FINAL PROJECT

TEAM LEADER: MONIKA PRIYADHARSHINI.G

TEAM MEM 1: JOEITA PRIYADARSHINI.A

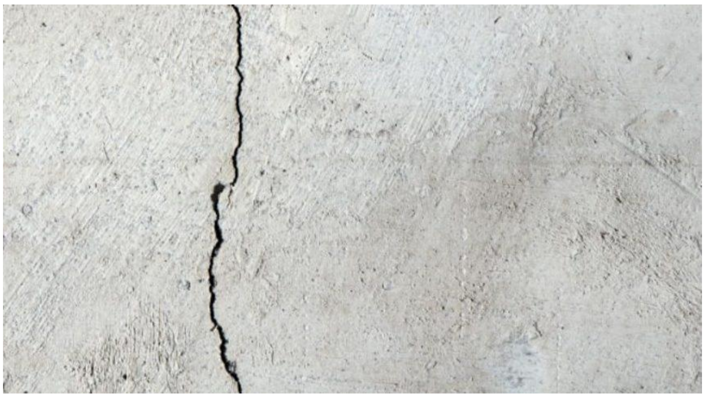
TEAM MEM2: DEEKSHANYA.G

SQUID:STREET QUALITY IDENTIFICATION

PROGRAM:

```
import numpy as np
import cv2
from matplotlib import pyplot as plt
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
organization = "96ei56"
deviceType = "SQUID"
deviceId = "12333"
authMethod = "token"
authToken = "27042023"
def ibmstart(x):
      def myCommandCallback(cmd):
        print("Command received: %s" % cmd.data['command'])
print(cmd)
   try:
      deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": 
authMethod, "auth-token": authToken}
      deviceCli = ibmiotf.device.Client(deviceOptions)
  except Exception as e:
      print("Caught exception connecting device: %s" % str(e))
      sys.exit()
     deviceCli.connect()
    lat=random.randint(9,37)
    long=random.randint(68,97)
    data = { 'latitude' : lat, 'longitude': long ,'Status': x}
 def myOnPublishCallback():
        print ("Published Status = %s" % x, "to IBM Watson")
    success = deviceCli.publishEvent("SQUID", "json", data, qos=0,
on_publish=myOnPublishCallback)
    if not success:
        print("Not connected to IoTF")
img = cv2.imread('Input Set/Cracked_07.jpg')
flag=0
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
blur = cv2.blur(gray,(3,3))
img_log = (np.log(blur+1)/(np.log(1+np.max(blur))))*255
img_log = np.array(img_log,dtype=np.uint8)
bilateral = cv2.bilateralFilter(img_log, 5, 75, 75)
edges = cv2.Canny(bilateral,100,200)
kernel = np.ones((5,5),np.uint8)
closing = cv2.morphologyEx(edges, cv2.MORPH_CLOSE, kernel)
orb = cv2.ORB_create(nfeatures=1500)
keypoints, descriptors = orb.detectAndCompute(closing, None)
featuredImg = cv2.drawKeypoints(closing, keypoints, None)
cv2.imwrite('Output Set/CrackDetected-7.jpg', featuredImg)
flag=1
plt.subplot(121),plt.imshow(img)
plt.title('Original'),plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(featuredImg,cmap='gray')
plt.title('Output Image'),plt.xticks([]), plt.yticks([])
print(flag)
ibmstart(flag)
plt.show()
```

INPUT:



OUTPUT: