SERVERLESS ARCHITECTURE:

➞Questions/Answers

Q1: Does "Serverless" mean there are absolutely no servers involved?

A: No, that is a common myth. It is actually a cloud computing execution model where servers still exist, but the cloud provider (like AWS or Google Cloud) dynamically manages their allocation and provisioning for you.

Q2: How does the "Event-Based" nature of serverless work?

A: In this model, code does not run continuously. It only runs when it is triggered by a specific event, such as a user click, an API call, or a file upload.

Q3: What does it mean when we say serverless functions are "Ephemeral"?

A: It means the resources are temporary. They spin up instantly to perform a specific task and then vanish immediately after the task is done.

Q4: How does serverless architecture handle "idle costs"?

A: It eliminates them completely. You do not pay for "ghost servers" (servers running but doing nothing); if your code isn't running, your bill is $0.

Q5: What is the "Stateless" property of a serverless function?

A: Serverless functions do not "remember" past interactions or store session data between executions. They simply execute their task and exit.

Q6: How does the "Water Tap" metaphor explain serverless utility?

A: Traditional hosting is like digging a well (you maintain the pipes), whereas serverless is like a tap: resources flow only on-demand when you turn the handle, and the utility company handles the infrastructure.

Q7: How does billing differ between Traditional and Serverless models?

A: In a traditional model, you pay by the hour, even if the server is idle. In serverless, you pay by the millisecond, charging you only for the exact time your code executes.

Q8: Who is responsible for OS updates and security patching in serverless?

A: The cloud provider manages the operating system, updates, and security patches entirely, allowing you to focus on code.

Q9: How does serverless architecture handle sudden spikes in traffic?

A: It offers infinite scaling. The system can scale from 1 user to 1 million users automatically and instantly without crashing.

Q10: What is the main strategic advantage for developers using serverless?

A: Speed and faster market entry. Developers can skip the time required for server setup and start shipping features immediately.

Q11: What are the typical components in a serverless workflow?

A: A standard workflow includes Web or Mobile Clients connecting to an API Gateway, which then triggers specific Functions (e.g., Function 1, Function 2) that interact with Databases or Authentication Services.

Q12: Why is serverless described as a "Shift in Mindset"?

A: It shifts the focus from worrying about hardware to focusing purely on code. You stop managing the OS and updates manually and let the provider handle everything.