**We would like you to design an e-commerce platform to support a service like the Amazon website where customers can buy items and sellers can maintain inventory and the platform owner can maintain a marketplace ecosystem. This platform that you will build will run at a similar scale as Amazon. Focus on the backend platform infrastructure (databases, operating systems, networking, automation, redundancy, monitoring etc) that will support the e-commerce applications, but not necessarily the front-end applications themselves. You have flexibility to choose technology building blocks that you are most familiar with, and please justify or explain why you made particular technical and architectural choices.**

**Important points to cover**:

* How will you handle the loss of a region?

To prevent risk of going out of business when we lose a region, high availability is not enough we need Disaster Recovery. In case of HA, we ensure there exists a fallback mechanism for our services. The service that runs in HA is handled by hosts running in different availability zones but in the same geographical region. This approach, however, does not guarantee that our business will be up and running in case the entire region goes down. DR takes things to a completely new level, wherein you need to be able to recover from a different region that’s separated by over 250 miles. Our DR implementation is an Active/Passive model, meaning that we always have minimum critical services running in different regions, but a major part of the infrastructure is launched and restored when required.

* How will you factor in growth and scale?

By using Kubernetes, we ensure that the system or service is designed to scale based on user demand, so if a microservice on the application is getting a lot of requests e.g. more products are being bought, then we need to scale that service by scaling out during peak times and scaling down during less busier times.

* Security, Reliability, Speed are all important factors too.
* You can design this architecture on the cloud or on-prem.

See Architecture Diagram attached on GitHub repo.

* The technology you choose will be important, so if you use a particular database make sure you can explain why it’s a good fit

Tech stack can include

* Angular and Node JS for front-end
* Java Spring Boot for Backend functionality
* RabbitMQ for Messaging between the different microservices
* Kubernetes on EKS for containerisation and orchestration
* PostgreSQL for the Databases (could choose a different one)

Spinnaker for cross cluster deployment

Kubernetes

Using the blue cluster for patching, testing, upgrades and blue-green deployemtns

PostgresQL