ChatGPT

AI-Assisted Technical Assessment – Sample Tasks & Rubrics

This companion document provides **examples of on-the-spot exercises** that can be used to evaluate candidates' ability to collaborate with AI tools during the technical portion of a hybrid interview. Each exercise includes a brief description, the skills/values it targets, and guidance for interviewers on observation and scoring. These scenarios are designed to work in conjunction with the overarching guidance described in the **AI-Assisted Technical Assessment Guidance**.

1 Guiding principles for task design

- **Realistic and open-ended** – The exercise should mirror an actual engineering problem and allow multiple valid solutions. Avoid classic "LeetCode" puzzles that generative models can easily solve; instead, ask the candidate to extend or improve an existing codebase, integrate new features or design infrastructure.
- **Requires human judgement** – Ensure that AI assistance alone cannot produce a complete, correct solution. Include requirements such as security hardening, error handling, observability, compliance or design trade-offs that require understanding beyond code generation.
- **Prompts and iteration** – Build tasks that necessitate iterative prompts and refinement. The candidate should need to ask follow-up questions to the AI, test responses, and adjust prompts when initial output is unsatisfactory [1].
- **Opportunity for reflection** – Allow time for the candidate to explain their thought process, prompt choices, and integration decisions. This is where follow-up questions reveal deeper knowledge [2].

2 Sample exercises

2.1 Enhance an API with authentication and observability

- **Brief**: Provide the candidate with a small REST API (e.g., built with Express.js or FastAPI) that lacks authentication and has minimal logging. Ask them to implement OAuth 2.0 authentication, integrate structured logging and Prometheus metrics, and prepare the service for horizontal scaling. They may use an AI assistant to generate code, but must validate and integrate it.
- **Skills & values**: Problem framing and prompt design; critical evaluation of AI output; integration & code quality; observability & guardrails; security & compliance; communication & collaboration.
- **Expected behaviours**:
- Candidate asks clarifying questions about existing architecture and constraints (e.g., "Does the service run behind a reverse proxy? What authentication providers do we support?").
- Prompts include context (language, framework) and request security best practices (e.g., token validation, secret management).
- Candidate reviews AI suggestions, modifies them to fit the codebase, writes unit tests and ensures sensitive information is not exposed in logs. [3]
- Candidate updates documentation (README or comments) and explains their implementation.

- **Interview guidance**: Observe how the candidate decomposes the task, how they iterate on prompts when AI output isn't perfect, and whether they verify the security and performance of the code. Use follow-up questions to probe understanding of OAuth flows and logging frameworks.

2.2 Refactor AI-generated code for maintainability

- **Brief**: Give the candidate a code snippet generated by an AI tool that solves a problem (e.g., data transformation or file parser) but lacks proper structure, error handling and tests. Ask the candidate to refactor the code into maintainable functions, add error handling, write tests and document the logic. They can prompt an AI assistant for suggestions or improvements, but must make final decisions.
- **Skills & values**: Validation & debugging; integration & code quality; iteration & improvement; observability & guardrails; continuous learning & mentorship.
- **Expected behaviours**:
- Candidate identifies issues such as variable naming, repeated logic and missing error handling.
- They prompt the AI for best practices (e.g., "How can I modularize this parser into separate functions?") and compare suggestions against their knowledge.
- They introduce unit tests covering edge cases and ensure the code handles invalid inputs gracefully.
- They explain trade-offs (e.g., readability vs. performance) and document decisions.
- **Interview guidance**: Ask the candidate why they kept or discarded certain AI suggestions. Observe whether they spot security or performance issues. Evaluate their ability to write tests and improve code quality beyond the AI's initial output. [4]

2.3 Design a scalable architecture with AI assistance

- **Brief**: Present a high-level scenario: "Our platform expects a ten-fold traffic increase next quarter. Design an architecture that can handle the load, ensure reliability, and allow feature experimentation." Give the candidate access to an AI assistant and blank drawing tools or whiteboard.
- **Skills & values**: Systems thinking; innovative spirit; data-informed iteration; collaboration & knowledge sharing; customer-centric craftsmanship; observability & guardrails.
- **Expected behaviours**:
- Candidate uses the AI to research scalability patterns (e.g., microservices, message queues, caching) and asks for pros/cons of different approaches.
- They combine AI insights with personal experience to propose a design that balances performance, maintainability and cost. They consider observability (dashboards, alerts) and deployment strategies (blue/green, canary).
- They justify decisions and discuss fallback plans (e.g., graceful degradation, kill switches).
- **Interview guidance**: Focus on how the candidate frames questions for the AI (e.g., specifying context and constraints) and how they integrate AI-provided architecture patterns. Ask them to compare alternatives and explain how the design aligns with user needs and business goals.

2.4 Debug a live issue with AI assistance

- **Brief**: Provide the candidate with a small application that has a latency spike after a recent change. Ask them to identify the root cause and fix it. They can use an AI assistant for suggestions, but must validate and test any fixes.

- **Skills & values**: Observability & guardrails; ownership & proactivity; data-informed iteration; integrity & reliability; collaboration.
- **Expected behaviours**:
- Candidate investigates logs, metrics and traces to narrow down the performance regression.
- They ask the AI for potential causes of latency spikes in similar contexts and follow up with specific prompts about the codebase.
- They validate AI suggestions by profiling and monitoring. After applying a fix, they confirm that latency returns to normal.
- They document the RCA (root cause analysis) and propose preventative measures (e.g., alerts, circuit breakers).
- **Interview guidance**: Assess whether the candidate jumps straight to asking the AI for fixes or first uses observability tools to gather evidence. Evaluate how they verify AI advice and whether they communicate the debugging process clearly.

2.5 Prompt-engineering mini-challenge

- **Brief**: Provide a small generative AI model (e.g., a simplified language model or code-assistant sandbox) and ask the candidate to design prompts that produce specific outputs. For example, "Generate a Python function that sorts a list of objects by a nested field and includes inline comments explaining each step."
- **Skills & values**: Prompt quality & problem framing; critical evaluation; iteration & improvement; customer-centric craftsmanship.
- **Expected behaviours**:
- Candidate starts by clarifying the desired output format and constraints (e.g., programming language, coding style, handling of edge cases).
- They design an initial prompt, test the output, and iteratively refine wording, examples or role instructions until the output meets requirements [5].
- They examine the generated code, identify any issues, and adjust the prompt or code accordingly.
- **Interview guidance**: Observe how systematically the candidate approaches prompt design, how they evaluate the AI's responses, and how they balance specificity with flexibility. Ask them how they measure the effectiveness of a prompt and what strategies they use to prevent hallucinations.

3 Example scoring sheet

Below is a simplified scoring sheet template for a 25-point AI-assisted technical exercise. Interviewers can adapt the weighting to align with your hybrid scoring framework.

Criterion	Max points	Candidate performance
Prompt framing & clarity	5	Did the candidate ask clear, context-rich questions? Did they iteratively refine prompts?
Validation & debugging	5	Did the candidate test AI outputs, identify bugs or hallucinations, and cross-reference documentation?

Criterion	Max points	Candidate performance
Integration & code quality	5	Did the candidate integrate AI suggestions into well-structured, secure and maintainable code? Were design patterns used appropriately?
Iteration & learning	5	Did the candidate learn from initial attempts, incorporate feedback and improve their approach over time?
Communication & collaboration	5	Did the candidate articulate their reasoning, ask clarifying questions, respond to feedback and maintain integrity about AI use?

For each criterion, assign a 1–5 rating (1 = poor, 3 = competent, 5 = outstanding) and record specific examples of behaviour observed. After the interview, compare scores across candidates and calibrate with other interviewers to ensure consistency.

4 Post-interview reflection

After the AI-assisted exercise, ask the candidate to reflect on their experience. Questions could include:

- “What did you learn about collaborating with AI during this exercise?”
- “If you had more time, how would you improve your solution or prompts?”
- “How do you balance using AI for efficiency with maintaining code quality and security?”

Encourage candidates to discuss both the benefits and limitations of AI. Honest reflection can reveal continuous learning mindset and integrity.

These sample exercises demonstrate how to test a candidate’s ability to harness AI tools effectively while adhering to core values. By combining open-ended tasks, rigorous evaluation criteria and thoughtful follow-up questions, interviewers can obtain a nuanced view of a candidate’s technical capability and cultural alignment.

---

1 3 Tips To Introduce ChatGPT Into Your Technical Interviews - CoderPad
https://coderpad.io/blog/interviewing/3-tips-to-introduce-chatgpt-into-your-technical-interviews/

2 When Candidates Use Generative AI for the Interview
https://sloanreview.mit.edu/article/when-candidates-use-generative-ai-for-the-interview/

3 How to Evaluate Engineers Who Use AI Tools in Technical Interviews - Fonzi AI Recruiter
https://fonzi.ai/blog/evaluate-engineers-using-ai

4 The Complete Guide to AI-Powered Technical Interviews: How to Assess Real Developer Skills - Axiom
https://axiompro.biz/blogs/the-complete-guide-to-ai-powered-technical-interviews-how-to-assess-real-developer-skills/

5 Thoughts on using ChatGPT in job interviews - DEV Community
https://dev.to/shaharke/thoughts-on-using-chatgpt-in-job-interviews-1ejo