

# Clean project

UA.DETI.IES

# Software engineering

---

## ❖ Example:

- Someone hired us to create an application to compete against Netflix

## ❖ How we do that?

## ❖ Who is responsible for

- deployments?
- defining the application features?
- infrastructure?
- ensuring scalability?
- so many other things ...



# Software engineering - let's revise

---

1. Software development process
  - Sequential model (Waterfall)
  - Incremental model
  - Evolutionary/Iterative models
2. Agile development methods
  - Agile principles and project management
3. DevOps Technical benefits
  - Continuous software delivery
  - Faster delivery of features (time to market)



# Roles

---



Team manager



Product owner



Architect



DevOps master

# Team manager



- ❖ Moderates the team discussions
  - Promote collaboration in the team
  - Take initiative to solve problems
- ❖ Manages and assign tasks
- ❖ Can be seen as a Scrum master
- ❖ **Responsible for delivering project outcomes in time**

# Product owner



- ❖ Represents the interests of the stakeholders
- ❖ Knows what the application should do
  - Features
  - Requirements
  - User stories
- ❖ Responsible for accepting the solution increments
  - Should revise new releases

# Architect



- ❖ Responsible for the software architecture
  - Modeling the applications
  - Interactions between components
- ❖ Knows the technologies used
  - Frontend
  - Backend
  - Caching
  - Message queues and others

# DevOps master

---



- ❖ Responsible for the infrastructure
- ❖ Ensures system portability
- ❖ Knows everything about:
  - Deployment machine
  - Git repository
  - Cloud infrastructure
  - Databases operations
  - Other aspects



# Roles

---



Team manager



Product owner



Architect



DevOps master



Developer

# Software Planning

---

## ❖ Specification

- Defining what the system should do

## ❖ Design and implementation

- Defining the organization of the system and implementing the system

## ❖ Validation

- Checking that it does what the customer wants

## ❖ Evolution

- Changing the system in response to changing the customer needs

# Software Planning

---

## ❖ Specification

- Defining what the system should do

## ❖ Design and implementation

- Defining the organization of the system and implementing the system

## ❖ Validation

- Checking that it does what the customer wants

## ❖ Evolution

- Changing the system in response to changing the customer needs

# Specification

---

- ❖ Definition of requirements and stories
  - Already discussed in previous classes
- ❖ Use tools for managing the development
  - Prioritize, assign, and track the work

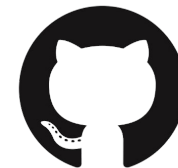
# Specification

---

- ❖ Definition of requirements and stories
  - Already discussed in previous classes
- ❖ Use tools for managing the development
  - Prioritize, assign, and track the work
- ❖ But... How to do that?
  - Using project planning tools
    - Some with code repository incorporated



**Pivotal  
Tracker**



**GitHub**

 **Jira Software**

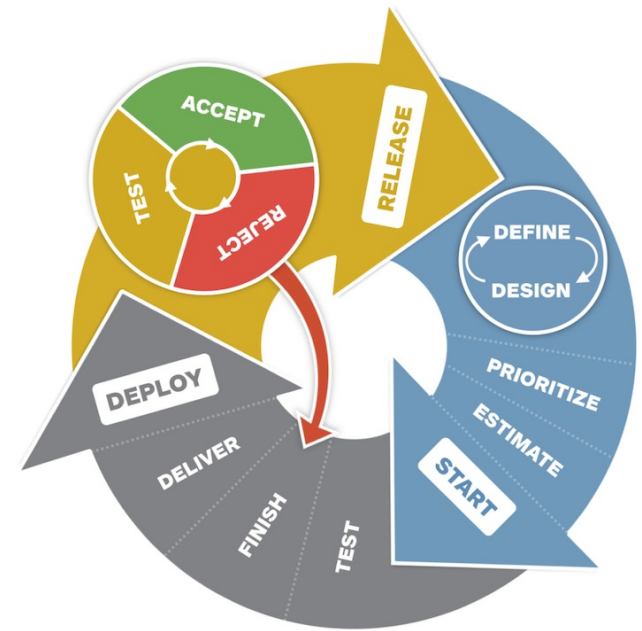


**GitLab**

# Pivotal Tracker

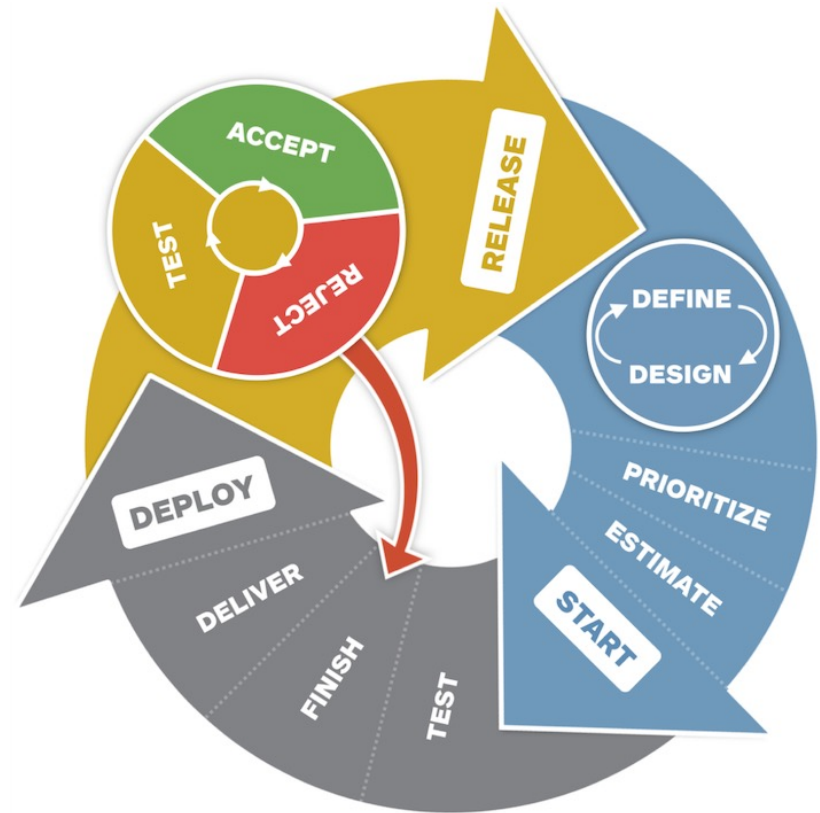
*muito antigo...*

- ❖ Agile project manager tool
- ❖ Allows the easy management of stories
  - Features, bugs, chores and releases
- ❖ Estimation of effort
  - Divide into 4 levels
- ❖ Backlog divide into iterations
- ❖ Provides good documentation



# Workflow Overview <sup>TOP</sup>

1. Write stories
2. Prioritize stories
3. Estimate stories *→ the cost*
4. Start stories
5. Finish and deliver stories
6. Test stories
7. Accept or reject stories
8. Stories move to the Done panel



# GitHub

❖ GitHub is more than a code repository



❖ Project management features

- Team management
- Issue tracking
  - Could follow similar principles as stories

❖ Community continuously creating new apps

- For personalized management

❖ Can GitHub replace the Pivotal Tracker?



*Sempre foi uma gestão de projeto*

*Servem  
propósitos  
diferentes*





BACK Backlog 13

Collect satellite data and deliver to farmers  
Added by Sam

Launch Plan  
Added by Sam

UI for accessing app on tractor screen  
Added by Vijay

Fix CSV rendering  
Added by Emily

Share farms' soil moisture data  
Added by Eddie

File sharing permissions  
Added by fabianperoz

🕒 In Progress 6

Crawl tractor engine data (John Deere)  
#68801 opened by Melinda

Performance updates for data script  
#71011 opened by Sam

User testing with farmers in China  
Added by Eddie

Figure out internationalization  
New doc editor (@jo  
Added by Sophie

🚀 Ready to deploy 1

[Data] Soil data collection scripts  
#71001 opened by Vijay

+ Add column

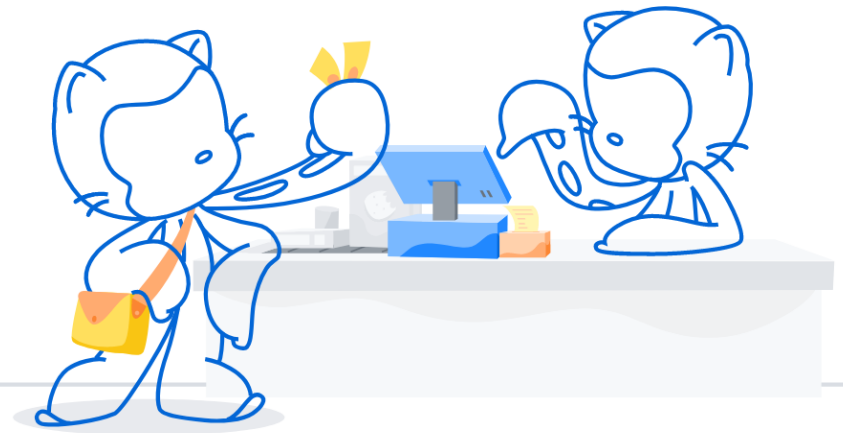
# GitHub Marketplace

## ❖ Apps to integrate in GitHub projects

## ❖ Different categories

- Code review
- Continuous integration
- Security
- Testing
- Monitoring
- Among others

"bad smells" → padrões no código que devem ser substituídos



# Software Planning

---

## ❖ Specification

- Defining what the system should do

## ❖ **Design and implementation**

- **Defining the organization of the system and implementing the system**

## ❖ Validation

- Checking that it does what the customer wants

## ❖ Evolution

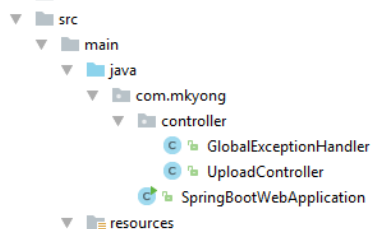
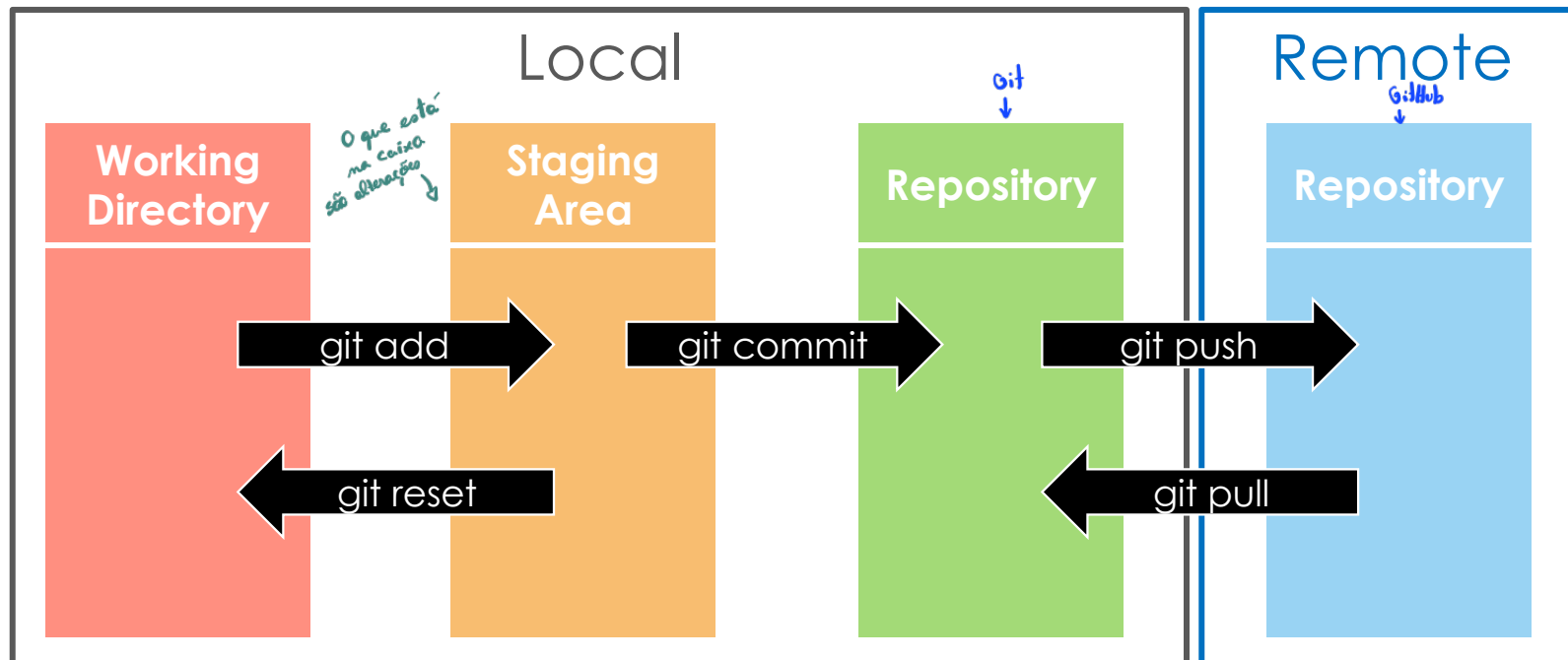
- Changing the system in response to changing the customer needs

# Feature-branching workflow

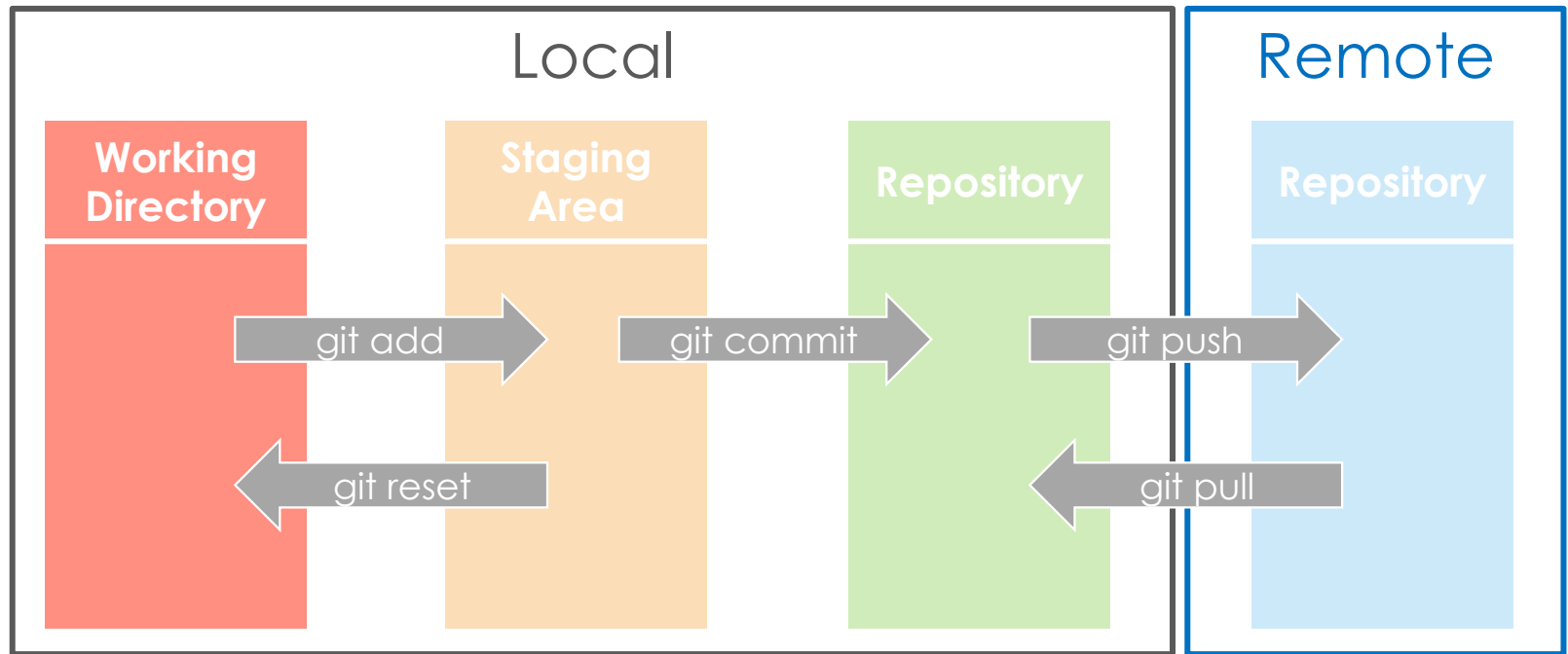
- ❖ Code repositories
  - Version control system
    - Git
    - SVN
    - (..)
- ❖ Not new for you, but...
- ❖ Let's see about how this works
  - And some good practices



# Git !



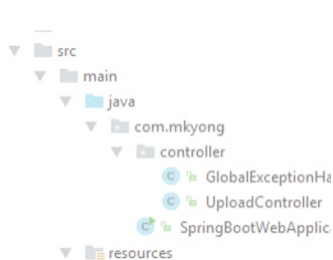
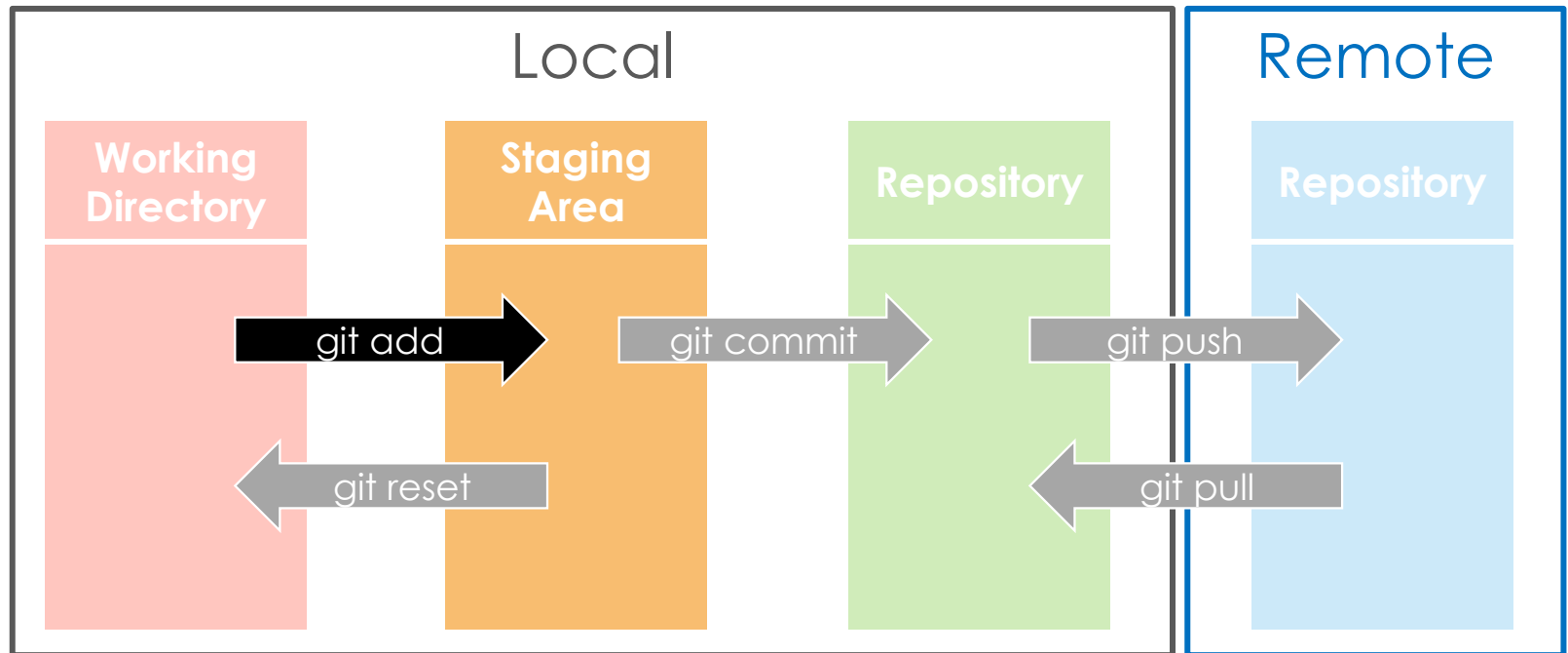
# Git



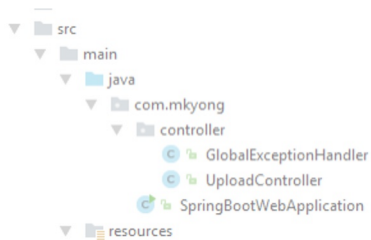
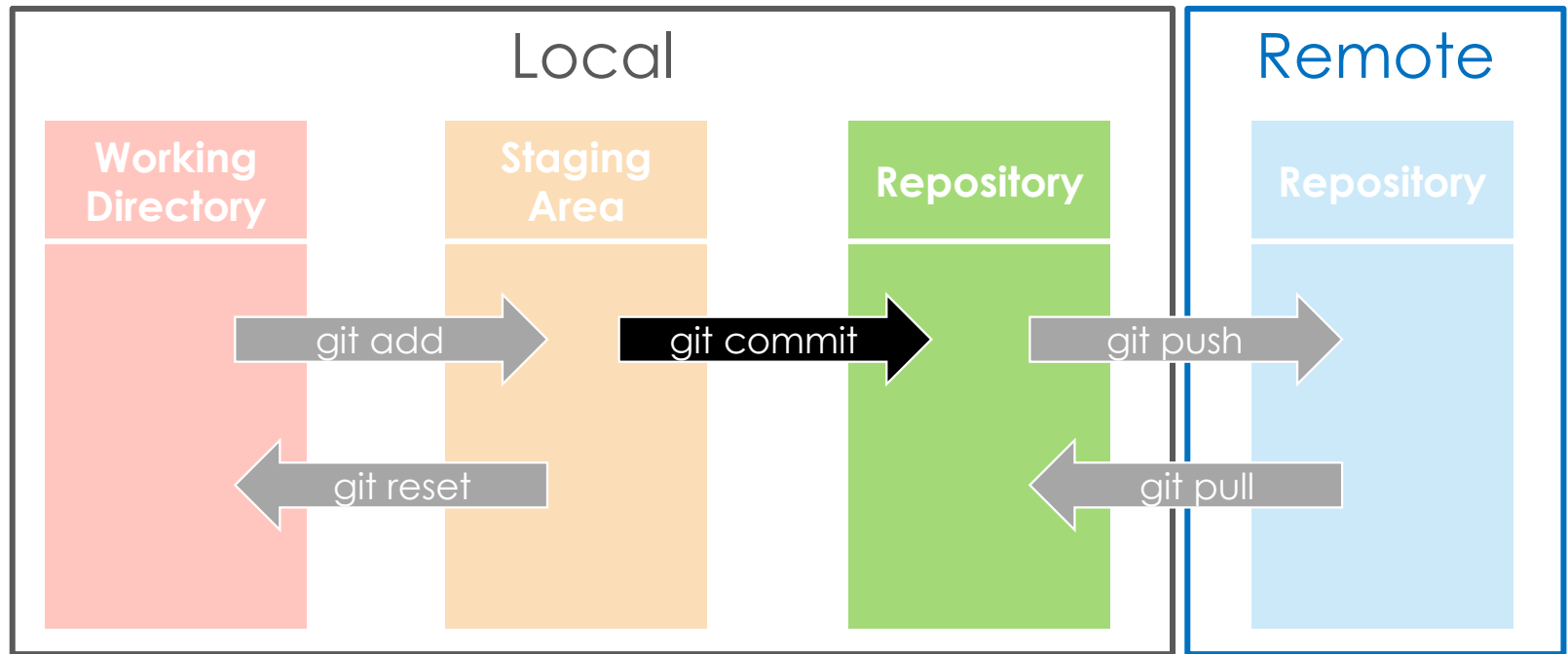
```
▼ src
  ▼ main
    ▼ java
      ▼ com.mkyong
        ▼ controller
          ● GlobalExceptionHandler
          ● UploadController
          ● SpringBootApplication
        ▼ resources
```



# Git

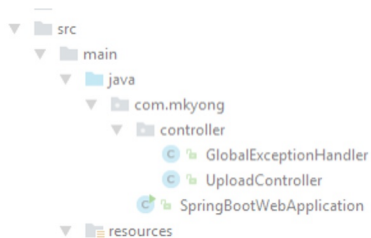
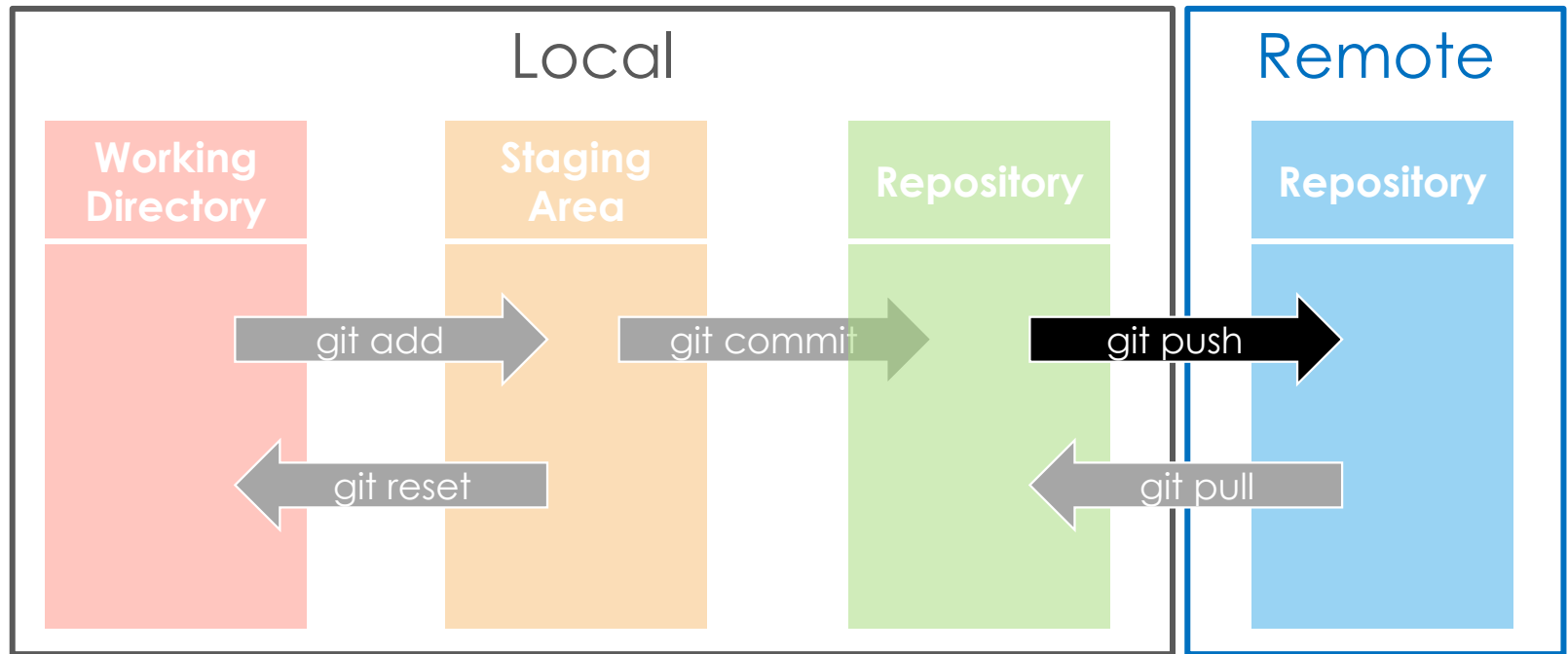


# Git





# Git



# What is a commit?

---

- ❖ Fundamental operation to record changes to the repository
- ❖ Unique SHA-1 hash that identifies the commit
- ❖ Includes
  - the content of all files being committed
  - the commit message
  - the author's name and email
  - the committer's name and email
  - the timestamp
  - maybe more...

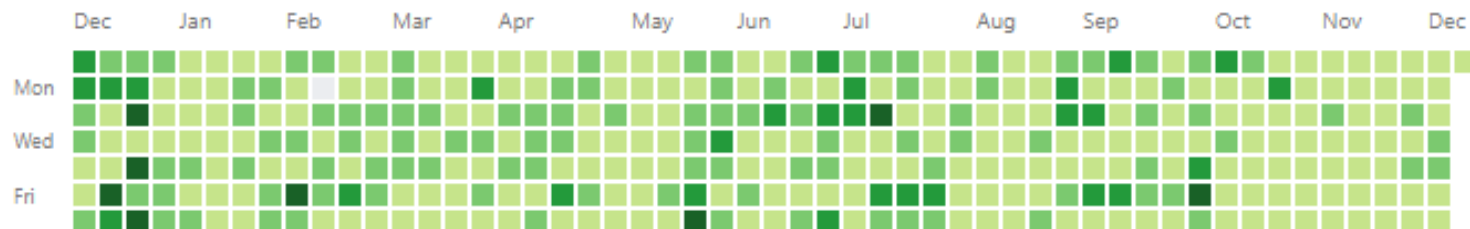
# Daily commits

## ❖ Scenario

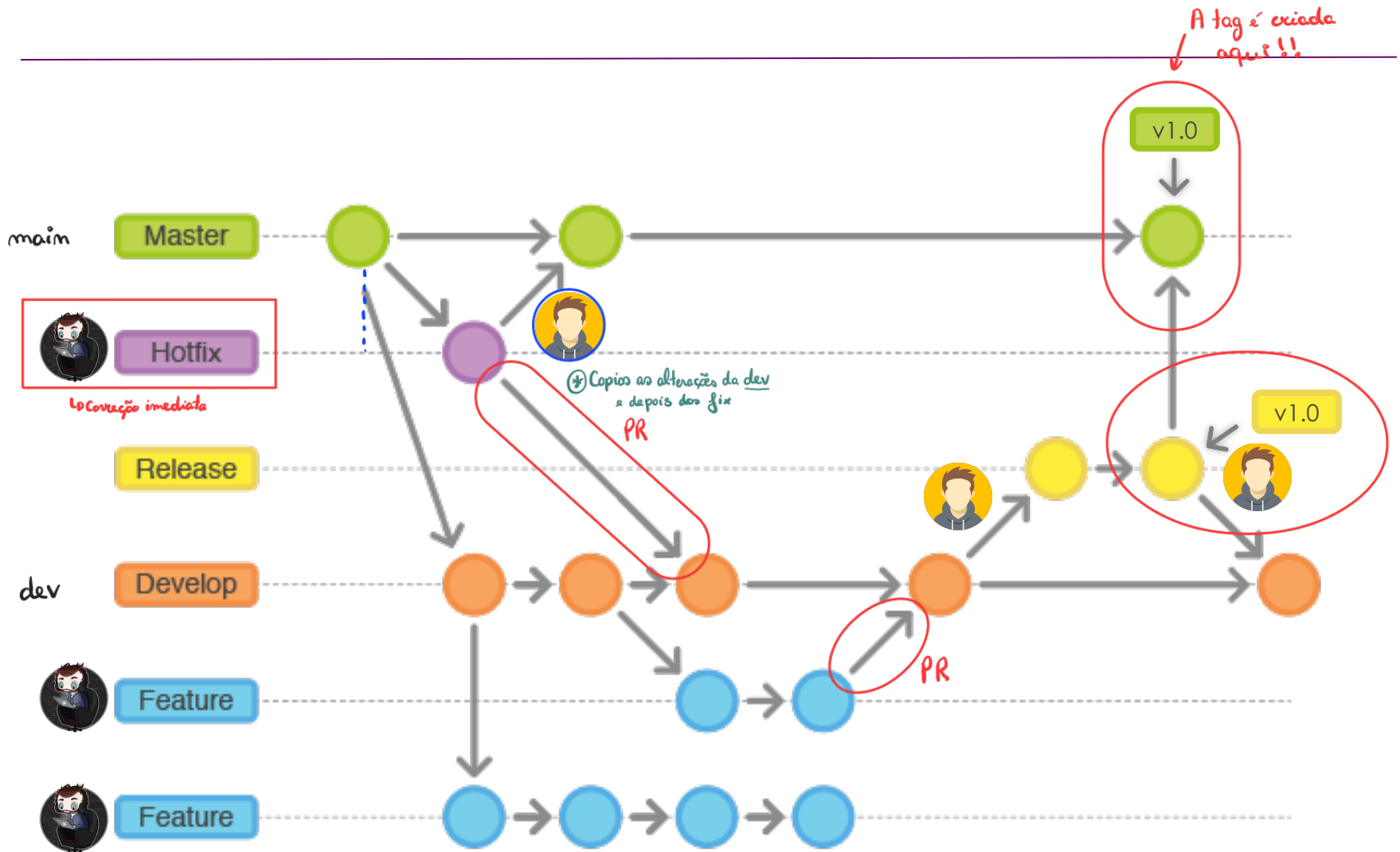
- Working on a project for two weeks without doing a single commit.
- The disk decides to die.
- **What should we do now?**

## ❖ Never wait to finish a task to create a commit

## ❖ **Every day**, commit the work and push the code to the repository



# Git workflow



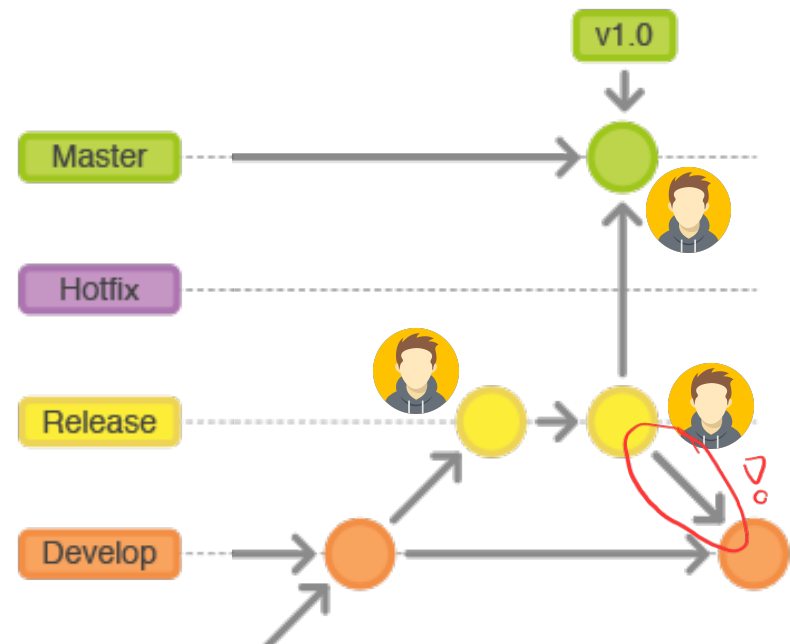
# New release

- ❖ Preparing the product to show the client
  - Closing one development cycle

- ❖ Checkout from dev

- ❖ When release is ready
  - Merge release into master
  - Merge branch into dev

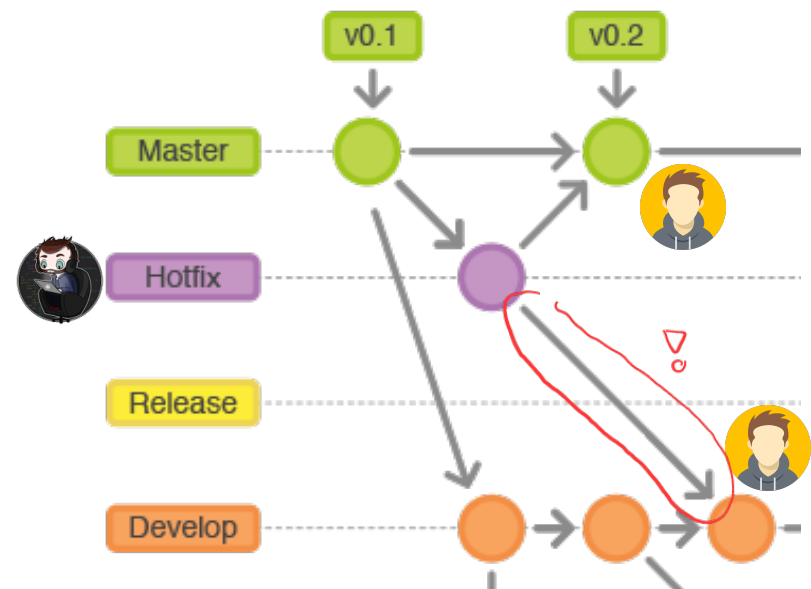
Pull Request



- ❖ Why these two merges? → For continuous check of bugs...

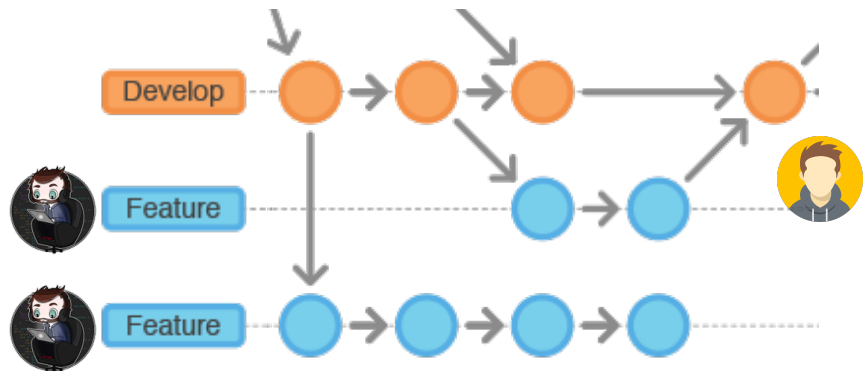
# Hotfix

- ❖ Catastrophic bug was found
- ❖ Procedure
  1. Checkout master
  2. Fix bug
  3. Merge into master and dev
- ❖ Typically, does not require a new branch for a release



# New feature

- ❖ **New branch** for each feature
- ❖ Checkout from dev
- ❖ When feature is complete
  - Merge dev into feature branch
  - Merge branch into dev



- ❖ Why these two merges? *To synchronize first the feature with dev!*


# Pull/Merge Requests


---

- ❖ Merging branches needs a request
  - Usually to protected branches (master and dev)
- ❖ Pull request needs approval
  - From git manager (DevOps master)
- ❖ Sometimes the implementation needs improvements
  - Feature is incomplete
  - Complex conflicts during merging



# Pull/Merge Requests

 base: dev ← compare: feature/configure-timezone-on-... ✓ **Able to merge.** These branches can be automatically merged.



Add option to configure timezone in the user profile

Write Preview

H B I ≡ <> 🔗 ⋮ ⋮ ☑ @ ↻ ↶

<!-- Provide a general summary of your changes in the Title above -->

**## Description**

<!-- Describe your changes in detail -->

**## Related Issue**

<!-- This project only accepts pull requests related to open issues -->

<!-- If suggesting a new feature or change, please discuss it in an Issue first -->

<!-- If fixing a bug, there should be an issue describing it with steps to reproduce -->

<!-- Please link to the issue here: -->

**## Motivation and Context**

<!-- Why is this change required? What problem does it solve? -->


<!-- If it fixes an open issue, please link to the issue here. -->

**## How Has This Been Tested?**

<!-- Please describe in detail how you tested your changes. -->

<!-- Include details of your testing environment, and the tests you ran to -->

<!-- see how your change affects other areas of the code, etc. -->

Attach files by dragging & dropping, selecting or pasting them. 

Create pull request ▾

**Reviewers** ⚙️

No reviews—at least 1 approving review is required.

**Assignees** ⚙️

No one—assign yourself

**Labels** ⚙️

None yet

**Projects** ⚙️

None yet

**Milestone** ⚙️

No milestone

**Linked Issues** ⓘ

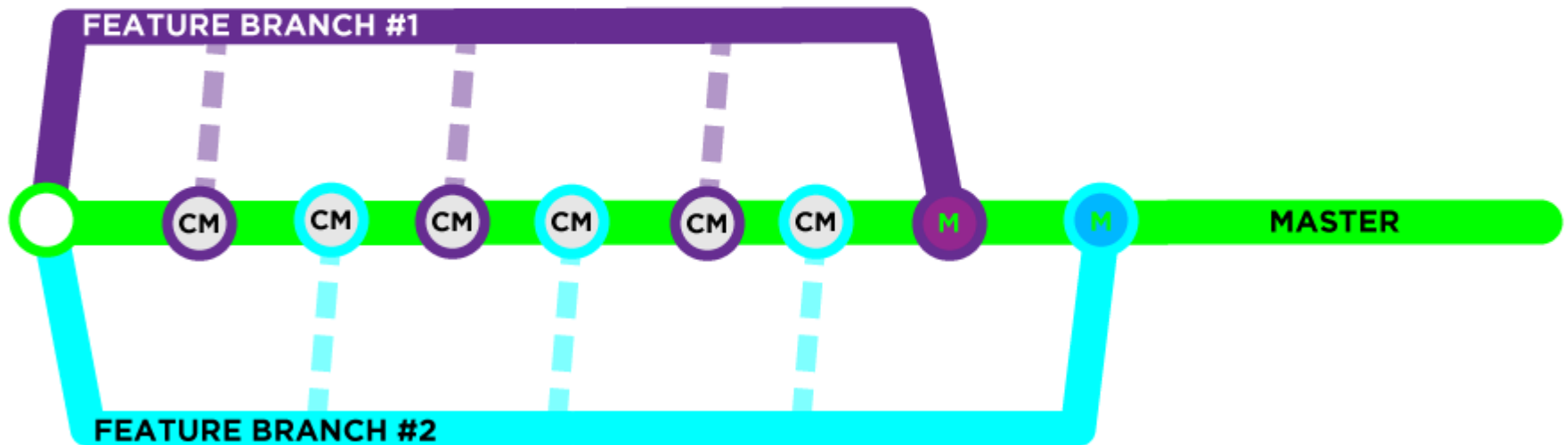
Use [Closing keywords](#) in the description to automatically close issues

**Helpful resources**

[GitHub Community Guidelines](#)

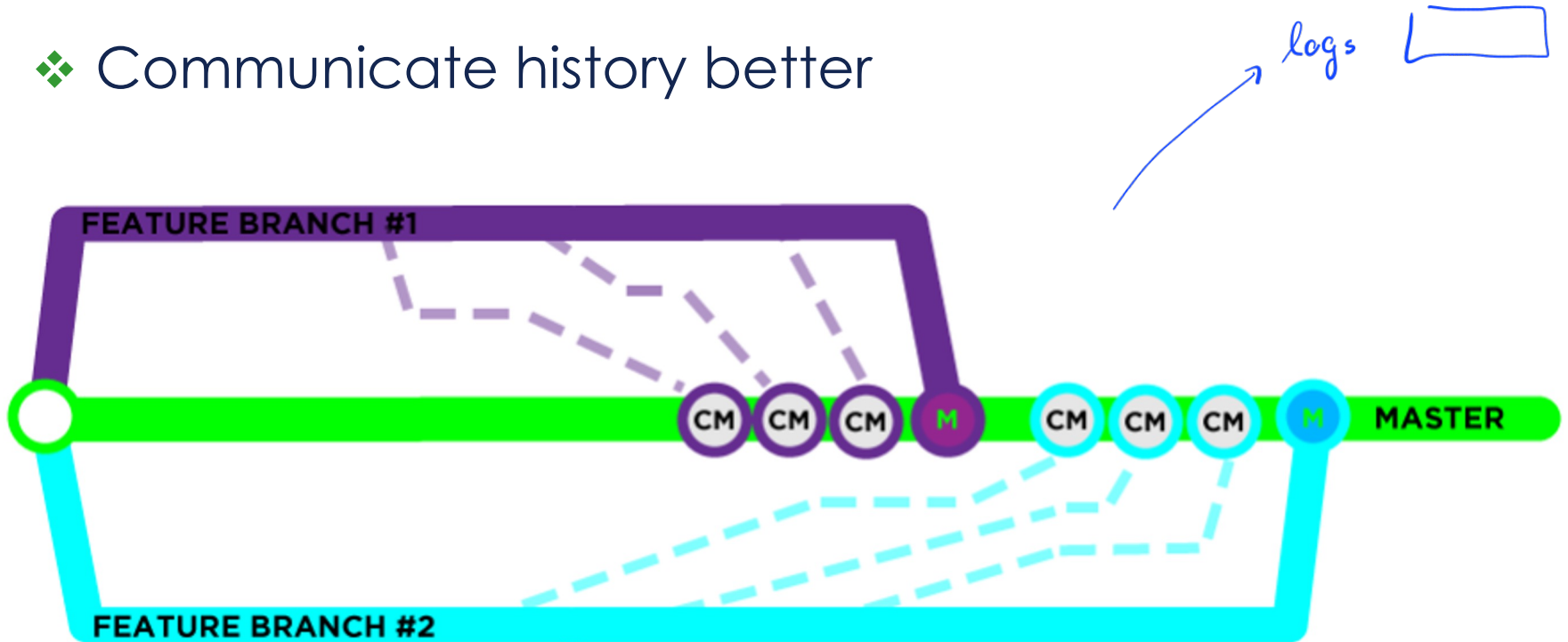
# Merge workflow

- ❖ Commits interlock
- ❖ Hard to follow commit history



# Rebase workflow

- ❖ Commits do not interlock
- ❖ Communicate history better



# Merge or rebase?

---

## ❖ Merge

- Advantages
  - Non-destructive, existing branches are not changed in any way, you just have another new commit - Easy to undo
- Disadvantages
  - Pollutes the history of your repo, makes it hard to understand the evolution

## ❖ Rebase

- Advantages
  - Much cleaner project history
  - Linear project history
- Disadvantages
  - Easy to do it wrong, rewrites history
  - Tougher to resolve conflicts

# Branching Names

---

- ❖ Each programmer likes his own convention
- ❖ These conventions are not standards
- ❖ Branch names are important
  - Like good names when coding variables

CATEGORY	DESCRIPTION
bug	Bug fixing
imp	Improvement on already existing features
new	New features being added
wip	Works in progress - Big features that take long to implement and will probably hang there
junk	Throwaway branch created to experimentation
release	New release before merging with master

# Examples Branching Names

---

- ❖ URL redirects to the wrong page #123
  - bug/fixURLRedirect (good)
  - bug/fix\_url\_redirect (also good)
  - bug/fix\_url\_redirect\_123 (better)
- ❖ Accounts: URL redirects to the wrong page #123
  - bug/accounts/fix\_url\_redirect\_123 (much better)

# Troubleshooting

- ❖ Merging conflicts that are too complex
  - Request the developer to update branch
  - Merging current dev into branch
- ❖ Committed sensitive data
  - It is possible to revert
  - But it could be a dangerous
- ❖ Dependencies
  - "It works in my machine"



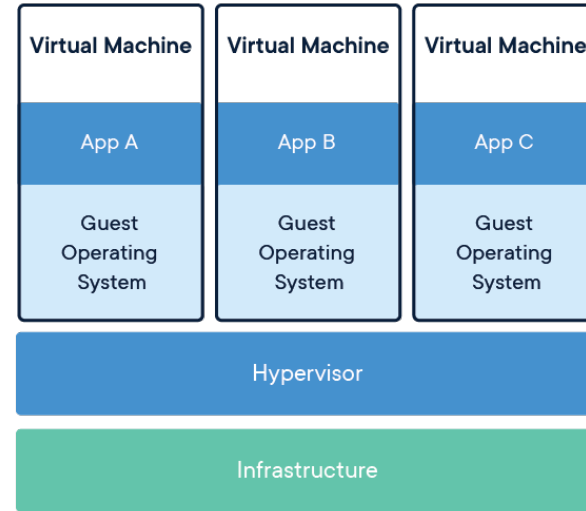
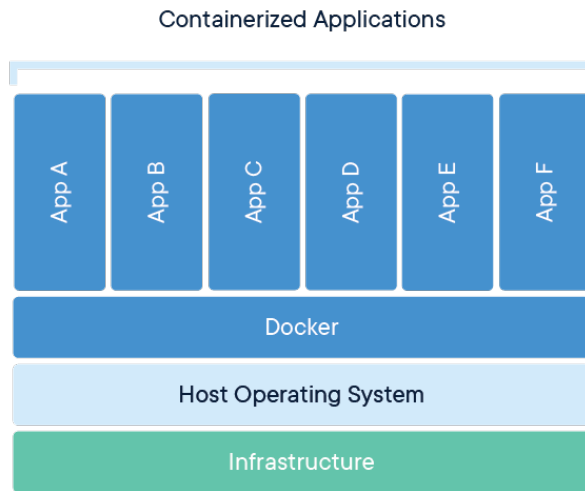
# Containers-based deployment

---

- ❖ A good solution for dependency problems
- ❖ Everyone is using the same environment
- ❖ Production and development environments are very similar
- ❖ Simplifies the integration of different services
- ❖ Easy to deploy



# Virtualization & Containerization

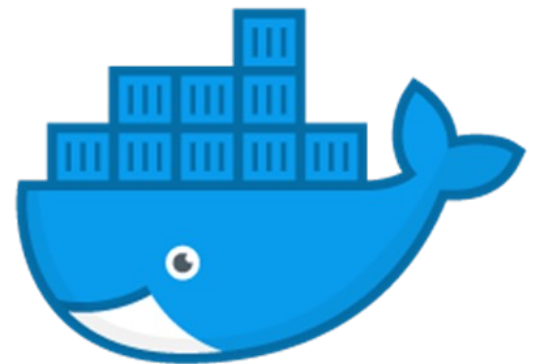


Virtualization	Containerization
More secure and fully isolated	Less secure and isolated at the process level
Heavyweight, high resource usage	Lightweight, less resource usage
Hardware-level virtualization	Operating system virtualization
Each virtual machine runs in its own operating system	All containers share the host operating system
Startup time in minutes and slow provisioning	Startup time in milliseconds and quicker provisioning

# Docker

---

- ❖ Already studied in practical classes
  - But let's review a few concepts
- ❖ Production and development images are the same
  - But with different configurations
- ❖ In production
  - **Always** use volumes for the sensitive data
  - Containers die, volumes not (usually)



# Images

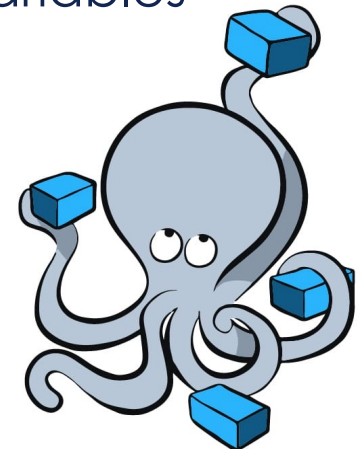
---

- ❖ Images are the bases of containers
- ❖ One Image can serve multiple containers
  - But one container can only have one image
- ❖ Allows inheritance
  - FROM ubuntu:20.04
  - FROM myImage:base
- ❖ Should I use an official image or create mine?

# Docker Compose

---

- ❖ Simplifies the integration between containers
- ❖ Allows container orchestration
  - Based on a certain order
- ❖ Do not change docker-compose.yml file
  - Instead, define variables (.env)
  - Create a .env-example file with the default variables
- ❖ After configured, the startup is trivial
  - `docker-compose up -d`



# Example

# Example – client's needs

---

- ❖ Client wants a web application to generate random number
- ❖ Procedures:
  - User sets a seed
  - Clicks generate a random number
  - Random number is generated
- ❖ Let's plan this project



Product owner

# Story

---

**Title:** Random number generator    **Priority:** 1    **Estimate:** 1

**As an** anonymous user  
**I want to** provide a number  
**so that I can** get a random number based on my input

## Acceptance criteria

**Given** the anonymous user wants to generate a random number

**When** the user provides a number in an HTML input component

**Then** the system shall display a random number based on the user's input

# Pivotal tracker

Random number generator

ID #175785144

Collapse

STORY TYPE

★ Feature

POINTS

1 Point

REQUESTER

JR John Rambo

OWNERS

<none>

FOLLOW THIS STORY

(1 follower)

STATE

Start

Unstarted

REVIEWS

+ add review

Requested: a minute ago

BLOCKERS

+ Add blocker or impediment

DESCRIPTION

**As a** anonymous user

**I want to** provide a number

**so that I can** get a random number based on my input

**Acceptance criteria**

**Given** the anonymous user wants to generate a random number

**When** the user provides a number in an HTML input component

**Then** the system shall display a random number based on the user's input

LABELS

Add a label

CODE

Paste link to pull request or branch...



# GitHub – another approach

## Random number generator #1

[Edit](#)[New issue](#)

joorafaelalmeida opened this issue 6 hours ago · 0 comments



joorafaelalmeida commented 6 hours ago



As a anonymous user

I want to provide a number

so that I can get a random number based on my input

### Acceptance criteria

Given the anonymous user wants to generate a random number

When the user provides a number in an HTML input component

Then the system shall display a random number based on the user's input

### Assignees

No one—assign yourself



### Labels



feature

### Projects



Release 1.0

To do ▾

### Milestone



No milestone

[Code](#)[Issues 1](#)[Pull requests](#)[Actions](#)[Projects 1](#)[Wiki](#)[Security](#)[Insights](#)[Settings](#)

### Release 1.0

Updated 3 minutes ago

1 To do



Random number generator



#1 opened by joorafaelalmeida

feature


0 In progress





0 Done



# GitHub Code Repository


 new/number/ran... ▾


 3 branches

 0 tags

[Go to file](#)

[Add file ▾](#)

 **Code**  
▾

Switch branches/tags 

Branches


Tags


master default

dev


✓ new/number/random\_number\_generato...

[View all branches](#)

 [Contribute ▾](#)

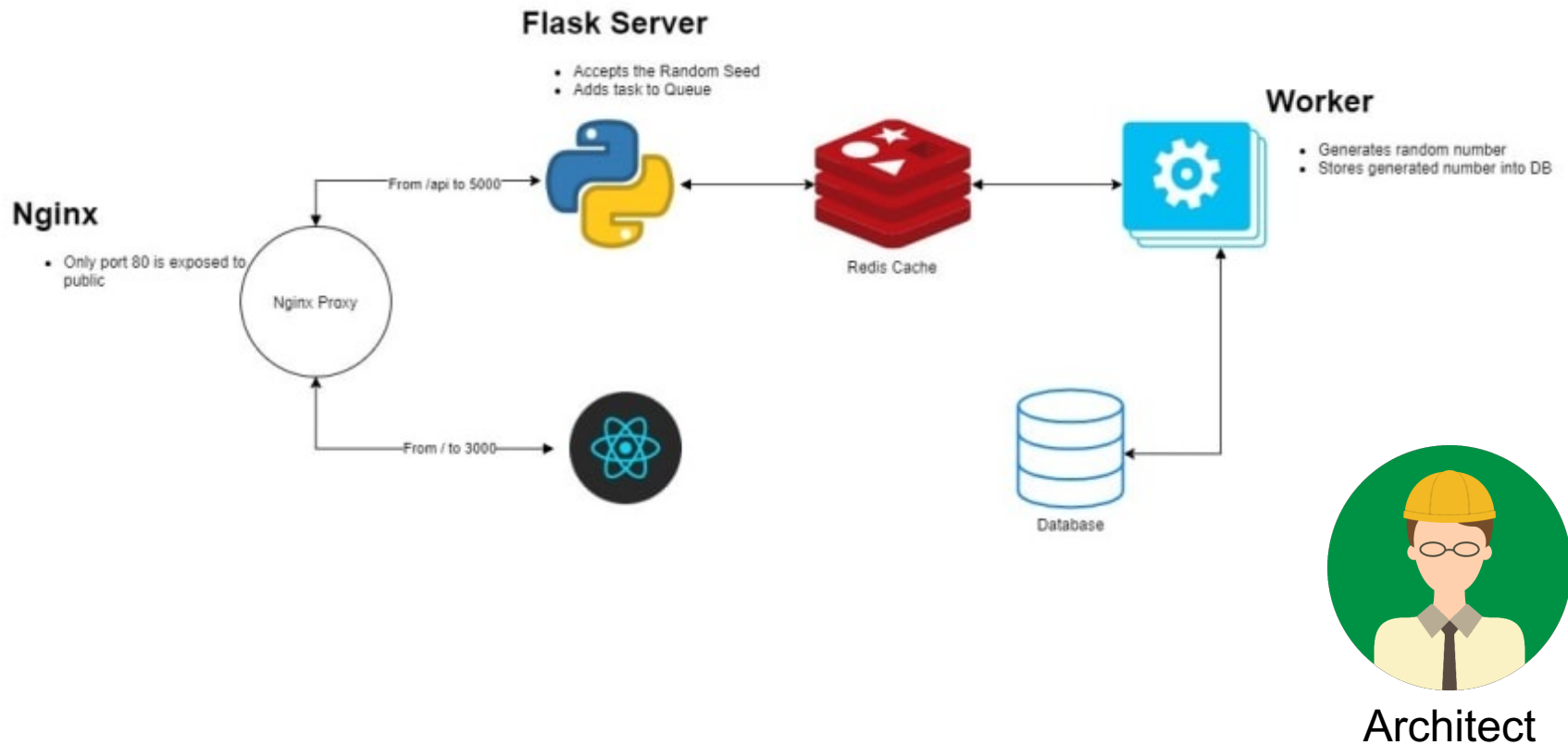
de5a683 on Nov 18, 2020  1 commit

Initial commit 3 hours ago

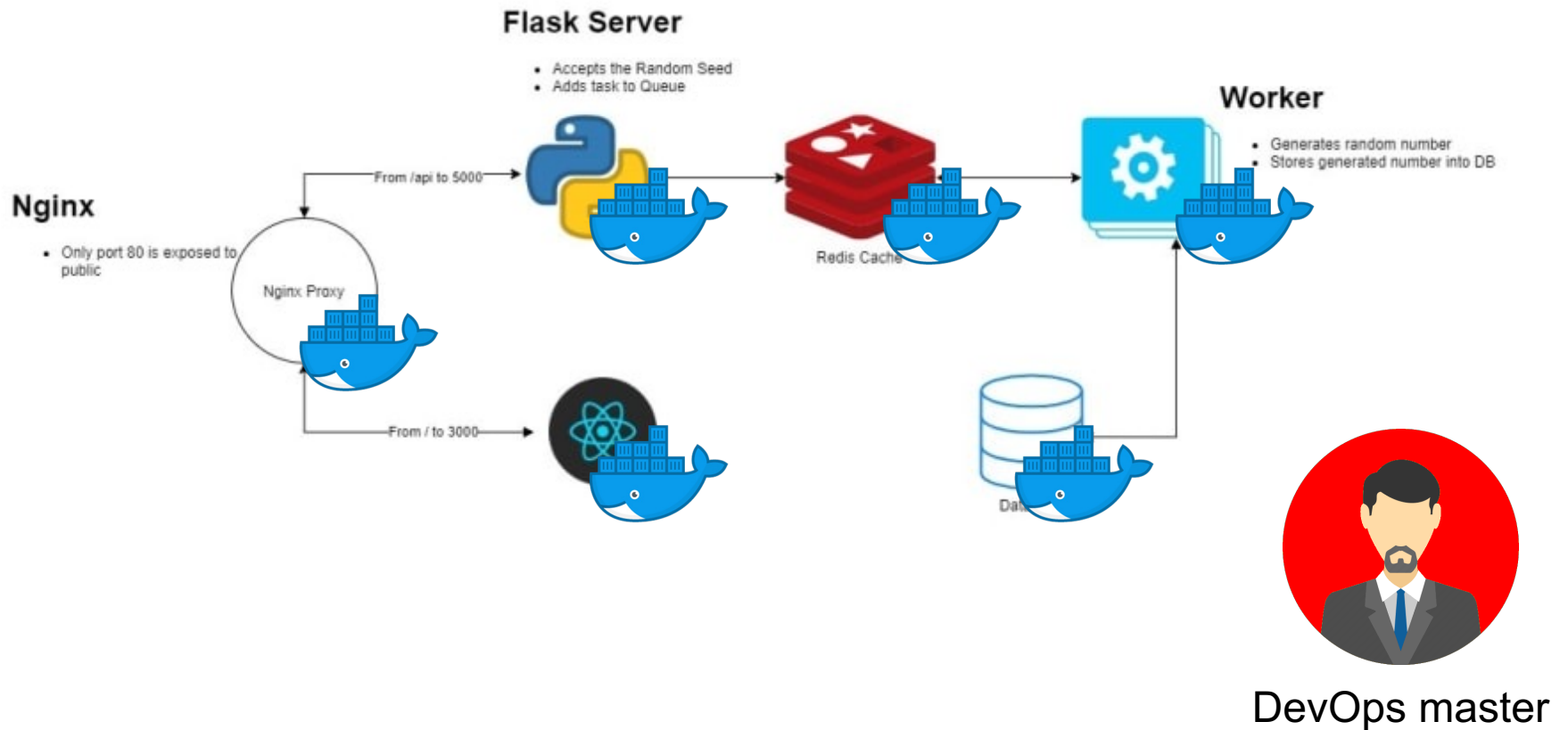


## RandomNumberGenerator

# System architecture



# System architecture



# Docker Compose

---

```
services:
  proxy:
    container_name: proxy
    build:
      ...
    ports:
      - 80:80

  database:
    container_name: database
    build: database/.
    volumes:
      - ./database/db_data:/var/lib/postgresql
      ...

  client:
    container_name: client
    build:
      ...
    environment:
      ...
```

```
api:
  container_name: api
  build:
    ...
  volumes:
    - ./api:/app

worker:
  container_name: worker
  build:
    ...

redis:
  container_name: redis
  build:
    ...
```

# Docker Compose Best Practices

---

- ❖ Use a file for variables (**.env**)
  - Exposed port numbers
  - Volumes' paths
- ❖ Use docker health checks
  - To coordinate the right timings to run each container automatically
- ❖ Create a custom network
  - Avoid problems when deployed in different hosts
- ❖ Do not expose unnecessary ports