DUMPSBOSS.

Salesforce Certified B2C Commerce Developer

Salesforce B2C-Commerce-Developer

Version Demo

Total Demo Questions: 36

Total Premium Questions: 367

Buy Premium PDF

https://dumpsboss.co

support@dumpsboss.co

support@dumpsboss.co
dumpsboss.co

Topic Break Down

Topic	No. of Questions
Topic 1, B2C Commerce Setup	53
Topic 2, Work With a B2C Site	17
Topic 3, Data Management Using Business Manager	86
Topic 4, Application Development	145
Topic 5, Testing and Debugging	66
Total	367

QUESTION NO: 1

A developer wants to create in Business Manager extension with thecartridge named

plugin_vm_extension.

Which two steps should the developer take for the extension option to show up in Business Manager?Choose 2 answers:

A. Add plugin_bm_extension to the cartridge path under business manager cartridge site

B. Add the appropiate roles and permission to the user to view the business manager extension.

C. Add plugin_bm_extension to the cartridge path under Storefront cartridge site path.

D. Activate a new code version for the Business Manager Site.

ANSWER: A B

Explanation:

“Add plugin_bm_extension to the cartridge path under business manager cartridge site” is required because Business Manager loads extension cartridges from the cartridge path configured specifically for the Business Manager site. Once the cartridge is present in that path, Business Manager can discover the extension metadata and its module definitions. The extension cartridge must also be deployed as part of the active code version so that its files and metadata are available to the instance.

“Add the appropiate roles and permission to the user to view the business manager extension.” is also required because Business Manager modules are protected by permissions. A developer defines access to extension modules through the extension configuration, and an administrator grants the relevant module permission to the appropriate Business Manager role. Users see the extension’s navigation entry only when their assigned role includes that permission. This role-based access model lets an extension be installed and available while limiting visibility and use to authorized users.

Salesforce documents the required Business Manager cartridge-path setup and extension-module configuration in the Business Manager Extensions guide. Permission assignment is managed through Business Manager’s roles and permissions administration features.

QUESTION NO: 2

The developer has been given the following business requirement:

The shipping method, "Free Standard Ground Shipping" has an exclusion for products with 'category equals or is child of electronics-televisions.'

The marketing department has scheduled a sale offering a "Free Standard Ground Shipping" method for brand XyzTv televisions for the next 3 months.

What method accomplishes this while following best practices'

A. Create a new shipping method and label it "Free Standard Ground Shipping". Give it the qualifier 'brand equals XyzTv', and add it to the checkout options.

B. Create an allow list for the existing shipping method by adding a product exclusion for 'brand equals XyzTv' to the exclusion list for "Free Standard Ground Shipping."

C. Extend the CheckoutShippingServices controller using module.superModule and add an exception for the specified brand.

D. Extend the code in cartridge/models/shipping/shippingMethod.js using module.superModule and add an exception for the specified brand.

ANSWER: D

Explanation:

Extend the code in `cartridge/models/shipping/shippingMethod.js` using `module.superModule` and add an exception for the specified brand. This keeps the temporary business-specific shipping availability logic in the SFRA shipping-method model, the layer responsible for producing the applicable shipping methods used by checkout. The extension can call the base model through `module.superModule`, preserve standard Shipping Manager behavior for every normal product and shipment, and add the existing free-shipping method only when the basket contains the qualifying XyzTv television condition. This is preferable to changing the configured category exclusion, because that exclusion remains the default protection for other television products throughout the sale. It also avoids duplicating an existing shipping method and avoids putting reusable shipping-domain logic into a route controller. Cartridge extension through `module.superModule` supports upgrade-friendly customization: custom code can augment SFRA behavior while retaining the base implementation. The implementation should be scoped to the campaign's brand and removed or disabled when the three-month promotion ends. Salesforce's SFRA customization guidance is available in the SFRA customization documentation, and the platform shipping APIs are documented in the ShippingMgr Script API reference.

QUESTION NO: 3

A developer set up a newsite with Taxation: Net. However, the business requirements changed and the

site now needs to be Taxation:Gross. The Business Manager interface does not give this option.

Which sequence of steps is necessary to change the site to gross taxation?

A. Make sure that the developer has "Administrator" Access, then change the Taxation setting to Gross

B. Unlock the site preferences and then change the Taxation setting to Gross

C. Change the global setting, "Enable Taxation Methods" to true, then change the Taxation setting to Gross

D. Create a new site with Taxation set to Gross, then delete the old site.

ANSWER: C

Explanation:

Change the global setting, "Enable Taxation Methods," to true, then change the Taxation setting to Gross is correct because the availability of the site-level taxation method is governed by the instance-wide taxation-method setting. When taxation methods are not enabled globally, Business Manager does not expose the per-site choice needed to switch between net and gross taxation. Enabling the setting makes the Taxation preference available for the site, allowing the developer to select Gross and have catalog prices interpreted as tax-inclusive. This is important because net and gross taxation affect how prices, discounts, and tax amounts are represented throughout storefront pricing and order calculations. The change should therefore be made deliberately, with appropriate testing of tax configuration, price books, and checkout behavior for the affected site. Salesforce Commerce B2C Commerce documentation describes taxation configuration and the platform's support for net and gross pricing models: Salesforce B2C Commerce Taxation. For additional platform guidance on tax calculation APIs and taxation behavior, see TaxMgr Script API.

QUESTION NO: 4

A merchant wants customers to be able to order gift vouchers via their site. Since they can issue an

unlimited number of these digital vouchers, this item should be available to sell at all items.

How can a developer use Business Manager to ensure that the gift vouchers are always available?

A. Check the perpetual flag in the product inventory record

B. Check the Available to Sell (ATS) flag dor the producto set

C. Set StockLevel = maxAllocation for the producto.

D. Manually set the inventory to a high number.

ANSWER: A

Explanation:

Check the perpetual flag in the product inventory record is correct because perpetual inventory is specifically intended for products that do not have a finite physical stock quantity. Gift vouchers are digital goods that can be issued repeatedly, so their availability should not depend on a decreasing on-hand inventory count. In Business Manager, the developer can configure the product's inventory record as perpetual within the relevant inventory list. Once marked perpetual, the inventory record represents an unlimited supply for availability purposes, allowing shoppers to continue ordering the voucher without normal stock depletion making it unavailable.

This configuration is appropriate when the merchant does not need to track a limited quantity of the item. The product remains governed by its normal catalog, site-assignment, and availability-date settings, but the inventory record itself no longer imposes a finite stock constraint. Salesforce's inventory model exposes this setting through the inventory record's perpetual property, which identifies inventory that is continuously available. See the ProductInventoryRecord Script API reference and Salesforce's B2C Commerce inventory documentation.

QUESTION NO: 5

In Log Center, a developer notes a number of Cross Site Request Forgery (CSRF) log entries. The developer knows that this happens when a CSRF token is either not found or is invalid, and is working to remedy the situation as soon as possible.

Which two courses of action might solve the problem?

Choose 2 answers

A. Add the token in the ISML template.

B. Extend the CSRF token validity to avoid timeouts.

C. Delete the existing CSRF whitelists in Business Manager.

D. Add csrfProtection.generateToken as a middleware step in the controller.

ANSWER: A D

Explanation:

Add the token in the ISML template. is required because a protected form submission must send the generated CSRF token back with the request. In an SFRA form, the template normally renders hidden fields using the token name and token value placed in the view data. Rendering those fields ensures that the POST request contains the parameter expected by the CSRF validation middleware. If the form omits the fields, validation logs a missing-token event.

Add csrfProtection.generateToken as a middleware step in the controller. supplies the token data that the ISML template needs. This middleware runs on the route that displays the form and adds a CSRF token to the pipeline dictionary/view data. The template can then render the token name and value into the form, while the form-processing route uses the appropriate CSRF validation middleware. Together, token generation before rendering and token inclusion in the submitted form establish the expected request-validation flow and address log entries caused by absent or invalid tokens.

Salesforce documents the CSRF protection workflow, including generating a token for rendering and validating it on form submission, in the B2C Commerce CSRF protection guide. The SFRA implementation is also available in the Storefront Reference Architecture.

QUESTION NO: 6

In order to implement site custom functionality, a developer creates a new cartridge.

Which step should the developer take to ensure their cartridge changes take effect?

A. Add the new cartridge to the cartridge path for the business Manager site.

B. Rebuild the site indexes to capture incremental changes.

C. Add the new cartridge to the cartridge path for the relevant Storefront site.

ANSWER: C

Explanation:

Adding the new cartridge to the cartridge path for the relevant Storefront site ensures that B2C Commerce can locate and execute the cartridge's controllers, templates, scripts, and other customized resources when requests are processed for that site. Cartridge paths are configured at the site level in Business Manager, so deploying or uploading cartridge code alone does not make its functionality active. The cartridge must be included in the appropriate site's path, and its placement in that path is significant because B2C Commerce resolves compatible cartridge resources according to cartridge-path precedence. After the cartridge path is updated, requests for the relevant storefront use the new code and resources, subject to normal caching behavior where applicable. This site-specific configuration also supports different storefronts using different sets or versions of cartridges in the same instance. Salesforce describes cartridge paths and their configuration in its Business Manager and B2C Commerce development documentation: B2C Commerce Cartridges and Business Manager.

QUESTION NO: 7

A developer hasa sandbox configured with a service and its profile and credential. Now there is a requirement to allow changes to the service URL manually from the sandbox.

Which B2C feature should the developer use to achieve the request?

A. Use the service credentialURL field

B. Use the service status area, set the override URL checkbox, and then populate the URL field with the required one.

C. Use a Site preference dedicated for the service URL

D. Use a Global preference dedicated for the service URL

ANSWER: B

Explanation:

Use the service status area, set the override URL checkbox, and then populate the URL field with the required one. B2C Commerce service configuration supports a URL override at the service level, specifically enabling an administrator or developer to direct calls from a nonproduction instance, such as a sandbox, to a different endpoint without changing the service's underlying credential or profile configuration. The override is managed in Business Manager's Administration > Operations > Services area, where the service status settings expose the override control and URL value. This is useful for sandbox testing because the integration code can continue to use the same named service while its runtime destination is adjusted manually for a test endpoint or environment-specific endpoint. The URL override is a configuration capability of the Service Framework and avoids requiring a code deployment merely to change the target endpoint. Salesforce's Service Framework documentation describes service configuration and administration: B2C Commerce Web Services and Service Framework documentation. For the broader B2C Commerce developer documentation catalog, see Salesforce B2C Commerce Developer Documentation.

QUESTION NO: 8

A merchant is selling a new product line of televisions. In order to deliver a good customer experience, the merchandising team wants the screen size to be incorporated into the search and navigation journey.

Which two things can the developer do to facilitate this for them?

Choose 2 answers

A. Create a new search refinement for a Boolean value true or false and label it "big screen."

B. Define a new searchable attribute for Screen Size.

C. Configure catalog-level search refinement definition for Screen Size.

D. Configure Screen Size threshold search refinement bucket definitions.

ANSWER: B C

Explanation:

Defining a new searchable attribute for Screen Size makes the product attribute available to the search index, allowing shoppers' searches to match television products based on their recorded screen-size data. The attribute should be populated consistently on the relevant products so it can contribute useful information to the indexed catalog and search experience.

Configuring a catalog-level search refinement definition for Screen Size makes that indexed attribute usable as a navigation refinement in the storefront. Refinements are configured against a catalog because they determine the facets shoppers can use to narrow product-search results within that catalog. For a numeric value such as screen size, the refinement can be configured to present meaningful selectable values or ranges, supporting navigation such as filtering televisions by their displayed screen measurement. Together, indexing the Screen Size attribute and defining it as a catalog refinement supports both search relevance and faceted navigation. Salesforce describes searchable attributes and search refinements in its B2C Commerce Search documentation and provides configuration guidance in Search Refinements.

QUESTION NO: 9

A Digital Developer noticed that cartridges in their workspace are NOT executing. The Developer confirms that the cartridges are uploaded to the B2C Commerce server connection's target version directory.

Which action potentially solves this problem?

A. Set the active code version to use the latest compatibility mode.

B. Remove invalid characters from the code version's name.

C. Remove invalid characters from cartridge file and folder names.

D. Set the server connection's target version directory to the active code version.

ANSWER: D

Explanation:

Set the server connection's target version directory to the active code version. In B2C Commerce, uploading a cartridge to an instance only makes its files available under a particular code-version directory; it does not cause the instance to run that code version. The instance executes the code version currently designated as active in Business Manager. Therefore, the directory selected in the development environment's server connection must correspond to that active version when the developer intends uploaded cartridge changes to be used. Aligning the target version directory with the active code version ensures the cartridge files are deployed where the runtime can load them. After deployment, the cartridge must also be included in the relevant cartridge path for the site or Business Manager context, as appropriate. This issue commonly occurs when a connection still targets an older, inactive, or differently named version directory while the instance is configured to execute another version. Salesforce's code-versioning guidance explains that only one code version is active on an instance at a time and that activation determines the version used by the storefront runtime. See Salesforce B2C Commerce Code Versioning and Salesforce B2C Commerce Cartridges.

QUESTION NO: 10

A developer is working on a new site for the U.S based on an existing Canadian site. One of the

requirements is a change to the address form. The current Canadian form has an list with the

correct two-letter abbreviation for the provinces.

The U.S. requirements are to:

Have an list with the correct two-letter abbreviation for the states in place of the

province field.

Set the U.S site locale.

Add the options list field definition to the XML file.

How should the developer set up the files before making the required edits?

A. Create a copy of existing address.xml file in the default folder. Rename that file to adres_US.xml

B. Create a new sub-folder in the forms folder. Name it US. Copy existing address.xml file in the new folder.

C. Create a copy of existing address.xml file in the default folder. Rename that file to address_en_US.xml

D. Create a new sub-folder in the forms folder. Name it en_US. Copy existing address.xml file in the new folder.

ANSWER: D

Explanation:

Create a new sub-folder in the forms folder. Name it en_US. Copy existing address.xml file in the new folder. is correct because B2C Commerce supports locale-specific form definitions through locale-named directories under a cartridge's `forms` directory. The developer should retain the shared/base form definition in `forms/default` and place the U.S.-specific version at `forms/en_US/address.xml`. When the site request locale is `en_US`, B2C Commerce resolves and uses that localized form definition, allowing the U.S. address form to define a state select field and its state abbreviations without changing the Canadian or default implementation.

Copying the existing address form first provides the same field structure and validation baseline, after which the developer can update the relevant field and option definitions for U.S. states. The locale identifier must use the Commerce locale format, `en_US`, rather than only a country code such as `US`. This folder-based localization model also keeps form presentation and field metadata organized by locale, making later maintenance of country-specific address requirements straightforward. See Salesforce's documentation on B2C Commerce forms and the localization framework.

QUESTION NO: 11

In the SFRA Page controller, the following route exists:

The result of navigating to the address below is an error page.

What is the correct way to use this controller route in an ISML template?

A)

B)

C)

A. ${Resource.msg('aboutus', 'content', null)}

B.

C.

ANSWER: B

Explanation:

`<isinclude url="${URLUtils.url('Page-Include', 'cid', 'about-us')}">` is the correct use because the `Page-Include` endpoint is intended to be invoked as a remote include from an ISML-rendered page. In SFRA, routes used

to render included content are commonly protected by the include middleware. That middleware validates that the request was made in an include context, which is why entering the generated `Page-Include` URL directly in the browser can result in an error page. `URLUtils.url()` constructs the storefront URL for the named route and supplies the page identifier through the `cid` parameter. The `isinclude` tag then makes the server-side include request and inserts the controller response into the containing template. This preserves the intended request flow for rendering Page Designer content or other partial output and allows the route's configured middleware and cache behavior to apply. The SFRA base cartridge's `Page.js` controller illustrates the Page routes and their middleware usage. Salesforce's template documentation also describes `isinclude` as the ISML mechanism for including content generated through a URL. See the SFRA Page controller and Salesforce ISML template documentation.

QUESTION NO: 12

A Digital Developer wants pass control to an ISML template from a JavaScript Controller and load product on the pipeline dictionary with the name myProduct.

Which code sample will achieve this?

A. ISML.renderTemplate ("helloworld.isml", { "myProduct": "product" });

B. ISML.renderTemplate ("helloworld.isml", { "product": myProduct });

C. ISML.renderTemplate ("helloworld.isml", { product: myProduct });

D. ISML.renderTemplate ("helloworld.isml", { myProduct: product });

ANSWER: D

Explanation:

`ISML.renderTemplate("helloworld.isml", { myProduct: product });` is correct because the second argument to `ISML.renderTemplate` is a JavaScript object whose properties are placed into the rendering context for the ISML template. The property name `myProduct` becomes the name exposed to ISML, while `product` is evaluated as the JavaScript variable containing the product object. Consequently, the template can access the supplied product through `myProduct`, for example with an ISML expression such as `${pdict.myProduct}` where pipeline-dictionary access is required by the template context.

This pattern cleanly separates the controller's responsibility of obtaining or preparing the product data from the template's responsibility of presenting it. The object passed at render time is specifically intended to provide values to the template under controlled, explicit names. The `dw.template.ISML` API documents `renderTemplate(template, parameters)` and identifies the parameters object as the source of values made available while the template is rendered. See the dw.template.ISML Script API reference and the B2C Commerce developer documentation.

QUESTION NO: 13

Universal Containers wants to change a content slot that is currently configured to display a content asset. Now they want the slot to display the top five selling boxes for the week.

Which two changes need to be madefor this to occur? (Choose two.)

A. Change the slot's configuration content type to "products."

B. Change the slot's configuration content type to "recommendations."

C. Change the slot's configuration template to the appropriate rendering template.

D. Delete the existing content asset.

ANSWER: B C

Explanation:

“Change the slot’s configuration content type to “recommendations.”” is required because weekly top-selling products are dynamic recommendation content, rather than a manually selected product list or a static content asset. In B2C Commerce, a content-slot configuration identifies the type of content that can be assigned to the slot. Using the recommendations content type enables the slot to use a recommendation source, such as a top-sellers recommendation, to supply the products displayed for the relevant period and context.

“Change the slot’s configuration template to the appropriate rendering template.” is also required. A content-slot configuration includes a rendering template that controls how the assigned content is rendered on the storefront. Once the slot supplies recommendation data, its template must be capable of iterating through the recommended products and rendering the desired five product boxes. The template therefore connects the recommendation results to the storefront presentation, including product names, images, prices, links, and any layout required by the site design.

Salesforce documents content-slot configuration and rendering concepts in the B2C Commerce Content Slots guide. Recommendation content and its storefront use are covered in the B2C Commerce Recommendations guide.

QUESTION NO: 14

Which method is efficient and scalable because it uses the product search index rather than searching the database?

A. ProducrIndexModel.getOrderableProductsOnly()

B. ProduccAvailabiliryModel.isOrderable()

C. ProductSearchModel.getProductSearchHits()

D. ProductVanari.cnMcciel.aerVariams ()

ANSWER: C

Explanation:

`ProductSearchModel.getProductSearchHits()` is the appropriate method because it retrieves the hits produced by a configured product search. In B2C Commerce, product searches are executed against the product search index, which is built and maintained for efficient storefront search, filtering, sorting, and category navigation. After search criteria are configured on a `ProductSearchModel` and the search is executed, `getProductSearchHits()` returns an iterator of `ProductSearchHit` objects. These hits can then be used to obtain product information for rendering search-result or product-listing pages.

Using search hits is scalable because the index is specifically optimized for query operations and can apply indexed attributes, refinements, sorting rules, and availability-related search settings without an application-level database scan. It also supports paging result sets, allowing a storefront to process only the subset of results needed for a page. Salesforce documents `ProductSearchModel` and its `getProductSearchHits()` method in the Script API: ProductSearchModel Script API. Additional implementation guidance for B2C Commerce product search is available in the B2C Commerce Search documentation.

QUESTION NO: 15

A job executes a pipeline that makes calls to an external system.

Which two actions prevent performance issues in this situation? (Choose two.)

A. Use synchronous import or export jobs

B. Configure a timeout for the script pipelet.

C. Disable multi-threading.

D. Use asynchronous import or export jobs.

ANSWER: B C

Explanation:

Configuring a timeout for the script pipelet ensures that a call which is delayed or unresponsive does not hold the job indefinitely. External dependencies can experience latency, connection failures, or outages, so a bounded timeout lets the pipeline fail predictably and allows error handling, logging, or a later retry strategy to take over. This protects job execution resources from being consumed by stalled requests.

Disabling multi-threading is also appropriate when a pipeline job calls an external system. It prevents concurrent job threads from making many simultaneous requests to the same remote dependency. This reduces contention for shared resources, avoids overwhelming a system that may have limited capacity or rate limits, and makes the job's interaction with the external service more controlled and reliable. Salesforce B2C Commerce job processing supports configuring execution behavior carefully, particularly where integrations introduce dependencies outside the Commerce environment. See Salesforce's B2C Commerce Job Framework documentation and B2C Commerce Pipelines documentation for job and pipeline implementation guidance.

QUESTION NO: 16 - (SIMULATION)

A developer is asked to write a log containing the ID and name of the product with a variable named myProduct.

Which snippet of code should be used?

ANSWER: See the explanation for the answer

Explanation:

Use `Logger.debug` with indexed placeholders and pass the product properties as arguments:

```
var Logger = require('dw/system/Logger');

Logger.debug('Product ID: {0}, Product Name: {1}', myProduct.ID, myProduct.name);
```

`{0}` is replaced by `myProduct.ID` and `{1}` is replaced by `myProduct.name`.

QUESTION NO: 17

A Digital Developer is tasked with setting up a new Digital Server Connection using UX Studio in their sandbox.

Which three items are required to accomplish this task? (Choose three.)

A. Instance Version

B. Instance Hostname

C. Business Manager Username

D. Keystore Password

E. Business Manager Password

ANSWER: A B D

Explanation:

To define a Digital Server Connection in UX Studio, the developer needs the instance-specific endpoint and compatibility information: **Instance Hostname** identifies the sandbox server that UX Studio must contact, while **Instance Version** tells the tooling which Commerce platform version and associated connection behavior to use. These values establish where the connection is directed and how UX Studio should communicate with that environment.

A **Keystore Password** is also required because UX Studio uses a local keystore to protect sensitive connection credentials. The password unlocks that protected store so the connection configuration can securely retain and use authentication information. This is distinct from providing Business Manager interactive login details as connection-creation fields. In

practice, developers should obtain the hostname and applicable instance version from their sandbox/environment details and choose a secure keystore password that can be retained according to team security procedures.

Salesforce's UX Studio documentation describes UX Studio as the Eclipse-based development environment used to connect to and work with B2C Commerce instances. See B2C Commerce UX Studio documentation and the B2C Commerce Developer Center for developer tooling and environment guidance.

QUESTION NO: 18

A merchant has complained to the developers that some products are not appearing in the storefront and has asked them to diagnose and solve the issue.

Which two factors might be causing a product to be hidden?

Choose 2 answers

A. Product has been set to searchable.

B. Product lacks a price.

C. Product does not have any images.

D. Product available to sell is less than 1.

E. Product is not online.

ANSWER: D E

Explanation:

A product that is not online can be hidden because the online flag controls whether the product is available for the current site and catalog context. Before a storefront can render a product as a purchasable, customer-facing item, the product must be online for the relevant time period. The Product API exposes this status through `isOnline()`, which reflects the product's online state.

A product with available-to-sell inventory below one can also be hidden when the storefront's product-search and availability configuration is set to show only orderable products. In B2C Commerce, the product availability model calculates whether inventory is available to fulfill an order. Search results and category listings can use that orderability information to omit products that cannot currently be ordered. This behavior is especially relevant where the merchant expects out-of-stock products to disappear rather than remain visible as unavailable.

Developers should therefore verify both the product's online status and its availability model, including inventory-list data, allocation, and the site preference governing whether only orderable products are shown. See the Product Script API and the ProductAvailabilityModel Script API.

QUESTION NO: 19

A developer is asked to improve the maintainability of a page by reducing its code repetition.

What are two techniques the developer should implement to achieve this?

Choose 2 answers.

A. Require and render templates with tags

B. Use local template includes

C. Implement template decorators paired with replace tags

D. Embed partial files using ISML expressions

ANSWER: B C

Explanation:

Use local template includes is correct because local includes enable an ISML template to insert a reusable template fragment during rendering. Common page elements such as headers, navigation, product tiles, forms, and shared content blocks can be maintained in one included template rather than duplicated across multiple page templates. This supports consistent markup and makes future changes more efficient.

Implement template decorators paired with replace tags is also correct because decorators provide a structured way to apply a common page layout or wrapper around page-specific content. A decorating template defines reusable outer content, while the `<isreplace/>` tag identifies where the content from the decorated template is inserted. This separates shared layout concerns from the unique content of individual pages, reducing repeated structural markup and improving consistency across the storefront. The pattern is particularly useful when many pages share the same shell but require different body content.

Salesforce documents local includes as a mechanism for reusing template content and template decorators as a layout-reuse pattern. See Local Includes and Template Decorators.

QUESTION NO: 20

A merchant uploads an image using the ContentImage Upload module of Business Manager.

Which three modules can the merchant or developer use to display the image on the Storefront?

Choose 3 answers

A. ISML templates

B. Content assets

C. Storefront catalogs

D. Content slots

E. Payment types

ANSWER: A B D

Explanation:

ISML templates, **Content assets**, and **Content slots** can display an image uploaded through the Content Image Upload area of Business Manager. An ISML template can directly render an image element and use the image's static URL, making the uploaded file available wherever the template is used. Content assets support rich HTML content, so a merchant can place an uploaded image into an asset's body and manage that reusable presentation content in Business Manager. Content slots provide the managed placement mechanism for storefront pages: a slot can be assigned content, including a content asset that contains the uploaded image, allowing merchants to control where that image-backed content appears without modifying page-template code. This separation supports both developer-controlled rendering through templates and merchant-controlled reusable content and page placement. Salesforce describes content assets and slots as core Content Management capabilities for delivering managed storefront content, while ISML is the server-side template technology used to render storefront markup. See the Salesforce documentation on managing B2C Commerce content and ISML templates.

QUESTION NO: 21

A Digital Developer adds the following line of code to a script.

```
dw.system.Logger.getLogger('login').debug("Login API has succeeded");
```

The code executes without error; however, the log file on disk does NOT contain the log message.

Which two actions should be completed to write the log message to disk? (Choose two.)

A. Ensure that the debug log level is enabled to write to file in the Custom Log Settings Business Manager module.

B. Archive old log files to make room in the log directory.

C. Ensure that the “login” category is added to the Custom Log Filters in the Log Settings Business Manager module.

D. Ensure that the debug log level has been added to the custom log level types in the Global Preferences business manager module.

ANSWER: C D

Explanation:

The logger call uses a custom category, `login`, and writes a message at the `debug` level. For a custom-category message to be emitted to a file, the category must be configured in Business Manager’s Custom Log Filters. Adding `login` makes that category eligible for logging under the site’s configured log-filter rules, allowing messages created through the corresponding `Logger` instance to be routed to the relevant log output.

The `debug` level must also be available in the configured custom log level types. Log-level configuration determines which severity levels are recorded. Once `debug` is included and enabled for the applicable logging configuration, the logger can persist the debug message instead of suppressing it due to the current threshold or configured level set. These settings work together: the category identifies the custom log stream and the log level authorizes the message severity for output. Salesforce’s Script API documents that `dw.system.Logger` supports named loggers and level-specific methods such as `debug()`; the Business Manager logging configuration controls whether those messages are written. See the dw.system.Logger Script API reference.

QUESTION NO: 22

A Digital Developer adds the following line of code to a script.

```
dw.system.Logger.getLogger('login').debug("Login API has succeeded");
```

The code executes without error; however, the log file on disk does NOTcontain the log message.

Which two actions should be completed to write the log message to disk? (Choose two.)

A. Ensure that the debug log level is enabled to write to file in the Custom Log Settings Business Manager module.

B. Archive old log files tomake room in the log directory.

C. Ensure that the “login” category is added to the Custom Log Filters in the Log Settings Business Manager module.

D. Ensure that the debug log level has been added to the custom log level types in the Global Preferences business manager module.

ANSWER: A C

Explanation:

Ensure that the debug log level is enabled to write to file in the Custom Log Settings Business Manager module, because a successful call to the Script API logger does not by itself guarantee persistence in a disk log. B2C Commerce controls whether each log severity is emitted to a file through the applicable custom log settings. When debug output is not configured for file output, the script can execute normally while the expected debug entry is absent from the log file.

Ensure that the “login” category is added to the Custom Log Filters in the Log Settings Business Manager module, because custom logger output is associated with a category and filtering determines which category-level messages are recorded. Adding the category to the configured filters allows messages generated under that logger category to be included in the log output. Together, the file-output setting for the debug severity and the category filter allow the debug message to reach disk.

Salesforce documents the logger API, including category-based loggers and log levels, in the dw.system.Logger Script API reference. Additional configuration guidance is available in Salesforce's B2C Commerce logging documentation.

QUESTION NO: 23

A merchant has a content slot on a page that currently displays products based on the top Sellers for the

current week.

They wish to change this functionality and, instead, have the slot render a specific content asset so that the

content experience is more personalized to the visitors.

Which two actions are necessary to make this change?

Choose 2 answers

A. Delete the existing content slot and create a new one.

B. Change the rendering template in the slot configuration

C. Change the default setting in the slot configuration

D. Change the content type for the slot configuration

ANSWER: B D

Explanation:

Changing the content type for the slot configuration is necessary because a slot configured to supply product content, such as a top-sellers recommendation, must be reconfigured to support a content asset. The content type determines the kind of content that Business Manager can assign to the slot configuration. Once the slot accepts content assets, the merchant can select the intended asset as the slot's assigned content for the applicable site, schedule, or campaign context.

Changing the rendering template in the slot configuration is also necessary because product content and a content asset have different data and presentation requirements. The rendering template controls how the slot's configured content is rendered on the storefront. A template intended for a product collection or recommendation result will not provide the appropriate rendering behavior for a single content asset. Assigning a content-asset-compatible template ensures the slot can retrieve and display the personalized asset correctly while preserving the existing slot placement on the page.

Salesforce Commerce Cloud describes content slots as reusable storefront locations whose configuration specifies the content type and rendering template. See Salesforce's Content Slots documentation and the Content Assets documentation.

QUESTION NO: 24

A Digital Developer wants to selectivelyretrieve products and process them from an iPhone.

Which action should the Developer take, given that JavaScript controllers CANNOT be used?

A. Use import/export in Business Manager.

B. Create a webservice to retrieve products.

C. Use OCAPI and invoke it in native language.

D. Use WebDAV Client to retrieve products.

ANSWER: C

Explanation:

Use OCAPI and invoke it in native language. The Open Commerce API provides REST-based endpoints that external applications, including an iPhone app written in Swift or Objective-C, can call directly over HTTP. A native mobile application can authenticate to the appropriate OCAPI resource, submit a product-search request with parameters that narrow the returned catalog data, and then process the JSON response locally on the device. This approach does not depend on storefront JavaScript controllers, because the mobile client communicates with the Commerce Cloud instance through its supported API layer rather than through an SFRA or SiteGenesis controller route.

OCAPI's Data API exposes product and product-search resources intended for programmatic access to Commerce Cloud data. Access is controlled through OCAPI settings, where the instance and client permissions specify the resources, methods, and data that the application may use. The developer should therefore configure a client ID and the required read permissions, then invoke the permitted product resource from the iPhone application's native networking code. See the Salesforce B2C Commerce OCAPI overview and the OCAPI Data API documentation.

QUESTION NO: 25

A Digital Developer creates a B2C Commerce server connection in their UX Studio workspace. The Developer adds new cartridges to the workspace, but the cartridges do NOT execute as the Developer expects.

Which three things should the Digital Developer verify to ensure the cartridges are uploaded? (Choose three.)

A. The Auto-Upload setting is enabled for the server connection.

B. The Active Server setting is enabled for the server connection.

C. The credentials for the server connection are correctly entered.

D. The cartridge is for the current version of B2C Commerce.

E. The server is configured to accept incoming connections.

ANSWER: A B C

Explanation:

For UX Studio to upload cartridge code to a B2C Commerce instance, the server connection must be usable by the workspace and selected for deployment. **The Auto-Upload setting is enabled for the server connection.** ensures that saved changes in the workspace are automatically transferred to the configured instance instead of requiring a manual upload. **The Active Server setting is enabled for the server connection.** identifies that connection as the active target used by UX Studio for operations such as automatic cartridge uploads. **The credentials for the server connection are correctly entered.** is also essential because UX Studio must authenticate to the instance's code-upload endpoint before it can transfer cartridge files.

These settings work together: an active target determines where the code is sent, auto-upload determines that workspace changes are sent automatically, and valid credentials authorize the transfer. After deployment, cartridge execution additionally depends on the relevant cartridge path being configured for the site, but that is separate from verifying whether the cartridge files were uploaded. Salesforce's B2C Commerce documentation describes cartridges as deployable units of storefront and business functionality and provides development guidance for using UX Studio and server connections. See the B2C Commerce cartridges documentation and the UX Studio documentation.

QUESTION NO: 26

Universal Containers wants to associate a region code value with an order to indicate the general area of its destination. This region code must be accessible whenever the order history is displayed.

What is required to accomplish this?

A. Store the region code value in a session variable.

B. Define a custom attribute on the Order system object type to store the region code value.

C. Define a custom object type to store the username with the region code.

D. Store the region code value in the geolocation system attribute of the Order.

ANSWER: B

Explanation:

Define a custom attribute on the Order system object type to store the region code value. Order data is persisted after checkout and is the appropriate business object for information that belongs to an individual order and must remain available for later uses, including order-history rendering. In Business Manager, a developer creates the attribute definition for the Order system object type, typically using an appropriate data type such as String or Enum, and then assigns the region value to the order's custom attribute during order processing. The stored value can subsequently be read from the order object when order history is displayed, such as through `order.custom.regionCode`.

Salesforce B2C Commerce custom attributes extend system objects without changing the underlying platform data model. This supports a durable, order-specific field that is available across requests and customer sessions, which is required for historical order information. Salesforce's documentation describes custom attributes as the mechanism for adding application-specific data to system objects and explains their use through the `custom` property in script. See Salesforce B2C Commerce custom attributes documentation and the Order Script API reference.

QUESTION NO: 27

Given the customer basket described below:

A customer has an existing basket that consists of multiple items.

One of the items is identified as a gift ítem by an attribute at the product line ítem.

The developer needs to write custom code to fetch the customer basket and then modify the basket based

upon the items in the cart. If the basket contains any gift items, modify the basket and create a separate

shipment for the gift item.

Four hooks are required to make the modification, beginning with modifyGETRespone and ending with

validatebasket.

Dw.ocapi.shop.basket.modifyGETResponse

-- missing hook –

-- missing hook --

dw.ocapi.shop.basket.validateBasket

What are the two missing hooks in the middle?

A. dw.ocapi.shop.basket.shipment.afterDELETE

B. dw.ocapi.shop.basket.shipment.beforePATCH

C. dw.ocapi.shop.basket.shipment.beforeDELETE

D. dw.ocapi.shop.basket.shipment.beforePOST

ANSWER: B D

Explanation:

The required shipment hooks are `dw.ocapi.shop.basket.shipment.beforePOST` and `dw.ocapi.shop.basket.shipment.beforePATCH`. A separate shipment must first be created for the gift line item, and the `beforePOST` extension point provides the appropriate point in the OCAPI shipment-creation lifecycle to apply custom basket logic before that shipment is persisted. Once the shipment exists, basket or shipment data can be updated to

associate the relevant gift item and shipping information with it. The `beforePATCH` extension point supports that update stage before the requested shipment changes are applied.

These hooks work with the basket response customization and basket validation lifecycle: `modifyGETResponse` can shape the basket representation returned by the GET operation, while `validateBasket` ensures the resulting basket remains valid after the custom shipment logic has been applied. Using the shipment-specific POST and PATCH hooks keeps the customization aligned with the OCAPI resource operation being performed and allows the implementation to place gift items into their own shipment consistently. Salesforce documents OCAPI hook extension points and their naming conventions in the OCAPI hooks documentation and describes basket and shipment APIs in the B2C Commerce OCAPI guide.

QUESTION NO: 28

A client that sells to multiplecountries in Europe needs to disable Apple Pay for Denmark.

Which Business Manager module is used to achieve this requirement?

A. Locale Payments

B. Payment Methods

C. Payment Processors

D. Apple Pay

ANSWER: A

Explanation:

Locale Payments is the Business Manager module used to control which payment methods are available for each locale. In a multi-country storefront, the merchant can configure the Denmark locale and exclude Apple Pay from the payment methods presented to shoppers in that locale. This supports country-specific payment availability while allowing the same payment method to remain enabled for other locales and markets where it is accepted.

Locale-specific configuration is important because a storefront's supported payment choices can differ based on local commercial requirements, customer expectations, and payment-provider availability. The setting is applied at the locale level, so the storefront can determine the appropriate payment options from the shopper's active locale during checkout. Apple Pay can therefore be retained as a configured payment capability for the site while being unavailable when Denmark is the selected locale. Salesforce's Commerce B2C documentation covers payment-method configuration and Business Manager administration in its Payment Methods guide and B2C Commerce documentation.

QUESTION NO: 29

Adigital instance has one site, with one master product catalog separate from the site catalog. Some, but NOT all, products in the master catalog are assigned to categories of the site catalog.

Using Business Manager, how can a Digital Developer create a catalog export file that contains only the products assigned to the site catalog?

A. Use the Catalog Export module to export the site catalog.

B. Use the Catalog Export module to export the master catalog, with a category-assignment search to export specific products.

C. Use the Site Import & Export module to export both the site catalog and the master catalog in a single archive.

D. Use the Site Import & Export module to export the master catalog, filtered by site catalog categories to export specific products.

ANSWER: B

Explanation:

Use the Catalog Export module to export the master catalog, with a category-assignment search to export specific products. Product records are maintained in the master catalog, while the site catalog provides the storefront category structure and product-to-category assignments. Therefore, the export must use the master catalog as its source so that the resulting XML includes the actual product definitions rather than only catalog-category structure and assignments.

In Business Manager's Catalog Export module, a category-assignment search can constrain the export to products assigned to categories in the site catalog. This produces a targeted master-catalog export containing the subset of products that are available through the site catalog, without including unrelated master-catalog products. The approach is particularly appropriate when product data needs to be moved or integrated while preserving the distinction between shared product information in the master catalog and site-specific merchandising organization in the storefront catalog.

Salesforce describes the relationship between master and storefront catalogs, including product assignment to storefront categories, in the B2C Commerce Catalogs documentation. See also the Catalog import and export documentation for Business Manager catalog data-transfer guidance.

QUESTION NO: 30

A Digital Developer added a file named MyBusinessController.js in the cartridge named app_project. The project design calls for this new file to override MyBusinessController.js in client_project. The client_project cartridge contains other necessary functionality. Additional functionality is also included in the storefront_core and storefront_controllers cartridges.

Which cartridge path meets the project requirements?

A. client_project:app_project:storefront_controllers:storefront_core

B. app_project:storefront_controllers:storefront_core

C. app_project:client_project:storefront_controllers:storefront_core

D. storefront_core:storefront_controllers:client_project:app_project

ANSWER: C

Explanation:

`app_project:client_project:storefront_controllers:storefront_core` is correct because B2C Commerce resolves cartridge resources according to the configured cartridge-path order. When files with the same relevant name and location are available in multiple cartridges, the cartridge listed first takes precedence. Placing `app_project` before `client_project` therefore ensures that its `MyBusinessController.js` implementation is selected as the override.

Keeping `client_project` immediately after `app_project` is also essential: functionality that is present only in `client_project` remains available for resolution. The remaining cartridges stay on the path afterward so their required shared storefront functionality can still be found when it is not supplied by either project-specific cartridge. This ordering supports a common overlay architecture: custom project cartridges are placed earlier, while cartridges providing base or fallback behavior follow them. Salesforce documents that the cartridge path controls the order in which cartridges are searched and that the path is evaluated from left to right. See the B2C Commerce cartridges documentation and the cartridge path documentation.

QUESTION NO: 31

Which code sample is required to use a custom tag provided in SiteGenesis in an ISML template?

A

```
<isinclude template="util/modules">
```

B

```
<isinclude template="util/customtags">
```

C

```
<isinclude url="${URLUtils.url("CustomTag-Start")}>
```

D

```
<isscript>
  dw.util.CustomTagMgr.createCustomTag("mytag");
<isscript>
```

A. Option A

B. Option B

C. Option C

D. Option D

ANSWER: B

Explanation:

The selected code sample is correct because SiteGenesis custom tags are made available to an ISML template through an `<ismodule>` declaration. This declaration maps a custom tag name to the ISML template that implements the tag and can declare the attributes that the tag accepts. After the module is declared, the template can invoke the named custom tag and pass the required attributes; the rendering engine resolves that invocation to the mapped template. This mechanism lets SiteGenesis and custom cartridges package reusable presentation fragments as semantic ISML tags rather than repeatedly writing direct include logic throughout storefront templates. The module declaration must be available in the template context before the custom tag is used, and its template path must identify the reusable ISML template supplied by the cartridge. Salesforce documents `ismodule` as the ISML element used to define custom tags and specifies that the declaration includes the implementation template and tag name. See the Salesforce Commerce Cloud ISML documentation and the ismodule reference for the custom-tag declaration and usage model.

QUESTION NO: 32

A developer wants to use an external application to manage their stores information (such as opening hours, and so on), and see their changes in their B2C Commerce Staging instance aas son as they are saved.

What is the appropriate technique the developer should perform to allow the merchant to create a new store in this scenario?

A. A POST request to the Stores Data OCAPI.

B. A PUT request to the Stores Data OCAPI.

C. A PATCH request to the Stores Data OCAPI.

D. An UPDATE request to the Stores Data OCAPI.

ANSWER: B

Explanation:

A PUT request to the Stores Data OCAPI is the appropriate technique because the B2C Commerce Open Commerce API uses a known store identifier in the resource URL when creating or replacing a store resource. The external application supplies the store ID and sends the complete store representation, including details such as its name, address, hours, and other supported store attributes, to the Stores Data API endpoint. If that store ID does not yet exist in the target Staging instance, the PUT operation creates it; if it already exists, the operation updates the resource with the submitted representation. This idempotent behavior is particularly useful for an external store-management system because it can safely synchronize a store by identifier without first needing separate create-versus-update logic. The integration must also be authorized through the instance's OCAPI settings, with the required Data API resource permissions enabled for the client ID and relevant site context. Salesforce documents the Stores resource and its supported operations in the B2C Commerce OCAPI Data API Reference. General OCAPI configuration and client-permission requirements are described in the B2C Commerce OCAPI documentation.

QUESTION NO: 33

A client has a requirement to allow users on the Storefront to filter by a newly created attribute.

After creating the search refinement, what else is necessary to achieve this?

A. Ensure the attribute has data and is indexed

B. Set the attribute as Searchable.

C. Change the productsearchrefinebar.isml template.

ANSWER: A

Explanation:

Ensure the attribute has data and is indexed is correct because a search refinement can only present values that are available in the product-search index. Creating the refinement defines how Business Manager should expose the attribute as a navigable filter, including its display name, refinement type, and ordering. However, the refinement has no usable values until products contain values for that attribute and the search index has processed those products.

Assigning attribute values to the relevant products supplies the data from which Commerce Cloud derives the refinement buckets or values. Rebuilding or updating the product search index then makes those values available to storefront search requests. Once indexed, the standard storefront search model can return the refinement information and the storefront can display it to shoppers, subject to the site catalog, category context, and refinement configuration. This is why both populated product data and indexing are essential operational steps after configuration. Salesforce documents that product search and navigation use the search index and that refinements are returned through the product search model: Salesforce B2C Commerce Search Refinements and ProductSearchModel API Reference.

QUESTION NO: 34

Which three operations should be done in a controller?

Choose 3 answers

A. Generate the response as JSON or HTML

B. Use the Script API to generate data for the view.

C. Use middleware functions when applicable

D. Create a plain JavaScript object representing a system object

E. Use the model needed for the view.

ANSWER: A C E

Explanation:

In Storefront Reference Architecture, a controller is responsible for handling an HTTP request and coordinating the work needed to produce the response. “Generate the response as JSON or HTML” is a controller responsibility because route handlers use the response object to render an ISML template or return JSON to the client. A controller should also “Use middleware functions when applicable.” Middleware provides reusable processing around a route, such as enforcing HTTPS, validating HTTP methods, checking customer authentication, or attaching consent-tracking behavior. This keeps common request-handling concerns consistent across routes.

“Use the model needed for the view.” is also correct because controllers obtain an appropriate model to prepare view-ready data, then pass that model to the renderer. SFRA models encapsulate presentation-oriented transformations of Commerce data, allowing templates to consume a stable, purposeful object rather than requiring templates to navigate raw platform objects. In this pattern, the controller remains focused on routing, orchestration, model selection, and response generation. Salesforce’s SFRA documentation describes controllers as the request/response layer and documents the role of route middleware and rendering behavior. See SFRA Controllers and SFRA Features.

QUESTION NO: 35

A merchant has a requirement to render personalized content ton a category page via a Content Slot that

targets VIP high-spending customers during a specific promotional period.

Which two items should the developer create to achieve the specified requirements?

Choose 2 answers:

A. VIP Customer Group

B. Page Template

C. Slot Configuration

D. Rendering Template

ANSWER: A C

Explanation:

VIP Customer Group is required to define the audience of high-spending customers who are eligible to receive the personalized slot content. In B2C Commerce, customer groups provide the customer-targeting mechanism used for personalized experiences. Customers can qualify for a group through rules or explicit assignment, allowing the merchant to identify the VIP audience consistently.

Slot Configuration is required because it controls what a content slot renders under defined conditions. A slot configuration can associate the category-page slot with the applicable content and apply targeting criteria, including the VIP customer group and the promotional start and end dates. When the storefront renders the content slot, B2C Commerce evaluates the configuration’s active schedule and customer-group targeting, then serves the configured content when the shopper qualifies during that period.

Together, the customer group supplies the audience definition, while the slot configuration supplies the scheduled and targeted delivery rules for the category-page content slot. Salesforce’s B2C Commerce documentation describes content slots and their configurable content-delivery behavior, as well as customer groups used for customer segmentation. See the B2C Commerce Content Slots guide and the B2C Commerce Customer Groups guide.

QUESTION NO: 36

A Digital Developer must resolve a performance issue with product tiles. The Developer determines that the product tiles are NOT being cached for a long enough period.

Which two methods can the Developer use to verify the cache settings for the product tiles? (Choose two.)

A. Enable cache information in the storefront toolkit and view the cache information for the product tile.

B. View the cache information provided by the Merchant Tools > Technical Reports Business Manager module.

C. View the product list page cache settings provided in the Administration > Manage Sites Business Manager module.

D. Enable the template debugger to verify the cache times for the producttile.isml template.

ANSWER: A D

Explanation:

Enabling cache information in the storefront toolkit and viewing the cache information for the product tile is a direct way to inspect caching behavior in the rendered storefront. The Storefront Toolkit can expose page and component-level diagnostic information while a developer is viewing the relevant product-listing page, making it useful for confirming the effective cache status and duration associated with a tile response.

Enabling the template debugger to verify the cache times for the `producttile.isml` template is also appropriate because product-tile output is generated through an ISML template. The template debugger provides visibility into the templates involved in rendering a request and their cache directives. This lets the developer identify the cache duration configured by the template that renders the tile and confirm whether it meets the intended performance strategy. These two approaches complement one another: one verifies behavior in the storefront response, while the other examines the template-level configuration responsible for that behavior. Salesforce documents storefront caching concepts and cache configuration in its B2C Commerce caching guide, and provides developer tooling details in the Storefront Toolkit documentation.