Using tools like terraform, which is a tool for writing architecture as code, we designed a full applications infrastructure.

What did we delivered : A terraform script capable of lauchning the complete Sparta internal tools architecture on AWS automatically.

Using Amazon machine images of past instances,

Terraform lets you design whole amazon cloud architecture using script. It can be very useful as it automates the whole process of launching different resources of an infrastructure making it very easy to launch multiple similar resource. The script is highly configurable and makes it simple to re-use parts of the script to design new architecture.

All that is needed apart the script is to write a few commands making the whole process of launching or destroying whole infrastructure completely automated.

Terraform can be very useful devops tools as it permits flexibility of modifying the configuration of all resources, but it also makes it relatively simple to switch Cloud Service Provides, for example Microsoft Azure.

Since the whole process of launching an architecture is automated, terraform makes it very simple to re-launch complete architecture in case problem arises with the applications that are already in production.

Finally, terraform makes the process far more efficient as it saves a lot of time that is usually required to launch all isntances, databases, elb manually and making sure security group and ACL and configured accordingly. It makes it simple to migrate most resources from availability zones. Most importantly, Terraform is free to download and use.

Sofware architecures :

While writing the code is interesting in itself, we were more interested in the design of the architecture in order to maximise effectivenes and make the deployed architecture flexible, resilient and cost effective. Other factor like user experience were taken into account, as we ended placing the server in the London region in order to deliver faster service.

Most importantly we decided to step away from the monolotith and divide the application into three tiers.

The view tier which is the front end.

The logic tier which is the API

And the data tier which is the database.

The application was made secure by seraprating each tier and making sure each part can only communicate with the part it must.

The app was made flexible and cost effective by setting up the API in a autoscaling group which means the instance were automated to launch a new instance when the CPU reached a certain limit and destroy a instance when the cpu decreased. This also made the application cost effictiveas as it only span up instance when needed and saved having an extra instance running all the time.