

Fichier JS	Lignes de code testées	Fonction testée	Résultat attendu	Comment vérifier le résultat attendu	Problème possible
index.js product.js cart.js	3 17 24	fetch(url) fetch(url) fetch(url + id)	L'API Fetch fournit une interface JavaScript pour l'accès et la manipulation des parties de la pipeline HTTP, comme les requêtes et les réponses. Cela fournit aussi une méthode globale fetch() qui procure un moyen facile et logique de récupérer des ressources à travers le réseau de manière asynchrone.	Il faut afficher la page, si fetch ne donne pas de retour de réponse ou de promesse, toutes les infos des produits ne s'afficheront pas. Il faut utiliser des console.log	les produits ne s'affichent pas car fetch ne renvoie pas de données
index.js	59 à 66	.catch(function(error))	Une promesse fetch() va retourner une TypeError quand un problème réseau s'est produit. En cas de rejet de la promesse on affiche une div avec un message d'erreur	ne pas taper la ligne de commande "node app.js" dans le terminal afin de tester l'affichage du message d'erreur de la div	la div contenant le message ne s'affiche pas
product.js	9 à 14	getId()	cette fonction permet de récupérer l'id et les paramètres passés dans l'uri du produit	Dans la console on doit avoir un message de navigation, indiquant dans quel URL nous sommes	l'id et les paramètres ne s'affichent pas
product.js	28 à 199	afficherLeProduit(peluche)	Cette fonction permet de créer la card complète de l'article choisi sur la page d'accueil, on y retrouve nom, options, description, photo, grâce à la bonne récupération des données	il faut afficher une page d'un article	les éléments ne s'affichent pas
product.js	155	btnAjout.addEventListener("click", envoiDuProduit)	Au clic sur le btnAjout la fonction envoiDuProduit et l'arte sont appelées	le produit choisi sera envoyé dans le local storage, si pas de quantité sélectionnée il y aura un message d'erreur, si une quantité a été sélectionnée alors le message d'alerte "ajouté au panier sera affiché" il faut placer un console.log dans l'écoute de l'événement afin de s'assurer que l'on est bien dans le "click"	les fonctions ne sont pas appelées
product.js	157 à 197	envoiDuProduit(event)	Elle ajoute un tableau stringifié au format JSON dans le local storage. Ce tableau représente les éléments qui sont dans le panier, on y trouve l'id, la couleur et la quantité. Si un produit est déjà dans le panier on augmente sa quantité.	en cliquant sur le bouton "Ajouter" on va appeler la fonction plusieurs fois, pour différentes couleurs de peluches, c'est dans l'onglet application de DevTools que l'on peut vérifier les valeurs. Si un produit est déjà dans le panier et qu'on en rajoute un identique, on pourra voir dans la console "déjà présent" et la quantité sera augmentée	La récupération des données peut être mal faite ou incomplète, par exemple récupérer le mauvais id, la mauvaise couleur...
cart.js	23 à 25	getById(id)	permet de récupérer les données de l'api ainsi que l'id du produit sélectionné au préalable	l'appel de fetch renvoie bien les données de l'API et l'id, la page s'affiche correctement	Fetch ne fonctionne pas, l'id n'est pas récupéré
cart.js	26 à 119	addTeddle(teddle)	permet la mise en page du panier sous forme d'un tableau. On y retrouve les données principales (nom, prix, qté, totaux)	ajout de produit au panier afin de voir s'ils sont pris en compte et s'ajoute dans le tableau en mettant à jour les prix	les produits ne s'ajoutent pas dans le tableau, les prix ne se mettent pas à jour
cart.js	95 à 111	cardBtn.onclick	permet de supprimer 1 élément du panier si celui-ci ne convient plus	vérifier que seulement 1 article est supprimé et pas tous. Si plus d'article, le panier sera donc égale à 0 on enlève la key panier	le bouton ne fonctionne pas, la clé est vide mais ne se supprime pas
cart.js	125 à 222	upPanier	permet d'afficher de manière dynamique le contenu du local storage, si le panier est vide "votre panier est vide" si le panier comporte des articles alors on appelle la fonction addTeddle(teddies)	dans l'onglet application de devTools il faut vérifier que les valeurs qui sont contenues dans la key panier correspondent aux produits affichés, les éléments (id, color, quantité) doivent être présents	mauvais affichage des produits
cart.js	159 à 168	validateEmail	cela permet de comparer le format de l'email que l'utilisateur rentre et une expression régulière définie. Si l'email est bon : email valide, si l'email n'est pas bon : email invalide	Essayer de rentrer des adresses emails incorrectes	l'email est considéré valide alors qu'il ne remplit pas les critères, ou inversement. Cela peut être dû à une mauvaise déclaration de l'expression de référence ou à un mauvais appel de la fonction
Confirm.js	17 à 23	endOfOrder.addEventListener("click")	Lors du clic sur le bouton "revenir à l'accueil", le local storage sera vidé entièrement et l'utilisateur redirigé	dans l'onglet application de devTools on vérifie que le local Storage est bien vide. On vérifie également à l'aide d'un console.log() que le bouton est bien écouté et que l'on rentre dans la fonction au moment du clic	le localStorage reste tel quel et ne se vide pas, l'écoute du bouton ne se fait pas