Serverless Architecture

The End of Infrastructure Management?

The Big Myth: "Serverless" doesn't mean no servers.

The Reality: It is a cloud computing execution model where the cloud provider (such as AWS, Google Cloud, or Azure) dynamically manages the allocation and provisioning of servers.

The Shift: Stop worrying about hardware. Start focusing on code.

The Goal: Build applications faster, cheaper, and smarter.

How It Actually Works

Event-Driven Computing (FaaS)

Event-Based

Code only runs when triggered (e.g., a user click, an API call, a file upload).

Ephemeral

Resources spin up instantly to do the task, then vanish.

Stateless

Functions do not "remember" past interactions; they just execute and exit.

Provider Managed

AWS, Google, or Azure handle all the patching and security.

The Water Tap Metaphor

Utility vs. Ownership

Traditional (The Well):

You dig the well, install the pump, and maintain the pipes. Hard to scale.

Serverless (The Tap):

You just turn the handle.

- **On-Demand:** Water flows only when you need it.
- **Pay-Per-Drop:** You only pay for what you use.
- **Zero Maintenance:** The utility company handles the infrastructure.

The Strategic Advantage

Cost, Speed, and Scale

Zero Idle Costs

Never pay for "ghost servers" at 3 AM. If code isn't running, the bill is $0.

Infinite Scaling

From 1 user to 1 million users in seconds. No crashing.

Faster Market Entry

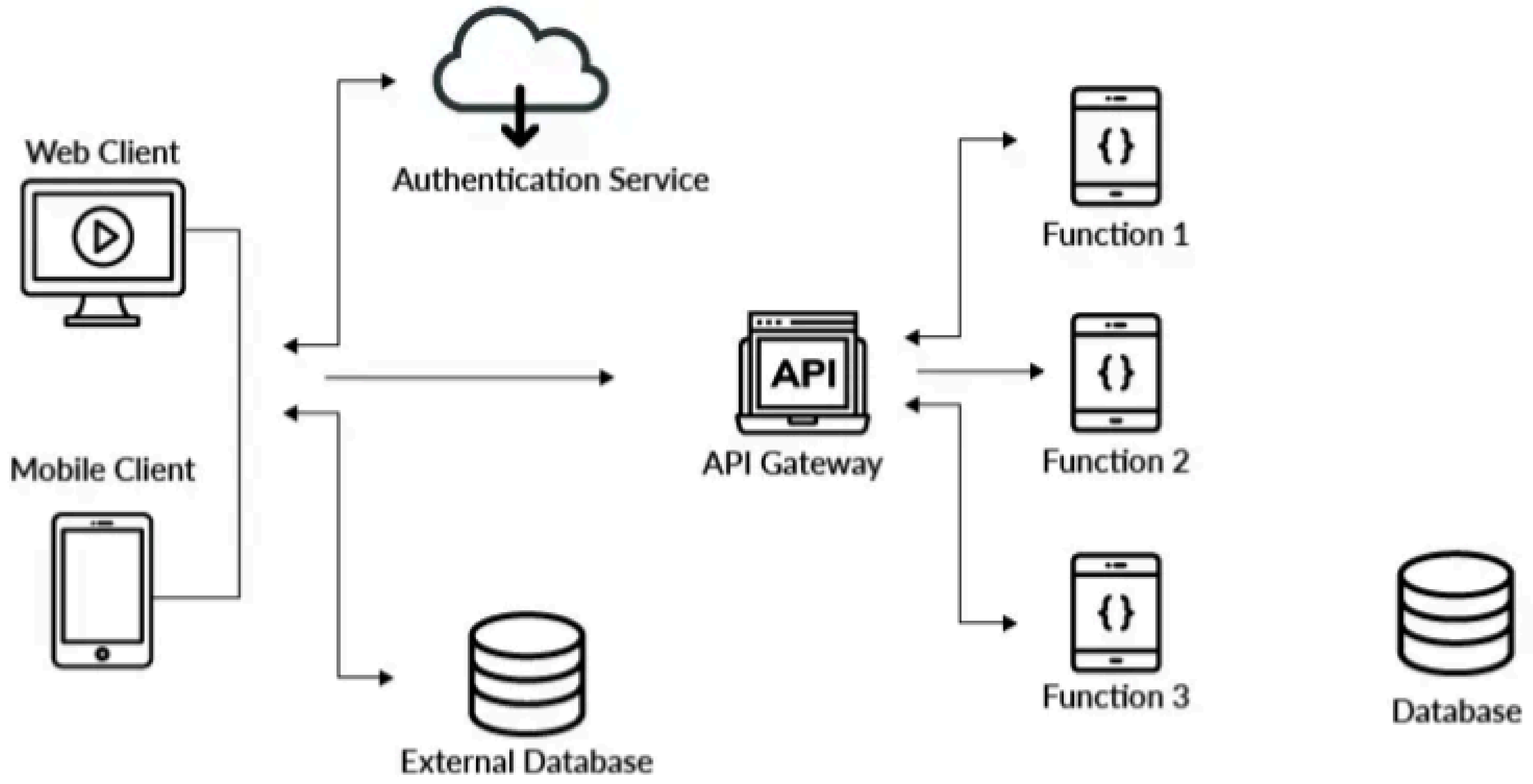
Developers skip setup time and ship features immediately.

Traditional vs. Serverless

A Shift in Mindset

Management:	*Traditional:* You manage the OS and updates.	*Serverless:* The Cloud Provider manages everything.
Scaling:	*Traditional:* Manual and slow.	*Serverless:* Automatic and instant.
Billing:	*Traditional:* Pay by the hour (even when idle).	*Serverless:* Pay by the millisecond (execution only).

Working Of Serverless Architecture


Web Client
Mobile Client
Authentication Service
External Database
API
API Gateway
Function 1
Function 2
Function 3
Database