WEB-ПРОГРАММИРОВАНИЕ ИНТЕРАКТИВНЫХ ЭЛЕМЕНТОВ ОБРАЗОВАТЕЛЬНОГО САЙТА

Крашенинникова Т.А.,
Волкова Т.И., к. п. н., доцент
г. Бирск, ФГБОУ ВПО Бирский филиал БашГУ

Обучение – одна из наиболее важных форм общественной деятельности. Поиск новых форм и альтернативных решений в обучении начался уже давно. Методики и технологии обучения, инструменты преподавателя, носители обучающей информации изменялись с развитием технических возможностей и ростом объёма знаний.

Современные компьютеры позволяют организовать интерактивное взаимодействие ученика с обучающими материалами. На компьютере можно осуществлять адаптивное, интерактивное обучение. Во-первых, в зависимости от способностей ученика можно адаптировать программу обучения, а, во-вторых, при работе с компьютером ученик находится хотя и в опосредованном, но непрерывном контакте с автором обучающих материалов – учителем. Таким образом, компьютерные технологии позволяют индивидуализировать обучение и создать условия для более эффективного освоения учебного материала.

Ресурсы, которые создаются специально для того, чтобы с их использованием учиться и учить, называются образовательными сайтами. Обучение с использованием образовательных сайтов – это новая ступень, новый уровень использования компьютера в образовании. Объем информации, доступный с помощью Интернет, огромен, а возможности компьютера, подключённого к Интернету, по использованию этой информации в образовательных целях практически безграничны.

Преподаватели высших учебных заведений, школьные учителя самостоятельно, а иногда даже совместно со своими учениками, создают образовательные сайты специально для последующего применения в

учебном процессе. С их использованием могут быть реализованы многие методические замыслы.

На образовательных сайтах используются хранилища электронных документов, статей, основанные на технологии баз данных и предназначенные для хорошо структурированного, упорядоченного хранения больших объёмов учебной информации. Они можут содержать небольшие статьи, посвящённые отдельным учебным темам, а могут содержать интерактивные элементы, такие как автоматизированная система тестирования, тренажеры. Такое приложение обычно имеет средства для быстрого поиска информации по ключевым словам.

Введение интерактивных элементов образовательного сайта в нормативную и практическую составляющую образования позволяет просматривать статьи по разделам, скачивать и просматривать файлы и проходить тестирование для контроля знаний.

При разработке и проектировании образовательного сайта необходимо придерживаться следующих общепринятых требовании и стандартов:

- централизация данных в единои базе;
- рееальное время режима работы;
- сохранение общей модели управления для различных образовательных учреждений;
- поддержка территориально-распределенных структур;
- работа на широком круге аппаратно-программных платформ и СУБД.

Разрабатываемые элементы сайта должны решать следующие задачи:

- Централизованное хранение и удобный доступ материалам и к системе тестирования;

- Удобное взаимодействие преподавателей и пользователей между собой путем легкого обмена материалами;

- Автоматизация тестирования для проведения контроля по

определенным темам.

Одна из главных задач – сделать содержимое страниц сайта доступным для максимального количества пользователей. Расположение информации на страницах сайта и переход между ними должно быть максимально удобным и логичным. А также сайт обязательно должен быть динамическим и содержать интерактивные элементы.

На основе вышеизложенных требований мы разработали следующую структуру создаваемого нами сайта:

Главный раздел сайта - это автоматизированная система тестирования. Система работает с четырьмя типами заданий: закрытого типа, открытого типа, множественный выбор, установление порядка следования. В тесте можно использовать любое количество заданий любых типов, можно только один, можно и все сразу. В заданиях с выбором ответа (множественный выбор, указание порядка) можно использовать до 5 (включительно) вариантов ответа, в заданиях закрытого типа используется 4 варианта ответа.

Следующий раздел - статьи, в котором можно прочитать опубликованный материал и организовать поиск по интересующей рубрике.

Раздел «Файловое хранилище» предназначен для удобного обмена файлами, в нем содержится набор файлов для скачивания.

Далее рассмотрим более подробно реализацию каждого из разделов.

Централизованное хранение данных образовательного сайта.

Для централизованного хранения данных образовательного сайта необходимо использовать реляционную базу данных. Выбор системы управления базами данных является одним из важных этапов при разработке автоматизированной системы тестирования. Выбранный программный продукт должен удовлетворять как текущим, так и будущим потребностям образовательного сайта. Существует множество баз и СУБД, но наиболее распространенной является MySQL в виду ее гибкости, стабильности и быстродействия, а также открытости программного обеспечения, что

является большим плюсом при проектировании программных продуктов для образовательных учреждений.

В БД необходимо хранить статьи, файлы, разделы и вопросы тестирования, данные о пользователе. Для этого необходимо будет создать несколько таблиц с соответствующими связями. Также было бы актуально создать отдельную таблицу для записи всех параметров настроек, которые задает администратор системы. К этим параметрам мы относим логин и пароль главного пользователя (администратора), настройки интерфейса системы, приветственный текст главной страницы и др.

Проектирование интерфейса. Формы, навигация и диалоговые окна.

Структура образовательного сайта и взаимодействия интерактивных элементов приведена на рисунке 1:

Рис.1 Структурная схема образовательно сайта

В программном продукте необходимо логически разделить доступ для администратора, который управляет сайтом (вносит изменения, добавляет, удаляет, редактирует статьи, добавляет и удаляет файлы, разделы и вопросы к тестам) и пользователя (ознакамливается со статьями, загружает файлы, проходит тестирование).

Для централизованной настройки и управления интерактивными элементами образовательного сайта система имеет удобную административную панель.

В системе существует один суперпользователь (admin) у которого нет ограничений на все действия в системе, он может добавлять и удалять любой материал сайта. Для этого созданны специальные формы и окна управления с определенными полями.

Работа в пользовательском интерфейсе осуществляется с помощью рубрикатора меню. Рубрикатор меню пользовательсткого интерфейса содержит следующие пункты:

1) Поиск и чтение статей

2) Поиск и скачивание файлов

3) Прохождение тестирования

Для каждого пункта создадим окно с соответствующими полями и формами.

Средства разработки.

Интерфейс образовательного сайта дает пользователю возможность в удобной форме получать и просматривать информацию. В целом с точки зрения средств реализации система представляет собой веб-приложение, разработанное на языке программирования PHP с использованием популярного фреймворка Bootstrap.

Разработка сложной автоматизированной системы, как правило, предполагает внедрение разнообразных сервисов, разграничения прав доступа к разделам и материалам, большой объем информации и постоянное

развитие проекта в дальнейшем.

Логика разделена через MVC Codeigniter. MVC (Model-View-Controller) предполагает разделение программного кода на 3 основные группы: модели для получения данных из таблиц БД, визуальные шаблоны представления программы, контроллеры для связи пользователя с системой. При написании приложения будем придерживаться этой концепции.

Model-view-controller (MVC, ***«Модель-представление-поведение»***, «Модель- представление-контроллер») — схема использования нескольких шаблонов проектирования, с помощью которых модель данных приложения, пользовательский интерфейс и взаимодействие с пользователем разделены на три отдельных компонента так, что модификация одного из компонентов оказывает минимальное воздействие на остальные. Данная схема проектирования часто используется для построения архитектурного каркаса, когда переходят от теории к реализации в конкретной предметной области.

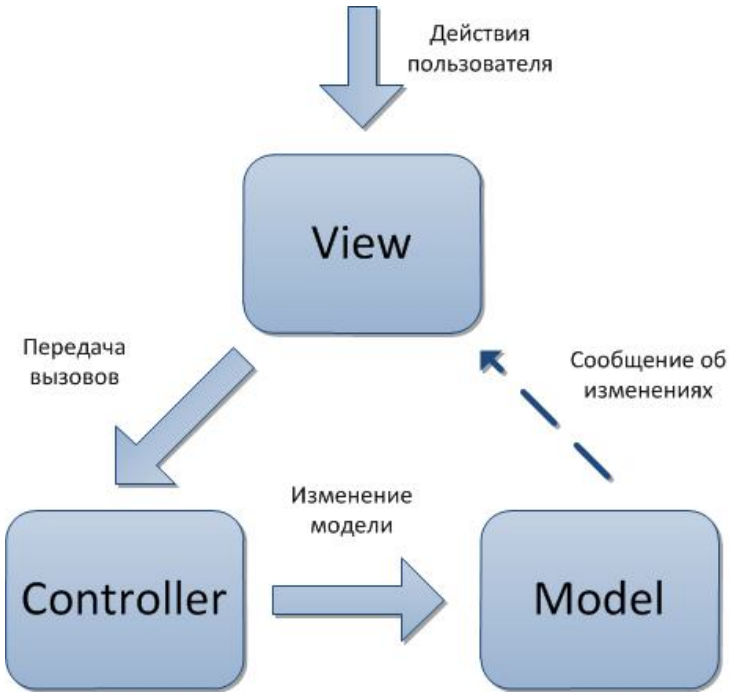

Рис 2 . Концепция Model-View-Controller

Модель (англ. Model). Модель предоставляет знания: данные и методы работы с этими данными, реагирует на запросы, изменяя свое состояние. Не содержит информации, как эти знания можно визуализировать.

Представление, вид (англ. View). Отвечает за отображение

информации (визуализация). Часто в качестве представления выступает форма (окно) с графическими элементами.

Контроллер (англ. Controller). Обеспечивает связь между пользователем и системой: контролирует ввод данных пользователем и использует модель и представление для реализации необходимой реакции.

Так же в проекте выделены модели, контроллеры и представления, которые хранятся в отдельных файлах ***php*** на сервере.

Для каждого интерактивного модуля существует свой контроллер (рис. 3), так же отдельный контроллер для администратора. Запросы броузера поступают в свой контроллер в зависимости от страницы.

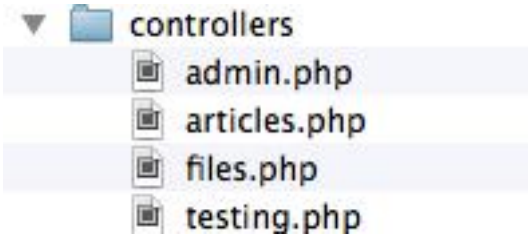

Рис.3 Структура папки controllers проекта

Например, в контроллере *articles* функция *post* берет данные из модели *app_model* (выбор всех статей), а затем отправляет эти данные в представление *subjects*. В модели происходит выборка нужных данных из БД с помощью *SQL-запроса*, а в представлении полученные данные получают «оболочку» с помощью HTML-тэгов. Таким образом, контроллер отдает браузеру пользователя конечный результат в виде готовой HTML- страницы, в которой содержится нужная информация.

В модуле администратора для того, чтобы увидеть результат работы какой-либо функции в контроллере используется технология ***AJAX***, суть которой состоит в подгрузке данных без перезагрузки всей страницы, для каждого управления интерактивным элементом существует свой js файл (Рис. 4) . Эта технология представлена в виде подключаемой библиотеки ***XAJAX***.

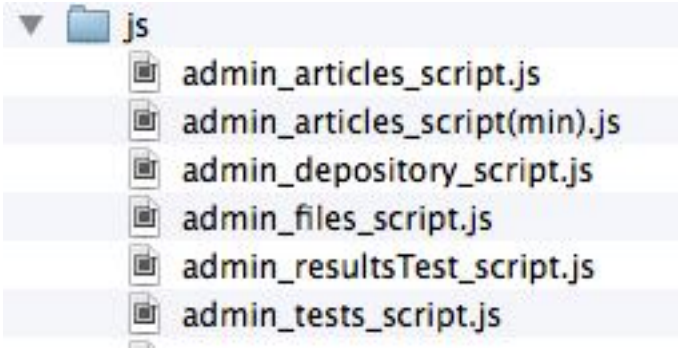

Рис.4 Структура папки js проекта

Пример работы такой системы: пользователь кликает по кнопке, на обработчике которой висит функция XAJAX. Эта функция генерирует запрос к серверу, который направляется на контроллер ***admin***. А контроллер, понимая от какой функции поступил запрос, возвращает тот или иной результат в виде HTML за счет скриптового файла ***admin_articles_script*** и файла представления ***admin_articles_view*** .Таким образом реализована концепция ***MVC***.

Структура БД. Проектирование и обзор таблиц

База данных состоит из 6 таблиц: articles, depository, questions, students, themes и users. *(рис. 5)* 3 таблицы связаны по ключу, 3 не имеют связей.

Таблица articles содержит статьи, категории статей, автора, а так же дату создания. Таблица depository содержит файлы из раздела «Файловое хранилище», эта таблица имеет такие поля, как название, категорию, размер и дату создания файла. В таблицу students записывается информация о пользователе, который проходит тестирование. В таблице user хранятся общие настройки: логин и пароль администратора. Названия тем хранятся в таблице themes, а вопросы - в questions. Каждому предмету соответствуют некоторые темы, которые необходимо выбрать, чтобы получить список вопросов (таблица questions).

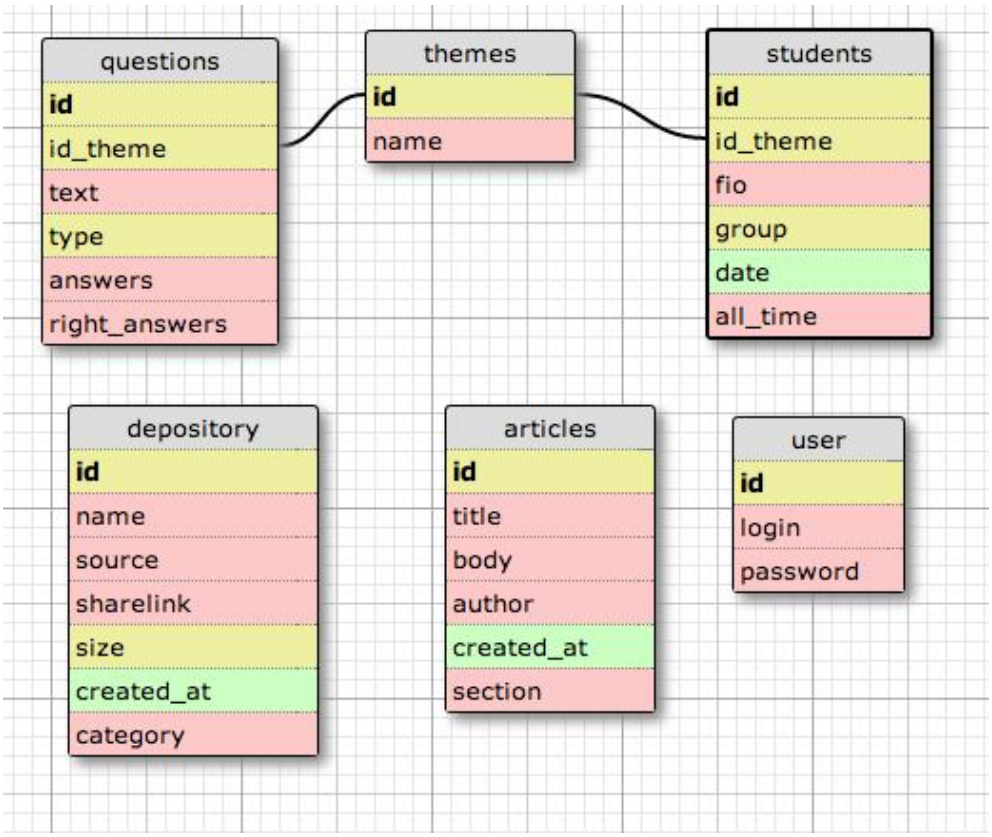

Рис 5. Структура таблиц БД

SQL-запросы к БД. Выборка вопросов из базы

Для работы с элементами сайта были написаны несколько основных функций, содержащие SQL-запросы. Программный код функций располагается в моделях, так как согласно концепции MVC именно они осуществляют SQL-запросы к БД. Рассмотрим их вид и модификации в зависимости от уровня доступа.

У пользователя есть три типа функций:

- Получение/поиск списка статей;
- Получение/поиск файлов и возможность их скачивать;
- Проходить тестирование, при этом внося свои данные в базу данных.

Из панели администратора можно осуществлять следующие функции:

- Получения/добавления/редактирования/удаления статей;
- Получение/добавление/удаление файлов;
- Просмотр/добавление/удаление тем тестирования;
- Добавление/удаление вопросов в раздел тестирования;

- Просмотр результатов тестирования, выгрузка в формат excel.

Функция получения всех статей:

```
public function post($id)
   {
      $data = array();
      $data['post'] = $this->app_model->getPostById($id);
      $this->load->view('user_article_view', $data);
   }
```

Функция содержит SQL-запрос:

```
function getAllPosts()
{
   $res = mysql_query("SELECT * FROM articles ORDER BY created_at DESC");
   $array = array();
   while ($mas = mysql_fetch_array($res))
   {
      $array[] = $mas;
   }
   return $array;
}
```

То есть получаем все записи из таблицы articles. Полученные данные возвращаем как массив данных $array. Данную операцию выполняем каждый раз, когда необходимо получить все статьи.

Функция поиска статьи:

```
function getSearchPost($search='')
   {
      $res = mysql_query("SELECT * FROM articles WHERE (body LIKE '%$search%' or title LIKE '%$search%') ORDER BY created_at DESC");
      $array = array();
      while ($mas = mysql_fetch_array($res))
      {
         $array[] = $mas;
      }
      return $array;
   }
```

Получаем статью из таблицы articles, где сама статья или её название равно переданному параметру, содержащий строку. Полученные данные возвращаем как массив данных $array. Данную операцию выполняем каждый раз, когда осуществляется поиск необходимой статьи.

Получение и поиск файлов осуществляется аналогичным образом из таблицы depository.

Функция прохождения тестирования с возможностью записи своих данных.

```
$fio = $_POST['fio_probros'];
$group = $_POST['group_probros'];
$date = $_POST['today_probros'];
$finish = date("Y-m-d H:i:s");
$d1 = strtotime($date);
$d2 = strtotime($finish);
$diff = $d2 - $d1;
$hours = floor($diff / 3600);
$min = floor($minutes = ($diff / 3600 - $hours) * 60);
 $seconds = ceil(($minutes - floor($minutes)) * 60);
$all_time2 = $hours . ":0" . $min . ":" . $seconds;
mysql_query("INSERT INTO `students`(`id_theme`, `fio`, `group`, `date`, `all_time`) VALUES ('$id_theme', '$fio','$group', '$finish', '$all_time2')");
```

Эта функция получает данные пользователя и записывает их в базу данных, так же производится получения время начала и конца тестирования и перевод результатов тестирования в процентное соотношение.

```
function getQuestionsByIdTheme($id)
   {
      $res = mysql_query("SELECT * FROM questions WHERE id_theme='$id' ");
      $array = array();
      while ($mas = mysql_fetch_array($res))
      {
         $array[] = $mas;
      }
      return $array;
   }
```

Эта функция отвечает за вывод вопросов в зависимости от темы.

Функция добавления статьи.

```
public function articles()
   {
      $user = new OAuth();

      if ($user->is_auth())
      {
```

```
if (isset($_GET['saveArticle']))
        {
          $id = $_REQUEST['id'];
          $section = $_REQUEST['section'];
          $title = $_REQUEST['title'];
          $author = $_REQUEST['author'];
          $created_at = date('Y-m-d H:i:s',strtotime($_REQUEST['date']));
          $body = trim($_REQUEST['body']);
          if (empty ($id))
          {
            mysql_query("INSERT INTO articles (title, body, author,
section, created_at) VALUES ('$title','$body','$author', '$section', '$created_at')");
          }
          else
          {
            mysql_query("UPDATE articles SET
          title = '$title',
          body ='$body',
          author = '$author',
          section = '$section',
          created_at = '$created_at'
          WHERE id='{$id}'
          ");
          }
          $array = array();
          $array['status'] = 'ok';
          echo json_encode($array);
          exit();
        }
        $data = array();
        $data['login_name'] = $user->getLogin();
        $this->load->view('adm_articles_view', $data);
} else {
        echo 'error: access denied';
    }
  }
```

Проверяем авторизацию, затем получаем данные из admin_articles_view и запросом сохраняем в базе данных.

Аналогичным образом происходит загрузка файлов.

Функция просмотра результатов тестирования, выгрузка в формат excel.

```
  public function get_results($get_group)
```

```
        {
            if ($get_group === 'all_group')
            {
                $sql_group = " ";
            }
            else
            {
                $sql_group = " AND students.group = $get_group ";
            }
            $res = mysql_query("SELECT students.*, themes.name AS theme_name FROM students JOIN themes ON themes.id = students.id_theme WHERE 1=1 " . $sql_group . " ORDER BY date DESC");
            $array = array();
            while ($mas = mysql_fetch_array($res))
            {
                $array[] = $mas;
            }
            return $array;
        }
```

Приведенная выше функция имеет sql запрос, формирующий массив с данными о результате тестирования, это осуществляется за счет слияния 2-х таблиц students и themes.

```
        public function print_result()
        {
            header('Content-Encoding: UTF-8');
            header('Content-type: text/csv; charset=UTF-8');
            header('Content-disposition: attachment;filename=Group-' . $_POST['group'] . '.csv');
            echo "\xEF\xBB\xBF"; // UTF-8 BOM
            foreach ($this->get_results($_POST['group']) as $data)
            {
                echo $data['fio'] . ";"  . $data['theme_name'] . "\n";
            }
            exit();
        }
```

Функция print_result отвечает за печать отчета по тестирвоанию в формат excel.

Развитие образовательных интернет-ресурсов происходит непрерывно: появляются новые образовательные сайты, порталы, базы

знаний, системы дистанционного обучения, разрабатываются новые формы представления обучающего материала и методики его использования.

Интерфейс интерактивных элементов.

Для создания внешнего интерфейса образовательного сайта применяются такие языки веб-программирования, как CSS, язык гипертекстовой разметки HTML.

За основу оформления был выбран простой и легко настраиваемый HTML, CSS и Javascript фреймворк Bootstrap.

При входе на сайт попадаем на стартовую страницу на которой имеются ссылки на интерактивные элементы (Рис.6).

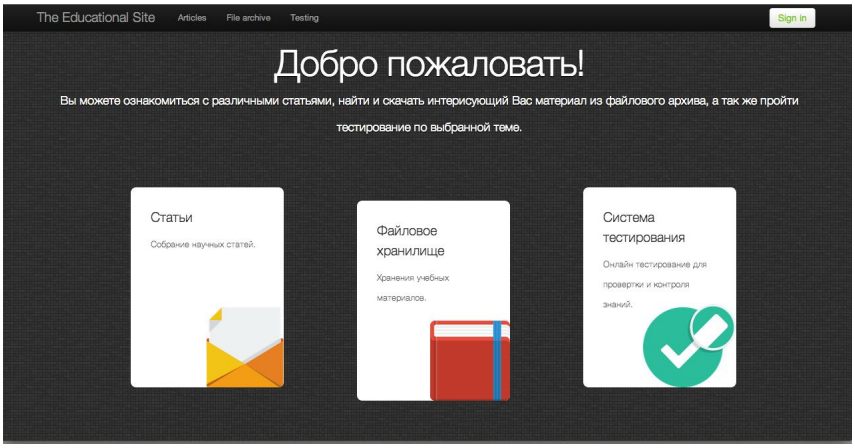

Рис. 6 Стартовое окно образовательного сайта.

Посетителям сайта доступны такие страницы как (рис.7):

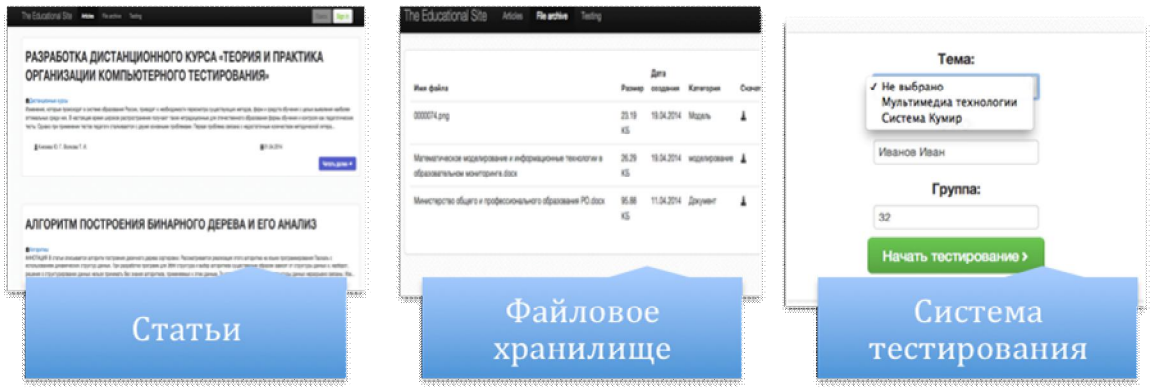

Рис. 7 Интерфейс интерактивных элементов

Тестирование осуществляется в несколько этапов (Рис. 8):

1. Заполнение данных и выбор темы тестирования

2. Прохождение тестирования
3. Получение результатов

Рис. 8 Интерфейс этапов прохождения тестирования

Обзор панели администрирования

Панель администратратора имеет страницу с боковым меню, состоящая из ***4 основных разделов***:

Управление статьями – раздел предназначен для добавления, редактирования и удаления статей.

Управление файлами – позволяет загружать, удалять файлы.

Тестирование – основной раздел для добавления, удаления разделов и вопросов тестирования (Рис. 9).

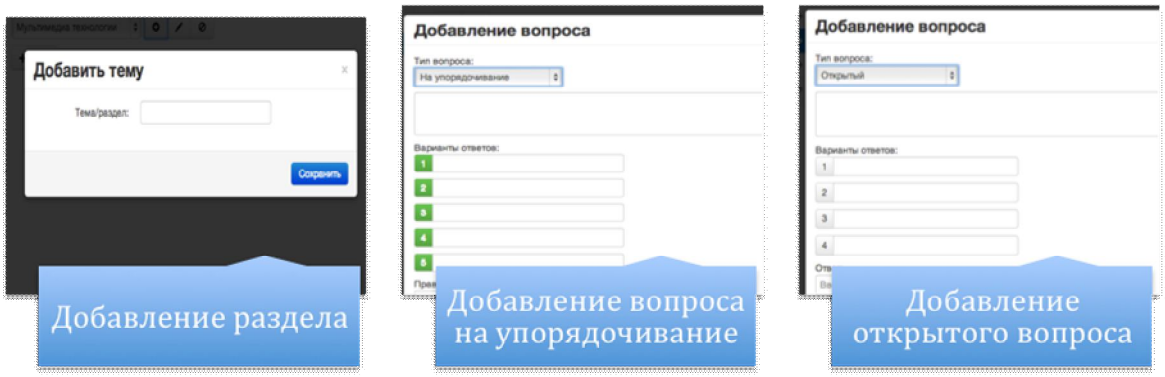

Рис. 9 Интерфейс страницы администратора для управления тестированием

Результаты тестирования – в этом разделе администратор может просматривать результаты тестирования, а так же может загрузить их в формат excel на локальный компьютер.

Разработанный образовательный сайт функционирует в сети Интернет и проходит опытно-экспериментальную проверку. Пользователи проходят тестирование, ознакамливаются со статьями и загружают необходимые файлы. Программный продукт позволяет работать как локально на одном ПК, так и в многопользовательском режиме по локальной сети. В дальнейшем планируется использовать его в масштабе вуза для централизованного хранения информации и осуществления контроля знаний.