# What is Cookies

HTTP is a stateless protocol, which means that each request from a client to a server is treated as a new, independent request. The server doesn't inherently remember information about previous requests from the same client.

A servlet cookie is a small piece of data that a servlet sends to a web browser as part of an HTTP response. The browser then stores this information on the client's computer. When the browser makes a subsequent request to the same web server, it sends the stored cookie data back to the server in the HTTP request header.

# How Cookies works

Server Creates a Cookie:

Your Servlet code creates a jakarta.servlet.http.Cookie object (or javax.servlet.http.Cookie for older versions of Java EE). You set a name and a value for this cookie.

Syntax: Cookie cookie = new Cookie("name", "value");

Server Adds Cookie to Response:

You add the created Cookie object to the HttpServletResponse object. The web container then includes this cookie in the HTTP response headers sent back to the client's browser.

Syntax: response.addCookie(cookie);

Client Stores Cookie:

The client's browser receives the response, parses the Set-Cookie header, and stores the cookie on the user's computer.

Client Sends Cookie with Subsequent Requests:

For all future requests to the same domain (and based on other cookie attributes like path), the browser automatically includes that cookie in the HTTP request headers.

Server Reads Cookie from Request:

Your Servlet code can then retrieve the cookies from the HttpServletRequest object, extract their names and values, and use that information.

# Key Attributes

When you create a Cookie object, you can set several important attributes:

1. name (Required): A unique name for the cookie (e.g., "username", "lastVisit").
2. value (Required): The data associated with the cookie (e.g., "Alice", "2025-07-11").
3. setMaxAge(int seconds):
   1. Positive Value: The cookie will persist on the client's machine for the specified number of seconds. This creates a persistent cookie. (e.g., cookie.setMaxAge(60 \* 60 \* 24 \* 30); for 30 days).
   2. 0: Deletes the cookie immediately from the client's browser.
   3. -1 (Default): The cookie is a session cookie. It will only last as long as the user's browser session (i.e., it's deleted when the browser is closed).
4. setPath(String path): Specifies the URL path for which the cookie is valid. If not set, it defaults to the path of the servlet that created it. A common value is "/" to make it available to the entire web application.
5. setDomain(String domain): Specifies the domain for which the cookie is valid. By default, it's the domain that sent the cookie. You can set it to a broader domain (e.g., "https://www.google.com/search?q=.example.com") to make it available to subdomains.
6. setSecure(boolean secure): If true, the cookie will only be sent over secure (HTTPS) connections.
7. setHttpOnly(boolean httpOnly):If true, the cookie cannot be accessed via client-side JavaScript (document.cookie). This is a crucial security measure to mitigate XSS (Cross-Site Scripting) attacks, as it prevents malicious scripts from stealing session cookies.