# **Project Overview**

## **Existing condition**

Marketview is an application **receive tick data/market data** from **six telekuers vendor**. The application **receives the data, format the data and then publish the data to end users**. The end

***(Do we need to use data streaming and ETL?)***

user see what is the market price of a given stock using their mobile phone or through internet application. The application is currently hosted in an on-prem environment.

The six telekuers feed will be delivered to an application hosted on on-prem. But a decision has been taken to migrate the Marketview application from on-prem environment to any public cloud. You have been assigned as a solution architect and cloud developer to architect, design, and deliver this migration.

The application is a 3-tier application. The end-users are spread across various countries in a particular region (say EMEA but you can replicate to any region of your choice).

## **Project Goal**

* Design and define overall **VPC, CIDR planning, subnetting and IP addressing**.
* Design a secure, scalable, and **highly available network architecture** using any public cloud to support application connectivity and latency requirements.
* Use **Terraform** for underlying infrastructure setup requirement
* Setup the production environment using **auto scaling**, **load balancers**, **bastion hosts**
* Use appropriate connectivity to connect to **on-prem to cloud** but it has to be cost effective
* Setup appropriate **logging and monitoring**
* Setup **identity and access management** and ensure proper access control mechanism in place
* Setup required **Public/Private subnets for web, app, database servers**
* Setup auto scaling group for web and app (define the **baseline requirement** as a minimum **how many vm’s** will be required (the peak time users accessing application is morning and evening)
* Ensure necessary **firewalls** in place

## **Project Assumptions**

* We assume Marketview is not an **in-house design application**. It’s a third party applications. There for we we need to minimize changing the code on the marketview application.
* For the current project there is **no requirement to change the application structure** (I.e change from exsisting condition to container based application) this is because on project goal there is no mention of containers but instead vm is mentioned.
* There is a single environment that required (production-only) as stated in the project Goal
* Cloud-babies.com used internal Active directory for user provisioning. And would like to enhance this for GCP cloud user
* Marketview app is installed in RHEL OS v.8 and using MYSQL as the database.

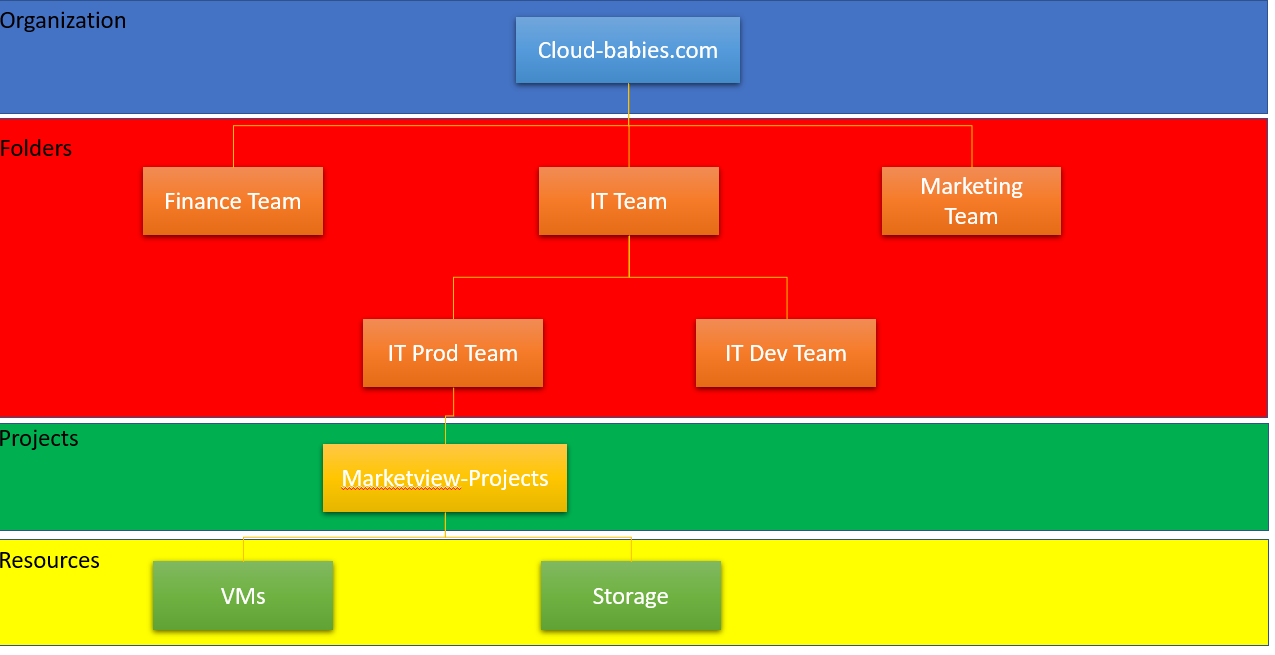
# **GCP SYSTEM DESIGN AND CONSIDERATION**

## **Geographic Zones and regions.**

For this project we asssme that this project we only used **single region** with **multiple AZ**. However we will consider the use of other region forAvailability purposes only. For the region we choose EMEA region with multiple zones. This is selected because it closed to majority of customer served by marketview application.

## **Resource Management**

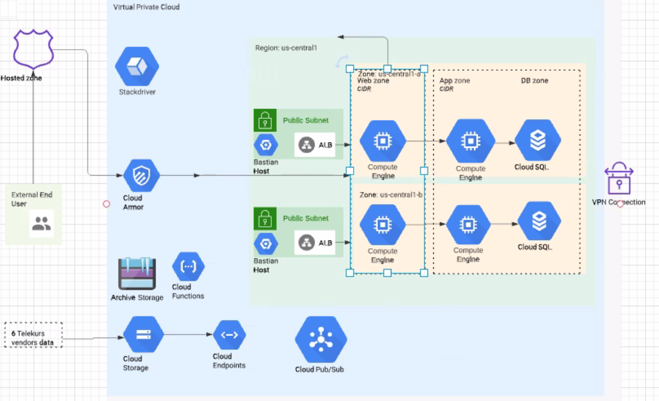
Google resource management consist of organization, folders and project that allow us to group and hierarchically organize GCP. For this project, we assume this is only for single organization, single folder and single project for marketview application. This is because we want to group all resource for Marketview into a single project to make it less complex and a single management of all the resource for marketview application



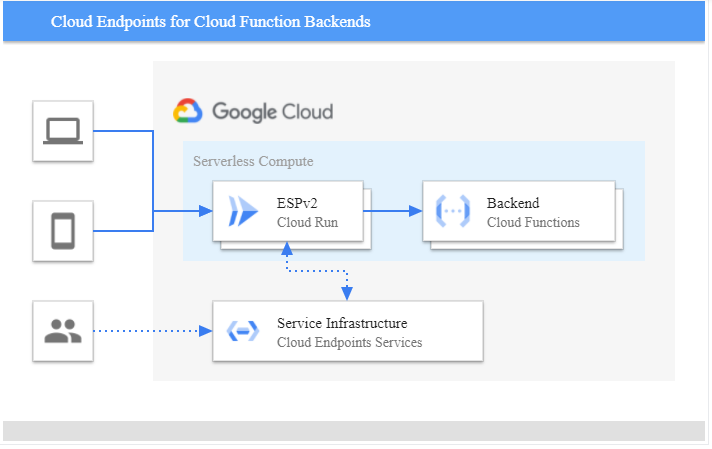
For this project, we used cloud-babies.com as our org. we also had multiple folder to support the Marketview application such as finance, IT and Marketing team. For IT Team we plit to IT prod and IT Dev. For the projects, currently we decice to choose a single project for all the Marketview application this is to make it simple and build up another projects as needed. This is where all the GCP resources will be located.

## **Project topology**

The picture below is the topology that we recommended for this scenario.



For this project we are going to use a single project, and single VPC. We are using multiple zone with a couple if CIDE which is Web CIDR, App CIDR and DB CIDR. For internal connection we are using VPN connection. For external connection we are using Public subnet where our ALB and bastion host reside. Telekuers vendor will connect to cloud storage where the cloud endpoint will treiger the aplication server when market data is received in cloud storage by the telekuers vendors. Cloud endpoint will act as API Gateway for the Cloud Function to provide API Management.



Taken from : <https://cloud.google.com/endpoints/docs/openapi/get-started-cloud-functions>

## **Identity and Access Management**

### **Google Account**

For google account, we recommended to use a fully managed google accounts that tied to exsisting corporate account using **Cloud Identity.** Since org want to integrate with existing active directory, we are using **Google Cloud Directory Sync (GCDS)** for user provisioning. We are using **free edition** of cloud editity for this project to avoid any billable GCP components. GDCS will be run on-premises and on separate server from domain controller which is domain-joined.

We also required 6 extrenal google account required for the 6 telekuers vendor to publish their data to our existing GCP infra.

### **Service Account**

A service account is **an identity that Google Cloud can use to run API requests on your behalf**. In the context of Compute Engine, this identity is used to identify apps running on your virtual machine instances to other Google Cloud services.

As for the best practice is to minimize the service account (one for each LOB). We will try to minimize the use of service account and to keep track of who or what can impersonate these service account. Default service account will not be used.

### **Google Group**

A Google group is a named collection of Google accounts and service accounts. For this project the group that we need to create is for monitor the application

* + 1. **gcp-org-admins**

Responsible for organizing the resource structure for the organization

* + 1. **gcp-network-admins**

Responsible for creating networks, subnets, firewall rules, and network resource such as cloud router, Cloud VPN, Etc

* + 1. **gcp-security -admins**

Responsible for managing security policies for the organization including access management

* + 1. **gcp-billing-admins**

Responsible for setting up billing account and monitoring the cloud resource usage.

* + 1. **gcp-devops-admins**

Devops practicioner for create and managing end-to-end pipelines cof CI/CD and monitoring andand system provision

* + 1. **gcp-developer**

Responsible for designing, coding and testing application.

### **Google Workspace Domain**

**Google Identity Domain**

## **Google Cloud Platform Authorization**

For authorization, it is best practice not to create IAM Member with service account user and service account creator at project level. And it is adviseable to login using corporate account/credential instead of gmail account. We also

### **Resources**

You can grant access to users for a Google Cloud resource. Some examples of resources are projects, Compute Engine instances, Cloud Storage buckets, and so on. Some services, such as Compute Engine and Cloud Storage, support granting IAM permissions at a granularity finer than the project level.

### **Roles**

There are three kinds of roles in IAM:

* **Basic roles.** Owner, Editor, and Viewer.
* **Predefined roles.** Predefined roles are IAM roles that give finer-grained access control than basic roles.
* **Custom roles.** Roles that you create to tailor permissions to the needs of your organization when predefined roles don't meet your needs.

### **IAM Policies**

You can grant roles to users by creating an IAM policy, which is a collection of statements that define who has what type of access. A policy is attached to a resource and is used to enforce access control whenever that resource is accessed. An IAM policy is represented by the IAM policy object.

### **Policy Hierarchy**

You can set an IAM policy at any level in the resource hierarchy: organization, folder, project, or the resource level. Resources inherit the policies of their parent resource. Set a policy at the organization level to have it automatically inherited by all its children folders and projects. Set a policy at the project level to have it inherited by all the project's child resources. The effective policy for a resource is the union of the policy set at that resource, and the policy inherited from higher up in the hierarchy.

## **Google Cloud Platform Compute**

Few best practice that we implemented for the Compute engine are:

* Ensure that Google Cloud VM instances are not using public IP addresses.
* Ensure that "On Host Maintenance" configuration setting is set to "Migrate" for all VM instances.
* Ensure that the Auto-Delete feature is disabled for the disks attached to your VM instances
* Ensure that IP Forwarding is not enabled for your Google Cloud virtual machine (VM) instances.
* Ensure that interactive serial console support is not enabled for your Google Cloud instances.
* Ensure that your production Google Cloud virtual machine instances are not preemptible.
* Ensure that project-wide SSH keys are not used to access your Google Cloud VM instances.
* Ensure that Shielded VM feature is enabled for your virtual machine (VM) instances.
* Ensure that automatic restart is enabled for your Google Cloud virtual machine (VM) instances.
* Ensure that deletion protection is enabled for your Google Cloud virtual machine (VM) instances.
* Ensure that your Google Cloud instance groups are using autohealing to proactively replace failing instances.
* Ensure that OS Login feature is enabled for your Google Cloud projects.
* Ensure that your virtual machine (VM) instance disks are encrypted with CSEKs.
* Ensure that your virtual machine (VM) instance disks are encrypted using Customer-Managed Keys (CMKs).
* Ensure that Google App Engine applications enforce HTTPS connections.
* Ensure that OS Login is configured with Two-Factor Authentication (2FA) for production VM instances.

For compute engine there is 2 type of that we need. Which is for the web and application servers.

For the Virtual Machine Image we will using custom machine images. The custom machine image will be created from VM.

for Desired Machine Type of **web server** we will be using **N2-standard-2** for web server which have the following specification:

* 2 vCPUs
* 8 GB of Memory
* 257 TB of HDD with local SSD option
* 10 Gbps Default Egress bandwidth

This fit perfectly for our web server configuration which have similar specification with existing physical server.

For the **Application server** we will use **n2-standard-4**. Which had the following configuration:

* 4 vCPU
* 16 GB of Memory
* 257 TB of HDD space with local SSD option
* 10 Gbps default egreess bandwidth

This server is higher spec than web server because it serve more complex processing of incoming data including data that receive from telekuers.

We might also used preemptible VM instance to provide additional compute resources.

<< Image Required>>

<<Storage Required>>

<<Network Requirement>>

<<Other Options>>

<<VM Backup>>

<<Server Migration>>

## **Google Cloud Platform Networking**

## **Google Cloud Platform Storage**

## **Google Cloud Platform Database**

## **Google Cloud Platform Analytics**

# **GCP OPERATION EXCELLENCE**

## **Increase Development and Release Velocity**

### **Release Engineering**

### **Automation**

### **Central Code Repositories**

### **Build Pipelines**

### **Testing**

### **Deployment**

## **Monitoring System Health and Business Health**

### **Logging**

### **Metrics**

### **Monitoring**

### **Dashboards**

### **Alerting**

### **Escalation path**

## **Disaster Recovery**

### **Planning**

### **Infrastructure Requirement**

### **Disaster Recovery on Google Cloud**

# **GCP Security, Privacy and Compliance**

## **Manage Risk with Controls**

### **Technical Controls**

### **Contractual Controls**

### **Third party Attestation**

### **Resources**

## **Manage Authentication and Authorization**

### **Grant appropriate Roles**

### **Service Account**

### **Organization Policy Service**

### **Cloud Asset Inventory**

### **Policy Intelligence**

## **Implement Compute Security Control**

### **Private Ip**

### **Compute Instance Usage**

### **Compute OS Images**

### **GKE and Docker**

*<< for this project we assume to use compute engine insteadof container>>*

### **Runtime Security**

### **Partner Solution for Host Protection**

## **Network Security**

### **Firewalls**

### **Network Intrusion Detection**

### **Traffic Management**

### **Network Connectivity**

### **Build In Tools**

## **Data Security Control**

**Encryption**

**Storage**

**Access Controll**

## **Database Access Control**

## **Data Security (Data Loss Prevention)**

## **Build Apps with supply Chain Security control**

### **Container Security**

#### Container Analysis

#### Binary Authorization

#### Web Security Scanner

## **Infrastructure Audit**

# **GCP Reliability**

## **Reliability Goals**

## **Observable Infrastructure and apps**

## **Design for scale and HA**

## **Build Efficient Alerts**

## **Incident Management Process**

# **GCP Performance and Cost Optimization**

## **AutoScaling and Data Processing**

### **Compute enginer Autoscaling**

### **Google Kubernetes Engine Autoscaling**

### **Serverless Autoscaling**

### **Data Processing**

## **GPU and TPU for increase performance**

### **GPU (Graphics Processing unit)**

*<< This project has no GPU requirement for Marketview Application>>*

### **TPU (Tensor Processing Unit)**

*<< This Project has no TPU requirement for Marketview Application>>*

## **Application Tuning**

### **Instrumentation**

### **Debugging**

### **Profiling**

## **Analyze Cost and Optimization**

### **Sustained use discount**

### **Commited use discount**

### **Preemptible VM**

# Index 1. Project TimeLine

Index 2. List of VM

# Library

<https://cloud.google.com/architecture/framework>

<https://www.terraform.io/docs/enterprise/before-installing/reference-architecture/gcp.html>