Microsoft

(AI-300)

Operationalizing Machine Learning and Generative AI Solutions

Total: **142 Questions**

Link:

Model training jobs are run manually from notebooks.
Experiment tracking is inconsistent
Model versions are registered without standardized metadata.
Deployment is performed manually by data scientists, with limited rollback capability.
The team has no standardized evaluation process for generative AI outputs.
The environment currently allows public network access. Authentication relies on user accounts rather than managed identities. Compute targets are manually created and shared across experiments. This has led to resource contention during peak usage.
Business Requirements -
Fabrikam Inc. has the following business requirements for the modernization initiative:
Provide a conversational interface that answers analytics questions by using internal documents and datasets.
Ensure that sensitive healthcare-related data is not exposed outside the Fabrikam Inc. Azure tenant.
Enable repeatable and auditable model training and deployment processes.
Support experimentation to compare prompt strategies and fine-tuned models.
Align the model with the ranked preferences and optimize behavior for the long term.
Minimize disruption to existing analytics workloads during rollout.
Technical Requirements -
To support the business goals, Fabrikam Inc. identifies these technical requirements:
Use Azure Machine Learning workspaces to centrally manage data assets, models, and environments.
Implement experiment tracking and model versioning for all training jobs.
Orchestrate training and evaluation by using pipelines rather than manually running notebooks.
Deploy traditional machine learning models with support for staged rollout and rollback.
Improve RAG-based solution output quality.
Use the existing evaluation datasets that are based on real data with input-output pairs.
Apply advanced fine-tuning techniques only when prompt engineering is insufficient
Issues and Constraints -
Fabrikam Inc. must comply with internal security policies that require the company to restrict network access and avoid long-lived secrets. The data science team has limited Azure DevOps experience, so solutions must favor managed services and automation over custom infrastructure.
Cost predictability is important. Leadership prefers serverless or managed compute options where possible but is willing to approve dedicated compute for stable production workloads.
Problem Statement -
Fabrikam Inc. must design and implement an Azure-based AI operations solution that enables reliable training, evaluation, deployment, and iteration of generative AI models. The solution must support experimentation and gradual rollout while ensuring governance, security, and operational stability. The data science and platform teams must collaborate to deliver this solution by using Azure Machine Learning and Microsoft Foundry capabilities.
You need to isolate training workloads while remaining cost-aware to address Fabrikam Inc.'s issues, constraints, and technical requirements.
What should you implement?

A. Training jobs that run on a single shared compute cluster
B. Fixed-size compute cluster
C. Dedicated compute clusters per experiment
D. Managed compute targets with autoscaling

Answer: D

Explanation:

Technical Justification for Correct Answer: D

Why D is the Best Option:

Isolation of Training Workloads: Managed compute targets with autoscaling effectively isolate training workloads, ensuring that experiments do not interfere with each other, addressing the issue of resource contention.
Cost-Awareness:

Autoscaling adapts compute resources to the current workload, minimizing costs by not over-provisioning.

Specifically engineered for immediate, operational web requests. It hosts a deployed machine learning model on web services (like Azure Kubernetes Service or Azure Container Apps) that listens for incoming HTTP requests, processes input data instantly, and provides a low-latency synchronous response back to the client application.

Incorrect Answer

Batch endpoint: Designed for running long-running asynchronous scoring jobs over large-scale datasets rather than immediate, real-time single-record streaming requests.

Job-based training pipeline: Used exclusively to automate the workflow steps of data preparation and model retraining; it does not serve active inference or hosting endpoints.

Registered model artifact: A saved version-controlled binary or asset file in the workspace; it represents a passive file store rather than active compute or a deployment hosting option.

High-volume scheduled scoring

Correct Selection: **Batch endpoint.**

Correct Answer

Batch endpoint: Optimized specifically to handle massive amounts of stored data or high-volume datasets. It functions asynchronously and integrates seamlessly with cron schedules or compute orchestration pipelines to process bulk scoring workloads at set intervals.

Incorrect Answer

Online endpoint with autoscaling: Aimed at variable streaming user traffic. While it scales compute dynamically, it is cost-inefficient and architecturally suboptimal for structured, massive offline batch workloads scheduled at specific time windows.

Managed compute cluster: A backend infrastructure component (a collection of VMs) used to run pipelines or workloads, but not a direct model deployment or consumption endpoint option itself.

Model registry version: Simply points to a specific iteration of a saved model within the environment asset tracking system; it possesses no compute engine or ingestion capability to execute scoring.

Question: 4

You manage an Azure Machine learning workspace. You develop a machine learning model.
You must deploy the model to use a low-priority VM with a pricing discount.
You need to deploy the model.
Which compute target should you use?

A. Azure Container Instances (ACI)
B. Azure Machine Learning compute clusters
C. Local deployment
D. Azure Kubernetes Service (AKS)

You need to prepare a file that contains training data.
Which file format should you use?

A. CSV
B. TSV
C. JSONL
D. JSON

Answer: C

Explanation:

Technical Justification for Correct Answer: C. JSONL

The correct file format for preparing training data to fine-tune an Azure OpenAI Service base model is **JSONL (C)**. Here's why:

JSONL (JSON Lines) is the preferred format because each line in the file is a separate JSON object. This structure is ideal for streaming data into the model for fine-tuning, as it allows for efficient, line-by-line processing. Each data point (e.g., a text sample) can be independently formatted as a JSON object, making the data easily parseable and flexible for the various input requirements of different models or tasks (e.g., text classification, question-answering).

Why Other Options are Less Suitable:

A. CSV (Comma Separated Values): While CSV is great for tabular data, its fixed-column structure makes it less flexible for the varied and potentially complex data formats required for fine-tuning AI models, especially when dealing with free text or variable-length inputs.

B. TSV (Tab Separated Values): Similar to CSV, TSV suffers from the same limitations of a fixed-column approach, lacking the flexibility needed for encapsulating diverse data points in a single, streamlined format suitable for AI model fine-tuning.

D. JSON: Although JSON is highly flexible and can encapsulate complex data structures, a single JSON file (as implied by the option without specifying an array or lines format) would typically contain all data within a single JSON structure. This makes the file less efficient for streaming and processing line by line compared to JSONL, especially with large datasets.

Key Technical Points for Certification Review:

Efficient Processing: JSONL supports line-by-line processing, crucial for large datasets.
Flexibility: Each line can have a different structure, accommodating varied input requirements.
Streaming Compatibility: Essential for models expecting data in a streaming-compatible format.

References:

Azure OpenAI Service Documentation - Fine-tuning Models
JSON Lines Specification

Question: 7

You have a deployment of an Azure OpenAI Service base model.
You plan to fine-tune the model.
You need to prepare a file that contains training data for multi-turn chat.
Which file encoding method should you use?

C. Apply a Blocklist:

Similar to Content Filters: Applying a blocklist is more about filtering out unwanted data types or sources rather than generating the necessary input data for evaluation.
Premature Without Data Insight: Blocklists are more effective once you have an understanding of the data landscape, which synthetic data generation can initially provide.

D. Enable Observability Metrics:

Analysis Tool, Not Data Provider: Enabling observability metrics is crucial for monitoring and analyzing the performance of LLM prompt variants but does not address the immediate need for a controlled input dataset.
Dependent on Pre-existing Data Flow: Observability metrics require data to be in flow to provide meaningful insights, making this step ineffective as a first action without generated or existing data.

Conclusion: Generating synthetic interaction data is the foundational step for creating a controlled evaluation environment. It directly addresses the need for consistent, non-live user traffic inputs, setting the stage for subsequent refinements and analyses.

References

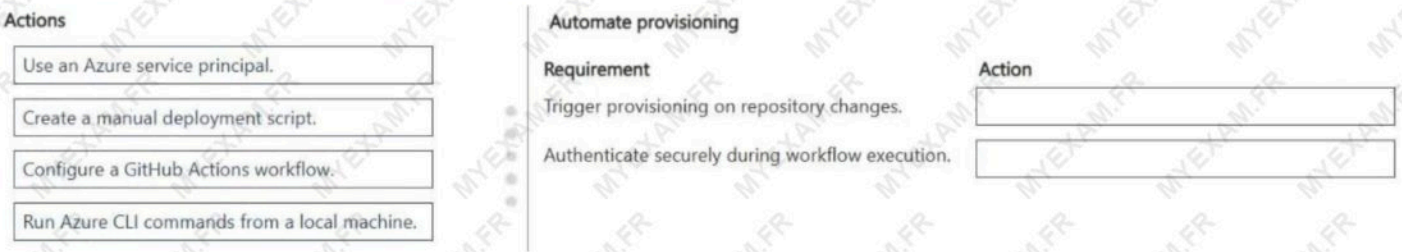
1. **Microsoft Documentation - Synthetic Data Generation for AI Models**: https://docs.microsoft.com/en-us/azure/machine-learning/how-to-data-synthesis
2. **Microsoft Foundry Documentation - Controlled Environment Testing**: Note: Since Microsoft Foundry is not a publicly recognized Microsoft product/service as of my last update, I'll provide a relevant alternative.

   Alternative: https://docs.microsoft.com/en-us/azure/cognitive-services/form-recognizer/generate-training-data

Please verify the existence and relevance of "Microsoft Foundry" as it seems to be a non-standard or potentially internal Microsoft tool/service not publicly documented at the time of this response. If "Microsoft Foundry" refers to a specific internal or newly released platform, direct documentation might not be publicly available.

Question: 10

DRAG DROP -
A team maintains Infrastructure as Code (IaC) templates to provision Azure Machine Learning resources. Provisioning must be triggered by changes in the templates and executed without manual intervention.
You need to automate resource provisioning.
Which action should you take for each requirement? To answer, move the appropriate actions to the correct requirements. You may use each action once, more than once, or not at all. You may need to move the split bar between panes or scroll to view content.
NOTE: Each correct selection is worth one point.



Answer:

Answer: B

Explanation:

Technical Justification for Correct Answer: B. Component

The correct asset to register for the given requirements is **B. Component**. Here's why:

Requirement for Consistent Asset Management Across Experimentation: Registering components in Azure Machine Learning (AML) allows for the management of reusable, modular pieces of code. This aligns perfectly with the need for a consistent way to manage assets created during experimentation.
Reuse and Governance Across Projects: Components, once registered, can be easily reused across different projects and workflows, facilitating collaboration and efficiency. Registration also enables governance by providing a centralized, version-controlled repository of components.
Specificity to the Need:

A. Model: While models can be registered in AML for versioning and deployment purposes, the question emphasizes the management of assets during experimentation for reuse across projects, which is broader than just the model.
B. Component: Best fits the requirement as components encapsulate not just the model but can also include data preprocessing steps, feature engineering, or any reusable piece of the workflow, making them ideal for managing the variety of assets generated during experimentation.
C. Environment: Registering an environment is more about managing dependencies and ensuring reproducibility of the runtime, not directly about managing reusable assets across projects in the context provided.
D. Pipeline: Pipelines are about orchestrating workflows; while they can be versioned and reused, the question points towards managing individual assets within those workflows, making Component more appropriate.

Why Other Options are Less Suitable:

Model (A): Too narrow, focused on the output rather than the broader assets created during experimentation.
Environment (C): Focuses on dependency management rather than asset reuse and governance.
Pipeline (D): Manages workflow orchestration, not the individual reusable components within those workflows.

Correct Answer Justification Summary:Registering **Components** in Azure Machine Learning is the most suitable approach for managing assets created during experimentation due to its focus on reusability, governance, and the modular management of workflow components, aligning closely with the team's requirements.

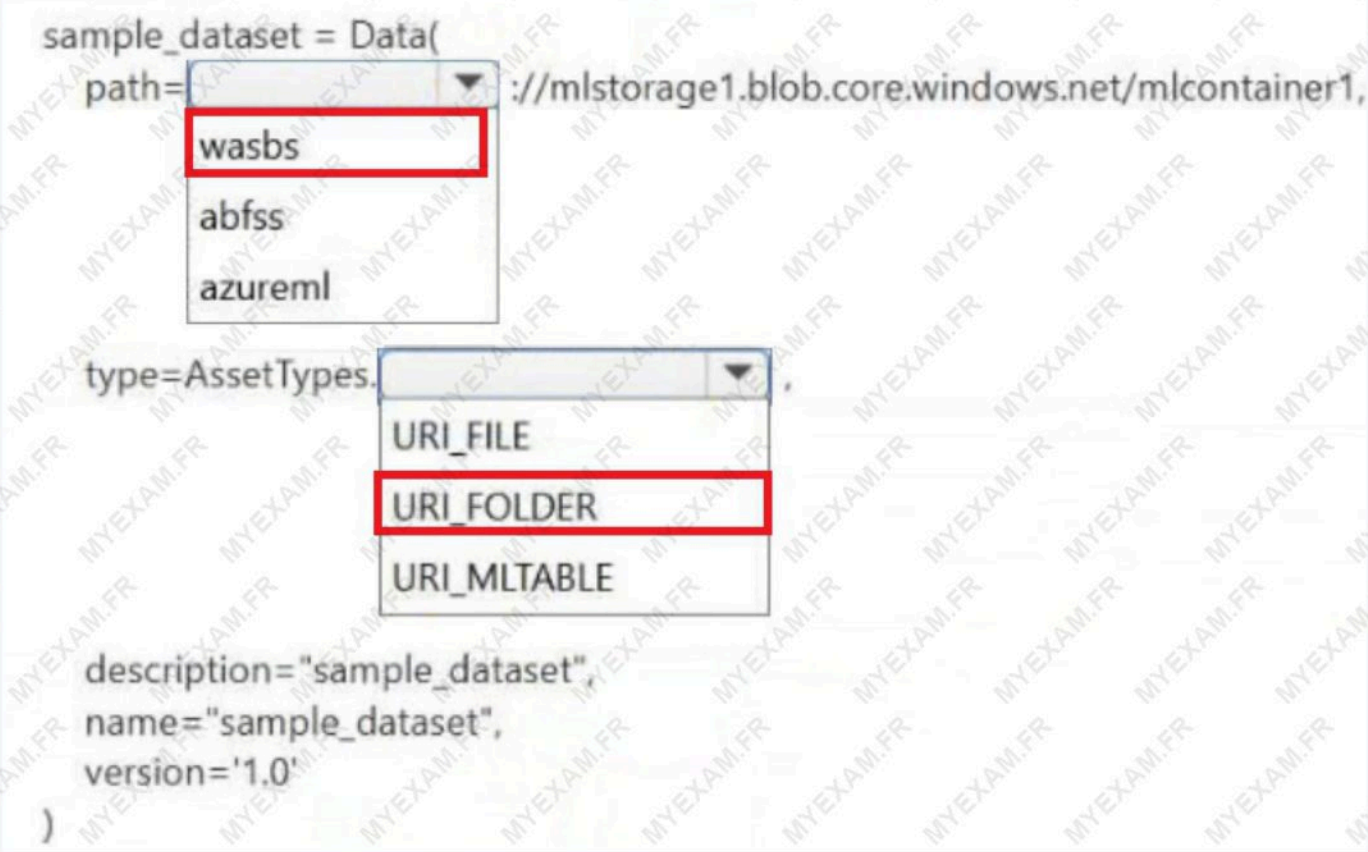
References

1. **Azure Machine Learning - Registering Components**
   https://docs.microsoft.com/en-us/azure/machine-learning/how-to-manage-components
2. **Azure Machine Learning - Asset Management Overview**
   https://docs.microsoft.com/en-us/azure/machine-learning/overview-what-is-azure-ml#manage-assets

Question: 14

HOTSPOT -
You manage an Azure Machine Learning workspace named workspace1 by using the Python SDK v2.
The default datastore of workspace1 contains a folder named sample_data. The folder structure contains the following content:

Answer Area



Explanation:

wasbs: This is the secure Windows Azure Storage Blob protocol scheme. Since the target host URI is a standard Azure Blob Storage domain (blob.core.windows.net), the wasbs protocol is the appropriate connector identifier.

URI_FOLDER: The provided path references a complete storage container tier (/mlcontainer1) rather than pointing to a standalone file. Because it encompasses multiple object files inside a collection path, it is registered under the URI_FOLDER type asset declaration.

Incorrect Answer

abfss: Used exclusively as the secure scheme identifier for Azure Data Lake Storage Gen2 (dfs.core.windows.net) hierarchical filesystems rather than standard Azure Blob Object endpoints.

azureml: Used internally within the workspace to reference an existing data asset already registered in Azure ML Studio (e.g., azureml:asset_name:version) rather than an external cloud directory URI.

URI_FILE / URI_MLTABLE: URI_FILE requires targeting a singular file directly with its extension (such as .csv), while URI_MLTABLE requires an accompanying MLTable schema configuration file in the target directory to define processing logic.

Question: 16

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear on the review screen.
You manage an Azure Machine Learning workspace. The Python script named script.py reads an argument named training_data. The training_data argument specifies the path to the training data in a file named dataset1.csv.
You plan to run the script.py Python script as a command job that trains a machine learning model.

SDK or CLI, specifying the parameter and its value for the job configuration.

References

Azure Machine Learning: How to pass parameters to a script
Azure Machine Learning Jobs: Configure and run jobs

Question: 19

HOTSPOT
-
You review the following Azure CLI command and the relevant Bicep excerpt.

```
01 az deployment group create \
02   --resource-group rg-foundry-dev \
03   --template-file ./foundry-environment.bicep \
04   --parameters location=eastus projectName=proj-dev-01
05
06 // foundry-environment.bicep (excerpt)
07 targetScope = 'resourceGroup'
08
09 param location string
10 param projectName string
11
12 resource aiFoundry 'Microsoft.CognitiveServices/accounts@2025-04-01-preview' = {
13   name: aiFoundryName
14   location: location
15   identity: {
16     type: 'SystemAssigned'
17   }
18   sku: {
19     name: 'S0'
20   }
21   kind: 'AIServices'

22   properties: {
24     allowProjectManagement: true
27     customSubDomainName: aiFoundryName
28     disableLocalAuth: true
29   }
30}
```

(Non-relevant sections are omitted.)
You need to validate what the snippet will do before it is merged. For each of the following statements, select Yes if the statement is true. Otherwise, select No.
NOTE: Each correct selection is worth one point.

Navigating to your personal GitHub account SSH settings and pasting this public string authorizes your Azure ML workspace compute to interact with your repositories.

From the terminal window in the Azure Machine Learning interface, run the git clone command.

With authenticating handshake materials successfully configured on both ends, you can now run the terminal cloning command securely without username or password prompts.

Incorrect Options

Add a private key to the GitHub account: You must never upload your private key (id_rsa) to any external platform, repository, or shared portal, as it compromises your machine's local identity and security credentials.

Question: 22

You manage an Azure Machine Learning workspace. You have an environment for training jobs which uses an existing Docker image.
A new version of the Docker image is available.
You need to use the latest version of the Docker image for the environment configuration by using the Azure Machine Learning SDK v2.
What should you do?

A. Change the description parameter of the environment configuration.
B. Modify the conda_file to specify the new version of the Docker image.
C. Use the create_or_update method to change the tag of the image.
D. Use the Environment class to create a new version of the environment.

Answer: C

Explanation:

Technical Justification for Correct Answer (C)

Why C is the best option: To update the Docker image version in an Azure Machine Learning workspace using the Azure Machine Learning SDK v2, the most direct and appropriate method is to **use the create_or_update method to change the tag of the image**. This approach:

Targets the Specific Change: Directly updates the Docker image tag, which is the precise change required.
Utilizes Idempotency: The create_or_update method is idempotent, meaning it safely updates the existing environment if the image tag is the only change, or creates it if it didn't exist, ensuring consistency without unnecessary recreations.
Minimizes Disruption: This method minimizes potential disruptions by only updating the specified parameter (the Docker image tag) within the existing environment configuration.

Why Other Options are Less Suitable:

A. Change the description parameter of the environment configuration:

Irrelevant to the Task: Updating the description does not change the Docker image version used by the environment.
No Impact on Functionality: This change would not achieve the goal of using the latest Docker image version.

GitHub Actions is the most suitable deployment automation tool for this scenario due to the following reasons:

Automated Deployments: GitHub Actions can be triggered by pull requests, automating the deployment process as required. Workflows can be defined to execute Azure Machine Learning deployment scripts.
Auditable: Each workflow run in GitHub Actions provides a detailed, logged history, making deployments auditable. This includes who triggered the run, the execution environment, and the outcome.
Approval Requirement: GitHub Actions supports the use of **Approval Actions** within workflows. This allows for a manual approval step to be inserted before the actual deployment to Azure Machine Learning, ensuring that deployments require approval before execution.

Why Other Options are Less Suitable:

A. Azure Monitor: Primarily a monitoring and analytics service, not designed for automating deployments based on pull requests or incorporating approval workflows directly. Its focus is more on alerting and troubleshooting rather than deployment automation.
C. MLflow: While MLflow is a comprehensive platform for managing the machine learning lifecycle and can integrate with deployment pipelines, it does not natively support triggering deployments from pull requests on GitHub with built-in approval steps as seamlessly as GitHub Actions. MLflow's strength lies in model management and experiment tracking.
D. Azure Machine Learning Pipelines: Although designed for automating ML workflows, Azure Machine Learning Pipelines are more focused on the workflow of machine learning tasks (e.g., data preparation, training) rather than the deployment process to production environments triggered by pull requests with approval gates. They are ideal for orchestrating ML tasks but less suited for the described deployment automation scenario.

References

For deeper insights into the selected tool and its integration capabilities:

1. **GitHub Actions Documentation - Automating Deployment**
   https://docs.github.com/en/actions/deployment/deploying-to-your-application
2. **Azure Machine Learning with GitHub Actions**
   https://learn.microsoft.com/en-us/azure/machine-learning/how-to-deploy-github-actions

Question: 26

A team develops and manages a conversational assistant by using Microsoft Foundry.
The team requires generative AI to automatically evaluate every pull request of an agentic application and fail the build if safety thresholds are exceeded.
You need to automate evaluations as part of CI.
What should you configure?

A. Blocklist applied to the model endpoint
B. Content filter configured in warning mode
C. Retrieval chunking strategy
D. GitHub Actions workflow that executes the runs

Answer: D

Explanation:

Cost predictability is important. Leadership prefers serverless or managed compute options where possible but is willing to approve dedicated compute for stable production workloads.
Problem Statement -
Fabrikam Inc. must design and implement an Azure-based AI operations solution that enables reliable training, evaluation, deployment, and iteration of generative AI models. The solution must support experimentation and gradual rollout while ensuring governance, security, and operational stability. The data science and platform teams must collaborate to deliver this solution by using Azure Machine Learning and Microsoft Foundry capabilities.
You need to recommend an experiment-tracking strategy that ensures consistent experiment results.
What should you recommend?

A. Azure Machine Learning job output logs
B. MLflow experiment tracking
C. Application Insights logs
D. Azure Monitor alerts

Answer: B

Explanation:

Technical Justification for Recommended Experiment-Tracking Strategy

Recommended Option: B. MLflow experiment tracking

Why B is the best choice:

Integration with Azure Machine Learning (AML): MLflow is natively integrated with Azure Machine Learning, making it seamless to track experiments across the platform Fabrikam Inc. is adopting for their modernization initiative. This integration ensures consistent tracking of model training jobs, which is crucial given the manual and inconsistent tracking they currently face.
Comprehensive Experiment Tracking: MLflow provides detailed tracking of experiments, including hyperparameters, models, and metrics, facilitating repeatable and auditable model training processes as required by Fabrikam's business requirements.
Support for Multiple Data Science Tools: Since Fabrikam's data science team uses Python-based scripts and Jupyter Notebooks, MLflow's compatibility with these tools ensures a unified tracking experience across their workflow.
Alignment with Governance and Security: By centrally managing experiment tracking through MLflow within AML, Fabrikam can better enforce security policies and reduce operational risks, especially considering their need to restrict network access and avoid long-lived secrets.

Why Other Options are Less Suitable:

A. Azure Machine Learning job output logs

Limited Scope: While useful for job-specific logging, these logs do not provide the comprehensive experiment tracking capabilities (e.g., hyperparameter tuning, model versioning across experiments) that MLflow offers.
Lack of Integrated Experiment Comparison: Not designed for comparing experiments at scale, which is crucial for Fabrikam's experimentation needs.

C. Application Insights logs

Primarily for Application Telemetry: Best suited for monitoring application performance and usage, not tailored for deep machine learning experiment tracking.
Insufficient for Model Development Lifecycle: Lacks the specific features needed for tracking model training, hyperparameters, and versioning in the context of ML development.

B. fairlearn.preprocessing.CorrelationRemover
C. fairlearn.reductions.GridSearch
D. fairlearn.postprocessing.ThresholdOptimizer

Answer: D

Explanation:

Technical Justification for Correct Answer D

Scenario Requirements Recap:

Binary Classification Model already exists (no retraining needed)
Fairlearn package for model fairness assessment and implementation
Eliminate the need for retraining the model

Why D (fairlearn.postprocessing.ThresholdOptimizer) is the Best Choice:

No Retraining Requirement: Since the model is already trained and retraining is not an option, **post-processing techniques** are the only viable approach among the given options. fairlearn.postprocessing.ThresholdOptimizer adjusts the decision threshold for different groups to achieve fairness, without altering the model itself.
Fairness Implementation: It directly addresses model fairness by optimizing thresholds for balanced accuracy across sensitive groups, aligning with the requirement to implement Fairlearn for fairness assessment and improvement.

Why Other Options are Less Suitable:

A. fairlearn.reductions.ExponentiatedGradient:

Requires Model Retraining: This reduction-based algorithm involves training a new model with fairness constraints, contradicting the "no retraining" requirement.
Focus: More on training fair models from scratch rather than post-hoc fairness adjustment.

B. fairlearn.preprocessing.CorrelationRemover:

Preprocessing Technique: Intended to remove correlations with sensitive attributes before training a model. Since the model is already trained, this approach is not applicable.
Requires Retraining: Effective use would necessitate retraining the model with the preprocessed data, which is not allowed.

C. fairlearn.reductions.GridSearch:

Model Selection and Retraining: Implies searching for the best model (possibly with different hyperparameters or fairness constraints) and retraining, which again, violates the no retraining condition.
Focus: On finding an optimal model through a grid search, not on post-hoc fairness adjustment.

References

Fairlearn Documentation - ThresholdOptimizer
Fairlearn Overview for Context

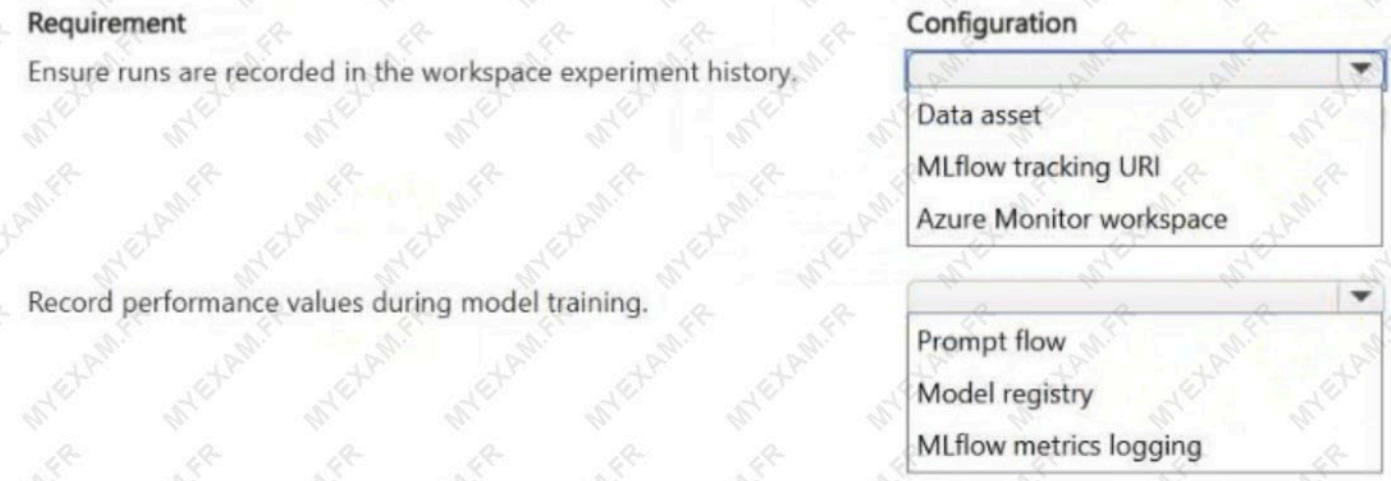
Question: 31

DRAG DROP -

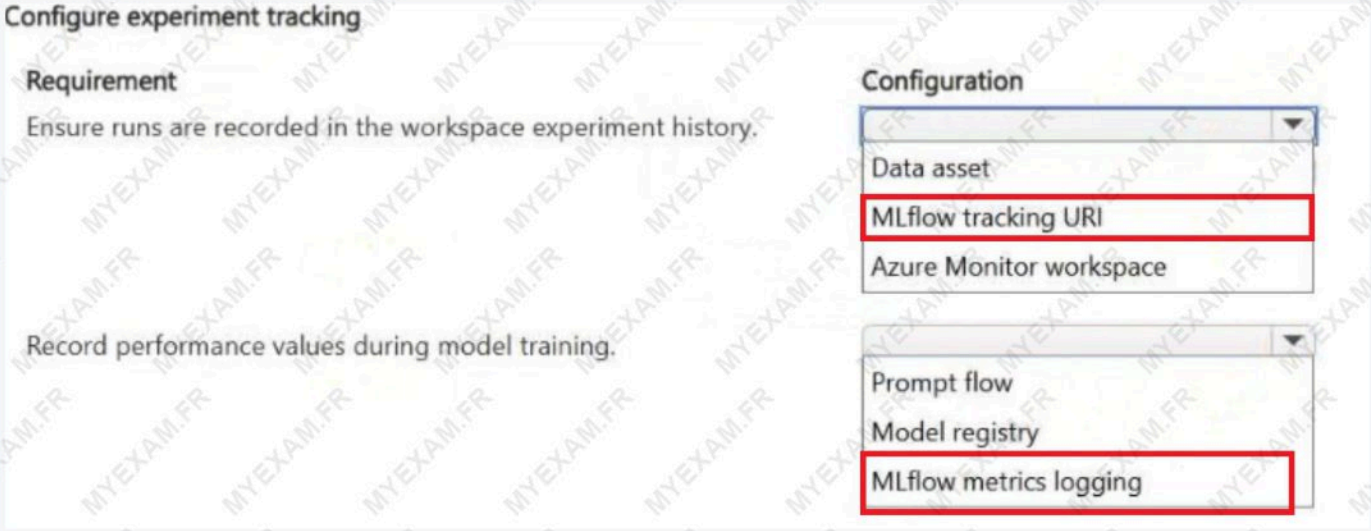
rather than a core metric performance monitoring solution configuration block. It is typically a downstream task launched by an alert rule or pipeline run, rather than a standalone option matching these specific architectural requirements.

Question: 34

HOTSPOT -
You use Azure Machine Learning to train models across multiple experiments by using the same workspace.
You must record training runs in a centralized location to compare results from different jobs.
During training, performance values must be captured so they appear in the experiment run history.
You need to configure experiment tracking.
What should you configure for each requirement? To answer, select the appropriate options in the answer area.
NOTE: Each correct selection is worth one point.



Answer:



Explanation:

Ensure runs are recorded in the workspace experiment history → MLflow tracking URI

Why: In Azure Machine Learning, MLflow is the standard tool for experiment tracking. To ensure your local or remote training jobs are routed and recorded directly inside the workspace's centralized experiment tracking interface, you must configure the environment's MLflow tracking URI to point to your specific Azure ML workspace endpoint.

Azure Machine Learning Documentation - Troubleshoot deployment issues

Question: 37

HOTSPOT -
You train a model in Azure Machine Learning.
You plan to capture experiment details for later comparison. The training code must log parameters and metrics for each run.
You review the following training script.

```
1  import mlflow
2  import mlflow.sklearn
3  from sklearn.linear_model import LogisticRegression
4  from sklearn.datasets import load_iris
5
6  X, y = load_iris(return_X_y=True)
7
8  mlflow.set_experiment("classification-experiment")
9
10 with mlflow.start_run():
11     model = LogisticRegression(max_iter=200)
12     model.fit(X, y)
13
14     mlflow.log_param("max_iter", 200)
15     y_pred = model.predict(X)
16     accuracy = accuracy_score(y, y_pred)
17     mlflow.log_metric("accuracy", accuracy)
18
19     mlflow.sklearn.log_model(model, "model")
```

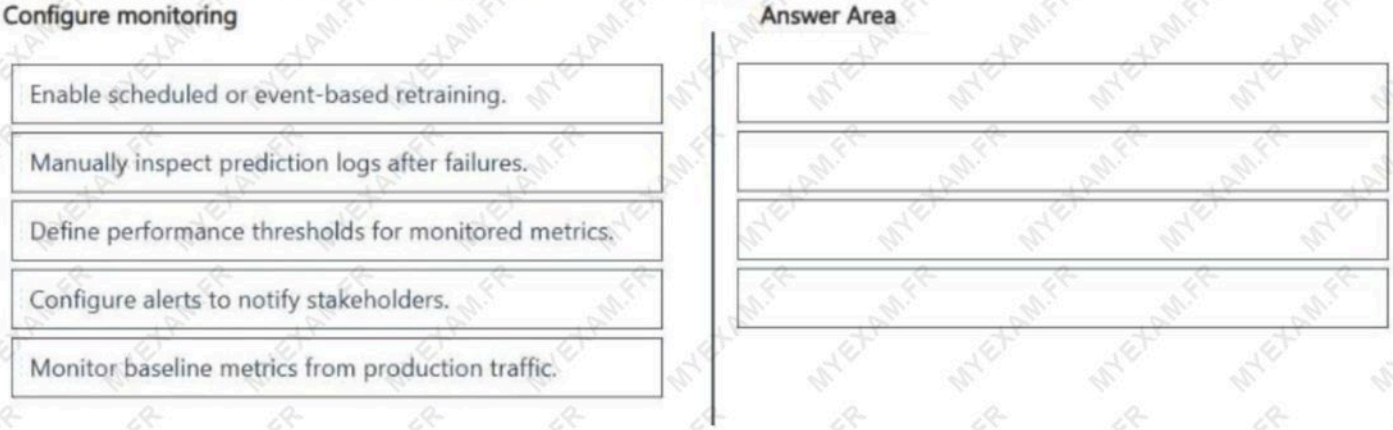
You need to verify whether the training script meets the experiment tracking requirement. For each of the following statements, select Yes if the statement is true. Otherwise, select No.
NOTE: Each correct selection is worth one point.

References:

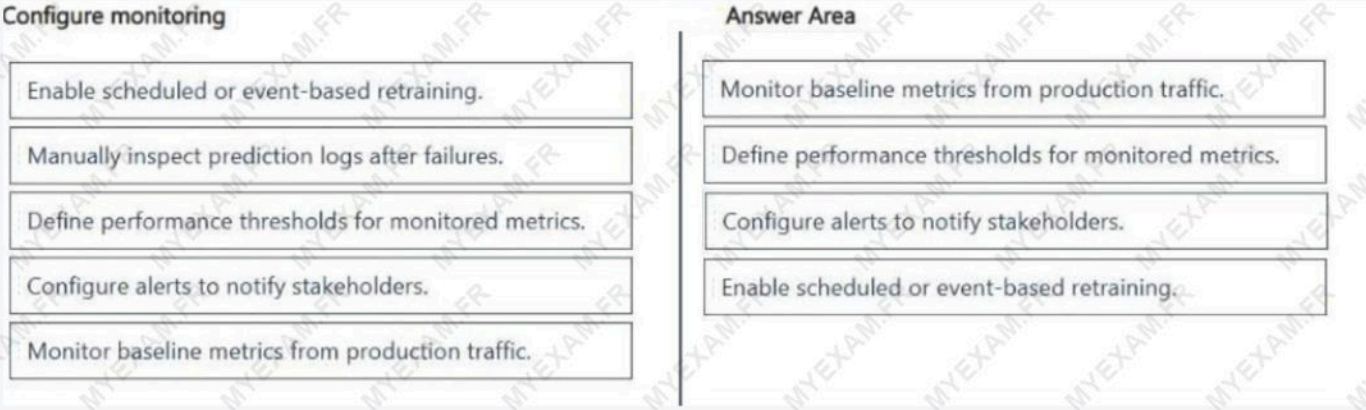
1. **Microsoft Documentation: Register a model in Azure Machine Learning**
   https://docs.microsoft.com/en-us/azure/machine-learning/how-to-manage-models?tabs=azure-portal#register-a-model
2. **MLflow Documentation: Managing Models with MLflow and Azure Machine Learning**
   https://docs.microsoft.com/en-us/azure/machine-learning/how-to-integrate-mlflow#store-and-manage-models (covers integration aspects relevant to capturing model artifacts)

Question: 39

DRAG DROP -
A team deploys a classification model to production and monitors performance and data changes.
The team wants to ensure that significant drops in prediction accuracy automatically trigger the following:
Stakeholders must be notified of the drops.
Retraining must be initiated when thresholds are exceeded
You need to configure monitoring to meet the requirements.
Which four actions should you perform in sequence? To answer, move the appropriate actions from the list of actions to the answer area and arrange them in the correct order.



Answer:



Explanation:

Monitor baseline metrics from production traffic.

Why first: Before you can catch an anomaly or performance drop, you must first establish a baseline context. Gathering streaming inference metrics from actual live production traffic provides the target dataset needed to understand normal operational distributions and model behavior.

Define performance thresholds for monitored metrics.

the existing (stable) model. This enables:

Risk Mitigation: Minimal impact on existing users, as most traffic remains on the proven model.
Controlled Testing: Insights into the new model's performance in a production-like setting with real user traffic.
Ease of Rollback: Simple to revert if issues are discovered, by adjusting the traffic split back to 100% for the original model.
Incremental Adoption: Allows for gradual increase in traffic to the new model based on performance metrics.

Why Other Options are Less Suitable:

B. The Model Asset Version in the Registry

Lack of Live Testing: Updating the model asset version in the registry does not facilitate live, controlled testing with production traffic.
All-or-Nothing Approach: Swapping versions in the registry would immediately redirect all traffic, risking disruption to users if the new model has unforeseen issues.
C. Deployment to a Separate Staging Endpoint

Isolated Testing: While a staging endpoint is useful for pre-production testing, it does not allow for testing with real production traffic, making it less effective for validating the model's behavior in the actual production environment.
Additional Complexity for User Routing: Manually routing a subset of users to a separate endpoint for testing adds complexity and may not accurately represent production conditions.
D. An Evaluation Script in Azure Machine Learning

Offline Evaluation: Evaluation scripts are beneficial for offline model comparison but do not support live traffic testing, which is crucial for validating the model's performance in real-world scenarios.
No Direct Production Traffic Insight: Does not provide feedback on how the model performs with actual production workload and user interactions.

References

Azure Machine Learning - Deploy models for online inference
Azure Machine Learning - Traffic Splitting for Online Endpoints

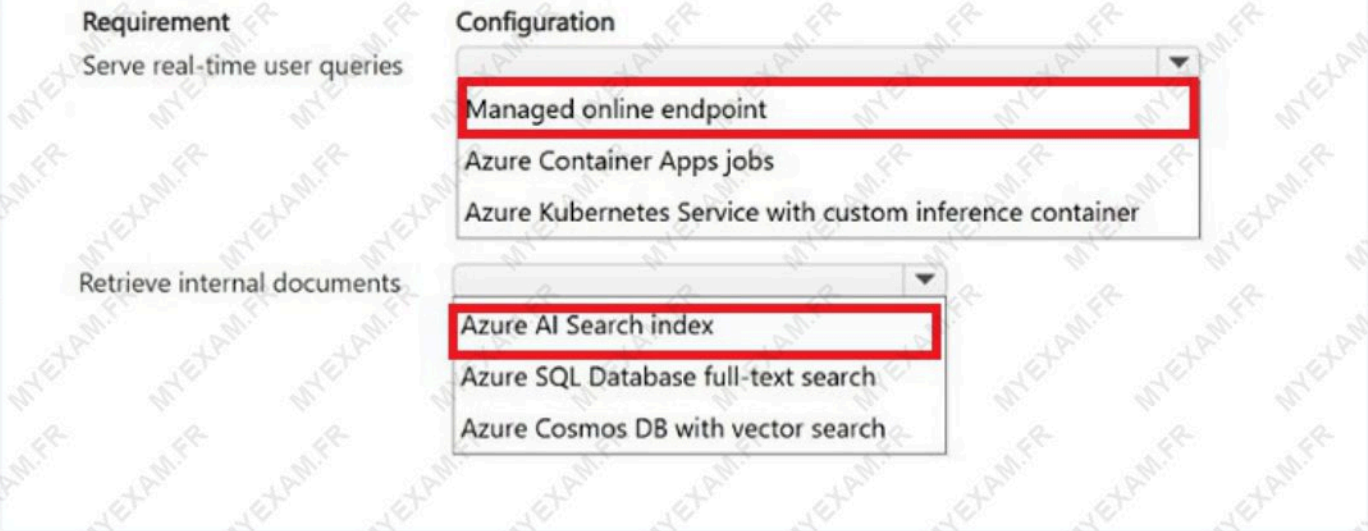
Question: 42

HOTSPOT -
You are monitoring a fine-tuned large language model deployed in Microsoft Foundry.
You evaluate the model before and after fine-tuning by using the same evaluation dataset.
You review the following evaluation results:

Evaluation metric	Baseline model	Fine-tuned model
Groundedness score	0.82	0.91
Relevance score	0.85	0.88
Hallucination rate	0.06	0.05
Response latency (ms)	420	430

You need to determine whether the fine-tuned model shows improved performance without introducing regression.
For each of the following statements, select Yes if the statement is true. Otherwise, select No.
NOTE: Each correct selection is worth one point.

Conversational model configuration



Explanation:

Serve real-time user queries → Managed online endpoint.

Why: In Azure Machine Learning and conversational AI architectures, a Managed online endpoint is built specifically for real-time, low-latency web requests. It takes incoming HTTP traffic, passes it synchronously to your conversational model, and returns immediate responses back to the chat client.

Retrieve internal documents → Azure AI Search index.

Why: For Retrieval-Augmented Generation (RAG) conversational architectures, Azure AI Search is Microsoft's premier enterprise search and indexing solution. It chunks, vectorizes, and indexes internal unstructured enterprise documents (like PDFs, Docs, and HTML), making it the native service to retrieve grounded, relevant context for your model.

Incorrect Answer :

Azure Container Apps jobs: Designed for running short-lived, transient background processing tasks on a schedule or event queue, rather than continuously listening to web traffic as a low-latency live chatbot endpoint.

Azure Kubernetes Service with custom inference container: While technically capable, this involves substantial administrative overhead to deploy, manage, and scale the underlying container infrastructure manually, whereas a managed online endpoint handles this natively.

Azure SQL Database full-text search / Azure Cosmos DB with vector search: While these databases can handle structured data search queries, they do not offer the turnkey document ingestion, text parsing, OCR capabilities, and hybrid keyword-vector retrieval combinations out-of-the-box that an enterprise Azure AI Search index provides for unstructured file stores.