



Adobe

Exam Questions AD0-E724

Adobe Commerce Developer Professional

About ExamBible

[Your Partner of IT Exam](#)

Found in 1998

ExamBible is a company specialized on providing high quality IT exam practice study materials, especially Cisco CCNA, CCDA, CCNP, CCIE, Checkpoint CCSE, CompTIA A+, Network+ certification practice exams and so on. We guarantee that the candidates will not only pass any IT exam at the first attempt but also get profound understanding about the certificates they have got. There are so many alike companies in this industry, however, ExamBible has its unique advantages that other companies could not achieve.

Our Advances

* 99.9% Uptime

All examinations will be up to date.

* 24/7 Quality Support

We will provide service round the clock.

* 100% Pass Rate

Our guarantee that you will pass the exam.

* Unique Gurantee

If you do not pass the exam at the first time, we will not only arrange FULL REFUND for you, but also provide you another exam of your claim, ABSOLUTELY FREE!

NEW QUESTION 1

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 do
- 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 do
- D. The table will be removed when the bin/magento module:uninstall vendor_ModuleName is run.
- E. Add a schema patch that implements Magento\Framework\Setup\Patch\RevertableInterface 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. Magento will take care of removing the table associated with the module as part of its schema management process.

NEW QUESTION 2

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 is cron_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 the management and monitoring of background tasks that are essential for various system functions and integrations.

NEW QUESTION 3

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

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

Answer: A

Explanation:

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

NEW QUESTION 4

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

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

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

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

Answer: AC

Explanation:

The correct answers are A and C. The errors in the snippet are:

? Column roll_no does not have an index. It is needed since attribute_identity is set to true.

? Column student_name does not have an attribute length. The attribute_identity attribute specifies whether the primary key of the table should be auto-incremented. If attribute_identity is set to true, then the roll_no column must have an index. The student_name column does not have an attribute length, which is required for string columns.

The following code shows how to fix the errors: XML

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

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

NEW QUESTION 5

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

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

Answer: C

Explanation:

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

NEW QUESTION 6

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.xml file within the view/adminhtml/ui_component directory 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 7

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

Answer: A

Explanation:

The developer can update the Staging environment with the latest code from Production by logging in to the Project Web Interface, choosing the Staging environment, and clicking Sync. This will synchronize the code, data, and media files from Production to Staging, creating an exact copy of Production on Staging. The developer can then deploy the new release to Staging and test it before pushing it to Production. Verified References: [Magento 2.4 DevDocs]

NEW QUESTION 8

What are two features with Adobe Commerce Cloud that come out of the box? (Choose Two.)

- A. Support ACL

- B. Continuous deployment provided with the platform
- C. A built in connector with all major blog platforms
- D. Fastly

Answer: BD

Explanation:

Adobe Commerce Cloud offers several out-of-the-box features, including built-in Fastly integration for CDN and web application firewall services, as well as continuous deployment capabilities through its cloud infrastructure.

? Continuous Deployment:

? uk.co.certification.simulator.questionpool.PList@200a841f

? Fastly Integration:

? Why Options B and D are Correct:

:

Adobe Commerce Cloud documentation onFastly CDN

Adobe Commerce Cloud documentation onDeployment and CI/CD

NEW QUESTION 9

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 10

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 10

An Adobe Commerce Cloud merchant has been experiencing significant downtime during production deployment. They have already checked that the application is in ideal state.

In addition to the configuration of the SCD.MATRIX variable to reduce amount of unnecessary theme files, what would be the next steps to reduce the downtime?

- A. 1. Check SCD is configured under the build phase.* 2. Increase the SCD.THREADS to speed up the build process.
- B. 1. Check SCD is configured under deploy phase.* 2. Decrease the SCD.THREADS to speed up the build process
- C. 1. Check SCD is configured under the build phase.* 2. Check if Adobe Commerce Cloud automatically adjusts SCD.THREADS.

Answer: A

Explanation:

To minimize downtime during deployment, one of the most effective strategies is to configure static content deployment (SCD) to run during the build phase and optimize the number of threads used during the process.

? Configuring SCD in the Build Phase:

? uk.co.certification.simulator.questionpool.PList@22cc61b1

? Increasing SCD.THREADS:

? Why Option A is Correct:

:Adobe Commerce Cloud documentation onSCD Configuration

NEW QUESTION 12

In a new release of a module, a developer decides to rename a table that was defined in the earlier versions. Which action, if any, allows the developer to prevent data loss?

- A. Define onCreate="migrateDataFromAnotherTable(old_table_name)" attribute in the table tag.
- B. Declarative schema supports RENAME TABLE', so the data will be migrated to the new table automatically.
- C. Define the table and columns mapping in the db.schema_whitelist.json

Answer: C

Explanation:

When renaming a table in Magento, to prevent data loss, the developer must define the table and its columns mapping in the `db_schema_whitelist.json` file. This declarative schema approach ensures that the data migration tool knows about the changes and can migrate data from the old table to the newly named table without losing any data.

NEW QUESTION 14

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

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` and `DownloadImage`, 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 18

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 file `Magento_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/model` is not a valid path, making option A the correct choice.

NEW QUESTION 22

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 plugin for `Magento\Ui\Component\Listing\Columns\Column::prepareColumn()`.* 2. Add the custom logic within the `afterPrepareColumn` method.
- B. 1. Create a custom class extending `Magento\Ui\Component\Listing\Columns\Column`.* 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 on Creating Custom Columns](#)

[Adobe Commerce documentation on Magento\Ui\Component\Listing\Columns\Column](#)

NEW QUESTION 23

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 25

An Adobe Commerce developer is tasked with adding a 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 29

An Adobe Commerce developer wants to generate a list of products using ProductRepositoryInterface and search for products using a supplier_id filter for data that is stored in a standalone table (i.e., not in an EAV attribute).

Keeping maintainability in mind, how can the developer add the supplier ID to the search?

A. Write a before plugin on \Magento\Catalog\Model\ProductRepository::getList() and register the search criteria passed

B. Write an event observer to listen for the event catalog_product_collection_load_before

C. Iterate through the registered search criteria, and if found, apply the needed join and filter to the events collection.

D. Add a Custom filter to the Virtual type "Magento\Catalog\Model\Api\SearchCriteria\CollectionProcessor\ProductFilterProcessor" for supplier_id field

E. In the custom filter, apply the needed join and filter to the passed \$collection.

F. Write a before plugin On \Magento\Framework\Api\SearchCriteria\CollectionProcessorInterface::process(). Iterate through the \$searchCriteria provided for supplier_id, and if found, apply the needed join and filter to the passed collection.

Answer: B

Explanation:

The developer can add a custom filter to the virtual type Magento\Catalog\Model\Api\SearchCriteria\CollectionProcessor\ProductFilterProcessor for supplier_id field. In the custom filter, the developer can apply the needed join and filter to the passed \$collection. This is the recommended way to extend the search criteria for products using dependency injection and plugins. Verified References: [Magento 2.4 DevDocs] [Magento Stack Exchange]

In Adobe Commerce, when you need to add a custom filter for a non-EAV attribute stored in a standalone table, the most maintainable approach is to create a custom filter for ProductFilterProcessor. This processor allows for customized search criteria handling, which can be extended to include custom joins and filters without altering core functionality or relying on plugins and observers.

? Why Custom Filter in ProductFilterProcessor is Preferred:

? uk.co.certification.simulator.questionpool.PList@205c9928

? Implementation of Custom Filter:

? Why Options A and C are Less Suitable:

:

Magento Developer Documentation onUsing Custom Filters in CollectionProcessor Magento DevDocs onDependency Injection and Virtual Types

NEW QUESTION 32

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

A. Builds application, Applies custom patches and Dump configuration for static content deployment.

B. Fastly configuration, Applies custom patches and Dump configuration for static content deployment.

C. Builds application, Applies custom patches, and Shows the list of S3 backup tar.gz files.

Answer: A

Explanation:

Some of the features provided by ECE-Tools are building application, applying custom patches, and dumping configuration for static content deployment. ECE-Tools is a set of scripts and tools designed to manage and deploy Adobe Commerce Cloud projects. It provides commands for building application code, applying patches for Magento core issues or custom modules, and dumping configuration settings for static content deployment optimization. Verified References: [Magento 2.4 DevDocs] 2

The ECE-Tools package for Adobe Commerce Cloud provides a range of tools and scripts to manage and streamline deployment and maintenance tasks. Among its key features:

? Application Builds:

? uk.co.certification.simulator.questionpool.PList@2bc44d9b

? Applying Custom Patches:

? Dump Configuration for Static Content Deployment:

? Why Option A is Correct:

: Adobe Commerce Cloud documentation onECE-Tools

NEW QUESTION 37

What action can be performed from the Cloud Project Portal (Onboarding UI) of an Adobe Commerce Cloud project?

A. Set your developer SSH public key.

B. Update Project and environment variables

C. Add a Technical Admin

Answer: C

Explanation:

The Cloud Project Portal (Onboarding UI) of an Adobe Commerce Cloud project is a web interface that allows you to perform various actions related to your project, such as creating and managing environments, deploying code, configuring services, and adding users1. One of the actions that you can perform from the Cloud Project Portal is adding a Technical Admin, which is a user role that has full access to all environments and can perform any action on the project2. To add a Technical Admin from the Cloud Project Portal, you need to follow these steps2:

? Log in to the Cloud Project Portal with your Magento account credentials.

? Click on the Users tab on the left sidebar.

? Click on the Add User button on the top right corner.

? Enter the email address of the user you want to add as a Technical Admin.

? Select the Technical Admin role from the Role dropdown menu.

? Click on the Send Invitation button.

The user will receive an email invitation to join your project as a Technical Admin. They will need to accept the invitation and set up their account before they can access your project2.

NEW QUESTION 42

An Adobe Commerce developer is writing an integration test. They checked some

Integration Tests for Magento core modules for reference and noticed that they use data fixtures initialized by adding annotations to test classes. For example:

```
/**
 * @magentoDataFixture Magento/AdminNotification/_files/notifications.php
 */
```

The developer wants to add their own fixture to test a MyVendor_MyModule they created. Which steps will make this possible?

A. 1. Create a PHP file with the fixture data inside their own module in [module dir]/Test/integration/_files/my_fixture.php.* 2. Add the following annotation to the test method:

```
/**
 * @magentoDataFixture MyVendor_MyModule::Test/Integration/_files/my_fixture.php
 */
```

B. 1. Create a PHP file With the fixture data in [magento root dir]/dev/tests/integration/testsuite/MyVendor/MyModule/_files/my_fixture.php.* 2. Add the following annotation to the test method:

```
/**
 * @magentoDataFixture MyVendor_MyModule::_files/my_fixture.php
 */
```

C. 1. Create a PHP file with the fixture data inside their own module in [module dir]/Test/integration/_files/my_fixture.php.* 2. Add the following annotation to the test method:

```
/**
 * @magentoDataFixture MyVendor/MyModule/_files/my_fixture.php
 */
```

Answer: A

Explanation:

? Magento Integration Test Fixtures:

? uk.co.certification.simulator.questionpool.PList@15827b0f

? Directory Structure for Module-Specific Fixtures:

? Annotation Usage in Option A:

? Why Other Options Are Incorrect:

:

Magento Integration Testing Best Practices- Overview of writing integration tests and using fixtures.

@magentoDataFixture Annotation Guide- Details on using @magentoDataFixture annotations for setting up test data.

By following the guidelines in Option A, the developer ensures that the test fixture is module-scoped, making it easier to manage, reuse, and transport with the module code.

NEW QUESTION 46

A merchant of an Adobe Commerce Cloud project wants to setup one of their websites using a subdomain. The merchant is considering the domain to be set as secondstore.example.com.

In addition to editing the magento-vars.php file, and apply a domain check and set

\$_SERVER["MAGE_RUN_CODE"] and \$_SERVER["MAGE_RUN_TYPE"].

What file is required to perform this action?

A. Configure secondstore.example.com subdomain route in NGINX virtual-host configuration file.

B. Configure secondstore.example.com subdomain route in .magento/services.yaml.

C. Configure secondstore.example.com subdomain route in .magento/routes.yaml.

Answer: C

Explanation:

In Adobe Commerce Cloud, routing configurations for custom domains and subdomains are managed within the .magento/routes.yaml file. This file defines how requests are directed to the application and is essential for setting up different stores or websites with unique subdomains.

? Using .magento/routes.yaml for Subdomain Configuration:

? uk.co.certification.simulator.questionpool.PList@277b7595

? Why Option C is Correct:

? Example Configuration: http://secondstore.example.com/: type: upstream

upstream: "mymagento:80"

:

Adobe Commerce Cloud documentation onConfiguring routes.yaml

NEW QUESTION 48

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

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

How would the developer implement the validations?

A) Add the Validations Within the HyVendor\MyModule\Controller\Adminhtml\CustomEntity\UploadPdf Controller

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

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

B) Add a virtual type for MyVendor\MyModule\Model\customPdfuploader specifying the allowedExtensions and the maxFileSize for the constructor, within the module's di.xml:

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

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

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

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

Answer: C

Explanation:

To add file upload validation for a custom form field in the Adobe Commerce admin panel, which should restrict the file type to .pdf and limit the file size to 2 MB, the recommended approach is to include the validation parameters directly within the <settings> node in the form's XML configuration. This ensures that the validation occurs on the client-side as well as server-side, providing immediate feedback to users before submission.

Option C is correct for the following reasons:

? Adding Validation Inside <settings>: By placing the <allowedExtensions> and

<maxFileSize> tags within the <settings> node of the XML configuration, the system will enforce these restrictions directly within the form component. This method leverages Magento's built-in support for validation settings, which ensures that only files matching the specified criteria (.pdf files of 2 MB or smaller) are accepted by the form.

? uk.co.certification.simulator.questionpool.PList@78475b0d

: Magento's documentation on UI components highlights how to enforce file type and size restrictions through XML configurations within <settings>, making it the standard and most efficient solution for this task.

Alternatives and Limitations:

Option A: Adding validation within the controller (UploadPdf) can provide additional server-side validation, but this does not prevent the user from selecting invalid files in the first place. Server-side validation alone lacks the user experience enhancement provided by client-side feedback.

Option B: Configuring a virtual type in di.xml for validation (e.g., setting allowedExtensions and maxFileSize within a custom uploader model) is effective for backend processing but is not as straightforward for direct form validation within the admin UI. It complicates the implementation by requiring custom backend logic where native XML validation can suffice.

Option C provides a straightforward, maintainable, and user-friendly way to implement file validation directly in the form configuration. It reduces the need for custom controller or model logic and leverages Magento's built-in form handling capabilities.

NEW QUESTION 51

Which index mode is valid?

- A. Update on refresh
- B. Update on invalidate
- C. Update on save

Answer: C

Explanation:

"Update on save" is a valid index mode in Magento, where the index is set to update automatically whenever a change is saved. This mode ensures that the index remains up-to-date with the latest data changes, such as product or category updates, immediately reflecting these changes on the storefront. This real-time indexing is crucial for maintaining data accuracy and consistency across the Magento site, especially in dynamic environments with frequent updates.

NEW QUESTION 52

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

- A. catalog_product_view_type_downloadable
- B. catalog_product_view_type_configurable
- C. catalog_product_view_category

Answer: A

Explanation:

The catalog_product_view_type_downloadable layout can be used to customize a product page layout for a downloadable product. This layout includes the product details, the product reviews, and the download links for the product's files.

For selling electronic versions of albums with individual songs, the downloadable product type in Adobe Commerce is appropriate. To customize the product page layout specifically for downloadable items, the layout handle catalog_product_view_type_downloadable is used. This layout handle allows developers to target downloadable products specifically and apply custom layouts or templates, making option A correct.

NEW QUESTION 55

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_config_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 57

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 the `forward` method 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 59

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

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

Answer: C

Explanation:

Multiple EAV attributes belonging to the same entity are grouped in the database based on their data types, such as datetime, decimal, int, text, or varchar. For

example, all attributes with datetime values are stored in one table, while all attributes with text values are stored in another table. The sizes or numeric/text values of attributes do not determine how they are grouped in the database.

Verified References: [Adobe Commerce Developer Guide - EAV data model]

Magento's EAV (Entity-Attribute-Value) model organizes attributes based on their data types to optimize storage and retrieval. Attributes are grouped into different tables based on whether they store values of types such as integer, varchar (text), decimal, datetime, etc. This organization allows Magento to efficiently manage the diverse data types associated with products, customers, and other entities, ensuring data integrity and optimizing database performance by using appropriate indexing and storage mechanisms for each data type.

NEW QUESTION 62

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 66

A developer needs to initialize the jQuery UI widget for a specific HTML tag. Which HTML attribute is used for this?

- A. x-magento-init
- B. data-mage-init
- C. data-ui

Answer: B

Explanation:

The data-mage-init HTML attribute is used to initialize the jQuery UI widget for a specific HTML tag. This attribute specifies the name of the widget and its configuration options as a JSON object.

The x-magento-init HTML attribute is used to initialize RequireJS modules for a specific HTML tag. The data-ui HTML attribute does not exist in Adobe Commerce.

Verified References: [Adobe Commerce Developer Guide - Initialize JavaScript components using the data-mage-init attribute]

To initialize the jQuery UI widget in Adobe Commerce, the data-mage-init attribute is used in the HTML tag. This attribute allows specifying the widget's component and its configuration in a JSON format. The x-magento-init is used for initializing JavaScript components in a similar manner but is typically used within <script> tags. There's no standard data-ui attribute used for this purpose in Magento 2.

NEW QUESTION 71

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

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

Answer: B

Explanation:

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

NEW QUESTION 74

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\PoisonPillPutInterface::put().
- B. Call the function \Magento\Framework\MessageQueue\Consumers\TriggerRestartInterface::trigger().
- C. Set the global Cache key trigger_consumer_restart to 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 79

Which two attribute input types can be used for a date? (Choose two.)

- A. Timezone
- B. Schedule
- C. Date and Time
- D. Date

Answer: CD

Explanation:

The two attribute input types that can be used for a date are Date and Time and Date. These input types allow the user to select a date or a date and time from a calendar widget.

The Timezone and Schedule input types do not exist in Adobe Commerce. Verified References: [Adobe Commerce User Guide - Create a product attribute]

In Magento, attribute input types define the type of data an attribute can hold and how it should be inputted or displayed in the UI. For dates, Magento provides specific input types to handle date-related data efficiently. The "Date" input type is used for attributes that require only a date, without a time component, suitable for birthdays, anniversaries, or any date-specific information. The "Date and Time" input type, on the other hand, includes both date and time components, ideal for events, promotions, or any scenario where the time of day is relevant. These input types ensure data consistency and provide a user-friendly interface for selecting dates and times.

NEW QUESTION 80

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 83

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

What is a possible reason for this?

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

Answer: C

Explanation:

For a payment method to be available in the admin panel (backend), the configuration must explicitly allow its internal use. This is controlled by the `can_use_internal` flag in `config.xml`.

? Configuration for Admin Use:

? `uk.co.certification.simulator.questionpool.PList@31da258e`

? Why Option C is Correct:

: Adobe Commerce DevDocs onPayment Method Configuration Magento Developer Guide onCustom Payment Method

NEW QUESTION 86

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 valid integration token in swagger in order to see the new method.
- C. The @return annotation is missing in the MyVendor\Blog\Api\PostRepositoryInterface 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 like Swagger (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.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>
```

? Option A suggests 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 B is 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 C mentions 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 Integration for 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 90

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 93

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 in 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 on Email Configuration

NEW QUESTION 94

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

- A. Order
- B. Shipping
- C. Category

Answer: B

Explanation:

According to the Adobe Commerce User Guide, a tax class can be assigned to either a product or a customer group in Adobe Commerce. A product tax class determines how a product is taxed, while a customer tax class determines how a customer is taxed based on their location and group membership. Shipping is considered as a product tax class in Adobe Commerce, and it can be assigned to different shipping methods or rates. The other options are not valid for assigning a tax class.

In Adobe Commerce, tax classes can be assigned to products and shipping. Categories, however, do not have tax classes assigned to them directly. Tax classes applied to shipping allow merchants to specify how taxes should be calculated for shipping costs, making option B the correct answer. Orders and categories do not have direct associations with tax classes in the same way products and shipping do.

NEW QUESTION 96

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

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

Answer: AD

Explanation:

By default, third-party modules are located in vendor/ or app/code/ directories. The vendor/ directory contains modules that are installed using Composer, while the app/code/ directory contains modules that are manually copied or cloned from a repository.

Third-party modules in Magento 2 are typically located in two directories by default: the vendor/directory and the app/code/directory. The vendor/directory is used for modules that are installed via Composer, Magento's dependency manager, which includes both Magento's core modules and third-party modules.

The app/code/directory is used for custom modules developed specifically for the project or for third-party modules that are manually installed without Composer.

These conventions provide a structured approach to managing Magento modules, whether they are part of Magento's core functionality, contributed by the community, or developed for specific projects.

NEW QUESTION 99

An Adobe Commerce developer wants to create a product EAV attribute programmatically which should appear as WYSIWYG in the admin panel. They have made sure that wysiwyg_enabled has been set to true, however, the attribute is not appearing as WYSIWYG in the admin panel.

What would be a possible reason?

- A. The is_html_allowed_on_front Option is Set to false.
- B. The input type is not set to text.
- C. The input type is not set to textarea.

Answer: C

Explanation:

The input_type attribute of a product EAV attribute specifies the type of input field that will be used to enter the value of the attribute in the admin panel.

The textarea input type is used for WYSIWYG fields. If the input_type attribute is not set to textarea, then the attribute will not appear as WYSIWYG in the admin panel.

To fix this, the developer should set the input_type attribute to textarea.

NEW QUESTION 101

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 103

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:

The way a developer would access RabbitMQ data on an Adobe Commerce Cloud Production environment is by using local port forwarding. This method allows the developer to connect to the RabbitMQ service instance through an SSH tunnel and access the RabbitMQ Management UI from a web browser. The developer needs to use the `magento-cloud ssh` command to establish the SSH connection and the `$MAGENTO_CLOUD_RELATIONSHIPS` variable to retrieve the RabbitMQ connection details and login credentials.

NEW QUESTION 104

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:

The `view/{area}/web/css/source/main.less` file 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 typically `view/{area}/web/css/source/_module.less`. However, the most commonly used file for adding global styles is `view/{area}/web/css/source/_extend.less`. The `view/{area}/web/css/style.less` file is not standard, and the `main.less` file is used as an entry point for compilation but typically does not contain custom styles directly.

NEW QUESTION 106

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

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

Answer: C

Explanation:

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

Here is an example of a plugin for the `executeRow()` method: PHP

```
class MyPlugin
{
    public function executeRow(
        \Magento\Catalog\Model\Indexer\Product\Price $subject,
        \Magento\Catalog\Model\Product $product, array $data
    ) {
        $adjustment = $this->getAdjustment($product);
        $product->setPrice($product->getPrice() + $adjustment);
    }
    private function getAdjustment(Product $product)
    {
        $adjustment = $product->getData('adjustment');
        if (!is_numeric($adjustment)) { return 0; }
        return $adjustment;
    }
}
```

This plugin will add the `adjustment` data from the product to the product price index. Once the product price index is updated, the layered navigation will display the correct price.

NEW QUESTION 110

An Adobe Commerce developer is working on a Magento 2 instance which contains a B2C and a B2B website, each of which contains 3 different store views for English, Welsh, and French language users. The developer is tasked with adding a link between the B2C and B2B websites using a generic link template which is used throughout the sites, but wants these links to display in English regardless of the store view.

The developer creates a custom block for use with this template, before rendering sets the translate locale and begins environment emulation using the following

code:

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

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

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

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

Answer: B

Explanation:

The startEnvironmentEmulation() method resets the translation locale to the one of the emulated stores, which overrides the locale the developer has set when the order of setLocale() and startEnvironmentEmulation() is used as displayed above.

The correct way to achieve the desired result is to use the emulate() method to temporarily modify the translation locale. The following code shows how to do this:

PHP

```
$this->_translate->emulate('en_US');
```

```
// Render the template
```

```
$this->_translate->revert();
```

This code will set the translation locale to English before rendering the template, and then revert the locale back to the default value after the template has been rendered. The startEnvironmentEmulation() method is used to emulate a different store view or website. This can be useful for testing purposes, or for developing features that need to work in different environments.

The emulate() method is used to temporarily modify the translation locale. This can be useful for rendering templates in a specific language, or for testing features that need to work in different languages.

NEW QUESTION 115

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 117

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 122

Which type of product has the ability to build customizable products from a variety of options?

- A. Grouped
- B. Virtual
- C. Bundle

Answer: C

Explanation:

A bundle product is a product that is made up of a collection of other products. This type of product is ideal for selling products that are often purchased together, such as a printer and ink cartridges.

A bundle product in Adobe Commerce has the ability to build customizable products from a variety of options. Bundle products are ideal for creating custom kits or packages where customers can choose from options for each component. For example, building a custom computer setup from selected components like monitors, keyboards, CPUs, etc. Grouped products are collections of standalone products that can be purchased individually, and virtual products are intangible.

items.

NEW QUESTION 123

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

- A. show.in
- B. apply_to
- C. allowed_in

Answer: B

Explanation:

The `apply_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 128

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

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

Answer: A

Explanation:

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

NEW QUESTION 132

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 133

.....

Relate Links

100% Pass Your AD0-E724 Exam with Exambible Prep Materials

<https://www.exambible.com/AD0-E724-exam/>

Contact us

We are proud of our high-quality customer service, which serves you around the clock 24/7.

Viste - <https://www.exambible.com/>