

Microsoft

MS-600

Building Applications and Solutions with Microsoft 365 Core Services

- Up to Date products, reliable and verified.
- Questions and Answers in PDF Format.

Full Version Features:

- 90 Days Free Updates
- 30 Days Money Back Guarantee
- Instant Download Once Purchased
- 24 Hours Live Chat Support

For More Information:

<https://www.testsexpert.com/>

• Product Version

Visit us at <https://www.testsexpert.com/ms-600/>

Latest Version: 18.0

Question: 1

You need to configure the initial login request in the access token JavaScript script.
Which code segment should you insert at line 01?

- A. `const scopes = ['https://graph.microsoft.com/.default'];`
- B. `const accessTokenRequest = {`
`};`
- C. `const scopes = ['https://graph.microsoft.com/Files.Read.All',`
`'https://graph.microsoft.com/Mail.Send.All'];`
- D. `const accessTokenRequest = {`
`scopes: ['https://graph.microsoft.com/Files.ReadWrite',`
`'https://graph.microsoft.com/Mail.Send']`
`};`

Answer: D

Explanation:

Scenario: ADatum identifies the following technical requirements for the planned E-invoicing capabilities:

Ensure that all operations performed by E-invoicing against Office 365 are initiated by a user. Require that the user authorize E-invoicing to access the Office 365 data the first time the application attempts to access Office 365 data on the user's behalf.

Reference: <https://docs.microsoft.com/en-us/graph/permissions-reference>

Question: 2

DRAG DROP

You need to protect the backend web service to meet the technical requirements.

Which four actions should you perform in sequence? To answer, move the actions from the list of actions to the answer area and arrange them in the correct order.

Actions

Set the App ID URI for the backend web service application registration

Register an application in Azure AD for the backend web service

Define the scopes in the E-invoicing application registration

Register an application in Azure AD for E-invoicing. Grant admin consent to the E-invoicing application registration to the API scopes of the backend web service application registration

Define the scopes in the backend web service application registration

Answer Area



Answer:

Register an application in Azure AD for the backend web service

Set the App ID URI for the backend web service application registration

Define the scopes in the E-invoicing application registration

Register an application in Azure AD for E-invoicing. Grant admin consent to the E-invoicing application registration to the API scopes of the backend web service application registration

Explanation:

Here is a quick overview of the steps:

Step 1: Register an application in Azure AD for the backend web service

Register an application (backend-app) in Azure AD to represent the API.

Step 2: Set the App ID URI for the backend service application registration

When the application is created (step 1) select Expose an API and click on Save and continue to create an Application ID URI.

Step 3: Define the scopes in the backend web service application registration

In the Add a scope page, create a new scope supported by the API. (e.g., Read) then click on Add scope

to create the scope. Repeat this step to add all scopes supported by your API.

Step 4: Register an application in Azure AD for E-invoicing.

Step 4.1 Register another application in Azure AD to represent a client application

Step 4.2 Now that you have registered two applications to represent the API and the Developer Console, you need to grant permissions to allow the client-app to call the backend-app.

Scenario:

Secure access to the backend web service by using Azure AD

E-invoicing will have internal logic that will dynamically identify whether the user should be allowed to call the backend API.

Reference: <https://docs.microsoft.com/en-us/azure/api-management/api-management-howto-protectbackend-with-aad>

Question: 3

You need to complete the MSAL.js code for SSO.

Which code segment should you insert at line 06?

- A. `storeAuthStateInCookie: false`
- B. `storeAuthStateInCookie: true`
- C. `cacheLocation: 'localStorage'`
- D. `cacheLocation: 'sessionStorage'`

Answer: C

Explanation:

Scenario: Implement single sign-on (SSO) and minimize login prompts across browser tabs.

When your application is open in multiple tabs and you first sign in the user on one tab, the user is also signed in on the other tabs without being prompted. MSAL.js caches the ID token for the user in the browser `localStorage` and will sign the user in to the application on the other open tabs.

By default, MSAL.js uses `sessionStorage` which does not allow the session to be shared between tabs. To get SSO between tabs, make sure to set the `cacheLocation` in MSAL.js to `localStorage`.

Reference: <https://docs.microsoft.com/bs-latn-ba/Azure/active-directory/develop/msal-js-sso>

Question: 4

How can you validate that the JSON notification message is sent from the Microsoft Graph service?

- A. The `ClientState` must match the value provided when subscribing.
- B. The `user_guid` must map to a user ID in the Azure AD tenant of the customer.
- C. The tenant ID must match the tenant ID of the customer's Office 365 tenant.
- D. The subscription ID must match the Azure subscription used by ADatum.

Answer: A

Explanation:

clientState specifies the value of the clientState property sent by the service in each notification. The maximum length is 128 characters. The client can check that the notification came from the service by comparing the value of the clientState property sent with the subscription with the value of the clientState property received with each notification.

Note: A subscription allows a client app to receive notifications about changes to data in Microsoft Graph.

Reference: <https://docs.microsoft.com/en-us/graph/api/resources/subscription>

Question: 5

Which type of authentication flow should you recommend for the planned integration with Office 365?

- A. device code
- B. implicit grant
- C. authorization code
- D. client credentials

Answer: C

Explanation:

To use Microsoft Graph to read and write resources on behalf of a user, your app must get an access token from the Microsoft identity platform and attach the token to requests that it sends to Microsoft Graph.

One common flow used by native and mobile apps and also by some Web apps is the OAuth 2.0 authorization code grant flow.

Scenario: Email the generated invoices to customers on behalf of the current signed-in user. Any emails generated by the system will contain the invoiced.

Use Azure AD to manage identities, authentication, and authorization.

Reference: <https://docs.microsoft.com/en-us/graph/auth-v2-user>

For More Information – Visit link below:
<https://www.testsexpert.com/>

Features:

■ Money Back Guarantee.....



■ 100% Course Coverage.....



■ 90 Days Free Updates.....



■ Instant Email Delivery after Order.....

