Développement d'une Application Web E-commerce (SPA) en HTML, CSS et JavaScript

Objectif du projet

Ce projet a pour objectif de concevoir une **application monopage (Single Page Application - SPA)** e-commerce fonctionnelle, en utilisant uniquement **HTML, CSS et JavaScript** sans recourir à des frameworks externes. L'étudiant devra structurer l'application selon le modèle **MVC (Modèle-Vue-Contrôleur)**, gérer la **navigation côté client à l'aide de la History API**, **interagir avec une API REST** via `fetch` et `async/await`, et **gérer l'état de l'utilisateur** à l'aide de `sessionStorage` et `localStorage`.

Contexte

Vous êtes chargé de développer une boutique en ligne moderne et dynamique, destinée à un public adulte. Cette boutique doit être conçue comme une SPA fluide, rapide et modulaire. L'architecture doit être organisée en **composants HTML/CSS/JS réutilisables**, et l'ensemble doit offrir une expérience utilisateur fluide, accessible sur tous types de terminaux.

Travail demandé

Vous devez développer une application SPA avec les éléments suivants :

Architecture générale

- Application structurée en composants : chaque section importante (header, footer, carte produit, formulaire, etc.) doit être isolée dans un fichier JS/HTML/CSS dédié.
- Utilisation du **pattern MVC** pour organiser la logique métier, l'affichage, et les interactions utilisateur.

Fonctionnalités principales

- **Routage côté client** :
  - Implémentation du routage avec la **History API** pour charger dynamiquement les vues (Accueil, Produits, Connexion, Contact) sans recharger la page.
- **Chargement dynamique des vues** :
  - Le contenu principal est injecté via JavaScript selon l'URL (ex: `/produits`, `/connexion`).

- **Affichage dynamique des produits** :
  - Récupération des produits à partir d'une **API REST fictive** (JSON local ou JSON Server).
  - Utilisation de `fetch` avec `async/await` et gestion des erreurs.
- **Connexion/Inscription utilisateur** :
  - Formulaire d'authentification avec stockage de session (`sessionStorage`).
  - Interface différente selon si l'utilisateur est connecté ou non.
- **Formulaire de contact** :
  - Validation côté client.
  - Possibilité d'affichage conditionnel d'un message de confirmation.
- **Panier** :
  - Ajout de produits à un panier (stocké dans `localStorage`).
  - Affichage du contenu du panier en temps réel.

Structure suggérée

- `index.html` (structure racine minimale avec une `div#app`).
- `app.js` : point d'entrée JavaScript qui contrôle le routage et le rendu dynamique.
- `router.js` : fichier gérant le routage avec la History API.
- `controllers/` : contient la logique métier par vue.
- `views/` : contient les fonctions générant dynamiquement le HTML.
- `models/` : contient la gestion des données (produits, utilisateur, panier).
- `components/` : composants réutilisables (carte produit, formulaire, header, etc.).

Critères d'évaluation

Les étudiants seront évalués selon les critères suivants :

- **Architecture SPA** : Respect de la séparation des responsabilités (MVC), modularité du code.
- **Routage client** : Utilisation efficace de la History API, navigation fluide.
- **Intégration API REST** : Bonne utilisation de `fetch`, `async/await`, gestion des erreurs.
- **Gestion de l'état** : Manipulation pertinente de `sessionStorage` et `localStorage`.
- **Qualité du code** : HTML sémantique, CSS bien organisé (Flexbox, Grid), JavaScript clair et structuré.
- **Expérience utilisateur** : Design réactif, ergonomie, accessibilité.
- **Interactivité** : Validation des formulaires, affichage dynamique, feedback utilisateur.

Livrables

- Un dossier complet contenant :
  - Le code source organisé (HTML/CSS/JS).
  - Un fichier `README.md` expliquant la structure du projet, les choix techniques, et les instructions de lancement.
- Un fichier `data/produits.json` (ou équivalent API simulée).
- Une démonstration du projet ou une vidéo de présentation (si demandé par l'enseignant).

NB :

- Ce projet est individuel et compte pour **25 %** de la note finale.
- Les fichiers doivent être déposés dans Léa au plus tard **le 09 juillet 2025**.
- Chaque étudiant aura 5 min pour simplement faire la démo de son projet au dernier cours qui aura lieu le **11 juillet 2025** et qui compte sur la note du projet.