Build Successfully**.

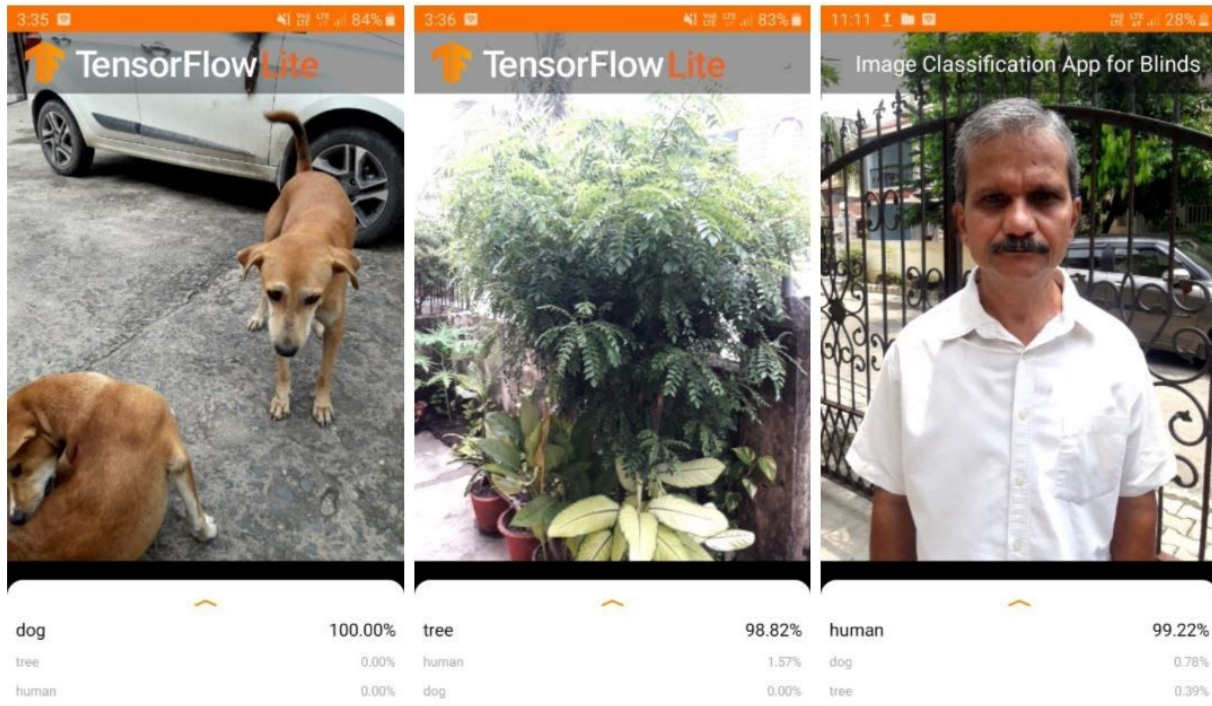


Step 7

- i) Connect the android device make sure that the android device is in Developer Mode (<https://developer.android.com/studio/debug/dev-options>). Stake Awake & USB Debugging need to be turned on in Developer Mode.
- ii) The Android Studio will recognize the Android Device. Now you can Run the App by clicking on Run then Run 'app' this will install the App (APK) in your Android Device.



2.4 Results



Note : This Chapter ends with the App being able to classify the data set used for classifying.

Chapter 3

3.1 Convert text into natural sounding speech using Python.

3.1.1 Speech Synthesis

The process of translating text input into audio data is called *synthesis* and the output of synthesis is called *synthetic speech*. Text-to-Speech takes two types of input: raw text or SSML-formatted data.

3.1.2 Text-to-Speech

- Allows developers to create natural-sounding, synthetic human speech as playable audio. You can use the audio data files you create using

Text-to-Speech to power your applications or augment media like videos or audio recordings.

- Converts Text or Speech Synthesis Markup Language (SSML) input into audio data like MP3 or LINEAR16 (the encoding used in WAV files), and we will learn it here using Python.
- Text-to-Speech is ideal for any application that plays audio of human speech to users. It allows you to convert arbitrary strings, words, and sentences into the sound of a person speaking the same things. Imagine that you have a voice assistant app that provides natural language feedback to your users as playable audio files. Your app might take an action and then provide human speech as feedback to the user.
- With Text-to-Speech, you can convert that response string to actual human speech to play back to the user.

3.1.3 Let's bring Python into the picture

There are a lot of APIs out there that offer this service, one of the commonly used services is Google Text to Speech, in this article, we will play around with it, along with another offline library: pyttsx3.

We will see 2 types of Synthesis here :

- ***Online Synthesis***
- ***Offline Synthesis***

3.1.4 Let's get started with pip

1. pip is the [package installer](#) for Python. You can use pip to install packages from the [Python Package Index](#) and other indexes.
2. To install pip3 on Ubuntu or Debian Linux, open a new Terminal window and enter `sudo apt-get install python3-pip`
3. Install the required modules using : `pip3 install gTTS pyttsx3 playsound`

3.2 gTTS

gTTS (Google Text-to-Speech), a Python library and CLI tool to interface with Google Translate's text-to-speech API. Write spoken mp3 data to a file, a file-like object (bytestring) for further audio manipulation, or stdout.

3.3 pyttsx3

pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline, and is compatible with both Python 2 and 3.

3.4 playsound

Pure Python, cross platform, single function module with no dependencies for playing sounds. The playsound module contains only one thing — the function (also named) playsound. It requires one argument — the path to the file with the sound you'd like to play. This may be a local file, or a URL.

3.5 Online Synthesis

- Open a new python file you can use any text editor like gedit or so on, using the terminal type the command `gedit online_tts.py`

- Next import :

import gtts

from playsound import playsound

- It's pretty straightforward to use this library, you just need to pass text to gTTS object that is an interface to [Google Translate](#)'s Text-to-Speech API

#make request to google to get synthesis

tts = gtts.gTTS("Hi, This is Ashwini Jha")

- The text has been sent & the actual audio file has been retrieved, Save the file

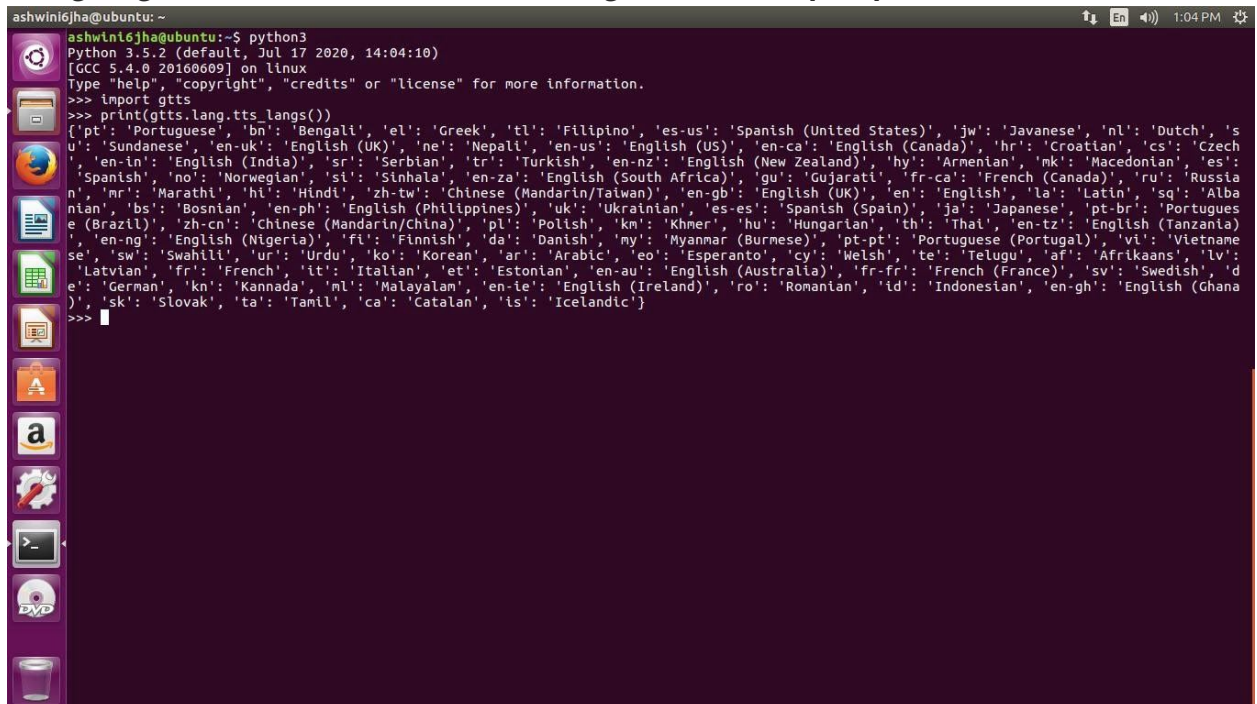
tts.save("hi.mp3")

- Now let's play the mp3 file using playsound module

playsound("hi.mp3")

- To use other languages, pass the lang parameter (languages are shown in the picture below)

tts = gtts.gTTS("Hi, This Ashwini Jha", lang="es") #es is for spanish



```
ashwini6jha@ubuntu: ~$ python3
Python 3.5.2 (default, Jul 17 2020, 14:04:10)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>> import gtts
>>> print(gtts.lang.tts_langs())
{'pt': 'Portuguese', 'bn': 'Bengali', 'el': 'Greek', 'tl': 'Filipino', 'es-us': 'Spanish (United States)', 'jw': 'Javanese', 'nl': 'Dutch', 'su': 'Sundanese', 'en-uk': 'English (UK)', 'ne': 'Nepali', 'en-us': 'English (US)', 'en-ca': 'English (Canada)', 'hr': 'Croatian', 'cs': 'Czech', 'en-in': 'English (India)', 'sr': 'Serbian', 'tr': 'Turkish', 'en-nz': 'English (New Zealand)', 'hy': 'Armenian', 'mk': 'Macedonian', 'es': 'Spanish', 'no': 'Norwegian', 'si': 'Sinhala', 'en-za': 'English (South Africa)', 'gu': 'Gujarati', 'fr-ca': 'French (Canada)', 'ru': 'Russian', 'mr': 'Marathi', 'hi': 'Hindi', 'zh-tw': 'Chinese (Mandarin/Taiwan)', 'en-gb': 'English (UK)', 'en': 'English', 'la': 'Latin', 'sq': 'Albanian', 'bs': 'Bosnian', 'en-ph': 'English (Philippines)', 'uk': 'Ukrainian', 'es-es': 'Spanish (Spain)', 'ja': 'Japanese', 'pt-br': 'Portuguese (Brazil)', 'zh-cn': 'Chinese (Mandarin/China)', 'pl': 'Polish', 'km': 'Khmer', 'hu': 'Hungarian', 'th': 'Thai', 'en-tz': 'English (Tanzania)', 'en-ng': 'English (Nigeria)', 'fi': 'Finnish', 'da': 'Danish', 'my': 'Myanmar (Burmese)', 'pt-pt': 'Portuguese (Portugal)', 'vi': 'Vietnamese', 'sw': 'Swahili', 'ur': 'Urdu', 'ko': 'Korean', 'ar': 'Arabic', 'eo': 'Esperanto', 'cy': 'Welsh', 'te': 'Telugu', 'af': 'Afrikaans', 'lv': 'Latvian', 'fr': 'French', 'it': 'Italian', 'et': 'Estonian', 'en-au': 'English (Australia)', 'fr-fr': 'French (France)', 'sv': 'Swedish', 'de': 'German', 'kn': 'Kannada', 'ml': 'Malayalam', 'en-ie': 'English (Ireland)', 'ro': 'Romanian', 'id': 'Indonesian', 'en-gh': 'English (Ghana)', 'sk': 'Slovak', 'ta': 'Tamil', 'ca': 'Catalan', 'is': 'Icelandic'}
```

- To get the list of all the other languages, just print ***print(gtts.lang.tts_lang())***

3.6 Offline Synthesis

Now you know how to use Google's API, but what if you want to use text to speech technologies offline ? Well, pyttsx3 library comes to the rescue.

- To get started, Open up a new python file using the terminal & gedit : ***gedit offline_tts.py***
- Import the pyttsx3 library : ***import pyttsx3***
- Initialize the TTS engine : ***engine = pyttsx3.init()***
- To convert text to speech use say() & runAndWait(): say() method adds an utterance to speak to the event queue & runAndWait() method runs the actual event loop until all commands are queued up.

engine.say(text)

- engine.runAndWait()***

- Get details of all voices available on your machine (as shown below)
- ```
voice = engine.getProperty("voices")
print("voices")
```

```

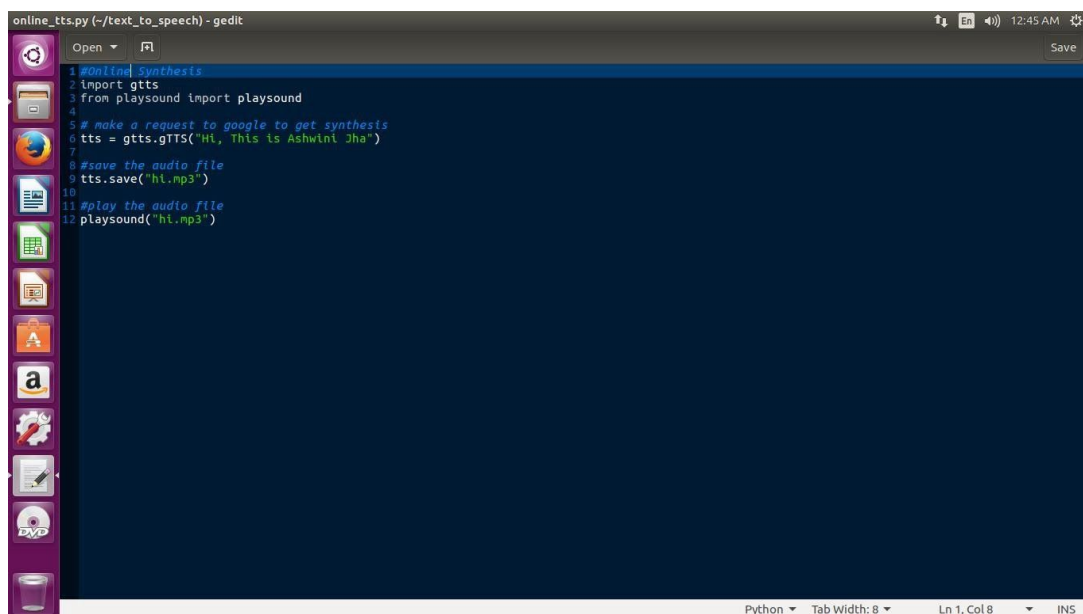
ashwin61jha@ubuntu:~$ python3
Python 3.5.2 (default, Jul 17 2020, 14:04:10)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import pytttsx3
>>> engine = pytttsx3.init()
>>> voices = engine.getProperty("voices")
>>> print(voices)
[<pytttsx3.voice.Voice object at 0x7f45c615d1d0>, <pytttsx3.voice.Voice object at 0x7f45c615d240>, <pytttsx3.voice.Voice object at 0x7f45c5c50a20>, <pytttsx3.voice.Voice object at 0x7f45c5c50a80>, <pytttsx3.voice.Voice object at 0x7f45c5c50b70>, <pytttsx3.voice.Voice object at 0x7f45c5c50c10>, <pytttsx3.voice.Voice object at 0x7f45c5c50cd0>, <pytttsx3.voice.Voice object at 0x7f45c5c50d60>, <pytttsx3.voice.Voice object at 0x7f45c5c50e10>, <pytttsx3.voice.Voice object at 0x7f45c5c50eb0>, <pytttsx3.voice.Voice object at 0x7f45c5c50ef0>, <pytttsx3.voice.Voice object at 0x7f45c5c50f20>, <pytttsx3.voice.Voice object at 0x7f45c5c50f80>, <pytttsx3.voice.Voice object at 0x7f45c5c50fd0>, <pytttsx3.voice.Voice object at 0x7f45c5c50ff0>, <pytttsx3.voice.Voice object at 0x7f45c5c51000>, <pytttsx3.voice.Voice object at 0x7f45c5c51090>, <pytttsx3.voice.Voice object at 0x7f45c5c512b0>, <pytttsx3.voice.Voice object at 0x7f45c5c512f0>, <pytttsx3.voice.Voice object at 0x7f45c5c51400>, <pytttsx3.voice.Voice object at 0x7f45c5c51430>, <pytttsx3.voice.Voice object at 0x7f45c5c514f0>, <pytttsx3.voice.Voice object at 0x7f45c5c51510>, <pytttsx3.voice.Voice object at 0x7f45c5c515f0>, <pytttsx3.voice.Voice object at 0x7f45c5c51630>, <pytttsx3.voice.Voice object at 0x7f45c5c516b0>, <pytttsx3.voice.Voice object at 0x7f45c5c51710>, <pytttsx3.voice.Voice object at 0x7f45c5c517b0>, <pytttsx3.voice.Voice object at 0x7f45c5c517f0>, <pytttsx3.voice.Voice object at 0x7f45c5c518b0>, <pytttsx3.voice.Voice object at 0x7f45c5c51980>, <pytttsx3.voice.Voice object at 0x7f45c5c51a20>, <pytttsx3.voice.Voice object at 0x7f45c5c51a50>, <pytttsx3.voice.Voice object at 0x7f45c5c51b00>, <pytttsx3.voice.Voice object at 0x7f45c5c51b80>, <pytttsx3.voice.Voice object at 0x7f45c5c51bc0>, <pytttsx3.voice.Voice object at 0x7f45c5c51bd0>, <pytttsx3.voice.Voice object at 0x7f45c5c51c70>, <pytttsx3.voice.Voice object at 0x7f45c5c51d00>, <pytttsx3.voice.Voice object at 0x7f45c5c51d40>, <pytttsx3.voice.Voice object at 0x7f45c5c51d80>, <pytttsx3.voice.Voice object at 0x7f45c5c51e50>, <pytttsx3.voice.Voice object at 0x7f45c5c51f50>, <pytttsx3.voice.Voice object at 0x7f45c5c51f80>]
>>>

```

- To choose a voice speaker  
`engine.setProperty("voice", voices[1].id) #second voice speaker`  
`engine.say(text)`  
`engine.runAndWait()`

## 3.7 Run the python scripts using the respective commands

### 3.7.1 Online Synthesis : `python3 online_tts.py` (You can use the name of your .py script)



```
online_tts.py (-/text_to_speech) - gedit
1 #Online Synthesis
2 import gtts
3 from playsound import playsound
4
5 # make a request to google to get synthesis
6 tts = gtts.gTTS("Hi, This is Ashwini Jha")
7
8 #save the audio file
9 tts.save("hi.mp3")
10
11 #play the audio file
12 playsound("hi.mp3")
```

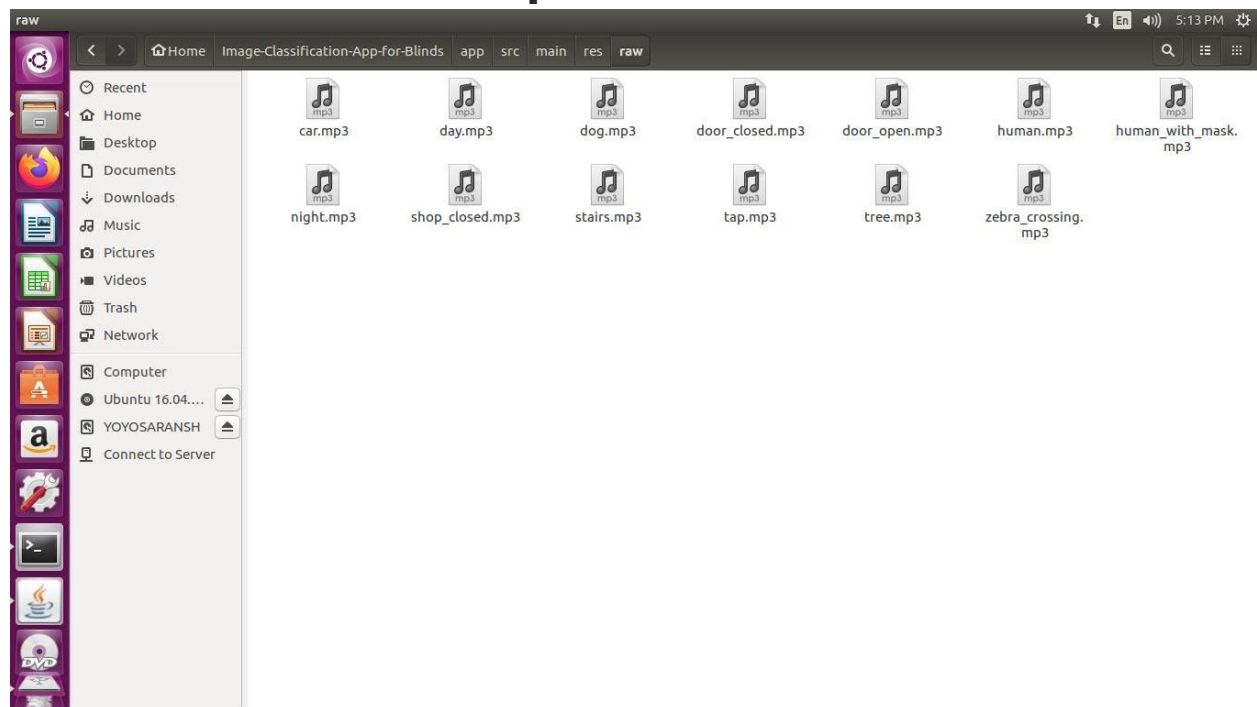
The screenshot shows a gedit text editor window titled "online\_tts.py (-/text\_to\_speech) - gedit". The editor contains a Python script for online text-to-speech synthesis. The script imports the 'gtts' module and the 'playsound' function from the 'playsound' module. It then creates a 'gtts.gTTS' object with the text "Hi, This is Ashwini Jha". The audio file is saved as "hi.mp3" and then played back using the 'playsound' function. The status bar at the bottom indicates the file is a Python script, the tab width is 8, and the cursor is at line 1, column 8.

### 3.7.2 Offline Synthesis : `python3 offline_tts.py` (You can use the name of your .py script)

```
offline_tts.py (~/.text_to_speech) - gedit
1 # Offline Synthesis
2 import pyttsx3
3
4 # initialize Text-to-speech engine
5 engine = pyttsx3.init()
6
7 # convert this text to speech
8 text = "It's a beautiful day today"
9
10 # get details of speaking rate
11 rate = engine.getProperty("rate")
12 print(rate)
13
14 # setting new voice rate (slower)
15 engine.setProperty("rate",100)
16
17 # for setting the voice raye (faster) just increase the rate
18 #engine.setProperty("rate",300)
19
20 engine.say(text)
21
22 # play the peech
23 engine.runAndWait()
```

Python Tab Width: 8 Ln 1, Col 20 INS

## 3.8 Result : Text-to-speech



**Note :** This Chapter ends, now as we have got the .mp3 files we can use them to add voice to the App.

## Chapter 4

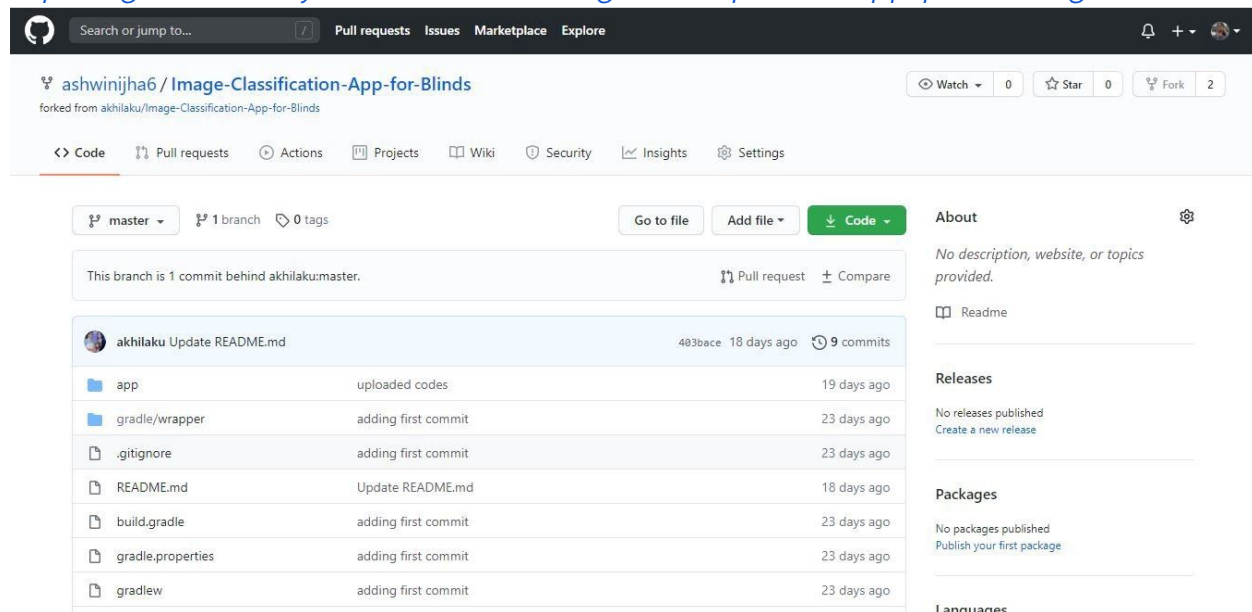
### 4.1 Adding Voice to the App

You must be thinking what is adding voice to the app, It is nothing but the **.mp3** files which you got as a result of *converting text to speech using python*, which we will add to the app. Let's see how.

#### Step1

Fork this [repository](#) & then clone it using `git clone`

[https://github.com/your\\_username/Image-Classification-App-for-Blinds.git](https://github.com/your_username/Image-Classification-App-for-Blinds.git)



#### Step 2

Delete the *models* & *labels.txt* (everything) present in [Image-Classification-App-for-Blinds/app/src/main/assets/](#)

#### Step 3

In the same location paste the *labels.txt* & the *models* Floating point & Quantized (downloaded from the teachable machine) as explained above.

#### Step 4

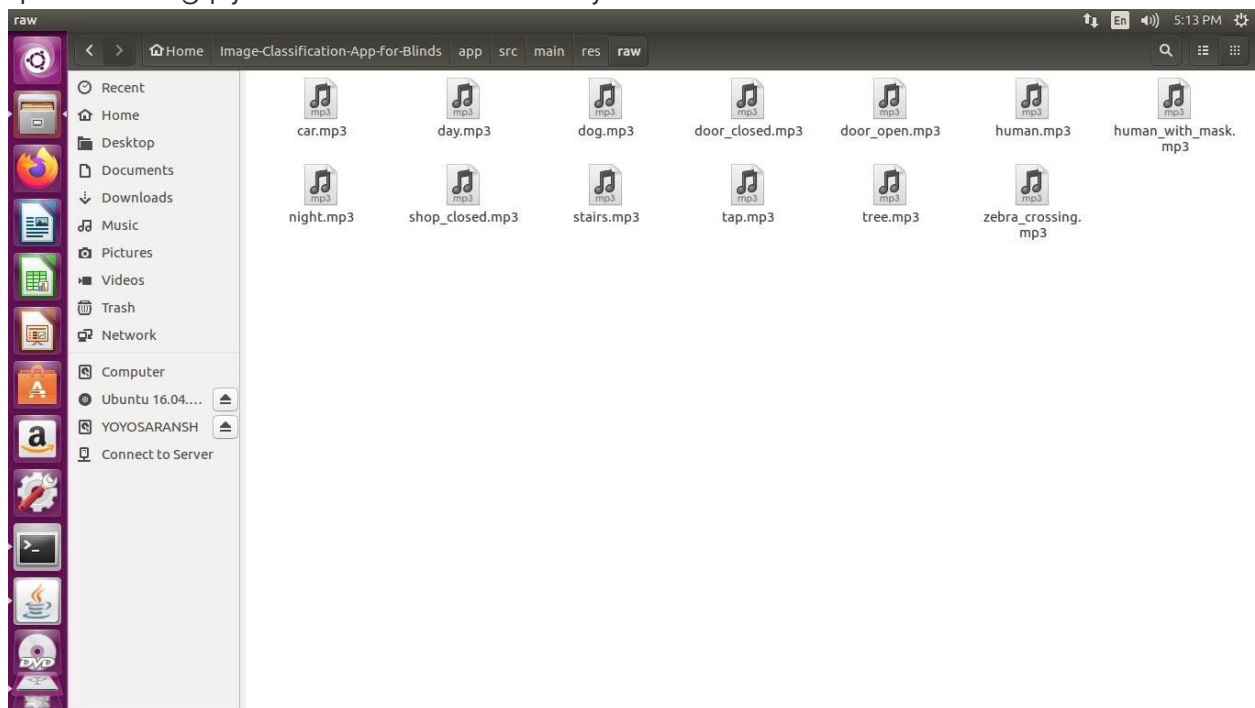
*Note :* You can even increase the accuracy & the number of classes & then paste the files as mentioned above (*labels.txt*, *model\_unquant.tflite*, *model.tflite*). For increasing the accuracy & classes see the section above "*Increasing the accuracy & classes*"

## Step 5

Now move to this location *Image-Classification-App-for-Blinds/app/src/main/res/* & create a directory *raw*.

## Step 6

Paste the *.mp3* files which you must have received as a result of converting text to speech using python in the *raw* directory.



## Step 7

Click on *Open an existing project* after opening *Android Studio* & move to the location of the repository you cloned earlier in your local machine & click *OK*.

## Step 8

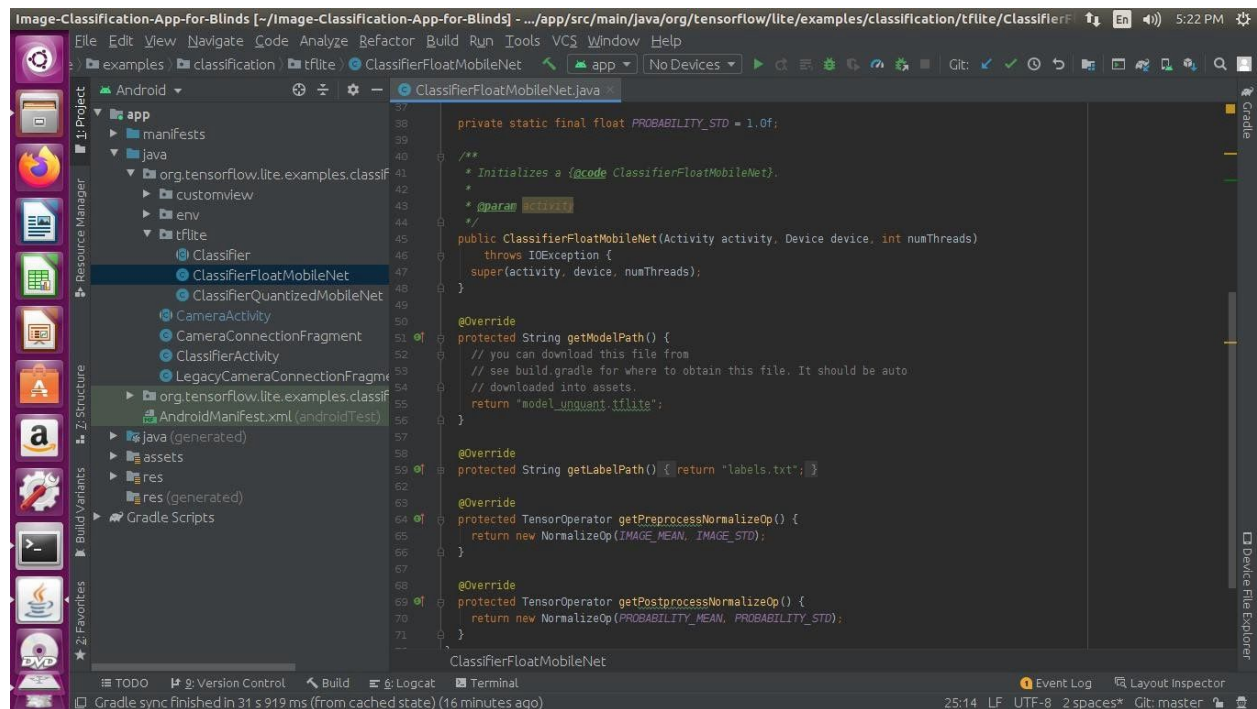
Once it opens in *Android Studio* wait for the *Gradle Sync Start* & when it shows the message *daemon started successfully* & *Sync finished*, Make the following changes in the code.

## Step 9



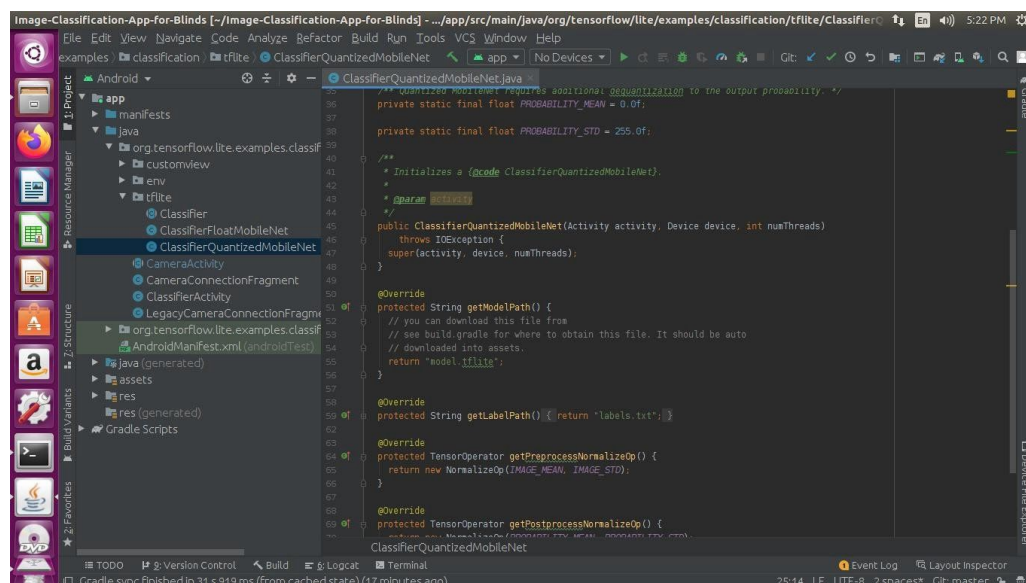
Move to this location

*Image-Classification-App-for-Blinds/app/src/main/java/org/tensorflow/lite/examples/classification/tflite/*, then in the *ClassifierFloatMobileNet.java* on line 55 return *"model\_unquant.tflite"* & on line 59 return *"labels.txt"*.



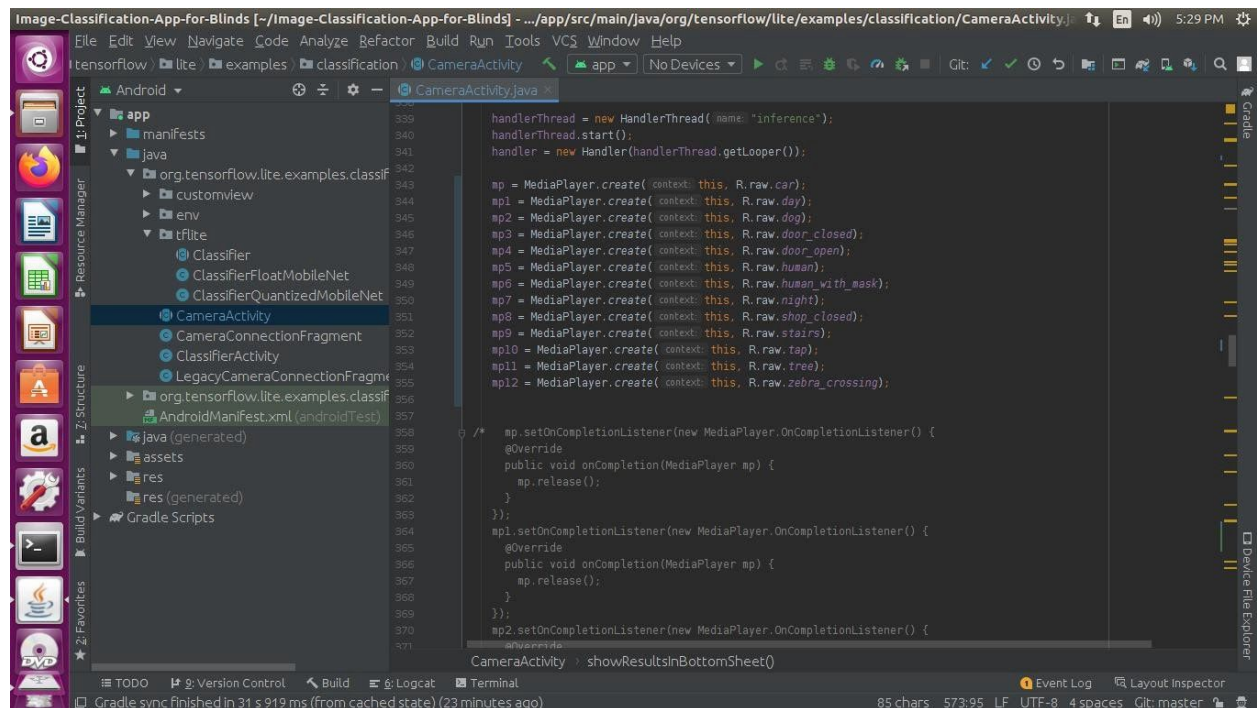
## Step 10

Then open the *ClassifierQuantizedMobileNet.java* & make the following changes, on line 55 return *"model.tflite"* & on line 59 return *"labels.txt"*.



## Step 11

Then make changes in the *CameraActivity.java*, from line 343 to 355 just replace (eg. *R.raw.car*) with *R.raw.your\_mp3\_file\_name* in the following location *Image-Classification-App-for-Blinds/app/src/main/java/org/tensorflow/lite/examples/classification/*. These are assigned to variable names (such as *mp*, *mp1*, *mp2* & so on.)

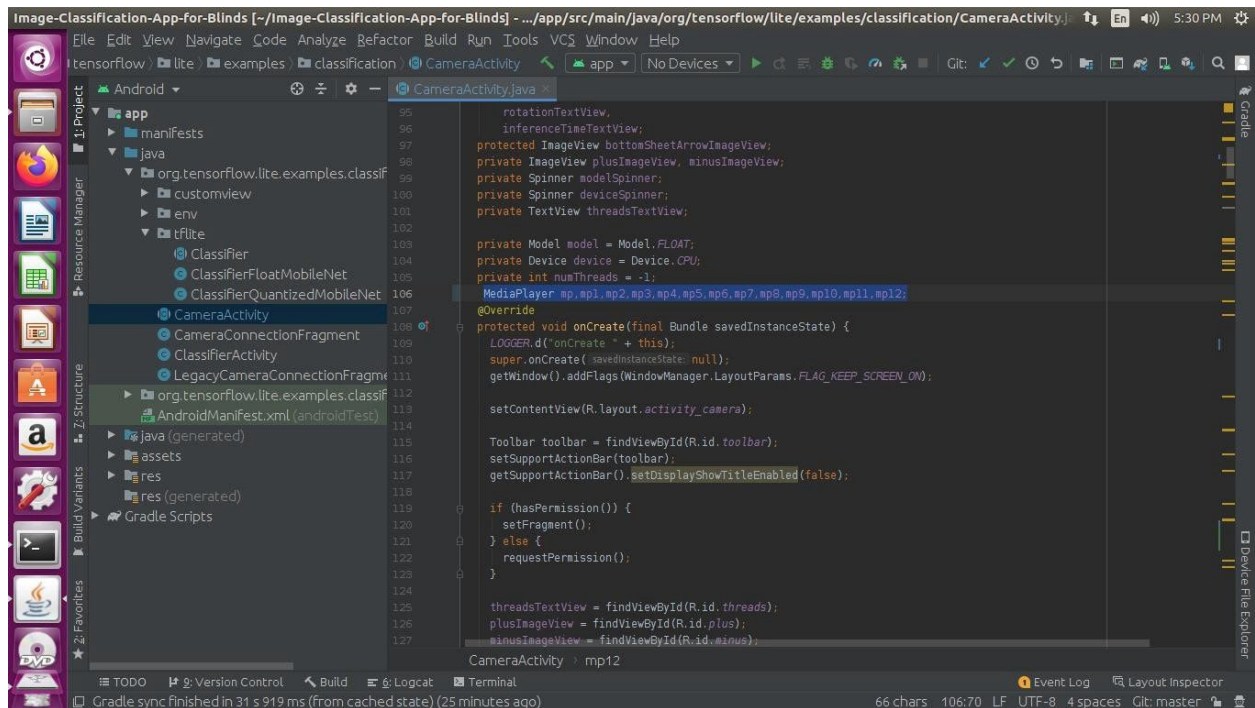


```
338 handlerThread = new HandlerThread("inference");
339 handlerThread.start();
340 handler = new Handler(handlerThread.getLooper());
341
342
343 mp = MediaPlayer.create(context, this, R.raw.car);
344 mp1 = MediaPlayer.create(context, this, R.raw.day);
345 mp2 = MediaPlayer.create(context, this, R.raw.dog);
346 mp3 = MediaPlayer.create(context, this, R.raw.door_closed);
347 mp4 = MediaPlayer.create(context, this, R.raw.door_open);
348 mp5 = MediaPlayer.create(context, this, R.raw.human);
349 mp6 = MediaPlayer.create(context, this, R.raw.human_with_mask);
350 mp7 = MediaPlayer.create(context, this, R.raw.night);
351 mp8 = MediaPlayer.create(context, this, R.raw.shop_closed);
352 mp9 = MediaPlayer.create(context, this, R.raw.stairs);
353 mp10 = MediaPlayer.create(context, this, R.raw.tap);
354 mp11 = MediaPlayer.create(context, this, R.raw.tree);
355 mp12 = MediaPlayer.create(context, this, R.raw.zebra_crossing);
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
```

Repeat the above (step 11) for *all the classes* used for classification.

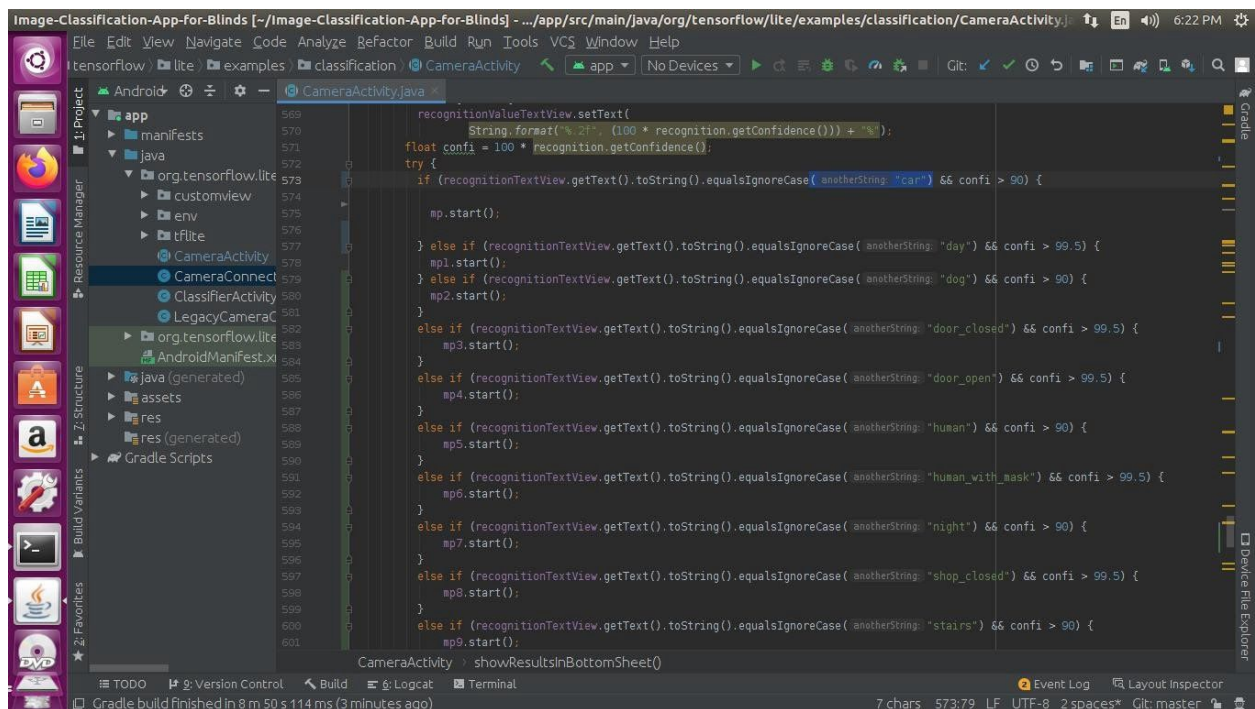
## Step 12

Update the variable names on line 106.



## Step 13

Now from line 573 to 610 just replace (eg. *anotherString: car*) with *anotherString: your\_label\_name* in CameraActivity.java.

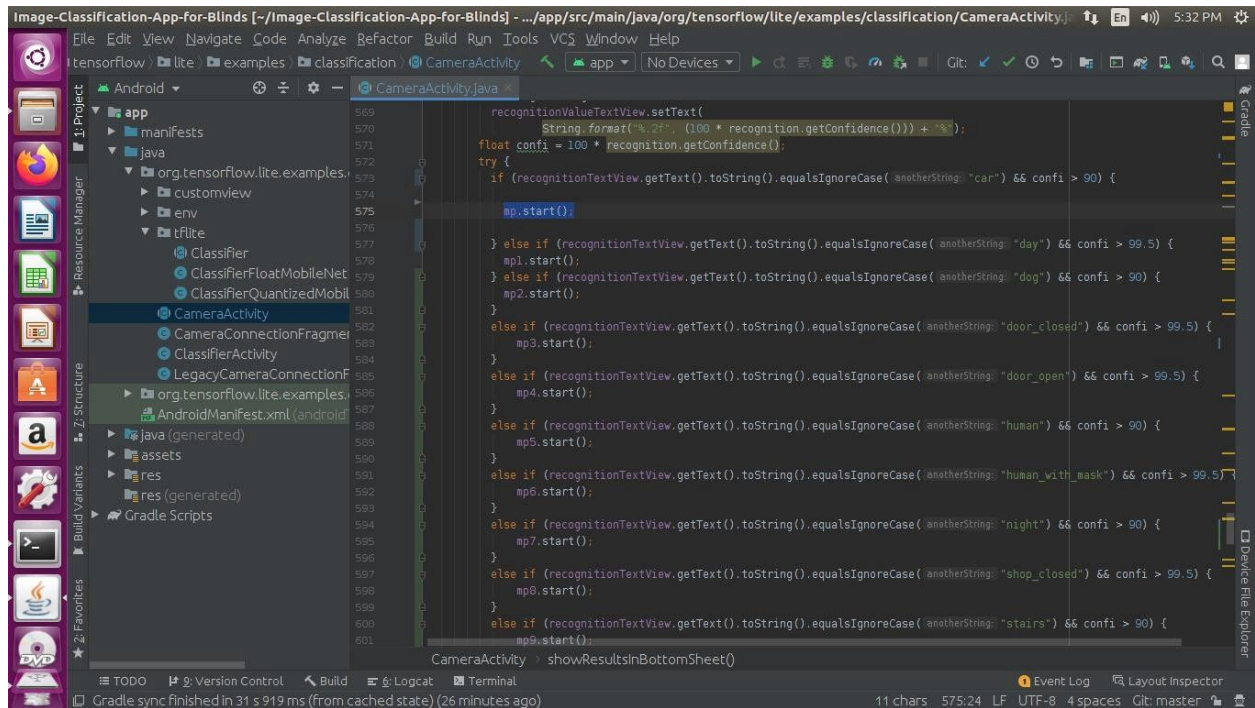


Repeat the above (step 13) for *all the classes* used for classification.



## Step 14

Also make changes in (eg. `mp.start()`) replace the `mp`, `mp1`, `mp2` & so on with the correct variable name to which the specified `class .mp3` file has been assigned to in `CameraActivity.java`.



## Step 15

Now click on *Build* & then *Make project* it shows *Gradle Build is Running*. Let the *Project Build Successfully*.

## Step 16

*Android*

i) Connect the android device make sure that the android device is in *Developer Mode* (<https://developer.android.com/studio/debug/dev-options>). *Stake Awake* & *USB Debugging* need to be turned on in *Developer Mode*.

ii) The Android Studio will recognize the Android Device. Now you can Run the App by clicking on *Run* then *Run 'app'* this will install the App (APK) in your Android Device as shown above.

**Note : This Chapter ends, The App Speaks : The voice has been added to the App now along with classification of the objects you can hear about the objects too.**

## Chapter 5

### 5.1 Changing the Layout of the App using Android Studio

#### 5.1.1 Certain changes have been made in the layout of the App which are stated below :

- The **res directory** contains the files related to the **LAYOUT**
- The location the directory is :

*Image-Recognition-and-Classification-Device-for-Blind-People-Using-Intel-NCS-2/Increased Accuracy App/app/src/main/*

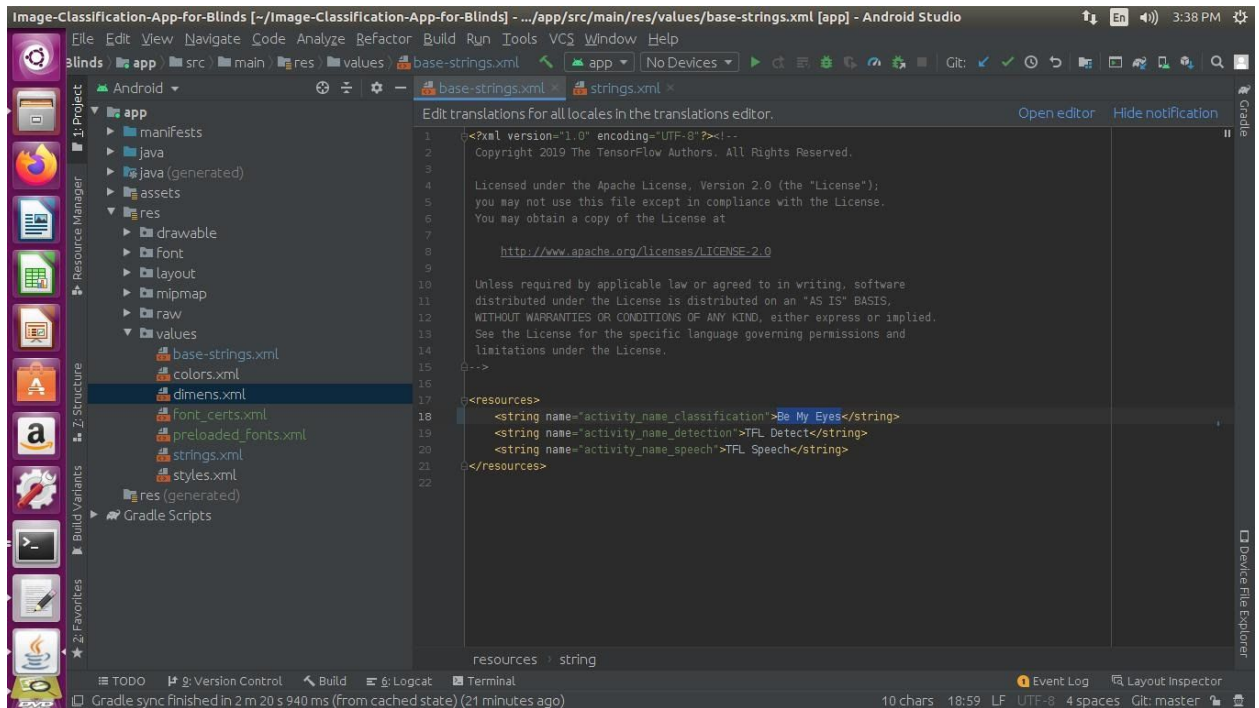
- Just place the res directory in main that is the location mentioned above to use the changed layout by replacing the res directory already present in the main.

### 5.2 Modifications

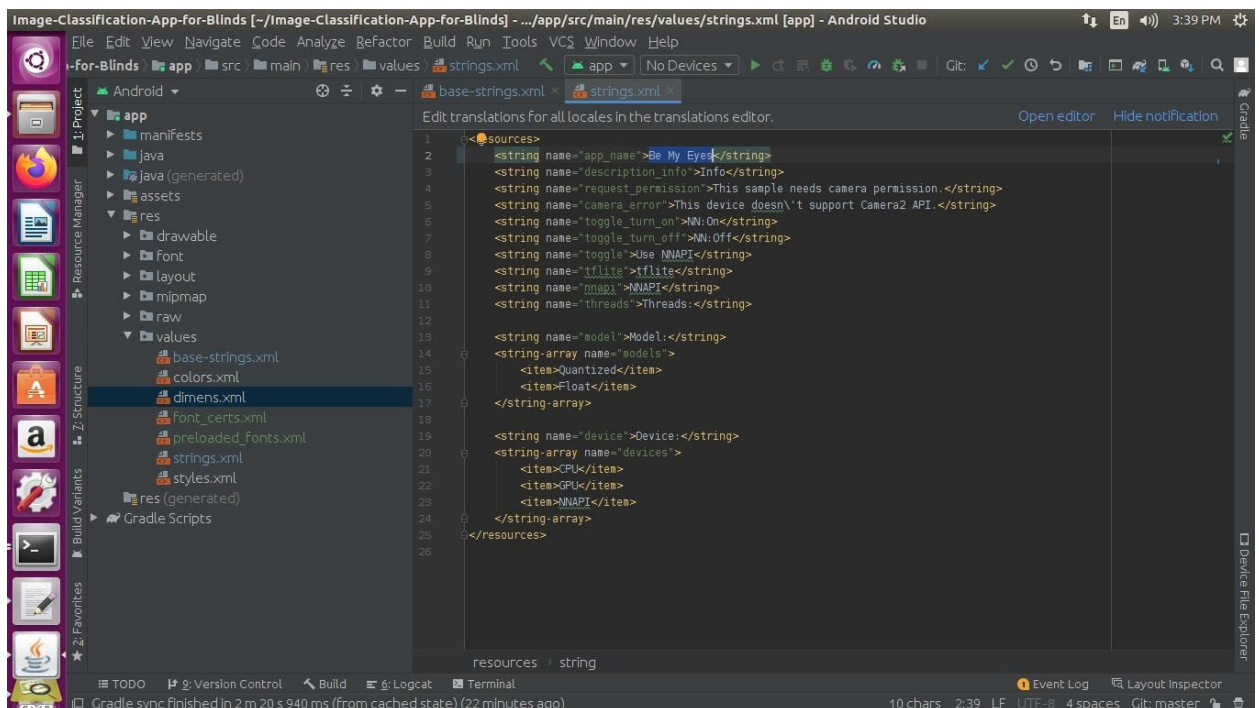
- Name of the App (Be My Eyes) in base\_strings.xml & strings.xml
- Name that is going to be displayed in the camera activity space (IMAGE CLASSIFICATION APP FOR BLIND)
- Colour of the name that is going to be displayed in the camera activity space (colour used #7BEFFD)
- Font of the name that is going to be displayed in the camera activity space (res/font/montserrat\_subrayada\_bold.xml)
- Extra tabs in the layout\_bottom\_sheet
- Logo of the App (Named as ic\_launcher.png, path specified in the screenshot)

### 5.3 Files that I have changed

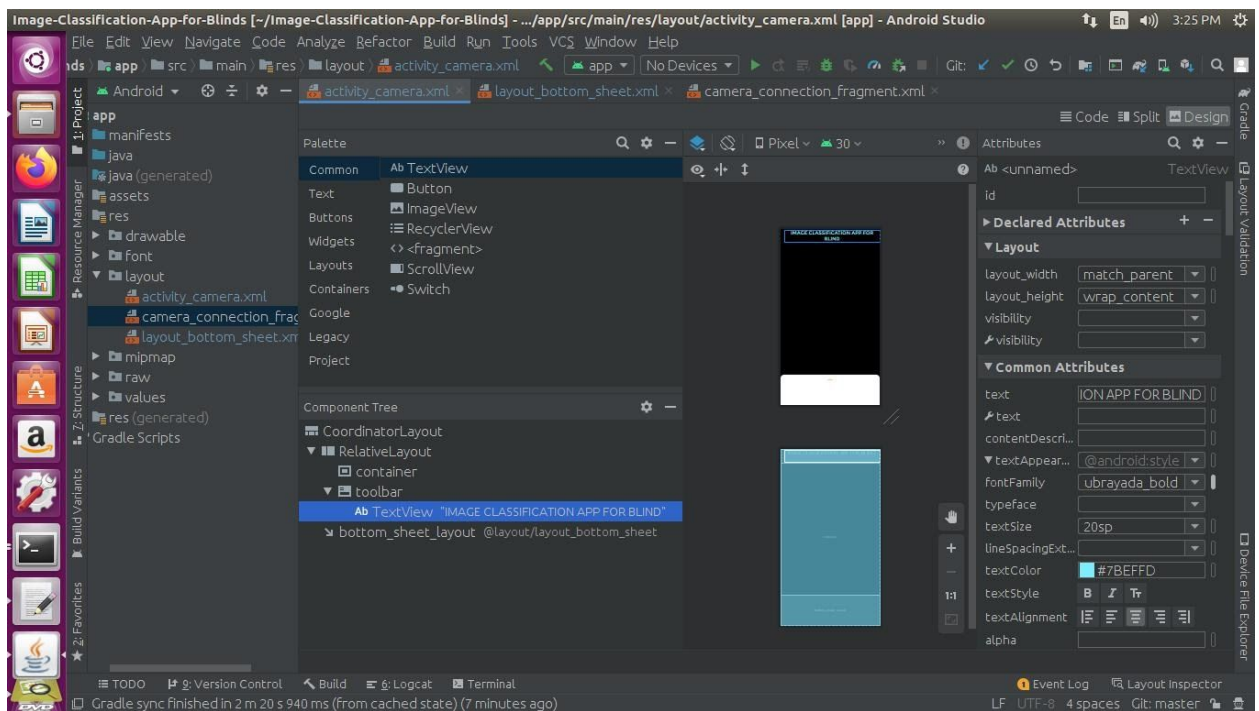
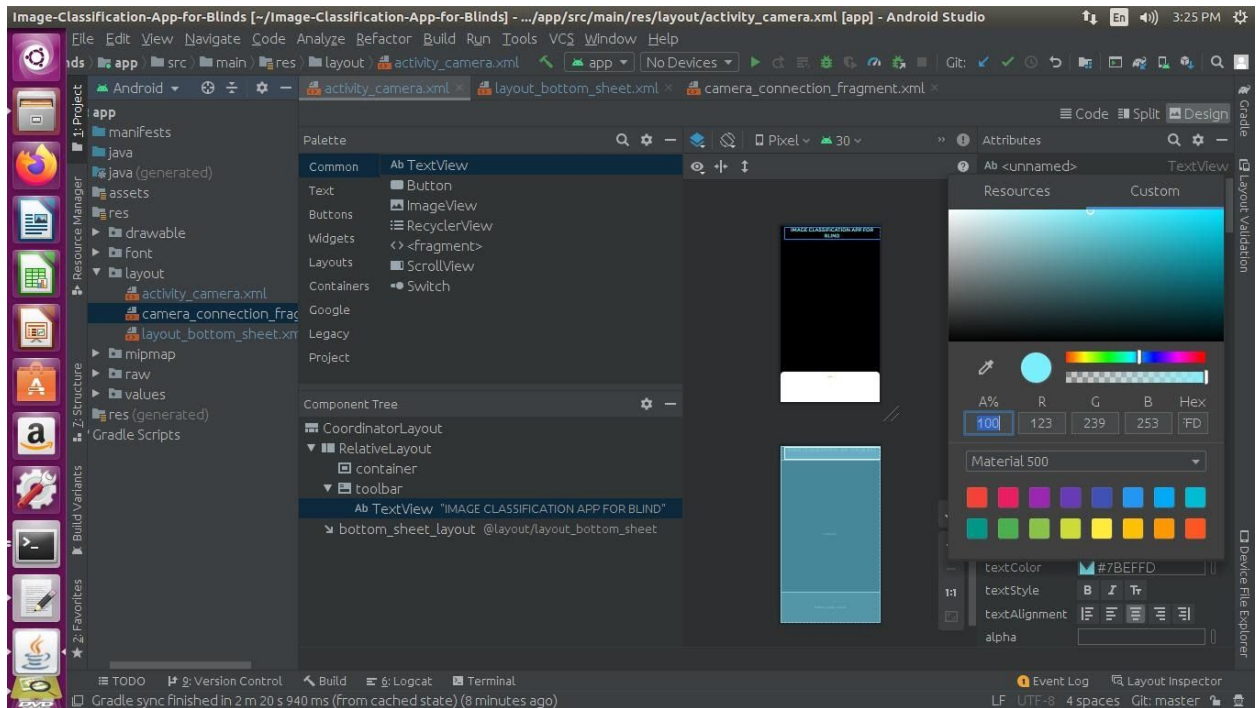
- base\_strings.xml



- strings.xml

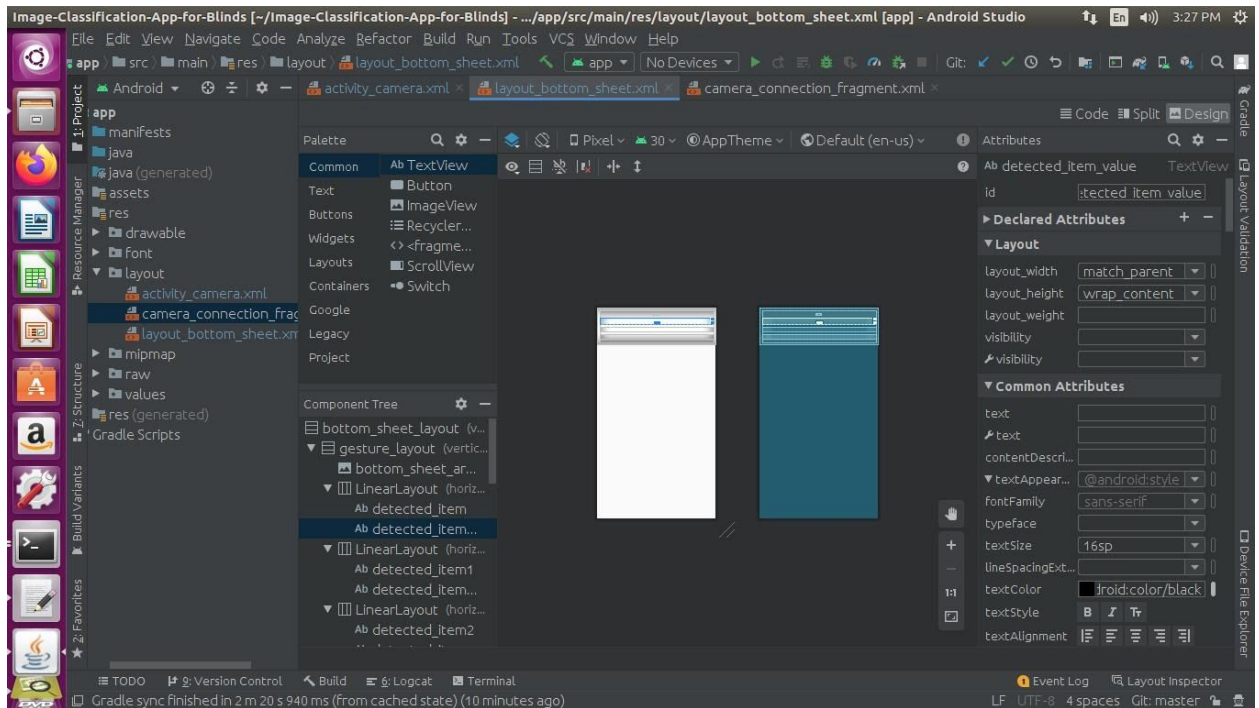


- activity\_camera.xml

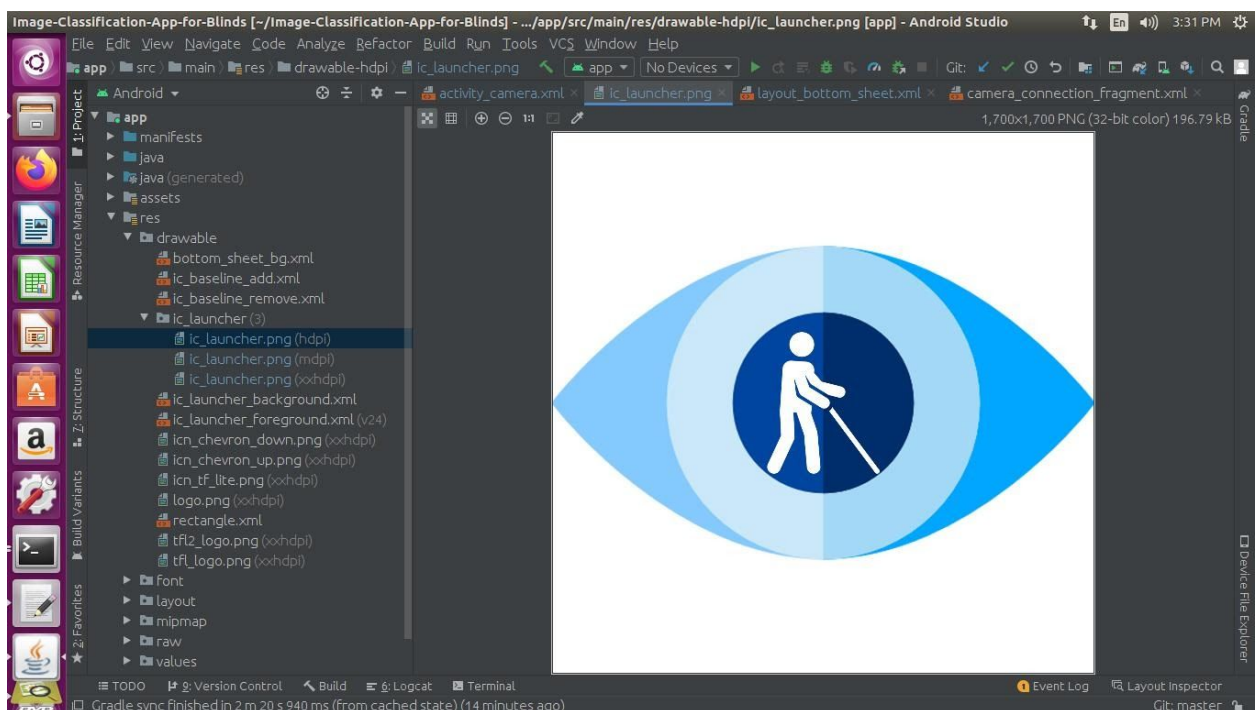


- layout\_bottom\_sheet.xml





- ic\_launcher.png (path specified in the screenshot logo\_of\_the\_app.jpg)



**The raw directory in res (res/raw) contains the .mp3 files of the classes which have been classified in the App, If you are using some other classes you can change the .mp3 files according to the classes you want**

**to classify for the  
IMAGE\_CLASSIFICATION\_APP\_FOR\_VISUALLY\_IMPAIRED.**

- Now click on *Build* & then *Make project* it shows *Gradle Build is Running*. Let the *Project Build Successfully*.
- *Android*
  - Connect the android device make sure that the android device is in *Developer Mode* (<https://developer.android.com/studio/debug/dev-options>). *Stake Awake & USB Debugging* need to be turned on in *Developer Mode*.
  - The Android Studio will recognize the Android Device. Now you can Run the App by clicking on *Run* then *Run 'app'* this will install the App (APK) in your Android Device.

***Note : This Chapter ends, The App has been built and can be used as an Assistive Technology for Visually Impaired People, so that they can classify objects / images near them & hear a voice (audio) feedback about the description of the objects it is classifying.***

## **Chapter 6**

### **6.1 Future Work**

- Play with more machine learning methods to get better accuracy.
- And classifying a wide variety of classes that can help the visually impaired.
- Also more work can be done on the layout of the App
- More models related to Pose detection, Sound Detection & Colour Detection can be added to the App.