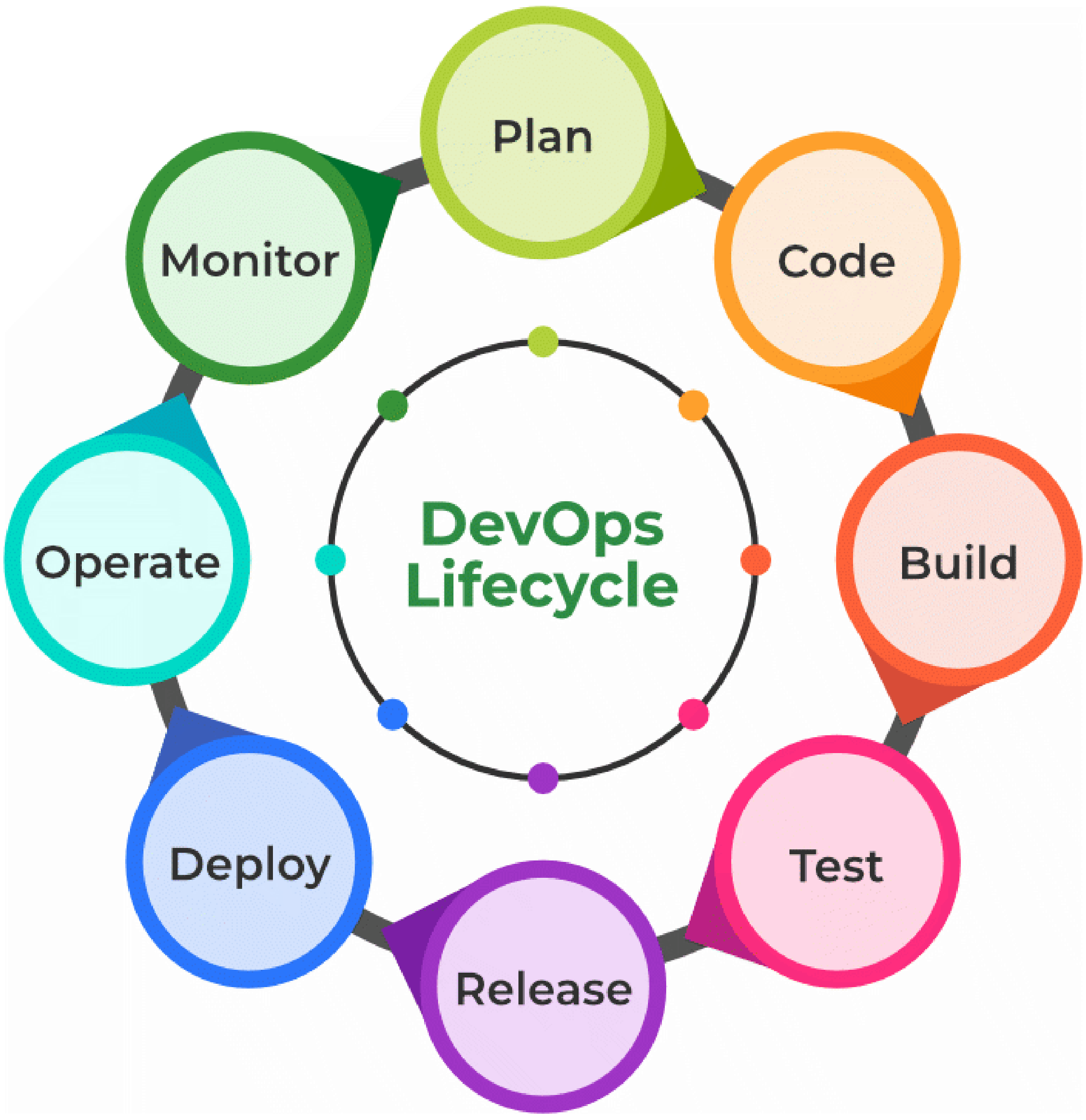
DEVOPS

DevOps is a methodology in the software development and IT industry, basically a combination of two words

Development and **Op**erations

- Promotes collaboration between Developer team and Operations team
- Deploys Code faster in an Automated and Repeatable way
- To increase an organization's speed when it comes to delivering Applications and Services.
- DevOps lifecycle is the methodology where professional development teams come together to bring products to market more efficiently and quickly

DEVOPS LIFECYCLE:



STAGES OF DEVOPS LIFECYCLE:

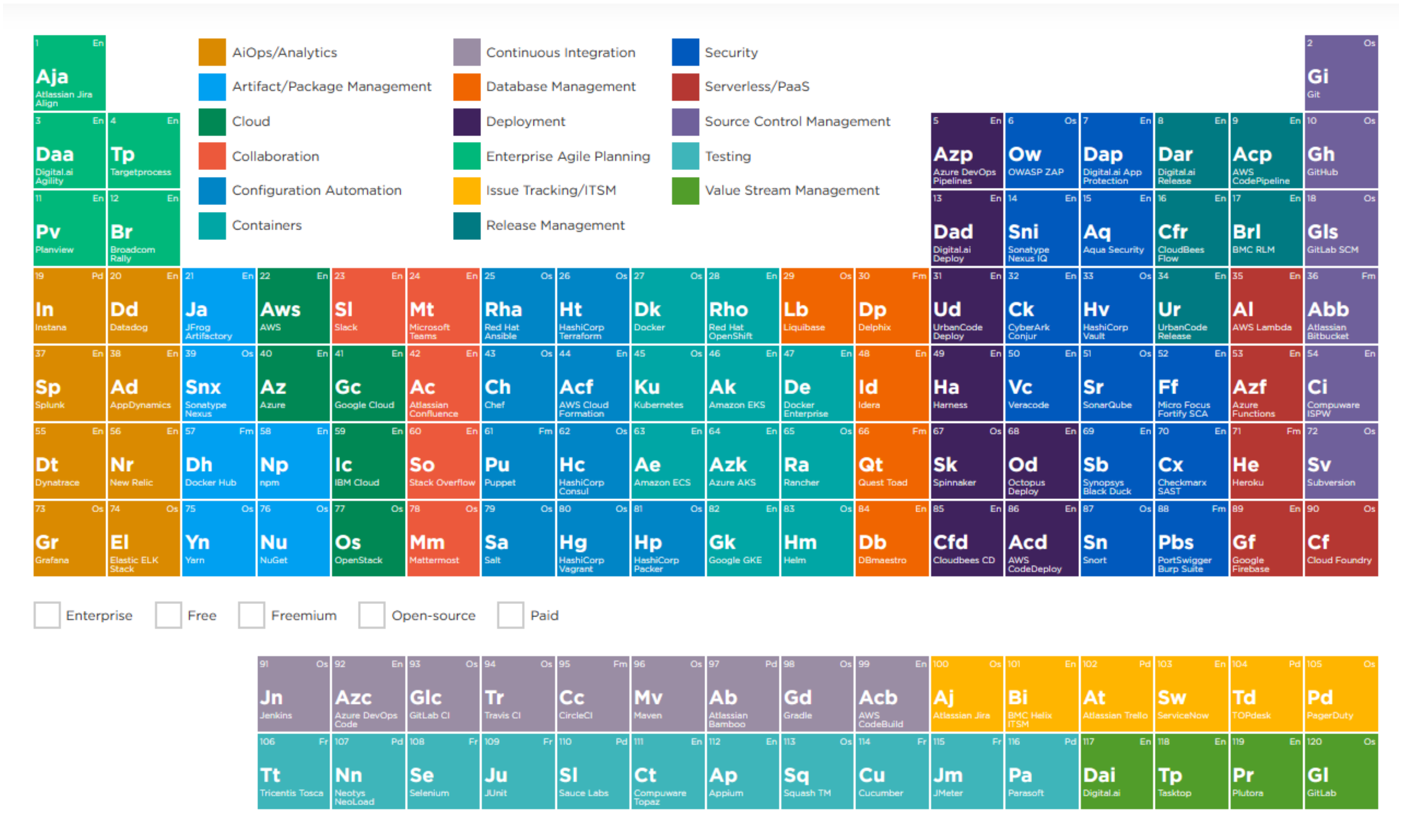
- **PLAN**: Determining the commercial needs and gathering the opinions of end-user by professionals in this level of the DevOps lifecycle.
- **CODE**: At this level, the code for the same is developed and in order to simplify the design, the team of developers uses tools and extensions that take care of security problems.
- **BUILD**: After the coding part, programmers use various tools for the submission of the code to the common code source.
- **TEST**: This level is very important to assure software integrity. Various sorts of tests are done such as user acceptability testing, safety testing, speed testing, and many more.
- **RELEASE**: At this level, everything is ready to be deployed in the operational environment.
- **DEPLOY**: In this level, Infrastructure-as-Code assists in creating the operational infrastructure and subsequently publishes the build using various DevOps lifecycle tools.
- **OPERATE**: At this level, the available version is ready for users to use. Here, the department looks after the server configuration and deployment.
- **MONITOR**: The observation is done at this level that depends on the data which is gathered from consumer behaviour, the efficiency of applications, and from various other sources.

USE OF DEVOPS:

- **FASTER DELIVERY**: DevOps enables organizations to release new products and updates faster and more frequently, which can lead to a competitive advantage.
- **IMPROVED COLLABARATION**: DevOps promotes collaboration between development and operations teams, resulting in better communication, increased efficiency, and reduced friction.
- **IMPROVED QUALITY**: DevOps emphasizes automated testing and continuous integration, which helps to catch bugs early in the development process and improve the overall quality of software.
- **INCREASED AUTOMATION**: DevOps enables organizations to automate many manual processes, freeing up time for more strategic work and reducing the risk of human error.
- **BETTER SCALABILITY**: DevOps enables organizations to quickly and efficiently scale their infrastructure to meet changing demands, improving the ability to respond to business needs.
- **INCREASES CUSTOMER SATISFACTION**: DevOps helps organizations to deliver new features and updates more quickly, which can result in increased customer satisfaction and loyalty.
- **IMPROVED SECURITY**: DevOps promotes security best practices, such as continuous testing and monitoring, which can help to reduce the risk of security breaches and improve the overall security of an organization’s systems.
- **BETTER RESOURCE UTILIZATION**: DevOps enables organizations to optimize their use of resources, including hardware, software, and personnel, which can result in cost savings and improved efficiency

DEVOPS PERIODIC TABLE:

LINK: [Periodic Table of DevOps Tools | Digital.ai]()



🟧	AiOps/Analytics
🟦	Artifact/Package Management
🟩	Cloud
🟥	Collaboration
🟦	Configuration Automation
🟦	Containers
⬜	Continuous Integration
🟧	Database Management
🟪	Deployment
🟩	Enterprise Agile Planning
🟨	Issue Tracking/ITSM
🟦	Release Management
🟦	Security
🟥	Serverless/PaaS
🟪	Source Control Management
🟦	Testing
🟩	Value Stream Management

1	En	Aja	Atlassian Jira Align
2	Os	Gi	Git
3	En	Daa	Digital.ai Agility
4	En	Tp	Targetprocess
5	En	Azp	Azure DevOps Pipelines
6	Os	Ow	OWASP ZAP
7	En	Dap	Digital.ai App Protection
8	En	Dar	Digital.ai Release
9	En	Acp	AWS CodePipeline
10	Os	Gh	GitHub
11	En	Pv	Planview
12	En	Br	Broadcom Rally
13	En	Dad	Digital.ai Deploy
14	En	Sni	Sonatype Nexus IQ
15	En	Aq	Aqua Security
16	En	Cfr	CloudBees Flow
17	En	Brl	BMC RLM
18	Os	Gls	GitLab SCM
19	Pd	In	Instana
20	En	Dd	Datadog
21	En	Ja	JFrog Artifactory
22	En	Aws	AWS
23	En	Sl	Slack
24	En	Mt	Microsoft Teams
25	Os	Rha	Red Hat Ansible
26	Os	Ht	HashiCorp Terraform
27	Os	Dk	Docker
28	En	Rho	Red Hat OpenShift
29	Os	Lb	Liquibase
30	Fm	Dp	Delphix
31	En	Ud	UrbanCode Deploy
32	En	Ck	CyberArk Conjur
33	Os	Hv	HashiCorp Vault
34	En	Ur	UrbanCode Release
35	En	Al	AWS Lambda
36	Fm	Abb	Atlassian Bitbucket
37	En	Sp	Splunk
38	En	Ad	AppDynamics
39	Os	Snx	Sonatype Nexus
40	En	Az	Azure
41	En	Gc	Google Cloud
42	En	Ac	Atlassian Confluence
43	Os	Ch	Chef
44	En	Acf	AWS Cloud Formation
45	Os	Ku	Kubernetes
46	En	Ak	Amazon EKS
47	En	De	Docker Enterprise
48	En	Id	Idera
49	En	Ha	Harness
50	En	Vc	Veracode
51	Os	Sr	SonarQube
52	En	Ff	Micro Focus Fortify SCA
53	En	Azf	Azure Functions
54	En	Ci	Compuware ISPW
55	En	Dt	Dynatrace
56	En	Nr	New Relic
57	Fm	Dh	Docker Hub
58	En	Np	npm
59	En	Ic	IBM Cloud
60	En	So	Stack Overflow
61	Fm	Pu	Puppet
62	Os	Hc	HashiCorp Consul
63	En	Ae	Amazon ECS
64	En	Azk	Azure AKS
65	Os	Ra	Rancher
66	Fm	Qt	Quest Toad
67	Os	Sk	Spinnaker
68	En	Od	Octopus Deploy
69	En	Sb	Synopsys Black Duck
70	En	Cx	Checkmarx SAST
71	Fm	He	Heroku
72	Os	Sv	Subversion
73	Os	Gr	Grafana
74	Os	El	Elastic ELK Stack
75	Os	Yn	Yarn
76	Os	Nu	NuGet
77	Os	Os	OpenStack
78	Os	Mm	Mattermost
79	Os	Sa	Salt
80	Os	Hg	HashiCorp Vagrant
81	Os	Hp	HashiCorp Packer
82	En	Gk	Google GKE
83	Os	Hm	Helm
84	En	Db	DBmaestro
85	En	Cfd	Cloudbees CD
86	En	Acd	AWS CodeDeploy
87	Os	Sn	Snort
88	Fm	Pbs	PortSwigger Burp Suite
89	En	Gf	Google Firebase
90	Os	Cf	Cloud Foundry
91	Os	Jn	Jenkins
92	En	Azc	Azure DevOps Code
93	Os	Glc	GitLab CI
94	Os	Tr	Travis CI
95	Fm	Cc	CircleCI
96	Os	Mv	Maven
97	Pd	Ab	Atlassian Bamboo
98	Os	Gd	Gradle
99	En	Acb	AWS CodeBuild
100	Os	Aj	Atlassian Jira
101	En	Bi	BMC Helix ITSM
102	Pd	At	Atlassian Trello
103	En	Sw	ServiceNow
104	Pd	Td	TOPdesk
105	Os	Pd	PagerDuty
106	Fr	Tt	Tricentis Tosca
107	Pd	Nn	Neotys NeoLoad
108	Fr	Se	Selenium
109	Fr	Ju	JUnit
110	Pd	Sl	Sauce Labs
111	En	Ct	Compuware Topaz
112	En	Ap	Appium
113	Os	Sq	Squash TM
114	Fr	Cu	Cucumber
115	Fr	Jm	JMeter
116	Pd	Pa	Parasoft
117	En	Dai	Digital.ai
118	En	Tp	Tasktop
119	En	Pr	Plutora
120	Os	Gl	GitLab

☐ Enterprise ☐ Free ☐ Freemium ☐ Open-source ☐ Paid

DEVOPS TOOLS:

git

VERISON CONTROL SYSTEM:

- A Version Control System (VCS) is a system that maintains different versions of your project when we work in a team or as an individual

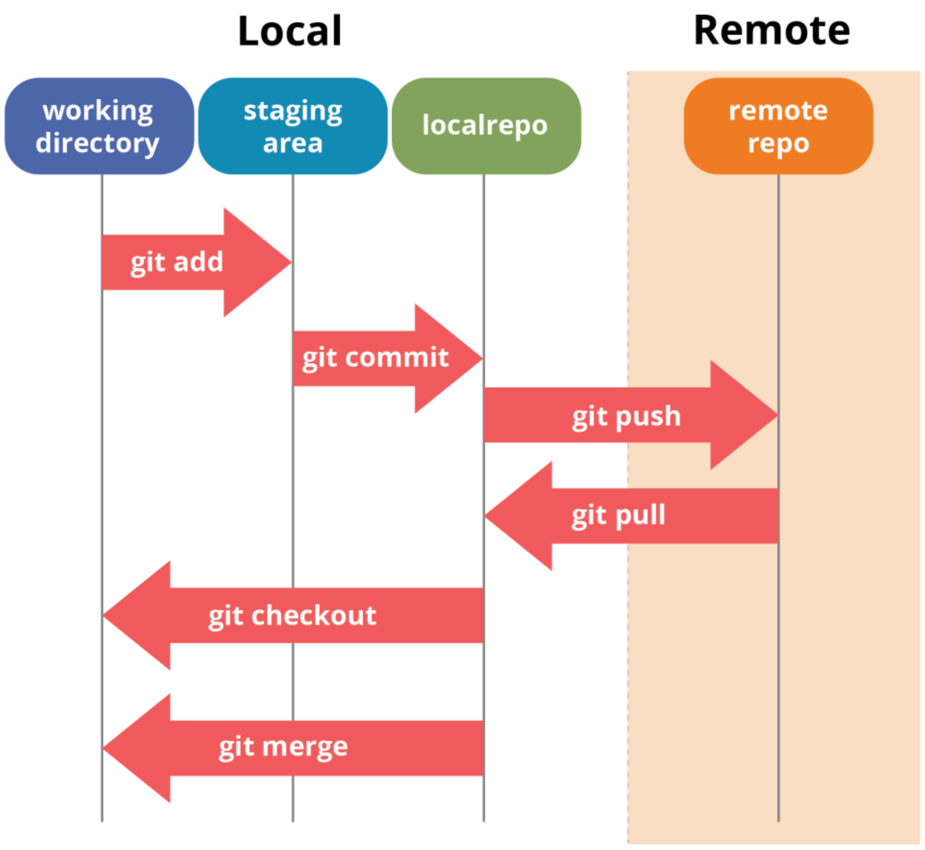
GIT – DISTRIBUTED VERSION CONTROL SYSTEM

- Distributed Version Control System
- Provide a Reliable and Efficient way to manage Projects and Track changes in Source Code over time

- Keep track of changes to source code and allows developers to collaborate on the same project
- Provides a secure way to store and share code

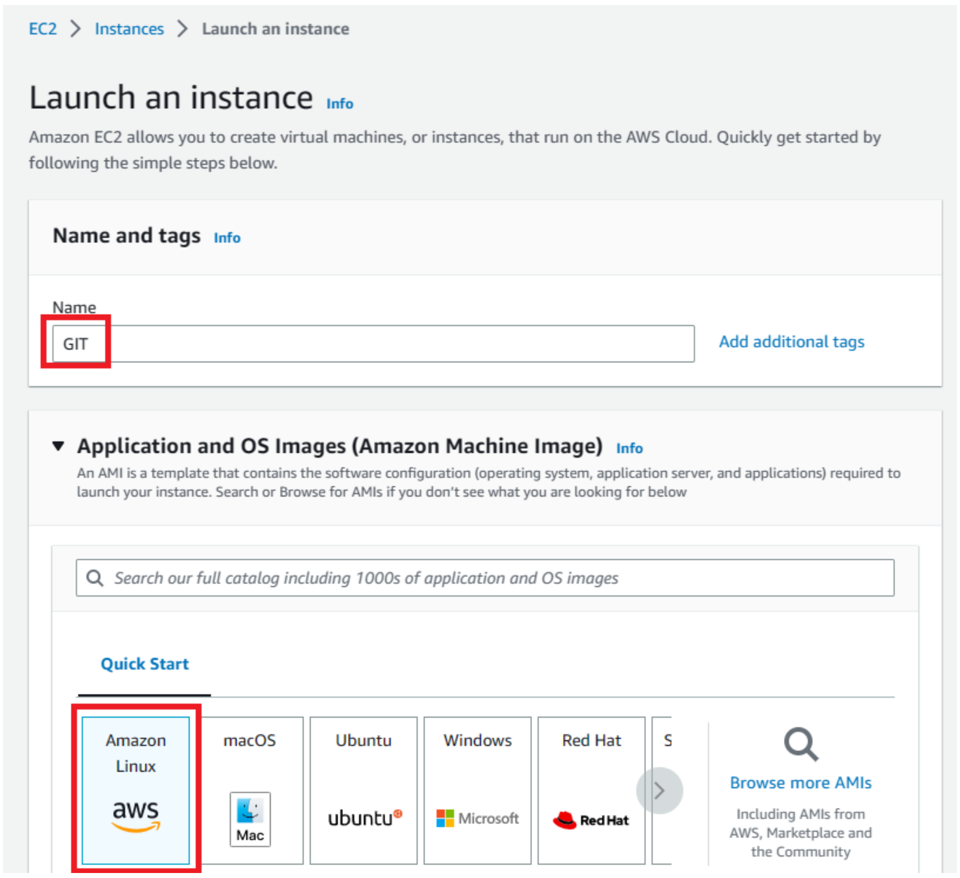
GIT REPOSITORY STRUCTURE:

- **WORKING DIRECTORY**: This is your local directory where you make the project (write code) and make changes to it.
- **STAGING AREA (OR INDEX)**: this is an area where you first need to put your project before committing. This is used for code review by other team members.
- **LOCAL REPOSITORY**: this is your local repository where you commit changes to the project before pushing them to the central repository on Github. This is what is provided by the distributed version control system. This corresponds to the .git folder in our directory.
- **CENTRAL REPOSITORY**: This is the main project on the central server, a copy of which is with every team member as a local repository.

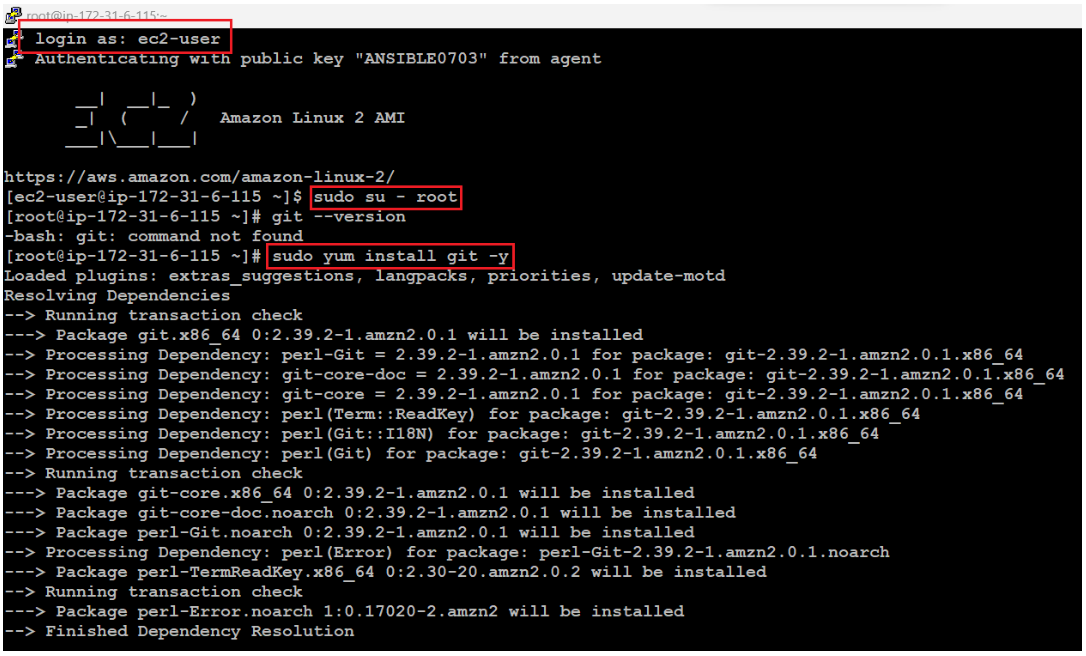


HOW TO INSTALL GIT?

STEP 1: Create a EC2 Instance (Name as GIT) with Ports # enabled as per your requirement (Default – SSH, HTTP & HTTPS)

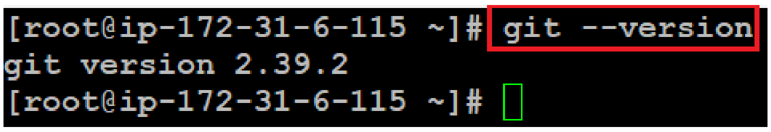

EC2 > Instances > Launch an instance
Launch an instance Info
Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.
Name and tags Info
Name
GIT
Add additional tags
Application and OS Images (Amazon Machine Image) Info
An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below
Search our full catalog including 1000s of application and OS images
Quick Start
Amazon Linux
macOS
Ubuntu
Windows
Red Hat
Browse more AMIs
Including AMIs from AWS, Marketplace and the Community

STEP 2: Login as **EC2-USER** (If Amazon Linux Machine) and switch to ROOT User using below commands:



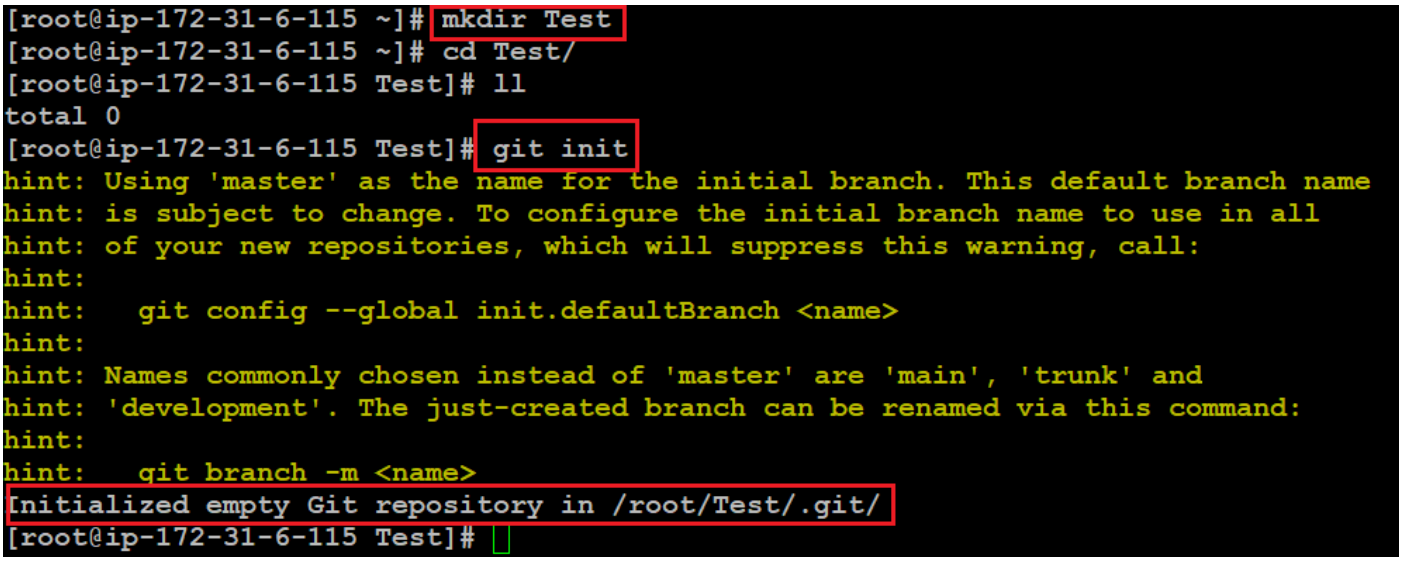
```
login as: ec2-user
Authenticating with public key "ANSIBLE0703" from agent

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-6-115 ~]$ sudo su - root
[root@ip-172-31-6-115 ~]# git --version
-bash: git: command not found
[root@ip-172-31-6-115 ~]# sudo yum install git -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Resolving Dependencies
--> Running transaction check
---> Package git.x86_64 0:2.39.2-1.amzn2.0.1 will be installed
--> Processing Dependency: perl-Git = 2.39.2-1.amzn2.0.1 for package: git-2.39.2-1.amzn2.0.1.x86_64
--> Processing Dependency: git-core-doc = 2.39.2-1.amzn2.0.1 for package: git-2.39.2-1.amzn2.0.1.x86_64
--> Processing Dependency: git-core = 2.39.2-1.amzn2.0.1 for package: git-2.39.2-1.amzn2.0.1.x86_64
--> Processing Dependency: perl(Term::ReadKey) for package: git-2.39.2-1.amzn2.0.1.x86_64
--> Processing Dependency: perl(Git::I18N) for package: git-2.39.2-1.amzn2.0.1.x86_64
--> Processing Dependency: perl(Git) for package: git-2.39.2-1.amzn2.0.1.x86_64
--> Running transaction check
---> Package git-core.x86_64 0:2.39.2-1.amzn2.0.1 will be installed
---> Package git-core-doc.noarch 0:2.39.2-1.amzn2.0.1 will be installed
---> Package perl-Git.noarch 0:2.39.2-1.amzn2.0.1 will be installed
--> Processing Dependency: perl(Error) for package: perl-Git-2.39.2-1.amzn2.0.1.noarch
---> Package perl-TermReadKey.x86_64 0:2.30-20.amzn2.0.2 will be installed
--> Running transaction check
---> Package perl-Error.noarch 1:0.17020-2.amzn2 will be installed
--> Finished Dependency Resolution
```

STEP 3: Check GIT Version



```
[root@ip-172-31-6-115 ~]# git --version
git version 2.39.2
[root@ip-172-31-6-115 ~]#
```

STEP 4: Create a Directory as "**TEST**" and navigate into the directory. Then initialize GIT using "**git init**" command as below:

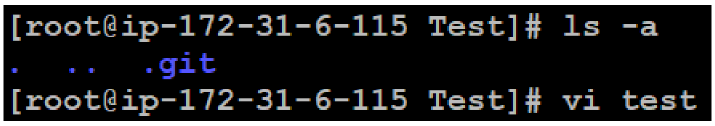


```
[root@ip-172-31-6-115 ~]# mkdir Test
[root@ip-172-31-6-115 ~]# cd Test/
[root@ip-172-31-6-115 Test]# ll
total 0
[root@ip-172-31-6-115 Test]# git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /root/Test/.git/
[root@ip-172-31-6-115 Test]#
```

STEP 5: Create a file using VI Editor and add your content in it.

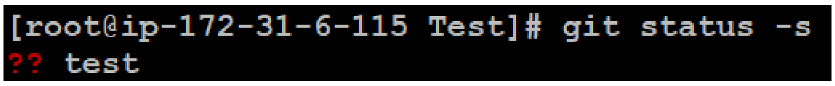
[ls-a gives you the list of files in your current directory including hidden files]

Example: First line



```
[root@ip-172-31-6-115 Test]# ls -a
.  ..  .git
[root@ip-172-31-6-115 Test]# vi test
```

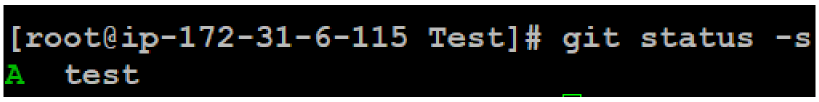
STEP 6: To check the status of the current file whether it is Tracked or untracked, enter below command:



```
[root@ip-172-31-6-115 Test]# git status -s
?? test
```

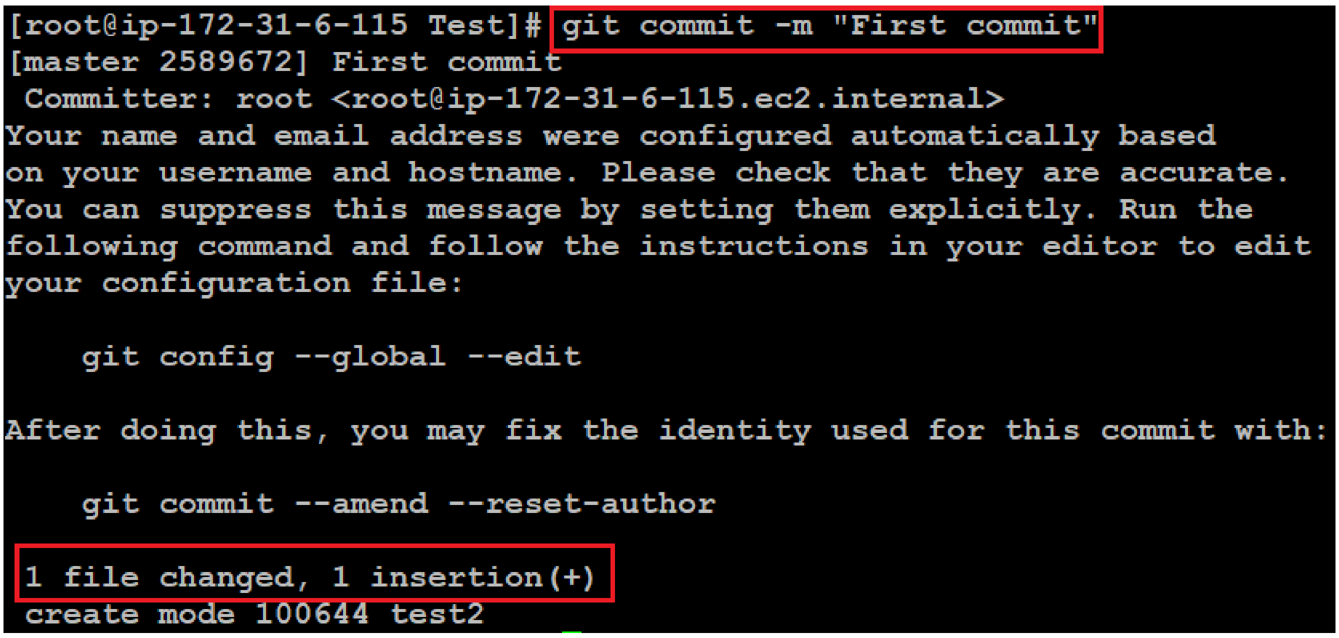
?? – means untracked files

STEP 7: To add the untracked files to the Staging Area, enter below command:



```
[root@ip-172-31-6-115 Test]# git status -s
A  test
```

STEP 8: To add the **ADDED** file to **LOCAL REPOSITORY**

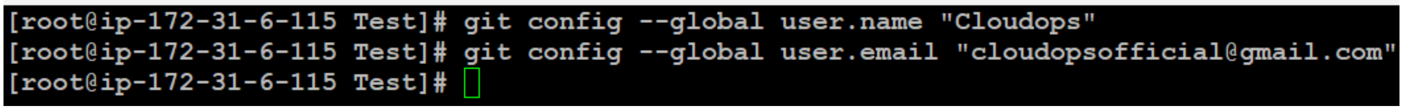


```
[root@ip-172-31-6-115 Test]# git commit -m "First commit"
[master 2589672] First commit
 Committer: root <root@ip-172-31-6-115.ec2.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 1 insertion(+)
 create mode 100644 test2
```

STEP 9: To configure your Name and Email Address, enter below commands:

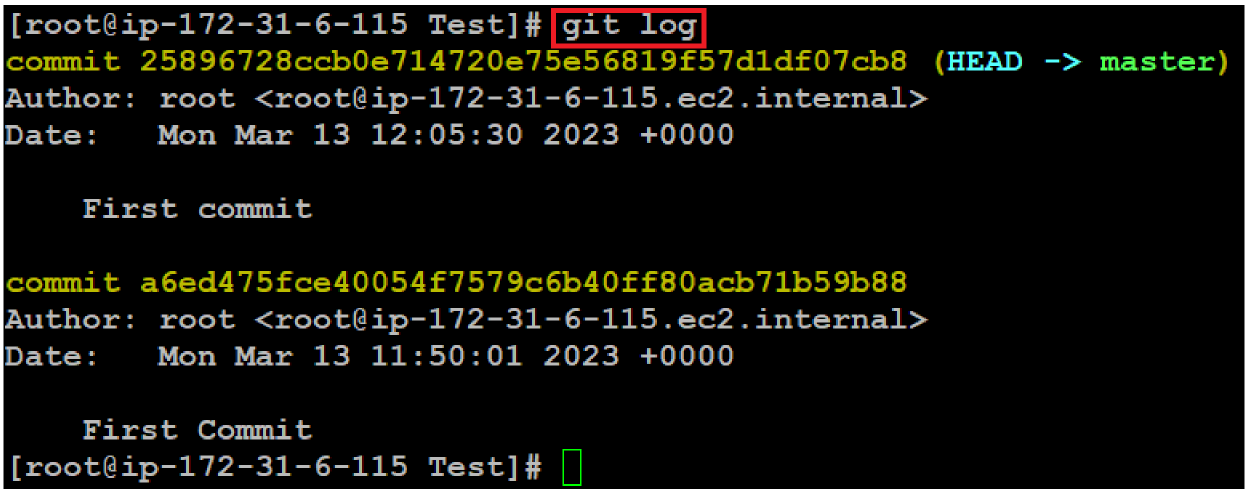
git config --global user.name "************"

git config --global user.email "************"



```
[root@ip-172-31-6-115 Test]# git config --global user.name "Cloudops"
[root@ip-172-31-6-115 Test]# git config --global user.email "cloudopsofficial@gmail.com"
[root@ip-172-31-6-115 Test]#
```

STEP 10: To check the activities in **GIT**

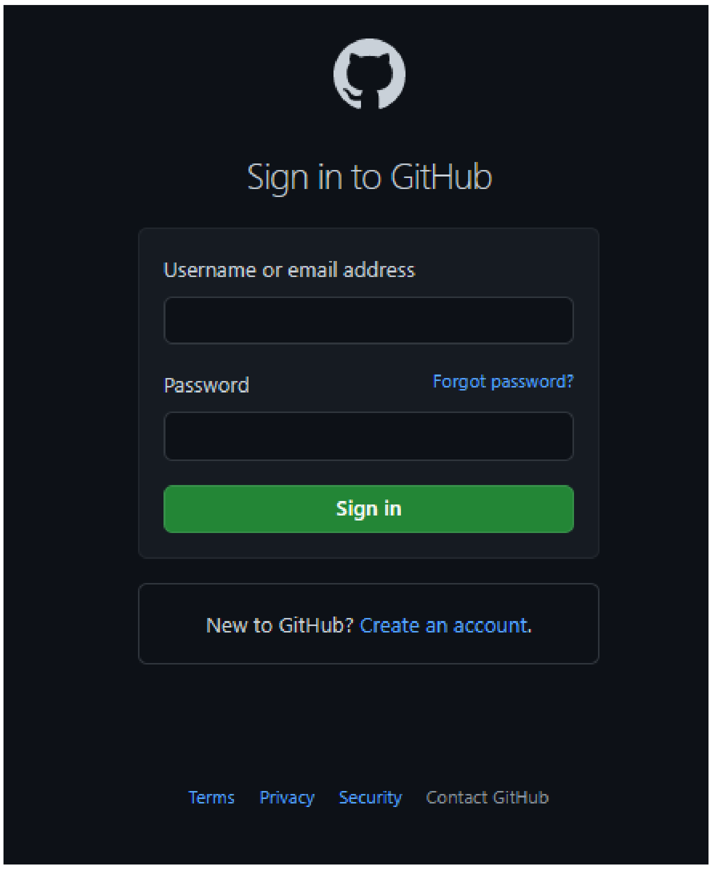


```
[root@ip-172-31-6-115 Test]# git log
commit 25896728ccb0e714720e75e56819f57d1df07cb8 (HEAD -> master)
Author: root <root@ip-172-31-6-115.ec2.internal>
Date:   Mon Mar 13 12:05:30 2023 +0000

    First commit

commit a6ed475fce40054f7579c6b40ff80acb71b59b88
Author: root <root@ip-172-31-6-115.ec2.internal>
Date:   Mon Mar 13 11:50:01 2023 +0000

    First Commit
[root@ip-172-31-6-115 Test]#
```

GIT HUB ACCOUNT CREATION:

STEP 1: Login https://github.com/ for GIT HUB ACCOUNT CREATION

STEP 2: Create a Account using Mail ID & Password. Once created, login as below:

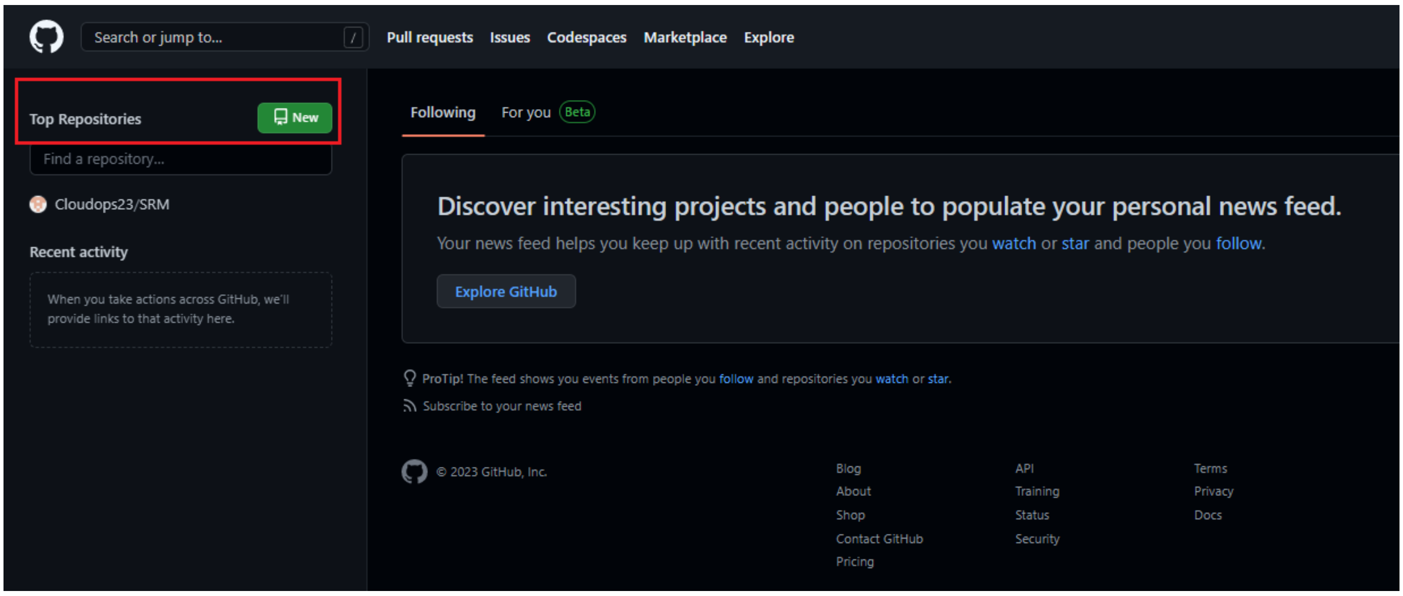

Sign in to GitHub

Username or email address

Password

New to GitHub? Create an account.

STEP 3: To create a REMOTE Repository, Click NEW



When you take actions across GitHub, we'll provide links to that activity here.

Discover interesting projects and people to populate your personal news feed.

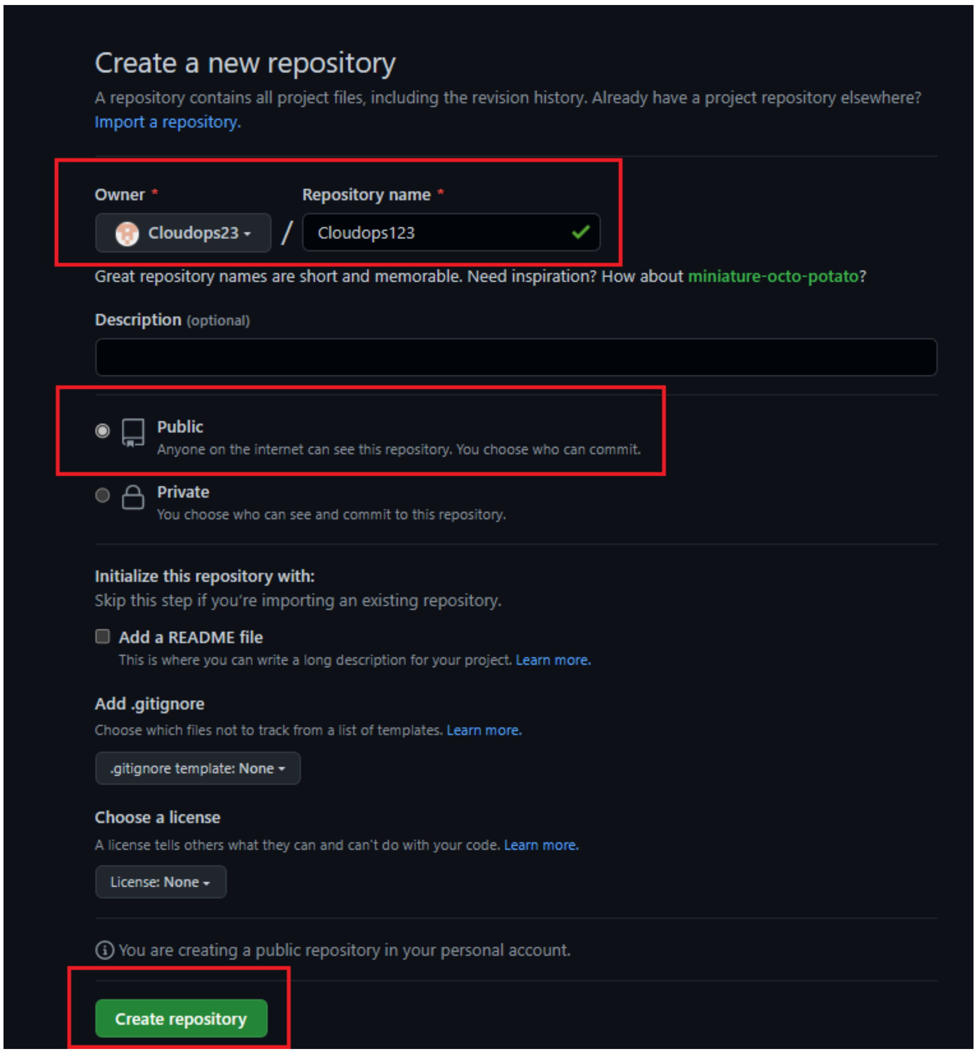
Your news feed helps you keep up with recent activity on repositories you watch or star and people you follow.

ProTip! The feed shows you events from people you follow and repositories you watch or star.

Subscribe to your news feed

© 2023 GitHub, Inc.

STEP 4: Select as highlighted for creating a new repository



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner * Cloudops23 / Repository name * Cloudops123

Great repository names are short and memorable. Need inspiration? How about miniature-octo-potato?

Description (optional)

[✓ Public] Anyone on the internet can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

Add a README file
This is where you can write a long description for your project. Learn more.

Add .gitignore
Choose which files not to track from a list of templates. Learn more.

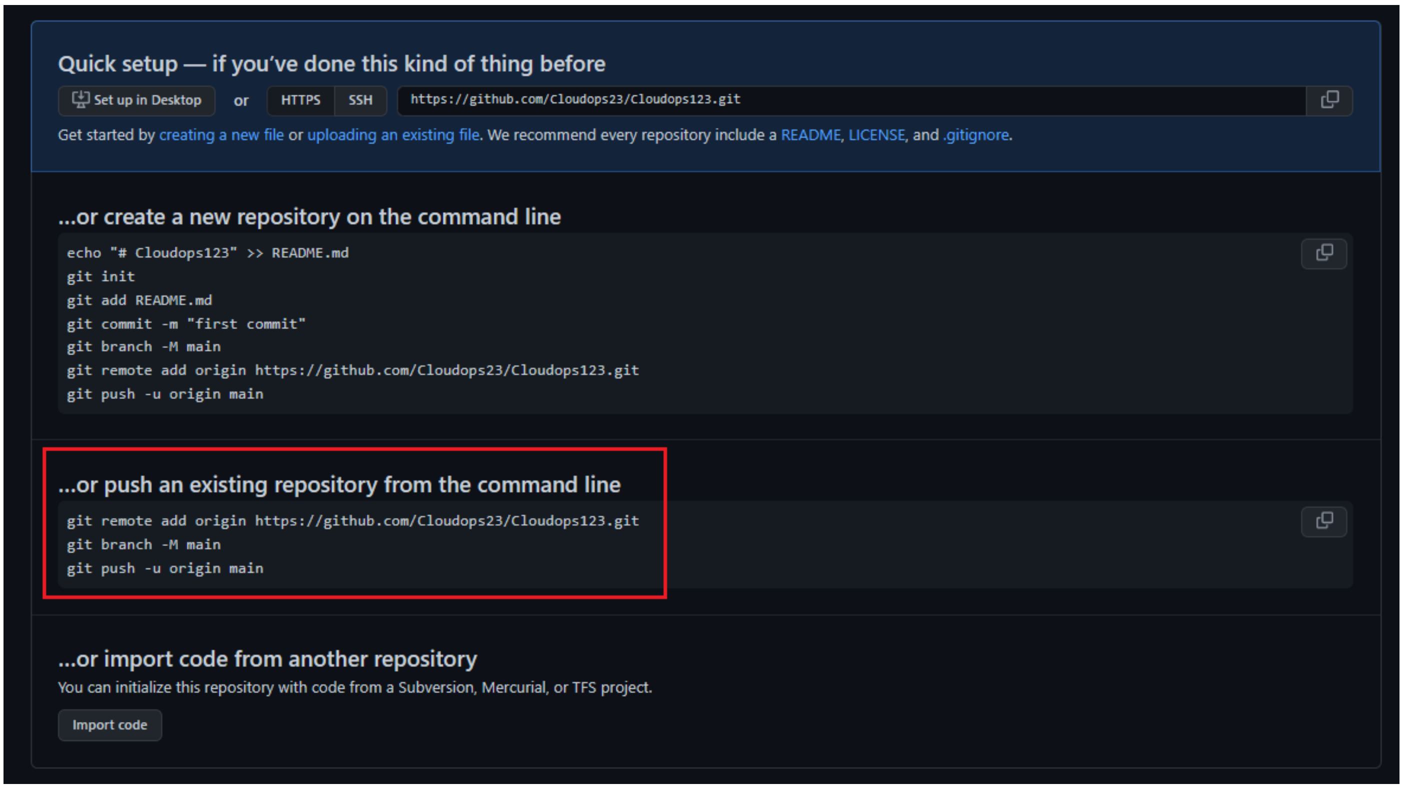
.gitignore template: None

Choose a license
A license tells others what they can and can't do with your code. Learn more.

License: None

You are creating a public repository in your personal account.

STEP 5: Enter highlighted commands in the respective server for connecting to **GITHUB** Repository



Quick setup — if you've done this kind of thing before

```
https://github.com/Cloudops23/Cloudops123.git
```

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

…or create a new repository on the command line

```
echo "# Cloudops123" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/Cloudops23/Cloudops123.git
git push -u origin main
```

…or push an existing repository from the command line

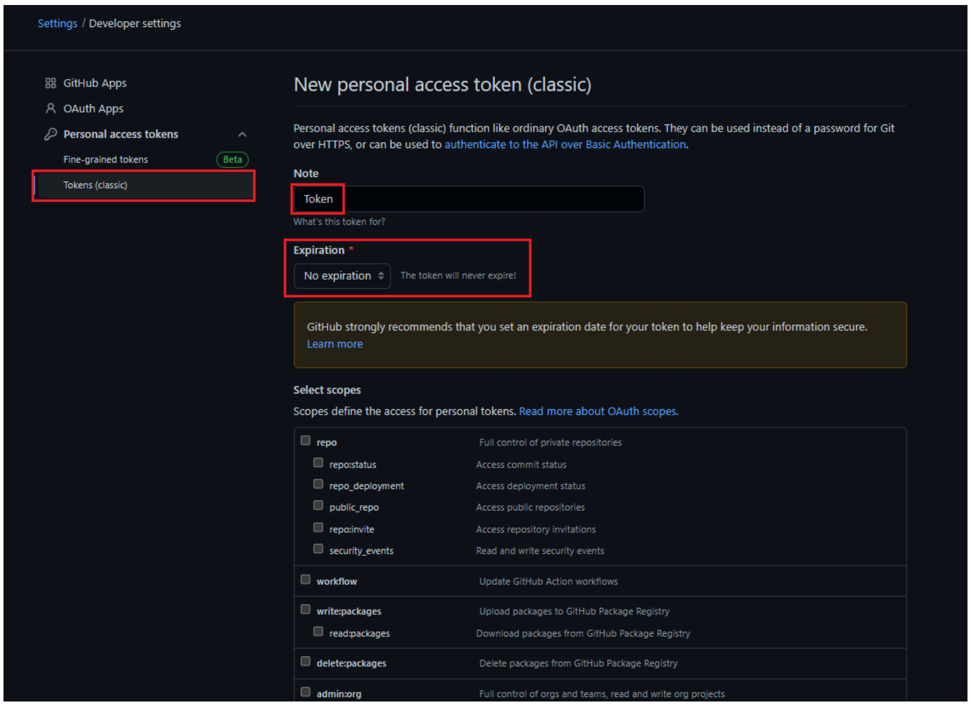
```
git remote add origin https://github.com/Cloudops23/Cloudops123.git
git branch -M main
git push -u origin main
```

…or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

STEP 6: Create **TOKENS** for Credentials

Select SETTINGS → DEVELOPER TOOLS →PERSONAL ACCESS TOKENS → GENERATE NEW TOKEN → 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: Token

What's this token for?

Expiration *: No expiration — The token will never expire!

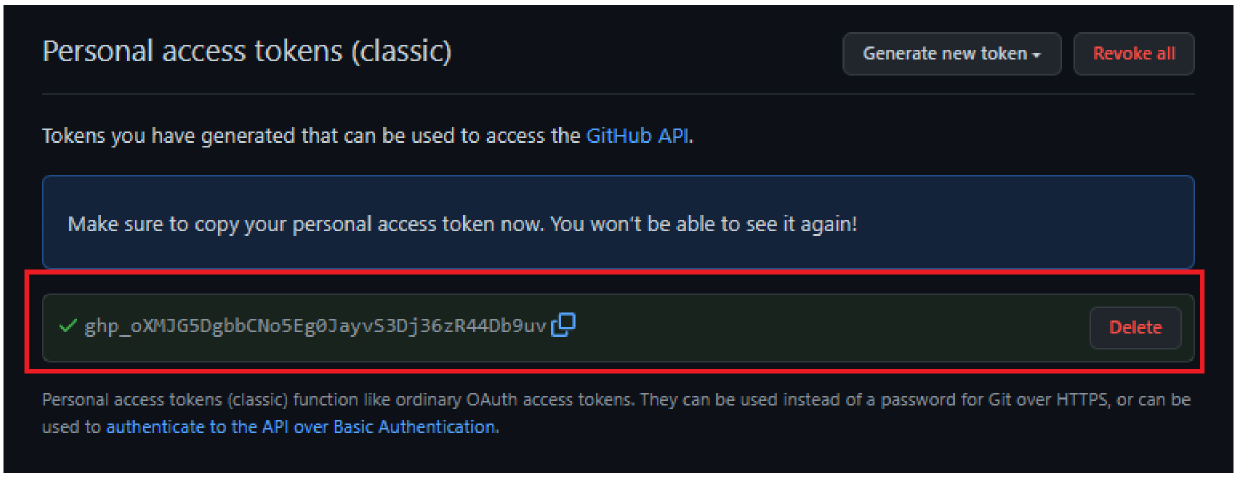
GitHub strongly recommends that you set an expiration date for your token to help keep your information secure. Learn more

Select scopes

Scopes define the access for personal tokens. Read more about OAuth scopes.

☐ repo	Full control of private repositories
☐ repo:status	Access commit status
☐ repo_deployment	Access deployment status
☐ public_repo	Access public repositories
☐ repo:invite	Access repository invitations
☐ security_events	Read and write security events
☐ workflow	Update GitHub Action workflows
☐ write:packages	Upload packages to GitHub Package Registry
☐ read:packages	Download packages from GitHub Package Registry
☐ delete:packages	Delete packages from GitHub Package Registry
☐ admin:org	Full control of orgs and teams, read and write org projects

STEP 7: Select all available options for better use. Save the generated TOKEN for future usage as below:



Personal access tokens (classic)

Tokens you have generated that can be used to access the GitHub API.

Make sure to copy your personal access token now. You won't be able to see it again!

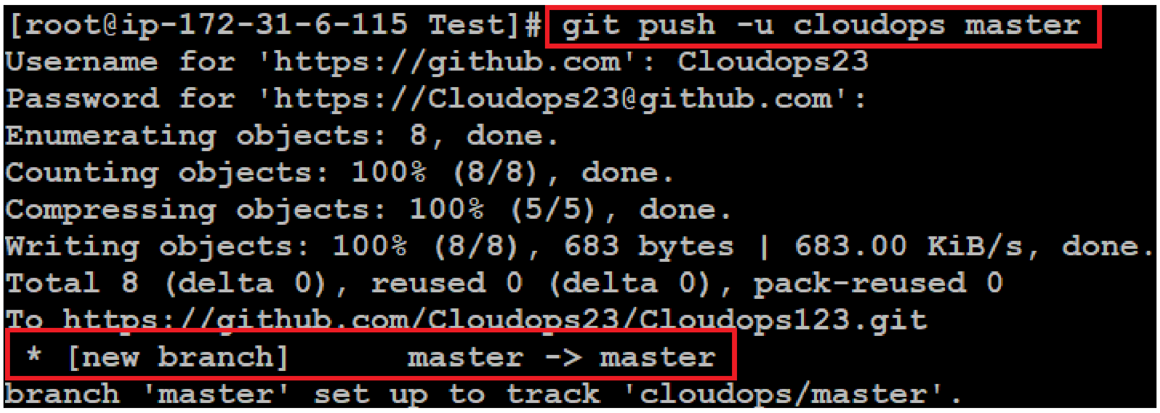
✓ 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

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.

HOW TO PUSH CODE FROM LOCAL TO REMOTE REPOSITORY

Make sure your are in the correct path before entering the PUSH Command

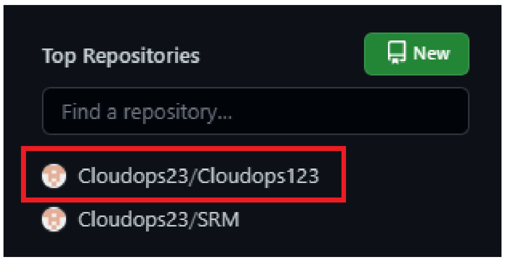
COMMAND: git push -u <Remote_Repo_name> <Branch>



```
[root@ip-172-31-6-115 Test]# git push -u cloudops master
Username for 'https://github.com': Cloudops23
Password for 'https://Cloudops23@github.com':
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Compressing objects: 100% (5/5), done.
Writing objects: 100% (8/8), 683 bytes | 683.00 KiB/s, done.
Total 8 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Cloudops23/Cloudops123.git
 * [new branch]      master -> master
branch 'master' set up to track 'cloudops/master'.
```

CHECK IF THE CODE IS PUSHED TO REMOTE REPOSITORY

Navigate to your respective Repository

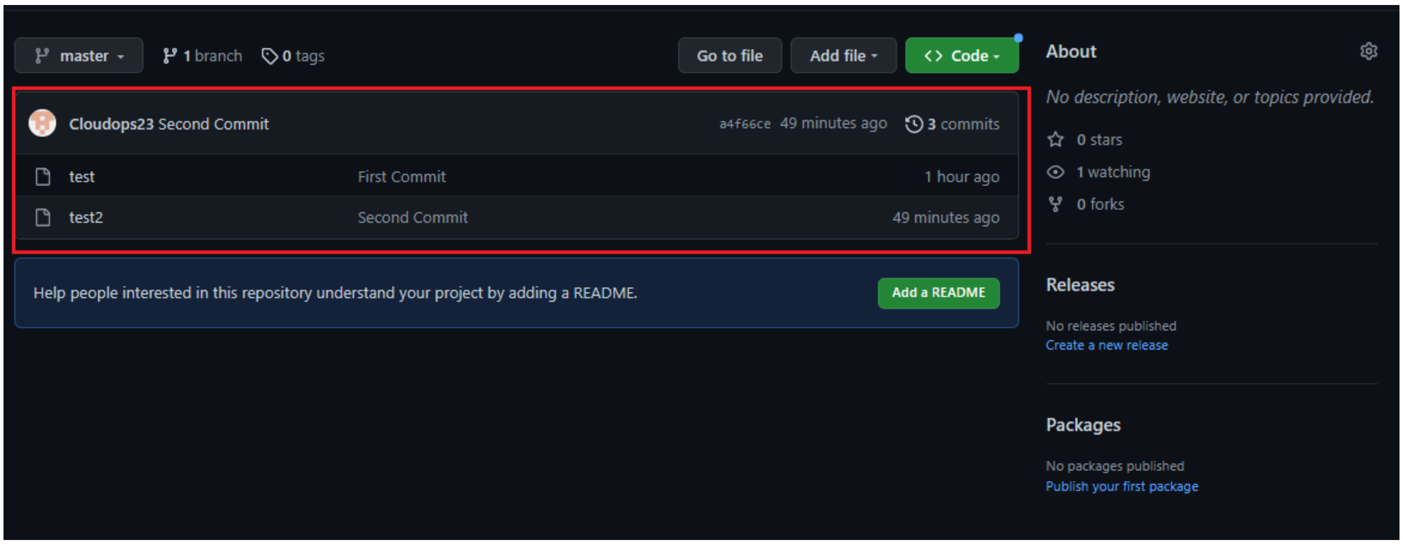


Top Repositories

Cloudops23/Cloudops123
Cloudops23/SRM

Below information can be seen in the screenshot

- Number of Commits executed
- Number of Codes available in the Directory
- Under which branch, Codes are in place



Cloudops23 Second Commit a4f66ce 49 minutes ago 3 commits

test	First Commit	1 hour ago
test2	Second Commit	49 minutes ago

Help people interested in this repository understand your project by adding a README.

No description, website, or topics provided.

No releases published

No packages published