**Introduction**

Problem Statement: A Traditional IT infrastructure incapable to handle the predicted growth of a service provided by a medical startup company.

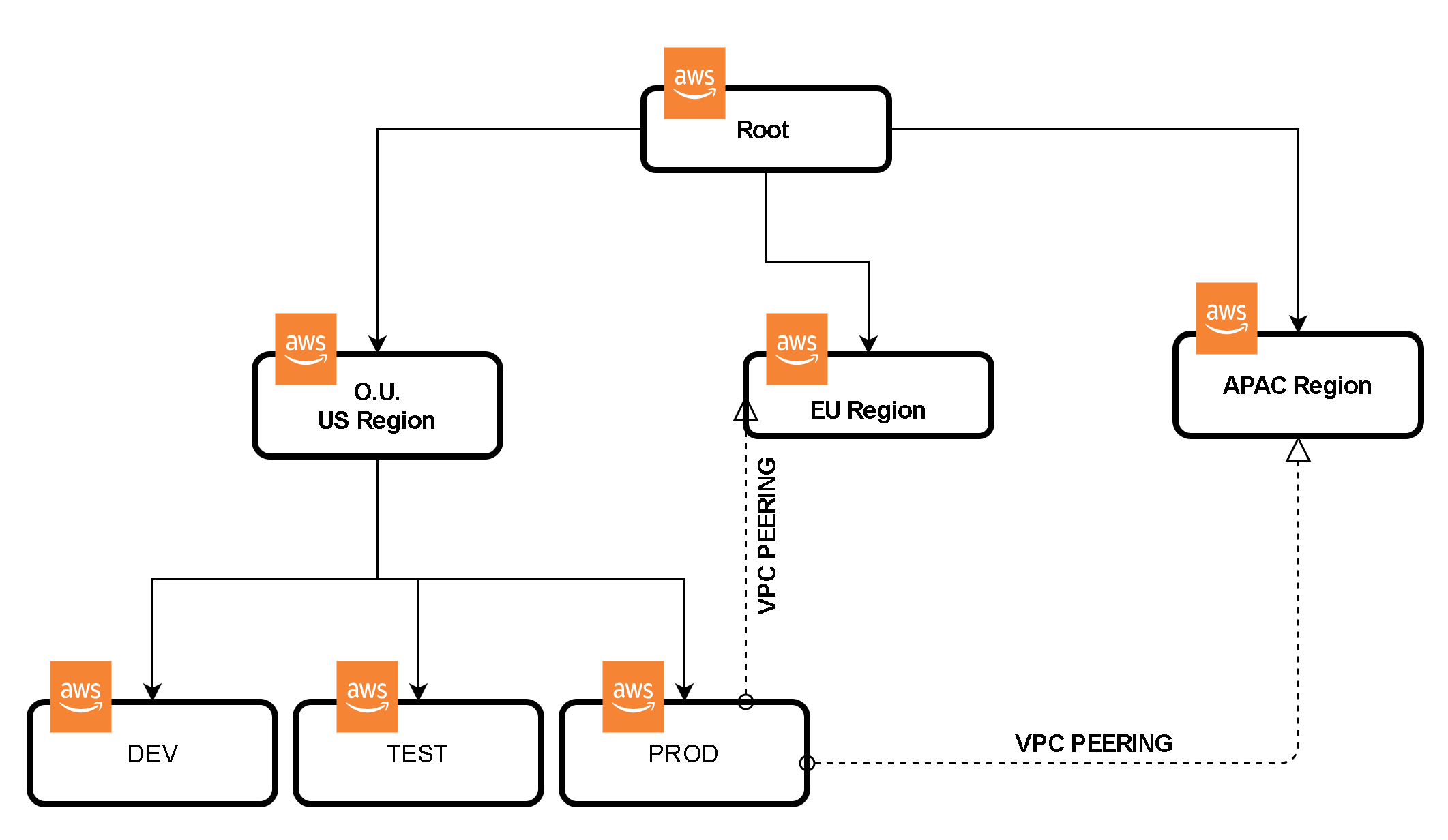
Solution: By migrating the existing IT infrastructure to a highly available, scalable and economical AWS Cloud infrastructure.

**Executive Summary**

Understanding the requirements from the current traditional architecture.

Need for a three-tier cloud architecture spread across multiple regions.

Seamless integration from Dev & Test to Production across multiple regions.



As per the requirement, we have designed the architecture to support multi-region and all development stages. For this, we have used seven AWS accounts and designed as shown above. The root account will have three sub-account which will represent the three regions. In the US region, we will have three sub-account which will have the full functioning AWS architecture for all development stages i.e. Dev, Test and Prod. A prod AWS account has fully tested Application which is ready for business and the copy of the whole system with ready to go the application will be transferred to the EU region and APAC region through VPC peering. Any modification or enhancement in an application can be done in one region and reflected in all regions.

**Overall Requirements and Assumptions**

* To make service available in United States, Europe and Asia Pacific regions.
* Ability to run Development, Testing and Production environments.
* Ability to provide a scalable software service.
* To comply with legal and data regulatory compliance w.r.t. US, EU and APAC.
* Ease of tracking the entire infrastructure’s activity.
* To secure patient related sensitive information for ex. Health records.
* To provide a secured payment gateway.

**Solution**

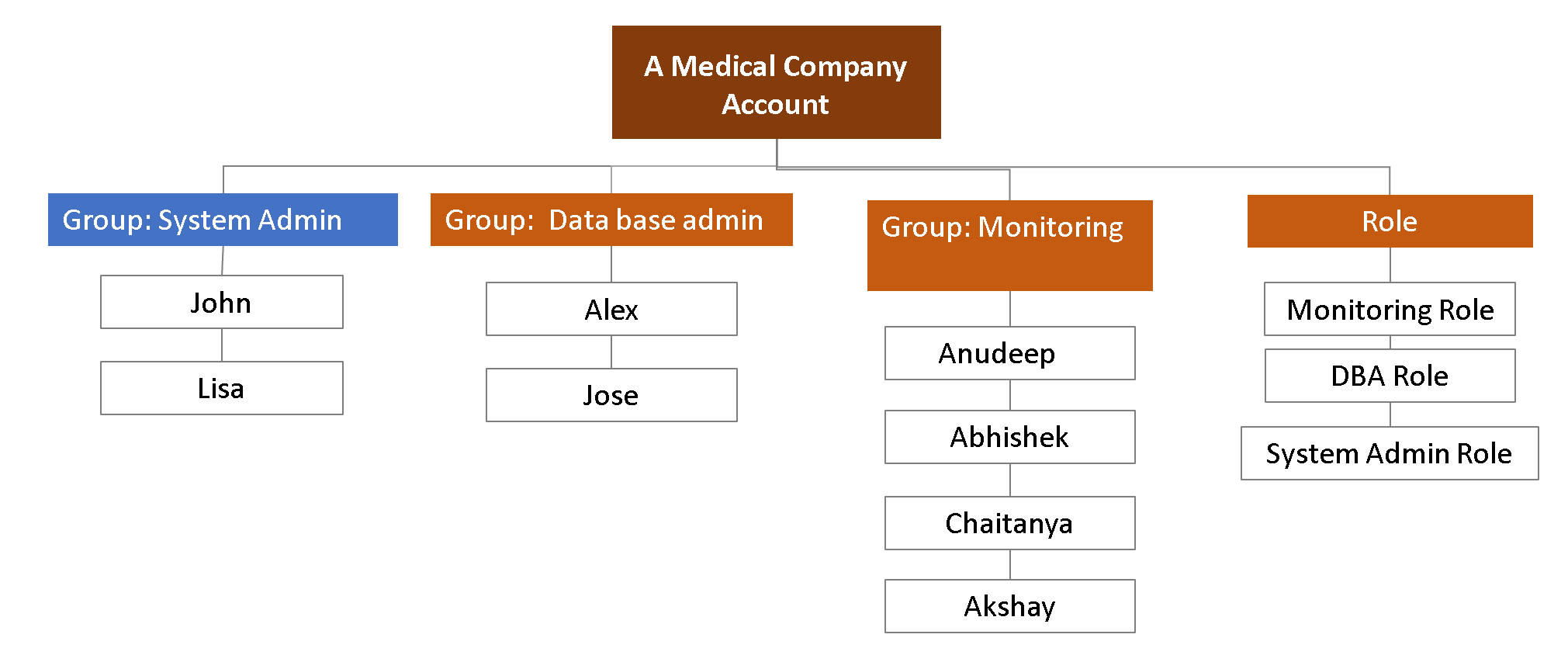
**Identify AWS Services**

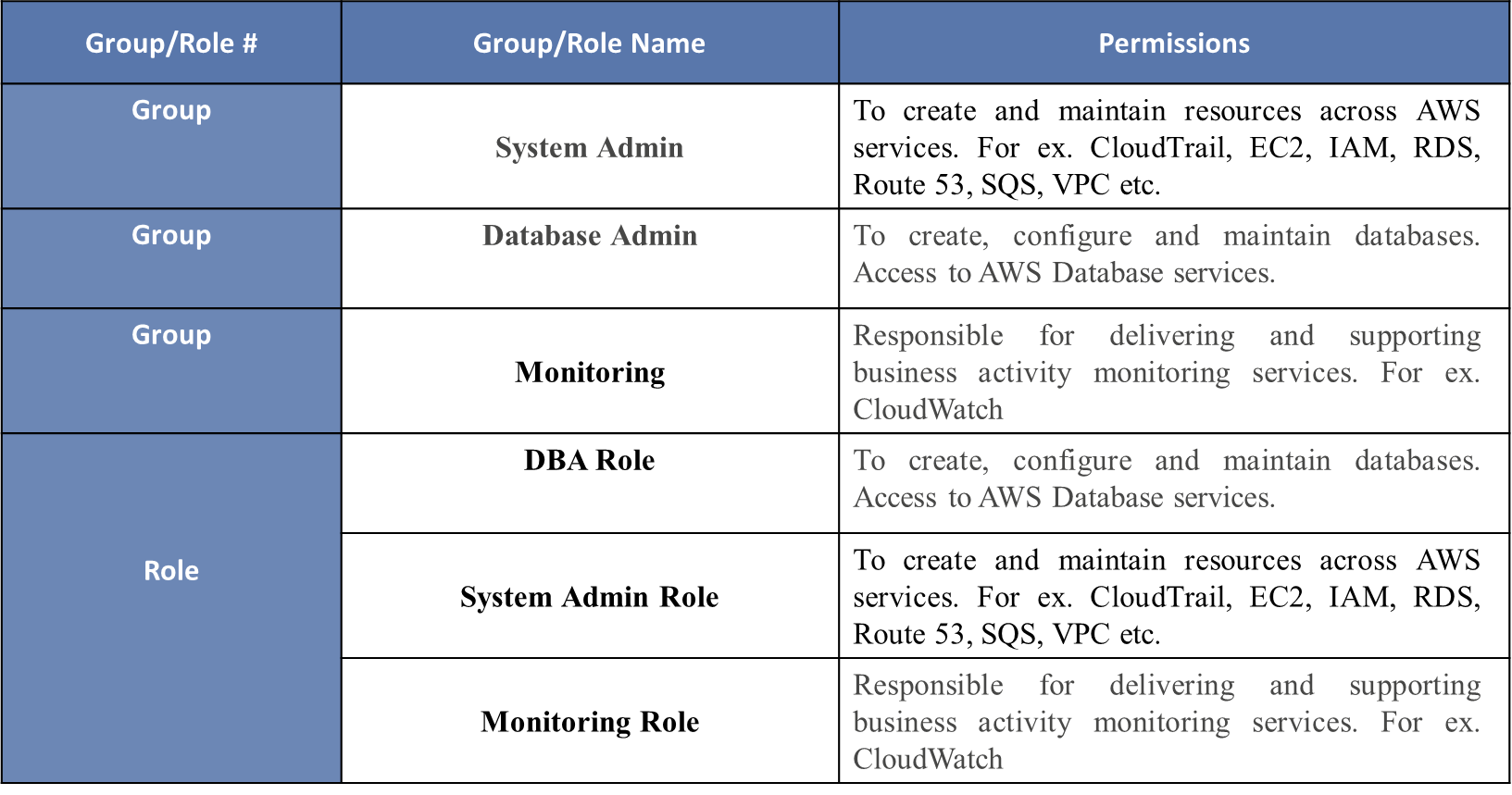
* IAM (Identity and Access Management): Manages access to AWS resources and services by creating polices, roles and groups. Will be used to create roles and permissions as per requirement.
* SNS: Helps in delivering messages or sending alerts to doctors when the customer is requesting a service.
* CloudWatch: Collects monitoring and operational data in form of logs, metrics and events to provide a unified view of AWS resources to detect anomalous behavior, set alarms, automated actions and troubleshoot. Important service to avoid downtime of the server.
* CloudTrail: Enables governance, compliance and auditing of AWS accounts. For ex. Log of each activity across AWS infrastructure to detect unusual activity and monitor AWS accounts.
* Route 53: A Cloud Domain system which routes the incoming traffic on basis of Latency, Failover, Geolocation to provide seamless customer service and routes the traffic based on the health of resources. For ex. Web servers. Helps in reachability of the company.
* Elastic Load Balancer: Distributes incoming traffic across multiple EC2 instances to make the service highly available. It will help reduce the lag for customers.
* Auto-Scaling Group: Automatically monitors and adjusts computing resources to maintain steady performance at the lowest possible cost. It will help provide bandwidth during peak hours.
* CloudFront: Helps to speed up the static and dynamic content of the application with high data transfer speed and low latency.
* Amazon Machine Image (AMI): Provides image file of instances to Auto-Scaling group for scalability.
* Relational Databases: To store the data entered by the customer in Master-Slave format with a Replica.
* S3 (Simple Storage Service): Provides scalable, high-speed web-based storage service. Helpful in storing instance snapshots and uploaded documents, images by the customer.
* Dynamo DB: To store the name of medicines extracted from the uploaded documents.
* Text-Tract: Use to scan the documents uploaded by the customer to extract text i.e. medicines.
* KMS (Key Management Service): Helps to create, store and control encryption keys to encrypt data.
* Stripe: Provides a secured payment gateway for customers to pay for the service used.

**User Authentication**

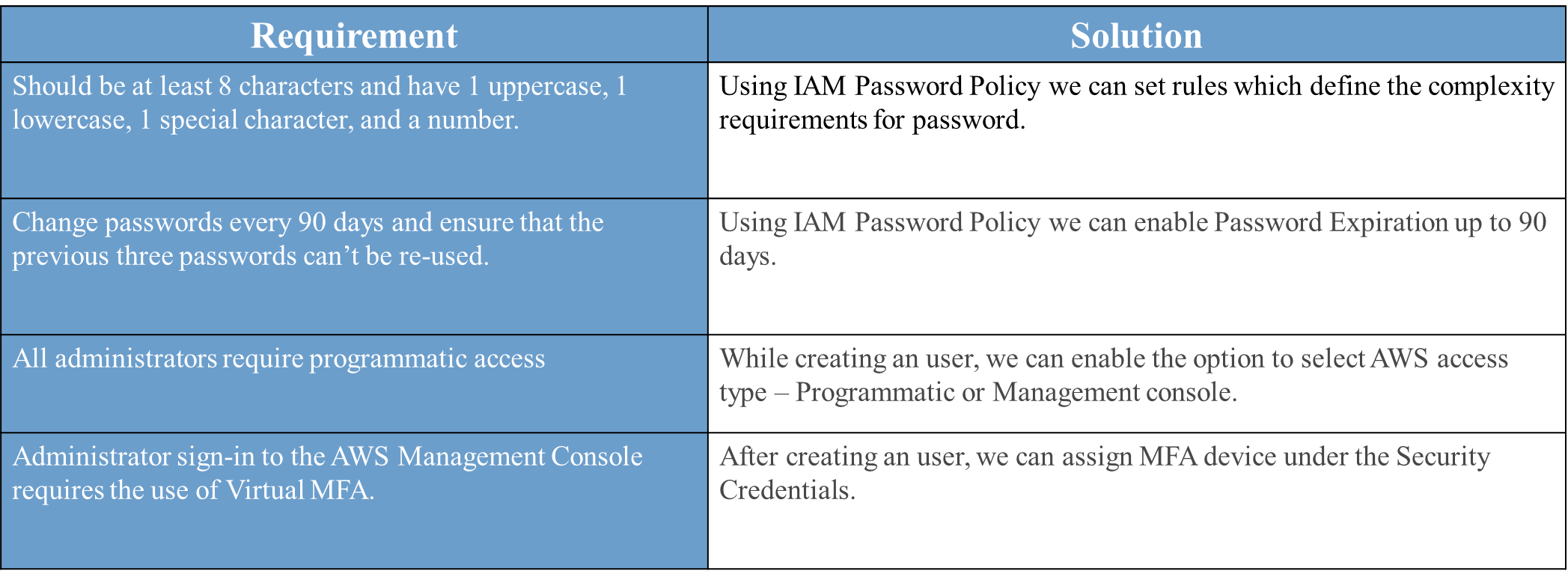
In our AWS architecture we have created System Adim and Databse admin with two person each and Monitoring group with four person as per the requirement of the company.

Along with groups, we have three specific roles created i.e. Monitoring, DBA, System Admin for cases where any one group needs a temporary access of other group.

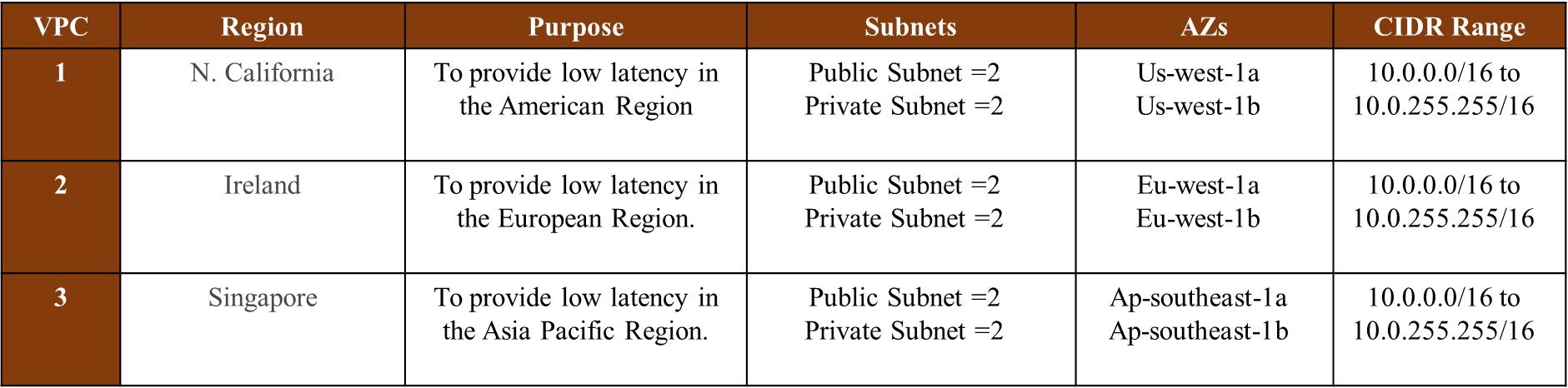




Using the IAM service, we can satisfy the criteria of the access and password protocols given below.



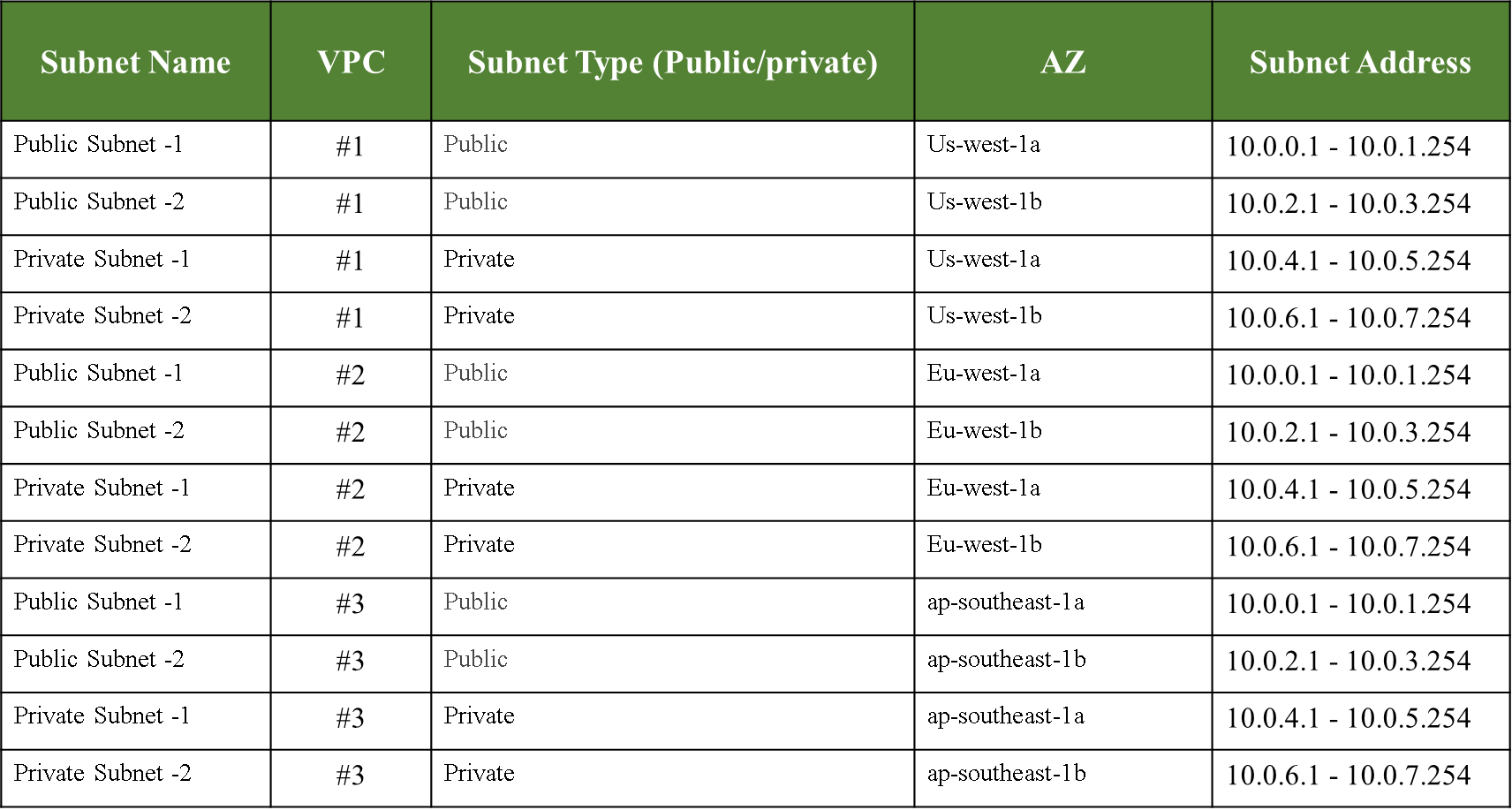
**Network and Security**

****

In the network, we have done IP address distribution as per the CIDR range. We have assigned CIDR range - 10.0.0.0/16 to each region. It will make available 65536 IP addresses for the whole network including reserved IP addresses.

Under the network, we have public and private subnets and other services which need an IP address. In each region, we have two public subnets and two private subnets. we have assigned 510 IP addresses to each subnet. So the IP Addresses will be always available to perform auto-scaling.

Following the table contains the range of available IP addresses for each subnet.

****

After assigning IP addresses to each subnet we have used leftover IP addresses for Database tier, other services and future purpose.

**Web and Application Tier**

Following the diagram shows the AWS architect solution which is a three-tier architecture i.e. Web Tier, Application Tier and Database Tier. Inside the VPC, Web Tier is kept under public subnet, App Tier is kept under private subnet and Database Tier is inside the same VPC. To make the system highly available, we have kept Web tier and App Tier in two different zones in each region.

We have used Route 53 to redirect the traffic to VPC. As soon as IG of VPC receives a The HTTP request, it passes the request to public subnet via External Load Balancer. Public Subnet has an Auto Scaling group which will scale out and in the instances as per the traffic using AMI.

As web pages are only responsible to visualize the content, we have kept them in Public Subnet. Business logic and data processing is done into Application Tier, so we have kept them in Private Subnet. This subnet receives the TCP request from Web Tier via Internal Load Balancer and Scale-out and in using Auto Scaling Group.

Database tier receives requests from App tier. It contains almost everything like User data, Doctors Data, Transactional information, patients and doctor’s documents etc. This whole system is dependent on the database to provide the services.

Here we have used two types of database, RDS and Dynamo DB. To make the RDS database always available we have used Master-Slave approach along with replica. Replica database is used for taking backup periodically.

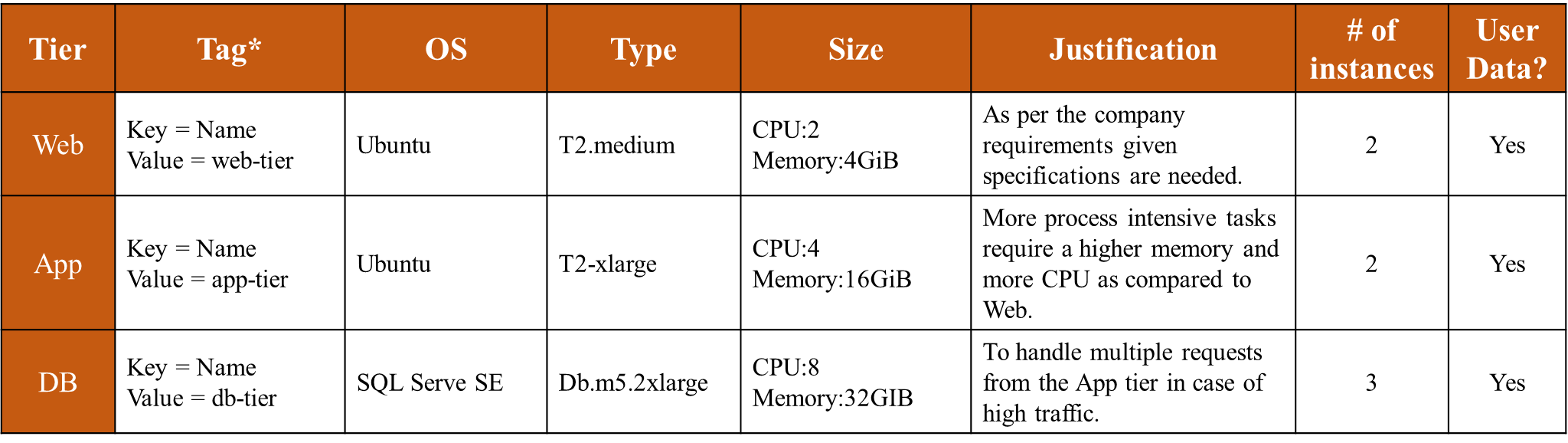
Dynamo DB is mainly used to store the extracted text from Images. When a user uploads some image file as a document, it goes to S3 bucket and then Textract service extracts the text from images and put them into Dynamo DB. These extracted texts are accessed by app tier which can be further processed and stored in structured/tabular format in RDS.

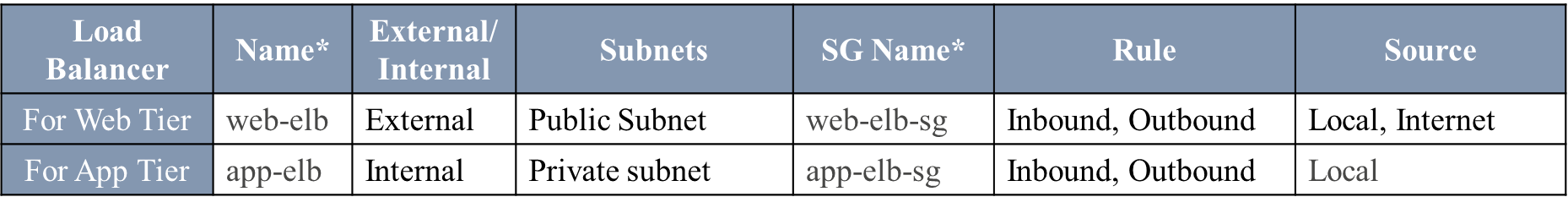
We have used S3 to store media files submitted by patients or doctors, Snapshot of Databases and Instances used in web and app tiers. This media file will be securely available on the web page with the help of KMS using CloudFront. Snapshot is an exact copy of the working system. We store the snapshot of the Databases and EC2 instances which provide flexibility to UP the system exact point in time when everything crashes.

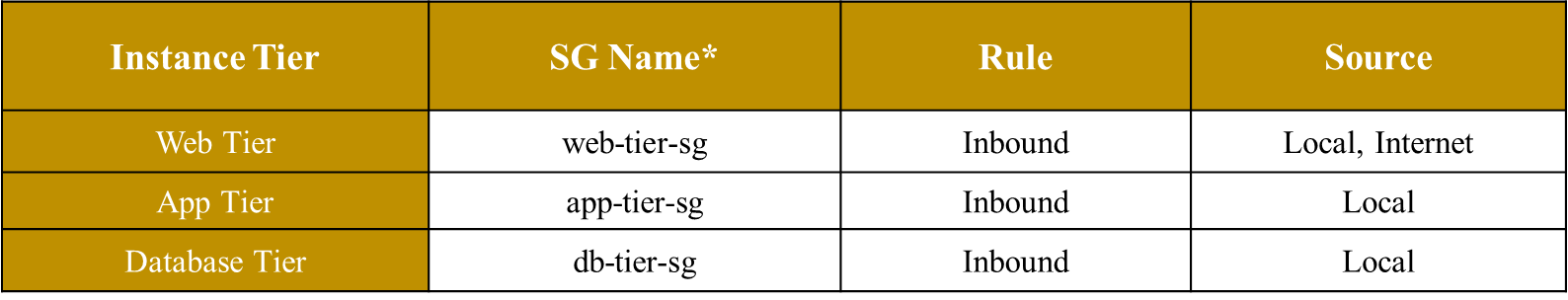
A close up of a map

Description automatically generated

We have used SNS to send a notification which will trigger other services like Textract when images will be uploaded by patients. This architecture supports secure payment transaction using third party application i.e. Stripe. Also used cloud trail and cloud watch to keep track of abnormal behaviour of the architecture and its contents.

****

****

****

**Business Continuity**

Our architecture requires to recover from disaster and resume operation with little disruption. Cloud workloads needs to reliable and always available.

For a medical startup company, our AWS solution includes the following services to maintain business continuity-

**For High Availability:**

Route53: As the company must cover America, Europe and Asia Pacific region, this service increases our reachability. It also helps divert the traffic the incoming traffic w.r.t. latency and health of our instances as per the region.

Elastic Load Balancer: This service helps distribute the incoming traffic within our servers. As it is important to use our server capacity efficiently.

Auto-Scaling Group: During scenarios where the company observers increased in incoming traffic this service can help increase the available instances for cope up with the traffic. With scalability this service also provides elasticity to our architecture.

**For Snapshots:**

Amazon Machine Image: This service is essential in providing the necessary snapshots to aid support Auto-Scaling group. It stores the images of instances which would be scaled.

**For Data Replication:**

We have designed a Master-Slave-Replica architecture for our database to maintain business continuity. Slave DB can take control during instances where Master DB fails. The Replica DB helps to generate data backup which can be used if both the DB fail.

**Auditing**

Auditing of AWS architecture is necessary to maintain security and helps us to build a compliant Cloud Architecture. We have used the following services to secure and maintain privacy-

CloudWatch: As this service works seamlessly Amazon S3, Elastic Load Balancing or Amazon EC2. It collects monitoring and operational data in form of logs, metrics and events to provide a unified view of AWS resources to detect anomalous behavior, set alarms, automated actions and troubleshoot.

CloudTrail: This service enables governance, compliance, operational auditing, and risk auditing of your AWS account. With CloudTrail, one can log, continuously monitor, and retain account activity related to actions across your AWS infrastructure.

**Next Steps and Conclusion**

We have proposed here our AWS architect solution for a medical startup company. They wanted to migrate their traditional IT infrastructure to the cloud to make their system always available, robust and scalable at low cost.

To achieve all requirements of the company, we took a three-tier architectural approach which supports multi-region and seamless integration from development stages across all regions. We configure the instances for Web, App and Databases as per the traditional system configuration which would be enough to handle the traffic with The load balancer and Autoscaling group.  In networking, we have 65536 total number of IP addresses which is distributed across the VPC and had kept enough number of IP address for future purpose. As the company will grow, they will expand the features and services and they won’t need to redesign the IP distribution.

Architecture is well designed to be highly available. We have kept Snapshot of all instances which will help to UP the system exact point in time when everything crashes.

This AWS architecture would be the best cloud infrastructure for a startup company which has the potential to grow very fast. It can be modified easily when required and cost would be optimum for the AWS services and maintenance.