**What do you know about Microsoft Azure and AWS? Have you ever worked with cloud technologies? If so, please describe.**

I have theoretical knowledge of cloud platforms like **Microsoft Azure** and **AWS**. Both are widely used for cloud computing services, offering solutions for hosting, storage, networking, AI, and application deployment.

* **Microsoft Azure** provides services like virtual machines, app services, and storage accounts, with a strong integration for Microsoft tools like Office and .NET.
* **AWS (Amazon Web Services)** is known for its scalability and extensive services like EC2 for computing, S3 for storage, and RDS for databases.

While I haven’t worked directly on these platforms, I am eager to learn and apply my skills in cloud technologies. My experience in development and web scraping provides a solid foundation to quickly adapt to cloud-based workflows.

**What is FIFO in AWS queue? What is the purpose of Queue?**

**FIFO in AWS Queue:**

**FIFO (First-In-First-Out)** is a type of Amazon Simple Queue Service (SQS) queue that ensures:

1. **Order:** Messages are processed in the exact order they are sent (First in, First Out).
2. **Duplication Prevention:** Ensures that each message is delivered and processed exactly once, preventing duplication.

**Purpose of a Queue in AWS:**  
An **AWS SQS Queue** serves as a message buffer to decouple components in a distributed application. The primary purposes include:

1. **Asynchronous Communication:** Allows different components to communicate without waiting for each other.
2. **Scalability:** Handles varying workloads by buffering messages for later processing.
3. **Reliability:** Ensures message delivery even if one part of the system is temporarily unavailable.
4. **Order Maintenance (FIFO):** For applications where message processing order is critical.

AWS SQS queues are widely used for building fault-tolerant, scalable systems in event-driven architectures.

**Have you ever setup the CI/CD pipeline? Do you know how to deploy the Lambda functions/Create them?**

I have theoretical knowledge of **CI/CD pipelines** and deploying **AWS Lambda functions**, but I haven’t set them up practically yet. Here’s an outline of what I know:

**CI/CD Pipeline Setup:**

* **Continuous Integration (CI):** Automates code integration, testing, and builds after every commit to a shared repository. Tools like GitHub Actions, Jenkins, or AWS Code Pipeline can be used.
* **Continuous Deployment (CD):** Automates the deployment of code changes to production.
* For **AWS CI/CD**, you can use **Code Pipeline** with **Code Build** for building and **Code Deploy** for deployments. This ensures smooth and automated delivery pipelines.

**Creating and Deploying AWS Lambda Functions:**

1. **Creating a Lambda Function:**

* Use the AWS Management Console, AWS CLI, or Infrastructure as Code (IaC) tools like **AWS Cloud Formation** or **Terraform**.
* Select runtime (e.g., Python, Node.js, etc.), define the handler, and upload the function code.
* Set permissions using IAM roles and policies

1. **Deploying Lambda Functions:**

* **Manually:** Upload code as a zip file or directly through the console.
* **Using CI/CD:** Automate Lambda deployments via tools like **Server less Framework**, **AWS SAM (Server less Application Model)**, or third-party CI/CD pipelines.
* **Versioning and Aliases:** Use Lambda’s versioning feature to manage different deployments and aliases for traffic routing.

I am eager to gain hands-on experience with these tasks and can quickly adapt to practical scenarios with guidance or requirements.