IAPLC_

PLAYBOOK · L'IA POUR LES MÉTIERS DU SI

Prompts Copilot pour consultants agiles

Chaque prompt à copier-coller. Remplacez les [crochets] par vos éléments. Structure commune : rôle · contexte · tâche bornée · format.

User stories & backlog

STORY · FORMAT INVEST

Tu es Product Owner senior, rigoureux et orienté valeur. Contexte du produit : [secteur, type d'utilisateurs, objectif du produit]. Voici le besoin métier exprimé, parfois flou : [colle le besoin]. Étape 1 : reformule le besoin en une phrase pour vérifier ta compréhension. Étape 2 : rédige 3 user stories au format INVEST (« En tant que [rôle], je veux [action], afin de [bénéfice] »), indépendantes et livrables séparément. Étape 3 : pour chaque story, ajoute 3 à 5 critères d'acceptation au format Gherkin (Étant donné / Quand / Alors) couvrant le cas nominal et au moins un cas d'erreur. Contraintes : pas de solution technique imposée dans la story, un bénéfice utilisateur clair, une story = une intention. Termine par une section « Hypothèses à valider avec le métier » et une section « Questions ouvertes » listant ce qui manque pour que ces stories soient prêtes.

DÉCOUPAGE D'EPIC

Tu es Product Owner. Voici une epic à découper : [colle l'epic]. Contexte : [contrainte de délai, priorité business, dépendances connues]. Découpe-la en user stories livrables indépendamment, chacune apportant une valeur perceptible par l'utilisateur (verticale, pas une couche technique). Pour chaque story, donne : un titre court, la user story au format INVEST, la valeur métier en une phrase, une estimation relative en points (suite de Fibonacci) avec une justification en une ligne, et les dépendances éventuelles vers d'autres stories. Propose ensuite un ordre de livraison qui maximise la valeur livrée tôt (approche « walking skeleton » d'abord), et identifie la story minimale qui permettrait déjà de tester l'idée en production. Signale toute story encore trop grosse (> 8 points) à re-découper.

DÉCOUPAGE D'UNE STORY TROP GROSSE (SPIDR)

Tu es Product Owner expérimenté dans le découpage de stories. Cette user story est trop grosse pour tenir dans un sprint : [colle la story et ses critères]. Applique méthodiquement le modèle SPIDR et propose, pour chacun des 5 axes, un découpage possible : (S) par Spike si l'incertitude domine, (P) par Chemins/parcours utilisateur, (I) par Interfaces (ex. web puis mobile), (D) par Données (un sous-ensemble d'abord), (R) par Règles métier (la règle simple d'abord, les exceptions ensuite). Pour chaque axe, indique la story la plus fine qui apporte déjà de la valeur en production, et ce qu'on repousse. Termine par ta recommandation : quel axe de découpage est le plus pertinent ici et pourquoi, avec la première story à embarquer.

CRITÈRES D'ACCEPTATION

Tu es Business Analyst méticuleux. Pour cette user story : [colle la story et le contexte fonctionnel], rédige des critères d'acceptation exhaustifs au format Gherkin (Étant donné / Quand / Alors). Couvre systématiquement : le parcours nominal, les parcours alternatifs, les cas d'erreur (données invalides, action non autorisée, ressource absente), les règles de gestion et les cas limites (valeurs vides, zéro, très grandes valeurs, concurrence). Regroupe les scénarios par thème et numérote-les. Pour toute règle que la story ne précise pas, ne l'invente pas : liste-la à part sous « ⚠ à clarifier avec le métier » avec la question exacte à poser. Termine par une checklist « prêt à développer ? » de 5 points.

DEFINITION OF DONE

Tu es Scrum Master aidant l'équipe à formaliser sa qualité. Contexte : [type de produit, niveau d'exigence qualité/sécurité, contraintes de conformité éventuelles, maturité de l'équipe]. Propose une Definition of Done complète mais réaliste, regroupée par catégorie : code (revue, standards, pas de TODO critique), tests (couverture, cas d'erreur, non-régression), documentation, sécurité & conformité, revue produit/PO, déploiement & observabilité. Chaque point doit être binaire et vérifiable (fait / pas fait), pas une intention vague. Distingue ce qui est « obligatoire à chaque story » de ce qui est « au niveau du release ». Termine par 3 questions à trancher en équipe pour finaliser cette DoD.

Sprint planning

PÉRIMÈTRE DE SPRINT

Tu es Scrum Master pragmatique. Données : vélocité des 3 derniers sprints = [X, Y, Z] points ; capacité de l'équipe ce sprint = [jours-homme disponibles, congés, cérémonies, support] ; objectif ou thème visé = [décris]. Voici le backlog priorisé : [colle stories + estimations]. Étape 1 : calcule une capacité prévisionnelle réaliste en points (moyenne pondérée de la vélocité, ajustée de la capacité, avec une marge de sécurité) et explique ton calcul. Étape 2 : propose un périmètre de sprint qui tient dans cette capacité et sert l'objectif. Étape 3 : signale les stories à risque de non-complétion, les dépendances bloquantes et les stories pas assez claires pour être embarquées. Termine par 2 scénarios : un périmètre « engagement » sûr et un périmètre « ambition » avec les stories bonus si l'équipe avance vite.

OBJECTIF DE SPRINT

Tu es Product Owner. Voici les stories candidates au prochain sprint : [colle]. Contexte : [enjeu business du moment, attentes des parties prenantes]. Propose 2 à 3 formulations d'objectif de sprint (sprint goal) : chacune tient en une phrase, est orientée résultat/valeur (pas une liste de tâches), et donne une direction claire pour arbitrer en cours de sprint. Pour chaque proposition : explique quel problème utilisateur ou business elle adresse, quelles stories du backlog la servent directement, et lesquelles n'y contribuent pas (donc négociables). Termine en recommandant l'objectif le plus pertinent et en proposant une phrase que l'équipe pourrait afficher sur le board pendant tout le sprint.

DÉCOUPAGE EN TÂCHES

Tu es développeur senior qui prépare le sprint planning. Découpe cette user story en tâches techniques concrètes de moins d'une journée chacune : [colle la story + critères d'acceptation]. Contexte technique : [stack, contraintes, dette connue]. Regroupe les tâches par couche (front, back, base de données, tests, devops/déploiement) et, pour chaque tâche : un intitulé actionnable commençant par un verbe, une estimation en heures, et ses pré-requis. Indique clairement l'ordre de réalisation, les tâches parallélisables et le chemin critique. Ajoute les tâches souvent oubliées : mise à jour de la doc, feature flag, migration de données, tests de non-régression, observabilité. Termine par les risques techniques et une tâche « spike » si une inconnue subsiste.

CARTE DES DÉPENDANCES

Tu es chef de projet technique. Voici les stories envisagées pour les prochains sprints : [colle]. Contexte : [équipes impliquées, systèmes tiers, jalons externes]. Identifie toutes les dépendances : techniques (une story en bloque une autre), de données (un jeu de données ou une API doit exister d'abord), et humaines/externes (une autre équipe, un prestataire, une validation métier). Représente-les sous forme de liste ordonnée « A doit précéder B, car… ». Propose ensuite un ordre de réalisation qui minimise les temps d'attente et les blocages, en démarrant tôt ce qui dépend d'acteurs externes (délais longs). Signale en priorité les dépendances hors de notre périmètre à sécuriser dès maintenant, avec l'action concrète à mener pour chacune.

Refinement / grooming

DEFINITION OF READY

Tu es Business Analyst exigeant sur la qualité de l'entrée en sprint. Contexte produit : [décris]. Évalue si cette user story est prête (Definition of Ready) : [colle la story, ses critères et tout élément joint]. Passe-la au crible sur : clarté du besoin et de la valeur, critères d'acceptation testables, règles de gestion explicites, dépendances identifiées, maquette/design si nécessaire, faisabilité technique confirmée, taille raisonnable pour un sprint. Pour chaque point : statut (OK / partiel / manquant) et ce qui manque précisément. Puis rédige les 5 questions les plus importantes à poser au métier avant de l'embarquer, classées par ordre d'impact. Conclus par un verdict clair : « prête », « prête sous condition de… », ou « à retravailler » avec les 2-3 actions prioritaires.

QUESTIONS DES « 3 AMIGOS »

Tu prépares un atelier « 3 amigos » (PO, développeur, QA) pour affiner une story avant le sprint. Voici la story : [colle]. Contexte : [produit, contraintes techniques, historique]. Génère les questions clés à se poser selon les 3 points de vue : côté PO/valeur (à quoi sert cette story, pour qui, quelle priorité, quel comportement attendu aux limites) ; côté développeur/faisabilité (quels impacts techniques, quels risques, quelles dépendances, quelles alternatives d'implémentation) ; côté QA/testabilité (comment on teste, quels cas passants et d'erreur, quelles données de test, quels effets de bord à surveiller). Pour chaque catégorie, propose 4 à 6 questions concrètes. Termine par une synthèse des 3 questions les plus susceptibles de faire émerger un désaccord ou un impensé, à traiter en priorité pendant l'atelier.

CADRER UN SPIKE

Tu es tech lead. Nous avons une incertitude technique qui bloque une estimation ou une décision : [décris l'incertitude et son enjeu]. Contexte : [stack, contraintes, échéance]. Rédige un spike exploitable : la question précise à trancher (une seule), pourquoi elle est bloquante et ce qu'elle débloque, un périmètre strictement borné et timeboxé (propose une durée), les pistes/options à explorer, et le livrable attendu en sortie (une reco documentée, un prototype jetable, un benchmark chiffré…). Définis le critère de réussite : « le spike est terminé quand on peut répondre à … / décider … ». Ajoute ce qu'il ne faut surtout PAS faire (ne pas partir en développement de la vraie feature), et les risques si on saute ce spike.

DETTE TECHNIQUE EN STORY

Tu es tech lead qui doit défendre un sujet technique auprès du métier. Reformule ce problème de dette technique : [décris le problème technique] en une story compréhensible et priorisable par des non-techniciens. Structure : un titre parlant, le problème expliqué par une analogie concrète, l'impact réel aujourd'hui (sur les utilisateurs, la vitesse de l'équipe, le risque, le coût), le coût de l'inaction si on ne fait rien dans 6 mois, le bénéfice attendu et comment on le mesurera, et des critères d'acceptation vérifiables. Ajoute une estimation d'effort en ordre de grandeur et une proposition de priorité argumentée (maintenant / prochain trimestre / à surveiller). Reste factuel et évite le catastrophisme : le but est une décision éclairée, pas de faire peur.

Daily & suivi

SYNTHÈSE POUR LE COPIL

Tu es chef de projet qui rédige pour un comité de pilotage pressé. À partir de ces notes brutes de daily / d'avancement : [colle]. Produis une synthèse en 3 blocs courts : (1) Avancées — ce qui a concrètement progressé, en valeur livrée, pas en activité ; (2) Blocages & risques — ce qui menace le délai ou le périmètre, avec le niveau de criticité ; (3) Décisions attendues du COPIL — formulées comme des questions fermées appelant un arbitrage. Contraintes : 5 à 7 lignes maximum au total, ton factuel, aucun jargon projet, aucun superlatif, chiffres quand ils existent. Termine par une phrase d'état global (« au vert / à surveiller / au rouge ») justifiée en quelques mots.

ESCALADE D'UN BLOCAGE

Tu m'aides à écrire un message d'escalade professionnel et efficace au sujet d'un blocage. Contexte : [décris le blocage, depuis quand, ce qu'il empêche]. Destinataire : [rôle de la personne]. Rédige un message court et structuré : une phrase de contexte, l'impact concret (délai, périmètre, budget, qui est affecté), ce que j'ai déjà tenté pour le résoudre, puis la demande précise — la décision à prendre OU l'aide exacte attendue OU la personne à débloquer — et l'échéance à laquelle j'ai besoin d'une réponse pour éviter tel effet. Ton : factuel, responsable, non accusateur, orienté solution. Propose aussi un objet d'email clair et, en option, une version plus courte pour une messagerie instantanée.

ANALYSE DE BURNDOWN

Tu es Scrum Master analytique. Voici l'avancement du sprint jour par jour (points restants) : [colle la série] ; capacité initiale = [points] ; jours restants = [N]. Analyse la tendance du burndown : sommes-nous en avance, dans les temps ou en retard par rapport à la ligne idéale ? Calcule le rythme de « burn » réel et projette la fin de sprint si ce rythme se maintient. Identifie les signaux faibles (plateau, chute tardive typique d'un « tout en test à la fin », scope ajouté en cours). Puis propose 2 à 3 actions concrètes adaptées à la situation (réduire le périmètre, lever un blocage, réaffecter, paralléliser la recette). Termine par le message d'alerte à passer à l'équipe et/ou au PO, formulé sans dramatiser.

RAPPORT HEBDO CLIENT

Tu es consultant qui rend compte à un client. Rédige le compte rendu d'avancement hebdomadaire à partir de : [éléments bruts : ce qui a été fait, en cours, points durs]. Contexte : [nom du projet, phase, sensibilités du client]. Structure : (1) Réalisé cette semaine — en bénéfices/valeur livrée, pas en tâches techniques ; (2) Prévu la semaine prochaine ; (3) Points d'attention & risques — avec, pour chacun, l'impact et l'action en cours ; (4) Décisions ou éléments attendus du client — avec l'échéance. Ton : clair, professionnel, rassurant sans masquer les difficultés. Évite le jargon technique ou explique-le. Termine par une phrase d'état global et propose un objet d'email. Longueur : tient sur un écran.

Rétrospective

ANALYSE DE RÉTRO

Tu es Scrum Master facilitateur, neutre et bienveillant. Voici les retours bruts de la rétrospective (post-its, verbatims) : [colle]. Étape 1 : regroupe les retours en 3 à 5 thèmes cohérents et nomme chaque thème. Étape 2 : pour chaque thème, identifie la cause racine probable (technique « 5 pourquoi » si utile), en distinguant les symptômes des vraies causes, et sans désigner de coupable. Étape 3 : propose pour chaque thème UNE action d'amélioration : concrète, mesurable, réaliste dans le prochain sprint, avec un responsable suggéré, un indicateur de réussite et une échéance. Priorise les actions par rapport effort/impact et recommande de n'en retenir que 2 ou 3 (au-delà, rien n'est fait). Termine par une observation sur ce qui a bien fonctionné et mérite d'être renforcé.

FORMAT DE RÉTRO SUR MESURE

Tu es coach agile créatif. Propose 3 formats de rétrospective adaptés à notre situation précise : [décris — équipe distante, sprint difficile, équipe qui s'ennuie des rétros, tensions, gros échec, gros succès…]. Pour chaque format : un nom, l'objectif visé et pourquoi il colle à notre contexte, un déroulé minuté sur 45 à 60 minutes (check-in, collecte, regroupement, définition d'actions, clôture), la question ou le support d'ouverture, le matériel nécessaire (physique ou outil en ligne), et les pièges à éviter pour l'animateur. Termine en recommandant le format le plus adapté ici et en donnant une astuce de facilitation pour faire parler les plus silencieux et éviter que les mêmes monopolisent la parole.

SUIVI DES ACTIONS

Tu es Scrum Master soucieux que les rétros produisent de vrais changements. Voici les actions d'amélioration décidées lors des 3 dernières rétrospectives, avec leur statut : [colle]. Fais le point structuré : lesquelles sont faites, en cours, ou abandonnées ; le taux de réalisation global ; et surtout les actions récurrentes qui reviennent rétro après rétro sans jamais aboutir. Pour ces dernières, propose une hypothèse expliquant pourquoi elles ne se font pas (trop vagues, pas de responsable, hors du pouvoir de l'équipe, pas de temps alloué) et une façon concrète de les débloquer (les redécouper, les sortir de l'équipe vers le management, les abandonner assumées). Termine par une recommandation pour que le suivi d'actions devienne systématique et léger.

SANTÉ D'ÉQUIPE

Tu es coach agile attentif au facteur humain. À partir de ces signaux observés durant le sprint : [décris — ambiance, tensions, charge de travail, heures supplémentaires, incidents, humeur en daily, turnover]. Propose une lecture de la santé de l'équipe sur 4 axes : charge/soutenabilité, collaboration/confiance, clarté (objectifs, rôles, priorités) et moral/engagement. Pour chaque axe : une évaluation nuancée (au vert / à surveiller / au rouge) justifiée par les signaux, un point de vigilance, et une piste d'action concrète et respectueuse. Distingue ce qui relève de l'équipe de ce qui relève du management. Termine par la seule chose à traiter en priorité cette semaine, et une suggestion de comment l'aborder sans mettre les gens sur la défensive.

Documentation

RELEVÉ DE DÉCISIONS

Tu es assistant de projet rigoureux. Transforme ce compte rendu de réunion brut (notes en vrac, parfois désordonnées) en relevé de décisions exploitable : [colle]. Produis d'abord un tableau des décisions prises : Décision / Responsable / Échéance / Statut. Puis un tableau des actions (to-do) : Action / Responsable / Échéance. Puis une liste des points ouverts ou en suspens, avec ce qui bloque et qui doit trancher. Supprime tout le bavardage, les digressions et les redites : ne garde que l'actionnable et le décidé. Si une décision, un responsable ou une échéance est ambigu ou manquant dans les notes, ne l'invente pas : signale-le explicitement sous « à confirmer » avec la question à poser. Termine par un résumé en 3 lignes à envoyer aux participants.

SPÉCIFICATION FONCTIONNELLE

Tu es Business Analyst qui rédige une spec claire pour les développeurs. Rédige une spécification fonctionnelle pour cette fonctionnalité/story : [colle la story, les critères et le contexte]. Structure : objectif et valeur métier ; périmètre (ce qui est inclus / explicitement exclu) ; règles de gestion numérotées et sans ambiguïté ; parcours nominal étape par étape ; parcours alternatifs et cas d'erreur ; données manipulées (entrées, sorties, validations) ; impacts sur les autres modules ou systèmes ; contraintes non fonctionnelles (performance, sécurité, accessibilité, RGPD si pertinent). Marque d'un « ⚠ à valider » toute zone d'incertitude plutôt que de la combler par une hypothèse implicite. Termine par une liste des questions ouvertes à trancher avant développement. Reste concis : privilégie les listes et tableaux au texte long.

NOTES DE VERSION

Tu es responsable produit qui prépare une mise en production. À partir de ces stories/tickets livrés dans la version : [colle la liste]. Rédige les notes de version en deux formats distincts. (1) Version utilisateur/métier : les nouveautés et améliorations présentées par bénéfice concret, en langage clair et positif, regroupées par thème (Nouveautés, Améliorations, Corrections), sans jargon technique. (2) Version technique/interne : la liste des changements, correctifs et évolutions, avec les points d'attention de déploiement (migrations de données, variables d'environnement, feature flags, ordre de déploiement, rollback possible) et les éventuelles ruptures de compatibilité. Signale ce qui nécessite une communication spécifique aux utilisateurs ou au support. Adapte le ton de la version utilisateur à : [type d'audience].

GUIDE UTILISATEUR

Tu es rédacteur technique qui écrit pour des utilisateurs non techniciens. Rédige un mini guide utilisateur pour cette fonctionnalité : [décris la fonctionnalité et à qui elle s'adresse]. Structure : à quoi ça sert (le bénéfice en une phrase) ; prérequis éventuels ; comment l'utiliser, en étapes numérotées, courtes et à l'impératif, chacune décrivant une action et son résultat visible ; 2 à 3 cas d'usage courants illustrés ; les erreurs fréquentes et comment les éviter ; une FAQ de 3 à 5 questions réelles. Ton : simple, direct, rassurant, sans jargon (ou expliqué). Indique où des captures d'écran seraient utiles par « [capture : …] ». Termine par « en cas de problème, … » avec la marche à suivre pour obtenir de l'aide.

ORDRE DU JOUR DE RÉUNION

Tu m'aides à préparer une réunion efficace. Sujet : [sujet et objectif de la réunion] ; durée : [X] min ; participants et rôles : [liste] ; décisions à prendre : [le cas échéant]. Rédige un ordre du jour minuté : pour chaque point, un intitulé, son objectif (informer / décider / brainstormer), le temps alloué, qui l'anime et l'output attendu. Commence par un rappel de l'objectif global de la réunion et se termine par un point « décisions & prochaines étapes ». Garde une marge de temps tampon. Ajoute, en amont, ce que les participants devraient préparer ou lire avant, pour ne pas perdre de temps. Termine par une question de clôture type pour valider que chacun repart avec ses actions, et propose un format court de compte rendu à remplir en fin de réunion.

Tests & recette

CAS DE RECETTE

Tu es testeur/QA méthodique. Génère les cas de test de recette pour cette user story : [colle la story et ses critères d'acceptation]. Contexte : [type d'application, profils d'utilisateurs, données sensibles éventuelles]. Produis un tableau : N° / Titre du cas / Préconditions / Étapes numérotées / Résultat attendu / Priorité. Couvre : tous les critères d'acceptation, les cas passants, les cas d'erreur (données invalides, champs vides, droits insuffisants, ressource absente), les cas limites (valeurs extrêmes, doublons, concurrence) et au moins un test de bout en bout du parcours réel. Marque les cas critiques à passer en priorité (smoke test). Ajoute les jeux de données nécessaires. Termine par ce qui n'est PAS couvert par cette recette et devrait l'être ailleurs (tests de charge, sécurité, accessibilité…).

STRATÉGIE DE TEST

Tu es responsable qualité pragmatique. Propose une stratégie de test pour cette fonctionnalité : [décris la fonctionnalité, sa criticité et le contexte technique]. Détaille : quels niveaux de test activer (unitaire, intégration, end-to-end, manuel exploratoire) et pourquoi ; ce qui est prioritaire compte tenu du risque (impact × probabilité de bug) ; ce qui peut raisonnablement rester non automatisé pour l'instant, et à quelle condition on l'automatisera plus tard ; les données et environnements de test nécessaires ; comment gérer la non-régression. Propose une pyramide de tests adaptée (proportion unitaire/intégration/e2e) et justifie-la. Termine par les 3 risques de qualité les plus élevés sur cette fonctionnalité et la parade de test associée à chacun.

CHASSE AUX CAS LIMITES

Tu es testeur expérimenté, spécialiste des cas que tout le monde oublie. Pour cette règle de gestion ou cette fonctionnalité : [colle], liste de façon exhaustive les cas limites et scénarios pièges. Balaie systématiquement : valeurs nulles / vides / zéro / négatives, très grandes valeurs et débordements, chaînes spéciales (accents, emojis, injection), doublons et unicité, ordre et tri, concurrence et accès simultanés, fuseaux horaires et changements d'heure, formats de date/nombre selon la locale, droits et permissions, ressource supprimée entre-temps, réseau lent ou coupé, pagination et gros volumes. Pour chaque cas : le comportement attendu (ou la question si indéfini), la probabilité et l'impact. Classe-les de « à tester absolument » à « edge case rare ». Termine par les 5 cas les plus dangereux à couvrir en priorité.

RAPPORT DE BUG CLAIR

Tu m'aides à écrire un rapport de bug qu'un développeur pourra traiter sans revenir vers moi. Voici ce que j'ai observé, en vrac : [décris le problème]. Reformule-le en un rapport structuré : un titre précis (symptôme + où), les étapes de reproduction numérotées et minimales, le résultat obtenu, le résultat attendu, la fréquence (systématique / intermittent), l'environnement (navigateur, appareil, version, compte/rôle), et les éléments joints utiles ([captures, logs, URL]). Propose une gravité argumentée (bloquant / majeur / mineur / cosmétique) selon l'impact utilisateur et l'existence d'un contournement. Reste factuel, sans interprétation de la cause. Si des informations manquent pour reproduire, liste précisément ce qu'il faudrait ajouter.

Parties prenantes & communication

TRAME DE COPIL

Tu es chef de projet qui prépare un support de comité de pilotage percutant. À partir de ces éléments : [avancement, jalons, risques, budget, décisions à prendre]. Contexte : [enjeux du projet, profil du COPIL, sujets sensibles]. Propose la trame d'un support de 5 à 7 slides : (1) une slide d'état global (santé projet, en une image) ; (2) avancement vs plan ; (3) prochains jalons ; (4) risques & plan d'action ; (5) budget/consommé si pertinent ; (6) décisions attendues du COPIL, formulées comme des choix clairs ; (7) prochaines étapes. Pour chaque slide : un titre qui porte le message clé et 3 puces maximum. Ton factuel, orienté décision, pas de jargon d'équipe. Termine par la « note d'intention » : les 2-3 messages que le COPIL doit absolument retenir.

ANNONCER UN RETARD

Tu m'aides à annoncer un retard à un client ou une partie prenante sans casser la confiance. Contexte : [nature et cause du retard, ampleur, ce qui reste maîtrisé, historique de la relation]. Rédige un message qui : reconnaît la situation clairement et tôt (pas de langue de bois) ; explique la cause factuellement, sans excuse à rallonge ni rejet de faute ; chiffre l'impact réel (nouveau délai, périmètre ajusté) ; présente le plan concret pour limiter la casse et éviter que ça se reproduise ; précise ce dont j'ai besoin de sa part le cas échéant ; et réaffirme l'engagement sur la valeur finale. Ton : honnête, responsable, posé, orienté solution. Propose un objet d'email, une version écrite complète, et 3 phrases clés si je dois l'annoncer à l'oral.

VULGARISER POUR LE MÉTIER

Tu es un excellent vulgarisateur technique. Explique cette contrainte ou décision technique : [colle] à un interlocuteur métier non technique : [précise son rôle et ce qui l'intéresse]. Commence par une analogie concrète tirée du quotidien, puis explique de quoi il s'agit sans jargon (ou en définissant chaque terme au passage), en te concentrant sur le « pourquoi ça compte pour lui ». Explique les options possibles avec, pour chacune, l'arbitrage en langage business (coût, délai, risque, valeur). Conclus par ce que ça change concrètement pour lui et la décision précise que tu attends, formulée comme un choix simple. Garde un ton respectueux, jamais condescendant. Longueur : court, quelques paragraphes ou une liste.

CARTOGRAPHIE DES PARTIES PRENANTES

Tu es consultant en conduite du changement. À partir de cette liste d'acteurs du projet : [colle noms/rôles et ce que tu sais d'eux]. Contexte : [enjeu du projet, changements induits, tensions connues]. Propose une cartographie des parties prenantes selon les axes pouvoir (capacité à influencer/bloquer) et intérêt (impact du projet sur eux). Positionne chaque acteur dans un des 4 quadrants (piloter de près, satisfaire, tenir informé, surveiller) et, pour chacun : sa posture probable (soutien, neutre, réticent, opposant), ce qu'il attend ou craint, ses leviers de motivation, et la stratégie de communication adaptée (informer, impliquer, rassurer, négocier) avec une action concrète à mener. Termine par les 3 acteurs à sécuriser en priorité et le risque si on les néglige.

Estimation & métriques

ESTIMATION PAR ANALOGIE

Tu es un développeur senior qui aide l'équipe à estimer. Voici 5 stories déjà réalisées et estimées par notre équipe, avec leurs points : [colle exemples : story → points]. Elles servent de référentiel. Estime maintenant cette nouvelle story par analogie : [colle la story et ses critères]. Étape 1 : identifie la ou les stories de référence les plus proches et explique en quoi. Étape 2 : compare la complexité (règles métier, incertitude, effort technique, tests, dépendances), pas seulement le volume de code. Étape 3 : propose une estimation en points Fibonacci, la story de référence la plus proche, et un facteur d'incertitude (faible/moyen/élevé) avec ce qui le fait monter. Si l'incertitude est trop élevée pour estimer sereinement, recommande un spike plutôt qu'un chiffre. Rappelle que c'est une estimation d'équipe à valider en planning poker, pas une vérité.

PRÉVISION DE FIN

Tu es chef de projet qui doit donner une prévision honnête. Données : il reste [N] points au backlog du release ; vélocité des 5 derniers sprints = [colle la série] ; durée d'un sprint = [X] semaines ; événements connus à venir = [congés, jours fériés, autres]. Calcule une prévision de fin sous forme de fourchette : scénario optimiste (haute vélocité), réaliste (vélocité moyenne) et pessimiste (basse vélocité, marge pour imprévus), en nombre de sprints et en date. Explique les hypothèses derrière chaque scénario et montre ton calcul. Liste les risques qui pourraient dégrader la prévision (scope qui gonfle, dépendances, dette, absences) et leur effet. Termine par la formulation à communiquer au client/COPIL : une fourchette assumée plutôt qu'une fausse date précise, avec la condition de fiabilité (« si le périmètre reste stable et la vélocité tient »).

PRIORISATION WSJF

Tu es Product Owner / RTE qui priorise un backlog avec la méthode WSJF (Weighted Shortest Job First). Voici les items à prioriser : [liste des features/stories avec un minimum de contexte]. Pour chaque item, estime sur une échelle relative (suite de Fibonacci) : la valeur métier/utilisateur, la criticité temporelle (urgence, fenêtre d'opportunité), la réduction de risque ou l'opportunité débloquée, puis la taille/effort. Calcule le score WSJF = (valeur + urgence + réduction de risque) ÷ taille, et présente un tableau trié du score le plus élevé au plus bas. Justifie chaque estimation en une ligne. Signale les items où l'estimation est très incertaine. Termine par l'ordre de traitement recommandé et 2 observations : ce qu'il faut faire vite car « petit et à forte valeur », et ce qu'il faut peut-être abandonner.

Facilitation & coaching

ANIMER UN ATELIER

Tu es facilitateur d'ateliers expérimenté. Conçois le déroulé complet d'un atelier. Objectif : [ce qu'on doit produire ou décider] ; durée : [X] ; participants : [nombre et profils] ; format : [présentiel / distanciel / hybride]. Propose : un objectif reformulé et le livrable concret attendu en sortie ; un brise-glace adapté ; des séquences minutées (divergence puis convergence), avec pour chacune la consigne exacte à donner, la technique de facilitation (post-its, dot voting, travail en sous-groupes…) et le temps ; les rôles (facilitateur, gardien du temps, scribe) ; le matériel ou l'outil en ligne nécessaire. Ajoute des techniques pour faire participer tout le monde et éviter que quelques-uns monopolisent, ainsi que les pièges à anticiper. Termine par la clôture (synthèse, validation des actions, tour de météo) et un plan B si l'atelier dérape ou prend du retard.

DÉSAMORCER UNE TENSION

Tu es coach d'équipe, neutre et bienveillant. Deux membres de l'équipe sont en désaccord (ou en tension) sur : [décris la situation, les positions de chacun et ce que tu as observé, sans prendre parti]. Aide-moi à préparer une conversation de médiation. Propose : une façon d'ouvrir la discussion qui pose un cadre sûr et non accusateur ; 3 à 5 questions ouvertes pour faire émerger les besoins et intérêts réels de chacun (au-delà des positions affichées) ; une manière de reformuler le désaccord en un problème commun à résoudre ensemble plutôt qu'un affrontement ; et des pistes pour trouver un terrain d'entente ou un compromis expérimental. Rappelle les principes de communication non violente (faits, ressenti, besoin, demande). Termine par les signaux qui indiqueraient qu'il faut escalader vers le manager, et ce qu'il ne faut surtout pas faire en tant que facilitateur.

ONBOARDING D'UN ARRIVANT

Tu es tech lead qui accueille un nouvel arrivant. Prépare un plan d'onboarding structuré sur 2 semaines. Contexte : [rôle et séniorité de l'arrivant, projet, stack technique, rituels d'équipe, outils]. Découpe par jour puis par semaine : les personnes clés à rencontrer et pourquoi ; la documentation et le code à lire dans l'ordre ; l'accès aux outils et environnements à obtenir ; les premières tâches à faible risque et à valeur pédagogique (une petite correction, une doc à améliorer) pour livrer vite en confiance ; les rituels auxquels participer. Prévois des points de contrôle réguliers (fin J2, fin S1, fin S2) avec les questions à se poser. Ajoute ce qui est souvent oublié (contexte métier, historique des décisions, à qui demander quoi) et un objectif clair et atteignable pour la fin des 2 semaines. Adapte l'exigence à la séniorité indiquée.

Discovery produit

GUIDE D'ENTRETIEN UTILISATEUR

Tu es chercheur UX rigoureux. Prépare un guide d'entretien utilisateur pour explorer ce problème/hypothèse : [décris le sujet, ce que tu cherches à comprendre et qui tu vas interroger]. Structure : un objectif de recherche clair (ce qu'on veut apprendre) ; une intro pour mettre à l'aise et cadrer sans biaiser ; des questions ouvertes, non orientées, centrées sur le vécu et les comportements réels passés (« racontez-moi la dernière fois que… ») plutôt que sur les opinions ou le futur hypothétique ; une organisation en entonnoir (du général au spécifique) ; des relances pour creuser (« pourquoi ? », « et ensuite ? », « qu'avez-vous fait à ce moment-là ? »). Évite les questions fermées, suggestives ou doubles. Ajoute une checklist des biais à éviter pendant l'entretien et une clôture. Termine par les 3 questions à ne surtout pas rater compte tenu de l'objectif.

SYNTHÈSE D'INTERVIEWS

Tu es chercheur UX qui synthétise des entretiens. Voici les notes brutes de [N] entretiens utilisateurs : [colle]. Analyse-les sans surinterpréter : fais ressortir les besoins et attentes récurrents (avec le nombre d'utilisateurs concernés par chacun, pour distinguer un signal d'un cas isolé), les points de douleur les plus cités et leur intensité, les verbatims marquants qui illustrent chaque thème, et les comportements/contournements observés. Distingue clairement les faits rapportés des interprétations que tu proposes. Puis dégage 3 opportunités produit à creuser, formulées comme des problèmes à résoudre (pas des solutions), avec pour chacune l'impact potentiel et le niveau de confiance vu les données. Termine par ce qui reste flou et mériterait des entretiens complémentaires ou un test.

PERSONA

Tu es Product Designer. À partir de ces éléments réels sur nos utilisateurs (entretiens, données, retours support) : [colle]. Rédige 2 personas distincts et crédibles, strictement ancrés dans les données fournies — n'invente aucun trait qui ne serait pas étayé ; si une information manque, marque « [à confirmer par la recherche] ». Pour chaque persona : un nom et une phrase de synthèse, son contexte et son rôle, ses objectifs, ses frustrations et points de douleur, son niveau d'aisance avec l'outil/le numérique, ses critères de décision, et ce qui le ferait adopter — ou rejeter — notre solution. Termine par ce qui différencie vraiment les 2 personas (pour éviter d'en faire des clones), et par la façon dont chacun devrait influencer nos priorités produit.

CRITÈRES DE SUCCÈS (KPI)

Tu es Product Manager orienté impact. Pour cette fonctionnalité ou initiative : [décris-la et le résultat business attendu]. Propose comment mesurer son succès. Distingue : 1 métrique « North Star » (l'indicateur d'impact principal), 2 à 3 métriques secondaires (adoption, usage, satisfaction, impact métier) et 1 à 2 contre-métriques (garde-fous : ce qui ne doit pas se dégrader). Pour chaque métrique : sa définition précise, comment et où la collecter, la fréquence de mesure, la valeur de référence actuelle si elle existe, et le seuil qui dirait « c'est réussi » vs « à revoir ». Évite les métriques de vanité (clics bruts sans lien avec la valeur). Termine par la façon d'instrumenter la collecte dès le départ (événements à tracer) et par la date à laquelle on décidera de poursuivre, ajuster ou arrêter.

LA RÈGLE SOUS TOUS CES PROMPTS

Rôle · Contexte · Tâche bornée · Format. Le contexte projet vaut dix « mots magiques ». Itérez plutôt que de recommencer. Ne collez jamais de données client sensibles hors d'une instance sous contrat (M365 Copilot).