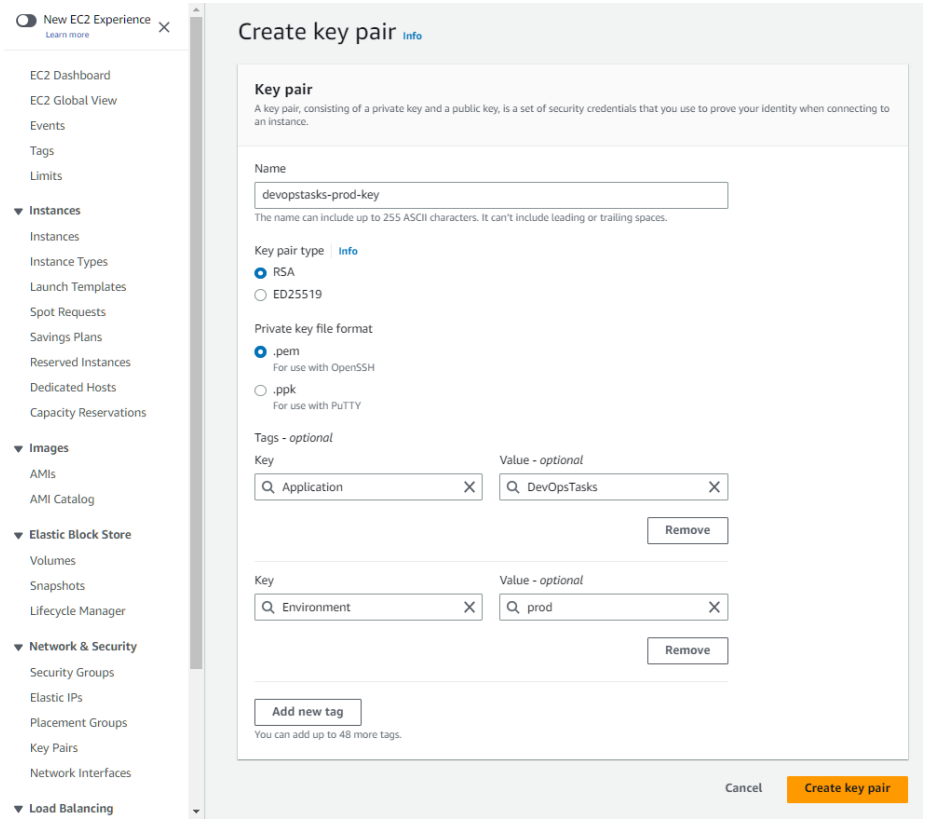
Deployment instruction

1. SSH key pair creation

You need to create a key pair for Ansible SSH connections to Web servers. Open AWS EC2 service, press "Key Pairs" on the left panel and click "Create key pair". Fill the creation form as you see on the screenshot.



Then press Create key pair. Now the private key file is automatically downloaded by your browser. Save the private key file in a safe place.

2. Create a Fork of this repository in your GitHub account

3. Clone your repository locally on the workstation

4. Install and configure AWS CLI on the workstation (see Installing or updating the latest version of the AWS CLI)

5. Install terraform on the workstation (see Install Terraform)

6. Create two parameters in AWS Systems Manager Parameter Store:

- Username for creating RDS Database.
  Type: String

Name: /config/DevOpsTasks_prod/db.username
Value: any username, for example "postgres"

- UserPassword for creating RDS Database
  Type: SecureString
  Name: /config/DevOpsTasks_prod/db.username
  Value: any password, for example "mysecretpas"

7. Setting AWS Region

By default, deployment is using the "eu-central-1" AWS region. If you need to use another region you must change:

- aws_region variable in terraform.tfvars located at DevOps/AWS/EC2/Terraform/Infrastructure/prod
- regions field in dynamic ansible inventory settings file ansible_inventory.aws.aws_ec2.ymllocated at DevOps/AWS/EC2/Ansible

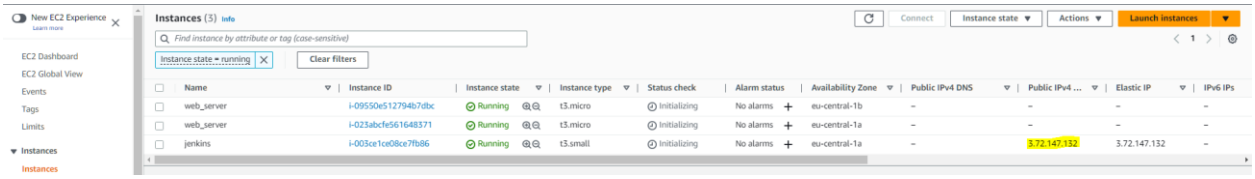
and push these changes to your GitHub repository

8. Creating infrastructure by Terraform

Change directory to DevOps/AWS/EC2/Terraform/Infrastructure/prod and run commands:

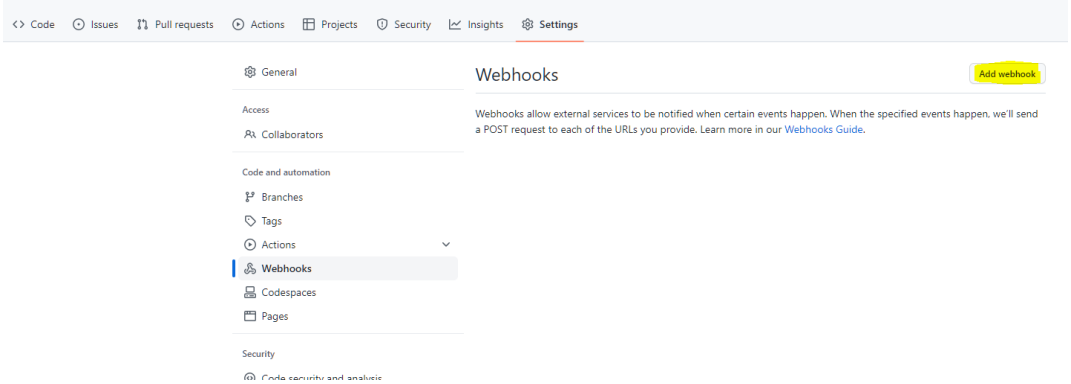
```
terraform init
terraform apply (type yes and press enter when it asks "Do you want to perform these actions?")
```

Terraform will create the entire infrastructure but without Jenkins settings. You should see two web and one Jenkins servers in AWS EC2 instances.



9. Setting up GitHub webhook.

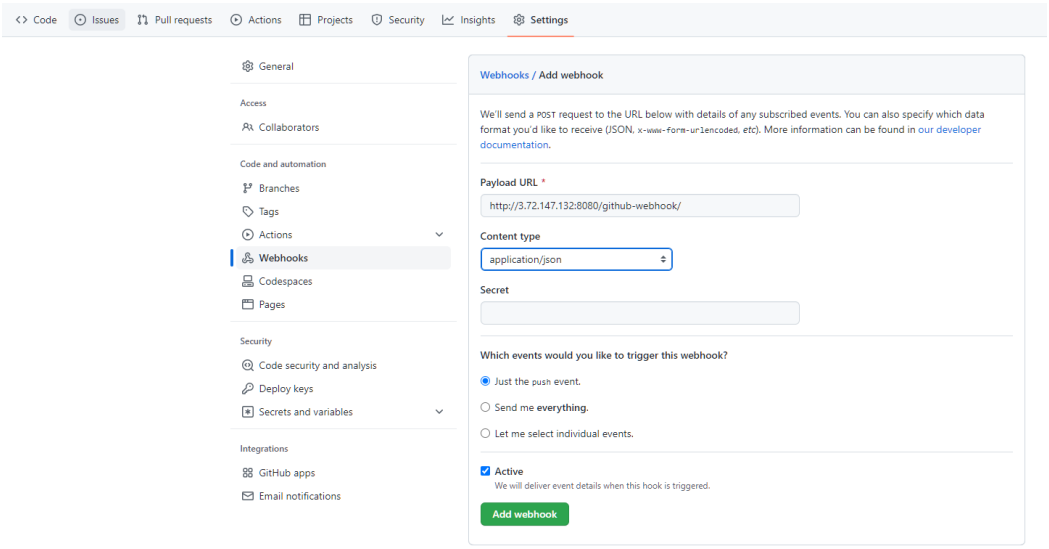
Webhooks allow you to build or set up integrations, such as GitHub Apps or OAuth Apps, which subscribe to certain events on GitHub.com. When one of those events is triggered, GitHub will send a HTTP POST payload to the webhook's configured URL. Webhooks can be used trigger CI builds. To setting up webhook open GitHub repository settings in your GitHub account. Select Webhooks and click Add webhook.



Set the following fields:

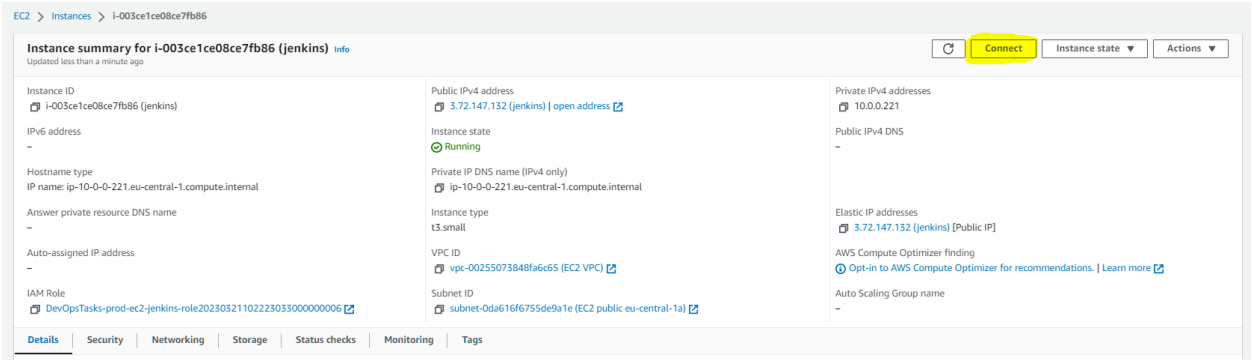
- Payload URL: "http://<jenkins server public IP>:8080/github-webhook/"
- Content type: "application/json"

press Add webhook.



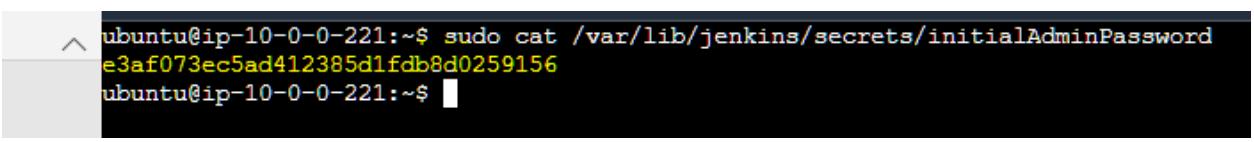
10. Unlocking Jenkins

Connect over SSH to Jenkins server from AWS Console.



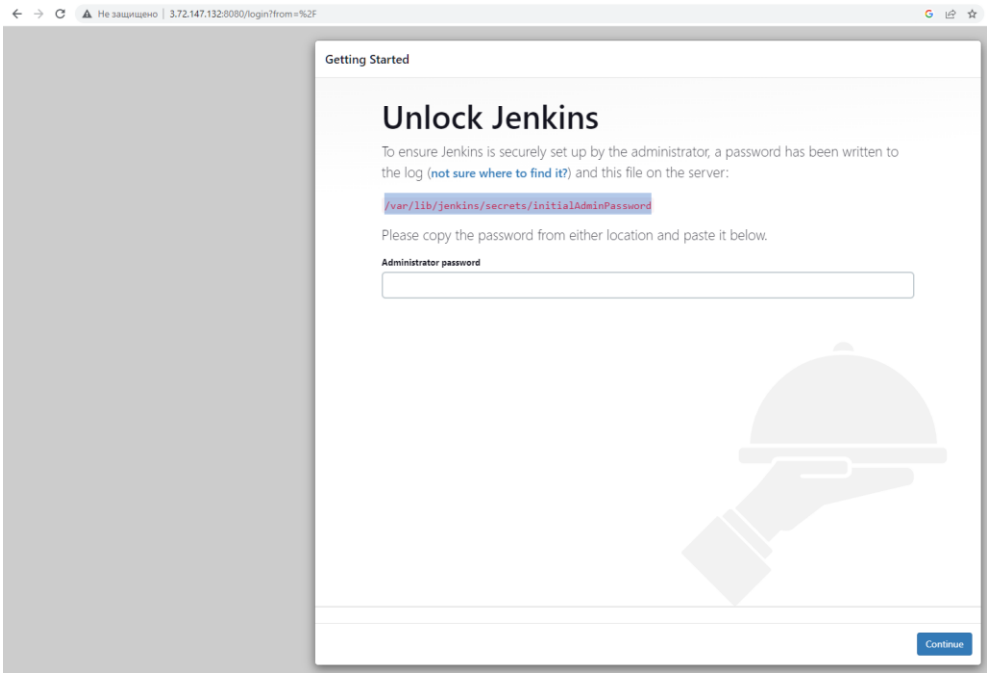
Run command:

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

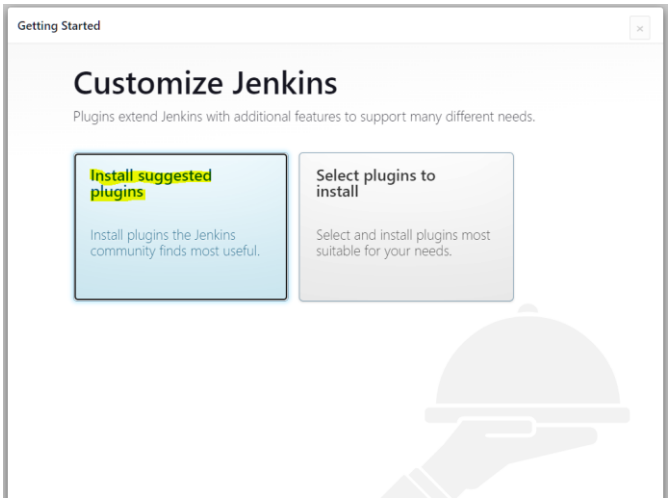


Copy received Administrator password.

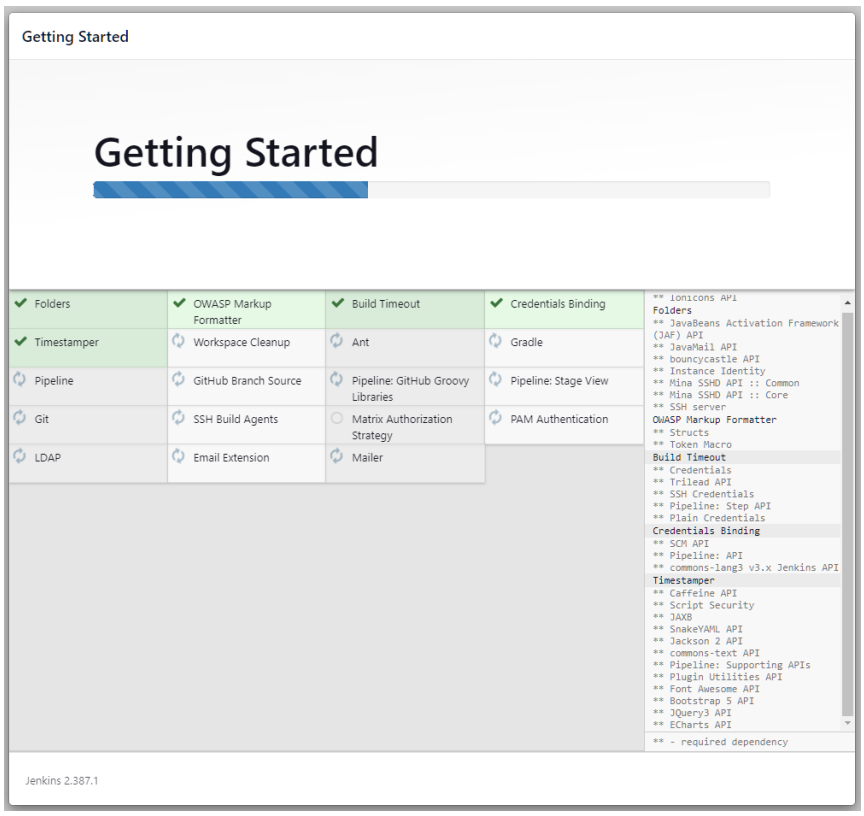
Open link <Jenkins server public ip>:8080 in your browser:



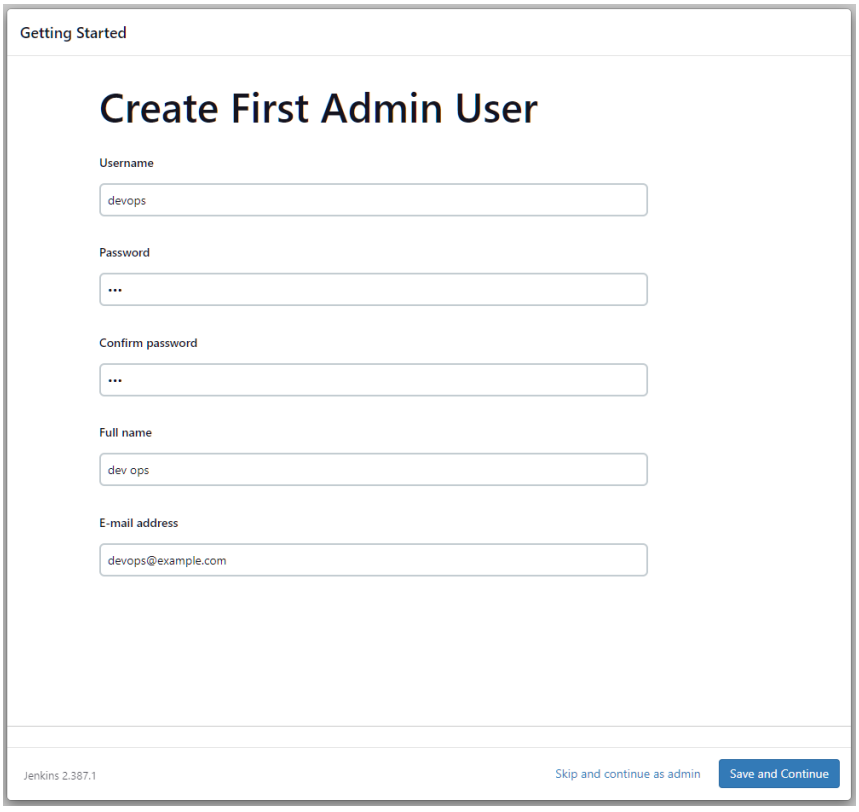
Paste the administrator password in a field and press continue

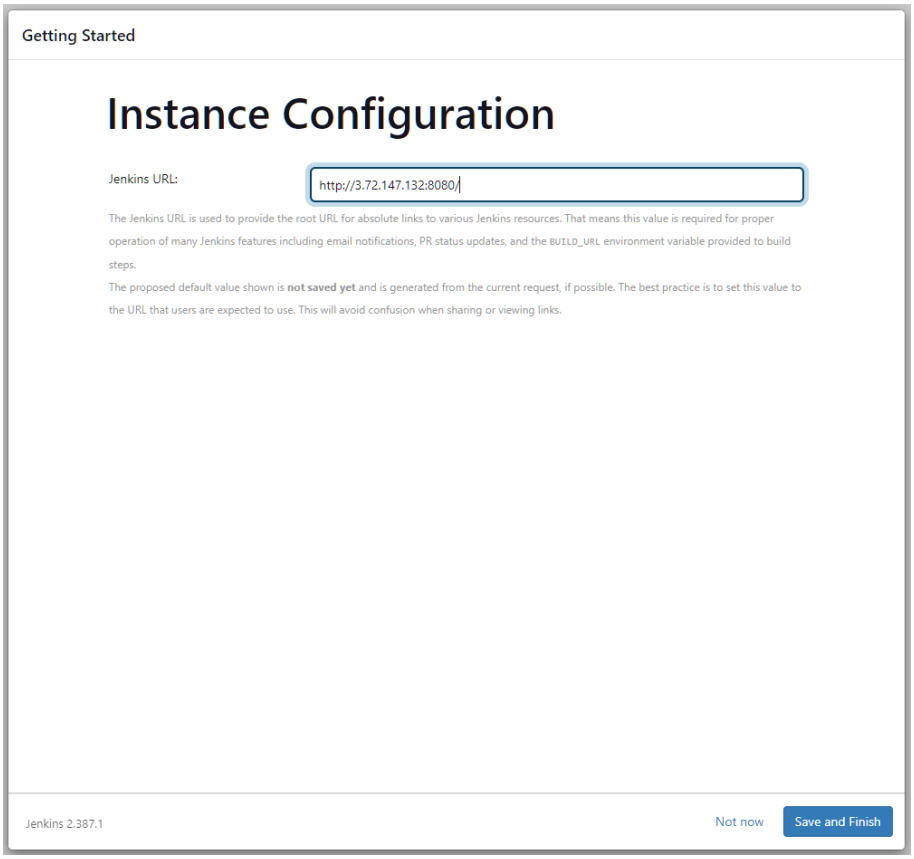


Click Install suggested plugins and wait until installing process is finished.



Fill in all fields to create a user and press «Save and Continue»



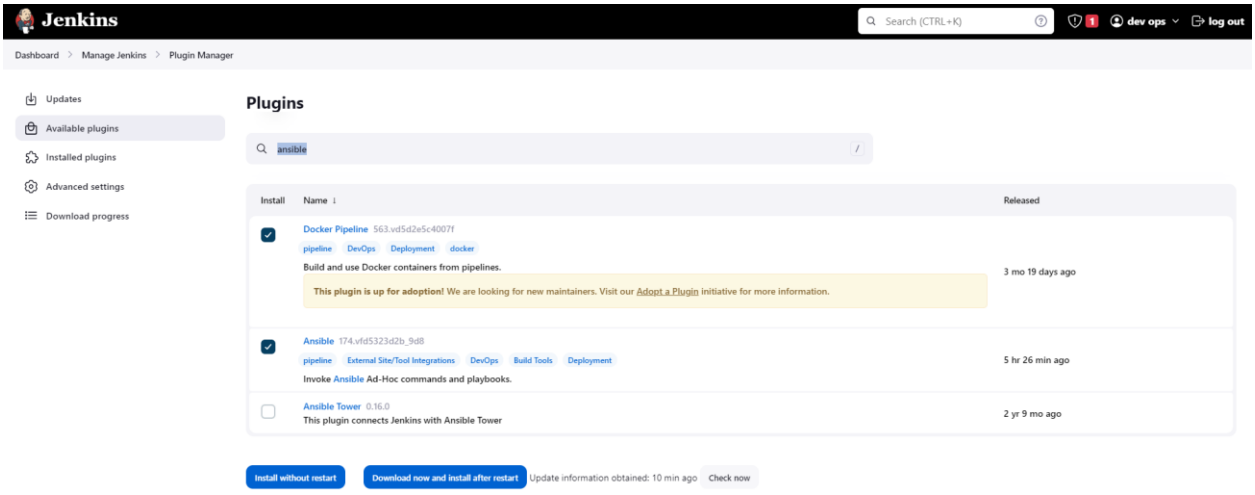


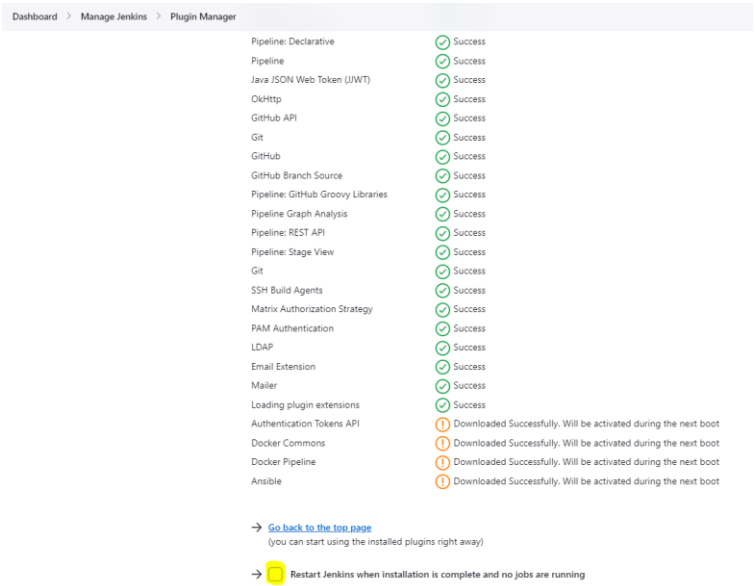
Click «Save and Finish»

11. Installing Jenkins plugins

Open Manage Jenkins -> Manage Plugins -> Available plugins

Select plugins: Docker Pipeline, Ansible and install them.





Click "Restart Jenkins when installation is complete and no jobs are running"

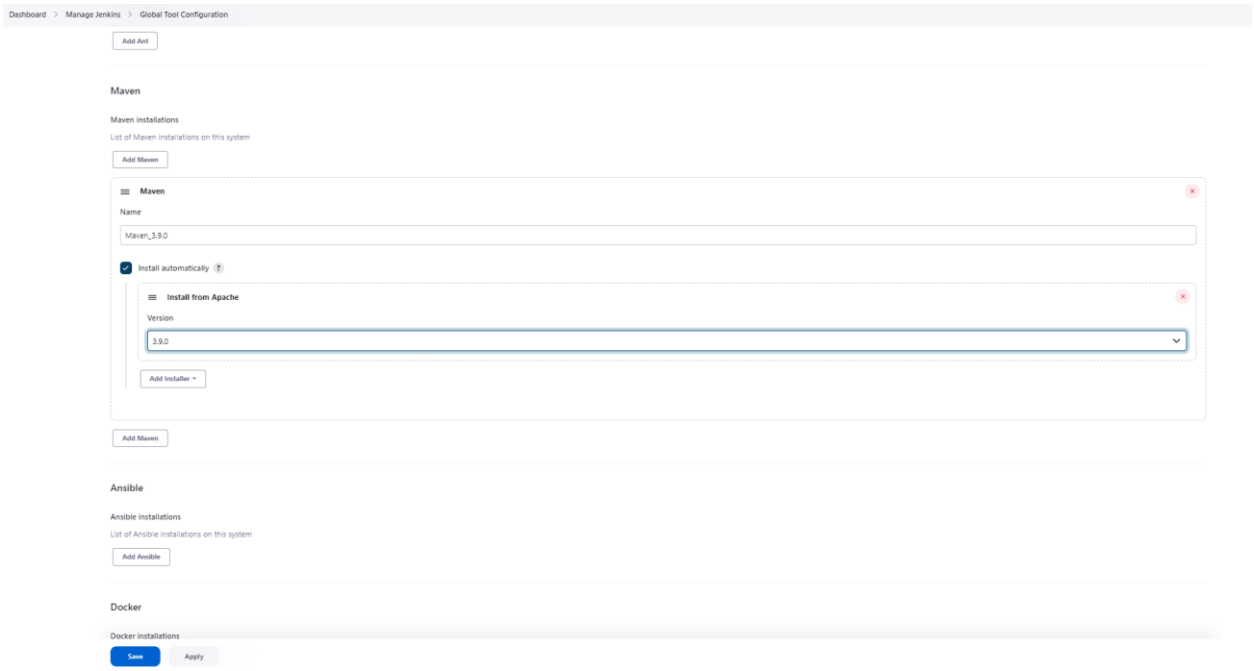
12. Setting up Maven

Open Manage Jenkins -> Global Tool Configuration

In the Maven section, click "Add Maven", fill the following fields:

- Name: Maven_3.9.0
- Version: 3.9.0

click Save.



13. Setting up SSH key for authentication on web servers

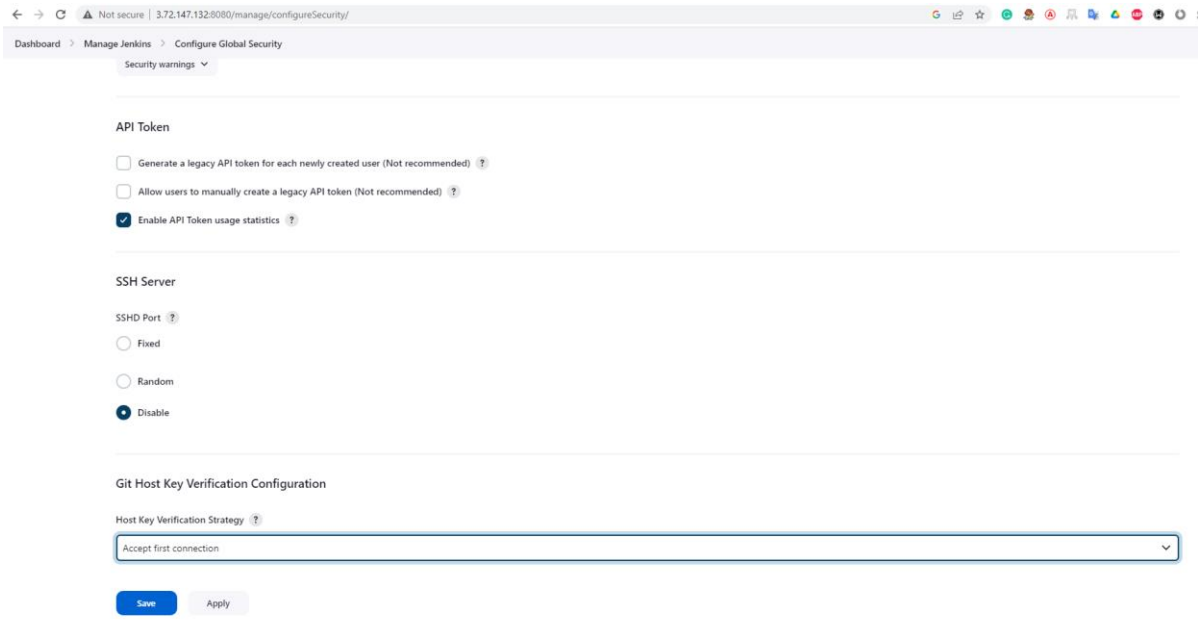
Open Manage Jenkins -> Manage Credentials. Put mouse cursor on Global domain and press Add credentials from drop-down list. In new window fill the following fields:

- Kind: "SSH Username with private key"
- ID: "devopstasks-prod-key"
- Username: "ubuntu"

Select «Enter directly» in private Key section and press Add. Paste private key downloaded on step 1 and click Create.

14. Setting up SSH keys for authentication on GitHub

Open Manage Jenkins -> Configure Global Security. In «Git Host Key Verification» section set «Host Key Verification Strategy» to «Accept first connection» and click Save.



You must set up an SSH key pair to authenticate Jenkins on GitHub. You need to generate this pair. You can use Jenkins SSH connection for key generation. Run command:

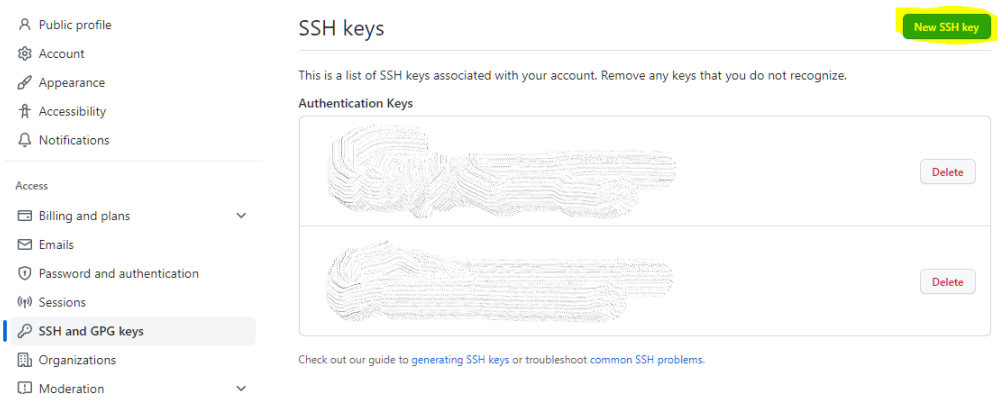
```
ssh-keygen -t rsa
```

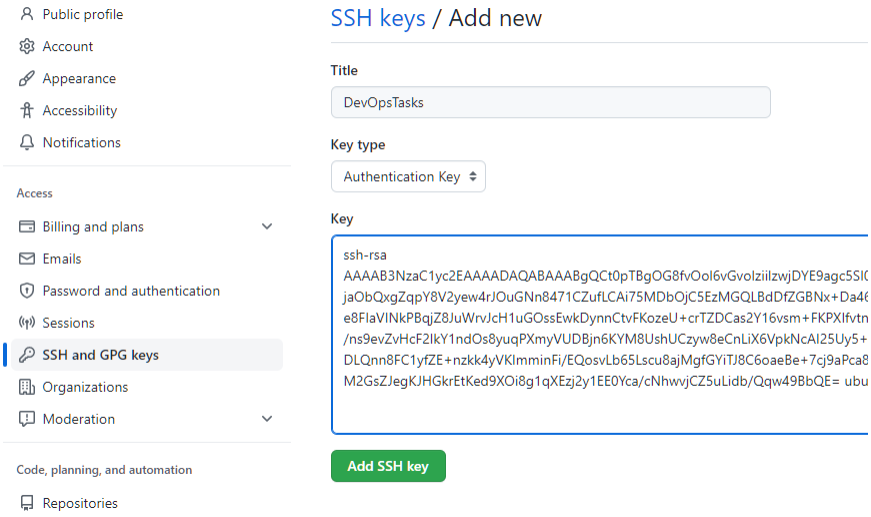
```
ubuntu@ip-10-0-0-221:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_rsa): git-hub
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in git-hub
Your public key has been saved in git-hub.pub
The key fingerprint is:
SHA256:J3UNx3YSEtNTrCrG02Wru3obkasgWhKBFk6yGoRrPrQ ubuntu@ip-10-0-0-221
The key's randomart image is:
+---[RSA 3072]----+
|o.o         =+o+.|
|o= o         ** o|
|o.+ .     . ...= |
|o=   .   . . .+  |
|= . .   S...o+ . |
| E   .   o= oo.  |
|  . . o .. oo.   |
|     + . . .o.   |
|    .     oo++   |
+----[SHA256]-----+
ubuntu@ip-10-0-0-221:~$ cat git-hub.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQCt0pTBgOG8fvOol6vGvolziilzwjDYE9agc5Sl0xOO5Sg3s6jEAQNt/rIpJzc47o52GLjaObQxgZqpY8V2yew4rJOu
KozeU+crTZDCas2Y16vsm+FKPXIfvtn9VVmYkW9eCuqLaDKbWgBPTTT/4/ns9evZvHcF2lkYlndOs8yuqPXmyVUDBjn6KYM8UshUCzyw8eCnLiX6VpkNcAI25Uy5+WA7
NTZV0XQgDSM2Gs2JegKJHGkrEtKed9XOi8g1qXEzj2y1EE0Yca/cNhwvjCZ5uLidb/Qqw49BbQE= ubuntu@ip-10-0-0-221
ubuntu@ip-10-0-0-221:~$
```

After executing the command, public (git-hub.pub) and private (git-hub) key files will be generated. To display the public key, run the command:

```
cat git-hub.pub
```

Copy public key. Open GitHub user settings and create new SSH key, paste public key in the "Key" field:





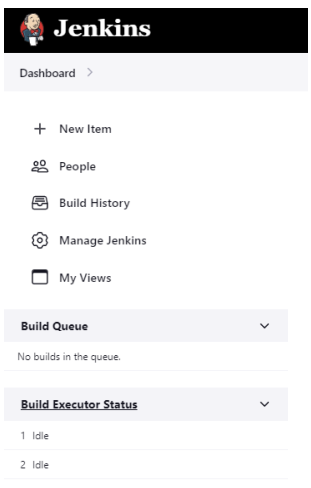
To display the private key, run the command:

```
cat git-hub
```

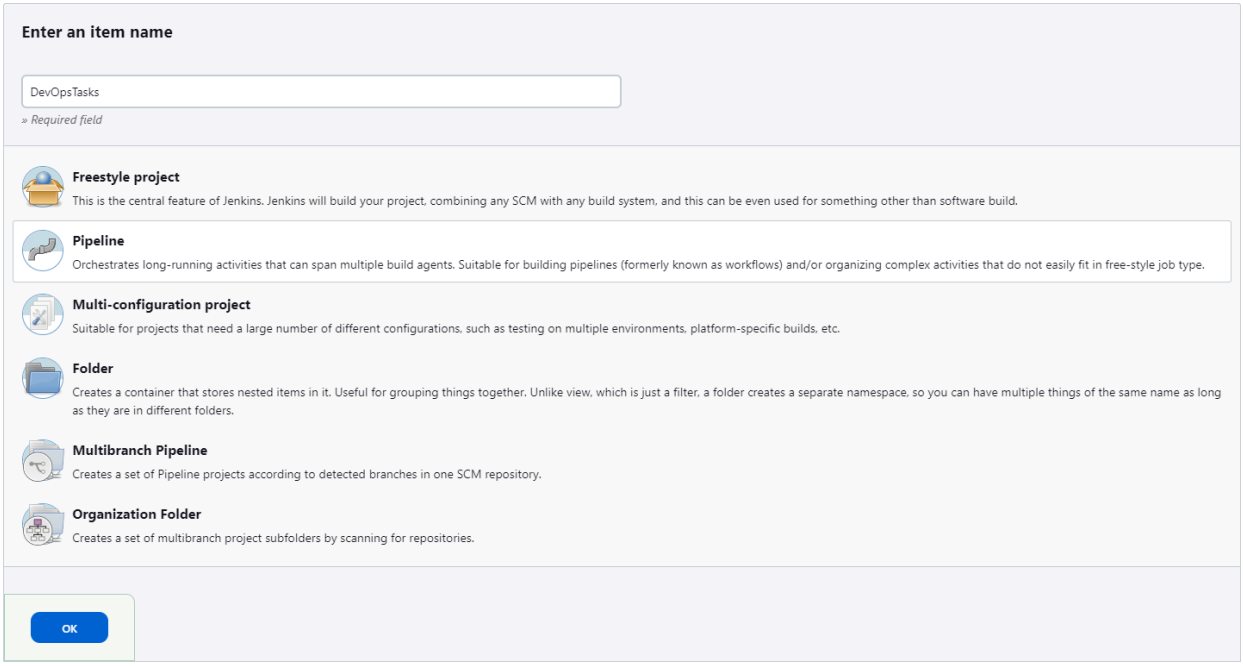
You will need the private key on the next step 15.

15. Setting up a job in Jenkins

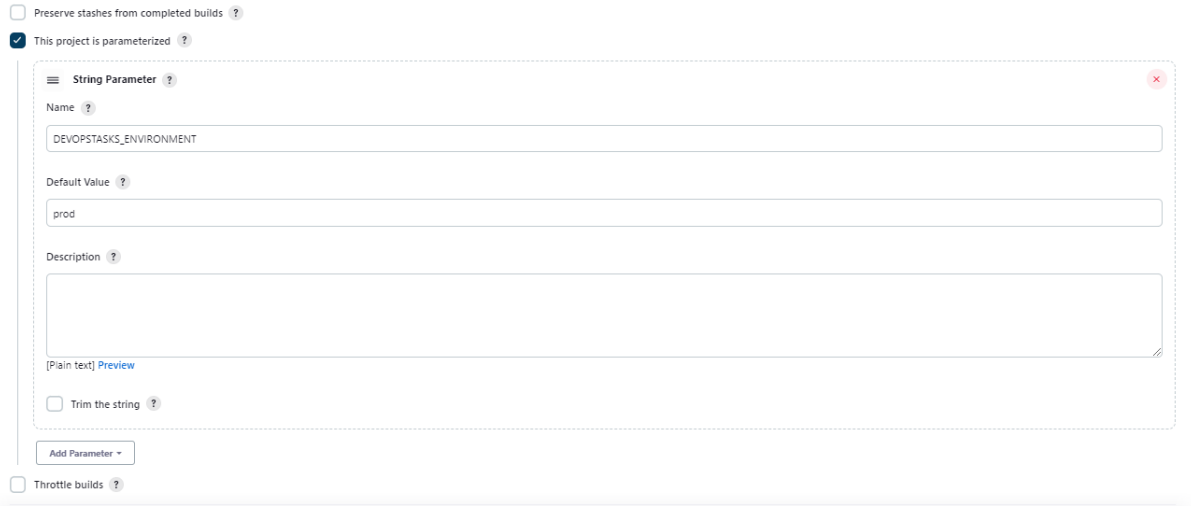
Open Jenkins dashboard and click "New Item".



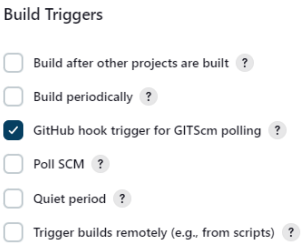
Fill in the name of the job, select the pipeline and click OK.



Select "This project is parameterized" in General section, click "Add parameter", select "String Parameter" and fill fields as on screenshot.



Select "GitHub hook trigger for GITScm polling" checkbox in Build Triggers section.



In Pipeline section set:

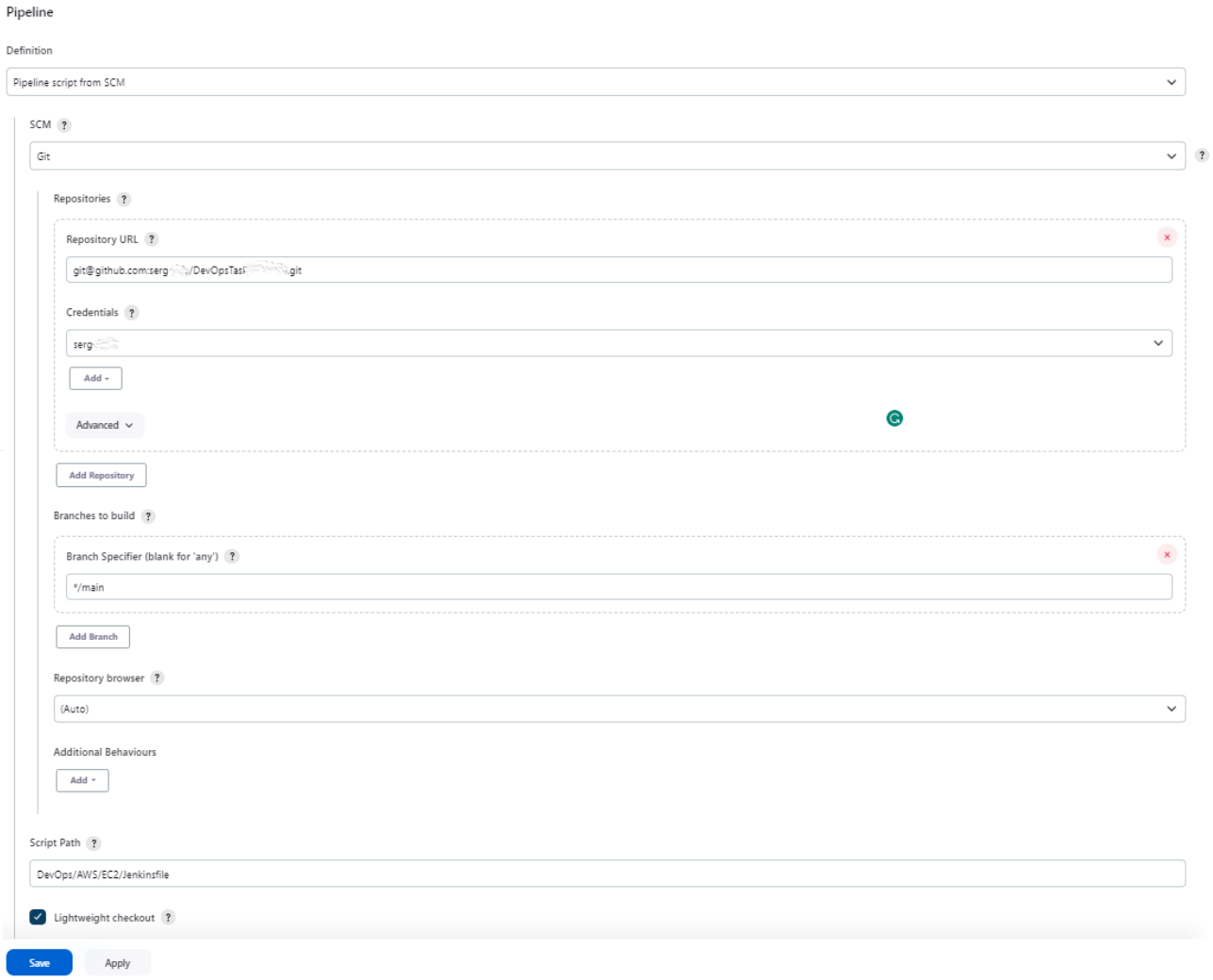
- Definition: Pipeline script from SCM
- SCM: Git

- Repository URL: an SSH URL to the forked repository, like: "git@github.com:accountname/repository.git"

Click Add bellow the credentials field and select Jenkins. In new window fill the following fields:

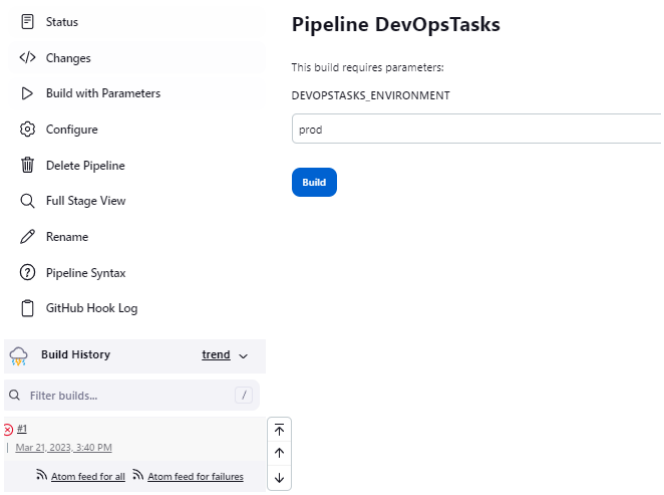
- Kind: SSH Username with private key
- ID: devopstasks-prod
- Username: your GitHub user name
- Branch Specifier: */main
- Script Path: DevOps/AWS/EC2/Jenkinsfile

Select «Enter directly» in the private Key section and press Add. Paste the private key generated on step 14 and click Add. Select added key in the Credentials field. Click Save.

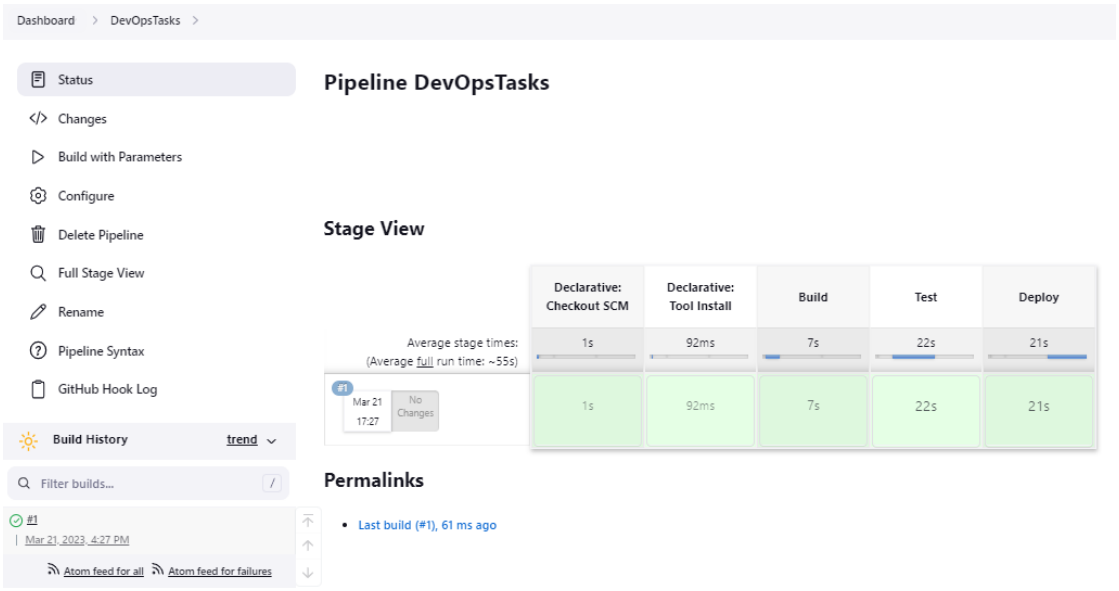


16. Running pipeline in manual mode

Click «Build with Parameters» and press Build



After the build is completed, you should see all stages green:

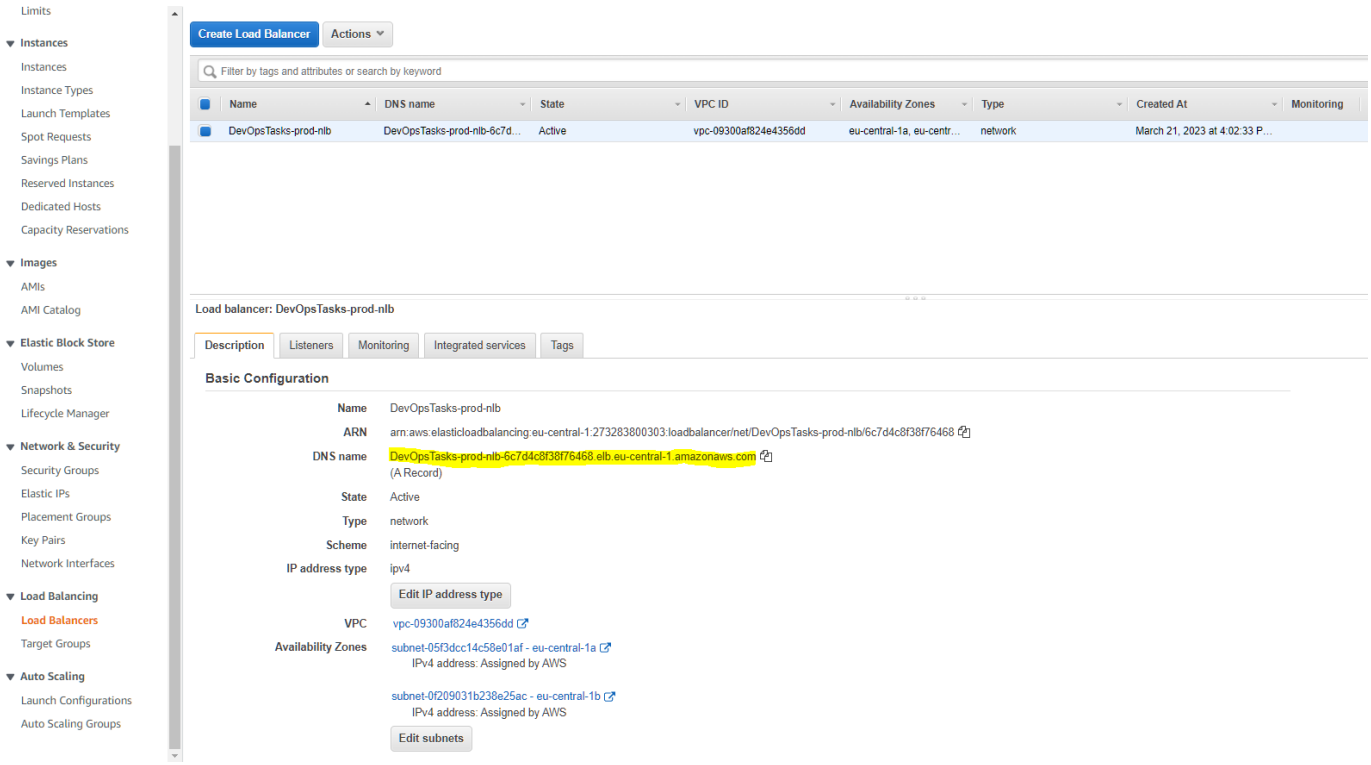


17. Running pipeline in automatic mode

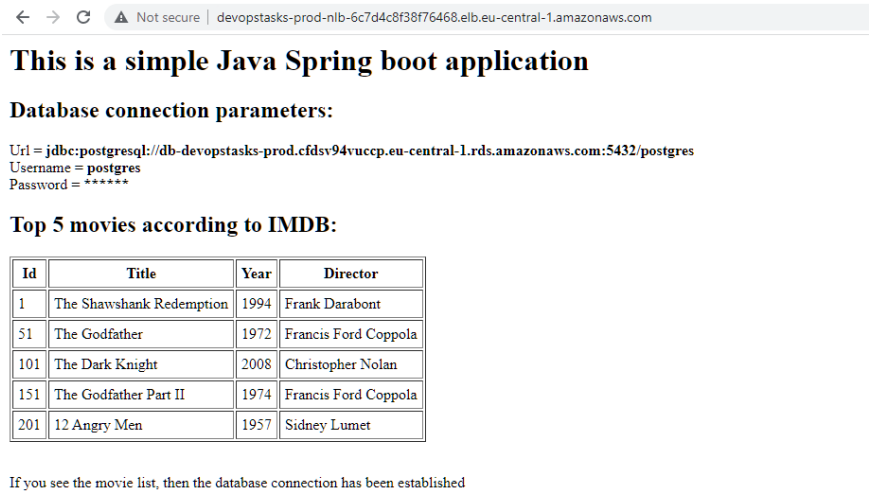
We have made all settings for automatically running the pipeline on push event in the GitHub repository. You can check it by changing and committing changes in any repository file.

18. Checking the application

Now you can find the load balancer DNS name in the AWS EC2 service:



You can open this link in a web browser and should see the application:



19. Clean up

For cleanup you should:

1. Remove created SSH key from GitHub user;
2. Remove created AWS Key pair;
3. Run "terraform destroy" command from DevOps/AWS/EC2/Terraform/Infrastructure/prod on you workstation.