

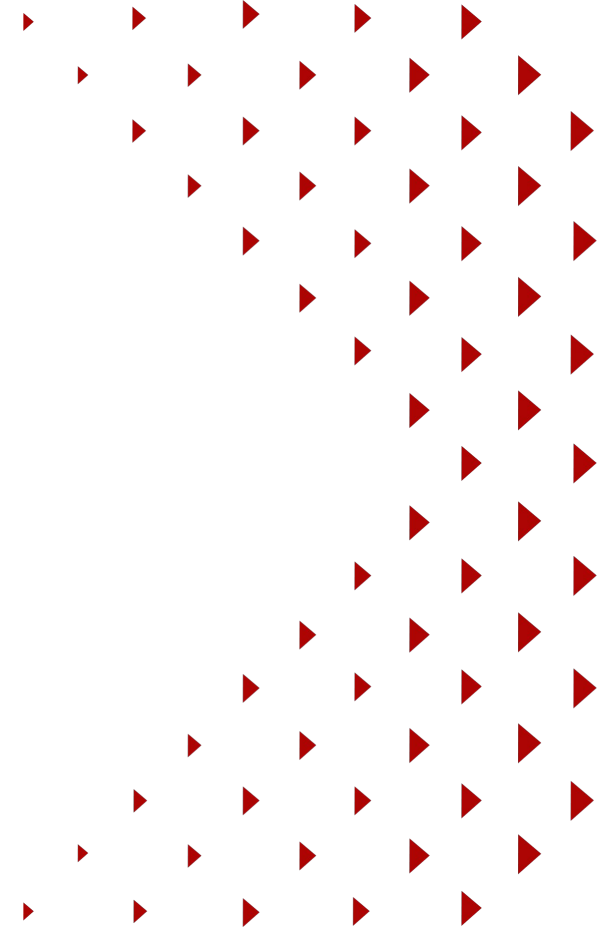


## **SESSION 15:**

# **STORING DATA IN THE BROWSER**

Module 1: Web Front-end Fundamental

Version: 2.0



1. **Web Storage API**
2. **Local Storage**
3. **Session Storage**
4. **Cookies**
5. **So sánh**

# 1. WEB STORAGE API - 1

- **Định nghĩa**

- Trong HTML5, Web Storage API cung cấp khả năng **lưu trữ dữ liệu kiểu key-value** trên trình duyệt trực quan hơn nhiều so với việc sử dụng cookie
- Web Storage API cung cấp hai Object để lưu trữ dữ liệu trên trình duyệt:
  - **localStorage**: lưu trữ dữ liệu không có ngày hết hạn
  - **sessionStorage**: lưu trữ dữ liệu cho một phiên (dữ liệu bị mất khi đóng trình duyệt)



## 2. LOCAL STORAGE - 1

- **Định nghĩa**

- **Local Storage** là một Object lưu trữ dữ liệu của Web Storage API, được phát triển từ phiên bản HTML5.
- Dữ liệu của localStorage sẽ tồn tại vô hạn ngay cả khi trình duyệt đóng, dữ liệu chỉ bị xóa hoặc thay đổi khi trình duyệt và JavaScript tác động vào
- Một số phương thức với localStorage:
  - localStorage.**getItem**(key): Lấy dữ liệu theo key
  - localStorage.**setItem**(key, value): thêm dữ liệu
  - localStorage.**removeItem**(key): xóa dữ liệu theo key
  - localStorage.**clear**(): Xóa toàn bộ dữ liệu trong localStorage

## 2. LOCAL STORAGE - 2

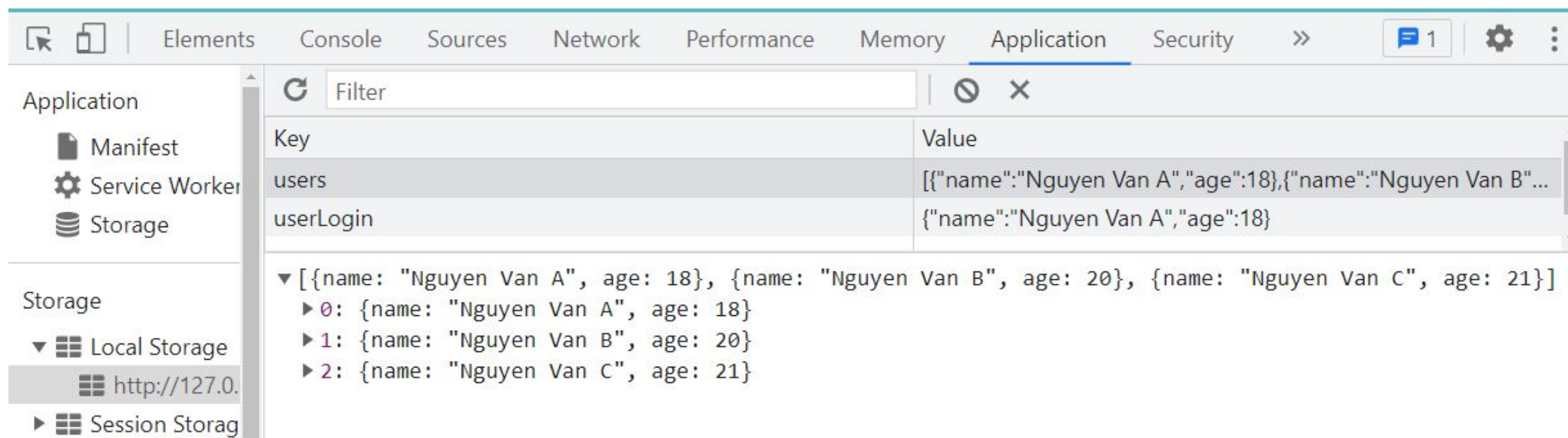
- **Phương thức chuyển đổi dữ liệu JSON**

- **JSON.stringify(value\_set):** Chuyển từ kiểu dữ liệu javascript sang JSON
- **JSON.parse("Chuỗi JSON"):** Chuyển từ kiểu JSON sang dữ liệu javascript
- *Lưu ý*
  - **key:** Một **chuỗi** chứa tên của khóa bạn muốn **tạo/cập nhật/lấy hoặc gỡ bỏ** => Luôn đưa key vào phương thức với dạng **String**
  - **value:** Một **chuỗi** chứa giá trị(**value\_set**) mà bạn muốn cung cấp (có thể là bất cứ kiểu dữ liệu nào: **String**, **Number**, **Array**, **Object**,...) cho **key** mà bạn đang **tạo/cập nhật**

## 2. LOCAL STORAGE - 3

### • Ví dụ

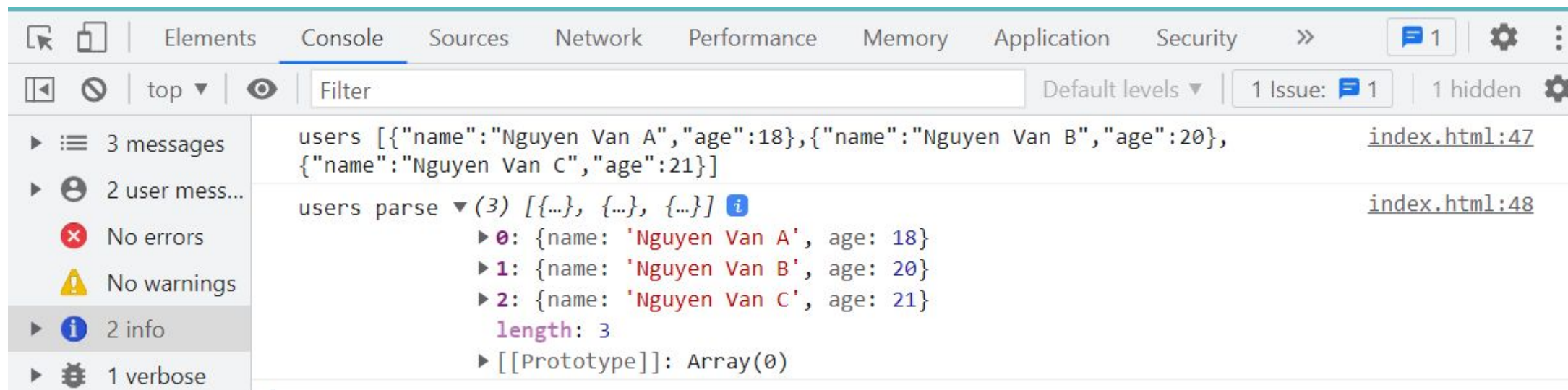
```
const persons = [
  { name: 'Nguyen Van A', age: 18 },
  { name: 'Nguyen Van B', age: 20 },
  { name: 'Nguyen Van C', age: 21 }
]
localStorage.setItem('users', JSON.stringify(persons))
localStorage.setItem('userLogin', JSON.stringify({ name: 'Nguyen Van A', age: 18 })))
```



## 2. LOCAL STORAGE - 4

- Ví dụ

```
console.log('users', localStorage.getItem('users'));  
console.log('users parse', JSON.parse(localStorage.getItem('users')));
```



## 3. SESSION STORAGE - 1

- **Định nghĩa**

- **Session Storage** là một Object lưu trữ dữ liệu của Web Storage API, được phát triển từ phiên bản HTML5.
- Dữ liệu của session storage chỉ tồn tại khi trình duyệt được mở (một phiên – a session), dữ liệu của session storage sẽ bị xóa ngay khi đóng trình duyệt
- Một số phương thức với sessionStorage:
  - sessionStorage.**getItem**(key): lấy dữ liệu theo key
  - sessionStorage.**setItem**(key, value): thêm dữ liệu
  - sessionStorage.**removeItem**(key): xóa dữ liệu theo key
  - sessionStorage.**clear**(): xóa toàn bộ dữ liệu trong sessionStorage



# 3. SESSION STORAGE -

## • Ví dụ

```
const persons = [
  { name: 'Nguyen Van A', age: 18 },
  { name: 'Nguyen Van B', age: 20 },
  { name: 'Nguyen Van C', age: 21 }
]
localStorage.setItem('users', JSON.stringify(persons))
localStorage.setItem('userLogin', JSON.stringify({ name: 'Nguyen Van A', age: 18 })))
```

The screenshot shows the Chrome DevTools Application tab. The left sidebar has a tree view with 'Application' expanded, showing 'Manifest', 'Service Worker', and 'Storage'. Under 'Storage', 'Local Storage' and 'Session Storage' are visible. 'Session Storage' is selected, showing a single entry for 'http://127.0.0.1:5500/'. The main pane displays a table of session storage items:

Key	Value
IsThisFirstTime_Log_From_LiveServer	true
users	[{"name":"Nguyen Van A","age":18}, {"name":"Nguyen Van B"...
userLogin	{"name":"Nguyen Van A","age":18}

Below the table, the 'users' value is expanded, showing an array of three objects:

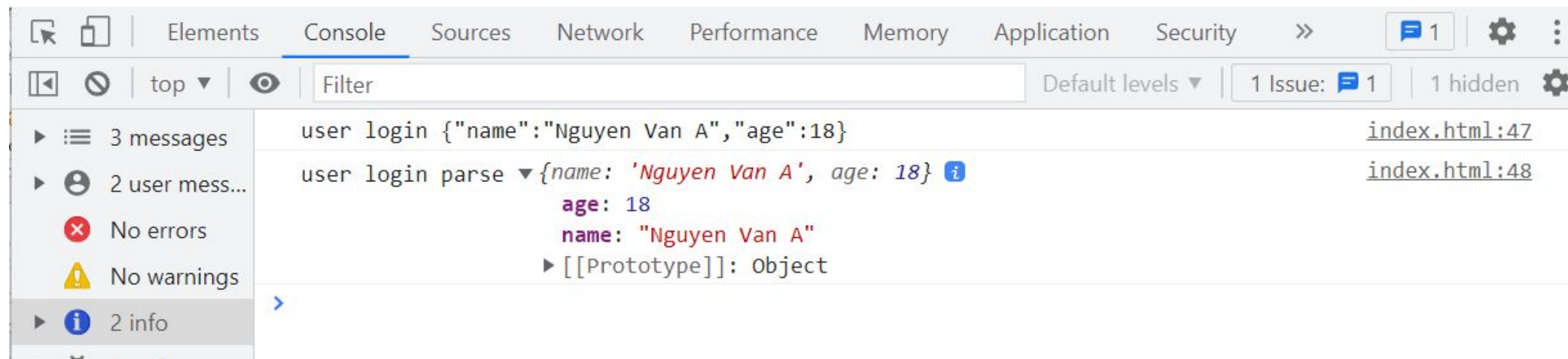
```
[{name: "Nguyen Van A", age: 18}, {name: "Nguyen Van B", age: 20}, {name: "Nguyen Van C", age: 21}]
```

- 0: {name: "Nguyen Van A", age: 18}
- 1: {name: "Nguyen Van B", age: 20}
- 2: {name: "Nguyen Van C", age: 21}

## 3. SESSION STORAGE - 3

- Ví dụ

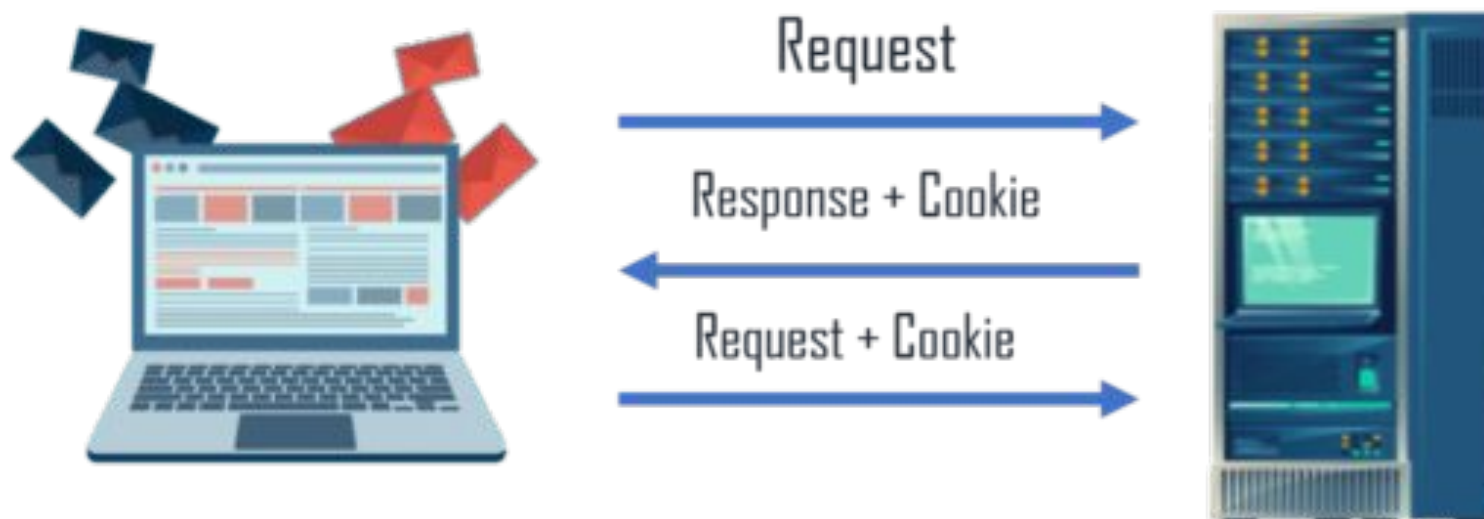
```
console.log('user login', sessionStorage.getItem('userLogin'));  
console.log('user login parse', JSON.parse(sessionStorage.getItem('userLogin')));
```



## 4. COOKIE - 1

- **Định nghĩa**

- **Cookie** là dữ liệu mà server(máy chủ) gửi lên client (trình duyệt) trong header của request HTTP, thời gian lưu trữ của cookie có thể là vĩnh viễn hoặc cũng có thể là một thời gian nhất định (do chúng ta thiết lập)
- Cookie chứa thông tin dưới dạng chuỗi thường ở dạng cặp tên-giá trị (name-value) được phân tách bằng dấu chấm phẩy. Nó duy trì trạng thái của người dùng và ghi nhớ thông tin của người dùng trong suốt tất cả các trang web.



## 4. COOKIE - 2

- **Cookie có một số trường thông tin sau:**

- **Expires** – Ngày cookie sẽ hết hạn. Nếu nó để trống, thì cookie sẽ hết hạn khi truy cập thoát khỏi trình duyệt
- **Domain** – Tên miền của site người dùng.
- **Path** – Đường truyền tới thư mục hoặc trang web mà thiết lập cookie. Nó có thể là trống nếu bạn muốn thu nhận cookie từ bất kỳ thư mục hoặc trang nào.
- **Secure** – Nếu trường này chứa từ “secure”, thì khi đó cookie chỉ có thể được thu nhận với một Server an toàn. Nếu trường này là trống, sẽ không có giới hạn nào.
- **name = value** – Cookie được thiết lập và được thu nhận cặp name-giá trị (name - value)

- **Đặc điểm:**

- Nhờ khả năng ghi nhớ và lưu trữ thông tin thành các tệp, Cookie giúp người dùng website cảm thấy tiện lợi, nhanh chóng và hữu ích hơn mỗi lần truy cập website từ lần thứ 2 trở đi vì không phải tốn công đăng nhập từ đầu.
- tệp Cookie được sử dụng để nghiên cứu và khảo sát hành vi khách hàng đối với các website doanh nghiệp, tổ chức. Đây là một yếu tố quan trọng trong các chiến lược [SEO marketing](#) hiện nay
- Từ các kết quả mà Cookie thu thập được, doanh nghiệp sẽ biết rõ về thói quen và nhu cầu của tệp khách hàng tiềm năng, từ đó đưa ra các chiến lược tối ưu website của mình để cạnh tranh với các đối thủ khác.

## 4. COOKIE - 3

### Ví dụ

- Trong JavaScript, chúng ta có thể tạo, đọc, cập nhật và xóa cookie bằng cách sử dụng thuộc tính **document.cookie**

- Tạo mới, thay đổi cookie :**

```
document.cookie = "name1=value2;name2=value2;nameN=valueN"
```

- Ví dụ:**




```
document.cookie = "username=JavaScript Partime; expires=Thu, 05 Jan 2024 23:59:00 UTC; path=/";
```

*Lưu ý: expires phải đúng định dạng của date*

- Đọc cookie**

```
let x = document.cookie;
```

## 5. SO SÁNH

			
Criteria	Local Storage	Session Storage	Cookies
Storage Capacity	5-10 mb	5-10 mb	4 kb
Auto Expiry	No	Yes	Yes
Server Side Accessibility	No	No	Yes
Data Transfer HTTP Request	No	No	Yes
Data Persistence	Till manually deleted	Till browser tab is closed	As per expiry TTL set



# KẾT THÚC

HỌC VIỆN ĐÀO TẠO LẬP TRÌNH CHẤT LƯỢNG NHẬT BẢN