

ORACLE®

Cloud Native Labs



Serverless Patterns

Basic Patterns with Functions and Serverless

Jesse Butler Cloud Native Advocate, Oracle Cloud Infrastructure

 @jlb13

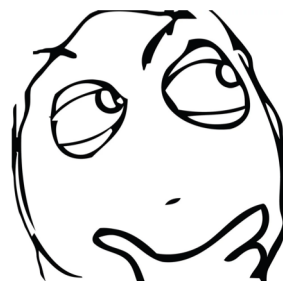
#OracleCloudNative
cloudnative.oracle.com

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Level Set

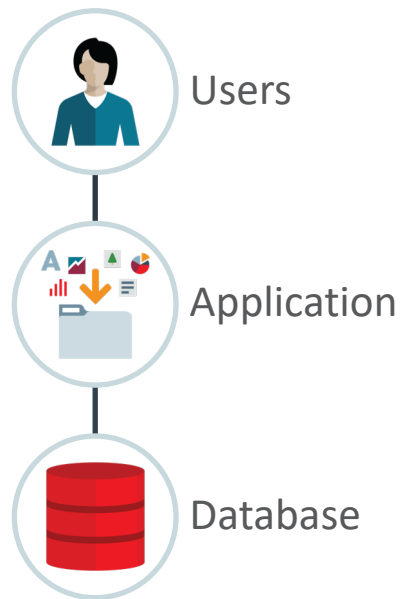
- Devs, Ops, DevOps?
- Serverless users?
- In production?
- Lambda? Azure? Something Else?



Monolithic Applications

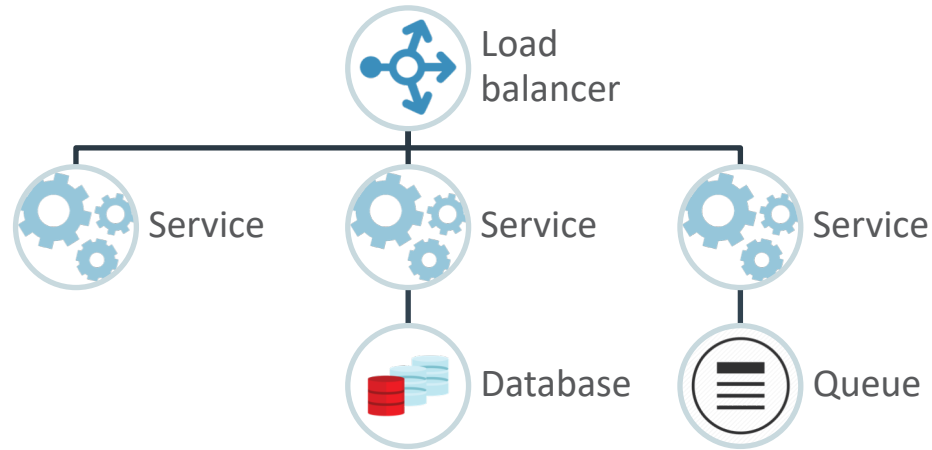


Monolithic Applications



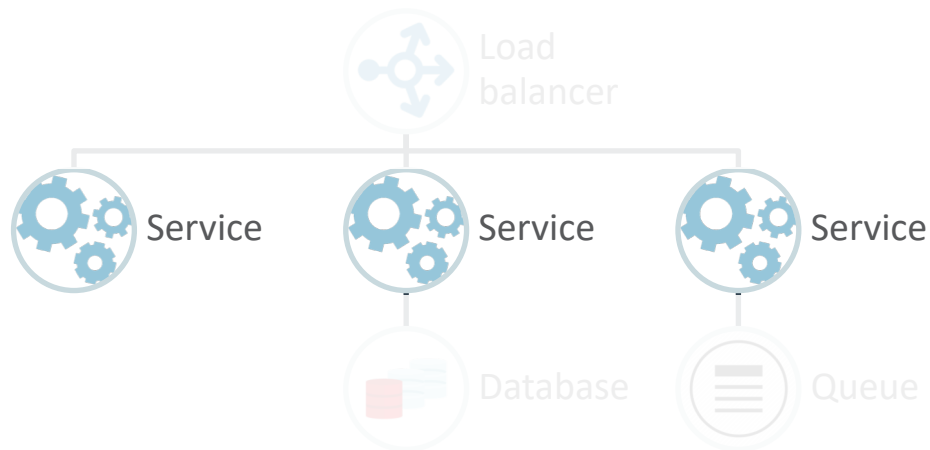
Microservices

Deploying Code to Systems We Build in the Cloud with Containers and Kubernetes

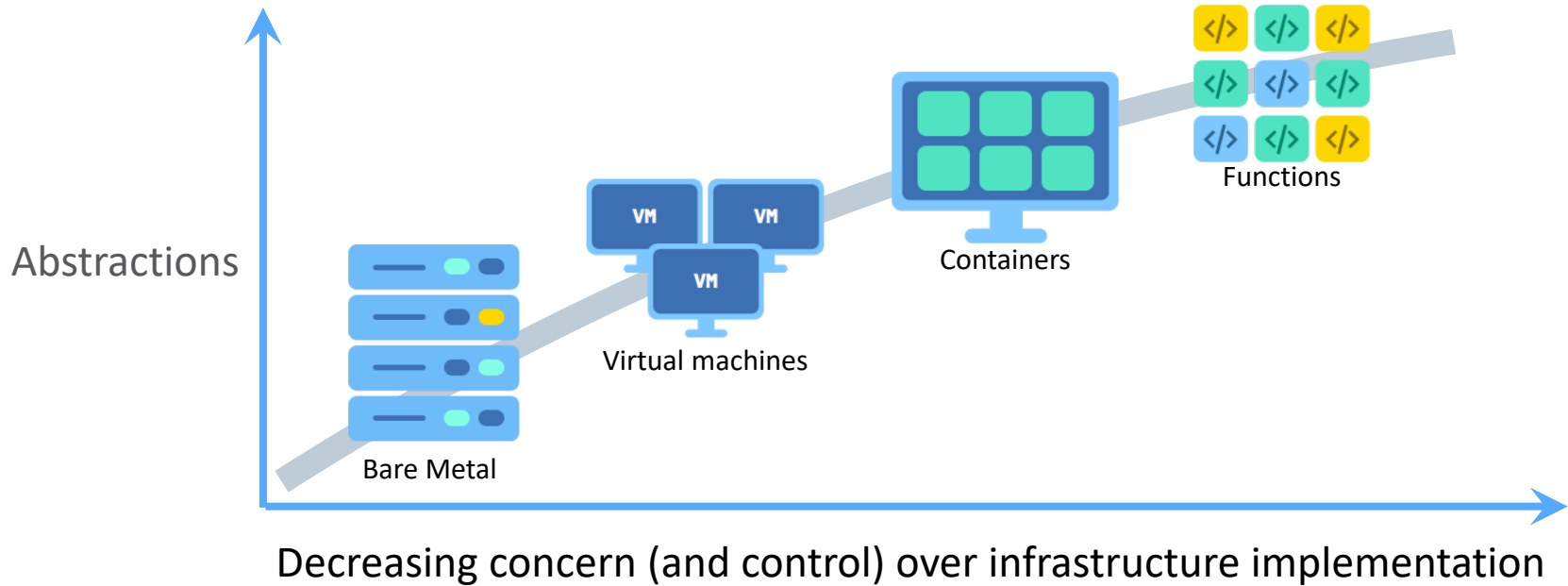


Serverless

Deploying Code to Systems We Build in the Cloud with Containers and Kubernetes

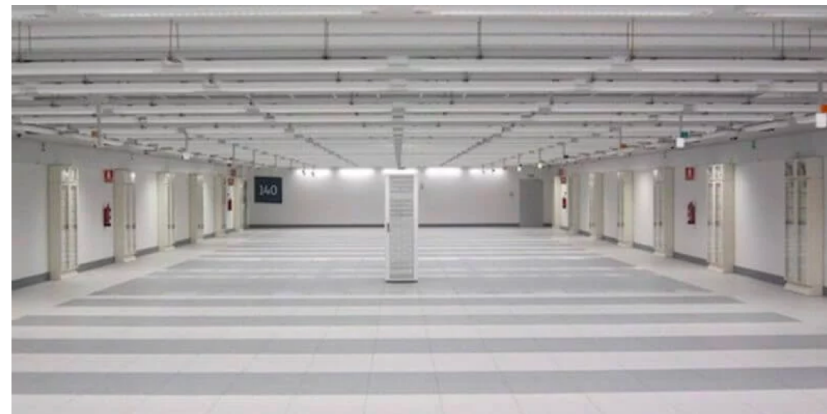


Trend towards Serverless

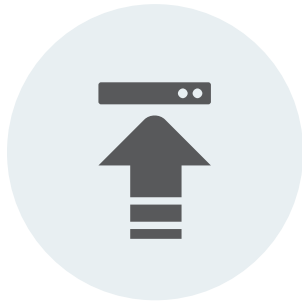


What Is Serverless?

- Event-driven architecture
- Invisible infrastructure
- Automatic scaling on demand
- Granular billing for execution time
- Fault tolerant and highly available



How Does it Work?



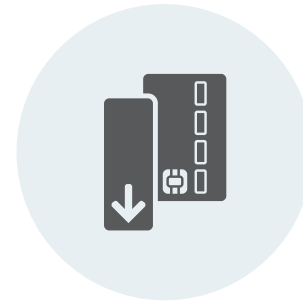
Upload
Function
Source Code



Configure
Function
Trigger



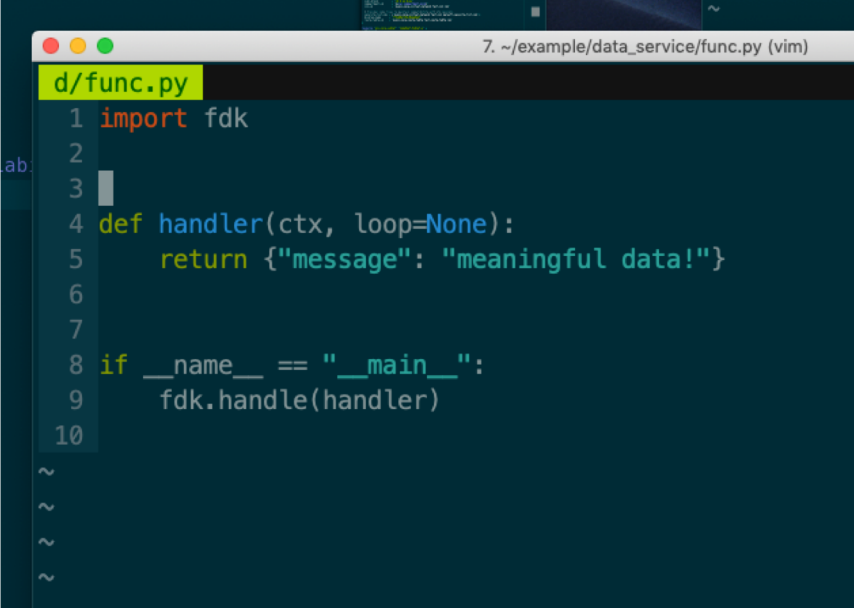
Function is
invoked
when
triggered



Pay for execution
time, not idle
time

Function Example

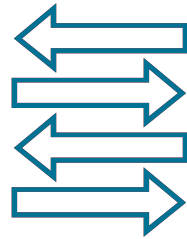
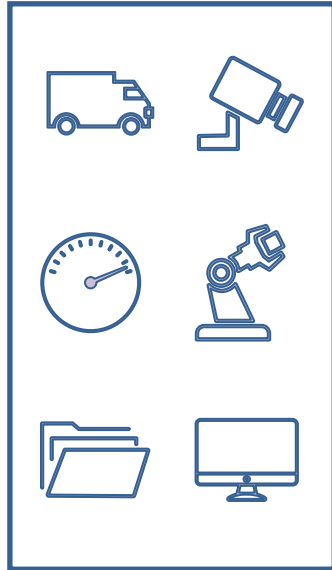
- Different projects and products differ in use and workflow
- This example function can be deployed in Oracle Functions
- Just the code, configured against any number of event triggers
- As with microservices, applications are composed of many functions

A screenshot of a code editor window titled "7. ~/example/data_service/func.py (vim)". The editor shows a Python file named "d/func.py" with the following code:

```
1 import fdk
2
3
4 def handler(ctx, loop=None):
5     return {"message": "meaningful data!"}
6
7
8 if __name__ == "__main__":
9     fdk.handle(handler)
10
```

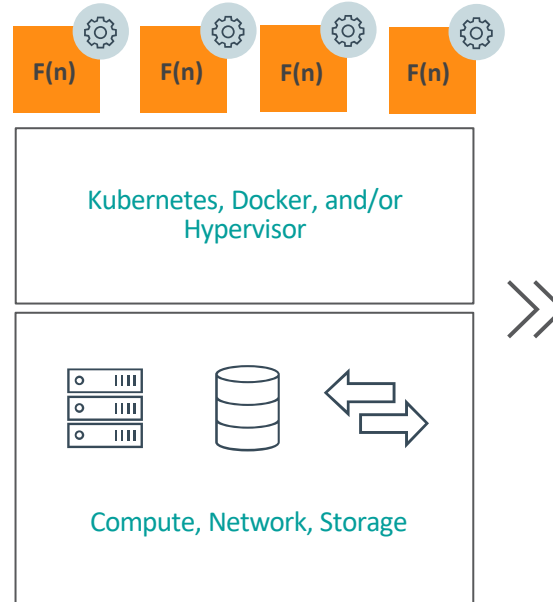
Functions-as-a-Service

Event Sources

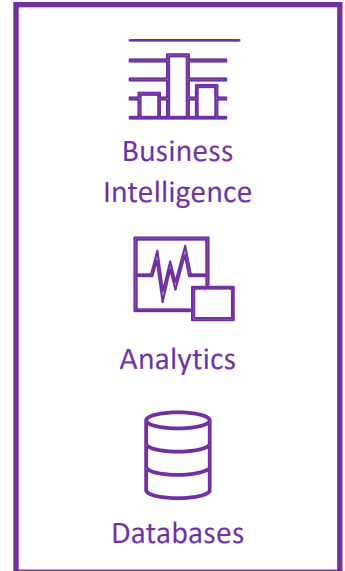


Triggers

Function Execution



Backend Services



Business Intelligence



Analytics



Databases

Oracle Functions

Oracle Functions

Functions-as-a-Service

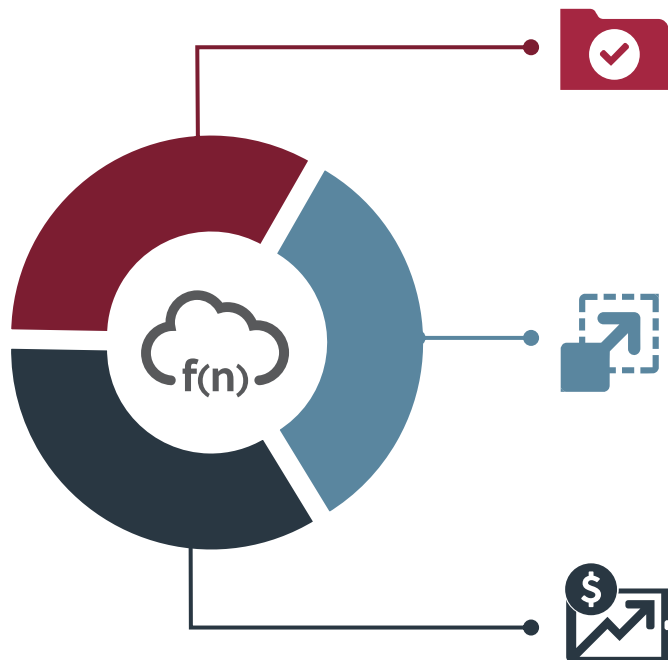
Oracle Cloud Integrated

Container Native

Open Source Engine

Multi-tenant

Secure



Pay Per Use

Pay for execution, not for idle time

Autonomous

Platform auto-scales functions
No servers to provision,
manage

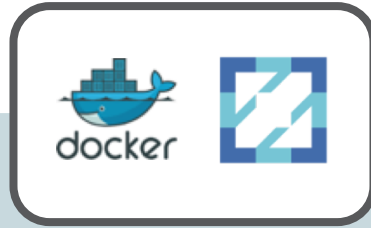
No Lock-in

Built on open-source Fn
Project and Docker

Key Features



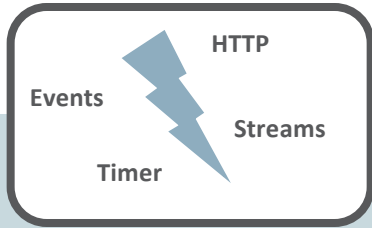
Open Source Engine



Container Native



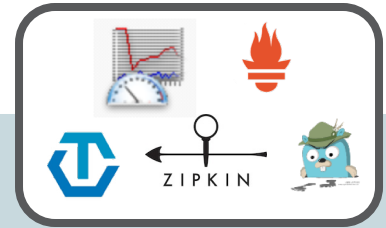
Function Dev Kits



Oracle Cloud Triggers



Fine-grained Billing



Advanced Diagnostics

Execution Model

- Synchronous
- Asynchronous
- Streaming



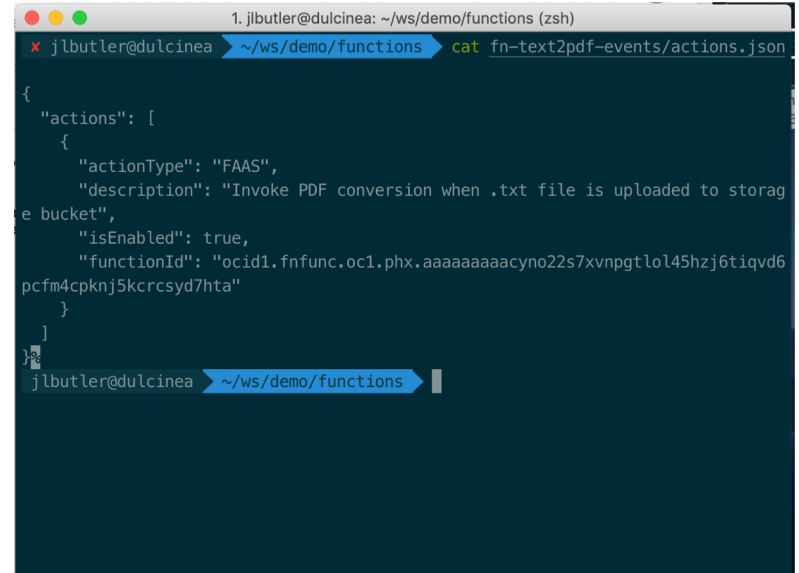
Events, Determined by Context

- Changes in data
- API Invocations
- Requests on endpoints
- Changes in resources
- Timers



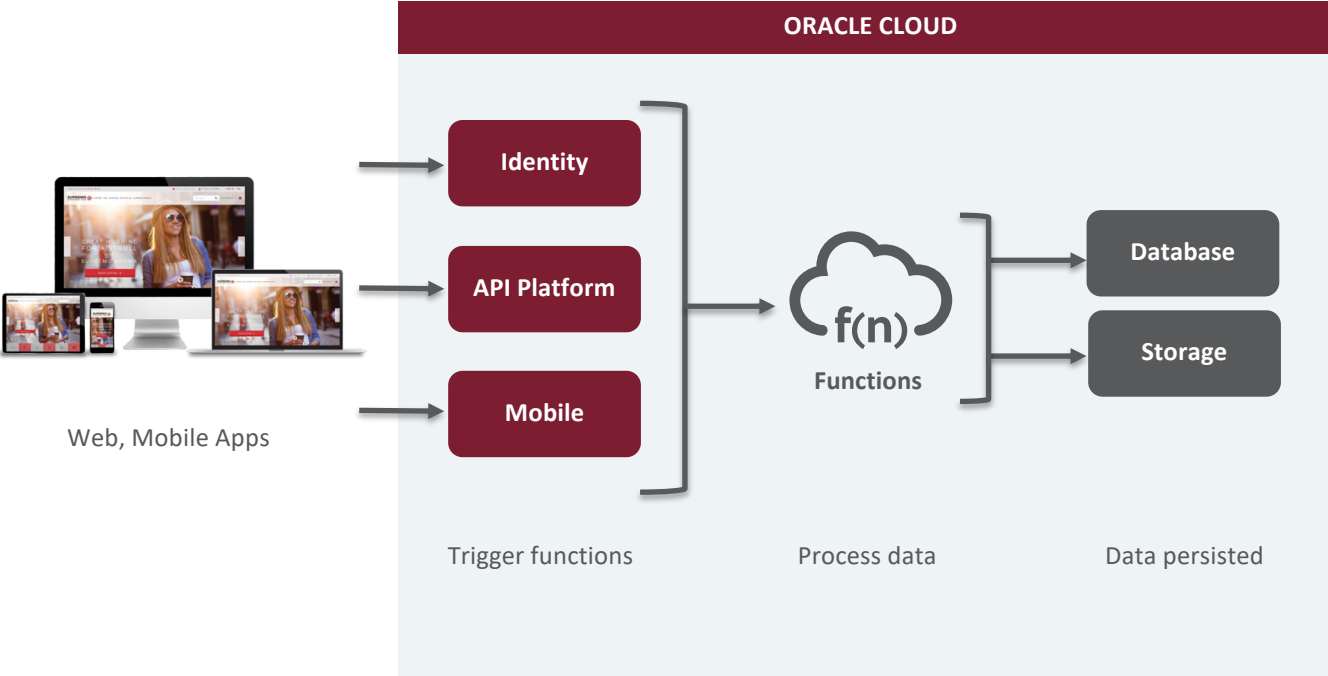
Using Events

- Events are configured differently by platform
- Abstractly, inform the platform to what event(s) should invoke this function
- CNCF Serverless WG has drafted the CloudEvents specification
- Oracle Functions use Cloud Events format for portability

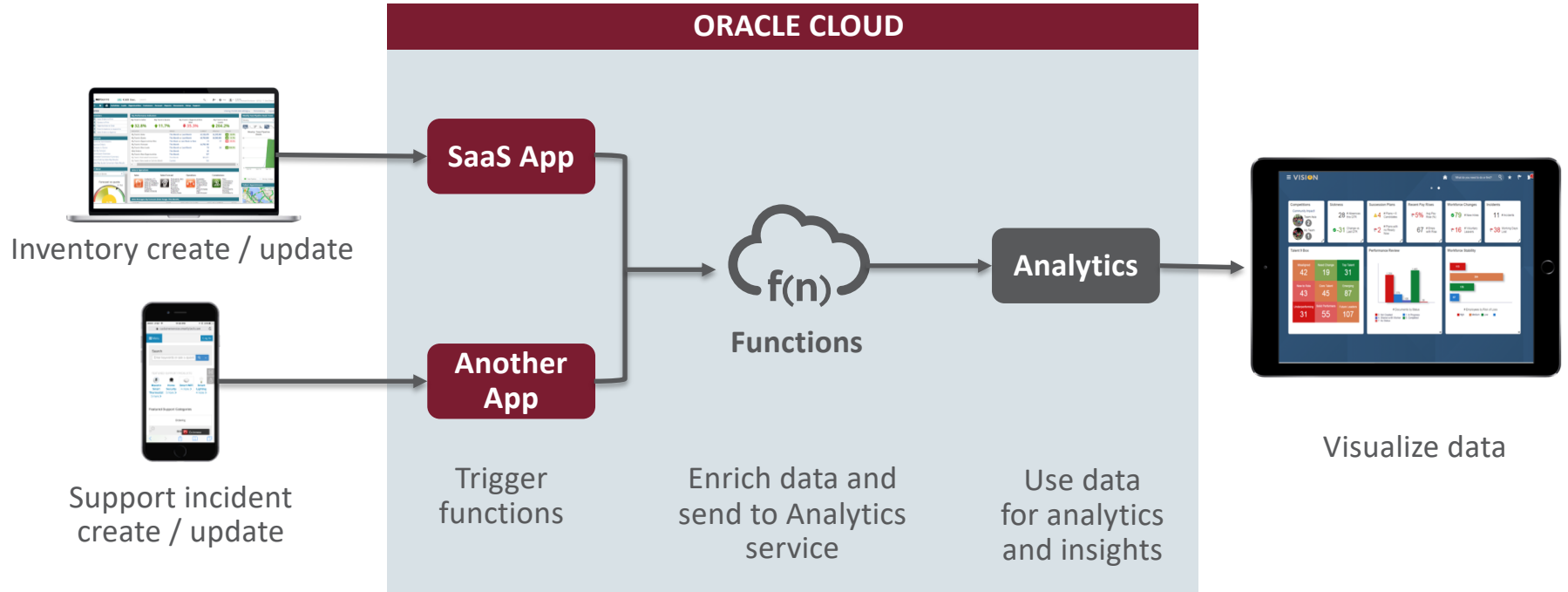


```
1. jlbutler@dulcinea: ~/ws/demo/functions (zsh)
jlbutler@dulcinea ~/ws/demo/functions cat fn-text2pdf-events/actions.json
{
  "actions": [
    {
      "actionType": "FAAS",
      "description": "Invoke PDF conversion when .txt file is uploaded to storage bucket",
      "isEnabled": true,
      "functionId": "ocid1.fnfunc.oc1.phx.aaaaaaaaacyno22s7xvnpgtlo45hzj6tiqvd6pcfm4cpknj5kcrcsyd7hta"
    }
  ]
}
```

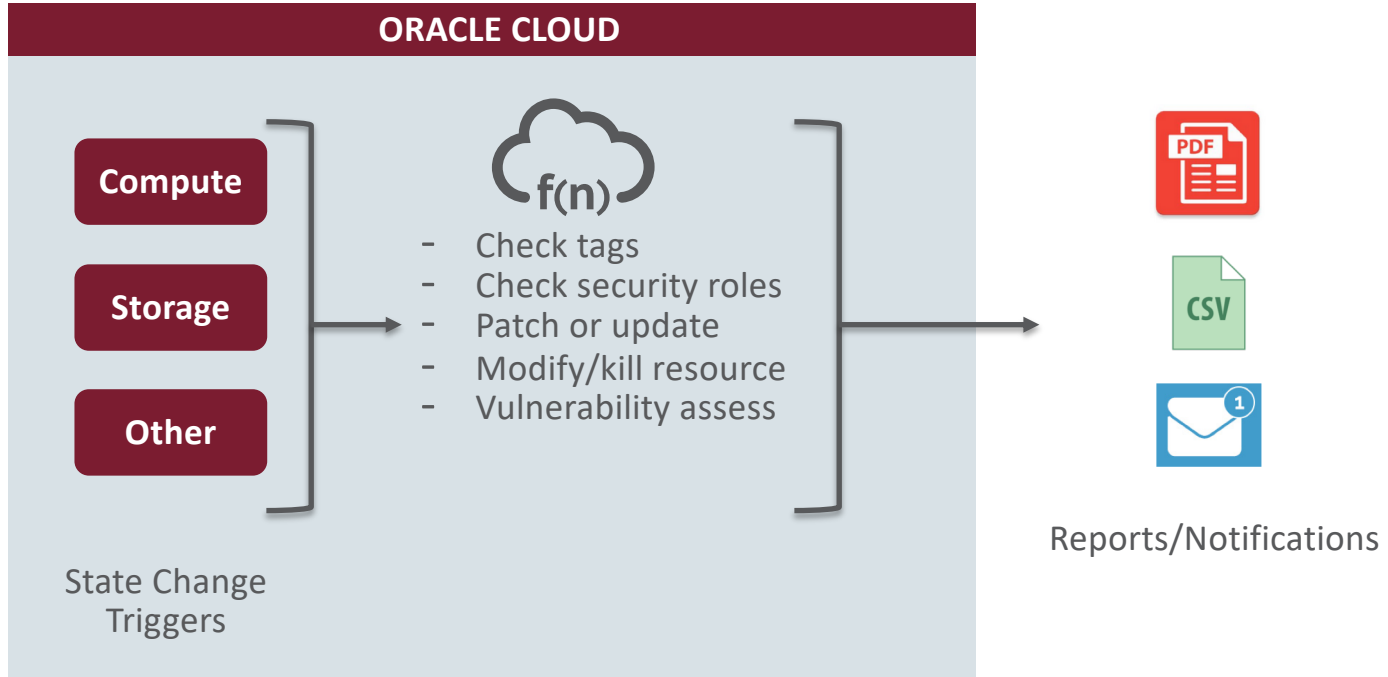
Use Case: Web and Mobile Backends



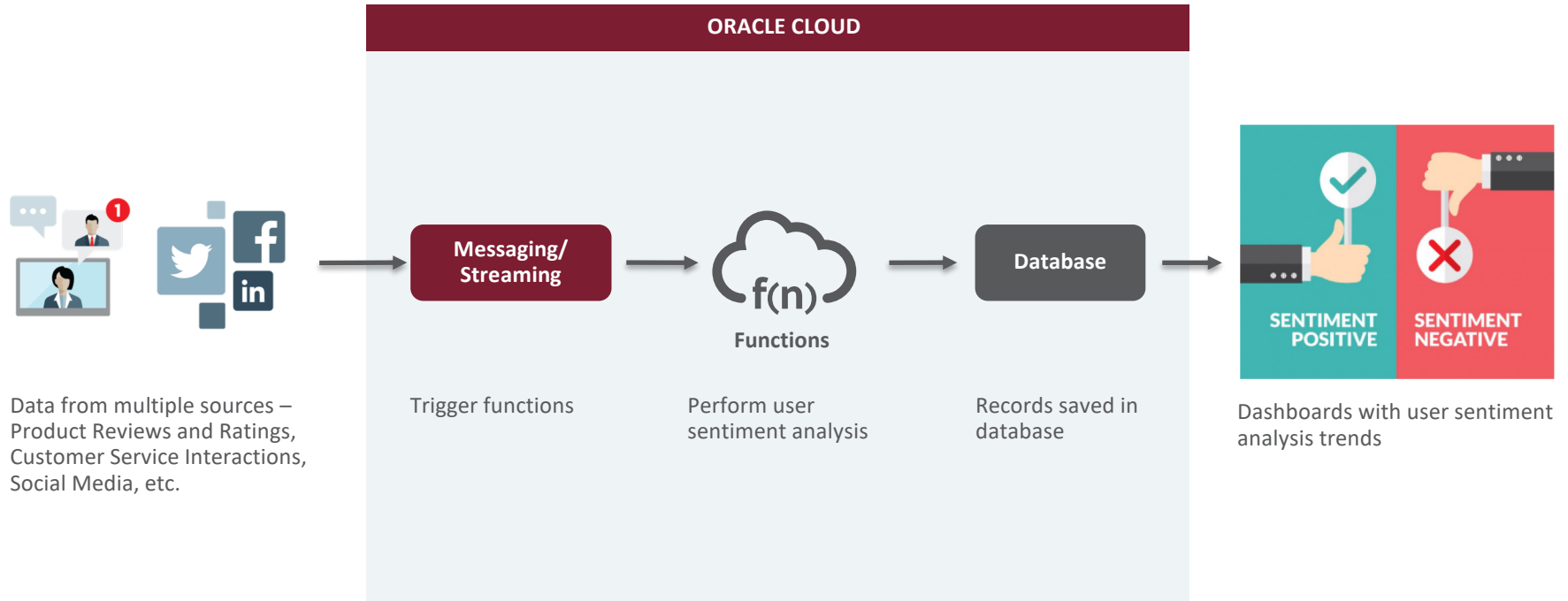
Extend and Enhance Existing Applications



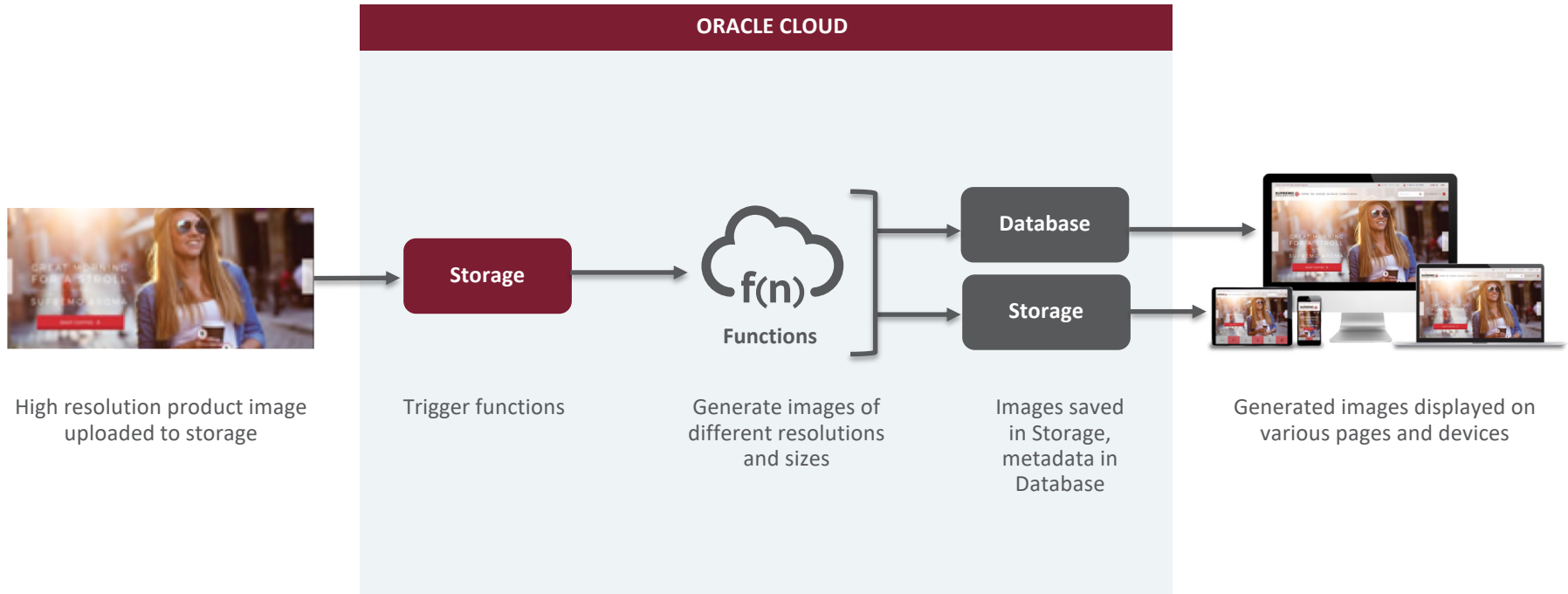
Automation and DevOps



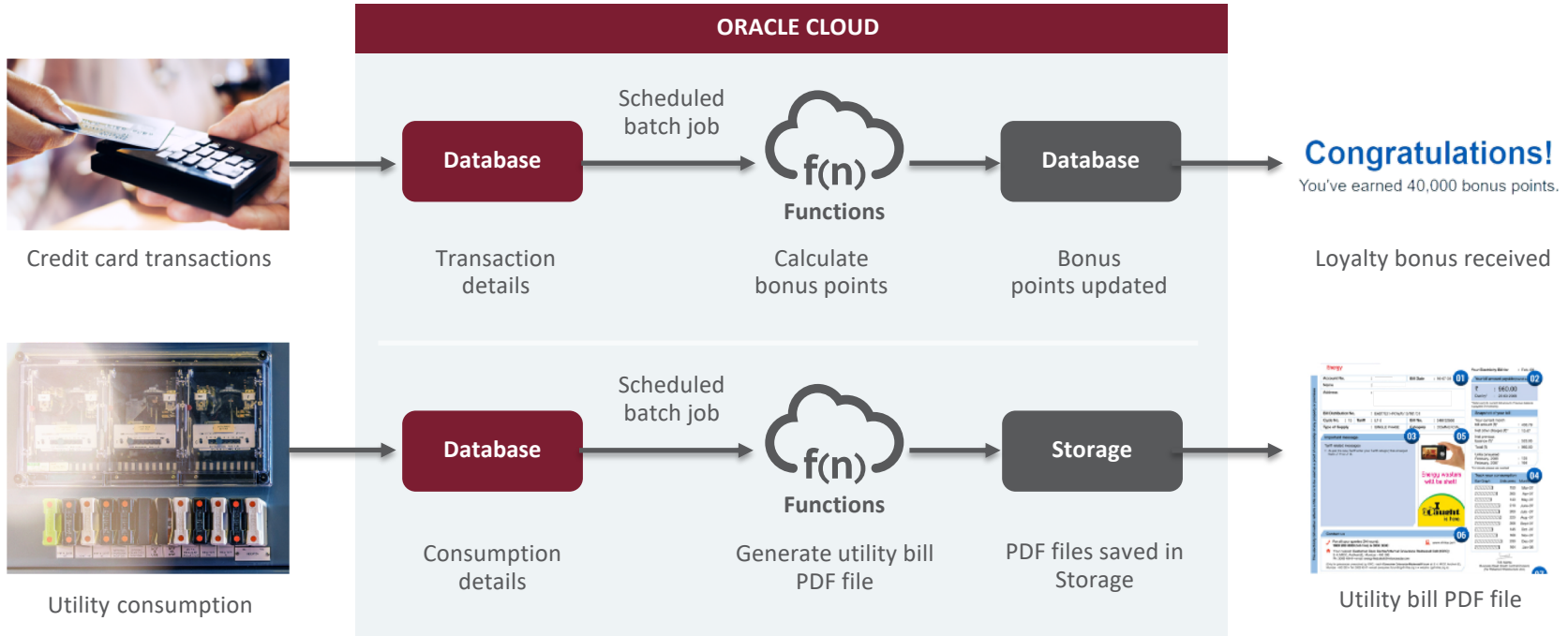
Use Case: Real-Time Stream Processing



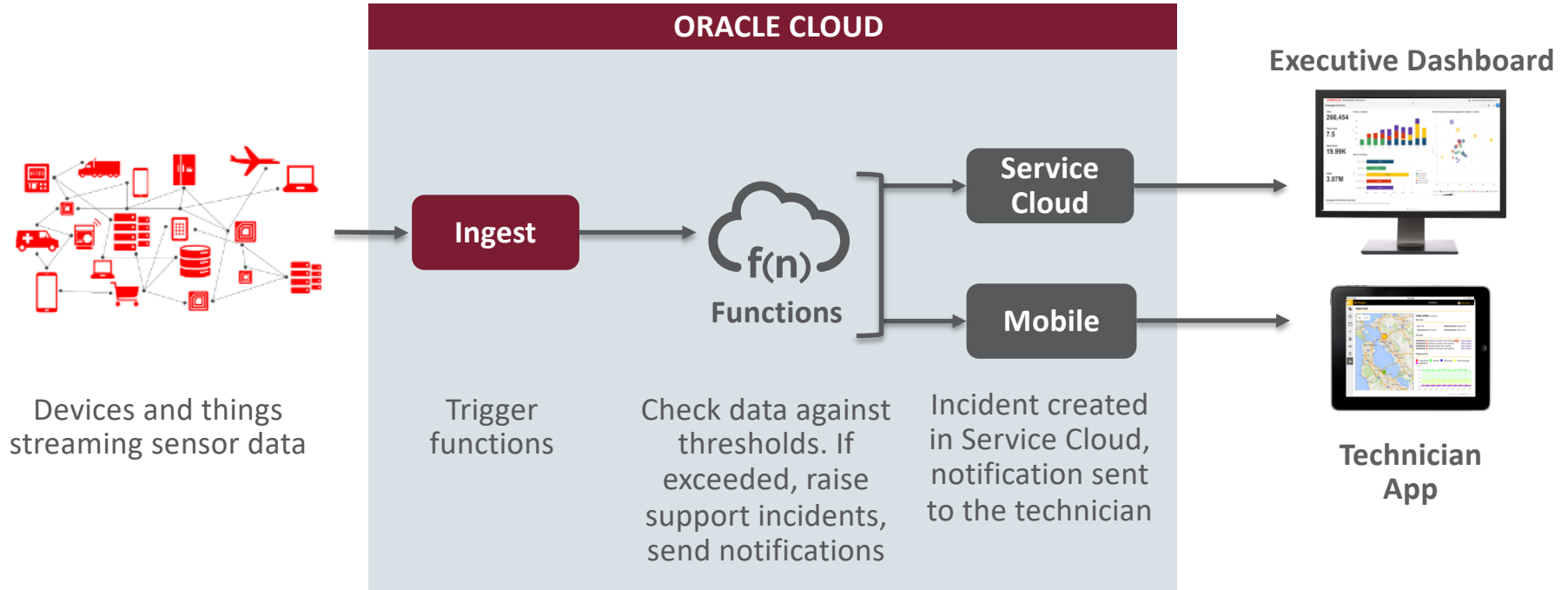
Use Case: Real-Time File Processing



Use Case: Batch Processing



Internet of Things



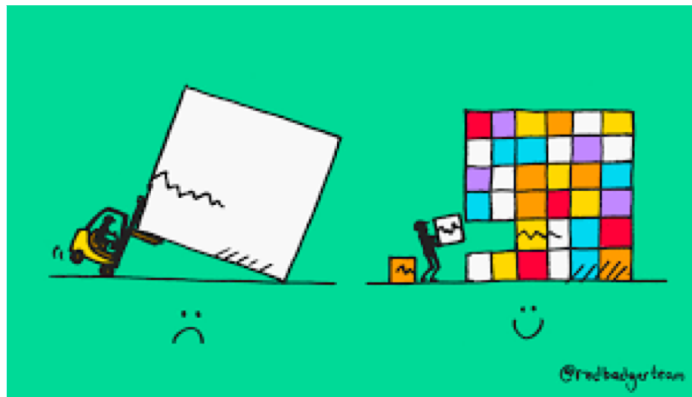
Mapping Execution, Events and Architecture

- Eventing is typically driven by context
- Execution model is an important choice



Choose a Pattern that Helps you Minimize

- Less is more: shoot for a single handler per module
- Better to proliferate than to decompensate
- Observability and triage become infinitely easier at the boundaries



Separate and Simplify

- Typically consumption of events define application boundaries
- Share libraries between functions and applications, not execution context



Worries

- Don't worry too much about cold starts
- Do worry about lock-in and data egress
- Pay attention to the system you are integrating with – keep it open



Dive In!

- The best way to learn is to do
- Often a POC becomes production
- You will not be alone
- Have fun!





ORACLE[®]
Cloud Native Labs

cloud.oracle.com/trial
cloudnative.oracle.com

Thanks!

 @jlb13