AWS Essential Components

**1. Amazon S3 Buckets**A screenshot of a computer

AI-generated content may be incorrect.

* Amazon S3 (Simple Storage Service) is an object storage service designed for scalability, high durability, and availability. An S3 bucket is essentially a container that holds objects (files) such as images, videos, HTML, CSS, JavaScript, backups, logs, and more.
* **Common Use Cases:**
  + **Static Website Hosting:** By enabling the static website hosting feature, you can serve HTML, CSS, JavaScript, and other static content directly from an S3 bucket.
  + **Media and File Storage:** Storing images, videos, documents, and backups.
  + **Data Lake Storage:** Serving as a repository for large volumes of unstructured data.
  + **Log Storage:** Collecting logs from various applications or AWS services.
  + **Disaster Recovery & Archival:** Storing backup data with configurable lifecycle policies.

**Required Permissions**

When using S3 buckets, you’ll need to manage permissions carefully to ensure both functionality and security. This is typically done through IAM policies and bucket policies.

* **Common IAM Actions:**
  + **s3:ListBucket:** To list the objects within a bucket.
  + **s3:GetObject:** To retrieve an object. This is required for users or services (like CloudFront or Amplify) to read files, especially in a public website scenario.
  + **s3:PutObject:** To add new objects (used when uploading files).
  + **s3:DeleteObject:** To remove objects.
* **For Static Website Hosting:**  
  If you’re hosting a website, you’ll often configure the bucket policy to allow public read access to objects:

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "PublicReadGetObject",

"Effect": "Allow",

"Principal": "\*",

"Action": "s3:GetObject",

"Resource": "arn:aws:s3:::your-bucket-name/\*"

}

]

}

**Note:** Public access should be managed carefully to avoid unintended exposure of sensitive data. Alternatively, you can use CloudFront in front of S3 to handle public access securely.

**2. AWS Amplify**

A screenshot of a computer

AI-generated content may be incorrect.

* AWS Amplify is a development and hosting platform designed to help developers build, deploy, and manage full-stack web and mobile applications quickly. It provides a suite of tools and services for front-end frameworks and static site generators along with integrated CI/CD pipelines.
* **Common Use Cases:**
  + **Web & Mobile App Development:** Quickly scaffold and deploy applications with managed backend services (e.g., authentication, APIs, databases).
  + **Static Website Hosting:** Leverage Amplify Hosting to automatically build, deploy, and host static web applications.
  + **CI/CD Integration:** Connect to code repositories (GitHub, GitLab, Bitbucket, AWS CodeCommit) for automated builds and deployments.
  + **Full-Stack Deployments:** Integrate backend services like AWS AppSync (GraphQL), AWS Lambda, or Amazon Cognito seamlessly into your app.

**Required Permissions**

AWS Amplify itself uses service roles that need specific permissions to interact with other AWS services:

* **For Amplify Console & Build Process:**
  + **S3 Permissions:**
    - **s3:PutObject & s3:DeleteObject:** Used during deployments to update the hosted content.
    - **s3:GetObject:** For reading existing assets.
  + **CloudFront Permissions:**
    - **cloudfront:CreateDistribution, cloudfront:UpdateDistribution, cloudfront:DeleteDistribution:** When Amplify sets up or updates the CDN distribution.
  + **CodeBuild/CodePipeline Permissions:** When using integrated CI/CD pipelines, Amplify requires permissions to run builds and deploy artifacts.
* **Service Role Configuration:**  
  Amplify typically creates or uses a service role with a managed policy that covers these permissions. You can review or customize these policies in the IAM console as needed.

**3. Hosting Web Pages on AWS Using S3 and Amplify**

**Using S3 for Static Website Hosting**

* **Setup:**
  1. **Create an S3 Bucket:** Create a bucket (often named after your domain) and upload your static website files.
  2. **Enable Static Website Hosting:** In the bucket properties, enable static website hosting and specify the index document (e.g., index.html) and error document.
  3. **Configure Bucket Policy:** Apply a policy (as shown above) to allow public read access to the objects.
  4. **Custom Domain (Optional):** Use Route 53 to point your custom domain to the S3 website endpoint (or use CloudFront for HTTPS and improved performance).

**Using AWS Amplify for Web Hosting**

* **Setup:**
  1. **Connect Your Repository:** Link your Git-based repository (GitHub, GitLab, Bitbucket, or CodeCommit) to the Amplify Console.
  2. **Configure Build Settings:** Amplify detects your project type (e.g., React, Angular, Vue, etc.) and sets up build commands (this is customizable via a amplify.yml file).
  3. **Deployment:** On every code commit, Amplify automatically builds and deploys your application. During this process, static assets are placed in an S3 bucket managed by Amplify, and a CloudFront distribution is set up to deliver content globally.
  4. **Custom Domain & SSL:** Amplify Console also makes it straightforward to configure custom domains and automatically provision SSL certificates using AWS Certificate Manager.

**Combined Workflow**

* **Local Development to Production:**
  + **Development:** You work on your website/application locally.
  + **Source Control:** Commit and push changes to your repository.
  + **CI/CD:** Amplify picks up the changes, runs the build process, and deploys the updated static assets to an S3 bucket.
  + **Distribution:** CloudFront (configured by Amplify) serves the content, caching files at edge locations for low latency.
  + **Permissions Management:** IAM roles and policies ensure that Amplify and related services (S3, CloudFront) can operate securely and only perform the actions they need.
  + By carefully configuring IAM policies and service roles, you ensure secure and efficient operation of your AWS-hosted web applications.