Rolling Back Canary Deployments

This lesson discusses how rolling back occurs in canary deployments.

Here's the good news. **Flagger** will automatically rollback if any of the metrics we set fails the number of times set as the `threshold` configuration option. Similarly, it will also roll back if there are no metrics.

In our case, there are at least three ways we can run an example that would result in a rollback.

1. We could start sending requests that would generate errors until we reach the `threshold` of the `request-success-rate` metric.
2. We can create some slow requests so that `request-duration` is messed up (above `500` milliseconds).
3. We could not send any request and force **Flagger** to interpret a lack of metrics as an issue.

We won't do a practical exercise that would demonstrate canary deployment rollbacks. To begin with, the quickstart application we're using does not have a mechanism to produce slow responses or "fake" errors. Also, such an exercise would be mostly a repetition of the practices we already explored. Finally, the last reason for avoiding rollbacks lies in the scope of the subject. Canary deployments with **Istio**, **Flagger**, **Prometheus**, and a few other tools are a huge subject that deserves more space than a part of a chapter. It is a worthy subject though and I will most likely release a set of articles, a book, or a course that would dive deeper into those tools and the process of canary deployments. Stay tuned.

So, you'll have to trust me when I say that if any metrics defined in the canary resource fail a few times, the result would be a rollback to the previous version.

Finally, I will not show you how to implement canary deployments in production. All you have to do is follow the same logic as the one we used with *jx-progressive* in the staging environment. All you'd need to do is set the `jx-progressive.canary.enable` variable to `true` inside the `values.yaml` file in the production environment repository. The production host is already set in the chart itself.

In the next lesson, we will evaluate if the canary deployments fit the needs that we defined previously.