Decentralized Virtual CDN With opportunistic offloading

The introduction of GDPR imposes a significant responsibility to system's architecture, sysadmins and programmers to provide new tools to support DPO's requirements to meet all legal aspects. Currently, we are facing a data sovereignty battle where regulators are demanding privacy first solutions that are not aligned with big tech players.
This lab assignment is an effort to bring solutions towards this goal. Specially, this assignment will have the following goals:

Use Google Cloud and to provide a CDN service, in the likes of cloudflare and akamai.
You can use manage and unmanaged services. **Keep in mind,** you will be evaluated in several criteria.
You will have to deploy servers to meet client demand, with clients generating requests from across the globe. In order to reduce latency you will have to deploy servers nearer the users; keep in mind that servers closer to the user might be costlier.
Using client's devices is "free" must it has several constraints. You can only use 100MB of space in the client devices for caching. You will have to deploy a reverse proxy with the client solution.
Example: a client downloading a video can share it among other clients that are near; for emulating this, clients can only communicate with other clients using the private network in the cloud.

Groups will also be evaluated on the performance of their solutions. All groups will be **competing between themselves**.
The grade will be evaluated in the following aspects:

C1. Performance – latency and throughput in the response to users.
2.5pts base + 2.5pts ranking= **5pts**

C2. Cost – cost to run the solution
4pts base +4pts ranking= **8pts**

C3. Architecture Design – Overall architecture solution quality
2.5pts base + 2.5pts ranking= **5pts**

C4 Predictive Model – Overall accuracy of the model
1pts base +1pts ranking= **2pts**

The base grade is 10pts if you deliver all the components, in a working fashion.
The rest of the points will be attributed based on the ranking per criteria.
The raking will follow a Quartile distribution, Q1 - 100%, Q2 - 50%, Q3 - 25%, Q4 - 0%.

Example:
Group K has the 10 points baseline, and score C1: Q4, C2: Q3, C3: Q2, C4:Q1 will have the following grade:

10pts + 0 + C2(25%*4) + C3(50%*2.5)+(100%*1)=13.25

C1. Performance
Given that we will not be using certificates, static Ips will be used instead.

For bootstrapping, you will have to create a small application server that will acts as a load balancer:

GET static_ip /cdn/lb/ip

Returns a Json with the IP of the CDN server that the client should use

C2. Cost

You will have to provide the full analytical cost of your solution, including, VMs, containers, storage used, ip traffic, etc. Must be included in the final report (Max 2 pags).

C3. Architecture

In the final report, you will have to provide a diagram and full explanation of the rational. Optimization should be fully detailed. The elasticity strategy must be explained in the fullest detail. (max 5 pages)

C4. Construct a ML model to perform workload prediction.

This will be used in C3. (max 2 pages)

Tools

You can use the following blocks:

- NGINX
- Memcache
- MariaDB, Cassandra, Postegresql, mongodb, redis, voltdb, ceph
- Filesystems: Ext4, ZFS, Btrfs
- All managed services from GCP
- Python, Java, Rust
- You can use C in Ningx
- You cannot use: PHP, Node.JS or any other form of javascript. This is not negotiable.
- In doubt, ask me first.

For client devices, assume a raspberry level device, so a micro-vm will perfectly emulate it (1 vcpu, 1GB).

You must provide an admin client to allow the insertion and deletion of assets. This will trigger a dissemination across all active servers in the infrastructure. You must use a self-signed certificate to support https.

We will use the standard REST/HTTP interfaces.

You must provide the client so you can leverage offloading opportunities.