# 1-3-1: Achieving Cloud Observability Using Tracing

After completing this episode, you should be able to:

- Identify and explain the importance of tracing to cloud observability, given a scenario.

**Description:** In this episode, the learner will examine tracing, tracing components, such as traces, spans, annotations, and exporters. We will explore the benefits of leveraging to achieve cloud observability.

- Describe tracing
    - The practice of monitoring and recording information about software executions, specifically within distributed systems.
- Describe the purpose of tracing
    - Helps in understanding the journey and behavior of requests as they flow through various services and components of cloud-based applications and infrastructure.
    - Targets the latency, failures, and overall performance of individual requests or transactions.
- Describe the benefits of tracing to cloud observability
    - Performance optimization
        - ♠ Help identify bottlenecks in the system by providing a detailed breakdown of where time is spent in a distributed transaction.
    - Error diagnosis
        - ♠ It enables developers to pinpoint the exact service or operation where a failure occurs
        - ♠ Facilitates quicker root cause analysis
    - Service dependency analysis
        - ♠ Tracing visually represents interactions between services, helping understand dependencies and the impact of one service on another.
    - Capacity planning
        - ♠ Analyzing trace data can help organizations better understand usage patterns and plan capacity accordingly to ensure performance and scalability. (demo resource scaling options)
    - Improved user experience
        - ♠ Tracing can contribute to optimizing application performance and reliability by reducing downtime and slow response times.
- Describe a real-world example of performing tracing in a cloud environment
    - Using the Performance Diagnostic feature in Azure on a cloud resource, such as a server, application, or service.

## Additional References

- Traces - represents the complete lifecycle of a single user request or transaction as it traverses through the different services in a system.
- Spans - represents a specific unit of work or operation within a service, with each trace comprised of multiple spans.
- Annotations - metadata added to spans to provide additional details about the execution, which can include:
    - Timestamps
    - Events
    - Other key-value information
- Trace exporters - components that send trace data to a backend system or observability platform for storage and analysis.