AI Tool Match — Product spec, UI mockups & tech stack

Goal: Global web app where a user types a task prompt (natural language) and the system recommends the *best* AI tool(s)/apps/sites for that task, with ranked reasons, filters and quick links.

---

1. Core value propositions

- **Fast, confident matching:** Convert a single free-text prompt into 3–5 ranked tool recommendations with reasons.
- **Explainability:** For each recommendation, show why it fits (capabilities, price tier, ease, languages, integrations).
- **Always-fresh catalog:** Continuously updated AI tool directory with metadata and signals (ratings, price, capabilities).
- **Search + wizard:** Users can either type freely or use a short guided wizard to refine constraints (budget, output type, preferred platform).

---

2. Target users

- Product managers, marketers, developers, creators, students, enterprises looking to apply AI without researching dozens of tools.

---

3. MVP feature list (must-have)

1. Free-text prompt input (single field)
2. Optional quick constraints UI (budget, realtime vs batch, mobile/desktop, required integrations, language)
3. NLP parser that maps prompt → intent tags + entities (e.g., "generate social media captions" → intent: text_gen, format: short, tone: marketing)
4. Tool database (seeded with 500–3000 tools) and structured metadata (categories, languages, pricing, free tier, integrations, supported formats, latency)
5. Recommender engine: rule layer + vector similarity + heuristic ranking → top 3 recommendations with score and reasoning bullet points
6. Filters and sort (price, ease, rating, privacy/enterprise-ready)
7. Detail page for each tool: quick facts, pros/cons, links, screenshots, API availability
8. Feedback capture (Was this helpful?) to improve ranking
9. Admin UI to add/edit tools, review user feedback, run audits
10. Analytics dashboard (CTR, conversion, satisfaction, change over time)

---

4. Nice-to-have (post-MVP)

- Live comparisons (side-by-side feature matrix)

- Demo flows / one-click trial (open tool in new tab with prefilled prompt where supported)
- Community reviews and use-case templates
- Personalised profile & history (save searches, favourite tools)
- Enterprise tier with team controls and private catalog
- Browser extension and mobile apps

---

5. User journeys (quick)

A. Fast match: Type prompt → system shows 3 ranked tools + “Why this” bullets → user clicks tool → visits tool site.

B. Wizard refine: Type prompt → ask two clarifying buttons (budget? platform?) → refined recommendations.

C. Trial + feedback: User clicks “Tried it” → rate success → data used to retrain ranking.

---

6. Example UI mockups (wireframes described)

Landing / Prompt screen

- Top: short headline + one-line value prop.
- Center: big input box with placeholder: *"Describe what you want to do (e.g. "Create 5 Instagram captions for a travel brand in friendly tone")"*.
- Below: quick constraint chips: Budget (Free / <$10 / $10–50 / >$50), Platform (Web / Mobile / API), Output (Text / Image / Video / Code), Integrations (Zapier / Google Sheets / Notion).
- CTA: Recommend button.

Results screen

- Left column: 3 recommended cards (rank 1 → 3), each card shows: logo, one-line summary, 3 bullets why good, estimated cost bracket, key feature icons (API, mobile, languages), score meter, action buttons (Open / Compare / Save).
- Right column: detail area for selected tool with screenshots and quick facts.
- Top: filter bar and a “Regenerate” / edit prompt button.

Tool detail page

- Header: logo, name, pricing badge, trust badges.
- Middle: short summary + 6 bullets (why recommended for this prompt). Show data points used (matching tags, vector similarity %).
- Bottom: alternatives, similar prompts where this tool worked.

---

7. Data model (simplified)

- **Tool**: id, name, url, categories[], supported_outputs[], integrations[], pricing_tier, free_tier(Boolean), languages[], tags[], description, last_verified_date, logo_url, api_available(Boolean), avg_latency_ms, enterprise_ready(Boolean), ratings[]

- **PromptLog**: id, user_prompt, parsed_tags[], constraints, timestamp, recommended_tools[], chosen_tool_id (nullable), feedback (success: bool, text)
- **Tag taxonomy**: hierarchical categories (text_gen → longform/shortform; vision → image_gen/segment/ocr)

---

8. Matching algorithm (hybrid approach)

1. **NLP parse (intent extraction):** Use an LLM (few-shot prompts or fine-tuned model) to output standardized tags and constraint fields.
2. **Vector search:** Embed the user prompt and all tool use-case descriptions (and example prompts) and retrieve tools with high cosine similarity.
3. **Rule layer:** Hard filters for must-have constraints (e.g., must have API, must be < $10), exclude mismatches.
4. **Heuristic ranking:** Combine vector similarity, tag match count, freshness (last_verified), popularity, and user context weight → final score.
5. **Explainability layer:** Generate the 2–3 bullet reasons using the LLM: cite the matching criteria (e.g., “low cost, built-for short social copy, mobile app available”).

---

9. Tech stack (suggested)

Frontend: React + Vite, Tailwind CSS, TypeScript. Optional preview component: single-file React demo for quick UX.
Backend: Node.js (NestJS) or Python (FastAPI). GraphQL or REST.
DB: PostgreSQL (primary), Redis (cache). Vector DB: Milvus or Pinecone or Qdrant.
NLP / Embeddings: Use OpenAI / Anthropic embeddings or open alternatives (Llama-2/Meta embeddings or HuggingFace inference) depending on cost / licensing. LLM for parsing & explainability: GPT-4o/GPT-4 or open options (Llama 3 + fine-tune).
Search: ElasticSearch for metadata + vector DB for semantic.
Hosting: Vercel for frontend, AWS/GCP for backend + managed DB + container infra.
Analytics: PostHog / Google Analytics; mixpanel for advanced funnels.
Monitoring: Sentry, Prometheus.
CI/CD: GitHub Actions.

---

10. Operational considerations

- **Tool ingestion:** manual curation + web scraping + partner APIs; store citations & last_verified_date.
- **Verification pipeline:** periodic checks (automated pings + human review) to keep metadata fresh.
- **Bias & transparency:** show if recommendation is sponsored/affiliate; provide rationale and confidence score.
- **Legal & privacy:** don’t store PII in prompts unless explicitly consented; provide opt-out and data deletion.

---

11. Monetisation ideas

- Affiliate / referral fees (disclose).
- Sponsored placements (clearly labelled).
- Premium tier: advanced filters, enterprise catalog, team features.
- API access for SaaS products to call your matching engine.

---

12. MVP roadmap (timeline estimate — high level)

- Week 0–2: product specification, taxonomy, seed tool database (100–500 tools) + basic frontend mockups.
- Week 3–6: implement prompt→NLP parse pipeline + vector embedding pipeline + initial ranking rules; basic UI for input + results.
- Week 7–10: admin dashboard, feedback loop, analytics, expand tool catalog, integrate vector DB.
- Week 11–16: polish, performance, paid features, public launch.

---

13. Key KPIs to track

- Prompt→click conversion rate (users who click a recommended tool).
- Recommendation satisfaction (thumbs up/down %).
- Time to first click (speed).
- Repeat usage rate and saved searches.
- Accuracy proxy: % of times top-3 contained the chosen tool.

---

14. Risks & mitigations

- **Stale data** → automated verification + crowdsourced reporting.
- **Wrong recommendations** → feedback loop + allow user to refine prompt / apply filters.
- **Legal / affiliate bias** → transparency, separate sponsored listings.

---

15. Next steps I can do for you right now

- Produce a clickable React single-file prototype of the prompt → results screen (can be previewed). *(I can create this in canvas if you want.)*
- Create an initial taxonomy + seed dataset (CSV) of 200 popular AI tools with tags and example prompts.
- Draft sample LLM prompt templates for intent extraction and explainability.

End of spec.