6 - Suppression de données

🤔 Concepts à comprendre

DELETE request

Pour supprimer une ressource via l'API :

```javascript
fetch(`${url}/${id}`, { method: 'DELETE' })
```

Points importants :

- L'ID de la ressource est dans l'URL
- Généralement pas de body dans la requête
- La réponse peut être vide (204 No Content) ou contenir un message de confirmation

Confirmation utilisateur

Principe UX : Ne jamais supprimer quelque chose sans demander confirmation !

Options :

1. `confirm()` natif JavaScript (simple mais pas très esthétique)

2. Modal Bootstrap de confirmation (plus professionnel)
3. Bouton "Annuler" avec un délai (comme Gmail)

Optimistic UI Update vs Pessimistic

Optimistic :

- On supprime d'abord de l'interface
- Puis on fait la requête API
- Si échec, on restaure l'élément
- **Avantage** : Interface réactive, sensation de rapidité
- **Inconvénient** : Donne une fausse impression si l'API échoue

Pessimistic :

- On fait d'abord la requête API
- Si succès, on supprime de l'interface
- **Avantage** : Toujours synchronisé avec l'API
- **Inconvénient** : Peut sembler lent

Recommandation : Approche pessimistic pour ce TP (plus simple et plus sûre).

✅ Tâches à réaliser

5.1 - Ajouter un bouton de suppression sur chaque carte

Objectif : Modifier votre fonction `createCarCard()` pour ajouter un bouton "Supprimer" sur chaque carte.

Réflexions :

1. **Où placer le bouton ?**
   - Dans le `card-body` à côté du bouton "See more" ?
   - En overlay sur l'image (icône de poubelle en haut à droite) ?
   - Les deux ?
2. **Style du bouton** :
   - Classe Bootstrap : `btn btn-danger` pour un bouton rouge
   - Ou `btn btn-outline-danger` pour un style plus discret
   - Icône : Utiliser Bootstrap Icons ou une simple croix
3. **Accessibilité** :
   - Texte du bouton clair : "Supprimer" ou "Delete"
   - Attribut `aria-label` si vous utilisez seulement une icône

Exemple de modification dans createCarCard() :

```javascript
// Dans votre fonction createCarCard(car) // Après avoir créé le bouton "See more" const deleteButton = document.createElement('button');
deleteButton.className = 'btn btn-danger btn-sm';
deleteButton.textContent = 'Supprimer'; // OU avec une icône :
// deleteButton.innerHTML = '<i class="bi bi-trash"></i>'; // Stocker l'ID de la voiture dans un attribut data
deleteButton.dataset.carId = car.id;
// Ajouter au card-body
cardBody.appendChild(deleteButton);
```

Question : Pourquoi utiliser `dataset.carId` ? Pour pouvoir récupérer facilement l'ID lors du clic.

5.2 - Créer une fonction de suppression

Objectif : Écrire une fonction asynchrone qui supprime une voiture via l'API.

Signature suggérée :

```javascript
async function deleteCar(carId) { // retourne true si succès, false sinon }
```

Structure :

```
FONCTION deleteCar(carId): ESSAYER: faire une requête DELETE vers l'API avec l'ID SI la réponse n'est pas OK: SI c'est une 404: la voiture n'existe déjà plus SINON: lancer une erreur retourner true EN CAS D'ERREUR: logger l'erreur afficher un message d'erreur à l'utilisateur retourner false
```

Points d'attention :

- Votre API retourne-t-elle un contenu dans la réponse DELETE ?
- Quel code de statut en cas de succès ? (200, 204, etc.)
- Comment gérer le cas où l'ID n'existe pas (404) ?

5.3 - Gérer l'événement de clic sur le bouton de suppression

Objectif : Connecter le bouton à la fonction de suppression.

Problème : Event Delegation

Si vous attachez l'événement directement sur le bouton lors de sa création

```javascript
deleteButton.addEventListener('click'
handleDelete);
```

Cela fonctionne, mais vous créez autant d'event listeners que de voitures. Pour 100 voitures = 100 listeners !

Meilleure solution : Event Delegation

Attachez un seul listener sur le conteneur parent et vérifiez quel élément a été cliqué :

```javascript
const container =
document.querySelector('.card-cont');
container.addEventListener('click',
async (event) => { // Vérifier si
l'élément cliqué est un bouton de
suppression if
(event.target.matches('button[data-
car-id]')) { // ou : if
(event.target.classList.contains('btr
delete')) const carId =
event.target.dataset.carId; await
handleCarDeletion(carId,
event.target); } });
```

Avantages :

- Un seul event listener
- Fonctionne même pour les éléments ajoutés dynamiquement
- Meilleure performance

5.4 - Implémenter la confirmation de suppression

Option 1 : confirm() natif

```javascript
async function handleCarDeletion(carId, button) {
const confirmed = confirm('Êtes-vous sûr de vouloir supprimer cette voiture ?'); if (!confirmed) {
return; // Annulation } // Continuer avec la suppression }
```

Option 2 : Modal Bootstrap de confirmation

Créer un modal dédié à la confirmation :

```
<!-- Dans index.html --> <div
class="modal fade"
id="confirmDeleteModal"
tabindex="-1"> <div class="modal-
dialog"> <div class="modal-content">
<div class="modal-header"> <h5
class="modal-title">Confirmer la
suppression</h5> <button
type="button" class="btn-close"
data-bs-dismiss="modal"></button>
</div> <div class="modal-body">
<p>Êtes-vous sûr de vouloir
supprimer cette voiture ?</p> <p
class="text-muted small">Cette
action est irréversible.</p> </div>
<div class="modal-footer"> <button
type="button" class="btn btn-
secondary" data-bs-dismiss="modal">
Annuler </button> <button
type="button" class="btn btn-danger"
id="confirmDeleteBtn"> Supprimer
</button> </div> </div> </div>
</div>
```

Gestion en JavaScript :

```javascript
let carToDelete = null; // Variable p
async function handleCarDeletion(carI
de la voiture à supprimer carToDelete
de confirmation const confirmModal =
bootstrap.Modal(document.getElementBy
confirmModal.show(); } // 3. Attacher
confirmation du modal
document.getElementById('confirmDelet
async () => { if (!carToDelete) retur
const success = await deleteCar(carTo
Supprimer la carte de l'interface //
message de succès } // Réinitialiser
```

5.5 - Supprimer la carte de l'interface

Objectif : Une fois la suppression confirmée par l'API, retirer visuellement la carte de la page.

Trouver et supprimer la carte

```javascript
// Depuis le bouton de suppression,
remonter jusqu'à l'article const
card = button.closest('.card');
card.remove();
```

💡 Points de vigilance

État désynchronisé

Si plusieurs utilisateurs utilisent l'application simultanément :

- User A supprime une voiture
- User B voit encore cette voiture et essaie de la supprimer
- Gérer ce cas avec un message approprié (404 = déjà supprimée)