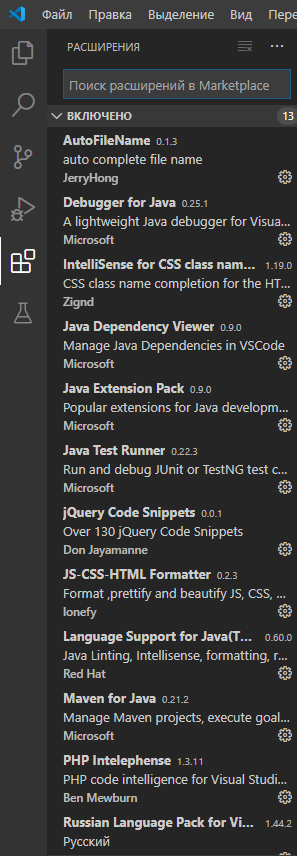
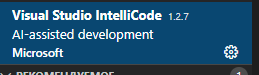
1.Plugins for Visiual studio code





2.Github мы при создании репозитория добавляем строку

git remote add origin

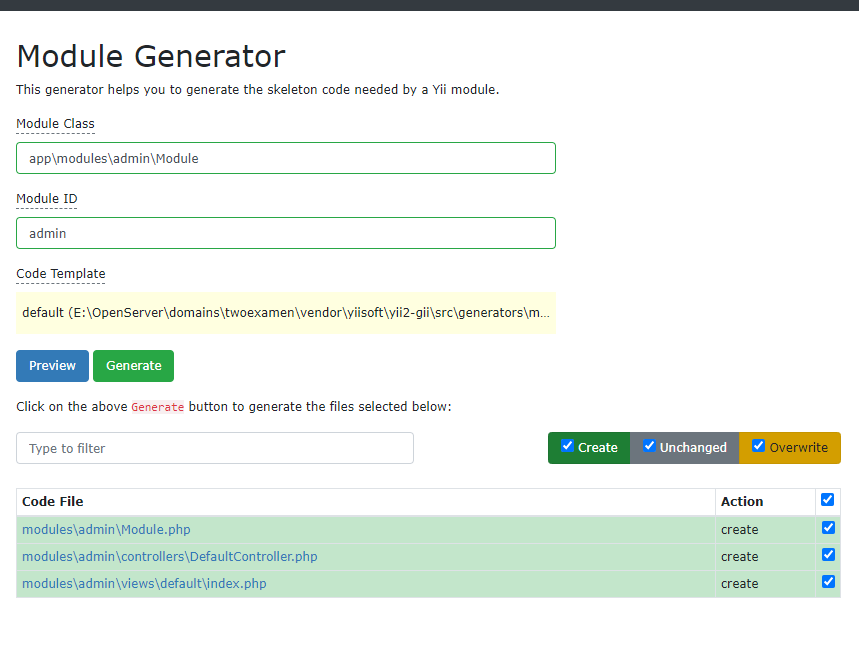
[https://ketrin1777:Aa385987@github.com/ketrin1777/](https://ketrin1777:Aa385987@github.com/ketrin1777/-PHP.git)

[https:///ketrin1777:Aa385987@github.com/ketrin1777/mvctest.git](https://ketrin1777:Aa385987@github.com/ketrin1777/-PHP.git)

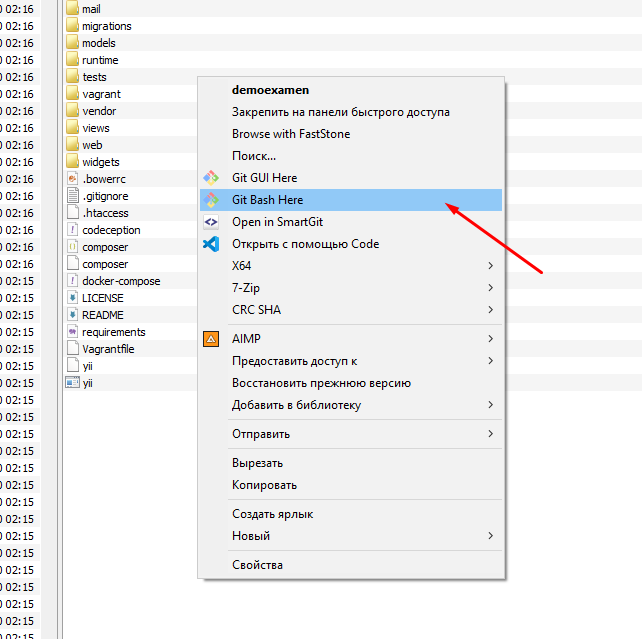
…

План создания сайта:

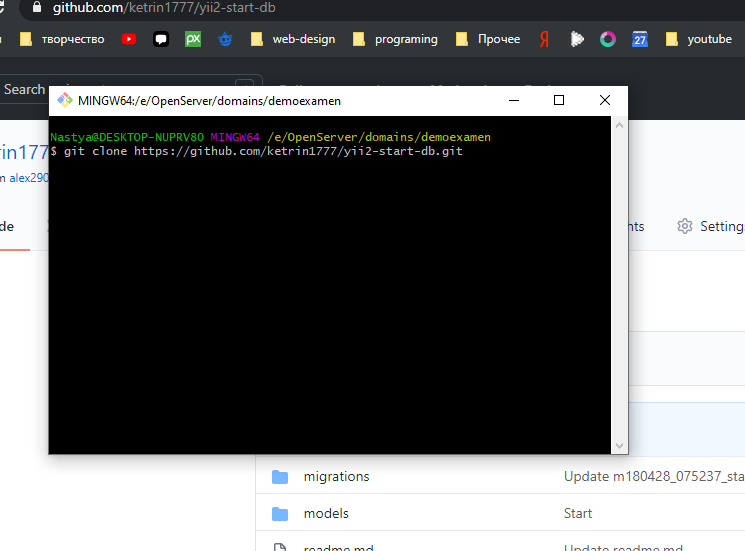
1. Установка фраемворка и подключение БД
2. Без web (Загуглить Yii без web)
3. Gii (генератор создания модуля админки)



1. Стартовая база <https://github.com/ketrin1777/yii2-start-db/blob/master/readme.md>



Перемещаем из папки, которая создалась, файлы в корень (заменяем все) перемещаем + shift



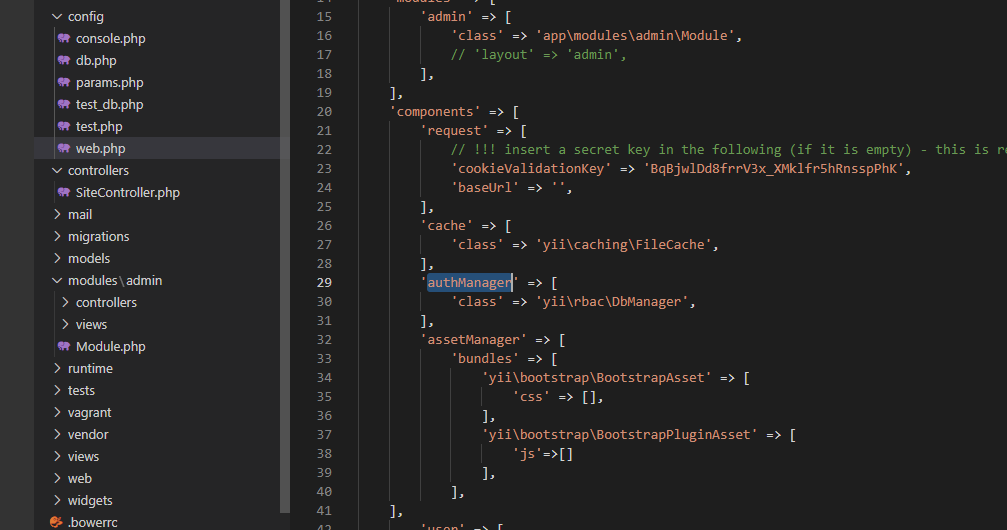
1. Создание RBAC [file:///D:/help/yii-docs-2.0-ru/guide-security-authorization.html](file://D:\\help\\yii-docs-2.0-ru\\guide-security-authorization.html)

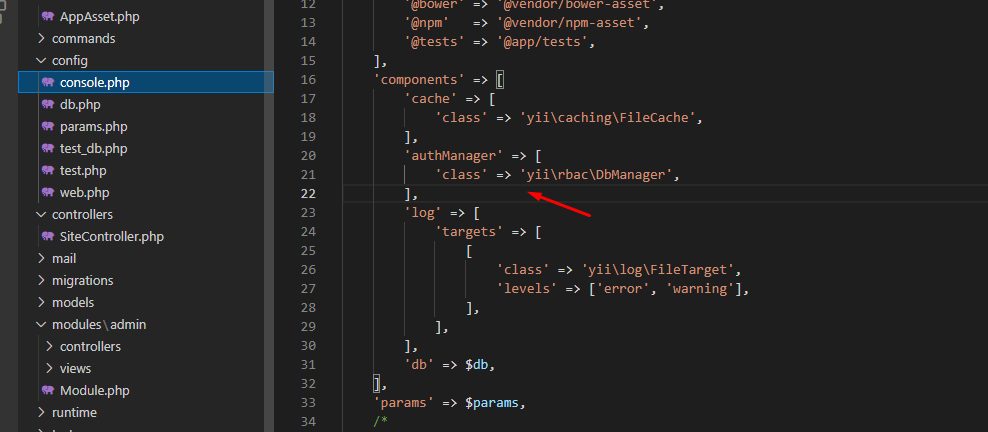
Сперва в config/web.php и в config/console.php в 'components' прописываем следующее

'authManager' => [

'class' => 'yii\rbac\DbManager',

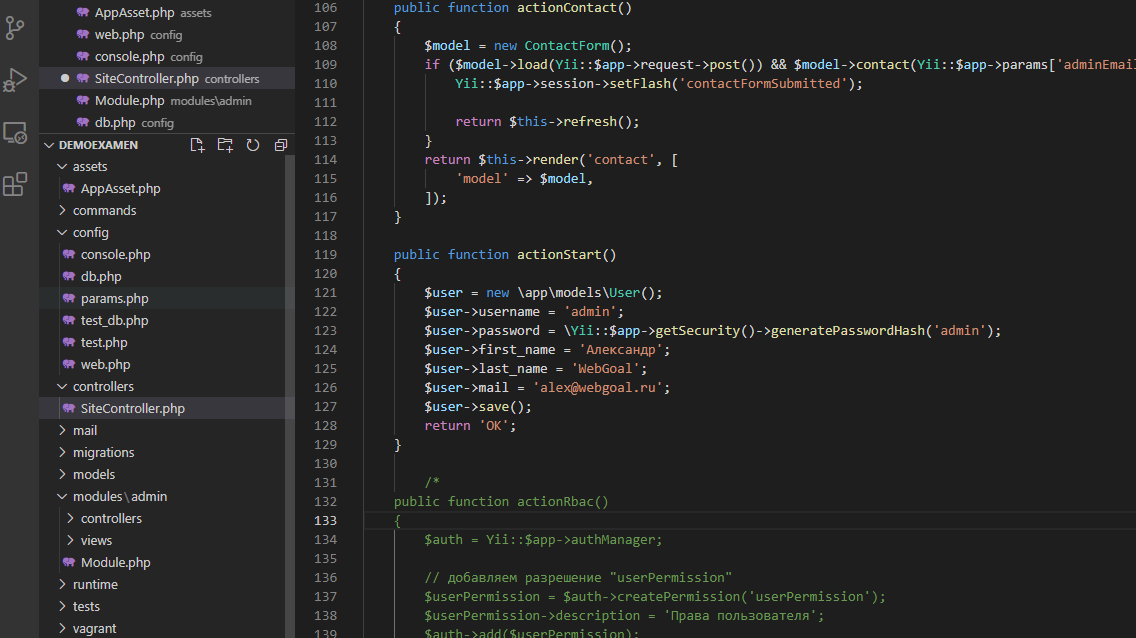
],





1. Создание роли пользователя и администратора и привязка пользователя к администратору

Потом в controllers/SiteController.php создаем пользователя админа (себя)



public function actionStart()

{

$user = new \app\models\User();

$user->username = 'admin';

$user->password = \Yii::$app->getSecurity()->generatePasswordHash('admin');

$user->first\_name = 'Александр';

$user->last\_name = 'WebGoal';

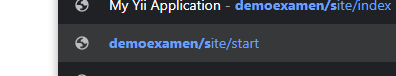
$user->mail = 'alex@webgoal.ru';

$user->save();

return 'ОК';

}

После этого проверяем что все получилось по следующему пути



твой\_домен/site/start

Перед тем как создать роли, мы создадим таблицу мигрируя, прописав команду:

yii migrate --migrationPath=@yii/rbac/migrations

После этого создадим роли [file:///D:/help/yii-docs-2.0-ru/guide-security-authorization.html](file://D:\\help\\yii-docs-2.0-ru\\guide-security-authorization.html) и наследование ролей (права), для этого, после создания админа, сотрем предыдущий код и заменим его на следующее:

public function actionRbac()

{

$auth = Yii::$app->authManager;

// добавляем разрешение "userPermission"

$userPermission = $auth->createPermission('userPermission');

$userPermission->description = 'Права пользователя';

$auth->add($userPermission);

// добавляем разрешение "adminPermission"

$adminPermission = $auth->createPermission('adminPermission');

$adminPermission->description = 'Права администратора';

$auth->add($adminPermission);

// добавляем роль "user" и даём роли разрешение "userPermission"

$user = $auth->createRole('user');

$user->description = 'Пользователь';

$auth->add($user);

$auth->addChild($user, $userPermission);

// добавляем роль "admin" и даём роли разрешение "adminPermission"

// а также все разрешения роли "user"

$admin = $auth->createRole('admin');

$admin->description = 'Администратор';

$auth->add($admin);

$auth->addChild($admin, $adminPermission);

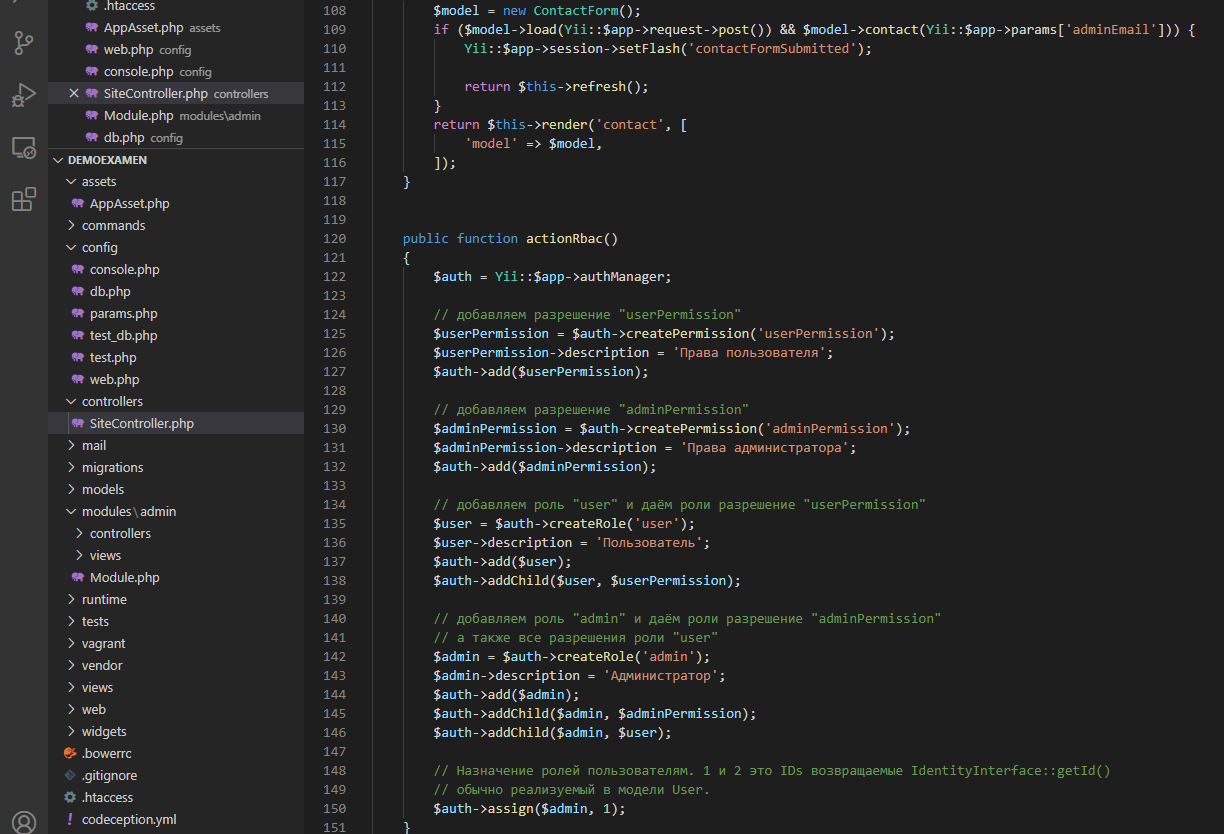
$auth->addChild($admin, $user);

// Назначение ролей пользователям. 1 и 2 это IDs возвращаемые IdentityInterface::getId()

// обычно реализуемый в модели User.

$auth->assign($admin, 1);

}



После этого проверяем что все получилось по следующему пути



твой\_домен/site/rbac

Сделаем так, чтобы в админку можно было попасть через авторизацию, для этого modules/admin/Modules.php пропишем следующий код:

public function behaviors()

{

return [

'access' => [

'class' => AccessControl::className(),

'rules' => [

[

'allow' => true,

'roles' => ['admin'],

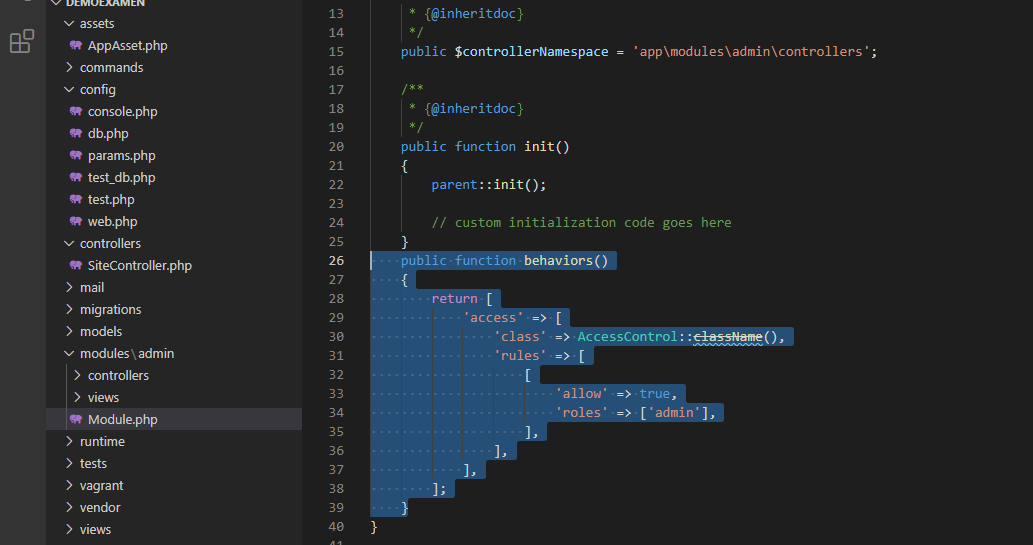
],

],

],

];

}



В config/web.php подключается модуль для работы в основном придложения

'modules' => [

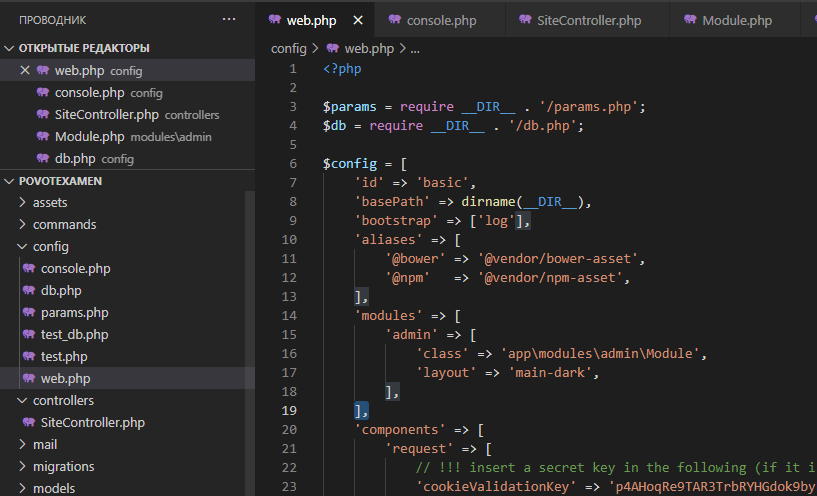
'admin' => [

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

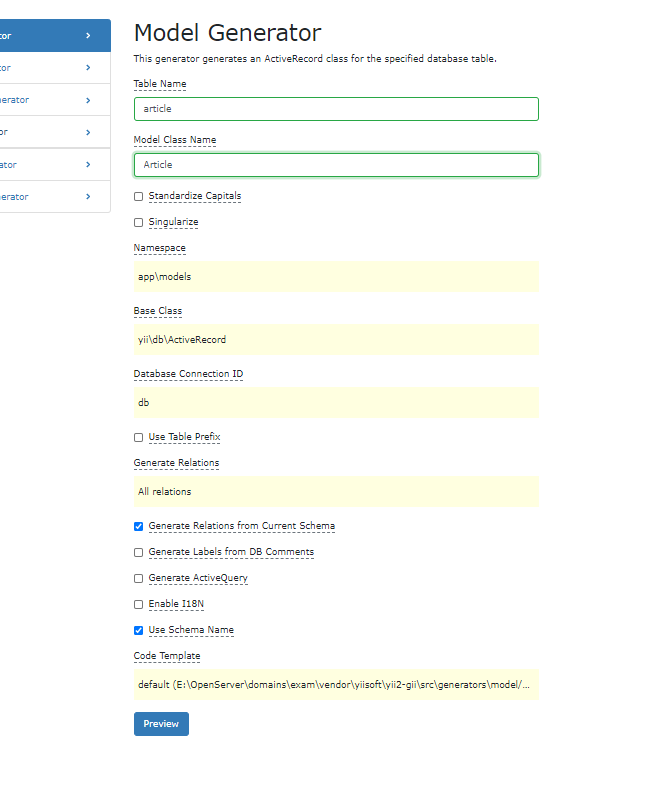
// 'layout' => 'main-dark',

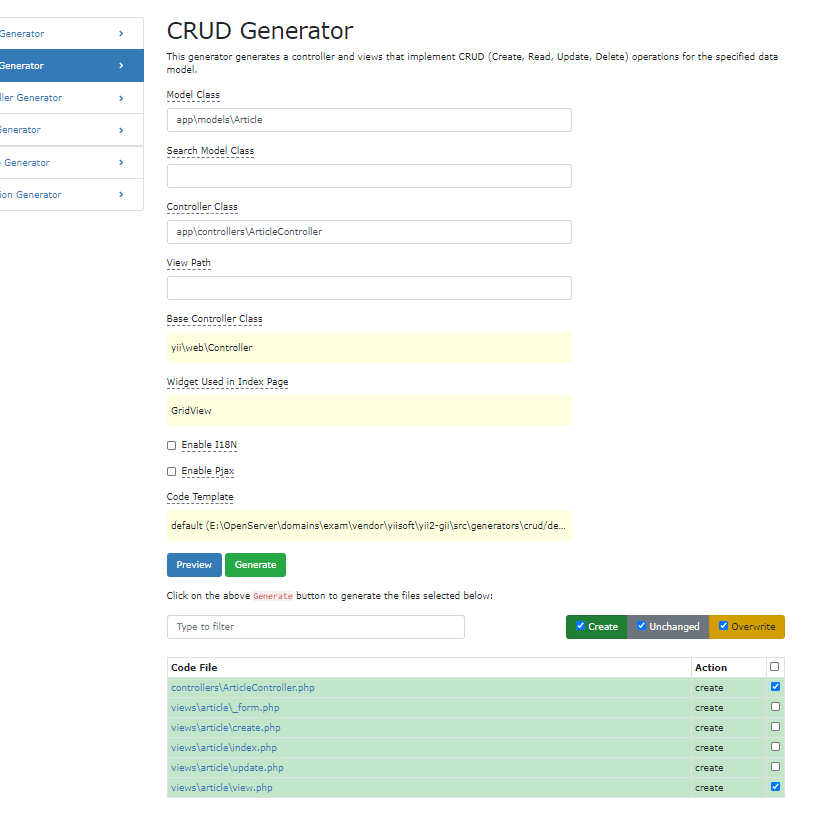
],

],

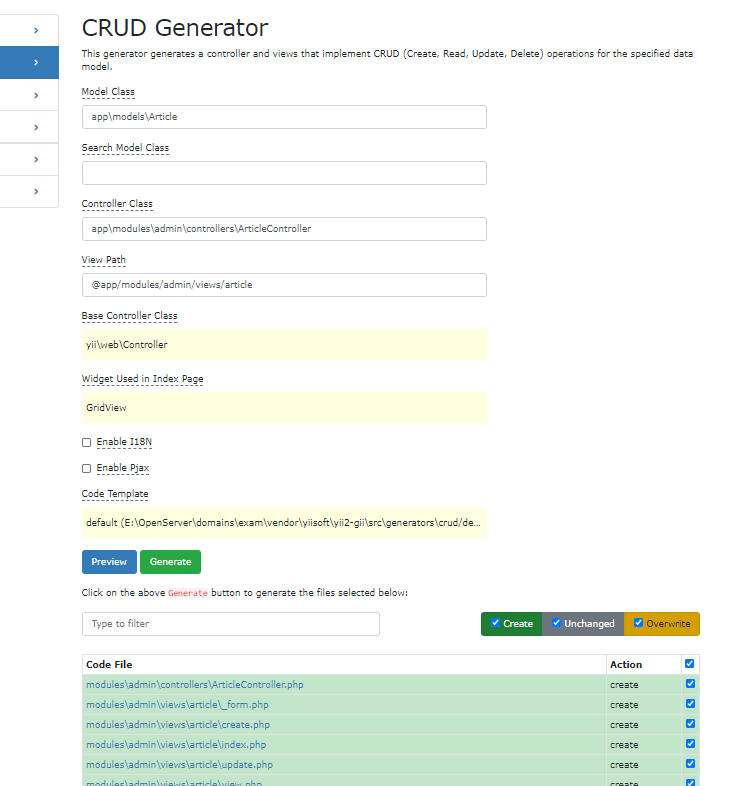


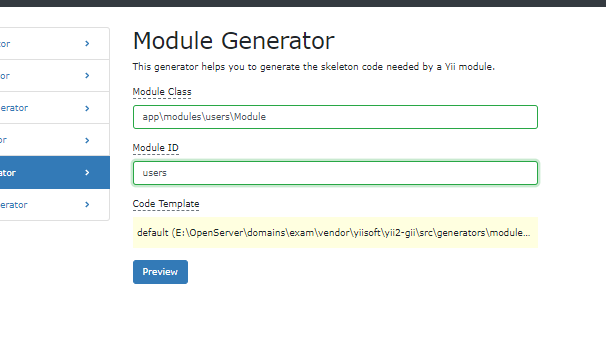
1. Верстка шаблона
2. Создание страниц просмотра контента (модель и генерация контроллера со страницей отображения данных)

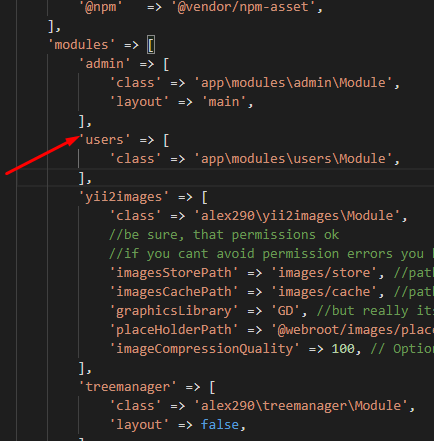




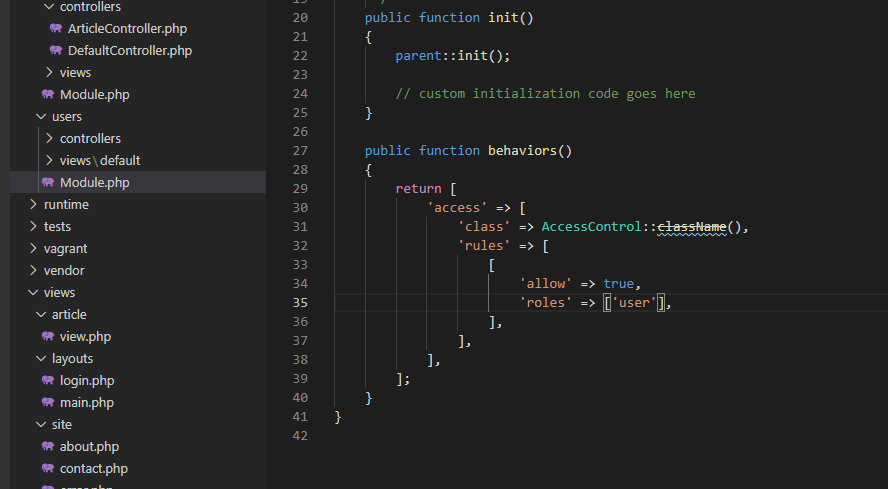
б) Создание админки (генерируем примерно следующий образом, модель второй раз не создаем, если генерировали обычные страницы)







[file:///D:/help/yii-docs-2.0-ru/guide-security-authorization.html](file://D:\\help\\yii-docs-2.0-ru\\guide-security-authorization.html)



public function behaviors()

{

return [

'access' => [

'class' => AccessControl::className(),

'rules' => [

[

'allow' => true,

'roles' => ['user'],

],

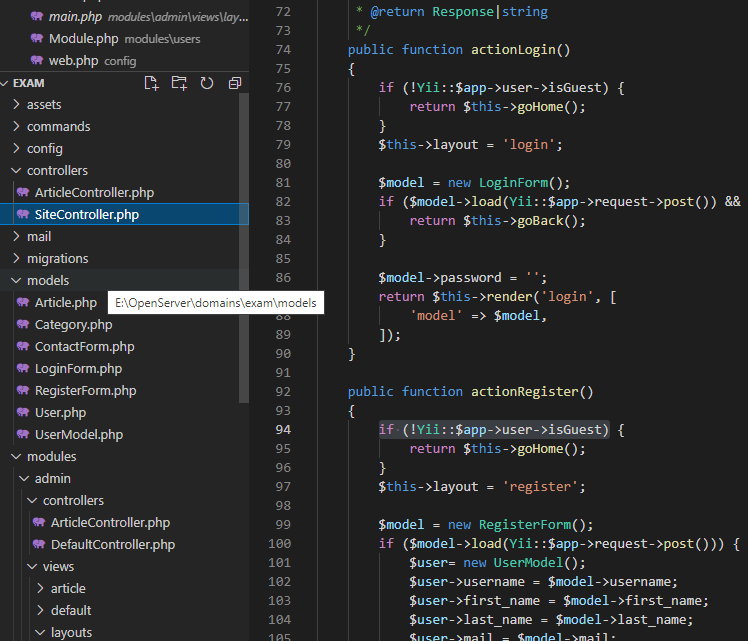
],

],

];

}

1. Загрузка шаблона администратора https://github.com/alex290/yii2-admin





Удалить заявку в index:

[

'value' => function ($data) {

$html = Html::a('Удалить', ['/users/zayavka/delete', 'id' => $data->id], ['class' => 'ml-2 mr-2 btn btn-outline-danger', 'data-method' => "post", 'data-confirm' => "Вы уверены, что хотите удалить этот элемент?"]);

return $html;

},

'format'=>'raw',

],

Взято с источника:

<https://medium.com/@alivander/%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0-%D1%81-git-%D0%B8-github-%D0%B2-%D1%80%D0%B5%D0%B4%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%B5-%D0%BA%D0%BE%D0%B4%D0%B0-atom-bd09ae13d4c4>

3.PHP commands

Error\_reporting(-1);

1. Мои заметки по ПРОГРАММИРОВАНИЮ сайта

1 ЧАСТЬ

* В OpenServer создаю папку, и в этой папке пишу эту команду:

При создании YIi2, в композере мы пишем без слова basic, как говорит нам документация, чтобы установилось в корень. Итог: composer create-project --prefer-dist yiisoft/yii2-app-basic ./

!(или загуглить Yii2 без веб)

* в папке web в .htaccess:

RewriteEngine On RewriteBase /

RewriteCond %{REQUEST\_FILENAME} !-f

RewriteCond %{REQUEST\_FILENAME} !-d

RewriteRule . index.php

.htaccess в корне сайта:

Options +FollowSymLinks

IndexIgnore \*/\*

RewriteEngine on

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

RewriteRule ^assets/(.\*)$ /web/assets/$1 [L]

RewriteRule ^css/(.\*)$ web/css/$1 [L]

RewriteRule ^js/(.\*)$ web/js/$1 [L]

RewriteRule ^images/(.\*)$ web/images/$1 [L]

RewriteRule (.\*) /web/$1

RewriteCond %{REQUEST\_FILENAME} !-f

RewriteCond %{REQUEST\_FILENAME} !-d

RewriteRule . /web/index.php

в конфиге web.php в 'request' добавляем строчку:

'baseUrl'=> '',

* Удаляю bootstrap 3. В config/web.php пишу в 'components' => [ следующий код:

'assetManager' => [

'bundles' => [

'yii\bootstrap\BootstrapAsset' => [

'css' => [],

],

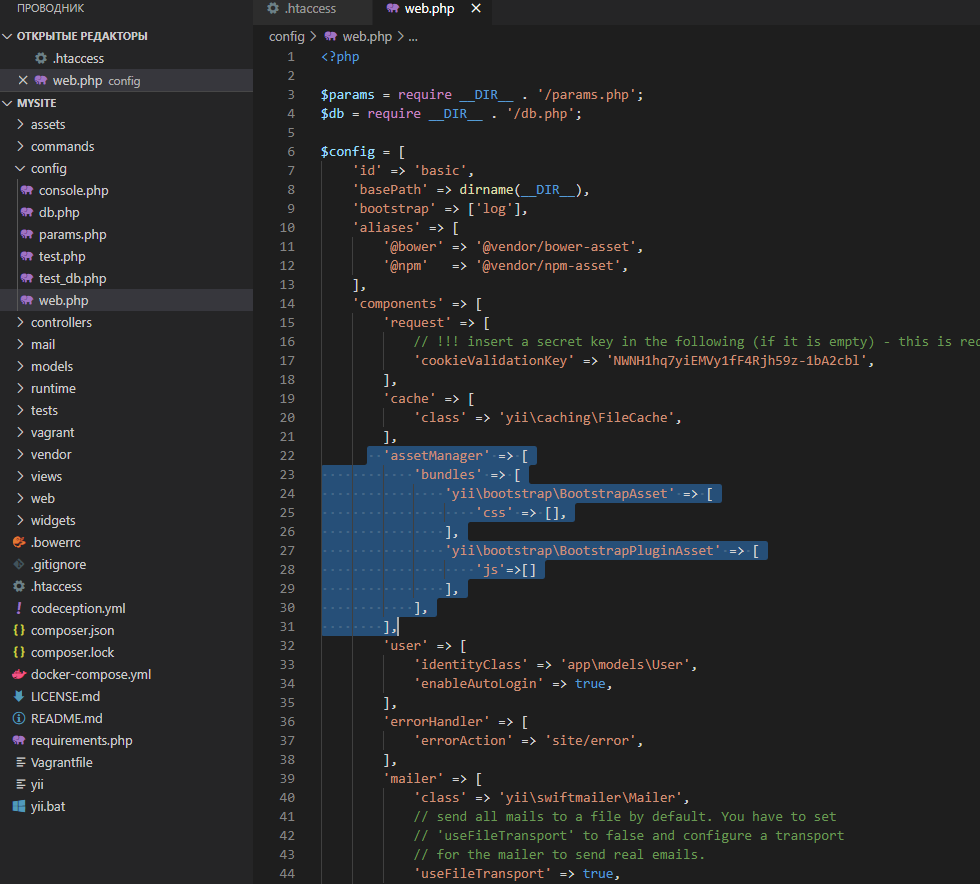
'yii\bootstrap\BootstrapPluginAsset' => [

'js'=>[]

],

],

],



 переустановка BootstrapAsset чтобы не грузились собственные стили

public function init()

    {

        parent::init();

        // переустановка BootstrapAsset чтобы не грузились собственные стили

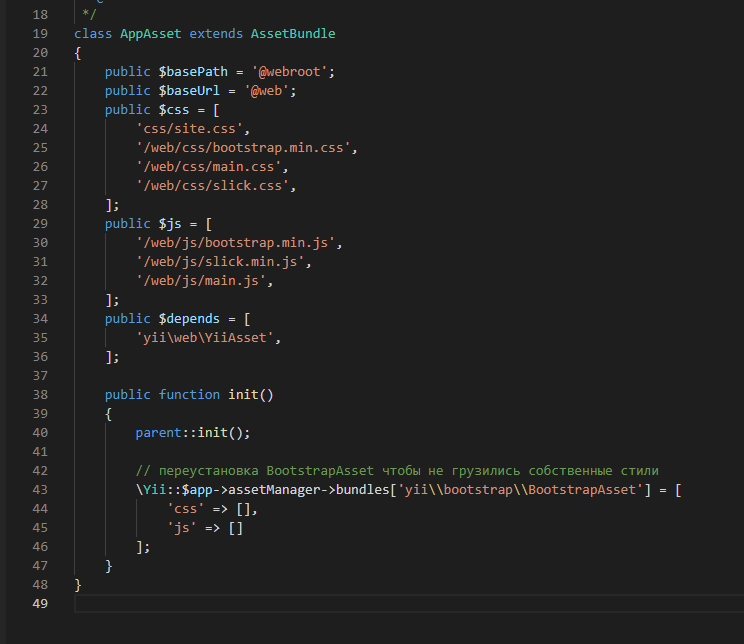
        \Yii::$app->assetManager->bundles['yii\\bootstrap\\BootstrapAsset'] = [

            'css' => [],

            'js' => []

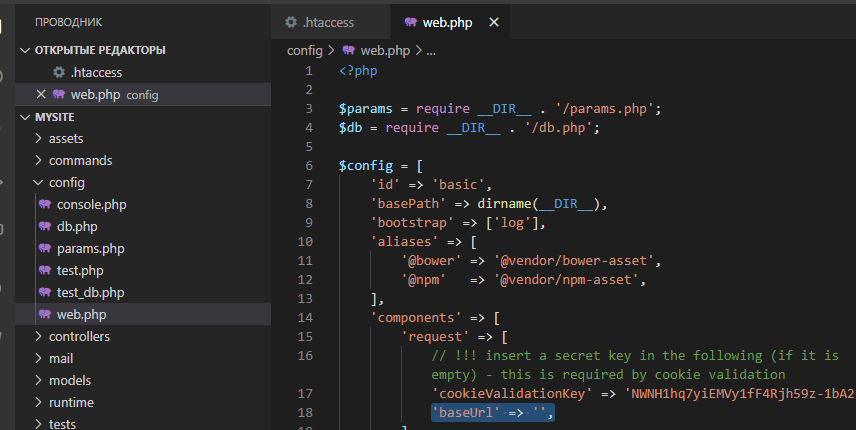
        ];

    }

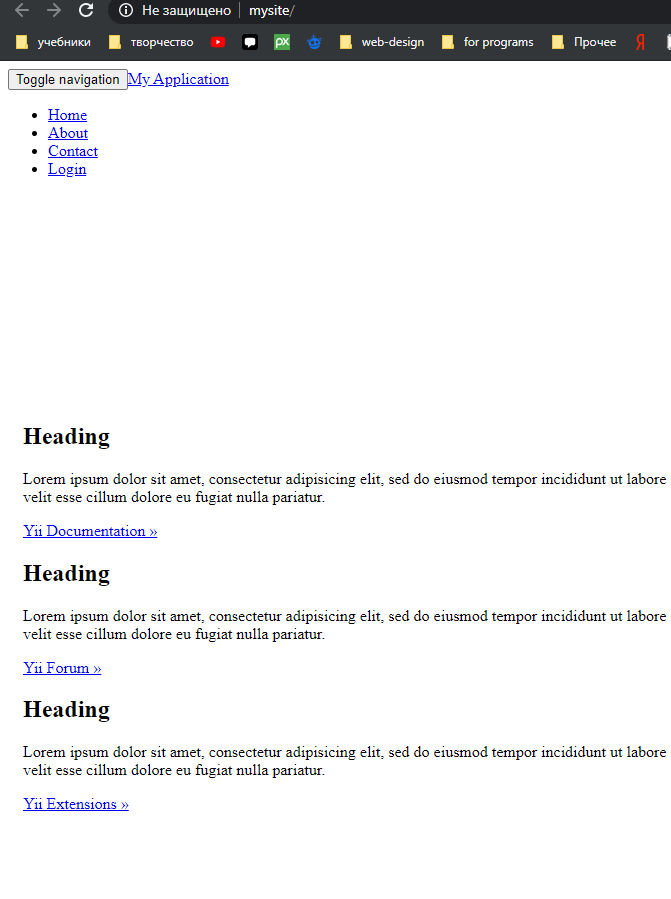


* Добавляем url, чтобы показывало что мы не через url, а web заходим на сайт. В том же месте в

'components' => [ мы пишем 'baseUrl' => '',



* Вот так выглядит сейчас наш будущий сайт



* Расскоментируем в config/web.php urlManage и добавляем строчку 'enableStrictParsing' => false,

'urlManager' => [

'enablePrettyUrl' => true,

'showScriptName' => false,

'enableStrictParsing' => false,

'rules' => [

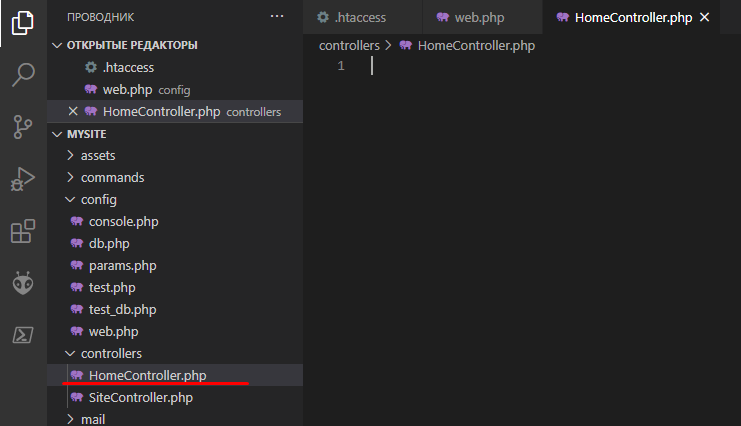
],

],

2 ЧАСТЬ - ПЕРЕНОС ШАБЛОНА

* Создаем контроллер, который будет отвечать за главную страницу.

В controllers создаем HomeController.php



* Добавляем туда следующий код:

<?php

namespace app\controllers;

class HomeController extends AppController{

public function actionIndex()

{

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

}

}

?>

* Cоздаем общий контроллер AppController.php и добавляем туда следующий код:

<?php

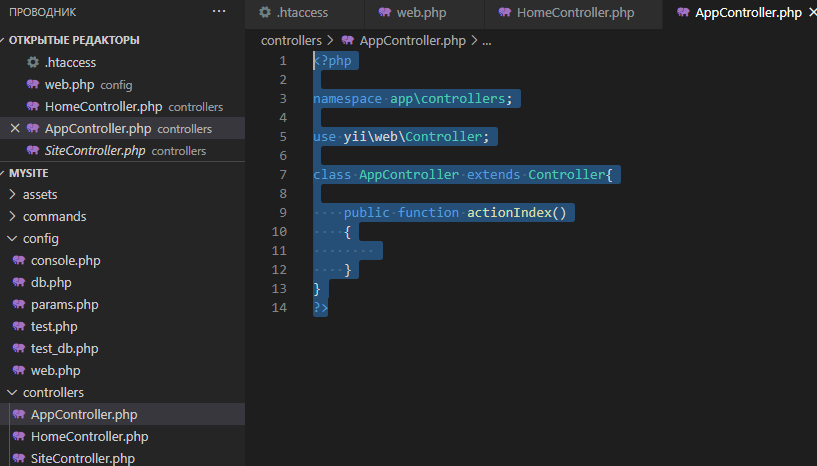
namespace app\controllers;

use yii\web\Controller;

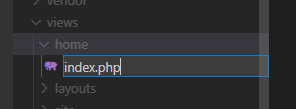
class AppController extends Controller{

}

?>



* return $this->render('index'); - это функция означает то, что мы рендерим вид, и сейчас мы создадим сам вид. В папке views создаём новую папку home, и в ней создаем файл index.php



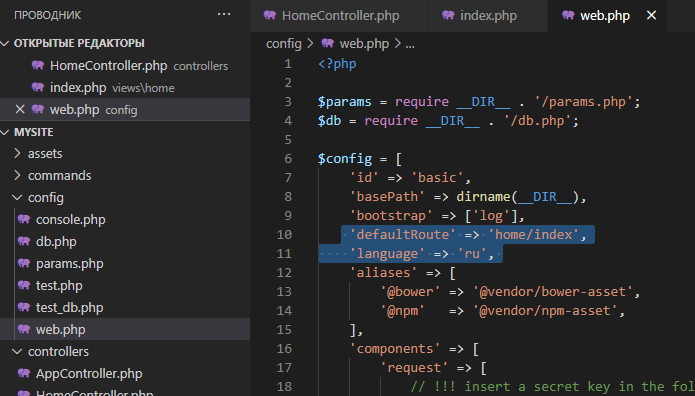
* В этот файл (index.php) добавляем следующий код:

<code><?= \_\_FILE\_\_ ?><code>

* В config/web.php добавляем в $config исходный путь и русский язык следующий код:

'defaultRoute' => 'home/index',

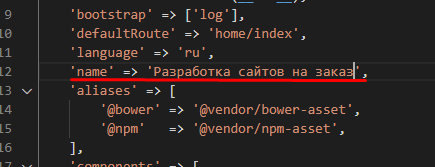
'language' => 'ru',



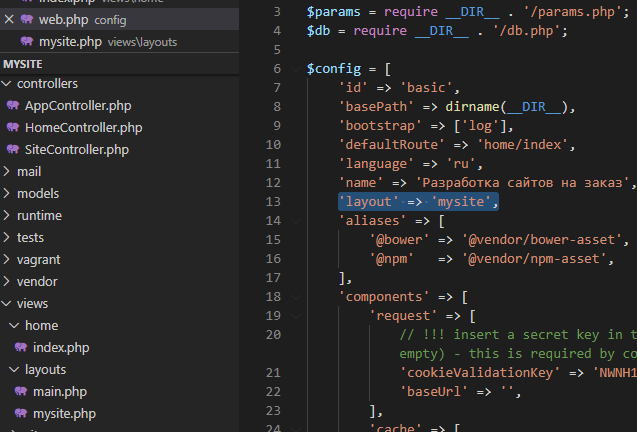
* Там же добавляем название сайта в конфигах приписываем строчку

'name' => '',

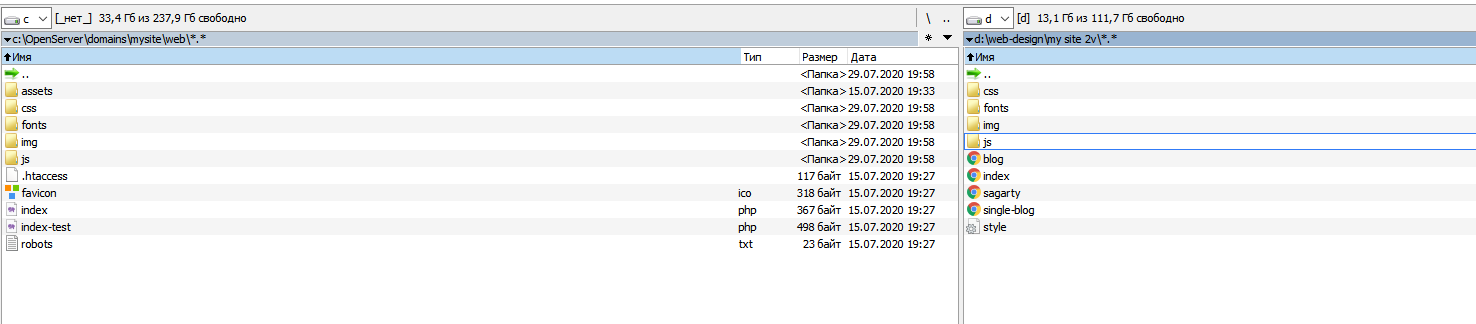
Пр:



* В папке layouts создаем любой php файл. Там будет наше содержимое сайта. В config/web.php добавляем в $config строчку кода 'layout' => 'mysite',





* Все папки, которые мы создавали(css, js, fonts, img) мы копируем в папку web 
* Дальше мы подгоняем свою верстку под свой сайт! Исправляем ошибки

3 ЧАСТЬ - ПЕРЕНОС ШАБЛОНА 2

* Удаляем старый Bootstrap. В AppAsset пишем следующий код:

public function init()

{

parent::init();

// переустановка BootstrapAsset чтобы не грузились собственные стили

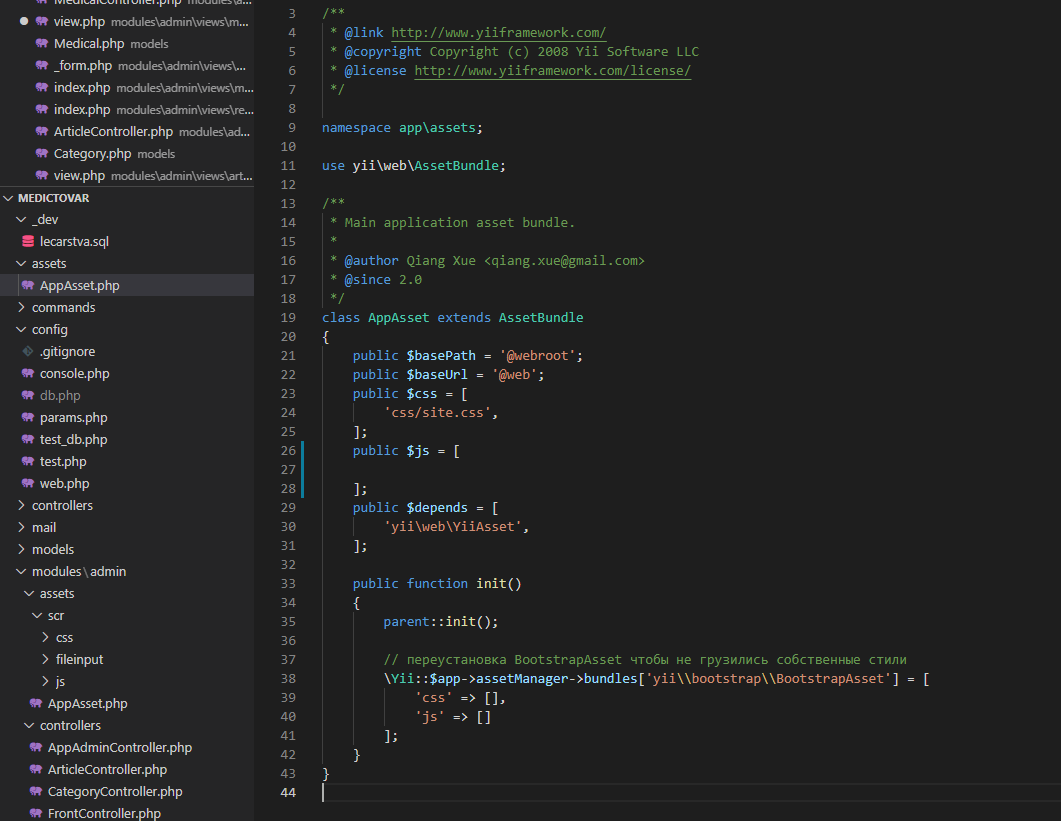
\Yii::$app->assetManager->bundles['yii\\bootstrap\\BootstrapAsset'] = [

'css' => [],

'js' => []

];

}



* В код, где мы у нас находится логотип в a href”...” вставляем код:

<a href="<?= \yii\helpers\Url::home()?>"

Это для того чтобы добиться на сервере эффект слэша

4 ЧАСТЬ - ПОДКЛЮЧЕНИЕ БД

* <https://github.com/ketrin1777/yii2-start-db> - мы подключаем уже готовые таблицы с моделями: пользователь, статья, категория

5 ЧАСТЬ - РАБОТА С КАРТИНКАМИ

* <https://github.com/alex290/yii2-images> - подключаем уже готовую модель картинок

Когда подключаем картинки в бд мы вставляем код в нужную нам модель(напомним что модели потом мы создаем в gii т.е. из таблицы бд автоматически переделывается в некий код, который и называется моделью)

    public function upload()

    {

        if ($this->validate()) {

            $path = 'upload/images' . $this->imageFile->baseName . '.' . $this->imageFile->extension;

            $this->imageFile->saveAs($path);

            $this->attachImage($path);

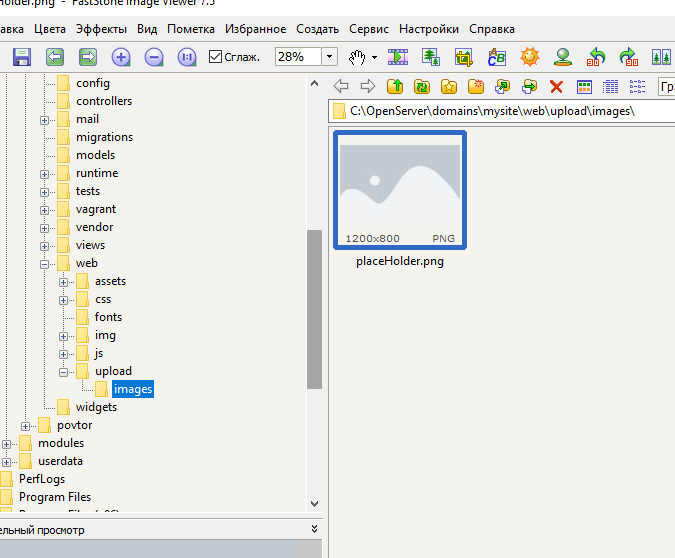
            return true;

        } else {

            return false;

        }

}

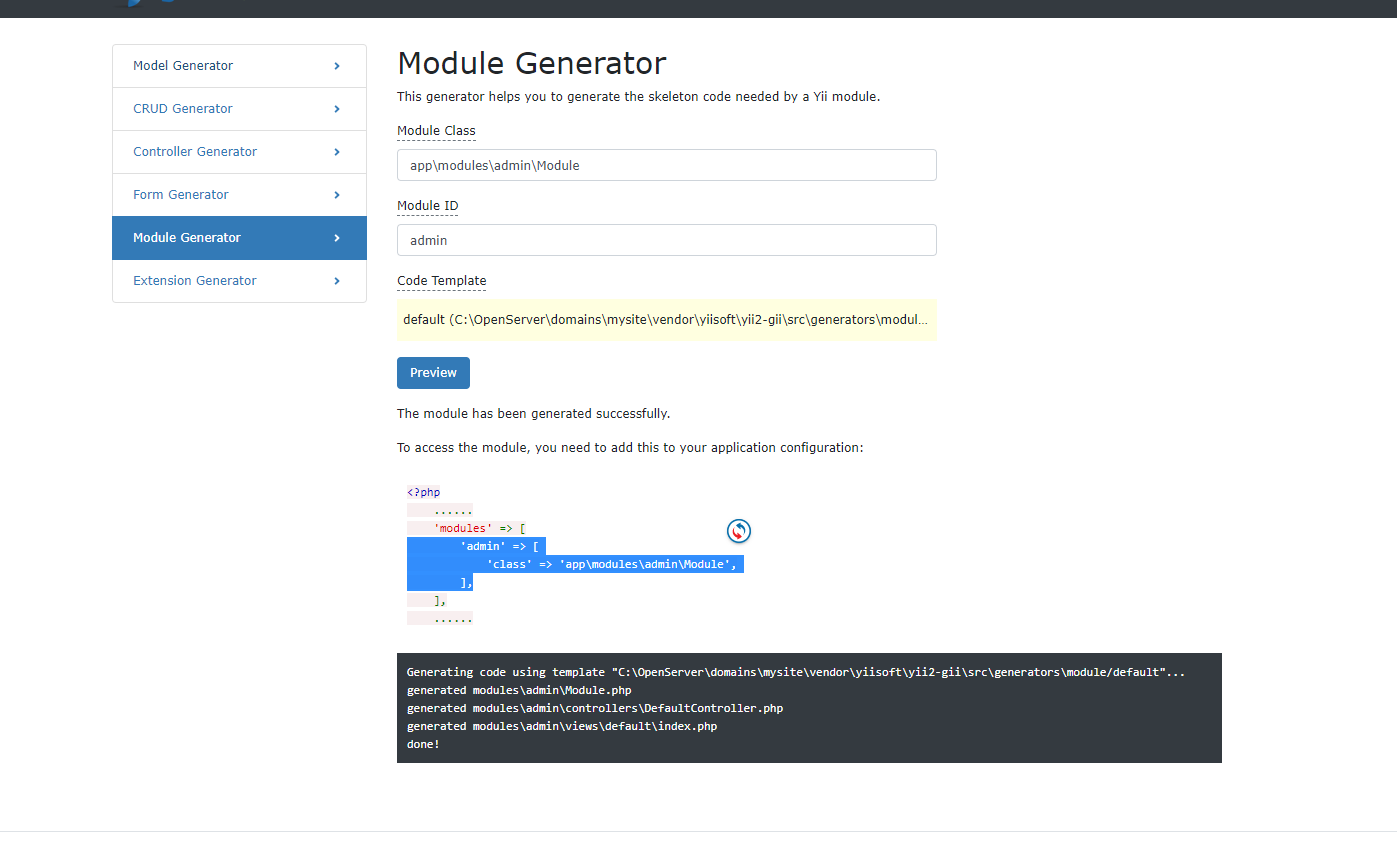


6 ЧАСТЬ - СОЗДАНИЕ АДМИНКИ

* Для начала мы сгенерируем модуль админки. Для этого прописываем ваш\_домен/gii

В Module Class пишем app\modules\admin\Module

А в Module ID пишем admin



* В modules/admin/Model.php мы закрываем доступ другим пользователям. Напишем следующий код:

public function behaviors()

{

return [

'access' => [

'class' => AccessControl::className(),

'only' => ['login', 'logout', 'signup'],

'rules' => [

[

'allow' => true,

'actions' => ['login', 'signup'],

'roles' => ['?'],

],

[

'allow' => true,

'actions' => ['logout'],

'roles' => ['@'],

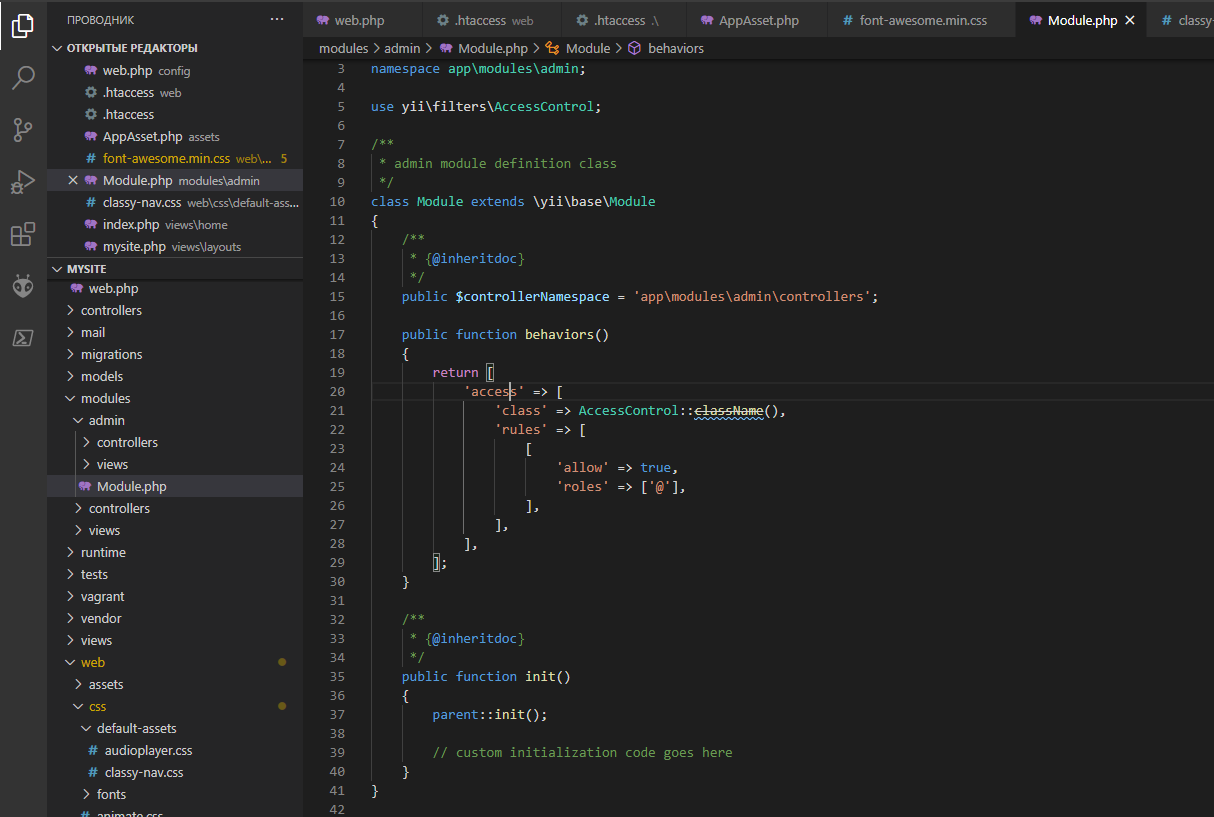
],

],

],

];

}



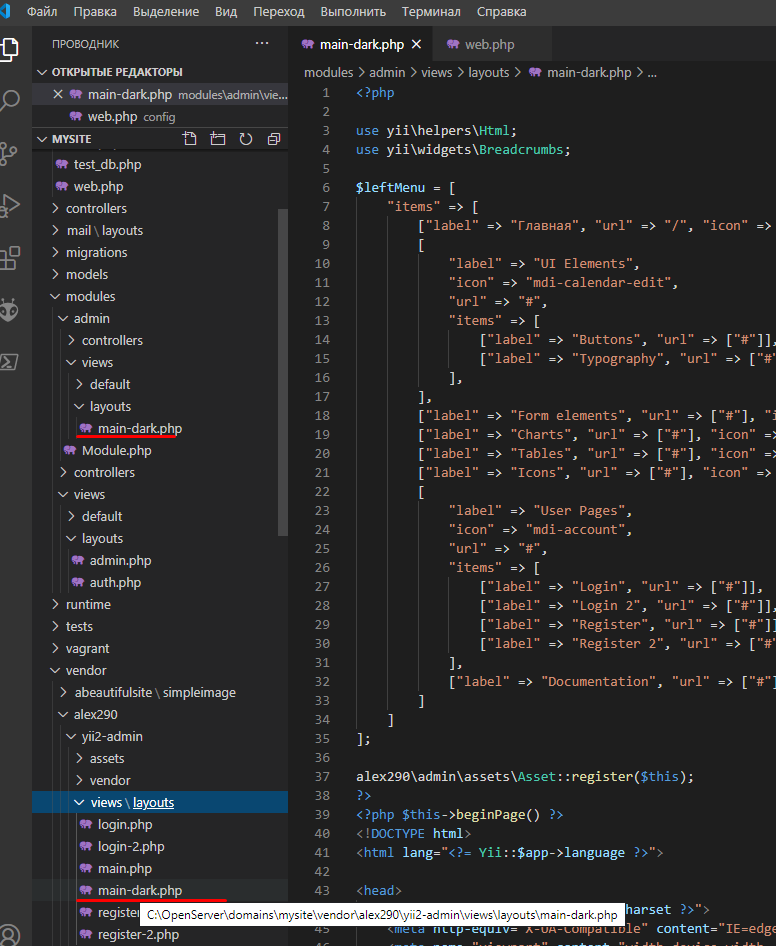
* Создаем админку, используя папин шаблон. Для начала заходим сюда: <https://github.com/alex290/yii2-admin>

и выполняем следующие действия:

1. Заходим в компостер и пишем:

php composer.phar require --prefer-dist alex290/yii2-admin

1. Видим, что загрузилось все благополучно в папку vendor и перетаскиваем main.php or main-dark.php шаблон в свою папку с модулем админки в подпапку вид



1. Дальше мы подсоединяем шаблон админки и класс, а значит пишем в config/web.php следующий код:

'modules' => [

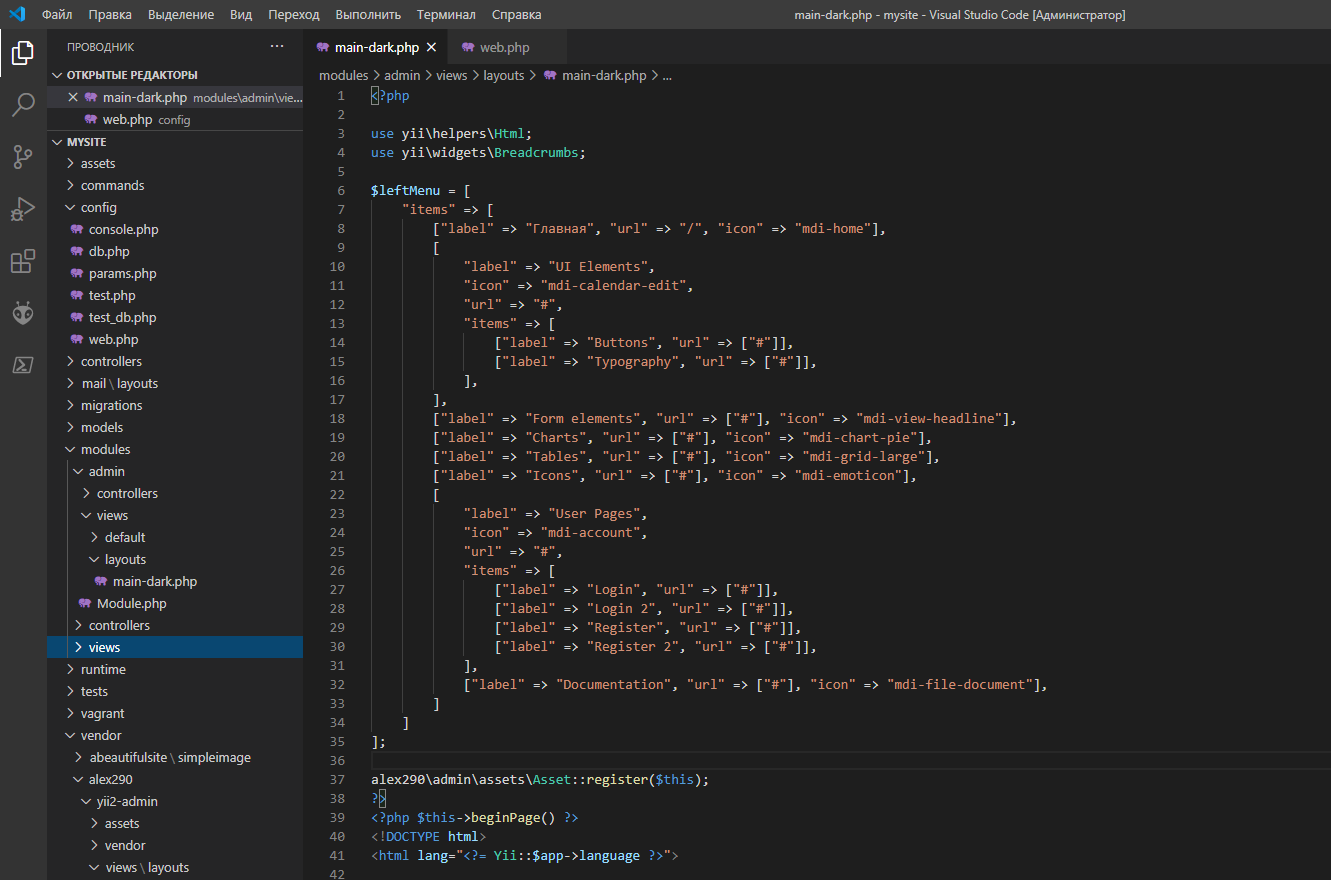
'admin' => [

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

'layout' => 'main-dark',

],

1. Дальше мы перетаскиваем левое меню вверх в main(…).php и для этого создадим переменную $leftMenu, а саму переменную перенесем вверх



      <div class="container-fluid page-body-wrapper">

            <!-- partial:partials/\_sidebar.html -->

            <nav class="sidebar dark sidebar-offcanvas" id="sidebar">

                <!--Боковое меню-->

                <?= alex290\admin\widgets\Menu::widget($leftMenu);

                ?>

            </nav>

            <!-- partial -->

            <div class="main-panel">

$leftMenu = [

    "items" => [

        ["label" => "Главная", "url" => "/", "icon" => "mdi-home"],

        [

            "label" => "UI Elements",

            "icon" => "mdi-calendar-edit",

            "url" => "#",

            "items" => [

                ["label" => "Buttons", "url" => ["#"]],

                ["label" => "Typography", "url" => ["#"]],

            ],

        ],

        ["label" => "Form elements", "url" => ["#"], "icon" => "mdi-view-headline"],

        ["label" => "Charts", "url" => ["#"], "icon" => "mdi-chart-pie"],

        ["label" => "Tables", "url" => ["#"], "icon" => "mdi-grid-large"],

        ["label" => "Icons", "url" => ["#"], "icon" => "mdi-emoticon"],

        [

            "label" => "User Pages",

            "icon" => "mdi-account",

            "url" => "#",

            "items" => [

                ["label" => "Login", "url" => ["#"]],

                ["label" => "Login 2", "url" => ["#"]],

                ["label" => "Register", "url" => ["#"]],

                ["label" => "Register 2", "url" => ["#"]],

            ],

            ["label" => "Documentation", "url" => ["#"], "icon" => "mdi-file-document"],

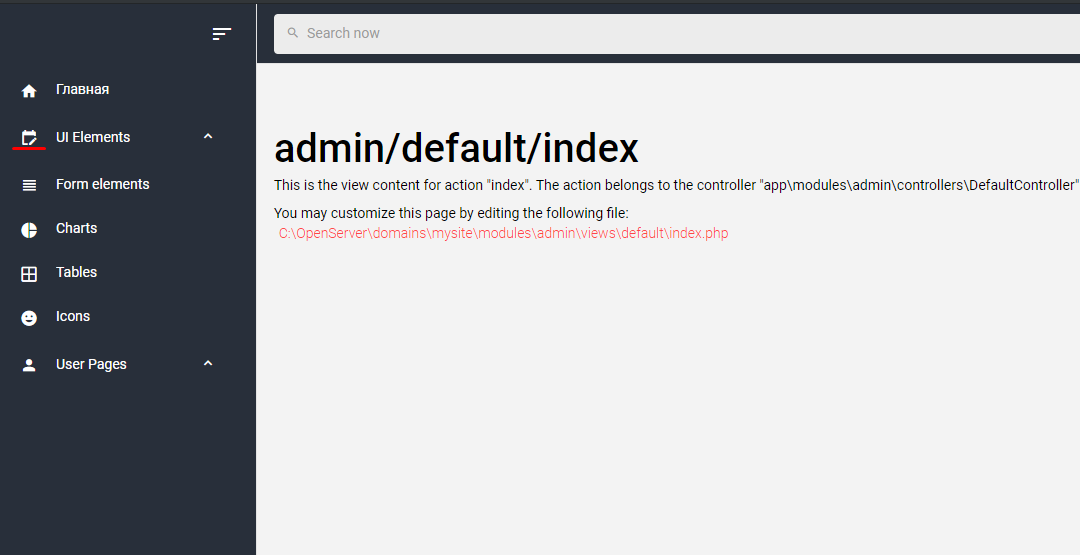
        ]

    ]

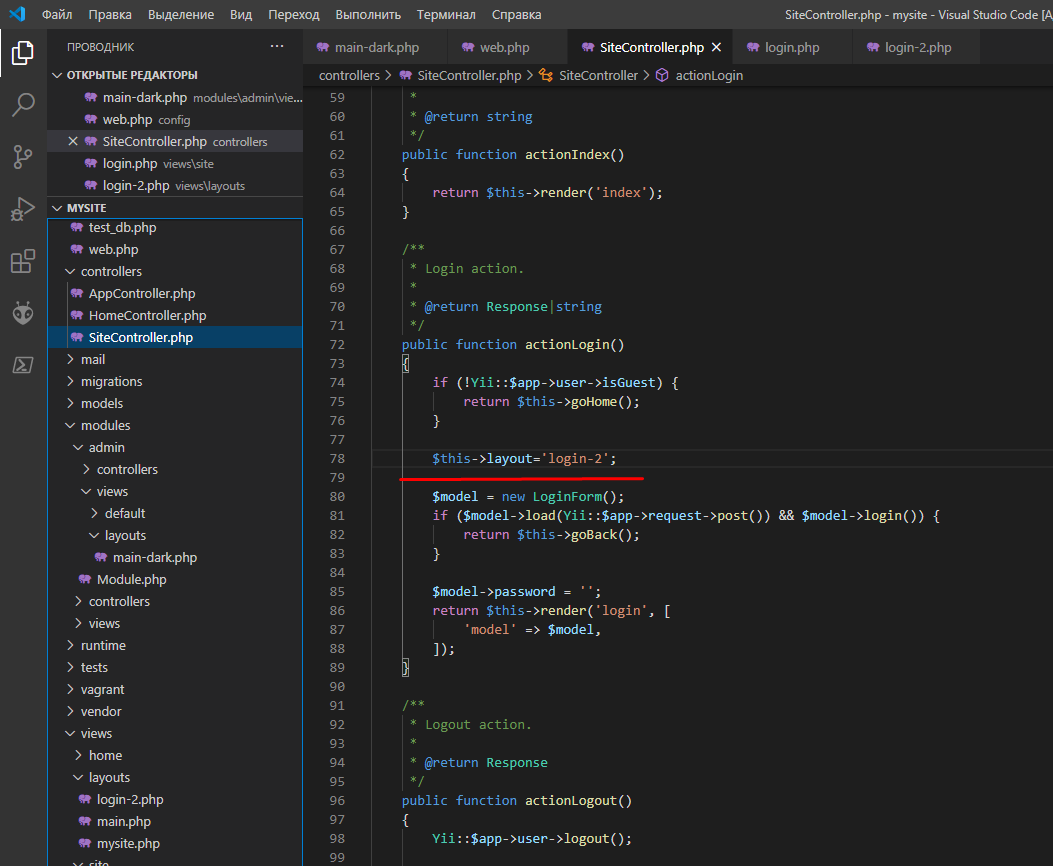
];

1. Дальше нам понадобиться сайт material design icons, где ищем иконки, которые будем подключать к классу mdi-... . Фактически мы преображаем нашу админку уже готовыми fonts-вскими шрифтами

Результат уже видим на картинке!



* Дальше мы подключим шаблон к логину. Выберем к примеру папин шаблон. Мы перенесем его в view/layouts



Дальше зайдем в controllers/SiteController находим:

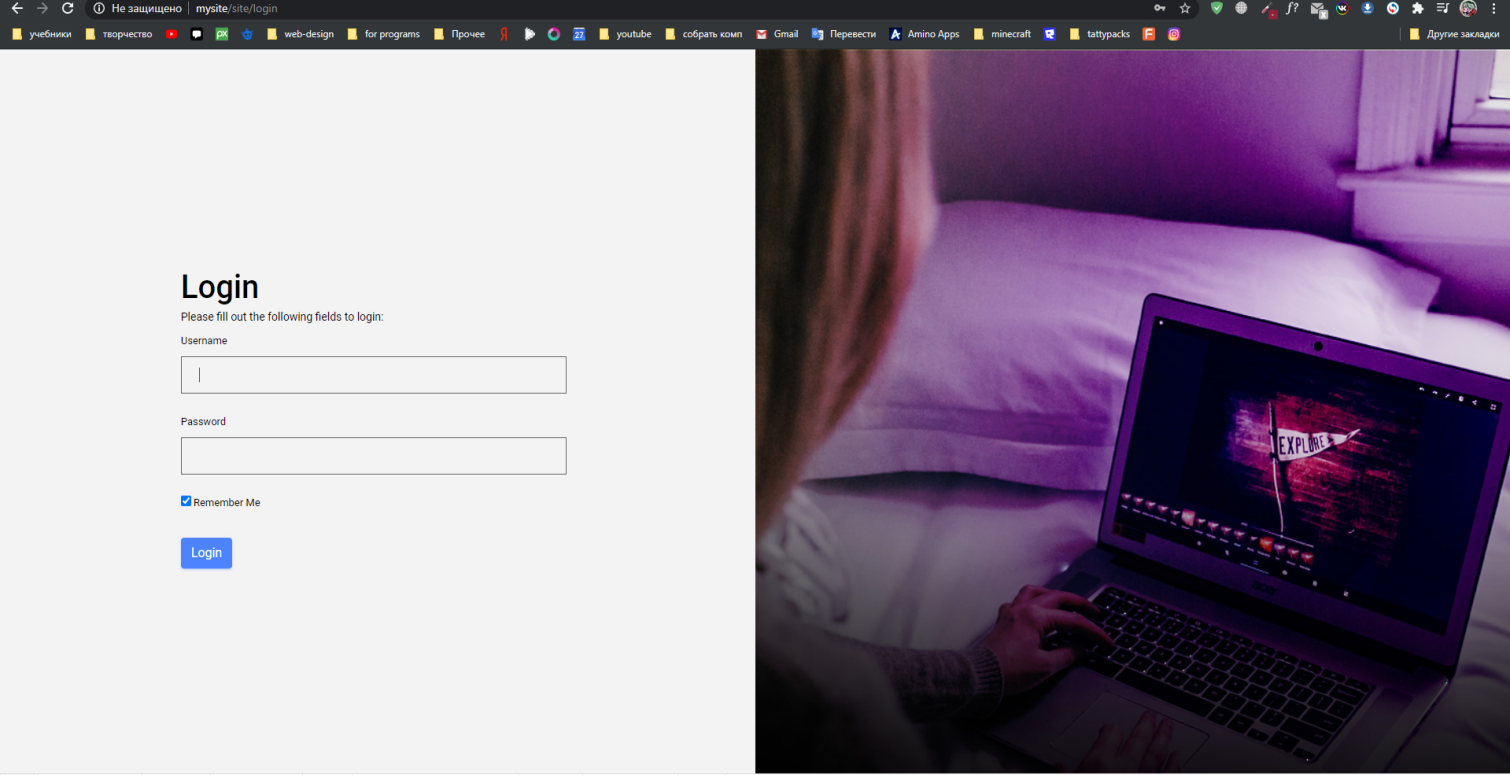
public function actionLogin()

И подключаем наш layout или же внешний вид странички входа в админку

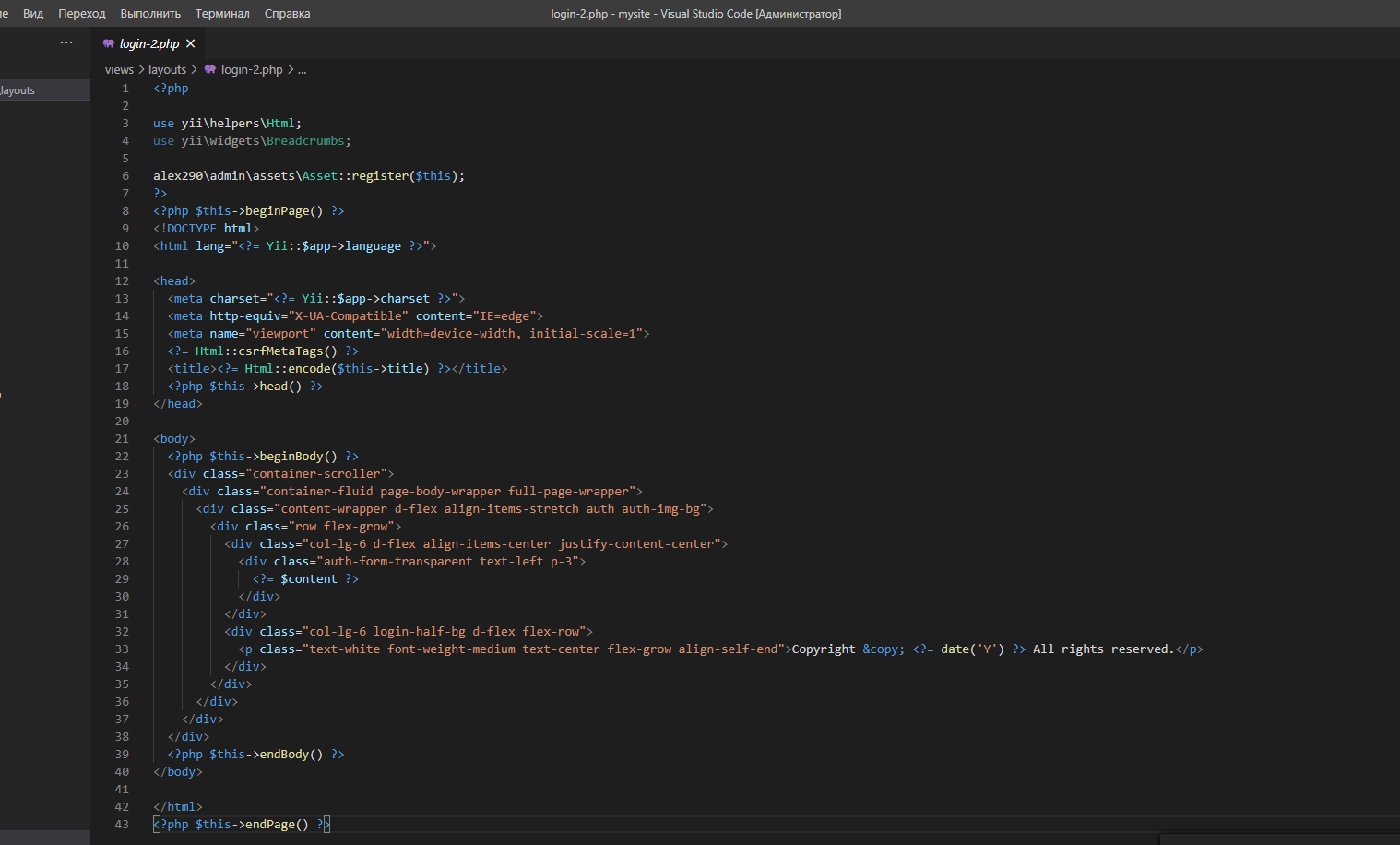
И пишем как на прошлой картинке после первого if следующее:

$this->layout='login-2';

В итоге получаем следующее:



* Т.к. мы использовали уже готовую админку, подгоняем layout под тот вид, который мы видим ниже на картинке:



* В modules/admin/controllers мы создадим 2 контроллера, 1-й AppAdminController.php, а 2-й MainController.php

В AppAdminController.php мы напишем:

<?php

namespace app\modules\admin\controllers;

use yii\web\Controller;

class AppAdminController extends Controller

{

}

?>

А в MainController.php напишем:

<?php

namespace app\modules\admin\controllers;

class MainController extends AppAdminController

{

public function actionIndex()

{

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

}

public function actionTest()

{

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

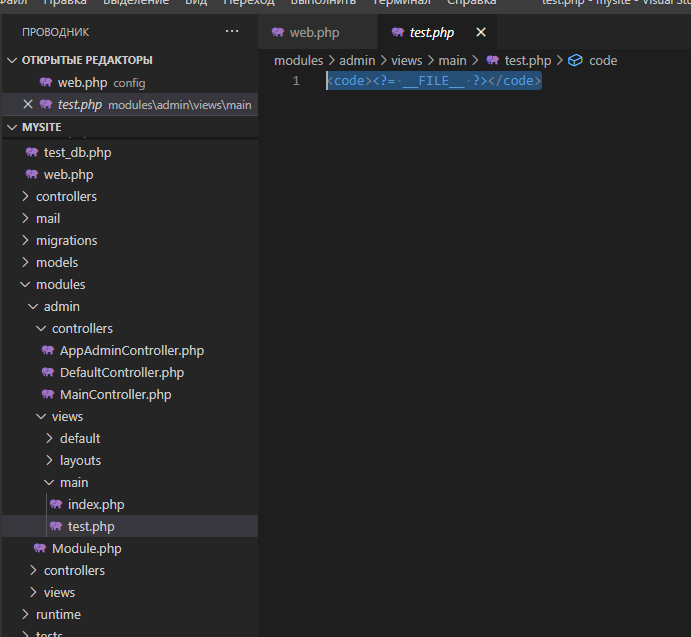
}

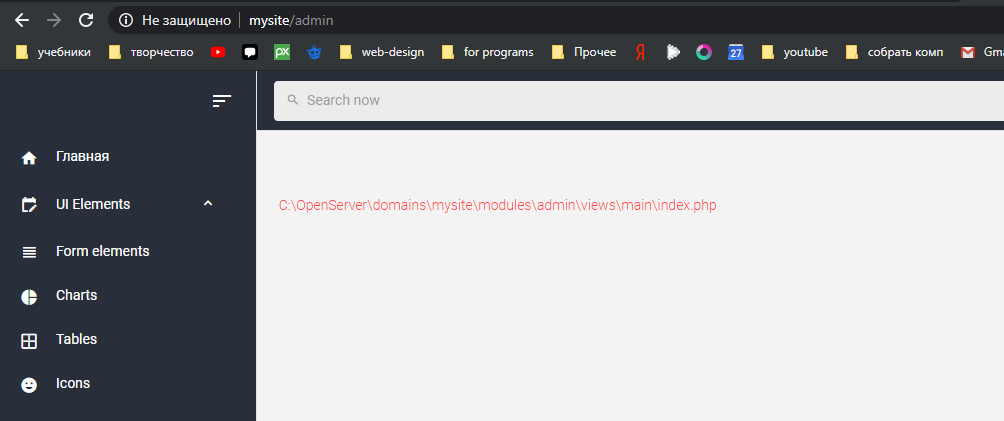
}

?>

P.S. Актион тест мы создали для проверки

* В modules/admin/views создадим папку main, а в папке main создадим 2 файла - index.php и test.php . Для проверки напишем пока туда код: <code><?= \_\_FILE\_\_ ?></code>





7 ЧАСТЬ - ЗАПОЛНЯЕМ АДМИНКУ

* Находим место в админке, где должен быть логотип и туда вставляем следующий код:

href="<?= Yii::$app->homeUrl ?>"

Это чтобы можно было попасть на страницу

* Дальше создадим возможность видеть записи. Для начала заходим ваш\_домен/gii и переходим в раздел GRUD
* При добавлении виджета для Категорий мы пишем обязательно, вместо старого виджета <https://github.com/alex290/yii2-treemanager> следующее:

<?php if ($arrModel->count() > 0) : ?>

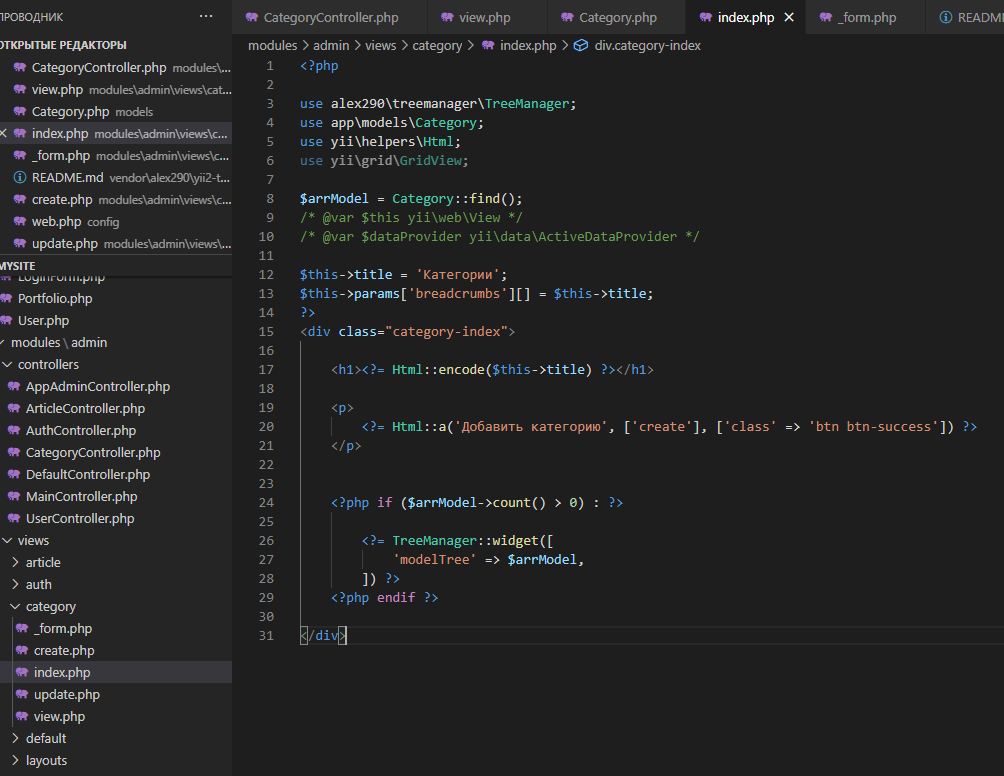
<?= TreeManager::widget([

'modelTree' => $arrModel,

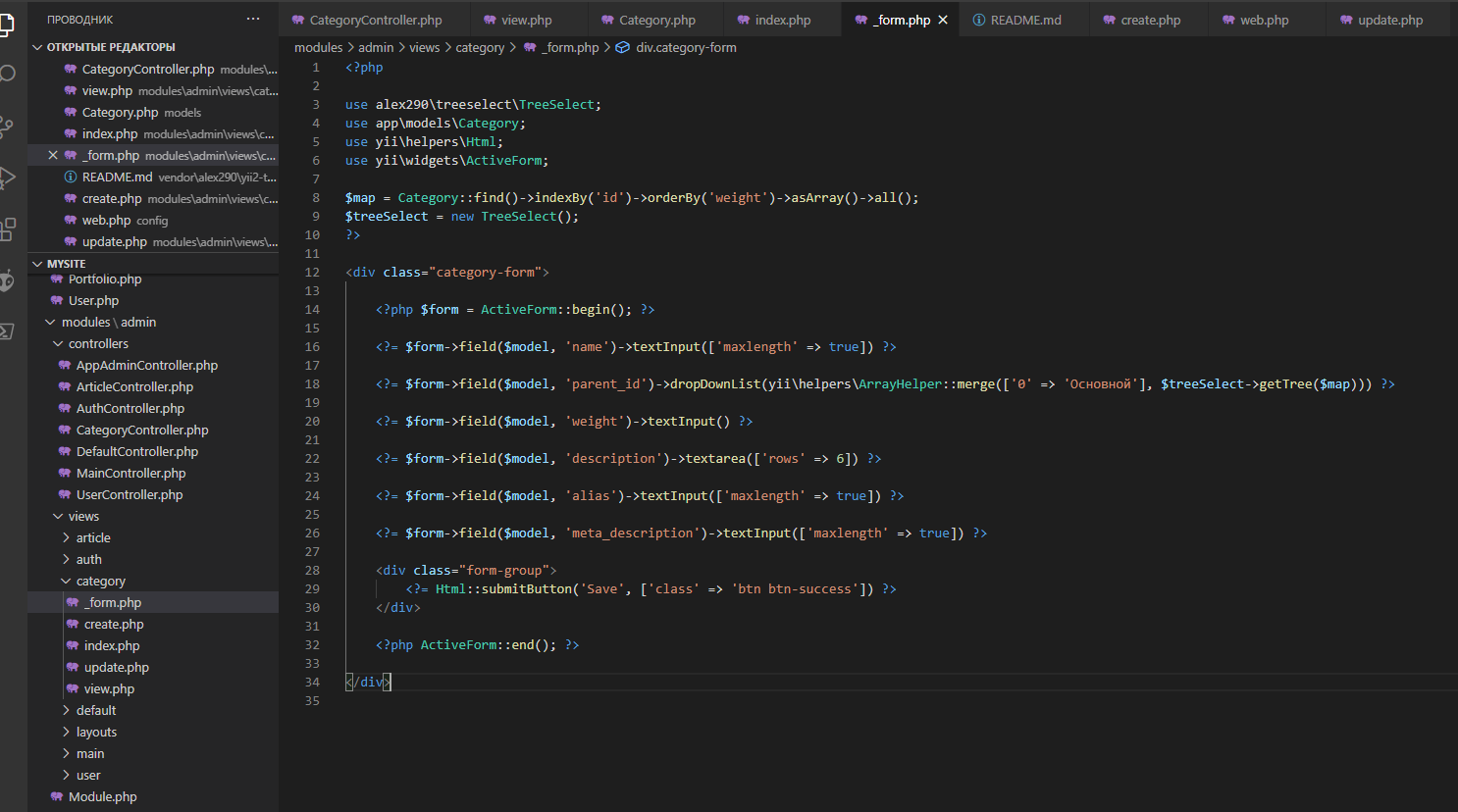
]) ?>

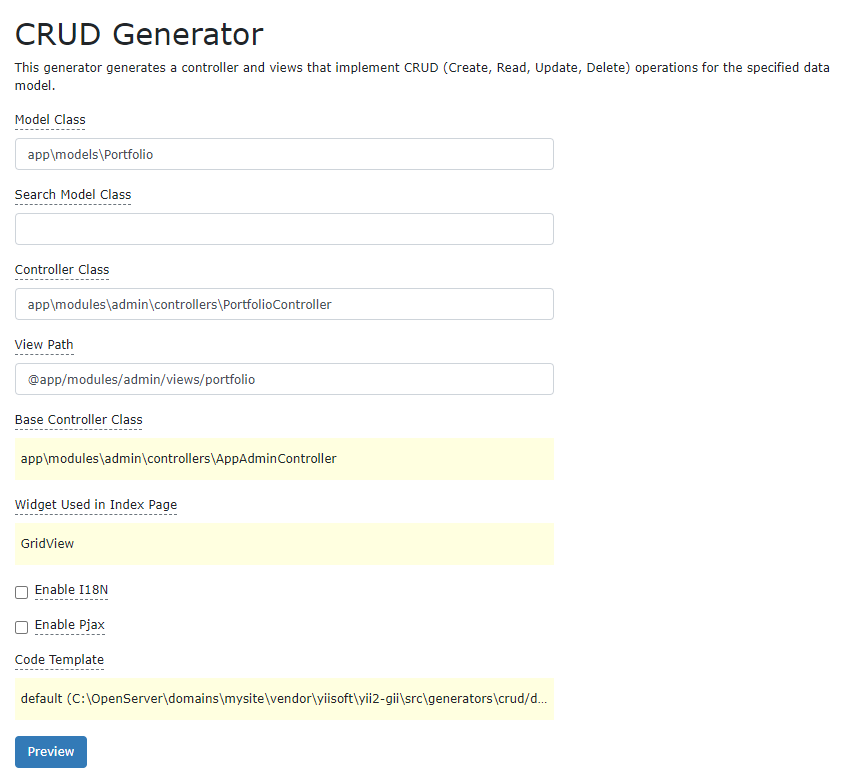
<?php endif ?>

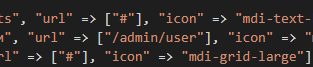
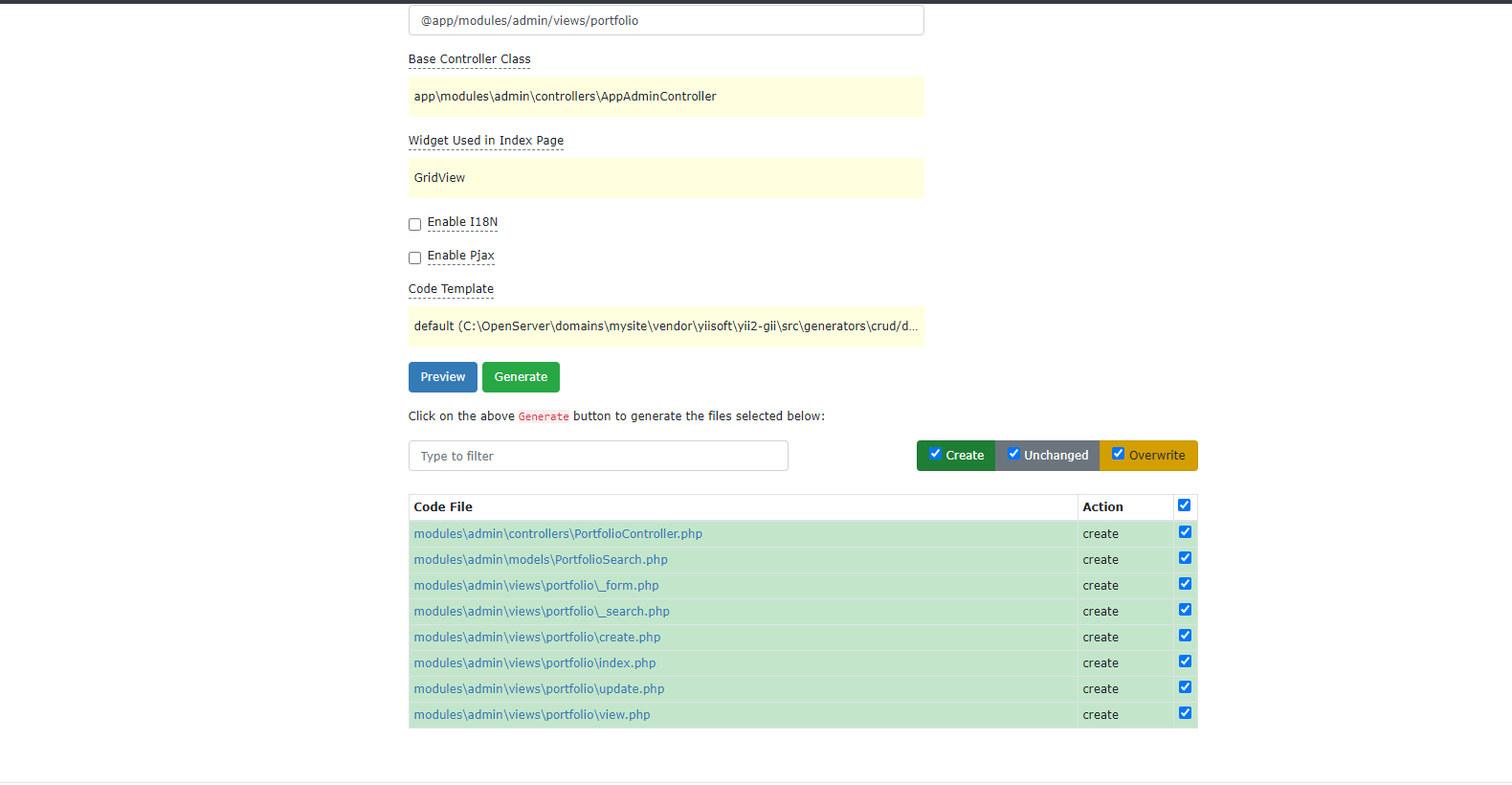
Пример:



* <https://github.com/alex290/yii2-treeselect>



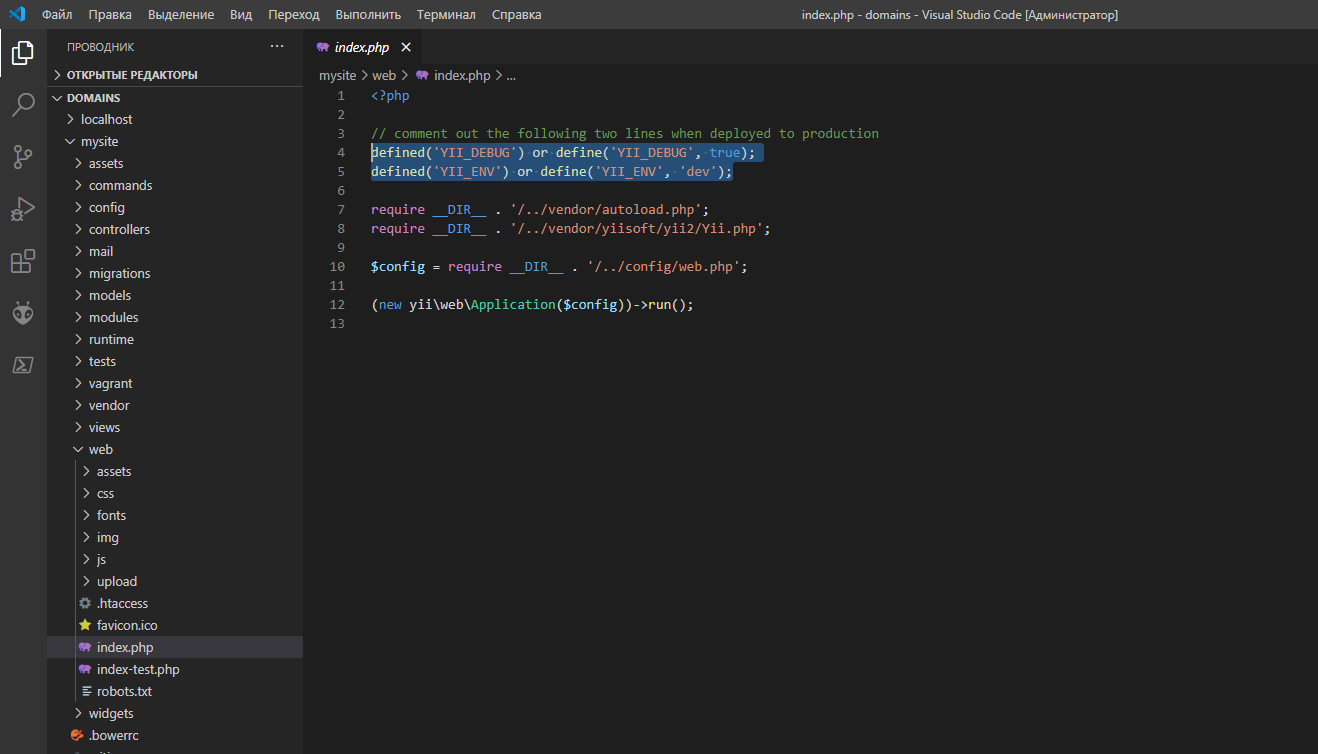




9 ЧАСТЬ - ПЕРЕНОС САЙТА НА ХОСТИНГ

* Надо отключить внизу панель Yii2, для этого мы переходим в web/index.php

Мы закомментируем первые 2 строки (после <?php), которые будут выделены ниже на скриншоте

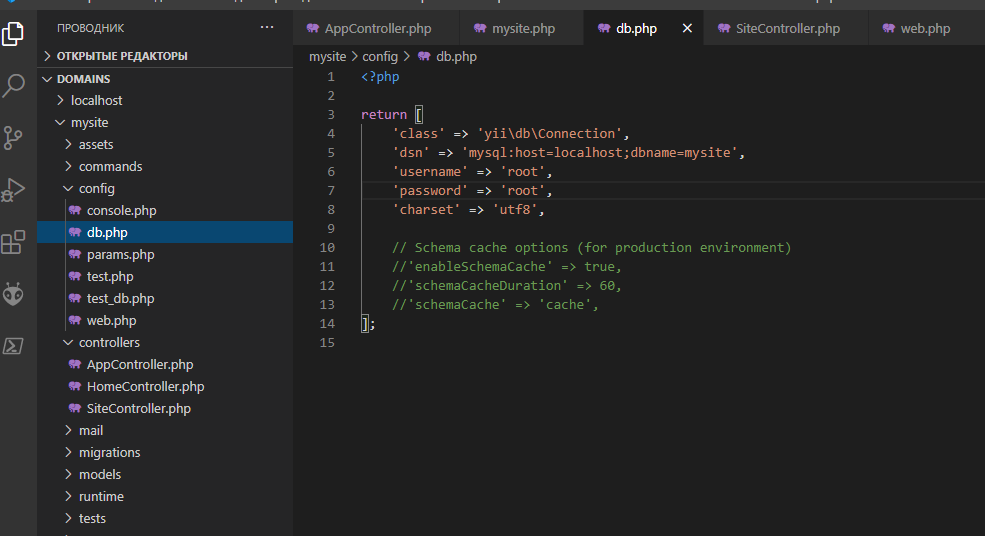


10 ЧАСТЬ - ДОПОЛНЕНИЕ

1. Это gii, чтобы фраемворк создавал необходимые вещи автоматически

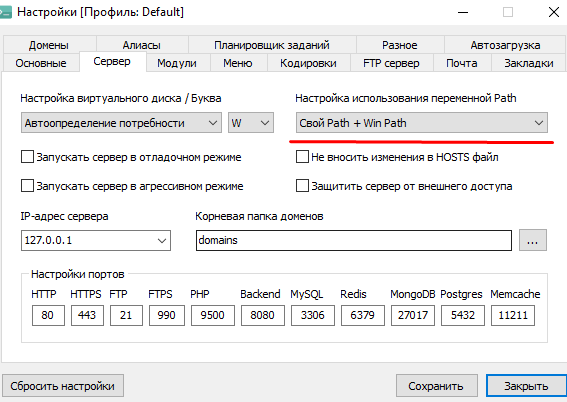
[http://ваш\_домен/](http://xn--_-7sbfkf5bif1g/site/start)gii

Пример: <http://sportdelivery/gii>



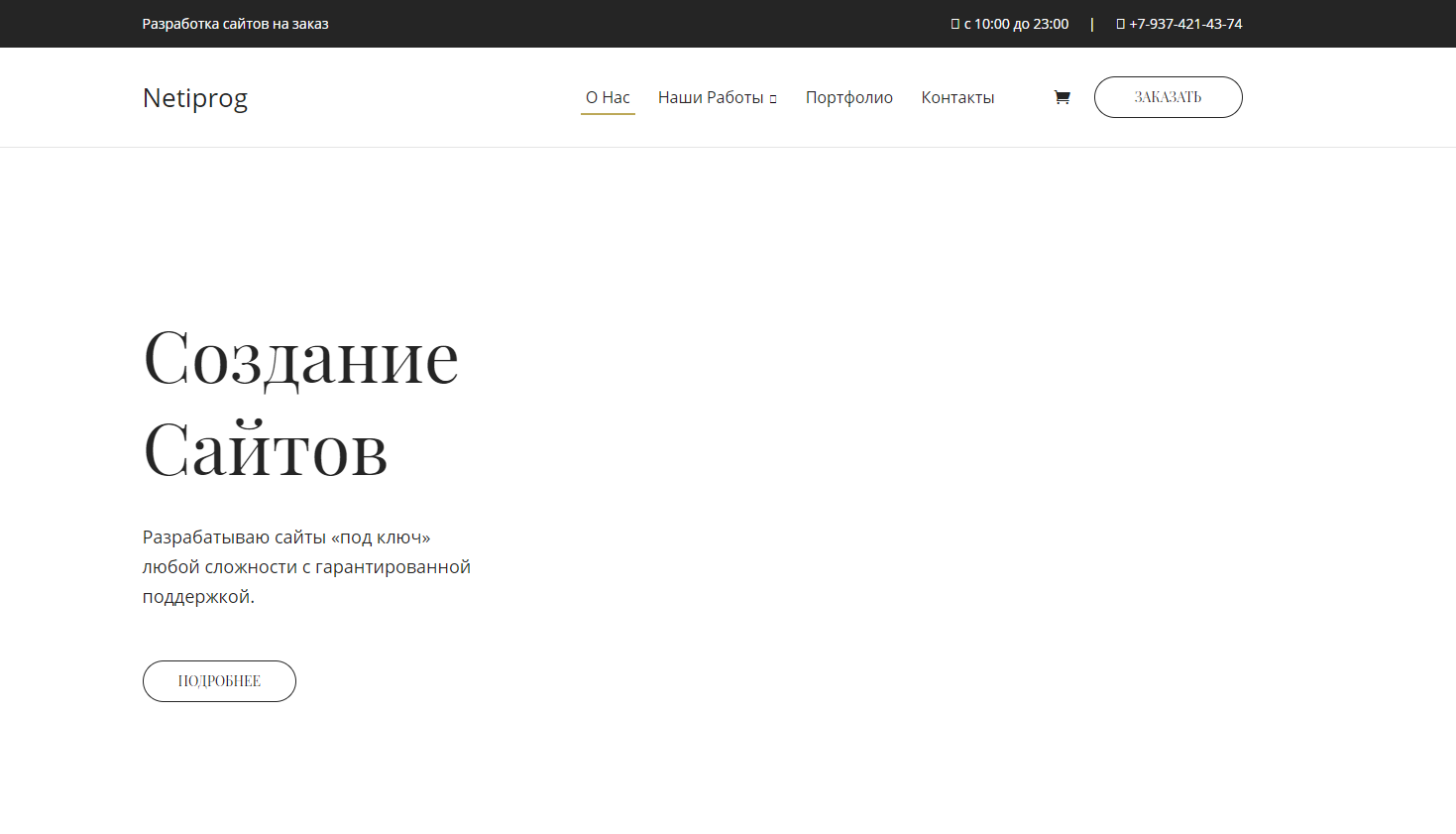
В db.php в config мы подключаем саму бд с которой мы будем работать

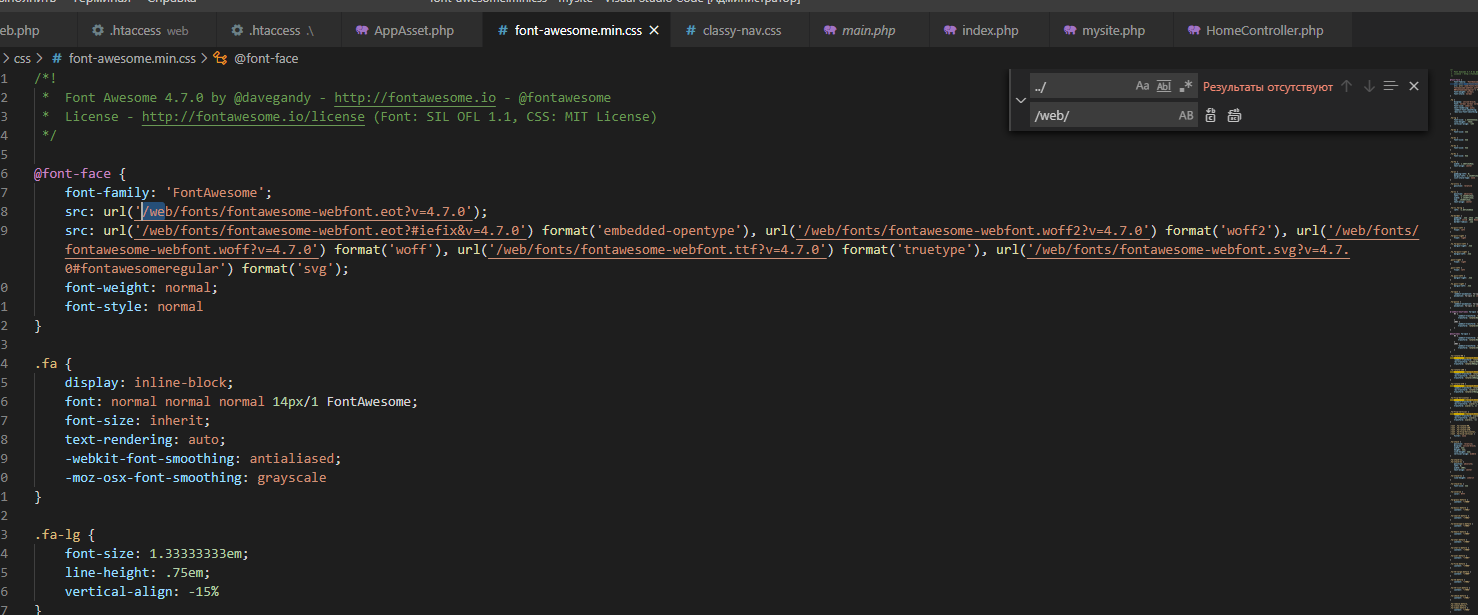
В сайт



Win path отвечает за git команды

* Если у вас проблемы с иконками и подключением и у вас такая проблема





Мы показываем что не из корня подключение, а из папки /web/, это проблема с httacces

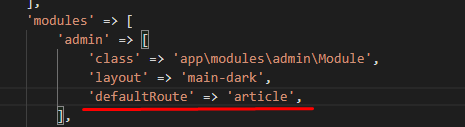
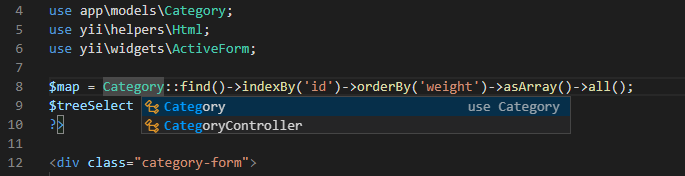
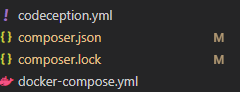
* Если мы хотим скачать файлы с github и видим примерно такую строчку:

php composer.phar require --prefer-dist alex290/yii2-admin "\*"

То мы, если скачиваем через OpenServer, используя composer, не пишем php и .phar

composer require --prefer-dist alex290/yii2-admin "\*"

Это связано с тем, что некоторые композеры работают напрямую, как наш, без этой команды, а некоторые через .phar

* В config/web.php если хотим указать в админке при запуске чтобы открывалась таблица БД, например со статьями, то просто прописываем название вашей БД как указано внизу 
* При добавлении use путей, мы нажимаем cntrl + пробел и нажимаем enter 
* Если стоит в компе старая версия композерато, пишем эти команды в композер: composer self-update и composer self-update --2
* 

Если вы работаете в git, оключаем желтое подсвечивание 