Main JAVA 🡪 Addition

Need / after the path

public class Main {

public static void main(String[] args) {

int first = 10;

int second = 20;

// add two numbers

int sum = first + second;

System.out.println(first + " + " + second + " = " + sum);

}

}

javac Main.java

java Main

Add Parameterized 🡪 JAVA

public class Add {

public static void main(String[] args) {

// Check if two arguments are passed

if (args.length != 2) {

System.out.println("Please provide two integer arguments.");

return;

}

// Parse the command-line arguments

int first = Integer.parseInt(args[0]);

int second = Integer.parseInt(args[1]);

// Add the two numbers

int sum = first + second;

// Print the result

System.out.println(first + " + " + second + " = " + sum);

}

}

javac Add.java

java Add %fisrt% %second%

Add Parameterized 🡪 PYTHON

import sys

def main():

if len(sys.argv) != 3:

print("Please provide two numbers.")

return

first = int(sys.argv[1])

second = int(sys.argv[2])

sum\_ = first + second

print(f"{first} + {second} = {sum\_}")

if \_\_name\_\_ == "\_\_main\_\_":

main()

python sum\_numbers.py %fisrt% %second%

Main PYTHON 🡪 Addition

def main():

first = 10

second = 20

# Add the two numbers

sum\_ = first + second

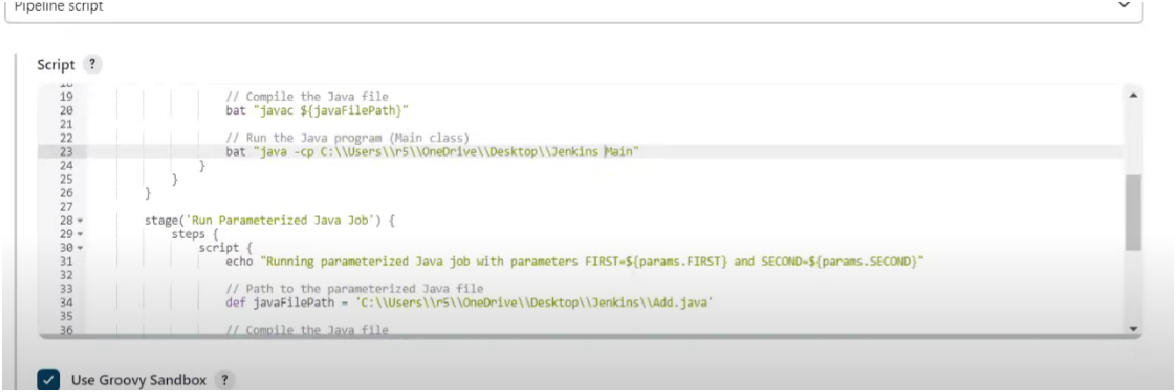
# Print the result

print(f"{first} + {second} = {sum\_}")

if \_\_name\_\_ == "\_\_main\_\_":

main()

pipeline



Docker

docker –version

docker ps 🡪showing running applications.

docker start container\_name 🡪 Start container.

docker stop container\_name 🡪 Stop container

docker remove container\_name 🡪 remove container

docker pull ubuntu

docker images

docker run -it ubuntu /bin/bash

ls

pwd 🡪 present working directory

apt update 🡪 a command that updates the package index files on your system with the latest information about available packages and their versions (sudo apt update)

exit

docker pull python

docker run -it python

print("Hello, Docker!")

exit()

docker run -it -v C:/Users/admin/OneDrive/Desktop/Docker-31:/usr/src/myapp -w /usr/src/myapp python python sum\_fixed\_numbers.py

sum\_fixed\_numbers.py

def main():

first = 10

second = 20

# Add the two numbers

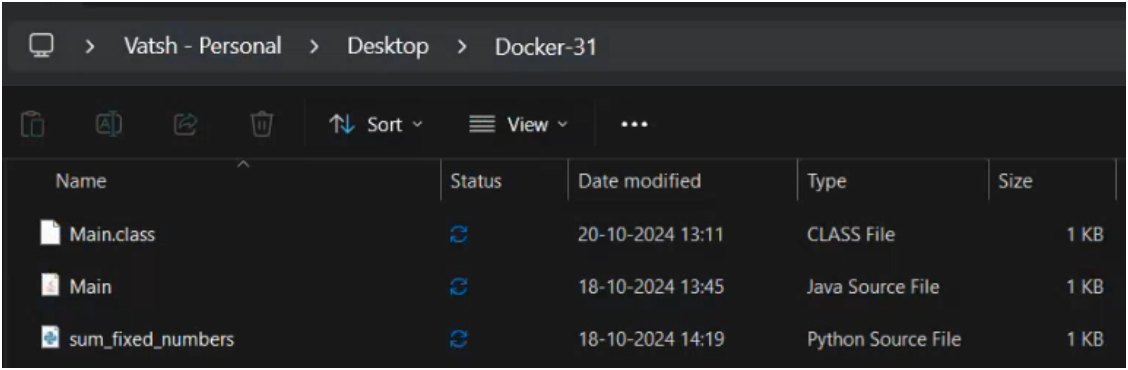
sum\_ = first + second

# Print the result

print(f"{first} + {second} = {sum\_}")

if \_\_name\_\_ == "\_\_main\_\_":

main()



docker pull openjdk

docker images

docker run -it -v C:/Users/admin/OneDrive/Desktop/Docker-31:/usr/src/myapp -w /usr/src/myapp openjdk javac Main.java

docker run -it -v C:/Users/admin/OneDrive/Desktop/Docker-31:/usr/src/myapp -w /usr/src/myapp openjdk java Main

***10+20=30***

How to push image in the docker hub?

docker push <username>/<repository>

How to craete repository?

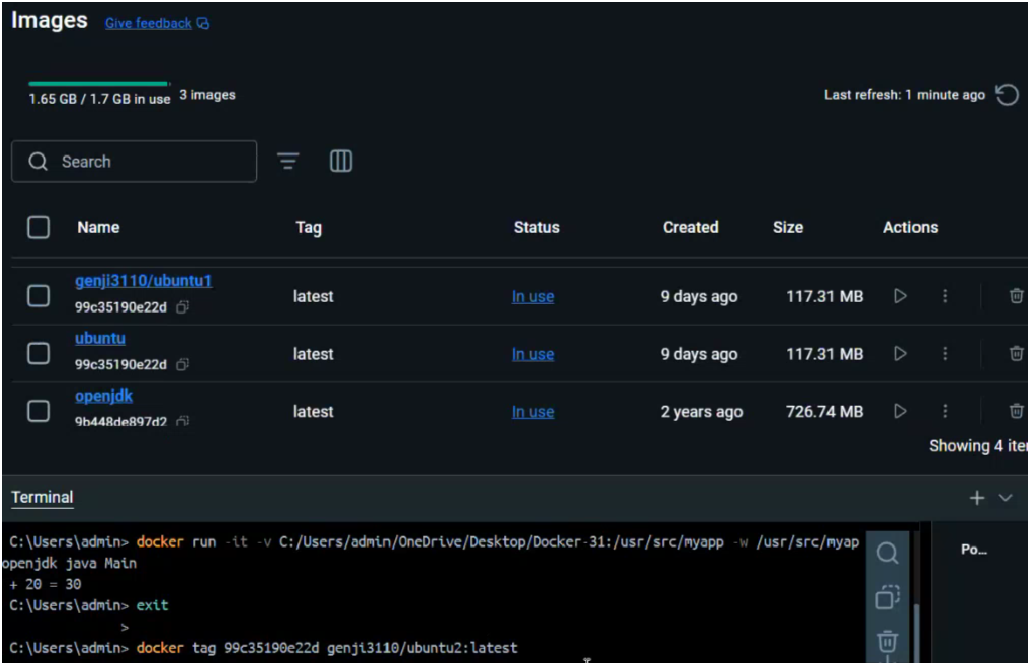
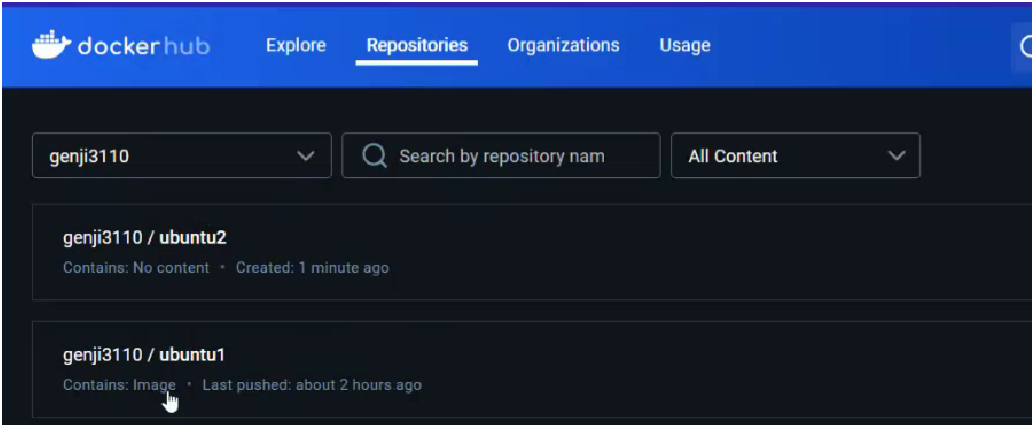
Got o dockor hub🡪 Repository🡪 put the name, public/ private

Push the images in the hub

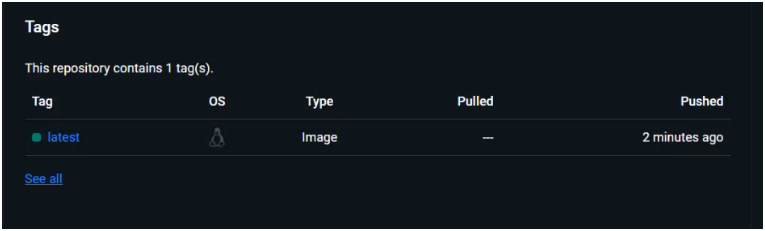
docker tag 99c35190e22d genji3110/ubuntu1:latest

99c35190e22d🡪 code of imges which one we want to push

genji3110/ubuntu1🡪 where we want to push the image.

docker push genji3110/ubuntu1:latest



Java Code

public class Main {

public static void main(String[] args) {

int first = 10;

int second = 20;

// add two numbers

int sum = first + second;

System.out.println(first + " + " + second + " = " + sum);

}

}

**GIT**

It is a version control system.

Version control system is a track changes in code.

It is open source, free, scalable , fast.

**GitHUb**

It is a website that alllows developers to store and manage their code using git

Readme file🡪 all information about project, how to use, which features are their in the project.

Name of repository🡪 public🡪 Add a Readme file🡪 Then its shows the new repository.

Add 🡪 commit

git —version 🡪 show version

pwd 🡪 path of working directory

Configure Git

We can configure the Git in two ways-🡪 global and local level

Global 🡪We can do the configuration in the overall project using only that email id.

Local🡪 we can do the chnages in the repository or a project using the different email id.

If we have a single account in the git then use the global.

git config –global user.name “Account\_name”

git config –global user.email “email\_address”

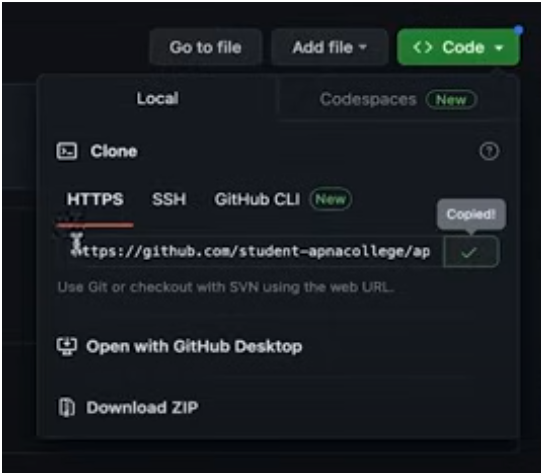
git config –list

Gitdemo folder in the vscode.

Clone🡪 Cloning a repository on our local machine(duplicate the system)

Git clone link\_of\_the\_project

Go to the



cd 🡪 change directory

cd project\_name

ls 🡪 shows file in the project

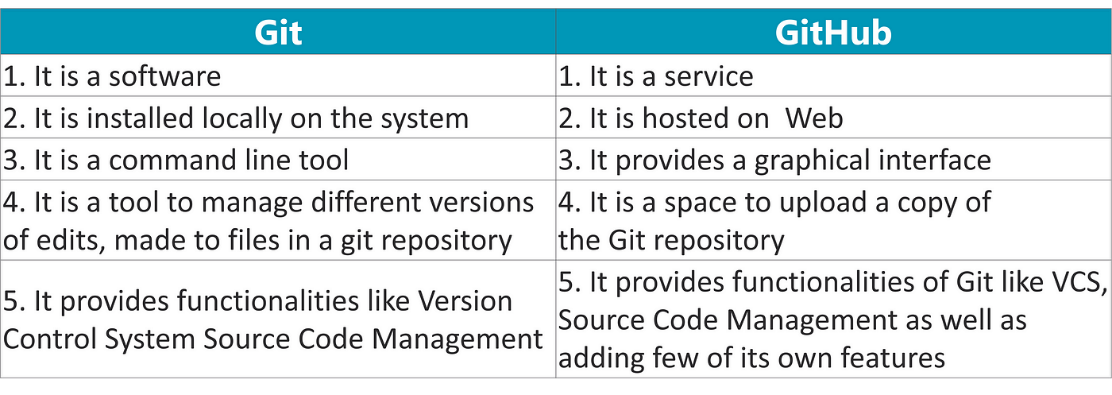
ls -a 🡪 shows hiden files also

**git branch**

1.Lists all branches in the repository or creates a new branch.

git branch 🡪List the branch

git branch <branch-name> 🡪To create a new branch:



git status🡪The git status command displays the state of the working directory and the staging area. (e.g., modified🡪changed,

unmodified🡪unchanged,

staged🡪file is ready to be committed, or

untracked🡪new files that git doesn’t yet track).

git init🡪 Initializes a new Git repository in the current directory.

*Git add*

git add <file> 🡪 Stages changes (modified or new files) to be committed.

git add🡪 To stage all changes

git commit -m "Commit message"🡪 Commits the staged changes to the repository with a message describing the changes.

1. AWS S3 Bucket

Steps to Create an S3 Bucket:

Log in to the AWS Management Console:

Go to AWS Console and log in.

Navigate to S3:

Search for "S3" in the AWS services and select it.

Create a New Bucket:

Click on “Create bucket.”

Enter a unique bucket name (bucket names must be globally unique).

Select a region.

Configure any necessary options (e.g., enabling versioning, encryption).

Click "Create bucket."

Steps for File Upload, Download, and Delete:

Upload a file:

Open the bucket you just created.

Click “Upload” and choose the file you want to upload from your computer.

Click "Upload" to confirm.

Download a file:

Navigate to the file in your bucket.

Click the file and choose "Download."

Delete a file:

Select the file(s) you want to delete.

Click “Actions” > “Delete” and confirm.

2. IAM User Creation and Permissions

Steps to Create an IAM User:

Go to IAM:

In the AWS Console, search for and open "IAM" (Identity and Access Management).

Create a New User:

In the left-hand panel, click “Users” and then “Add user.”

Enter a username.

Select the type of access for the user:

Programmatic access (for access keys, used for CLI or SDK).

AWS Management Console access (for logging into the console).

Set permissions for the user (either use an existing policy or create a custom one).

Set Permissions:

Choose “Attach existing policies directly” to give predefined permissions (e.g., S3 Full Access).

Optionally, add the user to a group with specific policies.

Review and Create:

Review the details and click “Create user.”

Save the access key and secret key if creating a programmatic user.

3. Create a Bucket in AWS CloudShell

CloudShell provides a terminal within the AWS Console where you can run AWS CLI commands without needing a local setup.

Steps:

Open AWS CloudShell:

In the AWS Console, click the “CloudShell” icon (typically in the top-right corner of the page).

Create an S3 Bucket using AWS CLI:

In the CloudShell terminal, run the following command to create a bucket:

bash

Copy code

aws s3 mb s3://your-bucket-name --region your-region

Replace your-bucket-name with a unique name and your-region with your preferred AWS region (e.g., us-east-1).

Upload a File to the S3 Bucket:

Once the bucket is created, you can upload a file using:

bash

Copy code

aws s3 cp /path/to/yourfile s3://your-bucket-name

List Files in the Bucket:

To list all files in the bucket:

bash

Copy code

aws s3 ls s3://your-bucket-name

Download a File from S3:

To download a file from your S3 bucket:

bash

Copy code

aws s3 cp s3://your-bucket-name/yourfile /local/destination

Delete a File:

To delete a file from the bucket:

bash

Copy code

aws s3 rm s3://your-bucket-name/yourfile

4. EC2 (Elastic Compute Cloud)

Steps to Launch an EC2 Instance:

Navigate to EC2:

In the AWS Console, search for and open "EC2."

Launch an EC2 Instance:

Click “Launch instance.”

Select an Amazon Machine Image (AMI) (e.g., Ubuntu, Amazon Linux).

Choose an instance type (e.g., t2.micro for free tier).

Configure security settings:

Allow inbound rules for SSH (port 22) if you need SSH access.

Add HTTP (port 80) if you want to host a webpage.

Click “Launch” and select/create a key pair.

Steps to Upload EC2 Webpage from AWS to GitHub:

SSH into the EC2 Instance:

If you’re using an Ubuntu instance:

bash

Copy code

ssh -i /path/to/your-key.pem ubuntu@your-ec2-public-ip

Install Git and Apache on the EC2 instance:

bash

Copy code

sudo apt update

sudo apt install git apache2 -y

Clone Your Website Repository from GitHub:

bash

Copy code

git clone https://github.com/your-username/your-repo.git

Move the Website Files to Apache Directory:

bash

Copy code

sudo cp -r your-repo/\* /var/www/html/

Start Apache Web Server:

bash

Copy code

sudo systemctl start apache2

Access the Webpage:

Visit the public IP of your EC2 instance in a browser (e.g., http://your-ec2-public-ip).

These steps will help you prepare for the AWS-related tasks in your DevOps practicals. Let me know if you need more details on any specific step!