In [1]:
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

Image Read(Access from loaction)

In [2]:
```
Rajat = cv2.imread('Rajat.jpg',0)
Anubhav = cv2.imread('Anubhav.jpg',0)
Solvay = cv2.imread('Solvay.jpg',0)
```

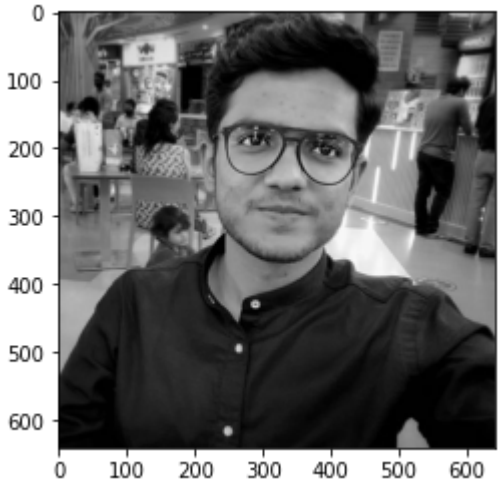
Convert RGB to Gray Scale (then it able to calculate)

In [3]:
```
plt.imshow(Rajat,cmap='gray')
```

Out[3]:
```
<matplotlib.image.AxesImage at 0xc740280>
```

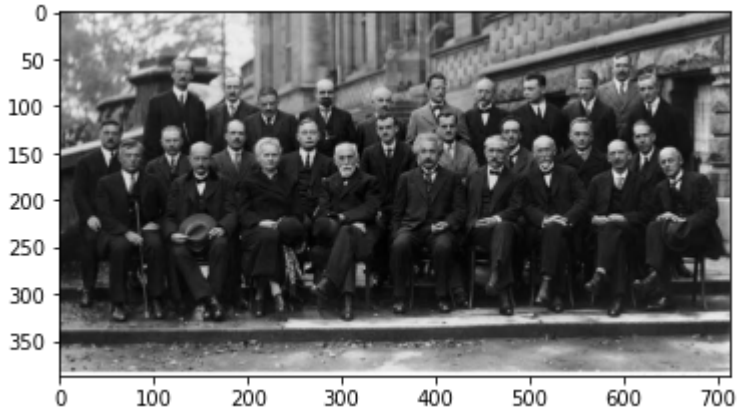
In [4]:
```
plt.imshow(Anubhav,cmap='gray')
```

Out[4]:
```
<matplotlib.image.AxesImage at 0xd7dab20>
```



In [5]:
```
plt.imshow(Solvay,cmap='gray')
```

Out[5]:
```
<matplotlib.image.AxesImage at 0xd818c70>
```



Face Detection

In [6]:
```
face_cascade = cv2.CascadeClassifier('D:/IDM/Program/opencv/sources/data/haarcascades/h
```

Function which dtecte face from Image

In [26]:
```
def detect_face (img):

    face_img = img.copy() #copy oringanm image into face_img variable
    face_rect = face_cascade.detectMultiScale(face_img)

    for(x,y,w,h) in face_rect:  #loop for rectangle in face
        cv2.rectangle(face_img,(x,y),(x+w,y+h),(255,0,0), 4) #(image_souce,intialpoint(

    return face_img
```

In [27]:
```
img1=detect_face(Rajat)
```

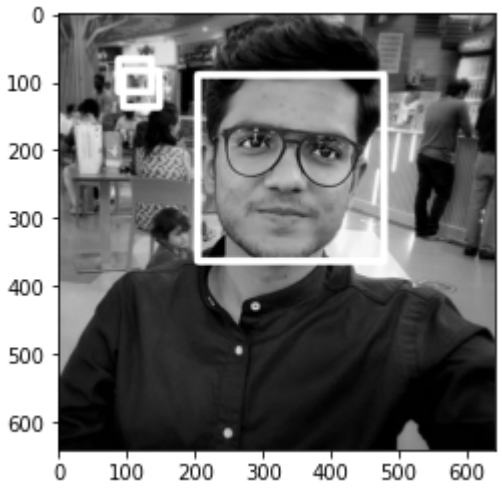
In [28]:
```
plt.imshow(img1,cmap='gray')
```

Out[28]:
```
<matplotlib.image.AxesImage at 0x11e6640>
```

In [10]:
```
img2 = detect_face(Anubhav)
```

In [11]:
```
plt.imshow(img2,cmap='gray')
```

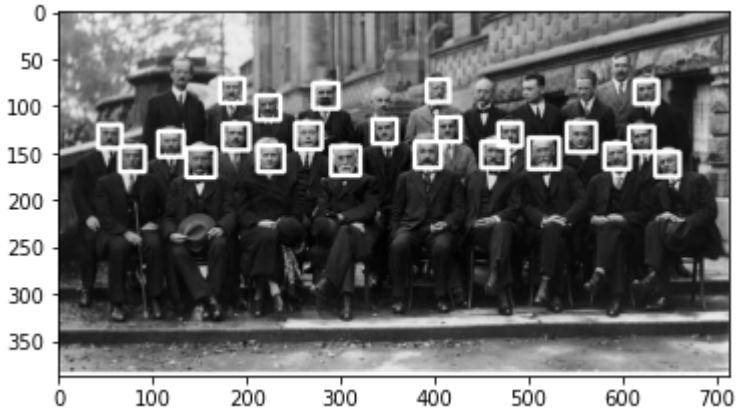
Out[11]: <matplotlib.image.AxesImage at 0x1475bfa0>



In [32]:
```
img3 = detect_face(Solvay)
```

In [33]:
```
plt.imshow(img3,cmap='gray')
```

Out[33]: <matplotlib.image.AxesImage at 0x1240448>

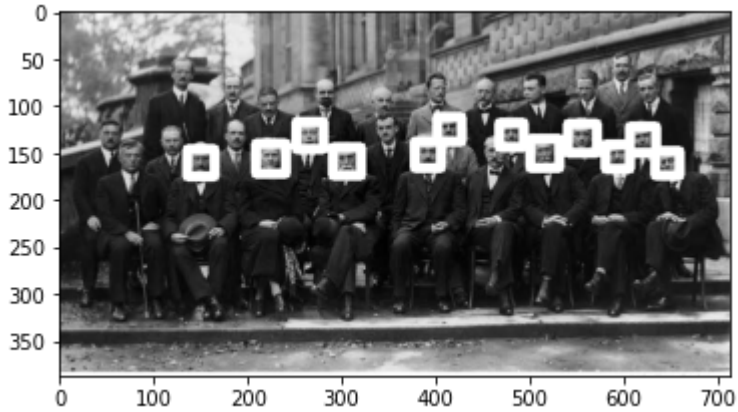


In [14]:
```
def detect_face1 (img):

    face_img = img.copy() #copy oringanm image into face_img variable
    face_rect = face_cascade.detectMultiScale(face_img, scaleFactor=1.2, minNeighbors=5

    for(x,y,w,h) in face_rect:  #loop for rectangle in face
        cv2.rectangle(face_img,(x,y),(x+w,y+h),(255,0,0), 10) #(image_souce,intialpoint

    return face_img
```

In [29]:
```
img4 = detect_face1(Solvay)
```

In [30]:
```
plt.imshow(img4,cmap='gray')
```

Out[30]: <matplotlib.image.AxesImage at 0x147520e8>



Real_Time Face Detection (using Web_Cam)

In [17]:
```
cap = cv2.VideoCapture(0)
while True:
    ret,frame = cap.read()
    #Access detect_face() to detect face in Web_Cam
    frame = detect_face(frame)
    cv2.imshow('Camrea',frame)

    if cv2.waitKey(1)==13:
        break

cap.release()
cv2.destroyAllWindows()
```

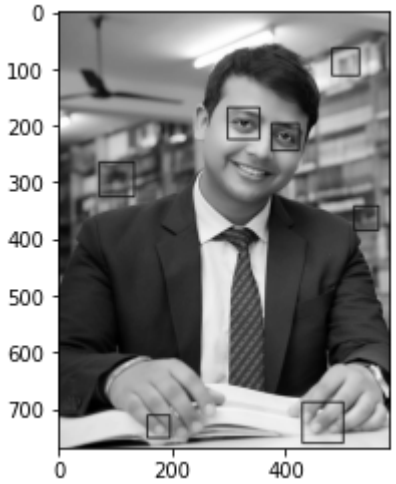
EYE Detection (using Image)

In [18]:
```
eye_cascade = cv2.CascadeClassifier('D:/IDM/Program/opencv/sources/data/haarcascades/ha
```

In [24]:
```
def detect_eye (img):

    eye_img = img.copy() #copy oringanm image into face_img variable
    eye_rect = eye_cascade.detectMultiScale(eye_img)

    for(x,y,w,h) in eye_rect:  #loop for rectangle in face
        cv2.rectangle(eye_img,(x,y),(x+w,y+h),(18,255,255), 2) #(image_souce,intialpoin

    return eye_img
```

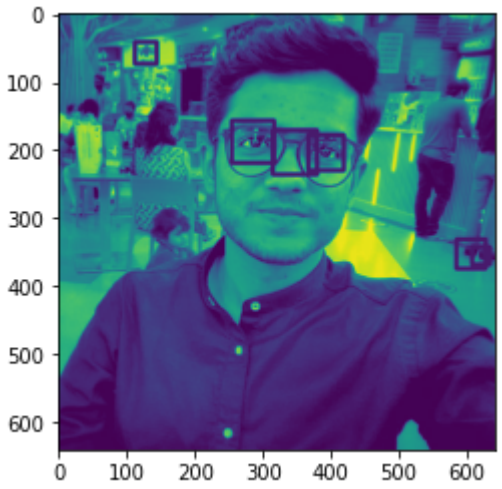
In [25]:
```
img6 = detect_eye(Rajat)
plt.imshow(img6,cmap='gray')
```

Out[25]:
```
<matplotlib.image.AxesImage at 0x11b6160>
```



In [21]:
```
img7 = detect_eye(Anubhav)
plt.imshow(img7)
```

Out[21]: <matplotlib.image.AxesImage at 0x137b38e0>



Real-Time EYE Detection (using Web_Cam)

In [22]:
```
cap = cv2.VideoCapture(0)
while True:
    ret,frame = cap.read()
    #Access detect_eye() to detect EYE in Web_Cam

    frame = detect_eye(frame)

    cv2.imshow('Camrea',frame)

    if cv2.waitKey(1)==13:
        break

cap.release()
cv2.destroyAllWindows()
```

Real-Time Face & EYE Detection (using

Web_Cam)

In [34]:
```
cap = cv2.VideoCapture(0)
while True:
    ret,frame = cap.read()
    #Access detect_eye() to detect EYE in Web_Cam

    frame = detect_face(frame)
    frame = detect_eye(frame)

    cv2.imshow('Camrea',frame)

    if cv2.waitKey(1)==13:
        break

cap.release()
cv2.destroyAllWindows()
```

In [ ]: