

Name: **Đỗ Minh Trí**

Staff Code: **SD5976**

## AWS Infrastructure

### I. Setting Up AWS Infrastructure with Terraform

We will deploy AWS infrastructure using Terraform through this repository:

[https://github.com/tri-dominh/sd5976\\_aws\\_infrastructure](https://github.com/tri-dominh/sd5976_aws_infrastructure)

First, I created an IAM user for practical DevOps training: **sd5976**.



The screenshot shows the AWS IAM User details page for 'sd5976'. It includes sections for Summary, ARN, Console access, Last console sign-in, and Access keys.

Summary	ARN	Console access	Access key 1
Created October 29, 2025, 14:26 (UTC+07:00)	arn:aws:iam::177629262279:user/sd5976	Enabled without MFA	AKIASSW4HOHDYUXX4GOR - Active Used today, 8 days old.
		Last console sign-in Today	Access key 2 Create access key

Configure this IAM user on local via **AWS CLI** with Access Key and Secret Access Key.

```
• ~/Desktop/sd5976_aws_infrastructure main ➔ aws configure
AWS Access Key ID [*****XWP5]: ****4G0R
AWS Secret Access Key [*****1m6F]: ****6ic5
Default region name [ap-southeast-1]:
Default output format [json]

• ~/Desktop/sd5976_aws_infrastructure main ➔ aws sts get-caller-identity
{
    "UserId": "****63C",
    "Account": "177629262279",
    "Arn": "arn:aws:iam::177629262279:user/sd5976"
}
```

Go to **sd5976\_aws\_infrastructure/terraform**, run **terraform init** to initialize the infrastructure.

```
● ~/Desktop/sd5976_aws_infrastructure/terraform ➔ main ➔ terraform init
Initializing the backend...
Initializing modules...
- ec2 in modules/ec2
- ecr_backend in modules/ecr
- ecr_frontend in modules/ecr
- eks in modules/eks
- security_groups in modules/security-groups
- vpc in modules/vpc
Initializing provider plugins...
- Finding hashicorp/tls versions matching ">~ 4.0"...
- Finding hashicorp/aws versions matching ">~ 5.70.0"...
- Finding latest version of hashicorp/local...
- Installing hashicorp/tls v4.1.0...
- Installed hashicorp/tls v4.1.0 (signed by HashiCorp)
- Installing hashicorp/aws v5.70.0...
- Installed hashicorp/aws v5.70.0 (signed by HashiCorp)
- Installing hashicorp/local v2.5.3...
- Installed hashicorp/local v2.5.3 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider selections it made above. Include this file in your version control repository so that Terraform can guarantee to make the same selections by default when you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see any changes that are required for your infrastructure. All Terraform commands should now work.

If you ever set or change modules or backend configuration for Terraform, rerun this command to reinitialize your working directory. If you forget, other commands will detect it and remind you to do so if necessary.
```

Use **terraform fmt** and **terraform validate** to format and check your code.

```
● ~/Desktop/sd5976_aws_infrastructure/terraform ➔ main ➔ terraform fmt
terraform.tfvars

● ~/Desktop/sd5976_aws_infrastructure/terraform ➔ main !1 ➔ terraform validate
Success! The configuration is valid.
```

Run **terraform plan** to preview resources that will be created.

```
Plan: 38 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ configure_kubectl          = "aws eks update-kubeconfig --region ap-southeast-1 --name cicd-pipeline-cluster"
+ docker_ip                   = (known after apply)
+ ecr_backend_repository_name = "sd5976-backend"
+ ecr_backend_repository_url  = (known after apply)
+ ecr_frontend_repository_name= "sd5976-frontend"
+ ecr_frontend_repository_url = (known after apply)
+ eks_cluster_endpoint        = (known after apply)
+ eks_cluster_name            = "cicd-pipeline-cluster"
+ jenkins_initial_password_command = (known after apply)
+ jenkins_ip                  = (known after apply)
+ jenkins_url                 = (known after apply)
+ ssh_connection_docker       = (known after apply)
+ ssh_connection_jenkins      = (known after apply)
+ vpc_id                      = (known after apply)

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.

○ ~/Desktop/sd5976_aws_infrastructure/terraform ➔ main ➔
```

Execute **terraform apply** to deploy the resources.

Type **yes** to confirm.

```
Plan: 38 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ configure_kubectl      = "aws eks update-kubeconfig --region ap-southeast-1 --name cicd-pipeline-cluster"
+ docker_ip                = (known after apply)
+ ecr_backend_repository_name = "sd5976-backend"
+ ecr_backend_repository_url = "177629262279.dkr.ecr.ap-southeast-1.amazonaws.com/sd5976-backend"
+ ecr_frontend_repository_name = "sd5976-frontend"
+ ecr_frontend_repository_url = "177629262279.dkr.ecr.ap-southeast-1.amazonaws.com/sd5976-frontend"
+ eks_cluster_endpoint     = (known after apply)
+ eks_cluster_name          = "cicd-pipeline-cluster"
+ jenkins_initial_password_command = (known after apply)
+ jenkins_ip                  = (known after apply)
+ jenkins_url                 = (known after apply)
+ ssh_connection_docker       = (known after apply)
+ ssh_connection_jenkins      = (known after apply)
+ vpc_id                      = (known after apply)

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes
```

Resource creation may take around 20 minutes or more.

```
Apply complete! Resources: 38 added, 0 changed, 0 destroyed.

Outputs:

configure_kubectl = "aws eks update-kubeconfig --region ap-southeast-1 --name cicd-pipeline-cluster"
docker_ip = "47.128.237.198"
ecr_backend_repository_name = "sd5976-backend"
ecr_backend_repository_url = "177629262279.dkr.ecr.ap-southeast-1.amazonaws.com/sd5976-backend"
ecr_frontend_repository_name = "sd5976-frontend"
ecr_frontend_repository_url = "177629262279.dkr.ecr.ap-southeast-1.amazonaws.com/sd5976-frontend"
eks_cluster_endpoint = "https://9A770A8B363B32838F392C1018289B7E.gr7.ap-southeast-1.eks.amazonaws.com"
eks_cluster_name = "cicd-pipeline-cluster"
jenkins_initial_password_command = "ssh -i cicd-pipeline-key.pem ubuntu@13.228.28.131 'sudo cat /var/lib/jenkins/secrets/initialAdminPassword'"
jenkins_ip = "13.228.28.131"
jenkins_url = "http://13.228.28.131:8080"
ssh_connection_docker = "ssh -i cicd-pipeline-key.pem ubuntu@47.128.237.198"
ssh_connection_jenkins = "ssh -i cicd-pipeline-key.pem ubuntu@13.228.28.131"
vpc_id = "vpc-0184de0dfb2c42624"

○ ~/Desktop/sd5976_aws_infrastructure/terraform ➜ main ➔ ]
```

After applying, you can see the following resources on AWS:

## VPC

Resources by Region		
You are using the following Amazon VPC resources		
<a href="#">VPCs</a> ▶ See all regions	Singapore 2	<a href="#">NAT Gateways</a> ▶ See all regions
<a href="#">Subnets</a> ▶ See all regions	Singapore 7	<a href="#">VPC Peering Connections</a> ▶ See all regions
<a href="#">Route Tables</a> ▶ See all regions	Singapore 4	<a href="#">Network ACLs</a> ▶ See all regions
<a href="#">Internet Gateways</a> ▶ See all regions	Singapore 2	<a href="#">Security Groups</a> ▶ See all regions
<a href="#">Egress-only Internet Gateways</a> ▶ See all regions	Singapore 0	<a href="#">Customer Gateways</a> ▶ See all regions
<a href="#">DHCP option sets</a> ▶ See all regions	Singapore 1	<a href="#">Virtual Private Gateways</a> ▶ See all regions
<a href="#">Endpoints</a> ▶ See all regions	Singapore 0	<a href="#">Site-to-Site VPN Connections</a> ▶ See all regions
<a href="#">Instance Connect Endpoints</a> ▶ See all regions	Singapore 0	<a href="#">Running Instances</a> ▶ See all regions
<a href="#">Endpoint Services</a> ▶ See all regions	Singapore 0	

The screenshot shows the AWS VPC dashboard with the following details:

- Left sidebar:** Includes sections for VPC dashboard, Virtual private cloud (with Your VPCs, Subnets, Route tables, Internet gateways, Egress-only internet gateways, DHCP option sets, Elastic IPs, Managed prefix lists, NAT gateways, Peering connections, and Route servers), Security (Network ACLs, Security groups), and PrivateLink and Lattice.
- Top bar:** Shows the URL https://ap-southeast-1.console.aws.amazon.com/vpcconsole/home?region=ap-southeast-1#vpcs, the AWS logo, a search bar, and various navigation icons.
- Main content area:**
  - Your VPCs (2) Info:** A table listing two VPCs:

Name	VPC ID	State	Block Public...	IPv4 CIDR	IPv6 CIDR	DHCP option set	Main route table
cldc-pipeline-vpc	vpc-0184de0dfb2c42624	Available	Off	10.0.0.0/16	-	dopt-00f00fdb9c3280f8e	rtb-0d7613a9b048
-	vpc-d55d38068b5fc353c	Available	Off	172.31.0.0/16	-	dopt-00f00fdb9c3280f8e	rtb-0d5f821e3f3t
  - Select a VPC above:** A placeholder text indicating which VPC to select.

## EC2

The screenshot shows the AWS EC2 Instances page. The left sidebar navigation includes: Dashboard, AWS Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Capacity Manager, Images (AMIs, AMI Catalog), Elastic Block Store (Volumes, Snapshots, Lifecycle Manager), Network & Security (Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces), and Load Balancing. The main content area displays a table titled "Instances (4) Info" with columns: Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, Public IPv4 DNS, Public IPv4 MAC, and Elastic IP. The instances listed are:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 MAC	Elastic IP
i-0e95090e556111d12	i-0e95090e556111d12	Running	t3.small	3/3 checks passed	View alarms +	ap-southeast-1a	-	-	-
cicd-pipeline-jenkins	i-0df7e2ca8a201da9e	Running	c7i-flex.large	3/3 checks passed	View alarms +	ap-southeast-1a	ec2-13-228-28-131.ap...	13.228.28.131	-
cicd-pipeline-docker	i-00b30d2c405854abc	Running	t3.small	3/3 checks passed	View alarms +	ap-southeast-1b	-	-	-

Below the table, a section titled "Select an instance" is shown.

## EKS

The screenshot shows the AWS EKS Clusters page. The left sidebar navigation includes: Amazon Elastic Kubernetes Service (Dashboard, Clusters selected), Settings (Dashboard settings, Console settings), Amazon EKS Anywhere (Enterprise Subscriptions), and Related services (Amazon ECR, AWS Batch). The main content area displays a table titled "Clusters (1) Info" with columns: Cluster name, Status, Kubernetes version, Support period, Upgrade policy, Created, and Provider. The cluster listed is:

Cluster name	Status	Kubernetes version	Support period	Upgrade policy	Created	Provider
cicd-pipeline-cluster	Active	1.31 <a href="#">Upgrade now</a>	⚠ Standard support until November 26, 2025	Extended support	24 minutes ago	EKS

https://ap-southeast-1.console.aws.amazon.com/eks/clusters/cicd-pipeline-cluster?region=ap-southeast-1

Amazon Elastic Kubernetes Service > Clusters > cicd-pipeline-cluster

## cicd-pipeline-cluster

End of standard support for Kubernetes version 1.31 is November 26, 2025.

**Cluster info**

Status: Active	Kubernetes version: 1.31	Support period: Standard support until November 26, 2025	Provider: EKS
Cluster health: 0	Upgrade insights: 6	Node health issues: 0	

**Overview** | Resources | Compute | Networking | Add-ons | Access | Observability | Update history | Tags

**Details**

API server endpoint: https://9A770A8B363B32838F392C101828987E.gr7.ap-southeast-1.eks.amazonaws.com	OpenID Connect provider URL: https://oidc.eks.ap-southeast-1.amazonaws.com/id/9A770A8B363B32838F392C101828987E
Certificate authority: LSQhLS1CRU4UTIBDRVJUSUZ400FUR50tL50iCk1j5URCVENDQWUy20f35UJB20U1JaajlQnK2SV15RF725ktbWkd0mHOQVFTTEJRQXgGVEVUTUJFR0ExvUUKQXhNS2EzVmIaWEp1WhsbgN5Q	Cluster IAM role ARN: arn:aws:iam::17762962279:role/cicd-pipeline-eks-cluster-role <a href="#">View in IAM</a>
EKS Auto Mode: Disabled	Created: 25 minutes ago
	Cluster ARN: arn:aws:eks:ap-southeast-1:17762962279:cluster/cicd-pipeline-cluster
	Platform version: eks.44

**EKS Auto Mode**

EKS automates routine cluster tasks for compute, storage, and networking to meet application compute needs.

EKS Auto Mode: Disabled

Manage

CloudShell Feedback Console Mobile App

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

## II. Access Jenkins

Terraform has already created an EC2 instance with Jenkins installed. Access Jenkins at:

<http://13.228.28.131:8080/>

The screenshot shows the AWS EC2 Instances page. The instance summary for i-0df7e2ca8a201da9e (cicd-pipeline-jenkins) is displayed. Key details include:

- Public IPv4 address:** 13.228.28.131 (highlighted with a red box)
- Instance state:** Running
- Private IP DNS name (IPv4 only):** ip-10-0-3-185.ap-southeast-1.compute.internal
- Instance type:** c7i-flex-large
- VPC ID:** vpc-0184de0dfb2c42624 (cicd-pipeline-vpc)
- Subnet ID:** subnet-09efafbec835ccaf5 (cicd-pipeline-public-1)
- Instance ARN:** arn:aws:ec2:ap-southeast-1:177629262279:instance/i-0df7e2ca8a201da9e

The left sidebar shows the navigation menu for EC2, including Dashboard, AWS Global View, Events, Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Capacity Manager, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security, Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces, and Load Balancing.

The screenshot shows the Jenkins 'Getting Started' page titled 'Unlock Jenkins'. It instructs the user to copy the password from the log or file and paste it into the 'Administrator password' field. The password is listed as:

```
/var/lib/jenkins/secrets/initialAdminPassword
```

The page includes a 'Continue' button at the bottom right.

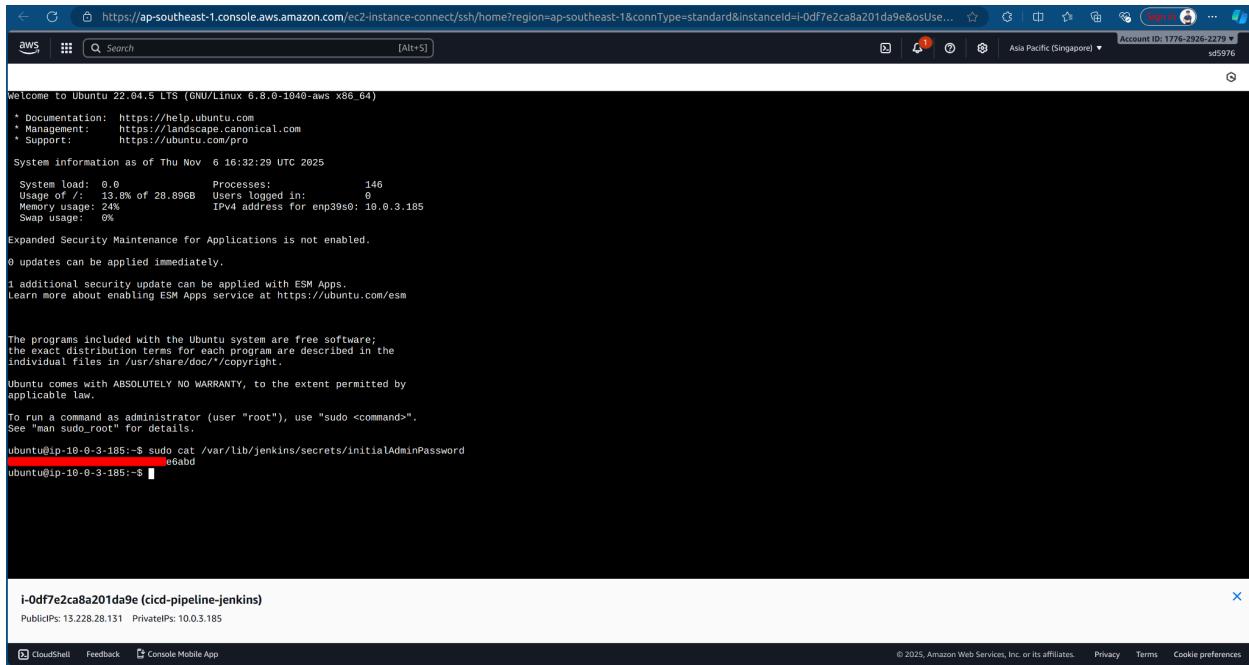
To get the initialAdminPassword, we need to connect to the EC2 instance **cicd-pipeline-jenkins**.

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with sections like Dashboard, AWS Global View, Events, Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Capacity Manager, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security, Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces, and Load Balancing. The main content area displays the instance summary for 'i-0df7e2ca8a201da9e (cicd-pipeline-jenkins)'. It includes details such as Instance ID (i-0df7e2ca8a201da9e), Public IPv4 address (13.228.28.131), Instance state (Running), Hostname type (IP name: ip-10-0-3-185.ap-southeast-1.compute.internal), Auto-assigned IP address (13.228.28.131 [Public IP]), IAM Role (c7n-flex-large), IMDSv2 (Optional), Operator (None), and various network and storage details. At the top right, there's a 'Connect' button, which is highlighted with a red box.

The screenshot shows the 'Connect to instance' dialog box. It has tabs for EC2 Instance Connect, Session Manager, SSH client, and EC2 serial console. The EC2 Instance Connect tab is active. It shows the instance ID (i-0df7e2ca8a201da9e (cicd-pipeline-jenkins)) and a connection type section. In the connection type section, 'Connect using a Public IP' is selected (indicated by a blue border and checked radio button), while 'Connect using a Private IP' is unselected. Below this, there are options for 'Public IPv4 address' (13.228.28.131) and 'IPv6 address' (None). A 'Username' field contains 'ubuntu'. A note at the bottom states: '(Note: In most cases, the default username, ubuntu, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.)'. At the bottom right, there are 'Cancel' and 'Connect' buttons, with 'Connect' being highlighted with a red box.

Retrieve the initial admin password:

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



```
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.8.0-1049-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

System information as of Thu Nov  6 16:32:29 UTC 2025

System load: 0.0          Processes:           146
Usage of /: 13.8% of 28.89GB   Users logged in: 0
Memory usage: 24%          IPv4 address for enp39s0: 10.0.3.185
Swap usage: 0%

Expanded Security Maintenance for Applications is not enabled.
0 updates can be applied immediately.

1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

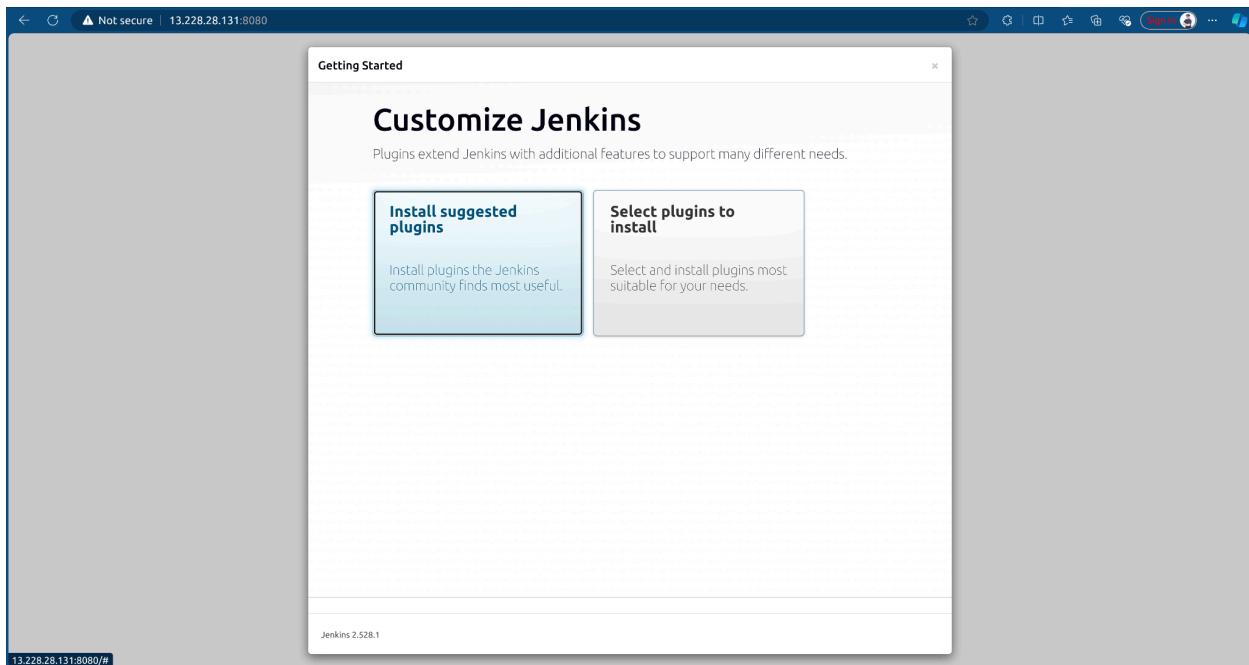
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

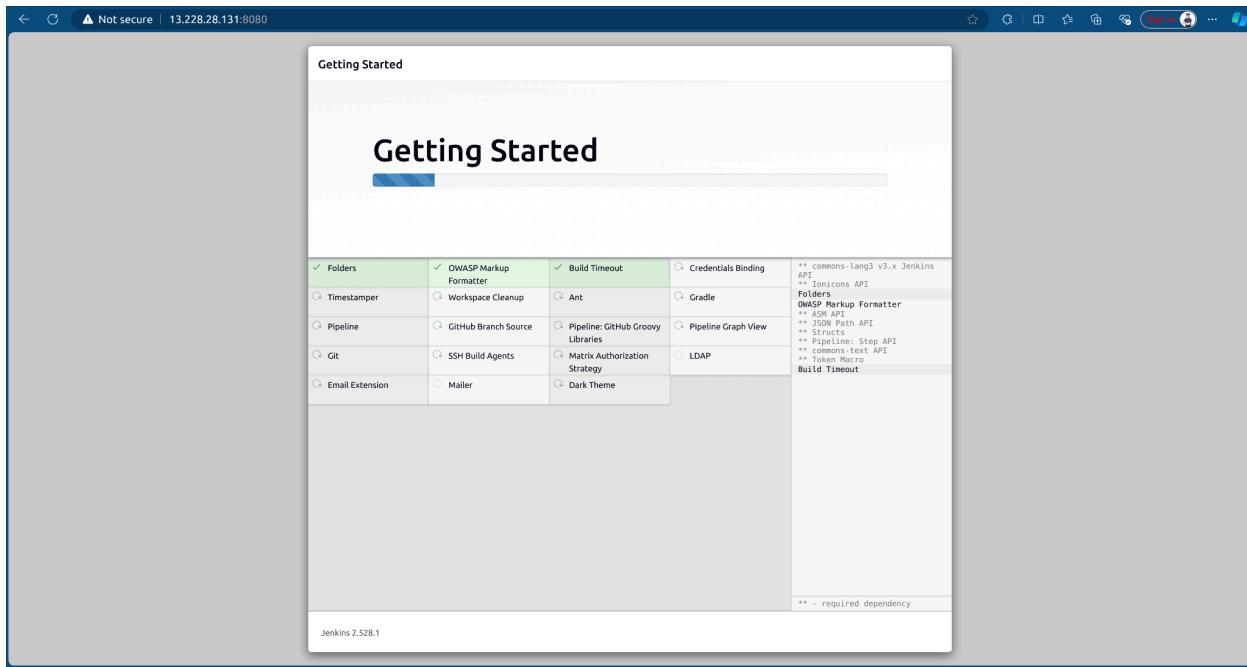
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

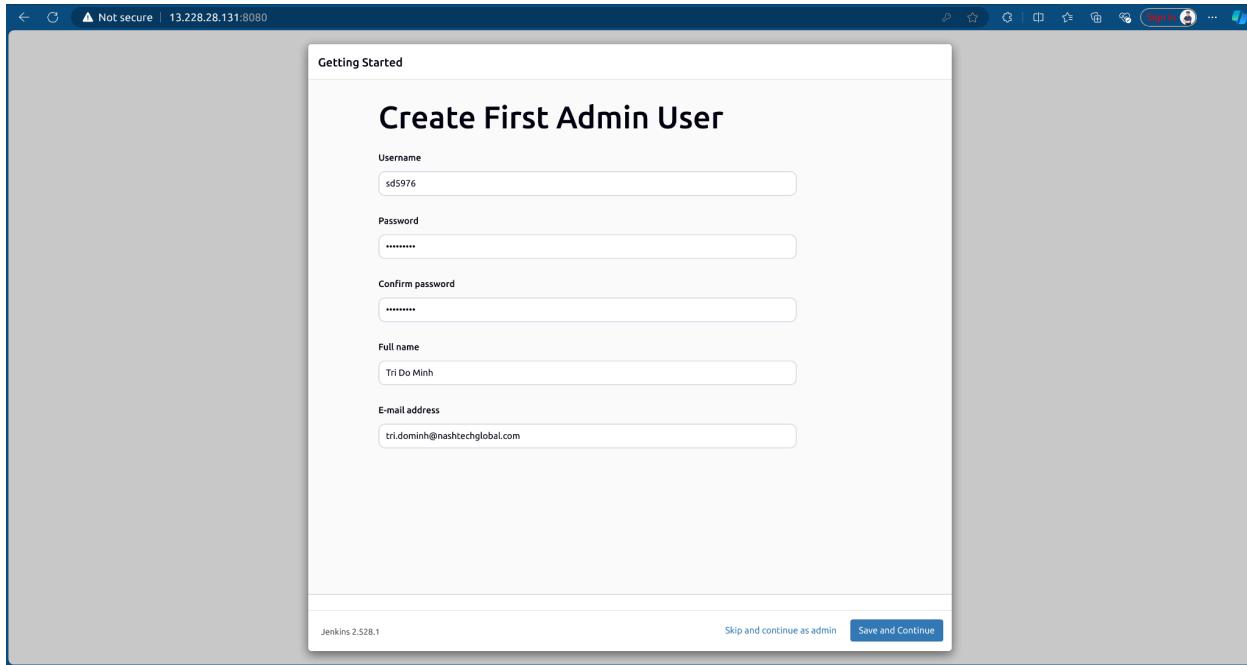
ubuntu@ip-10-0-3-185:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
e6abd
ubuntu@ip-10-0-3-185:~$ 
```

Copy the password and paste it into Jenkins UI, then select **Install suggested plugins**.





After installation, create an Admin User to log in to the dashboard.



The screenshot shows the Jenkins home page at [13.228.28.131:8080](http://13.228.28.131:8080). The top navigation bar includes links for 'Sign in' and 'Help'. The main content area features a 'Welcome to Jenkins!' message with a sub-instruction: 'This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.' Below this, there are sections for 'Start building your software project' (with a 'Create a job' button) and 'Set up a distributed build' (with links for 'Set up an agent', 'Configure a cloud', and 'Learn more about distributed builds'). On the left sidebar, there are links for 'New Item', 'Build History', 'Build Queue' (showing 'No builds in the queue.'), and 'Build Executor Status' (showing '9/2'). At the bottom right, there are links for 'REST API' and 'Jenkins 2.528.1'.

To run CI/CD, we need to install some plugins:

- Docker
- Docker Pipeline
- Pipeline: AWS Steps

Go to **Manage Jenkins → Plugins → Available plugins** tab.

The screenshot shows the 'Available plugins' tab in the Jenkins plugin manager. The left sidebar has links for 'Updates', 'Available plugins' (which is selected and highlighted in grey), 'Installed plugins', 'Advanced settings', and 'Download progress'. The main area displays a table of available plugins. The 'Docker' and 'Docker Pipeline' plugins are selected for installation, indicated by a checked checkbox next to their names. Other visible plugins include 'Pipeline Graph Analysis', 'PAM Authentication', 'JavaMail API', 'Command Agent Launcher', 'Oracle Java SE Development Kit Installer', 'SSH server', and 'Pipeline: REST API'. The table columns are 'Install', 'Name', 'Released', and 'Health'. A red box highlights the 'Install' button in the top right corner of the table header.

## Restart Jenkins after plugin installation.

Not secure | 13.228.28.131:8080/manage/pluginManager/updates/

Jenkins / Manage Jenkins / Plugins

Plugins

Updates Available plugins Installed plugins Advanced settings Download progress

Plugin	Status
Commons Compress API	Success
Docker API	Success
Docker	Success
Docker Pipeline	Success
Loading plugin extensions	Success
Amazon Web Services SDK : Minimal	Success
Amazon Web Services SDK : SQS	Success
Amazon Web Services SDK : SNS	Success
Amazon Web Services SDK : Api Gateway	Success
Amazon Web Services SDK 2 : Core	Success
Amazon Web Services SDK 2 : EC2	Success
Amazon Web Services SDK : EC2	Success
AWS Credentials	Success
Amazon Web Services SDK : CloudFormation	Success
Amazon Web Services SDK : Elastic Beanstalk	Success
Amazon Web Services SDK : Elastic Load Balancing V2	Success
Amazon Web Services SDK : IAM	Success
Amazon Web Services SDK : ECR	Success
Amazon Web Services SDK : CloudFront	Success
Amazon Web Services SDK : Lambda	Success
Amazon Web Services SDK : Organizations	Success
Amazon Web Services SDK : CodeDeploy	Success
Pipeline: AWS Steps	Success
Loading plugin extensions	Success

→ Go back to the top page  
(you can start using the installed plugins right away)

→  Restart Jenkins when installation is complete and no jobs are running

REST API Jenkins 2.528.1

Not secure | 13.228.28.131:8080/manage/pluginManager/updates/

Jenkins is restarting

Your browser will reload automatically when Jenkins is ready.

**Safe Restart**  
Builds on agents can usually continue.

### III. Deploying the MSA Application to EKS with Jenkins CI/CD

Prepare the MSA repository containing *frontend* and *backend*:

[https://github.com/tri-dominh/sd5976\\_msa](https://github.com/tri-dominh/sd5976_msa)

The screenshot shows the GitHub repository page for 'sd5976\_msa'. The main branch has one commit from 'tri-dominh' titled 'Init project' made at 21680f8 1 minute ago. The commit includes several files: 'backend', 'frontend', 'react-express-mongodb', '.gitignore', 'README.md', 'compose.yaml', and 'output.png'. All files are listed as 'Init project'.

Each source (*frontend/backend*) should have a **Jenkinsfile** defining agent and stages: install, build, validate, test, and push to ECR.

[https://github.com/tri-dominh/sd5976\\_msa/blob/main/backend/Jenkinsfile](https://github.com/tri-dominh/sd5976_msa/blob/main/backend/Jenkinsfile)

[https://github.com/tri-dominh/sd5976\\_msa/blob/main/frontend/Jenkinsfile](https://github.com/tri-dominh/sd5976_msa/blob/main/frontend/Jenkinsfile)

The screenshot shows the GitHub repository page for 'sd5976\_msa' with the 'main' branch selected. The 'backend' directory is expanded, showing its contents. Inside 'backend', there is a 'Jenkinsfile' highlighted with a red box. The right pane displays the commit history for this file, which is identical to the main branch's commit history: 'Init project' at 21680f8 1 minute ago. Below the commit table, a snippet of the Jenkinsfile is shown:

```
Snippet of backend(Node.js) DockerFile
You will find this Dockerfile file in the root directory of the project.
FROM node:13.13.0-stretch-slim
```

Next, we need to configure Jenkins to access GitHub and AWS resources.

Create a GitHub Access Token for Jenkins to access the MSA repository.

Go to **Settings** → **Developer settings** → **Personal access tokens** → **Tokens (classic)** → **Generate new token (classic)** → Enter token name and select scopes to allow Jenkins access to GitHub → Create.

The screenshot shows the GitHub profile settings page at <https://github.com/settings/profile>. The 'Developer settings' section is highlighted with a red box. It contains fields for 'Company' (NashTech), 'Location', and 'Display current local time'. There's also an 'ORCID ID' section and a 'Connect your ORCID ID' button. Below these are optional fields for hiding activity and showing achievements. At the bottom are 'Update profile' and 'Contributions & activity' sections.

The screenshot shows the GitHub developer settings page at <https://github.com/settings/tokens>. The 'Personal access tokens (classic)' section is highlighted with a red box. It shows a list of generated tokens and a button to 'Generate new token'. A sub-section titled 'Generate new token (classic)' is also highlighted with a red box, describing it as 'Fine-grained, repo-scoped' and 'For general use'. The left sidebar includes 'GitHub Apps', 'OAuth Apps', and 'Personal access tokens' (which is selected). The bottom navigation bar includes links for Terms, Privacy, Security, Status, Community, Docs, Contact, Manage cookies, and Do not share my personal information.

<https://github.com/settings/tokens/new>

Settings / Developer Settings

GitHub Apps  
OAuth Apps  
Personal access tokens  
Fine-grained tokens  
Tokens (classic)

New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

Note: sd5976-practical-devops

What's this token for?

Expiration: 30 days (Dec 07, 2025)

The token will expire on the selected date.

Select scopes: Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

<input type="checkbox"/> repo	Full control of private repositories
<input type="checkbox"/> repo:status	Access commit status
<input type="checkbox"/> repo_deployment	Access deployment status
<input type="checkbox"/> public_repo	Access public repositories
<input type="checkbox"/> repository	Access repository invitations
<input type="checkbox"/> security_events	Read and write security events
<input checked="" type="checkbox"/> workflow	Update GitHub Action workflows
<input type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input type="checkbox"/> read:packages	Download packages from GitHub Package Registry
<input type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input checked="" type="checkbox"/> admin:org	Full control of orgs and teams, read and write org projects
<input type="checkbox"/> write:org	Read and write org and team membership, read and write org projects
<input type="checkbox"/> read:org	Read org and team membership, read org projects
<input type="checkbox"/> manage_runners:org	Manage org runners and runner groups
<input checked="" type="checkbox"/> admin:public_key	Full control of user public keys
<input type="checkbox"/> write:public_key	Write user public keys
<input type="checkbox"/> read:public_key	Read user public keys

<https://github.com/settings/tokens/new>

<input type="checkbox"/> security_events	Read and write security events
<input checked="" type="checkbox"/> workflow	Update GitHub Action workflows
<input type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input type="checkbox"/> read:packages	Download packages from GitHub Package Registry
<input type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input checked="" type="checkbox"/> admin:org	Full control of orgs and teams, read and write org projects
<input type="checkbox"/> write:org	Read and write org and team membership, read and write org projects
<input type="checkbox"/> read:org	Read org and team membership, read org projects
<input type="checkbox"/> manage_runners:org	Manage org runners and runner groups
<input checked="" type="checkbox"/> admin:public_key	Full control of user public keys
<input type="checkbox"/> write:public_key	Write user public keys
<input type="checkbox"/> read:public_key	Read user public keys
<input checked="" type="checkbox"/> admin:repo_hook	Full control of repository hooks
<input type="checkbox"/> write:repo_hook	Write repository hooks
<input type="checkbox"/> read:repo_hook	Read repository hooks
<input checked="" type="checkbox"/> admin:org_hook	Full control of organization hooks
<input type="checkbox"/> gist	Create gists
<input type="checkbox"/> notifications	Access notifications
<input type="checkbox"/> user	Update ALL user data
<input type="checkbox"/> read:user	Read ALL user profile data
<input type="checkbox"/> user:email	Access user email addresses (read-only)
<input type="checkbox"/> user:follow	Follow and unfollow users
<input type="checkbox"/> delete_repo	Delete repositories
<input type="checkbox"/> write:discussion	Read and write team discussions
<input type="checkbox"/> read:discussion	Read team discussions
<input type="checkbox"/> admin:enterprise	Full control of enterprises
<input type="checkbox"/> manage_runners:enterprise	Manage enterprise runners and runner groups
<input type="checkbox"/> manage_billing:enterprise	Read and write enterprise billing data
<input type="checkbox"/> read:enterprise	Read enterprise profile data
<input type="checkbox"/> scm:enterprise	Provisioning of users and groups via SCIM

The screenshot shows the GitHub Developer Settings page at <https://github.com/settings/tokens>. The left sidebar is titled "Developer Settings" and includes options for "GitHub Apps", "OAuth Apps", "Personal access tokens", "Fine-grained tokens", and "Tokens (classic)". The "Tokens (classic)" option is selected. The main area is titled "Personal access tokens (classic)" and contains a button "Generate new token". Below it, a table lists a single token:

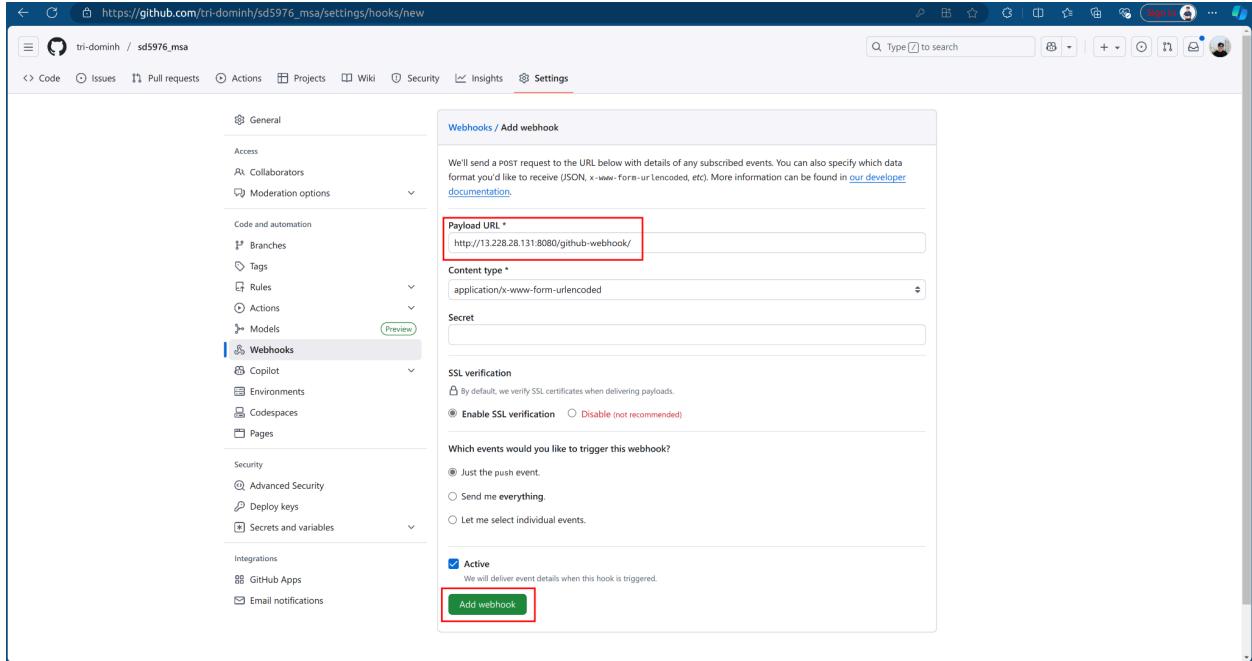
sd5976-practical-devops	admin:org, admin:org_hook, admin:public_key, admin:repo_hook, repo, workflow	Never used	Delete
Expires on Sun, Dec 7 2025.			

A note below the table states: "Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication."

At the bottom of the page, there is a footer with links: © 2025 GitHub, Inc. Terms Privacy Security Status Community Docs Contact Manage cookies Do not share my personal information.

This screenshot is identical to the one above, but it includes a prominent blue warning message box at the top of the token list area: "Make sure to copy your personal access token now. You won't be able to see it again!"

Then, we create a Github webhook to trigger Jenkins whenever we push new code. In the **sd5976-msa** repo, go to **Settings → Webhooks → Add webhook**, enter the Jenkins URL with */github-webhook/*



After that, we create the credentials in Jenkins which allow Jenkins access to Resources. Go to **Jenkins Credentials → Click on (global) → Add Credentials (Secret text kind)**.

- AWS Account ID, AWS Access Key ID, AWS Secret Access Key.
- GitHub Token.

Not secure | 13.228.28.131:8080/manage/ Jenkins / Manage Jenkins

Manage Jenkins

Building on the built-in node can be a security issue. You should set up distributed builds. See [the documentation](#).

System Configuration

- System: Configure global settings and paths.
- Tools: Configure tools, their locations and automatic installers.
- Plugins: Add, remove, disable or enable plugins that can extend the functionality of Jenkins.
- Nodes: Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

Docker: Plugin for launching build Agents as Docker containers.

Clouds: Add, remove, and configure cloud instances to provision agents on-demand.

Appearance: Configure the look and feel of Jenkins.

Security

- Security: Secure Jenkins; define who is allowed to access/use the system.
- Credentials: Configure credentials (highlighted with a red box).
- Credential Providers: Configure the credential providers and types.
- Users: Create/delete/modify users that can log in to this Jenkins.

Status Information

- System Information: Displays various environmental information to assist trouble-shooting.
- System Log: System log captures output from java.util.logging output related to Jenkins.
- Load Statistics: Check your resource utilization and see if you need more computers for your builds.
- About Jenkins: See the version and license information.

Troubleshooting

- Manage Old Data: Scrub configuration files to remove remnants from old plugins and earlier versions.

Set up agent Set up cloud Dismiss

Search settings

Not secure | 13.228.28.131:8080/manage/credentials/ Jenkins / Manage Jenkins / Credentials

Credentials

Stores scoped to Jenkins

T	P	Store	Domain	ID	Name

Icon: S M L

Domains (global) (highlighted with a red box)

REST API Jenkins 2.528.1

Not secure | 13.228.28.131:8080/manage/credentials/store/system/domain/\_/ Jenkins / Manage Jenkins / Credentials / System / Global credentials (unrest...)

**Global credentials (unrestricted)**

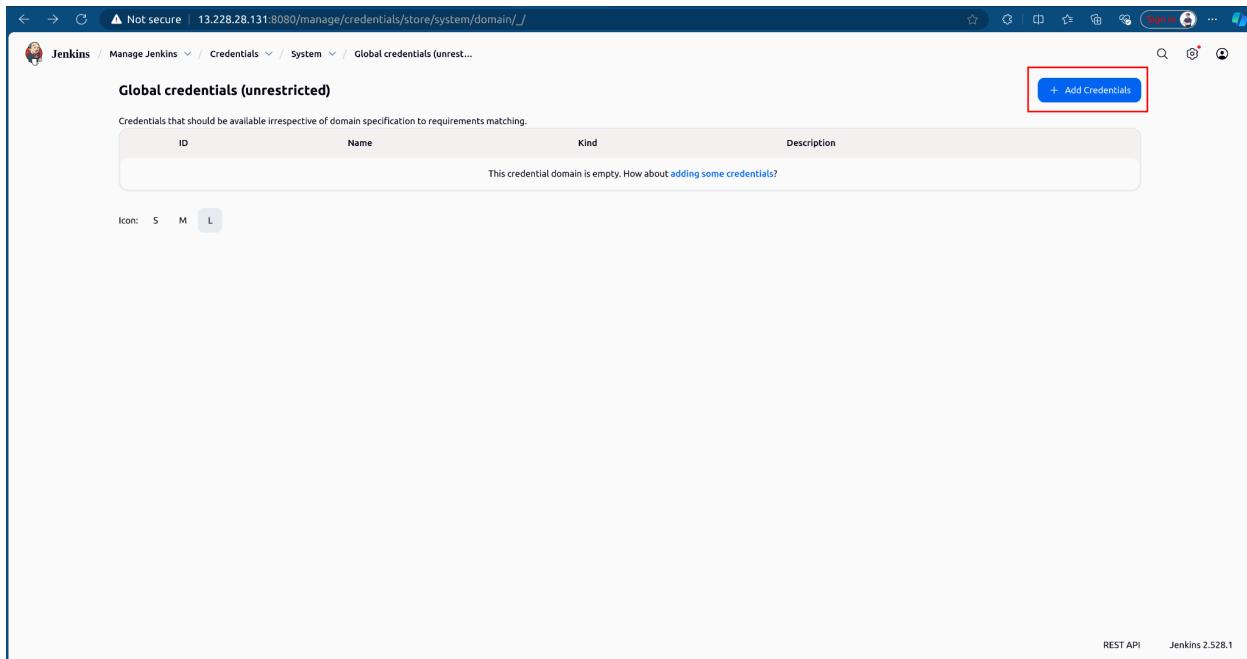
Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
This credential domain is empty. How about <a href="#">adding some credentials?</a>			

Icon: S M L

+ Add Credentials

REST API Jenkins 2.528.1



Not secure | 13.228.28.131:8080/manage/credentials/store/system/domain/\_/ Jenkins / Manage Jenkins / Credentials / System / Global credentials (unrest...)

**Global credentials (unrestricted)**

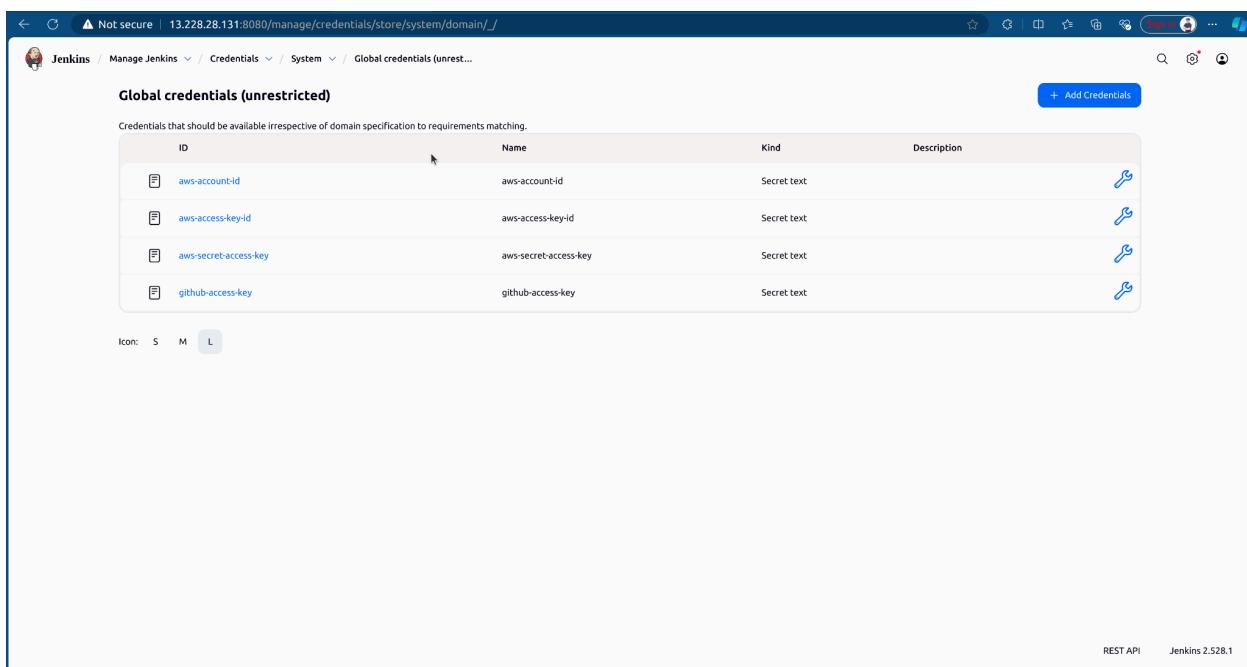
Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
aws-account-id	aws-account-id	Secret text	
aws-access-key-id	aws-access-key-id	Secret text	
aws-secret-access-key	aws-secret-access-key	Secret text	
github-access-key	github-access-key	Secret text	

Icon: S M L

+ Add Credentials

REST API Jenkins 2.528.1



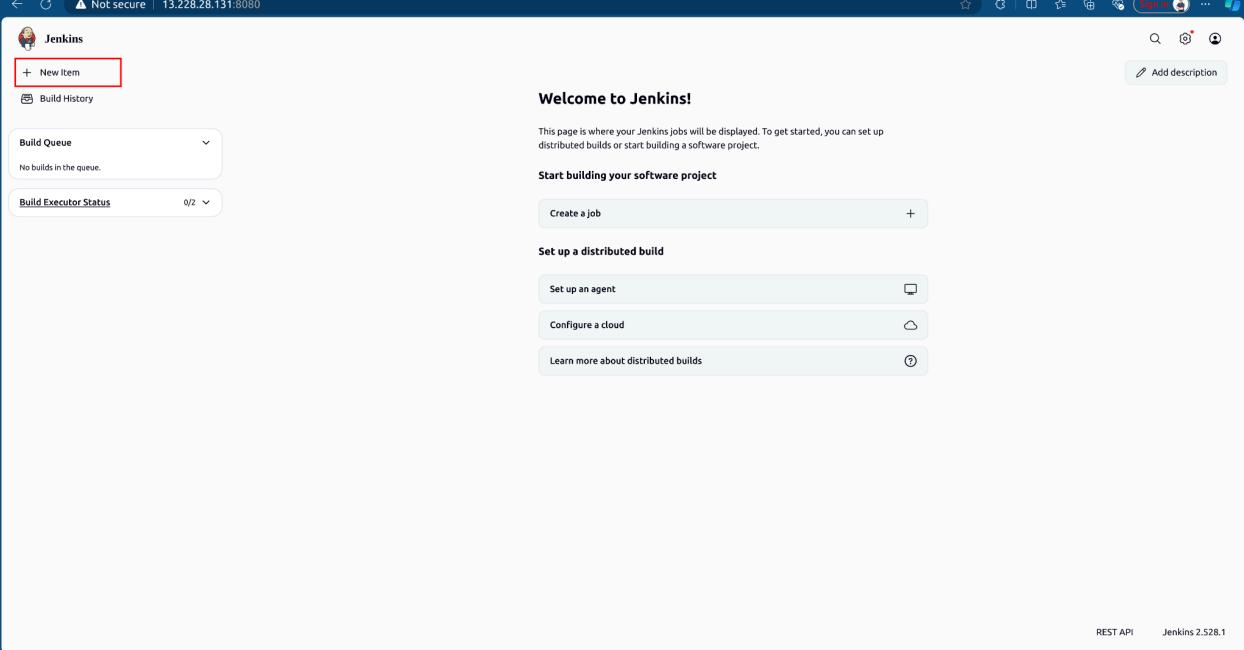
Add GitHub Server for Jenkins, go to **Settings** → **System** → In **GitHub** section → **Add GitHub Server** → Enter name, select credentials **github-access-key** and tick **Manage hooks** → Save.

The screenshot shows the Jenkins 'Manage Jenkins' interface at the URL <http://13.228.28.131:8080/manage/>. The 'System Configuration' section is highlighted with a red box around the 'System' icon. Other sections shown include Tools, Clouds, Plugins, Appearance, Nodes, Security, Credentials, Credential Providers, and Users. The 'System' section contains links for 'System Information', 'System Log', 'Load Statistics', and 'About Jenkins'.

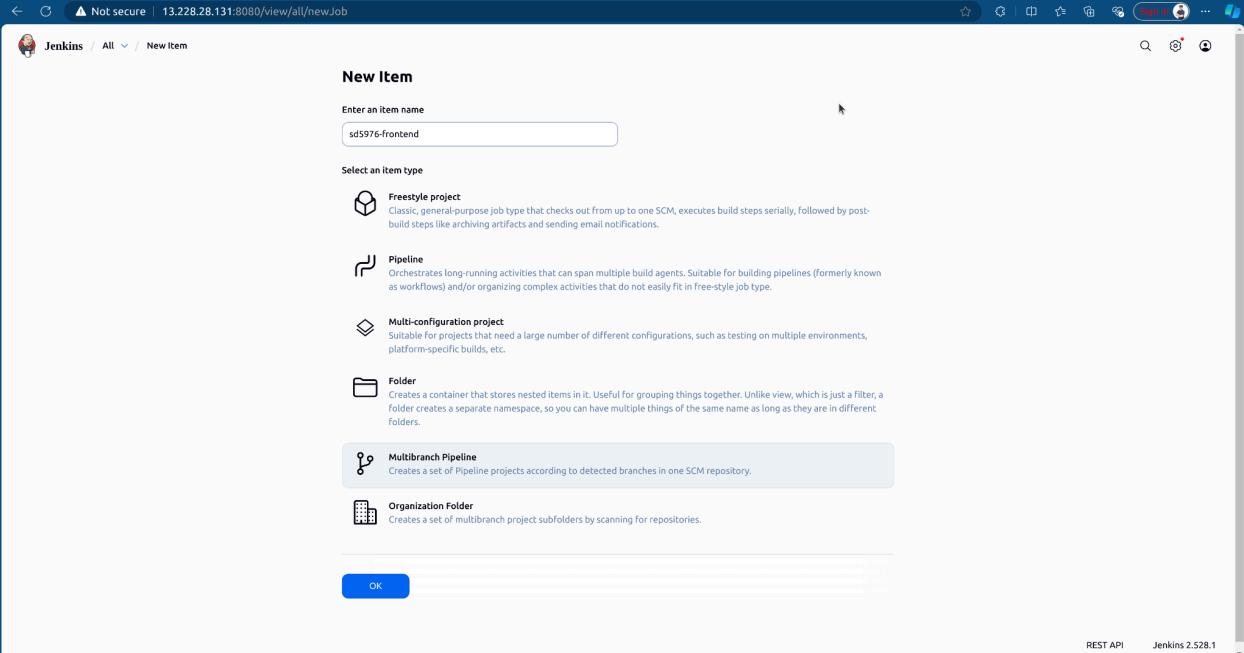
The screenshot shows the Jenkins 'Manage Jenkins' interface at the URL <http://13.228.28.131:8080/manage/configure>. The 'GitHub' section is expanded, showing the 'GitHub Servers' configuration. A new server named 'sd5976-github-msa' is being added. The 'Name' field is highlighted with a red box. The 'API URL' field contains 'https://api.github.com'. The 'Credentials' dropdown is set to 'github-access-key', which is also highlighted with a red box. The 'Manage hooks' checkbox is checked and highlighted with a red box. Below the form are 'Advanced' and '+ Add GitHub Server' buttons. At the bottom, there are 'GitHub API usage' buttons for 'Save' and 'Apply', with 'Save' highlighted with a red box.

## IV. Creating CI jobs for MSA Application

First, we build a job for the *Frontend*, click **New item** → Enter name and select **Multibranch pipeline job**.



The screenshot shows the Jenkins dashboard at the URL <http://13.228.28.131:8080>. The 'New Item' button is highlighted with a red box. The dashboard includes sections for 'Build Queue' (No builds in the queue), 'Build Executor Status' (0/2), and 'Welcome to Jenkins!' which provides instructions for setting up distributed builds or starting a software project. There are also links for 'Create a job', 'Set up an agent', 'Configure a cloud', and 'Learn more about distributed builds'.



The screenshot shows the 'New Item' creation dialog at the URL <http://13.228.28.131:8080/view/all/newJob>. The 'Item name' field contains 'sd5976-frontend'. The 'Select an item type' section lists several options: 'Freestyle project' (classic job type), 'Pipeline' (orchestrates long-running activities), 'Multi-configuration project' (suitable for testing on multiple environments), 'Folder' (creates a container for nested items), 'Multibranch Pipeline' (selected, creates a set of Pipeline projects according to detected branches in one SCM repository), and 'Organization Folder' (creates a set of multibranch project subfolders by scanning for repositories). A blue 'OK' button is at the bottom.

Configure Jenkins job to point to the *Frontend* branch and Jenkinsfile of *Frontend*.

The screenshot shows the Jenkins job configuration page for 'sd5976-frontend'. The 'General' section is active, displaying the 'Display Name' field with 'sd5976-frontend' and the 'Enabled' switch turned on. The 'Branch Sources' section contains a 'Git' configuration with a 'Project Repository' set to 'https://github.com/tri-dominh/sd5976\_msa.git'. The 'Build Configuration' section is visible at the bottom.

The screenshot shows the Jenkins job configuration page for 'sd5976-frontend'. The 'Build Configuration' section is active, displaying the 'Mode' dropdown set to 'by Jenkinsfile' and the 'Script Path' input field containing 'frontend/Jenkinsfile'. The 'Orphaned Item Strategy' section is also visible at the bottom.

After creating, the job will run Jenkins based on the Jenkinsfile and source.

Jenkins / sd5976-frontend / main / #1 / Pipeline Overview

#1

21680f8 Started 2 min 4 sec ago Queued 6.7 sec Took 2 min 4 sec and counting

Graph

Start Checkout SCM Checkout Install Dependencies Lint Test Build Application Build Docker Image Login to AWS ECR Push to ECR Post Actions End

Post Actions

Checkout SCM 1.0s  
Checkout 0.73s  
Install Dependencies 1m 20s  
Lint 34ms  
Test 30ms  
Build Application 9.3s  
Build Docker Image 1m 54s  
Login to AWS ECR 3.0s  
Push to ECR 30s  
Post Actions 0.36s

Cleaning up...  
docker rmi \${AWS\_ACCOUNT\_ID}.dkr.ecr.ap-southeast-1.amazonaws.com/sd5976-frontend:1 || true docker rmi \${AWS\_ACCOUNT\_ID}.dkr.ecr.ap-southeast-1.amazonaws.co...  
Frontend Pipeline completed successfully!  
Image: \${AWS\_ACCOUNT\_ID}.dkr.ecr.ap-southeast-1.amazonaws.com/sd5976-frontend:1

Warning: A secret was passed to "echo" using Groovy String Interpolation, which is insecure.  
Affected argument(s) used the following variable(s): \${AWS\_ACCOUNT\_ID}  
See <https://jenkins.io/redirect/groovy-string-interpolation> for details.  
Image: \*\*\*\*.dkr.ecr.ap-southeast-1.amazonaws.com/sd5976-frontend:1

0.36s Started 19s ago Jenkins Jenkins 2.528.1

CI will run and push the *Frontend* image to ECR.

https://ap-southeast-1.console.aws.amazon.com/ecr/repositories/private/177629262279/sd5976-frontend?region=ap-southeast-1

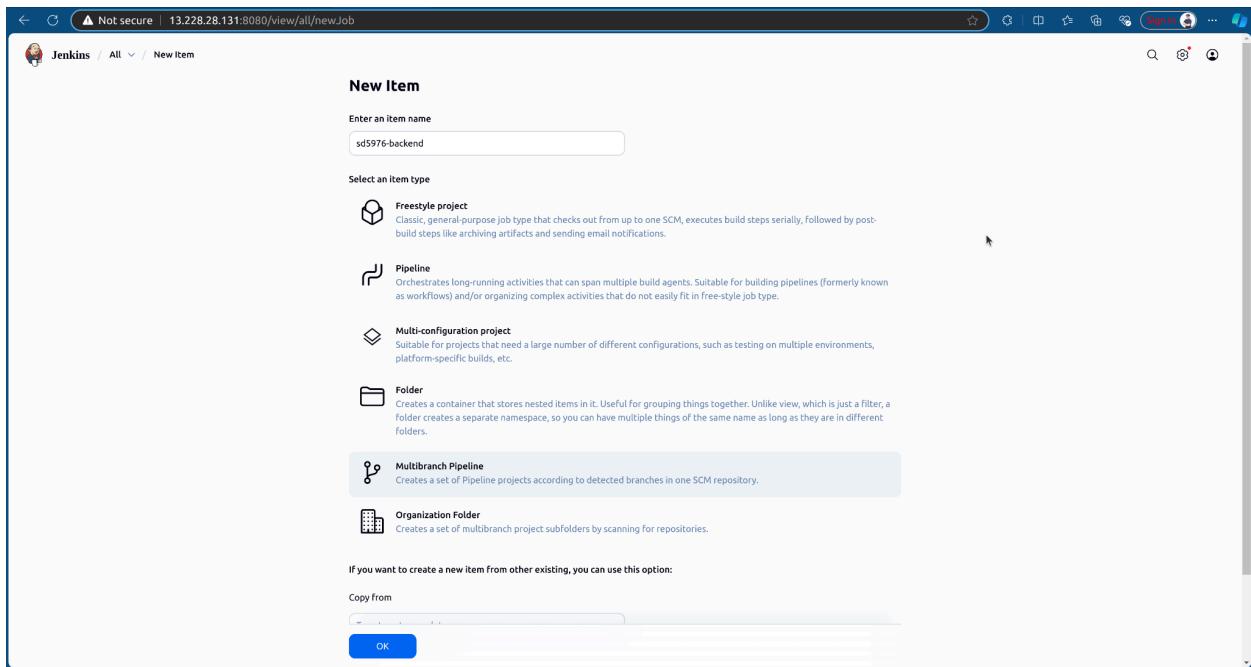
Amazon ECR > Private registry > Repositories > sd5976-frontend

Images (1)

Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest	Last recorded pull time
latest_1	Image	November 07, 2025, 00:42:22 (UTC+07)	486.73	<a href="#">Copy URI</a>	<a href="#">sha256:a60aedf80c394e4a90016a70e9c612...</a>	-

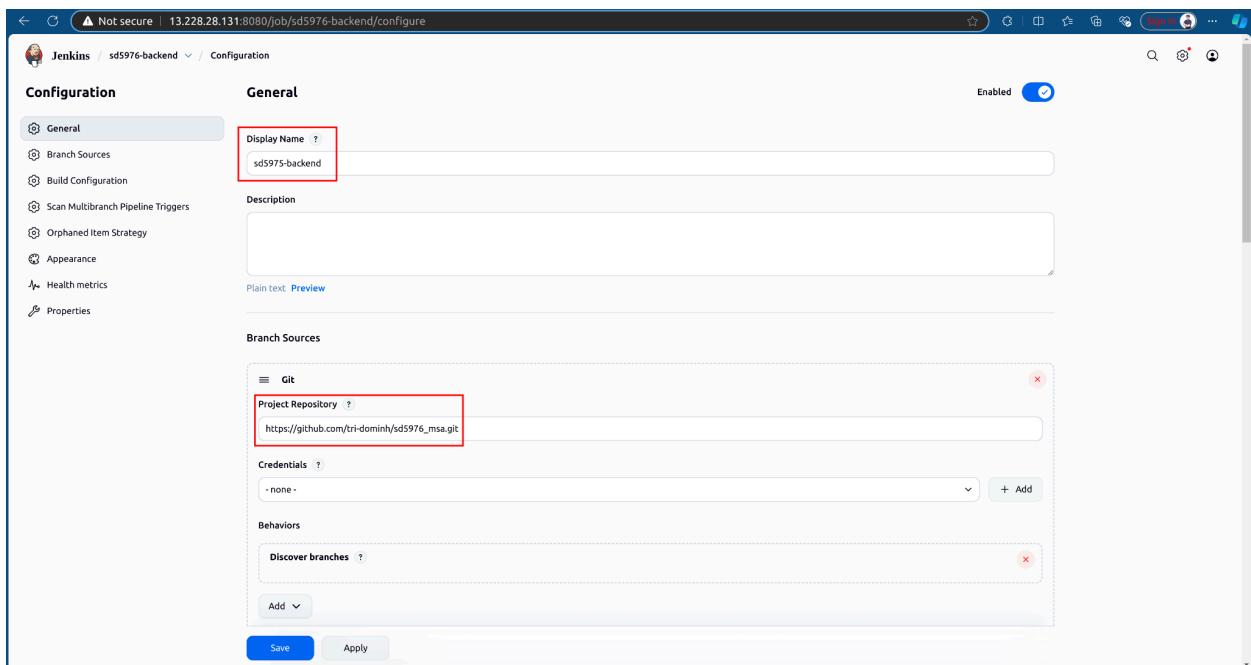
CloudShell Feedback Console Mobile App © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Next, we will create a similar job for the *Backend*, click **New item** → Enter name and select **Multibranch pipeline** job.



The screenshot shows the Jenkins 'New Item' creation interface. In the 'Enter an item name' field, the text 'sd5976-backend' is entered. Under 'Select an item type', the 'Multibranch Pipeline' option is selected, indicated by a blue background. Other options shown include 'Freestyle project', 'Pipeline', 'Multi-configuration project', 'Folder', and 'Organization Folder'. A note at the bottom left says 'If you want to create a new item from other existing, you can use this option:' followed by a 'Copy from' dropdown and 'OK' and 'Cancel' buttons.

Configure Jenkins job to point to *Backend* branch and Jenkinsfile of *Backend*.



The screenshot shows the Jenkins 'Configuration' page for the 'sd5976-backend' job. The 'General' tab is active. The 'Display Name' field contains 'sd5976-backend' with a red border. The 'Branch Sources' section is expanded, showing a 'Git' configuration with 'Project Repository' set to 'https://github.com/tri-dominh/sd5976\_msa.git' and 'Credentials' set to '-none-' with a red border. The 'Behaviors' section includes a 'Discover branches' field with a red border. At the bottom are 'Save' and 'Apply' buttons.

Jenkins / sd5976-backend / Configuration

**Build Configuration**

Mode: by Jenkinsfile

Script Path: backend/Jenkinsfile

**Scan Multibranch Pipeline Triggers**

Periodically if not otherwise run:

**Orphaned Item Strategy**

Jobs for removed SCM heads (i.e. deleted branches) can be removed immediately or kept based on a desired retention strategy. By default, jobs will be removed as soon as Jenkins determines their associated SCM head no longer exists. As an example, it may be useful to configure a different retention strategy to be able to examine build results of a branch after it has been removed.

Abort builds: ?

Discard old items

Days to keep old items: if not empty, old items are only kept up to this number of days

Max # of old items to keep: if not empty, only up to this number of old items are kept

Save Apply

After creating, the job will run Jenkins based on the Jenkinsfile and source.

Jenkins / sd5975-backend / main / #1 / Pipeline Overview

#1

21680FB Started 5.1 sec ago Queued 8.1 sec Took 5.1 sec and counting

Rerun

Graph

Start → Checkout SCM → Checkout → Install Dependencies → Lint → Test → Build Docker Image → Login to AWS ECR → Push to ECR → Post Actions → End

Post Actions

Cleaning up... 0.92s 19ms

docker rmi \${AWS\_ACCOUNT\_ID}.dkr.ecr.ap-southeast-1.amazonaws.com/sd5976-backend:1 || true docker rmi \${AWS\_ACCOUNT\_ID}.dkr.ecr.ap-southeast-1.amazonaws.com... 0.63s 0.28s

Backend Pipeline completed successfully! 13s 21ms

Image: \${AWS\_ACCOUNT\_ID}.dkr.ecr.ap-southeast-1.amazonaws.com/sd5976-backend:1 43ms 23ms

Warning: A secret was passed to "echo" using Groovy String interpolation, which is insecure.  
Affected argument(s) used the following variable(s): [AWS\_ACCOUNT\_ID]  
See <https://jenkins.io/redirect/groovy-string-interpolation> for details.

Image: \*\*\*\*.dkr.ecr.ap-southeast-1.amazonaws.com/sd5976-backend:1 24s 1.0s

Push to ECR 7.8s 0.38s

Jenkins 2.528.1

CI will run and push the *Backend* image to ECR.

The screenshot shows the AWS ECR console interface. On the left, there's a navigation sidebar with sections for 'Amazon Elastic Container Registry', 'Private registry', 'Public registry', and links to 'ECR public gallery', 'Amazon ECS', and 'Amazon EKS'. The main content area is titled 'Images (1)' and shows a single image entry: '1, latest' (Image type), pushed on November 07, 2025, at 00:50:40 (UTC+07), with a size of 89.78 MB. The image URI is sha256:25ae2677816fe9cd59599c01257d8... and the digest is sha256:25ae2677816fe9cd59599c01257d8... . There are buttons for 'Delete', 'Details', 'Scan', and 'View push commands'.

## V. Deployment

First, we need to add inbound rules to allow the Jenkins EC2 instance's security group to access the EKS cluster. Go to **cicd-pipeline-cluster** → **Networking** → Click on the security group → **Edit Inbound rules**.

The screenshot shows the AWS EKS console interface. The left sidebar includes 'Amazon Elastic Kubernetes Service', 'Clusters' (selected), 'Settings', 'Amazon EKS Anywhere', 'Related services', and 'Documentation'. The main content area is for the cluster 'cicd-pipeline-cluster'. It shows 'Cluster info' with status 'Active', Kubernetes version '1.31', support period until November 26, 2025, and provider 'EKS'. The 'Networking' tab is selected, highlighted with a red box. Under 'Networking', there are sections for 'VPC', 'Subnets', 'Cluster security group', 'Additional security groups', and 'API server endpoint access'. A note says 'Click "Monitor cluster" to view your cluster's observability data.' At the bottom, there's a section for 'Prometheus metrics' with a note about collecting metrics.

## Add HTTPS inbound rule from Jenkins EC2 instance's security group **cicd-pipeline-jenkins-sg** and save rules.

Screenshot of the AWS VPC console showing the "Edit inbound rules" page for a specific security group. The page displays two inbound rules:

Type	Protocol	Port range	Source	Description
All traffic	All	All	Custom (sg-08e63dc93d801f04e)	
HTTPS	TCP	443	Custom (sg-0a7bc3e4fd87f0c0c)	Use: sg-0a7bc3e4fd87f0c0c Security Groups: cicd-pipeline-jenkins-sg   sg-0a7bc3e4fd87f0c0c cicd-pipeline-jenkins-sg

At the bottom, there are "Preview changes" and "Save rules" buttons.

**VPC dashboard** screenshot showing the security group **sg-08e63dc93d801f04e - eks-cluster-sg-cicd-pipeline-cluster-329966485**. The "Inbound rules" section lists the same two rules as above, along with their details and descriptions.

Second, we create a new Multibranch Pipeline job. Refer to:

[https://github.com/tri-dominh/sd5976\\_devops\\_cicd\\_deployment/tree/main/deployment](https://github.com/tri-dominh/sd5976_devops_cicd_deployment/tree/main/deployment)

New Item

Enter an item name  
sd5976-deployment

Select an item type

- Freestyle project**  
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.
- Pipeline**  
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- Multi-configuration project**  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Folder**  
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a Folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different Folders.
- Multibranch Pipeline**  
Creates a set of Pipeline projects according to detected branches in one SCM repository.
- Organization Folder**  
Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:

Copy from

OK

Configure Jenkins to point to Deployment resource and Jenkinsfile.

Configuration General Enabled

Display Name  
sd5976-deployment

Description

Plain text Preview

Branch Sources

Git

Project Repository  
https://github.com/tri-dominh/sd5976\_devops\_cicd\_deployment.git

Credentials  
- none -

Behaviors

Discover branches

Add

Save Apply

Not secure | 13.228.28.131:8080/job/sd5976-deployment/configure

**Configuration**

**Build Configuration**

Mode: by Jenkinsfile

Script Path: deployment/Jenkinsfile

**Scan Multibranch Pipeline Triggers**

Periodically if not otherwise run

**Orphaned Item Strategy**

Abort builds

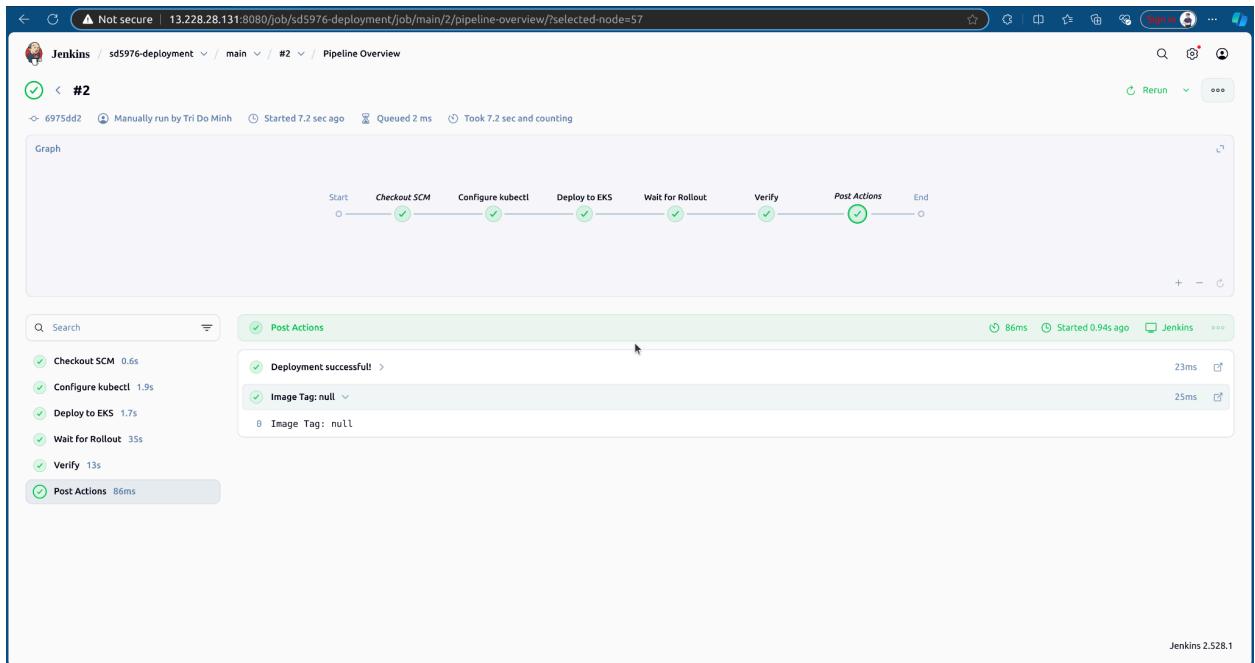
Discard old items

Days to keep old items: If not empty, old items are only kept up to this number of days

Max # of old items to keep: If not empty, only up to this number of old items are kept

**Save** **Apply**

After creating, the job will run Jenkins based on the Jenkinsfile and source.



Jenkins will deploy *Frontend & Backend* from ECR to EKS in **sd5976-msa** namespace.

Use **kubectl get pods -n sd5976-msa** to get all pods inside the namespace.

```
~/Desktop/sd5976_msa | main ➔ kubectl get pods -n sd5976-msa

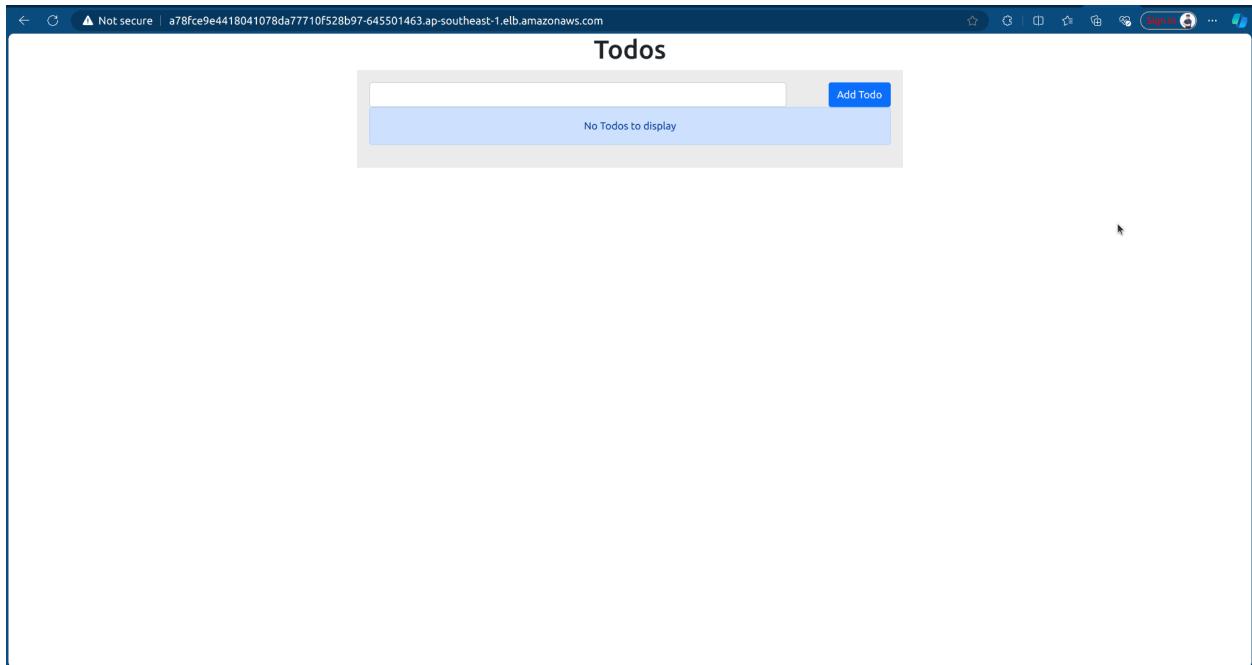
NAME                      READY   STATUS    RESTARTS   AGE
backend-cc7cd898b-2vj9q   1/1     Running   0          11m
backend-cc7cd898b-9sxcr   1/1     Running   0          11m
frontend-5c776bf9c5-rthpd 1/1     Running   0          11m
frontend-5c776bf9c5-tf6qk 1/1     Running   0          11m
mongodb-78547887ff-h686t  1/1     Running   0          11m

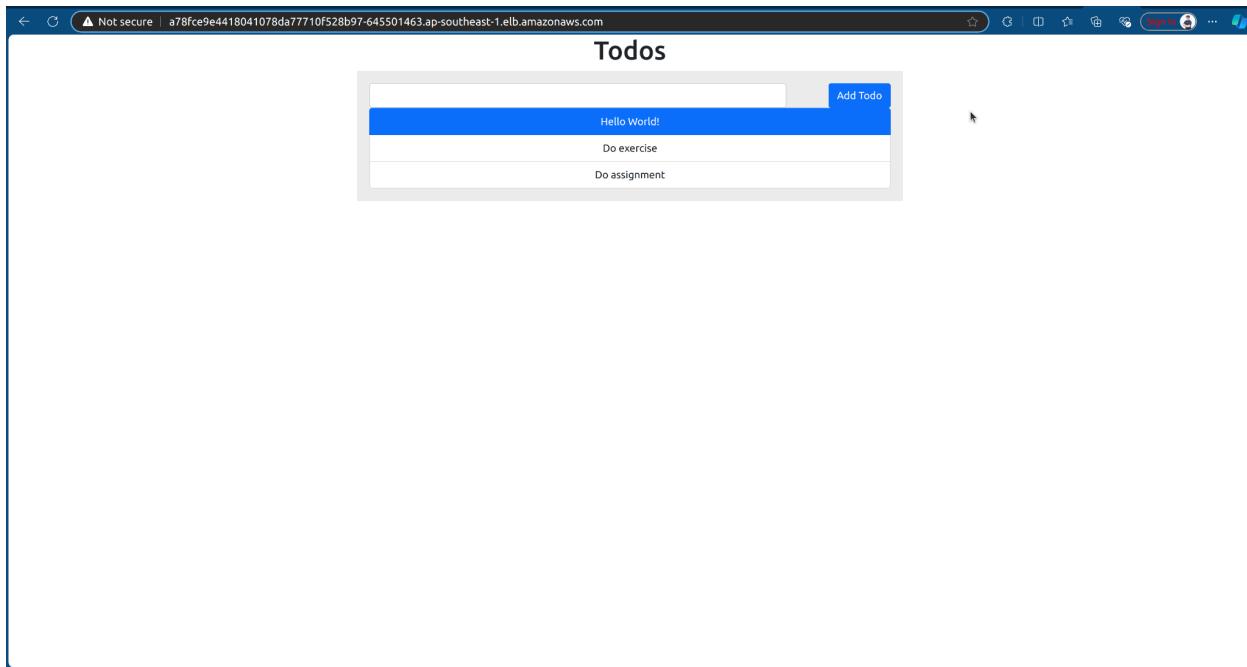
~/Desktop/sd5976_msa | main ➔
```

Use **kubectl get svc frontend -n sd5976-msa** to get the Frontend URL from EKS.

```
~/Desktop/sd5976_msa | main ➔ kubectl get svc frontend -n sd5976-msa
NAME      TYPE        CLUSTER-IP   EXTERNAL-IP
frontend  LoadBalancer  172.20.241.30  a78fce9e4418041078da77710f528b97-645501463.ap-southeast-1.elb.amazonaws.com  ✓ | base Py | cicd-pipeline-cluster o
PORT(S)   AGE
80:31416/TCP  12m
~/Desktop/sd5976_msa | main ➔
```

We can access the application using the frontend URL from EKS.





## VI. Monitoring with Prometheus and Grafana

Define .yaml file to deploy Prometheus and Grafana in sd5976\_aws\_infrastructure:

[https://github.com/tri-dominh/sd5976\\_aws\\_infrastructure/tree/main/monitoring](https://github.com/tri-dominh/sd5976_aws_infrastructure/tree/main/monitoring)

Deploy metrics-server to collect cluster metrics.

**kubectl apply -f**

<https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml>

```
~/Desktop/sd5976_aws_infrastructure | main ➜ kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml
serviceaccount/metrics-server created
clusterrole.rbac.authorization.k8s.io/system:aggregated-metrics-reader created
clusterrole.rbac.authorization.k8s.io/system:metrics-server created
rolebinding.rbac.authorization.k8s.io/metrics-server-auth-reader created
clusterrolebinding.rbac.authorization.k8s.io/metrics-server:system:auth-delegator created
clusterrolebinding.rbac.authorization.k8s.io/system:metrics-server created
service/metrics-server created
deployment.apps/metrics-server created
apiservice.apiregistration.k8s.io/v1beta1.metrics.k8s.io created
~/Desktop/sd5976_aws_infrastructure | main ➜
```

Deploy Prometheus and Grafana

**kubectl apply -f monitoring/prometheus.yaml**

**kubectl apply -f monitoring/grafana.yaml**

```

~/De/sd5976_aws_infrastructure | main ➔ kubectl apply -f monitoring/prometheus.yaml
pipeline-cluster o
namespace/monitoring created
serviceaccount/prometheus created
clusterrole.rbac.authorization.k8s.io/prometheus created
clusterrolebinding.rbac.authorization.k8s.io/prometheus created
configmap/prometheus-config created
deployment.apps/prometheus created
service/prometheus created

~/De/sd5976_aws_infrastructure | main ➔ kubectl apply -f monitoring/grafana.yaml
pipeline-cluster o
configmap/grafana-datasources created
deployment.apps/grafana created
service/grafana created

```

Forward Prometheus port from EKS to local.

**kubectl port-forward -n monitoring svc/prometheus 9090:9090**

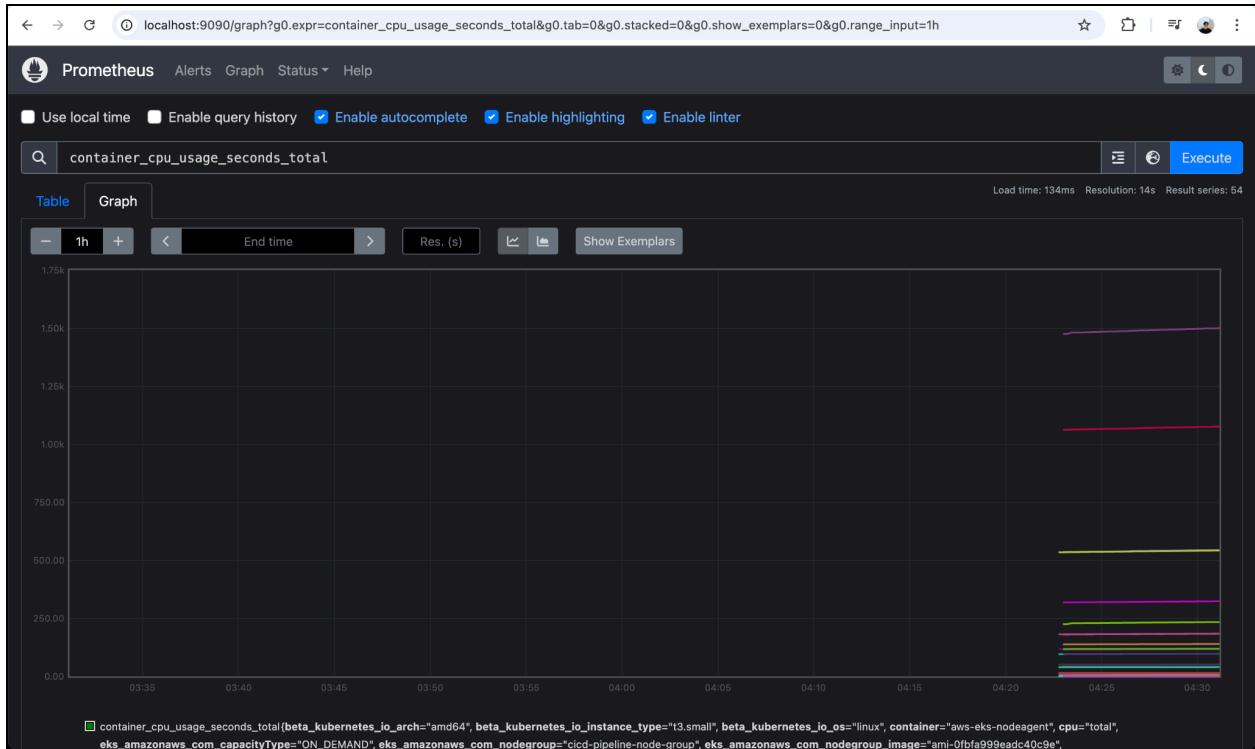
```

~/De/sd5976_aws_infrastructure | main ➔ kubectl port-forward -n monitoring svc/prometheus 9090:9090
Forwarding from 127.0.0.1:9090 -> 9090
Forwarding from [::1]:9090 -> 9090

```

We can monitor Prometheus via <http://localhost:9090>

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
https://kubernetes.default.svc/api/v1/nodes/ip-10-0-33-146.ap-southeast-1.compute.internal/proxy/metrics/cadvisor	UP	<pre> beta_kubernetes_io_arch="amd64" beta_kubernetes_io_instance_type="t3.small" beta_kubernetes_io_os="linux" eks.amazonaws.com_capacityType="ON_DEMAND" eks.amazonaws.com_nodegroup="cld-pipeline-node-group" eks.amazonaws.com_nodegroup_image="ami-0fbfa999eadc40c9e" failure_domain_beta_kubernetes_io_region="ap-southeast-1" failure_domain_beta_kubernetes_io_zones="ap-southeast-1a" instance="ip-10-0-33-146.ap-southeast-1.compute.internal" jobs="kubernetes-cadvisor" k8s_io_cloud_provider_aws="7af5e5d08af42c4fb1edf7dff94cfc" kubernetes_io_arch="amd64" kubernetes_io_hostname="ip-10-0-33-146.ap-southeast-1.compute.internal" kubernetes_io_os="linux" node="ip-10-0-33-146.ap-southeast-1.compute.internal" node_kubernetes_io_instance_type="t3.small" topology_kubernetes_io_zone_id="apse1-az2" topology_kubernetes_io_region="ap-southeast-1" topology_kubernetes_io_zone="ap-southeast-1a" </pre>	10.416s ago	44.250ms	
https://kubernetes.default.svc/api/v1/nodes/ip-10-0-62-100.ap-southeast-1.compute.internal/proxy/metrics/	UP	<pre> beta_kubernetes_io_arch="amd64" beta_kubernetes_io_instance_type="t3.small" beta_kubernetes_io_os="linux" </pre>	19.279s ago	34.162ms	



Forward Grafana port from EKS to local.

**kubectl port-forward -n monitoring svc/grafana 3000:3000**

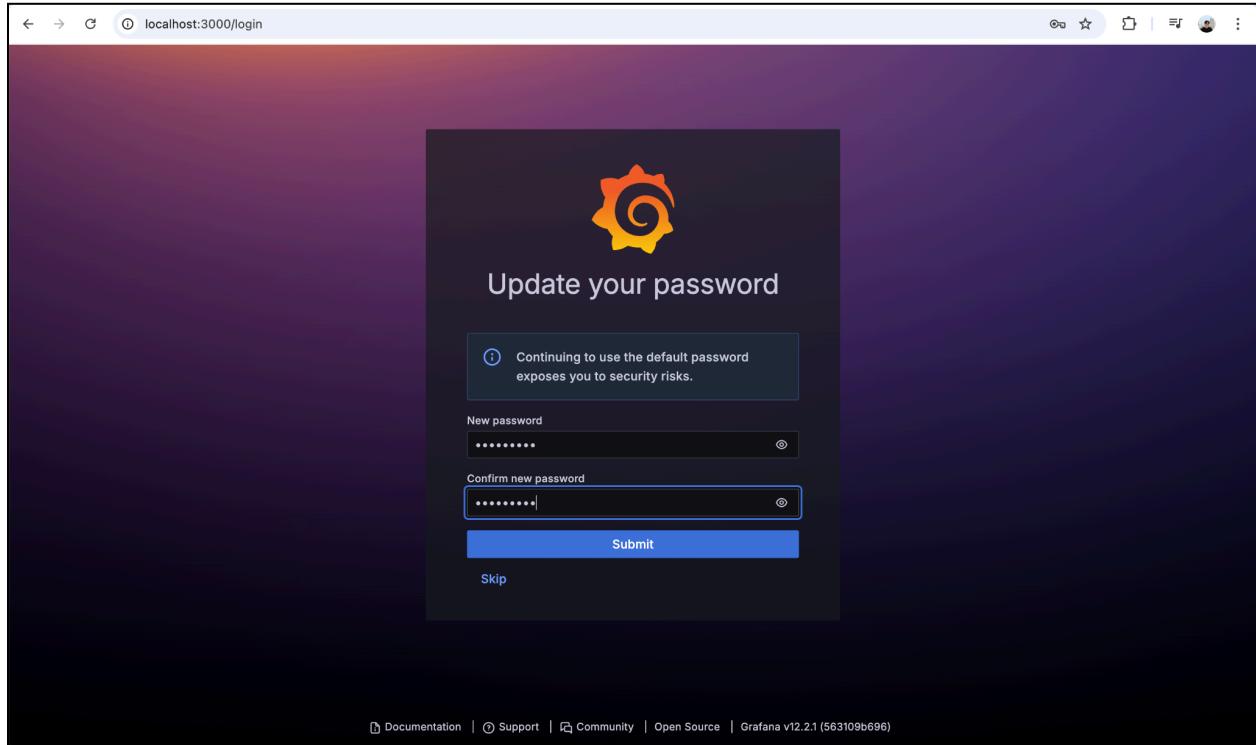
```
~/Desktop/sd5976_aws_infrastructure | main ➔ kubectl port-forward -n monitoring svc/grafana 3000:3000
Forwarding from 127.0.0.1:3000 -> 3000
Forwarding from [::1]:3000 -> 3000
█
```

We can access Grafana via <http://localhost:3000/login>

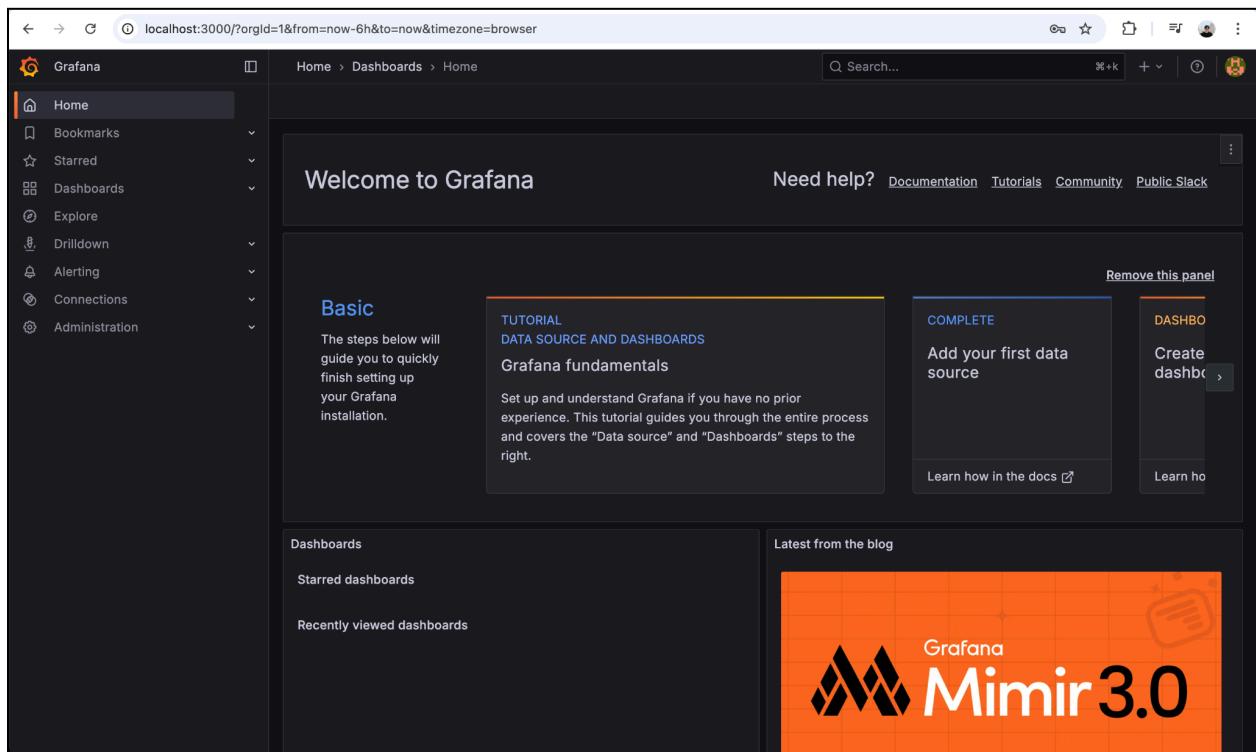
Default username: **admin**

Default password: **admin**

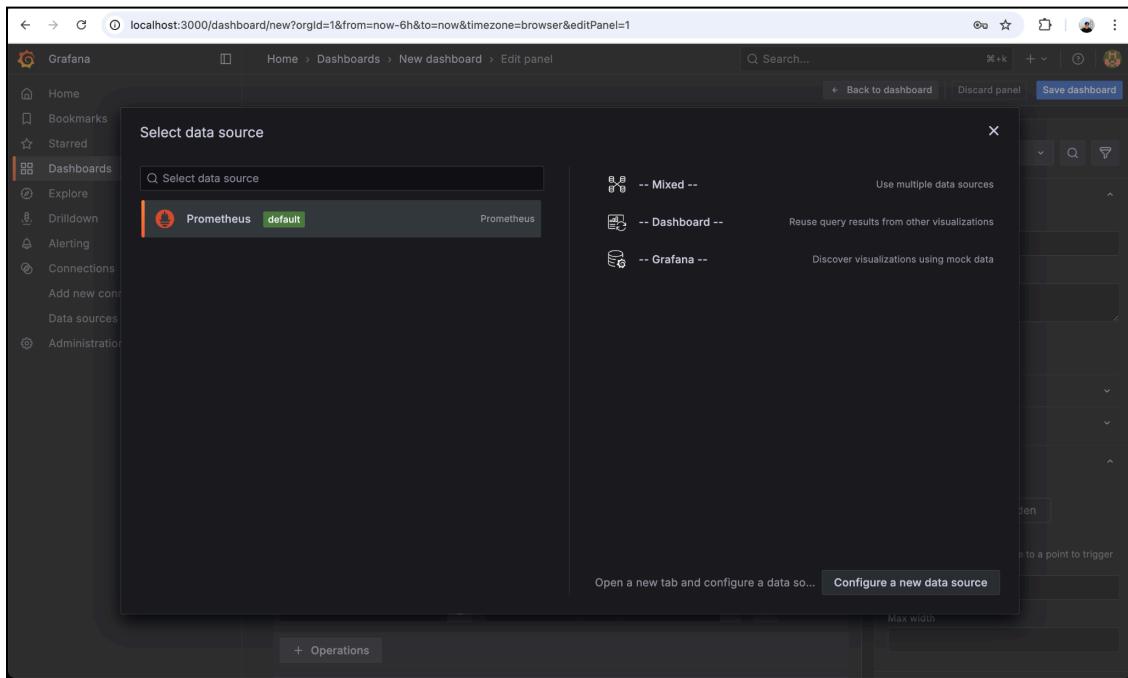
We need to update the password.



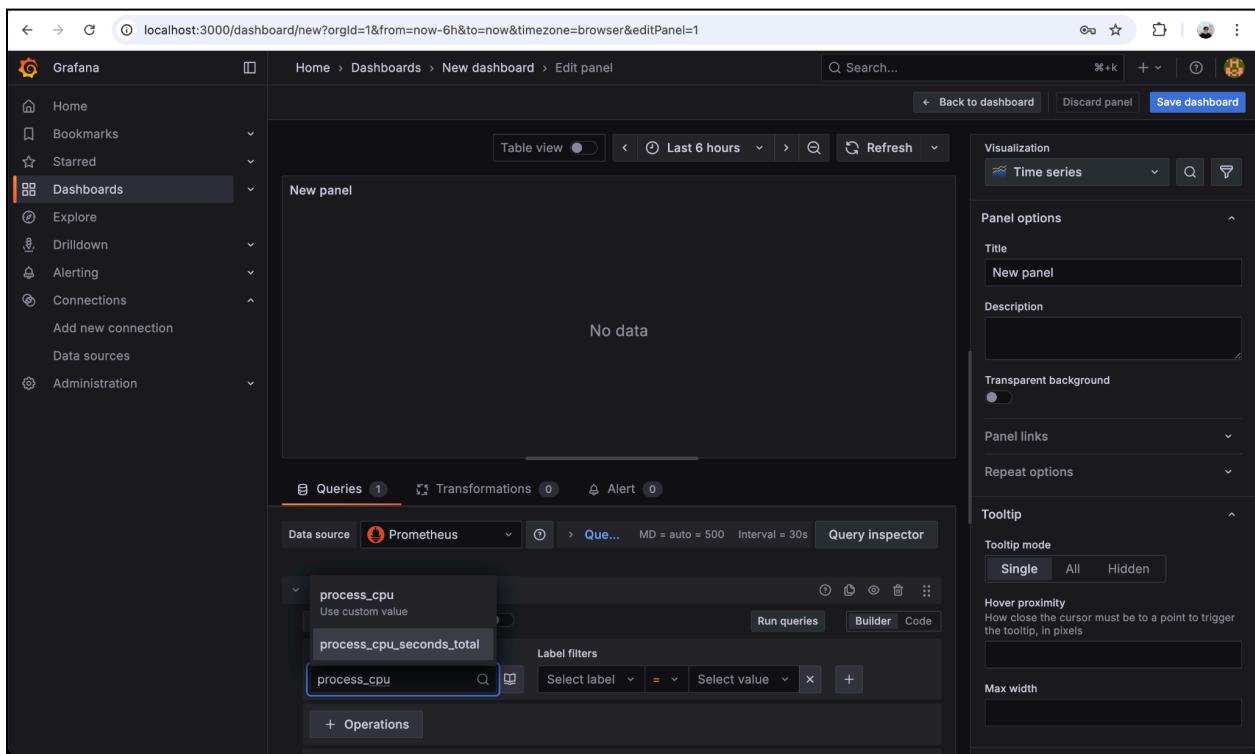
Logged in the Grafana successfully.



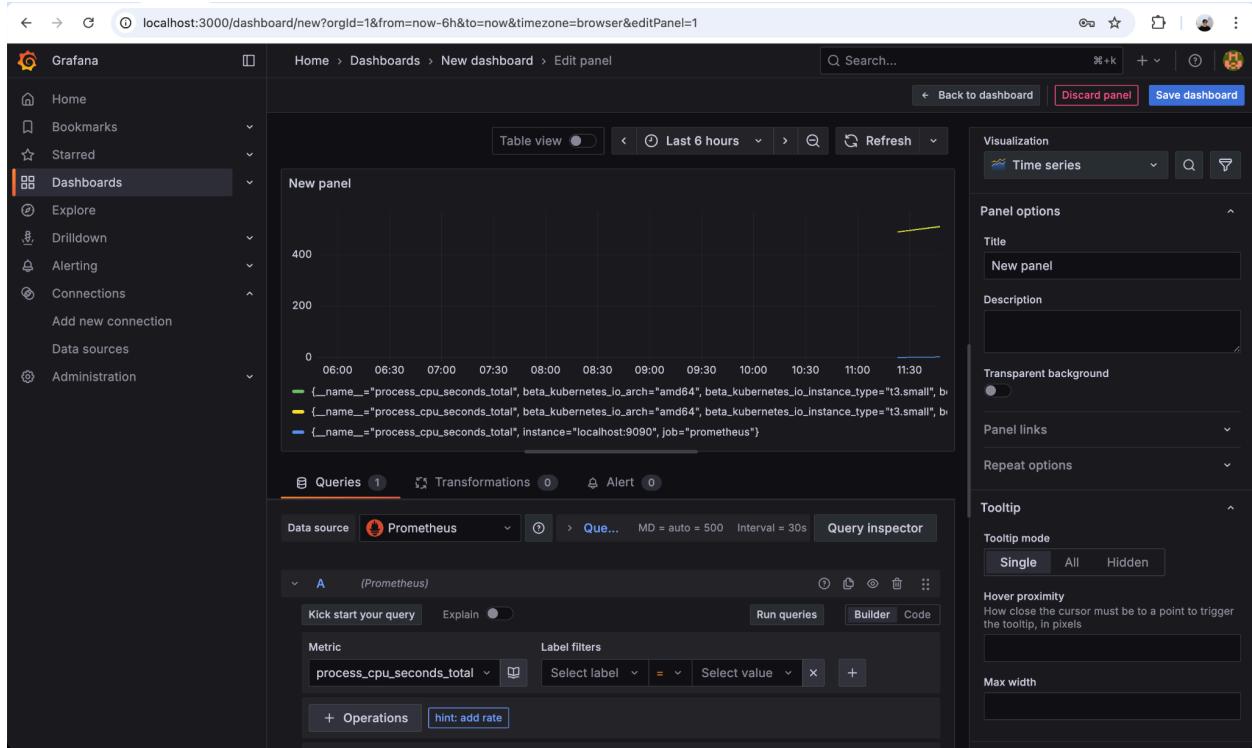
Go to **Dashboard** → **Create dashboard** → **Add visualization** → **Select Prometheus (default)**.



In metrics, search and select **process\_cpu\_seconds\_total** and run the queries.



After showing on the panel, click **Save dashboard**.



Now we can monitor with Grafana.

