/kahootali

DevOps - Week 1 - Intro to Git

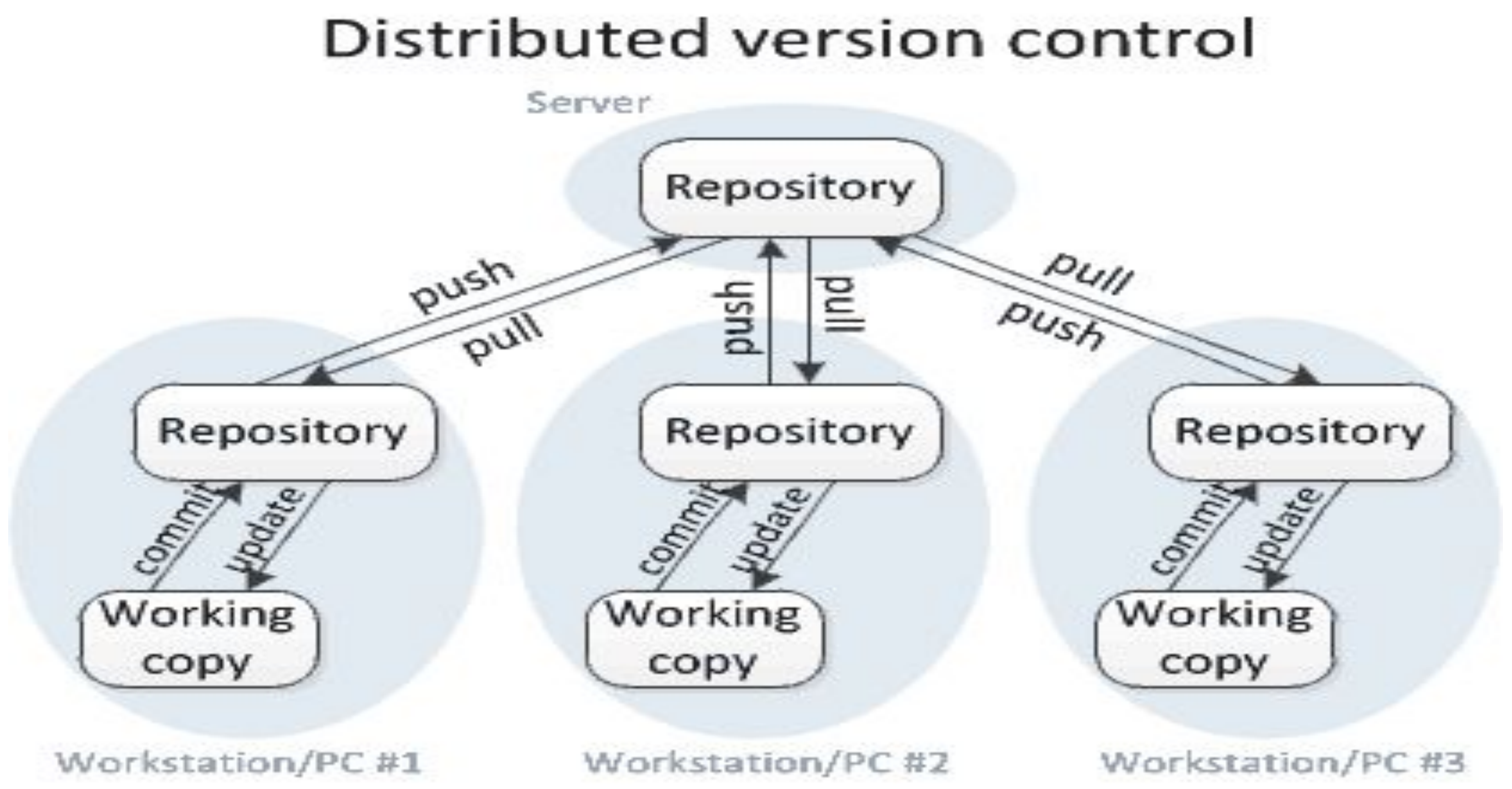
Muhammad Ali Kahoot
Dice Analytics

Disclaimer

QUIZ

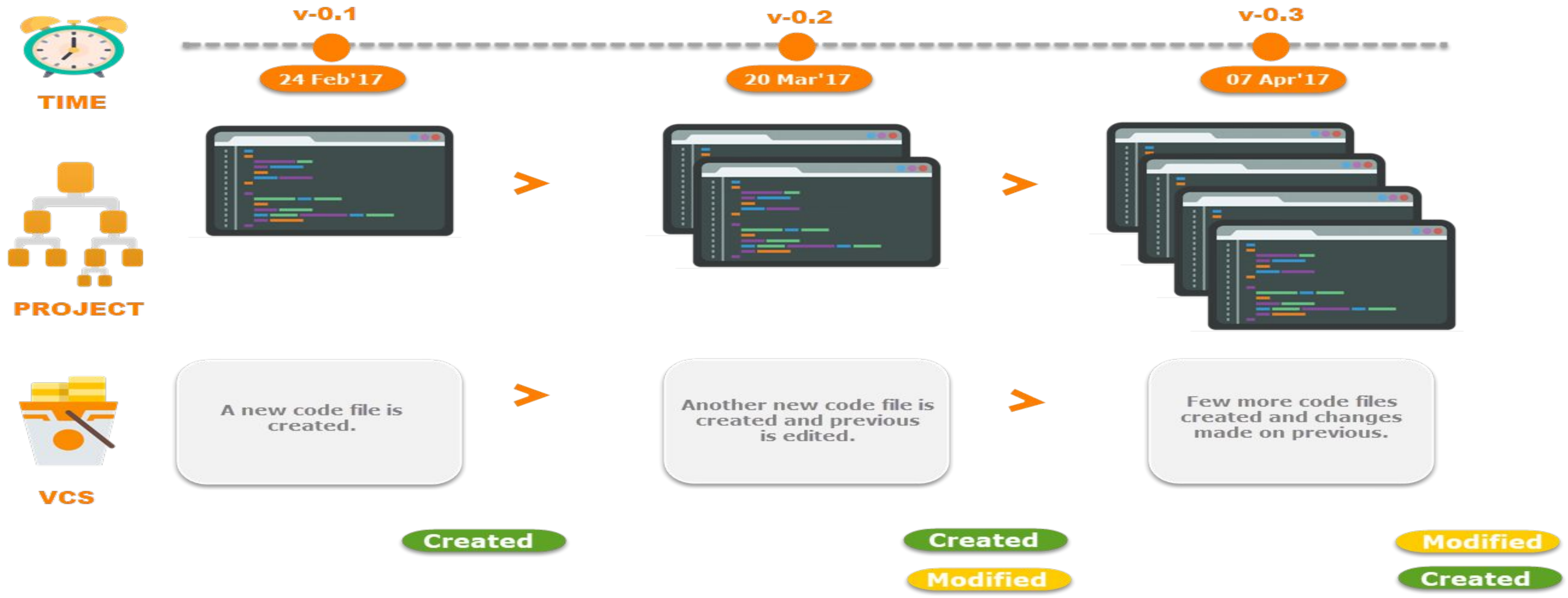
- What is DevOps
- Explain Microservices
- What are APIs
- CI vs Continuous Delivery vs Continuous Deployment
- Write benefits of Cloud Computing
- What is Logging & Monitoring

Version Control System



/kahootali

Version Control System


TIME
v-0.1
24 Feb'17
v-0.2
20 Mar'17
v-0.3
07 Apr'17
PROJECT
VCS
A new code file is created.
Another new code file is created and previous is edited.
Few more code files created and changes made on previous.
Created
Created
Modified
Modified
Created

DevOps Course By M. Ali Kahoot - Dice Analytics

Why Version Control?

- Multiple Developers working on same code base
- Central Cloud Backup
- Incremental Progress, Know which version is live
- Branching -> multiple branches for different environments
- Change History? Can blame/praise the Commit/Developer

Version Control System Tools

SUBVERSION

git

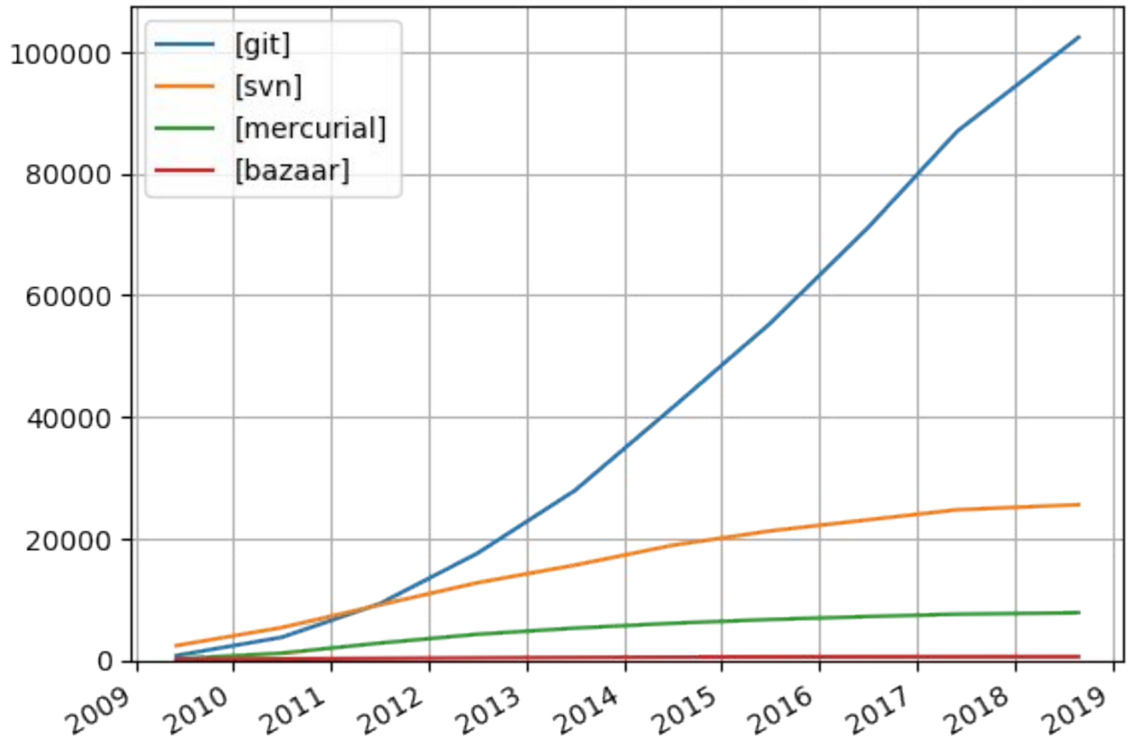
VS2012
Team Foundation

Bazaar

mercurial

/kahootali

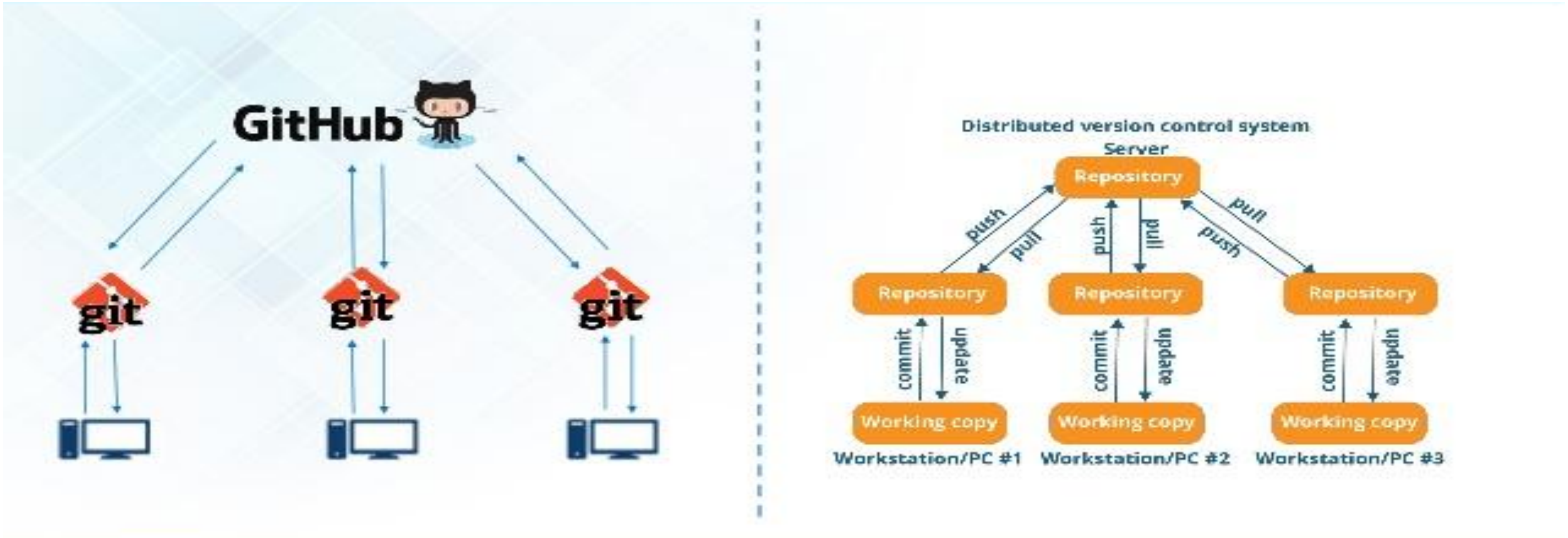
Version Control System Tools



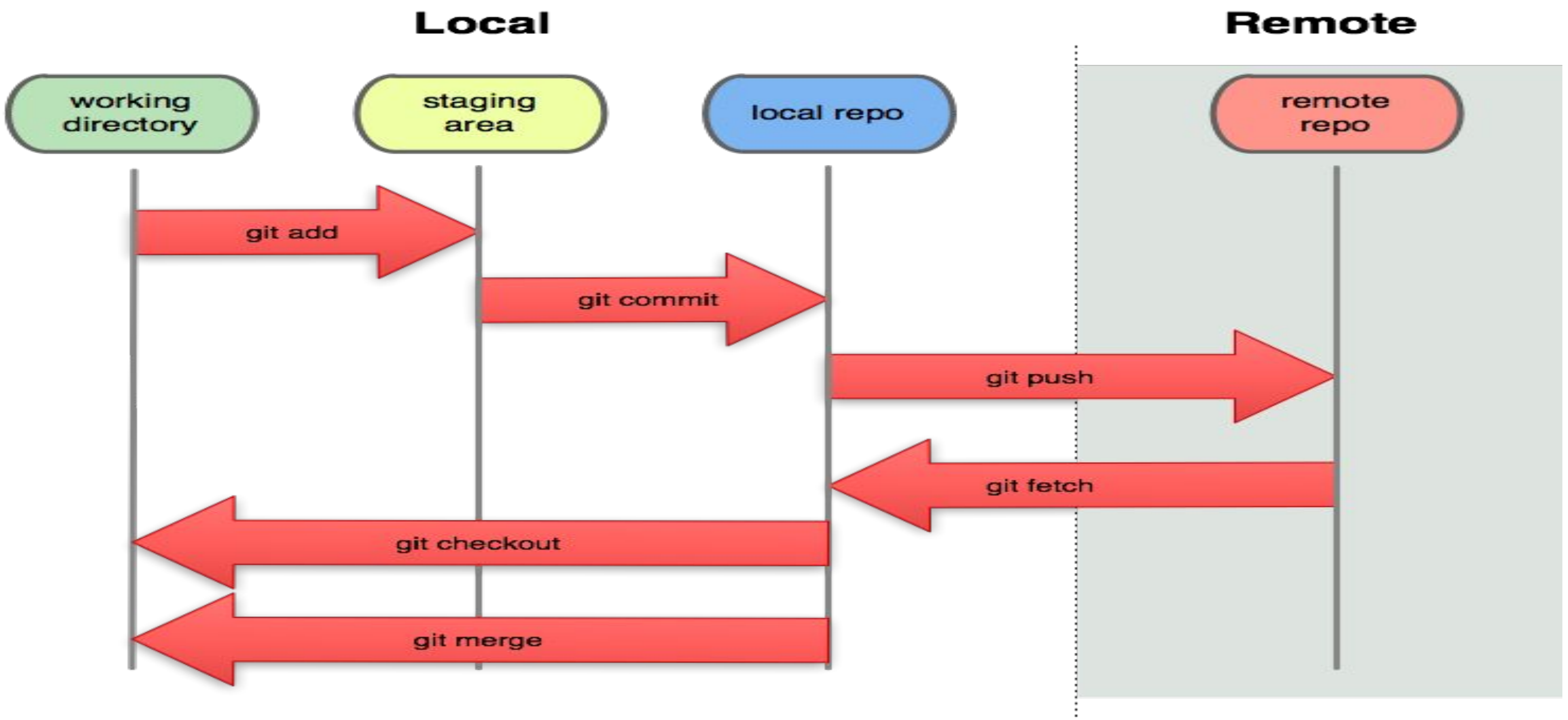
DevOps Course By M. Ali Kahoot - Dice Analytics

Git & Github

From: https://www.slideshare.net/EdurekaIN/what-is-git-what-is-github-git-tutorial-github-tutorial-devops-tutorial-edureka



Stages of Git



Install Git

Ubuntu: $ sudo apt-get install git-all

OR

- Install Git: https://git-scm.com/downloads

Git Repository

- A single project containing the code base of your application/service Can be
  - Private: Only certain people with access can see the repo
  - Public: Anyone can see the repo
- In your local git repository, a .git folder is created, it basically tracks all the changes/versions of your codebase.
- There is a local repository & a remote repository, you make changes to local repository and push them to remote.

LAB - GIT REPOSITORY

- Creating Github Account
- Create & Cloning a Repository
- Add SSH key

CREATING GITHUB ACCOUNT

Go to github.com and Signup, your username will be your github address

E.g. github.com/kahootali

/kahootali

CONFIGURE GIT

Congure your git with your config so that your commits can have this conguration.

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

For pushing changes to Github/Cloning a private repo, it needs to authenticate you, so everytime, you will have to enter your username/pwd. So to ease this, we can add an SSH key of our PC, so that Github knows whenever you are trying to clone/push changes, that this PC is yours and has SSH(secure shell).

Adding SSH Key

Generate SSH key locally

```
ssh-keygen -t rsa -b 4096 -C "your_email@example.com”
eval $(ssh-agent -s)
ssh-add ~/.ssh/id_rsa
cat ~/.ssh/id_rsa.pub
```

Copy the SSH key and then we need to go to the GitHub account and click on Profile -> Settings -> SSH & GPG Keys -> New SSH Key -> Paste.

This will add an SSH key to your account, to test if it is working enter

```
ssh -T git@github.com
```

Creating Git Repo

- Click the + at the top right, New Repository, Enter Repository Name, Description, Chose Public and initialize with a README.

- Your Repository will be created with following link

github.com/<username>/<repo-name>

CLONING A REPOSITORY

Once the repo is created, you can clone it, by clicking the Clone or Download, and copying the link. Go to bash

Using https: Always have to enter username & pwd

```
git clone https://github.com/<user-name>/<repo-name>.git
```

Using ssh: Will use SSH key

git clone git@github.com:<user-name>/<repo-name>.git

Git Commands

• git status:	Tells the changes made in the local repo only
• git diff:	Shows file differences not yet staged
• git add <file>:	Add the file to staging
• git reset <file>:	Remove the file from staging
• git commit:	Commit to local Repository
•git push:	Push the local repository changes to remote repo

Task

Create Github Account

Create Github Repository

Clone it

Make changes

Check different stages of git

Check differences in files

Git Lab

You should have cloned the repo created in previous Lab.

```
cd <repo-name>
ls
```

You will be seeing the README.md le. So We will add a new file in the repo.

```
touch a.txt
```

Edit this file either in vim or any text editor. Now see the status of your repo.

```
git status
```

It will show a file is added that is currently untracked i.e. it is not present in git history

Git Lab

Now add to Staging Area

```
git add a.txt
git status
```

It means file is in staging, but it should be committed to local repo. Now edit the README.md file, add a bit description like "This is a test repo" or anything else.

```
git status
```

It shows that new file a.txt is yet to be committed, and README.md has been modified but it needs to be staged for commit first.

Git Lab

Now see the differences

```
git diff

diff --git a/README.md b/README.md index 5297f8f..2b26b81 100644

--- a/README.md

+++ b/README.md

@@ -1 +1,2 @@

-# test-git

\ No newline at end of file

+# test-git

+This is a test repo
```

It shows that -# test-git was removed and +# test-git, +This is a test repo were added.

Git Lab

Add this file to Staging and then see status

```
git add README.md

git status

On branch master

Your branch is up to date with 'origin/master'.

Changes to be committed:

(use "git reset HEAD <file>..." to unstage)

modified: README.md new file: a.txt
```

Check difference in staging area

```
git diff --staged
```

Git Lab

Commit Change to Local Repo

```
git commit -m "Add a.txt and update README"

git status

On branch main

Your branch is ahead of 'origin/main by 1 commit.

(use "git push" to publish your local commits) nothing to commit, working tree
clean
```

It shows that your local branch is ahead of origin/main by 1 commit that we just did. And the working tree is clean i.e. no other file has been changed or is in staging area

Git Reset

Now we have two changes in a commit. We want to reset these differences. There are 3 possible options.

Git Commands

- git reset(--mixed): Uncommit and move change to working dir
- git reset --soft: Uncommit but file will be in staging
- git reset --hard: Uncommit, remove from stage, remove from local repo

- git reset [commit]: Undoes all commits after [commit], preserving changes locally
- git reset --hard [commit]: Discards all history and changes back to the specific commit

GIT RESET: SOFT

Remove the commit from local repo, but keep the les in staging area

```
git reset --soft HEAD^
git reset --soft HEAD~10
git reset --soft COMMIT_HASH
```

Notice that files are still in staging area but have been removed from local repo, now you will just need to commit them again.

GIT RESET: MIXED

(Default) Remove the commit from local repo, and also remove changes from staging area

```
git reset --mixed HEAD^

git reset HEAD~1

git reset COMMIT_HASH
```

Notice changes have been removed from local repo as well as staging area. They are in working directory. So we will need to add and commit files again.

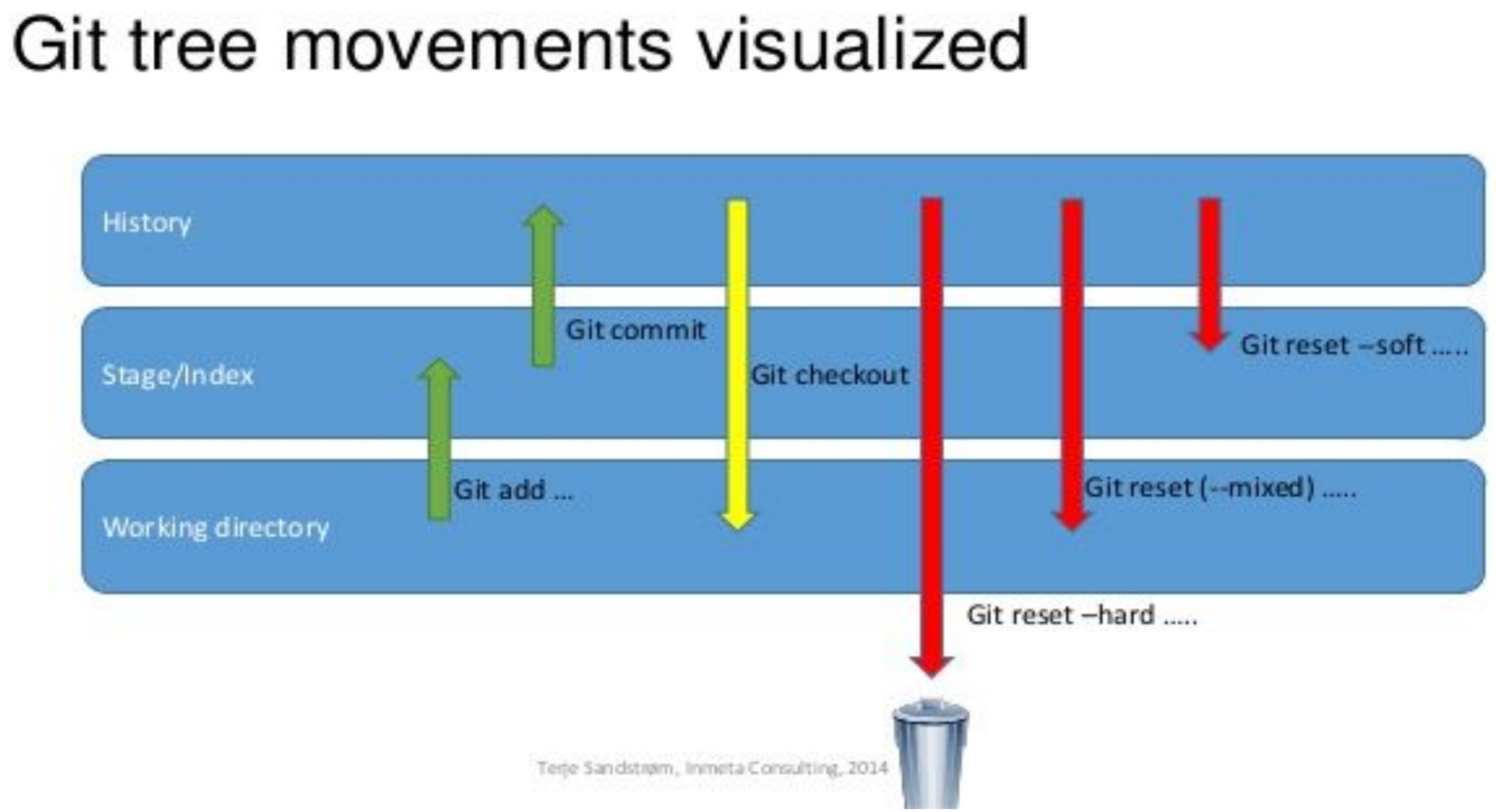
GIT RESET: HARD

Remove the commit from local repo, and staging area and even change the local files.

```
git reset --hard HEAD^
git reset --hard HEAD~1
git reset --hard COMMIT_HASH
```

Notice the changes have been removed from local repo, as well as the working tree i.e. the files are now in initial state that was before your working.

Git Commands



GIT: PUSH & PULL

Push:

- Transfer commits from your local repository to a remote repository
- Share modifications with remote team members
- Commands
  - git push origin

Pull:

- Download content from a remote repository and update the local repository
- Get modification from remote team members
- Commands:
  - git pull origin

LAB: GIT PUSH

- Edit a file or create a new file
- Add to staging area
- Create a Commit to Local Repo
- Push:
    - git push origin main

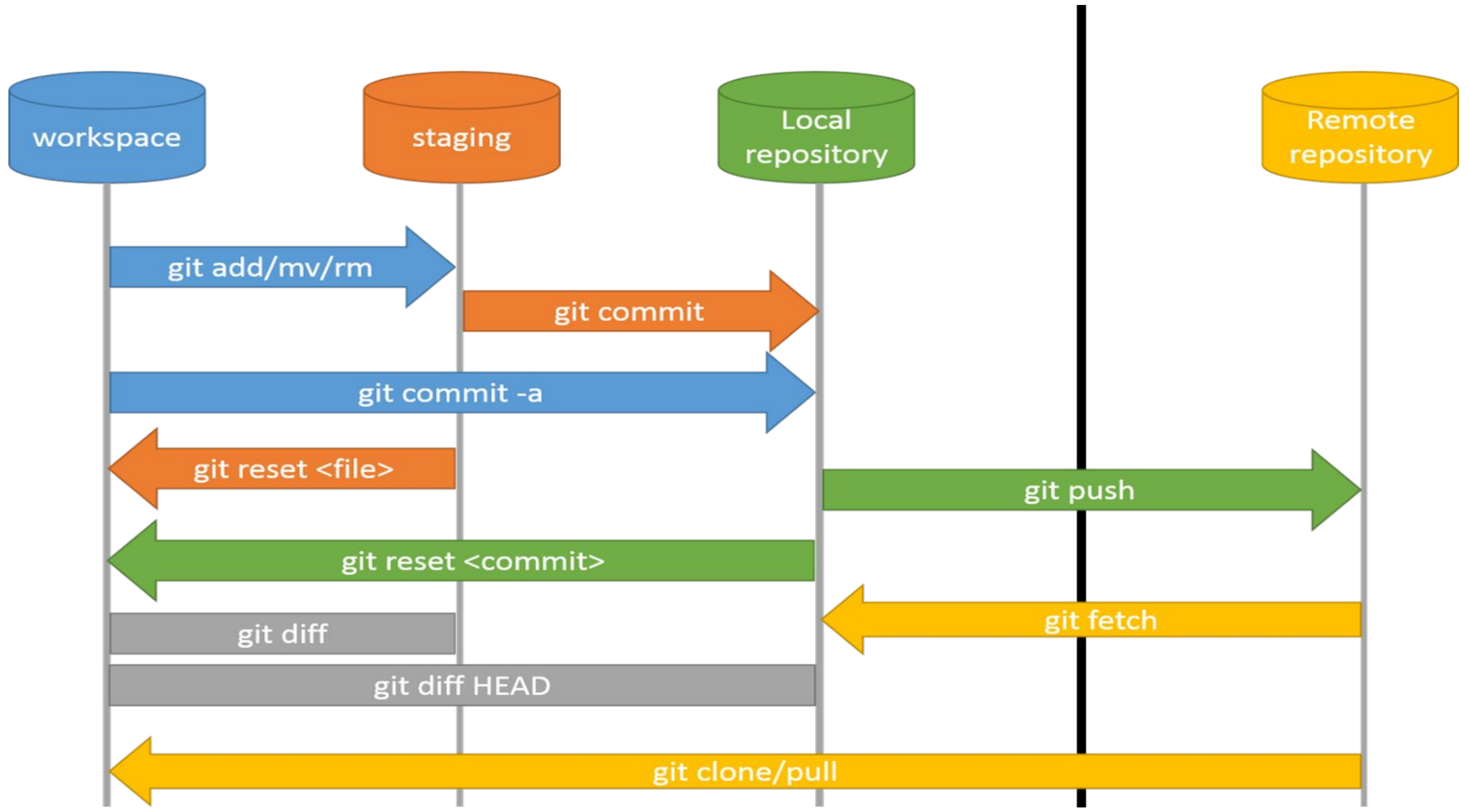
LAB: GIT PULL

- Go to github.com UI and create a new file from UI & Commit from UI
- Check the Remote Commit
- Pull:
    - git pull origin main

List contents of your local repository, you will see the newly created file

/kahootali

From: https://medium.com/@mehulgala77/github-fundamentals-clone-fetch-push-pull-fork-16d79bb16b79



GITHUB PAGES

- Static site hosting service
- Host your
    - Project website
    - Organization website
    - Personal website
- Does not support server side code such as PHP, Python or Ruby
- GitHub Pages are free for public repos, for private repos you would need to buy subscription

Things to do before next class

- Install Docker on the Ubuntu system, you can search how to install Docker on Ubuntu, it's really simple, to verify, you must be able to run the command "docker info"
- Complete all the labs in git slides
- Learn about
  - API Calls: https://apifriends.com/api-management/whats-api-call/
  - Bash Shell scripting: https://s3cloudhub.medium.com/shell-scripting-crash-course-d3f8ffe8c6ea

/kahootali

Assignment 1

- Optional: Host your profile or Resume through Github Pages
  https://dev.to/joojodontoh/host-your-resume-on-github-pages-514g

- Assignment can be seen at LMS

/kahootali

Assignment 1

Answer the following questions in a Readme of a Git Repo and each answer should be in a separate commit, and host this repo on Github Pages

- Q1) Agile vs DevOps
- Q2) Define CI, Continuous Delivery & Continuous Deployment
- Q3) What are the benefits of Cloud Computing
- Q4) Difference b/w Git & Github
- Q5) Stages of Git
- Q6) 3 methods of git reset