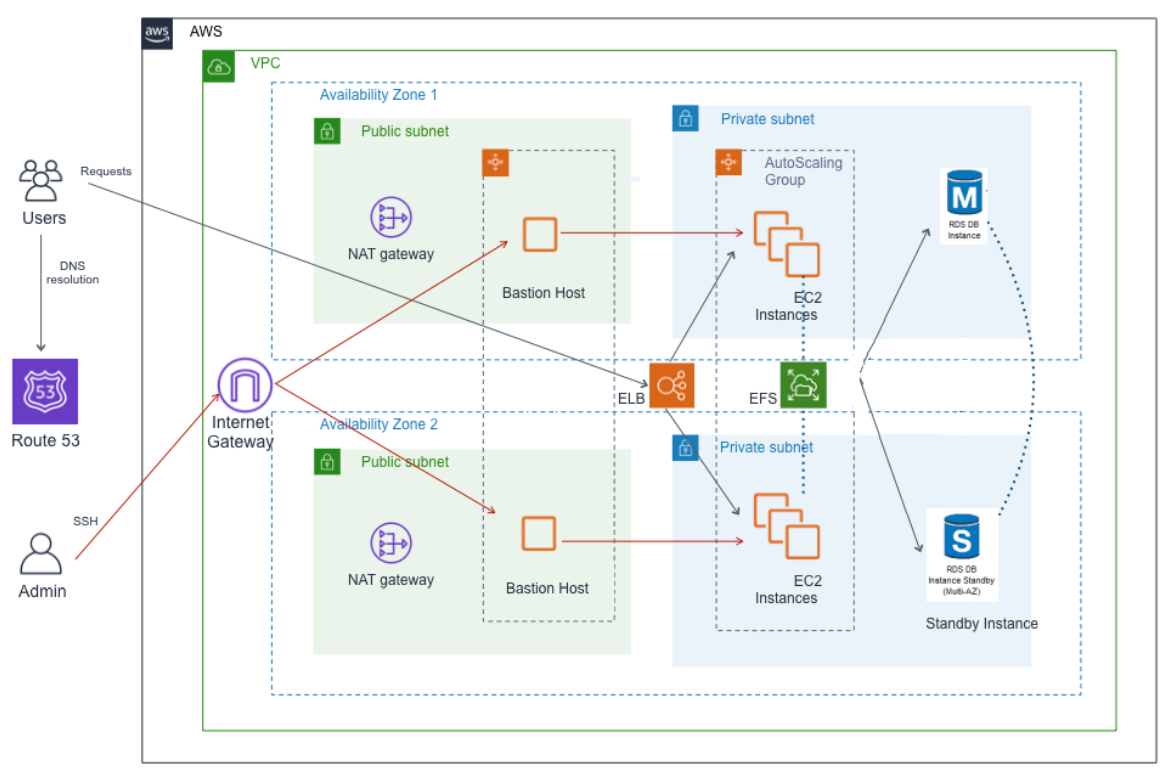
**Solution to Case Study Symbiosis**

**Table of Contents:**

1. **Proposed Solution 1**
2. **Proposed Solution 2**
3. **URL to Web Application**
4. **Recommended solution**
5. **Proposed Solution -1:**

Below is the architecture diagram representation of Solution1:



**Figure 1: Architecture Diagram Solution 1**

**Explanation of Solution 1:**

**Requirement**: A **highly-available (HA)** architecture that spans two Availability Zones in the AWS region.

**Solution**: These would be three main elements of the solution as shown above:

1. A **Virtual Private Cloud (VPC)** with public and private subnets.
2. An **Internet Gateway** that allows access to Bastion hosts.
3. An **Elastic Load Balancer (ELB)** to distribute HTTP and HTTPS requests across the application instances.

**Recommendation**: Classic Load Balancer can perform Load balancing in Layer 4 and layer 7. However, **it doesn’t support features like Host and path-based routing**. It is also only recommended when we are using EC2 Classic Instances. Hence, my recommendation is to use **Application Load balancer** which operate on Layer 7 and support features like Host and path-based routing

**Elements in the public subnets are:**

1. EC2 instances to act as Bastion hosts
2. Managed NAT Gateways to allow outbound traffic to application instances and DB instances.

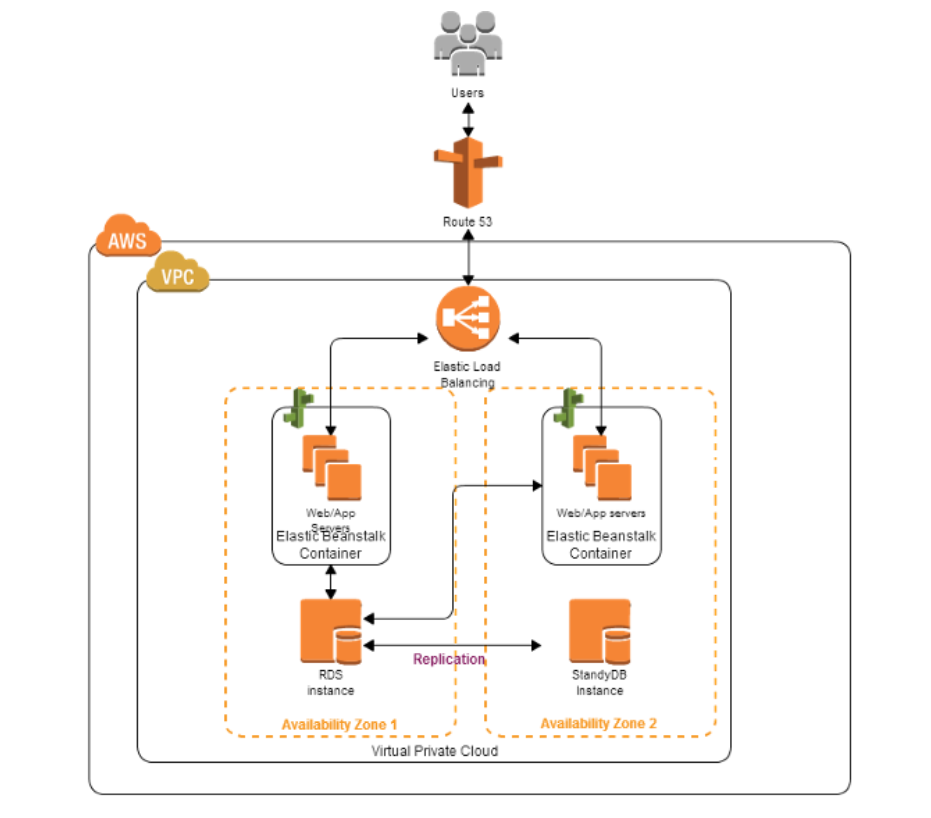
**Elements in the private subnets are:**

1. EC2 instances that host the application servers. An Auto Scaling group to ensure high availability for these application server EC2 instances.
2. An Elastic File System (EFS) to share assets across instances.
3. MYSQL DB instances (via Amazon RDS) with multi-AZ enabled. It’s a fully managed service so user need not have to worry for hardware provisioning, database setup, patching and backup.

**Performance Benefits:**

1. HA architecture that ensures your cluster can continue working if a node (or a whole AWS zone) is down.
2. Replication model allows workload distribution across multiple application services.
3. The Auto Scaling Group allows you to scale dynamically the number of on-demand instances.
4. Application instances are isolated in the private subnets. By configuring the corresponding security groups, inbound traffic is exclusively allowed from Bastion hosts, through Secure Shell Access (SSH) on port 22 and the Elastic Load Balancing, HTTP and HTTPS traffic on ports 80 and 443.
5. **Proposed Solution -2:**

Below is the architecture diagram representation of Solution2:



**Figure 2: Architecture Diagram Solution 2**

**Explanation of Solution 2:**

In the second solution I will suggest to use Elastic Beanstalk service to handle below:

1. Auto-scaling
2. Load Balancing
3. Any other required services/ support.

AWS Elastic Beanstalk is an easy-to-use service for deploying and scaling web applications and services developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker on familiar servers such as Apache, Nginx, Passenger, and IIS.

If the Infrastructure is created using Beanstalk services then no additional monitoring is required as it will be managed by Elastic Beanstalk. Only we would be required to only upload the code/ provide the S3 bucket path and Elastic Beanstalk will automatically handles the deployment, from capacity provisioning, load balancing, auto-scaling to application health monitoring. At the same time, you retain full control over the AWS resources powering your application and can access the underlying resources at any time. We can create RDS database from elastic Beanstalk console or can create our own database and then can configure the same in Beanstalk.

There is no additional cost for using Beanstalk, you are only required to pay charges for the underlying infrastructure used/deployed by Beanstalk.

**Note:** We can also create RDS replica if require to reduce load on Master Databases and can use elasticache for caching database queries and Amazon cloud front to minimize latency and enhanced user experience for worldwide user base.S3 bucket is always required to keep your code and additional healthcare equipment information/ data, content or images to be used by application.

1. **URL to Web Application**

<http://case-study-1086127650.us-east-1.elb.amazonaws.com/>

1. **Recommended Solution:** Proposed Solution 2 is a better solution.