ChatGPT

Proposal: AI-Powered Insight Capture & Structured Document Generation

1. Market Research & Competitive Analysis

TAM/SAM/SOM Estimates: The target market spans AI-driven tools for capturing and structuring knowledge – including product requirements, customer feedback analytics, meeting assistants, and knowledge management. This space is large and growing: for example, AI meeting assistants alone are projected to reach **$15+ billion by 2032** [1], reflecting strong enterprise demand for AI-powered insight capture. Typeform's CMO noted that even with 135,000 customers, they've barely scratched their potential user base [2]. By combining adjacent segments (feedback analysis, knowledge bases, product management tools), the **Total Addressable Market (TAM)** easily sits in the **tens of billions USD globally**. The **Serviceable Addressable Market (SAM)** can focus on tech-forward B2B organizations in ASEAN/APAC first – e.g. several thousand mid-to-large SaaS and enterprise firms in the region – representing a multi-billion opportunity. The **Serviceable Obtainable Market (SOM)** in year 1 might target a few hundred early-adopter companies in ASEAN (perhaps ~$5–10M in potential annual revenue), before scaling regionally and globally as the product gains traction.

Competitive Landscape: This product sits at the intersection of several categories, with notable competitors offering partial solutions:

- **Typeform** – A leader in conversational forms and surveys for data collection. *Strengths:* Highly engaging form UX ("more like a conversation" that builds trust and yields higher response rates [3]). *Pricing:* Tiered plans (e.g. ~$39/mo basic for 1 user, 100 responses) that scale by responses and seats [4] [5]. *Gaps:* Typeform collects data but **does not generate structured artifacts** beyond raw responses; it's not an AI "interviewer" that adapts deeply or produces documents like PRDs.

- **Dovetail** – A customer **insights platform** for user research analysis [6]. *Strengths:* Centralizes interview transcripts, tags, and notes; recently introduced AI features for auto-classifying Voice-of-Customer data. *Pricing:* Seat-based (about **$15 per user/month** for core analysis features [7]) plus add-ons (e.g. automated classification $50 for 500 data points [8]). *Gaps:* Focuses on research **analysis** and tagging rather than **interactive knowledge capture**; doesn't output execution-ready docs like proposals or roadmaps.

- **Viable** – An AI platform analyzing qualitative feedback at scale. *Strengths:* Among the first to leverage GPT-3/4 for **in-depth analysis** beyond simple summarization [9]. It integrates with support tickets, surveys, etc., and groups feedback into themes to inform product decisions [10]. *Pricing:* Primarily enterprise (likely usage-based on number of feedback items). *Gaps:* Concentrates on **post-hoc analysis** of existing text data, not a live adaptive interview. It outputs reports and dashboards, not tailored documents like PRDs or postmortems for internal use.

- **Glean** – An AI-powered enterprise search and knowledge discovery tool. *Strengths:* Indexes all internal knowledge (wikis, docs, tickets) to answer questions and surface relevant info. *Gaps:*

Solves knowledge **retrieval**, not knowledge **capture** – it doesn't help extract insights from people via dialog, nor produce new structured artifacts.

- **Fireflies.ai** – An AI meeting assistant that transcribes and summarizes meetings. *Strengths:* Automatically records calls (Zoom, Teams, etc.) and generates summaries and action items. *Pricing:* Freemium with **Pro ~$10–18/user/month** and Business ~$19–29/user/month for unlimited transcripts [11] [12]. *Gaps:* Focused on live meeting notes; not designed for **interactive chat interviews** outside of meetings, and not tailored to produce artifacts like a refined proposal or training manual.

- **Others:** *Enterpret* (AI-driven voice-of-customer feedback analysis used by product teams, e.g. Notion uses it to scale feedback loops by 10×), *SurveySparrow* (conversational survey tool marketing itself for "scalable feedback ops" [13]), and general-purpose LLMs (e.g. using ChatGPT directly). These indicate demand for better feedback handling and insight generation. However, **no single tool currently covers our use cases end-to-end** – i.e. conducting an adaptive interview and automatically outputting a polished, execution-ready artifact (PRD, backlog, etc.) without extensive manual work.

Opportunities in Underserved Use Cases: The gaps above point to underserved needs: - **Feedback Operations ("Feedback Ops"):** Many companies struggle to loop customer feedback into product planning. Tools like Viable and Enterpret help analysis, but an opportunity exists for an AI that directly engages stakeholders (internal or external) in a guided conversation to capture feedback and immediately turn it into a prioritized backlog or requirements document. Experts note the importance of structured "feedback ops" to triage and act on input efficiently [14], and startups are emerging to productize this function [13] – our solution could lead here by capturing raw feedback and producing actionable output in one flow. - **Operational Retrospectives:** Post-mortems and retrospectives are often unstructured docs that depend on a manager's notetaking. There's an opportunity for an AI interviewer to conduct incident debriefs or project retros, ensuring key questions are asked and then compiling a timeline and lessons learned. Current meeting assistants capture transcripts, but **don't ensure the right probing questions** are asked; our adaptive approach can fill that void, standardizing retros with minimal effort. - **Knowledge Capture & Transfer:** Organizations rely on tribal knowledge that often leaves with individuals. There's no popular tool that *interviews* experts to create structured SOPs or training docs. This use case – capturing an expert's know-how in a documented, reusable format – is underserved. Traditional knowledge management systems (wikis, etc.) rely on manual writing, whereas an AI interviewer could dramatically streamline this "brain dump" process.

By targeting these niches, we differentiate from survey tools and transcription bots. **In summary**, the market is large and growing, competitors validate certain features (conversational UI, AI analysis) but no one covers the full adaptive interview to structured artifact pipeline. The field is ripe for a product that combines **conversational data collection, intelligent analysis, and automatic document structuring**, addressing the above pain points in one solution.

2. Pricing Strategy

A balanced pricing strategy should combine *user-based plans* with *usage-based elements*, to accommodate both product-led growth and enterprise procurement:

- **Freemium Tier:** Offer a free tier to drive adoption and product-led growth. This could allow a single user to conduct a small number of interviews per month (e.g. 3 sessions) with basic output, and perhaps watermarking or limited export formats. **Goal:** Seed usage among

individual PMs or startup founders, create word-of-mouth, and capture leads for upsell. Many competitors use freemium: Fireflies has a free tier (limited transcripts) and Dovetail offers a free plan for 1 project [15], validating that free access lowers entry barriers.

- **Pro (Team) Tier:** A paid **per-seat plan** (monthly or annual) suitable for small teams and SMEs. This could be priced in the range of **$20–$50 per user/month**, aligning with analogous tools (e.g. Dovetail Pro at $15/user [7], Fireflies ~$18/user [16], higher for more value-add). The plan would include a generous number of AI interview sessions or tokens per month (for example, 20 interviews or X thousand tokens of processing) and all core features. *Rationale:* Seat-based pricing is familiar in B2B SaaS and maps to the value each active user (PM, researcher, etc.) gets. It also encourages departmental adoption (e.g. a product team of 5 can justify $100–$250/month).

- **Usage-Based Components:** To account for heavy usage without deterring average users, we can include **fair-use limits** in each tier with the option to buy more. For instance, Pro tier includes 20 sessions; additional interviews or extra-large outputs can incur a usage fee (or require a higher tier). This hybrid mirrors how Typeform tiers include a response quota [5] [17] and how Fireflies imposes "fair-use" limits on transcription hours [12]. Usage-based add-ons ensure high-volume customers pay their share while keeping base pricing predictable for most.

- **Enterprise & Premium Plans:** For larger organizations, offer an enterprise tier with volume discounts, advanced features, and dedicated support. Likely **custom pricing (negotiated)** per deployment. Features could include single sign-on, custom data retention policies, on-premise/virtual private cloud options, and integration with enterprise knowledge bases or Jira/Azure DevOps. Enterprise deals might be structured as an annual platform fee (in the five-to-six figures) based on expected seats or usage. *Comparables:* Glean (enterprise search) and Typeform's Enterprise plan operate via custom quotes, and Dovetail's top plan (~$900/month for 15 users as per some sources) scales up for larger teams [18].

- **Tiered Plan Structure:** We can summarize a possible tiering:

Plan	Target User	Pricing (Annual)	Key Limits	Key Features
Free	Individuals, trial users *"Personal"*	$0 (free)	3 interviews/month Basic models only	Export to text/Markdown, community support
Pro Team	SMBs, product teams *"Professional"*	~$30/user/mo (annual) [19] [7]	20 interviews/user/mo (then usage fees) Up to GPT-4 tier model	All output formats, basic integrations (Jira, Confluence), email support

Plan	Target User	Pricing (Annual)	Key Limits	Key Features
Enterprise	Enterprises, large orgs *"Enterprise"*	Custom ($$$)	Unlimited interviews (within SLA) Dedicated instance	Advanced model access, enterprise integrations (SSO, knowledge DB), priority support, onboarding services

Table: Proposed pricing tiers (illustrative)

- **Comparison to Competitors:** Our **seat-plus-usage** approach aligns with market expectations. For example, Typeform effectively charges ~$25–$80 per seat (if you divide plans by allowed users) [4] [17], and Fireflies is $10–$29 per user [11]. Given our product's higher value delivery (it produces complete artifacts, not just data collection or transcripts), a slightly higher price point is justified, especially if we save hours of work per output. However, we will monitor competitor pricing closely. A **value-based pricing** narrative is important: we price relative to the cost/time saved in drafting a PRD or proposal. If our AI saves a PM 5–10 hours of work per document, paying even a few hundred dollars a month is a bargain in a business context.

- **Freemium vs Enterprise Balance:** Initially, we lean on **freemium and lower-tier plans to drive adoption** (product-led growth model). Over time, as we build features like admin controls and data governance, we can push an **enterprise upsell**. The freemium funnel feeds the pipeline: those 95% of TAM not ready to buy now can still try the product and be nurtured until value is proven [20]. Once teams rely on it, upselling to org-wide licenses or enterprise features becomes a natural next step.

- **Pricing Evolution:** We should remain flexible. As we expand globally, we might implement regional pricing (for APAC markets, if needed, to lower barriers). Also, **tiered pricing by functionality** could emerge (e.g. a "Standard" vs "Advanced" plan where advanced includes multi-user interview support or specialized industry templates). Initially, though, simplicity is key – clear tiers that map to team size and usage will make buying easy.

In summary, **seat-based pricing with generous included usage** (and freemium entry) is recommended. This aligns with competitor strategies and B2B buyer expectations, while usage-based components capture value from heavy users. We will continuously benchmark against competitors' plans and be ready to adjust (for example, if a competitor undercuts on price in a segment, or if our cost structure for AI API calls requires pricing tweaks). Overall, our pricing should communicate **value and scalability**, allowing a small team to start affordably and large enterprises to scale up confidently.

3. Go-to-Market (GTM) Strategy

Initial Buyer Personas: We will focus on a few key personas who feel acute pain points in insight capture and documentation: - **Product Managers (PMs) and Product Owners** – They constantly gather customer needs and stakeholder inputs to write PRDs, roadmaps, etc. This tool directly turns their interviews or ideas into these artifacts, saving them time. Early-adopter PMs (especially in tech startups or innovation teams) are ideal champions. - **Founders and Startup Teams** – In small startups, founders often wear PM, sales, and CX hats, and need to produce proposals, specs, and retrospectives efficiently.

They value anything that accelerates execution and captures knowledge before it's lost. - **Customer Experience (CX) and Research Leads** – Heads of CX or UX research who conduct user interviews, feedback sessions, and need to summarize insights for other teams. They can use the tool to ensure no feedback is lost and automatically produce reports or action items, i.e. power their "feedback ops" team. - **Operations and Engineering Managers** – Particularly for postmortems and process docs. DevOps or IT managers running incident postmortems, or Ops managers documenting processes, could use the chat to streamline these normally tedious write-ups. - **Consultants or Solutions Engineers** – Professionals who do discovery with clients and then craft proposals or scopes of work. They benefit from an interview-to-proposal flow that ensures they ask all the right questions and get a polished doc.

In ASEAN/APAC, we anticipate **tech-savvy organizations** (e.g. regional SaaS companies, digital agencies, and innovation units of larger enterprises) will provide our beachhead. These personas exist globally, but starting in APAC allows us to leverage local networks and success stories before expanding outward.

Channels & Tactics:

- **Inbound Content Marketing:** We will establish thought leadership around "insight capture" and "AI for knowledge work". This means publishing high-quality content – blog posts, whitepapers, webinars – on topics like *"How to turn customer feedback into a product roadmap overnight"*, *"Adaptive interviews: a new approach to requirements gathering"*, and *"Boosting feedback loops with AI"*. By sharing actionable tips and industry research, we attract the 95% of the market not yet actively shopping [20], educating them on the problem and our solution. SEO will target keywords around product requirements, postmortem templates, etc. As an example, Typeform's content on improving survey experiences helped it capture interest in a crowded market [21]; we can do similar by highlighting underserved needs like feedback ops and knowledge capture.

- **Product-Led Growth (PLG) & Virality:** The freemium model and an intuitive onboarding are central to PLG. We'll allow users to experience a "wow moment" (e.g. the first time the AI generates a full PRD draft) within minutes of signup. In-app prompts will encourage sharing (for instance, allowing users to invite team members to collaborate on an interview or review output). Successful outputs can carry subtle branding or "Generated by [OurProduct]" footers to spark curiosity in others. We aim for small teams to adopt on their own – similar to how Slack or Notion spread – and later seek management buy-in once value is clear.

- **Sales-Assisted Pilots:** For mid-size and enterprise prospects (perhaps identified via inbound inquiries or targeted outreach), we'll run **pilot programs**. These are time-bound trials (e.g. 4-6 weeks) with a defined use case (like a division in a bank using it for internal retrospectives). We provide a bit of hand-holding: onboarding sessions, custom prompt tuning if needed, and integration support (maybe hooking into their Confluence or CRM). The goal is to demonstrate ROI in a real environment. Sales-assisted doesn't mean heavy traditional sales; it's more *customer success* and *solutions engineering* helping large customers realize value. Many PLG companies evolve into **product-led sales**, where usage data identifies hot leads and then human touch converts them [22] [23]. We'll adopt that model – use in-app analytics to see which teams have high engagement, then offer them tailored enterprise deals.

- **Community and Evangelism:** We should build a community of early users and fans. This could be via a Slack/Discord group or a forum where product managers and researchers discuss tips for insight capture. We can host AMAs or office hours with our team. Evangelists (perhaps popular PMs or thought leaders in APAC) could be given early access in exchange for feedback and case studies. This fosters word-of-mouth. Developers and techies are excited by AI;

engaging with local tech meetups or AI conferences in the region (as speakers or sponsors) will also build awareness.

- **Regional Focus (ASEAN/APAC) to Global Expansion:** We will start by securing a core base in APAC – leveraging networks in Singapore, Malaysia, Indonesia, etc. Tactics include partnerships with regional startup incubators or SaaS resellers, and success stories highlighting local companies. Once we refine the product and model in APAC, we'll expand globally:

- *Localization:* Ensure the UI and AI support key languages (English first, but eventually languages like Bahasa Indonesia, Thai, etc., and later major global languages). This is important for adoption across Asia and later Europe/Latin America.
- *Global Marketing:* Use APAC case studies to pitch similar companies in the US/EU. For example, if we succeed with a Singaporean fintech's product team, use that story to target fintech product teams in the US.

- *Enterprise Sales as we mature:* In later stages, hire sales reps or partners in key markets (US, Europe) to handle large accounts, while maintaining a PLG core. By then, our brand and inbound engine (content, SEO) should already be generating demand overseas.

- **Channels Overview:**

- *Digital content* (blogs, SEO, social media posts focusing on LinkedIn where PMs and execs engage).
- *Webinars and workshops:* perhaps "live demo" webinars showing how to go from chat to PRD in 30 minutes.
- *Email marketing:* offer a newsletter about "Insightful Product Management" with curated tips (building trust and staying top-of-mind).
- *Paid acquisition:* carefully use targeted ads (LinkedIn, Google) for keywords like "product requirements template" or "AI meeting summary", to capture active searchers. However, expect higher ROI from content/organic channels given this is a relatively new category.

Growth & Expansion Timeline: In the first 6–12 months, focus on **product-market fit** with our personas in APAC. Measure success via active usage and conversion rates from free to paid. Once we have, say, 10+ happy logo customers and strong retention, use that foundation to raise additional capital or fuel expansion into English-speaking markets globally. Our go-to-market evolves from **founder-led sales in APAC** to a **scalable mix of PLG and inside sales globally**. Eventually, we envision this product as a **global B2B SaaS platform**, but by starting in our region and expanding outward, we manage risk and learn rapidly from a diverse user base.

4. Product Scope & Demo Use Cases

The product will provide an **adaptive chat interview interface** and the ability to transform the dialogue into a variety of structured outputs. We will showcase its capabilities through **six primary demo tracks** (use case scenarios), each highlighting how an interactive interview leads to execution-ready artifacts. For each track, we define the chat's behavior (how it interviews the user), the output schema and format, the "wow moments" to delight users, and trust/reliability features to ensure confidence in the results. We also suggest additional use cases beyond the initial six.

a. User Feedback → PRD + Feature Backlog

Scenario: A Product Manager wants to turn raw user feedback into a Product Requirements Document (PRD) and a development backlog. Instead of writing a PRD from scratch, they engage in a chat where the AI interviews them about the insights they've gathered.

- **Chat Behavior:** The AI acts as a **product discovery interviewer**. It asks the PM high-level questions first (e.g. *"What problem or user pain point are we addressing?"*), then drills down into details: user personas, key use cases, constraints, and success metrics. It adapts based on answers – for instance, if the PM mentions an insight like "users are frustrated with slow report generation," the AI might follow up: *"Which users are most affected? What improvement do they expect?"*. The tone is that of a knowledgeable **product coach**: guiding the PM to think of all angles (business goals, user experience, technical feasibility).

- **Output Schema:** Two outputs are produced: **(1) A PRD** – structured with sections like *Problem Statement, Objectives, User Stories, Functional Requirements, Non-functional Requirements, Metrics, Timeline*. The AI populates each section with the information gleaned in the chat. For example, under *User Stories* it might list key scenarios and acceptance criteria discussed. **(2) A Backlog** – essentially a list of feature tasks or user stories (often in the form of Jira tickets). This can be exported as a CSV or directly pushed to a tool like Jira via integration. Each backlog item might include a title, description, priority, and link to the PRD section.

- **"Wow" Moments:** As the chat progresses, the AI might summarize on-the-fly ("Let me recap the key needs you've mentioned…"), demonstrating it's "understanding" the user's input. One wow moment is when the **completed PRD is shown in seconds** after the interview – fully formatted, with coherent writing that reads as if an experienced PM wrote it. Another wow factor: the backlog is automatically prioritized if the PM gave hints about importance (the AI can say: *"Based on our discussion, I've ordered features by priority. Let's review if this matches your expectation."*). The PM can then adjust via chat commands (e.g. "swap priority of item 2 and 3"). The fluidity of going from conversation to a structured, lengthy PRD is magical.

- **Export & Integration:** The PRD can be exported as Markdown, PDF or directly to a Confluence/ Notion page. The backlog can be exported to CSV or via API to project management tools. We ensure formatting (tables, bullet points) is clean. Perhaps provide a PRD template library – the user can pick a template at start (e.g. "classic PRD" vs "Amazon PRFAQ style") and the AI follows that schema.

- **Trust & Reliability:** The AI should avoid fabricating requirements not discussed. To build trust, it might highlight assumptions it made. For instance, if the user didn't specify a target user persona, the AI might assume one based on context – but mark it for confirmation: *"(Assumed persona: SMB end-user – please confirm or edit.)"*. A **confirmation loop** at the end allows the PM to review the outline before final generation: the AI can show a PRD outline and ask if anything is missing or if certain sections are correct. This helps catch misinterpretations. Additionally, each backlog item in the output could link back to the portion of the conversation that generated it, providing traceability.

b. Sales Discovery → Proposal + Scope Document

Scenario: A Sales or Solutions Engineer has an initial discovery call with a prospective client. Using our tool, they conduct a chat *after* or *in place of* the live call to ensure all requirements are captured, then automatically get a tailored sales proposal and project scope.

- **Chat Behavior:** The AI takes on the role of a **sales discovery assistant**. It greets the user (the salesperson) with: *"Let's draft a proposal. I'll ask you about the client's needs, timeline, budget, etc."*. It then systematically covers key areas: client background, problem they want solved, specific requirements, any solution preferences, timeline/deadline, budget sensitivity, and any stakeholders or decision criteria. It adapts in real-time; if the user mentions a specific requirement ("They need integration with Shopify"), the AI will drill deeper ("Okay, what specific Shopify data or functions should our solution integrate with?"). Essentially, it mimics an **experienced consultant** who knows what to ask to scope a project fully.

- **Output Schema:** Two outputs: **(1) A Proposal** – a client-facing document (PDF or slide deck outline) that includes an **Executive Summary**, proposed solution overview, value proposition, timeline, and pricing. This reads in a persuasive tone, ready to send to the client. **(2) A Detailed Scope of Work (SoW)** – an internal/technical document or appendix that lists deliverables, milestones, and responsibilities in detail. The proposal might be high-level for an executive audience, while the scope is more granular for project managers. Both are generated from the same interview data: the "proposal" is basically a polished narrative version, and the "scope" is more structured (task list, timeline gantt or table, etc.).

- **"Wow" Moments:** One wow factor is **tone adaptation** – the AI can instantly switch the output tone from formal to casual if the user asks, or generate the proposal in the client's industry vernacular. E.g., *"Generate the proposal as a formal enterprise-style PDF"* versus *"Make a one-page informal proposal email"*. Seeing a professionally worded proposal draft seconds after answering questions is powerful. Another wow moment: **auto-insertion of relevant marketing content** – if we have stored company boilerplate (about us, case studies), the AI can include relevant snippets (like a past success story in a similar industry) in the proposal. The salesperson might exclaim that the AI remembered to add a case study without them explicitly prompting – a delightful surprise demonstrating the AI's context awareness.

- **Export & Integration:** The outputs can be exported to common formats: the proposal as PDF or Word, possibly with our branding that can be replaced with the company's. The Scope document could export to Word or task management formats. Integration with CRM is key: e.g., push the summary to Salesforce as notes on the opportunity, or log the scope in project management tools if the deal is won. We might also allow *template editing*: sales teams often have proposal templates; they could upload a template and the AI fills the blanks accordingly.

- **Trust & Reliability:** Accuracy is crucial in anything client-facing. The AI should confirm all factual details (client name spelling, agreed pricing, etc.) before finalizing. A reliability feature could be a **checklist** the AI provides: *"Here are key items I included – [Client's problem statement], [Solution outline], [Timeline: Q4 2026], [Budget: $50k]. Please verify these are correct."* This acts as a final review step. Also, to avoid hallucinations, if the user is unsure of something (say they didn't mention budget), the AI leaves a placeholder or suggests a typical range *without asserting it as fact*. It might flag sections with missing info. Internally, we could implement a **validation module** that scans the proposal for inconsistent or contradictory statements. For example, if timeline in one section says 3 months and elsewhere 6 months, the system catches it before output (using a contradiction detection technique).

c. Postmortem → Timeline + Action Items

Scenario: After an incident or project failure, a team lead wants to create a postmortem report. Instead of writing a doc from scratch, they use the AI to interview them (and possibly others) about what happened, then generate a timeline of events and a set of action items to prevent future issues.

- **Chat Behavior:** The AI behaves like a **postmortem facilitator**. It takes a neutral, probing tone: *"Let's document what happened. Can you walk me through the incident from start to finish?"*. As the user describes events, the AI may break in to clarify sequence ("Do you recall what time that error was first detected?") and root cause factors ("What underlying causes did you identify?"). It will ensure coverage of key postmortem sections: **Timeline of Events**, **Root Cause**, **Impact (affected systems/users)**, **Resolution steps**, and **Follow-up Actions**. The AI might reference known postmortem best practices (if trained on that) – for example, reminding the user to describe *why* an issue wasn't caught earlier (to tease out process improvements). It effectively leads the user through a structured reflection.

- **Output Schema:** The output has two main parts: **(1) A Timeline** – often presented as a chronological bullet list or table (timestamps and events). Each entry might look like "**10:05** – Monitoring alert X triggered. **10:15** – Engineering on-call John Doe responded…". This gives a factual reconstruction. **(2) Action Items / Recommendations** – a list of concrete follow-ups, each with an owner if mentioned and a due date or priority. For example, "**Action 1:** Implement database replication monitoring (Owner: DevOps Team, Due: EOW)." The overall postmortem report can include a summary narrative as well, but the timeline and action list are the structured highlights.

- **"Wow" Moments:** The AI's ability to **remember details across the conversation** shines here. If the user earlier mentioned "We rolled back the deploy at 10:30," and later forgets a detail, the AI can surface it: *"Earlier you said the rollback was at 10:30, which I've noted in the timeline. Did anything occur between 10:15 and 10:30 that we should add?"*. This demonstrates excellent context memory and feels like a human scribe who never misses a beat. Another wow factor: transforming a messy incident debrief into a **clear, blameless postmortem report** with structured recommendations. Users might also be impressed by extras like an automatically generated **incident severity tag or diagram** (if we integrate with a visualization, e.g., a sequence diagram of events). Even just how quickly the timeline is organized and the action items phrased in a smart way (e.g. using imperative tone for actions) can elicit a "wow, that would have taken me hours" response.

- **Export & Integration:** The postmortem can export to a Confluence page or PDF. Importantly, integration with tracking tools: perhaps automatically create tickets in Jira for each action item (if configured), so the follow-ups are actionable. The timeline could be exported as a CSV or JSON for analysis. If the company uses incident management tools (e.g. PagerDuty or Statuspage), an integration could sync key info (like incident IDs, times). Also, a possibility: output a sanitized summary for external audiences (if it's a customer-facing RCA) vs an internal detailed version – the AI could produce both if asked.

- **Trust & Reliability:** Postmortems must be factual. The AI should ask for clarification rather than guess any facts. A reliability feature could be that the AI doesn't fabricate timestamps – it uses only what the user provides. If the timeline has gaps, it can flag them: e.g. *"[Time not specified]"* or ask *"Do we know approximately when X happened?"*. The confirmation loop here could involve the AI reading back the final timeline: *"Can you review this timeline to ensure accuracy?"*. Also, to ensure a *blameless* tone (important in modern postmortems), the AI avoids charged language –

we could have a content check that removes phrasing that assigns personal blame (focusing on process failures instead). This aligns with reliability/trust by making the report constructive rather than finger-pointing, which users will appreciate as a best practice.

d. Stakeholder Alignment → Roadmap + Decision Log

Scenario: A Product Lead or Project Manager is consolidating input from various stakeholders (marketing, engineering, leadership) on strategy. They want to ensure everyone is aligned on a roadmap and keep a log of major decisions made during planning. The AI interview helps capture each stakeholder's priorities and concerns, then produces a unified roadmap and a record of decisions.

- **Chat Behavior:** The AI operates as a **facilitator/moderator**. It might simulate a meeting by sequentially asking about each stakeholder's perspective: *"What are the top priorities from the Sales perspective? Any specific deadlines they're pushing for?"* then *"What about Engineering – any technical constraints or big refactors on their roadmap?"*. The user (perhaps the PM) might input answers on behalf of different stakeholders or even invite those stakeholders into the chat one by one. The AI will reconcile these inputs, asking follow-ups to resolve conflicts: e.g., if Marketing wants feature A by Q1 but Engineering says it can only be Q2, the AI highlights this and asks the PM to set a decision: *"How do we prioritize Feature A given the timing conflict?"*. Essentially, the chat helps surface trade-offs and decisions explicitly. It maintains a neutral, synthesizing tone (*"I hear two main goals: expand to new market X and fix technical debt Y. Let's clarify which comes first."*).

- **Output Schema: (1) A Product Roadmap** – likely in the form of a timeline or feature release plan. This could be a visual roadmap (quarters vs initiatives) or simply a table: Feature/Project, Target quarter or date, Owner, Status. It will list the agreed priorities in order. **(2) A Decision Log** – a list of key decisions made and the rationale. Each entry might include the date, decision description, and context. For example: "**Decision:** Postpone Feature B to Q4. **Made by:** Leadership team on 2026-02-01. **Reason:** Need to focus on Feature A for market launch [24]." The decision log is essentially meeting minutes distilled to decisions and reasoning, which is invaluable for later reference ("Why did we decide X?").

- **"Wow" Moments:** One wow moment is seeing the **roadmap auto-prioritized** after discussing many inputs. The AI might say, *"Given everything discussed, here's a draft roadmap order: 1) Launch in ASEAN market (Q2), 2) Integrate CRM feature (Q3)..."* – turning a complex discussion into a clear plan. Users will also love the **consistency check** the AI provides: if a stakeholder changes something mid-way (e.g., "Actually, we need to swap those priorities"), the AI can update the roadmap live and confirm the change. Another impressive aspect is the decision log – in multi-stakeholder meetings, decisions often get lost; the AI capturing them in real-time (and even reading them back: *"So far, decisions made: A, B, and C – correct?"*) feels like having a perfect secretary. The stakeholders can even ask the AI during the chat: *"What did we decide about the budget cap?"*, and if that was discussed, the AI can recall it. That kind of memory and summarization is a "wow" because it shows the AI truly keeping track across a long conversation (something research highlights as challenging for LLMs, requiring careful context management [25]).

- **Export & Integration:** The roadmap can be exported to a format that product teams use – CSV, or directly into a roadmap tool if possible (even a simple integration like pushing to Trello or a timeline view in a web page). The decision log might live in Confluence or a project wiki; exporting it to a shared document ensures transparency. We could also integrate with communication tools: after a planning session, auto-email the roadmap and decision log to all stakeholders as meeting notes. Perhaps even hook into Slack: the AI could post a summary in a

channel for those who didn't attend. These integrations ensure the artifacts actually get used and seen.

- **Trust & Reliability:** In alignment scenarios, **completeness and neutrality** are important. The AI must not "take sides" or omit a stakeholder's concern. To enforce this, a reliability feature could be that the AI explicitly lists each stakeholder input it recorded and asks the user to confirm nothing is missing. Also, decisions should be logged with context to avoid later misinterpretation. We might include the rationale in the log (which our AI can derive from the discussion, or ask the user to state). For example: *"Decision: Move Launch to Q2 (because Q1 had resource conflicts with Project Z)."*. This matches how effective alignment documents are written. By providing rationale, we avoid confusion and build trust that the AI captured the *why* not just the *what*. From a technical standpoint, we'll ensure the conversation memory handles many turns – possibly summarizing earlier parts as needed (applying context engineering so that nothing important is lost even in a lengthy multi-turn chat [26] [27]).

e. Expert Knowledge → SOP / Training Document

Scenario: A veteran employee or subject-matter expert (SME) needs to transfer their knowledge (say, a specific operational procedure or technical process) into a documented Standard Operating Procedure (SOP) or a training manual for others. The AI conducts an interview to extract this knowledge in a structured way.

- **Chat Behavior:** The AI acts like an **instructional designer or journalist** interviewing the expert. It starts broad: *"Can you describe the process you want to document, at a high level?"*. Then it breaks it down step-by-step: *"What's the first thing someone needs to do? Could you walk me through the next step…?"*. It also asks for clarifications and reasons: *"Why is Step 3 done that way – is there a specific rationale or best practice behind it?"*. The AI uses teaching techniques – e.g., asking for examples or common pitfalls: *"What are common mistakes to avoid in this step?"*. The tone is curious and detail-oriented, ensuring no critical tacit knowledge is left unstated. Essentially, it's trying to **download the brain** of the expert by probing like a seasoned interviewer who knows little and wants to know everything.

- **Output Schema:** The output is typically an **SOP document** or **Training Guide**. It will be structured with a clear sequence (if it's a process): numbered steps or phases. Each step can have sub-bullets for details, tools needed, and tips. For example:

- Step 1: Do X… (with Purpose, Responsible Role, Tools required)

- Step 2: Do Y… Additionally, we might include a **FAQ or Q&A section** if during the interview the expert mentioned "People often ask if they can skip this – the answer is no because…". The training doc could also have a summary or an introduction explaining the context of the procedure and an appendix for additional resources. If it's more of a knowledge area (not a strict procedure), the output might be formatted as a **wiki article** or **frequently asked questions** style. The key is, it's structured for easy consumption by a novice learner or new employee.

- **"Wow" Moments:** The expert will experience the **joy of having a document appear that truly reflects their expertise without them writing it themselves**. A wow moment is when the AI, through its questions, surfaces something the expert forgot to mention initially. For instance, the AI might ask, *"You said after backup, the system is restarted. What verification is needed to ensure the restart was successful?"*. The expert goes, "Ah, good point, we should verify the service status." The AI's thoroughness feels like a competent collaborator. Another wow: the final SOP comes

with polish – perhaps the AI automatically generates a **flowchart diagram or checklist** if appropriate (maybe through a plugin/tool). Seeing a diagram of their process that the AI compiled (even if rudimentary) can be impressive. Also, if the expert mentions an acronym or jargon, the AI could automatically include a short definition or glossary section – showing it's anticipating what a new person might not know. That level of foresight in the output is delightful.

- **Export & Integration:** The SOP document can be exported to the company's knowledge base (Confluence, Notion) or as a PDF/Word for sharing. We might integrate with an LMS (Learning Management System) if the context is training (so the content can be uploaded as a course module). If there are images or diagrams, those are embedded (perhaps the AI could even prompt "Do you have any reference images I should include for step 4?"). The output should adhere to any template the company uses for SOPs – maybe allow the user to specify a template or format style at the start.

- **Trust & Reliability:** Accuracy and completeness are paramount – missing a critical step in an SOP can be dangerous. To ensure reliability, the AI can employ **teach-back**: after assembling the steps, the AI might "read back" a summary: *"So the procedure I have is: 1) do X, 2) do Y, 3) do Z. Does that sound correct and complete?"*. This catches omissions. Also, we might include a feature where the expert can highlight or mark any part of the output that needs checking, and the AI will prompt further on that. The AI should avoid any information that wasn't provided – e.g., not inventing a step. If something seems implied but unsaid, it should explicitly ask. We might also provide a **confidence indicator** or quality score for the output (maybe via a secondary model evaluation) to flag if some steps seem under-specified. For example, if the AI is unsure about a detail, it highlights it for the user to confirm. Citing external standards if relevant (like "according to ISO guidelines for this process…") could bolster trust, but only if we have those references available reliably. In essence, the expert should feel *"this captures what I said and doesn't add random stuff"*.

Furthermore, research in AI-assisted requirements gathering emphasizes the importance of human verification – we'd apply that here: the AI makes it easy for the expert to verify every part of the output, thus combining AI speed with human judgment for accuracy [28] [29].

f. Opinion Research → Stance Map + Summary Report

Scenario: A researcher or analyst is collecting a range of opinions on a topic (e.g. employee opinions on a new policy, or industry experts' stances on a trend). The goal is to map out different stances and produce a summary of the overall perspective landscape. The AI can either interview one knowledgeable person to capture multiple viewpoints or synthesize across multiple interviews.

- **Chat Behavior:** The AI functions as a **discussion moderator/analyst**. It might start by clarifying the topic: *"We're examining opinions on remote work policy. Are we gathering others' opinions through you, or your own analysis of many voices?"*. Suppose the user has already done several interviews or surveys; the AI will then ask the user to relay the key points of each perspective: *"Tell me about the main distinct viewpoints that emerged. What arguments support each? Any notable quotes or data?"*. If instead the user is a single domain expert describing the debate, the AI prompts them to articulate each side: *"What is the pro-remote stance and their reasons? What about the anti-remote stance? Any middle-ground positions?"*. The AI organizes the conversation by stance: focusing on one position at a time, ensuring depth of reasoning for each. It will also ask for the prevalence or weight of each stance if known (e.g., *"What percent of people favored each*

side, if we know?"), to help with mapping. The tone is inquisitive and impartial, seeking to fairly represent each distinct opinion.

- **Output Schema: (1) A Stance Map** – this could be a visual or structured representation of the spectrum of opinions. For example, it might present a table or chart: columns for each stance (Position A, Position B, etc.), and rows summarizing their *core claim, supporting points, concerns, and notable quotes*. Alternatively, it might be a pro-con list if it's basically two opposing sides, or a mind map diagram (if we embed an image) showing how stances relate. The map is essentially a structured comparison. **(2) A Summary Report** – a written analysis that summarizes the findings: an introduction stating the context, a section for each stance explaining it in narrative form, and a conclusion that might suggest any consensus or highlight the divergence. This reads like a brief you'd give to an executive to quickly get the gist of "where people stand on X issue."

- **"Wow" Moments:** One wow moment is how the AI can bring coherence to many viewpoints. If the user dumps a chaotic set of opinions, the AI output grouping them into clear themes is impressive. For instance, *"It seems the opinions cluster into three groups: 1) those fully in favor of remote work (citing flexibility and productivity), 2) those fully against (citing culture and collaboration loss), and 3) a hybrid stance (remote OK but with regular in-office meetings)."* Seeing that **structured clustering** emerge feels like a powerful insight extraction – akin to what a human analyst might take days to conclude, done in real-time [30] [31]. Another wow factor: the stance map may highlight **nuances**. For example, within the pro-remote group, the AI might note a split between "cost-saving focus" vs "employee happiness focus" – sub-stances that even the user might not have categorized explicitly. If the AI can point that out, it demonstrates deep analysis. Also, the summary report's polish – quoting a particularly poignant comment for illustration and providing an impartial tone – will impress users as it reads like an expert research summary.

- **Export & Integration:** The stance map (if textual like a table) can be exported to a document or slide deck. If we manage a visual (perhaps using an integration to generate charts), that can be saved as an image/PDF. The summary report can be exported to Word or PDF. Potential integrations: if this is academic or policy research, integrate with reference management (maybe the AI can insert reference codes if the user provided data from sources). Or if for internal use, post the summary to an internal blog or newsletter.

- **Trust & Reliability:** Representing opinions carries the risk of bias or misrepresentation. The AI must be **fair and accurate**: it should use the language that each stance uses, not skew it. We could implement a **validation step** where the user is shown the stance breakdown before the full report is written: *"I've identified these distinct positions: A, B, C. Did I capture those correctly?"*. Also, the AI might explicitly ask if there are any **key voices or quotes** to include for authenticity, preventing it from hallucinating any quotes. If multiple interviews are synthesized, the user might feed those transcripts or notes – the AI could cite them in the output (at least as "One participant noted X"). To increase trust, the summary could include direct small snippets from real people (if available) rather than entirely AI-generated paraphrasing – this shows the output is grounded in actual input. We also ensure any quantification (like "70% favored X") is either provided by user or not stated at all unless clearly known, to avoid false stats. Essentially, the user should feel the summary is a faithful mirror, not a distortion. Given that our AI is functioning as an analytical tool here, we might incorporate known techniques from qualitative research (the AI could even mention a method: *"Using thematic analysis, we derived 3 themes..."* to echo how a human would do it [32] [33], lending credibility). Ultimately, by engaging the user in confirming the stance map and by quoting source opinions, we maximize reliability of the output.

Additional Demo Use Cases

Beyond the six tracks above, our platform's chat-driven insight capture can apply to many scenarios. Here are a few more use cases we can demonstrate to show versatility:

- **7) User Persona & Journey Mapping:** The AI can conduct an interview with a UX researcher or product marketer to create **persona profiles and customer journey maps**. For example, after user interviews, the researcher tells the AI about different user types, their behaviors, goals, pain points. The AI then produces a set of **Persona documents** (with name, photo placeholder, demographics, needs, frustrations) and a **Journey Map** for each (stages like Awareness -> Onboarding -> Active Use, with user thoughts/feelings at each stage). *Wow moment:* automatically generating empathy maps or journey diagrams from qualitative input. *Reliability:* the AI ensures each insight is grounded in the research fed by the user, not stereotypes, by confirming details for each persona.

- **8) Project Kickoff → Charter & Risk Register:** Using the chat at the **start of a project** to draft a Project Charter (defining goals, scope, stakeholders) and a Risk Register. The AI asks a project lead about objectives, team members, known risks, and mitigation plans. Output: a **Charter document** (with background, objectives, scope, team, timeline) and a **Risk Register** table (listing risks, likelihood, impact, owner, mitigation). *Wow moment:* identification of typical risks based on project type (AI might even suggest risks the user didn't mention, asking for confirmation – showcasing its knowledge of similar projects). *Integration:* Could export risk items to a tracking system. *Trust:* The user confirms any suggested risk is relevant; AI clearly marks any assumptions.

- **9) Case Study Generation:** For marketing teams, the AI can interview an account manager or customer success rep about a customer's success story and produce a **Case Study** document. Questions cover the client background, challenge, how our product/service helped, results (with metrics), and a quote from the client. The output is a ready-to-publish case study narrative. *Wow moment:* turning raw anecdotes into a polished story with a logical flow (Challenge → Solution → Results) in seconds. *Reliability:* Ensuring any ROI or metric cited was provided and accurate; if not provided, AI prompts for it or leaves a placeholder.

Each of these additional demos underlines the core value proposition: **adaptive dialogue that yields structured, ready-to-use outputs**. They expand our reach into areas like UX research, project management, and marketing, showing that the platform can be a general "insight-to-artifact" generator across domains.

For each use case – including these additional ones – we emphasize a few common strengths of our product: - The chat experience uses **intent modeling and dynamic questioning** to adapt to the user (reflecting state-of-the-art intent engineering so the AI knows what to ask next for the desired output). - The output follows an **established schema or best-practice template**, ensuring the result is immediately useful (e.g., PRD template, Journey map format, etc.). - We build in "**wow**" factors like remembering context deeply, producing polished language, and perhaps light creativity (where appropriate, like visually organizing information). - We incorporate **trust features** like confirmation summaries, source traceability (e.g., linking a case study quote to which customer said it), and the option for the user to iterate ("regenerate with changes") safely.

By walking through these demos, internal stakeholders can envision exactly how the product functions and the breadth of its applicability. It's not just a chatbot or just a document tool – it's an **AI collaborator** that interviews you (or your data) and does the heavy lifting of documentation and insight

organization. The defined demo tracks provide a focused way to build and showcase the product initially, hitting the sweet spots we identified in the market analysis (product management, feedback, ops, knowledge transfer, etc.). As we progress, we can expand to more use cases, but these examples will guide our development and marketing in the early stages.

5. Relevant Research and Innovation Context

Our approach sits at the cutting edge of AI application in requirement elicitation, knowledge engineering, and conversational agents. To build a product grounded in proven concepts, we consider key insights from academic and industry research:

- **LLMs in Requirements & Intent Engineering:** Recent studies indicate that large language models like GPT-4 can significantly streamline the process of eliciting and documenting requirements [34] [28]. They have shown *efficiency in asking the right questions and accuracy in capturing user needs*, improving communication among stakeholders in software projects [28]. This reinforces our chat interview approach – essentially a form of automated requirements workshop. However, research also cautions about limitations: LLMs may introduce inaccuracies or ambiguities if not guided properly [29]. This underlines why our product includes confirmation steps and trust features. Prompt engineering techniques and **prompt patterns** have emerged to guide LLMs toward desired outputs [35]. We will leverage these findings – for example, using a structured prompt template for each demo use case (ensuring the AI asks about who/what/when for postmortems, or sections for PRDs). Moreover, **iterative development with human feedback** is highlighted as crucial [36]. Our chat loops (where the AI asks and refines based on user input) align with an iterative paradigm of requirement refinement, as recommended by research for LLM usage.

- **Agentic Chat Behavior & Conversational Structure:** Our product envisions AI agents that conduct interviews with human-like adaptability and empathy. A 2025 study introduced an *"Interview Bot"* and found that **LLM-based chatbots can effectively engage participants in qualitative interviews, collecting meaningful data** – though not yet perfectly replacing human interviewers [24]. The chatbot was designed to mimic human techniques (follow-up probes, showing understanding) and could adapt questions based on responses [37]. This validates our approach of an **adaptive interviewer AI**. It also suggests areas to be mindful: the study noted current bots sometimes fall short of human interviewers in fully open-ended settings [38]. We mitigate this by focusing our bot on more structured outcomes (which gives it a clear goal). Another relevant concept is **conversation memory and context management**. Agents engaged in long conversations need to manage context carefully, using strategies like summarizing earlier parts and focusing on relevant info [26] [27]. We plan to implement a memory module so that our AI can handle lengthy interviews without losing track – for instance, using summarization of earlier answers to stay within context window, and retrieving important points when needed. Academic surveys on multi-turn LLM conversations emphasize evaluating consistency and the agent's ability to handle many turns [39], which we will use as internal benchmarks (e.g., ensuring our agent doesn't contradict something said 10 turns ago – possibly via a **self-consistency check** in the conversation).

- **Insight Extraction & Structuring:** There's ongoing research on using LLMs for **insight extraction from unstructured data**. One approach uses multi-LLM setups to improve key insight identification [40]. While our case involves a human in the loop (the user provides info via chat), the underlying challenge is similar: how to go from raw inputs to distilled insights. We can draw from techniques like **thematic analysis** assisted by AI [33] – essentially what our stance

mapping does – and ensure our AI is clustering and categorizing information in ways that mirror established qualitative analysis methods (which improves credibility of outputs). The viability of our approach is further supported by industry use: e.g., **Viable's platform** uses fine-tuned LLMs to analyze support tickets and feedback, delivering nuanced, actionable insights [41] [42]. They've shown that GPT-4 level AI can categorize data into themes and answer complex questions about it [10], which is analogous to our AI asking "does this feedback imply a requirement or an action item?" during an interview. This gives confidence that with proper tuning, our system can reliably perform the insight synthesis we need.

- **Context & Memory in AI Agents:** Our product effectively creates **personalized AI agents per session** (or per user). Research and engineering blogs talk about AI agents requiring memory beyond the immediate prompt. Solutions like context engineering, scratchpads, and long-term memory stores are emerging [43] [44]. We plan architecture (discussed below) that uses a *session memory object* (perhaps Cloudflare Durable Objects or a vector store) to accumulate conversation state. For example, if a user comes back later to update a PRD, the agent can recall the prior interview's key points. This aligns with the trend of giving agents persistent memory [45] [46]. Also, our conversation structure is akin to a **semi-structured interview** – which in UX research is prized for balancing consistency with adaptability. It's noted that adapting questions and asking follow-ups leads to richer insights that purely fixed questionnaires would miss [47] [48]. Our design essentially automates a semi-structured interview, and we lean on those principles (have a guide, but allow dynamic probing). This should maximize the quality of captured insights, as confirmed by research in qualitative methods.

- **Reliability and Trust in AI-Generated Artifacts:** There is growing research on ensuring LLM outputs are factual and consistent. Techniques like contradiction detection within generated text [49] or using a second LLM as a "judge" for consistency are being explored [50]. We intend to incorporate lightweight versions of this: e.g., after drafting a document, run a consistency check to catch obvious contradictions or unanswered questions (an internal eval could flag "Section 2 mentions X, but Section 5 doesn't include it where expected"). Also, **human-AI collaboration** research suggests keeping the human in the loop for validation leads to the best outcomes – our approach always lets the user review and edit the AI's output, using the AI more as an amplifier than an autonomous final decision-maker. By referencing these studies and implementing their recommendations (such as structured prompt patterns, memory strategies, and eval loops), we position our product on a solid foundation of what's been proven to work and what pitfalls to avoid (like unchecked hallucinations or bias introduction).

In summary, our product concept is reinforced by current research: LLMs can effectively elicit requirements and insights [28], AI interviewers can scale qualitative data collection [51], and proper context management is key for long conversations [44]. We will continue to monitor and incorporate new findings (e.g., improvements in "LLM as a coach" or intent-detection algorithms) to maintain a cutting-edge yet reliable system. Our goal is not to do academic R&D, but to **apply the best of AI research pragmatically**: ensuring our adaptive interviews yield high-quality, trustworthy content that aligns with both industry best practices and scientific understanding of AI's strengths and weaknesses.

6. Technical Considerations

To deliver this product, we propose a robust yet agile technical architecture, along with an evaluation framework to ensure quality. We also outline the use of modern AI orchestration tools (Vercel AI SDK, DSPy) for efficient development and optimization.

Architecture Overview: The app will be built as a **full-stack application on Cloudflare and Next.js**, leveraging Cloudflare's edge network for global performance and scalability: - **Frontend:** A Next.js application (React) for the chat UI and dashboard. Next.js gives us SSR (server-side rendering) for fast initial loads, and its component model suits building an interactive chat interface. We'll deploy the frontend to Cloudflare's infrastructure (using Cloudflare Pages or the Next.js -> Cloudflare Workers adapter [52]). This ensures that users in APAC (and globally) get low-latency access due to Cloudflare's CDN/edge. - **Backend (API Layer):** We utilize **Cloudflare Workers** (serverless functions at the edge) to handle API requests, including orchestrating chat sessions and calling AI models. Cloudflare Workers provide high scale and run in Cloudflare's data centers worldwide (low latency for model queries, which is important for a snappy chat feel). Each chat session can be anchored by a **Cloudflare Durable Object**, which acts as a stateful coordinator – perfect for maintaining conversation state (memory) and caching intermediate results. Durable Objects ensure that each user's conversation is routed to the same instance with state, and can scale to millions of such agents as needed [53] [54]. - **AI Model Hosting:** We have two main options: use an external AI API (OpenAI, Anthropic, etc.) or Cloudflare's own Workers AI. Cloudflare recently introduced Workers AI which can host certain models on their edge (with support for OpenAI API compatibility) [55] [56]. For latency and cost, we might experiment with hosting smaller models (for quick prompts or summarization) on Workers AI, while calling larger models (GPT-4 or domain-specific LLMs) via API when needed. Notably, Cloudflare Workers now support **streaming responses** well, especially with the Vercel AI SDK integration [57], so we can stream the chatbot's answer token-by-token for responsiveness. - **Data Storage:** Beyond Durable Objects for ephemeral conversation state, we'll use **Cloudflare D1 (or Workers KV)** for persistent storage – e.g., storing user profiles, chat transcripts (with user permission), templates, and generated documents. For searchability and vector embeddings (to enable context retrieval of past chats or corporate knowledge), we might integrate a vector database (could be an external one or use Durable Objects with vector libs if small scale). - **Security & Compliance:** Running on Cloudflare gives us built-in security (WAF, DDoS protection). We'll ensure data is encrypted at rest (Cloudflare's infrastructure and any external DB). For enterprise, we might allow a self-hosted option later (deploying a containerized version in their cloud), but initially multi-tenant cloud is fine with strict data isolation per tenant.

AI Orchestration and Development Tools: We plan to utilize modern frameworks to build and optimize our AI agent: - **Vercel AI SDK:** This SDK provides convenient React hooks and utilities for chat applications (it's designed to work with Next.js). We'll use it on the frontend to handle streaming and to easily call our backend AI endpoints. The Cloudflare workers AI provider is compatible with Vercel's SDK, enabling us to call models in the same way as OpenAI's API but routed through Cloudflare's edge [58]. The SDK also simplifies handling of aborts, retries, and updating UI with token streams. By adopting it, our small team avoids writing a lot of boilerplate for managing the chat connection – it *"makes it easy to use Workers AI the same way you'd call any other LLM"* [58], which speeds development. - **DSPy (Declarative Self-Proving Python):** This is an emerging framework for modular AI systems [59]. While our stack is JS/TS for the web app, we can use DSPy concepts for orchestrating prompts and ensuring reliability. DSPy allows one to define the AI reasoning steps as code modules rather than giant prompt strings, and it can auto-optimize prompts and even do few-shot learning under the hood [59] [60]. For instance, we could use a DSPy **ReAct** or **ChainOfThought** module [61] [62] for complex tasks: the agent could internally break down tasks (first gather requirements, then format PRD). Also, DSPy's **evaluation algorithms** (like GEPA, which optimizes prompts via trials [63]) might help us improve prompts systematically rather than manually tweaking. In practice, we might integrate DSPy for agent logic on the backend (it's Python, so possibly run it in a container or via Pyodide in Worker if feasible). If that's complex, at least we borrow the approach: define structured prompt templates and use optimization techniques described by DSPy to refine them. The goal is to **iterate fast on prompt quality** using a

code-driven approach instead of ad-hoc trial/error with raw strings – this should yield more reliable outputs.

- **Multi-Model Strategy:** We might use different models for different tasks to optimize cost and performance. For example, use a cheaper model for straightforward tasks (extracting bullet points, grammar fixing) and a powerful model for complex generation. Our architecture can facilitate tool use: Cloudflare Workers AI supports function calling and even tool usage patterns [56] [64]. We could allow our agent to call an external knowledge base or perform calculations if needed (like if user says "calculate NPS from these scores" in feedback use case, the agent could do math). This agent-with-tools approach is at the frontier of AI – Cloudflare is literally promoting Workers as *"the best platform for building AI agents"* with tool calls and long context [55] [56]. We should design our system to leverage that: perhaps each domain use case has some tools (e.g., a calendar tool for timeline generation, a knowledge lookup tool for industry data in proposals, etc.). We'll use frameworks (LangChainJS or the Cloudflare `agents-sdk` [65]) to implement these agents efficiently.

Evaluation & Quality Assurance Plan: Delivering execution-ready artifacts means quality is paramount. We devise a multi-pronged evaluation approach: - **Automated Quality Scoring:** Every structured output can be run through an evaluation routine. For example, use an LLM-as-a-judge approach on key dimensions – coherence, completeness, correctness. OpenAI's Evals or similar frameworks can be employed: e.g., define an eval that checks if all required sections of a PRD are present and not contradictory. There's an OpenAI eval example for structured outputs [66] – we could adapt that to say "score this PRD vs an ideal answer if we have one". Initially, we might not always have a ground-truth reference, but we can at least ensure format correctness (e.g., JSON validity if we use JSON internally – Cloudflare's JSON mode for LLMs can enforce schema [67] [68]). We'll integrate such checks into development: perhaps using a test suite of sample interviews and verifying the outputs meet certain criteria (no TBD sections unless intentionally, actions items phrased with a verb, etc.). Over time, as we gather real outputs and feedback, we can train a model or use heuristics to predict a quality score and flag low-scoring outputs for review. - **Human in the Loop Feedback:** Since our tool will be used internally by our team initially (and maybe beta users), we can collect qualitative feedback. We should instrument the app to allow users to rate the output or highlight issues ("This summary missed something" or "This part is wrong"). That data can feed back into prompt adjustments or model fine-tuning. Essentially, we apply the RLHF (Reinforcement Learning from Human Feedback) concept at a small scale – not actually training the base model, but iteratively improving our prompts and rules. - **Contradiction & Hallucination Detection:** We will implement specific safeguards: for contradiction, as noted, a second-pass check by an LLM (or even deterministic rules) to find inconsistent statements. For factual hallucination, particularly in scenarios where facts matter (e.g., proposals with metrics, case studies with client names), we plan to constrain the AI. One method is Cloudflare Workers AI's **JSON mode with schema** [67] – for example, we can force the model to output certain fields and types, reducing chance to go off-script. Another method is few-shot prompting with examples that demonstrate how to handle unknown info (e.g., "<if unknown, ask or mark TBD>"). We could also maintain a list of *forbidden content* to ensure it doesn't slip irrelevant info (like it shouldn't produce company financials in a PRD unless provided). - **Confirmation Loops and User Approval:** As described in use cases, the system often presents an outline or summary for user confirmation before finalizing the detailed doc. This is part of quality control – by getting user sign-off on structure and key points, we greatly reduce the chance of major errors in final output. Additionally, for critical outputs (say a big enterprise proposal), the tool could even suggest a review: *"It's always good to have a human review the final document for nuances. Do you want to share it with a colleague for feedback?"* (Maybe even facilitate that via a link). Encouraging a human loop isn't a failure; it's a feature to ensure reliability for high-stakes documents. - **Continuous Learning and Prompt Optimization:** Using frameworks like DSPy or OpenAI Evals, we'll continuously refine our prompts and agent logic. DSPy can help with **prompt**

optimization algorithms that adjust the wording or structure to improve outcomes [59] [60]. For example, we could A/B test different interview styles (more open vs more direct questions) to see which yields more complete info from users. We can also track metrics like average follow-up questions needed, or how often users manually edit outputs, as proxies for prompt effectiveness. This will be an ongoing engineering effort – effectively treating prompts as code, with version control and improvements over time.

Deployment and DevOps: Cloudflare + Next.js gives us a modern CI/CD pipeline. We can use tools like Cloudflare Wrangler to deploy backend changes, and Vercel or Cloudflare Pages for front-end continuous deployment. The global nature of our deployment means we should test in various network conditions – we might simulate usage from ASEAN vs US to ensure latency is acceptable. Next.js serverless functions can also run on Cloudflare Workers (via adaptation) so some logic might be in Next backend, but likely we'll centralize in Workers for clarity.

Scalability: With the architecture chosen, we can handle scale. Each conversation is lightweight text until final output, and heavy compute (AI model) can be scaled horizontally: Cloudflare Workers AI can scale across their GPU cluster, and/or external API calls can scale with those providers. Durable Objects scale by sharding per session or user. Millions of concurrent sessions are feasible (Cloudflare claims tens of millions of isolate instances in parallel). So as we gain more users, we mostly watch our API costs (model inference costs) and optimize those (caching partial results, using cheaper models when possible, etc.). We will also employ caching for static content and perhaps for repeat questions (if many users ask similar things, though each interview is unique, some suggestions or boilerplate outputs can be cached).

Using Cloudflare's AI Tools: We'll stay close to Cloudflare's evolving features. For example, their `agents-sdk` (in beta) helps build AI agents that maintain state and even schedule tasks [65]. We can use this to allow long-running processes (maybe an agent that continues research in background or schedules a follow-up interview email). Also Cloudflare's AI gateway and monitoring will help us track usage and performance (they now have usage stats and even AI Gateway integration for monitoring calls [57]). This is useful for both cost tracking and reliability (e.g., if a model call fails or is slow, we get analytics). Essentially, by building on Cloudflare, we get a lot of infrastructure out-of-the-box and can focus on our application logic.

Diagram – High-Level Architecture: *(If an architecture diagram were to be included, it would show: User browsers (in various regions) -> hitting Cloudflare Edge -> Next.js front-end (static content from Cloudflare CDN, dynamic chat via Workers). The Workers environment has a Chat Orchestrator (Durable Object) for each session, which communicates with either Workers AI (hosted models) or external AI APIs. Data stores (D1/KV) attached for persistence. The front-end uses Vercel AI SDK to open a streaming connection to the Workers for chat. This flows both ways with minimal latency due to edge deployment.)*

In conclusion, our technical plan is to **build on a modern, globally-distributed stack** that ensures our AI interviews are responsive and scalable. We will employ the latest frameworks to orchestrate complex AI behaviors (ensuring modularity and upgradability of our AI prompts/logic). By implementing strong evaluation and feedback loops, we aim for high quality from day one, and only improve over time as we gather more data. This architecture not only serves our MVP but is robust enough to grow into an enterprise-grade solution, aligned with Cloudflare's vision of hosting AI agents at the edge for speed and security [55]. It positions us well to deliver a reliable, fast, and delightful experience to users across ASEAN and the world.

---

1 Global AI Meeting Assistants Market Size, Share, and Trends ...
https://www.databridgemarketresearch.com/reports/global-ai-meeting-assistants-market?srsltid=AfmBOooqRIMUA7DOeLqguWExJeXJny46BtH0rtsDQ2dQaZ6AryFhc-07C

2 3 21 How Typeform Stands Out In A Crowded Market
https://ducttapemarketing.com/how-typeform-stands-out-in-a-crowded-market/

4 5 17 Plans & Pricing | Typeform
https://www.typeform.com/pricing

6 Top 7 Dovetail Alternatives & Competitors - Kimola Blog
https://kimola.com/blog/dovetail-alternatives-and-competitors

7 Pricing - Dovetail
https://dovetail.com/pricing/

8 Information on our new pricing plans - Dovetail
https://dovetail.com/blog/information-on-our-new-pricing-plans/

9 10 41 42 Accurately analyzing large scale qualitative data | OpenAI
https://openai.com/index/viable/

11 Learn about the Fireflies' pricing plans
https://guide.fireflies.ai/articles/3734844560-learn-about-the-fireflies-pricing-plans

12 Fireflies AI Pricing 2026: Complete Breakdown & Analysis - Outdoo AI
https://www.outdoo.ai/blog/fireflies-ai-pricing

13 Sector Deep Dive #8: POST-TRAINING INFRA - by Prateek Joshi
https://www.infrastartups.com/p/sector-deep-dive-8-post-training

14 How you will win the AI agents race: Fast Feedback Loops - LinkedIn
https://www.linkedin.com/pulse/how-you-win-ai-agents-race-fast-feedback-loops-diksha-singh-nkdjf

15 Dovetail Pricing Changes - PriceTimeline
https://pricetimeline.com/data/price/dovetail

16 Fireflies.ai Pricing: A Full Breakdown of Plans and Costs | AFFiNE
https://affine.pro/blog/fireflies-ai-pricing-tips

18 Dovetail Reviews, Pricing & Features - 2025 | TEC
https://www3.technologyevaluation.com/solutions/58521/dovetail?srsltid=AfmBOoqbY0hqqBua1S3EiIWEwjCEpuhLfIbr_PWxcTVwTH-g2DWSrbTx

19 Fireflies.ai Pricing Plans Comparison and Cost Guide
https://www.cloudeagle.ai/blogs/blogs-fireflies-ai-pricing-guide

20 The 95:5 rule is the new 60:40 rule - Marketing Week
https://www.marketingweek.com/peter-weinberg-jon-lombardo-95-5-rule/

22 Product-led Sales: An In-depth Blueprint for Action - OpenView
https://openviewpartners.com/blog/product-led-sales-blueprint/

23 Mastering Product-Led Growth - Monetization - My PM Interview
https://www.mypminterview.com/p/mastering-product-led-growth-monetization

24 37 38 51 scitepress.org
https://www.scitepress.org/Papers/2025/133878/133878.pdf

25 26 27 43 44 45 46 Context Engineering
https://www.blog.langchain.com/context-engineering-for-agents/

[28] [29] [34] [35] [36] Using ChatGPT in Software Requirements Engineering: A Comprehensive Review
https://www.mdpi.com/1999-5903/16/6/180

[30] [31] [32] [33] Semi-structured interview analysis methods for in-depth insights - Insight7 - Call Analytics & AI Coaching for Customer Teams
https://insight7.io/semi-structured-interview-analysis-methods-for-in-depth-insights/

[39] Evaluating LLM-based Agents for Multi-Turn Conversations: A Survey
https://arxiv.org/html/2503.22458v1

[40] A scientific-article key-insight extraction system based on multi-actor ...
https://www.nature.com/articles/s41598-025-85715-7

[47] [48] Semi-structured interviews: The UX researcher’s secret weapon | by MMU RASHID | Bootcamp | Medium
https://medium.com/design-bootcamp/semi-structured-interviews-the-ux-researchers-secret-weapon-eb6a140bd7a8

[49] Contradiction Detection in RAG Systems: Evaluating LLMs as ... - arXiv
https://arxiv.org/html/2504.00180v1

[50] 7 Strategies To Solve LLM Reliability Challenges at Scale - Galileo AI
https://galileo.ai/blog/production-llm-monitoring-strategies

[52] Deploy Next.js to Cloudflare Workers: OpenNext Adapter
https://blog.cloudflare.com/deploying-nextjs-apps-to-cloudflare-workers-with-the-opennext-adapter/

[53] [54] [55] [56] [57] [58] [64] [65] [67] [68] Making Cloudflare the best platform for building AI Agents
https://blog.cloudflare.com/build-ai-agents-on-cloudflare/

[59] [60] [61] [62] [63] DSPy
https://dspy.ai/

[66] Introducing Structured Outputs in the API - OpenAI
https://openai.com/index/introducing-structured-outputs-in-the-api/