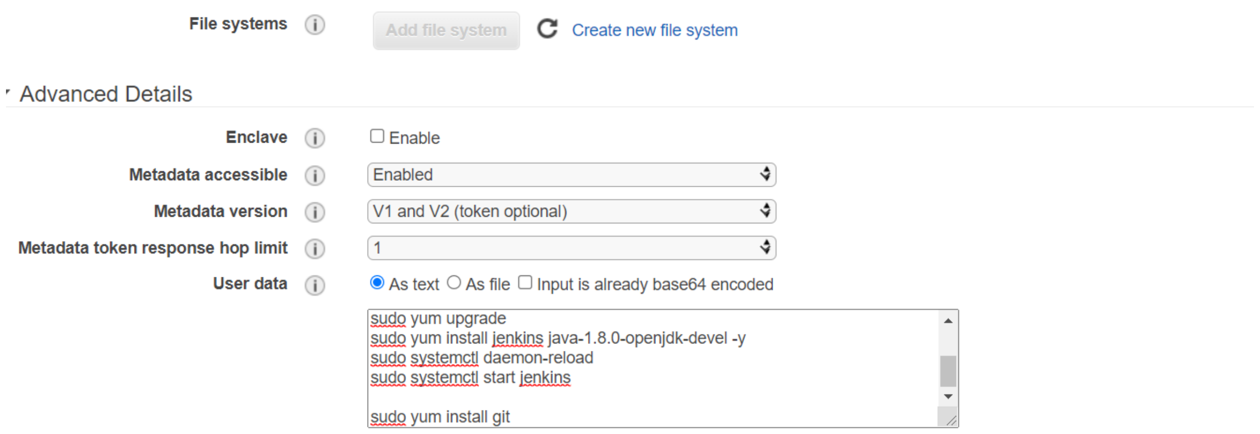
1. Create an IAM user and call the user “Jenkins-user”.
2. Then give the user Administrator ElasticBeanStalk access.
3. Once the permissions have been added, copy and save and Jenkins-user Access Key and Secret Key by downloading the “.csv” file and saving it somewhere safe you’ll remember.
4. Create an Elastic Beanstalk by entering the name, key, value, platform and application code.
5. Create a new EC2 Amazon Linux instance using the following bootstrap script as shown below.

```
#!/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

sudo yum install git -y
```

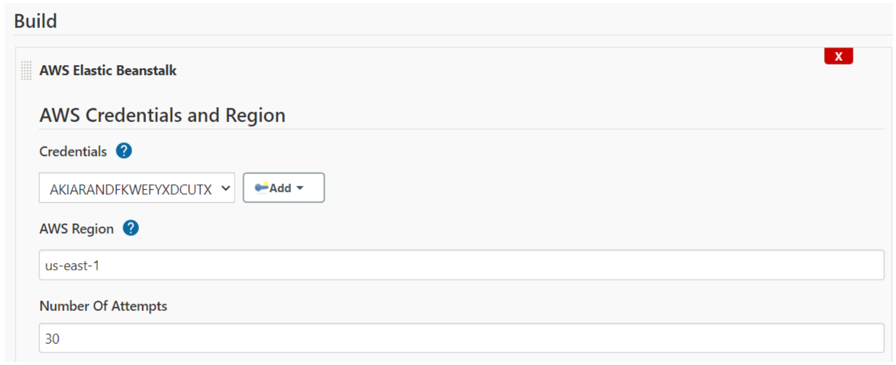


Enclave	☐ Enable
Metadata accessible	Enabled
Metadata version	V1 and V2 (token optional)
Metadata token response hop limit	1
User data	● As text ○ As file ☐ Input is already base64 encoded

```
sudo yum upgrade
sudo yum install jenkins java-1.8.0-openjdk-devel -y
sudo systemctl daemon-reload
sudo systemctl start jenkins

sudo yum install git
```

6. For security groups settings, set SSH port of 22 (My ip), a Custom TCP Rule of 8080 and HTTP port of 80.
7. Then launch your EC2 instance
8. After launching ssh into your instance, go to your terminal and enter the directory which contains your SSH keys and do the following:
   cd (directory where your key is located)
   ssh -i (EC2 keys) ec2-user@(public IPV4 address of your instance)
9. Check that Jenkins is running by using the command sudo systemctl status Jenkins.
10. Once Jenkins is running, enter your browser and enter the public IPV4 address of your instance with port 8080. Example 3.54.67.256:8080

11. Then after Jenkins loads, use the command sudo cat /var/lib/jenkins/secrets/initialAdminPassword to get that password you'll need to enter.
12. Paste the password that you got from previous step.
13. Now install suggested plugins for Jenkins.
14. Setup up an admin account with your username, password and email.
15. Go to Jenkins plugins and install both AWSEB Deployment and CloudBees Credentials plugins.
16. Fork the DEPLOY04_FLASK_APP repository.
17. Then created a freestyle project after giving it a name
18. Then add the link to the git repository in the git source management part of your configurations.
19. Then add your credentials into Cloudbee as shown in the figure below:



Build

AWS Elastic Beanstalk

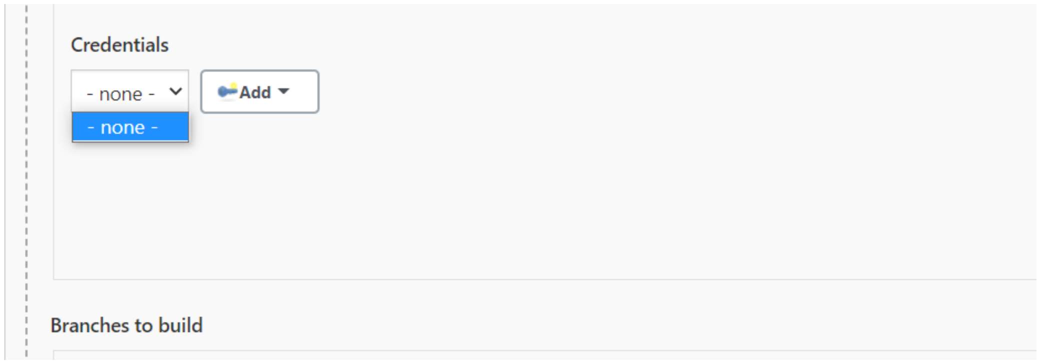
AWS Credentials and Region

Credentials: AKIARANDFKWEFYXDCUTX

AWS Region: us-east-1

Number Of Attempts: 30

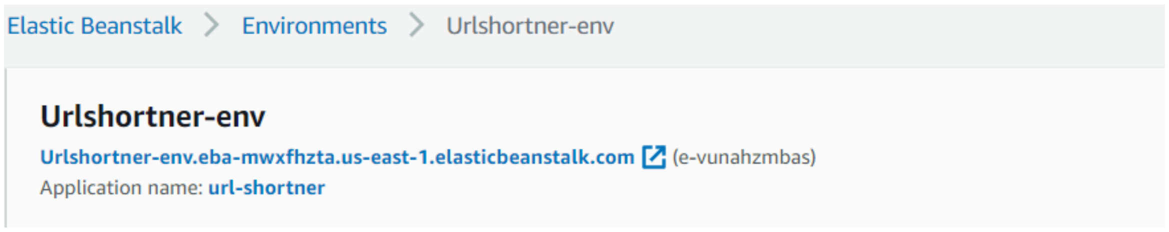
NOTE: if the credentials are done in the source code management area instead of cloudbees it may not show when you try to add it as shown below:



Credentials: - none -

Branches to build

20. Then fill in the application and environment's names.
21. Then in the packaging section, enter a period (.) in the Root Object Field.
22. After, fill in the Version Label Format with "python-01${BUILD_ID}.
23. Once done save changes and then start the build to deploy the application to AWSEB.
24. Once successful check your AWSEB to see if it loads correctly as shown below.

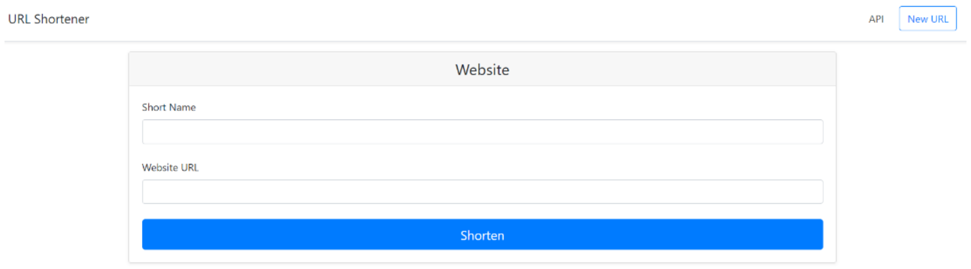

Elastic Beanstalk > Environments > Urlshortner-env

Urlshortner-env

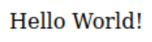
Urlshortner-env.eba-mwxfhzta.us-east-1.elasticbeanstalk.com (e-vunahzmbas)

Application name: url-shortner

25. Once the link was made the “url-shortner” was shown as seen blow:



26. Do did the same for the flask app made as shown below:

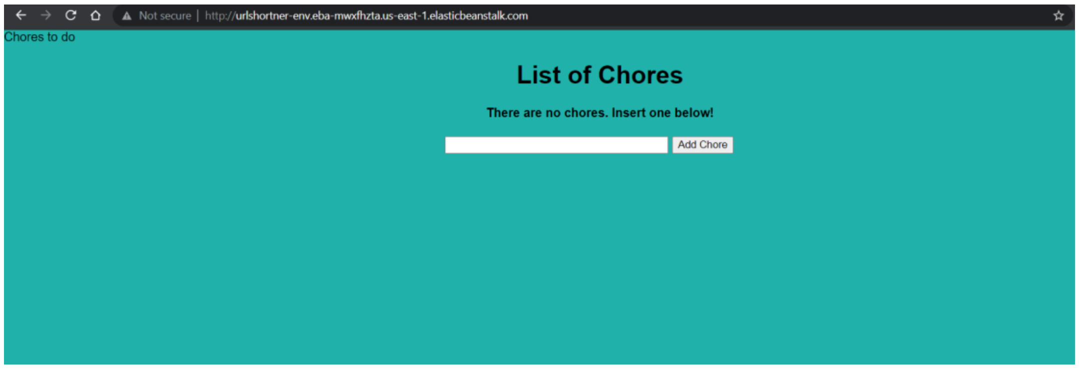

Hello World!

Chore-App
My Todo Flask App

27. When creating my custom flask app I created a virtual environment and did so using the command (`$ pip install virtualenv`).
28. Then used the command ($ virtualenv env) which allowed me to install and use anything in the given environment.

NOTE: This is important because in a large group all the group, teammates would just need to install the env to be able to access whatever was installed and used.

29. Then entered the environment using the command ($ source env/bin/activate).
30. After I ran the command ($ pip3 install flask flask-sqlalchemy). This allowed me to create a database which I used later.
31. Once completed I deployed it in Jenkins following steps from 17-24 and opened my elastic beanstalk. The results are shown below:

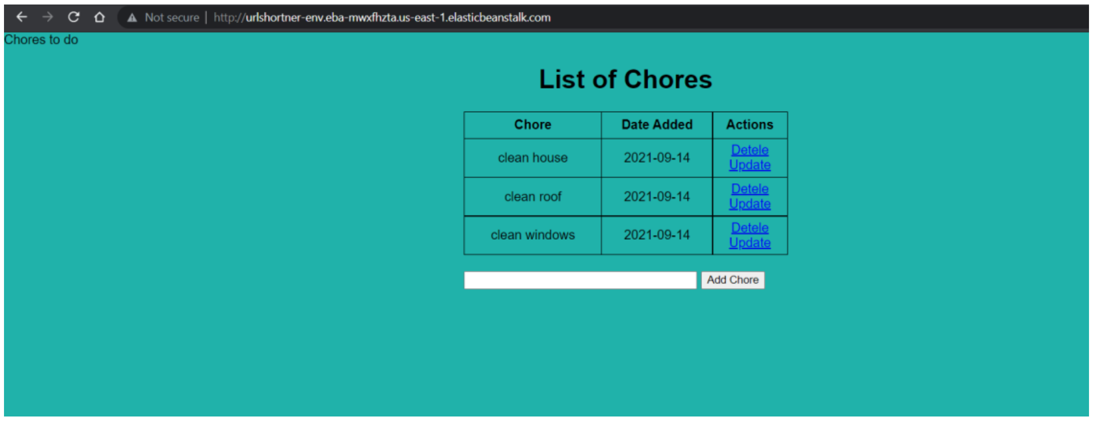

Not secure | http://urlshortner-env.eba-mwxfhzta.us-east-1.elasticbeanstalk.com
Add Chore

Chores to do

List of Chores

There are no chores. Insert one below!

Entering chores

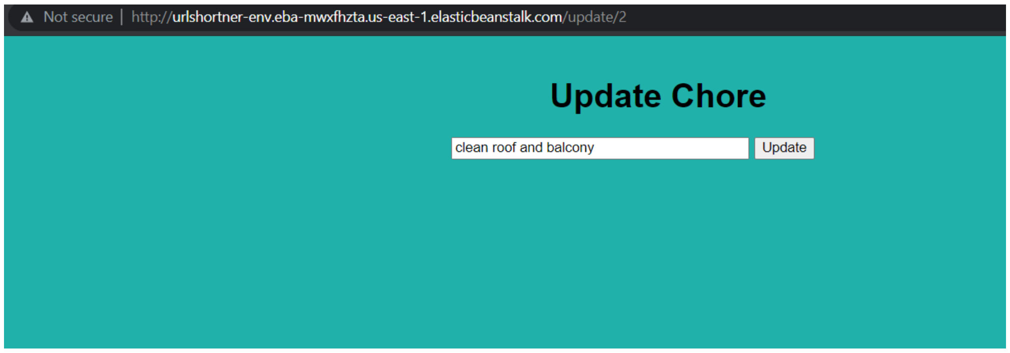

Not secure | http://urlshortner-env.eba-mwxfhzta.us-east-1.elasticbeanstalk.com
Add Chore

Chores to do

List of Chores

Chore	Date Added	Actions
clean house	2021-09-14	Detele Update
clean roof	2021-09-14	Detele Update
clean windows	2021-09-14	Detele Update

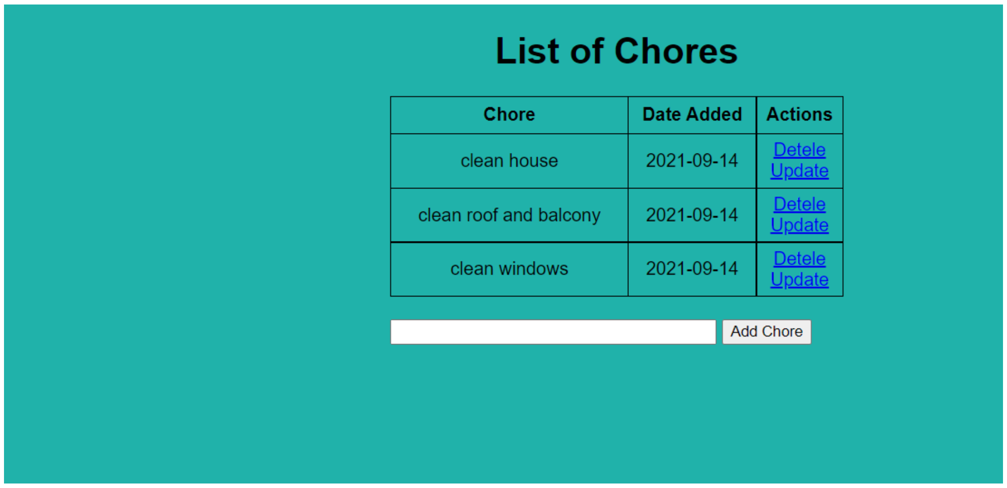
Updating chores



Not secure | http://urlshortner-env.eba-mwxfhzta.us-east-1.elasticbeanstalk.com/update/2

Update Chore

clean roof and balcony | Update

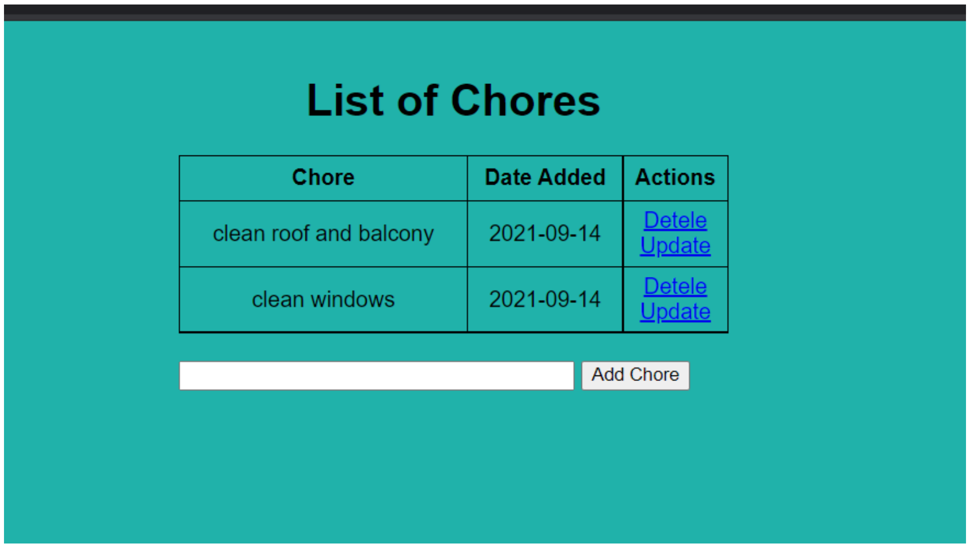


List of Chores

Chore	Date Added	Actions
clean house	2021-09-14	Detele Update
clean roof and balcony	2021-09-14	Detele Update
clean windows	2021-09-14	Detele Update

Add Chore

Deleted Chore



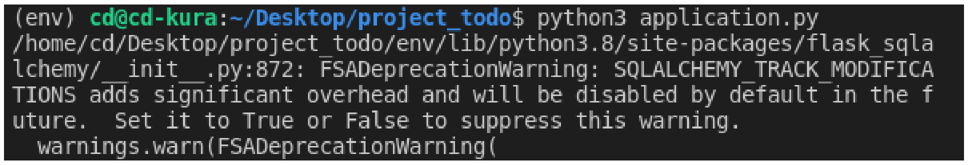
List of Chores

Chore	Date Added	Actions
clean roof and balcony	2021-09-14	Detele Update
clean windows	2021-09-14	Detele Update

Add Chore

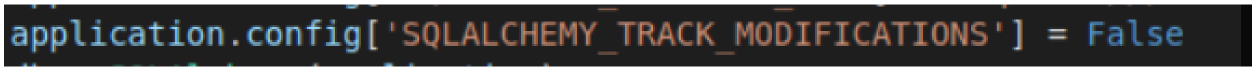
Problems:

1. Used the command ($ python3 application.py) to run my flask app. This ran successfully for a moment and allowed me to refresh my localhost and instantly see the changes without stopping my flask app each time and restarting it. However, I got the error shown below after a certain point of testing.

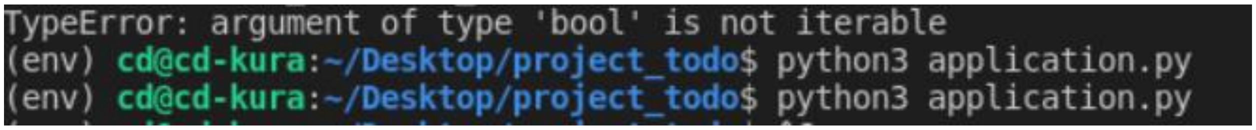


```
(env) cd@cd-kura:~/Desktop/project_todo$ python3 application.py
/home/cd/Desktop/project_todo/env/lib/python3.8/site-packages/flask_sqla
lchemy/__init__.py:872: FSADeprecationWarning: SQLALCHEMY_TRACK_MODIFICA
TIONS adds significant overhead and will be disabled by default in the f
uture.  Set it to True or False to suppress this warning.
  warnings.warn(FSADeprecationWarning(
```

NOTE: although I found the following solution, when used it didn't work.



```
application.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
```

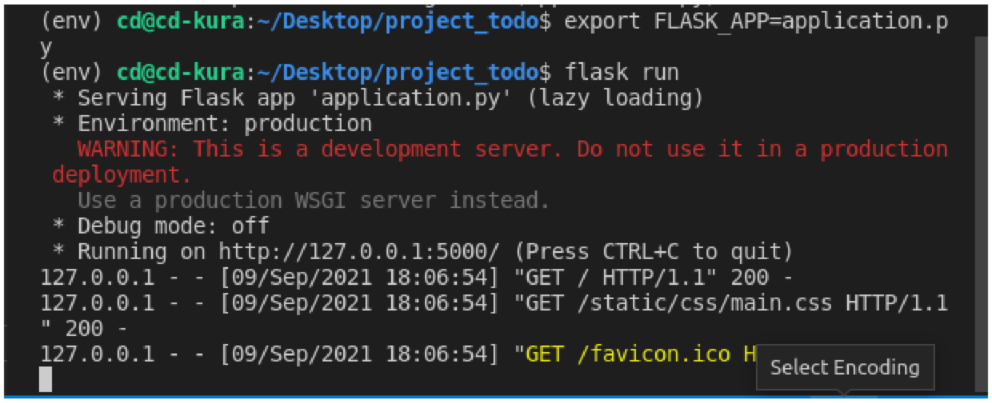


```
TypeError: argument of type 'bool' is not iterable
(env) cd@cd-kura:~/Desktop/project_todo$ python3 application.py
(env) cd@cd-kura:~/Desktop/project_todo$ python3 application.py
```

2. Used the wrong Environment name when deploying my custom app although I got a successful build, my build didn't go to my elastic beanstalk.

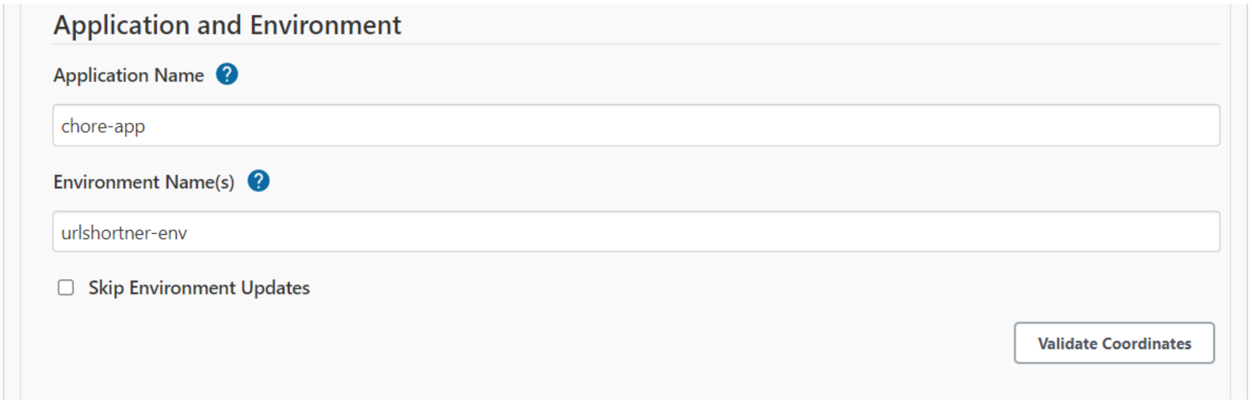
Solutions

1. Fixed my first problem using the following commands:



```
(env) cd@cd-kura:~/Desktop/project_todo$ export FLASK_APP=application.p
y
(env) cd@cd-kura:~/Desktop/project_todo$ flask run
 * Serving Flask app 'application.py' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production
 deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [09/Sep/2021 18:06:54] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [09/Sep/2021 18:06:54] "GET /static/css/main.css HTTP/1.1
" 200 -
127.0.0.1 - - [09/Sep/2021 18:06:54] "GET /favicon.ico H
```

2. Fixed my second problem by correcting and changing my path as shown below:



Application and Environment

Application Name

chore-app

Environment Name(s)

urlshortner-env

☐ Skip Environment Updates