Project Name

AGISIT 2021-20212

Authors

Team 39A

Number	Name	Username	Email
87704	Samuel Vicente	https://git.rnl.tecnico.ulisboa.pt/ist187704	samuel.vicente@tecnico.ulisboa.pt
86392	Bruno Dias	https://git.rnl.tecnico.ulisboa.pt/ist186392	bruno.ramos.dias@tecnico.ulisboa.pt
92510	Lúcia Silva	https://git.rnl.tecnico.ulisboa.pt/ist192510	lucia.silva@tecnico.ulisboa.pt

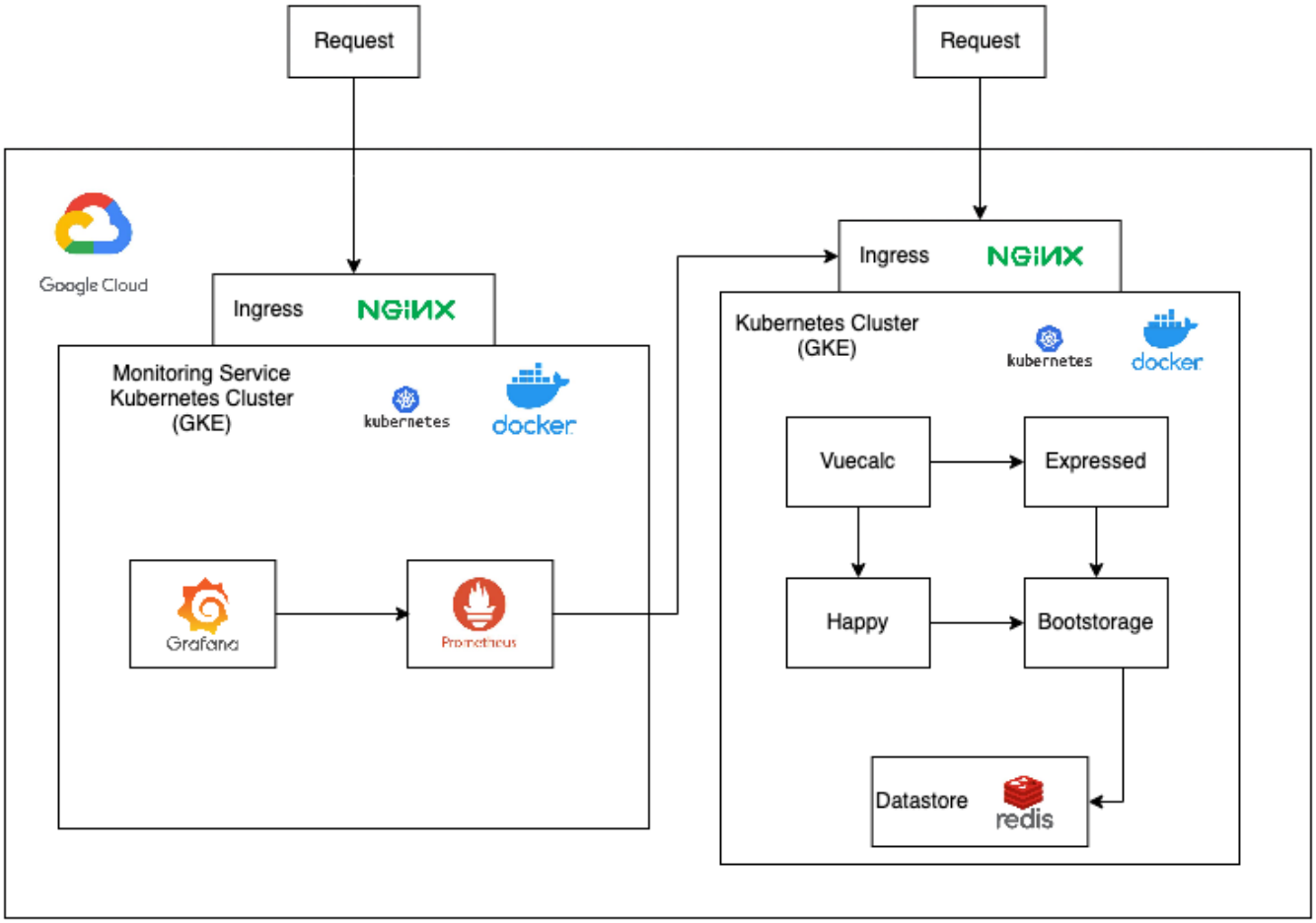
Module leaders

Member "Lúcia" was the leader for designing the Architecture of the solution and for the configuration of the "Monitoring system", with contributions of "Samuel" in the definition of the Compute nodes capabilities, and of "Bruno" in the networking and metrics details.

Member "Samuel" was the leader for the Provisioning of the Infrastructure, helped by "Lúcia" on the networking details and by "Bruno" on the scalability and high availability strategies.

Member "Bruno" was the leader for the deployment of the Applications in the Nodes/Containers helped by "Samuel" on the Load Balancing strategy, and by "Lúcia" on the orchestration configuration.

Getting Started



Kubernetes cluster

We will use the microservices based application Kubernetes Starterkit, which has the following microservices:

- Expressed and Happy: Basic calculator functions as APIs;
- BootStorage: Service that stores and retrieves data provided to it via the former two;
- VueCalc: Serves as the front end application for the service;

We will deploy this application on Google Cloud Platform using the Google Kubernetes Engine API with Terraform to create and manage the Kubernetes cluster.

Each Kubernetes Pod will only have one container running a specific microservice.

Pods:

- 1+ Expressed;
- 1+ Happy;
- 1+ BootStorage;
- 1+ VueCalc;
- 1 Redis DataStore;

Every microservice has the capability of scaling horizontally, meaning we can create new pods on demand if the services load so demands, only the Redis Pod will not scale.

We will use the Kubernetes load balancer services and Ingress as API gateway to the frontend in order to route the client requests to the correct VueCalc Pod.

Monitoring Service

We plan on deploying the Monitoring Service on a separate Kubernetes cluster using the Google Kubernetes Engine API with Terraform to create and manage the Kubernetes cluster.

The cluster will have 1 master and 1 node running the Prometheus Pod and Grafana Pod; Ingress will be used to forward the client requests to Grafana. There is no need for scaling as only a small team should have access to the Monitoring Service.

Versioning

We use RNL Git for versioning.