

## Objectifs

- Utiliser JQuery

## Remise:

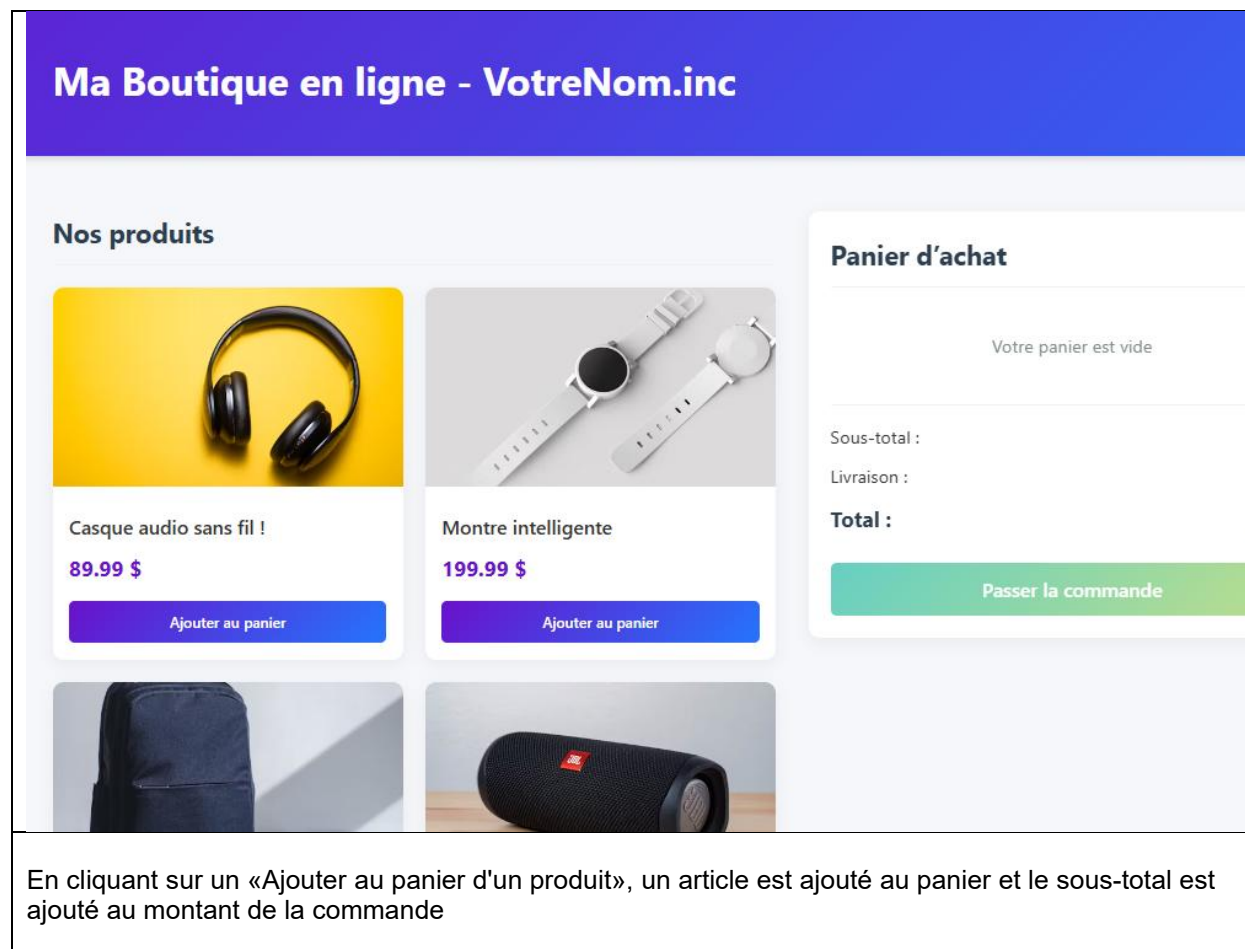
- Le travail sera remis sur Léa.

## Contexte :

- Le TP2 se fait en équipe de 2 étudiants.
- On vous fournit une base de code comprenant tout le code HTML et CSS. Vous devrez compléter les fonctions demandées en utilisant au maximum *jQuery*. Le code en JS sera pénalisé.

## Fonctionnement

Il s'agit de créer une page web d'une simple boutique en ligne.



## Ma Boutique en ligne - VotreNom.inc

### Nos produits



Casque audio sans fil !

89.99 \$

Ajouter au panier



Montre intelligente

199.99 \$

Ajouter au panier

### Panier d'achat



Casque audio sans fil !

89.99 \$

- 1

Sous-total :

Livraison :

Total :

9

Passer la commande

Ventes totales du magasin : 0.00 \$ [clear](#)

En cliquant de nouveau sur un article choisi, la quantité est incrémentée. Il est aussi possible d'utiliser les boutons + - et x pour modifier la quantité ou de le supprimer. Une quantité = 0 supprime automatiquement le produit du panier. La mise à jour du Total facture est effectuée en temps réel.

## Ma Boutique en ligne - VotreNom.inc



### Nos produits



Casque audio sans fil !

89.99 \$

Ajouter au panier



Montre intelligente

199.99 \$

Ajouter au panier



Sac à dos pour ordinateur portable

49.99 \$

Ajouter au panier



Haut-parleur Bluetooth

59.99 \$

Ajouter au panier

### Panier d'achat



Casque audio sans fil !

89.99 \$

- 2 + x



Montre intelligente

199.99 \$

- 3 + x



Haut-parleur Bluetooth

59.99 \$

- 1 + x

Sous-total :

839.94 \$

Livraison :

5.00 \$

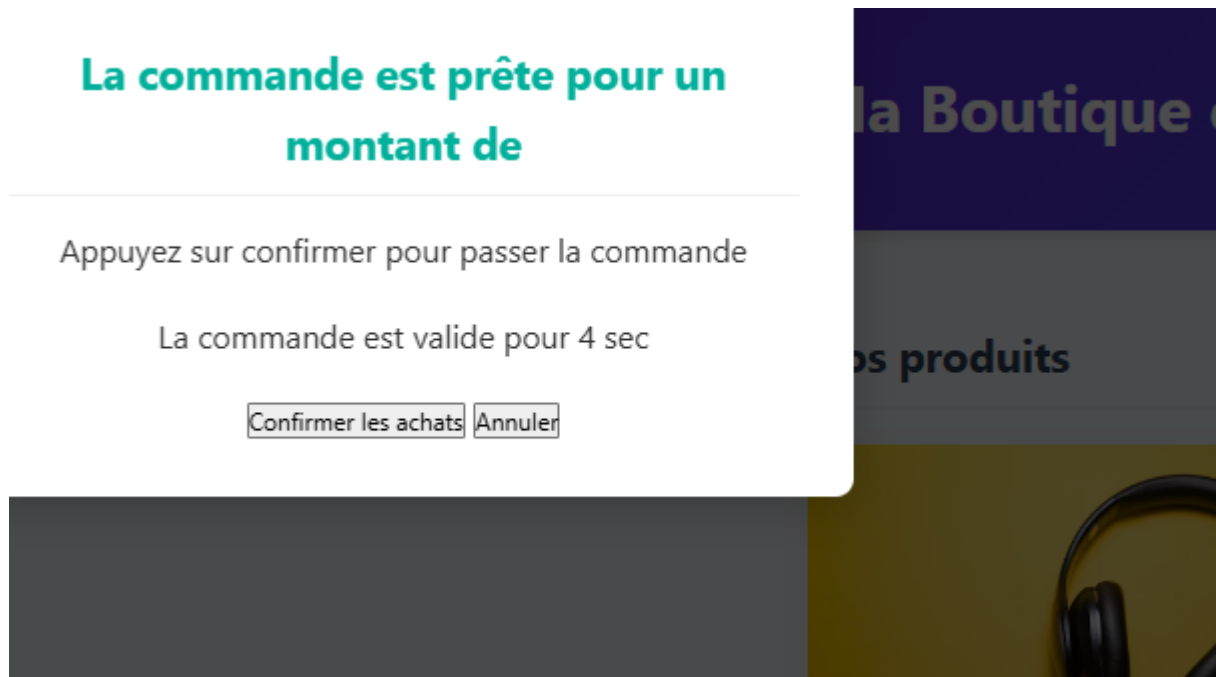
Total :

844.94 \$

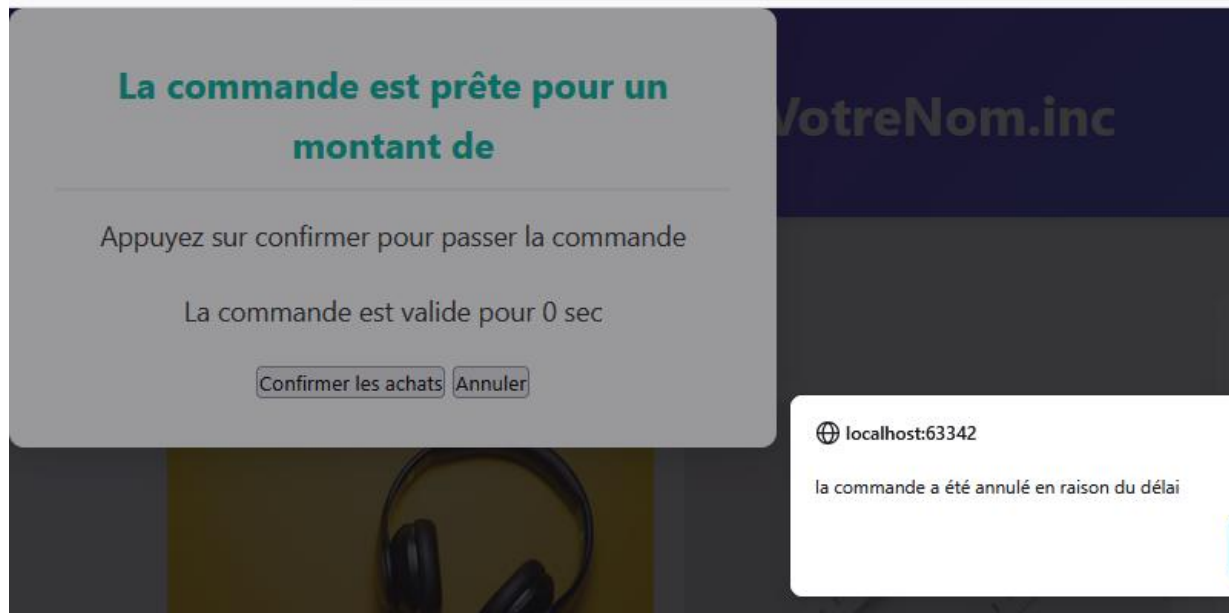
Passer la commande

Ventes totales du magasin : 0.00 \$ [clear](#)

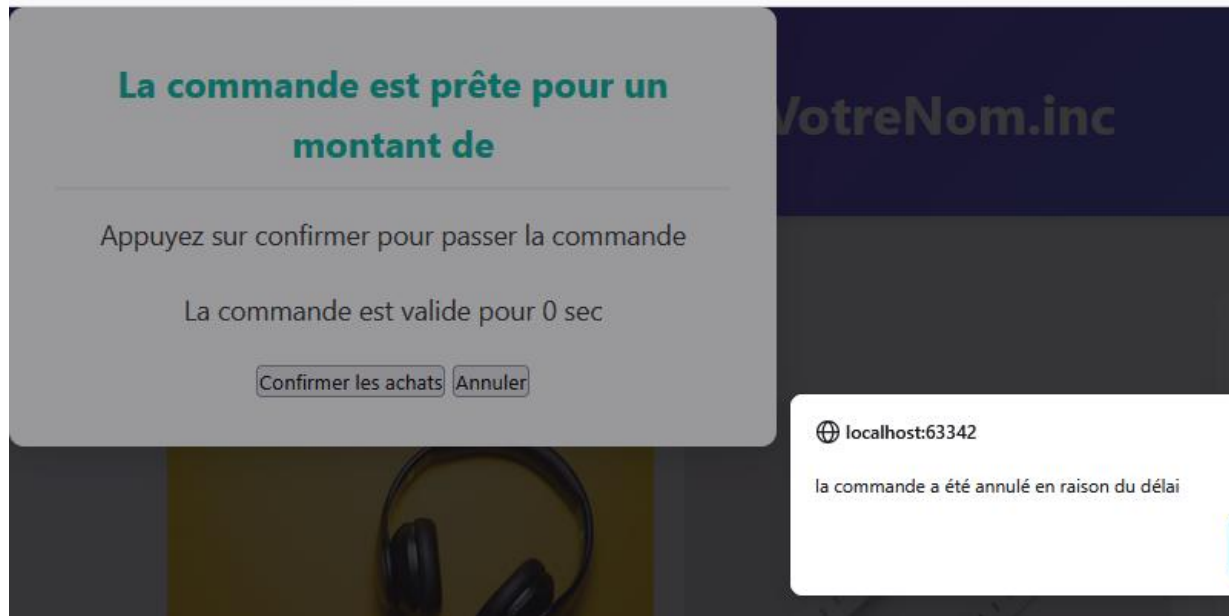
En cliquant sur passer la commande, une fenêtre s'affiche indiquant qu'il reste 5s pour confirmer la commande.



Si on dépasse 5s, un message indiquant l'annulation de la commande



En confirmant avant le délai, le montant de la commande est ajouté au Total des ventes du magasin



En tout temps on peut mettre à zéro les ventes totales du magasin en appuyant sur le bouton «Clear»

## À faire

1) Programmez les fonctions suivantes:

a) **initStore** :

- La méthode *initStore* contient déjà le code pour garder le contenu du cart dans la mémoire du navigateur même lorsqu'on quitte ce dernier. C'est le local storage qui permet de le faire <https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage>.
- Elle affiche les ventes totales (celles qui proviennent du *local storage*) dans la barre du bas (*depenses-totales*).
- Cette méthode affiche tous les produits disponibles en appelant la méthode *renderProducts* (voir plus loin).
- Elle doit également mettre à jour l'affichage du cart. On devrait obtenir un cart vide au premier rendu.

### Panier d'achat

Votre panier est vide

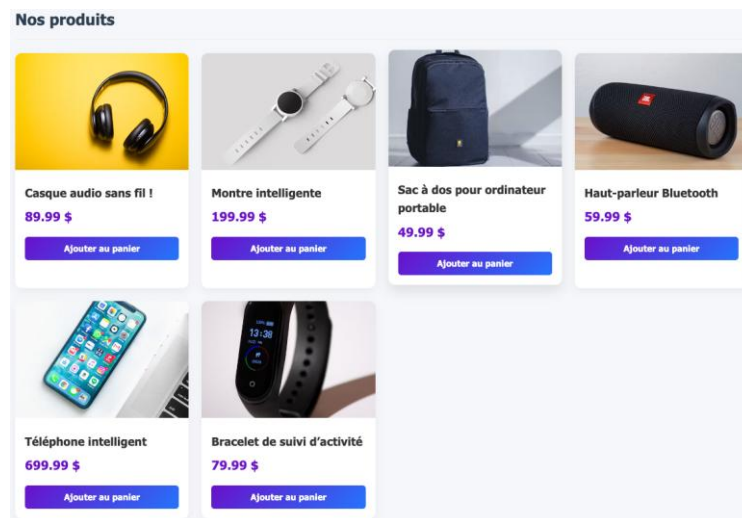
Sous-total :	0.00 \$
Livraison :	0.00 \$
<b>Total :</b>	<b>0.00 \$</b>

Passer la commande

b) **ConfigureButtons**:

- checkout-btn** :  
(1) Branche la méthode *checkout* (voir plus loin).
- checkout-confirmation**:

- (1) but : L'utilisateur confirme qu'il achète les produits du cart.
- (2) Ajoute le prix final du panier dans les ventes totales et l'affiche dans *depenses-totales* (barre du bas)
- (3) Elle vide le cart complètement.
- (4) Elle ferme la fenêtre de confirmation (*order-confirmation*).
- iii) **cancel-checkout:**
  - (1) But: l'utilisateur n'achète pas les produits du cart.
  - (2) Elle vide le cart complètement.
  - (3) Elle ferme la fenêtre de confirmation (*order-confirmation*).
- iv) **cart-button:**
  - (1) but: afficher et masquer le *cart*
  - (2) Si le cart est ouvert, elle le ferme.
  - (3) Si le cart est fermé, elle l'ouvre.
- v) **clear-total:**
  - (1) but: effacer le contenu du *local storage* et remettre l'application à 0.
  - (2) remet les ventes totales à 0; (dans la variable et dans le *local storage*)
  - (3) Affiche un montant de 0 dans la barre du bas (*depenses-totales*).
- c) **renderProduct:**



- i) Dans la division *products-container*, elle affiche chacun des produits contenus dans le tableau statique *products*.
- ii) ATTENTION, les boutons *addToCart* associés à chaque produit doivent être configurés à chaque fois qu'on recrée les produits. Chaque bouton doit retrouver l'ID du produit qui lui est associé, et appeler la fonction *addToCart* en lui passant ce dernier. Notez bien que l'attribut *data-\** permet de stocker une donnée dans le code HTML. Par exemple, *data-id* correspond à la donnée *id* et elle peut être récupérée avec la méthode *data(id)* de JQuery <https://api.jquery.com/data/> et [https://developer.mozilla.org/en-US/docs/Web/HTML/How\\_to/Use\\_data\\_attributes](https://developer.mozilla.org/en-US/docs/Web/HTML/How_to/Use_data_attributes)
- d) **addToCart:**
  - i) fonction utilisée lorsque l'utilisateur appuie sur l'un des boutons "*ajouter au panier*".
  - ii) Elle reçoit l'id du produit et peut donc retrouver le produit à partir de ce dernier.
  - iii) Elle vérifie ensuite si un exemplaire du produit est déjà dans le cart, auquel cas elle ne fait qu'augmenter l'attribut *quantite*. Quel attribut *quantite* ? Continuez à lire.
  - iv) Si aucun exemplaire du produit n'est dans le cart, elle ajoute une copie du produit dans le cart en lui ajoutant le nouvel attribut **quantite** au passage. Évidemment la *quantite* est à 1 pour le premier exemplaire du produit mis dans le cart.
  - v) Elle appelle ensuite *updateCartDisplay* pour s'assurer que tous les changements soient mis à jour dans le site.
  - vi) Finalement, elle fait clignoter le cart-count à la droite de la barre du haut. Elle l'agrandit de 50% pendant 300ms et puis le remet à sa taille normale (100%). La propriété CSS *transform* est utile pour le faire <https://developer.mozilla.org/en-US/docs/Web/CSS/Reference/Properties/transform>.
- e) **updateCartCount :**
  - i) but: mettre à jour le nombre de produits total dans le *cart*.
  - ii) Elle calcule le nombre total d'items dans le *cart* en se basant sur les quantités de chaque produit qui sont dans le *cart*.
- f) **calculatePrices:**

- i) calcule le *subtotal* des items en tenant compte des quantités voulues pour chaque type de produit.
- ii) Le shipping est **5\$** s'il y a un item et **0\$** autrement.
- iii) Le total est la somme du *shipping* et du *subtotal*. Notez qu'il n'y a pas de taxe.
- iv) Elle doit retourner les 3 valeurs dans un objet. Guidez-vous sur la variable *actualPrices*.
- g) **updatePrices**:
  - i) met à jour l'affichage des prix (subtotal, shipping et total) dans l'affichage du cart.
- h) **ChangeQuantite**(event delta):
  - i) Ajoute ou retire un de la quantité d'un produit qui est dans le cart.
- i) **clearCart** : efface le contenu du cart et mets à jour son affichage.
- j) **checkout**: (voir checkout-btn)
- k) **removeItem**(event) : retire l'élément sur lequel l'utilisateur vient de cliquer. C'est le *remove-item* qui devrait déclencher cette méthode.
- l) **checkout**:
  - (1) Cette méthode ne travaille que s'il y a quelque chose dans le cart.
  - (2) Elle fait apparaître la fenêtre de confirmation (order-confirmation).
  - (3) Elle fait en sorte que la fenêtre de confirmation se ferme après un délai préétabli. Elle affiche aussi un compte à rebours qui indique le nombre de secondes restantes avant la fermeture de la fenêtre à l'annulation du cart.
  - (4) Peu importe s'il est confirmé ou annulé, le cart doit être vidé.
  - (5) Assurez-vous finalement que l'affichage du cart soit mis à jour à la fin.

## Github

(5 points) de la note sont réservés pour tout travail effectué sur Github

---

FIN