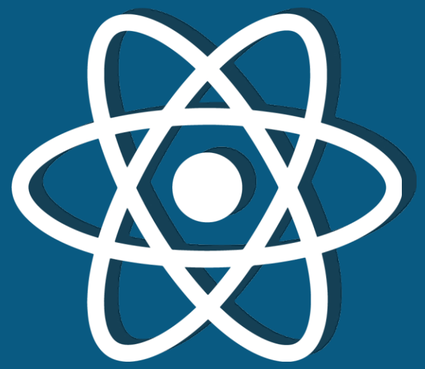


React With Redux Certification Training

COURSE OUTLINE
MODULE 08

1. Introduction to Web Development and React
2. Components and Styling the Application Layout
3. Handling Navigation with Routes
4. React State Management using Redux
5. Asynchronous Programming with Saga Middleware

6. React Hooks
7. Fetching Data using GraphQL
8. **React Application Testing and Deployment**
9. Introduction to React Native
10. Building React Native Applications with APIs

Topics

Following are the topics covered in this module:

- Testing
- Why should we perform testing?
- Types of testing
- Testing React application using Jest and Enzyme
- Installation of Jest and Enzyme
- Maintaining application code using Git
- Version control system
- Why should we use Git?
- Git file workflow
- Git commands
- Running application on production server: Nginx
- Architecture of Nginx
- How to configure Nginx?
- Deployment of an application using Docker
- Problems before containers
- How containers solve the problems
- What is Docker?
- Docker file
- Docker image
- Docker containers
- Docker hub
- Basic Docker commands

Objectives

After completion of this module you should be able to:

- Setup testing environment using Jest and Enzyme
- Add Snapshot testing
- Integrate Test Reducers
- Push React application code to GitHub repository
- Run React application on production server: Nginx
- Build Docker image of the React application

Testing

Software Testing

Software Testing is a process, to evaluate the functionality of a software application, with an intent to find whether the developed software met the specified requirements or not, and to produce a bug free quality product.

Why Should We Perform Testing?

01 ***Cost*** of ***fixing the bugs*** is larger if testing is not done in early stage

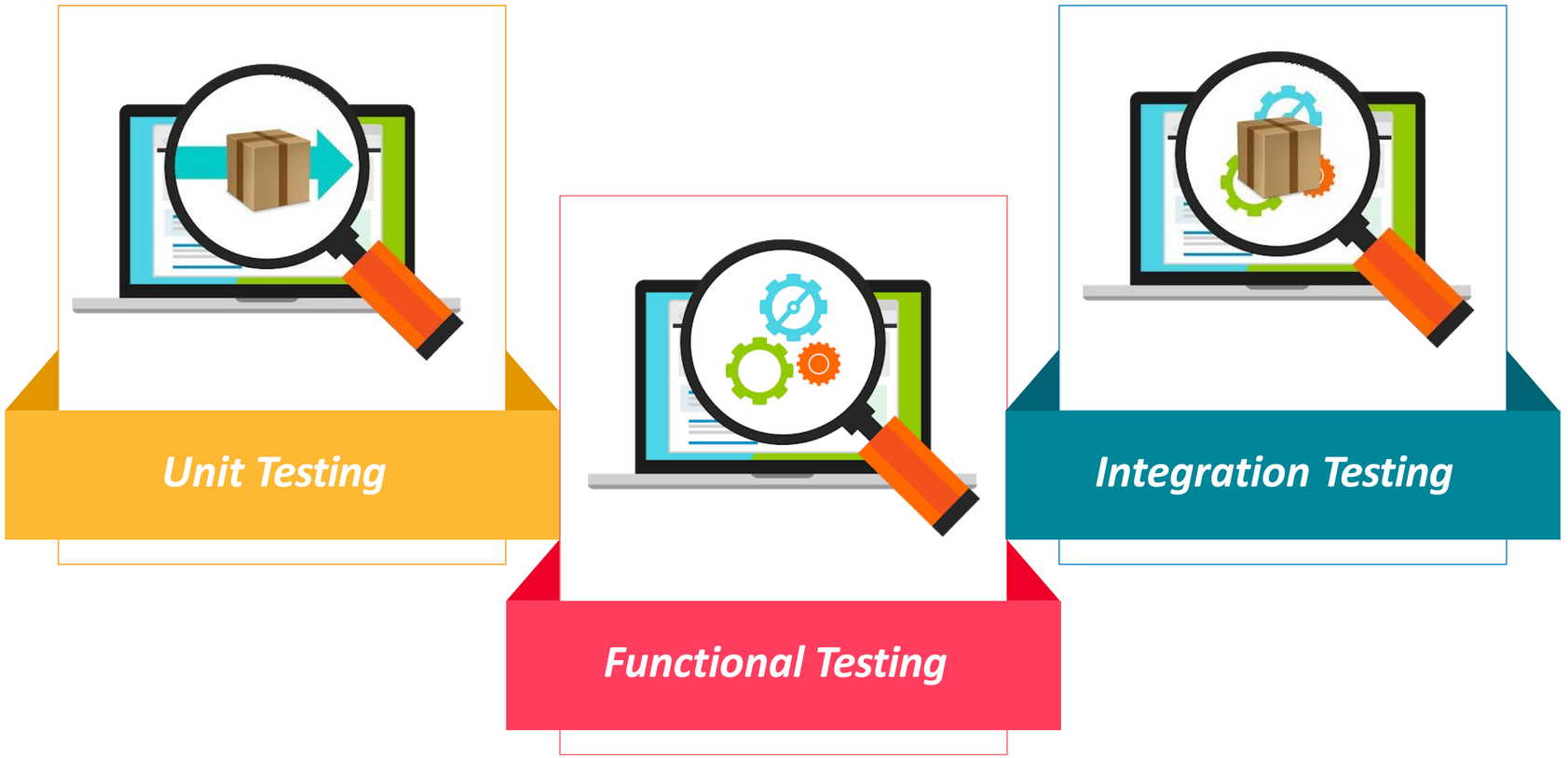
02 To produce ***good quality*** products

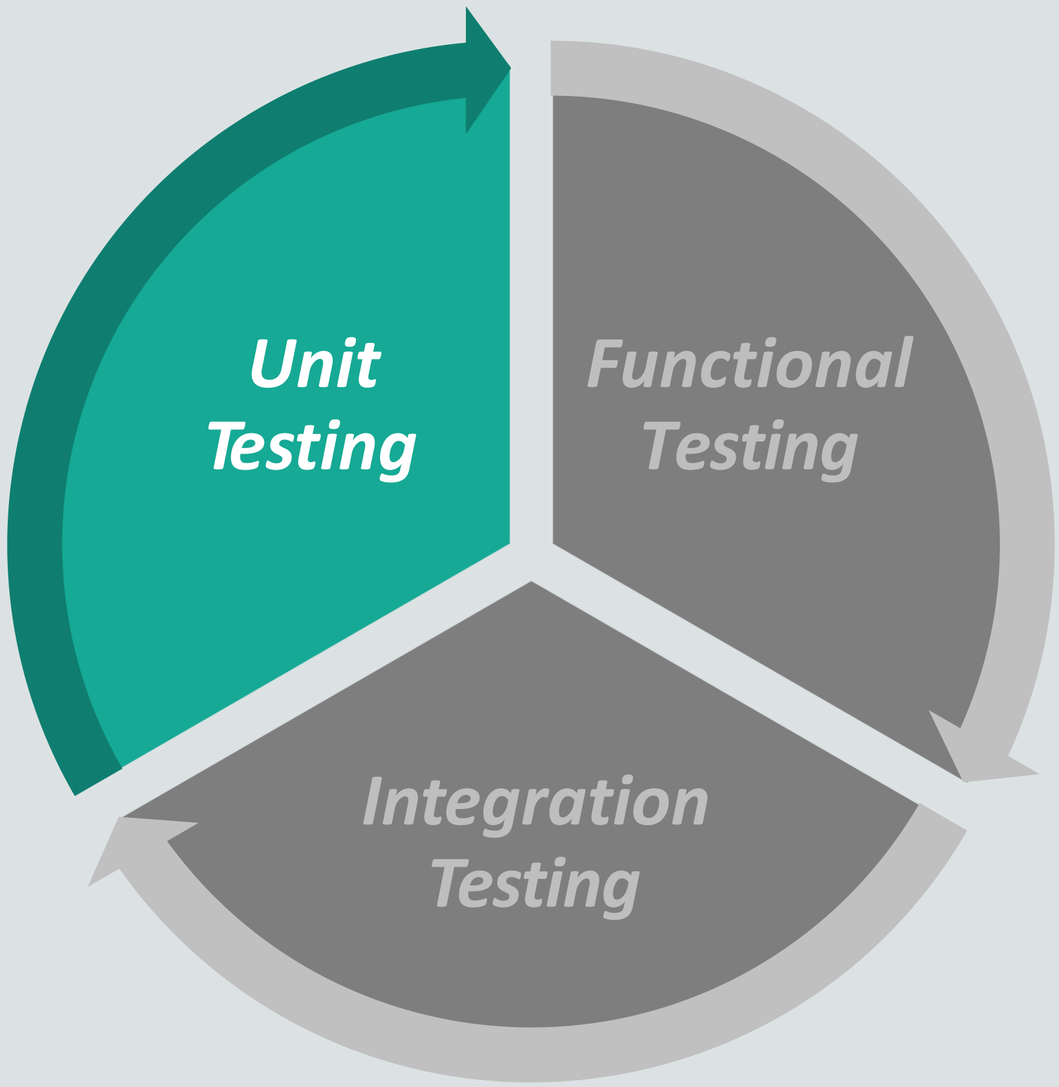
03 To make software application ***defect free***

04 To ***check*** if ***behaviour of an application*** is same in development and production environment

Types Of Testing

The different types of testing are:




Unit Testing
Functional Testing
Integration Testing

Unit Testing

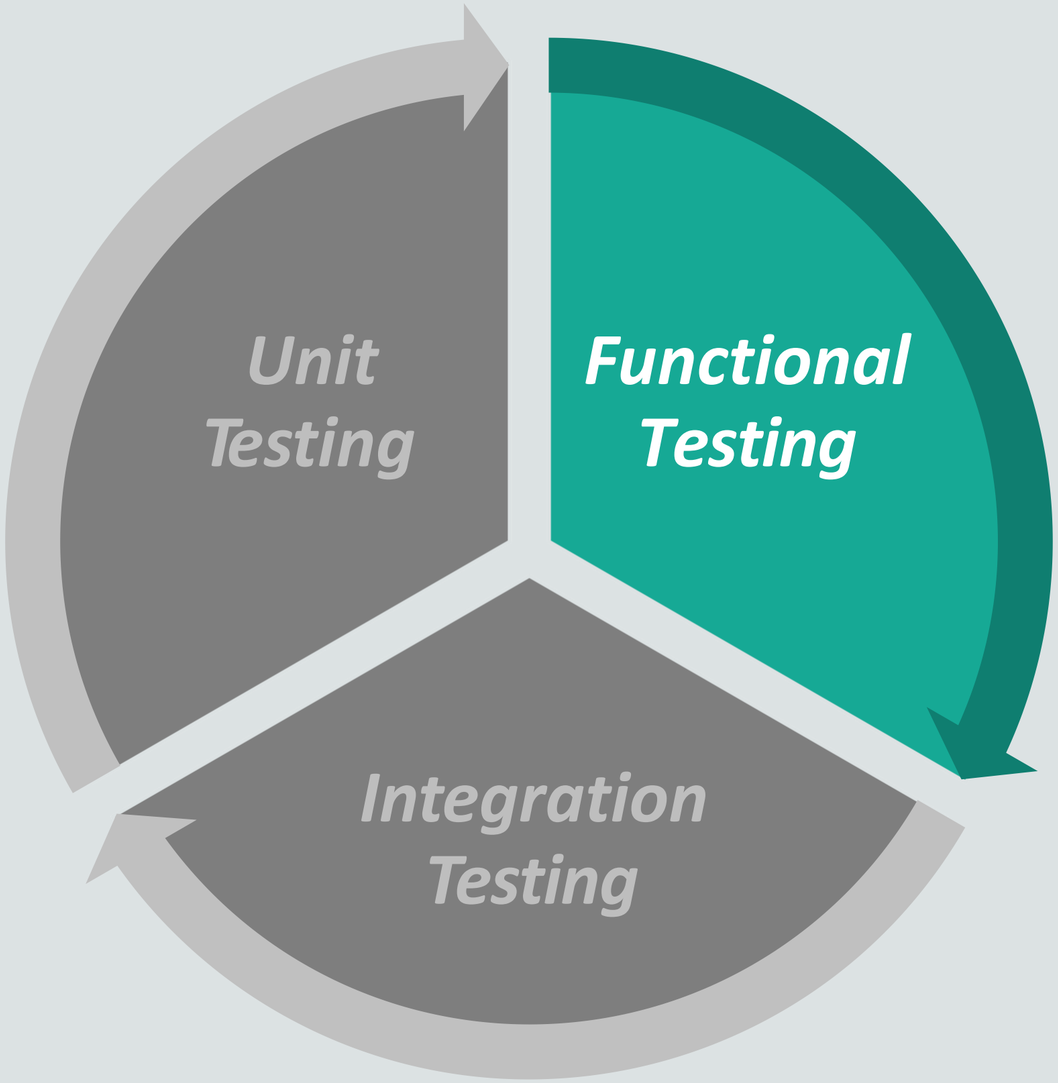
01 ***Unit testing*** is used to test the ***small pieces*** of an independent code, unconcerned with how they are wrapped into your application

02 Used to ***prevent regressions*** (bugs that occur repeatedly)

03 Easy to ***pinpoint*** and ***fix*** the ***problem***

04 If it takes ***longer time to execute***, it is likely that the code ***under test*** is much more complex than it should be

05 Popular tool to perform unit testing on React applications is ***Jest***


Unit Testing
Functional Testing
Integration Testing

Functional Testing

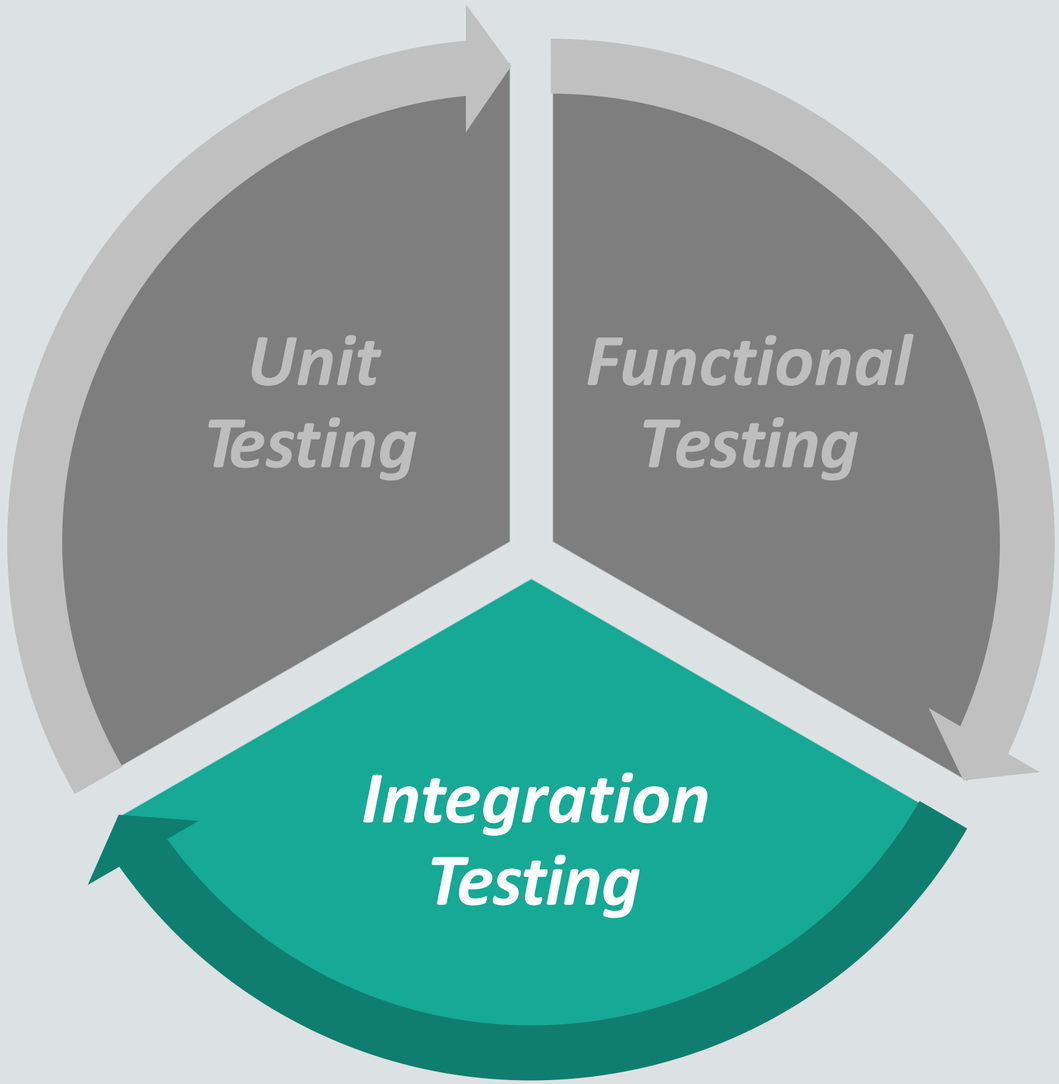
01 ***Functional tests*** are made against combinations of many units

02 You can use ***any number of external objects*** or ***systems*** like database, UI, security, and other application layers

03 ***Takes longer time*** to execute than ***unit test***

04 ***Functional Test*** represents major releases whereas ***unit test represents only minor changes***, due to which they are ***expected*** to change less often than unit test

05 Eg: When user enters ***username, password and clicks on send***, that user will be logged in. We can easily see that this functional group will ***comprise*** of many unit tests, one for validating a username, one for handling a button click and so on


Unit Testing
Functional Testing
Integration Testing

Integration Testing

01 ***Integration testing*** ensures that the entire system application is working correctly

02 It can be executed only within a ***realistic environment*** like a real databases, servers, and other systems which are like the target production environment

03 It is used ***incase*** you need to test separate ***systems*** like ***database and application***

04 It is ***slower*** than other tests as it is ***complex*** to write

05 ***For Example***: A functional test may successfully test the ability of an ideal system to open a help dialog box, but when integrated with a new browser or other runtime, it is found that the expected functionality is not achieved

Testing React Applications Using Jest And Enzyme

Jest And Enzyme

Jest** is a fast testing framework. It acts as a **test runner**, **assertion library**, and **mocking library

***Enzyme** is a JavaScript testing utility for React that makes it easier to assert, manipulate, and traverse your React components output*

Enzyme provides** additional utility methods for **rendering a component** (or multiple components), **finding elements**, and **interacting with elements

***create-react-application** setup comes bundled with **Jest**, so no need to install it separately but enzyme must be installed*

Installation Of Jest And Enzyme

If not using create-react-application install Jest:
npm install --save-dev jest babel-jest

Install Enzyme:
npm install --save-dev enzyme enzyme-adapter-react-16 enzyme-to-json

Update your package.json :
"jest": {"snapshotSerializers": ["enzyme-to-json/serializer"]}

Note:

- ***enzyme-to-json*** provides a better component format for snapshot comparison
- Without the serializer, each time the component created in a test must have the enzyme-to-json method ***.toJson()*** used individually before it can be passed to Jest's snapshot matcher. With the serializer you never use it individually

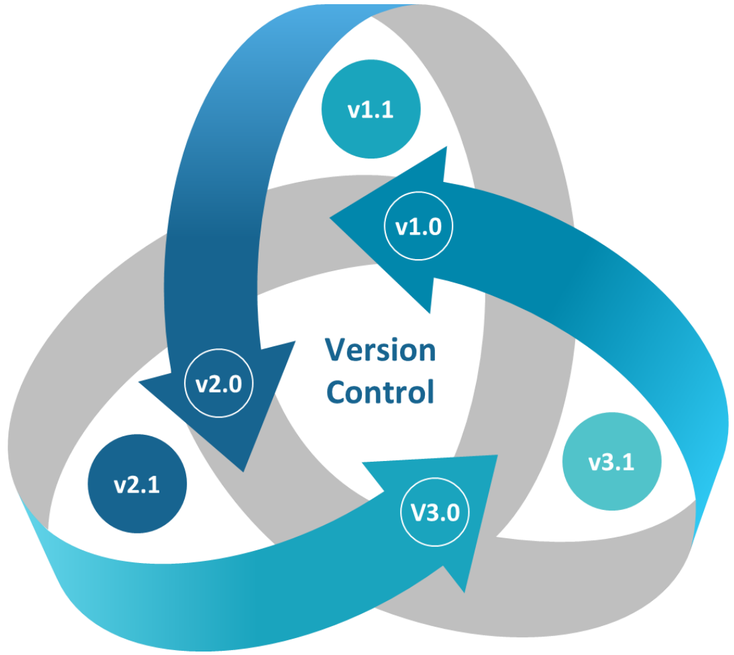
Demo 1: Snapshot Testing And Integrating Testing Reducers

Maintaining Application Code Using Git

What Is Version Control System?

Version Control System is the one that records changes of documents, computer programs, large websites and other collection of information over time.

- It allows multiple users to manage multiple revisions of the same unit of information
- It is used to ***maintain*** the ***records*** of different versions of code
- The most popular tool used as version control system is ***Git***



Why Should We Use Git?

Git is an open source Distributed Version Control System(DVCS) for tracking changes in source code during software development.

Snapshots: Git records changes made to a file rather than file itself. That means if a file is not changed it is not stored again

Distributed: Every user has his own copy of the repository which stores data locally

Fast Operations: Almost every operation on Git is local, hence the speed offered by Git is lightening fast compared to other VCS's

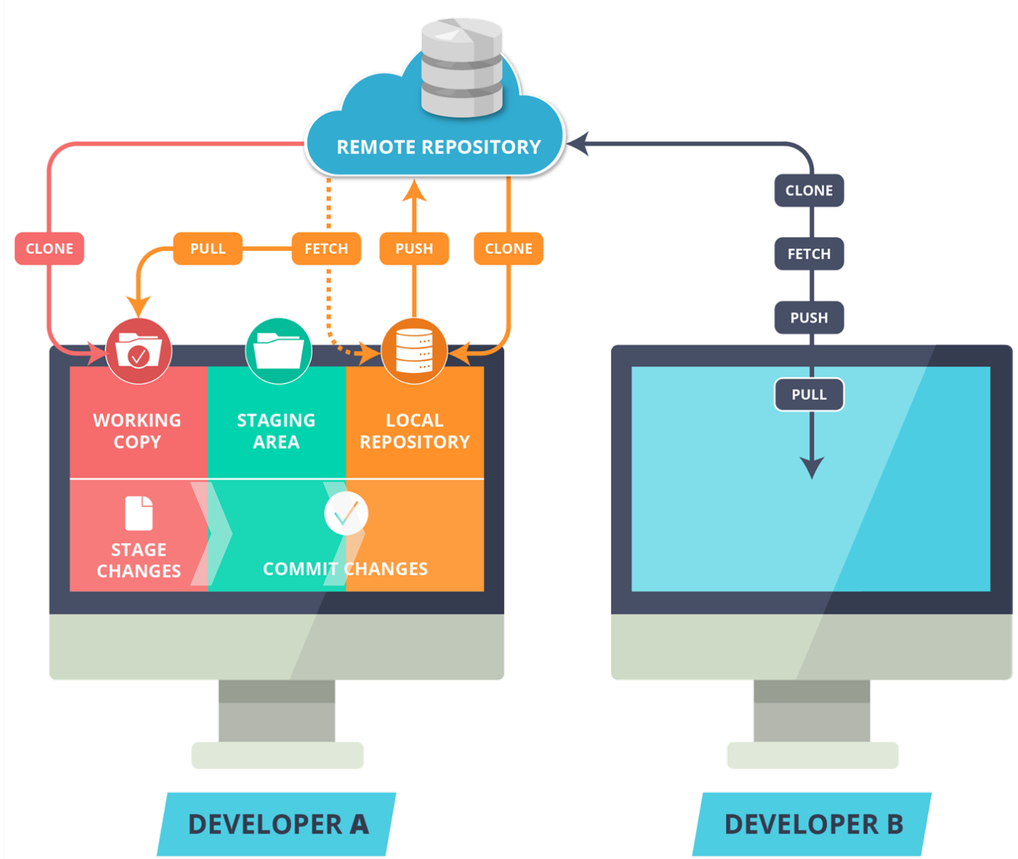
Branch Handling: Every collaborator's working directory is in a separate branch. Different branches can be merged with ease

Robust: Nearly every task in Git is recorded hence it is hard to loose any change or data in Git

The Git File Workflow

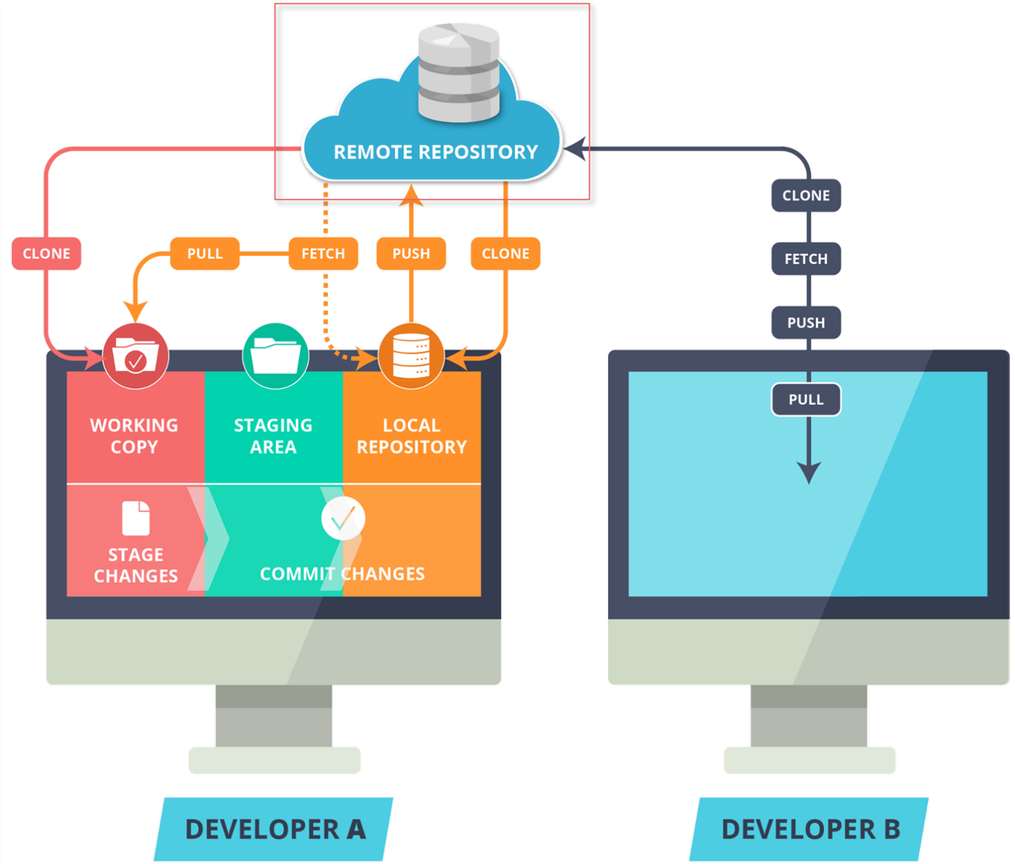
The Git File Workflow

- Use Git workflow to manage your project effectively
- Working with set of guidelines increases Git’s consistency and productivity



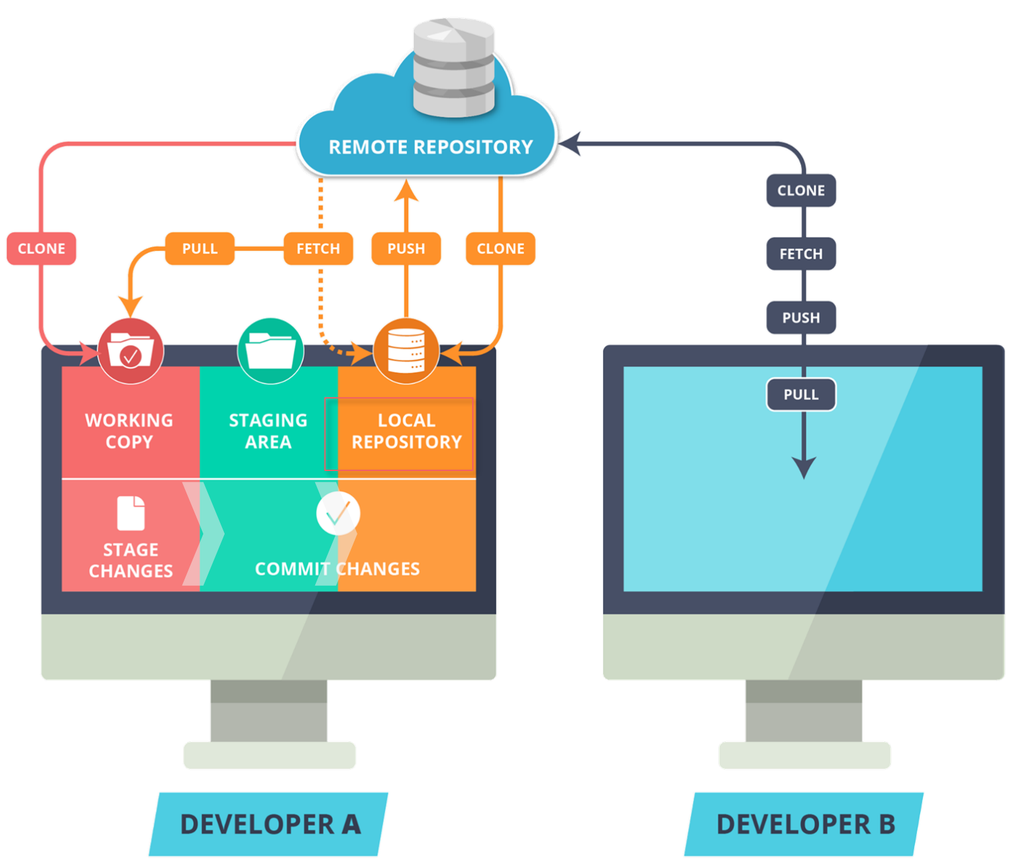
The Git File Workflow: Remote Repository

The ***remote repository*** is the server where all the collaborators upload changes made to the files.



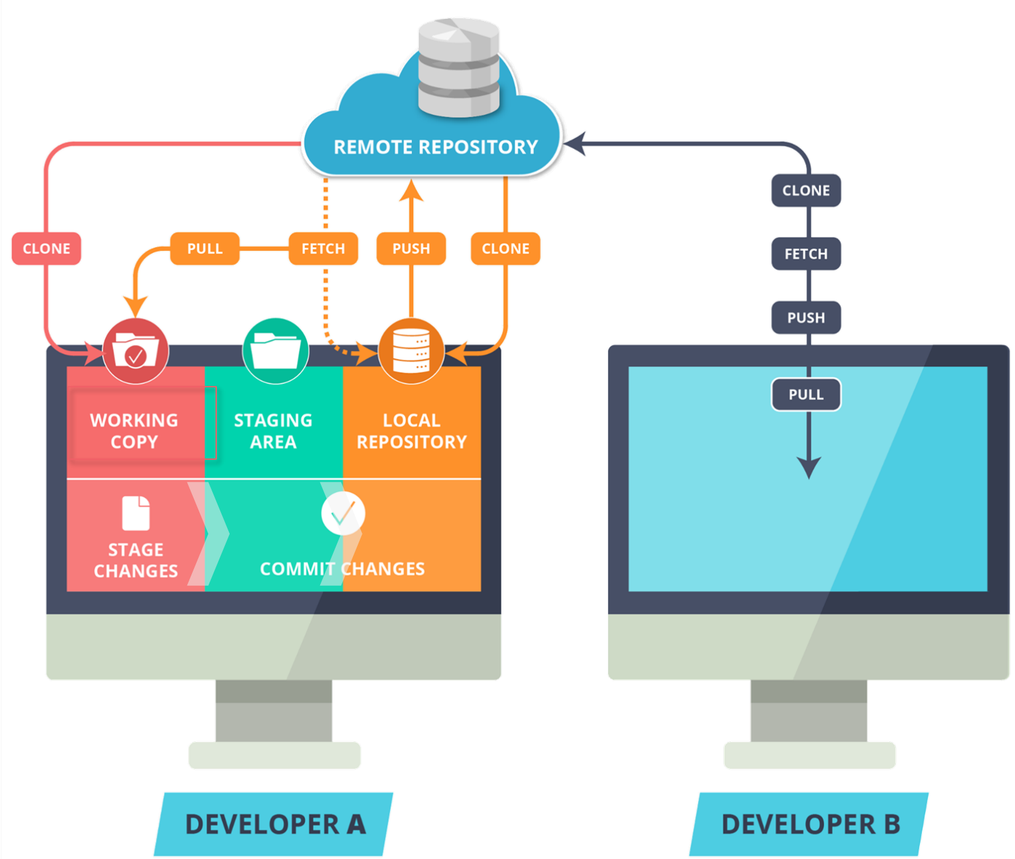
The Git File Workflow: Local Repository

Local Repository is user's copy of the Database. The user accesses all the files through local repository and then push the changes made, to the "remote repository".



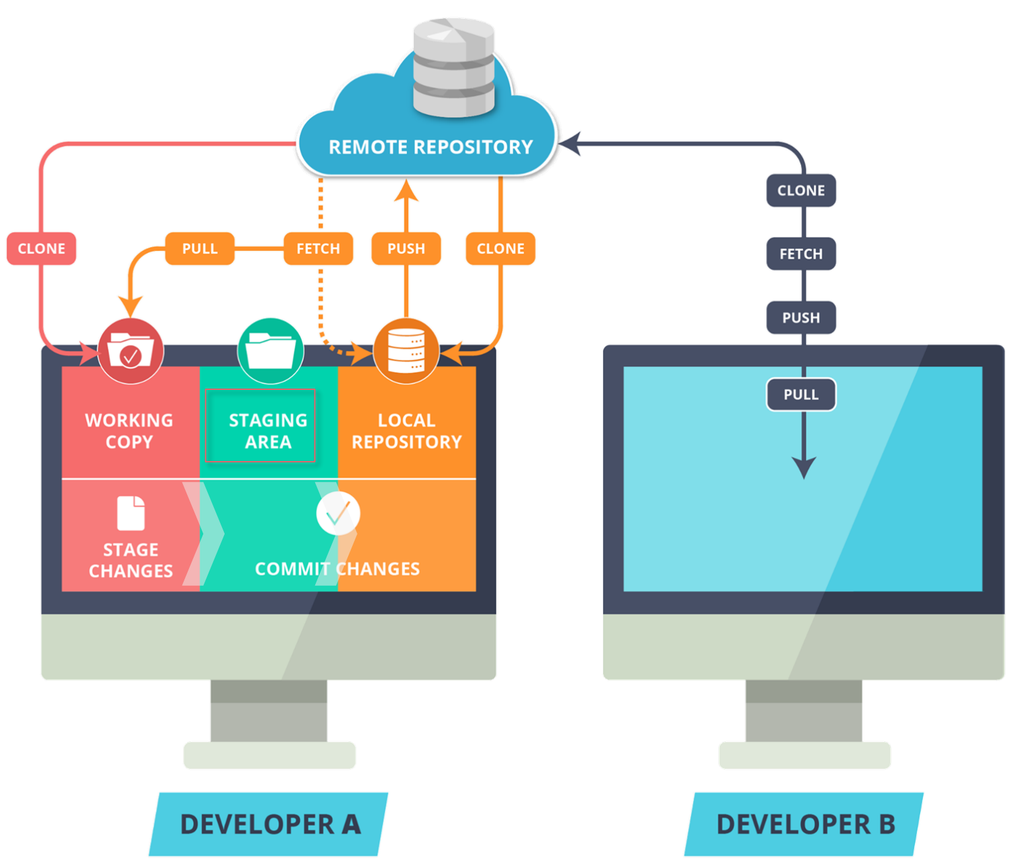
The Git File Workflow: Workspace

Workspace is user's active directory. The user modifies existing files and creates new files in this space. Git tracks these changes done in your local repository.



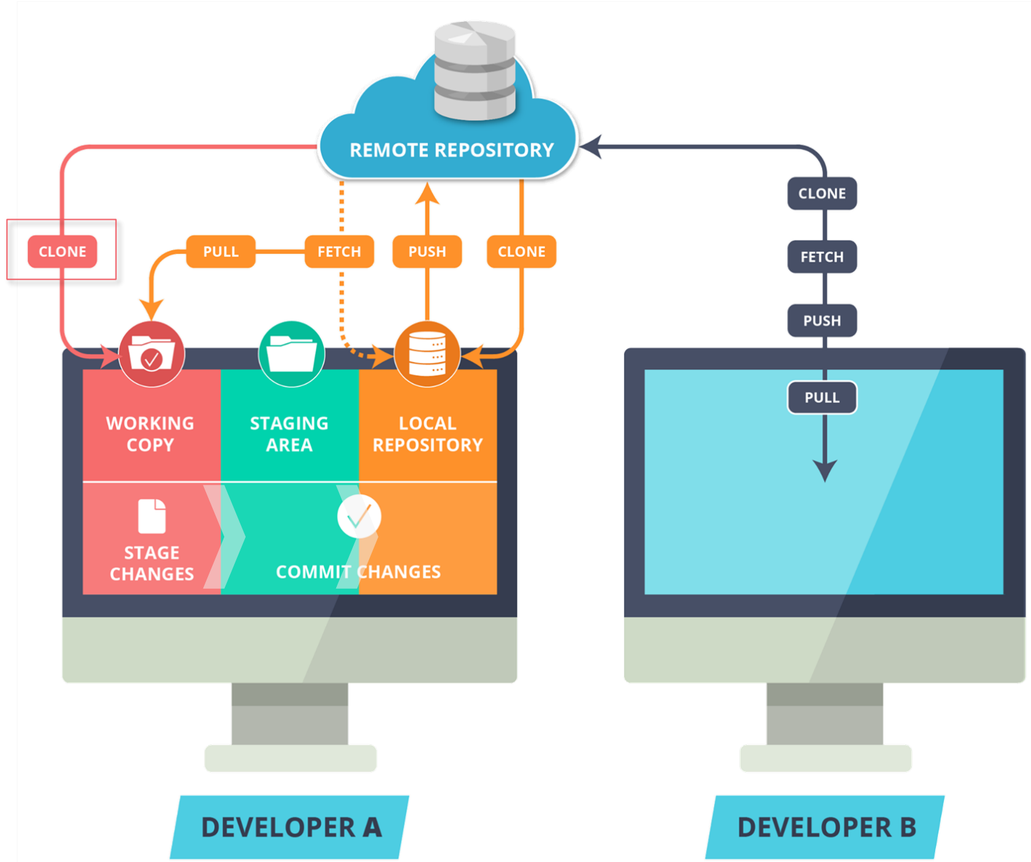
The Git File Workflow: Stage

Stage is a place where all the modified files marked to be committed are placed.



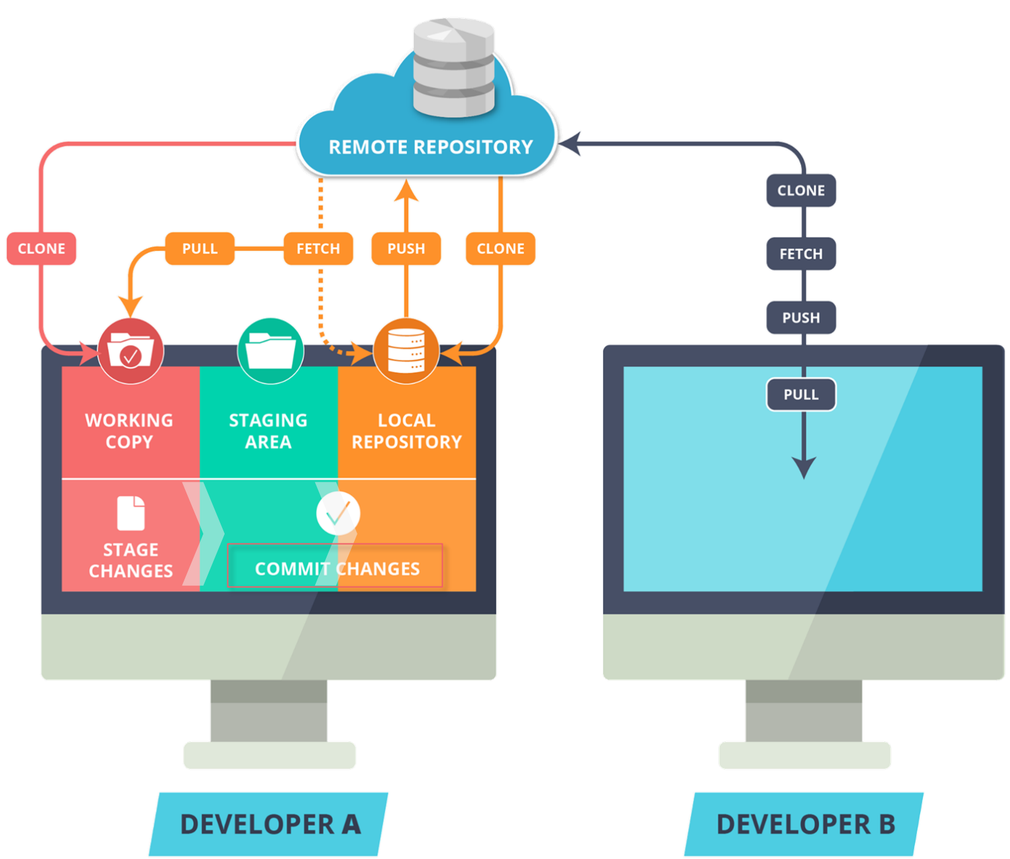
The Git File Workflow: Clone

Clone creates a copy of an existing remote repository inside the local repository.



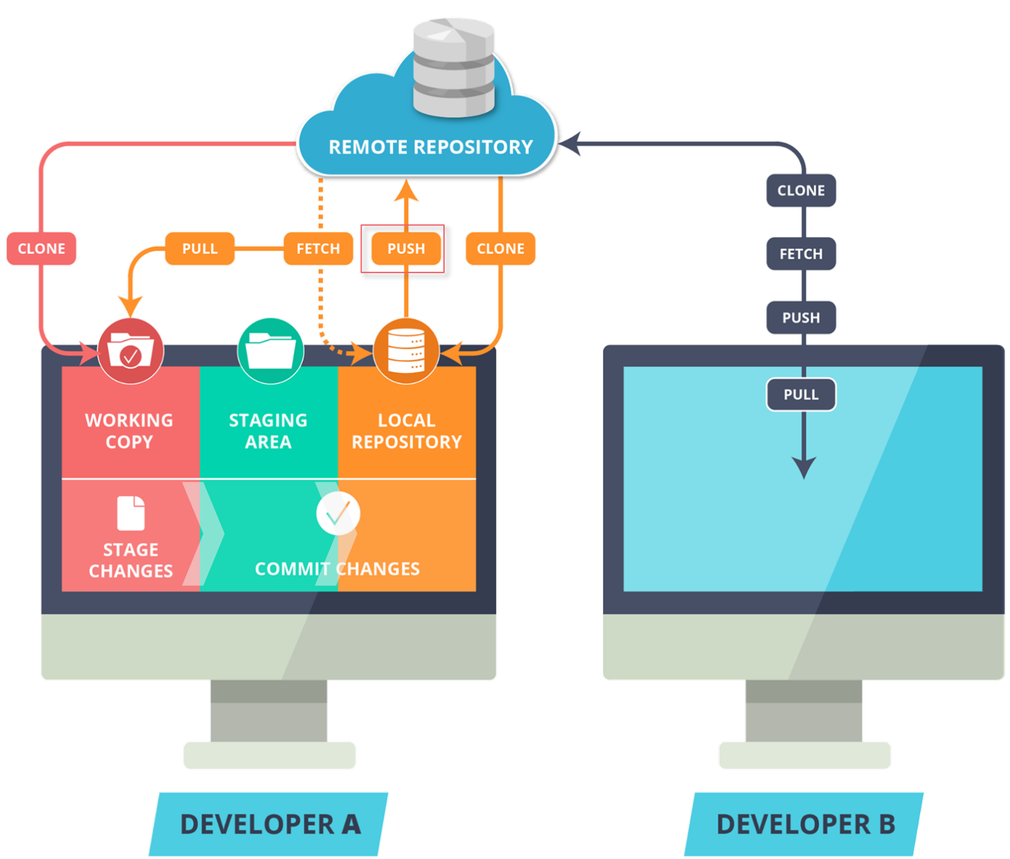
The Git File Workflow: Commit

Commit command commits all the files in the staging area to the local repository.



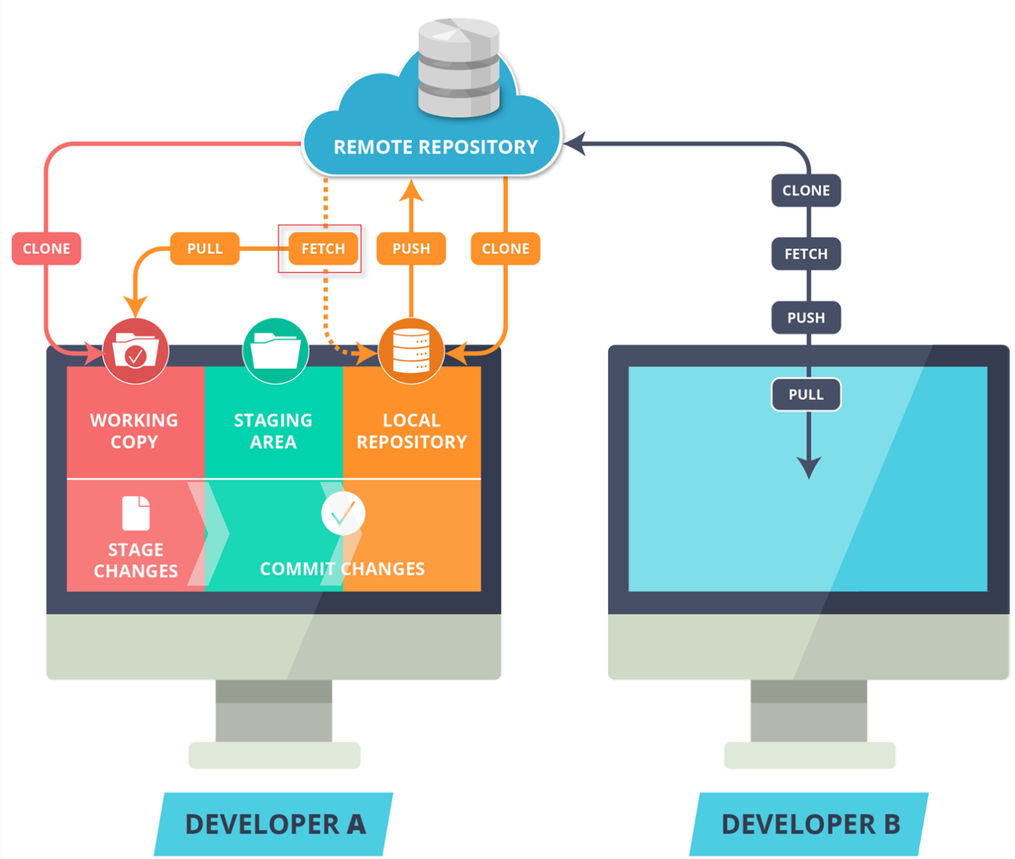
The Git File Workflow: Push

Push command moves all the changes made in the Local Repository to the remote repository.



The Git File Workflow: Fetch

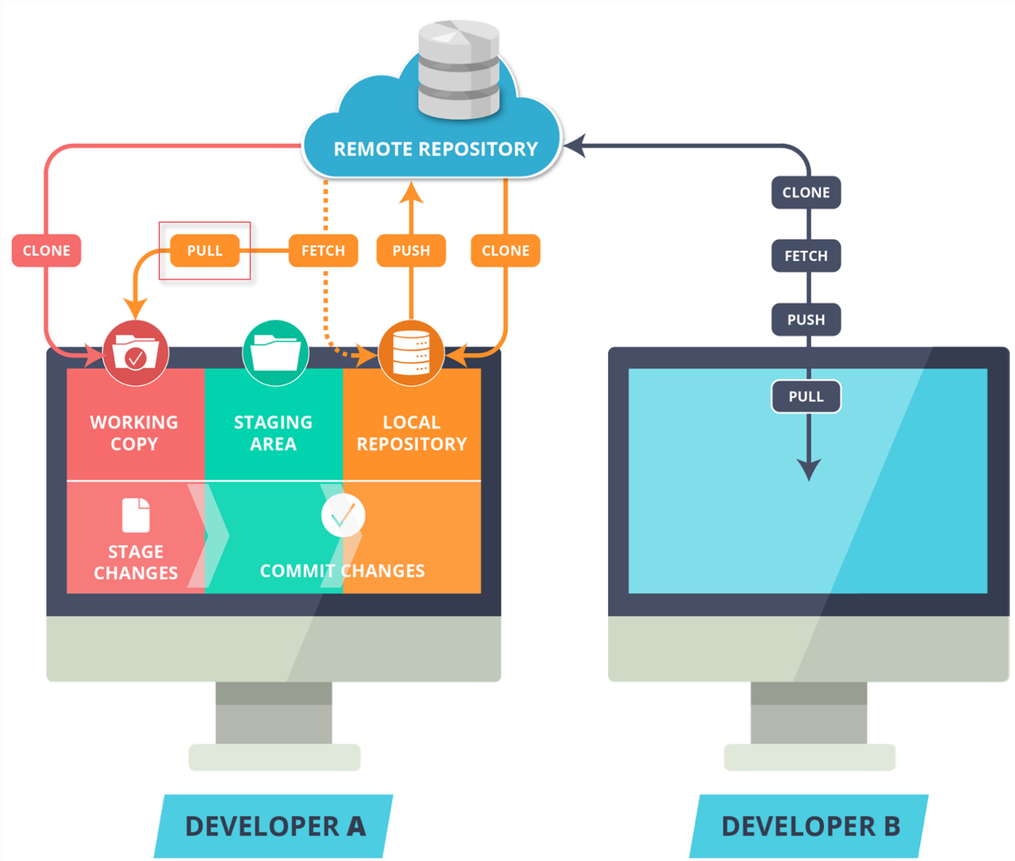
Fetch collects the changes made in the Remote repository and copies them to the local repository. This command doesn't affect our workspace.



The Git File Workflow: Pull

Pull gets all the changes from the remote repository and copies them to the local repository.

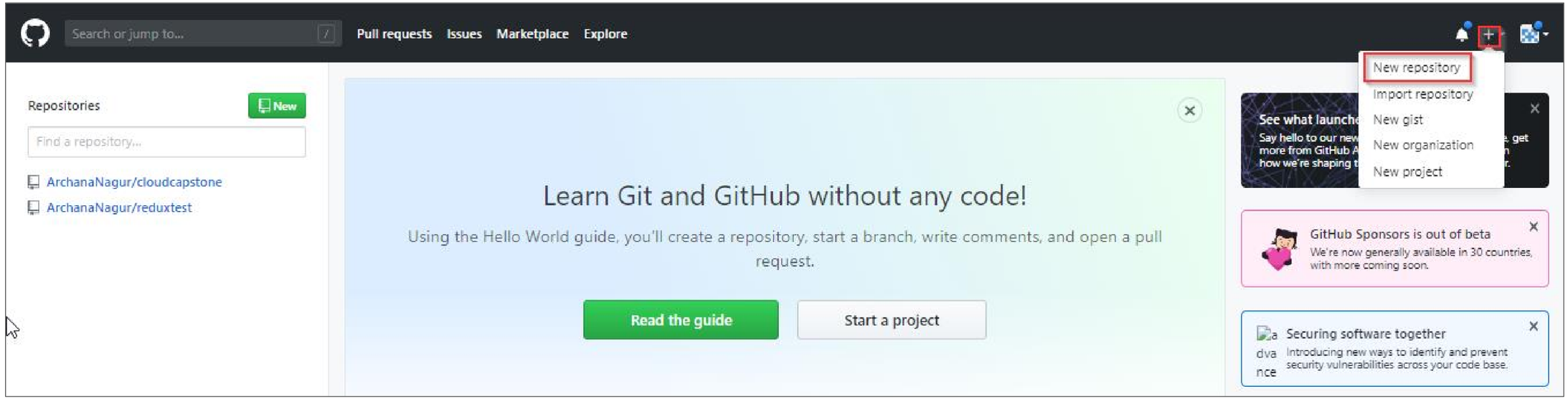
Pull merges those changes to the current working directory.



Demo 2: How To Push Code To GitHub

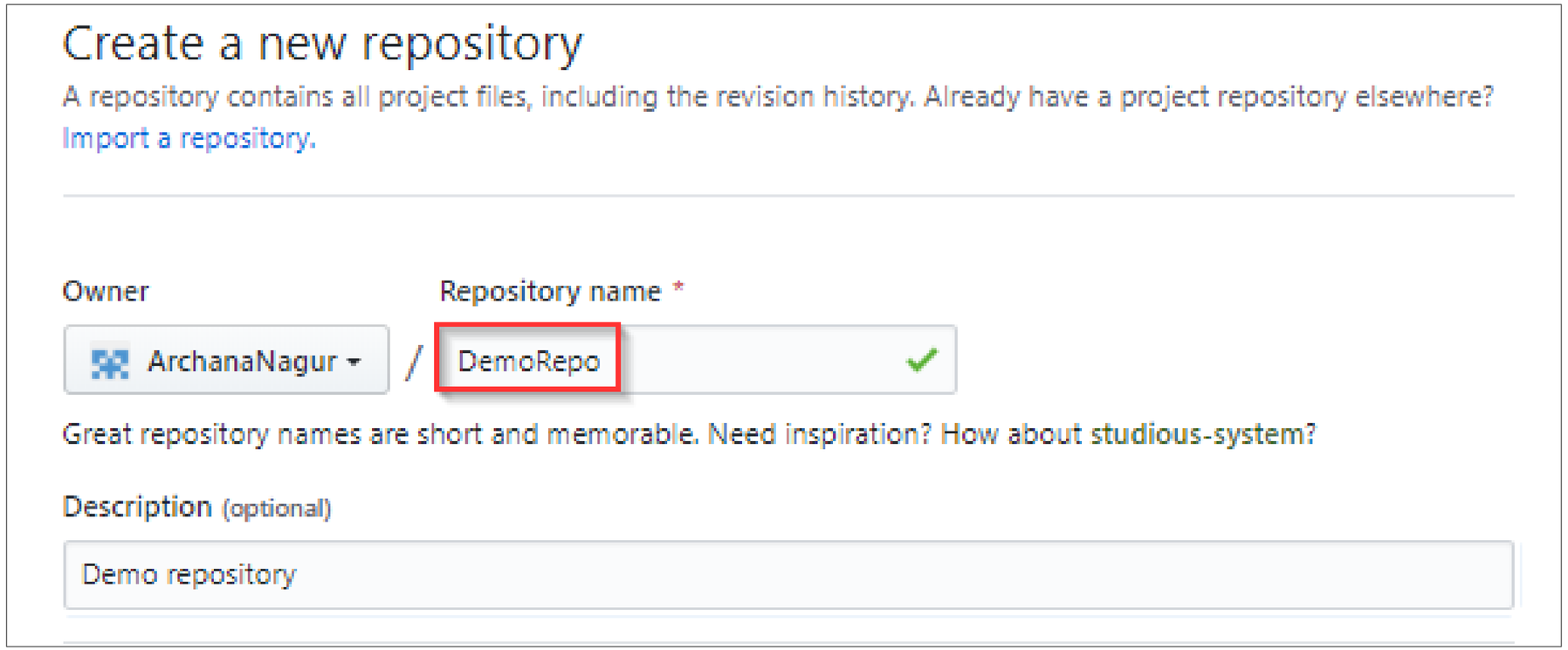
Creating A Remote Repository: New Repository

- Sign-up at github.com
- Click on New repository to create a new repository



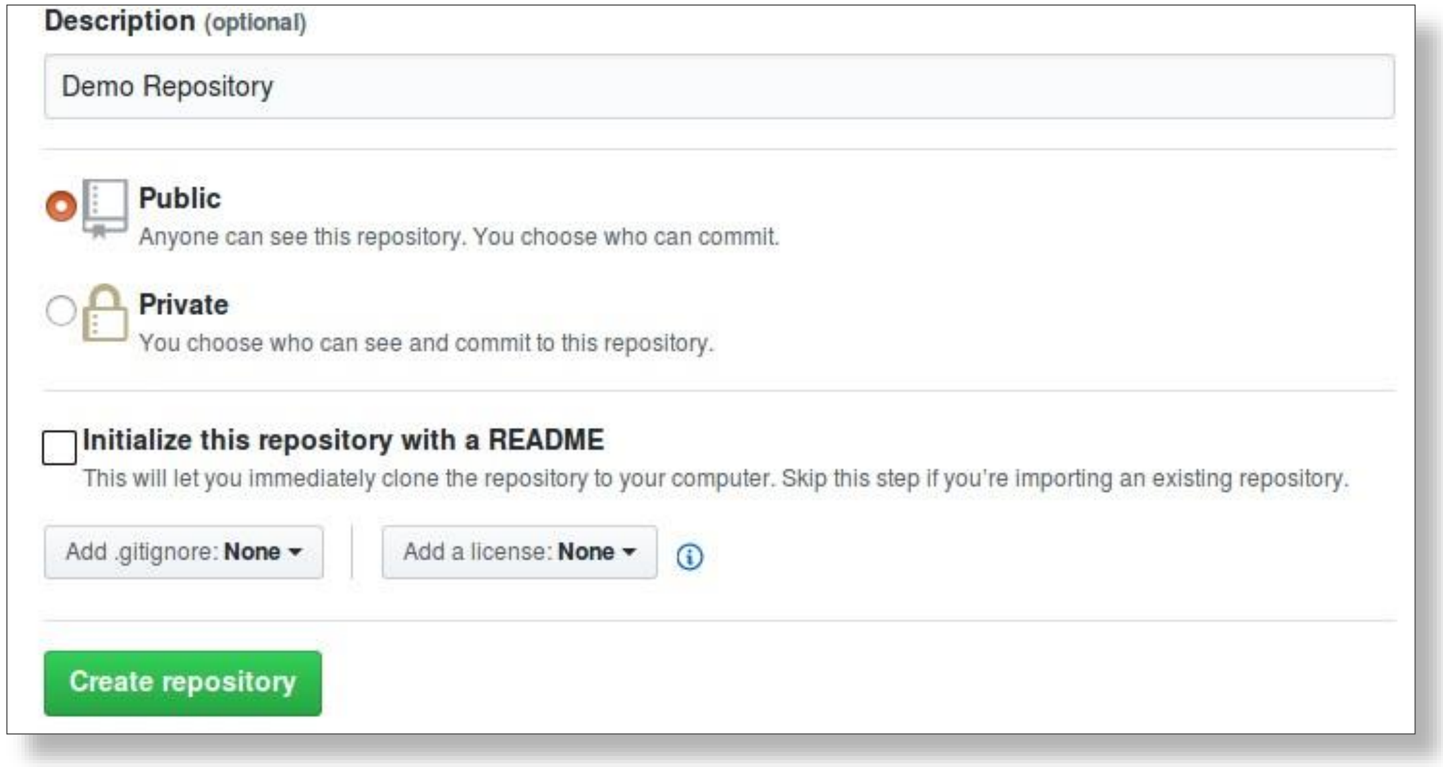
Creating A Remote Repository: Adding Description

- Under **Repository name**, give a name to your repository
- Give some Description about your repository under **Description** section



Creating A Remote Repository: Create Repository

- For a free repository choose ***Public***
- For a private repository you must pay the specified amount
- Finally click on ***Create Repository***




To push code to GitHub repository, open any application code and install Git in your system.

Install Git

- Download Git from https://git-scm.com/download
- Run it in the folder which you want to push to Git
- **Initialize Git** in that folder:

```
git init
```

```
PS C:\Users\archana\Desktop\React\ReactJSDemo\musicshop> git init
Initialized empty Git repository in C:/Users/archana/Desktop/React/ReactJSDemo/musicshop/.git/
```

- Add permissions to your file locally:

```
git config --local user.name “User_name“
git config --local user.email
“email@gmail.com"
```

```
PS C:\Users\archana\Desktop\React\ReactJSDemo\musicshop> git config --local user.name "ArchanaNagur"
PS C:\Users\archana\Desktop\React\ReactJSDemo\musicshop> git config --local user.email "[redacted]@gmail.com"
```

Work With Git

- To add remote repository to local repository, use **git add remote** followed by remote link

```
git remote add origin
https://github.com/user_name/repo_name.git
```

```
PS C:\Users\archana\Desktop\React\ReactJSDemo\musicshop> git remote add origin https://github.com/ArchanaNagur/DemoRepo.git
```

- Add files to your repository:

```
git add.
```

```
git commit -m 'first commit'
```

```
PS C:\Users\archana\Desktop\React\ReactJSDemo\musicshop> git commit -m 'first commit'
[master (root-commit) 17582d6] first commit
```

Push And Pull From Your Git Repository

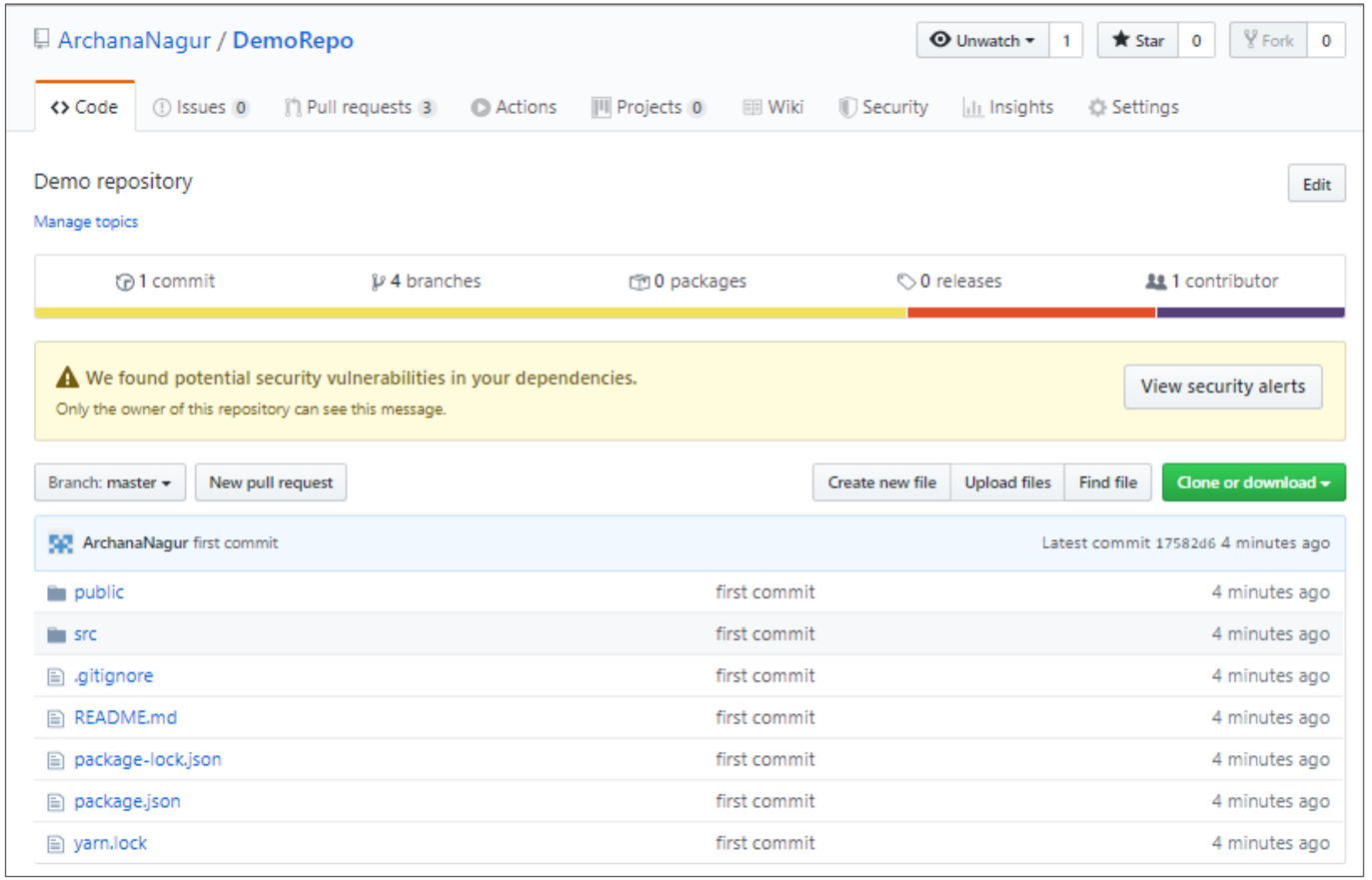
- Push to your Git repository:

```
git push origin master
```

```
PS C:\Users\archana\Desktop\React\ReactJSDemo\musicshop> git push origin master
Enumerating objects: 23, done.
Counting objects: 100% (23/23), done.
Delta compression using up to 4 threads
Compressing objects: 100% (23/23), done.
Writing objects: 100% (23/23), 316.35 KiB | 3.48 MiB/s, done.
Total 23 (delta 0), reused 0 (delta 0)
To https://github.com/ArchanaNagur/DemoRepo.git
 * [new branch]      master -> master
```

Here ***master*** refers to master branch in local repository.

Check Application Code In GitHub Repository



Additional GIT Commands

- To list all the remotes attached to your local repository

```
Syntax: git remote -v
```

```
PS C:\Users\archana\Desktop\React\ReactJSDemo\musicshop> git remote -v
origin  https://github.com/ArchanaNagur/DemoRepo.git (fetch)
origin  https://github.com/ArchanaNagur/DemoRepo.git (push)
```

- ***Fetch*** command copies the changes from remote to local repository

```
Syntax: git fetch origin
```

- ***Pull*** copies all the changes from remote to local repository, It then merges the changes with the present working directory

```
Syntax: git pull origin
```

- The ***git log*** command shows all the commits so far on the current branch

```
Syntax: git log
```

Removing Files From Repository

- The **git rm** command deletes the file from git repository as well as users system

```
Syntax: git rm <filename>
```

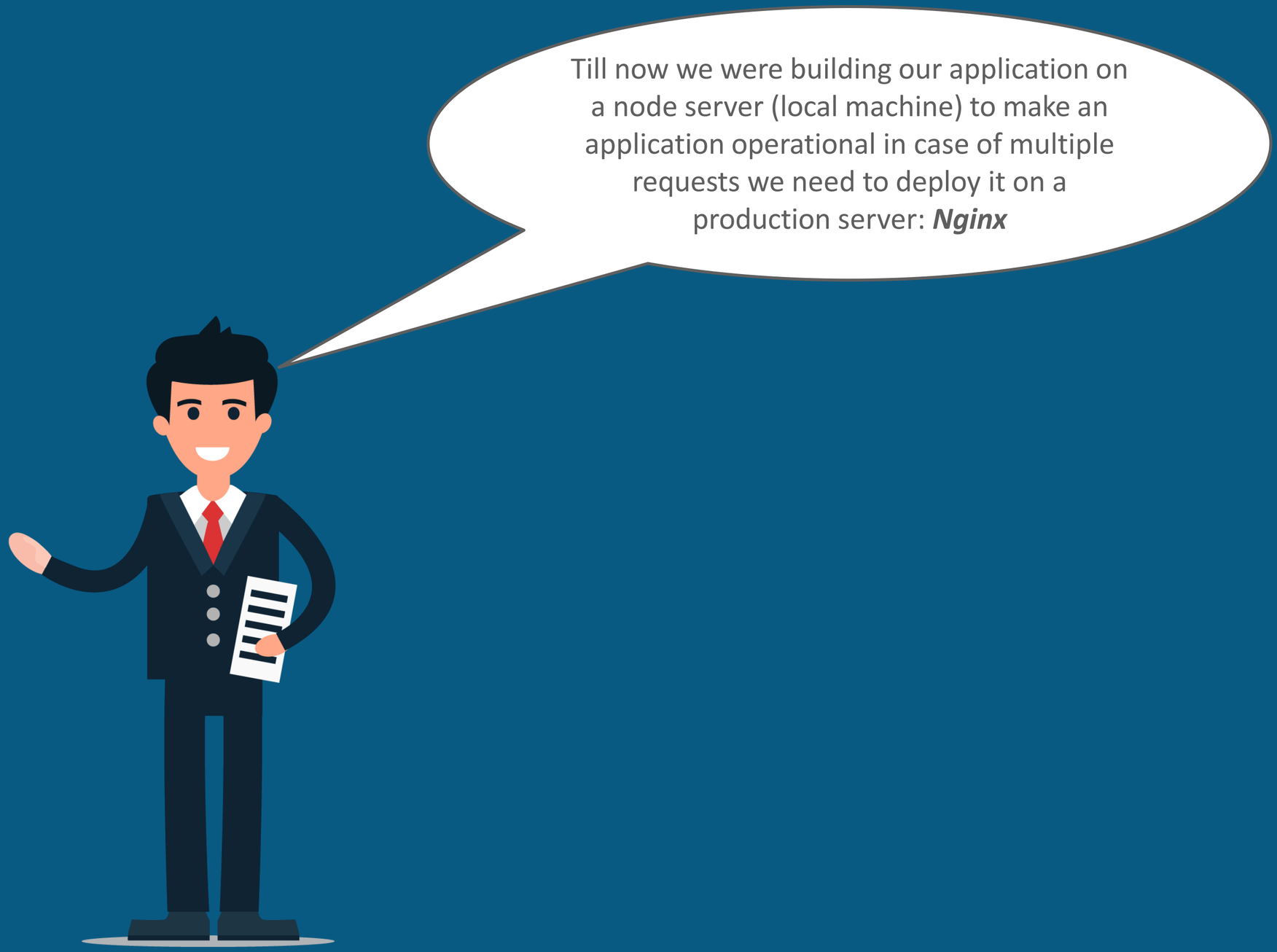
- To remove the file from git repository but not from the system use **cached** option

```
Syntax: git rm --cached <filename>
```

- An error shows up if you try to delete a staged file. You can force Git to remove a staged file by using **–f** flag

```
Syntax: git rm -f <filename>
```

Running Application On Production Server


Till now we were building our application on a node server (local machine) to make an application operational in case of multiple requests we need to deploy it on a production server: ***Nginx***

Nginx

What Is Nginx?

Nginx is a web server software by which you can make sure that your page load time is reduced.

01 An ***open source*** software and provides HTTP server capabilities

02 Mainly used for ***reverse proxying, caching and load balancing***

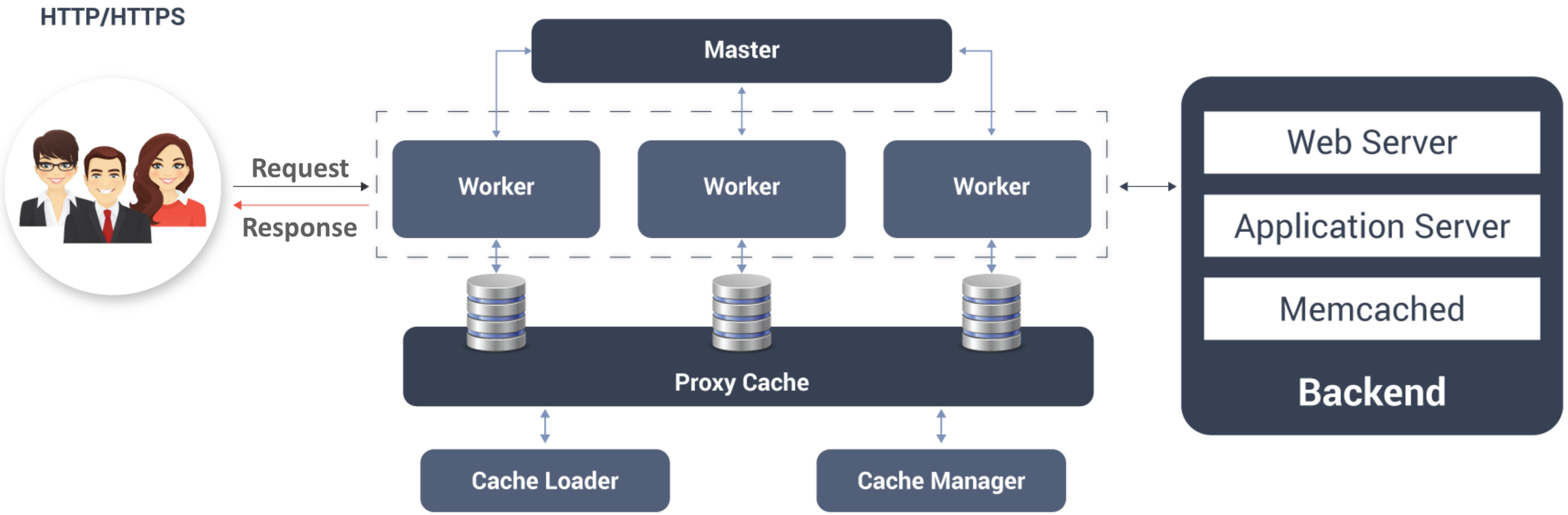
03 Designed for maximum ***performance and stability***

04 Uses a ***Non-threaded and event-driven*** architecture

Architecture Of Nginx

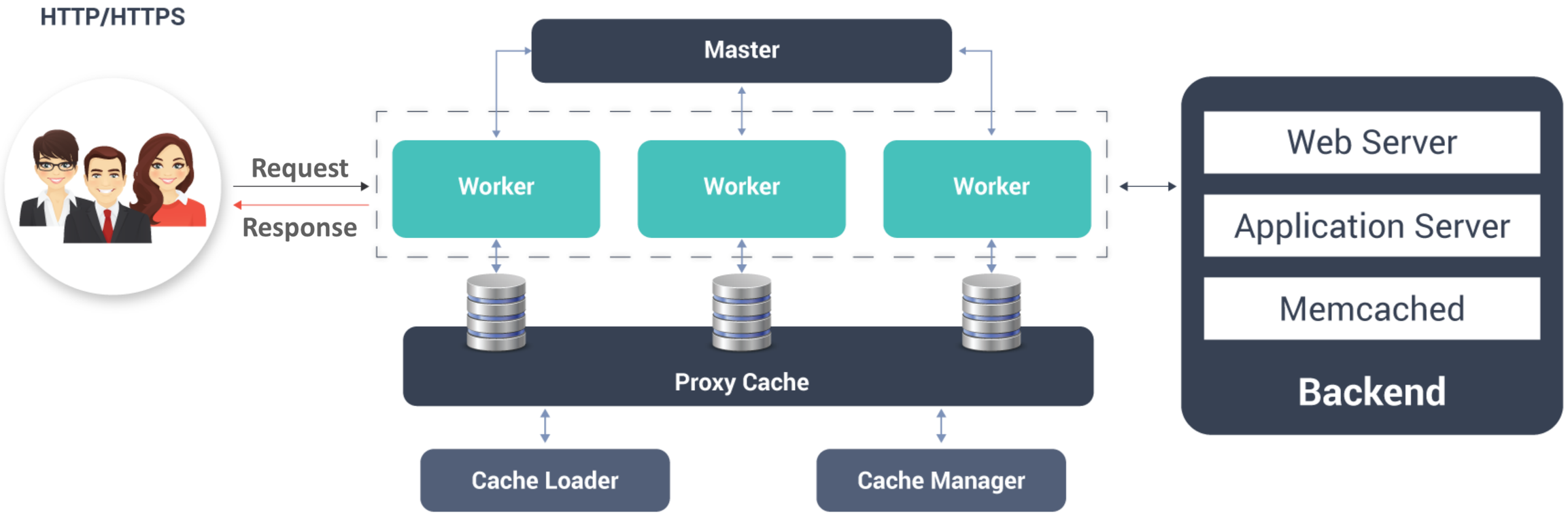
Architecture Of Nginx

Nginx uses master- slave architecture by supporting event-driven, asynchronous and non-blocking model.



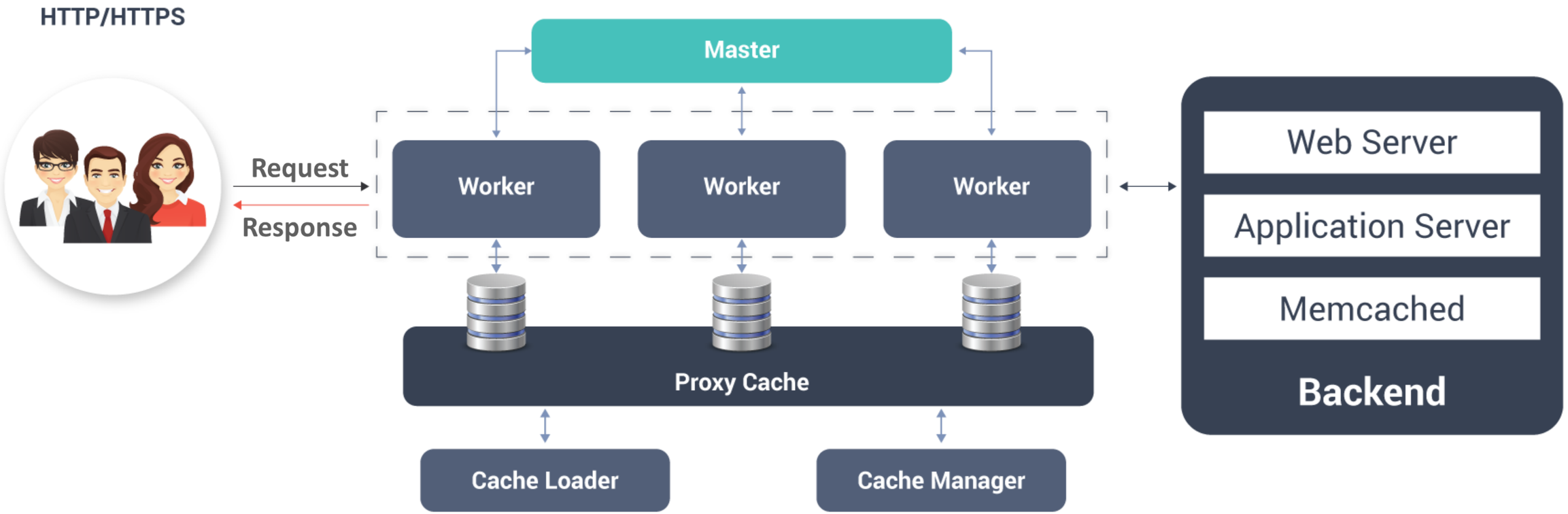
Architecture Of Nginx: Worker

Worker processors accepts the requests and starts processing the requests.



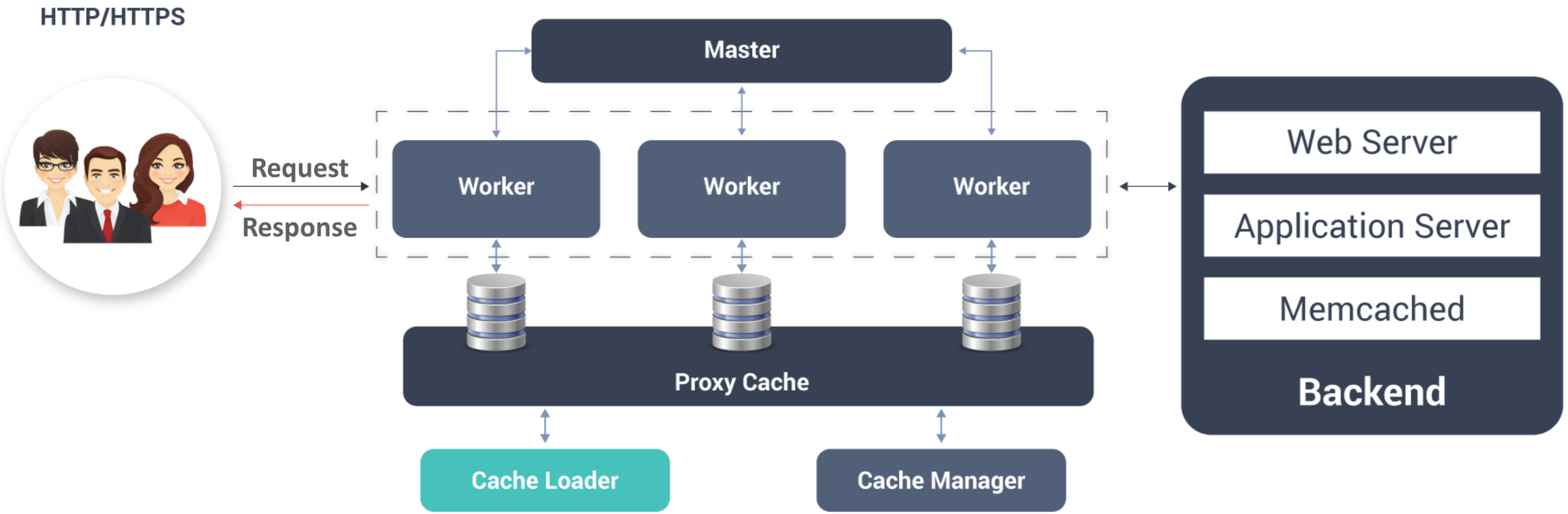
Architecture Of Nginx: Master

Master is responsible for reading and validating configuration, also it is responsible for starting, stopping and processing the worker.



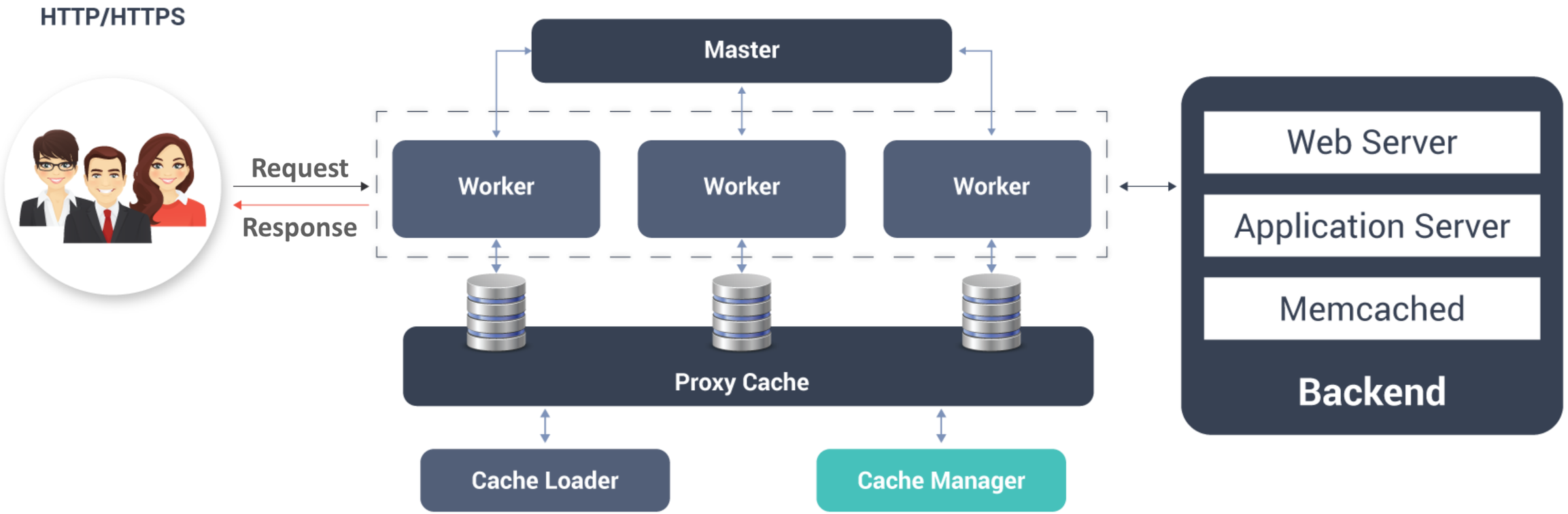
Architecture Of Nginx: Cache Loader

Cache Loader is responsible for checking the on disk cache items and populating engines in memory database with cache metadata.



Architecture Of Nginx: Cache Manager

Cache Manager is responsible for Cache expiration and invalidation.



How To Configure Nginx

01

Install Nginx from:
http://nginx.org/en/download.html

NGINX

02

Adjust Firewall

03

Configur the server port

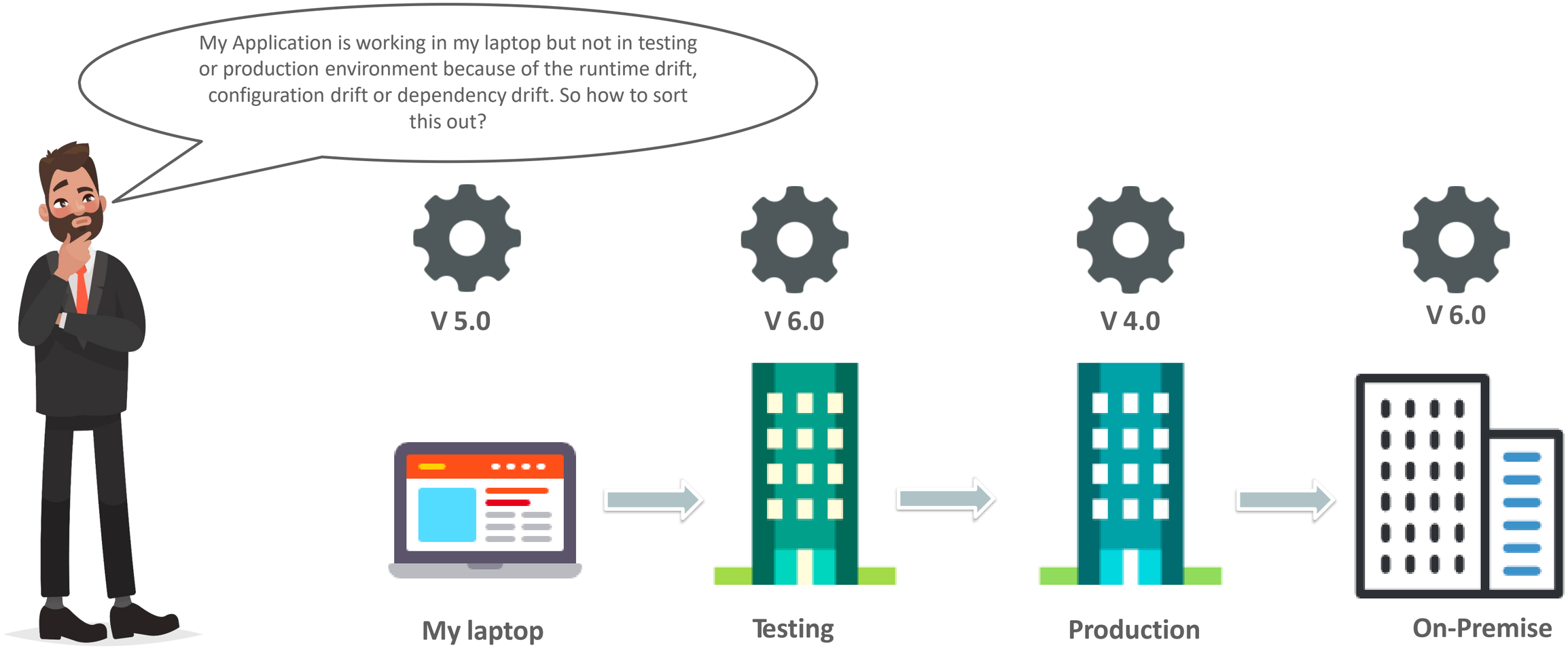
04

Check the working using:
Nginx
(type in command prompt)

Demo 3 :Running Your Application On Nginx

Deployment Using Docker Image Of React Application

Problems Before Containers



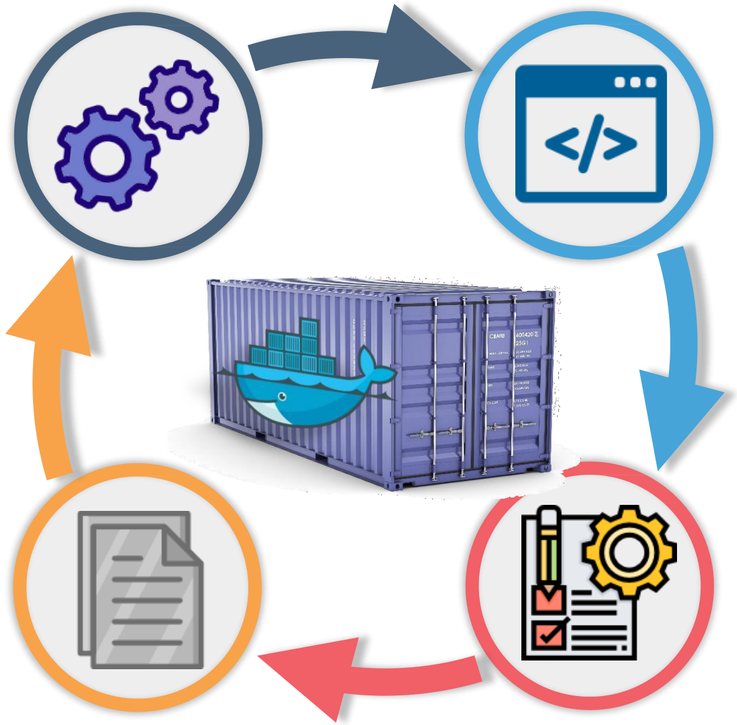
How Containers Solve That Issue

Containers are the solution to it. As it packs up the code and all its dependencies in single unit such that it can run reliably from one computing environment to another.

Even this can be done by a VM right?

Runtime Environment

Code

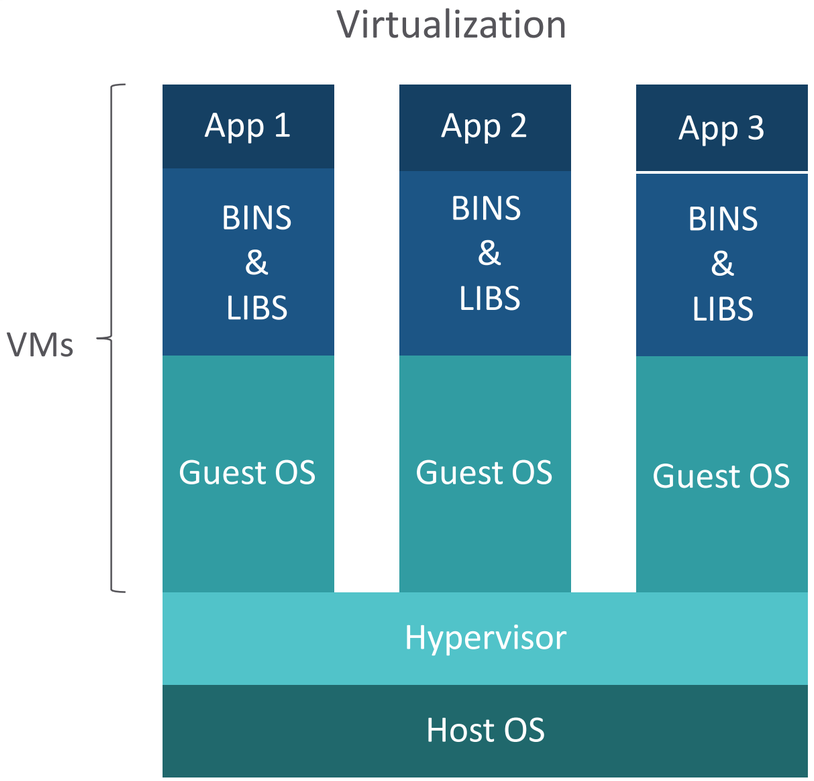


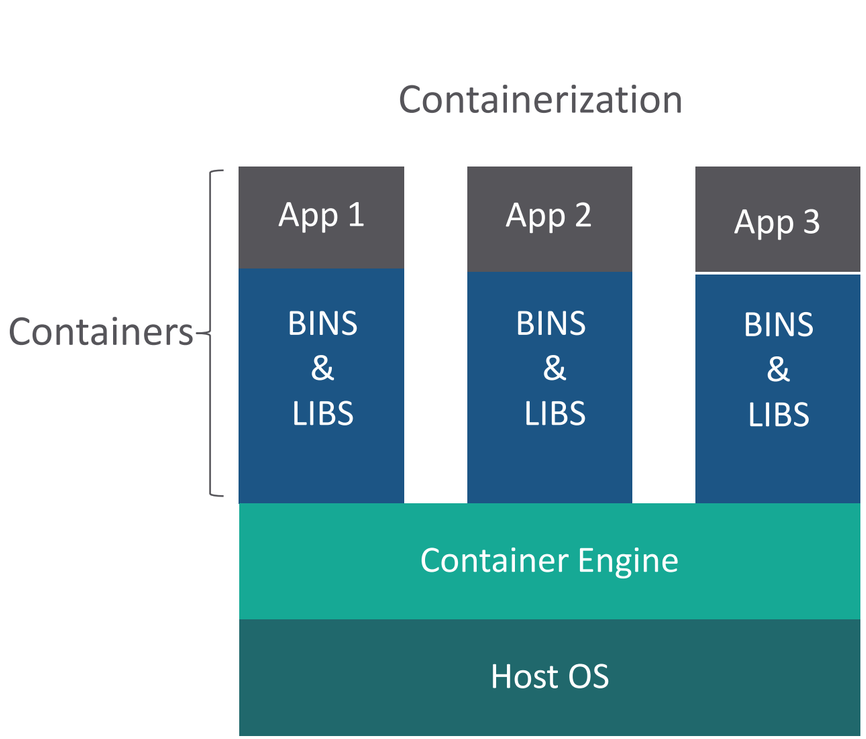
Dependencies

Configuration

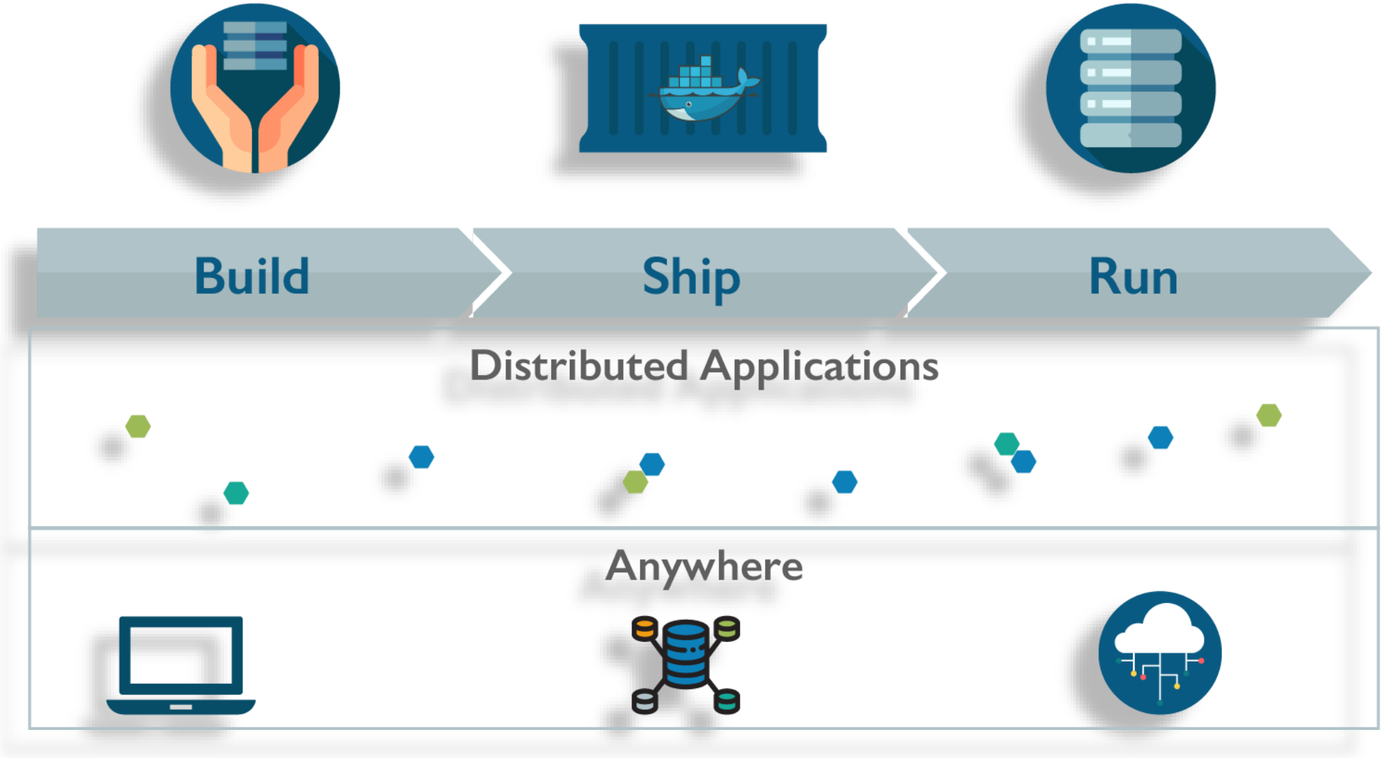
Why Not In Virtual Machine?

VMs take up a lot of resources as it has a full copy of a OS + virtual copy of hardware, which adds up to lot of RAM and CPU, whereas in containers you just need supporting programs, libraries and system resources to run a application.





Why Docker?



Develop an app using Docker containers with any language and any toolchain.

Ship the “Dockerized” app and dependencies anywhere - to QA, teammates, or the cloud - without breaking anything.

Scale to 1000s of nodes, move between data centers and clouds, update with zero downtime and more.

What Is Docker?

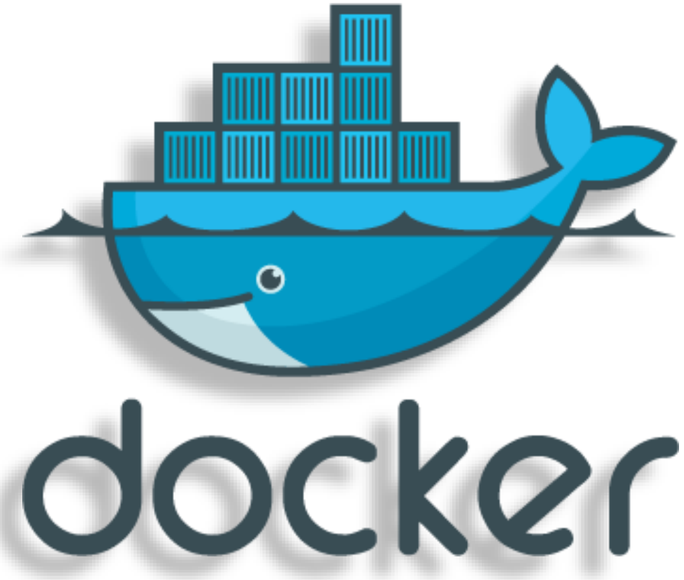
What Is Docker?

"BUILD, SHIP & RUN ANY SOFTWARE ANYWHERE"

Docker is a tool designed to build, deploy, and run applications with ease by using containers

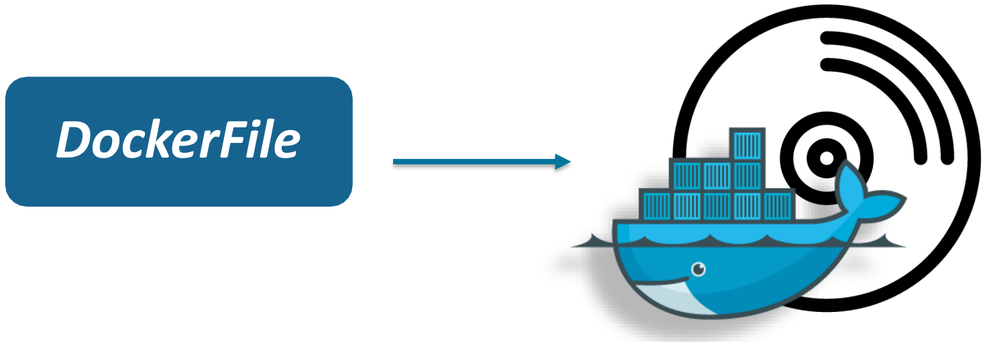
It allows a developer, **packaging of an application** with all of the requirements such as **libraries and other dependencies** and ship it all as one package

It ensures that your application **works seamlessly** in any environment (Development, Test or Production environment)



Terminologies In Docker

DockerFile



01

Docker file is the **basic building block** of Docker containers

Docker file is a file with **a set of instructions** and forms the base for any Docker Image

02

03

Every time, **base image** is going to **be based upon another image**. You are going to pick up a base image and build your application on that image

What Is An Image?

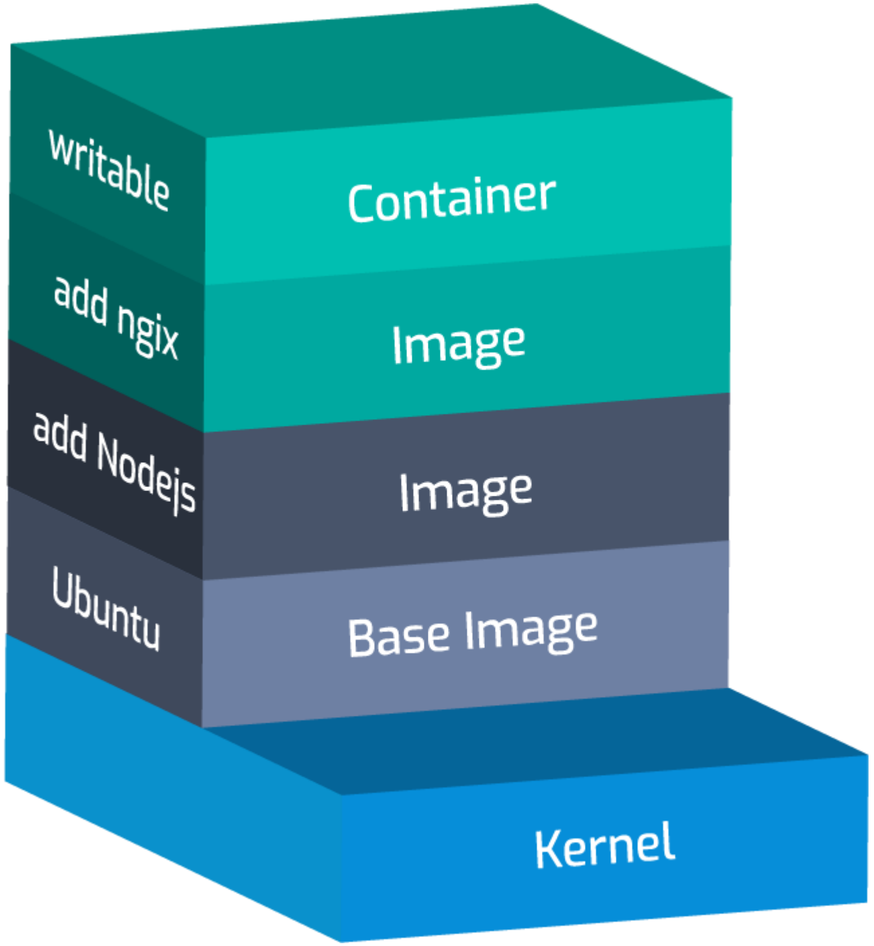


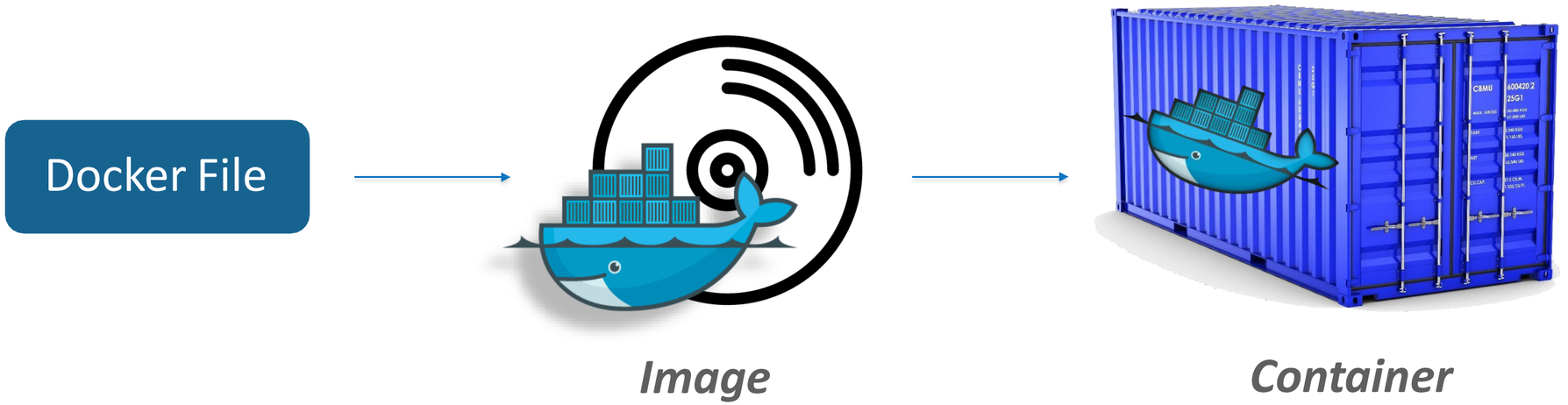
Image is a **read only file system**, that is used as a **template** to **launch a container**

Start from the **base image** that have your dependencies and add your **custom code**

When an **image is turned into a container** the **Docker engine** takes the image and adds a **read-write filesystem** on top (as well as initializing various settings such as the IP address, name, ID, and resource limits)

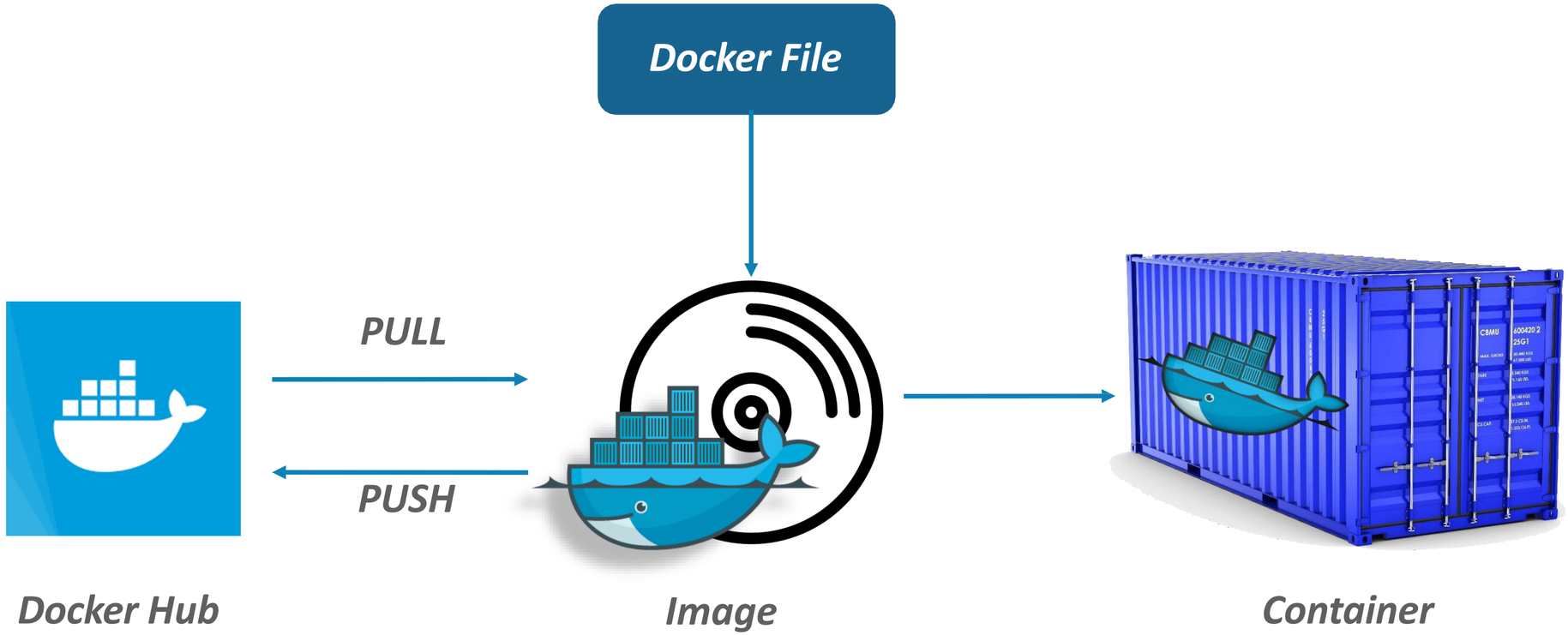
What Is Docker Container?

- Docker container is a package with everything required to **make the software run**, **except the OS**, hence making the application portable
- All you **need is libraries** and settings to make the software work on any system
- It makes the container: efficient, self-contained, lightweight and guarantees that the packaged software will always run, regardless of where it is deployed



Docker Hub

Docker Hub is a registry service on the cloud that allows you to push or pull Docker images that are official images or built by other users or vendors.



Basic Docker Commands

Commands	*Description*	*Example*
docker pull	Pulls a new Docker image from the Docker Hub	$ docker pull ubuntu
docker images	Lists down all the images in your local repo	$ docker images
docker run	Executes a Docker image & creates a running Container	$ docker run ubuntu
docker build	Compiles the Dockerfile for building custom Docker images	$ docker build –t MyUbuntuImage .
docker container	Performs various operations on the container like logs, kill, rm, start, run	$ docker container start
docker login	Logins to Docker Hub repo from the CLI	$ docker login
docker push	Pushes a Docker image on your local repo to the Docker Hub	$ docker push vardhanns/MyUbuntuImage

Basic Docker Commands

Commands	*Description*	*Example*
`docker ps`	Lists all the running containers, if '–a' flag is specified, shutdown containers are also displayed	**$ docker ps**
`docker stop`	Shuts down the container whose Container ID is specified in arguments	**$ docker stop fe6e370a1c9c**
`docker kill`	Kills the container by stopping its execution immediately	**$ docker kill fe6e370a1c9c**
`docker rm`	Removes the container whose Container ID is specified in arguments	**$ docker rm fe6e370a1c9c**
`docker rmi`	Removes the image whose name has been specified in arguments	**$ docker rmi MyUbuntuImage**
`docker exec`	Access an already running container and perform operations inside the container	**$ docker exec -it fe6e370a1c9c bash**
`Docker commit`	Creates a new image of an edited container on the local repo	**$ docker commit fe6e370a1c9c vardhanns/MyModifiedImage**

Demo 4: Build Docker Image Of The Application


Questions


Comments
Ratings
Suggestions
FEEDBACK
Survey
Likes
Ideas