Planification des tests

Présentation

Dans l’objectif d’avoir une application le plus fonctionnel possible, il est important de mettre en place des tests.
Cela permet de vérifier la bonne fonctionnalité de l’application, d’anticiper le comportement en cas d’échec et de prévoir l’évolution de l’application.

Méthodologie

L’application tourne autour de 3 classes : ‘Product’, ‘Cart’ et ‘Validation’.
Le lancement se fait par l’introduction d’un fichier ‘index.js’ pour gérer les pages : ‘index.htm’, ‘product.html’, ‘cart.html’
Le fichier ‘confirm.js’ quant à lui, remet tout à zéro après confirmation du paiement.

Détail

Product.js

constructor : Tester qu’à l’initialisation de la class tous les éléments soit bien initialisé :
L14 : element
L16 : tester le bon fonctionnement de la condition
L28 : tester l’initialisation du panier, class Cart

setUrl : L37 :Tester la modification de ‘this.url’ via la méthode

getProduct : l49 : Tester la réception des données en Json

getUrlParams :
L65 : Tester le retour avec param uri = null
L71 : Tester le retour avec différents paramètres

createProduct :
L84 : Tester la bonne prise en charge de l’url de l’API en fonction de la page
L86 à 100 : Tester la Promise
Tester l’initialisation des fonctions du Panier (Cart)
L102 à 110 : Tester la réaction de la Promise en cas d’échec

getSpecifier : L126 à 136 : Tester la condition
L140 : Tester le retour de la fonctionna

productList : L154 : tester la condition
L224 : Tester le retour de la fonction

removeProduct : L232 : Tester la suppression du bouton

getCart : L241 : Tester le retour de la fonction

Cart.js

constructor : Vérifier l'initialisation des éléments
    L13 : Vérifier la construction du panier dans le Storage

add : L40 : Tester la promise en cas de succès
    L46 : Tester la Promise en cas d'échec
    L57 : Tester la création du bouton supprimer

delete : L68 à 73 : Tester la suppression du produit dans le tableau de produits
    L76 : Tester la modification du Storage
    L78 : tester la condition pur l'affichage du nombre de produit dans le panier
    L84 : Tester la quantité de produit présente dans le panier pour supprimer ou non le bouton « supprimer »

getProduct : Tester le retour en JSON

checkDelteButton : Tester la suppression du bouton si les conditions sont remplis

createDeleteButton : Tester la création du bouton

removeDeleteButton : Tester la suppresion du bouton

listenAddProduct : L155 :Tester l'écoute d'événement au click sur le bouton ajouter

listenDeleteProduct : L171 :Tester l'écoute d'événement au click sur le bouton suprimer

getCart : L190 : Tester le retour de la fonction

viewerCart : tester le paramètres
    L202 et 214 : Tester l'insertion HTML
    L222 : Tester l'écoute au click sur le formulaire
    L224-225 : Tester l'apparition du formulaire
    L228 : tester l'initialisation de la class Validation
total : Tester le paramètres
    L243 : tester le retour de la fonction

Validation.js

constructor : Tester l’initialisation des éléments

post : Tester le post vers l’API
      L34 : tester le retour de la fonction

getValue : L44 : Tester l’écoute lors de l’envoi du formulaire
      L47 : Tester la validation des donnée
      L64 : tester la promesse (Réussite et échec)
      L82 : tester la création d’une nouvelle promesse (échec et réussite)

getProduct : L115 : Tester le Storage
      L136 : tester le retour de la fonctionna

validate : L148 à 151 : Tester les expressions regulières
      L153 à 172 : Tester le retour en cas d’échec

isInvalid : Tester les paramètres
      L184 – 185 : Tester l’insertion des class HTML
      L186 – 187 : Tester l’insertion dans le DOM