# Getting Started with Deploying Releases

In this lesson, we will discuss releasing features to production with zero downtime.

## Deploying Releases #

In today's competitive environment, we have to release features to production as soon as they are developed and tested. The need for frequent releases fortifies the need for zero-downtime deployments.

We learned how to deploy our applications packaged as Pods, how to scale them through ReplicaSets, and how to enable communication through Services. However, all that is useless if we cannot update those applications with new releases. That is where Kubernetes Deployments come in handy.

The desired state of our applications is changing all the time. The most common reasons for new states are new releases. The process is relatively simple. We make a change and commit it to a code repository. We build it, and we test it. Once we're confident that it works as expected, we deploy it to a cluster.

It does not matter whether that deployment is to a development, test, staging, or production environment. We need to deploy a new release to a cluster, even when that is a single-node Kubernetes running on a laptop. No matter how many environments we have, the process should always be the same or, at least, as similar as possible.

# Why Zero Downtime? #

The deployment must produce no downtime. It does not matter whether it is performed on a testing or a production cluster. Interrupting consumers is disruptive, and that leads to loss of money and confidence in a product.

Gone are the days when users did not care if an application sometimes did not work. There are so many competitors out there that a single bad experience might lead users to another solution. With today's scale, 0.1% of failed requests is considered disastrous.

While we might never be able to reach 100% availability, we should certainly not cause downtime ourselves and must minimize other factors that could cause downtime.

# Kubernetes Deployments #

Kubernetes Deployments provide us with the tools we need to avoid such failures by allowing us to update our applications without downtime.

Let's explore how Kubernetes Deployments work and the benefits we gain by adopting them.

> **i** All the commands from this chapter are available in the 06-deploy.sh Gist.

# Creating A Cluster #

Creating a cluster at the beginning of each chapter allows us to jump into any part of the course without worrying whether there is a requirement to meet from previous chapters. It also allows us to pause between chapters without stressing our laptops by running a VM that is not in use.

Let's get it over with.

```
cd k8s-specs
git pull
minikube start --vm-driver=virtualbox
kubectl config current-context
```

The code was updated, the cluster is up-and-running, and we can start exploring Deployments.

In the next lesson, we will explore deploying new releases of applications.