Running tests on an AWS server

Browser tests can be run on an AWS server using Chrome or Firefox browsers. The current runtime for a set of tests is roughly one hour.

Steps 1-6 describe how to set up a server for the first time, and 7-8 describe how to run tests.

Wranglers must have access to AWS EC2 in order to complete this setup guide.

Server setup

1) Launch the server

From the AWS EC2 console, click “Launch instance”. Use the latest AMI. Choose “m4.large” for instance size and “dev” for keys. These keys are stored in a file called key-file.pem, which can be obtained from a software developer. This file should be stored locally in ~/.ssh.

Once the instance is running, open a terminal and ssh to the server:

```
$ ssh -i ~/.ssh/key-file.pem ubuntu@i-###.instance-url.org
```

The correct instance ID can be obtained by checking the AWS EC2 console.

1a) [Optional] Modify ssh config file

This step is for convenience and is not required. Locally, open ~/.ssh/config and copy the following information in, replacing the instance ID with the correct ID.

```
Host *
 UseKeychain yes
 StrictHostKeyChecking no
 AddKeysToAgent yes
 ForwardAgent yes
 IdentitiesOnly yes
Host awstests
   Hostname i-####.instance-url.org
   User ubuntu
   IdentityFile ~/.ssh/key-file.pem
```

This stores the info for the instance, so the ssh command to access the server can be shortened to:

```
$ ssh awstests
```

2) Create credentials.json

Contact another wrangler for a copy of credentials.json.
In the home directory of the instance (/home/ubuntu), create credentials.json.

```
$ cd
$ touch credentials.json
$ vi credentials.json
```

Copy and paste in the contents of credentials.json, then save and exit (`:wq`) the file.

3) Install browsers and drivers for Chrome and Firefox
These browsers and corresponding drivers are necessary in order to run the testing script.

```
$ sudo apt install chromium-browser
$ sudo apt install firefox
$ sudo apt install chromium-chromedriver
$ sudo apt install firefox-geckodriver
```

4) Clone the pyencoded-tools repo into the home directory
The repository must be cloned into the home directory. The scripts are hardcoded to import from ~/pyencoded-tools, and will crash with ImportError if the repository is located at a different path.

```
$ cd
$ git clone https://github.com/ENCODE-DCC/pyencoded-tools.git
```

Double check that the directory /home/ubuntu/pyencoded-tools now exists.

5) Make a virtual environment
For organizational purposes, create a separate virtual environment for running browser tests.

```
$ cd ~/pyencoded-tools
$ python3 -m venv qancode
```

6) Install dependencies
Install the tests' dependencies using pip:

```
$ cd ~/pyencoded-tools
$ pip install -r requirements.txt
```

Run tests

7) Run tests
The run_tests.py script can run each of check_trackhubs, compare_facets, and find_differences tests as Public, test2, or test4 users, using either Chrome or Firefox browsers. However, current protocol only requires running tests as a Public user using Chrome.

The script takes two positional arguments: the production site URL followed by the demo site URL.

Sample commands to run tests:

```
$ cd ~/pyencoded-tools/qancode
$ source bin/activate
$ nohup python run_tests.py https://demo-url.org/
https://production-url.org/ &
```

As shown above, use nohup to put the command in the background to prevent the process from terminating if you disconnect. If the process has already begun, enter the following commands to detach from the process: `ctrl+Z` , bg, `disown`

The stdout of the scripts will be saved into a file named nohup.out in the current directory (~/pyencoded-tools/qancode). Check the stdout using `tail` or `cat`:

```
$ tail -f nohup.out
```

If nohup.out contains stdout from a previous run, you can delete the file (`rm nohup.out`). Otherwise, when another nohup command is run, it will append the new output to the existing file.

Other options for run_tests.py:

- `--dry-run` : print stdout only and do not store output
- `-t` : list of test users (2, 4). If no parameters are provided, -t defaults to the Public user.
- `-b` : list of browsers (Chrome-headless, Firefox-headless)
- `-p` : list of tests ('compare_facets', 'check_trackhubs', 'find_differences')

Sample command to run tests with additional parameters:

```
$ nohup python run_tests.py https://demo-url.org/
https://production-url.org/ -b Chrome-headless -t &
```

Allow the tests to complete. A typical runtime is roughly one hour.
To exit the instance while tests are running, use the following command:

```
$ exit
```

8) Retrieve output
When the tests are complete, ssh to the instance again.

```
$ ssh awstests
```

The output is saved in a subdirectory of the home directory called output. Zip the output folder to transfer the files more easily:

```
$ cd
$ tar -zcvf output.tar.gz output
```

Exit the instance. Then, get the IPv4 Public IP for the instance by checking AWS EC2. Replace the IP address in the following command, which will download the zipped output file to your local machine:

```
$ scp ubuntu@192.168.1.1:output.tar.gz ~/your/path
```

The zipped file will now be available locally.