# Task 1 – Customer Requirements

Step 1: Address customer requirements (Medi-Advice):

1. Extremely available and scalable architecture:

* We need to make sure that Web and App tiers are scaling automatically with demand.
* Database should have repetition and mechanism to handle the failures in case it happens.
* Meet “Five nines” (99.999%) accessibility.

1. Global performance Improvements:

* Reduce the time of response in both US and Ireland users.
* Improve the accessibility of global product handouts using content delivery network (CDN) which speeds webpage loading.

1. Cost-Effectiveness:

* Use AWS services efficiently aiming minimal costs
* Use AWS free-tier services where applicable.

1. Automation:

* Apply auto scaling processes, failovers, and most important recovery options.

1. Security:

* Make sure communication end to end is secure and isolate tier using private or public subnets depending on the case.
* Make sure databases and backend services are secure.

Step 2: Identify AWS services and products that can be used in Medi-Advice:

Main Features:

* ELB (Elastic Load Balancer): Can be used to handle the distribution of the traffic across web and application servers.
* ASG (Auto Scaling Group): Can be used to handle the high traffic demand the company is facing.
* Amazon EC2: Can be used to host the web and application tiers with instances operating Linux.
* Amazon RDS: Might be used to manage database services matching AZ redundancy.
* Amazon CloudFront: To handle Content Delivery Network, serving as advertisements globally.

Assisting Feature Services:

* Amazon S3 Bucket: Mainly for files and backups storage.
* AWS Lambda: Can be used to automate document processing.
* Amazon Route 53: Safe route for DNS and cross-region disruptions.
* AWS VPC: Will be used to provide network partition with private or public subnets.
* IAM: For accessibility control.

Step 3: The architecture proposal for Medi-Advice in Tiers:

1. Web Tier:

* Create instances in an Auto Scaling Group using EC2.
* Make use of Elastic Load Balancers to manage the traffic flow and integrate our CloudFront for caching.

1. App Tier:

* Separate private EC2 instances in an ASG.
* Ensure security with NACLs layers for VPC and security groups.

1. Global Content Delivery:

* Use S3 Bucket with CloudFront as the origin of serving brochures.

1. Database Level:

* Amazon RDS (multi-AZ deployments) for higher availability.
* Leverage read copies to offload read traffic.

1. Automation:

* Implement CloudWatch to real time monitoring and watching scaling policies.
* Using the Lambda to automate backups and failover processes.

1. Cost-Effectiveness optimized:

* We will use T2 or T3 instances for cost-benefit.
* Also enable reserved instances or some saving plan available for predictable workloads.

# Task 2 – Architecture Definition:

A diagram for this architecture will be included for all the components of the architecture:

1. Network Layer:

* VPC that supports:
  + Two different zones (AZs), to handle US and Ireland.
  + Public and private subnets in each AZ.
  + An Internet Gateway (IGW) for allow public access to the web and app.
  + NAT gateway to manage the private subnet internet access.
  + Routing tables for the subnets, controlling the traffic load.

1. Web Layer:

* Instances (EC2) using an Auto Scaling Group (ASG).
* Application Load Balancer (ALB) together with public subnets, this way distributing incoming traffic.

1. Application Layer:

* Security Groups controlling the access to only the Web layer.
* Private instances using (EC2) an Auto Scaling Group.

1. Database Layer:

* Amazon RDS with Multi-AZ launch.
* Install and prepare a security group that allows only App Layer to access.

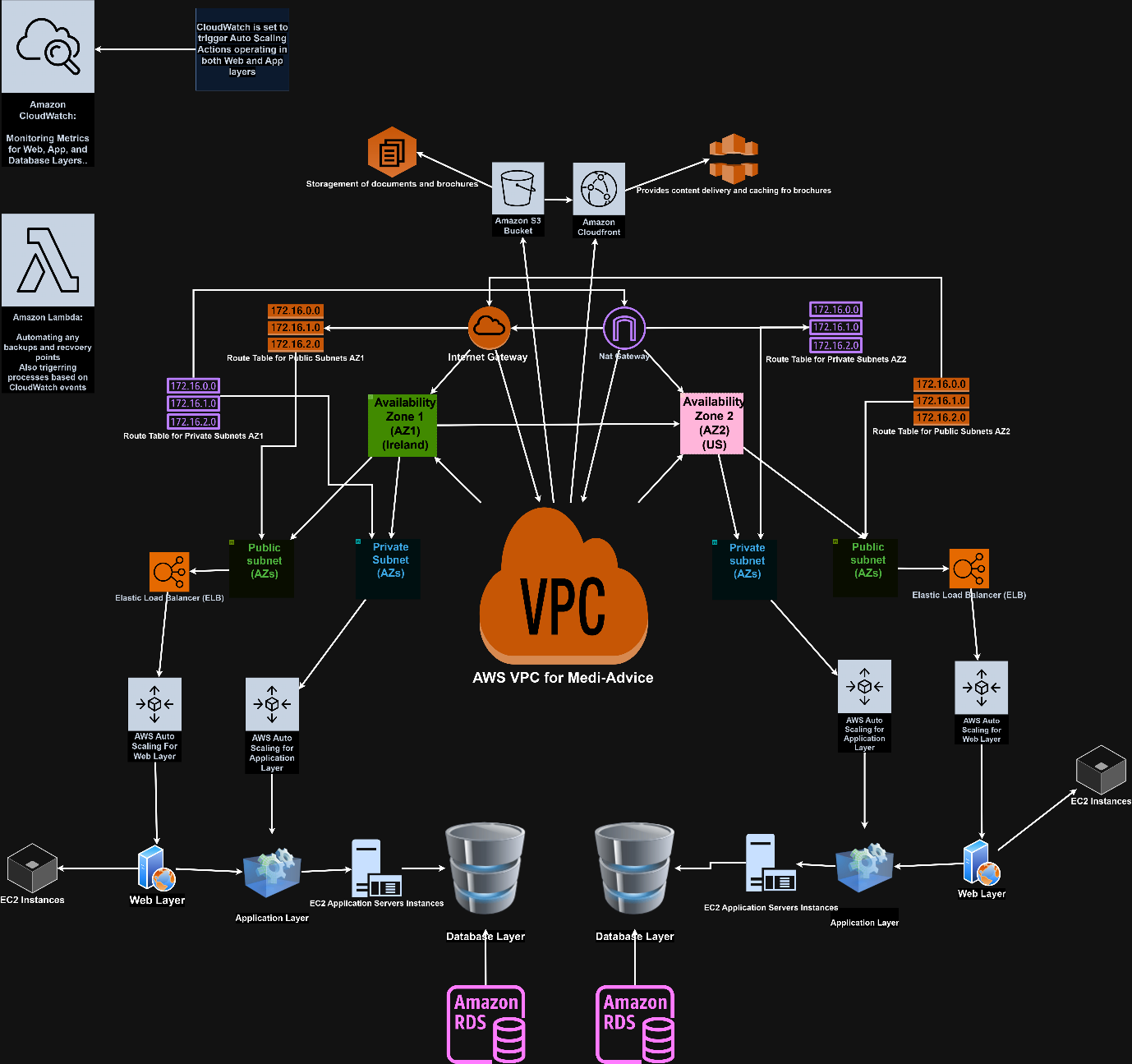
1. Content Processing and Storage:

* Featuring Amazon Cloud Front for global caching.
* Amazon S3 bucket for brochure saving.

1. Automation and Monitoring Processes:

* CloudWatch technology for monitoring all the metrics and scaling.
* AWS Lambda to be used as task automation.

Diagram:



# Task 3 – Exploring AWS best practices

## Task 3a: The Anti-Patterns

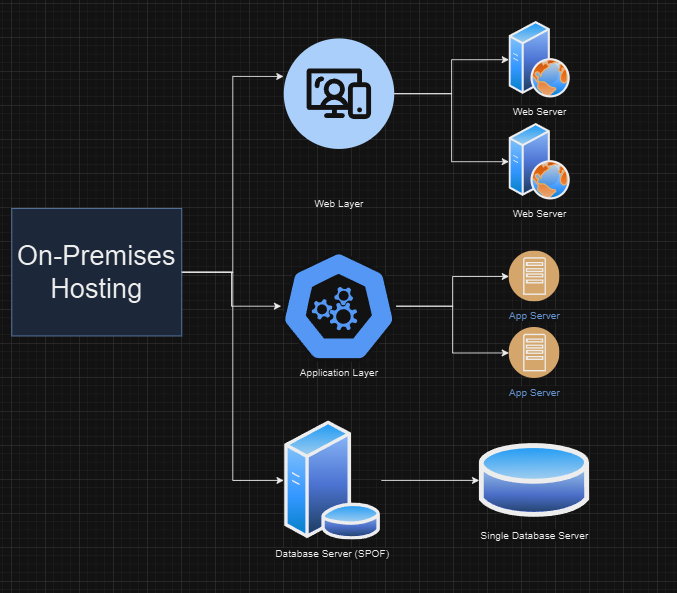
Anti-Pattern N1: SPOF, which stands for Single-Point of Failure in the Database

* **Problem:** The database server in use is a single instance. If it fails, the application servers will not work as they will not be able to retrieve, read or store data, leading to a potential downtime of the application.
* **Solution:** Use Amazon RDS with Multi-AZ deployment to create a replica of the database automatically across the reachability zones, making sure then it will be available for use if any downtown from failure comes.

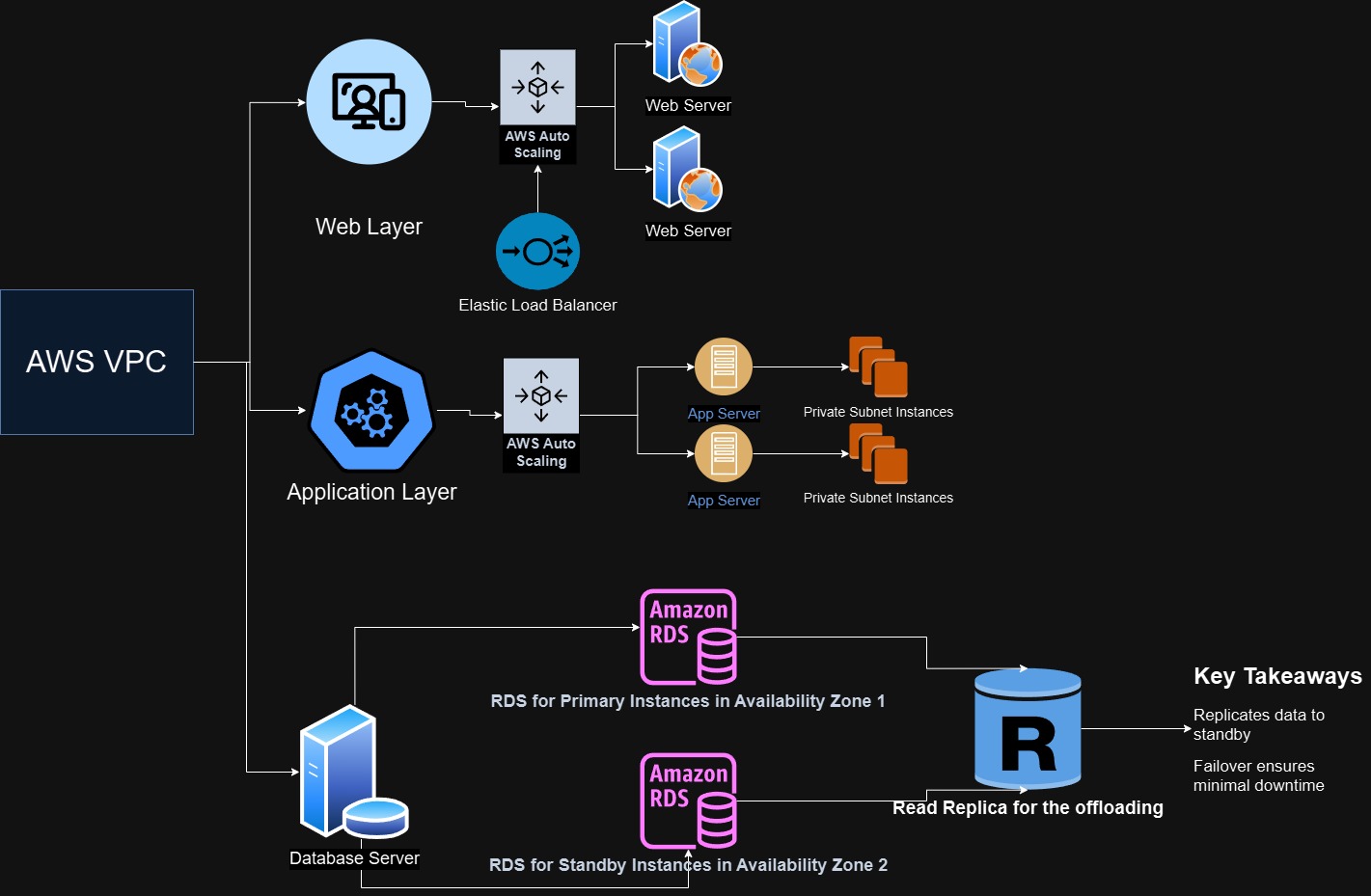
Another measure to be taken would be create read replicas to take out the traffic load of read operations, which would make it more scalable.

Here is a diagram demonstrating the problem-solution to clarify:

Problem



Solution



Anti-Patter N2: Manual Scaling of Web Servers

* **Problem:** The current system needs a technician to manually create or delete web servers based on demand, which makes the whole process slow since you will be exposed to factors such as human error.
* **Solution:** Iterate an Auto Scaling Group (ASG) in AWS to handle the scalability of EC2 instances automatically based on traffic income demand.

Join both ASG with CloudWatch measurements for metric (E.G., CPU utilization, RAM utilization, traffic) to automate scaling and make sure the operation will run uninterrupted during high peak times.

Here is a diagram demonstrating the problem-solution to clarify:

Problem

A diagram of a server

Description automatically generated

Solution

A computer screen shot of a diagram

Description automatically generated

## Task 3b: Cost and Durability Optimization

Using the 6 pillars of the AWS Well-Architected Framework we will give an explanation-solution for this task:

1. Operational Excellence

* Add monitoring and logging features using AWS CloudWatch and AWS CloudTrail to ensure visibility into the performance statistics and its operations into the system.
* Use some practices from Infrastructure as Code (IaC) tools like AWS CloudFormation for reliable, quotable deployments.

1. Security

* Isolate all the layers using private and public subnets.
* Make sure the communication with the Security Groups and Network Access Control Lists (NACLs) is secure and reliable.
* Secure the data in transit and at rest (E.g., S3) by encrypting it.

1. Reliability

* Implement the use of Route 53 for DNS-based failover and cross-region failures, this way you will have a safe space if anything goes wrong, we can take that as a precaution method.
* Implement Elastic Load Balancers to share and distribute traffic across all Availability Zones.
* Make the use of Multi-AZ deployments for RDS and important components.

1. Performance Efficiency

* Use CloudFront to deliver content faster in a global basis.
* Upgrade EC2 instance types, make sure to pick the ones with cost-efficiency such as T2 or T3 or M5 if heavier loads are going to be present.

1. Cost Optimization

* Make sure to resize EC2 instances and RDS configurations meeting all the needs and criteria.
* Approach the options for saving plans or reserved instances for apparent workloads.
* Whenever possible, apply AWS free tier plans.

1. Sutainability

* Use AWS services in regions that are closer to users, this way reducing the latency and energy consumptions.
* Set up some ASG policies aiming to reduce scalability of unused services whenever possible.

# Task 4 – VPC Creation and Demonstration and its Components