How to Set up a Website-Server Connection For Database Integration

Introduction

The integration of a server system within a website can be useful for many reasons. Increasing user interactivity can increase website traffic and make it so that website admins can get more information to build new projects. Moreover, website developers can use this website-server connection to obtain data sent from users. Using this server connection can also be used to make a feedback page which can be useful for fixing bugs and improving existing programs.

Making Your Website

This step is relatively simple, with a little bit of HTML and CSS knowledge, a website should be a breeze. If you do not have any experience in these languages, simply type in the code below into a text editor and read the explanation of the programs.

```
<html>

<head>
        <title>Leave Feedback</title>
</head>

<body style="text-align: center;">

        <a href="Feedback.html">Leave Feedback</a>
        <a href="Devs.html">Devs</a>

        <br>

        <h1 class="title">Leave Feedback</h1>

        <br>

        <h3>Tell Us How You Like (Or Dislike) Our Website!</h3>
        <p>When we get feedback from users like you, we can improve overall user
experience.</p>

        <br>

        <form action="http://localhost/feedbackLogger.php" method="post" target="_blank"
name="form">
                <input name="name" type="text" placeholder="Your name" required/>
                <br>
                <input name="email" type="text" placeholder="Email" required/>
                <br>
                <input size="30" name="header" type="text" placeholder="Subject" required/>
                <br>
                <textarea cols="32" name="message" rows="5" placeholder="Your message"></
textarea>
                <br>
                <input type="submit" target="_blank" value="Submit">
        </form>

        <br>
        <br>
        <br>

</body>

</html>
```

This first document will be for our first website page. This page will be used for user input.

```
<html>

<head>
        <title>Devs</title>
</head>

<body style="text-align: center;">

        <a href="Feedback.html">Leave Feedback</a>
        <a href="Devs.html">Devs</a>

        <br>

        <h1 class="title">Devs</h1>

        <br>

        <img src="Dev.png" style="background-color: white;height: 250px; width: 250px;"
class="imgcenterwithBorder">
        <br>

        <h3>User Feedback Data</h3>
        <button onclick="window.open('http://localhost/feedbackSender.php');">User Data</
button>

        <br>
        <br>
        <br>

</body>

</html>
```

This second document will be for the developer page, which we will use to look at the data received from the server. These websites are not very visually appealing, but they will fulfill our purposes.

Setting up a Local Server

In this situation, we will be using a local server to simplify the process of our website-server connection. I recommend downloading an application such as MAMP, which is the one we will be using in this example. This app will allow you to create a server locally on your computer. Follow their download link here: https://www.mamp.info/en/downloads/.

Creating a Database

Next we will need to write a script to create our database. For this, we will use a PHP file with an integrated SQL script. Remember, all of these files must be saved in the htdocs folder for MAMP to properly run the file.

```php
<?php
// Sets the variables needed to establish server connection
$servername = "localhost";
$username = "root";
$password = "root";
$dbname = "userFeedback";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

// sql to create table
$sql = "ALTER TABLE userData
ADD COLUMN header VARCHAR(30) AFTER email;";

if ($conn->query($sql) === TRUE) {
  echo "new column made successfully";
} else {
  echo "Error making new column: " . $conn->error;
}

// Closes connection with the server
$conn->close();
?>
```

As you can see from the comments (written after “//”) the first chunk of code establishes the variables needed to establish a server connection. In the second chuck, we make a connection with the database. The third chunk is where we are creating the table in our new database. You can now run this program from your local server to create your database. This program can be named “feedbackLogger.php”.

```php
<?php header('Access-Control-Allow-Origin: *');
// Gets variables from user's request. Puts user's data into our variables.
$name = $_POST['name'];
$email = $_POST['email'];
$header = $_POST['header'];
$message = $_POST['message'];

// Makes variables for database connection
$data = array(
    'user' => array('name' => $name, 'email' => $email, 'header' => $header, 'message' =>
$message)
    );

$servername = "localhost";
$username = "root";
$password = "root";
$dbname = "userFeedback";

// Creates connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Checks connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

// Prepares and Adds the User Data into the Database
$stmt = $conn->prepare("INSERT INTO userData (name, email, header, message) VALUES
(?, ?, ?, ?)");
$stmt->bind_param("ssss", $name, $email, $header, $message);

// Executes the Prepare and Add command
$stmt->execute();


echo "New records created successfully";

// Closes connection with the server
$stmt->close();
$conn->close();
?>
```

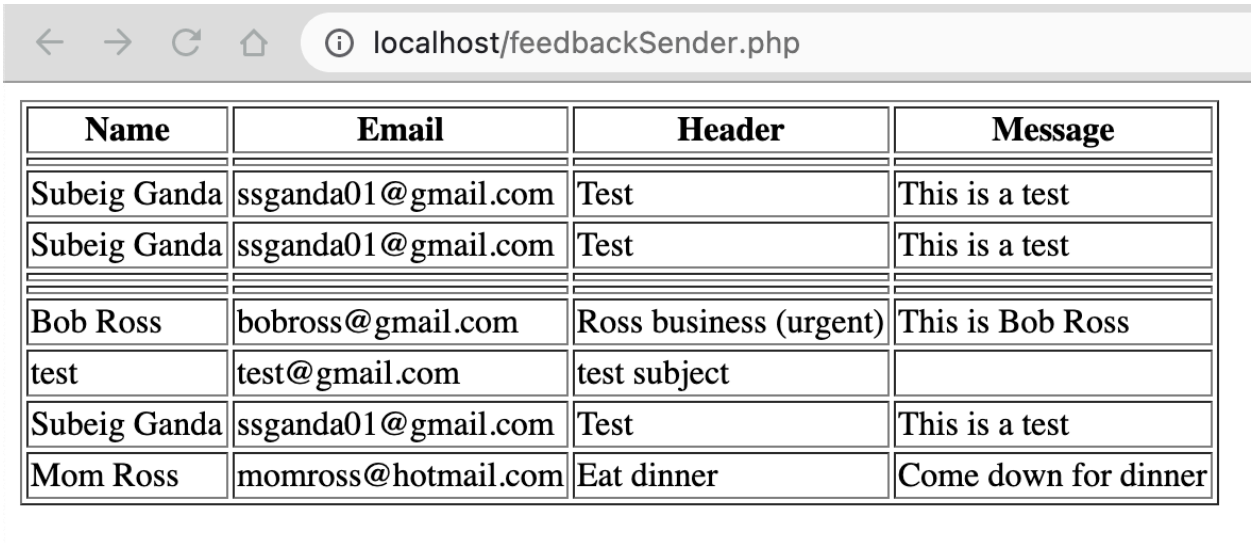
Now let's write the program that will log the user data coming from our website. We set our variables for connecting to the server as previously. This time we will be adding new variables and inputting the user data into them. After creating a connection to the server, using the same code as before, we make a command to log the user information (remember this is stored in the new variables we created). This program can be named "feedbackSender.php". Now fill in this file's address into the Feedback.html document as shown in the example (This address will be in the form http://localhost/feedbackLogger.php).

```php
<?php header('Access-Control-Allow-Origin: *');
// Makes variables for database connection
$servername = "localhost";
$username = "root";
$password = "root";
$dbname = "userFeedback";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

$sql = "SELECT name, email, header, message FROM userData";
$result = $conn->query($sql);

//Puts data into table
echo "<table border='1'>
<tr>
<th>Name</th>
<th>Email</th>
<th>Header</th>
<th>Message</th>
</tr>";

while($row = mysqli_fetch_array($result))
{
echo "<tr>";
echo "<td>" . $row['name'] . "</td>";
echo "<td>" . $row['email'] . "</td>";
echo "<td>" . $row['header'] . "</td>";
echo "<td>" . $row['message'] . "</td>";
echo "</tr>";
}
echo "</table>";

//This code does not work... was meant for creating a table in a html document through json
parsing.
/*$data = array(
        'name' => $row['name'],
        'email' => $row['email'],
        'header' => $row['header'],
        'message' => $row['message']
);

$myJSON = json_encode($data)
echo $myJSON;*/

// Closes connection with the server
$conn->close();
?>
```

Finally, we must make a program that displays the data from the database in a neat and orderly table format. The first three chunks of this program should be familiar by now: creating variables needed for connecting to the server, creating a connection, and making sure that connection is proper. In the next chunk, we will be accessing our data table, which is located in our database. This table contains all of the information that users have submitted to our website feedback page. The last section of this code takes our data and inputs it into a table that can easily be read by website admins and developers. Now fill in this file's address into the Devs.html document as shown below (This address will be in the form http://localhost/feedbackSender.php).

Testing the Programs

The website-server connection should be working now and your database should be storing user input. To test this, make sure your local server is turned on, and try submitting some feedback on the Feedback.html page. On clicking submit, you should be taken to a page saying "New records created successfully". If you do not receive that message, make sure your server is on and try again. Now go to the Devs.html page and click on the user data button. This should take you a page showing the user data you just inputted from the Feedback.html page. The data will be ordered in a neat table format as shown below. Recall, if this does not work, try checking whether your server is turned on, and check if the file locations are labeled correctly in your code ("feedbackLogger.php" format).



Conclusion

The website-server connection now runs, and our database is collecting and sending us user data. Website data can get a lot more complex, but this is a great first step to making a website more useful. User interactivity is one of the most sought-after things by web developers which makes this an important skill to have in your back pocket.