

Live Dashboard for IP Monitoring using Python and Flask

R Manoj Kumar¹, N Gokulakannan², A Shyamala Prasanna³

¹Student, Bannari Amman Institute of Technology, Sathyamangalam.

²Student, Bannari Amman Institute of Technology, Sathyamangalam.

³Assistant Professor, Bannari Amman Institute of Technology, Sathyamangalam.

Abstract - In this Project, We are going to create a Dashboard For IP Monitoring and to know the live status dynamically through the dashboard The Dashboard consists of the following features like IP, Status of the IP (whether it is in working Condition or Not), Default Gateway, Routers (whether the IP is Reachable or Not) and Date and Time of Last Updated status in the Dashboard. These Dashboard Details are also stored in the database to find the Utilization Measurement of each IP. The Input is in the Format of .xlsx and it has N number of IPs in it. For storage MongoDB is used and for the backend Python and Flask Framework is used. Jinja2 Template is used for Decoding the Datas that sent from Flask Framework to HTML. Ngrok is used for connecting Localhost and Web server (Third Party).

Key Words: IP, Default gateway, Mongoddb, Python, Flask Framework, Ngrok.

1. INTRODUCTION

In many organizations, they have certain IPs or Hardware Modules that need to be monitored and that live status is known to all in that organization. Along with the status we need to add some more information about the IP or Hardware Modules. In this project, the open IPs were taken from the internet to test its status, default gateway, traceroute of the IP along with the Last Updated Date and Time.

1.1 INTERNET PROTOCOL

Internet Protocol (IP) was developed by VintCerf and BobKahn in 1974. It is an end-to-end connection process, in which the packets (known as datagram – the fundamental unit of DATA TRANSFER) are transferred from source to destination and the transmission depends on the IP Address. That's why IP resembles the postal management system by itself. The address is allowed to be mentioned at the package in the package's header and then the package is dropped to the system but there is not any link between the sender and receiver. Also, IP establishes two-way communication because IP establishes the connection for two of the hosts and allows them to send and receive messages back & forth over a particular time period which is priorly mentioned.

1.2 PINGING THE IP ADDRESS

To determine whether the destination hosts (whose IP address is mentioned on the package's header) are reachable or not, the IP pinging is used.

SYNTAX: ping <IP ADDRESS>

```
C:\Users\Uhnder>ping 142.250.76.69

Pinging 142.250.76.69 with 32 bytes of data:
Reply from 142.250.76.69: bytes=32 time=15ms TTL=120
Reply from 142.250.76.69: bytes=32 time=17ms TTL=120
Reply from 142.250.76.69: bytes=32 time=51ms TTL=120
Reply from 142.250.76.69: bytes=32 time=18ms TTL=120

Ping statistics for 142.250.76.69:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 15ms, Maximum = 51ms, Average = 25ms
```

Fig -1: Pinging the IP

1.3 TRACEROUTING THE IP ADDRESS

Tracert is used in troubleshooting, and it will tell you whether the packet has reached the destination or stopped on the network. If stopped it indicates mismatch configuration in the settings of the router or IP address that do not exist. Tracert has a maximum of 30 hops which is default and 1 hop will tell you the default gateway and last hop consist of the IP which we entered in tracert command (if Reachable). In Between hops consists of Routers.

SYNTAX: tracert -d -h <maximum hops> -j <hostl-list> -w <timeout> <target_host>

```
C:\Users\Uhnder>tracert 142.250.76.69

Tracing route to maa05s14-in-f5.1e100.net [142.250.76.69]
over a maximum of 30 hops:
  0  28 ms  12 ms  18 ms  10.10.64.1
  1  42 ms  1 ms  1 ms  192.168.200.1
  2  5 ms  5 ms  5 ms  10.0.0.161
  3  15 ms  15 ms  16 ms  209.85.174.10
  4  21 ms  17 ms  16 ms  74.125.242.129
  5  20 ms  16 ms  17 ms  142.250.228.245
  6  17 ms  16 ms  16 ms  maa05s14-in-f5.1e100.net [142.250.76.69]
```

Fig -2: Trace Routing IP Address

2. TECHNICAL REVIEW

2.1 PYTHON – PROGRAMMING LANGUAGE

Python is the simplest and easiest language among all computer vision languages. The creator of Python language is Guido VanRossum he named this language as Python. The name behind the python is he named the word the Python after the show in BBC Monty Python’s Flying Cars. Python is a rare language and it is easy, powerful, simple because it is easy to learn and easy to program and debugging is easy. Python has easy syntax so programmers love to code by using Python.



Fig -3: Python Logo

2.2 FLASK WEB FRAMEWORK

Flask is a micro web framework written in python programming language and it was created in 2004 by Armin Ronacher. The Main object of Flask Framework is to make web based applications quickly, easily, as well as ability to improve difficult application. It consists of WSGI utility Libraries like jinja2 template and Werkzeug and it becomes the most popular python based web framework.



Flask

Fig -4: Flask Logo

Features:

- Based on Unicode
- Secure the Cookies from insecure channel
- Documentation is Extensive

2.3 JINJA2 TEMPLATE

Jinja2 is one of the Libraries in python. It’s basically a templating language for python. It is More Readable because of its syntax and is easy to visualize from Html script.

Main features are template inheritance, debugging becomes easy, optimal.

Basic Syntax of jinja2:

{{...}} → for printing the content in-between curly brackets.

{{% ... %}} → statements.

{{# ... #}} → comments to explain the scripts/code.

2.4 NGROK

Ngrok is an Developers efficient tool to test works like localhost, web based works and APIs. The main use of Ngrok is to create a public secure URL and allow localhost to connect to the internet behind all the firewalls(Third Party).In this project, Ngrok console is used to communicate with localhost(127.0.0.1:5000).The advantage of Ngrok is rapid response speed.



Fig -5: Ngrok Logo

2.4 MONGODB

MongoDB is an open source database and its written c++. Data stored and queried back in the format of BSON(Binary Serialized format).It supports Master-Slave replica which avoids duplication and backup. it’s a document oriented database and doesn’t have joins, so it makes it easier to write a query.



Fig -6: MongoDB Logo

3. PROPOSED SYSTEM

The block diagram shown in figure 6.1 explains about how the IP is monitored and datas is stored in Database and explains briefly about the process.

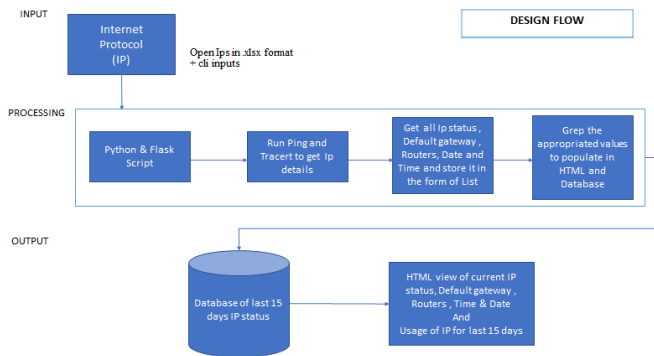


Fig -7: Block diagram

S.No	Ip	Status	Default Gateway	Routers	Last Updated Date And Time
1	142.250.76.69	Working	10.10.64.1	192.168.200.1 10.0.161 209.85.174.10 74.125.242.129 142.250.228.245 Reachable	Wed Mar 24 19:07:33 2021
2	216.58.197.46	Working	10.10.64.1	192.168.200.1 10.0.161 209.85.174.10 108.170.253.97 108.170.254.108 Reachable	Wed Mar 24 19:08:11 2021
3	74.125.24.189	Working	10.10.64.1	192.168.200.1 10.0.161 209.85.174.10 108.170.253.120 72.14.239.151 74.125.17.258 64.253.174.185 Unreachable	Wed Mar 24 19:10:28 2021
4	192.168.200.1	Working	10.10.64.1	Unreachable	Wed Mar 24 19:10:46 2021
5	192.168.96.98	Not Working	10.10.64.1	Unreachable	Wed Mar 24 19:14:03 2021

Fig- 10: Output in the Localhost

3.1 INPUT

Input consists of n number of Internet protocols (IP) are stored in the .xlsx format.

Fig-8: IP Excel sheet

3.2 PROCESSING

A python script which automatically collects the information of Internet Protocol(IP) and grep the appropriate information and convert it into a list of lists. With the help of flask it is hosted in localhost. For running python and flask script we need to activate the virtual environment.

```

C:\Users\user>python main.py
[{"ip": "142.250.76.69", "status": "Working", "default_gateway": "10.10.64.1", "routers": ["192.168.200.1", "10.0.161", "209.85.174.10", "74.125.242.129", "142.250.228.245"], "last_updated": "Wed Mar 24 19:00:43 2021"}, {"ip": "216.58.197.46", "status": "Working", "default_gateway": "10.10.64.1", "routers": ["192.168.200.1", "10.0.161", "209.85.174.10", "108.170.253.97", "108.170.254.108"], "last_updated": "Wed Mar 24 19:08:11 2021"}, {"ip": "74.125.24.189", "status": "Working", "default_gateway": "10.10.64.1", "routers": ["192.168.200.1", "10.0.161", "209.85.174.10", "108.170.253.120", "72.14.239.151", "74.125.17.258", "64.253.174.185"], "last_updated": "Wed Mar 24 19:10:28 2021"}, {"ip": "192.168.200.1", "status": "Working", "default_gateway": "10.10.64.1", "routers": [], "last_updated": "Wed Mar 24 19:10:46 2021"}, {"ip": "192.168.96.98", "status": "Not Working", "default_gateway": "10.10.64.1", "routers": [], "last_updated": "Wed Mar 24 19:14:03 2021"}]
  
```

Fig -9: Running Python & Flask Script

After running the script, the debugger becomes active (whenever there is a change in script it will automatically re-run the script). The last line in the Fig -5 states that the script is successfully compiled and executed, now running on <http://127.0.0.1:5000/>.

127.0.0.1 - Localhost
5000 - Port Number

Now this dashboard data is also accumulated in MongoDB to find the utilization and usage of IPs in the past. For that we have to create a new connection in MongoDB (mongodb://mongodb0.example.com:27017) and now we have to connect it by using "client=pymongo.MongoClient('mongodb://127.0.0.1:27017/')" in python script.

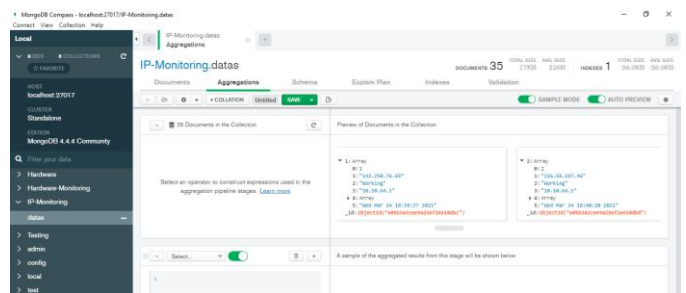


Fig -11: MongoDB Compass

3.3 OUTPUT

Now we have to connect localhost to the internet using Ngrok. For that we have to download Ngrok software and connect to the localhost port using "ngrok http <port_number>" in the command prompt.

```

ngrok by @inconshreveable

Session Status      online
Account             manojmango02@gmail.com (Plan: Free)
Version             2.3.35
Region              United States (us)
Web Interface       http://127.0.0.1:4040
Forwarding           http://d155405b73d6.ngrok.io -> http://localhost:5000
                    https://d155405b73d6.ngrok.io -> http://localhost:5000

Connections
t1l  oppa  rt1l  rt5  p50  p90
0    0     0.00  0.00  0.00  0.00
  
```

Fig -12: Connecting Ngrok with Localhost

The link generated by ngrok is <http://d155405b73d6.ngrok.io> . if we surf this link, we will get the exact output which we see in the localhost.

Fig -13: Output in the Ngrok Generated Link

4. CONCLUSIONS

Before the development of the project, we need to ping and tracert IP each and every time when we want to know the information of Internet Protocol. Therefore, live dashboard hardware tracker, status monitoring of IP using python & flask scripts will automatically run after every certain interval of time using app.scheduler library in python, which will help the user/developer to know the information of Internet Protocol just by clicking a link. Dashboard also has a date & time column which will help users/developers to know the last updated time of the information. In the end, this project provides convenience to the user to access the information collected which perfected the existence of technology to assist human's needs.

FUTURE SCOPE

This project can be scaled (in near future) with machine learning and log analytics, we can understand

- a.) Failure pattern analysis at each Hardware Module.
- b.) help automatic triage, differentiate hw/sw issues & probably fix specific sets of issues without much downtime.
- c.) Predict the replacement of hw parts based on utilization rate.

REFERENCES

- [1] Zhang, Gary. "HWmonitor-CPU Temperature Monitor for Windows 10". Garyzzc. Retrieved 16 December 2018.
- [2] Lokhande, P. S.; Aslam, Fankar; Hawa, Nabeel; Munir, Jumal; Gulamgaus, Murade "Efficient way of web development using python and flask", ijarcs March 2015.
- [3] Sashank Sridhar, Sowmya Sanagavarapu,S. Chitrakala "Cross-Platform Remote Desktop Sharing with IP Tunneling", 3 July 2020.