SCALABLE SYTEM

CURRENT ARCHITECTURE

VPS

Application 1

single process

| FE+BO | — | BE |

Application 2

single process

| FE+BO | — | BE |

Redis caching

MongoDB single node
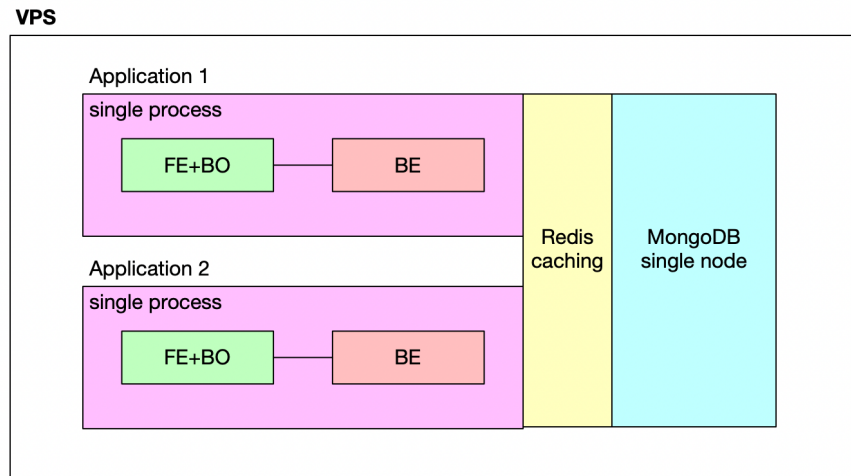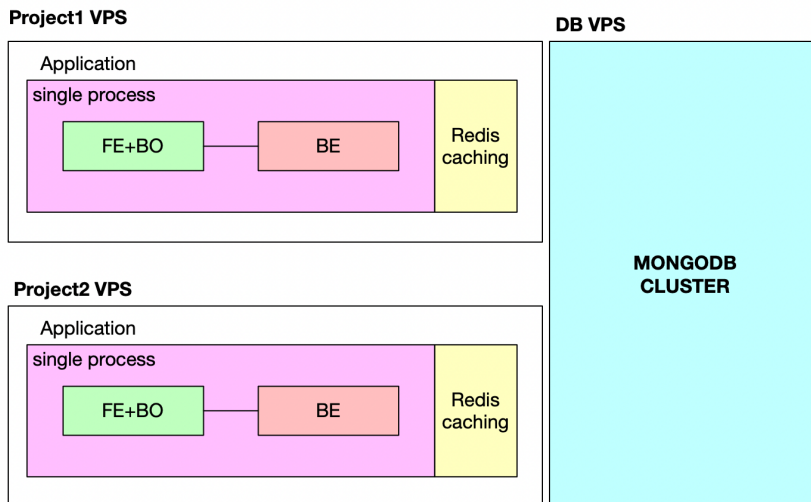
SCALABLE ARCHITECTURE

Stateless Database service => Use only one DATABASE in separated VPS and be clustered

Project1 VPS

Application

single process

| FE+BO | — | BE |

Redis caching

DB VPS

MONGODB CLUSTER

Project2 VPS

Application

single process

| FE+BO | — | BE |

Redis caching

Stateless Application
- FE+BO are using ReactJS mean static file hosted => does not make heavy load this stage
- BE has to handle logic, access database => take some times maybe problem at this stage
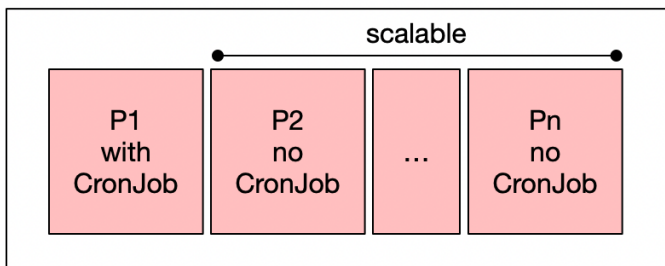 => need scale here

Application caching
Using Redis as caching for application, it shares resource across process
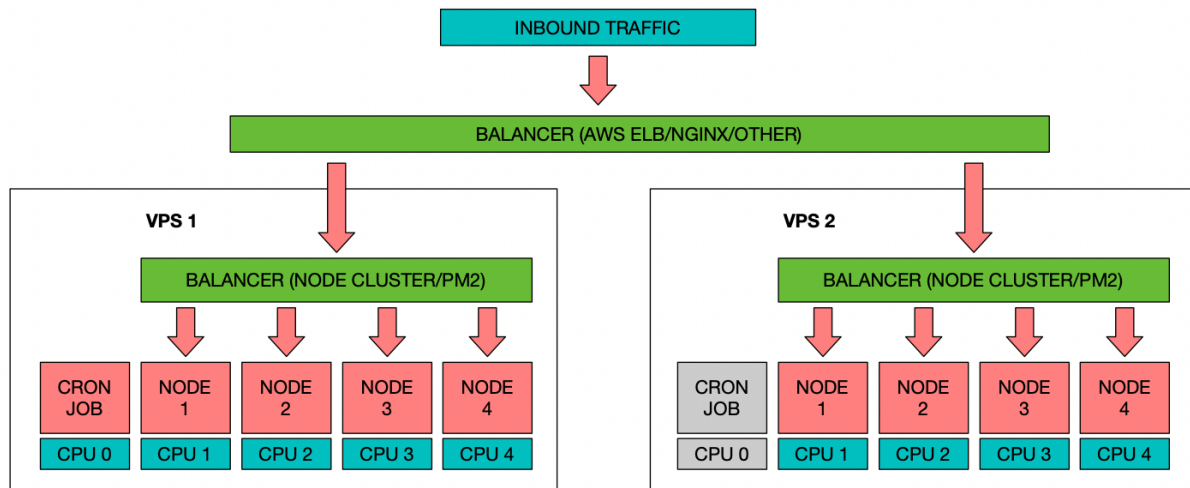
Scaling application process
Use PM2 cluster to scale application. But need separate cronjob to make stateless in process

BE application

scalable			
P1 with CronJob	P2 no CronJob	…	Pn no CronJob

Traffic balancing
Use Nginx with upstream to share traffic to another VPS

INBOUND TRAFFIC

BALANCER (AWS ELB/NGINX/OTHER)

VPS 1

BALANCER (NODE CLUSTER/PM2)

CRON JOB	NODE 1	NODE 2	NODE 3	NODE 4
CPU 0	CPU 1	CPU 2	CPU 3	CPU 4

VPS 2

BALANCER (NODE CLUSTER/PM2)

CRON JOB	NODE 1	NODE 2	NODE 3	NODE 4
CPU 0	CPU 1	CPU 2	CPU 3	CPU 4

Full system

INBOUND TRAFFIC

VPS: DNS

BALANCER (AWS ELB/NGINX/OTHER)

VPS: Project

BALANCER (NODE CLUSTER/PM2)

CRON JOB	NODE 1	NODE 2	NODE 3	NODE 4
CPU 0	CPU 1	CPU 2	CPU 3	CPU 4

VPS: Backup

CRON JOB	NODE 1
CPU 0	CPU 1

MONGODB REPLICATE SET CLUSTER

Primary Node VPS1	Secondary Node VPS2	Secondary Node VPS3