Ricardo Deodutt

You will have to make a pipeline script with a Build, Test, and Deployment stage. After you create the script; trigger your build for every 10 minutes. Once you have successfully run a scheduled job, find a way to schedule your ec2 to shutdown by the end of class.

❖ *Screenshots at least 2 successful scheduled trigger builds*
❖ *Fork this repo 👉(https://github.com/kura-labs-org/DEPLOY02_CRON_JOB) for the blank Jenkinsfile and make a pull request with:*

https://github.com/Deodutt/DEPLOY02_CRON_JOB

For this assignment I created a new EC2 Instance.

Userdata/bootstrap script

```
#!/bin/bash
sudo yum update –y
sudo wget -O /etc/yum.repos.d/jenkins.repo \
https://pkg.jenkins.io/redhat-stable/jenkins.repo
sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key
sudo yum upgrade
sudo yum install jenkins java-1.8.0-openjdk-devel -y
sudo systemctl daemon-reload
sudo systemctl start jenkins
```

Security Group Rules
SSH | PORT 22 | My IP
HTTP | PORT 80 | custom 0.0.0.0 /0
Custom TCP Rule | 8080 | custom 0.0.0.0/0

To access ec2 in powershell

```
ssh -i .\rixardo.pem ec2-user@<IPV4address>/
```

To check if jenkins is running

```
sudo systemctl status jenkins
```

Access jenkins using
<IPV4address>:8080

To access the password

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

Install Amazon EC2 plugin in jenkins

Install Git
sudo yum install git

Once I set up Jenkins, I forked the repository.

Then create an GitHub access key using the following documentation provided by GitHub -> https://docs.github.com/en/github/authenticating-to-github/keeping-your-account-and-data-secure/creating-a-personal-access-token

Create a pipeline script with a build, test, and deployment stage in the jenkinsfile that is in forked repository..

In jenkins,

1. Create a new item
2. Name the item
3. Select pipeline
4. Scroll down to Pipeline
5. Click on Pipeline script from SCM
6. SCM = Git
7. Repository URL = the forked repository.
8. Add a new credentials
9. Click on Jenkins
10. Username = <github username>
11. Password = <access key>
12. Id = jenkins-webhook-id

Once everything is filled out, press save.
Then change the credentials to the information you filled out (In the dropdown next to add)
Repository browser (Auto)

Script path is Jenkinsfile

Then select the Build Triggers tab.
Select Build periodically.

Create a cron job syntax using -> https://crontab.guru/#*/10_*_*_*_*
TZ=America/New_York
***/10 * * * ***

Jenkins suggest to use the following CRON syntax instead. “Spread load evenly by using ‘H/10 * * * *’ rather than ‘*/10 * * * *”

TZ=America/New_York

H/10 * * * *

This basically means build at every 10th minute.

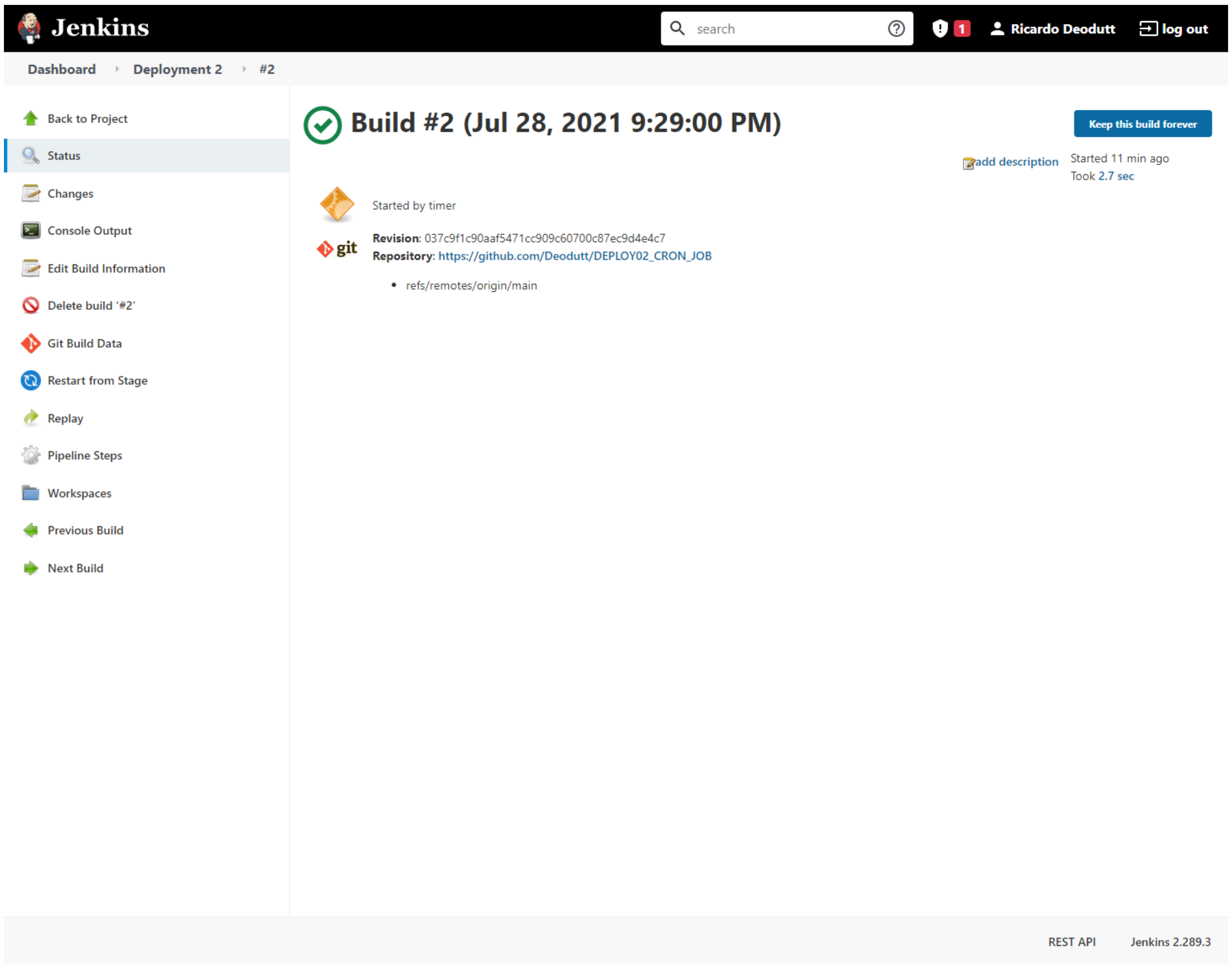
Apply and Save.

Now click build now to start.

There is another way of applying the build trigger. When creating a Multi Pipe pipeline, Attach the following under agent at the top.

```
 triggers {
      cron('H/10 * * * *')
   }
```

First Build at 9:29



Build #2 (Jul 28, 2021 9:29:00 PM)

Started by timer

Revision: 037c9f1c90aaf5471cc909c60700c87ec9d4e4c7

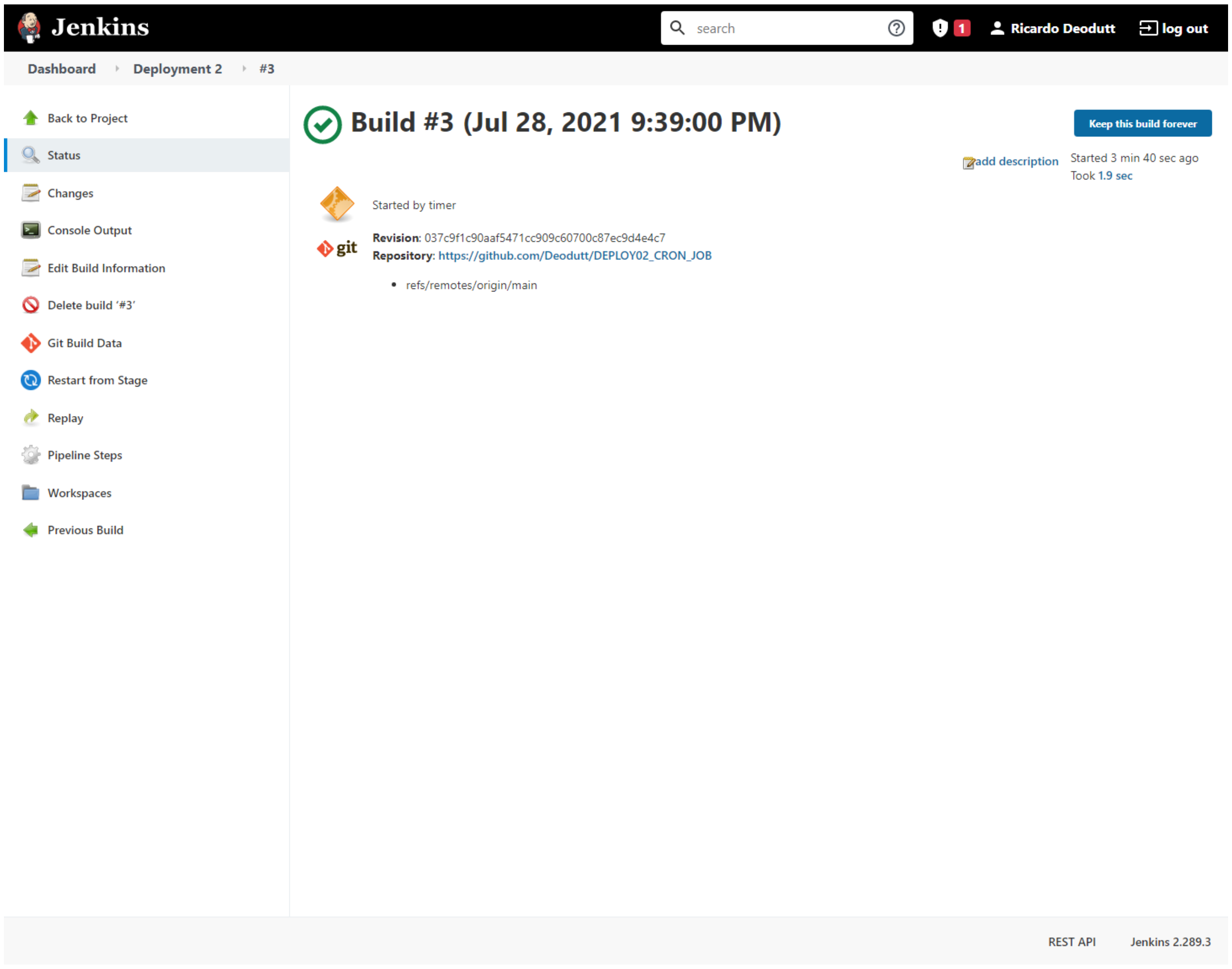
Repository: https://github.com/Deodutt/DEPLOY02_CRON_JOB

- refs/remotes/origin/main

Started 11 min ago

Took 2.7 sec

Second Build at 9:39pm



Jenkins

search

1 Ricardo Deodutt log out

Dashboard › Deployment 2 › #3

- Back to Project
- Status
- Changes
- Console Output
- Edit Build Information
- Delete build '#3'
- Git Build Data
- Restart from Stage
- Replay
- Pipeline Steps
- Workspaces
- Previous Build

Build #3 (Jul 28, 2021 9:39:00 PM)

Keep this build forever

add description

Started 3 min 40 sec ago
Took 1.9 sec

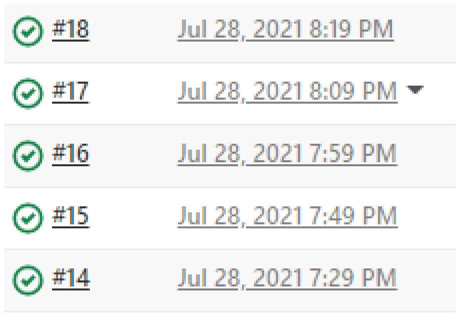
Started by timer

Revision: 037c9f1c90aaf5471cc909c60700c87ec9d4e4c7
Repository: https://github.com/Deodutt/DEPLOY02_CRON_JOB

- refs/remotes/origin/main

REST API Jenkins 2.289.3

A few successful builds later….



#18	Jul 28, 2021 8:19 PM
#17	Jul 28, 2021 8:09 PM
#16	Jul 28, 2021 7:59 PM
#15	Jul 28, 2021 7:49 PM
#14	Jul 28, 2021 7:29 PM

Find a way to schedule your ec2 to shutdown by the end of class.

I wanted to solve this problem by using material that we learnt in class. I thought the only way I could do that is by creating a bash script. So I did some research on certain topics that will help me solve this problem.

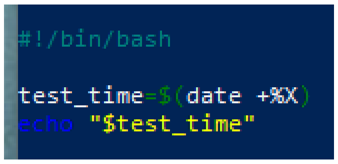
Using date command ->
https://stackoverflow.com/questions/18458839/how-can-i-get-the-current-date-and-time-in-the-terminal-and-set-a-custom-command

And Running the script in the background
https://askubuntu.com/questions/88091/how-to-run-a-shell-script-in-background

And then shut down the instance when the date is the same as the required date.
https://dev.to/aws/auto-stop-ec2-instances-when-they-finish-a-task-2f0i
sudo shutdown now

So the main purpose of this bash script is I want to have two variables. One variable is currentTime and the second variable is shutdownTime.

I first needed to test how to store a variable with the date command and echo it out.

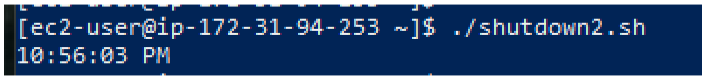


```
#!/bin/bash

test_time=$(date +%X)
echo "$test_time"
```

I had a problem using the variable test_time = $(date +$X) because apparently spacing matters in bash scripting…..

The timing is a little bit off.



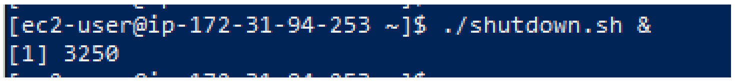
```
[ec2-user@ip-172-31-94-253 ~]$ ./shutdown2.sh
10:56:03 PM
```

The date command is used to display the system date and time. It uses the time zone set by the OS. By default, Amazon uses UTC time zone.

Also, whenever I code, I like to comment things out and come back at it. I figured out I can do this using # at the start of the line. This speeds things up when I debug the script.

sudo shutdown now -h
Is used to shutdown the instance that you are currently connected to, instantly.

To run a bash script in the background, run your command and add “&” at the end of it.



```
[ec2-user@ip-172-31-94-253 ~]$ ./shutdown.sh &
[1] 3250
```

This basically creates a process in the background.

To kill a process use…
kill – Kill a process by ID
killall – Kill a process by name

So now I have to fix the timing of the instance, so it matches EST.
https://stackoverflow.com/questions/11931566/how-to-set-the-time-zone-in-amazon-ec2/11931955

So you have to identify a time zone to use on the instance.
ls /usr/share/zoneinfo/America/New_York

Then you have to update the /etc/sysconfig/clock file with the new time zone
sudo open the /etc/sysconfig/clock with a text editor.
sudo nano /etc/sysconfig/clock

Locate the zone entry and change it to the time zone.
America/New_York

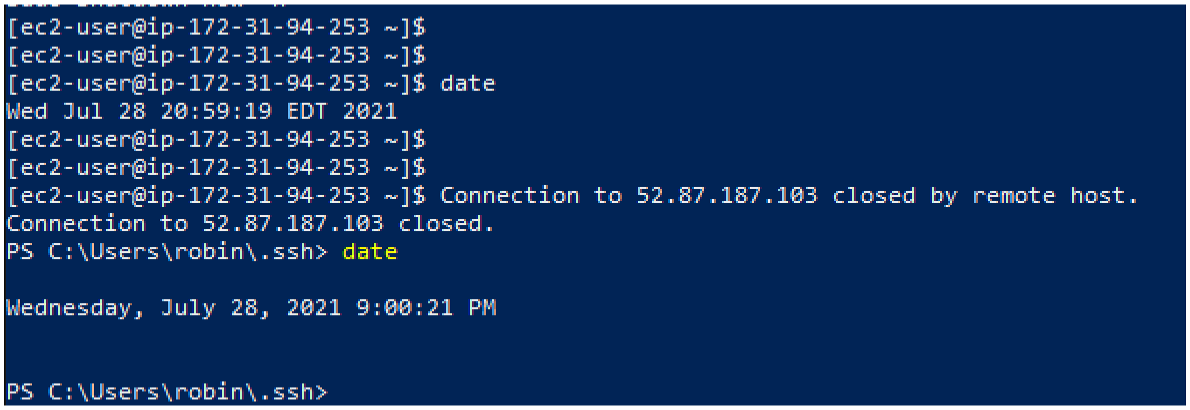
Create a symbolic link between /etc/localtime and your time zone file.
This is so the instance finds the time zone when it references local time information.
sudo ln -sf /usr/share/zoneinfo/America/New_York /etc/localtime

Once time has been changed, simply edit the bash script and put the correct timing which is **09:00:00 PM** which is equivalent to 9pm in eastern time.
Once you save everything, run the bash script in the background using ./shutdown.sh &

So I noticed that it's a **hassle** trying to change the amazon default time zone each time I create a new instance. So I did some research and asked some people who have a foot in the field and they recommend sticking with the UTC time zone.

To test if the command works, just change the shutdownTime variable to a minute ahead the current real life time and run the command in the background.

Everytime you shut down the ec2 instance, you can start it back up on aws website. It takes a couple of minutes to start sometimes.



```
[ec2-user@ip-172-31-94-253 ~]$
[ec2-user@ip-172-31-94-253 ~]$
[ec2-user@ip-172-31-94-253 ~]$ date
Wed Jul 28 20:59:19 EDT 2021
[ec2-user@ip-172-31-94-253 ~]$
[ec2-user@ip-172-31-94-253 ~]$
[ec2-user@ip-172-31-94-253 ~]$ Connection to 52.87.187.103 closed by remote host.
Connection to 52.87.187.103 closed.
PS C:\Users\robin\.ssh> date

Wednesday, July 28, 2021 9:00:21 PM


PS C:\Users\robin\.ssh>
```

This method is not very efficient because it may use up a lot of resources.
I am currently working on another method.