Identifier le besoin de l'utilisateur

1. Objectifs pédagogiques

- Définir ce qu'est un **besoin utilisateur** en UI/UX Design et pourquoi il est au cœur de tout projet web
- Distinguer un besoin d'une solution technique (le piège classique du développeur)
- Utiliser la formule standard pour écrire un besoin clair et testable
- Mettre en œuvre des techniques de recherche (entretiens, observation, analytics) pour découvrir les besoins
- Synthétiser les données grâce aux Personas et à la Carte d'empathie
- Traduire un besoin en **exigences fonctionnelles et non fonctionnelles** exploitables en développement web
- Relier un besoin à des choix concrets de code (HTML sémantique, ARIA, performance, responsive)

2. Définition : qu'est-ce qu'un besoin utilisateur ?

Un **besoin utilisateur** est un problème ou un objectif qu'une personne cherche à résoudre ou à atteindre dans un contexte donné. En UX, il ne s'agit pas de ce que l'utilisateur *veut* (une fonctionnalité précise), mais de ce qu'il *doit accomplir* pour améliorer sa situation.

La distinction est fondamentale : un utilisateur peut dire « je veux un bouton rouge », mais son besoin réel est peut-être « je veux trouver rapidement comment finaliser ma commande ». Se concentrer sur le besoin permet de concevoir la **bonne solution**, et non simplement d'exécuter une demande littérale qui pourrait être inefficace.

En développement web, ce piège est fréquent : un client demande « un carrousel en page d'accueil ». Le développeur qui code le carrousel répond à la demande. Le développeur qui identifie le besoin (« *mettre en avant les 3 offres phares pour augmenter les conversions* ») peut proposer une solution plus performante : une grille de 3 cartes visibles sans JavaScript, meilleure pour le SEO et l'accessibilité.

La recherche utilisateur permet de dépasser les demandes superficielles pour identifier les besoins profonds.

apcpedagogie.com

1. ## La formule magique pour écrire un besoin

Pour formaliser un besoin sans imposer de solution, les équipes UX utilisent une structure standardisée. Cette formule force à clarifier **qui** est l'utilisateur, **ce qu'il cherche à faire**, et **pourquoi** il veut le faire.

> **Structure :**
>
> `En tant que [type d'utilisateur],`
> `je veux [action/objectif],`
> `afin de [bénéfice/résultat].`

Exemple concret (secteur bancaire) :

- ❌ *Mauvais besoin (orienté solution) :* « Je veux un bouton « virement rapide » sur l'écran d'accueil. »
- ✅ *Bon besoin (orienté problème) :* « En tant que client pressé, je veux effectuer un virement en moins de 30 secondes, afin de ne pas être en retard à mon rendez-vous. »

Le deuxième énoncé laisse l'équipe de design libre de proposer la meilleure solution (un bouton, un raccourci, une automatisation, etc.).

Exemple concret (projet web e-commerce) :

- 📥 *Demande client :* « Ajoutez une popup d'inscription à la newsletter dès l'arrivée sur le site. »
- 🔄 *Besoin reformulé :* « En tant que visiteur hésitant, je veux comprendre la valeur de la newsletter avant de laisser mon email, afin de décider en connaissance de cause. »
- 🛠️ *Traduction en exigences web :* bandeau discret non bloquant, champ email avec validation HTML5 (`type="email"` + `required`), message de confirmation accessible (ARIA `aria-live="polite"`), absence de modale bloquante (bon pour le Core Web Vitals et l'accessibilité).

- ## Les méthodes pour découvrir les besoins

apcpedagogie.com

On ne devine pas les besoins : on les **découvre** grâce à des méthodes de recherche qualitative.

1. L'entretien semi-structuré

C'est la méthode reine. Contrairement à un questionnaire fermé, l'entretien semi-structuré permet de creuser les réponses. On prépare un guide de questions ouvertes, mais on laisse l'utilisateur dériver vers ce qui lui importe vraiment. L'objectif est de comprendre son **contexte**, ses **frustrations** et ses **astuces actuelles** (workarounds).

Exemples de questions ouvertes à poser :

- « Racontez-moi la dernière fois où vous avez fait [tâche]. Comment cela s'est-il passé ? »
- « Qu'est-ce qui vous a le plus frustré dans cette expérience ? »
- « Comment contournez-vous ce problème aujourd'hui ? »
- « Sur quel appareil étiez-vous ? Étiez-vous pressé ? » (contexte d'usage web : mobile, connexion lente, etc.)

2. L'observation in situ

Parfois, les utilisateurs disent une chose et en font une autre. L'observation consiste à regarder l'utilisateur accomplir sa tâche dans son environnement réel. Cela révèle des besoins **implicites** que l'utilisateur lui-même ne verbalise pas.

Exemple web : vous observez un utilisateur remplir un formulaire d'inscription. Vous remarquez qu'il clique fréquemment sur le bouton « Retour » du navigateur parce que l'étape 2 ne lui permet pas de modifier son email saisi à l'étape 1. Le besoin implicite : « *pouvoir corriger une information précédente sans perdre la progression* ». En code, cela se traduit par un state management côté client (localStorage ou état React) conservant les données entre les étapes.

3. L'analyse des données existantes

Avant même de parler à un utilisateur, on peut examiner les traces laissées par les utilisateurs actuels : analytics (taux de rebond, pages les plus visitées), tickets de support, avis clients, ou rapports de recherche antérieurs.

Exemple concret : Google Analytics montre que 70 % des visiteurs quittent la page produit sans scroller sous la ligne de flottaison. Le besoin identifié : *« voir l'essentiel (prix,*

apcpedagogie.com

disponibilité, bouton d'achat) sans avoir à scroller ». Cela se traduit en UI par un repositionnement du CTA au-dessus de la ligne de flottaison, et en code par un CSS `display: grid` avec les éléments clés en `order: -1` sur mobile.

4. Les tests utilisateurs sur prototype ou site existant

On confronte un prototype (Figma) ou un site déjà en ligne à de vrais utilisateurs. On leur donne une mission (« Trouvez le prix de ce produit et ajoutez-le au panier ») et on observe. En développement web, cela peut se faire sur un site en production avec des outils comme Hotjar (heatmaps, enregistrements de sessions) ou Maze (tests à distance sur maquettes cliquables).

5. Les questionnaires quantitatifs

Utiles pour valider une hypothèse à grande échelle (par exemple avec Google Forms ou Typeform). Ils ne remplacent pas les entretiens pour découvrir un besoin, mais permettent de mesurer sa fréquence. Exemple : « Sur une échelle de 1 à 5, à quel point trouvez-vous difficile de retrouver une commande passée ? »

L'entretien individuel permet de comprendre le « pourquoi » derrière les comportements.

1. Synthétiser les besoins : Personas et Carte d'empathie

Une fois les données collectées, il faut les organiser pour qu'elles guident la conception. Deux outils complémentaires sont essentiels.

Le Persona

Le persona est une **représentation fictive mais réaliste** de votre utilisateur cible. Il synthétise ses objectifs, ses comportements, ses compétences et son contexte. Attention : on ne crée pas plus de 3 à 5 personas pour rester efficace. Le persona permet à toute l'équipe (développeurs, marketing, design) de se mettre d'accord sur « pour qui on travaille ».

Exemple de persona web :

> **Sophie, 34 ans, responsable RH dans une PME**

apcpedagogie.com

- **Objectif :** trouver rapidement des candidats qualifiés sur une plateforme de recrutement.
- **Contexte :** utilise son smartphone dans les transports, connexion 4G instable.
- **Compétence numérique :** moyenne (sait utiliser les sites, pas les outils techniques).
- **Frustration :** les formulaires trop longs sur mobile, les filtres qui se réinitialisent.
- **Besoin clé :** filtrer les candidats en 2 clics et sauvegarder sa recherche.
- **Conséquence en développement web :** filtres en AJAX sans rechargement, persistance des filtres en URL (query params), design mobile-first, formulaires courts avec sauvegarde automatique.

La Carte d'empathie

La carte d'empathie approfondit un persona en explorant quatre dimensions (les 4 quadrants) :

- **Ce qu'il Dit :** citations directes issues des entretiens. Ex. : « Je perds toujours mes filtres quand je reviens sur le site. »
- **Ce qu'il Pense :** ses croyances et préoccupations non verbalisées. Ex. : « Ce site n'est pas fait pour les gens pressés comme moi. »
- **Ce qu'il Fait :** ses actions observables et ses contournements. Ex. : il ouvre 5 onglets pour comparer, il screenshot les résultats.
- **Ce qu'il Ressent :** ses émotions. Ex. : frustration, perte de confiance, sentiment d'inefficacité.

On y ajoute souvent les **Pains** (obstacles) et les **Gains** (bénéfices attendus) pour transformer directement ces insights en opportunités de design et en tickets de développement.

La carte d'empathie organise les insights pour créer une compréhension partagée de l'utilisateur.

1. Du besoin à l'exigence : le passage au développement web

apcpedagogie.com

Identifier le besoin ne suffit pas : il faut le **traduire en exigences** que l'équipe de développement peut implémenter et tester. On distingue deux familles.

Les exigences fonctionnelles

Elles décrivent **ce que le système doit faire**.

- Le formulaire doit valider l'email côté client avant l'envoi.
- L'utilisateur doit pouvoir sauvegarder ses filtres de recherche.
- La page doit afficher un message de confirmation après l'ajout au panier.

Les exigences non fonctionnelles

Elles décrivent **comment le système doit se comporter**: performance, accessibilité, responsive, sécurité. Ce sont souvent elles qui font la différence entre un site « qui marche » et un site **agréable à utiliser**.

- **Performance :** le Largest Contentful Paint (LCP) doit être inférieur à 2,5 s sur mobile 4G.
- **Accessibilité :** le contraste texte/fond doit respecter le ratio WCAG AA (4,5:1 minimum).
- **Responsive :** la mise en page doit être utilisable dès 320 px de large.
- **Sémantique :** chaque formulaire doit avoir des `` associés et des messages d'erreur annoncés par lecteur d'écran (`aria-live`).

Exemple de traduction complète besoin → code :

> **Besoin :** « En tant qu'utilisateur malvoyant, je veux savoir immédiatement si mon mot de passe est valide, afin de ne pas rester bloqué sur un formulaire. »
>
> **Traduction technique :**
>
> - Champ `` avec `` explicite.
> - Message d'erreur dans un conteneur `role="alert"` ou `aria-live="assertive"`.
> - Indication visuelle ET textuelle (pas seulement un changement de couleur : WCAG 1.4.1).
> - Attribut `aria-describedby` reliant le champ à son message d'aide.

apcpedagogie.com

1. Activités pratiques

Activité 1 : Reformuler une demande en besoin (10 min)

Pour chaque demande client ci-dessous, écrivez le besoin réel avec la formule « En tant que... je veux... afin de... ».

1. « Mettez un slider automatique en haut de la page d'accueil. »
2. « Ajoutez un chatbot sur toutes les pages. »
3. « Le bouton d'achat doit être vert fluo. »
4. « Faites une page de contact avec 12 champs obligatoires. »

Piste de correction : pour la 1, le besoin probable est « mettre en avant plusieurs offres sans que l'utilisateur ait à chercher ». Pour la 4, le besoin est peut-être « qualifier les demandes entrantes », ce qui peut se faire avec moins de champs et un formulaire progressif.

Activité 2 : Audit d'un site existant (20 min)

Choisissez un site que vous utilisez régulièrement (banque, e-commerce, université). Réalisez un mini-audit en 3 étapes :

1. **Observation :** notez 3 frictions que vous rencontrez (bouton mal placé, formulaire trop long, contraste insuffisant).
2. **Besoin :** reformulez chaque friction en besoin utilisateur.
3. **Piste technique :** proposez pour chaque besoin une amélioration concrète en HTML/CSS/JS (ex. : remplacer un `
` par un `

Activité 3 : Construire un persona à partir d'entretiens (30 min)

En binôme, réalisez un entretien de 10 minutes : l'un joue l'utilisateur, l'autre le chercheur. Le chercheur pose au moins 5 questions ouvertes sur une tâche web précise (ex. : « la dernière fois que vous avez acheté un billet de train en ligne »). Puis rédigez un persona d'une demi-page avec : objectif, contexte, frustration, besoin clé, et 2 conséquences techniques pour le développement.

Activité 4 : Carte d'empathie et tickets de développement (20 min)

apcpedagogie.com

À partir du persona créé à l'activité 3, remplissez les 4 quadrants de la carte d'empathie (Dit / Pense / Fait / Ressent), puis listez 3 Pains et 2 Gains. Enfin, transformez chaque Pain en un **ticket de développement** formulé ainsi : « En tant que [persona], je veux [action], afin de [bénéfice]. Critères d'acceptation : [critères testables]. »

Activité 5 : Test utilisateur express (15 min)

Prenez un site web quelconque. Donnez à un camarade une mission précise (« Trouve le prix du produit X et mets-le dans le panier »). Observez sans aider. Notez : où il hésite, ce qu'il clique en premier, ce qu'il dit à voix haute. Déduisez-en 2 besoins non satisfaits et 2 améliorations UI concrètes.

1. Ce qu'il faut retenir

- Un besoin est un **problème à résoudre**, pas une fonctionnalité à coder.
- La formule **« En tant que... je veux... afin de... »** aide à garder le cap et à éviter le biais de solution.
- Les besoins se découvrent par la **recherche** (entretiens, observation, analytics, tests), pas par l'intuition seule.
- Les **Personas** et la **Carte d'empathie** sont les ponts entre la recherche brute et la conception UI/UX.
- Un besoin identifié doit être traduit en **exigences fonctionnelles et non fonctionnelles** testables par l'équipe de développement.
- En développement web, cela se traduit concrètement par : HTML sémantique, attributs ARIA, performance (Core Web Vitals), responsive mobile-first, et validation de formulaires accessible.
- Un bon développeur UI/UX ne demande pas seulement « quoi coder », mais **« pour qui et pourquoi »**.

Source : https://apcpedagogie.com/identifier-le-besoin-de-lutilisateur · téléchargé le 05/11/2026 · © APC Pédagogie · apcpedagogie.com