ROD ANTHONY C. BALAORO 09/11/25

IV-ACSAD ELEC 3

How-To Guide: Getting Started with AWS and Hosting a Website

1. What AWS is and Its Key Benefits

Amazon Web Services (AWS) is a comprehensive and widely used cloud computing platform. It provides on-demand access to a variety of services, including computing power, storage, and databases. The key benefits of using AWS are its high performance, reliability, security, global reach, and a "pay-as-you-go" pricing model, where you only pay for the resources you use. This guide will walk you through the essential steps to get started with AWS by deploying a simple website.

Amazon Web Services (AWS) is the world's most popular cloud platform. It allows individuals and businesses to:

- Host websites and applications.
- Scale servers up or down as needed.
- Pay only for what they use.
- Automatically benefit from **performance, reliability, security, and global reach**.
- Access a vast catalog of services, including **AI, Machine Learning, and Analytics**.

2. Signing Up for AWS for Free

1. Go to the AWS website.
2. Click **Create a Free Account**.
3. Enter your email address, account name, and verify your email.
4. Set up a strong password, then continue.
5. Complete the registration form with your personal information and agree to the terms.
6. Provide your credit card details (no charges if you stay within the Free Tier).
7. Enter and verify your mobile number.
8. Choose the **Free Basic Support** option.
9. Complete the signup to access the **AWS Management Console**.

3. Setting a Goal: Hosting a Website on AWS

In the tutorial, the sample project is a bakery website. The objective is to successfully host this website using AWS's services:

Example project: *Mike's Macaroon Market* – an online bakery store.
To host the website, AWS will provide:

- **Compute resources (EC2)** to provide server power for the web application.
- **Storage (S3)** to store files like images and media.
- **Database (RDS)** to manage structured data such as inventory and customer orders.

This setup teaches beginners how to connect different AWS services into a functioning website.

4. Exploring the AWS Management Console

- The console is the central dashboard for all AWS services.
- Check service usage and costs through billing dashboards.
- Receive updates and alerts in the **AWS Health** section.
- Select a region (data center location) where your resources will run—important for reducing costs and improving performance for your audience.
- Access **AWS Health**, documentation, and support.

Overall, the console provides an organized, user-friendly interface to manage cloud resources.

5. EC2 – Launching a Virtual Server

1. From the Console, search for **EC2** and click **Launch Instance**.
2. Name it (e.g., *Macaroon Market Website*).
3. Choose **Amazon Linux (Free Tier eligible)**.
4. Select default instance type (t2.micro).
5. Skip key pair (for this tutorial).
6. Allow HTTP traffic on **port 8080**.
7. Use default storage (8 GB).
8. Click **Launch Instance**.
9. Modify security group → Add inbound rule: **HTTP, Port 8080, 0.0.0.0/0**.
10. Save and copy the **public IP** of your instance (used later to test the site).

6. S3 – Configuring File Storage

1. Search for **S3** and click **Create Bucket**.
2. Name your bucket (e.g., *mikes-macaroon-market*).
3. Uncheck *Block all public access* → Acknowledge.
4. Add tags (e.g., Website: Macaroon, Environment: Production).
5. Create the bucket.

6. Open the bucket → Permissions → Bucket policy → paste a JSON policy granting public read access. (Make sure the bucket name in the policy matches yours.)
7. Upload files manually (optional—the web app can upload automatically).
8. Generate **access keys**: Account → Security Credentials → Access Keys → Create. Save the Access Key ID and Secret Key (download CSV or copy immediately).
9. Use these keys in your application environment variables so it can interact with the bucket.

7. RDS – Creating a Database Instance

1. From the Services menu → Database Go to **RDS → Create Database**.
2. Choose **Easy Create**.
3. Select **PostgreSQL** as the engine.
4. Choose the smallest free-tier eligible database size.
5. Enter a DB name (e.g., *macaroon-db*).
6. Either set your own password or let AWS generate one (save it securely).
7. Expand the **EC2 connection settings** and link the database to your EC2 instance.
8. Click **Create database**.
9. Once provisioned, copy the **endpoint address** from the Connectivity & Security section.
10. Save the DB endpoint, username, and password—you'll need them to configure the website.

8. Connecting to an EC2 Instance and Running Commands

1. In **EC2 Console**, select your instance → Click **Connect** → Use **EC2 Instance Connect**.
2. A browser-based terminal opens (or use SSH if you downloaded a key).
3. Run setup commands:
    - Install Git: sudo dnf install git -y
    - Clone website code: git clone <repository-url>
    - Navigate into project folder: cd <project-folder>
    - Export environment variables for:
        - export S3_BUCKET="your-bucket-name"
        - export AWS_REGION="us-east-1"
        - export AWS_ACCESS_KEY_ID="your-access-key-id"
        - export AWS_SECRET_ACCESS_KEY="your-secret-key"
        - export DB_HOST="your-db-endpoint"
        - export DB_PASSWORD="your-db-password"
    - Install Node.js: sudo dnf install nodejs -y
    - Install dependencies: npm install
    - Run app in background: npm start & disown
4. Copy **public IP address** from console → Open in browser with http://<IP>:8080.

   (Replace placeholders with your real values.)

9. Tips for Cost Savings on AWS

- Set **spending limits** to avoid surprise charges.
- Set budgets and billing alerts to avoid surprise charges.
- Deploy in cost-efficient regions (e.g., US East – North Virginia).
- Use free-tier eligible services (smallest instance sizes) for learning and testing.
- Stop or terminate unused instances/resources when you're done.
- Use **Reserved Instances** or **Savings Plans** for predictable workloads.
- Take advantage of **Spot Instances** for non-critical workloads at lower costs.
- Tag resources for better tracking and cost monitoring.
- Regularly check **AWS Cost Explorer** for optimization suggestions.

---

10. Continuing Your AWS Journey

After building a website with core AWS services, you or the user can expand your knowledge by trying out advanced services like Artificial Intelligence (AI) and Machine Learning (ML). AWS offers tools for natural language processing, image recognition, fraud detection, and predictive analytics. Additionally, AWS certifications are highly valued in the IT industry and can validate your expertise in cloud computing, boosting your career opportunities in the global job market.

- Explore advanced services like **AI, Machine Learning, Data Analytics, and Computer Vision**.
- Consider earning **AWS Certifications** to boost career opportunities.
- Practice hands-on projects to build real-world skills.