Planification des tests unitaires

Suite de tests unitaires pour couvrir au minimum 80% de la base de code pour le frontend, soit sur 4 pages html et 4 fichiers Javascript associés.

Globalement à tester :

- Vérification de la disponibilité des APIs et gestion des erreurs,
- Passage d'une page à l'autre avec l'affichage demandé,
- Article mis au panier (ou caddy),
- Suppression de l'article du panier,
- Modification de la quantité de l'article (en plus ou moins),
- Calcul du prix total,
- Validation des champs du formulaire et du formulaire lui-même,
- Page de confirmation après interrogation de l'API.

NOM DU TEST	ACTION A EFFECTUER	RESULTAT ATTENDU	CODE A CONTROLER
Connexion à la page index	Entrez l'adresse du site dans le navigateur	La page d'accueil s'ouvre avec tous les choix thématique	Page index.html disponible et en ligne
Barre de recherche	Cherchez un mot ou un élément d'un mot pour trouver son thème plus facilement dans la liste	A chaque saisie d'un caractère le résultat s'affine	index.js // fonction barre de recherche
Connexion à la page catalogue suivant le thème choisi	Au clic sur un bouton du catalogue choisi, nous arrivons bien au bon catalogue	La page catalogue thématique s'ouvre avec la liste de tous les articles disponibles sur l'API	index.js // fonction d'écoute et utilisation de localStorage
Requête à l'API du thème	Gestion de la requète avec des promesses et vérification du résultat et gestion de l'erreur	Retour d'un fichier objet avec le contenu des éléments à afficher. Affichage de vérification sur la console	product.js // fonction global askApis(theme) ligne 31 environ
Test panier vide	La barre de navigation autorise l'accès au panier: vérifiez que le panier est bien vide	Message d'alerte indiquant que la panier est vide	Gestion direct sur la page my_caddy.html <main> <première div id="alertHead">
Test panier vide, mais formulaire rempli	Bloquez l'envoi du formulaire sans avoir passer une commande d'article(s)	Affichage d'une alerte indiquant l'absence de commande	my_caddy.js // fonction d'écoute du bouton Envoyez (environ ligne 135)
Affichage de tous les produits du thème	Affichez les articles quelque soit leur nombre en gérant de façon dynamique la page html et en ajoutant l'identifiant unique de l'article	Affichage de chaque produit dans une carte avec photo, nom, texte descriptif et prix	products.js // fonction displayProduct (environ ligne 46)
Sélection de l'article grâce à son bouton	Dispose une écoute sur chaque bouton de l'article, quelque soit le nombre d'article. Récupère son identifiant et le numéro d'ordre de l'article dans l'affichage. Gère le choix de l'option en fonction du thème, car l'APIs ne renvoient pas le même nom du tableau d'option (colors, lenses ou varnish) et, quelque soit le nombre d'options disponibles	Préparez l'affichage de la sélection avec des identifiants, la bon intitulé de l'option avec son contenu spécifique. Affichage de contôle sur la console des 2 identifiants	products.js // fonction selectOneProduct (environ ligne 65)

NOM DU TEST	ACTION A EFFECTUER	RESULTAT ATTENDU	CODE A CONTROLER
Affichage de l'article selectionné	Gestion dynamique de l'affichage de l'article. Trois boutons utilisateur sur écoute: retour à la liste des articles (fonction backToTheList), menu déroulant choix d'option (et son enregistrement immédiat), panier	Affichage de l'article avec deux informations supplémentaires: les options disponibles et un bouton pour ajouter au pannier. Modification du titre h1 de la page. Activation d'un bouton du menu	products.js // fonction displayOneProduct (environ ligne 92)
Enregistement au panier	Utilisation d'indexedDB (requête et promesse) pour l'enregistrement d'un objet contenant tous les champs de la sélection (plus quantité à 1 et date de cette action). Gestion des erreurs à toutes les manipulations de la base de données et validation de l'information complète. Ouverture de la base indexée sur l'id article, puis transactions	Affichage d'une alerte validant la saisie utilisateur. Enregistrement de tous les champs connu de l'article selectionné dans un objet. Affichage sur la console de la bonne réalisation de la promesse	products.js // fonction putInMyCaddy (environ ligne 121 et ligne 11 pour l'ouverture de la base de données)
Retour à la liste thématique, depuis l'article unique ou du panier	Avant de revenir à la fonction displayProduct, effacement de toutes les variables ayant servi à pointer l'article selectionné et effacement de l'affichage de la sélection	Changement du titre h1. Mise à zéro de l'affichage et des variables de sélection d'un article. Le retour a la liste maintient le panier	products.js // fonction backToTheList (environ ligne 147)
Affichage du contenu du panier	Interrogation indexedDB (requête et promesses). Utilisation de la boucle Cursor (spécifique à indexedDB) pour affichage de tous les articles, quelque soit leur nombre. Introduction de l'id article dans le code html pour bien imputer la modification du nombre ou la suppression de l'article	Résumé de l'article avec photo réduite, nom et son option, quantité modifiable et suppression possible. Affichages de vérification sur la console	my_caddy.js // fonction displayData (environ ligne 35) après ouverture de la base indexedDB (ligne 14 environ)
Calcul du prix et tableau des id (en même temps)	A chaque affichage, deux compteurs (mis préalablement à 0) calculent le prix total et enregistrent un tableau des id	Le prix total apparaît sur la page my_caddy.html et un tableau des id dans la console.	my_caddy.js // fonction displayData (environ lignes 54 et 55)
Levée de deux exceptions sur Alert-info et prix non nul	1er) affichage dynamisque du bandeau alerte géré en html: modification du titre si panier plein ET si non fermé par l'utilisateur 2nd) affichage du prix uniquement différent de 0	Affichage de l'alerte info modifiée si l'utilisateur ne l'a pas fermée et, le prix n'apparaît que si le panier contient un article	my_caddy.js // fonction displayData (environ lignes 65)

NOM DU TEST	ACTION A EFFECTUER	RESULTAT ATTENDU	CODE A CONTROLER
Modification des choix à l'intérieur du panier: quantité ou supprimer	Tous les boutons sur écoute, quelque soit leur nombre. Interrogation d'indexedDB avec une transaction: suppression simple dans la base à l'aide de l'id; modfication de la quantité de l'id puis mise à jour de l'objet dans la base	Modification de la quantité ou suppresion d'un article: nouvel affichage et totaux (prix et tableau des id) mis à jour. La navigation est toujours possible vers la liste pour remettre l'article. Affichages de vérification	my_caddy.js // fonction modifyChoice (environ lignes 79)
Validation du formulaire	Les données du formulaires doivent être correctes et bien formatées. Gestion directe en html grâce au bouton submit, d'un pattern RegEx et required	Si champ oublié, une info-bulle indique qu'il faut le remplir. Si format non respecté, une autre info-bulle le rappelle. Envoi possible uniquement quand les cinq champs sont correctement remplis. Formulaire récupéré en JS et bouton submit désactivé	my_caddy.html lignes 77 à 106 environ. My_caddy.js (ligne 137 environ) pour la déactivation du bouton Envoyer
Récupération données du formulaire	Uniquement lorsque tous les champs sont validés	Stockage des 5 champs dans 5 variables	my_caddy.js // fonction dataRetrievalContactForm (environ lignes 145)
Requête à l'API du thème avec les données du formulaires et le tableau des identifiants	Gestion de la requète avec des promesses et vérification du résultat et gestion de l'erreur	Retour d'un string orderId. Affichage de vérification sur la console. Sauvegarde sur localStorage du prix total et du numéro de confirmation de commande. Affichage d'une alerte pour indiquer que l'action Envoyer a fonctionné	my_caddy.js // fonction global askApis(theme) ligne 155 environ

NOM DU TEST	ACTION A EFFECTUER	RESULTAT ATTENDU	CODE A CONTROLER
Levée d'une exception Undefined	La navigation sur le site ne permet pas de sortir d'un catalogue thématique sans arriver à la commande, mais la flêche retour du navigateur oui. Il est donc possible de commander dans les trois catalogues thématiques disponibles à ce jour. Cela fonctionne, mais les APIs ne le gèrent pas encore => modification du texte à afficher en évitant de laisser apparaître "undefined".	Affichage sur la page order_confirmation.html du prix total et du numéro de commande ou d'un message alternatif. Affichage de contrôle sur la console	order_confirmation.js environ ligne 12 à17
Mise à zéro du panier	Videz la base de données IndexedDB du navigateur du panier commandé	Affichage de contrôle sur la console	order_confirmation.js // fonction clearData (ligne 43 environ) et ouverture de la base (ligne 25 environ)