The Art of Complex Deployments in Kubernetes

USING ARGO ROLLOUTS

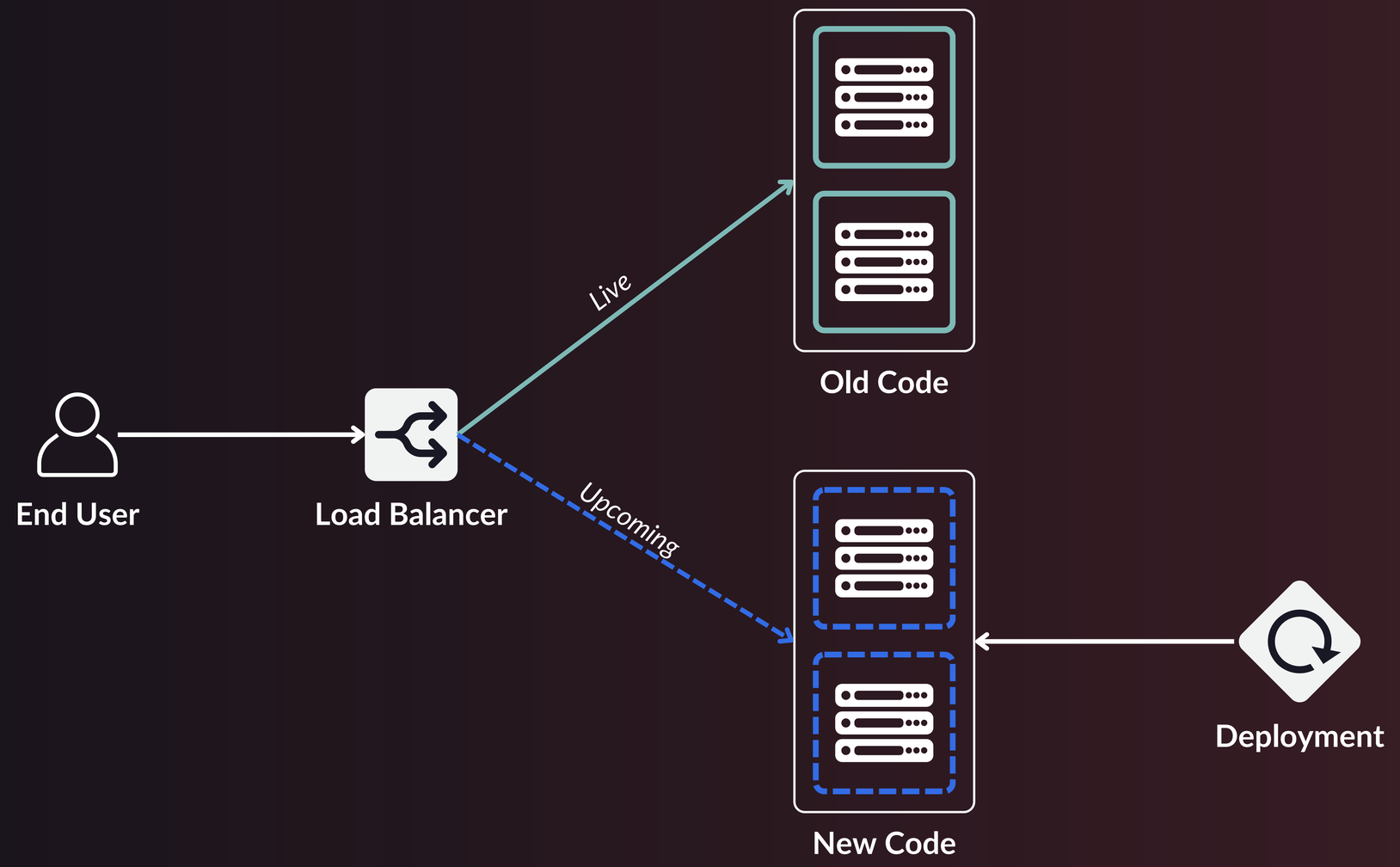
Kubernetes

Argo Rollouts

Hey! I'm Karan

- Software Engineer @ Storylane (YC S21)
- Ex SDE 2 @ HackerRank
- Domains: DevOps, Backend
- Blog: karanjagtiani.com/blog

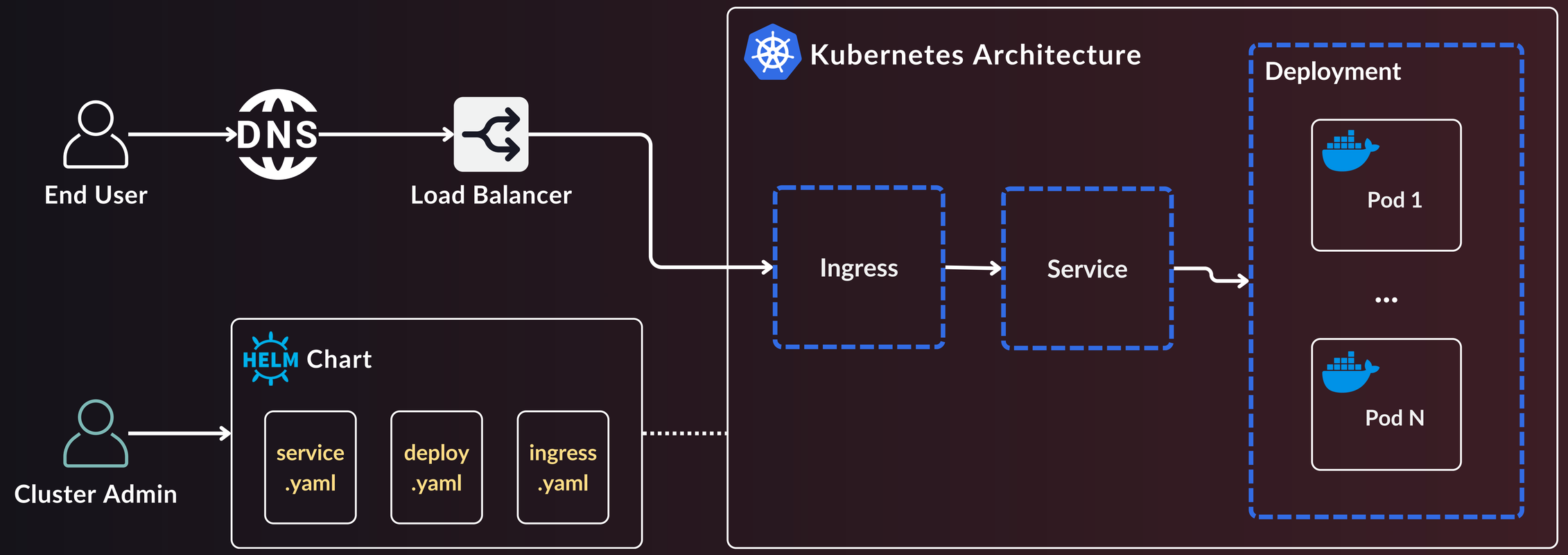
Introduction to Zero-Downtime Deployments


Live
Old Code
End User
Load Balancer
Upcoming
New Code
Deployment

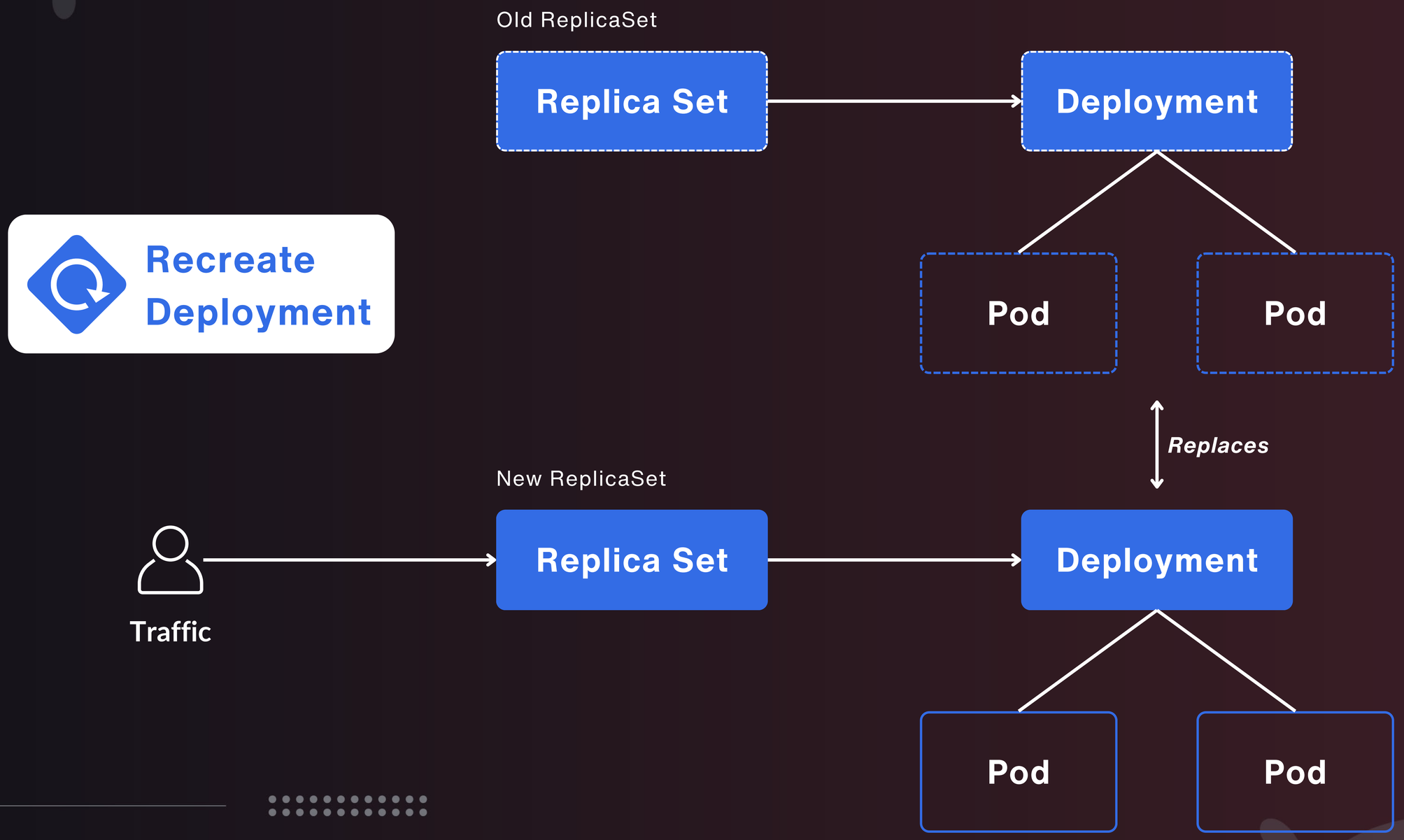
Why is Zero Downtime Necessary?

- Business Revenue
- User Experience
- Customer Trust
- SLA Obligations

Basic Kubernetes Architecture


End User
DNS
Load Balancer
Kubernetes Architecture
Ingress
Service
Deployment
Pod 1
...
Pod N
Cluster Admin
HELM Chart
service .yaml
deploy .yaml
ingress .yaml

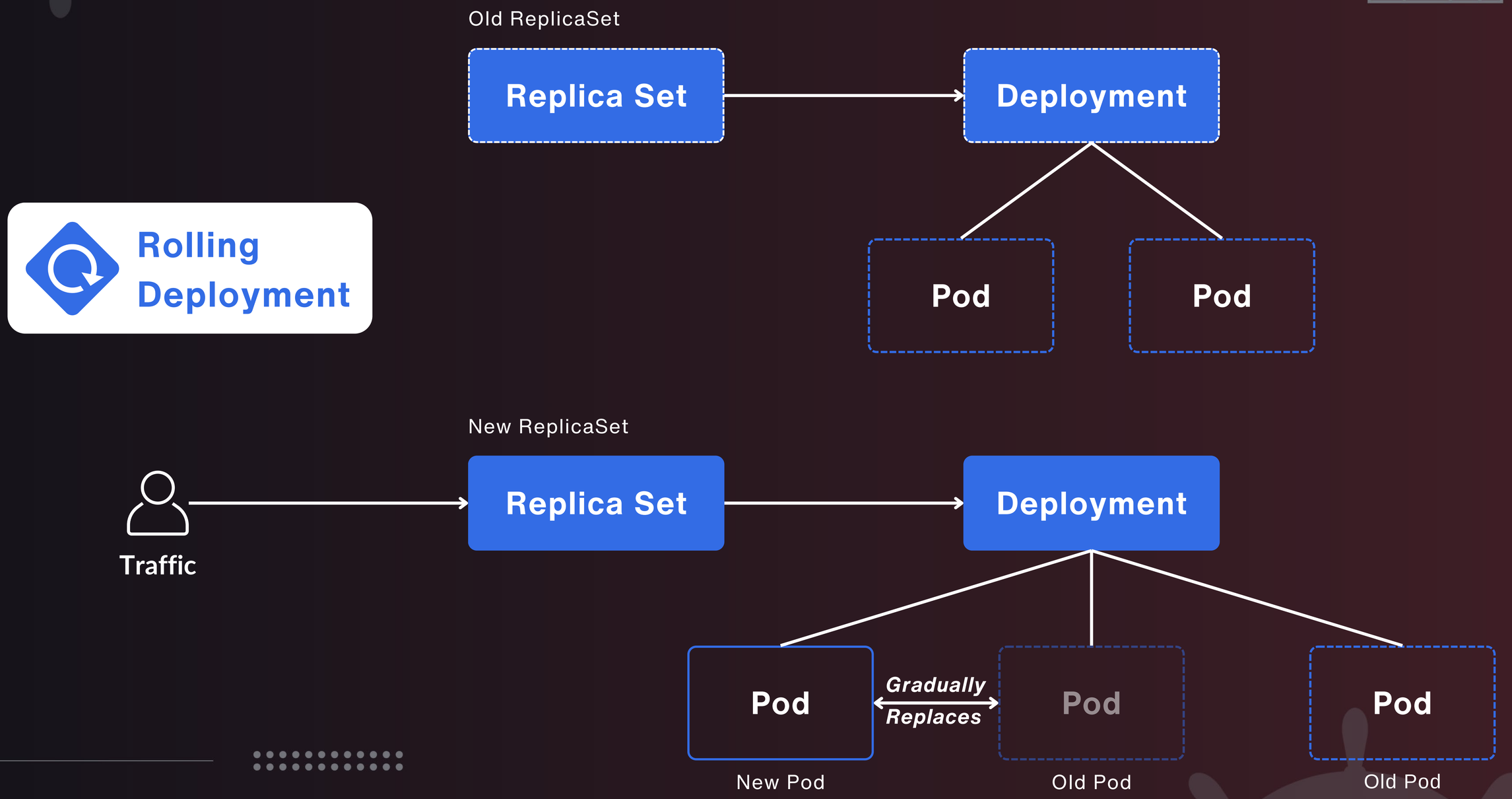
Understanding Kubernetes Deployments


Old ReplicaSet
Replica Set
Deployment
Pod
Pod
Recreate Deployment
Replaces
New ReplicaSet
Traffic
Replica Set
Deployment
Pod
Pod

Pros		Cons
Simple to understand and implement	1	Downtime while creating the new deployment
Does not require additional resources	2	Not Suitable for high-availability applications
Helps prevent potential data conflicts	3	Slower rollout process
Helpful where the architecture does not support running multiple versions	4	Potential risk during a deployment failure

Use Cases

- Non-Production Environments
- Stateful Applications with Single-Tenant Databases
- Applications with Low Availability Requirements

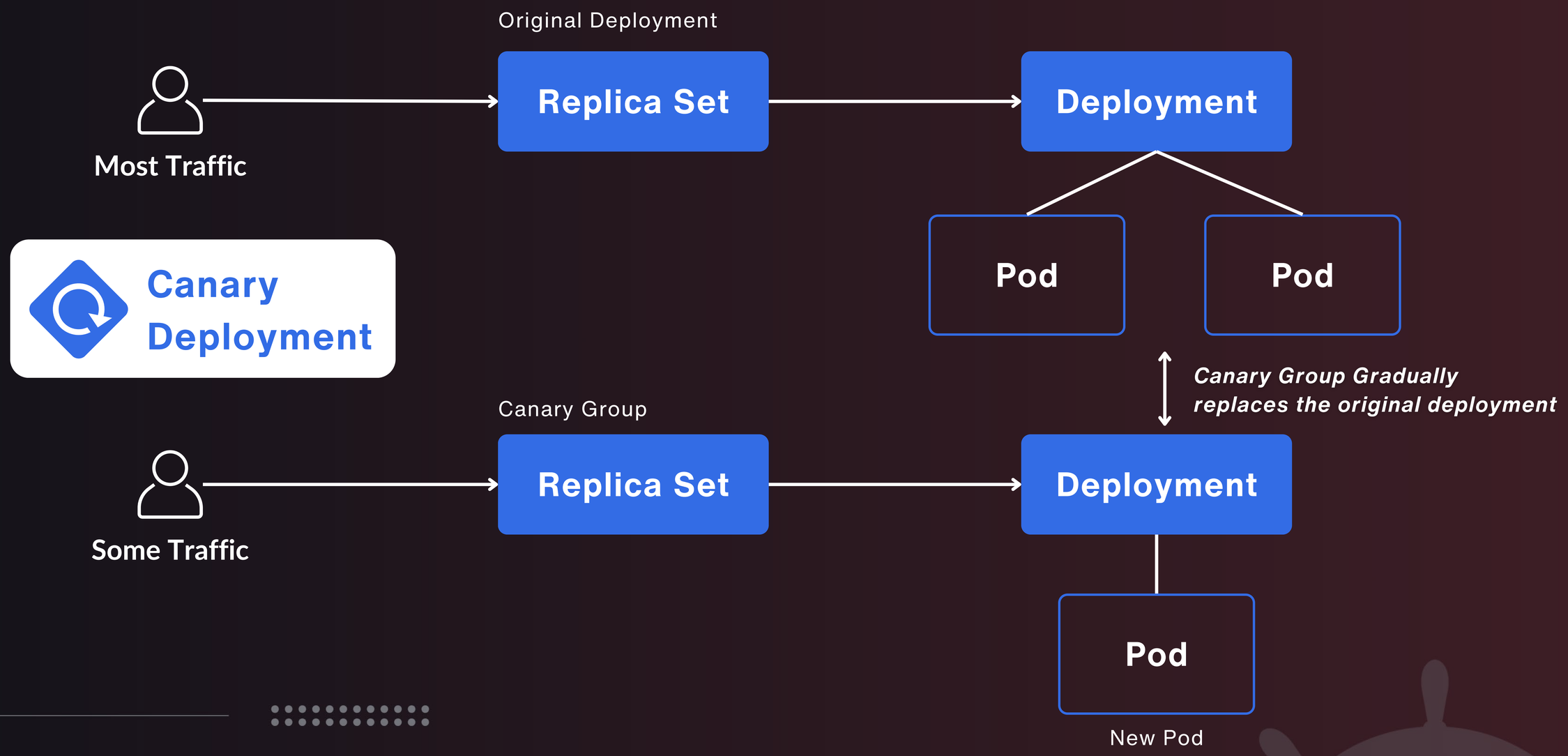

Rolling Deployment
Old ReplicaSet
Replica Set
Deployment
Pod
Pod
New ReplicaSet
Traffic
Replica Set
Deployment
Pod
Gradually Replaces
Pod
Pod
New Pod
Old Pod
Old Pod

Pros		Cons
Zero Downtime Deployments	1	Complexity in Stateful Applications
Gradual Rollout	2	Rollback Complexity
Rolling updates are resource-efficient	3	Monitoring & identifying issues is tedious
Health Checks Integration	4	Performance degradation during rollout

Use Cases

- Frequent Releases: ***dev*** or ***staging*** environments
- Scenarios requiring Gradual Rollouts
- Where maintaining multiple complete environments is costly

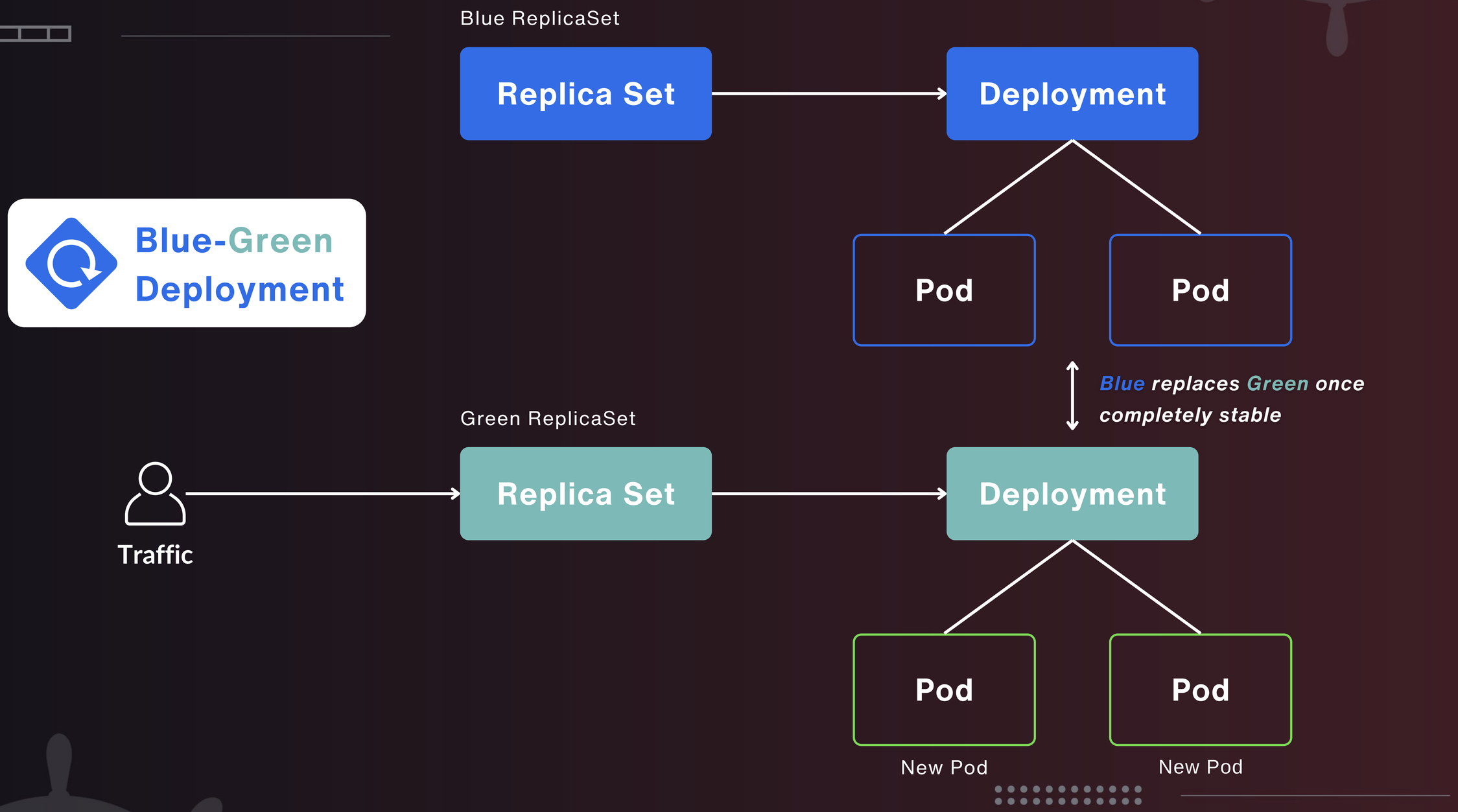
Strategies for Reliable & Zero-Downtime Deployments


Original Deployment
Most Traffic
Replica Set
Deployment
Pod
Pod
Canary Deployment
Canary Group Gradually replaces the original deployment
Canary Group
Some Traffic
Replica Set
Deployment
Pod
New Pod

Pros		Cons
Risk Mitigation	1	Complexity in Traffic Routing
Real-world Feedback	2	Monitoring Overhead
Gradual Resource Utilization	3	User Experience Inconsistency
Quick Rollback	4	Limited Testing Scope

Use Cases

- When Real-user Feedback is Critical
- In Performance-sensitive Deployments
- Continuous Deployment Environments


Blue ReplicaSet
Replica Set
Deployment
Pod
Pod
Blue-Green Deployment
Blue replaces Green once completely stable
Green ReplicaSet
Traffic
Replica Set
Deployment
Pod
Pod
New Pod
New Pod

Pros		Cons
Minimal Downtime	1	Resource Intensive
Immediate Rollback	2	Complexity in Data Management
Simplified Testing	3	Potential for Unused Resources
Load Testing and Staging	4	Configuration and Routing Complexity

Use Cases

- Critical Production Environments
- Environments requiring robust testing before release
- Highly-available services

What is Argo Rollouts?

Argo Rollouts

Blue-Green Deployment

Canary Release

Automated Rollbacks

Traffic Shifting

Demo

Best Practices for Zero-Downtime Deployments

Rigorous Testing

Real-Time Monitoring

Graceful Degradation

Traffic Control

Common Pitfalls and Challenges

- Inadequate Rollback Procedures
- Overlooking Dependency Management
- Insufficient Load Testing
- Ignoring Database Migrations
- Neglecting User Experience During Rollouts
- Incomplete Monitoring Configuration

LET'S CONNECT.

www.karanjagtiani.com

karanjagtiani04@gmail.com

/karanjagtiani

/KaranJagtiani

karan.social

argo-rollouts-demo