

100% Money Back
Guarantee

Vendor:Microsoft

Exam Code:MB-820

Exam Name:Microsoft Dynamics 365 Business
Central Developer

Version:Demo

QUESTION 1

A company has a task that is performed infrequently. Users often need to look up the procedure to complete the task.

The company requires a wizard that leads users through a sequence of steps to complete the task.

You need to create the page to enable the wizard creation.

Which page type should you use?

- A. NavigatePage
- B. Card
- C. RoleCenter
- D. List

Correct Answer: A

For a task that is performed infrequently and requires users to follow a sequence of steps, a wizard-like interface is ideal. In Microsoft Dynamics 365 Business Central, the NavigatePage page type (A) is best suited for this purpose. NavigatePage is designed to guide users through a series of steps or pages, allowing them to complete a task by making choices or entering data in a structured manner. This page type is often used for setup wizards, data migration tasks, or any other process that benefits from a step-by-step approach. Unlike the other page types like Card (B), RoleCenter (C), or List (D), NavigatePage specifically supports the navigation and decision-making flow required for wizard creation, making it the optimal choice for this requirement.

QUESTION 2

HOTSPOT

You need to define the properties of the comments field of the Non-conformity page.

How should you complete the code segment? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Hot Area:

ExtendedDataType property

```
group(commentsGroup)
{
  field("comments"; NonConformityComments)
  {
    ApplicationArea = All;
    

|                    |
|--------------------|
| MultiLine = True;  |
| MultiLine = False; |
| NotBlank= True;    |
| NotBlank= False;   |



|                  |
|------------------|
| DataType         |
| ExtendDataType   |
| ExtendedDatatype |
| RichDataType     |

 = 

|                 |
|-----------------|
| LongContent     |
| None            |
| RichContent     |
| TextRichContent |

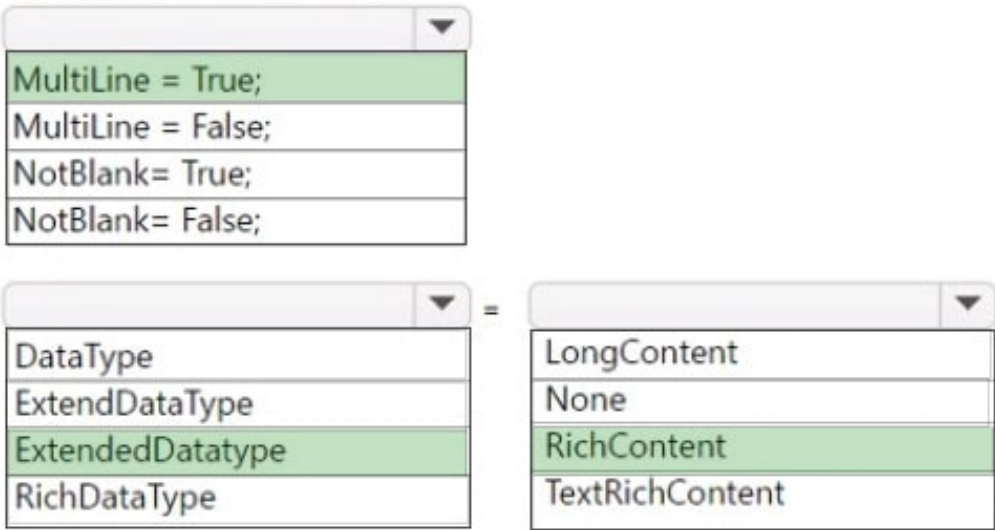
;
  }
}
```

```
var
  NonConformityComments: Text;
```

Correct Answer:

ExtendedDataType property

```
group(commentsGroup)
{
  field("comments"; NonConformityComments)
  {
    ApplicationArea = All;
    MultiLine = True;
    MultiLine = False;
    NotBlank = True;
    NotBlank = False;
    DataType
    ExtendDataType
    ExtendedDatatype
    RichDataType
  }
}
```



The diagram illustrates the configuration of the ExtendedDataType property. It shows a dropdown menu with 'ExtendedDatatype' selected, and another dropdown menu with 'RichContent' selected. The first dropdown menu also shows 'MultiLine = True;' and 'MultiLine = False;' options.

```
var
  NonConformityComments: Text;
```

Scenario: When a purchase order with incorrect quantity or quality issues is received, the entity must create a non-conformity document in the system. The following information must be included in the document:

*
Comments: can include comments with rich text and pictures to illustrate quality problems

*
etc.

Box 1: MultiLine = true;

Rich Text and content controls

The Rich Text feature in Business Central is designed to handle multimedia content, such as social media posts, email bodies, quick annotations, file contents, or any field that requires a mix of text, tables, links, and images.

Both ExtendedDataType and Multiline are required to render a Rich Text Editor ExtendedDataType = RichContent;
MultiLine = true;

Box 2: ExtendedDataType

Box 3: RichContent

Reference: <https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/devenv-richtext-content-controls>

QUESTION 3

HOTSPOT

You need to write the code to call the subcontractor's REST API.

How should you complete the code segment? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Hot Area:

REST services

```
procedure CallSubcontractorAPI(Url: Text[2048]; Username: Text[100]; Password:
Text[100]; Body: Text)
var
    httpClient: HttpClient;
    ResponseMessage: HttpResponseMessage;
    RequestHeaders, ContentHeaders: HttpHeaders;
    httpContent: HttpContent;
    Base64Convert: Codeunit "Base64 Convert";
    Response: Text;
begin
    RequestHeaders := httpClient.DefaultRequestHeaders();
    RequestHeaders.Add(
        'Authentication', 'Basic ' +
        Base64Convert.FromBase64(Username + ':' + Password)
    );

    httpClient.GetHeaders(ContentHeaders);
    ContentHeaders.Remove('Content-Type');
    ContentHeaders.Add('Content-Type', 'application/json');

    httpContent := Body;
    httpContent.Clear();
    httpContent.WriteFrom(Body);

    if
        httpClient.Post(Url, httpContent)
        httpClient.Post(Url, httpContent, Response)
        httpClient.Post(Url, httpContent, ResponseMessage)
        httpClient.Send(Url, httpContent, ResponseMessage)
    then
        ResponseMessage.Content.ReadAs(Response);
end;
```

Correct Answer:

REST services

```
procedure CallSubcontractorAPI(Url: Text[2048]; Username: Text[100]; Password:
Text[100]; Body: Text)
var
    httpClient: HttpClient;
    ResponseMessage: HttpResponseMessage;
    RequestHeaders, ContentHeaders: HttpHeaders;
    httpContent: HttpContent;
    Base64Convert: Codeunit "Base64 Convert";
    Response: Text;
begin
    RequestHeaders := httpClient.DefaultRequestHeaders();
    RequestHeaders.Add(
        'Authentication', 'Basic ' +
        Base64Convert.FromBase64(Username + ':' + Password)
        Base64Convert.ToBase64(Username + ':' + Password)
        Base64Convert.ToBase64(Username) + Base64Convert.ToBase64>Password)
        Username + ':' + Password
    );

    httpContent.GetHeaders(ContentHeaders);
    ContentHeaders.Remove('Content-Type');
    ContentHeaders.Add('Content-Type', 'application/json');
    httpContent := Body;
    httpContent.Clear();
    httpContent.WriteFrom(Body);

    if
        httpClient.Post(Url, httpContent)
        httpClient.Post(Url, httpContent, Response)
        httpClient.Post(Url, httpContent, ResponseMessage)
        httpClient.Send(Url, httpContent, ResponseMessage)
    then
        ResponseMessage.Content.ReadAs(Response);
end;
```

Box 1: \\Authorization\\

Business Central and the AL language have made web service code much easier with the HttpClient and Json types available. Handling the HTTP Authorization header is easier too with the TempBlob table, which can now encode the basic

authentication string using base64.

See below for an example of how to add a basic authorisation header to the AL HttpClient:

```
procedure AddHttpBasicAuthHeader(UserName: Text[50]; Password: Text[50], var HttpClient : HttpClient);
```

```
var
```

```
    AuthString: Text;
```

```
    TypeHelper: "Type Helper";
```

```
begin
```

```
    AuthString := STRSUBSTNO('%1:%2', UserName, Password);
```

```
    AuthString := TypeHelper.ConvertValueToBase64(AuthString);
```

```
    AuthString := STRSUBSTNO('Basic %1\\', AuthString);
```

```
HttpClient.DefaultRequestHeaders().Add(\\Authorization\\', AuthString);
```

end;

Box 2: Base64Convert.ToBase64(Username + \\:\\ + Password)

How To Connect To External APIs From Business Central (HTTP Request)

Authentication

There are several methods of authentication, basic, OAuth, etc. In the next example, you can see a way to use basic authentication.

We introduce 2 new types, HttpRequestMessage and HttpHeaders. They become fundamental when creating more complex API calls.

The way to build the call is quite simple:

Set the information for the message

Set the authorization in the header of the message

Send the request

Read and handle the response

Example:

Box 3: httpContent.WriteFrom(Body)

Example

The following example illustrates how to use the HttpContent type to send a simple POST request containing JSON data.

```
// Add the payload to the content content.WriteFrom(payload);
```

```
// Replace the default content type header with a header associated with this request  
content.GetHeaders(contentHeaders); contentHeaders.Clear(); contentHeaders.Add(\\Content-Type\\',  
\\application/json\\');
```

```
// Assigning content to request.Content will actually create a copy of the content and assign it.
```

```
// After this line, modifying the content variable or its associated headers will not reflect in
```

```
// the content associated with the request message
```

```
request.Content := content;
```

```
request.SetRequestUri(uri);
```

```
request.Method := \\POST\\';
```

Box 4: httpClient.Post(Uri,httpContent,ResponseMessage) POST as required, and include ResponseMessage.

Scenario: The API for sending subcontracting orders must be called by sending an authenticated POST request to the given endpoint.

Scenario, full:

An external business partner of Contoso, Ltd. exposed a REST API for receiving details about new subcontracting orders and for sending the planned release date of each subcontracting order received.

Integration with business partner for subcontracting

Contoso, Ltd. must connect Business Central to the external API provided by the business partner. This will be used for the partner to send the details of new subcontracting orders to fulfill the sales demand, and for receiving the planned

release date of each order sent. The integration requirements are as follows:

The business partner will provide a REST API secured with basic authentication. Credentials to access the API will be shared with Contoso, Ltd.

The API for sending subcontracting orders must be called by sending an authenticated POST request to the given endpoint.

The API for retrieving the order no. and planned release date of each subcontracting order responds with the following JSON:

Reference:

<https://dankinsella.blog/http-basic-authentication-with-the-al-httpclient/> <https://businesscentralgeek.com/how-to-connect-to-external-apis-from-business-central-http-request> <https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/methods-auto/httpcontent/httpcontent-data-type>

QUESTION 4

HOTSPOT

You create an "AddItemsToJson" procedure and publish it.

```
01 procedure AddItemsToJson() RequestText: Text
02 var
03     Item: Record Item;
04     ItemObject: JsonObject;
05     ItemsArray: JsonArray;
06 begin
07     Clear(ItemsArray);
08     Clear(ItemObject);
09     If Item.FindSet() then begin
10         repeat
11             ItemObject.Add('No', Item."No.");
12             ItemObject.Add('Description', Item.Description);
13             ItemsArray.Add(ItemObject);
14         until Item.Next() = 0;
15         ItemsArray.WriteTo(RequestText);
16     end;
17 end;
```


The procedure fails to run.

You need to fix the errors in the code.

For each of the following statements, select Yes if the statement is true. Otherwise, select No.

Hot Area:

JSON file processing

Statement	Yes	No
In line 13, replace the Add method with Insert.	<input type="radio"/>	<input type="radio"/>
In line 15, replace the WriteTo method with ReadFrom.	<input type="radio"/>	<input type="radio"/>
Change the ItemObject variable type from JsonObject to JsonToken.	<input type="radio"/>	<input type="radio"/>
Move line 08 in the beginning of REPEAT..UNTIL.	<input type="radio"/>	<input type="radio"/>

Correct Answer:

JSON file processing

Statement	Yes	No
In line 13, replace the Add method with Insert.	<input type="radio"/>	<input checked="" type="radio"/>
In line 15, replace the WriteTo method with ReadFrom.	<input type="radio"/>	<input checked="" type="radio"/>
Change the ItemObject variable type from JsonObject to JsonToken.	<input type="radio"/>	<input checked="" type="radio"/>
Move line 08 in the beginning of REPEAT..UNTIL.	<input checked="" type="radio"/>	<input type="radio"/>

In line 13, replace the Add method with Insert. = NO
In line 15, replace the WriteTo method with ReadFrom. = NO
Change the ItemObject variable type from JsonObject to JsonToken. = NO
Move line 08 in the beginning of REPEAT .. UNTIL.

= YES The provided code is intended to serialize a list of items from the Item table into a JSON array format. Here is a breakdown of the code and the necessary corrections:

In line 13, "ItemsArray.Add(ItemObject)": This line is correctly using the Add method to add the ItemObject to the ItemsArray. The Add method is the correct method to use for adding items to a JsonArray. Therefore, there is no need to replace

Add with Insert.

In line 15, "ItemsArray.WriteTo(RequestText)": The WriteTo method is used correctly to serialize the ItemsArray into a JSON formatted string and store it in the RequestText variable. The ReadFrom method is used for the opposite operation,

i.e., to deserialize a JSON formatted string into a JSONArray, which is not the goal in this context. Hence, no change is needed here. Change the ItemObject variable type from JsonObject to JsonToken: The ItemObject variable is intended to

hold JSON objects representing individual items, making JsonObject the appropriate type. JsonToken is not a type used in this context within AL for Business Central, and thus the variable type should remain as JsonObject.

Move line 08, "Clear(ItemObject)": This line should be moved inside the repeat loop to ensure that the ItemObject is cleared for each item in the loop. Placing it before the repeat would only clear it once before the loop starts, which could lead

to incorrect serialization as the previous item's properties would not be cleared from the ItemObject.

The logic for serializing records into JSON is a common operation when interfacing with APIs or web services in Business Central, and the pattern shown in the code is typical for such operations.

QUESTION 5

You are cleaning up sandbox environments for a company.

The company requires data to be cleared from the environments each time an extension is published.

You need to configure the launch.json file.

Which schemaUpdateMode property should you set?

- A. ForceUpgrade
- B. ForceSync
- C. Synchronize
- D. Recreate

Correct Answer: D

In the context of cleaning up sandbox environments for a company where data needs to be cleared each time an extension is published, the schemaUpdateMode property in the launch.json file should be set to Recreate (D). Setting this property to Recreate ensures that every time the extension is published, the existing tables and data are dropped, and then the tables are recreated based on the current extension's schema. This mode is particularly useful in development and testing environments where you need a clean slate for testing each version of the extension without the remnants of previous data affecting the outcomes. It's important to use this setting cautiously, as it results in the loss of all existing data in the tables defined by the extension, which is suitable for a sandbox environment but not for production environments.

QUESTION 6

You need to define the data types for the fields of the Non-conformity table.

Which two data types should you use? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

- A. Integer for the Non-conformity Number field
- B. DateTime for the Non-conformity Date field
- C. Char for the Non-conformity Number field
- D. Date for the Non-conformity Date field
- E. Code for the Non-conformity Number field

Correct Answer: CD

Scenario: When a purchase order with incorrect quantity or quality issues is received, the entity must create a non-conformity document in the system. The following information must be included in the document:

*

Non-conformity Number: must use the No. Series table from Business Central online to manage this field and use these features: Alphanumeric values Number format that includes "NC" and the year as part of the number; for example, NC24-001

*

Non-conformity Date: stores only the creation date

Reference: <https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/devenv-format-property>

QUESTION 7

DRAG DROP

You need to handle the removal of the Description field and the Clone procedure without breaking other extensions.

Which three 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.

NOTE: More than one order of answer choices is correct. You will receive credit for any of the correct orders you select.

Select and Place:

Actions

Set the Clone procedure as ObsoleteState = Pending and ObsoleteReason = 'Not in use' in version 2.0.0.0.

Set the Description field as ObsoleteState = Pending and ObsoleteReason = 'Not in use' in version 2.0.0.0.

Set the Description field as ObsoleteState = Removed; in version 2.0.0.1.

Remove the Description field in version 2.0.0.0.

Set the Clone procedure as ObsoleteState = Removed; in version 2.0.0.1.

Remove the Clone procedure in version 2.0.0.0.

Remove the Description field from the Issue table in version 2.0.0.1.

Add the [Obsolete('xxx')] attribute to the Clone procedure in version 2.0.0.0.

Actions to handle the field and procedure removal

Correct Answer:

Actions

Set the Description field as ObsoleteState = Pending and ObsoleteReason = 'Not in use' in version 2.0.0.0.

Remove the Description field in version 2.0.0.0.

Set the Clone procedure as ObsoleteState = Removed; in version 2.0.0.1.

Remove the Clone procedure in version 2.0.0.0.

Add the [Obsolete('xxx')] attribute to the Clone procedure in version 2.0.0.0.

Actions to handle the field and procedure removal

Set the Clone procedure as ObsoleteState = Pending and ObsoleteReason = 'Not in use' in version 2.0.0.0.

Set the Description field as ObsoleteState = Removed; in version 2.0.0.1.

Remove the Description field from the Issue table in version 2.0.0.1.

Step 1: Set the Clone procedure as ObsoleteState = Pending and ObsoleteReason = '\\Not in use\\' in version 2.0.0.0
ObsoleteState Property Marks whether the object will be deprecated.

Property Value

*

No

Not obsolete. This is the normal/default setting.

*

Pending

Will become obsolete in a future version.

Syntax

```
ObsoleteState = Pending;
```

Remarks

By coding against this property, you can use this property as a way to communicate through code to other developers which objects and elements will become obsolete over time and those which are already obsolete, enabling them to adjust

their application code accordingly.

Note

For all elements, except for Tables and Table fields, setting ObsoleteState = Removed will throw Compiler Error AL0169 because after an appropriate warning state of Pending, these elements can be deleted.

Note

When developing using Dynamics NAV Development Environment (C/SIDE), you do not get warnings or errors when you compile objects that reference table objects, fields, or keys that are marked as Pending or Removed.

ObsoleteState

property is only detected by the AL compiler, which will return warnings for references to elements marked as Pending and errors for references to elements marked as Removed.

Step 2: Set the Description field as ObsoleteState = Removed in version 2.0.01

Step 3: Remove the Description field from the issue table in version 2.0.0.1

Note: ObsoleteReason Property

Specifies why the object has been marked as Pending in the ObsoleteState property.

Syntax

```
ObsoleteReason = '\\Not Needed\\';
```

Remarks

Use this property to inform developers about an object or element that will become obsolete in time or is already obsolete. Use the ObsoleteTag Property to specify additional information which can be valuable to other developers.

Scenario:

The Issue Management process must be split into two extensions:

-

ISSUE BASE: main extension

-

ISSUE EXT: second extension with dependency from ISSUE BASE

In the version 1.0.0.0 of the ISSUE BASE extension, you plan to create an Issue table that contains a global Decimal variable named IssueTotal.

In the version 1.0.0.0 of the ISSUE BASE extension, you plan to define a table named Issue Category with a Description field defined as follows:

```
field(2; Description; Text[50])
{
    DataClassification = CustomerContent;
}
```

The Issue table defined in ISSUE BASE extension contains a Clone procedure defined as follows:

```
procedure Clone()
begin
end;
```

In the ISSUE EXT extension, you create a tableextension object of the Issue table.

The tableextension object of the Issue table must access the IssueTotal: Decimal variable.

After weeks of usage, you discover that you must remove the Description field and the Clone procedure because they are no longer required.

Reference:

<https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/properties/devenv-obsoletestate-property>

<https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/properties/devenv-obsolete-reason-property>

QUESTION 8

HOTSPOT

You plan to create a table to hold client data.

You have the following data integrity requirements:

1.

Lookups into other records must be established.

2.

Validate if a record exists in a destination record.

You need to select the table field property to use for each requirement.

Which table field property should you use? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Hot Area:

Build tables

Requirement

Establish lookups into other records.

Validate if a record exists.

Table field property

	▼
DataClassification	
ExternalAccess	
TableRelation	
ValidateTableRelation	
	▼
CalcFormula	
Access	
AccessByPermission	
ValidateTableRelation	

Correct Answer:

Build tables

Requirement

Establish lookups into other records.

Validate if a record exists.

Table field property

	▼
DataClassification	
ExternalAccess	
TableRelation	
ValidateTableRelation	
	▼
CalcFormula	
Access	
AccessByPermission	
ValidateTableRelation	

For the data integrity requirements, the table field properties to use are:

To establish lookups into other records, use the TableRelation property. To validate if a record exists in a destination record, use the ValidateTableRelation property.

In Business Central, when creating tables to hold data, maintaining data integrity is crucial:

TableRelation Property: This property is used to create a relationship between the field in one table and a field in another table, which is typically used for lookups. When you set the TableRelation property on a field, it allows users to select from a list of values that exist in the related table.

ValidateTableRelation Property: This property is used to ensure that the value entered in a field matches one of the values in a related table. If a user tries to enter a value that doesn't exist in the related table, an error will occur.

QUESTION 9

DRAG DROP

A company is examining Connect apps and Add-on apps for use with Business Central. You need to describe the development language requirements for Connect apps and Add-on apps.

How should you describe the app language requirements? To answer, move the appropriate app types to the correct descriptions. You may use each app type 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.

Select and Place:

App types		Connect app and Add-on app descriptions	App type
Add-on app	•	Description	
Connect app	•	Developed by using any coding language	
	•	Developed by using AL language in Visual Studio Code	

Correct Answer:

App types		Connect app and Add-on app descriptions	App type
	•	Description	Connect app
	•	Developed by using any coding language	
	•	Developed by using AL language in Visual Studio Code	Add-on app

Developed by using any coding language: Connect app
Developed by using AL language in Visual Studio Code: Add-on app

In Microsoft Dynamics 365 Business Central, there are distinct types of applications that can be developed: Connect apps and Add-on apps. Each has its own development language requirements:

Connect apps:

Add-on apps:

The language and environment used for developing these apps are key differentiators between Connect apps and Add-on apps.

QUESTION 10

HOTSPOT

A company plans to customize its per tenant extension reports. The company has the following requirements for the customization:

1.

Child data items must not be displayed on the request page for some master detail reports.

2.

Selecting key filter fields takes users too much time. The customization must decrease the amount of time to select the fields.

You need to optimize the report request page.

Which actions should you configure? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Hot Area:

Report request page

Observation

Child data items of some master detail reports must not be displayed on the request page.

Decrease the amount of time to select filter fields.

Action

<input type="checkbox"/>	Set the Print:OnlyIfDetail property to true.
<input type="checkbox"/>	Set the UseRequestPage property to true.
<input type="checkbox"/>	Set the DataItemTableView sorting property.
<input type="checkbox"/>	Set the DataItemLinkReference property to the parent data item.
<input type="checkbox"/>	Set the SaveValues Property to true.
<input type="checkbox"/>	Specify the request page options.
<input type="checkbox"/>	Specify the RequestFilterfields property.
<input type="checkbox"/>	Specify the RequestFilterHeading property.

Correct Answer:

Report request page

Observation

Child data items of some master detail reports must not be displayed on the request page.

Decrease the amount of time to select filter fields.

Action

Set the Print:OnlyIfDetail property to true.
Set the UseRequestPage property to true.
Set the DataItemTableView sorting property.
Set the DataItemLinkReference property to the parent data item.

Set the SaveValues Property to true.
Specify the request page options.
Specify the RequestFilterFields property.
Specify the RequestFilterHeading property.

For the given requirements, you should configure the following actions:

For child data items not to be displayed on the request page for some master- detail reports, set the DataItemLinkReference property to the parent data item. To decrease the amount of time to select key filter fields, specify the

RequestFilterHeading property.

In Dynamics 365 Business Central, when customizing report request pages, certain properties can be set to control the behavior and display of the report options:

Hiding Child Data Items:The DataItemLinkReference property is used to link a child data item to a parent data item in the data model of a report. Setting this property correctly will ensure that the child data items are related to the correct parent data item and will be displayed or hidden accordingly on the request page. If the goal is to prevent child data items from being displayed, you need to make sure they are correctly linked and configured to not appear. **Optimizing Filter**

Field Selection:The RequestFilterHeading property is used to group filter fields on the request page. By specifying this property, you can create a more organized and user-friendly interface, which can significantly speed up the process of selecting filters. This property allows you to categorize filters into headings, making it quicker and easier for users to find and set the necessary filters for the report.

By adjusting these properties on the report request page as part of the per tenant extension customization, you will address the company's requirements to optimize the user experience when running reports.

QUESTION 11

HOTSPOT

You need to create the Install codeunit that is required in the extension used for installing or updating the Housekeeping app.

Which data type or declaration should you use? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Hot Area:

Data types or declarations for an Install codeunit

Requirement

Data type for information

Start of the declaration of the method or procedure to perform the tasks

Data type or declaration

ModuleDependencyInfo
ModuleInfo
SessionInformation

global procedure
local procedure
procedure

Correct Answer:

Data types or declarations for an Install codeunit

Requirement

Data type for information

Start of the declaration of the method or procedure to perform the tasks

Data type or declaration

ModuleDependencyInfo
ModuleInfo
SessionInformation

global procedure
local procedure
procedure

Box 1: ModuleInfo

Data type for information

ModuleInfo

Represents information about an application consumable from AL.

Scenario: Department-specific requirements. Housekeeping department

The department requires the development of an extension with a new API page named RoomsAPI.

*

The housekeeping team will use RoomsAPI to publish room details, update when work is complete, or provide repair notifications from the canvas app.

*-> This custom API page must expose a custom table named Rooms and have an ID 50000. The table must be able to update from the PMS. The PMS team must know the *endpoint to connect to the custom API*.

Note: Data Types and Methods in AL

The following data types are available as part of the AL Language. Each data type has various methods that support it.

*

ModuleInfo

Represents information about an application consumable from AL.

Incorrect:

*

ModuleDependencyInfo

Provides information about a dependent module.

*

SessionInformation

Is a complex data type for exposing Session information into AL.

Box 2: local procedure Procedure scope To declare a local method, start the declaration with local:

local procedure Mymethod(); To declare a global method, omit local: procedure Mymethod();

Scenario: The code required to perform tasks cannot be accessible from other parts of the application. Reference:
<https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/methods-auto/library>
<https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/devenv-al-methods>

QUESTION 12

You create a Business Central report.

You need to insert values on the Request page to be saved for the next time the report is run.

What should you do?

- A. Set the Transact! on Type property to Update.
- B. Declare a Savevalues variable and assign it to true on the OnOpenPage () trigger.
- C. Set the Use Request Page property to true.
- D. Set the SaveValues property to true.

Correct Answer: D

To ensure that the values inserted on the Request page of a Business Central report are saved for the next time the report is run, the SaveValues property (D) should be set to true. This property is available on the Request page of the report and, when set to true, allows the system to remember the values entered by the user, so they do not have to re-enter them each time they run the report. This feature enhances user experience by reducing repetitive data entry and ensuring consistency in report parameters across multiple executions. The other options mentioned, such as setting the Transaction Type property to Update (A) or declaring a Savevalues variable in the OnOpenPage trigger (B), are not directly related to saving user input on a report's Request page.