PRIME TOOLS INSTALLATION

GITHUB KB - EVERY WEEK THE REFERENCE DOCUMENTS WILL BE HERE

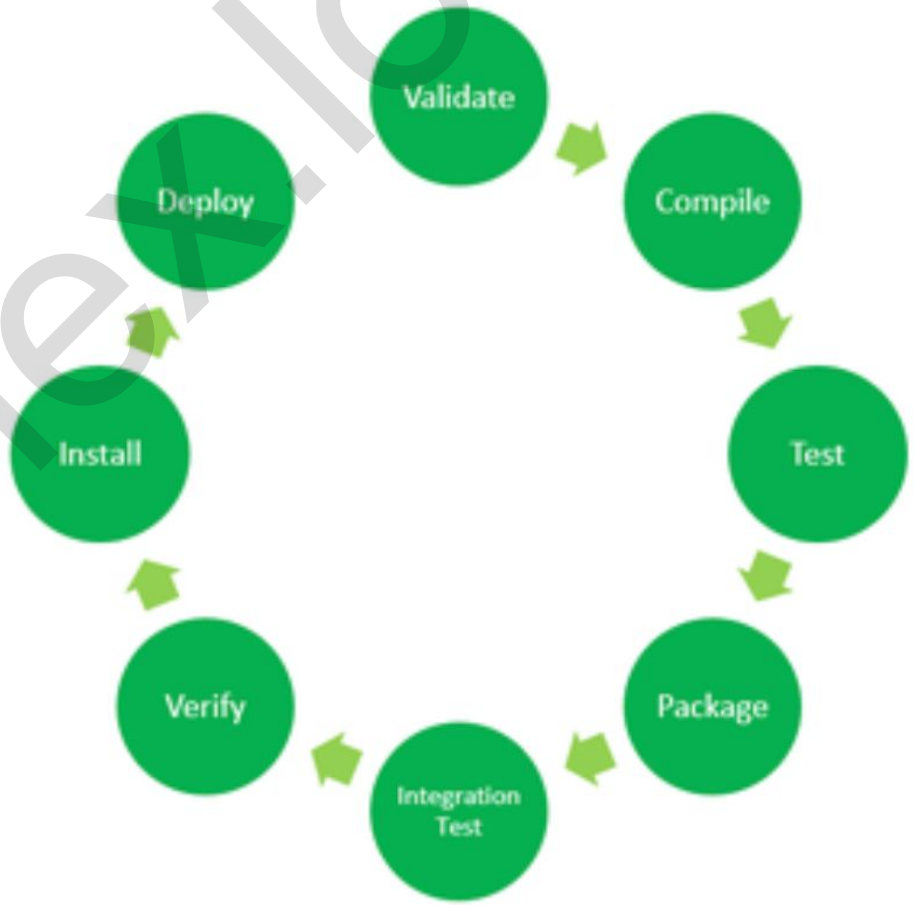
- https://github.com/praveen1994dec/Knowledge_Base.git

This github repo is for reference documents only

Github and all tools installation setup will be done shortly :)

Maven

- **Maven Lifecycle:** Below is a representation of the default Maven lifecycle and its 8 steps: Validate, Compile, Test, Package, Integration test, Verify, Install, and Deploy.



8 Phases of the Default Maven Lifecycle

1. **Validate:** This step validates if the project structure is correct. For example – It checks if all the dependencies have been downloaded and are available in the local repository.
2. **Compile:** It compiles the source code, converts the .java files to .class, and stores the classes in the target/classes folder.
3. **Test:** It runs unit tests for the project.
4. **Package:** This step packages the compiled code in a distributable format like JAR or WAR.
5. **Integration test:** It runs the integration tests for the project.
6. **Verify:** This step runs checks to verify that the project is valid and meets the quality standards.
7. **Install:** This step installs the packaged code to the local Maven repository.
8. **Deploy:** It copies the packaged code to the remote repository for sharing it with other developers.

- **mvn clean:** Cleans the project and removes all files generated by the previous build.
- **mvn compile:** Compiles source code of the project.
- **mvn test-compile:** Compiles the test source code.
- **mvn test:** Runs tests for the project.
- **mvn package:** Creates JAR or WAR file for the project to convert it into a distributable format.
- **mvn install:** Deploys the packaged JAR/ WAR file to the local repository.
- **mvn site:** generate the project documentation.
- **mvn validate:** validate the project's POM and configuration.
- **mvn idea:idea:** generate project files for IntelliJ IDEA or Eclipse.
- **mvn release:perform:** Performs a release build.
- **mvn deploy:** Copies the packaged JAR/ WAR file to the remote repository after compiling, running tests and building the project.

Curl vs Wget

- Curl is designed to be a more versatile tool and can handle a variety of data formats, including JSON, XML, and CSV. It is also able to upload data and interact with APIs.

- Wget, on other hand, is designed to be a simple, reliable tool for dowloading files.

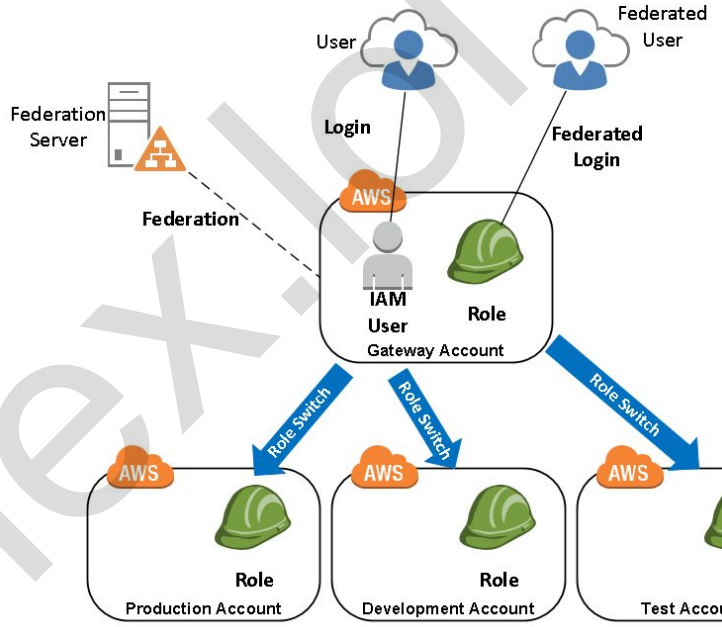
CURL	WGET
Supports a wide range of protocols, including HTTP, HTTPS, FTP, FTPS, SCP, SFTP, and more.	Supports HTTP and FTP protocols.
Can handle a variety of data formats, including JSON, XML, and CSV.	Can download recursively to download all linked files.
Supports authentication and cookies.	Can handle slow or unstable connections with ease.
Can interact with APIs.	Can resume interrupted downloads.

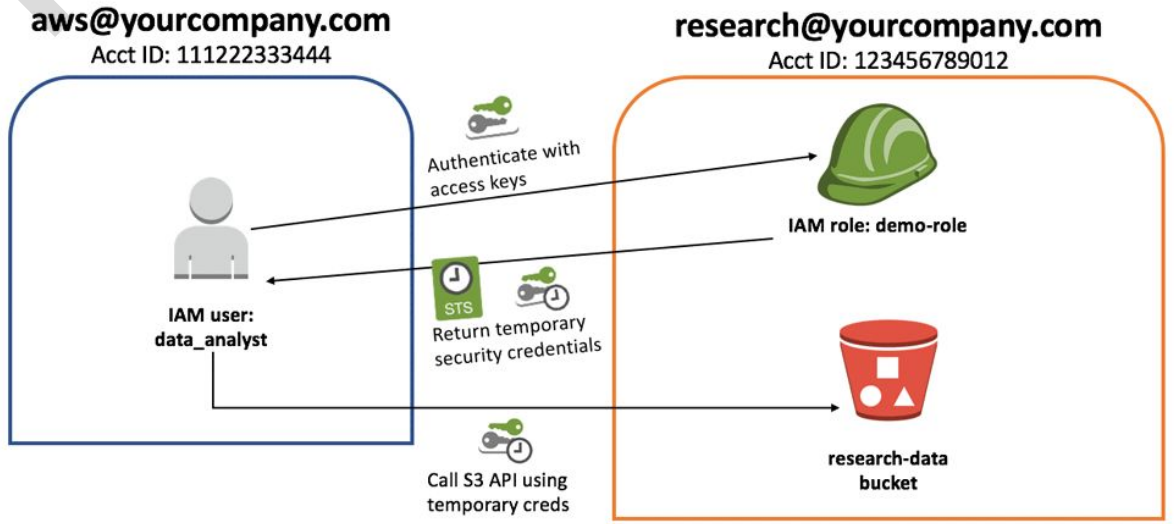
IAM Roles

- AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources. You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resources.

Features

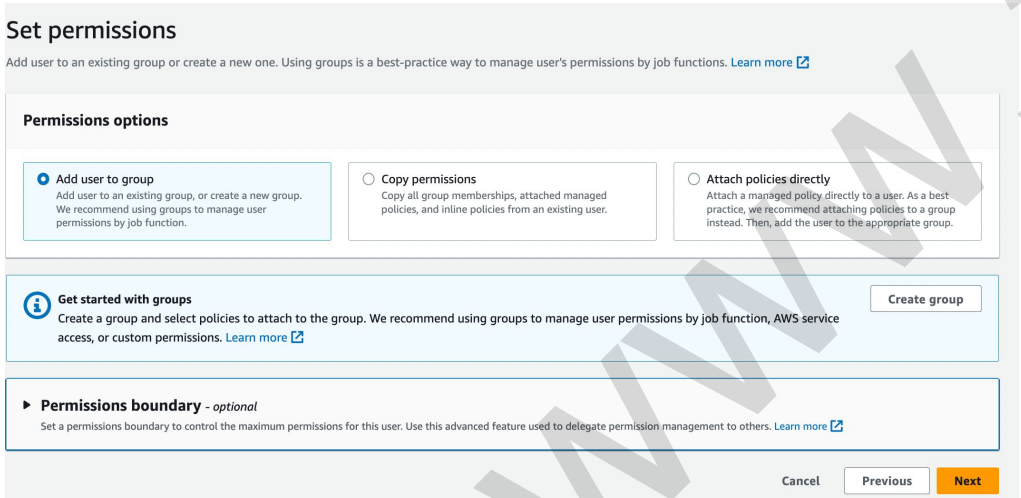
- Shared access to your AWS account
- Secure access to AWS resources for applications that run on Amazon EC2
- Multi-factor authentication (MFA)
- Identity federation

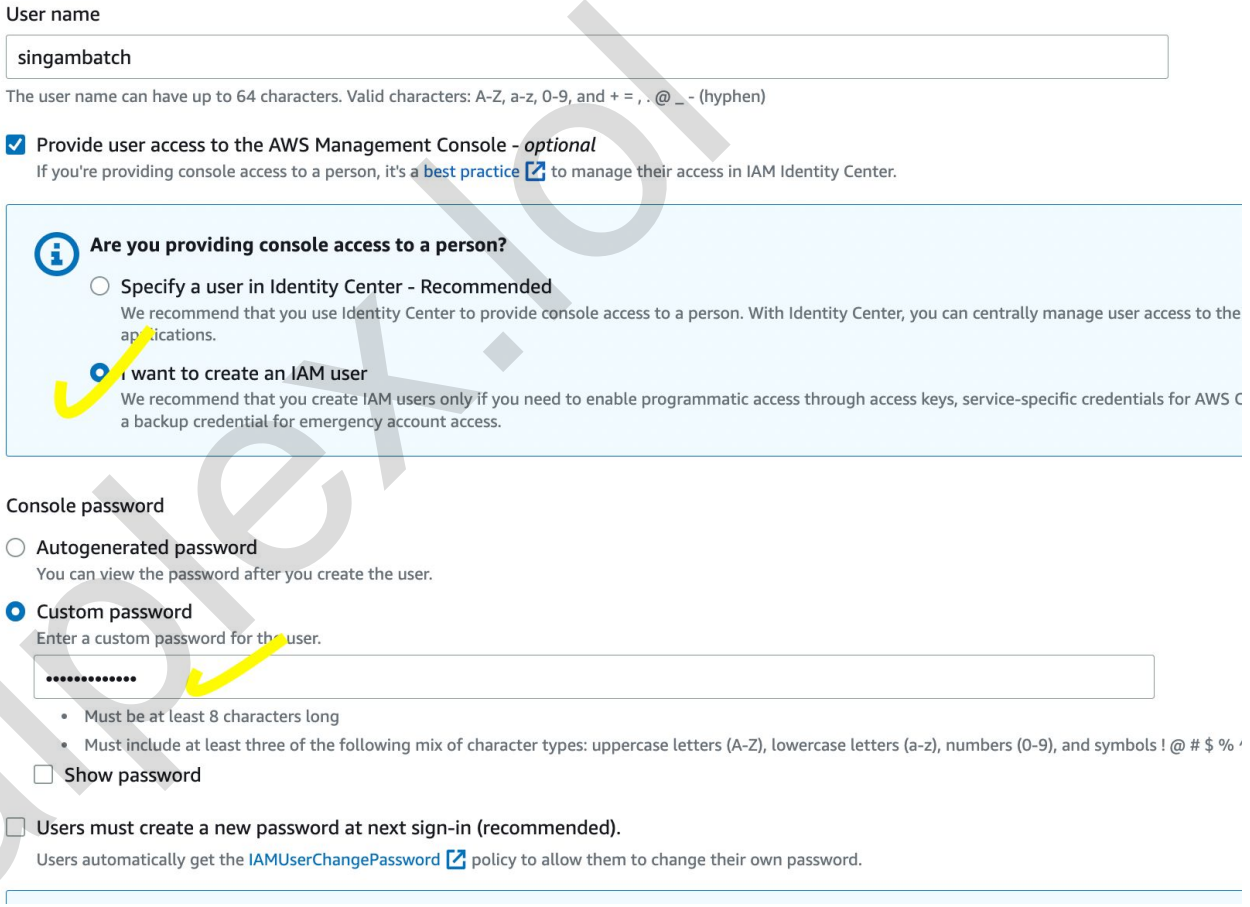




AWS LOGIN

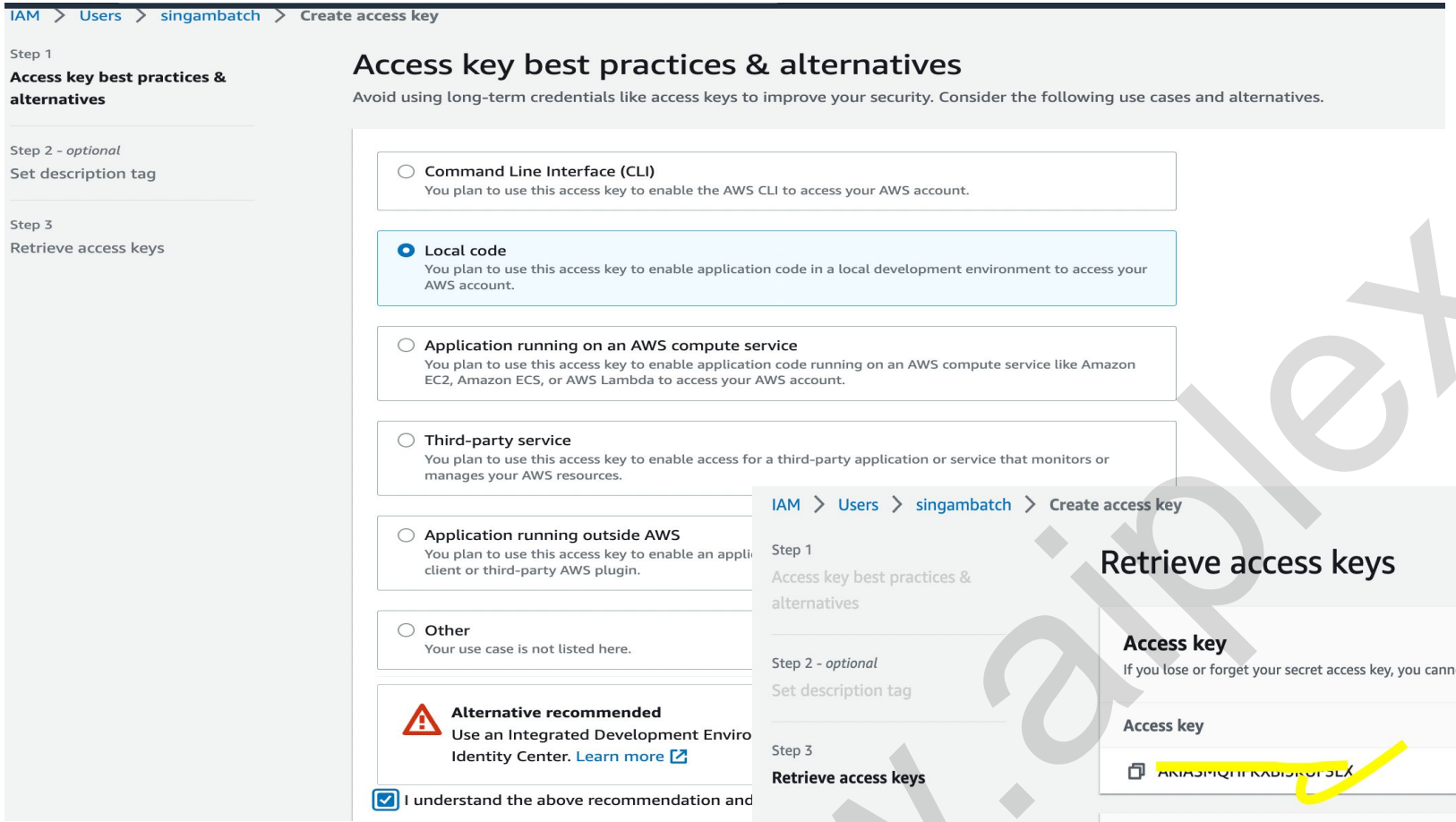
- https://aws.amazon.com/console/
- Create IAM user
- Complete and download the csv

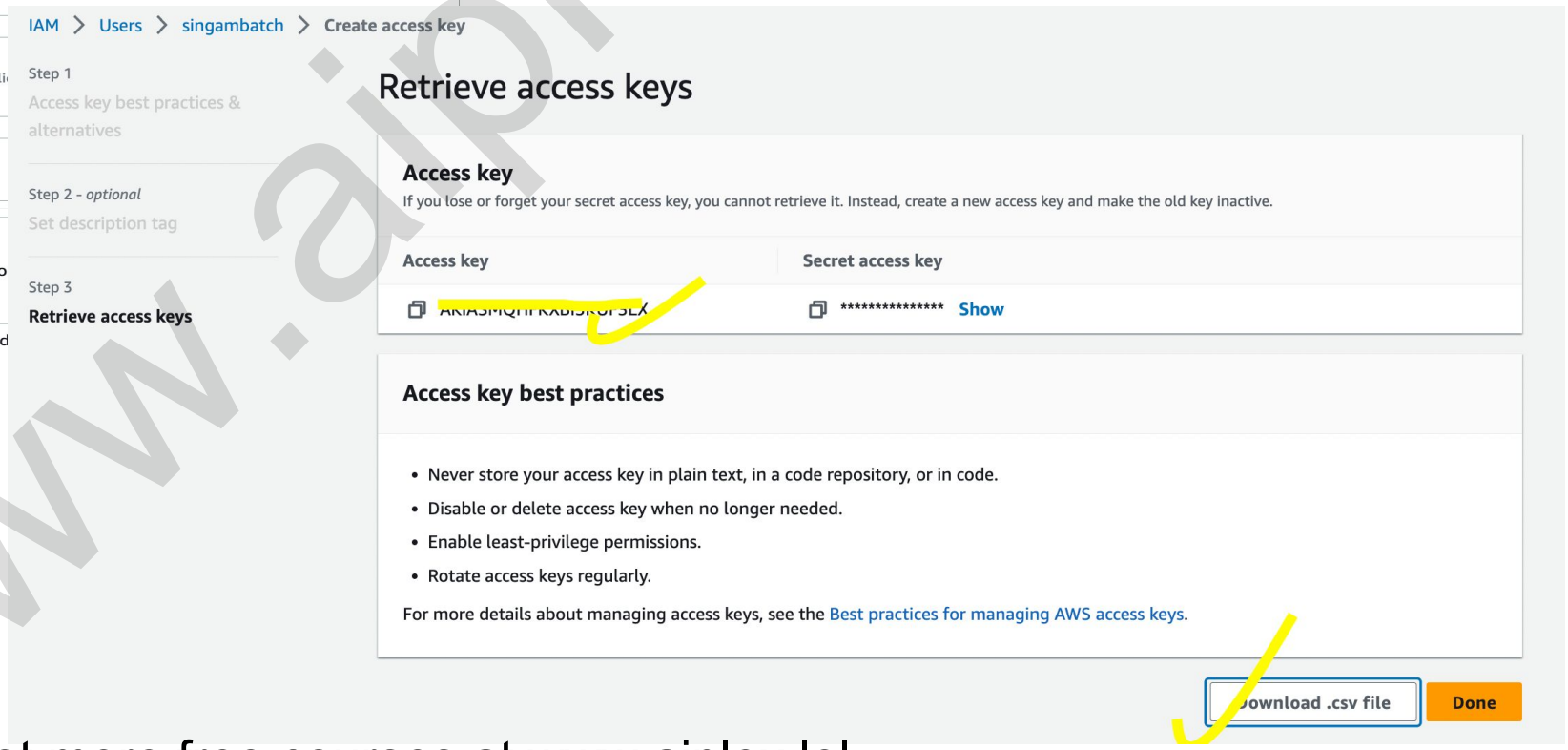




Save the file

Create the Keys





Add EC2 Permissions to User

IAM > Users > devops

devops

Go to Users -> Username-> Click on Permission-> Add policy->Attach Policy directly

Add

1) Administrator Access
2) AmzonEC2FullAcess

Summary

ARN
arn:aws:iam::164297528770:user/devops

Console access
Enabled without MFA

Created
May 28, 2023, 10:24 (UTC+05:30)

Last console sign-in
Never

Permissions | Groups | Tags (1) | Security credentials | Access Advisor

Console sign-in

Console sign-in link

Console password

Permissions | Groups | Tags (1) | Security credentials | Access Advisor

Permissions policies (2)

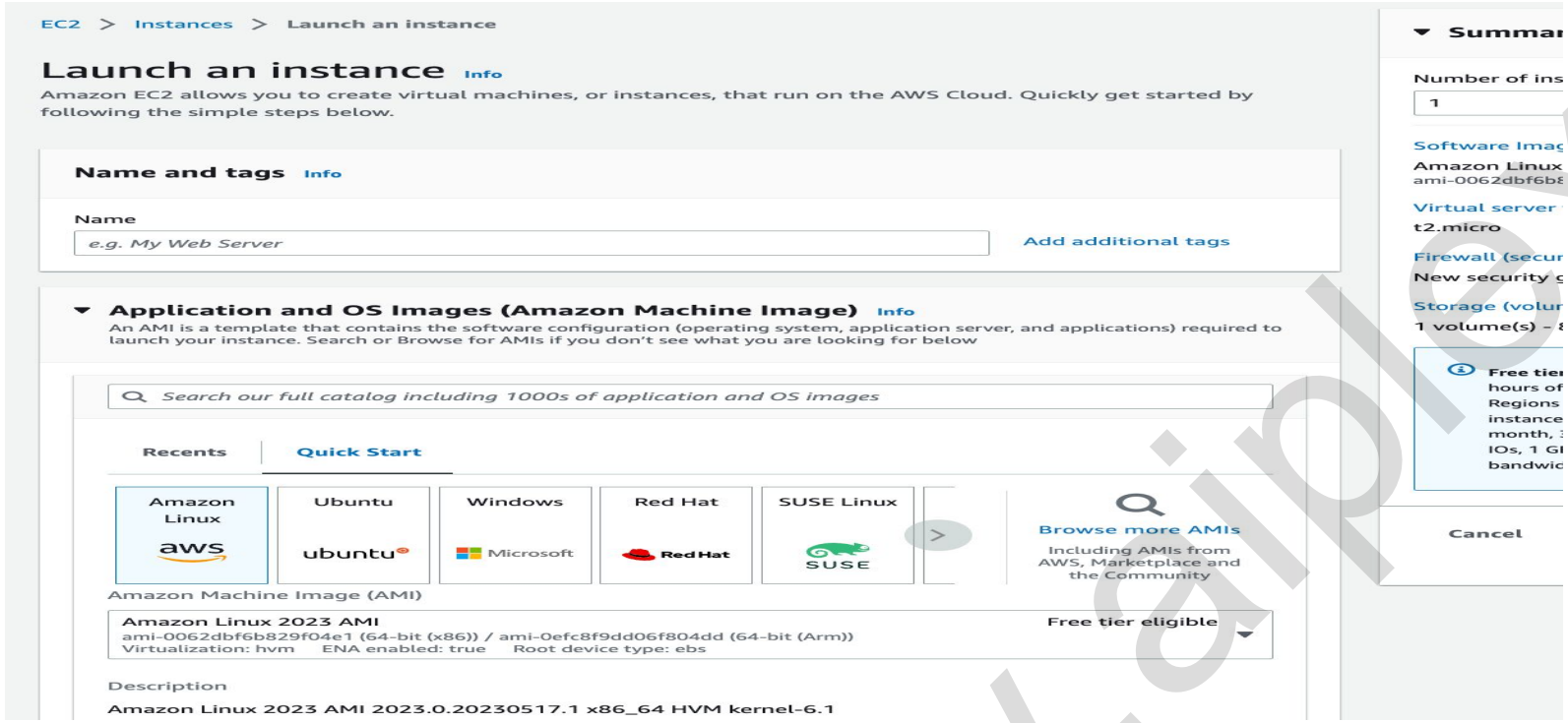
Permissions are defined by policies attached to the user directly or through groups.

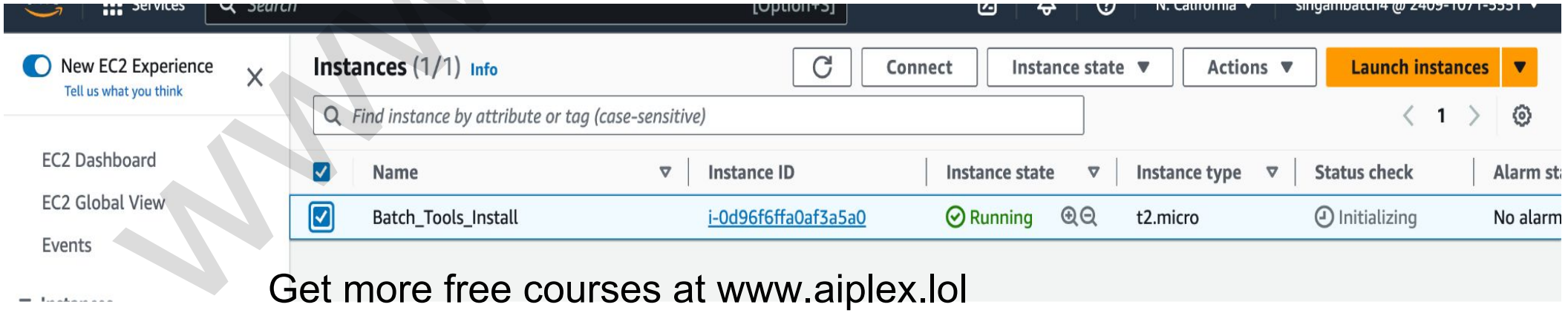
Remove | Add permissi

Search | All types | 1

Policy name	Type	Attached via
AdministratorAccess	AWS managed - job function	Directly
AmazonEC2FullAccess	AWS managed	Directly

Create the EC2 Instance in US-WEST-1 region/T2 MICRO/8 GB





SOFTWARES DOWNLOAD

1. **Visual Studio Code [IN LAPTOP TO VISUALIZE THE CODE]**

   https://code.visualstudio.com/download

   2. Python Install. —> python3 –version [IN EC2]

   yum install python

   python –-version

   3. Install AWS CLI [IN EC2]

curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"

yum install unzip

unzip awscliv2.zip

./aws/install

aws --version

4) Install Git [IN EC2]

yum install git

CREATE THE GITHUB ACCOUNT - https://github.com/DEVOPS-WITH-WEB-DEV/Splunk_Grafana_Setup.git

git clone https://github.com/DEVOPS-WITH-WEB-DEV/Splunk_Grafana_Setup.git

5) Install Java [IN EC2]

yum install java

java –version

6) Install Maven [IN EC2]

cd /opt/

wget http://mirrors.estointernet.in/apache/maven/maven-3/3.6.3/binaries/apache-maven-3.6.3-bin.tar.gz

tar xvzf apache-maven-3.6.3-bin.tar.gz

vim /etc/profile.d/maven.sh

export MAVEN_HOME=/opt/apache-maven-3.6.3

export PATH=$PATH:$MAVEN_HOME/bin

mvn -v

7**) Install Jenkins [Jenkins Always Updates the packages so make sure if any error google it]**

cd /opt/

sudo yum update –y

sudo wget -O /etc/yum.repos.d/jenkins.repo http://pkg.jenkins.io/redhat/jenkins.repo

sudo rpm --import https://pkg.jenkins.io/redhat/jenkins.io-2023.key

sudo yum install jenkins -y

systemctl daemon-reload

sudo systemctl start jenkins

sudo systemctl enable Jenkins

systemctl status jenkins

8) **Install Docker [IN EC2]**

yum install docker -y

usermod -aG docker jenkins [Add jenkins user to docker group]

systemctl start docker

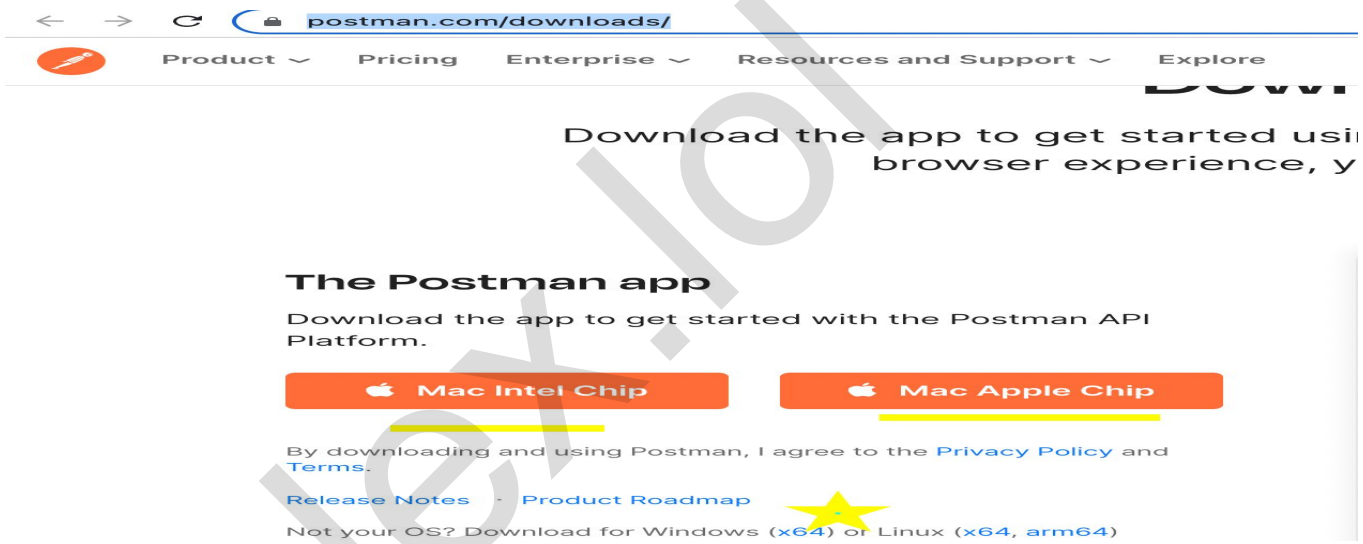
systemctl **enable** docker

9) **Install Kubectl [IN EC2]**

```
curl -o kubectl https://amazon-eks.s3-us-west-2.amazonaws.com/1.14.6/2019-08-22/bin/linux/amd64/kubectl
chmod +x ./kubectl
mkdir -p $HOME/bin
cp ./kubectl $HOME/bin/kubectl
export PATH=$HOME/bin:$PATH
echo 'export PATH=$HOME/bin:$PATH' >> ~/.bashrc
source $HOME/.bashrc
kubectl version --short –client
```

10. Install POSTMAN [In Laptop]

https://www.postman.com/downloads/



11. Install eksctl [In EC2] [AMAZON EKS]

curl --silent --location "https://github.com/weaveworks/eksctl/releases/latest/download/eksctl**_$(uname -s)_amd64.tar.gz"** | tar xz -C /tmp

sudo mv /tmp/eksctl /usr/bin

eksctl version

12. Install Node/NPM [In EC2]

Sudo yum install nodejs

node –v

npm -v

13) Install Minikube

curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64

sudo install minikube-linux-amd64 /usr/local/bin/minikube

minikube start –driver docker

[Error will come as minimum system requirement is t2.medium, later in hands-on we will take the t2.medium as we don’t want cost to occur now]

14) Terraform / Slack / other Softwares will be setup as per the sessions scheduled