Министерство Высшего и среднего специального
Образования Республики Узбекистан

Национальный Университет Узбекистана

Механико-математический факультет

Кафедра “Информатики и Прикладного
Программирования”

Тема: **Разработка распределенной системы моделирования и дистанционного обучения английскому языку**

Выпускная квалификационная работа

Выполнила: студентка IV курса

направления «Прикладная математика

и информатика»

Махаммадиева Дилбар Рахматиллаевна

Научный руководитель:

доц. Ташпулатов Ф.А.

Предварительная защита выпускной квалификационной работы была проведена на заседании кафедры: протокол № 11 от 23 мая 2014 года.

Ташкент 2014

ВВЕДЕНИЕ

С момента обретения нашей страной независимости международные отношения Узбекистана со странами мирового сообщества перешли на качественно новый уровень. В период глобализации экономики, интеграции политической и культурной жизни, способствующих активизации и укреплению международных связей в различных сферах, большую роль играет знание иностранных языков, в частности, английского как языка международного общения.

В декабре 2012 года Президент Узбекистана И.А. Каримов подписал постановление «О мерах по дальнейшему совершенствованию системы изучения иностранных языков», касающееся узбекской учебной программы преподавания иностранных языков, и рекомендовал запустить ее в 2013-2014 учебном году. В основном, понятием «иностранный язык» подразумевается английский.

Одной из наиболее развивающихся формой обучения является дистанционное обучение. Она дает сегодня возможность создания систем массового непрерывного самообучения, всеобщего обмена информацией, независимо от временных и пространственных поясов. Кроме того, системы дистанционного образования предоставляют равные возможности всем людям независимо от социального положения (школьникам, студентам, гражданским и военным, безработными и т. д.) в любых районах страны и за рубежом реализовать права человека на образование и получение информации. Именно эта система может наиболее адекватно и гибко реагировать на потребности общества и обеспечить реализацию конституционного права на образование каждого гражданина страны. Исходя из вышеуказанных факторов можно заключить, что дистанционное обучение войдет в 21 век как самая эффективная система подготовки и непрерывного поддержания высокого квалификационного уровня специалистов. В связи с этим темой для выпускной

квалификационной работы была выбрана “Разработка распределенной системы моделирования и дистанционного обучения английскому языку”.

Для реализации своего проекта, мною был выбран Yii Framework, как один из самых развивающихся фреймворков. Для хранения всех данных была использована MySQL DataBase. Данные из этих таблиц будут обрабатываться с помощью объектно –ориентированного языка программирования php, который является основным языком разработки приложений для Yii Framework.

Результатом данной работы является создание Web-приложения по обработке информации созданной базы данных при помощи всех вышеперечисленных компонентов.

Цели и задачи проекта

Цель: Создание web – приложения для дистанционного обучения английскому языку, с использованием базы данных, разработанной на MySQL DataBase.

На сегодняшний день информационные технологии развиваются быстрым темпом. Все большее количество людей получают доступ в сеть Интернет. Все больше учреждений используют информационные системы для организации и управления внутренними процессами. Это касается и образовательных структур.

Информационная система это организационно-упорядоченная взаимосвязанная совокупность средств, и методов ИТ, а также используемых для хранения, обработки и выдачи информации в интересах достижения поставленной цели. Такое понимание информационной системы предполагает использование в качестве основного технического средства переработки информации ЭВМ и средств связи, реализующих информационные процессы и выдачу информации, необходимой в процессе принятия решений задач из любой области.

ИС является средой, составляющими элементами которой являются компьютеры, компьютерные сети, программные продукты, БД, люди, различного рода технические и программные средства связи и т.д. Хотя сама идея ИС и некоторые принципы их организации возникли задолго до появления компьютеров, однако компьютеризация в десятки и сотни раз повысила эффективность ИС и расширила сферы их применения.

Внедрение информационных технологий в образовательный процесс позволяет значительно облегчить организацию множества внутривузовских процессов, сбор информации о студентах, преподавателях и другом персонале, систематизацию и анализ результатов обучения. Также внедрение ИТ позволяет реализовывать многие новые формы, например, дистанционное обучение. При этом само понятие дистанционного обучения содержит в себе различные методы, например, обучение посредством удаленного тестирования или просмотра обучающих видеоуроков.

В связи с этим возникла необходимость создания среды для организации поддержки и управления дистанционным учебным процессом через обучение посредством просмотров видео, чтение книг, лекций и прослушивание аудио. Организовывать весь учебный процесс – это отдельная задача, поэтому за основную задачу была принята разработка веб-приложения для информационного обеспечения обучающихся через просмотры видеокурсов, аудиофайлов и обучающих лекций. Данная система должна позволять студентам проходить обучение из любой точки мира, где есть доступ в сеть Интернет. Преподаватели и авторы должны иметь возможность загружать видеолекции и назначать группы студентов к их просмотру. Также должен быть реализован модуль, отвечающий за обратную связь обучающихся с преподавателями, чтобы студенты могли задать свои вопросы по видеоурокам напрямую преподавателю.

Приложение I к дипломной работе на тему:

Разработка распределенной системы моделирования и дистанционного обучения английскому языку

-1-

-2-

-3-

EnglishClub 17 YEARS | HOME For Everyone | LEARN ENGLISH For Learners | TEACH ENGLISH For Teachers | MyEnglishClub Member Pages

the world's premier FREE website for learners + teachers of English

About EnglishClub | Join EnglishClub | Blog | Joe's Cafe | 7 Secrets | Power of 7 | Language Tools
Word of the Day | Weekly News | This Week in History | Guestbook | FREE Downloads

Only two English words in current use end in -gry.
What are they?

Interesting Facts About English

EnglishClub.com : Learn English : Vocabulary : Survival English : Grocery Shopping in English

Grocery Shopping in English

Listen to podcast of article.

Grocery Shopping in English [0.01 MB] Hide Player | Play in Popup | Download

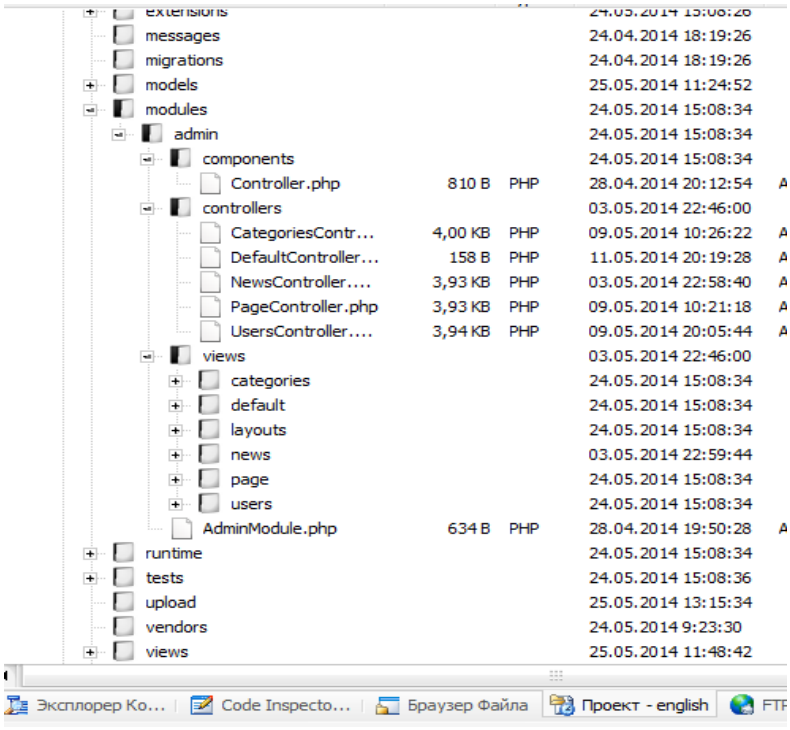
A grocery store is also called a **supermarket** or a **greengrocer**. Chain grocery stores are referred to by name. Learn the names of these stores before you travel to a new country. Superstores or department stores often have full grocery sections inside.

Shop Smart

Grocery stores are set up for you to buy more than you need. Fresh foods and **staples** are usually placed in the outside aisles or at the far end of a grocery store.

At the front you will find **convenience** foods and sale items. At the checkout you will find things you probably don't need, such as chocolate bars and magazines. The store is counting on you to throw a few of these items into your cart or basket. This is called impulse shopping. Can you **resist the temptation**?

-4-



CategoriesController

```
<?php

class CategoriesController extends Controller
{
	/**
	 * @var string the default layout for the views. Defaults to '//layouts/column2', meaning
	 * using two-column layout. See 'protected/views/layouts/column2.php'.
	 */
	public $layout='/layouts/column2';

	/**
	 * @return array action filters
	 */
	public function filters()
	{
		return array(
			'accessControl', // perform access control for CRUD operations
			'postOnly + delete', // we only allow deletion via POST request
		);
	}

	/**
	 * Specifies the access control rules.
	 * This method is used by the 'accessControl' filter.
	 * @return array access control rules
	 */
	public function accessRules()
	{
		return array(
```

```
                array('allow',  // allow all users to perform 'index' and 'view' actions
                        'actions'=>array('index','view'),
                        'users'=>array('*'),
                ),
                array('allow', // allow authenticated user to perform 'create' and 'update' actions
                        'actions'=>array('create','update'),
                        'users'=>array('@'),
                ),
                array('allow', // allow admin user to perform 'admin' and 'delete' actions
                        'actions'=>array('admin','delete'),
                        'users'=>array('admin'),
                ),
                array('deny',  // deny all users
                        'users'=>array('*'),
                ),
        );
}

/**
 * Displays a particular model.
 * @param integer $id the ID of the model to be displayed
 */
public function actionView($id)
{
        $this->render('view',array(
                'model'=>$this->loadModel($id),
        ));
}

/**
 * Creates a new model.
 * If creation is successful, the browser will be redirected to the 'view' page.
 */
public function actionCreate()
{
        $model=new Categories;

        // Uncomment the following line if AJAX validation is needed
        // $this->performAjaxValidation($model);

        if(isset($_POST['Categories']))
        {
                $model->attributes=$_POST['Categories'];
                if($model->save())
                        $this->redirect(array('view','id'=>$model->id));
        }

        $this->render('create',array(
                'model'=>$model,
        ));
}

/**
```

```
	 * Updates a particular model.
	 * If update is successful, the browser will be redirected to the 'view' page.
	 * @param integer $id the ID of the model to be updated
	 */
	public function actionUpdate($id)
	{
		$model=$this->loadModel($id);

		// Uncomment the following line if AJAX validation is needed
		// $this->performAjaxValidation($model);

		if(isset($_POST['Categories']))
		{
			$model->attributes=$_POST['Categories'];
			if($model->save())
				$this->redirect(array('view','id'=>$model->id));
		}

		$this->render('update',array(
			'model'=>$model,
		));
	}

	/**
	 * Deletes a particular model.
	 * If deletion is successful, the browser will be redirected to the 'admin' page.
	 * @param integer $id the ID of the model to be deleted
	 */
	public function actionDelete($id)
	{
		$this->loadModel($id)->delete();

		// if AJAX request (triggered by deletion via admin grid view), we should not redirect the
browser
		if(!isset($_GET['ajax']))
			$this->redirect(isset($_POST['returnUrl']) ? $_POST['returnUrl'] : array('admin'));
	}

	/**
	 * Lists all models.
	 */
	/**
	 * Manages all models.
	 */
	public function actionIndex()
	{
		$model=new Categories('search');
		$model->unsetAttributes();  // clear any default values
		if(isset($_GET['Categories']))
			$model->attributes=$_GET['Categories'];

		$this->render('index',array(
			'model'=>$model,
```

```
			));
		}

		/**
		 * Returns the data model based on the primary key given in the GET variable.
		 * If the data model is not found, an HTTP exception will be raised.
		 * @param integer $id the ID of the model to be loaded
		 * @return Categories the loaded model
		 * @throws CHttpException
		 */
		public function loadModel($id)
		{
			$model=Categories::model()->findByPk($id);
			if($model===null)
				throw new CHttpException(404,'The requested page does not exist.');
			return $model;
		}

		/**
		 * Performs the AJAX validation.
		 * @param Categories $model the model to be validated
		 */
		protected function performAjaxValidation($model)
		{
			if(isset($_POST['ajax']) && $_POST['ajax']==='categories-form')
			{
				echo CActiveForm::validate($model);
				Yii::app()->end();
			}
		}
}
```

PageController

```
<?php

class PageController extends Controller
{
		/**
		 * @var string the default layout for the views. Defaults to '//layouts/column2', meaning
		 * using two-column layout. See 'protected/views/layouts/column2.php'.
		 */
		public $layout='/layouts/column2';

		/**
		 * @return array action filters
		 */
		public function filters()
		{
			return array(
				'accessControl', // perform access control for CRUD operations
```

```
                'postOnly + delete', // we only allow deletion via POST request
            );
    }

    /**
     * Specifies the access control rules.
     * This method is used by the 'accessControl' filter.
     * @return array access control rules
     */
    public function accessRules()
    {
        return array(
            array('allow',  // allow all users to perform 'index' and 'view' actions
                'actions'=>array('index','view'),
                'users'=>array('*'),
            ),
            array('allow', // allow authenticated user to perform 'create' and 'update'
actions
                'actions'=>array('create','update'),
                'users'=>array('@'),
            ),
            array('allow', // allow admin user to perform 'admin' and 'delete' actions
                'actions'=>array('admin','delete'),
                'users'=>array('admin'),
            ),
            array('deny',  // deny all users
                'users'=>array('*'),
            ),
        );
    }

    /**
     * Displays a particular model.
     * @param integer $id the ID of the model to be displayed
     */
    public function actionView($id)
    {
        $this->render('view',array(
            'model'=>$this->loadModel($id),
        ));
    }

    /**
     * Creates a new model.
     * If creation is successful, the browser will be redirected to the 'view' page.
     */
    public function actionCreate()
    {
```

```
		$model=new Page;

		// Uncomment the following line if AJAX validation is needed
		// $this->performAjaxValidation($model);

		if(isset($_POST['Page']))
		{
			$model->attributes=$_POST['Page'];
			if($model->save())
				$this->redirect(array('view','id'=>$model->id));
		}

		$this->render('create',array(
			'model'=>$model,
		));
	}

	/**
	 * Updates a particular model.
	 * If update is successful, the browser will be redirected to the 'view' page.
	 * @param integer $id the ID of the model to be updated
	 */
	public function actionUpdate($id)
	{
		$model=$this->loadModel($id);

		// Uncomment the following line if AJAX validation is needed
		// $this->performAjaxValidation($model);

		if(isset($_POST['Page']))
		{
			$model->attributes=$_POST['Page'];
			if($model->save())
				$this->redirect(array('view','id'=>$model->id));
		}

		$this->render('update',array(
			'model'=>$model,
		));
	}

	/**
	 * Deletes a particular model.
	 * If deletion is successful, the browser will be redirected to the 'admin' page.
	 * @param integer $id the ID of the model to be deleted
	 */
	public function actionDelete($id)
	{
```

```
		$this->loadModel($id)->delete();

		// if AJAX request (triggered by deletion via admin grid view), we should not redirect the browser
		if(!isset($_GET['ajax']))
			$this->redirect(isset($_POST['returnUrl']) ? $_POST['returnUrl'] : array('admin'));
	}

	/**
	 * Lists all models.
	 */


	/**
	 * Manages all models.
	 */
	public function actionIndex()
	{
		$model=new Page('search');
		$model->unsetAttributes();  // clear any default values
		if(isset($_GET['Page']))
			$model->attributes=$_GET['Page'];

		$this->render('index',array(
			'model'=>$model,
		));
	}

	/**
	 * Returns the data model based on the primary key given in the GET variable.
	 * If the data model is not found, an HTTP exception will be raised.
	 * @param integer $id the ID of the model to be loaded
	 * @return Page the loaded model
	 * @throws CHttpException
	 */
	public function loadModel($id)
	{
		$model=Page::model()->findByPk($id);
		if($model===null)
			throw new CHttpException(404,'The requested page does not exist.');
		return $model;
	}

	/**
	 * Performs the AJAX validation.
	 * @param Page $model the model to be validated
	 */
```

```
	protected function performAjaxValidation($model)
	{
		if(isset($_POST['ajax']) && $_POST['ajax']==='page-form')
		{
			echo CActiveForm::validate($model);
			Yii::app()->end();
		}
	}
}
```

NewsController

```
<?php

class NewsController extends Controller
{
	/**
	 * @var string the default layout for the views. Defaults to '//layouts/column2', meaning
	 * using two-column layout. See 'protected/views/layouts/column2.php'.
	 */
	public $layout='/layouts/column2';

	/**
	 * @return array action filters
	 */
	public function filters()
	{
		return array(
			'accessControl', // perform access control for CRUD operations
			'postOnly + delete', // we only allow deletion via POST request
		);
	}

	/**
	 * Specifies the access control rules.
	 * This method is used by the 'accessControl' filter.
	 * @return array access control rules
	 */
	public function accessRules()
	{
		return array(
			array('allow',  // allow all users to perform 'index' and 'view' actions
				'actions'=>array('index','view'),
				'users'=>array('*'),
			),
			array('allow', // allow authenticated user to perform 'create' and 'update' actions
				'actions'=>array('create','update'),
				'users'=>array('@'),
			),
			array('allow', // allow admin user to perform 'admin' and 'delete' actions
				'actions'=>array('admin','delete'),
				'users'=>array('admin'),
			),
			array('deny',  // deny all users
```

```
                        'users'=>array('*'),
                ),
        );
}

/**
 * Displays a particular model.
 * @param integer $id the ID of the model to be displayed
 */
public function actionView($id)
{
        $this->render('view',array(
                'model'=>$this->loadModel($id),
        ));
}

/**
 * Creates a new model.
 * If creation is successful, the browser will be redirected to the 'view' page.
 */
public function actionCreate()
{
        $model=new News;

        // Uncomment the following line if AJAX validation is needed
        // $this->performAjaxValidation($model);

        if(isset($_POST['News']))
        {
                $model->attributes=$_POST['News'];
                if($model->save())
                        $this->redirect(array('view','id'=>$model->id));
        }

        $this->render('create',array(
                'model'=>$model,
        ));
}

/**
 * Updates a particular model.
 * If update is successful, the browser will be redirected to the 'view' page.
 * @param integer $id the ID of the model to be updated
 */
public function actionUpdate($id)
{
        $model=$this->loadModel($id);

        // Uncomment the following line if AJAX validation is needed
        // $this->performAjaxValidation($model);

        if(isset($_POST['News']))
        {
```

```
                $model->attributes=$_POST['News'];
                if($model->save())
                        $this->redirect(array('view','id'=>$model->id));
        }

        $this->render('update',array(
                'model'=>$model,
        ));
    }

    /**
     * Deletes a particular model.
     * If deletion is successful, the browser will be redirected to the 'admin' page.
     * @param integer $id the ID of the model to be deleted
     */
    public function actionDelete($id)
    {
        $this->loadModel($id)->delete();

        // if AJAX request (triggered by deletion via admin grid view), we should not redirect the
browser
        if(!isset($_GET['ajax']))
            $this->redirect(isset($_POST['returnUrl']) ? $_POST['returnUrl'] : array('admin'));
    }

    /**
     * Lists all models.
     */
    /**
     * Manages all models.
     */
    public function actionIndex()
    {
        $model=new News('search');
        $model->unsetAttributes();  // clear any default values
        if(isset($_GET['News']))
            $model->attributes=$_GET['News'];

        $this->render('index',array(
            'model'=>$model,
        ));
    }

    /**
     * Returns the data model based on the primary key given in the GET variable.
     * If the data model is not found, an HTTP exception will be raised.
     * @param integer $id the ID of the model to be loaded
     * @return News the loaded model
     * @throws CHttpException
     */
    public function loadModel($id)
    {
        $model=News::model()->findByPk($id);
```

```
		if($model===null)
			throw new CHttpException(404,'The requested page does not exist.');
		return $model;
	}

	/**
	 * Performs the AJAX validation.
	 * @param News $model the model to be validated
	 */
	protected function performAjaxValidation($model)
	{
		if(isset($_POST['ajax']) && $_POST['ajax']==='news-form')
		{
			echo CActiveForm::validate($model);
			Yii::app()->end();
		}
	}
}
```

UsersController

```
<?php

class UsersController extends Controller
{
	/**
	 * @var string the default layout for the views. Defaults to '//layouts/column2', meaning
	 * using two-column layout. See 'protected/views/layouts/column2.php'.
	 */
	public $layout='/layouts/column2';

	/**
	 * @return array action filters
	 */
	public function filters()
	{
		return array(
			'accessControl', // perform access control for CRUD operations
			'postOnly + delete', // we only allow deletion via POST request
		);
	}

	/**
	 * Specifies the access control rules.
	 * This method is used by the 'accessControl' filter.
	 * @return array access control rules
	 */
	public function accessRules()
	{
		return array(
			array('allow',  // allow all users to perform 'index' and 'view' actions
				'actions'=>array('index','view'),
```

```
                        'users'=>array('*'),
                ),
                array('allow', // allow authenticated user to perform 'create' and 'update' actions
                        'actions'=>array('create','update'),
                        'users'=>array('@'),
                ),
                array('allow', // allow admin user to perform 'admin' and 'delete' actions
                        'actions'=>array('admin','delete'),
                        'users'=>array('admin'),
                ),
                array('deny',  // deny all users
                        'users'=>array('*'),
                ),
        );
}

/**
 * Displays a particular model.
 * @param integer $id the ID of the model to be displayed
 */
public function actionView($id)
{
        $this->render('view',array(
                'model'=>$this->loadModel($id),
        ));
}

/**
 * Creates a new model.
 * If creation is successful, the browser will be redirected to the 'view' page.
 */
public function actionCreate()
{
        $model=new Users;

        // Uncomment the following line if AJAX validation is needed
        // $this->performAjaxValidation($model);

        if(isset($_POST['Users']))
        {
                $model->attributes=$_POST['Users'];
                if($model->save())
                        $this->redirect(array('view','id'=>$model->id));
        }

        $this->render('create',array(
                'model'=>$model,
        ));
}

/**
 * Updates a particular model.
 * If update is successful, the browser will be redirected to the 'view' page.
```

```
	 * @param integer $id the ID of the model to be updated
	 */
	public function actionUpdate($id)
	{
		$model=$this->loadModel($id);

		// Uncomment the following line if AJAX validation is needed
		// $this->performAjaxValidation($model);

		if(isset($_POST['Users']))
		{
			$model->attributes=$_POST['Users'];
			if($model->save())
				$this->redirect(array('view','id'=>$model->id));
		}

		$this->render('update',array(
			'model'=>$model,
		));
	}

	/**
	 * Deletes a particular model.
	 * If deletion is successful, the browser will be redirected to the 'admin' page.
	 * @param integer $id the ID of the model to be deleted
	 */
	public function actionDelete($id)
	{
		$this->loadModel($id)->delete();

		// if AJAX request (triggered by deletion via admin grid view), we should not redirect the
browser
		if(!isset($_GET['ajax']))
			$this->redirect(isset($_POST['returnUrl']) ? $_POST['returnUrl'] : array('admin'));
	}

	/**
	 * Lists all models.
	 */
	/**
	 * Manages all models.
	 */
	public function actionIndex()
	{
		$model=new Users('search');
		$model->unsetAttributes();  // clear any default values
		if(isset($_GET['Users']))
			$model->attributes=$_GET['Users'];

		$this->render('index',array(
			'model'=>$model,
		));
	}
```

```
	/**
	 * Returns the data model based on the primary key given in the GET variable.
	 * If the data model is not found, an HTTP exception will be raised.
	 * @param integer $id the ID of the model to be loaded
	 * @return Users the loaded model
	 * @throws CHttpException
	 */
	public function loadModel($id)
	{
		$model=Users::model()->findByPk($id);
		if($model===null)
			throw new CHttpException(404,'The requested page does not exist.');
		return $model;
	}

	/**
	 * Performs the AJAX validation.
	 * @param Users $model the model to be validated
	 */
	protected function performAjaxValidation($model)
	{
		if(isset($_POST['ajax']) && $_POST['ajax']==='users-form')
		{
			echo CActiveForm::validate($model);
			Yii::app()->end();
		}
	}

}
```

DefaultController

```
<?php

class DefaultController extends Controller
{
   public $layout='/layouts/column2';
	public function actionIndex()
	{
		$this->render('index');
	}


}
```

Главный шаблон админки

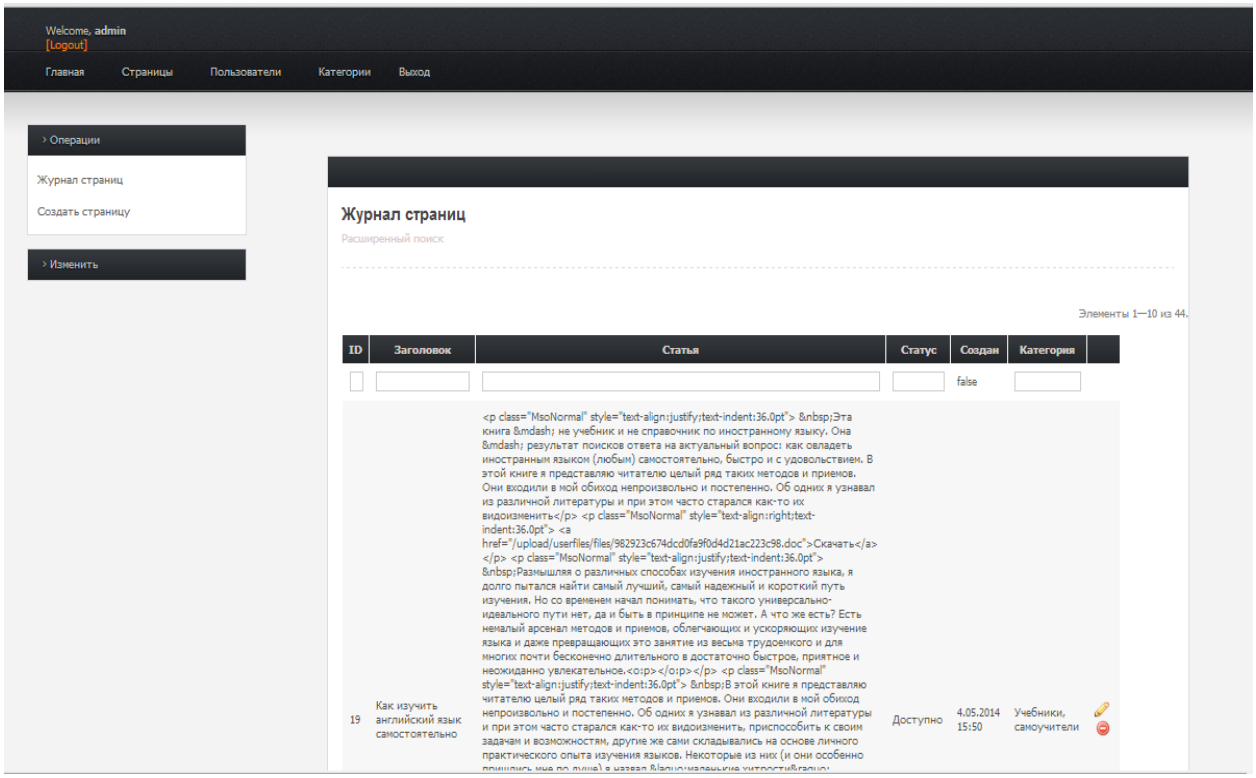
```
<?php /* @var $this Controller */ ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
```

```
<link rel="stylesheet" type="text/css" href="<?php echo Yii::app()->request->baseUrl;
?>/css/admin/style.css"  media="screen" />
<link rel="stylesheet" type="text/css" href="<?php echo Yii::app()->request->baseUrl;
?>/css/admin/navi.css" media="screen" />
<script type="text/javascript" src="<?php echo Yii::app()->request->baseUrl; ?>/js/admin/jquery-
1.7.2.min.js"></script>
<script type="text/javascript">
$(function(){
	$(".box .h_title").not(this).next("ul").hide("normal");
	$(".box .h_title").not(this).next("#home").show("normal");
	$(".box").children(".h_title").click( function() { $(this).next("ul").slideToggle(); });
});
</script>

	<title><?php echo CHtml::encode($this->pageTitle); ?></title>
</head>

<div class="wrap">
	<div id="header">
		<div id="top">
			<div class="left">
				<p>Welcome, <strong>
	 <?php
		if(Yii::app()->user->checkAccess('2')){
	    echo Yii::app()->user->name;
	   }
	   ?>
	 </strong>
	  <?php $this->widget('zii.widgets.CMenu',array(
							'items'=>array(array('label'=>'[Logout]',
'url'=>array('site/logout'), 'visible'=>!Yii::app()->user->isGuest)),
		'htmlOptions'=>array('class'=>'new'),
							)); ?>

	</p>
			</div>
			<div class="right">

			</div>
		</div>
		<div id="nav">
			<ul>
				<li class="upp"><a
href="http://localhost/english/index.php?r=site/index">Главная</a>
				</li>
	<li class="upp"><a href="http://localhost/english/index.php?r=admin/page">Страницы</a>
					<ul>
						<li>&#8250; <a
href="http://localhost/english/index.php?r=admin/page/create">Создать</a></li>
						<li>&#8250; <a
href="http://localhost/english/index.php?r=admin/page/update">Редактировать</a></li>
						<li>&#8250; <a
href="http://localhost/english/index.php?r=admin/page">Список</a></li>
```

```
                                        </ul>
                                    </li>
                                    <li class="upp"><a href="#">Пользователи</a>
                                        <ul>
            <li>&#8250; <a href="">Создать</a></li>
                                            <li>&#8250; <a href="">Редактировать</a></li>
                                            <li>&#8250; <a href="">Список</a></li>
                                        </ul>
                                    </li>
                                    <li class="upp"><a href="#">Категории</a>
                                        <ul>
                                            <li>&#8250; <a href="">Создать</a></li>
                                            <li>&#8250; <a href="">Редактировать</a></li>
                                            <li>&#8250; <a href="">Список</a></li>
                                        </ul>
                                    </li>
          <a href="#"> <?php $this->widget('zii.widgets.CMenu',array(
                                    'items'=>array(
                                        array('label'=>'Выход', 'url'=>array('/site/logout'),
'visible'=>!Yii::app()->user->isGuest),
                    'htmlOptions'=>array('class'=>'upp'),
                    'linkOptions'=>array('class'=>'upp')
                                    ),
                            )); ?>
          </a>

                            </ul>
                        </div>
                </div>

                <div id="content">
                        <div id="sidebar">
                            <div class="box">
                                <div class="h_title">&#8250; Операции</div>
                        <ul id="home">
                    <li class="b1">

                        <?php
                            $this->beginWidget('zii.widgets.CPortlet');
                            $this->widget('zii.widgets.CMenu', array(
                                'items'=>$this->menu,
                                'htmlOptions'=>array('class'=>'b1'),
                            ));
                            $this->endWidget();
                        ?>
                    </li>
                                    </ul>
                            </div>

                            <div class="box">
                                <div class="h_title">&#8250; Изменить</div>
                                <ul>
                                    <li class="b1">
```

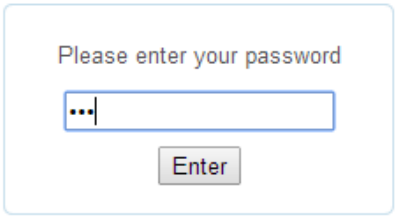
```
                <?php $this->widget('zii.widgets.CMenu',array(
                                'items'=>array(
                                        array('label'=>'Страницы', 'url'=>array('/admin/page')),
                                        array('label'=>'Пользователи', 'url'=>array('/admin/users')),
                                        array('label'=>'Категории', 'url'=>array('/admin/categories')),
                    array('label'=>'Новости', 'url'=>array('/admin/news'))
                                ),
                        )); ?>
            </li>
                                </ul>
                        </div>
                </div>
                <div id="main">

                        <?php echo $content; ?>

                </div>
                <div class="clear"></div>
        </div>

        <div id="footer">
                <div class="left">
                        <p>Design: <a href="http://kilab.pl">Paweł Balicki</a> | Admin Panel: <a
href="">YourSite.com</a></p>
                </div>
                <div class="right">
                        <p><a href="">Example link 1</a> | <a href="">Example link 2</a></p>
                </div>
        </div>
</div>

</body>
</html>
```

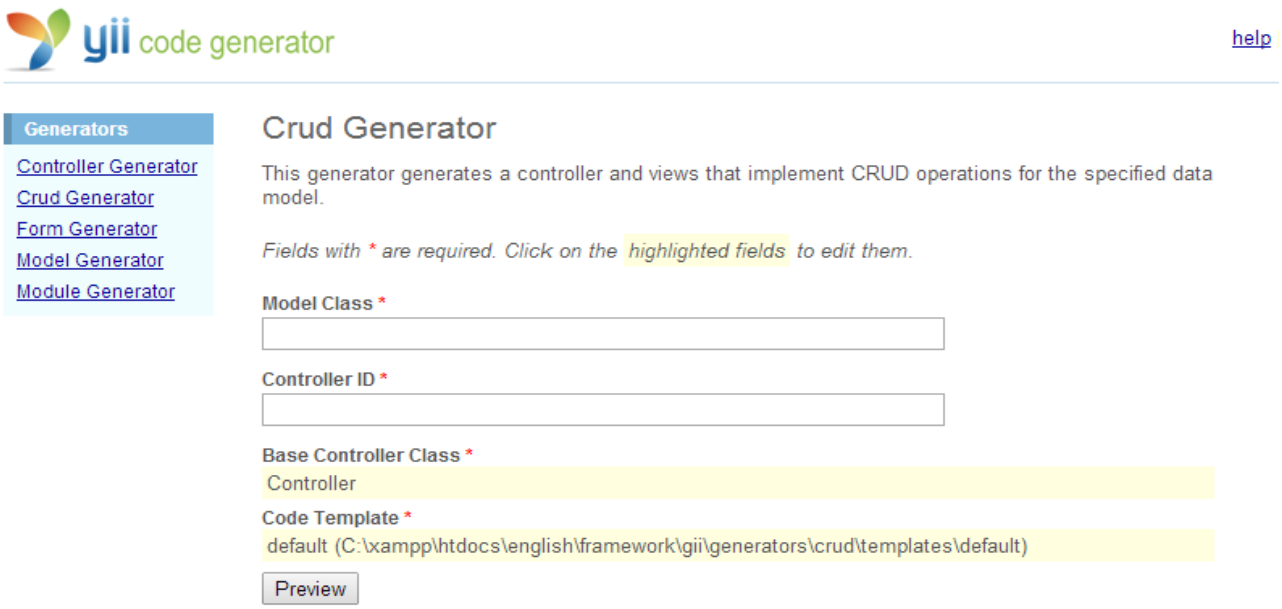
-5-


Welcome, admin
[Logout]
Главная
Страницы
Пользователи
Категории
Выход
› Операции
Журнал страниц
Создать страницу
› Изменить
Журнал страниц
Расширенный поиск
Элементы 1—10 из 44.
ID
Заголовок
Статья
Статус
Создан
Категория
false
19
Как изучить английский язык самостоятельно
Доступно
4.05.2014 15:50
Учебники, самоучители

-6-


yii code generator


Please enter your password
Enter



-7-

```
<?php
        class UserIdentity extends CUserIdentity {
  // Будем хранить id.
  protected $_id;

  // Данный метод вызывается один раз при аутентификации пользователя.
  public function authenticate(){
    // Производим стандартную аутентификацию, описанную в руководстве.
    $user = Users::model()->find('LOWER(username)=?', array(strtolower($this->username)));
    if(($user===null) || ($this->password!==$user->password)) {
      $this->errorCode = self::ERROR_USERNAME_INVALID;
    } else {
      // В качестве идентификатора будем использовать id, а не username,
      // как это определено по умолчанию. Обязательно нужно переопределить
      // метод getId(см. ниже).
      $this->_id = $user->id;

      // Далее логин нам не понадобится, зато имя может пригодится
      // в самом приложении. Используется как Yii::app()->user->name.
      // realName есть в нашей модели. У вас это может быть name, firstName
      // или что-либо ещё.
      $this->username = $user->username;
      $this->errorCode = self::ERROR_NONE;
    }
   return !$this->errorCode;
  }

  public function getId(){
    return $this->_id;
```

```
   }
}
?>
```

-8-

SiteController

```
<?php

class SiteController extends Controller
{
	/**
	 * Declares class-based actions.
	 */
	public function actions()
	{
		return array(
			// captcha action renders the CAPTCHA image displayed on the contact page
			'captcha'=>array(
				'class'=>'CCaptchaAction',
				'backColor'=>0xFFFFFF,
			),
			// page action renders "static" pages stored under 'protected/views/site/pages'
			// They can be accessed via: index.php?r=site/page&view=FileName
			'page'=>array(
				'class'=>'CViewAction',
			),
		);
	}

	/**
	 * This is the default 'index' action that is invoked
	 * when an action is not explicitly requested by users.
	 */

	public function actionIndex()
	{

      // renders the view file 'protected/views/site/index.php'
		// using the default layout 'protected/views/layouts/main.php'
		$this->render('index');
	}


	public function actionArticles()

	{
		$this->render('articles');

	}

	/**
```

```
	 * This is the action to handle external exceptions.
	 */
	public function actionError()
	{
		if($error=Yii::app()->errorHandler->error)
		{
			if(Yii::app()->request->isAjaxRequest)
				echo $error['message'];
			else
				$this->render('error', $error);
		}
	}

	/**
	 * Displays the contact page
	 */
	public function actionContact()
	{
		$model=new ContactForm;
		if(isset($_POST['ContactForm']))
		{
			$model->attributes=$_POST['ContactForm'];
			if($model->validate())
			{
				$name='=?UTF-8?B?'.base64_encode($model->name).'?=';
				$subject='=?UTF-8?B?'.base64_encode($model->subject).'?=';
				$headers="From: $name <{$model->email}>\r\n".
					"Reply-To: {$model->email}\r\n".
					"MIME-Version: 1.0\r\n".
					"Content-Type: text/plain; charset=UTF-8";

				mail(Yii::app()->params['adminEmail'],$subject,$model->body,$headers);
				Yii::app()->user->setFlash('contact','Thank you for contacting us. We will
respond to you as soon as possible.');
				$this->refresh();
			}
		}
		$this->render('contact',array('model'=>$model));
	}

	/**
	 * Displays the login page
	 */
	public function actionLogin()
	{
    $model=new LoginForm;

		// if it is ajax validation request
		if(isset($_POST['ajax']) && $_POST['ajax']==='login-form')
		{
			echo CActiveForm::validate($model);
			Yii::app()->end();
		}
```

```
                // collect user input data
                if(isset($_POST['LoginForm']))
                {
                        $model->attributes=$_POST['LoginForm'];
                        // validate user input and redirect to the previous page if valid
                        if($model->validate() && $model->login())
                                $this->redirect(array('/admin/default/index'));
                }
                // display the login form
                $this->renderPartial('login',array('model'=>$model));
        }
        public function actionRegistration()
        {
    $model=new Users;
    $model->scenario='registration';

                // collect user input data
                if(isset($_POST['Users']))
                {
                        $model->attributes=$_POST['Users'];
                        if($model->save()){
                         Yii::app()->user->setFlash('registration','Thank you for registration');
       }

                }
                // display the login form
                $this->render('registration',array('model'=>$model));
        }

        /**
         * Logs out the current user and redirect to homepage.
         */
        public function actionLogout()
        {
                Yii::app()->user->logout();
                $this->redirect(Yii::app()->homeUrl);
        }

}
```

NewsController

```
<?php

class NewsController extends Controller
{
   //const $STATUS_PUBLISHED;

   public function actionSearchPost()
   {
      $arResult = array();

     $searchParam = $_POST['SiteSearchForm']['string'];
```

```
//      if(isset($_POST['SiteSearchForm']['string']) && !empty($_POST['SiteSearchForm']['string']))
//      {
         $connection = Yii::app()->db;
         $sql = "
               SELECT *
               FROM  `news`
               WHERE  `title` LIKE  '%{$searchParam}%' OR `content` LIKE  '%{$searchParam}%'
               ";

         $command = $connection->createCommand($sql);
         //$command->bindParam(":searchParam", $_POST['SiteSearchForm']['string'], PDO::PARAM_STR);

//       echo '<pre>';
//       print_r($command);
//       echo '</pre>';

         $arResult = $command->queryAll();

 //      }
    $this->render('search',array('model'=> $arResult));

   }

          public function actionIndex($id)
          {
                    $models = News::model()->findAllByAttributes(array('id'=>$id));

      $this->render('index',array('models'=>$models));
          }
   public function actionView($id)
          {
                    $model = News::model()->findByPk($id);

      $this->render('view',array('model'=>$model));
          }
   public function search($id)
          {
                    $model = News::model()->findByPk($id);

      $this->render('view',array('model'=>$model));
          }


   public function accessRules()
          {
                    // NOTE: you should only define rules for those attributes that
                    // will receive user inputs.
                    return array(
         array('allow','actions'=>array('index','view','postedinmonth','postedondate','search'),
                                        'users'=>array('*'),
            )
                              // The following rule is used by search().
                              // @todo Please remove those attributes that should not be searched.
```

```
                );
        }


}
```

PageController

```
<?php

class PageController extends Controller
{
   //const $STATUS_PUBLISHED;

   public function actionSearchPost()
   {
      $arResult = array();

     $searchParam = $_POST['SiteSearchForm']['string'];
//      if(isset($_POST['SiteSearchForm']['string']) && !empty($_POST['SiteSearchForm']['string']))
//      {
         $connection = Yii::app()->db;
         $sql = "
               SELECT *
               FROM  `page`
               WHERE  `title` LIKE  '%{$searchParam}%' OR `content` LIKE  '%{$searchParam}%'
               ";

         $command = $connection->createCommand($sql);
         //$command->bindParam(":searchParam", $_POST['SiteSearchForm']['string'], PDO::PARAM_STR);

//       echo '<pre>';
//       print_r($command);
//       echo '</pre>';

         $arResult = $command->queryAll();

 //      }
    $this->render('search',array('model'=> $arResult));

   }

          public function actionIndex($id)
          {
                $models = Page::model()->findAllByAttributes(array('category_id'=>$id));
      if ($id == 4){
         $this->render('audio',array('models'=>$models));
      }
      else
      $this->render('index',array('models'=>$models));
          }
   public function actionView($id)
          {
                $model = Page::model()->findByPk($id);
```

```
    $this->render('view',array('model'=>$model));
        }
  public function search($id)
        {
                $model = Page::model()->findByPk($id);

    $this->render('view',array('model'=>$model));
        }


  public function accessRules()
        {
                // NOTE: you should only define rules for those attributes that
                // will receive user inputs.
                return array(
      array('allow','actions'=>array('index','view','postedinmonth','postedondate','search'),
                                        'users'=>array('*'),
          )
                        // The following rule is used by search().
                        // @todo Please remove those attributes that should not be searched.
                );
        }


}
```

Главный шаблон:

```
<?php /* @var $this Controller */ ?>
<!DOCTYPE HTML>
<html>
<head>
        <meta http-equiv="content-type" content="text/html" />
        <link rel="stylesheet" type="text/css" href="<?php echo Yii::app()->request->baseUrl;
?>/css/style.css" />

  <link rel="stylesheet" type="text/css" href="<?php echo Yii::app()->request->baseUrl; ?>/css/styles.css">
  <script type="text/javascript" src="<?php echo Yii::app()->request->baseUrl; ?>/js/jquery.js"></script>
  <script type="text/javascript" src="<?php echo Yii::app()->request->baseUrl; ?>/js/jquery.smooth-
scroll.js"></script>
  <script type="text/javascript" src="<?php echo Yii::app()->request->baseUrl; ?>/js/scripts.js"></script>
        <title><?php echo CHtml::encode($this->pageTitle); ?></title>
</head>

<body>
  <div class="wrapper">
    <div class="main">
      <div class="header">
      <div class="left"></div>
        <ul>
                                <?php $this->widget('zii.widgets.CMenu',array(
                                        'items'=>array(
```

```
                                                array('label'=>'ГЛАВНАЯ', 'url'=>array('/site/index')),
                                                array('label'=>'О НАС', 'url'=>array('/site/page',
'view'=>'about')),
                                                array('label'=>'IELTS', 'url'=>array('/site/articles')),
                                                array('label'=>'КОНТАКТЫ', 'url'=>array('/site/contact')),
            array('label'=>'МОЙ КАБИНЕТ', 'url'=>array('/admin/'),'visible'=>!Yii::app()->user->isGuest),
            array('label'=>'РЕГИСТРАЦИЯ', 'url'=>array('/site/registration'),'visible'=>Yii::app()->user-
>isGuest),
                                                array('label'=>'ВОЙТИ', 'url'=>array('/site/login'),
'visible'=>Yii::app()->user->isGuest),
                                                array('label'=>'ВЫЙТИ',
'url'=>array('/site/logout'),'visible'=>!Yii::app()->user->isGuest),
                                                //array('label'=>'SITE MAP', 'url'=>array('/site/contact')),
                                            ),
                                    )); ?>
        </ul>
      <div class="right"></div>
      </div>
      <div class="center">
        <div class="search">
         <?php $this->widget('SiteSearch'); ?>
        </div>
        <div class="sidebar">
          <div class="categories">
            <h4>КАТЕГОРИИ</h4>

              <?php
                    $this->beginWidget('zii.widgets.CPortlet');
                    $this->widget('zii.widgets.CMenu', array(
                        'items'=>Categories::menu(),
                        'htmlOptions'=>array('class'=>'operations'),
                    ));
                    $this->endWidget();
               ?>
          </div>
          <div class="mail">
            <p>
              <h3>Рассылка</h3>
              <input type="text" class="enter" placeholder="Введите свой E-mail">
              <a href="">Отписаться</a>
              <a href=""><input type="button" value="Подписка" class="button2"></a>
            </p>
          </div>
          <div class="news">
            <h4>СВЕЖИЕ НОВОСТИ</h4>
            <?php
              $this->beginWidget('NewsWidget');
              $this->endWidget();
              ?>
            <p class="date">June 30, 2010</p>
            <p class="a_title">Sed ut perspiciatis unde</p>
            <p class="text">Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium
doloremque.</p>
```

```
                <p class="date">June 14, 2010</p>
                <p class="a_title">Neque porro quisquam est</p>
                <p class="text">Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit
consequuntur magni.</p>

                <p class="date">May 29, 2010</p>
                <p class="a_title">Minima veniam, quis nostrum</p>
                <p class="text">Uis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil
molestiae.</p>
              </div>
            </div>
            <div class="logo_pic">
                <div class="block-slider container">
                  <ul>
                    <li style="background-image: url(<?php echo Yii::app()->request->baseUrl;
?>/images/slide1.png);">
                            </li>
                    <li style="background-image: url(<?php echo Yii::app()->request->baseUrl;
?>/images/slide3.jpg);">
                             </li>
                    <li style="background-image: url(<?php echo Yii::app()->request->baseUrl;
?>/images/slide5.jpg);">
                            </li>
                    <li style="background-image: url(<?php echo Yii::app()->request->baseUrl;
?>/images/slide6.jpg);">
                            </li>
                    <li style="background-image: url(<?php echo Yii::app()->request->baseUrl;
?>/images/slide7.jpg);">
                            </li>
                    <li style="background-image: url(<?php echo Yii::app()->request->baseUrl;
?>/images/slide8.jpg);">
                            </li>
                    <li style="background-image: url(<?php echo Yii::app()->request->baseUrl;
?>/images/slide10.jpg);">
                            </li>
                    <li style="background-image: url(<?php echo Yii::app()->request->baseUrl;
?>/images/slide12.jpg);">
                            </li>
                    <li style="background-image: url(<?php echo Yii::app()->request->baseUrl;
?>/images/slide13.jpg);">
                            </li>
                    <li style="background-image: url(<?php echo Yii::app()->request->baseUrl;
?>/images/slide14.jpg);">
                            </li>
                    <li style="background-image: url(<?php echo Yii::app()->request->baseUrl;
?>/images/slide15.jpg);">
                            </li>
                  </ul>
                </div>

             </div>
            <div class="content">
```

```
                                        <?php echo $content; ?>
            </div>
          </div>
          <div class="clear"></div>

        <div class="footer">
            <p class="tel">24/7 ГОРЯЧАЯ ЛИНИЯ <span>8.800.146.56.74</span></p>
            <div class="about">Все права защищены. Разработка сайта: Дильбар</div>
        </div>
      </div>
    </div>
    <script type="text/javascript">
          $(window).load(function(){
                var $container = $('.portfolioContainer');
                $container.isotope({
                      filter: '*',
                      animationOptions: {
                            duration: 750,
                            easing: 'linear',
                            queue: false
                      }
                });

                $('.portfolioFilter a').click(function(){
                      $('.portfolioFilter .current').removeClass('current');
                      $(this).addClass('current');

                      var selector = $(this).attr('data-filter');
                      $container.isotope({
                            filter: selector,
                            animationOptions: {
                                  duration: 750,
                                  easing: 'linear',
                                  queue: false
                            }
                       });
                       return false;
                });
          });

</script>
</body>
</html>
```

-9-

```
<?php

class SiteSearch extends CWidget
{
          public function run()
          {
          $form = new SiteSearchForm();
          $this->render('siteSearch', array('form'=>$form));
```

```
        }
}
public function actionSearchPost()
   {
      $arResult = array();

     $searchParam = $_POST['SiteSearchForm']['string'];
//      if(isset($_POST['SiteSearchForm']['string']) && !empty($_POST['SiteSearchForm']['string']))
//      {
         $connection = Yii::app()->db;
         $sql = "
               SELECT *
               FROM  `page`
               WHERE  `title` LIKE  '%{$searchParam}%' OR `content` LIKE  '%{$searchParam}%'
               ";

         $command = $connection->createCommand($sql);
         //$command->bindParam(":searchParam", $_POST['SiteSearchForm']['string'], PDO::PARAM_STR);

//       echo '<pre>';
//       print_r($command);
//       echo '</pre>';

         $arResult = $command->queryAll();

 //      }
    $this->render('search',array('model'=> $arResult));

   }
```

Целью работы является информационное обеспечение очного и дистанционного обучения через просмотры видеокурсов и прослушивание аудиофайлов.

Анализ и оценка существующих систем

Перед тем, как приступить к реализации проекта, следует проанализировать существующие на данный момент системы подобного характера и тематики. С помощью этого мы можем сделать верные выводы о том, как необходимо создавать собственную систему на основе достоинств и недостатков рассмотренных ресурсов.

В качестве первого аналога был выбран образовательный сайт “Английский язык по Раймонду Мёрфи”, (http://www. english03.ru). Данный сайт направлен на обучение посредством видеоуроков, лекций, фильмов и книг. В данной системе реализованы функции загрузки/просмотра видеолекций, ограничения просмотра определенным пользователям, комментирования видеолекций. Приложение I- [1]

В качестве достоинства данной системы стоит отметить скорость доступа к видеолекциям, а также традиционность доступа к видеоинформации.

Также можно считать достоинством наличие простого доступа пользователям сети Интернет и особенно зарегистрированным пользователям к информации, что должно способствовать популяризации канала.

Наличие комментариев к видео подразумевает существование учетных записей в системе, однако эти учетные записи принадлежат пользователям.

Вторым аналогом можно назвать образовательный сайт “ LEARN AMERICAN ENGLISH ”, (http://learnamericanenglishonline.com). Приложение I -[2]

Проект явно реализовывался под какую-то конкретную задачу, поскольку за последние 6 месяцев на него не было загружено ни одного видеоурока. Главным достоинством данного проекта является возможность выбора уровня обучения, таким образом, обучающийся сможет проходить курс, просматривая материалы по порядку.

Также стоит отметить, что на проекте существует форум (на данный момент закрытый), в котором шли обсуждения на тему улучшения данных курсов. Форум соответственно предполагал наличие учетных записей, которые хранились в базе данных этого проекта. Какого-то механизма загрузки данных найти не удалось, но сами видеофайлы хранятся на сервере проекта.

Наконец, третьим аналогом можно назвать образовательный портал ENGLISH CLUB (http://edition.englishclub.com/survival/grocery-shopping/). Отличительной особенностью этого раздела является его многогранность. Этот портал предназначен не только для студентов, но и для учителей. Каждый из разделов разделен на подразделы: чтение, письмо, разговор, слушание. Так же стоит отметить, что любой пользователь сможет зарегистрироваться в их клубе, после чего у него будет свой кабинет, где он сможет делиться своими материалами с другими членами этого клуба. Приложение I -[3]

По моему мнению, это отличный, очень быстрый англоязычный сайт. Много текстов озвучено, с возможностью бесплатно скачать в Mp3. Есть флэш карты с возможностью распечатать. Хорошо работает поисковая система. Найти можно почти всё, только необходимо зарегистрироваться.

Предмет и объект исследования.

Предметом исследования в данной дипломной работе является Yii Framework, написанный на языке PHP. Используя данный фреймворк, мы получаем возможность исследования поведения нашего объекта, а именно, веб-приложение, которое будет использоваться для обеспечения учебного процесса.

Прикладное значение результата.

Данный проект ориентирован на любого пользователя, желающего получить знания по английскому языку. С его помощью пользователи смогут получать необходимую информацию и делиться ею. Сайт предназначен для удобного предоставления пользователям статей, лекций, видео и аудио материалов.

Он должен привлекать постоянную аудиторию, за счет которой посещаемость сайта будет постоянно расти, а вместе с этим и позиции сайта в поисковых системах, так как в последнее время огромное влияние на позиции сайта в поисковых системах играет поведенческий фактор.

ГЛАВА 1

Выбор программных средств

1.1.Выбор веб-сервера

Веб-сервер — сервер, принимающий HTTP-запросы от клиентов, обычно веб-браузеров, и выдающий им HTTP-ответы, как правило, вместе с HTML-страницей, изображением, файлом, медиа-потоком или другими данными.

Веб-сервером называют как программное обеспечение, выполняющее функции веб-сервера, так и непосредственно компьютер, на котором это программное обеспечение работает.

Клиент, которым обычно является веб-браузер, передаёт веб-серверу запросы на получение ресурсов, обозначенных URL-адресами. **Ресурсы** — это HTML-страницы, изображения, файлы, медиа-потоки или другие данные, которые необходимы клиенту. В ответ веб-сервер передаёт клиенту запрошенные данные. Этот обмен происходит по протоколу HTTP.

На август 2013 года наиболее распространённым веб-сервером, занимающим более 65 % рынка**, является Apache** — свободный веб-сервер, наиболее часто используемый в UNIX-подобных операционных системах;

Apache HTTP Server поддерживает модульность. Существует более 500 модулей, выполняющих различные функции. Часть из них разрабатывается командой Apache Software Foundation, но основное количество — отдельными open source-разработчиками.

Модули могут быть как включены в состав сервера в момент компиляции, так и загружены динамически, через директивы конфигурационного файла.

В модулях реализуются такие вещи, как:

- Поддержка языков программирования.

- Добавление функций.
- Исправление ошибок или модификация основных функций.
- Усиление безопасности.

Механизм виртуальных хостов

Apache имеет встроенный механизм виртуальных хостов. Он позволяет полноценно обслуживать на одном IP-адресе множество сайтов (доменных имён), отображая для каждого из них собственное содержимое.

Для каждого виртуального хоста можно указать собственные настройки ядра и модулей, ограничить доступ ко всему сайту или отдельным файлам. Некоторые MPM, например Apache-ITK позволяют запускать процесс httpd для каждого виртуального хоста с отдельными идентификаторами uid и guid.

Также, существуют модули, позволяющие учитывать и ограничивать ресурсы сервера (CPU, RAM, трафик) для каждого виртуального хоста.

Интеграция с другим ПО и языками программирования.

Существует множество модулей, добавляющих к Apache поддержку различных языков программирования и систем разработки.

К ним относятся:

- PHP (mod_php).
- Python (mod python, mod wsgi).
- Ruby (apache-ruby).
- Perl (mod perl).
- ASP (apache-asp)[6].
- Tcl (rivet[7])

Кроме того, Apache поддерживает механизмы CGI и FastCGI, что позволяет исполнять программы на практически всех языках программирования, в том числе C, C++, Lua, sh, Java.

Безопасность

Apache имеет различные механизмы обеспечения безопасности и разграничения доступа к данным. Основными являются:

- Ограничение доступа к определённым директориям или файлам.
- Механизм авторизации пользователей для доступа к директории на основе HTTP-аутентификации (mod_auth_basic) и digest-аутентификации (mod_auth_digest).
- Ограничение доступа к определённым директориям или всему серверу, основанное на IP-адресах пользователей.
- Запрет доступа к определённым типам файлов для всех или части пользователей, например запрет доступа к конфигурационным файлам и файлам баз данных.
- Существуют модули, реализующие авторизацию через СУБД или PAM.

В некоторых MPM-модулях присутствует возможность запуска каждого процесса Apache используя различные uid и gid с соответствующими этим пользователям и группам пользователей.

Также, существует механизм suexec, используемый для запуска скриптов и CGI-приложений с правами и идентификационными данными пользователя.

Для реализации шифрования данных, передающихся между клиентом и сервером используется механизм SSL, реализованный через библиотеку OpenSSL. Для удостоверения подлинности веб-сервера используются сертификаты X.509.

Существуют внешние средства обеспечения безопасности, например mod_security.

1.2. Выбор СУБД

Самыми известными СУБД при работе с сайтами являются MySQL, PostgreSQL. Также используются MSSQL, Oracle, Firebird и некоторые другие. Большая популярность MySQL и PostgreSQL по сравнению с проприетарными СУБД обусловлена большим сообществом разработчиков, открытостью продуктов и огромными возможностями по настройке быстродействия баз данных.

Если сравнивать конкретно MySQL и PostgreSQL, то можно выявить следующие преимущества MySQL:

- *соответствие стандартам SQL* начиная с пятой версии MySQL большое внимание разработчиками удалялось соответствию стандартам SQL. В MySQL запросы максимально соответствют стандартам SQL'99;

- *большее количество платформ* MySQL изначально разрабатывался как кроссплатформенная СУБД. В Windows MySQL работает как служба, в *nix – как демон. PostgreSQL изначально разрабатывался как СУБД, работающая в *nix-системах;

- *скорость работы на простых запросах* – огромное преимущество MySQL заключается именно в скорости работы простых запросов. Благодаря тому, что в MySQL используются различные типы таблицам, а типом таблиц по умолчанию является MyISAM, реализуется огромная скорость при работе с простыми запросами. В то же время, тип тпблиц InnoDB позволяет осуществлять транзакции, следить за целостностью данных, но в данной случае уже не будет выигрыша в скорости запросов;

- *стабильность работы* – исторически сложилось, что MySQL довольно стабильная СУБД. PostgreSQL – более молодая, в то время как из-за более раннего начала разработки у MySQL сложилось большее сообщество разработчиков;

- *безопасность, связанная со стабильностью* – сообщество разработчиков MySQL за все эти годы нашли и устранили огромное количество уязвимостей, что позволяет считать MySQL одной из самых защищенных СУБД;

- *работа с большими объектами* в MySQL реализована поддержка бинарных объектов практически неограниченных размеров в полях типа BLOB, что отсутствует у PostgreSQL;

- *возможности для легкого изменения таблиц* – в MySQL реализованы возможности легкого изменения таблиц, что отсутствует в PostgreSQL.

В то же время у PostgreSQL есть свои преимущества:

- *стабильность* – несмотря на то, что сообщество разработчиков MySQL больше, сама PostgreSQL изначально проектировалась как более стабильная СУБД. Плюс в этом свою роль сыграло то, что MySQL долго избавлялся от наследия своих третьей и четвертой сравнительно нестабильных версий;

- *скорость работы (процедуры)* PostgreSQL выигрывает в производительности на сложных запросах, логически построенных процедурах;

- *целостность данных* – PostgreSQL позволяет оперировать с данными, не перекладывая логику на ЯП. При разработке кода программисту не придется думать о целостности данных в БД;

- *специальные вещи (триггеры, процедуры, функции...)* - многие вещи, которые реализуются в MySQL только в последних релизах.

Как можно увидеть, главным преимуществом MySQL являются скорость работы на простых запросах (логика БД довольно простая и не требует процедур для реализации). Это преимуществом было выбрано в качестве основного при выборе СУБД.

В то же время было необходимо выбрать тип таблиц MySQL. Исторически сложилось, что типом таблиц по умолчанию в MySQL является MyISAM. Вторым по популярности типом таблиц является InnoDB. В настоящее время разрабатывается альтернатива InnoDB – Falcon, однако использование его на production-серверах не рекомендуется. В то же время существуют и другие типы таблиц, например:

- *HEAP* (все хранится в памяти)
- *MERGE* (совокупность таблиц MyISAM)
- *Maria* (обновленный MyISAM с возможностями транзакций)

При анализе преимущества и недостатков стандартного типа таблиц MySQL были выявлены следующие его преимущества:

- *полнотекстовый поиск*
- *преимущество в скорости на простых выборках*
- *работа “из коробки”*

Анализ преимуществ InnoDB выявил следующие пункты:

- *поддержка транзакций*
- *целостность/внешние ключи*
- *преимущество в скорости на сложных выборках*

- *более полное соответствие стандартам*

Как можно увидеть, InnoDB позволяет переложить логику на СУБД, в то время как стандартный тип таблиц позволяет использовать преимущество простых выборок (а их будет гораздо больше чем сложных). Также в MyISAM реализована возможность полнотекстового поиска (хотя она довольно требовательна к наличию индексов). И что немаловажно возможность работы с типом “из коробки” [13]. Трудно сказать, преимущество это или недостаток, однако при развертывании каких-либо систем преимущество отдается проверенным продуктам. Настройка движка InnoDB до сих пор является довольно объемной темой, проработка которой не относится к написанию дипломной работы. Поэтому было отдано предпочтение типу таблиц MyISAM.

Вывод: Беря во внимание вышеизложенные факты, мною было принято решение в качестве веб-сервера выбрать готовую сборку XAMPP, в состав которого входят сразу два важных для меня пакета: Apache, MySQL. Проще говоря, XAMPP- это кроссплатформенная сборка веб-сервера, содержащая Apache, MySQL, интерпретатор скриптов PHP, язык программирования Perl и большое количество дополнительных библиотек, позволяющих запустить полноценный веб-сервер. К тому же, пользовательский интерфейс программы настолько прост, что ее называют «сборкой для ленивых. Установка XAMPP занимает меньше времени, чем установка каждого компонента в отдельности. Данный web-сервер распространяется в полной, стандартной и уменьшенной версиях. Все дополнительные модули также доступны для скачивания.

1.3. Выбор серверного языка программирования и фреймворка

Связующим звеном между СУБД и веб-сервером является язык программирования. В настоящее время самыми популярными и ипользуемыми серверными ЯП при разработке веб-приложений являются python, php, jsp и ruby. Стандартом ЯП для создания клиентских веб-приложений в сети

Интернет является javascript. Вопросом для разработчика в настоящее время становится лишь *“какой фреймворк для работы с javascript стоит выбрать”*. Самыми известными фреймворками являются jquery, prototype, dojo, extjs.

К серверным языкам веб-программирования можно отнести:

PHP – это, пожалуй, один из самых популярных языков на сегодня. Благодаря этой популярности можно легко отыскать специалиста по PHP для создания и поддержки сайта. Этот язык прост и легок в освоении. Расшифровывается как «Hypertext Preprocessor». Файлы, содержащие код php, имеют расширение .php.

PHP позволяет создавать динамические веб-страницы с разнообразным функционалом. На сегодняшний день накоплено уже большое количество готовых решений в виде скриптов, которые можно быстро подключить к вашему сайту.

Perl – еще один популярный язык веб-программирования, который, правда, реже используется при создании веб-приложений. Данный язык был создан Ларри Уоллом в 1987 году, а само название расшифровывается как Practical Extraction and Report Language , что означает «практический язык для извлечения данных и составления отчётов».

Интересно отметить, что неким «талисманом» языка Perl является верблюд, символизирующий способность выполнять тяжелую работу.

Главным достоинством языка Perl является скорость и богатые возможности по работе с текстами. Широко применяемые сегодня в других языках веб-программирования регулярные выражения изначально встроены в Perl.

ASP.NET – технология компании Microsoft, предназначенная для создания веб-страниц. Технология ASP.NET неотделима от платформы Microsoft .NET. По сути, ASP.NET – это именно технология, а не язык, которая позволяет разработчикам писать код на любых языках программирования, которые входят в пакет .NET. К ним относятся: C#, Visual Basic.NET и JScript .NET

Разработка и поддержка веб-сайтов на основе ASP.NET обычно дорогостояща, поэтому используется не так часто.

Итак, в качестве серверного ЯП был выбран php, который на данный момент является самым попялурным серверным языком программирования для

создания веб-приложений. PHP обладает рядом преимуществ, приведенных ниже.

Главными факторами PHP являются предоставление средств для быстрого и эффективного решения поставленных задач и практичность, обусловленная шестью важными характеристиками:

- *традиционностью* – многие конструкции языка позаимствованы из других известных языков программирования, что позволяет прикладывать меньше усилий при знакомстве с ним и его изучении. PHP специально нацелен на работу в сети Интернет. На сегодняшний день PHP является одним из популярных языков для создания веб-приложений;

- *простотой* – сценарий PHP может состоять из большого числа строк или из одной строки — все зависит от специфики поставленной задачи. Программисту не приходится подгружать библиотеки или указывать специальные параметры компиляции. Механизм PHP просто начинает выполнять код после первой экранирующей последовательности (<?) и продолжает выполнение до того момента, когда он встретит парную экранирующую последовательность (?>). Если код имеет правильный синтаксис, он исполняется в точности так, как указал программист. PHP – язык, который может быть встроен непосредственно в HTML-код страниц, которые, в свою очередь будут корректно обрабатываться PHP-интерпретатором. Большое разнообразие функций PHP избавят вас от написания многострочных пользовательских функций. В то же время существует больше количество фреймворков и CMS, написанных как разработчиками-одиночками, так и большими сообществами программистов;

- *эффективностью* – важное преимущество PHP заключается в том, что он не нуждается в компиляторе, и позволяет обрабатывать сценарии непосредственно на сервере. По некоторым оценкам, большинство PHP-сценариев (особенно не очень больших размеров) обрабатываются быстрее

аналогичных им программ, написанных на других ЯП. Однако, чтобы не делали разработчики PHP, откомпилированные исполняемые файлы будут работать значительно быстрее – в десятки, а иногда и в сотни раз, поскольку откомпилированные программы по сути являются уже инструкциями в машинном коде, в то время как интерпретатор PHP лишь построчно исполняет инструкции, описанные программистом. В то же время, производительность PHP вполне достаточна для создания вполне объемных и многофункциональных веб-приложений;

- *безопасностью* – PHP предоставляет в распоряжение разработчиков и администраторов гибкие и эффективные средства безопасности, такие как, например, механизмы безопасности, находящиеся под управлением администраторов; при правильной настройке PHP это обеспечивает максимальную свободу действий и безопасность. Например, можно ограничить максимальное время выполнения и использование памяти (неконтролируемый расход памяти отрицательно влияет на быстродействие сервера) или устанавливать ограничения на каталоги, в которых пользователь может просматривать и исполнять сценарии PHP, а также использовать сценарии PHP для просмотра конфиденциальной информации на сервере. В стандартный набор функций PHP входит также ряд надежных механизмов шифрования. Другое преимущество заключается в том, что исходный текст сценариев PHP нельзя просмотреть в браузере, поскольку сценарий интерпретируется до его отправки по запросу пользователя. Реализация PHP на стороне сервера предотвращает похищение нетривиальных сценариев;

- *гибкостью* – поскольку PHP является встраиваемым языком, он отличается исключительной гибкостью по отношению к потребностям разработчика. Хотя PHP обычно рекомендуется использовать в сочетании с HTML, он с таким же успехом интегрируется и в JavaScript, XML и другие языки. Нет проблем и с зависимостью от браузеров, поскольку PHP является

серверным ЯП и никак не связан с браузерами. В сущности, сценарии PHP могут передаваться любым устройствам с браузерами, включая сотовые телефоны, электронные записные книжки, пейджеры и портативные компьютеры, не говоря уже о традиционных ПК. PHP в целом является платформенно-независимым языком, поскольку он не содержит кода, ориентированного на конкретный веб-сервер. Благодаря этим возможностям PHP занимает достойное место среди современных технологий и обеспечивает масштабирование проектов до необходимых пределов;

- *бесплатным распространением* важным фактором в развитии проекта PHP оказалась поддержка пользователей со всего мира. Бесплатное распространение исходных текстов PHP оказало неоценимую услугу пользователям. Вдобавок, отзывчивое сообщество пользователей PHP является своего рода «коллективной службой поддержки», и в популярных электронных конференциях можно найти ответы даже на самые сложные вопросы.

Существуют и php-фреймворки, которые облегчают и упрощают процесс создания сайтов. Среди них можно выделить: Yii, CodeIgneiter, ZendFramework, CakePHP и многие другие. Для реализации поставленных задач, мною был выбран Yii Framework.

Yii – это фреймворк

Yii – это фреймворк написанный на языке PHP. Главным плюсом Yii является отличная поддержка ООП, скорость работы и, конечно же, тех поддержка от разработчиков. Фреймворк включает в себя большой набор библиотек, которые помогут в создании полноценного веб приложения отвечающего всем современным стандартам (интегрированное использованием Ajax, встроенная поддержка интернационализации приложения, простой инструмент работы с базой данных). Тот код, который занимал бы 100 строчек чистого php кода может быть сокращен до десяти благодаря встроенным методам фреймворка. Yii является бесплатным программным обеспечением и распространяется под лицензией «new BSD».

Возможности

- Высокая производительность относительно других фреймворков написанных на PHP
- Парадигма Модель-вид-контроллер
- Интерфейсы DAO и ActiveRecord для работы с базами данных (PDO)
- Поддержка интернационализации
- Кэширование страниц и отдельных фрагментов
- Перехват и обработка ошибок
- Ввод и валидация форм
- Аутентификация и авторизация
- Использование AJAX и интеграция с jQuery
- Генерация базового PHP-кода для CRUD-операций (скаффолдинг)
- Поддержка тем оформления для их лёгкой смены
- Возможность подключения сторонних библиотек
- Миграции базы данных
- Автоматическое тестирование
- Поддержка REST

Структура сайта на yii практически такая же как и для всех остальных сайтов, есть папка для хранения изображений, стилей, тем и различных вспомогательных файлов, добавляется лишь папка с самим фреймворком и папка protected, в которой реализуется весь функционал сайта.

Рассмотрим папку protected, в ней содержится множество папок, но мы рассмотрим 4 из них, чтобы не забивать вам мозг лишней (пока лишней) информацией:

- config (файл main.php) — через этот файл можно настроить под ваши нужды работу сайта, например подключить какие-нибудь модули

наподобие ЧПУ, базу данных, GII (позже узнаете что это и как им пользоваться) и т.д.;

- controllers (контроллеры) — это папка с классами, которые управляют логикой проекта;
- models (модели) — это папка с моделями, которые определяют правила работы с базой данных (с информацией на сайте);
- views (представления) — эта папка содержит файлы отображения данных конечному пользователю.

Схема работы yii такова: при заходе пользователя на сайт (заполнении формы, переходе на другую страницу сайта и т.д.) начинает работать контроллер (какой контроллер начинает работать зависит от действий пользователя), который в свою очередь начинает использовать модели (если нужно извлечь какие-нибудь данные из базы данных или наоборот записать туда) и представления (если пользователю необходимо отобразить результаты работы контроллера).

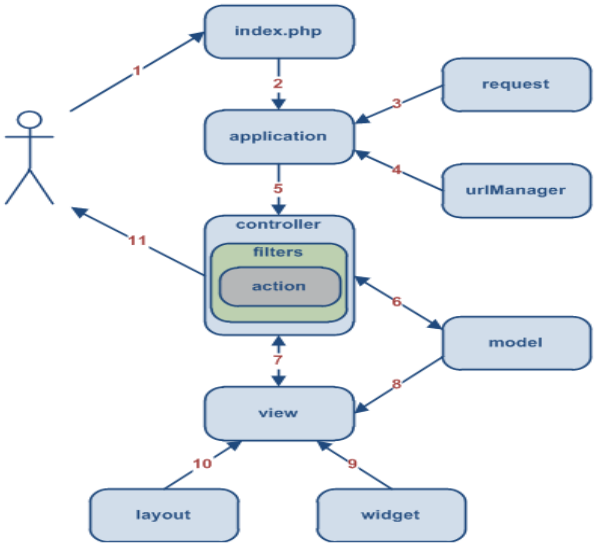
Контроллеры состоят из событий (action), которые могут выполнять различные действия. Все эти события (action) начинаются со слова action и заканчиваются нашей приставкой, например Index.

Например, когда пользователь нажимает на нашем тестовом сайте ссылку «Contact» активируется контроллер Site и действие «contact».

Следующая диаграмма описывает типичную последовательность процесса обработки пользовательского запроса приложением

1. Пользователь осуществляет запрос посредством URL http://www.example.com/index.php?r=post/show&id=1, и веб-сервер обрабатывает его, запуская скрипт инициализации index.php.
2. Скрипт инициализации создает экземпляр приложения и запускает его на выполнение.

3. Приложение получает подробную информацию о запросе пользователя от компонента приложения request.
4. Приложение определяет запрошенные контроллер и действие при помощи компонента urlManager. В данном примере контроллером будет post, относящийся к классу PostController, а действием — show, суть которого определяется контроллером.
5. Приложение создаёт экземпляр запрашиваемого контроллера для дальнейшей обработки запроса пользователя. Контроллер определяет соответствие действия show методу actionShow в классе контроллера. Далее создаются и применяются фильтры (например, access control, benchmarking), связанные с данным действием, и, если фильтры позволяют, действие выполняется.
6. Действие считывает из базы данных модель Post с ID равным 1.
7. Действие подключает представление show, передавая в него модель Post.
8. Представление получает и отображает атрибуты модели Post.
9. Представление подключает некоторые виджеты.
10. Сформированное представление вставляется в макет страницы.
11. Действие завершает формирование представления и выводит результат пользователю



1.4. Модель- Представление-Контроллер

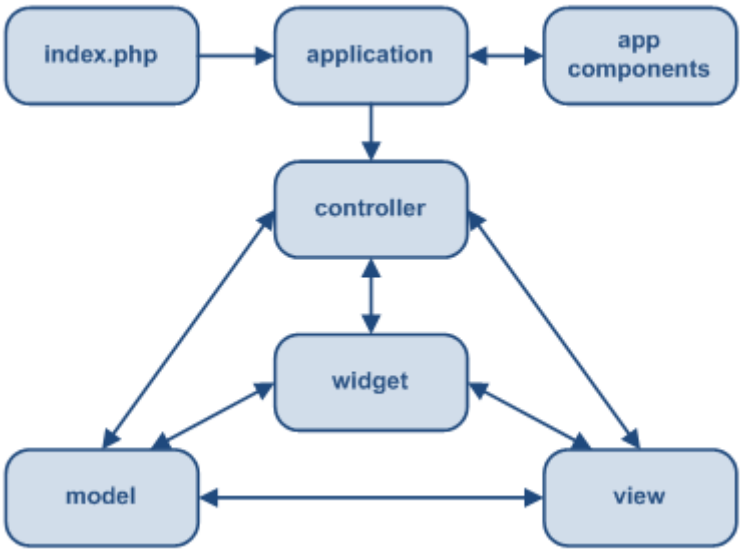
Yii использует шаблон проектирования Модель-Представление-Контроллер (MVC, Model-View-Controller), который широко применяется в веб-программировании.

MVC предназначен для разделения бизнес-логики и пользовательского интерфейса, чтобы разработчики могли легко изменять отдельные части приложения, не затрагивая другие. В архитектуре MVC модель предоставляет данные и правила бизнес-логики, представление отвечает за пользовательский интерфейс (например, текст, поля ввода), а контроллер обеспечивает взаимодействие между моделью и представлением.

Помимо этого, Yii использует фронт-контроллер, называемый приложением (application), который инкапсулирует контекст обработки запроса. Приложение собирает информацию о запросе и передает её для дальнейшей обработки соответствующему контроллеру.

Следующая диаграмма отображает структуру приложения Yii:

Статическая структура приложения Yii



ГЛАВА 2 . База Данных

2.1.Проектирование базы данных

Наша задача состоит в том, что нужно организовать и систематизировать работу сайта по обучению английскому языку.

Существует пользовательская часть (front end) и административная часть (back end). У каждого пользователя есть 3 статуса: обычный пользователь, зарегистрированный пользователь и администратор. Каждый из этих статусов предполагает различные возможности над управлением деятельностью сайта.

Целью создания базы данных является автоматизация учета информации о пользователях сайта.

Назначение проектируемой базы:

- хранение информации о пользователях сайта;
- хранение информации о страницах сайта;
- хранение информации о категориях;
- хранение информации о статьях;
- обновление, добавление и удаление информации.

2.2. Структура объектов БД

Анализ информации проектируемой базы данных позволяет выделить следующие основные информационные объекты:

Пользователи, категории, страницы и новости.

Атрибутами объекта **Пользователи** являются:

- Идентификатор пользователя
- Его имя
- Пароль
- Дата создания
- Роль
- Адрес электронной почты

Атрибутами объекта **Категории:**

- Идентификатор категории
- Название категории

Атрибутами объекта **Страницы** являются:

- Идентификатор страницы
- Наименование страницы
- Содержание страницы
- Ее статус (скрыто/доступно)
- Дата создания
- Идентификатор категории, к которой относится данная статья

Атрибутами объекта **Новости** являются:

- Идентификатор
- Наименование
- Содержание

- Статус
- Дата создания

Номер (код) для каждого объекта необходим для однозначной идентификации записей и в дальнейшем будет использоваться как ключевое поле.

2.3. Описание таблиц базы данных

Теперь составим описание типов каждого поля таблицы БД.

Таблица «Пользователи» содержит 6 полей. Листинг таблицы «Пользователи» :

Таблица «Пользователи»

Поле	**Тип поля**	**Свойства поля**
Идентификатор пользователя	integer	Ключевое поле Длина-11 символов
Имя	varchar	Обязательное поле. Длина- 255 символов.
Пароль	varchar	Обязательное поле. Длина- 255 символов.
Дата создания	Ineger	Длина- 11 символов.
Роль	tinyint	Обязательное поле. Длина -1 символ
E-mail	varchar	Обязательное поле.

		Длина -255 символов

Таблица «Категории» содержит 2 поля. Листинг таблицы «Категории:

Таблица «Категории»

Поле	**Тип поля**	**Свойства поля**
Идентификатор категории	integer	Ключевое поле Длина-11 символов
Название категории	varchar	Обязательное поле. Длина- 255 символов.

Таблица «Страницы» содержит 4 поля. Листинг таблицы «Страницы»:

Таблица «Страницы»

Поле	**Тип поля**	**Свойства поля**
Идентификатор страницы	integer	Ключевое поле Длина-11 символов
Наименование страницы	varchar	Обязательное поле. Длина- 255 символов.
Содержание страницы	text	Обязательное поле.

Ее статус	tinyint	Обязательное поле. Длина- 1 символ.
Дата создания	integer	Длина -1 символ
Идентификатор категории, к которой относится данная статья	integer	Обязательное поле. Длина -5 символов

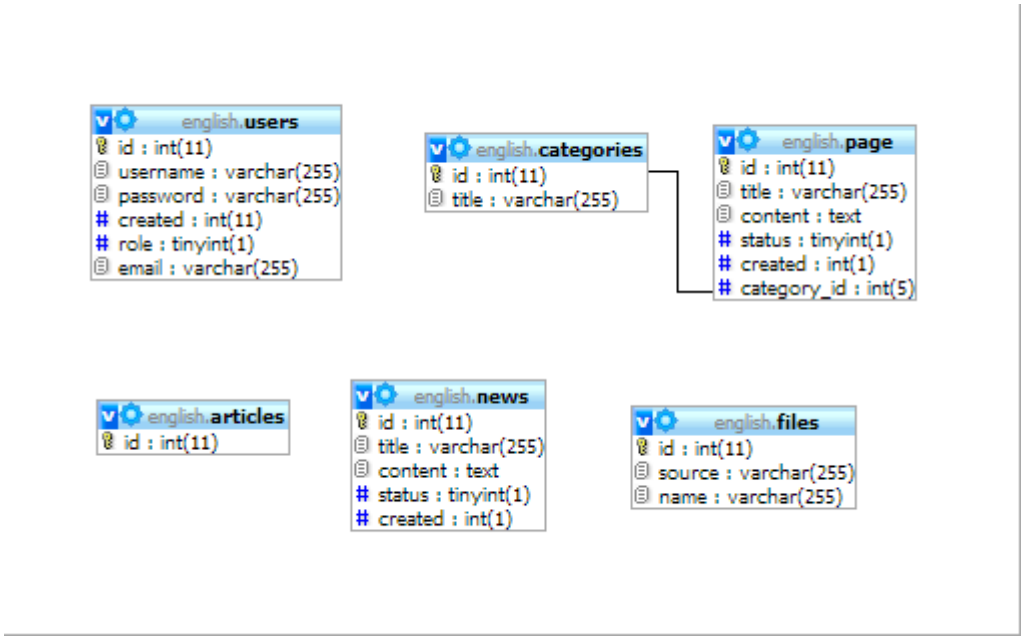
Таблица «Новости» содержит 4 поля. Листинг таблицы «Новости»:

Таблица «Новости»

Поле	**Тип поля**	**Свойства поля**
Идентификатор новостей	integer	Ключевое поле Длина-11 символов
Наименование новости	varchar	Обязательное поле. Длина- 255 символов.
Содержание новостей	text	Обязательное поле.
Ее статус	tinyint	Обязательное поле. Длина- 1 символ.
Дата создания	integer	Длина -1 символ

2.4. Определение взаимосвязей между таблицами

В данном проекте имеет смысл связать лишь две таблицы: категории и страницы, чтобы при создании страницы пользователь смог добавить ее лишь в имеющуюся категорию.



ГЛАВА 3 Разработка веб - приложения.

3.1.Разработка административной части сайта

Административная часть должна включать в себя: информацию о новых пользователях, раздел настроек сайта, возможность создания статей, новостей, видео и аудио курсов, управление учётными записями пользователей и другие возможности.

Предполагается, что доступ к административной части будет возможен каждому зарегистрированному пользователю, но набор функций у каждого из них будет различен, в соответствии с его ролью на этом сайте. По умолчанию каждый новый пользователь имеет статус “Пользователь” и доступ в свой кабинет.

Все пользователи, входящие в пользовательскую группу "администратор" по сути должны обладать правами для управления содержимым и для назначения прав доступа к ним.

Рассмотрим конкретные права для каждого из существующих ролей.

В системе предусмотрено разделение пользователей по следующим ролям:

- администратор;
- пользователь;
- посетитель;

Для каждой роли пользователей определены соответствующие ее правам возможности.

«Администратор»:

- добавление, редактирование и удаление страниц на сайте;
- изменение настроек доступа для различных групп пользователей;
- добавление, изменение и удаление новостей сайта;
- добавление, изменение и удаление категорий сайта;
- добавление, редактирование и удаление учетных записей пользователей системы;
- загрузка и удаление информации в систему.

«Пользователь»:

- доступ в личный кабинет;
- создание и редактирование статей для сайта;
- добавление, изменение и удаление новостей сайта;
- добавление, изменение и удаление категорий сайта;

«Посетитель»:

- возможность просмотра содержимого сайта.

3.2.Административная часть как отдельный модуль

Для того ,чтобы грамотно отделить пользовательскую и административную части, мною было принято решение вынести административную часть в виде отдельного модуля.

Модуль — это самодостаточная программная единица, состоящая из моделей, представлений, контроллеров и иных компонентов. Во многом модуль схож с приложением. Основное различие заключается в том, что модуль не может использоваться сам по себе — только в составе приложения. Пользователи могут обращаться к контроллерам внутри модуля абсолютно так же, как и в случае работы с обычными контроллерами приложения.

Модули могут быть полезными в нескольких ситуациях. Если приложение очень объёмное, мы можем разделить его на несколько модулей, разрабатываемых и поддерживаемых по отдельности. Кроме того, некоторый часто используемый функционал, например, управление пользователями, комментариями и пр., может разрабатываться как модули, чтобы впоследствии можно было с лёгкостью воспользоваться им вновь.

Модуль **admin** содержит в себе следующее:

Контроллеры:

1.CategoriesController

2.DefaultController

3.NewsController

4.PageController

5.UserController

Представления делятся на папки:

1.Categories

2.Layouts

3.Default

4.News

5.Page

6.Users

А все модели вынесены за пределы модуля, т.к. ни едины как для административной, так и для пользовательской части. Структура папок модуля админ и код каждого из составляющей модуля представлены в Приложении I [4].

Интерфейс административной части представлен в Приложении I [5].

Модуль Gii.

Начиная с версии 1.1.2, в состав Yii входит веб-инструмент для генерации кода, называемый *Gii*. Он заменяет существовавший до этого консольный генератор yiic shell.

Gii является модулем и должен быть использован в составе существующего приложения Yii. Для использования Gii необходимо отредактировать файл конфигурации приложения. По умолчанию, в целях безопасности, Gii доступен только для localhost. Если необходимо дать доступ к нему с других компьютеров, нужно задать свойство GiiModule::ipFilters .

В составе Gii есть готовый набор генераторов кода. Каждый генератор отвечает за свой тип кода. К примеру, генератор контроллера создаёт класс контроллера вместе с несколькими шаблонами отображения; генератор модели создаёт класс ActiveRecord для определённой таблицы БД.(Приложение 1[6])

3.3. Аутентификация и авторизация

Аутентификация и авторизация необходимы на страницах, доступных лишь некоторым пользователям.

Аутентификация — проверка, является ли некто тем, за кого себя выдаёт. Обычно она подразумевает ввод логина и пароля, но также могут быть

использованы и другие средства, такие как использование смарт-карты, отпечатков пальцев и др.

Авторизация — проверка, может ли аутентифицированный пользователь выполнять определённые действия (их часто обозначают как ресурсы). Чаще всего это определяется проверкой, назначена ли пользователю определённая роль, имеющая доступ к ресурсам.

В Yii встроен удобный фреймворк аутентификации и авторизации (auth), который, в случае необходимости, может быть настроен под ваши задачи.

Центральным компонентом auth-фреймворка является предопределённый компонент приложения «user» — объект, реализующий интерфейс IWebUser. Данный компонент содержит постоянную информацию о текущем пользователе. Мы можем получить к ней доступ из любого места приложения, используя Yii::app()->user.

Используя этот компонент, мы можем проверить, аутентифицирован ли пользователь, используя CWebUser::isGuest. Мы можем произвести вход или выход. Для проверки прав на определённые действия удобно воспользоваться CWebUser::checkAccess. Также есть возможность получить уникальный идентификатор и другие постоянные данные пользователя.

Определение класса Identity

Как было упомянуто ранее, аутентификация — это процесс проверки личности пользователя. Типичное веб-приложение для такой проверки обычно использует логин и пароль. Тем не менее, может потребоваться реализовать проверку другими методами. Чтобы добавить поддержку различных методов аутентификации, в Yii имеется соответствующий identity класс.

Мы реализуем класс identity, который содержит нужную нам логику аутентификации. Такой класс должен реализовать интерфейс IUserIdentity. Для различных подходов к аутентификации могут быть реализованы различные

классы (например, OpenID, LDAP, Twitter OAuth или Facebook Connect). При создании своей реализации необходимо расширить класс CUserIdentity, являющийся базовым классом, который реализует проверку по логину и паролю.

Главная задача при создании класса Identity — реализация метода IUserIdentity::authenticate. Данный метод используется для описания основного алгоритма аутентификации. Также данный класс может содержать дополнительную информацию о пользователе, которая необходима нам в процессе работы с его сессией.

В приведённом примере (Приложение I [7]) мы используем класс identity и покажем, как реализовать аутентификацию по базе данных. Данный подход типичен почти для всех приложений. Пользователь будет вводить логин и пароль в форму. Введённые данные будем проверять с использованием модели ActiveRecord, соответствующей таблице пользователей в БД. В данном примере показано следующее:

Реализация метода authenticate() для проверки данных по БД.

Перекрытие метода CUserIdentity::getId() для возврата _id. По умолчанию в качестве ID возвращается имя пользователя.

Использование метода setState() (CBaseUserIdentity::setState) для хранения информации, необходимой при каждом запросе.

3.4.Разработка пользовательской части сайта

Пользовательская часть сайта состоит из:

Контроллеры:

1.PageController

2.SiteController

3.NewsController

Представления делятся на папки:

1.Layouts

2.News

3.Page

4.Site

Листинг данных контроллеров и представлений изложен в Приложении I [8]

Основной шаблон сайта хранится в файле main.php.

Он, в свою очередь, также разделен на 4 части: header, content, left_sidebar и footer.

Эти блоки формируются на сайте средствами языка программирования php.

Шапка сайта одинаковая для всех страниц сайта, она содержит в себе небольшое изображение, и текст, наиболее полно отражающий идею сайта. В ней также присутствует ссылка на регистрацию для неавторизованного пользователя и ссылка на авторизацию, при наличии логина или пароля.

Для уже зарегистрированного пользователя станет доступной ссылка на вход в личный кабинет.

Блок поиска предназначен для отправки ключевого слова по которому будет, происходит поиск в новостях сайта. Этот запрос отправляется в GET переменной search в файл обработчик search.php.

Описанные выше блоки не меняют своё содержание на всех страницах сайта, при посещении пользователем сайта. В процессе перехода по ссылкам меняется только блок с основным содержанием.

Блок с основным содержанием формируется в файлах, которые непосредственно производят вывод информации в браузер, на всех страницах

сайта, изменяются только файлы: index.php, login.php, articles.php, contact.php, about.php, registration.php.

На главной странице index.php в блоке основного содержания выводится краткое описание 3 последних новостей, с ссылками на них, дата добавления, автор и категория.

На странице авторизации login.php в блоке основного содержания отображается форма с двумя полями, для ввода логина и пароля, и кнопка активирующая эту форму

На странице профиля admin.php в блоке основного содержания отображается имя пользователя, заданное при регистрации. После нажатия на ссылку редактирования регистрационных данных пользователя в блоке основного содержания отображается форма, в которой он может редактировать свои логин, пароль и email.

На странице регистрации registration.php в блоке основного содержания отображается форма, состоящая из 4 полей: пароль, имя, email и поле для ввода кода с картинки. При правильном заполнении этих полей в браузер выводится сообщение об успешной регистрации пользователя.

Поиск по сайту.

Для организации поиска по сайту был дополнительно создан контроллер SiteSearch (Приложение I [9]). Так как основная часть выводимой информации, которая может понадобиться пользователю представлена в модели Page, которая содержит в себе практически всю информацию, отображенную на сайте и модели News, то поиск будет осуществляться по этим двум моделям, а точнее по их содержимому в БД.

Виджеты.

Для вывода меню на главной странице и новостей в левой части сайта, удобнее всего было воспользоваться компонентом - виджетом.

Виджет (widget) — это экземпляр класса CWidget или унаследованного от него. Это компонент, применяемый, в основном, с целью оформления. Виджеты обычно встраиваются в представления для формирования некоторой сложной, но в то же время самостоятельной части пользовательского интерфейса. К примеру, виджет календаря может быть использован для рендеринга сложного интерфейса календаря. Виджеты позволяют повторно использовать код пользовательского интерфейса.

Для подключения виджета необходимо выполнить в коде:

```
<?php $this->beginWidget('path.to.WidgetClass'); ?>
<?php $this->endWidget(); ?>
```

Изменить поведение виджета можно путём установки начальных значений его свойств при вызове CBaseController::beginWidget или CBaseController::widget.

Как и у контроллера, у виджета может быть собственное представление. По умолчанию файлы представлений виджета находятся в поддиректории views директории, содержащей файл класса виджета. Эти представления можно рендерить при помощи вызова CWidget::render() точно так же, как и в случае с контроллером. Единственное отличие состоит в том, что для представления виджета не используются макеты. Также следует отметить, что $this в представлении указывает на экземпляр виджета, а не на экземпляр контроллера.

Заключение.

В рамках работы над дипломным проектом была разработана система для информационного обеспечения дистанционного обучения через просмотр видеолекций, прослушивания аудио, чтение литературы и статей. В разработанной системе предусмотрено разделение пользователей на группы с разным уровнем доступа к элементам системы. Реализованные в разработанной среде возможности позволяют: регистрировать пользователей в системе, размещать новые и изменять имеющиеся материалы в системе, назначать уровень доступа к конкретным материалам.

При разработке сетевой среды использовались следующие инструменты: кроссплатформенная сборка веб-сервера XAMPP версии 1.7.7, в состав которой входит Apache 2.2.21, язык серверных сценариев PHP 5.3.8, СУБД MySQL 5.0.8.

Разработанная система позволяет эффективно управлять дистанционным учебным процессом, обеспечивать очный учебный процесс, а также способствует популяризации изучения языка. Реализация видео- и аудио прослушивания способствует лучшему усвоению нового материала у студентов, воздействует на слуховую и зрительную память, что гораздо эффективней простых текстовых материалов. В разработанной системе пользователю предоставляются лишь те данные, которые администратор считает нужными к просмотру, интерфейс лишен ненужных, отвлекающих элементов, что способствует более полному усвоению материала без отвлечения на второстепенные дела.

В качестве дальнейшего совершенствования web-сайта представляется возможным доработка интерфейса сайта с целью дальнейшего повышения его информативности, привлекательности и удобства. Также стоит решить вопрос с хранением данных. Реализация перечисленных нововведений позволит

улучшит работу с системой и увеличит информационно-содержательную ценность ресурса.

Разработанный сайт удовлетворяет всем требованиям, поставленным на этапе постановки задачи. При разработке web-сайта был использован готовый модуль аутентификации. Данный модуль был доработан с учетом специфики web-сайта и успешно внедрен в его структуру.

Работа сайта на контрольных данных показала ее работоспособность и эффективность.

.

Список использованной литературы.

1. Web Database Application with PHP and MySQL, 2nd Edition By David Lane, Hugh E. Williams. © O'Reilly, May 2004. ISBN: 0-596-00543-1.

2.Сайты: Learn American English Online, Grocery Shopping in English, Essential Grammar in Use, (http://www. english03.ru), (http://learnamericanenglishonline.com), (http://edition.englishclub.com/survival/grocery-shopping/).

3. Yii Framework. http:// ru.wikipedia.org/wiki/Yii

4. Материал из Википедии — свободной энциклопедии о веб-серверах. http://ru.wikipedia.org/wiki/web-server

5. Веб-сервер Apache http://docs.altlinux.org/archive/2.4/master/alt-docs-master/ch06s16.html

6. Модуль http://yiiframework.com/doc/guide/1.1/ru/basics.module

7.Обзор фреймворка yii http://workmake.ru/veb-razrabotka/freymvorki/nachinaem-izuchat-yii/

8.Аутентификация и авторизация http://yiiframework.ru/doc/guide/en/topics.auth

9. Gii генератор кода http://www.yiiframework.com/doc/guide/1.1/ru/topics.gii

10.Виджеты http://yiiframework.com/doc/guide/1.1/ru/ basics.views

11.MVC http://yiiframework.com/doc/guide/1.1/ru/ basics.mvc