Exam : **Sitecore XM Cloud Developer**

Title : Sitecore XM Cloud Developer Certification Exam

Version : DEMO

1.A developer is working with Sitecore's Authoring and Management API to manage their Sitecore content using GraphQL. They want to explore and interact with the API using the GraphQL integrated development environment (IDE).
Which of the following statements is correct about using the GraphQL IDE?
A. A developer needs to be in the sitecore\Admin role to access the GraphQL IDE.
B. A developer needs to be in the sitecore\Sitecore Client Users role to access the GraphQL IDE.
C. The GraphQL IDE provides read-only access to the API.
D. The GraphQL IDE is only available for non-production environments to ensure secure interactions.
Answer: B
Explanation:
Access to the GraphQL IDE for exploring and managing Sitecore content via the Authoring and Management API requires a developer to have the sitecore\Sitecore Client Users role. This role grants the necessary permissions to use the IDE for various operations, not just read-only access.
Reference: The Sitecore XM Cloud documentation specifies the role requirements for using the GraphQL IDE1. It also provides guidance on setting up and authoring content with the GraphQL playgrounds, which are part of the IDE2.

2.A developer is tasked with creating an item using the Sitecore Authoring and Management GraphQL API.
Which of the following GraphQL mutations is the correct way to create a new item?
A. createOrUpdateItem
B. create TemplateItem
C. createItem
D. updateItem
Answer: C
Explanation:
The correct GraphQL mutation to create a new item in Sitecore XM Cloud is createItem. This mutation allows developers to specify the necessary details such as the item’s name, template ID, parent ID, language, and fields to create a new content item within the Sitecore content tree.
Reference: The usage of the createItem mutation is documented in the Sitecore XM Cloud Developer’s Guide, which provides examples and explanations for authoring operations, including item creation1. Additionally, the Sitecore Stack Exchange provides insights into the available mutations for item management, confirming the use of createItem for creating new items2.

3.A developer wants to create a webhook that sends an HTTP request to a specified endpoint when the workflow moves to the approved state.
What type of webhook should they use?
A. Submit handler
B. Event handler
C. Submit action
D. Validation action
Answer: C
Explanation:
In Sitecore XM Cloud, a webhook submit action is used to send an HTTP request to a specified endpoint

when an item changes workflow state or a workflow command runs. Therefore, for a developer wanting to create a webhook that triggers when the workflow moves to the approved state, a webhook submit action would be the appropriate choice.
Reference: This information is confirmed by the Sitecore XM Cloud documentation, which details the different types of webhooks available and their specific uses, including the webhook submit action for workflow state changes1.

4.When an item is published, the Experience Edge for XM Connector publishes a static snapshot of the Layout Service output of that item.
If a change is made to a data source item that is referenced on the page, how is that content made visible on the website?
A. A developer must publish the data source item.
B. A developer must publish the related page items.
C. A developer must publish to the web database.
D. A developer must reconnect to the Experience Edge Connector module.
Answer: A
Explanation:
When a change is made to a data source item in Sitecore XM Cloud, the updated content becomes visible on the website after the data source item itself is published. This is because the Experience Edge for XM Connector publishes a static snapshot of the Layout Service output, and any changes to the data source items require republishing to reflect on the website.
Reference: The Sitecore XM Cloud documentation explains that the Experience Edge for XM Connector uses a static publishing model, meaning that dynamic content structures are flattened at publishing time. Therefore, if a data source item changes, it must be republished for the changes to take effect on the website12.

5.Which of these options best describes the purpose of the following query to the Experience Edge GraphQL schema?

```
query {
layout(site: "experienceedge", routePath: "/", language: "en") {
item {
homeItemPath: path
contentRoot: parent {
id
path
}
}
}
}
```

A. To get an item by ID
B. To get the root item of a site
C. To get the item layout for a URL
D. To get information about a specific content site
Answer: C

Explanation:
The query to the Experience Edge GraphQL schema is designed to retrieve the layout information for a specific URL, which in this case is the root path ("/") of a site named "experienceedge". This allows developers to access the Layout Service JSON for the item, which is essential for rendering the page in a headless setup.
Reference: The Sitecore XM Cloud documentation describes the Experience Edge schema as a read-only GraphQL schema that supports common front-end use cases for headless Sitecore development, including querying an item's layout by site and route path1.