Atlassian Private Cloud

celix Solutions GmbH

Thomas Rieder

Head of Engineering

celix

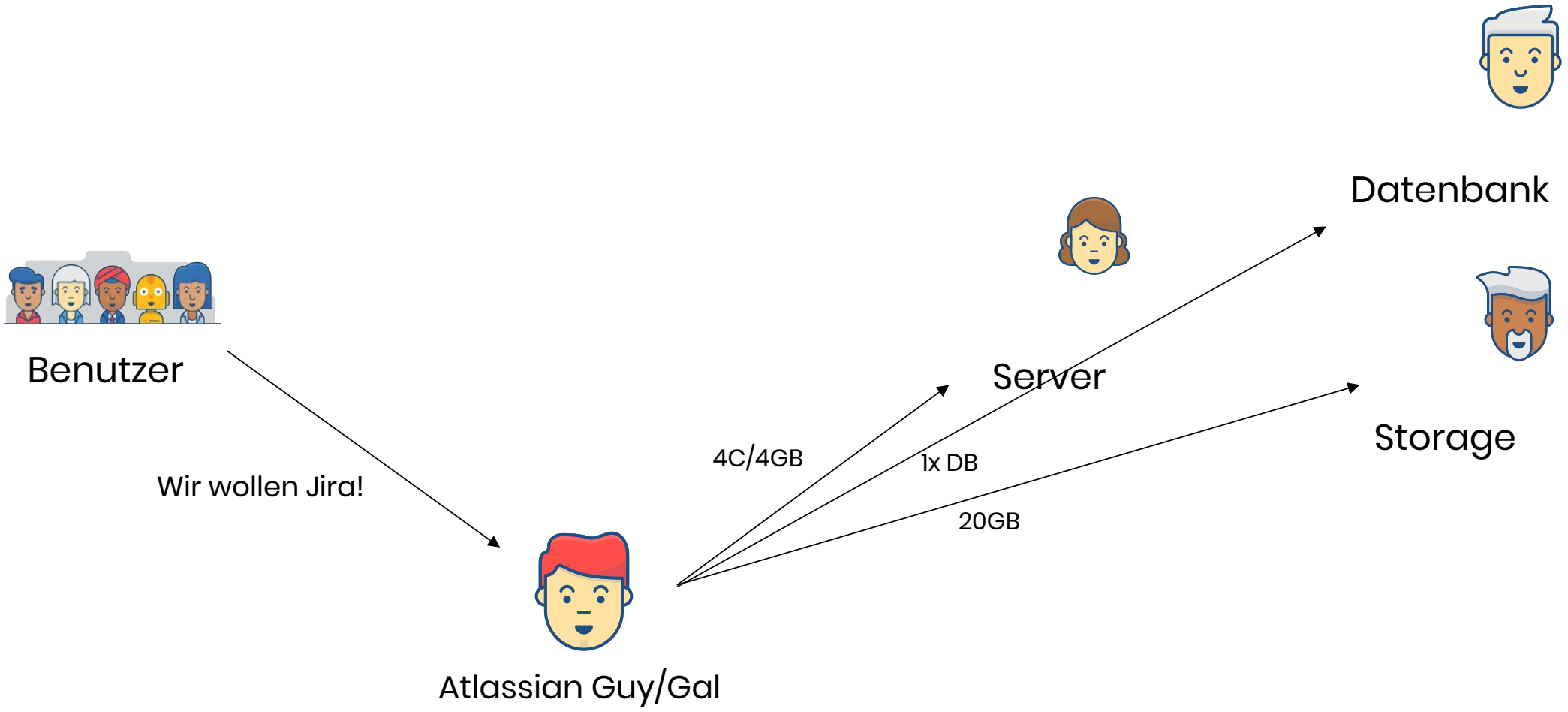
Jira Deployment in 2018

Benutzer

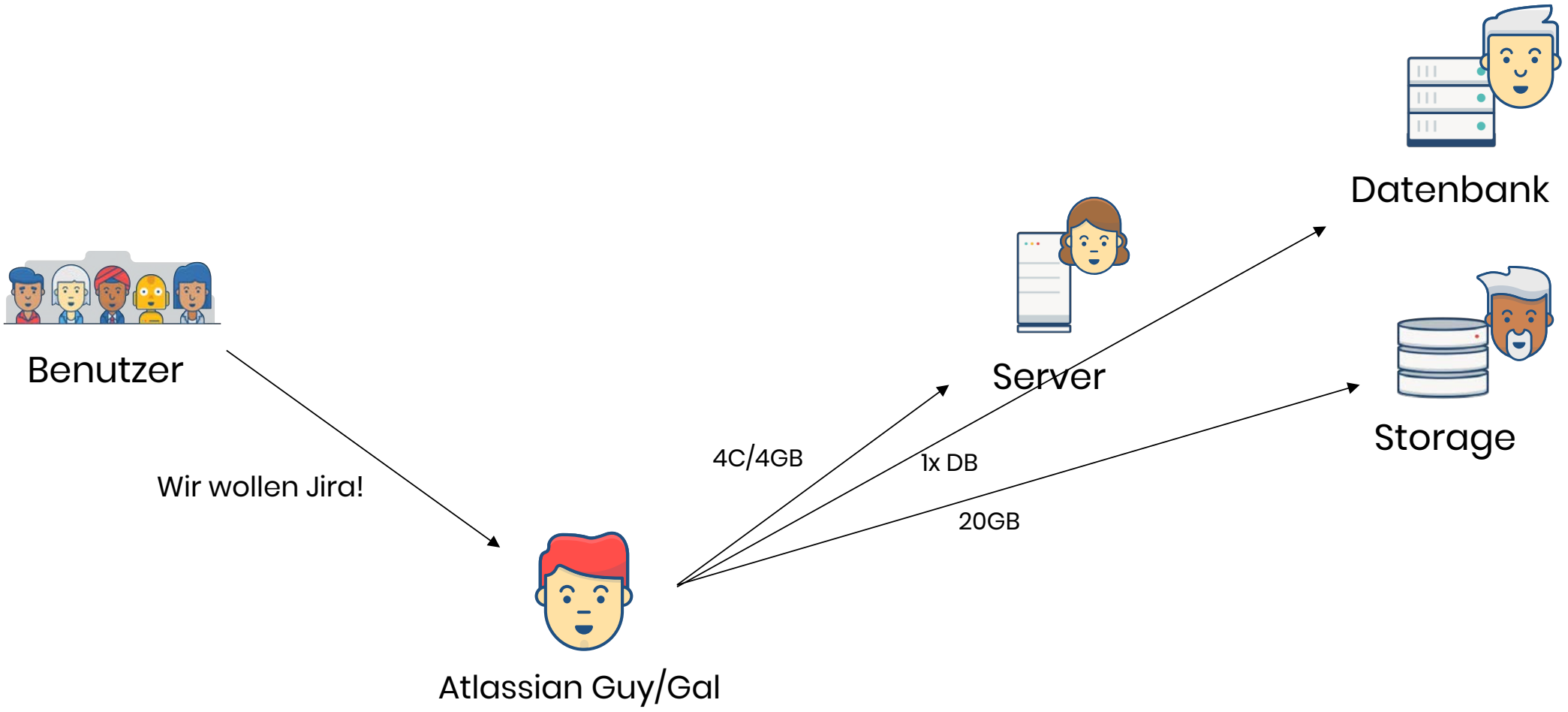
Wir wollen Jira!

Atlassian Guy/Gal

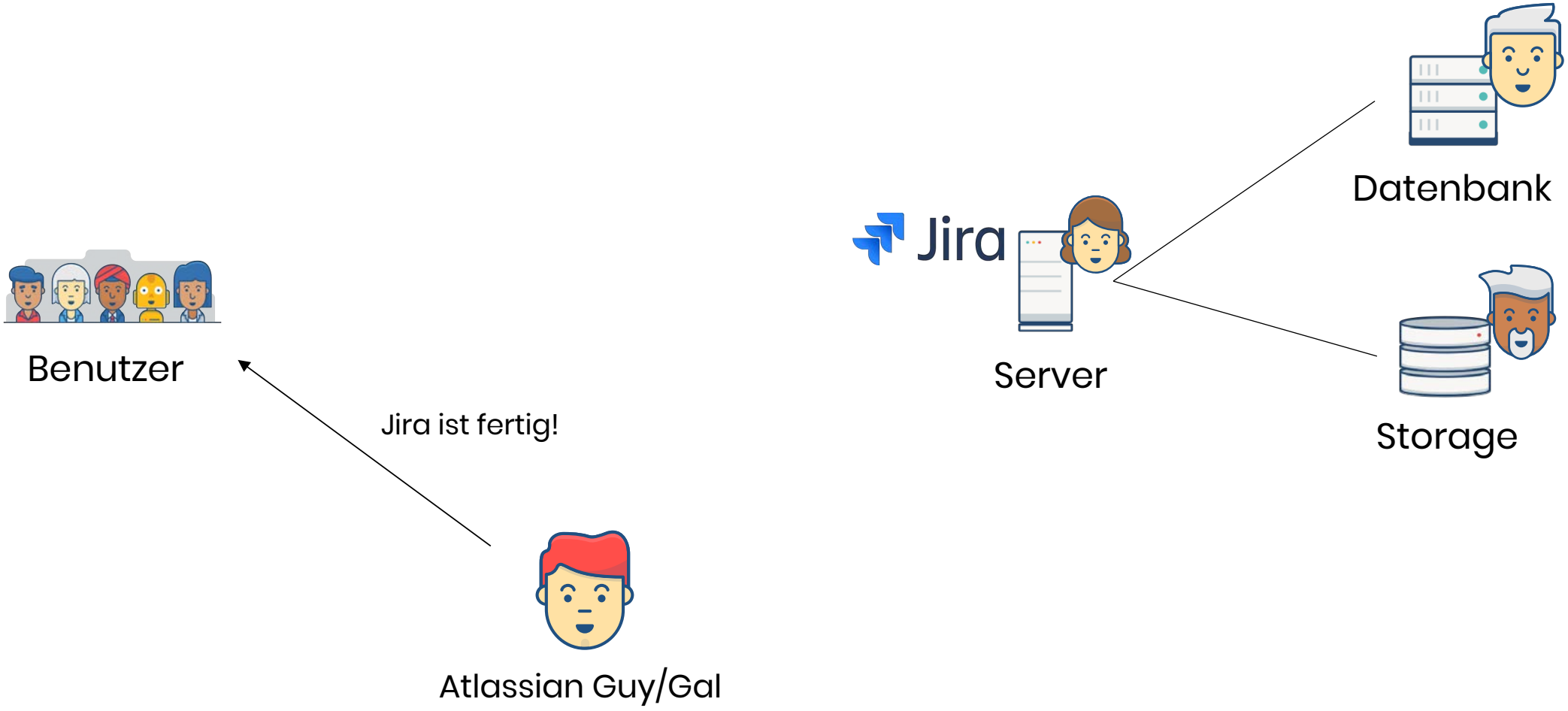
Jira Deployment in 2018



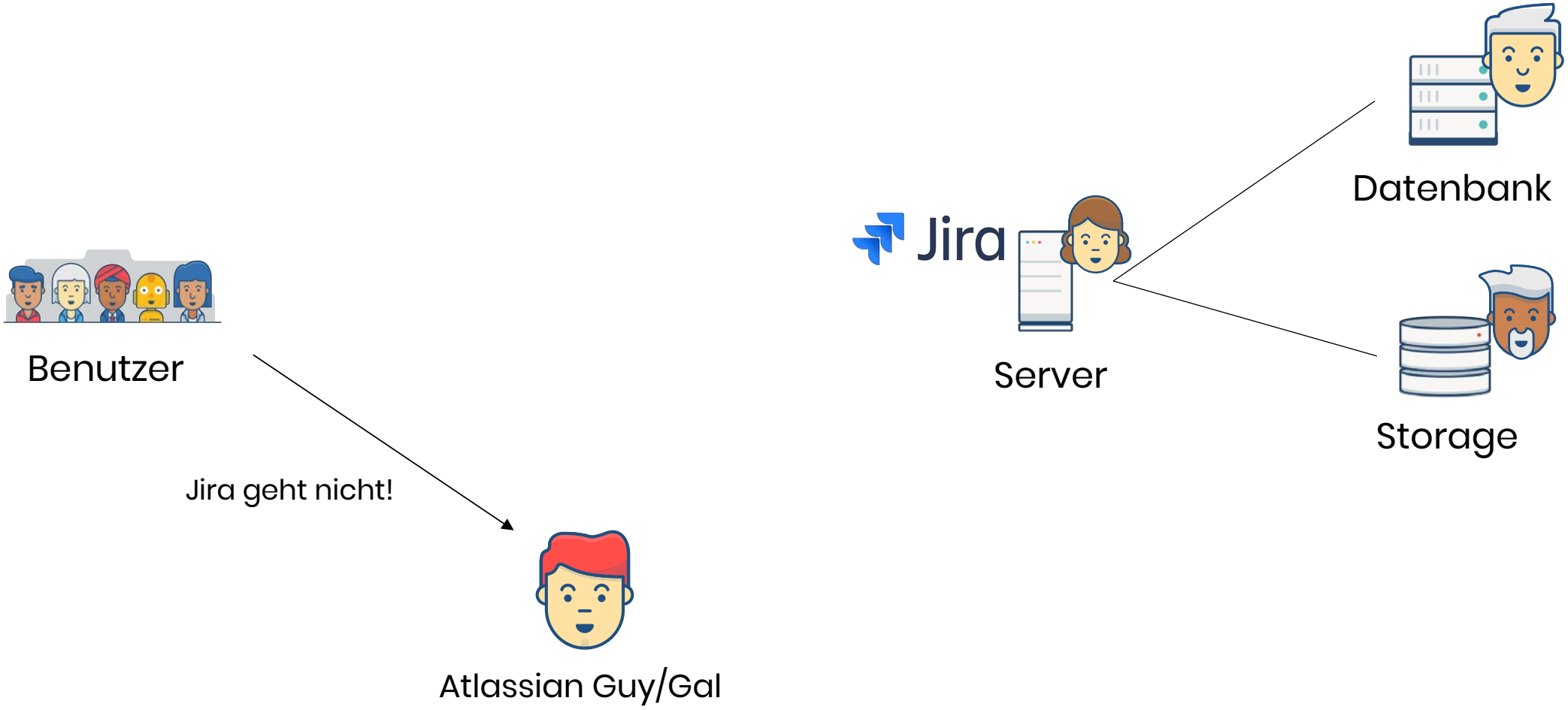
Jira Deployment in 2018



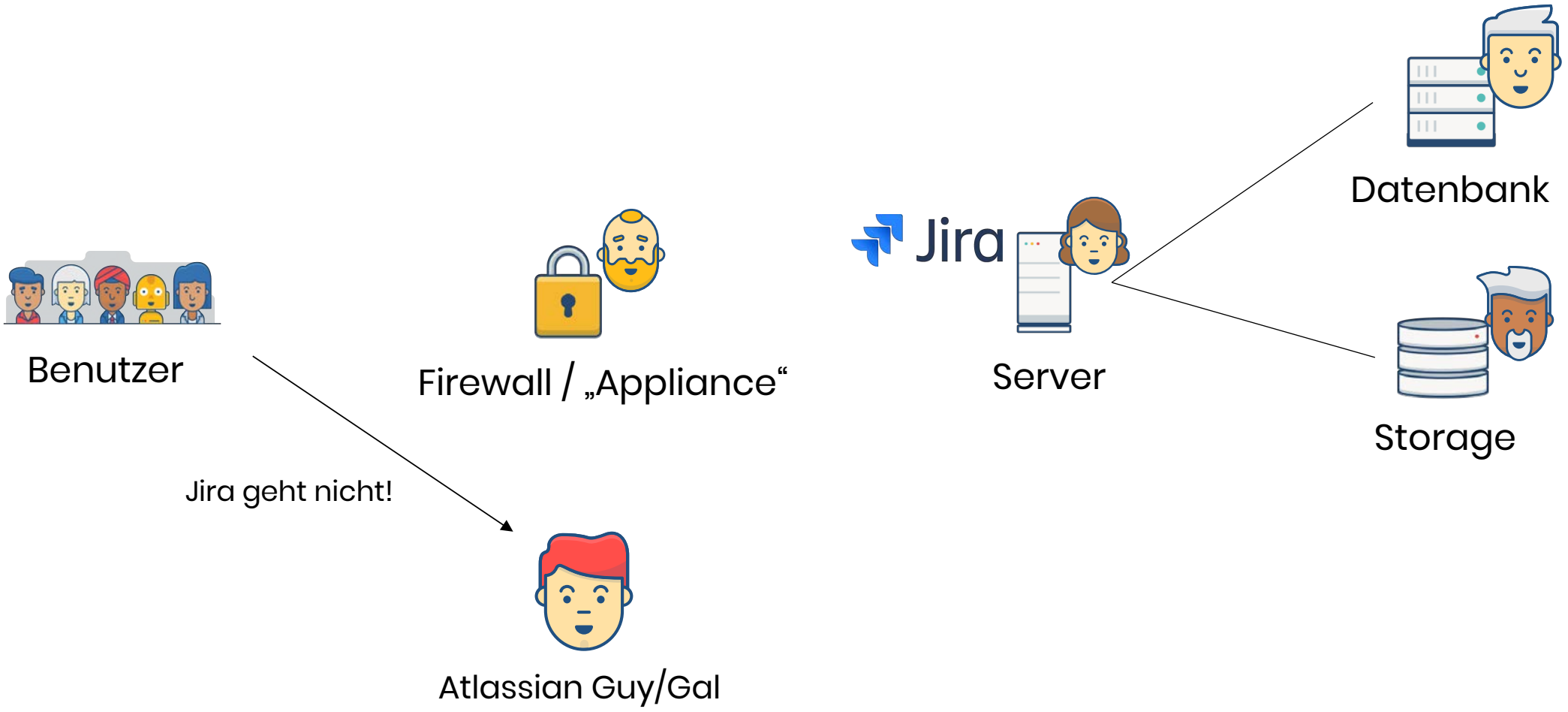
Jira Deployment in 2018



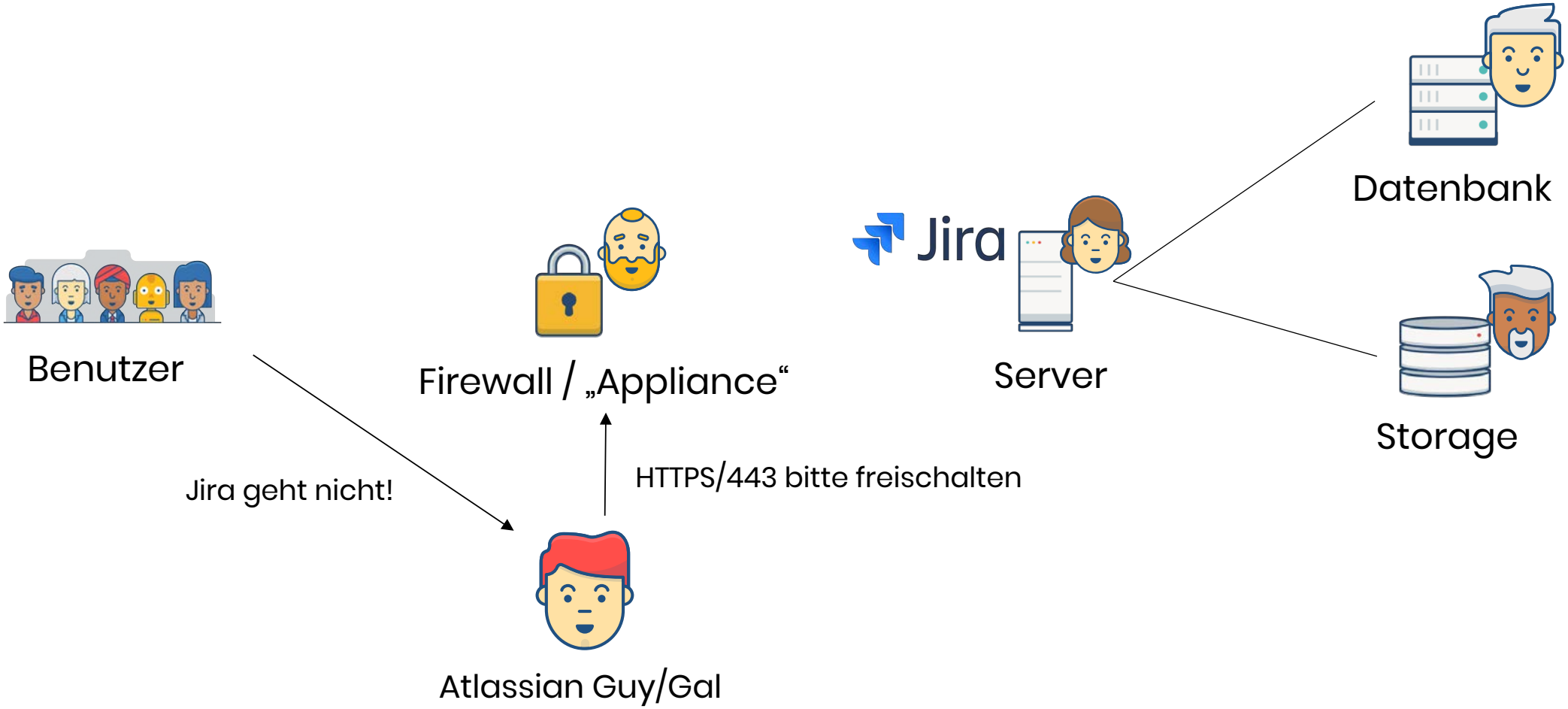
Jira Deployment in 2018



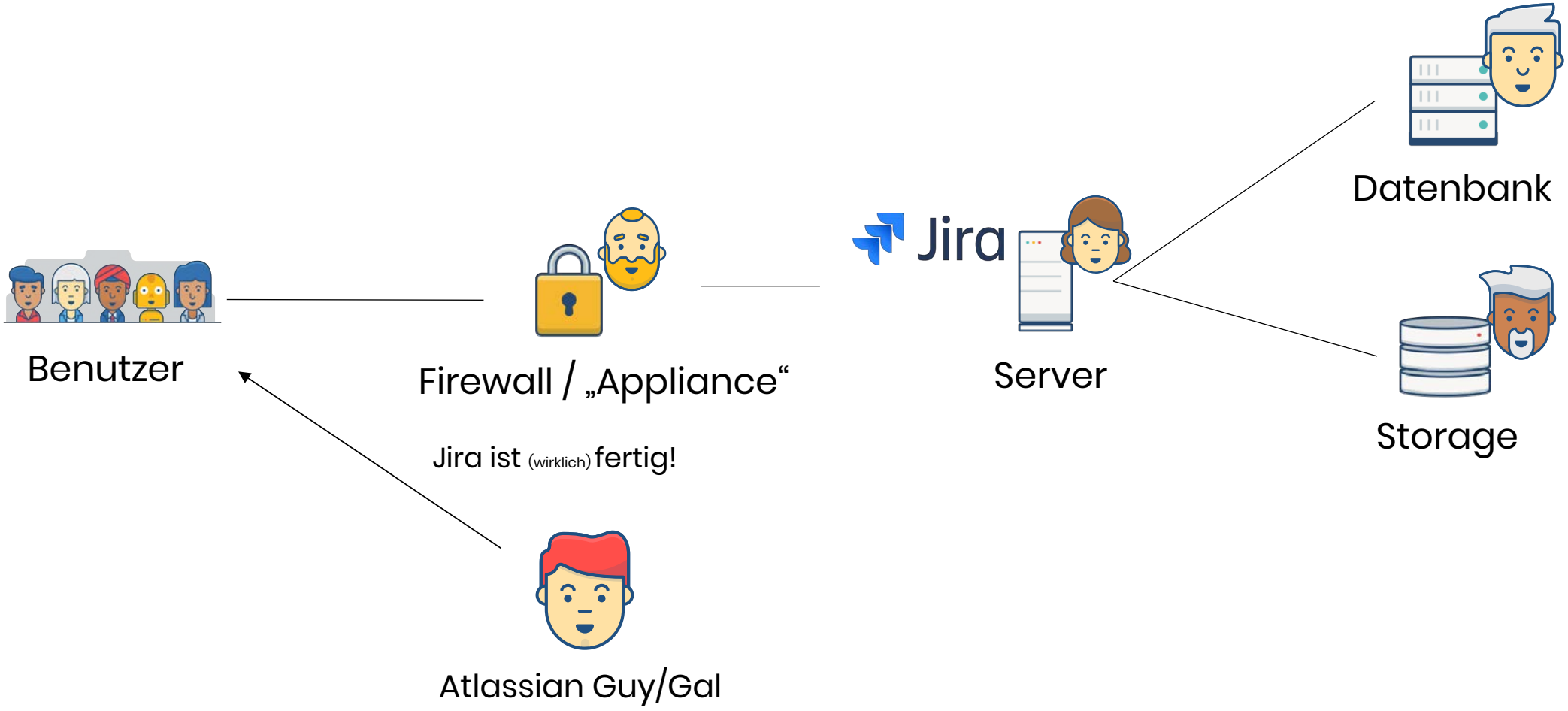
Jira Deployment in 2018



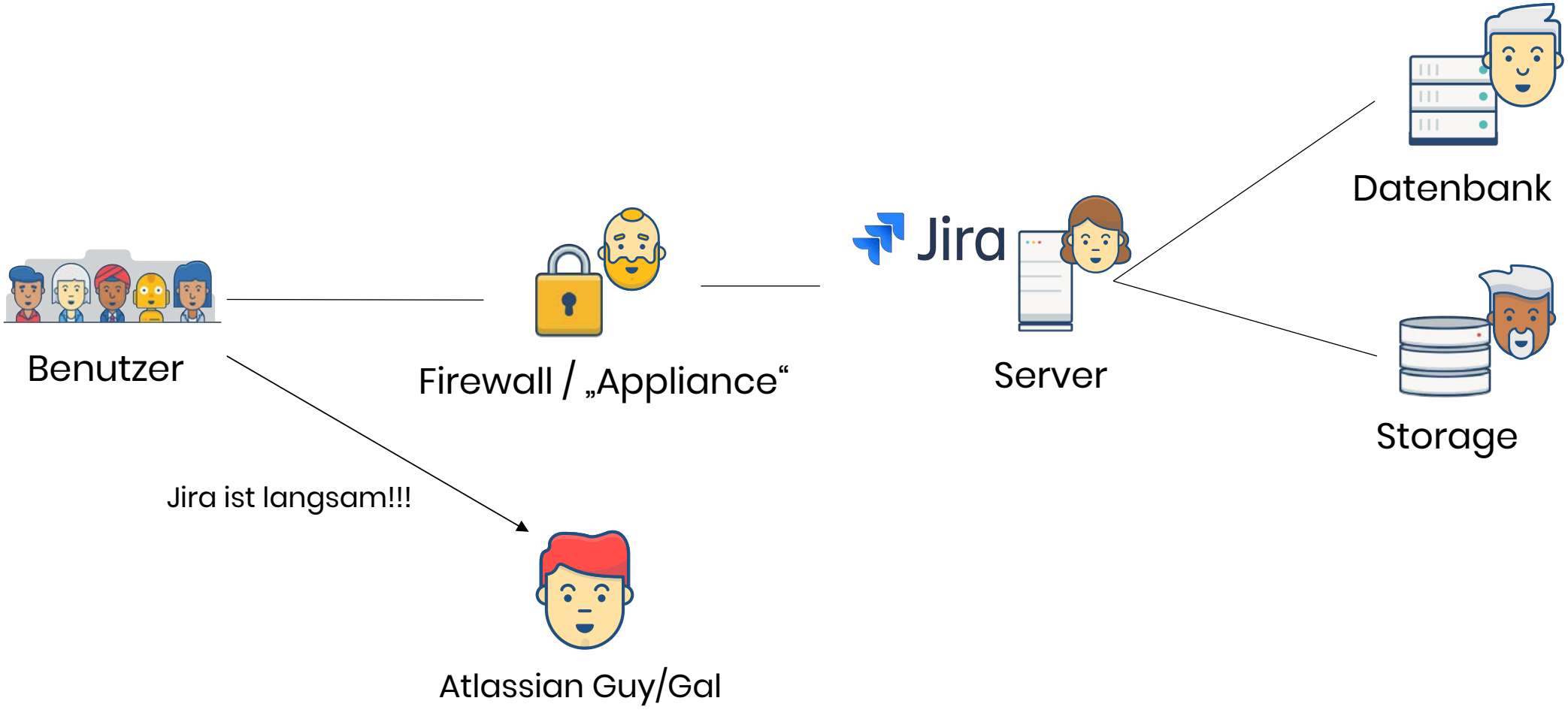
Jira Deployment in 2018



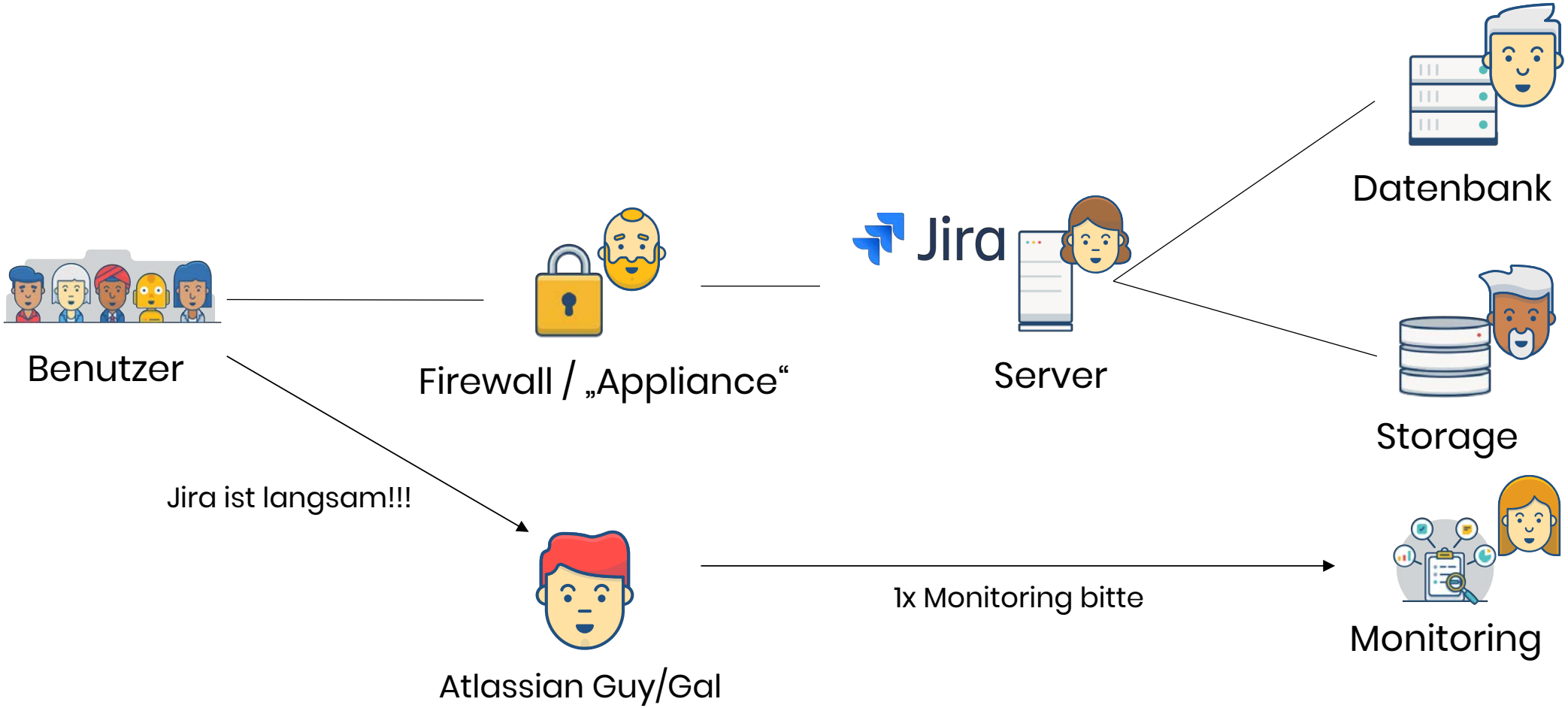
Jira Deployment in 2018



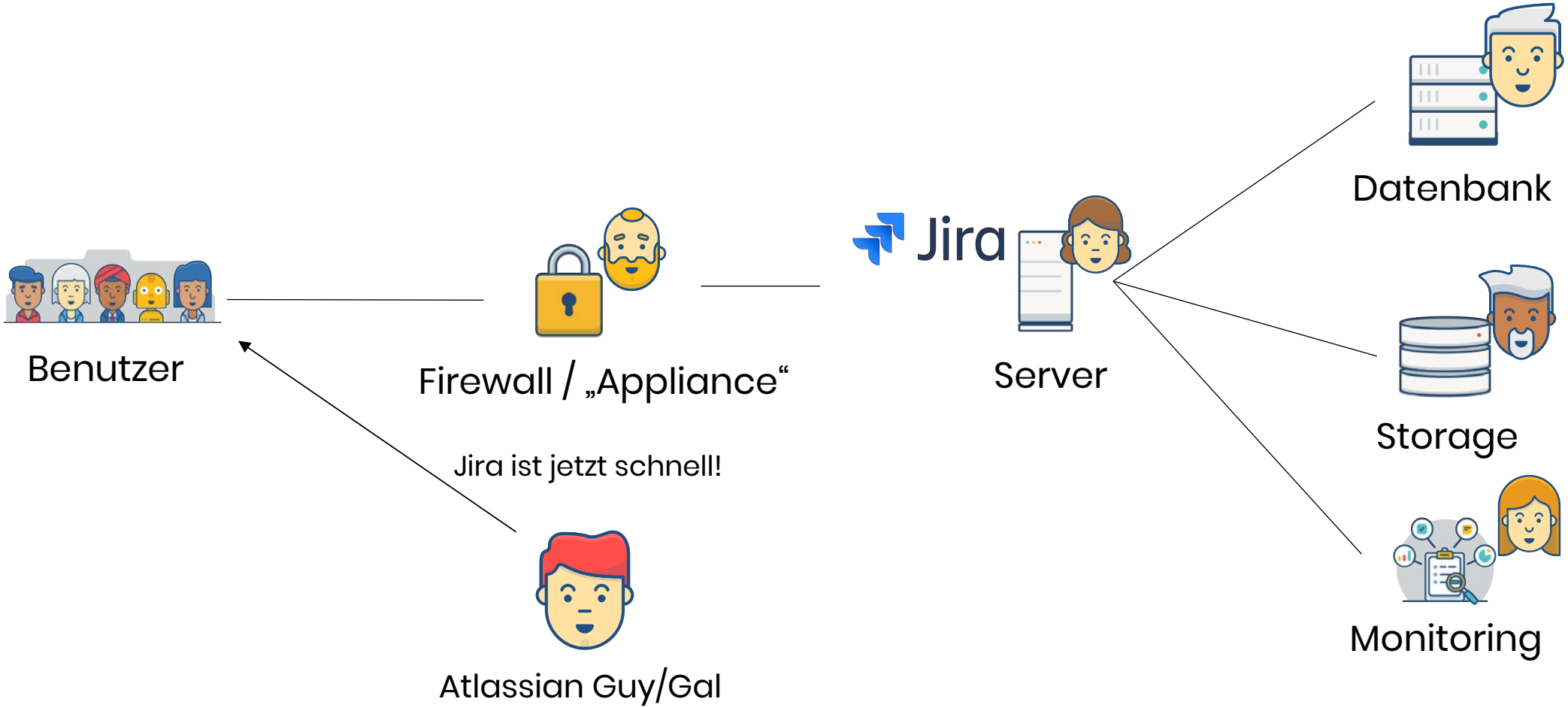
Jira Deployment in 2018



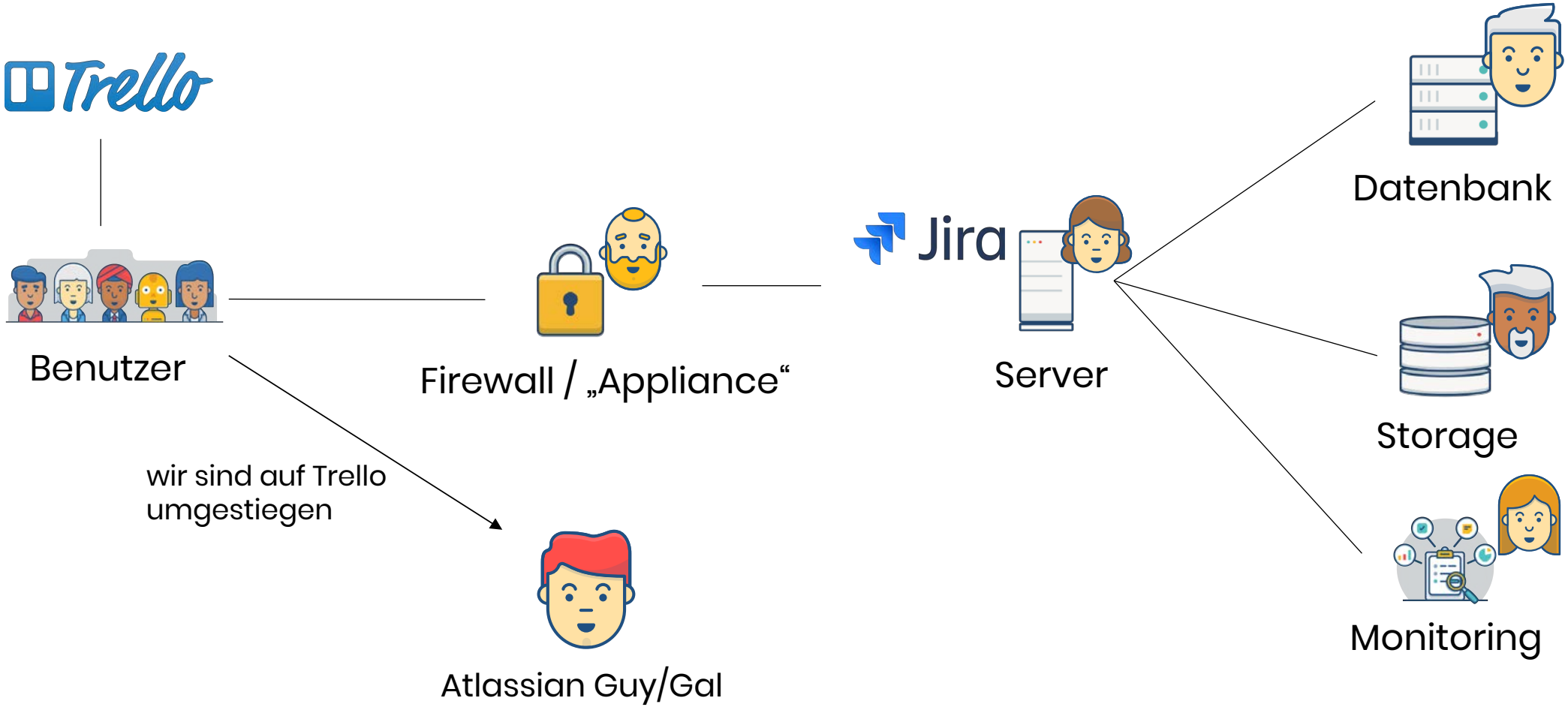
Jira Deployment in 2018



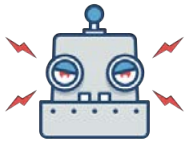
Jira Deployment in 2018



Jira Deployment in 2018



Probleme

Lange Vorlaufzeiten bei Änderungen 

Abhängigkeit bei der Konfiguration von zentralen Diensten (Datenbank, Load Balancer)

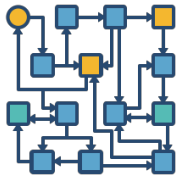
Pro Dienst eine eigene Maschine → CPU/RAM Overhead

Ausliefern von Änderungen vs. Zustand

- Nachvollziehbarkeit nur durch Protokollführung gegeben

Möglicher Ansatz: IT Automation 

z.B. Puppet, Ansible oder Chef



Custom-Lösungen, bei denen für jede Änderung ein Experte notwendig ist

Abstrahierung auf Betriebssystem-Ebene, statt Service-Ebene

Keine „Turnkey“ Lösungen

Hochverfügbarkeit und Skalierung müssen trotzdem gelöst werden

Private Cloud

Cloud Computing Dienste, die intern verfügbar sind

bietet die Vorteile der Public Cloud, mit mehr Kontrolle

- Self-Service
- Skalierbarkeit

Infrastructure-as-a-Service — Platform-as-a-Service

Low-Level — High-Level

Private Cloud

Cloud Computing Dienste, die intern verfügbar sind

bietet die Vorteile der Public Cloud, mit mehr Kontrolle

- Self-Service
- Skalierbarkeit

Infrastructure-as-a-Service **Containers-as-a-Service** Platform-as-a-Service

Low-Level High-Level

Containers-as-a-Service

 Kubernetes (K8s) = „Container Orchestration“

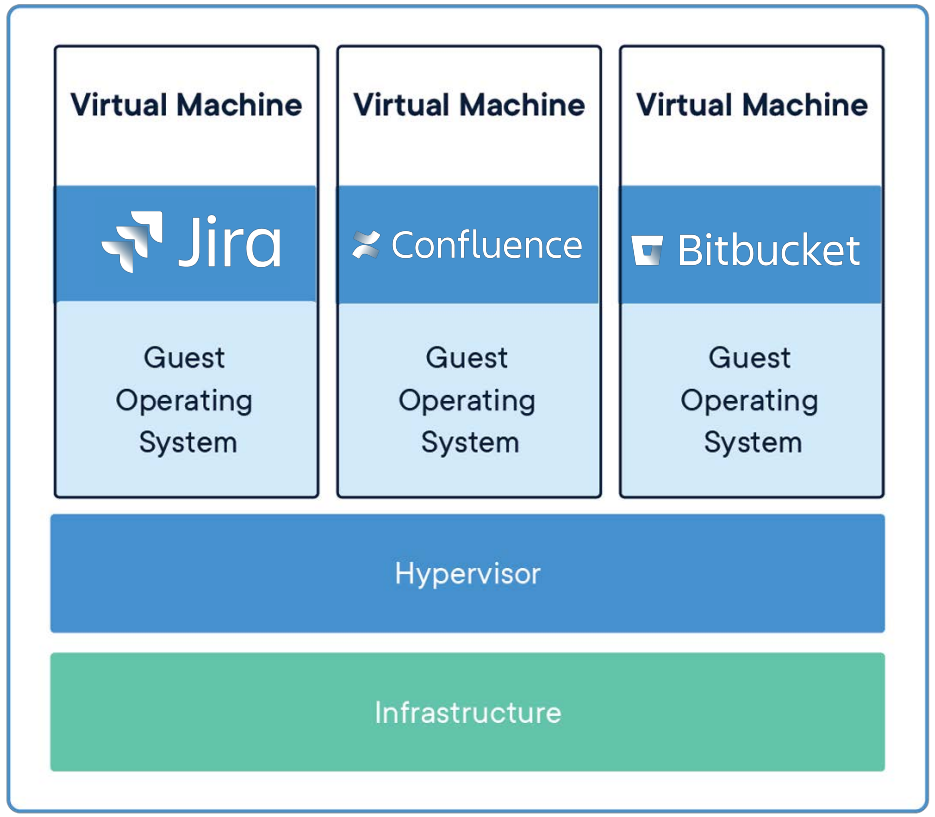
Warum Kubernetes?

- von Google entwickelt
- wird von der Cloud Native Computing Foundation betreut
- hat sich als Standard etabliert

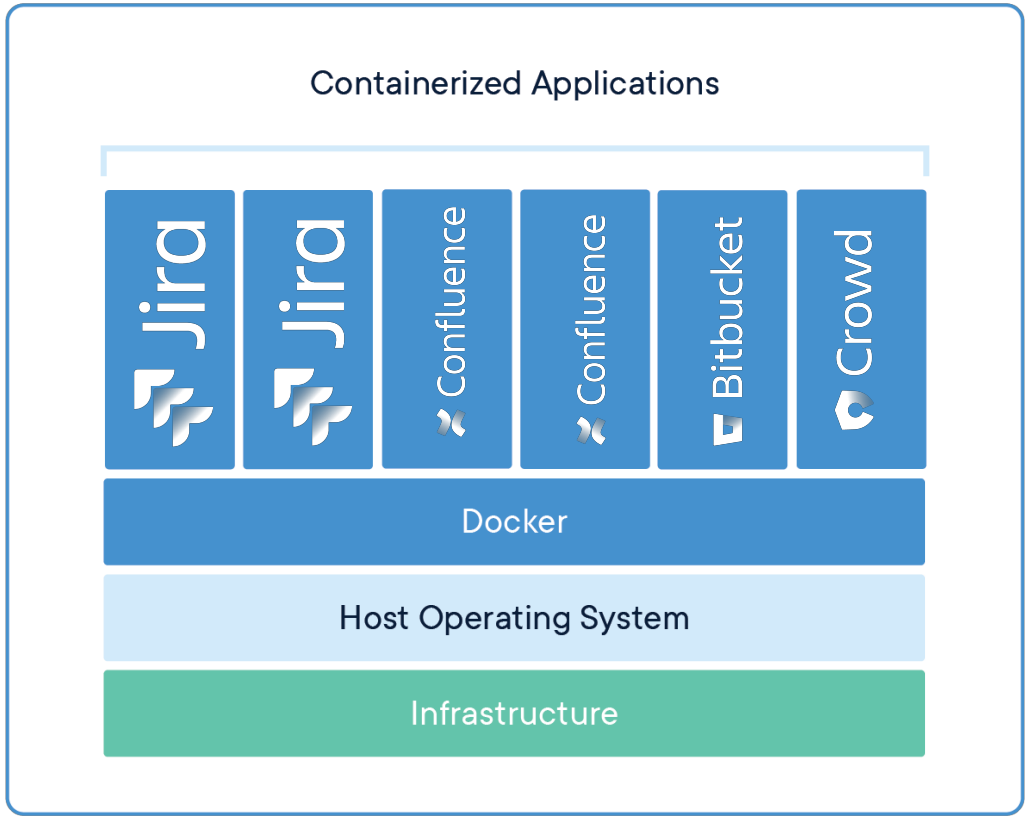
Cloud Native Ansatz

Pictures taken from https://github.com/kubernetes/kubernetes

Virtual Machines

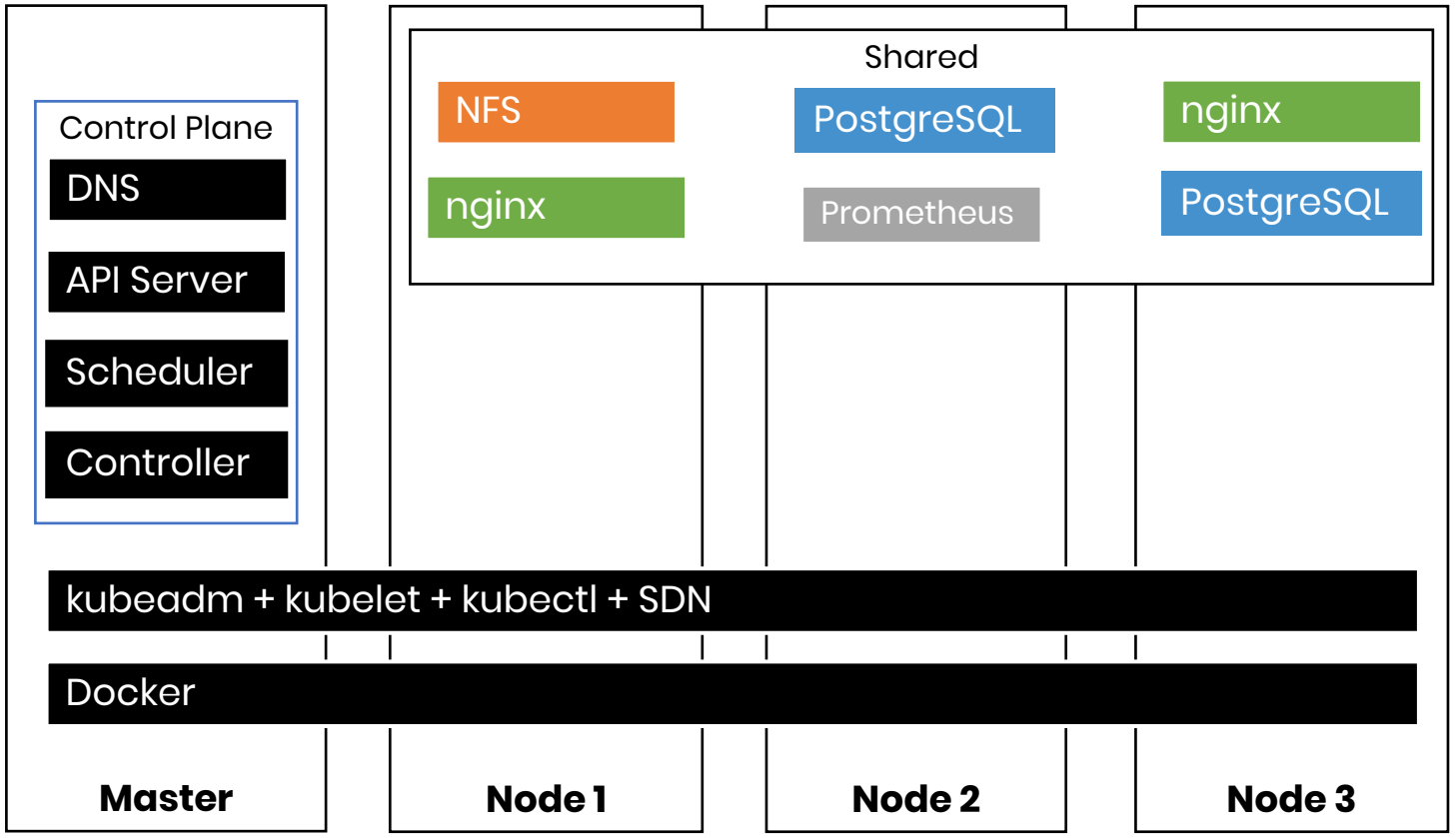


Containers

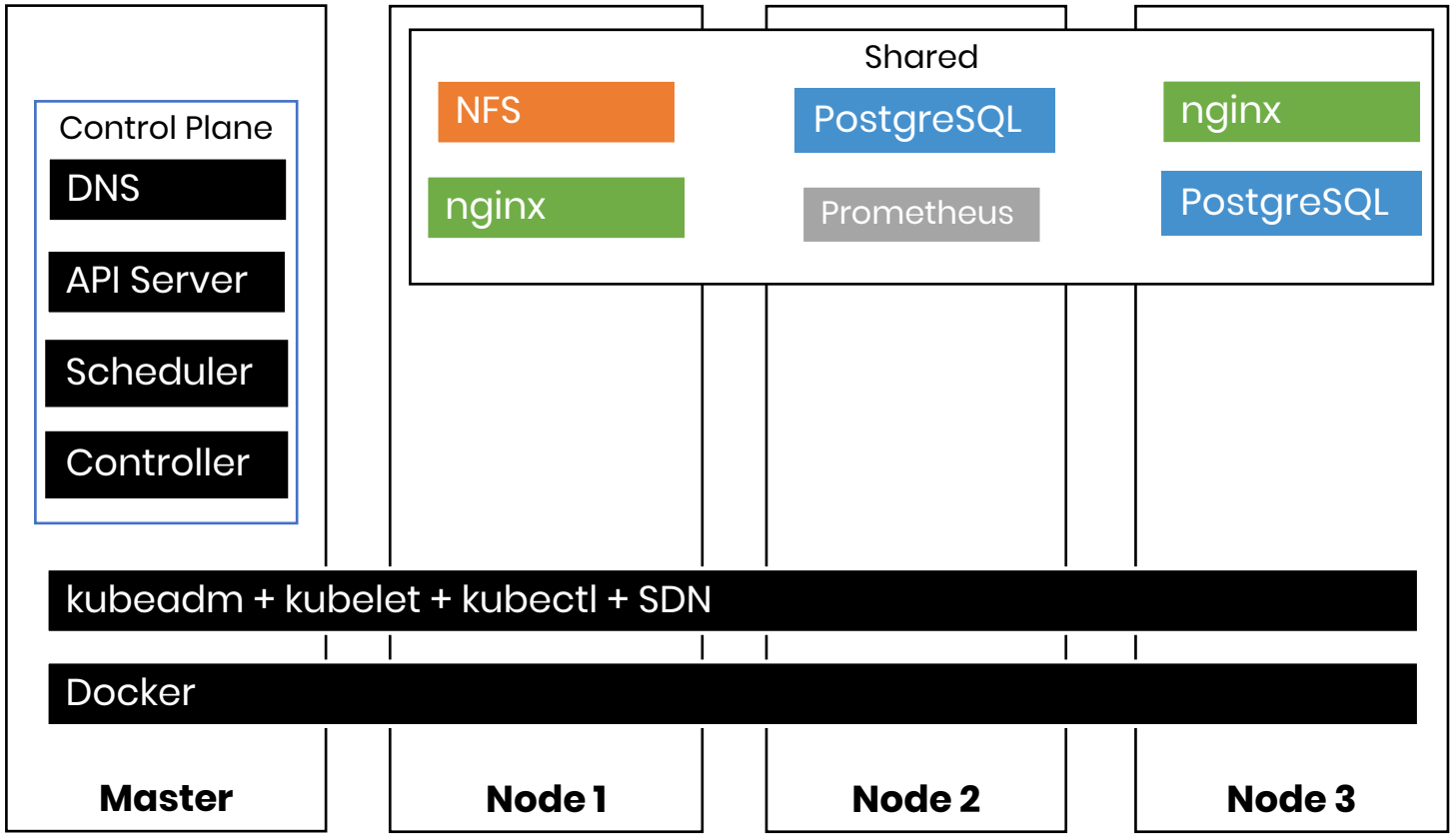


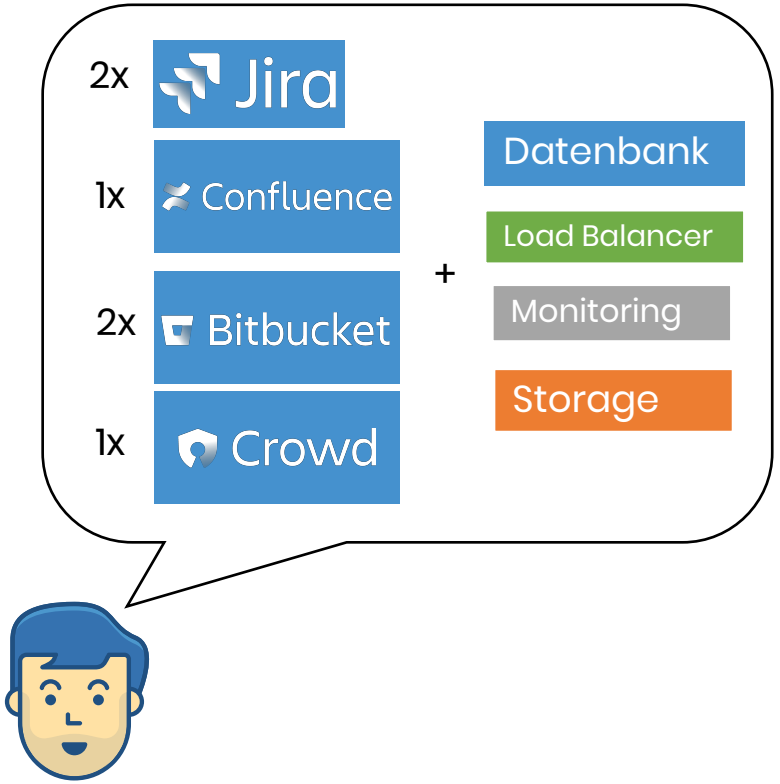
Pictures taken from https://www.docker.com/resources/what-container

Kubernetes

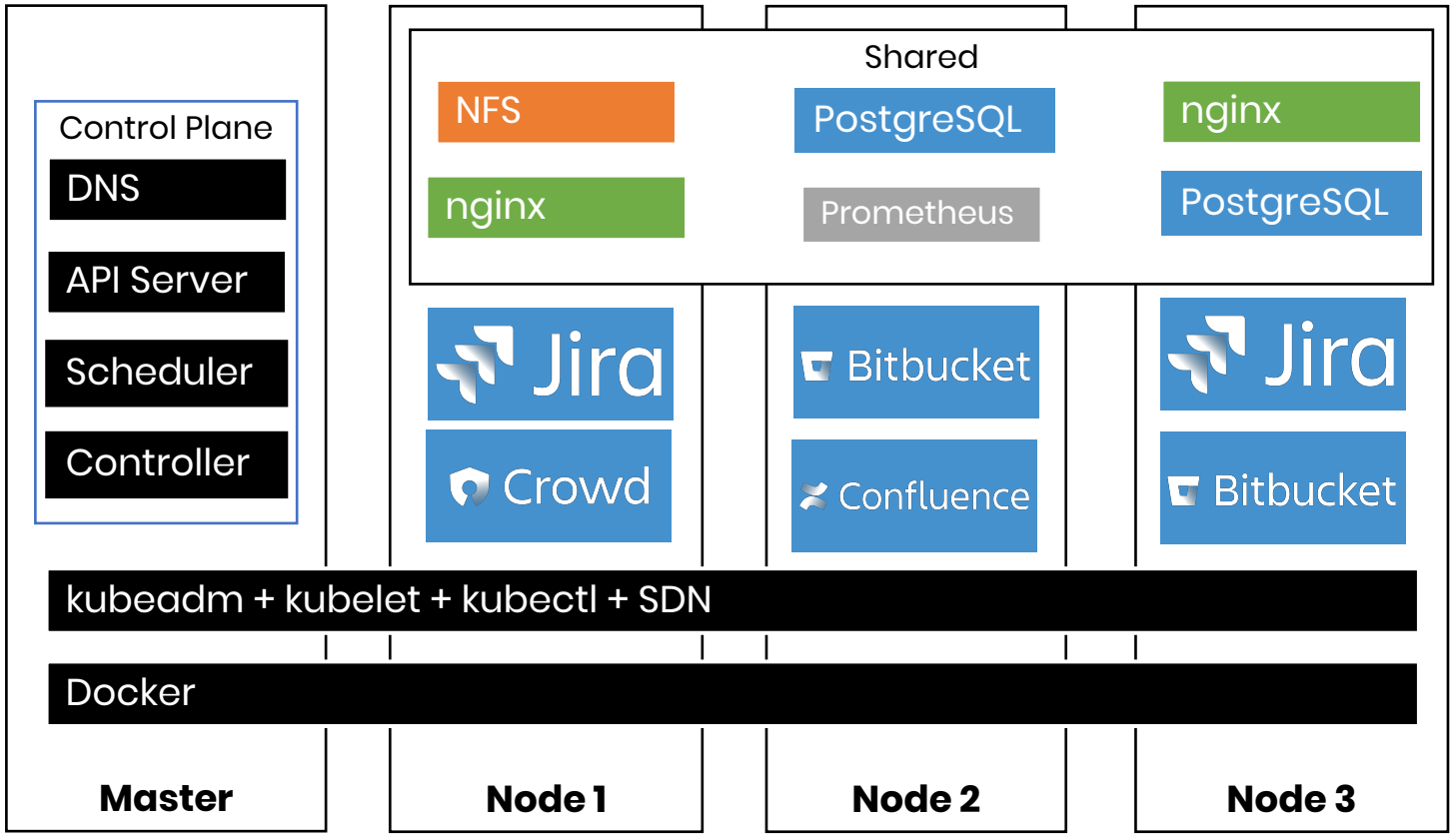

Control Plane
DNS
API Server
Scheduler
Controller
Shared
NFS
nginx
PostgreSQL
Prometheus
nginx
PostgreSQL
kubeadm + kubelet + kubectl + SDN
Docker
Master
Node 1
Node 2
Node 3

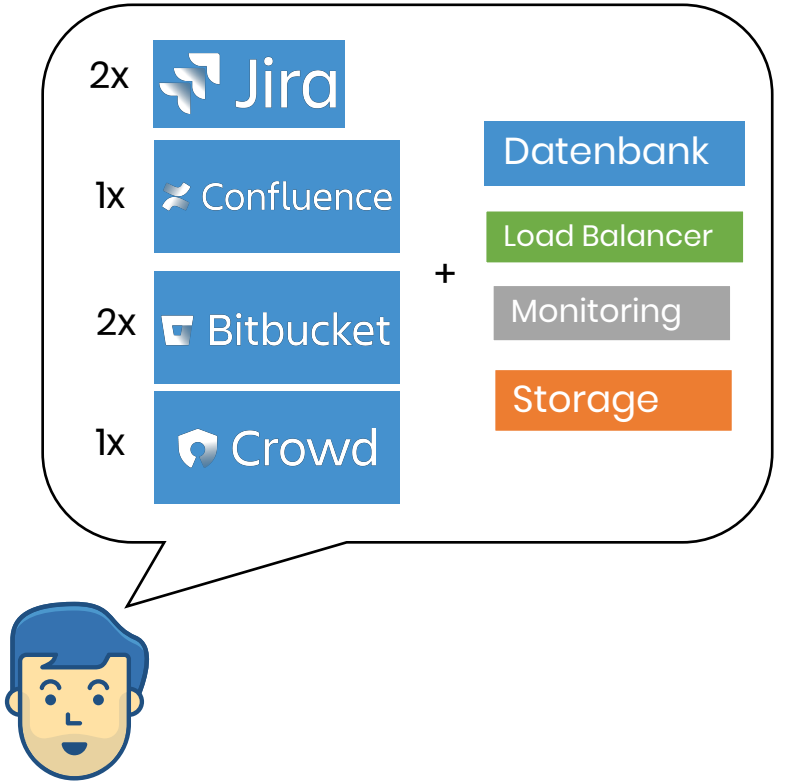
Kubernetes


Control Plane
DNS
API Server
Scheduler
Controller
Shared
NFS
nginx
PostgreSQL
Prometheus
nginx
PostgreSQL
kubeadm + kubelet + kubectl + SDN
Docker
Master
Node 1
Node 2
Node 3

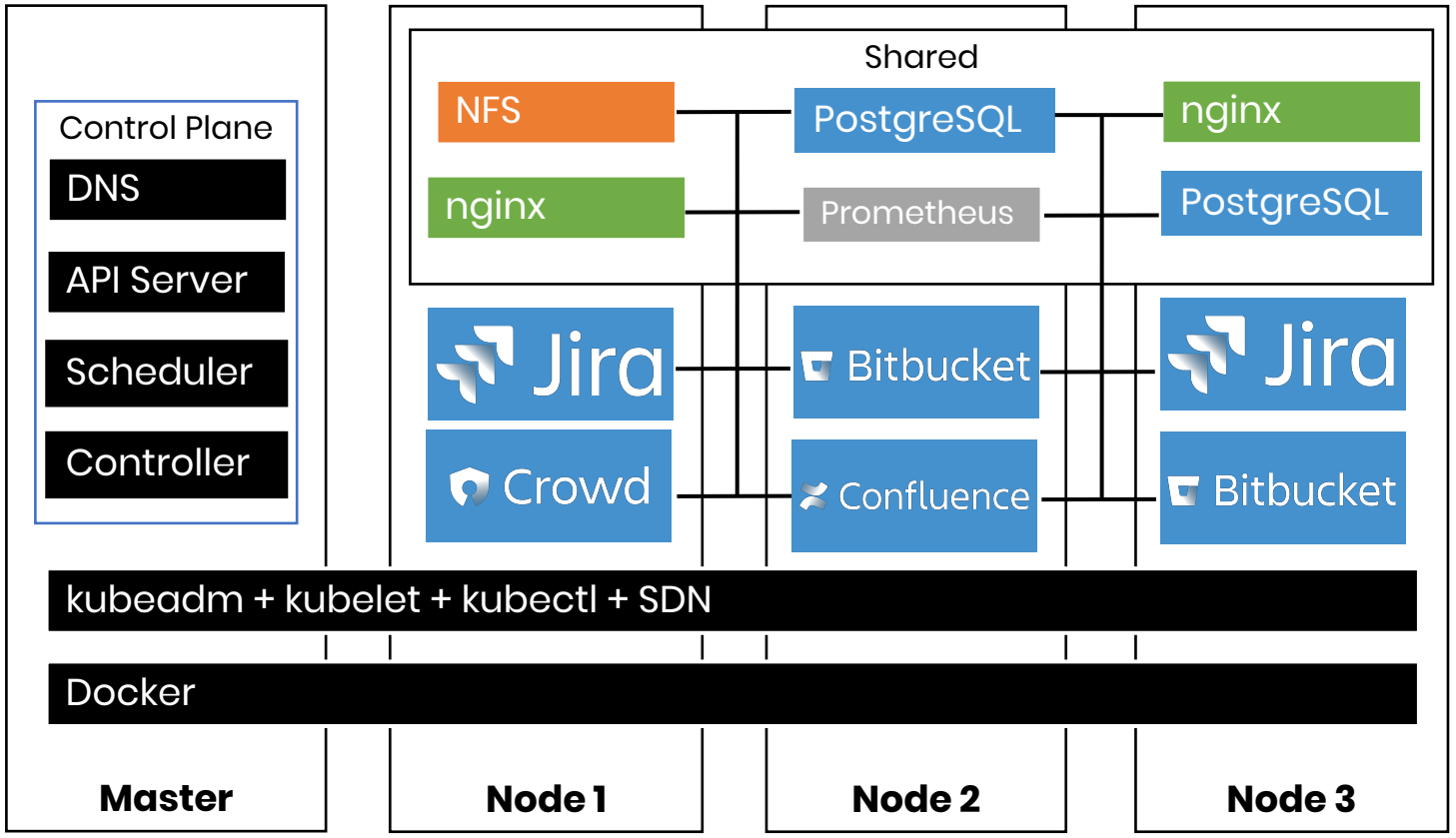

2x
Jira
1x
Confluence
2x
Bitbucket
1x
Crowd
+
Datenbank
Load Balancer
Monitoring
Storage

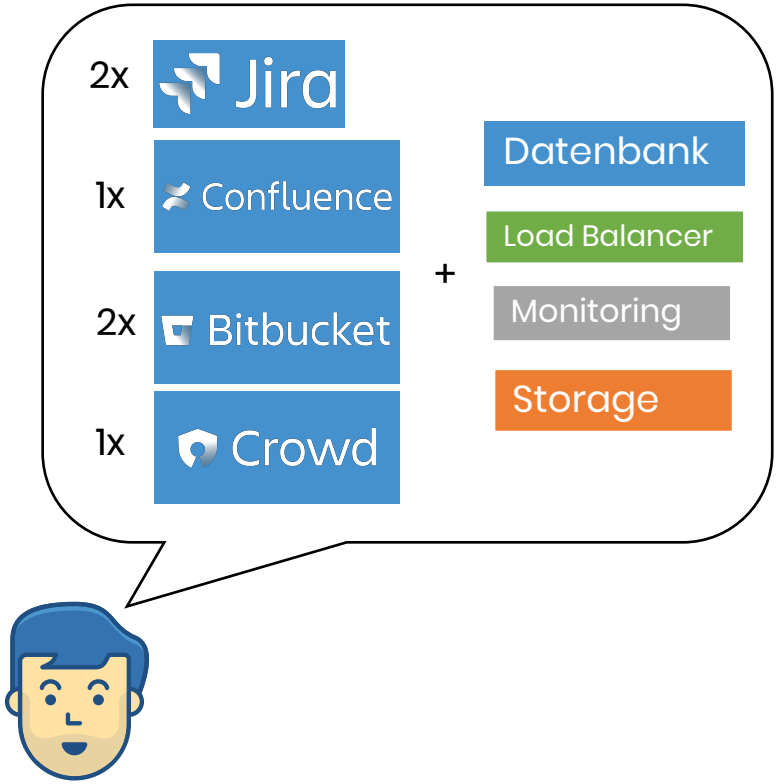
Kubernetes


Control Plane
DNS
API Server
Scheduler
Controller
Shared
NFS
nginx
PostgreSQL
Prometheus
nginx
PostgreSQL
Jira
Crowd
Bitbucket
Confluence
Jira
Bitbucket
kubeadm + kubelet + kubectl + SDN
Docker
Master
Node 1
Node 2
Node 3

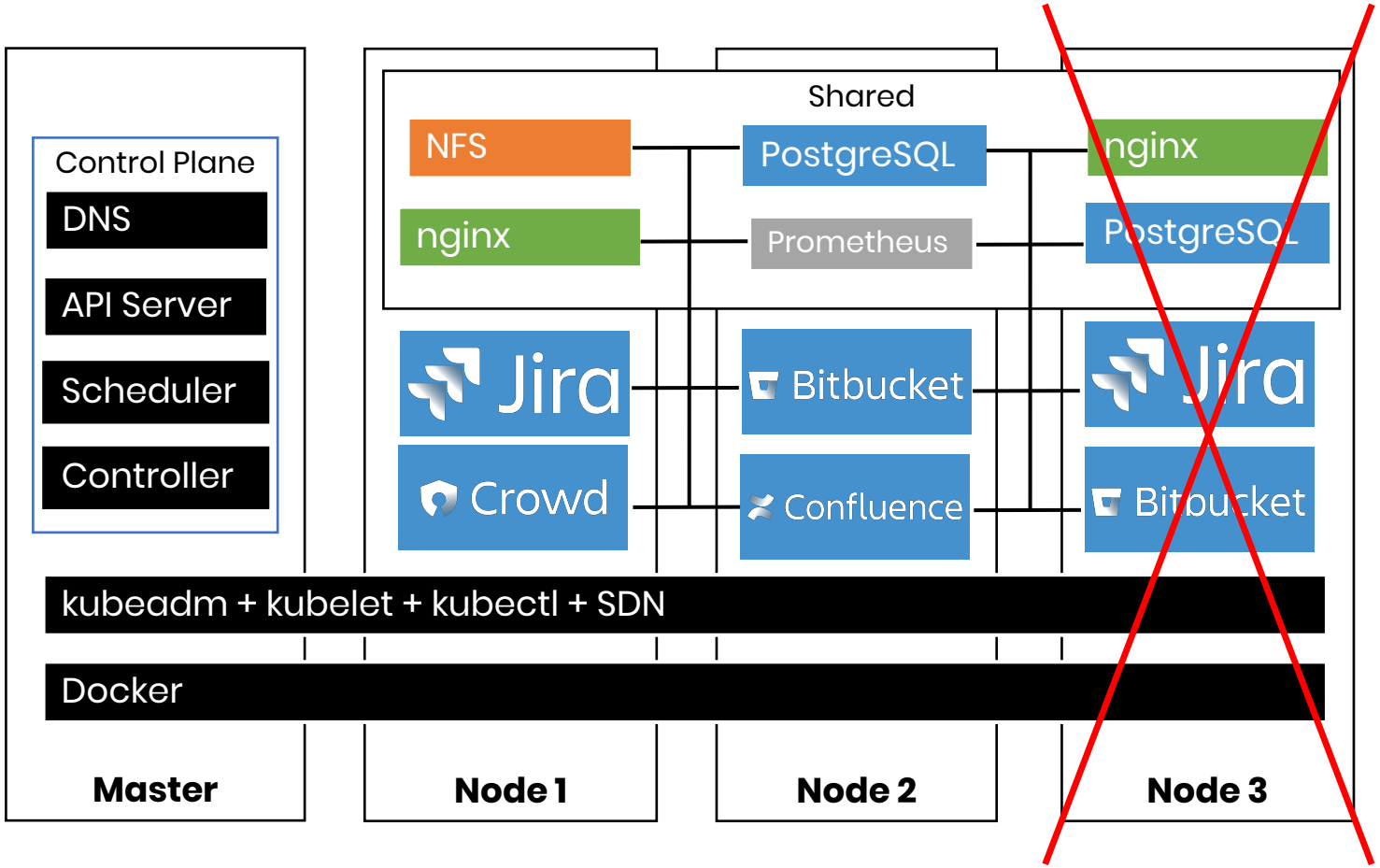

2x
Jira
1x
Confluence
2x
Bitbucket
1x
Crowd
+
Datenbank
Load Balancer
Monitoring
Storage

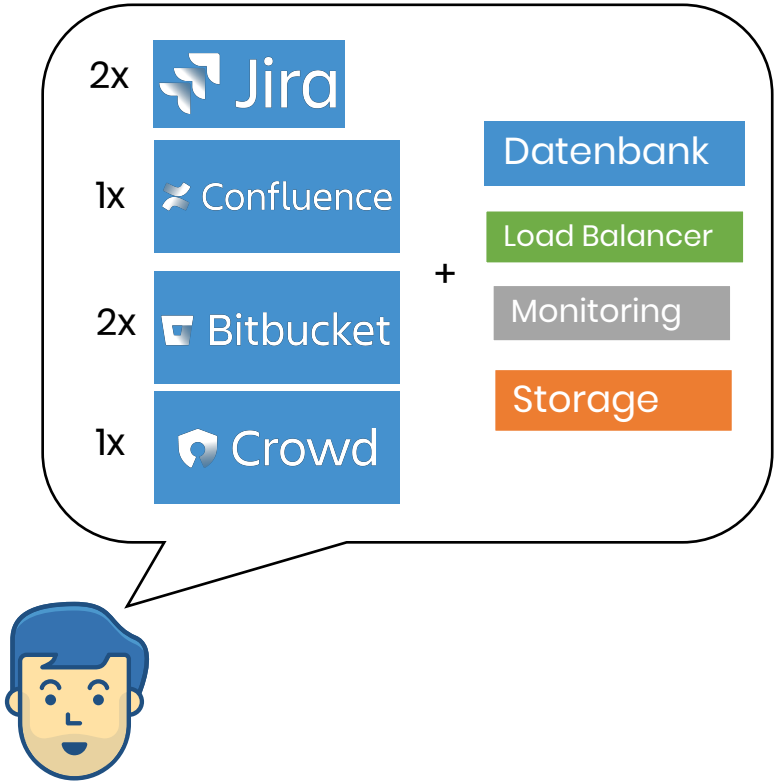
Kubernetes

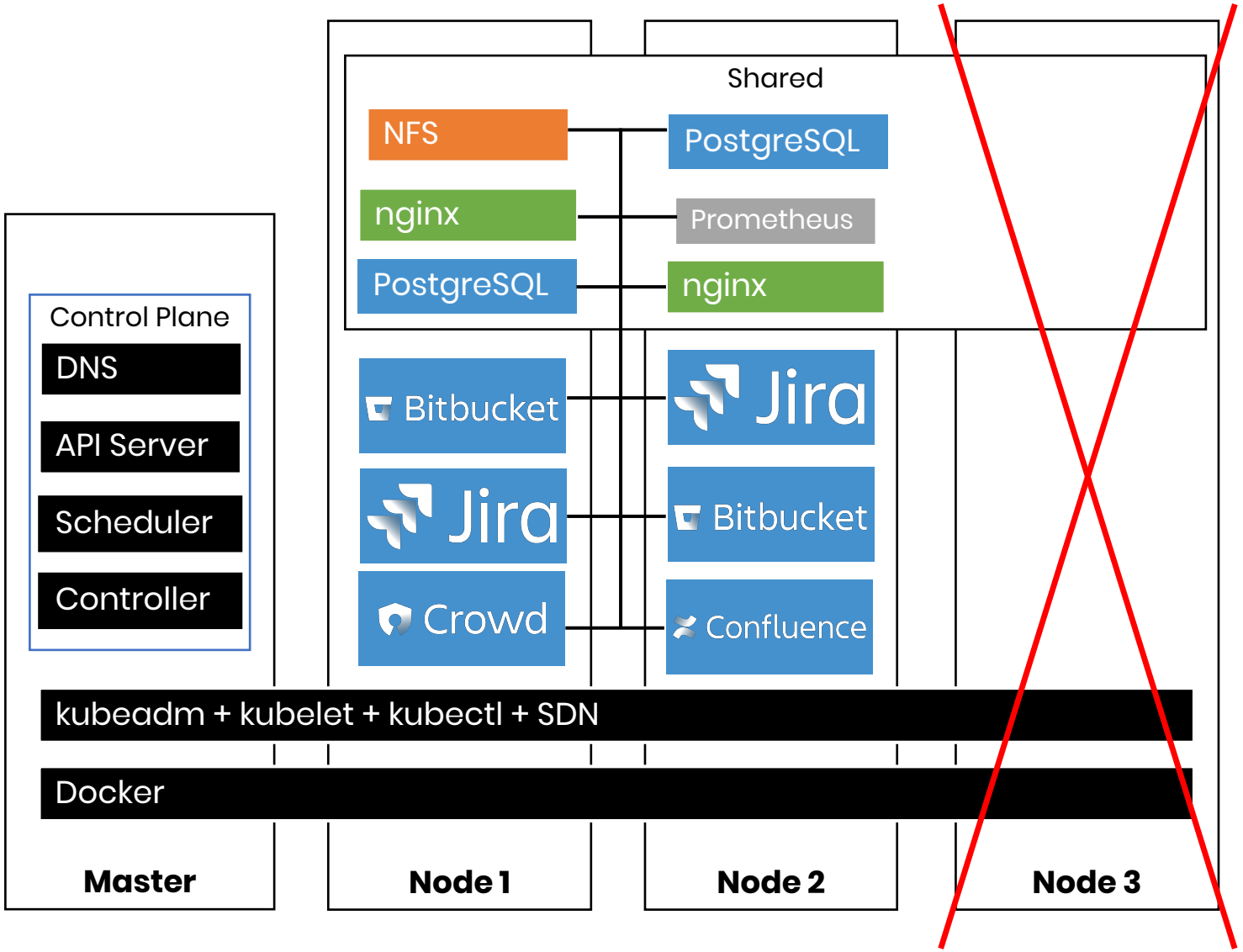

Control Plane
DNS
API Server
Scheduler
Controller
Shared
NFS
nginx
PostgreSQL
Prometheus
nginx
PostgreSQL
Jira
Crowd
Bitbucket
Confluence
Jira
Bitbucket
kubeadm + kubelet + kubectl + SDN
Docker
Master
Node 1
Node 2
Node 3

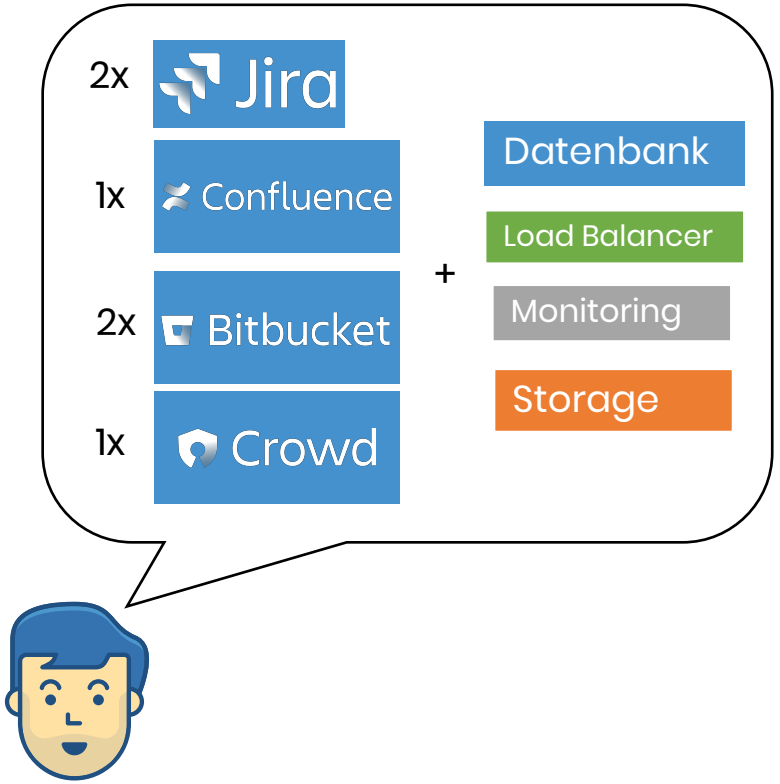

2x
Jira
1x
Confluence
2x
Bitbucket
1x
Crowd
+
Datenbank
Load Balancer
Monitoring
Storage

Kubernetes

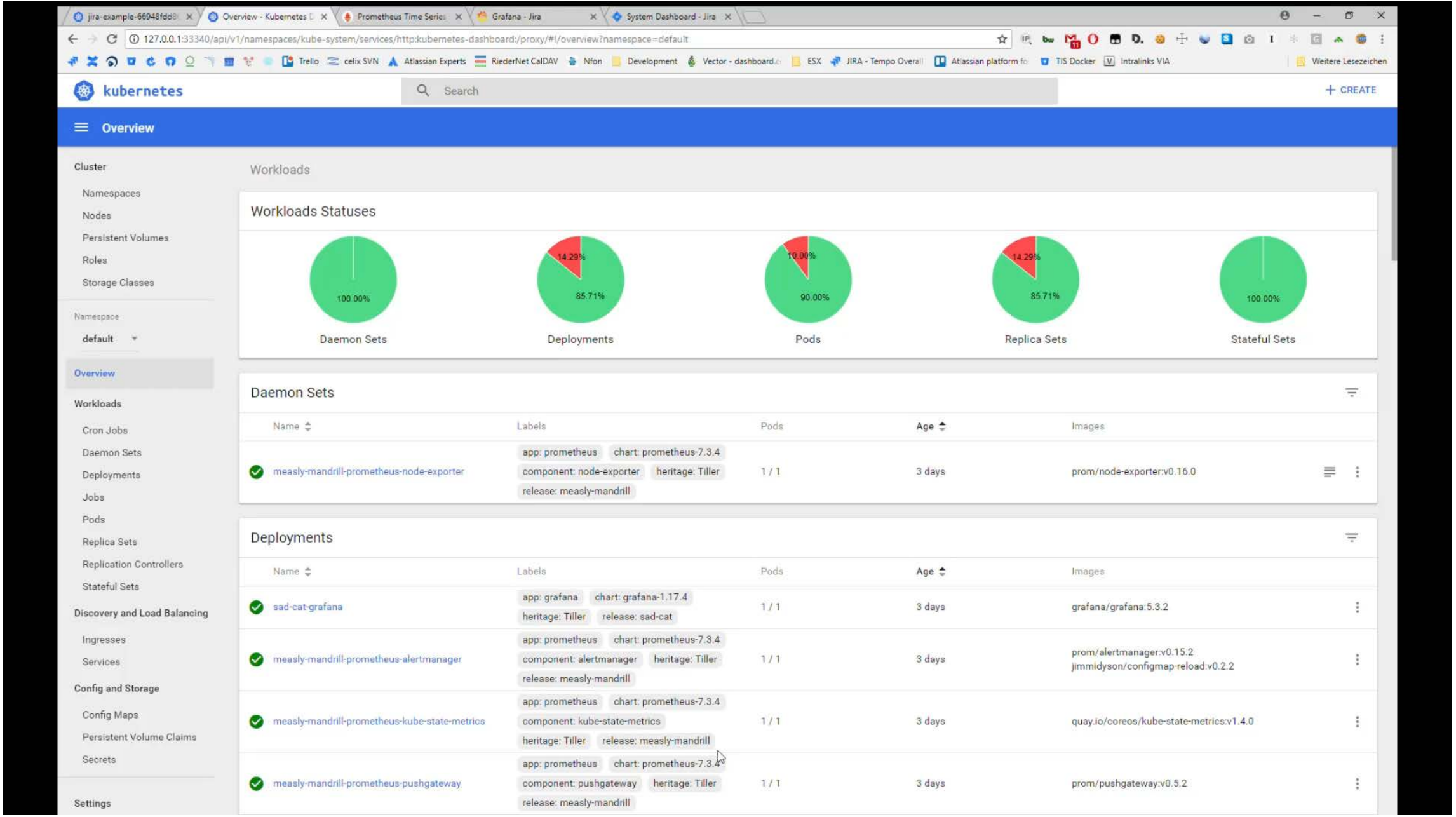

Control Plane
DNS
API Server
Scheduler
Controller
Shared
NFS
nginx
PostgreSQL
Prometheus
nginx
PostgreSQL
Jira
Crowd
Bitbucket
Confluence
Jira
Bitbucket
kubeadm + kubelet + kubectl + SDN
Docker
Master
Node 1
Node 2
Node 3

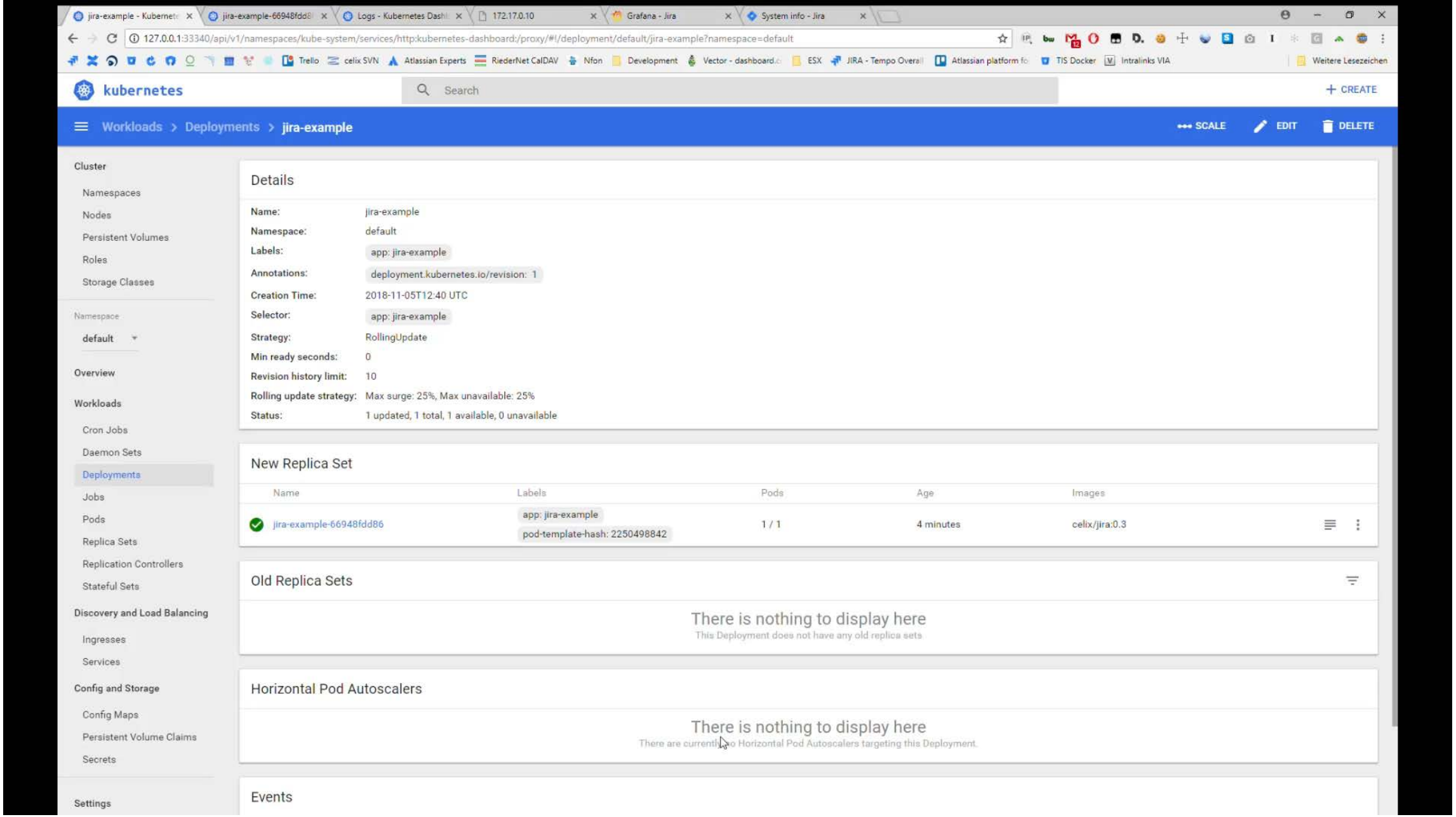

2x
Jira
1x
Confluence
2x
Bitbucket
1x
Crowd
+
Datenbank
Load Balancer
Monitoring
Storage


Shared
NFS
PostgreSQL
nginx
Prometheus
PostgreSQL
nginx
Control Plane
DNS
API Server
Scheduler
Controller
Bitbucket
Jira
Jira
Bitbucket
Crowd
Confluence
kubeadm + kubelet + kubectl + SDN
Docker
Master
Node 1
Node 2
Node 3


2x
Jira
1x
Confluence
2x
Bitbucket
1x
Crowd
+
Datenbank
Load Balancer
Monitoring
Storage

DEMO


jira-example-66948fdd8
Overview - Kubernetes
Prometheus Time Series
Grafana - Jira
System Dashboard - Jira
127.0.0.1:33340/api/v1/namespaces/kube-system/services/http:kubernetes-dashboard:/proxy/#!/overview?namespace=default
kubernetes
Search
+ CREATE
Overview
Cluster
Namespaces
Nodes
Persistent Volumes
Roles
Storage Classes
Namespace
default
Overview
Workloads
Cron Jobs
Daemon Sets
Deployments
Jobs
Pods
Replica Sets
Replication Controllers
Stateful Sets
Discovery and Load Balancing
Ingresses
Services
Config and Storage
Config Maps
Persistent Volume Claims
Secrets
Settings
Workloads
Workloads Statuses
100.00%
Daemon Sets
14.29%
85.71%
Deployments
10.00%
90.00%
Pods
14.29%
85.71%
Replica Sets
100.00%
Stateful Sets
Daemon Sets
Name
Labels
Pods
Age
Images
measly-mandrill-prometheus-node-exporter
app: prometheus
chart: prometheus-7.3.4
component: node-exporter
heritage: Tiller
release: measly-mandrill
1 / 1
3 days
prom/node-exporter:v0.16.0
Deployments
Name
Labels
Pods
Age
Images
sad-cat-grafana
app: grafana
chart: grafana-1.17.4
heritage: Tiller
release: sad-cat
1 / 1
3 days
grafana/grafana:5.3.2
measly-mandrill-prometheus-alertmanager
app: prometheus
chart: prometheus-7.3.4
component: alertmanager
heritage: Tiller
release: measly-mandrill
1 / 1
3 days
prom/alertmanager:v0.15.2
jimmidyson/configmap-reload:v0.2.2
measly-mandrill-prometheus-kube-state-metrics
app: prometheus
chart: prometheus-7.3.4
component: kube-state-metrics
heritage: Tiller
release: measly-mandrill
1 / 1
3 days
quay.io/coreos/kube-state-metrics:v1.4.0
measly-mandrill-prometheus-pushgateway
app: prometheus
chart: prometheus-7.3.4
component: pushgateway
heritage: Tiller
release: measly-mandrill
1 / 1
3 days
prom/pushgateway:v0.5.2


jira-example - Kubernetes
jira-example-66948fdd8
Logs - Kubernetes Dashb
172.17.0.10
Grafana - Jira
System info - Jira
127.0.0.1:33340/api/v1/namespaces/kube-system/services/http:kubernetes-dashboard:/proxy/#!/deployment/default/jira-example?namespace=default
Trello
celix SVN
Atlassian Experts
RiederNet CalDAV
Nfon
Development
Vector - dashboard.c
ESX
JIRA - Tempo Overall
Atlassian platform fo
TIS Docker
Intralinks VIA
Weitere Lesezeichen
kubernetes
Search
+ CREATE
Workloads > Deployments > jira-example
SCALE
EDIT
DELETE
Cluster
Namespaces
Nodes
Persistent Volumes
Roles
Storage Classes
Namespace
default
Overview
Workloads
Cron Jobs
Daemon Sets
Deployments
Jobs
Pods
Replica Sets
Replication Controllers
Stateful Sets
Discovery and Load Balancing
Ingresses
Services
Config and Storage
Config Maps
Persistent Volume Claims
Secrets
Settings
Details
Name: jira-example
Namespace: default
Labels: app: jira-example
Annotations: deployment.kubernetes.io/revision: 1
Creation Time: 2018-11-05T12:40 UTC
Selector: app: jira-example
Strategy: RollingUpdate
Min ready seconds: 0
Revision history limit: 10
Rolling update strategy: Max surge: 25%, Max unavailable: 25%
Status: 1 updated, 1 total, 1 available, 0 unavailable
New Replica Set
Name
Labels
Pods
Age
Images
jira-example-66948fdd86
app: jira-example
pod-template-hash: 2250498842
1 / 1
4 minutes
celix/jira:0.3
Old Replica Sets
There is nothing to display here
This Deployment does not have any old replica sets
Horizontal Pod Autoscalers
There is nothing to display here
There are currently no Horizontal Pod Autoscalers targeting this Deployment.
Events

Kubernetes-as-a-Service

Self-Hosted:

- Plain Kubernetes 
- RedHat / IBM OpenShift 

Cloud Provider:

- Amazon EKS – Managed Kubernetes Service
- Google Kubernetes Engine
- Microsoft Azure Kubernetes Service

Pictures taken from https://www.openshift.com/ and https://github.com/kubernetes/kubernetes

Atlassian Cloud vs. Private Cloud

	Atlassian Cloud	K8s
Ort	Public Cloud	Public Cloud oder Private Cloud
Anbieter	Amazon	Self-Hosted, Amazon, Microsoft, Google
Services	Jira, Confluence, Bitbucket	alle Container (Atlassian, Eigenentwicklungen, ...)
Upgrades	automatisch	händisch
Hochverfügbarkeit	liegt an Atlassian	ja
Benutzer	max. 5000	∞
Addons	Cloud only	Server only
Betrieb **Security** **Backups** **Compliance**	Atlassian	händisch

Vorteile der Private Cloud mit K8s

Self-Service

Auslieferung eines Zustandes anstatt von Änderungen

Services sind vollständig (!) in Code definiert

Dienste sind von der zu Grunde liegenden Infrastruktur entkoppelt

- z.B. Verschlüsselung von Traffic transparent für die Dienste (SDN)

Operations kümmert sich um den Betrieb der Infrastruktur, statt um das Anlegen von Datenbanken

Nachteile der Private Cloud mit K8s

Container pro Dienst notwendig

Neue Tools zu lernen

On-Premise: Betrieb eines Kubernetes Clusters

Cloud Provider: Daten nicht In-House

Danke!