

A background image showing a group of people in a modern office or co-working space. They are seated at a long wooden table, each with a laptop. One person in the foreground is wearing large headphones and is seen from the back. The scene is dimly lit, with a warm, ambient glow. A dark blue diagonal overlay covers the left side of the image, where the text is placed.

**CONTAINERS**


vs.

**SERVERLESS**

from a DevOps  
standpoint

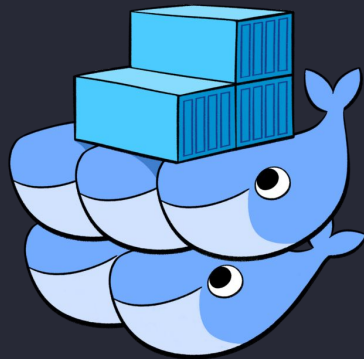
# WHO AM I?

I'm Adnan Rahić

- Developer Advocate at  dashbird
- Teaching people to code since 2016
- Tech Writer on Medium
- Author of *"Serverless JavaScript by Example"*

*( there will be a free giveaway  
at the end - stay tuned 🎁 )*





VS.

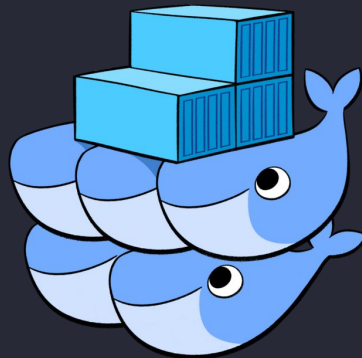


# WHAT WILL WE BE TALKING ABOUT TODAY?

**PS:** I think we all know serverless still uses servers, and no, there will be **no pictures of boats** in this presentation.

# AGENDA

- What are containers?
- Container pros/cons & use cases
- Container live DevOps session
- What is serverless?
- Serverless pros/cons & use cases
- Serverless live DevOps session

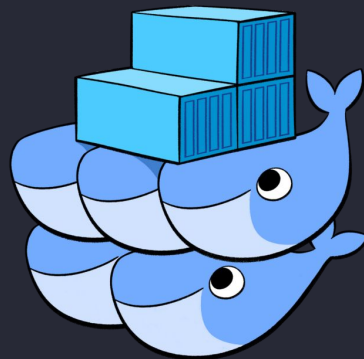


vs.

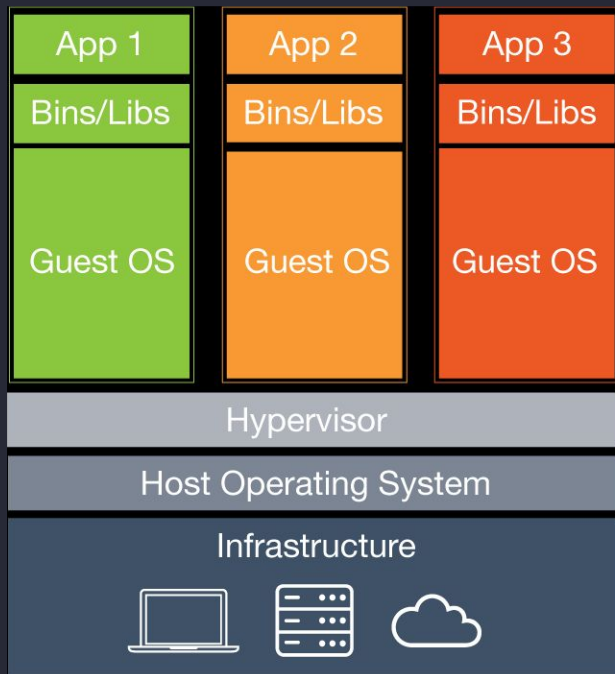


# WHAT ARE CONTAINERS?

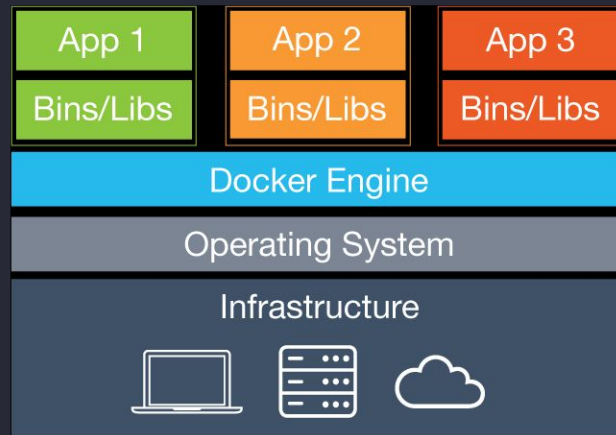
- Isolated, stateless environment
- Abstracting infrastructure
- Like a tiny VM, but not really



# WHAT ARE CONTAINERS?



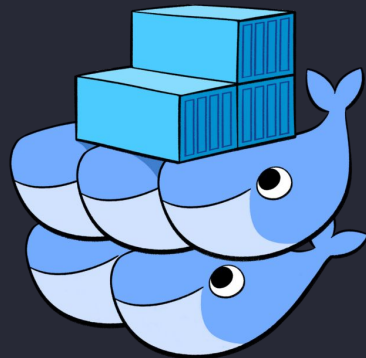
Each VM needs a separate OS



Containers share host OS kernel

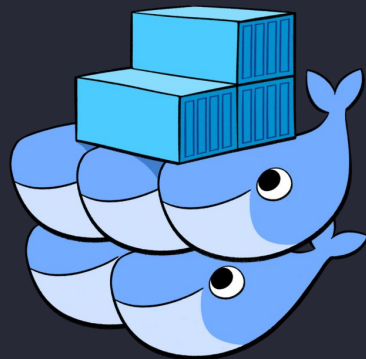
# CONTAINER CONS

- Does not auto-scale
- Harder to deploy
- Steep learning curve
- More complex with more moving parts
- More expensive



# CONTAINER PROS

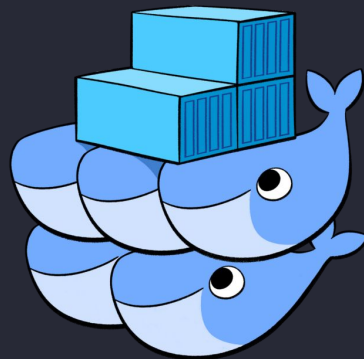
- Full control
- Almost infinite scalability
- Easy debugging/monitoring
- Easy collaboration (unified environment)





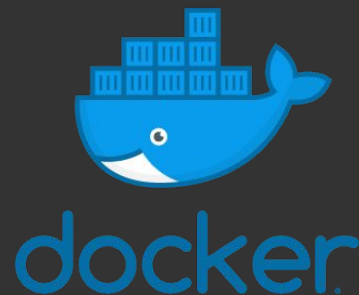
# CONTAINER USE-CASES

- Monoliths
- Microservices
- Web APIs
- Long running processes
- Data science



# LIVE DEVOPS SESSION #1!

Deploying a  
container  
to an AWS cluster  
with Docker Swarm



+



# WHAT IS SERVERLESS?

- Function as a Service (FaaS)
- Event based system for running code
- We don't care about servers - only the code
- Abstracting away the infrastructure



# SERVERLESS CONS

- Defined limits (memory, cpu, execution time)
- Risk of overloading system
- Bad monitoring/debugging solutions
- Latency



# SERVERLESS PROS

- Pay for how much you use
- Low maintenance
- Simple & Easy to deploy
- Auto-scaling
- Faster time-to-market



# SERVERLESS USE-CASES

- Web APIs
- Process data streams, images etc
- Short running processes
- Periodic processes / cron



# LIVE DEVOPS SESSION #2!

Deploying a  
serverless app  
to AWS Lambda with  
the Serverless Framework



+



# THE VERDICT!

- Containers
  - Flexibility
  - Control
  - Legacy migration
- Serverless
  - Fast deployment & development
  - Auto-scaling
  - Lower cost



# THANKS!

## Any questions?

You can find me at

- **@adnanrahic** on Medium, GitHub, Twitter, and LinkedIn!
- [adnan@dashbird.io](mailto:adnan@dashbird.io)



# RESOURCES

- Docker Swarm CloudFormation setup on AWS -  
<https://docs.docker.com/docker-for-aws/#quickstart>
- Express app with Docker repo -  
<https://github.com/adnanrahic/express-docker-app>
- Serverless setup tutorial -  
<https://dev.to/adnanrahic/how-to-deploy-a-nodejs-application-to-aws-lambda-using-serverless-2nc7>
- Express app with Serverless repo -  
<https://github.com/adnanrahic/express-sls-app>