Django app on Kubernetis

You already may have containerized your application. Now it is time to scale to thousands of users. This application is built to work on following orchestration engines:

1. Kubernetes, the top open-source container orchestration platform, offers a robust and adaptable environment for handling containerized applications. It automates tasks such as scaling, load balancing, and container health checks, with a broad extension ecosystem.

2. Docker Swarm is a container orchestration solution provided by Docker. Designed to be simple to set up and use, it's a good choice for smaller applications or organizations already using Docker as it uses the same command-line interface and API as Docker.

3. OpenShift is an enterprise Kubernetes platform developed by Red Hat. It adds developer and operational tools on top of Kubernetes, simplifying the deployment and management of containerized applications.

4. Nomad, developed by HashiCorp, is a lightweight and user-friendly orchestrator capable of managing containers and non-containerized applications.

5. Apache Mesos is an open-source distributed system kernel, and DC/OS (data center operating system) is an enterprise-grade platform built on Mesos. DC/OS extends Mesos with additional features for managing and scaling containerized applications.

Software teams primarily work with managed platforms offered by well-known cloud providers such as AWS, Google Cloud Platform, and Azure. These cloud providers offer services like Amazon EKS, Google GKE, and Azure AKS, all of which are managed Kubernetes solutions. These services streamline the setup, expansion, and administration of Kubernetes clusters and seamlessly integrate with the respective cloud environments, ensuring efficient container orchestration and application deployment.

Few considerations before you get started.

Since you will have multitude of instances and each instance may require a common location to share data. For ex. you may want to share the users logged in or common configuration files etc.

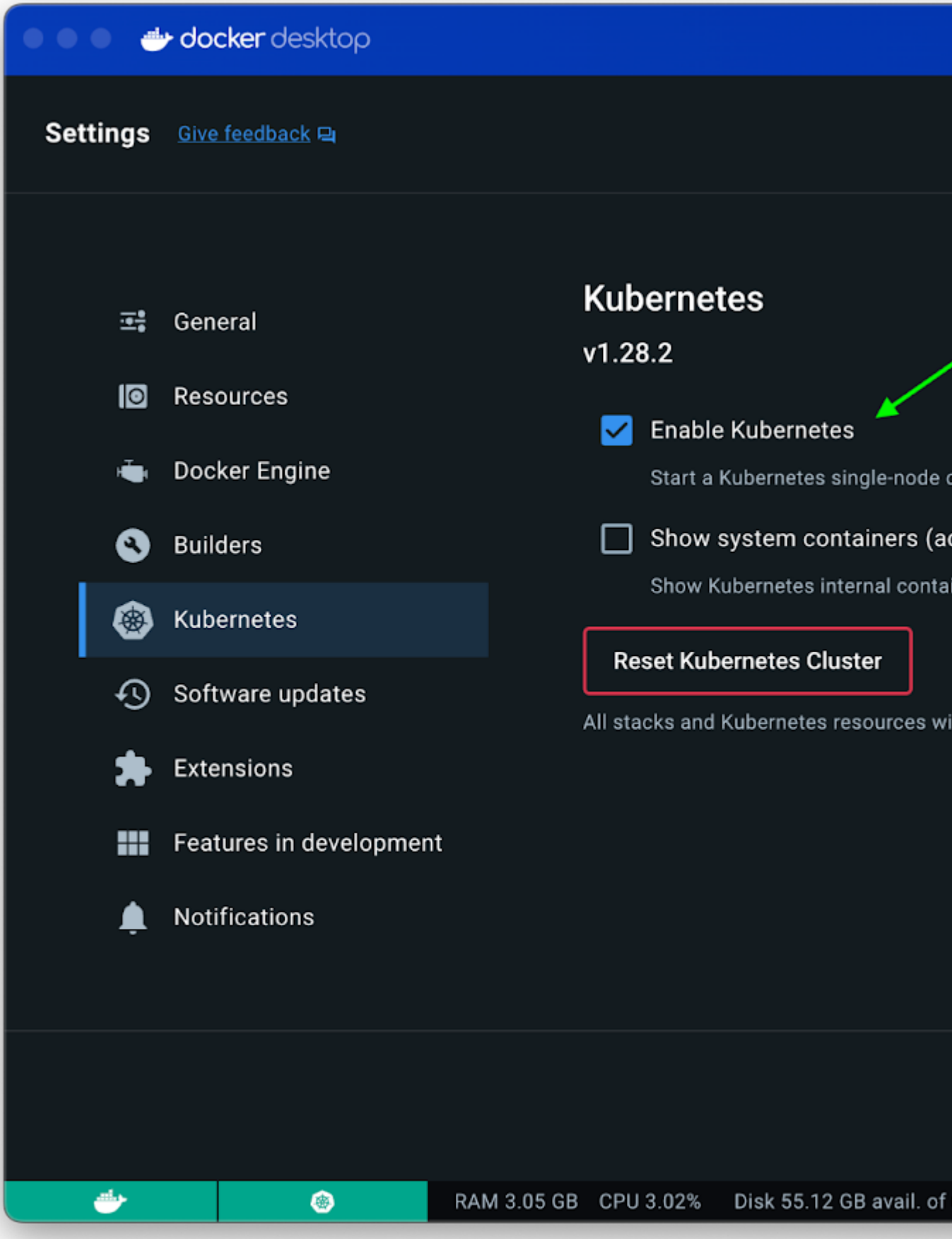
Consider:

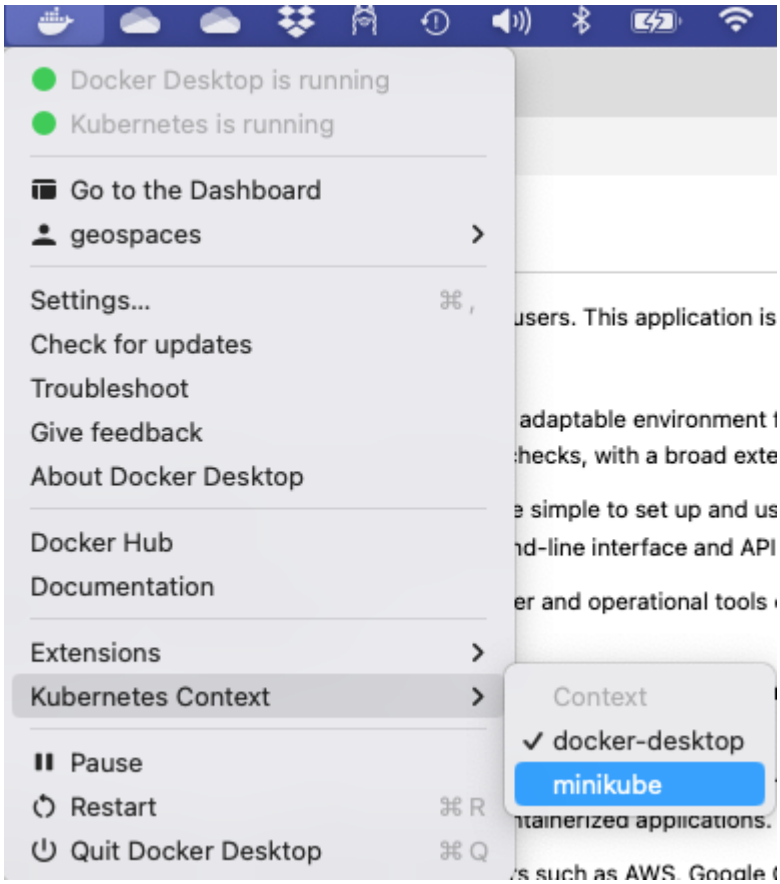
1. Using postgres for data and user management
2. Sketch common disk locations required by all instances
3. Consider how you will update all the instances instantaneously

Lets get to it.

Here:

- We will deploy the django app to Docker desktop and test it.
- Once you have undersood the mechanics, you can deploy them to any kubernetis cluster.


docker desktop
Settings
Give feedback
General
Resources
Docker Engine
Builders
Kubernetes
Software updates
Extensions
Features in development
Notifications
Kubernetes
v1.28.2
Enable Kubernetes
Start a Kubernetes single-node
Show system containers (a
Show Kubernetes internal conta
Reset Kubernetes Cluster
All stacks and Kubernetes resources wi
RAM 3.05 GB
CPU 3.02%
Disk 55.12 GB avail. of

=> 

lets create a namespace:

```sh showLineNumbers kubectl create ns django-app

OUT>> namespace/django-app created ```

# References:

1. https://blog.jetbrains.com/pycharm/2024/03/deploying-django-apps-in-kubernetes/

\newpage
```