**Практическая работа Разработка веб-приложения «Заметки»**

**Часть 2.**

**Обмен данными между серверными страницами**

Каждая заметка на главной странице блога может быть прокомментирована. Для реализации этой функции необходимо сделать из заголовка каждой заметки гиперссылку, перейдя по которой посетитель попадет на страницу со списком комментариев к выбранной заметке. Кроме того, на этой же странице должна отображаться сама выбранная для комментирования заметка.

Следовательно, необходимо реализовать механизм обмена данными между страницами таким образом, чтобы при переходе по гиперссылке передавалась информация о том, какая именно заметка была выбрана.

Для этого необходимо ввести некий идентификатор, значение которого будет совпадать с id комментируемой заметки, и который будет передаваться при переходе по гиперссылке.

**Упражнение 1. Создание гиперссылки**

1. Создайте новую страницу comments.php, которая будет содержать комментарии к выбранной заметке.
2. Реализуйте соединение с сервером.
3. Реализуйте подключение к БД.
4. Для передачи идентификатора заметки введем аргумент note. В качестве значения он будет получать значение поля id таблицы notes.
5. На странице blog.php найдите фрагмент кода, передающего заголовок заметки title (echo $note [‘title’];). Его необходимо отредактировать таким образом, чтобы он стал гиперссылкой на страницу комментариев comments.php, а также передавал id выбранной заметки:

while ($note = mysqli\_fetch\_array($select\_note)){

echo $note['id'], "<br>";

?>

<a href="comments.php?note=<?php echo $note['id']; ?>">

<?php echo $note ['title'], "<br>";?></a>

<?php

echo $note ['created'], "<br>";

echo $note ['content'], "<br>";

}

?>

Здесь мы создаем гиперссылку на страницу comments.php и в этой гиперссылке передаем идентификатор note, значение которого равно значению элемента массива $note[‘id’], т.е. значению id заметки.

**Упражнение 2. Страница comments.php**

1. Перейдите на страницу comments.php. На данной странице должны отображаться комментарии к выбранной записи, а также сама комментируемая запись (для удобства посетителя сайта).
2. Данную задачу можно выполнить по аналогии с выводом заметок на странице blog.php. Основное отличие заключается в том, что вначале необходимо со станицы blog.php получить переданный с помощью идентификатора note id заметки. Это делается с помощью метода $\_GET:

*//Переменной $note\_id необходимо присвоить id заметки, переданной с помощью метода $\_GET со страницы blog.php*

*$note\_id = $\_GET['note'];*

1. Далее необходимо вывести значения полей created, title, content из таблицы notes для заметки с полученным id. Для этого используется SQL запрос

SELECT… FROM… WHERE…

В нем с помощью оператора SELECT выбираем необходимые поля таблицы; с помощью

FROM определяем таблицу-источник выборки; WHERE задает условие отбора, по которому выбираем заметку с выбранным id:

//Формируем SQL-запрос на выборку с учетом переданного id заметки

$query = "SELECT created, title, article FROM notes WHERE id = $note\_id";

4. После формирования SQL-запроса его необходимо реализовать с помощью функции mysqli\_query() и вывести данные на страницу с помощью функции mysqli\_fetch\_array().

5. Затем аналогичным образом выведите комментарии к выбранной заметке. Обратите внимание, что комментариев должен строиться следующим образом:

$query\_comments = "SELECT \* FROM comments WHERE art\_id =$note\_id";

В условии WHERE мы реализуем поддержку связи между таблицами, которые связаны по полям id (таблица notes) и art\_id (таблица comments).

В переменной $note\_id содержится следовательно, для выбора комментариев к этой заметке необходимо, чтобы значение поля art\_idcreated также было равно $note\_id.

1. Проверьте корректность данных между страницами comments.php. При переходе по ссылке с blog.php на comments.php в адресной строке браузера должен отображаться id выбранной заметки, переданный с помощью идентификатора note.
2. Для того, чтобы выводились все комментарии, а не только первый – реализуйте цикл.
3. Если у заметки нет ни одного комментария – об этом надо сообщить.

6.1 Под областью комментариев добавьте надпись «Эту запись еще никто не комментировал».

6.2 В коде программы создайте циклы с условием if: если хотя бы один комментарий существует – он должен быть выведен (т.е. элементы массива должны быть отображены); если количество комментариев равно нулю – должна выводиться надпись «Эту запись еще никто не комментировал».

**Ввод и правка данных с помощью формы**

**Упражнение 3. Отправка почты**

Данное упражнение позволяет реализовать отправку сообщения через форму на сервер.

* 1. Создайте страницу email.php. Добавьте название страницы и пояснительный тест, форму с двумя текстовыми полями: Тема сообщения и Текст сообщения, кнопку Отправить, гиперссылку для возврата на главную страницу сайта.
  2. Самостоятельно реализуйте обработкку данных формы с помощью функции mail(). «Получить» отправленное сообщение вы можете по локальному адресу: С:\WebServers\tmp\!sendmail\
  3. Проверьте корректность работы, создайте гиперссылки с главной страницы сайта на страницу email.php и со страницы email.php на страницу blog.php.
  4. Самостоятельно реализуйте проверку заполнения всех полей формы для того, чтобы исключить отправку «пустого» письма.

**Упражнение 4. Страница для добавления заметок**

В этом упражнении будет проиллюстрировано создание страницы для добавления новых заметок – newnote.php.

* 1. Создайте новую страницу newnote.php, добавьте название и пояснительный текст.
  2. Создать html-форму с именем «newnote», метод обработки данных – POST.
  3. На форме разместите два поля: одно (типа text) для добавления заголовка заметки – с именем «title», другое (textarea) для добавления самой заметки – с именем «article». Добавьте параметры размера элементов формы.
  4. Также поместите на поле кнопку отправки с именем «submit».

*⇒ Не забывайте именовать html-форму и элементы html-формы (атрибут name). Эти имена важны при дальнейшей обработки данных, полученных через форму, в php-скриптах.*

* 1. Добавление даты создания заметки
  2. В таблице notes, заполняемой через создаваемую нами форму, осталось незаполненным поле art\_id (поле с датой создания заметки) – для него мы не создавали элемент формы. PHP позволяет получать текущую дату автоматически, с помощью функции date(). Формат ее вызова: date(<формат>). MySQL требует формат даты <год>-<месяц>-<число>, при этом год – 4 цифры, месяц – 2 цифры, число – 2 цифры. Согласно шаблону, вид вызова функции: date(“Y-m-d”). Мы автоматизируем процесс получения текущей даты из формы.
  3. Разместите на форме после второго текстового поля скрытое поле с именем «created».
  4. Значение поле created будет получать через php- функцию date(). Результат добавления поля:

<input type="hidden" name = "created" id = "created"

value ="<?php echo date("Y-m-d");?>"/>

Вариант реализации html-формы

<html>

<body>

<p>Добавить новую заметку: </p>

<form id="newnote" name="newnote" method="post">

<input type="text" name="title" id="title" size="20" maxlength="20"/>

<textarea name="article" cols="55" rows="10" id=" article"> </textarea>

<input type="hidden" name = "created" id = "created" value ="<?php echo date("Y-m-d");?>"/>

<input type="submit" name="submit" id="submit" value="Отправить" />

</form>

<a href="blog.php">Возврат на главную страницу сайта</a>

</body>

</html>

1. Обработка html-формы. Вам необходимо создать php-скрипт, который выполнит два шага:

• Получит данные, введенные пользователем в поля созданной html-формы (т.е. новую заметку);

• Передаст эти данные в базу, где хранятся уже созданные ранее заметки.

6.1. Получение данных через форму. Для получения данных через форму необходимо:

6.1.1. Подключиться к серверу;

6.1.2. Выбрать базу данных;

6.1.3. Получить данные из полей формы. Данные мы получаем из элементов формы используя названия (атрибут name) этих элементов. Данные формы помещаются в массив $\_POST, а затем присваиваются переменным php. Принцип получения:

$имя\_переменной = $\_POST [‘АтрибутNameЭлементаФормы’];

Таким образом информация, введенная пользователем в форму, «присваивается» в качестве значения для переменной php.

//Получение данных из формы

$title = $\_POST['title'];

$created = $\_POST['created'];

$article = $\_POST[‘article’];

6.2. Передача данных в базу

6.2.1. Данные в базу передаются по обычному принципу: формирование SQL-запроса – реализация

Формирование запроса: (в нем поле id получает свое значение автоматически):

//Формирование запроса

$query = "INSERT INTO notes (title, created, article)

VALUES ('$title', '$created', '$article)";

В запросе используется SQL-инструкция INSERT. Синтаксис инструкции:

INSERT INTO tblName (tblField 1, tblField 2, … , tblField N)

VALUES (value 1, value 2, … , value N);

В ней tblNamt – имя таблицы, tblField – имя поля таблицы (перечисляются в том порядке, в котором располагаются в таблице), value – вставляемое значение поля таблицы (порядок должен соответствовать порядку имен полей).

6.2.2. Реализуйте запрос с помощью функции mysqli\_query().

7. Проверьте корректность работы формы и обработки данных.

8. Самостоятельно программно исключите возможность передачи в базу данных пустой записи.

9. Добавьте гиперссылки между страницами blog.php и newnote.php.

Вариант реализации кода страницы newnote.php

<title>Страница для добавления заметки</title>

</head>

<p>Добавить новую заметку: </p>

<form id="newnote" name="newnote" method="post" action="">

<input type="text" name="title" id="title" size="20" maxlength="20"/>

<textarea name=" article" cols="55" rows="10" id=" article"> </textarea>

<input type="hidden" name = "created" id = "created" value ="<?php echo date("Y-m-d");?>"/>

<input type="submit" name="submit" id="submit" value="Отправить" />

</form>

<a href="blog.php">Возврат на главную страницу сайта</a>

</body>

</html>

<?php

//Подключение к серверу

require\_once ("connections/MySiteDB.php");

//Выбор БД

$select\_db = mysqli\_select\_db ($link, $db);

//Получение данных из формы

$title = $\_POST['title'];

$created = $\_POST['created'];

$article = $\_POST[‘article’];

if (($title)&&($created)&&($article))

{

//Формирование запроса

$query = "INSERT INTO notes (title, created, article) VALUES ('$title',

'$created', '$article’)";

//Реализация запроса

$result = mysqli\_query ($link, $query);

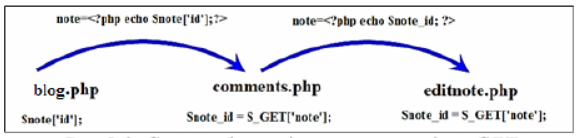
}

?>

**Упражнение 5. Страница для редактирования заметок**

В этом упражнении необходимо создать страницу editnote.php, добавить название и пояснительный текст. Переход на эту страницу будет осуществляться со страницы comments.php (т.к. в начале этой страницы выводится текст комментируемой заметки).

1. Откройте страницу comments.php. Создайте между текстом комментируемой заметки и повторяющейся областью комментариев пустой абзац и введите текст «Изменить заметку». Сделайте ее гиперссылкой для перехода на страницу editnote.php.
2. Гиперссылка на editnote.php. Для передачи информации на страницу editnote.php о том, какая именно заметка модифицируется (заметка с каким id), необходимо передать идентификатор заметки со страницы comments.php в строке URL-адреса через гиперссылку.
3. При его получении на странице editnote.php используется метод GET (принцип работы аналогичен тому, что был использован при передачи идентификатора comments.php),



*Схема обмена данными методом GET*

Дополните гиперссылку со страницы comments.php на страницу editnote.php:

<a href="editnote.php?note=<?php echo $note\_id; ?>">Исправить заметку </a>

4. Работа со страницей editnote.php

4.1. На странице editnote.php создайте «editnote», метод обработки данных – POST.

4.2. На форме разместите два поля: одно (типа text) для изменения заголовка заметки – с именем «title», другое (textarea) для изменения самой заметки – с именем «article». Добавьте параметры размера элементов формы.

4.3. Также поместите на поле кнопку отправки с именем «submit».

4.4. Далее необходимо создать php-скрипт для обработки данных формы. Этот скрипт должен выполнять следующее:

• Отображать редактируемую заметку в полях формы (т.е. помещать

данные из базы в поля формы);

• Получать измененные данные из формы;

• Передавать изменение данные в таблицу.

5. Заполнение полей формы

5.1. Введите переменную $note\_id, которая получит в качестве значения идентификатор обрабатываемой заметки. Это значение она должна получить через массив $\_GET.

5.2. Реализуйте соединение с сервером.

5.3. Выберите базу данных.

5.4. Далее необходимо сформировать запрос на получение заметки с выбранным id из базы данных, для размещения ее в полях формы. Запрос реализуется с помощью оператора SELECT, условием запроса должно быть id выбранной заметки.

5.5. Реализуйте сформированный запрос.

5.6. С помощью функции mysqli\_fetch\_array() поместите результат выполнения запроса (т.е. полученную строку) в массив.

Вариант реализации кода:

<?php

//получение идентификатора

$note\_id = $\_GET['note'];

//Соединение с сервером

require\_once ("connections/MySiteDB.php");

//Выбор БД

$select\_db = mysqli\_select\_db ($link, $db);

//Запрос к БД на получение строки, содержащей заметку с выбранным id

$query = "SELECT \* FROM notes WHERE id = $note\_id";

//Реализация запроса к БД

$result = mysqli\_query ($link, $query);

//Помещение выбранной строки в массив

$edit\_note = mysqli\_fetch\_array ($result);

6. Необходимо, чтобы записи полученной заметки отображались в соответствующих полях формы. Для этого:

6.1. Добавьте на html-форму скрытое поле с именем note (оно будет содержать id заметки).

6.2. В html-форме задаем значение value для всех элементов из массива:

<!-- $edit\_note - это имя массива, в который помещается результат выполнения функции mysqli\_fetch\_array(); -->

<form id="editnote" name="editnote" method="post" action="">

<label for="title">Заголовок заметки</label>

<input type="text" name="title" id="title" value = "<?php echo $edit\_note['title'];?>" />

<label for="article">Текст заметки </label>

<textarea name="article" id=" article">

<?php echo $edit\_note[article’];?></textarea>

<input type="hidden" name = "note" id = "note" value="<?php echo $edit\_note['id']?>" />

<input type="submit" name="submit" id="submit" value="Изменить" />

</form>

1. Получение данных из формы после изменения. Принцип реализации похож на добавление новой заметки:

7.1. Получите из формы измененные данные с помощью метода $\_POST;

7.2. Передайте данные в таблицу с помощью SQL-запроса. Разница заключается только в SQL-запросе - при добавлении используется INSERT, а при обновлении UPDATE.

*⇒ Оператор UPDATE обновляет поля таблицы в соответствии с их новыми значениями в строках. Синтаксис запроса на обновление:*

UPDATE tblName SET fieldName1 = expr1, fieldName2 = expr2, … ,

fieldName N = expr N WHERE …

где tblName – имя таблицы, fieldName = expr - указывается, какие именно поля надо изменить и какими должны быть их новые значения.

Вариант кода получения и передачи данных из формы

<?php

//Собственно обновление данных

//Получение обновленных значений из формы

$title = $\_POST['title'];

$article = $\_POST[‘article’];

//Создание запроса на обновление

$update\_query = "UPDATE notes SET title = '$title', article = '$article’

//Реализация запроса на обновление

$update\_result = mysqli\_query ($link, $update\_query);

?>

1. Проверьте корректность работы скриптов.
2. Создайте гиперссылку для возврата на страницу комментариев.

Вариант полной реализации editnote.php кода

<?php

$note\_id = $\_GET['note'];

require\_once ("connections/MySiteDB.php");

$select\_db = mysqli\_select\_db ($link, $db);

$query = "SELECT \* FROM notes WHERE id = $note\_id";

$result = mysqli\_query ($link, $query);

$edit\_note = mysqli\_fetch\_array ($result);

?>

<html>

<body>

<p>Страница редактирования заметки </p>

<form id="editnote" name="editnote" method="post" >

<label for="title">Заголовок заметки</label>

<input type="text" name="title" id="title"

<label for=" article">Текст заметки </label>

<textarea name=" article" id=" article">

<input type="hidden" name = "note" id = "note"

<input type="submit" name="submit" id="submit" value="Изменить" />

</form>

<a href="blog.php">Вернуться на главную страницу сайта</a>

</body>

</html>

<?php

$title = $\_POST['title'];

$article = $\_POST[‘article’];

$update\_query = "UPDATE notes SET title = '$title', article = '$article’

$update\_result = mysqli\_query ($link, $update\_query);

?>

**Упражнение 6. Создание страницы удаления заметок**

Самостоятельно создайте страницу для удаления заметки deletenote.php. Переход на эту страницу также должен осуществляться со страницы comments.php.

Для реализации удаления записи из БД используется SQL- оператор DELETE. Синтаксис оператора DELETE:

DELETE FROM tblName WHERE …

где tblName – имя таблицы.

Не забудьте реализовать удаление комментариев к удаляемым записям.

**Работа с заметками**

**Упражнение 7. Работа со страницей blog.php**

1. Необходимо сделать так, чтобы последняя добавленная заметка отображалась в самом верху списка заметок. Для отредактируйте код SQL-запроса к БД таким образом, чтобы данные передавались в необходимом порядке. С этой целью используются следующий синтаксис:

SELECT fieldName FROM tblName ORDER BY fieldName order

fieldName – имя поля (полей) таблицы,

tblName – имя таблицы – источника,

order – порядок следования записей. Он может быть ASC – по возрастанию, DESC – по убыванию, RAND – в случайном порядке.

Например:

SELECT \* FROM table ORDER BY name ASC

*т.е. необходимо выбрать все поля из таблицы table и расположить их в порядке возрастания значений поля name (т.е. в алфавитном порядке, если поле строкового типа).*

2. Необходимо на странице deletenote.php сделать ссылку возврата на страницу комментариев. При этом необходимо учитывать, что при переходе по простой ссылке (без дополнительных параметров) заметка все равно удалится, поэтому необходимо в гиперссылке «вернуть» странице комментариев принятые от нее аргументы GET, чтобы она снова вывела ту же самую заметку.

Вариант реализации кода

<?php

$note\_id = $\_GET['note'];

require\_once ("connections/MySiteDB.php");

$select\_db = mysqli\_select\_db ($link, $db);

$query = "SELECT \* FROM notes WHERE id = $note\_id";

$result = mysqli\_query ($link, $query);

$delete\_note = mysqli\_fetch\_array ($result);

?>

<html>

<body>

<p>Страница редактирования заметки </p>

<form id="editnote" name="editnote" method="post" >

<label for="title">Заголовок заметки</label>

<input type="text" name="title" id="title"

<label for="article">Текст заметки </label>

<input type="text" name=" article" id=" article"

<input type="hidden" name = "note" id = "note"

<input type="hidden" name = "MM\_update" value="editnote" />

<input type="submit" name="submit" id="submit" value="Удалить" />

</form>

<?php

$submit = $\_POST['submit'];

if ($submit)

{

$delete\_query = "DELETE FROM notes WHERE id = $note\_id";

$delete\_result = mysqli\_query ($link, $delete\_query);

}

?>

</body>

</html>

**Упражнение 8. Работа с комментариями к заметкам**

Самостоятельно создайте страницу для добавления комментариев к заметкам, а также страницу для удаления комментариев.