Plan de test de l'application web pour le site e-commerce Orinco

Référence

Ce plan de test est établi par rapport aux attentes données par les documents fournis par la société Orinco. Ils suivent les étapes de validation de la commande des articles à travers les différentes pages du site.

Eléments à tester

PAGE D'ACCUEIL

- Affichage des informations envoyées par l'API (nom et image dans une carte)
- Carte de chaque produit clickable avec lien vers la page produit spécifique

PAGE PRODUIT

- Affichage des informations du produit sélectionné, envoyées par l'API (nom, image, description, prix, option dans une carte)
- Affichage des options dans une liste déroulante
- Présence d'un bouton « Commander » qui ajoute le produit au panier

PAGE PANIER

- Contenu du localStorage
- Tableau des produits sélectionnés :
    - 4 colonnes avec les informations envoyées par l'API (référence, image, nom, prix)
    - 1 ligne par produit sélectionné
- Montant total du panier
    - Valeur
    - Affichage
- Nombre d'articles présents dans le panier
    - Affichage : panier vide ou contient n article(s)
- Affichage conditionnel des boutons « Continuer les Achats » et « Vider le panier »
- Affichage du formulaire

PAGE DE CONFIRMATION

- Affichage d'un message de remerciement
- Affichage du montant de la commande
- Affichage de la référence de la commande
- Affichage d'un bouton « Accueil » qui renvoie à la liste des produits présents dans l'API

PIEDS DE PAGE

- Affichage de liens selon la page
  - « Contactez-nous » : ouverture d'une page vide dans un nouvel onglet
  - « Mon panier » : ouverture de la page panier.html
  - « Accueil » : ouverture de la page index.html, contenant la liste des produits fournie par l'API

EN-TETE

- Affichage du logo
  - Lien vers la page index.html, contenant la liste des produits fournie par l'API

Fonctionnalités à tester

RECUPERATION DES DONNEES DE L'API (PAGE D'ACCUEIL)

- Connexion à l'API
- Contenu de l'objet listeProduits

SELECTION D'UN ARTICLE

- Ouverture d'une page dans le même onglet contenant les caractéristiques du produit sélectionné en page d'accueil
- Informations affichées à la sélection du produit par rapport aux données de l'API
- Choix d'une option possible mais qui ne crée aucune action
- Actualisation de la page

GESTION DU PANIER

- Action du bouton « Commander » de la page produit = ajout d'un produit dans le panier
- Ouverture de la page panier
- Ajout d'un produit dans le panier

- Affichage de l'article et de ses informations propres
- Validation du nombre d'articles dans le panier
- Conservation du panier quand on change de page ou qu'on quitte le navigateur
- Validation du montant total
- Action du bouton « vider le panier » (vide le panier et actualise la page)
    - Si le panier est déjà vide
    - S'il contient des produits

VALIDATION DU FORMULAIRE

- Validité des champs selon les contraintes:
    - Décrire chaque contrainte demandée par champ (format du mail, pas de contrainte de longueur sur textes, code postal nombre en 5 chiffres)

Champ du formulaire	Contraintes	Obligatoire
Nom	Chaînes de caractères de 1 à 30 lettres (sans chiffres)	OUI
Prénom	Chaîne de caractères de 1 à 30 lettres (sans chiffres)	OUI
E-Mail	Format mail (contient @)	OUI
Adresse	Chaînes de caractères alphanumériques avec minimum 5 caractères	OUI
Code Postal	Nombre de 5 chiffres	OUI
Ville	Chaîne de caractères de 2 à 30 lettres (sans chiffres)	OUI

- Réponse si non valide (pour chaque champ)
- Action du bouton « Acheter »
- Ouverture page « Confirmation »
    - Envoi du montant global de la commande
    - Envoi des informations saisies dans le formulaire de contact
    - Réception du numéro de référence envoyé par le serveur

VALIDATION DE LA COMMANDE

- Si le panier est vide : la commande ne doit pas passer, un message d'erreur apparaît
- Si le formulaire n'est pas valide : la commande ne doit pas passer, un message d'erreur apparaît sur le 1er champ invalide et le cadre de tous les champs invalides est rouge

PAGE DE CONFIRMATION

- Vérification dans la page de confirmation du montant indiqué de la commande
- Vérification du numéro de référence renvoyé par le serveur
- Action du bouton « Accueil »
    - Ouverture de la page index.html avec la liste des produits de l'API

Méthodologie

OUTIL

On peut utiliser l'outil **JEST**, framework de tests en Javascript qui intègre plusieurs fonctionnalités en un seul package :

- Création des tests
- Assertions
- Mock
- Tests asynchrones
- Tests du DOM

ENVIRONNEMENT

Une fois l'installation de Jest en lui-même, ajout de plusieurs extensions dans l'environnement de travail de façon à couvrir tous les besoins :

- babel pour aider l'environnement à connaître les commandes jest
- fetch-mock pour pouvoir tester les commandes fetch
- jest-dom pour rendre les erreurs de DOM plus lisibles

FONCTIONNEMENT

- Création de fichiers intitulés nomTest.test.js que jest reconnaît directement grâce à : .test.js et qu'il exécute à la commande jest
- Dans ce ou ces fichier(s), on écrit le(s) test(s) à effectuer :

```
test ('nomTest', function(){

…..

expect (ce-que-l-on-teste).toBe(ce-que-l-on-attend)

}
```

- toBe peut être remplacé par un grand nombre de Matchers en fonction du test : toEqual, not.toBe, etc.
- On peut regrouper plusieurs tests dans un seul fichier en utilisant la commande describe ('NomPack', function()){} dans laquelle on met tous les tests avec la possibilité d'intégrer des méthodes pour effectuer des actions avant chaque test par exemple. On peut aussi mettre les tests dans plusieurs fichiers.

- Les tests asynchrones
  - Si les tests sont dans des fichiers séparés ils seront effectués en parallèle
  - Si les tests sont dans le même fichier
    - Ils sont réalisés naturellement les uns après les autres selon l'ordre dans lequel ils sont mis
    - On peut les faire réaliser en parallèle avec la commande test.concurrent('nomTest', function(){}
- Sélection des tests à réaliser
  - test.only pour ne faire que ce test
  - test.skip pour ne pas faire spécifiquement ce test
- Mock
  - Simulation d'appels et contrôle des actions
  - const mock = jest.fn()
  - Si on n'y met rien, il renvoie automatiquement 'undefined' mais on peut lui dire ce qu'elle doit faire :
    - Ex : const mock = jest.fn().mockReturnValue(ce-qui-doit-être-renvoyé)
    - mockReturnValue peut être remplacé par mockReturnThis, mockReturnOnce, mockReolvedValue, etc.
- Fetch
  - Pour tester et mocker la fonction fetch (en ajoutant le module jest-fetch-mock)
- Test du DOM
  - Par exemple pour tester l'affichage d'un contenu après un click