Lab 2: Hello World Baby

Node is a fast performing non-blocking I/O JavaScript run-time platform which is mostly used for web servers, API and microservices because of its small footprint, and interpreted nature. Node allows developers to port and share the front-end code *to the server* easily.

You can compare Apache httpd and PHP stack or Tomcat and Java to Node when doing this simple Hello World deployment.

Task

- Have Hello World HTTP web server written in Node working and accessible to the entire world via public AWS DNS.

Walk-through

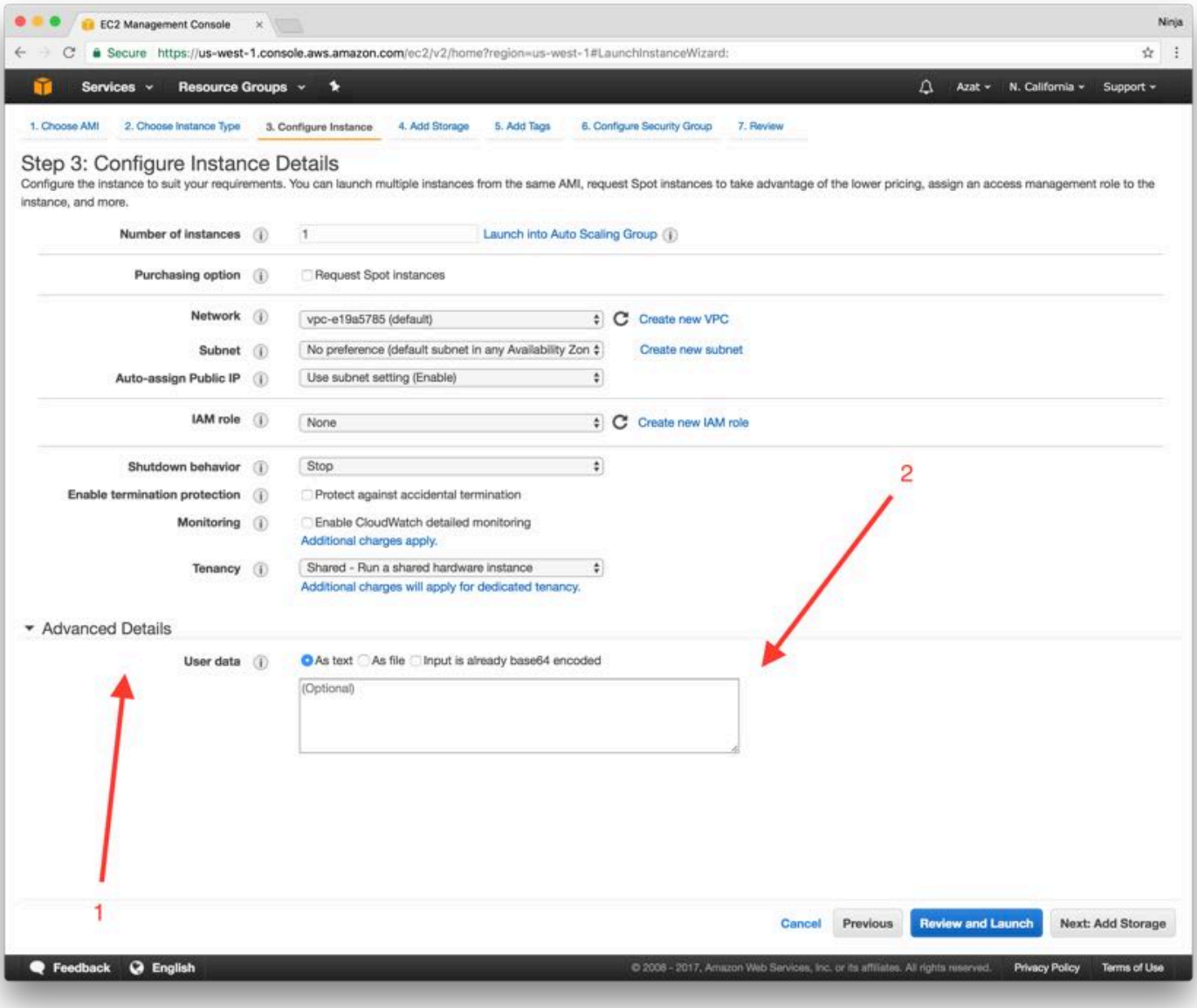
If you would like to attempt the task, then skip the walk-through and go for the task directly. However, if you need a little bit more hand holding or you would like to look up some of the commands or code or settings, then follow the walk-through.

1. Create an EC2 Instance

Log in to the web console and navigate to the EC2 dashboard. Select “Launch instance” to start the wizard. On the first screen of the instance wizard, find in Quick Start an Amazon Linux. We recommend using *“Amazon Linux 64-bit, HVM, SSD, EBS”* because it’s eligible for free tier on t2.micro.

On the next screen *Choose an instance type*, select “t2.micro” (Free tier eligible).

Next, open Advanced settings as shown on the screen capture below



Paste the following code for bash command into User Data. For bonus, change the Hello World to Hello {YOUR_NAME}. You'll need to create your own gist or write the source code in the User Data or load from somewhere else. (The second line is a better log because there's a delay in web console and cloud-init-output has too much. Below is the description of what the script does.) **Becareful when copying from a PDF - there might be unnecessary line breaks.**

```
#!/bin/bash -ex
# output user data logs into a separate place for debugging
exec > >(tee /var/log/user-data.log|logger -t user-data -s 2>/dev/console) 2>&1
# get node into yum
curl --silent --location https://rpm.nodesource.com/setup_6.x | bash -
# install node (and npm) with yum
yum -y install nodejs
# install pm2 to restart node app
npm i -g pm2@2.4.3
# get source code for hello world node server from GitHub's gist (could be private Git Hub repo or private S3)
curl "https://gist.githubusercontent.com/azat-co/5c035301e13037e52cd689205b08c121/raw/e22a4606401ce63af715792b3fe50ef869b0557f/hello-world-server.js" > /home/ec2-user/hello-world-server.js
sudo chmod 755 /home/ec2-user/hello-world-server.js # optional
# start the server (port 3000)
pm2 start /home/ec2-user/hello-world-server.js -i 0 --name "node-app"
```

It's good to restart the app when on reboot. This code uses crontab for restarting the app after rebooting the instance (or stopping and starting it). You can add this code to User Data script above:

```
# restart pm2 and thus node app on reboot
crontab -l | { cat; echo "@reboot pm2 start /home/ec2-user/hello-world-server.js -i 0 --name \"node-app\""; } | crontab -
```

Alternatively, you can use etc/rc.local, etc/init.d/rc.local, systemd, init.d, upstart, systemv, sysv or any other mechanism to ensure restart on the reboot and stop-start.

To test that the restarting works, you may add the reboot command (good for testing but not needed):

```
sudo reboot
```

Add any one of the restarting commands (crontab or any other) code to your User Data script.

The script above does many things. Here they are:

1. Install nvm (Node version manager)
2. Configure nvm to run with the "nvm" command
3. Install Node version 6.7
4. Create a source code file by using echo command
5. Install pm2 (process manager)
6. Launch pm2 with source code and an optimal number of processes (vertical scaling to maximize

CPU usage on this instance) - crontab or any other
7. Configure pm2 to start Node servers on the instance startup (for stop->start or for reboots)

Leave screens 3 and 4 with the default settings. Add tag named role with value “aws-course” on screen 5. Configure security group to have these ports open:

- HTTP 80
- HTTPS 443
- TCP/IP 3000
- SSH 22

2. Test website

Copy the public URL for the newly created EC2 instance. Paste it in the browser and **add :3000** to navigate to port 3000. You’ll see the Hello World.

Alternatively, execute curl from your developer machine:

```
curl PUBLIC_URL:3000
```

3. Test app restart

Now you need to test if the Node servers will be run on the startup coming from a stop or reboot. Go to web console and stop the instance.

Wait for a moment and see that Hello World is inaccessible from the public URL plus port number.

Start the same instance again and copy the new public URL. Add port 3000 and navigate to the address. Make sure you can see Hello World again.

Moreover, SSH to the EC2 instance and find the node process by

```
ps aux | grep node
```

You might see something like this:

```
[ec2-user@ip-172-31-2-188 ~]$ ps aux | grep 'node'
root      2668  1.8  2.6 909804 27416 ?        Sl   03:52   0:00 node /home/ec2-user/h
ello-world-server.js
ec2-user  2681  0.0  0.2 110460  2188 pts/0    S+   03:52   0:00 grep --color=auto nod
e
```

Terminate it with `kill -9 {PROCESS_ID}`, for example:

```
sudo kill -9 2668
```

Go to the web service and verify that you still see the message. That's because pm2 restarted the app. If you run `ps` again, then you'll see a different process ID.

You enabled restart not only on the instance reboot but also on the process failure which could happen due to memory leaks, bugs or attacks.

Developers can also use nvm to install Node (and add the paths to the user profile for Amazon Linux - ec2-user; for Ubuntu, it's ubuntu). This is advantageous because you can run multiple versions of Node at the same time.

```
#!/bin/bash
curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.33.1/install.sh | bash
. ~/.nvm/nvm.sh
nvm install 6.7
sudo sed -i '$i export NVM_DIR="$HOME/.nvm"' /home/ec2-user/.bashrc
sudo sed -i '$i [ -s "$NVM_DIR/nvm.sh" ] && . "$NVM_DIR/nvm.sh"' /home/ec2-user/.bashr
c
#sudo sed -i '$i export PATH=$PATH:/.nvm/versions/node/v6.7.0/bin' /home/ec2-user/.bas
hrc
```