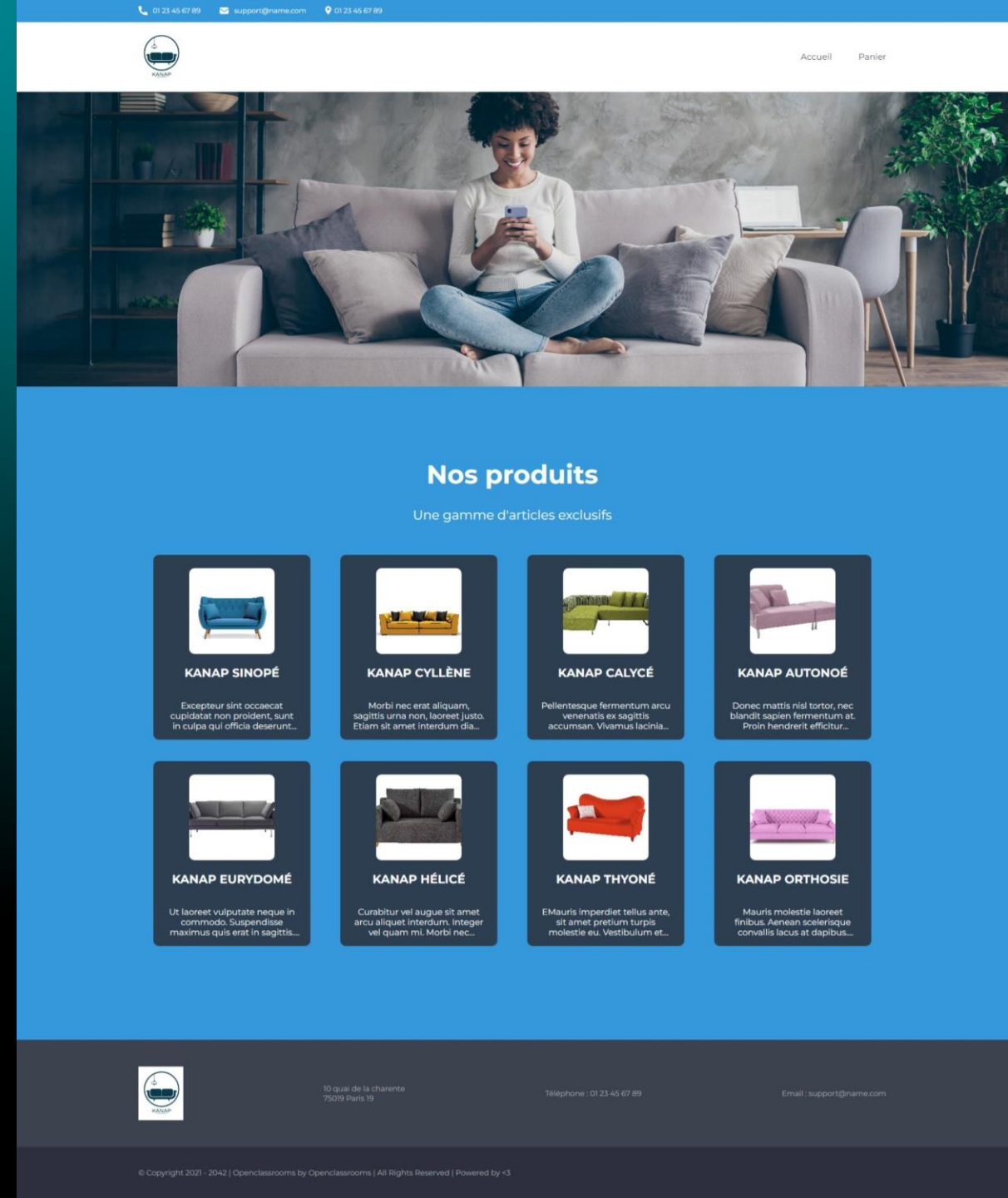


PRESENTATION DU PROJET 5

Kanap



Technologies utilisées


- HTML, CSS, JavaScript.
- Visual Studio Code, Node
- Outils de développement
- Git et GitHub
- API (interface de programmation)

Page d'accueil avec nos produits

09 23 45 67 89

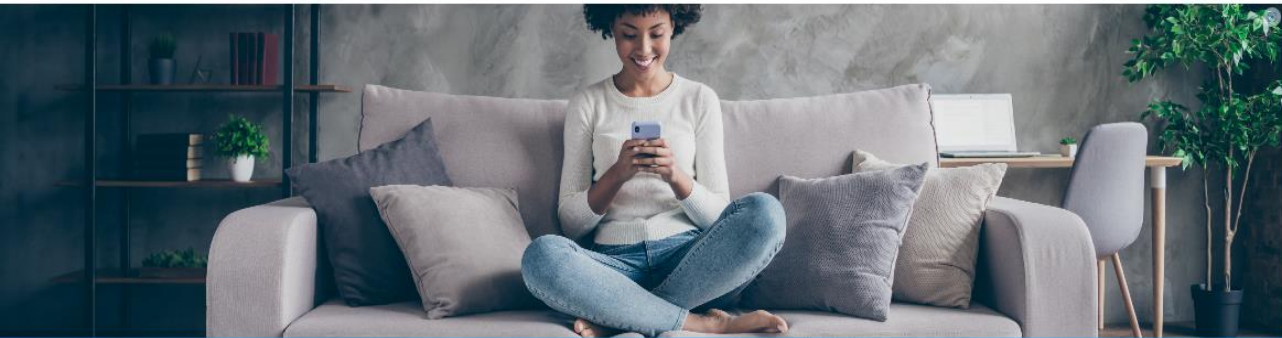
support@name.com

09 23 45 67 89




Accueil

Panier




Nos produits

Une gamme d'articles exclusifs




KANAP SINOPE

Excepleur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt...




KANAP CYLLÈNE

Morbi nec erat aliquam, sagittis urna non, laoreet justo. Etiam sit amet interdum dia...




KANAP CALYCÉ

Pellentesque fermentum arcu venenatis ex sagittis accumsan. Vivamus lacinia...




KANAP AUTONOE

Donec mattis risi tortor, nec blandit sapien fermentum at. Proin hendrerit efficitur...




KANAP EURYDOMÉ

Ut laoreet vulputate neque in commodo. Suspendisse maximus quis erat in sagittis...




KANAP HÉLICÉ

Curabitur vel augue sit amet arcu aliquet interdum. Integer vel quam mi. Morbi nec...



KANAP THYONÉ

Et Mauris imperdiet tellus ante, sit amet pretium turpis molestie eu. Vestibulum et...



KANAP ORTHOSIE

Mauris molestie laoreet finibus. Aenean scelerisque convallis lacus at dapibus...

Application

Manifeste

Service Workers

Stockage

Stockage

Stockage local

http://127.0.0.1:5500

Stockage de session

IndexedDB

Web SQL

Cookies

Jetons d'approbation

Groupes d'intérêt

Cache

Stockage en cache

Cache précédent/avant

Services d'arrière-plan

Extraction en arrière-plan

Synchronisation en arrière-pl

Notifications

Gestionnaire de paiement

Synchronisation périodique e

Messagerie Push

Rapport API

Clé

Valeur

cart

▼ []

Aucune propriété

Console

Problèmes

Couverture

Ressources pour les développeurs

Console réseau

2 messages

1 user message

No errors

No warnings

1 info

1 verbose


Promise {<pending>}


(index)	colors	_id	name	price	imageU	descri	altTxt
0	Array(_	'107fb...	'Kanap...	1849	'http:...	'Excep...	"Photo...
1	Array(_	'415b7...	'Kanap...	4499	'http:...	'Morbi...	"Photo...
2	Array(_	'05574...	'Kanap...	3199	'http:...	'Pelle...	"Photo...
3	Array(_	'a5572...	'Kanap...	1499	'http:...	'Donec...	"Photo...
4	Array(_	'8906d...	'Kanap...	2249	'http:...	'Ut la...	"Photo...
5	Array(_	'77711...	'Kanap...	999	'http:...	'Curab...	"Photo...
6	Array(_	'03470...	'Kanap...	1999	'http:...	'EMaur...	"Photo...
7	Array(_	'a6ec5...	'Kanap...	3999	'http:...	'Mauri...	"Photo...


Array(8)

La page panier

01 23 45 67 89support@name.com01 23 45 67 89

AccueilPanier





Kanap Sinopé

Prix : 1849€

Description :

Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Choisir une couleur : Blue

Nombre d'article(s) (0-100) : 1

Ajouter au panier

BienvenueÉlémentsConsoleSourcesApplication x16

Application

ManifesteService WorkersStockage

Stockage

Stockage localhttp://127.0.0.1:5500Stockage de sessionIndexedDBWeb SQLCookiesJetons d'approbationGroupes d'intérêt

Cache

Stockage en cacheCache précédent/avant

Services d'arrière-plan

Extraction en arrière-planSynchronisation en arrière-planNotificationsGestionnaire de paiementSynchronisation périodique eMessagerie PushRapport API

CléValeur

cart[[{"color": "Blue", "id": "107fb5b75607497b96722bda5b504926", "quantity": 1}]]

ConsoleProblèmesCouvertureRessources pour les développeursConsole réseau+X

topFilterNiveaux par défaut16

2 messages

1 user message

No errors

No warnings

1 info

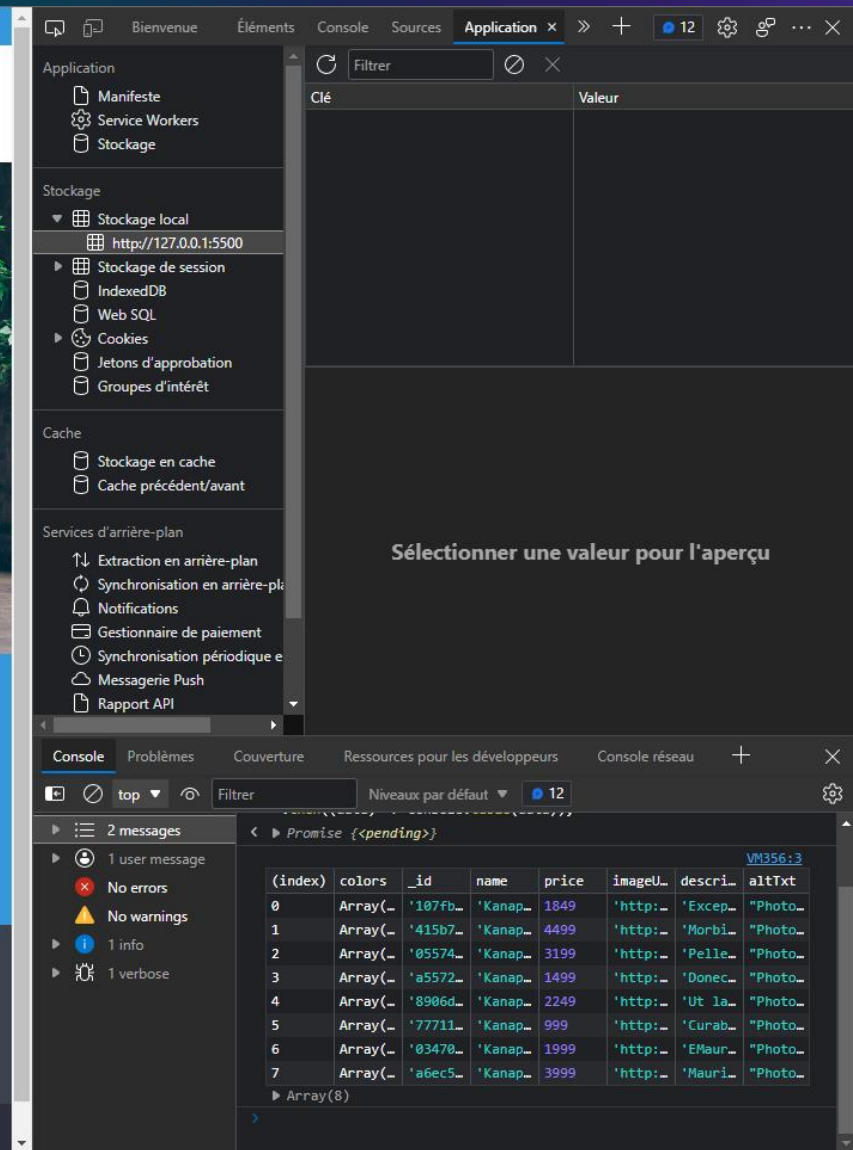
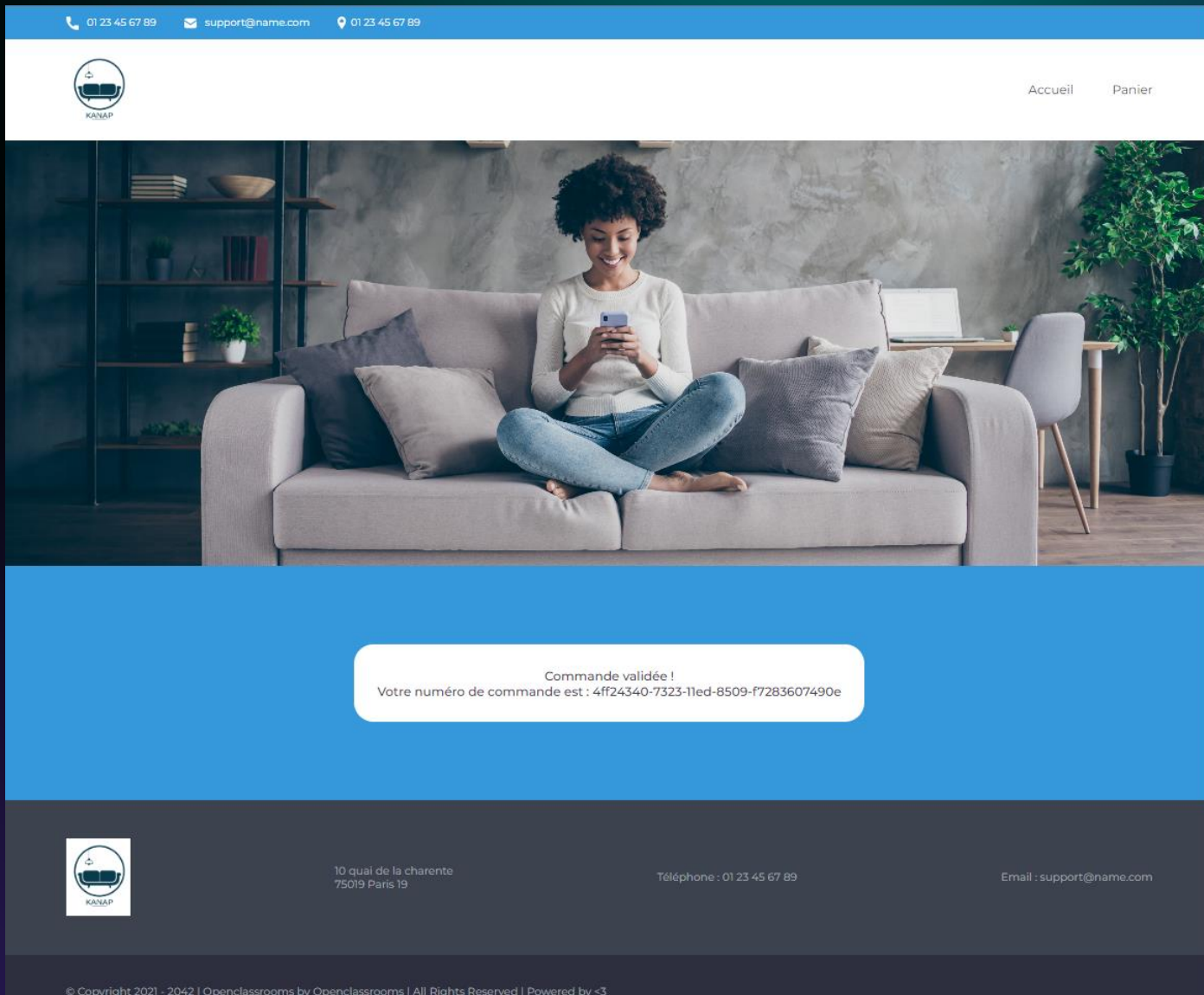
1 verbose

Promise {<pending>}

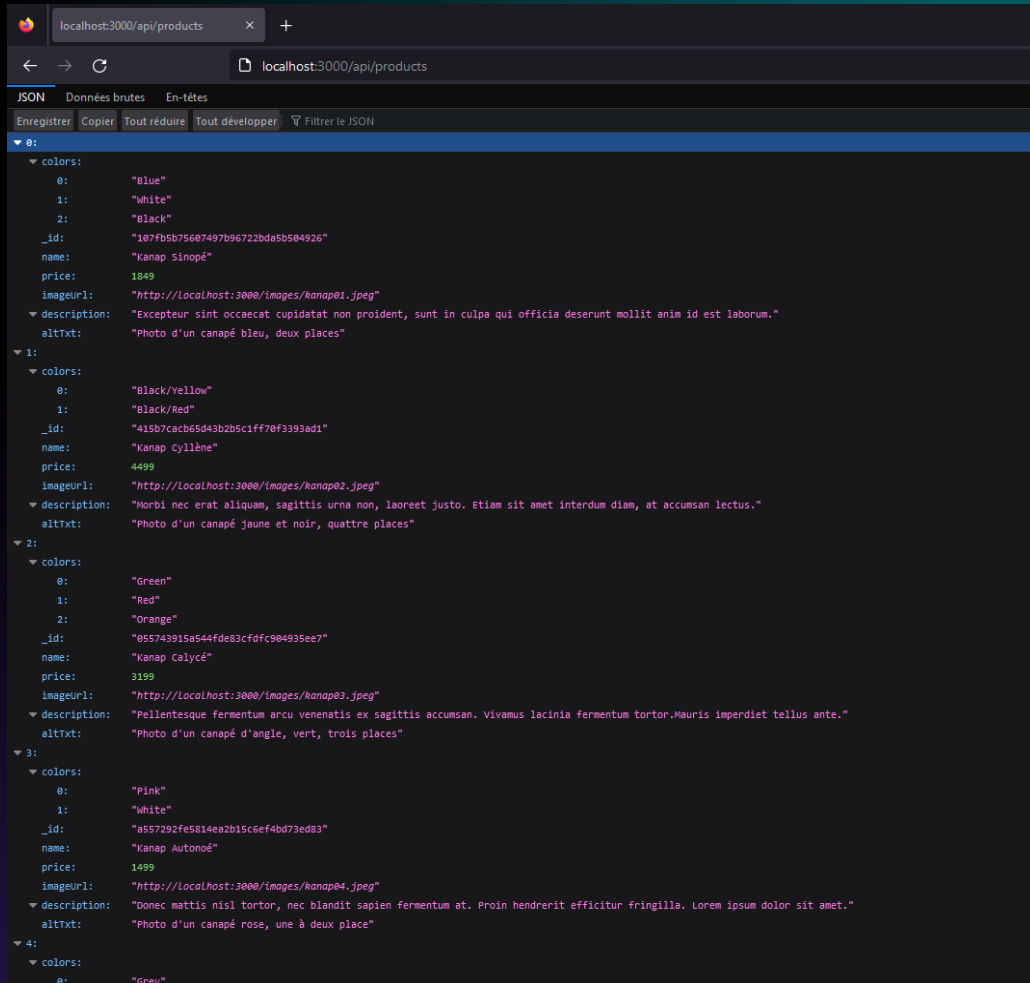
(index)	colors	_id	name	price	imageU	descri	altTxt
0	Array(1)	'107fb...	'Kanap...	1849	'http:...	'Excep...	'Photo...
1	Array(1)	'415b7...	'Kanap...	4499	'http:...	'Morbi...	'Photo...
2	Array(1)	'05574...	'Kanap...	3199	'http:...	'Pelle...	'Photo...
3	Array(1)	'a5572...	'Kanap...	1499	'http:...	'Donec...	'Photo...
4	Array(1)	'8906d...	'Kanap...	2249	'http:...	'Ut la...	'Photo...
5	Array(1)	'77711...	'Kanap...	999	'http:...	'Curab...	'Photo...
6	Array(1)	'03470...	'Kanap...	1999	'http:...	'EMaur...	'Photo...
7	Array(1)	'a6ec5...	'Kanap...	3999	'http:...	'Mauri...	'Photo...

Array(8)

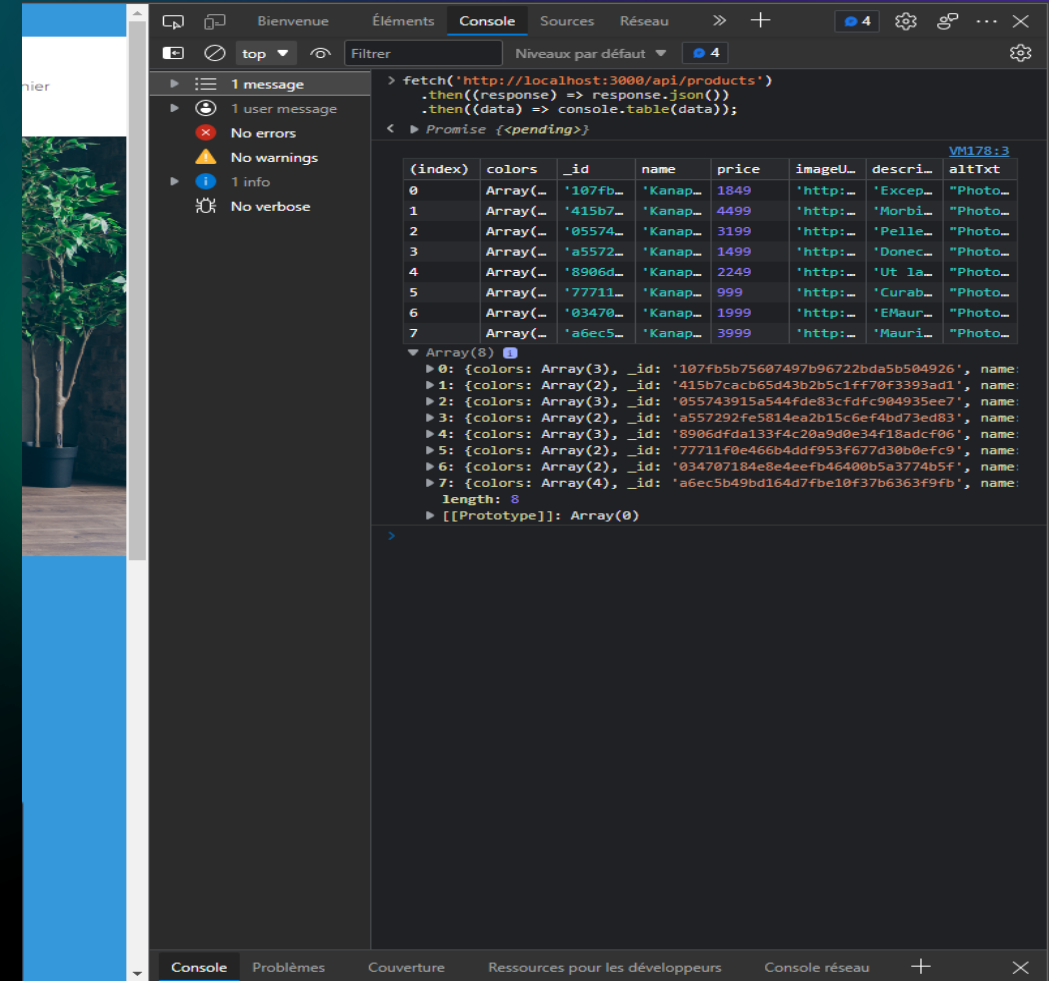
La page confirmation



Récupération de la liste de produit



La liste des produits retourné par l'API en JSON
URL des API
Catalogue de canapés :
<http://localhost:3000/api/products>



Requêter l'API avec la méthode Fetch pour
récupérer l'ensemble des produits puis
insérer chaque élément dans la page
d'accueil

Affichage des produits de façon dynamique dans le stockage local.



Affichage des produits de façon statique dans la console sous forme d'un tableau.



The screenshot shows the VS Code interface with the following components:

- Left Panel:** Contains the 'Application' view with sections for 'Stockage' (Local, Session, IndexedDB, Web SQL, Cookies, etc.) and 'Cache'.
- Right Panel (Top):** Displays the 'Clé' (Key) and 'Valeur' (Value) for the 'cart' key. The value is a JSON array: `[{"color": "Blue", "id": "107fb5b75607497b96722bda5b504926", "quantity": 1}]`.
- Right Panel (Bottom):** Shows the 'Console' tab with a message: `[Violation] 'setTimeout' handler took 202ms`. Below this, a `fetch` call is shown, and the response is displayed as a table.

(index)	colors	_id	name	price	imageUr1	descripti...	altTxt
0	Array(3)	'107fb5b7...	'Kanap Si...	1849	'http://1...	'Excepteu...	"Photo d'...
1	Array(2)	'415b7cac...	'Kanap Cy...	4499	'http://1...	'Morbi ne...	"Photo d'...
2	Array(3)	'05574391...	'Kanap Ca...	3199	'http://1...	'Pellente...	"Photo d'...
3	Array(2)	'a557292f...	'Kanap Au...	1499	'http://1...	'Donec ma...	"Photo d'...
4	Array(3)	'8906dfda...	'Kanap Eu...	2249	'http://1...	'Ut laore...	"Photo d'...
5	Array(2)	'77711f0e...	'Kanap Hé...	999	'http://1...	'Curabitu...	"Photo d'...
6	Array(2)	'03470718...	'Kanap Th...	1999	'http://1...	'EMauris ...	"Photo d'...
7	Array(4)	'a6ec5b49...	'Kanap or...	3999	'http://1...	'Mauris m...	"Photo d'...

Quelques exemples de l'utilisation de l'Api

Requêter l'API avec la methode Get pour obtenir l'ensemble des produits

```
15  
16  function createProductsCards() {  
17      fetch("http://localhost:3000/api/products")  
18      .then((response) => {  
19          return response.json();  
20      })
```


Requête POST avec fetch

la fonction `fetch()` pour envoyer une requête HTTP POST à l'URL spécifiée et la méthode "POST" indique au serveur que nous voulons envoyer des données au serveur.

```
376 mergeInputs(); // la fonction "mergeInputs" est appelée, puis une requête HTTP POST est envoyée à l'URL "http://localhost:3000/api/products/  
377 const url = "http://localhost:3000/api/products/order";  
378 fetch(url, {  
379     // méthode fetch() pour récupérer les données  
380     method: "POST", // méthode POST pour envoyer des données au serveur  
381     body: JSON.stringify(orderProducts), // méthode stringify() pour convertir un objet JavaScript en chaîne de caractères JSON  
382     headers: {  
383         "Content-Type": "application/json", // Définit le type de données à envoyer au serveur (ici, "POST" et "JSON" )  
384     },  
385 })  
386 .then((response) => {  
387     return response.json();  
388 })  
389 .then((response) => {  
390     localStorage.clear(); // Cela vide le stockage local et redirige l'utilisateur vers une page de confirmation avec un paramètre d'URL "ord  
391     window.location.href = `confirmation.html?order=${response.orderId}`; // méthode location.href pour rediriger l'utilisateur vers une page  
392 })  
393 .catch((error) => {  
394     console.log(error);  
395 });  
396 } else {  
397     alert(  
398         "Le formulaire n'a pas pu être validé. Tous les champs sont-ils correctement remplis?"  
399     );  
400 }  
401 }
```

Le DOM

(ajout, suppression et modification d'éléments)

```
💡 "createElement" cree un nouvel element HTML
const createInnerContent = function () {
  cartContent += `<article class="cart__item" data-id="${cart[i].id}" data-color="${productColor}">
    <div class="cart__item__img"></div>
    <div class="cart__item__content">
      <div class="cart__item__content__description">
        <h2>${productName}</h2>
        <p>${productColor}</p>
        <p>${productPrice} €</p>
      </div>
      <div class="cart__item__content__settings">
        <div class="cart__item__content__settings__quantity">
          <p>Qté : </p>
          <input type="number" class="itemQuantity" name="itemQuantity" min="1" max="100" value="${cart[i].quantity}">
        </div>
        <div class="cart__item__content__settings__delete">
          <p class="deleteItem">Supprimer</p>
        </div>
      </div>
    </div>
  </article>`;
};
```

URLSearchParams

manipuler les paramètres de recherche dans une URL

```
14
15 // const currentPageUrl = document.location.href; // création de l'objet currentPageUrl
16 // const url = new URL(currentPageUrl); // crée un nouvel objet url
17 // const orderId = url.searchParams.get("order"); // URLSearchParams manipuler les paramètres de recherche dans une
18 // const orderIdSpan = document.getElementById("orderId");
19 // const response = new URL("http://localhost:3000/api/order"); // création de l'objet response
20 console.log(response);
21 // orderIdSpan.textContent = `${orderId}`; // la propriété textContent de l'objet orderIdSpan pour rempl
22
23 // ce code affiche l'ID de commande de la page courante dans un élément HTML avec l'ID orderId
24
25 // la page de confirmation est affichée après avoir validé le formulaire de commande
26
```


L'utilisation du local storage

```
129  const pushLocalStorageQuantity = function () {
130  // "push" ajoute un ou plusieurs elements a la fin d'un tableau et retourne la nouvelle taille du tableau
131  getParentArticle = this.closest("article");
132  getProductToUpdate();
133  if (updatedProduct) {
134      const indexOfUpdatedProduct = cart.indexOf(updatedProduct);
135      updatedProduct.quantity = parseInt(
136          quantityInputField[indexOfUpdatedProduct].value
137      );
138  localStorage.setItem("cart", JSON.stringify(cart)); // "setItem" ajoute un element au localStorage
139  eventListener();
140  getCartTotal();
141  convertCartToArray();
142  }
143  };
144  // localStorage
```

Fin de la presentation