#5 - Computing and Computing at Scale

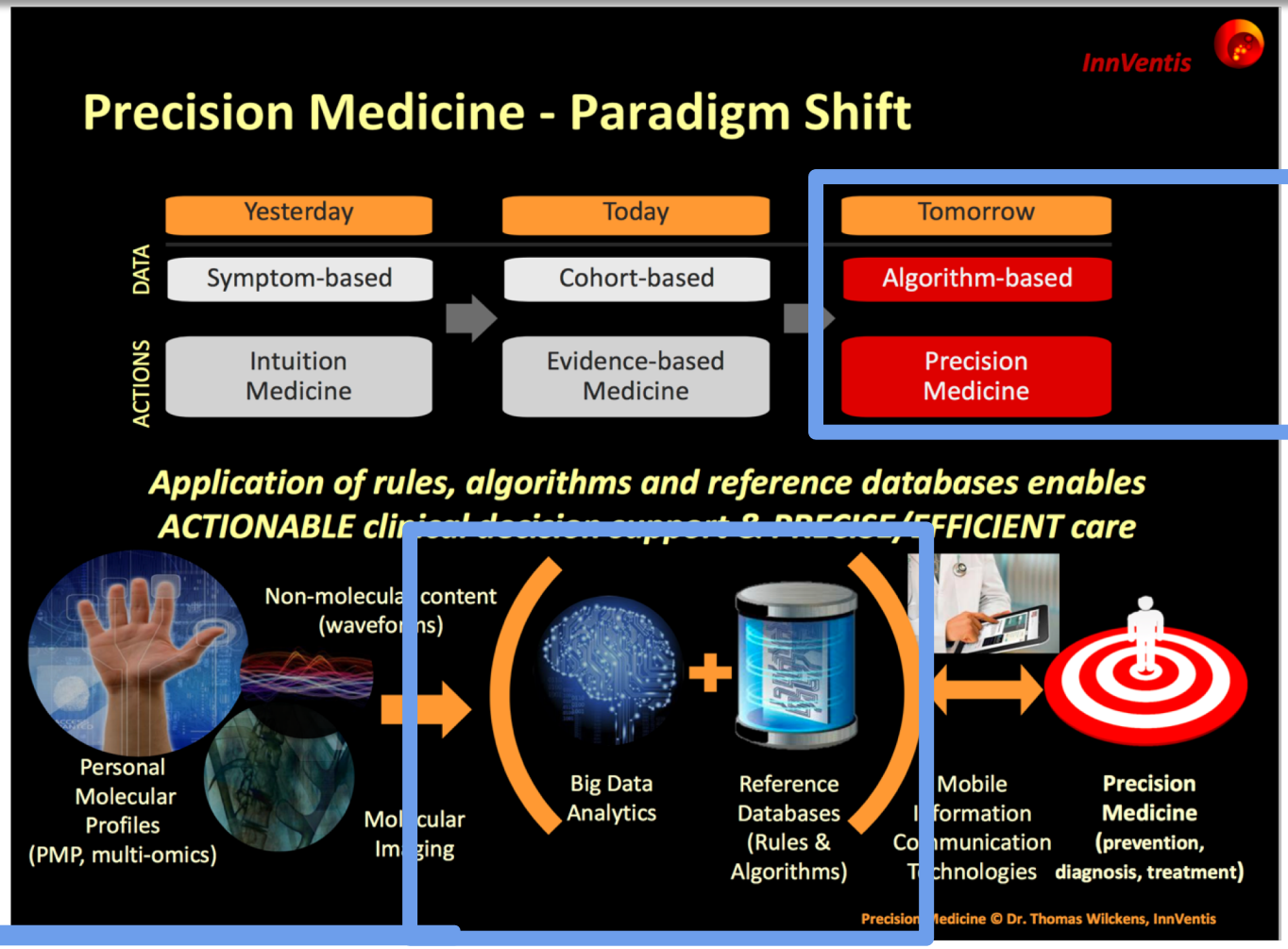
EMORY
UNIVERSITY
SCHOOL OF
MEDICINE

5min Recap

EMORY
UNIVERSITY
SCHOOL OF
MEDICINE

How do **Data Sci. & Engg.** enable Precision Medicine ?

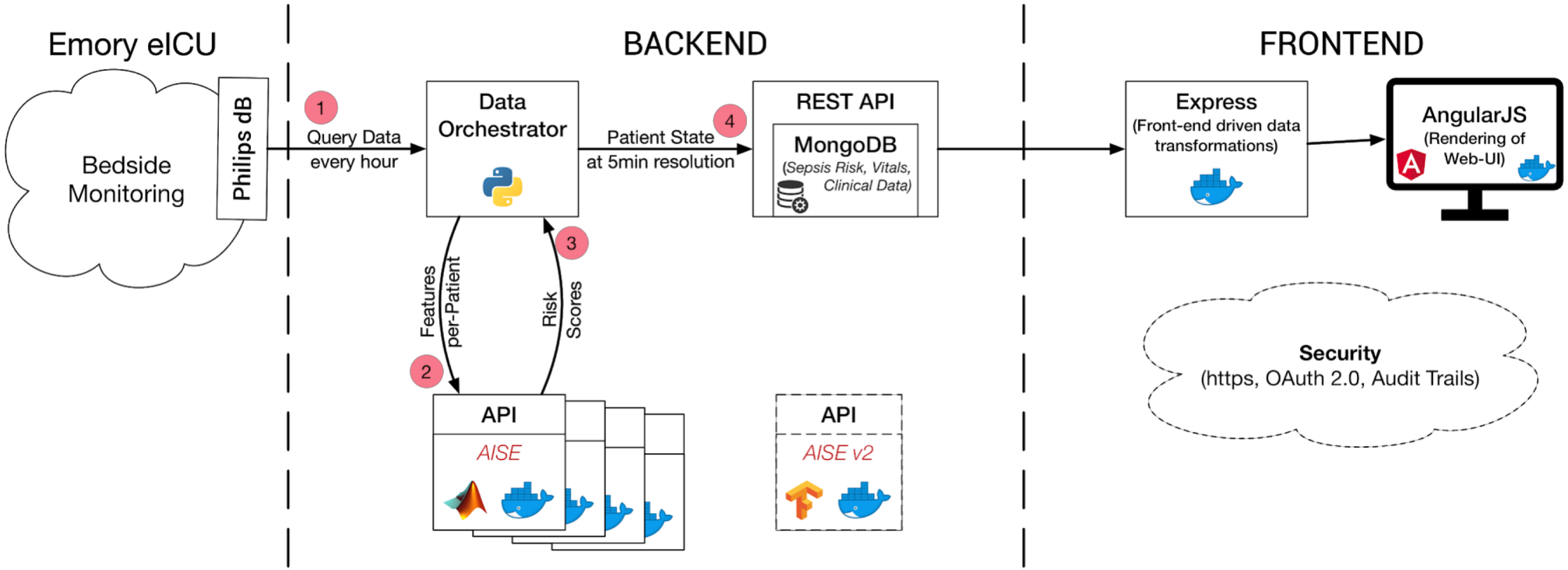
EMORY UNIVERSITY SCHOOL OF MEDICINE



Data Science

Algorithms

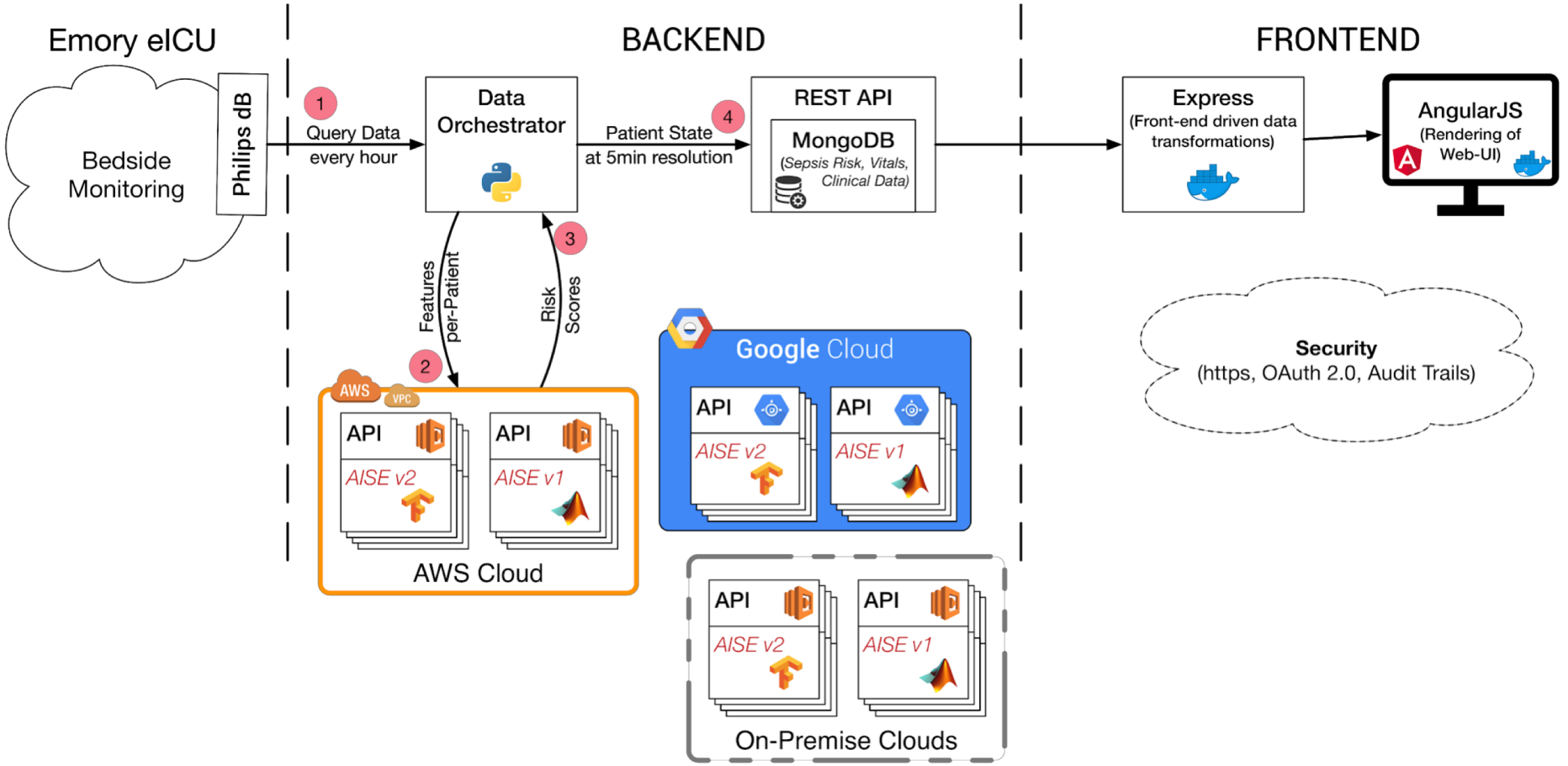
Data Engineering



- ✓ Real-time Data Streams
- ✓ Easy to test and maintain
- ✓ Easier to upgrade *AISE*
- ✓ Time series view of data simplifies UI Interactions
- ✓ Secure

- ○ AISE deployment is not elastic
- ○ No-automatic failovers
- ○ Hard to deploy multiple algorithms

Architecture of Production Deployment ver 1.0



- Depend AWS/Google Services for Scalability and Recovery from Failure
  - Deployed AISE on AWS Lambda and Google App Engine
- Rapid Deployment < 1day
- Also speeds up model development by allowing prototyping at with real-world constraints

Next Steps

Cloud Service Model

Pizza-as-a-Service

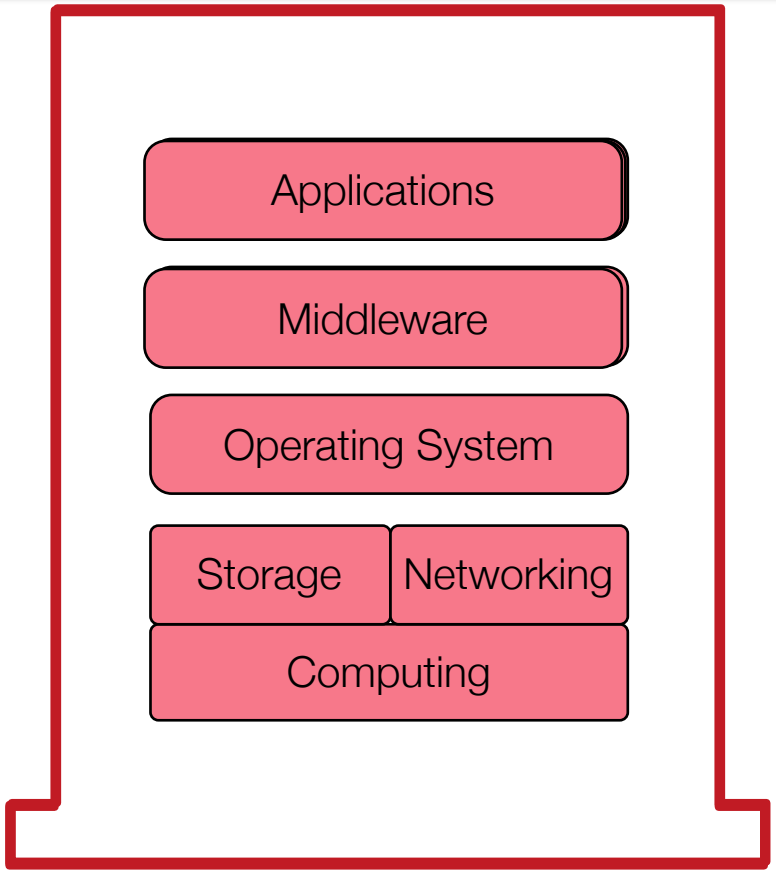
DIY vs. Take-n-Bake vs. Delivery vs. Pizzeria

On-Premises

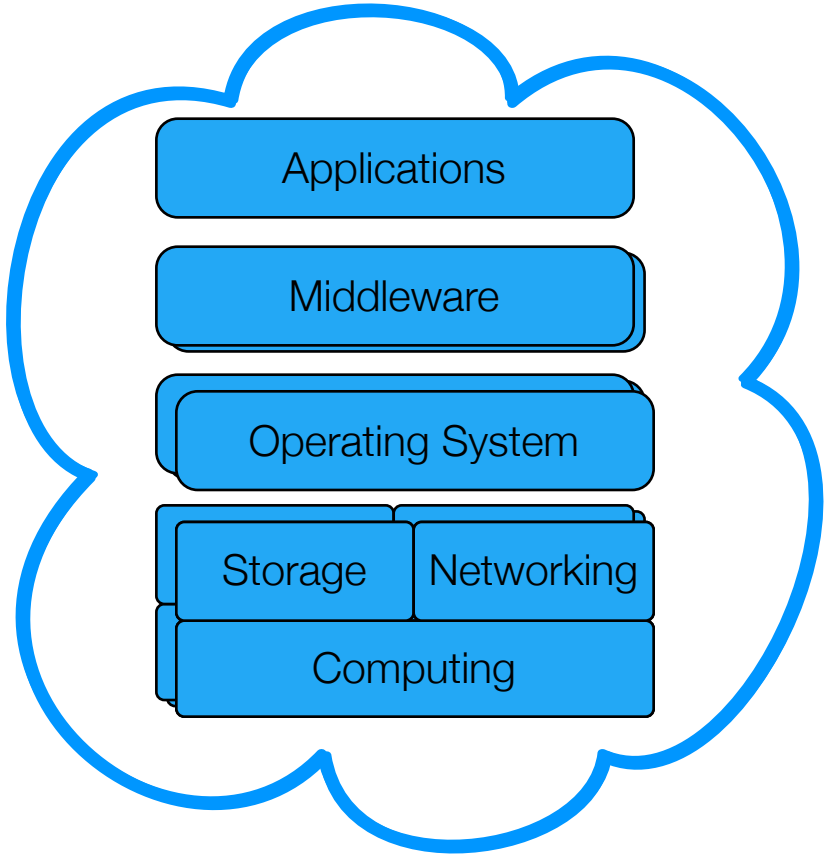
Infrastructure-as-a-Service (IaaS)

Platform-as-a-Service (PaaS)

Software-as-a-Service (SaaS)



You Manage



Cloud Based

EMORY
UNIVERSITY
SCHOOL OF
MEDICINE

Using the Cloud

(IaaS/PaaS/SaaS)

Cloud computing is a set of managed **resources** and **services**

- Computing cycles
- Storage systems of varying capability and cost

- Data management systems (*Databases*)
- ML, AI Capabilities as managed services
- Security, Networking and other System services

Pay-per-use (on-demand, shared, rapid provisioning)

Today:
Computing Resources & Systems

EMORY
UNIVERSITY
SCHOOL OF
MEDICINE

EMORY
UNIVERSITY
SCHOOL OF
MEDICINE

Cloud Computing

When does it make sense?

- Fixed Big Machines vs. On-Demand Computing
- Loads:
  - Embarrassingly parallel
  - CPU heavy or Memory heavy
  - Needs GPUs
- Things to consider (Cost)
  - Storage
  - Storage Latency
  - Transfer of data from cloud

EMORY
UNIVERSITY
SCHOOL OF
MEDICINE

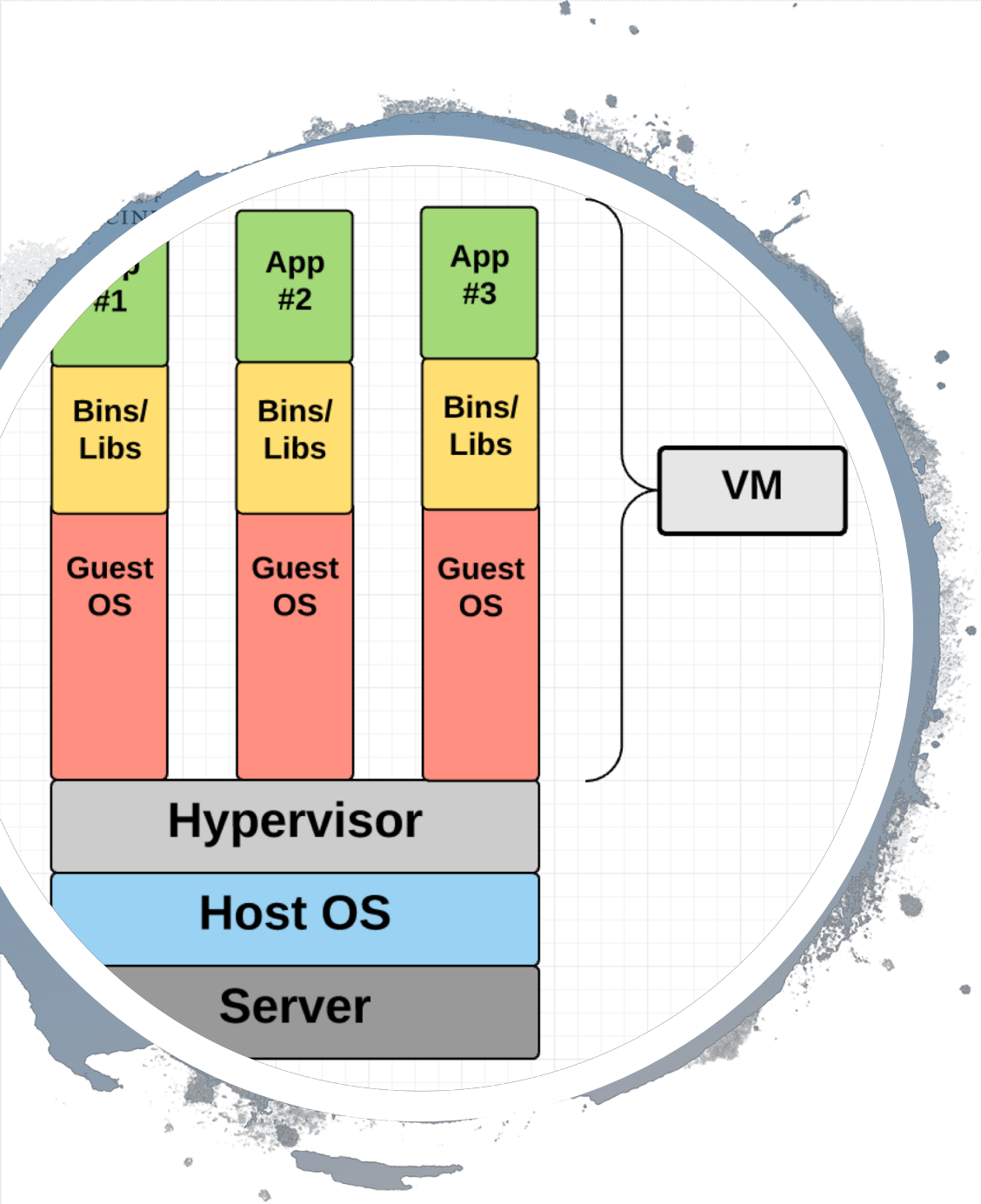
Today's Outline

Virtual Machines

Containerization

Serverless

Virtualization



Regular Machines: Multiple programs share HW resources. Operating System (OS) manages HW resources amongst programs.

Virtual Machine: A Hypervisor helps share resources between machines (VMs)

VMs → Share hardware

Each VM → Own OS

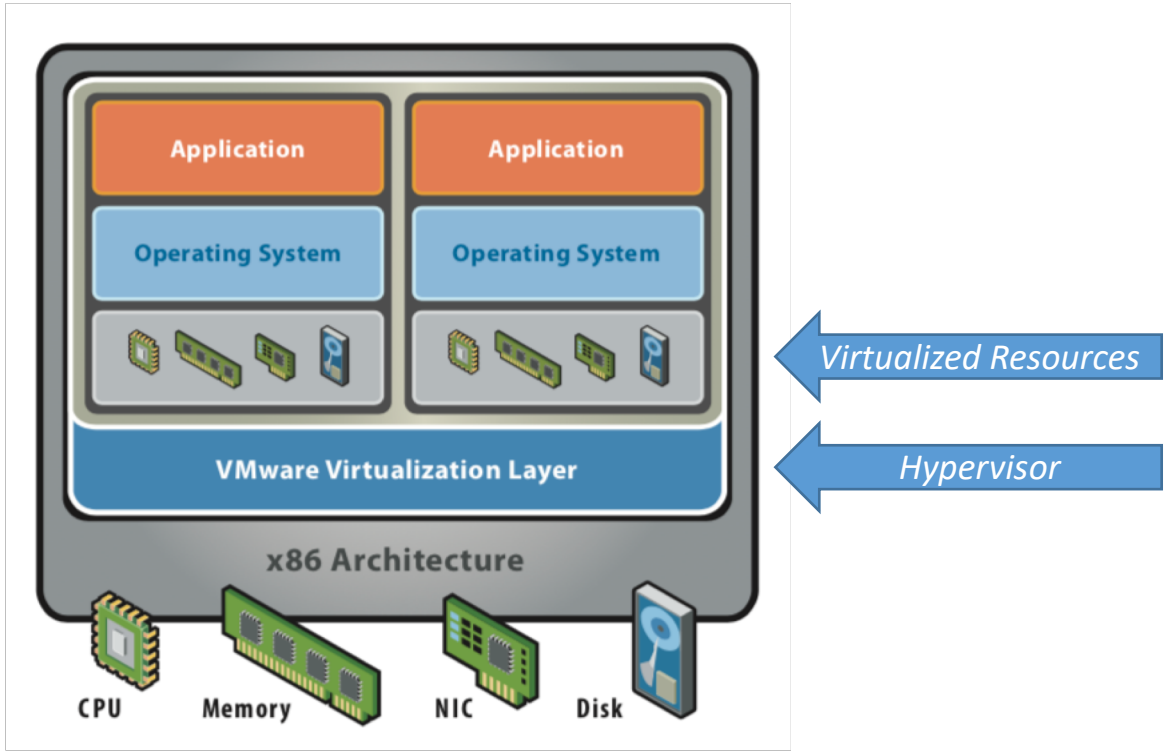
Source: Preethi Kasireddy FreeCodeCamp - Medium

Hypervisor

Virtual Machine → OS on top of **virtualized**
CPU, RAM… (via Hypervisor)

== Software/Hardware/Firmware.

- VMs run on a Hypervisor
- Hypervisor runs on the Guest Machine
- Hypervisor provides VM with resources (CPU, RAM, Disk, Net, GPU)



Source: VMWare

Hypervisor (Hosted vs. Bare-Metal)

- Hosted: Software that runs on OS
  - VMWare Player, Fusion, Workstation; VirtualBox; Parallels
  - Underlying hardware is less important. So VM is portable
    - *Move VM from Mac to PC to Linux*
  - Performance is lowered

- Bare-Metal: Runs natively on hardware.
  - VMWare ESX; Xen…
  - Underlying hardware is important so portability is sacrificed.
  - Performance, Stability and Scalability improves because hypervisor interfaces directly with underlying resources

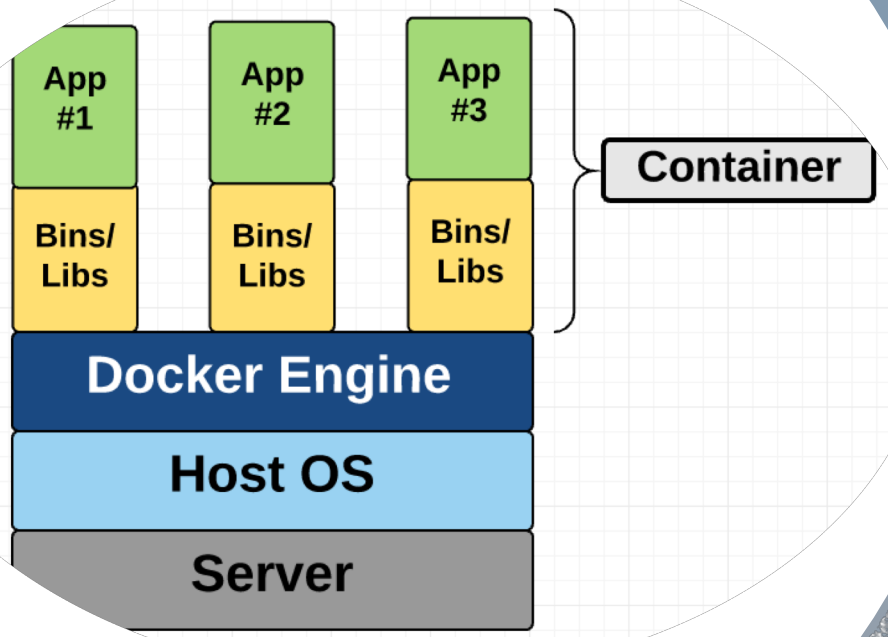
Containers

EMORY
UNIVERSITY
SCHOOL OF
MEDICINE

Containers

EMORY
UNIVERSITY
SCHOOL OF
MEDICINE

Containers

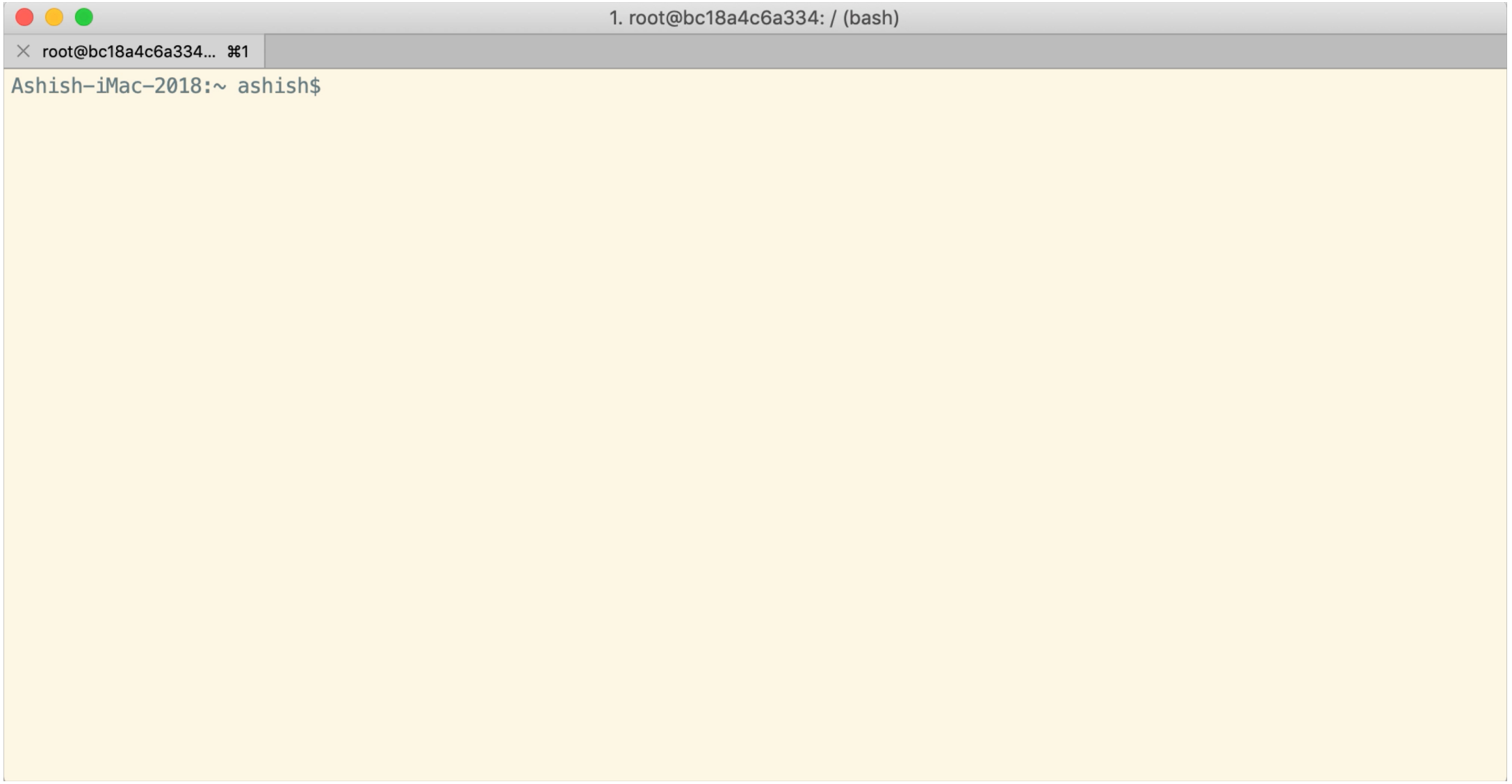


Looks like a VM, but shares the OS.

- Inside a container, what you see is a fully functional OS with one difference

- Only the application that you need exists inside the container

Source: Preethi Kasireddy FreeCodeCamp - Medium



EMORY UNIVERSITY SCHOOL OF MEDICINE

What does that mean?

Recap

What version of Docker? ➢docker version

Give me some info about the host ➢docker --info

Start an ubuntu container and run bash in it ➢docker run -it **ubuntu** bash

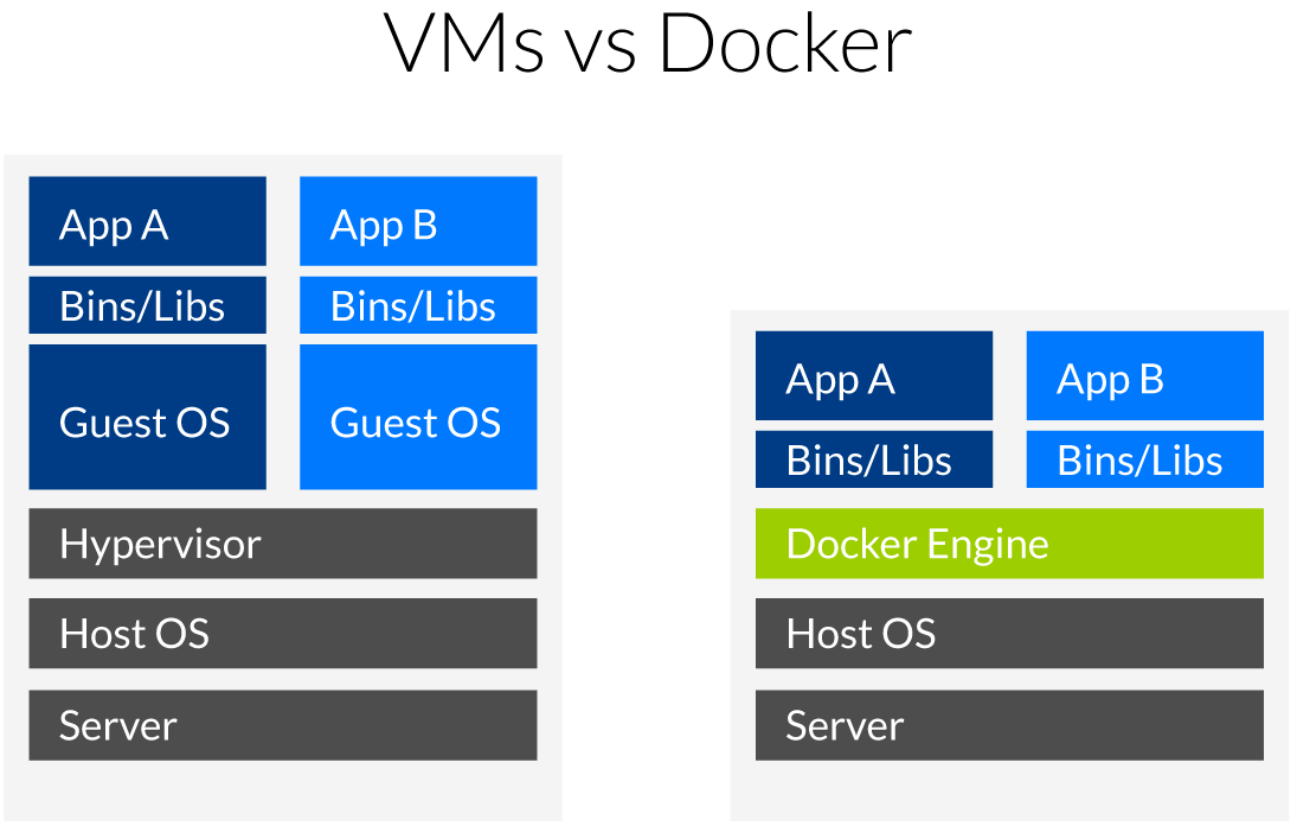
```
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
124c757242f8: Extracting [==============================>                    ]  19.66MB/31.76MB
9d866f8bde2a: Download complete
fa3f2f277e67: Download complete
398d32b153e8: Download complete
afde35469481: Download complete
```

Only one process running in it ➢ps

VM vs. Container

- Provides OS level virtualization by abstracting the “user space”.
- Looks like a VM w/ private IP, network, storage…
- Containers share the host system's kernel with other containers

EMORY UNIVERSITY SCHOOL OF MEDICINE



Benefits

- **Rapid application development:** Minimize the overhead to deploy an application by providing the minimal runtime requirements only.

- **Modularity and Scalability**

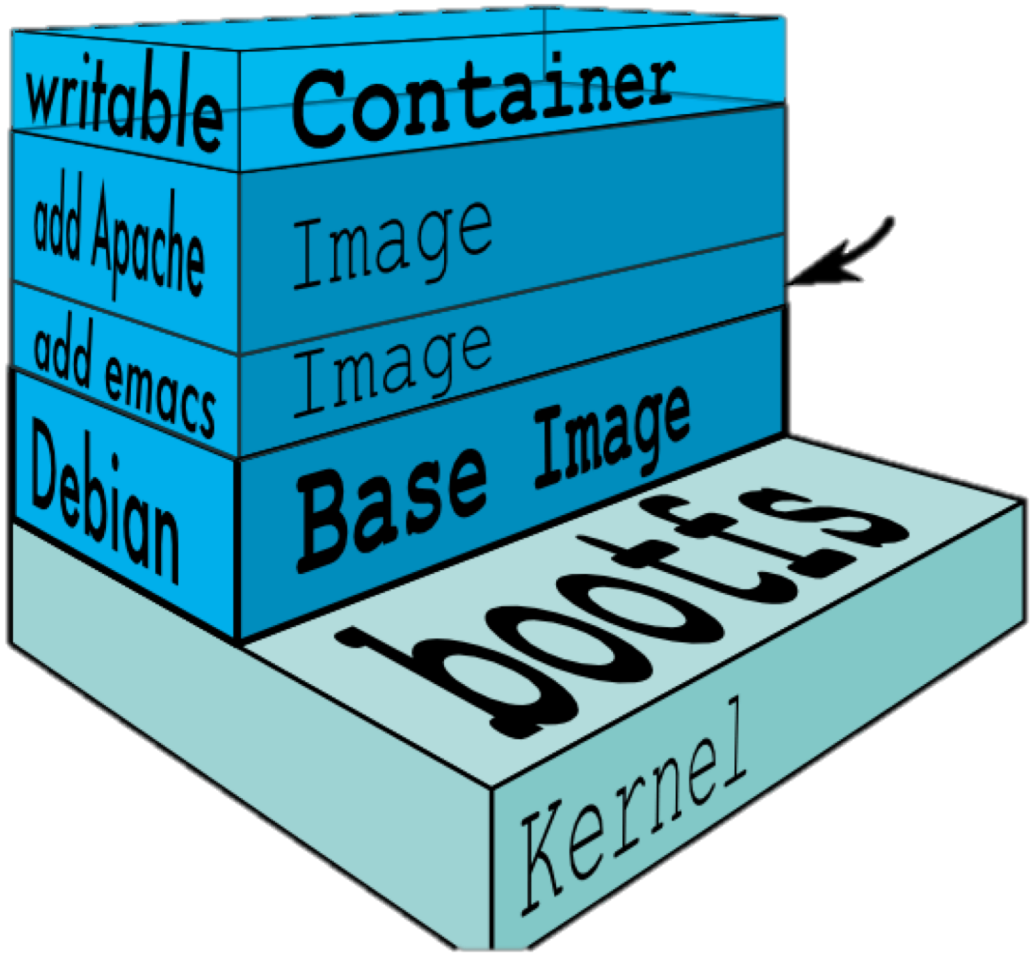
- **Portability:** An application and all its dependencies are bundled into a single container which can be shipped around. The container is independent from the host OS.

Benefits

- **Version control, component reuse, Share:** DockerHub == github for containers. You can share your containers.

- **Lightweight:** Regular Docker images are quite small to provide rapid delivery and to reduce the time needed to deploy new application containers.
  A Docker container spins up in a few seconds, only a fraction of the time required to boot a VM.

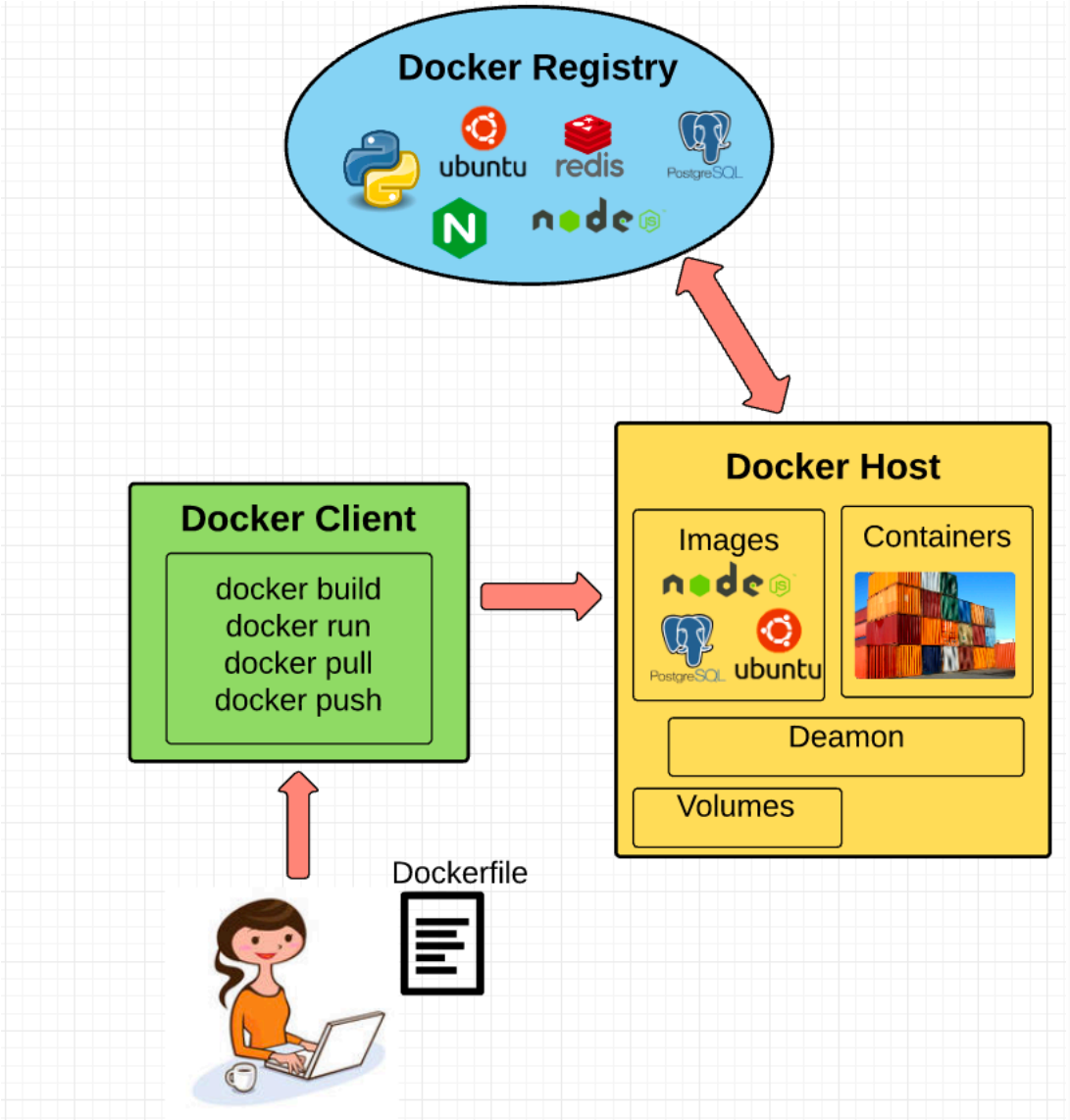
EMORY
UNIVERSITY
SCHOOL OF
MEDICINE



Docker Container

- Wraps an application's software into an invisible box with everything the application needs to run.
- Includes the OS, application code, runtime, system tools, system libraries, and etc.
- Docker containers are built off Docker images.
- Since images are read-only, Docker adds a read-write file system over the read-only file system of the image to create a container.

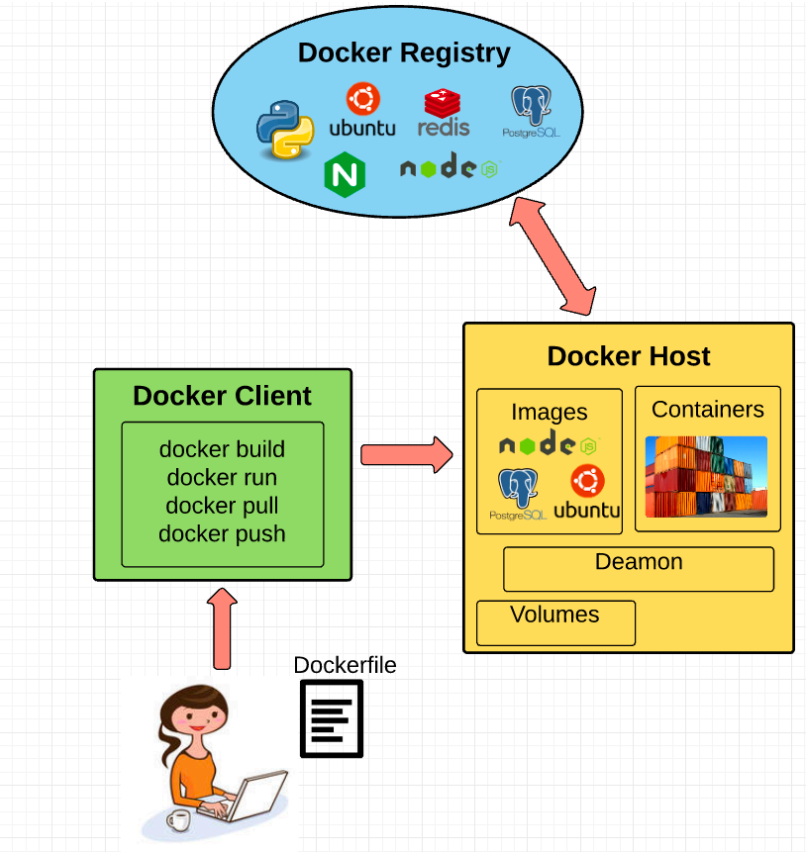
EMORY
UNIVERSITY
SCHOOL OF
MEDICINE



Core Concepts

Key Concepts

- Dockerfile ← Describe how to build a container. What goes in it
- Engine ← Makes containers run
- Client ← Interact with Docker.
- Image ← read-only templates that you build from a set of instructions written in your Dockerfile



Orchestration

EMORY
UNIVERSITY
SCHOOL OF
MEDICINE

EMORY
UNIVERSITY
SCHOOL OF
MEDICINE

Kubernetes

A Graphical Introduction

https://cloud.google.com/kubernetes-engine/kubernetes-comic/

Scientific Workflows

- Tools and Workflows
  - Tools → [Dockerized applications]
  - Workflow → Directed Acyclic Graphs that denote computation

Plenty of workflow solutions to go around

So of course we decided to create a new one.

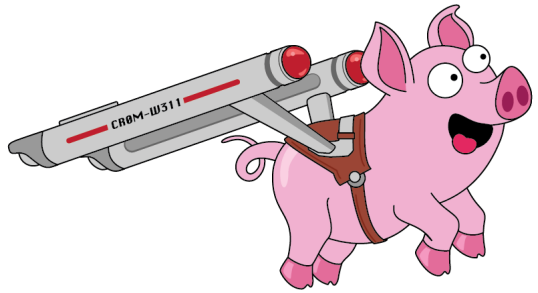
Randall Munroe, XKCD
https://www.xkcd.com/927/

Ref: Cromwell & WDL – Jeff Gentry, Broad Institute

Meet WDL + Cromwell

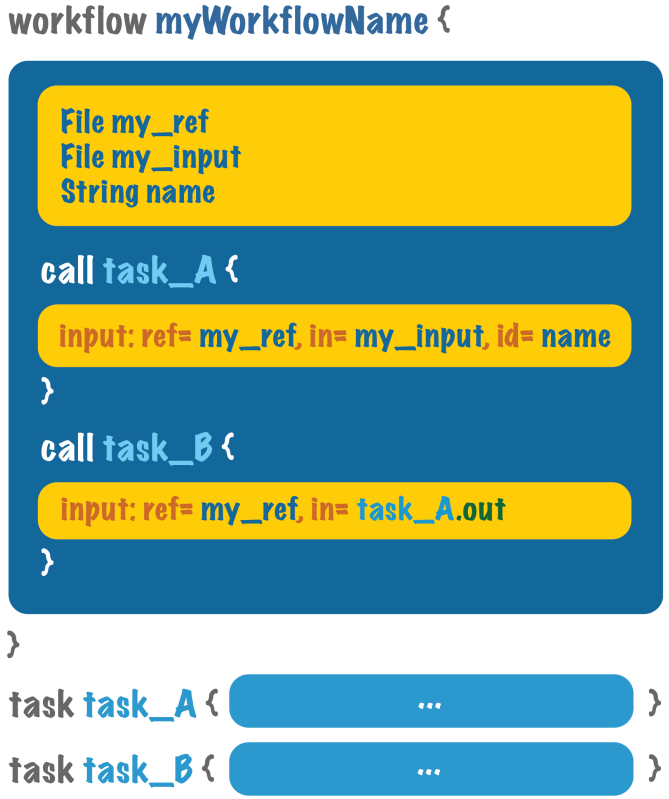
- Workflow language that humans can read/write
  - Methods developers and biomedical scientists at large
  - https://software.broadinstitute.org/wdl/

- Execution engine that can
  - Run on any platform (on-prem *and* on Cloud)
  - Scale elastically based on workflow needs
  - https://github.com/broadinstitute/cromwell



Ref: Cromwell & WDL – Jeff Gentry, Broad Institute

Workflow Description Language



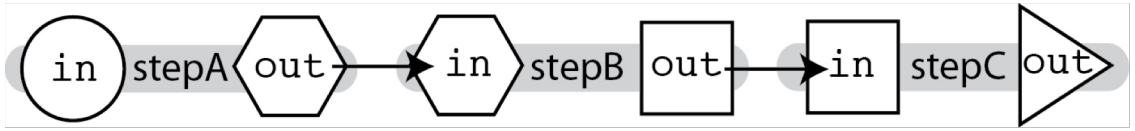

```
workflow helloHaplotypeCaller {
  call haplotypeCaller
}
task haplotypeCaller {
  File GATK
  File RefFasta
  File RefIndex
  File RefDict
  String sampleName
  File inputBAM
  File bamIndex
  command {
    java -jar ${GATK} \
        -T HaplotypeCaller \
        -R ${RefFasta} \
        -I ${inputBAM} \
        -o ${sampleName}.raw.indels.snps.vcf
  }
  output {
    File rawVCF = "${sampleName}.raw.indels.snps.vcf"
  }
}
```

https://software.broadinstitute.org/wdl/

Ref: Cromwell & WDL – Jeff Gentry, Broad Institute

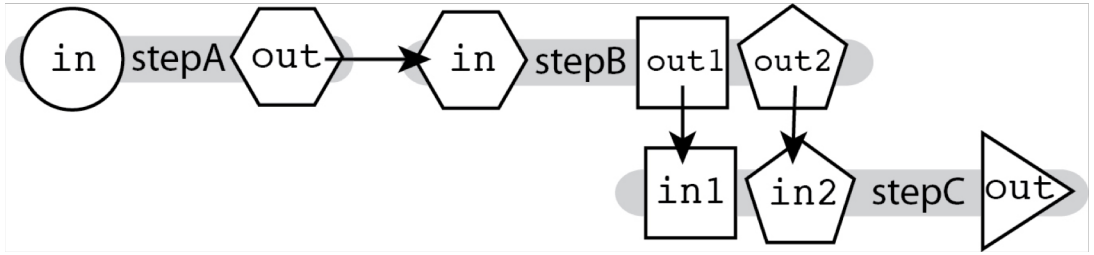
Basic WDL plumbing

LINEAR CHAINING



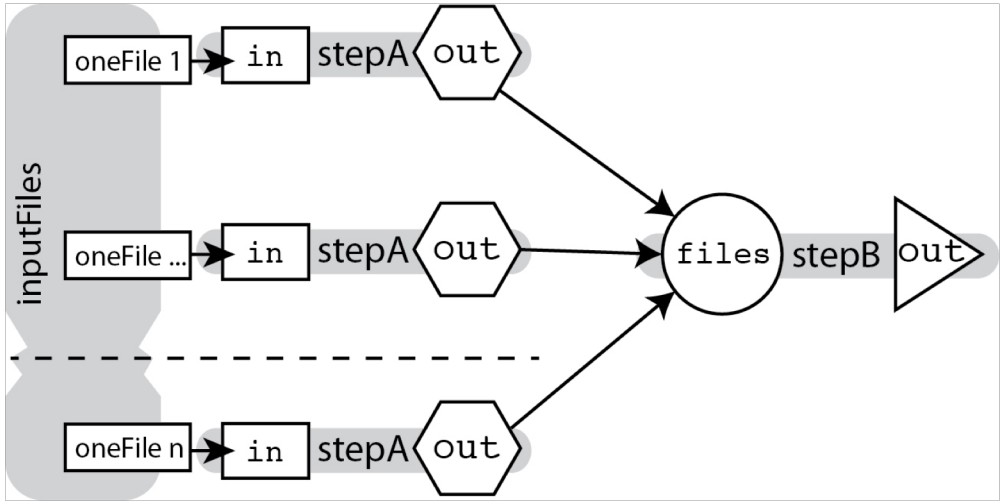

```
call stepA
call stepB { input: in=stepA.out }
call stepC { input: in=stepB.out }
```

MULTI-IN/OUT



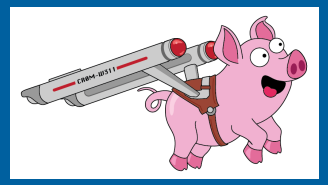

```
call stepC { input :
                in1=stepB.out1,
                in2=stepB.out2 }
```

SCATTER-GATHER



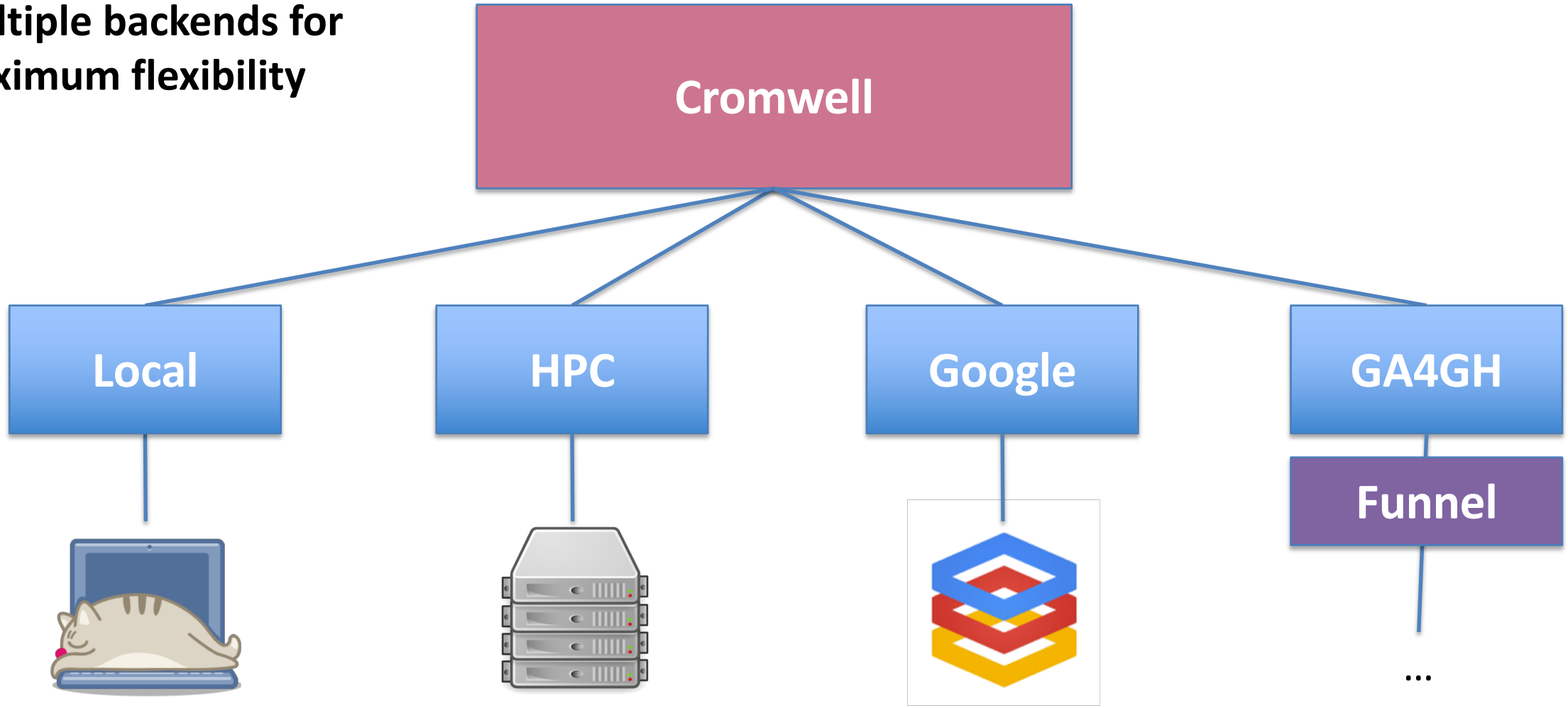

```
Array[File] inputFiles

scatter(oneFile in inputFiles) {
    call stepA { input: in=oneFile }
}

call stepB { input: files=stepA.out }
```

Ref: Cromwell & WDL – Jeff Gentry, Broad Institute



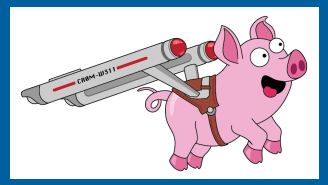
Cromwell execution engine

Multiple backends for maximum flexibility



Coming Soon: AWS, Azure, Alicloud

Ref: Cromwell & WDL – Jeff Gentry, Broad Institute



Two main ways to run Cromwell

One-off

- Simple self-contained command

```
java -jar cromwell.jar \
     run hello.wdl \
     hello_inputs.json
```

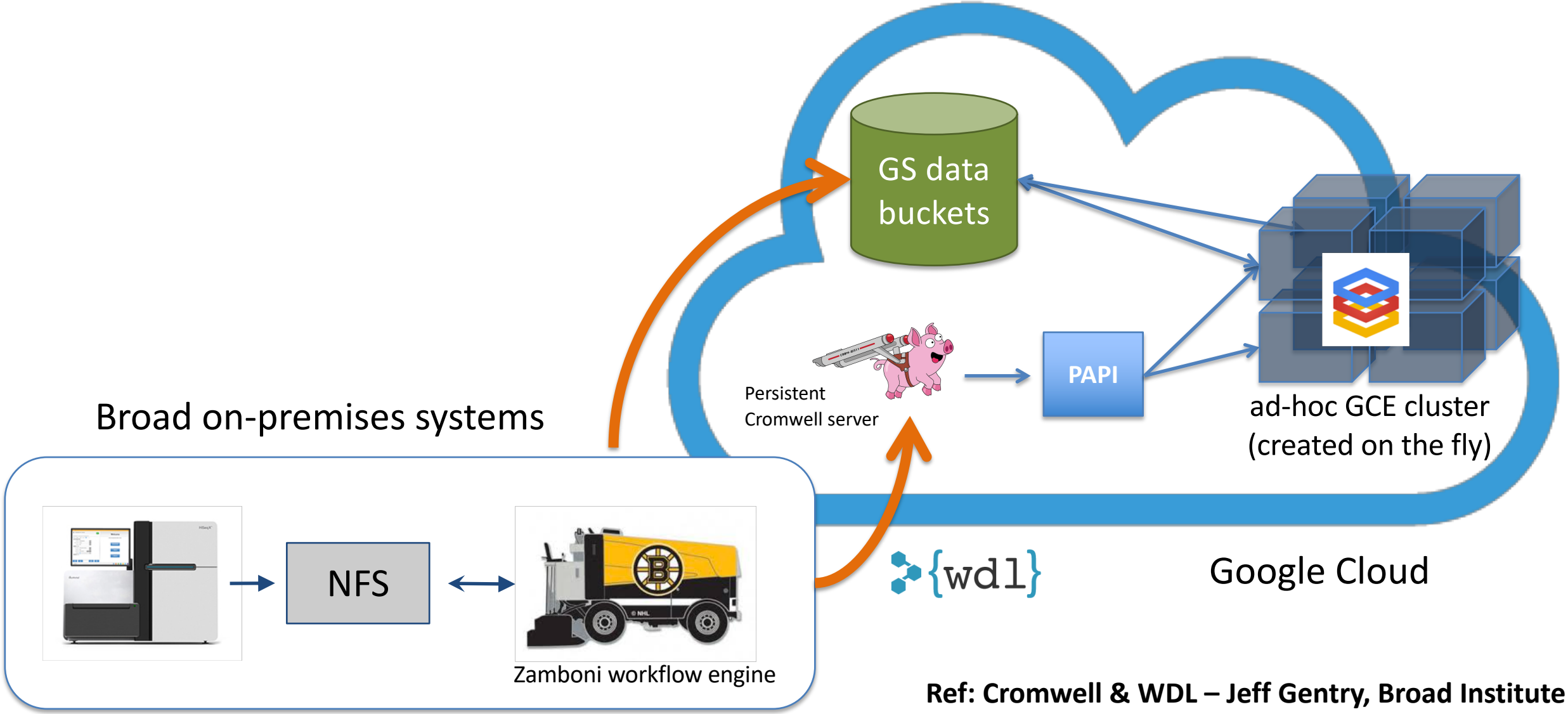
- Appropriate for independent analysts

Server mode

- API endpoints
- More scalable
- Some devops needs
- Appropriate for production environments
- Call-caching!
  (aka “ka-ching”)

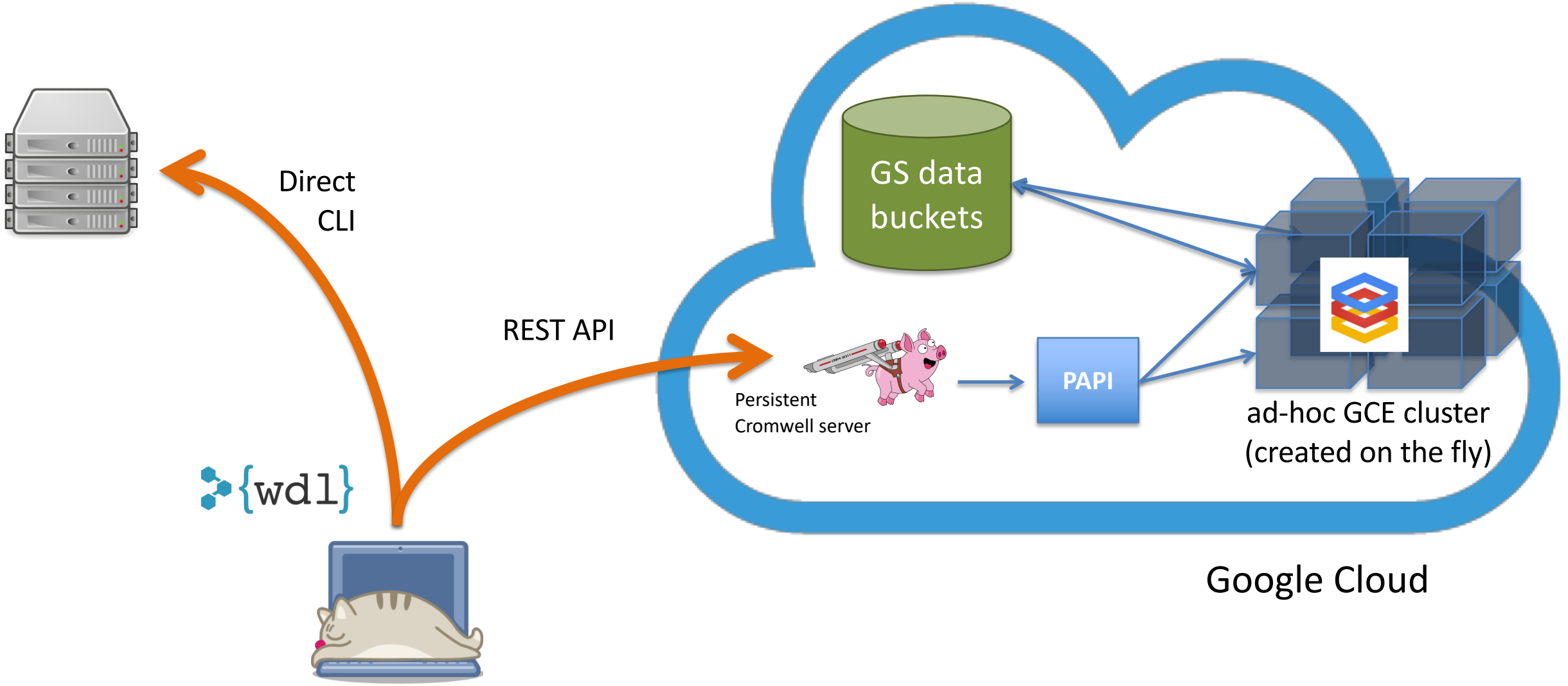
Ref: Cromwell & WDL – Jeff Gentry, Broad Institute

Our production system: Genomes On The Cloud



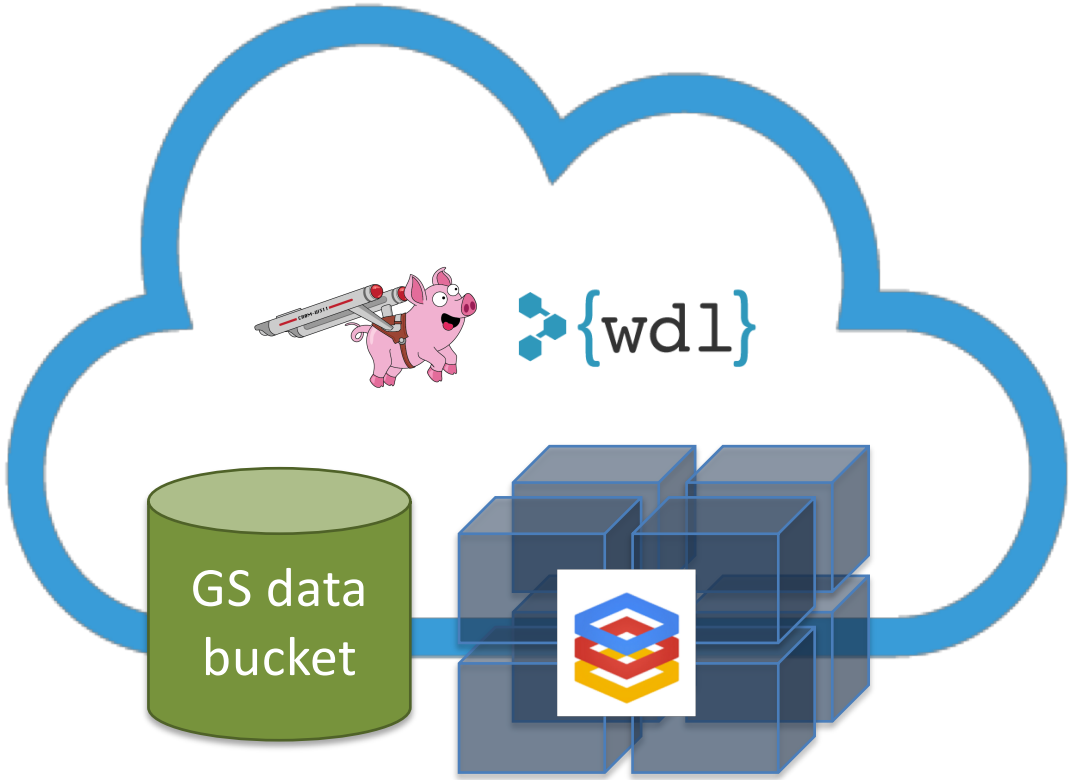
Ref: Cromwell & WDL – Jeff Gentry, Broad Institute

Our development setup: on-prem + on-cloud



Ref: Cromwell & WDL – Jeff Gentry, Broad Institute

Example external implementation: Google wdl_runner



ad-hoc GCE cluster
(created on the fly)

Barebones implementation:

- Creates GCE VM
- Executes wdl_runner.py
  - Sets up Cromwell
  - Parses WDL workflow
  - Submits jobs to PAPI
  - Polls for completion
  - Copies metadata & outputs to output path
- Destroys GCE VM

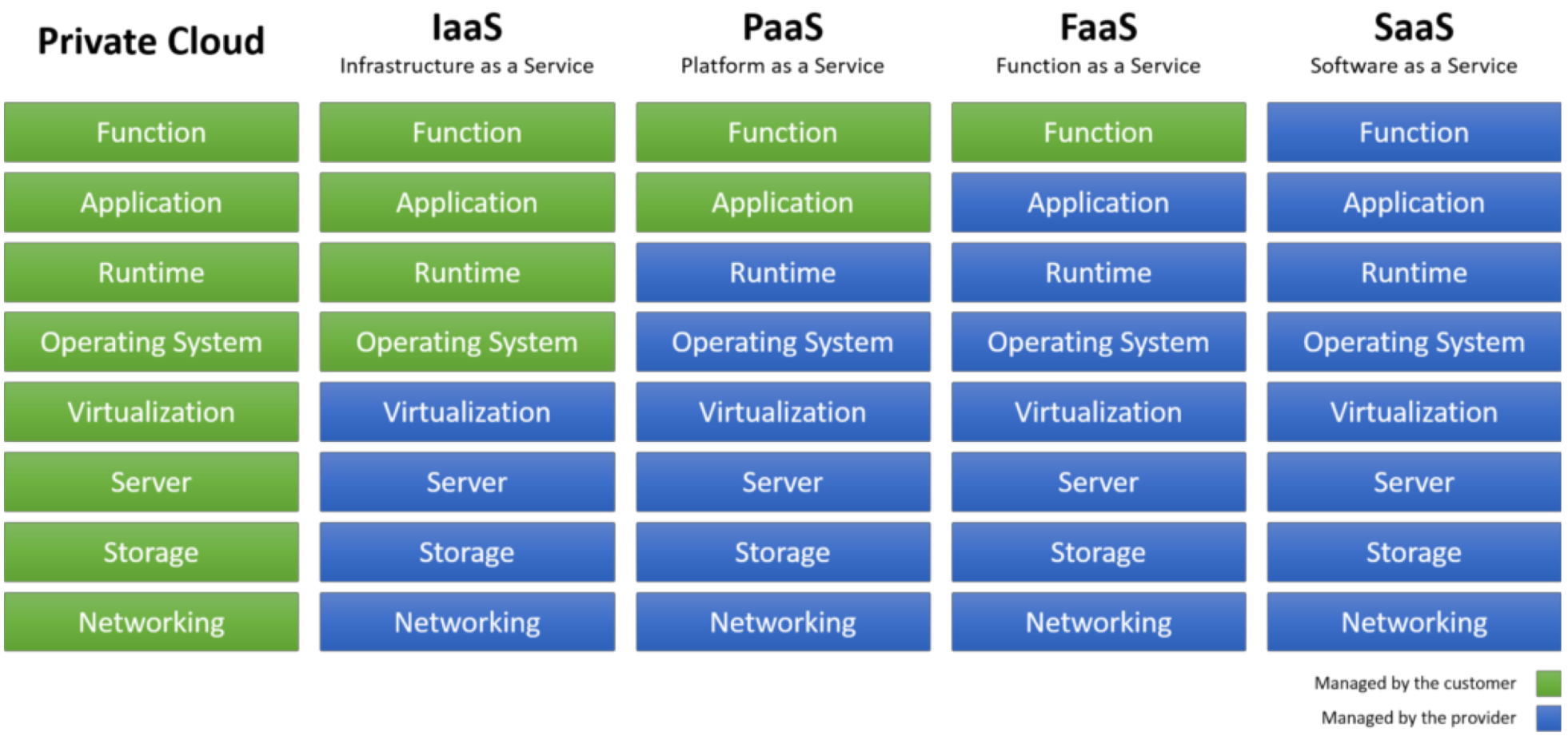
https://cloud.google.com/genomics/v1alpha2/gatk

Ref: Cromwell & WDL – Jeff Gentry, Broad Institute

Serverless Computing – FaaS

EMORY
UNIVERSITY
SCHOOL OF
MEDICINE

Serverless or Function-As-A-Service



Source: Nuno Costa, "**From servers to functions, the serverless story**", Medium

FaaS (Serverless) Providers

AWS Lambda

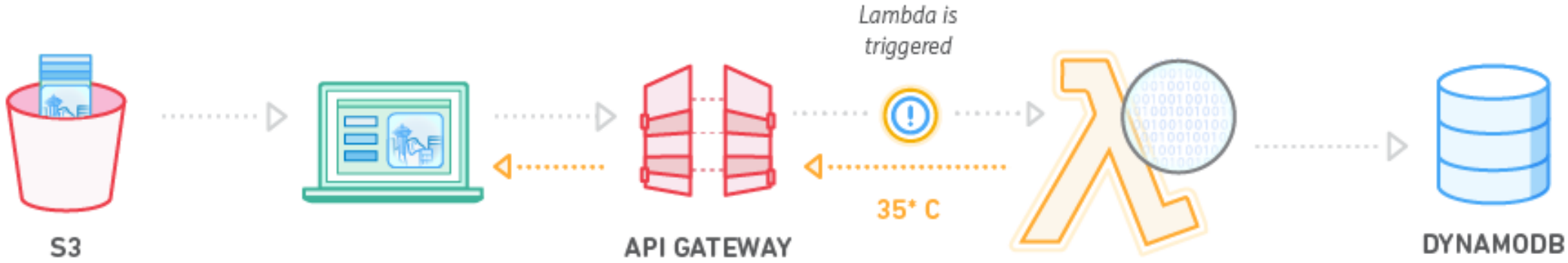
Google Cloud Functions

Azure Functions

IBM CloudFunctions Apache OpenWhisk

EMORY UNIVERSITY SCHOOL OF MEDICINE

Example: *Weather Application*

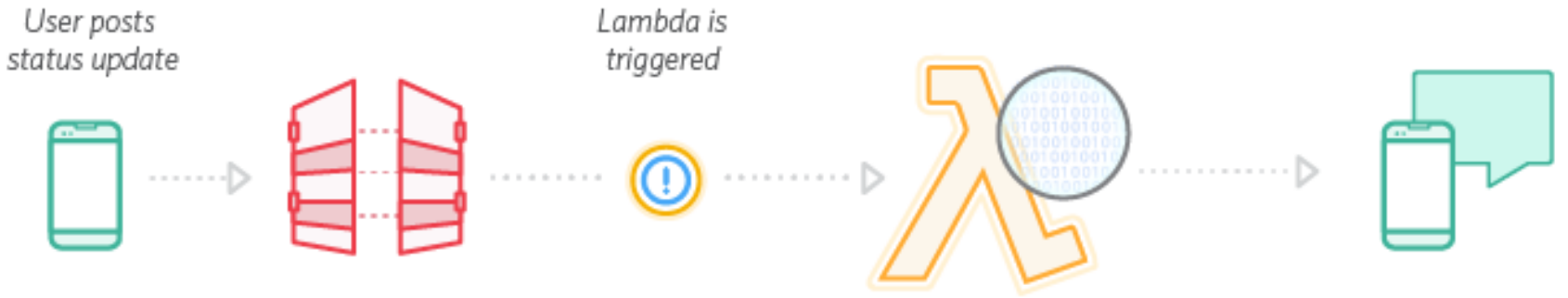


Ref: https://aws.amazon.com/serverless/

Simple Web-App → Client-Server (no Server)

EMORY UNIVERSITY SCHOOL OF MEDICINE

Example: *Mobile Backend for Social Media App*



Ref: https://aws.amazon.com/serverless/

Mobile App — Status Updates

EMORY
UNIVERSITY
SCHOOL OF
MEDICINE

Example: *Image Thumbnail Creation*

Ref: https://aws.amazon.com/serverless/

On-Demand Service: Create Thumbnail / OCR on bank check…

Example: *Analysis of Streaming Social Media Data*

Ref: https://aws.amazon.com/serverless/