Содержание

[Установка yii2 2](#_Toc127113375)

[Создание БД 4](#_Toc127113376)

[Генерация моделей и CRUD контроллеров 5](#_Toc127113377)

[Регистрация пользователей 6](#_Toc127113378)

[Разделение прав доступа 7](#_Toc127113379)

[Вывод решенных проблем на главную 10](#_Toc127113380)

[Анимация при наведении на фото 11](#_Toc127113381)

[Добавление шрифта 12](#_Toc127113382)

[AJAX счётчик решенных проблем со звуком 12](#_Toc127113383)

[Подключение PJAX 13](#_Toc127113384)

[Добавление в корзину 13](#_Toc127113385)

[Отображение картинок для админа в index 15](#_Toc127113386)

[Отображение категории в товаре выпадающим списком 15](#_Toc127113387)

[Отображение товара с фото и категорией 16](#_Toc127113388)

[Настройка Visual Studio Code 16](#_Toc127113389)

# Установка yii2

* скачиваем зип с yii
* открываем файлы сервера и пихаем “внутренности” зипа по пути:
* **Home/web/{domen}/public\_html**
* заходим в **Home/web/{domen}/public\_html/web**и переносим файлы index.php и .htaccess в **Home/web/{domen}/public\_html**
* редактируем файл index.php (удаляем типа корень(точечки)

require \_\_DIR\_\_ . **'/vendor/autoload.php'**;

require \_\_DIR\_\_ . **'/vendor/yiisoft/yii2/Yii.php'**;

$config = require \_\_DIR\_\_ . **'/config/web.php'**;

* заходим в **Home/web/{domen}/public\_html/config** и в **web.php** раскомментируем ‘urlManager’

'urlManager' => [

            'enablePrettyUrl' => true,

            'showScriptName' => false,

            'rules' => [

            ],

        ],

* далее заходим **Home/web/{domen}/public\_html/config**

здесь заходим в db.php и меняем строчки

   'dsn' => 'mysql:host=localhost;dbname=**Имя БД**',

    'username' => '**имя аккаунта БД**',

    'password' => '**пароль аккаунта БД**',

1. Устанавливаем ssl сертификат (при необходимости)
2. Установите русский язык и название приложения в файле **config/web.php**

$config = [

    'id' => 'basic',

**'name'=>'Мир цветов',**

**'language'=>'ru-RU',**

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

    'bootstrap' => ['log'],

    'aliases' => [

        '@bower' => '@vendor/bower-asset',

        '@npm'   => '@vendor/npm-asset',

    ],

1. здесь же, в **config/web.php,** меняем в самом низу 'allowedIPs' (добавляем звездочку после запятой, это означает что допускаются все айпи адреса)

**'allowedIPs' => ['127.0.0.1', '::1','\*' ],**

1. в **config/web.php**

в 'cookieValidationKey' добавляем свое любое слово.

'components' => [

        'request' => [

            'cookieValidationKey' => **'meow'**,

        ],

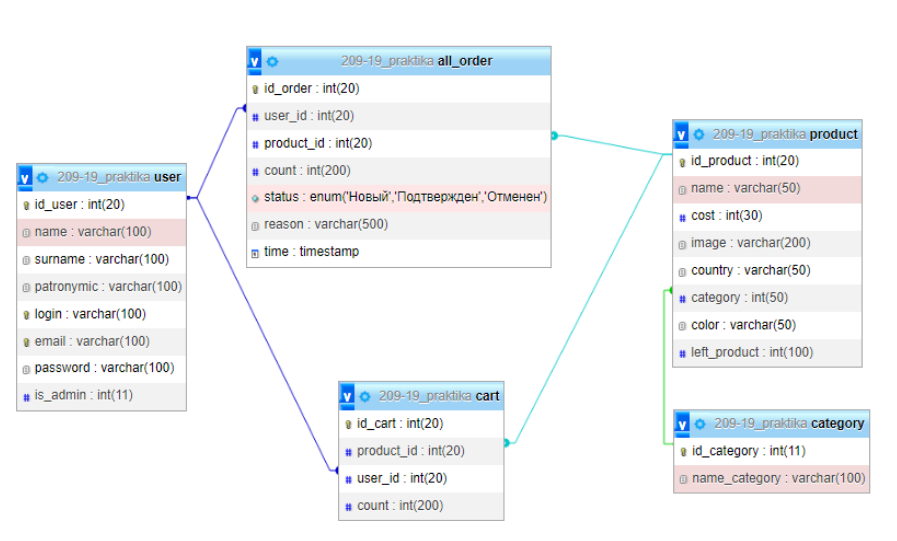
1. переходим по ссылке проверяя работоспособность

[**https://{domen}/gii**](about:blank)

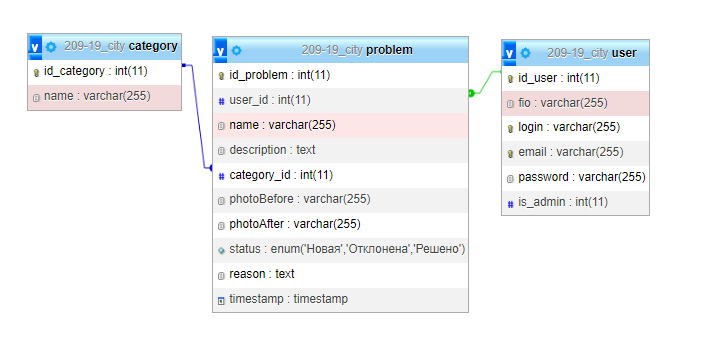
1. **в папке public\_html/web** берем css файл и переносим в папку выше (в сам **public\_html**)
2. **в public\_html** удаляем index.html

# Создание БД

База данных магазина



База данных решения проблем



# Генерация моделей и CRUD контроллеров

* Переходим на [**https://{domen}/gii**](about:blank)
* **Model Generator** создаем на все таблицы модельки
* **USER** перезаписываем, но старый сохраняем в блокнотик
* В самом верху добавляем **implements IdentityInterface**

class User extends \yii\db\ActiveRecord **implements IdentityInterface**

* findIdentity и findByUsername(Login) немного меняем

 public static function findIdentity($id\_user)

    {

**return self::findOne($id\_user);**

    }

        /\*\*

     \* Finds user by login

     \*

     \* @param string $login

     \* @return static|null

     \*/

    public static function findByUsername($username)

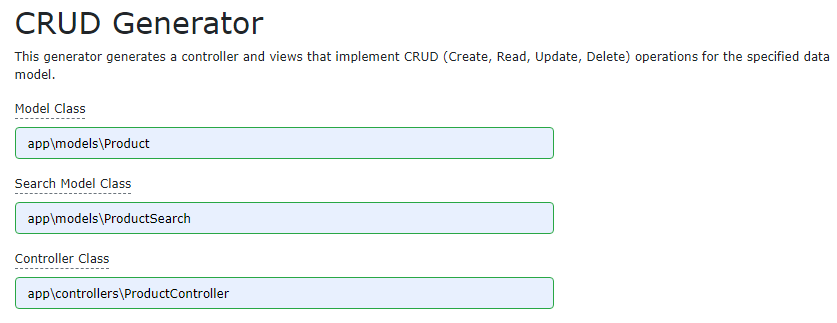
    {

**return self::find()->where(['login' => $username])->one();**

    }

* findIdentityByAccessToken, getAuthKey и validateAuthKey удаляем содержимое, оно нам не надо
* getId и validatePassword оставляем как было
* Настройка CRUD

Делаем со всеми нужными таблицами где нужны страницы (Create, Read, Update, Delete)



# Регистрация пользователей

* Копируем **модель юзера** и переименовываем **в RegForm**, не забываем **переименовать** в документе **класс User на RegForm**, пишем, что он расширяет,наследует класс юзер (**extends User**)
* **tableName удаляем**, оно и так наследуется от юзера
* Оставляем rules и attributeLabels (В attributeLabels дописываем новые поля (повторение пароля и согласие на обработку)

 public function attributeLabels()

    {

        return [

            'name' => 'Имя',

            'surname' => 'Фамилия',

            'patronymic' => 'Отчество',

            'login' => 'Логин',

            'email' => 'Email',

            'password' => 'Пароль',

            'password\_repeat' => 'Повторите пароль',

            'agree' => 'Согласие на обработку данных',

        ];

    }

* Добавляем две публичные переменные (public $password\_repeat и public $agree)
* В UserController меняем actionCreate (модель = нью не юзер, а рег форм)

 $model = **new RegForm**();

* Добавляем новые поля в форму (views/user/\_form)

<?= $form->field($model, 'password\_confirm')->passwordInput(['maxlength' => true]) ?>

<?= $form->field($model, 'agree')->**checkbox**() ?>

* модель RegForm rules
* тут добавляем паттерны и сообщения об ошибках

public function rules()

    {

        return [

            [['fio', 'login', 'email', 'password', 'password\_confirm', 'agree'], 'required'],

            ['fio', **'match'**, **'pattern' => '/^[А-Яа-я\s\-]{8,}$/u', 'message' => 'Допускаются только кирилица, пробелы и дефис'**],

            ['login', 'match', 'pattern' => '/^[A-Za-z]{2,}$/u', 'message' => 'Допускаются только латинские буквы'],

            ['login', 'unique', 'message' => 'Такой логин уже используется'],

            ['email', 'email', 'message' => 'Некорректный email'],

            [**'password\_confirm', 'compare', 'compareAttribute' => 'password', 'message' => 'Пароли должны совпадать'**],

            [**'agree', 'boolean'**],

            [**'agree', 'compare', 'compareValue' => true, 'message' => ''**],

            [['admin'], 'integer'],

            [['fio', 'login', 'email', 'password'], 'string', 'max' => 255],

        ];

    }

* Криейт юзер из контроллера (берем часть из LoginForm (function login)

 public function actionCreate()

    {

        $model = new RegForm();

        if ($model->load($this->request->post()) && $model->save()) {

**Yii::$app->user->login($model);**

**return $this->redirect(['/lk']);**

        }

        return $this->render('create', [

            'model' => $model,

        ]);

    }

# Разделение прав доступа

* Создаем админ контроллер (через ControllerGenerator)
* Cоздаем метод бефор эктион  берем кусочек из зила

public function beforeAction($action)

    {

    // your custom code here, if you want the code to run before action filters,

    // which are triggered on the [[EVENT\_BEFORE\_ACTION]] event, e.g. PageCache or AccessControl

    if (Yii::$app->user->isGuest || Yii::$app->user->identity->admin !=1) {

       $this->redirect(['/site/login']);

       return false;

   }

   if (!parent::beforeAction($action)){

           return false;

    }

   // other custom code here

    return true; // or false to not run the action

}

* Изменяем Nav-виджет

 echo Nav::widget([

        'options' => ['class' => 'navbar-nav'],

**'items'=>$items,**

 ]);

NavBar::end();

* Добавляем сверху (после NavBar::begin ([ ]); )

$items=[];

    if(Yii::$app->user->isGuest ){

        $items[] = ['label' => 'Каталог', 'url' => ['/product/catalog?sort=-id']];

        $items[] = ['label' => 'О нас', 'url' => ['/site/about']];

        $items[] = ['label' => 'Где нас найти', 'url' => ['/site/contact']];

        $items[] = ['label' => 'Регистрация', 'url' => ['/user/create'], 'class'=>'bg-dark'];

        $items[] = ['label' => 'Авторизация', 'url' => ['/site/login']];

    }

    else{

        if(Yii::$app->user->identity->is\_admin==1){

        $items[] = ['label' => 'Административная панель', 'url' => ['/admin']];

        }

        else{

            $items[] = ['label' => 'Каталог', 'url' => ['/product/catalog?sort=-id']];

            $items[] = ['label' => 'О нас', 'url' => ['/site/about']];

            $items[] = ['label' => 'Где нас найти', 'url' => ['/site/contact']];

            $items[] = ['label' => 'Корзина', 'url' => ['/cart/index'],'class'=>'bg-light'];

        }

        $items[] = '<li class="nav-item">'

            . Html::beginForm(['/site/logout'])

            . Html::submitButton(

                'Выйти (' . Yii::$app->user->identity->login . ')',

                ['class' => 'nav-link btn btn-link logout']

            )

            . Html::endForm()

            . '</li>';

    }

# Вывод решенных проблем на главную

* SiteController

public function actionIndex()

    {

        $problems = Problem::find()->where(['status' => 'Решено'])->orderBy(['timestamp' => SORT\_DESC])->limit(4)->all();

        return $this->render('index',[

            'problems'=>$problems,

        ]);

    }

* site/index.php

<div class="body-content">

<div class="main-container">

<div class="container ">

<div class="card-group row  justify-content-center g-4 ">

<?php

foreach ($problems as $problem){

echo '

<div class=" text-center col-lg-5 " >

<div class="overflow-hidden card mb-2" style="min-height:29rem; max-height:29rem;">

<div class="card-body overflow-auto" >

<h5 class="card-title">'.$problem->name.'</h5>

<p class="card-text">'.$problem->description.'</p>

</div>

<img src="web/uploads/'.$problem->photoAfter.'" class="overflow-hidden card-img-bottom scale" style="min-height:20rem;" alt="..."

data-before="web/uploads/'.$problem->photoBefore.'"

data-after="web/uploads/'.$problem->photoAfter.'"

onMouseOver="over(this)"

onMouseOut="out(this)">

<p class="card-text text-muted">'.$problem->timestamp.'</p>

</div>

</div>';

}

?>

</div>

</div>

</div>

</div>

* **Важное замечание:** если необходимо сделать фильтрацию товаров, тогда SiteController не трогаем. Тогда в url будут работать get-запросы ('/product/catalog?sort=price')
* В product/catalog.php

$products=**$dataProvider->getModels();**

# Фильтрация по категориям

* сatalog.php

<div class='btn-group'>

  <button class='btn btn-success dropdown-toggle m-1' type='button' data-bs-toggle='dropdown' data-bs-auto-close='true' aria-expanded='false'>

   По цене

  </button>

  <ul class='dropdown-menu '>

    <li><a href='/product/catalog?sort=price' class='dropdown-item' >По возрастанию</a></li>

    <li><a href='/product/catalog?sort=-price' class='dropdown-item' >По убыванию</a></li>

  </ul>

</div>

<div class='btn-group'>

  <button class='btn btn-secondary dropdown-toggle m-1' type='button' data-bs-toggle='dropdown' data-bs-auto-close='true' aria-expanded='false'>

   По наименованию

  </button>

  <ul class='dropdown-menu '>

    <li><a href='/product/catalog?sort=-name' class='dropdown-item' href='#'>По возрастанию (Я-А)</a></li>

    <li><a href='/product/catalog?sort=name' class='dropdown-item' href='#'>По убыванию (А-Я)</a></li>

  </ul>

</div>

<div class='btn-group'>

  <button class='btn btn-secondary dropdown-toggle m-1' type='button' data-bs-toggle='dropdown' data-bs-auto-close='true' aria-expanded='false'>

   Страна поставщика

  </button>

  <ul class='dropdown-menu '>

    <li><a href='/product/catalog?sort=-country' class='dropdown-item' href='#'>По возрастанию (Я-А)</a></li>

    <li><a href='/product/catalog?sort=country' class='dropdown-item' href='#'>По убыванию (А-Я)</a></li>

  </ul>

</div>

<div class='btn-group'>

  <button class='btn btn-secondary dropdown-toggle m-1' type='button' data-bs-toggle='dropdown' data-bs-auto-close='true' aria-expanded='false'>

   По категории

  </button>

  <ul class='dropdown-menu '>

    <li><a href='/product/catalog?ProductSearch[category]=1' class='dropdown-item' href='#'>цветы</a></li>

    <li><a href='/product/catalog?ProductSearch[category]=2' class='dropdown-item' href='#'>упаковка</a></li>

    <li><a href='/product/catalog?ProductSearch[category]=3' class='dropdown-item' href='#'>дополнительно</a></li>

  </ul>

</div>

* Примеры фильтрации

|  |  |
| --- | --- |
| **URL адрес** | **Выполняемое действие** |
| https://<host>/product/index?sort=price | Сортировка товаров по возрастанию цены |
| https://<host>/product/index?sort=-price | Сортировка товаров по убыванию цены |
| https://<host>/product/index? ProductSearh[category\_id]=5 | Поиск товаров у которых *category\_id=5* |
| https://<host>/product/index? ProductSearh[name]=питан | Нестрогий поиск по полю *name* |
| https://<host>/product/view?id=14 | Переход на страницу карточки товара с *id*=14 |

# Анимация при наведении на фото

* Добавляем data-моменты к картинке

<img src="web/uploads/'.$problem->photoAfter.'" class="overflow-hidden card-img-bottom scale" style="min-height:20rem;" alt="..."

data-before="web/uploads/'.$problem->photoBefore.'"

data-after="web/uploads/'.$problem->photoAfter.'"

onMouseOver="over(this)"

onMouseOut="out(this)">

* снизу добавляем js-код

<script>

function over(el){

        el.src=el.dataset.before;

    }

    function out(el){

        el.src=el.dataset.after;

    }

</script>

* Css-код

<style>

    .scale {

    display: inline-block;

    transition: 1s; /\* Время эффекта \*/

   }

   .scale:hover {

    transform: scale(1.05); /\* Увеличиваем масштаб \*/

    overflow: hidden;

   }

</style>

# Добавление шрифта

@font-face {

    font-family:'meow';

    src: url('/web/fonts/Nunito-VariableFont\_wght.ttf') format('truetype');

    font-weight: normal;

}

body {

    font-family: 'meow';

}

# AJAX счётчик решенных проблем со звуком

* В этом же скрипте добавляем js-код

var i= 0;

var b=1;

function counter(){

        $.ajax({

        method:'GET',

        url:'<?=Url::toRoute('/site/counter')?>',

        dataType: 'text',

        success: function(response){

            if(i != response){

                console.log(i +"!= "+ response+" тут звук");

                //звук

                var a = new Audio('web/mlem.mp3');

                a.play();

                i=response;

                console.log(i +"= "+ response+" переделка");

            }

            $("#counter").html('Решено проблем: '+response);

            if (b==1){

            document.getElementById("spin").remove();

            b=0;

            };

        }

    });

     }

    setInterval(counter,3000);

* добавляем id counter

<p id="counter" class="lead"> Счётчик обновляется</p>

# Подключение PJAX

Необходимо подключить модуль Pjax к валидируемому свойству email в файле **views/user/\_form.php**

use yii\widgets\Pjax;

редактировать поле email в этом же файле

field($model, 'email', ['enableAjaxValidation' => true])->textInput(['maxlength' => true],) ?>

# Добавление в корзину

* Js-код

function add\_product(id\_product, items){

    let form=new FormData();

    form.append('product\_id', id\_product);

    form.append('count', items);

    let request\_options={method: 'POST', body: form};

    fetch('https://pr-ohotnikova.xn--80ahdri7a.site/cart/create', request\_options)

        .then(response=>response.text())

        .then(result=>{

            console.log(result)

            let title=document.getElementById('staticBackdropLabel');

            let body=document.getElementById('modalBody');

            if (result=='false'){

                title.innerText='Ошибка';

                body.innerHTML="<p>Ошибка добавления товара, вероятно, товар уже раскупили</p>"

            } else {

                title.innerText='Информационное сообщение';

                body.innerHTML="<p>Товар успешно добавлен в корзину</p>"

            }

            let myModal = new

            bootstrap.Modal(document.getElementById("staticBackdrop"), {});

            myModal.show();

        })

}

* В main.php меняем footer, добавляя модальное окно

<footer id="footer" class="mt-auto py-3 bg-dark">

    <script src="/js/add\_product.js"></script>

    <div class="modal fade" id="staticBackdrop" data-bs-backdrop="static" data-bskeyboard="false" tabindex="-1" aria-labelledby="staticBackdropLabel"

         aria-hidden="true">

        <div class="modal-dialog">

            22

            <div class="modal-content">

                <div class="modal-header">

                    <h1 class="modal-title fs-5" id="staticBackdropLabel"></h1>

                    <button type="button" class="btn-close" data-bsdismiss="modal" aria-label="Close"></button>

                </div>

                <div class="modal-body" id="modalBody">

                </div>

                <div class="modal-footer">

                    <button type="button" class="btn btn-primary" data-bs-dismiss="modal">Закрыть</button>

                </div>

            </div>

        </div>

    </div>

    <div class="container ">

        <div class="row text-muted text-success">

            <div class="col-md-6 text-center text-white text-md-start">&copy; Магазин цветов <?= date('Y') ?></div>

        </div>

    </div>

</footer>

* Меняем CartController

 public function actionCreate()

    {

        $product\_id = Yii::$app->request->post('product\_id');

        $items=Yii::$app->request->post('count');

        $product = Product::findOne($product\_id);

        if (!$product) return false;

        if ($product->count > 0) {

            $product->count -= $items;

            $product->save(false);

            $model = cart::find()->where(['user\_id' => Yii::$app->user->identity->id\_user])->andWhere(['product\_id' => $product\_id])->one();

            if ($model) {

                $model->count += $items; //\*\*\*

     $model->save();

     return $product->count;

            }

            $model = new cart();

            $model->user\_id = Yii::$app->user->identity->id\_user;

            $model->product\_id = $product->id\_product;

            $model->count = $items; //\*\*\*

            if ($model->save(false)) return $product->count;

 }

        return 'false';

    }

# Оформление заказа

* Функция, перекидывающая товары из корзины в заказы (AllOrderController/actionCreate()):

/\*\*

\* Creates a new AllOrder model.

\* If creation is successful, the browser will be redirected to the 'view' page.

\* @return string|\yii\web\Response

\*/

public function actionCreate(){

$identity = Yii::$app->user->identity->id\_user;

$carts = Cart::find()->where(['user\_id'=> $identity])->all();

foreach ($carts as $cart){

$order = new AllOrder();

$order->user\_id = $identity;

$order->product\_id = $cart->product\_id;

$order->count = $cart->count;

$order->save(false);

$cart->delete();}

return $this->redirect(['all-order/index?AllOrderSearch[user\_id]='.Yii::$app->user->identity->id\_user]);}

# Отображение картинок для админа в index

<?= GridView::widget([

        'dataProvider' => $dataProvider,

        'filterModel' => $searchModel,

        'columns' => [

            ['class' => 'yii\grid\SerialColumn', 'options'=>['style'=>'width:20px']],

            ['attribute'=>'Фото', 'format'=>'html','options'=>['style'=>'width:100px'],

                'value'=>function($data){return"<img src='{$data->getProduct()->One()->image}' alt='photo' style='width:100px; '";}],

            ['attribute'=>'Товар','options'=>['style'=>'width:20%', 'class'=>'item'], 'value'=> function($data){return $data->getProduct()->One()->name;}],

           ['attribute'=>'Стоимость','options'=>['style'=>'width:20%', 'class'=>'price'], 'value'=> function($data){return $data->getProduct()->One()->price.' руб.';}],

           ['attribute'=>'count', 'value'=>'count','options'=>['style'=>'']],

            [

                'class' => 'yii\grid\ActionColumn', 'options'=>['style'=>'width:100px'],

                'urlCreator' => function ($action, Cart $model, $key, $index, $column) {

                    return Url::toRoute([$action, 'id\_cart' => $model->id\_cart]);

                },

                'buttonOptions'=>['class'=>'btn btn-default'],

                'template'=> '<div class="btn-group btn-group-sm text-center" role="group">{update}{delete}{view}{add}</div>',

            ],

        ],

    ]); ?>

# Отображение категории в товаре выпадающим списком

<?php

    $li=[];

    $categories=\app\models\Category::find()->all();

    foreach ($categories as $category)

    {

        $li[$category->id\_category]=$category->name\_category;

    }

    ?>

* В product/\_form.php

<?= $form->field($model, 'category')->dropDownList($li) ?>

# Отображение товара с фото и категорией

<?= DetailView::widget([

        'model' => $model,

        'attributes' => [

           // 'id\_product',

            'name',

            'price',

            ['attribute'=>'Фото', 'format'=>'html',

                'value'=>function($data){return"<img src='{$data->image}' alt='photo' style='width: 120px;'>";}],

            'country',

            ['attribute'=>'Категория', 'value'=>function($data){return $data->getCategory()->one()->name\_category;}],

            'color',

            'count',

            'description',

        ],

    ]) ?>

# Решение и отклонение проблем

<?= GridView::widget([

'dataProvider' => $dataProvider,

'filterModel' => $searchModel,

'columns' => [

['class' => 'yii\grid\SerialColumn'],

'id\_problem',

'name\_problem',

'description\_problem:ntext',

'date',

'status',

[

'class' => 'yii\grid\ActionColumn',

'template' => '{cancel} {solve}',

'buttons' => [

'cancel' => function ($url, $model) {

if ($model -> status == 'Новая') {

return Html::a('Отклонить', ['/problem/cancel', 'id\_problem'=>$model->id\_problem]);}},

'solve' => function ($url, $model) {

if ($model -> status == 'Новая') {

return Html::a('Решить', ['/problem/solve', 'id\_problem'=>$model->id\_problem]);}

},],],],]); ?>

Добавляем 2 модели ProblemCancel и ProblemSolve

ProblemCancel:

<?php

namespace app\models;

use Yii;

use yii\base\Model;

use app\models\User;

/\*\*

\* This is the model class for table "problem".

\*

\* @property int $id\_problem

\* @property string $name\_problem

\* @property string $description\_problem

\* @property string $date

\* @property int $user\_id

\* @property int $category\_id

\* @property string $status

\* @property string $photoBefore

\* @property string|null $photoAfter

\* @property string|null $reason

\*

\*

\* @property Category $category

\* @property User $user

\*/

class ProblemCancelForm extends Problem

{

/\*\*

\* {@inheritdoc}

\*/

public function rules()

{

return [

[['reason'], 'required']];

}

public function attributeLabels(){

return [

'reason' => 'Причина отказа',

];}}

ProblemSolve:

<?php

namespace app\models;

use Yii;

use yii\base\Model;

/\*\*

\* This is the model class for table "problem".

\*

\* @property int $id\_problem

\* @property string $name\_problem

\* @property string $description\_problem

\* @property string $date

\* @property int $user\_id

\* @property int $category\_id

\* @property string $status

\* @property string $photoBefore

\* @property string|null $photoAfter

\*

\* @property Category $category

\* @property User $user

\*/

class ProblemSolveForm extends Problem

{

/\*\*

\* {@inheritdoc}

\*/

public function rules(){

return [

[['photoAfter'], 'required'],

[['photoAfter'], 'file', 'extensions' => 'png, jpg, jpeg, bmp', 'maxSize' => 10\*1024\*1024],

];}}

ProblemController:

public function actionCancel($id\_problem){

$model = ProblemCancelForm::findOne($id\_problem);

if ($this->request->isPost && $model->load($this->request->post())) {

$model->status = 'Отклонена';

$model->save();

return $this->redirect(['index']);

}

return $this->render('cancel', [

'model' => $model,

]);

}

public function actionSolve($id\_problem){

$model = ProblemSolveForm::findOne($id\_problem);

$time = time();

if ($this->request->isPost) {

if ($model->load($this->request->post())) {

$model->photoAfter = UploadedFile::getInstance($model, 'photoAfter');

$model->photoAfter->saveAs('web/uploads/' . $model->photoAfter->baseName .$time. '.' . $model->photoAfter->extension);

$file\_name = ($model->photoAfter->baseName . $time . '.' . $model->photoAfter->extension);

$model->photoAfter = $file\_name;

$model->status = 'Решена';

$model->date = new Expression('NOW()');

$model->save();

return $this->redirect('/problem');}}

else {

$model->loadDefaultValues();

}

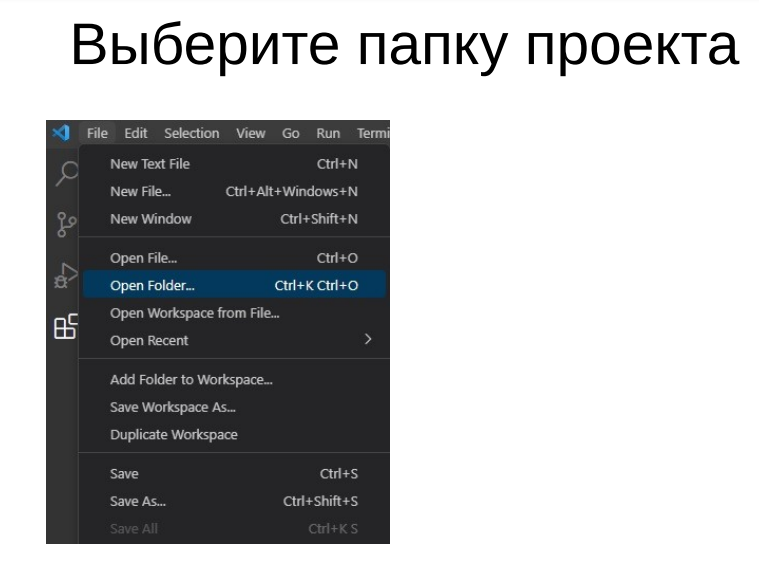
return $this->render('solve', [

'model' => $model,

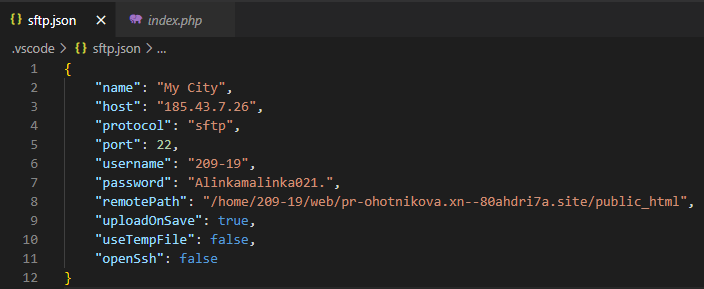
]);}

# Настройка Visual Studio Code

* Создаем пустую папку и открываем ее



* Ctrl+shift+p и выбираем sftp config



* Сохраняем файл
* Загружаем проект в локальную папку

