N°	Fichier testé	Lignes de code	Noms des fonctions	Fonctionnalité	Action
1	**script.js**	5 - 26	`displayProducts`	Une page d'accueil montrant (de manière dynamique) tous les articles disponibles à la vente.	Ouvrir la page d'accueil du site web dans un navigateur
2	**product.js**	34 - 40	`displayProduct`	Afficher le produit sélectionné en amont sur la page d'accueil avec le bon id, l'image et le descriptif fourni par l'API	Sélectionner un produit sur la page d'accueil pour arriver sur la page produit
3	**product.js**	43 - 63	`saveProduct`	Créé un tableau « cart ». Converti l'objet « cart » en chaîne de caractères JSON puis sauvegarde l'objet dans le local storage.	Sur la page product, afficher l'inspecteur du navigateur puis dans l'onglet « appli » vérifier si les produits ajoutés via les inputs s'ajoutent bien au local storage.
4	**product.js**	75 - 85 87-97	`displayConfirmationPopUp` `displayErrorPopUp`	Affiche un message pop-up afin de valider l'ajout au panier ou avertir l'utilisateur en cas de mauvais renseignement des inputs.	Appuyer sur le bouton « ajouter au panier » soit avec une quantité et une couleur, soit juste avec une quantité ou juste une couleur soit vides
5	**product.js**	99 - 101	`closePopUp`	Ferme la fenêtre pop-up.	Appuyer sur le bouton « ajouter au panier » une fenêtre pop-up apparait puis disparait
6	**product.js**	103 - 110	`onClickAddToCart`	Au click sur le bouton « ajouter au panier » (si les inputs sont correctement renseignés), l'objet contenant la couleur, l'id et la quantité est converti en chaîne JSON et s'ajoute dans le local storage. En cas d'absence de la clé « cart » dans le local storage, celle-ci est créée.	Appuyer sur le bouton « ajouter au panier » une fois la couleur et la quantité sélectionnée.
7	**cart.js**	22 - 27	`refreshCart`	Lors de la suppression d'un produit, le panier doit visuellement faire disparaitre le produit de la page panier et le total doit également se mettre à jour.	Cliquer sur le bouton « supprimer » d'un produit.
8	**cart.js**	29 - 59	`displayCart`	Les produits stockés dans le local storage, doivent tous s'afficher dans la page panier.	Ouvrir la page « panier » dans un navigateur. Au moins un produit doit avoir été mis au panier.
9	**cart.js**	62 - 69	`changeHandlerQty`	Empêche la saisie de chiffres supérieurs à 100 ou inférieurs à 0 dans les inputs quantity.	Dans la page « panier », modifier la quantité de n'importe quel produit.
10	**cart.js**	61 - 73	`calculateCartAmount`	Le prix et la quantité totale doivent s'afficher à la fin de la liste des produits dans la page panier.	Ouvrir la page « panier » dans un navigateur
11	**cart.js**	82 - 96	`addEventListenerOnQtyInput`	Ecouter les inputs « Qté» des produits.	Modifier la quantité d'un produit dans le panier.
12	**cart.js**	113 - 122	`addEventListenerOnDeleteBtn`	Ecouter les boutons «Supprimer» des produits.	Supprimer un produit dans le panier.
13	**cart.js**	98 - 106	`modifyCartQty`	Modifie la quantité du produit concerné par l'input.	Modifier la quantité d'un produit dans le panier
14	**cart.js**	124 - 151	`displayPopUpProductDeleted`	En cas de suppression d'un produit dans le panier, un message s'affiche demandant à l'utilisateur de valider ou non la suppression.	Cliquer sur le bouton « supprimer » d'un produit.
15	**cart.js**	158 - 167	`deleteItem`	En cas de clic sur le bouton « supprimer », le produit doit être retiré du localstorage et la fonction refreshCart et appelée.	Cliquer sur le bouton « supprimer » d'un produit.
16	**cart.js**	299 - 301	`closeHelperSubmit`	Permet de fermer le message d'erreur en cas de mauvaise quantité saisie.	Dans la page « panier », mettre une quantité erronée dans n'importe quel input Qté. Cliquer sur le bouton « commander !», un message d'erreur apparait puis disparait au bout de 1,8s.
17	**cart.js**	254 - 281	`sendOrder`	Lors du clic sur le bouton « Commander ! », le remplissage du formulaire de contact est analysé et validé selon les conditions des regExp mises en place. Une vérification des quantité est également effectuée. Si tout est bien rempli, alors un objet comprenant les informations de contact et les produits sont envoyés vers l'API. En réponse, un numéro de commande est réceptionné et stocké dans l'url de la page confirmation.	Appuyer sur le bouton « Commander ! ».
18	**confirmation.js**	3 - 13	Pas de fonction	En arrivant sur la page confirmation, l'id de la commande est récupéré de l'url et il est affiché sur la page.	Ouvrir la page « confirmation »
19					
20					
21					
22					
23					
24					
25					
26					

Résultat attendu	Résultat observé
Affichage de l'ensemble des produits avec début de description et image	OK l'ensemble des produits s'affichent correctement.
Affichage du produit avec le bon id, l'image et la description	OK, le bon produit ainsi que l'image et le descriptif sont biens affichés. Le produit est appelé via son id sur l'API.
En cas de premier produit sélectionné, une clé « cart » doit s'ajouter dans le local storage avec l'id, la couleur et la quantité en item. Les produits avec un id et une couleur identique doivent rester sur une ligne.	OK, la clé « cart » se créée bien dans le local storage en cas de premier produit sélectionné, les items se remplissent bien en fonction des inputs remplis. En cas de produit avec un id et une couleur identique sélectionné, il met a jour la quantité et ne créé pas de nouvelle ligne.
En cas d'inputs vides ou mal remplis, un message sur fond rouge apparait indiquant un problème de sélection. En cas de bon remplissage des inputs (couleur + quantité) un message apparait indiquant l'ajout au panier.	OK, Un message apparait bien en dessous du bouton « ajouter au panier ».
La fenêtre pop-up doit disparaitre toute seule au bout de 1.8s après ouverture	OK, La fenêtre disparait bien tout seule.
Dans le localstorage, un objet doit être créé ou mis à jour, stockant les informations (id, Couleur, quantité) du produit dans la clé « cart ». En cas d'ajout du même produit (couleur et id identiques), la quantité seule doit se mettre à jour (pas de nouvelle ligne).	OK, Au click sur le bouton, le produit s'ajoute bien dans le local storage avec les infos id, couleur et quantité. Si on ajoute une nouvelle fois le même produit avec la même couleur, alors seule la quantité se met à jour.
Le produit supprimé doit disparaitre de la page visuellement et le total doit se mettre à jour en conséquence. Cela ne doit pas recharger entièrement la page pour des raisons de performances. Les écouteurs d'évènements du panier sont également rappelés.	OK, le produit est bien supprimé de la page et le total se met à jour sans recharger entièrement la page.
Le/les produits (image, nom, prix et couleur) stockés dans le local storage doivent tous s'afficher dans la page sans bugs visuels.	OK, les produits du local storage s'affichent tous correctement.
Si une quantité supérieur à 100 est saisi au clavier, alors l'input se met sur 100, si une quantité inférieure à 0 est saisie, alors l'input se remet sur 0. (Bien sûr le local storage est mis à jour lui aussi).	OK, impossible de mettre une quantité supérieure à 100 ou inférieure à 0 même au clavier. Le local Storage est mis à jour.
A la fin de la liste des produits, le total des quantités et du prix doit s'afficher.	Ok, le total s'affiche correctement.
Lors du clic sur l'input de quantité, la fonction modifyQty doit se lancer.	OK, la fonction modifyQty est bien appelée.
Lors du clic sur le bouton « supprimer », la fonction deleteItem doit se lancer.	OK, la fonction deleteItem est bien appelée.
Au clic, un message apparait au centre de la page indiquant a l'utilisateur si il souhaite confirmer la suppression. Si « confirmer » est sélectionné, le produit disparait de la page et du localstorage, si « annuler » est sélectionné, la fenêtre se ferme et rien ne se passe.	OK, le message apparait bien au clic. Lorsqu'on clique sur « confirmer » cela supprime bien le produit du panier et lorsqu'on clique sur « annuler » cela ferme la fenêtre sans toucher au panier.
Au clic, le produit est supprimé du local storage et de la page.	OK, Le produit est bien supprimé du local storage et de la page.
Lors d'une mauvaise saisie de quantité, au clic sur le bouton « commander ! » un message apparait demandant de vérifier les quantités saisies. Elle se referme au bout de 1,8s.	OK, le message se ferme bien au bout de 1,8s.
Les inputs des utilisateurs doivent être analysés et validés pour vérifier le format et le type de données avant l'envoi à l'API. Un message d'erreur apparait en cas de mauvais remplissage. Un numéro de commande est reçu via l'API, puis stockée dans l'url de la page confirmation vers laquelle l'utilisateur est redirigé.	OK, un message s'affiche bien en cas d'erreur de remplissage et l'envoi du formulaire est bloqué. En cas de réussite, le numéro de commande est stocké dans l'url de la page confirmation et l'utilisateur est redirigé vers celle-ci.
Lors de l'ouverture de la page confirmation, l'id de la commande envoyé par l'API est affiché dans un encart au milieu de la page.	OK, le message ainsi que l'id de la commande s'affichent correctement.