**Statelessness in REST APIs**

As per the REST (REpresentational **“State”** Transfer) architecture, the server does not store any state about the client session on the server-side. This restriction is called **Statelessness**. Each request from the client to server must contain all of the information necessary to understand the request, and cannot take advantage of any stored context on the server. Session state is therefore kept entirely on the client. **client is responsible for storing and handling all application state-related information on client** side.

It also means that the client is responsible for sending any state information to the server whenever it is needed. There should not be any *session affinity* or *sticky sessions* on server.

*Statelessness means that every HTTP request happens in complete isolation. When the client makes an HTTP request, it includes all information necessary for the server to fulfill that request. The server never relies on information from previous requests. If that information was important, the client would have sent it again in this request.*

To enable clients to access these stateless APIs, its necessary that servers also should include every piece of information that the client may need to create the state on its side.

For becoming stateless, do not store even [authentication/authorization](https://restfulapi.net/security-essentials/) details of client. Provide credentials with the request. Each request MUST stand alone and should not be affected by the previous conversation happened from the same client in past.

**Application State vs Resource State**

Please do not confuse between application state and resource state. Both are completely different things.

**Application state** is server-side data which servers store to identify incoming client requests, their previous interaction details, and current context information.

**Resource state** is the current state of a resource on a server at any point of time – and it has nothing to do with the interaction between client and server. It is what you get as a response from the server as API response. You refer to it as resource representation.

**REST statelessness means being free on application state.**

**Advantages of Statelessness**

There are some very noticeable advantages for having **REST APIs stateless**.

1. Statelessness helps in scaling the APIs to millions of concurrent users by deploying it to multiple servers. Any server can handle any request because there is no session related dependency.
2. Being stateless makes REST APIs less complex – by removing all server-side state synchronization logic.
3. A stateless API is also easy to [cache](https://restfulapi.net/caching/) as well. Specific software can decide whether or not to cache the result of an HTTP request just by looking at that one request. There’s no nagging uncertainty that state from a previous request might affect the cacheability of this one. It improves the performance of applications.
4. The server never loses track of “where” each client is in the application because the client sends all necessary information with each request.

Reference: Roy T. Fielding on [Stateless](https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm#sec_5_1_3)