

«Липецкий государственный технический университет»

Кафедра прикладной математики

Отчет по лабораторной работе №5
«Создание веб-сервера на базе Windows»
по курсу «Операционные системы и сети ЭВМ»

Студент

Быкова Н.В.

подпись, дата

Группа ПМ-12-1

Преподаватель

Николаев Д.А.

подпись, дата

Липецк 2013 г.

Содержание

1	Цель работы	3
2	Ход работы	3
2.1	Установка пакета ХАМРР	3
2.2	Написание сайта	6
2.3	Работа с СУБД	9
2.4	Регистрация и авторизация	11
3	Вывод	17
	Список литературы	17

1 Цель работы

Мы должны освоить основные конструкции языка HTML и создать небольшой сайт, научиться с помощью phpmyadmin работать с СУБД и создать базу данных из одной таблицы, а также с помощью языка php реализовать возможность регистрации и авторизации пользователей на сайте.

2 Ход работы

2.1 Установка пакета XAMPP

Скачаем XAMPP с сайта apachefriends.org. XAMPP – это локальный сервер, который представляет собой набор программ, которые позволяют разрабатывать сайт на локальном компьютере без подключения к Интернету. Если бы мы создавали сайт используя только html-страницы, ну может еще с дизайном CSS, то никакой локальный сервер нам не понадобился бы. Но если сайт динамичный, т.е. создан с использованием Php, MySQL, скриптов и т.д., то для его отладки и тестирования понадобится локальный сервер. Так вот, XAMPP используется для разработки клиент-серверных приложений. То есть, например, есть у нас какое-то приложение, пользователь со своего компьютера вводит туда какие-то данные, эти данные посылаются на сервер, где они обрабатываются базой данных MySQL и раскладываются по таблицам. Данные хранятся там, до того момента пока пользователь не затребует их для просмотра или изменения. Когда у пользователей возникает необходимость получить определенную информацию, они отдают команду программе получить с сервера необходимые сведения, та в свою очередь отправляет запрос к MS SQL Server-у, тот на основании этого запроса формирует данные и отправляет их обратно программе, которая уже и показывает пользователям ту информацию которая им нужна. Отсю-

да нетрудно сделать вывод: Программа – Клиент не может работать без Сервера баз данных. А так как мы собираемся создавать таблицу базы данных для регистрации новых пользователей на сайте, то без клиент-серверной архитектуры нам не обойтись. При запуске мы видим следующее:

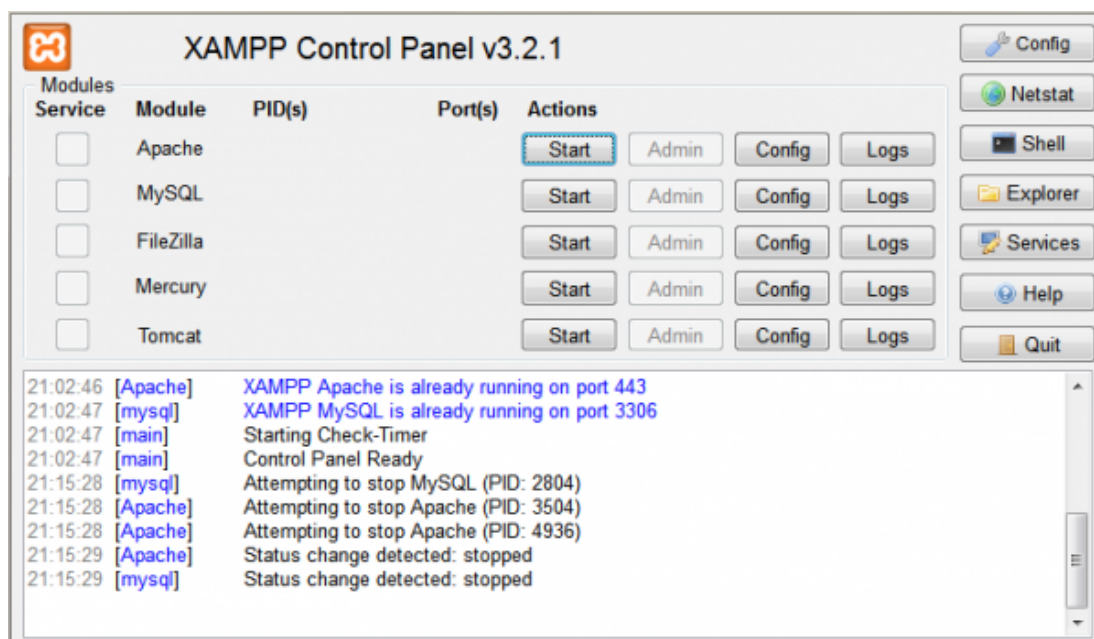


Рис. 1. XAMPP Control Panel

На рис. 1 видно, что у нас есть Apache, MySQL, FileZilla, Mercury, Tomcat. Но нам потребуются только первые два компонента. Apache – это веб-сервер, MySQL – система управления базами данных.

Основными достоинствами Apache считаются надёжность и гибкость конфигурации. Благодаря этому ПО можно осуществлять подключение внешних модулей, использующихся для предоставления данных, модифицировать сообщения об ошибках, применять СУБД для аутентификации пользователей. Apache оснащен встроенным механизмом виртуальных хостов. Благодаря этому на одном IP-адресе можно обслуживать большое количество веб-проектов (доменных имен), при этом отображая для каждого из них свое содержимое.

MySQL – это одна из самых популярных и самых распространенных СУБД (система управления базами данных) в интернете. Она не предназна-

чена для работы с большими объемами информации, но ее применение идеально для интернет сайтов, как небольших, так и достаточно крупных. MySQL отличается хорошей скоростью работы, надежностью, гибкостью. Работа с ней, как правило, не вызывает больших трудностей. Поддержка сервера MySQL автоматически включается в поставку PHP.

PHP это язык программирования, специально разработанный для написания web-приложений, исполняющихся на Web-сервере. Нам потребуется это язык для написания авторизации и регистрации пользователей на сайте, так как html отвечает только за оформление сайта, а php посылает серверу команды, которые надо выполнить (в нашем случае, это будет подключение к базе данных и сохранение/проверка логина и пароля).

Запустим в XAMPP обе программы. Так как я в ходе выполнения лабораторной работы случайно удалила главную страницу программы, где было меню, то запустить я могу только базы данных MySQL.

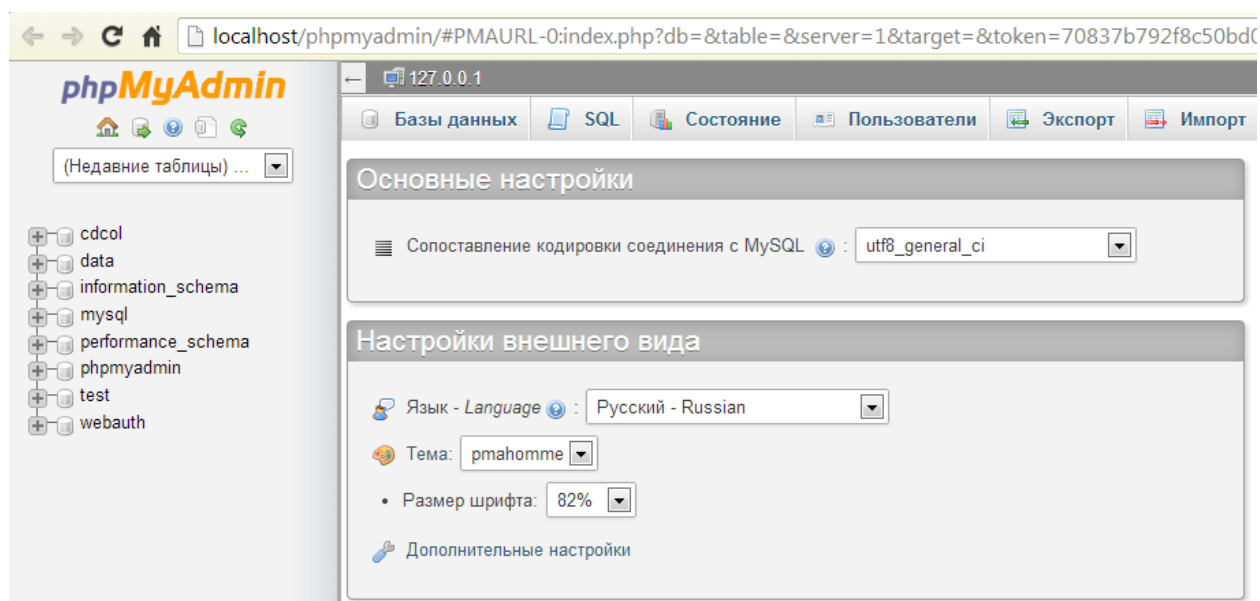


Рис. 2. Запуск phpmyadmin

Видим, что программа открылась в браузере, только вместо привычного адреса у нас localhost. В целом, вся информация, нужная нам представлена на этой странице, только чуть правее:

По рис. 3 можно определить имя пользователя, php расширение,

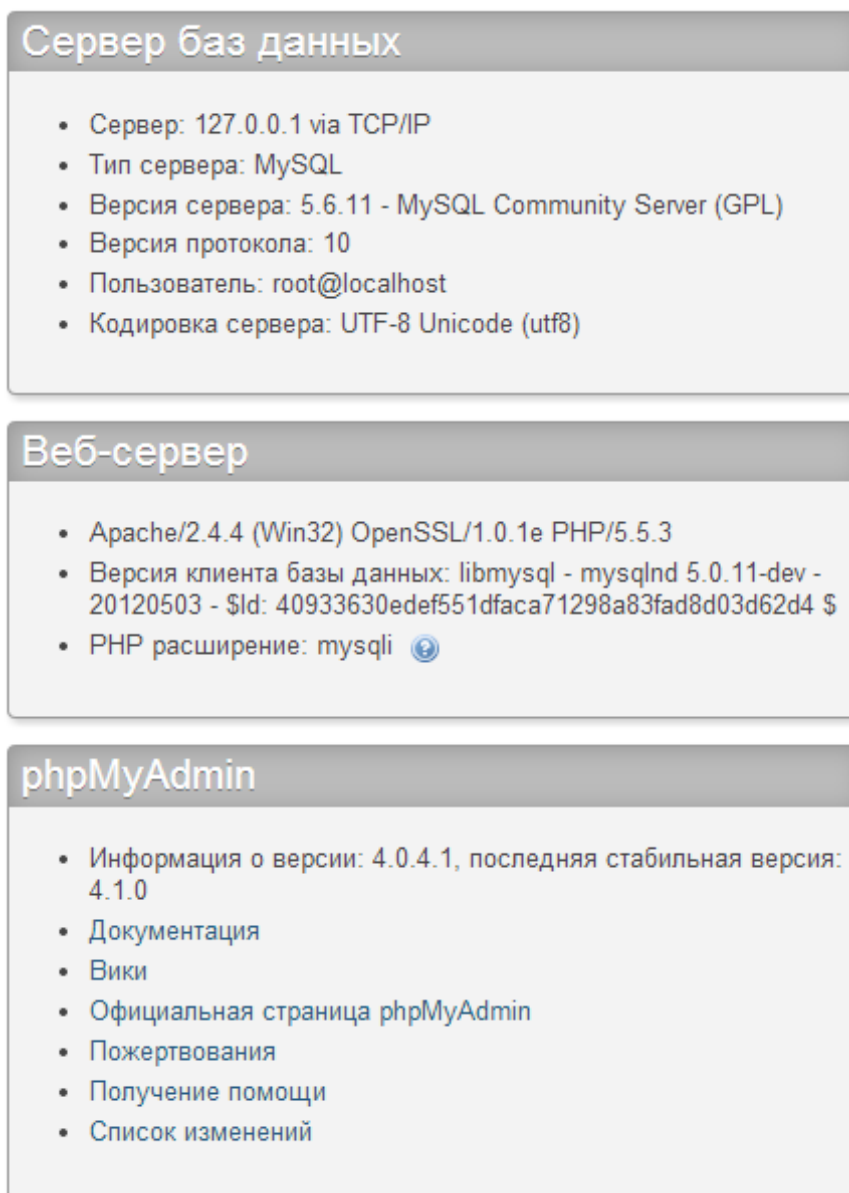


Рис. 3. Основные сведения

сервер. По сути, это все, что нам надо. Перейдем к следующему пункту в лабораторной работе.

2.2 Написание сайта

Освоим основные конструкции языка HTML и создадим небольшой сайт-визитку. Сайт можно создавать с помощью фреймов, а можно и с помощью таблицы. Я создала с помощью фреймов. Страницы сайта я набирала в Notepad++, с кодировкой UTF8 и расширением html. Главная страница выглядит так:

```

1 <html>
2   <head>
3     <title>"Сюрреализм - это я!"</title>
4   </head>
5   <frameset cols="*,800,*">
6     <noframes>Извините, но Ваш браузер не поддерживает фреймы..
7   </noframes>
8     <frame src="dekor.html" scrolling="no" noresize>
9     <frameset rows="160,*">
10      <frame src="logo.html" scrolling="no" marginwidth="0"
11      marginheight="0" noresize>
12      <frameset cols="200,600">
13        <frame src="menu.html" noresize>
14        <frame src="text.html" name="osnovnoe"
15        marginwidth="10" marginheight="10" noresize>
16      </frameset>
17    </frameset>
18    <frame src="dekor.html" scrolling="no" noresize>
19  </frameset>
20 </html>

```

Рис. 4. Код главной страницы

Сначала идет тег `<html>`, который указывает начало документа. Потом мы указываем, как у нас будет называться вкладка в браузере («Сюрреализм - это я!»). А дальше идет основной код страницы. Он заключён в контейнер `<frameset> ... </frameset>`. Так как бывает, что у некоторых пользователей фреймы не отображаются или отключены, я включила в код команду `<noframes>`, которая в таких случаях выведет сообщение об ошибке. Cols – это столбцы. Мы задали три основных столбца, но только один имеет жесткий размер, остальные просто занимают все оставшееся место на экране. Все это выполнено с помощью [1] и [2].

Дальше мы первому столбцу дали ссылку на файл `dekor.html`, где у нас хранится вставка изображения для заднего фона. В 15 строке видно, что 7 строка повторяется. Всё верно, это мы задали третий столбец. Заметим, что у нас в этих строках есть еще 2 дополнения: `scrolling` и `noresize`. Первый, со значением «но» убирает ненужные полосы прокрутки, второй - запрещает пользователям менять границы фреймов.

Строки 8 – 13 задают фрейм в центре. Сначала мы делим его на две строки, одну размером 160, другую динамичную. Поля фреймов или

иначе расстояние от границ фрейма до текста или картинки, как в нашем случае, задаются в пикселях при помощи атрибутов `marginwidth` и `marginheight`. Так что картинка из файла `logo.html` прекрасно отображается. Динамичную строку мы делим еще на два столбца. В первом будет меню, а второй это контент сайта. В строке с контентом (ссылка `text.html`) есть еще команда `name`. Это, в своем роде, флажок, который показывает где будут открываться ссылки из меню. Меню выглядит вот так:

```

1  <html>
2  <head>
3      <title>"Сюрреализм - это я!"</title>
4      <style>
5          a {
6              text-decoration: none;
7          }
8      </style>
9  </head>
10 <body bgcolor="#FFF8DC">
11     <font face="Monotype Corsiva" size="6"color="#A0522D"><center>
12     Меню:</center></font><hr color=#8B4513">
13         <font face="Cursive" size="4">
14             <a href="/xampp/htdocs/text.html" target="osnovnoe"><font
15             color="#8B4513">Главная</font></a><hr color=#8B4513">
16             <a href="/xampp/htdocs/biography.html" target="osnovnoe"
17             ><font color="#8B4513">Биография Дали</font></a><hr color=
18             #8B4513">
19             <a href="/xampp/htdocs/foto.html" target="osnovnoe"><font
20             color="#8B4513">Фото Дали</font></a><hr color=#8B4513">
21             <a href="/xampp/htdocs/pictures.html" target="osnovnoe"
22             ><font color="#8B4513">Репродукции картин</font></a><hr
23             color=#8B4513">
24             <a href="/xampp/htdocs/surrealizm.html" target="osnovnoe"
25             ><font color="#8B4513">Сюрреализм и Дали</font></a><hr
26             color=#8B4513">
27             <a href="/xampp/htdocs/aforizm.html" target="osnovnoe"
28             ><font color="#8B4513">Афоризмы Дали</font></a><hr color=
29             #8B4513">
30         </font>
31     </body>
32 </html>

```

Рис. 5. Код меню

На рис. 5 видна «закладка» `target`, которая, в случае отсутствия флага `name` открывала бы ссылку в новой вкладке. В этом коде в основном только ссылки и настройка цветов и шрифтов. Упоминания заслуживают 4 – 8 строки. Здесь применена стилевая обработка, которая убирает подчеркивания ссылок.



Рис. 5. Код меню

В конце работы мы получили готовый сайт, посвященный Сальвадору Дали (рис.6) и 15 html-страниц.

2.3 Работа с СУБД

Научимся с помощью phpmyadmin работать с СУБД и создадим свою базу данных из одной таблицы и занесем в нее несколько логинов и паролей пользователей. Заходим в ХАМРР, запускаем, как в прошлый раз, phpmyadmin. Слева, где представлены базы данных, разворачиваем mysql и нажимаем на «Создать новую таблицу». Вводим название таблицы – laba, и начинаем заполнение. В первую строку вводим имя – id, тип – int, длина – 10. Во вторую – login, varchar, 50, и третья – password, varchar, 16. Жмем «сохранить», и вот что получаем:

Видим, что у нас выскочило, какое-то непонятное сравнение latin1_swedish_ci, хотя у нас используется utf8_general_ci. Меняем. Чуть позже

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополните
<input checked="" type="checkbox"/> 1	id	int(10)			Нет	Нет	
<input type="checkbox"/> 2	login	varchar(50)	latin1_swedish_ci		Нет	Нет	
<input type="checkbox"/> 3	password	varchar(16)	latin1_swedish_ci		Нет	Нет	

Рис. 6. Таблица в базе данных mysql

выяснится, что колонка id нам бесполезна, так как в новом php расширении mysql не автоинкрементации для нее. Так как добавление новых пользователей я проводила уже после удаления этой колонки, то команда несколько не соответствует таблице. Вот так таблица стала выглядеть:

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополните
<input checked="" type="checkbox"/> 1	login	varchar(50)	utf8_general_ci		Нет	Нет	
<input checked="" type="checkbox"/> 2	password	varchar(16)	utf8_general_ci		Нет	Нет	

Рис. 7. Измененная таблица в базе данных mysql

Теперь добавим вручную новых пользователей в базу данных. Заходим в SQL, стираем все, что есть в поле и вводим нужные команды (рис.8).

Выполнить SQL-запрос(ы) к базе данных mysql:

```

1 INSERT INTO `laba` (`login`, `password`) VALUES ('Olya', '1234');
2 INSERT INTO `laba` (`login`, `password`) VALUES ('Ivanushka', 'skazka011101');

```

Рис. 8. Добавление пользователей

Добавление прошло успешно, что можно увидеть на странице «Обзор», где появилась такая надпись:

Olya	1234
Ivanushka	skazka011101

Рис. 9. Проверка добавления

Все отлично, логины и пароли числятся в базе данных. Теперь перейдем к написанию регистрации и авторизации.

2.4 Регистрация и авторизация

С помощью языка php реализуем возможность авторизации и регистрации пользователей на сайте. Начнем с регистрации, и для начала напишем форму.

```
8  <font face="Cursive" size="4" color="#A0522D">
9  Регистрация</font>
10
11  <font face="Cursive" size="3" color="#A0522D">
12  <form action="save.php" method="post" target="regi">
13  <p>
14  <label> Логин: <br></label>
15  <input name="login" type="text" size="50" maxlength="50">
16  </p>
17  <p>
18  <label> Пароль: <br></label>
19  <input name="password" type="password" size="16" maxlength="16">
20  </p>
21  <input type="submit" name="submit" value="Зарегистрироваться">
22  <p align="center">
23  <a href='form.php' target="regi"><font color="#8B4513" size="4">Назад
24  </font></a></p></font>
25  </form>
```

Рис. 10. Форма регистрации

Форма написана на языке html с помощью сайта [2], видны ограничения на количество вводимых символов: для логина – 50, для пароля – 16. Есть ссылка на файл save.php, который проверяет, отправляет и сохраняет введенные пользователем данные в созданную ранее таблицу. В конце есть ссылка назад, которая переправит нас на форму для авторизации (в случае, если пользователь вспомнил, что он уже зарегистрирован или нажал на «Регистрация» по ошибке). Тег <label> устанавливает связь между определенной меткой, в качестве которой обычно выступает текст, и элементом формы (у меня <input>). Такая связь необходима, чтобы изменять значения элементов формы при нажатии курсором мыши на текст. Тег <input> главным образом предназначен для создания текстовых

полей, различных кнопок, переключателей и флажков. Хотя элемент `<input>` не требуется помещать внутрь контейнера `<form>`, определяющего форму, но если введенные пользователем данные должны быть отправлены на сервер, где их обрабатывает серверная программа, то указывать `<form>` обязательно. Основным атрибутом тега `<input>`, определяющий вид элемента — `type`. Он позволяет задавать следующие элементы формы: текстовое поле (`text`), поле с паролем (`password`), кнопка для отправки формы (`submit`). Теперь, с помощью книги [3] напишем документ `save.php`. Вот ключевые моменты:

```
7  <?php
8      if (isset($_POST['login'])) {$login=$_POST['login']; if($login == '')
9          {unset($login);}}
10     if (isset($_POST['password'])) {$password=$_POST['password']; if(
11         $password == '') {unset($password);}}
12     if (empty($login) or empty($password))
13     {
14         exit("Вы ввели не всю информацию, вернитесь назад и заполните все
15         поля! <a href='reg.php'>Регистрация</a>");
16     }
```

Рис. 11. Файл `save.php`. Часть 1

Функция `isset` — определяет, установлена ли переменная. `$_POST` это ассоциативный массив, то есть он для обращения к элементам использует ключи, логически связанные со значениями. В этот массивы помещаются данные, передаваемые скрипту извне (так называемый, пользовательский ввод). В принципе, пользователь может влиять на этот массив. И именно поэтому все элементы этих массивов должны тщательно проверяться на допустимые значения. Функция `unset` — уничтожает переданную в качестве аргумента переменную. Функция `empty` — определяет, установлена ли переменная. `Exit` — выводит сообщение и прекращает выполнение текущего скрипта. То есть, здесь мы заносим введенный пользователем логин (пароль) в переменную `$login` (`$password`), если он пустой, то уничтожаем переменную. Кстати, это как раз объясняет, как `php` взаимодействует с полями для ввода `html`. Именно отсюда начинается это взаимодействие, и начинается оно с

проверки введенных пользователем данных.

```
21 include("bd.php");
22 $result = mysqli_query($mysqli, "SELECT login FROM laba WHERE login='
    $login'");
23 $myrow = mysqli_fetch_array($result);
24 if (!empty($myrow['login']))
25 {
26     exit ("Извините, введенный Вами логин уже зарегистрирован. <a
        href='reg.php'>Назад</a>");
27 }
28 $result2 = mysqli_query ($mysqli, "INSERT INTO laba (login, password)
    VALUES ('$login', '$password')");
29 if ($result2=='TRUE')
30 {
31     echo "Вы успешно зарегистрированы! Здравствуйте, ".$login;
32 }
33 else
34 {
35     echo "Ошибка! Вы не зарегистрированы.<a href='reg.php'>Назад</a>";
36 }
37 ?>
```

Рис. 12. Файл save.php. Часть 2

На рис. 12 мы сначала подключаемся к базе данных (строка 21). Конструкция `include` предназначена для включения файлов в код сценария PHP во время исполнения сценария PHP. Затем, в строках 22 – 27 проверяем существование пользователя с таким же логином, ну и если такого нет, то сохраняем данные (строка 28). Команда `mysqli_fetch_array` возвращает массив соответствующий выбранной строке или `NULL`, если в результирующей таблице, представленной параметром `result`, больше нет доступных строк. И в конце (строка 29 – 36) проверка ошибок. Команда `echo` - выводит одну и более строк на экран. Кроме того, я еще поставила команды, которые не позволяют работать тегам и скриптам, и удалят лишние пробелы. Как видим, в `save.php` на поверхность всплывает еще один файл – `bd.php`. В нём содержится код подключения к базе данных:

Строка `$mysqli = new mysqli` означает подключение к базе данных с использованием процедурного интерфейса. Вот так, в случае ошибки, мы увидим сообщение о ней. Что означает этот код, подписано в самом рис. 13, в комментариях. Но есть две разных команды для вывода ошибок: `mysqli_connect_error` и `mysqli_connect_errno`. Первая выдает строку с

```

1  <?php
2      $mysqli = new mysqli(
3          '127.0.0.1', /* Хост, к которому мы подключаемся */
4          'root',      /* Имя пользователя */
5          '',          /* Используемый пароль */
6          'mysql', 3306); /* База данных для запросов по умолчанию */
7
8  if ($mysqli->connect_errno) {
9      echo "Не удалось подключиться к MySQL: (" . $mysqli->connect_errno . ") "
10         . $mysqli->connect_error;}
11  ?>

```

Рис. 13. Подключение к базе данных

описанием ошибки, вторая - с кодом ошибки. Если ошибок нет, то ничего не выводят.

Осталось сделать форму для авторизации, которую мы поместим на главной странице, и код проверки введенных логина и пароля с теми, которые уже хранятся в базе данных. Форма авторизации практически ничем не отличается от формы регистрации, кнопка называется по-другому и вписывается еще такой код:

```

<?php
    if (empty($_SESSION['login']) or empty($_SESSION['id']))
    {
        echo "Вы вошли на сайт, как гость <br><a href='#'></a>";
    }
    else
    {
        echo "Здравствуйте ".$_SESSION['login']." <br><a
            href=index.html></a>";
    }
}
?>

```

Рис. 14. Часть формы авторизации

На рис. 14 фигурирует массив `_SESSION`. Предназначение массива `$_SESSION` – хранение всех переменных сессии текущего пользователя. Здесь, в случае, если пользователь не вошел на сайт, под формой авторизации будет отображаться сообщение о том, что пользователь зашел как гость, в противоположном случае, сайт его поприветствует. Хотя для второго случая необходимо, чтобы была реализована возможность запоминания пользователя, а я ее не прописывала. Форма авторизации ссылается на `testreg.php`, который от `save` тоже не сильно отличается.

```

25 // подключаемся к базе
26 include ("bd.php");
27 $result = mysqli_query($mysqli, "SELECT * FROM laba WHERE login='$login'");
28 //извлекаем из базы все данные о пользователе с введенным логином
29 $myrow = mysqli_fetch_array($result);
30 if (empty($myrow['password']))
31 {
32     exit ("Извините, введенный вами login или пароль неверный. <a
33 href='form.php'>Назад</a>");
34 }
35 else {
36     if ($myrow['password']==$password) {
37         $_SESSION['login']=$myrow['login'];
38         echo "Здравствуйте, ".$_SESSION['login']."!";
39     }
40     else {
41         exit ("Извините, введенный вами login или пароль неверный. <a
42 href='form.php'>Назад</a>");
43     }
44 }

```

Рис. 15. Файл testreg.php

В приведенном рис. 15 видно сравнение введенного логина, а затем и пароля с теми, что уже присутствуют в базе данных. При совпадении, сайт приветствует пользователя, иначе – выдает ошибку и дает ссылку, чтобы вернуться назад и заполнить поля вновь.

The image shows a web page with a registration form on the left and a text block on the right. The form is titled "Галерея картин Афоризмы Дали" and "Регистрация". It contains fields for "Ваш логин:" and "Ваш пароль:", a "Войти" button, and a message "Вы вошли на сайт, как гость". The text block on the right is titled "Сальвадор Дали." and contains a paragraph about his life and work.

Рис. 16. Регистрация на сайте

Мы встроили код, в частности файл form.php на сайт. Правда, из-за того, что логину отводилось 60 символов, поле для ввода сильно растянулось,

так что пришлось подредактировать файлы и выставить максимальную длину 16. Для вставки form на сайт мы разобили на главной странице фрейм еще на две части, чтобы было удобно сделать «закладку» и «флаг». А также убрали из меню подчеркивания, чтобы отображение сайта было удобным. И для регистрации не пришлось бы проматывать сайт вверх-вниз. Проверим работает ли авторизация. Чуть ранее я зарегистрировала пользователя Admin, с паролем admin111.

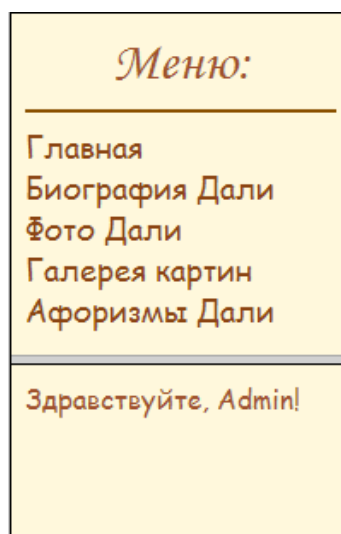


Рис. 17. Авторизация пользователя

Видим, что сайт меня поприветствовал. Кстати, если написать имя пользователя с маленькой буквы, то мы также зайдём на сайт. А что будет, если ввести неправильную информацию или оставить поля пустыми?

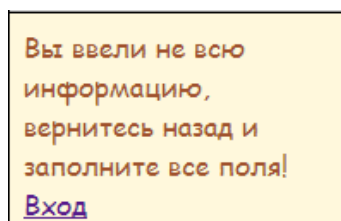
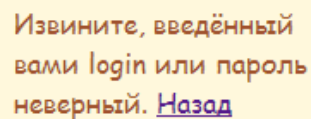


Рис. 18. Пользователь не ввел ничего

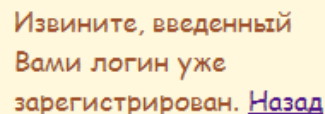
Итак, на рис. 18 видна реакция на незаполненные поля, а рис. 19 показывает, какая ошибка высвечивается, если неправильно ввести логин или пароль. В моем случае я недописала пароль. А точно ли работает



Извините, введенный
вами login или пароль
неверный. [Назад](#)

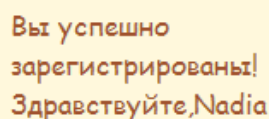
Рис. 19. Ошибка в пароле

регистрация? Попробуем зарегистрировать пользователя с именем Admin, и с именем Nadia.



Извините, введенный
Вами логин уже
зарегистрирован. [Назад](#)

Рис. 20. Регистрация пользователя Admin



Вы успешно
зарегистрированы!
Здравствуй, Nadia

Рис. 21. Регистрация пользователя Nadia

В первом случае мы получили отказ в регистрации, ввиду того, что пользователь с таким именем в базе данных уже есть. А во втором случае меня успешно зарегистрировали. Проверим нашу таблицу в базе данных:

Итак, мой новый пользователь занесен последним в таблицу. То есть, и регистрация, и авторизации работают корректно.

3 Вывод

В ходе лабораторной освоили основные конструкции языка HTML и создали небольшой сайт, научились с помощью phpmyadmin работать с СУБД и создали базу данных из одной таблицы, а также с помощью языка php реализовали возможность регистрации и авторизации пользователей на сайте.

login	password
Admin	admin111
N1	1234
User	test
A2	tests
Example	example
Test	qwerty
lala	qwerty
B1	b1b1
Salomon	privetiki
A4	proverka
Olya	1234
Ivanushka	skazka011101
Nadia	alilya

Рис. 22. Просмотр таблицы laba

Список литературы

- [1] Маркин А. Руководство по созданию сайта / Режим доступа:
www.URL: <http://www.eltisbook.ru/sait/sait.php>
- [2] Семикопенко А. А. Учебник HTML для начинающих / Режим доступа:
www.URL: <http://www.webremeslo.ru/html/glava0.html>
- [3] Веллинг Л., Томсон Л. Разработка веб-приложений с помощью PHP и MySQL.: – М.: И.Д. «Вильямс», 2010. - 848 с.