Introduction

In this project we are going to see how to successfully build an pipeline to host and scale an app using various CI/CD tools

We will leverage tools such as Git, Docker, Jenkins, Ansible, Kubernetes and spin an virtual machine to build our pipeline

We will go through three major stages to publish our app

1.Code

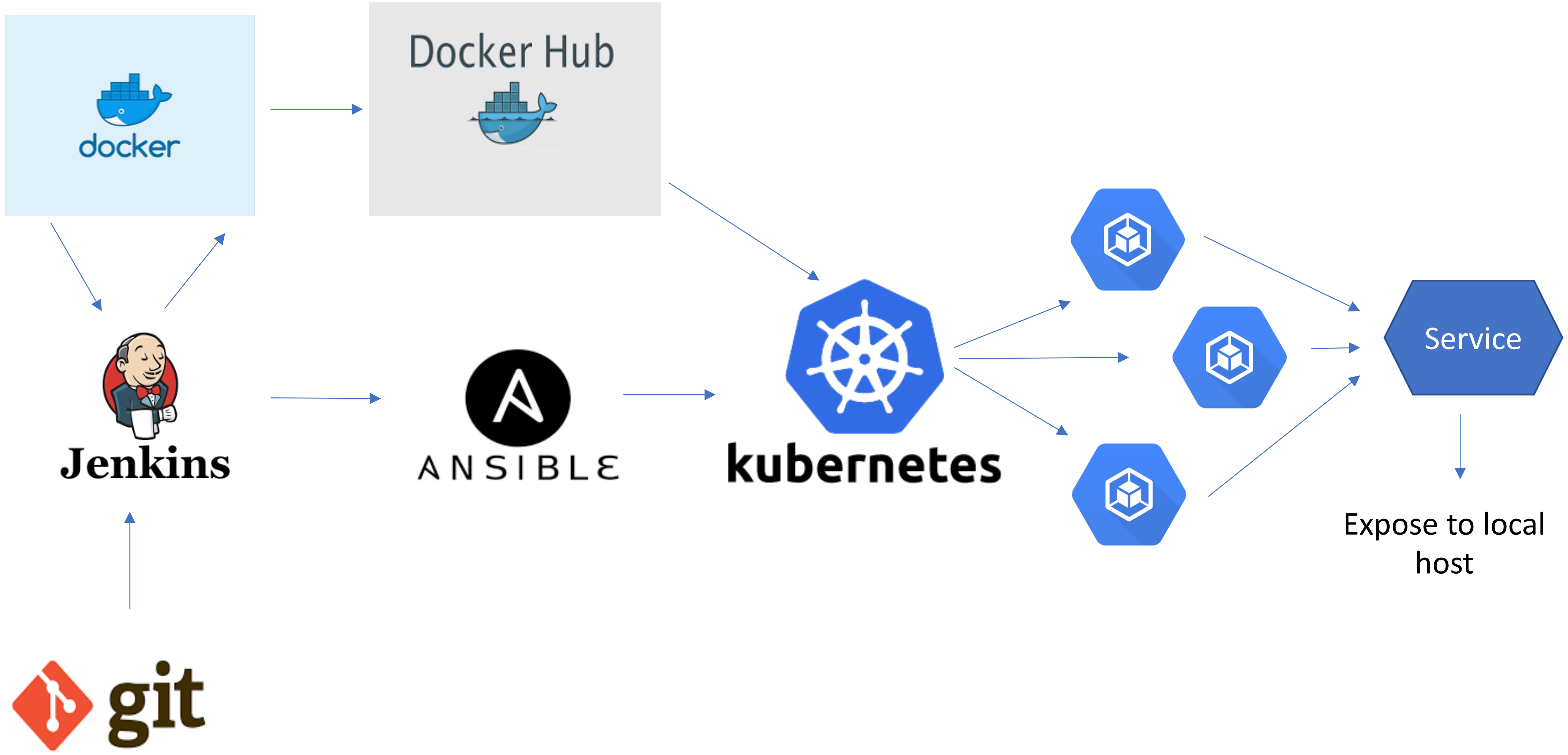
2.Build

3.Deploy

Tools Used

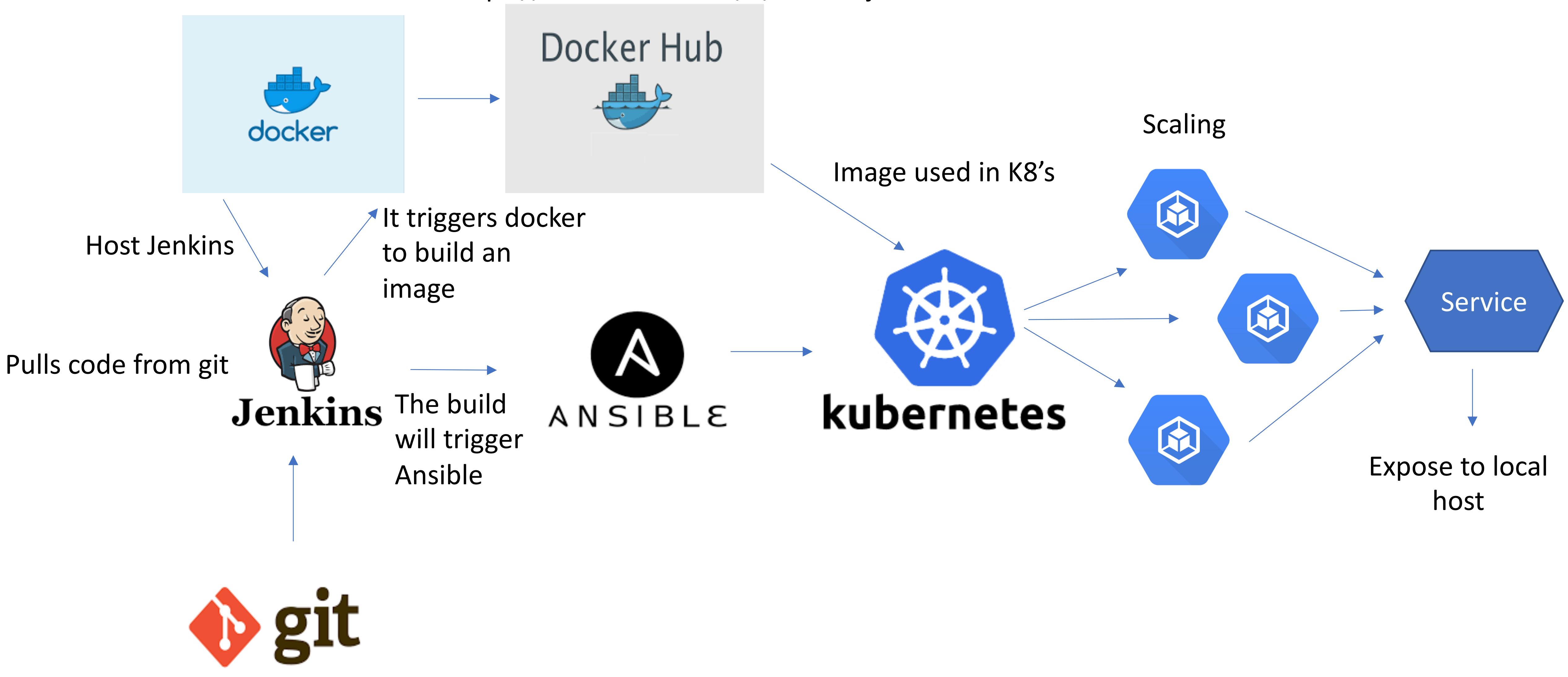
- Git
- Docker
- Jenkins
- Ansible
- Kubernetes
- Virtual machine

Overview of the pipeline



Explanation of the pipeline

https://hub.docker.com/u/sarankaja



https://github.com/kajasaran/2020_03_DO_Boston_casestudy_part_1

Explanation

- The changes we make to the code are pushed to Git.
- We then open docker and run Jenkins on one docker container.
- We configure a Jenkins pipeline using our git repo.
- The changes made in git trigger a build in Jenkins and it will start building the code that has been pulled from git and runs test cases.
- Once the testcases are successful it pushes the code to the docker and creates an image which is pushed to our docker hub. It also triggers Ansible to build the project.
- Ansible is configured to run Kubernetes with three pods once this build is triggered, pods expose an port to view the output in localhost.
- We Should successfully see the output once everything is configured properly

Requirements before running pipeline

We should have :

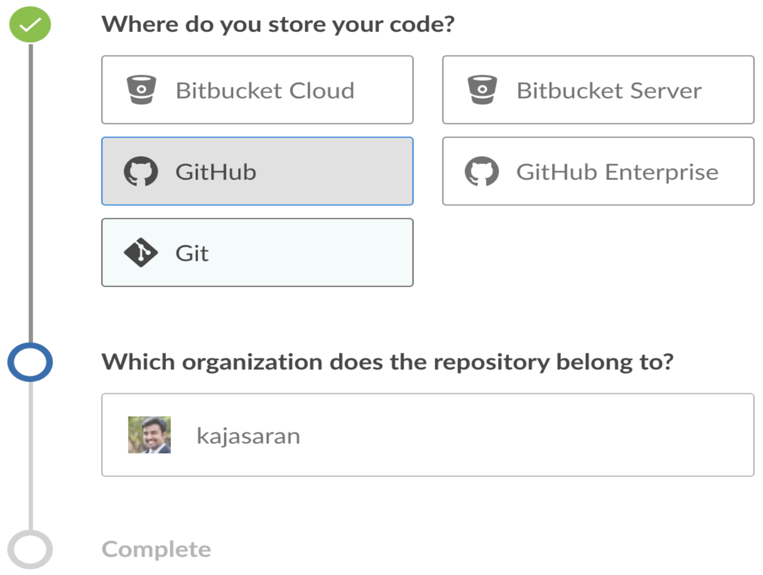
1. Requirements.txt
2. Dockerfile
3. Playbook.yaml

Once you have all the above files working

Create Jenkinsfile invoking all the services

steps to build a pipeline

1) Run docker
2) Run Jenkins container on docker with blue ocean image
3) Configure Jenkins
4) Open blue ocean-> New pipeline



Where do you store your code?

Bitbucket Cloud | Bitbucket Server | [✓ GitHub] | GitHub Enterprise | Git

Which organization does the repository belong to?

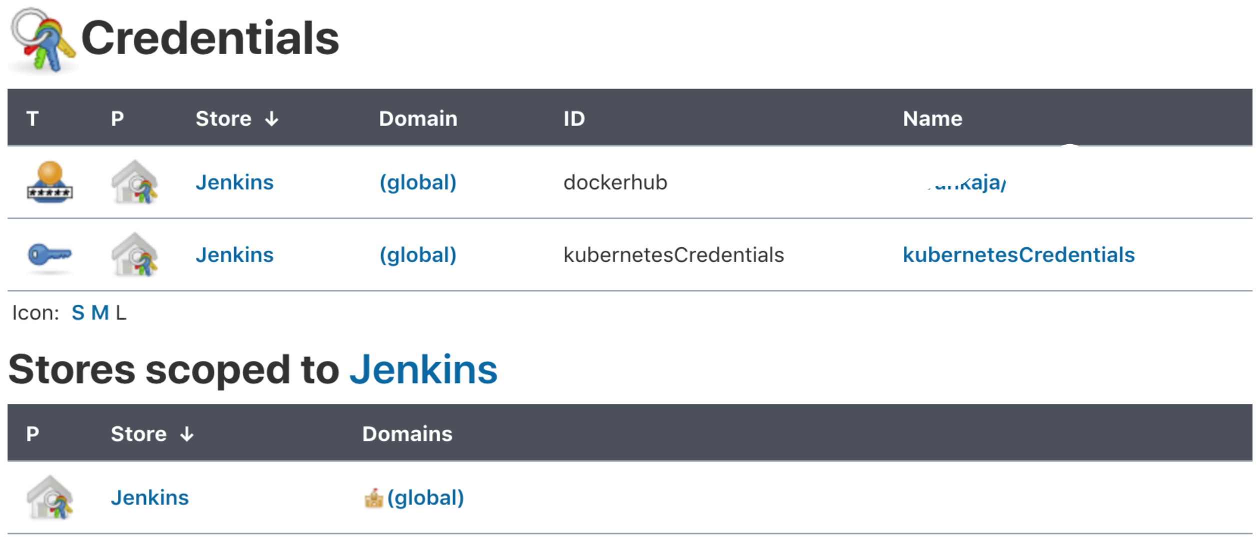
kajasaran

Complete

Contd.

4. The above step will create a webhook with your git using your Jenkins file

5. create security keys in jenkins



Credentials

T	P	Store ↓	Domain	ID	Name
		Jenkins	(global)	dockerhub	[illegible]kaja/
		Jenkins	(global)	kubernetesCredentials	kubernetesCredentials

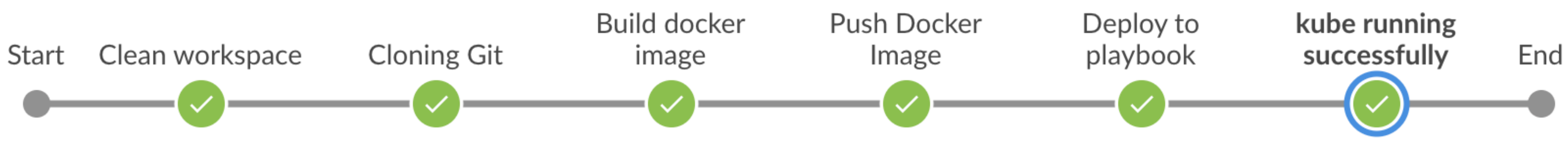
Icon: S M L

Stores scoped to Jenkins

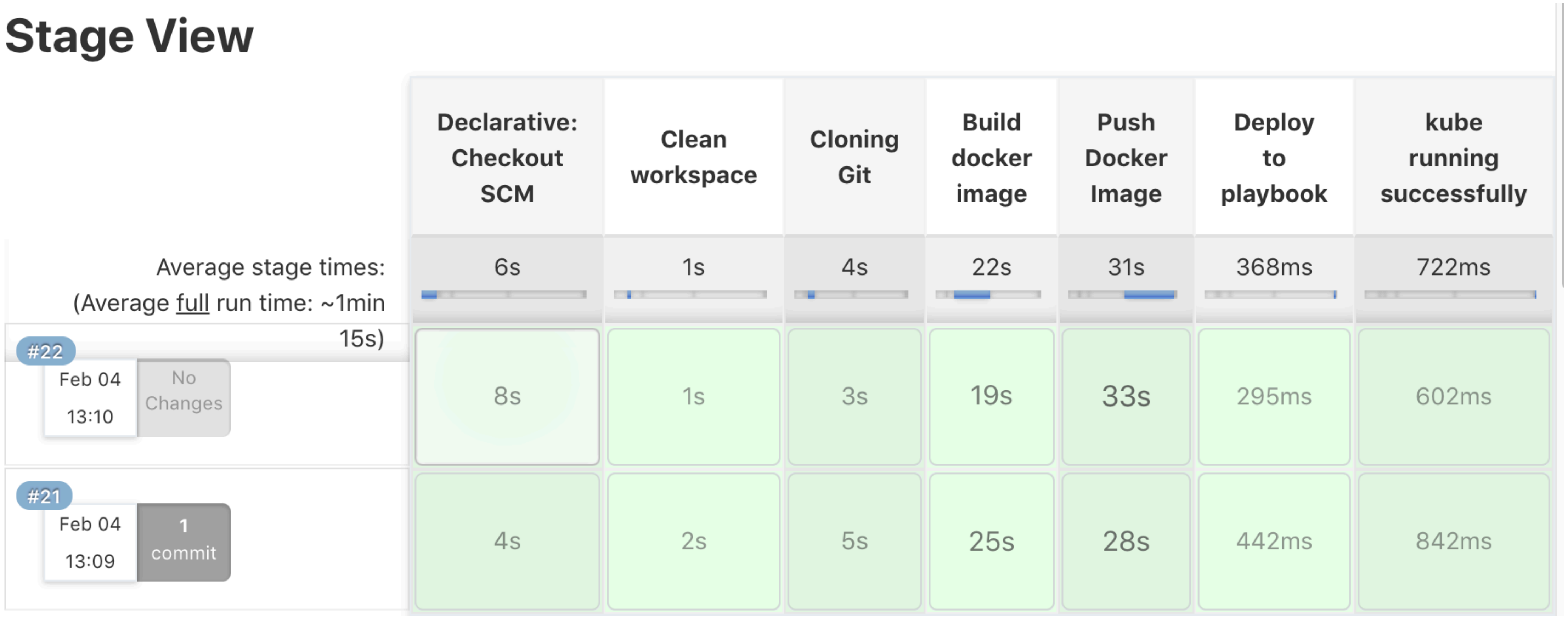
P	Store ↓	Domains
	Jenkins	(global)

Blue ocean pipeline output

Once all the files are configured properly the pipeline appears as follows



Jenkins pipeline output

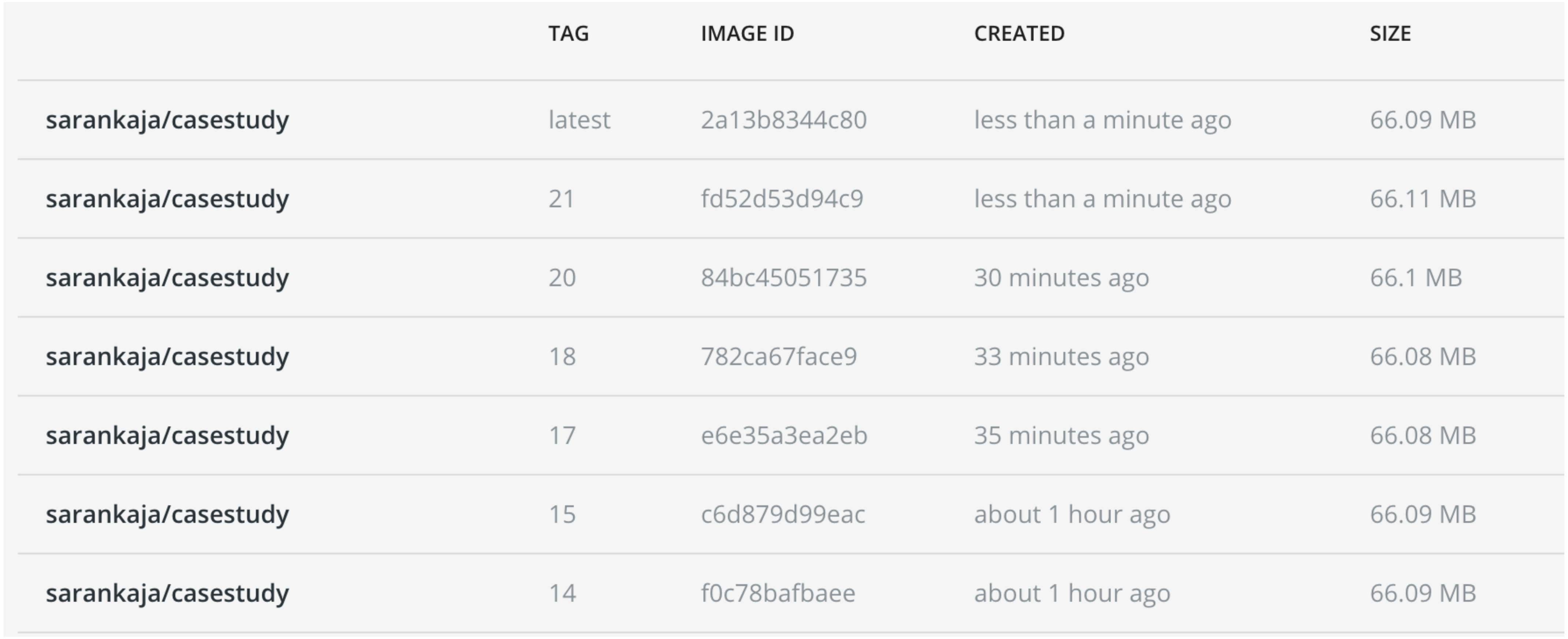


Stage View

	Declarative: Checkout SCM	Clean workspace	Cloning Git	Build docker image	Push Docker Image	Deploy to playbook	kube running successfully
Average stage times: (Average full run time: ~1min 15s)	6s	1s	4s	22s	31s	368ms	722ms
#22 Feb 04 13:10 No Changes	8s	1s	3s	19s	33s	295ms	602ms
#21 Feb 04 13:09 1 commit	4s	2s	5s	25s	28s	442ms	842ms

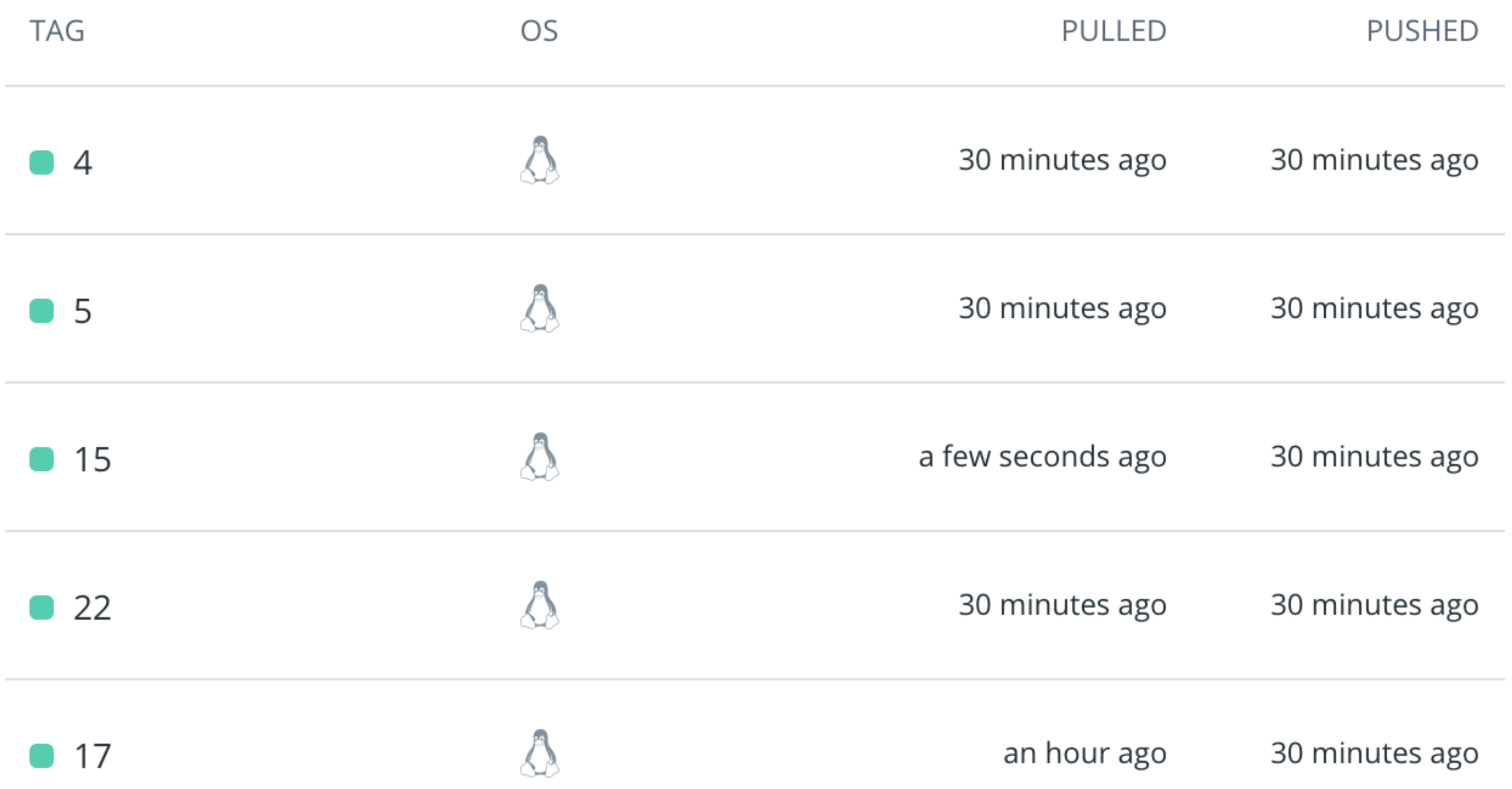
Docker Output

- You can see the image been built after each iteration



	TAG	IMAGE ID	CREATED	SIZE
sarankaja/casestudy	latest	2a13b8344c80	less than a minute ago	66.09 MB
sarankaja/casestudy	21	fd52d53d94c9	less than a minute ago	66.11 MB
sarankaja/casestudy	20	84bc45051735	30 minutes ago	66.1 MB
sarankaja/casestudy	18	782ca67face9	33 minutes ago	66.08 MB
sarankaja/casestudy	17	e6e35a3ea2eb	35 minutes ago	66.08 MB
sarankaja/casestudy	15	c6d879d99eac	about 1 hour ago	66.09 MB
sarankaja/casestudy	14	f0c78bafbaee	about 1 hour ago	66.09 MB

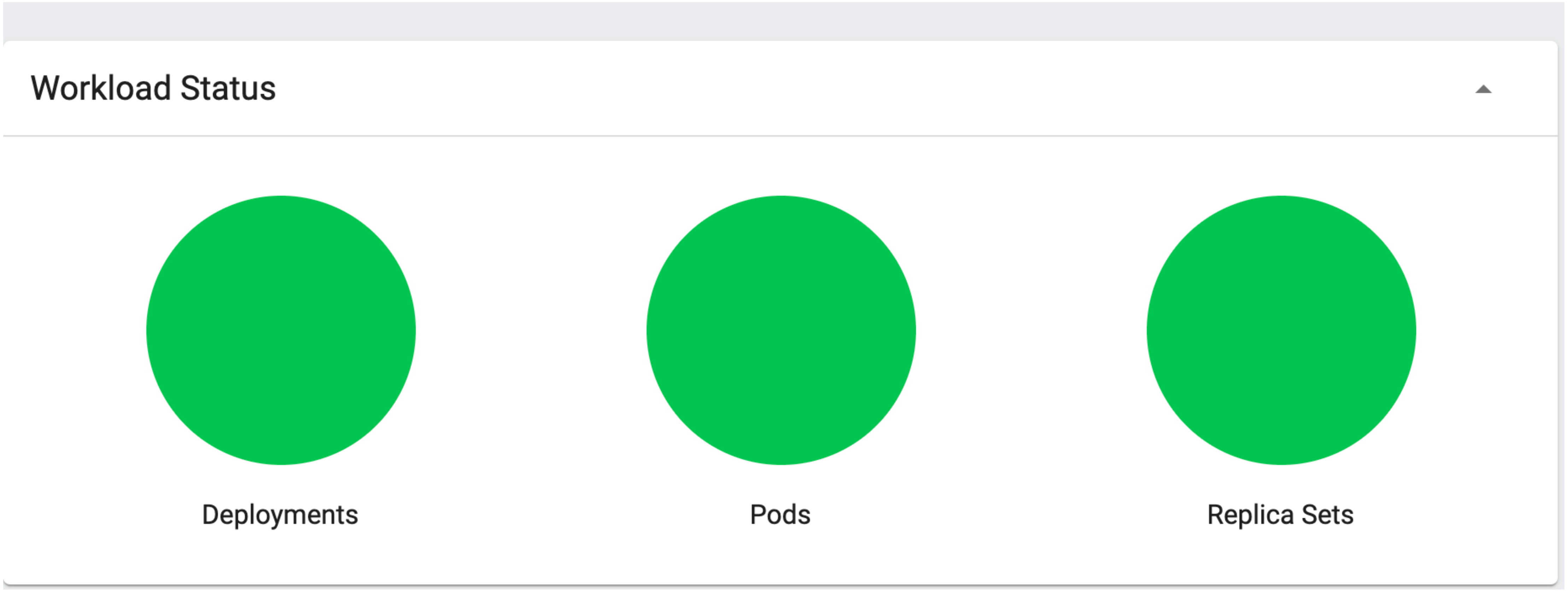
Docker Hub Output



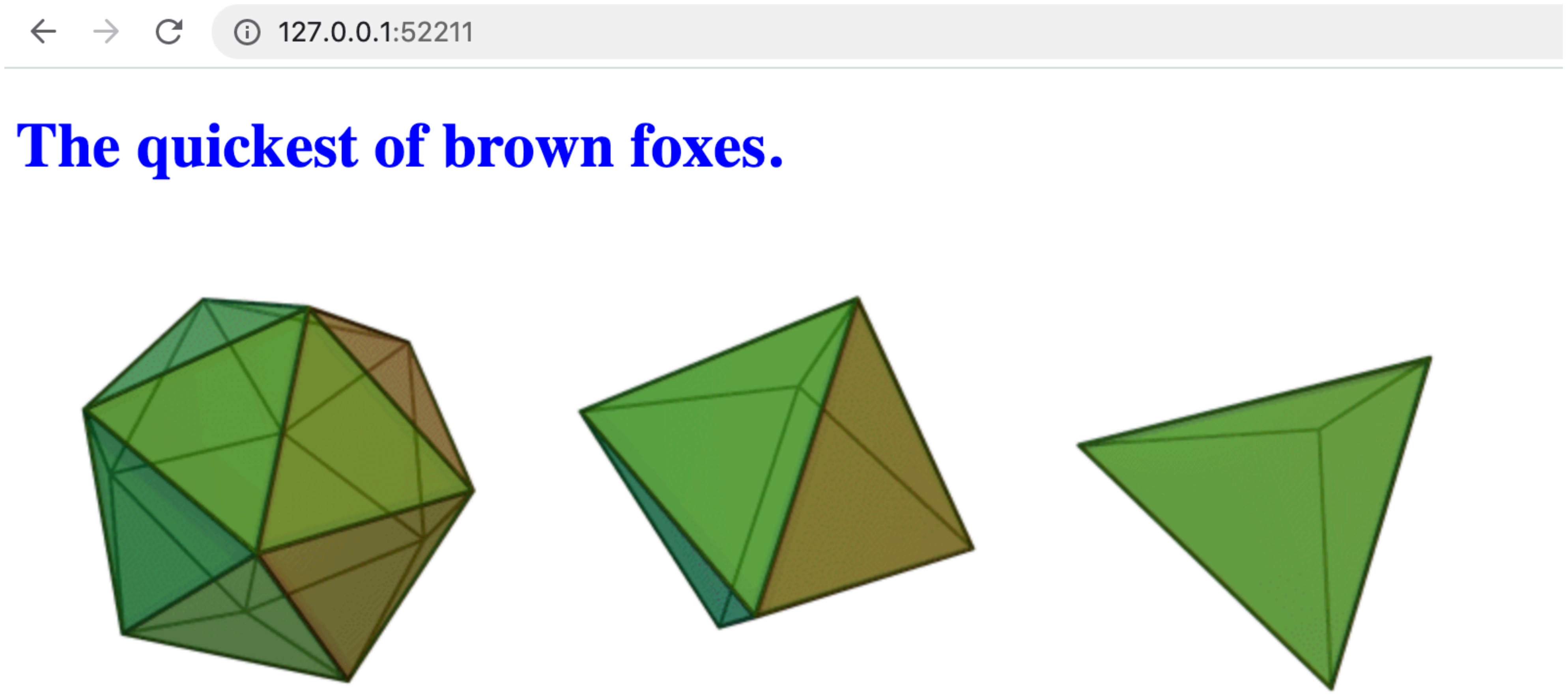
TAG	OS	PULLED	PUSHED
🟩 4	🐧	30 minutes ago	30 minutes ago
🟩 5	🐧	30 minutes ago	30 minutes ago
🟩 15	🐧	a few seconds ago	30 minutes ago
🟩 22	🐧	30 minutes ago	30 minutes ago
🟩 17	🐧	an hour ago	30 minutes ago

Kubernetes Output

- You can see the final output In Kubernetes where all the pods are green



Our desired output



The quickest of brown foxes.

Steps to test at every stage and build

1) Clone the repository
2) Test if the flask app is working in all the stages in local
3) Deploy the flask app in Jenkins
4) Create the **Jenkinsfile** to run all the services

Next Steps

- Migrating this to Cloud
- And adding testing and monitoring tools