Adobe

Exam Questions AD0-E724

Adobe Commerce Developer Professional

NEW QUESTION 1
What is the default store ID for the admin panel?

A. 1
B. -1

Answer: A

Explanation:
In Magento, the default store ID for the admin panel is 0. This is used as a fallback mechanism where, if the current store view's ID is not 0, Magento automatically adds 0 as a fallback. This ensures that the admin panel has a unique identifier, differentiating it from the frontend store views, which typically start with IDs higher than 0. This setup is crucial for Magento's multi-store architecture, allowing for distinct configurations and behaviors between the admin and frontend contexts.

NEW QUESTION 2
An Adobe Commerce developer is creating a module (Vendor.ModuleName) to be sold on the Marketplace. The new module creates a database table using declarative schema and now the developer needs to make sure the table is removed when the module is disabled.
What must the developer do to accomplish this?

A. There is nothing further the developer needs to d
B. The table will be removed when the module is disabled and bin/magento setup:upgrade is run.
C. There is nothing further the developer needs to d
D. The table will be removed when the when bin/magento module:uninstall vendor_ModuleName is run.
E. Add a schema patch that implements Magento\Framework\setup\Patch\PatchRevertabieinterface and drops the table in the revert function.

Answer: A

Explanation:
When you disable a module, Magento removes all of its declarative schema changes from the database the next time you run bin/magento setup:upgrade."
This means that when the developer disables the module and runs setup:upgrade, Adobe Commerce will automatically handle the removal of the database table created by the module's declarative schema.
For reference, here are some key points from the documentation:
? [Disable a Module](https://x.com/i/grok?text=Disable%20a%20Module)- This section explains how Magento handles the database schema when a module is disabled.
? Declarative Schema Configuration- Provides an overview of how declarative
schema works, including its behavior when modules are disabled or uninstalled. Therefore, based on the official Adobe Commerce documentation, the correct action for the
developer is to do nothing further beyond disabling the module and running bin/magento setup:upgrade. Magentowill take care of removing the table associated with the module as part of its schema management process.

NEW QUESTION 3
Which file should a developer use to set the default value when creating configuration fields for admin?

A. etc/adminhtml/config.xml
B. etc/config xml
C. etc/adminhtml/system.xml

Answer: A

Explanation:
When creating configuration fields for the admin panel and setting their default values, a developer should use theetc/config.xmlfile within their module. This file is used to declare default values for the module's configuration options. When Magento is installed or the module is enabled, these values are automatically loaded into the database, under thecore_config_datatable, setting the initial configuration for the module. This approach ensures that the module has sensible defaults and operates correctly upon installation

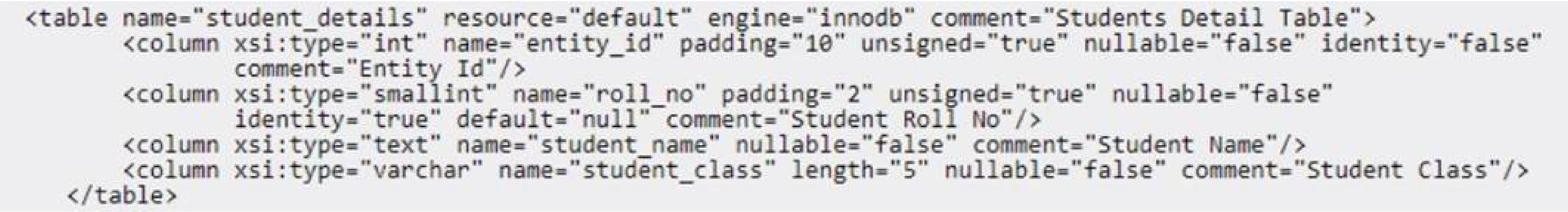
NEW QUESTION 4
How can a developer prioritize a plugin's execution, if possible?

A. The developer can use sortOrder property by specifying a lower value than the target plugin.
B. The developer can use sortOrder property by specifying a higher value than the target plugin.
C. This cannot be achieved as the plugins are always executed by their module's load order in app/etc/config.php file.

Answer: A

Explanation:
A developer can prioritize the execution of a plugin by using thesortOrder property within the plugin's declaration in thedi.xmlfile. Specifying a lower value for the sortOrderproperty gives the plugin higher priority, meaning it will be executed before other plugins with a higher sortOrder value. This allows developers to control the order of plugin execution, which can be critical for ensuring the desired outcome when multiple plugins are affecting the same method.

NEW QUESTION 5
An Adobe Commerce developer is trying to create a custom table using declarative schema, but is unable to do so.



```
  <table name="student_details" resource="default" engine="innodb" comment="Students Detail Table">
        <column xsi:type="int" name="entity_id" padding="10" unsigned="true" nullable="false" identity="false"
               comment="Entity Id"/>
        <column xsi:type="smallint" name="roll_no" padding="2" unsigned="true" nullable="false"
               identity="true" default="null" comment="Student Roll No"/>
        <column xsi:type="text" name="student_name" nullable="false" comment="Student Name"/>
        <column xsi:type="varchar" name="student_class" length="5" nullable="false" comment="Student Class"/>
    </table>
```

What are two errors in the snippet above? (Choose two.)

A. Column (roll_no) does not have inde
B. It is needed since attribute identity is set to true.
C. Column (entity_id) does not have inde
D. It is needed since attribute identity is set to false.
E. Column (student_name) does not have attribute length.
F. null is not a valid value for column (roll_no).

Answer: AC

Explanation:
The correct answers are A and C. The errors in the snippet are:
? Column roll_no does not have an index. It is needed since attribute_identity is set to true.
? Column student_name does not have an attribute length. Theattribute_identityattribute specifies whether the primary key of the table should be auto-incremented. Ifattribute_identityis set totrue, then theroll_nocolumn must have an index. Thestudent_namecolumn does not have an attribute length, which is required for string columns.
The following code shows how to fix the errors: XML

```
<table name="vendor_module_table">
<entity_id>
<type>int</type>
<identity>true</identity>
<unsigned>true</unsigned>
<nullable>false</nullable>
</entity_id>
<roll_no>
<type>int</type>
<identity>false</identity>
<unsigned>true</unsigned>
<nullable>false</nullable>
<primary_key>true</primary_key>
<index>true</index>
</roll_no>
<student_name>
<type>string</type>
<length>255</length>
<nullable>false</nullable>
</student_name>
</table>
```

Once the errors have been fixed, the table can be created successfully.

NEW QUESTION 6
An Adobe Commerce developer adds a new extension attribute to add an array of values to the invoices that are fetched through the APIs.
After a while, their technical manager reviews their work and notices something wrong with the extension_attributes. xml file that the developer created in their module:
What is the problem with this xml snippet?

A. The extension attribute references the wrong interface, it should have referenced the Magento\saies\Api\data\invoiceinterface.
B. The extension attribute references the repository instead of the interface it implements (Magento\saies\Api\invoiceRepositorymterface).
C. The type is wrong, string [] should be replaced with array.

Answer: A

Explanation:
When adding extension attributes in Adobe Commerce, it's essential to reference the correct interface in the extension_attributes.xml file. Extension attributes in Magento are used to add custom data fields to existing entities, and these should extend the core entity interfaces, not their repositories.
In this case:
? The developer needs to reference Magento\Sales\Api\Data\InvoiceInterface, which represents the data structure of invoices, rather than any repository or incorrect
interfaces.
? The extension_attributes.xml should specify the data interface (e.g., Magento\Sales\Api\Data\InvoiceInterface) to ensure the extension attribute is correctly applied to the invoice entity and is accessible when fetching invoices through APIs.
Correct XML format:

```
<?xml version="1.0"?>
<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="urn:magento:module:Magento_ExtensionAttributes:etc/extension_attributes.xsd">
<extension_attributes for="Magento\Sales\Api\Data\InvoiceInterface">
<attribute code="your_custom_attribute" type="string[]" />
</extension_attributes>
</config>
```

Additional Resources:
? Adobe Commerce Developer Guide: Using Extension Attributes
? Extension Attributes XML Reference

NEW QUESTION 7
An Adobe Commerce developer has created a before plugin for the save() function within the Magento\Framework\App\cache\Proxy class. The purpose of this plugin is to add a prefix on all cache identifiers that fulfill certain criteria.
Why is the plugin not executing as expected?

A. Another around plugin defined for the same function does not call the callable.
B. Cache identifiers are immutable and cannot be changed.
C. The target ClaSS implements Magento\Framework\ObjectManager\NoninterceptableInterface.

Answer: C

Explanation:
According to the Plugins (Interceptors) guide for Magento 2 developers, plugins are class methods that modify the behavior of public class methods by intercepting them and running code before, after, or around them. However, some classes in Magento 2 implement the NoninterceptableInterface interface, which prevents plugins from being generated for them. The Magento\Framework\App\cache\Proxy class is one of them, as it extends from Magento\Framework\ObjectManager\NoninterceptableInterface. Therefore, the plugin is not executing as expected because the target class implements NoninterceptableInterface. Verified References: https://devdocs.magento.com/guides/v2.3/extension-dev-guide/plugins.html

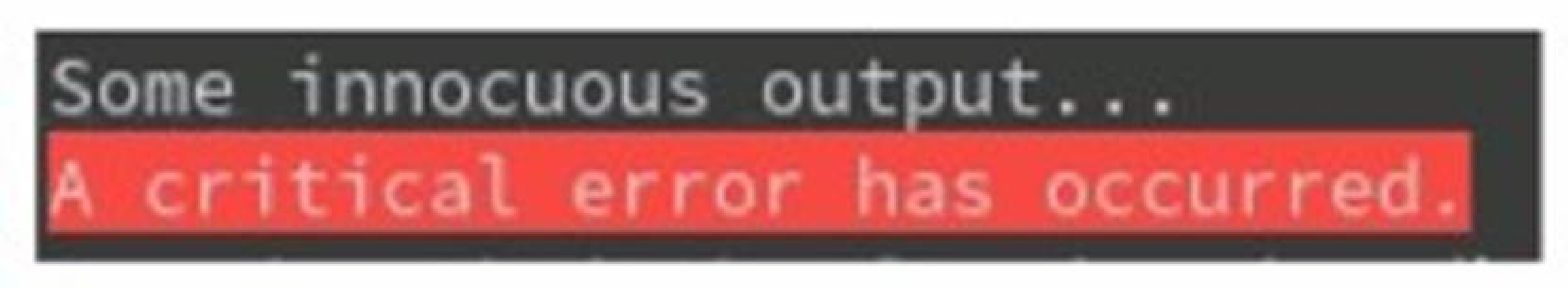
NEW QUESTION 8
How should a developer display a custom attribute on the category edit page in the admin panel when a new module Vendor.Category is created?

A. Create view/adminhtml/layout/catalog_category_edit xml in the module, and then define a block that would display the field for the attribute.
B. The field for the attribute will appear automatically.
C. Create view/adminhtml/ui_component/category_form.xml file in the module, and then define the field for the attribute.

Answer: C

Explanation:
To display a custom attribute on the category edit page in the Magento admin panel, a developer should use the UI component approach. This involves creating a category_form.xmlfile within theview/adminhtml/ui_componentdirectory of the module. This XML file defines the form fields (including any custom attributes) that should appear on the category edit page. The UI component system in Magento provides a flexible way to manage form elements and their interactions on the admin side, ensuring a consistent user

NEW QUESTION 9
An Adobe Commerce developer is creating a new console command to perform a complex task with a lot of potential terminal output. If an error occurs, they want to provide a message that has higher visibility than some of the other content that may be appearing, so they want to ensure it is highlighted in red (as seen in the screenshot):



```
Some innocuous output...
A critical error has occurred.
```

How can they customize the appearance of this message?

A. Call the setDecorationType(Stype) method On the Symfony\Console\Output\OutputInterface Object before Calling writeln().
B. Wrap the output content in tags like <error>, <info>, or <comment>.
C. Throw a new commandException with the desired message passed as an argument.

Answer: B

Explanation:
In Adobe Commerce, when developing custom console commands, you can customize output messages by using special tags provided by Symfony Console, which Adobe Commerce relies on. These tags are designed to help differentiate types of messages and can be used to add color or emphasis to the output, enhancing visibility. For critical error messages, wrapping the message in the <error> tag will display it in red, as shown in your screenshot. The available tags include:
? <error> for red-colored error messages.
? <info> for informational messages (often displayed in blue).
? <comment> for comments or warnings (usually yellow).
$output->writeln('<error>A critical error has occurred.</error>');
This method is effective and widely used for output customization in Symfony-based console commands.
Additional Resources:
? Adobe Commerce Developer Guide: Console Command Customization
? Symfony Console Output Formatting

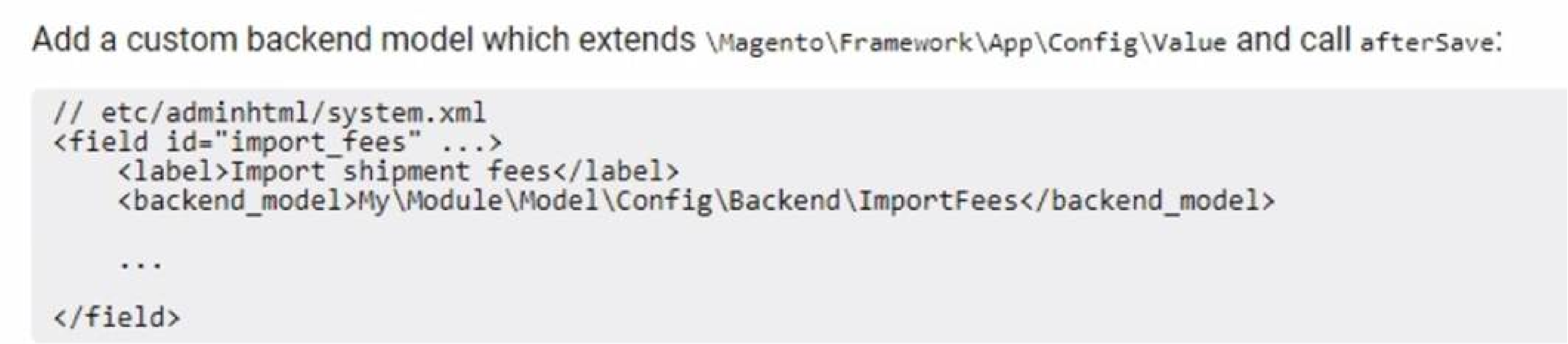
NEW QUESTION 10
What are two ways to access the PHP error logs on Adobe Commerce Cloud? (Choose Two.)

A. Use the dedicated command from Cloud CLI for Commerce.
B. Navigate to the dedicated entry in the Project Web Interface.
C. Connect to the the servers via SSH and localize the log files.
D. Use the Adobe Admin Log application.

Answer: AC

Explanation:
Two ways to access the PHP error logs on Adobe Commerce Cloud are to use the dedicated command from Cloud CLI for Commerce and to connect to the servers via SSH and localize the log files. The Cloud CLI for Commerce is a command-line tool that allows developers to interact with their Adobe Commerce Cloud projects and environments. The developer can use the command magento-cloud log php to view or download the PHP error logs from any environment.
In Adobe Commerce Cloud, accessing PHP error logs can be done through the Cloud CLI or by directly connecting to the server via SSH.
? Cloud CLI for Accessing Logs:
? uk.co.certification.simulator.questionpool.PList@3e5e7f0f
? SSH for Direct Log Access:
? Why Options A and C are Correct:
: Adobe Commerce Cloud documentation onAccessing Logs

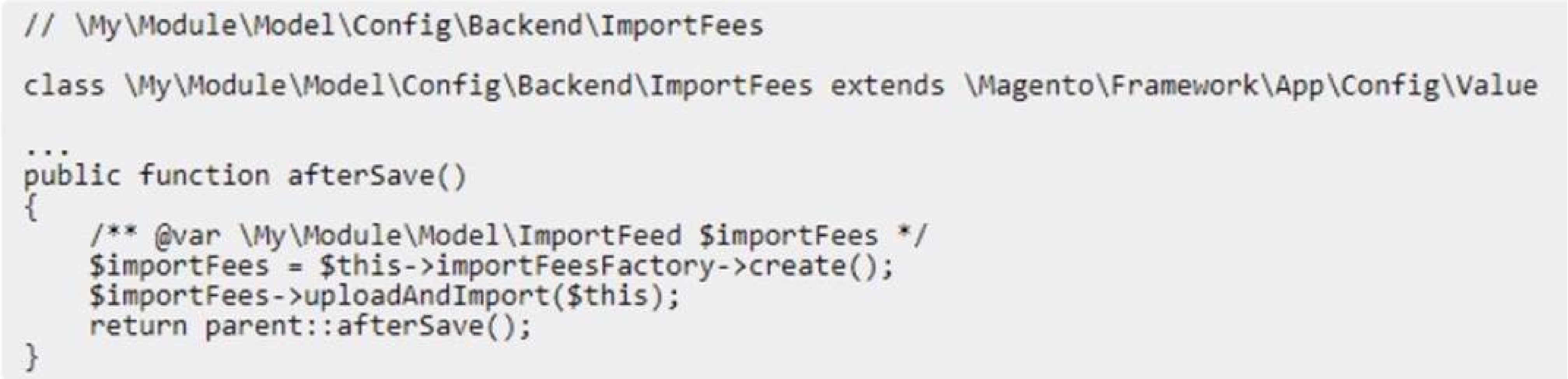
NEW QUESTION 10
A logistics company with an Adobe Commerce extension sends a list of reviewed shipment fees to all its clients every month in a CSV file. The merchant then uploads this CSV file to a "file upload" field in admin configuration of Adobe Commerce.
What are the two requirements to display the "file upload" field and process the actual CSV import? (Choose two.)
A)



Add a custom backend model which extends `\Magento\Framework\App\Config\Value` and call `afterSave`:

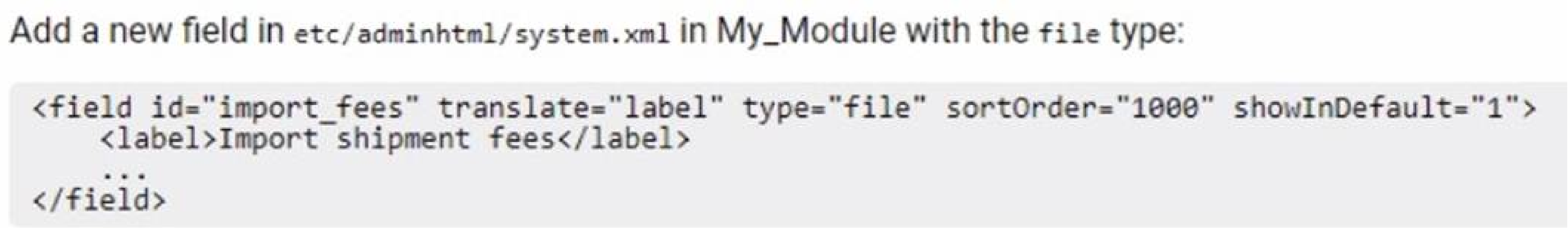
```
// etc/adminhtml/system.xml
<field id="import_fees" ...>
    <label>Import shipment fees</label>
    <backend_model>My\Module\Model\Config\Backend\ImportFees</backend_model>

    ...

</field>
```

B)



```
// \My\Module\Model\Config\Backend\ImportFees

class \My\Module\Model\Config\Backend\ImportFees extends \Magento\Framework\App\Config\Value

...
public function afterSave()
{
    /** @var \My\Module\Model\ImportFeed $importFees */
    $importFees = $this->importFeesFactory->create();
    $importFees->uploadAndImport($this);
    return parent::afterSave();
}
```

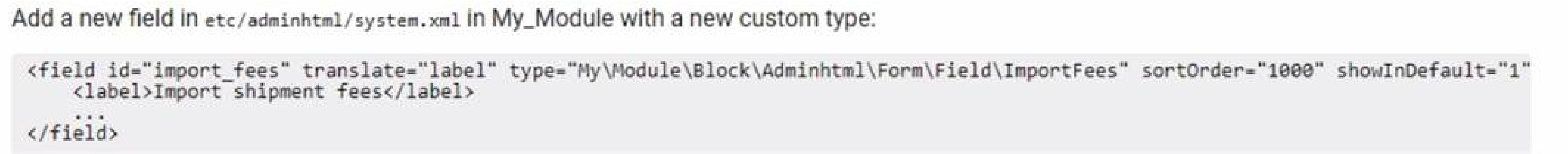
C)



Add a new field in `etc/adminhtml/system.xml` in My_Module with the `file` type:

```
<field id="import_fees" translate="label" type="file" sortOrder="1000" showInDefault="1">
    <label>Import shipment fees</label>
    ...
</field>
```

D)



Add a new field in `etc/adminhtml/system.xml` in My_Module with a new custom type:

```
<field id="import_fees" translate="label" type="My\Module\Block\Adminhtml\Form\Field\ImportFees" sortOrder="1000" showInDefault="1"
    <label>Import shipment fees</label>
    ...
</field>
```

A. Option A
B. Option B
C. Option C
D. Option D

Answer: BC

Explanation:
The file type within system.xml instructs Magento to render a file input field. This is crucial for allowing users to upload a file in the admin configuration, which then can be processed by the backend model.
References: The Magento official documentation outlines how system.xml configuration files can be used to add custom fields to the system configuration in the Magento backend.
Options A and D are incorrect because they do not specifically address the requirements of setting up a file upload field and processing a CSV import. Option A configures a backend model but lacks the necessary file type definition. Option D defines a custom type for the file field but does not directly address the file upload or CSV processing requirements.

NEW QUESTION 11
A custom theme Is being developed in the Adobe Commerce store, and the developer needs to override the current parent theme styles.

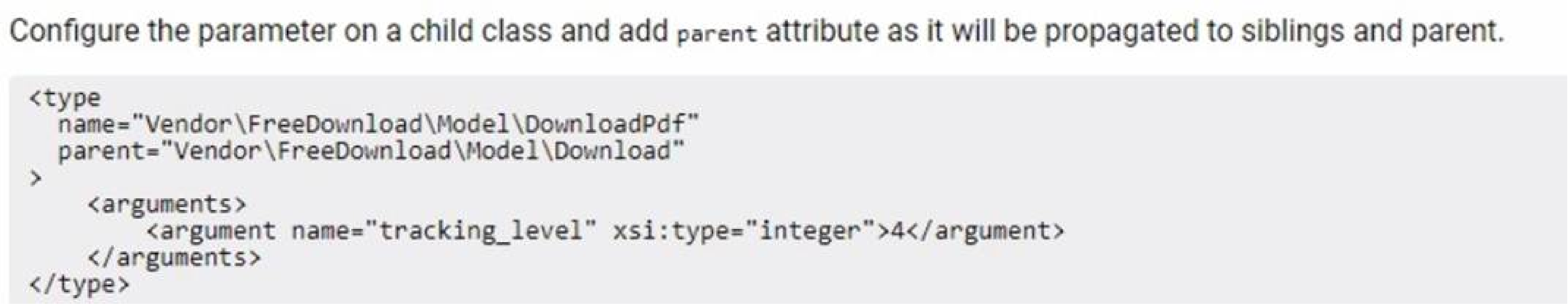
Which less file should the developer use to achieve this goal?

A. _theme.override.less
B. _hemeless
C. _source.less

Answer: B

Explanation:
To override the current parent theme styles in a custom theme being developed for Adobe Commerce, the developer should use the_theme.lessfile. This file is specifically designed for customizing and overriding the default styles provided by the parent theme, making option B the correct choice. The_theme.lessfile is a central place for theme-specific customizations.

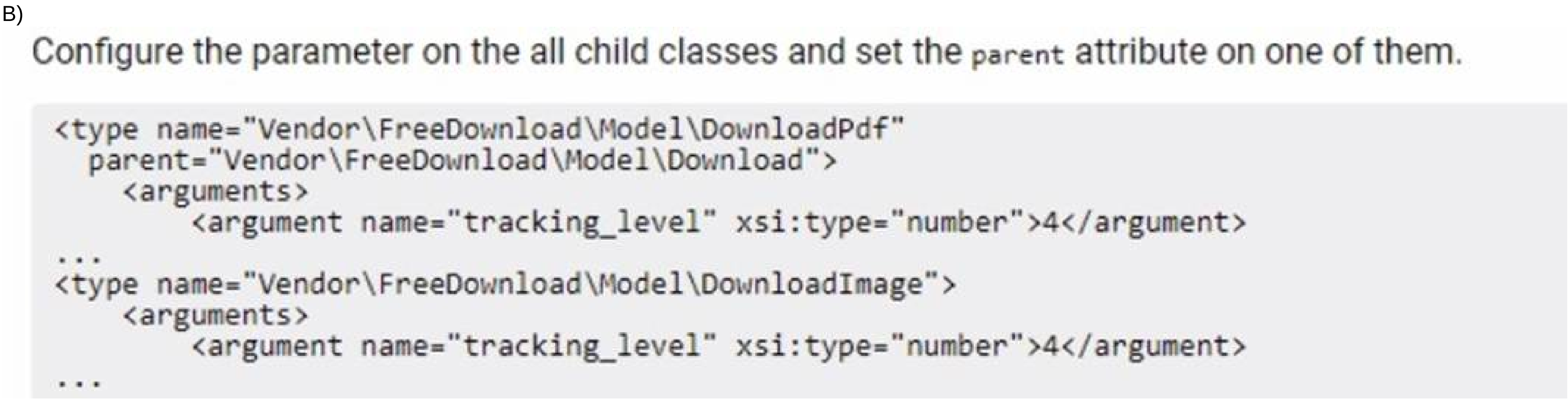
NEW QUESTION 12
An Adobe Commerce developer is asked to change the tracking level on a custom module for free downloading of pdf and images.
The module contains following models: Vendor\FreeDownload\Model\Download Vendor\FreeDownload\Model\DownloadPdf extends Vendor\FreeDownload\Model\Download
Vendor\FreeDownload\Model\DownloadImage extends Vendor\FreeDownload\Model\Download
Download class has a parameter for tracking_level.
How will the developer configure the tracking_level parameter, in di.xml.to have a value of 4 for Download class and all classes that extend Download?
A)



Configure the parameter on a child class and add `parent` attribute as it will be propagated to siblings and parent.

```
<type
  name="Vendor\FreeDownload\Model\DownloadPdf"
  parent="Vendor\FreeDownload\Model\Download"
>
    <arguments>
        <argument name="tracking_level" xsi:type="integer">4</argument>
    </arguments>
</type>
```

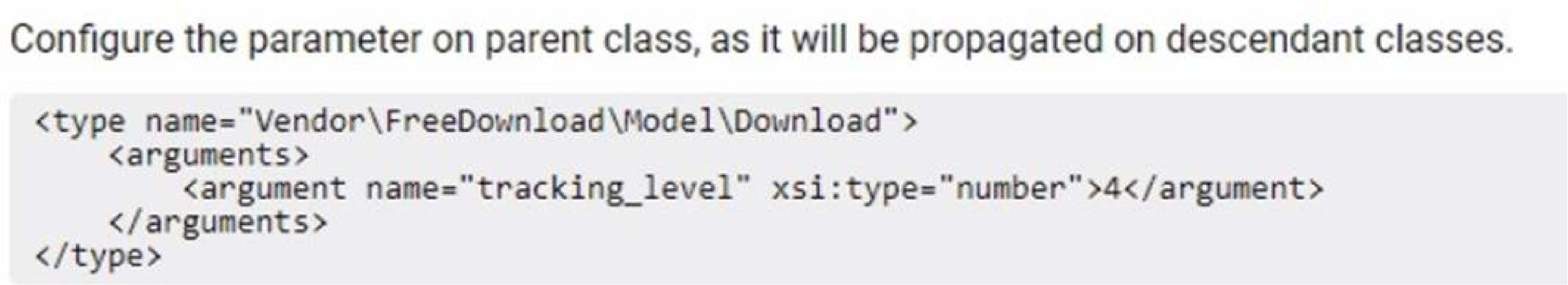
B)



Configure the parameter on the all child classes and set the `parent` attribute on one of them.

```
<type name="Vendor\FreeDownload\Model\DownloadPdf"
   parent="Vendor\FreeDownload\Model\Download">
      <arguments>
          <argument name="tracking_level" xsi:type="number">4</argument>
...
<type name="Vendor\FreeDownload\Model\DownloadImage">
      <arguments>
          <argument name="tracking_level" xsi:type="number">4</argument>
...
```

C)



Configure the parameter on parent class, as it will be propagated on descendant classes.

```
<type name="Vendor\FreeDownload\Model\Download">
    <arguments>
        <argument name="tracking_level" xsi:type="number">4</argument>
    </arguments>
</type>
```

A. Option A
B. Option B
C. Option C

Answer: C

Explanation:
To configure a parameter for a parent class so that it propagates to all descendant classes, the correct approach is to define the parameter on the parent class within di.xml. This way, all child classes inheriting from this parent will automatically use the parameter value unless explicitly overridden.
Option C is correct for the following reasons:
? Configuring on the Parent Class (Vendor\FreeDownload\Model\Download):By setting the tracking_level parameter directly on the Download class, you ensure that all classes extending this class, such as DownloadPdf andDownloadImage, will inherit the tracking_level parameter value. This method leverages Magento's dependency injection configuration, which allows parameters set on a parent class to cascade to child classes.
? uk.co.certification.simulator.questionpool.PList@15a6494
: Magento??s official developer documentation outlines that class dependencies and configuration parameters defined in di.xml at a higher level are accessible to descendant classes. This is a standard practice in Magento for setting parameters that affect a hierarchy of classes.
Avoiding Redundant Configuration:Unlike Option A, which sets the parameter on an individual child class, or Option B, which redundantly sets the parameter on multiple child classes, Option C is optimal as it centralizes the configuration. This reduces the risk of discrepancies and simplifies maintenance.
Options A and B are incorrect because:
Option A configures the parameter on a single child class, which will not affect other child classes such as DownloadImage.

Option B redundantly sets the parameter for each child class individually, which is unnecessary when the parameter can be inherited from the parent.

NEW QUESTION 14
A developer is making customizations in the checkout, and access to the quotes shipping address is needed. Which file provides the shipping address of the current quote?

A. Magento_Checkout/js/model/quote
B. Magento_Quote/js/model/model
C. Magento_Checkout/js/model/quote-shipping-address

Answer: A

Explanation:
This file provides the shipping address of the current quote by using the getShippingAddress() method. For example, the following code snippet gets the shipping address from the quote object and logs it to the console:
define(['Magento_Checkout/js/model/quote'
],function(quote) { 'use strict';
varshippingAddress = quote.getShippingAddress(); console.log(shippingAddress);
});
The file Magento_Quote/js/model/model does not exist in Magento 2, and the file Magento_Checkout/js/model/quote-shipping-address is not a valid way to access the shipping address of the current quote. You can read more about the quote object and its methods in the Magento 2 developer documentation.
In Adobe Commerce, the shipping address of the current quote is accessed through the JavaScript fileMagento_Checkout/js/model/quote. This file includes various quote-related data, including shipping and billing addresses, items in the cart, and totals. There is no Magento_Checkout/js/model/quote-shipping-addressfile, and Magento_Quote/js/model/modelis not a valid path, making option A the correct choice.

NEW QUESTION 18
Which is a correct CMS content element in Adobe Commerce?

A. Widget
B. Sheet
C. Image

Answer: A

Explanation:
A widget is a CMS content element that can be used to display dynamic content on a page. Widgets can be used to display things like product reviews, social media feeds, or even custom content.
In Adobe Commerce, widgets are a correct CMS content element. Widgets allow merchants to add dynamic data or content blocks to CMS pages, static blocks, and various other locations throughout the store's layout without needing to directly edit the site's code. Options B (Sheet) and C (Image) are not recognized CMS content elements in the context of Adobe Commerce's terminology, making option A the correct answer.

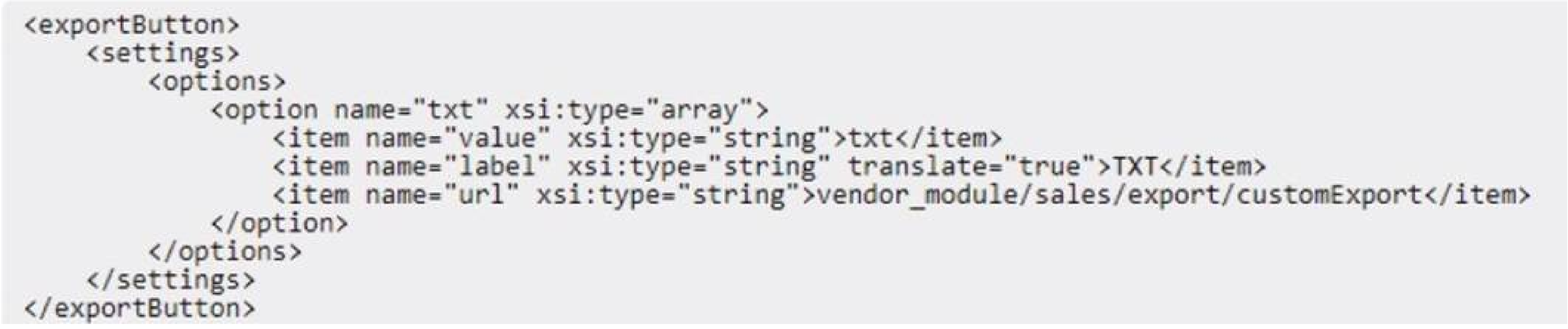
NEW QUESTION 20
A developer is investigating a problem with the shopping cart. Some of the cart records in the database are being checked. Which table should the developer check?

A. sates.quote
B. sales.cart
C. quote

Answer: C

Explanation:
? A quote is a temporary object that represents the customer??s shopping cart before it is converted into an order1. A quote can be created by a guest or a registered customer, and it can be active or inactive depending on whether the customer has completed the checkout process or not2.
? The quote table stores the basic information about the quote, such as the customer ID, email, currency, subtotal, grand total, shipping method, payment method, and so on2. Each row in the quote table corresponds to one quote object, identified by a unique entity ID2.
? The quote item table stores the details of each product added to the quote, such
as the product ID, SKU, name, price, quantity, discount amount, tax amount, and so on2. Each row in the quote item table corresponds to one quote item object, identified by a unique item ID2. A quote item object is associated with a quote object by the quote ID column in the quote item table2.
? When a customer adds a product to the cart, Magento creates or updates a quote
object and a quote item object for that product. The quote object is initialized by the Magento\Quote\Model\Quote class, which implements the Magento\Quote\Api\Data\CartInterface interface3. The quote item object is initialized by the Magento\Quote\Model\Quote\Item class, which implements the Magento\Quote\Api\Data\CartItemInterface interface3.
? Magento uses various classes and interfaces to manipulate the quote and quote
item objects, such as the Magento\Quote\Model\QuoteRepository class, which implements the Magento\Quote\Api\CartRepositoryInterface interface for saving and retrieving quotes from the database3, and the Magento\Quote\Model\QuoteManagement class, which implements the Magento\Quote\Api\CartManagementInterface interface for creating and submitting quotes to orders3.
? Magento also uses various events and observers to perform additional actions on
the quote and quote item objects, such as applying discounts, calculating taxes, validating stock availability, and so on. Some of these events are: sales_quote_add_item, sales_quote_collect_totals_before, sales_quote_collect_totals_after, sales_quote_save_before, sales_quote_save_after, sales_model_service_quote_submit_before, sales_model_service_quote_submit_after.
? Magento provides various APIs for interacting with the quote and quote item objects, such as the Magento\Quote\Api\CartManagementInterface API for creating empty carts and placing orders from carts, the Magento\Quote\Api\CartItemRepositoryInterface API for adding, updating, and removing items from carts, and the Magento\Quote\Api\CouponManagementInterface API for managing coupons for carts.

NEW QUESTION 24
An Adobe Commerce developer is tasked with adding an new export option for the order grid, they have added the following code for the export button within sales_order_grid.xml:



```
<exportButton>
    <settings>
        <options>
            <option name="txt" xsi:type="array">
                <item name="value" xsi:type="string">txt</item>
                <item name="label" xsi:type="string" translate="true">TXT</item>
                <item name="url" xsi:type="string">vendor_module/sales/export/customExport</item>
            </option>
        </options>
    </settings>
</exportButton>
```

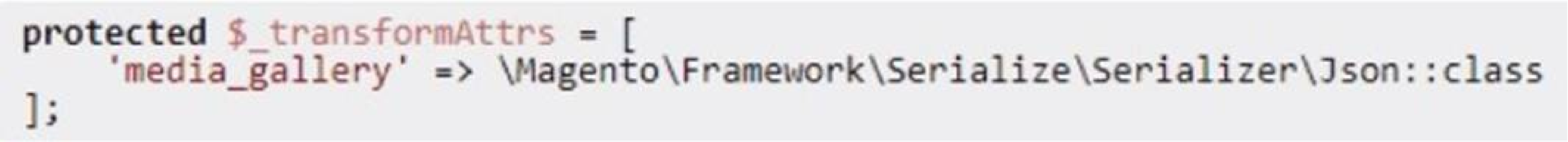
Upon testing, they are getting redirected, what would be a cause for this error?

A. The option's uri attribute is not valid.
B. The layout cache needs to be refreshed.
C. The developer has to add a formkey for the new export option.

Answer: C

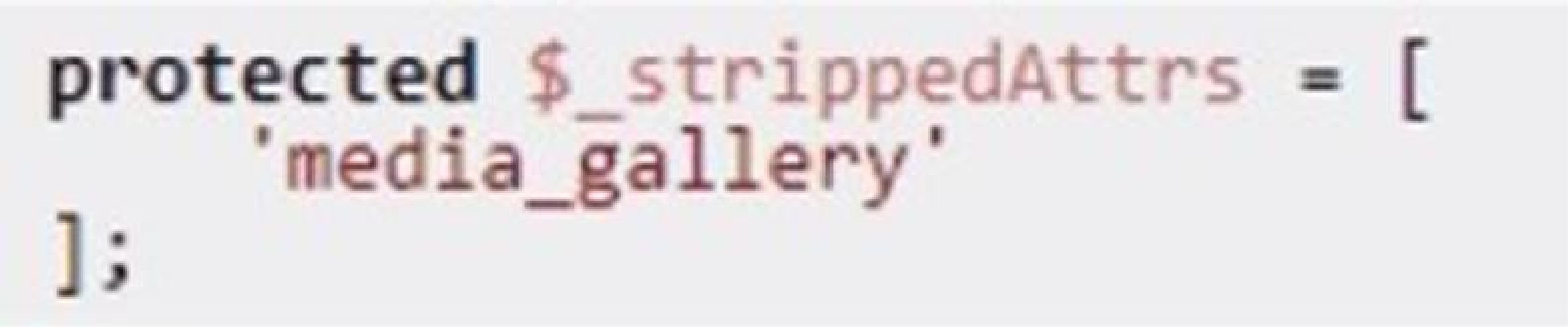
Explanation:
The developer has to add a formkey for the new export option because the formkey is required for security reasons. Without the formkey, the request will be rejected and redirected to the dashboard page. Verified References: [Magento 2.4 User Guide] [Magento 2.4 DevDocs]
When adding custom export options to grids in Magento, it's crucial to include a form key for actions that involve form submission. Magento relies on form keys for CSRF (Cross-Site Request Forgery) protection, so omitting the form key can lead to redirects or failed operations.
? Form Key Requirement:
? uk.co.certification.simulator.questionpool.PList@755b4fcf
? Why Option C is Correct:
? Solution:
: Adobe Commerce documentation onForm Key and CSRF Protection Magento DevDocs onAdding Buttons to Grids

NEW QUESTION 27
An Adobe Commerce Developer has written an importer and exporter for a custom entity. The client is using this to modify the exported data and then re-importing the file to batch update the entities.
There is a text attribute, which contains information related to imagery in JSON form, media_gallery. This is not a field that the client wants to change, but the software they are using to edit the exported data seems to be modifying it and not allowing it to import correctly.
How would the developer prevent this?
A) Specify a serializer class for the attribute using the $_transformAttrs class property array for both the exporter and importer so it gets converted:



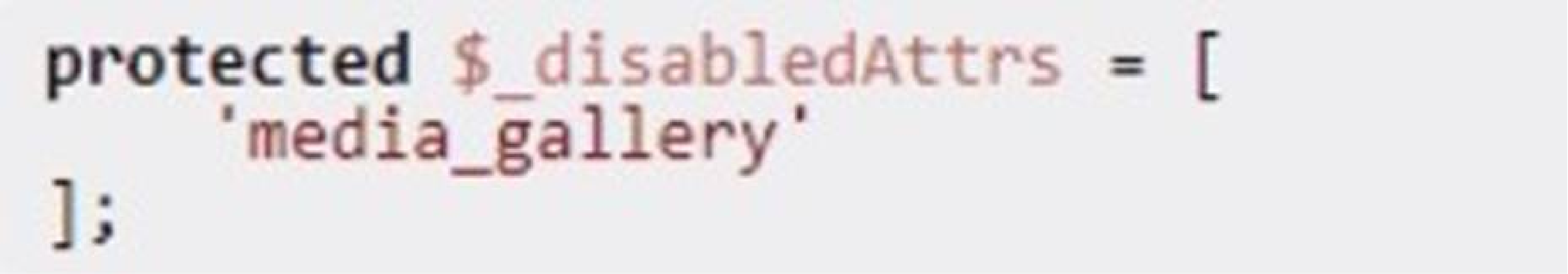
```
protected $_transformAttrs = [
    'media_gallery' => \Magento\Framework\Serialize\Serializer\Json::class
];
```

B) Strip the attribute from the imported file by adding it to the s_strippedAttrs class property array:



```
protected $_strippedAttrs = [
    'media_gallery'
];
```

C) Prevent it from being exported by adding it to the $_disat>iedAttrs class property array:



```
protected $_disabledAttrs = [
    'media_gallery'
];
```

A. Option A
B. Option B
C. Option C

Answer: C

Explanation:
To prevent the media_gallery attribute from being exported as part of the custom entity's data, the developer should add this attribute to the $_disabledAttrs class property array. This effectively excludes the attribute from the export process, ensuring that it does not appear in the exported file and thus will not be modified or re-imported inadvertently.
Option C is correct for the following reasons:
? Preventing Export with $_disabledAttrs:Adding media_gallery to the
$_disabledAttrs array tells the system to skip this attribute during export. This prevents the attribute from being included in the export file, thus removing the risk of

it being altered by external software that handles the file. Since the attribute will not be present in the exported data, it will remain unchanged in the database when re-importing.
? uk.co.certification.simulator.questionpool.PList@6993fe6
: The Adobe Commerce documentation on import/export processes outlines how
$_disabledAttrs can be used to filter out sensitive or unnecessary attributes from export files.
Alternative Options and Their Limitations:
Option A: Using $_transformAttrs with a serializer is useful for encoding or decoding attribute data during export/import, but it does not prevent the attribute from being included in the export. This would only help if the media_gallery data needed transformation, not exclusion.
Option B: $_strippedAttrs is applicable for filtering attributes from the imported file, not the exported file. It would not stop the attribute from being included in the export and hence does not align with the need to prevent modifications during export.
By configuring $_disabledAttrs, the developer effectively ensures the media_gallery attribute remains unmodified by preventing it from being included in export files altogether.

NEW QUESTION 31
ECE-Tools provides a set of tools that can be used to manage and maintain your Adobe Commerce Cloud environment. What are some of the features provided by ECE-Tools?

A. Builds application, Applies custom patches and Dump configuration for static content deployment.
B. Fastly configuration, Applies custom patches and Dump configuration for static content deployment.
C. Builds application, Applies custom patches, and Shows the list of S3 backup tar.gz files.

Answer: A

Explanation:
Some of the features provided by ECE-Tools are building application, applying custom patches, and dumping configuration for static content deployment. ECE-Tools is a set of scripts and tools designed to manage and deploy Adobe Commerce Cloud projects. It provides commands for building application code, applying patches for Magento core issues or custom modules, and dumping configuration settings for static content deployment optimization. Verified References: [Magento 2.4 DevDocs] 2
The ECE-Tools package for Adobe Commerce Cloud provides a range of tools and scripts to manage and streamline deployment and maintenance tasks. Among its key features:
? Application Builds:
? uk.co.certification.simulator.questionpool.PList@2bc44d9b
? Applying Custom Patches:
? Dump Configuration for Static Content Deployment:
? Why Option A is Correct:
: Adobe Commerce Cloud documentation onECE-Tools

NEW QUESTION 33
Which price type should be used if the developer wants to provide a discount for a product based on quantity, for example, being able to buy two for X amount each?

A. Tier Price
B. Special Price
C. Group Price

Answer: A

Explanation:
Tier prices are used to provide discounts for products based on quantity. For example, you could set a tier price that allows customers to buy two products for X amount each.
The tier price is used when a developer wants to offer a discount based on the quantity purchased. For example, buying two or more units of a product at a reduced price per unit. Tier pricing allows setting different prices for different quantities, encouraging customers to buy more. Special price is a flat discounted price regardless of quantity, and group price is used to set special prices for specific customer groups, not for quantity-based discounts.

NEW QUESTION 34
What action can be performed from the Cloud Project Portal (Onboarding Ul) of an Adobe Commerce Cloud project?

A. Set your developer SSH public key.
B. Update Project and environment variables
C. Add a Technical Admin

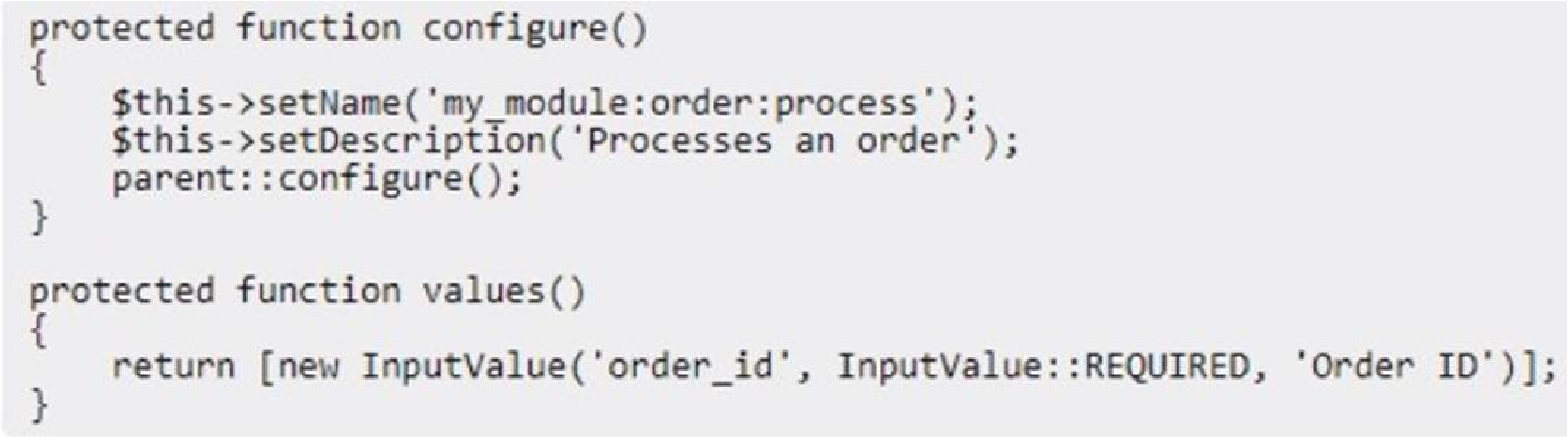
Answer: C

Explanation:
The Cloud Project Portal (Onboarding UI) of an Adobe Commerce Cloud project is a web interface that allows you to perform various actions related to your project, such as creating and managing environments, deploying code, configuring services, and adding users1. One of the actions that you can perform from the Cloud Project Portal is adding a Technical Admin, which is a user role that has full access to all environments and canperform any action on the project2. To add a Technical Admin from the Cloud Project Portal, you need to follow these steps2:
? Log in to the Cloud Project Portal with your Magento account credentials.
? Click on the Users tab on the left sidebar.
? Click on the Add User button on the top right corner.
? Enter the email address of the user you want to add as a Technical Admin.
? Select the Technical Admin role from the Role dropdown menu.
? Click on the Send Invitation button.
The user will receive an email invitation to join your project as a Technical Admin. They will need to accept the invitation and set up their account before they can access your project2.

NEW QUESTION 38
An Adobe Commerce developer has created a process that exports a given order to some external accounting system. Launching this process using the Magento

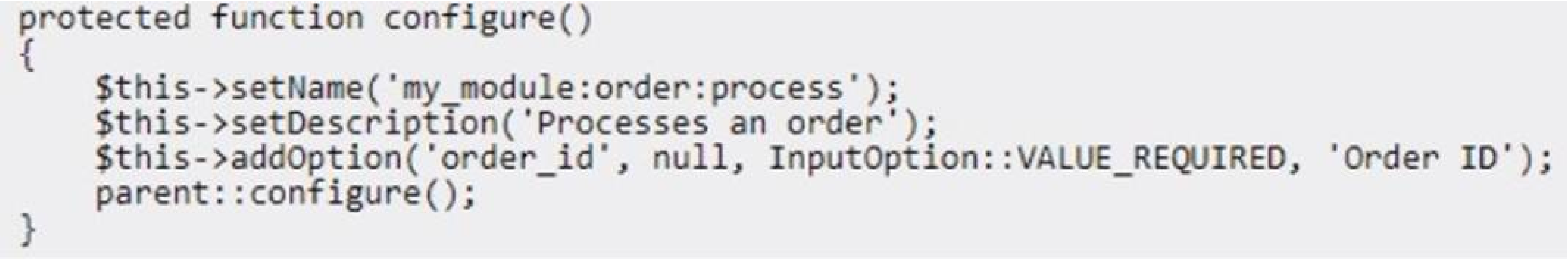
CLI with the command php bin/magento my_module:order: process --order_id=<order_id> is required.
Example: php bin/magento my_module:order:process --order_id=1245. What is the correct way to configure the command?

A)



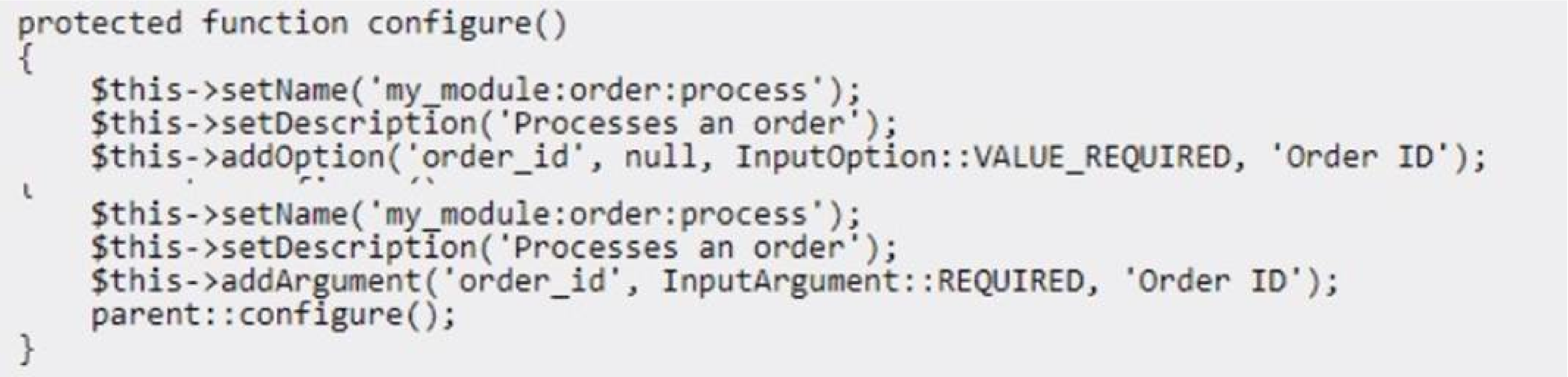
```
protected function configure()
{
    $this->setName('my_module:order:process');
    $this->setDescription('Processes an order');
    parent::configure();
}

protected function values()
{
    return [new InputValue('order_id', InputValue::REQUIRED, 'Order ID')];
}
```

B)



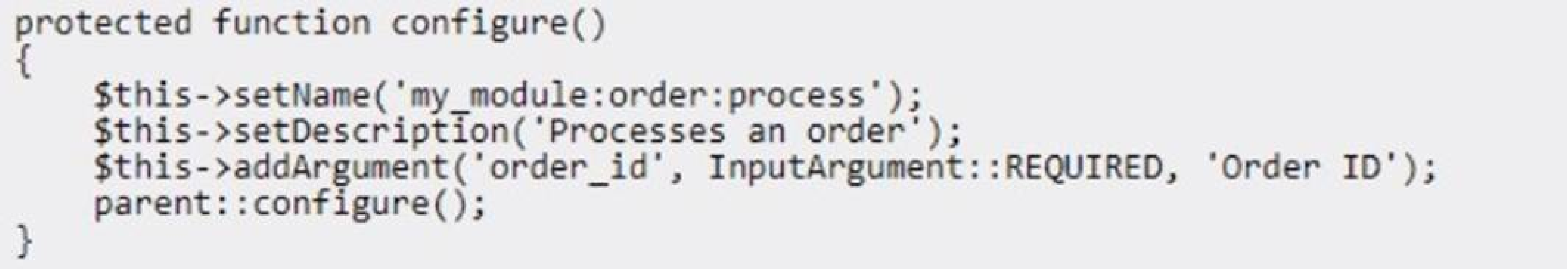
```
protected function configure()
{
    $this->setName('my_module:order:process');
    $this->setDescription('Processes an order');
    $this->addOption('order_id', null, InputOption::VALUE_REQUIRED, 'Order ID');
    parent::configure();
}
```

C)



```
protected function configure()
{
    $this->setName('my_module:order:process');
    $this->setDescription('Processes an order');
    $this->addOption('order_id', null, InputOption::VALUE_REQUIRED, 'Order ID');

    $this->setName('my_module:order:process');
    $this->setDescription('Processes an order');
    $this->addArgument('order_id', InputArgument::REQUIRED, 'Order ID');
    parent::configure();
}
```

D)



```
protected function configure()
{
    $this->setName('my_module:order:process');
    $this->setDescription('Processes an order');
    $this->addArgument('order_id', InputArgument::REQUIRED, 'Order ID');
    parent::configure();
}
```

A. Option B
B. Option C
C. Option C
D. Option D

Answer: D

Explanation:
To properly configure a Magento CLI command that includes a required argument, such as --order_id, which is mandatory for processing an order, the best approach is to use the addArgument method within the configure function. This method defines required arguments for the command, ensuring the user provides the necessary data.

Option D is correct for the following reasons:

? Using addArgument for Required Inputs:The addArgument method is used here to declare order_id as a required argument. This is more appropriate than addOption
when the parameter is essential for command execution and should not be omitted. By specifying InputArgument::REQUIRED, the command ensures that the order_id must be provided by the user.

? uk.co.certification.simulator.questionpool.PList@3259569d
: According to Magento??s official developer documentation, addArgument is used for required command-line arguments, and this is standard practice for defining necessary inputs in CLI commands.

Properly Configured Command Name and Description:The setName and setDescription methods are correctly used in this option to specify the command??s name and its purpose. This helps in making the command self-descriptive, improving usability and readability for other developers or administrators using the CLI.

Options A, B, and C are incorrect because they either:
Use addOption instead of addArgument, which is less suitable for mandatory parameters (Option B).
Define the same parameter redundantly (Option C).

Use other non-standard configurations that do not align with Magento's best practices for required CLI parameters (Option A).

NEW QUESTION 40

A merchant of an Adobe Commerce Cloud project wants to setup one of their websites using a subdomain. The merchant is considering the domain to be set as secondstore.example.com.
In addition to editing the magento-vars.php file, and apply a domain check and set
$_SERVER["MAGE_RUN_CODE"] and $_SERVER["MAGE_RUN_TYPE"].
What file is required to perform this action?

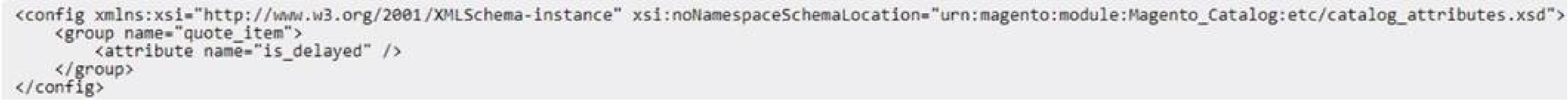
A. Configure secondstore.example.com subdomain route in NGINX virtual-host configuration file.
B. Configure secondstore.example.com subdomain route in .magento/services.yaml.
C. Configure secondstore.example.com subdomain route in .magento/routes.yaml.

Answer: C

Explanation:
In Adobe Commerce Cloud, routing configurations for custom domains and subdomains are managed within the .magento/routes.yaml file. This file defines how requests are directed to the application and is essential for setting up different stores or websites with unique subdomains.
? Using .magento/routes.yaml for Subdomain Configuration:
? uk.co.certification.simulator.questionpool.PList@277b7595
? Why Option C is Correct:
? Example Configuration: http://secondstore.example.com/: type: upstream
upstream: "mymagento:80"
:
Adobe Commerce Cloud documentation onConfiguring routes.yaml

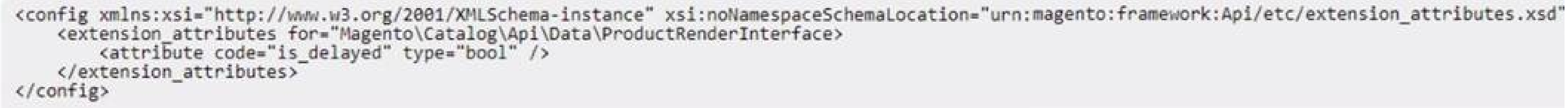
NEW QUESTION 42

There is the task to create a custom product attribute that controls the display of a message below the product title on the cart page, in order to identify products that might be delivered late.
The new EAV attribute is_delayed has been created as a boolean and is working correctly in the admin panel and product page.
What would be the next implementation to allow the is_delayed EAV attribute to be used in the .phtml cart page such as $block->getProduct()->getIsDelayed()?
A)
Create a new file etc/catalog_attributes.xmi:



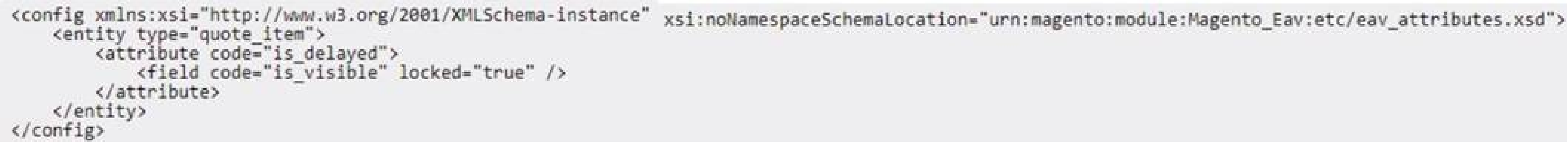
```
<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="urn:magento:module:Magento_Catalog:etc/catalog_attributes.xsd">
    <group name="quote_item">
        <attribute name="is_delayed" />
    </group>
</config>
```

B)
Create a new file etc/extension attributes.xmi:



```
<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="urn:magento:framework:Api/etc/extension_attributes.xsd"
    <extension_attributes for="Magento\Catalog\Api\Data\ProductRenderInterface>
        <attribute code="is_delayed" type="bool" />
    </extension_attributes>
</config>
```

C)
Create a new file etc/eav attributes.xmi:



```
<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="urn:magento:module:Magento_Eav:etc/eav_attributes.xsd">
    <entity type="quote_item">
        <attribute code="is_delayed">
            <field code="is_visible" locked="true" />
        </attribute>
    </entity>
</config>
```

A. Option A
B. Option B
C. Option C

Answer: A

Explanation:
To allow theis_delayedEAV attribute to be used in the .phtml cart page, the developer needs to create a new file calledetc/catalog_attributes.xmi. This file will contain the definition of theis_delayedattribute.
The following code shows how to create theetc/catalog_attributes.xmifile: XML
<?xml version="1.0"?>
<catalog_attributes>
<attribute code="is_delayed" type="int">
<label>Is Delayed</label>
<note>This attribute indicates whether the product is delayed.</note>
<sort_order>10</sort_order>
<required>false</required>
</attribute>
</catalog_attributes>
Once theetc/catalog_attributes.xmifile has been created, theis_delayedattribute will be available in the .phtml cart page. The attribute can be accessed using thegetIsDelayed()method of theProductclass.
PHP
$product = $block->getProduct();
$isDelayed = $product->getIsDelayed();
TheisDelayedvariable will contain the value of theis_delayedattribute. If the value of the attribute is 1, then the product is delayed. If the value of the attribute is 0, then the product is not delayed.

NEW QUESTION 43
Which action, if any, should be taken to forbid Adobe Commerce Admin from performing specific actions?

A. Create a new user role with custom-defined resources, and assign it to the admin user
B. This action cannot be taken since all admin users must have full access.
C. Enable custom roles in the store configuration, and assign admin user ID(s).

Answer: A

Explanation:
To forbid Adobe Commerce Admin from performing specific actions, a developer should create a new user role with custom-defined resources, and assign it to the admin user. This can be done by going toSystem > Permissions > Rolesand creating a new role. In theResourcessection, the developer can select the specific resources that they want to restrict the admin user from accessing.
To restrict specific actions within the Adobe Commerce Admin, the recommended approach is to utilize Magento's Access Control List (ACL). This can be done by creating a new user role with custom-defined resources and assigning this role to the admin user. This approach allows for granular control over what actions an admin user can perform by specifying allowed resources within the role. Magento's ACL system is designed to manage permissions effectively, ensuring that users only have access to the necessary functionalities required for their role.

NEW QUESTION 46
On the Adobe Commerce Cloud Project Web Interface, what will be performed when clicking on the "Delete" button of an integration environment?

A. The environment is marked as "inactive", the git branch is still present but the database is deleted.
B. The environment is completely delete
C. Including git branch and database.
D. The environment is marked as "inactive", the git branch and the database are still present.

Answer: B

Explanation:
On the Adobe Commerce Cloud Project Web Interface, clicking on the "Delete" button of an integration environment will completely delete the environment, including the associated git branch and database. This action is irreversible and is used to remove an environment that is no longer needed. The environment, once deleted, frees up resources for the project and cannot be restored.

NEW QUESTION 48
How would a developer add a sensitive environment-specific configuration value on an Adobe Commerce Cloud project?

A. Add sensitive Environment-specific variable in the Project Web Interface.
B. Connect to the server using SSH and add the configuration in the app/etc/config.php file.
C. Use the Cloud CLI for Commerce command Mgento-cloud config:set to add the configuration

Answer: A

Explanation:
To add a sensitive environment-specific configuration value on an Adobe Commerce Cloud project, the developer should use the Project Web Interface. This interface allows for the secure entry of sensitive data, which is then encrypted and stored securely. This method ensures that sensitive information is not exposed in the codebase or version control.

NEW QUESTION 50
Which action, if any, can be taken to change the URL key of the product?

A. The product URL key is generated automatically, so it cannot be changed.
B. Use URL rewrite to map product id with the custom URL key.
C. In the product admin form, under the Search Engine Optimization fieldset, the URL key can be set

Answer: C

Explanation:
The URL key of a product is the text that is used to generate the product's URL. This text can be set by the merchant in the product admin form.
The URL key of a product in Adobe Commerce can be changed in the product admin form under the "Search Engine Optimization" fieldset. Here, the URL key can be set or edited manually, allowing for customization beyond the automatically generated value. This field is typically used to ensure that the product URL is search-engine friendly and relevant to the product.

NEW QUESTION 53
A developer defined a new table in db.schema.xml while creating a new module.
What should be done to allow the removal of columns from the database when deleting them from db.schema.xml?

A. The removable columns should be defined in db_schema_whitelist.json.
B. The columns should have "removable" attribute set to "true" in the db.schema.xml.
C. The removable columns should be defined in db.schema_blacklist.json.

Answer: A

Explanation:
If a developer wants to allow the removal of columns from the database when deleting them fromdb.schema.xml, they need to define the removable columns in thedb_schema_whitelist.jsonfile. This file will tell Magento which columns can be removed from the database.
To allow columns to be removed from the database when they are deleted from db_schema.xml, they must be listed in thedb_schema_whitelist.jsonfile. This file acts as a reference for which database schema elements are safe to modify or delete, providing a safeguard against unintentional data loss during schema updates.

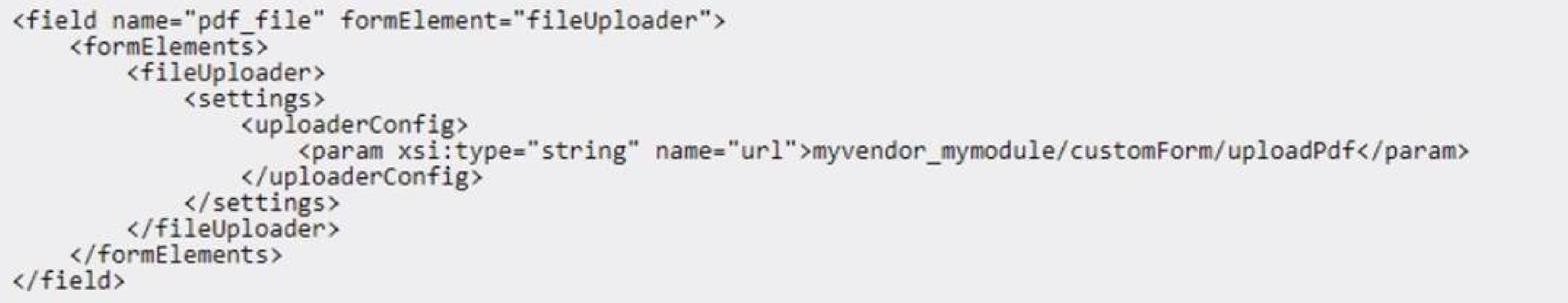
NEW QUESTION 55
How should a record be removed from a database which is using a model that was inherited from the \Magento\Framework\Model\AbstractModel class?

A. Call the "unset" method on this model object
B. Call the "remove" method on this model object
C. Call the "delete" method on this model object

Answer: C

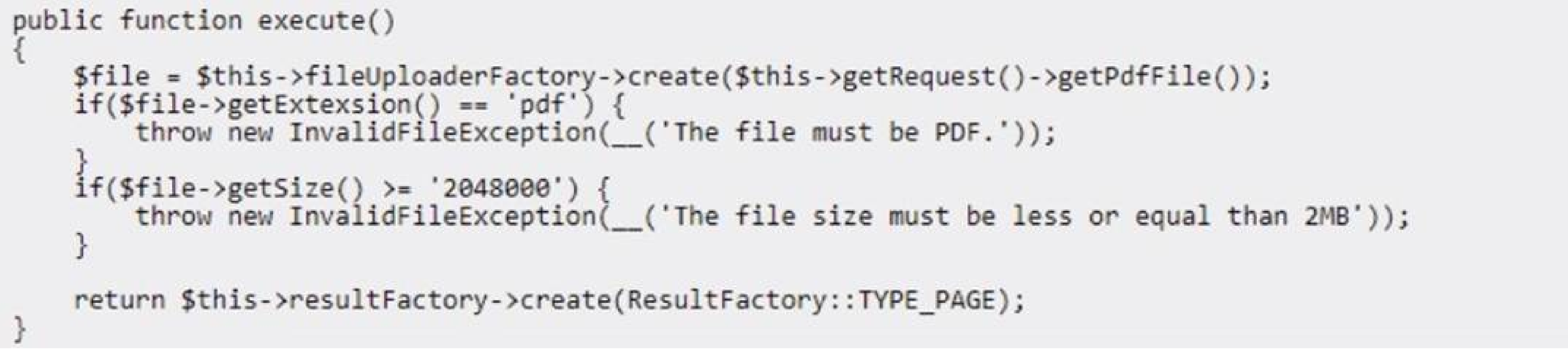
Explanation:
The "delete" method on the\Magento\Framework\Model\AbstractModelclass is used to remove a record from the database. This method will also cascade the delete to any related records.
In Magento, models that inherit from the\Magento\Framework\Model\AbstractModel class interact with the database through the ORM (Object-Relational Mapping) layer. To remove a record from the database using such a model, thedeletemethod is used. This method encapsulates the logic for deleting the record associated with the model instance from its corresponding database table. By calling$model->delete(), where$modelis an instance of a model inheriting fromAbstractModel, Magento will perform the necessary operations to remove the record from the database, ensuring data integrity and consistency within the application.

NEW QUESTION 58
An Adobe Commerce developer is tasked to add a file field to a custom form in the administration panel, the field must accept only .PDF files with size less or equal than 2 MB. So far the developer has added the following code within the form component xml file, inside the fieldset node:



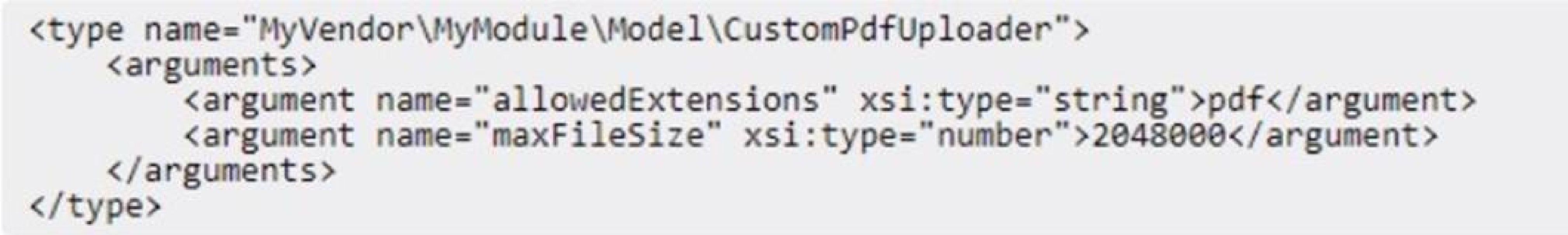
```
<field name="pdf_file" formElement="fileUploader">
    <formElements>
        <fileUploader>
            <settings>
                <uploaderConfig>
                    <param xsi:type="string" name="url">myvendor_mymodule/customForm/uploadPdf</param>
                </uploaderConfig>
            </settings>
        </fileUploader>
    </formElements>
</field>
```

How would the developer implement the validations?
A) Add the Validations Within the HyVendor\MyModule\Controller\Adminhtml\CustomEntity\UploadPdf Controller



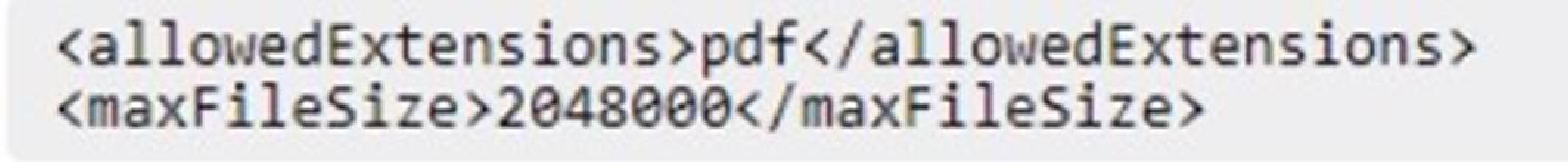
```
public function execute()
{
    $file = $this->fileUploaderFactory->create($this->getRequest()->getPdfFile());
    if($file->getExtexsion() == 'pdf') {
        throw new InvalidFileException(__('The file must be PDF.'));
    }
    if($file->getSize() >= '2048000') {
        throw new InvalidFileException(__('The file size must be less or equal than 2MB'));
    }

    return $this->resultFactory->create(ResultFactory::TYPE_PAGE);
}
```

B) Add a virtual type forMyvendor\MyModuie\Modei\customPdfupioader specifying the aiiowedExtensions and the maxFiiesize for the constructor, within the module's di.xmi:



```
<type name="MyVendor\MyModule\Model\CustomPdfUploader">
    <arguments>
        <argument name="allowedExtensions" xsi:type="string">pdf</argument>
        <argument name="maxFileSize" xsi:type="number">2048000</argument>
    </arguments>
</type>
```

C) Add the following code inside the<settings> node:



```
<allowedExtensions>pdf</allowedExtensions>
<maxFileSize>2048000</maxFileSize>
```

A. Option A
B. Option B
C. Option C

Answer: C

Explanation:
To add file upload validation for a custom form field in the Adobe Commerce admin panel, which should restrict the file type to .pdf and limit the file size to 2 MB, the recommended approach is to include the validation parameters directly within the
<settings> node in the form??sXML configuration. This ensures that the validation occurs on the client-side as well as server-side, providing immediate feedback to users before submission.
Option C is correct for the following reasons:
? Adding Validation Inside <settings>:By placing the <allowedExtensions> and
<maxFileSize> tags within the <settings> node of the XML configuration, the system will enforce these restrictions directly within the form component. This method

leverages Magento??s built-in support for validation settings, which ensures that only files matching the specified criteria (.pdf files of 2 MB or smaller) are accepted by the form.
? uk.co.certification.simulator.questionpool.PList@78475b0d
: Magento??s documentation on UI components highlights how to enforce file type and size restrictions through XML configurations within <settings>, making it the standard and most efficient solution for this task.
Alternatives and Limitations:
Option A: Adding validation within the controller (UploadPdf) can provide additional server- side validation, but this does not prevent the user from selecting invalid files in the first place. Server-side validation alone lacks the user experience enhancement provided by client-side feedback.
Option B: Configuring a virtual type in di.xml for validation (e.g., setting allowedExtensions and maxFileSize within a custom uploader model) is effective for backend processing but is
not as straightforward for direct form validation within the admin UI. It complicates the implementation by requiring custom backend logic where native XML validation can suffice.
Option C provides a straightforward, maintainable, and user-friendly way to implement file validation directly in the form configuration. It reduces the need for custom controller or model logic and leverages Magento??s built-in form handling capabilities.

NEW QUESTION 63
Which two attribute input types does Magento already have by default? (Choose two.)

A. Multiple Select
B. Text Field
C. Geographic coordinate
D. Numeric Field

Answer: AB

Explanation:
The two attribute input types that Adobe Commerce already has by default are Multiple Select and Text Field. Multiple Select allows the user to select multiple values from a list of options. Text Field allows the user to enter text in a single line.
The Geographic coordinate and Numeric Field input types do not exist in Adobe Commerce.
Verified References: [Adobe Commerce User Guide - Create a product attribute] Magento, by default, provides various attribute input types to accommodate different data entry needs for product and customer attributes. The "Text Field" input type allows for
single-line text input, suitable for short, textual data such as names, titles, or any other brief information. The "Multiple Select" input type enables the selection of multiple options from a predefined list, useful for attributes with multiple applicable values like colors, sizes, or features. These input types are part of Magento's flexible attribute system, allowing for customizable data entry fields that cater to diverse product and customer data requirements.

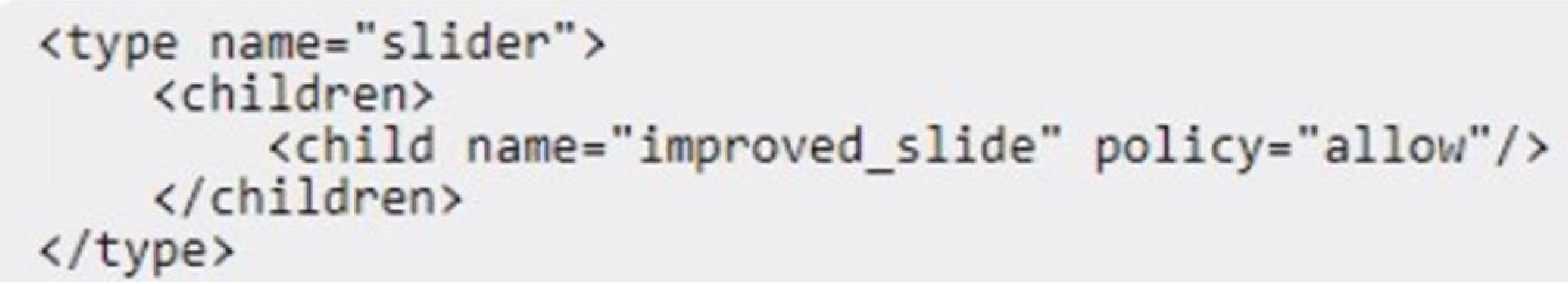
NEW QUESTION 67
A product has been added to the Adobe Commerce Store, and it contains a value for the custom product attribute. A merchant reports that the attribute value is not displayed in the Additional Information tab on the product detail page.
Which action will correct this problem?

A. The attribute must be moved to the specific group in the attribute set
B. The attribute property "Use in Product Tab' must be set to "yes"
C. The attribute property "Visible on Catalog Pages on Storefront" must be set to "yes".

Answer: C

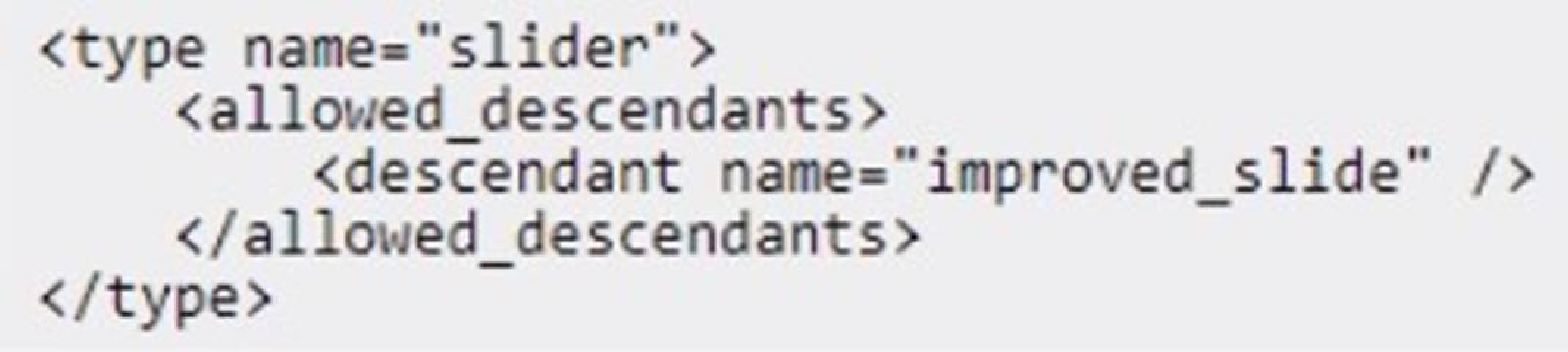
Explanation:
The "Visible on Catalog Pages on Storefront" attribute property determines whether or not the attribute value is displayed in the Additional Information tab on the product detail page. If this property is set to "no", the attribute value will not be displayed.
For a custom product attribute to be displayed in the Additional Information tab on the product detail page in Magento, it needs to be visible on the catalog pages on the storefront. This visibility is controlled by the attribute property "Visible on Catalog Pages on Storefront". When this property is set to "yes", Magento includes the attribute in the Additional Information tab, making it visible to customers browsing the product. This setting ensures that only relevant and intended attributes are shown on the storefront, allowing for better product information management and customer experience.

NEW QUESTION 71
An Adobe Commerce developer has been asked to modify the PageBuilder slider content type to allow a new custom content type (other than slide) to be assigned as a child. The developer has already created the new content type called improved_slide in their module. They now need to create a new view/adminhtml/pagebuilder/content_type/slider. xml file in their module to allow the new content type to be a child of slider content types.
What is the correct xml to accomplish this?
A)



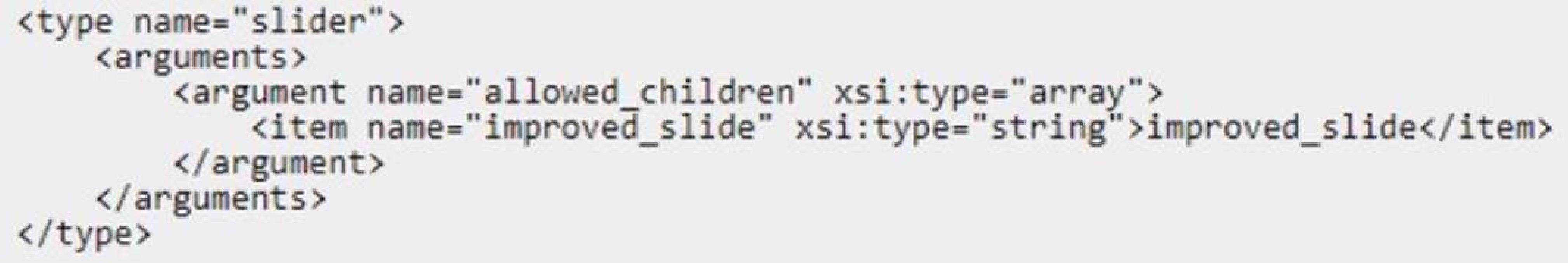
```
<type name="slider">
    <children>
        <child name="improved_slide" policy="allow"/>
    </children>
</type>
```

B)



```
<type name="slider">
    <allowed_descendants>
        <descendant name="improved_slide" />
    </allowed_descendants>
</type>
```

C)



```
<type name="slider">
    <arguments>
        <argument name="allowed_children" xsi:type="array">
            <item name="improved_slide" xsi:type="string">improved_slide</item>
        </argument>
    </arguments>
</type>
```

A. Option A
B. Option B
C. Option C

Answer: C

Explanation:
The correct answer is Option C. This XML configuration is the correct way to define allowed child content types for a slider content type in Magento's PageBuilder.
Magento PageBuilder Content Type Structure:
In PageBuilder, each content type can specify which other content types are allowed as children.
This is done by defining the allowed_children array within the content type??s XML configuration.
Analyzing Option C:
Arguments Definition: Option C uses the <arguments> node to define a new argument named allowed_children.
Array Structure: This argument is an array (xsi:type="array") that includes an item specifying the improved_slide as an allowed child with xsi:type="string".
This configuration is correct because it explicitly defines which child content types are allowed for the slider content type, adhering to Magento??s structure for allowed child
elements in PageBuilder.
Why Options A and B are Incorrect:
Option A: Uses a <children> node with policy="allow", which is not the standard way to define allowed children for PageBuilder content types. This format is incorrect and won??t be recognized by PageBuilder.
Option B: Uses <allowed_descendants>, which also doesn??t align with the way Magento??s PageBuilder expects child content types to be declared. The correct term is allowed_children, not allowed_descendants.
References:
Magento PageBuilder Development Guide - This guide provides insights into customizing PageBuilder and managing content types.
Configuring Content Types in PageBuilder - Documentation on how to define allowed children for custom content types.
Adobe Commerce PageBuilder Content Types XML Reference - Details on the correct XML structure for PageBuilder content types.
Option C??s configuration aligns with Adobe Commerce PageBuilder??s structure for defining which content types can be nested within another, making it the correct choice.

NEW QUESTION 75
An Adobe Commerce Cloud project is using Enhanced Integration Environments with two install a new payment module.
The developer is using Cloud CLI for Commerce tool.
What would a developer do to test this new feature under the integration environment?

A. * 1. Duplicate one of the integration environment branches.* 2. Create a new active branch from integration and install the module.* 3. Push the changes.
B. * 1. Create a new branch from integration and install the module.* 2. Push the changes.* 3. Branch active status check is not necessary.
C. * 1. Deactivate one of the active integration environment branches.* 2. Create a new active branch from integration and install the module.* 3. Push the changes.

Answer: C

Explanation:
The developer can test the new feature under the integration environment by deactivating one of the active integration environment branches, creating a new active branch from integration and installing the module, and pushing the changes. This is because Enhanced Integration Environments have a limit of four active branches at a time, and each branch has its own dedicated database and services. The developer can use the Cloud CLI for Commerce tool to manage the branches and deploy the code changes. Verified References: [Magento 2.4 DevDocs] 1
Enhanced Integration Environments in Adobe Commerce Cloud have a limit on the number of active branches. If both integration branches are currently active, one must be deactivated to create a new active branch for testing.
? Creating a New Active Integration Branch:
? uk.co.certification.simulator.questionpool.PList@400543f4
? Why Option C is Correct:
: Adobe Commerce Cloud documentation onBranch Management

NEW QUESTION 77
A Project Architect needs to add a new developer who needs to be able to push code in an Adobe Commerce Cloud project. No integration with a third-party

repository provider is setup.
What two actions would be required to ensure the developer has access? (Choose Two.)

A. The developer's SSH public key must be added into a file named~/.ssh/authorized_keys
B. The developer needs to add SSH public key in the Cloud Account dashboard settings
C. The developer's email must be added under Users in the Cloud Project Web UI
D. The Adobe Commerce admin user must be created and the developer's SSH public key must be added on their local system

Answer: BC

Explanation:
To ensure the developer has access to push code in an Adobe Commerce Cloud project, the developer??s email must be added under Users in the Cloud Project Web UI and the developer needs to add SSH public key in the Cloud Account dashboard settings. The Cloud Project Web UI is a web interface that allows managing and configuring Adobe Commerce Cloud projects and environments. The developer??s email must be added under Users to grant them access to the project and assign them a role and permissions. The Cloud Account dashboard settings is a web interface that allows managing and configuring Adobe Commerce Cloud accounts and SSH keys. The developer needs to add SSH public key in the settings to enable secure connection to the project and environments via SSH. Verified References: [Magento 2.4 DevDocs]

NEW QUESTION 81

What is one purpose of a customer data JS library?

A. It stores the customers credit card info for usage in the checkout.
B. It stores private customer data in local storage
C. It stores the customer's username and password for easier frontend login.

Answer: B

Explanation:
The customer data JS library is used to store private customer data in local storage. This data can be used to improve the customer's experience on the store, such as by remembering their shipping address or their preferred payment method.
The customer data JS library in Adobe Commerce is used for managing customer data on the client side, such as the shopping cart, comparison list, and wishlist. It does not store sensitive information like credit card details or usernames and passwords. Instead, it utilizes local storage to keep a private data section where customer-specific data is stored securely and accessed via JavaScript, making option B correct.

NEW QUESTION 85

A seller would like to offer an electronic version of an album by selling each song individually. Which layout can be used to customize a product page layout for this item?

A. catalog_product_view_type_downloadable
B. catalog_product_view_type_configurable
C. catalog_product_view_category

Answer: A

Explanation:
Thecatalog_product_view_type_downloadablelayout can be used to customize a product page layout for a downloadable product. This layout includes the product details, the product reviews, and the download links for the product's files.
For selling electronic versions of albums with individual songs, the downloadable product
type in Adobe Commerce is appropriate. To customize the product page layout specifically for downloadable items, the layout
handlecatalog_product_view_type_downloadableis used. This layout handle allows developers to target downloadable products specifically and apply custom layouts or templates, making option A correct.

NEW QUESTION 88

During database migration in the Adobe Commerce Cloud integration environment, a developer experienced a disk space error causing the database import to fail. How would the developer fix this issue?

A. Increase the disk space of the database service.
B. Add a new database node and enable split database.
C. Change the database engine to PostgreSQL that has no disk space limit.

Answer: A

Explanation:
The developer can fix this issue by increasing the disk space of the database service. The database service is one of the services that run on the Adobe Commerce Cloud platform and provide functionality for the application. The database service uses MySQL as the database engine andstores data for products, customers, orders, etc. The disk space of the database service determines how much data can be stored and processed by the database. If the disk space is insufficient, the database import can fail with a disk space error. The developer can increase the disk space of the database service by modifying the .magento/services.yaml file and redeploying the environment. Verified References: [Magento 2.4 DevDocs]

NEW QUESTION 93

Which two actions will the developer need to take to translate strings added in JS files? (Choose two.)

A. define (['jquery, 'mage/translate'), function ($, $t) {.„»;
B. $ trans(,<string>')
C. $.mage._('<string>);
D. translate('<string>');

Answer: AD

Explanation:
To translate strings added in JavaScript files in Adobe Commerce, developers need to use themage/translateRequireJS module along with the $.mage. ('<string>')function to mark strings for translation. This approach ensures that any text strings embedded within JavaScript code can be localized according to the store's current locale, providing a consistent and accessible user experience across different languages and regions. Themage/translatemodule and the$.mage. ()function work together to retrieve the corresponding translated strings from Magento's translation dictionaries, dynamically replacing the original text in the JavaScript code with the appropriate translations.

NEW QUESTION 98
A developer is working on a task that includes a custom controller creation. A controller should forward the request to a different action.
How can the developer complete this task?

A. Specify the forward action in the controllerjorward.xml configuration file.
B. Implement a forwardToAction method in the controller, and return the action where the request should be forwarded.
C. Return the forward object with action as an argument in the object's forward method

Answer: C

Explanation:
To forward the request to a different action, the developer can use the following code in the controller:
return $resultForward->forward(??action??); where $resultForward is an instance of
\Magento\Framework\Controller\Result\ForwardInterface and ??action?? is the name of the action where the request should be forwarded.
There is no controllerjorward.xml configuration file or forwardToAction method in Adobe Commerce.
Verified References: [Adobe Commerce Developer Guide - Forward action result]
In Magento, to forward a request from one controller action to another, a developer can utilize the forward method available in the controller action class. This is achieved by returning a result from the action method that instructs Magento to forward the request to another action. The forward object is obtained by calling the$this->resultForwardFactory-
>create()method within the controller action. Then, the target action is specified by calling theforwardmethod on this object with the action name as the argument, such as
$resultForward->forward('targetAction'). This approach is consistent with Magento's emphasis on using result objects to control the flow of request processing within its MVC architecture.

NEW QUESTION 99
How are multiple EAV attributes belonging to the same entity grouped in the database?

A. Based on the sizes of values they contain
B. Based on all numeric values being stored in one table while text values are stored in the other
C. Based on the types of values they contain

Answer: C

Explanation:
Multiple EAV attributes belonging to the same entity are grouped in the database based on their data types, such as datetime, decimal, int, text, or varchar. For example, all attributes with datetime values are stored in one table, while all attributes with text values are stored in another table.
The sizes or numeric/text values of attributes do not determine how they are grouped in the database.
Verified References: [Adobe Commerce Developer Guide - EAV data model]
Magento's EAV (Entity-Attribute-Value) model organizes attributes based on their data types to optimize storage and retrieval. Attributes are grouped into different tables based on whether they store values of types such as integer, varchar (text), decimal, datetime, etc. This organization allows Magento to efficiently manage the diverse data types associated with products, customers, and other entities, ensuring data integrity and optimizing database performance by using appropriate indexing and storage mechanisms for each data type.

NEW QUESTION 101
When attempting operations that require lengthy processing, a merchant on Adobe Commerce Cloud receives a timeout error after 180 seconds.
How would the developer deal with this issue?

A. 1. Modify admin timeout into .magento.app.yamifile.* 2. Commit and push that code from the local environment.* 3. Move code to Production environment.
B. 1. In the Fastly Configuration section > Advanced Configuration.* 2. Set the Admin path timeout value in seconds.* 3. Save config and Upload VCL to Fastly.
C. 1. Modify admin timeout into app/etc/config.php file.* 2. Commit and push that code from the local environment.* 3. Submit a support ticket to apply the changes.

Answer: B

Explanation:
The developer can deal with this issue by modifying the admin path timeout value in seconds in the Fastly Configuration section > Advanced Configuration in the Admin Panel. Fastly is a cloud-based caching service that improves site performance and security for Adobe Commerce Cloud projects. Fastly has a default timeout value of 180 seconds for admin requests, which means that any request that takes longer than 180 seconds will be terminated and result in a timeout error. The developer can increase this value to allow longer processing time for admin requests without causing errors. The developer also needs to save the configuration and upload VCL to Fastly to apply the changes. Verified References: [Magento 2.4 DevDocs]

NEW QUESTION 106
How would a developer access RabbitMQ data on an Adobe Commerce Cloud Production environment?

A. Using Project Web Interface
B. Using local port forwarding
C. Using RabbitMyAdmin

Answer: B

Explanation:
To access RabbitMQ data on an Adobe Commerce Cloud Production environment, you can use local port forwarding. This allows you to forward a port on your

local machine to a port on the Production environment. This way, you can connect to RabbitMQ from your local machine.
A developer would access RabbitMQ data on an Adobe Commerce Cloud Production environment using local port forwarding. This is done via an SSH tunnel that securely forwards a port from the local machine to the RabbitMQ service on the cloud environment. RabbitMyAdmin (an option that does not exist) and the Project Web Interface do not provide direct access to RabbitMQ data.

NEW QUESTION 110
Which command can be used to display a full list of enabled and disabled Magento modules?

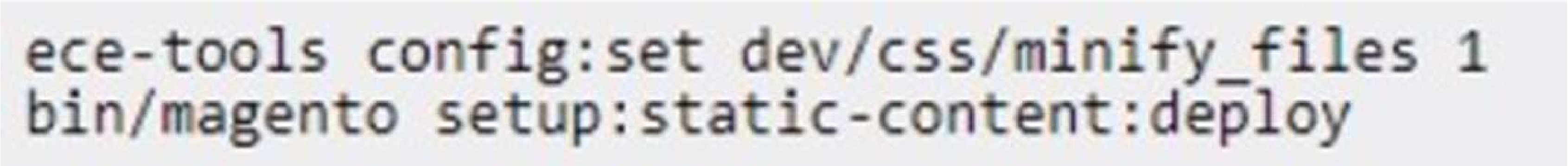
A. bin/magento module:all
B. bin/magento modulestatus
C. bin/magento module:show

Answer: B

Explanation:
The command to display a full list of enabled and disabled Magento modules isbin/magento module:status. This command provides a comprehensive overview of all modules within the Magento instance, categorizing them into enabled and disabled modules. This information is crucial for debugging and development purposes, as it allows developers to quickly understand which components of Magento are active and which are not, facilitating troubleshooting and configuration adjustments.

NEW QUESTION 111
How would a developer enable the magnification of CSS files on an Adobe Commerce Cloud Staging environment?

A. Locally from the command linebin/magento config:set --lock-config dev/css/minify_files 1 Commit the app/etc/config.php file and redeploy.
B. Update the stores > setting > configuration > Advanced > Developer > css configuration in the Admin Panel.
C. SSH to the Adobe Commerce Staging environmen
D. From the command line



```
ece-tools config:set dev/css/minify_files 1
bin/magento setup:static-content:deploy
```

Answer: C

Explanation:
For Adobe Commerce Cloud environments, modifying configuration settings often involves using the ece-tools command-line interface. To enable CSS minification on a staging environment, SSH into the environment and use ece-tools to set the configuration.
? Using ece-tools for Cloud Environments:
? uk.co.certification.simulator.questionpool.PList@79fa8c23
? Why Option C is Correct:
? Steps:
: Adobe Commerce Cloud documentation onUsing ece-tools

NEW QUESTION 116
A message queue currently has queue/consumer-wait-for-messages set to true, which allows the consumer process to run until a message is inserted into the queue. A piece of functionality is driven by data stored in the model
\Magento\variable\Model\variable and this value is only loaded once during the consumer run. If the variable is updated we want the consumer to restart so that the new value is loaded into memory without having to reload the variable on each message consumed.
The Adobe Commerce developer has created an after plugin on the
\Magento\Variable\Model\variable:: save() function.
How would the developer use the plugin to trigger the consumer restart?

A. Call the function\Magento\Framework\MessageQueue\PoisonPill\Poi5onPillPutInterface::put().
B. Call the function\Magento\Framework\MessageQueue\Consumers\TriggerRe5tartInterface:trigger().
C. Set the global Cache key trigger_consumer_restart t0 1.

Answer: A

Explanation:
In Adobe Commerce, when a consumer process needs to restart, the PoisonPillPutInterface can be used. The put() method of this interface triggers a "poison pill," which signals running consumers to restart. This is particularly useful when updated data needs to be reloaded into memory, as in this scenario with the \Magento\Variable\Model\Variable. Poison Pill Mechanism:
The poison pill method tells the message queue consumer to stop its current process and restart, allowing it to pick up any configuration or data changes that occurred since the last start.
Why Option A is Correct:
By calling PoisonPillPutInterface::put(), the consumer will receive a restart signal, which is ideal for cases where data loaded at the beginning of the consumer??s lifecycle must be updated.
Option B is incorrect because TriggerRestartInterface::trigger() does not exist in the Magento framework. Option C is also incorrect as setting a cache key alone does not trigger a consumer restart.
Implementation:
Use an after plugin on the save() method to trigger PoisonPillPutInterface::put() after the variable is saved.
References:
Magento DevDocs on Message Queues and Poison Pill

NEW QUESTION 117
An Adobe Commerce Developer is tasked with creating a custom form which submits its data to a frontend controller They have decided to create an action and have implemented the \Magento\Framework\App\Action\HttpPostActioninterface class, but are not seeing the data being persisted in the database, and an error

message is being shown on the frontend after submission.
After debugging and ensuring that the data persistence logic is correct, what may be cause and solution to this?

A. Magento does not allow POST requests to a frontend controller, therefore, the submission functionality will need to be rewritten as an API endpoint.
B. The developer forgot to implement a validatePostDataQ method in their actio
C. They should implement this method: all non-validated POST datagets stripped out of the request and an error is thrown.
D. Form key validation runs on all non-AJAX POST requests, the developer needs to add the for_key to their requests.

Answer: C

Explanation:
According to the Magento Stack Exchange answer, form key validation is a security feature that prevents CSRF attacks by checking if the form key in the request matches the one generated by Magento. If the developer does not include the form_key in their custom form, the validation will fail and an error will be shown. Therefore, the developer needs to add the form_key to their requests by using <?= $block->getBlockHtml (??formkey??) ?> in their template file. Verified References:https://magento.stackexchange.com/questions/95171/magento-2-form- validation
In Adobe Commerce, when handling POST requests from forms on the frontend, form key validation is enabled by default as a security measure to prevent Cross-Site Request Forgery (CSRF) attacks. This validation checks that the form submission is coming from the same origin by including a unique token (form key) in the request. If this form key is missing or incorrect, the request will fail, and an error message may be shown on the frontend.
In this scenario:
? Since the developer has used \Magento\Framework\App\Action\HttpPostActionInterface, which is appropriate for handling POST requests, it??s likely that the error they encounter is due to missing form key validation.
? The solution is to ensure that the form includes a hidden input field for the form key. Adobe Commerce automatically adds this key in forms if you use the \Magento\Framework\Data\Form\FormKey model to get the form key value. To implement this:
? Ensure the form includes the form key:
<input name="form_key" type="hidden" value="<?= $block->escapeHtml($block->getFormKey()) ?>" />
? The form key should also be included in the POST data sent to the controller. If it??s missing, Adobe Commerce will not process the request.
Additional Resources:
? Adobe Commerce Developer Guide: Form Key
? Magento 2.4 Form Key and CSRF Protection

NEW QUESTION 121
A developer has informed the Adobe Support team about a planned traffic surge on an Adobe Commerce Cloud project that will take place in a little over 48 hours. What is an advantage of this prior notice?

A. When the traffic arrives, extra server resources will be available.
B. The project will temporarily use an upgraded Fastly plan
C. The Support team will monitor the website during that time

Answer: C

Explanation:
Informing the Adobe Support team about a planned traffic surge allows them to monitor the website during that time. With prior notice, the support team can ensure that they are prepared to quickly respond to any issues that arise due to the surge. While extra server resources or an upgraded Fastly plan may be possible outcomes, the primary advantage of advance notice is proactive monitoring and support during expected high traffic events.

NEW QUESTION 122
An Adobe Commerce developer is asked to create a new payment method for their project. This project has administrators who use the backend to manage customer information and occasionally place orders. When testing the new payment method on the frontend everything worked as expected, however, the payment method is missing in the admin.
What is a possible reason for this?

A. In the module di.xml, there were no default 3DS verification types configured as a VirtualType.
B. In the module config.xmi, the boolean value for can_capture was set to false.
C. In the module config.xmi, the node can_use_internal was not set to true.

Answer: C

Explanation:
For a payment method to be available in the admin panel (backend), the configuration must explicitly allow its internal use. This is controlled by the can_use_internal flag in config.xml.
? Configuration for Admin Use:
? uk.co.certification.simulator.questionpool.PList@31da258e
? Why Option C is Correct:
: Adobe Commerce DevDocs onPayment Method Configuration Magento Developer Guide onCustom Payment Method

NEW QUESTION 127
An Adobe Commerce developer is asked to implement a 15% surcharge for all users from a 'Wholesale' customer group. Keeping best practices in mind, what is a correct to accomplish this?

A. Declare a new total collector class to calculate the modified total if the current user is in the group, register it in the module's etc/sales .xml file, modify the checkout_cart_index.xmland checkout_index_index.xml layouts to include a new child in the totals block.
B. Create a Cart Price Rule that applies only to the 'Wholesale' grou
C. Specify no conditions for the rule, and in the Actions section, specify for the rule to apply a "Percent of product price discount", with the 'Discount Amount" field set to -15.
D. Create an Observer to the cataiog_product_get_final_price even
E. Check if the current customer is in the 'Wholesale' group, and if so, retrieve theproduct from the $observer->getEventC) data and Call $product->setData('final_price',$product->getData('final_price') * 1.15).

Answer: A

Explanation:
The best practice to add a surcharge in Magento is to create a custom total collector that calculates and applies the surcharge. This approach integrates smoothly with Magento??s sales and checkout processes.
? Total Collector for Surcharge:
? uk.co.certification.simulator.questionpool.PList@1f664a4e
? Why Option A is Correct:
: Adobe Commerce DevDocs onCustom Total Collectors Magento Sales Documentation onTotal Calculations

NEW QUESTION 128
The value of a product attribute in the Adobe Commerce system needs to have a different format before it is displayed. Which attribute model class is responsible for this?

A. Frontend
B. Backend
C. Source

Answer: A

Explanation:
The frontend attribute model class is responsible for formatting the value of a product attribute before it is displayed on the storefront. This class can implement methods such as getValue, getLabel, getInputType, and getOptionText to modify the attribute value. The backend attribute model class is responsible for saving and loading the attribute value to and from the database. The source attribute model class is responsible for providing the list of options for an attribute.
Verified References: [Adobe Commerce Developer Guide - Attribute models]
In Adobe Commerce (Magento 2), the formatting of product attribute values before they are displayed is handled by the attribute's frontend model. The frontend model, typically extending\Magento\Eav\Model\Entity\Attribute\Frontend\AbstractFrontend, is responsible for preparing the attribute value for display in the frontend, including formatting dates, adding pricing structures, and more. The backend model deals with saving and loading attribute data, while the source model provides options for select attributes.

NEW QUESTION 129
Which method type can be intercepted by plugins?

A. final
B. static
C. public

Answer: C

Explanation:
In Magento, plugins (Interceptors) can only intercept public methods. This is because the plugin system relies on Magento's object manager to dynamically create proxy classes that can intercept method calls. Since private and final methods are not accessible from outside the class they are defined in, and static methods are not called on an object instance, these method types cannot be intercepted. This mechanism allows for the extension and customization of Magento's core behavior in a transparent and non-intrusive manner.

NEW QUESTION 132
A new customer registered on the Integration environment of an Adobe Commerce Cloud project but did not receive a welcome email What would be blocking the email from being sent?

A. SendGrid has not been configured for this environment.
B. On all Integration environments, email is always disabled.
C. The Outgoing Emails setting is disabled into Environment Settings in the Project Web Interface.

Answer: B

Explanation:
In Adobe Commerce Cloud, outgoing emails are disabled by default on Integration environments to prevent test or development emails from being sent to real customers.
? Email Configuration on Integration Environments:
? uk.co.certification.simulator.questionpool.PList@773ee80d
? Why Option B is Correct:
: Adobe Commerce Cloud documentation onEmail Configuration

NEW QUESTION 136
An Adobe Commerce developer has created a module that adds a product attribute to all product types via a Data Patch-According to best practices, how would the developer ensure this product attribute is removed in the event that the module is uninstalled at a later date?

A. Add an Uninstall.php file extending \l1agento\Framework\Setup\UninstallInterface tO the module's Setup directory and implement the uninstall method.
B. Add instructions to the module's README.md file instructing merchants and developers that they must manually remove this attribute if they want to uninstall the module.
C. Make the Data Patch implement\Magento\Framework\setup\Patch\PatchRevertabieinterface and implement the revert method to remove the® product attribute.

Answer: C

Explanation:
According to the Develop data and schema patches guide for Magento 2 developers, data patches can also implement PatchRevertabieinterface to provide rollback functionality for their changes. The revert() method contains the instructions to undo the data modifications made by the patch. To ensure that the product attribute is removed when the module is uninstalled, the developer should make the data patch implement PatchRevertabieinterface and implement the revert

method to remove the product attribute using EavSetupFactory or AttributeRepositoryInterface. Verified References:https://devdocs.magento.com/guides/v2.3/extension-dev-guide/declarative- schema/data-patches.html
According to Adobe Commerce (Magento) best practices, when creating modules that add database schema changes or data through Data Patches, it's crucial to consider the reversibility of these changes for module uninstallation. Here's how each option relates to this practice:
? Option A: Adding an Uninstall.php file that extends
\Magento\Framework\Setup\UninstallInterface is indeed a method to handle module uninstallation in Magento. This interface requires the implementation of an uninstall method where you could write the logic to remove the product attribute. However, this approach is more commonly used for broader setup/teardown operations beyond simple data patches. The official Magento documentation discusses this approach under module uninstallation:
? uk.co.certification.simulator.questionpool.PList@5a27f89e
But for data patches specifically, the recommended approach is different.
? Option B: Adding instructions in the README.md file for manual removal by merchants or developers is not a best practice for module management in Magento. This approach relies on human action which can be error-prone and inconsistent, especially in a production environment. Magento encourages automated processes for module lifecycle management to ensure reliability and consistency.
? Option C: This is the correct and recommended approach according to Magento best practices for data patches. By implementing
\Magento\Framework\Setup\Patch\PatchRevertableInterface in your Data Patch class, you ensure that the patch can be reverted. This interface requires you to implement a revert method, which should contain the logic to remove the changes made by the patch, in this case, the product attribute. Here's how it works:
? uk.co.certification.simulator.questionpool.PList@539e110b
This approach ensures that your module's changes can be automatically undone if the module is uninstalled, maintaining the integrity of the Magento installation. Here's a reference from Magento documentation:
? uk.co.certification.simulator.questionpool.PList@760a9f5e Example implementation:

```
php
use Magento\Framework\Setup\Patch\DataPatchInterface;
use Magento\Framework\Setup\Patch\PatchRevertableInterface; use Magento\Eav\Setup\EavSetup;
use Magento\Eav\Setup\EavSetupFactory;
use Magento\Framework\Setup\ModuleDataSetupInterface;
class AddProductAttribute implements DataPatchInterface, PatchRevertableInterface
{
private $eavSetupFactory; private $moduleDataSetup;
public function construct( EavSetupFactory $eavSetupFactory,
ModuleDataSetupInterface $moduleDataSetup
) {
$this->eavSetupFactory = $eavSetupFactory;
$this->moduleDataSetup = $moduleDataSetup;
}
public function apply()
{
/** @var EavSetup $eavSetup */
$eavSetup = $this->eavSetupFactory->create(['setup' => $this->moduleDataSetup]);
$eavSetup->addAttribute(
\Magento\Catalog\Model\Product::ENTITY, 'custom_attribute',
[
'type' => 'varchar',
'label' => 'Custom Attribute', 'input' => 'text',
'required' => false, 'sort_order' => 100, 'global' =>
\Magento\Eav\Model\Entity\Attribute\ScopedAttributeInterface::SCOPE_GLOBAL, 'group' => 'General',
]
);
}
public function revert()
{
/** @var EavSetup $eavSetup */
$eavSetup = $this->eavSetupFactory->create(['setup' => $this->moduleDataSetup]);
$eavSetup->removeAttribute(\Magento\Catalog\Model\Product::ENTITY, 'custom_attribute');
}
public static function getDependencies()
{
return [];
}
public function getAliases()
{
return [];
}
}
```

This ensures that if the module is uninstalled, the product attribute will be automatically removed, adhering to Magento's modular and reversible development practices.

NEW QUESTION 140
A merchant wants to include taxes In an Adobe Commerce store. Which option can have a tax class assigned to it?

A. Order
B. Shipping
C. Category

Answer: B

Explanation:
According to the Adobe Commerce User Guide, a tax class can be assigned to either a product or a customer group in Adobe Commerce. A product tax class determines how a product is taxed, while a customer tax class determines how a customer is taxed based on their location and group membership. Shipping is considered as a product tax class in Adobe Commerce, and it can be assigned to different shipping methods or rates. The other options are not valid for assigning a tax class.
In Adobe Commerce, tax classes can be assigned to products and shipping. Categories, however, do not have tax classes assigned to them directly. Tax classes applied to shipping allow merchants to specify how taxes should be calculated for shipping costs, making option B the correct answer. Orders and categories do

not have direct associations with tax classes in the same way products and shipping do.

NEW QUESTION 141
In which two directories are third-party modules located by default? (Choose two.)

A. vendor/
B. app/packages/
C. app/modules/
D. app/code/

Answer: AD

Explanation:
By default, third-party modules are located in vendor/ or app/code/ directories. The vendor/ directory contains modules that are installed using Composer, while the app/code/ directory contains modules that are manually copied or cloned from a repository.
Third-party modules in Magento 2 are typically located in two directories by default: the vendor/directory and theapp/code/directory. Thevendor/directory is used for modules that are installed via Composer, Magento's dependency manager, which includes both Magento's core modules and third-party modules. Theapp/code/directory is used for custom modules developed specifically for the project or for third-party modules that are manually installed without Composer. These conventions provide a structured approach to managing Magento modules, whether they are part of Magento's core functionality, contributed by the community, or developed for specific projects.

NEW QUESTION 146
An Adobe Commerce developer has added an iframe and included a JavaScript library from an external domain to the website. After that, they found the following error in the console:
Refused to frame [URL] because it violates the Content Security Policy directive.
In order to fix this error, what would be the correct policy ids to add to the csp_whitelist.xml file?

A. frame-src and script-src
B. default-src and object-src
C. frame-ancestors and connect-src

Answer: A

Explanation:
The Content Security Policy (CSP) in Adobe Commerce (Magento) restricts the types of content that can be loaded on a page to protect against malicious attacks, such as cross-site scripting (XSS). When an iframe is added, and a JavaScript library is loaded from an external source, these resources must be whitelisted explicitly using the csp_whitelist.xml file.
In this specific case:
? The frame-src directive controls the sources from which iframes can be embedded. Since the developer is embedding an iframe from an external domain, they need to whitelist this domain for frame-src.
? The script-src directive controls the sources from which JavaScript files can be loaded. The external JavaScript library must be whitelisted under script-src to allow it to execute.
Therefore, the correct policy IDs to whitelist are:
? frame-src: to allow the embedding of content from an external domain in an iframe.
? script-src: to allow the loading and execution of JavaScript files from the external domain.
Here??s how to update the csp_whitelist.xml file with the correct directives:

```
<?xml version="1.0"?>
<whitelist xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="urn:magento:module:Magento_Csp:etc/csp_whitelist.xsd">
<policy id="frame-src">
<values>
<value id="your-external-domain.com"/>
</values>
</policy>
<policy id="script-src">
<values>
<value id="your-external-domain.com"/>
</values>
</policy>
</whitelist>
```

Replace your-external-domain.com with the actual domain of the external iframe and JavaScript source.
Additional Resources:
? Adobe Commerce Developer Guide: Content Security Policy (CSP)
? CSP Policies and Directives: Explanation of all supported CSP directives and how to configure them.

NEW QUESTION 147
An Adobe Commerce developer has been tasked with applying a pricing adjustment to products on the website. The adjustments come from a database table. In this case, catalog price rules do not work. They created a plugin for getPrice on the price model, but the layered navigation is still displaying the old price.
How can this be resolved?

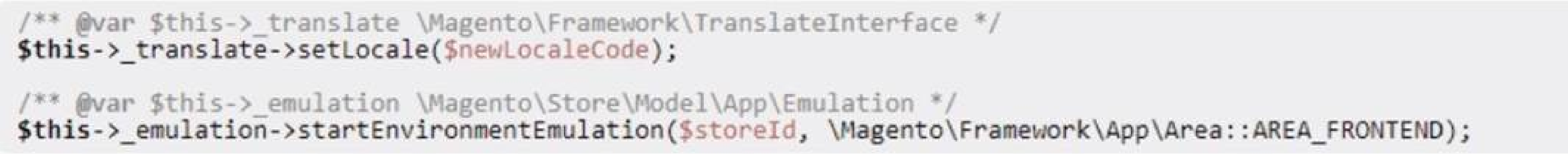
A. Create an implementation for \Magento\Catalog\Hodel\Product\PriceModifierInterf ace.
B. Create an after plugin On \Magento\Catalog\Api\Data\BasePriceInterface:: getPrice.
C. Create a plugin for\Magento\Catalog\Model\Indexer\Product\Price::executeRow.

Answer: C

Explanation:
The developer can resolve this issue by creating a plugin for theMagento\Catalog\Model\Indexer\Product\Price::executeRow()method. This method is responsible for updating the product price index.
The plugin can be used to add the pricing adjustment from the database to the product price index. Once the product price index is updated, the layered navigation

will display the correct price.
Here is an example of a plugin for theexecuteRow()method: PHP
class MyPlugin
{
public function executeRow(
\Magento\Catalog\Model\Indexer\Product\Price $subject,
\Magento\Catalog\Model\Product $product, array $data
) {
$adjustment = $this->getAdjustment($product);
$product->setPrice($product->getPrice() + $adjustment);
}
private function getAdjustment(Product $product)
{
$adjustment = $product->getData('adjustment');
if (!is_numeric($adjustment)) { return 0;
}
return $adjustment;
}
}
This plugin will add theadjustmentdata from the product to the product price index. Once the product price index is updated, the layered navigation will display the correct price.

NEW QUESTION 152
An Adobe Commerce developer is working on a Magento 2 instance which contains a B2C and a B2B website, each of which contains 3 different store views for English, Welsh, and French language users. The developer is tasked with adding a link between the B2C and B2B websites using a generic link template which is used throughout the sites, but wants these links to display in English regardless of the store view.
The developer creates a custom block for use with this template, before rendering sets the translate locale and begins environment emulation using the following code:



```
/** @var $this->_translate \Magento\Framework\TranslateInterface */
$this->_translate->setLocale($newLocaleCode);

/** @var $this->_emulation \Magento\Store\Model\App\Emulation */
$this->_emulation->startEnvironmentEmulation($storeId, \Magento\Framework\App\Area::AREA_FRONTEND);
```

They find that the template text is still being translated into each stores language. Why does this occur?

A. startEnvironmffntEmulation() SetS and locks the locale by Using the setLocale() Optional Second $lock parameter, i.
B. setLocale($newLocaleCode,true), to override and lock the locale of the emulated stor
C. If this is set and locked initially then the environment emulation will not be able to override this.
D. startEnvironmentEmuiation() resets the translation locale to the one of the emulated stores, which overrides the locale the developer has set when the order of setLocate and startEnvironmentEmulation is used as displayed above.
E. setLocate() does not change translation locale after it has been initially set, the $this->_translate->emulate($newLocaiecode) method exists to temporarily modify this by pushing the new locale to the top of the current emuiatedLocales stack.

Answer: B

Explanation:
ThestartEnvironmentEmulation()method resets the translation locale to the one of the emulated stores, which overrides the locale the developer has set when the order ofsetLocale()andstartEnvironmentEmulation()is used as displayed above.
The correct way to achieve the desired result is to use theemulate()method to temporarily modify the translation locale. The following code shows how to do this:
PHP
$this->_translate->emulate('en_US');
// Render the template
$this->_translate->revert();
This code will set the translation locale to English before rendering the template, and then revert the locale back to the default value after the template has been rendered. ThestartEnvironmentEmulation()method is used to emulate a different store view or website. This can be useful for testing purposes, or for developing features that need to work in different environments.
Theemulate()method is used to temporarily modify the translation locale. This can be useful for rendering templates in a specific language, or for testing features that need to work in different languages.

NEW QUESTION 153
Which hashing algorithm will Adobe Commerce choose to hash customer passwords?

A. If the Sodium extension is installed, SHA256 will be chosen, otherwise MD5 will be used as the Magento default hashing algorithm.
B. If the Sodium extension is installed, Argon 2ID13 will be chosen, otherwise SHA256 will be used as the Magento default hashing algorithm.
C. It does not matter if the Sodium extension is installed or not, the Magento hashing default algorithm will be SHA256.

Answer: B

Explanation:
If the Sodium extension is installed, Argon 2ID13 will be chosen as the Magento default hashing algorithm. Otherwise, SHA256 will be used.
The Sodium extension is a PHP extension that provides cryptographic functions. Argon 2ID13 is a password hashing algorithm that is considered to be more secure than SHA256. If the Sodium extension is installed, Magento will use Argon 2ID13 as the default hashing algorithm for customer passwords. If the Sodium extension is not installed, Magento will use SHA256 as the default hashing algorithm.
Adobe Commerce uses secure hashing algorithms for customer passwords. As of the more recent updates, Adobe Commerce defaults to using the Argon2ID13 hashing algorithm, provided that the Sodium PHP extension is available. Argon2ID is considered a secure and modern hashing algorithm designed to protect against brute-force attacks.
If the Sodium extension is not available, Adobe Commerce falls back to using SHA256, which, while secure, is not as robust as Argon2ID13.
This functionality ensures that customer data is safeguarded with the highest level of security available based on the server configuration.
Additional Resources:

? Adobe Commerce Developer Guide: Hashing Algorithms
? PHP Documentation: Argon2 and Sodium

NEW QUESTION 155
Which two recommended practices would a developer use on an Adobe Commerce Cloud Enhanced Integration Environment to get the best performance? (Choose two.)

A. Enable fastly CDN
B. Restrict catalog size
C. Disable cron and manually run as needed
D. Remove all of the integration's inactive branches.

Answer: AD

Explanation:
On an Adobe Commerce Cloud Enhanced Integration Environment, enabling Fastly CDN (Content Delivery Network) can significantly improve performance by caching content closer to the user's location, reducing load times. Additionally, removing all of the integration's inactive branches helps to optimize the environment by decluttering and focusing resources on active development. Restricting catalog size may not be feasible or desirable, and disabling cron jobs can disrupt necessary background operations unless specifically needed for performance testing or troubleshooting.

NEW QUESTION 159
A developer needs to extend the existing jQuery widget. Which jQuery function is used for this purpose?

A. $.mage
B. $.ui
C. $.widget

Answer: C

Explanation:
To extend an existing jQuery widget in Adobe Commerce, the$.widgetfunction is used. This function is part of jQuery UI's widget factory and is a powerful tool for creating stateful plugins with a consistent API. It allows developers to create a new widget that inherits from an existing widget, enhancing or modifying its functionality, making option C the correct answer.

NEW QUESTION 163
Which two methods add sorting to collections inherited from the \Magento\Framework\Model\ResourceModel\Db\Collection\AbstractCollection class? (Choose two.)

A. setOrder
B. setSorting
C. addSorting
D. addOrder

Answer: AD

Explanation:
The two methods that add sorting to collections inherited from the \Magento\Framework\Model\ResourceModel\Db\Collection\AbstractCollection class are setOrder and addOrder. These methods allow adding one or more order clauses to a collection query.
The setSorting and addSorting methods do not exist in Adobe Commerce. Verified References: [Adobe Commerce Developer Guide - Collections]
In Magento 2, collections inherited from \Magento\Framework\Model\ResourceModel\Db\Collection\AbstractCollectionclass can be sorted using thesetOrderandaddOrdermethods. ThesetOrdermethod is used to set the order for a field in the collection, specifying the field by which to sort and the direction of the sorting (ASC or DESC). TheaddOrdermethod is similar but allows for adding multiple sorting orders to the collection, enabling more complex sorting scenarios. There are nosetSortingoraddSortingmethods in the standard Magento 2 collection classes.

NEW QUESTION 168
An integration named Marketing is created on the Adobe Commerce instance. The integration has access on Magento_Customer:: customer resources and the access token is xxxxxx.
How would the rest API be called to search the customers?

A. Using the integration access token as Bearer:curl -X GET https://magentourl/rest/Vl/customers/search?searchCriteria... -H 'Authorization: Bearer XXXXXX'
B. Passing integration name and access token as http auth credentials: curl -X GET https://Marketing:XXXXXX(Slmagentourl/rest/Vl/customers/search?5earchCriteria . . .Using integration name as username and access token as password, get the admin token (yyyyyy) via:curl -X POST https://magentourl/rest/Vl/integration/admin/token -d '{"username":"Marketing", "password":"XXXXXX"}' -H 'Content-
C. Type: application/json'Use the admin token as Bearercurl -X GET https://magentourl/rest/Vl/customers/search?searchCriteria... -H 'Authorization: Bearer YYYYYY'

Answer: A

Explanation:
When using an integration token to access Magento??s REST API, you can authenticate requests by including the token in the Authorization header as a Bearer token. This allows the system to recognize the permissions assigned to the integration and grant access to the specified resources.
? Using the Access Token as Bearer Token:
? uk.co.certification.simulator.questionpool.PList@69ee4bb7
? Why Option A is Correct:
? Example Command: curl -X GET "https://magentourl/rest/V1/customers/search?searchCriteria[filterGroups][0][filters][0][field] =email&searchCriteria[filterGroups][0][filters][0][value]=example@example.com" -H "Authorization: Bearer XXXXXX"

References:
Adobe Commerce REST API documentation on Authentication Magento Integration Tokens Guide on Using Tokens

NEW QUESTION 173
Which attribute option restricts Catalog EAV attributes to only certain product types?

A. show.in
B. apply_to
C. allowed_in

Answer: B

Explanation:
Theapply_toattribute option in Magento's Catalog EAV model restricts the use of certain attributes to specific product types. By specifying product types in the apply_tofield, developers can control which attributes are applicable to which types of products, ensuring that attributes are only available where they are relevant and meaningful.

NEW QUESTION 178
What is an advantage of the read-only core file system using Adobe Commerce Cloud?

A. Ensures that all changes to the production environment are tracked.
B. Improves website performance.
C. Reduces the number of attackable surfaces significantly

Answer: A

Explanation:
The read-only core file system on Adobe Commerce Cloud ensures that all changes to the production environment are tracked. This is because any changes to the code must go through version control, and the deployment pipeline, which includes stages like build, staging, and production. This approach helps maintain consistency across environments, ensures deployment best practices, and reduces human error by preventing direct changes on production servers.

NEW QUESTION 180
A developer wants to deploy a new release to the Adobe Commerce Cloud Staging environment, but first they need the latest code from Production.
What would the developer do to update the Staging environment?

A. 1. Log in to the Project Web Interface.* 2. Choose the Staging environment, and click Merge
B. 1. Checkout to Production environment* 2. Use the magento-cloud synchronize <environment-ID> Commerce CLI Command
C. 1, Log in to the Project Web Interface.* 2. Choose the Staging environment, and click Sync

Answer: C

Explanation:
To update the Staging environment with the latest code from the Production
environment on an Adobe Commerce Cloud project, the developer would log in to the Project Web Interface, choose the Staging environment, and then click Sync.
This action synchronizes the environments, bringing the latest changes from Production into Staging.

NEW QUESTION 182
What does a URL Rewrite do?

A. It updates the URL that is stored on the server.
B. It changes the way a URL appears in the browser
C. It updates the URL to a domain that is not being Indexed.

Answer: B

Explanation:
A URL Rewrite in Magento changes the way a URL appears in the browser. This is particularly useful for improving the readability and SEO of a URL. For example, a URL rewrite can be used to transform a long and complex URL into a shorter and more user-friendly version. It's important to note that while a URL rewrite changes the URL's appearance in the browser, it doesn't change the actual location of the resource on the server. This distinction is crucial for understanding how Magento handles URL rewrites and redirects, facilitating a more intuitive navigation structure within the store without altering the underlying server resources.

NEW QUESTION 187
......