Setting up MiPinG

This document describes how to generally setup MiPinG and how to use it to get personality predictions.

Version history

Date	Author	Content
2020-10-01	Henning Usselmann	Initial creation

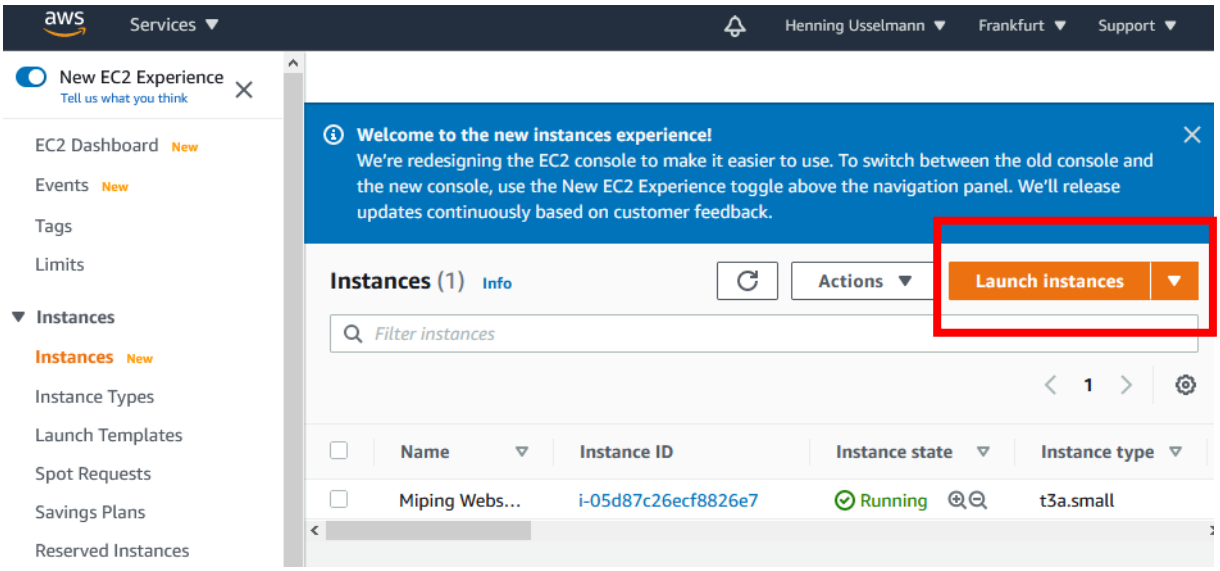
Inhalt

MiPinG needs 2 GB of memory, around 12 GB of disk space (including operating system) and needs Python to properly work. You definitely need a pair of **Twitter API Developer keys** to locally install MiPinG.

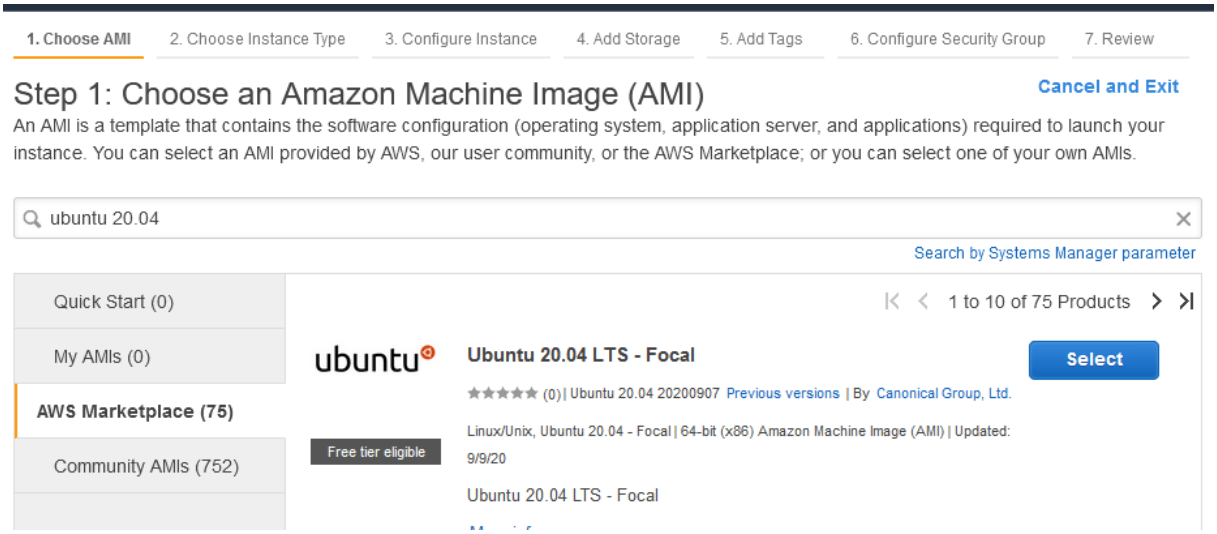
Prerequisites (installing Linux in AWS)

You need a unix system to install and use MiPinG. We will show the process with an Amazon AWS server. You can set this server up from any computer you like.

1. Create an account at Amazon AWS https://aws.amazon.com/
2. Go to EC2 dashboard → Instances → Launch Instance



3. We will select Ubuntu's 20.04 LTS – Focal version – search for "ubuntu 20.04" in the AWS marketplace. You might need to accept the terms of use.



4. MiPinG needs at least 2 GB of memory, therefore we choose “t3a.small” as Instance type. Go directly to review and launch.

1. Choose AMI | 2. Choose Instance Type | 3. Configure Instance | 4. Add Storage | 5. Add Tags | 6. Configure Security Group | 7. Review

Step 2: Choose an Instance Type

	t3	t3.micro	2	1	EBS only	Yes	Up to 5 Gigabit
	t3	t3.small	2	2	EBS only	Yes	Up to 5 Gigabit
	t3	t3.medium	2	4	EBS only	Yes	Up to 5 Gigabit
	t3	t3.large	2	8	EBS only	Yes	Up to 5 Gigabit
	t3	t3.xlarge	4	16	EBS only	Yes	Up to 5 Gigabit
	t3	t3.2xlarge	8	32	EBS only	Yes	Up to 5 Gigabit
	t3a	t3a.nano	2	0.5	EBS only	Yes	Up to 5 Gigabit
	t3a	t3a.micro	2	1	EBS only	Yes	Up to 5 Gigabit
■	t3a	t3a.small	2	2	EBS only	Yes	Up to 5 Gigabit
	t3a	t3a.medium	2	4	EBS only	Yes	Up to 5 Gigabit
	t3a	t3a.large	2	8	EBS only	Yes	Up to 5 Gigabit
	t3a	t3a.xlarge	4	16	EBS only	Yes	Up to 5 Gigabit
	t3a	t3a.2xlarge	8	32	EBS only	Yes	Up to 5 Gigabit
⊘	t4g	t4g.nano	2	0.5	EBS only	Yes	Up to 5 Gigabit
⊘	t4g	t4g.micro	2	1	EBS only	Yes	Up to 5 Gigabit
⊘	t4g	t4g.small	2	2	EBS only	Yes	Up to 5 Gigabit
⊘	t4g	t4g.medium	2	4	EBS only	Yes	Up to 5 Gigabit

Cancel | Previous | Review and Launch | Next: Configure Instance Details

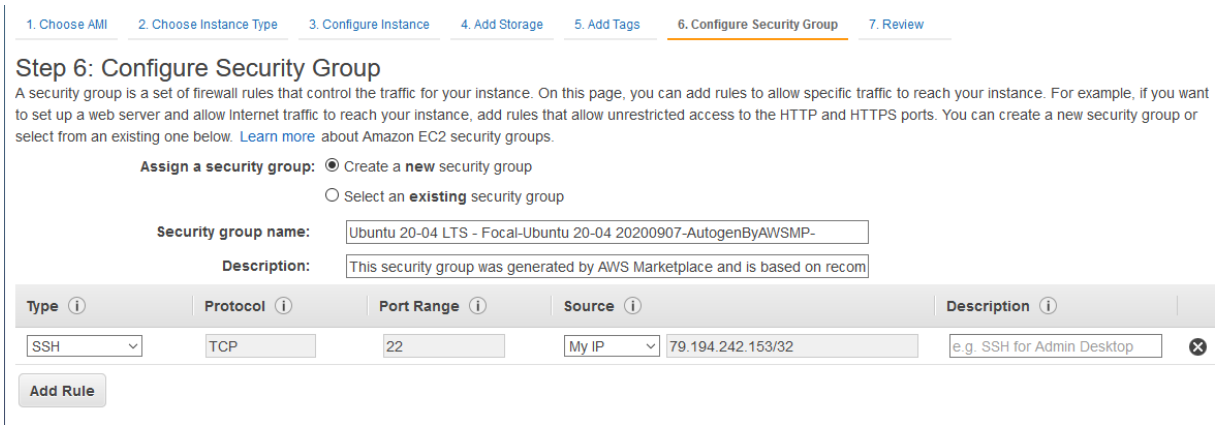
5. Edit security groups

Security Groups — Edit security groups

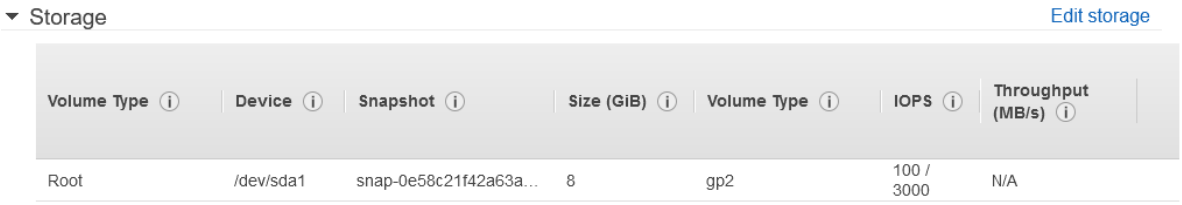
Security group name Ubuntu 20-04 LTS - Focal-Ubuntu 20-04 20200907-AutogenByAWSMP-
Description This security group was generated by AWS Marketplace and is based on recommended settings for Ubuntu 20.04 LTS - Focal version Ubuntu 20.04 20200907 provided by Canonical Group, Ltd.

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	0.0.0.0/0	

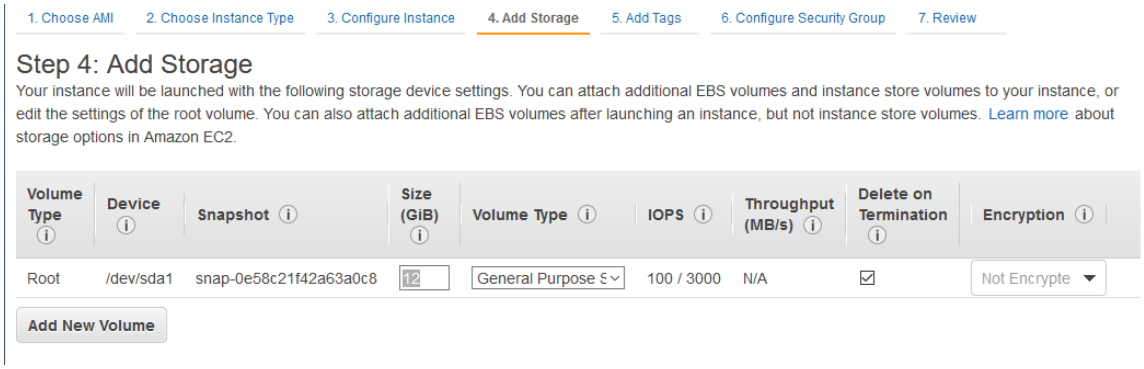
6. Create a new security group to allow and limit access to the machine (Firewall settings). It is recommended to choose “My IP” for SSH access. This has to be kept in might, in case you want to have access from different networks. Give the rule a meaningful description.



7. Go again to review and launch and edit storage

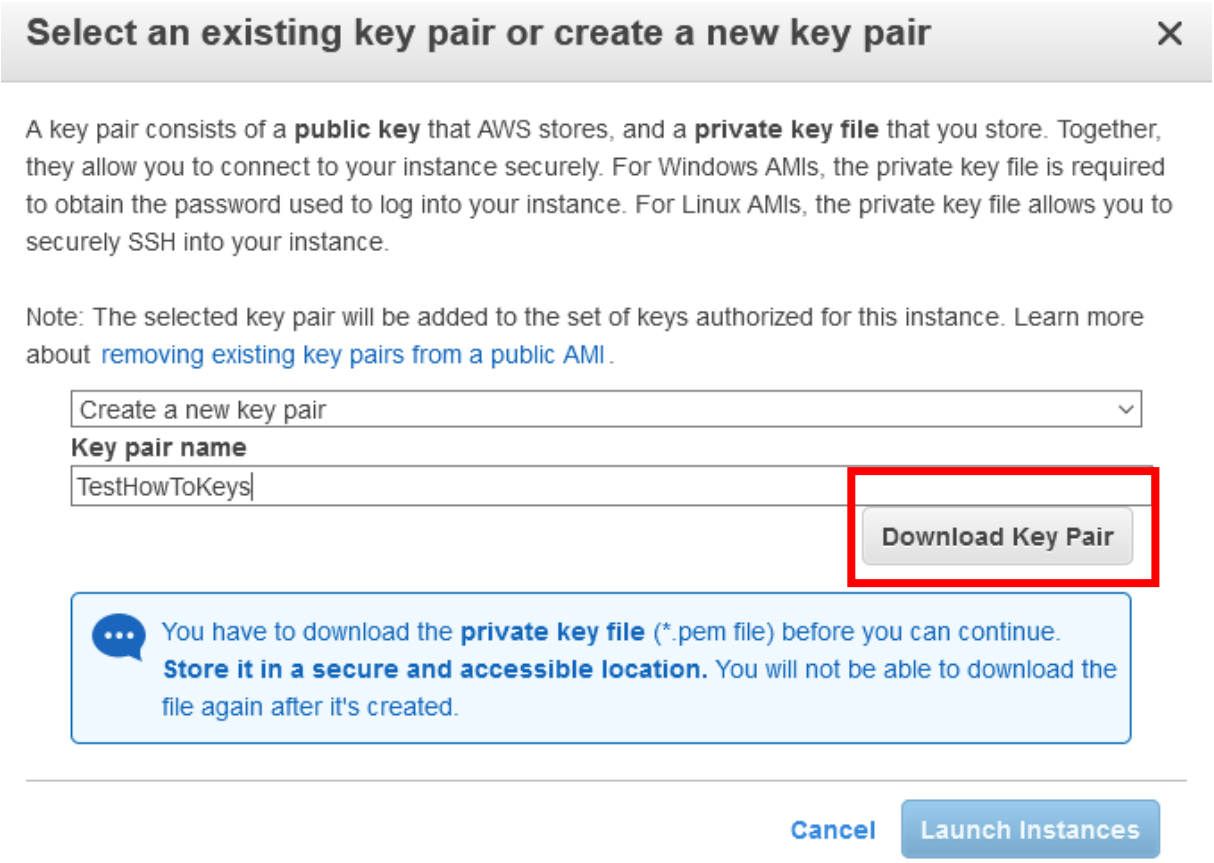


8. Increase the storage to at least 12 GB

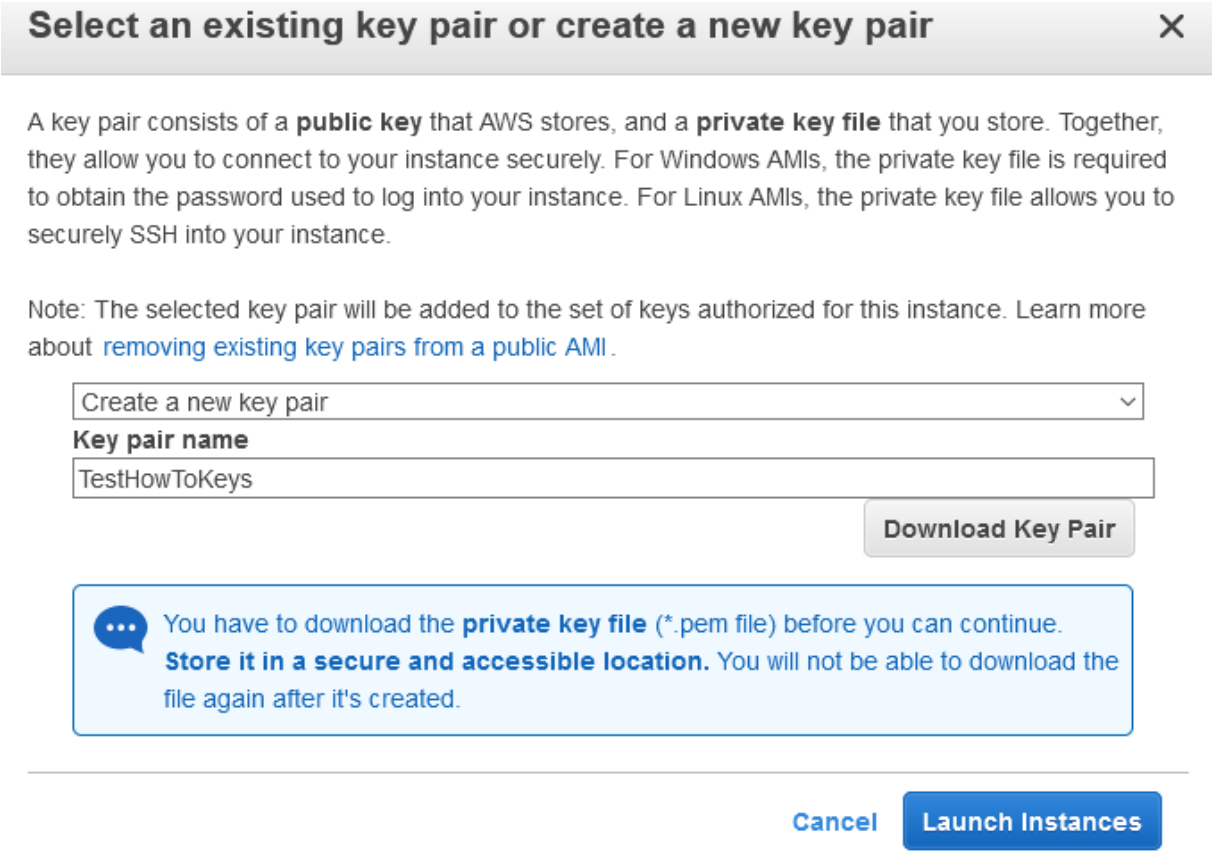


9. Press launch

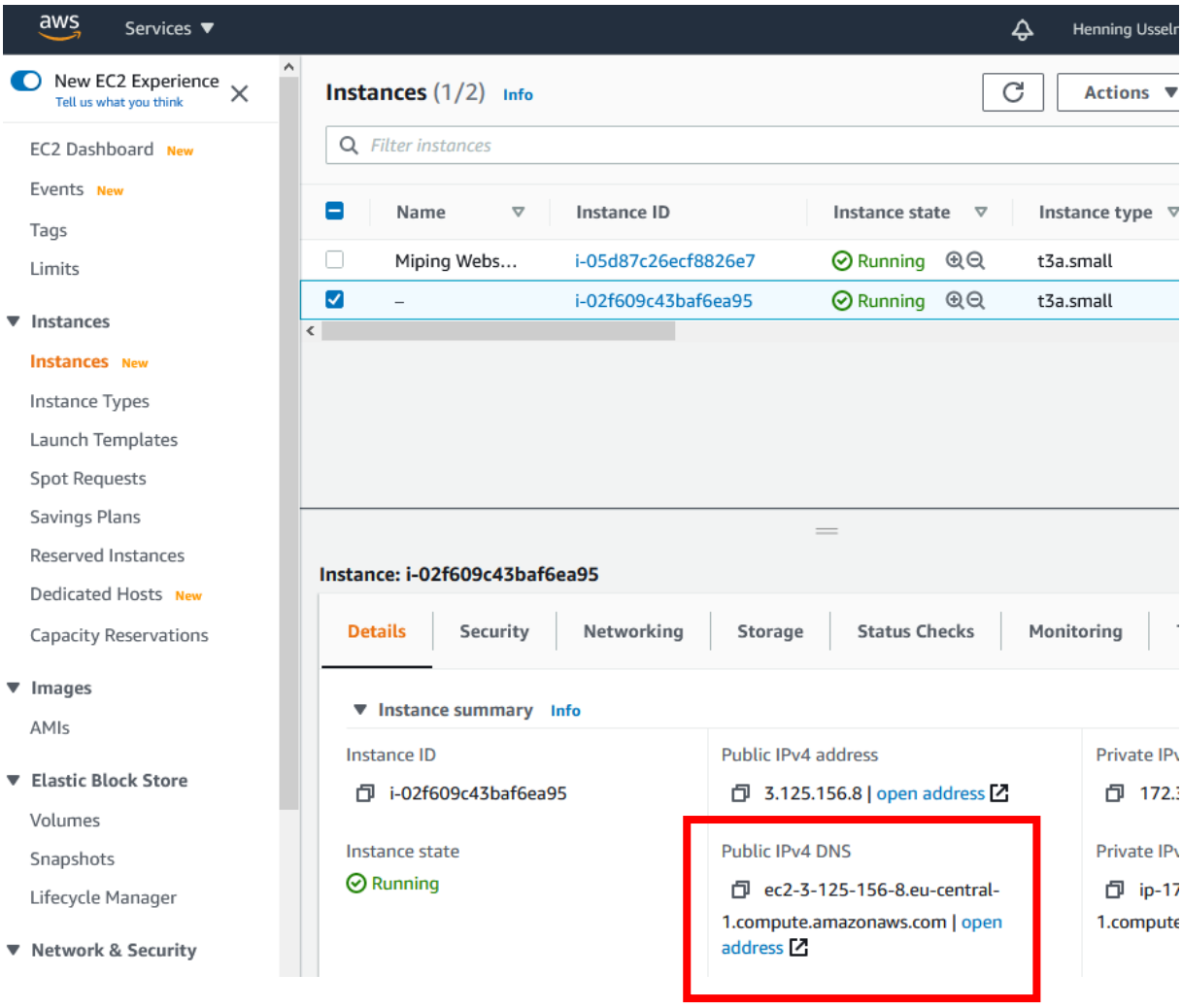
10. You have to provide or create access keys. We will create new ones and download the key pair. Save it on your local device, e.g. in C:\tmp\TestHowToKeys.pem



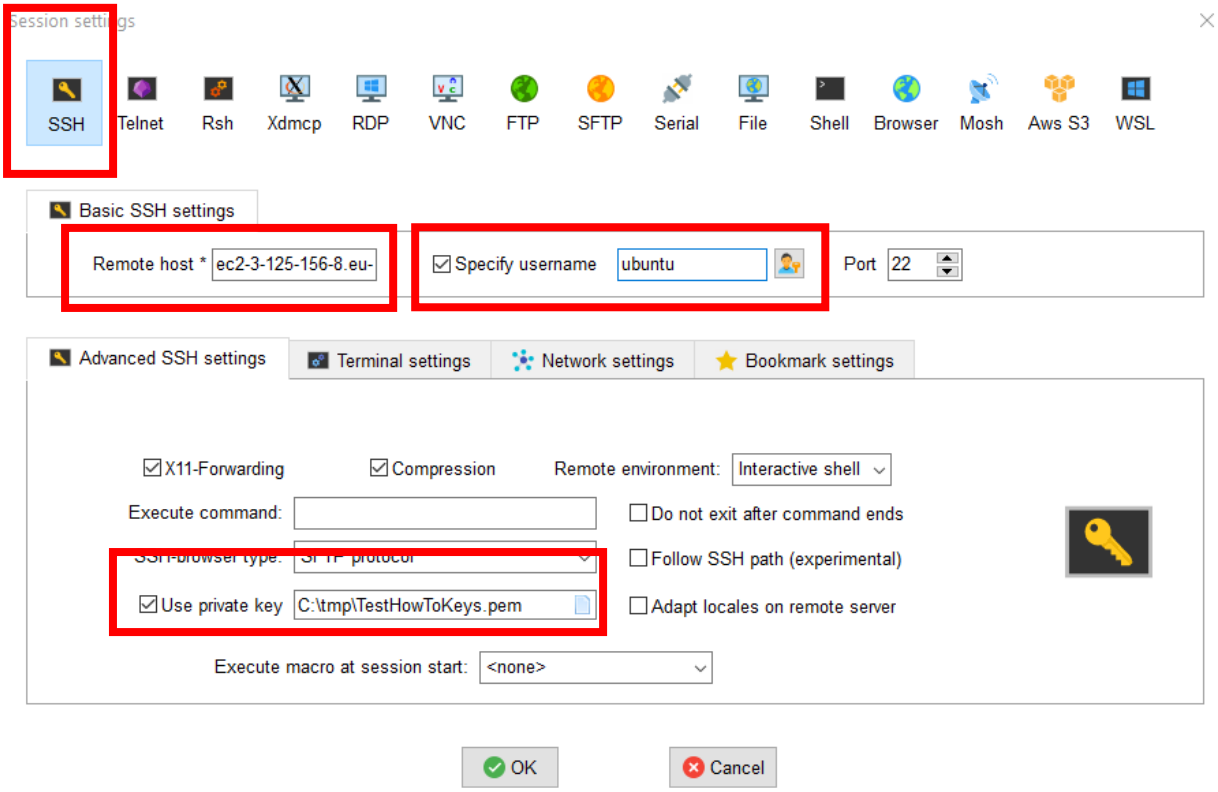
11. Once downloaded you are able to click “launch instances”



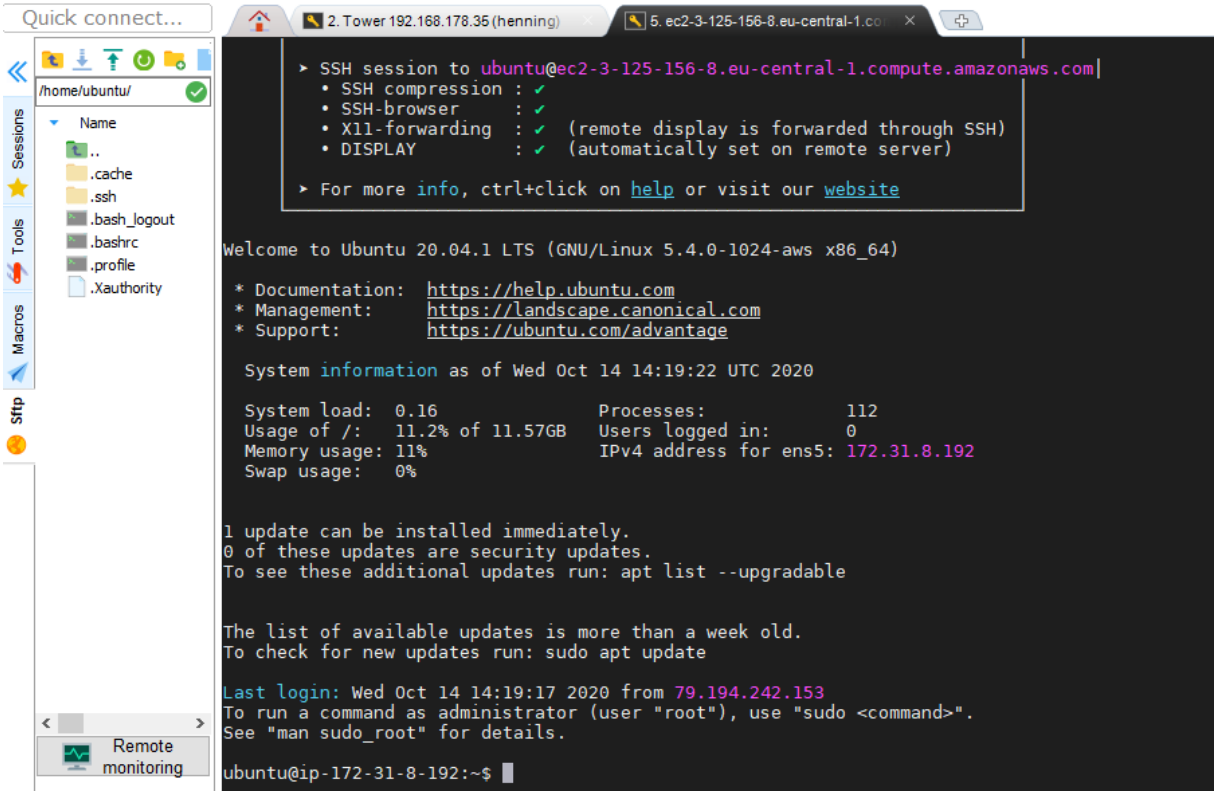
12. Go back to the EC2 dashboard and click instances. You will see your new instance and its public DNS address. Copy the address. In our case: ec2-3-125-156-8.eu-central-1.compute.amazonaws.com.



13. Download any SSH client you like. I recommend MobaXterm (https://mobaxterm.mobatek.net/). Launch this client and create a new session. Choose SSH, enter the public DNS address into the remote host field, click advanced SSH settings, and click use private key. Enter the path to your previously downloaded key. Specify the username as “Ubuntu”. Click OK.



14. The login to the server should be successful and you should see the following output.

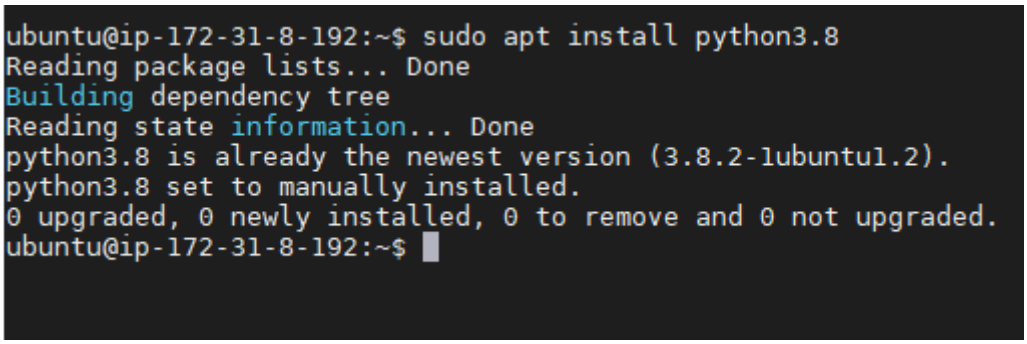


The following setup should be the same, regardless of whether you are using a server or local computer.

Installing Miping onto Linux

1. Install Python via

```
sudo apt install python3.8s
```



2. Create a virtual environment (for enclosing our activities in a Python virtual environment)

sudo apt-get update

sudo apt-get install python3.8-venv

Confirm with “Y”.

python3.8 -m venv tutorial-env

3. Activate virtual environment

source tutorial-env/bin/activate

You see (tutorial-env) at the beginning of the line.

```
ubuntu@ip-172-31-8-192:~$ python3.8 -m venv tutorial-env
ubuntu@ip-172-31-8-192:~$ source tutorial-env/bin/activate
(tutorial-env) ubuntu@ip-172-31-8-192:~$
```

4. Install and upgrade pip

sudo apt install python3-pip

pip install pip --upgrade

```
Processing triggers for libc-bin (2.31-0ubuntu9) ...
(tutorial-env) ubuntu@ip-172-31-8-192:~$ pip install pip --upgrade
Collecting pip
  Downloading pip-20.2.3-py2.py3-none-any.whl (1.5 MB)
     |[redacted]| 1.5 MB 8.8 MB/s
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 20.0.2
    Uninstalling pip-20.0.2:
      Successfully uninstalled pip-20.0.2
Successfully installed pip-20.2.3
```

5. If you want to use the MiPinG website, you need to install the webserver nginx

sudo apt install nginx

6. Actually install MiPinG via the pip package manager

`pip install miping`

This will download all dependencies:

```
Using legacy 'setup.py install' for googlemaps, since package 'wheel' is not installed.
Using legacy 'setup.py install' for PyYAML, since package 'wheel' is not installed.
Using legacy 'setup.py install' for termcolor, since package 'wheel' is not installed.
Using legacy 'setup.py install' for fire, since package 'wheel' is not installed.
Installing collected packages: gunicorn, supervisor, python-dotenv, idna, certifi, chardet, urllib3, requests, googlemaps,
ix, pycparser, cffi, oauthlib, requests-oauthlib, tweepy, PySocks, typing-extensions, protobuf, numpy, onnx, onnxconverter-
ommon, threadpoolctl, joblib, scipy, scikit-learn, skl2onnx, termcolor, fire, keras2onnx, onnxmltools, MarkupSafe, Jinja2,
lick, Werkzeug, itsdangerous, Flask, PyYAML, python-dateutil, pytz, pandas, cryptography, pyOpenSSL, onnxruntime, miping
    Running setup.py install for googlemaps ... done
    Running setup.py install for termcolor ... done
    Running setup.py install for fire ... done
    Running setup.py install for PyYAML ... done
Successfully installed Flask-1.1.2 Jinja2-2.11.2 MarkupSafe-1.1.1 PySocks-1.7.1 PyYAML-5.3.1 Werkzeug-1.0.1 certifi-2020.6.
0 cffi-1.14.1 chardet-3.0.4 click-7.1.2 cryptography-3.0 fire-0.3.1 googlemaps-4.4.2 gunicorn-20.0.4 idna-2.10 itsdangerous
1.1.0 joblib-0.16.0 keras2onnx-1.7.0 miping-0.1.5 numpy-1.19.1 oauthlib-3.1.0 onnx-1.7.0 onnxconverter-common-1.7.0 onnxmlt
ols-1.7.0 onnxruntime-1.4.0 pandas-1.1.0 protobuf-3.12.4 pyOpenSSL-19.1.0 pycparser-2.20 python-dateutil-2.8.1 python-doten
-0.14.0 pytz-2020.1 requests-2.24.0 requests-oauthlib-1.3.0 scikit-learn-0.23.2 scipy-1.5.2 six-1.15.0 skl2onnx-1.7.0 super
isor-4.2.0 termcolor-1.1.0 threadpoolctl-2.1.0 tweepy-3.9.0 typing-extensions-3.7.4.2 urllib3-1.25.10
(tutorial-env) ubuntu@ip-172-31-8-192:~$
```

7. Run the ontime configuration script. We need to run it 3 times. It sets up our local webserver and does all configurations for us.

It is important to run it first as the local user, and then as root user and finally again as local user.

`python tutorial-env/lib/python3.8/site-packages/miping/one_time_setup.py --setup_webserver True -d "localhost"`

The first run will download and extract the GloVe database file from (https://miping-glove.s3.eu-central-1.amazonaws.com/glove.zip) this will take a while, since this file is big. It is expected that some errors are shown, this is due to the different user permissions.

Output:

```
(tutorial-env) ubuntu@ip-172-31-8-192:~$ python tutorial-env/lib/python3.8/site-packages/miping/one_time_setup.py --setup_we
bserver True -d "localhost"
Setting up miping
Will setup webserver
Domain will be: localhost
Successfully created the directory /home/ubuntu/data
Successfully created the directory /home/ubuntu/data/glove
Creation of the directory /var/log/miping failed
Downloading GloVe file, this takes a while
Setting up webserver
Making sure nginx webserver is installed
nginx is already installed
Copy and modify nginx
Modify nginx config
Removing nginx default server from sites enabled
[Errno 13] Permission denied: '/etc/nginx/sites-enabled/default'
Copy nginx sites-available
[Errno 13] Permission denied: '/etc/nginx/sites-available/localhost.txt'
Try to run script as root
Copy nginx sites-enabled
[Errno 13] Permission denied: '/etc/nginx/sites-available/localhost' -> '/etc/nginx/sites-enabled/localhost'
Copy and modify supervisor config
Modify supervisor config
Copy and modify start_webserver.sh
Modify webserver.sh
Supervisor binary /home/ubuntu/tutorial-env/bin/supervisord
Copy and modify stop_webserver.sh
Creating SSL key and certificate
[Errno 13] Permission denied: '/home/ubuntucert.pem'
Modify .env
Glove path: /home/ubuntu/data/glove/glove.db
Please fill .env with keys and config
(tutorial-env) ubuntu@ip-172-31-8-192:~$
```

sudo python3 tutorial-env/lib/python3.8/site-packages/miping/one_time_setup.py --setup_webserver True -d "localhost"

Output:

```
(tutorial-env) ubuntu@ip-172-31-8-192:~$ sudo python3 tutorial-env/lib/python3.8/site-packages/miping/one_time_setup.py --se
tup_webserver True -d "localhost"
Setting up miping
Will setup webserver
Domain will be: localhost
Successfully created the directory /var/log/miping
GloVe database file already exists
Setting up webserver
Making sure nginx webserver is installed
nginx is already installed
Copy and modify nginx
file exists
Modify nginx config
Removing nginx default server from sites enabled
Copy nginx sites-available
Copy nginx sites-enabled
Copy and modify supervisor config
file already exists
Copy and modify start_webserver.sh
Files already exist
Creating SSL key and certificate
Copying certificate to /etc/ssl/certs
Copying key to /etc/ssl/private
(tutorial-env) ubuntu@ip-172-31-8-192:~$
```

python tutorial-env/lib/python3.8/site-packages/miping/one_time_setup.py --setup_webserver True -d "localhost"

Output: the errors are expected

```
(tutorial-env) ubuntu@ip-172-31-8-192:~$ python tutorial-env/lib/python3.8/site-packages/miping/one_time_setup.py --setup_we
bserver True -d "localhost"
Setting up miping
Will setup webserver
Domain will be: localhost
GloVe database file already exists
Setting up webserver
Making sure nginx webserver is installed
nginx is already installed
Copy and modify nginx
file exists
Modify nginx config
Removing nginx default server from sites enabled
Copy nginx sites-available
[Errno 13] Permission denied: '/etc/nginx/sites-available/localhost.txt'
Try to run script as root
Copy nginx sites-enabled
Exists already
Copy and modify supervisor config
file already exists
Copy and modify start_webserver.sh
Files already exist
Copying certificate to /etc/ssl/certs
[Errno 13] Permission denied: '/etc/ssl/certs/cert.pem'
Try with root
Copying key to /etc/ssl/private
[Errno 13] Permission denied: '/etc/ssl/private/key.pem'
Try with root
(tutorial-env) ubuntu@ip-172-31-8-192:~$
```

Configuration

By performing

Ls –a

You can see the ".env" file.

```
(tutorial-env) ubuntu@ip-172-31-8-192:~$ ls -a
.  ..  .Xauthority  .bash_logout  .bashrc  .cache  .env  .profile  .ssh  .sudo_as_admin_successful  data  tutorial-env
(tutorial-env) ubuntu@ip-172-31-8-192:~$
```

Open it with your favourite editor:

.env - Editor

Datei Bearbeiten Format Ansicht Hilfe

```
# Twitter
twitter_consumer_key=
twitter_consumer_secret=

# user level access - read only - only necessary for streaming
twitter_access_token=
twitter_access_token_sec=

# GloVe
# relative path inside miping repository for glove file or data base file
# default path
glove_file_path=/home/ubuntu/data/glove/glove.db
# if true, glove_file_path points to SQL lite database file, if false glove flat file
glove_database_mode=True

# google recaptcha key in webapplication
# remove if no recaptcha needed
google_recaptcha=

# Google, necessary for location validation
google_places_api=


# IBM Watson Personality Insight, only needed for own training approaches
IBM_URL=
IBM_IAM_APIKEY=
```

You need to at least a twitter_consumer_key and twitter_consumer_secret. You have to apply for this at https://developer.twitter.com/ .

This will look something like this:

```
# Twitter
twitter_consumer_key=wFBpWTPI
twitter_consumer_secret=KSH7
```

You can also see, that the onetime script already entered the correct glove_file_path for the GloVe data base file. If you manually downloaded it, you would have to enter the correct path here.

To activate reCaptcha just provide your individual key here. You need to exchange the Javascript file at the following location. Just swap script.js with script-captcha.js. You also need to put your Public reCaptcha key in the respective form in index.html. Afterwards restart the webserver nginx.

~/tutorial-env/lib/python3.8/site-packages/miping/webapp/webfiles/www/js

```
ubuntu@ip-172-31-8-192:~/tutorial-env/lib/python3.8/site-packages/miping/webapp/webfiles/www/js$ ls
scripts-captcha.js  scripts.js
ubuntu@ip-172-31-8-192:~/tutorial-env/lib/python3.8/site-packages/miping/webapp/webfiles/www/js$
```

The other parameters can be left empty, as they are only relevant for training.

Start the webserver

You only need to execute one script.

Cd data

Ls

The setup script downloaded all necessary files:

```
(tutorial-env) ubuntu@ip-172-31-8-192:~$ cd data
(tutorial-env) ubuntu@ip-172-31-8-192:~/data$ ls
glove  localhost.txt  miping-gunicorn.conf  start_webserver.sh  stop_webserver.sh
(tutorial-env) ubuntu@ip-172-31-8-192:~/data$
```

To start or stop the webserver just execute the respective script.

sudo sh start_webserver.sh

sudo sh stop_webserver.sh

By entering

curl --insecure https://localhost

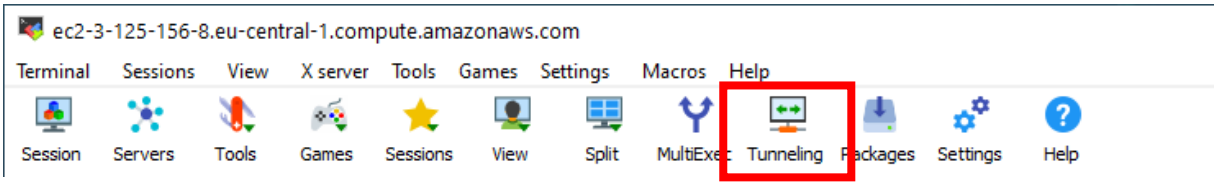
you should get the homepage of MiPinG back.

```
(tutorial-env) ubuntu@ip-172-31-8-192:~/data$ curl --insecure  https://localhost
<!DOCTYPE html>

<html lang="en">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="css/normalize.css">
    <link rel="stylesheet" href="css/skeleton.css">
    <link rel="stylesheet" href="css/styles.css">
    <link rel="icon" type="image/svg" href="images/favicon.svg">
    <title>MiPing Home</title>

    <meta name="description" content="Mining Personality In German">
    <meta name="author" content="Henning Usselmann">
</head>

<body>
  <div class="container">

    <div class="row headerMenu">
      <div class="twelve columns" style="margin-top: 2%; margin-left: 2%;">
        <div>
            <div id="headerImgDiv"><img src="images/favicon.svg" alt="MiPinG Logo" /></div>
            <div id="headerTextDiv"><h5> <b>Mi</b>ning <b>P</b>ersonality <b>In</b> <b>G</b>erman</h5></div>
        </div>
      </div>
    </div>
    <div class="row headerMenu">
      <div class="eight columns offset-by-two">
        <ul class="nav">
            <li class="active"><a href="index.html">Home</a></li>
            <li><a href="html/about.html">About</a></li>
            <li><a href="html/big5.html">Big Five Personality</a></li>
            <li><a href="html/privacy.html">Privacy</a></li>
            <li><a href="html/impressum.html">Impressum – Legal Notice</a></li>
      </div>
    </div>

    <div class="row description">
        <div class="six columns offset-by-three headerText">
            <h6>Introduction</h6>
```

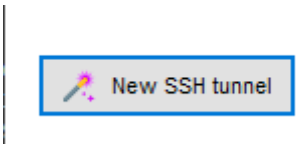
Access website

Since we configured MiPinG to host the website locally, we need to setup a tunnel via MobaXTerm. On your local machine you could just enter “https://localhost” into your browser.

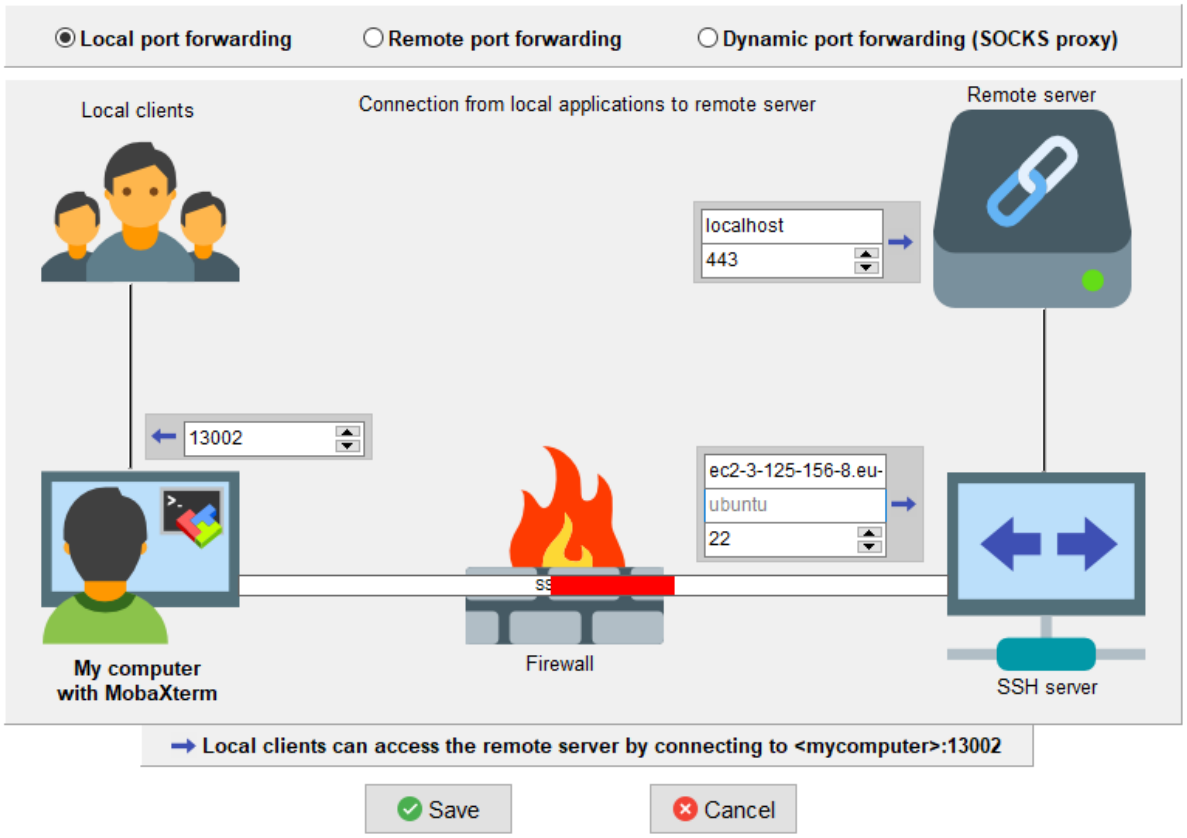
1. Click “Tunneling”



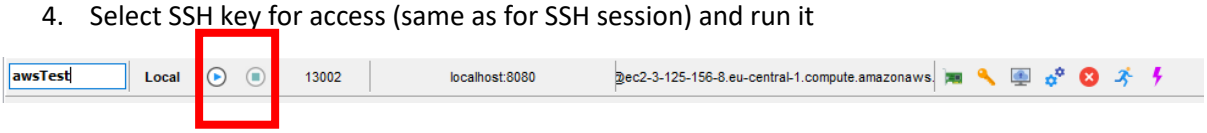
2. New SSH Tunnel



3. Local port forwarding – setup any port you like on the left side. On the right side choose “localhost” and port “443” (HTTPS). Below enter you public DNS from the AWS server, Ubuntu user name and port 22.

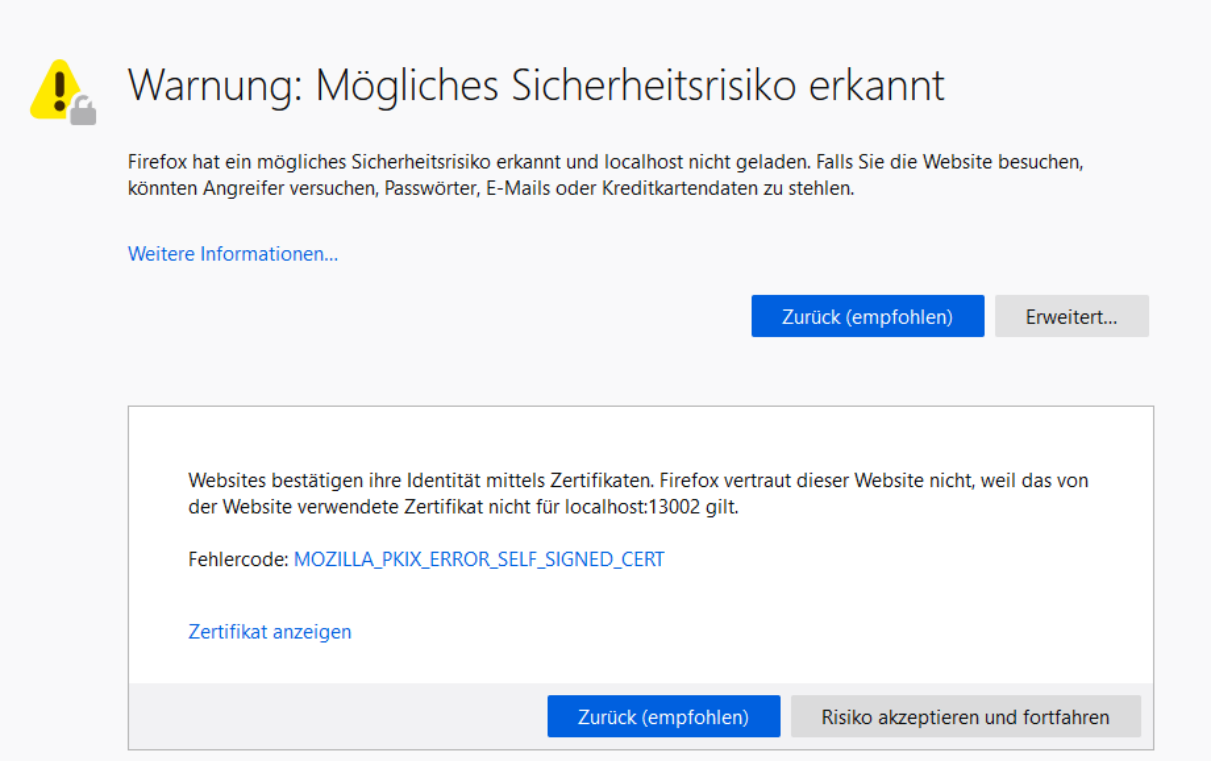


4. Select SSH key for access (same as for SSH session) and run it

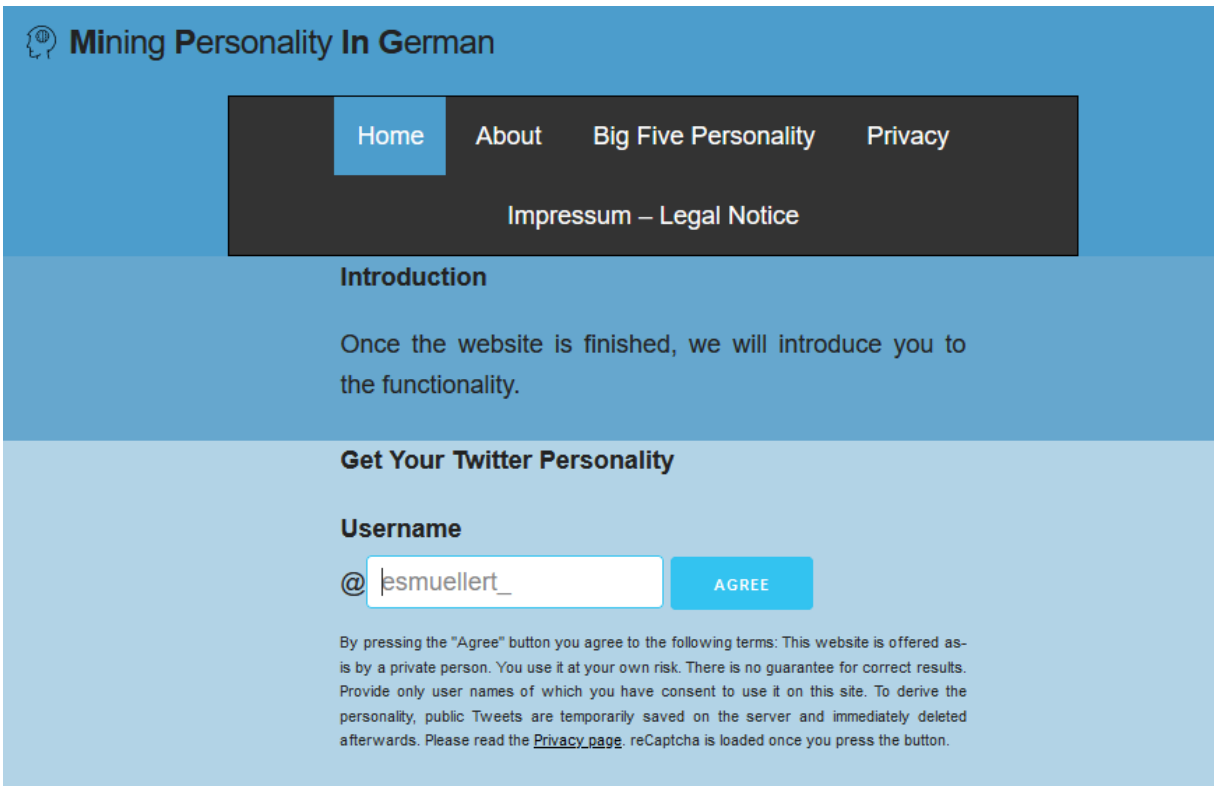


5. Using your favourite browser, you can now go to https://localhost:13002/

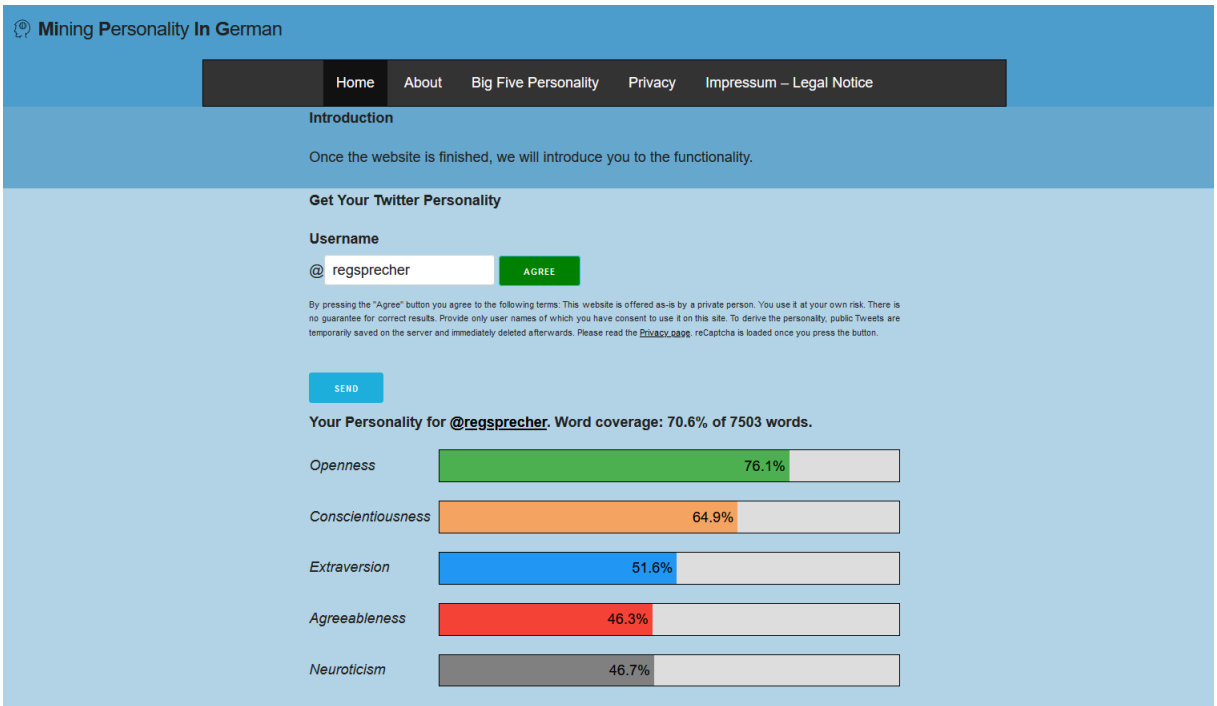
Since the SSL certificate is self-signed during the onetime configuration, a warning message is shown. Depending on your browser there are different ways of ignoring this issue. For Firefox you can accept the risk and continue.



6. MiPinG is ready to use.



Enter a public Twitter name and you will get the personality results:



Access the API directly

If you are interested in an automatic approach of using MiPinG you can setup the whole server in the same way. Start the server the same way.

You can make a get request to the following API Endpoint:

`curl --insecure https://localhost/api/test`

```
(tutorial-env) ubuntu@ip-172-31-8-192:~/data$ curl --insecure https://localhost/api/test
<h1 style='color:Black'>MiPing Backend is up and running. reCaptcha is set to: False . Twitter keys are set.</h1>(tutorial-e
nv) ubuntu@ip-172-31-8-192:~/data$
```

If the result is positive, you can start querying the API (if you did not activate reCaptcha).

You need to provide the username as the following formatted JSON: {"twitterHandle":"regsprecher"}

A full working example would be the following:

`curl --insecure --header "Content-Type: application/json" --request POST --data '{"twitterHandle":"regsprecher"}' https://localhost/api/personality`

Resulting in the following response:

{"big5_agreeableness":0.4628736972808838,"big5_conscientiousness":0.6485477685928345,"big5_extraversion":0.5162984132766724,"big5_neuroticism":0.466714084148407,"big5_openness":0.7610849142074585,"coverage":0.7061175529788085,"userName":"regsprecher","wordCount":7503}

```
ubuntu@ip-172-31-8-192:~/tutorial-env/lib/python3.8/site-packages/miping/webapp/webfiles/www/js$ curl --insecure --header "C
ontent-Type: application/json" --request POST --data '{"twitterHandle":"regsprecher"}' https://localhost/api/personality
{"big5_agreeableness":0.4628736972808838,"big5_conscientiousness":0.6485477685928345,"big5_extraversion":0.5162984132766724,
"big5_neuroticism":0.466714084148407,"big5_openness":0.7610849142074585,"coverage":0.7061175529788085,"userName":"regsprecheh
r","wordCount":7503}
ubuntu@ip-172-31-8-192:~/tutorial-env/lib/python3.8/site-packages/miping/webapp/webfiles/www/js$
```

Pretty JSON:


```
{
    "big5_agreeableness": 0.4628736972808838,
    "big5_conscientiousness": 0.6485477685928345,
    "big5_extraversion": 0.5162984132766724,
    "big5_neuroticism": 0.466714084148407,
    "big5_openness": 0.7610849142074585,
    "coverage": 0.7061175529788085,
    "userName": "regsprecher",
    "wordCount": 7503
}
```


Via this, you could automatically process more data.