**ЧПУ**

Создаем 2 файла .htaccess в корне документа и один в папке **web**

в корень документа добавляем

Options +FollowSymLinks

IndexIgnore \*/\*

RewriteEngine on

RewriteCond %{REQUEST\_URI} !^/(web)

RewriteRule (.\*) /web/$1

RewriteCond %{REQUEST\_FILENAME} !-f

RewriteCond %{REQUEST\_FILENAME} !-d

RewriteRule . /web/index.php

в папке **web**

RewriteEngine On RewriteBase /

RewriteCond %{REQUEST\_FILENAME} !-f

RewriteCond %{REQUEST\_FILENAME} !-d

RewriteRule . index.php

в папку **web.config** убираем коменты с

'urlManager' => [

'enablePrettyUrl' => true,

'showScriptName' => true,

'rules' => [

],

],

И В REQUEST добавляем

'baseUrl' => '',

**МИГРАЦИИ**

yii migrate/create create\_post\_table --fields="author\_id:integer:notNull:foreignKey(user),category\_id:integer:defaultValue(1):foreignKey,title:string,body:text"

В проекте открываем модальное окно и в нем прописуем

yii migrate/create  - команда создания

create\_post\_table  - название таблицы

author\_id:integer:notNull:foreignKey(user), - foreign key для связи

title:string – обычное поле(обычные поля пишутся через точку с запятой )

Примеры всех типов строк в таблице

**pk**: an auto-incremental primary key type, will be converted into "int(11) NOT NULL AUTO\_INCREMENT PRIMARY KEY"

bigpk: an auto-incremental primary key type, will be converted into "bigint(20) NOT NULL AUTO\_INCREMENT PRIMARY KEY"

unsignedpk: an unsigned auto-incremental primary key type, will be converted into "int(11) UNSIGNED NOT NULL AUTO\_INCREMENT PRIMARY KEY"

**char**: char type, will be converted into "**char(1)"**

**string**: string type, will be converted into "**varchar(255)**"

**text**: a long string type, will be converted into "**text**"

**smallint**: a small integer type, will be converted into "**smallint(6)"**

**integer**: integer type, will be converted into "**int(11)"**

**bigint**: a big integer type, will be converted into "**bigint(20)"**

**boolean**: boolean type, will be converted into "**tinyint(1)"**

**float`**: float number type, will be converted into "**float**"

**decimal**: decimal number type, will be converted into "**decimal**"

**datetime**: datetime type, will be converted into "**datetime**"

**timestamp**: timestamp type, will be converted into "**timestamp**"

**time**: time type, will be converted into "**time**"

**date**: date type, will be converted into "**date**"

**money**: money type, will be converted into "**decimal(19,4)"**

**binary**: binary data type, will be converted into "**blob**"

PHP YII MIGRATE - это завершающая команда которая создаст миграции

**Подключение ASSET и Перенос Шаблона**

После того как мы сделали ЧПУ Переносим все **js css img** fonts в папку web

Переименовываем layaout **main.php** в **\_main.php** и создаем новый **main.php**

далее шаблон верстки переносим в **main.php** и подключаем стили в **asset**

public $depends = [

'yii\web\YiiAsset',

'yii\bootstrap\BootstrapPluginAsset',

];

подключает **jquery bootstrap.min.css bootsrap.js** так что если стоят зависимости то не нужено подключать еще раз в ассетах

Далее По аналогии с стандартным шаблоном мы подключаем все методы

<?php

/\* @var $this \yii\web\View \*/

/\* @var $content string \*/

use yii\helpers\Html;

use yii\bootstrap\Nav;

use yii\bootstrap\NavBar;

use yii\widgets\Breadcrumbs;

use app\assets\AppAsset;

AppAsset::register($this);

?>

<?php $this->beginPage() ?>

Язык

<html lang="<?= Yii::$app->language ?>">

и в **config>web** добовляем

'basePath' => dirname(\_\_DIR\_\_),

'bootstrap' => ['log'],

'language' => 'ru-Ru', -- **вот это значение**

в **<head>** помещяем

<meta charset="<?= Yii::$app->charset ?>">

<meta name="viewport" content="width=device-width, initial-scale=1">

<?= Html::csrfMetaTags() ?>

<title><?= Html::encode($this->title) ?></title>

<?php $this->head() ?>

По шаблону закрываем все открытые теги

СКРИПТЫ ДЛЯ IE 9

Если у нас подключаются скрипты которые треубуют Ie8 и ниже то мы создаем новвый aseet

<?php

/\*\*

\* @link <http://www.yiiframework.com/>

\* @copyright Copyright (c) 2008 Yii Software LLC

\* @license <http://www.yiiframework.com/license/>

\*/

namespace app\assets;

use yii\web\AssetBundle;

/\*\*

\* @author Qiang Xue <qiang.xue@gmail.com>

\* @since 2.0

\*/

class ltAppAsset extends AssetBundle { // не забыть поменять иммя класса

public $basePath = '@webroot';

public $baseUrl = '@web';

public $js = [

'js/html5shiv.js',

'js/respond.min.js',

];

public $jsOptions = [

'condition' => 'lte IE9', // ЭТО УСЛОВИЕ

'position' => \yii\web\View::POS\_HEAD // ЭТО ПОЗИЦИЯ ГДЕ ОН ПОДКЛЮЧИТСЯ

];

}

И подключаем его в наш шаблон

**ИЛИ ЕСТЬ 2 ВАРИАНТ ПОДКЛЮЧИТЬ ПРЯМО В ШАБЛОН**

<?php

$this->registerJsFile('js/html5shiv.js', ['position' => \yii\web\View::POS\_HEAD, 'condition' => 'lte IE9']);

$this->registerJsFile('js/respond.min.js', ['position' => \yii\web\View::POS\_HEAD, 'condition' => 'lte IE9']);

?>

**LaYaout(Шаблон)**

Виды выводятся в шаблон с помощью тега:

<?= $content ;?>

**Подключение layout для отдельных страниц в контроллере мы указуем если для всего контроллера**

public $layout = 'aa';

**Если для определенного вида**

public function actionIndex()

{

$this->layout = 'aa';

return $this->render('index');

}

**КАРТИНКИ**

**1)** Для все картинок мы можем сделать относительный путь

**добавить / слеш в начале пути картинки**

**2)** выводим картинки **подключаем**

use yii\helpers\Html;

<?= Html::img("@web/images/products/{$hit->img}", ['alt' => $hit->name])?>

<?= Html::img("@web/images/home/sale.png", ['alt' => 'Распродажа', 'class' => 'new'])?>

**@web - алиас на папку web**

**{$hit->img} - имя картинки из базы**

**['alt' => $hit->name] - альт из базы**

**ССЫЛКИ(Роутинг)**

<a href="<?= \yii\helpers\Url::to(['category/view', 'id' => $category['id']]) ?>">

<?= Html::a(Yii::t('app', 'Log-in'), ['/user/account/login']); ?>

создаем ссылку указуем куда она ведет в контроллер **category** вид **view**

дальше (по желанию) указуем как она будет показываться

'urlManager' => [

'enablePrettyUrl' => true,

'showScriptName' => false,

'rules' => [

'category/<id:\d+>' => 'category/view'**, // здесь определяется как будет выглядеть наша ссылка**

],

],

**МОДЕЛИ ТАБЛИЦ**

1) **Смотрим какое количество таблиц у нас используется в нашей БД и определяем связи**

у нас есть две таблицы которые связзаные между собой

создаем две модели таблицы

где указываем **ActiveRecord** расширяемый класс для модели в бд

class Category extends ActiveRecord

в классе таблицы определяем к какой таблице мы будем обращаться

public function getProducts(){

return $this->hasMany(Product::className(), ['category\_id' => 'id']);

}

говорит о том что наша таблица категории будет иметь в себе продукты , также hasMany гвоорит о том что у одной категории может быть множество продуктов

Product::className() - вызываем модель подключающейся таблицы

(Category::className(), ['id' => 'category\_id']) - указует на то что в таблице категорий поле 'id' будет свзяаано с полем 'category\_id' в таблице категорий

<?php

namespace app\models;

use yii\db\ActiveRecord;

class Category extends ActiveRecord{

public static function tableName(){

return 'category'; // указывает таблицу которую мы будем использовать

}

public function getProducts(){

return $this->hasMany(Product::className(), ['category\_id' => 'id']);

}

}

<?php

namespace app\models;

use yii\db\ActiveRecord;

class Product extends ActiveRecord{

public static function tableName(){

return 'product'; // указывает таблицу которую мы будем использовать

}

public function getCategory(){

return $this->hasOne(Category::className(), ['id' => 'category\_id']);

}

}

дальше идем в config db.php

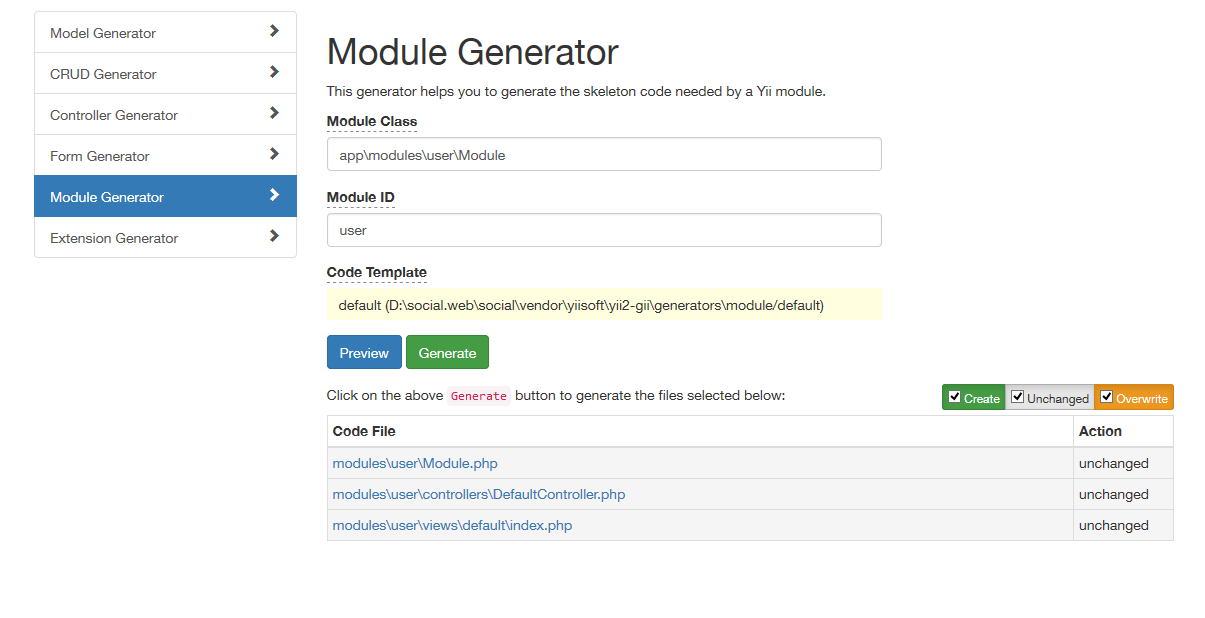
**Всего по немногу**

'defaultRoute' => 'category/index' – **web.php – создает домашнюю страницу по умолчанию**

**МОДУЛИ**

Cоздаем модуль через **GII**

для этого заходим в gii и



после этого нам нужно в папку web.php в $config сохранить получеенный нами код

'modules' => [

'user' => [

'class' => 'app\modules\user\Module',

],

],

в котором мы обьявляеем наш подключенный моудль

**РЕГИСТАРЦИЯ**

для Регитсрации мы используем Yii::$app->user - метод свойства $app

метод user мы так же подключаем в web.php   
'user' => [

'identityClass' => 'app\models\User', // путь к модели которую мы должны использовать

'enableAutoLogin' => true, // автологин

'loginUrl' => ['site/login'] // - нужно направить на наш контроллер где происходит логин

],

**1.МОДЕЛЬ USER**