1. Configure VPC peering in cross regions.

**Step 1: Create VPC Peering Connection**

1. **Login to AWS Console.**
2. Go to **VPC Dashboard**.
3. Select **Peering Connections** from the left panel.
4. Click **Create Peering Connection**.
5. **Configure the following:**
   * **Peering connection name**.
   * **Local VPC (Requester region and VPC ID)**.
   * **Remote VPC (Accepter region and VPC ID)**.
   * Select **Another account** if VPCs are in different AWS accounts.
6. Click **Create Peering Connection**.

A screenshot of a computer

AI-generated content may be incorrect.

**Step 2: Accept Peering Connection**

1. Switch to the **Accepter VPC region/account**.
2. Go to **Peering Connections**.
3. You should see a pending request.
4. Select the request and **accept it**.

A computer screen shot of a computer

AI-generated content may be incorrect.

**Step 3: Update Route Tables**

1. Go to **Route Tables** in both regions.
2. For each VPC's route table:
   * Edit routes.
   * Add a route where:
     + **Destination**: Peered VPC's CIDR block.
     + **Target**: VPC Peering Connection ID.
3. Save routes.

**Step 4: Adjust Security Groups and NACLs**

1. In both VPCs, modify **Security Groups** to allow traffic from the other VPC's CIDR.
2. Ensure **NACLs** allow traffic from the peered VPC

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Create subnets private and public in both regions

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Step 5: Test the Peering**

1. Launch instances in both VPCs.
2. Try **pinging / SSH / HTTP** between the instances using **private IPs**.
3. Ensure traffic is working bidirectionally.

A computer screen shot of a computer program

AI-generated content may be incorrect.

1. Purchase one domain from godaddy.

**✅ Steps to Purchase a Domain from GoDaddy**

**Step 1: Visit GoDaddy**

* Go to <https://www.godaddy.com>

**Step 2: Search for a Domain**

1. In the search bar, enter the domain name you want (e.g., yourdomain.com).
2. Click **Search**.
3. If available, you’ll see a message saying **"Yes, your domain is available!"**.
4. If it's not available, GoDaddy will suggest alternatives.

**Step 3: Add Domain to Cart**

1. Click **Add to Cart** next to your chosen domain.
2. Click **Continue to Cart** (may appear automatically).

**Step 4: Review Options (Optional Add-ons)**

1. Decide if you want:
   * **Domain Privacy Protection** (recommended to hide your personal info).
   * **Email hosting** (optional).
   * **Website builder** (optional).
2. Click **Continue to Cart**.

**Step 5: Create/Sign In to GoDaddy Account**

1. If you don’t have an account, create one using email and password.
2. Or **sign in if you already have a GoDaddy account**.

**Step 6: Checkout and Payment**

1. Review your cart.
2. Choose the **number of years** to register the domain (default is usually 1 year).
3. Enter your **payment information** (credit card, PayPal, etc.).
4. Confirm and click **Complete Purchase**.

**Step 7: Confirmation**

* You will receive an email confirming your domain purchase.
* You can now manage your domain from **GoDaddy Dashboard > My Products**.

A screenshot of a computer

AI-generated content may be incorrect.

1. Deploy static webiste in s3.

**Step 1: Create an S3 Bucket**

1. Go to the **AWS Management Console**.
2. Open the **S3 service**.
3. Click **Create bucket**.
4. **Bucket Name**: Enter a **unique bucket name** (ideally matching your domain, e.g., example.com).
5. **Region**: Select your preferred region.
6. **Uncheck "Block all public access"** (you will confirm this in later steps).
7. Acknowledge the warning and **create the bucket**.

A screenshot of a computer

AI-generated content may be incorrect.

**Step 2: Upload Website Files**

1. Click on the created bucket.
2. Go to the **Objects tab**.
3. Click **Upload**.
4. Add your **HTML, CSS, images, etc. files (like index.html)**.
5. Complete the upload.

A screenshot of a computer

AI-generated content may be incorrect.

**Step 3: Enable Static Website Hosting**

1. In the bucket, go to the **Properties tab**.
2. Scroll down to **Static website hosting**.
3. Select **Enable**.
4. Enter:
   * **Index document**: index.html
   * (Optional) **Error document**: error.html
5. Note the **bucket website endpoint URL**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. Create CDN and attach one SSL certificate.

**Step 1: Prepare SSL Certificate (AWS Certificate Manager - ACM)**

1. Go to **AWS Certificate Manager (ACM)**.
2. Ensure you're in **us-east-1 (N. Virginia)** (CloudFront only supports SSL certificates from this region).
3. Click **Request a certificate**.
4. Choose **Public certificate**.
5. Enter your domain name (e.g., example.com, www.example.com).
6. Choose **DNS validation**.
7. ACM will provide **CNAME records**.
8. Go to **your DNS provider (GoDaddy or Route 53)**, and create these **CNAME records**.
9. Wait until the certificate status is **Issued**.

A screenshot of a computer

AI-generated content may be incorrect.

**Create CloudFront Distribution**

1. Go to **CloudFront console**.
2. Click **Create Distribution**.
3. In the **Origin Settings**:
   * **Origin domain**: Enter your **S3 website endpoint** (use the static website endpoint, e.g., your-bucket-name.s3-website-us-east-1.amazonaws.com).
   * **Origin protocol policy**: HTTP only (since S3 static website does not support HTTPS directly).
4. In the **Default cache behavior settings**:
   * **Viewer protocol policy**: Redirect HTTP to HTTPS (forces secure connections).
5. In the **Settings** section:
   * **Alternate domain names (CNAMEs)**: Add your domain name (e.g., example.com, www.example.com).
   * **Custom SSL certificate (example.com)**: Select the certificate you created in ACM.
   * **Price class**: Choose your preferred price class (default is all locations).
6. Click **Create Distribution**.
7. Wait until the distribution status becomes **Deployed** and **Enabled**.
8. Copy the **CloudFront domain name**

A screenshot of a computer

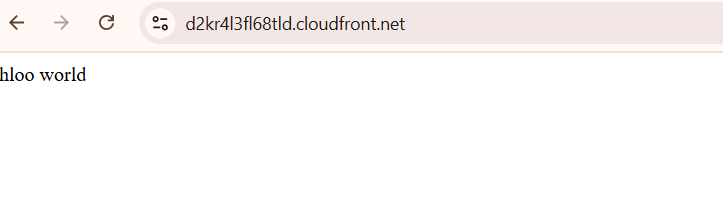
AI-generated content may be incorrect.

A close-up of a computer screen

AI-generated content may be incorrect.

**Test your Setup**

* Should load your **S3 static website over HTTPS via CloudFront**.



1. Create Route53 hosted zone and MAP the domain with CDN.

**✅ Step 1: Create Route 53 Hosted Zone**

1. Go to the **AWS Route 53 console**.
2. Click **Hosted zones**.
3. Click **Create hosted zone**.
4. **Domain name**: Enter your domain name (e.g., example.com).
5. **Type**: Public hosted zone.
6. Click **Create hosted zone**.

A screenshot of a computer

AI-generated content may be incorrect.

**✅ Step 2: Update GoDaddy Nameservers to AWS Route 53**

1. In the hosted zone, you will see **NS (Name Server) records**.
2. Copy these **4 NS records** (they look like ns-123.awsdns-45.com, etc.).
3. Go to **GoDaddy** → **DNS management** for your domain.
4. Click **Change nameservers**.
5. Select **Custom nameservers**.
6. Paste the 4 **AWS Route 53 NS records**.
7. Save changes.

**✅ Now your domain is managed via Route 53.**

A screenshot of a computer

AI-generated content may be incorrect.

**✅ Step 3: Create Route 53 Records to point to CloudFront CDN**

**Create A Record (Alias) for root domain (example.com):**

1. In the **Route 53 Hosted Zone**.
2. Click **Create record**.
3. Choose:
   * **Record name**: (leave empty for example.com).
   * **Record type**: A.
   * **Value/Route traffic to**: Alias to CloudFront distribution.
   * **Select your CloudFront distribution from the dropdown.**
4. Save.

**Create CNAME Record for www:**

1. Click **Create record**.
2. Choose:
   * **Record name**: www.
   * **Record type**: CNAME.
   * **Value**: Your CloudFront domain name
   * **save**

A screenshot of a computer

AI-generated content may be incorrect.

**✅ Step 4: Validate the Setup**

1. Wait a few minutes for DNS propagation.
2. Test:

https://nithinmora.shop/

A close-up of a chat

AI-generated content may be incorrect.

6) Update the index.html in s3 bucket and the updated file should be accessible by using domain name.

**✅ Step 1: Upload (or Replace) index.html in S3 Bucket**

1. Go to **AWS S3 console**.
2. Select your **S3 bucket** (used for the website hosting).
3. Go to the folder (root or appropriate folder) where index.html is located.
4. **Delete or overwrite** the old index.html.
5. Click **Upload**.
6. Choose your updated index.html file.
7. After uploading, **ensure it has public read access**:
   * Select the file.
   * Go to **Permissions** tab.
   * Under **Access Control List (ACL)**, ensure **Everyone (public access) → Read is enabled**.

A screenshot of a computer

AI-generated content may be incorrect.

**✅ Step 2: Invalidate CloudFront Cache (if using CloudFront)**

By default, CloudFront caches your old index.html file, even after you upload a new one.  
You must **invalidate the cache to force CloudFront to fetch the new version**.

1. Go to **CloudFront console**.
2. Select your **CloudFront distribution**.
3. Go to the **Invalidations** tab.
4. Click **Create invalidation**.
5. Enter the following:
   * /\* — to invalidate all cached files, including index.html.
   * Or just /index.html if you only want to invalidate the homepage.
6. Click **Create invalidation**.
7. Wait until the invalidation status shows **Completed**.

A screenshot of a computer

AI-generated content may be incorrect.

**✅ Step 3: Test via Domain Name**

1. Open your browser.
2. Visit your domain
3. You should now see the **updated index.html**.

A white rectangular object with a white border

AI-generated content may be incorrect.

1. Share the Domain name in slack to test the connetivity.

A screenshot of a computer

AI-generated content may be incorrect.