This project has done a short part of requirements but below will be schema and information about the possibility to build the project related to all requirements.

What was done:
1. Modify the app from the main repository to satisfy requirements (it was modified for correspondence with Python3.8), push modified app on my repository on GitHub https://github.com/Andr1500/notejam_python3.8.git.

2. Create another project on the GCP platform (name - “notejam”), initialize this project in the Cloud Shell environment. Clone the repo from GitHub to the Cloud Shell.

3. Create Dockerfile, build Docker image, and push the image to GCP Container Registry.

4. Create terraform script(main.tf), init terraform and run this script. As the output of the work Terraform script in output is the link to the built app.

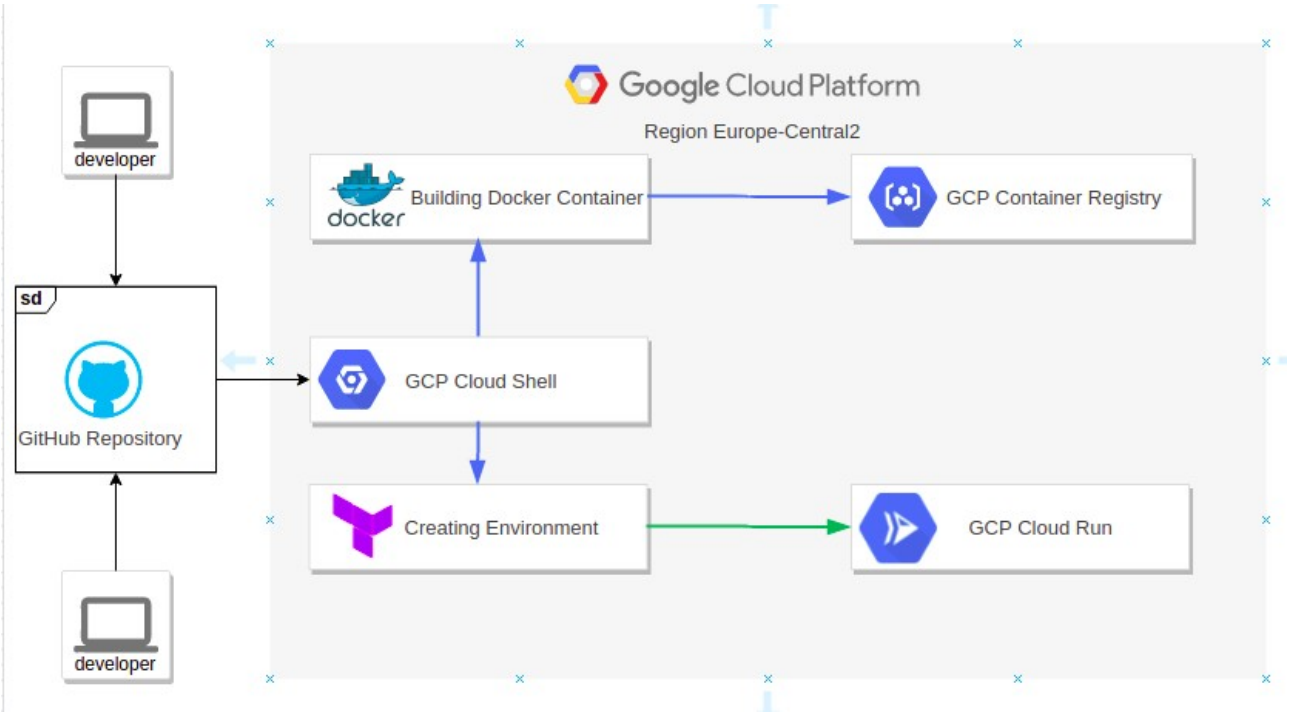

Figure 1: Present infrastructure

Posibility of improwing this app for satisfy requirements:

1. “The Application must serve variable amount of traffic. Most users are active during business hours. During big events and conferences the traffic could be 4 times more than typical.”

This application based on GCP Cloud Run. Cloud Run is regional, scalable tool → https://cloud.google.com/run/docs/about-instance-autoscaling

2. “The Customer takes guarantee to preserve your notes up to 3 years and recover it if needed”

All notes and user’s data will be store in GCP Cloud SQL service. This data will be backuping by Cloud SQL Backups service and will be archiving by GCP Cloud Storage service.

3. “The Customer ensures continuity in service in case of datacenter failures.”

Cloud Run is regional service and it will be work if one of the datacenters will be down. HTTP(S) Load Balanser is global service and will work in case if datacenter will be down. Cloud SQL service is zonal service, for better availability it can be stored in 2 availability zones: primary zone and secondary zone. GCP Cloud SQL Backups should be multi-regional. GCP Cloud Sorage should be multi-regional.

4. “The Service must be capable of being migrated to any regions supported by the cloud provider in case of emergency.”

This schema is capable to migration to any region in case of emergency.

5. “The Customer is planning to have more than 100 developers to work in this project who want to roll out multiple deployments a day without interruption / downtime.”

If we have different branches for each developer, we can separate deployments in Jenkins for each developer, Docker images can be separated by tags. Terraform file “main.tf” also can be in the repository of the developer. If developer needs to delete built envitorment(in case if the developer don’t need ii any more), the developer can push blank “main.tf” file, it will erase the environment.

It relates to the process of construction of Jenkins job:
When developer makes any changes in his repo, GitHub hook triggered, the code will be pushed to Jenkins job and the job will be started. Structure of Jenkins job:
a) start with Unit test, if test failed, the job will stop and developer will receive the information about fail.
b) If Unit test will success, Docker container will be build and will be pushed to Google Container Registry.
c) When the Docker container will be pushed, next necessary environment will be created with Terraform file in the Jenkins job.

6. “The Customer wants to provision separated environments to support their development process for development, testing, production in the near future”

It is possible to separate environments with creation different projects in GCP .

7. “The Customer wants to see relevant metrics and logs from the infrastructure for quality assurance and security purposes.”

GCP Cloud Monitoring can be in use for monitoring GCP services: Cloud Run, Cloud Container Registry, Cloud SQL, load Balanser, SQL Backups, Cloud Storage. Cloud Monitoring can be enough for observability metrics and logs. Visualization for the customer can be done with Grafana tool which can be integrated with Cloud Monitoring.

This project has a concept that the developers and client have minimum access to the GCP environment.

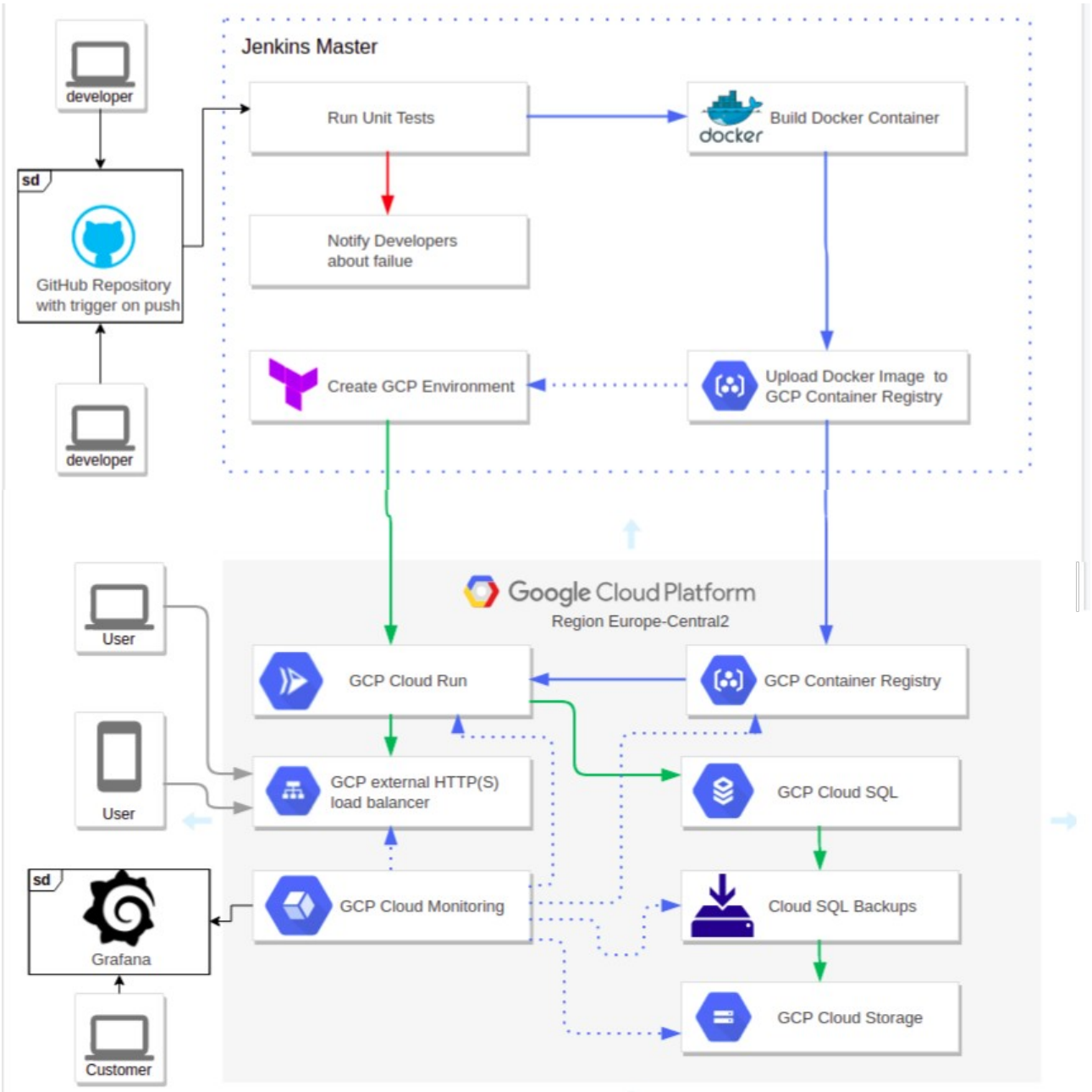

Figure 2: Full infrastructure