Phase 1

We have calculated the cost using the AWS calculator and created the Phase 1 diagram. Also we have set up an initial vision of the final infrastructure during this phase.

Phase 2

This phase was more technical and many foundational blocks were created including but not limited to:

- Our project VPC where all our infrastructure will be hosted
- Internet gateway to enable our EC2 instances to be reachable
- Route table in order to pass traffic from our EC2 to internet and vice versa
- A public subnet to hold our EC2
- An EC2 instance created using an Ubuntu AMI and provided user data

At this stage our web server and database were hosted on the same instance meaning that it does not align with the Amazon Best Practices Framework. Hence this requires changes which will be done in the further stages.

Phase 3

Calculated a new estimate and created a new diagram to reflect the infrastructure evolution.

This phase proved to be the most challenging one due to the high number of moving parts. However our work can be outlined in the following fashion:

Step 1.

Created an micro EC2 instance which will host C9 Development Environment

Step 2.

Instantiated a RDS service taking into account the potential load we may receive. Hosting the database in a multi AZ option in order to fulfill the high reliability requirements.

Step 3.

Connected to our old database in order to dump its data.

Logged in our new RDS instance in order to migrate our data.

Step 4.

Launched a new EC2 instance which only hosted the web server and configured it to consume the secret manager in order to get the database credentials.

Step 5.

Terminated our old EC2 instance, the careful consideration of steps meant that the potential users would not experience down time.

Phase 4

Calculated a new estimate and created a new diagram to reflect the infrastructure evolution.

Step 1.

Created a launch template to be used by the auto scaling group

Step 2.

Created an auto scaling group and configured it to be across two availability zones and to use the previous made launch template.

The settings for the auto scaling number of instances are:

Minimum : 1
Desired: 2
Maximum: 4

Step 3.

Created an Elastic Load Balancer and assigned our auto scaling group to it.

Performed a sanity check by connecting through the EC2 public IP addresses and the ELB DNS address.

Step 4.

Performed a load balancing test with 1000 requests per second. Our infrastructure handles this spike in load effortlessly.

We increased the load to 2000 and it handled it perfectly.

However when we increased the load to 10000 requests per seconds the infrastructure started to hiccup. We suspect this is due to the long period of health status check (300 seconds) However this was purely experimental since the task expects 1000 requests per seconds to be handled easily.