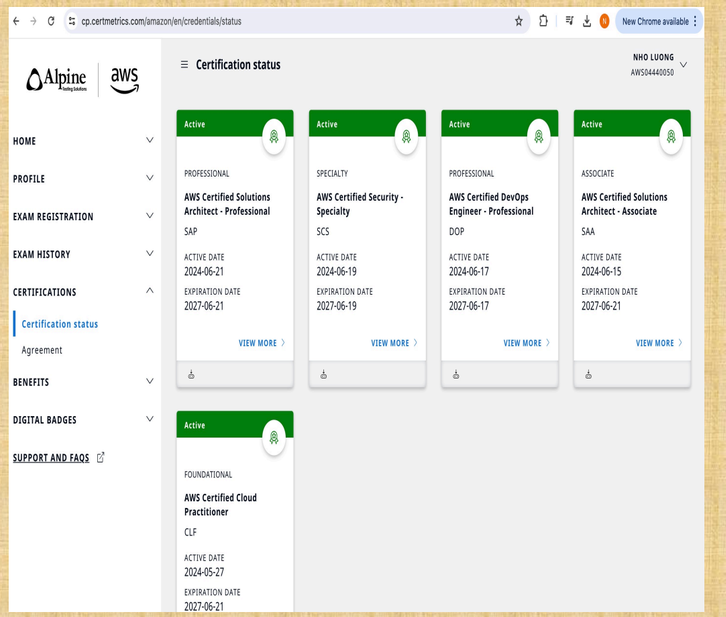
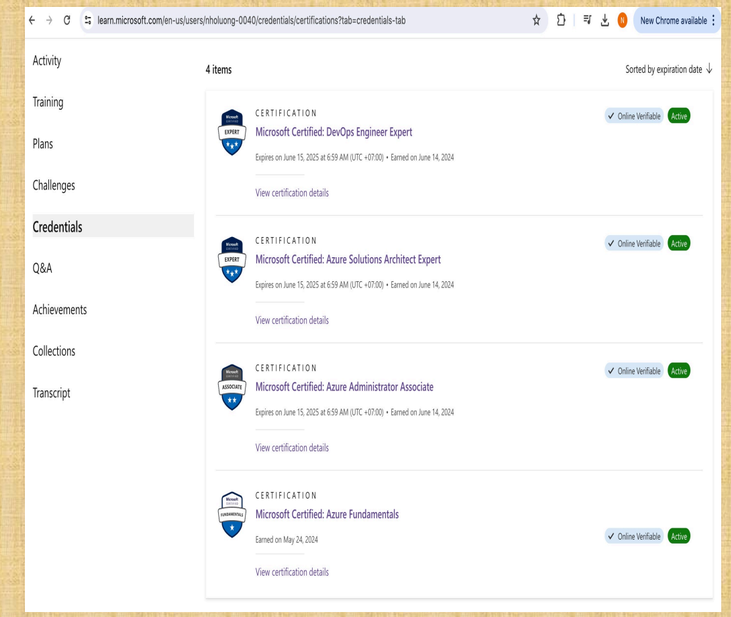
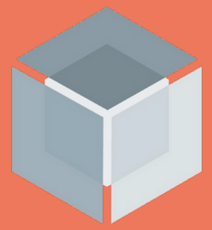
Microservices and Deploying Methodologies

Author: Nho Luong
Skill: DevOps Engineer Lead







Monolithic Applications

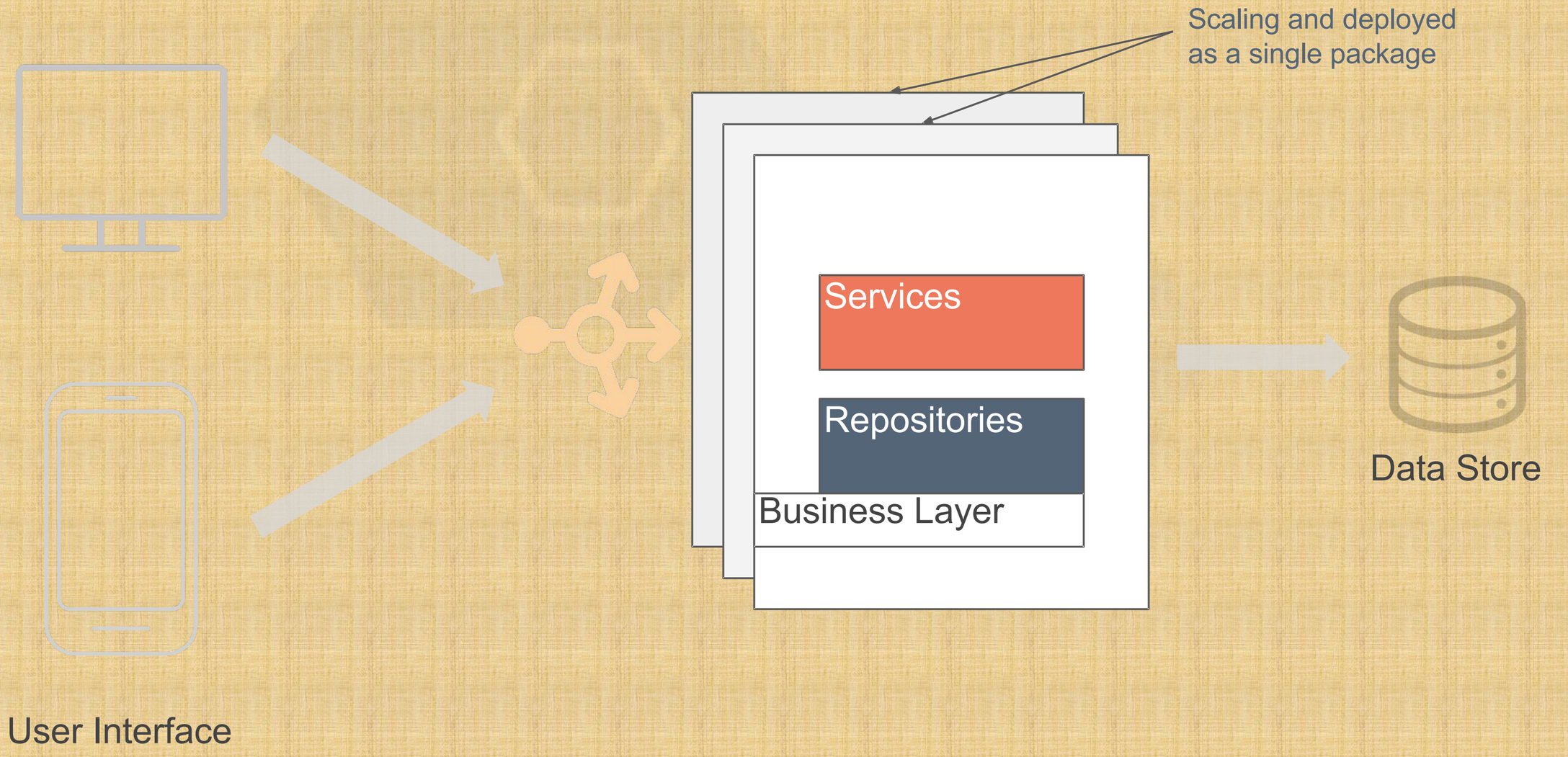
Traditional way to build enterprise app, and deployed and scaled as a single package

Main Components:

- Client-side user interface
- Server-side application
  - Design Pattern(Services & Repositories)
- A database server

Author: Nho Luong
Skill: DevOps Engineer Lead

Monolithic Application



Author: Nho Luong
Skill: DevOps Engineer Lead

Monolithic Application: Example

Markets

- market_id
- market_name

Products

- product_id
- product_name
- product_price
- market_id

Users

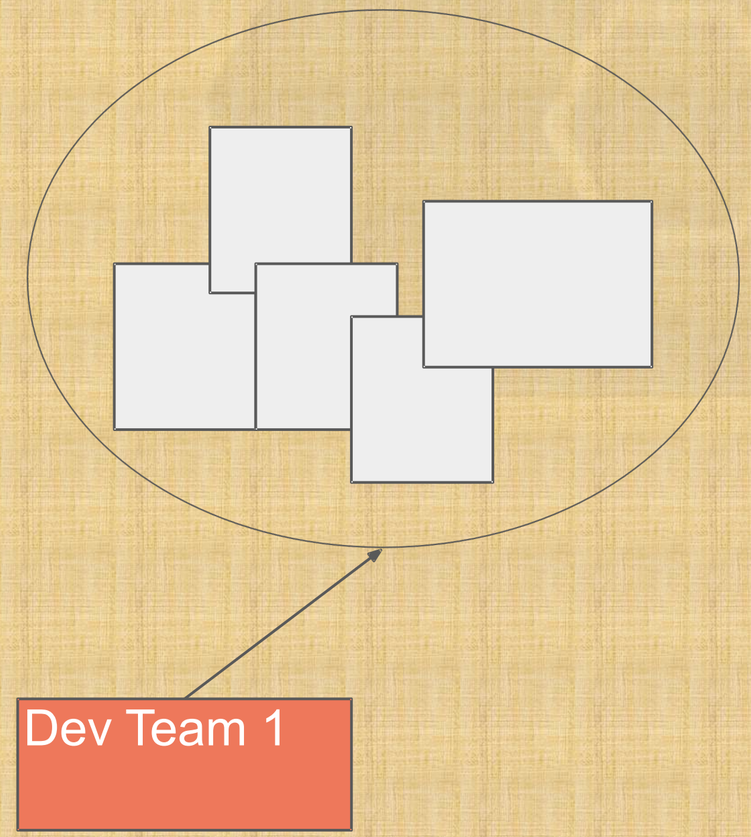
- user_id
- user_name
- market_id

A query to get the list of products and market name of the logged in user say with id=40 under price 300?

```
SELECT product_name, market_name FROM products
INNER JOIN markets on products.maket_id = markets.id INNER JOIN
users on users.market_id = markets.id WHERE users.user_id = 40
AND products.product_price < 300
```

Author: Nho Luong
Skill: DevOps Engineer Lead

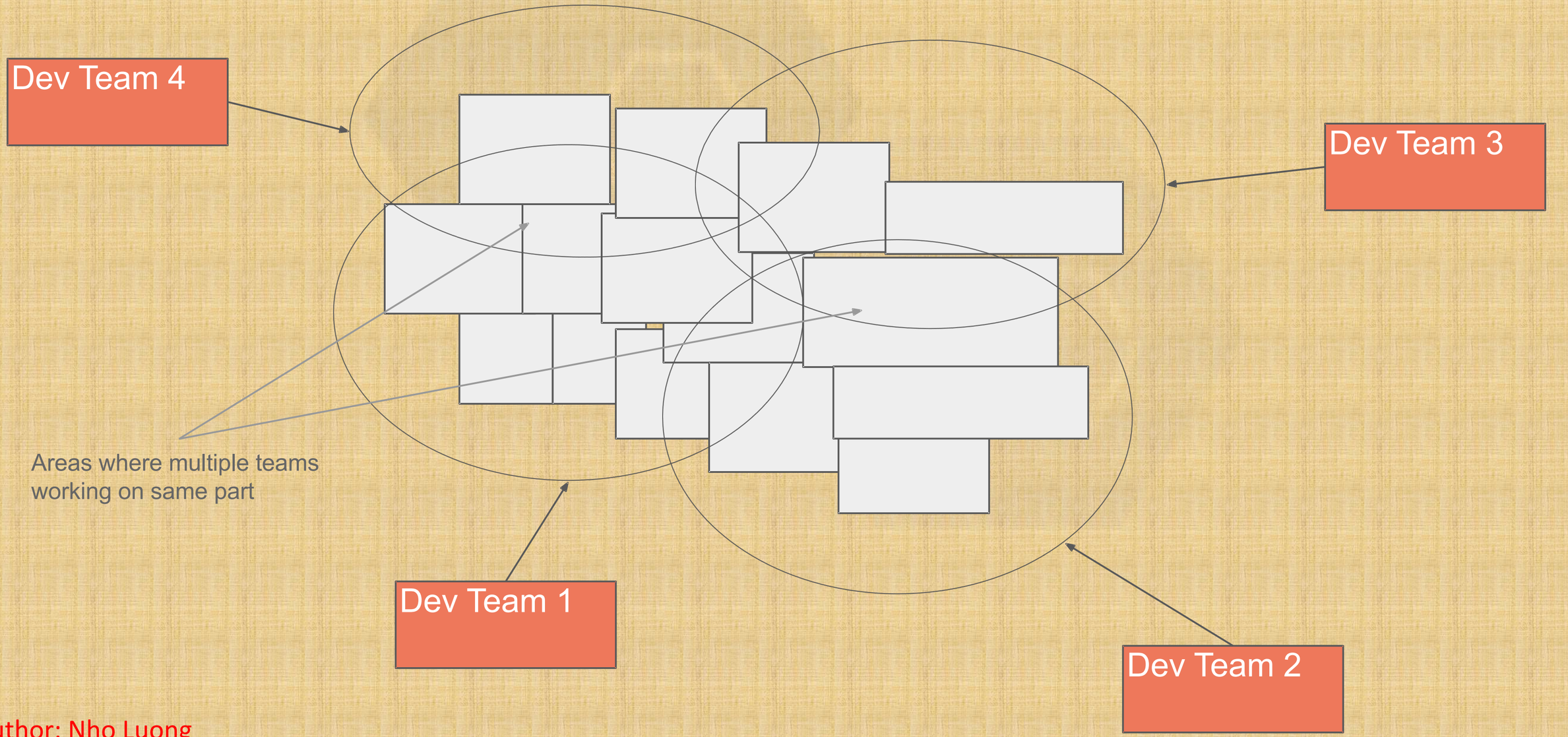
Monolithic Application - MVP version



What if a big hit?

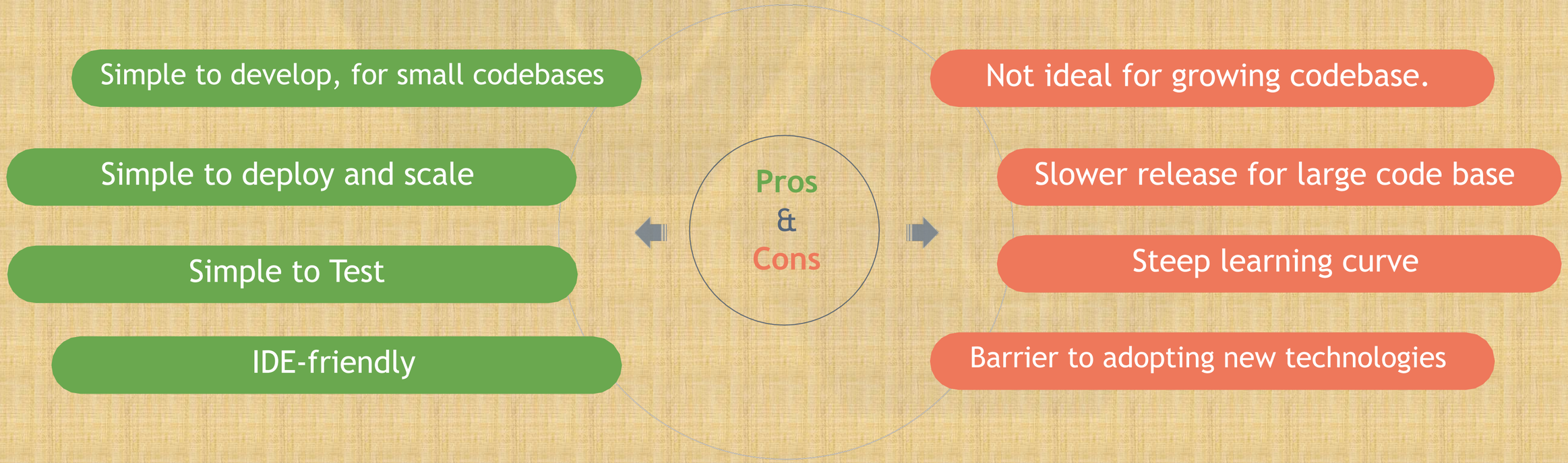
Author: Nho Luong
Skill: DevOps Engineer Lead

Monolithic Application - After few releases

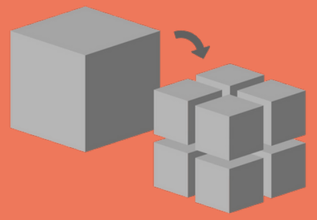


Author: Nho Luong
Skill: DevOps Engineer Lead

Monolithic - Pros / Cons



Author: Nho Luong
Skill: DevOps Engineer Lead



Micro Services

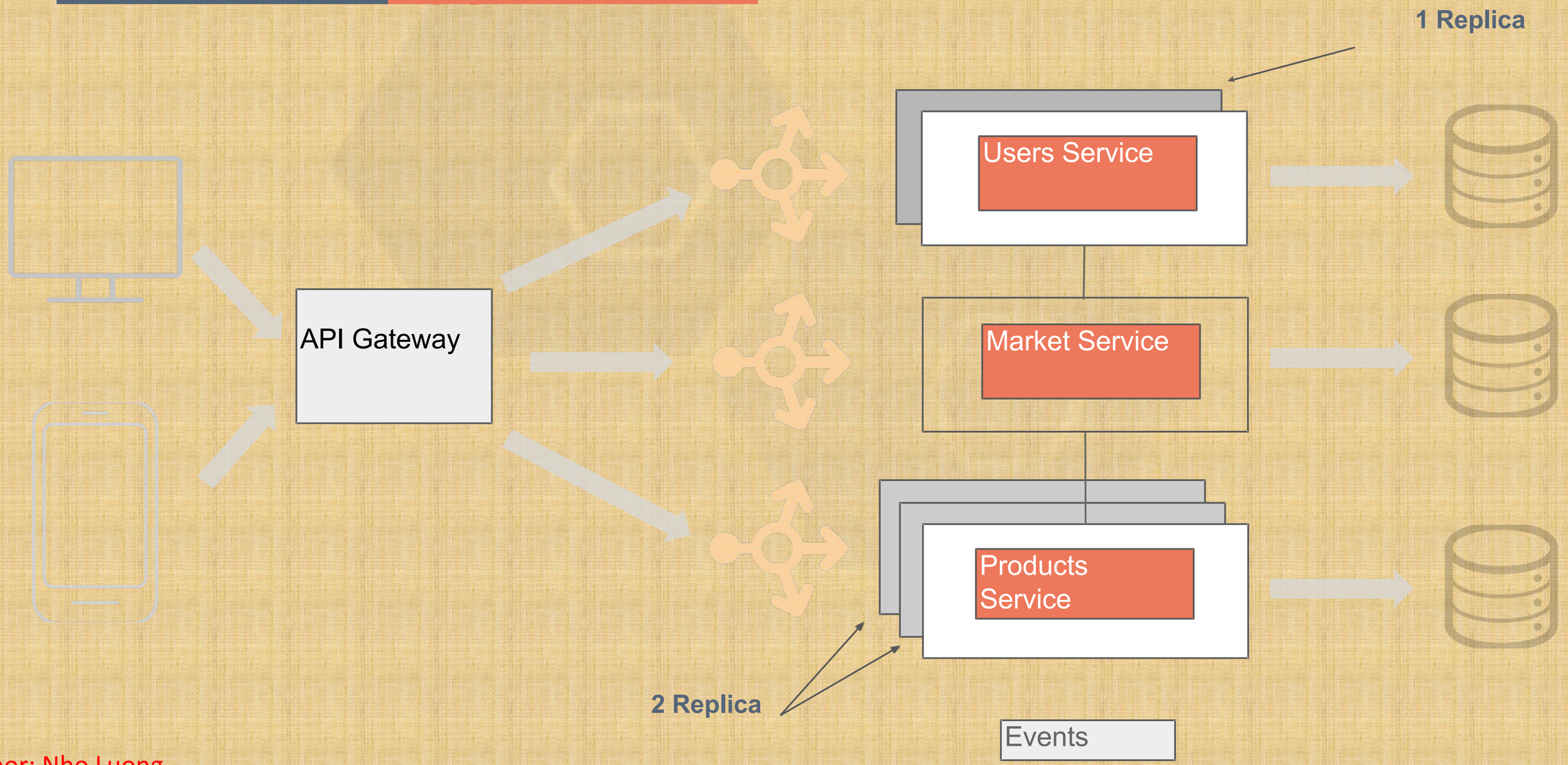
Suites of independently deployable services

Design Considerations

- Organized around business capability
- Decentralized Database
- Loosely Coupled and Highly Cohesive
- Independently deployed

Author: Nho Luong
Skill: DevOps Engineer Lead

Microservice Application



Author: Nho Luong
Skill: DevOps Engineer Lead

Microservice Application: Example

markets
{
id: 1, name: US,
currency: '$',
created_at: '123',
updated_at: 'abc'
}

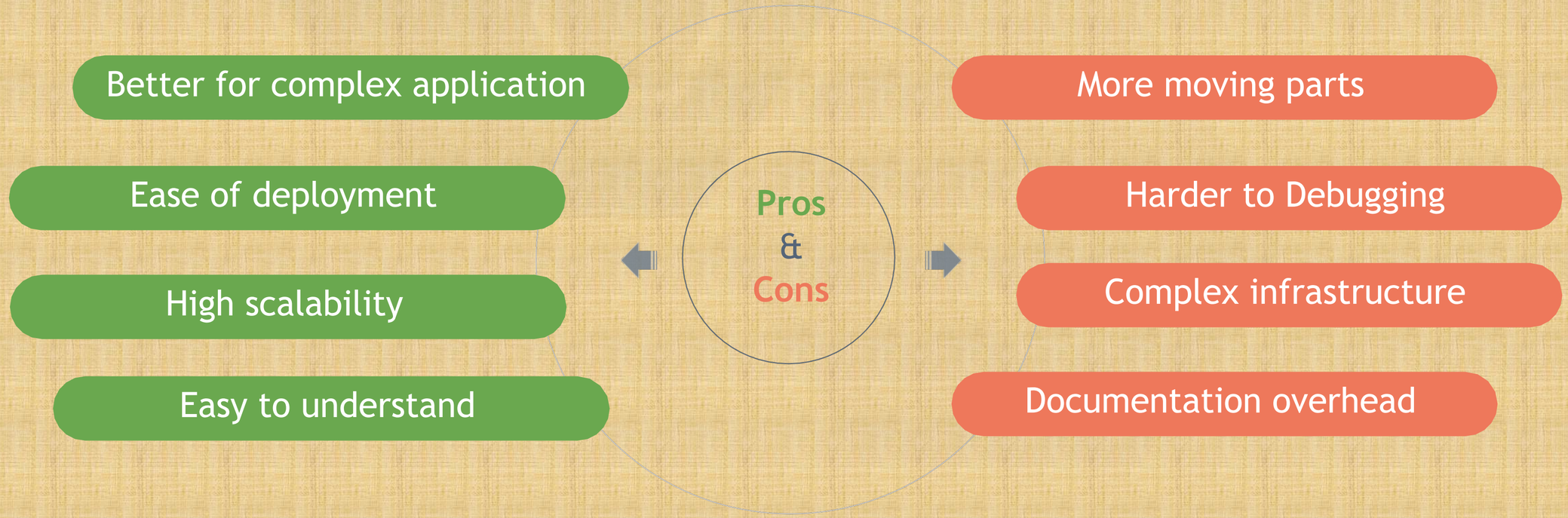
products
{
id: 1,
name: 'AWS' price:
300, market: {
id: 1,
name: US
}
}

Users
{
id: 1,
name: 'John Smith'
market: {
id: 1,
name: US
}
}

- Separate database per service as per business domain
- Maintain Duplicacy of data with single source of truth
- Eventual Consistency among services

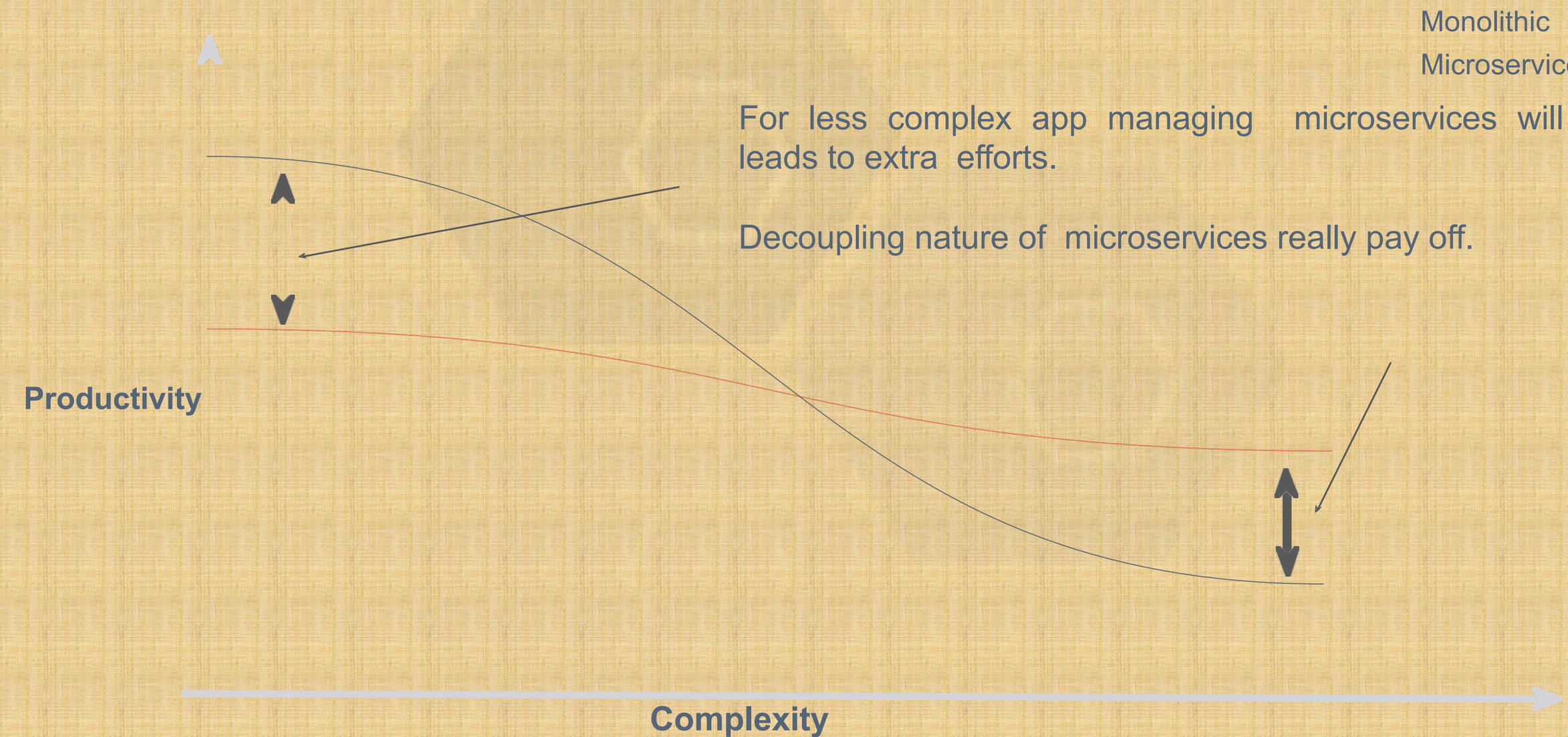
Author: Nho Luong
Skill: DevOps Engineer Lead

Microservice - Pros / Cons



Author: Nho Luong
Skill: DevOps Engineer Lead

Comparison (Productivity vs Complexity)



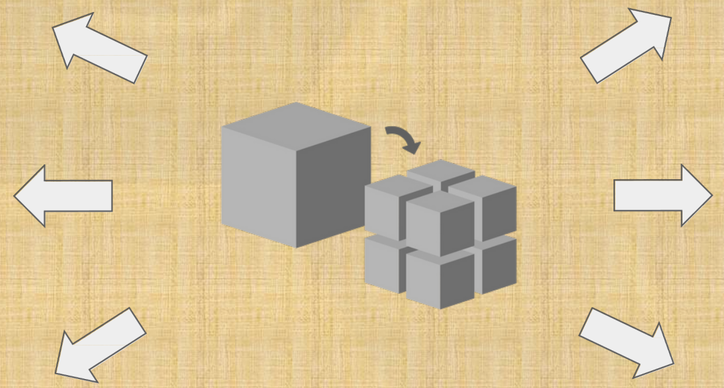
Author: Nho Luong
Skill: DevOps Engineer Lead

Deployment Challenges with Microservices

Deployment and configuration of high number of services

Update strategy with zero downtime

Scaling of particular services and distribution of traffic



Rollback in case of any trouble

Monitoring of services for node failures

Service Logs and Debugging

Author: Nho Luong
Skill: DevOps Engineer Lead

Kubernetes (K8s)

An open-source system for automating deployment, scaling, and management of containerized apps

Why to Choose?

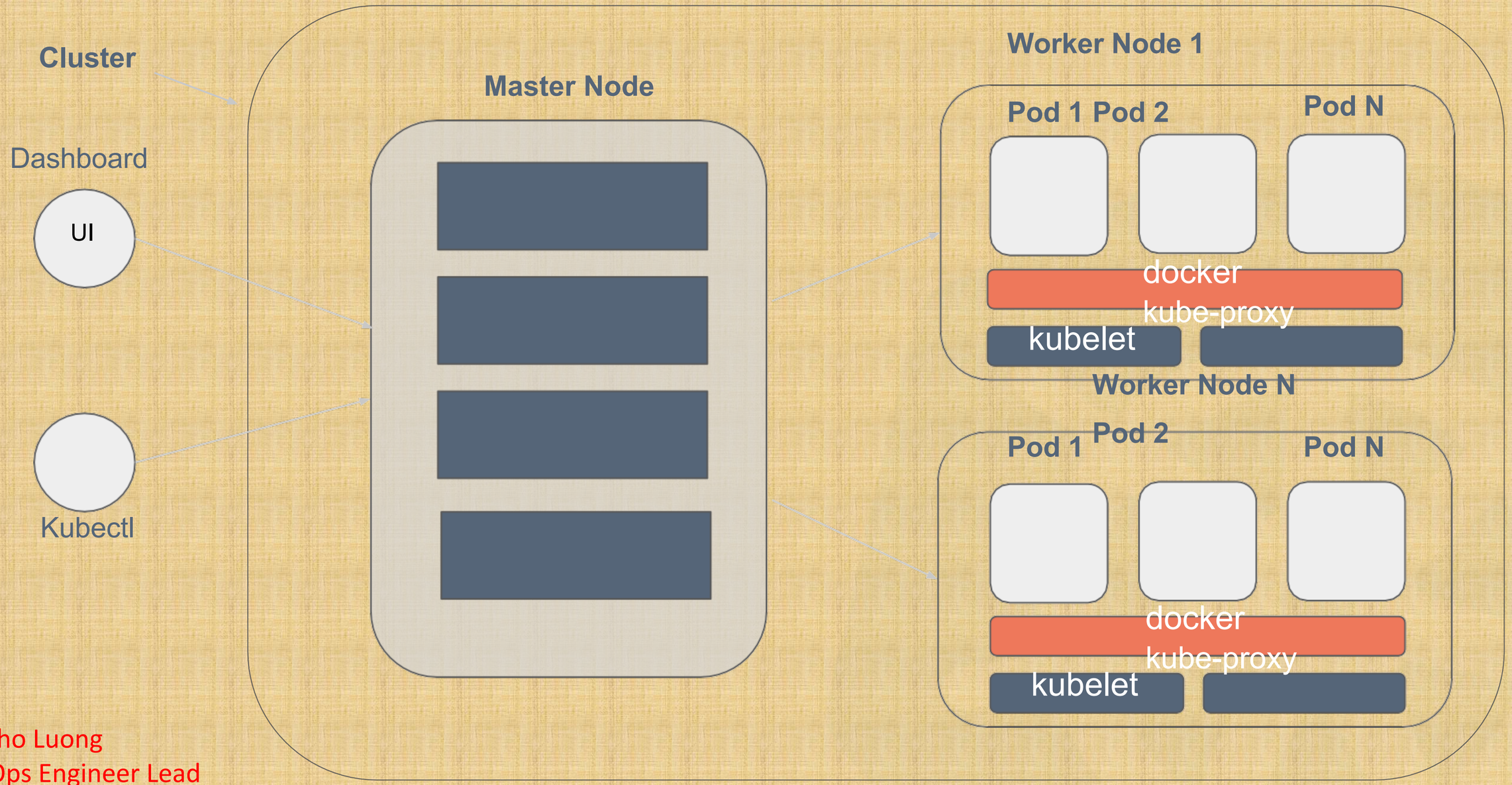
- Invented by Google
- Managed by open source community
- Adopted by Microsoft and Amazon
- Run Anywhere

Author: Nho Luong
Skill: DevOps Engineer Lead

Overview of K8s

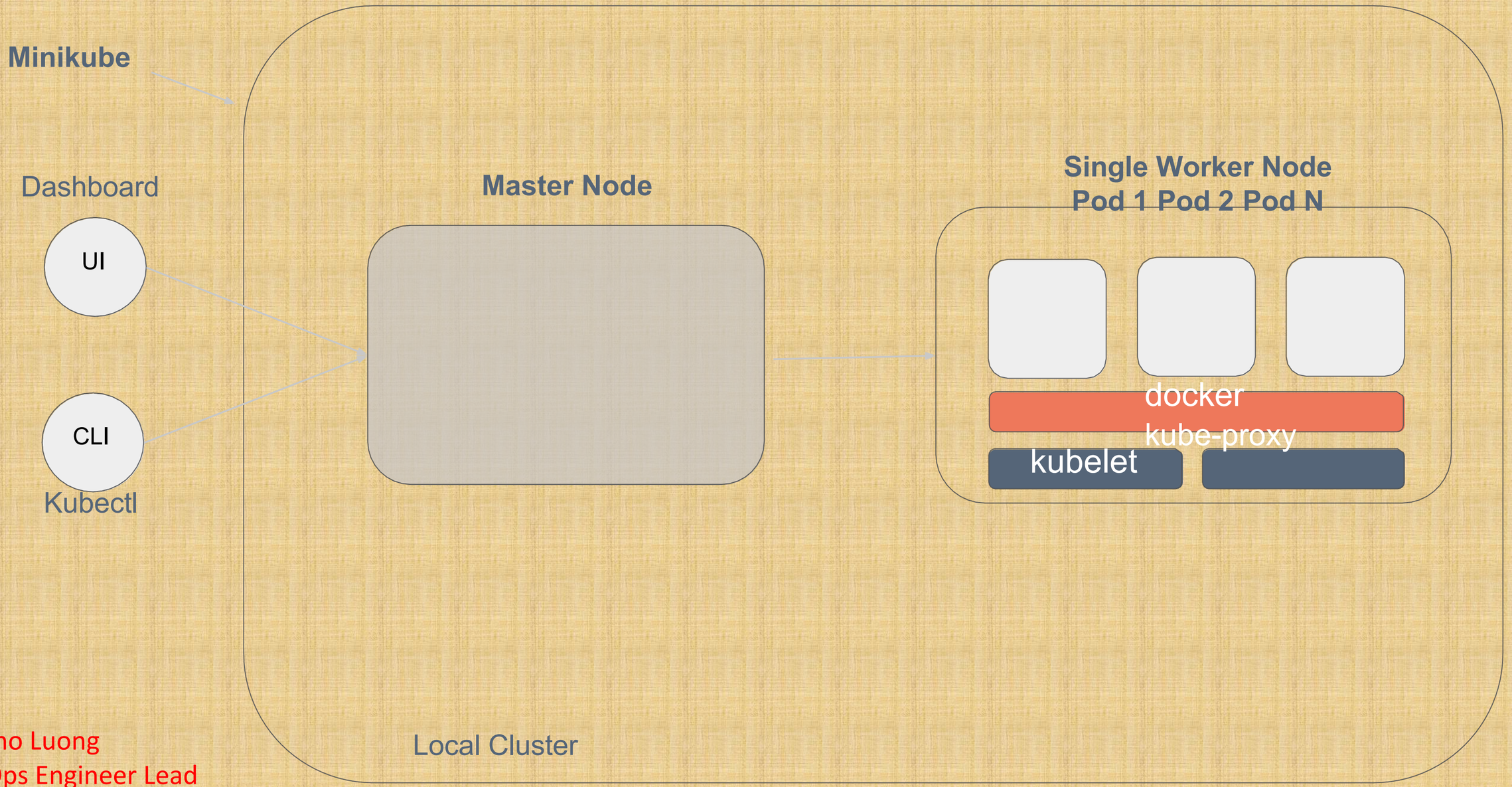
Author: Nho Luong
Skill: DevOps Engineer Lead

Basic components of Kubernetes (Cloud)



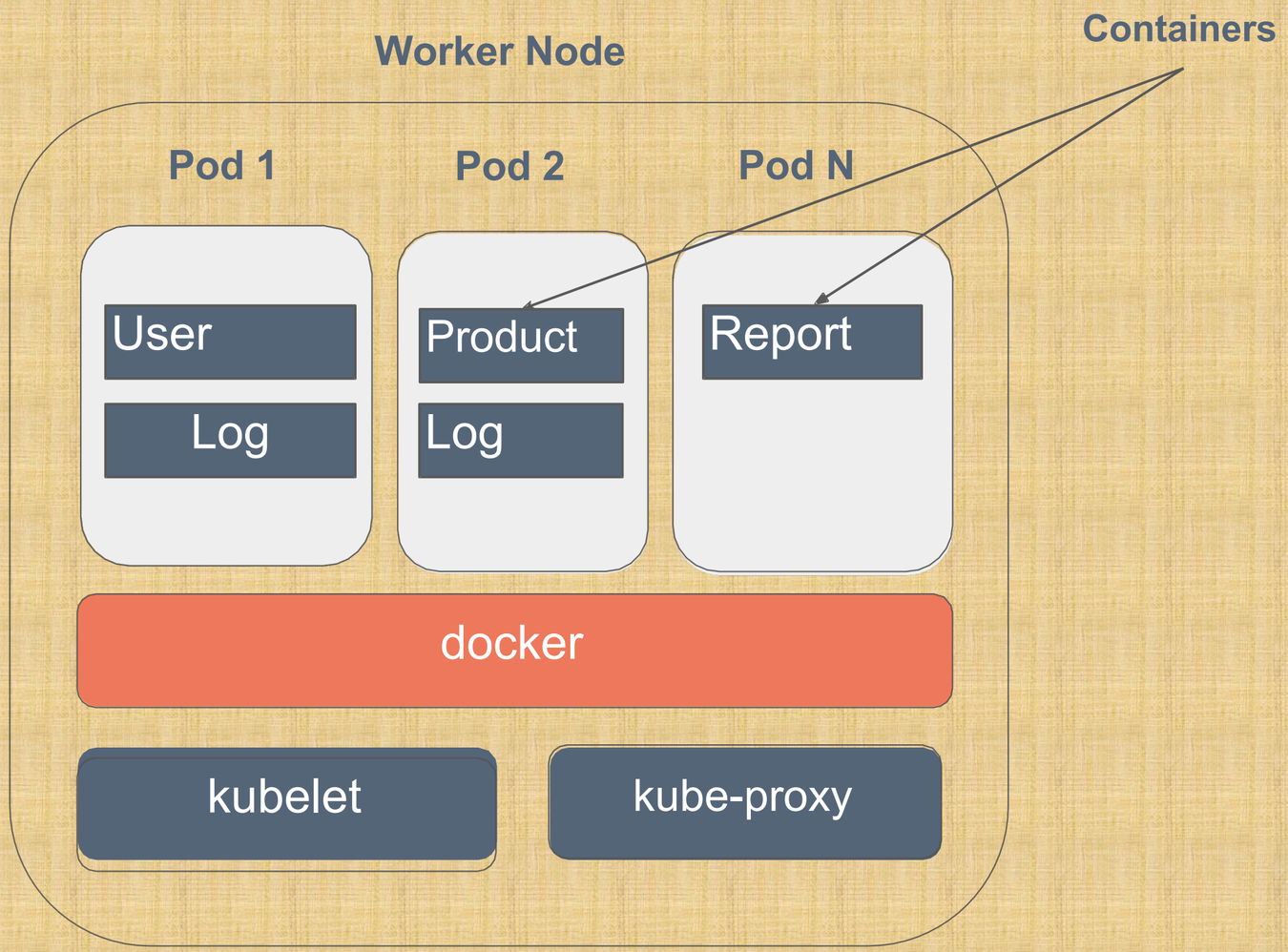
Author: Nho Luong
Skill: DevOps Engineer Lead

Basic components of Kubernetes (local)



Author: Nho Luong
Skill: DevOps Engineer Lead

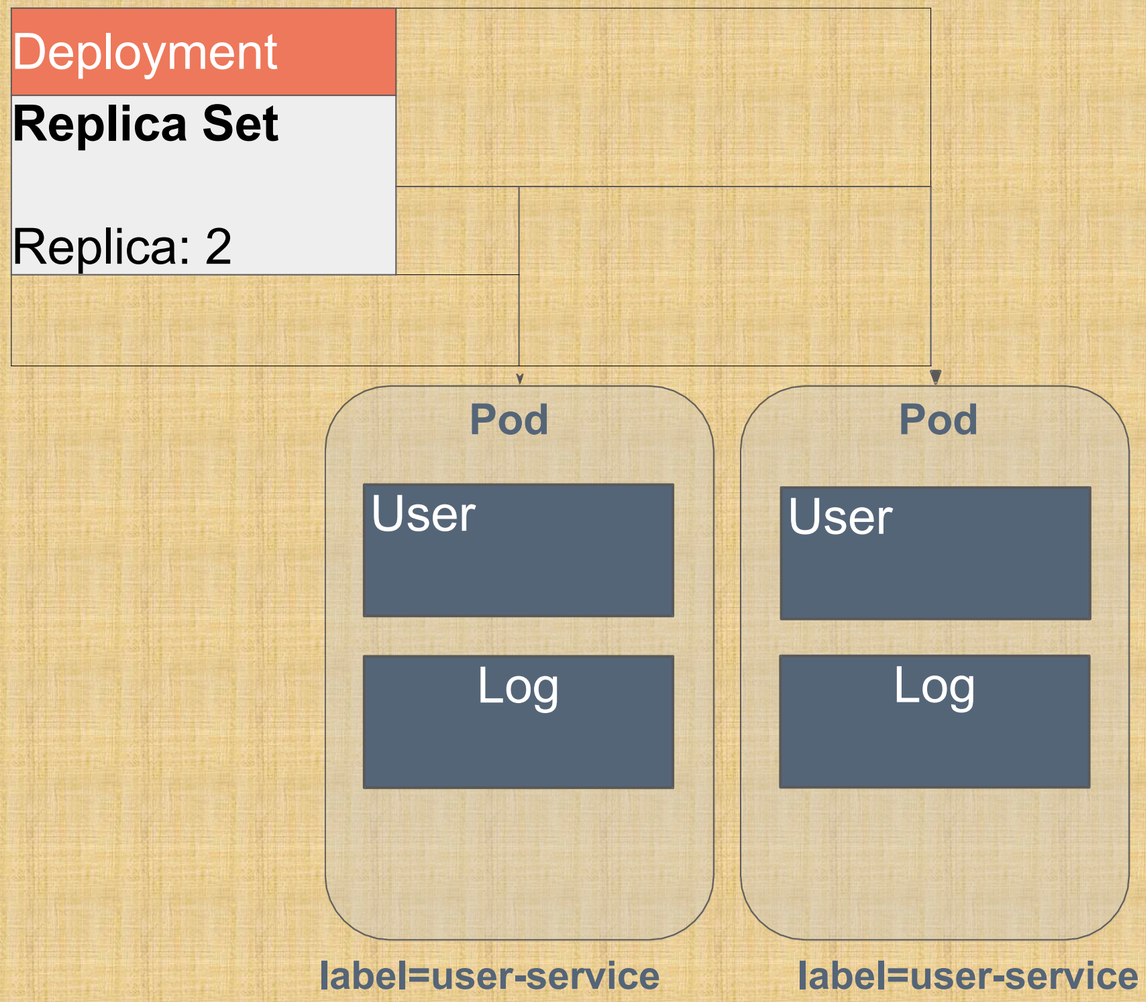
Basic components: Pods



- Basic and smallest building block.
- All the connected containers deployed to a single pod
- Containers inside pod will live and die together
- All containers have common IP, and resources.
- Each POD has unique IP.

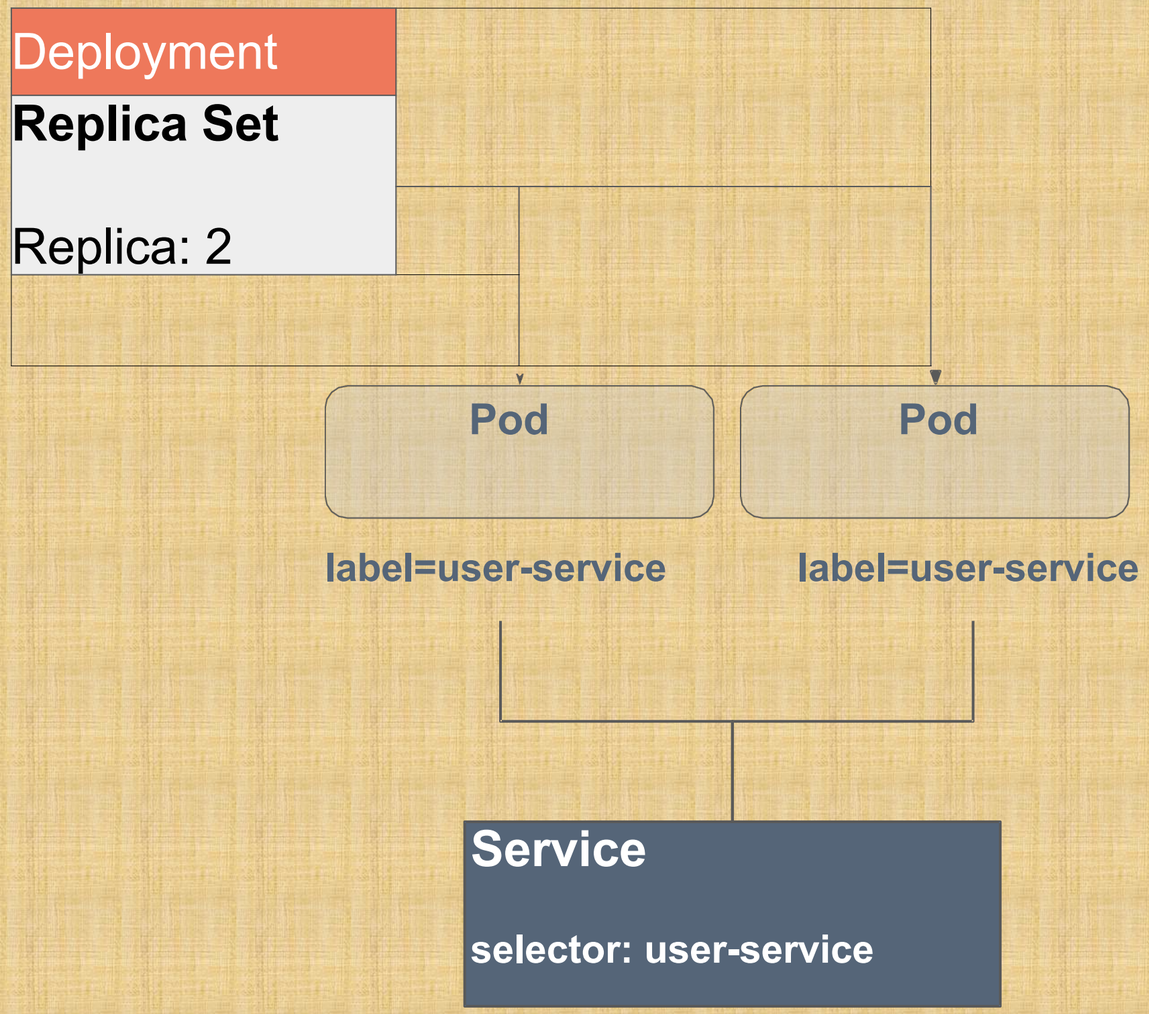
Author: Nho Luong
Skill: DevOps Engineer Lead

Basic components: Deployments



- Scaling up and down of the pods.
- Manages releases and rollback
- Self Healing of system
- Assign a unique label to every set of pods, and used by services for discovery.

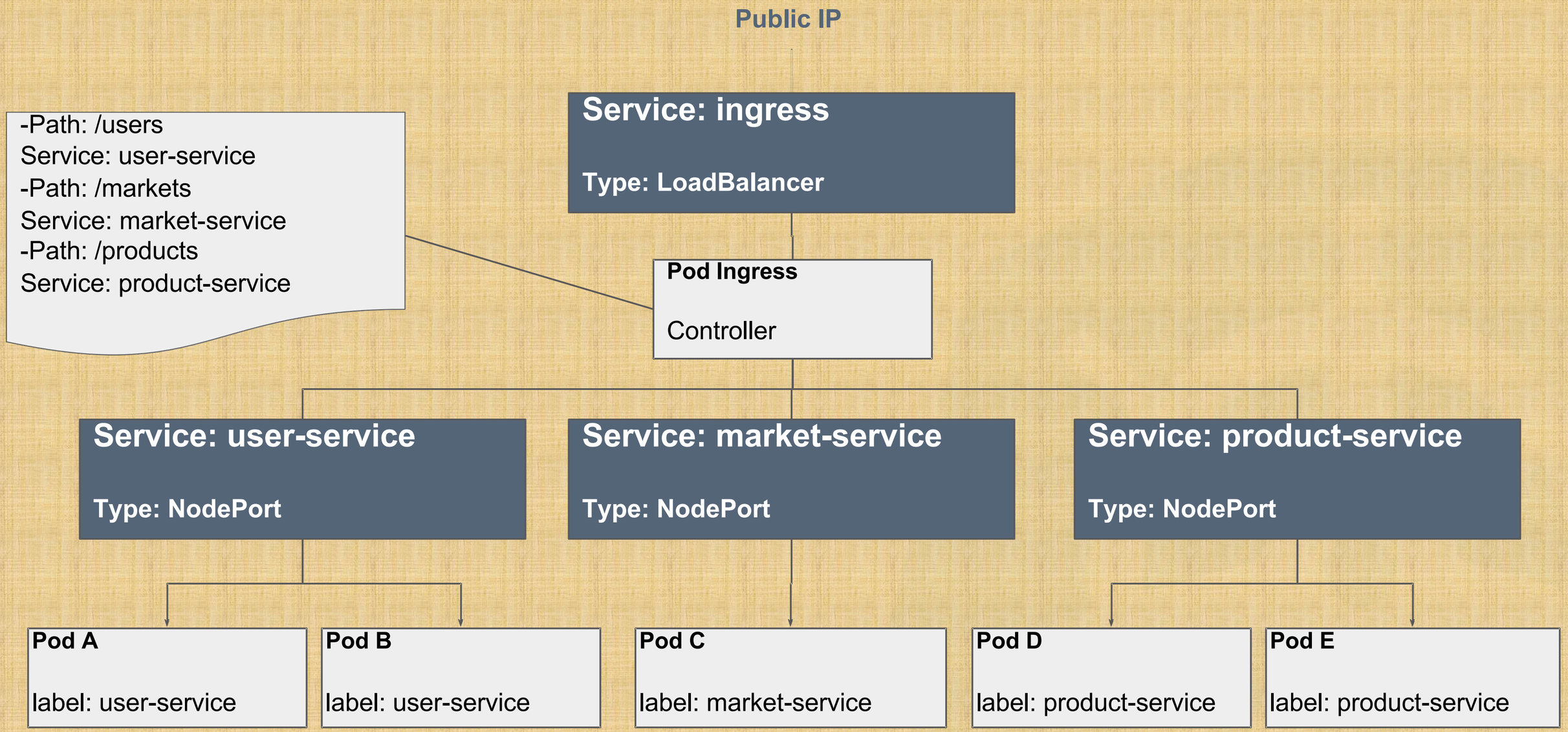
Basic components: Service



- Stable endpoint that load balances traffic among pods with similar label
- Discover the respective pods by the selectors
- With Service you don't have to remember how many pods are running or where

Author: Nho Luong
Skill: DevOps Engineer Lead

Basic components: Ingress



K8s: Rollbacks

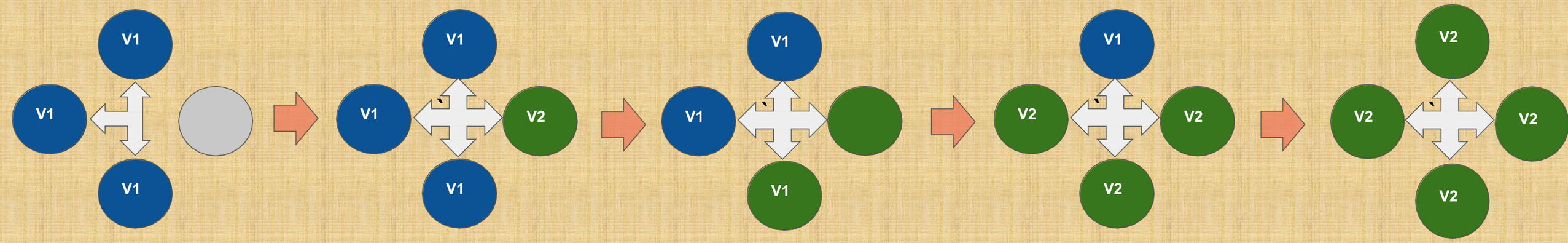
- Rolling out your application to previous version
- Set the revision that you want
- Kubernetes will scale up the corresponding ReplicaSet, and scaled down the current one

Handy Commands:

- kubectl rollout history deployments products
- kubectl rollout status deployments products
- kubectl rollout undo deployments products

Author: Nho Luong
Skill: DevOps Engineer Lead

Deployment Strategies: Rolling Updates



- Replace each pod in the deployment with a new one
- Backwards compatibility
- New ReplicaSet created and old ones gets decreased

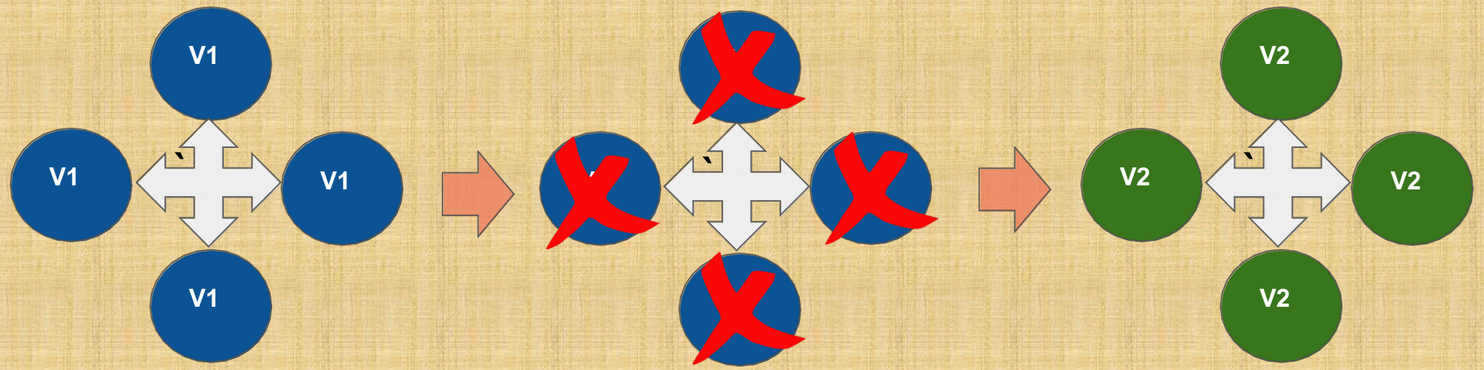
Author: Nho Luong
Skill: DevOps Engineer Lead

Deployment Strategies: Rolling Updates (example)

```yaml
spec:
  replicas: 3
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 2        # how many pods we can add at a time
      maxUnavailable: 0  # maxUnavailable define how many pods can be unavailable during the rolling update
```

Author: Nho Luong
Skill: DevOps Engineer Lead

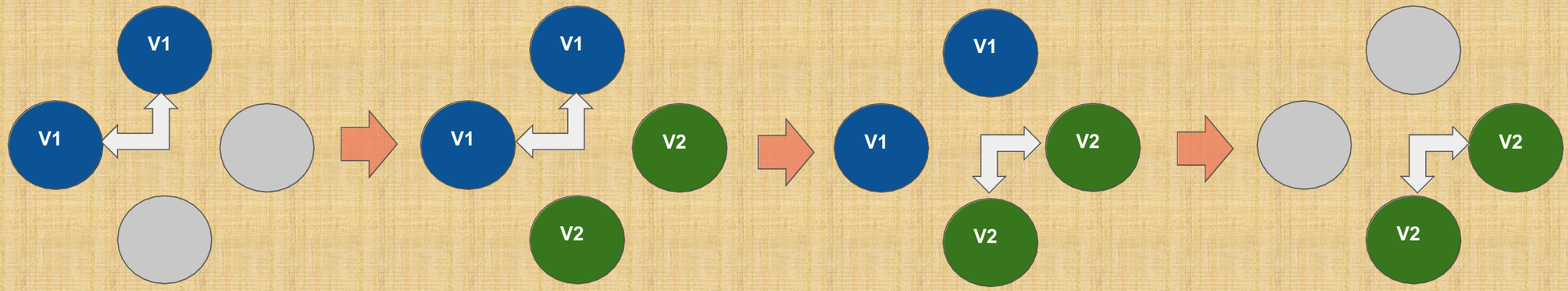
Deployment Strategies: Recreate Strategy



Terminate the old version and release the new one

Author: Nho Luong
Skill: DevOps Engineer Lead

Deployment Strategies: Blue/Green



- Run two complete deployments of your application
- “green” version of the application is deployed alongside the “blue” version

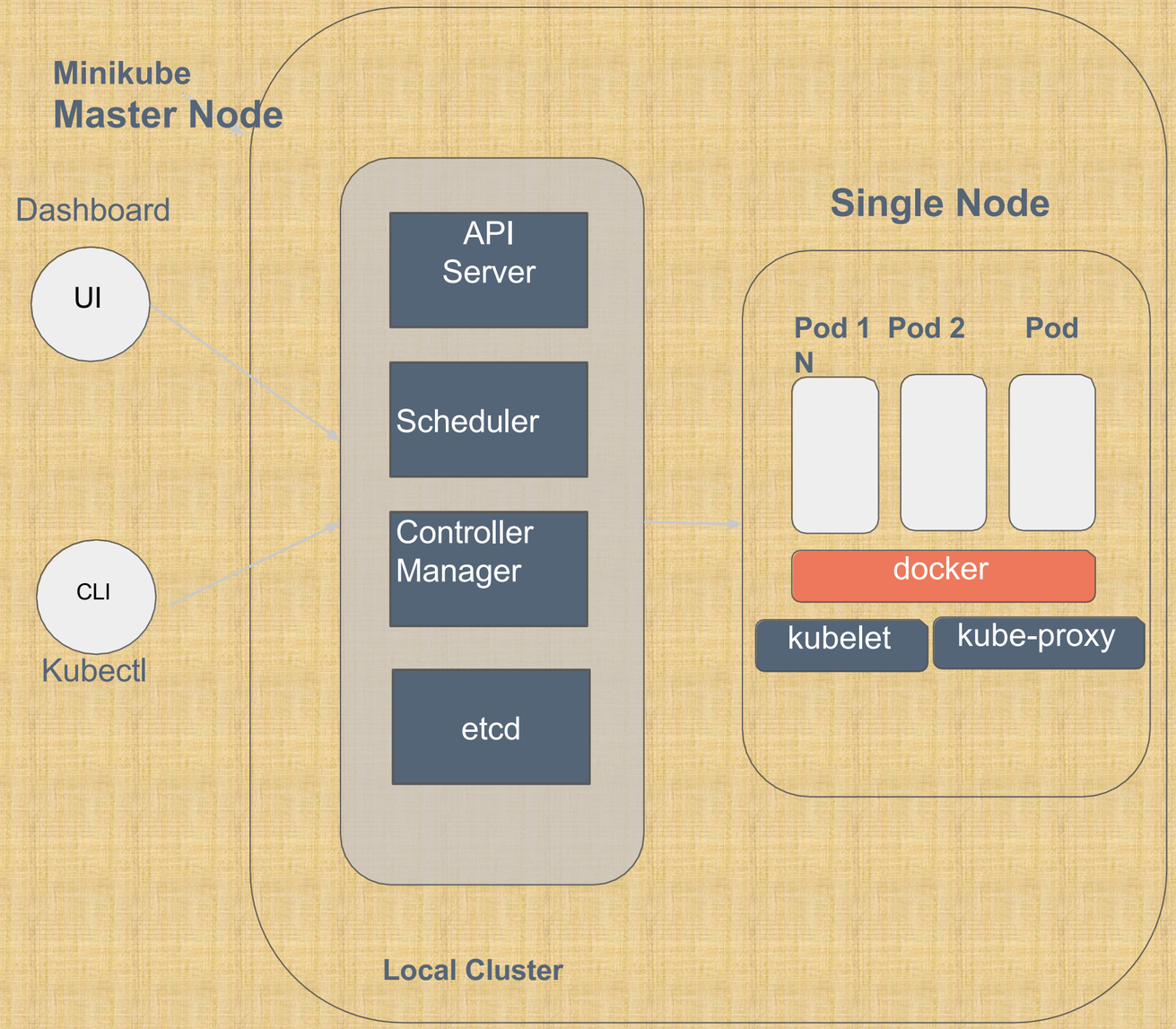
Author: Nho Luong
Skill: DevOps Engineer Lead

Deployment Strategies: Blue/Green (example)

```yaml
apiVersion: v1
kind: Service
metadata:
  name: teapot
spec:
  type: ClusterIP
  selector:
    app: teapot
    deployment: ${DEPLOYMENT}
  ports:
  - containerPort: 80
    port: 80
```

Author: Nho Luong
Skill: DevOps Engineer Lead

Minikube: K8s cluster on Local



Author: Nho Luong
Skill: DevOps Engineer Lead

Minikube: Prerequisites

- VM Driver
  - **macOS**: VirtualBox, VMware Fusion, HyperKit
  - **Linux**: VirtualBox, KVM
  - **Windows**: VirtualBox, Hyper-V
- Install Minikube
  - brew cask install minikube (macOS)
- Install kubectl
- VT-x/AMD-v virtualization must be enabled in BIOS
- Internet connection on first run

Few handy commands:

- minikube start (pass VM option), default is VirtualBox
- minikube addons enable ingress

Author: Nho Luong
Skill: DevOps Engineer Lead

Minikube: K8s Cluster

```
As-MacBook-Pro:openvpn sapna$ kubectl config get-contexts
CURRENT   NAME
          connectors-aks-cluster-admin
          ds-governance-aks-cluster-admin
          ds-management-aks-cluster-admin
          ds-taxonomy-aks-cluster-admin
          ds-taxonomy-prd-aks-cluster-admin
          ds-visualization-aks-cluster-admin
          enablers-aks-cluster-admin
          ent-integration-aks-cluster
          ent-integration-aks-cluster-admin
          gdp-brpint-aks-cluster-admin
          gdpclient-aks-cluster
          gdpclient-aks-cluster-admin
          gdpintegration1-aks-cluster-admin
          gke_civic-ripsaw-176319_us-central1-f_dev
          gke_civic-ripsaw-176319_us-central1-f_preprod
          gke_civic-ripsaw-176319_us-central1-f_prod
          gke_civic-ripsaw-176319_us-central1-f_test
*         gke_devops-230413_us-central1-a_demo-poc
          minikube
          tvstack-acs-rgmgmt
```

Hurrah! Now we have K8s cluster running named as minikube

- kubectl config get-contexts

Author: Nho Luong
Skill: DevOps Engineer Lead

Minikube: K8s Cluster

```
As-MacBook-Pro:openvpn sapna$ kubectl config use-context minikube
Switched to context "minikube".
As-MacBook-Pro:openvpn sapna$ k config get-contexts
CURRENT   NAME                                          CLUSTER
          connectors-aks-cluster-admin                  connectors-a
          ds-governance-aks-cluster-admin               ds-governanc
          ds-management-aks-cluster-admin               ds-managemer
          ds-taxonomy-aks-cluster-admin                 ds-taxonomy-
          ds-taxonomy-prd-aks-cluster-admin             ds-taxonomy-
          ds-visualization-aks-cluster-admin            ds-visualiza
          enablers-aks-cluster-admin                    enablers-aks
          ent-integration-aks-cluster                   ent-integrat
          ent-integration-aks-cluster-admin             ent-integrat
          gdp-brpint-aks-cluster-admin                  gdp-brpint-a
          gdpclient-aks-cluster                         gdpclient-ak
          gdpclient-aks-cluster-admin                   gdpclient-ak
          gdpintegration1-aks-cluster-admin             gdpintegrati
          gke_civic-ripsaw-176319_us-central1-f_dev     gke_civic-ri
          gke_civic-ripsaw-176319_us-central1-f_preprod gke_civic-ri
          gke_civic-ripsaw-176319_us-central1-f_prod    gke_civic-ri
          gke_civic-ripsaw-176319_us-central1-f_test    gke_civic-ri
          gke_devops-230413_us-central1-a_demo-poc      gke_devops-2
*         minikube                                      minikube
          tvstack-acs-rgmgmt                            tvstack-acs-
As-MacBook-Pro:openvpn sapna$
```

Change context to minikube

- kubectl config use-context minikube
- kubectl config get-contexts

Author: Nho Luong
Skill: DevOps Engineer Lead

Minikube: Deployment (Markets Service)

markets-deployment.yaml

```yaml
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: markets
spec:
  replicas: 2
  template:
    metadata:
      labels:
        app: demo-markets
    spec:
      containers:
      - name: demo-markets
        image: nodexperts/demo-markets:v1.4.0
        imagePullPolicy: Always
        env:
        - name: PORT
          value: "10000"
        - name: API_PREFIX
          value: "/markets/api"
        ports:
        - containerPort: 10000
        readinessProbe:
          httpGet:
            port: 10000
            path: /markets/api/health-check
          initialDelaySeconds: 1
          periodSeconds: 5
          timeoutSeconds: 4
          successThreshold: 2
          failureThreshold: 3
```

Container Port is the port that the application is running on.

Readiness probes: when a Container is ready to start accepting traffic.

A Pod is considered ready when all of its Containers are ready

Author: Nho Luong
Skill: DevOps Engineer Lead

Minikube: Deployment (Products Service)

products-deployment.yaml

```yaml
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: products
spec:
  replicas: 4
  template:
    metadata:
      labels:
        app: demo-products
        version: "1.0"
    spec:
      containers:
      - name: demo-products
        image: nodexperts/demo-products:v1.4.0
        imagePullPolicy: Always
        env:
        - name: PORT
          value: "9000"
        - name: API_PREFIX
          value: "/products/api"
        ports:
        - containerPort: 9000
        readinessProbe:
          httpGet:
            port: 9000
            path: /products/api/health-check
          initialDelaySeconds: 1
          periodSeconds: 5
          timeoutSeconds: 4
          successThreshold: 2
          failureThreshold: 3
```

- A Deployment named demo-product is created
- four replicated Pods
- Pods are labeled as demo-products
- Image from docker
- Environment variables

Author: Nho Luong
Skill: DevOps Engineer Lead

Minikube: Deployment (Users Service)

users-deployment.yaml

```yaml
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: users
spec:
  replicas: 2
  template:
    metadata:
      labels:
        app: demo-users
    spec:
      containers:
      - name: demo-users
        image: nodexperts/demo-users:v1.0.0
        imagePullPolicy: Always
        env:
        - name: PORT
          value: "11000"
        - name: API_PREFIX
          value: "/users/api"
        ports:
        - containerPort: 11000
        readinessProbe:
          httpGet:
            port: 11000
            path: /users/api/health-check
          initialDelaySeconds: 1
          periodSeconds: 5
          timeoutSeconds: 4
          successThreshold: 2
          failureThreshold: 3
```

- A Deployment named demo-users is created
- Two replicated Pods
- Pods are labeled as demo-users
- Image from docker
- Environment variables

Author: Nho Luong
Skill: DevOps Engineer Lead

Minikube: Service (Markets Service)

markets-service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: demo-markets
  labels:
    app: demo-markets
spec:
  selector:
    app: demo-markets
  ports:
  - port: 80
    protocol: TCP
    targetPort: 10000
    # nodePort: 10000
  type: NodePort
```

dress, those IPs are not exposed outside the cluster

ts demo-markets Pods

Author: Nho Luong
Skill: DevOps Engineer Lead

Minikube: Service (Products Service)

products-service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: demo-products
  labels:
    app: demo-products
spec:
  selector:
    app: demo-products
  ports:
  - port: 80
    protocol: TCP
    targetPort: 9000
    name: http
    # nodePort: 9000
  type: NodePort
```

ess, those IPs are not exposed outside the cluster

demo-products Pods

Author: Nho Luong
Skill: DevOps Engineer Lead

Minikube: Service (Users Service)

users-service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: demo-users
  labels:
    app: demo-users
spec:
  selector:
    app: demo-users
  ports:
  - port: 80
    protocol: TCP
    targetPort: 11000
    name: http
    # nodePort: 11000
  type: NodePort
```

address, those IPs are not exposed outside the cluster

s demo-users Pod

Author: Nho Luong
Skill: DevOps Engineer Lead

Minikube: Ingress

ingress.yaml

```yaml
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: demo-ingress
  annotations:
    ingress.kubernetes.io/rewrite-target: /
spec:
  rules:
  - http:
      paths:
        - path: /products/api/*
          backend:
            serviceName: demo-products
            servicePort: 80
        - path: /markets/api/*
          backend:
            serviceName: demo-markets
            servicePort: 80
        - path: /users/api/*
          backend:
            serviceName: demo-users
            servicePort: 80
```

Ingress objects are the rules that define the routes that should exist.

Annotations to configure some options depending on the Ingress controller

Target URI where the traffic must be redirected

Author: Nho Luong
Skill: DevOps Engineer Lead

Minikube: UI Deployment and Service

```yaml
  name: ui
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: demo-ui
    spec:
      containers:
      - name: demo-ui
        image: nodexperts/demo-app-ui:v1.4.0
        imagePullPolicy: Always
        env:
        - name: PORT
          value: "5000"
        # - name: REACT_APP_PRODUCTS_URL
        #   value: "http://34.96.84.42/products/api"
        # - name: REACT_APP_MARKETS_URL
        #   value: "http://34.96.84.42/markets/api"
        # - name: REACT_APP_USER_URL
        #   value: "http://34.96.84.42/users/api"
        - name: REACT_APP_PORT
          value: "5000"
        # For minikube
        - name: REACT_APP_PRODUCTS_URL
          value: "https://http://34.96.84.42/products/api"
        - name: REACT_APP_MARKETS_URL
          value: "https://192.168.99.100/markets/api"
        - name: REACT_APP_USER_URL
          value: "https://192.168.99.100/users/api"
        - name: REACT_APP_PORT
```

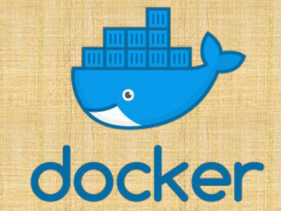
demo-app-ui.service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: demo-ui
  labels:
    app: demo-ui
spec:
  selector:
    app: demo-ui
  ports:
  - port: 80
    protocol: TCP
    targetPort: 5000
    # nodePort: 10000
  type: LoadBalancer
```

Author: Nho Luong
Skill: DevOps Engineer Lead

Clouds and Tools

We at **Successive Technologies** use following tools for building **Cloud Agnostic Platform.**



Author: Nho Luong
Skill: DevOps Engineer Lead

Thank You

Author: Nho Luong
Skill: DevOps Engineer Lead