The bridge to possible

Kubernetes and You!

Josh Ingeniero, Technical Solutions Specialist

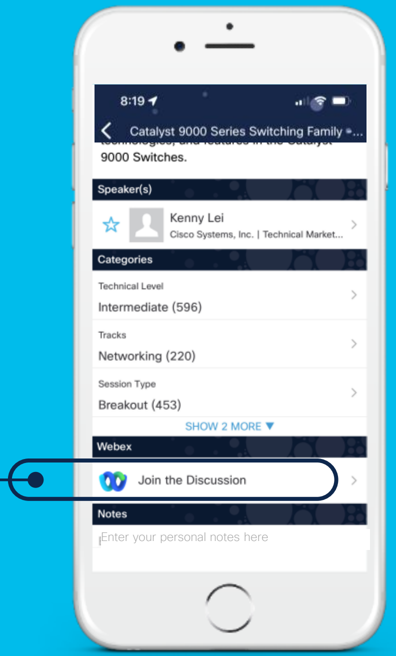
Cisco Webex App

Questions?

Use Cisco Webex App to chat
with the speaker after the session

How

1. Find this session in the Cisco Live Mobile App
2. Click “Join the Discussion”
3. Install the Webex App or go directly to the Webex space
4. Enter messages/questions in the Webex space



8:19

Catalyst 9000 Series Switching Family •...

9000 Switches.

Speaker(s)

☆ Kenny Lei
Cisco Systems, Inc. | Technical Market...

Categories

Technical Level
Intermediate (596)

Tracks
Networking (220)

Session Type
Breakout (453)

SHOW 2 MORE ▼

Webex

Join the Discussion

Notes

Enter your personal notes here

Webex spaces will be moderated until February 24, 2023.

Josh Ingeniero

- Technical Solutions Specialist, Cisco
- Cross-architecture Programmability and Automation
- Containerisation advocate
- CCNP, DevNet Professional, KCNA

github.com/joshingeniero

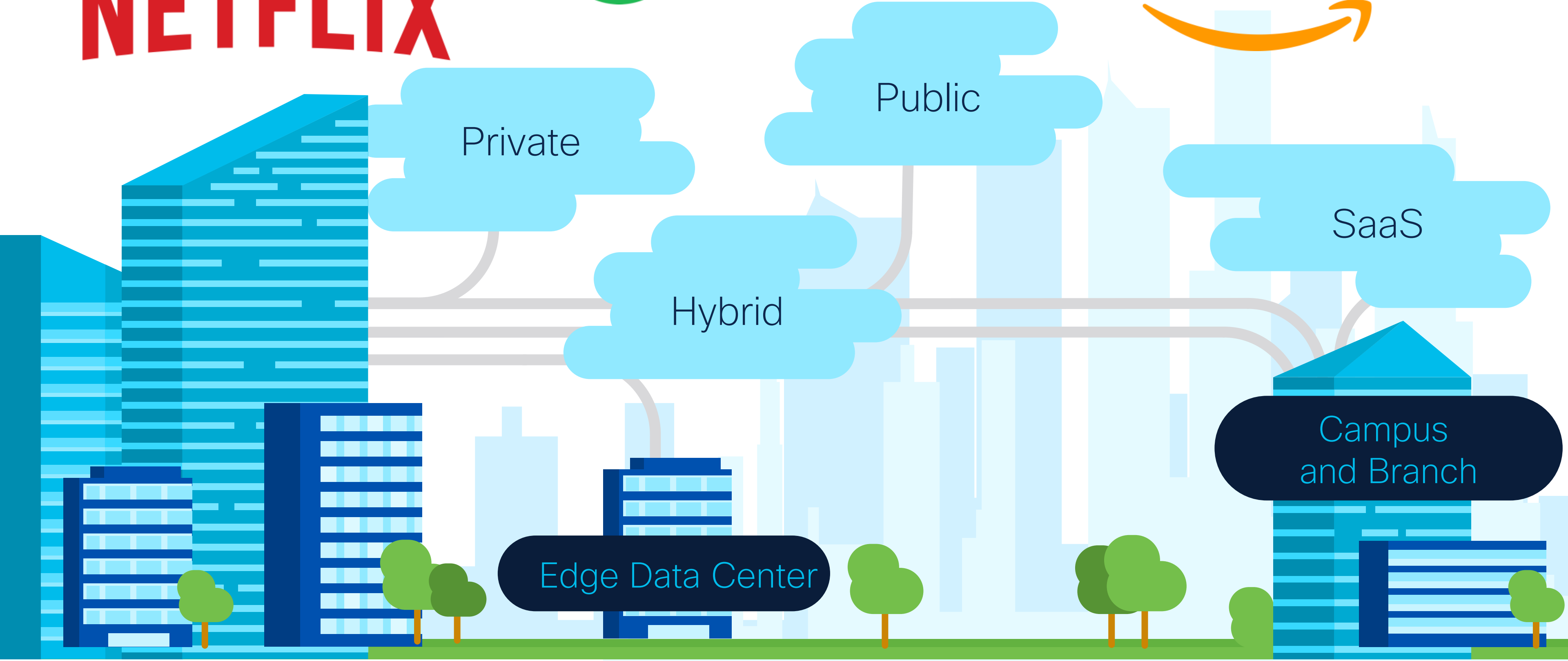
linkedin.com/in/joshingeniero

There has been an explosive growth for interest in cloud.

NETFLIX

Spotify®

amazon



Private

Public

Hybrid

SaaS

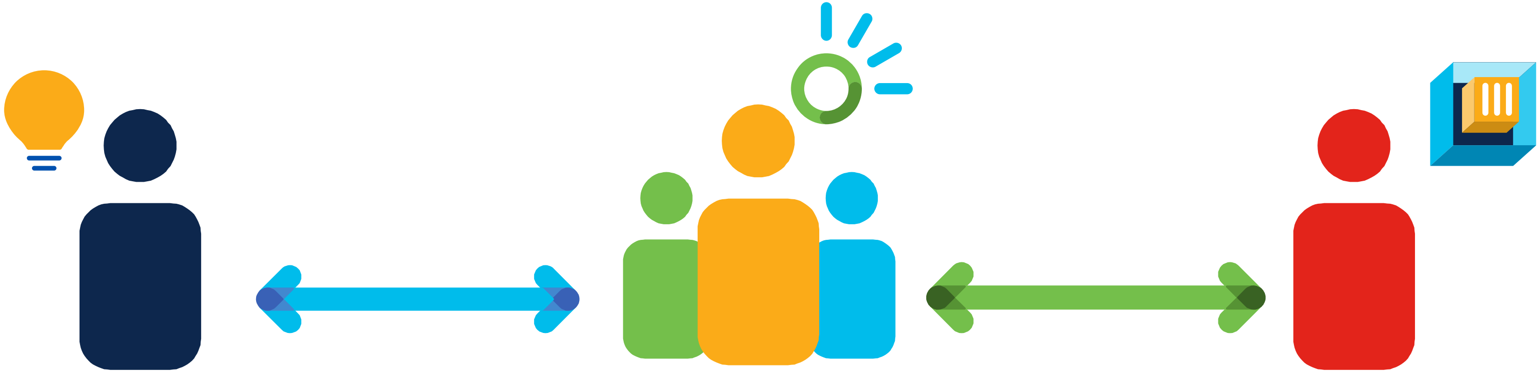
Edge Data Center

Campus and Branch

Agenda

- Introduction
- Software Stakeholders
- Modern Software Development
- Cloud-native Applications
- Demo

Stakeholders

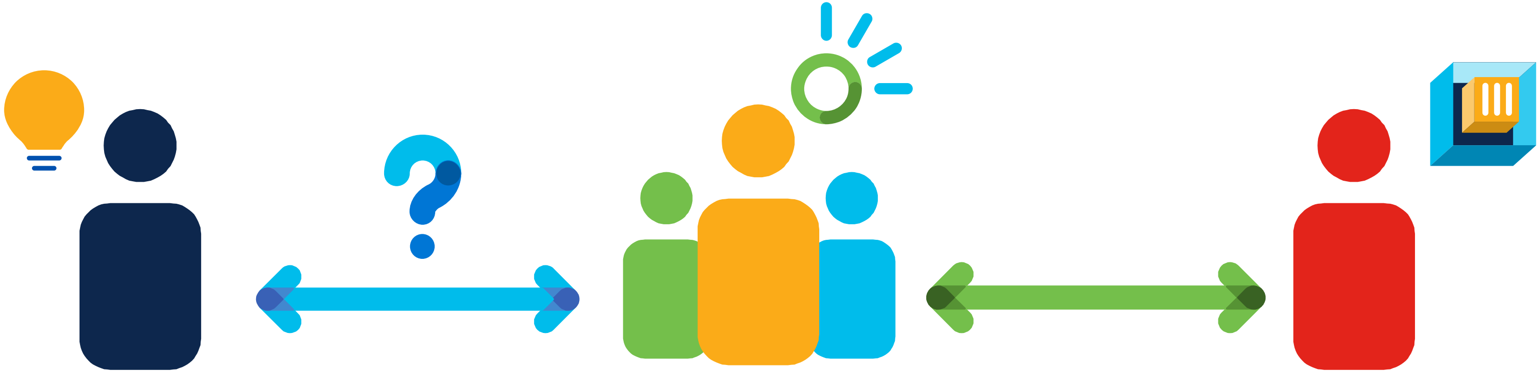


Customer

Developers

Operations

Stakeholders

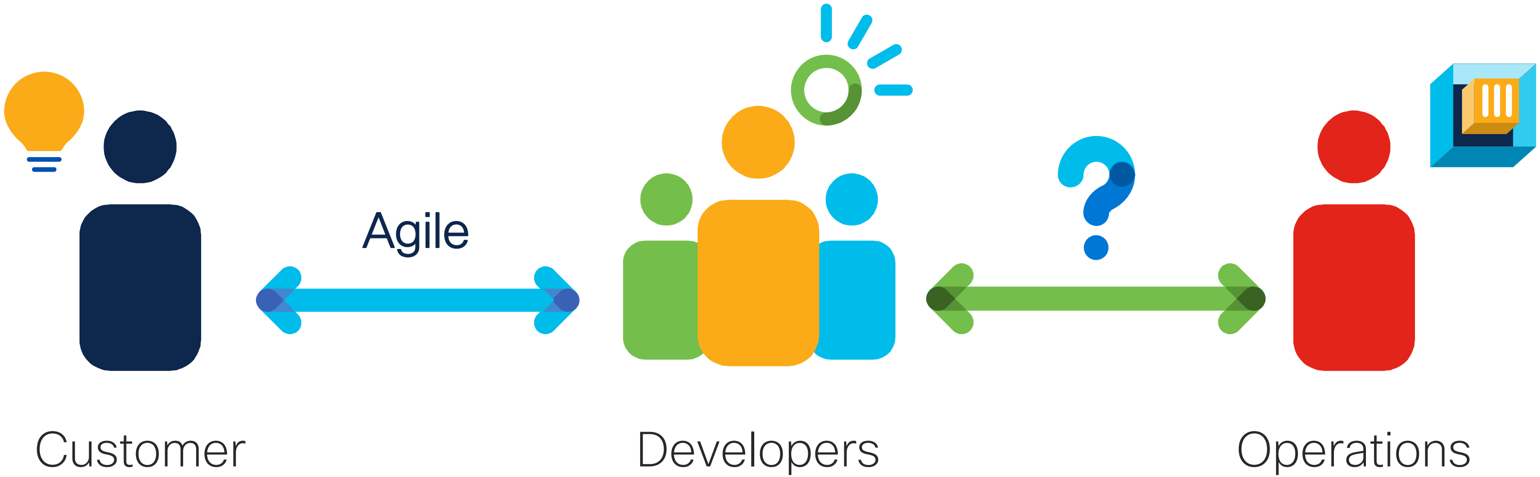


Customer

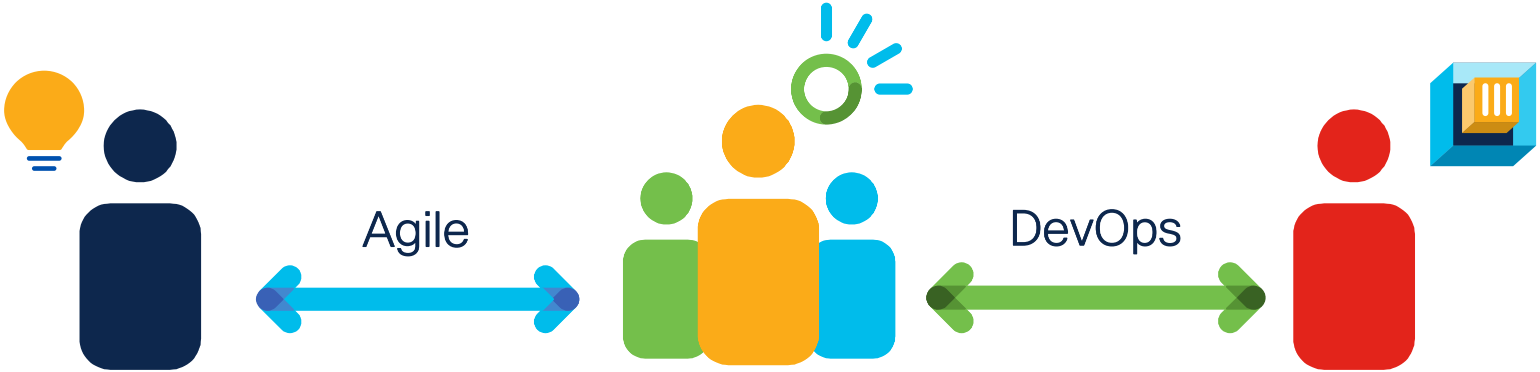
Developers

Operations

Stakeholders



Stakeholders



Agile

DevOps

Customer

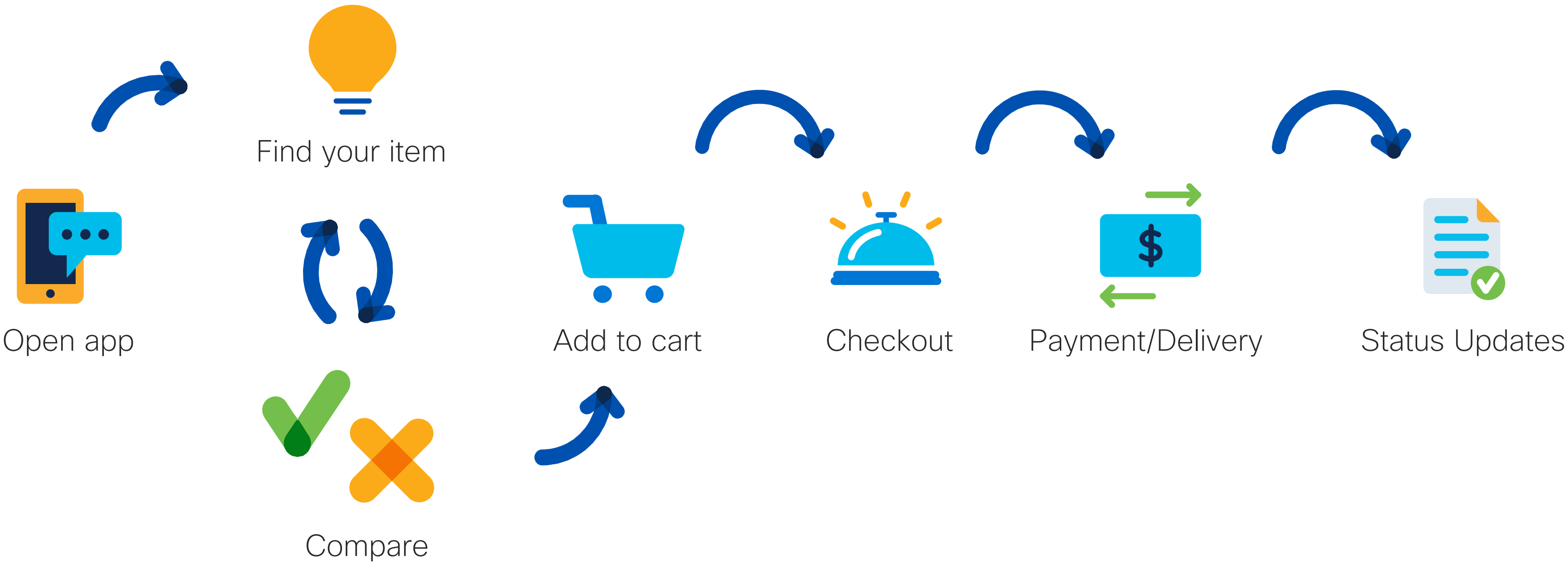
Developers

Operations

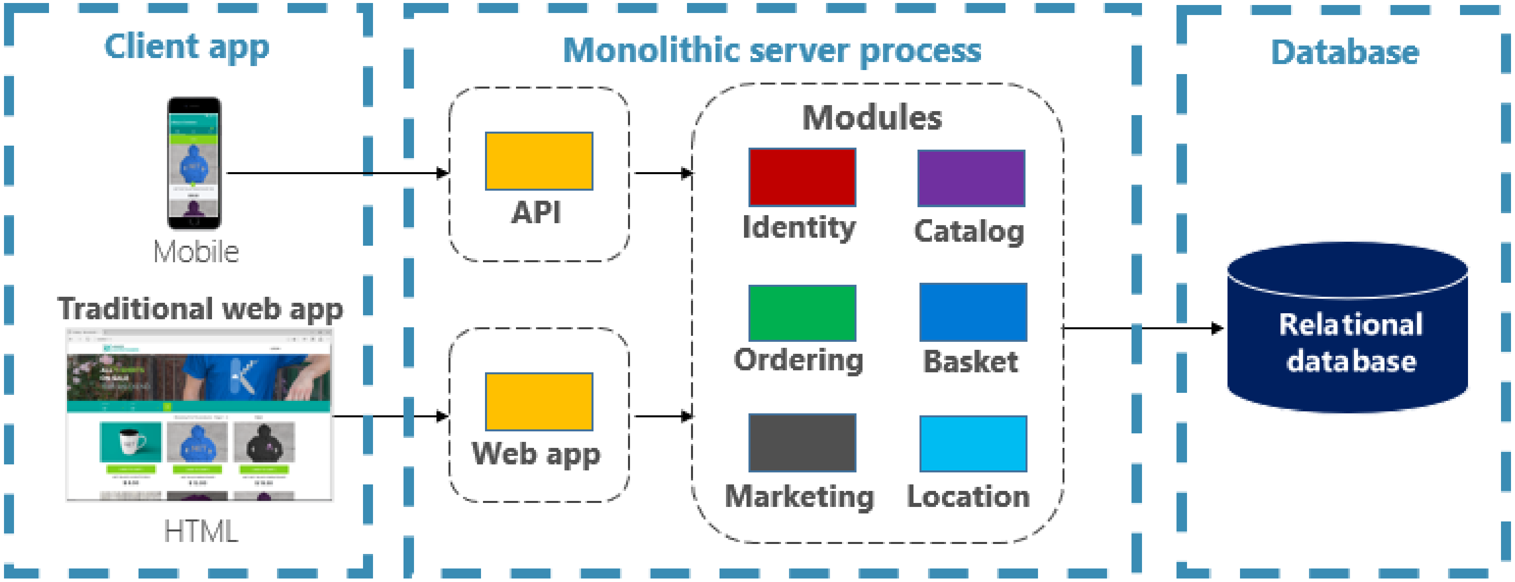
Modern Software Development

Modern Software Development

Customer Journey



Traditional Software Development



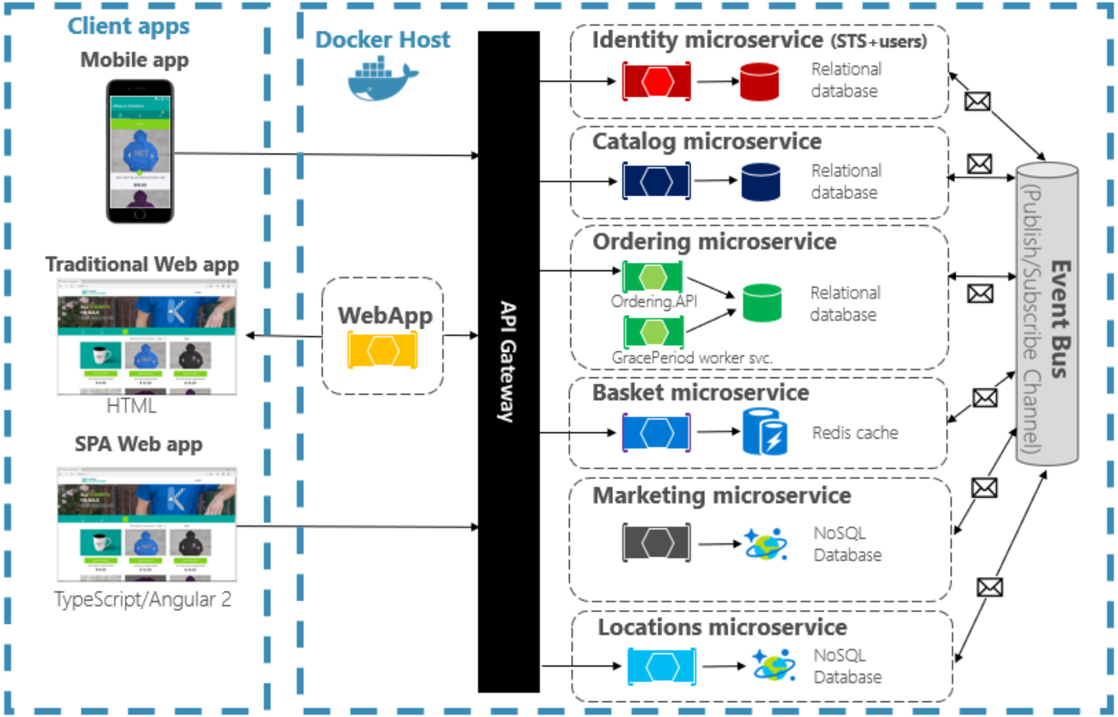
Monolithic Software Design

Modern Software Development

NETFLIX

amazon

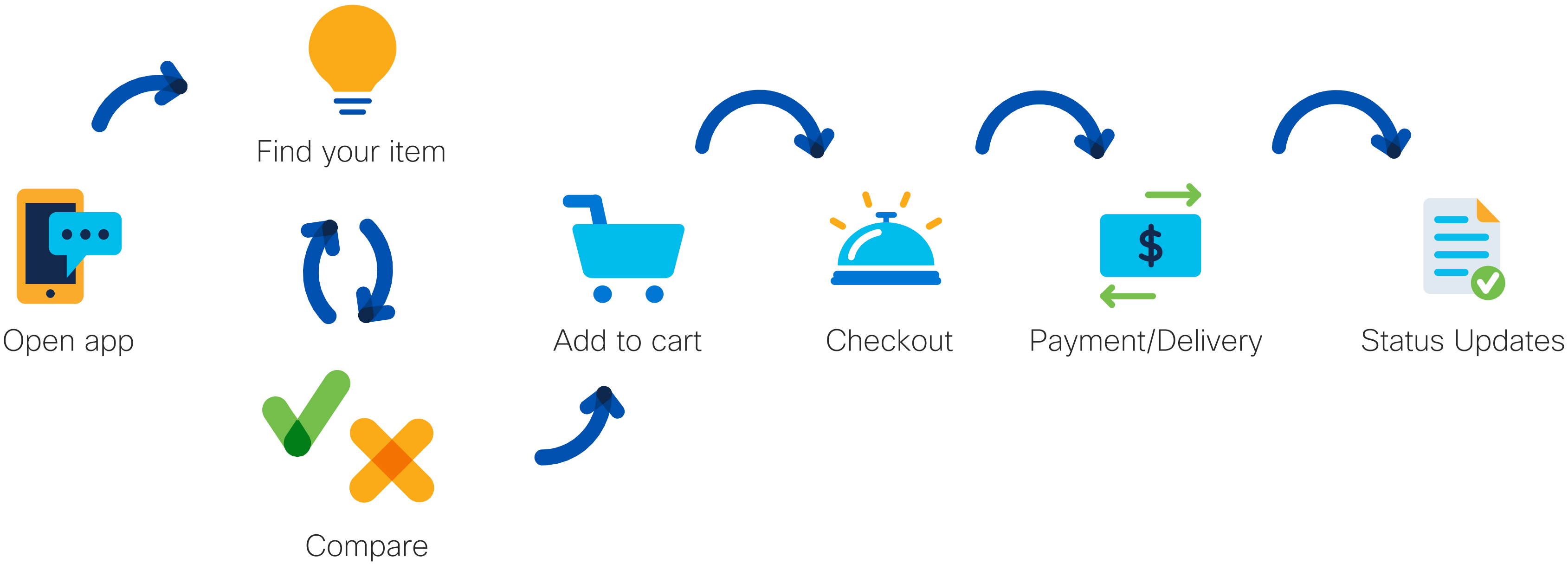
Spotify



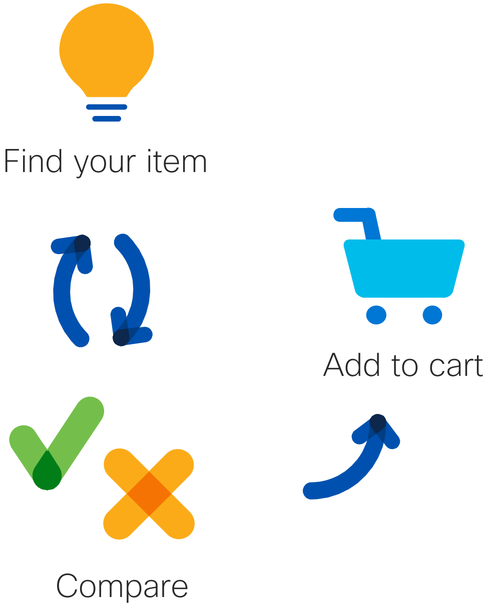
Cloud-native (Microservices) Software Design

Cloud-native Applications

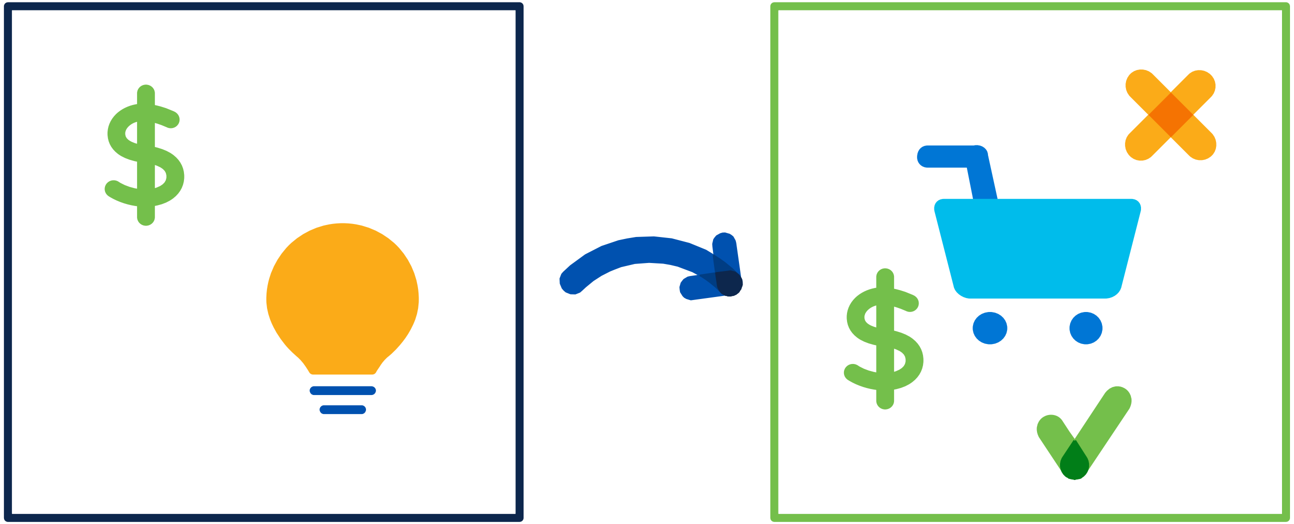
Microservices



Microservices



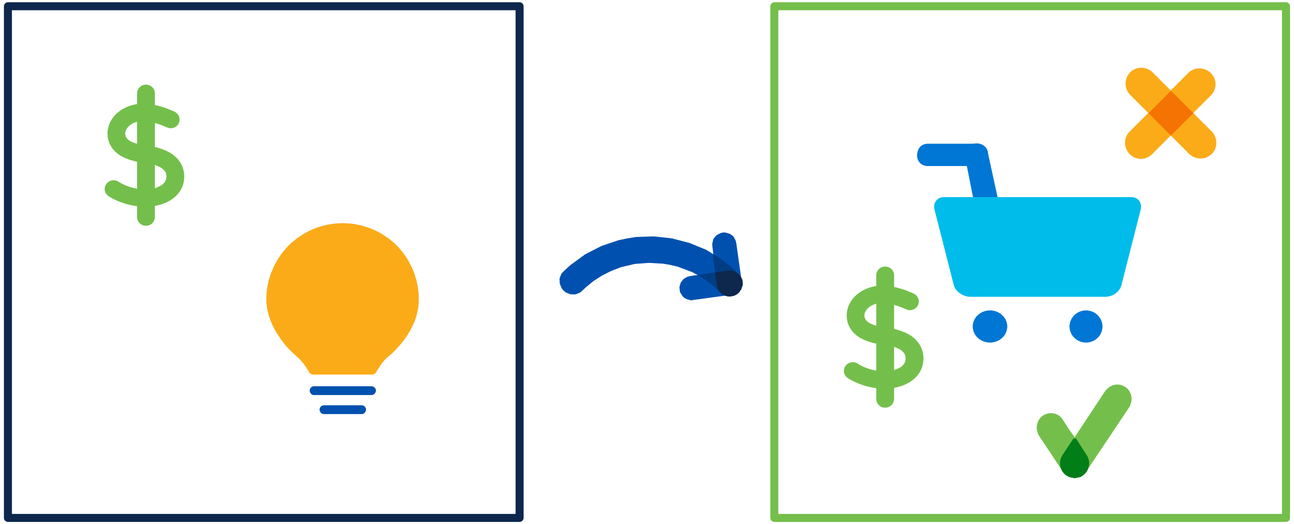
Microservices



Catalog Microservice

Cart Microservice

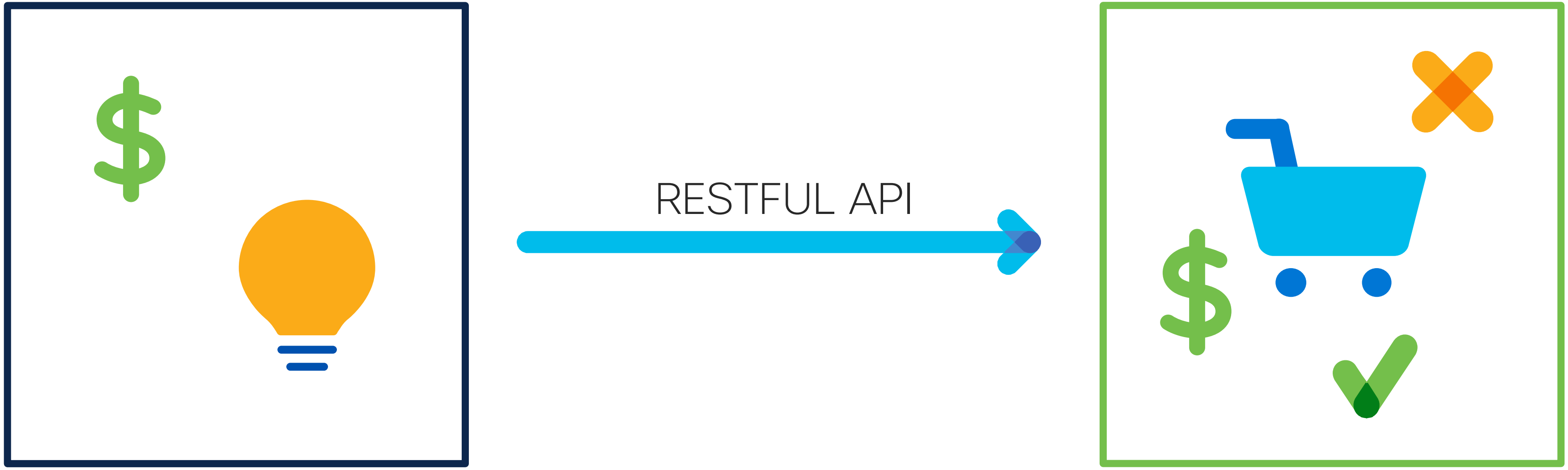
API’s and Communication



Catalog Microservice

Cart Microservice

API’s and Communication

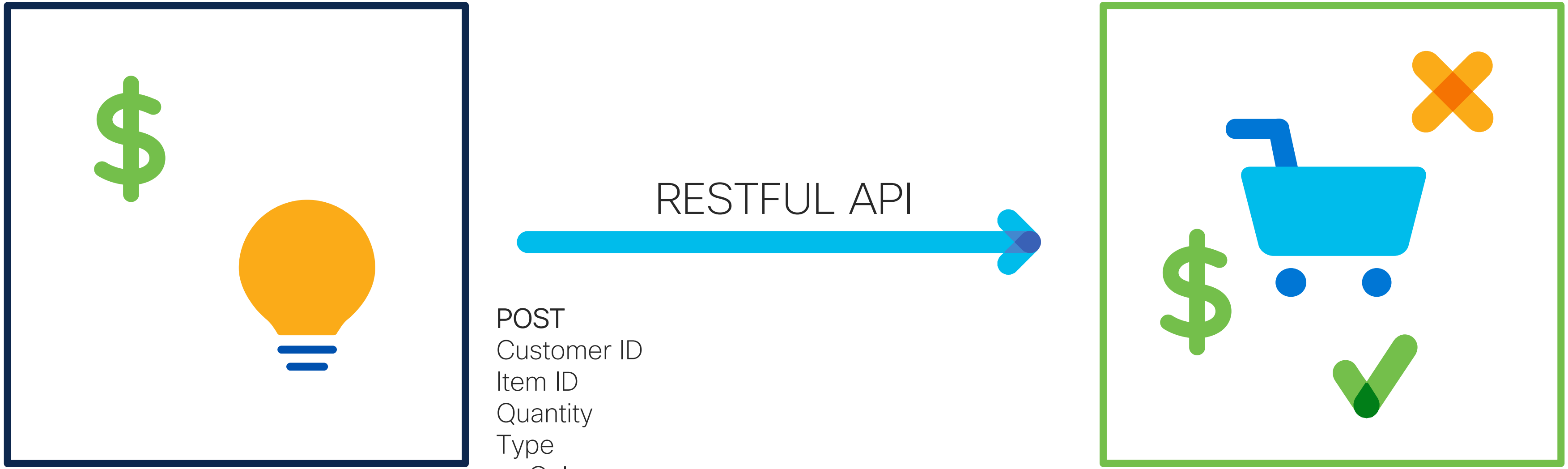


RESTFUL API

Catalog Microservice

Cart Microservice

API's and Communication



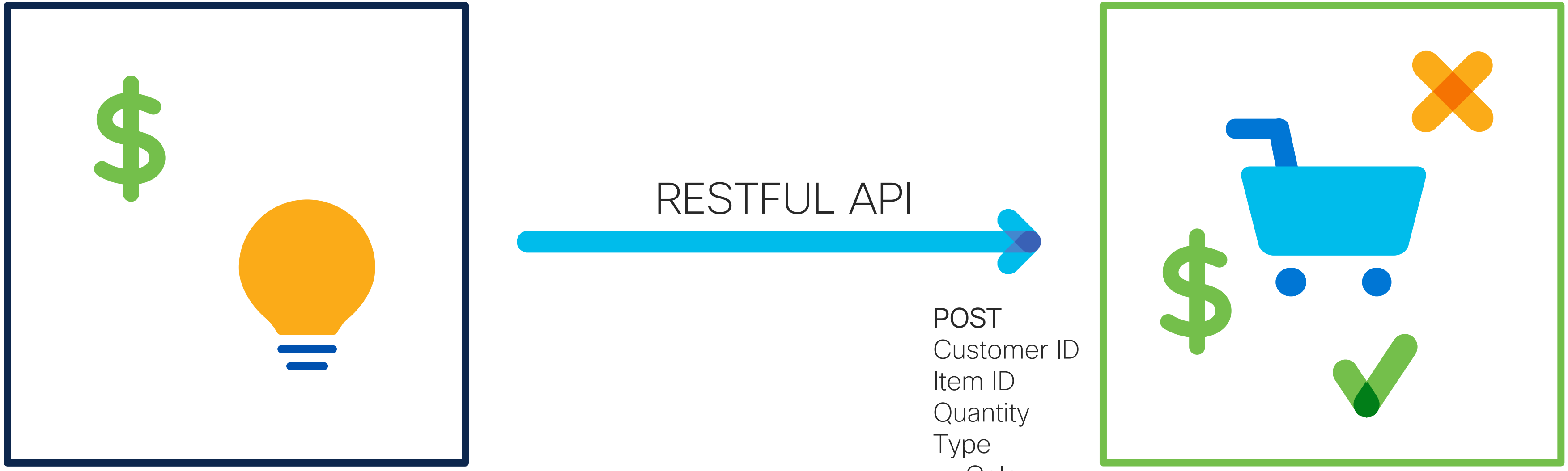
RESTFUL API

POST
Customer ID
Item ID
Quantity
Type
- Colour
- Size

Catalog Microservice

Cart Microservice

API’s and Communication



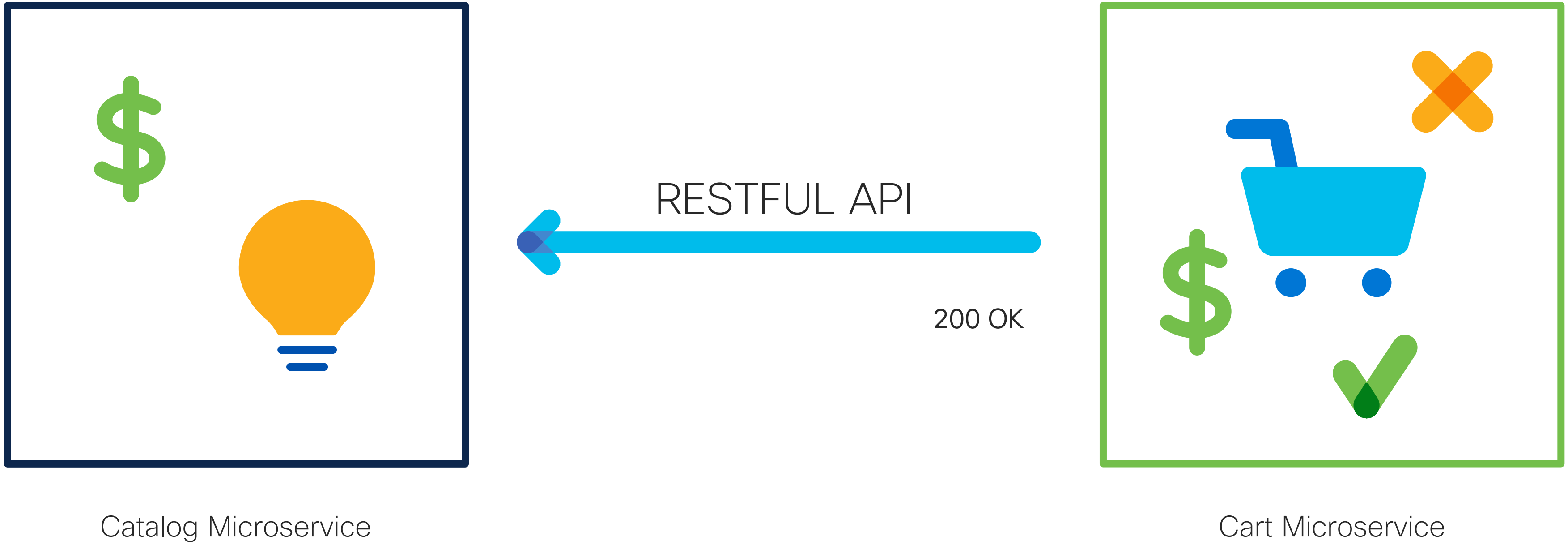
RESTFUL API

POST
Customer ID
Item ID
Quantity
Type
- Colour
- Size

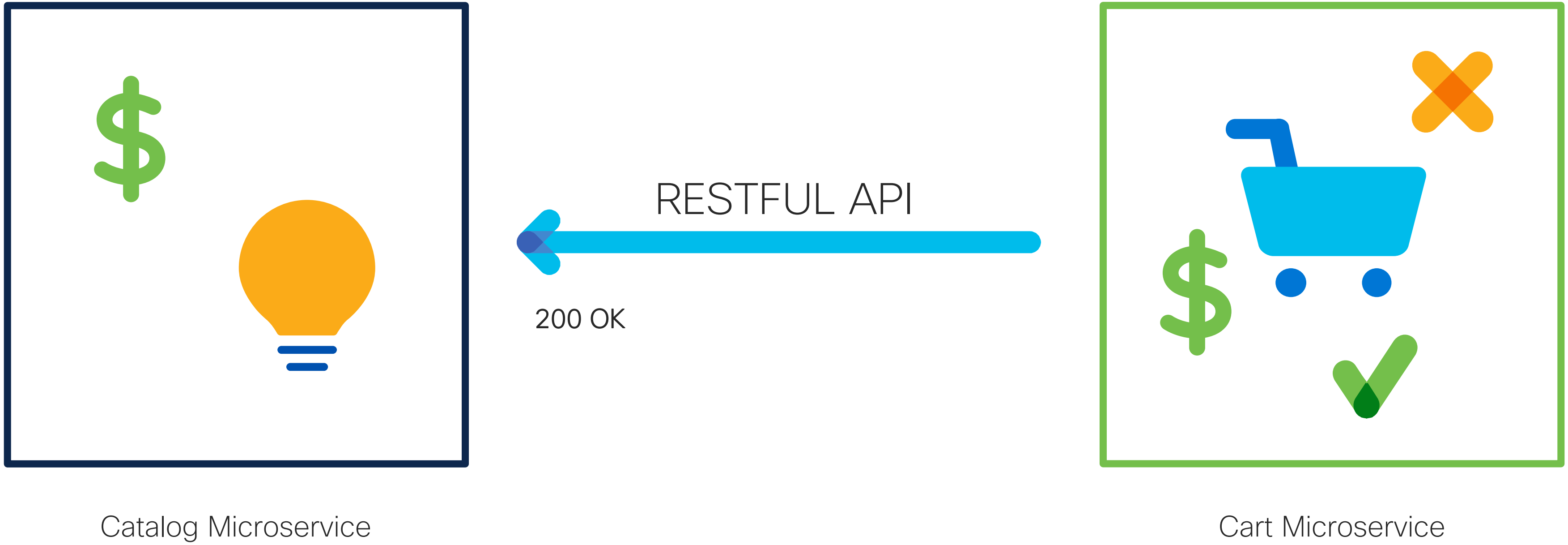
Catalog Microservice

Cart Microservice

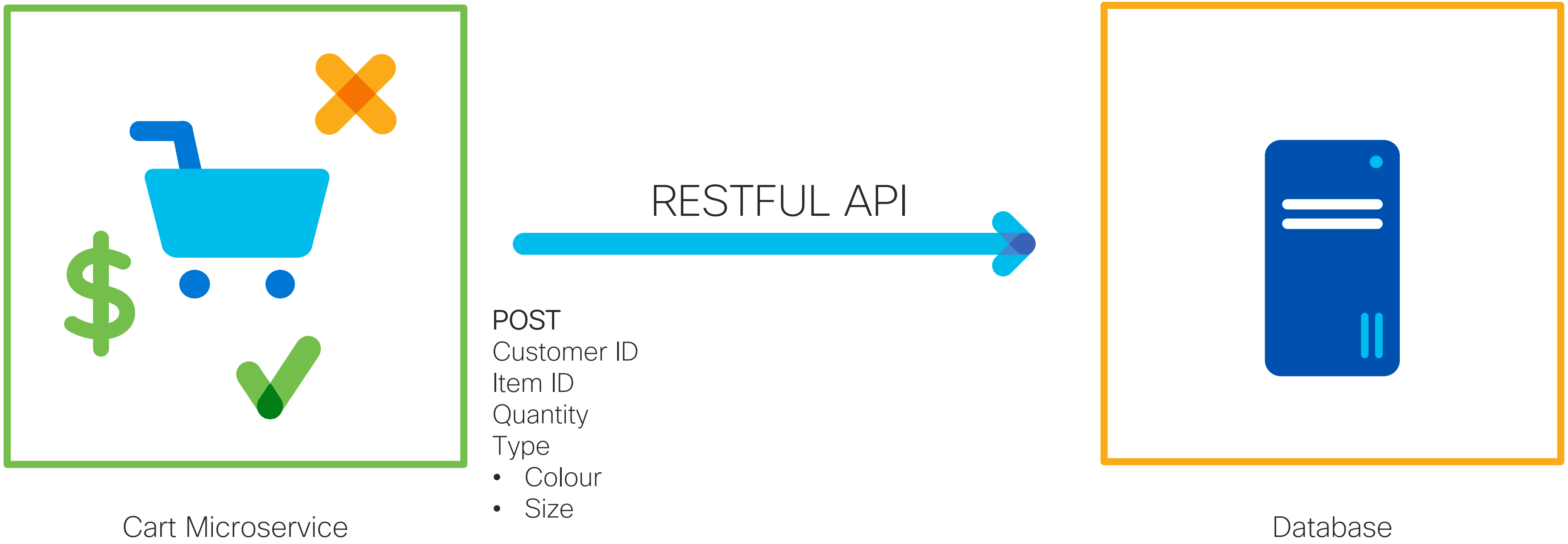
API’s and Communication



API’s and Communication



API's and Communication



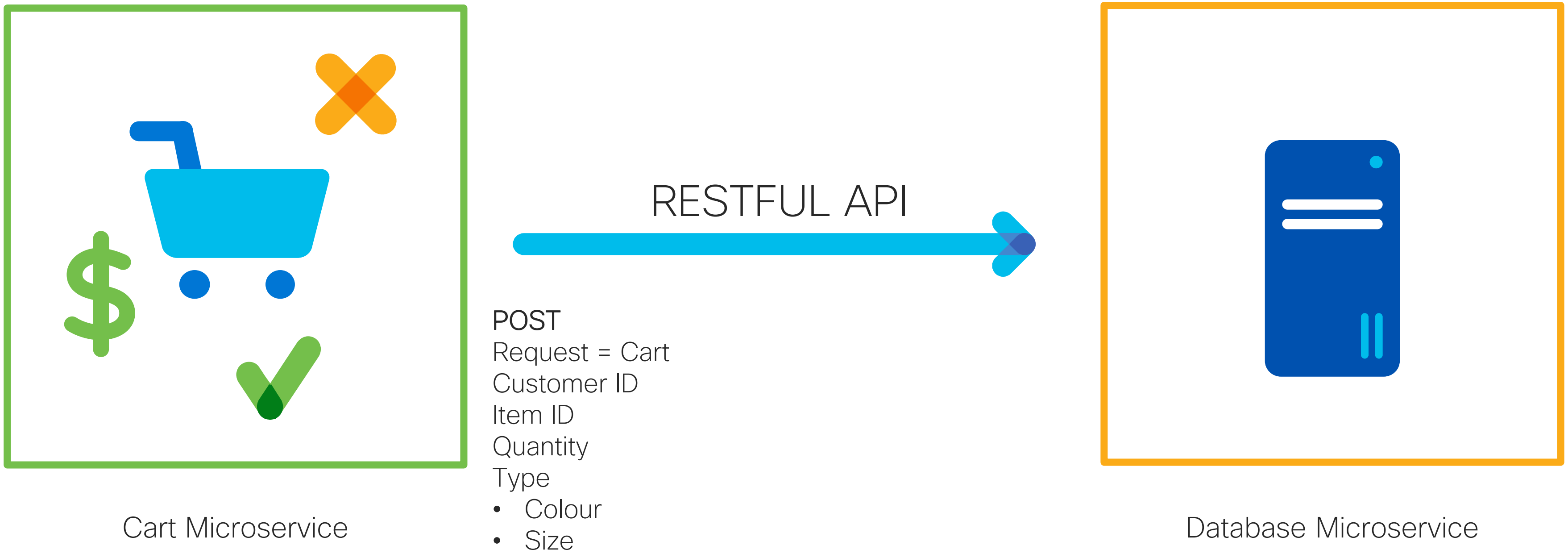
RESTFUL API

POST
Customer ID
Item ID
Quantity
Type
- Colour
- Size

Cart Microservice

Database

API's and Communication



RESTFUL API

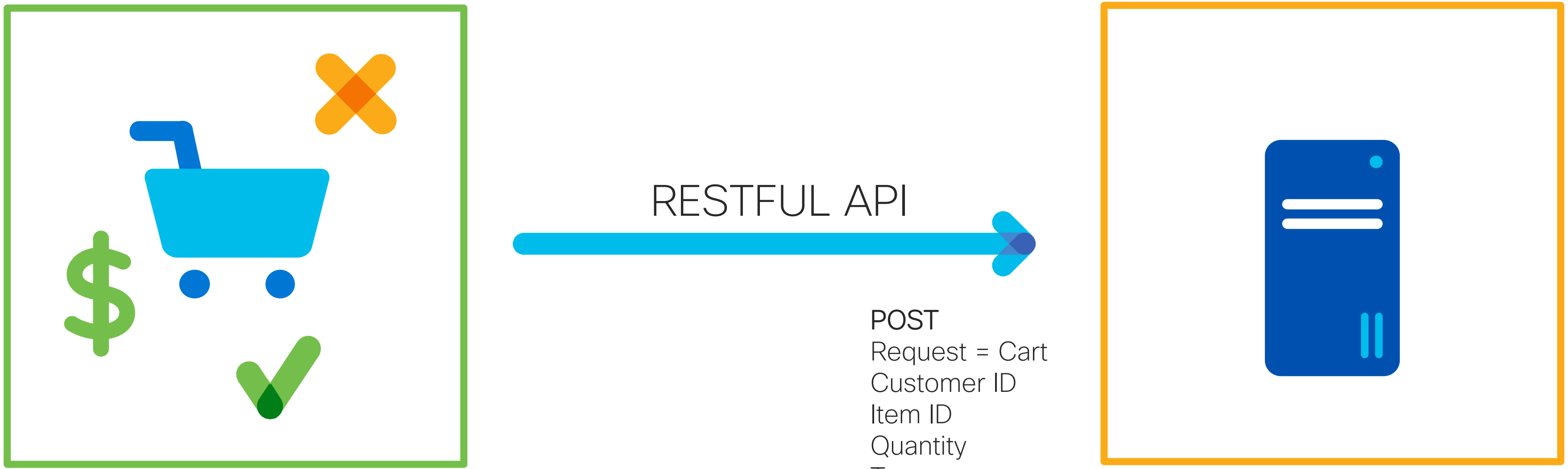
POST
Request = Cart
Customer ID
Item ID
Quantity
Type

- Colour
- Size

Cart Microservice

Database Microservice

API's and Communication

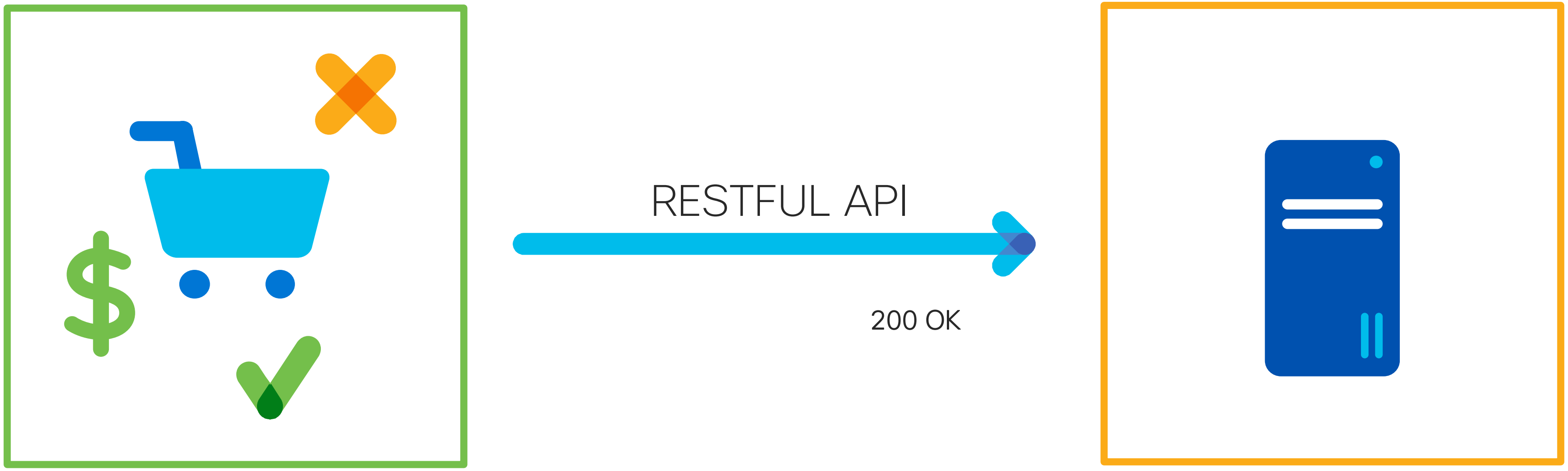


POST
Request = Cart
Customer ID
Item ID
Quantity
Type
- Colour
- Size

Cart Microservice

Database Microservice

API’s and Communication



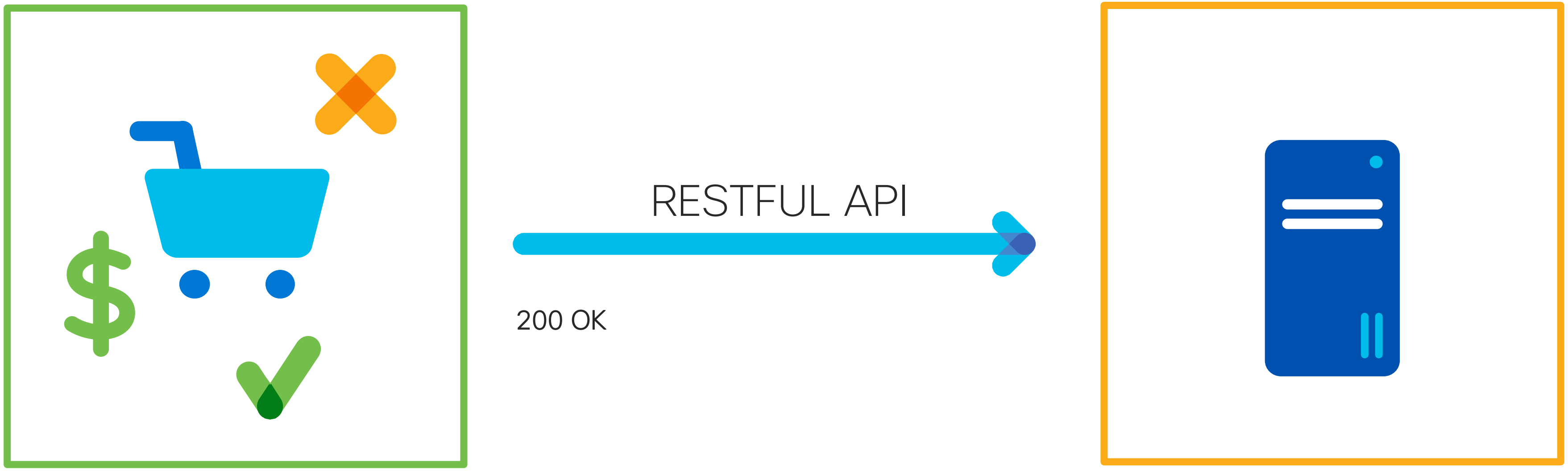
RESTFUL API

200 OK

Cart Microservice

Database Microservice

API's and Communication

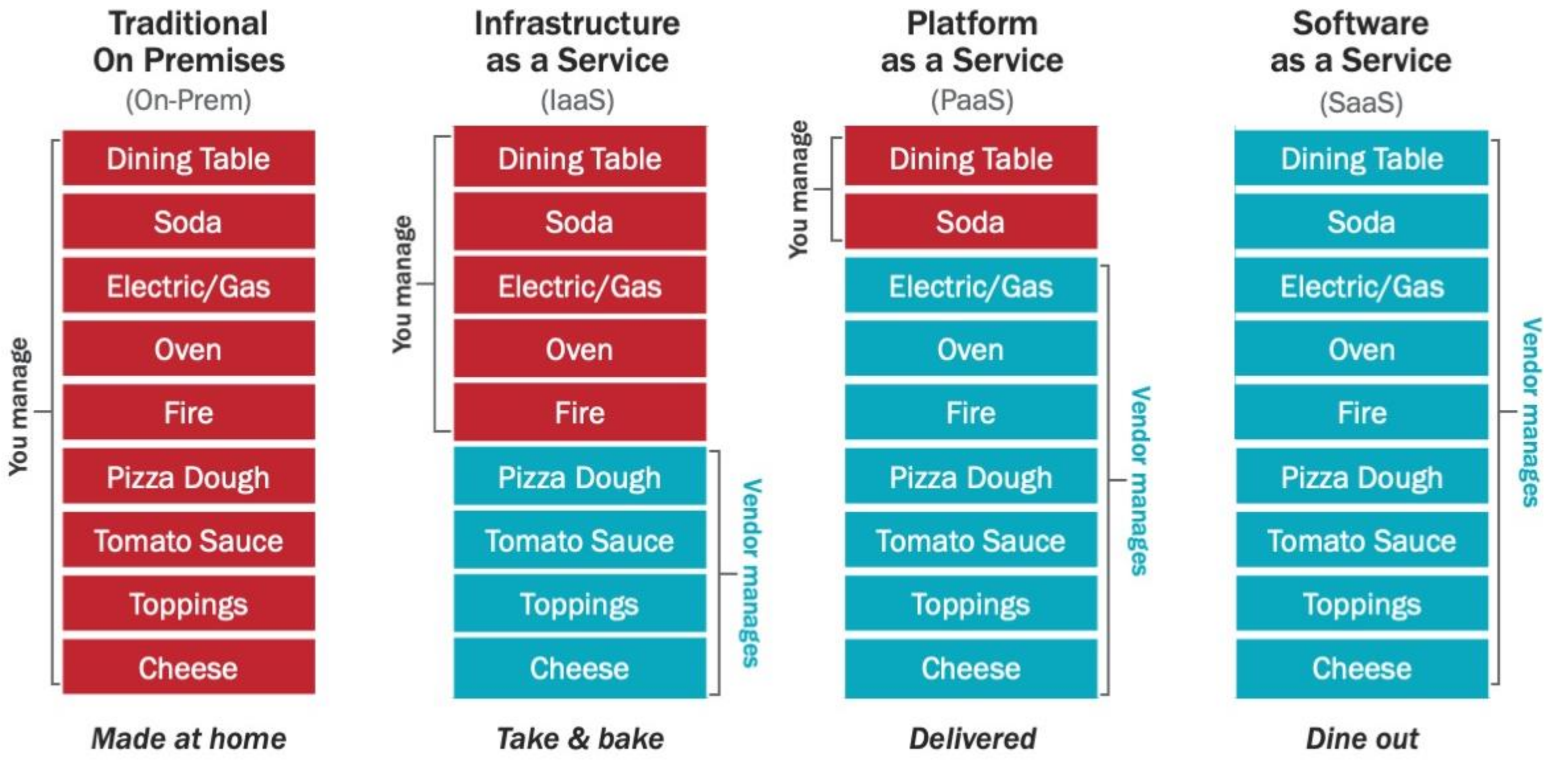


Cart Microservice

Database Microservice

Deploying Applications

Pizza as a Service



Traditional On Premises (On-Prem)

You manage:
- Dining Table
- Soda
- Electric/Gas
- Oven
- Fire
- Pizza Dough
- Tomato Sauce
- Toppings
- Cheese

Made at home

Infrastructure as a Service (IaaS)

You manage:
- Dining Table
- Soda
- Electric/Gas
- Oven
- Fire

Vendor manages:
- Pizza Dough
- Tomato Sauce
- Toppings
- Cheese

Take & bake

Platform as a Service (PaaS)

You manage:
- Dining Table
- Soda

Vendor manages:
- Electric/Gas
- Oven
- Fire
- Pizza Dough
- Tomato Sauce
- Toppings
- Cheese

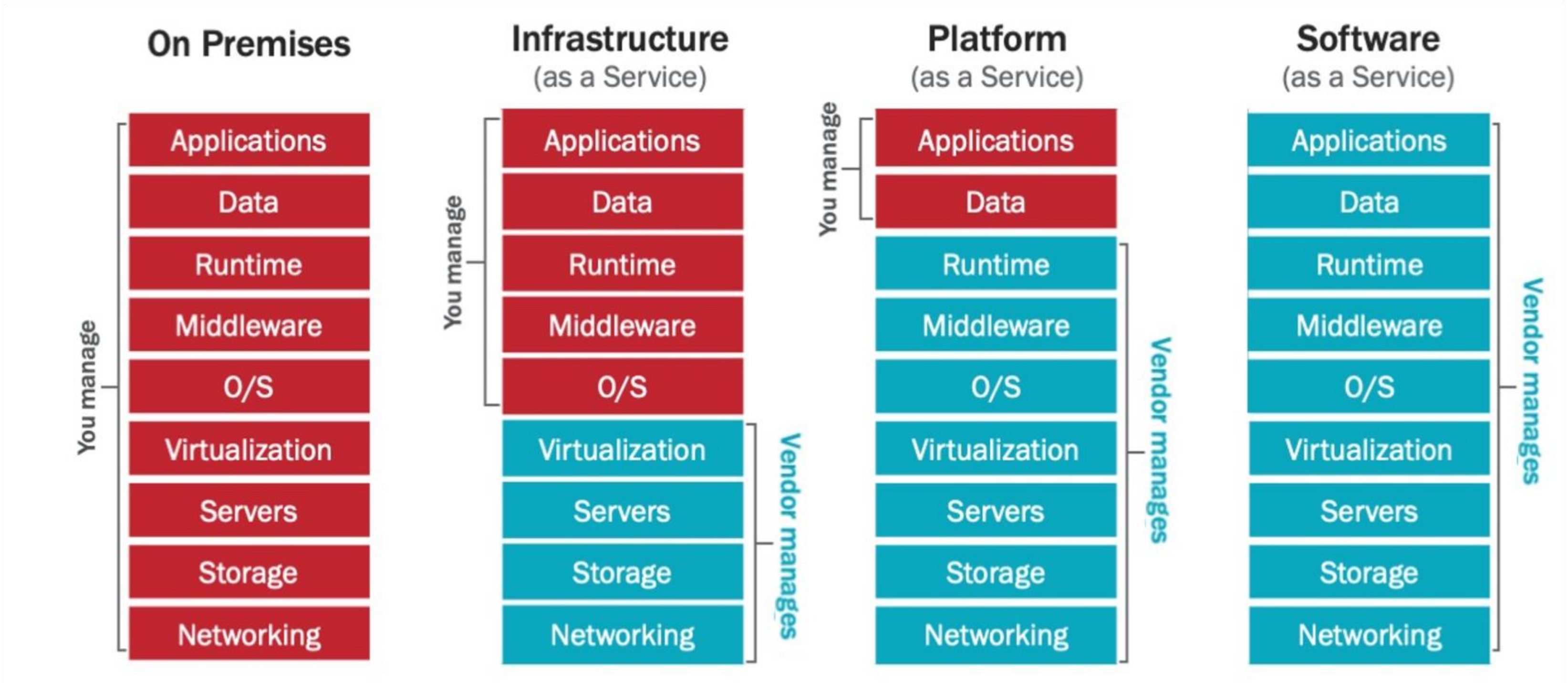
Delivered

Software as a Service (SaaS)

Vendor manages:
- Dining Table
- Soda
- Electric/Gas
- Oven
- Fire
- Pizza Dough
- Tomato Sauce
- Toppings
- Cheese

Dine out

Deployment Models



On Premises

You manage

Applications
Data
Runtime
Middleware
O/S
Virtualization
Servers
Storage
Networking

Infrastructure
(as a Service)

You manage

Applications
Data
Runtime
Middleware
O/S

Vendor manages

Virtualization
Servers
Storage
Networking

Platform
(as a Service)

You manage

Applications
Data

Vendor manages

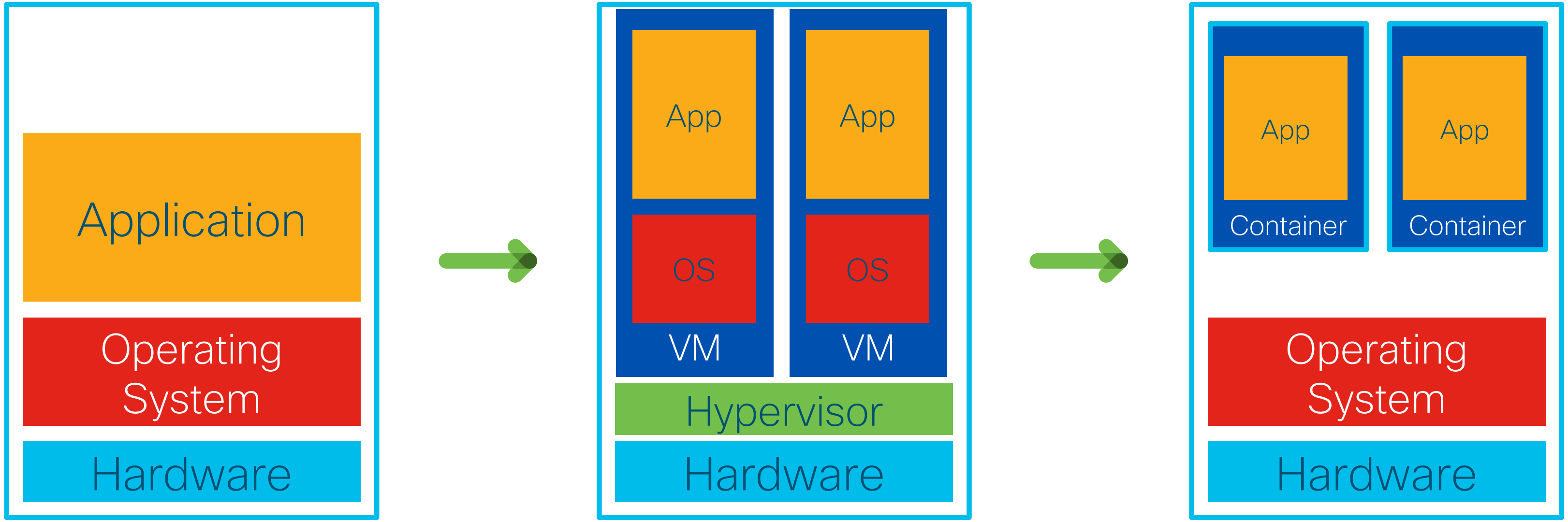
Runtime
Middleware
O/S
Virtualization
Servers
Storage
Networking

Software
(as a Service)

Vendor manages

Applications
Data
Runtime
Middleware
O/S
Virtualization
Servers
Storage
Networking

Deployment Types



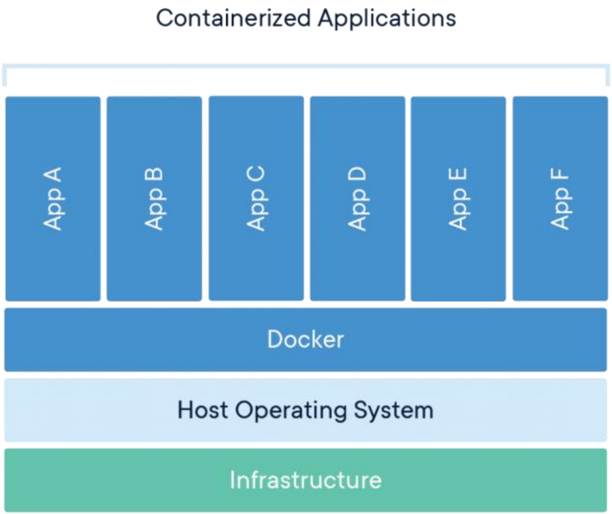
No virtualisation
(Bare Metal)

Virtual Machines
(VM)

Containers

Containers and Kubernetes

Containers

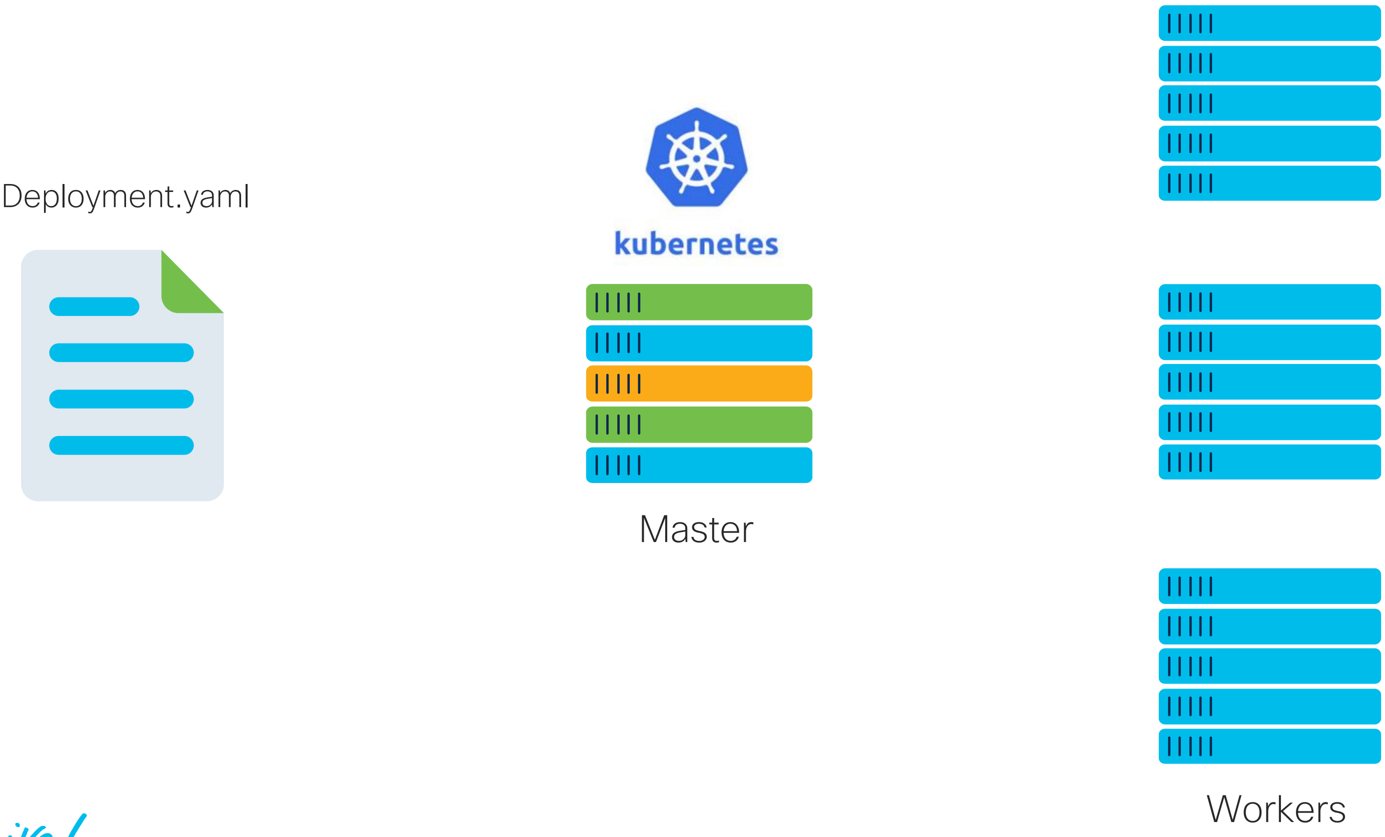


```
FROM python:3.8.2

RUN mkdir /app
WORKDIR /app
ADD . /app/
RUN pip install -r /app/requirements.txt

EXPOSE 5000
CMD ["python", "/app/app.py"]
```

Containers

- Agile
- CI/CD
- DevOps
- Observability
- Consistency
- Portability
- Application-Centric Management
- Microservices
- Isolation
- Utilisation

Kubernetes



Deployment.yaml

Master

Workers

Kubernetes

Services

- Exposing containers
- DNS name or IP

Storage

- Automatically mount a storage system of your choice

Rollout and Rollback

- Desired state
- Create, remove, or adopt resources

Bin packing

- CPU
- Memory
- Maximise resources

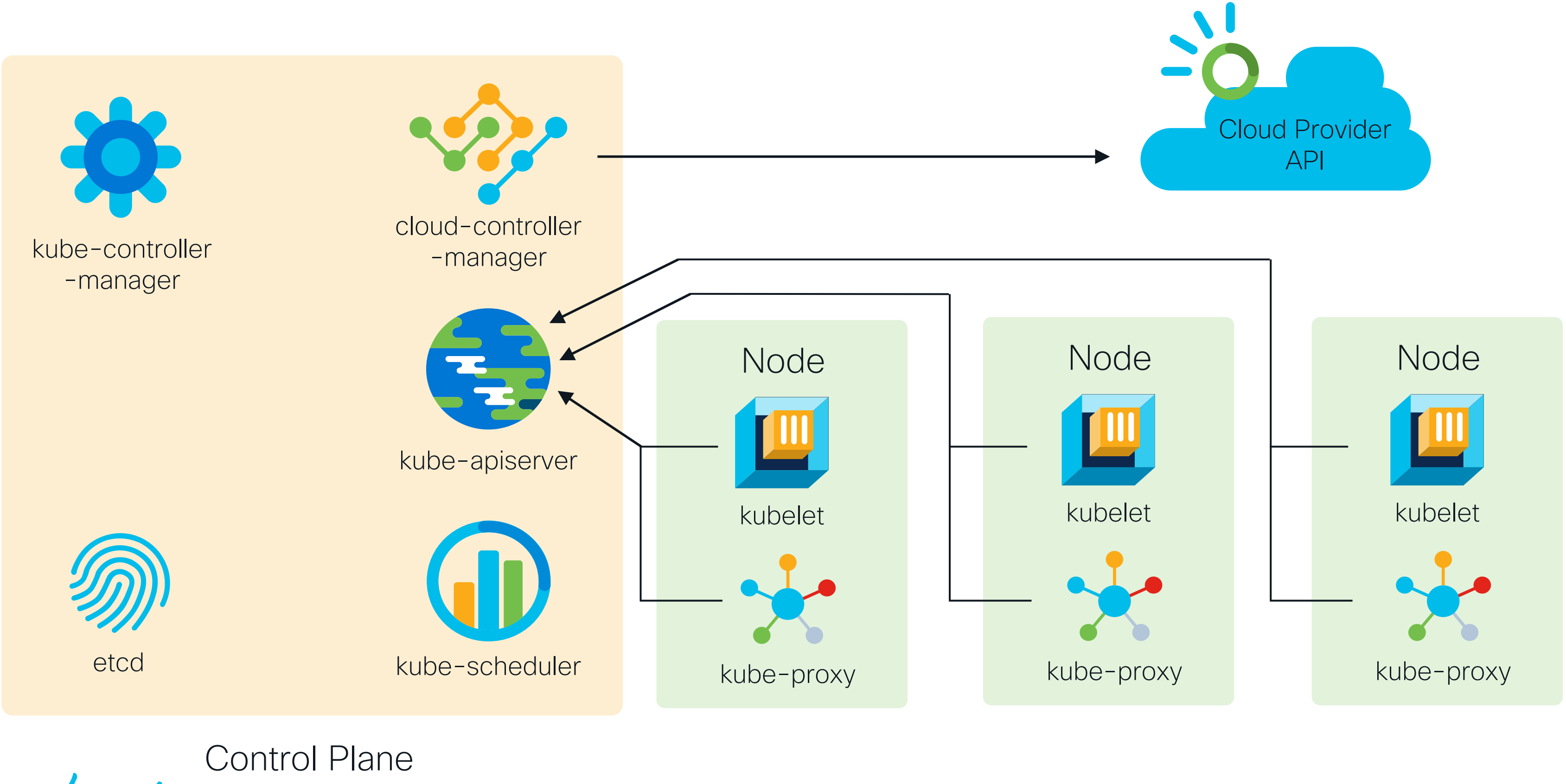
Self-healing

- Restarting failed containers
- Kill non-compliant containers

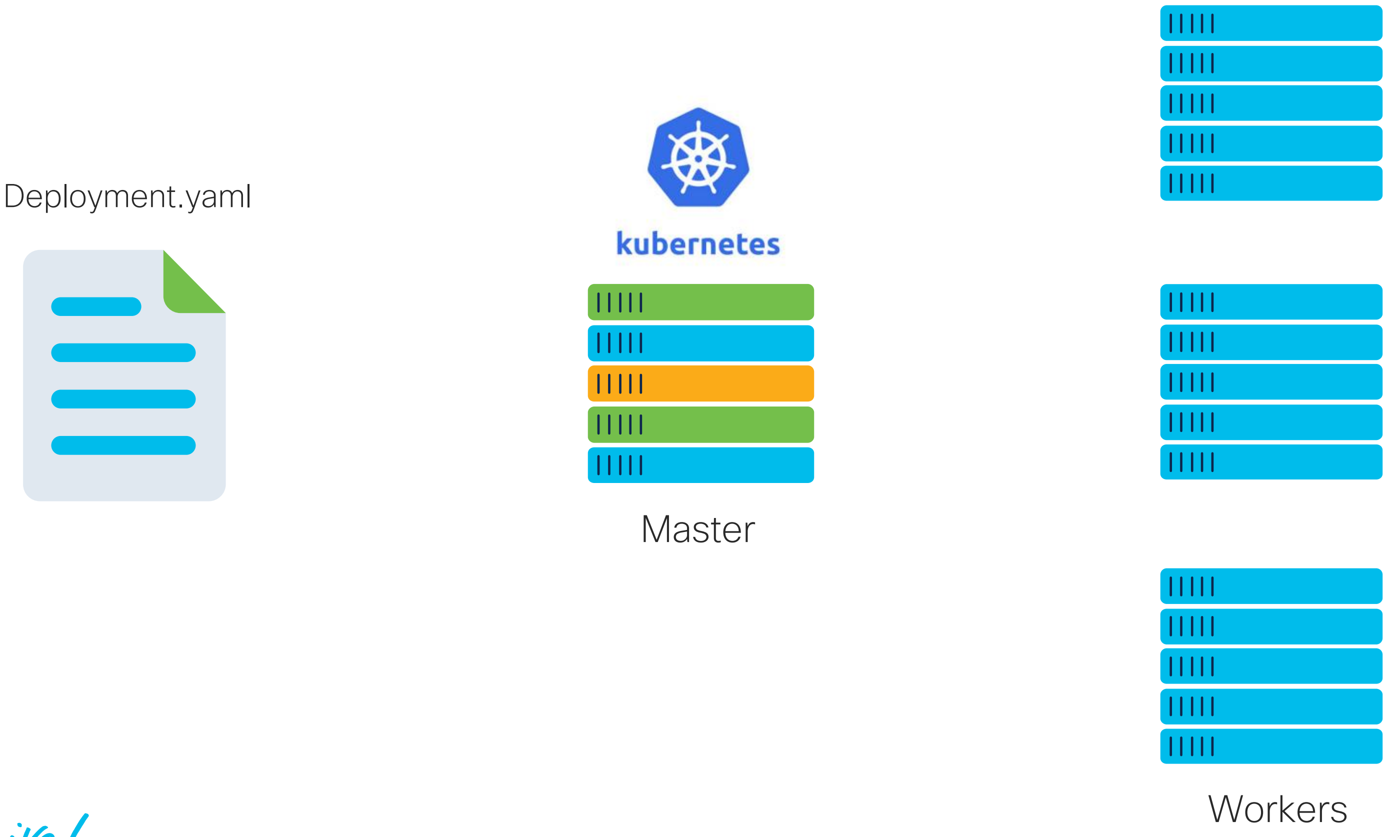
Secrets

- Store and manage sensitive info
- SSH, passwords, OAuth tokens

Kubernetes Cluster



Kubernetes

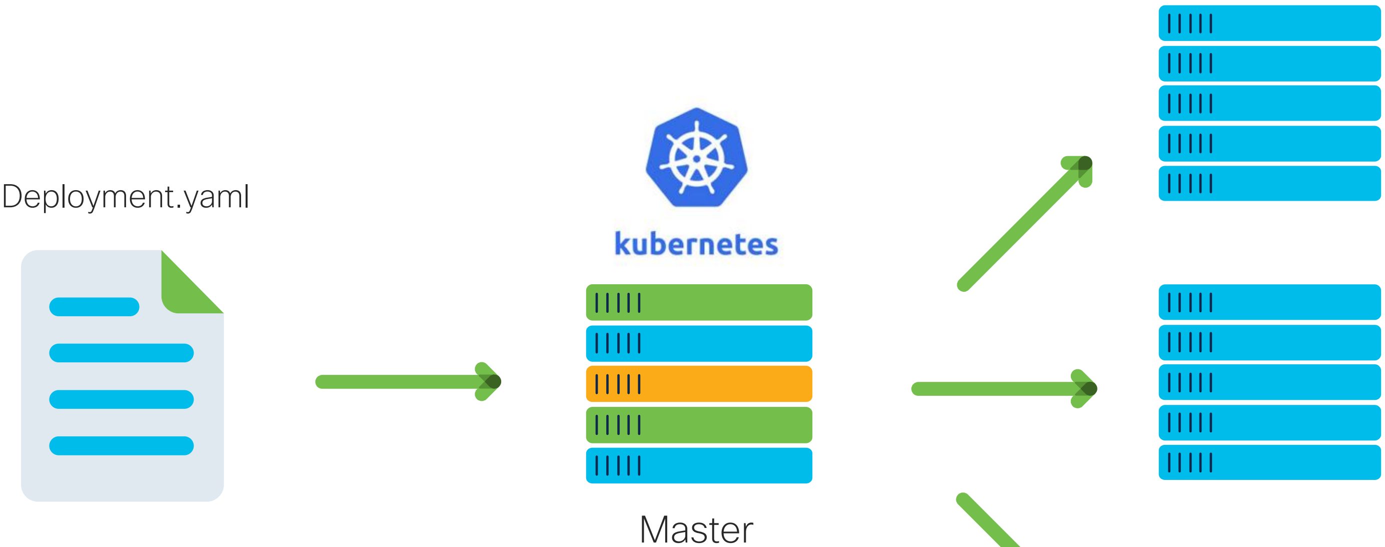


Deployment.yaml

Master

Workers

Kubernetes



Deployment.yaml

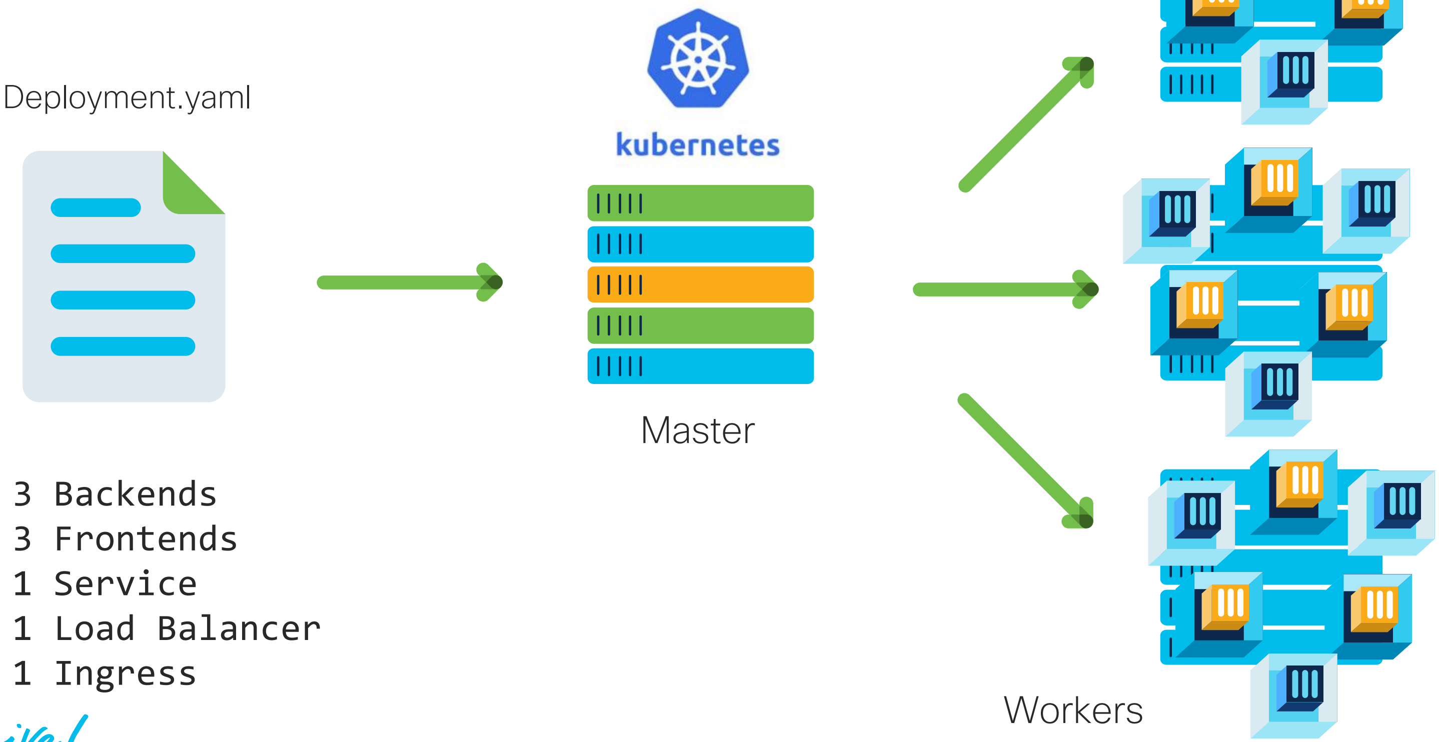
kubernetes

Master

```
3 Backends
3 Frontends
1 Service
1 Load Balancer
1 Ingress
```

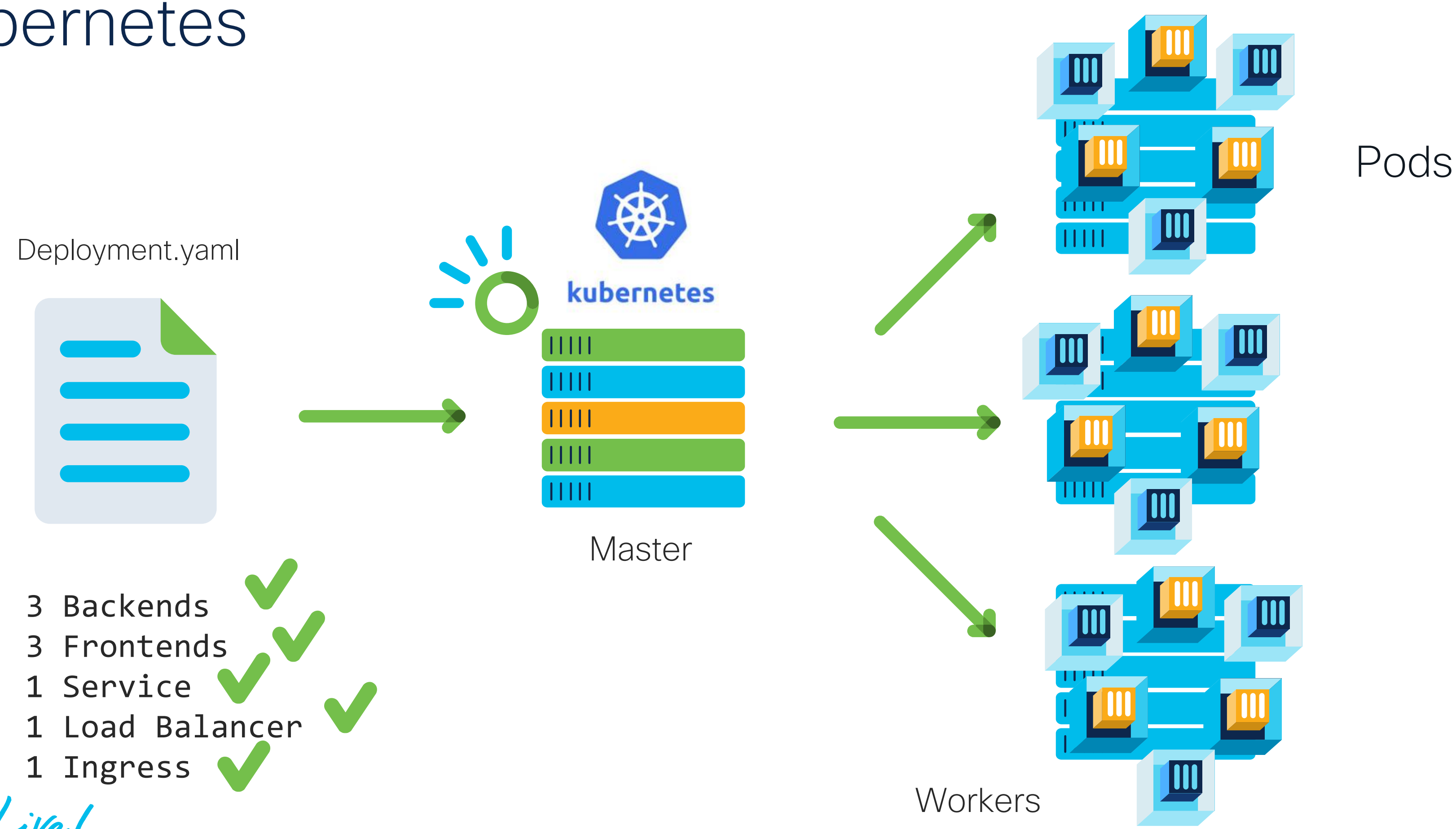
Workers

Kubernetes



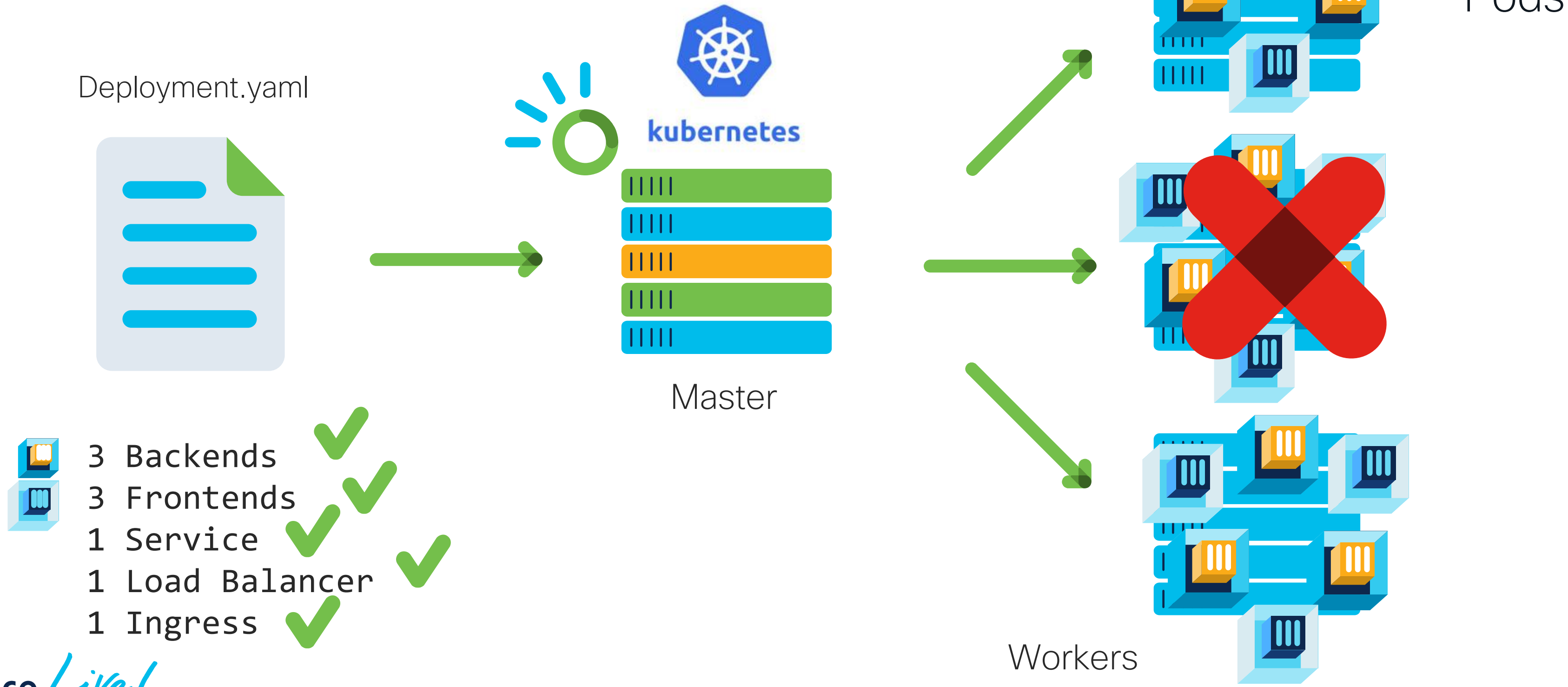
```
3 Backends
3 Frontends
1 Service
1 Load Balancer
1 Ingress
```

Kubernetes



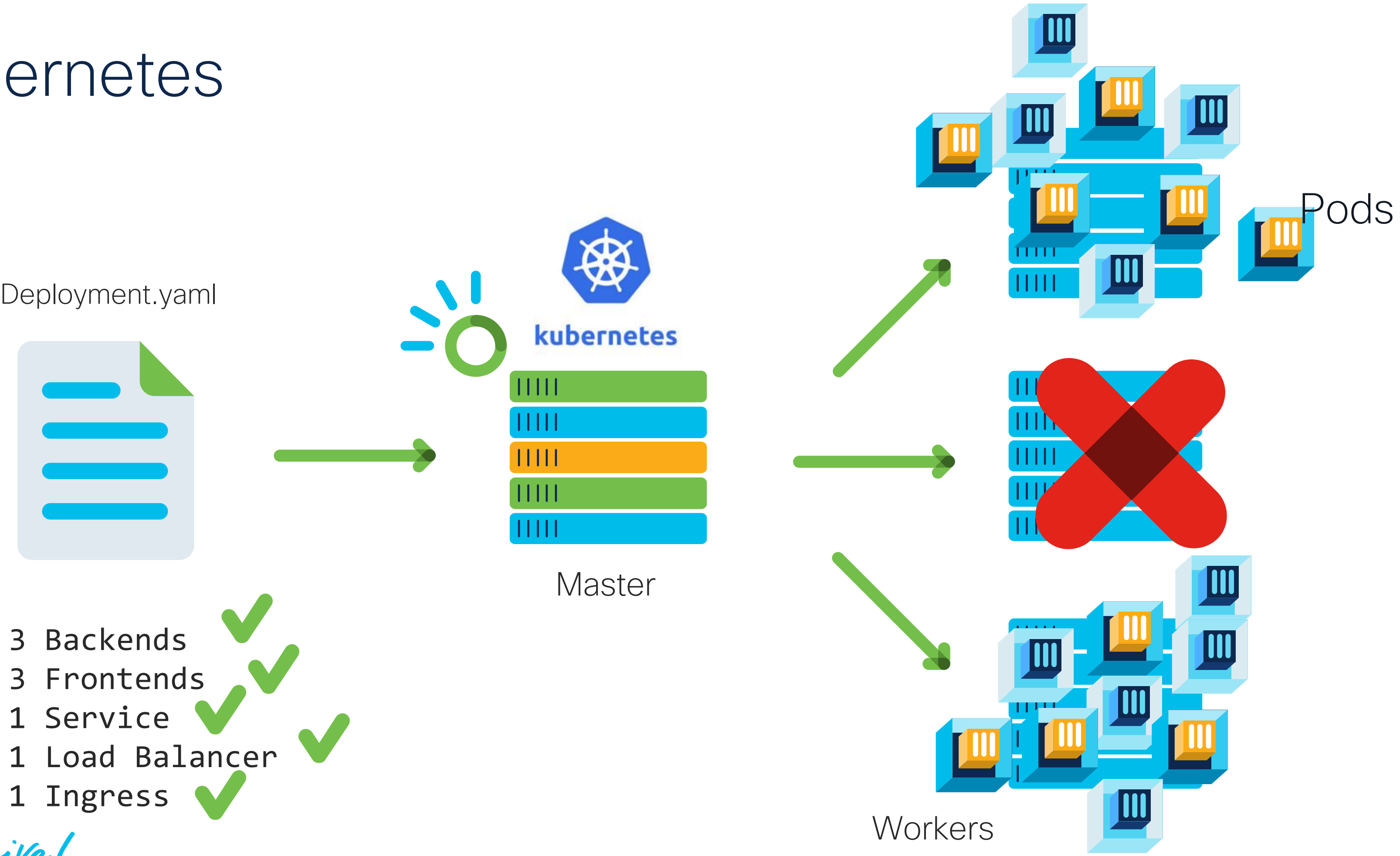
3 Backends ✓
3 Frontends ✓
1 Service ✓
1 Load Balancer ✓
1 Ingress ✓

Kubernetes



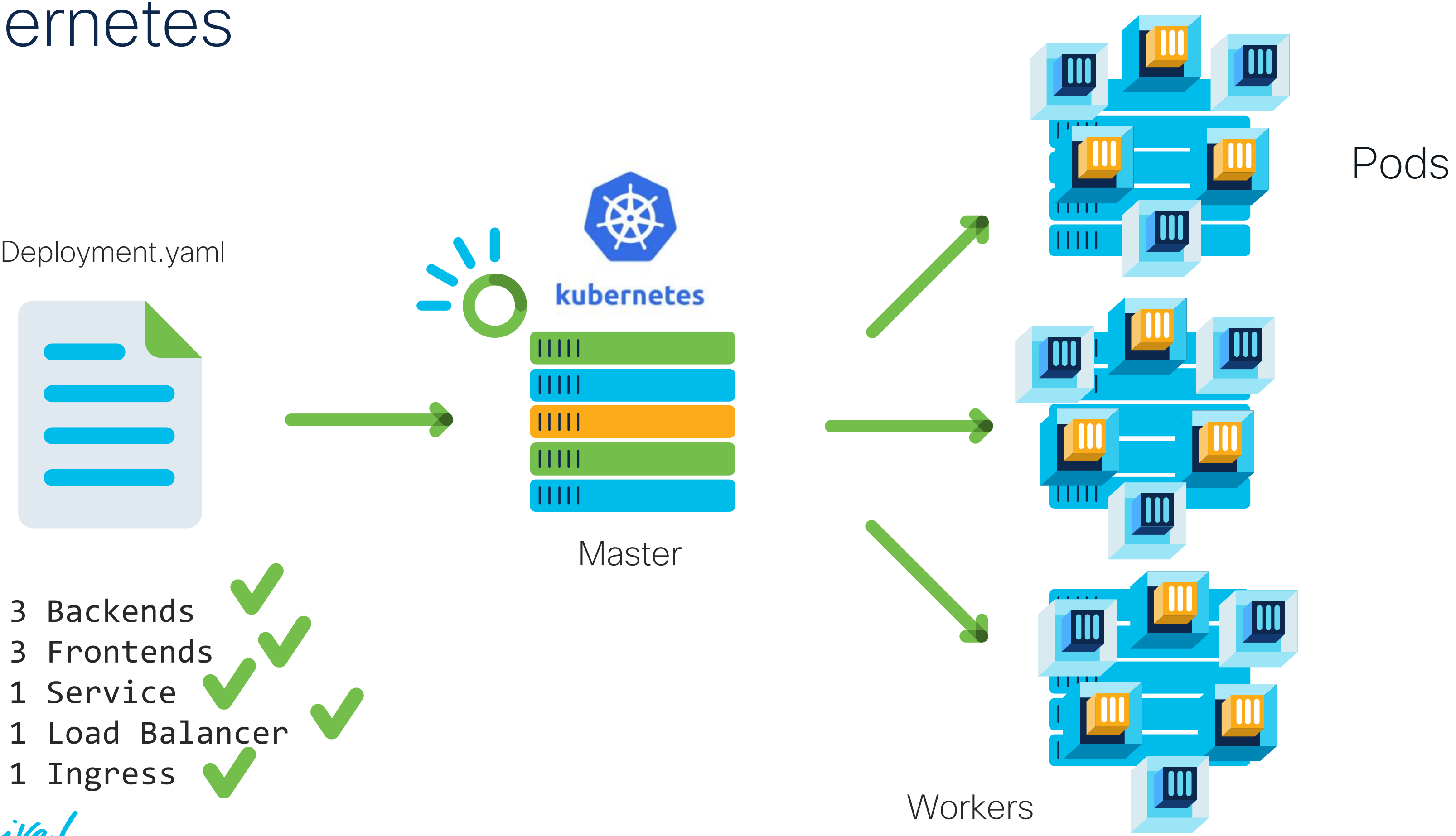
3 Backends ✓
3 Frontends ✓
1 Service ✓
1 Load Balancer ✓
1 Ingress ✓

Kubernetes



- 3 Backends ✓
- 3 Frontends ✓
- 1 Service ✓
- 1 Load Balancer ✓
- 1 Ingress ✓

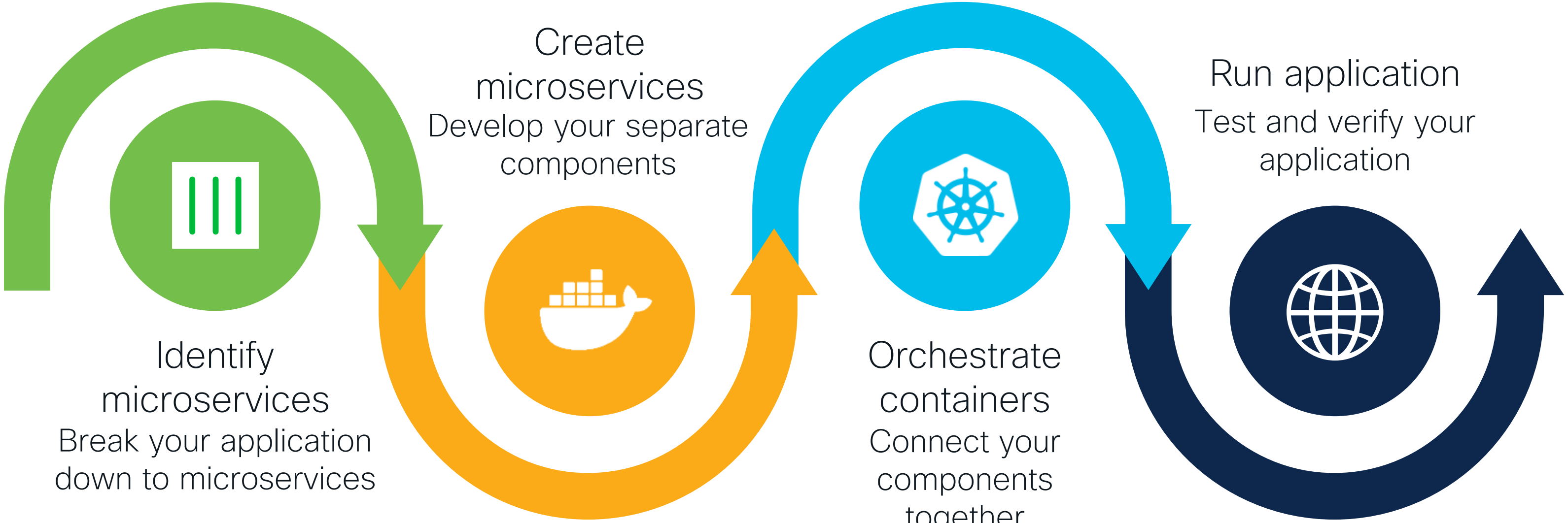
Kubernetes



```
3 Backends ✓
3 Frontends ✓
1 Service ✓
1 Load Balancer ✓
1 Ingress ✓
```

Basic Setup

microservices-basic



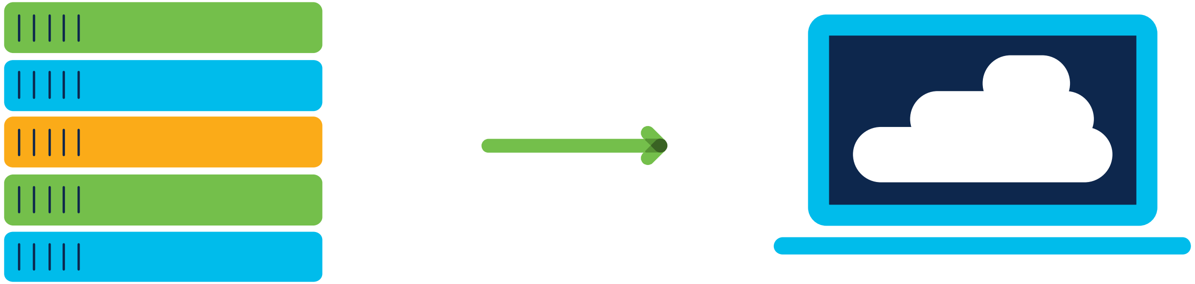
Identify microservices
Break your application down to microservices

Create microservices
Develop your separate components

Orchestrate containers
Connect your components together

Run application
Test and verify your application

microservices-basic

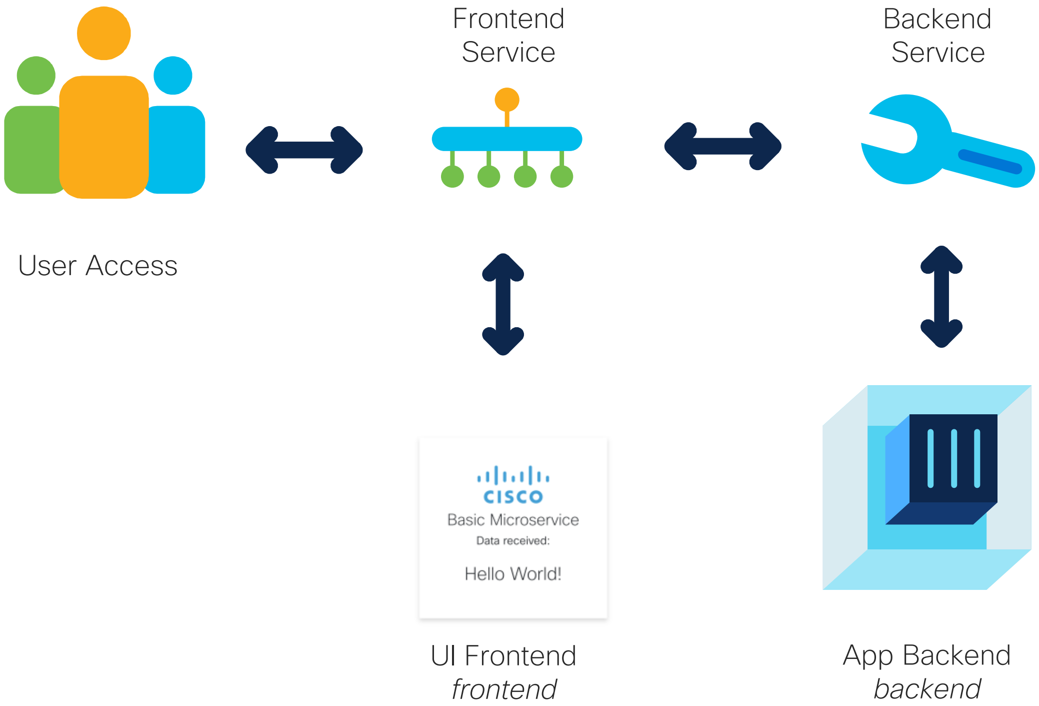


Backend

Frontend

https://github.com/joshingeniero/microservices-basic

Setup



cisco

Basic Microservice

Data received:

Hello World!

Deployment (Backend)

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: backend
spec:
  selector:
    matchLabels:
      app: backend
  replicas: 2
```

```yaml
template:
    metadata:
      labels:
        app: backend
    spec:
      containers:
        - name: backend
          image: backend
          imagePullPolicy: Never
          ports:
            - containerPort: 5002
      imagePullSecrets:
        - name: secret
```

Service (Backend)

```
apiVersion: v1
kind: Service
metadata:
  name: backend-service
spec:
  selector:
    app: backend
  ports:
    - protocol: "TCP"
      port: 6002
      targetPort: 5002
```

Deployment (Frontend)

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: frontend
spec:
  selector:
    matchLabels:
      app: frontend
  replicas: 2
```

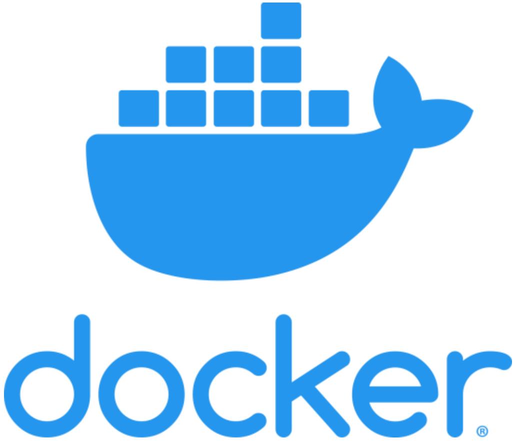
```
template:
  metadata:
    labels:
      app: frontend
  spec:
    containers:
      - name: frontend
        image: frontend
        imagePullPolicy: Never
        ports:
          - containerPort: 5001
    imagePullSecrets:
      - name: secret
```

Service (Frontend)

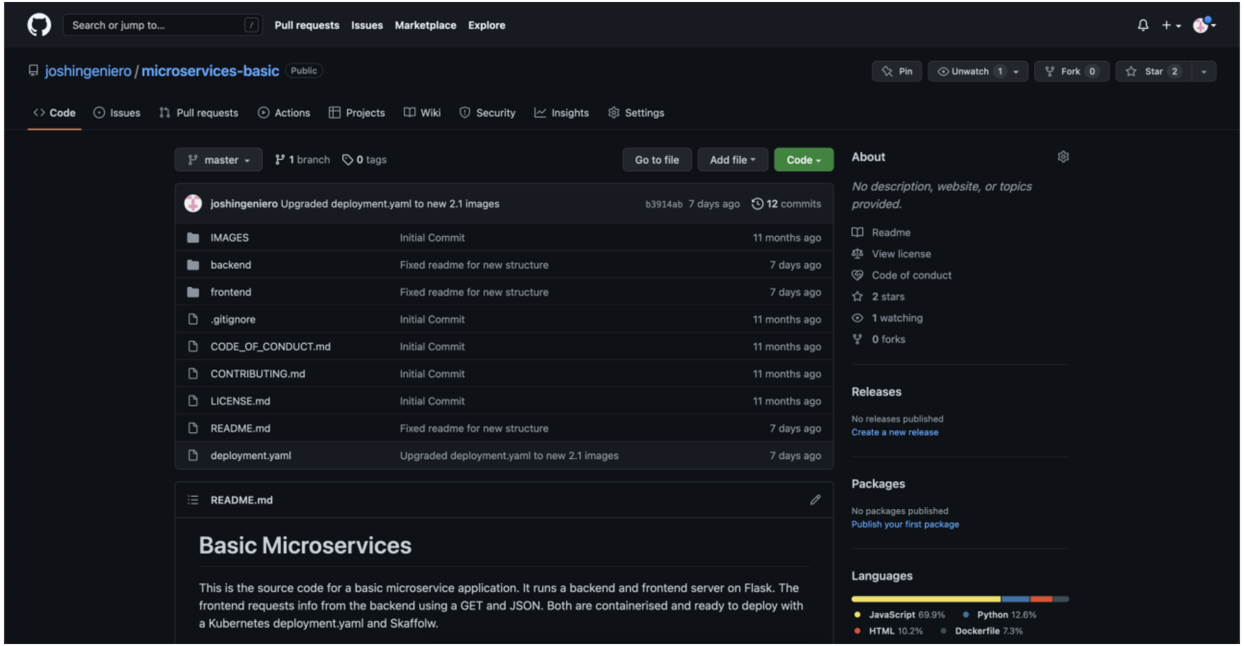
```
apiVersion: v1
kind: Service
metadata:
  name: frontend-service
spec:
  selector:
    app: frontend
  ports:
    - protocol: "TCP"
      port: 6001
      targetPort: 5001
  type: LoadBalancer
```

Demo

Setup



Clone Code



joshingeniero Upgraded deployment.yaml to new 2.1 images b3914ab 7 days ago 12 commits

IMAGES	Initial Commit	11 months ago
backend	Fixed readme for new structure	7 days ago
frontend	Fixed readme for new structure	7 days ago
.gitignore	Initial Commit	11 months ago
CODE_OF_CONDUCT.md	Initial Commit	11 months ago
CONTRIBUTING.md	Initial Commit	11 months ago
LICENSE.md	Initial Commit	11 months ago
README.md	Fixed readme for new structure	7 days ago
deployment.yaml	Upgraded deployment.yaml to new 2.1 images	7 days ago

README.md

Basic Microservices

This is the source code for a basic microservice application. It runs a backend and frontend server on Flask. The frontend requests info from the backend using a GET and JSON. Both are containerised and ready to deploy with a Kubernetes deployment.yaml and Skaffolw.

About

No description, website, or topics provided.

- Readme
- View license
- Code of conduct
- 2 stars
- 1 watching
- 0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

- JavaScript 69.9%
- Python 12.6%
- HTML 10.2%
- Dockerfile 7.3%

https://github.com/joshingeniero/microservices-basic

Running the application

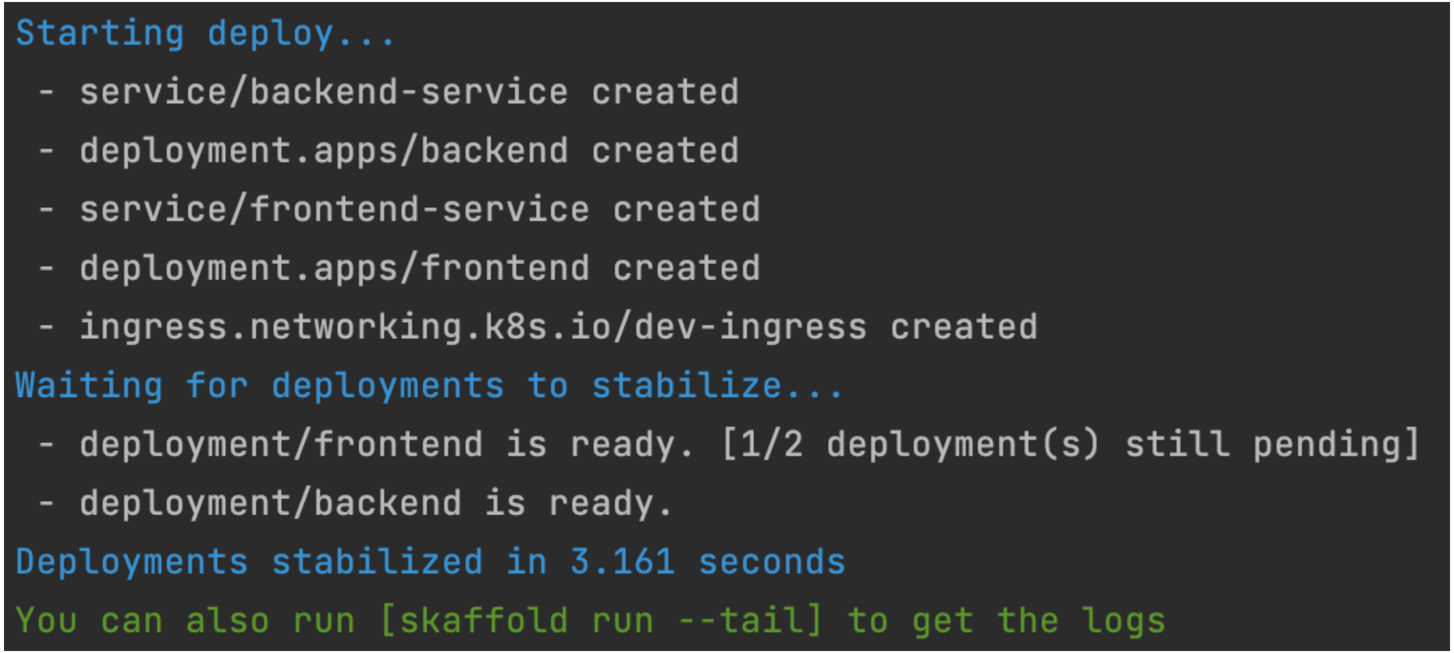
- Run the deployment once for testing

```
$ skaffold run
```

- Run the deployment continuously for developing

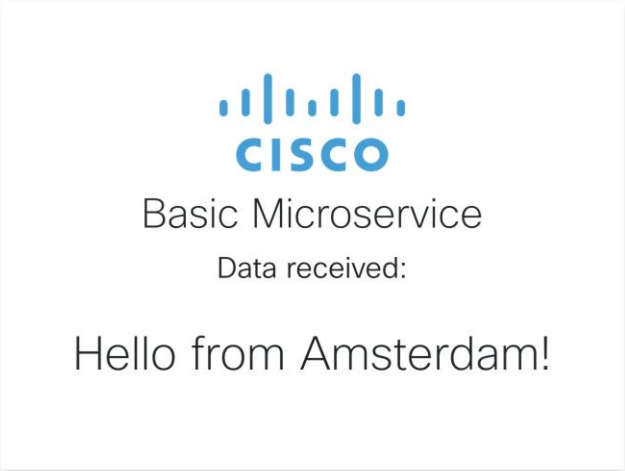
```
$ skaffold dev
```

Running the application



```
Starting deploy...
 - service/backend-service created
 - deployment.apps/backend created
 - service/frontend-service created
 - deployment.apps/frontend created
 - ingress.networking.k8s.io/dev-ingress created
Waiting for deployments to stabilize...
 - deployment/frontend is ready. [1/2 deployment(s) still pending]
 - deployment/backend is ready.
Deployments stabilized in 3.161 seconds
You can also run [skaffold run --tail] to get the logs
```

Testing the application



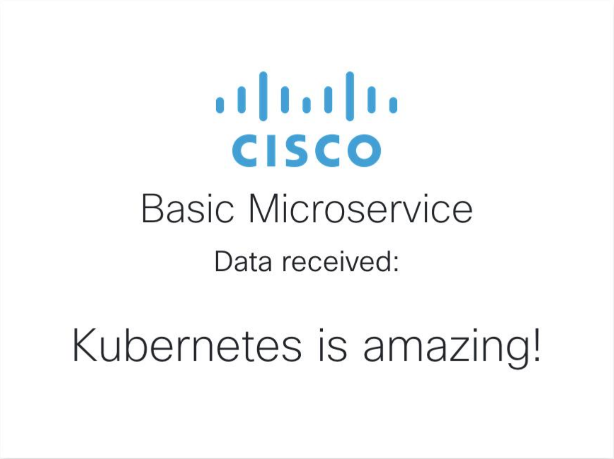
Basic Microservice

Data received:

Hello from Amsterdam!

http://localhost:6001/

Testing the application



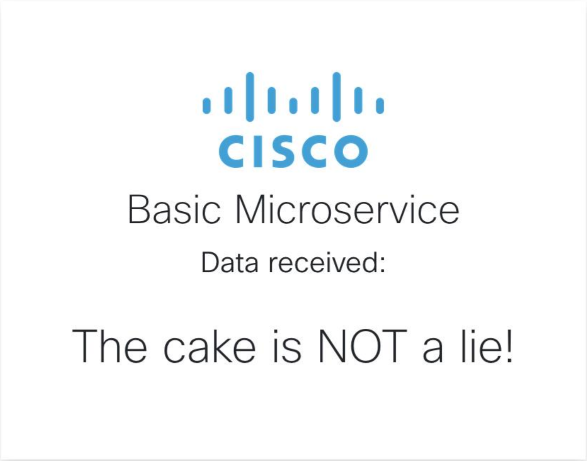
Basic Microservice

Data received:

Kubernetes is amazing!

http://localhost:6001/info

Testing the application



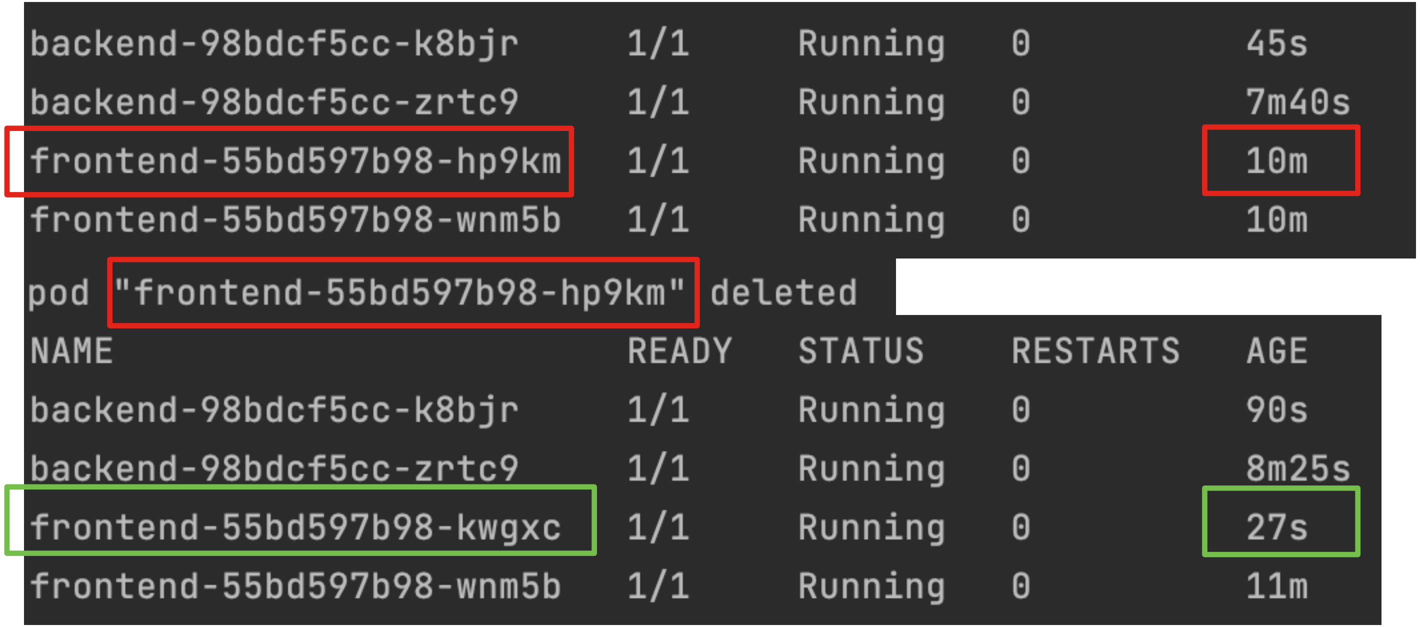
Basic Microservice

Data received:

The cake is NOT a lie!

http://localhost:6001/cake

Testing the application



```
backend-98bdcf5cc-k8bjr      1/1     Running   0          45s
backend-98bdcf5cc-zrtc9      1/1     Running   0          7m40s
frontend-55bd597b98-hp9km    1/1     Running   0          10m
frontend-55bd597b98-wnm5b    1/1     Running   0          10m
pod "frontend-55bd597b98-hp9km" deleted
NAME                         READY   STATUS    RESTARTS   AGE
backend-98bdcf5cc-k8bjr      1/1     Running   0          90s
backend-98bdcf5cc-zrtc9      1/1     Running   0          8m25s
frontend-55bd597b98-kwgxc    1/1     Running   0          27s
frontend-55bd597b98-wnm5b    1/1     Running   0          11m
```

Deleting a pod

Testing the application

```json
{
  "data": {
    "root": "Hello from Melbourne!",
    "info": "Kubernetes is amazing!",
    "maker": "The cake is NOT a lie!"
  }
}
```

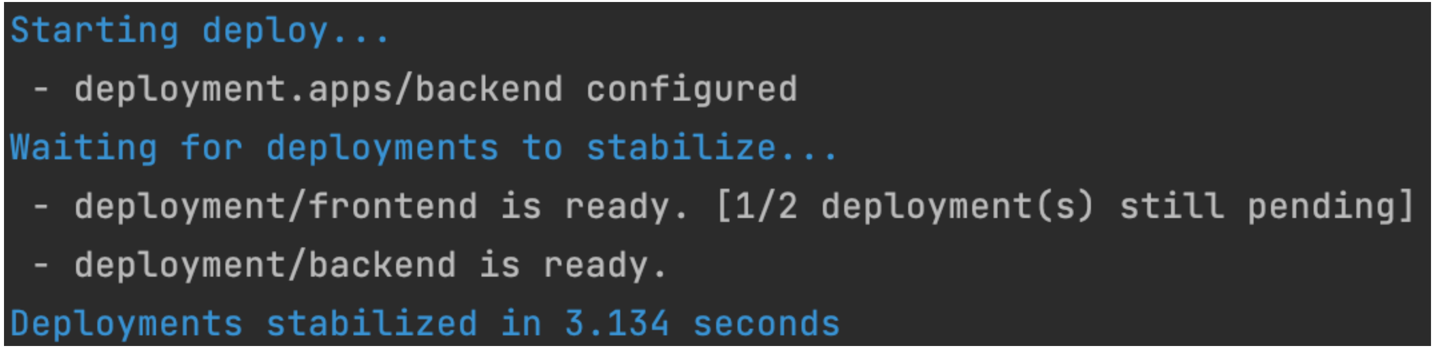
backend/database.json

Modifying the application

```
{
  "data": {
    "root": "Hello from Cisco Live!",
    "info": "You are amazing!",
    "maker": "The cake is a lie!"
  }
}
```

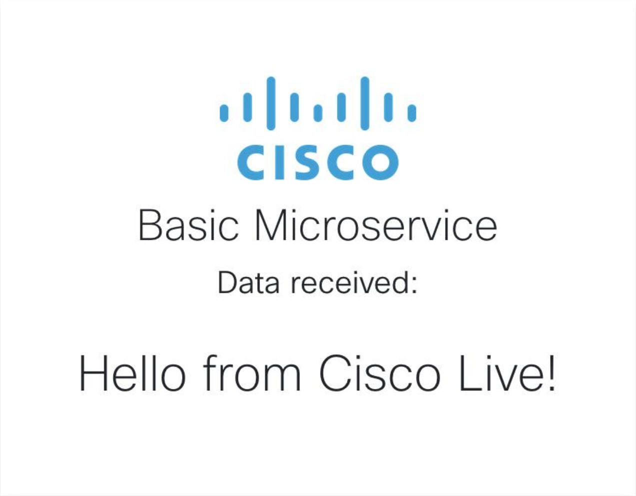
backend/database.json

Modifying the application



```
Starting deploy...
 - deployment.apps/backend configured
Waiting for deployments to stabilize...
 - deployment/frontend is ready. [1/2 deployment(s) still pending]
 - deployment/backend is ready.
Deployments stabilized in 3.134 seconds
```

Modifying the application



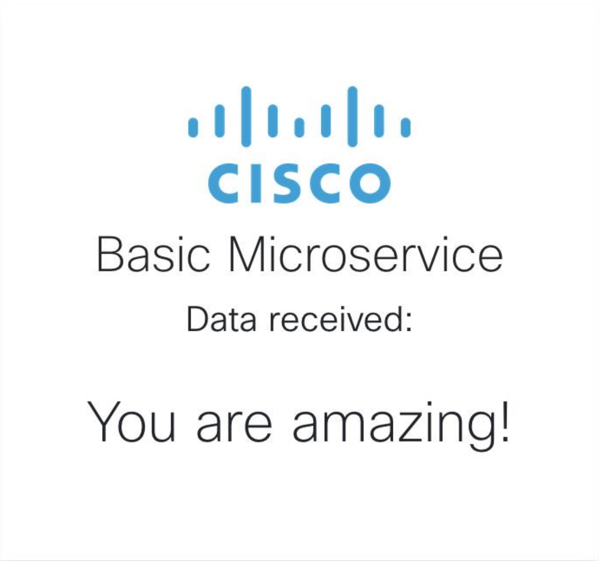
Basic Microservice

Data received:

Hello from Cisco Live!

http://localhost:6001/

Modifying the application



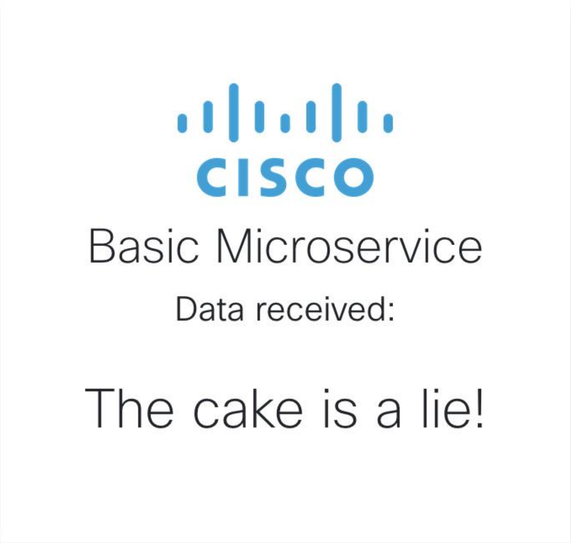
Basic Microservice

Data received:

You are amazing!

http://localhost:6001/info

Modifying the application



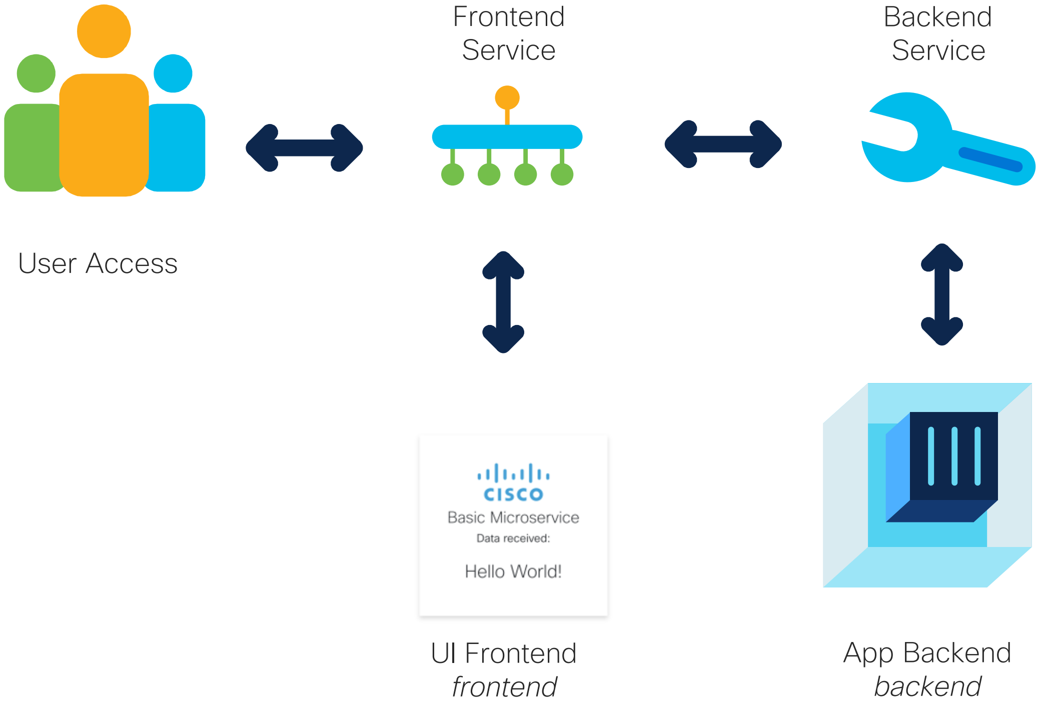
Basic Microservice

Data received:

The cake is a lie!

http://localhost:6001/cake

Your first deployment!



Frontend Service

Backend Service

User Access

Basic Microservice

Data received:

Hello World!

UI Frontend
frontend

App Backend
backend

In summary…

- Cloud-native development is a **Journey**
- Containers **enable** the microservice infrastructure
- Kubernetes defines a **state** for your applications and spins up the necessary pods and services
- Cloud-native development can help you develop **agile**, **scalable**, and **unique** applications

Call to Action

- Get your hands on the microservices-basic demo
  - https://github.com/joshingeniero/microservices-basic
- Discover Docker and containers
  - https://www.docker.com/101-tutorial
- Try out Kubernetes
  - https://kubernetes.io/docs/tutorials/
  - https://kubernetes.io/docs/concepts/security/overview/
- Explore Cisco DevNet
  - https://developer.cisco.com/startnow

Complete your Session Survey

- Please complete your session survey after each session. Your feedback is important.
- Complete a minimum of 4 session surveys and the Overall Conference survey (open from Thursday) to receive your Cisco Live t-shirt.

- All surveys can be taken in the Cisco Events Mobile App or by logging in to the Session Catalog and clicking the "Attendee Dashboard" at https://www.ciscolive.com/emea/learn/sessions/session-catalog.html

Continue Your Education

- Visit the Cisco Showcase for related demos.
- Book your one-on-one Meet the Engineer meeting.
- Attend any of the related sessions at the DevNet, Capture the Flag, and Walk-in Labs zones.
- Visit the On-Demand Library for more sessions at ciscolive.com/on-demand.

The bridge to possible

Thank you

cisco Live!