## New Module Install

#php bin/magento setup:upgrade --keep-generated or

#php bin/magento setup:upgrade

#php bin/magento indexer:reindex

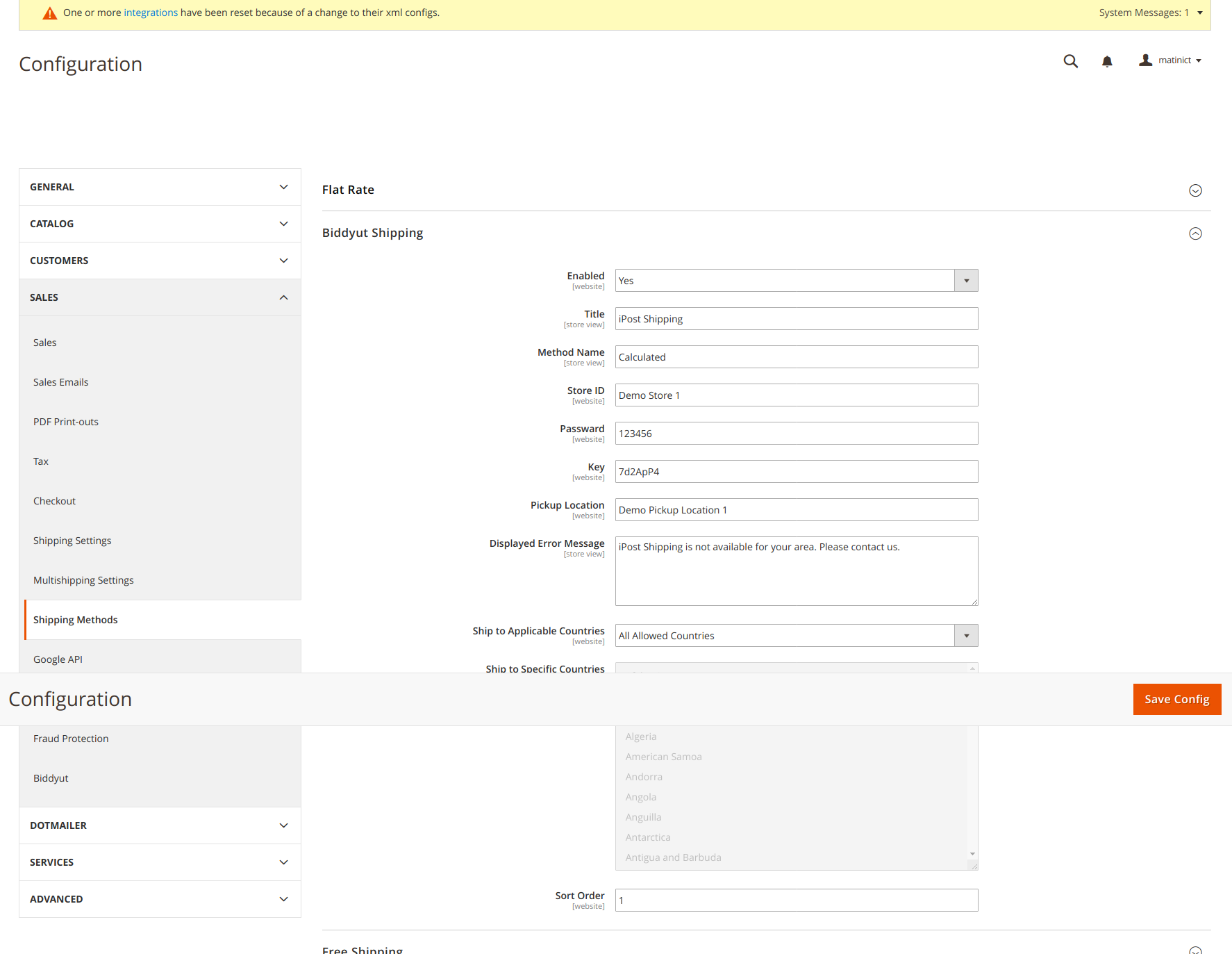
#php bin/magento cache:flush

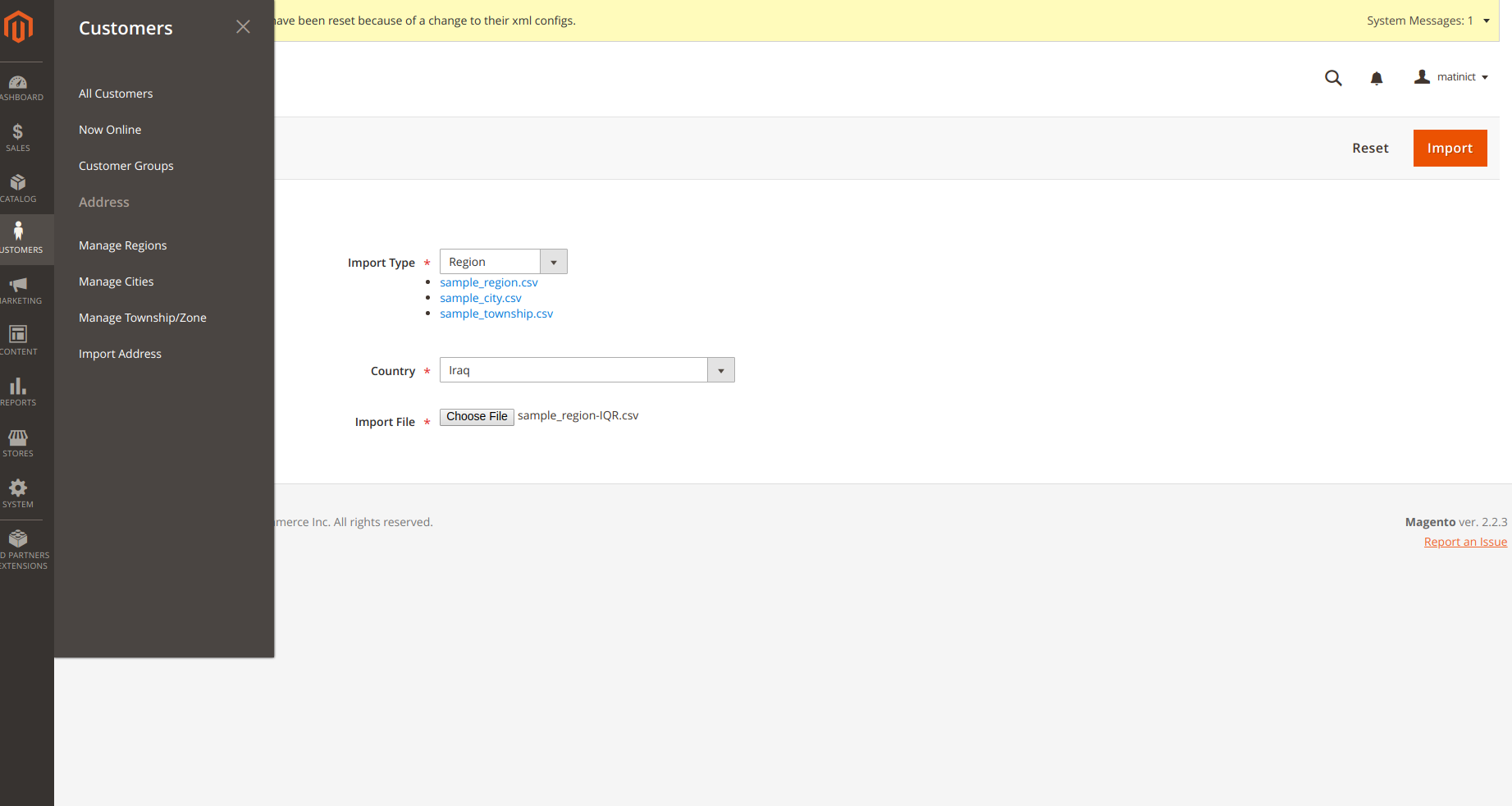
#php bin/magento cache:clean

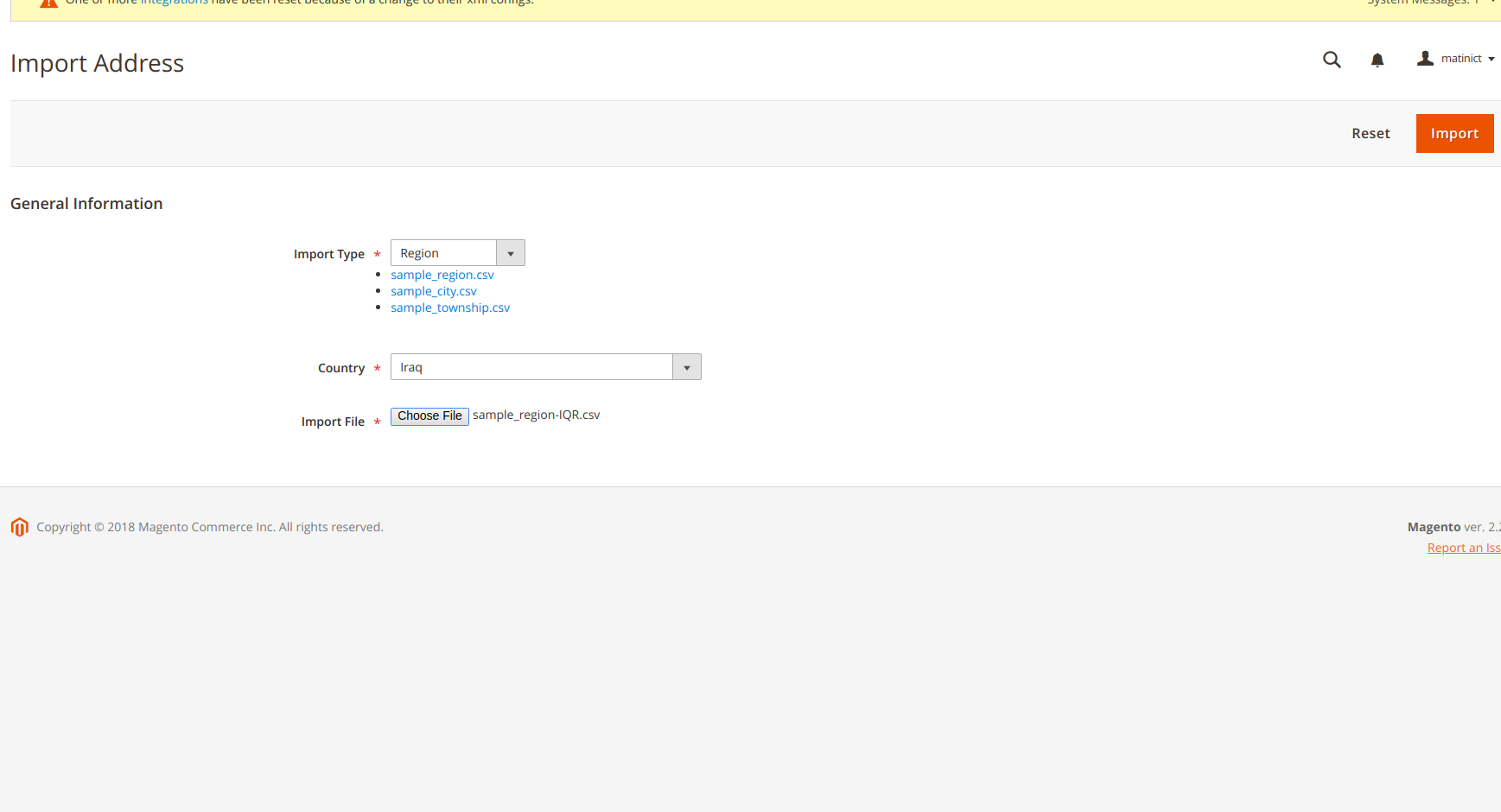
#sudo chmod 777 -R generated/\* var/\* pub/\*

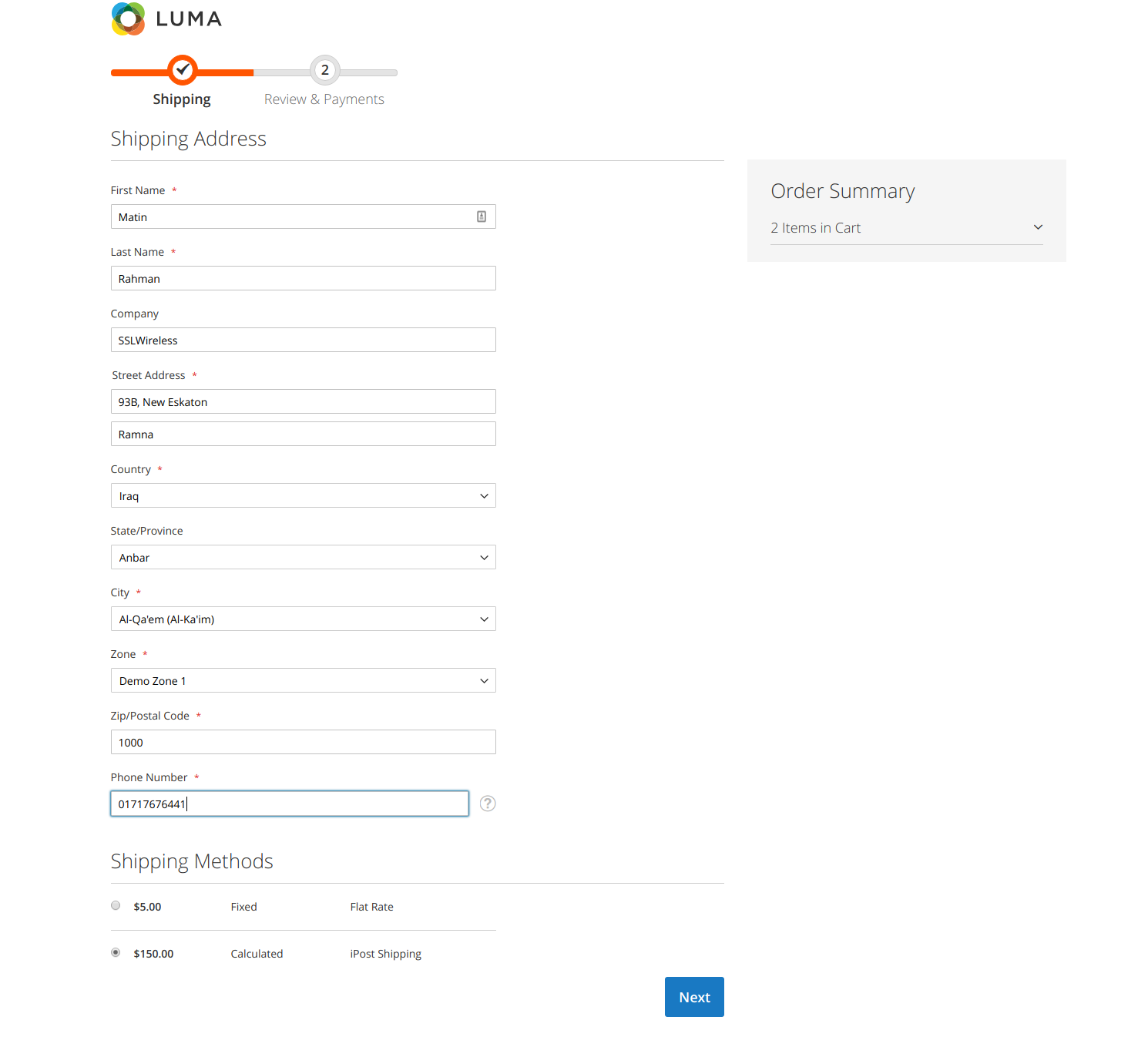
#rm var/cache/\* generated/code/\* var/view\_preprocessed/\* var/page\_cache/\* pub/static/frontend/\* pub/static/adminhtml/\* -R

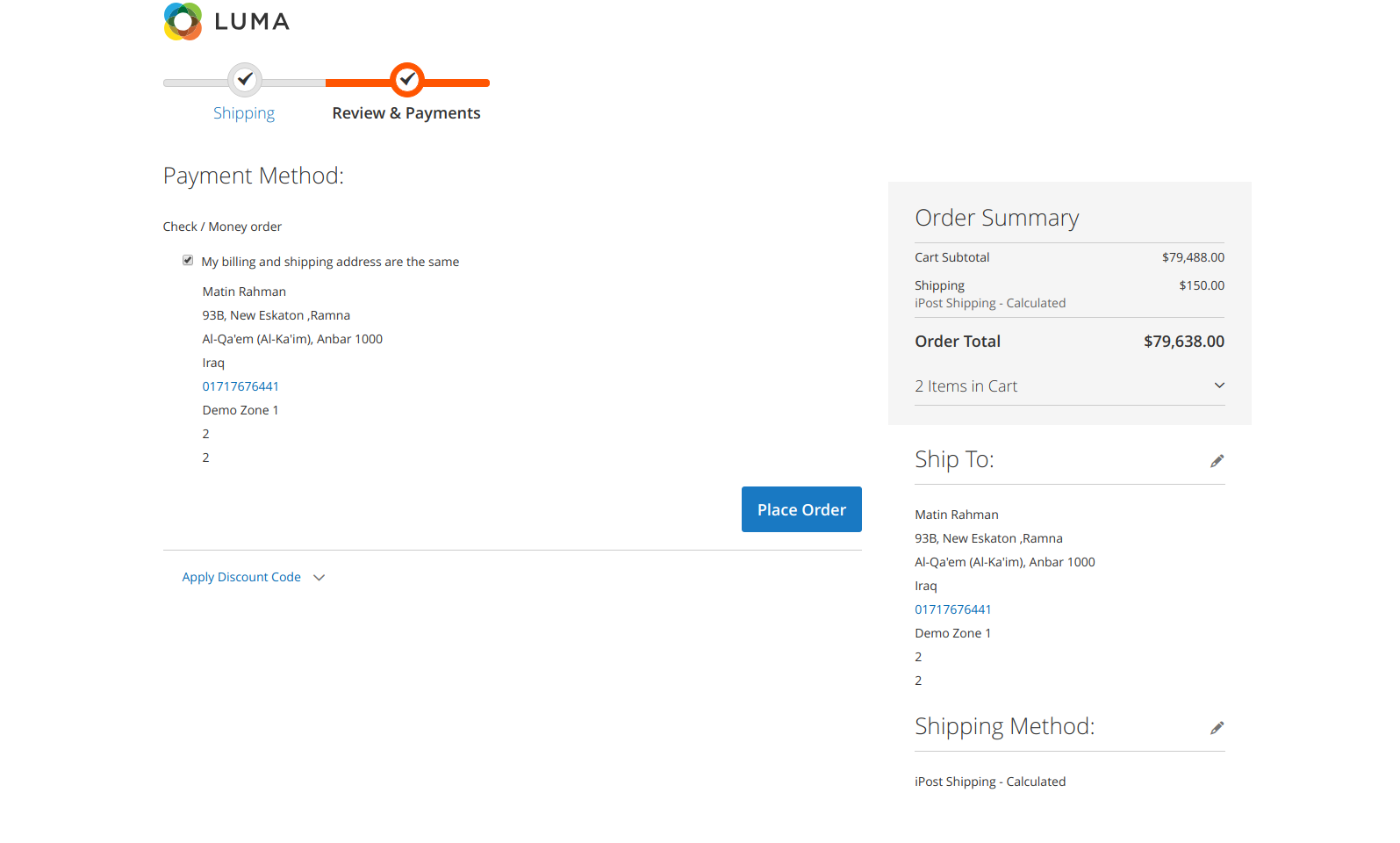
\*\*\* Magento 2.2.3 Need to Update temando vendor module , plz check

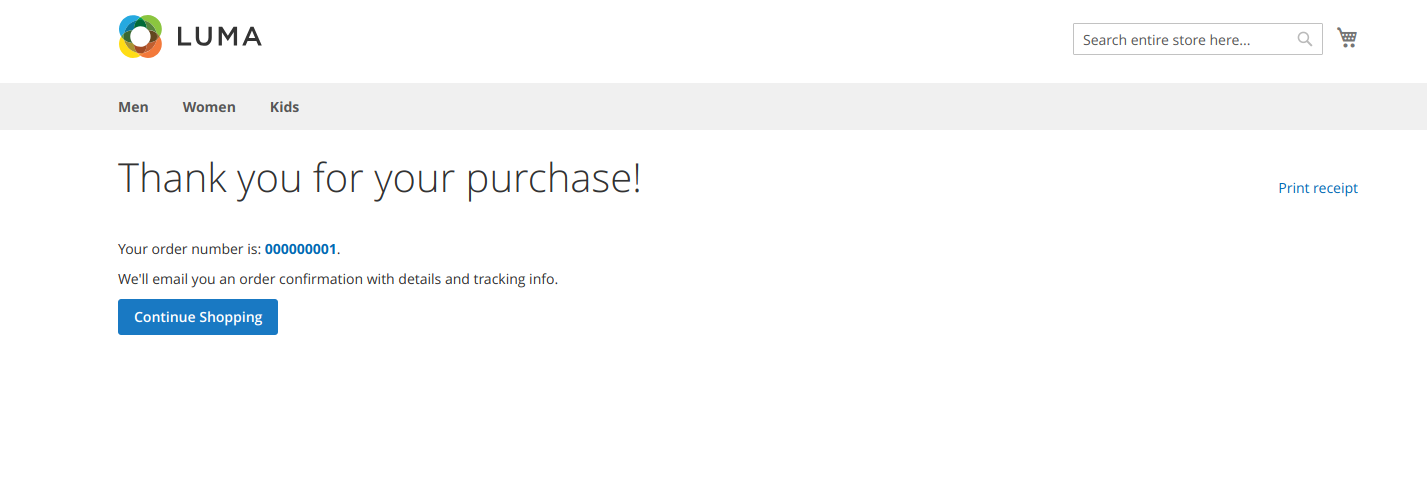


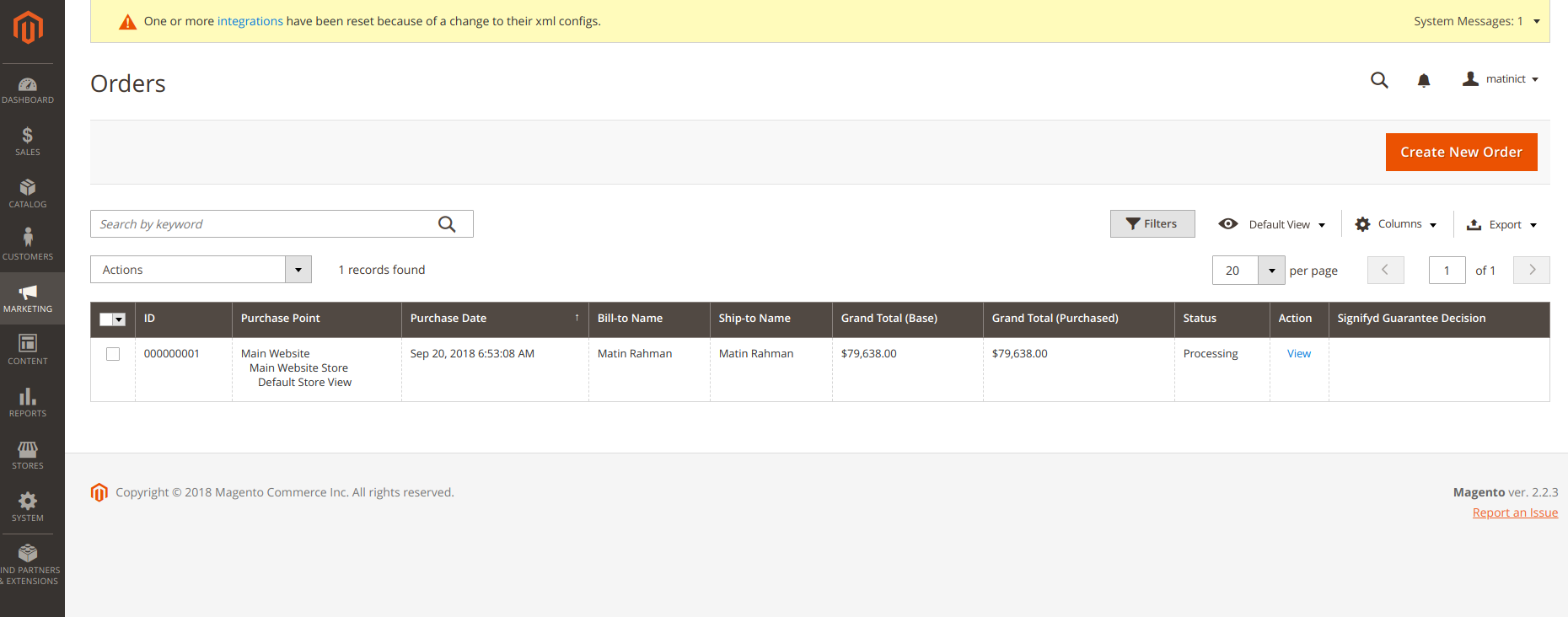


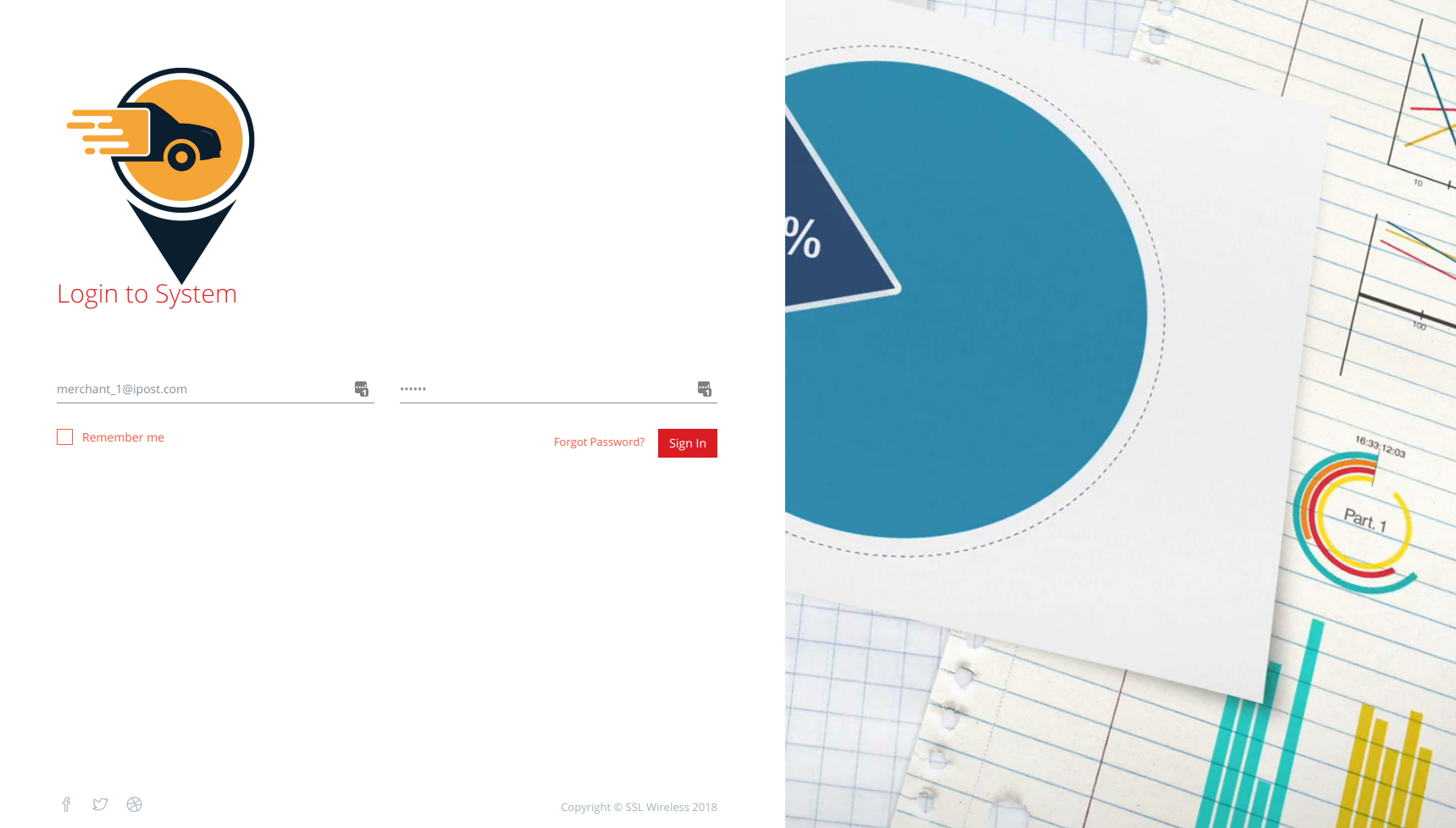


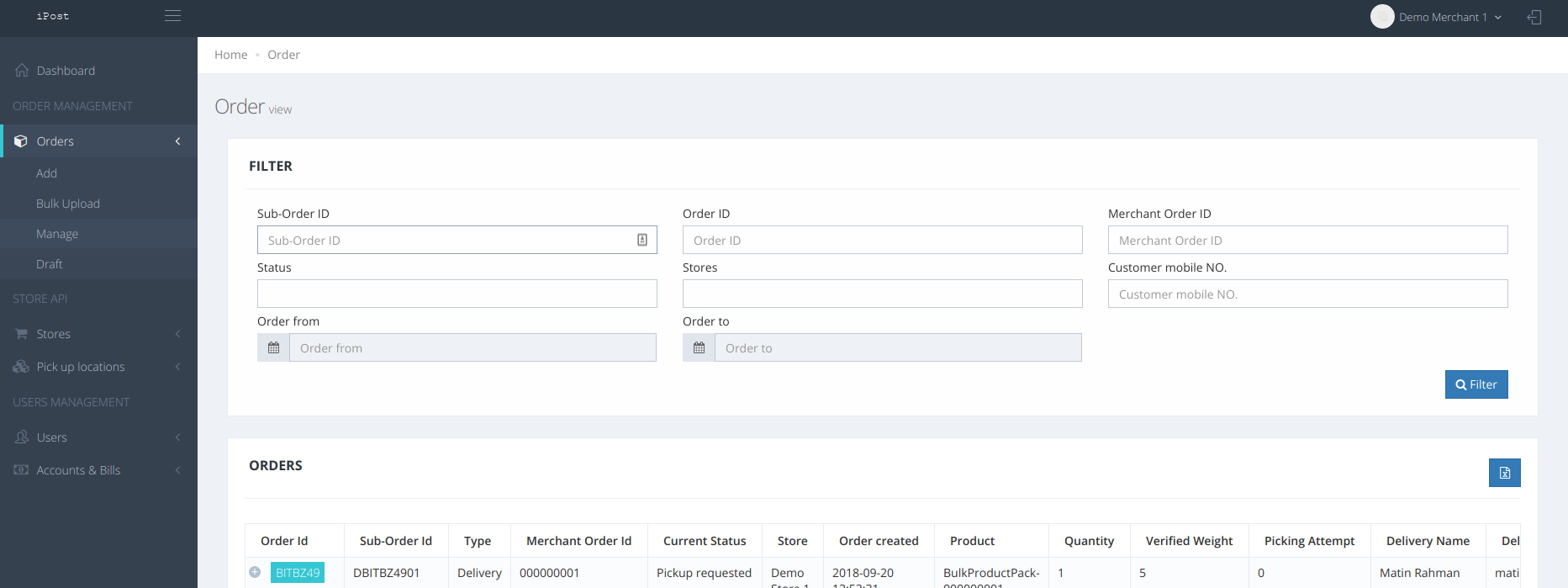


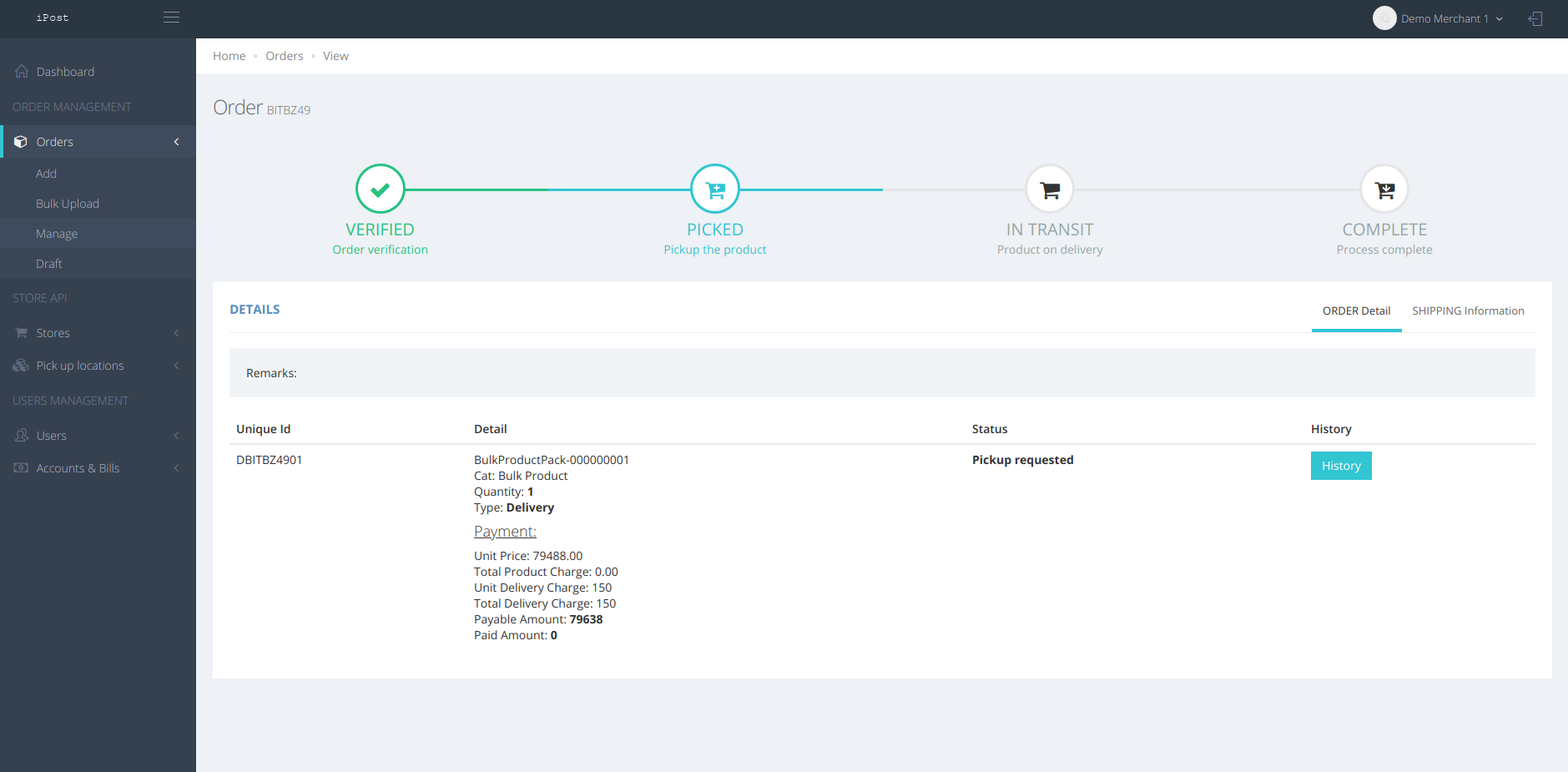


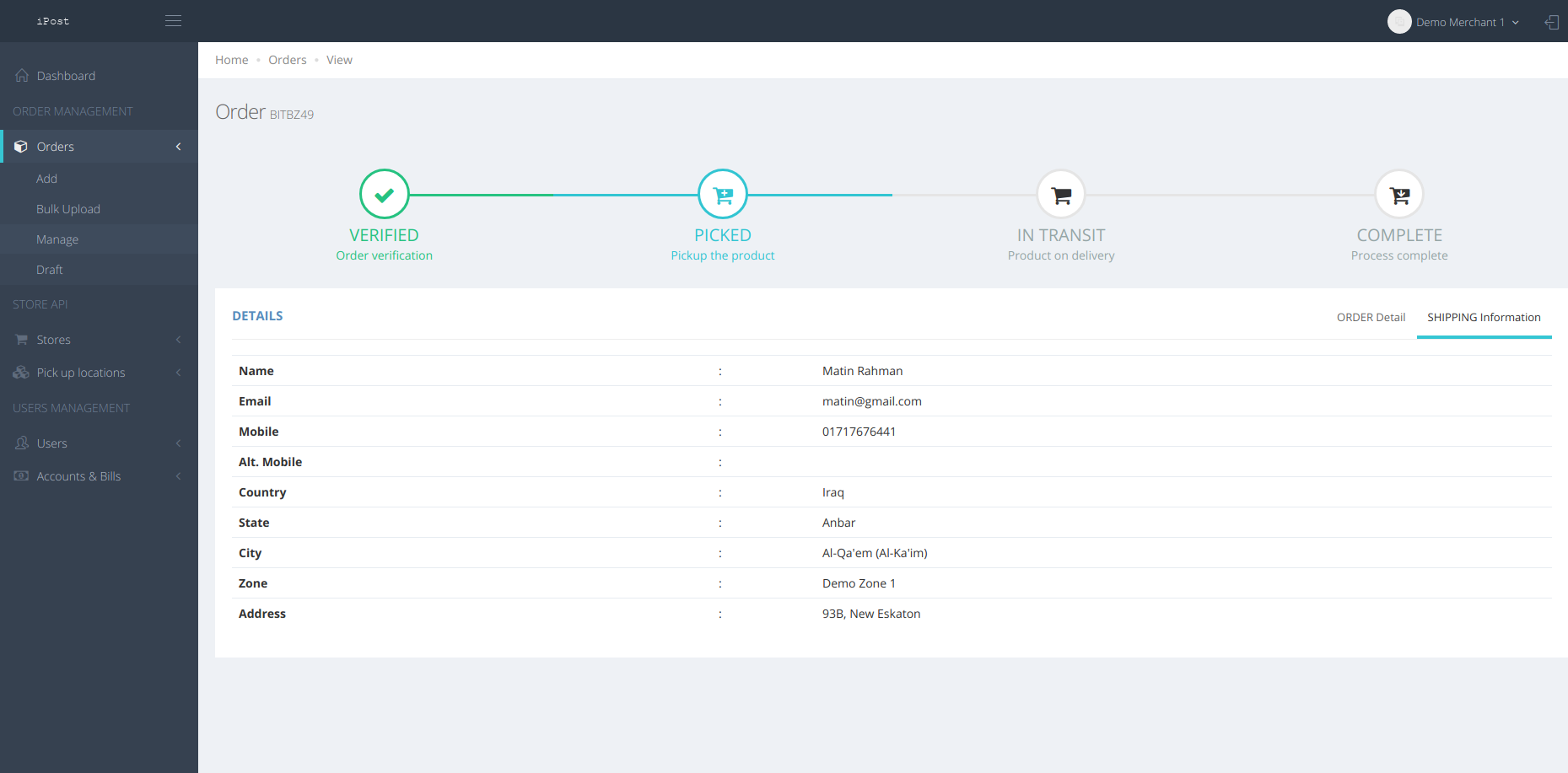












Issue:

\*\*\* Magento 2.2.3 Need to Update temando vendor module , plz check

Vendor/temando/module-shipping-m2/etc/events.xml updated as below:

<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="urn:magento:framework:Event/etc/events.xsd">

<event name="sales\_quote\_address\_save\_before">

<observer name="temando\_persist\_checkout\_fields" instance="Temando\Shipping\Observer\SaveCheckoutFieldsObserverBefore"/>

</event>

<event name="sales\_quote\_address\_save\_after">

<observer name="temando\_persist\_checkout\_fields" instance="Temando\Shipping\Observer\SaveCheckoutFieldsObserver"/>

</event>

<event name="sales\_quote\_address\_load\_after">

<observer name="temando\_load\_checkout\_fields" instance="Temando\Shipping\Observer\LoadCheckoutFieldsObserver"/>

</event>

<event name="sales\_order\_save\_after">

<observer name="temando\_manifest\_order" instance="Temando\Shipping\Observer\ManifestOrderObserver"/>

</event>

<event name="sales\_quote\_address\_collect\_totals\_before">

<observer name="temando\_load\_checkout\_fields" instance="Temando\Shipping\Observer\LoadCheckoutFieldsObserver"/>

</event>

</config>

Vendor/temando/module-shipping-m2/Observer/SaveCheckoutFieldsObserverBefore.php updated as below:

<?php

/\*\*

\* Refer to LICENSE.txt distributed with the Temando Shipping module for notice of license

\*/

namespace Temando\Shipping\Observer;

use Magento\Framework\Event\Observer;

use Magento\Framework\Event\ObserverInterface;

use Magento\Framework\Exception\NoSuchEntityException;

use Temando\Shipping\Api\Data\Checkout\AddressInterface;

use Temando\Shipping\Api\Data\Checkout\AddressInterfaceFactory;

use Temando\Shipping\Model\ResourceModel\Repository\AddressRepositoryInterface;

/\*\*

\* Save checkout fields with quote shipping address.

\*

\* @package Temando\Shipping\Observer

\* @author Christoph Aßmann <christoph.assmann@netresearch.de>

\* @license http://opensource.org/licenses/osl-3.0.php Open Software License (OSL 3.0)

\* @link http://www.temando.com/

\*/

class SaveCheckoutFieldsObserverBefore implements ObserverInterface

{

/\*\*

\* @var AddressRepositoryInterface

\*/

private $addressRepository;

/\*\*

\* @var AddressInterfaceFactory

\*/

private $addressFactory;

/\*\*

\* SaveCheckoutFieldsObserver constructor.

\* @param AddressRepositoryInterface $addressRepository

\* @param AddressInterfaceFactory $addressFactory

\*/

public function \_\_construct(

AddressRepositoryInterface $addressRepository,

AddressInterfaceFactory $addressFactory

) {

$this->addressRepository = $addressRepository;

$this->addressFactory = $addressFactory;

}

/\*\*

\* @param Observer $observer

\* @return void

\*/

public function execute(Observer $observer)

{

if($address = $observer->getEvent()->getQuoteAddress()){

if($attributes = $address->getExtensionAttributes()){

$checkoutFields = $attributes->getCheckoutFields();

if(is\_null($checkoutFields)){

$attributes->setCheckoutFields(array());

}

}

}

// /\*\* @var \Magento\Quote\Api\Data\AddressInterface|\Magento\Quote\Model\Quote\Address $quoteAddress \*/

// $quoteAddress = $observer->getData('quote\_address');

// if ($quoteAddress->getAddressType() !== \Magento\Quote\Model\Quote\Address::ADDRESS\_TYPE\_SHIPPING) {

// return;

// }

//

// if (!$quoteAddress->getExtensionAttributes()) {

// return;

// }

//

// // persist checkout fields

// try {

// $checkoutAddress = $this->addressRepository->getByQuoteAddressId($quoteAddress->getId());

// } catch (NoSuchEntityException $e) {

// $checkoutAddress = $this->addressFactory->create(['data' => [

// AddressInterface::SHIPPING\_ADDRESS\_ID => $quoteAddress->getId(),

// ]]);

// }

//

// $extensionAttributes = $quoteAddress->getExtensionAttributes();

// $checkoutAddress->setServiceSelection($extensionAttributes->getCheckoutFields());

// $this->addressRepository->save($checkoutAddress);

}

}