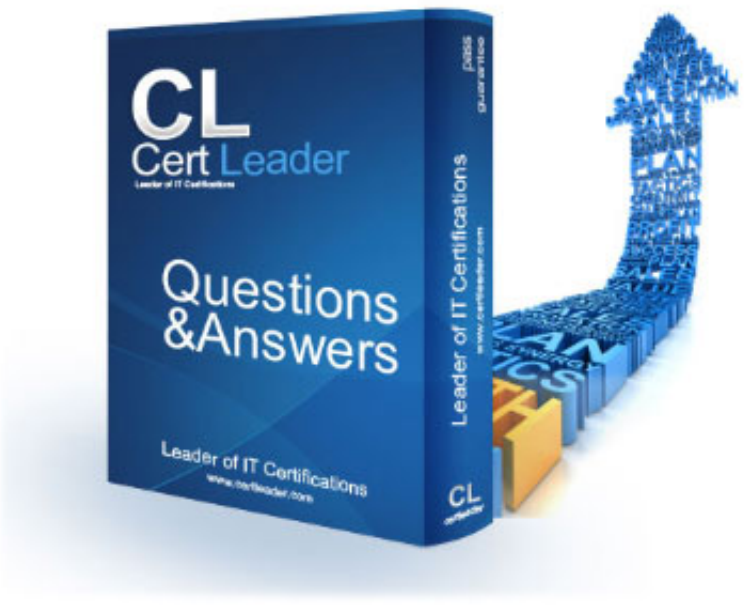
AD0-E724 Dumps

Adobe Commerce Developer Professional

https://www.certleader.com/AD0-E724-dumps.html



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 Adobe Commerce table stores all cron data?

A. schedule
B. cronjob
C. cron_schedule

Answer: C

Explanation:
The Adobe Commerce table that stores all cron job data iscron_schedule. This table maintains records of all scheduled cron tasks, including their statuses, execution times, and any messages related to their execution. It plays a central role in Magento's scheduling system, allowing for themanagement and monitoring of background tasks that are essential for various system functions and integrations.

NEW QUESTION 4
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 5
Which has its own root category?

A. Websites
B. Stores
C. Store Views

Answer: B

Explanation:
In Magento, each store has its own root category. The root category acts as the top-level category under which all other categories and products are organized for that particular store. This structure allows for different catalog structures across multiple stores within a Magento installation, enabling a tailored product offering and navigation experience for each store. Theability to assign a unique root category to each store is a fundamental aspect of Magento's multi-store functionality, providing the flexibility to cater to diverse market segments or branding requirements.

NEW QUESTION 6
An Adobe Commerce developer was asked to provide additional information on a quote. When getting several quotes, the extension attributes are returned, however, when getting a single quote it fails to be returned.
What is one reason the extension attributes are missing?

A. The developer neglected to add coiiection="trueM to their attribute in etc/extension_attributes.xmi file.ottribute code="my_attributesM type="MyVendor\MyModule\Api\Data\^AttributeInterface[]M collection="true" />
B. The developer neglected to provide a plugin On Hagento\Quote\Api\CartRepositoryInterface: :get.
C. The developer neglected to implement an observer on the coiiection_ioad_after event.

Answer: B

Explanation:
In Adobe Commerce, to retrieve custom extension attributes on an entity such as a quote, you need to ensure that these attributes are properly loaded when accessing the entity directly via repository interfaces. When fetching a single quote, it is essential to use a plugin on CartRepositoryInterface::get to include the extension attributes as this method is responsible for loading individual quote data.
If the plugin is missing for the get method, the extension attributes won??t be loaded for single quote retrieval, leading to their absence in the result.
Additional Resources:
? Adobe Commerce Developer Guide: Extension Attributes
? Magento 2 Repositories and Plugins

NEW QUESTION 7
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 8
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 9
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 10
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 10
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 15
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 17
A developer is tasked with creating a new feature in an Adobe Commerce Cloud project. The developer decides to create an integration environment for a better development process.
Which Cloud CLI for Commerce command would the developer use?

A. magento-cloud environment:branch <environment-name> <parent-environment-ID>
B. magento-cloud create:environment-branch <environment-name> <parent-environment- ID>
C. magento-cloud environment:create:branch environment-name > <parent-environment- ID>

Answer: A

Explanation:
The Cloud CLI for Commerce command that a developer would use to create an integration environment for a better development process is magento-cloud environment:branch <environment-name> <parent-environment-ID>. This command creates a new branch in the Git repository and a new environment in the Cloud project, using the specified parent environment as a base. The new environment inherits the code, data, and media files from the parent environment.
To create a new integration environment for development in an Adobe Commerce Cloud project, the developer would use the Cloud CLI for Commerce command:magento-cloud environment:branch <environment-name> <parent-environment-ID>. This command creates a new environment by branching from the specified parent environment, providing a separate environment for developing new features or testing without affecting the live site.

NEW QUESTION 19
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 22
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 27
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 30
An Adobe Commerce developer creates a new website using a data patch. Each website will have unique pricing by website. The developer does not have visibility into the production and staging environments so they do not know what the configuration currently is.
How would they ensure the configuration is deployed and consistent across all environments?
A)

Run the CLI command below and commit the changes to the repository:

```
bin/magento config:set catalog/price/scope 1 --lock-config
```

B)

Run the CLI command below and commit the changes to the repository:

```
bin/magento config:set catalog/price/scope 1
```

C)

Create a custom module and override the value in config.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="urn:magento:module:Magento_Store:etc/c
    <default>
        <catalog>
            <price>
                <scope>1</scope>
            </price>
        </catalog>
    </default>
</config>
```

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

Answer: A

Explanation:
The correct answer isOption A. This approach ensures that the configuration is set using the CLI with the --lock-config flag, which prevents the setting from being overridden by configuration changes in other environments.
? Understanding the Configuration Requirement:
? uk.co.certification.simulator.questionpool.PList@3d6d9941
? Using the CLI with --lock-config:
? Why Other Options Are Incorrect:
: Environment Configuration in Adobe Commerce- Details on managing environment-specific configurations and using the --lock-config flag.
Using the Magento CLI for Configuration Management- Documentation on setting and locking configuration values via CLI.
Option A is the best solution to ensure that the configuration is consistently deployed across all environments while protecting it from changes by leveraging the

--lock-config option.

NEW QUESTION 31
Which Cloud CLI for Commerce command can be used to quickly view a specific log file for an Adobe Commerce Cloud project?

A. magento-cloud log
B. magento-cloud logs:list
C. magento-cloud logs:show

Answer: C

Explanation:
The Cloud CLI for Commerce command that can be used to quickly view a specific log file for an Adobe Commerce Cloud project ismagento-cloud logs:show. This command allows developers to view log files directly from the command line, which is useful for debugging and monitoring the application's state without needing to access the file system directly.

NEW QUESTION 36
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 41
Which theme directory contains the static files that can be loaded directly?

A. web
B. preprocessed
C. assets

Answer: A

Explanation:
Thewebdirectory contains the static files that can be loaded directly. This directory includes files such as CSS, JavaScript, and images.
In Adobe Commerce themes, thewebdirectory is used to store static files such as CSS, JavaScript, images, and fonts. These files can be loaded directly by the browser. The preprocessedandassetsdirectories do not exist as standard directories in the theme structure for containing directly loadable static files.

NEW QUESTION 42
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 47
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 48
What are the only writeable folders in the application root on a remote Adobe Commerce Cloud project?
A)

- var
- app/etc
- pub/media
- pub/static
- /tmp

B)

- m2-hotfixes
- var
- pub/static
- app/etc

C)

- update
- var
- app/etc
- /tmp
- lib

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 52
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 54
An Adobe Commerce developer is being tasked with creating a new cron job to run a method that has already been written. What are the minimally required steps to accomplish this?

A. Create a crontab.xmi file and a new system configuration in system.xmi for the schedule.
B. Create crontab.xmi and cron_groups.xmi files to assign the new job to a cron group.
C. Create a crontab.xmi file and set a schedule for the new cron job.

Answer: C

Explanation:
According to the Configure and run cron guide for Magento 2 developers, the crontab.xmi file is used to declare and configure cron jobs for a module. The file should specify the name, instance, method and schedule of the cron job. Therefore, creating a crontab.xmi file and setting a schedule for the new cron job are the minimally required steps to accomplish this task. Verified References:https://devdocs.magento.com/guides/v2.3/config-guide/cli/config-cli- subcommands-cron.html
To set up a new cron job in Adobe Commerce, you need to define it in the crontab.xml file. This file is essential to schedule cron tasks and is all that's required for simple cron job configurations. You specify the cron job schedule, method, and class in this file.
Here's a minimal example of crontab.xml:
<?xml version="1.0"?>
<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="urn:magento:module:Magento_Cron:etc/crontab.xsd">
<group id="default">
<job name="your_custom_cron_job" instance="Vendor\Module\Cron\YourJob" method="execute">
<schedule>0 * * * *</schedule> <!-- Runs every hour -->
</job>
</group>
</config>
? group id: This specifies the cron group; "default" is commonly used, but you can define custom groups in cron_groups.xml if necessary.
? job: Defines the cron job, with name, instance (path to the class), and method
attributes.
? schedule: Uses standard cron expressions to specify the frequency.
Additional Resources:
? Adobe Commerce Developer Guide: Cron Jobs
? Magento 2 Crontab XML Configuration

NEW QUESTION 56
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 57
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 58
When checking the cron logs, an Adobe Commerce developer sees that the following job occurs daily: main.INFO: Cron Dob inventory_cleanup_reservations is successfully finished. However, the inventory_reservation table in the database is not emptied. Why are there records remaining in the inventory_reservation table?

A. Only reservations matching canceled orders are removed by the cron job.
B. Only reservations no longer needed are removed by the cron job.
C. The "Auto Cleanup" feature from Multi Source Inventory was disabled in configuration.

Answer: B

Explanation:
The reason why there are records remaining in the inventory_reservation table is that only reservations no longer needed are removed by the cron job. The inventory_reservation table tracks the quantity of each product in each order and creates a reservation for each product when an order is placed, shipped, cancelled or refunded. The initial reservation has a negative quantity value and the subsequent reservations have positive values. When the order is complete, the sum of all reservations for the product is
zero. The cron job removes only those reservations that have a zero sum from the table, leaving behind any reservations that are still needed for incomplete orders. Verified References: [Magento 2.4 DevDocs] [Magento Stack Exchange]

NEW QUESTION 61
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 64
What is the correct way to inject CMS block in a layout?

A. <block class="Magento\Cms\Block\Block" name="blockjdentifier"> <arguments>
q<argumentname="block_id"xsi:type="string">my_cms_block_identifier</argument></arguments> </block>
B. <block class="Magento\Cms\Block\Block" name="block_identifier"> q<actionmethod="setBlock'>my_cms_block_identifier</action> </block>
C. <referenceBlock name="content"> <block class="Magento\Cms\Block\Block" name="block_identifier' identifier="my_cms_block_Identrfier" /> </referenceBlock>

Answer: A

Explanation:
The correct way to inject a CMS block into a layout in Adobe Commerce is by using the<block>element with the classMagento\Cms\Block\Blockand specifying the block identifier through an<argument>element with the name "block_id". This is shown in option A. The<block>tag defines the block class and name, and

the<arguments>tag contains child<argument>tags for each argument, where the "block_id" argument specifies the identifier of the CMS block to be injected.

NEW QUESTION 66
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 70
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 72
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 73
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 76
.....

Thank You for Trying Our Product

* 100% Pass or Money Back

All our products come with a 90-day Money Back Guarantee.

* One year free update

You can enjoy free update one year. 24x7 online support.

* Trusted by Millions

We currently serve more than 30,000,000 customers.

* Shop Securely

All transactions are protected by VeriSign!

100% Pass Your AD0-E724 Exam with Our Prep Materials Via below:

https://www.certleader.com/AD0-E724-dumps.html