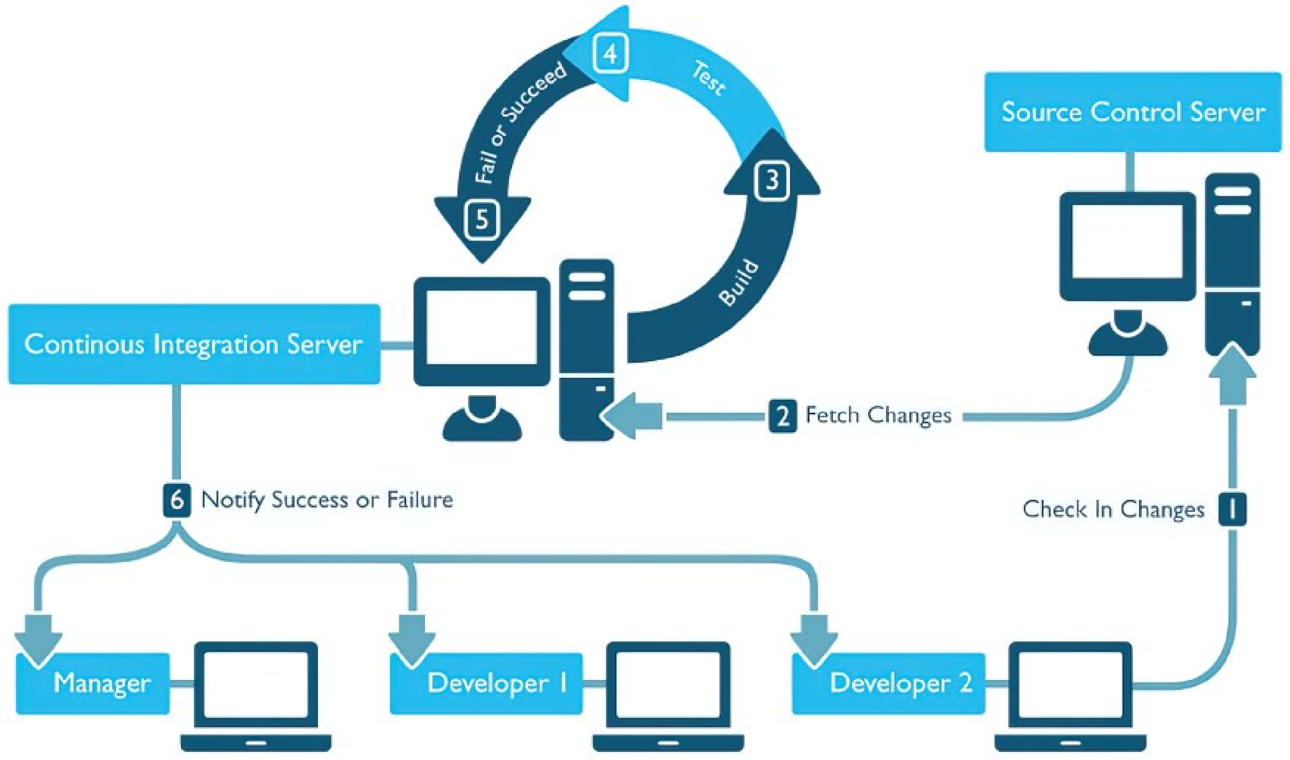
# CONTINUOUS INTEGRATION

RAFAEL DIREITO
INSTITUTO DE TELECOMUNICAÇÕES - AVEIRO

## CONTINUOUS INTEGRATION – WHAT IS IT?

- **Software development practice** that **allows** that whenever a new code change is committed to a code repository, an **automated build** is triggered:
  - This **build** will be **validated against several requirements and tests**, which will be performed **automatically**;
  - **After** the building and testing phase, **the code developer**s will be **informed if the newly committed code follows the standards needed** to follow through to the integration phase.
- Introduced in 1991, by Grady Booch.



## CONTINUOUS INTEGRATION – (SOME) BENEFITS

Early detection of bugs, which simplifies the process of fixing them;

If the developers wish to roll back to a previous version, less code will be lost;

The current build is constantly available for testing, demo, or release purposes;

The process of continuously integrating new code leads the developers to create modular and less complex code;

↓

Faster releases;

Increase in customer satisfaction;

Cost reduction.

## CONTINUOUS INTEGRATION – CHALLENGES AND DIFFICULTIES

- Continuous Integration **abruptly changes the Software Development Lifecycle**
- This leads to some **challenges**, that may be motivated by:

Lack of investment in CI tools and during the implementation of this methodology;

Developers' resistance to the adoption of the CI paradigm;

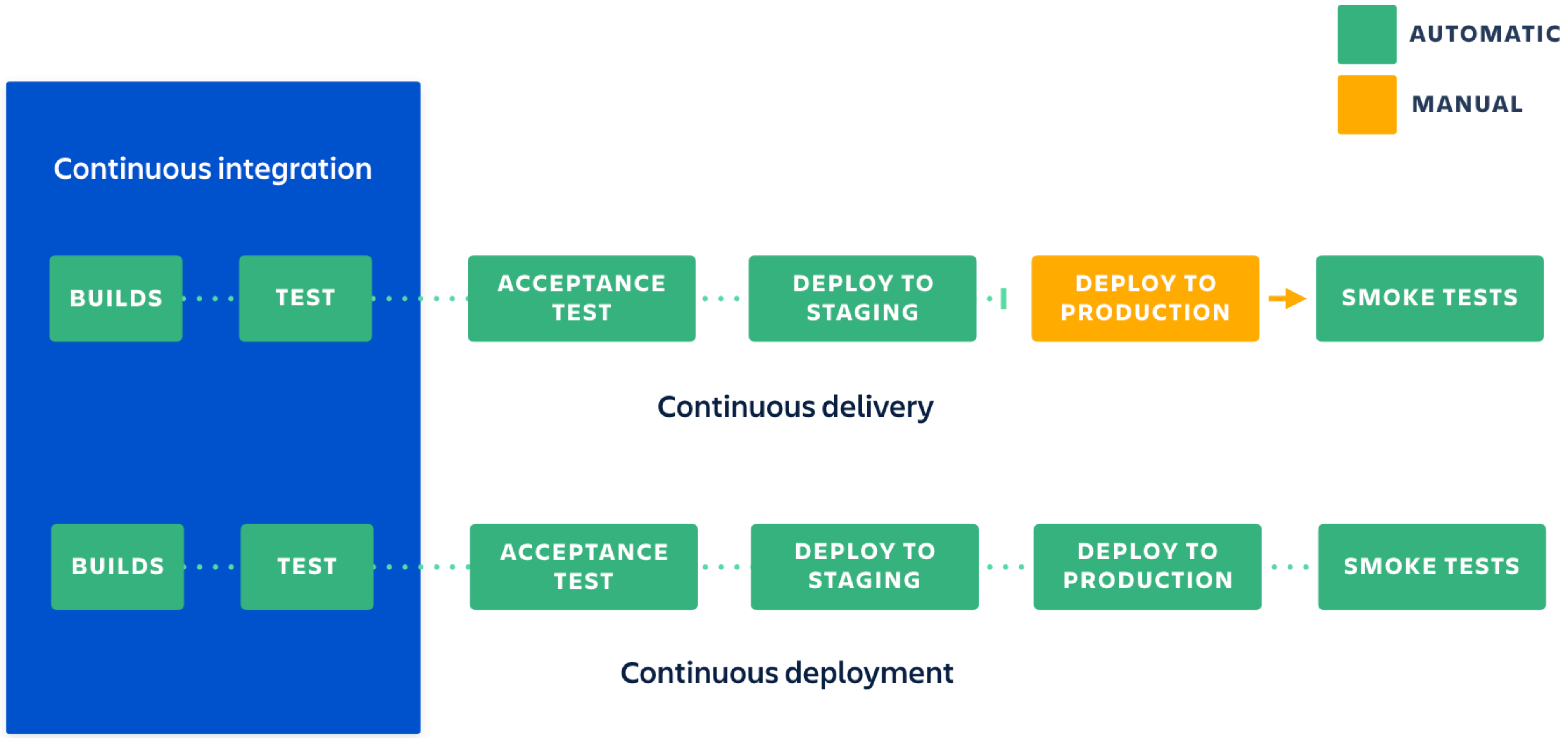
Difficulty in changing old organizational culture and policies;

Lack of proper testing strategies.

## CONTINUOUS (X)

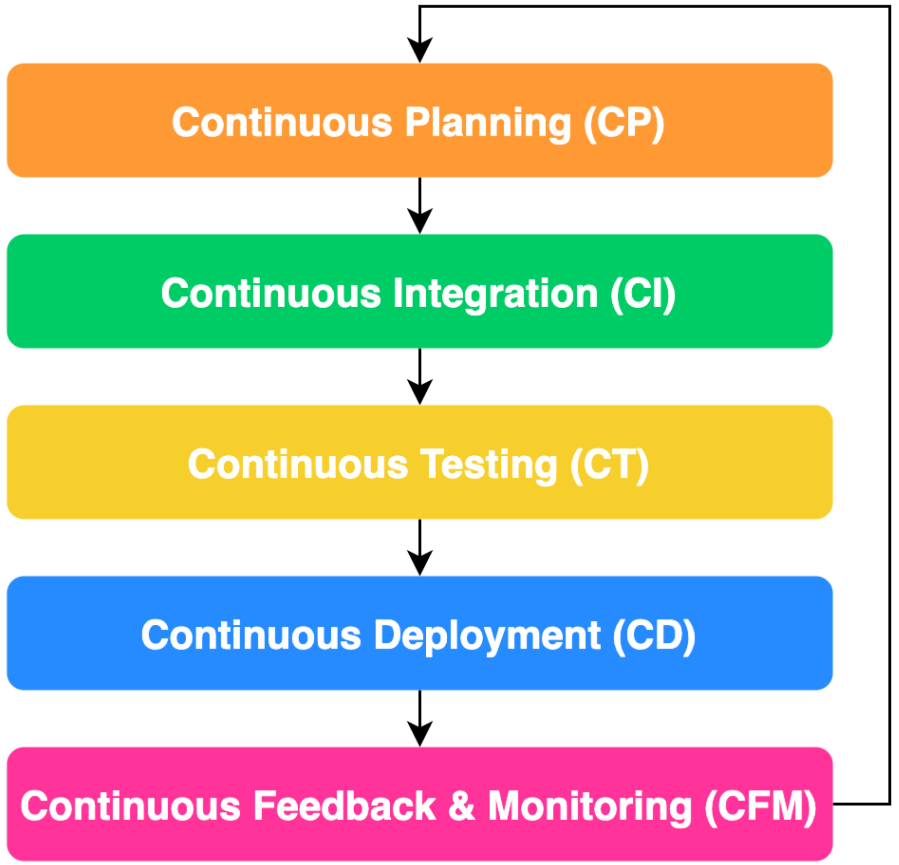
| | | |
|---|---|---|
| CONTINUOUS | PLANNING | Constantly assesses events, both internal and external, to help drive decisions and adjust the plan |
| | INTEGRATION | |
| | TESTING | Automated execution of tests as part of the software delivery pipeline in order to obtain feedback on the business risks associated with a software release |
| | DELIVERY | Extension of continuous integration since it automatically deploys all code changes to a TESTING environment after the build stage |
| | DEPLOYMENT | Extension of continuous integration since it automatically deploys all code changes to a PRODUCTION environment after the build stage |
| | MONITORING | Ability to maintain ongoing awareness of information security, vulnerabilities, and threats to facilitate risk-based decision making. |

# CONTINUOUS DELIVERY VS CONTINUOUS DEPLOYMENT

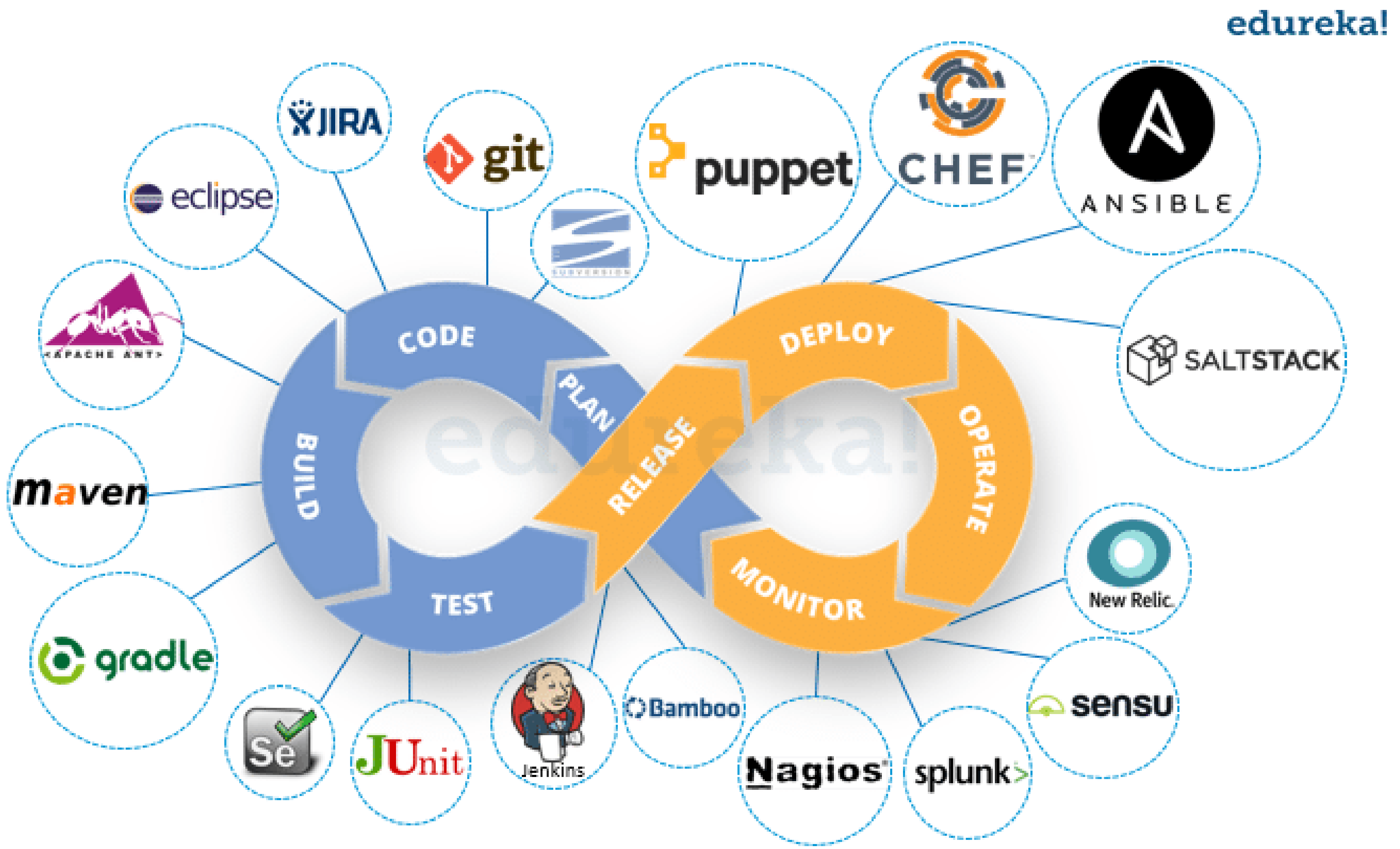


# DEVOPS

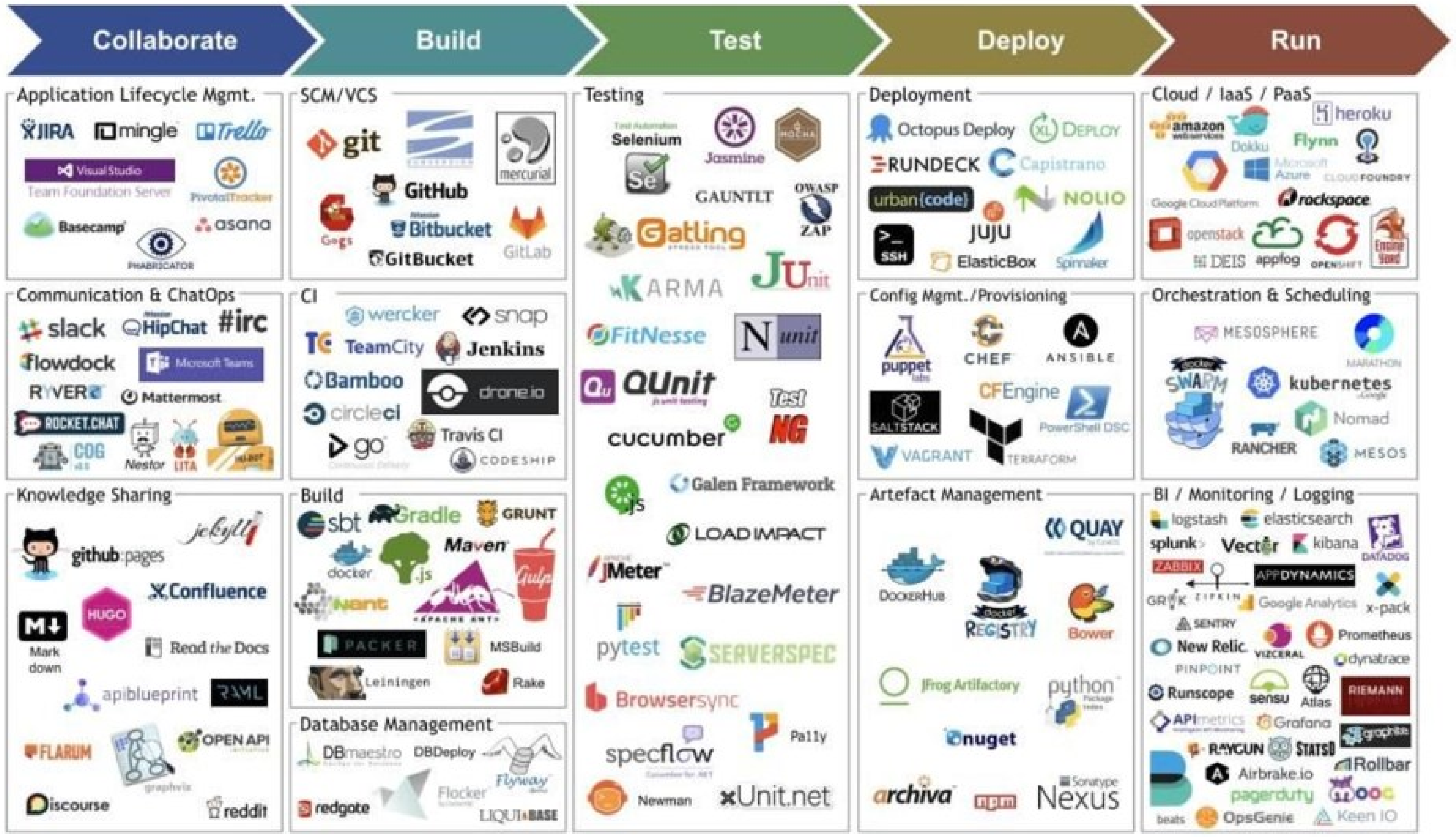
- Firstly introduced in 2009;
- Derived from the **combination of Development (Dev) and Operations (Ops)** ;
- Enables better collaboration between the development and operation teams and recognizes the need to continuously integrate software development with operational deployment, thus extending Agile;
- Defined by 4 dimensions:
  - Collaboration
  - Automation
  - Measurement
  - Monitoring


Continuous Planning (CP)
Continuous Integration (CI)
Continuous Testing (CT)
Continuous Deployment (CD)
Continuous Feedback & Monitoring (CFM)

# DEVOPS LIFECYCLE



# DEVOPS TOOLS



## CONTINUOUS INTEGRATION TOOLS - ANALYSIS

| Tool | Interaction via API | Possibility of self-hosting? | Requires SCM repository integration? | Customization | Cost |
|---|---|---|---|---|---|
| **Jenkins** | Yes | Yes | No | Wide variety of plugins | Free |
| **Circle CI** | Yes | Yes | Yes | Does not offer plugins, but claims to have all needed customizations built-in | Free on the cloud, but with limited builds (Freemium) |
| **TeamCity** | Yes | Yes | No | Wide variety of plugins | The free self-hosted version only allows for 100 different build configurations (Freemium) |

## CONTINUOUS INTEGRATION TOOLS - ANALYSIS

| Tool | Interaction via API | Possibility of self-hosting? | Requires SCM repository integration? | Customization | Cost |
|---|---|---|---|---|---|
| **Gitlab CI** | Yes | Yes | Yes | Offers a limited number of plugins | Free on the cloud, but with time restrictions and free self-hosted option, but with feature restrictions (Freemium) |
| **Travis CI** | Yes | Yes | Yes | Offers a limited number of plugins | Free but with build restrictions (Freemium) |
| **Bamboo** | Yes | Yes | Yes | Offers a limited number of plugins | No free options |

## CONTINUOUS INTEGRATION TOOLS - ANALYSIS

| Tool | Interaction via API | Possibility of self-hosting? | Requires SCM repository integration? | Customization | Cost |
|---|---|---|---|---|---|
| **Drone CI** | Yes | Yes | No | Wide variety of plugins | Free on the cloud, but with some restrictions. Offers a free version for on-premises installations but with limited features (Freemium) |
| **GitHub Actions** | Yes | Self-hosted runners | Yes | Wide variety of plugins | Free but with build restrictions (Freemium) |

## CI/CD PIPELINE – BEST PRACTICES

- **Write up the current development process therefore**, you can know the procedures that require to change and one that can be easily automated.
- **Start off with a small proof of project** before going ahead and complete whole development process at once.
- Set up a pipeline with **more than one stage in which fast fundamental tests run first**.
- **Start each workflow from the same, clean, and isolated environment**.
- Run open source tools that **cover everything from code style to security scanning**.
- **Peer code review** each pull request to solve a problem in a collaborative manner.
- You have to **define success metrics before you start the transition to CD automation**. This will help you to consistently analyze your software, developing progress help refining where needed.

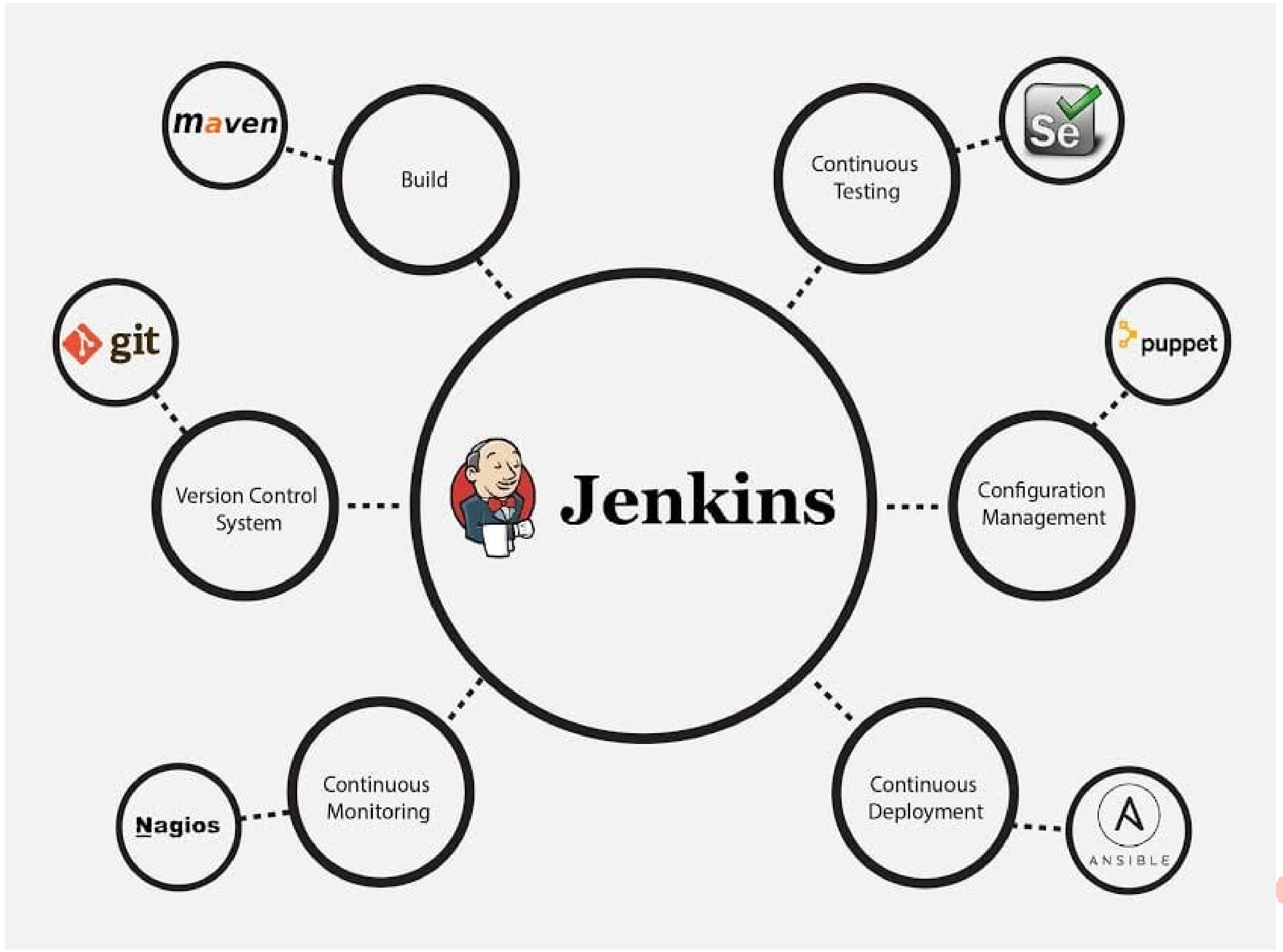
Source: https://www.guru99.com/ci-cd-pipeline.html

## JENKINS

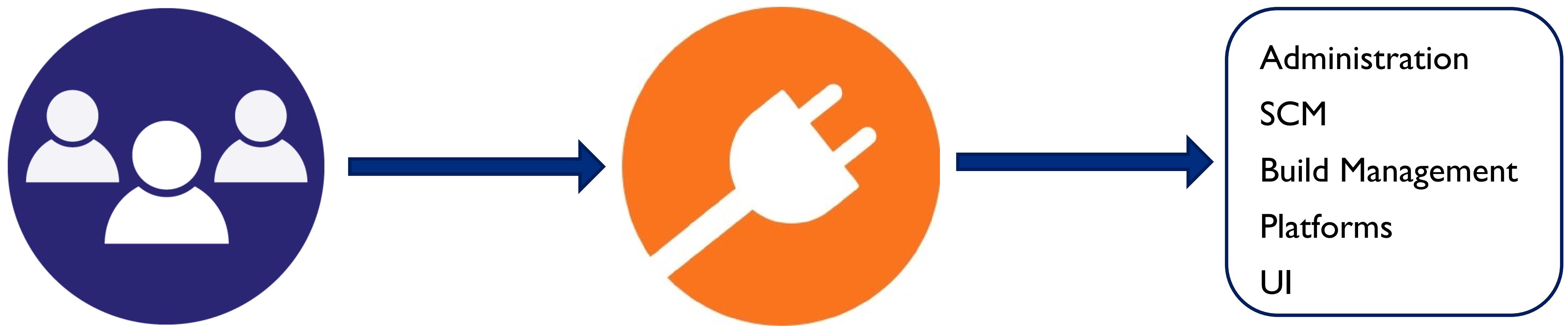
## JENKINS

- Jenkins is an **open-source automation tool** written in Java with plugins built for Continuous Integration purposes.
- Jenkins is used to **build and test your software projects continuously** making it **easier** for developers **to integrate** changes to the project, and making it easier for users to obtain a fresh build. It also allows you **to continuously deliver your software by integrating with a large number of testing and deployment technologies**.

Source: https://www.edureka.co/blog/what-is-jenkins/



## JENKINS COMMUNITY



**Community**
Extensive and Intervenient

> 1600 Plugins

## JENKINS WIDE ADOPTION

Great Features

Possibility for On-Premises Deployment

Excellent Support from Jenkins Community

## HOW TO INTERACT WITH JENKINS

Jenkins API

Jenkins CLI

Jenkins UI

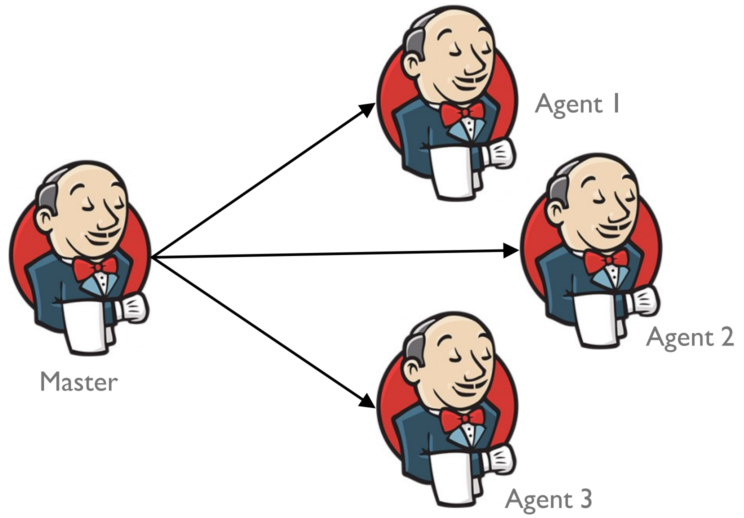
Jenkins Wrappers

## HOW TO INSTALL JENKINS

**Standalone**

Small Projects

**Cluster**



Larger Projects

## DEMO

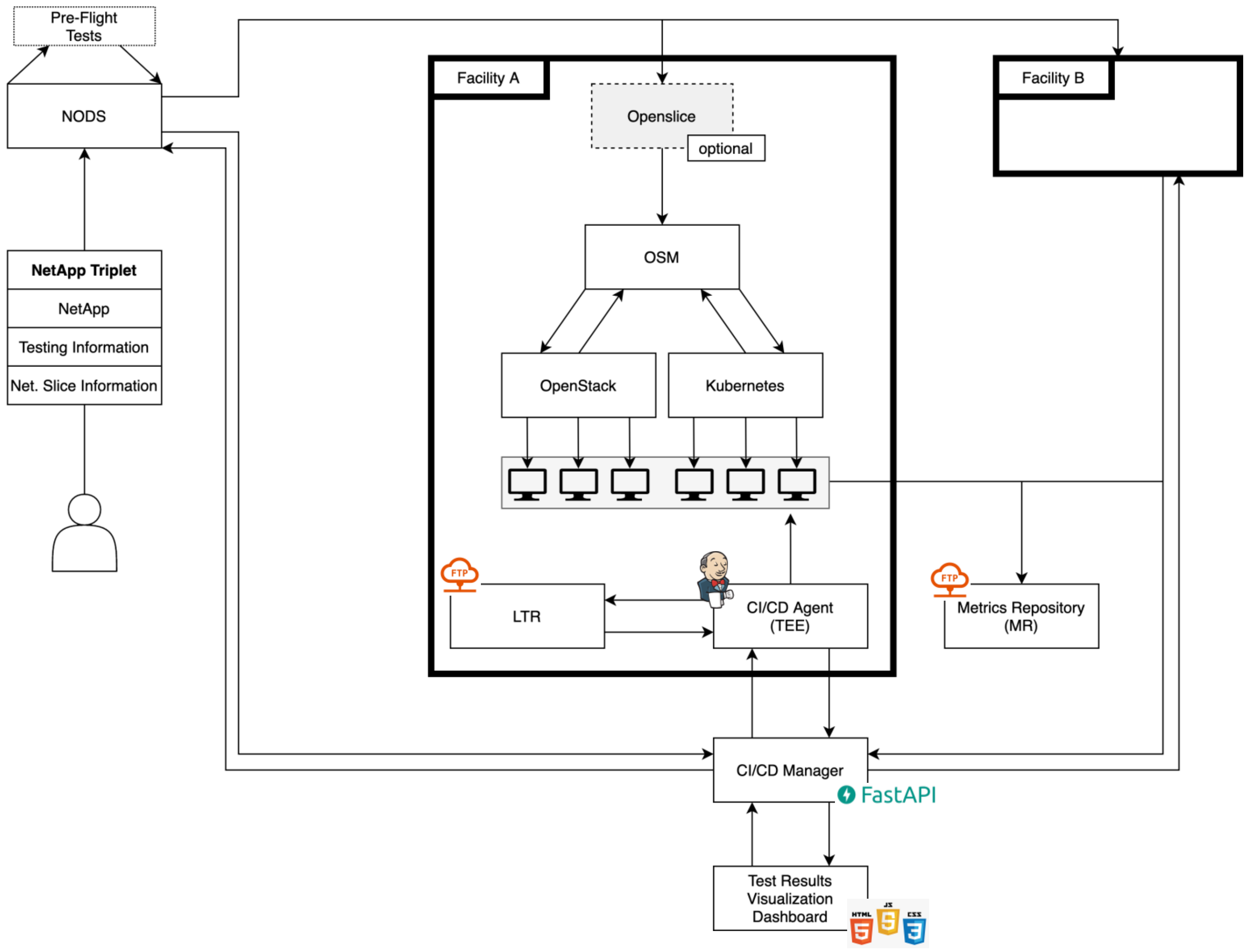
JENKINS – TQS HOMEWORK 2017

HTTPS://GITHUB.COM/RAFAEL-DIREITO/SEMINARS

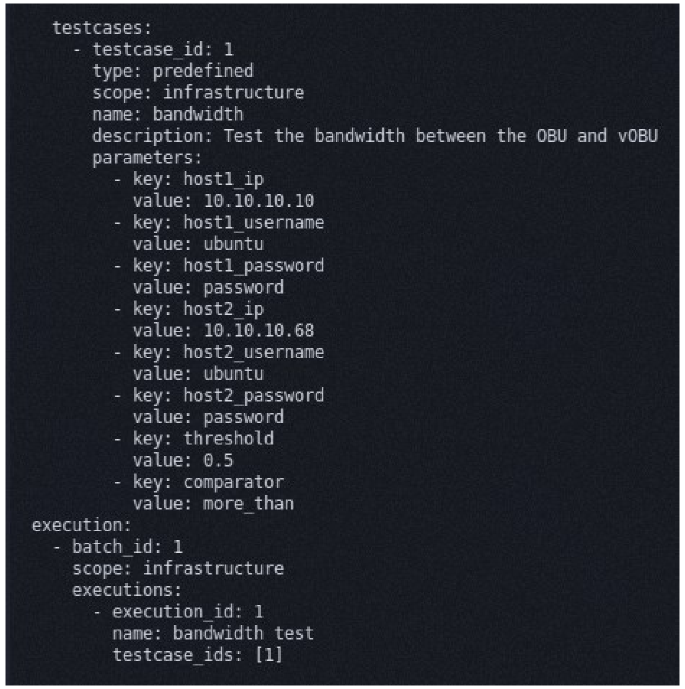
# CI/CD PIPELINE REAL WORLD USE CASE

5GASP H2020 PROJECT

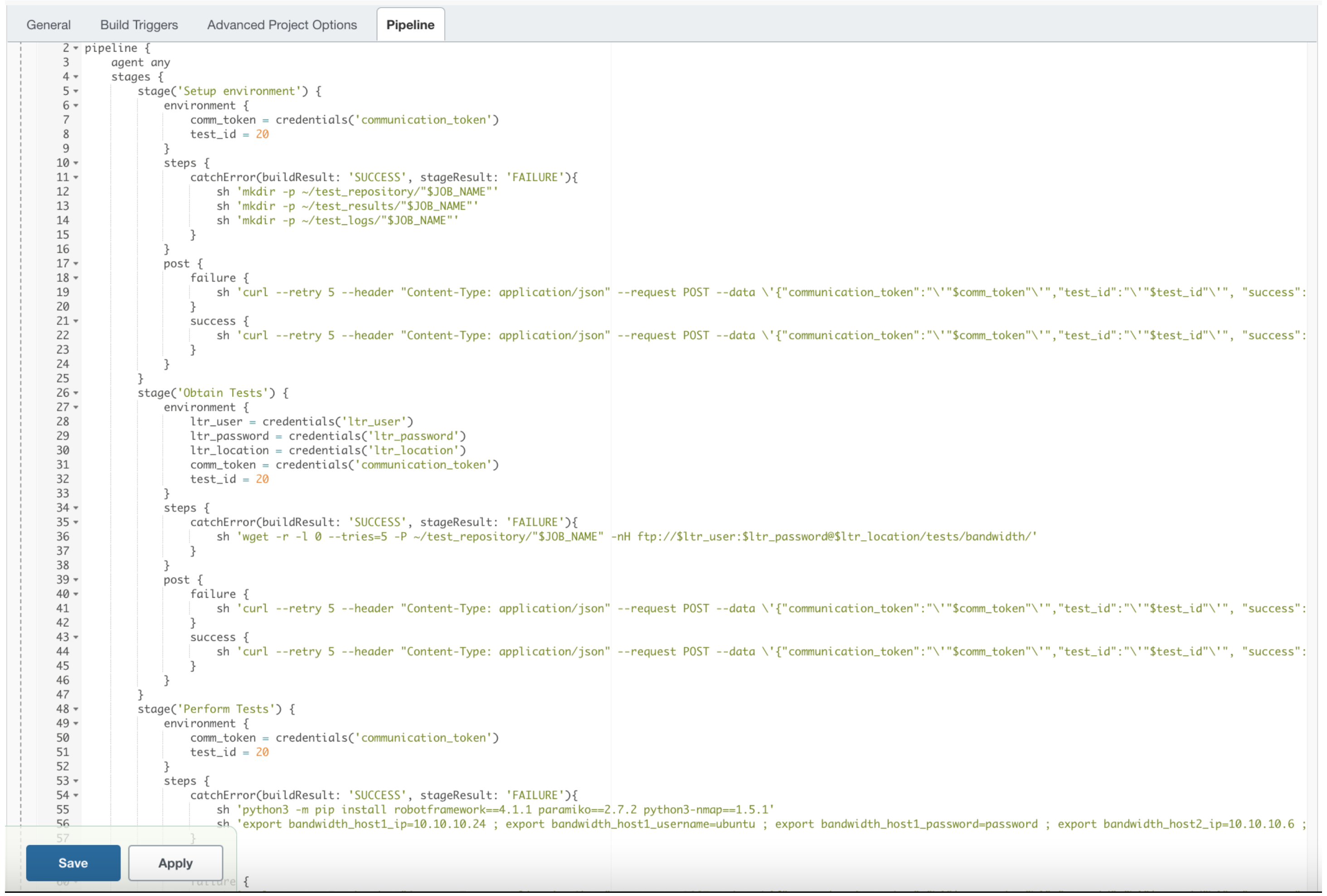
# 5GASP - CI PIPELINE



# ACHIEVED RESULTS (MARCH, 2022)

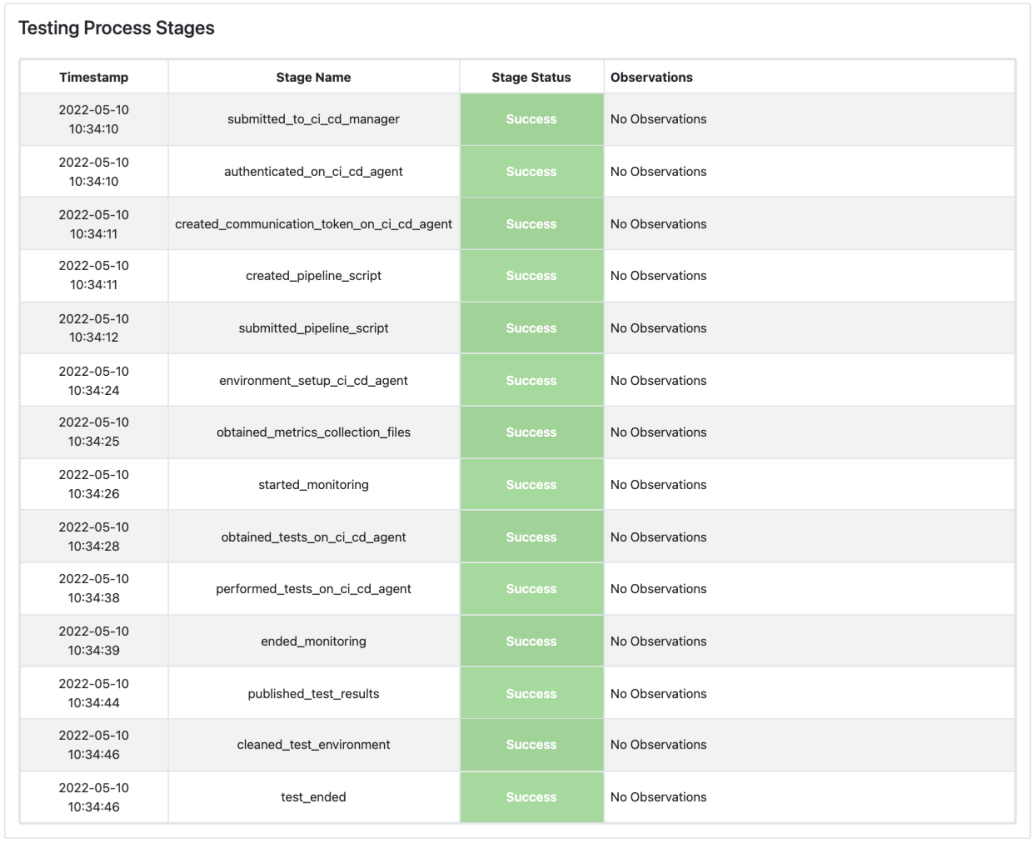


```
    testcases:
      - testcase_id: 1
        type: predefined
        scope: infrastructure
        name: bandwidth
        description: Test the bandwidth between the OBU and vOBU
        parameters:
          - key: host1_ip
            value: 10.10.10.10
          - key: host1_username
            value: ubuntu
          - key: host1_password
            value: password
          - key: host2_ip
            value: 10.10.10.68
          - key: host2_username
            value: ubuntu
          - key: host2_password
            value: password
          - key: threshold
            value: 0.5
          - key: comparator
            value: more_than
  execution:
    - batch_id: 1
      scope: infrastructure
      executions:
        - execution_id: 1
          name: bandwidth test
          testcase_ids: [1]
```



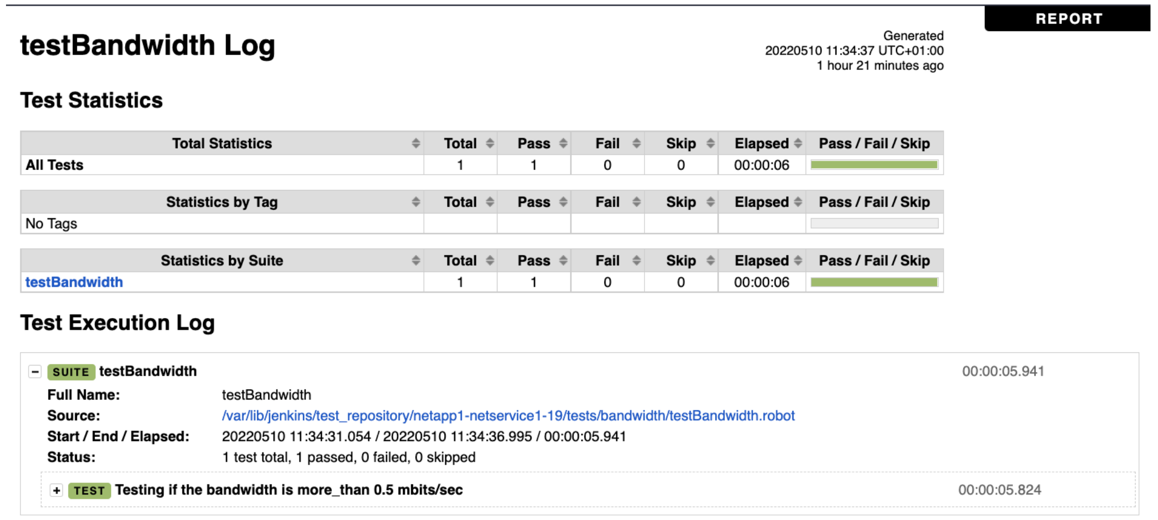
```
pipeline {
    agent any
    stages {
        stage('Setup environment') {
            environment {
                comm_token = credentials('communication_token')
                test_id = 20
            }
            steps {
                catchError(buildResult: 'SUCCESS', stageResult: 'FAILURE'){
                    sh 'mkdir -p ~/test_repository/"$JOB_NAME"'
                    sh 'mkdir -p ~/test_results/"$JOB_NAME"'
                    sh 'mkdir -p ~/test_logs/"$JOB_NAME"'
                }
            }
            post {
                failure {
                    sh 'curl --retry 5 --header "Content-Type: application/json" --request POST --data \'{"communication_token":"\'"$comm_token"\'","test_id":"\'"$test_id"\'", "success":
                }
                success {
                    sh 'curl --retry 5 --header "Content-Type: application/json" --request POST --data \'{"communication_token":"\'"$comm_token"\'","test_id":"\'"$test_id"\'", "success":
                }
            }
        }
        stage('Obtain Tests') {
            environment {
                ltr_user = credentials('ltr_user')
                ltr_password = credentials('ltr_password')
                ltr_location = credentials('ltr_location')
                comm_token = credentials('communication_token')
                test_id = 20
            }
            steps {
                catchError(buildResult: 'SUCCESS', stageResult: 'FAILURE'){
                    sh 'wget -r -l 0 --tries=5 -P ~/test_repository/"$JOB_NAME" -nH ftp://$ltr_user:$ltr_password@$ltr_location/tests/bandwidth/'
                }
            }
            post {
                failure {
                    sh 'curl --retry 5 --header "Content-Type: application/json" --request POST --data \'{"communication_token":"\'"$comm_token"\'","test_id":"\'"$test_id"\'", "success":
                }
                success {
                    sh 'curl --retry 5 --header "Content-Type: application/json" --request POST --data \'{"communication_token":"\'"$comm_token"\'","test_id":"\'"$test_id"\'", "success":
                }
            }
        }
        stage('Perform Tests') {
            environment {
                comm_token = credentials('communication_token')
                test_id = 20
            }
            steps {
                catchError(buildResult: 'SUCCESS', stageResult: 'FAILURE'){
                    sh 'python3 -m pip install robotframework==4.1.1 paramiko==2.7.2 python3-nmap==1.5.1'
                    sh 'export bandwidth_host1_ip=10.10.10.24 ; export bandwidth_host1_username=ubuntu ; export bandwidth_host1_password=password ; export bandwidth_host2_ip=10.10.10.6 ;
                }
                failure {
```

# ACHIEVED RESULTS (MARCH, 2022)



Testing Process Stages

| Timestamp | Stage Name | Stage Status | Observations |
|---|---|---|---|
| 2022-05-10 10:34:10 | submitted_to_ci_cd_manager | Success | No Observations |
| 2022-05-10 10:34:10 | authenticated_on_ci_cd_agent | Success | No Observations |
| 2022-05-10 10:34:11 | created_communication_token_on_ci_cd_agent | Success | No Observations |
| 2022-05-10 10:34:11 | created_pipeline_script | Success | No Observations |
| 2022-05-10 10:34:12 | submitted_pipeline_script | Success | No Observations |
| 2022-05-10 10:34:24 | environment_setup_ci_cd_agent | Success | No Observations |
| 2022-05-10 10:34:25 | obtained_metrics_collection_files | Success | No Observations |
| 2022-05-10 10:34:26 | started_monitoring | Success | No Observations |
| 2022-05-10 10:34:28 | obtained_tests_on_ci_cd_agent | Success | No Observations |
| 2022-05-10 10:34:38 | performed_tests_on_ci_cd_agent | Success | No Observations |
| 2022-05-10 10:34:39 | ended_monitoring | Success | No Observations |
| 2022-05-10 10:34:44 | published_test_results | Success | No Observations |
| 2022-05-10 10:34:46 | cleaned_test_environment | Success | No Observations |
| 2022-05-10 10:34:46 | test_ended | Success | No Observations |



REPORT

testBandwidth Log

Generated
20220510 11:34:37 UTC+01:00
1 hour 21 minutes ago

Test Statistics

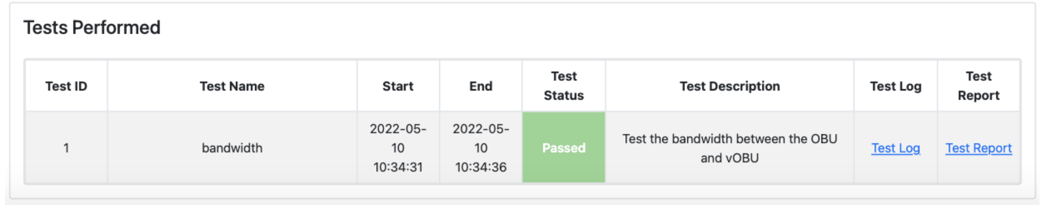
| Total Statistics | Total | Pass | Fail | Skip | Elapsed | Pass / Fail / Skip |
|---|---|---|---|---|---|---|
| All Tests | 1 | 1 | 0 | 0 | 00:00:06 | 🟩 |

| Statistics by Tag | Total | Pass | Fail | Skip | Elapsed | Pass / Fail / Skip |
|---|---|---|---|---|---|---|
| No Tags | | | | | | |

| Statistics by Suite | Total | Pass | Fail | Skip | Elapsed | Pass / Fail / Skip |
|---|---|---|---|---|---|---|
| testBandwidth | 1 | 1 | 0 | 0 | 00:00:06 | 🟩 |

Test Execution Log

```
− SUITE testBandwidth                                                        00:00:05.941
  Full Name:           testBandwidth
  Source:              /var/lib/jenkins/test_repository/netapp1-netservice1-19/tests/bandwidth/testBandwidth.robot
  Start / End / Elapsed: 20220510 11:34:31.054 / 20220510 11:34:36.995 / 00:00:05.941
  Status:              1 test total, 1 passed, 0 failed, 0 skipped
  + TEST Testing if the bandwidth is more_than 0.5 mbits/sec                 00:00:05.824
```



Tests Performed

| Test ID | Test Name | Start | End | Test Status | Test Description | Test Log | Test Report |
|---|---|---|---|---|---|---|---|
| 1 | bandwidth | 2022-05-10 10:34:31 | 2022-05-10 10:34:36 | Passed | Test the bandwidth between the OBU and vOBU | Test Log | Test Report |

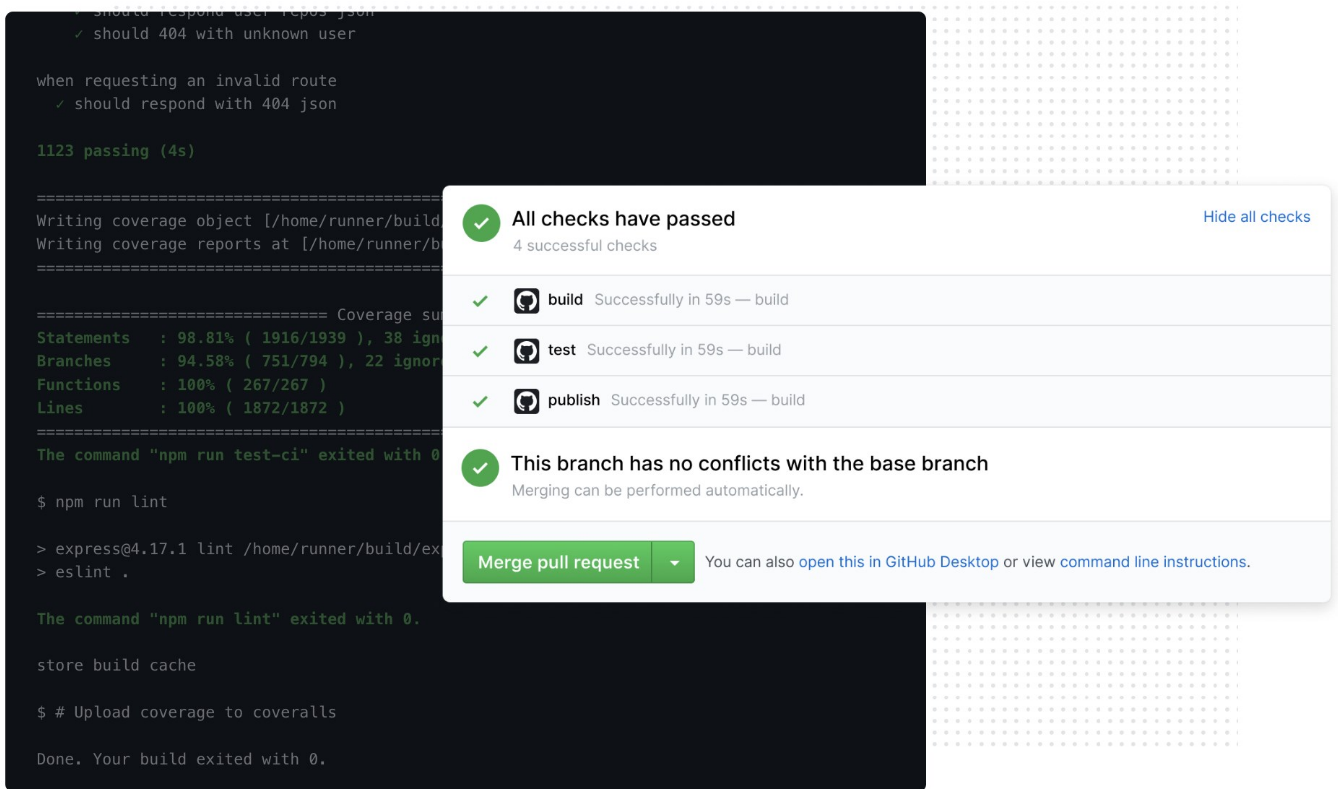
## DEMO

5GASP H2020 PROJECT

## GITHUB ACTIONS

# GITHUB ACTIONS

- GitHub Actions makes it easy to **automate all your software workflows**, now with world-class CI/CD;
- With GitHub Actions you can build, test, and deploy your code right from GitHub;
- GitHub Actions are defined using workflows, defined in YAML files.



```
    ✓ should 404 with unknown user

when requesting an invalid route
  ✓ should respond with 404 json

1123 passing (4s)

==============================================
Writing coverage object [/home/runner/build
Writing coverage reports at [/home/runner/b
==============================================

=============================== Coverage su
Statements   : 98.81% ( 1916/1939 ), 38 ign
Branches     : 94.58% ( 751/794 ), 22 ignor
Functions    : 100% ( 267/267 )
Lines        : 100% ( 1872/1872 )
==============================================
The command "npm run test-ci" exited with 0

$ npm run lint

> express@4.17.1 lint /home/runner/build/ex
> eslint .

The command "npm run lint" exited with 0.

store build cache

$ # Upload coverage to coveralls

Done. Your build exited with 0.
```

All checks have passed
4 successful checks

Hide all checks

✓ build Successfully in 59s — build

✓ test Successfully in 59s — build

✓ publish Successfully in 59s — build

This branch has no conflicts with the base branch
Merging can be performed automatically.

Merge pull request

You can also open this in GitHub Desktop or view command line instructions.

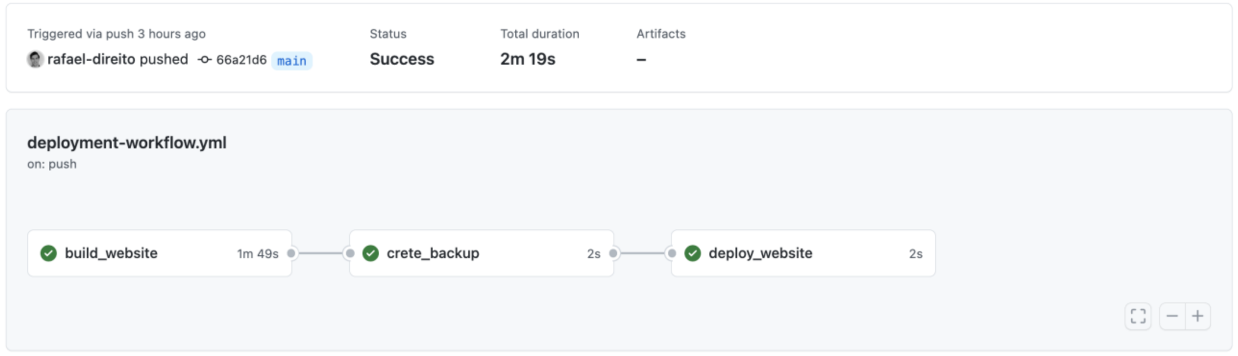
Source: https://github.com/features/actions

# GITHUB ACTIONS

```
name: GitHub Actions Deployment Workflow
on:
  push:
    branches: [ main ]
jobs:
  build_website:
    runs-on: self-hosted
    steps:
      - run: | # get inside the project and deal with the dependencies
          cd ${{ github.workspace }}/project-documentation
          if [ -e yarn.lock ]; then
            yarn install --frozen-lockfile
          elif [ -e package-lock.json ]; then
            npm ci
          else
            npm i
          fi
      - run: |
          cd ${{ github.workspace }}/project-documentation
          npm run build

  crete_backup:
    needs: build_website
    runs-on: self-hosted
    steps:
      - run: | # create a zip file with all the files and store it in the backups folder
          tar -czvf /var/www/docusaurus-website-backups/backup-$(date '+%Y-%m-%d_%H:%M:%S').tar.gz /var/www/docusaurus-website
      - run: | # remove all backups older than 7 days
          find /var/www/docusaurus-website-backups/ -maxdepth 1 -type f -mtime +7 -print | xargs /bin/rm -f

  deploy_website:
    needs: crete_backup
    runs-on: self-hosted
    steps:
      - run: | # delete old website files
          rm -rf /var/www/docusaurus-website/*
```



Triggered via push 3 hours ago
rafael-direito pushed 66a21d6 main

Status
Success

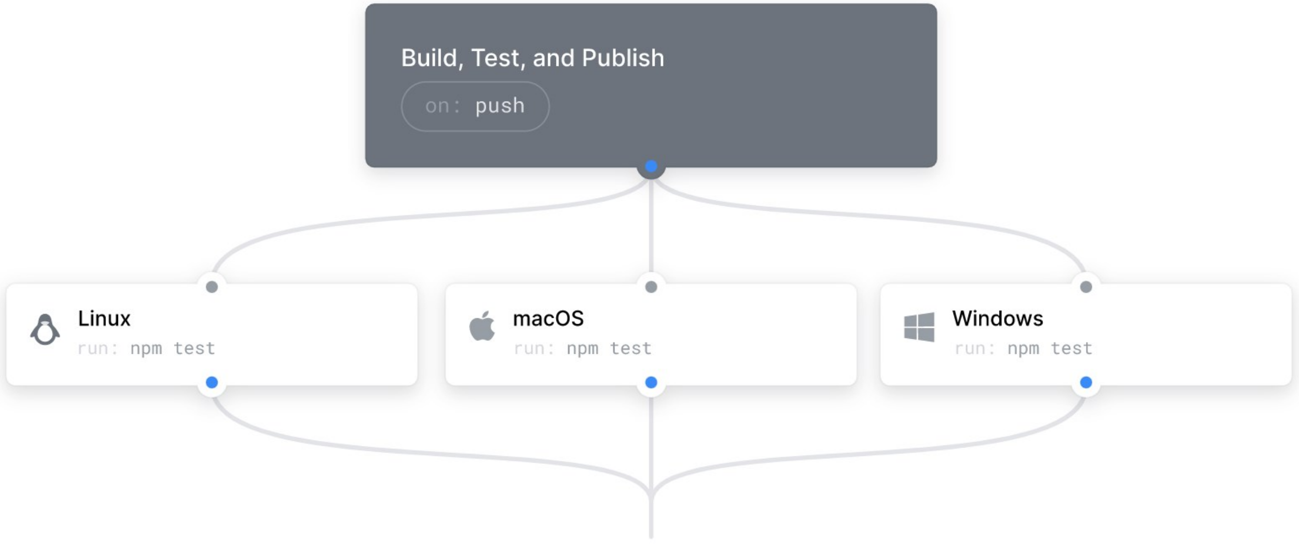
Total duration
2m 19s

Artifacts
–

deployment-workflow.yml
on: push

build_website 1m 49s
crete_backup 2s
deploy_website 2s

## GITHUB ACTIONS



**Linux, macOS, Windows, ARM, and containers**

Hosted runners for every major OS make it easy to build and test all your projects. Run directly on a VM or inside a container. Use your own VMs, in the cloud or on-prem, with self-hosted runners.

**Matrix builds**

Save time with matrix workflows that simultaneously test across multiple operating systems and versions of your runtime.

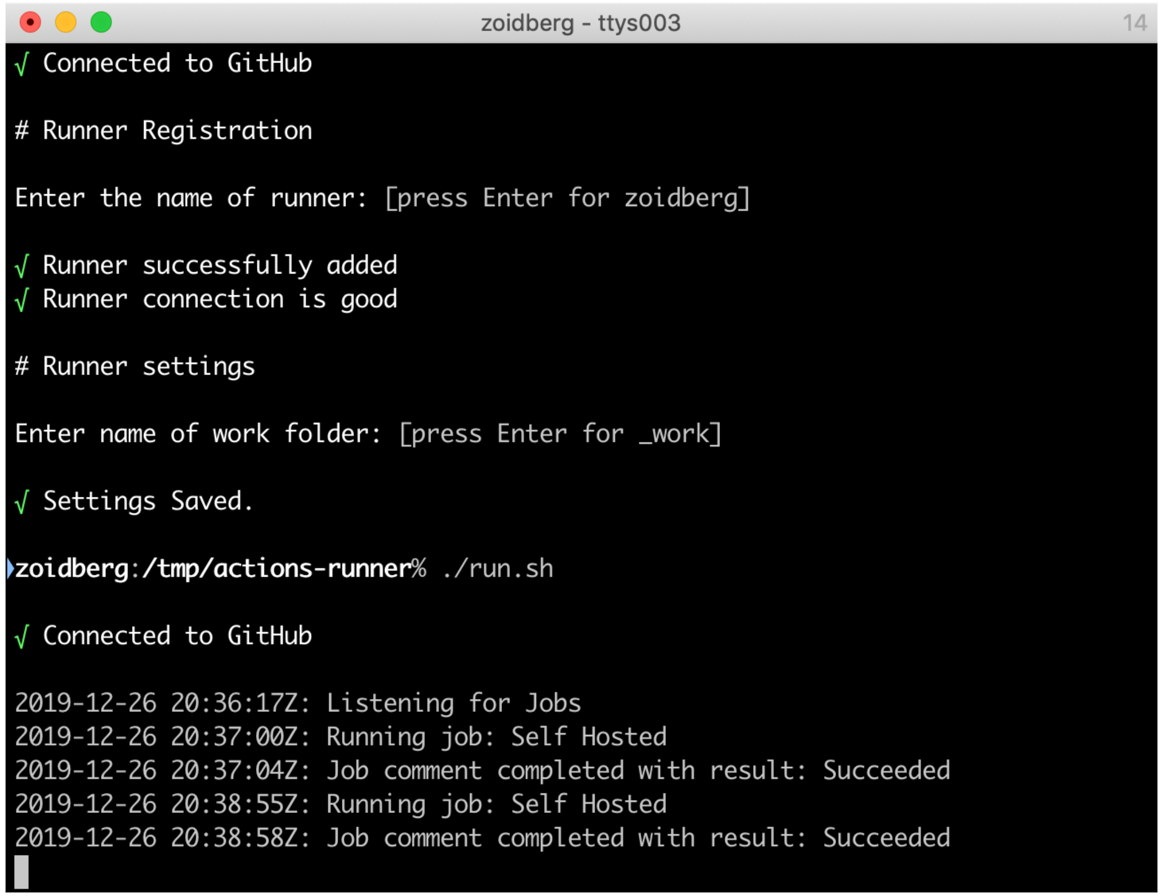
**Any language**

GitHub Actions supports Node.js, Python, Java, Ruby, PHP, Go, Rust, .NET, and more. Build, test, and deploy applications in your language of choice.

Source: https://github.com/features/actions

# GITHUB ACTIONS – SELF HOSTED RUNNERS

- Self-hosted runners offer m**ore control of hardware, operating system, and software tools** than GitHub-hosted runners provide;
- With self-hosted runners, you can **create custom hardware configurations that meet your needs with processing power or memory to run larger jobs, install software available on your local network, and choose an operating system** not offered by GitHub-hosted runners.



```
√ Connected to GitHub

# Runner Registration

Enter the name of runner: [press Enter for zoidberg]

√ Runner successfully added
√ Runner connection is good

# Runner settings

Enter name of work folder: [press Enter for _work]

√ Settings Saved.

zoidberg:/tmp/actions-runner% ./run.sh

√ Connected to GitHub

2019-12-26 20:36:17Z: Listening for Jobs
2019-12-26 20:37:00Z: Running job: Self Hosted
2019-12-26 20:37:04Z: Job comment completed with result: Succeeded
2019-12-26 20:38:55Z: Running job: Self Hosted
2019-12-26 20:38:58Z: Job comment completed with result: Succeeded
```

Source: https://docs.github.com/en/actions/hosting-your-own-runners/about-self-hosted-runners

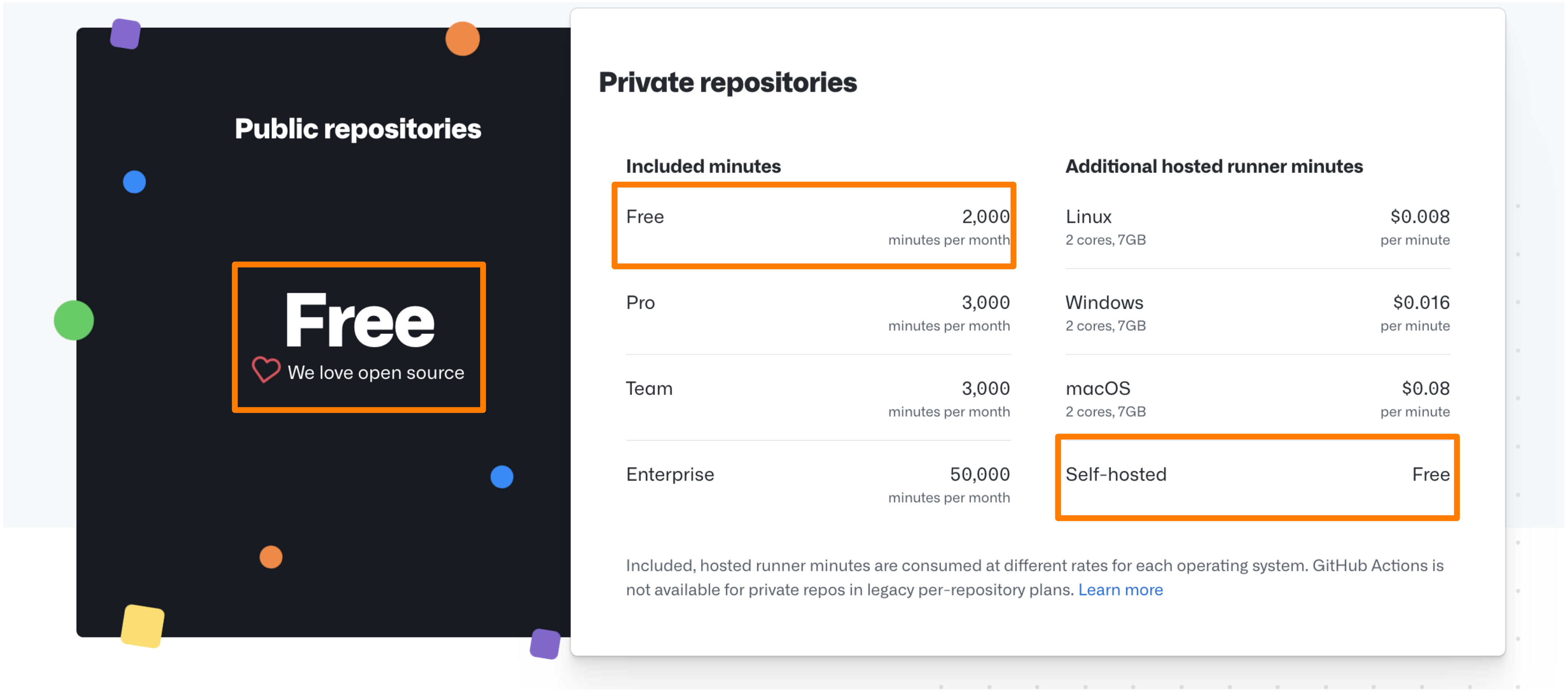
# GITHUB ACTIONS – SELF HOSTED RUNNERS

- **GitHub-hosted runners:**
  - Receive automatic updates for the operating system, preinstalled packages and tools, and the self-hosted runner application.
  - Are managed and maintained by GitHub.
  - Provide a clean instance for every job execution.
  - Use free minutes on your GitHub plan, with per-minute rates applied after surpassing the free minutes.

- **Self-hosted Runners**
  - Receive automatic updates for the self-hosted runner application only. You are responsible for updating the operating system and all other software.
  - Can use cloud services or local machines that you already pay for.
  - Are customizable to your hardware, operating system, software, and security requirements.
  - Don't need to have a clean instance for every job execution.
  - Are free to use with GitHub Actions.

Source: https://docs.github.com/en/actions/hosting-your-own-runners/about-self-hosted-runners

# GITHUB ACTIONS – COST



Public repositories

Free

We love open source

Private repositories

| Included minutes | | Additional hosted runner minutes | |
|---|---|---|---|
| Free | 2,000 minutes per month | Linux 2 cores, 7GB | $0.008 per minute |
| Pro | 3,000 minutes per month | Windows 2 cores, 7GB | $0.016 per minute |
| Team | 3,000 minutes per month | macOS 2 cores, 7GB | $0.08 per minute |
| Enterprise | 50,000 minutes per month | Self-hosted | Free |

Included, hosted runner minutes are consumed at different rates for each operating system. GitHub Actions is not available for private repos in legacy per-repository plans. Learn more

Source: https://github.com/features/actions

# DEMO

GITHUB ACTIONS

DEPLOYMENT OF A DOCUSAUROS WEBSITE

HTTPS://GITHUB.COM/RAFAEL-DIREITO/SEMINARS