Problem Statement

Imagine you are one of two founders of a start-up that specializes in document processing, you want to build a general document-form understanding API that your future customers can use to automate their paper forms. You don't know how many customers you will have nor how many requests per day. But some market research has shown that your potential customers can process thousands of forms a day. Knowing the rough requirements, you decided to use cloud services for the flexibility to scale up and down.

- *For the model, being a resource constrained start-up, your first instinct is to use some pre-trained models online to get yourself started, and build the best-in-class model once you have collected enough data from your customers.*

- *The other founder is skeptical of the efficiency of your approaches and suggests to use some over-the-counter solutions from big tech companies. Your job is to convince the partner that your solution is cost and scaling efficient by building a prototype that demonstrates the (potential) ability to provide your services at your customers' scales. Of course you will also need to explain the plan once you are done.*

How it’s all done

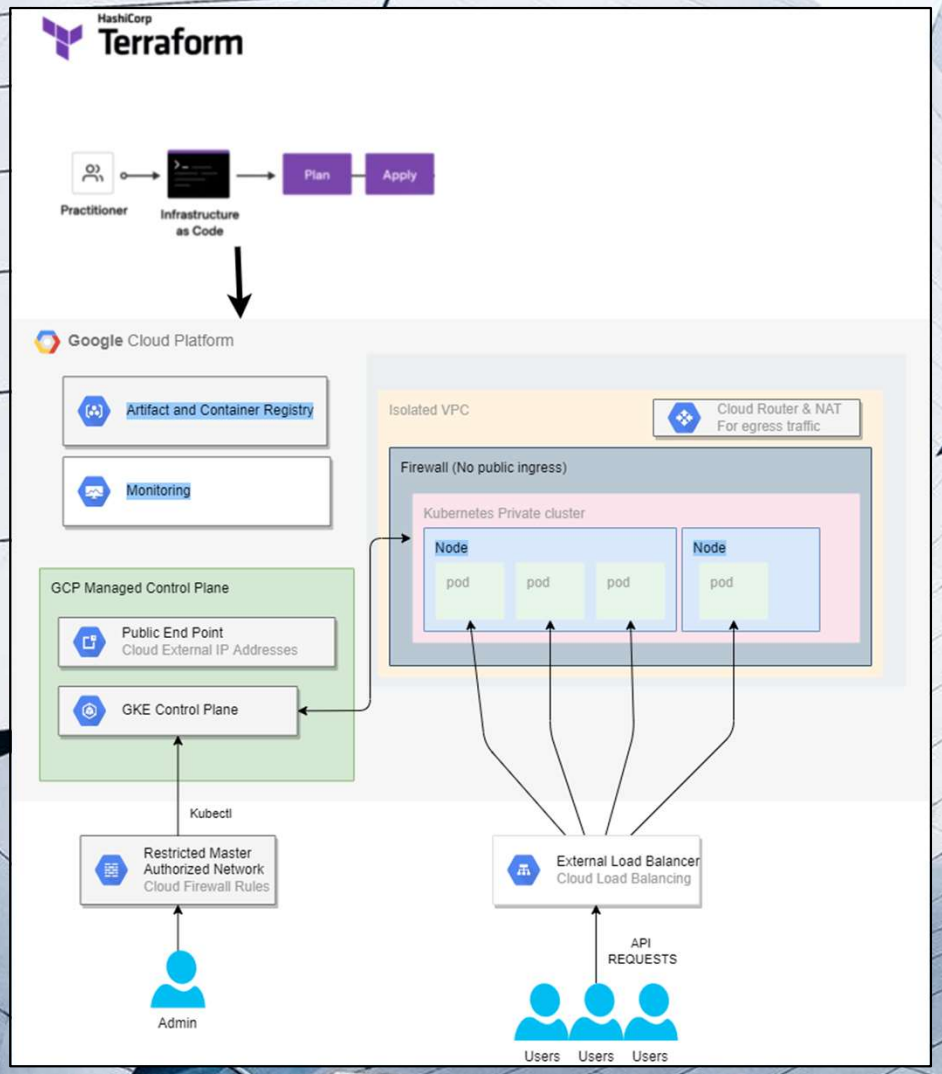
Implementation Flow

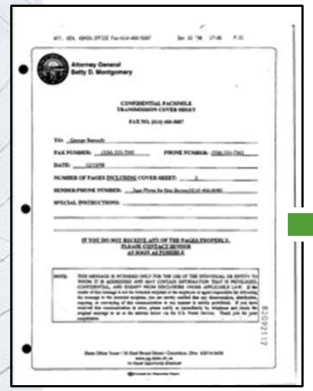
Priorities based on business needs:

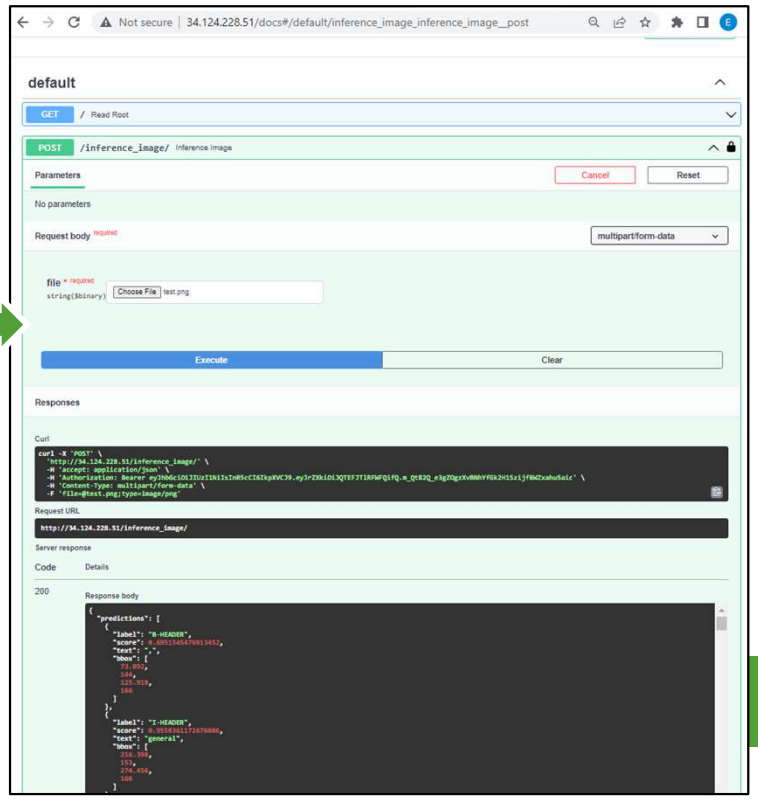
1. **Cost efficient and highly scalable solution**
2. Product to be minimally viable in the interest of time

Infrastructure	Model Serving App	Deployment & Scaling	Testing
Google Cloud – Offers managed GKE with built in monitoring capabilites.	**Fast API** - Ease of setting up API endpoint with authentication and auto docs	**Docker** – For packaging app and reproducibility across environments	**Locust** – A modern web load testing tool capable of simulating millions of simultaneuous users.
Terraform – to provision resources using IAAC approach	**Hugging Face** – Model available with inference codes	**Kubernetes** – Auto scalability of resources based on demand	

Architecture Diagram & API







Security

All API endpoints, internal or external facing must be secured.

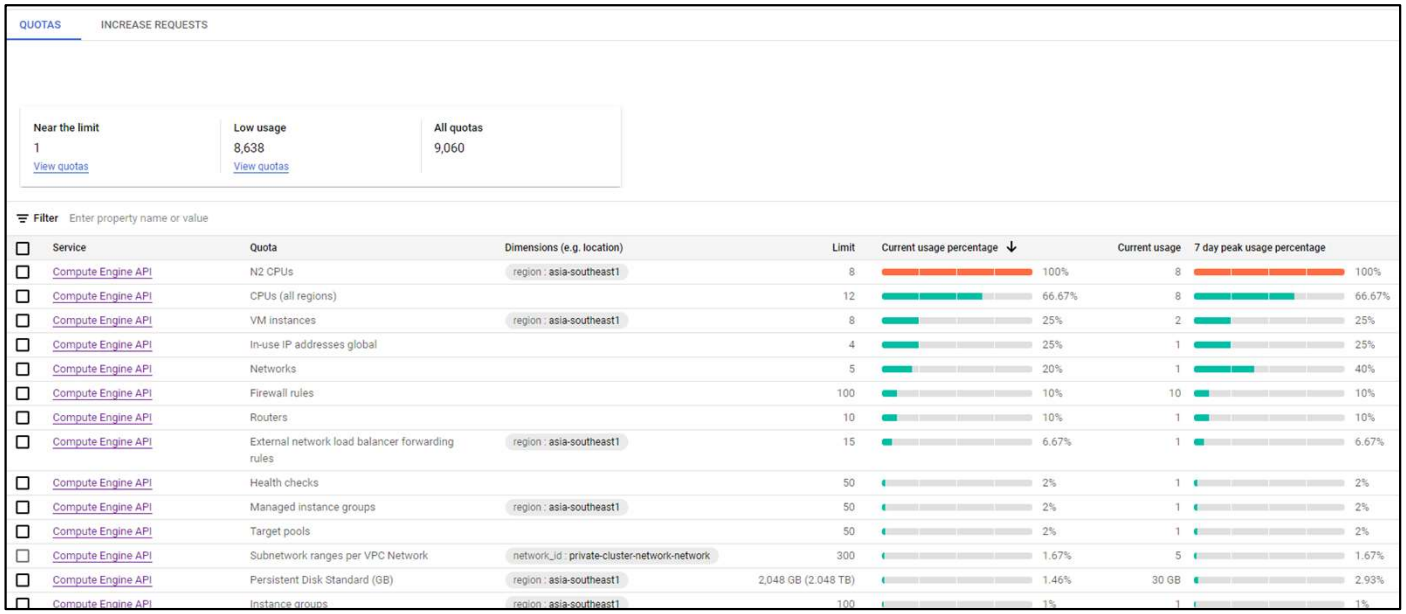
There are a few important architecture decisions that was made for cyber-security:

- Infrastructure as code (IAAC) using Terraform
  - Mitigate human click ops error, can be versioned and roll back to a “safe” state
- Private K8s Cluster (Hosted in Isolated VPC)
  - Limiting cyber attack surface. Control plane end point is restricted to admin IP.
- Services are exposed via external load balancer
  - Routing logic, single point of ingress. Traffic patterns can be monitored easily
- VPC Firewall rules setup to block all public inbound traffic
  - Watertight VPC
- FastAPI Endpoint secure with Authentication Token
  - Only people with valid token can access and make API call.

Limitation

Due to basic plan on GCP. **Quotas cannot be lifted in due time**.
Project were worked on **constraints of max 8 vCPUs**. While model latency is better when served on instance with more vCPU (achieved ~ 1000ms per request on 8vCPUs), the actual deployment is configure with 1 node (N2-HighCPU-8) and autoscaling pods of max 1.5vCPU & 1GB RAM.

This is done per above, so that the autoscaling capability can be demonstrated.



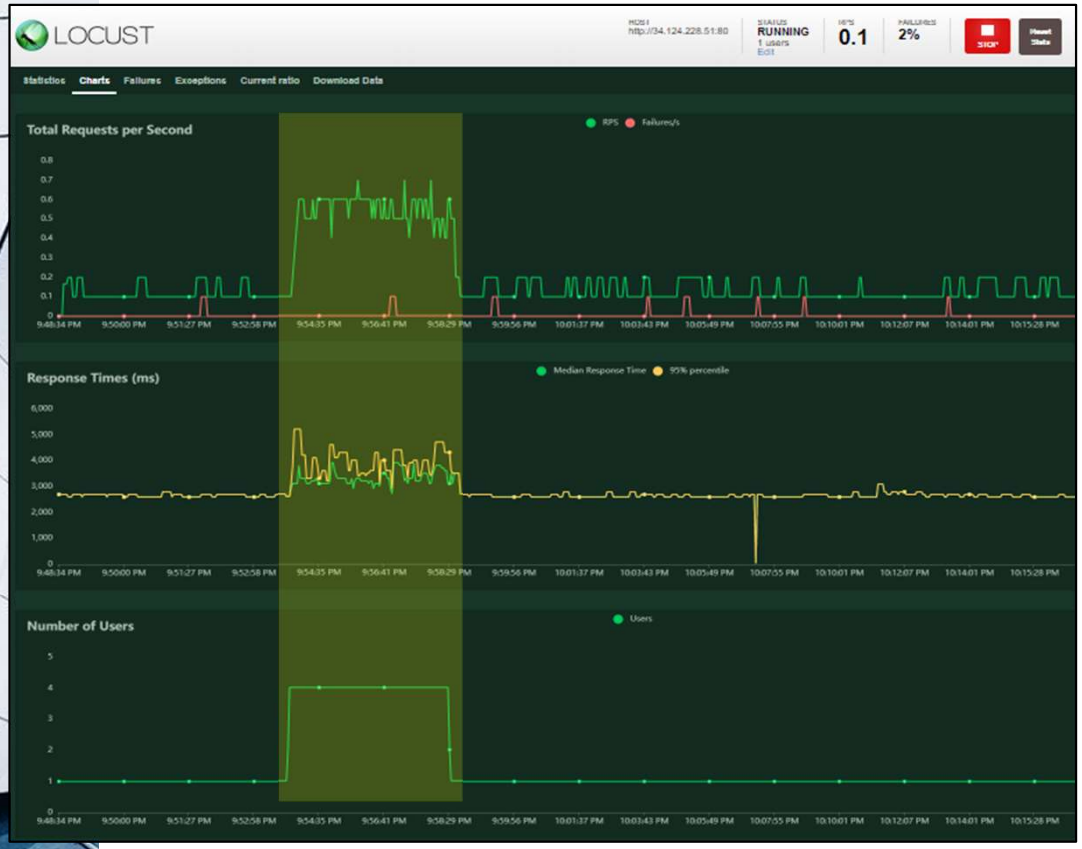
QUOTAS | INCREASE REQUESTS

Near the limit	Low usage	All quotas
1	8,638	9,060
View quotas	View quotas	

Filter: Enter property name or value

Service	Quota	Dimensions (e.g. location)	Limit	Current usage percentage	Current usage	7 day peak usage percentage
Compute Engine API	N2 CPUs	region : asia-southeast1	8	100%	8	100%
Compute Engine API	CPUs (all regions)		12	66.67%	8	66.67%
Compute Engine API	VM instances	region : asia-southeast1	8	25%	2	25%
Compute Engine API	In-use IP addresses global		4	25%	1	25%
Compute Engine API	Networks		5	20%	1	40%
Compute Engine API	Firewall rules		100	10%	10	10%
Compute Engine API	Routers		10	10%	1	10%
Compute Engine API	External network load balancer forwarding rules	region : asia-southeast1	15	6.67%	1	6.67%
Compute Engine API	Health checks		50	2%	1	2%
Compute Engine API	Managed instance groups	region : asia-southeast1	50	2%	1	2%
Compute Engine API	Target pools		50	2%	1	2%
Compute Engine API	Subnetwork ranges per VPC Network	network_id : private-cluster-network-network	300	1.67%	5	1.67%
Compute Engine API	Persistent Disk Standard (GB)	region : asia-southeast1	2,048 GB (2.048 TB)	1.46%	30 GB	2.93%
Compute Engine API	Instance groups	region : asia-southeast1	100	1%	1	1%

Validation & Testing



- GKE cluster with only 1 node
- Pods configure to request for 1.5 CPU and 1GB Ram
- Spike in traffic simulated at 9:53pm to 9:58pm

Result:
Pods observed to auto scale up to maximum 4 pods throughout the spike and scale down back to 1 pod shortly after.

More realistic simulation cannot be done due to GCP quotas as mentioned in slide 6.

Projected Cost (Conservative)

Overall Infra Cost for 5 Nodes GKE

Region: asia-east2 (Hong Kong)

Service Name	Description	Unit Cost (USD)	Total SKU/ Month (USD)	Projected Cost/ Month (USD)
Network	Egress cost for API responses (Egress between Google Cloud regions within Asia (per GB))	$0.05/GB	100 GB	5.00
Storage	Artifact & Model Registry (Model + Code Docker Image)	$0.023/GB	5 GB	0.12
Kubernetes Engine Standard	Managed GKE	$0.10/hr	1 cluster	72.00
Compute Engine	Preemptible N1-highcpu-8 Assume Autoscaling Node Pool (1-5)	$0.0835/hr	5 node	300.60
Persistent Disk	PD Standard Attached to Nodes (30GB per node)	$0.0498/GB	150 GB	537.84
HTTP Load Balancing	Load balancer forwarding rule minimum charge	$0.035/hr	1 rule	25.20
*Assumption on a 5 Nodes Cluster running on full capacity all the time			**Total**	**940.76**

Breakdown Cost / 1000 API Calls

Image Size	Latency	1 Node (N1-HighCPU-8)	5 Nodes (N1-HighCPU-8)		
		API Capacity / Day	API Capacity / Day	API Capacity / Month	USD Cost / 1000 APIs
300DPI (2480 x 3508)	5 Seconds	288 calls/day	1440 calls/day	43k	**21.88**
200DPI (1650 x 2340)	3 Seconds	480 calls/day	2400 calls/day	72k	**13.07**
72DPI (595 x 842)	1 Second	1440 calls/day	7200 calls/day	216k	**4.36**

* Latency can be improved with better compute engines and accelerators

** Typical document image is ~72 DPI. However, preprocessing/resizing to 300DPI will yield more accurate result

1 Node is always on in GKE regardless. However, this is not a concern since **customer potentially process thousands of pages in a day**

Preemptible nodes save 80% of the compute cost. As long as our node pool are configured for multi nodes, that should be fairly robust. If availability is absolutely a concern, we can have a smaller dedicated node pool in addition of the preemptible pool.

Competitor Analysis

aws  **Amazon Textract**

Equivalent option:
USD 50 / 1000 pages

Analyze Document API		First million pages in a month
Queries	Per 1,000 Pages	$15.00
Tables	Per 1,000 Pages	$15.00
Tables + Queries	Per 1,000 Pages	$20.00
Forms	Per 1,000 Pages	$50.00

Azure  **Form Recognizer**

Equivalent option:
USD 10 / 1000 pages

Document type	Price
All	0 - 500 pages free per month
Customised	$50 per 1,000 pages
Pre-built: Layout, Receipt, Business Card, ID	$10 per 1,000 pages

*For comparison, we assume self-hosted model accuracy performance is on par with competitor

Self Hosted Option

Google Cloud · Terraform · FastAPI · HUGGING FACE

Before optimization
Min USD 4.36/1000 pages (72DPI)
Max USD 21.88/1000 pages (300DPI)

After optimization
At least 25% reduction

With self hosted option, we are able to **customize** our model without additional cost. While Azure will only allow us to do so with **5x** the base price!

Future Enhancement

Infrastructure

Improve cost, throughput and latency using better nodes & accelerator. Here's how?

Add Cloud Armor to deter DDOS or bots

Deployment

Setup git actions for auto deployment

Provision dev, stg, prod environments

Use specialized framework for model serving. Here's Why

Use a more performant OCR backend. LayoutLMV defaults to Tesseract.

Model Serving App

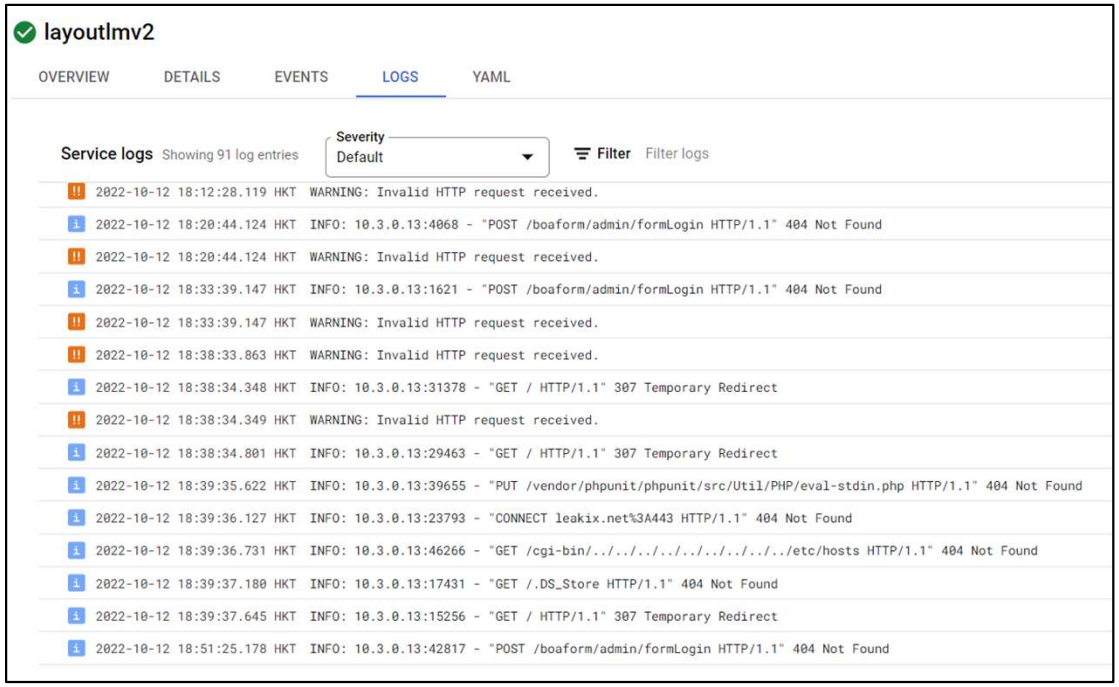
Add unit test for all methods.

Curate relevant metrics and add alerts

Testing & Monitoring

Evidence that security is TOP concern

Request traffics of bot searching for vulnerability



Mitigation/Prevention

- Use of Cloud Armor / WAF between public traffic and Load Balancer
- Only allow ingress traffic from designated clients' IP ranges

Qualification of Requirements

Some requirements for your solution:

- *An architectural diagram*
  Provided in slide 4
- *Cloud based (doesn't matter which cloud)*
  Using Google Cloud Platform
- *Storage for your data and model*
  Model and Code versioned on container registry. We can possible host the models files separately on cloud storage or a model registry of some form in the future.
- *Services deployed on K8s cluster*
  Deployed on Google Kubernetes Engine provisioned using Terraform
- *There's no model performance requirement, but you should have some mechanisms to make it better in the future.*
  Future enhancement provided in slide 10
- *An API for your customers to try*
  Provided in slide 12.