AUTOMATED BLOOD DONORS FINDER WORKING MODEL DEMO

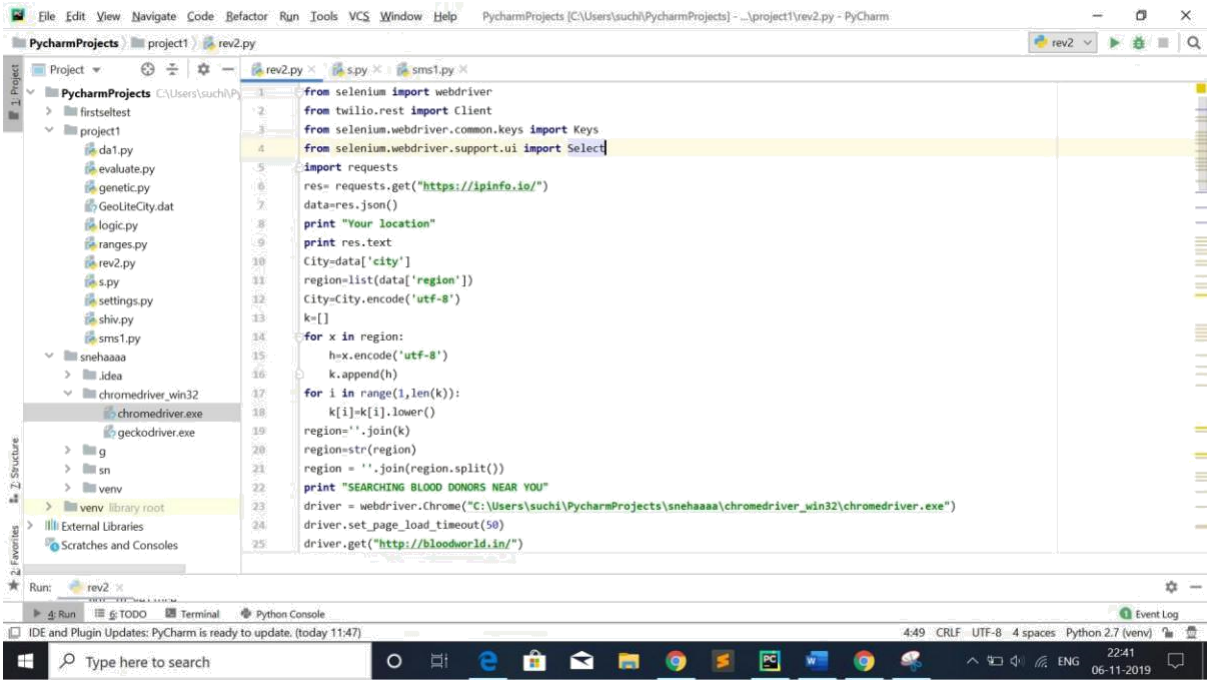
ALGORITHM

1. Firstly, when in emergency, all we would have to do is just to type our blood type and everything else will be taken care by this automated bot.
2. The model finds the location of the user from where this code was ran using their ip address.
3. Later, it sends this location to any blood donor website that is specified in the code and can find the registered donors' numbers from that site using selenium tool.
4. Now, model sends a dummy message to all the blood donors in the same city to find if they are right now available in the city. For, this model needs to geo-tag. Its not possible to geo tag for unauthorised beginners. So, as a modified version, I find their ip addresses. This is just a learner's project and has never been used in real world since its not permitted.
5. As soon as ip addresses of users are found, model filters only the users' who are in same location as the victim.
6. After this step, we now know the users and their numbers in the same location as victim.
7. Later, I have used twilip api to send mass message to all the available registered donors asking for help to donate blood to the victim.
8. Summing up, this automated bot has reduced manual workload of identifying and notifying donors.

TOOLS USED

1. selenium as automation tool through PyCharm IDE
2. twilio API to send texts to everyone's mobile.
3. iplogger and ipinfo to obtain ip addresses and match them to their current geolocations.
4. Python to code.

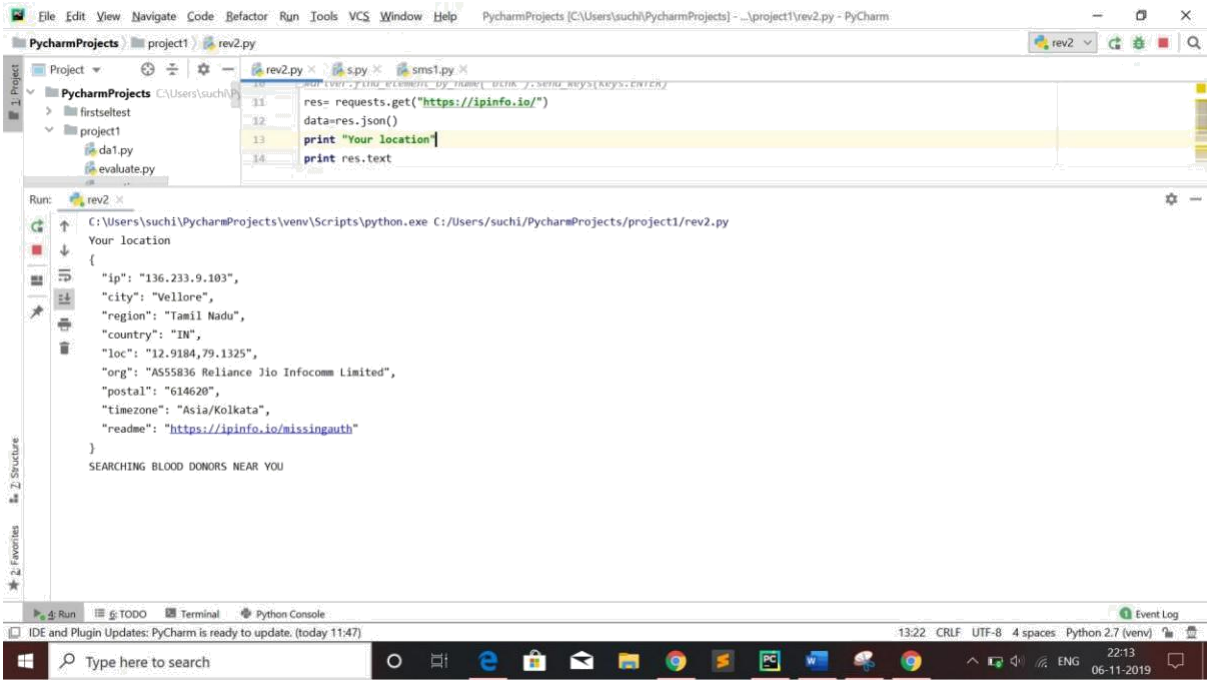
CODE (sample) SNIPPET:



```
from selenium import webdriver
from twilio.rest import Client
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.ui import Select
import requests
res= requests.get("https://ipinfo.io/")
data=res.json()
print "Your location"
print res.text
City=data['city']
region=list(data['region'])
City=City.encode('utf-8')
k=[]
for x in region:
    h=x.encode('utf-8')
    k.append(h)
for i in range(1,len(k)):
    k[i]=k[i].lower()
region=''.join(k)
region=str(region)
region = ''.join(region.split())
print "SEARCHING BLOOD DONORS NEAR YOU"
driver = webdriver.Chrome("C:\Users\suchi\PycharmProjects\snehaaaa\chromedriver_win32\chromedriver.exe")
driver.set_page_load_timeout(50)
driver.get("http://bloodworld.in/")
```

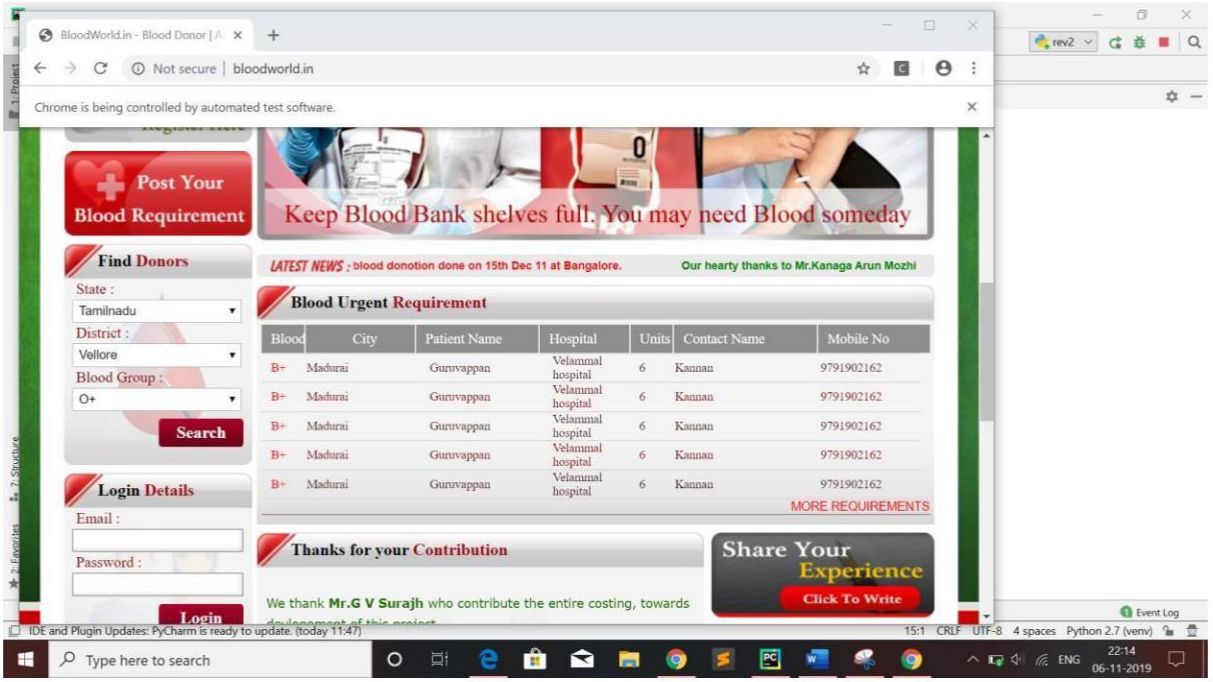
OUTPUT

1. FIRSTLY, it's detecting our location as soon as we run the code.

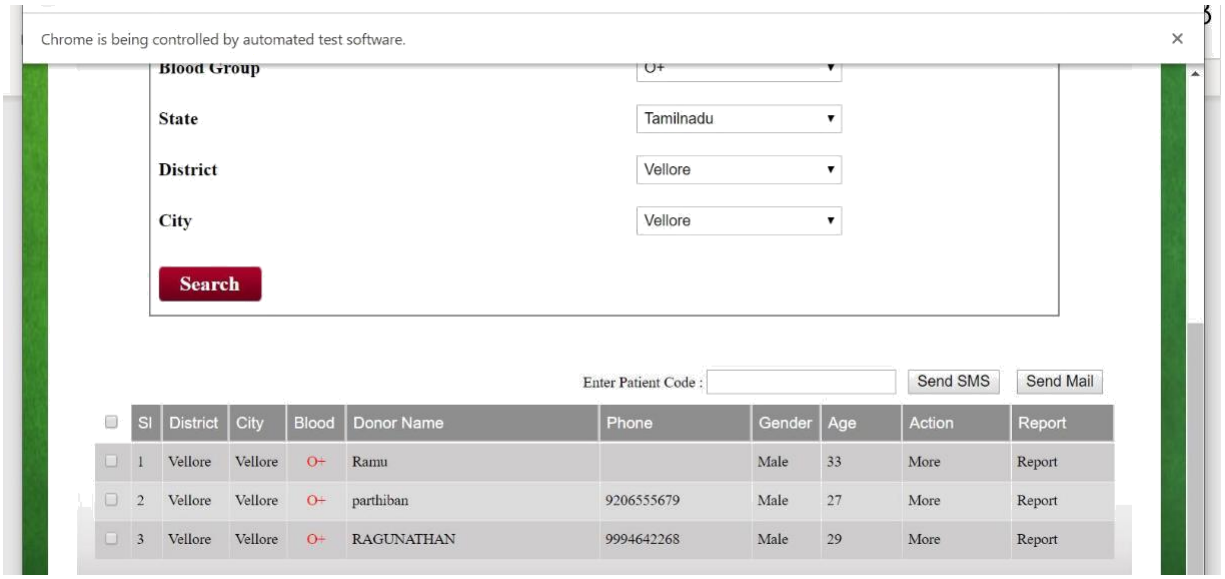


```
C:\Users\suchi\PycharmProjects\venv\Scripts\python.exe C:/Users/suchi/PycharmProjects/project1/rev2.py
Your location
{
  "ip": "136.233.9.103",
  "city": "Vellore",
  "region": "Tamil Nadu",
  "country": "IN",
  "loc": "12.9184,79.1325",
  "org": "AS55836 Reliance Jio Infocomm Limited",
  "postal": "614620",
  "timezone": "Asia/Kolkata",
  "readme": "https://ipinfo.io/missingauth"
}
SEARCHING BLOOD DONORS NEAR YOU
```

Then, to our location, it automatically checks the available blood donors in "https//bloodworld.in" website for O+ blood donors.



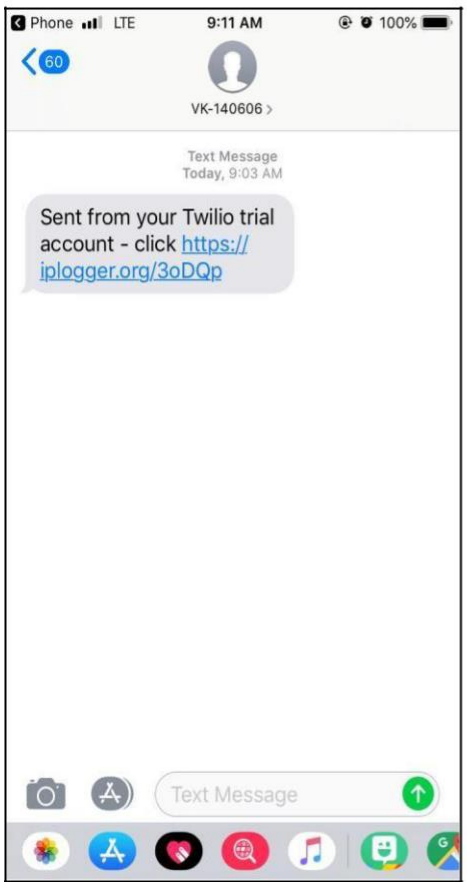
Right now, it has detected our location as Vellore and has searched for O+ donors in Vellore. We have these donors' numbers in the website.



The model has now successfully loaded the same numbers:

```
C:\Users\suchi\PycharmProjects\venv\Scripts\python.exe C:/Users/suchi/PycharmProjects/project1/rev2.py
Your location
{
  "ip": "136.233.9.103",
  "city": "Vellore",
  "region": "Tamil Nadu",
  "country": "IN",
  "loc": "12.9184,79.1325",
  "org": "AS55836 Reliance Jio Infocomm Limited",
  "postal": "614620",
  "timezone": "Asia/Kolkata",
  "readme": "https://ipinfo.io/missingauth"
}
SEARCHING BLOOD DONORS NEAR YOU
Available PEOPLE'S CONTACT NUMBERS:
['+919206555679', '+919994642268', '+918610394147', '+918870808876']
click https://iplogger.org/3oDQp              ----this message is sent to them----------
```

Now as we can see, this message has been sent to all the donors from that website to detect their location. To simply demonstrate this, I have added dummy numbers to the code ("+918610394147", "+918870808876")



Mobile 1

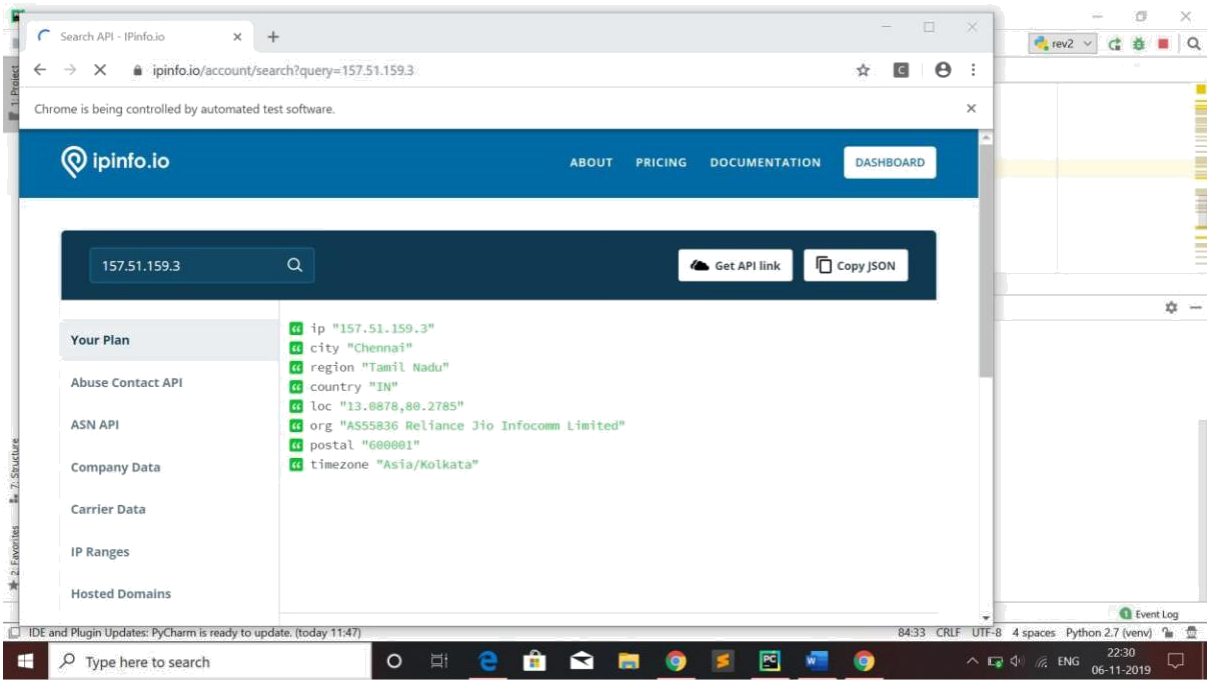
Mobile 2

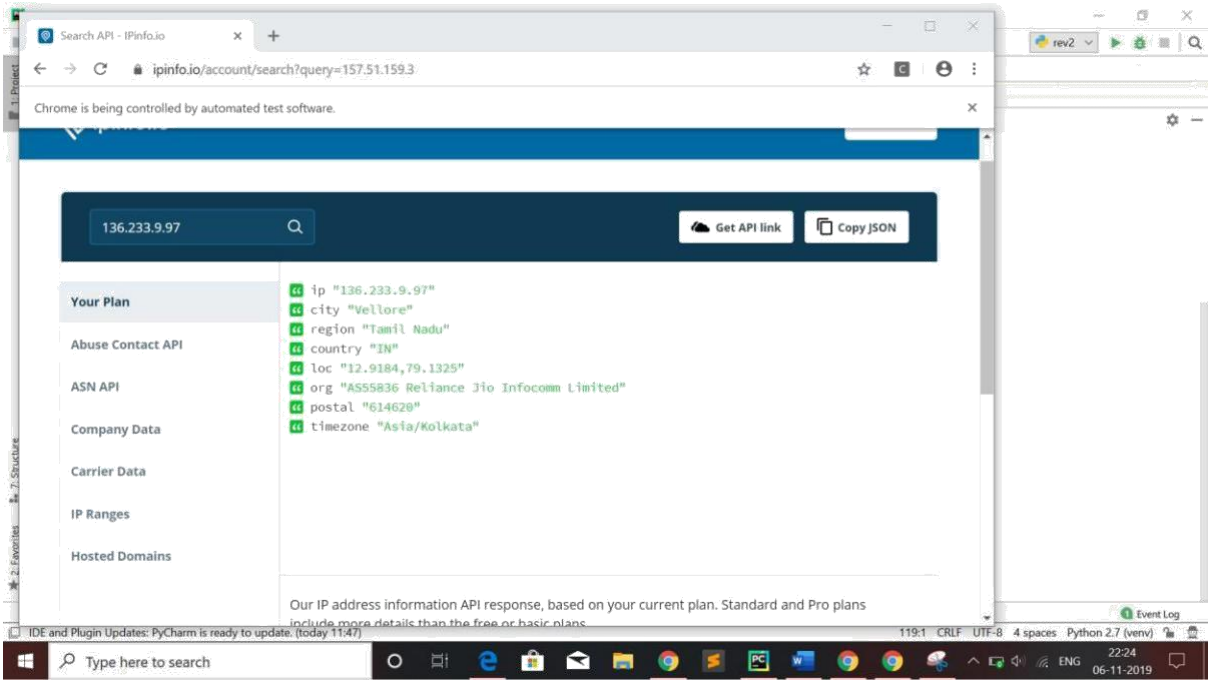
As we see, mode has sent a normal link message to all the donors' registered from that place.

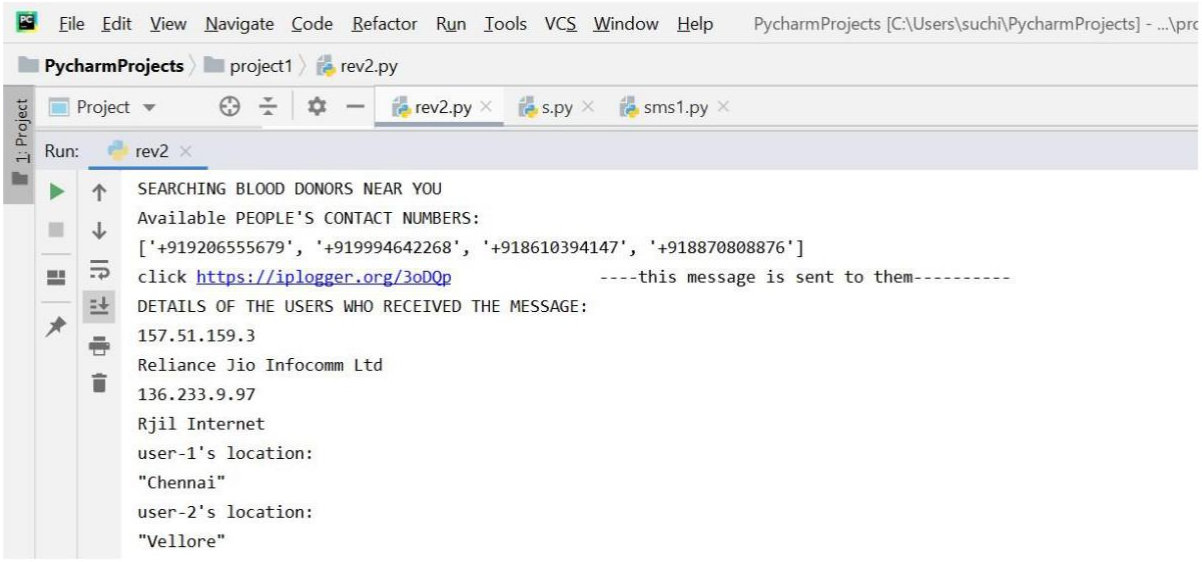
This has simply been sent to detect their locations.

As soon as the message has been sent, the iplogger website gets updated with everyone's ip address (to whomever the message has been sent)

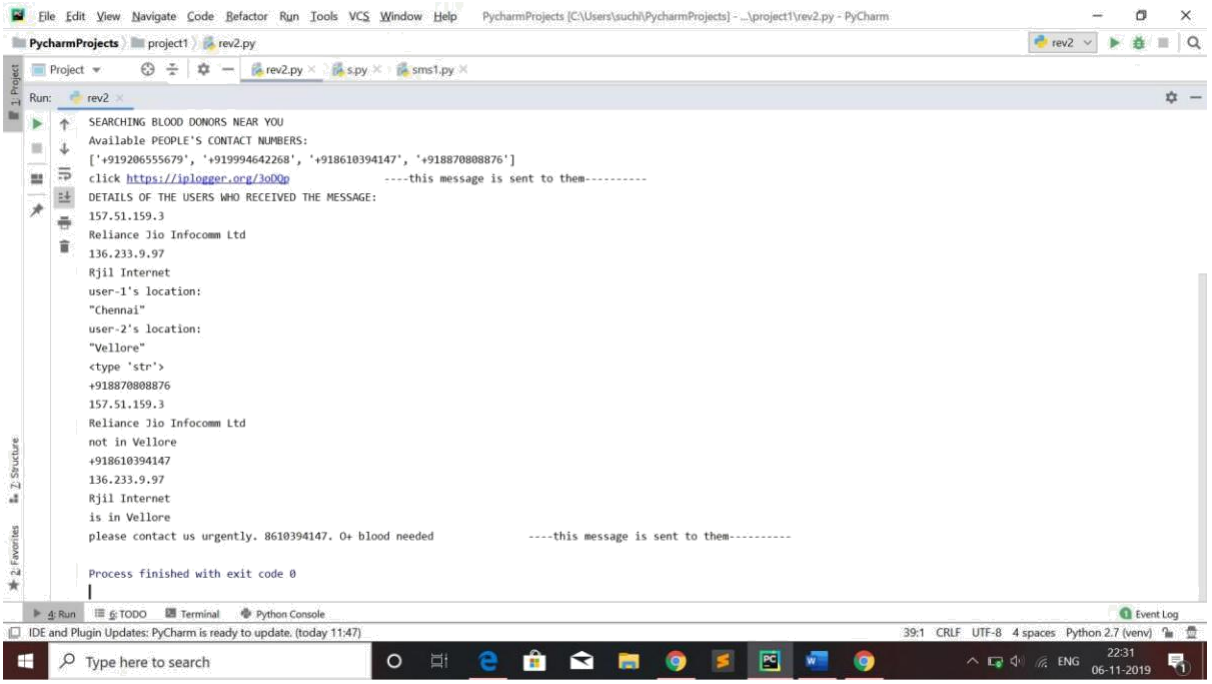
Now, later the model matches these ip addresses and find their locations



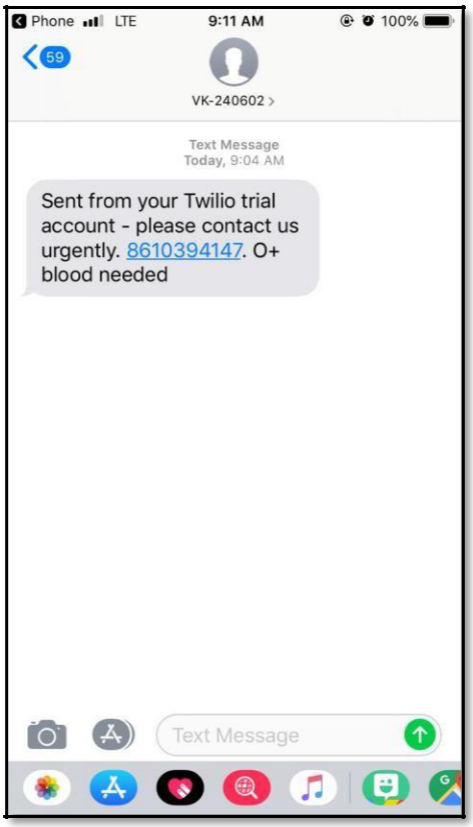




Now, model has received the location of all the people to whomever it have sent a dummy message.



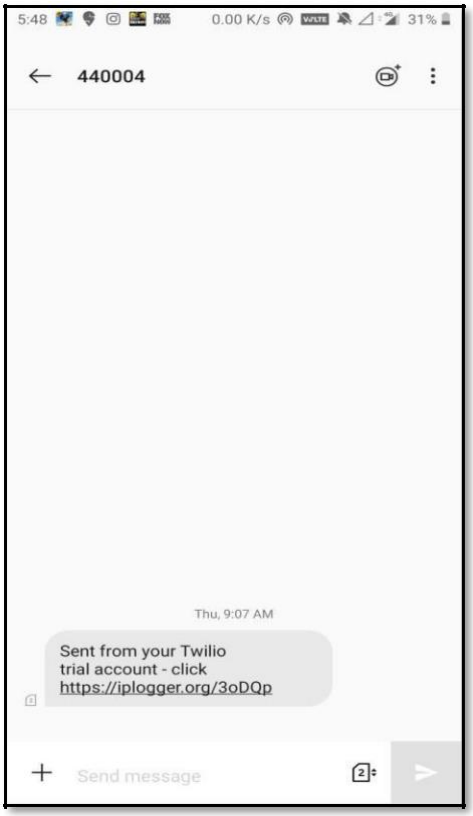
Here, model has retrieved the details of the users who are in Vellore and then has sent a message to “contact us urgently and that we are in need of O+ blood”.



Mobile 1

Received the Emergency Message

Along with the tracing message



Mobile 2

Received only one message

We can see that only +8610394147 has received the message but the other number hasn't received the message since they are not in Vellore.

This way we can smartly geotag people around us alone and text them or call them alone to ask for help. This saves much time for us and also helps us to know available people around us in a very short time. To do this manually might take a lot of time if we have 1000 registered donors and only if 5-10 are available in them (hypothetically). But this way, we save much time in case of emergencies.