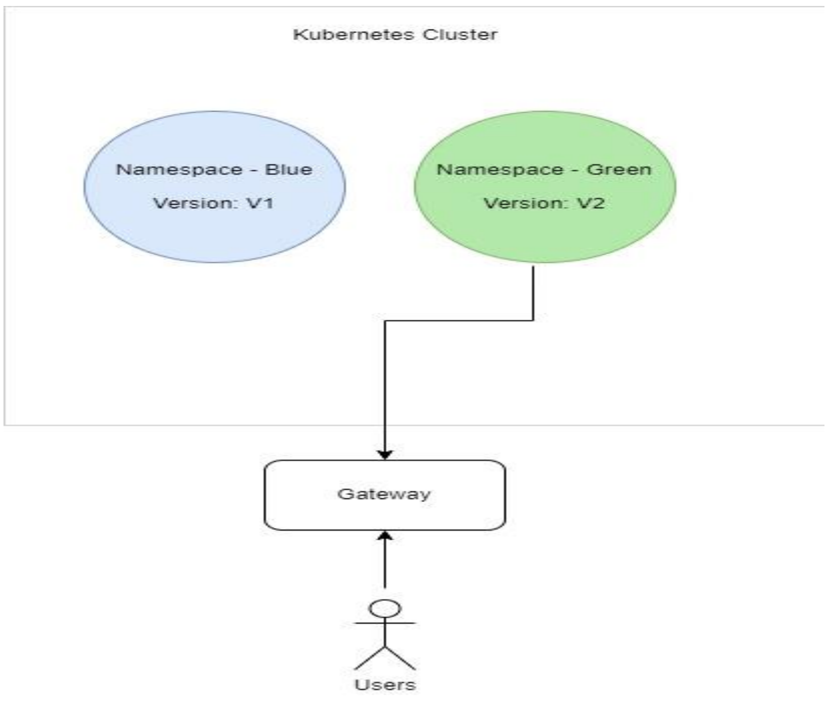
Having 1 Kubernetes Cluster and 2 copies of Namespace



For this scenario of password generator, to have a blue/green deployment I would prefer having just one cluster, but we can have 2 copies of the namespaces, one blue and the other green. In this method let's say blue namespace have the stable state or current version(V1) of service.

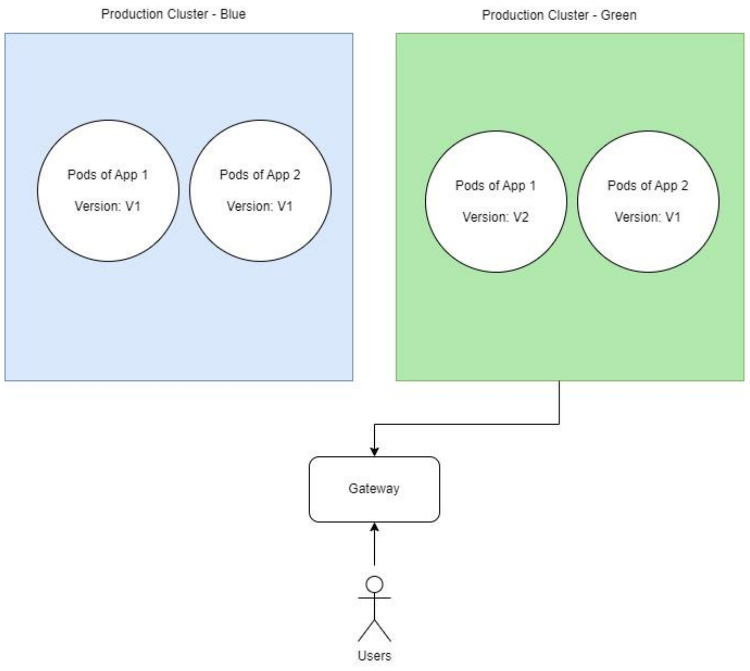
A new version(V2) of application gets ready for deployment:

1. Deploy V2 to green namespace of password-generator
2. Perform all the necessary tests on this deployment
3. Closely monitor performance and functionality to ensure the new version meets all requirements.
4. Utilize Kubernetes services or an ingress controller to control the traffic flow to the V2 pods.
5. Switch traffic to the latest pods

This way we can easily integrate the Agile methodology also along with this solution. We could automate the switch and deployment using tools like ansibe.

Having 2 separate clusters (An alternative solution)

When talking on a general basis to have an efficient setup of blue/green deployment can be designed based on the complexity of the architecture or the number of microservices. For example, consider a case when agile methodology is not required and its okay to have 2 separate clusters like a sandbox cluster which is very similar to the production cluster. In this case I would prefer naming the sandbox cluster blue and the production cluster green.



In this scenario, we have 2 Kubernetes clusters, one blue and other green. Initially let's say blue has all stable state or current version(V1) of services.

A new version(V2) gets ready for deployment:

1. Deploy V2 to cluster green
2. Cluster green can be used for Beta testing
3. With the green environment live for a subset of users or internal testing, closely monitor performance and functionality to ensure the new version meets all requirements.
4. Utilize Kubernetes services or an ingress controller to control the traffic flow between the two environments.
5. Switch traffic completely from blue to green

During the next release, the same steps can be followed to switch back to blue cluster.

Conclusion:

If the architecture has limited number of microservices and have minimal release cycle, I prefer having 2 separate clusters. This way it gives reduced downtime and easiness to rollback as well as a possibility for disaster recovery incase the cluster itself is facing issues.

On the other hand, if the company follows Agile methodology and have several microservices involved and is ready to take a risk over disaster recovery, I would suggest having a single cluster with 2 copies of namespace. This way we just need to update the service to point to the new deployments. This method helps in continuous delivery and less headache to handling the switch.