HOST A STATIC WEBSITE ON S3 USING CLOUDFRONT + SSL + CUSTOM DOMAIN

(Layman friendly + interview ready)

FINAL ARCHITECTURE (BEST PRACTICE VIEW)

```
User (Browser)
   ↓
Route 53 (DNS)
   ↓
CloudFront (CDN + HTTPS/SSL)
   ↓
S3 Bucket (index.html, website files)
```

STEP-BY-STEP (FROM ZERO)

STEP 0 — What you need

- A domain (e.g. from **GoDaddy**)
- An AWS account
- A static website (`index.html`)

STEP 1 — Create S3 bucket (Website content)

Service: Amazon S3

Do this

- Create an S3 bucket
- Upload `index.html` and other files

Why

S3 stores the actual website files.

STEP 2 — Enable S3 static website hosting

Do this

- S3 → Properties → Static website hosting
- Enable
- Index document:

```
index.html
```

Why

This allows S3 to behave like a website.

STEP 3 — Create CloudFront distribution (CDN)

Service: Amazon CloudFront

Do this

- Create distribution
- Origin:
    - Type: **S3 static website**
- Leave cache & behavior defaults

Why

CloudFront makes the site fast, global, and secure.

STEP 4 — Set Default Root Object (IMPORTANT)

Do this

- CloudFront → Settings → Edit

- Set:

```
Default root object = index.html
```

Why

Without this, CloudFront doesn't know what file to load at `/`.

STEP 5 — Create Route 53 hosted zone

Service: Amazon Route 53

Do this

- Create **Public hosted zone**
- Domain name: `yourdomain.com`

Why

Route 53 will manage DNS inside AWS.

STEP 6 — Point GoDaddy to Route 53

Do this

- Copy 4 nameservers from Route 53
- Replace GoDaddy nameservers with them

Why

This hands DNS control from GoDaddy to AWS.

STEP 7 — Request SSL certificate (HTTPS)

Service: AWS Certificate Manager

⚠️ **Region MUST be us-east-1**

Do this

- Request **Public certificate**
- Domains:

```
yourdomain.com
www.yourdomain.com
```

- DNS validation
- Let ACM auto-create records in Route 53

Why

CloudFront only accepts certificates from us-east-1.

STEP 8 — Attach domain + SSL to CloudFront

Do this

- CloudFront → Edit distribution
- Add **Alternate domain name**
- Select ACM certificate
- Save

Why

This links your custom domain + HTTPS to CloudFront.

STEP 9 — Route domain to CloudFront (DNS)

In Route 53

Do this

- Create **A record**
- Alias = Yes
- Alias target = CloudFront distribution
- Record name = blank

(Optional)

- Repeat for `www`

Why

This sends users to CloudFront when they open the site.

STEP 10 — Make S3 public (ONLY for website endpoint)

⚠️ Required **only** because origin type = *S3 static website*

10A — Disable block public access

- S3 → Permissions → Block public access
- Uncheck all

10B — Add bucket policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::your-bucket-name/*"
    }
  ]
}
```

Why

CloudFront must be able to read website files from S3.

🎯 YOU ARE DONE

After deployment:

`https://yourdomain.com`

Shows **index.html from S3** over **HTTPS** via **CloudFront**.

TROUBLESHOOTING (REAL-WORLD + INTERVIEW GOLD)

Problem: “Site not reachable”

Cause

- DNS propagation or CloudFront still deploying

Fix

- Wait 10–30 minutes
- Test CloudFront URL directly

Problem: CloudFront URL works, custom domain doesn’t

Cause

- Nameserver propagation delay

Fix

- Wait
- Flush DNS
- Test from another network

Problem: 403 Access Denied

Cause

- S3 bucket is private but using website endpoint

Fix

- Make bucket public OR switch to OAC + REST endpoint

Problem: `/index.html` works but root doesn't

Cause

- Default root object missing

Fix

```
Default root object = index.html
```

INTERVIEW ONE-LINERS (MEMORIZE THESE)

- **S3** → Stores static website files
- **CloudFront** → CDN + HTTPS in front of S3
- **ACM** → Free SSL certificates (us-east-1 for CloudFront)
- **Route 53** → DNS routing to CloudFront
- **GoDaddy** → Registrar only

MASTER INTERVIEW ANSWER ★

"I host static content in S3, serve it through CloudFront with an ACM SSL certificate from us-east-1, and use Route 53 alias records to map the custom domain to CloudFront."

PRODUCTION NOTE (ADVANCED)

For real production:

- Use **S3 REST endpoint**
- Enable **Origin Access Control (OAC)**
- Keep S3 **private**