Le **localStorage** est une API de stockage web qui permet de sauvegarder des données directement dans le navigateur de l'utilisateur. Voici une explication complète :

**Qu'est-ce que le localStorage ?**

Le localStorage fait partie des **Web Storage APIs** et permet de stocker des données de manière **persistante** dans le navigateur. Contrairement aux cookies, ces données ne sont pas envoyées automatiquement au serveur à chaque requête.

**Caractéristiques principales**

**Persistance** : Les données restent même après fermeture du navigateur (contrairement au sessionStorage qui se vide à la fermeture de l'onglet)

**Capacité** : Environ 5-10 MB par domaine (beaucoup plus que les cookies qui sont limités à 4 KB)

**Portée** : Les données sont isolées par domaine et protocole (<http://example.com> et <https://example.com> ont des localStorage séparés)

**Synchrone** : Les opérations sont bloquantes (contrairement à IndexedDB qui est asynchrone)

**Méthodes principales**

javascript

*// Stocker une valeur*

localStorage.setItem('clé', 'valeur');

*// Récupérer une valeur*

const valeur = localStorage.getItem('clé');

*// Supprimer une clé spécifique*

localStorage.removeItem('clé');

*// Vider tout le localStorage*

localStorage.clear();

*// Obtenir le nombre d'éléments*

const nombreElements = localStorage.length;

*// Obtenir une clé par son index*

const clé = localStorage.key(0);

**Limitation importante : Que des chaînes de caractères**

Le localStorage ne peut stocker que des **strings**. Pour les objets ou tableaux, il faut utiliser JSON :

javascript

*// Stocker un objet*

const utilisateur = { nom: 'Jean', age: 30 };

localStorage.setItem('utilisateur', JSON.stringify(utilisateur));

*// Récupérer un objet*

const utilisateurRecupere = JSON.parse(localStorage.getItem('utilisateur'));

**Exemple concret dans votre code**

Dans votre fichier modal.js, le localStorage est utilisé pour stocker le **token d'authentification** :

javascript

*// Récupération du token pour les requêtes API*

const token = localStorage.getItem("token");

*// Utilisation dans les headers*

headers: { Authorization: `Bearer ${token}` }

Cela permet de :

1. **Maintenir la session** : L'utilisateur reste connecté même après rechargement de la page
2. **Authentifier les requêtes** : Chaque appel API inclut le token pour prouver l'identité
3. **Éviter les reconnexions** : Pas besoin de se reconnecter à chaque visite

**Cas d'usage courants**

* **Authentification** : Stocker tokens, session utilisateur
* **Préférences** : Thème sombre/clair, langue, paramètres
* **Cache local** : Données fréquemment utilisées
* **Panier e-commerce** : Articles avant connexion
* **Brouillons** : Formulaires en cours de rédaction

**Bonnes pratiques**

**Vérifier l'existence** avant d'utiliser :

javascript

if (localStorage.getItem('token')) {

*// L'utilisateur est connecté*

}

**Gérer les erreurs** (quotas dépassés, navigation privée) :

javascript

try {

localStorage.setItem('clé', 'valeur');

} catch (e) {

console.error('Impossible de sauvegarder', e);

}

**Nettoyer régulièrement** pour éviter l'accumulation de données obsolètes.

Le localStorage est donc un outil puissant pour améliorer l'expérienc