

DAY 9

Sessions or Tokens

Sessions are a way to maintain user state and authenticate users in web applications.

1. User Authentication

When you visit a website or use a web application, you must log in to access certain features or personalized content. Sessions or tokens are used to authenticate users, proving that they are who they claim to be. This ensures that only authorized users can access restricted areas or perform certain actions.

2. Maintaining User State

Web applications often need to remember user information as they navigate through different pages or interact with the app. Sessions or tokens help maintain this user state. For example, if you add items to your shopping cart on an e-commerce website, the website needs to remember those items as you browse other pages. Sessions or tokens store this information so that it can be accessed later.

3. Security

Sessions and tokens play a crucial role in securing web applications. They help prevent unauthorized access by verifying the identity of users before granting them access to sensitive data or functionality. Additionally, they can be used to protect against common security threats such as session hijacking or cross-site request forgery (CSRF) attacks.

4. Personalization and Customization

Sessions or tokens allow web applications to provide personalized experiences to users. By storing information about users' preferences, settings, or past interactions, the application can tailor the content or functionality to better meet their needs. This enhances user satisfaction and engagement with the application.

5. Scalability

Sessions and tokens are designed to scale with the size and complexity of web applications. They provide a flexible and efficient way to manage user authentication and state, even as the number of users or requests increases. This scalability is essential for ensuring that web applications can handle growing traffic and user demands without sacrificing performance.

In summary, sessions and tokens are essential components of web development, enabling user authentication, maintaining state, ensuring security, personalizing experiences, and supporting scalability. They form the foundation of secure and user-friendly web applications that can provide personalized and seamless experiences to users.

Sessions based Authentication

- A session is a way or a small file, most likely in JSON format, that stores information about the user, such as a unique ID, time of login expirations, and so on. It is generated and stored on the server so that the server can keep track of the user requests.

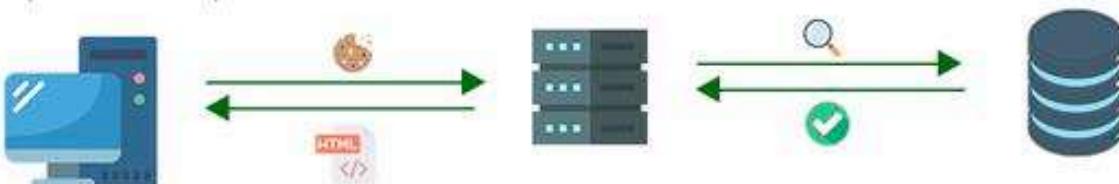
Working

1. The user sends a login request to the server.
2. The server authenticates the login request, sends a session to the database, and returns a cookie containing the session ID to the user.
3. Now, the user sends new requests (with a cookie).
4. The server checks in the database for the ID found in the cookie, if the ID is found it sends the requested pages to the user.

The user sends login request



The user sends new request
(with a cookie)



The server looks up in the database for the ID Found in the cookie,
if the ID is found it sends the requested pages to the user

Cookies

A cookie is a **small piece of data that a website stores on a user's computer or device**. Cookies are commonly used by websites to remember information about the user's browsing activity, preferences, and interactions with the site. Here's a beginner-friendly explanation of cookies

How Cookies Work

- **Creation:** When you visit a website for the first time, the website may send a small text file (the cookie) to your browser.
- **Storage:** Your browser stores the cookie on your computer or device. Each cookie is associated with a specific website and contains information such as a unique identifier and any data the website wants to remember about you.
- **Sending with Requests:** When you revisit the website or navigate to different pages on the same site, your browser automatically includes the cookie in the HTTP requests it sends to the website's server.
- **Server Processing:** The website's server receives the cookie along with the request. It can then read the cookie to retrieve the stored information about you.
- **Usage:** Websites use cookies for various purposes, such as remembering your login status, storing your preferences, tracking your activities for analytics purposes, and personalizing your browsing experience.

Types of Cookies

- **Session Cookies:** These cookies are temporary and are deleted when you close your browser. They are often used for maintaining your login state during a browsing session.
- **Persistent Cookies:** These cookies are stored on your device for a specified duration, even after you close your browser. They can be used for purposes such as remembering your preferences or login information across multiple visits to a website.

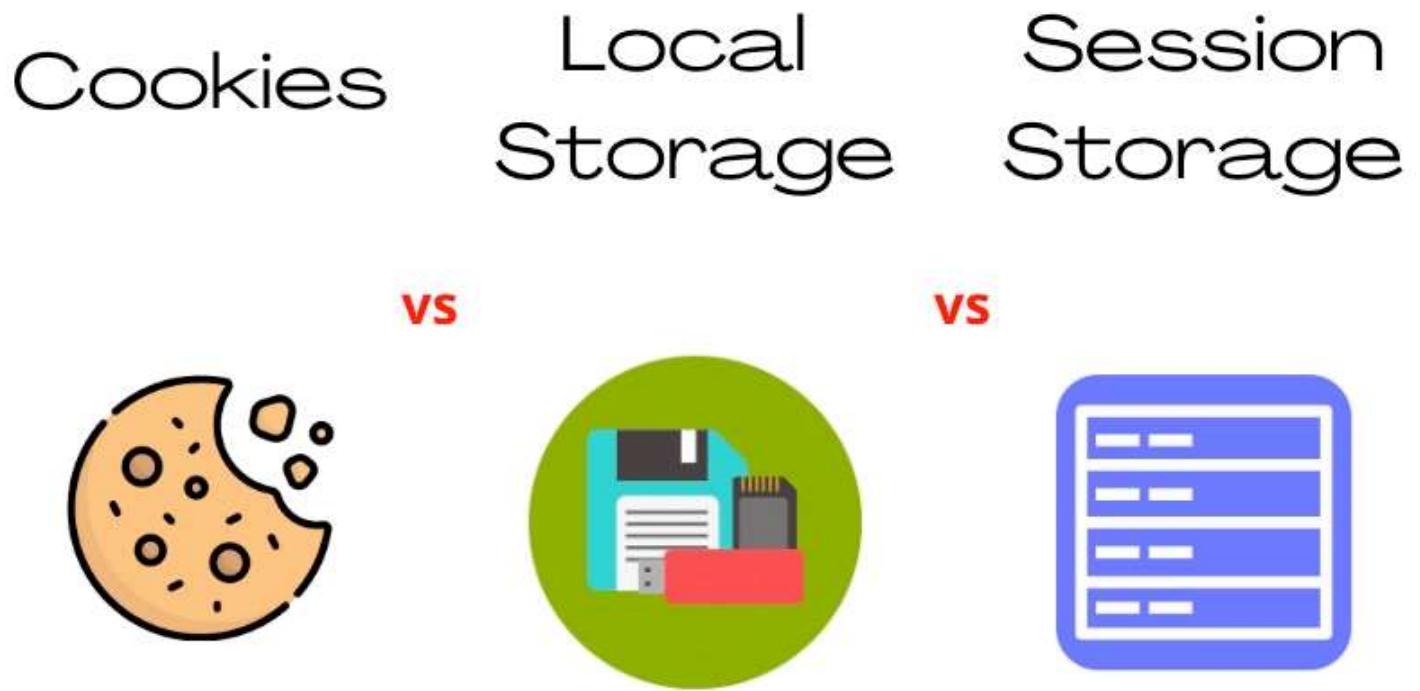
Privacy and Security

Cookies can raise privacy concerns because they can be used to track users' browsing behavior across different websites. However, cookies cannot execute code or transmit viruses, and they can only store information that the website itself provides.

In summary, **cookies are small text files** stored on your computer by websites you visit. They help websites remember information about you and enhance your browsing experience by personalizing content, maintaining login status, and tracking preferences. While cookies are an essential part of web functionality, they also raise privacy considerations, and users can manage their cookie settings in their web browsers.

Client Storage option

It's not mandatory that the cookies received from the server will be stored in the "Cookies" storage of your browser. The browser provides multiple storage options, each serving different purposes and offering various capabilities:



- 1. Cookies Storage:** Cookies received from the server are typically stored in the "Cookies" storage of your browser. These cookies can include **session cookies**, which are deleted when you close your browser, and persistent cookies, which are stored for a longer period.
- 2. Local Storage:** Local storage is a mechanism that **allows web applications to store data locally in the browser**. Unlike cookies, data stored in local storage is not automatically sent to the server with every request. This storage option is often used for caching data, storing user preferences, or implementing client-side features.

3. **Session Storage:** Session storage is similar to local storage but is scoped to a particular browsing session. Data stored in session storage is cleared when you close the browser or tab. It's commonly used for temporary data storage or for maintaining a state within a single browsing session.

Important Points

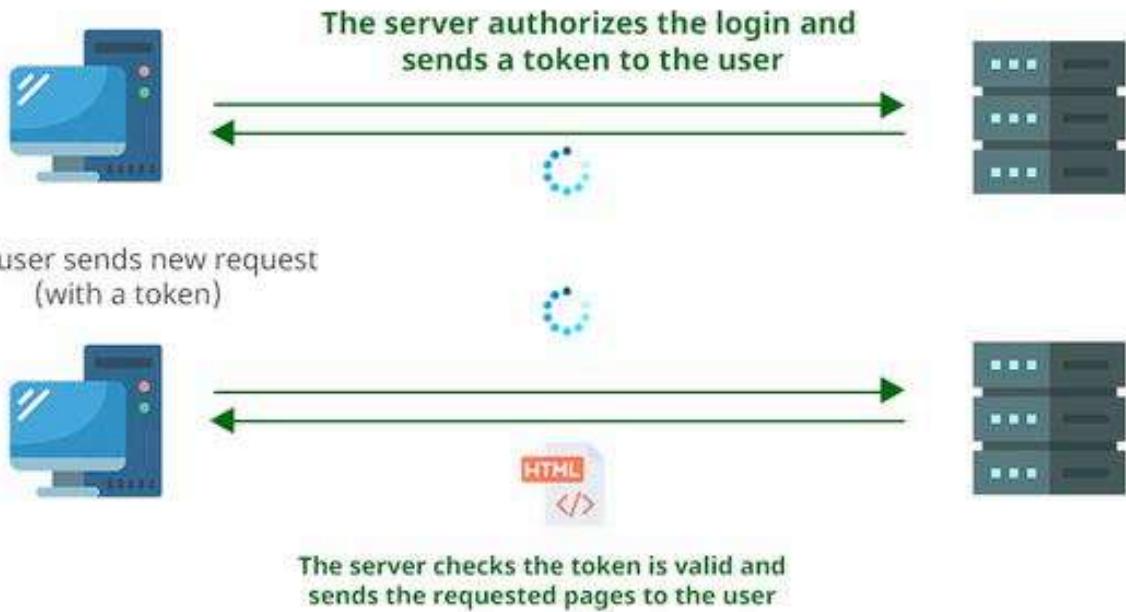
- While cookies are commonly used for storing session identifiers (session IDs) or authentication tokens received from the server, web applications can choose to store this information in local storage or session storage instead.
- The choice of storage mechanism depends on factors such as security requirements, application architecture, and use case. Cookies offer automatic inclusion in HTTP requests and can be more secure if configured properly, but local storage and session storage provide more control over data management on the client side.
- While cookies are a common way to store data received from the server, web applications have the flexibility to choose other storage options such as local storage or session storage based on their requirements.

Token based Authentication

- A token is an authorization file that cannot be tampered with. It is generated by the server using a secret key, sent to and stored by the user in their local storage. Like in the case of cookies, the user sends this token to the server with every new request, so that the server can verify its signature and authorize the requests.
- Working
 - The user sends a login request to the server.
 - The server authorizes the login and sends a token to the user.
 - Now, the user sends a new request(with a token).
 - The server checks whether the token is valid or not, if the token is valid it sends the requested pages to the user.

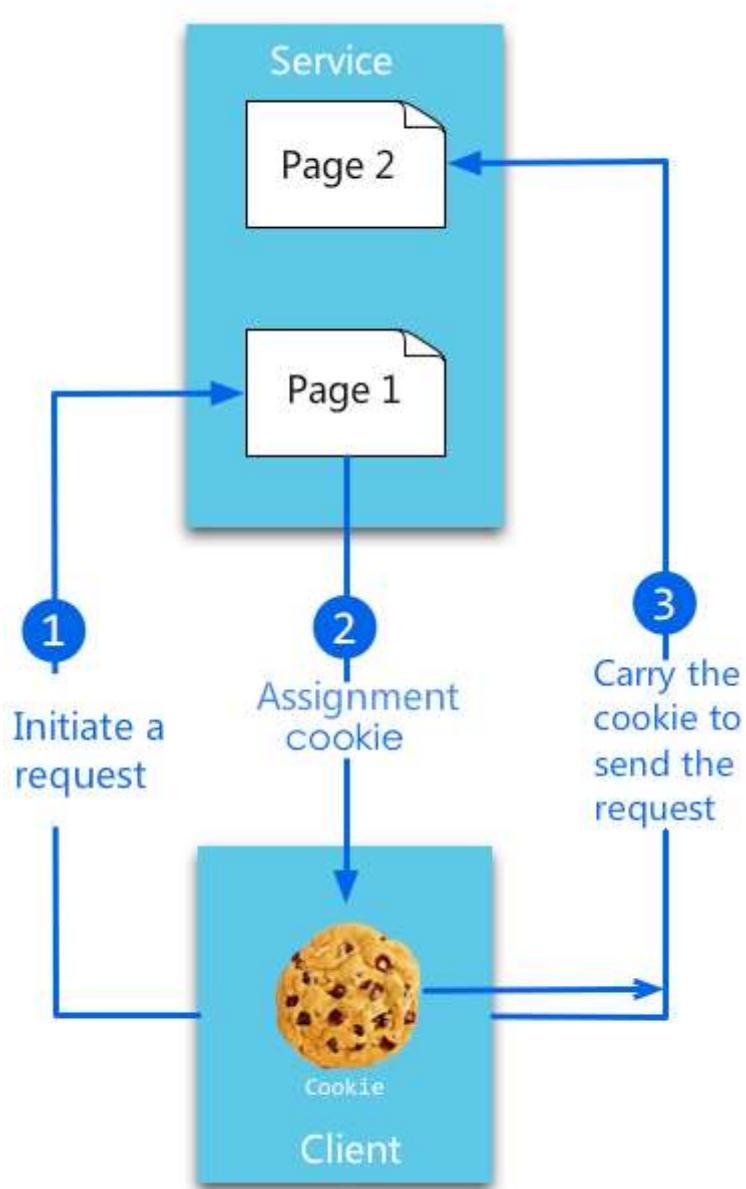
Note- Those are **not authentication files**, they are **authorization ones**. While receiving a token, the server does **not** look up who the user is, it simply authorizes the user's requests relying on the validity of the token.

The user sends login request



Difference between Session & Token

	Criteria	Session authentication method	Token-based authentication method
1	Which side of the connection stores the authentication details	Server	User
2	What the user sends to the server to have their requests authorized	A cookie	The token itself
3	What does the server do to authorize users' requests	Looking up in its databases to find the right session thanks to the ID the user sends with a cookie	Decrypting the user's token and verifying its signature
4	Can the server admins perform securities operations like logging users out, changing their details, etc	Yes, because the session is stored on the server	No, because the token is stored on the user's machine

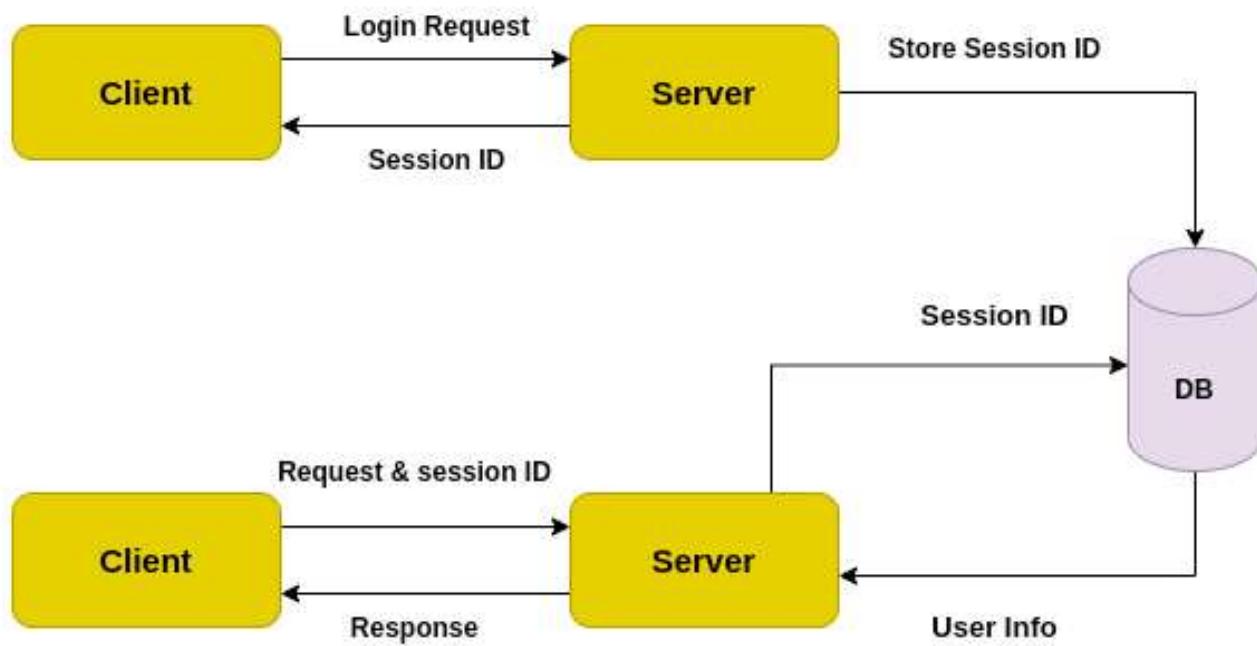


Screenshot of a browser developer tools Storage panel, specifically the Cookies section for the domain <https://www.geeksforgeeks.org>.

Name	Value	Domain	Path	Expires / Max-Age	Size	HttpOnly	Secure	SameSite	Last Accessed
authToken	f6881afeedaecc07da9b4cd142...	.geeksforgeek...	/	Session	41	true	true	None	Thu, 08 Feb 2024 09:53:4...

How Session works

Session ID



How Token works

