Timothy Forte
IV- ACSAD

How to Host a Simple Website on AWS

1) What AWS is and its key benefits

Amazon Web Services (AWS) is a global cloud platform offering on-demand compute, storage, databases, AI/ML, and more. It operates across multiple geographic Regions and Availability Zones for reliability and low latency. Key benefits: elastic scaling, pay-as-you-go pricing, managed services, and global reach.

2) Signing up for AWS for free (and optional billing setup)

1. Visit the AWS Free Tier page and create your account. New customers can begin on free or credit-backed plans.

2. Security immediately after sign-up:

- Enable Multi-Factor Authentication (MFA) on the root user.
- Avoid creating root access keys; use IAM users/roles instead.

Optional (recommended): Billing guardrails

- Create AWS Budgets (e.g., zero-spend or low monthly cap).
- Turn on Free Tier usage alerts to get notified before charges.

3) Setting a goal: Hosting a website on AWS

We will deploy a small Node.js web app with: EC2 for the application server, S3 for static image storage, and RDS (PostgreSQL) for the database. Choose the AWS Region closest to your users to reduce latency.

4) Exploring the AWS Management Console

Sign in to the AWS Management Console to access services and account dashboards. Use the Region selector (top-right) and review service health via the AWS Health Dashboard.

5) EC2 — Launching a virtual server

- Services ‣ EC2 ‣ Instances ‣ Launch instances.
- Name: e.g., my-website.
- AMI: Amazon Linux (default).
- Instance type: a small Free-Tier-eligible size if available.
- Key pair: If using EC2 Instance Connect (browser SSH), you can proceed without a key pair for this demo.
- Network/Security: you will later open inbound TCP 8080 (demo).

• Storage: 8 GiB (default).

• Launch. Then on the instance's Security tab, open the created Security Group ‣ Edit inbound rules ‣ Add rule: Custom TCP, Port 8080, Source 0.0.0.0/0 (demo only; restrict in production).

• Note the Public IPv4 address for testing.

6) S3 — Configuring file storage

Goal: Create a public bucket for static images (demo only; do not expose sensitive data).

• Services ‣ S3 ‣ Create bucket.

• Bucket name must be DNS-compatible (e.g., my-macaroons-images).

• For public asset hosting, uncheck Block all public access (acknowledge the warning).

• Create the bucket.

• Add a bucket policy to allow public READ of objects (replace YOUR_BUCKET_NAME).

Example bucket policy:

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

Security note: Public buckets are fine for static images in demos. For private content, keep public access blocked and prefer presigned URLs or CloudFront with Origin Access Control.

7) RDS — Creating a database instance (PostgreSQL)

• Services ‣ RDS ‣ Create database.

• Use Easy create. Engine: PostgreSQL. Choose a small size eligible for free usage if available.

• Set DB identifier (e.g., my-website-db).

• Under connectivity, associate your EC2 instance so security groups are configured automatically.

• After creation, copy the DB endpoint, port, and credentials (username/password).

Timothy Forte
IV- ACSAD

8) Connecting to an EC2 instance and running commands

Connect with EC2 Instance Connect (browser-based SSH): EC2 ‣ Instances ‣ Connect ‣ EC2 Instance Connect ‣ Connect.

On the instance (example commands):

```
# Update and install Git/Node.js (Amazon Linux)
sudo dnf -y update
sudo dnf -y install git nodejs npm

# Retrieve your app code
git clone https://github.com/your-org/your-repo.git
cd your-repo

# Configure environment (replace placeholders)
export S3_BUCKET="YOUR_BUCKET_NAME"
export AWS_REGION="YOUR_REGION"
export AWS_ACCESS_KEY_ID="YOUR_IAM_ACCESS_KEY_ID"
export AWS_SECRET_ACCESS_KEY="YOUR_IAM_SECRET_ACCESS_KEY"
export DB_HOST="YOUR_RDS_ENDPOINT"   # e.g., mydb.xxxxxx.us-east-
1.rds.amazonaws.com
export DB_PORT="5432"
export DB_USER="YOUR_DB_USERNAME"
export DB_PASSWORD="YOUR_DB_PASSWORD"

# Install dependencies & start (background)
npm install
npm run start & disown
```

Test in your browser: http://YOUR-EC2-PUBLIC-IP:8080

Best practice: Use IAM users/roles with least privilege; avoid root access keys. Store and rotate secrets securely.

9) Tips for cost savings on AWS

• Create Budgets and turn on Free Tier usage alerts to receive notifications before charges.

• Stop instances when idle; terminate when finished to avoid ongoing costs.

• Use Savings Plans or Reserved Instances for steady workloads.

• Use EC2 Spot Instances for flexible, fault-tolerant tasks (tests, batch jobs).

10) Continuing your AWS journey with AI/ML and Certifications

Explore Amazon SageMaker for building, training, and deploying ML models. Browse AI services for vision, language, and recommendations. For credentials, start with AWS

Timothy Forte
IV- ACSAD

Certified Cloud Practitioner, then consider Solutions Architect – Associate or Developer – Associate.

Appendix A — Minimal IAM policy (S3 uploads only)

Scope the policy to only the bucket your app needs (replace names/ARNs). Attach to an IAM user/role used by your app; avoid root keys.

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

Appendix B — Quick checklist

• MFA on root; no root access keys.

• Budgets + Free Tier alerts configured.

• Security group only opens required ports (demo used 8080).

• Public S3 access only for truly public assets; keep private content blocked.

• Stop/terminate resources when finished.