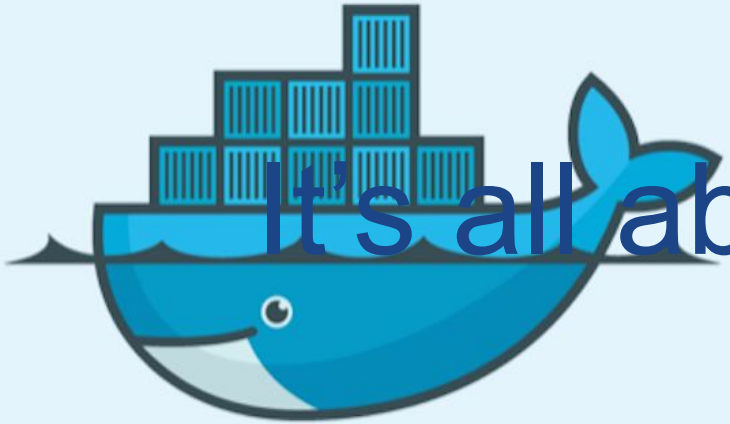


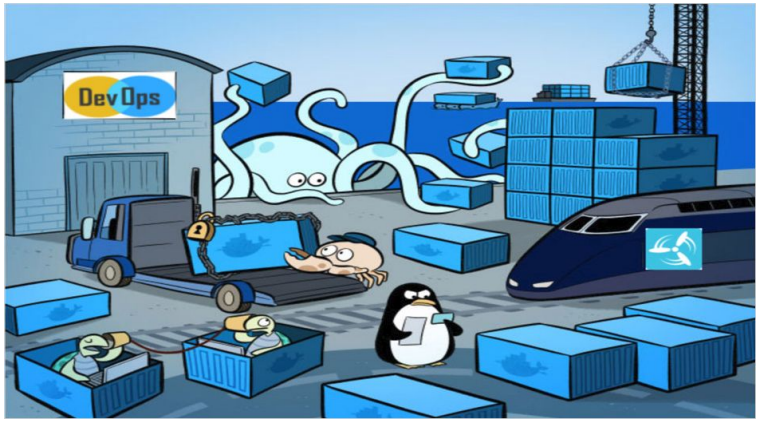
It's all about shipping!

Docker Tutorial

- What's it all about?
- Terminology & Eco-system
- Assignments: Requirements and Tips
- Example

It’s all about shipping!

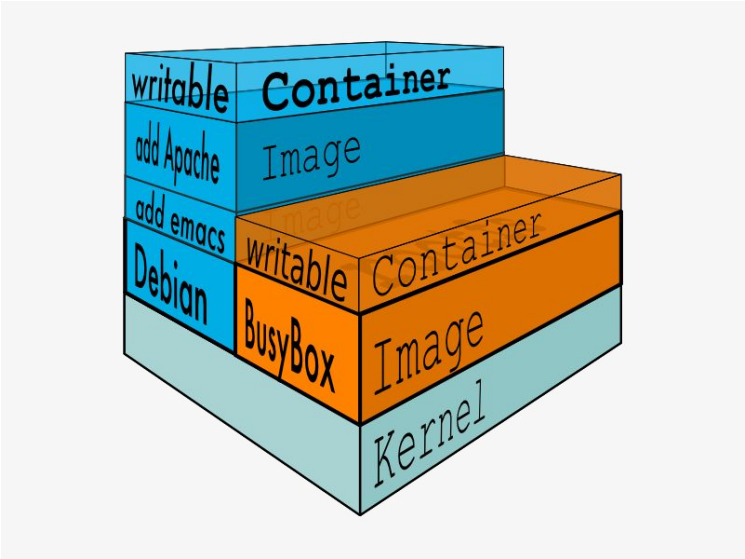
Docker is a platform for developers and sysadmins to **develop, deploy, and run** applications with containers.



The use of Linux containers to deploy applications is called *containerization*. Containers are not new, but their use for easily deploying applications is.

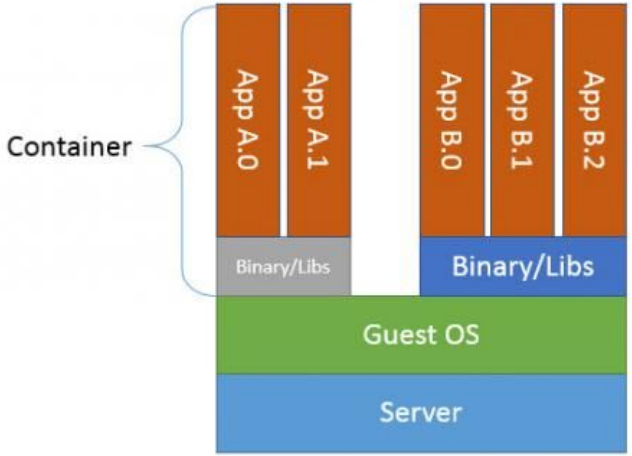
What are containers?

Containers leverage the low-level mechanics of the host operating system, to provide most of the isolation of virtual machines at a fraction of the computing power.



How do container compare to VMs?

Traditional Virtualization

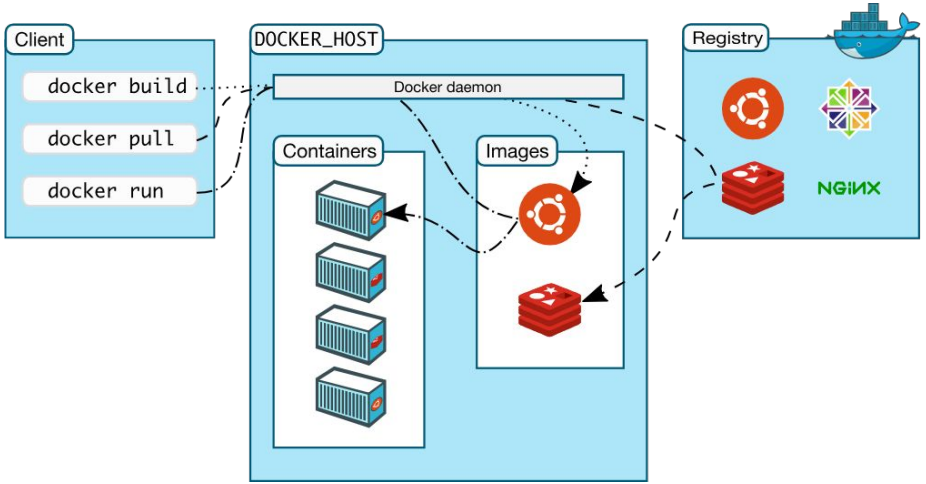


Docker

Terminology

- The **Docker daemon** is the background service running on the host (local or in a docker-machine VM) managing building, running and distributing Docker containers. (*systemctl status docker.service, docker-machine ls*)

- The **Docker client** is the command line Tool that allows the user to interact with the Docker daemon. (*docker ps, ...*)

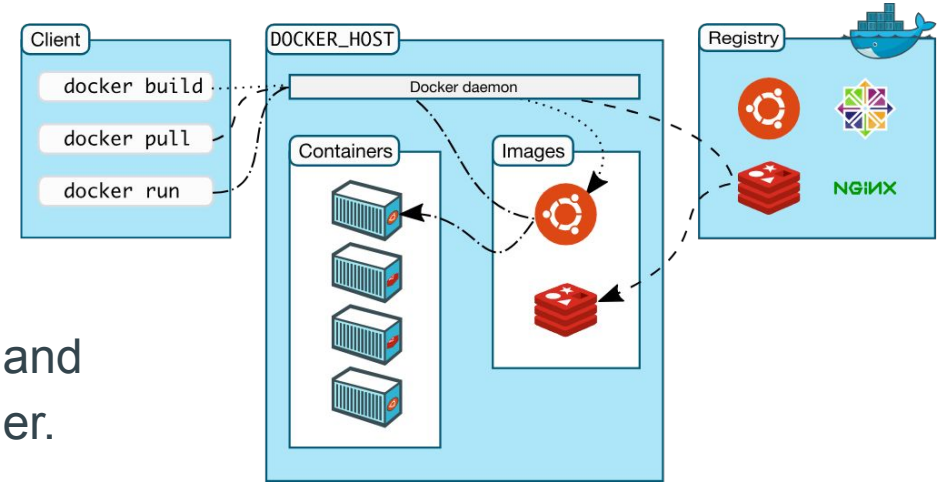


Terminology

- **Docker images** are the blueprints of the application which is used as the basis of containers. *(docker pull/docker build)*

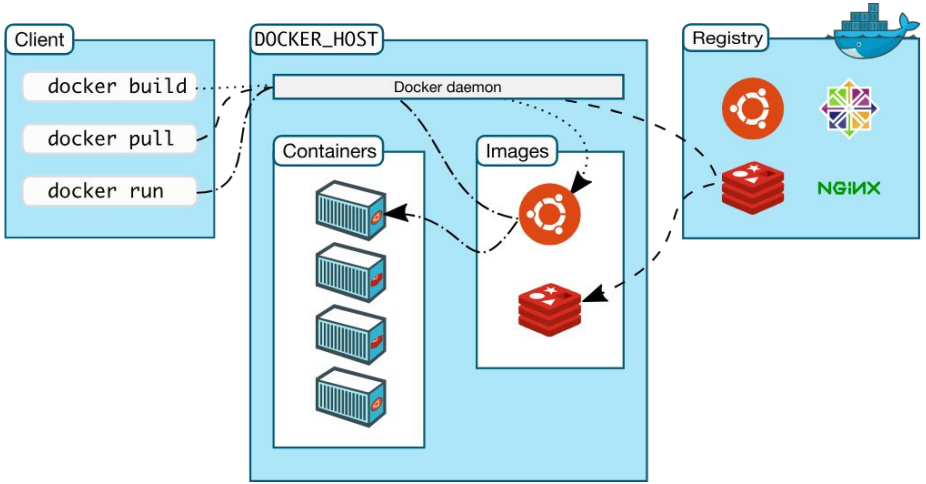
- **Docker container** are created from Docker images and run the actual application. (*docker run/docker create*)

- A **stack** is a group of interrelated services that share dependencies, and are orchestrated and scaled together.



Terminology

- The **DockerHub** a public registry of Docker images. (*https://hub.docker.com/*)
- **docker-compose** is a tool for defining and running multi-container Docker applications.
- **Docker-machine** manages per-Built VM images running the **Docker daemon**



Example 1: Setting Stuff Up

- Create Docker machine imse1
- Set imse1 a swarm manager
- Deploy Portainer stack on imse1

Example 2: Packaging

- Using **`docker build -t <name> <context>`** to package your app
- Using **`docker run [OPTINS] <name>`** to deploy your app on imse1
- Using the **`--link`** option to let container discover each other

- Use **`docker stack deploy`** to make life easier!

Assignment Requirements

- The `docker stack deploy` command **must be sufficient** to run your application:
  - Use the “build” instruction to build your image
  - Keep your network tight - expose only necessary ports

- Your work will only be graded if it is properly containerized.

- Support is provided for Linux and Docker machine. Every other installation is your business.

Docker Swarm Mode

- **Scaling:** For each service, you can declare the number of tasks you want to run. When you scale up or down, the swarm manager automatically adapts by adding or removing tasks to maintain the desired state.

- **Desired state reconciliation:** The swarm manager node constantly monitors the cluster state and reconciles any differences between the actual state and your expressed desired state.

- **Cluster management integrated with Docker Engine:** Use the Docker Engine CLI to create a swarm of Docker Engines where you can deploy application services.

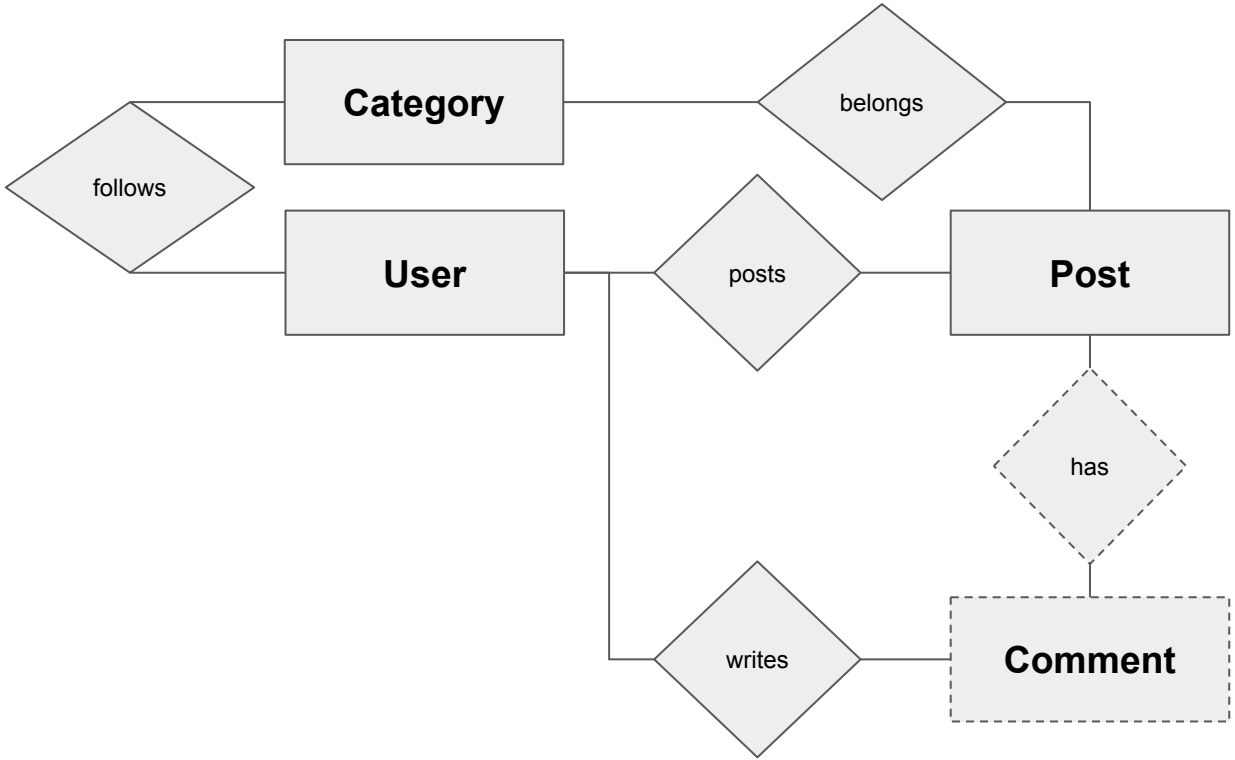
Docker Swarm Mode

- **Multi-host networking:** You can specify an overlay network for your services. The swarm manager automatically assigns addresses to the containers on the overlay network when it initializes or updates the application.
- **Service discovery:** Swarm manager nodes assign each service in the swarm a unique DNS name and load balances running containers. You can query every container running in the swarm through a DNS server embedded in the swarm.
- **Load balancing:** You can expose the ports for services to an external load balancer. Internally, the swarm lets you specify how to distribute service containers between nodes.

Example 3: Scaling your App

- Define multiple replicas of the app
- Add a second node to the cluster
- Remove second node from the cluster

Something like Reddit
Logical Model



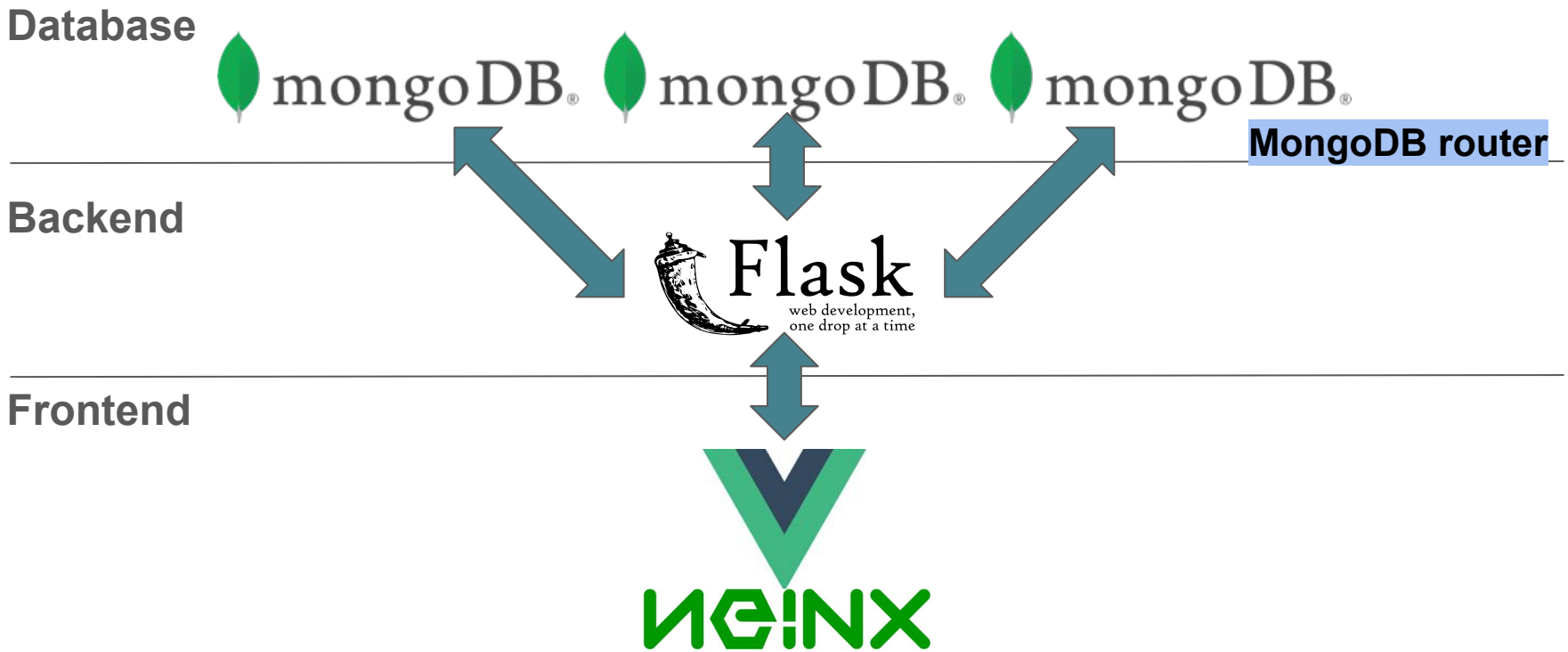
Something like Reddit Components

Database

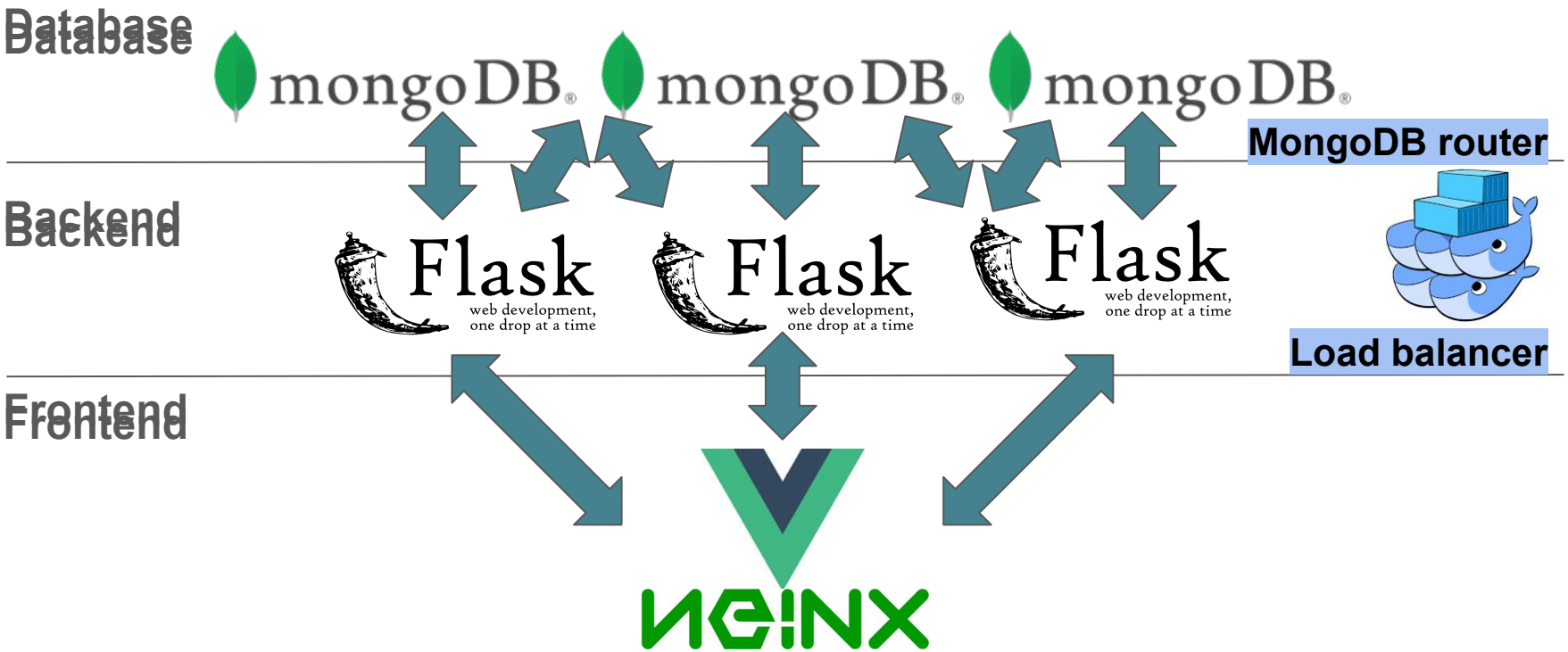
Backend

Frontend

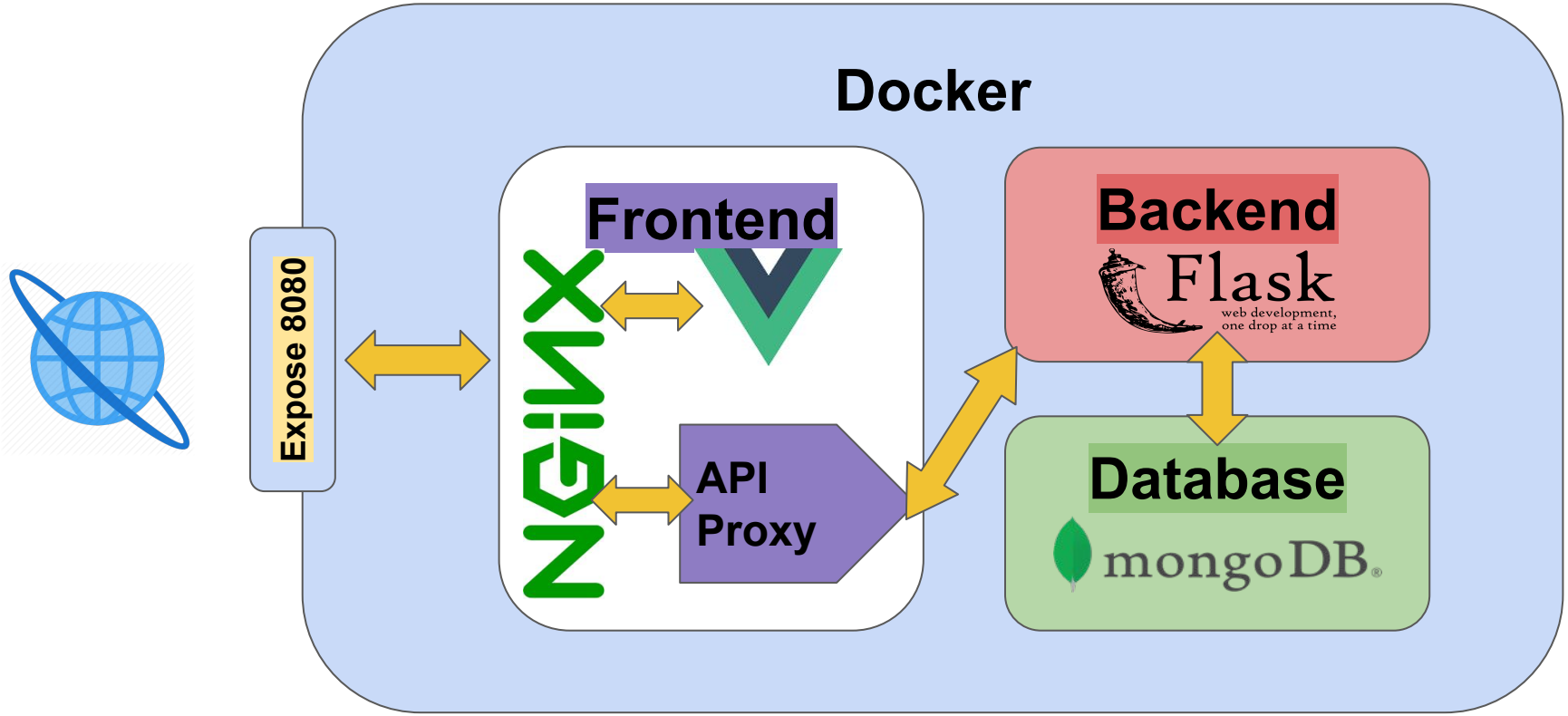
Something like Reddit
MongoDB Cluster



Something like Reddit
Scaled-out Middleware



Something like Reddit
Docker Stack Overview



Something like Reddit Implementation Details

- MongoDB Aggregation Pipelines (comment_per_user, posts_per_user)
- Using docker-compose (on local docker host) for clutter-free development
- Token authentication for REST API (Postman collection) (using TTL index for expiration and the “unlimited-admin”

Something like Reddit
Implementation Details

- Make basic functionality via the startpage of your app accessible (e.g. filler, migration, …)

- Highlight implementation details in your documentation (e.g. use of aggregation pipelines, indexing, ...)

Resources

- Have your docker machine up and running
  https://docs.docker.com/machine/

- Download tutorial examples from GitHub:
  https://github.com/vigne/imse-example

FIN.

This tutorial is based on the Docker documentation available at: https://docs.docker.com/