Accelerate Your Certification with Microsoft AI-103 Practice Questions

Last chance to prepare smart! Get your hands on free Microsoft Developing AI Apps and Agents on Azure exam PDF questions. Study real AI-103 dumps with verified answers and fast-track your certification success with PrepBolt AI-103 exam pdf questions and answers.

Thank you for Downloading AI-103 exam PDF Demo

https://prepbolt.com/AI-103.html

Question 1

Question Type: MultipleChoice

Case Study: Mix Questions

Mix Questions

AI-103 Mix Questions IN THIS CASE STUDY

Note: This section contains one or more sets of questions with the same scenario and problem. Each question presents a unique solution to the problem. You must determine whether the solution meets the stated goals. More than one solution in the set might solve the problem. It is also possible that none of the solutions in the set solve the problem.

After you answer a question in this section, you will NOT be able to return. As a result, these questions do not appear on the Review Screen.

You have a Microsoft Foundry project that contains an agent. The agent generates summaries from retrieved policy documents.

Users report that some responses omit required regulatory clauses, even when the clauses are present in the retrieved content.

You need to improve response completeness.

Solution: You increase the value of the max_tokens parameter.

Does this meet the goal?

Options:

A- Yes

B- No

Answer:

B

Explanation:

The solution does not meet the goal. Increasing max_tokens only raises the maximum number of tokens the model is allowed to generate. Microsoft's Azure OpenAI reference defines max_tokens as the maximum number of tokens allowed for the generated answer, and the quota guidance notes that increasing it can help when responses are being truncated.

In this scenario, the problem is not described as output truncation. The required regulatory clauses are

already present in the retrieved policy documents, but the agent omits them during summarization. That is a response completeness issue: Microsoft Foundry RAG evaluator guidance defines response completeness as the recall aspect of the response, meaning the response should not miss critical information compared with expected content or ground truth.

A larger token budget might permit a longer answer, but it does not force the model to identify, verify, or include each mandatory clause. It can also increase cost and latency. The appropriate control is a reflection or completeness verification pass that checks the draft against the retrieved policy clauses and regenerates or revises the response when required content is missing. Reference topics: RAG response completeness, model output limits, max_tokens, reflection, and response validation.

Question 2

Question Type: MultipleChoice

Case Study: Mix Questions

Mix Questions

AI-103 Mix Questions IN THIS CASE STUDY

Note: This section contains one or more sets of questions with the same scenario and problem. Each question presents a unique solution to the problem. You must determine whether the solution meets the stated goals. More than one solution in the set might solve the problem. It is also possible that none of the solutions in the set solve the problem.

After you answer a question in this section, you will NOT be able to return. As a result, these questions do not appear on the Review Screen.

You have a Microsoft Foundry project that contains an agent. The agent generates summaries from retrieved policy documents.

Users report that some responses omit required regulatory clauses, even when the clauses are present in the retrieved content.

You need to improve response completeness.

Solution: You add a reflection pass that regenerates the response if the required clauses are missing.

Does this meet the goal?

Options:

A- Yes

B- No

Answer:

A

Explanation:

Yes, the solution meets the goal. The problem is not retrieval availability, because the required regulatory clauses are already present in the retrieved policy documents. The failure occurs during generation: the agent produces a summary that omits required content. A reflection pass is the correct application-level control because it adds a verification step before the response is returned. The pass can compare the draft answer against the retrieved clauses, detect missing mandatory content, and trigger regeneration or revision until the summary includes the required clauses.

This aligns with Microsoft Foundry's evaluation and observability model, where generated responses are assessed for reliability, groundedness, relevance, and quality throughout the AI application lifecycle. Foundry observability guidance describes evaluation as a mechanism for measuring response quality and improving AI outputs across development and production workflows. The Azure AI evaluation SDK also defines completeness as the extent to which a generated response contains all necessary and relevant information with respect to the provided ground truth. Reflection operationalizes that quality check inside the application flow, rather than merely reporting the defect after the fact. Reference topics: model reflection, response completeness, RAG generation quality, retrieved context verification, and agent response optimization.

Question 3

Question Type: MultipleChoice

Case Study: Mix Questions

Mix Questions

AI-103 Mix Questions IN THIS CASE STUDY

Note: This section contains one or more sets of questions with the same scenario and problem. Each question presents a unique solution to the problem. You must determine whether the solution meets the stated goals. More than one solution in the set might solve the problem. It is also possible that none of the solutions in the set solve the problem.

After you answer a question in this section, you will NOT be able to return. As a result, these questions do not appear on the Review Screen.

You have a Microsoft Foundry project that contains an agent. The agent generates summaries from retrieved policy documents.

Users report that some responses omit required regulatory clauses, even when the clauses are present in the retrieved content.

You need to improve response completeness.

Solution: You increase the value of the temperature parameter.

Does this meet the goal?

Options:

A- Yes
B- No

Answer:

B

Explanation:

The solution does not meet the goal. Increasing temperature changes the sampling behavior of the generative model, not the completeness-checking logic of the application. Microsoft's Azure OpenAI reference defines temperature as a sampling control where higher values make output more random, while lower values make output more focused and deterministic. Raising the value can increase variation and creativity, but it does not ensure that all required regulatory clauses from the retrieved policy documents are included.

The reported issue is a recall/completeness failure: relevant clauses are already present in retrieved content, but the generated summary omits them. Microsoft Foundry RAG evaluator guidance defines Response Completeness as whether a response covers critical information compared to expected information or ground truth, and distinguishes it from groundedness, which checks that responses do not go beyond grounding context.

A more suitable implementation would add a reflection, verification, or completeness review pass that compares the draft summary against the retrieved clauses and revises the response before returning it. Increasing temperature could make outputs less predictable and may worsen omission risk. Reference topics: model parameters, temperature, RAG response completeness, retrieved context, and model reflection.

Question 4

Question Type: MultipleChoice

Case Study: Mix Questions

Mix Questions

AI-103 Mix Questions IN THIS CASE STUDY

Note: This section contains one or more sets of questions with the same scenario and problem. Each question presents a unique solution to the problem. You must determine whether the solution meets the stated goals. More than one solution in the set might solve the problem. It is also possible that none of the solutions in the set solve the problem.

After you answer a question in this section, you will NOT be able to return. As a result, these questions do not appear on the Review Screen.

You have a Microsoft Foundry project that contains an agent. The agent generates summaries from retrieved policy documents.

Users report that some responses omit required regulatory clauses, even when the clauses are present in the retrieved content.

You need to improve response completeness.

Solution: You run an evaluation flow that scores responses for completeness and blocks responses that fall below a defined threshold.

Does this meet the goal?

Options:

A- Yes
B- No

Answer:

B

Explanation:

The solution does not meet the goal. A completeness evaluation flow is useful for detecting incomplete responses, but detection and blocking do not improve the response itself. Microsoft Foundry RAG evaluators define Response Completeness as a metric that measures whether a response covers all critical information from the expected response or ground truth. It is a system evaluation signal used to assess response quality and produce pass/fail or scored results.

In this scenario, the issue is that the agent omits required regulatory clauses even though the clauses are present in retrieved content. Blocking low-scoring responses would prevent incomplete answers from being returned, but it would not revise the summary, add the missing clauses, or improve the generation process. The appropriate improvement is to add a response-generation control such as a reflection or verification pass that checks the draft summary against the retrieved policy content and regenerates or amends the answer before returning it. Evaluation can support the quality gate, but by itself it is an assessment mechanism, not a completeness-enhancement mechanism. Reference topics:

Microsoft Foundry RAG evaluators, response completeness, grounded generation, reflection, and response quality optimization.

Question 5

Question Type: MultipleChoice

Case Study: Mix Questions

Mix Questions

AI-103 Mix Questions IN THIS CASE STUDY

Note: This section contains one or more sets of questions with the same scenario and problem. Each question presents a unique solution to the problem. You must determine whether the solution meets the stated goals. More than one solution in the set might solve the problem. It is also possible that none of the solutions in the set solve the problem.

After you answer a question in this section, you will NOT be able to return. As a result, these questions do not appear on the Review Screen.

You have a multimodal AI generative model that accepts image uploads and uses extracted image text to generate responses.

You discover that users can upload unsafe images and embed hidden instructions into images to manipulate the model.

You need to implement controls to mitigate the risk.

Solution: You configure protected material detection.

Does this meet the goal?

Options:

A- Yes
B- No

Answer:

B

Explanation:

The solution does not meet the goal. Protected material detection is intended to identify large

language model output that matches known protected text or code, such as copyrighted text, selected web content, song lyrics, articles, recipes, or code. Microsoft describes protected material detection as a control for preventing AI-generated content from reproducing known protected material, not as a control for image safety or prompt injection.

The stated risk has two parts: users can upload unsafe images, and users can embed hidden instructions in images to manipulate the model. Unsafe image uploads require image moderation, because Azure AI Content Safety provides image APIs that detect harmful content across modalities and can support blocking decisions by harm category and severity. Hidden instructions extracted from images are indirect prompt injection or document attacks; Microsoft Prompt Shields are the capability designed to detect user prompt attacks and document attacks, including harmful instructions embedded in third-party content.

Therefore, protected material detection alone does not mitigate either primary risk. Reference topics: Azure AI Content Safety, image moderation, Prompt Shields, document attacks, indirect prompt injection, and protected material detection.

Question 6

Question Type: MultipleChoice

Case Study: Mix Questions

Mix Questions

AI-103 Mix Questions IN THIS CASE STUDY

Note: This section contains one or more sets of questions with the same scenario and problem. Each question presents a unique solution to the problem. You must determine whether the solution meets the stated goals. More than one solution in the set might solve the problem. It is also possible that none of the solutions in the set solve the problem.

After you answer a question in this section, you will NOT be able to return. As a result, these questions do not appear on the Review Screen.

You have a multimodal AI generative model that accepts image uploads and uses extracted image text to generate responses.

You discover that users can upload unsafe images and embed hidden instructions into images to manipulate the model.

You need to implement controls to mitigate the risk.

Solution: You configure a prompt shield for documents.

Does this meet the goal?

Options:

A- Yes

B- No

Answer:

B

Explanation:

The solution does not fully meet the goal. A prompt shield for documents is the correct control for the embedded-instruction portion of the scenario. Microsoft defines Prompt Shields as protection against prompt manipulation, including attacks embedded in third-party or document-like content that is supplied to a generative model. OCR-extracted text from uploaded images is untrusted contextual content, so document attack protection is appropriate for detecting hidden instructions that attempt to override the model's intended behavior.

However, the scenario contains two separate risks: unsafe image uploads and hidden instructions embedded in images. Prompt Shields for documents address indirect prompt injection, but they do not classify or block harmful visual content in the uploaded image itself. Azure AI Content Safety image moderation is the control that scans images for harmful content categories such as sexual content, violence, hate, and self-harm with severity levels that can be used for blocking decisions.

Therefore, document prompt shielding alone is incomplete. A complete mitigation would combine image moderation for unsafe images with Prompt Shields for document attacks, and optionally Spotlighting for lower-trust third-party content. Reference topics: Content Safety image moderation, Prompt Shields, document attacks, indirect prompt injection, and multimodal safety.

Question 7

Question Type: MultipleChoice

Case Study: Mix Questions

Mix Questions

AI-103 Mix Questions IN THIS CASE STUDY

Note: This section contains one or more sets of questions with the same scenario and problem. Each question presents a unique solution to the problem. You must determine whether the solution meets the stated goals. More than one solution in the set might solve the problem. It is also possible that none of the solutions in the set solve the problem.

After you answer a question in this section, you will NOT be able to return. As a result, these questions

do not appear on the Review Screen.

You have a multimodal AI generative model that accepts image uploads and uses extracted image text to generate responses.

You discover that users can upload unsafe images and embed hidden instructions into images to manipulate the model.

You need to implement controls to mitigate the risk.

Solution: You configure a prompt shield for user prompts.

Does this meet the goal?

Options:

A- Yes
B- No

Answer:

B

Explanation:

The solution does not meet the goal. Prompt Shields for user prompts are designed to detect direct attempts by a user to manipulate the model through the prompt itself. In this scenario, the malicious instructions are embedded inside uploaded images and then introduced into the model context through extracted image text. That pattern is an indirect prompt injection or document attack, not merely a direct user-prompt attack. Microsoft's Prompt Shields guidance distinguishes between user prompt attacks and document attacks, and states that document attacks involve harmful instructions embedded in supplied documents or third-party content.

The solution is also incomplete because users can upload unsafe images. Azure AI Content Safety includes image APIs that detect harmful content in images and support moderation across modalities. A complete mitigation would combine image moderation for unsafe visual content with Prompt Shields for document attacks, and optionally Spotlighting, so OCR-derived or embedded image text is treated as lower-trust context. Prompt Shields for user prompts alone would not reliably block unsafe images or hidden instructions extracted from those images. Reference topics: Azure AI Content Safety, Prompt Shields, user prompt attacks, document attacks, image moderation, and multimodal safety.

Question 8

Question Type: MultipleChoice

Mix Questions

AI-103 Mix Questions IN THIS CASE STUDY

Note: This section contains one or more sets of questions with the same scenario and problem. Each question presents a unique solution to the problem. You must determine whether the solution meets the stated goals. More than one solution in the set might solve the problem. It is also possible that none of the solutions in the set solve the problem.

After you answer a question in this section, you will NOT be able to return. As a result, these questions do not appear on the Review Screen.

You have a multimodal AI generative model that accepts image uploads and uses extracted image text to generate responses.

You discover that users can upload unsafe images and embed hidden instructions into images to manipulate the model.

You need to implement controls to mitigate the risk.

Solution: You configure image moderation to block unsafe content before processing the images.

Does this meet the goal?

Options:

A- Yes
B- No

Answer:

B

Explanation:

The solution does not fully meet the goal. Image moderation is appropriate for one part of the risk: blocking unsafe image content before the image is processed. Azure AI Content Safety provides image APIs that detect harmful content, and its harm categories and severity levels can be used to classify and block objectionable image content. This addresses unsafe photos, but it does not address hidden instructions embedded in images.

The second risk is prompt manipulation through extracted image text. After OCR extracts text from the uploaded image, that text becomes untrusted third-party content supplied to a generative model. Microsoft defines document attacks as malicious instructions embedded in third-party content, where the objective is to cause the model to execute unintended commands or alter intended behavior. Prompt Shields are the control designed to detect user prompt attacks and document attacks, including indirect attacks that come from uploaded or referenced content.

Therefore, image moderation alone is incomplete. A complete mitigation would combine image moderation for harmful visual content with Prompt Shields for document attacks, and optionally Spotlighting, so extracted or embedded text is treated as lower trust. Reference topics: Azure AI Content Safety, image moderation, Prompt Shields, document attacks, indirect prompt injection, and multimodal safety.

Question 9

Question Type: Hotspot

Case Study: Mix Questions

Mix Questions

AI-103 Mix Questions IN THIS CASE STUDY

Your company is piloting a customer support agent in a Microsoft Foundry project name Project1. Project1 is connected to an existing Application Insights resource, and the company's support team reviews runs in the Traces tab.

The Foundry Agent Service is configured to perform the following actions:

* Retrieve the Application Insights connection string by calling

project_client.telemetry.get_application_insights_connection_string().

* Call configure_azure_monitor(connection_string=...) to enable telemetry.

A separate LangChain service configured to use OpenTelemetry and has the following configurations:

* Uses AzureAIOpenTelemetryTracer(connection_string=..., enable_content_recording=False)

* Passes the tracer by using config={"callbacks":[azure_tracer]}

Company policy has the following requirements:

* Telemetry from LangChain and OpenTelemetry must be distinguishable within the same Application Insights resource.

* Secrets and credentials must NOT be stored in prompts, tool arguments, or span attributes.

For each of the following statements, select Yes if the statement is true. Otherwise, select No.

NOTE: Each correct selection is worth one point.

Answer Area

Statements	Yes	No
The LangChain service will appear in Traces without configuring a tracer.	○	○
Setting different `OTEL_SERVICE_NAME` values separates the services in Application Insights.	○	○
When using `enable_content_recording=False`, prompts and tool data will be captured in the telemetry.	○	○

Answer:

See the Answer in the Premium Version!

Thank You for trying AI-103 PDF Demo

To try our AI-103 practice exam software visit link below

https://prepbolt.com/AI-103.html

Start Your AI-103 Preparation

Use Coupon “SAVE50” for extra 50% discount on the purchase of Practice Test Software. Test your AI-103 preparation with actual exam questions.