

# ÉNONCÉ DE PROJET – Mini Boutique en Ligne Inspirée de Jumia

Vous travaillez en groupe pour concevoir **une mini-boutique en ligne** inspirée des fonctionnalités essentielles de Jumia.

Le projet se déroule en trois grandes parties :

1. **UI/UX** (personas → zoning → wireframes → mockups + règles UI)
2. **Spécifications techniques** (UML)
3. **Développement d'un MVP React**
4. **Workflow Git/GitHub** (obligatoire pour toute fonctionnalité)

Chaque équipe devra produire les livrables demandés et respecter la méthodologie Git à travers des **branches par fonctionnalité**, des **Pull Requests**, et un **merge workflow propre**.

## Objectif final

Construire un MVP fonctionnel permettant :

- Navigation entre pages (Home, Product, Cart, Checkout),
- Recherche et filtres produits,
- Gestion du panier (useContext + useReducer),
- Interface responsive (mobile/desktop),
- Organisation en composants réutilisables.

## PARTIE 1 — UI/UX Design

### Question 1 – Création de Personas

Créez **trois personas** représentatifs des utilisateurs potentiels (ex : acheteur occasionnel, client fidèle, administrateur produit).

Chaque persona doit inclure :

**Nom, âge, objectif principal, frustrations, besoins, scénarios d'usage.**

### Question 2 – Zoning & Architecture des pages

Réalisez **les zonings** des pages suivantes en mobile + desktop :

- Page d'accueil (Home)
- Page Catégorie / Liste produits
- Page Produit
- Page Panier
- Page Checkout

## Question 3 – Wireframes Low-Fidelity

Produisez des **wireframes monochromes** (papier, Figma ou autre).

Ils doivent montrer les éléments principaux : navbar, search, filtres, cards produits, CTA “Add to cart”, etc.

## Question 4 – Mockups High-Fidelity

Créez au moins **3 écrans finalisés** (UI complète) :

- Home
- Product
- Cart

Palette, typographie et composants doivent être inspirés de Jumia.

## Question 5 – Application des lois UI

Expliquez comment vous avez appliqué **les lois suivantes** dans votre design :

- Fitts
- Hick
- Miller
- Gestalt
- Jakob
- Hiérarchie visuelle

# PARTIE 2 — UML

## Question 6 – Diagramme de Cas d’Usage (Use Cases)

Créez un diagramme incluant au minimum :

- Visiteur
- Client
- Admin

Avec des cas : rechercher, filtrer, consulter produit, ajouter au panier, checkout, gérer produits.

## Question 7 – Diagramme de Classes

Modélisez les entités essentielles :

Product, Category, User, Cart, CartItem, Order, Payment.

## Question 8 – Diagramme d'Activité

Décrivez le **processus de Checkout**.

## Question 9 – Diagramme de Séquence

Décrivez la séquence **"Ajouter un produit au panier"**.

## Question 10 – Diagramme d'États

Modélisez les états d'une **commande** : Draft → Paid → Shipped → Delivered → Cancelled.

# PARTIE 3 — Développement MVP React

### Contraintes techniques :

- Utiliser **React + react-router-dom**
- Gestion du panier via **useContext + useReducer**
- Search + filtres
- Composants réutilisables
- Style libre (CSS, Tailwind, etc.)

### IMPORTANT — RÈGLE GIT OBLIGATOIRE :

- **Chaque fonctionnalité doit être développée dans une branche dédiée.**
- **Aucune fonctionnalité ne doit être poussée directement dans main.**
- **Chaque étudiant doit créer AU MOINS une Pull Request.**

## Question 11 – Initialisation du projet

- Créez un repo GitHub par groupe.
- Configurez les branches : main (production) + develop (optionnel).
- Créez la structure React avec Vite ou Create React App.

## Question 12 – Fonctionnalité 1 : Navbar + Routing

Créer une branche :feature/navbar-routing

À faire :

- Navbar (logo, search input, cart icon)
- Définir les routes : /, /product/:id, /cart, /checkout
- Créer les pages vides correspondantes

Pull Request → merge.

## Question 13 – Fonctionnalité 2 : Product List + ProductCard

Créer une branche :feature/product-list

À faire :

- Charger un fichier JSON de produits ou mock API
- Afficher une grille de ProductCard
- Cliquer sur une card → page produit

Pull Request → merge.

## Question 14 – Fonctionnalité 3 : Search

Créer une branche :feature/search-bar

À faire :

- Récupérer la query depuis URL (useSearchParams)
- Filtrer les produits
- Update URL lorsqu'on cherche

Pull Request → merge.

## Question 15 – Fonctionnalité 4 : Filtres

Créer une branche :feature/filters

À faire :

- Filtres par catégorie, prix, rating
- Ajouter les filtres dans l'URL
- Affichage dynamique

Pull Request → merge.

## Question 16 – Fonctionnalité 5 : Panier (useContext + useReducer)

Créer une branche :feature/cart

À faire :

- Créer CartContext + reducer

- Ajouter un produit
- Modifier quantité
- Supprimer un produit
- Calcul du total

Pull Request → merge.

## Question 17 – Fonctionnalité 6 : Page Checkout

Créer une branche : feature/checkout

À faire :

- Récapitulatif panier
- Formulaire adresse
- Confirmation de commande (mock)

Pull Request → merge.

## Question 18 – Fonctionnalité 7 : Responsive

Créer une branche :feature/responsive-ui

Responsive obligatoire : mobile, tablette, desktop.

Pull Request → merge.

# PARTIE 4 — Workflow Git obligatoire

## Question 19 – Création de branches

Pour chaque fonctionnalité, montrez :

- la branche créée
- commits associés
- Pull Request correspondante

## Question 20 – Pull Request & Code Review

Documentez :

- 1 PR envoyée par chaque étudiant
- 1 review réalisée par chaque étudiant
- captures d'écran des merges

## Question 21 – Synchronisation avec main (merge upstream)

Chaque groupe devra montrer sa gestion de :

- git fetch
- git pull
- merges propres (pas de conflits laissés ouverts)

## LIVRABLES FINAUX

À remettre :

1. **PDF UI/UX** (personas → zoning → wireframes → mockups + lois UI)
2. **PDF UML** (tous diagrammes)
3. **Lien GitHub du projet React**
4. **Liste des branches créées + PRs réalisées**
5. **Lien vers la version déployée (Netlify, Vercel, etc.)**