1. **Project composition:**

This project is a web-based project, we used React.js in the front-end and Python in the backend (includes Flask web framework). Backend end can be separated into 2 parts: Web-Communication module(app.py) and Pi-Communication module (main.py). They used 2 different python interpreters and communicate by txt files.

1. **Team member contributions:**

Heng Sun (Jimmy) is responsible for developing and testing the front-end codes, also communicates with backend developers to negotiate technical details. His contribution counts about 33.3%.

Zihan Wei (Alex) is responsible for developing Web-Communication part python codes and solved TCP connection problems for the Pi-Communication. His contribution counts about 33.3%.

Shoujing Ke is responsible for developing Pi-Communication part python codes. His contribution counts about 33.3%.

1. **References:**
2. Flask tutorial:

<https://www.tutorialspoint.com/flask/flask_quick_guide.htm>

1. TA’s Help documents:

<https://docs.google.com/document/d/1WtB4MHnqvBntdf9GrMAJeu-163esEZZmeOkfD-muQOo/edit>

1. Debugging:

<https://stackoverflow.com>

<https://www.csdn.net>

1. **IDE Screenshots:**

Graphical user interface, application

Description automatically generated

A picture containing text

Description automatically generatedTable

Description automatically generatedTable

Description automatically generated with low confidenceGraphical user interface, application

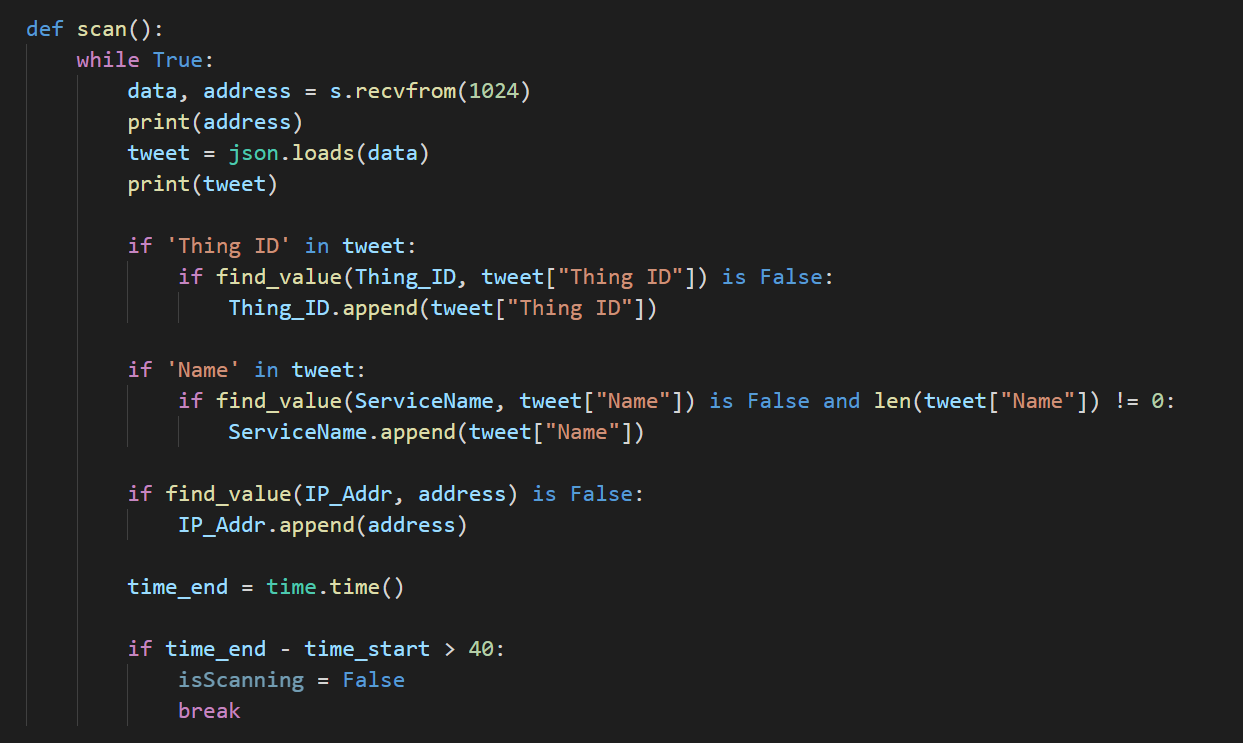
Description automatically generatedGraphical user interface, application

Description automatically generated

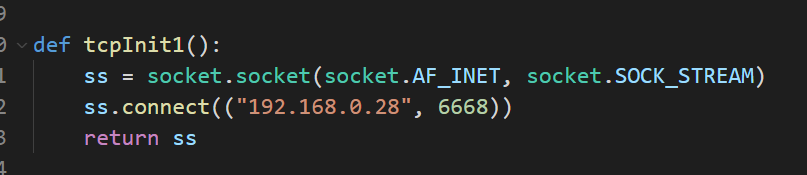
1. **Code Clarification:**
2. Pi-Communication part:



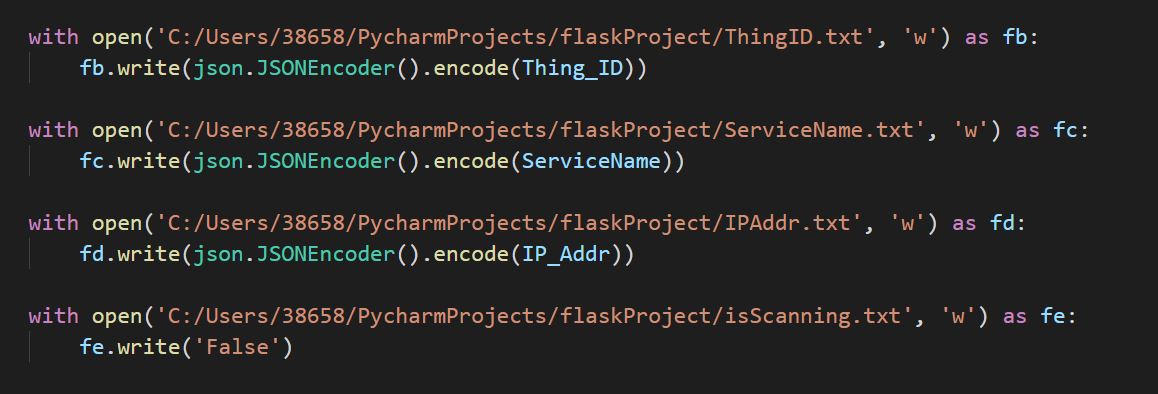
These codes are copied from the help document that the TA provides, used to receive tweets from the raspberry pi.



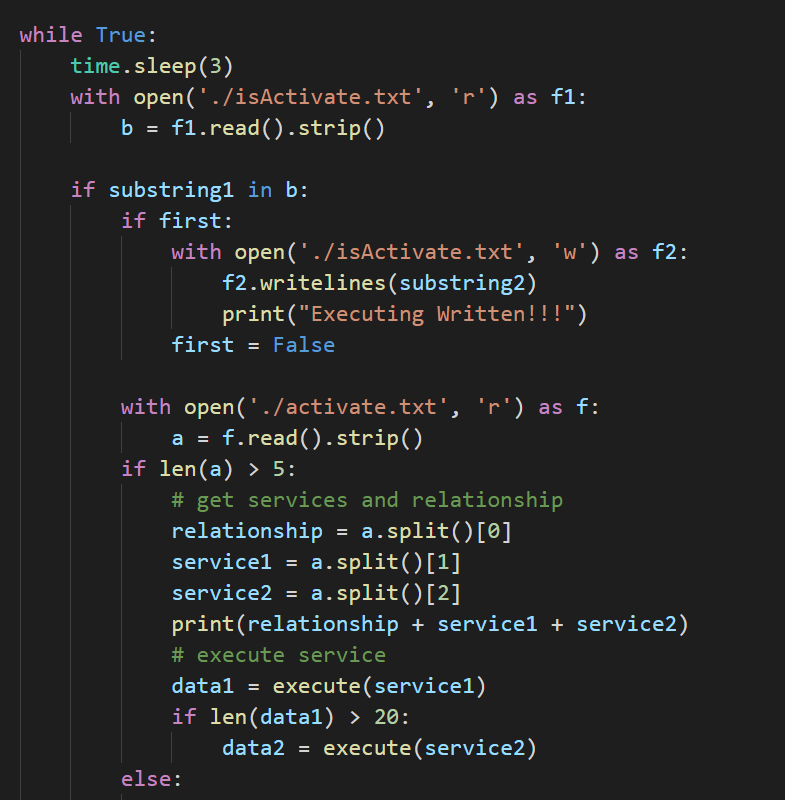
This part of code is responsible for extract attributes that will be used to filter services from the Pi’s broadcast tweets.



This function is used to establish TCP connections with the Pi. Connections need to be closed and re-establish every time the host wants to send packages to the Pi.



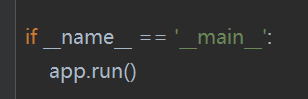
Writing data to those txt files so that the app.py can send those information about the VSS to the front end.



Scan those txt files in every 3 seconds to know the recipe and then execute.

1. Web-Communication part:

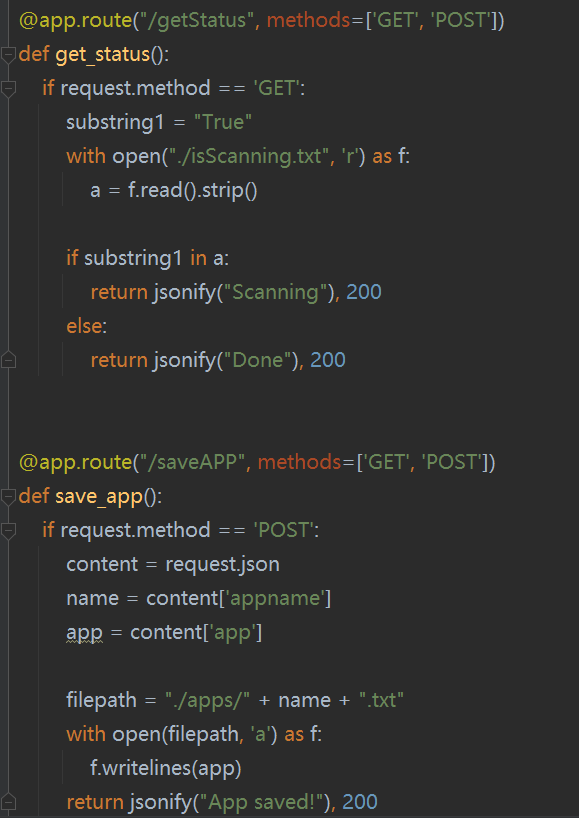




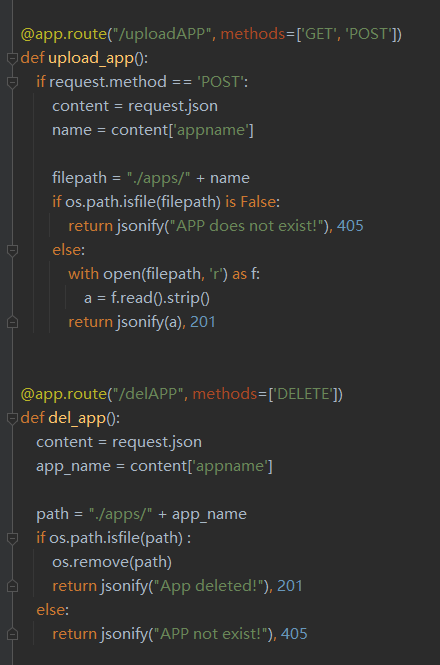
Use the Flask framework and run it to receive requests from the front-end.

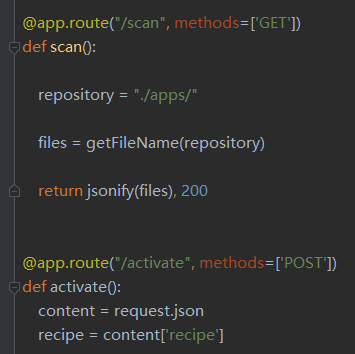


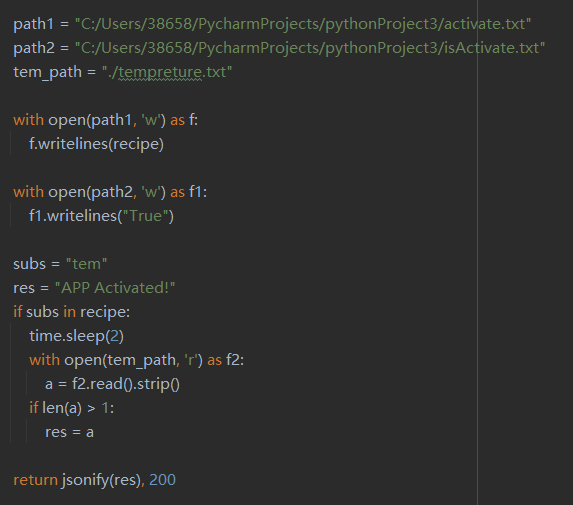
Codes for return ThingID, Pi’s IP addresses and services names to front-end



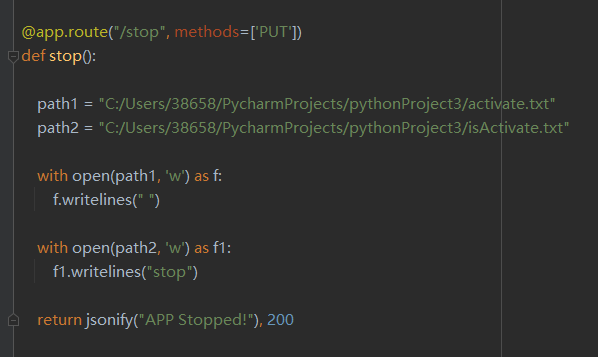
Codes for save APPs and get the scanning status.

  
codes for upload APPs and delete APPs.





Codes for scan the working directory and activate APPs



Codes for stopping running APPs.