Step 1 : Get access to web cam & video feed.

Use browsers, media device Interface, → which lets you get access to connect devices like camera, Microphone & Even Screen sharing.

<video> tag in index.html is initially used to view the web cam feed.

& then it is passed to the face detection model to do the actual detection.

In script.JS

→ getUserMedia() is a API, takes an object as argument & inside that, we can specify the parameter that we want.

Will return stream object. → align this to video object.

Step 2 : Go to tensorflow.org.JS

↓

models → simple face detection

add the script in both index.html & Script.JS.

Step 3

NOTES

* function to detect the actual face. | make prediction

We cannot call detect faces function Even before the model is loaded.

so we need to wait for blazeface model to load.

- as soon as video feed is loaded, we wait Event listner called loaded data will help us.

so Once Video Element gets loaded, we can Execute our Model.

O/P: Console:

we will see that, we have bottomRight, Landmarks etc.

* Our prediction is an array, & it will return different array If there were multiple faces in our camera.

Step 4 ~~Make~~ Display Result

We will first Draw actual video feed on Canva

And then on top of that we will draw a Rectangle enclosing the face & different face point.

<canvas> → in index.html

Our canvas width & height should match with the width & height of our video in script.js

we only need 2D context → getContext

- Our detectFaces() function as previously is called once [& is fine when working with photo] but when working on video, it needs to be called some 20 - 30 times/sec
- ctx.rect() takes top left coordinate & also takes width & height.

so subtract bottom right to top left. to get width & height.