Steps:

1. Gather Images through Webcam (Dataset):

We have written a python script which captures the images real-time through webcam and stores it in a directory.

1. Label the captured images (mask/no mask) using LabelImg tool.

We have done its installation and usage instruction from its [GitHub page](https://github.com/tzutalin/labelImg). In this step, faces in the images will be annotated with a label and its specific XML files will also be created in that labelled folder.

1. Split training and testing data

 We divide the dataset into two parts- **train**(80% of images with their corresponding XML files) and **test**(remaining 20% of images with their corresponding XML files).

1. Create label\_map.pbtxt file

In label\_map file, it maps each label class (mask/no mask) to an int value.

1. Create generate\_tfrecords.py script

This script (generate\_tfrecords.py) coverts the annotations into the TFRecord format. It iterates through all .xml files and converts it to .record files for both training and testing dataset.

1. Setting up and installing the tensorflow object detection API

* We are using tensorflow v1 for this project 2.2.0
* Cloning the Tensorflow Model garden Repository from [this](https://github.com/tensorflow/models) Github Page.
* The tensorflow object Detection API uses Protobuf to configure model and training parameters. Therefore we downloaded and compiled Protobuf libraries.

1. Downloading a pre-trained model to apply transfer learning from this [link](https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md).

Here, we are using the SSD Resnet50 V1 FPN 640X640 model provided by tensorflow for training. It serves as starting checkpoint for our training jobs.

## Set up the directory structure and environment on Google Colab

## Here we are setting up the required directory structure in Tensorflow folder, which we can use to store the useful scripts and some paths in Google Colab for smooth execution.

## Configuring the pipeline file

## We made few configurations according to our selected pre-trained model and training dataset in pipeline.config file.

## Starting tensorboard

## As , tensorBoard allows us to track and visualize various training metrics while training is ongoing. Therefore we load tensorboard extension in Google Colab.

## Training the model

## Running the model\_main.py file and setting training steps to 25000.

## Exporting the trained model

## Here, we extract our saved trained model for object detection.

## Testing the model

## We are finally testing the model by loading the saved\_model and loading label\_map and testing image.

## Running the inference

## Finally, we see the loaded given images with bounding boxes, labels, and accuracy.