

Plateforme e-learning — Conception UI/UX, Roadmap & UML

Document professionnel : zoning → wireframes → mockups → prototype
React.js + roadmap détaillé + diagrammes UML (use case, séquence, classe, état/transition).

1. Résumé exécutif

Créer une plateforme e-learning centrée sur l'utilisateur qui présente une **liste de cours en Card List**. Chaque carte contient : titre, image, courte description, tags, rating, bouton "**Afficher cours**" (page course detail) et bouton "**Passer quiz**" (redirection vers page quiz). Ce document décrit le processus complet de conception UI/UX, un plan de développement (roadmap) prêt pour une équipe, et les diagrammes UML nécessaires pour la partie analyse et conception.

2. Objectifs produit

- Permettre à l'apprenant de parcourir, visualiser et suivre des cours.
- Offrir un parcours clair pour accéder au contenu et évaluer les acquis via quiz.
- Optimiser la conversion inscription → début de cours.
- Interface responsive (mobile-first) et accessible (WCAG AA).

3. Utilisateurs cibles (personas)

1. **Étudiant(e) lycéen(ne)** — recherche orientation pratique, veut tests rapides.
2. **Professionnel(le) en reconversion** — recherche parcours certifiant, filtrage par compétences.
3. **Formateur / Admin** — publie cours, crée quiz, suit statistiques.

Pour chaque persona : besoins, frustrations, scénarios principaux (par ex. "trouver un cours et passer le quiz en ≤ 5 minutes").

4. Zoning (architecture de l'information)

Pages principales :

- Homepage (liste de cours, filtres)
- Course List (cards)

- Course Detail (contenu, syllabus, bouton Start Course)
- Quiz Page (questionnaire, timer, navigation)
- Profile / Dashboard (mes cours, résultats)
- Admin Panel (gestion cours/quiz)
- Auth (Login/Register/Forgot)

Composants réutilisables : CardCourse, Navbar, Footer, FiltersPanel, Pagination, ModalConfirm, RatingStars, ProgressBar.

5. User Flows clés

1. **Découvrir & Ouvrir cours** : Homepage → Cliquer Card Afficher cours → Course Detail → Start → Lecture
2. **Passer quiz** : Card → Cliquer Passer quiz → Quiz Page → Répondre → Résultats
3. **S'inscrire** : Accès à contenu restreint → Redirection vers Auth → Retour au cours

Chaque flow doit inclure états d'erreur (connexion requise, questions non répondues, timeout) et messages UX.

6. Wireframes (structure basse fidélité)

Pour chaque page, wireframe simple :

- **Card List (grid)** : header (recherche + filtre), colonnes grid 1/2/3 selon breakpoint, chaque card : image en haut, titre, tags, courte description (2 lignes), row boutons (Afficher cours, Passer quiz), footer card (rating + durée).
- **Quiz Page** : top bar (progression, timer), zone question (texte + supports), réponses (radio/checkbox), boutons Précédent / Suivant, Soumettre (final).

(Fournir ces wireframes au format PNG/Sketch/Figma — voir prototype)

7. Mockups (haute fidélité) — recommandations visuelles

Design system minimal :

- Typographie : Inter / Roboto (body), headings semi-bold
- Palette : Primary (#0066CC), Secondary (#00A676), Neutral greys (pour surfaces)

- Spacing : 8pt grid
- Buttons : Primary filled, Secondary outline
- Cards : ombre légère, coins 12px

Accessibilité : contrast ratio $\geq 4.5:1$ pour textes normaux, focus visible, labels explicites.

Micro-interactions : hover sur cards, animation légère lors de la soumission du quiz, transition de page fluide.

8. Prototype technique (React.js) — structure et composants

Stack recommandé : React (Vite), React Router, Zustand / Redux Toolkit (state global), Tailwind CSS (rapidité), react-hook-form (forms), Jest + React Testing Library.

Arborescence proposée :

```
/src
  /components
    CardCourse.jsx
    FiltersPanel.jsx
    QuizQuestion.jsx
    ProgressBar.jsx
    Navbar.jsx
  /pages
    Home.jsx
    CourseDetail.jsx
    Quiz.jsx
    Profile.jsx
    Admin.jsx
  /routes
    AppRouter.jsx
  /store
    courseSlice.js
    authSlice.js
    quizSlice.js
  /utils
  /assets
  main.jsx
```

Exemple CardCourse (concept) :

```
// CardCourse.jsx (structure)
return (
  <article className="card">
    <img src={course.image} alt="" />
    <h3>{course.title}</h3>
    <p className="desc">{truncate(course.desc, 100)}</p>
    <div className="actions">
      <Link to={` /course/${course.id}`}>Afficher cours</Link>
      <Link to={` /quiz/${course.id}`}>Passer quiz</Link>
    </div>
  </article>
)
```

Quiz Page : charger les questions via store, gérer navigation entre questions, sauvegarde locale (localStorage) et timeout. Afficher score et feedback.

11. Diagrammes UML (texte & mermaid) — base pour la documentation

- 11.1 Use Case (acteurs) — mermaid
- 11.2 Sequence — parcours "Passer quiz"
- 11.3 Class diagram (simplifié)
- 11.4 State Machine — état d'un quiz
- 11.5 Activité : Lire cours , passer Quiz , S'inscrire

12. Livrables fournis (exemple)

- Figma : wireframes + mockups + UI kit
- Repo Git : starter React app, composants de base
- UML : fichiers mermaid/PNG
- Plan tests et checklist d'accessibilité