Fichier JS	Lignes testées	Fonction testée	Résultat attendu	Comment vérifier le résultat attendu	Problème possible
index.js	1 à 38	addProduit	Si l'appel à l'API fonctionne, la fonction va créer une page contenant la liste des produits. Chaque produit est présenté dans une carte intégralement cliquable avec une image, un nom de produit, un descriptif et un prix. Au clic, l'utilisateur est ramené à la page concernant le produit.	Lancer la page d'index du site et vérifier qu'elle contient bien les cartes de chaque produit. Ensuite, cliquer sur une carte afin d'être redirigé vers la page produit.	Si la liste est trop longue, le site peut être interminable à parcourir pour l'utilisateur
product.js	15 à 46	addProductInfo	Appel à l'API afin de créer la page produit. Si l'appel est réussi, la fonction affiche l'image, le nom, la description ainsi que le prix unitaire du produit. Il est également possible de sélectionner une lentille et la quantité d'appareil photo que l'utilisateur souhaite acquérir.	Lancer la page et vérifier que le produit s'affiche correctement avec les bonnes informations.	Si l'appel à l'API ne fonctionne pas, la page ne s'affichera pas.
product.js	48 à 53	chooseLenses	Affiche un menu déroulant permettant à l'utilisateur de choisir la lentille qu'il souhaite acheter avec son appareil photo.	Lancer la page et vérifier que le menu déroulant propose bien les bonnes lentilles pour chaque appareil photo.	Si les informations relatives aux lentilles ne sont pas renseignées dans l'API, l'affichage du menu ne se fera pas.
product.js	69 à 74	objProduitInfo	Stocke les informations relatives au produit que l'utilisateur consulte afin de les envoyer vers le panier si le client souhaite acheter le produit.	Lancer la page et vérifier grâce à un console.log(objProduit) que les informations relatives au produit affiché sont bien stockées.	
product.js	79 à 95	selector.addEventListener	Le bouton reste grisé tant que l'utilisateur n'a pas choisi une quantité et une lentille pour l'appareil photo.	Lancer la page et vérifier que le bouton est grisé par défaut. Tester en modifiant uniquement la lentille ou la quantité, le bouton reste grisé. Tester en appliquant une quantité et une lentille, le bouton devient cliquable.	
product.js	97 à 109	btn.addEventListener	Vérifie si le panier existe déjà lorsque l'utilisateur ajoute un produit. S'il n'existe pas, cela crée le panier et ajoute le produit. S'il existe, on passe à la fonction ajoutProdDoublon.	Ouvrir le local storage et vérifier que le produit a bien été ajouté dans le panier.	
product.js	111 à 131	ajoutProdDoublon	Vérifie si le produit est déjà dans le panier afin d'éviter les doublons. Si le produit est déjà dans le panier, un message d'erreur alerte l'utilisateur. S'il n'y est pas, le produit est ajouté au panier.	Ouvrir le local storage et vérifier que le produit a bien été ajouté dans le panier sans supprimer un autre produit.	

cart.js	6 à 57	displayProduct	Affiche le produit présent dans le panier avec : son image, son prix unitaire, son nom, la lentille choisie et la quantité présente dans le panier.	Lancer la page panier après avoir ajouté un produit au panier et vérifier que toutes les informations sont présentes.	
cart.js	60 à 70	select	Affiche le sélecteur permettant de changer la quantité de produit dans le panier.	Sur la page Panier, vérifier que le sélecteur de quantité propose de modifier la quantité de produit.	
cart.js	72 à 82	modifyQuantity	Lorsque l'utilisateur change la quantité du produit dans le panier, le local storage se met à jour avec la nouvelle quantité et le prix total est recalculé.	Ouvrir le local storage après modification de la quantité pour vérifier que l'information a bien été mise à jour.	
cart.js	84 à 90	modifyTotal	Lors de la modification de la quantité d'un produit, le prix est recalculé par rapport à la nouvelle quantité choisie.	Après modification de la quantité, vérifier que le montant total affiché en bas de page a bien changé.	
cart.js	92 à 99	calculTotal	Crée chaque carte produit dans le panier à partir de la fonction displayProduct et affiche le total final du panier.	Lancer la page Panier après avoir ajouté au moins deux produits dans le panier. La page affiche pour chaque produit une carte contenant toutes les informations et en bas de page, le prix total est indiqué.	
cart.js	102 à 114	deleteProduct	Permet à l'utilisateur de supprimer un produit de son panier. Le produit en question est également supprimé du local storage et le total du panier est mis à jour.	Ouvrir le local storage pour vérifier que la suppression du produit est bien effective. Vérifier également que le prix a bien été mis à jour en bas de la page Panier.	
cart.js	116 à 124	modifyIdx	Lors de la suppression du produit, les id de chaque carte produit sont modifiés pour correspondre à l'index du produit dans le local storage.	Ouvrir l'inspecteur d'élément sur le navigateur ainsi que le local storage afin de comparer que les id correspondent bien au index de chaque produit.	
cart.js	167 à 174	disableConf	Si le panier est vide, l'utilisateur ne peut pas valider son panier pour accéder au formulaire d'expédition.	Ouvrir la page Panier avec un panier vide afin de vérifier que le bouton de validation du panier est bien grisé.	
cart.js	176 à 192	verifyInfo	Permet de vérifier si les informations renseignées dans le formulaire d'expédition correspondent à ce qui est attendu.	Valider le panier afin d'accéder au formulaire. Remplir le formulaire avec les informations clients et cliquer sur Passer la commande. Un pop-up d'erreur s'affiche si les informations renseignées ne correspondent pas à celles attendues.	

cart.js	196 à 215	sendForm	Récupère les produits du panier ainsi que les informations clients pour les envoyer vers l'API.	Valider le panier, renseigner le formulaire et cliquer sur Passer la commande afin d'accéder à la page de confirmation de la commande.	
cart.js	217 à 245	addEventListener	Appelle la fonction verifyInfo pour vérifier les informations clients. Si la fonction retourne « true », les produits du panier ainsi que les informations clients sont envoyés vers l'API et l'utilisateur est redirigé vers une page de confirmation de sa commande. Si la fonction retourne « false », un message d'erreur apparaît demandant à l'utilisateur de corriger son formulaire d'expédition.	Valider le panier, renseigner le formulaire et cliquer sur Passer la commande. Si les informations renseignées correspondent à celles attendues, l'utilisateur est automatiquement redirigé vers la page de confirmation de commande. Si ce n'est pas le cas, un pop-up d'erreur s'affiche.	
validation.js	1 à 11	confirmationCommande	Si le local storage est vide, l'utilisateur arrive sur une page lui indiquant une erreur. Sinon, l'utilisateur arrive sur une page de confirmation de commande avec un récapitulatif.	Valider le panier contenant au moins un produit, remplir le formulaire d'expédition et confirmer la commande pour accéder à la page de confirmation.	
validation.js	15 à 25	conf	Le site affiche un message de confirmation avec un récapitulatif mentionnant : le numéro de commande, le nom et prénom du client, son adresse et sa ville ainsi que son adresse mail.	Sur la page de confirmation, vérifier que les informations à l'écran correspondent bien à celles renseignées à la validation du panier.	