**Magento2**

**Install Language Pack**

**>>** [**https://amasty.com/knowledge-base/add-magento-2-language-pack.html**](https://amasty.com/knowledge-base/add-magento-2-language-pack.html)

**>>** [**https://www.mageplaza.com/blog/how-to-install-language-pack.html#how-to-install-magento-2-language-pack-for-your-store**](https://www.mageplaza.com/blog/how-to-install-language-pack.html%23how-to-install-magento-2-language-pack-for-your-store)

**>> https://github.com/mageplaza/magento-2-greek-language-pack**

**Unsplash Random Image url –**

<https://source.unsplash.com/random/13x22>

**App –** Magento important folder

* Developer mostly work in this folder

**Code –** All custome module store in this folder,

Custome module use for overiding functionlity.

**Design –** Themes stores in this folder.

**i18in** - language packages are stored in this folder.

**Etc** – All configration file stored in this folder.

* Ex.database configration file -env.php
* **Config.php** - is an autogenerted file, which stores the list of installed modules.
* **Db\_Schema** - This is the file where you will define your tables, columns, pretty much everything you need.
* **View.xml** - view.xml file which contains image configurations for all images and thumbnails.
* **Di.xml** - Each module can have a global and area-specific di.xml file. Magento reads all the di.xml configuration files declared in the system and merges them all together by appending all nodes.
* **NonComposerComponentRegistration.php** - which registers components from app.

**.DS\_store** - The .DS\_Store file is a file created by Mac OS X

that holds meta-information about a directory.

**Bin –** All CLI(Command line interface) run by this folder.

* If this folder is not exist then any magento command will not work.

**Setup -** This contains all files which are related to Magento’s installation setup.

**Dev –** This folder is used for Testing.

* All Testing tool is stored in this folder.

**Generated –** It is auto created folder, Whenever we run setup command this folder will automatic generate.

**Lib –** All library stored in this folder.

* Libraries in programming languages are **collections of prewritten code** that users can use to optimize tasks.

**Pub –**

**Error –** All default error of magento stored in this folder, and you can create any other error.

**Media –** All **Images of Theme,products,pages** stored in this folder.

**Static –** It is also a auto created folder, This folder will automatically created whenever we run this **setup:static-content:deploy** command.

**Html,CSS,JS of themes stored in this static folder.**

**But we can’t override this static file.**

**Var –** All **classes, sessions ,cache** and **logs** are stored in this folder.

**Cache -** Magento collects configuration from all modules, merges it, and saves the merged result to the cache. This cache also contains store-specific settings stored in the file system and database. Clean or flush this cache type after modifying configuration files.

**Log -** Errors stored in this folder.

**Vendor –** Vendor is **core folder of** **magento,** The whole code of page stored in this folder,but we can’t override in this folder.

**For overriding we create module.**

**What is the difference between Magento 2 website, store, and store view?**

<https://amasty.com/knowledge-base/what-is-the-difference-between-magento-2-website-store-and-store-view.html>

**Attribute –** In stores there are so many products Ex. Shirt,tshirt,shoes and watch.

For example there are many types of material use for making watches,ex.metal,plastic,leather and etc.

So the material which use for creating product is called **Attribute.**

**For creating new attribute**

**Admin -> Store->Attributes->product->Add new Attribute.**

**Youtube >** [**https://youtu.be/ccpbzVsoU6w**](https://youtu.be/ccpbzVsoU6w)

**Attribute set -**  store->Attributes-> Attribute set->Default

All default attribute stores in default.

**About application modes –**

[**https://devdocs.magento.com/guides/v2.3/config-guide/bootstrap/magento-modes.html**](https://devdocs.magento.com/guides/v2.3/config-guide/bootstrap/magento-modes.html)

**Theme Structure** - <https://bsscommerce.com/confluence/magento-2-theme-structure/>

**Layout Structure –**

<https://devdocs.magento.com/guides/v2.3/frontend-dev-guide/layouts/layout-overview.html>

In magento you can verify by checking the xml file and for each page there will separate layout will be defined.

Home page - cms\_index\_index.xml

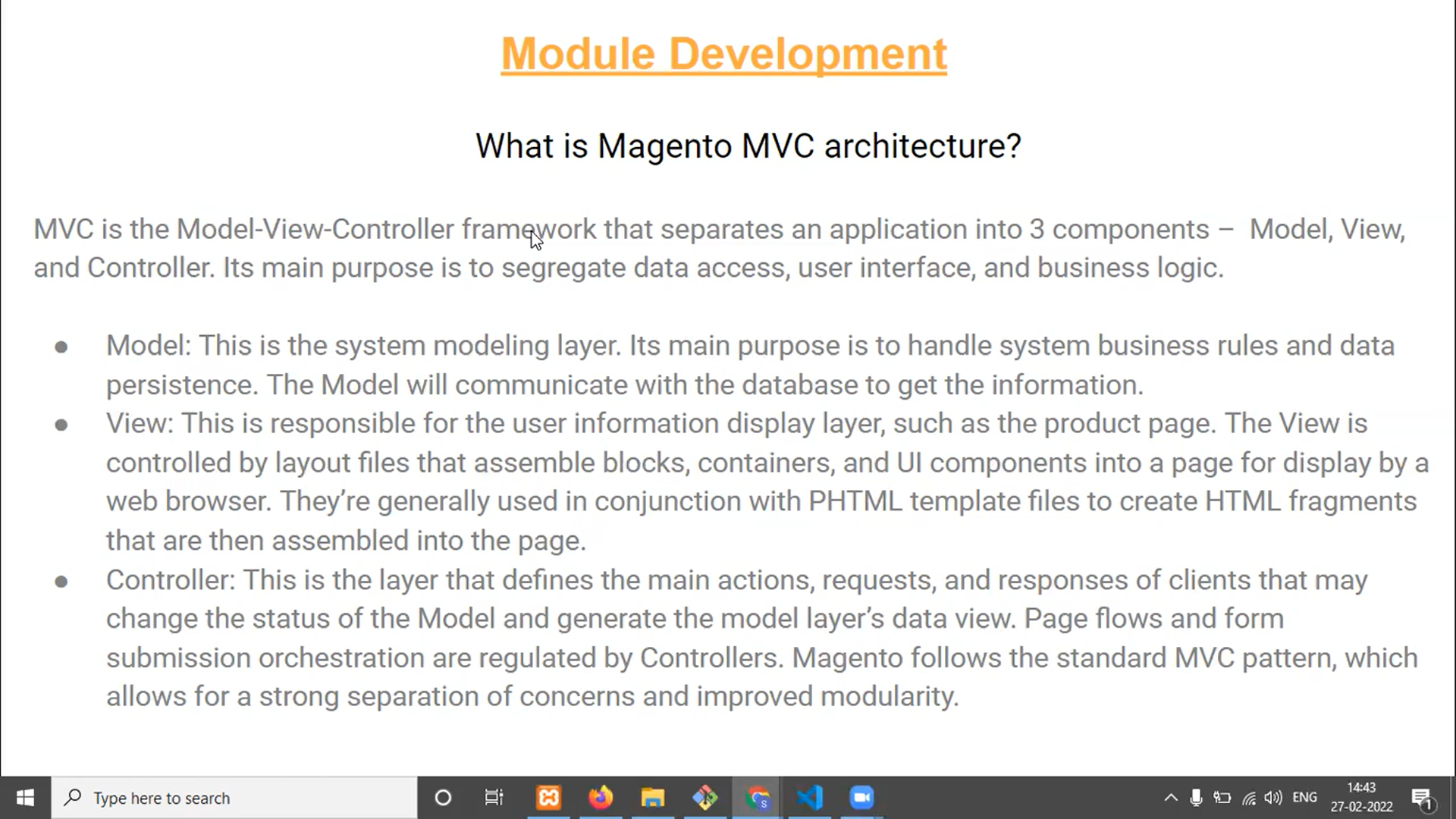
Category Page - catalog\_category\_view.xml

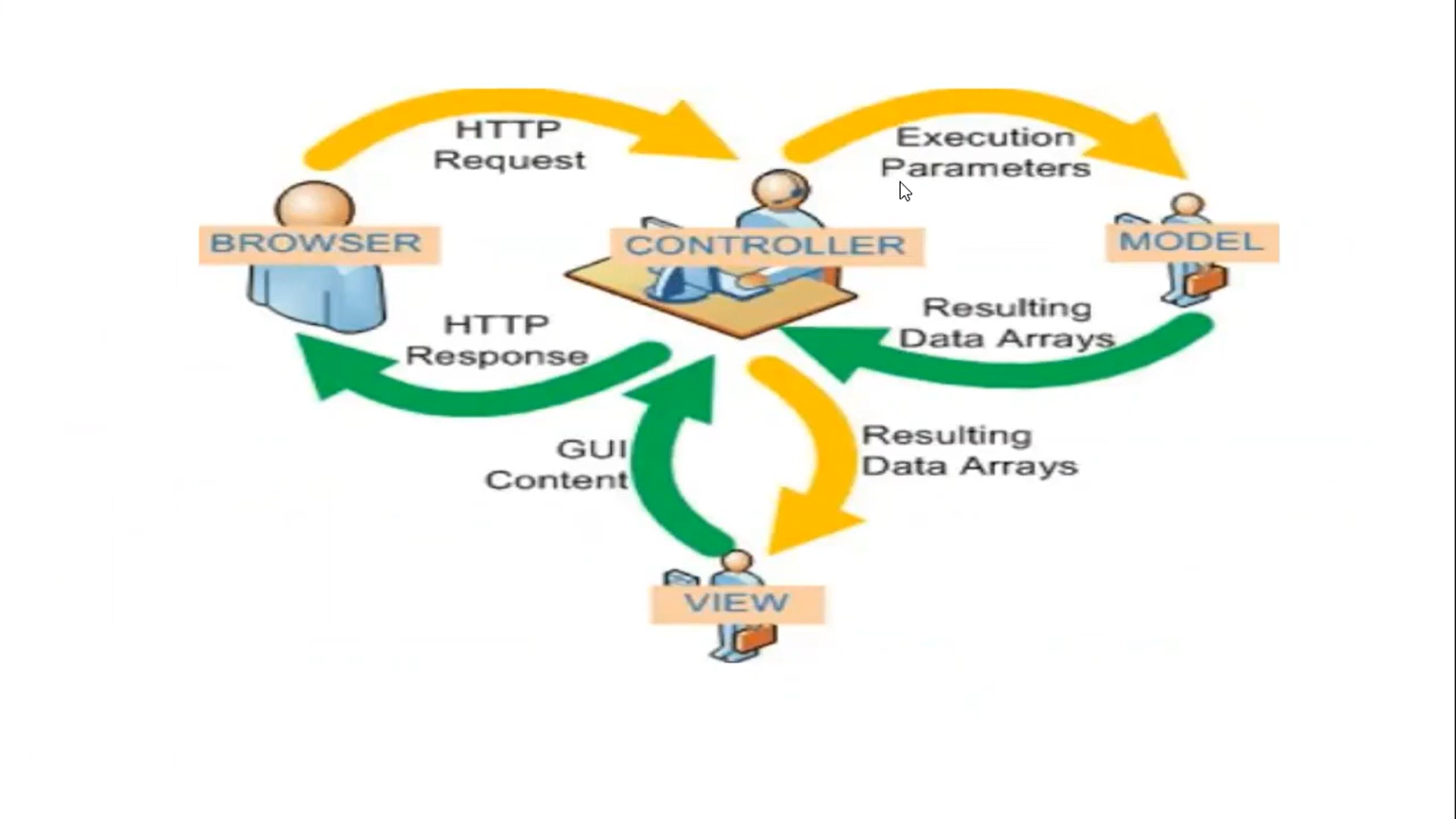
Product Page - catalog\_product\_view.xml

Cart page - checkout\_cart\_index.xml

Checkout - checkout\_index\_index.xml

**Extenstion –**



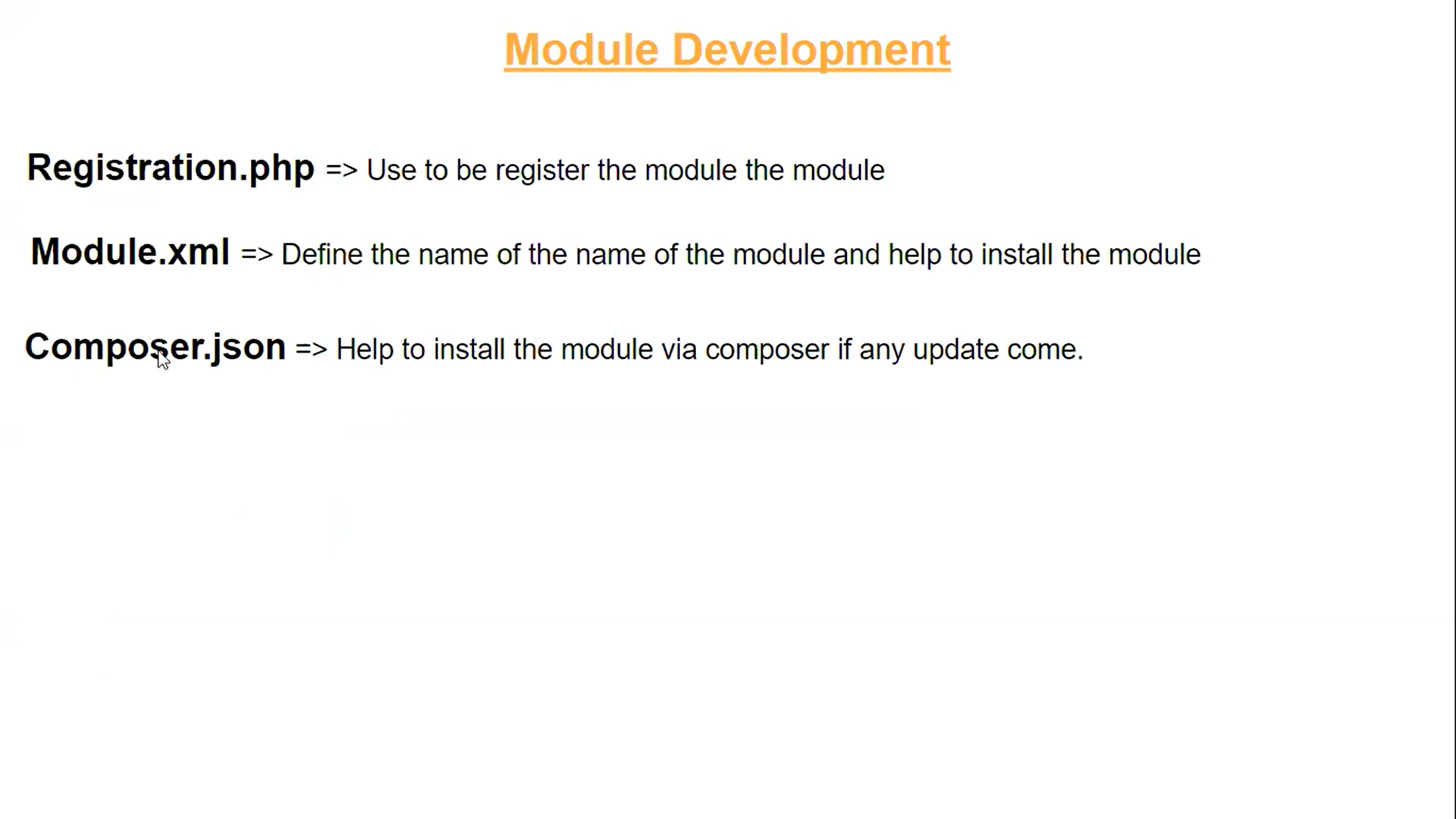




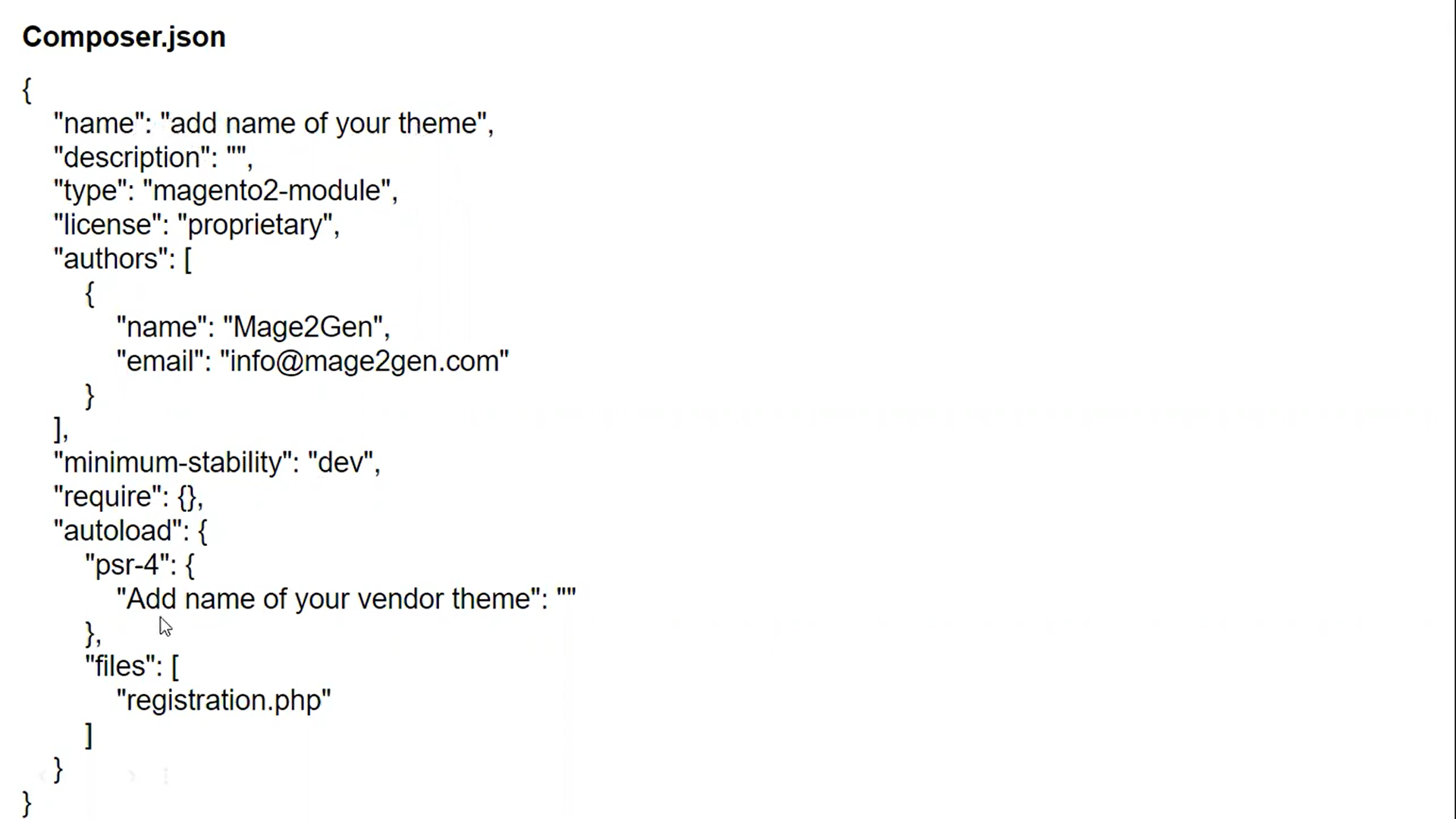
<https://belvg.com/blog/general-module-structure-in-magento-2.html>

**Composer.json -** It is use for upgrading module.

Ex. Module 1.0 to Module2.0.







**Block -** Block are responsible for rendering content (everything from frontend, backend, emails, and more). If content is being displayed somewhere chances are it is a block.

Block ek function he jo data ko controller se leta he,aur controller data ko view me bhejne me help karta he.

**Controller -** controllers are responsible for processing all requests made via a url.

**etc** – contains the configuration files.

**Helper –**

Helpers are classes that provide functionalities for various features throughout the Magento website. Available for use anywhere on the website, Magento 2 Helper can be called in controllers, views, models and even in other helpers too!

Helper me created function ko ham kahi bhi call kar sakte he.

Magento 2 Helper Class includes various functions and methods which are used commonly throughout the application.

**Model** – contains the files that are responsible for connection with the database (optional directory).

Model is used for executing database query.

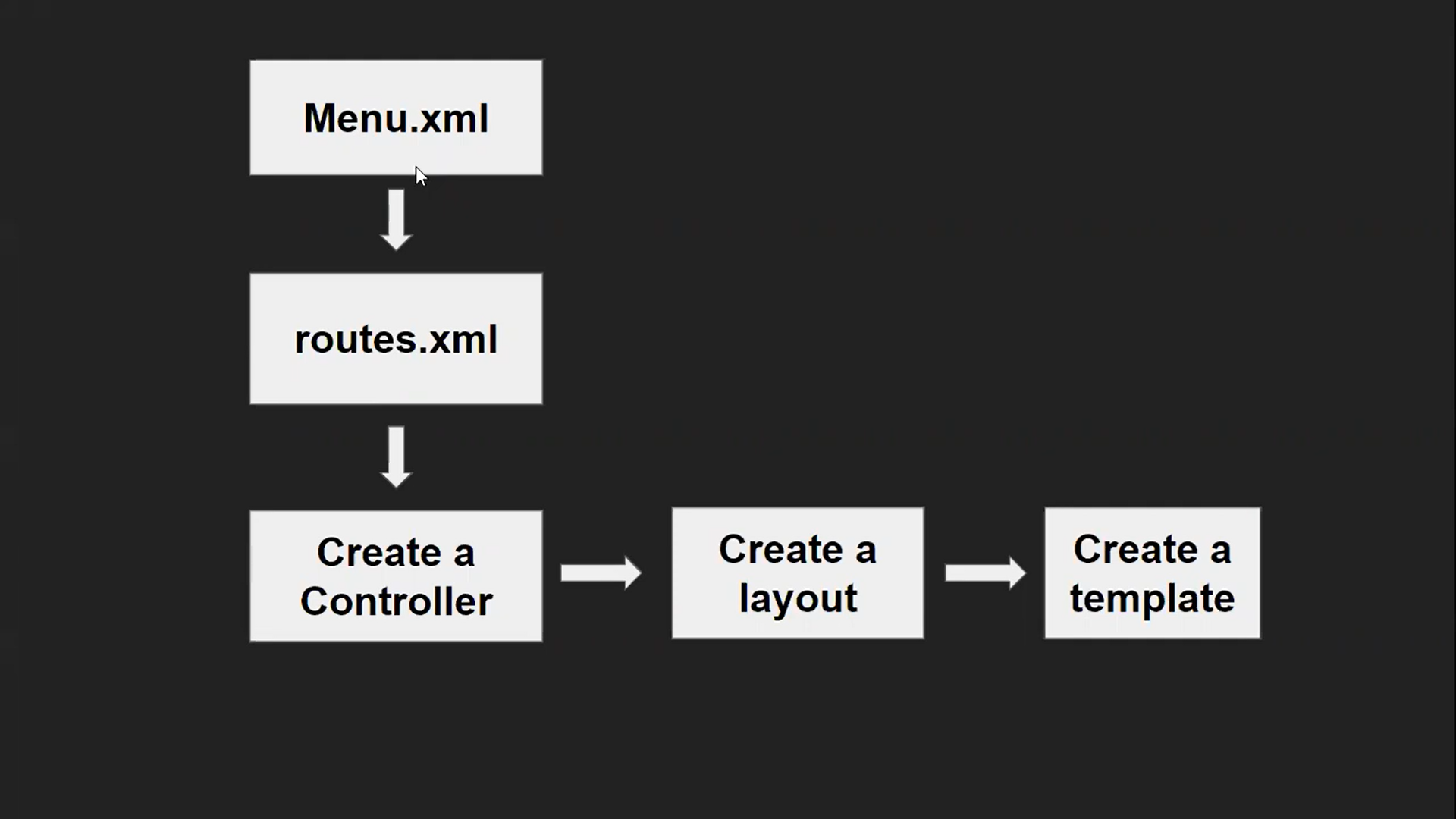
**Setup** – includes the files that are necessary to make changes in the database – i.e. creating tables, fields, or other records required for the module performance (optional directory).

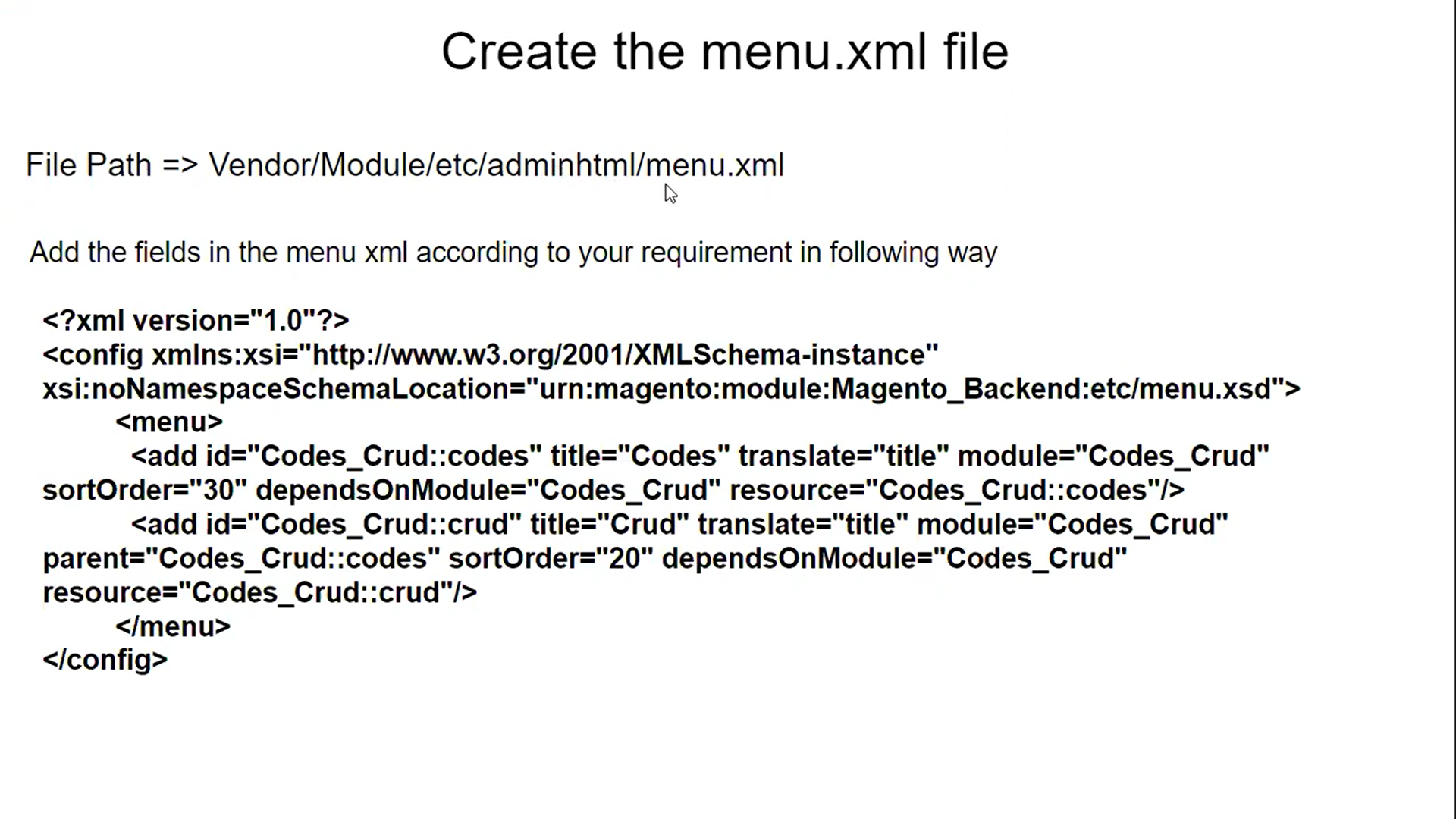
**Test** – contains the files that are necessary to perform the module test. (optional directory).

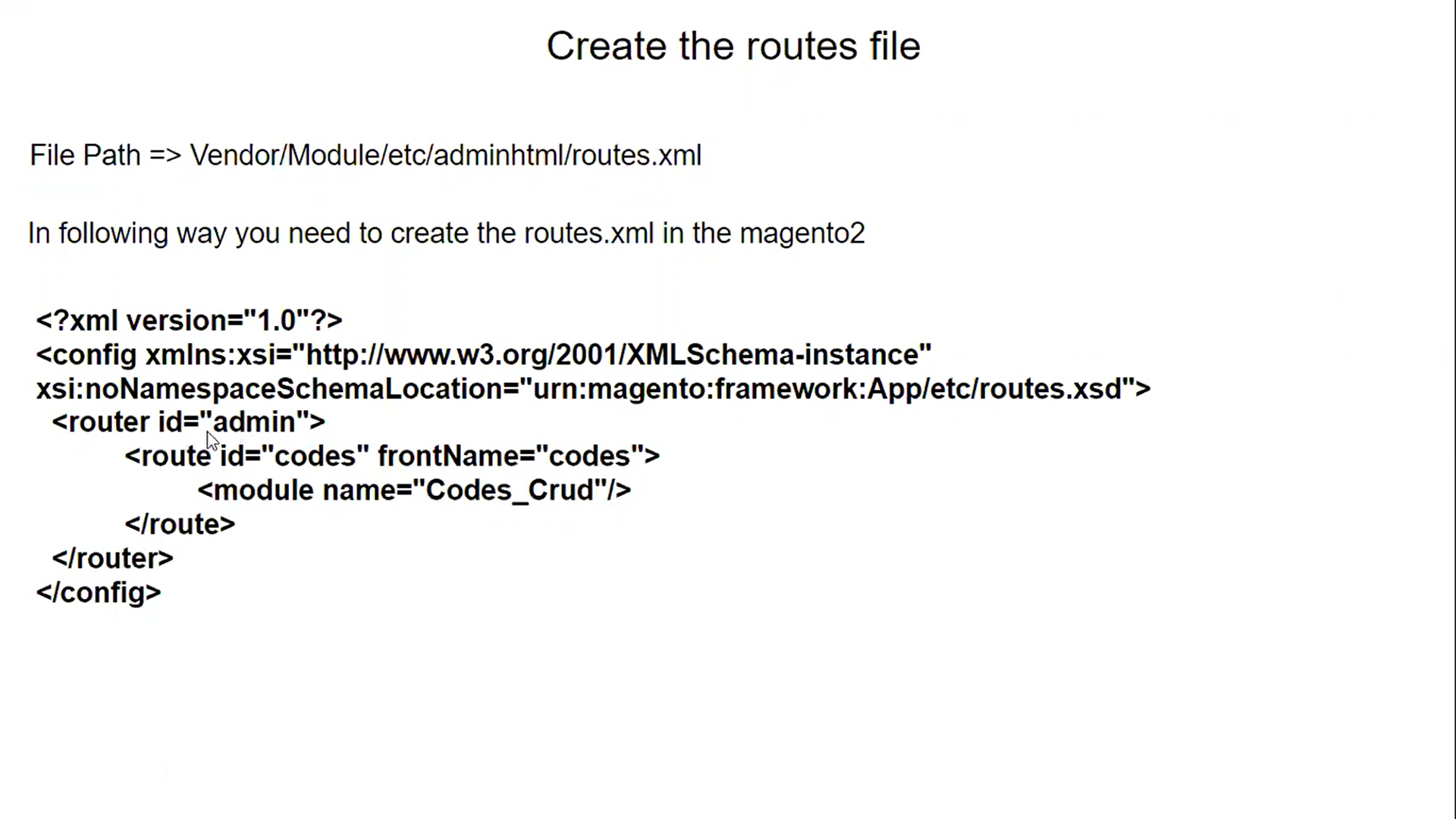
**View** – includes the files that are necessary for information output on the frontend and backend. Commonly these files contain html code (optional directory).

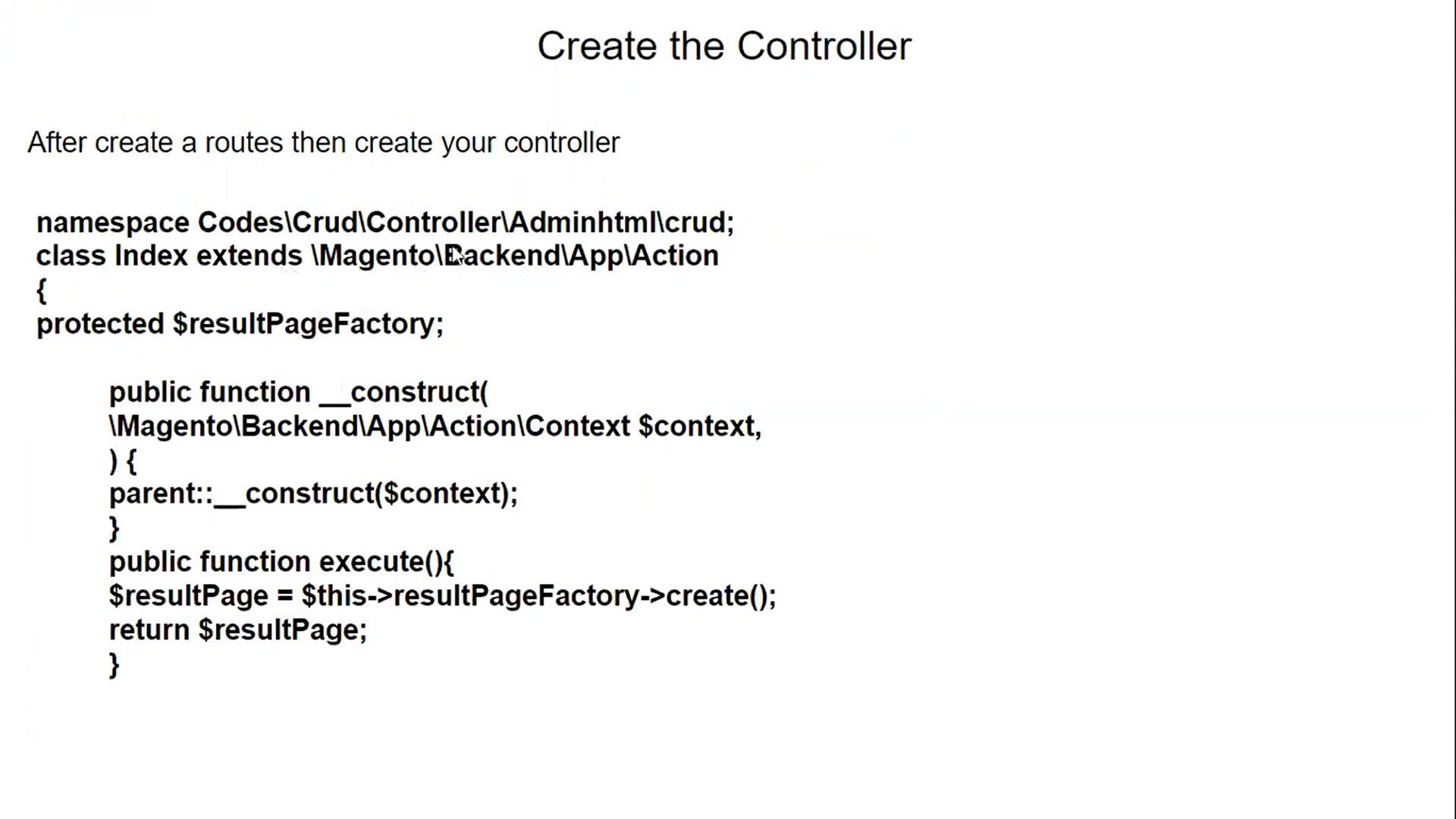
**Create Grid in backend using route, model, block, controller, ui\_component.**

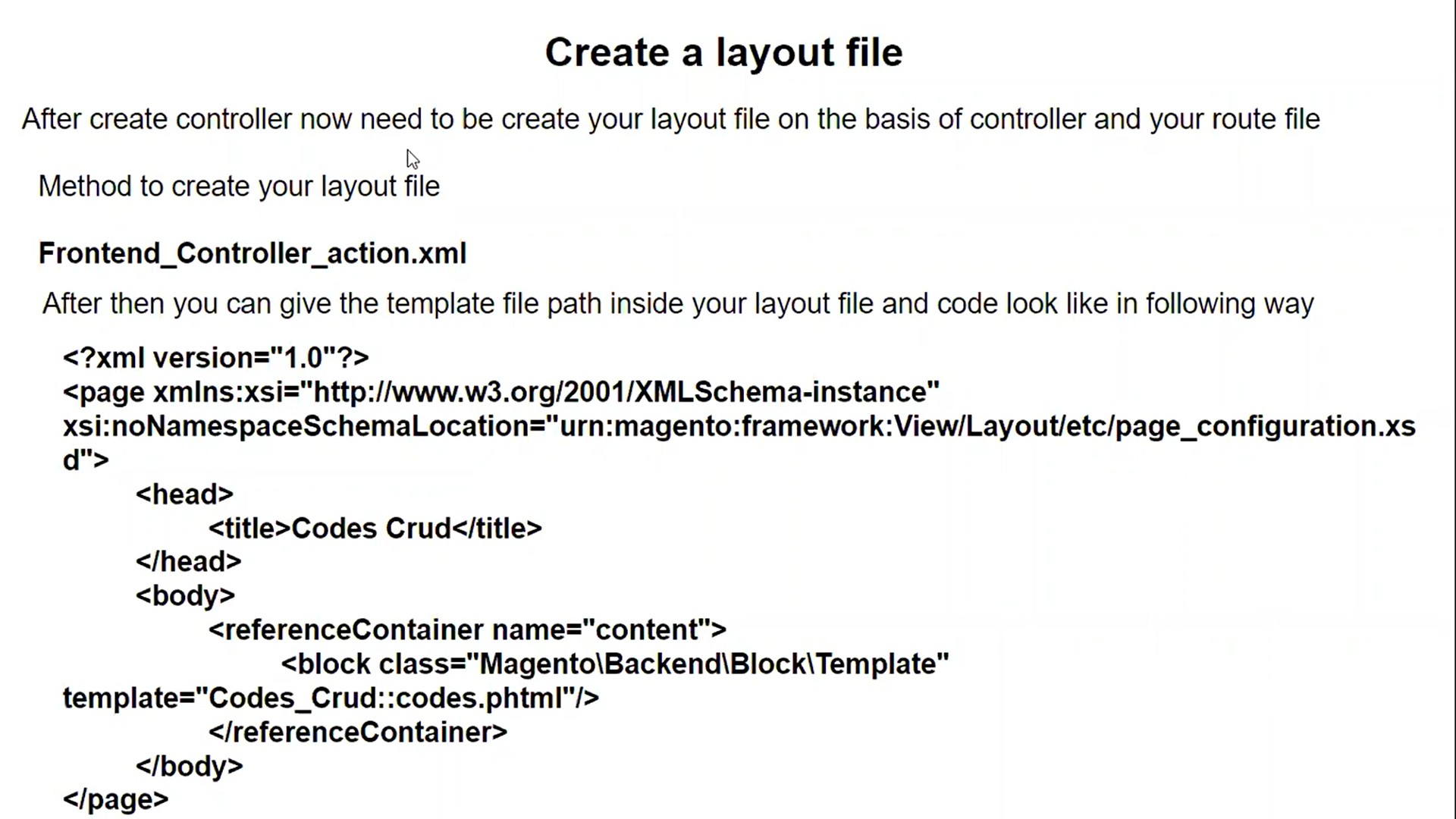
**Link -** [**https://www.magedelight.com/resources/how-to-create-ui-component-grid-and-form-in-magento-2/**](https://www.magedelight.com/resources/how-to-create-ui-component-grid-and-form-in-magento-2/)











**Action** -

Action = “purchased/index/index” is equal to

Action = “frontname/controller/actionMethod”.

// Create layout file withou block >

<block class="Magento\Framework\View\Element\Template" template="Vendor\_Module::custom.phtml">

<block class="Magento\Framework\View\Element\Template" template="Dharma\_AdvanceSearch::homepage.phtml" />

**Create UI Component Grid and Form in Magento -**

<https://www.rohanhapani.com/magento-2-create-ui-component-grid-and-form/>

**Magento 2 Add Pagination in Custom Collection**

<https://dolphinwebsolution.com/magento-2-add-pagination-in-custom-collection-of>

<argument> attributes

Name – argument name

xsi:type- Argument type.string, boolean, object, number, null, array, options, url, helper.

// important

To pass multiple arguments use the following construction:

|  |  |
| --- | --- |
| 1  2  3  4  5 | <arguments>  <argument name="item1" xsi:type="string">Custom string</argument>  <argument name="item2" xsi:type="boolean">true</argument>  ...  </arguments> |

Arguments values set in a layout file can be accessed in [templates](https://devdocs.magento.com/guides/v2.4/frontend-dev-guide/templates/template-overview.html) using the getData('{ArgumentName}') and hasData('{ArgumentName}') methods. The latter returns a boolean defining whether there’s any value set. {ArgumentName} is obtained from the name attribute the following way: for getting the value of <argument name="some\_string"> the method name is getData('some\_string').

* The *array* type:

|  |  |
| --- | --- |
| 1  2  3  4  5 | <argument name="custom\_array" xsi:type="array">  <item name="array\_key\_one" xsi:type="string">First Item</item>  <item name="array\_key\_two" xsi:type="string">Second Item</item>  ...  </argument> |

* The *url* type:

|  |  |
| --- | --- |
| 1  2  3  4  5 | <argument name="shopping\_cart\_url" xsi:type="url" path="checkout/cart/index" >  <param name="param1">param1value</param>  <param name="param2">param2value</param>  ...  </argument> |

The *url* may have parameters, but they are optional.

* The *helper* type:

|  |  |
| --- | --- |
| 1  2  3  4  5 | <argument name="helper\_method\_result" xsi:type="helper" helper="Vendor\CustomModule\Helper\Class::someMethod">  <param name="firstParam">firstValue</param>  <param name="secondParam">secondValue</param>  ...  </argument> |

**arguments**

<arguments> is a required container for <argument>. It does not have its own attributes.

|  |  |
| --- | --- |
| 1  2  3 | <arguments>  <argument name="css\_class" xsi:type="string">header links</argument>  </arguments> |

**Get Product from category.**

<https://magento.stackexchange.com/questions/310607/magento2-show-products-from-a-specific-category-on-homepage>

**Disable Module Output / Disable module**

<https://magento.stackexchange.com/questions/257852/add-module-enable-disable-functionality-in-magento-2>

**Sorting Products From Category –**

 // $products->setPageSize(3); // fetching only 10 products

        $products->addAttributeToSort('price', 'ASC');

        // for sorting product according to price, In low to high

        // $products->addAttributeToSort('price', 'desc');

        // for sorting product according to price, In high to low

**Resource Model -** <https://www.pierrefay.fr/formation-magento2/creer-un-model-base-de-donn%C3%A9e.html>

To Understand Model see above link.

Simple Example

<?php

namespace Pfay\Contacts\Controller\Test;

class Index extends \Magento\Framework\App\Action\Action

{

public function execute()

{

$contact = $this->\_objectManager->create('Pfay\Contacts\Model\Contact');

$contact->setName('Paul Dupond');

$contact->save();

die('test');

}

}

**getCollection**

<?php

namespace Meetanshi\Extension\Controller\Adminhtml\Index;

use Magento\Backend\App\Action;

class Index extends \Magento\Backend\App\Action

{

    public function execute()

    {

        $contactModel = $this->\_objectManager->create('Meetanshi\Extension\Model\Extension');

        $collection = $contactModel->getCollection()->addFieldToFilter('name', array('like' => 'Nitin'));

        foreach ($collection as $contact) {

            var\_dump($contact->getData());

            // echo "<h1>". $contact['name'] ."</h1>";

        }

        die('test');

    }

}

**Send Mail –**

<https://www.tutorialspoint.com/php/php_sending_emails.htm>

        $email = $\_REQUEST["email"];

        $emailDisplay = new \Magento\Framework\DataObject(array('text' => "$email"));

        $this->\_eventManager->dispatch('meetanshi\_extension\_send\_mail', ['email' => $emailDisplay]);

Error -

Exception #0 (Magento\Framework\Exception\RuntimeException): Type Error occurred when creating object: Meetanshi\Extension\Controller\Index\Index\Interceptor

Just name rename code or meta folder in generated folder to bkpcode or bkpmeta

Solution link -

<https://meetanshi.com/blog/solution-type-error-occurred-when-creating-object-in-magento-2/>

Link - <https://devdocs.magento.com/guides/v2.3/comp-mgr/cli/cli-upgrade.html>

# [**Get all dropdown and multi select attrinbutes in system configuration**](https://magento.stackexchange.com/questions/245974/get-all-dropdown-and-multi-select-attrinbutes-in-system-configuration)

<https://magento.stackexchange.com/questions/245974/get-all-dropdown-and-multi-select-attrinbutes-in-system-configuration>

How to get system.xml fields value.

<https://vrajeshpatel.in/2021/05/22/how-to-create-system-xml-configuration-and-get-data-from-system-configuration-in-magento-2/>

How to Filter product by price.

$collection->addAttributeToSelect('\*')->addFieldToFilter( 'price' , array('from' => $priceFrom, 'to' => $priceTo) );

            // $collection->addAttributeToSelect('\*')->addFieldToFilter( 'price' , array('from' => 30, 'to' => 50) )->setPageSize(5);

# How to add link in Category Menu in Magento 2

<https://webkul.com/blog/how-to-add-link-in-category-menu-in-magento-2/>

Magento Api - <https://blog.landofcoder.com/all-about-magento-2-api/>

How to add ajax add to cart

C:\wamp64\www\magento237\vendor\magento\module-catalog-widget\view\frontend\templates\product\widget\content\grid.phtml

Past this code at last of above file

<script type="text/x-magento-init">

    {

        "[data-role=tocart-form], .form.map.checkout": {

            "catalogAddToCart": {}

        }

    }

</script>

How to redirect on custom page

$location = $block->getBaseUrl() . 'checkout/cart/';

            header("Location: $location");

            exit;

How to call block on cms page > {{block class="Dharma\AdvanceSearch\Block\Index" template="Dharma\_AdvanceSearch::advance.phtml"}}

How to install sample data

<https://www.cloudways.com/blog/magento-2-sample-data/>

Step 1 > php bin/magento sampledata:deploy

Step 2 >

Use the Public key as your username and the Private key as your password.

Public Key: e042610264e17ebe7f5e7c25a6f44732

Private Key: 925c984b853aefaa4b12506b553091c6

After second step this message will be shown

**Do you want to store credentials for repo.magento.com in C:/wamp64/www/magento237/var/composer\_home/auth.json ? [Yn]**

**Than press > y**

Step 3 > php -dmemory\_limit=4G bin/magento setup:upgrade

Step 4 > php -dmemory\_limit=4G bin/magento setup:di:compile

Step 5 > php -dmemory\_limit=4G bin/magento setup:static-content:deploy -f

Step 6 > cache cleand and flush

**Magento defult theme location** > C:\wamp64\www\magento237\vendor\magento\theme-frontend-luma

**How to add image in phtml file**

First add image in this path > app\code\Vendor\module\view\frontend\web\images

Example >

app\code\Avologic\AdvanceSearch\view\frontend\web\images

Than write below code in phtml

<img src="<?php echo $this->getViewFileUrl('Avologic\_AdvanceSearch::images/second-1.png'); ?>">

Redirect in Magento

public function execute()

{

$resultRedirect = $this->resultRedirectFactory->create();

$resultRedirect->setPath('routename/controllerpath/controllername');

return $resultRedirect;

}

No need to declare $this->resultRedirectFactory in construct as its auto declared in construct of \Magento\Framework\App\Action\Action to which your custom controller should extend. Otherwise you'll get error during compilation: resultRedirectFactory is already declared

For deleting folder with all files and foder

* A rm -r foldername

Intall using composer / cmd

# General CLI installation

<https://devdocs.magento.com/extensions/install/>

payment-fee

composer require prince/magento2-paymentfee

if show invalidargumentException than change public key and private key in **auth.json.sample**

**Create log file in var/log folder**

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

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

$logger->addWriter($writer);

$logger->info('Simple Text Log');

$logger->info('Array Log'.print\_r($myArrayVar, true));