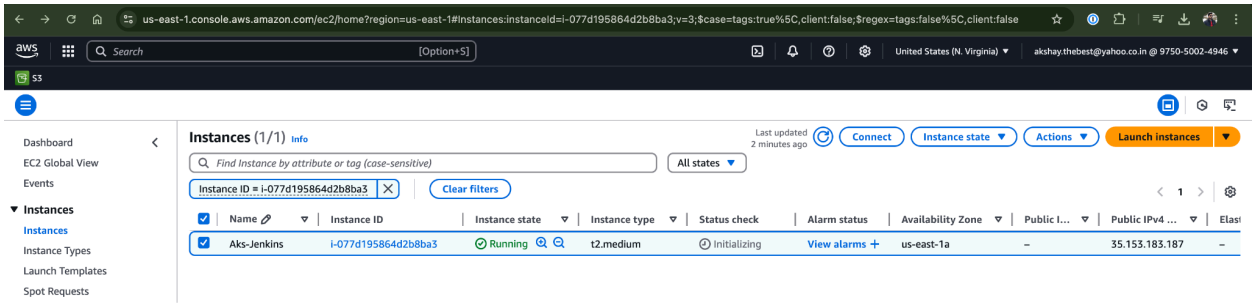
Jenkins CI CD pipeline for flask application

1. **Creating AWS instance**



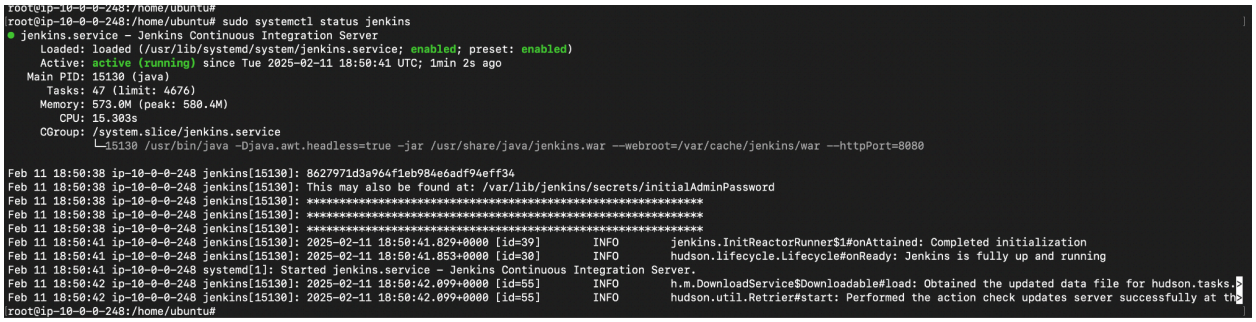
2. **Update and install all dependencies for Jenkins/java.**

```
# Update system packages
sudo apt update
sudo apt upgrade -y

# Install Java (Jenkins requirement)
sudo apt install default-jdk

# Add Jenkins repository key
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \
  /usr/share/keyrings/jenkins-keyring.asc > /dev/null

# Add Jenkins repository
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null

# Update package list and install Jenkins
sudo apt update
sudo apt install jenkins -y

# Start Jenkins service
sudo systemctl start jenkins
sudo systemctl enable jenkins

# Check Jenkins status
```

sudo systemctl status jenkins

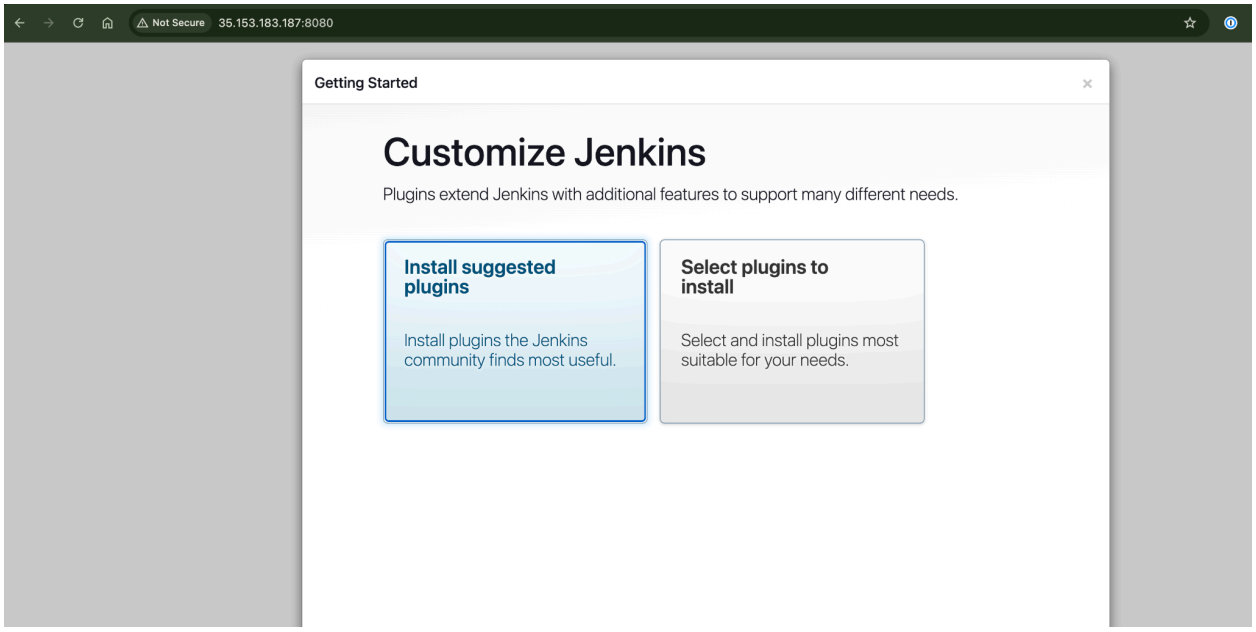
3. **Setup Jenkins server**

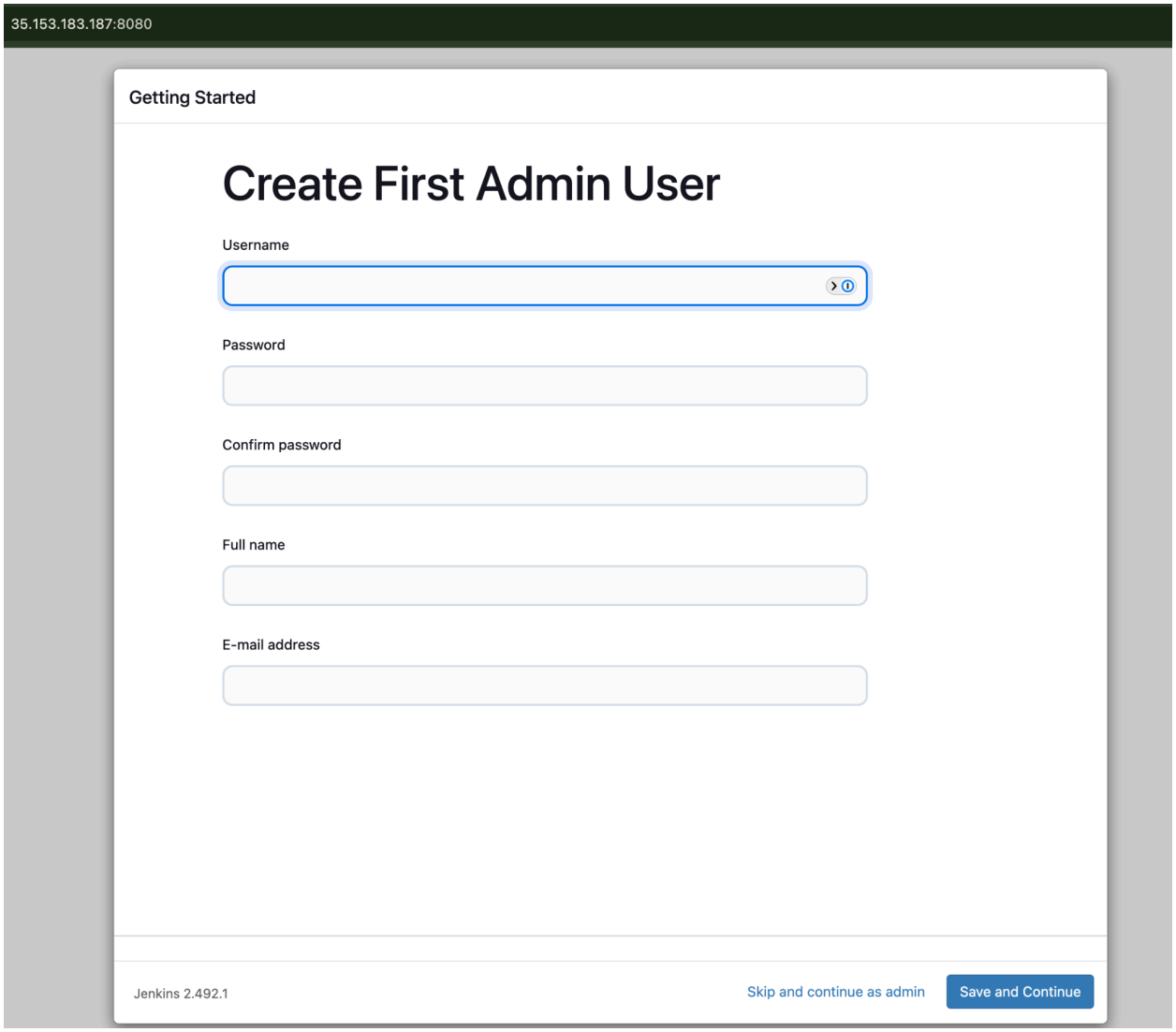
You'll see the Jenkins initial setup page. It will tell you the location of the initialAdminPassword file. SSH into your instance and *`cat /var/lib/jenkins/secrets/initialAdminPassword`* to get the password.

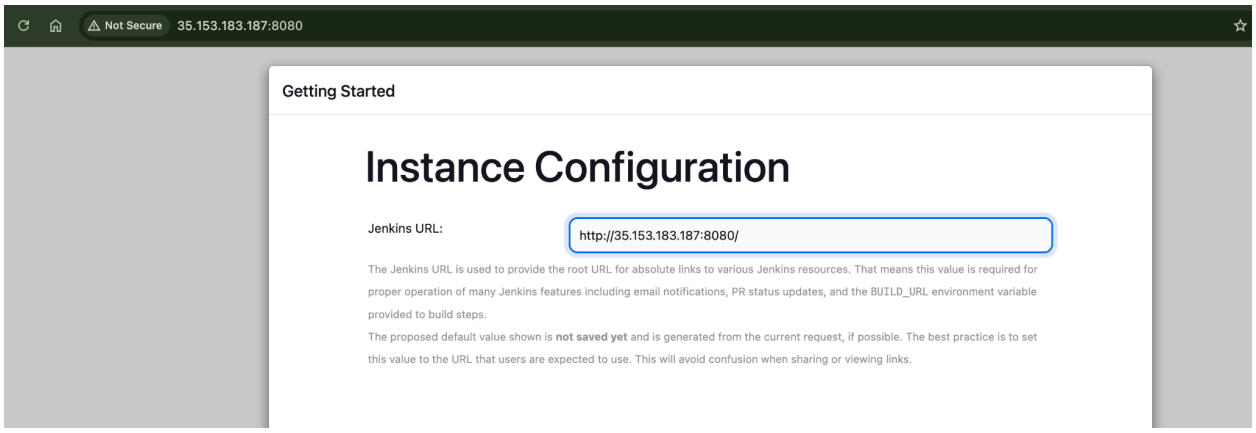


Install Python: `sudo apt install -y python3 python3-pip`

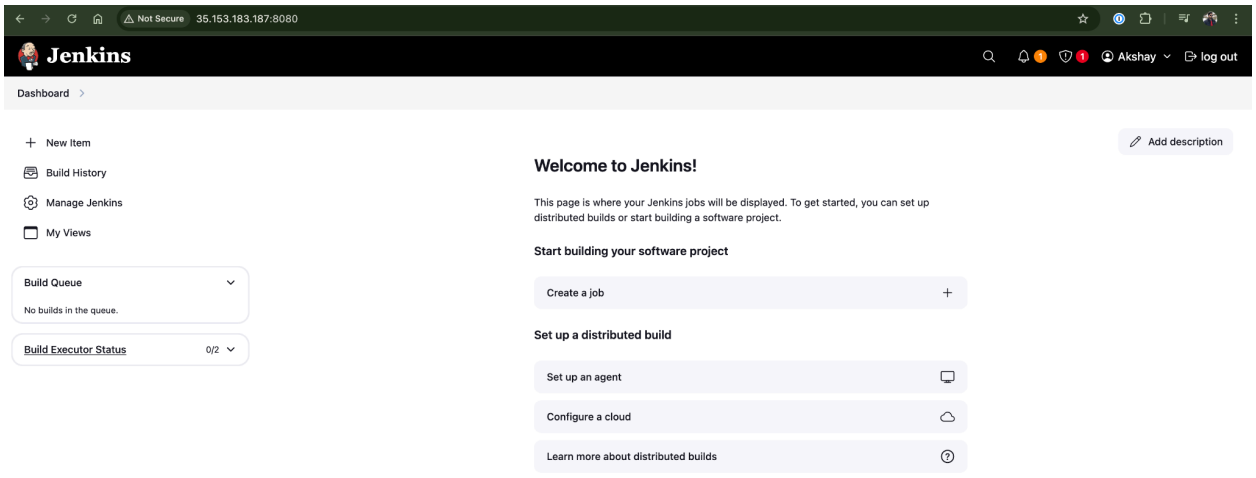
Open chrome with URL— http://<EC2 public IP>:8080




35.153.183.187:8080
Getting Started
Create First Admin User
Username
Password
Confirm password
Full name
E-mail address
Jenkins 2.492.1
Skip and continue as admin
Save and Continue


Not Secure
35.153.183.187:8080
Getting Started
Instance Configuration
Jenkins URL:
http://35.153.183.187:8080/
The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the BUILD_URL environment variable provided to build steps.
The proposed default value shown is not saved yet and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Installation finished



4. **Clone git and create Jenkinsfile.**

Fork [UnpredictablePrashant/FlaskTest]

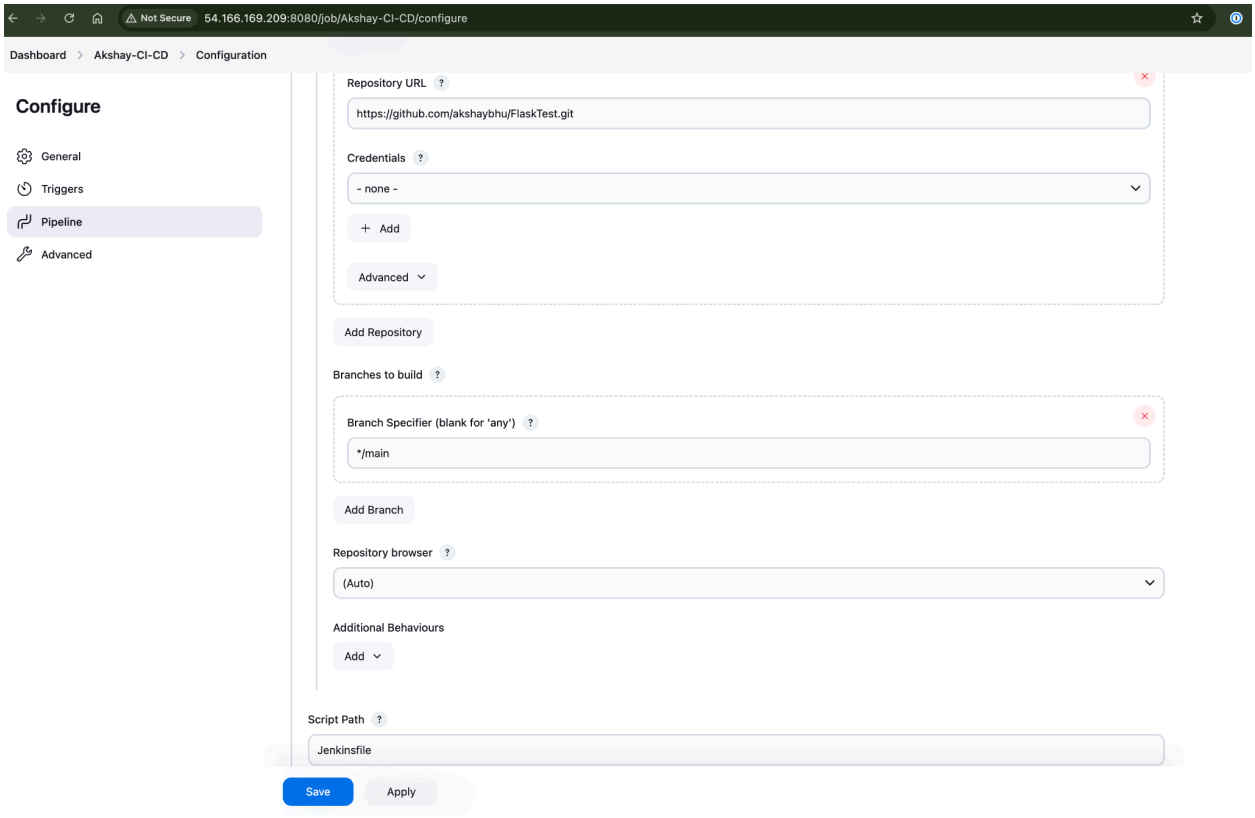
And clone git repository

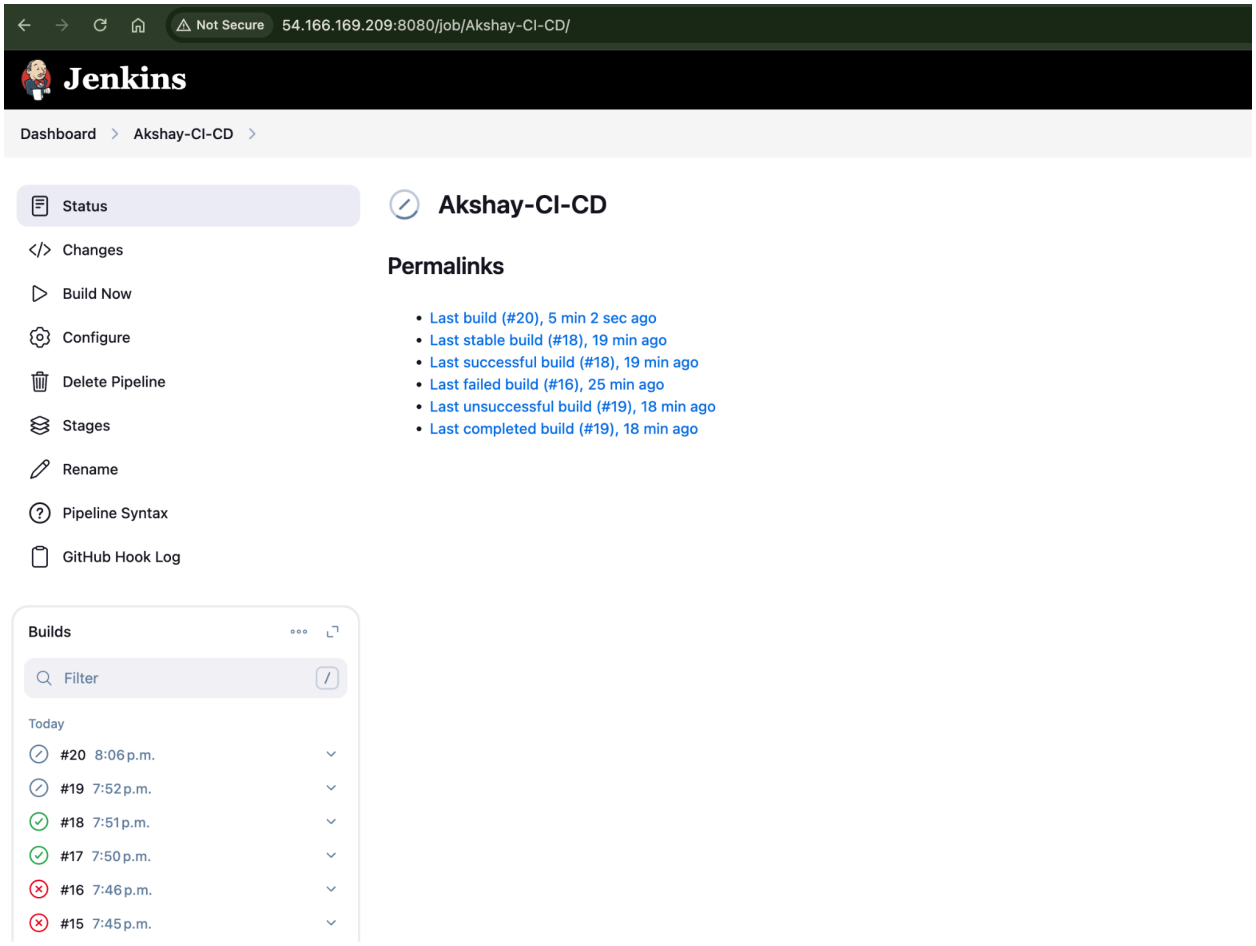
```
root@ip-10-0-0-248:/home/ubuntu# ls
requirements.txt
root@ip-10-0-0-248:/home/ubuntu# git clone https://github.com/akshaybhu/FlaskTest.git
Cloning into 'FlaskTest'...
remote: Enumerating objects: 12, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 12 (delta 2), reused 1 (delta 1), pack-reused 6 (from 1)
Receiving objects: 100% (12/12), 4.45 KiB | 1.48 MiB/s, done.
Resolving deltas: 100% (2/2), done.
root@ip-10-0-0-248:/home/ubuntu# ls
FlaskTest  requirements.txt
root@ip-10-0-0-248:/home/ubuntu# mv requirements.txt FlaskTest/
root@ip-10-0-0-248:/home/ubuntu# ls
FlaskTest
root@ip-10-0-0-248:/home/ubuntu# cd FlaskTest/
root@ip-10-0-0-248:/home/ubuntu/FlaskTest# ls
Jenkinsfile  README.md  app.py  requirements.txt  test_app.py
root@ip-10-0-0-248:/home/ubuntu/FlaskTest#
root@ip-10-0-0-248:/home/ubuntu/FlaskTest#
root@ip-10-0-0-248:/home/ubuntu/FlaskTest#
```

Jenkinsfile is created and uploaded to the same cloned repository.

> Moved default port of 5000 to 5010 due to mac use of this port.

In the Jenkins web UI, create a new item (a "Pipeline" project is recommended). Configure the pipeline. You can use a Pipeline script (Jenkinsfile) for this.





5. Create Jenkins pipeline & Github Webhook

Created a JenkinsFile inside FlaskTest repository using

Create Jenkins Pipeline Job
Open Jenkins Dashboard → Click New Item.
Select Pipeline, name it **Akshay-CI-CD**, and click OK.

Under Pipeline Definition, choose "Pipeline script from SCM."
Select Git, enter your repository URL.

Under Branches to build, enter main.
Save and run the pipeline.
Configure Github Webhook

Go to the GitHub repository → Click Settings → Click Webhooks.
Click "Add Webhook" and set:
Payload URL:
http://54.166.169.209:8080/github-webhook/
Content type: application/json
Trigger: Select Just the push event.
Click Add Webhook.

Update the Jenkins Pipeline Job

Open Jenkins Dashboard → Click on **Akshay-CI-CD** Job.
Click Configure.
Under Build Triggers, check - GitHub hook trigger for GITScm polling.
Save the configuration.

Verification

```
ubuntu@ip-10-0-0-248:~$
ubuntu@ip-10-0-0-248:~$ curl http://54.145.101.157:5010
Hello, World!ubuntu@ip-10-0-0-248:~$
ubuntu@ip-10-0-0-248:~$
ubuntu@ip-10-0-0-248:~$ curl http://localhost:5010
Hello, World!ubuntu@ip-10-0-0-248:~$
```

Hello, World!