

Building Scalable NodeJS Apps on Google's Scalable Infrastructure for JuniorDev

Eric Jiang
@lorderikir

Junior Software Engineer (Backend/Infrastructure), Monash University



**Disclaimer: These slides are based off the
Training Module for our new Student
Developers**

Hi, I'm Eric Jiang



- I founded and currently am currently the Backend & Infrastructure Engineer at MonPlan (monplan.apps.monash.edu) ([Monash Course Planning Tool](#)) at Monash University
- Part of the [Student Innovation Team](#)
- I also co-founded and maintaining GeckoDM and MARIE.js
- Also have worked at Localz too!

Student Innovation Program at Monash University

Unfortunately this program is applicable to Current Monash Students only

- If you have an idea don't leave it in the dark, we would ❤️ to see it.
 - Test.Drive
 - Email me: Eric.Jiang@monash.edu or hit me up on the Junior Dev Slack (@lorderikir) and I'll redirect you to right person
 - We also hire Students for casual project-based positions too! 🎉

Talk Summary

- ① Introduction to Google Cloud
- ② What is Google App Engine
 - a. GAE Environments
 - b. What is Scaling and Why is it Important?
- ③ Deep-Dive
 - a. Deploying a simple API to Google App Engine
 - b. Automating Deployments and Testing
- ④ Other Cool GCP Products

GCP Products

PRICING DETAILS

Click below for pricing details on these products. Our [paid Support packages](#) offer additional technical help.



App Engine



BigQuery



Bigtable



Compute Engine



Container Engine



Cloud Dataflow



Cloud Dataproc



Cloud Datastore



Cloud Pub/Sub



Cloud DNS



Cloud SQL



Cloud Storage



Prediction API



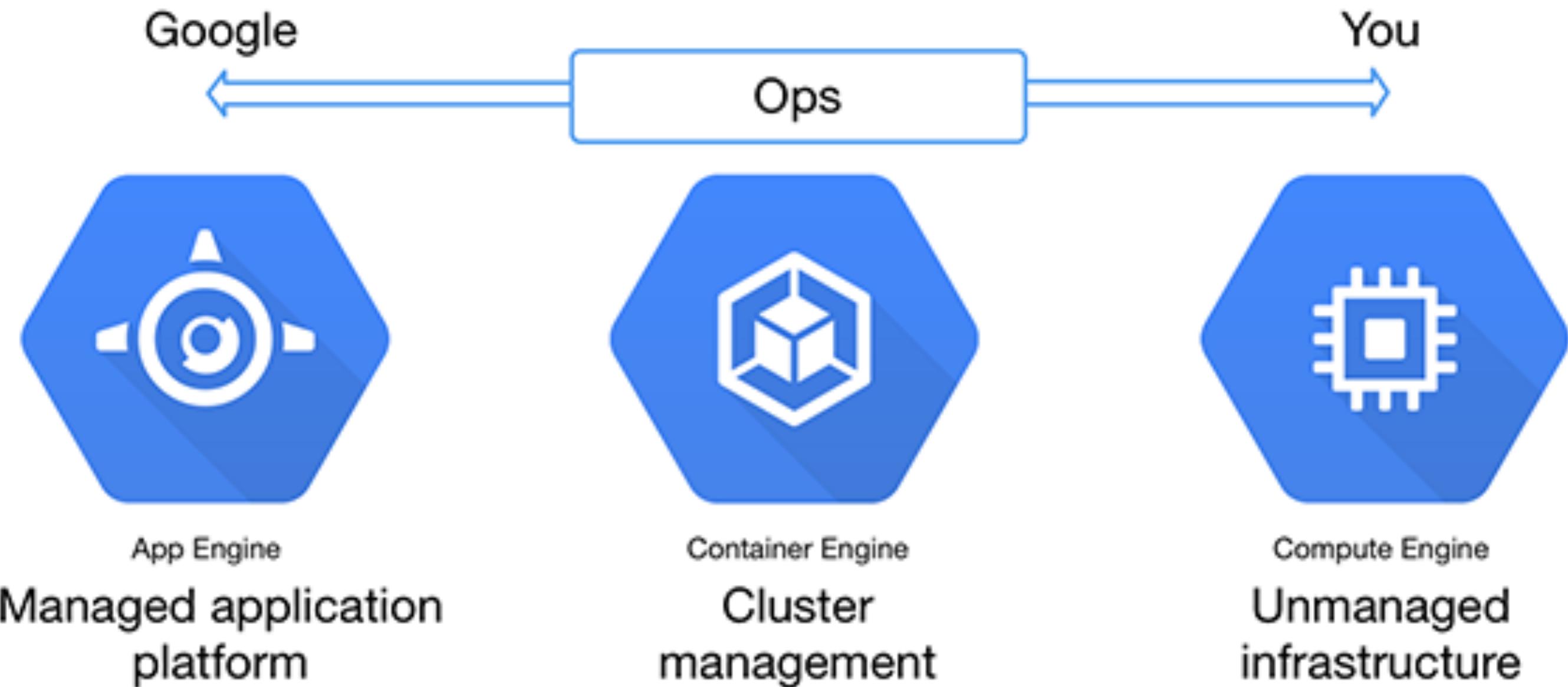
Translate API



Vision API



Support



Google App Engine

Language	Environment
Java 7 (and Kotlin ¹)	Standard
Java 8	Standard /Flexible
Node.js 8	Standard (Beta)
Node.js (>4)	Flexible
Python 2.7	Standard
Python 3.5	Flexible

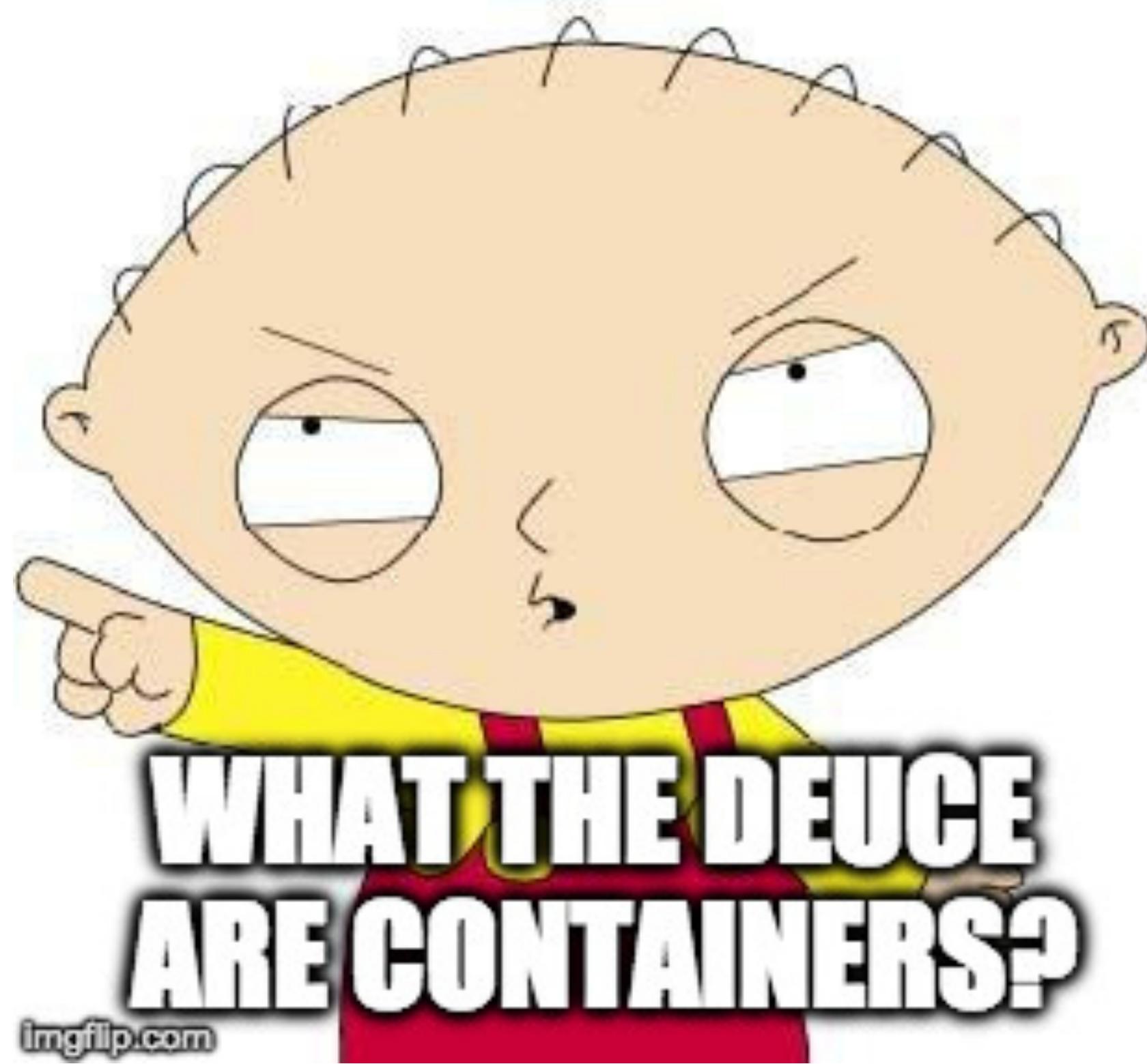
So what is the difference between the Environments?

Standard Environments run in a specialised environment. Though building the application is more constrained then other environments, it means that scaling up is faster.

Flexible Environment applications run off a Docker container, it is designed for applications that receive constant traffic.

When deployed they are Google Compute Engine as Virtual Machines Because they run off Docker, you can write your own Dockerfile Configuration to deploy, and deploy it anywhere, you can even move it to AWS.

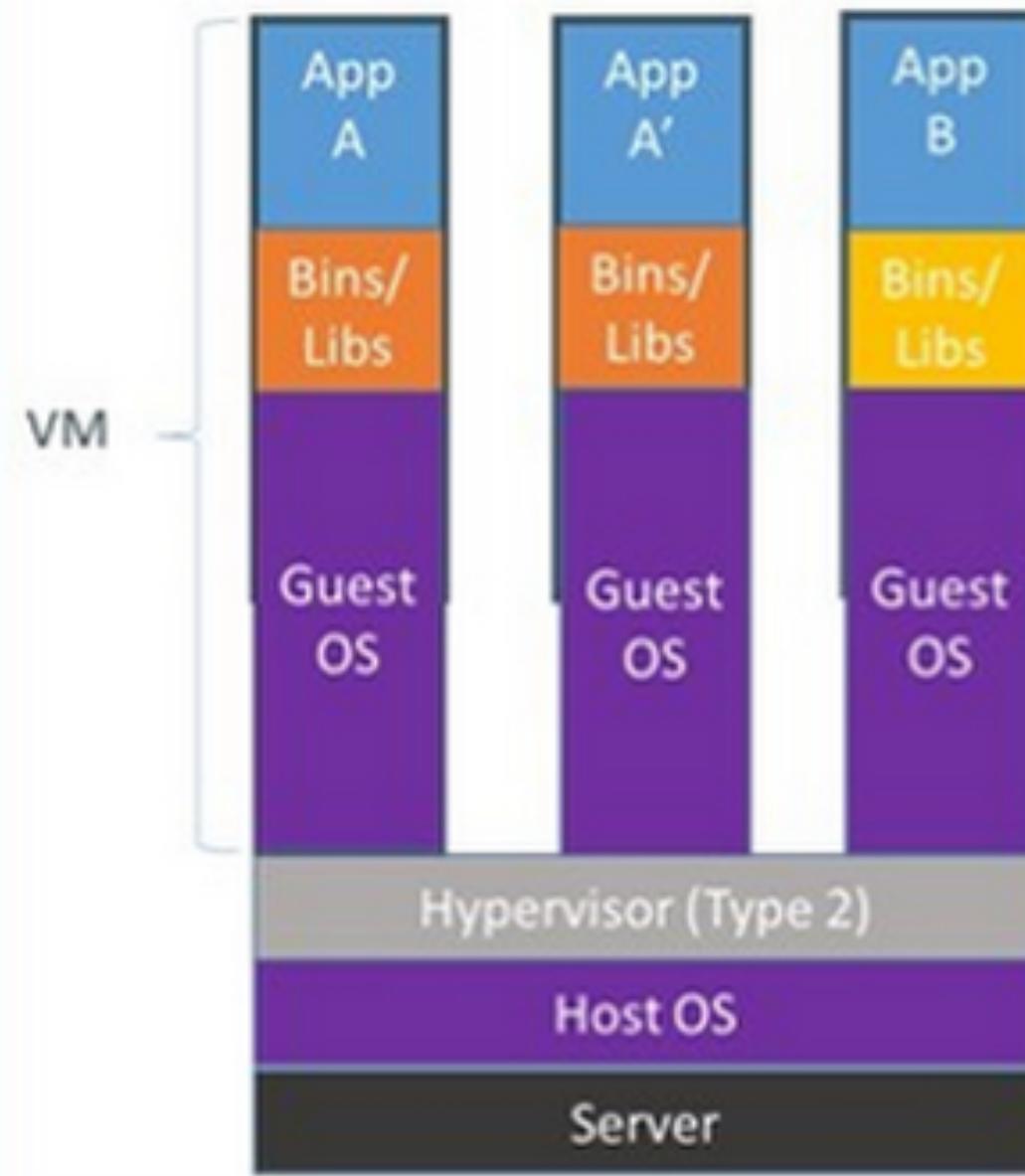
Note: all of these runs as containers...



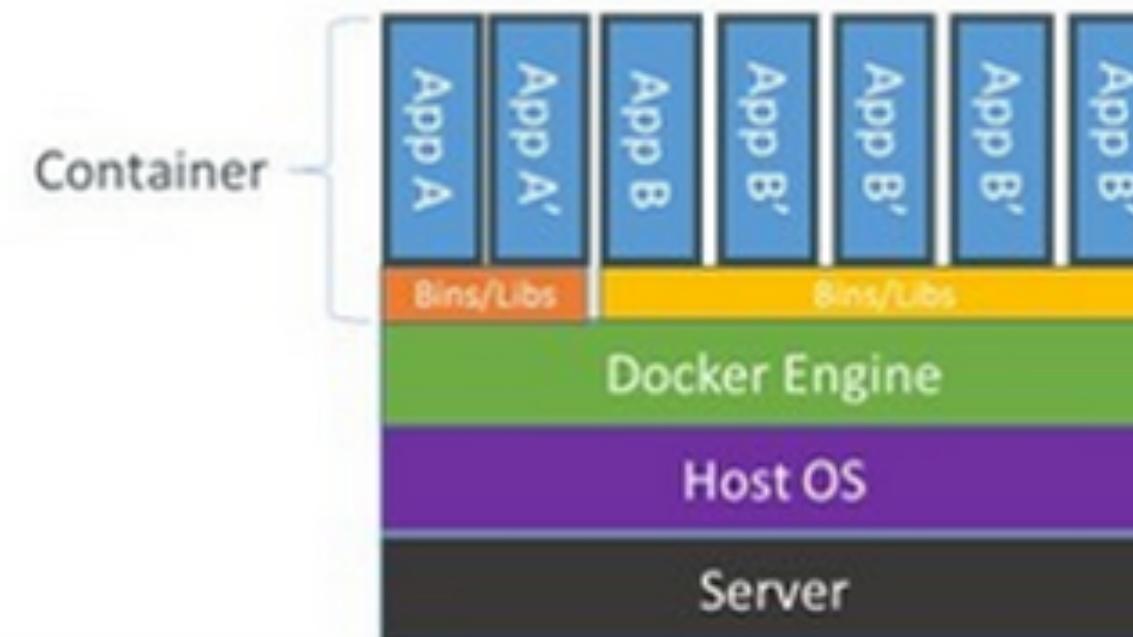
**WHAT THE DEUCE
ARE CONTAINERS?**

imgflip.com

Containers vs. VMs



Containers are isolated,
but share OS and, where
appropriate, bins/libraries



The Standard Environment has Instance Classes

“Each application running in the standard environment has an instance class, which determines its compute resources and pricing.”

Instance Class	Memory Limit	CPU Limit	\$/hr (Sydney)
F1 (default)	128 MB	600 MHz	\$0.068
F2	256 MB	1.2 GHz	\$0.135
F4	512 MB	2.4 GHz	\$0.270
F4_1G	1024 MB	2.4GHz	\$0.405

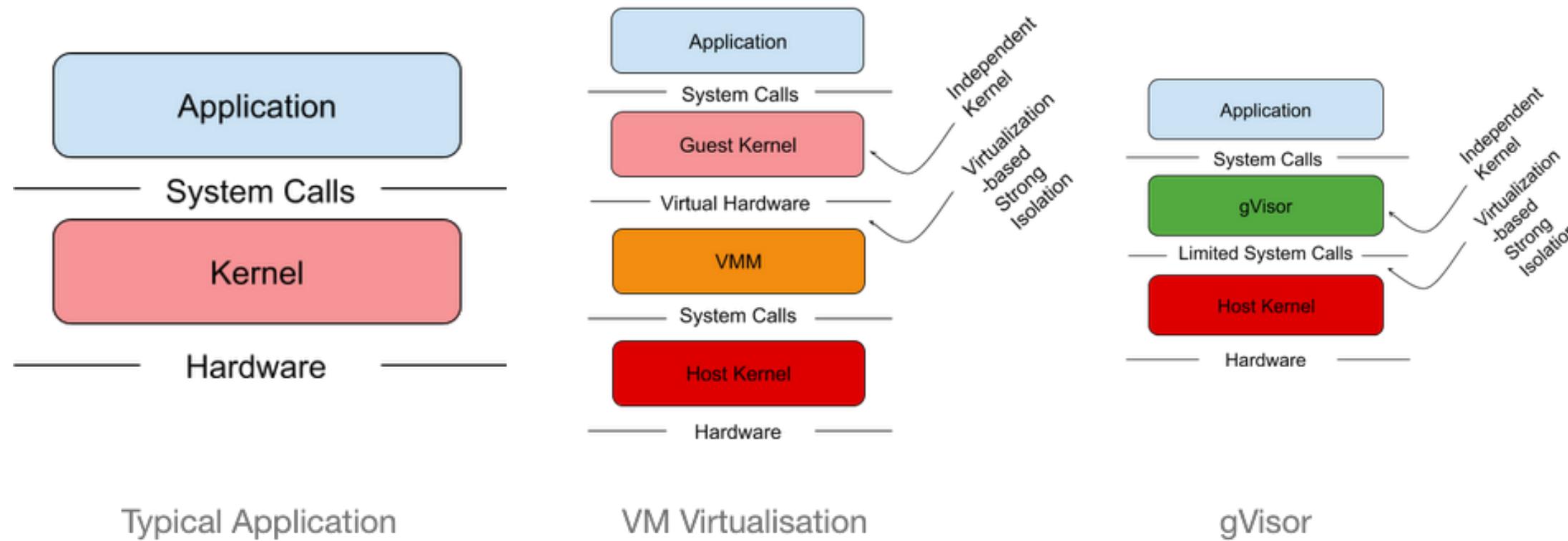
So Back then running NodeJS Application was Impossible on App Engine Standard



imgflip.com

Along came gVisor (which also pissed a lot of Docker Devs off)

“gVisor is more lightweight than a VM while maintaining a similar level of isolation. The core of gVisor is a kernel that runs as a normal, unprivileged process that supports most Linux system calls.”

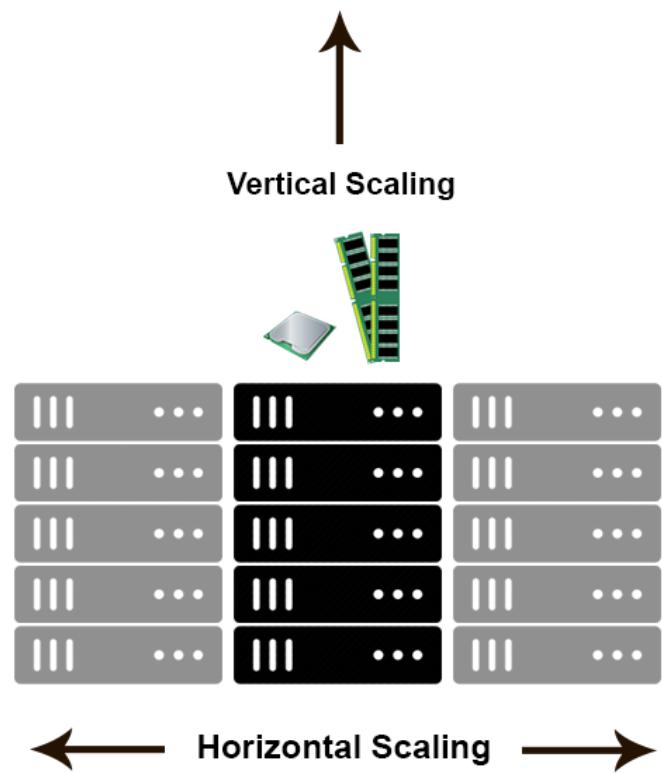


Scaling Modern Web Applications

Me when I look at Scaling:



Scaling Vertically vs Scaling Horizontally



Horizontal Scaling: scale by adding more machines into your pool of resources machine.

Vertical Scaling: scale by adding more power (CPU, RAM) to an existing machine.

Benefits of Horizontal Scaling

- Dynamic scaling allows spinning up more instances and nodes faster, i.e. if you suddenly get a influx of traffic
- Vertical Scaling is limited to capacity of resources, simply adding more resources
- Just simplying load testing isn't good enough, examples of this include Niantic (PokemonGo) and Australian Census 2016

Disclaimer: NodeJS on GAE is still in beta. USE AT YOUR OWN RISK



Time for a demo



Let's Deploy a simple expressJS Application

index.js:

```
const express = require('express');
const app = express();
const APP_PORT = process.env.port || 8080;

const exampleRouter = (req, res) => {
  res.send('hello world!');
}

app.get('/', (req, res) => res.send('Hello from Google App Engine!'));
app.use('/test', exampleRouter);
app.get('/teapot', (req, res) => {
  res.status(418).send("Hello I'm a teapot running on Node Standard GAE");
});

app.listen(APP_PORT, () =>
  console.log(`Example app listening on port ${APP_PORT}!`)
);
```

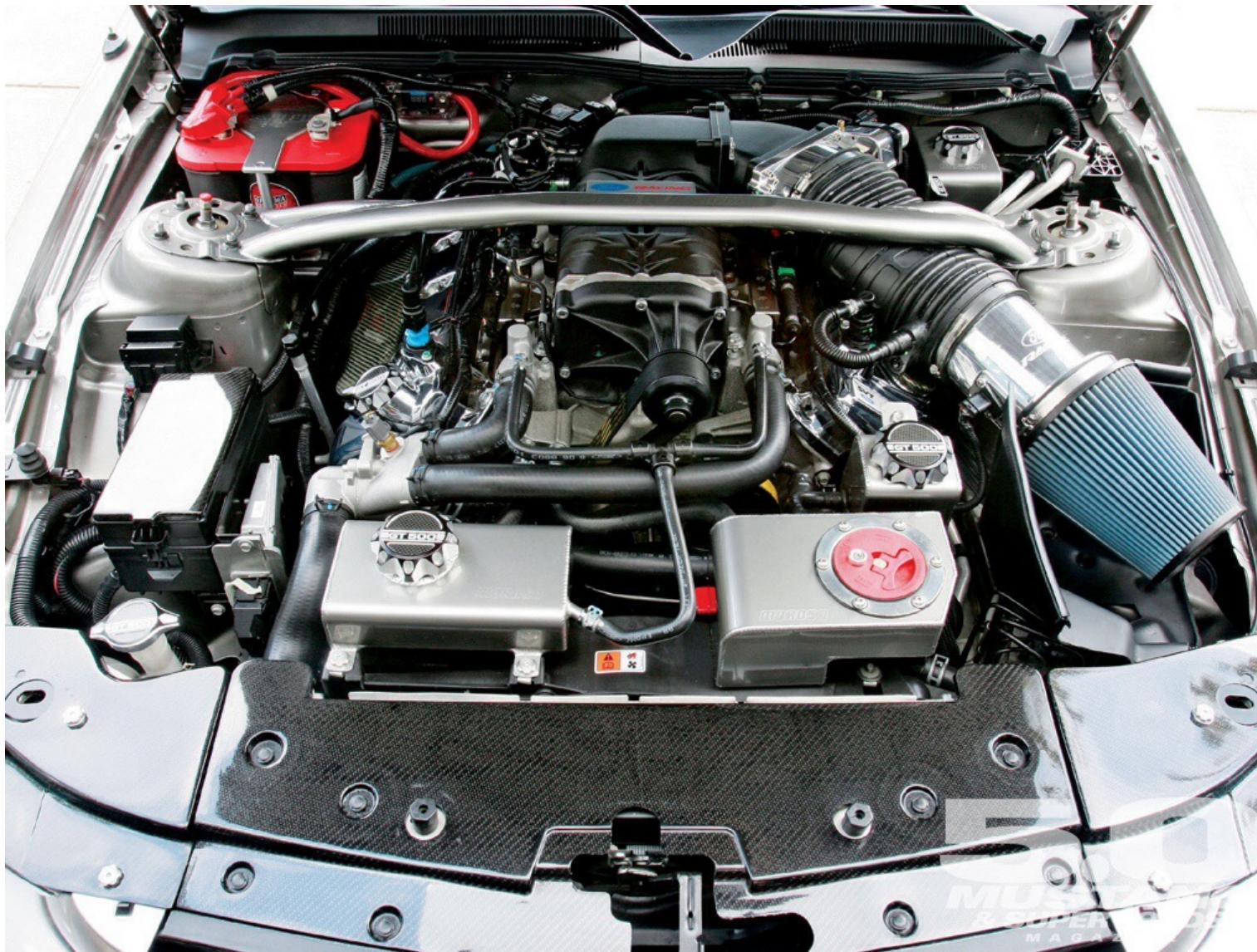
We need an app.yaml to help define the runtime settings on GAE

app.yaml:

runtime: nodejs8

Now we need to run gcloud app deploy

So what happens under the hood?



What happens when the Cloud SDK deploys to GAE

- ① Builds the Files (For the Java Envrioment this builds it into a WAR package)
- ② Bundles the Files and Pushes it Up to Google Cloud Storage
- ③ Reconfigures Google App Engine to deploy
 - i. Change scaling configuration
 - ii. reallocate traffic to new version

Automating Deployments

- Like a lot of environments deploy apps to GAE is really straightforward and easy
- All you need is the Google Cloud SDK and the simple command
`gcloud app deploy`
- However, you will need Service Accounts (I like to call them bot accounts) to automate this.
- You can find out more <https://s.lorderikir.me/2PxBtRJ>

Other Awesome Products on GCP

- Cloud ML (Google Cloud Machine Learning) which is built off TensorFlow
- Compute Engine - Google VMs
- BigQuery - Big Data (really awesome for Big Data analysis, companies like REA, Papercut uses this for Big Data analysis) - Our team uses it to provide Faculties and Business Units with reporting functionality.
- Kubernetes Engine - Containerisation! what else do you want?
- Cloud Storage - CDN provider of files (like [Amazon S3](#))
- Network Balancer - for Load Balancing of traffic for your applications
- Cloud ML APIs such as Natural Language Processing, Data Loss Prevention, etc.

Drag and drop | Cloud Vision X +

Uni Work

https://cloud.google.com/vision/docs/drag-and-drop

Google Cloud Why Google Products Solutions Pricing Security Documentation Customers Partners Support Marketplace

Contact sales

AI & Machine Learning Products

view the raw response.

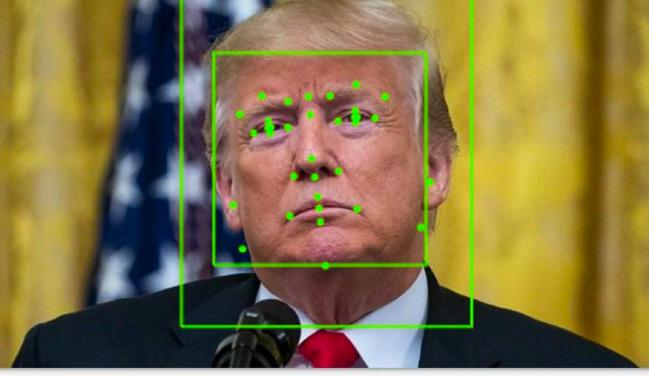
Cloud Vision API Product Overview Documentation

Quickstarts All Quickstarts Quickstart: Embedded API Explorer Quickstart: Drag and Drop

How-to Guides All How-to Guides Enable the API Authenticate Make a Request Detect Text (OCR) Detect Labels PDF/TIFF Document Text Detection Detect Other Features Beta Features Base64 Encode

APIs & Reference All APIs & References Client Libraries Migrating to Python Client Library v0.25.1

Faces Labels Web Properties Safe Search JSON

_103104631_048761837.jpg

Joy	█	Very Unlikely			
Sorrow	█	Unlikely			
Anger	█	Very Unlikely			
Surprise	█	Very Unlikely			
Exposed	█	Very Unlikely			
Blurred	█	Very Unlikely			
Headwear	█	Very Unlikely			
Roll:	-5°	Tilt:	16°	Pan:	-6°
Confidence	90%				

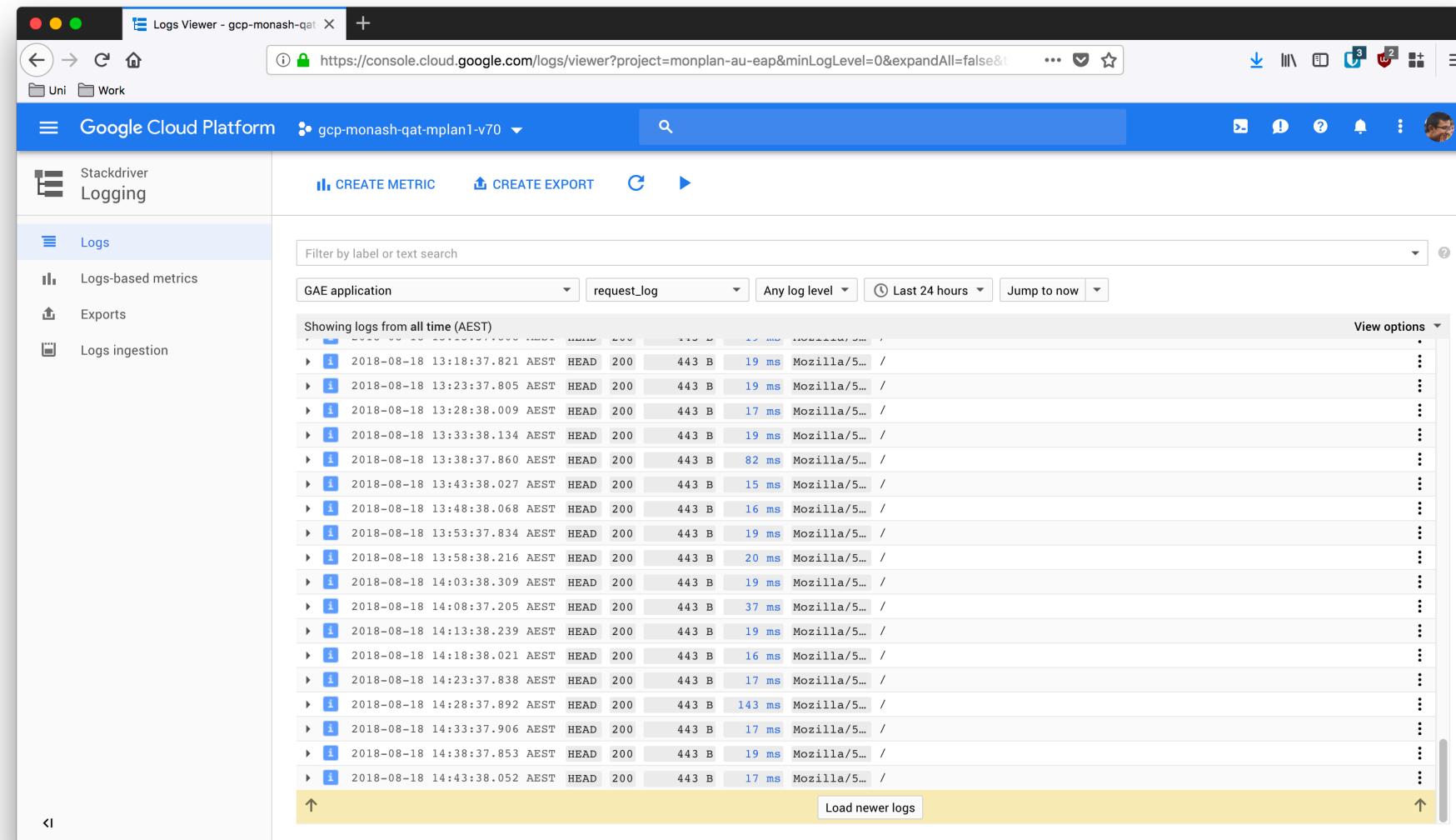
Was this page helpful? Let us know how we did:
☆☆☆☆☆

SEND FEEDBACK

So what happens if something goes wrong?



In comes, StackDriver Logging



So what is Stackdriver Logging?

“Stackdriver Logging allows you to store, search, analyze, monitor, and alert on log data and events from Google Cloud Platform and Amazon Web Services (AWS)”

Stackdriver Logging is baked into Google App Engine applications

Questions? 🤔 🎤 📣

and the chance to get some free stickers?

Special thanks to:

- Ben Theunissen, REA Group
- Hugo Muller-Downing, Localz

Goodbye  and thanks for
listening 

Eric Jiang

Twitter: @lorderikir

GitHub: lorderikir

If you want to talk more about what I do or what is the Student Innovation Team hit me up!