PassQueen
a.k.a dumpsqueen.com

Developing in Agentic AI Systems

Microsoft GH-600

Version Demo

Total Demo Questions: 5

Total Premium Questions: 54

Buy Premium PDF

https://passqueen.com

support@passqueen.com

Topic Break Down

Topic	No. of Questions
Topic 1, Design agentic AI solutions	4
Topic 2, Develop agentic AI solutions	38
Topic 3, Implement responsible AI practices for agentic AI solutions	12
Total	54

QUESTION NO: 1

You have a GitHub Enterprise repository in which an agent creates pull requests targeting the `main` branch.

You must require approval from designated reviewers before pull requests that modify files in `.github/workflows/*` or `/infra/*` can be merged.

What should you configure?

A. a ruleset and an agents.md file

B. a ruleset and a .copilotignore file

C. a branch protection rule and a CODEOWNERS file

D. a branch protection rule and copilot-instructions.md

ANSWER: C

Explanation:

a branch protection rule and a CODEOWNERS file is correct because these GitHub features work together to enforce path-specific review requirements on pull requests targeting `main`. A CODEOWNERS file maps repository paths to the users or teams responsible for reviewing changes. For example, entries for `/.github/workflows/*` and `/infra/*` can identify the designated owners for workflow definitions and infrastructure files. The leading slash scopes each pattern to the repository root, ensuring that the ownership rule is applied to the intended directories.

The branch protection rule for `main` must enable required pull-request reviews and the setting to require review from Code Owners. When a pull request changes a path covered by CODEOWNERS, GitHub requests review from the listed owners and prevents merging until the required Code Owner approval is provided. This is especially appropriate for agent-created pull requests because the enforcement occurs server-side in GitHub and applies consistently regardless of whether the author is a person, automation, or an AI agent. See GitHub Docs: About code owners and GitHub Docs: About protected branches.

QUESTION NO: 2 - (DRAG DROP)

DRAG DROP -

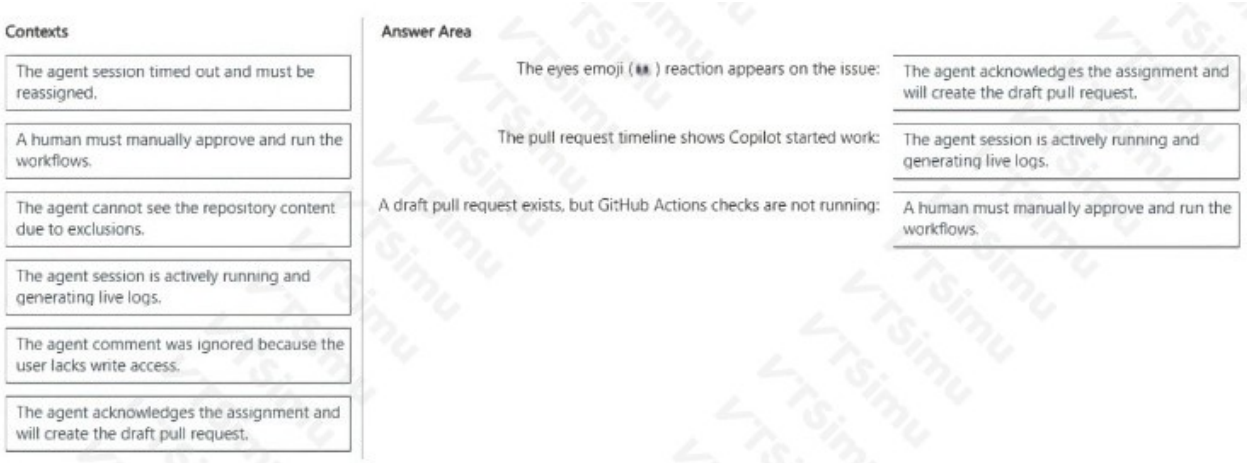
You have a GitHub Enterprise Cloud Organization that uses the GitHub Copilot coding agent to resolve issues asynchronously.

When an issue is assigned to GitHub Copilot, the agent creates a draft pull request, but your team cannot always tell whether the agent is actively working, has completed its session, or is awaiting workflow approval.

Which execution context does each signal indicate? To answer, drag the appropriate context to the correct signals. Each signal may be used once, more than once, or not at all. You may need to drag the split bar between panes or scroll to view content.

NOTE: Each correct selection is worth one point.

ANSWER:



Explanation:

GitHub Copilot coding agent exposes progress through lightweight issue and pull-request signals, so the three visible events correspond to successive parts of its asynchronous execution flow. The eyes reaction on an issue is the acknowledgement signal: Copilot has recognized the assignment and accepted responsibility for it. That acknowledgement is associated with the agent proceeding to create its draft pull request, which gives the team a tangible work item to follow while the agent continues in the background.

After the pull request has been created, the pull-request timeline can show that Copilot started work. This timeline event represents a live execution session rather than only an assignment acknowledgement. At that point the agent is actively working through the task and generating logs, allowing collaborators to observe that the session is underway. GitHub documents that coding-agent activity and progress are surfaced through the pull request and its associated session information. See Assigning GitHub Copilot to an issue.

A draft pull request with checks that have not started reflects a workflow gate, not missing coding work. GitHub Actions workflows can require approval before they run, particularly when repository policy or the event context requires an authorized person to approve execution. In this state, a human must manually approve and run the workflows before those checks can begin. This is consistent with GitHub's workflow-approval model, described in Approving workflow runs from public forks, and with the coding agent's use of pull requests and Actions-based validation. Thus the acknowledgement reaction identifies assignment acceptance, the timeline entry identifies active work, and inactive checks identify a required human workflow approval.

QUESTION NO: 3

Contoso is retrying the AI-assisted modernization of App1 from .NET 6 to .NET 10. The Copilot modernization workflow identified 47 issues, including one security vulnerability and 46 API incompatibilities across projects.

Which two activities are unsafe to delegate entirely to the agent and require human involvement? Each correct answer presents a complete solution.

A. Validate the assessment.md file for accuracy.

B. Generate the assessment.md file.

C. Approve all Git commits.

D. Review the plan.md file for dependencies.

E. Validate whether the tasks.md file exists.

ANSWER: A C

Explanation:

Validate the assessment.md file for accuracy. requires human involvement because the assessment is an AI-generated analysis that drives the modernization scope and subsequent remediation work. Although the modernization agent can inspect the application, identify breaking changes, and generate an assessment, a developer must verify that its findings accurately represent the application's architecture, dependencies, usage patterns, and business-critical behavior. This is especially important where an identified security vulnerability and numerous API incompatibilities can have different severity, impact, and remediation requirements.

Approve all Git commits. also requires human involvement. Agent-generated changes should enter the normal repository review and governance process rather than being accepted automatically. A human reviewer must assess correctness, security implications, test coverage, compatibility impact, and whether changes meet organizational standards before commits or pull requests are approved and merged. GitHub's coding-agent guidance emphasizes review of agent-created pull requests, while GitHub Copilot's responsible-use guidance notes that users remain responsible for reviewing and validating generated output. See GitHub Copilot coding agent documentation and responsible use of the coding agent.

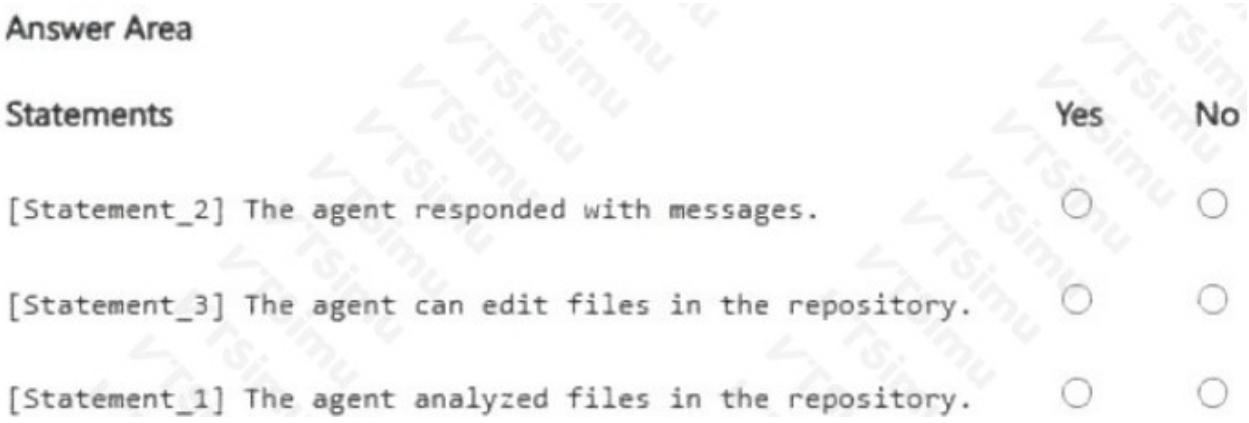
QUESTION NO: 4 - (HOTSPOT)

HOTSPOT -

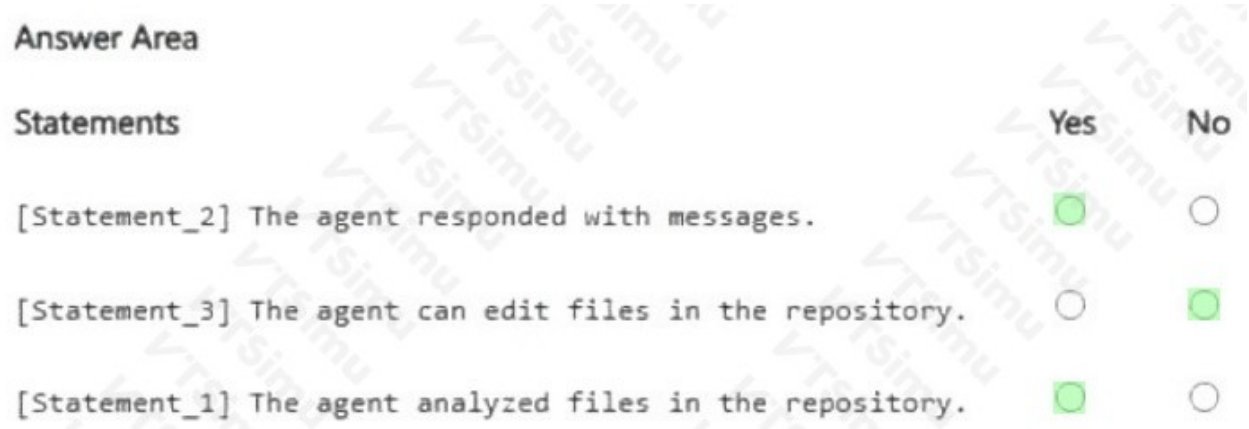
You have the following agent logs.

```
2026-03-19 20:50:35.796 [info] Latest entry: ccreq:latest.copilotmd
2026-03-19 20:50:35.796 [info] ccreq:852dff08.copilotmd | markdown
2026-03-19 20:50:47.919 [info] message 0 returned. finish reason: [stop]
2026-03-19 20:50:47.921 [info] request done: requestId: [9ce8eb0c-6df0-46c3-b5fe-0944cdab974d] model deployment ID: []
2026-03-19 20:50:47.922 [info] ccreq:9d5464cd.copilotmd | success | gpt-4o-mini -> gpt-4o-mini-2024-07-18 | 469ms | [copilotLanguageModelWrapper]
2026-03-19 20:52:22.276 [info] message 0 returned. finish reason: [stop]
2026-03-19 20:52:22.282 [info] request done: requestId: [35e76faf-8e9f-4ae4-af05-67d253879f6e] model deployment ID: []
2026-03-19 20:52:22.286 [info] ccreq:c4f325bf.copilotmd | success | gpt-4o-mini-2024-07-18 | 480ms | [title]
2026-03-19 20:52:22.573 [info] message 0 returned. finish reason: [stop]
2026-03-19 20:52:22.577 [info] request done: requestId: [a5db6877-13dc-488c-bfd5-5b1b483e86d2] model deployment ID: []
2026-03-19 20:52:22.581 [info] ccreq:c20009d5.copilotmd | success | gpt-4o-mini-2024-07-18 | 773ms | [progressMessages]
2026-03-19 20:52:22.653 [info] message 0 returned. finish reason: [stop]
2026-03-19 20:52:22.655 [info] request done: requestId: [915aaf76-fa4c-4a41-8ea2-cd2c1ca93ad3] model deployment ID: []
2026-03-19 20:52:22.655 [info] ccreq:1c805fcd.copilotmd | success | gpt-4o-mini-2024-07-18 | 846ms | [progressMessages]
2026-03-19 20:52:30.530 [info] ccreq:ffe1d495.copilotmd | success | gpt-5.3-codex | 8533ms | [panel/editAgent]
2026-03-19 20:52:35.563 [info] ccreq:2f500477.copilotmd | success | gpt-5.3-codex | 4892ms | [panel/editAgent]
2026-03-19 20:52:42.692 [info] ccreq:ecd182f4.copilotmd | success | gpt-5.3-codex | 6983ms | [panel/editAgent]
```

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:



Explanation:

The correct selections are: the agent responded with messages — Yes; the agent can edit files in the repository — No; and the agent analyzed files in the repository — Yes. The logs contain a successful request associated with `[progressMessages]`. Progress messages are agent-generated status or communication output, so this demonstrates that the agent returned messages during its work. The logs also show successful requests for `[panel/editAgent]`, which identifies activity in the edit-agent experience. Together with the sequence of requests involving the Copilot model wrapper, title generation, and progress messages, this is evidence of agent processing over the repository context and supports the conclusion that repository content was analyzed. However, the log lines record model-request activity and panel operations only. They do not show a file-write operation, a commit, a patch application, or another confirmed repository modification action. An edit-oriented agent interface is not itself proof that the agent has permission to write repository files. Therefore, repository edit capability cannot be inferred from these logs and must be marked No. This distinction is important when interpreting agent telemetry: request names and successful model invocations can establish that processing and messaging occurred, but authorization to modify source files requires explicit evidence of a write-capable operation or permission. Microsoft's guidance on GitHub Copilot coding-agent workflow and repository interaction is available at GitHub Copilot coding agent documentation, while agent observability concepts are covered in Azure AI Foundry agent tracing documentation.

QUESTION NO: 5 - (HOTSPOT)

HOTSPOT -

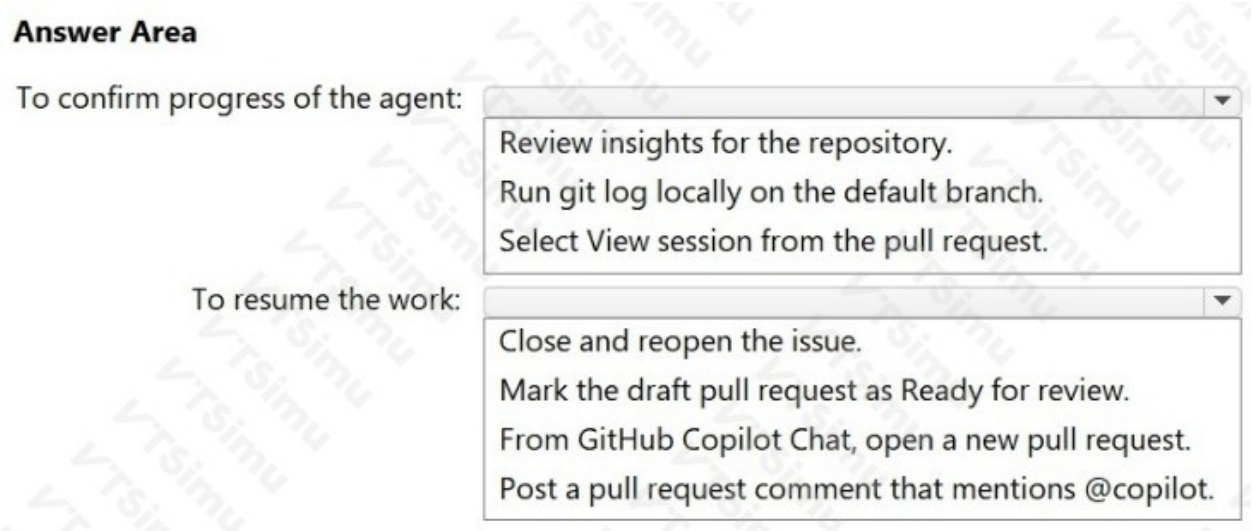
You have a GitHub repository that uses the GitHub Copilot coding agent to resolve issues and create draft pull requests.

You assign an issue to Copilot. Copilot creates a draft pull request. The pull request timeline shows Copilot started work, followed by status updates. The most recent status update is 55 minutes old.

You discover that the agent is no longer making progress.

You need to ensure that the work resumes without redoing the completed steps or changing the previously chosen approach.

What should you do? To answer, select the appropriate options in the answer area. NOTE: Each correct selection is worth one point.



ANSWER:

Explanation:

Use **Select View session from the pull request** to confirm the coding agent's progress. A Copilot coding-agent pull request is associated with an agent session, and the session view is the appropriate place to inspect what the agent has done, review its current activity, and understand whether it encountered a problem. This is especially useful when the pull request timeline shows that work began and then status updates stopped for an extended period. The session provides the context needed to assess the state of the active task without relying only on the timeline's summary messages. GitHub documents this workflow in its documentation for tracking Copilot coding-agent sessions.

To continue the same work, **post a pull request comment that mentions @copilot**. Mentioning Copilot on the existing draft pull request sends a follow-up instruction within the context of that already-created coding task. Because the comment is attached to the current pull request, Copilot can use the existing branch, prior changes, issue context, and earlier decisions. This lets the agent resume or continue the work without discarding completed implementation steps or replacing the approach it had already taken. The comment can ask Copilot to continue, investigate the lack of progress, or proceed from the current state. GitHub's Copilot coding-agent guidance describes interacting with the agent through pull request comments, including using an `@copilot` mention to provide additional direction. Together, viewing the session first and then mentioning `@copilot` on that pull request preserves continuity and gives the agent the information and instruction needed to resume.