Adobe

AD0-E709 Exam

Adobe Commerce Developer Expert

Question: 1

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

```
class MyObserver implements ObserverInterfece
{
    public function execute(\Magento\Framework\Event\Observer $observer)
    {
        $config = $observer->getData('config');
        $filePath = $config->getData('import_fees');
        /** @var \My\Module\Model\ImportFeed $importFees */
        $importFees = $this->importFeesFactory->create();
        $importFees->uploadAndImport($filePath);
    }
}
```

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

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

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

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

    ...

</field>
```

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

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

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

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

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

Answer: B, C

Explanation:

Question: 2

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

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

B)

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

C)

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

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

Answer: B

Explanation:

Question: 3

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 filedset 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 validation?

A)

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>
```

B)

Add the validations within the `MyVendor\MyModule\Controller\Adminhtml\CustomEntity\UploadPdf` controller:

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

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: A

Explanation:

Question: 4

A product has some stock and quantity in the Sources panel in its edit view m the admin:

But when trying to add this product to the cart on the frontend. the following error is displayed:
"The requested qty is not available".
Why was this error received?

A. The sales channel tor the current website is disabled.
B. The total quantity of this product in all active quotes has reached its available stock.
C. The number of reservations for this product has already reached its available stock.

Answer: C

Explanation:

Question: 5

An Adobe Commerce Developer is tasked with creating a custom form which submits its data to a (rontend controller. They have decided to create an action and have implemented the\magnto\Framework\App\Action|HttpPostInterface class, but are not seeing the data being persisted in the database, and an error message is being shown on the frontend after submission.
After debugging and ensuring that The data persistence logic is correct, what may be cause and solution to this?

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

Answer: B

Explanation: