1. **Create acl (acl.xml)**

etc/acl.xml

it is global xml file it created only etc folder of module

Syntax : <?xml version="1.0"?>

<!--

/\*\*

\* Copyright © Magento, Inc. All rights reserved.

\* See COPYING.txt for license details.

\*/

-->

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

<acl>

<resources>

<resource id="Magento\_Backend::admin">

<resource id="Ezest\_Practice::practice" title="Practice" translate="title" sortOrder="40">

<resource id="Ezest\_Practice::manage" title="Manage Practice" translate="title" sortOrder="10" />

<resource id="Ezest\_Practice::add" title="add" translate="title" sortOrder="20" />

</resource>

<resource id="Magento\_Backend::stores">

<resource id="Magento\_Backend::stores\_settings">

<resource id="Magento\_Config::config">

<resource id="Ezest\_Practice::config\_practice" title="Practice Section" translate="title" sortOrder="50" />

</resource>

</resource>

<resource id="Magento\_Backend::stores\_other\_settings">

<resource id="Ezest\_Practice::group" title="Practice Groups" translate="title" sortOrder="10" />

</resource>

</resource>

</resource>

</resources>

</acl>

</config>

Apply resource in controller:

/\*\*

\* {@inheritdoc}

\*/

protected function \_isAllowed()

{

return $this->\_authorization->isAllowed('Ezest\_Practice::edit');

}

Resource id **“'Ezest\_Practice::edit”** which Is created in acl.xml

$this->\_authorization->isAllowed : It is exist in parent class

$this->\_authorization = Magento\Framework\Authorization\Interceptor

To restrict frontend controller

**$auth = $object\_manger->get('Magento\Framework\AuthorizationInterface');**

**if($auth->isAllowed('Pulsestorm\_AclExample::config')){**

**//user is logged in here**

**}else{**

**//user is not logged in here**

**}**

Resource id : **Pulsestorm\_AclExample::config'**

**2. Cron**

**File Name : etc/crontab.xml**

**Several Magento features require at least one cron job, which schedules activities to occur in the future. A partial list of these activities follows:**

Catalog price rules

Newsletters  
Generating Google sitemaps  
Customer Alerts/Notifications (product price change, product back in stock)  
Reindexing  
Private sales (Magento Commerce only)  
Automatic updating of currency rates  
All Magento e-mails (including order confirmation and transactional)

<?xml version="1.0" ?>

<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="urn:magento:module:Magento\_Cron:etc/crontab.xsd">

<group id="default">

<job instance="Ezest\Practice\Cron\Test" method="execute" name="ezestpractice\_helloworld\_cron">

<schedule>\* \* \* \* \*</schedule>

</job>

</group>

</config>

\* \* \* \* \* command to be executed

| | | | |

| | | | +----- Day of week (0 - 7) (Sunday=0 or 7)

| | | +------- Month (1 - 12)

| | +--------- Day of month (1 - 31)

| +----------- Hour (0 - 23)

+------------- Minute (0 - 59)

<?php

namespace Ezest\Practice\Cron;

class Test

{

public function execute()

{

$writer = new \Zend\Log\Writer\Stream(BP . '/var/log/cron.log');

$logger = new \Zend\Log\Logger();

$logger->addWriter($writer);

$logger->info(\_\_METHOD\_\_);

}}

Command options:

bin/magento cron:run [–group=”"]

where --group specifies the cron group to run (omit this option to run cron for all groups)

To run the indexing cron job, enter:

php bin/magento cron:run --group index

To run the default cron job, enter:

php bin/magento cron:run --group default

ex. php bin/magento cron:run –group="default"

**3. Indexer**

1. The validity of all indexes is stored in the table **indexer\_state**.
2. When the cron job runs, it compares the version\_id of its **mview\_state** table with the version\_ids of the changelog table
3. Magento’s cron job “**indexer\_reindex\_all\_invalid**” rebuilds the invalid indexes.
4. **bin/magento indexer:reindex** This command rebuilds the indexes from scratch. For webshops of some size, this is a non-trivial operation. For a webshop with thousands of products and many categories and store views, it easily takes an hour to complete.

Syntax :

mview.xml

<?xml version="1.0" encoding="UTF-8"?>

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

<view id="ezest\_practice\_indexer" class="Ezest\Practice\Model\Indexer\Test" group="indexer">

<subscriptions>

<table name="catalog\_product\_entity" entity\_column="entity\_id" />

</subscriptions>

</view>

</config>

indexer.xml

<?xml version="1.0"?>

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

<indexer id="ezest\_practice\_indexer" view\_id="ezest\_practice\_indexer" class="Ezest\Practice\Model\Indexer\Test">

<title translate="true">Ezest Practice Indexer</title>

<description translate="true">Practice of custom indexer</description>

</indexer>

</config>

class:

namespace Ezest\Practice\Model\Indexer;

class Test implements \Magento\Framework\Indexer\ActionInterface, \Magento\Framework\Mview\ActionInterface

{

/\*

\* Used by mview, allows process indexer in the "Update on schedule" mode

\*/

public function execute($ids){

//code here!

echo 'heloop';

}

/\*

\* Will take all of the data and reindex

\* Will run when reindex via command line

\*/

public function executeFull(){

//code here!

echo 'heloot';

}

/\*

\* Works with a set of entity changed (may be massaction)

\*/

public function executeList(array $ids){

//code here!

echo 'heloo8';

}

/\*

\* Works in runtime for a single entity using plugins

\*/

public function executeRow($id){

echo 'heloos';

//code here!

}

}

1. **menu.xml**

<?xml version="1.0"?>

<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="../../../../Magento/Backend/etc/menu.xsd">

<menu>

<add id="Ezest\_Practice::parent" title="Practice" module="Ezest\_Practice" sortOrder="100" resource="Ezest\_Practice::practice"/>

<add id="Ezest\_Practice::practice" title="Ezest Practice" module="Ezest\_Practice" sortOrder="20" parent="Ezest\_Practice::parent" action="practice/practice" resource="Ezest\_Practice::manage"/>

</menu>

</config>

1. **config.xml**

**<?xml version="1.0"?>**

**<!--**

**/\*\***

**\* Copyright © Magento, Inc. All rights reserved.**

**\* See COPYING.txt for license details.**

**\*/**

**-->**

<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="urn:magento:module:Magento\_Store:etc/config.xsd">

<default>

<practice\_section>

<general>

<display\_text>205</display\_text>

<enable>1</enable>

</general>

</practice\_section>

<practice\_section2>

<general>

<display\_text2>

602

</display\_text2>

</general>

</practice\_section2>

</default>

</config>

7.**events.xml**

This can be placed area wise and globally.

<?xml version="1.0"?>

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

<event name="ezest\_practice\_display\_text">

<observer name="mp\_display\_text" instance="Ezest\Practice\Observer\ChangeDisplayText"/>

</event>

</config>

To create observer :

$textDisplay = new \Magento\Framework\DataObject(array('text'=>'heloo ezest practice'));

echo get\_class($this->\_eventManager);

$this->\_eventManager->dispatch('ezest\_practice\_display\_text',['mp\_text'=>$textDisplay]);

echo $textDisplay->getText();

**\_eventManager = Magento\Framework\Event\Manager**

namespace Ezest\Practice\Observer;

class ChangeDisplayText implements **\Magento\Framework\Event\ObserverInterface**

{

public function execute**(\Magento\Framework\Event\Observer** $observer)

{

$displayText = $observer->getData('mp\_text');

echo $displayText->getText() . " - Event </br>";

$displayText->setText('Execute event successfully.');

}}