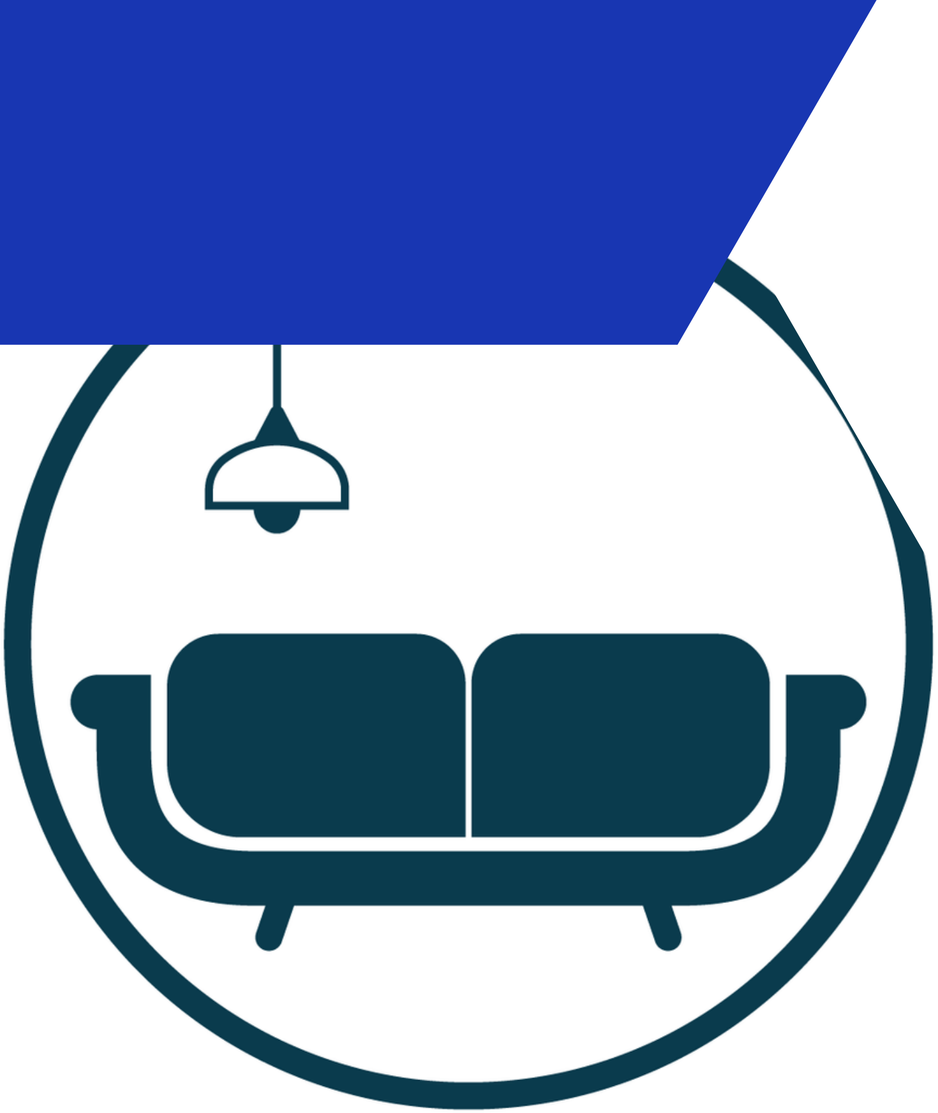


PROJET KANAP

But du projet : Construire un site e-commerce de vente de canapés

Objectifs:

Rendre le site dynamique en utilisant le langage Java Script

Etablir un plan de test et tester le site

Présenté par : Magali Bernard

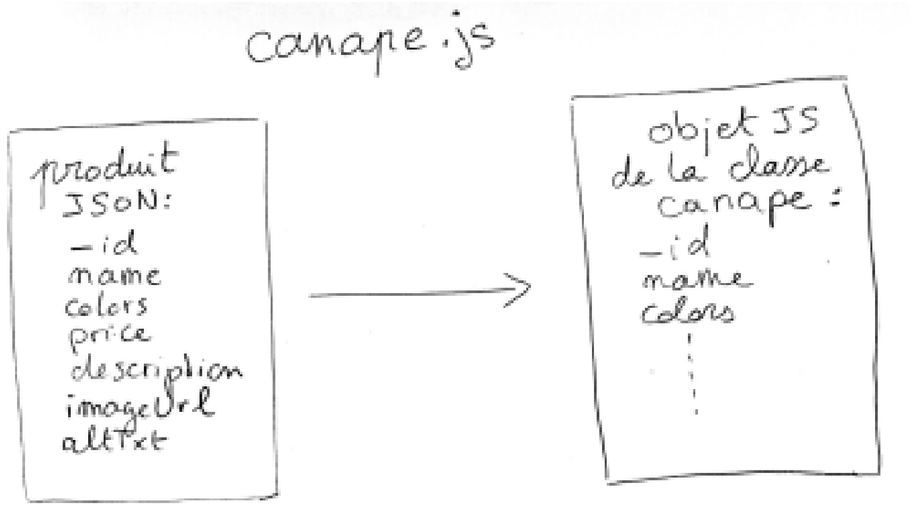
Date : 23/09/22

Page d'acceuil : Fonctions implémentées

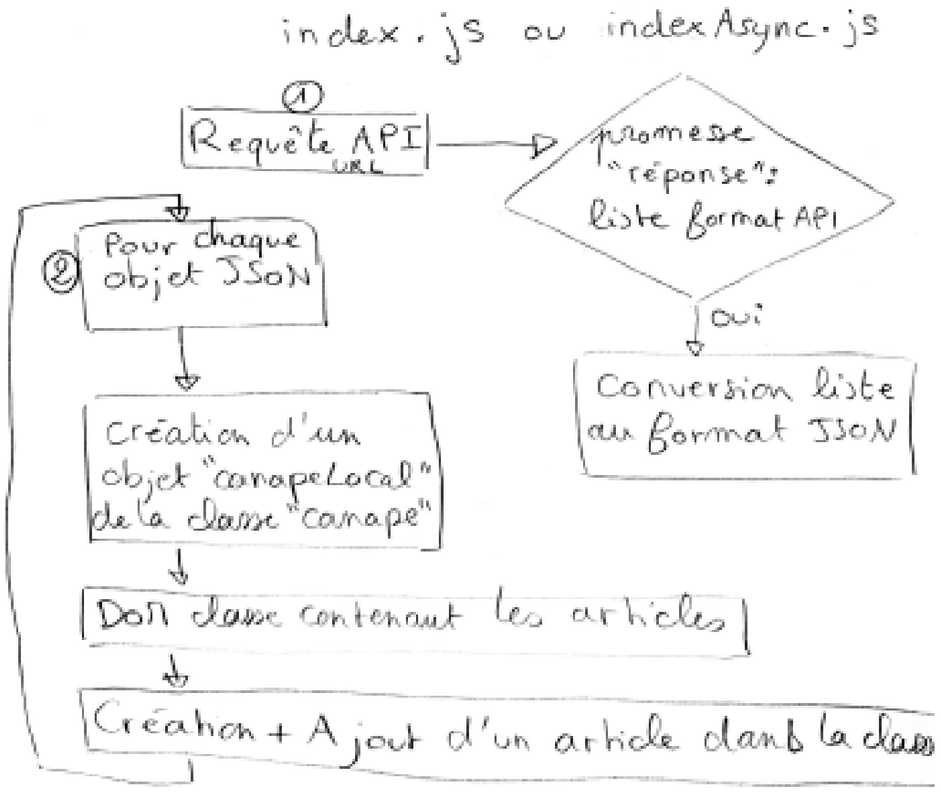
1. *Récupèrer les données sur les canapés dans l'API à l'adresse "http://localhost:3000/api/products"*

2. Traiter ces données pour créer des liens html correspondants qui s'affichent sur la page d'acceuil.

3. Ces liens pointent vers la page produit dont l'URL contient l'id du produit cliqué.




```
class canape{
    constructor(jsonProduct){
        jsonProduct && Object.assign(this, jsonProduct);
    }
}
```




```
//L'utilisation de async(inutile pour "reponse" qui est une promesse et qui est donc déjà asynchrone)
et de await permet au compilateur de continuer
// à executer la suite du code JS même si ce qui est attendu par la fonction asynchrone n'est pas
encore arrivé et de revenir executer la fonction
//asynchrone lorsque ce qui est attendu sera arrivé.

let reponse =await fetch("http://localhost:3000/api/products");

//le resultat "reponse" d'une promesse étant une promesse, idem pour le resultat "listProductJson" de
la preomesse "reponse":
let listProductJson=await reponse.json();


    for (let ProductJson of listProductJson) {
      let canapeLocal = new canape(ProductJson);
      document.querySelector(
        ".items"
      ).innerHTML += ` <a href="./product.html?id=${canapeLocal._id}">
            <article>
              <img src=${canapeLocal.imageUrl} alt=${canapeLocal.altTxt}>
              <h3 class="productName">${canapeLocal.name}</h3>
              <p class="productDescription">${canapeLocal.description}</p>
            </article>
          </a> `;
    }
```

Page produit: Fonctions implémentées

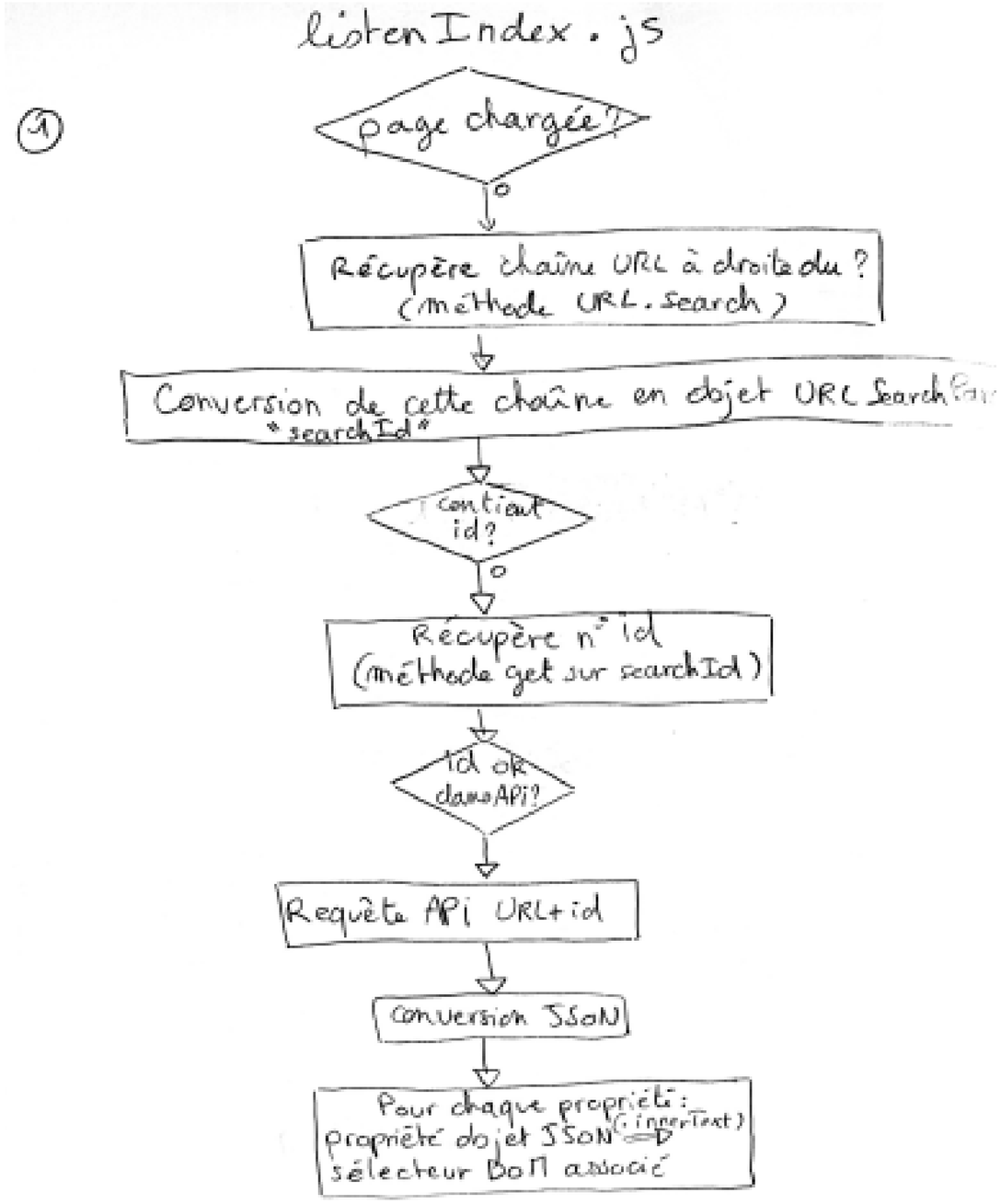
1

Inspecter l'url de la page produit et en extraire l'id du produit cliqué en page d'acceuil

Requêter l'API grâce à l'id pour obtenir les caractériqtiques du produit à afficher

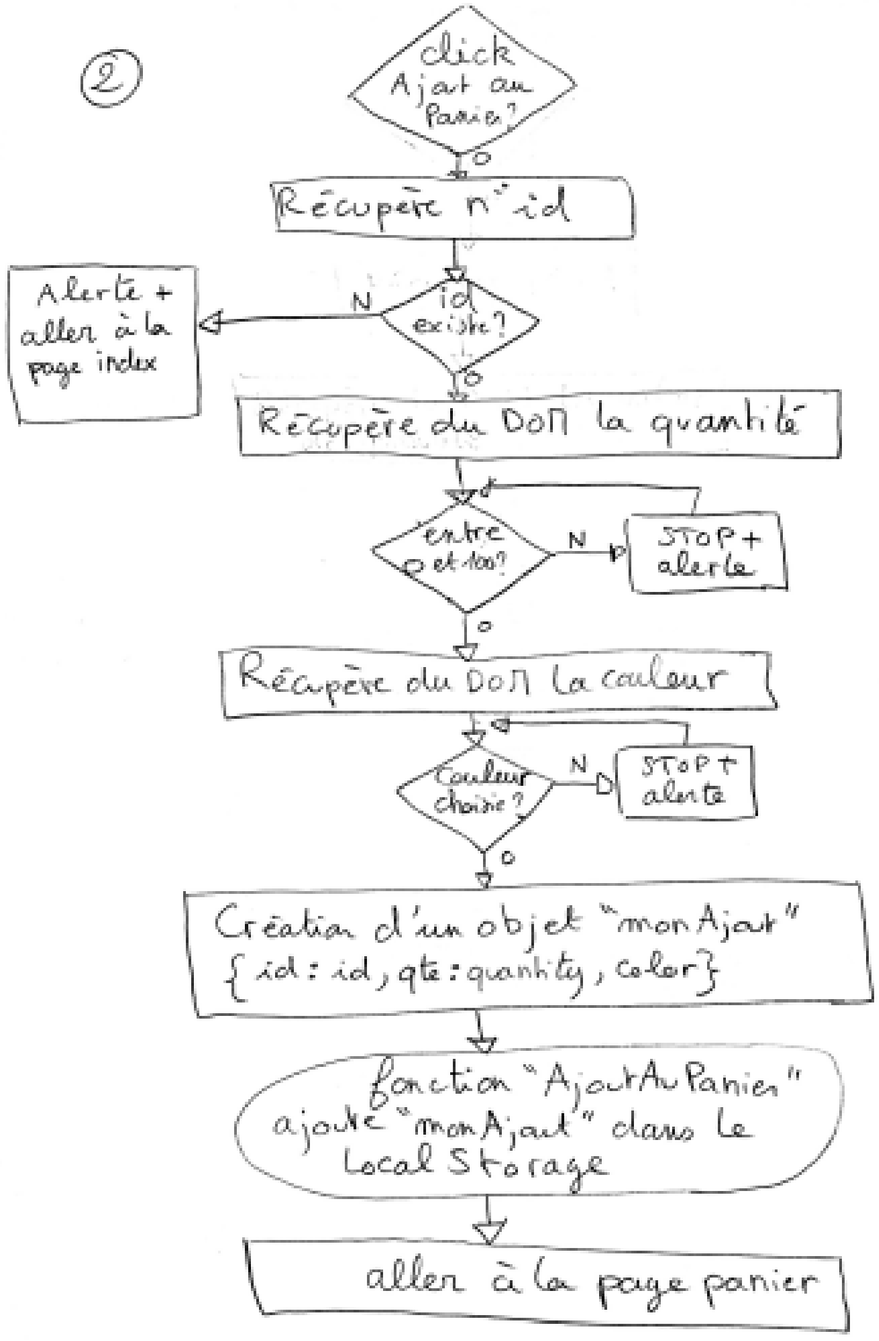
2

Ecouter le click sur le bouton d'ajout au panier et dans ce cas récupérer la couleur et la quantité choisies par le client dans un objet {id,quantite,couleur}puis le stocker dans le local storage




```
let searchId = new URLSearchParams(window.location.search);

// console.log(searchId.has('id'));      You, maintenant •
if (searchId.has("id")) {
  let canapId = searchId.get("id");
```

```
        let myPromise = fetch("http://localhost:3000/api/products/" + canapId); //le résultat de la
promesse contient l'article voulu au format API
        myPromise
          .then(function (result) {
            if (result) {
              return result.json();
            }
          })
          .then((ProductJson) => {

            //-------------------------------------AFFICHAGE DU CHOIX DE COULEURS SPECIFIQUE A
L'ARTICLE-----------------------------------------------------------
            for (let color of ProductJson.colors) {
              //pour chaque item du tableaux des couleurs de l'article, on ajoute cette couleurs aux
autres dans le html de l'article.
              // console.log(color);
              document.querySelector(
                "#colors"
              ).innerHTML += ` <option value="${color}">${color}</option> `;
            }

            //-----------------------------------AFFICHAGE DES AUTRES
PROPRIETES-----------------------------------------------------------------------
            document.querySelector(
              ".item__img"
            ).innerHTML = `<img src=${ProductJson.imageUrl}>`;
            document.querySelector("#title").innerText = ProductJson.name;
            document.querySelector("#price").innerText = ProductJson.price;
            document.querySelector("#description").innerText =
              ProductJson.description;
          })

          .catch((error) => {
            console.log(error);
          });
```




```
        let buttonAjoutPanier = document.getElementById("addToCart"); //on recupere le pointage du BOUTON dans le
DOM de cart.html dans une variable
        buttonAjoutPanier.addEventListener(
          "click",
          () => {
            let searchId = new URLSearchParams(window.location.search); //URL.search correspond à la chaine de
caractère de l'url à droite du ?
            let canapId = searchId.get("id");

            //récup de la quantité choisie par le user dans le DOM à condition qu'elle soit valide (comprise entre
1 et 100) sinon alerte et mise à 0:
            //  let quantite = document.querySelector("#quantity");

            let quantity = document.getElementById("quantity").value;

            if (quantity <= 0 || quantity > 100) {
              alert("La quantité doit être un nombre entre 1 et 100.");
              console.log(quantity);
            } else {
              //récup de la couleur DOM si elle est choisie:

              let color = document.getElementById("colors").value;
              if (color != "") {
                //Ajout de l'objet {id,qte,color} au LocalStorage via la fonction ajoutAuPanier:
                let monAjout = { id: canapId, qte: quantity, color }; //objet js au format Json SANS guillemet
permis car on est en JS
                ajoutAuPanier(monAjout);
                window.location = "cart.html";
              } else {
                alert("Sélectionner une couleur avant de commander");
              }
            }
          },
          false
        );
```

Page panier: Fonctions de gestion du LocalStorage

Gérer le panier du localstorage (LS): ajout, suppression, sauvegarde, récupération

```
//LS:CETTE FONCTION SAUVE UN TABLEAU D'OBJETS {ID,QTE,COULEUR} DANS LE LS AU FORMAT TXT
function savePanier(Panier) {
  //Panier est un tableau d'objet JSON indice: 0 valeur: {Id,qte,couleur}      You, il y a 1 seconde • Un
  localStorage.setItem("monPanier", JSON.stringify(Panier)); //on sauve un Panier txt
}
//LS:CETTE FONCTION EXTRAIT LE PANIER ACTUEL DU LOCALSTORAGE ET LE TRANSFORME EN TABLEAU D OBJET JSON
export function recupPanier() {
  let panierRecup = localStorage.getItem("monPanier"); //panierRecup est un txt
  if (panierRecup == null) {
    return [];
  } else {
    return JSON.parse(panierRecup); //on recupere un tableau d'objet JSon ex: indice 0 {Id,qte,couleur}
  }
}
```

```
//LS:CETTE FONCTION SUPPRIME UNE LIGNE ID/QUANTITE/COULEUR DU LOCALSTORAGE
function supprimeLigne(idQteColor) {
  let Panier = recupPanier();
  let indexPanier = Panier.findIndex((iqc) => iqc === idQteColor);

  console.log(indexPanier);//index négatif??
  Panier.splice(indexPanier, 1);
  console.log(Panier);
  savePanier(Panier);
}
```

```
//LS:CETTE FONCTION AJOUTE L ARTICLE CLICKER DANS LE LS, SAUF S IL EXISTE DEJA, DANS CE CAS LA FONCTION MET A JOUR SA QUANTITE
export function ajoutAuPanier({ id, qte, color }) {
  // 1 arg objet  avec 3 propriété
  // console.log(id);//ok
  let PanierA = recupPanier(); //on recupère le tableau  d'objets du panier [clé:0, valeur:{id:"id",qte:"quantité",color:"couleur"}]
  console.log(PanierA);
  let memeProduit = PanierA.find((p) => p.id == id && p.color == color);
  if (memeProduit != undefined) {
    //on ajoute la quantite en arg de ajoutAuPanier à la qte initiale
    memeProduit.qte = parseInt(memeProduit.qte, 10) + parseInt(qte, 10);
    //parseInt(string,10) transforme une string en nombre en base 10
  } else {
    PanierA.push({ id, qte, color });
  }
  savePanier(PanierA);
}
```

Page panier: Fonctions implémentées

Afficher les articles listés dans le LS dans la page panier en respectant les consignes sur la couleur

Calculer et afficher la quantité totale et le prix total

```
      function affichePanier() {
  let Panier = recupPanier();
  let prixTotal = 0;
  let qteTotale = 0;
  for (let idQteColor of Panier) {
    //pour chaque article du panier:
    //on recupere chaque objet à afficher au panier dans l'api à partir de son id:
    let myPromise = fetch(
      "http://localhost:3000/api/products/" + idQteColor.id
    ); //le résultat de la promesse contient l'article voulu au format ?
    myPromise
      .then(function (result) {
        if (result) {
          return result.json(); //on convertit l'objet au format Json
        }
      })
      .then((ProductJson) => {
        //------------------------------------------------------------CREATION D UN ARTICLE DOM (AFFICHAGE)
        let article = document.createElement("article");
        article.setAttribute("class", "cart__item");
        article.setAttribute("data-id", idQteColor.id);
        article.setAttribute("data-color", idQteColor.color);
        article.innerHTML = ` <div class="cart__item__img">
       <img src=${ProductJson.imageUrl} alt=${ProductJson.altTxt}>
       </div>
       <div class="cart__item__content">
         <div class="cart__item__content__description">
           <h2>${ProductJson.name}</h2>
           <p>${idQteColor.color}</p>
           <p>${ProductJson.price} euros</p>
         </div>
         <div class="cart__item__content__settings">
           <div class="cart__item__content__settings__quantity">
             <p>Qté : </p>
             <input type="number" class="itemQuantity" name="itemQuantity" min="1" max="100"
value="${idQteColor.qte}">
           </div>
           <div class="cart__item__content__settings__delete">
             <p class="deleteItem">Supprimer</p>
           </div>
         </div>
       </div> `;
        document.querySelector("#cart__items").appendChild(article);
```

```
//-----------CALCUL QUANTITE TOTALE ET PRIX TOTAL
qteTotale += parseInt(idQteColor.qte, 10);

document.querySelector("#totalQuantity").innerText = qteTotale;

prixTotal += ProductJson.price * parseInt(idQteColor.qte, 10);
document.querySelector("#totalPrice").innerText = prixTotal;
```

Page panier: Fonctions implémentées

Ecouter le click sur le bouton supprimer et dans ce cas supprimer la ligne de l'article et mettre à jour quantité et prix total sans recharger la page

Ecouter le choix de quantité et en cas de changement mettre à jour la quantité totale et le prix total sans recharger la page

```
let supprime = article.querySelector(".deleteItem");
supprime.addEventListener(
  "click",
  () => {
    //suppression dans le DOM:
    article.remove();
    qteTotale -= parseInt(idQteColor.qte, 10);
    document.querySelector("#totalQuantity").innerText = qteTotale;
    prixTotal -= ProductJson.price * parseInt(idQteColor.qte, 10);
    document.querySelector("#totalPrice").innerText = prixTotal;
    //suppression dans le LS:
    supprimeLigne(idQteColor);
  },
  false
);
```

```
//LS:CETTE FONCTION TROUVE LA LIGNE A MODIFIER DU LS ET REMPLACE SA QUANTITE PAR LA NOUVELLE
function modifierQtePanier(idQteColor, newQte) {
  let Panier = recupPanier();
  let articleQuiChange = Panier.find(
    (iqc) => iqc.id == idQteColor.id && iqc.color == idQteColor.color
  );
  let vieilleQte=articleQuiChange.qte;
  articleQuiChange.qte = newQte;
  let prixTotal=document.querySelector("#totalPrice");
  // console.log("selecteur prix total: ",prixTotal);…
  savePanier(Panier);
  // console.log(Panier);
  let qteTotale=document.querySelector("#totalQuantity");
  // console.log("qte totale innertext avant calcul: ",qteTotale.innerText);
  qteTotale.innerText= parseInt(qteTotale.innerText) - parseInt(vieilleQte) + parseInt(newQte);
  console.log("qte totale innertext: ",qteTotale.innerText);
  // console.log(document.querySelector(".cart__item__content__description p"));…
  let prixUnitaire=parseInt(document.querySelector(".cart__item__content__description p").nextElementSibling.innerText);
  prixTotal.innerText=parseInt(prixTotal.innerText)-(parseInt(vieilleQte)*prixUnitaire)+(parseInt(newQte)*prixUnitaire);
  // console.log("prix total: ",prixTotal.innerText);
}
```

Page panier: au click sur "Commander"

Ecouter le click sur le bouton commander et dans ce cas analyser les données saisies par le client dans le formulaire de contact

Si les données sont invalides, alerter et guider le client pour qu'il corrige

```
      function validation(champ, saisie) {
  let valid; //deviendra faux si un champ est faux
  if (champ.id == "email") {
    let regExMail = /^[a-z0-9.-_]+[@]{1}[a-z0-9.-_]+[.]{1}[a-z]{2,10}$/i;
    valid = regExMail.test(saisie);
    // console.log(champ.name + " est de type email");
  } else if (champ.id == "address") {
    let regExAdresse = /^[0-9a-z][0-9a-z,.-]+/i;
    valid = regExAdresse.test(saisie);
  } else {
    let regExTxt = /^[a-z][a-z]+/i;
    valid = regExTxt.test(saisie);
  }

  //----------------------------GESTION MESSAGES D'ERREUR-------------------------------------------------
  let messageErreur = champ.nextElementSibling; //il s'agit du paragraphe suivant chanque champ
  // console.log(champ.nextElementSibling);
  if (valid == false) {
    messageErreur.innerHTML = `Saisie invalide ou manquante.`;
    if (champ.type == "email") {
      messageErreur.innerHTML += `Le champ Email doit être de type: caractères@caractères.caractères`;
    } else if (champ.id == "address") {
      messageErreur.innerHTML += `Le champ Adresse ne doit contenir que des lettres, chiffres, ou les catactères ",
- ."`;
    } else {
      messageErreur.innerHTML += `Les champs Prénom Nom et Ville ne doivent contenir que des lettres.`;
    }
    console.log(saisie + " n'est pas valide");

    return valid; //on renvoie valid en reponse de la fonction validation pour mettre à jour la variable validForm
en début de code
  } else {
    messageErreur.innerHTML = "";
    console.log(saisie + " est valide");
    return valid;
  }
}
```

Page panier: au click sur "Commander"

Si les données du formulaire sont valides, requêter l'API pour obtenir un numéro de commande

Aller à la page "URLconfirm+NDC"

Vider les paniers

```
 if (validForm) {
//on cree un objet JSON vide:
let monContact = {};
//on le remplit avec les couples nom de champ/valeur saisie:
champs.forEach((champ) => {
  let nomChamp = champ.name;
  let valeurSaisie = champ.value;
  monContact[nomChamp] = valeurSaisie;
});

//----CREATION D'UN OBJET JSON COMPOSE DE L'OBJET CONTACT ET DU TABLEAU D'ID COMME ATTENDU PAR L'API
let objetAPI = {
  contact: monContact,
  products: listIDpanier,
};
console.log(objetAPI);
//------------REQUETE API=> obtention n° de commande et redirection page confirmation
recupNcommande(objetAPI).then((reponseJson) => {
  let numDeCommande = reponseJson.orderId;
  console.log(reponseJson);
  //on utilise le format d'URL vu pour l'affichage d'une page produit "URL?clé=valeur":
  window.location = "confirmation.html?NDC=" + numDeCommande;
  console.log(numDeCommande);//ok
  //----------------------------on vide le panier LS et le panier affiché:
  localStorage.clear();
  document.querySelector("#cart__items").innerHTML = "";
})
```

Page panier, au click sur "Commander": Fonction "recupNcommande" contenant la requête

```
function recupNcommande(objetAPI) {
  return fetch("http://localhost:3000/api/products/order", {
    method: "POST",
    headers: {
      Accept: "application/json",
      "Content-Type": "application/json",
    },
    body: JSON.stringify(objetAPI),
  })
    .then(function (result) {
      if (result.ok) {
        console.log(result.ok);
        return result.json();
      }
      throw Error("Erreur de l'API");
    })

    .catch(function (erreur) {
      alert(erreur.message);
    });
}
```

Page confirmation: Fonction

Afficher le numéro de commande obtenu à l'étape précédente

Plan de tests d'acceptation

	Fonctionnalité	Action	Résultat attendu	Résultat observé
	PAGE ACCEUIL			
1	Une page d'accueil montrant (de manière dynamique) tous les articles disponibles à la vente.	Ouvrir sur la page d'accueil du site web dans un navigateur	Affichage de l'ensemble des produits	OK
2	Au clic sur un produit l'utilisateur est redirigé sur la page du produit	Cliquer sur un produit de la page acceuil	Affichage de la page produit.	OK
	PAGE PRODUIT			
3	Gestion d'une erreur d'URL suite à une modification de l'utilisateur	Modifier l'URL de l'API de sorte que l'id ne figure pas dedans	Affiche une alerte et redirige vers la page d'acceuil	OK
4	Prise en compte de la couleur et de la quantité choisies dans le panier	Choisir une couleur et une quantité et cliquer sur « Ajouter au panier »	La page panier s'affiche avec l'article ajouté assorti de la couleur et de la quantité choisies	OK
5	Il est interdit de saisir autre chose qu'un nombre entre 0 et 100 dans la quantité	Entrer un nombre négatif ou des lettres dans la quantité	Un message d'alerte s'affiche, l'exécution s'arrête	OK
6	Il est obligatoire de choisir une couleur avant de commander	Commander sans choisir de couleur	Un message d'alerte s'affiche, l'exécution s'arrête	OK
	PAGE PANIER			
7	Si on ajoute un article de même couleur et de même id qu'un article déjà dans le panier, il n'apparaît qu'une fois avec la quantité ajustée	Ajout d'un article de même couleur et de même id qu'un article déjà dans le panier	L'article n'apparaît qu'une fois avec la quantité ajustée	OK
8	Si on ajoute un article de même id mais d'une autre couleur, il apparaît sur 2 lignes distinctes	Ajout d'un article de même id mais d'une autre couleur	L'article apparaît sur 2 lignes distinctes	OK
9	Modification de la quantité avec prise en compte dans le total	Modifier la quantité d'un article du panier	La quantité totale d'article du panier se met à jour	OK
10	Suppression d'une ligne d'article au click sur le bouton Supprimer	Cliquer sur le bouton « Supprimer » d'un article	La ligne de l'article se supprime	OK
11	Impossibilité de commander si le panier est vide	Vider le panier et cliquer sur « Commander »	Affichage d'un message d'erreur et on reste sur la page panier	OK
12	Validation formulaire champs texte : pas de chiffre (sauf adresse)	Entrer des chiffres dans les champs Prénom, Nom et Ville	Affichage d'un message d'erreur sous chaque champ mal rempli avec consigne d'aide	OK
13	Validation formulaire champs email: doit contenir le @ et un .	Entrer une saisie sans @ et/ou sans . dans le champ email	Affichage d'un message d'erreur sous chaque champ mal rempli avec consigne d'aide	OK
14	Il est interdit de saisir autre chose qu'un nombre entre 0 et 100 dans la quantité	Entrer un nombre négatif ou des lettres dans la quantité	Un message d'alerte s'affiche, l'exécution s'arrête	OK
	PAGE CONFIRMATION			
15	Affichage du numéro de commande	Dans la page panier contenant au moins un article, remplir le formulaire correctement et cliquer sur « Commander »	Affichage du numéro de commande	OK
16	Bloquer l'accès à la page s'il n'existe pas de numéro de commande	Taper cette URL dans le navigateur : http://127.0.0.1:5500/front/html/confirmation.html ?	Message d'erreur puis rédirection vers la page d'acceuil	OK

Plan de tests: Fonctions supplémentaires implémentées pour prévenir un mauvais usage par le client

Page produit

Manipulation de l'URL de la page

```
if (searchId.has("id")) {
  let canapId = searchId.get("id");
  //La méthode get("txt") appliquée à un objet de type URLSearc
  console.log(canapId); //ok

  //------------------Gestion d'une mauvaise URL: vérification
  //-----------------on récupère la liste des id dans l'API:
  let listIDpanier = [];
  let Promise = fetch("http://localhost:3000/api/products");
  Promise.then(function (result) {···
  }).then((listProductJson) => {
    listIDpanier = listProductJson.map((elementPanier) => {···
    });
    console.log(listIDpanier);
    let idExist = listIDpanier.find((p) => p == canapId);
    //------------si l'id dans l'URL existe dans l'API, on trai
    if (idExist != undefined) {       You, la semaine dernière
    }
    //si l'id n'existe pas:
    else {
      alert("Un problème est survenu, cliquez sur ok.");
      window.location = "index.html";
    }
  });
}
```

Plan de tests: Fonctions supplémentaires implémentées pour prévenir un mauvais usage par le client

Page produit

Saisie invalide de quantité: lettres ou hors 1-100

Oubli du choix de la couleur

```
let quantity = document.getElementById("quantity").value;

if (quantity <= 0 || quantity > 100) {
  alert("La quantité doit être un nombre entre 1 et 100.");
  console.log(quantity);
} else {
  //récup de la couleur DOM si elle est choisie:

  let color = document.getElementById("colors").value;
  if (color != "") {
    //Ajout de l'objet {id,qte,color} au LocalStorage via la
    let monAjout = { id: canapId, qte: quantity, color }; //
    ajoutAuPanier(monAjout);
    window.location = "cart.html";
  } else {
    alert("Sélectionner une couleur avant de commander");
  }
}
```

Plan de tests: Fonctions supplémentaires implémentées pour prévenir un mauvais usage par le client

Page panier

Gestion en cas de panier vide ou de formulaire invalide

```
if (recupPanier().length !== 0) {
  let listIDpanier = recupPanier().map((elementPanier) => {
    return elementPanier.id;
  });
  // console.log(listIDpanier);//ok
  //---------------------------------------RECUPERATION DU FOR
  if (validForm) {       You, il y a 2 semaines • gestion alert
  } else {
    alert("Le formulaire est à compléter et/ou à corriger");
  }
} else {
  alert("Le panier est vide");
}
```

Gestion en cas de saisie invalide de quantité: lettres ou hors 1-100

```
let quantite = article.querySelector(".itemQuantity");
quantite.addEventListener(
  "change",
  (Event) => {
    if (Event.target.value >= 1 && Event.target.value < 101) {
      modifierQtePanier(idQteColor, Event.target.value);       Yo
    }
    else{
      alert("La quantité doit être un nombre entre 1 et 100.");
    }
  },
  false
);
```

Plan de tests: Fonctions supplémentaires pour prévenir un mauvais usage par le client

Page confirmation

Gestion en cas de manipulation de l'URL de la page

Merci pour votre attention !