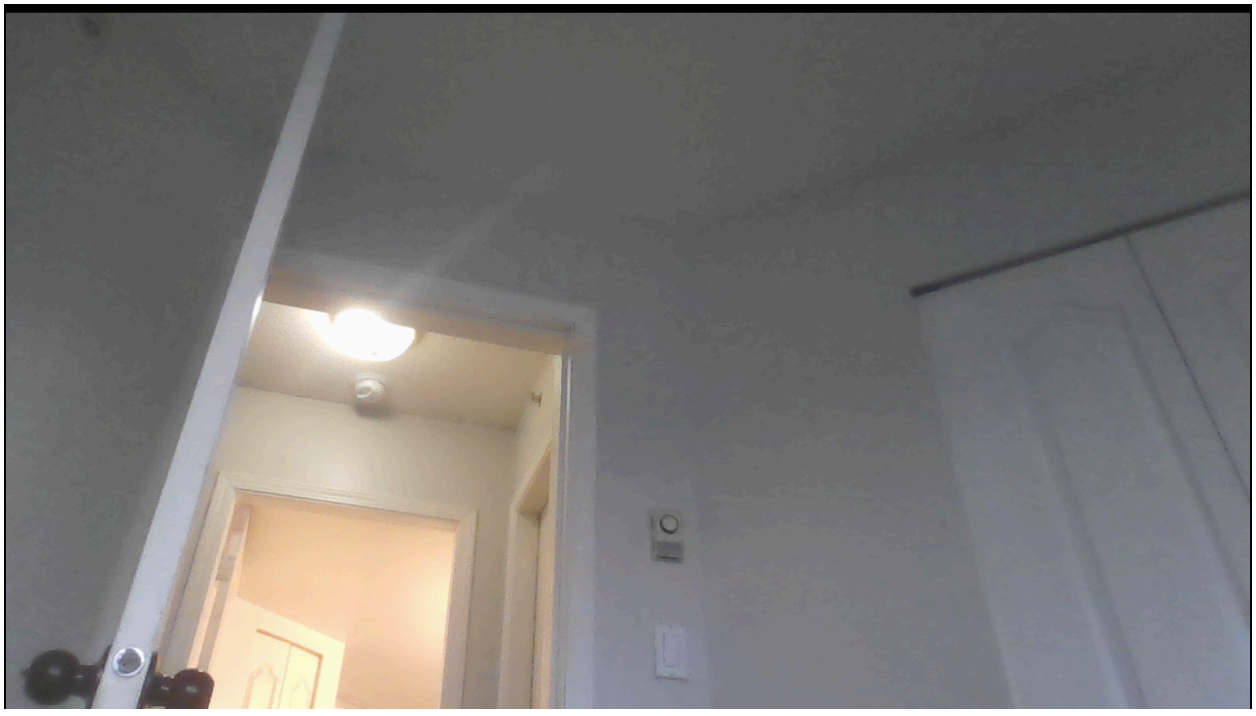
5330 Project 1, Jinda Zhang, Abdulaziz Arif Suria

A short description of the overall project in your own words. (200 words or less)

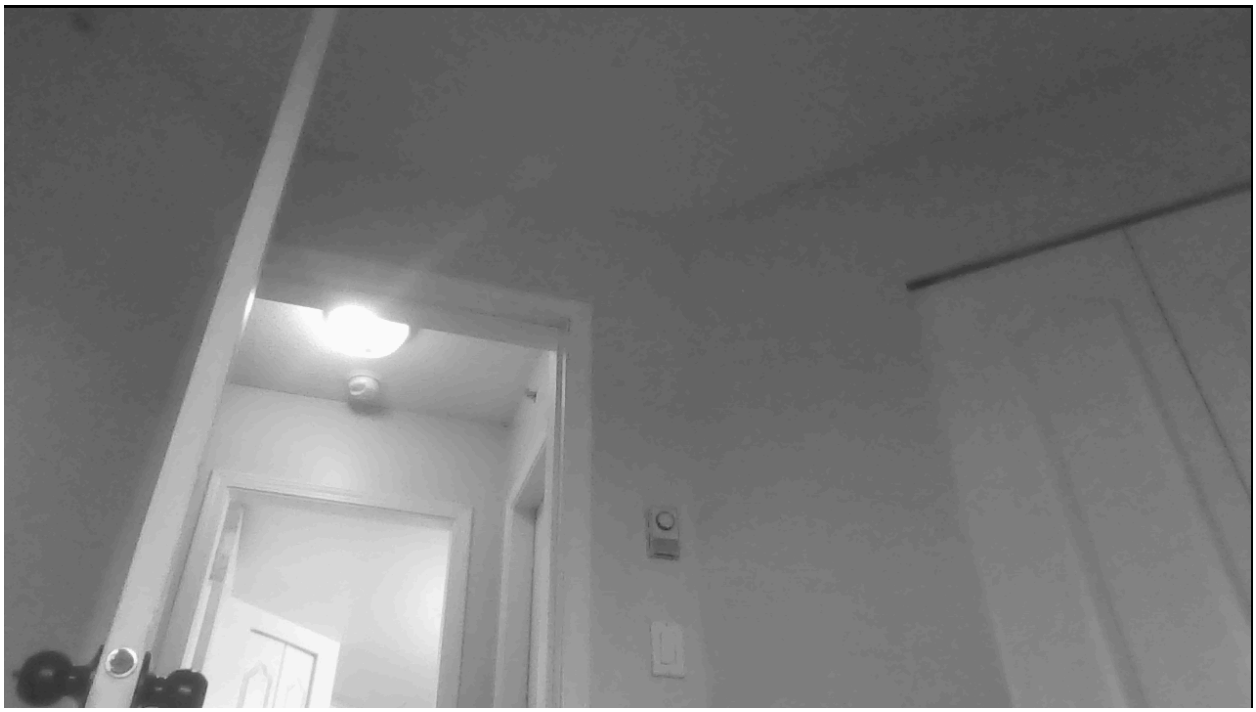
The overall project involves developing a C++ program using the OpenCV library to manipulate and display images and live video streams. The tasks include basic image loading and display, live video display with keyboard input handling, implementation of various image filters (greyscale, sepia, blur), face detection, and creation of custom effects. The project implements fundamental tasks like displaying images to more advanced functionalities such as implementing custom filters, face detection, and creating special effects.

Required Image 1: show the original and cvtColor version of the greyscale image in your report. **(Press ‘g’ key)**

original:



After conversion:



OpenCV docs: https://docs.opencv.org/3.4/de/d25/imgproc_color_conversions.html
Look up in the OpenCV documentation how each color channel is weighted in the conversion and explain it in your report:

RGB $\leftrightarrow$ GRAY

Transformations within RGB space like adding/removing the alpha channel, reversing the channel order, conversion to/from 16-bit RGB color (R5:G6:B5 or R5:G5:B5), as well as conversion to/from grayscale using:

$$\text{RGB[A] to Gray:} \quad Y \leftarrow 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B$$

and

$$\text{Gray to RGB[A]:} \quad R \leftarrow Y, G \leftarrow Y, B \leftarrow Y, A \leftarrow \max(ChannelRange)$$

The conversion from a RGB image to gray is done with:

```
cvtColor(src, bwsrc, cv::COLOR_RGB2GRAY);
```

More advanced channel reordering can also be done with **cv::mixChannels**.

Required image 2: show your customized greyscale image in your report, explain how you decided to generate your greyscale version, and highlight the differences with the default greyscale image. **(Press ‘h’ key)**

explain how you decided to generate your greyscale version:
subtract the red channel from 255 and copy the value to all three color channels, which is different from the Default is RGB[A] to Gray:Y←0.299 · R+0.587 · G+0.114 · B

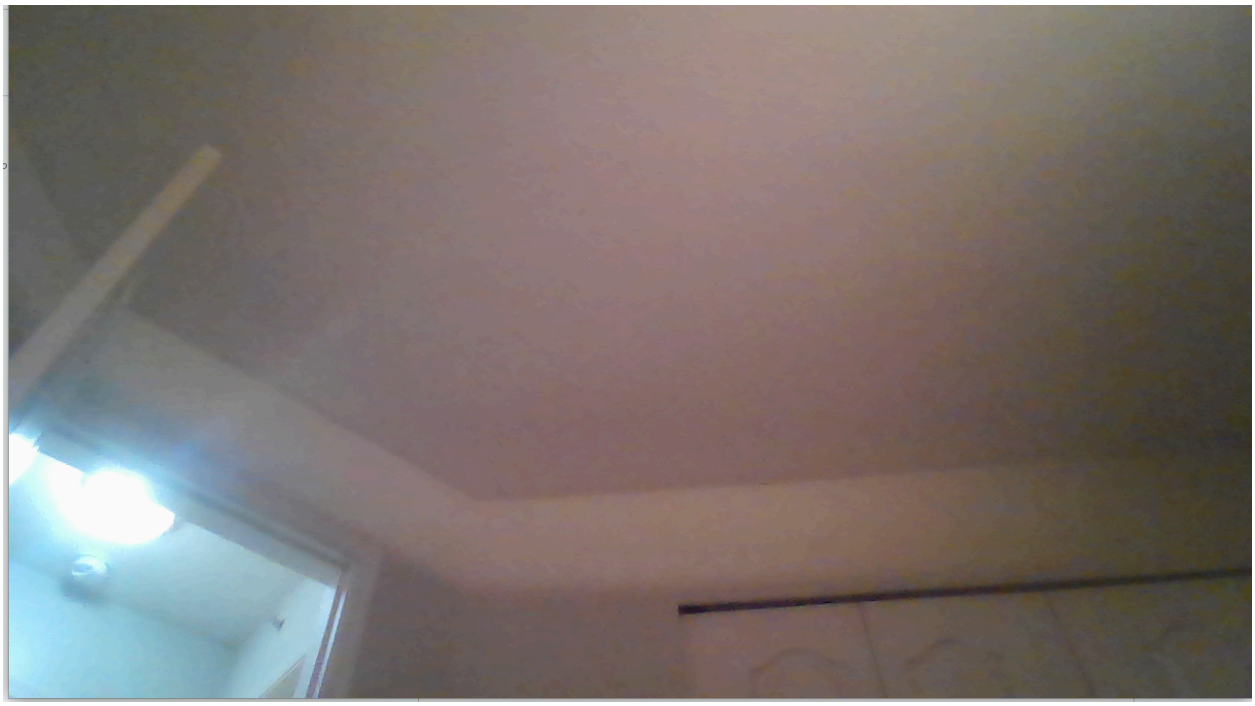
Required image 3: show your sepia tone filter in your report and explain how you ensured using the original RGB values in the computation. **(Press ‘j’ key)**

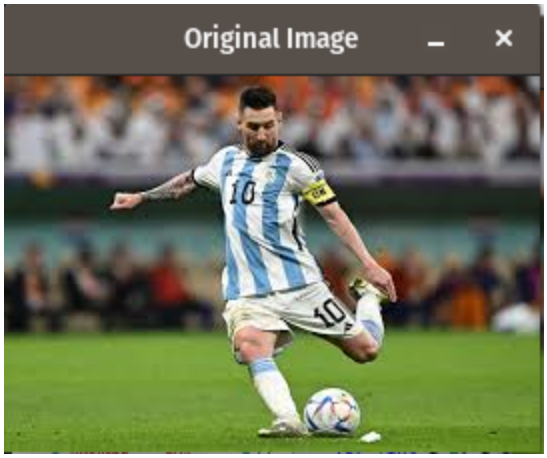
explain how you ensured using the original RGB values in the computation:
I use separate variables to store the original RGBs in the image. We use the following coefficients.

```
0.272, 0.534, 0.131    // Red coefficients
0.349, 0.686, 0.168    // Green coefficients
0.393, 0.769, 0.189     // Blue coefficients
```

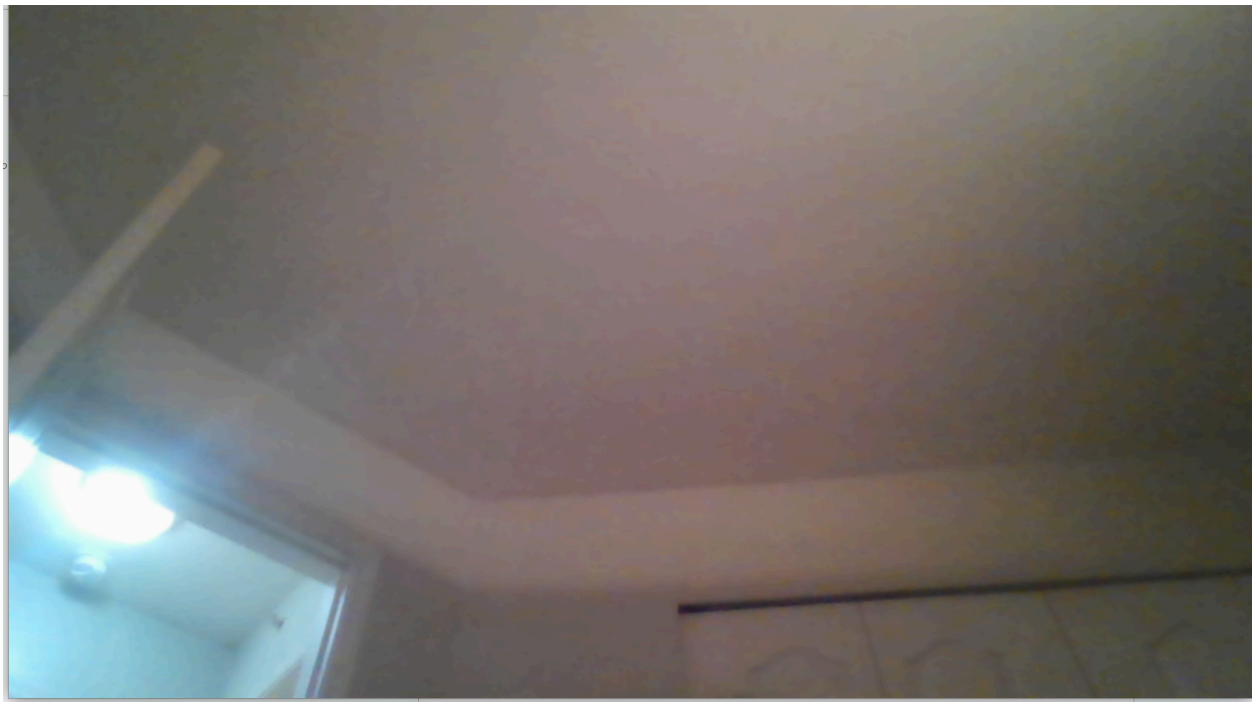
Required Image 4: show the original and the blurred image in your report along with the timing information. **(Press ‘b’ key)**

original:





blur:

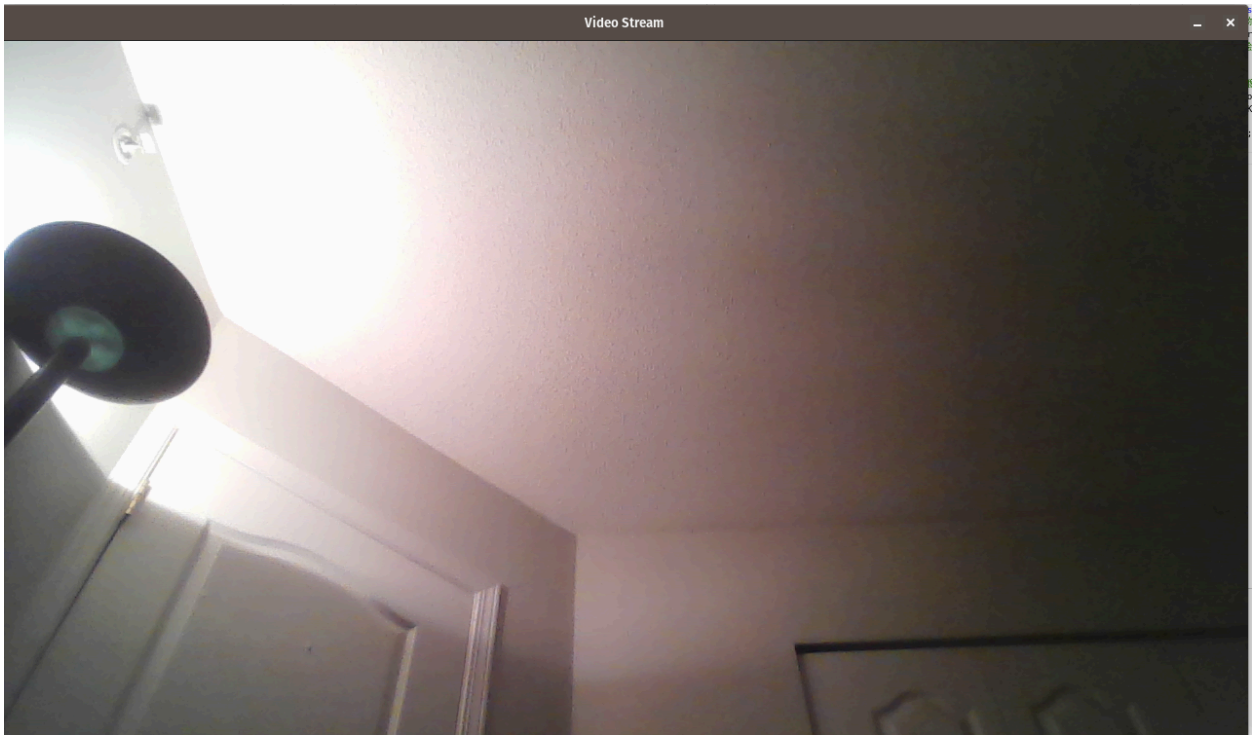


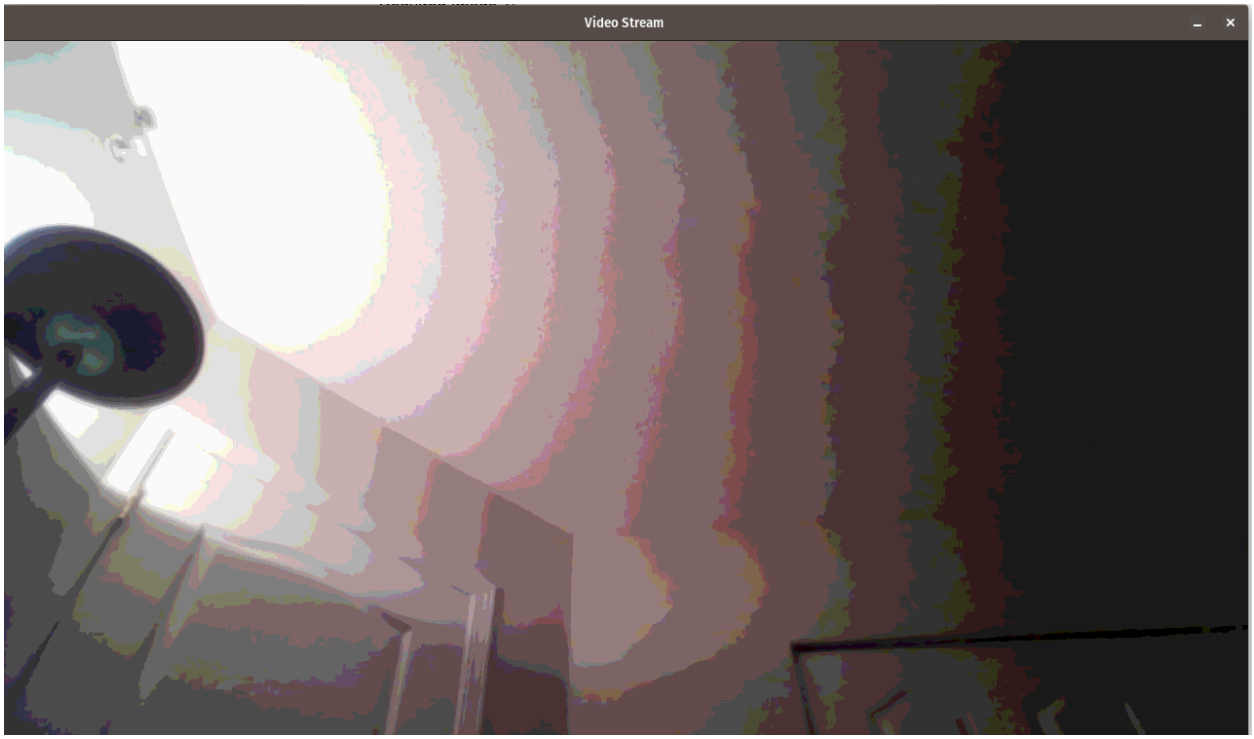
Include your timing information in your report and explain what you changed to make the filter faster.

Time per image (1): 0.0280 seconds
Time per image (2): 0.0086 seconds
My change is instead of using a 5*5 filter, use two 1*5 filter separable filters to get the effects reflected.

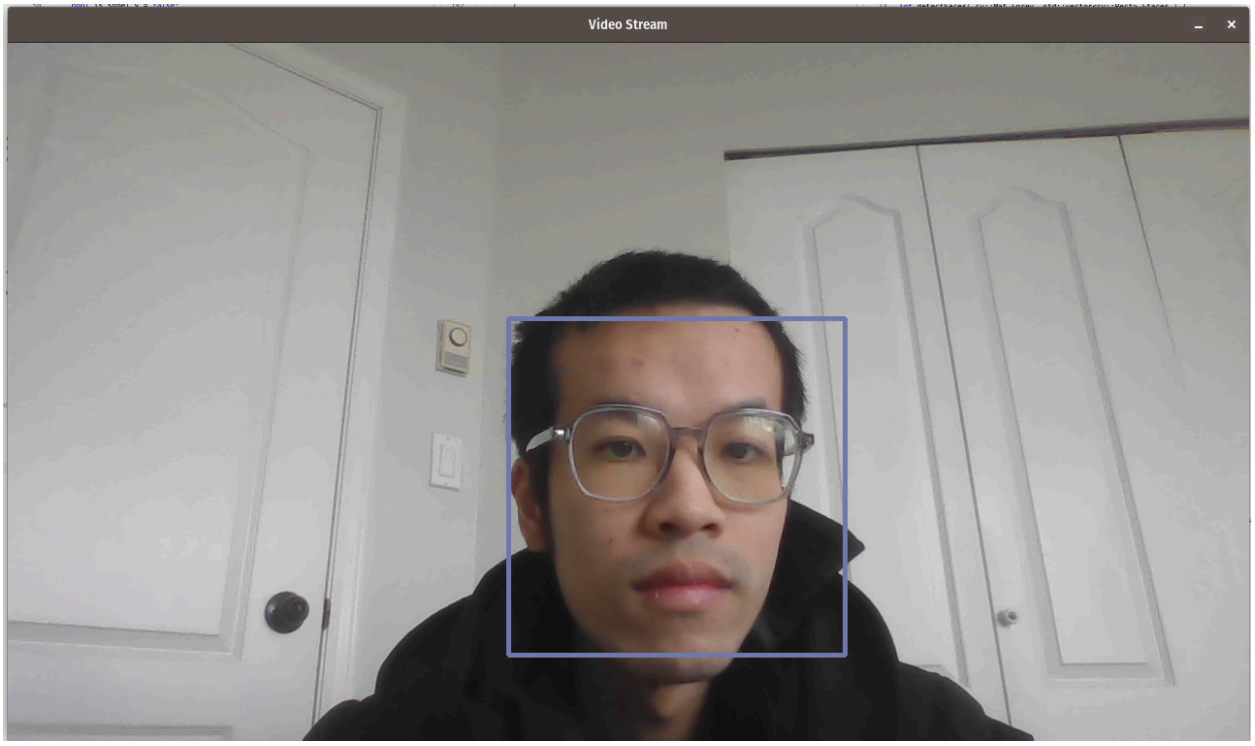
Required Image 5: show the original and the gradient magnitude image in your report. **(Press ‘m’ key)**

Required Image 6: show the original and the blur quantized image in your report. **(Press ‘i’ key)**





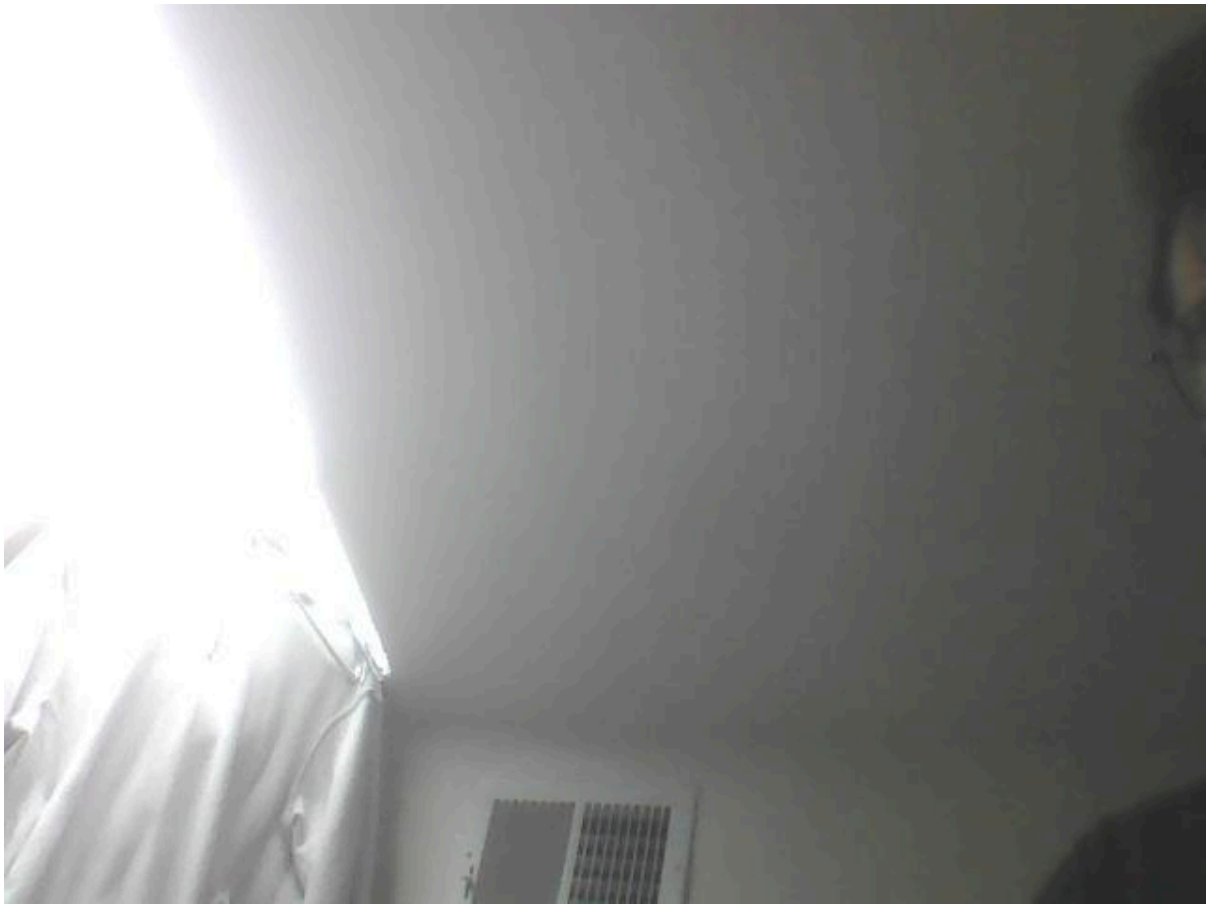
Required Image 7: show a face being detected in your video stream. **(Press ‘f’ key)**



Required Images 8-10:(Task11)

We have applied the following extensions:

Image 8 :Negative Filter Effect: We have adopted a negative filter effect where each pixel value is replaced by 255-pixel value. **(Press ‘n’ key) (original vs negative displayed below)**



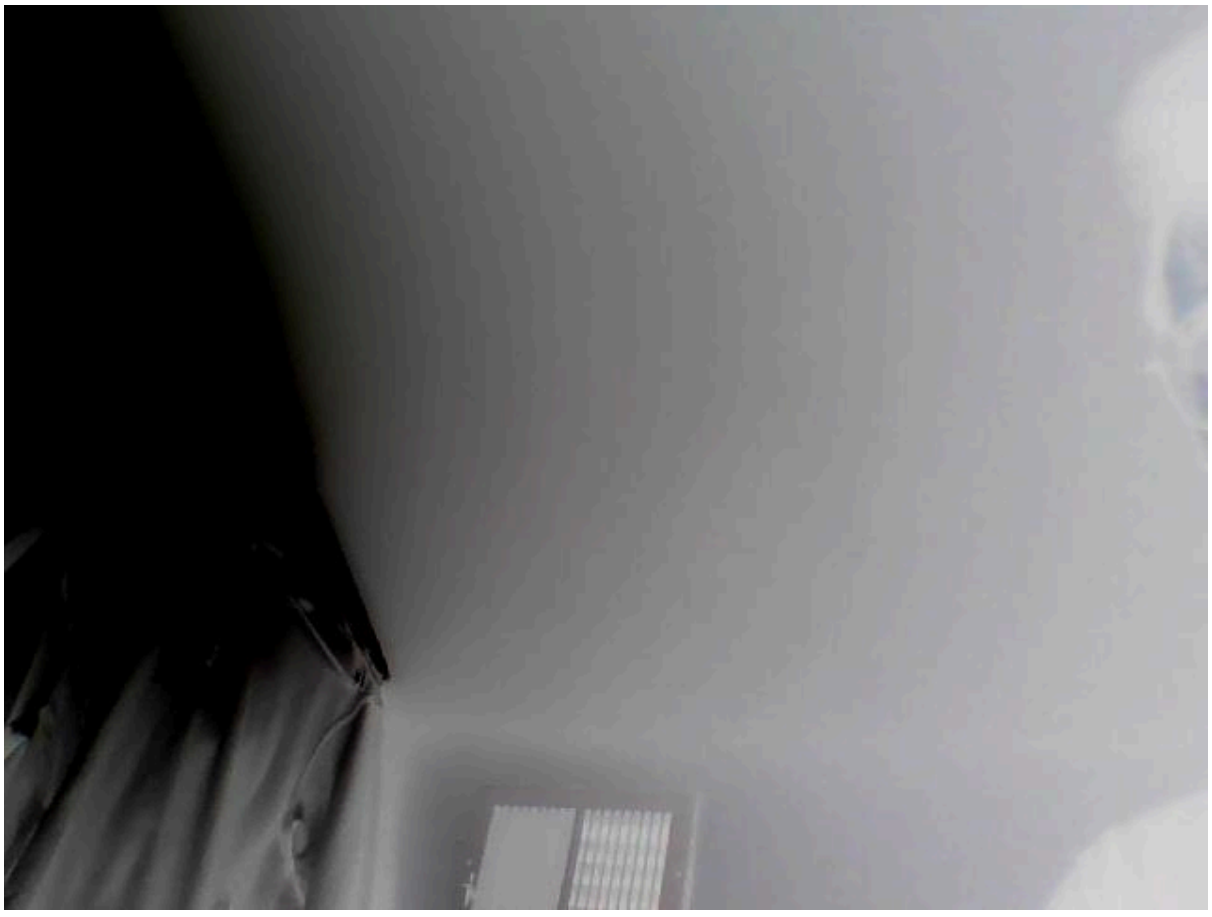


Image 9 :Face color Effect: We have created a filter which makes faces colorful and everything else greyscale using masking techniques.**(Press ‘c’ key)**

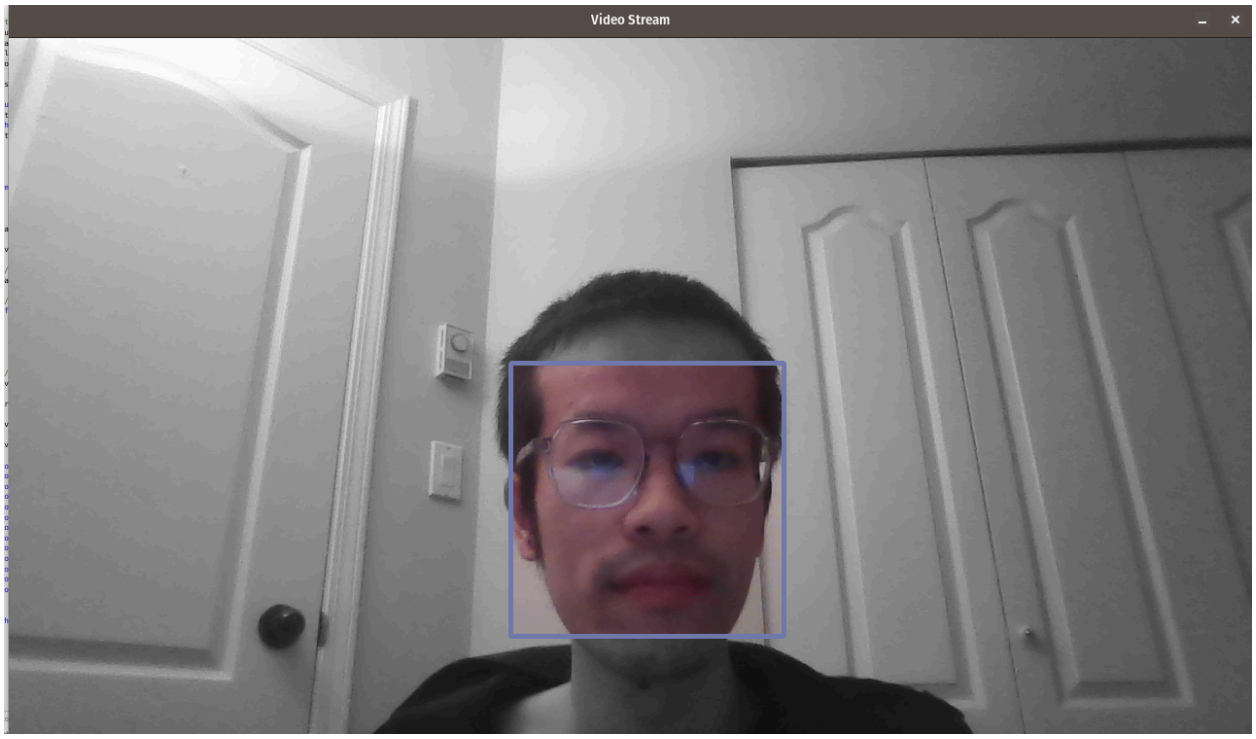
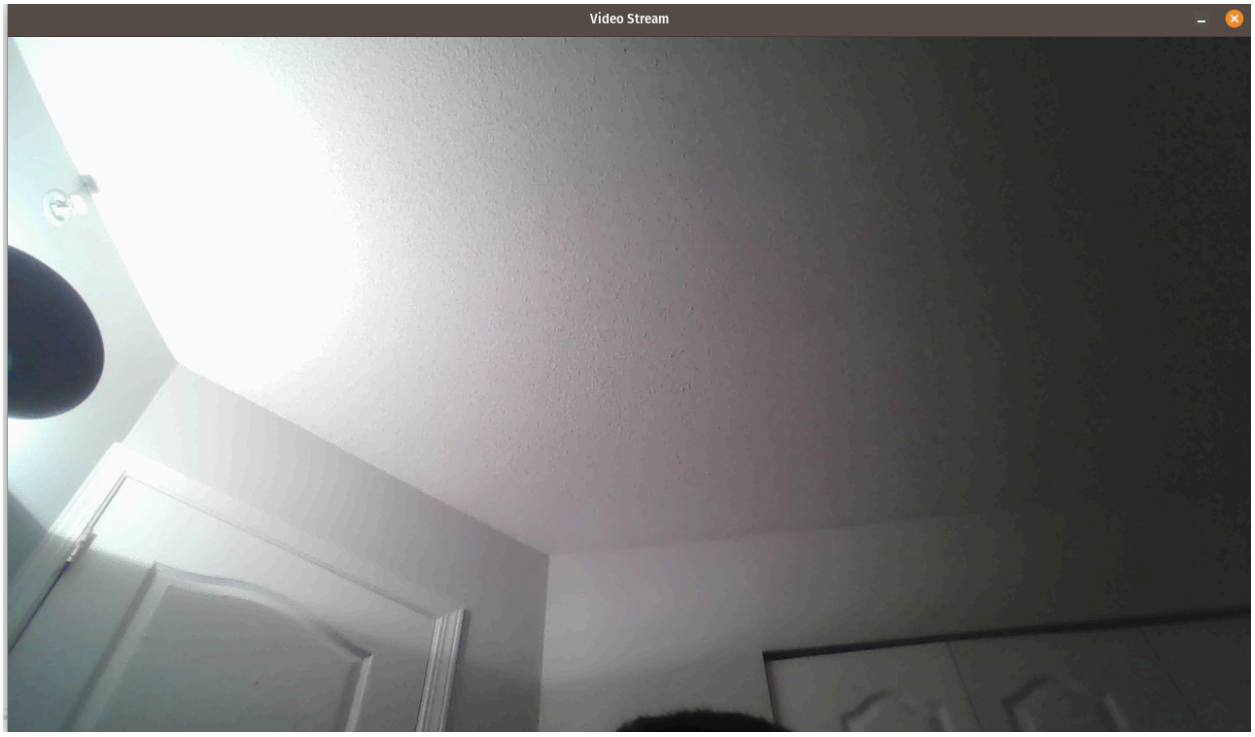
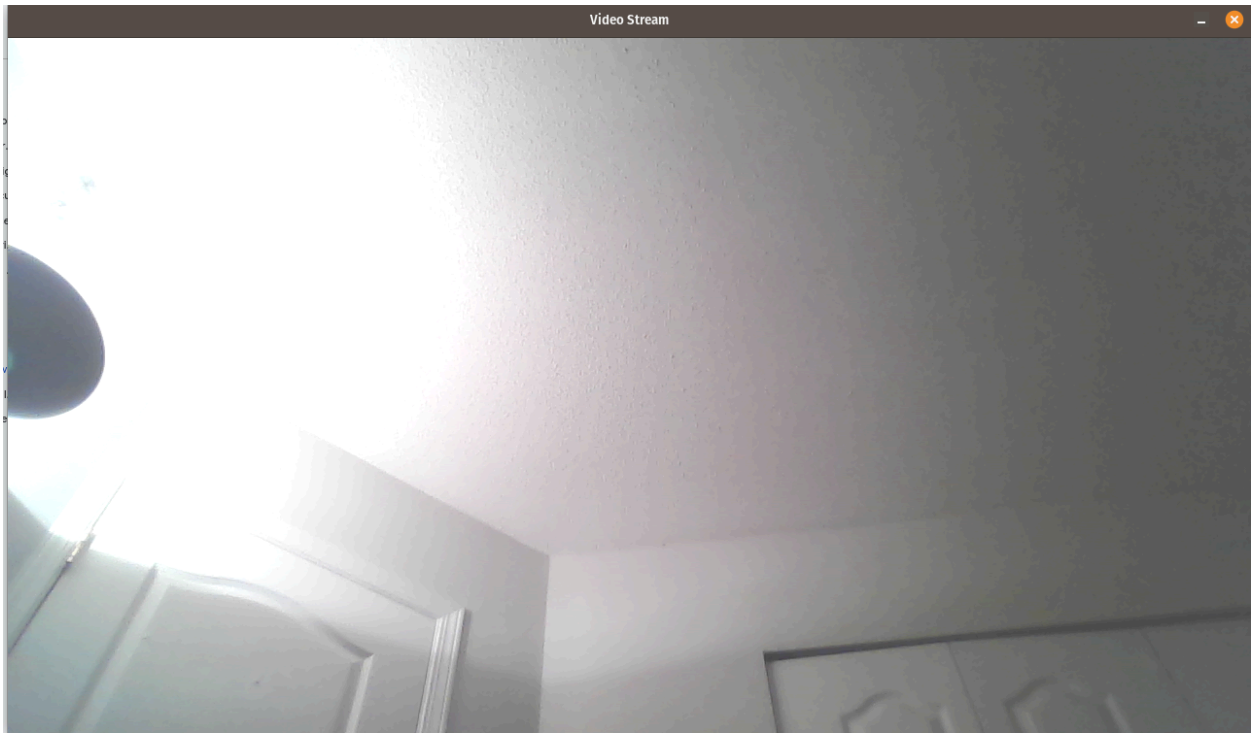


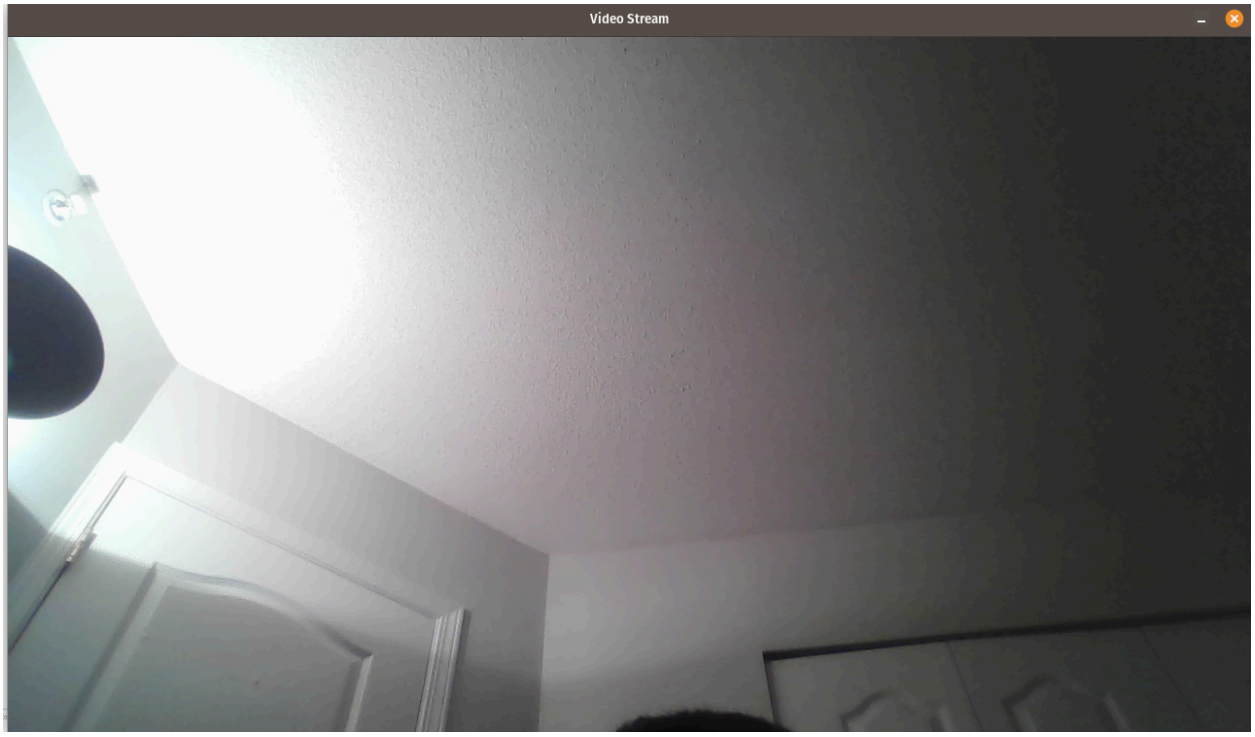
Image 10 :Brighten/darken image: We have created a functionality which brightens the image by 50 pixels and darkens by 50 pixels by modifying each pixel value with a value of 50. **(Press ‘w’ key : brighten, press ‘e’ key : darken)**

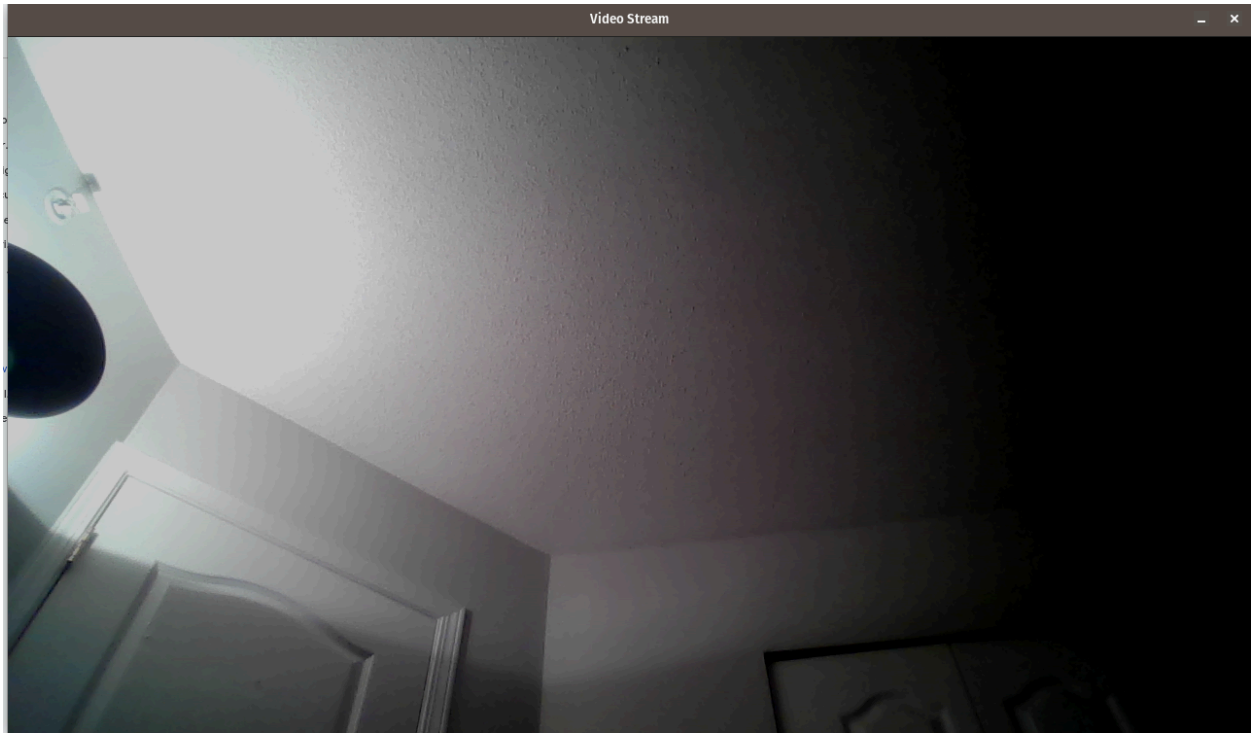
(Brighten : original vs brightened below)





(Darken : original vs darkened below)

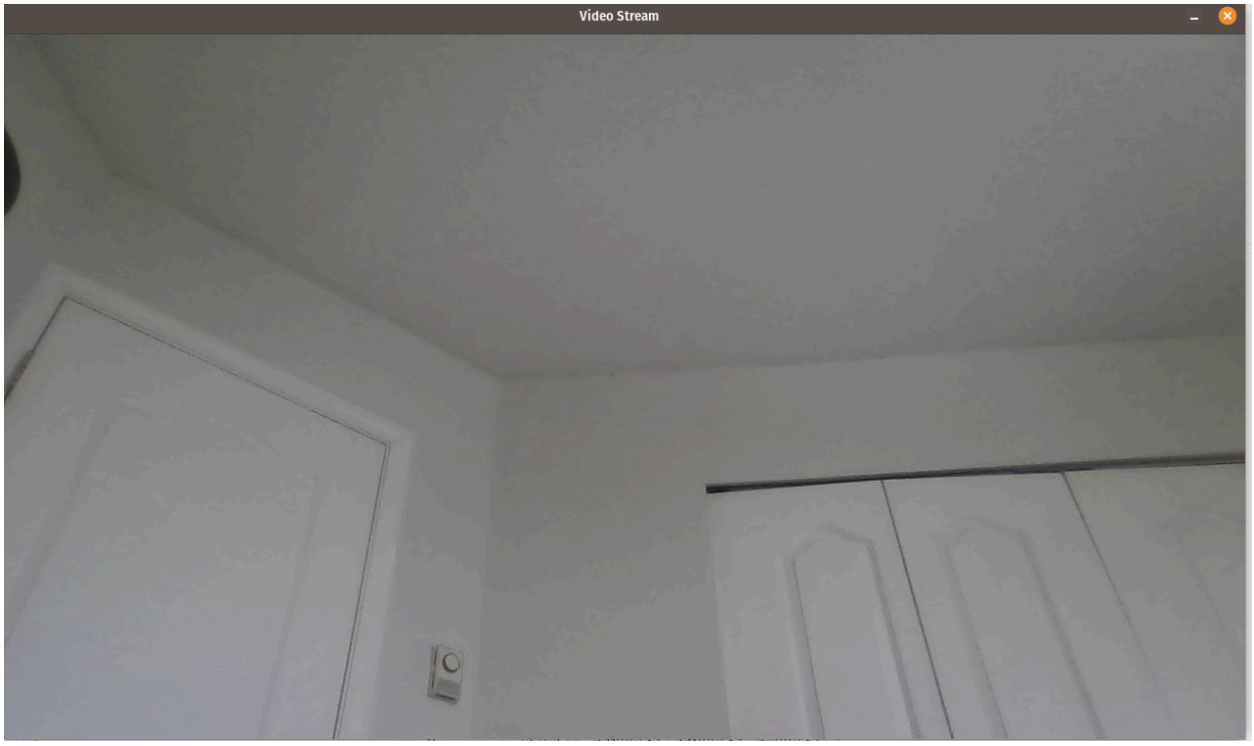

Video Stream

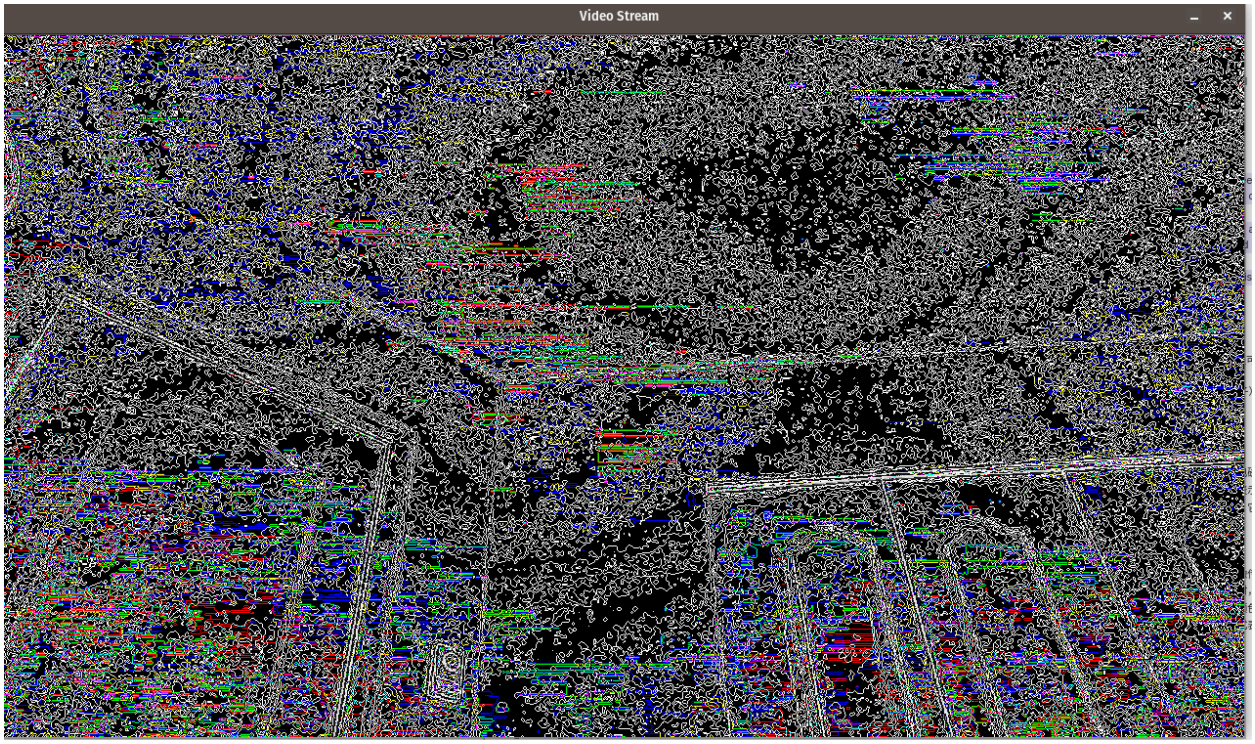

Video Stream

Extensions: We have adopted these additional filter effects as part of our extension.

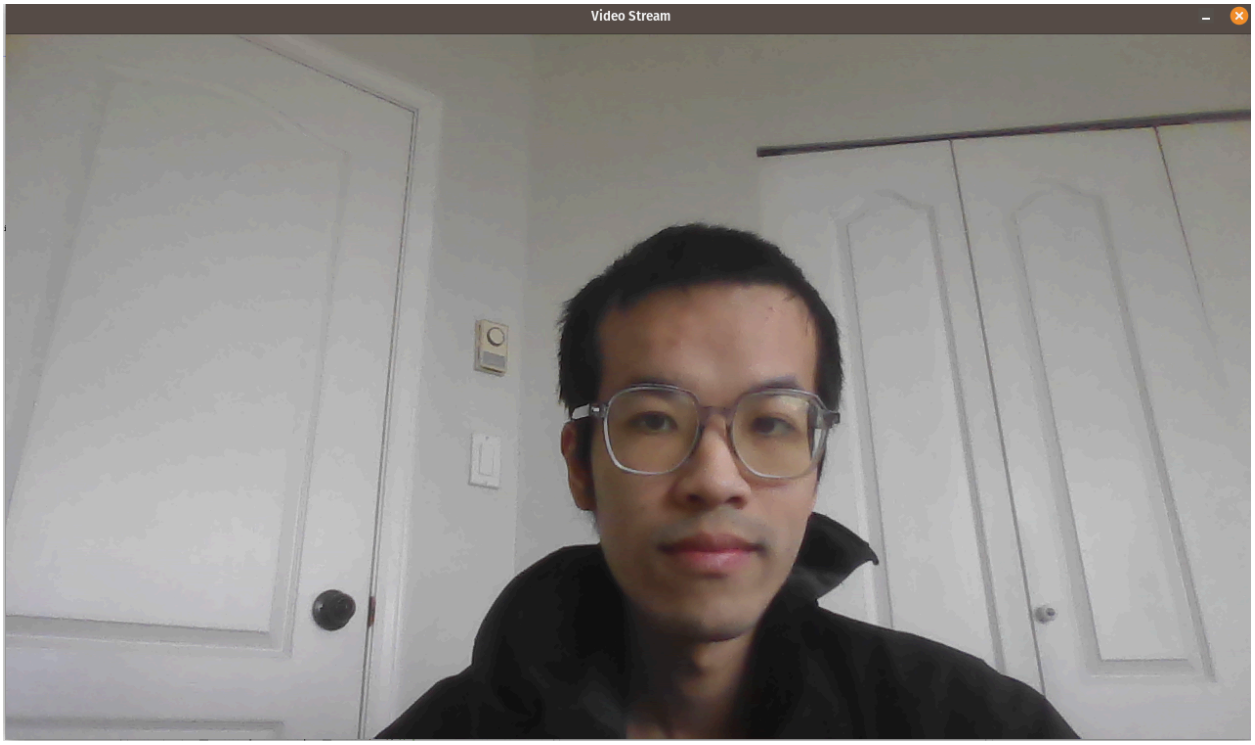
Extension 1 Laplacian Filter: We have applied a laplacian filter effect with the following filter. (original vs laplacian) **(Press ‘l’ key)**

0	−1	0
−1	4	−1
0	−1	0

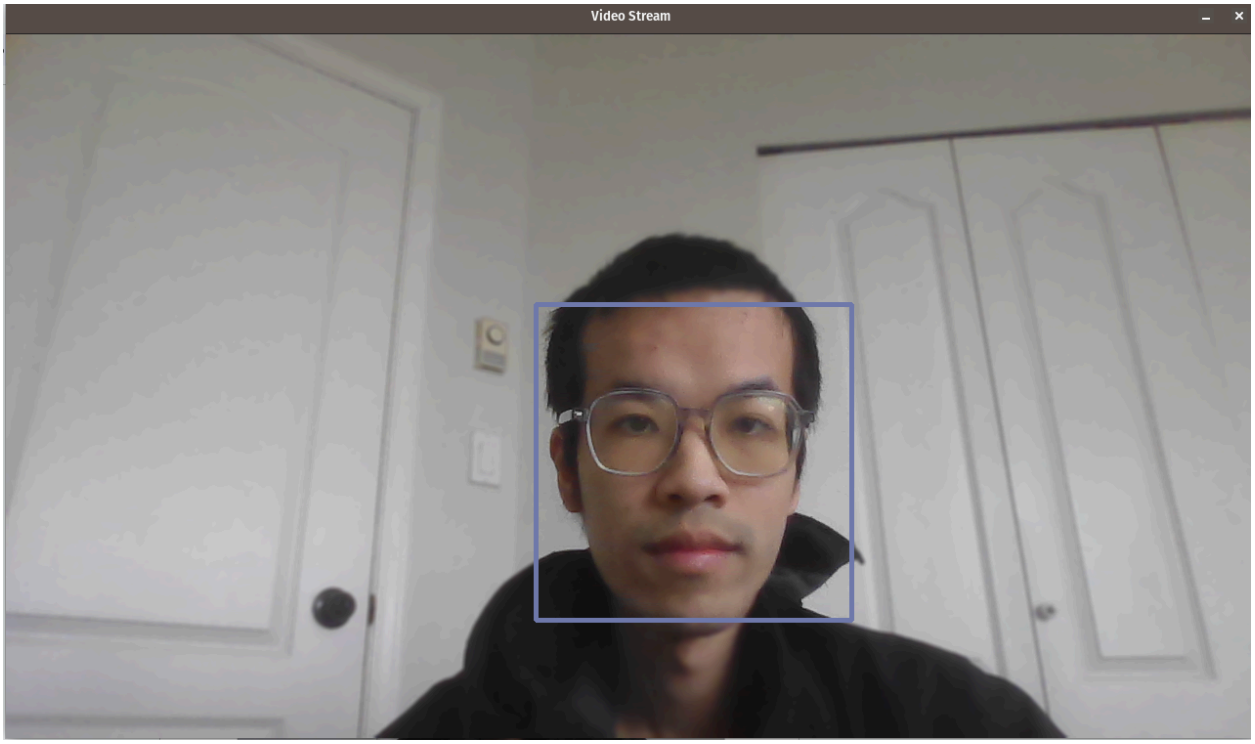



Video Stream

Extension 2 Precise face detection: We have adapted face detection with blur in background using bitwise operations. **(Press ‘k’ key)**



Face detection blur in background



Extension 3: Implement your effects for still images and enable the user to save the modified images.

We have created a new file imgModifier.cpp which allows user to select a file and a command to create a modified image and save it to a particular location.

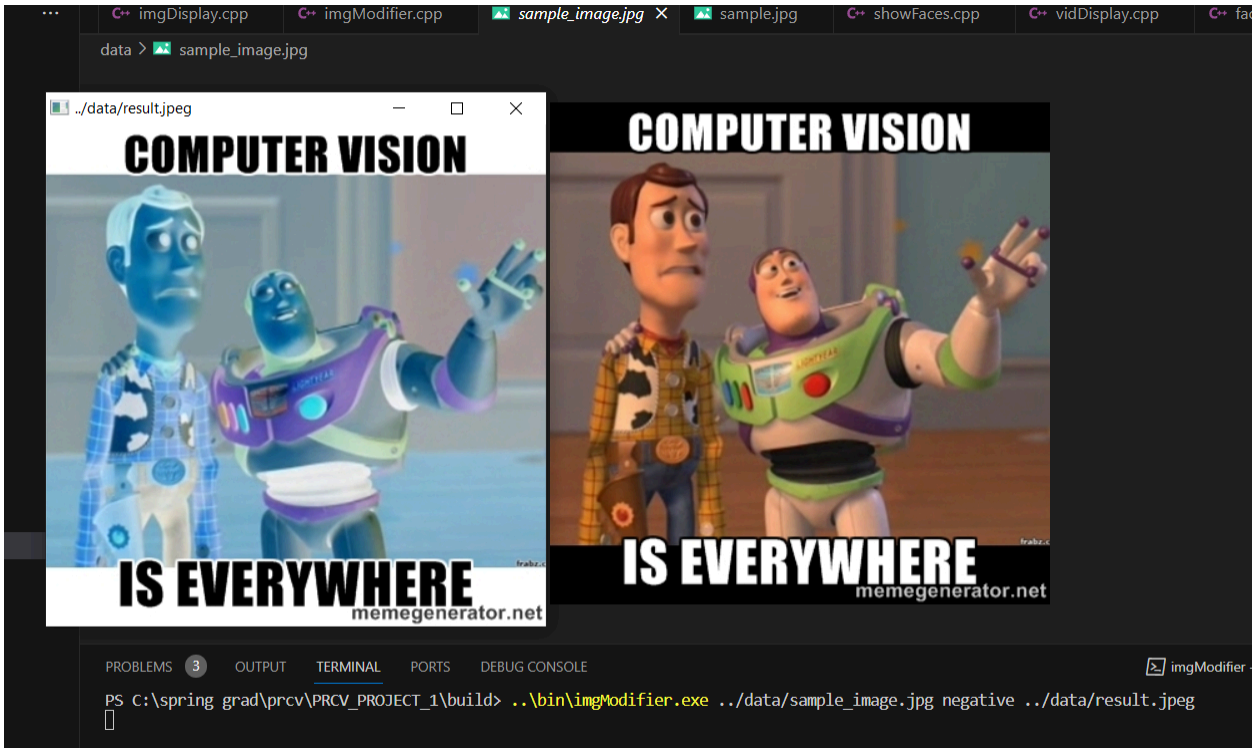
The usage of the file is as follows:
imageModifier.exe <image filename> <command> <image savefilename>

We have adopted the following commands and effects:
NEGATIVE = "negative"
CUSTOM_GREY = "customgrey"
SEPIA = "sepia"
BLUR = "blur"
GREY = "grey"
SOBEL_X = "sobelx"
SOBEL_Y = "sobely"
MAGNITUDE = "magnitude"
BLUR_QUANTIZE = "blur_quantize"
FACE_DETECT = "face_detect"
FACE_BLUR = "face_blur"

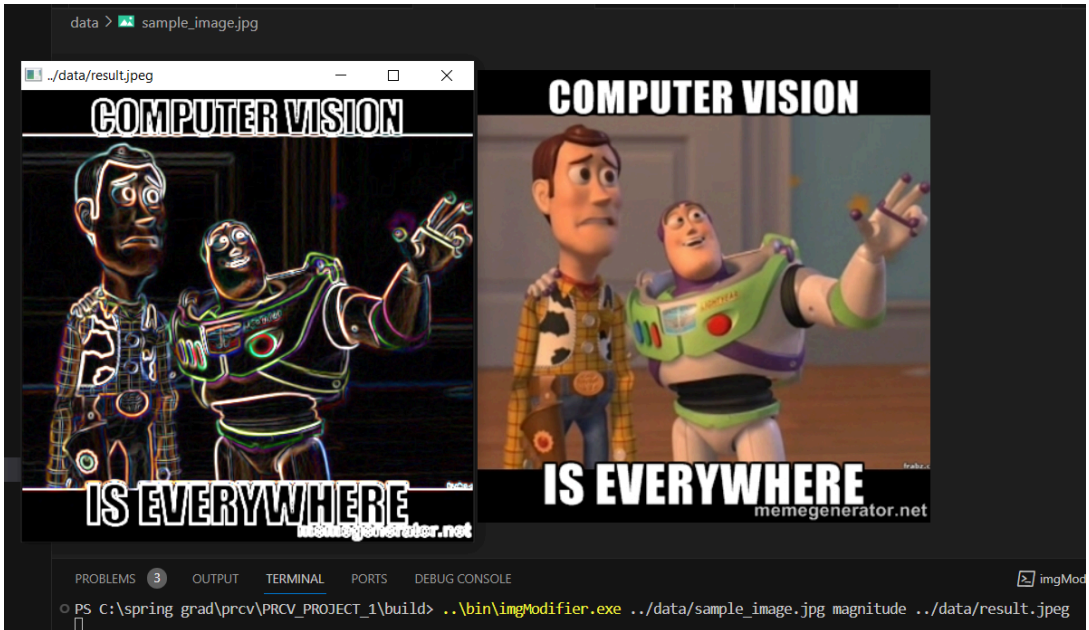
FACE_COLOR = "face_color"
LAPLACE = "laplace"
BRIGHTEN = "brighten"
DARKEN = "darken"

When the user runs the program the resultant effect is displayed on screen and is saved at the desired location. The user can click on ‘q’ to exit the preview of the saved image.

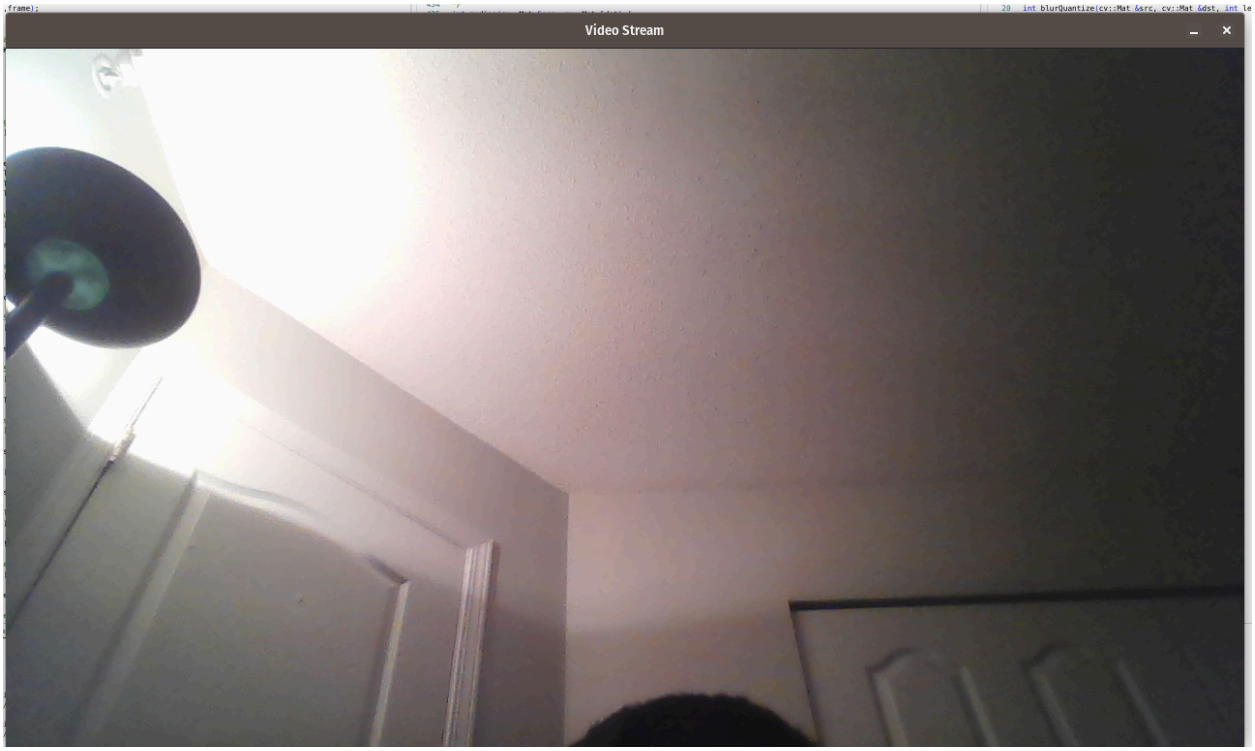
Sample of the “negative” command on still image

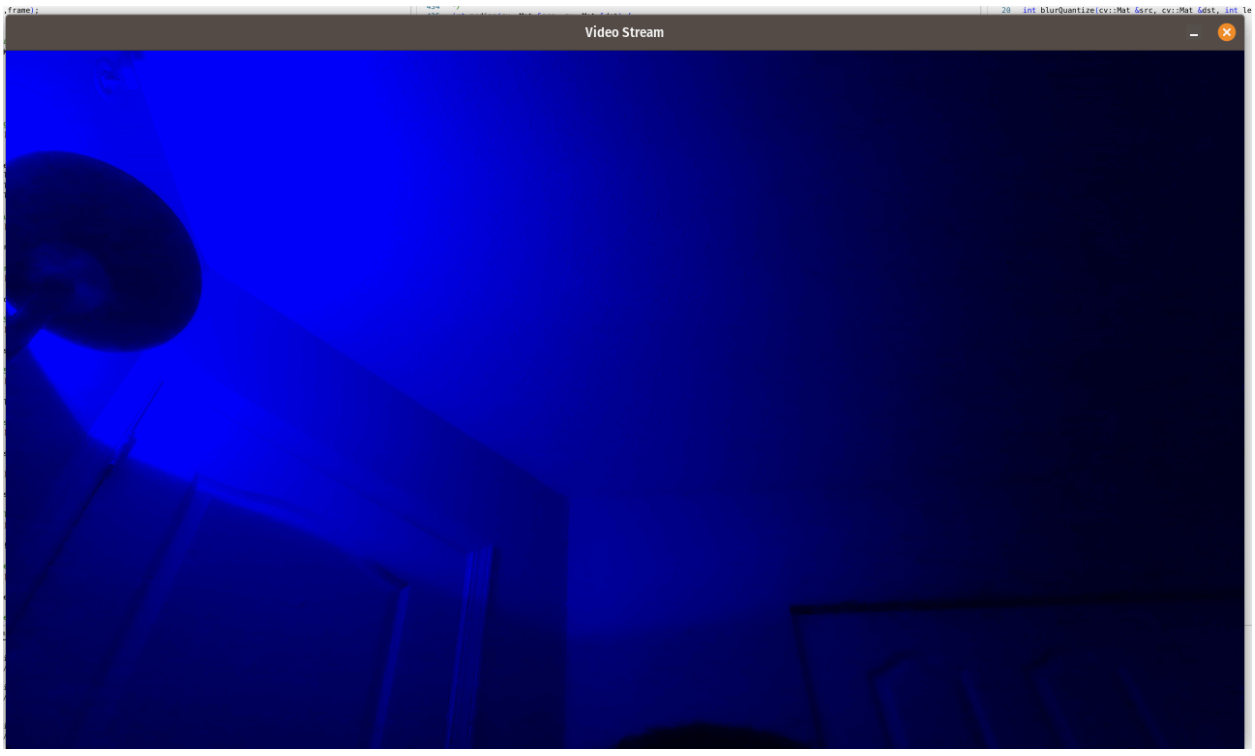


Sample of “magnitude” command on still image



Extension 4: Only blue





A short reflection of what you learned.

Justin: I get familiar with C++ and OpenCV, and debugging C++ code. Learned how to implement different filters in efficient way, e.g. two 1*5 is faster than one 5*5 filter. Learned Mask usage in OpenCV.

Abdulaziz: I got familiar with the different setup options available for OpenCV with C++. Learned about separable filters, face detection techniques and quick access via pointer allocation.

Acknowledgement of any materials or people you consulted for the assignment.

https://docs.opencv.org/4.x/

https://piazza.com/class/lraa0q8k4p03sr/

Consulted Erica Shepherd for C++ setup and doubts. She was really helpful.