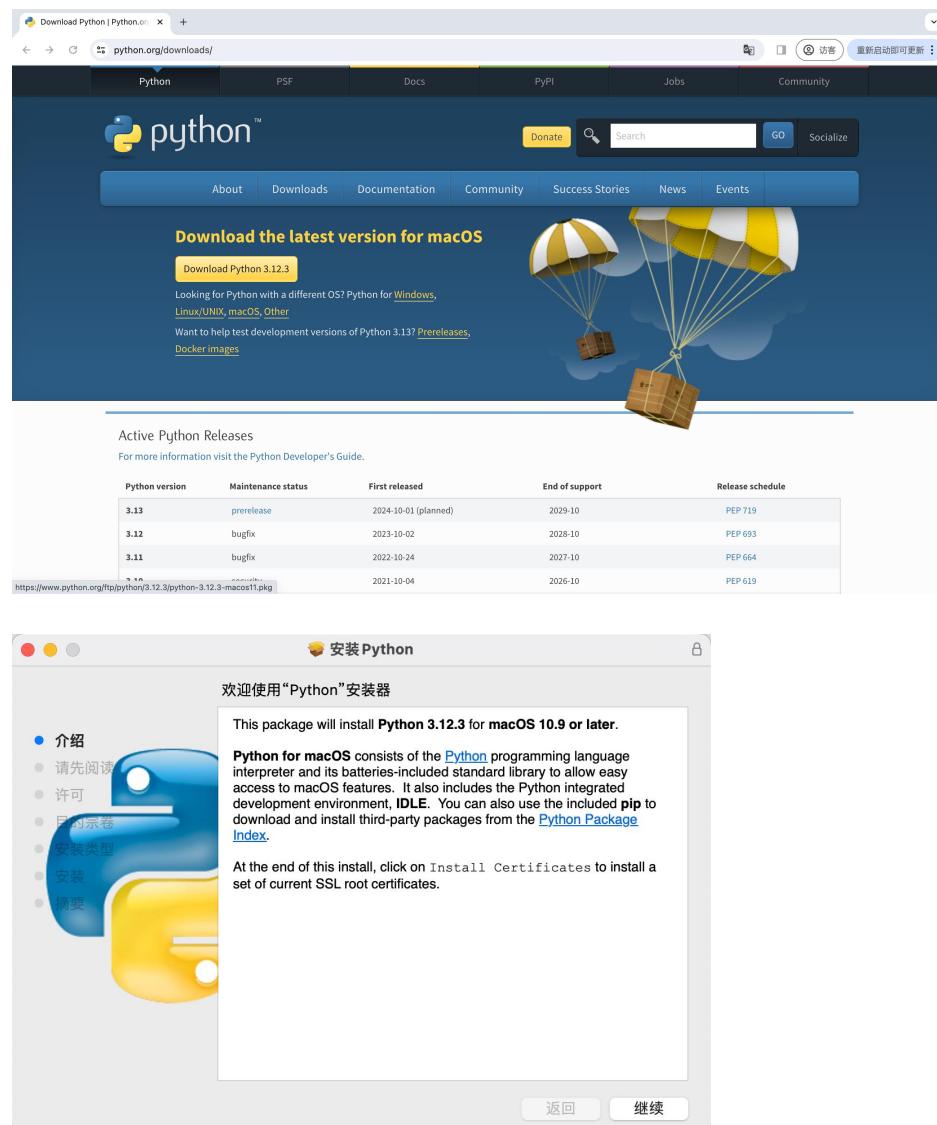


Installation(all base on Macbook M1):

1. python

Download from website then install following the guidelines.

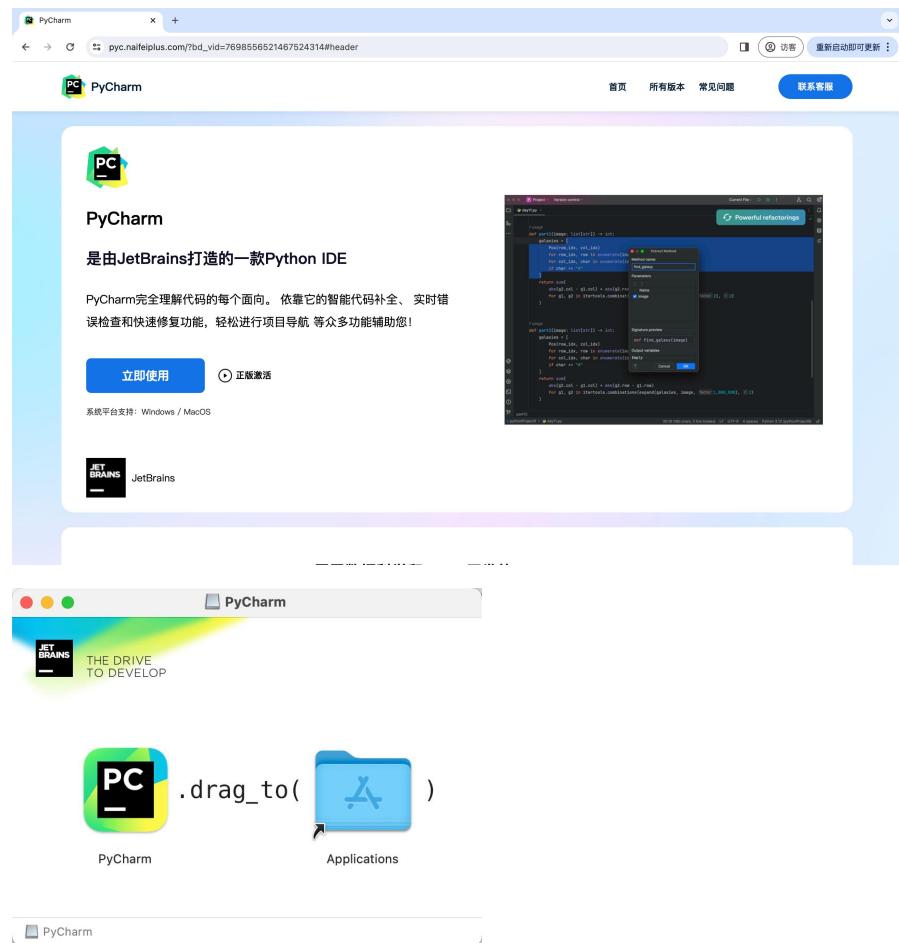


Can check python version by 'python --version' after installation in terminal.

```
silenc@shailengdeMacBook-Pro ~ % python3 --version
Python 3.12.3
silenc@shailengdeMacBook-Pro ~ %
```

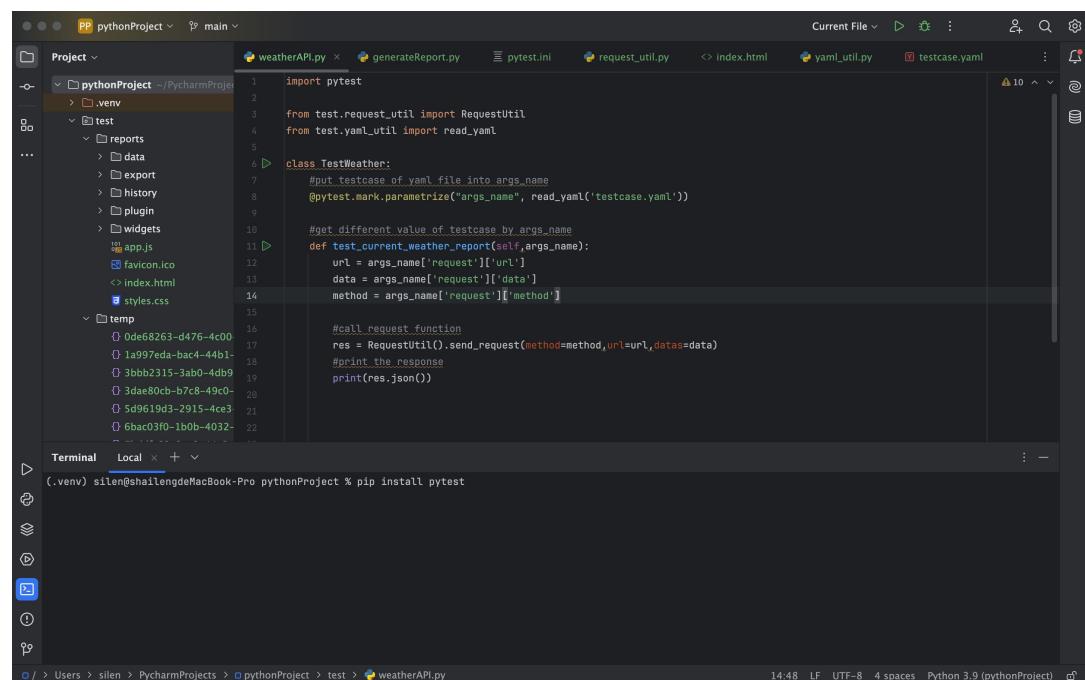
2. pycharm

Download from website then install following the guidelines.



3. pytest

Can install pytest by 'pip install pytest' in terminal of pycharm.



Can check pytest version by ‘pytest --version’ after installation.

```
import pytest
from test.request_util import RequestUtil
from test.yaml_util import read_yaml

class TestWeather:
    #put testcase of yaml file into args_name
    @pytest.mark.parametrize("args_name", read_yaml('testcase.yaml'))
    #get different value of testcase by args_name
    def test_current_weather_report(self,args_name):
        url = args_name['request']['url']
        data = args_name['request']['data']
        method = args_name['request']['method']

        #call request function
        res = RequestUtil().send_request(method=method,url=url,data=data)
        #print the response
        print(res.json())
```

```
(.venv) silen@shailengdeMacBook-Pro pythonProject % pytest --version
pytest 8.1.1
(.venv) silen@shailengdeMacBook-Pro pythonProject %
```

4. requests

Can install requests by ‘pip install requests’ in terminal of pycharm.

```
import pytest
from test.request_util import RequestUtil
from test.yaml_util import read_yaml

class TestWeather:
    #put testcase of yaml file into args_name
    @pytest.mark.parametrize("args_name", read_yaml('testcase.yaml'))
    #get different value of testcase by args_name
    def test_current_weather_report(self,args_name):
        url = args_name['request']['url']
        data = args_name['request']['data']
        method = args_name['request']['method']

        #call request function
        res = RequestUtil().send_request(method=method,url=url,data=data)
        #print the response
        print(res.json())
```

```
(.venv) silen@shailengdeMacBook-Pro pythonProject % pip install requests
```

Can check requests version by ‘pip show requests’ after installation.

The screenshot shows the PyCharm IDE interface. On the left is the project tree for 'pythonProject'. In the center is the code editor with 'weatherAPI.py' open, containing Python test code. On the right is the terminal window. The terminal output shows the command 'pip show requests' being run, and the response displays the package's name, version, summary, author, email, license, location, requirements, and required-by information. The status bar at the bottom indicates the terminal is in 'Local' mode, the path is '/Users/silen/PycharmProjects/pythonProject/test/weatherAPI.py', and the environment is Python 3.9 (pythonProject).

```
(.venv) silen@shailengdeMacBook-Pro pythonProject % pip show requests
Name: requests
Version: 2.31.0
Summary: Python HTTP for Humans.
Home-page: https://requests.readthedocs.io
Author: Kenneth Reitz
Author-email: me@kennethreitz.org
License: Apache 2.0
Location: /Users/silen/PycharmProjects/pythonProject/.venv/lib/python3.9/site-packages
Requires: certifi, charset-normalizer, idna, urllib3
Required-by:
```

5. homebrew

Can install homebrew by following command in terminal one by one, when see the green info means homebrew installed successfully.

```
echo export PATH=/opt/homebrew/bin:$PATH >> ~/.zshrc
```

```
source ~/.zshrc
```

```
HOMEBREW_CORE_GIT_REMOTE=https://mirrors.ustc.edu.cn/homebrew-core.git
```

```
/bin/bash -c "$(curl -fsSL https://cdn.jsdelivr.net/gh/ineo6/homebrew-install/install.sh)"
```

The screenshot shows a terminal window titled 'silen — zsh — 80x24'. It displays the output of the homebrew installation script. The process includes cloning the Homebrew repository, pulling a merge request, downloading the portable Ruby bottle, and finally installing it. A message at the end congratulates the user on a successful installation and provides links for further help and rewards.

```
remote: Total 2793 (delta 0), reused 0 (delta 0), pack-reused 2793
Receiving objects: 100% (2793/2793), 776.59 KiB | 3.33 MiB/s, done.
Resolving deltas: 100% (1325/1325), done.
From https://mirrors.ustc.edu.cn/homebrew-services
 * [new branch]      master      -> origin/master
HEAD is now at a6fcf4f Merge pull request #643 from Homebrew/sync-triage-config
==> Downloading https://mirrors.ustc.edu.cn/homebrew-bottles/bottles/portable-ruby/portable-ruby-3.1.4.arm64_big_sur.bottle.tar.gz
#####
==> Pouring portable-ruby-3.1.4.arm64_big_sur.bottle.tar.gz
==> Next steps:
自动配置环境变量
执行成功

重要信息 !!!重要信息 !!!重要信息 !!!
如果遇到 command not found brew, 请执行下面脚本完成安装或者直接重新打开终端:
eval "$( /opt/homebrew/bin/brew shellenv )"

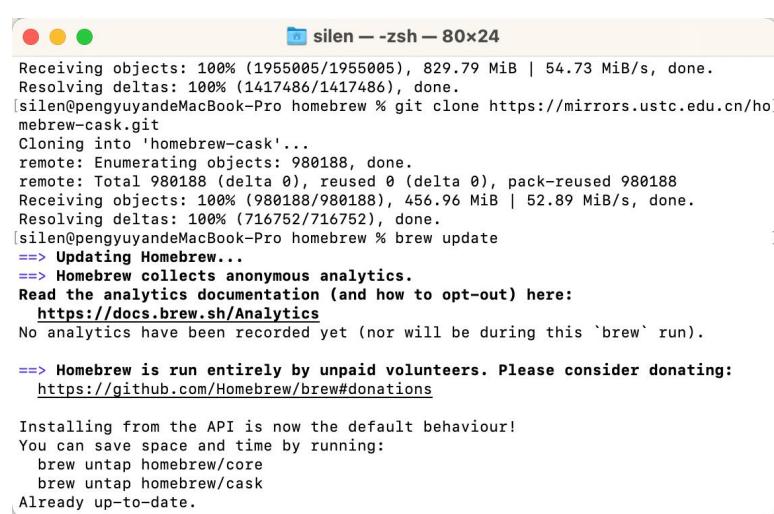
==> 🎉 恭喜, 安装成功!
- 验证命令: brew help
- 请收藏, 谨防失联: https://brew.idayer.com

==> 如果有幸帮助到你, 可以考虑请我喝杯咖啡~
☕ 喝咖啡: https://brew.idayer.com/reward/
```

Then can update homebrew by following command in terminal one by one.

```
cd "$(brew --repo)"  
git clone https://mirrors.ustc.edu.cn/brew.git  
  
cd "$(brew --repo)/Library/Taps/homebrew"  
  
git clone https://mirrors.ustc.edu.cn/homebrew-core.git  
  
git clone https://mirrors.ustc.edu.cn/homebrew-cask.git
```

brew update

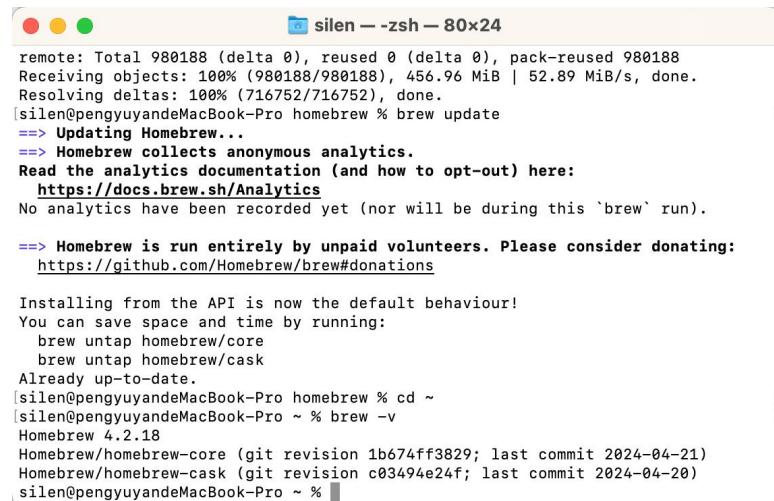


```
silen -- zsh -- 80x24  
Receiving objects: 100% (1955005/1955005), 829.79 MiB | 54.73 MiB/s, done.  
Resolving deltas: 100% (1417486/1417486), done.  
[silen@penguyandeMacBook-Pro homebrew % git clone https://mirrors.ustc.edu.cn/ho  
mebrew-cask.git  
Cloning into 'homebrew-cask'...  
remote: Enumerating objects: 980188, done.  
remote: Total 980188 (delta 0), reused 0 (delta 0), pack-reused 980188  
Receiving objects: 100% (980188/980188), 456.96 MiB | 52.89 MiB/s, done.  
Resolving deltas: 100% (716752/716752), done.  
[silen@penguyandeMacBook-Pro homebrew % brew update  
==> Updating Homebrew...  
==> Homebrew collects anonymous analytics.  
Read the analytics documentation (and how to opt-out) here:  
  https://docs.brew.sh/Analytics  
No analytics have been recorded yet (nor will be during this `brew` run).  
  
==> Homebrew is run entirely by unpaid volunteers. Please consider donating:  
  https://github.com/Homebrew/brew#donations  
  
Installing from the API is now the default behaviour!  
You can save space and time by running:  
  brew untap homebrew/core  
  brew untap homebrew/cask  
Already up-to-date.
```

After updating, check installationg of homebrew by following command in terminal one by one.

cd ~

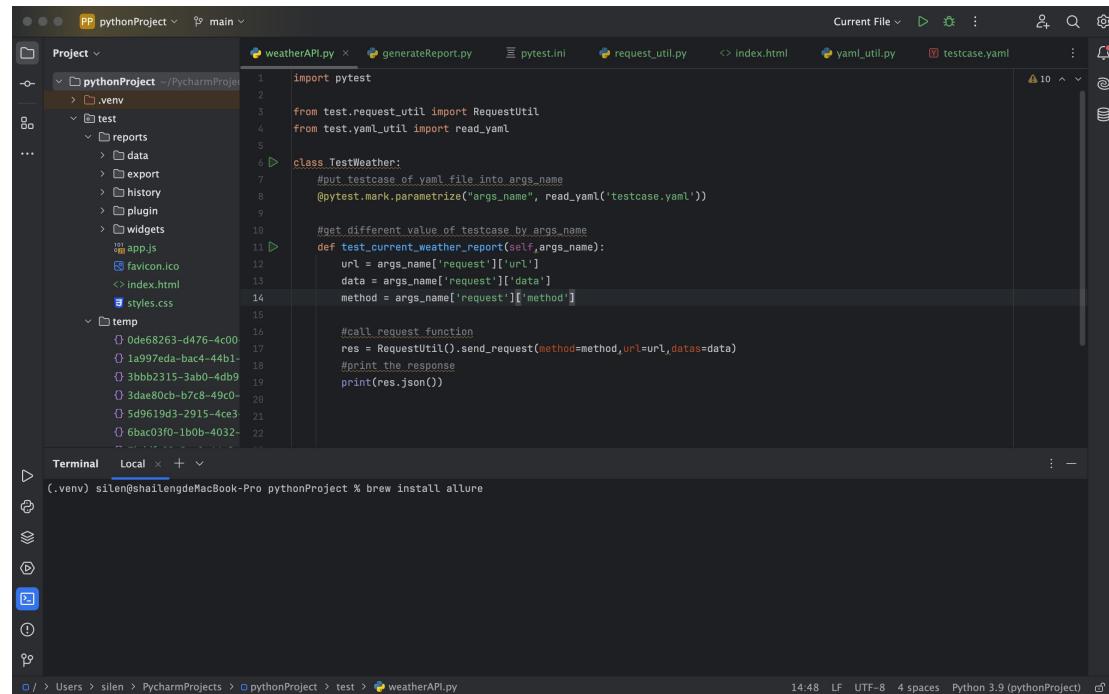
brew -v



```
silen -- zsh -- 80x24  
remote: Total 980188 (delta 0), reused 0 (delta 0), pack-reused 980188  
Receiving objects: 100% (980188/980188), 456.96 MiB | 52.89 MiB/s, done.  
Resolving deltas: 100% (716752/716752), done.  
[silen@penguyandeMacBook-Pro homebrew % brew update  
==> Updating Homebrew...  
==> Homebrew collects anonymous analytics.  
Read the analytics documentation (and how to opt-out) here:  
  https://docs.brew.sh/Analytics  
No analytics have been recorded yet (nor will be during this `brew` run).  
  
==> Homebrew is run entirely by unpaid volunteers. Please consider donating:  
  https://github.com/Homebrew/brew#donations  
  
Installing from the API is now the default behaviour!  
You can save space and time by running:  
  brew untap homebrew/core  
  brew untap homebrew/cask  
Already up-to-date.  
[silen@penguyandeMacBook-Pro homebrew % cd ~  
[silen@penguyandeMacBook-Pro ~ % brew -v  
Homebrew 4.2.18  
Homebrew/homebrew-core (git revision 1b674ff3829; last commit 2024-04-21)  
Homebrew/homebrew-cask (git revision c03494e24f; last commit 2024-04-20)  
silen@penguyandeMacBook-Pro ~ %
```

6. allure

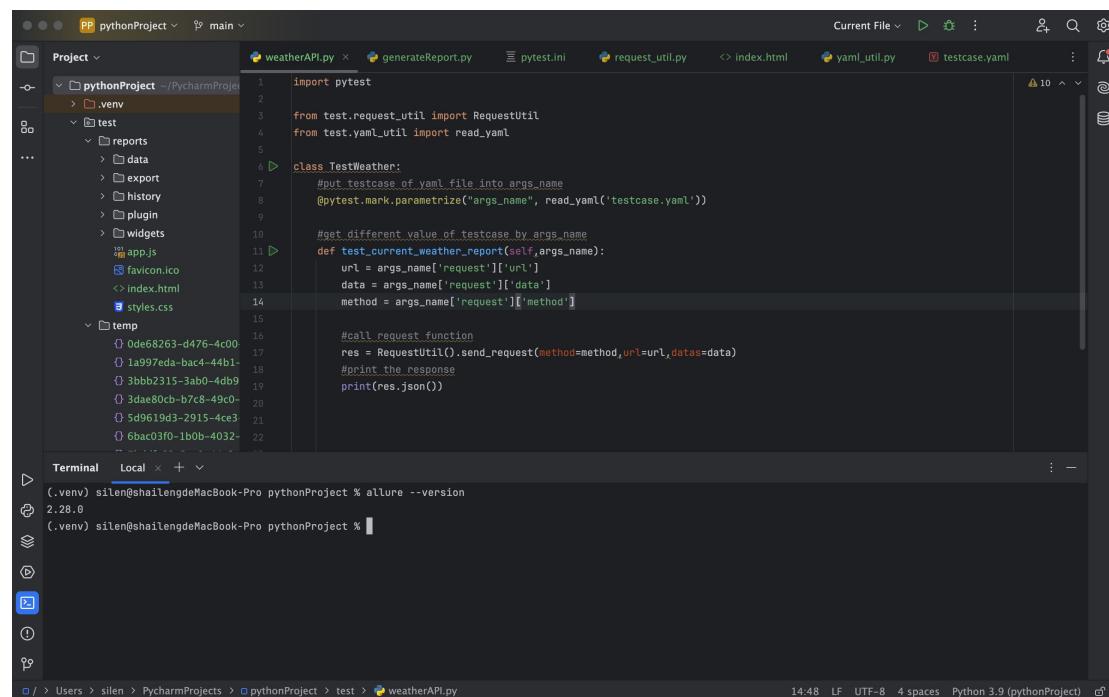
Can install allure by ‘brew install allure’ in terminal of pycharm.



The screenshot shows the PyCharm interface with the terminal tab active. The command `brew install allure` is being run in the terminal. The output shows the installation process, including the download and extraction of the package, and the final message indicating success: `2.28.0`.

```
(.venv) silen@shailengdeMacBook-Pro pythonProject % brew install allure
  % Total    Downloaded   Speed     Time    Estimated Left
  0       0B      0B/s      0:00      0:00:00
  25      19B      0B/s      0:00      0:00:00
  50      38B      0B/s      0:00      0:00:00
  75      57B      0B/s      0:00      0:00:00
 100     76B      0B/s      0:00      0:00:00
2.28.0
(.venv) silen@shailengdeMacBook-Pro pythonProject %
```

Can check allure version by ‘allure --version’ after installation.



The screenshot shows the PyCharm interface with the terminal tab active. The command `allure --version` is being run in the terminal. The output displays the installed version of allure: `2.28.0`.

```
(.venv) silen@shailengdeMacBook-Pro pythonProject % allure --version
2.28.0
(.venv) silen@shailengdeMacBook-Pro pythonProject %
```

Then also need to install allure-pytest by using “pip install allure pytest” in terminal of pycharm.

The screenshot shows the PyCharm interface. On the left, the project tree displays a directory structure under 'functionTestingProject'. On the right, the code editor shows a Python file named 'weatherAPI.py' with the following code:

```
1 import pytest
2 from test.request_util import RequestUtil
3 from test.yaml_util import read_yaml
4
5 class TestWeather:
6     #put testcase_of_yaml_file_into_args_name
7     @pytest.mark.parametrize("args_name", read_yaml('testcase.yaml'))
8
9     #get different value of testcase_by_args_name
10    def test_current_weather_report(self,args_name):
11        url = args_name['request']['url']
12        data = args_name['request']['data']
13        method = args_name['request']['method']
14
15        #call request function
16        res = RequestUtil().send_request(method=method,url=url,data=data)
17        #print the response
18        print(res.json())
```

Below the code editor is a terminal window showing the output of the pip command:

```
(.venv) silen@pengyuyandeMacBook-Pro functionTestingProject % pip install allure-pytest
Collecting allure-pytest
  Obtaining dependency information for allure-pytest from https://files.pythonhosted.org/packages/69/7b/430830ed7bf1ef078f9e55eb4b19cc059ef5310a20b5bd73eeb99e2c5ff1/allure_pytest-2.13.5-py3-none-any.whl.metadata
    Downloading allure_pytest-2.13.5-py3-none-any.whl.metadata (2.8 kB)
Requirement already satisfied: pytest>=4.5.0 in ./venv/lib/python3.12/site-packages (from allure-pytest) (8.1.1)
Collecting allure-python-commons==2.13.5 (from allure-pytest)
  Obtaining dependency information for allure-python-commons==2.13.5 from https://files.pythonhosted.org/packages/d0/18/79a66d6adc301281803398f7288583dff8d1d3f2672c07f/allure_python_commons-2.13.5-py3-none-any.whl.metadata
    Downloading allure_python_commons-2.13.5-py3-none-any.whl.metadata (5.5 kB)
Collecting attrs>=16.0.0 (from allure-python-commons==2.13.5->allure-pytest)
  Collecting attrs>=16.0.0 (from allure-python-commons==2.13.5->allure-pytest)
    Downloading attrs-2.12.0-py3-none-any.whl.metadata (5.5 kB)
```

The terminal also shows the current user, terminal type, and Python version.

7. jmeter

Download from website then install following the guidelines.

The screenshot shows the Apache JMeter download page. At the top, there are links for 'About', 'Download', 'Documentation', 'Tutorials', and 'Community'. The 'Download' section is highlighted with a red box around 'Download Releases'. Below this, there is a note about verifying file integrity using PGP keys. The main heading is 'Download Apache JMeter'. Underneath, it says 'Apache JMeter 5.6.3 (Requires Java 8+)'.

Download Apache JMeter

We recommend you use a mirror to download our release builds, but you must [verify the integrity](#) of the downloaded files using signatures downloaded from our main distribution directories. Recent releases (48 hours) may not yet be available from all the mirrors.

You are currently using <https://dlcdn.apache.org/>. If you encounter a problem with this mirror, please select another mirror. If all mirrors are failing, there are [backup mirrors](#) (at the end of the mirrors list) that should be available.

Other mirrors: <https://dlcdn.apache.org/> Change

The **KEYS** link links to the code signing keys used to sign the product. The **PGP** link downloads the OpenPGP compatible signature from our main site. The **SHA-512** link downloads the sha512 checksum from the main site. Please [verify the integrity](#) of the downloaded file.

For more information concerning Apache JMeter, see the [Apache JMeter](#) site.

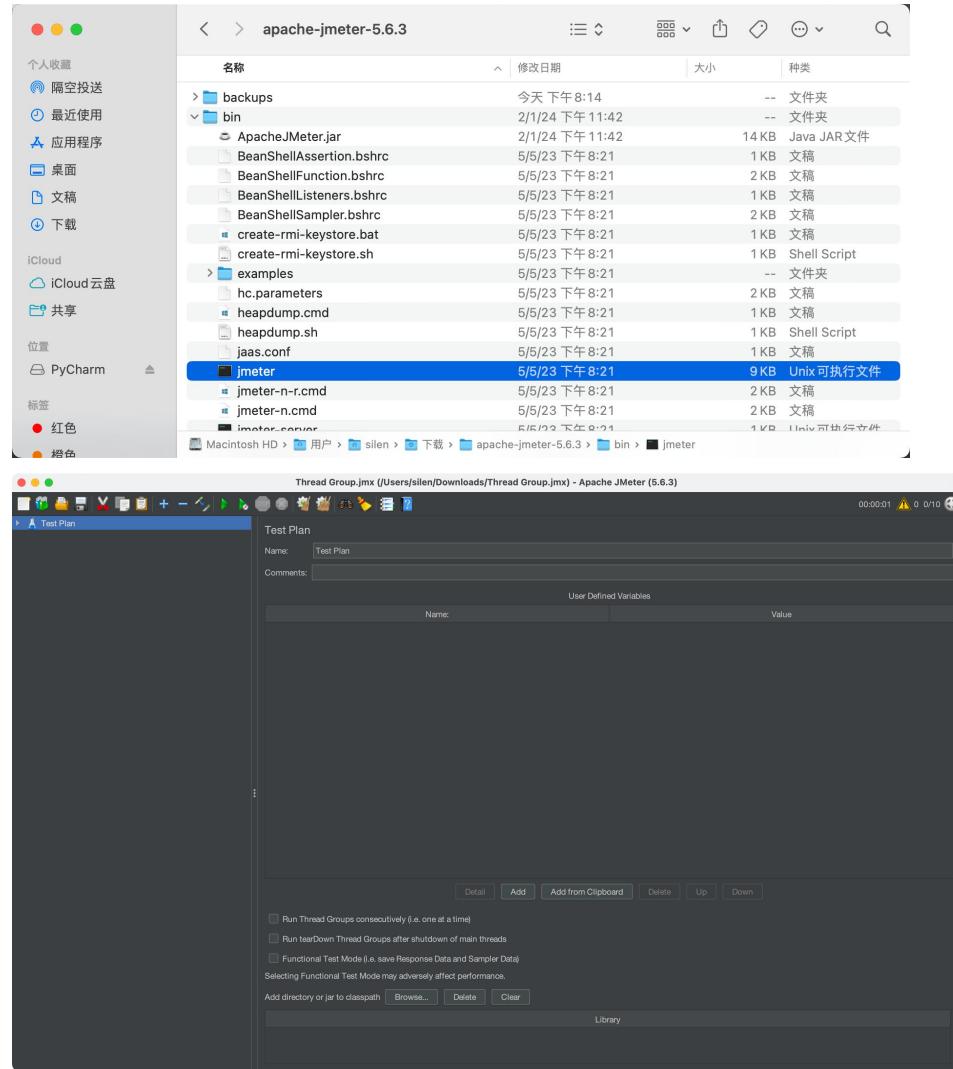
Apache JMeter 5.6.3 (Requires Java 8+)

Binaries

[apache-jmeter-5.6.3.tgz sha512 pgp](#)
[apache-jmeter-5.6.3.zip sha512 pgp](#)

Source

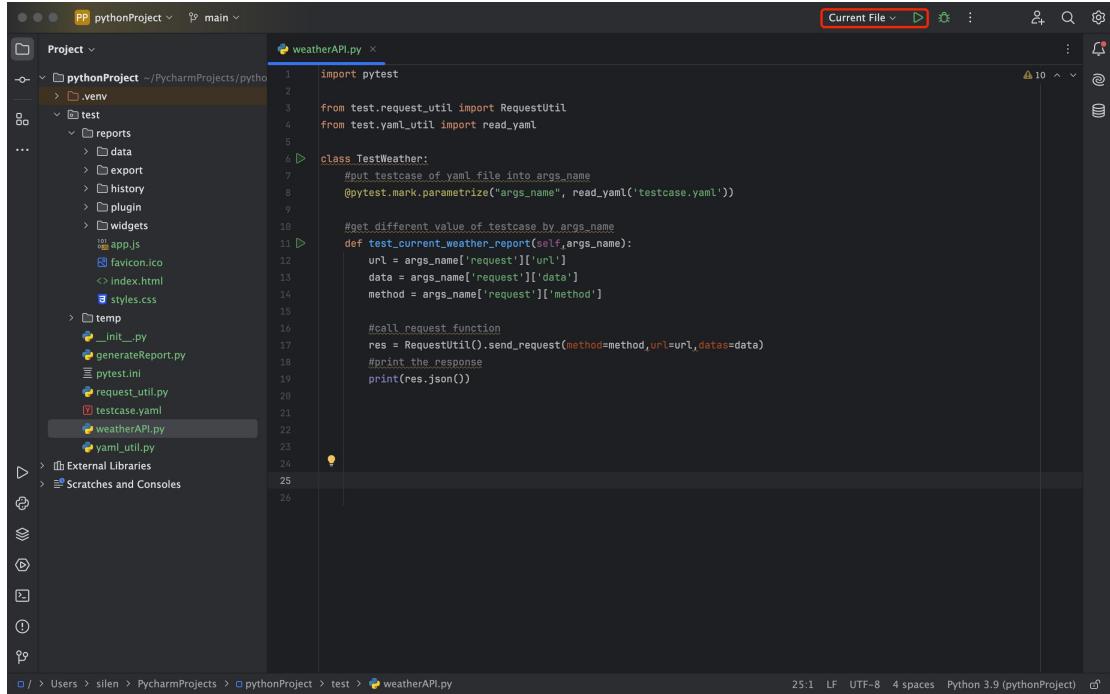
Unzip the file and open jmeter in bin then can see the home screen of jmeter.



Guidelines:

1. Execute testcase of function testing

After opening python project, find weatherAPI.py file, then click run button.

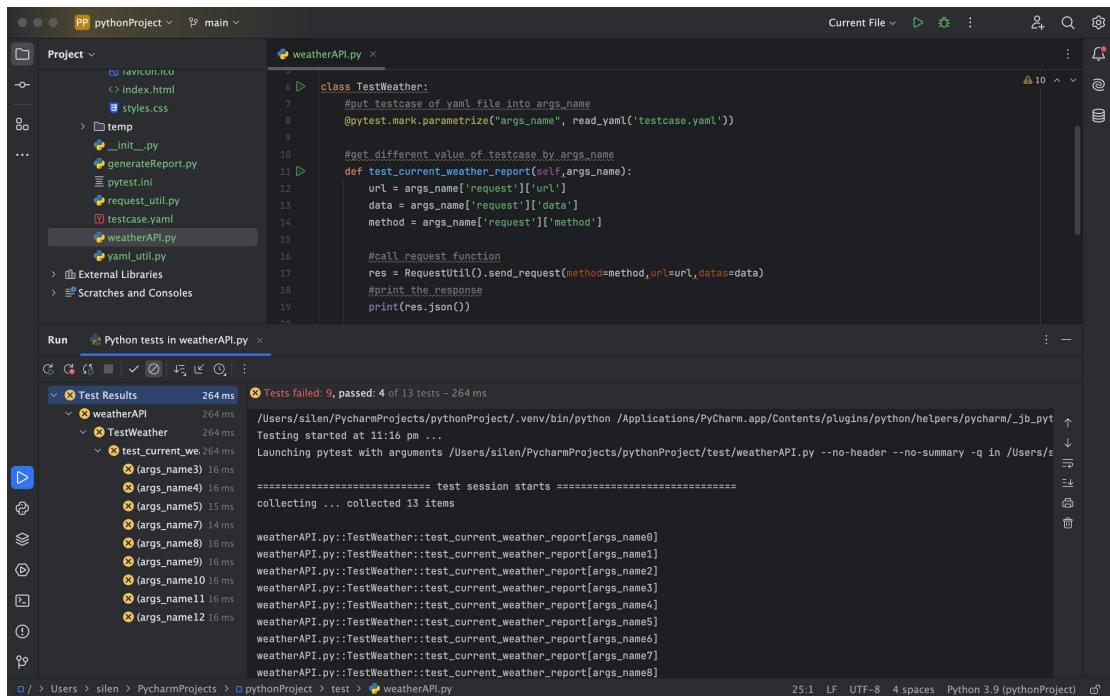


```
import pytest
from test.request_util import RequestUtil
from test.yaml_util import read_yaml

class TestWeather:
    #put testcase of yaml file into args_name
    @pytest.mark.parametrize("args_name", read_yaml('testcase.yaml'))
    #get different value of testcase by args_name
    def test_current_weather_report(self,args_name):
        url = args_name['request'][['url']]
        data = args_name['request'][['data']]
        method = args_name['request'][['method']]

        #call request function
        res = RequestUtil().send_request(method=method,url=url,data=data)
        #print the response
        print(res.json())
```

Then can see the brief running info(responses) of testcases, 4 of 11 testcases are passed, 9 of 11 testcases are failed.



Run Python tests in weatherAPI.py

Test Results 264 ms

Tests failed: 9, passed: 4 of 13 tests - 264 ms

```
/Users/silen/PycharmProjects/pythonProject/.venv/bin/python /Applications/PyCharm.app/Contents/plugins/python/helpers/pycharm/_jb_pyt
Testing started at 11:16 pm ...
Launching pytest with arguments /Users/silen/PycharmProjects/pythonProject/test/weatherAPI.py --no-header --no-summary -q in /Users/s
=====
collecting ... collected 13 items
=====
weatherAPI.py::TestWeather::test_current_weather_report[args_name0]
weatherAPI.py::TestWeather::test_current_weather_report[args_name1]
weatherAPI.py::TestWeather::test_current_weather_report[args_name2]
weatherAPI.py::TestWeather::test_current_weather_report[args_name3]
weatherAPI.py::TestWeather::test_current_weather_report[args_name4]
weatherAPI.py::TestWeather::test_current_weather_report[args_name5]
weatherAPI.py::TestWeather::test_current_weather_report[args_name6]
weatherAPI.py::TestWeather::test_current_weather_report[args_name7]
weatherAPI.py::TestWeather::test_current_weather_report[args_name8]
weatherAPI.py::TestWeather::test_current_weather_report[args_name9]
weatherAPI.py::TestWeather::test_current_weather_report[args_name10]
weatherAPI.py::TestWeather::test_current_weather_report[args_name11]
weatherAPI.py::TestWeather::test_current_weather_report[args_name12]
```

Testcases can be checked in testcase.yaml, including following scenarios:

Testcase1:Language of response-en

Testcase2:Language of response-tc

Testcase3:Language of response-sc

Testcase4:Missing dataType-en

Testcase5:Missing dataType-tc

Testcase6:Missing dataType-sc

Testcase7:Missing lang

Testcase8:Missing dataType and lang

Testcase9:Wrong dataType-en

Testcase10:Wrong dataType-tc

Testcase11:Wrong dataType-sc

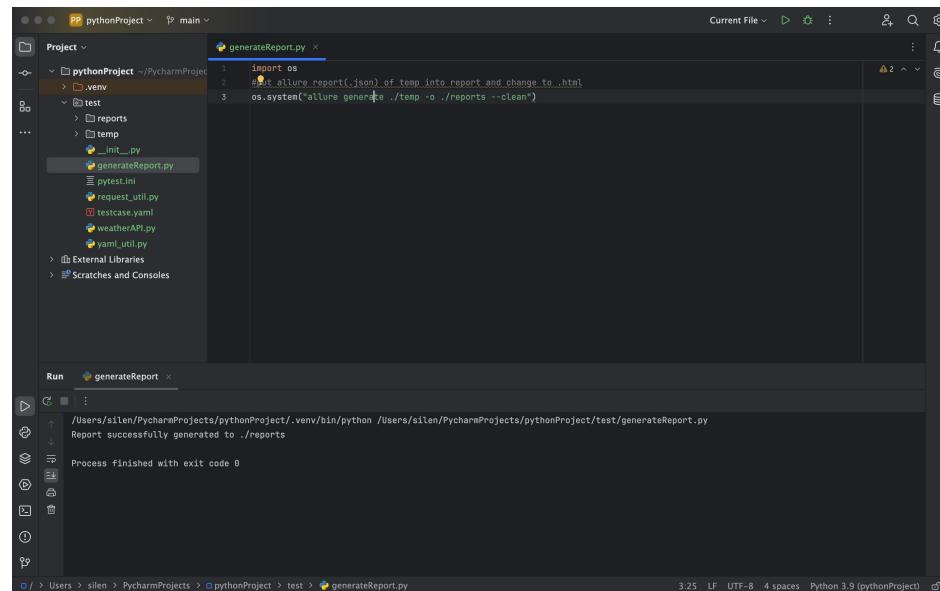
Testcase12:Wrong lang

Testcase13:Wrong dataType and lang

So only Testcase1, Testcase2, Testcase3 and Testcase7 should pass, the testing result meet expectation.

2. Generate function testing report

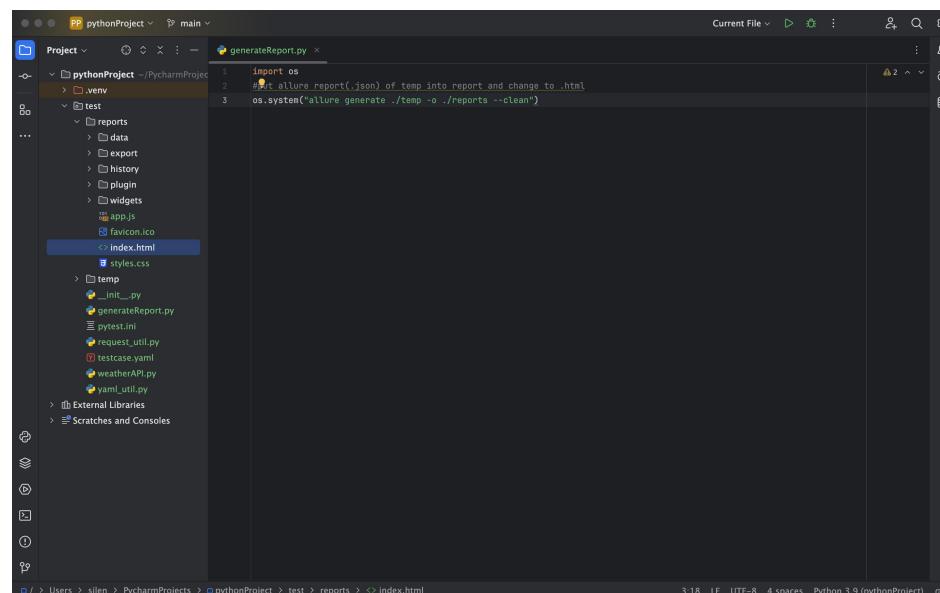
Find generateReport.py file, then click run button.



```
import os
#git allure_report(.json) of temp into report and change to ..html
os.system("allure generate ./temp -o ./reports --clean")
```

/Users/silen/PycharmProjects/pythonProject/.venv/bin/python /Users/silen/PycharmProjects/pythonProject/test/generateReport.py
Report successfully generated to ./reports
Process finished with exit code 0

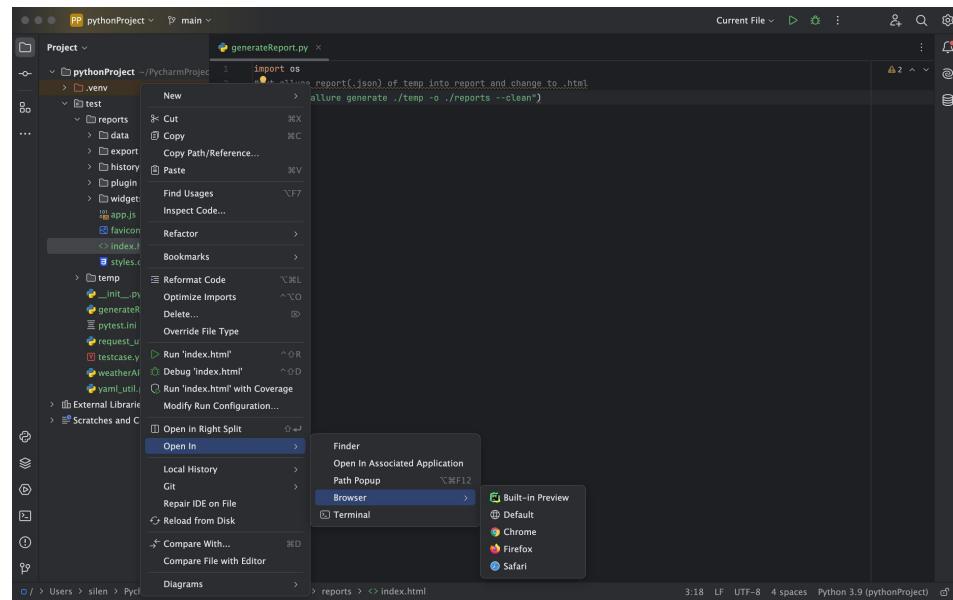
Then can see an index.html report is generated under reports directory.



```
import os
#git allure_report(.json) of temp into report and change to ..html
os.system("allure generate ./temp -o ./reports --clean")
```

index.html

Then open the index.html report in browser. Can check the detailed info of testcases.

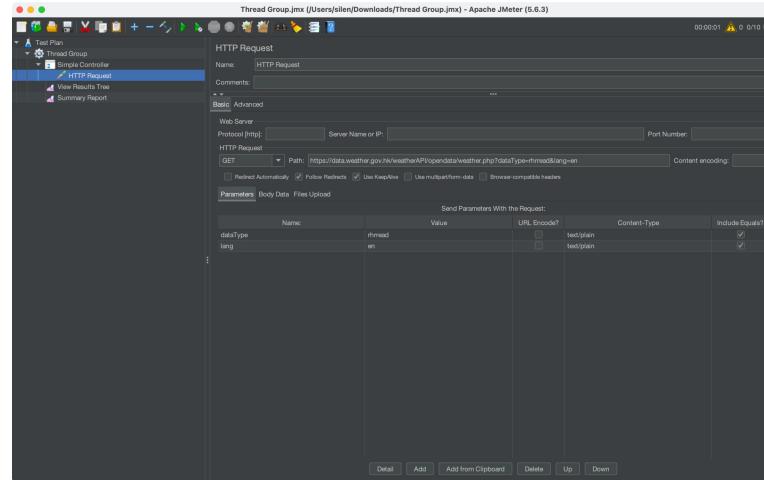


The screenshot shows the Allure Report interface for the 'weatherAPI' suite. The main dashboard includes a summary section with 13 test cases and a 30.76% completion rate. Below this are sections for 'SUITES', 'ENVIRONMENT', 'FEATURES BY STORIES', and 'CATEGORIES'. The 'SUITES' section shows one item total for 'weatherAPI'. The 'CATEGORIES' section shows one item total for 'Test defects'. The 'EXECUTORS' section indicates no information about tests executors.

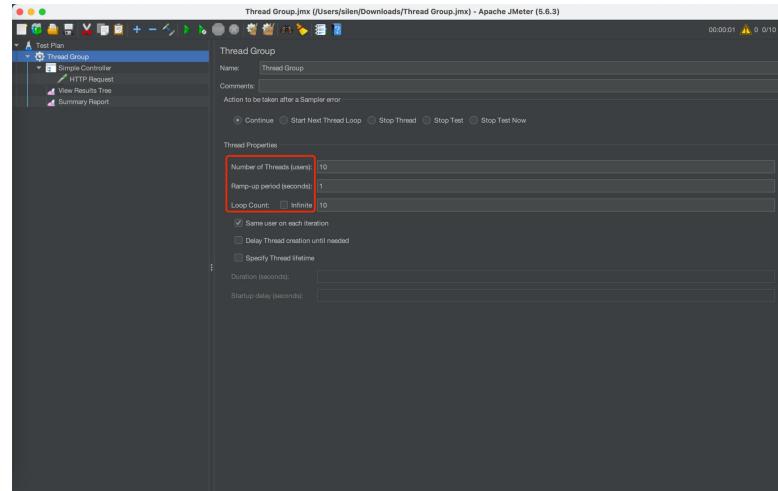
The screenshot shows a detailed view of a test case named 'test_current_weather_report' from the 'weatherAPI' suite. The test has 10 steps, all of which are marked as 'Passed'. The steps are numbered 1 through 10 and show various assertions and their results. The 'Parameters' section shows 'args_name6' with a value of '{"name": "Testcase7:Missing lang", "request": {"method": "get", "url": "https://data.weather.com/weather/current?lang=en-US"}, "expected": [{"status": "OK", "language": "en-US"}]}'. The 'Execution' section notes that no information about test execution is available.

3. Performance testing

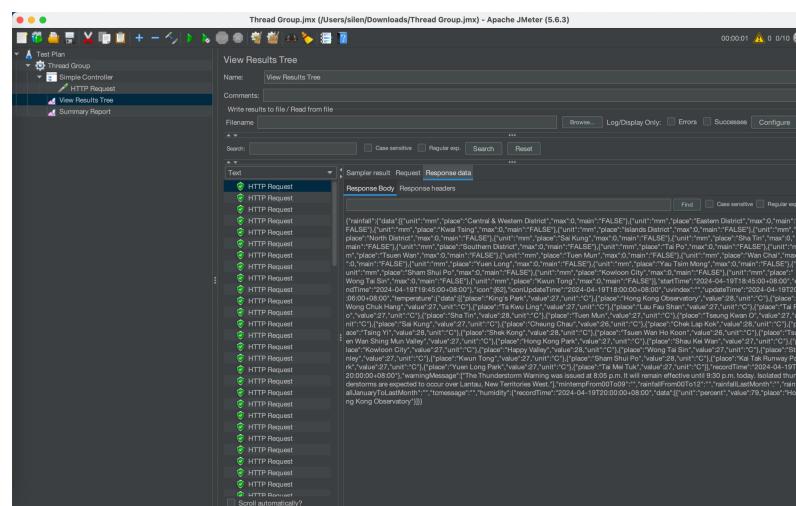
Add HTTP Request module with all request info. Then add View Results Tree module and Summary Report module to check the testing result.



Then can set the Number of Threads, Ramp-up period and Loop Count to simulate high concurrency for performance testing. As following setting, the Thread Group will make 10*10 requests. Then click execute button.



In View Results Tree, it shows all Request, Response Body and Response headers of every requests.



In Summary Report, it shows the general data like min/max/avg corresponding speed of that 100 requests or fail rate etc. Then compare these data with them in production environment to output the performance testing report.

The screenshot shows the Apache JMeter interface with a test plan containing a single thread group. The summary report shows the following data for 100 samples:

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
HTTP Request	100	15	9	53	8.22	0.00%	93.8/sec	332.54	16.49	3630.0
TOTAL	100	15	9	53	8.22	0.00%	93.8/sec	332.54	16.49	3630.0