Face augmentation assignment solution:

I have used dlib and opencv for face landmark detection and augmentation

a) Facial landmarks detection:

We can use dlib librray for face detection and inside each face region we can use pre-trained facial landmark detector inside the dlib library to estimate the location of ***68 (x, y)-coordinates*** that map to facial structures on the face.

```
Import dlib

detector = dlib.get_frontal_face_detector()

predictor =dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")

# detect faces in the grayscale image

rects = detector(gray, 1)

# determine the facial landmarks for the face region

shape = predictor(gray, rect)
```

The indexes of the 68 coordinates can be visualized on the image below:

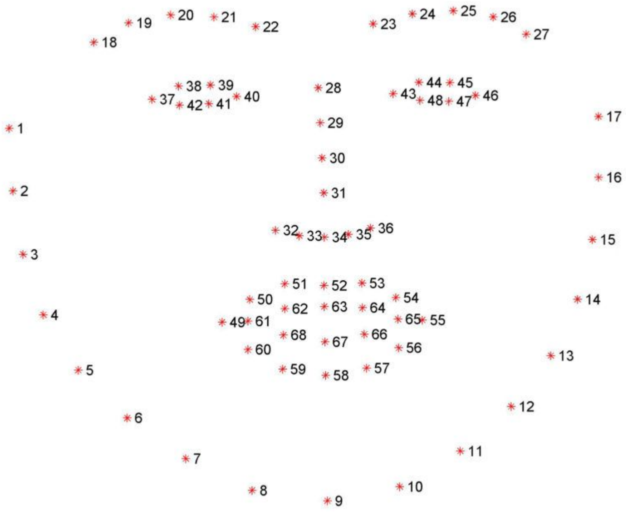


Fig. 1

b) Face component augmentation:

1) From the facial landmarks we can extract the marker points of the individual face components like Upper lip, lower Lip, Left eyebrow, Right eyebrow.
2) Once we have the marker points we fill the closed contour region formed by the marker points on a separate mask of the same dimension as the input image

```
mask = np.zeros(image.shape[0:2], np.uint8)
cv2.drawContours(mask, [pts], -1, 255 ,-1)
```

3) We use the mask to extract the list of all boundary points around the face component

```
        _, c, _ = cv2.findContours(mask,cv2.RETR_TREE,cv2.CHAIN_APPROX_NONE)
```

4) From the list of boundary points we find the extreme left and extreme right point index.

```
#smooth the contour curve
 c = cv2.blur(c, (1, 5), 0)
#Leftmost point in the contour
 start = c[:, :, 0].argmin()
#Rightmost point on the contour
 end = c[:, :, 0].argmax()
```

5) We use the extreme points to traverse between their indexes along the contour and add color gradient over the original image

```
        for idx in range(start, end):
            x, y = c[idx][0]
```

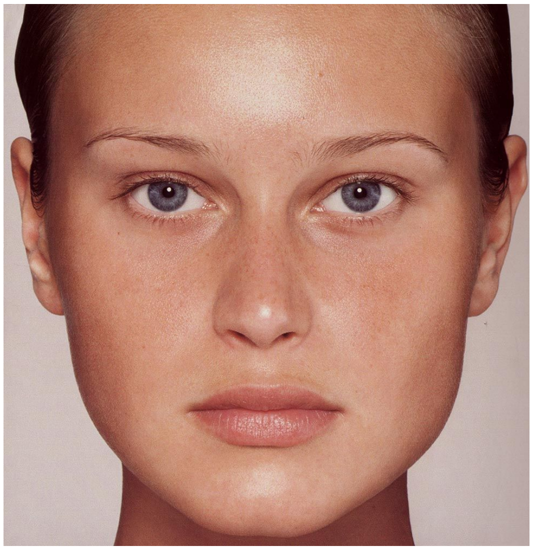
```
tempList = []
while True:
    if mask[y][x] == 255:
        tempList.append((y,x))
        y -= 1
    else:
        break
l = len(tempList)
if l > 0:
    s = 150/l
ct = 0
for p in tempList:
    v = 255 - int(s*ct)
    clone[p[0]][p[1]] = [v, 0, v]
    ct += 1
```

Demo application Usage:

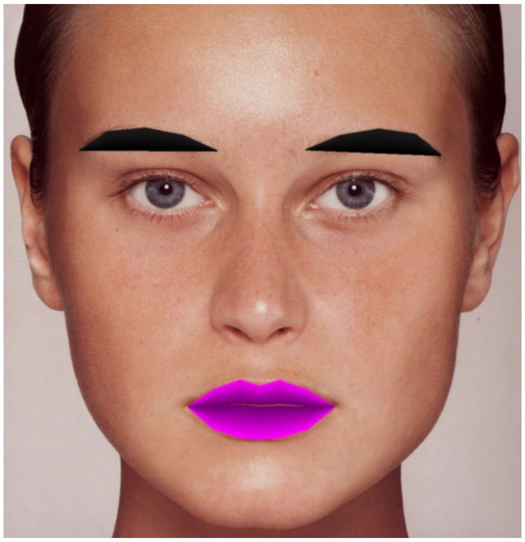
I have written python script based demo application in app.py and face_ar.py file.

a) To test on a sample image run app.py file as shown below :
   Python app.py img1.jpg
b) To test on device camera run face_ar.py file:
   Python face_ar.py

Please find below a sample input and output image:



Input Image (Fig. 2)



Output Image (Fig. 3)