Projet 5

Orinoco

Présentation des Livrables

- Création du repo Github contenant:
  - un dossier Back-End (fourni),
  - un dossier Front-End incluant :
    - un dossier ‘JS’ comprenant mes fichiers javascript,
    - un dossier ‘Sass’ comprenant mes fichiers sass,
    - un dossier ‘Pages’ comprenant mes pages html,
    - un dossier ‘images’ comprenant le logo et le favicon,
    - un dossier ‘CSS’ comprenant mon style.css compiler et prefixer.

- Le site hébergé sur Github page => Orinoco

- Le plan de test

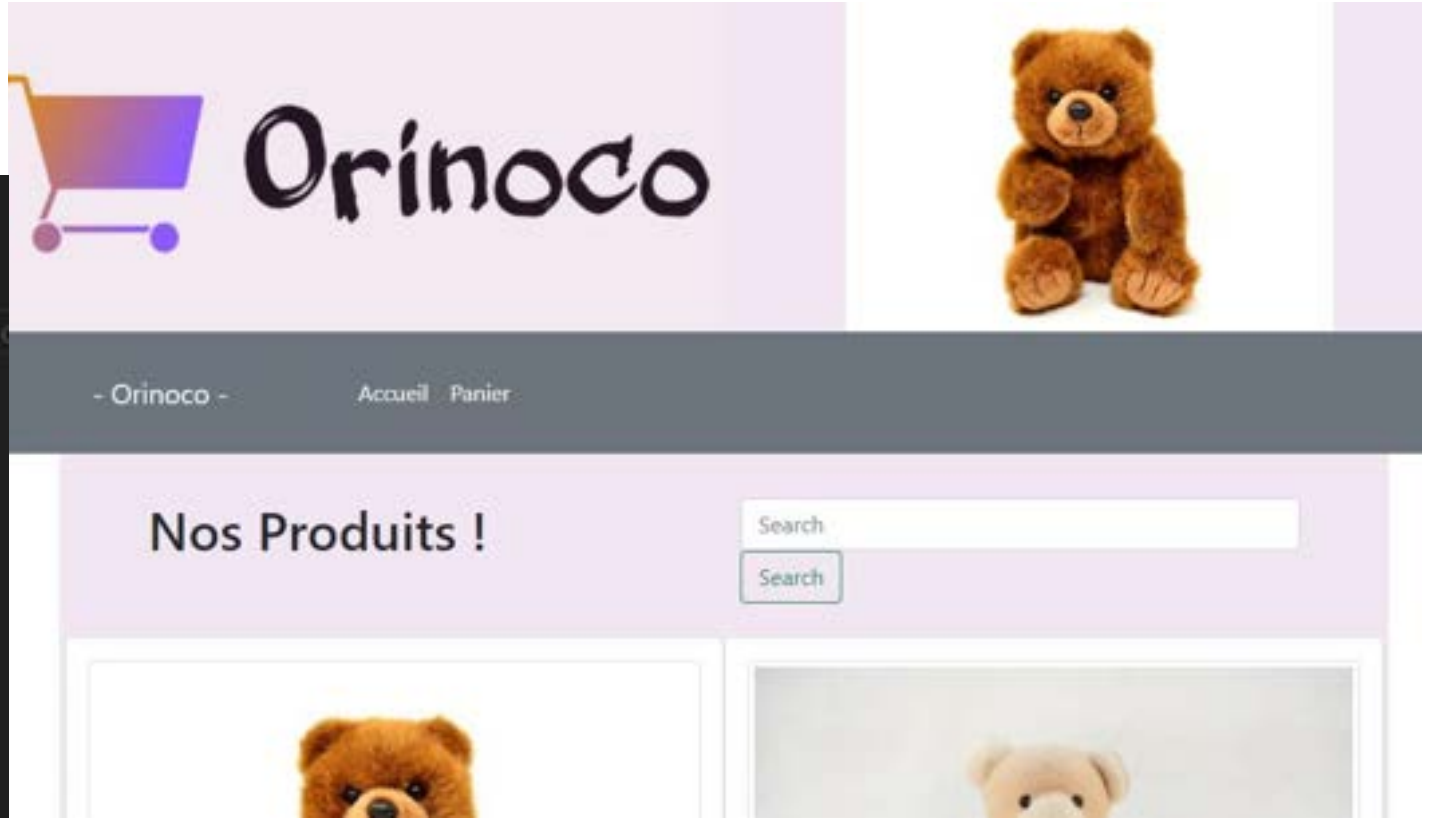
Passage validateur HTML, CSS et SEO

- Validateur HTML
- Validateur CSS
- SiteChecker SEO

Le code - Arrivée en page d’accueil (1/2)

- Appel de l’objet avec une promesse asynchrone, accepté ou refusé par l’API. Fetch utilise par défaut la requête GET et me renvoi la réponse en format JSON dans mon .then (.catch si un problème est survenu)

```
const adresse = fetch("http://localhost:3000/api/teddies")
  .then((res) => {
    if (res.ok) {
      return res.json();
    }
  })
  .then((value) => {
    // console.table(value);
    addCarte(value);
  })
  .catch((err) => {
    console.log(err);
    // Une erreur est survenue
  });
```



Le code - Arrivée en page d'accueil (2/2)

- La fonction addCarte et le résultat de la demande à l'API. Je met en forme en HTML les données « value » en utilisant ma class FicheOurson et les inserts avec innerHTML pour mes cartes et mon carrousel.

```
class FicheOurson {
  constructor(_id, name, price, description, imageUrl, colors, quantity) {
    this._id = _id;
    this.name = name;
    this.price = +price;
    this.description = description;
    this.imageUrl = imageUrl;
    this.colors = colors;
    this.quantity = +quantity;
  }
}
```

```
function addCarte(value) {
  // incrementation pour ma class active de mon carousel
  let incrementation = 0;
  //boucle pour chaque iteration d'un produit
  for (FicheOurson of value) {
    //recupère l'élément Ourson dans le HTML pour les cartes
    const carte = document.getElementById("Ourson");      You, 3 weeks ago
    carte.innerHTML +=
    `<div class="col-sm-6 p-3 mb-5 card">
      <a href="./Front-End/Pages/produit.html?_id=${FicheOurson._id}" class
        <img src="${FicheOurson.imageUrl}" class="img-fluid img-thumbnail p
      </a>
      <div class="row card-body justify-content-around">
        <h2 class="col-6 card-title text-center">${FicheOurson.name}</h3>
      </div>
      <div class="card-body">
        <p class="card-text text-truncate">${FicheOurson.description}</p>
      </div>
      <a href="./Front-End/Pages/produit.html?_id=${FicheOurson._id}" class
    </div>`;
    //recupère l'élément carouselOurson dans le header pour créer le carous
    const carouselOurson = document.getElementById("carouselOurson");
    if (incrementation == 0) {
      carouselOurson.innerHTML +=
      `<div class="carousel-item active">
        <img src="${FicheOurson.imageUrl}" alt="ourson en pelluche : ${FicheOurson.name}"
      </div>`;
    } else {
      carouselOurson.innerHTML +=
      `<div class="carousel-item">
        <img src="${FicheOurson.imageUrl}" alt="ourson en pelluche : ${FicheOurson.name}"
      </div>`;
    }
    incrementation++;
  }
```

Le code - Arrivée en page produit (1/3)

- Récupération de l’id de l’article vers lequel nous somme redirigé puis nous modifions l’adresse d’appel de l’API.

- J’insère ensuite mes données de la même manière avec innerHTML

```
//récupérer l'id dans l'url des objets
const parsedUrl = new URL(window.location.href);
const oursonCible = parsedUrl.searchParams.get("_id");
// console.log(oursonCible);

//modification de l'adresse d'appel à l'API
const newUrl = `http://localhost:3000/api/teddies/${oursonCible}`
// console.log(newUrl);
```

```
fetch(newUrl)
  .then(function (res) {
    if (res.ok) {
      return res.json();
    }
  })
  .then((ourson) => {
    carteId(ourson);
    function carteId(ficheOurson) {
      //recupère l'élément Ourson dans le HTML pour les cartes
      const carteProduit = document.getElementById("oursonProduit");
      carteProduit.innerHTML = `
          <div class="col-sm-6 p-3 m-5 mb-5 card text-center">
              <img src="${
                ficheOurson.imageUrl
              }" class="img-fluid img-thumbnail p-1 mx-auto" alt="image repr
    ficheOurson.name
    }"></img>
              <div class="row card-body justify-content-between text-center"
                  <h2 class="col-6 card-title">${ficheOurson.name}</h3>
                  <h3 class="col-6">${prixConverti(ficheOurson.price)}</h3>
              </div>
```

Le code - Arrivée en page produit (2/3)

- Je crée ensuite `<select id= »checkBoxColor » class= »form-select » aria-label= »Choix couleur »></select>` dans lequel je glisse mes options de couleurs avec une boucle « for »

```
  choixColor(ficheOurson);
  ajouterPanier(ficheOurson);
}

function choixColor(ficheOurson) {        You, 4 days ago • Mise
  // checkBox des couleurs
  const checkProduit = document.getElementById("checkBoxColor");
  for (colors of ficheOurson.colors) {
    checkProduit.innerHTML += `
          <option value="${colors}">${colors}</option>`;
  }
}
```

Le code - Arrivée en page produit (3/3)

- Compilation des données du produit avant le clique grâce à (e) => { e.preventDefault().
- Création d'un nouveau produit que l'ont compare avec les produits contenu ou non dans le localStorage en prenant en compte la couleur choisie.
- Si le produit est déjà présent, j'incrémente seulement la quantité, sinon j'ajoute le produit au panier.
- un message d'alert averti l'utilisateur de l'ajout au panier.

```
function ajouterPanier(ficheOurson) {
  //ajout des produits au panier
  const ajoutPanier = document.getElementById("ajoutPanier");
  ajoutPanier.addEventListener("click", (e) => {
    e.preventDefault();
    const list = document.getElementById("checkBoxColor");
    const quantity = document.getElementById("quantity");
    // créer un nouveau produit
    let produitOurson = new Produit(
      newUrl,
      ficheOurson._id,
      ficheOurson.name,
      ficheOurson.price,
      ficheOurson.description,
      ficheOurson.imageUrl,
      list.value,
      quantity.value
    );
```

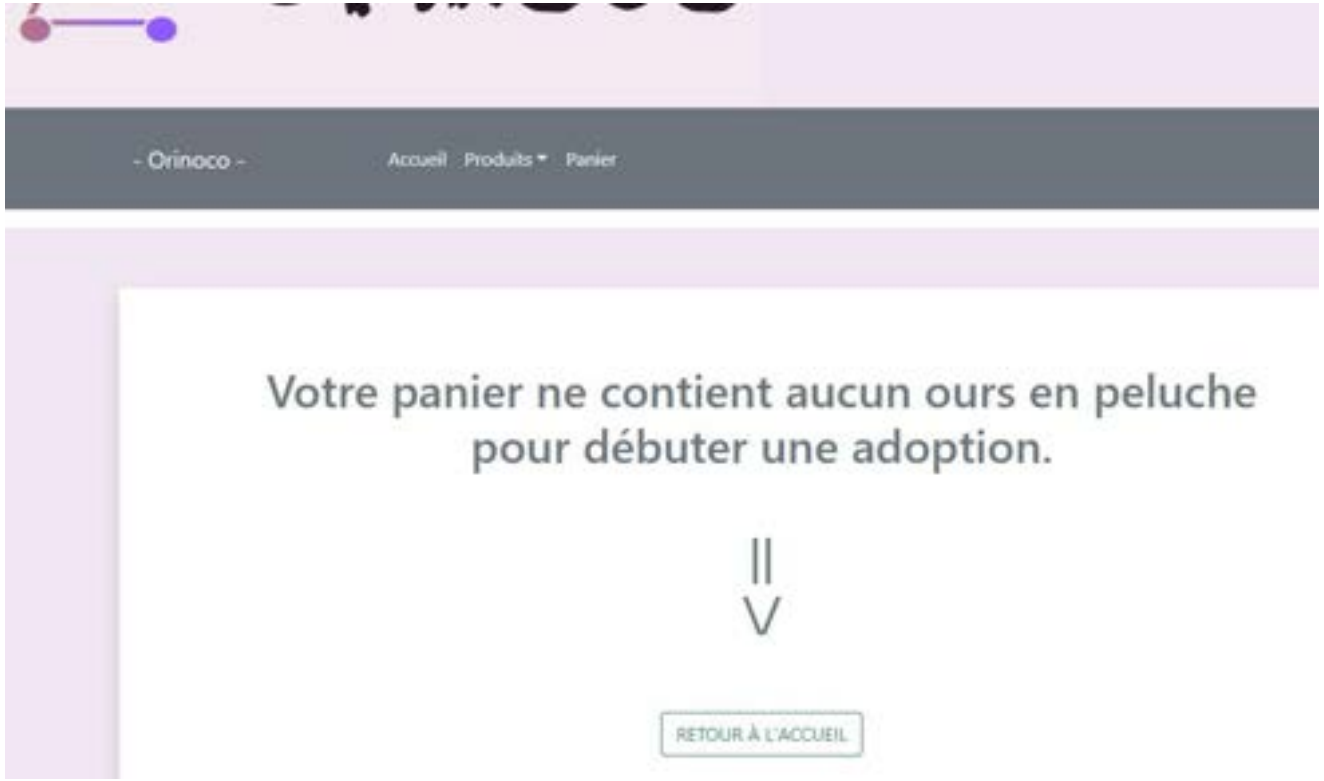
```
// vérifie si le produit est déja présent dans le localstorage et s
let presenceLocalStorage = false;
let indexModification;
for (products of panier) {
  switch (products.checkBoxColor) {
    case produitOurson.checkBoxColor:
      presenceLocalStorage = true;
      indexModification = panier.indexOf(products);
  }
}

// si déja Present incrémente seulement la quantité
if (presenceLocalStorage) {
  panier[indexModification].quantity =
    +panier[indexModification].quantity + +produitOurson.quantity;
  localStorage.setItem("oursonCommande", JSON.stringify(panier));
  // si absent, ajoute le produit au localStorage
} else {
  panier.push(produitOurson);
  localStorage.setItem("oursonCommande", JSON.stringify(panier));
  alert("L'article et ajouté au panier");
}
location.reload();
});
```

Le code - Arrivée en page panier (1/8)

- Création d’un panier vide

- Dissimulation du formulaire et mise en avant de « panierVide » représenter par notre carte d’information destinée à l’utilisateur.

```
// indique que le panier est vide
if (panier.length < 1) {
  formulaire.classList.add("d-none");
```



Le code - Arrivée en page panier (2/8)

- Pour tous produit présent dans le localStorage « oursonCommande », l'ajoute sous forme de tableau sur la page panier et dissimule la carte « panierVide ».

```
} else {
  formulaire.classList.add("d-none");
  panierVide.classList.add("d-none");
  const fullpanier = document.getElementById("panier");
  fullpanier.classList.toggle("d-none");
  for (product of panier) {
    listeCommande(product);
  }
```

```
//ajoute le tableau de commande
function listeCommande(product) {
  const indexProduit = panier.indexOf(product);
  const listeProduit = document.getElementById("listePanier");
  listeProduit.innerHTML += `
      <tr class="text-center bg-light">
          <td>
              <img src="${
                product.imageUrl
              }" class="img-fluid img-thumbnail" alt="${product.name}">
          </td>
          <td class="align-middle">
              <span>${product.name}</span>
          </td>
          <td class="align-middle">
              <span>${product.checkBoxColor}</span>
          </td>
          <td class="align-middle">
              <button aria-label="Bouton moins" type="button" class="rou
              <span class="mx-0 mx-lg-3"> ${product.quantity}</span>
              <button aria-label="Bouton plus" type="button" class="rour
          </td>
          <td class="align-middle">
              <span>${prixConverti(product.price)}</span>
          </td>
```

Le code - Arrivée en page panier (3/8)

- Un bouton d'ajout de quantité à été ajouté afin que l'utilisateur puisse en faire la modification à sa convenance.

- L'ajout se fait par le clic utilisateur sur le bouton + en incrémentant de 1 la quantité de l'objet défini dans le localStorage « oursonCommande » et actualise la page.

```
// ajouter produit
function addProduct(event) {
  const index = event.target.getAttribute("data-index");
  panier[index].quantity++;
  localStorage.setItem("oursonCommande", JSON.stringify(panier));
  location.reload();
}
const buttonAdd = document.getElementsByClassName("plus");
for (add of buttonAdd) {
  add.addEventListener("click", addProduct);
}
```

```
<td class="align-middle">
    <button aria-label="Bouton moins" type="button" class="rounded minus data-toggle="modal"
    data-target="#exampleModal" data-index="${indexProduit}"><span class="fas
    fa-minus-square text-danger" data-index="${indexProduit}"></span></button>
    <span class="mx-0 mx-lg-3"> ${product.quantity}</span>
    <button aria-label="Bouton plus" type="button" class="rounded plus" data-toggle="modal"
    data-target="#exampleModal" data-index="${indexProduit}"><span class="fas fa-plus-square
    text-success" data-index="${indexProduit}"></span></button>
</td>
```

Le code - Arrivée en page panier (4/8)

- Un bouton de suppression de quantité à été ajouté afin que l'utilisateur puisse en faire la modification à sa convenance.

- Le retrait se fait par le clic utilisateur sur le bouton - en décrémentant de 1 la quantité de l'objet défini dans le localStorage « oursonCommande » et actualise la page.

```
//supprimer un produit
function minusProduct(event) {
  const index = event.target.getAttribute("data-index");
  if (panier[index].quantity > 1) {
    panier[index].quantity--;
  } else {
    panier.splice(index, 1);
  }
  localStorage.setItem("oursonCommande", JSON.stringify(panier));
  location.reload();
}
const buttonMinus = document.getElementsByClassName("minus");
for (minus of buttonMinus) {
  minus.addEventListener("click", minusProduct);
}
```

```
<td class="align-middle">
    <button aria-label="Bouton moins" type="button" class="rounded minus data-toggle="modal"
    data-target="#exampleModal" data-index="${indexProduit}"><span class="fas
    fa-minus-square text-danger" data-index="${indexProduit}"></span></button>
    <span class="mx-0 mx-lg-3"> ${product.quantity}</span>
    <button aria-label="Bouton plus" type="button" class="rounded plus" data-toggle="modal"
    data-target="#exampleModal" data-index="${indexProduit}"><span class="fas fa-plus-square
    text-success" data-index="${indexProduit}"></span></button>
</td>
```

Le code - Arrivée en page panier (5/8)

- un clic sur le bouton « vider le panier » efface le localStorage.

```
function boutonEffacePanier() {
  const buttonclearPanier = document.getElementById("effacerPanier");
  buttonclearPanier.addEventListener("click", () => {
    effacerPanier();
    location.reload();
  });
}
```

```
// supprimer le Panier
function effacerPanier() {
  localStorage.clear();
}
```

- Une fois que le contenu du panier nous convient, le clic sur « valider le panier » entraîne la disparition des boutons et l’affichage du formulaire.

```
function affichageBouton() {
  //affiche le formulaire et cache les boutons valider/supprimer panier
  const validationpanier = document.getElementById("validationpanier");
  const validationButton = document.getElementById("validationButton");
  validationpanier.addEventListener("click", () => {
    formulaire.classList.toggle("d-none");
    validationButton.classList.add("d-none");
  });
}
```

Le code - Arrivée en page panier (6/8)

- Les données du formulaire comportent des restrictions que l'utilisateur doit respecter pour valider la commande, elles sont toutes requises et indique à l'utilisateur si une erreur est présente.

- Toutes les valeurs sont ensuite stockée dans la variable contact.

```
function validationFormulairePOST() {
  const commande = document.getElementById("commande");
  const regexName = /^(([a-zA-ZÀ-ÿ]+[\s\-]{1}[a-zA-ZÀ-ÿ]+)|([a-zA-ZÀ-ÿ]+))$/;
  const regexCity =
    /^(([a-zA-ZÀ-ÿ]+[\s\-]{1}[a-zA-ZÀ-ÿ]+)|([a-zA-ZÀ-ÿ]+)){1,10}$/;
  const regexMail = /^[a-zA-Z0-9._-]+@[a-zA-Z0-9._-]{2,}\.[a-z]{2,4}$/;
  const regexAddress = /^(([a-zA-ZÀ-ÿ0-9]+[\s\-]{1}[a-zA-ZÀ-ÿ0-9]+)){1,10}$/;

  commande.addEventListener("click", (event) => {
    // Récupération des infos pour l'envoie en POST
    let contact = {
      firstName: document.getElementById("firstName").value,
      lastName: document.getElementById("lastName").value,
      address: document.getElementById("address").value,
      city: document.getElementById("city").value,
      email: document.getElementById("email").value,
    };
    // Validation que le formulaire est correctement rempli
    if (
      (regexName.test(contact.firstName) == true) &
      (regexName.test(contact.lastName) == true) &
      (regexAddress.test(contact.address) == true) &
      (regexCity.test(contact.city) == true) &
      (regexMail.test(contact.email) == true)
    ) {
      event.preventDefault();
```

Le code - Arrivée en page panier (7/8)

- Récupération de la date, du mois, de l'heure et minutes puis stockage de celle ci dans le localStorage « date »

```
const todayDate = new Date();
let nowadays = todayDate.getDate();
let month = todayDate.getMonth() + 1;
let todayHours = todayDate.getHours();
let todayMinutes = todayDate.getMinutes();
if (nowadays < 10) {
  nowadays = "0" + nowadays;
}
if (month < 10) {
  month = "0" + month;
}
if (todayHours < 10) {
  todayHours = "0" + todayHours;
}
if (todayMinutes < 10) {
  todayMinutes = "0" + todayMinutes;
}
const date = nowadays + "-" + month + "-" + todayDate.getFullYear();
const hours = todayHours + ":" + todayMinutes;
const fullDate = { date, hours };
const infoCommande = JSON.parse(localStorage.getItem("date")) || [];
infoCommande.push(fullDate);
localStorage.setItem("date", JSON.stringify(infoCommande));
```

Le code - Arrivée en page panier (8/8)

- Je transforme les informations reçu de contact et products en JSON. (stringify permet à l'api de lire les données)

- Je crée ensuite une variable pour les paramètres de fetch avec la méthode POST.

- Je demande à l'API de me renvoyer un numéro de commande et de me rediriger vers la page commande.html.

- En cas d'erreur un message s'affiche dans l'outil d'inspection.

```
// Transformation en JSON des infos reçu de contact et products
let infoFinalCommande = JSON.stringify({ contact, products });

// Paramétres pour la requéte fetch
const recapitulatif = {
  method: "POST",
  headers: {
    Accept: "application/json",
    "Content-Type": "application/json",
  },
  body: infoFinalCommande,
};
// console.table(recapitulatif);

// Mise à jour du order pour la page de commande.html
fetch("http://localhost:3000/api/teddies/order", recapitulatif)
  .then((response) => response.json())
  .then((ourson) => {
    localStorage.setItem("order", JSON.stringify(ourson));
    // console.table(ourson);
    document.location.href = "commande.html";
  })
  .catch((erreur) => console.log("erreur : " + erreur));
else {
alert("le formulaire n'est pas correctement rempli.");
```

Le code - Arrivée en page commande (1/3)

- Je met en forme en HTML les données que je récupère dans le localStorage « order » et le localStorage « date ».

```
const commande = JSON.parse(localStorage.getItem("order")) || [];
const informations = document.getElementById("informations");
informations.innerHTML += `
  <div class="col-sm-12 p-3 mb-5 card text-center">
      <p class="fs-5 card-text">Merci d'avoir choisi notre site pour votre
      <p class="fs-5"> Votre commande passée le <span class="fw-bold">${
        date[0].date
      }</span> à <span class="fw-bold">${
  date[0].hours
}</span> d'un montant total de <span class="fw-bold">${prixConverti(
  prixTotalPanier()
)}</span> a été validée.</p>
      <p class="fs-5">Référence de la commande => <span class="fw-bold">${
        commande.orderId
      }</span>.</p>
  </div>
  <div class="col-sm-12 p-3 mb-5 card text-center">
    <p class="fs-5">Votre commande sera envoyée à l'adresse suivante :
    <div class=" fs-5 text-center fw-bold">
        <p class="text-capitalize">${commande.contact.firstName} ${
  commande.contact.lastName
}</p>
        <p class="text-capitalize">${commande.contact.address}</p>
        <p class="text-capitalize">${commande.contact.city}</p>
    </div>
    <p class="fs-5">Un exemplaire de votre facture vous est transmise par
      commande.contact.email
    }</span>.</p>
    </div>
```

Le code - Arrivée en page commande (2/3)

- Le prix Total est calculé en fonction des produits et de la quantité de ceux-ci présent dans le localStorage « oursonCommande ».

```
//affiche le prix total
prixTotal();
```

```
// conversion des prix
function prixConverti(prixProduit) {
  let prix = `${prixProduit}`;
  prix = Intl.NumberFormat("fr-FR", {
    style: "currency",
    currency: "EUR",
    minimumFractionDigits: 2,
  }).format(prix / 100);
  return prix;
}

// calcul du total
function prixTotalPanier() {
  let totalpanier = 0;
  panier.forEach((produitOursonCommande) => {
    totalpanier =
      totalpanier +
      produitOursonCommande.price * produitOursonCommande.quantity;
  });
  return totalpanier;
}

//affiche le total du Panier
function prixTotal() {
  const prixTotal = document.getElementById("prixTotal");
  prixTotal.innerHTML += `${prixConverti(prixTotalPanier())}`;
}
```

Le code - Arrivée en page commande (3/3)

- Affiche le tableau de chaque produit présent dans le localStorage « oursonCommande » et retire les boutons d’ajout suppression de quantité.

- Le bouton imprimer permet d’afficher dans une nouvelle fenêtre l’impression de la page commande.

- Le clic sur le bouton « panier » ou « acceuil » vide le localStorage.

```
// affiche Récapitulatif de ma commande
for (product of panier) {
  listeCommande(product);
}

//cache les boutons d'ajout suppression de quantité
const deletedItem = document.getElementsByClassName("rounded");
for (element of deletedItem) {
  element.classList.add("d-none");
}
```

```
//bouton imprimer
const imprimer = document.getElementById("imprimer");
imprimer.addEventListener("click", (e) => {
  e.preventDefault;
  window.print();
});

//vide le localStorage
const clickHome = document.getElementById("accueil");
clickHome.addEventListener("click", () => {
  effacerPanier();
});
const clickPanier = document.getElementById("apercuPanier");
clickPanier.addEventListener("click", () => {
  effacerPanier();
});
```