DevOps

[Route53 2](#_Toc193691391)

[Understanding AWS Route 53 and DNS Configuration 2](#_Toc193691392)

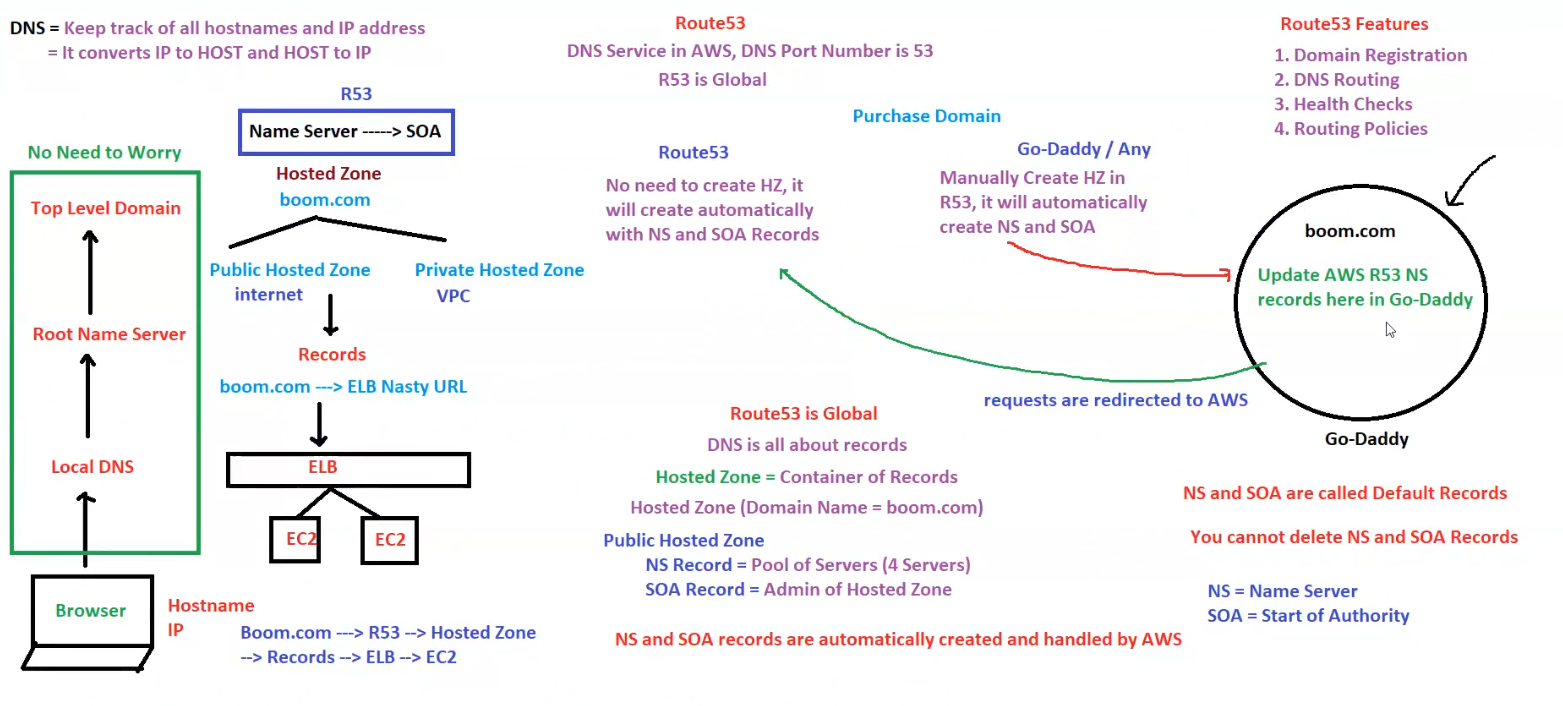
[AWS Route 53 DNS Records 3](#_Toc193691393)

[Route 53 Routing Policies 6](#_Toc193691394)

[Geo-Location Routing 7](#_Toc193691395)

[Latency-Based Routing Policy 8](#_Toc193691396)

# Route53



## Understanding AWS Route 53 and DNS Configuration

**1. DNS Basics**

* **DNS (Domain Name System):** Keeps track of all hostnames and IP addresses.
  + Converts IP to Hostname and Hostname to IP.

**2. Route 53 Overview**

* **Route 53 (R53) is AWS's DNS service.**
* **DNS Port Number:** 53.
* **Route 53 is a global service.**
* **Key Features:**
  1. **Domain Registration**
  2. **DNS Routing**
  3. **Health Checks**
  4. **Routing Policies**

**3. Understanding Hosted Zones (HZ)**

* **Hosted Zone:** A container of DNS records.
* **Public Hosted Zone:** Accessible via the internet.
* **Private Hosted Zone:** Restricted to a VPC.

**4. How DNS Resolution Works in AWS**

* **Domain Name:** boom.com (Example)
* **Steps in DNS resolution:**
  1. **User enters boom.com in the browser.**
  2. **DNS request flows through the hierarchy:**
     + Local DNS → Root Name Server → Top-Level Domain (TLD) Server.
  3. **Route 53 handles the domain via the Hosted Zone.**
  4. **Records in the Hosted Zone map boom.com to an ELB (Elastic Load Balancer) URL.**
  5. **ELB directs traffic to EC2 instances.**
  6. **Response is sent back to the user.**

**5. Name Server (NS) & SOA (Start of Authority)**

* **NS (Name Server) Record:**
  + Pool of four servers that handle DNS queries.
* **SOA (Start of Authority) Record:**
  + Admin details for the Hosted Zone.
* **These records are created automatically by AWS and cannot be deleted.**

**6. Purchasing a Domain and Integrating with Route 53**

* **Domains can be purchased from providers like GoDaddy.**
* **Steps to use Route 53 with an external domain:**
  1. **Purchase a domain from GoDaddy or any provider.**
  2. **Manually create a Hosted Zone in Route 53.**
  3. **AWS automatically generates NS and SOA records.**
  4. **Update NS records in GoDaddy with AWS Route 53 NS records.**
  5. **Requests will then be redirected to AWS for resolution.**

**7. Important Notes**

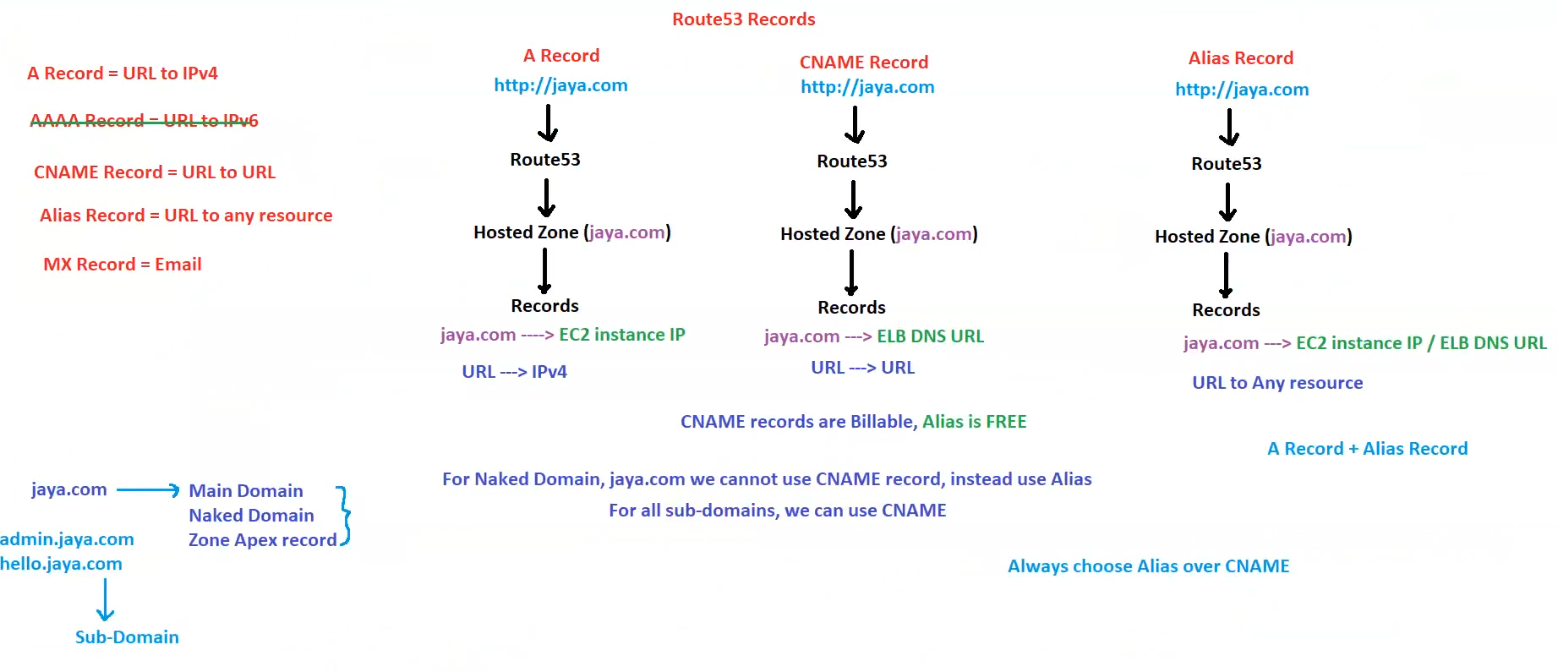
* **Route 53 automatically creates Hosted Zones with NS and SOA records.**
* **NS and SOA records are default and cannot be deleted.**
* **DNS in AWS is all about records mapping domains to resources.**

## AWS Route 53 DNS Records

**Understanding AWS Route 53 DNS Records**

**1. Types of DNS Records**

* **A Record:** Maps a URL to an IPv4 address.
* **AAAA Record:** Maps a URL to an IPv6 address (crossed out in the image, meaning it's not a focus here).
* **CNAME Record:** Maps a URL to another URL.
* **Alias Record:** Maps a URL to any AWS resource.
* **MX Record:** Used for email routing.



**2. Route 53 DNS Record Types & Their Functionality**

**A Record (Address Record)**

* Example: http://jaya.com
* **How it works:**
  + Route 53 processes the request.
  + Hosted Zone (jaya.com) contains the record.
  + Maps jaya.com to an **EC2 instance's IP**.
  + Translates a URL to an **IPv4** address.

**CNAME Record (Canonical Name)**

* Example: http://jaya.com
* **How it works:**
  + Route 53 processes the request.
  + Hosted Zone (jaya.com) contains the record.
  + Maps jaya.com to an **ELB DNS URL**.
  + Translates a URL to another **URL**.
* **Key Points:**
  + **CNAME records are billable.**
  + **CNAME cannot be used for the root domain (naked domain, e.g., jaya.com).**
  + **CNAME can be used for subdomains (e.g., admin.jaya.com, hello.jaya.com).**

**Alias Record**

* Example: http://jaya.com
* **How it works:**
  + Route 53 processes the request.
  + Hosted Zone (jaya.com) contains the record.
  + Maps jaya.com to **any AWS resource (EC2 instance, ELB, etc.)**.
  + Can resolve both **IP addresses and URLs**.
* **Key Points:**
  + **Alias records are free (unlike CNAME records).**
  + **Alias records can be used for naked domains (e.g., jaya.com).**
  + **Recommended to always use Alias over CNAME where possible.**
  + **Combines functionality of A Record and Alias Record.**

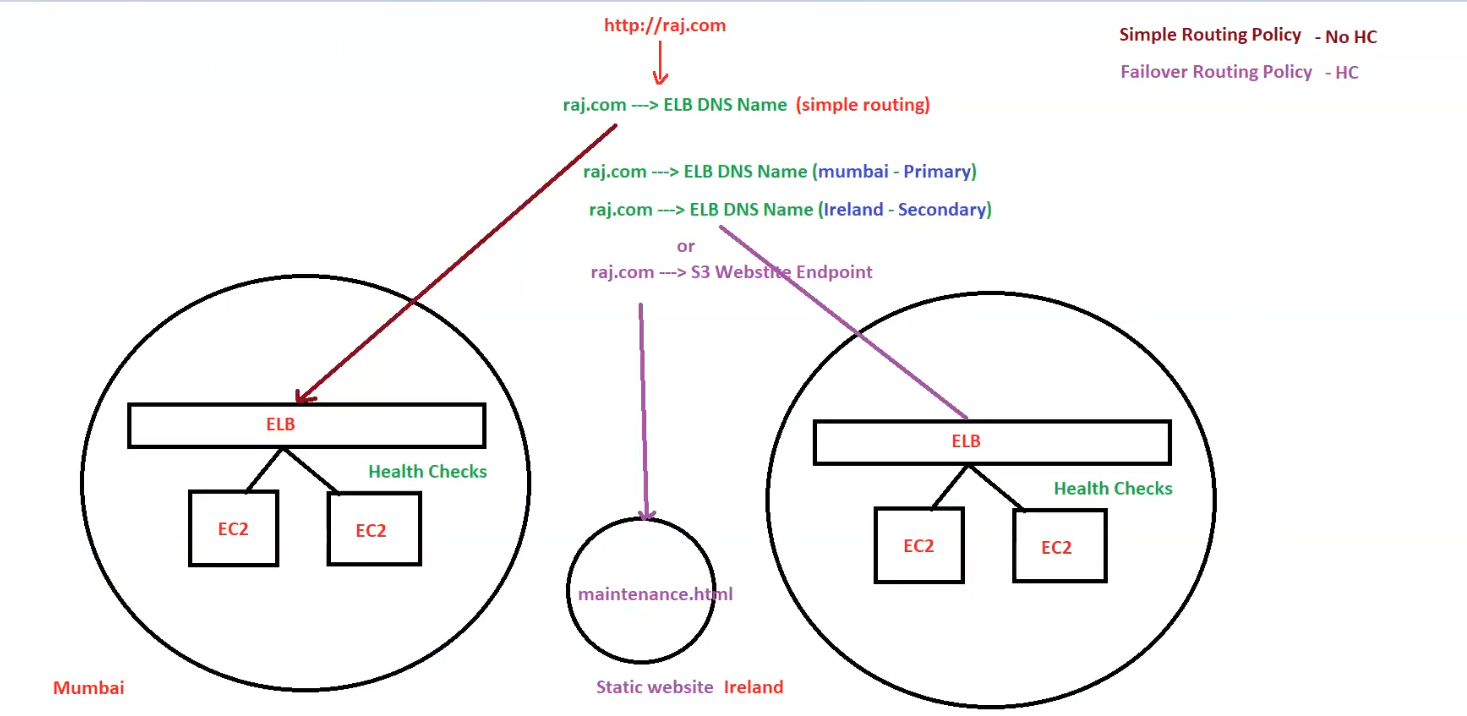
**3. Domain & Subdomain Structure**

* **Main Domain (Naked Domain / Zone Apex Record)**
  + jaya.com
* **Subdomains**
  + admin.jaya.com
  + hello.jaya.com
* **Important Considerations:**
  + For the **naked domain** (jaya.com), **CNAME cannot be used**, so **Alias should be used instead**.
  + For **subdomains**, CNAME can be used.

**4. Best Practices**

* **Always choose Alias over CNAME where possible.**
* **Use Alias for the root domain (naked domain).**
* **Use CNAME for subdomains when required.**
* **Remember that CNAME records are billable, whereas Alias records are free.**

## Route 53 Routing Policies



**1. Simple Routing Policy (No Health Check - HC)**

* **raj.com** is mapped to an **ELB DNS Name**.
* Traffic is routed to a single **Elastic Load Balancer (ELB)** without any health checks.
* If the primary ELB (Mumbai region) goes down, there is no automatic failover.

**2. Failover Routing Policy (With Health Checks - HC)**

* raj.com is mapped to:
  + **Primary ELB in Mumbai**
  + **Secondary ELB in Ireland**
* **Health Checks (HC) are enabled**:
  + If the **Mumbai ELB** fails, Route 53 automatically redirects traffic to the **Ireland ELB**.
  + Ensures high availability.
* Alternative Failover Option:
  + Instead of an ELB, traffic can be redirected to an **S3 static website endpoint** (e.g., maintenance.html).
  + Useful when the entire infrastructure is down, and users need a maintenance page.

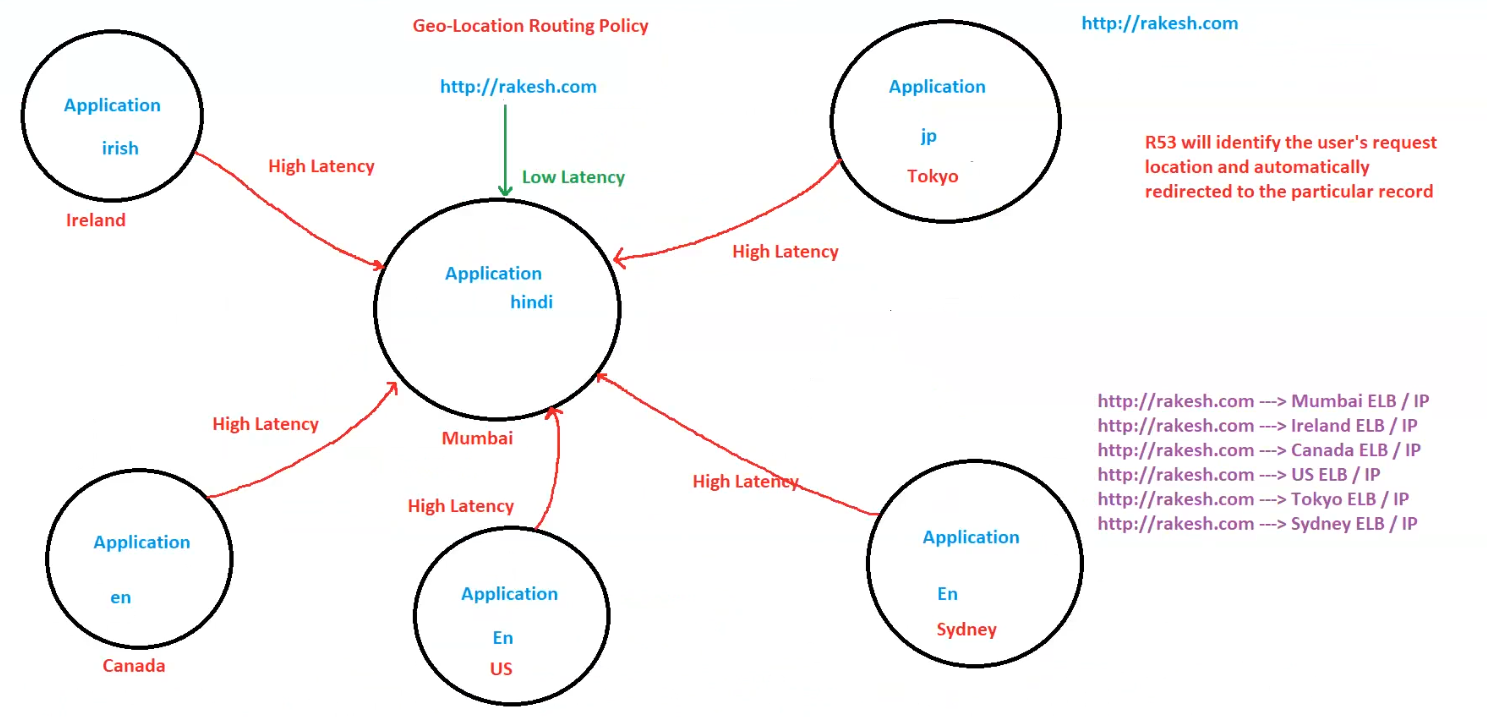
**3. Components in the Diagram**

* **Mumbai Region:**
  + **ELB** distributes traffic to two EC2 instances.
  + Health checks ensure EC2 instances are healthy.
* **Ireland Region (Failover Setup):**
  + **ELB** distributes traffic to EC2 instances.
  + Acts as a backup if Mumbai ELB fails.
* **S3 Static Website (Fallback Option):**
  + If both ELBs fail, traffic is directed to a maintenance page hosted on S3.

**4. Key Takeaways**

✅ **Simple Routing:** Directs traffic without failover. No health checks.  
✅ **Failover Routing:** Uses health checks to detect failure and switch to backup.  
✅ **Best Practice:** Use **failover routing** for high availability.  
✅ **Alternative Failover:** Redirect to **S3 static site** in case of a major outage.

## Geo-Location Routing



**1. How Geo-Location Routing Works**

* **Route 53 identifies the user’s request location** and automatically redirects them to the appropriate application instance.
* Each region has a specific **Elastic Load Balancer (ELB) / IP**, and Route 53 ensures users connect to the best region for their location.

**2. Traffic Flow Based on User Location**

* **User from India (Mumbai region)** → Routed to **Mumbai ELB (Low Latency)**
* **User from Ireland** → Routed to **Ireland ELB**
* **User from Canada** → Routed to **Canada ELB**
* **User from the US** → Routed to **US ELB**
* **User from Tokyo (Japan)** → Routed to **Tokyo ELB**
* **User from Sydney (Australia)** → Routed to **Sydney ELB**

Users accessing from locations far from Mumbai may experience **high latency**, and Route 53 ensures they get connected to a closer region.

**3. Key Benefits of Geo-Location Routing**

✅ **Improved Performance** - Users are redirected to the nearest region for the fastest response time.  
✅ **Localization** - Different applications can be served based on language (e.g., Hindi in India, Japanese in Tokyo).  
✅ **Compliance & Regulations** - Certain countries may require data to be stored and processed within specific regions.  
✅ **Traffic Distribution** - Helps balance traffic across multiple geographic locations.

**4. Summary of ELB Routing in the Image**

* [**http://rakesh.com**](http://rakesh.com/) **→ Mumbai ELB / IP** (Primary)
* [**http://rakesh.com**](http://rakesh.com/) **→ Ireland ELB / IP**
* [**http://rakesh.com**](http://rakesh.com/) **→ Canada ELB / IP**
* [**http://rakesh.com**](http://rakesh.com/) **→ US ELB / IP**
* [**http://rakesh.com**](http://rakesh.com/) **→ Tokyo ELB / IP**
* [**http://rakesh.com**](http://rakesh.com/) **→ Sydney ELB / IP**

**🚀 Best Practice:**

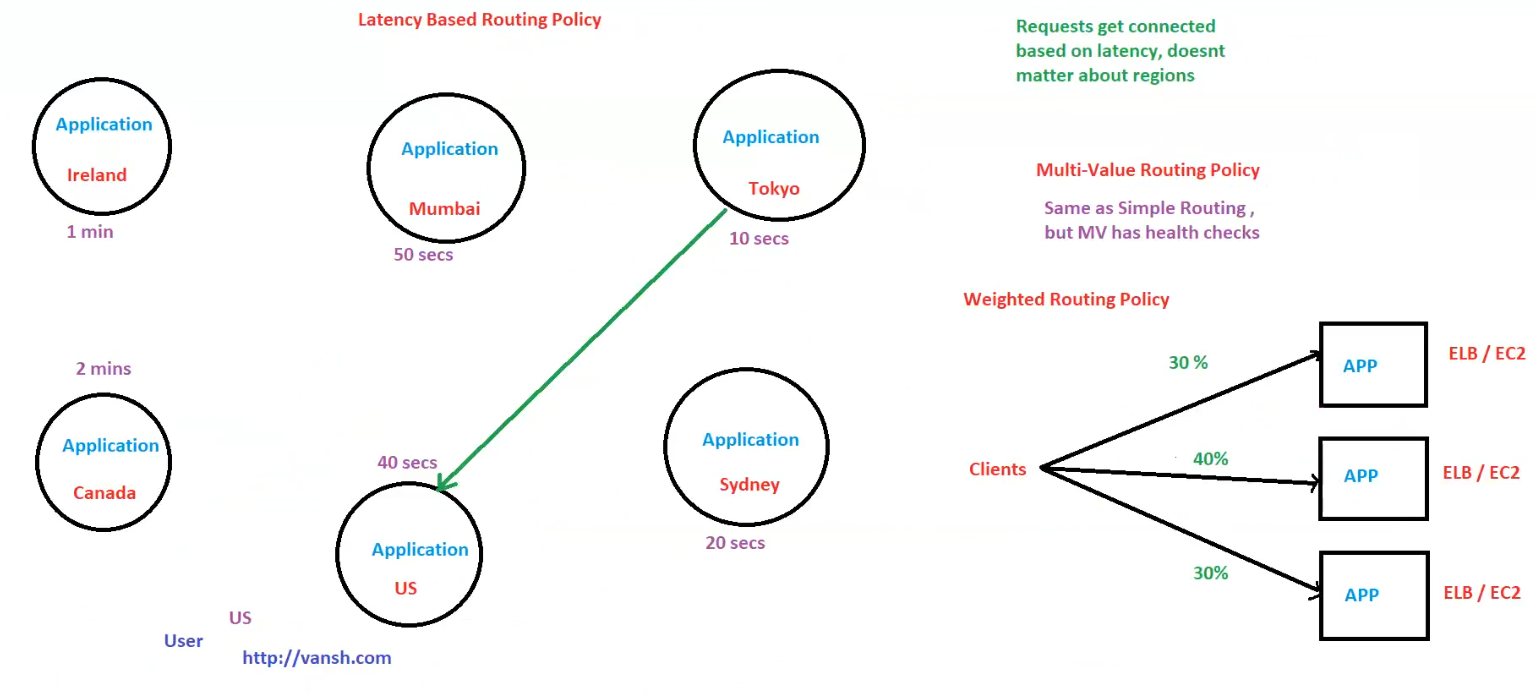
For users with **dynamic locations**, consider using **Latency-Based Routing** instead of Geo-Location Routing to direct them to the lowest latency endpoint dynamically.

## Latency-Based Routing Policy

**1. Latency-Based Routing Policy**

* Requests are routed **based on latency**, regardless of region.
* The user is directed to the application with the **lowest response time**.
* In the diagram:
  + **Tokyo (10 secs)** → **Fastest**
  + **Sydney (20 secs)**
  + **US (40 secs)**
  + **Mumbai (50 secs)**
  + **Ireland (1 min)**
  + **Canada (2 mins)** → **Slowest**
* **Example**:
  + A user in the US accessing [**http://vansh.com**](http://vansh.com/) will be routed to **Tokyo (10 secs)** instead of the US application (40 secs), since **Tokyo has lower latency.**

✅ **Best for**: Performance optimization, ensuring users get connected to the fastest application.



**2. Multi-Value Routing Policy**

* **Similar to Simple Routing but includes Health Checks**.
* Ensures requests are only routed to healthy endpoints.
* If an endpoint is **unhealthy**, it is removed from the routing list.
* **Example**:
  + If **Tokyo** becomes **unhealthy**, Route 53 will direct traffic to **Sydney (20 secs)** instead.

✅ **Best for**: Load balancing with health checks to improve availability.

**3. Weighted Routing Policy**

* Distributes traffic among multiple resources based on assigned weights.
* Traffic is split using percentages.
* **Example in the image**:
  + **30%** of users → **First APP (ELB/EC2)**
  + **40%** of users → **Second APP (ELB/EC2)**
  + **30%** of users → **Third APP (ELB/EC2)**

✅ **Best for**:

* Gradual traffic migration.
* A/B testing or Blue-Green Deployments.
* Handling traffic spikes by distributing load.

**🚀 Summary**

* **Latency-Based Routing** → Directs traffic to the lowest latency region.
* **Multi-Value Routing** → Works like simple routing but considers health checks.
* **Weighted Routing** → Distributes traffic based on set percentages.