Continuous Integration/Deployment with Gitlab CI

Who we are?

David Hahn, Dipl-Inf.(FH)
Senior Software Engineer
d.hahn@salt-and-pepper.eu

Denis Filimonov, B.Sc
Software Engineer
d.filimonov@salt-and-pepper.eu

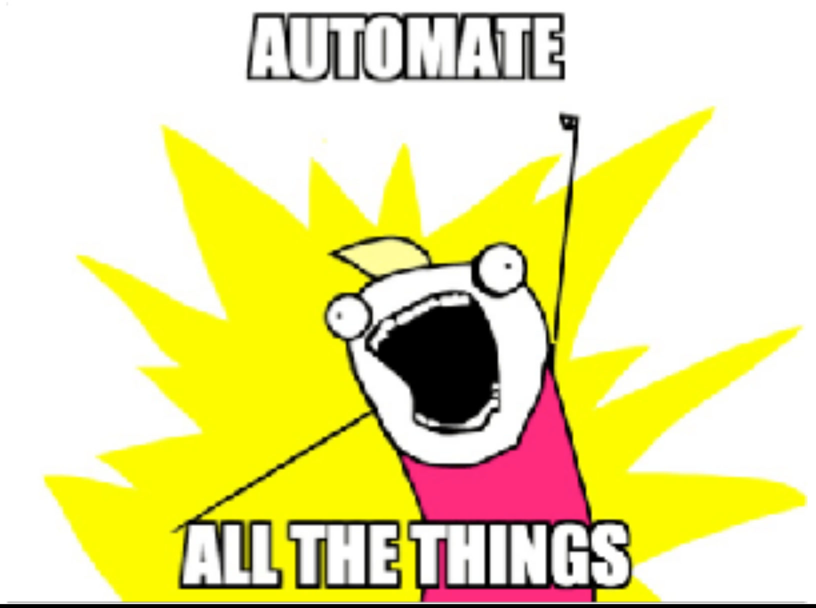
Agenda

- Introduction
    - Continuous Integration
    - Continuous Delivery and Deployment
    - About Gitlab
- Gitlab CI
    - About Gitlab CI
    - Stages and Pipelines
    - UI
    - Runners
    - CI as Code
- Show Cases
    - Node.js + React
    - Java + Angular
    - Electron

Continuous Integration
Definition

Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily - leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible.

Martin Fowler



Continuous Integration

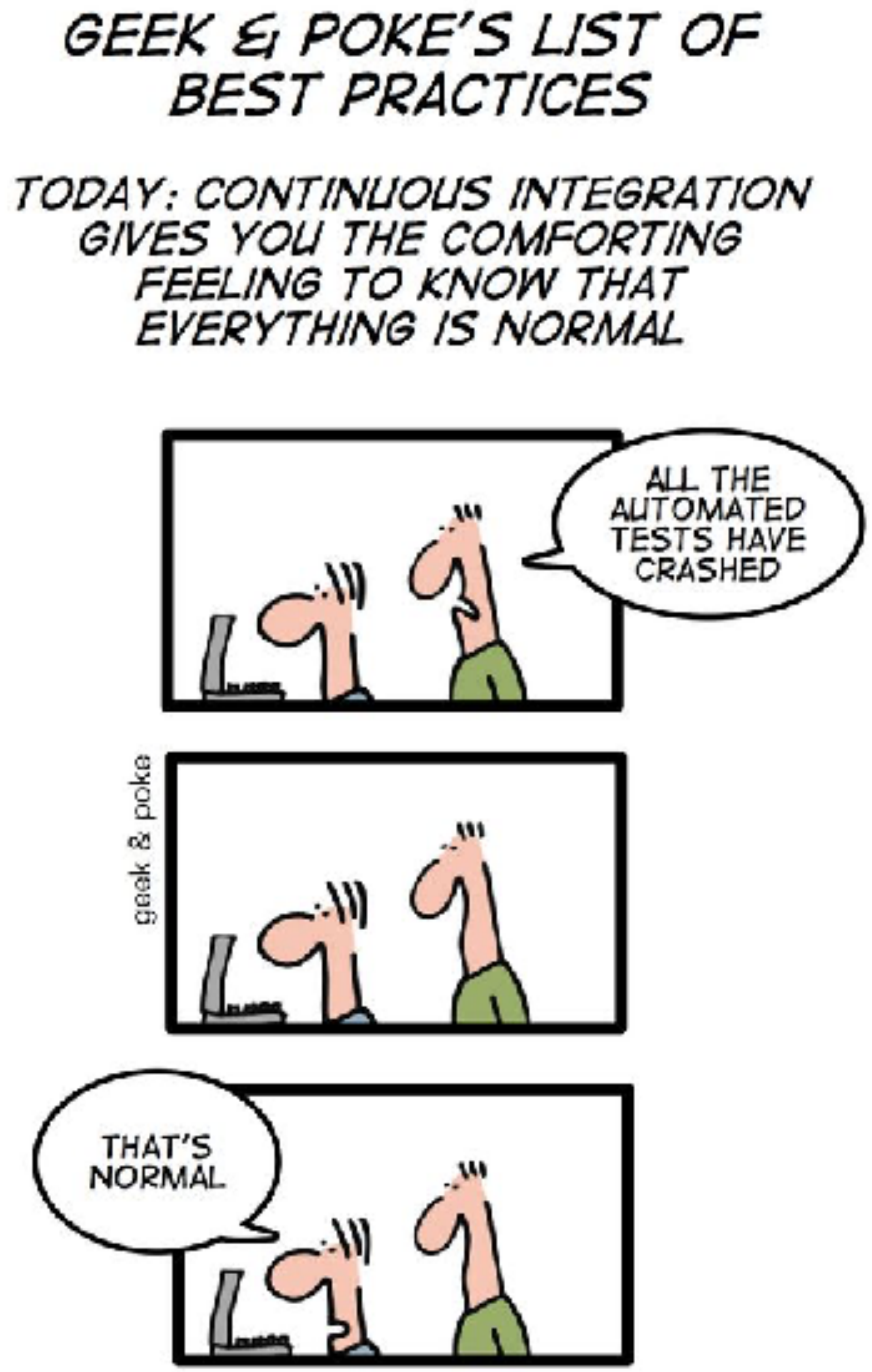
How and Why?

How:

- Maintain a single source repository
- Automate the build
- Make your build self-testing
- Keep the build fast
- Keep the build on the CI machine
- Test in a clone of production environment
- Make it easy for everyone to get the latest executable
- Make the process transparent for everyone

Why:

- Detect development problems earlier
- Reduce risks of cost, schedule and budget
- Find and remove bugs earlier
- Deliver new features and get user feedback more rapidly


GEEK & POKE'S LIST OF BEST PRACTICES
TODAY: CONTINUOUS INTEGRATION GIVES YOU THE COMFORTING FEELING TO KNOW THAT EVERYTHING IS NORMAL
ALL THE AUTOMATED TESTS HAVE CRASHED
geek & poke
THAT'S NORMAL

Continuous Delivery/Deployment

Definition

Continuous Delivery is a development discipline where you build software in such a way that the software can be released to production at any time. Continuous Deployment means that every change goes through the pipeline and automatically gets put into production, resulting in many production deployments every day.

Martin Fowler

Continuous Delivery/Deployment

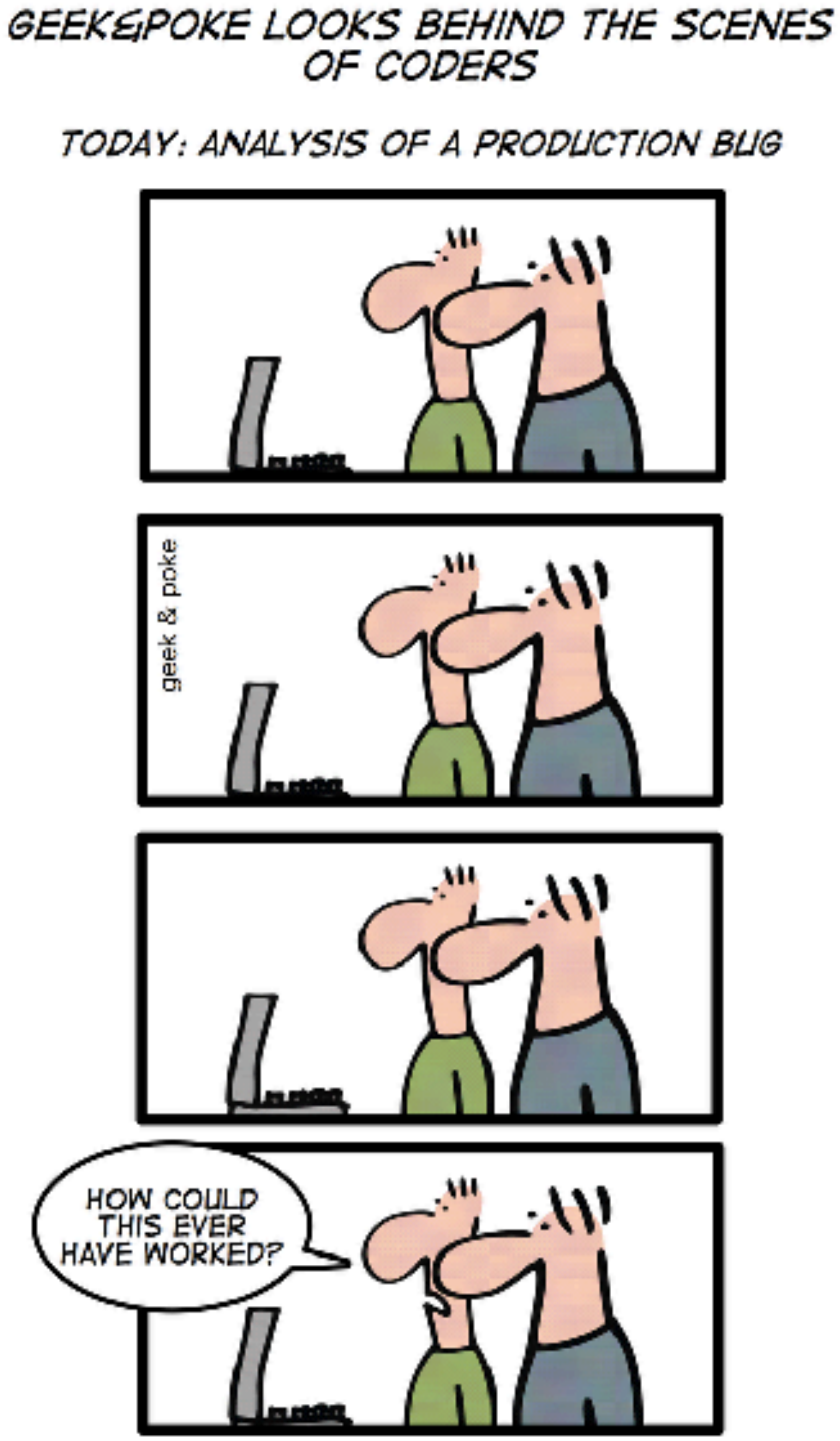
How and Why?

How:

- Continuously integrating the software done by the development team
- Run automated tests
- Push build to production-like environment
- Can release one version at the push of the button

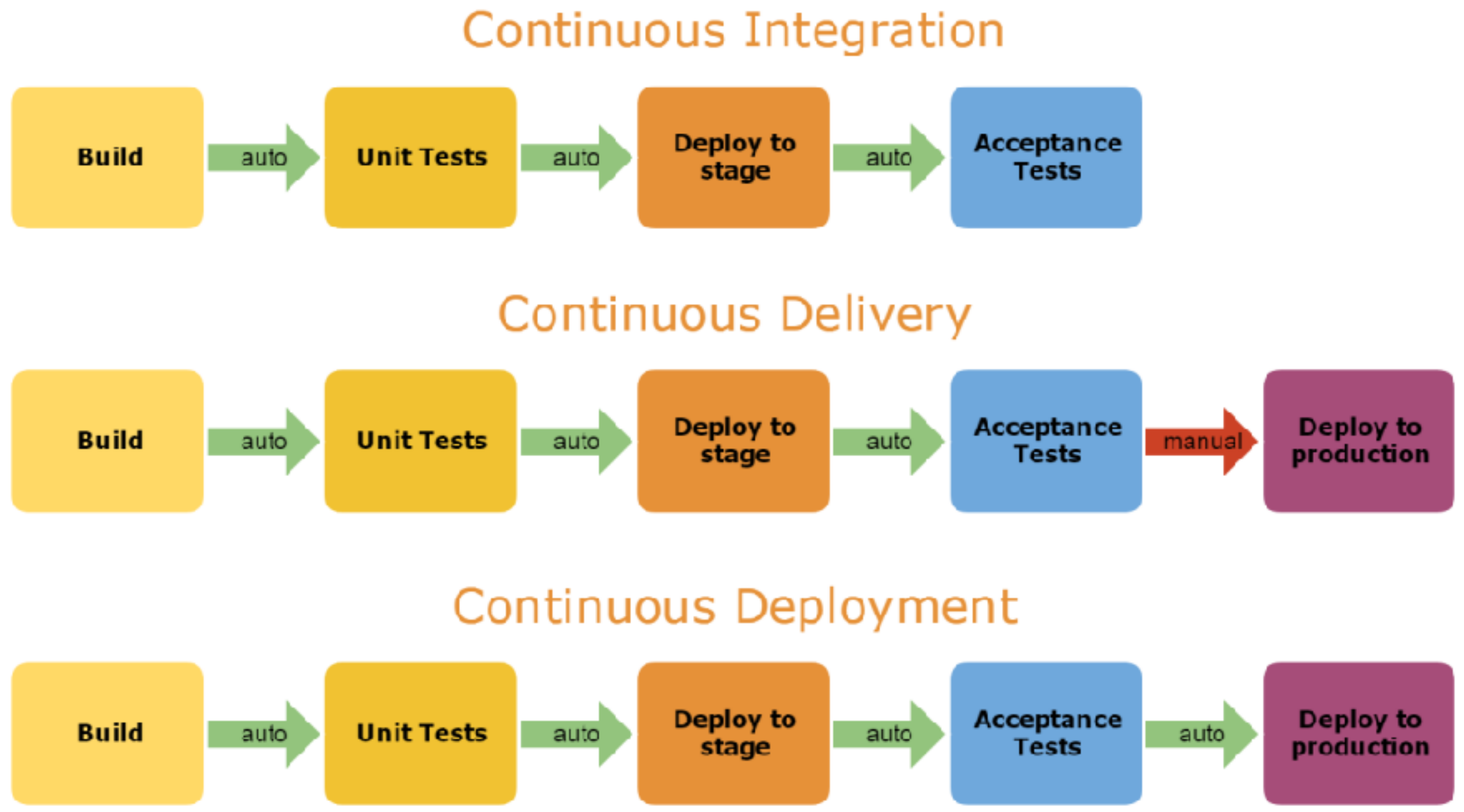
Why:

- Reduce deployment risks
- Change the version in production more rapidly
- Get the feedback earlier


GEEK&POKE LOOKS BEHIND THE SCENES OF CODERS
TODAY: ANALYSIS OF A PRODUCTION BUG
geek & poke
HOW COULD THIS EVER HAVE WORKED?

CI and CD

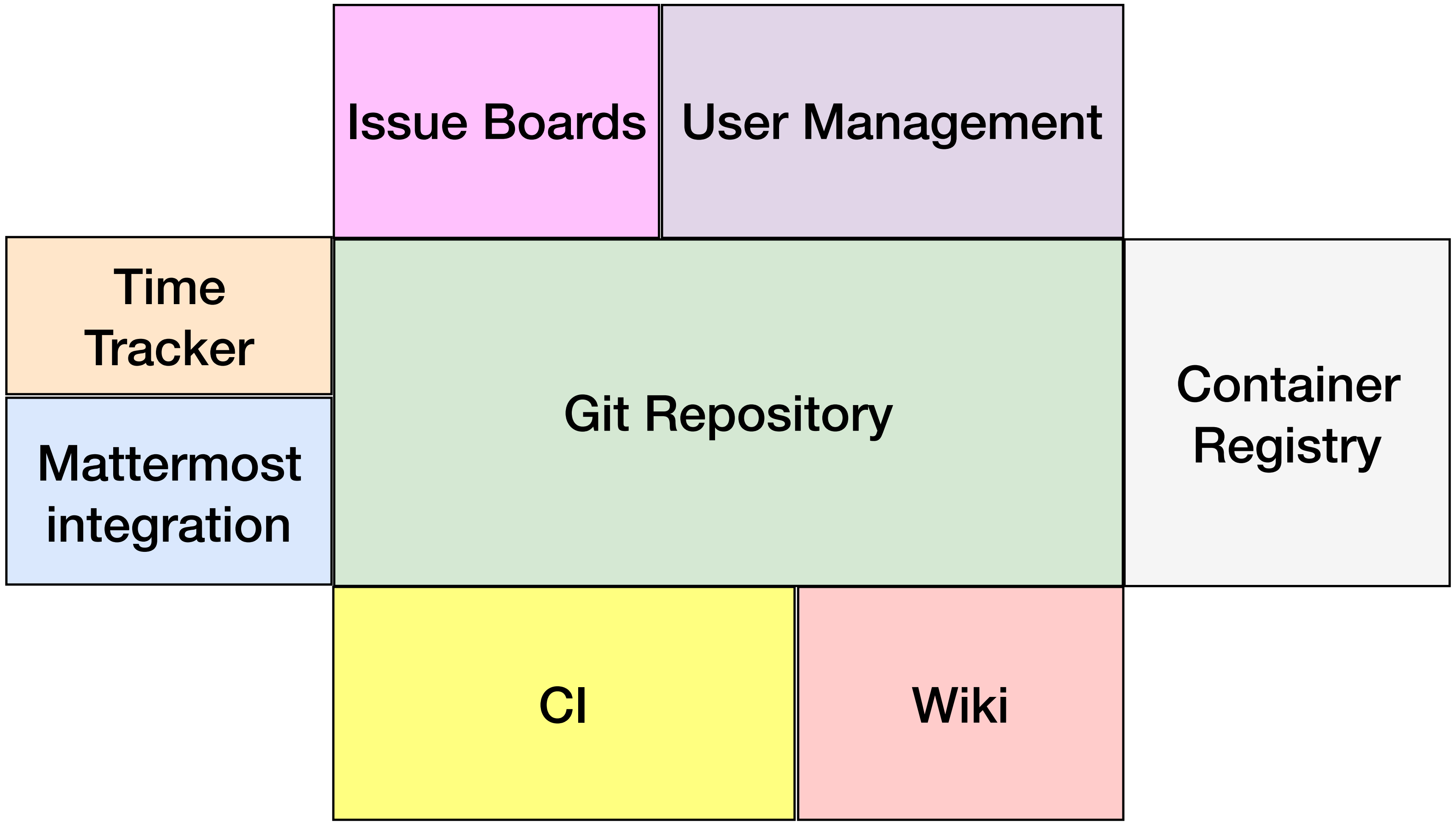
Summary



Gitlab

What is it?

- Git based hosting and collaboration platform
- Open source, freemium
- Hosted (free) or on premise
- Actively maintained



Gitlab CI

What and Why?

What:

- Fully integrated with Gitlab
- Integrated since v. 8.0
- Build scripts hosted in repo
- Git hooks
- Hosted (free) or on premise
- Actively maintained

GitLab CI

Open Source Continuous Integration made easy

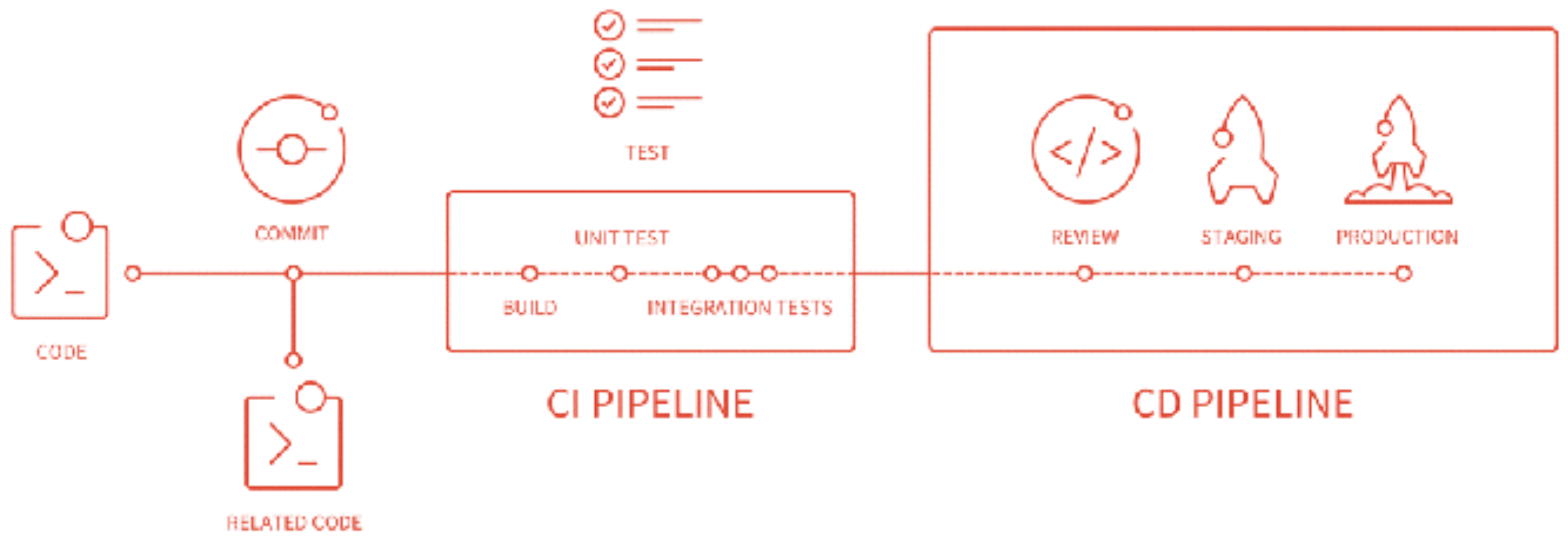
Why:

- Code and build scripts in the same repo
- Easy to start
- Scalable
- Isolated test environment

Gitlab CI

Pipelines and Stages

A pipeline is a group of jobs that get executed in stages(batches). All of the jobs in a stage are executed in parallel, and if they all succeed, the pipeline moves on to the next stage. If one of the jobs fails, the next stage is not executed.



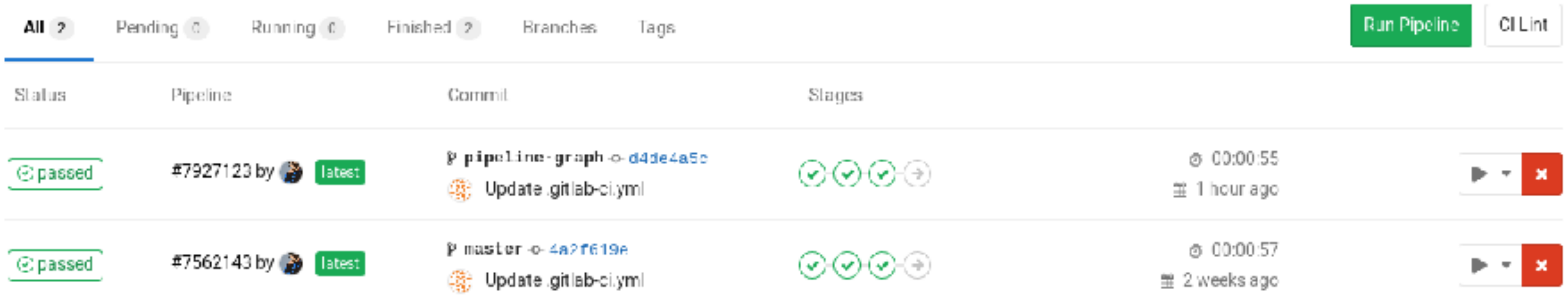
Pipelines are defined in .gitlab-ci.yml by specifying jobs in stages:

```
backend_test:
  stage: test
  script:
    - npm install
    - npm test
```

Gitlab CI

UI

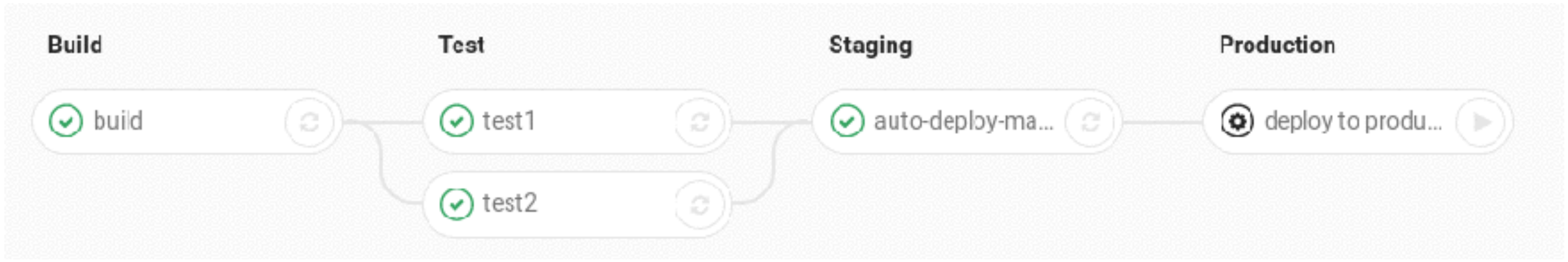
Pipeline status:



All 2 · Pending 0 · Running 0 · Finished 2 · Branches · Tags · Run Pipeline · CI Lint

Status	Pipeline	Commit	Stages		
passed	#7927123 by latest	pipeline-graph -o- d4de4a5c Update .gitlab-ci.yml	✓ ✓ ✓	00:00:55 1 hour ago	
passed	#7562143 by latest	master -o- 4a2f619e Update .gitlab-ci.yml	✓ ✓ ✓	00:00:57 2 weeks ago	

Job status:

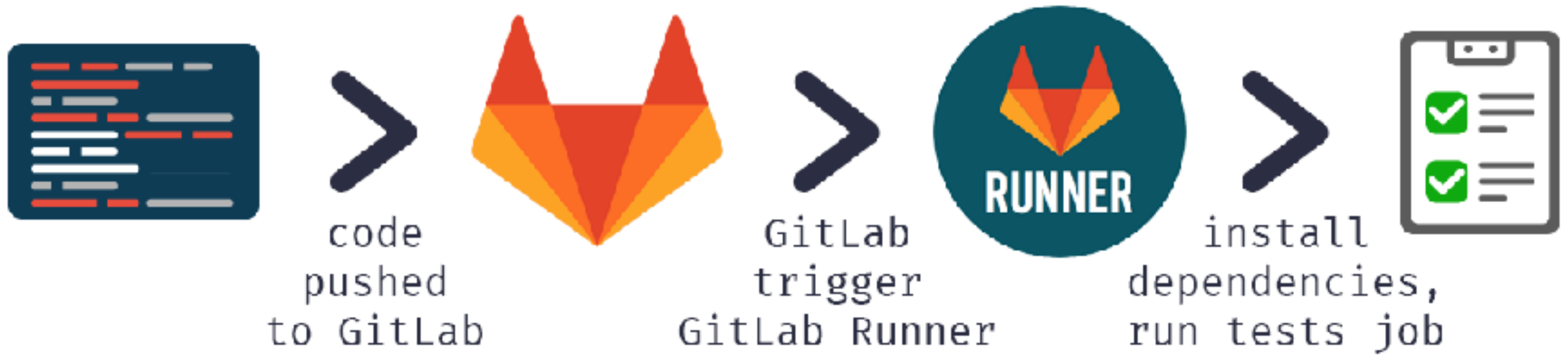


Gitlab CI

Runners

Runner is an application, that processes builds. It receives commands from Gitlab CI.
It is possible to tag runners so jobs run on runners which can process them (e.g. different OS).

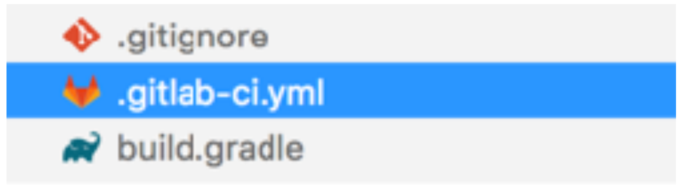
- Works with Linux, Windows and OSX
- Works as a Docker container
- Shared or specific
- Multiple Executors:
  - Docker -> In Docker container
  - Shell -> Locally
  - Docker SSH -> In Docker container over SSH
  - SSH -> Remote using SSH


code pushed to GitLab
GitLab trigger GitLab Runner
RUNNER
install dependencies, run tests job

Gitlab CI

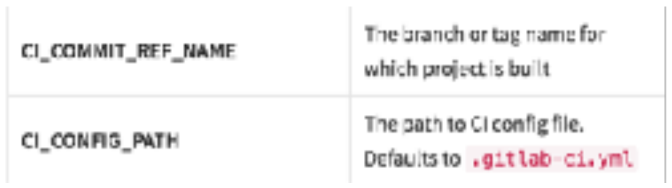
CI as Code

.gitlab-ci.yml in root



.gitignore
.gitlab-ci.yml
build.gradle

Gitlab CI Variables



CI_COMMIT_REF_NAME	The branch or tag name for which project is built
CI_CONFIG_PATH	The path to CI config file. Defaults to `.gitlab-ci.yml`

Define Stages

```
stages:
    - test
    - build
    - deploy
```

Define environment

```
image: node:8.3
```

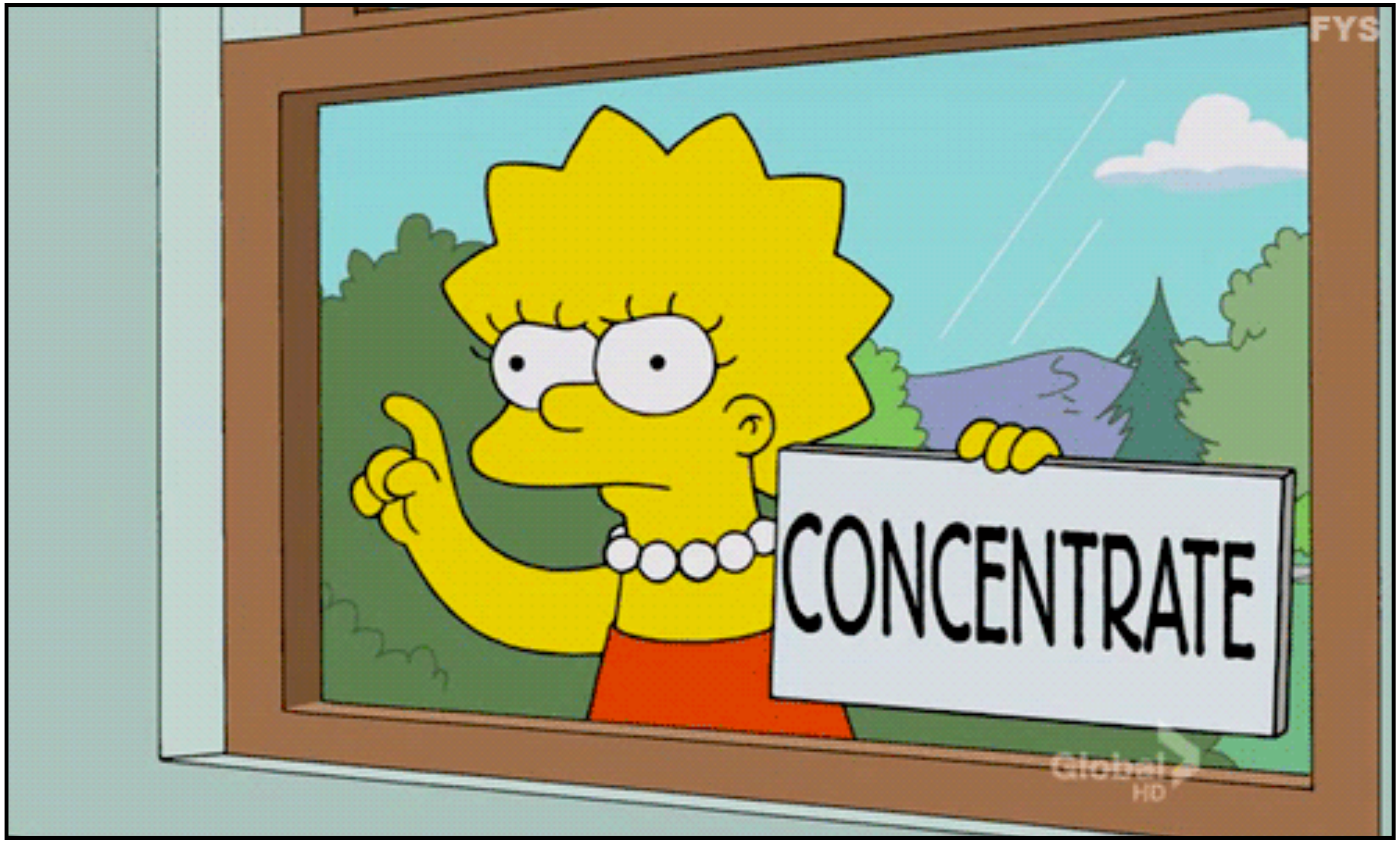
Example

```
image: ruby:2.1

before_script:
  - bundle install

stages:
  - build
  - test
  - deploy

job1:
  stage: build
  script:
    - execute-script-for-job1
  only:
    - master
  tags:
    - docker
```

Gitlab CI

CI as Code: Stage

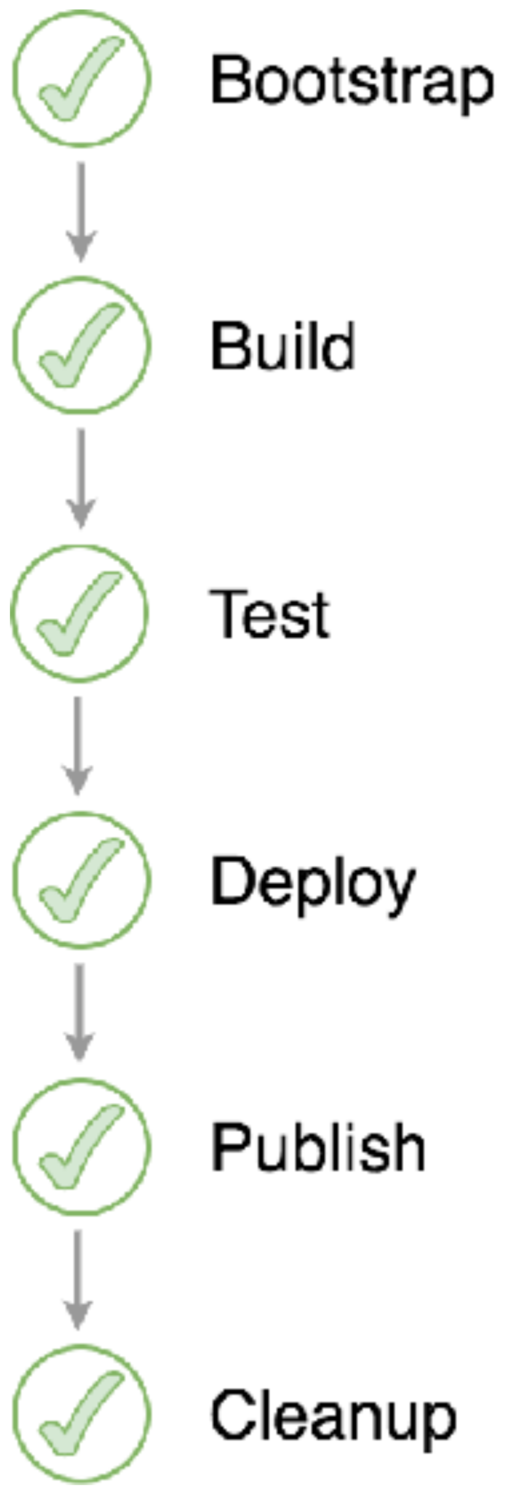
	`job1:` `stage: build`
Variables	`variables:` `DATABASE_URL: "test"`
Before script	`before_script:` `- execute-before-script-for-job1`
Script	`script:` `- execute-script-for-job1` `- something else`
After Script	`after_script:` `- execute-after-script-for-job1`
Artifacts	`artifacts:` `paths:` `- .variables` `expire_in: 1 week`
Only/ Except	`only:` `- master` `except:` `- develop`
Tags	`tags:` `- ruby` `- postgres`
When	`when: manual`


CONCENTRATE

Show Case

Node.js + React

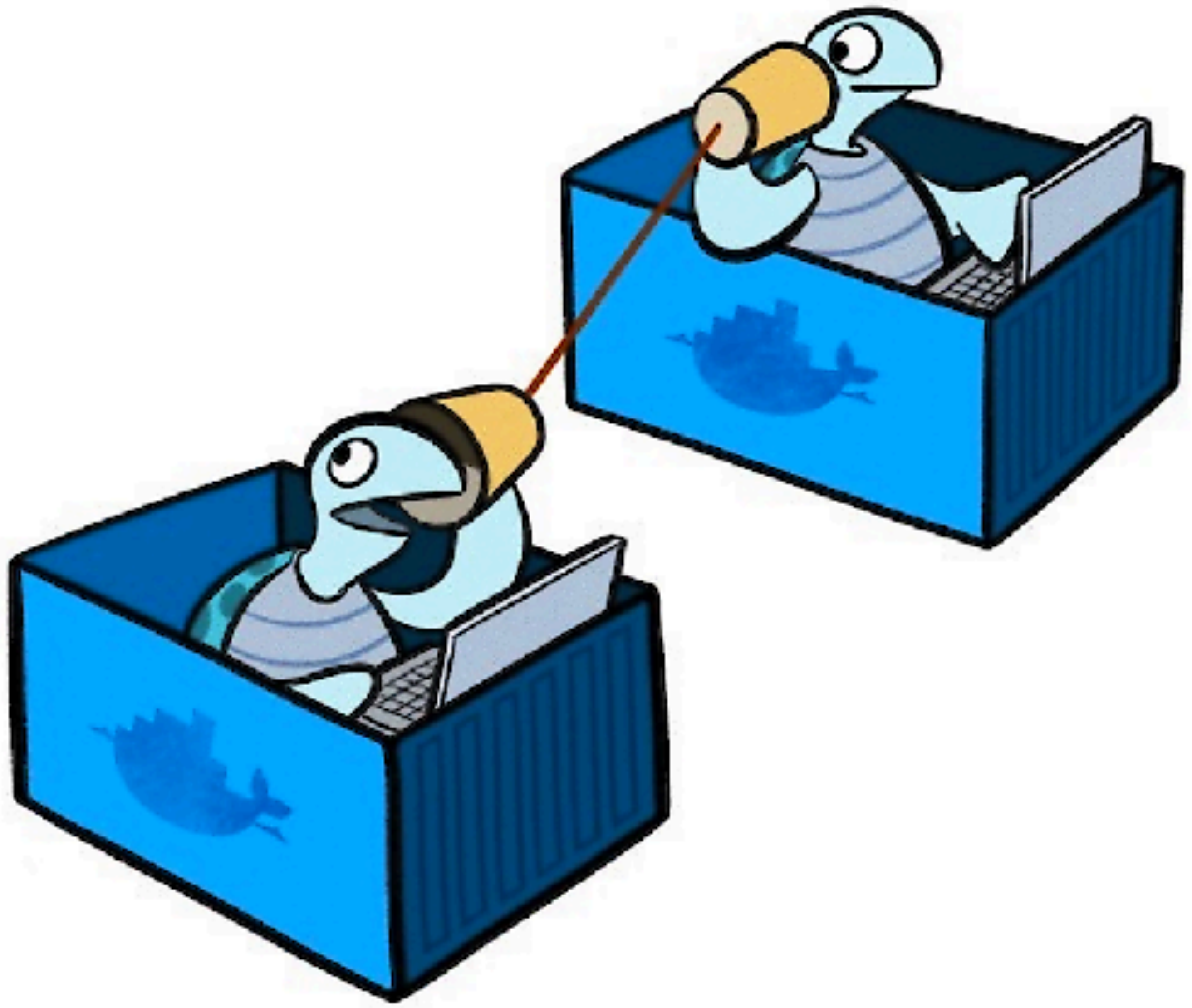
- Using of docker runner
- Using of Gitlab Pages
- Using of Node.js for Tests and Build
- Deploying Single Page Application without Backend
- Using of Gitlab CI Cache



Show Case

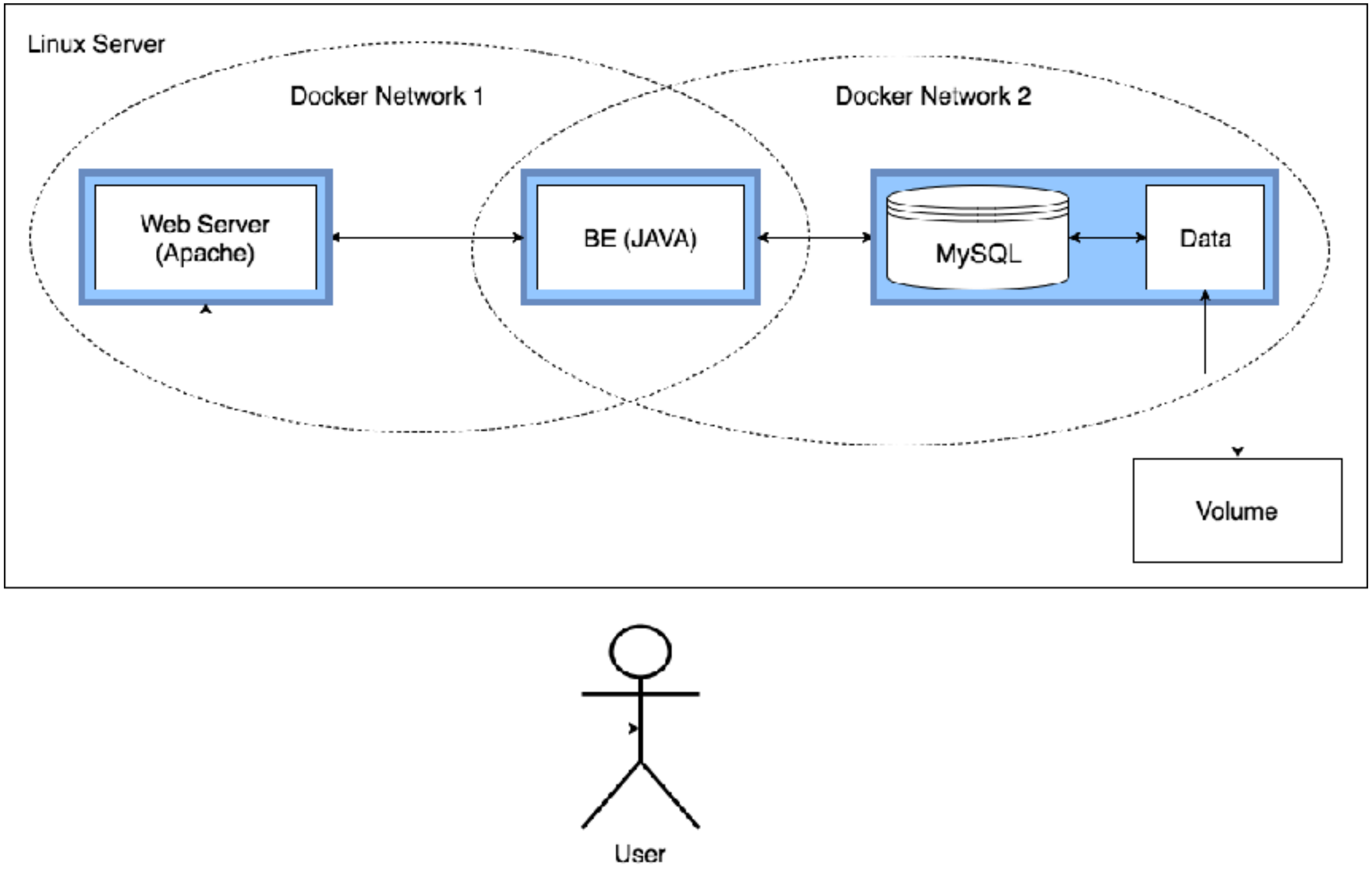
Java + Angular

- Using of shell runner
- Connecting the containers in docker network
- Using of docker-compose for simple management of containers



Show Case

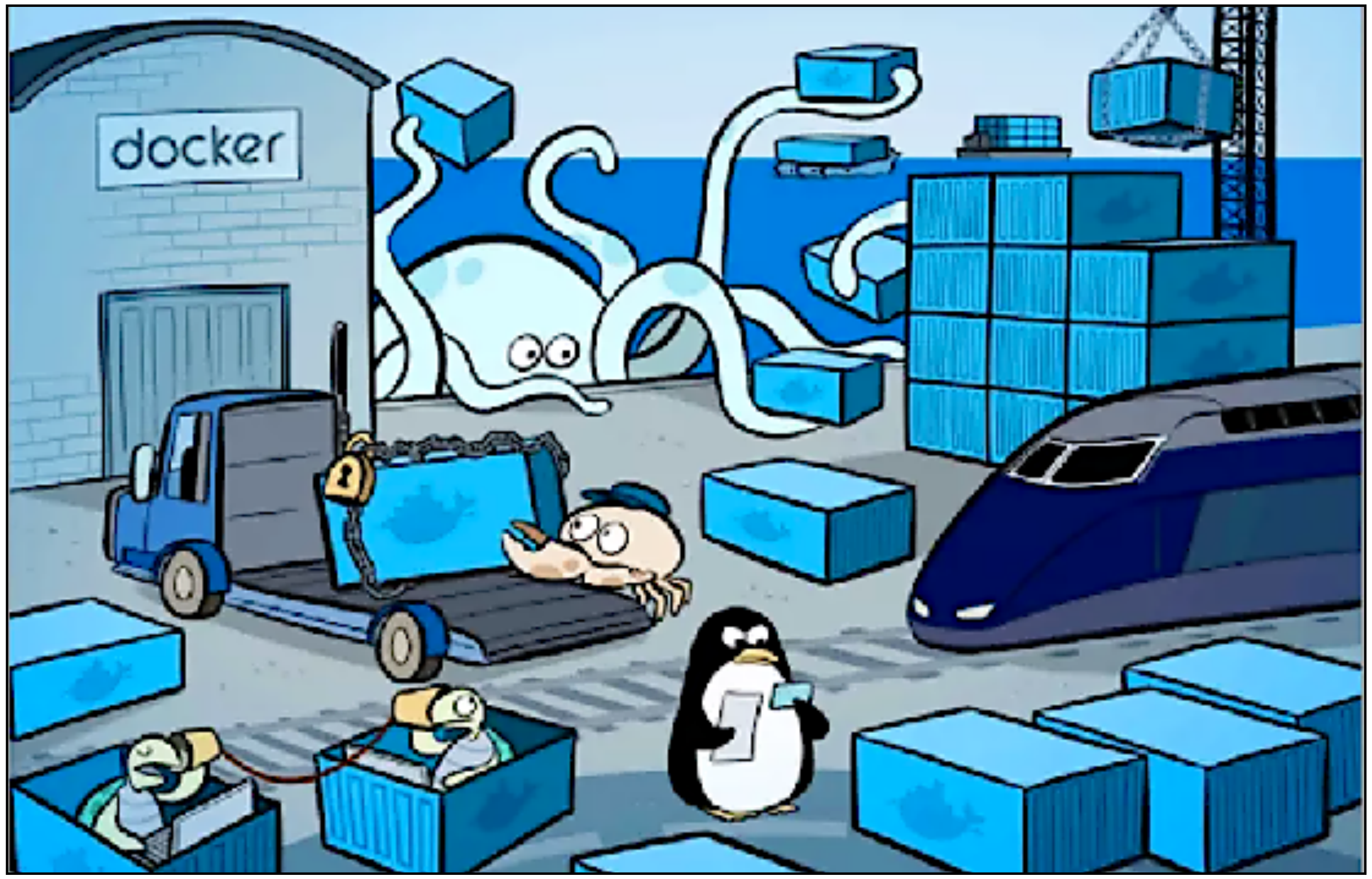
Java + Angular: Architecture



Show Case

Java + Angular: Docker-Compose

Compose is a tool for defining and running multi-container Docker applications. With Compose, you use a YAML file to configure your application's services. Then, with a single command, you create and start all the services from your configuration.



```
version: '2'

services:
  frontend:
    build:
      context: .
      dockerfile: Dockerfile
      args:
        - HTTP_PORT:${HTTP_PORT}
        - HTTPS_PORT:${HTTPS_PORT}
    image: ${PROJECT_NAME}
    volumes:
      - "./dist:/dist"
    ports:
      - "${HTTPS_PORT}:443"
      - "${HTTP_PORT}:80"
    restart: unless-stopped
    networks:
      - default
      - backend
    external_links:
      - ${BACKEND_NAME}:backend

networks:
  backend:
    external:
      name: ${BACKEND_NETWORK_NAME}
```

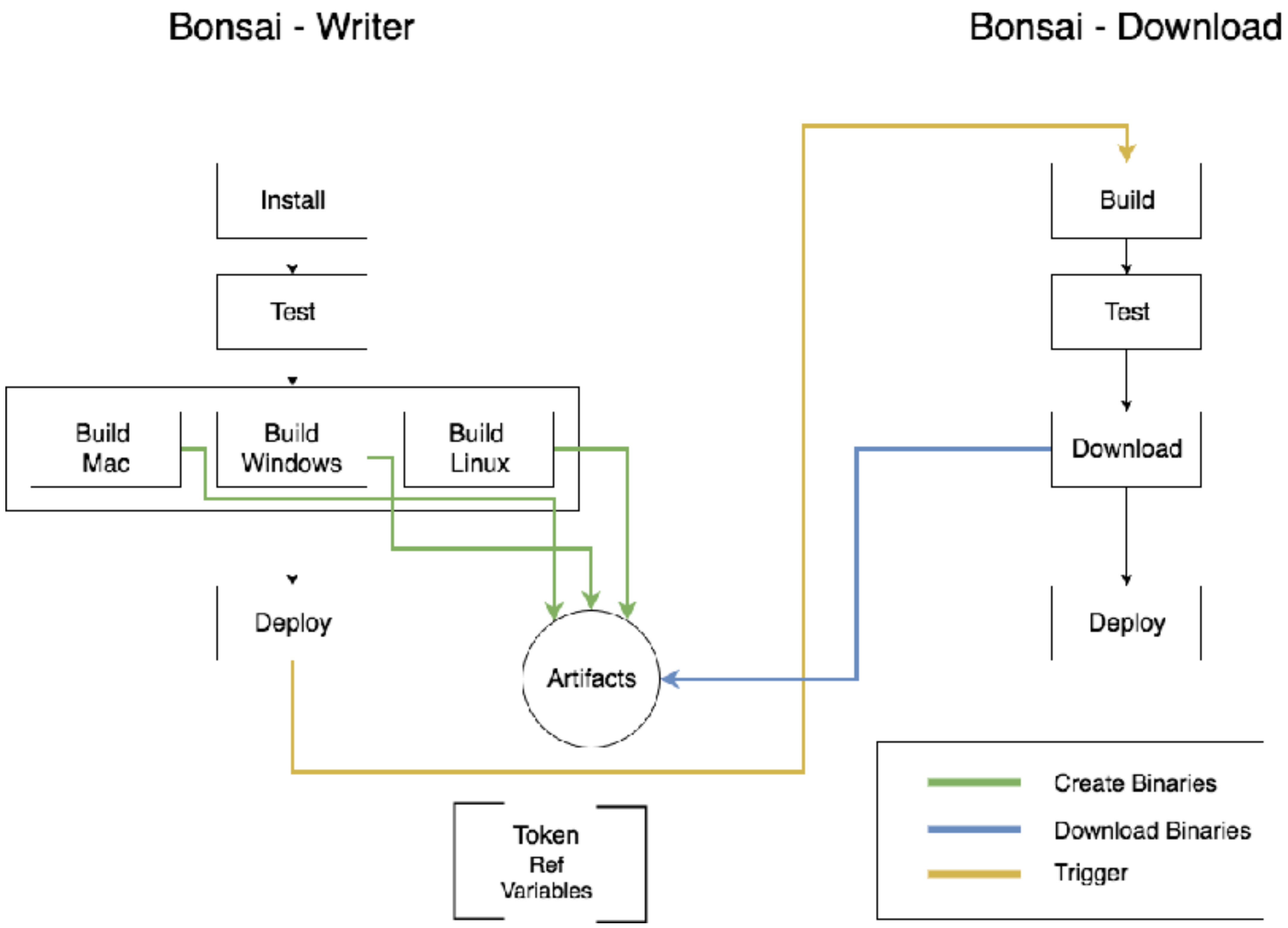
Show Case

Electron

- Build on 3 platforms with different runners
- Artifacts Api
- Build Trigger Api

Show Case

Electron: Pipelines



Gitlab CI vs Jenkins

Pros:

- Parallel builds
- Docker integration
- Fully Integrated in Gitlab
- Configurability per branch already on jobs-level
- Permission inheritance (Repository Manager)

- Highly customisable
- Community. A lot of resources and tutorials
- Supports multiple version control systems
- Easy to get up and running
- Cross-platform
- Build-in time based execution
- UI and Dashboard
- Report integration

Contras:

- Integrated only in Gitlab

- Poor quality of some plug-ins
- High overhead (setup and host).

Gitlab CI

Another features

Environments

Auto DevOps

Review Apps

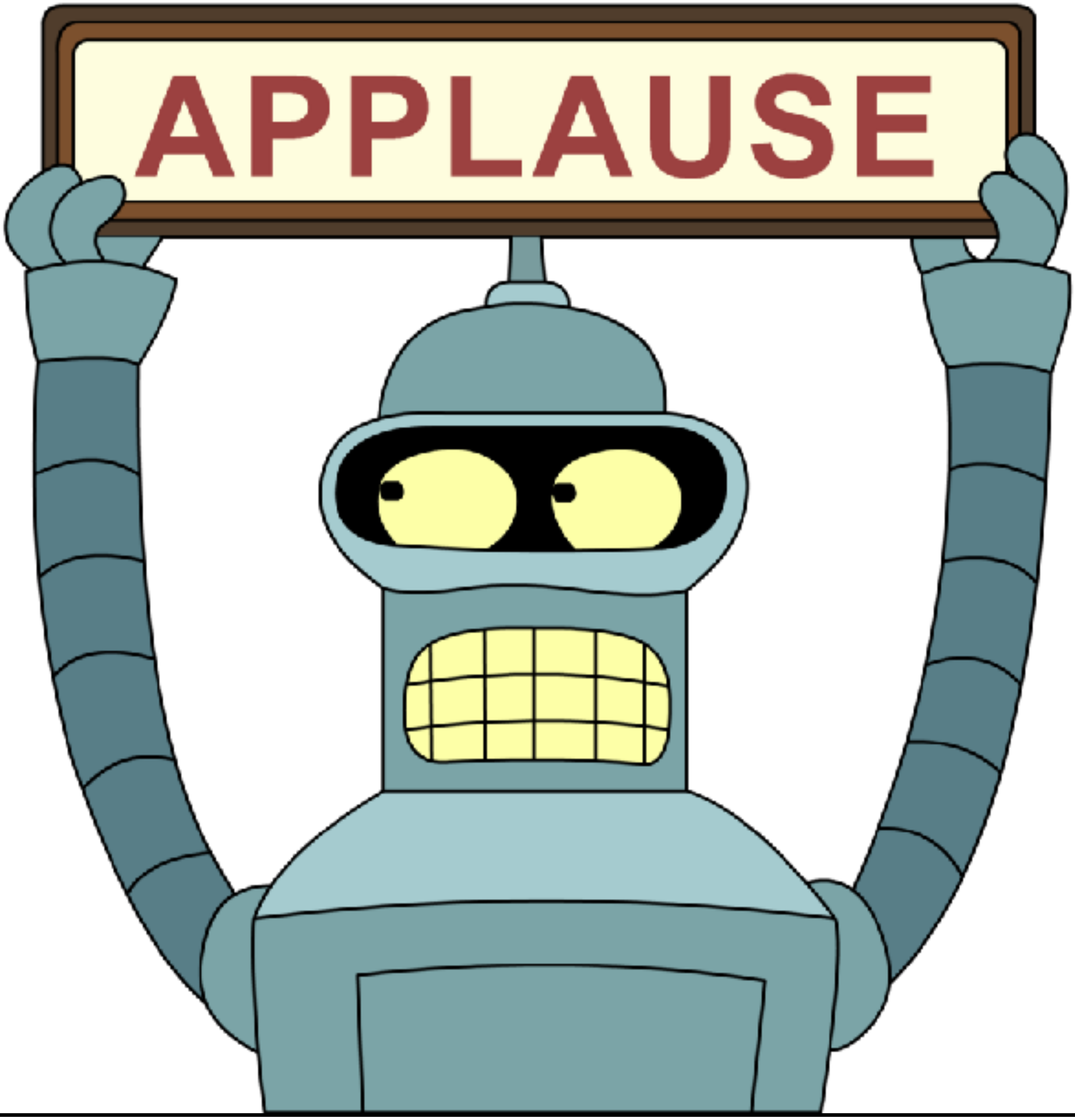
Trigger pipelines through the GitLab API

Trigger pipelines on a schedule

Deploy Boards - Check the current health

…

But not today!


APPLAUSE

ANY QUESTIONS
DO YOU HAVE?

Sources

Links:

- https://about.gitlab.com/features/gitlab-ci-cd/
- https://www.inovex.de/blog/modern-cicd-with-jenkins-2-and-gitlab-ci-comparison/
- https://martinfowler.com/
- https://docs.docker.com/compose/
- https://www.docker.com

Pictures:

- http://sqlity.net/wp-content/uploads/2015/01/Benefits_of_Continuous_Integration.png
- https://www.zuehlke.com/blog/app/uploads/2015/11/geek-and-poke.png
- http://electric-cloud.com/wp-content/uploads/use-case-graphic_continuous-delivery.png
- https://docs.gitlab.com/ee/ci/img/cicd_pipeline_infograph.png
- https://image.slidesharecdn.com/ranchergitlab320171006-171007014521/95/rancher-gitlab-3-21-638.jpg?cb=1507341238
- https://i.pinimg.com/originals/2b/be/4b/2bbe4b9818440b610eadd30195fff3fa.gif
- https://denibertovic.com/talks/supercharge-development-env-using-docker/img/what_is_docker.png
- http://blog.arungupta.me/wp-content/uploads/2015/12/docker-networking.png
- https://media.serious.io/98dadb4361c5f588/static.gif
- http://geekandpoke.typepad.com/.a/6a00d8341d3df553ef0134887d41fd970c-pi