DevOps Essential

Aman Shrestha
Associate Software Engineer

1. Launch an EC2 instance and login via ssh furthermore, install httpd (apache).

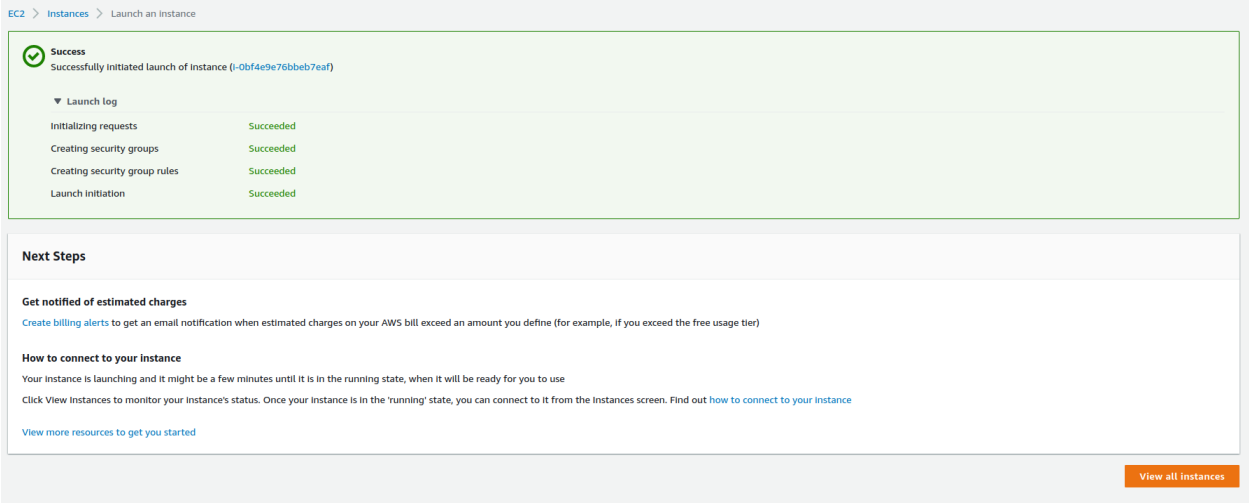


Fig: Successful EC2 instance launch

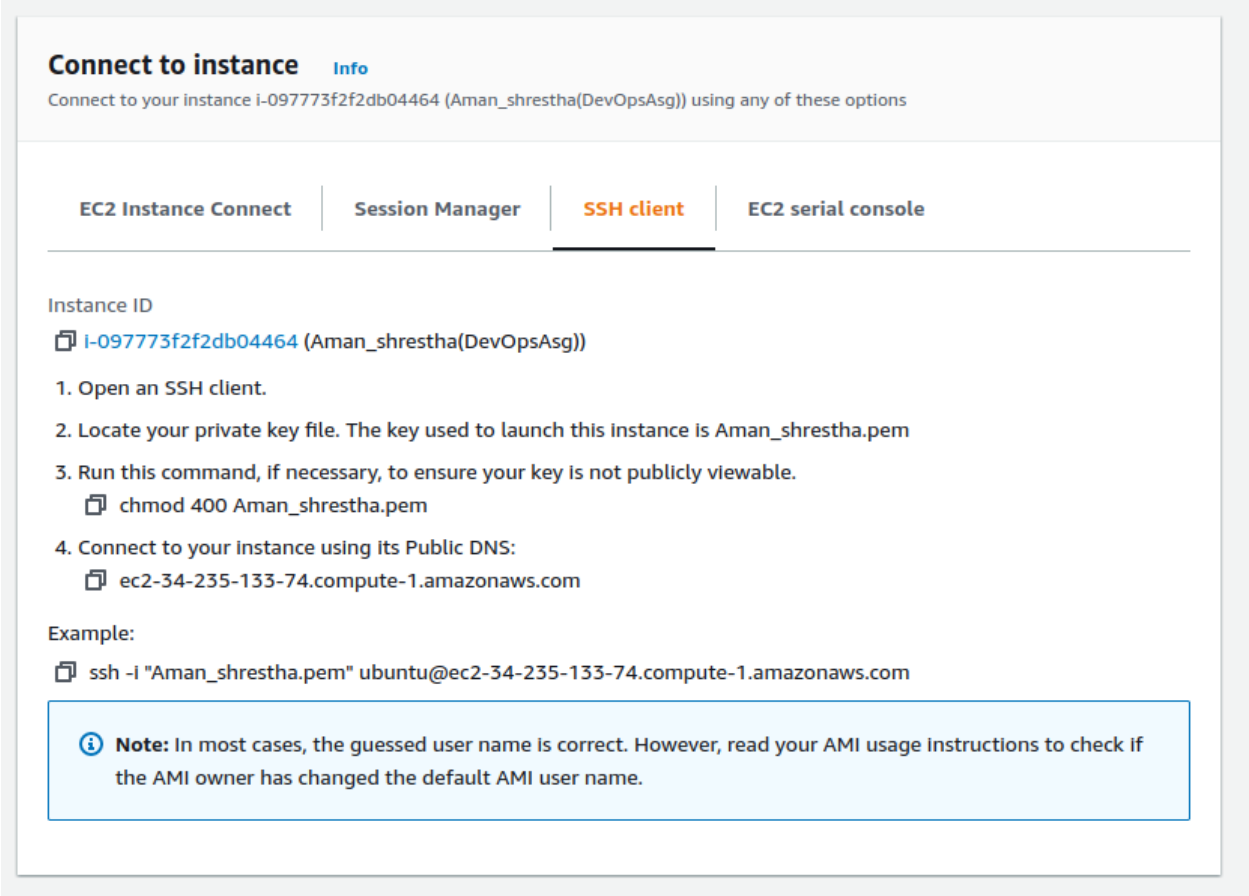


Fig: Connection instruction to an EC2 instance

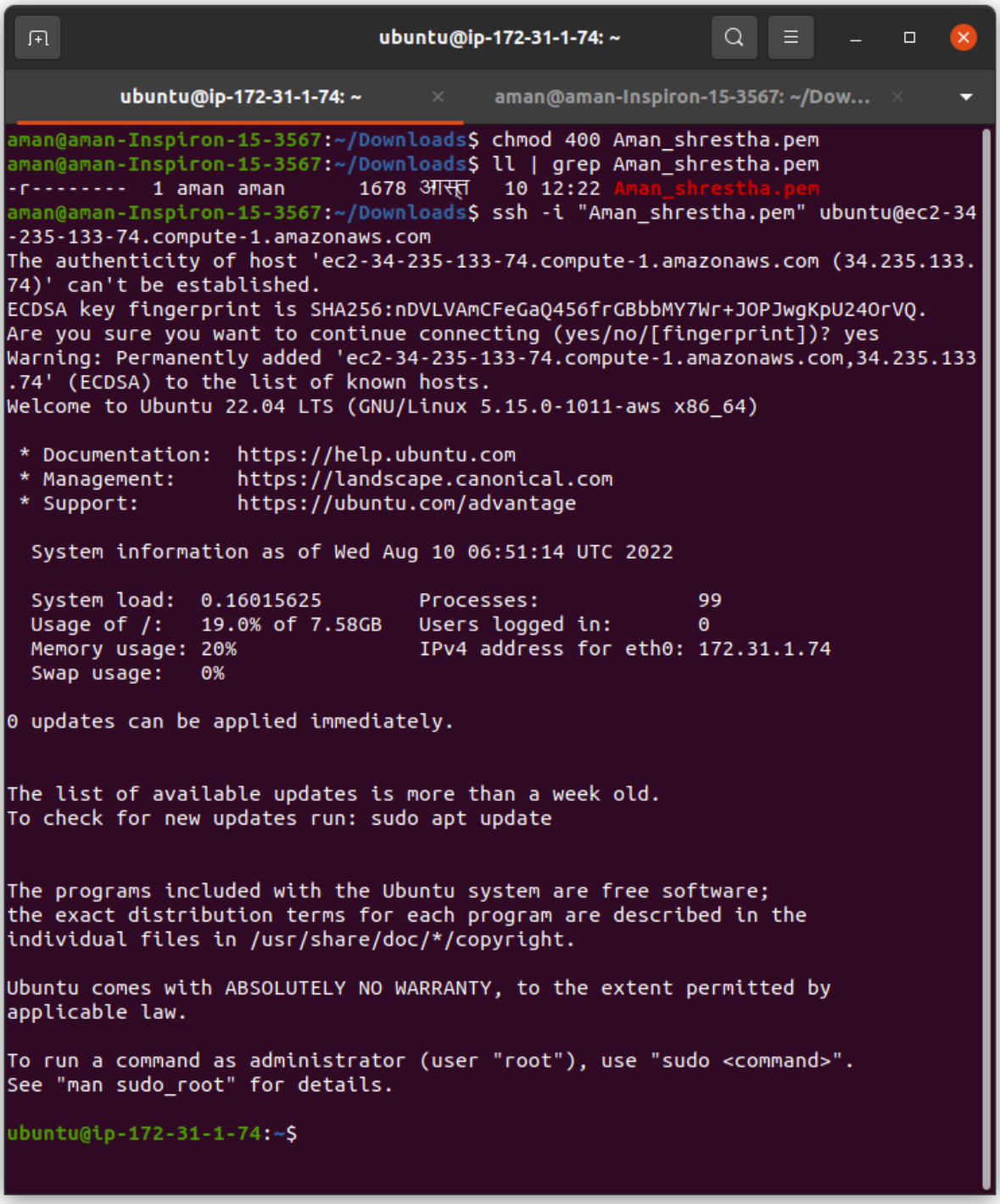


Fig: Logging to EC2 instance from SSH

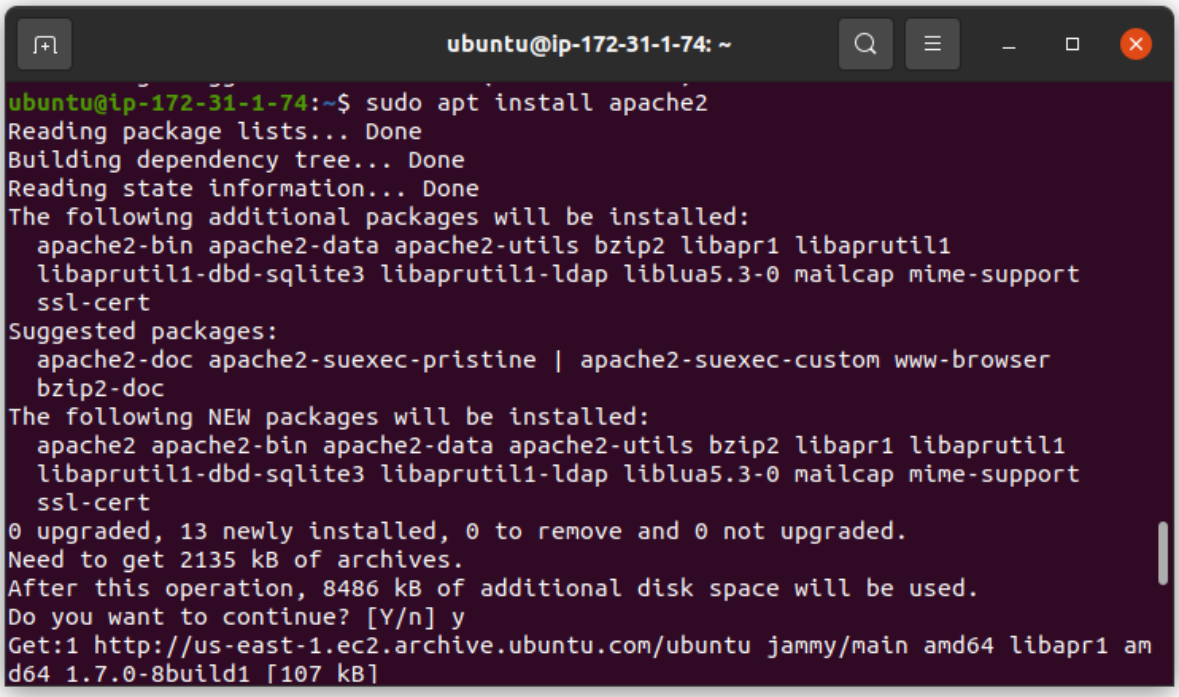


Fig: Installation of httpd (Apache)

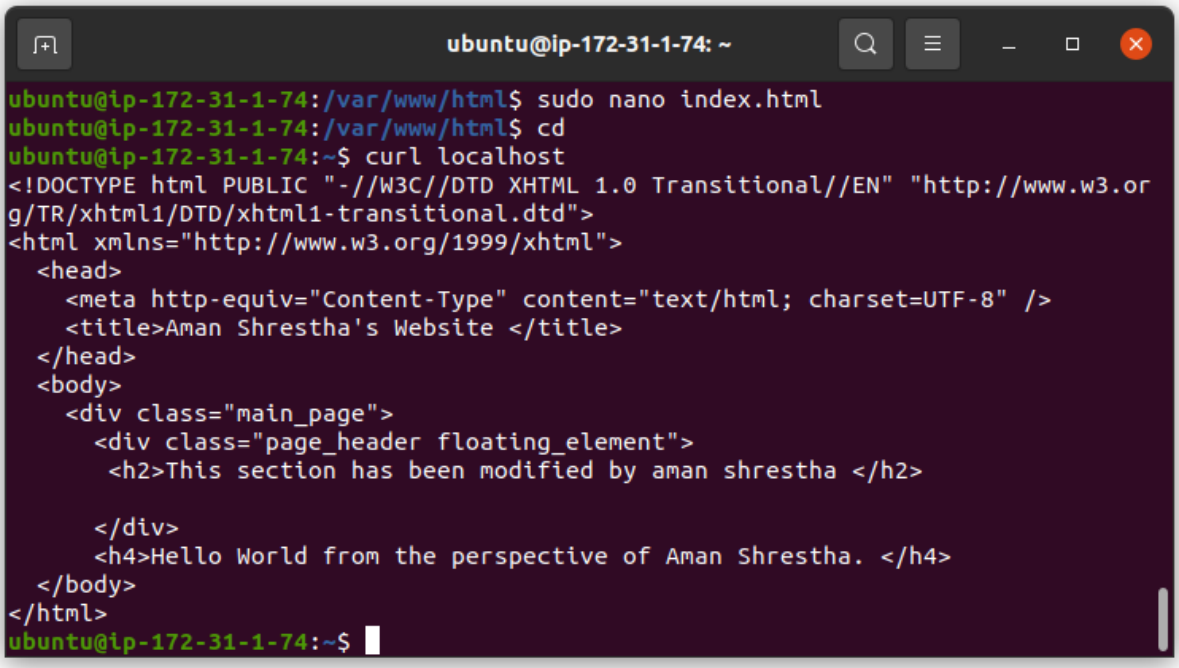


Fig: Changing default APACHE server response

2. Create a Dockerfile for node js and react and push the created docker images to the docker hub repository.

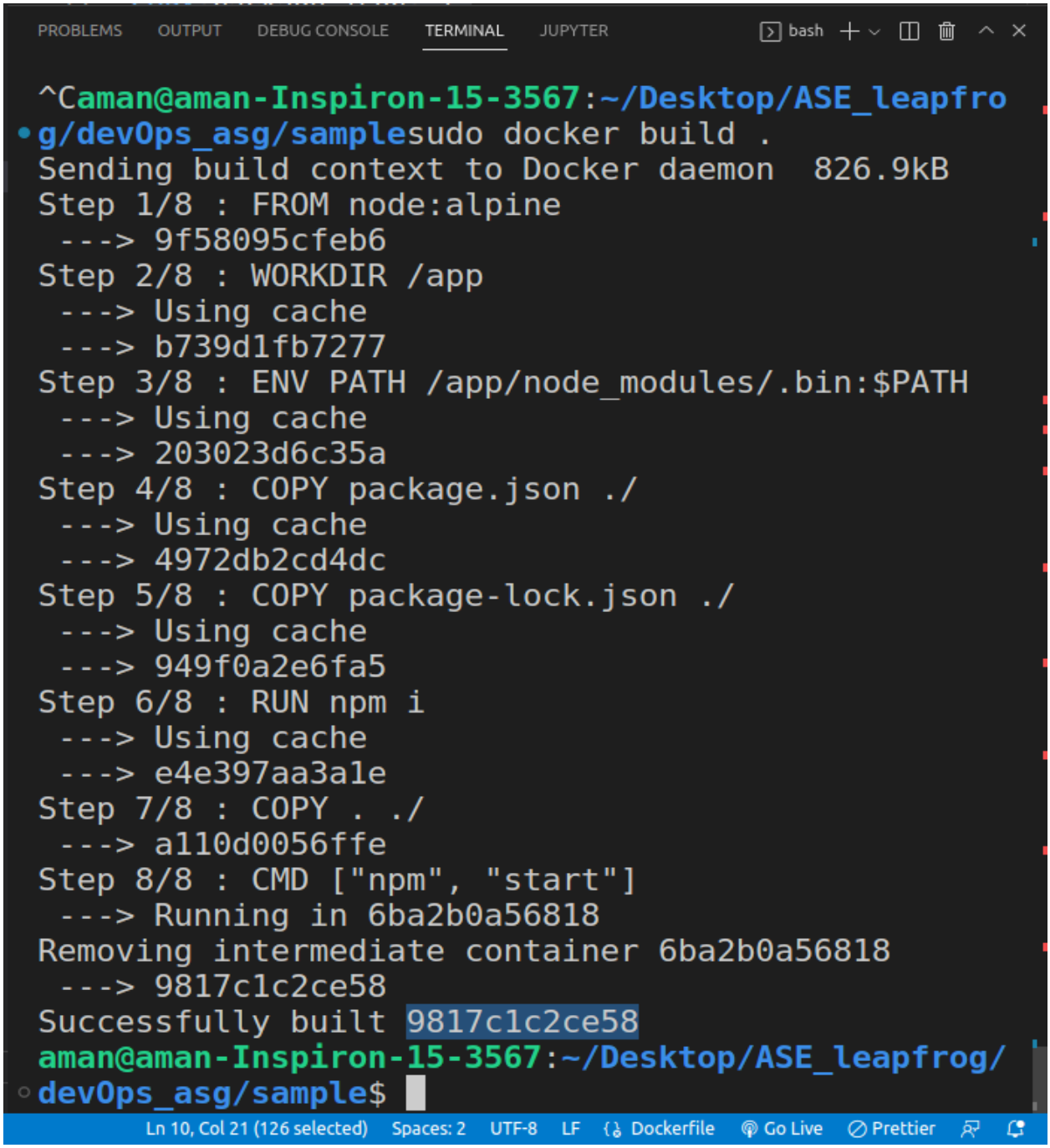


Fig: Docker image build

The above command searches for Dockerfile and builds an image for it returning image ID.

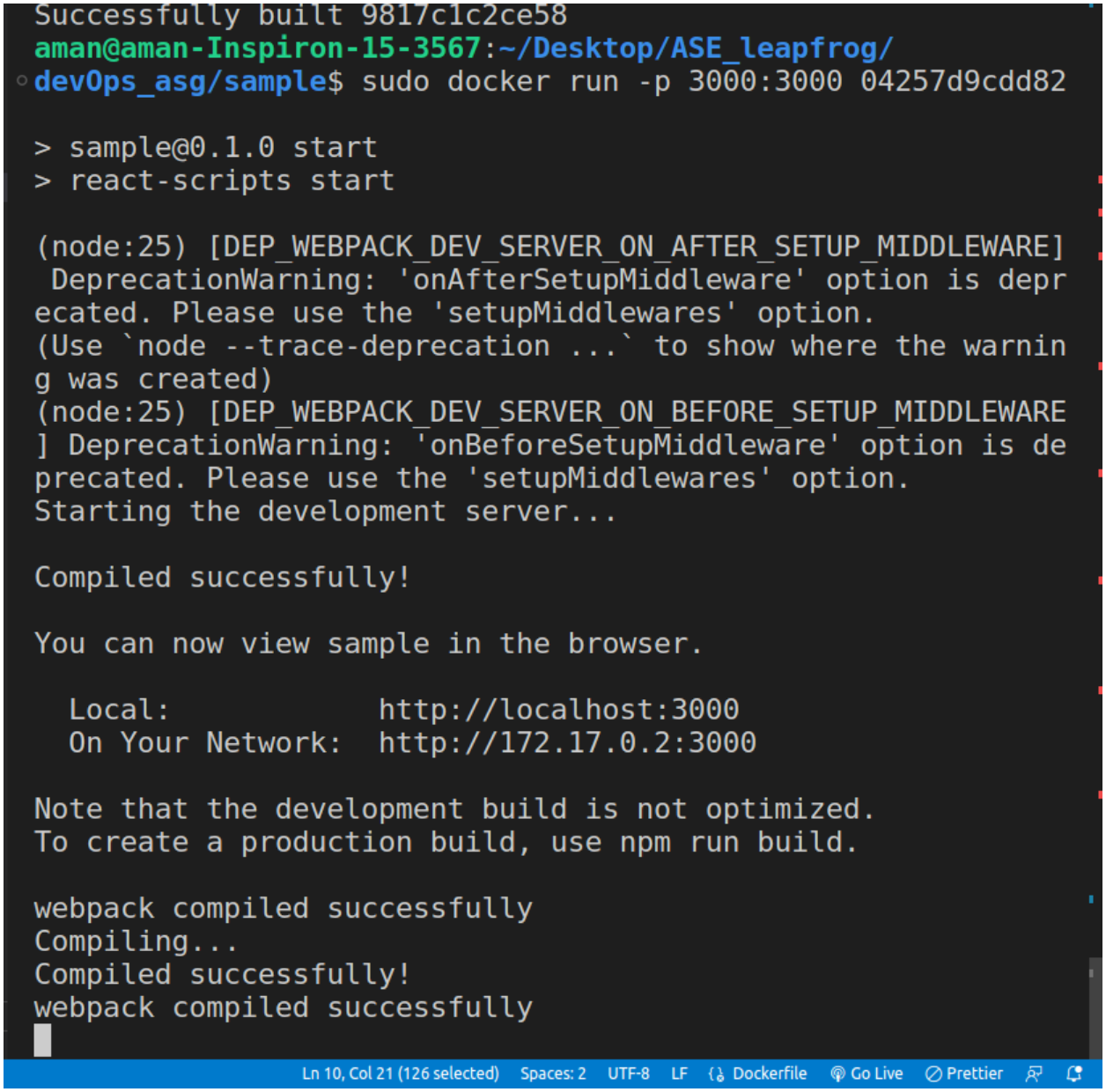


Fig: Run container using docker image at container port 3000

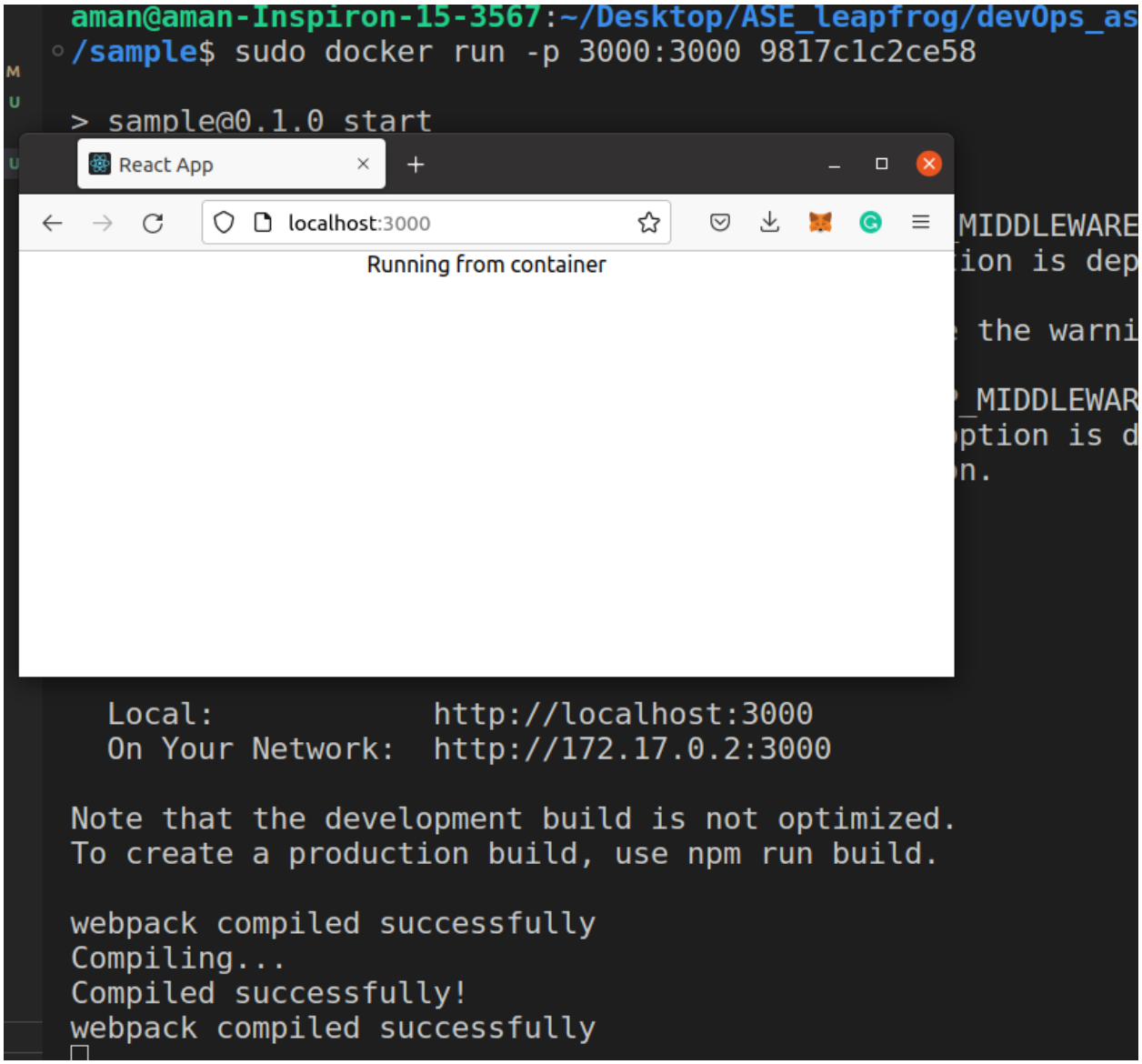


Fig: Running react app from container

The above snaps show we were successfully able to create an image and run it from the container itself in our local machine. Time to take it to the next level (Push it to docker hub). Let's go….

```
aman@aman-Inspiron-15-3567:~/Desktop/ASE_leapfrog/devOps_asg
/sample$ sudo docker login
Authenticating with existing credentials...
WARNING! Your password will be stored unencrypted in /root/.
docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#
credentials-store

Login Succeeded
aman@aman-Inspiron-15-3567:~/Desktop/ASE_leapfrog/devOps_asg
```

Fig: DockerHub login

```
aman@aman-Inspiron-15-3567:~/Desktop/ASE_leapfrog/devOps_asg
/sample$ sudo docker build -t sample_app .
Sending build context to Docker daemon  826.9kB
Step 1/8 : FROM node:alpine
 ---> 9f58095cfeb6
Step 2/8 : WORKDIR /app
 ---> Using cache
 ---> b739d1fb7277
Step 3/8 : ENV PATH /app/node_modules/.bin:$PATH
 ---> Using cache
 ---> 203023d6c35a
Step 4/8 : COPY package.json ./
 ---> Using cache
 ---> 4972db2cd4dc
Step 5/8 : COPY package-lock.json ./
 ---> Using cache
 ---> 949f0a2e6fa5
Step 6/8 : RUN npm i
 ---> Using cache
 ---> e4e397aa3a1e
Step 7/8 : COPY . ./
 ---> Using cache
 ---> a110d0056ffe
Step 8/8 : CMD ["npm", "start"]
 ---> Using cache
 ---> 9817c1c2ce58
Successfully built 9817c1c2ce58
Successfully tagged sample_app:latest
aman@aman-Inspiron-15-3567:~/Desktop/ASE_leapfrog/devOps_asg
/sample$
Ln 10, Col 21 (126 selected)  Spaces: 2  UTF-8  LF  Dockerfile  Go Live  Prettier
```

Fig: Tagging docker image as sample_app

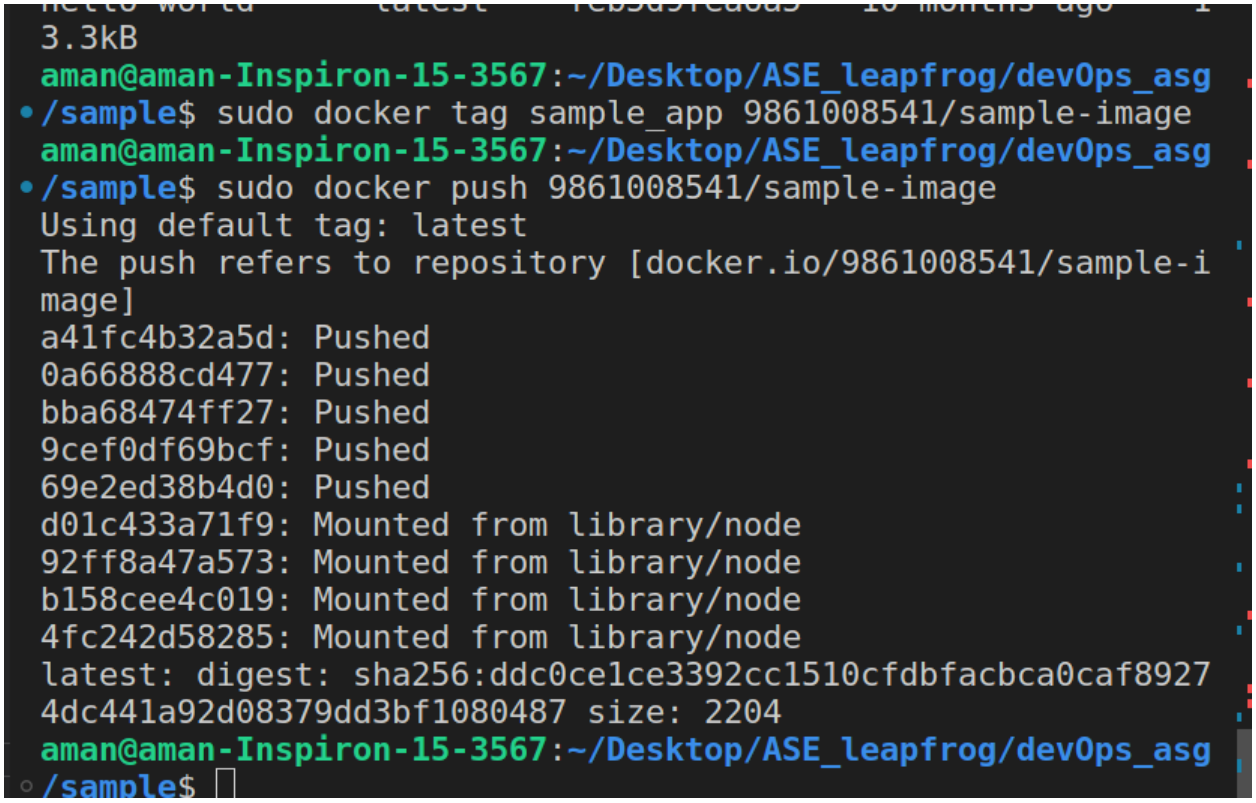


Fig: Pushing Image into DockerHub

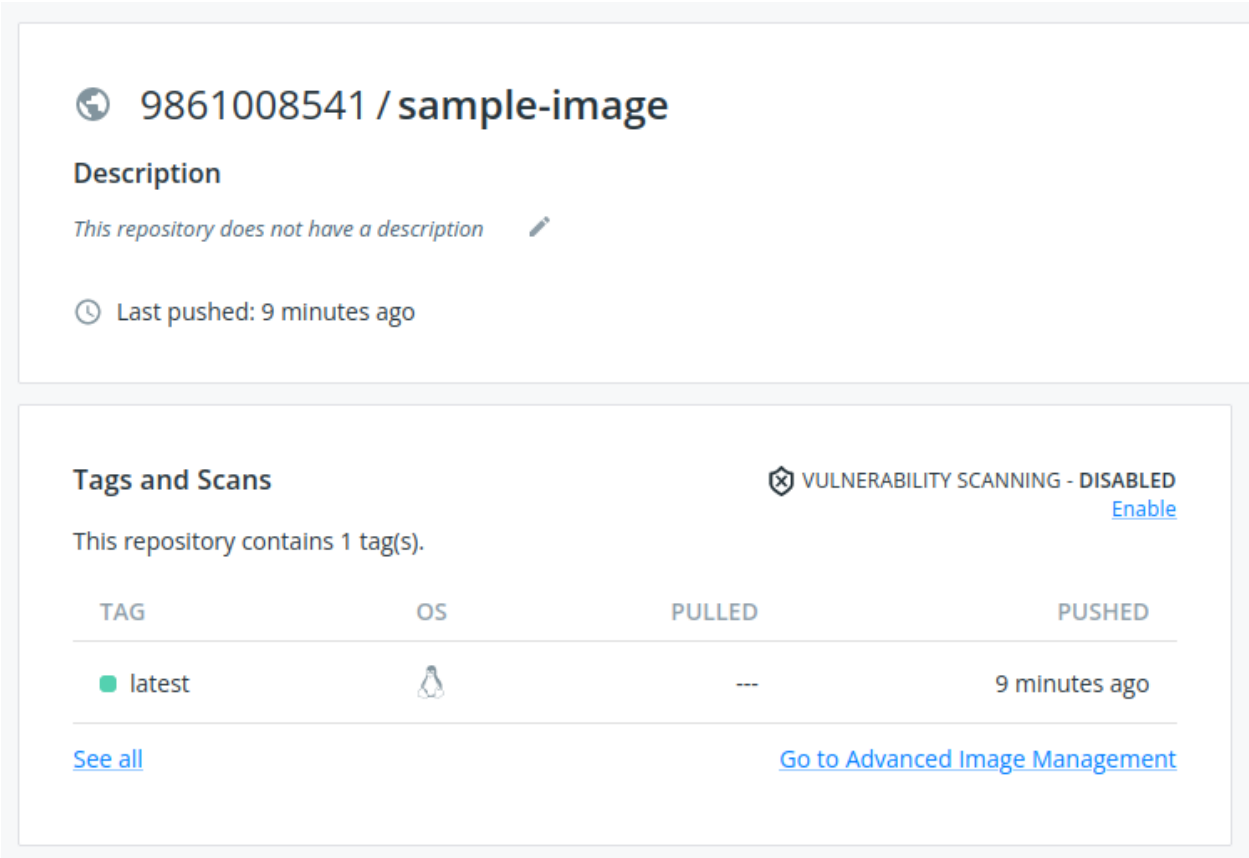
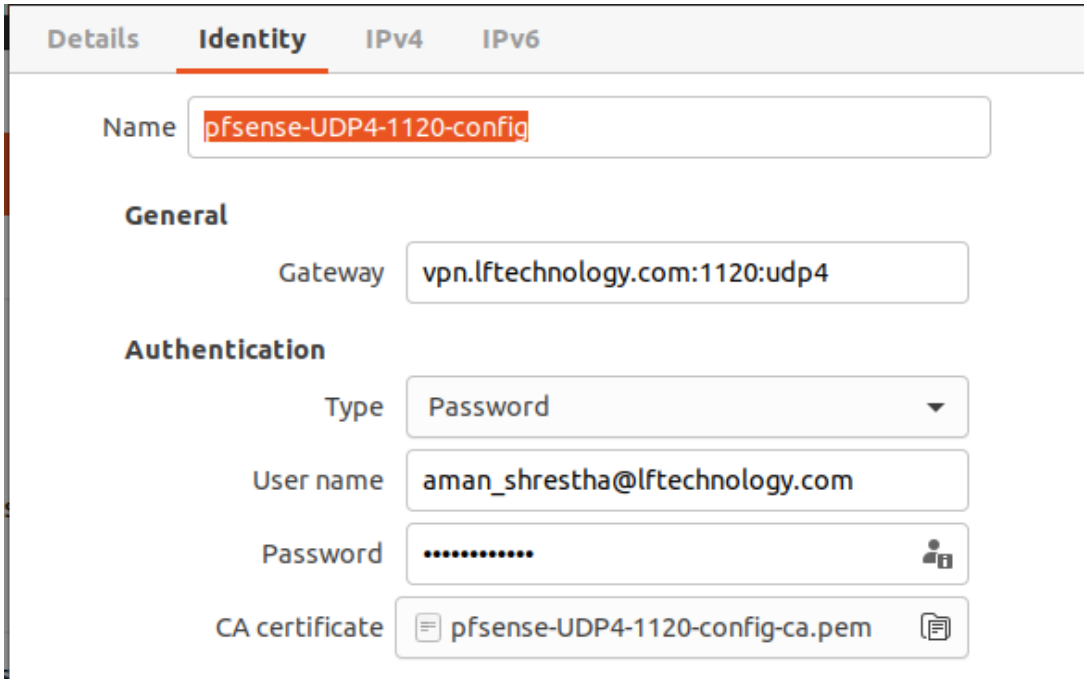
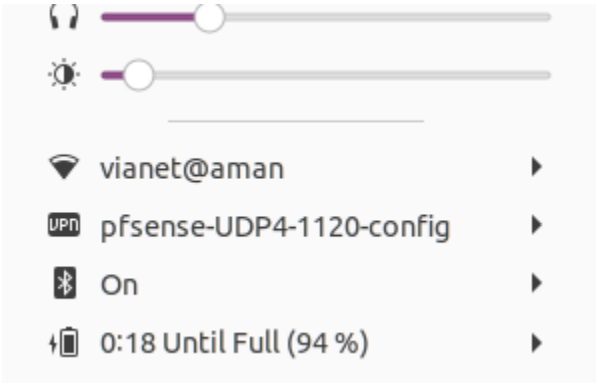


Fig: Output in docker hub

We can see the newly created repository in the docker hub was pushed a docker image of our application.

3. Setup office VPN and access self-learning platforms.





The IPV4 address of 182.93.83.51 confirms connectivity to the office VPN.

4. List down common HTTP status codes(only 10) with their meaning.

1. 100 Continue
   Only a part of the request has been received by the server, but as long as it has not been rejected, the client should continue with the request.

2. 200 OK
   The request succeeded. The result meaning of "success" depends on the HTTP method (GET, HEAD, PUT, POST)

3. 201 Created
   The request succeeded, and a new resource was created as a result. This is typically the response sent after POST requests or some PUT requests.

4. 301 Moved Permanently
   The URL of the requested resource has been changed permanently. The new URL is given in the response

5. 400 Bad Request
   The server cannot or will not process the request due to something that is perceived to be a client error (e.g., malformed request syntax, invalid request message framing, or deceptive request routing).

6. 401 Unauthorized
   The clients are not authenticated to access the resources.

7. 403 Forbidden
   The client does not have access rights to the content; that is, it is unauthorized, so the server is refusing to give the requested resource. Unlike 401 Unauthorized, the client's identity is known to the server.

8. 404 Not Found

The server can not find the requested resource. In the browser, this means the URL is not recognized. In an API, this can also mean that the endpoint is valid but the resource itself does not exist.

9. 405 Method Not Allowed
   The request method is known by the server but is not supported by the target resource. For example, an API may not allow calling DELETE to remove a resource.

10. 500 Internal Server Error
    The request was not completed. The server met an unexpected condition.

5. Find network id and Broadcast for 150.10.20.30

Address Class	RANGE	Default Subnet Mask
A	1.0.0.0 to 126.255.255.255	255.0.0.0
B	128.0.0.0 to 191.255.255.255	255.255.0.0
C	192.0.0.0 to 223.255.255.255	255.255.255.0
D	224.0.0.0 to 239.255.255.255	Reserved for Multicasting
E	240.0.0.0 to 254.255.255.255	Experimental
Note: Class A addresses 127.0.0.0 to 127.255.255.255 cannot be used and is reserved for loopback testing.		

Here, the first octet 150 falls between the range 128 and 191, this makes the above IP a class B IP address. The default Subnet mask for class B as shown in the table is 255.255.0.0

The binary notation for the Given Ip address
1001 0110. 0000 1010. 0001 0100. 0001 1110
Similarly, the binary notation for Subnet mask 255.255.0.0 is
1111 1111. 1111 1111. 0000 0000. 0000 0000

As we can see from the above bits representation, the first 16 bits of the subnet mask which are all 1's and corresponding bits in the IPV4 address belong to the network part. Likewise, the last 16 bits of the subnet mask which are all 0's and corresponding bits in the IPV4 address belong to the host part.

The above IP address can be denoted as 150.10.20.30/16 where /16 represents the number of network bits.
The Host 16 bits represents the number of devices within that network plus broadcast id and network id.
So, a total of 2^16 - 2 hosts are allowed within that particular network.

If we replace all the host bits within that IP address with 1's, we will get Broadcast id
The binary notation for Broadcast id
1001 0110. 0000 1010. 1111 1111. 1111 1111
Decimal Equivalent 150.10.255.255

Similarly, if we replace all the host bits within that IP address with 0's, we will get the network id
The binary notation for Network id
1001 0110. 0000 1010. 0000 0000. 0000 0000
Decimal Equivalent : 150.10.0.0

Conclusion
Network Id: 150.10.0.0/16
Broadcast Id: 150.10.255.255