



CAHIER DES CHARGES

Méthodologie de réalisation du cahier des charges



LES OBJECTIFS

Comprendre le rôle et les enjeux du cahier des charges

Savoir structurer un cahier des charges complet, en distinguant les parties fonctionnelles, techniques et organisationnelles.

Être capable de formaliser les besoins d'un client sous forme claire et exploitable par une équipe technique.

Maîtriser les bonnes pratiques de rédaction pour produire un document lisible, précis et adapté à tous les interlocuteurs du projet.





DEFINITION ET USAGES

QU'EST CE QU'UN CAHIER DES CHARGES (CDC)

Le cahier des charges est un document contractuel qui définit précisément :

- Ce qui **doit être fait**,
- Par **qui**,
- **Pourquoi**,
- Et dans quelles **conditions**

Il sert de référence commune entre un **client** (commanditaire) et un **prestataire** (équipe, freelance, agence...).

OBJECTIFS DU CDC

Un bon CDC permet de :

- **Fixer** clairement les **besoins** et **attentes**
- **Éviter les malentendus**
- **Planifier** les tâches et les responsabilités
- **Estimer** les délais et les coûts
- Suivre l'**avancement** du projet

CAHIER DES CHARGES

Exprimez le besoin et le périmètre du projet

Contexte du projet

Contexte du projet

[Situez ici le projet dans son contexte.

Décrivez comment ce projet est né, les besoins initiaux de l'entreprise, les avantages ou bénéfices espérés, le contexte technique, les aspects légaux si applicable. Une personne externe à l'entreprise comprendra clairement le Pourquoi de votre initiative.]

Exemple :

Notre Centre de Montage Régional Lillois est la dernière entité du groupe à n'avoir pas encore installé le système SAP.

Il produit des sièges pour nos clients constructeurs automobiles.

Les solutions informatiques existantes sont dépassées, coûteuses et ne sont plus maintenues.

De plus, les processus d'intégration avec les systèmes du siège sont principalement manuels et génératrices d'erreurs.

Les procédures de clôture mensuelles sont longues et souvent incorrectes, ce qui génère du travail supplémentaire.

Enfin, le manque de flexibilité des solutions actuelles nous empêche la mise en place des applications nécessaires à notre stratégie d'expansion.

Pour toutes ces raisons, le comité de Direction a approuvé le passage au systèmes SAP du groupe avec un démarrage du projet au plus tôt.

POURQUOI IL EST SI IMPORTANT

- Un besoin mal formulé revient à avancer à l'aveugle : **sans repères clairs, difficile d'atteindre les objectifs** du projet.
- La gestion de projet ne laisse **pas de place à l'approximation** : elle exige rigueur, précision et clarté à chaque étape.
- Le cahier des charges métier **lève les zones d'ombre** : il **force à expliciter les concepts** flous et à clarifier les attentes implicites.
- **C'est un outil structurant** : il canalise les réflexions des parties prenantes, encadre leurs actions et sert de **référence commune** tout au long du projet.
- **L'absence de formulation** précise du besoin **favorise les dérives de périmètre** et met en péril les délais, le budget ou les livrables.

QUAND LE REDIGER

Le cahier des charges est rédigé en début de **phase planification**.

En effet, à ce stade, vous avez rencontré les bénéficiaires du projet et documenté l'expression des besoins.

Vous avez donc maintenant une **bonne idée des attentes fonctionnelles** ainsi que de certaines **contraintes, exigences**, et **risques identifiés** lors des réunions et interviews.

Cette expression des besoins est la base du périmètre détaillé.



LA STRUCTURE DU CAHIER DES CHARGES

1. PRESENTATION GENERALE

1.1 Contexte du projet

Expliquer pourquoi le projet est lancé (problème, opportunité, client...).

1.2 Objectifs

Définir ce que le projet vise à accomplir (résultats attendus).

1.3 Parties prenantes

Qui est concerné ? (client, utilisateurs finaux, prestataires...)

1.1 CONTEXTE

Cette section explique l'origine du projet. Elle permet à toutes les parties prenantes de comprendre pourquoi ce projet existe.

Le contexte peut relever de :

- Un problème à résoudre :
 - Exemple : Le site e-commerce actuel ne s'adapte pas aux mobiles, ce qui fait perdre des ventes.
- Une opportunité à saisir :
 - Exemple : Le marché des plantes d'intérieur est en forte croissance, et peu d'acteurs locaux sont présents en ligne.
- Une commande client ou une initiative interne :
 - Exemple : Le client souhaite digitaliser son activité de conseil en nutrition via une application mobile.

But : Ancrer le projet dans la réalité, montrer les enjeux, et créer du sens autour de son lancement.



1.2 OBJECTIFS

Les objectifs définissent ce que le projet cherche à accomplir. Ils peuvent être :

- **Fonctionnels** : ce que l'outil ou la solution doit permettre de faire.
- Ex. : Permettre aux utilisateurs de réserver une séance en ligne en moins de 3 clics.
- **Business** : ce que l'organisation espère en retirer.
- Ex. : Augmenter de 20 % les ventes en ligne en 6 mois.
- **Qualitatifs** ou **quantitatifs** : accessibilité, performance, taux de conversion, gain de temps, etc.

On peut aussi distinguer :

- Les objectifs principaux (prioritaires),
- Les objectifs secondaires (bonus, options, évolutifs...).



1.3 PARTIES PRENANTES

Cette section identifie tous les acteurs impliqués dans le projet, avec leurs rôles et intérêts.



Exemples de parties prenantes :

- **Client commanditaire** : finance et valide le projet (ex. : direction générale).
- **Utilisateurs finaux** : ceux qui utiliseront réellement l'outil au quotidien (ex. : clients, salariés...).
- **Chef de projet** : assure la coordination globale.
- **Équipe technique** : développeurs, designers, testeurs...
- **Prestataires externes** : agence, freelance, consultant UX...
- **Administrateurs ou support** : ceux qui géreront la solution après mise en ligne.

2. PERIMETRE DU PROJET

2.1 Fonctionnalités attendues

- Liste précise des fonctionnalités, souvent sous forme de user stories ou bullet points.

2.2 Contraintes

- Techniques, réglementaires, budgétaires, organisationnelles.

2.3 Exclusions (hors-périmètre)

- Ce qui n'est pas inclus dans le projet (important pour éviter les malentendus).

2.1 FONCTIONNALITES ATTENDUES

Cette section décrit ce que le système devra permettre de faire. Elle constitue le cœur du besoin fonctionnel.



Formes possibles :

- Liste à puces des grandes fonctionnalités.
- User stories : "En tant que [type d'utilisateur], je veux [fonctionnalité] afin de [bénéfice]".
- Diagrammes de cas d'usage (en complément visuel).

Exemples de fonctionnalités pour un site e-commerce :

- Création d'un compte utilisateur.
- Ajout d'un produit au panier.
- Paiement en ligne via carte bancaire.
- Accès à l'historique des commandes.
- Interface d'administration pour gérer les produits.

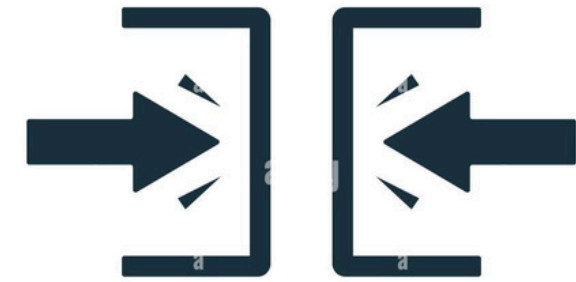
Objectif : poser les bases du développement en cadrant le "quoi", sans encore dire "comment"

2.2 CONTRAINTES

Les contraintes sont des règles à respecter obligatoirement. Elles influencent la manière de concevoir, développer ou déployer le projet.

Types de contraintes :

- **Techniques :**
 - Utilisation obligatoire d'un CMS spécifique (ex : WordPress).
 - Compatibilité mobile/tablette (responsive design).
 - Langage imposé (ex : JavaScript, Python).
- **Réglementaires :**
 - Respect du RGPD (gestion des cookies, consentement, droit à l'oubli).
 - Accessibilité (ex : conformité WCAG pour un site public).
- **Budgétaires :**
 - Budget maximum alloué au développement.
 - Limitation du nombre de jours/homme.
- **Organisationnelles :**
 - Échéances imposées (ex : lancement prévu pour une date événementielle).
 - Disponibilités restreintes des parties prenantes pour les validations.



2.3 LES EXCLUSIONS

Spécifier ce qui **ne sera pas traité dans le cadre du projet** est essentiel pour **éviter les malentendus**, les **attentes implicites** ou les **dérives** de périmètre ("scope creep").

Exemples d'exclusions :

- L'internationalisation du site (pas de traduction prévue dans la version initiale).
- La refonte de l'identité visuelle (logo, charte graphique fournie par le client).
- Le référencement naturel (aucun accompagnement SEO au lancement).



3. SPECIFICATIONS DETAILLEES

3.1 Cas d'usage / scénarios utilisateurs

- Comment les utilisateurs interagiront avec le produit (ex : se connecter, acheter un produit...).

3.2 Maquettes ou schémas (si disponibles)

- Wireframes, diagrammes d'architecture, parcours utilisateur...

3.1 CAS D'USAGES / SCENARIOS UTILISATEURS

Cette section décrit comment les utilisateurs interagiront concrètement avec le produit. Elle permet de traduire les besoins fonctionnels en situations réelles d'utilisation.



Formes courantes :

- **Cas d'usage** : description narrative d'une séquence d'actions menant à un objectif précis.
- **Scénarios utilisateurs** : histoires courtes qui illustrent les besoins, motivations et comportements des utilisateurs.
- **User stories (en complément)** : "En tant que [type d'utilisateur], je veux [objectif] afin de [bénéfice]."

Exemple :

- Cas d'usage : Connexion à un compte
 - L'utilisateur saisit son identifiant et son mot de passe. Le système vérifie les informations. Si elles sont correctes, l'utilisateur accède à son tableau de bord.
- Scénario : Achat d'un produit
 - L'utilisateur navigue dans le catalogue, sélectionne un produit, l'ajoute au panier, renseigne ses informations de paiement et confirme la commande.

3.2 MAQUETTES

Les éléments visuels accompagnent et enrichissent la compréhension des spécifications. (si disponibles)

Types de livrables :

- **Wireframes** : maquettes simplifiées qui montrent la structure des pages ou écrans sans détails graphiques.
- **Mockups** : maquettes plus abouties avec éléments graphiques et couleurs.
- **Diagrammes d'architecture** : schémas représentant la structure technique (serveurs, bases de données, API...).
- **Parcours utilisateur** (user journey maps) : illustrations des étapes et émotions vécues par un utilisateur lors de son interaction avec le produit.

Intérêt

- Faciliter la communication entre clients, designers, développeurs et autres parties prenantes.
- Réduire les risques de malentendus et erreurs d'interprétation.
- Permettre une validation visuelle avant la phase de développement.



4. ORGANISATION DU PROJET

4.1 Planning prévisionnel

- Jalons, grandes étapes, rétroplanning.

4.2 Livrables attendus

- Ce qui doit être livré, dans quel format, à quel moment.

4.3 Critères de réussite / validation

- Comment jugera-t-on que le projet est terminé et conforme.

4.1 PLANNING PREVISIONNEL



Le planning prévisionnel permet de découper le projet en phases claires, avec des échéances estimées. Il constitue la base du suivi du projet.

Contenu typique :

- Phases principales du projet (ex. : analyse, conception, développement, tests, livraison).
- Jalons (ou milestones) : points clés marquant l'avancement (ex. : validation de maquettes, livraison d'un prototype).
- Rétroplanning : planification à rebours depuis la date de livraison finale.

Exemple de phases :

Semaine 1-2 : Recueil des besoins / ateliers utilisateurs
Semaine 3-4 : Rédaction du cahier des charges final
Semaine 5-6 : Conception UX/UI et validation client
Semaine 7-10 : Développement technique
Semaine 11 : Tests et recettes
Semaine 12 : Mise en production

4.2 LIVRABLES ATTENDUS

Un livrable est un élément produit pendant ou à la fin du projet, remis au client ou à une autre partie prenante.



Types de livrables :

- **Documents** : cahier des charges validé, spécifications techniques, documentation utilisateur.
- **Maquettes** : wireframes, prototypes interactifs.
- **Développements** : code source, livrables déployables, API fonctionnelles.
- **Outils ou environnements** : site en ligne, espace d'administration, hébergement configuré.
- **Tests et validations** : rapports de tests, protocoles de recette.

Format et échéance :

Pour chaque livrable, il est utile de préciser :

- Le contenu attendu
- Le format (PDF, Figma, archive ZIP, lien web...)
- La date ou phase de livraison
- Le responsable de la production

4.3 CRITERES DE VALIDATION

Cette section définit les conditions de validation du projet et comment on jugera qu'il est terminé avec succès.



Critères fonctionnels :

- L'ensemble des fonctionnalités décrites sont implémentées et opérationnelles.
- Les parcours utilisateurs sont fluides et sans blocages.
- L'outil répond aux besoins exprimés par les utilisateurs finaux.

Critères techniques :

- Le produit fonctionne sur les navigateurs et appareils prévus.
- Les performances sont conformes aux attentes (temps de réponse, rapidité).
- Aucun bug bloquant ou critique en production.

Critères de conformité :

- Respect du planning et du budget.
- Respect des exigences RGPD, accessibilité, sécurité, etc.
- Validation écrite par le client après la phase de recette.

5. ASPECTS TECHNIQUES

5.1 Technologies attendues

- Frameworks, langages, systèmes cibles, hébergement...

5.2 Sécurité, RGPD, accessibilité

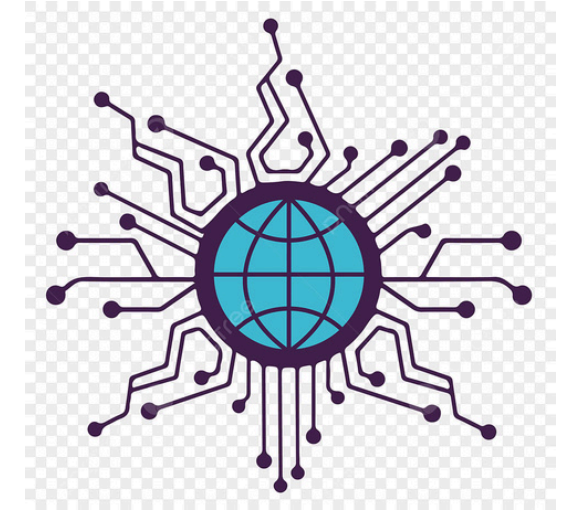
- Conformités légales, protection des données, accessibilité web (si pertinent).

5.1 TECHNOLOGIES ATTENDUES

Cette partie spécifie les choix techniques imposés, recommandés ou souhaités pour le projet. Elle oriente le développement, l'infrastructure et les compétences requises.

Éléments à préciser :

- **Langages :**
 - Frontend : JavaScript (React, Vue...), TypeScript, HTML/CSS.
 - Backend : Node.js, PHP, Python, etc.
- **Frameworks / bibliothèques :**
 - Ex : Express.js, Django, Symfony, Laravel, Bootstrap...
- **Base de données :**
 - MySQL, PostgreSQL, MongoDB, SQLite...
- **Systèmes cibles :**
 - Web (responsive), mobile natif ou hybride (ex : React Native, Flutter), desktop...
- **Hébergement et infrastructure :**
 - Serveur dédié, mutualisé, cloud (ex : AWS, Azure, OVH, Heroku...).
 - Environnement attendu : Linux/Windows, conteneurs Docker, CI/CD...



5.2 SÉCURITÉ, RGPD, ACCESSIBILITÉ



Cette partie couvre les obligations légales, éthiques et techniques liées au traitement des données personnelles, à la sécurité du système et à l'accessibilité.

Sécurité :

- Protection des données sensibles (mots de passe, paiements, données médicales...).
- Chiffrement (stockage, transmission via HTTPS, tokens...).
- Sécurité des accès : gestion des rôles, authentification forte, prévention des injections SQL, XSS...

RGPD (Règlement Général sur la Protection des Données) :

- Collecte des données strictement nécessaires au fonctionnement.
- Consentement explicite des utilisateurs (bandeaux cookies, opt-in).
- Droit à l'oubli et portabilité des données.

Accessibilité :

- Respect des normes WCAG 2.1 (niveau AA au minimum si public concerné).
- Navigation clavier, contraste couleurs, lecteurs d'écran...
- Prise en compte des handicaps visuels, moteurs ou cognitifs.