PLAN DE TEST PROJET 5 - G BLEIN

	Fichier JS	Lignes	Fonction testée / Objectif	Résultat attendu	Méthode de vérification	Problèmes possibles	Résultat observé
1	index.js	1 à 11	Récupération des données sur l'API via la méthode 'Fetch'	La requête doit interagir avec l'API pour récupérer les données du fichier *product.j s* présent dans le back, et répertoriant tous les modèles disponibles. Sinon envoyer un message d'erreur	Effectuer un *console.log*	Serveur inaccessible. Changement d'adresse ou de nom de l'API	OK / Message d'erreur
2	index.js	15 à 29	Display Products -> Affichage de tous les produits	Affichage dynamique de tous les produits en catalogue sur la page *index.html*	Vérification de l'affichage ou non sur *index.html*	Les éléments n'apparaissent pas du tout, ou alors seulement 1 seul	Tous les produis et élements apparaissent bien
3	product.js	1 à 4	Récupération de l'ID du produit choisi via la méthode URLsearchparams	Obtenir la valeur 'ID' du produit qui a été choisi	Effectuer un *console.log*	Incapacité à récupérer l'ID	ID récupéré
4	product.js	8 à 10	Récupération des données sur l'API via la méthode 'Fetch'	La requête doit interagir avec l'API pour récupérer les données concernant LE produit qui a été choisi	Effectuer un *console.log*	Serveur inaccessible. Changement d'adresse ou de nom de l'API	OK
5	product.js	12 à 27	Affichage du produit choisi et des choix de couleurs possibles	Affichage dynamique du produit sélectionné et avec toutes ses informations, sur la page *product.html*	Vérification de l'affichage ou non sur *index.html*	Les éléments n'apparaissent pas du tout, ou alors seulement 1 seul	Le produit choisi apparait bien, ainsi que les couleurs disponibles
6	product.js	32 à 33	Ecoute du click sur le bouton "AddToCart" via la méthode 'addEventListener'	La communication doit être correcte avec le DOM pour que chaque click soit bien écouté	Effectuer un *console.log*	Problème de communication avec le DOM	OK
7	product.js	44 à 78	AddFirstProductToCart -> envoi d'un premier produit vers le panier si celui-ci est vide	Création d'un Array vide 'productCart' contenant les informations à propos du produit et de la quantité sélectionnée. Envoi de ce tableau vers le Local Storage	Effectuer un *console.log*. Regarder également ce qui apparaît dans *Application-LocalStorage*	Problème de communication avec le DOM ou avec le local storage. Non écriture au format JSON	OK Le tableau apparaît bien dans le LocalStorage
8	product.js	102 à 110	Si le produit à valider est déjà validé, modification de la quantité dans le local storage. Utilisation de la méthode *.find* pour inspecter le local storage et comparer les ID et les couleurs	Au click, on identifie si le produit (même ID et même couleur) est déjà dans le panier. Puis on modifie la quantité dans le local storage et on envoie une alerte "quantité modifiée"	Effectuer un *console.log*. Regarder également ce qui apparaît ou est modifié dans le local storage	Incapacité à comparer les produits. Problèmes de communication avec le local storage	Identification correcte Quantité modifiée dans le local storage
9	product.js	112 à 128	AddNewProductToCart -> envoi d'un nouveau produit vers le panier si celui-ci n'est pas vide	Envoi de productCart vest le local storage. Rajout d'une autre ligne/autre index dans le local storage	Effectuer un *console.log*. Regarder également ce qui apparaît ou est modifié dans le local storage	Problème de communication avec le local storage. Remplacement et non ajout d'un index. Non écriture au format JSON	OK Le tableau avec tous ses index apparaît bien dans le local storage
10	cart.js	4 à 10	Message d'affichage "votre panier est vide"	Affichage d'un message pour prévenir le client s'il ny'a rien dans son panier	Vérification de l'affichage ou non sur *cart.html*	Problème de communication avec le local storage	Le message appparaît bien
11	cart.js	12 à 83	Affichage de tous les éléments	Affichage dynamique de tous les produits du panier sur la page *cart.html*	Vérification de l'affichage ou non sur *cart.html*	Les éléments n'apparaissent pas du tout, ou alors seulement 1 seul	Tous les produits et élements apparaissent bien
12	cart.js	85 à 101	Fonction de suppression d'un produit (item)	Identidication du produit à supprimer via la méthode .filter Suppression dynamique du produit est refresh de la page	Vérification de l'affichage ou non sur *cart.html* Inspection du local storage	Suppression inactive, ou suppression non désirée d'autres produits (problème d'identification)	La fonction de suppression est efficace
13	cart.js	105 à 122	Get Totals -> Affichage dynamique des totaux	Les totaux de quantité et de prix sont affichées dynamique dans les balises prévues à cet effet	Vérification de l'affichage ou non sur *cart.html* Console.log	Totaux inexacts. Problème de communication avec le DOM ou avec le local storage	Les totaux sont corrects et apparaissent bien
14	cart.js	124 à 144	modifyQty -> Modification dynamique de la quantité	Modification dynamique de la quantité sur *cart.html* Identification du produit concernée et modification dans le local storage. Refresh de la page	Vérification de l'affichage ou non sur *cart.html* Inspection du local storage	Modification inactive, ou modification non désirée d'autres produits (problème d'identification)	La fonction de modification est efficace
15	cart.js	149 à 233	getForm -> Gestion du formulaire	Gestion du formulaire à l'aide de la méthode AddEvestListener, et aussi de la méthode .test pour comparer avec les Regex qui ont été définies	Vérification de l'affichage ou non sur *cart.html* Console.log	Les Regex sont mal définies. Problèmes de communication avec le DOM	Le formulaire est géré correctement
16	cart.js	236 à 267	postForm -> Création de l'order	Ecoute du click sur le bouton "commander !" . Création d'un objet "contact" et d'un autre "products" qui seront rassemblés dans "finalOrder"	Effectuer un *console.log*. Regarder également ce qui apparaît ou est modifié dans le local storage	Problème de communication avec le DOM ou avec le local storage	L'order est bien créé
17	cart.js	269 à 282	postForm -> Envoi de l'order	Méthode Fetch (Post) pour envoyer "finalOrder" vers l'API et récupérer un numéro de commande Redirection vers *confirmation.html*	Effectuer un *console.log*. Regarder également ce qui apparaît ou est modifié dans le local storage	Problèmes de communication avec l'API	Envoi et redirection OK
18	confirmation.js	1 à 2	Récupération de l'ID de la commande via la méthode URLsearchparams	Obtenir la valeur 'ID' de la commande qui a été effectuée	Effectuer un *console.log*	Incapacité à récupérer l'ID	ID récupéré
19	confirmation.js	5 à 9	Affichage de l ID de commande dans le message de remerciement. Effacement du local storage	L'affichage du numéro de commande doit s'afficher dans le message de remerciement. Ensuite, on vide complètement le local storage	Effectuer un contrôle visuel de *confirmation.html* Regarder que plus rien n'apparaît dans le local storage	Problème de communication avec le DOM ou avec le local storage.	Affichage OK Local storage vide