Федеральное агенство по образованию
Государственное образовательное учреждение высшего профессионального образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Кафедра «Информационная безопасность систем и технологий»

РАЗРАБОТКА WEB-ПРИЛОЖЕНИЙ НА ЯЗЫКЕ PHP

Пояснительная записка к курсовому проекту
по дисциплине «Информационные технологии»

ПГУ 3.090106.001 ПЗ

Руководитель КР,
к.т.н. _____ Н. А. Егорова

Исполнитель КР,
студент _____ М. А. Захаров

г. Пенза, 2009

Утверждаю
Зав. кафедрой ИБСТ
_____С. Л. Зефиров
«____» _____ 2009 г.

Задание

на курсовое проектирование

по теме: «Разработка web-приложений на языке PHP»

1 Дисциплина Информационные технологии

2 Вариант задания 35

3 Студент Захаров М.А. группа 06УИ1

4 Исходные данные на проектирование

4.1 Требования к сайту:

- тема сайта — электронная цифровая подпись;
- сайт должен содержать не менее пяти страниц;
- информация на сайте должна быть структурирована в виде учебного материала по отдельным темам;
- должна быть реализована процедура тестирования обучаемого;
- число категорий пользователей по правам доступа — 3;
- для всех пользователей доступ к сайту разрешен с правом «просмотр»;
- для администратора доступ к сайту разрешен с правом «супервизор».

4.2 Функциональные требования к программе:

- хранение информации наполнения web-сайта в базе данных MySQL;
- возможность добавления информации наполнения web-сайта;
- возможность редактирования информации наполнения web-сайта;
- возможность выборочного удаления информации наполнения web-сайта;
- языки программирования — HTML, PHP;
- СУБД MySQL.

5 Структура проекта

5.1 Пояснительная записка (содержание работы):

- описание технологии создания web-сайта;

- схема расположения информации на сайте;
- схема хранения информации в базе данных;
- описание категорий пользователей и их прав доступа к сайту;
- разработка алгоритма;
- разработка программы;
- руководство администратора и пользователя сайта.

5.2 Графическая часть

- не предусмотрена.

5.3 Экспериментальная часть

- не предусмотрена.

6 Календарный план выполнения проекта

6.1 Сроки выполнения работ по разделам:

- разработка схемы расположения информации на сайте к 01.03.2009 г.
- разработка алгоритма к 29.03.2009 г.
- разработка и тестирование сайта к 01.05.2009 г.
- оформление ПЗ и подготовка к защите к 17.05.2009 г.

Дата защиты проекта 22 мая 2009 г.

Руководитель работы Егорова Н. А.

Задание получил 10 февраля 2009 г.

Студент Захаров М. А.

Нормоконтролёр Егорова Н. А.

Реферат

Отчёт содержит 36 с, 10 рисунков, 8 источников.

PHP, MYSQL, CSS, JAVASCRIPT, ДИНАМИЧЕСКИЙ ВЕБ-САЙТ

Объектом исследования является технология создания динамических веб-сайтов с использованием языка программирования PHP и СУБД MySQL.

Цель работы — создание динамического веб-сайта, наполнение которого хранится в БД MySQL, с возможностью добавления, редактирования и выборочного удаления содержимого; реализовать процеду тестирования посетителей сайта, создать три категории пользователей.

Результатом выполнения курсовой работы является динамический веб-сайт, использующий технологии PHP, MySQL, JavaScript, CSS.

Содержание

Введение

В последнее время благодаря бурному развитию Интернета в программировании выделяют отдельное направление — *Web-программирование*, т. е. создание сценариев для Web. Поначалу оно не могло и сравниться по своей сложности с другими областями программистского ремесла, не «дотягиваясь» не только до системного, но даже и до прикладного уровня. В наши дни, однако, роль Web-программирования в структуре глобальной сети возрастает, соответственно увеличивается и средняя оценка сложности сценариев. Многие системы (например, поисковые) по объёму кода приближаются к размеру исходных кодов серьёзных прикладных программ. Доля же статических страниц в Web постоянно падает; на смену им приходят *динамические* страницы, сгенерированные автоматически тем или иным сценарием. [1]

Сегодня разработано огромное количество средств, интернет-решений, которые предоставляют самые широкие возможности для создания абсолютно любых сайтов.

Конечно, можно воспользоваться готовыми шаблонами, в этом случае программирование не понадобится. Однако ценность такого сайта, мягко говоря, будет невелика. Да и удобство обращения с ним, «технологичность», интерактивность портала будет желать лучшего.

Web-программирование, в настоящее время, позволяет создавать уникальные, удобные и функциональные сайты. Основными средствами программирования, на сегодняшний день, являются PHP и MySQL, однако могут применяться и другие решения, такие как HTML, DHTML, JavaScript, XML/XSL, Java, Flash, Perl.

Однако именно разнообразие инструментов, ставит Web-программирование «во главу угла» при создании любого сайта. Именно программирование позволяет реализовывать любые проекты, значительно разнообразить сайт, автоматизировать и облегчить работу с Интернет порталом.

В частности, Web-программирование предлагает следующие, наиболее популярные решения:

а) упрощение обновления информации на сайте;

б) разнообразные интерактивные элементы, которые можно добавлять на сайт. Среди них наиболее популярными являются форумы, гостевые книги, возможность отправки e-mail с сайта, и многое другое.

в) хранение большого количества данных в специальных базах. Прайсы, отзывы посетителей, описание товаров, фотоальбомы, статистические данные — это далеко не полный перечень возможностей, которые даёт Web-программирование в данном конкретном случае;

г) использование сайта для интерактивной связи с, например, торговыми представителями в любой точке мира;

Программирование позволяет сделать сайт более интерактивным. Добавление разнообразных динамических разделов позволит повысить его информативность, и, как следствие, популярность ресурса. Иными словами, оно используется для решения абсолютно любых задач, какими бы сложными или необычными они не были.

1 Обзор проекта

1.1 Инструментальные средства

Apache HTTP-сервер — свободный веб-сервер. С апреля 1996 и до настоящего времени является самым популярным HTTP-сервером в Интернете. Основными достоинствами Apache считаются надёжность и гибкость конфигурации. Он позволяет подключать внешние модули для предоставления данных, использовать СУБД для аутентификации пользователей, модифицировать сообщения об ошибках и т. д. [2]

PHP — язык сценариев, встраиваемый в HTML на стороне сервера, предназначенный для создания динамических веб-страниц и принадлежащий к категории продуктов с открытым исходным кодом. Не налагая ограничений на броузеры, он предоставляет простое и универсальное, переносимое между платформами решение для электронной коммерции, сложных веб-приложений, включая управляемые базами данных.

MySQL — свободная система управления базами данных (СУБД). Существует множество пакетов СУБД. Эти программы обладают различными возможностями, гибкостью и ценой. Однако все они работают приблизительно одинаково. В данной курсовой работе используются базы данных MySQL. Эта программа часто используется вместе с PHP по нескольким причинам. Во-первых, MySQL обладает очень широкими возможностями. Она обладает большинством возможностей, предоставляемых в дорогих и мощных пакетах для работы в базами данных. Базы данных MySQL используют стандартную форму широко известного языка SQL. MySQL выпускается с открытым кодом и распространяется бесплатно, работает с большинством операционных систем и может использоваться со многими языками программирования. Она работает очень быстро даже с большими наборами данных. Кроме того, в PHP встроено множество функций для поддержки баз данных MySQL [3].

1.2 Преимущества динамического Web-сайта

Каждая отображаемая страница динамических Web-сайтов основана на шаблонной странице, в которую вставляется постоянно меняющееся информаци-

онное наполнение, которое обычно хранится в базе данных. Когда пользователь запрашивает страницу, соответствующая информация извлекается из базы, вставляется в шаблон, образуя новую Web-страницу, и пересылается Web-сервером в пользовательский браузер, который и отображает её должным образом. Кроме информационного наполнения, динамически могут создаваться также и элементы навигации по Web-сайту. Таким образом, если вам нужно обновить содержимое своего сайта, вы просто добавляете текст для новой страницы, который затем вставляется в базу данных с помощью определённого механизма. В результате получается, что Web-сайт как бы сам себя обновляет.

Сразу после того как динамический сайт создан и запущен в работу, начинают проявляться его преимущества. Теперь в вашем распоряжении имеется сравнительно небольшое количество шаблонных страниц, с помощью которых генерируются сотни, а может быть, и тысячи Web-страниц. Вид (дизайн) сайта может быть легко изменён с помощью модификации этих шаблонов. Изменение содержимого базы данных можно производить через Web-интерфейс с использованием HTML-формы, не вторгаясь при этом в технические детали каждой специфической СУБД [4].

1.3 Как работает динамический Web-сайт

Основная операция Web-сервера проиллюстрирована на рисунке 1.1.

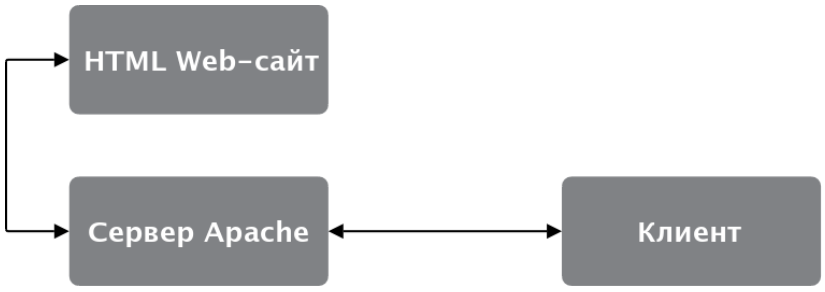

Рисунок 1.1 — Отношение типа клиент/сервер

В данном примере клиент соединяется с Web-сервером (в нашем случае, сервер Apache) и запрашивает страницу HTML. Когда клиент соединился, сервер Apache возвращает требуемую страницу клиенту. В этом примере сервер Apache действует как посредник между клиентом и сервером.

Для сервера, отсылающего обычные статические страницы, такая архитектура подходит. Архитектура же сайта, который включает в себя базу данных, несколько сложнее [5]. Приложения Web-баз данных, которые будут разрабатываться в этой курсовой работе, наследуют глобальную структуру Web-баз данных, показанную на рисунке 1.2.

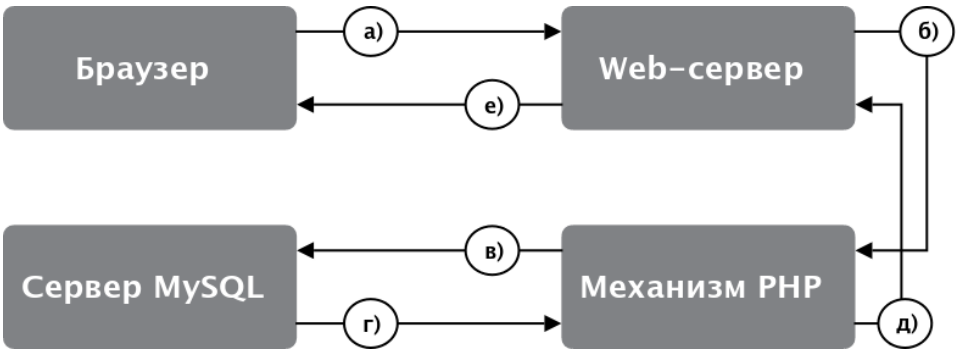

Рисунок 1.2 — Базовая архитектура Web-баз данных

Типичная транзакция Web-базы данных состоит из этапов, обозначенных буквами на рисунке 1.2.

а) Web-браузер пользователя отправляет HTTP-запрос определенной Web-страницы.

б) Web-сервер принимает запрос и передает его механизму PHP на обработку.

в) Механизм PHP начинает синтаксический анализ сценария. В сценарии присутствует команда подключения к базе данных и выполнения запроса в ней. PHP открывает соединение с сервером MySQL и отправляет необходимый запрос.

г) Сервер MySQL принимает запрос в базу данных, обрабатывает его, а затем отправляет результаты обратно в механизм PHP.

д) Механизм PHP завершает выполнение сценария, форматируя результаты запроса в виде HTML, после чего отправляет результаты в HTML-формате Web-серверу.

е) Web-сервер пересылает HTML в браузер, с помощью которого пользователь просматривает страницу.

2 Реализация проекта

2.1 Создание базы данных

Общая схема хранения данных в базе данных для данного проекта представлена на рисунке 2.1.

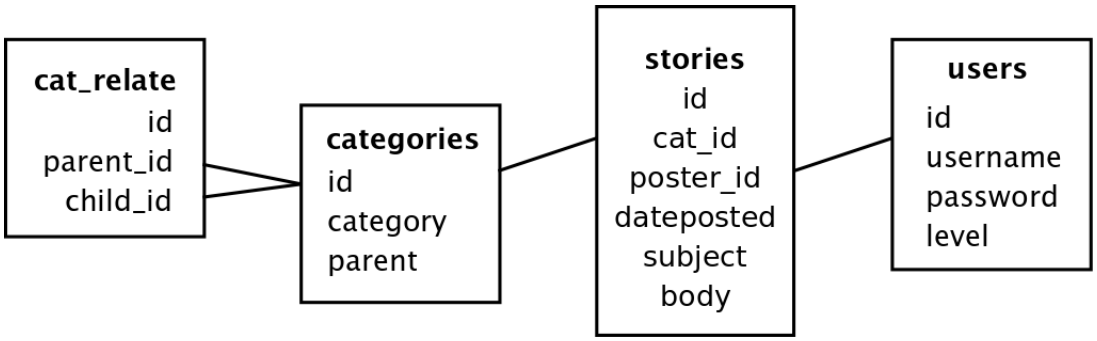

Рисунок 2.1 — База данных

База данных содержит четыре таблицы:

а) `cat_relate`. Эта таблица необходима для того, чтобы однозначно определить к какому из разделов относится подраздел;

б) `categories`. В данной таблице содержатся названия разделов и подразделов, отображаемые в меню навигации (поле «`parent`» указывает на уровень раздела);

в) `stories`. В таблице хранятся все статьи сайта;

г) `users`. Таблица содержит список зарегистрированных пользователей сайта.

Для сайта реализовано три категории пользователей:

а) администратор — способен добавлять новые статьи, удалять содержимое сайта, добавлять или удалять разделы и подразделы;

б) зарегистрированный пользователь — имеет право добавлять новые статьи

в) незарегистрированный пользователь — способен лишь просматривать содержимое сайта.

В данном проекте реализована возможность хранения паролей в зашифрованном виде с помощью алгоритма хеширования MD5 [6]. Каждый пользователь имеет свой пароль и его знает только пользователь. Получить настоящий пароль можно только полным перебором. Результат преобразования представляется как

последовательность из 32 шестнадцатеричных чисел. К примеру, хэш пароля администратора сайта — `21232f297a57a5a743894a0e4a801fc3`.

2.2 Построение основной части системы

2.2.1 Автоматизация подключения к СУБД

Обычно на сайте существует сразу несколько скриптов, которым нужен доступ к одной и той же базе данных. Соответственно, код, ответственный за подключение к СУБД, лучше всего выделить в отдельный файл, который потом будет подключаться ко всем программам. В листинге 2.1 приведён php-файл, осуществляющий подключение к базе данных, а в листинге 2.2 определены переменные, необходимые для подключения.

Листинг 2.1 — db.php

```php
<?php
require("config.php");
$db = mysql_connect($dbhost, $dbuser, $dbpassword);
mysql_query('SET NAMES utf8');
mysql_select_db($dbdatabase, $db);
?>
```

Листинг 2.2 — config.php

```php
<?php
$dbhost = "localhost";
$dbuser = "root";
$dbpassword = "maxim";
$dbdatabase = "dssbase";
$config_basedir = "http://localhost/dss/";
?>
```

2.2.2 Создание вспомогательных функций

Для данного проекта были созданы две вспомогательные фунцкции:

а) pf_fix_slashes();

б) pf_check_number.

В PHP существует такое понятие, как «магические кавычки». Магические кавычки — эффект автоматической замены кавычки на обратный слэш и кавычку при операциях ввода/вывода в PHP. Первая функция реализует экранирование символов в случае, если `magic_quotes_gpc` отключена.

Листинг 2.3 — functions.php

```
1   <?php
2  function pf_fix_slashes($string) {
3     if (get_magic_quotes_gpc() == 1) {
4        return($string);
5     }
6     else {
7        return(addslashes($string));
8     }
9  }
10 function pf_check_number($value) {
11    if(isset($value) == FALSE) {
12       $error = 1;
13    }
14    if(is_numeric($value) == FALSE) {
15       $error = 1;
16    }
17    if($error == 1) {
18       return FALSE;
19    }
20    else {
21       return TRUE;
22    }
23 }
24 ?>
```

2.2.3 Интерфейсная часть

Сценарии, приведённый в листингах 2.4, 2.5 реализуют шаблонную часть страницы. Они отображаюся в верхней и нижней части каждой страницы. В результате любой вывод, генерируемый сценарием, отображается в ячейке основного содержимого страницы. В дальнейшем два этих сценария будут подключатся к каждой странице директивой `require`.

Листинг 2.4 — header.php

```
<?php
  session_start();
  require("config.php");
?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
 "http://www.w3.org/TR/html4/loose.dtd">
<head>
  <title><?php echo $config_sitename; ?></title>
  <link href="stylesheet.css" rel="stylesheet">
  <meta http-equiv="content-type" content="nt-type" ; charset=UTF-8" >
</head>
<body>
  <div id="header">
  <h1>Цифровая подпись</h1>
  </div>
  <div id="menu">
    <a href="<?php echo $config_basedir; ?>">На главную</a>
  </div>
  <div id="container">
    <div id="bar">
      <?php
        require("bar.php");
      ?>
    </div>
    <div id="main">
```

Листинг 2.5 — footer.php

```
    </div>
</div>
</body>
</html>
```

2.2.4 Создание иерархического меню

Меню содержит список разделов и подразделов сайта. Пользователь щелкнул на одном из разделов сайта. Далее поворот событий зависит от того, есть ли у раздела подразделы. Если в разделее есть подразделы, то будет выведен спи-

сок подразделов (меню станет иерархическим). Внешний вид меню представен на рисунке 2.2.

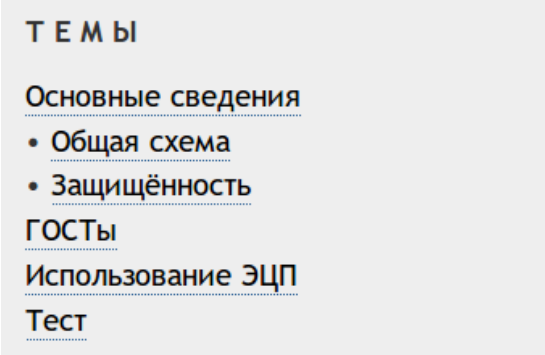

Рисунок 2.2 — Меню сайта

Код сценария `bar.php` приводится в листинге 2.6.

Листинг 2.6 — bar.php

```
1   <?php
2 session_start();
3 require("db.php");
4
5 echo "<table class='visible' width='100%'cellspacing=0 cellpadding=5>";
6 echo "<tr><th class='visible'>Информация</th></tr>";
7 echo "<tr><td>";
8 if($_SESSION['SESS_USERNAME']) {
9     echo "Вы вошли как <strong>" . $_SESSION['SESS_USERNAME']
10  . "</strong> - <a href='userlogout.php'>Выйти</a>";
11    echo "<p>";
12    if($_SESSION['SESS_USERLEVEL'] > 0) {
13        echo "<a href='addstory.php'>Написать статью</a><br />";
14    }
15    if($_SESSION['SESS_USERLEVEL'] == 10) {
16        echo "<a href='addcat.php'>Новый раздел</a><br />";
17    }
18    echo "<p>";
19 }
20 else {
21    echo "<a href='userlogin.php'>Войти</a>";
22 }
23 echo "</td></tr>";
24 echo "</table>";
```

```
25
26 echo "<h1>Темы</h1>";
27 $sql = "SELECT * FROM categories WHERE parent = 1;";
28 $result = mysql_query($sql);
29 $numrows = mysql_num_rows($result);
30 if($numrows == 0) {
31    echo "<p>No categories</p>";
32 }
33
34 else {
35    while($row = mysql_fetch_assoc($result)) {
36       if($_SESSION['SESS_USERLEVEL'] == 10) {
37          echo "<a href='deletecat.php?id=" . $row['id'] . "'>[X]</a> ";
38       }
39       echo "<a href='index.php?parentcat=" . $row['id'] . "'>"
40  . $row['category'] . "</a><br>";
41
42      if($row['id'] == $_SESSION['SESS_PARENT']) {
43         $childsql = "SELECT categories.id, categories.category
44 FROM categories INNER JOIN cat_relate
45 ON categories.id = cat_relate.child_id
46 WHERE cat_relate.parent_id = " . $_SESSION['SESS_PARENT'] . ";";
47         $childresult = mysql_query($childsql);
48
49 while($childrow = mysql_fetch_assoc($childresult)) {
50            if($_SESSION['SESS_USERLEVEL'] == 10) {
51               echo "<a href='deletecat.php?id=" . $childrow['id']
52 . "'>[X]</a> ";
53            }
54            echo " &bull; <a href='index.php?parentcat=" . $row['id']
55 . "&amp;childcat=" . $childrow['id'] . "'>" . $childrow['category']
56 . "</a><br>";
57         }
58      }
59   }
60 }
61 ?>
```

В данной программе используются *сессии*. Они представляют собой механизм, позволяющий хранить некоторые (и произвольные) данные, индивидуаль-

ные для каждого пользователя, между запусками сценария. На странице анализируются идентификаторы пользователей — переменные, хранящиеся в сессии.

Если у авторизованного пользователя идентификатор `10` (администратор ресурса), то слева от категорий отображается ссылка на удаление раздела.

На рисунке 2.3 показан вид меню для аминистратора.

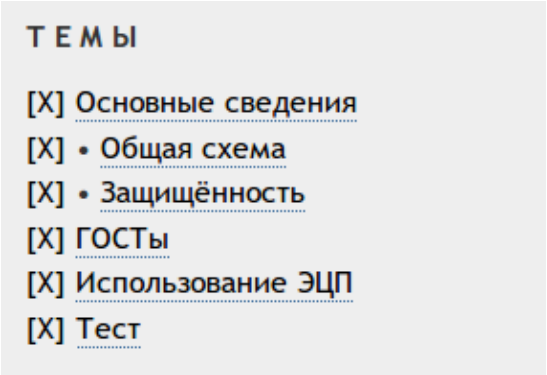

Рисунок 2.3 — Вид меню для администратора

Для повышения удобства пользованием сайта была создана информационная панель (см. рисунок 2.4), которая отображает информацию о пользователе и о списке его привелегий. Внешний вид панели зависит от значения переменной `SESS_USERLEVEL`. Зарегистрированные пользователи могут добавлять новые статьи, администратор способен также добавлять новые разделы и подразделы.

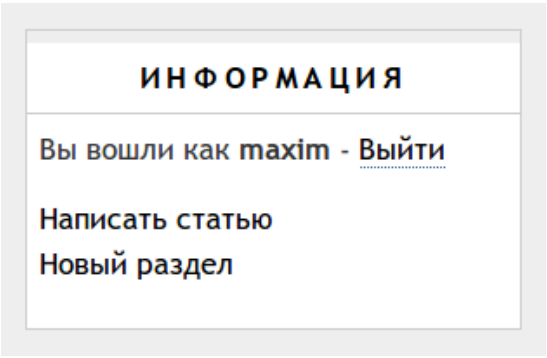

Рисунок 2.4 — Информационная панель

2.2.5 Главная страница сайта

Главной страницей сайта будет программа `index.php`, и она выполняет три основные цели:

а) установливает переменную сессии для раздела и подраздела, выбранных пользователем;

б) при выборе раздела или подраздела выводит список статей в данной категории;

в) если не выбрана ни одна из категорий, выводит на страницу 5 последних статей.

Переменные метода `GET`, которые передаются на странице, определяют функциональность страницы. Если переменные `GET` отсутствуют, на странице отображаются последние статьи из всех имеющихся категорий. Если странице передаётся переменная `parentcat`, то будут выводиться статьи в этой категории. Если передаются обе переменные `parentcat` и `childcat`, то отображаются статьи для текущего подраздела.

Переменная `$currentcat` определяет текущую категорию. У неё три возможных состояния:

а) Если существует только переменная `parentcat`, то `currentcat` присваивается её значение.

б) Если существует переменная `childcat`, то `currentcat` присваивается её значение.

в) Если отсутствуют обе переменные, значение `currentcat` устанавливается в `0`.

Внешний вид главной страницы показан на рисунке 2.5.

Листинг 2.7 — index.php

```
1   <?php
2 require("db.php");
3 session_register("SESS_PARENT");
4 session_register("SESS_CHILD");
5 if(isset($_GET['parentcat']) && isset($_GET['childcat'])) {
6    if(is_numeric($_GET['parentcat'])) {
7       $_SESSION['SESS_PARENT'] = $_GET['parentcat'];
8    }
9    if(is_numeric($_GET['childcat'])) {
10       $currentcat = $_GET['childcat'];
11       $_SESSION['SESS_CHILD'] = $_GET['childcat'];
12    }
```

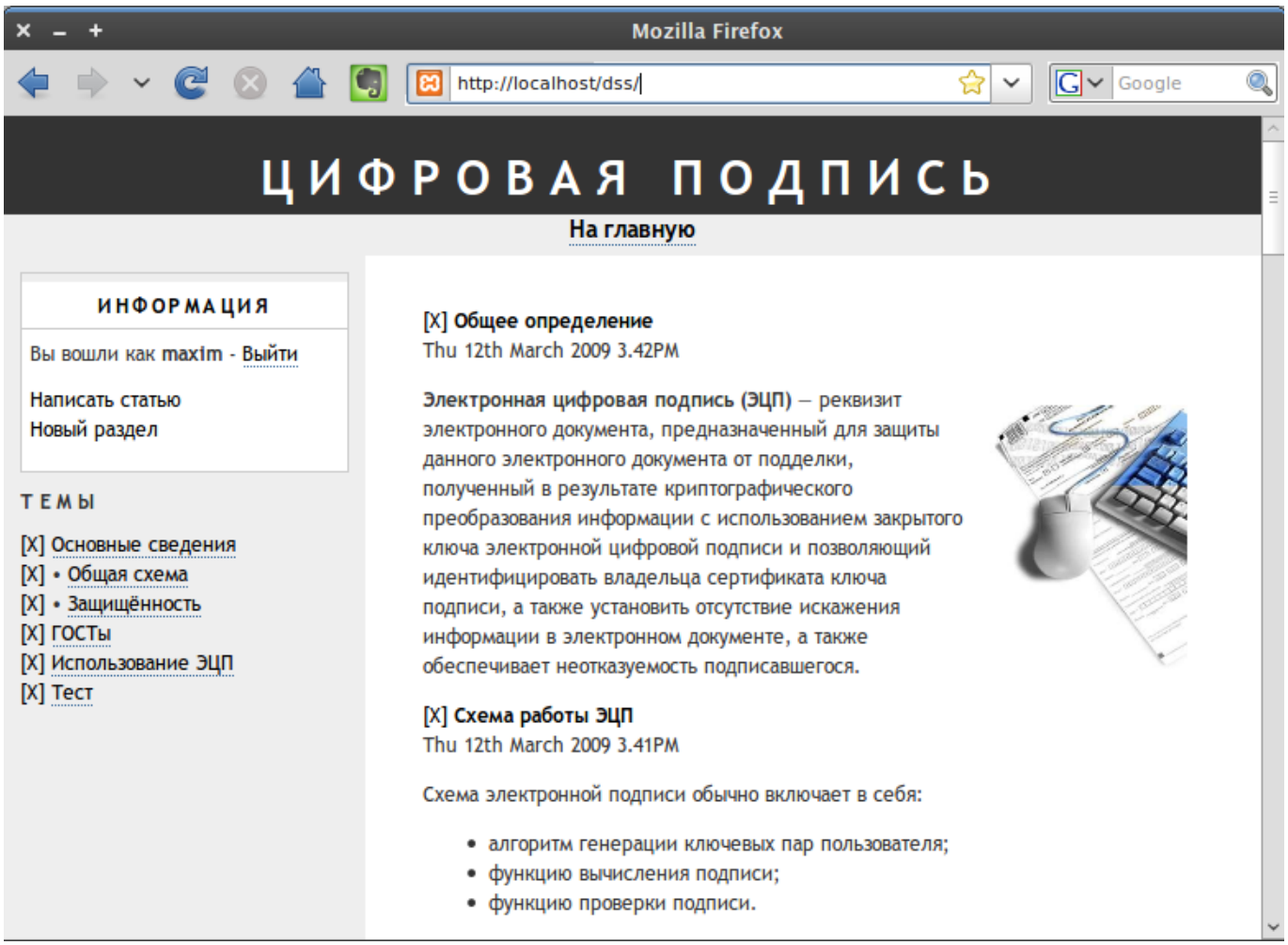

Рисунок 2.5 — Главная страница сайта

```
13 }
14 else if(isset($_GET['parentcat'])) {
15     if(is_numeric($_GET['parentcat'])) {
16         $currentcat = $_GET['parentcat'];
17         $_SESSION['SESS_PARENT'] = $_GET['parentcat'];
18         $_SESSION['SESS_CHILD'] = 0;
19     }
20 }
21 else {
22     $currentcat = 0;
23 }
24
25 require("header.php");
26 if($currentcat == 0) {
27     $sql = "SELECT * FROM stories ORDER BY dateposted DESC LIMIT 5;";
28 }
```

```
29
30 else {
31    $parentsql = "SELECT parent FROM categories WHERE id = "
32  . $currentcat . ";";
33    $parentres = mysql_query($parentsql);
34    $parentrow = mysql_fetch_assoc($parentres);
35
36    if($parentrow['parent'] == 1) {
37       $sql = sprintf("SELECT stories.* FROM stories INNER JOIN
38  cat_relate ON stories.cat_id = cat_relate.child_id WHERE
39  cat_relate.parent_id = %d UNION SELECT stories.* FROM stories WHERE
40  stories.cat_id = %d;" , $currentcat, $currentcat);
41    }
42    else {
43       $sql = "SELECT * FROM stories WHERE cat_id = " . $currentcat .
44 ";";
45    }
46 }
47
48 $result = mysql_query($sql);
49 $numrows = mysql_num_rows($result);
50
51 if($numrows == 0) {
52    echo "<h1>No Stories</h1>";
53    echo "<p>There are currently no stories in this category.   </p>";
54 }
55 else {
56    while($row = mysql_fetch_assoc($result)) {
57       if($_SESSION['SESS_USERLEVEL'] == 10) {
58          echo "<a href='deletestory.php?id=" . $row['id']
59  . "'>[X]</a> ";
60       }
61       echo "<strong><a href='viewstory.php?id=" . $row['id']
62          . "'>"
63          . $row['subject']
64          . "</a></strong><br />";
65       echo date("D jS F Y g.iA", strtotime($row['dateposted']));
66       echo "<p>" . $row['body'] . "</p>";
67    }
68 }
```

```
69
70 require("footer.php");
71 ?>
```

2.3 Прикладная часть

2.3.1 Авторизация пользователей

Следующий шаг — это создание страницы для входа на сайт. Эта страница будет содержать форму для ввода имени пользователя и пароля, а также сценарии PHP, выполняющие поиск введенных имени и пароля в таблице списка пользователей. Если такой пользователь есть в списке, ему будут открыты все страницы сайта, в противном случае он получит только права гостя (обычного посетителя).

Листинг 2.8 — userlogin.php

```
1  <?php
2  session_start();
3  require("config.php");
4  require("db.php");
5  require("functions.php");
6  if($_SESSION['SESS_USERNAME']) {
7     header("Location: " . $config_basedir . "userhome.php");
8  }
9
10 if($_POST['submit']) {
11    $sql = "SELECT * FROM users WHERE username = '"
12  . pf_fix_slashes($_POST['username']) . "' AND password = '"
13  . md5(pf_fix_slashes($_POST['password'])) . "'";
14    $result = mysql_query($sql);
15    $numrows = mysql_num_rows($result);
16
17    if($numrows == 1) {
18       $row = mysql_fetch_assoc($result);
19       session_register("SESS_USERNAME");
20       session_register("SESS_USERID");
21       session_register("SESS_USERLEVEL");
22       $_SESSION['SESS_USERNAME'] = $row['username'];
```

```
23          $_SESSION['SESS_USERID'] = $row['id'];
24          $_SESSION['SESS_USERLEVEL'] = $row['level'];
25          header("Location: " . $config_basedir);
26      }
27      else {
28          header("Location:".$config_basedir."/userlogin.php?error=1");
29      }
30  }
31
32  else {
33      require("header.php");
34      echo "<h1>Авторизация</h1>";
35      if($_GET['error']) {
36          echo "<p>Incorrect login, please try again!</p>";
37  }
38  ?>
39
40  <form action="<?php echo $SCRIPT_NAME ?>" method="post">
41  <table>
42  <tr>
43  <td>Имя пользователя</td>
44  <td><input type="text" name="username"></td>
45  </tr>
46  <tr>
47  <td>Пароль</td>
48  <td><input type="password" name="password"></td>
49  </tr>
50  <tr>
51  <td></td>
52  <td><input type="submit" name="submit" value="Войти!"></td>
53  </tr>
54  </table>
55  </form>
56  <?php
57  }
58  require("footer.php");
59  ?>
```

Последний — четвертый — шаг будет заключаться в создании страницы для корректного выхода с сайта. После того как зарегистрированный пользова-

тель выполнит процедуру выхода, сайт «забудет» о том, что он зарегистрирован в списке пользователей, и будет считать его обычным гостем. Если же этот посетитель снова захочет зайти на административные страницы, он должен будет выполнить процедуру входа заново.

Процедура выхода с сайта описана в программе 2.9

Листинг 2.9 — userlogout.php

```
<?php
session_start();
require("config.php");
session_unregister("SESS_USERNAME");
session_unregister("SESS_USERID");
session_unregister("SESS_USERLEVEL");
header("Location: " . $config_basedir);
?>
```

В программе разрегистрируются переменные текущей сессии, созданные в программе 2.8: SESS_USERNAME, SESS_USERID, SESS_USERLEVEL.

2.3.2 Добавление статей

```
  <?php
session_start();
require("config.php");
require("functions.php");
require("db.php");
require_once 'HTML/QuickForm.php';

if($_SESSION['SESS_USERLEVEL'] < 1) {
   header("Location:" . $config_basedir);
}

$form = new HTML_QuickForm('firstForm');

$catsql = "SELECT id, category FROM categories ORDER BY category;";
$catres = mysql_query($catsql);
while($catrow = mysql_fetch_assoc($catres)) {
   $catarr[$catrow['id']] = $catrow['category'];
}
```

```
19
20 $s =& $form->createElement('select','cat_id','Категория ');
21 $s->loadArray($catarr);
22 $form->addElement($s);
23
24 $form->addElement('text', 'subject', 'Тема', array('size' =>
25  50, 'maxlength' => 255));
26 $form->addElement('textarea', 'body', 'Статья:', array('size' =>
27  50, 'maxlength' => 255, 'rows' => 20, 'cols' => 80));
28 $form->addElement('submit', null, 'Добавить!');
29
30 $form->addRule('subject', 'Пожалуйста, введите тему', 'required', null,
31  'client');
32 $form->addRule('body', 'Введите текст статьи!', 'required', null,
33  'client');
34
35 if($form->validate()) {
36    $form->freeze();
37    $form->process("process_data", false);
38    $insertid = mysql_insert_id();
39    header("Location: " . $config_basedir );
40 }
41
42 else {
43    require("header.php");
44    echo "<h1>Добавить статью</h1>";
45    $form->display();
46 }
47
48 function process_data ($values) {
49    $sql = "INSERT INTO stories(cat_id, poster_id, dateposted, subject,
50  body) VALUES("
51        . $values['cat_id']
52        . ", " . $_SESSION['SESS_USERID']
53        . ", NOW()"
54        . ", '" . pf_fix_slashes($values['subject']) . "'"
55        . ", '" . pf_fix_slashes($values['body'])
56        . "');";
57    $result = mysql_query($sql);
58 }
```

```
59 require("footer.php");
60 ?>
```

2.3.3 Удаление статей

```
1   <?php
2 session_start();
3 require("config.php");
4 require("db.php");
5 require("functions.php");
6 if($_SESSION['SESS_USERLEVEL'] != 10) {
7    header("Location: " . $config_basedir);
8 }
9 if(pf_check_number($_GET['id']) == TRUE) {
10    $validid = $_GET['id'];
11 }
12 else {
13    header("Location: " . $config_basedir);
14 }
15 if($_GET['conf']) {
16    $delsql = "DELETE FROM stories WHERE id = " . $validid . ";";
17    mysql_query($delsql);
18    header("Location: " . $config_basedir);
19 }
20 else {
21    require("header.php");
22    echo "<h1>Are you sure you want to delete this question?</h1>";
23    echo "<p>[<a href='" . $SCRIPT_NAME . "?conf=1&id=" . $validid .
24 "'>Yes</a>] [<a href='index.php'>No</a>]</p>";
25 }
26 require("footer.php");
27 ?>
```

2.4 Дизайн

Веб-дизайн — сложная наука, в задачи которой входит проектирование пользовательских веб-интерфейсов для сайтов или веб-приложений. Веб-дизайнеры проектируют логическую структуру веб-страниц, продумывают наиболее удоб-

ные решения подачи информации, а так же занимаются художественным оформлением веб-проекта.

Для оформления внешнего вида страниц используется CCS — технология описания внешнего вида документа, написанного языком разметки. Получение навыков в веб-дизайне не является целью данного курсового проекта, поэтому используется готовый css-шаблон из книги Jono Bacon [7]. В нём используется крайне популярный вид компановки сайта, представленный на рисунке 2.6.

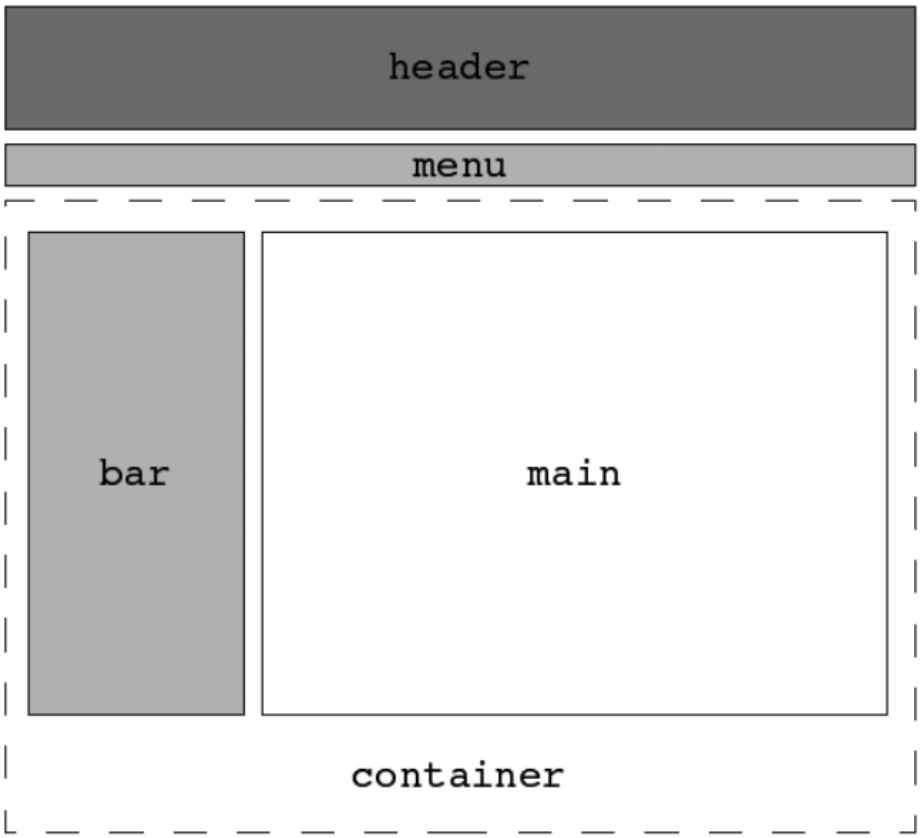

Рисунок 2.6 — Логическая структура страницы

Листинг 2.10 — stylesheet.css

```
1  body {
2   font-family: "trebuchet ms", verdana, sans-serif;
3   font-size: 12px;
4   line-height: 1.5em;
5   color: #333;
6   background: #ffffff;
7   margin: 0;
8   padding: 0;
9   text-align: center;
```

```
10      width: 100%;
11  }
12  p {
13       margin-top: 10px;
14  }
15
16  #header {
17      position: absolute;
18      top: 0px;
19      left: 0px;
20      height: 60px;
21      width: 100%;
22      background: #333;
23      padding-top: 8px;
24  }
25  #header h1 {
26      font-size: 30px;
27      text-transform: uppercase;
28      letter-spacing: 0.3em;
29      color: #fff;
30  }
31  #menu {
32          font-family: "trebuchet ms", verdana, sans-serif;
33          font-size: 14px;
34          font-weight: bold;
35          position: absolute;
36          height: 27px;
37          top: 60px;
38          left: 0px;
39          width: 100%;
40          padding: 0px;
41          color: #000000;
42          background-color: #eee
43  }
44  #container {
45        position: absolute;
46        top: 85px;
47        left: 0px;
48        background: #ffffff;
49        margin: 0 auto 0 auto;
```

```
50      text-align: left;
51      width: 100%;
52      height: 100%;
53  }
54  #bar {
55      float: left;
56      width: 200px;
57      background: #eee;
58      padding: 10px;
59      margin-right: 30px;
60      height: 100%;
61  }
62  img {
63      border: 0;
64  }
65  #bar h1 {
66          font-size: 12px;
67          text-transform: uppercase;
68          letter-spacing: 0.3em;
69  }
70  #main {
71      margin: 15px 15px 15px 240px;
72      padding: 15px 15px 15px 15px;
73      background: #FFFFFF;
74  }
75  table {
76     border: thin solid #cccccc;
77     background: #ffffff;
78  }
79  th {
80     letter-spacing: 2.5px;
81     background: #eeeeeee;
82     color: #000000;
83     text-transform: uppercase;
84     text-align: center;
85     border-top: thick solid #eeeeee;
86     border-bottom: thin solid #cccccc;
87  }
88  .screenshot-right
89  {
```

```
90      padding: 15px;
91      float: right;
92  }
```

3 Интерактивный тест

Тест для проверки знаний реализован на *JavaScript* — скриптовый язык, чаще всего использующийся при создании сценариев поведения браузера, встраиваемых в веб-страницы. Пример написания теста с использованием JavaScript приведён в книге Юрия Ломова [8].

Листинг 3.1 — stylesheet.css

```
<body onload="hideNoDHTML();"> <script type="text/javascript">
<!-- function hideNoDHTML() {
  document.getElementyId('nodhtml').style.display = 'none';
  showQuestions(); }

function showQuestions() {
  document.getElementById('questions').style.display = 'block';
  document.getElementById('results').style.display = 'none'; }

function showResults() { var i = 0;

  if(document.getElementById('choice12').checked == true) { i++; }

  if(document.getElementById('choice23').checked == true) { i++; }

  if ((document.getElementById('choice31').checked == true) &&
  (document.getElementById('choice32').checked == false) &&
  (document.getElementById('choice33').checked == true) &&
  (document.getElementById('choice34').checked == false)) { i++; }

  if((document.getElementById('choice41').checked == false) &&
  (document.getElementById('choice42').checked == true) &&
  (document.getElementById('choice43').checked == true) &&
  (document.getElementById('choice44').checked == false)) { i++; }

  if(document.getElementById('text5').value == 'ГОСТ Р
  34.10-2001') { i++; }

  document.getElementById('questions').style.display = 'none';
  document.getElementById('results').style.display = 'block';

```

```
32        document.getElementById('results').innerHTML = '<h2>Результаты
33        теста</h2>\n<p>Количество правильных ответов: <strong>' + i +
34        '</strong>.</p>';
35
36        if(i == 5) { document.getElementById('results').innerHTML += '<p
37          style="color: #096">Поздравляем с отличным результатом!</p>';
38        }
39
40        if(i <= 2) { document.getElementById('results').innerHTML += '<p
41          style="color: #c00">К сожалению, результат очень плохой.</p>';
42        }
43
44        document.getElementById('results').innerHTML += '<p
45        class="buttons"><input type="button" value="Вернуться к
46        вопросам" onclick="showQuestions();" /></p>'; }
47
48      function getHelp() { document.getElementById('choice12').checked =
49        true;
50
51        document.getElementById('choice23').checked = true;
52
53        document.getElementById('choice31').checked = true;
54        document.getElementById('choice32').checked = false;
55        document.getElementById('choice33').checked = true;
56        document.getElementById('choice34').checked = false;
57
58        document.getElementById('choice41').checked = false;
59        document.getElementById('choice42').checked = true;
60        document.getElementById('choice43').checked = true;
61        document.getElementById('choice44').checked = false;
62
63        document.getElementById('text5').value = 'ГОСТ Р 34.10-2001'; }
64      //--> </script>
65
66      <h1>Интерактивный тест на знание ЭЦП</h1>
67
68      <div id="questions"> <form action=""> <h2>1. Каково, на ваш
69      взгляд, основное предназначение ЭЦП?</h2> <p><input type="radio"
70      name="question1" id="choice11" /> <label for="choice11">Визуальное
71      оформление веб-страниц</label></p> <p><input type="radio"
```

```
72      name="question1" id="choice12" /> <label for="choice12">Защита
73      электронного документа от подделки</label></p> <p><input
74      type="radio" name="question1" id="choice13" /> <label
75      for="choice13">Регистрация пользователей</label></p> <p><input
76      type="radio" name="question1" id="choice14" /> <label
77      for="choice14">Создание зашифрованных разделов жёсткого
78      диска</label></p>
79
80      <h2>2. Стандарт электронной цифровой подписи:</h2> <p><input
81      type="radio" name="question2" id="choice21" /> <label
82      for="choice21">POP3</label></p> <p><input type="radio"
83      name="question2" id="choice22" /> <label
84      for="choice22">UTF</label></p> <p><input type="radio"
85      name="question2" id="choice23" /> <label
86      for="choice23">ElGamal</label></p> <p><input type="radio"
87      name="question2" id="choice24" /> <label
88      for="choice24">PHP</label></p>
89
90      <h2>3. RSA – это</h2> <p><input type="checkbox" id="choice31" />
91      <label for="choice31">Алгоритм, пригодный для
92      шифрования</label></p> <p><input type="checkbox" id="choice32" />
93      <label for="choice32">Операционная система</label></p> <p><input
94      type="checkbox" id="choice33" /> <label
95      for="choice33">Криптографический алгоритм с открытым
96      ключом</label></p> <p><input type="checkbox" id="choice34" />
97      <label for="choice34">Открытый ключ</label></p>
98
99      <h2>4. Назовите возможные угрозы ЭЦП?</h2> <p><input
100     type="checkbox" id="choice41" /> <label
101     for="choice31"></label>Злоумышленник узнает алгоритм
102     шифрования</p> <p><input type="checkbox" id="choice42" /> <label
103     for="choice32">Злоумышленник может попытаться подобрать документ к
104     данной подписи, чтобы подпись к нему подходила</label></p>
105     <p><input type="checkbox" id="choice43" /> <label
106     for="choice33">Злоумышленник может попытаться подделать подпись
107     для выбранного им документа</label></p> <p><input type="checkbox"
108     id="choice44" /> <label for="choice34">Злоумышленник узнает о
109     существовании ЭЦП</label></p>
110
111     <h2>5. Российский стандарт электронной цифровой подписи:</h2>
```

```
112     <p><input type="text" size="20" maxlength="20" id="text5" /></p>
113
114     <p class="buttons"><input type="button" value="Показать
115     результаты" onclick="showResults();" /> <input type="button"
116     value="Подсказать ответы" onclick="getHelp();" /> <input
117     type="reset" value="Очистить форму" /></p> </form> </div>
118
119     <div id="results"></div> </body>
```

Результат работы программы представлен на рисунке 3.1.

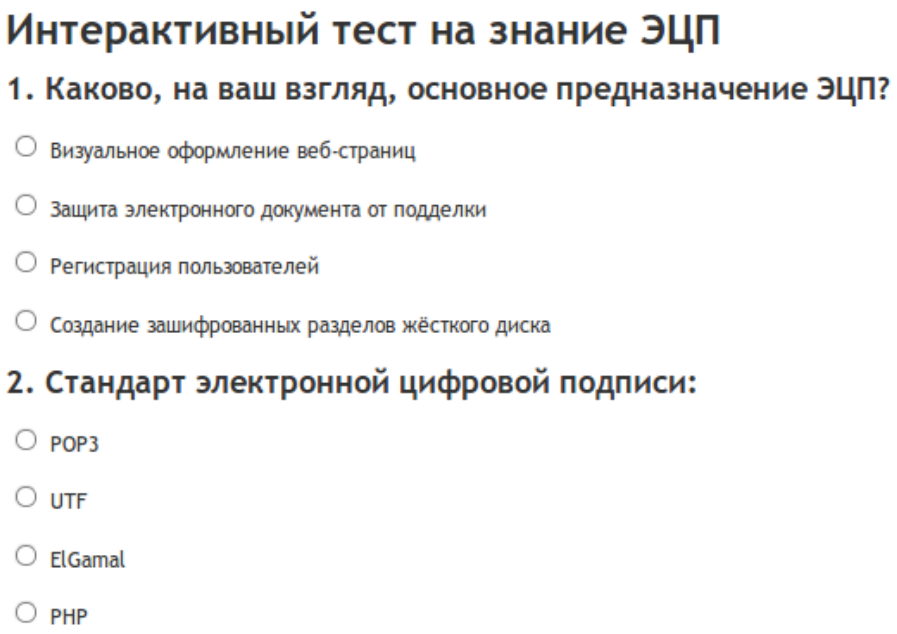

Рисунок 3.1 — Интерактивный тест

Если тест пройден правильно, пользователь получит отличную оценку (рисунок 3.2).

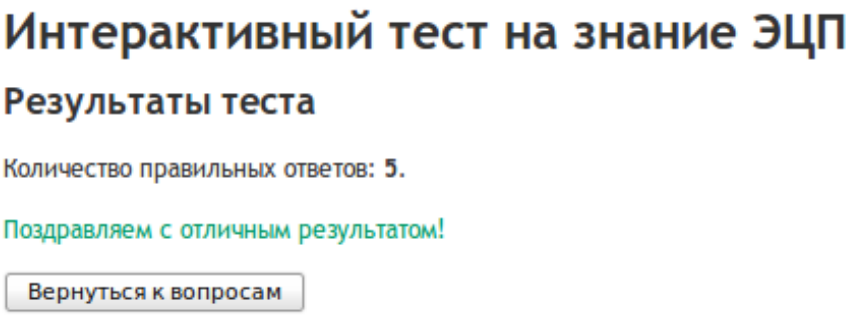

Рисунок 3.2 — Результат работы теста

4 Инструкция пользователя сайта

В данном проекте был реализован динамический веб-сайт, содержимое которого хранится в БД MySQL. Одно из главных преимуществ сайта — возможность изменения его содержимого без знаний принципов функционирования MySQL.

Авторизовавшись, пользователь способен быстро создавать новые статьи в любом из разделов или подразделов. На сайте реализована удобная панель (см. рисунок 2.4), преставляющая собой интерфейс для входа на сайт, выхода с сайта, добавления новых статей.

Для администратора сайта реализовано удаление статей одним кликом мышки.

Неавторизованные пользователи могут просматривать содержимое сайта, проверить «на лету» насколько хорошо они усвоили метериал с помощью процедуры тестирования.

Заключение

В результате выполнения курсовой работы былы написан динамический веб-сайт с возможностью выборочного добавления, удаления содержимого. Реализована процедура тестирования посетителей сайта, создано три категии пользователей. В процессе работы над проектом былы изучены технологии веб-программирования с использованием PHP, MySQL, JavaScript, CSS.

Таким образом, задание на курсовой проект было выполнено в полном объёме.

Список использованных источников

1 *Аггерих Л. и др.* Профессиональное PHP программирование : Пер. с англ. — 2-е изд. — СПб.: Символ-Плюс, 2003. — 1048 с.

2 *Википедия.* Apache — Википедия, свободная энциклопедия. — 2009. — [Online; accessed 1-май-2009]. `http://ru.wikipedia.org/?oldid=15326288`.

3 *Харрис Э.* PHP/MySQL для начинающих : Пер. с англ. — М.: КУДИЦ-ОБРАЗ, 2005. — 384 с.

4 *Евдокимов М.* Динамический web-сайт // *КомпьютерПресс.* — 2001. — № 6.

5 *Томсон Л., Веллинг Л.* Разработка Web-приложений на PHP и MySQL : Пер. с англ. — 2-е, испр. изд. — СПб.: ООО «ДиаСофтЮП», 2003. — 672 с.

6 *Википедия.* Md5 — Википедия, свободная энциклопедия. — 2009. — [Online; accessed 16-май-2009]. `http://ru.wikipedia.org/?oldid=15740489`.

7 *Bacon J.* Practical PHP and MySQL: building eight dynamic Web applications. Negus live Linux series. — pub-PH:adr: Prentice-Hall, 2007. — Pp. XIII, 512. `http://www.loc.gov/catdir/toc/ecip0619/2006027701.html`.

8 *Ломов А. Ю.* HTML, CSS, скрипты: практика создания сайтов. — СПб.: БХВ-Петербург, 206. — 416 с.