Fichier JS	Lignes de code testées	Fonction testée	Résultat attendu	Comment vérifier le résultat attendu	Problème possible
app.js	Ligne : 6	main	On attend le retour de getArticles pour executer tous le code, on boucle notre function displayArticle pour créer tous nos articles depuis getArticles qui est un fetch de l'API.	Il faut déjà que la func displayArticle soit fonctionnelle	Si on fait pas cette fonction en async / await ca ne marcheras pas toujours faire du js asynchrone lorsqu'on attend la réponse d'une API
app.js	Ligne : 19 - 28	getArticles	On récupere les données de l'api (c'est un fetch sur le localhost)	Si le fetch de l'api marche et qu'on arrive a afficher nos données dans un console.log de la data alors cette fonction est censé marcher	Si le server est off il y aura une erreur ou si on met une mauvaise request url
app.js	Ligne : 31 - 42	displayArticles	On créé un article avec du innerhtml et qui a comme parametre article dont on passera getArticles plus haut dans une boucle pour afficher tous nos produit de l'API	On peut commencer a log tous nos élément de notre tableau d'objet si cela marche alors on peut les utiliser pour les mettre dans notre innerhtml	Si le server est off il y aura une erreur et possible d'erreur d'intégration dynamique
article_id.js	Ligne : 12 - 14	getArticleID	Cela nous retourne l'URL de notre élément avec son ID en utilisant la function searchParams	Faire un searchParams.get("id")	Peu retourner le mauvais id
article_id.js	Ligne : 17 - 26	getArticle	On récupere les données de l'api (c'est un fetch sur le localhost) avec comme parametre l'id de notre élément lors du clic	Fetch de notre api avec comme parametre articleID qui récupére l'id de searchParams qui doit etre l'id d'un de nos élément de notre API donc on peut test l'url du localhost avec l'id pour voir si c'est fonctionel	si le server est off il y aura une erreur ou si on met une mauvaise request url
article_id.js	Ligne : 29 - 66	hydrateArticle	Cette fonction créée l'html d'un seul élément d'un produit de l'api	Log la data de notre fetch de localhost...ID	Si le server est off il y aura une erreur et erreur possible d'intégration dynamique de nos éléments
article_id.js	Ligne : 69 - 75	createLenses	Cette fonction a une boucle for qui créé de lhtml elle aura comme parametre lense que j'utiliseais dans hydrateArticles pour parcourir mon tableau de article.lenses de mon produit pour tous les affichers/créer lors de la création du produit	Faire une boucle for de article.lenses	pas de possibilitée de choisir le type de lentille, ou si le server est off affichage impossible
article_id.js	Ligne : 79 - 83	increaseValue	Cela permet d'incrémenter la valeur situé dans la quantité avec notre button + et de changer la valeur de 1 a 2 par exemple dans la quantitée de produit	Si la valeur s'incrémente alors ca marche	Pas de limitation lors de l'incrémentation donc l'user peut mettre 1000+ produit par exemple
article_id.js	Ligne : 85 - 92	decreaseValue	Cela permet de decrémenter la valeur de la quantitée avec notre button - et de changer la valeur de 2 a 1 par exemple dans la quantitée de produit, j'ai mis une limitation pour ne pas passer sous 0 donc on ne peut pas avoir de quantitée de -1 ou plus par exemple	Si la valeur se décrémente alors ca marche	Absence d'informations
article_id.js	Ligne : 97 - 129	addToCart	Cela va nous créer un objet de produit qu'on va push dans un tableau avec comme clé product en format json dans notre local storage lors du clic de l'user sur ajouter au panier et si nous avons quelque chose dans le tableau alors on va push le tableau camProduct dans le tableau du localStorage et si le localStorage est vide alors on aura un tableau vide	log de camproduct lors du clic et log de productInLocalStorage pour voir si notre tableau camproduct est bien dans notre localstorage	Local storage vide apres ajout du panier
cart.js	Ligne : 13 - 91	makeCart	Cela créé un item avec de l'innerhtml depuis notre tableau d'éléments (productInlocalStorage), on fait donc une boucle for pour chaque élément de ce tableau pour afficher chaque produits en question dans notre page panier	Faire un log de notre localStorage : product	absence d'informartions du produit ou prix erronés de la quantité ou le prix
cart.js	Ligne : 77 - 80	clearCart	Cela clear le localStorage et refresh la page pour pouvoir afficher le message Panier vide dans notre page panier	localStorage.removeItem("product")	local storage pas vidée et l'user aura toujours son panier de remplie
cart.js	Ligne : 96 - 184	makeForm	Créé notre formulaire avec de l'innerhtml, avec un add evenlistener en preventdefault et ajouter des conditions a notre form que lorsque que la value de nos inputs sont true a la regexp alors l'envoie du formulaire est possible et donc nous avons une redirection vers validation.html	envoie du formulaire et redirection sur la page de confirmation	Impossible de submit la commande
cart.js	Ligne : 171 - 183	getInfoForms	Récupère les valeurs du formulaires dans le local storage qu'on réutliseras plus tard lors de la page Validation.html	log de nos variables qui contient les value de chaque input	Possible que les données de l'user ne soit pas récupérées
cart.js	Ligne : 227 - 308	lnameChecker, fnamec	Ce sont des fonctions qui ont pour but de controler nos input avec des regexp si ce n'est pas valide alors notre variable = null et si c'est valide et c'est bon pour la regexp alors notre variable est true et a la	impossible de soumettre le formulaire si l'un des champs obligatoire n'est pas rempli	possible de soumettre le formulaire si l'un des champs obligatoire n'est pas rempli
cart.js	Ligne : 69 - 74	clearItem	Cette fonction a un parametre index qui nous retournera l'index de notre tableau pour pouvoir le supprimer celui-ci, en gros cela supprime un élément de notre tableau donc supprime un élément dans	On log notre tableau avec l'index en question, on le splice on set item pour renvoyer notre nouveau tableau qu'on parse en json et on reload la page	Supression du mauvais élément ou suppression impossible
valid.js	Ligne : 2 - 11	makeValidation()	Cette function nous créée l'html et nous retourne les données du client lors de ca commande dont un bon de commande généré aléatoirement avec Math.random et clear le local storage aprés ca.	récupération des bonnes données saisie par l'user	Données de l'user non récupérées ou erronées
valid.js					
valid.js					
valid.js					