AWS Fundamentals & Core Skills

- **Day 1: AWS Fundamentals & Core Skills**
  - Introduction to AWS Core Services, IAM basics, Security/Compliance.
  - Services overview: S3, Lambda, CloudWatch, CloudTrail, Glue, AWS PostgreSQL RDS
  - Setup IAM (Identity and Access Management) & billing alerts- done in AWS management console
  - **Hands-on**:
    - Create S3 bucket (Upload Different File types, Bucket Versioning)- completed using AWS console
    - GitHub basics (repo setup, PRs)-completed
    - Create IAM users, groups, and roles- completed using AWS console

1. Storage & Database (Where data lives)

- **S3 (Simple Storage Service): * What it is:** An object storage service that acts like a "hard drive in the cloud."
- **How it works:** You create "Buckets" and upload "Objects" (files like images, backups, or logs). It is designed for 99.999999999% (11 nines) durability, meaning it's almost impossible to lose your data.

- **AWS PostgreSQL RDS (Relational Database Service):**
- **What it is:** A managed service for running PostgreSQL databases.
- **How it works:** AWS handles the "heavy lifting" like patching, backups, and scaling. You get a production-ready database without having to manage the underlying server or hardware.

2. Compute & Automation (How data moves)

- **Lambda:**
- **What it is: Lambda:** "Serverless" code. You upload your code (Python, Node.js, etc.), and it runs only when triggered by an event (like a file being uploaded to S3). You don't manage any servers.

- **Best For:** Automating tasks, processing data in real-time, or building small backend APIs.

3. Monitoring & Security (The Watchmen)
   - These services keep an eye on your account to make sure everything is working and safe.
   - **CloudWatch:** * **What it is:** A performance monitor. It tracks **metrics** (CPU usage, disk space) and **logs** (error messages from your code).
   - **Analogy:** The **Dashboard** of your car. It tells you how fast you're going and if the engine is overheating.
   - **CloudTrail:** * **What it is:** An audit log. It records **every single action** taken in your account—who logged in, who deleted a bucket, who changed a password.
   - **Analogy:** The **Security Camera** in a bank. It doesn't tell you how the bank is performing; it tells you who walked through the door and what they touched.

4. Data Integration (How data is prepared)
   - **Glue:**
   - **What it is:** A fully managed **ETL** (Extract, Transform, Load) service.
   - **How it works:** It "glues" different data sources together. It can "crawl" your RDS database or S3 files to discover their structure (schema), clean the data, and move it to a data warehouse for analysis.

How they all work together: A Scenario

- Imagine you are building a photo-sharing app:
- **User** uploads a photo to **S3**.
- **CloudTrail** logs that "User Priya uploaded a file."
- The upload triggers a **Lambda** function to resize the photo.
- **Lambda** saves the photo's name and size into your **PostgreSQL RDS** database.

- **CloudWatch** monitors how many photos are being uploaded and sends you an email if the system crashes.
- Once a month, **Glue** takes all that data from RDS and S3 to create a report for your business team.
- In AWS, IAM (Identity and Access Management) is the security framework that handles **Authentication** (who you are) and **Authorization** (what you can do).

To understand these four components, it is helpful to use a simple analogy of an **Office Building**:

1. IAM User (The Employee)

A **User** is an individual person or an application that interacts with AWS.

- **Identity:** Each user has a unique set of credentials (username/password for the console or Access Keys for the CLI).
- **Analogy:** This is like an individual employee with their own name badge.
- **Best Practice:** Create one user per real person. Never share credentials.

2. IAM Group (The Department)

A **Group** is a collection of IAM users. It is a way to manage permissions for many people at once.

- **Function:** You don't give permissions to each person individually; you give them to the group, and everyone in that group "inherits" those rules.
- **Analogy:** This is like the "Accounting Department." Every employee in this department automatically gets access to the billing spreadsheets.
- **Benefit:** If a new person joins the team, you just add them to the group instead of manually setting up 20 different permissions.

3. IAM Role (The "Temporary Hat")

A **Role** is an identity that does not belong to a specific person. Instead, it is meant to be **assumed** by anyone or anything that needs it temporarily.

- **Key Difference:** Roles do not have permanent passwords. When a user or service "assumes" a role, AWS provides **temporary** security credentials that expire.
- **Analogy:** This is like a "Visitor Badge" or a "Hard Hat." If an office worker needs to enter the server room for 10 minutes, they put on the "Technician Hat" to get in. Once they leave, they take the hat off and lose those special powers.
- **Use Case:** Allowing an EC2 server to talk to an S3 bucket without storing dangerous "permanent keys" on the server.

4. IAM Policy (The Rulebook)

A **Policy** is a JSON document that defines the actual permissions. It is the "brain" behind the other three.

- **Contents:** It lists exactly what **Actions** are allowed (like s3:GetObject), on which **Resources** (like a specific bucket), and under what **Conditions** (like "only from this IP address").
- **How it works:** You attach a Policy to a User, a Group, or a Role. Without a policy, an identity has **zero** power by default.