**Student Task 5 Dynamic ZABBIX Maps for JumpingJIVE**

**Developer manual**

Written by Jingyao Su

Latest version updated on 27.03.2019

# 1. Introduction

## 1.1 Basics

The scripts are written in Python 3, consisting of 3 .py files:

‘dynamicmap.py’

‘login.py’

‘images.py’

Main functions shall be realized in ‘dynamicmap.py’. The other two work as modules.

## 1.2 Functions

The scripts are expected to realize following functions:

1. Get host information

2. Create images as background of maps

3. Create Wettzell station map and EVN global map

4. Determine station/antennas location on the map

5. Add or delete stations/antennas on the map

6. Update and delete maps

7. Possibility to switch on/off the antennas on the map

## 1.3 Command line argument

Except Python IDE, the scripts are expected to be called and run from Command Prompt by command line argument.

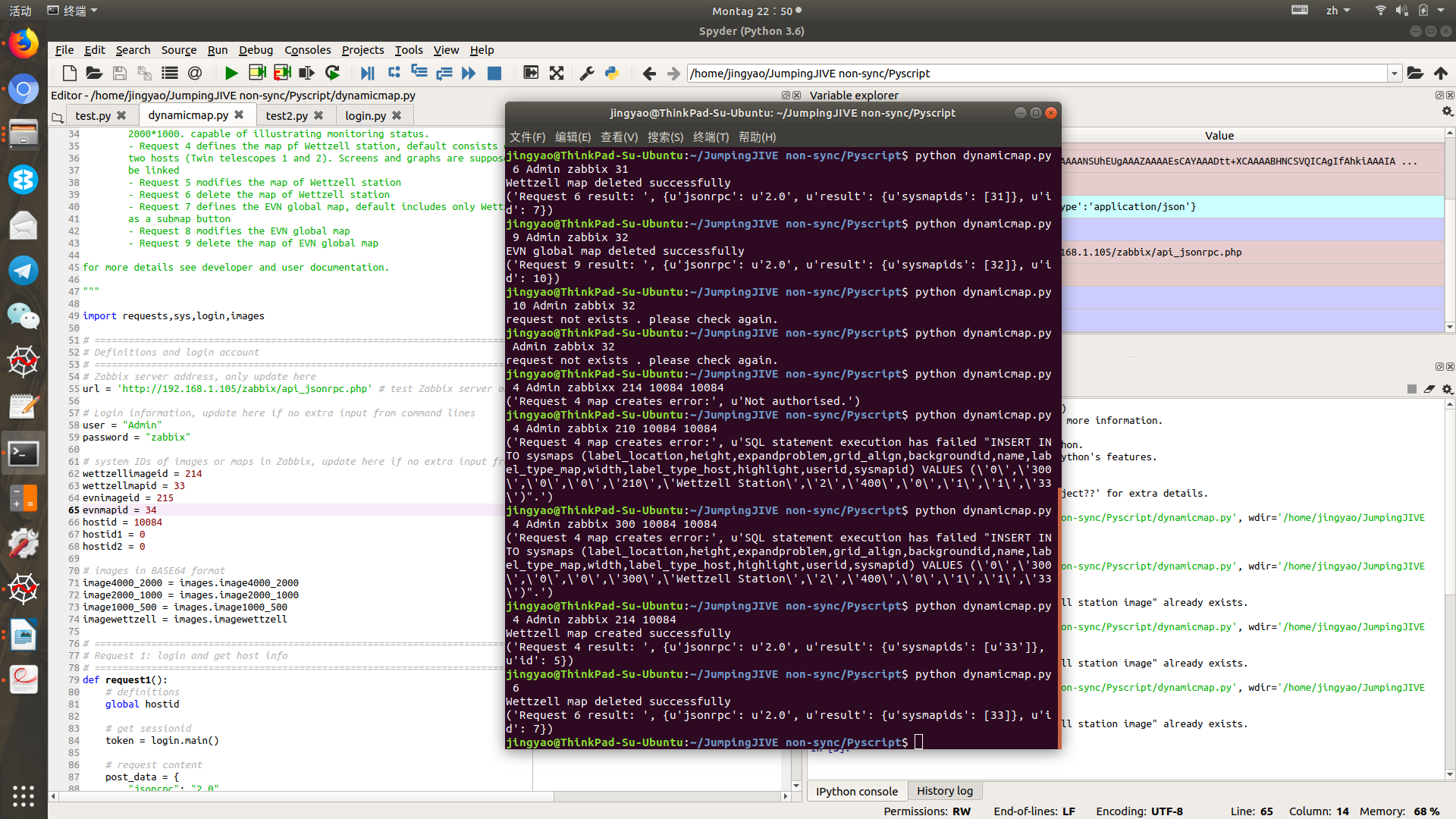


Fig.2 Command line argument in Ubuntu Terminal

## 1.4 Environment

Since Python 3 should be already installed in the Ubuntu Linux system, only few third-party Python packages (requests) are needed to be installed in extra.

To see which version of Python 3 you have installed, open a command prompt and run

$ python3 --version

If you are using Ubuntu 16.10 or newer, then you can easily install Python 3.6 with the following commands:

$ sudo apt-get update

$ sudo apt-get install python3.6

If you’re using another version of Ubuntu (e.g. the latest LTS release), to install Python 3.6:

$ sudo apt-get install software-properties-common

$ sudo add-apt-repository ppa:deadsnakes/ppa

$ sudo apt-get update

$ sudo apt-get install python3.6

Python 3.4 and later include pip by default. To see if pip is installed, open a command prompt and run

$ command -v pip

If no pip is installed, run

$ apt-get install pip

Then to install ‘requests’ package, run

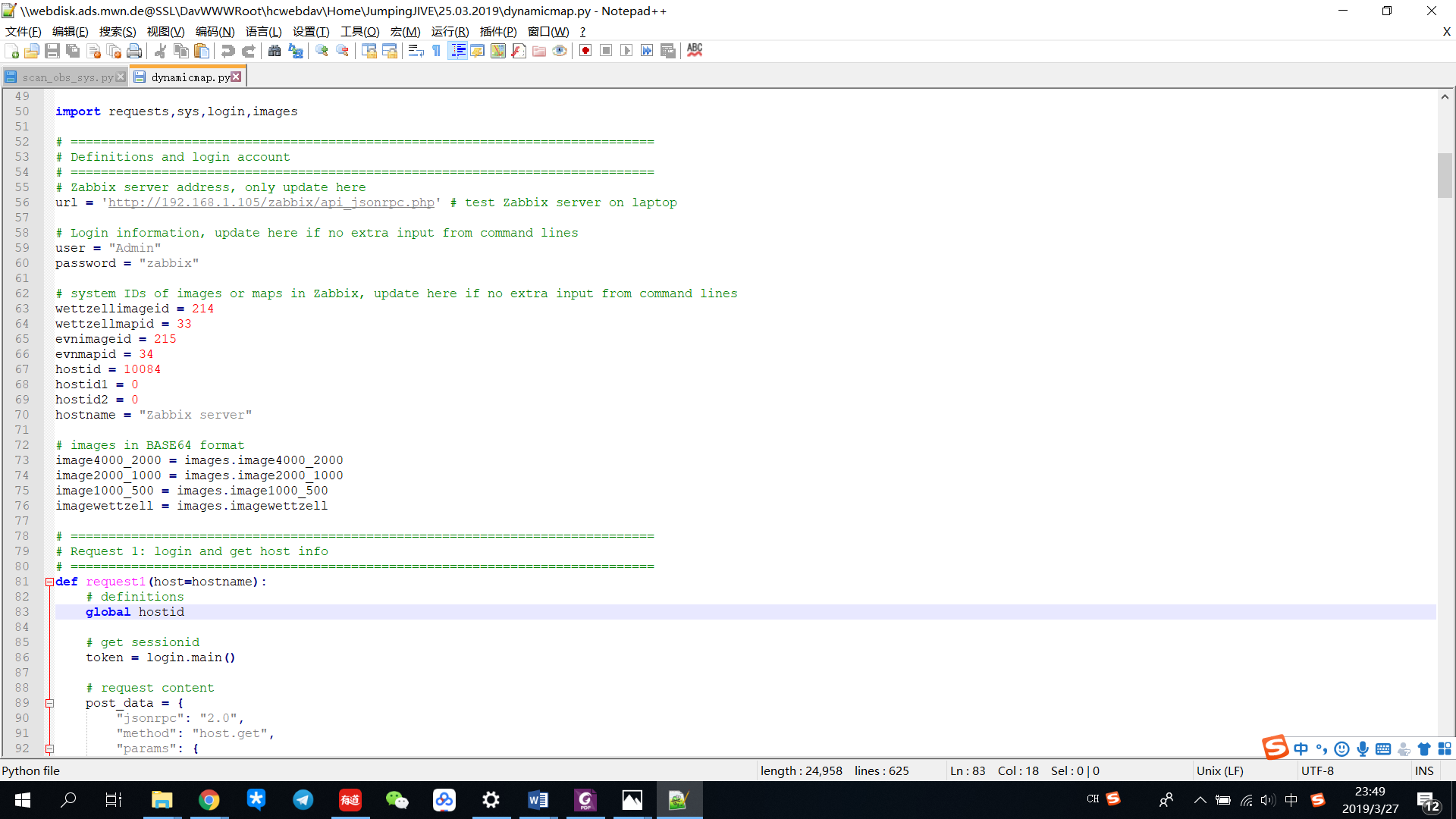
$ pip install requests

# 2. Script structure

## 2.1 ‘dynamicmap.py’

1. Definition and login account

Login account of Zabbix system and IP address of Zabbix server, as well as other definitions are expected to be updated in this part. In addition, these parameters except server address can also be determined from command line argument, which is expected to have higher priority, which means the values in this part should be overwritten by command line arguments. Where should be paid attention is the server address can only be defined here.



Definitions are:

‘url’: Zabbix server address

‘user’: User account of Zabbix system

‘password’: User password of Zabbix system

‘wettzellimageid’: background image ID for Wettzell station map

‘wettzellmapid’: map ID of Wettzell station in Zabbix system

‘evnimageid’: background image ID for EVN global map

‘evnmapid’: map ID of EVN global network in Zabbix system

‘hostid’: host ID of telescope/antenna to be monitored

‘hostid1’: host ID of twin telescope 1 at Wettzell station

‘hostid2’: host ID of twin telescope 2 at Wettzell station

‘hostname’: name of host whose information is expected to be obtained

Also the images are introduced in the formation of BASE64 from ‘images.py’.

‘image4000\_2000’: background image for EVN global map with size of 4000\*2000

‘image2000\_1000’: background image for EVN global map with size of 2000\*1000

‘image1000\_500’: background image for EVN global map with size of 1000\*500

‘imagewettzell’: background image for Wettzell station map

2. Request 1 to 9

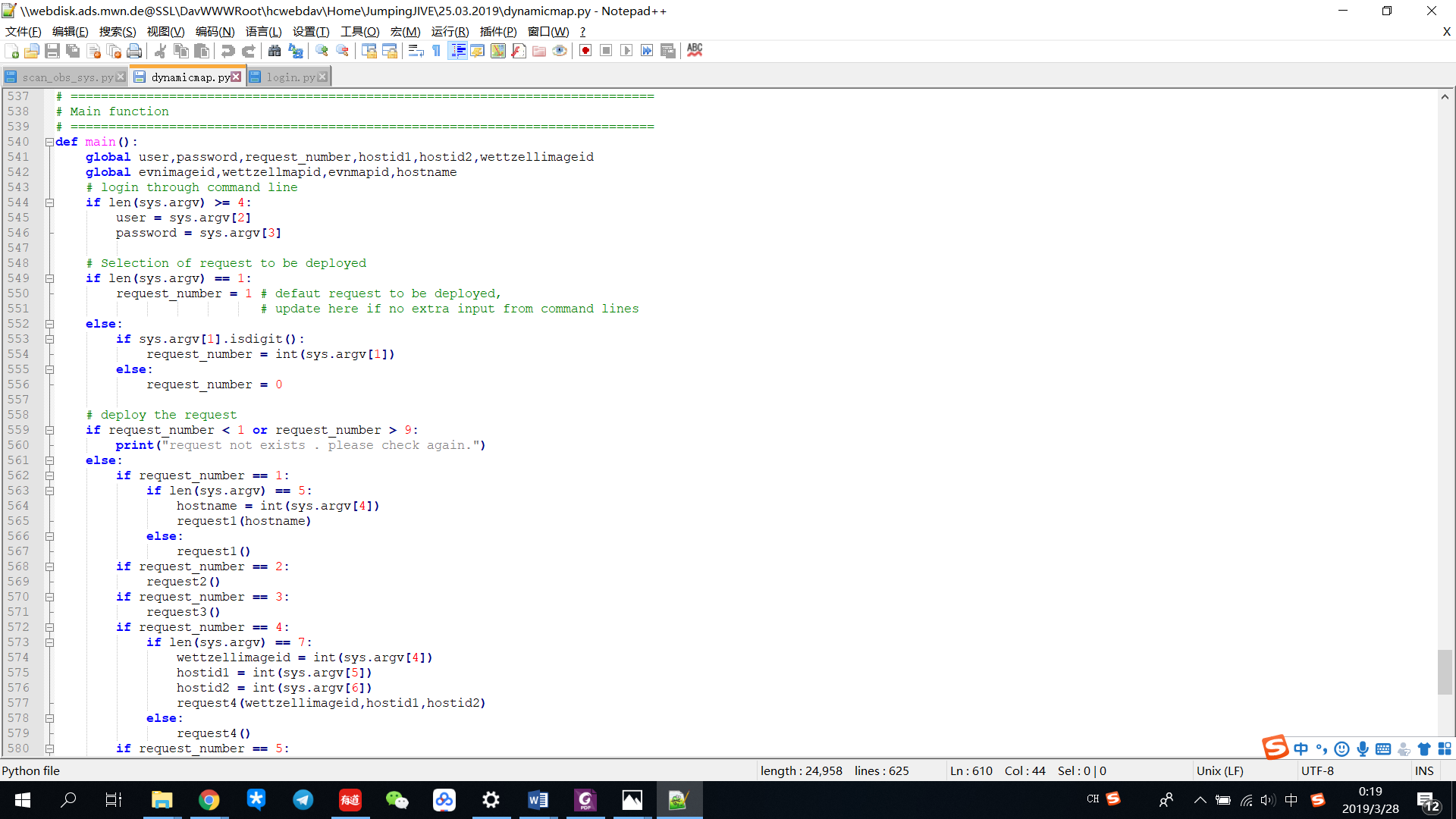
All of 9 requests are programmed as functions and thus are expected to run when it is called.

Details in Chapter 3 to 5 of this manual.

3. Main function

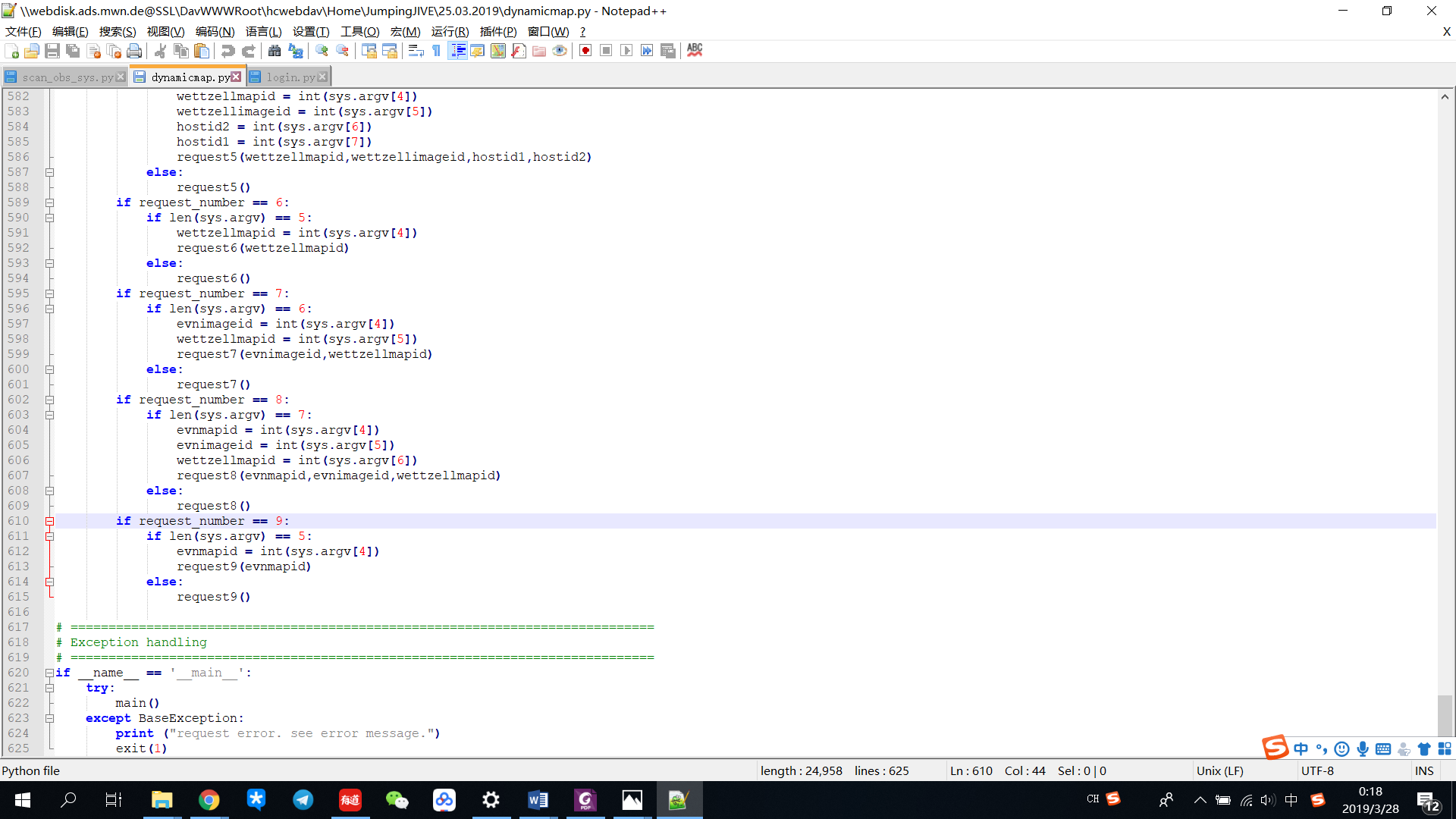
Command line arguments are expected to read in this part and overwrite default values.

After checking availability of input data, the requests should be run according to requirements, i.e. if there is argument, run it, while if no, run default.



4. Exception handling

This part is expected to deal with exceptions in case of the Python error handling with endless outputs of the whole call stack and line numbers.

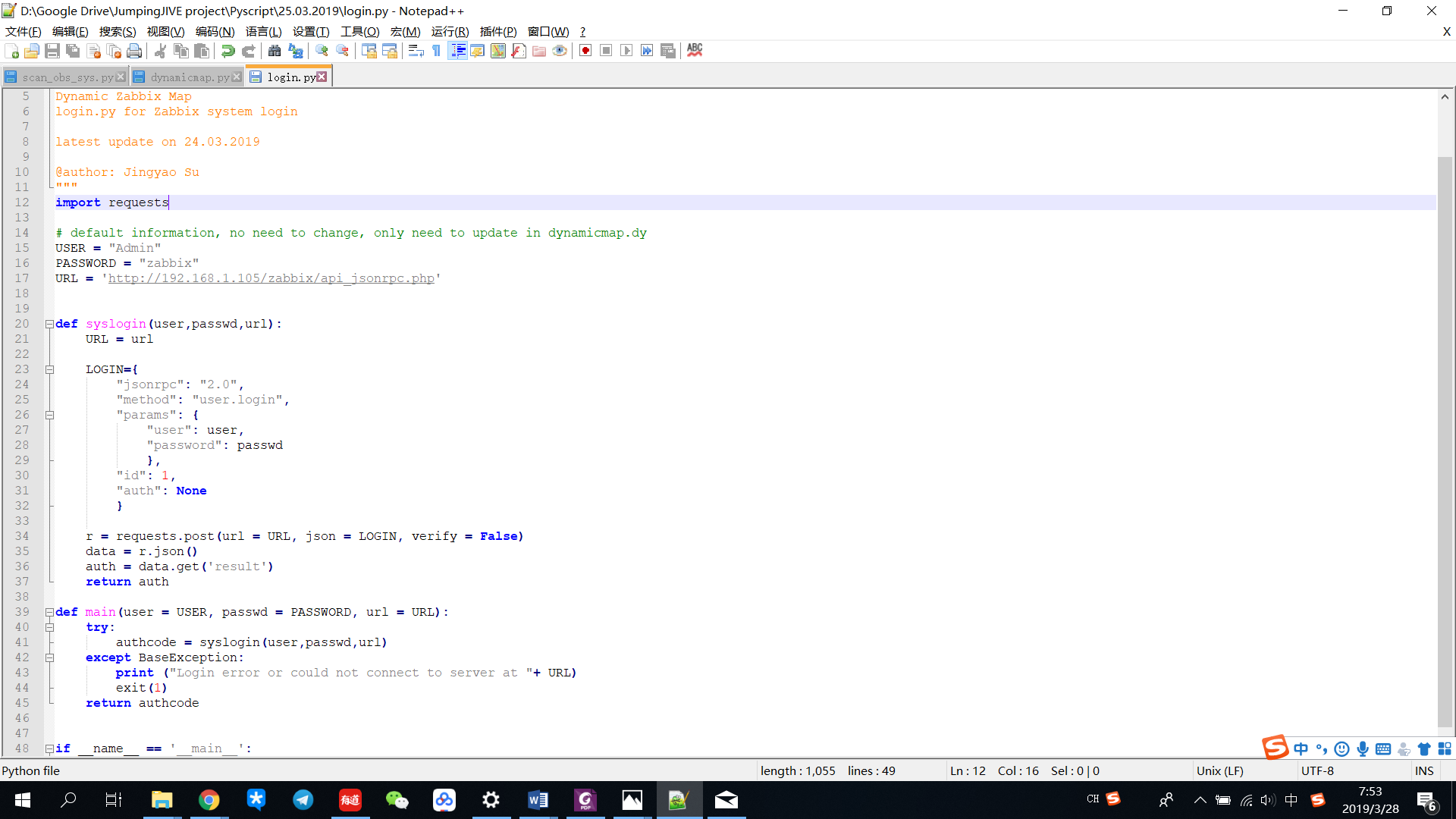


## 2.2 ‘login.py’

This file is expect to deal with system login. Whenever it is called with the input of user and password, it should return a token(auth). Otherwise without input the default account is set to be ‘Admin’ and ‘zabbix’ for user name and password respectively.

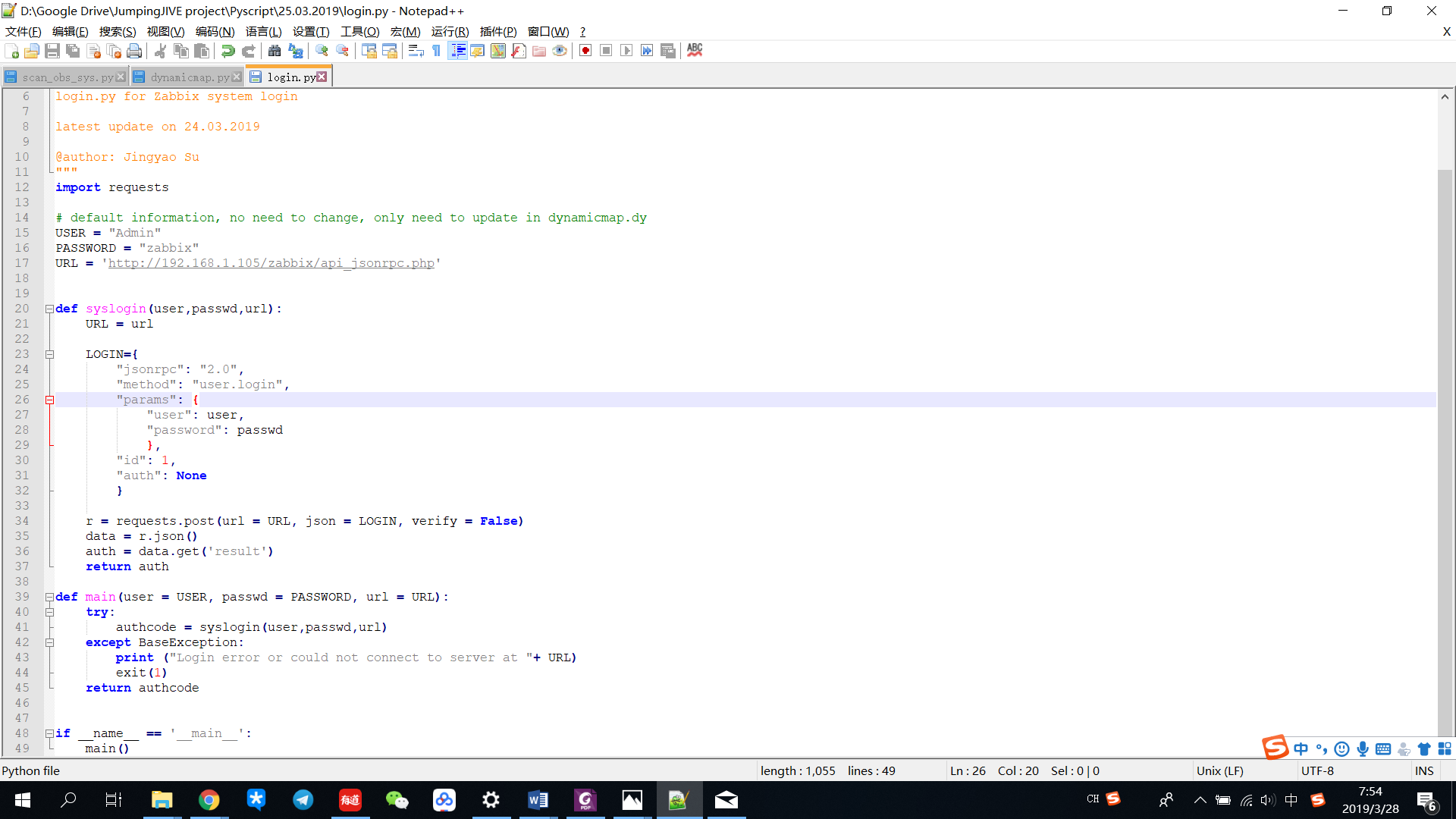
Meanwhile, user does not need to update anything (user account or server address) in this file, but only operate in ‘dynamicmap.py’.

1. system login



This part is expected to realize the login function. Input should be user name, password and URL of Zabbix server. These inputs are expected to be defined from ‘dynamicmap.py’ by arguments or default value.

2. Main function as exception handling

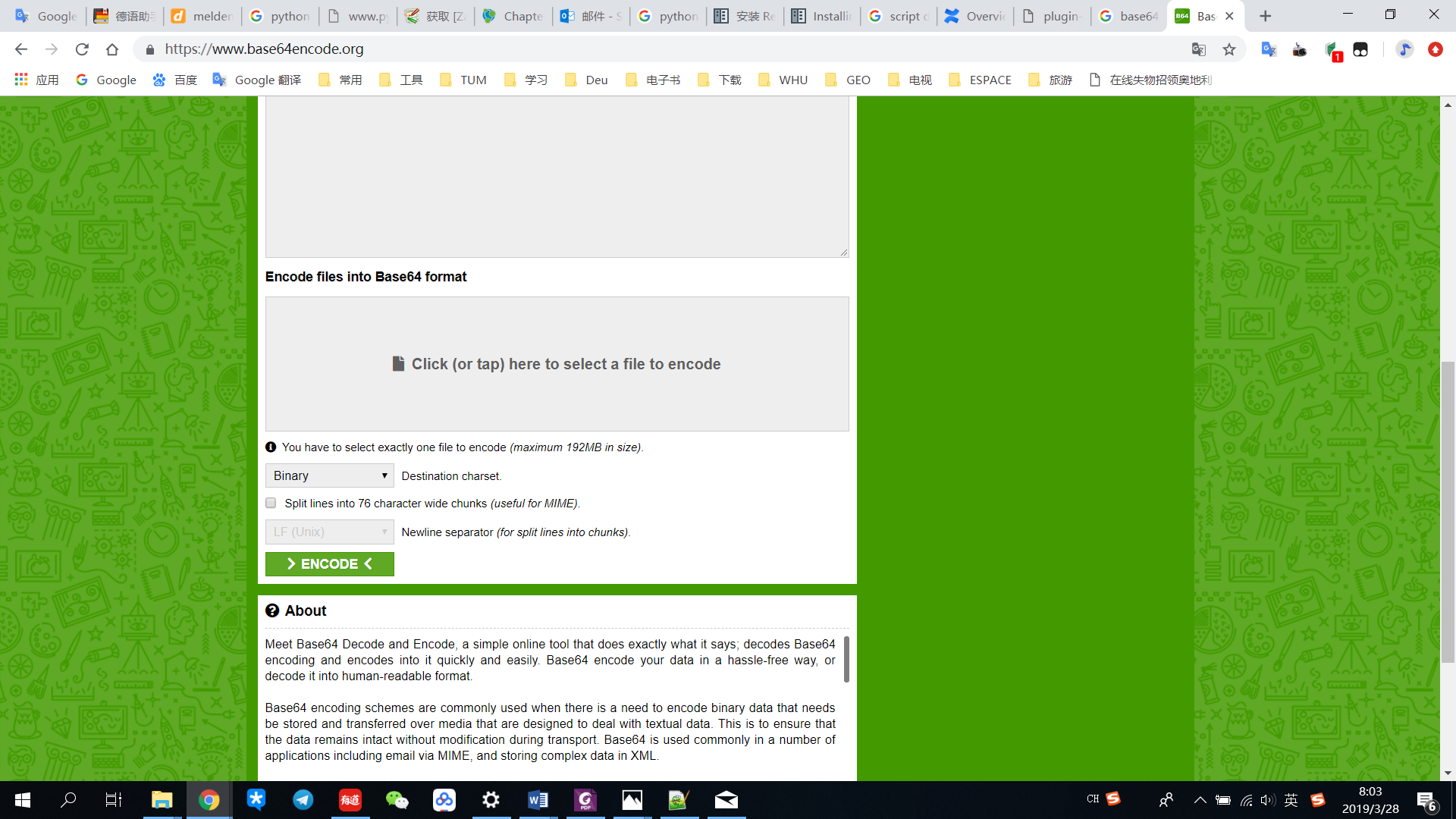


This part is expected to deal with exceptions in case of the Python error handling with endless outputs of the whole call stack and line numbers.

## 2.3 ‘images.py’

Images in formation of BASE64 are expected to be defined in this file and can be introduced to ‘dynamicmap.py’. Currently there are 4 images

The BASE64 code can be obtained by an online tool ‘BASE64 Decode and Encode’. The address is: https://www.base64encode.org/

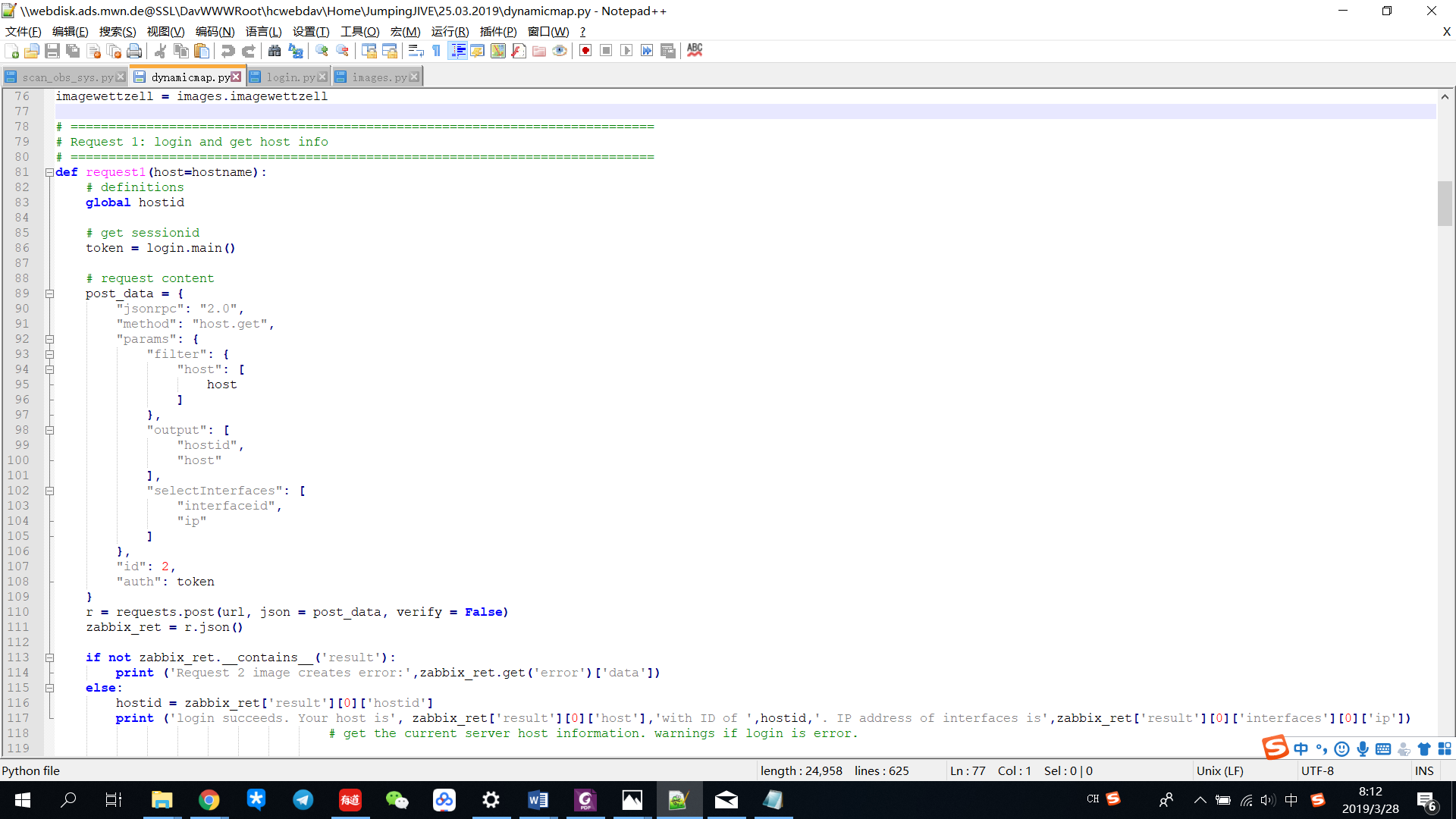


How to use it:

* Select an image file to upload
* Choose ‘Binary’ as destination charset
* Click ‘Encode’ and get a ‘Success!’ result
* Click ‘CLICK OR TAP HERE’ to download the code.

# 3. Get host info

This chapter deals with function of getting host information. Related part of script is ‘Request 1: login and get host info’ from line 78 to 118.



Input is expected to be the host to be monitored. Default is current server.

Firstly ‘login.py’ is called to get a login token, which is supposed to be referred for any request.

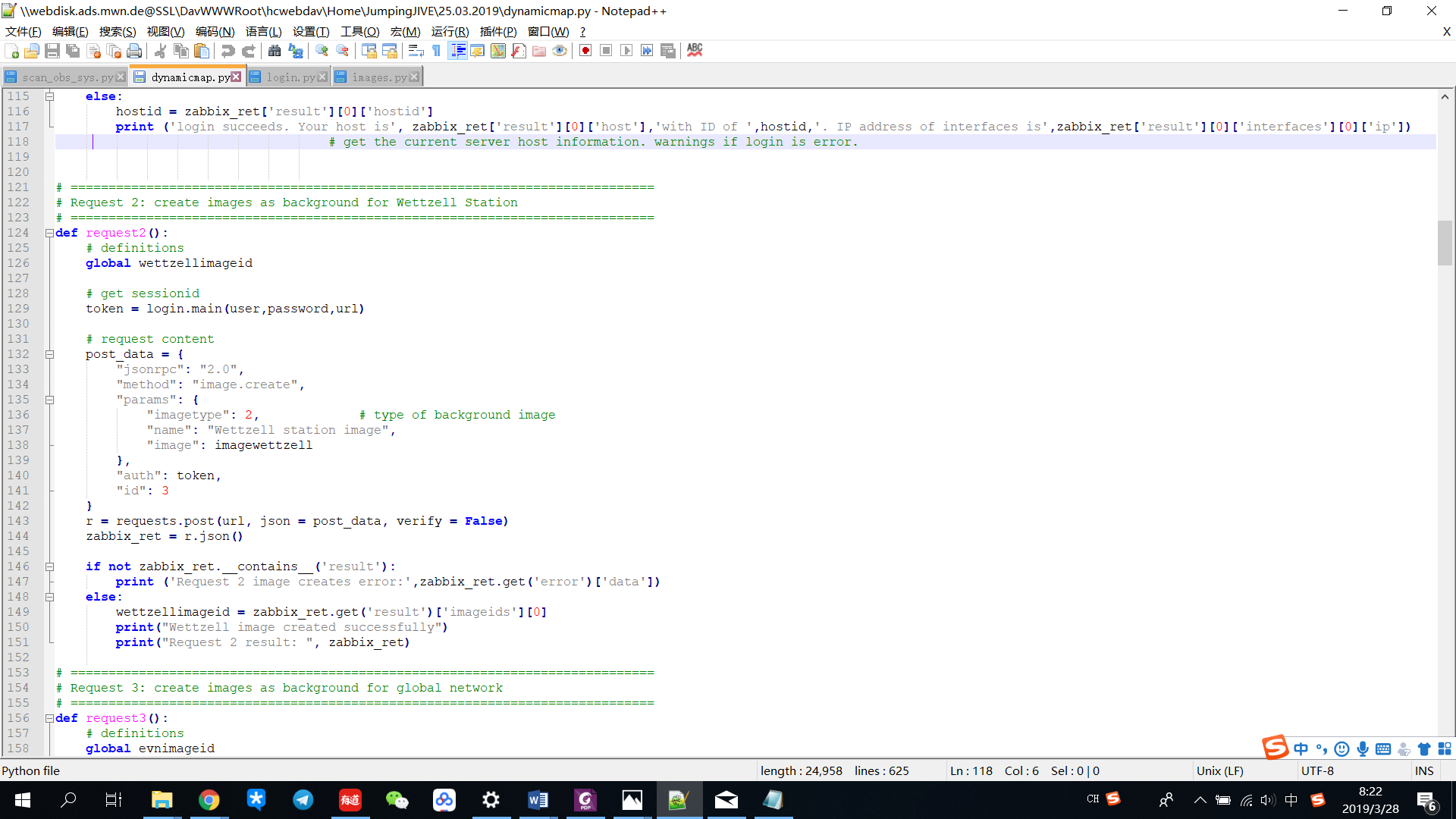
Variable of ‘post\_data’ defines the parameters of request.

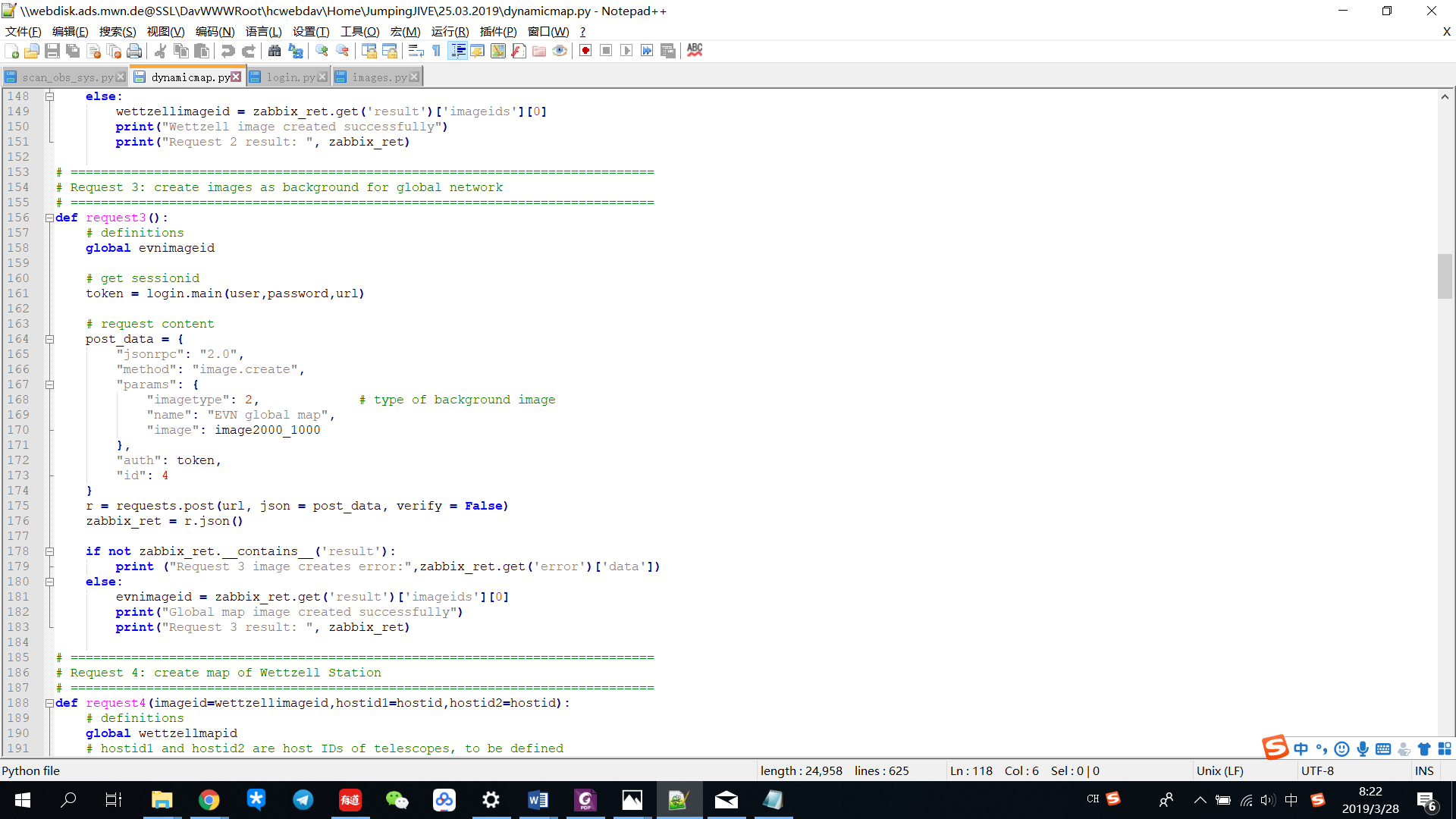
Here lies host name, which is supposed to be updated by user, ‘output’, which includes host ID and host name and shows in the request result, as well as IP address.

After checking if the request runs successfully or not, the result is expected to be printed.

# 4. Image creation

This chapter deals with the function of background image creation. Related part of script is ‘Request 2: create images as background for Wettzell Station’ from line 121 to 151 and ‘Request 3: create images as background for global network’ from line 153 to 183.





These two parts have same structure.

There is no input required when calling this function.

Firstly ‘login.py’ is called to get a login token, which is supposed to be referred for any request.

Variable of ‘post\_data’ defines the parameters of request.

Here lies ‘imagetype’, which defines the type of image in Zabbix system, here setting ‘2’ as background image, ‘name’, which defined the name of image, as well as ‘image’, which should be BASE64 code. Default image is background image for Wettzell station map or background image for global map with size of 2000\*1000.

After checking if the request runs successfully or not, the result is expected to be printed. Result is expected to include the ID of stored image, which is named as a variable of ‘wettzellimageid’ or ‘evnimageid’. Otherwise the result show error message.

# 5. Map creation, updating and deletion

This chapter deals with the function of map creation, updating and deletion. Related part of script is ‘Request 4: create map of Wettzell Station’ from line 185 to 269, ‘Request 5: update map of Wettzell Station’ from line 271 to 351, ‘Request 6: delete map of Wettzell station’ from line 353 to 377, ‘Request 7: create map of EVN global network’ from line 379 to 444, ‘Request 8: update map of EVN global network’ from 446 to 508 and ‘Request 9: delete map of EVN global networks’ from line 510 to 535.

Basic structure is as follows:

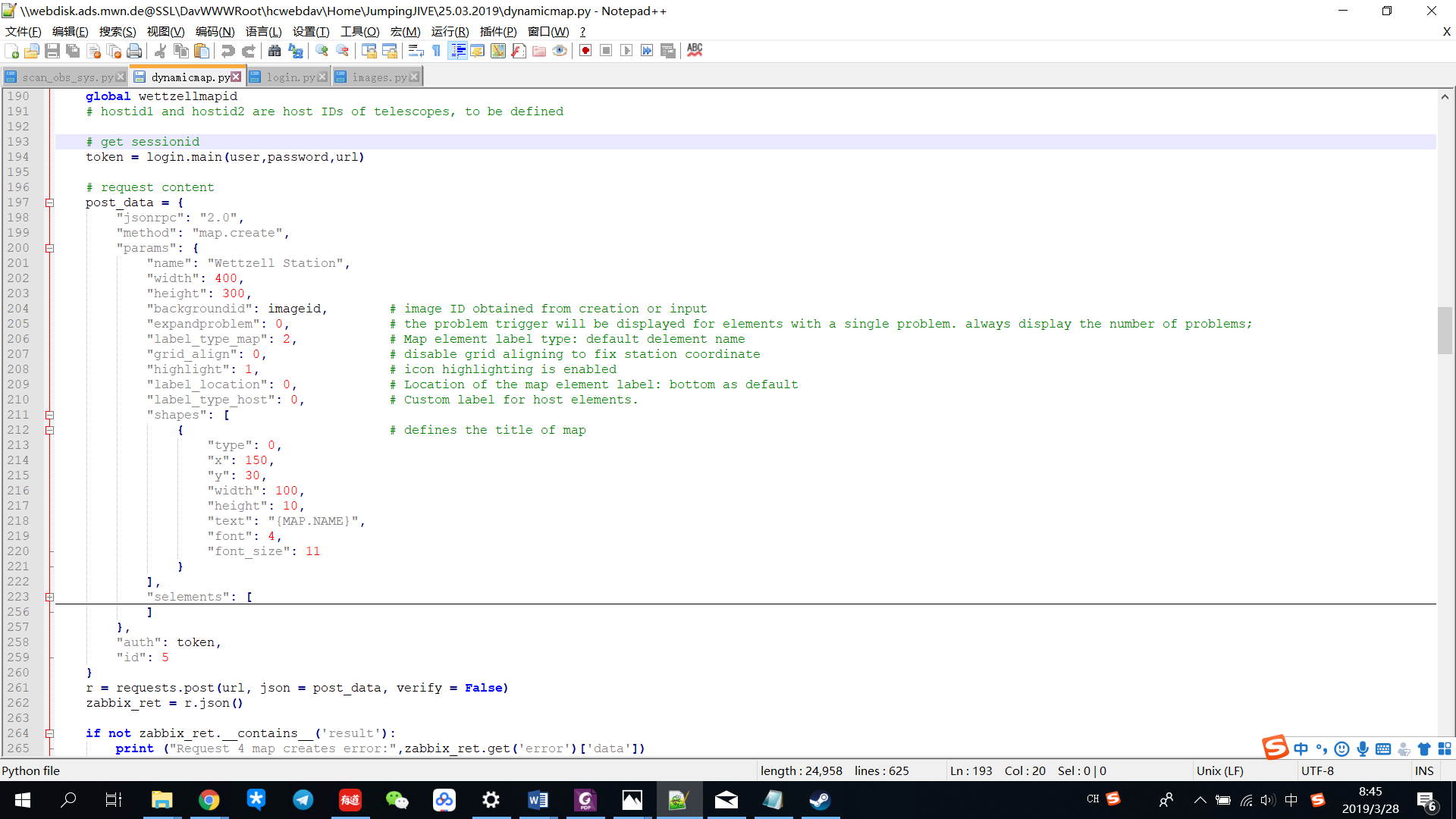
Firstly is input setting.

* For map creation, inputs required should be background image ID and IDs of the map elements, which are supposed to be ‘hostid’ for host(antenna/telescope) and ‘sysmapids’ for submap (station with separate map, e.g. Wettzell Station) when calling this function.
* For map updating, inputs required has additional ID of current map, i.e. ‘wettzellmapid’ for Wettzell Station map and ‘envmapid’ for EVN global map.
* For map deletion, input required is supposed to be only the ID of current map.

Then ‘login.py’ is called to get a login token, which is supposed to be referred for any request.

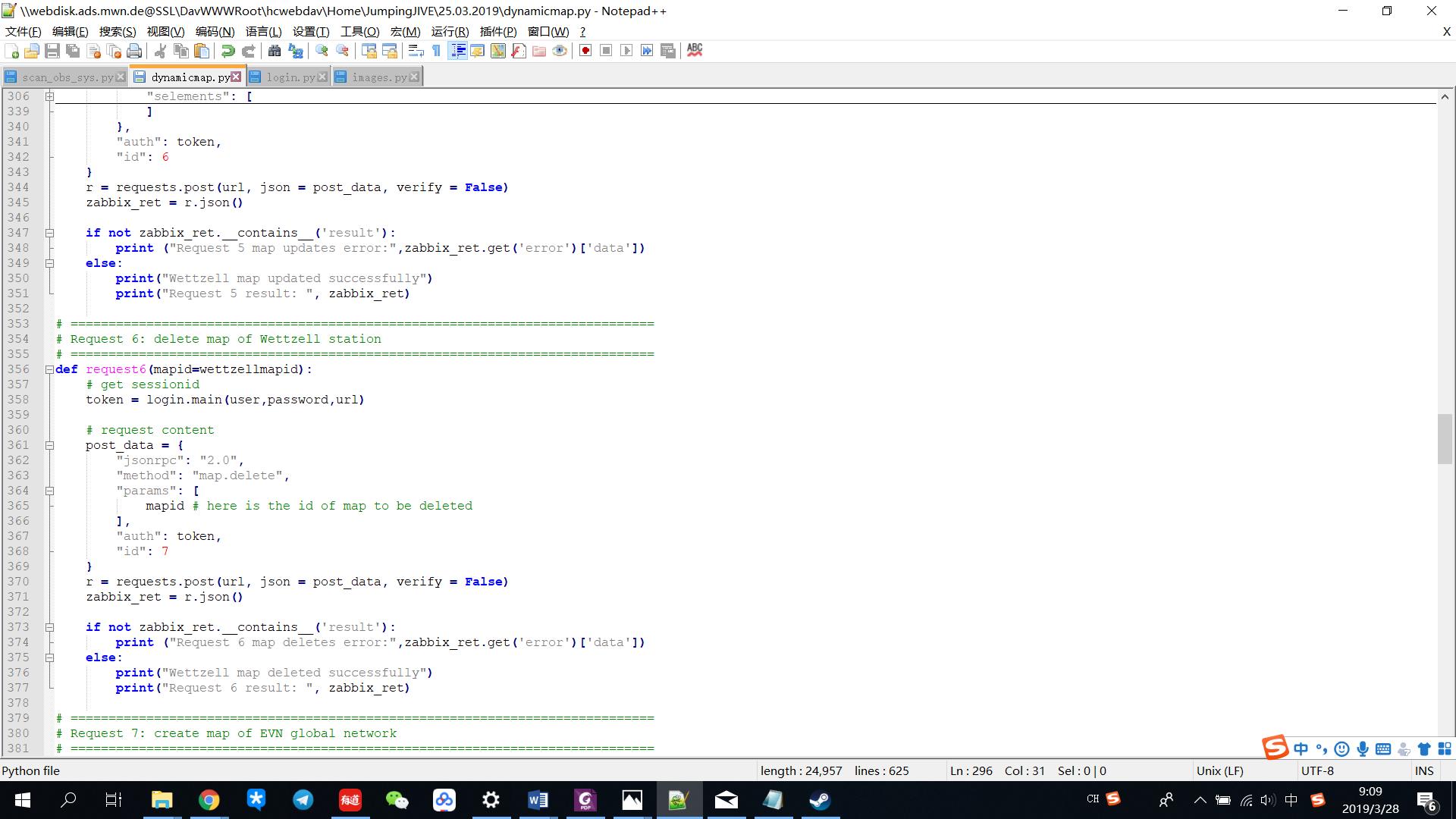
Variable of ‘post\_data’ defines the parameters of request.

For map creation and map updating, ‘param’ defines all the properties of map to be created or to be updated. Map updating has an additional parameter of ‘sysmapid’.



* "sysmapid": current map ID, i.e.’wettzellmapid’ or ‘evnmapid’. This ID is obtained from its creation result.
* "name": "Wettzell Station": name of map
* "width": width of map
* "height": height of map
* "backgroundid": defined from variable ‘imageid’, background image ID
* "expandproblem": the problem trigger will be displayed for elements with a single problem. ‘0’ means always displaying the number of problems
* "label\_type\_map": Map element label type: ‘2’ means default is element name
* "grid\_align": ‘0’ means disable grid aligning to fix station coordinate
* "highlight": ‘1’ means icon highlighting is enabled
* "label\_location": Location of the map element label: ‘0’ means bottom as default
* "label\_type\_host": Custom label for host elements.
* "shapes": defines the title of map
* "selements": defines the map elements, detailed description in next part

For map deletion, ‘param’ only includes ID of map to be deleted.



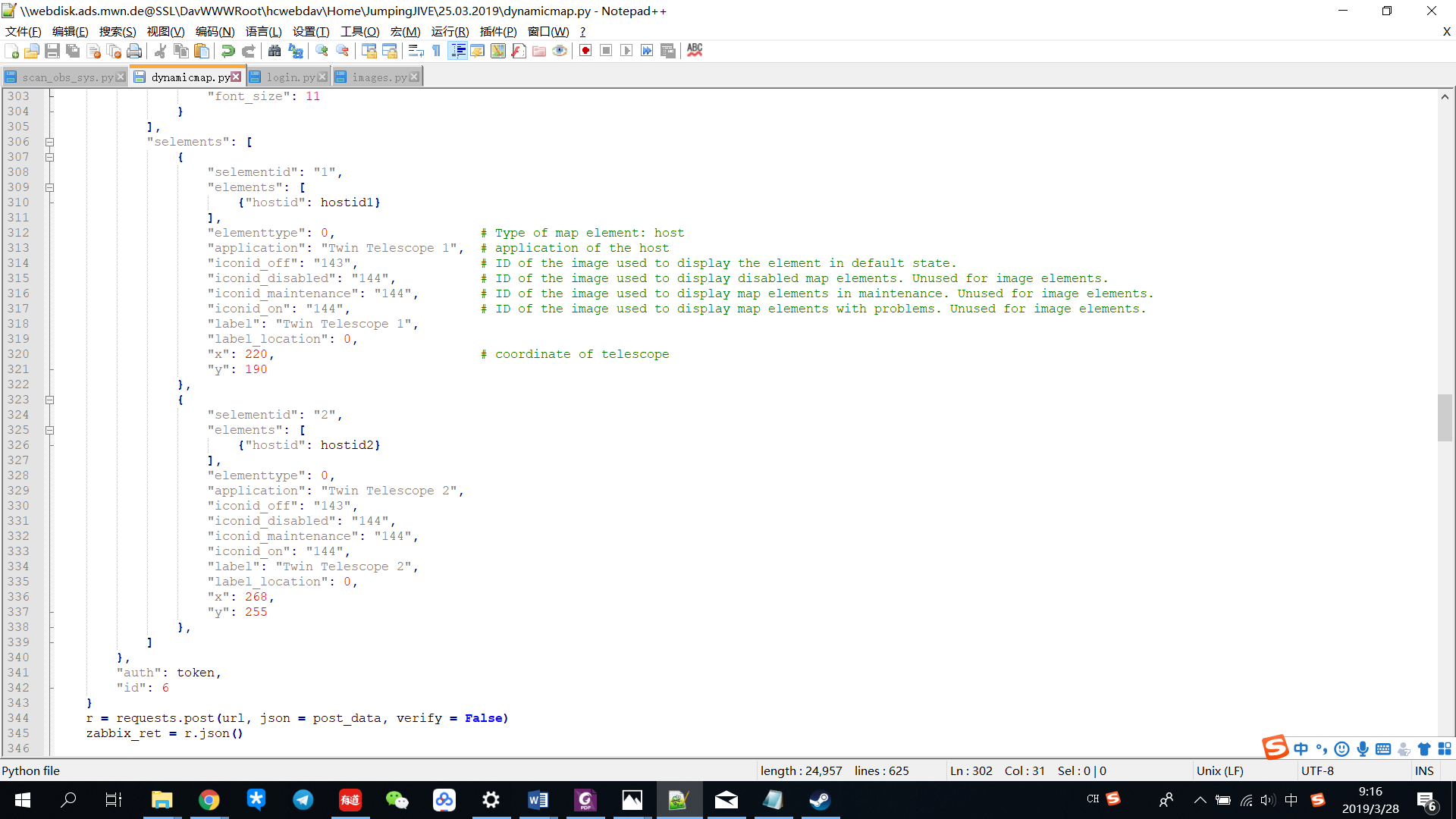
After checking if the request runs successfully or not, the result is expected to be printed. Result is expected to include the ID of stored image, which is named as a variable of ‘wettzellimageid’ or ‘evnimageid’. Otherwise the result show error message.

Important is the map element definition, which determines what is going to show on the map and thus it is possible to switch an antenna on or off.

Basic idea is simply adding or deleting certain lines when an antenna/telescope/station is expected to switch on or off. Following is the parameters of map elements:

* "selementid": ID of map element
* "elements": map element, host or submap, value should be host ID or map ID, such as {"hostid": hostid1} or {"sysmapid": submapid}
* "elementtype": Type of map element: ‘0’ as host, ‘1’ as submap
* "application": application of the host
* "iconid\_off": ID of the image used to display the element in default state.
* "iconid\_disabled": ID of the image used to display disabled map elements. Unused for image elements.
* "iconid\_maintenance": ID of the image used to display map elements in maintenance. Unused for image elements.
* "iconid\_on": ID of the image used to display map elements with problems. Unused for image elements.
* "label": label of element
* "label\_location": location of label
* "x", "y": coordinate of antenna button on the map

Following is code for an element type of ‘host’, which represents antenna/telescope, from Request 5:



Following is code for an element type of ‘submap’, which represents station with separate map, from Request 8:

