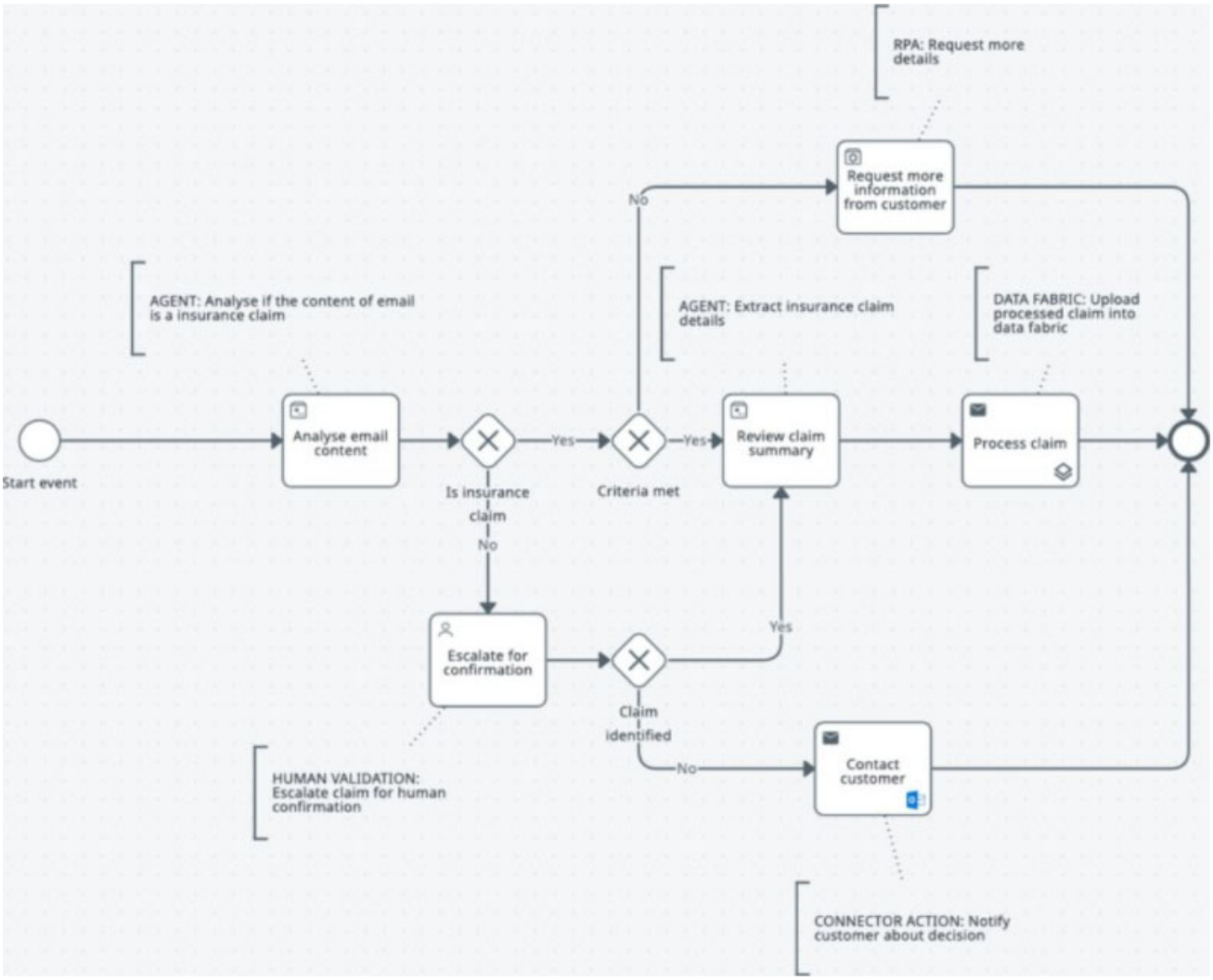
Exam : **UiAAPv1**

Title : UiPath Test Cloud Architect Professional

Version : DEMO

1.An insurance company needs to automate end-to-end policy validation by ingesting customer records from their CRM, invoking multiple REST APIs to verify coverage and eligibility against external data sources, and reconciling results for audit compliance. The process must accommodate authentication challenges, intermittent API outages, and changing response schemas, providing clear progress and error reporting for business users.
Which workflow design approach is best?
A. Create a modular UiPath workflow with centralized API authentication, schema validation/adaptation for varying responses, resiliency (auto-retries, graceful outage handling), and real-time progress/error logs for business users.
B. Organize the workflow so each API call operates independently, with hardcoded credentials; in case of outages, aggregate error codes for later manual investigation, and generate a single compliance summary after all processing is complete.
C. Use separate UiPath workflows for each stage, CRM ingestion, API verification, and reconciliation, allowing manual intervention if an authentication failure or schema mismatch occurs, and emailing stakeholders occasional audit summaries for review.
D. Configure the workflow to only process customer records during periods of known API uptime, skipping records and deferring validation when external data sources are unavailable, then batch process any skipped validations later and compile a multi-stage report.
Answer: A
Explanation:
A modular UiPath workflow provides a scalable, secure, and resilient approach to end-to-end policy validation. Centralized API authentication ensures consistent and secure credential management, while schema validation and adaptation accommodate changes in external API response structures.
Automated retries with appropriate error handling maintain workflow reliability during intermittent API outages.
Real-time progress tracking and detailed error logging provide business users with visibility into processing status, exceptions, and reconciliation results. This architecture also supports audit compliance through traceable execution records, consistent validation processes, and comprehensive reporting, while minimizing manual intervention and simplifying ongoing maintenance.

2.In the depicted insurance-claim automation, the agent analyzes incoming emails and escalates uncertain cases for confirmation before proceeding.



When configuring Human-in-the-Loop for this process, which configuration best ensures accurate claim identification while maintaining automation efficiency?

A. Configure human approval before notifying the customer of the final claim decision, regardless of how the claim was classified.

B. Configure human validation for a subset of processed claims selected based on predefined business priority categories.

C. Configure escalation to a human reviewer when the agent can't confidently confirm the email is an insurance claim.

D. Configure human review after the claim summary is generated to validate extracted details prior to uploading the processed claim into the data fabric.

Answer: C

Explanation:

Configuring Human-in-the-Loop escalation when the agent cannot confidently identify an email as an insurance claim ensures accurate claim classification while maintaining automation efficiency. Uncertain cases are routed to a human reviewer for confirmation before processing continues, reducing classification errors and preventing incorrect claim handling. This approach allows confidently identified claims to proceed automatically, minimizing unnecessary human intervention while ensuring reliability and accuracy in the claims automation process.

3.A logistics optimization agent is designed to operate within a fleet coordination system and must dynamically adjust delivery schedules based on real-time route conditions. The agent is expected to autonomously execute operational updates while remaining aligned with its defined objective and measurable outcomes.
Which implementation most accurately reflects proper tool integration within the agent story framework?
A. Provide fleet analytics dashboards as read-only resources that the agent monitors to surface scheduling insights for dispatcher review and manual approval.
B. Configure notification services as output channels that the agent uses to broadcast driver alerts after dispatchers have manually completed schedule revisions.
C. Expose route optimization and fleet scheduling automations as executable tools that the agent can invoke directly when adjusting delivery plans.
D. Connect historical delivery data repositories as static reference sources, the agent queries to generate route change recommendations, which dispatchers then validate and apply.
Answer: C
Explanation:
Exposing route optimization and fleet scheduling automations as executable tools enables the agent to autonomously perform operational actions within the agent story framework. These tools allow the agent to analyze real-time route conditions, invoke relevant automations, and dynamically update delivery schedules without requiring manual intervention.
This approach aligns tool integration with the agent's defined objectives and measurable outcomes by enabling intelligent decision-making, direct execution of scheduling adjustments, and continuous optimization of fleet operations.

4.How can activity input methods be configured in UiPath agent tools and what are the key considerations for cascading fields?
A. Activity input methods include Prompt (default), Argument, and Static Value. Cascading fields cannot use the Prompt input method as agents cannot infer changes in field structures at runtime.
B. Activity input methods are mostly limited to Prompt, though in some cases Argument or Static Value can be selected depending on the connector. Cascading fields generally work with the Prompt method because agents can usually handle dynamic field updates during execution without major issues.
C. Activity input methods are fully automated with no optional configurations available. Cascading fields do not impact agent tool functionality in any way.
D. Activity input methods primarily default to Static Value for most fields to ensure predictable behavior and minimize runtime errors. Cascading fields are automatically detected and adjusted by the agent at runtime for smooth handling of dependent or conditional inputs.
Answer: A
Explanation:
UiPath agent tools support three configurable activity input methods: Prompt (default), Argument, and Static Value. Prompt allows the agent to infer field values at runtime using the tool's description and available context. Argument populates fields using predefined agent input arguments, while Static Value assigns fixed values that remain unchanged across executions.
Cascading fields cannot use the Prompt input method because selecting a value in one field can change the structure of dependent fields. Since the tool schema cannot dynamically adapt at runtime, the agent

cannot reliably infer values for these fields. Proper configuration ensures predictable tool execution and reliable data handling.

5.Which statement most accurately distinguishes deterministic evaluators from LLM-based (model-graded) evaluators when designing agent evaluations?
A. Deterministic evaluators apply predefined rule-based matching, always produce consistent results for identical inputs, omit a Model selection field since they do not use LLMs, and are best for strict exact matches such as booleans, specific numeric values, exact strings, or arrays of primitives.
B. Deterministic evaluators produce probabilistic scores by sampling training examples and therefore can vary between identical runs to reflect uncertainty.
C. Deterministic evaluators use a lightweight LLM to score outputs, allow selecting model and temperature for repeatable comparisons, and are therefore preferable for flexible language evaluation.
D. Deterministic evaluators judge an agent's run behavior across intermediate steps and tool calls by applying semantic heuristics rather than exact matching of final outputs.
Answer: A
Explanation:
Deterministic evaluators use predefined, rule-based matching criteria to assess agent outputs, ensuring consistent and repeatable results for identical inputs. They do not rely on large language models (LLMs), so their configuration does not include a Model selection field.
These evaluators are particularly suitable for validating outputs that require strict exact matching, such as Boolean values, specific numeric values, exact strings, and arrays of primitive data types. Their predictable behavior makes them ideal for evaluating structured outputs where accuracy, consistency, and reproducibility are essential.
In contrast, LLM-based (model-graded) evaluators use language models to assess outputs based on semantic meaning, contextual relevance, and qualitative criteria, making them suitable for evaluating more complex or flexible responses.