IAPLC_

PLAYBOOK · AI FOR IT PROFESSIONS

Copilot prompts for agile consultants

Every prompt, copy-paste ready. Replace the [brackets] with your own. Shared structure: role · context · bounded task · format.

User stories & backlog

STORY · INVEST FORMAT

You are a senior, value-driven Product Owner. Product context: [sector, user type, product goal]. Here is the business need, sometimes vague: [paste the need]. Step 1: restate the need in one sentence to confirm your understanding. Step 2: write 3 INVEST user stories ("As a [role], I want [action], so that [benefit]"), independent and separately shippable. Step 3: for each, add 3 to 5 acceptance criteria in Gherkin (Given / When / Then) covering the happy path and at least one error case. Constraints: no technical solution imposed in the story, a clear user benefit, one story = one intent. End with an "Assumptions to validate with the business" section and an "Open questions" section listing what's missing for these stories to be ready.

EPIC BREAKDOWN

You are a Product Owner. Here is an epic to break down: [paste the epic]. Context: [deadline constraint, business priority, known dependencies]. Break it into independently shippable user stories, each delivering value perceivable by the user (vertical, not a technical layer). For each story give: a short title, the INVEST user story, the business value in one sentence, a relative estimate in points (Fibonacci) with a one-line rationale, and any dependencies on other stories. Then propose a delivery order that maximizes value delivered early ("walking skeleton" first), and identify the minimal story that would already let us test the idea in production. Flag any story still too big (> 8 points) to split further.

SPLITTING A LARGE STORY (SPIDR)

You are a Product Owner experienced in story splitting. This user story is too big to fit one sprint: [paste the story and its criteria]. Methodically apply the SPIDR model and, for each of the 5 axes, propose a possible split: (S) by Spike if uncertainty dominates, (P) by user Paths/flows, (I) by Interfaces (e.g. web then mobile), (D) by Data (a subset first), (R) by business Rules (the simple rule first, exceptions later). For each axis, point to the thinnest story that already delivers value in production, and what we defer. End with your recommendation: which splitting axis fits best here and why, with the first story to pull in.

ACCEPTANCE CRITERIA

You are a meticulous Business Analyst. For this user story: [paste the story and functional context], write exhaustive acceptance criteria in Gherkin (Given / When / Then). Systematically cover: the happy path, alternate paths, error cases (invalid data, unauthorized action, missing resource), business rules and edge cases (empty values, zero, very large values, concurrency). Group scenarios by theme and number them. For any rule the story does not specify, do not invent it: list it separately under "⚠ to clarify with the business" with the exact question to ask. End with a 5-point "ready to develop?" checklist.

DEFINITION OF DONE

You are a Scrum Master helping the team formalize its quality bar. Context: [product type, quality/security expectations, compliance constraints, team maturity]. Propose a complete but realistic Definition of Done, grouped by category: code (review, standards, no critical TODO), tests (coverage, error cases, non-regression), documentation, security & compliance, product/PO review, deployment & observability. Each item must be binary and verifiable (done / not done), not a vague intention. Distinguish what is "mandatory per story" from what is "at the release level". End with 3 questions the team should settle to finalize this DoD.

Sprint planning

SPRINT SCOPE

You are a pragmatic Scrum Master. Data: velocity of the last 3 sprints = [X, Y, Z] points; team capacity this sprint = [available person-days, time off, ceremonies, support]; goal or theme = [describe]. Here is the prioritized backlog: [paste stories + estimates]. Step 1: compute a realistic forecast capacity in points (velocity weighted, adjusted for capacity, with a safety margin) and explain your calculation. Step 2: propose a sprint scope that fits this capacity and serves the goal. Step 3: flag stories at risk of not completing, blocking dependencies, and stories too unclear to pull in. End with 2 scenarios: a safe "commitment" scope and an "ambition" scope with bonus stories if the team moves fast.

SPRINT GOAL

You are a Product Owner. Here are the candidate stories for the next sprint: [paste]. Context: [current business stake, stakeholder expectations]. Propose 2 to 3 sprint-goal formulations: each fits in one sentence, is outcome/value-oriented (not a task list), and gives a clear direction for trade-offs mid-sprint. For each: explain which user or business problem it addresses, which backlog stories directly serve it, and which don't (hence negotiable). End by recommending the most relevant goal and proposing a sentence the team could keep on the board all sprint long.

TASK BREAKDOWN

You are a senior developer preparing sprint planning. Break this user story into concrete technical tasks of less than one day each: [paste the story + acceptance criteria]. Technical context: [stack, constraints, known debt]. Group tasks by layer (front, back, database, tests, devops/deployment) and, for each: an actionable title starting with a verb, an estimate in hours, and its prerequisites. Clearly indicate the order of work, parallelizable tasks and the critical path. Add the often-forgotten tasks: doc update, feature flag, data migration, non-regression tests, observability. End with technical risks and a "spike" task if an unknown remains.

DEPENDENCY MAP

You are a technical project manager. Here are the stories planned for upcoming sprints: [paste]. Context: [teams involved, third-party systems, external milestones]. Identify all dependencies: technical (one story blocks another), data (a dataset or API must exist first), and human/external (another team, a vendor, a business sign-off). List them as an ordered "A must precede B, because…". Then propose an order of work that minimizes wait times and blockers, starting early on anything depending on external actors (long lead times). Prioritize the out-of-scope dependencies to secure now, with the concrete action to take for each.

Refinement / grooming

DEFINITION OF READY

You are a Business Analyst demanding on entry quality into the sprint. Product context: [describe]. Assess whether this user story is ready (Definition of Ready): [paste the story, its criteria and any attachment]. Scrutinize: clarity of need and value, testable acceptance criteria, explicit business rules, identified dependencies, mockup/design if needed, confirmed technical feasibility, reasonable size for a sprint. For each point: status (OK / partial / missing) and exactly what's lacking. Then write the 5 most important questions to ask the business before pulling it in, ranked by impact. Conclude with a clear verdict: "ready", "ready provided…", or "needs rework" with the 2-3 priority actions.

"3 AMIGOS" QUESTIONS

You are preparing a "3 amigos" workshop (PO, developer, QA) to refine a story before the sprint. Here is the story: [paste]. Context: [product, technical constraints, history]. Generate the key questions from the 3 viewpoints: PO/value (what's this for, for whom, what priority, expected behavior at the limits); developer/feasibility (technical impacts, risks, dependencies, implementation alternatives); QA/testability (how we test, passing and error cases, test data, side effects to watch). For each category, propose 4 to 6 concrete questions. End with a synthesis of the 3 questions most likely to surface a disagreement or a blind spot, to tackle first.

FRAME A SPIKE

You are a tech lead. We have a technical uncertainty blocking an estimate or a decision: [describe the uncertainty and its stake]. Context: [stack, constraints, deadline]. Write an actionable spike: the precise question to settle (only one), why it's blocking and what it unblocks, a strictly bounded and timeboxed scope (propose a duration), the options to explore, and the expected deliverable (a documented recommendation, a throwaway prototype, a quantified benchmark…). Define the success criterion: "the spike is done when we can answer … / decide …". Add what must NOT be done (don't start building the real feature), and the risks of skipping this spike.

TECH DEBT AS A STORY

You are a tech lead who must sell a technical topic to the business. Reframe this technical debt: [describe the technical problem] as a story the business can understand and prioritize. Structure: a meaningful title, the problem explained with a concrete analogy, the real impact today (on users, team speed, risk, cost), the cost of inaction if nothing is done in 6 months, the expected benefit and how we'll measure it, and verifiable acceptance criteria. Add an order-of-magnitude effort estimate and a reasoned priority proposal (now / next quarter / watch). Stay factual and avoid doom: the goal is an informed decision, not fear.

Daily & tracking

STEERING-COMMITTEE SUMMARY

You are a project manager writing for a busy steering committee. From these raw daily/progress notes: [paste]. Produce a summary in 3 short blocks: (1) Progress — what concretely moved forward, in delivered value, not activity; (2) Blockers & risks — what threatens the deadline or scope, with a criticality level; (3) Decisions needed from the committee — phrased as closed questions calling for a trade-off. Constraints: 5 to 7 lines max total, factual tone, no project jargon, no superlatives, numbers where they exist. End with an overall status sentence ("green / watch / red") justified in a few words.

ESCALATING A BLOCKER

Help me write a professional, effective escalation about a blocker. Context: [describe the blocker, since when, what it prevents]. Recipient: [the person's role]. Write a short, structured message: one sentence of context, the concrete impact (schedule, scope, budget, who's affected), what I've already tried, then the precise ask — the decision to make OR the exact help needed OR the person to unblock — and the deadline by which I need an answer to avoid a given effect. Tone: factual, accountable, non-accusatory, solution-oriented. Also propose a clear email subject and, optionally, a shorter version for instant messaging.

BURNDOWN ANALYSIS

You are an analytical Scrum Master. Here is the sprint progress day by day (remaining points): [paste the series]; initial capacity = [points]; days remaining = [N]. Analyze the burndown trend: are we ahead, on track, or behind the ideal line? Compute the actual burn rate and project the sprint end if this rate holds. Identify weak signals (plateau, late drop typical of "everything in test at the end", scope added mid-sprint). Then propose 2 to 3 concrete actions fit for the situation (reduce scope, remove a blocker, reassign, parallelize QA). End with the alert message to give the team and/or PO, phrased without drama.

WEEKLY CLIENT REPORT

You are a consultant reporting to a client. Write the weekly progress report from: [raw inputs: what was done, in progress, sticking points]. Context: [project name, phase, client sensitivities]. Structure: (1) Done this week — in benefits/delivered value, not technical tasks; (2) Planned next week; (3) Watch points & risks — each with impact and action underway; (4) Decisions or items needed from the client — with deadline. Tone: clear, professional, reassuring without hiding difficulties. Avoid technical jargon or explain it. End with an overall status sentence and propose an email subject. Length: fits on one screen.

Retrospective

RETRO ANALYSIS

You are a facilitating Scrum Master, neutral and caring. Here are the raw retrospective inputs (sticky notes, verbatims): [paste]. Step 1: group the inputs into 3 to 5 coherent themes and name each. Step 2: for each theme, identify the likely root cause ("5 whys" if useful), separating symptoms from real causes, without naming a culprit. Step 3: propose ONE improvement action per theme: concrete, measurable, realistic within the next sprint, with a suggested owner, a success indicator and a deadline. Prioritize actions by effort/impact and recommend keeping only 2 or 3 (beyond that, nothing gets done). End with an observation on what worked well and deserves reinforcing.

TAILORED RETRO FORMAT

You are a creative agile coach. Propose 3 retrospective formats fit for our precise situation: [describe — remote team, tough sprint, team bored of retros, tensions, big failure, big success…]. For each format: a name, the goal it serves and why it fits our context, a run of show timed over 45 to 60 minutes (check-in, gather, group, define actions, close), the opening question or prompt, the material needed (physical or online tool), and the pitfalls to avoid as facilitator. End by recommending the best-fit format here and giving a facilitation tip to get the quietest to speak and prevent the same people from dominating.

ACTION FOLLOW-UP

You are a Scrum Master keen that retros produce real change. Here are the improvement actions decided in the last 3 retrospectives, with their status: [paste]. Take structured stock: which are done, in progress, or dropped; the overall completion rate; and above all the recurring actions that come back retro after retro without ever landing. For those, propose a hypothesis for why they don't happen (too vague, no owner, outside the team's power, no time allocated) and a concrete way to unblock them (re-split, move out of the team to management, drop deliberately). End with a recommendation to make action follow-up systematic and lightweight.

TEAM HEALTH

You are an agile coach attentive to the human factor. From these signals observed during the sprint: [describe — mood, tensions, workload, overtime, incidents, daily energy, turnover]. Read the team's health across 4 axes: workload/sustainability, collaboration/trust, clarity (goals, roles, priorities) and morale/engagement. For each axis: a nuanced assessment (green / watch / red) justified by the signals, a watch point, and a concrete, respectful action idea. Separate what belongs to the team from what belongs to management. End with the single thing to address first this week, and a suggestion for how to raise it without putting people on the defensive.

Documentation

DECISION LOG

You are a rigorous project assistant. Turn this raw meeting note (scattered, sometimes messy) into a usable decision log: [paste]. First produce a decisions table: Decision / Owner / Deadline / Status. Then an actions (to-do) table: Action / Owner / Deadline. Then a list of open or pending points, with what's blocking and who must decide. Cut all the chatter, digressions and repetitions: keep only the actionable and the decided. If a decision, owner or deadline is ambiguous or missing from the notes, do not invent it: flag it under "to confirm" with the question to ask. End with a 3-line summary to send to participants.

FUNCTIONAL SPEC

You are a Business Analyst writing a clear spec for developers. Write a functional specification for this feature/story: [paste the story, criteria and context]. Structure: objective and business value; scope (what's included / explicitly excluded); numbered, unambiguous business rules; the happy path step by step; alternate paths and error cases; data handled (inputs, outputs, validations); impacts on other modules or systems; non-functional constraints (performance, security, accessibility, GDPR if relevant). Mark with "⚠ to validate" any uncertainty rather than filling it with an implicit assumption. End with a list of open questions to settle before development. Stay concise: prefer lists and tables to long prose.

RELEASE NOTES

You are a product owner preparing a release. From these delivered stories/tickets: [paste the list]. Write release notes in two distinct formats. (1) User/business version: the new features and improvements presented by concrete benefit, in clear, positive language, grouped by theme (New, Improvements, Fixes), no technical jargon. (2) Technical/internal version: the list of changes, fixes and evolutions, with deployment watch points (data migrations, environment variables, feature flags, deployment order, rollback possible) and any breaking changes. Flag what needs specific communication to users or support. Adapt the tone of the user version to: [audience type].

USER GUIDE

You are a technical writer writing for non-technical users. Write a mini user guide for this feature: [describe the feature and who it's for]. Structure: what it's for (the benefit in one sentence); any prerequisites; how to use it, in numbered, short, imperative steps, each describing an action and its visible result; 2 to 3 illustrated common use cases; frequent mistakes and how to avoid them; a 3 to 5 question FAQ of real questions. Tone: simple, direct, reassuring, no jargon (or explained). Indicate where screenshots would help with "[screenshot: …]". End with "if something goes wrong, …" and the steps to get help.

MEETING AGENDA

Help me prepare an effective meeting. Topic: [topic and goal of the meeting]; duration: [X] min; attendees and roles: [list]; decisions to make: [if any]. Write a timed agenda: for each item, a title, its objective (inform / decide / brainstorm), the time allotted, who runs it and the expected output. Start with a reminder of the overall goal and end with a "decisions & next steps" item. Keep a time buffer. Add, beforehand, what attendees should prepare or read in advance to avoid wasting time. End with a closing question to check everyone leaves with their actions, and propose a short meeting-notes template to fill in at the end.

Tests & QA

ACCEPTANCE TESTS

You are a methodical QA/tester. Generate acceptance test cases for this user story: [paste the story and its acceptance criteria]. Context: [app type, user profiles, any sensitive data]. Produce a table: No. / Case title / Preconditions / Numbered steps / Expected result / Priority. Cover: all acceptance criteria, passing cases, error cases (invalid data, empty fields, insufficient permissions, missing resource), edge cases (extreme values, duplicates, concurrency) and at least one end-to-end test of the real journey. Mark the critical cases to run first (smoke test). Add the datasets needed. End with what is NOT covered by this acceptance and should be tested elsewhere (load, security, accessibility…).

TEST STRATEGY

You are a pragmatic QA lead. Propose a test strategy for this feature: [describe the feature, its criticality and the technical context]. Detail: which test levels to activate (unit, integration, end-to-end, exploratory manual) and why; what's priority given the risk (bug impact × probability); what can reasonably stay un-automated for now, and under what condition we'll automate it later; the test data and environments needed; how to handle non-regression. Propose a suitable test pyramid (unit/integration/e2e proportion) and justify it. End with the 3 highest quality risks on this feature and the test countermeasure for each.

EDGE-CASE HUNT

You are an experienced tester, a specialist in the cases everyone forgets. For this business rule or feature: [paste], exhaustively list the edge cases and traps. Systematically sweep: null / empty / zero / negative values, very large values and overflows, special strings (accents, emojis, injection), duplicates and uniqueness, order and sorting, concurrency and simultaneous access, time zones and DST changes, date/number formats by locale, permissions, a resource deleted meanwhile, slow or dropped network, pagination and large volumes. For each case: the expected behavior (or the question if undefined), the probability and the impact. Rank them from "must test" to "rare edge case". End with the 5 most dangerous cases to cover first.

CLEAR BUG REPORT

Help me write a bug report a developer can act on without coming back to me. Here's what I observed, roughly: [describe the problem]. Reframe it into a structured report: a precise title (symptom + where), minimal numbered reproduction steps, the actual result, the expected result, the frequency (systematic / intermittent), the environment (browser, device, version, account/role), and useful attachments ([screenshots, logs, URL]). Propose a reasoned severity (blocker / major / minor / cosmetic) based on user impact and whether a workaround exists. Stay factual, no guessing at the cause. If information is missing to reproduce, list exactly what should be added.

Stakeholders & communication

STEERING-DECK OUTLINE

You are a project manager preparing a punchy steering-committee deck. From these inputs: [progress, milestones, risks, budget, decisions to make]. Context: [project stakes, committee profile, sensitive topics]. Propose a 5 to 7 slide outline: (1) an overall status slide (project health in one image); (2) progress vs plan; (3) next milestones; (4) risks & action plan; (5) budget/spend if relevant; (6) decisions needed from the committee, phrased as clear choices; (7) next steps. For each slide: a title that carries the key message and max 3 bullets. Factual, decision-oriented tone, no team jargon. End with the "intent note": the 2-3 messages the committee must absolutely remember.

ANNOUNCING A DELAY

Help me announce a delay to a client or stakeholder without breaking trust. Context: [nature and cause of the delay, scale, what's still under control, relationship history]. Write a message that: acknowledges the situation clearly and early (no spin); explains the cause factually, without a long apology or blame-shifting; quantifies the real impact (new deadline, adjusted scope); presents the concrete plan to limit the damage and prevent recurrence; states what I need from them if anything; and reaffirms commitment to the final value. Tone: honest, accountable, calm, solution-oriented. Propose an email subject, a full written version, and 3 key sentences if I must announce it verbally.

EXPLAIN TO THE BUSINESS

You are an excellent technical explainer. Explain this technical constraint or decision: [paste] to a non-technical business stakeholder: [their role and what they care about]. Start with a concrete everyday analogy, then explain what it's about with no jargon (or defining each term as you go), focusing on "why it matters to them". Explain the possible options with, for each, the trade-off in business terms (cost, time, risk, value). Conclude with what it concretely changes for them and the precise decision you need, framed as a simple choice. Keep a respectful, never condescending tone. Length: short, a few paragraphs or a list.

STAKEHOLDER MAPPING

You are a change-management consultant. From this list of project actors: [paste names/roles and what you know of them]. Context: [project stake, induced changes, known tensions]. Propose a stakeholder map along the power (ability to influence/block) and interest (project impact on them) axes. Place each actor in one of the 4 quadrants (manage closely, keep satisfied, keep informed, monitor) and, for each: their likely stance (supporter, neutral, reluctant, opponent), what they expect or fear, their motivation levers, and the fitting communication strategy (inform, involve, reassure, negotiate) with a concrete action to take. End with the 3 actors to secure first and the risk of neglecting them.

Estimation & metrics

ESTIMATION BY ANALOGY

You are a senior developer helping the team estimate. Here are 5 stories our team already delivered and estimated, with their points: [paste examples: story → points]. They serve as a reference set. Now estimate this new story by analogy: [paste the story and its criteria]. Step 1: identify the closest reference story/stories and explain why. Step 2: compare complexity (business rules, uncertainty, technical effort, tests, dependencies), not just code volume. Step 3: propose an estimate in Fibonacci points, the closest reference story, and an uncertainty factor (low/medium/high) with what raises it. If uncertainty is too high to estimate confidently, recommend a spike rather than a number. Remind that this is a team estimate to validate in planning poker, not a truth.

COMPLETION FORECAST

You are a project manager who must give an honest forecast. Data: [N] points remain in the release backlog; velocity of the last 5 sprints = [paste the series]; sprint length = [X] weeks; known upcoming events = [time off, holidays, other]. Compute a completion forecast as a range: optimistic scenario (high velocity), realistic (average velocity) and pessimistic (low velocity, buffer for the unexpected), in number of sprints and in date. Explain the assumptions behind each scenario and show your calculation. List the risks that could degrade the forecast (scope creep, dependencies, debt, absences) and their effect. End with the wording to communicate to the client/committee: an owned range rather than a false precise date, with the reliability condition ("if scope stays stable and velocity holds").

WSJF PRIORITIZATION

You are a Product Owner / RTE prioritizing a backlog with WSJF (Weighted Shortest Job First). Here are the items to prioritize: [list of features/stories with minimal context]. For each item, estimate on a relative scale (Fibonacci): business/user value, time criticality (urgency, opportunity window), risk reduction or opportunity unlocked, then size/effort. Compute the WSJF score = (value + urgency + risk reduction) ÷ size, and present a table sorted from highest to lowest score. Justify each estimate in one line. Flag items where the estimate is very uncertain. End with the recommended order of work and 2 observations: what to do quickly because "small and high value", and what should perhaps be dropped.

Facilitation & coaching

RUN A WORKSHOP

You are an experienced workshop facilitator. Design the full run of show for a workshop. Goal: [what we must produce or decide]; duration: [X]; participants: [number and profiles]; format: [in person / remote / hybrid]. Provide: a restated goal and the concrete deliverable expected; a fitting icebreaker; timed sequences (divergence then convergence), each with the exact instruction to give, the facilitation technique (sticky notes, dot voting, breakout groups…) and the time; the roles (facilitator, timekeeper, scribe); the material or online tool needed. Add techniques to get everyone involved and prevent a few from dominating, plus the pitfalls to anticipate. End with the close (synthesis, action validation, mood check) and a plan B if the workshop derails or runs late.

DEFUSE A TENSION

You are a team coach, neutral and caring. Two team members disagree (or are in tension) over: [describe the situation, each one's position and what you observed, without taking sides]. Help me prepare a mediation conversation. Propose: a way to open the discussion that sets a safe, non-accusatory frame; 3 to 5 open questions to surface each person's real needs and interests (beyond stated positions); a way to reframe the disagreement as a shared problem to solve together rather than a fight; and paths to find common ground or an experimental compromise. Recall the principles of nonviolent communication (facts, feeling, need, request). End with the signals indicating it should be escalated to the manager, and what a facilitator must NOT do.

NEWCOMER ONBOARDING

You are a tech lead welcoming a newcomer. Prepare a structured 2-week onboarding plan. Context: [newcomer's role and seniority, project, tech stack, team rituals, tools]. Break it down by day then by week: the key people to meet and why; the documentation and code to read in order; the tools and environments access to obtain; the first low-risk, learning-valuable tasks (a small fix, a doc to improve) to ship fast with confidence; the rituals to join. Plan regular checkpoints (end of day 2, end of week 1, end of week 2) with the questions to ask. Add the often-forgotten things (business context, decision history, who to ask what) and a clear, achievable goal for the end of the 2 weeks. Adapt the bar to the stated seniority.

Product discovery

USER-INTERVIEW GUIDE

You are a rigorous UX researcher. Prepare a user-interview guide to explore this problem/hypothesis: [describe the topic, what you want to understand and who you'll interview]. Structure: a clear research goal (what we want to learn); an intro to put at ease and frame without biasing; open, non-leading questions focused on real past experience and behavior ("tell me about the last time you…") rather than opinions or hypothetical futures; a funnel organization (general to specific); probes to dig deeper ("why?", "and then?", "what did you do at that moment?"). Avoid closed, leading or double-barreled questions. Add a checklist of biases to avoid during the interview and a close. End with the 3 questions you must not miss given the goal.

INTERVIEW SYNTHESIS

You are a UX researcher synthesizing interviews. Here are the raw notes from [N] user interviews: [paste]. Analyze them without over-interpreting: surface the recurring needs and expectations (with the number of users concerned by each, to tell a signal from an isolated case), the most-cited pain points and their intensity, the striking verbatims illustrating each theme, and the observed behaviors/workarounds. Clearly separate reported facts from the interpretations you propose. Then draw out 3 product opportunities to dig into, framed as problems to solve (not solutions), each with its potential impact and your confidence level given the data. End with what stays unclear and would warrant more interviews or a test.

PERSONA

You are a Product Designer. From these real facts about our users (interviews, data, support feedback): [paste]. Write 2 distinct, credible personas, strictly grounded in the data provided — invent no trait that isn't backed; if information is missing, mark "[to confirm through research]". For each persona: a name and a one-line summary, their context and role, their goals, their frustrations and pain points, their comfort with the tool/digital, their decision criteria, and what would make them adopt — or reject — our solution. End with what truly differentiates the 2 personas (to avoid making them clones), and how each should influence our product priorities.

SUCCESS METRICS (KPIS)

You are an impact-driven Product Manager. For this feature or initiative: [describe it and the expected business outcome]. Propose how to measure its success. Distinguish: 1 "North Star" metric (the main impact indicator), 2 to 3 secondary metrics (adoption, usage, satisfaction, business impact) and 1 to 2 counter-metrics (guardrails: what must not degrade). For each metric: its precise definition, how and where to collect it, the measurement frequency, the current baseline if it exists, and the threshold that says "success" vs "revisit". Avoid vanity metrics (raw clicks with no link to value). End with how to instrument collection from the start (events to track) and the date on which we'll decide to continue, adjust or stop.

THE RULE BENEATH EVERY PROMPT

Role · Context · Bounded task · Format. Project context beats ten “magic words”. Iterate rather than restart. Never paste sensitive client data outside a contracted instance (M365 Copilot).