Exam Questions AD0-E724

Adobe Commerce Developer Professional

https://www.2passeasy.com/dumps/AD0-E724/

NEW QUESTION 1
Which file is used to add a custom router class to the list of routers?

A. routes.xml
B. di.xml
C. config.xml

Answer: A

Explanation:
To add a custom router class to the list of routers in Magento, theroutes.xml file is used. This file should be located in theetcdirectory of the module, under the appropriate area (eitherfrontendoradminhtml). Within theroutes.xmlfile, you define a router with an ID, a route with an ID and frontName, and specify the module that the route corresponds to. This setup allows Magento to recognize and utilize the custom router when processing URLs, directing requests to the appropriate controllers based on the custom routing logic defined.

NEW QUESTION 2
An Adobe Commerce developer is developing a custom module. As part of their implementation they have decided that all instances of their Custom\Module\Model\Example class should receive a new instance of Magento\Filesystem\Adapter\Local.
How would the developer achieve this using di. xml?
A)

```
<type name="Custom\Module\Model\Example">
    <arguments>
        <argument name="adapter" xsi:type="object" shared="false">Magento\Filesystem\Adapter\Local</argument>
    </arguments>
</type>
```

B)

```
<type name="Custom\Module\Model\Example">
    <arguments>
        <argument name="adapter" xsi:type="object" singleton="false">Magento\Filesystem\Adapter\Local</argument>
    </arguments>
</type>
```

C)

```
<type name="Custom\Module\Model\Example">
    <arguments>
        <argument name="adapter" xsi:type="object" transient="true">Magento\Filesystem\Adapter\Local</argument>
    </arguments>
</type>
```

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

Answer: C

NEW QUESTION 3
A developer would like to initialize a theme in Adobe Commerce. Which two files are required to complete this task? (Choose two.)

A. theme.less
B. registration.php
C. composerjson
D. themexml

Answer: BC

Explanation:
To initialize a theme in Adobe Commerce, at least two files are required: registration.phpandtheme.xml. Theregistration.phpfile is used to register the theme within the system, andtheme.xmldefines the theme's name, parent (if any), and other metadata. Thetheme.lessfile is not required for theme initialization but may be used for custom styling. The correct option fortheme.xmlis represented as "theme.xml" (D), not "themexml" as mentioned in the options.

NEW QUESTION 4
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 5
There is an integration developed using a cron service that runs twice a day, sending the Order ID to the integrated ERP system if there are orders that are able to create an invoice. The order is already loaded with the following code:
$order = $this->orderRepository->get($orderId);
In order to verify if the store has invoices to be created, what implementation would the Adobe Commerce developer use?
A)

```
if ($order->canInvoice()) {
    // send integration to the ERP
}
```

B)

```
if ($order->hasInvoice()) {
    // send integration to the ERP
}
```

C)

```
if (!$order->isPaymentReview()) {
    // send integration to the ERP
}
```

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

Answer: A

Explanation:
The correct implementation to check if an order is eligible for invoicing is to use the $order->canInvoice() method. This method checks whether the order meets all necessary conditions for an invoice to be created, such as the order not being fully invoiced or canceled.
Option A is correct for the following reasons:
? Using canInvoice() for Invoicing Eligibility:The $order->canInvoice() method is specifically designed to verify if an order can have an invoice generated. It returns true only if the order is in a state where it can be invoiced. This makes it the appropriate method for determining whether the order should be sent to the ERP system for invoicing.
? uk.co.certification.simulator.questionpool.PList@45e8e59a
: Magento??s developer documentation on the Order model highlights canInvoice() as the recommended approach for determining invoice eligibility, particularly when automating processes like ERP integration.
Alternatives and Limitations:
Option B: The $order->hasInvoice() method only checks if there is already an invoice associated with the order, which does not indicate whether the order is eligible for new invoicing. It returns true if any invoice exists for the order, which is not suitable for this scenario.
Option C: The $order->isPaymentReview() method checks if the order is in a payment review state, which is not directly related to invoice creation eligibility. It would not provide accurate information on whether the order can be invoiced.
By using canInvoice(), the developer ensures that the cron job will only send orders that are ready for invoicing to the ERP system, adhering to Adobe Commerce??s order processing logic.

NEW QUESTION 6

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):

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 7
Under which section should the soft dependency for a module be listed in app/code/<Vendor>/<Module>/composer.json file?

A. suggest*: {
B. }optional": {
C. }soft": {
D. }

Answer: A

Explanation:
Soft dependencies for a module should be listed under the "suggest" section in thecomposer.jsonfile of the module. This section is used to list packages that enhance or work well with the module but are not strictly required for the module to function. By using the "suggest" section, developers can inform others about optional packages that could improve functionality or integration with the module, without making them mandatory dependencies.

NEW QUESTION 8
For security reasons, merchant requested to a developer to change default admin url to a
unique url for every branch/environment of their Adobe Commerce Cloud project. Which CLI command would the developer use update the admin url?

A. ece-tools variable:update ADMIN_URL
B. magento-cloud variable:set ADMIN_URL
C. bin/magento adminuri:set <admin_uri>

Answer: B

Explanation:
The CLI command that the developer would use to update the admin url is magento-cloud variable:set ADMIN_URL. This command sets an environment variable called ADMIN_URL with a custom value for the admin url on a specific environment. Environment variables are configuration settings that affect the behavior of the Adobe Commerce Cloud application and services. By setting an environment variable for ADMIN_URL, the developer can change the default admin urlto a unique url for every branch/environment of their Adobe Commerce Cloud project. Verified References: [Magento 2.4 DevDocs]

NEW QUESTION 9
What is one way a developer can upgrade the ECE-Tools package on an Adobe Commerce Cloud project?

A. Cloud CLI for Commerce tool
B. Project Web Interface
C. Composer

Answer: C

Explanation:
According to the Adobe Commerce Developer Documentation, one way a developer can upgrade the ECE-Tools package on an Adobe Commerce Cloud project is by using Composer, which is a dependency management tool for PHP projects. The ECE- Tools package contains scripts and tools that help manage and deploy Adobe Commerce Cloud projects on the cloud infrastructure. To upgrade the ECE-Tools package using Composer, the developer needs to run the

following command in the project root directory: composer update magento/ece-tools --with-dependencies
On an Adobe Commerce Cloud project, the recommended way to upgrade the ECE-Tools package is through Composer, a dependency manager for PHP. Composer is used to manage the dependencies of the project, including ECE-Tools, and it provides the necessary commands to update packages to their latest versions or to specific versions as required.

NEW QUESTION 10
Which log file would help a developer to investigate 503 errors caused by traffic or insufficient server resources on an Adobe Commerce Cloud project?

A. mysql-slow.log
B. access.log
C. cloud.log

Answer: B

Explanation:
The access.log file stores the information about the requests and responses that occur on the web server, such as the IP address, timestamp, request URI, response code, andreferer URL1. By checking the access.log file, a developer can identify the requests that resulted in 503 errors and the possible causes of those errors, such as traffic spikes, insufficient server resources, or misconfiguration1.
To check the access.log file on an Adobe Commerce Cloud project, a developer can use the following command in the CLI:
grep -r "\" 50 [0-9]" /path/to/access.log
This command will search for any lines in the access.log file that contain a response code starting with 50, which indicates a server error1. The developer can then compare the timestamps of those lines with the exception.log and error.log files to find more details about the errors1.
Alternatively, if the error has occurred in the past and the access.log file has been rotated (compressed and archived), the developer can use the following command in the CLI (Pro architecture only):
zgrep "\" 50 [0-9]" /path/to/access.log.<rotation ID>.gz
This command will search for any lines in the compressed access.log file that contain a response code starting with 501. The rotation ID is a number that indicates how many
times the log file has been rotated. For example, access.log.1.gz is the most recent rotated log file, and access.log.10.gz is the oldest rotated log file1.

NEW QUESTION 10
Which command should be used to refresh the cache using the command line?

A. magentocacheclean<type>
B. magentocacherefresh<type>
C. magentocachedelete<type>

Answer: A

Explanation:
To refresh the cache using the command line in Magento, the command bin/magento cache:clean <type>is used. This command clears the cache for the specified type, such as configuration, layout, or block HTML. Cleaning the cache is an essential maintenance task that helps to update the store's frontend with the latest changes and ensure that customers see the most current content. This command is particularly useful during development or after making changes to the store's configuration or layout.

NEW QUESTION 15
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 16
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 17
An Adobe Commerce developer is tasked with adding custom data to orders fetched from the API. While keeping best practices in mind, how would the developer achieve this?

A. Create an extension attribute on Nagento\sales\Api\E)ata\orderinterface and an after plugin onMagento\Sales\Model\Order: :getExtensionAttributes() to add the custom data.
B. Create an extension attribute On Magento\Sales\Api\Data\OrderInterface and an after plugin On Magento\Sales\Api\OrderRepositoryInterface On geto and getListo to add the custom data.
C. Create a before plugin on Magento\sales\model\ResourceModel\order\collection: :load and alter the query to fetch the additional dat
D. Data will then be automatically added to the items fetched from the API.

Answer: B

Explanation:
The developer should create an extension attribute on theMagento\Sales\Api\Data\OrderInterfaceinterface and an after plugin on theMagento\Sales\Api\OrderRepositoryInterface::get()andMagento\Sales\Api\OrderReposit oryInterface::getList()methods.
The extension attribute will store the custom data. The after plugin will be used to add the custom data to the order object when it is fetched from the API.
Here is the code for the extension attribute and after plugin: PHP

```
namespace MyVendor\MyModule\Api\Data;
interface OrderExtensionInterface extends \Magento\Sales\Api\Data\OrderInterface
{
/**
* Get custom data.
*
* @return string|null
*/
public function getCustomData();
/**
* Set custom data.
*
* @param string $customData
* @return $this
```

```
*/
public function setCustomData($customData);
}
namespace MyVendor\MyModule\Model;
class OrderRepository extends \Magento\Sales\Api\OrderRepositoryInterface
{
/**
* After get order.
*
* @param \Magento\Sales\Api\OrderRepositoryInterface $subject
* @param \Magento\Sales\Api\Data\OrderInterface $order
* @return \Magento\Sales\Api\Data\OrderInterface
*/
public function afterGetOrder($subject, $order)
{
if ($order instanceof OrderExtensionInterface) {
$order->setCustomData('This is custom data');
}
return $order;
}
/**
* After get list.
*
* @param \Magento\Sales\Api\OrderRepositoryInterface $subject
* @param \Magento\Sales\Api\Data\OrderInterface[] $orders
* @return \Magento\Sales\Api\Data\OrderInterface[]
*/
public function afterGetList($subject, $orders)
{
foreach ($orders as $order) {
if ($order instanceof OrderExtensionInterface) {
$order->setCustomData('This is custom data');
}
}
return $orders;
}
}
```

Once the extension attribute and after plugin are created, the custom data will be added to orders fetched from the API.

NEW QUESTION 19
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 22
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 25
An Adobe Commerce developer successfully added a new column to the customers grid. This column needs the data to be formatted before showing its content in the grid.
According to best practices, how would the developer add the custom logic to render the column?

A. 1. Create an after pluginforMagento\Ui\Component\Listing\Columns\Column::prepareColumn().* 2. Add the custom logic within the afterPreparecoiumn method.
B. 1. Create a custom class extending flagento\Ui\Component\Listing\Columns\Colunm.* 2. Add the custom logic within the prepareDataSource method.* 3. Add an attribute class to the column node within the module's customer_listing.xml.
C. 1. Override the Magento\Customer\Ui\Component\DataProvider Class using a preference.* 2. Override the getData() method and add the custom logic per row.

Answer: B

Explanation:
The best practice to add custom logic for data formatting in a grid column is to create a new class extending \Magento\Ui\Component\Listing\Columns\Column. The prepareDataSource method is designed for processing and formatting data before it is displayed in the UI component.
? Using prepareDataSource in a Custom Column Class:
? uk.co.certification.simulator.questionpool.PList@3b879431
? Why Option B is Correct:
? Implementation Steps:
: Magento UI Components Guide onCreating Custom Columns
Adobe Commerce documentation onMagento\Ui\Component\Listing\Columns\Column

NEW QUESTION 30
What database engine is part of the infrastructure of Adobe Commerce Cloud projects?

A. Percona
B. MariaDB
C. MySQL

Answer: B

Explanation:
The database engine that is part of the infrastructure of Adobe Commerce Cloud projects is MariaDB. MariaDB is a fork of MySQL that offers improved performance, scalability, and security features.
The database engine that is part of the infrastructure of Adobe Commerce Cloud projects is MariaDB. Adobe Commerce Cloud is configured to use MariaDB, which is a binary drop-in replacement for MySQL and is chosen for its performance, reliability, and feature set.

NEW QUESTION 32
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 37
Which CLI command should be used to determine that static content signing is enabled?

A. bin/magento config:show dev/static/status
B. bin/magento config:show dev/static/sign
C. bin/magento config:show dev/static/sign/status

Answer: B

Explanation:
After a thorough search of the provided documents, I couldn't find a direct reference to the specific CLI command for determining if static content signing is enabled in Magento. However, the typical command for checking configuration settings in Magento is bin/magento config:show <path>, where<path>is the configuration path for the setting you wish to view. Based on Magento's configuration path patterns and the options provided, the most logical choice would beB. bin/magento config:show dev/static/sign, although this cannot be confirmed without further context or documentation.

NEW QUESTION 39
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 43
Which characteristic is associated with a persistent cart?

A. By default, a persistent cookie will become inactive in 30 days.
B. While using the persistent cart, guest users do not need to log in or register to checkout
C. While the customer is logged in, If the session cookie expires, the persistent cookie will remain active

Answer: C

Explanation:
A persistent cart is a cookie that is stored on the customer's computer. This cookie allows the customer to continue shopping even if they close their browser. If the customer is logged in, the persistent cookie will remain active even if the session cookie expires.
Associated with a persistent cart in Adobe Commerce is the characteristic that while the customer is logged in, if the session cookie expires, the persistent cookie will remain active. This ensures that the customer's shopping cart is preserved even if they have been inactive and the session has expired. The persistent cookie allows the cart to be restored when the customer returns to the store.

NEW QUESTION 47
Which action, if any, should be taken to enable filtering by attribute in the category's layered navigation?

A. Set the category's "Anchor' display setting to "yes".
B. Select "With layered navigation" from the category's display mode
C. Filtering by the attribute is enabled for every category automatically.

Answer: A

Explanation:
To enable filtering by attribute in a category's layered navigation, you should set the category's "Is Anchor" setting to "Yes". This setting is found in the category's display settings within the admin panel. When a category is set as "Anchor", Magento will automatically include filtering options in the layered navigation block for all attributes that
are configured to be used in layered navigation.

NEW QUESTION 51
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 54
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 56
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 57
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 62
Which file on an Adobe Commerce Cloud project allows a developer to upgrade the PHP version and enable/disable a PHP extension?

A. magento.app.yaal
B. .magent
C. en
D. yaml
E. php.ini

Answer: B

Explanation:
The.magento.env.yamlfile is used on an Adobe Commerce Cloud project to customize the environment configuration, including the PHP version and enabling/disabling PHP extensions. This YAML configuration file provides the ability to manage service configurations and is essential for customizing the Cloud environment.

NEW QUESTION 66
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 71
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 74
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 76
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 80
An Adobe Commerce developer is tasked with creating a custom block that will be displayed on every page in the footer of the site.
After completing and optimizing the development, the developer notices that the block takes too much time to be generated on each page and decides to store it in the system cache after enabling it for all cache groups.
What would be the minimum requirement to achieve this?

A. Set a value for the cache_Lifetime data property of the block.
B. Set a value for cache_key data property of the block.
C. Set values for both cache_lifetime and cache_key data properties of the block.

Answer: A

NEW QUESTION 82
An Adobe Commerce Cloud developer wants to be sure that, even after transferring database from Production to Staging, the payment configurations are still valid on the Staging environment.
What does the developer need to add to be sure that the configurations are always properly set?

A. Lines in the dedicated core_conf ig_data_stg table.
B. Project level environment variables.
C. Environment level environment variables.

Answer: C

Explanation:
The developer needs to add environment level environment variables to be sure that the payment configurations are always properly set on the Staging environment. Environment variables are configuration settings that affect the behavior of the Adobe Commerce Cloud application and services. Environment variables can be set at the project level or the environment level. Project level variables apply to all environments, while environment level variables override the project level variables for a specific environment. The developer can use environment level variables to customize the payment configurations for the Staging environment without affecting other environments. Verified References: [Magento 2.4 DevDocs]

NEW QUESTION 87
Which property allows multiple cron jobs to share the same configuration?

A. name
B. group
C. schedule

Answer: B

Explanation:
In Magento, thegroupelement in the cron job configuration allows multiple cron jobs to share the same configuration settings, such as schedule, status, and execution time limits. This grouping facilitates the management of cron jobs, making it easier to configure and maintain them, especially when multiple tasks require similar or identical settings. By assigning cron jobs to a specific group, they inherit the group's configuration, streamlining the setup process and ensuring consistent execution parameters across
related tasks.

NEW QUESTION 90
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 91
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 92
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 93
How should a grid or form be included in an admin page layout using the Ul Component?

A. <referenceContainername='content"> q <uiComponentname="example_listing.xml7></referenceContainer>
B. <referenceContainername='contenf> q <uiComponent name="example_listing7></referenceContainer>
C. <referenceContainername='content"><uiComponentname="Vendor_Module::ul_component/example_listing.xml7></referenceContainer>

Answer: B

Explanation:
To include a grid or form in an admin page layout using the UI Component, the correct approach is to use the<uiComponent name="example_listing"/>within a <referenceContainer name='content'>block of the layout XML file. This method directly references the UI component's configuration file (e.g.,example_listing.xml) which defines the structure andfunctionality of the UI component, such as grids or forms. This configuration file is located under theview/adminhtml/ui_componentdirectory of the corresponding module.

NEW QUESTION 95
What will happen if a developer fails to mention the start date in the "From" field when creating a price rule?

A. The price rule will not be saved.

B. The price rule will go into effect as soon as it is saved
C. The price rule will be saved, but it will not go into effect until the start date is added

Answer: B

Explanation:
If a developer fails to mention the start date in the "From" field when creating a price rule, the price rule will be saved. However, the price rule will not go into effect until the start date is added.
If a developer fails to mention the start date in the "From" field when creating a price rule in Adobe Commerce, the system will default to the current date, and the price rule will go into effect as soon as it is saved. The absence of a start date means there is no delay in the activation of the rule, and it becomes effective immediately upon saving.

NEW QUESTION 97
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 99
On an Adobe Commerce Cloud platform, what type of environment will be provisioned when launching the CLI for Commerce command magento-cloud environment:branch
<environment-name> <parent-environment-id>?

A. An empty integration environment without any code or database.
B. An integration environment with fresh Adobe Commerce Cloud installation.
C. An integration environment with the code and database from the parent environment.

Answer: C

Explanation:
The type of environment that will be provisioned when launching the CLI for Commerce command magento-cloud environment:branch <environment-name> <parent- environment-id> is an integration environment with the code and database from the parent environment. Integration environments are temporary environments that are used for testing and development purposes on the Adobe Commerce Cloud platform. They can be created from any branch of code and have their own dedicated database and services. When creating an integration environment using the CLI for Commerce command, the code and database from the parent environment are copied to the new integration environment, creating an exact replica of the parent environment. Verified References: [Magento 2.4 DevDocs]

NEW QUESTION 101
What are the only writeable folders in the application root on a remote Adobe Commerce Cloud project?
A)

B)

C)

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

Answer: B

Explanation:
For an Adobe Commerce Cloud project, the only writeable folders in the application root on a remote environment are essential for the application to run correctly and store temporary and dynamic data. Among the options given, Option B lists directories that are typically writable:m2-hotfixes,var,pub/static, andapp/etc. Them2-hotfixes directory is specifically for Magento Commerce Cloud and is used for applying hotfixes that are executed during the build phase. Thevardirectory

contains various logs, sessions, and reports. Thepub/staticdirectory holds the compiled static view files, andapp/etccontains configuration files that can be modified by the application at runtime.

NEW QUESTION 103
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 106
An Adobe Commerce developer added a new API method to search and retrieve a list of Posts for a custom Blog functionality. This is the content of the module's etc/webapi.xml file:

```
<?xml version="1.0" ?>
<routes xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="urn:magento:module:Magento_Webapi:etc/webapi.xsd">
    <route url="/V1/myvendor-blog/post/search" method="GET">
        <service class="MyVendor\Blog\Api\PostRepositoryInterface" method="getList"/>
        <resources>
            <resource ref="MyVendor_Blog::Post_view"/>
        </resources>
    </route>
</routes>
```

The new code has been deployed to production and the merchant is using https:
//merchant. domain. com/swagger to review the new endpoint, but it is not visible in swagger.
What would be a reason for this?

A. The webapi.xml file should be moved into the etc/webapi_rest/webapi.xml file.
B. Since the new endpoint is not anonymous, the merchant needs to enter a validintegration token in swagger in order to see the new method.
C. The greturn annotation is missing in the MyVendor\Blog\Api\PostRepositoryInterf ace class.

Answer: B

Explanation:
In Adobe Commerce, when defining new API endpoints through the webapi.xml configuration file, the visibility of these endpoints in tools likeSwagger(now OpenAPI) depends on several factors, including authentication settings. According to the provided webapi.xml file:
<?xml version="1.0"?>
<routes xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="urn:magento:module:Magento_Webapi:etc/webapi.xs d">
<route url="/V1/myvendor-blog/post/search" method="GET">
<service class="MyVendor\Blog\Api\PostRepositoryInterface" method="getList"/>
<resources>
<resource ref="MyVendor_Blog::Post_View"/>
</resources>
</route>
</routes>
? Option Asuggests moving the file to etc/webapi_rest/webapi.xml. However, this is incorrect because Adobe Commerce does not require separate XML files for REST and SOAP APIs in this context. The webapi.xml is used for defining routes for both. The structure and location provided in the question are correct for defining REST API routes.
? Option Bis the correct answer. The resource reference
MyVendor_Blog::Post_View indicates that this API endpoint is not anonymous; it requires authentication. In Adobe Commerce, if an API endpoint requires authentication, it won't be visible in Swagger (or the OpenAPI UI) unless you provide valid authentication credentials or tokens. This is part of Magento's security model where protected resources require tokens or OAuth to access. For the merchant to see this endpoint in Swagger, they must provide an integration token or OAuth token which has permissions for MyVendor_Blog::Post_View. This is detailed in the Adobe Commerce Developer Documentation under[Web API Authentication](https://x.com/i/grok?text=Web%20API%20Authentication).
? Option Cmentions the @return annotation missing in the interface class. While
proper annotations in PHPDoc are important for IDE autocompletion and documentation generation, they are not directly related to the visibility of an endpoint in Swagger. The visibility in Swagger is determined by the configuration
in webapi.xml and the authentication settings, not by PHPDoc annotations. Thus, this option is incorrect in the context of the question.
For further reading on how to manage and configure API endpoints in Adobe Commerce, including authentication, refer to the official Adobe Commerce Developer Guide:
? Web API Configuration
? Web API Authentication
? Swagger Integrationfor testing and documenting APIs.

This explanation covers the necessary aspects of Adobe Commerce API development, focusing on the configuration, authentication requirements, and how these affect the visibility of API endpoints in development tools like Swagger.

NEW QUESTION 107
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 110
An Adobe Commerce developer has created a new shipping carrier Everything has been implemented and the collectRates() and getAllowedMethodsQ functions can be seen below:

```
public function collectRates(RateRequest $request) {
    if (!$this->getConfigFlag('active')) {
        return false;
    }

    $result = $this->rateResultFactory->create();
    $method = $this->rateMethodFactory->create();

    $method->setCarrier($this->_code);
    $method->setCarrierTitle($this->getConfigData('title'));
    $method->setMethod($this->_code);
    $method->setMethodTitle($this->getConfigData('name'));

    $method->setPrice(0);
    $method->setCost(10);

    $result->append($method);

    return $result;
}
```

```
public function getAllowedMethods() {
    return [$this->_code => $this->getConfigData('name')];
}
```

Given the above code, what would be the displayed cost of the shipping method and final amount charged to the customer?

A. The shipping method would display SO but customers would pay a $10 handling fee for their order.
B. The shipping method would display $0 and customers would pay $0 for using the new shipping method.
C. The shipping method would display $10 and customers would pay $10 for using the new shipping method.

Answer: B

NEW QUESTION 115
How can a custom CMS Page be set as a store home page?

A. In the CMS Page admin grid, select the checkbox for the page under the "Home Page' column.
B. In the CMS Page admin form, set the 'Default Home Page" value to "yes"
C. In the store configuration, set a custom CMS Page to be a CMS home page

Answer: C

Explanation:
To set a custom CMS Page as a store home page, the developer or merchant should follow these steps:
? In the Admin panel, go to Content > Pages and create or edit a CMS Page that will
be used as a home page.
? In the Admin panel, go to Stores > Configuration > General > Web > Default Pages.
? In the CMS Home Page field, select the CMS Page that was created or edited in step 1.
? Save the configuration.
There is no ??Home Page?? column in the CMS Page admin grid or ??Default Home Page?? value in the CMS Page admin form.
Verified References: [Adobe Commerce User Guide - Set up your home page]
In Adobe Commerce, to set a custom CMS page as the store's homepage, you need to go to the store configuration. Specifically, navigate to Content > Design > Configuration, select the relevant store view, and then under the "Default Pages" tab, set the "CMS Home Page"
option to your custom CMS page. Options A and B do not exist in the Adobe Commerce admin panel for setting a home page.

NEW QUESTION 119
An Adobe Commerce developer is about to deploy a critical feature to their Adobe Commerce Cloud (Pro Plan) production. They want to create a snapshot in order to be able to rollback if there is an issue with the feature.
How would they create the snapshot?

A. Use the dedicated button on Project Web Interface.
B. Use the Cloud CLI for Commerce dedicated command.
C. Create a ticket to Adobe Commerce Cloud support.

Answer: B

Explanation:
To create a snapshot before deploying changes in Adobe Commerce Cloud (Pro Plan), the recommended approach is to use the Cloud CLI, which provides a dedicated command for creating snapshots. This allows for quick rollback if any issues arise post-deployment.
? Creating a Snapshot with Cloud CLI:
? uk.co.certification.simulator.questionpool.PList@474cafe0
? Why Option B is Correct:
: Adobe Commerce Cloud documentation onCreating and Managing Snapshots https://experienceleague.adobe.com/docs/commerce-cloud-service/user-guide/architecture/pro-architecture.html?lang=en#backup-and-disaster-recovery

NEW QUESTION 120
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 124
A new custom module is built for the existing Adobe Commerce store. A merchant has requested a few frontend updates. For this, a developer has to implement a custom style.
What is the location of the less file that will be included by default?

A. view/{area}/web/css/style less
B. view/{area}/web/css/source/main less
C. view/{area}/web/css/source/_module.less

Answer: B

Explanation:
Theview/{area}/web/css/source/main.lessfile is the default less file that is included by default. This file contains the main styles for the module.
In a custom module in Adobe Commerce, the default location for including custom LESS styles is typicallyview/{area}/web/css/source/_module.less. However, the most commonly used file for adding global styles isview/{area}/web/css/source/_extend.less. Theview/{area}/web/css/style.lessfile is not standard, and themain.lessfile is used as an entry point for compilation but typically does not contain custom styles directly.

NEW QUESTION 128
An Adobe Commerce developer is creating a new module to extend the functionality of the cart. The module is installed in app/code/CompanyName/ModuleName/.
How would an Adobe Commerce developer extend the existing CartItemPrices GraphQL schema to include a custom base_price field?
A) Create and Configure a <prefffrence> for Hagento\QuoteGraphQl\Model\Resolver\CartItemPrices that adds the base_price field in the resolve() function.
B) Add the following to the module's etc/schema.graphqis file:

```
type CartItemPrices {
    base_price: Money!
}
```

A black text on a white background AI-generated content may be incorrect.
C) Add the following to the module's etc/graphqi/di.xmi file:

```
<type name="Magento\QuoteGraphQl\Model\Resolver\CartItemPrices">
    <arguments>
        <argument name="extendedConfigData" xsi:type="array">
            <item name="base_price" xsi:type="number"/>
        </argument>
    </arguments>
</type>
```

A screen shot of a computer code
AI-generated content may be incorrect.

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

Answer: B

Explanation:
The developer can extend the existing CartItemPrices GraphQL schema to include a custom base_price field by adding the following code to the module??s etc/schema.graphqls file:
extend type CartItemPrices { base_price: Money! @doc(description: ??The base price of the cart item??) }
This code adds a new field called base_price to the CartItemPrices type and specifies that it is of type Money and it is not nullable. The @doc directive adds a description for the field that will be shown in the schema documentation. The developer also needs to create a custom resolver class for the base_price field and declare it in the di.xml file of the module. Verified References: [Magento 2.4 DevDocs] [Magento Stack Exchange]

NEW QUESTION 131
On an Adobe Commerce Cloud platform, at what level is the variable env: composer_auth located in the Project Web Interface?

A. In the Environment-specific variables.
B. In the Integration variables.
C. In the Project variables.

Answer: C

Explanation:
The variable env: composer_auth is located in the Project variables section in the Project Web Interface. This variable is used to store the authentication credentials for Composer repositories that require access keys or tokens. The developer can set this variable at the project level to apply it to all environments, or override it at the environment level if needed. Verified References: [Magento 2.4 DevDocs] 2

NEW QUESTION 133
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 138
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 139
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 143
A merchant is experiencing performance issues on integration environments of their Adobe Commerce Cloud Pro plan and wants to upgrade to Enhanced Integration Environments.
What are the steps necessary prior to redeploying in order to upgrade to Enhanced Integration Environments?

A. 1. Limit the number of Integration branches to two* 2. Submit a support ticket requesting the upgrade
B. 1. Limit the number of Integration branches to three* 2. Set the ENV.ENVIRONMENT in .magento.env.yaml to ENHANCEDJNTEGRATION
C. 1. Limit the number of Integration branches to four* 2. Configure integration environments in the cloud GUI and set the Enhanced switch to On

Answer: A

Explanation:
Upgrading to Enhanced Integration Environments in Adobe Commerce Cloud requires specific steps to ensure that the environment is prepared for the upgrade, which includes managing integration branch limits and coordinating with Adobe support.
? Limiting Integration Branches:
? uk.co.certification.simulator.questionpool.PList@6a215712
? Submitting a Support Ticket:
? Why Option A is Correct:
: Adobe Commerce Cloud documentation onEnhanced Integration Environments

NEW QUESTION 146
.....

THANKS FOR TRYING THE DEMO OF OUR PRODUCT

Visit Our Site to Purchase the Full Set of Actual AD0-E724 Exam Questions With Answers.

We Also Provide Practice Exam Software That Simulates Real Exam Environment And Has Many Self-Assessment Features. Order the AD0-E724 Product From:

https://www.2passeasy.com/dumps/AD0-E724/

Money Back Guarantee

AD0-E724 Practice Exam Features:

* AD0-E724 Questions and Answers Updated Frequently

* AD0-E724 Practice Questions Verified by Expert Senior Certified Staff

* AD0-E724 Most Realistic Questions that Guarantee you a Pass on Your FirstTry

* AD0-E724 Practice Test Questions in Multiple Choice Formats and Updatesfor 1 Year