Znode®

relevant ecommerce™

SITECORE MULTISTORE/INCREMENTAL IMPORT USER GUIDE 8.1.2

April 2016

relevant ecommerce™

Sitecore Multistore/Incremental Import User Guide

Multifront Version 8.1.2

PREPARED BY:

Znode LLC

234 W. Florida St. Suite #304, Milwaukee, WI 53204

www.znode.com

Sitecore Multistore/Incremental Import User Guide

The changes below have been implemented to account for the following features:

- Sitecore – Multistore
- Incremental Import

Multistore

Multistore allows administrators the ability to manage multiple stores in Sitecore. With the 8.0.0 release, a second start item, NutWholesalerHome, will be added to the content tree. This also includes a second demo store which requires some minor configuration to render. Up to 20 start items can be added to the Sitecore content tree.

Also, included in this feature is the ability to either import data for all stores in Sitecore or for a specific store. The Znode toolbar in Content Editor has been updated to support this update.

Minimum Requirements in Znode for Multistore

Multistores can display in Sitecore when an additional store is created in Znode admin. The Portal ID will be leveraged in determining which catalog is associated to the different storefronts. One catalog will be associated with one store. The following steps need to occur in Znode admin for a Sitecore admin to be able to import an additional store:

- Create a new store in Znode Admin
  - Enter store URL in the URL tab in the managing store section
- Associate/add a catalog to the new store
- Associate/add a categories to the catalog
- If necessary, associate/add products to the categories
- Add new store URL to the Domain table and capture the API key

For more information on store setup, please refer to the Multifront 8 Merchant Quick Start Guide document.

Edit Sitecore Configuration Files

Now that the store in Znode has been created and a catalog has been associated to the store, some minor configuration is required to two Sitecore configuration files.

ZnodeHttpRequestProcessor File:

1. From the \SitecoreWebsiteFolder \Website\App_Config\Include folder open the ZnodeHttpRequestProcessor config file.
2. Locate the sites section and copy the last store that appears in the file as seen on the next page.

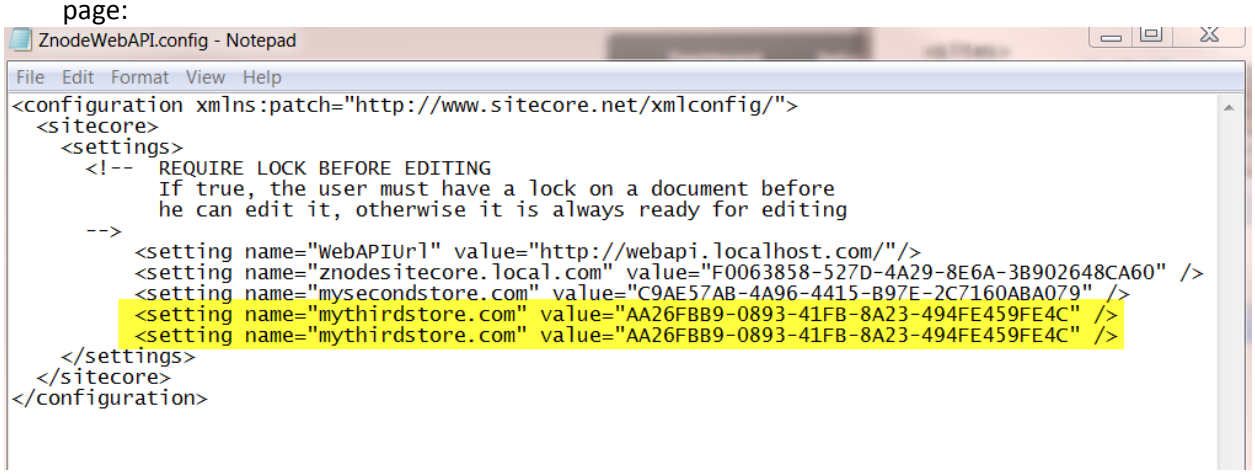

```
    <sites>
      <site name="multifrontsite" hostName="multifrontsitecore7.com" virtualFolder="/"
physicalFolder="/" content="master" rootPath="/sitecore/content" startItem="/MultifrontHome"
database="web" domain="extranet" allowDebug="true" cacheHtml="false" htmlCacheSize="10MB"
registryCacheSize="0" viewStateCacheSize="0" xslCacheSize="5MB" filteredItemsCacheSize="2MB"
enablePreview="true" enableWebEdit="true" enableDebugger="true" disableClientData="false"
patch:after="site[@name='modules_website']"/>

    <site name="nutwholesaler" hostName="
content="master" rootPath="/sitecore/cont                    tWholesalerHome" database="web"
domain="extranet" allowDebug="true" cache                    heSize="10MB" registryCacheSize="0"
viewStateCacheSize="0" xslCacheSize="5MB" filteredItemsCacheSize="2MB" enablePreview="true"
enableWebEdit="true" enableDebugger="true" disableClientData="false" patch:after="site
[@name='multifrontsite']"/>

    <site name="Thirdstore" hostName="mythirdstore.com" virtualFolder="/" physicalFolder="/"
content="master" rootPath="/sitecore/content" startItem="/WineandCheesehome" database="web"
domain="extranet" allowDebug="true" cacheHtml="false" htmlCacheSize="10MB" registryCacheSize="0"
viewStateCacheSize="0" xslCacheSize="5MB" filteredItemsCacheSize="2MB" enablePreview="true"
enableWebEdit="true" enableDebugger="true" disableClientData="false" patch:after="site
[@name='nutwholesaler']"/>
    </sites>
```

3. After copying/pasting the latest store text, change the following data:
    - Site Name – In the example below, changed from Thirdstore to userguidestore
    - Host Name – In the example below, changed from mythirdstore.com to myuserguidestore.com
    - Start Item – In the example below, changed from WineandCheesehome to Userguidehome

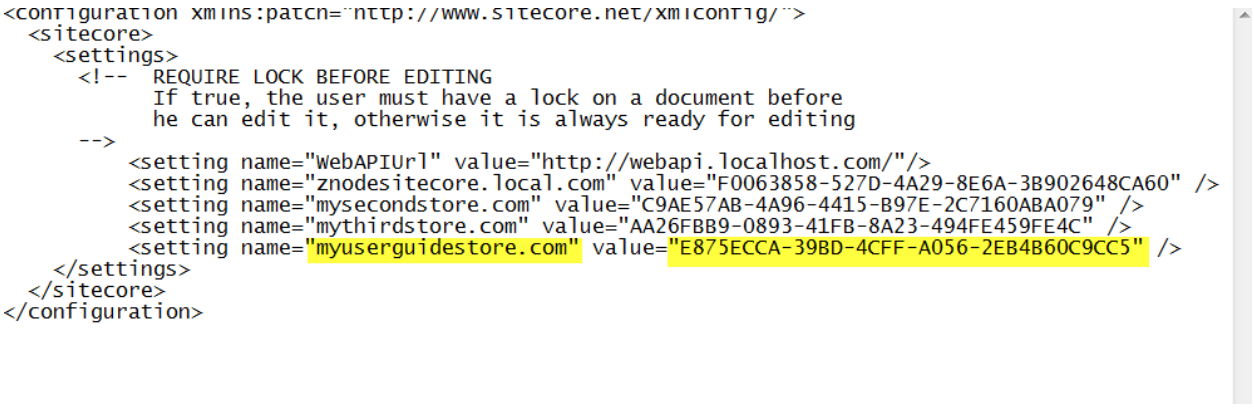
```
    <sites>
      <site name="multifrontsite" hostName="multifrontsitecore7.com" virtualFolder="/"
physicalFolder="/" content="master" rootPath="/sitecore/content" startItem="/MultifrontHome"
database="web" domain="extranet" allowDebug="true" cacheHtml="false" htmlCacheSize="10MB"
registryCacheSize="0" viewStateCacheSize="0" xslCacheSize="5MB" filteredItemsCacheSize="2MB"
enablePreview="true" enableWebEdit="true" enableDebugger="true" disableClientData="false"
patch:after="site[@name='modules_website']"/>

    <site name="nutwholesaler" hostName="mysecondstore.com" virtualFolder="/" physicalFolder="/"
content="master" rootPath="/sitecore/content" startItem="/NutWholesalerHome" database="web"
domain="extranet" allowDebug="true" cacheHtml="false" htmlCacheSize="10MB" registryCacheSize="0"
viewStateCacheSize="0" xslCacheSize="5MB" filteredItemsCacheSize="2MB" enablePreview="true"
enableWebEdit="true" enableDebugger="true" disableClientData="false" patch:after="site
[@name='multifrontsite']"/>

    <site name="thirdstore" hostName="mythirdstore.com" virtualFolder="/" physicalFolder="/"
content="master" rootPath="/sitecore/content" startItem="/WineandCheesehome" database="web"
domain="extranet" allowDebug="true" cacheHtml="false" htmlCacheSize="10MB" registryCacheSize="0"
viewStateCacheSize="0" xslCacheSize="5MB" filteredItemsCacheSize="2MB" enablePreview="true"
enableWebEdit="true" enableDebugger="true" disableClientData="false" patch:after="site
[@name='nutwholesaler']"/>

    <site name="userguidestore" hostName="myuserguidestore.com" virtualFolder="/" physicalFolder="/"
content="master" rootPath="/sitecore/content" startItem="/Userguidehome" database="web"
domain="extranet" allowDebug="true" cacheHtml="false" htmlCacheSize="10MB" registryCacheSize="0"
viewStateCacheSize="0" xslCacheSize="5MB" filteredItemsCacheSize="2MB" enablePreview="true"
enableWebEdit="true" enableDebugger="true" disableClientData="false" patch:after="site
[@name='nutwholesaler']"/>
    </sites>
```

ZnodeWebAPI File:

1. From the \SitecoreWebsiteFolder \Website\App_Config\Include folder open the ZnodeWebAPI config file.
2. Locate the settings section and copy/paste the last store that appears in the file as seen on the next page:



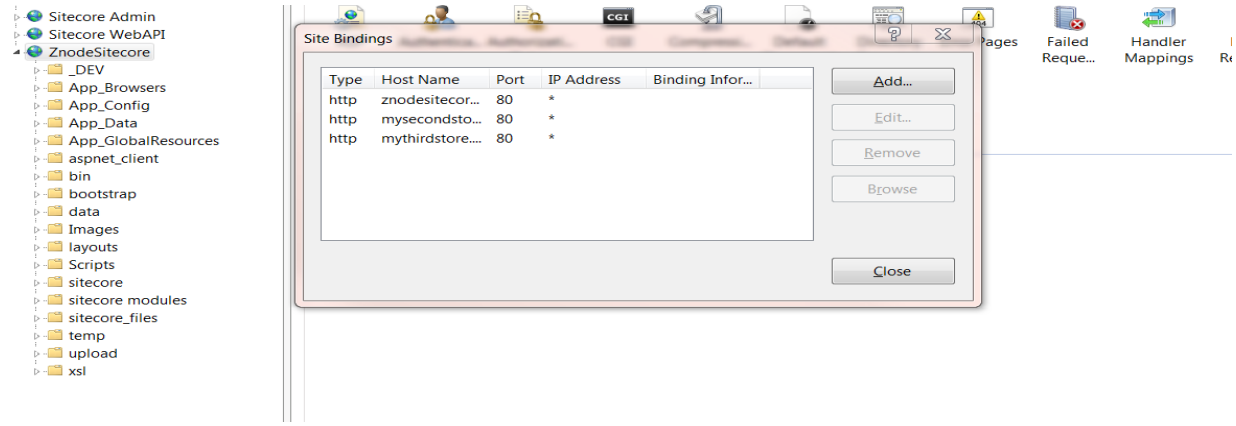
3. Change the setting name and the API key value as seen below:



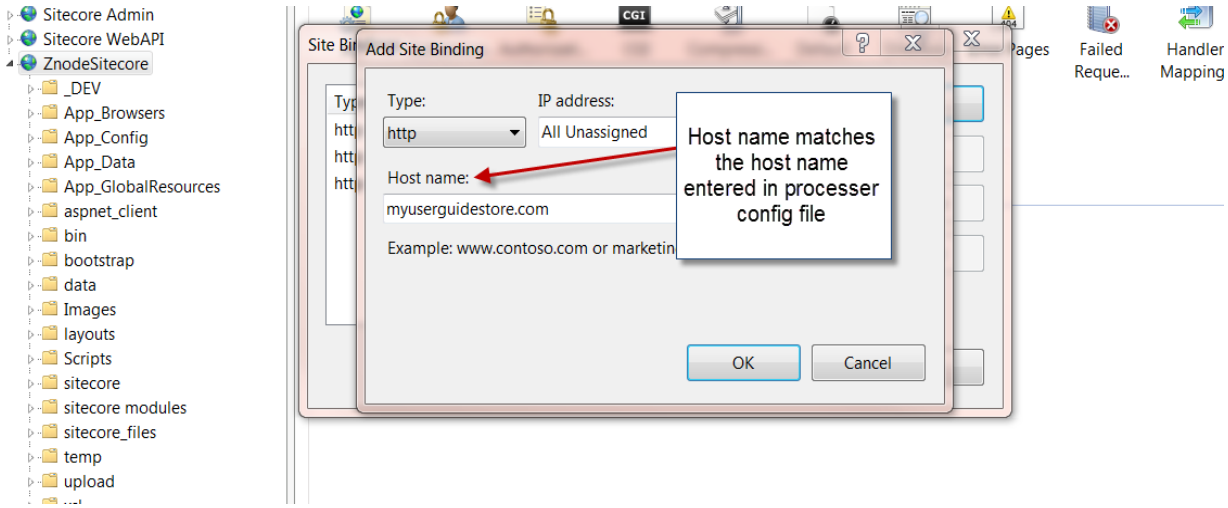
Add Additional Store to IIS

Once the configuration changes have been made, the new store will need to be added to the Sitecore website.

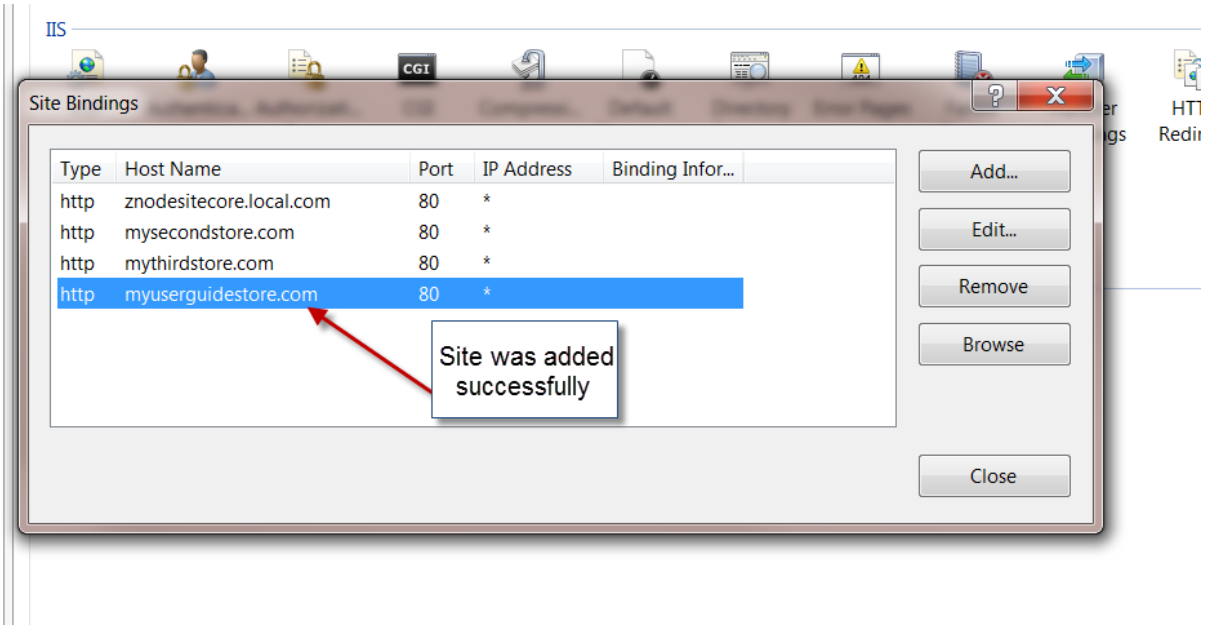
1. In IIS, select the Sitecore site and choose the Edit Bindings option which will open the Site Bindings dialog box.



2. Select the Add button and enter the new sites host name.



3. Select OK and confirm site was added.

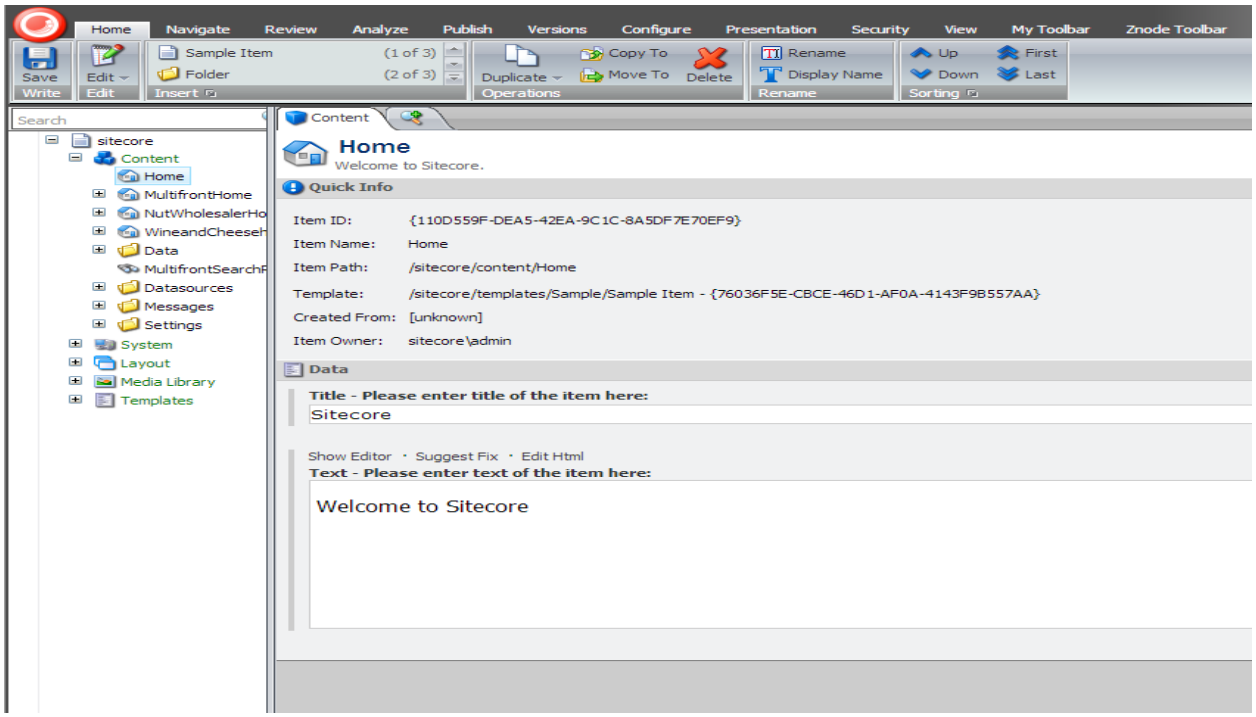


Edits in Content Editor

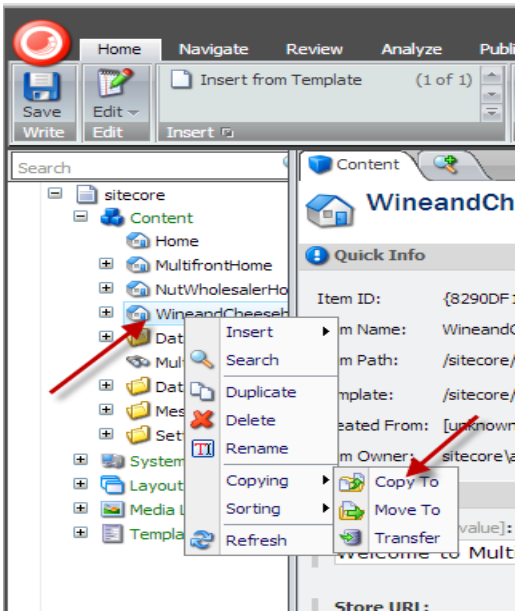
With the store in Znode being created and the required updates to the configuration files/IIS being performed, the final step is adding the store. The following steps show an admin adding a store by copying existing content:

Please Note: An admin can add a store without copying existing content.

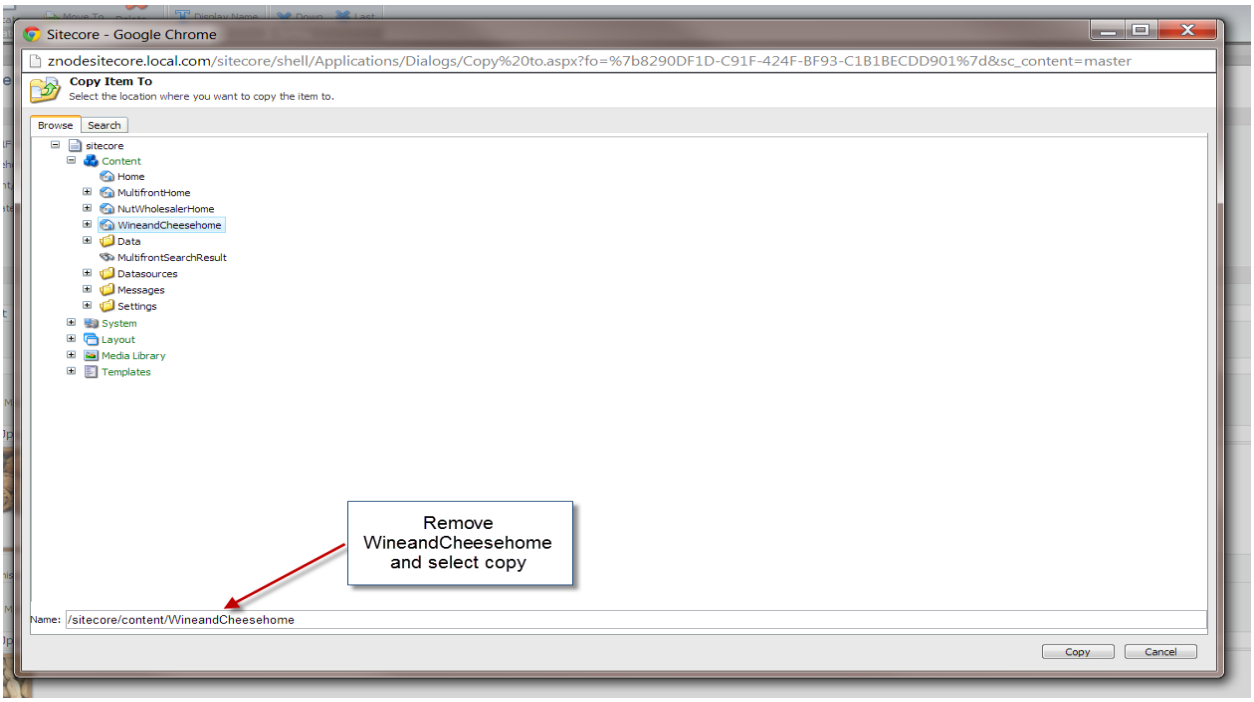
1. Log into Sitecore and navigate to the Content Editor.



2. Select the previous store's site name that has been added and create a copy. In the example below, WineandCheesehome was the last site added to Sitecore.

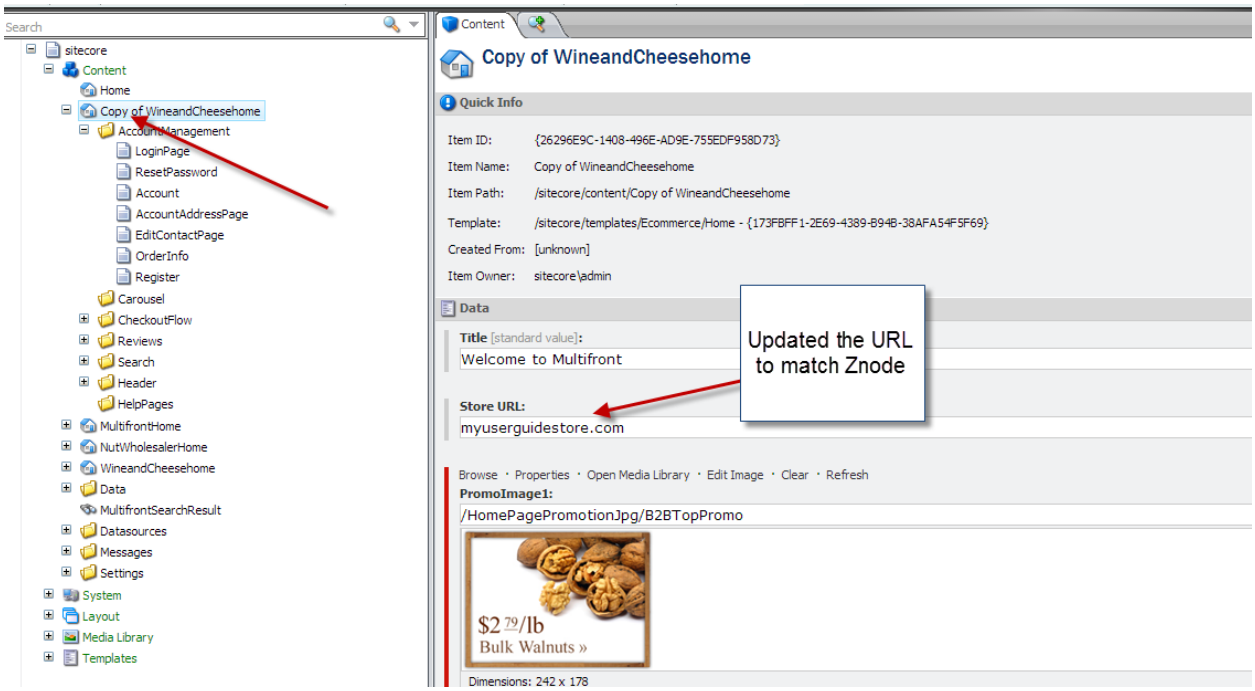


3. Remove the WineandCheesehome page in the path as seen below:

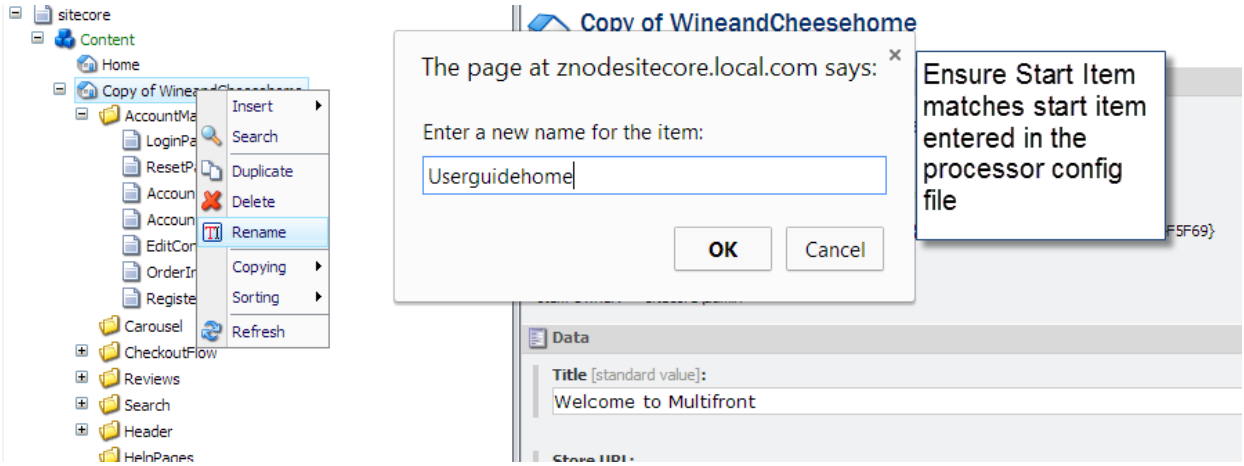


4. The Copy of the Wine and Cheese store now appears as expected. Select the store so the new URL can be entered to match the URL in Znode admin.

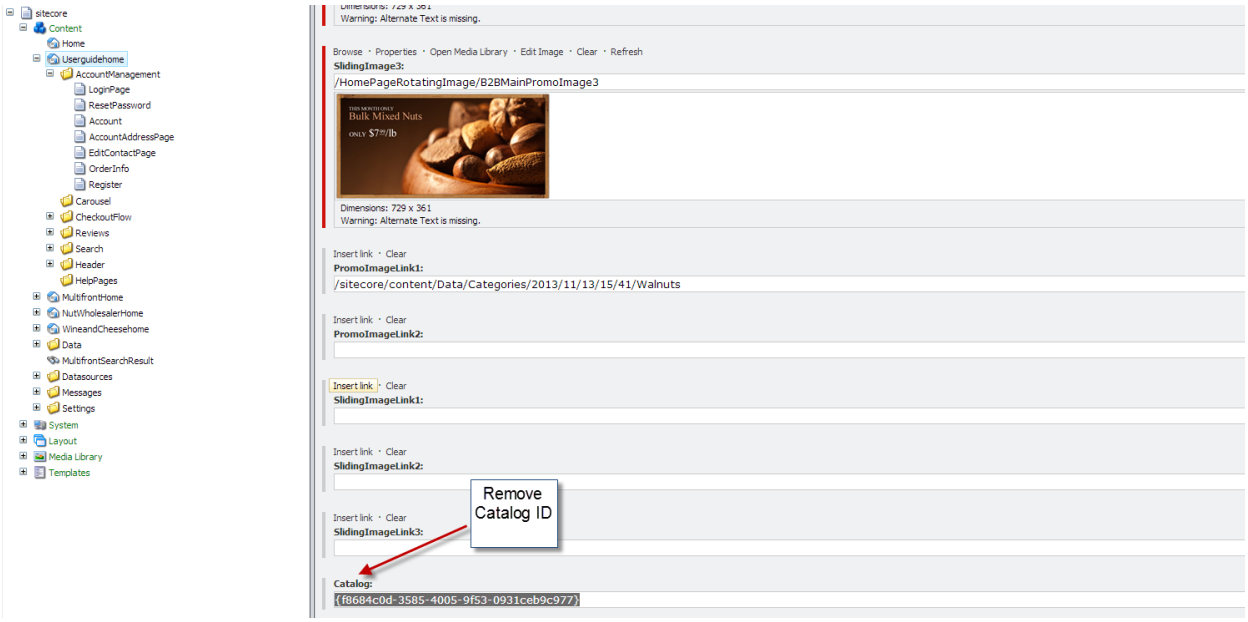
   Please Note: Do not include the https and www in the URL field.



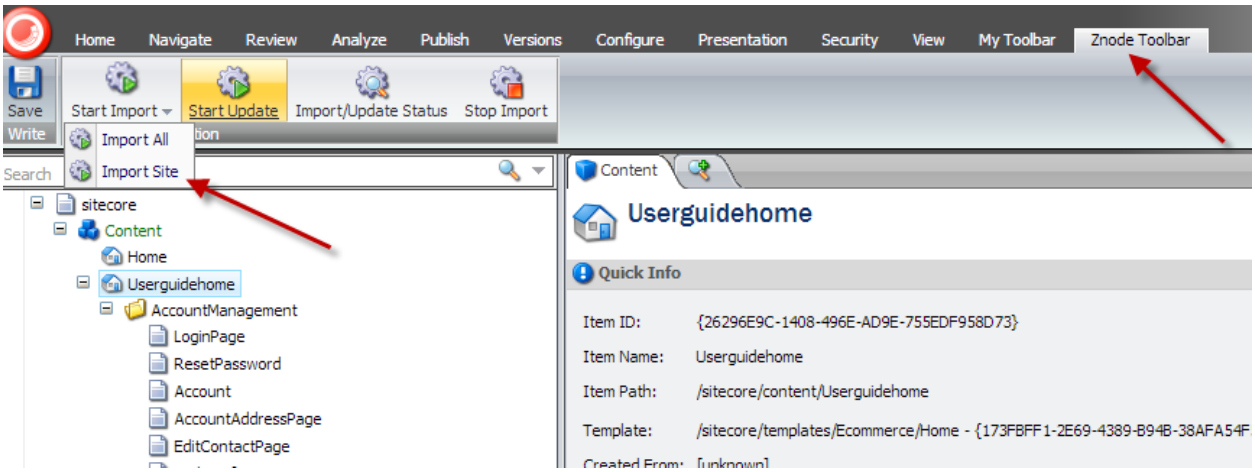
5. Rename the start item to match the new store's start item entered in the configuration file.



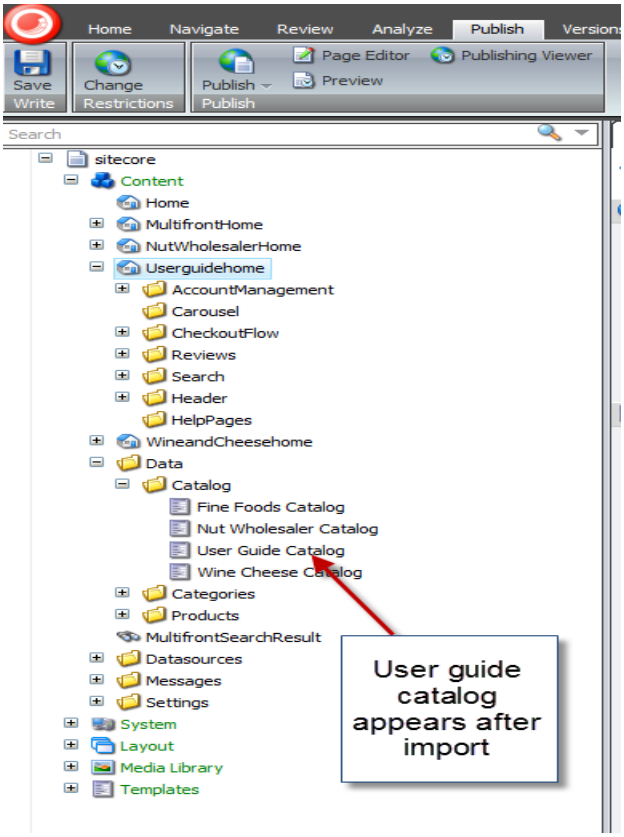
6. Remove the copied store's (which was the Wine and Cheese ID) Catalog ID as seen below.



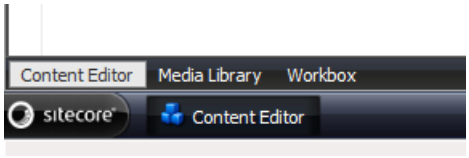
7. Select Import Site from the Znode Toolbar and run import.



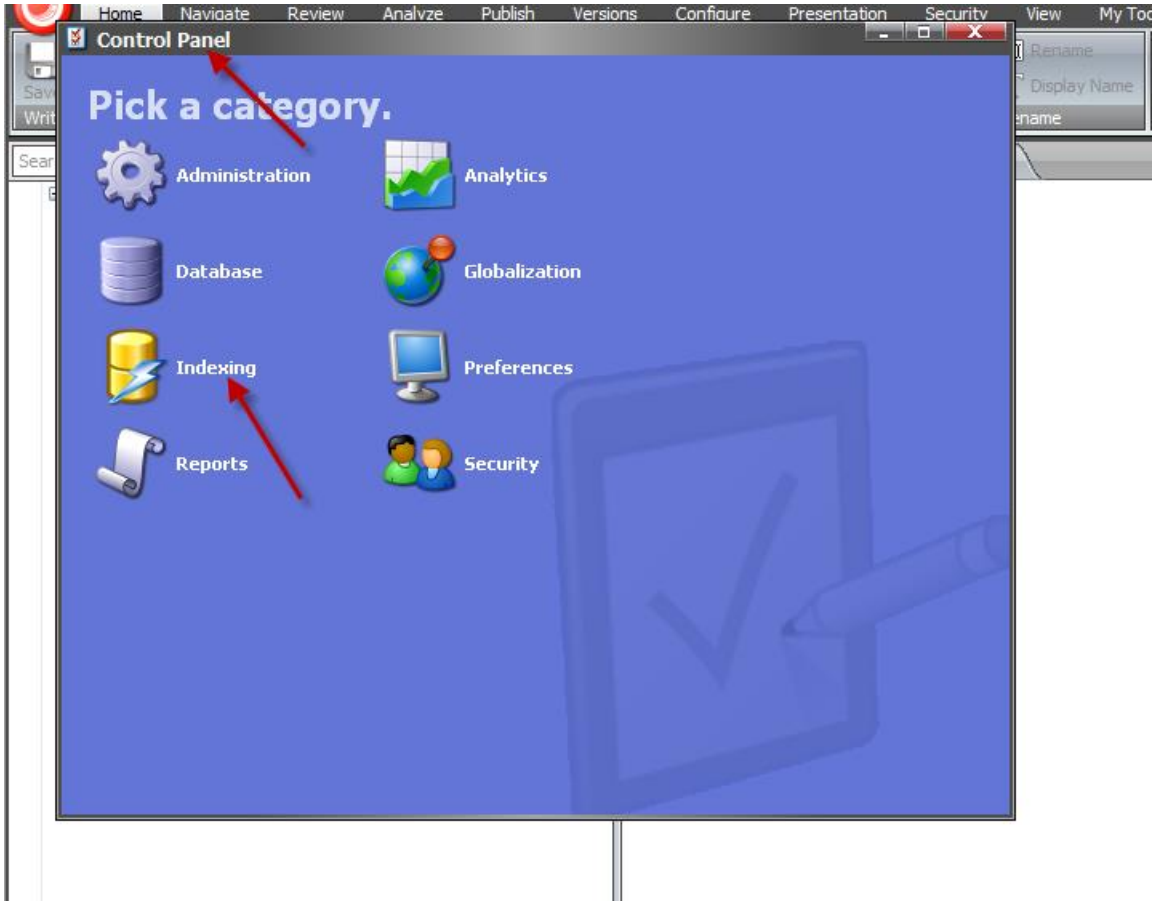
8. Once import is completed successfully, the admin can validate the store catalog data as seen below.



9. To ensure, categories and product will appear successfully on the demo site, running the index is suggested. From Content Editor, select the Sitecore start button.

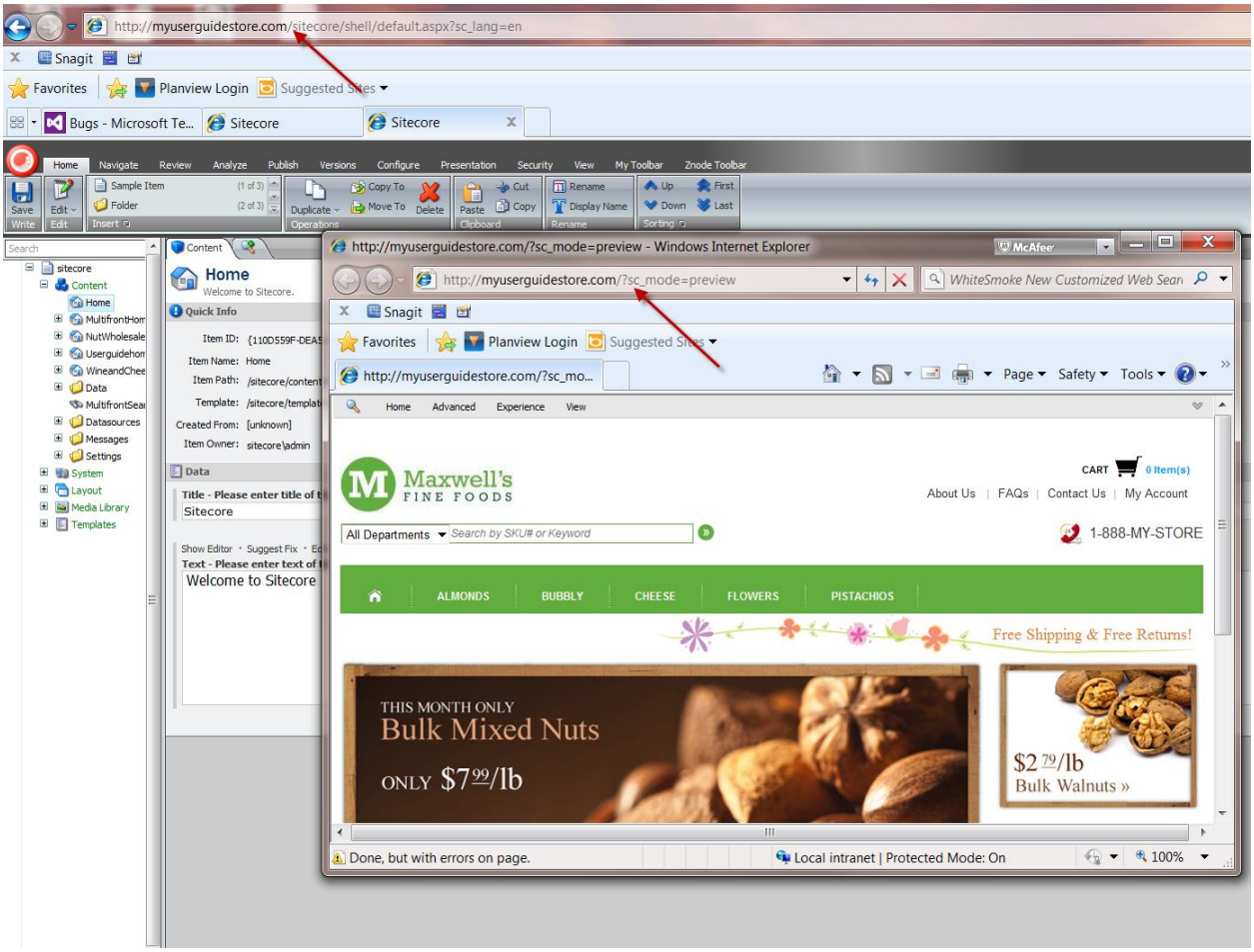


10. From the start button, choose the control panel option which will display the Index. Rebuild the index.



Render New Store's Demo Site

1. Confirm the new store's demo site appears. Open a browser and enter the following data:
    a. Mystorename.com/sitecore. In this example, enter in myuserguidestore.com/sitecore
    b. From Content Editor, perform a preview and the demo site will appear with the correct categories.



Incremental Import

Incremental import allows Sitecore administrators the ability to avoid performing a full import when certain changes in Znode Site Admin occur. The following changes have been made in order to support this feature.

- An internal audit table, **ZnodeSourceModificationAudit,** has been created which will track product catalog changes. Order of operation is as follows:
    - Portal
    - Catalog
    - Categories
    - Products
- Triggers have been incorporated to track the Site Admin updates.
- When an incremental import is selected, only the changes in the audit table will occur.
- Updates to the Znode Toolbar in Sitecore Admin have been performed.
    - Ability to import incremental update for all stores that are in Sitecore

The following table lists the changes that will get added to the Znode Audit Table and sent to Sitecore when an Incremental Import is run.

Modification in Site Admin	Audit record	Import Action
	Portal	
Catalog associated to a different portal	Portal is Modified	If associated with specified portal catalog already exists, no change in Sitecore occurs. Import associated catalog if it does not exists in Sitecore.
	Catalog	
New Catalog added	Catalog is New	Add Sitecore item; Import the catalog
Catalog disabled/removed	Catalog is Disabled or Removed	No changes in Sitecore
Catalog name is modified	Catalog is Modified	Update name and description for the correspondent SC item;
	Category	
New Category added	Category is New	Add Sitecore item, associate with the parent category or catalog (if new category is root)
Category disabled/removed	Category is Disabled or Removed	Remove correspondent Sitecore category item; Remove child categories and/or products
-Category name or description modified -Category (child) associated to a different parent category -Category associated to a different catalog	Category (child) is Modified	Update name and description for the correspondent Sitecore item; change association with the parent category or catalog if different
	Product	
New Product added	Product is New	Add Sitecore item, associate with the parent category
Product disabled/removed	Product is Disabled or Removed	Remove correspondent Sitecore item
-Product field modified -Product associated to a different category	Product is Modified	Update name and description for the correspondent Sitecore item; change association with the parent category if different