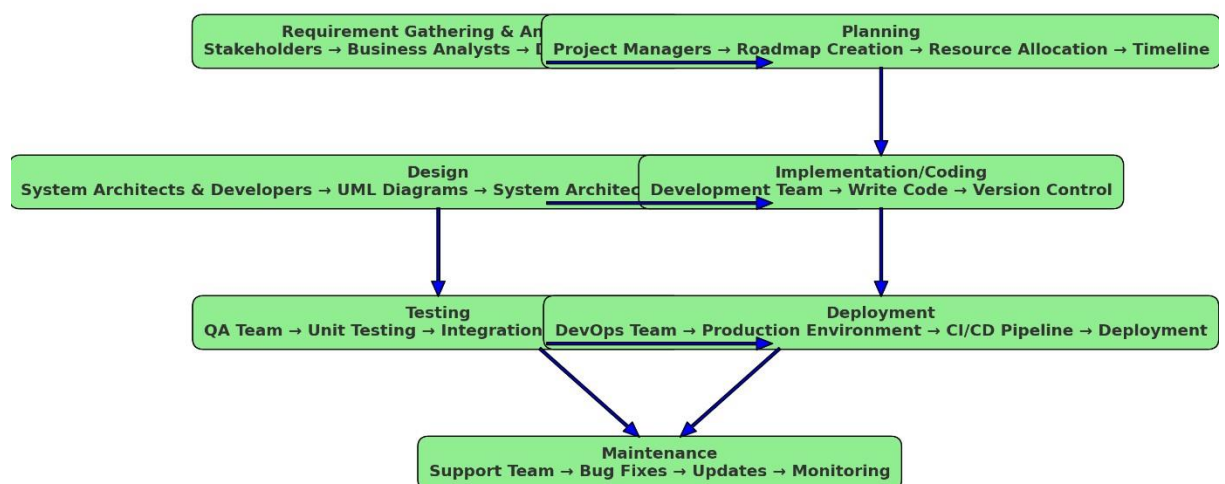


## **DEVOPS**

<b>No:-</b>	<b>Content</b>
<b>1.</b>	<b>DevOps theory</b> <ul style="list-style-type: none"><li>• Stages of software development (SDLC)</li><li>• What is Waterfall model</li><li>• What is Agile model</li><li>• Difference between waterfall and agile model</li><li>• What is DevOps</li></ul>
<b>2.</b>	<b>Version Control</b> <ul style="list-style-type: none"><li>• What is Git and GitHub</li><li>• How to push and pull sources onto remote repository</li></ul>
<b>3.</b>	<b>Git cloning using(https and SSH)</b>
<b>4.</b>	<b>Git pull and fetch</b>

## DevOps theory

- **Stages of software development**
  - 1.Requirement gathering and analysis.
  - 2.Planning
  - 3.Design
  - 4.Implementation and coding
  - 5.Tesing
  - 6.Deployment
  - 7.Maintenance



- **What is Waterfall model:** - The Waterfall model is a simple and traditional approach to software development. In this model, the process is like a waterfall flowing down through different stages, and each stage must be completed before moving to the next one. Once a stage is finished, you can't go back to it easily.

- **What is Agile model:** - The Agile model is a flexible and iterative approach to software development. Instead of delivering the entire project at once, work is broken down into small, manageable pieces called "sprints" (usually 2-4 weeks long). Each sprint focuses on delivering a part of the project that is functional and usable.
- **Difference between waterfall and agile model**

Aspect	Agile	Waterfall
Life Cycle	It is a continuous iteration life cycle model to develop and test a software product.	It is a linear sequential model to develop and test a software product.
Process	In this The entire process of development is divided into sprints	The <u>software development</u> process is broken down into different phases.
Flexibility	Agile development model is flexible to make changes at any point of time (or at any stage of development process) .	In Waterfall model to make changes after one phase is difficult and costly.
client involvement	Continuous client Interaction and feedback	There is very little client involvement and very little feedback is taken.
Delivery Time	Its delivery time is very short and functional software is available very quickly.	Its delivery time is very long, the entire project must be completed before delivery.

## **What is DevOps**

### **Plan:**

- Gather requirements and decide what features to develop.
- Use tools like Jira or Trello to track tasks.

### **Code:**

- Developers write the code for the features.
- Version control tools like Git are used to manage code.

### **Build:**

- Combine code from different developers to create a build.
- Tools like Jenkins or Maven help automate this step.

### **Test:**

- Automated tests are run to check if the code works correctly.
- Tools like Selenium or JUnit are used for testing.

### **Release:**

- Once testing is successful, the build is ready for release.
- Tools like Jenkins or Docker help in deploying the build.

**Deploy:**

- The code is deployed to production servers.
- Tools like Kubernetes or AWS help manage deployment.

**Operate:**

- The application runs in the production environment.
- Monitoring tools like CloudWatch or Nagios track performance.

**Monitor:**

- Check if the application is performing well or has issues.
- Any problems found are reported, and the cycle starts again.

**What is Git and GitHub**

**Git** is a version control system that helps developers track changes to files and collaborate on projects. It allows multiple people to work on a project at the same time without interfering with each other's work. Key features of Git include:

- **Version tracking:** Keeps a history of all changes made to the project.
- **Branching:** Allows developers to work on different features or fixes independently.
- **Merging:** Combines changes from different branches back into the main project.

**GitHub**, on the other hand, is a cloud-based platform built around Git. It provides a space for developers to host and manage Git repositories online. Key features of GitHub include:

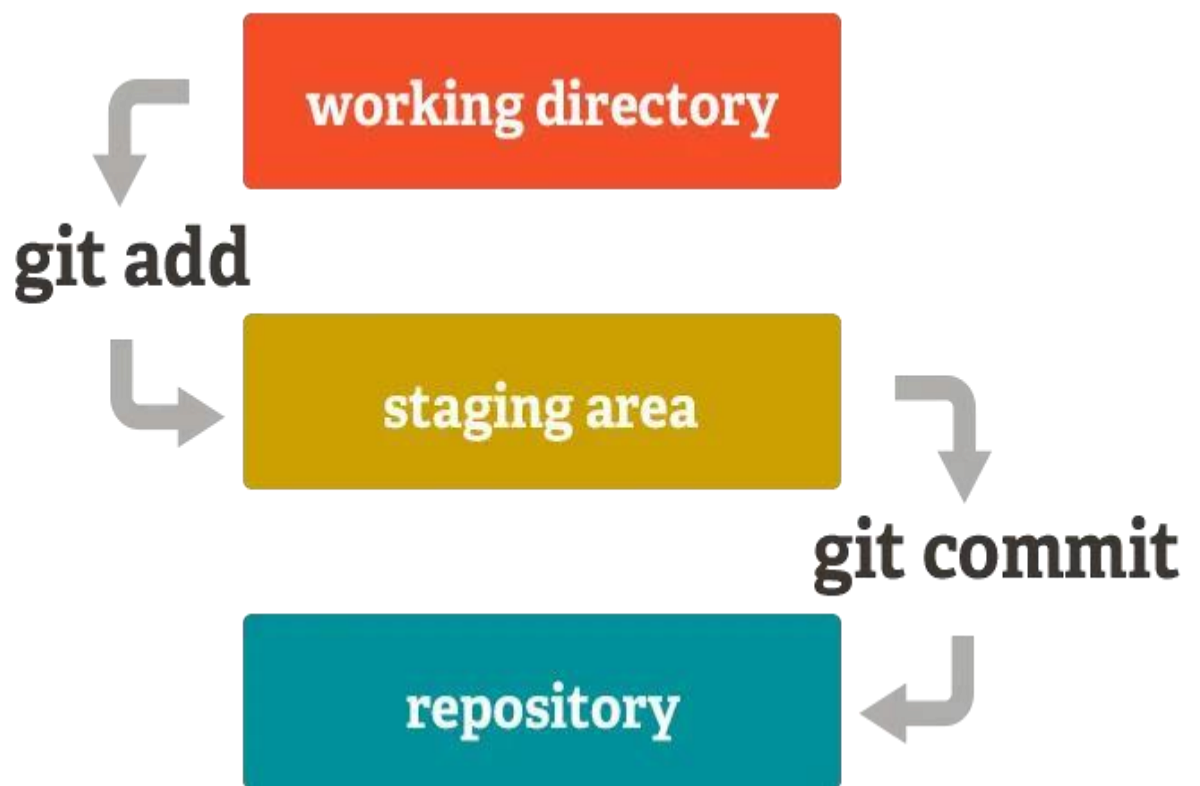
- **Collaboration:** Allows developers to share their code with others and collaborate easily.

- **Pull requests:** A feature that helps in reviewing, discussing, and merging code changes.
- **Project management:** Offers tools like issue tracking and project boards to manage software development.

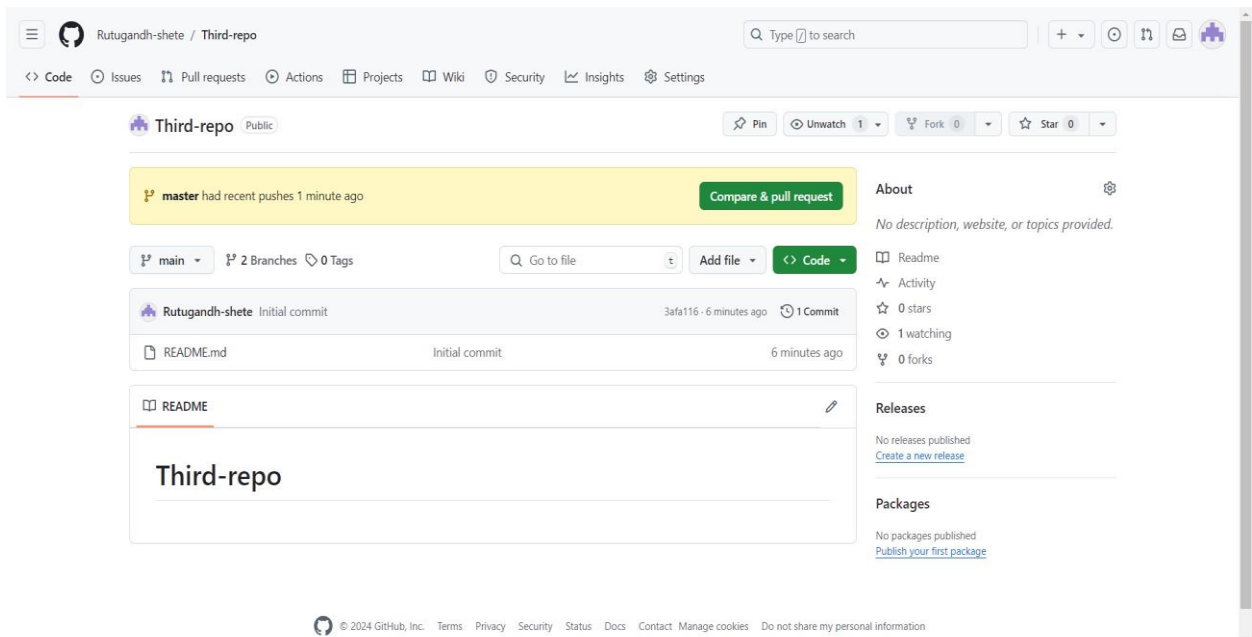
### How to push and pull sources onto remote repository:

#### Steps:

- First, we will consider one diagram.



- Create one repo in GitHub account



- Let's create file.

```
rutugandh@LAPTOP-V2RDGVA6:~$ touch 1.html
rutugandh@LAPTOP-V2RDGVA6:~$ ls
1.html
```

- This command initializes a new Git repository in your current project directory. It creates a .git folder, which contains all the metadata and history for version control.
- Git config --global: This command is used to set global configuration options for Git, such as your username and email. These settings will apply to all your repositories on your machine.

```
git config --global user.mail "rutu123@gmail.com"
git config --global user.name "rutu"
```

- Git config --global --list: To see the list of username and Email.

```
rutugandh@LAPTOP-V2RDGVA6:~$ git config --global --list
user.mail=rutu123@gmail.com
user.name=rutu
```

- Git add: This command stages a specific file for the next commit.

```
rutugandh@LAPTOP-V2RDGVA6:~$ git add 1.html
rutugandh@LAPTOP-V2RDGVA6:~$
```

- To check status use git status

```
rutugandh@LAPTOP-V2RDGVA6:~$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   1.html
```

- Git commit is used to commit code in local repo

```
rutugandh@LAPTOP-V2RDGVA6:~$ git commit -m "hello third commit"
[master b8e2707] hello third commit
Committer: rutu <rutugandh@LAPTOP-V2RDGVA6.localdomain>
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:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

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

    git commit --amend --reset-author

1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 1.html
```

- Git log to check committed code/sources

```
rutugandh@LAPTOP-V2RDGVA6:~$ git log
commit b8e2707e8cc124243ff220877e622e733a596391 (HEAD -> master)
Author: rutu <rutugandh@LAPTOP-V2RDGVA6.localdomain>
Date:   Thu Sep 26 08:44:44 2024 +0530

    hello third commit

commit f7aa11d6e1c763b17d31cb846e198eaa287eaaf1 (origin/master)
Author: rutugandhshete <rutugandh@LAPTOP-V2RDGVA6.localdomain>
Date:   Tue Sep 24 12:13:01 2024 +0530
```

- Git remote add origin

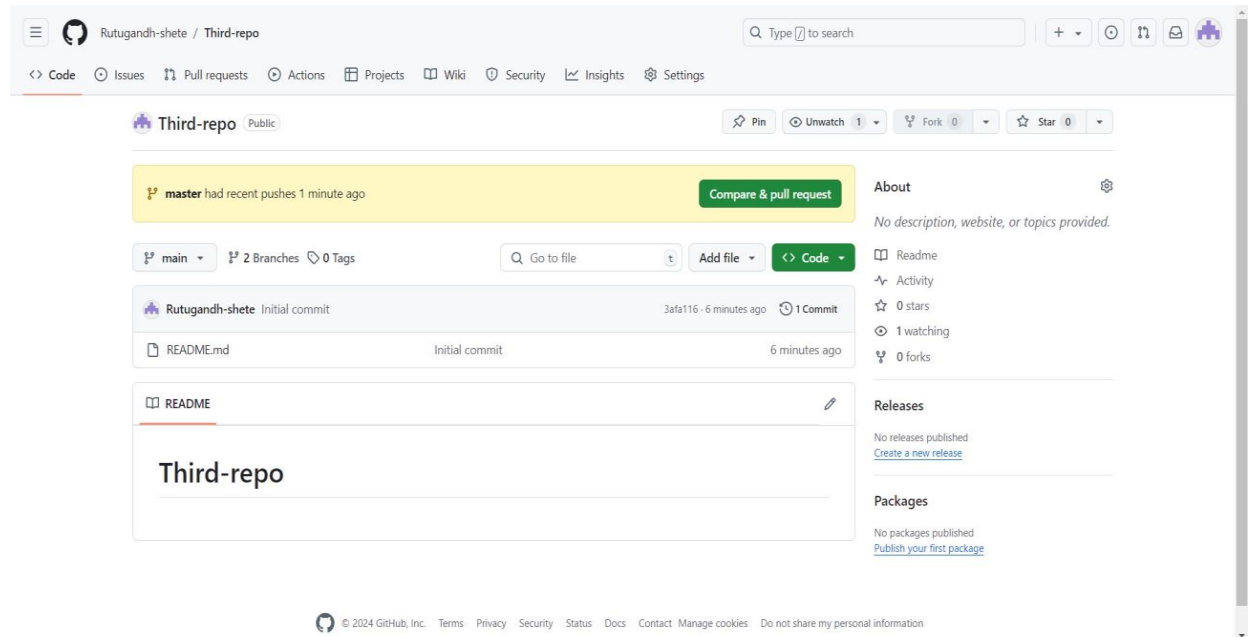
```
rutugandh@LAPTOP-V2RDGVA6:~$ git remote add origin1 https://github.com/Rutugandh-shete/Third-repo.git
rutugandh@LAPTOP-V2RDGVA6:~$
```

- Git push origin1 master “paste github link” and add username password(create token)

```
rutugandh@LAPTOP-V2RDGVA6:~$ git push origin1 master
Username for 'https://github.com': Rutugandh-shete
Password for 'https://Rutugandh-shete@github.com':
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 431 bytes | 107.00 KiB/s, done.
Total 5 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/Rutugandh-shete/Third-repo/pull/new/master
remote:
To https://github.com/Rutugandh-shete/Third-repo.git
 * [new branch]      master -> master
```



- One master branch is created and one 1.html is pushed into repo.



## Git cloning using (https and SSH):

### Steps:

- Git cloning using https “git clone “link” ”
- Git cloning using SSH:
  1. Firstly, use command “ssh-keygen”. It is used to create public key.

```
rutugandh@LAPTOP-V2RDGVA6:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/rutugandh/.ssh/id_rsa):
Created directory '/home/rutugandh/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/rutugandh/.ssh/id_rsa
Your public key has been saved in /home/rutugandh/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:zk/AtsLGMSkyUvskCaz64MejTZu8ClckeGzqrmiMfU rutugandh@LAPTOP-V2RDGVA6
The key's randomart image is:
+---[RSA 3072]---+
|
|.
| * B . o
| . & o o o
| . = + . S
|.oo* . B +
|=B+oE . B .
|*O* . o
|@=..
+---[SHA256]-----+
```

2. Use `ls -la` to see .ssh hidden file to configure to remote repo using ssh.

```
rutugandh@LAPTOP-V2RDGVA6:~$ ls -la
.  .bash_history  .bashrc  .gitconfig  .motd_shown  .ssh  .viminfo  new.sh
.. .bash_logout  .git      .lessht     .profile     .sudo_as_admin_successful  1.html
```

3. Change directory into .ssh “cd .ssh”

```
rutugandh@LAPTOP-V2RDGVA6: ~/.ssh$ ls -la
.  ..  id_rsa  id_rsa.pub
```

4. Then read file id\_rsa.pub file which contain key and copy it

```
rutugandh@LAPTOP-V2RDGVA6: ~/.ssh$ cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGCksGuqS9EMYY0JB+Ps+kZPD1wJpCjrcA70F6Ufr9Q3tNginZLCHYqeRuqLetxfe126t0ijvJmM9fMD/+iv
zhLrykHuhDn0LZNCLeEHq82IF3crmb0uF0GVJ1/jxUejsI5USjYVB2eM6VKH4fj9NxoXDWewcGpymvsJQKIR8jfRajCIDItiDHbZIKGh3MzMuJzezOSTZGt
1pD964DRJdkOGID12RXwzzdoUwXlN0/yT/paa7JiHLSlEJg2Fxm+v8lW9TVBwJDz80sMKnmZlbFyxFRU+vYjJtvhFvNpP28cAQdLeL7w6CDG+TdhRLYqcdG
7e8eLG7AU2jX/9FR82CbZ3CF6KPEYE02ouYodtLVEB70GEyJMI071gmJB/x65vfc0v4pGadwETvwhj4P0VPsFt7xa02vI8UQZkjKjH6RwJe5TVnHzcXoR0uP
RVmhqaT6d/zb4b1C0S0dFX5GhT/6qitILqM+pRd9xBa50C+00eQVFmWcMwyQTViR10Qo/Sc= rutugandh@LAPTOP-V2RDGVA6
```


5. Go to github account → settings → SSH and GPG keys → create new ssh key and paste that id\_rsa.pub key

#### SSH keys

[New SSH key](#)

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

#### Authentication keys



rutugandh@LAPTOP-V2RDGVA6

SHA256:zk/Ats1GMSKyUvskCaz64MejTzu8C1ckeGzqrm1MfU

Added on Sep 26, 2024

Last used within the last week — Read/write

SSH

Delete

Check out our guide to [connecting to GitHub using SSH keys](#) or troubleshoot [common SSH problems](#).

6. Git clone “ssh\_link”

### Git pull and fetch: git pull origin main

**git pull:** Fetches the latest changes from the remote repository and *automatically merges* them into your current branch.

### git fetch: git fetch origin

Only downloads the latest changes from the remote repository but **does not merge** them. You have to merge them manually if needed.

### git restore: git restore <file\_name>

git restore is used to undo changes in your working directory.