Running moodle.org site on Kubernetes

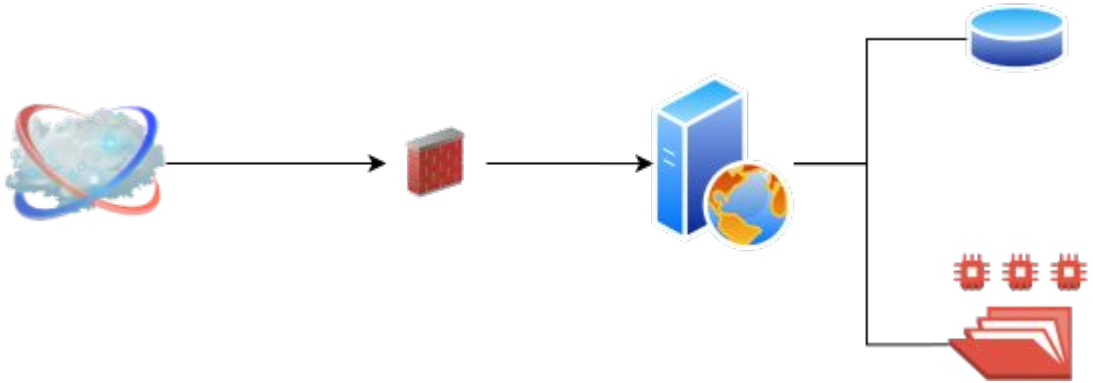
Eduard Cercós

20 04 2021


moodle
Empowering educators to improve our world

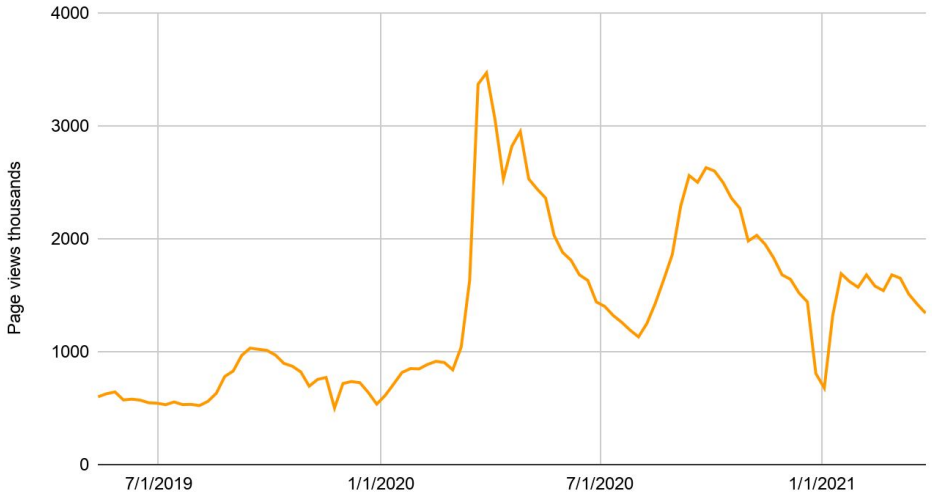
Where we came from

- A single server
- Low traffic → affordable
- Split services to 2 layers: Frontend + backend
- Added load balancing in preparation for scalability

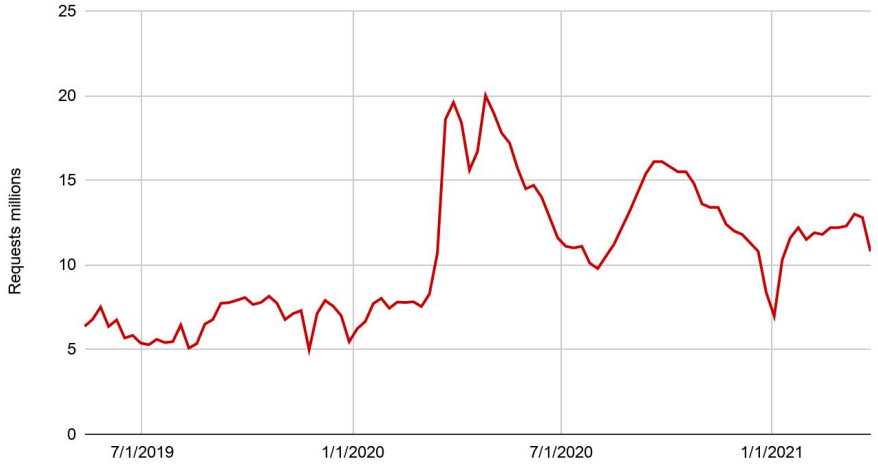


What is this!

Page views thousands



Requests millions

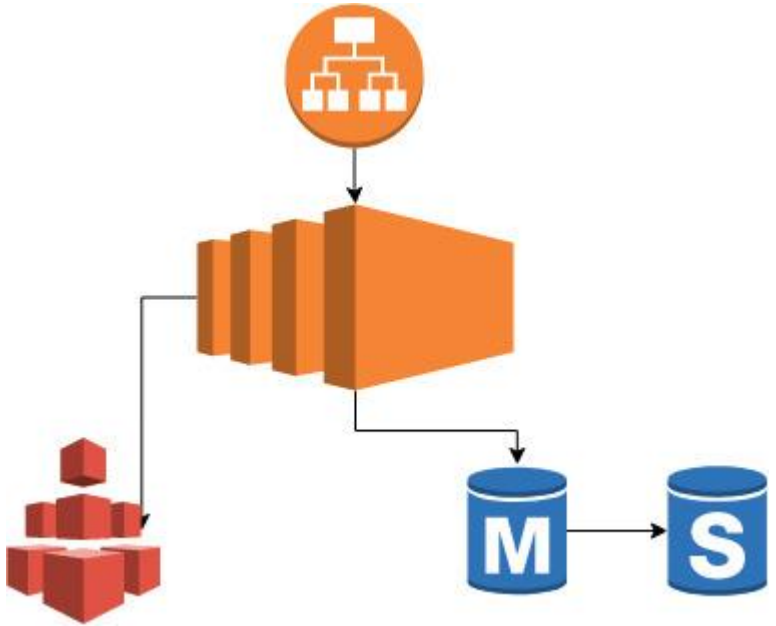


We were prepared

- Kubernetes cluster for internal and external sites
  - HA master
  - several identical nodes
  - easy to scale (IaC)
  - HA services

A kubernetes cluster

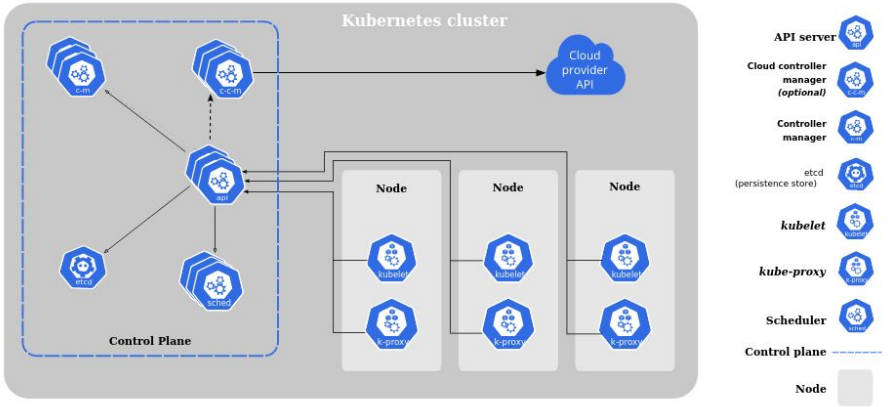
- In AWS, using 3 AZ
- Multiple Nodes per Zone, scalable
- External services like
  - Load Balancers
  - RDS
- EBS + EFS



Do you know it?

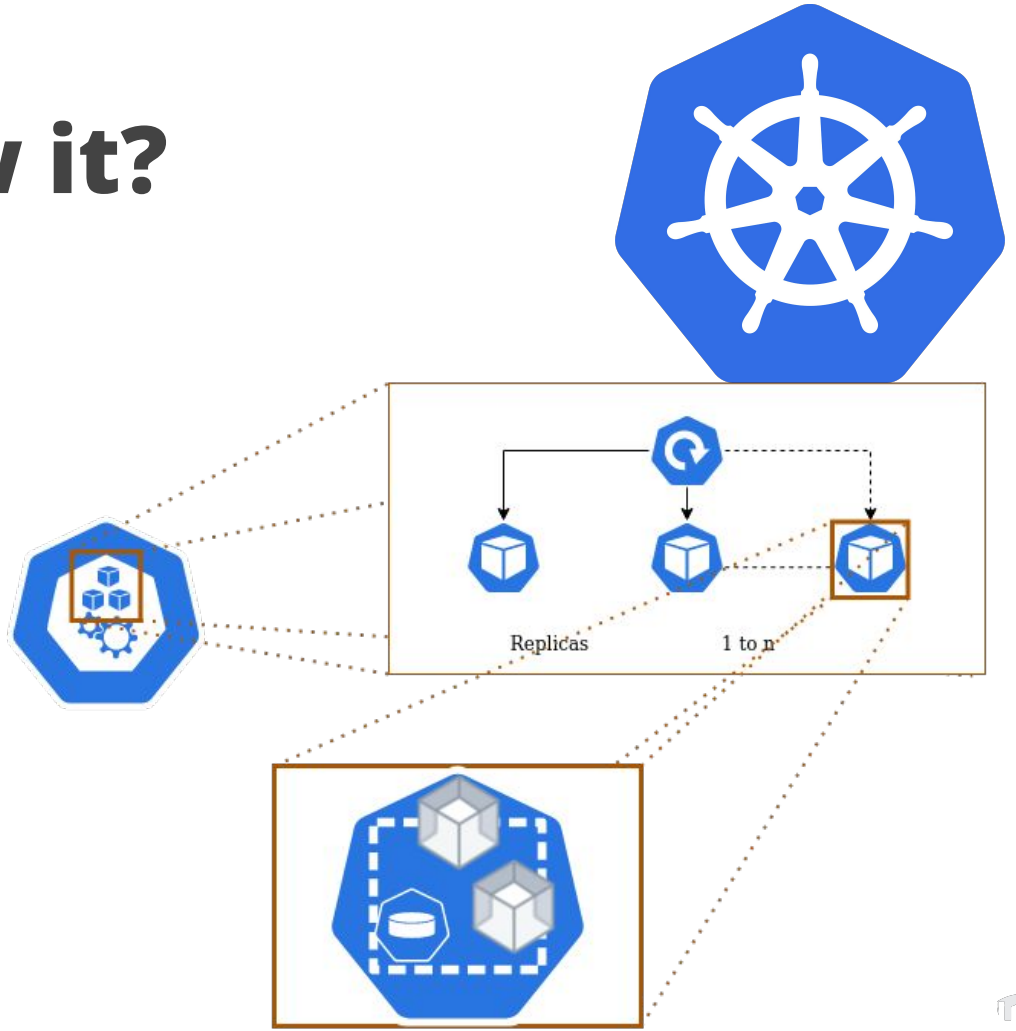
- **Kubernetes orchestration**

  Pods, services, deployments, daemonsets, ingresses (https://kubernetes.io/docs)



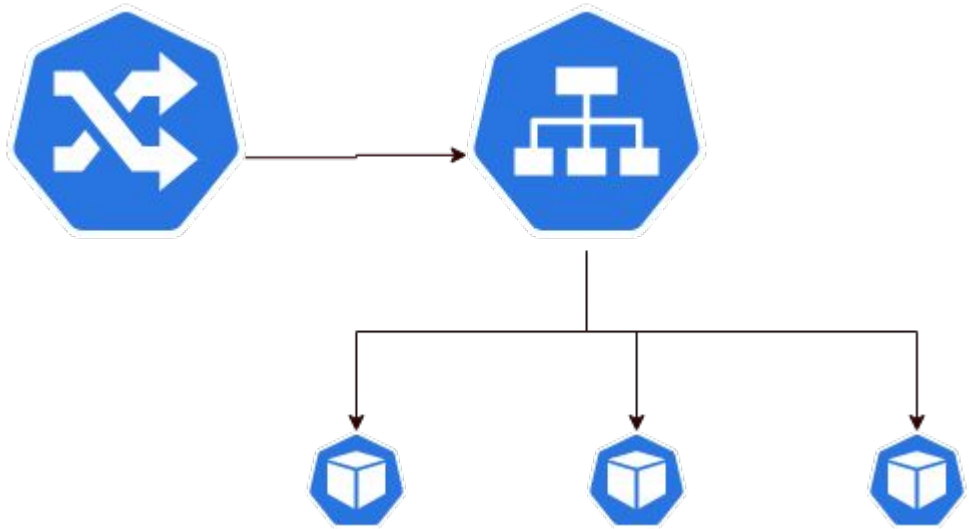
Do you know it?

- **Workloads:**
  - Deployment, Replicaset, StatefulSet, DaemonSet, Job and CronJob
    - Pod lifecycle, PV, Containers
    - Probes
    - Resource control
- Pods are ephemeral!!

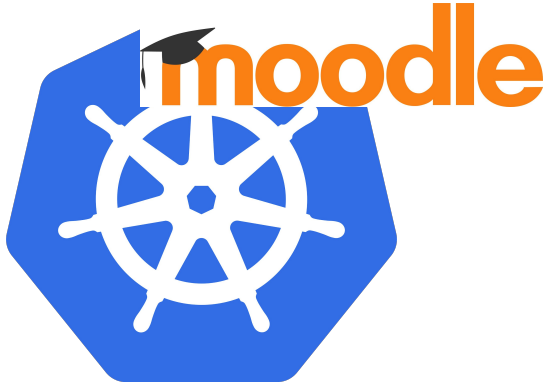


A bit more

- **Services and networking**
  - SVC entry point for pods (using networking abstraction)
  - Easy plugable to load balancers or Ingress objects
  - To intercommunicate within the cluster



Into the wild



- **Design services:**
  - Nginx with php-fpm (clients, cron)
  - MUC service: Redis
  - Session service: Redis
  - Database (external)
  - Preserve data
- **Previous experience**
  - learn.moodle.org
    - MOOC courses, ~4000 participants
    - spaced in time (low traffic)

Solution I

- **Deployments**
  - Web app (nginx + PHP)
  - Cron

```
  "kind": "Deployment",
   "metadata": {
      "name": "moodle-org"
   },
   "spec": {
      "replicas": 3,
      "selector": {
         "matchLabels": {
            "app": "moodle-org",
...
```

```
"affinity": {
   "nodeAffinity": {
      ...
                  "key": "failure-domain.beta.kubernetes.io/zone"
      ...
   "podAntiAffinity": {
      ...
                  "topologyKey": "kubernetes.io/hostname"
```

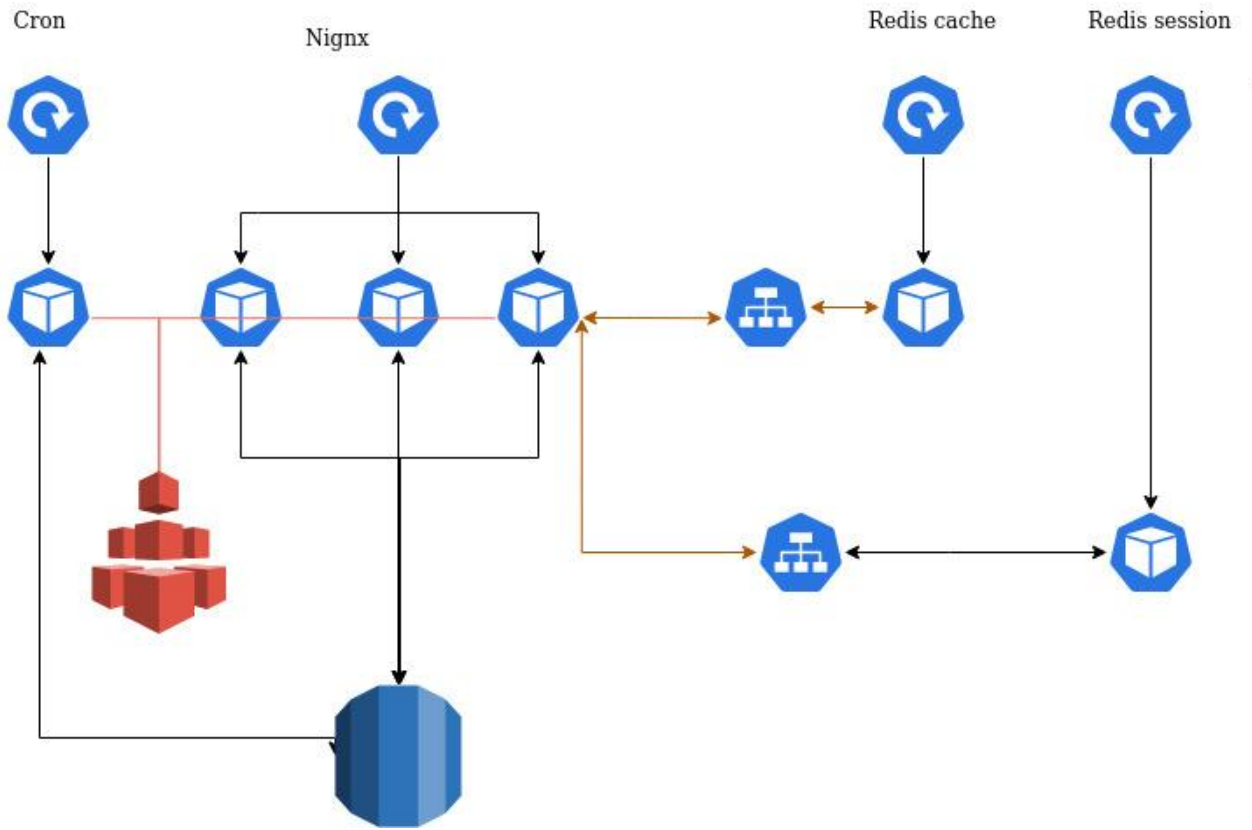
Solution II

- **Deployments:**
  - **Limitation to same AZ (to reduce Interzone Bandwidth)**
  - Redis cache
  - Redis session

Use them wisely! →

```
"containers": [
....
                "resources": {
                    "limits": {
                        "cpu": 2 , “memory”: “4GiB”
                    },
                    "requests": {
                        "cpu": "500m", “memory”: “1GiB”
                    }
                },
```

So far



Solution III

- Services
  - http
  - cache
  - session
  - No cron service!
- Ingress: moodle.org → http

```
"kind": "Service",
 "spec": {
  "ports": [
   {
    "targetPort": 80,
    "protocol": "TCP",
    "port": 80,
    "name": "http"
   }
  ],
  "selector": {
   "app": "moodle-org"
  }
```

```
kind: Ingress
...
  name: moodle-org
  namespace: default
spec:
  rules:
  - host: moodle.org
    http:
      paths:
      - backend:
          serviceName: moodle-org
          servicePort: 80
        path: /
```

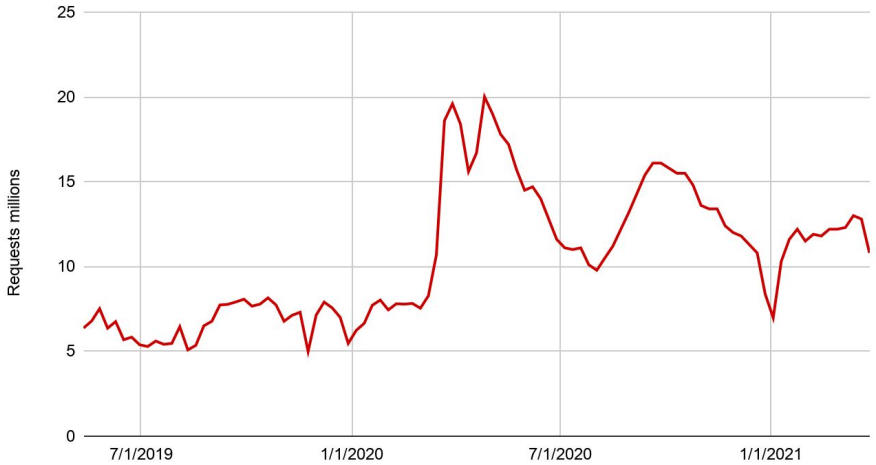
Solution IV (wip)

- Self healing per container
  - livenessProbe (tested & used in other sites))
  - readinessProbe (wip)
  - startupProbe (wip)

```
...
livenessProbe:
  failureThreshold: 3
  httpGet:
    path: /
    port: 9821
    scheme: HTTP
  initialDelaySeconds: 30
  periodSeconds: 20
  successThreshold: 1
  timeoutSeconds: 5
...
```

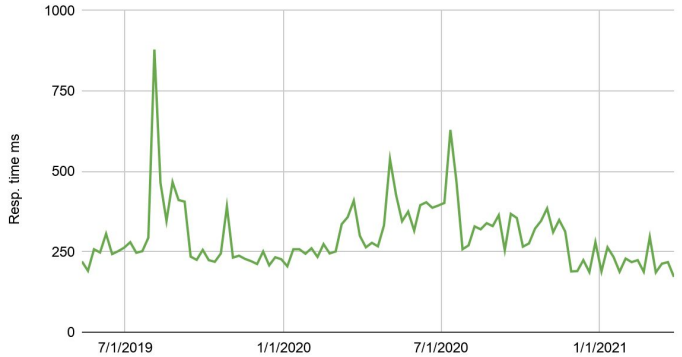
It works!

Requests millions

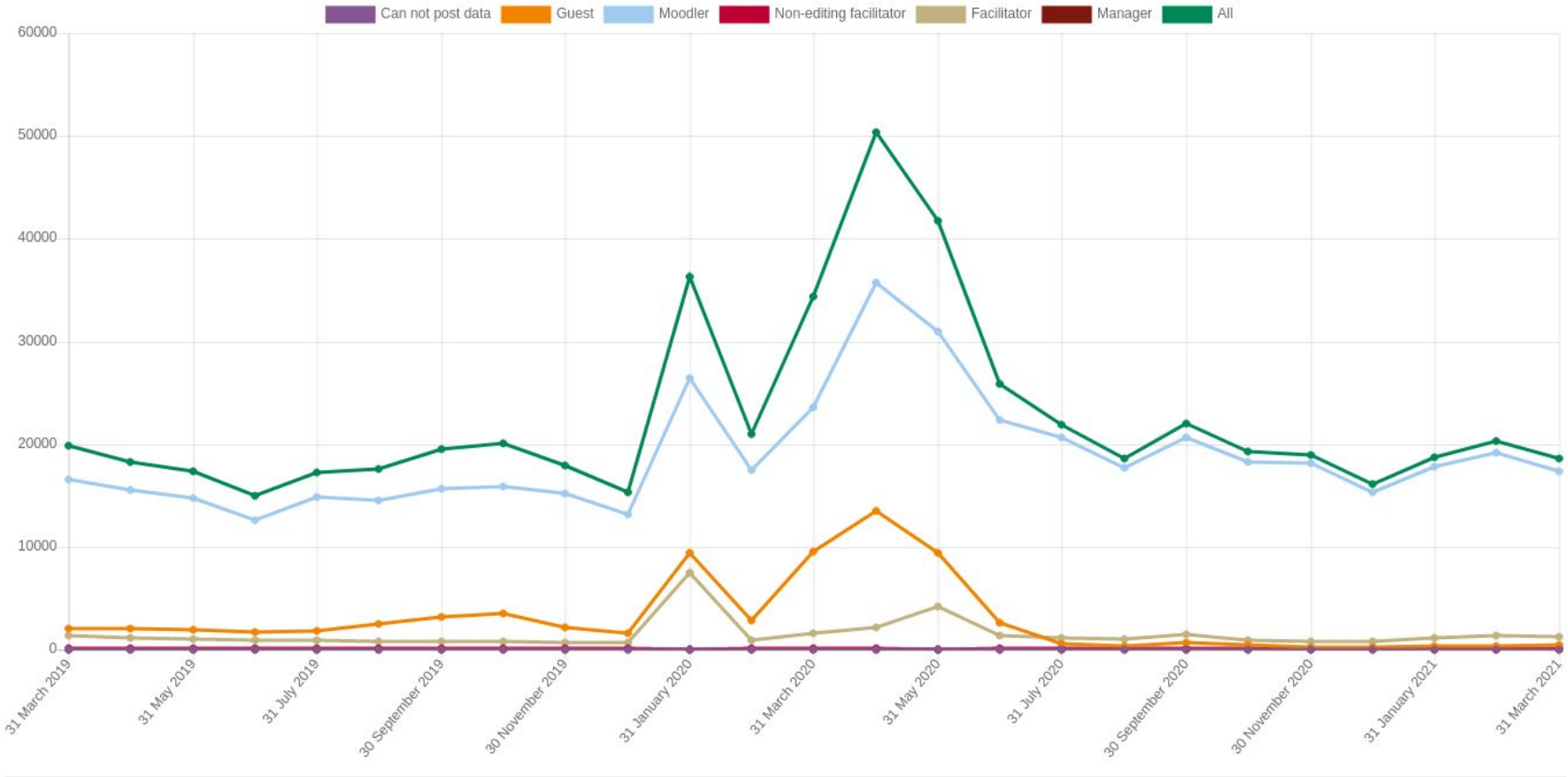


Load time sec

Resp. time ms

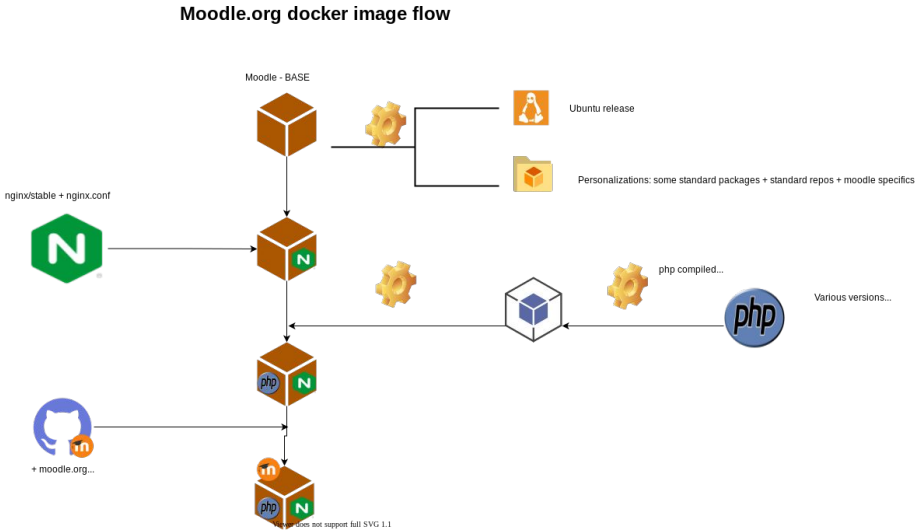


Moodle.org - Posts (all roles)



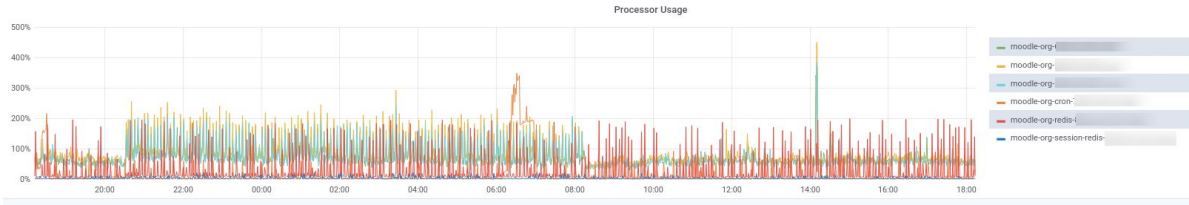
Strengths

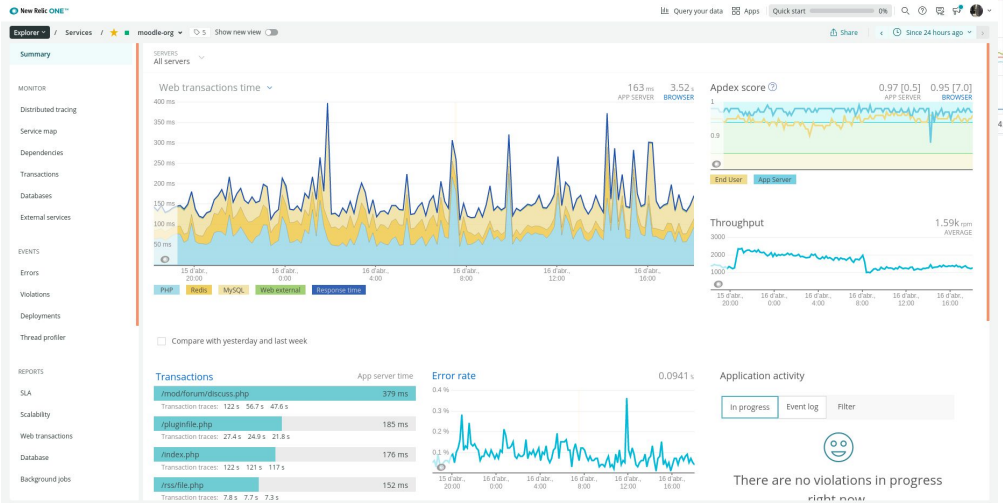
- Easily horizontal scaling (HPA in progress):
  kubectl scale --replicas=4
- Resource control (redis cleanup)
- CI/CD (almost) totally integrated with kubernetes
- Short/no downtime deployments
- But still: Moodle upgrades downtime :(



Monitoring

- Grafana + prometheus
- Loki + Graylog
- New Relic APM





Thank you!!

Eduard Cercós
DevOps Engineer
eduard@moodle.com
moodle.org Community Forums