### Testing/demonstration

This document is actually a part of instruction. In this document readers will see how we implement our concepts into solving the problems. If they want to see the instruction, they can tend to the other document in the same folder.

Next is the second assignment:

Our group chooses the following question:

1.How can your system be upgraded in a safe manner and avoiding downtime?

2.What else would be important in a DevOps approach?

If the readers want to see the concepts involved in these questions, they can go to another file to find these concepts.

In the first task, six AWS instances are required. They are 1 assistant server, 1 database server, 3 OpenStreetMap program servers (including 1 backup), and 1 nginx server. Only assistant server needs to be setup manually. The other servers can be setup by Ansible playbook running on the assistant server. Assistant server can also be used to update program server.

In a folder called ‘Ass2\_Q2’, readers can find a folder called Shell Scripts. There are 6 shellscript files in this folder. All of these files should be rewritten in the form of a playbook. The meaning of breaking shellscript into 6 small parts is to facilitate rewriting them to the form of playbook. The rewritten playbook file was placed in a ‘Q2\_playbook.yml’ file called the Playbook folder. Also in this folder, there is a file called ‘update.yml’. It means that Update OpenStreetMap program server then run in assistant\_server. Let me explain these in detail.

We have four servers now. They are assistant\_server which contains Ansible and other tools, database\_server which is postgresql, map\_program\_server which contains map programs and nginx\_server which is Nginx.

There are four steps that set up instances:

Step 1: Use codes from ansible.sh shell script set up assistant\_server.

Step 2: Use Ansible in assistant\_server set up a database\_server and record its IP. This requires codes from database.sh(1), assistant.sh, database.sh(2).

Step 3: Use Ansible in assistant\_server set up two map\_program\_server with database\_server's IP and record the IPs of those two map\_program\_server. This requires codes from map\_server.sh.

Step 4: Use Ansible in assistant\_server set up nginx\_server and config gninx with map\_program\_server's IP. This requires codes from conf\_nginx.sh.

Step 5: Use Ansible in assistant\_server update those two map\_program\_server with their IPs. This requires codes from update.sh.

Now let us focus on the second question:

This section consists of three documents. The idea is as following. We now have an old database. The first thing to do is to back up the data in this database and then delete the original database. Finally, restore a new database with the backed up data. The file name corresponds to the respective step.