Damien Gilmore

FirmStep Coding Test

Technologies: CSS, JavaScript, PHP, MySQL.

Framework. Bootstrap.

To complete this assignment I used the Bootstrap Front end framework for speed of development and consistency. Using the bootstrap grid system I chose two columns on one row. Bootstrap offers many advantages including button/input groups and dropdown menus in addition to the more traditional CSS tables, grids, buttons and forms. This makes it much more straightforward to design a clean and simple layout.

Column one of the page index.PHP contains the entry forms which allow the user to enter data,(Customer name, tile etc.) as well as buttons representing the different types of clients visiting the office and radio buttons which allow the user to select the service. (Housing, Council Tax, etc.).

JavaScript was used to write functions containing code to be executed after onClick events allowing the content of the page to be changed dynamically based on user input.

When a user chooses a Customer type by clicking a button such as “Organisation” the onClick event associated with that button makes a call to a JavaScript function and passes a parameter.

onclick="showhideCitizen(2)"

Function showHideCitizen() accepts this parameter which corresponds to the users choice and changes the visible input fields accordingly though the use of simple if else conditional statements.

Function showhideCitizen(param)

{

if(param == 3)

{…………….}

else if(param == 2)

{…………………………

addRecordDB() is a JavaScript function containing PHP code which opens a connection to the database and inserts a new record containing all the user input captured by PHP’s post method, which collects the data present in the input fields of the form.

<form method="post">

User input, which is stored in the value fields of the form input elements is parsed into PHP variables:

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

$Service = test\_input($\_POST["optradio"]);

$Type = test\_input($\_POST["CustType"]);

$Title = test\_input($\_POST["titleOption"]);

$FirstName = test\_input($\_POST["fname"]);

$LastName = test\_input($\_POST["sname"]);

}

A connection to the database is then established :

try {

$conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);

A MySQL Statement is stored in a PHP variable:

$sql = "INSERT INTO WalkInQueue(Type, Title, FirstName, Lastname, Service)

VALUES ('$Type', '$Title' , '$FirstName' , '$LastName' , '$Service')";

Finally the PHP variable $sql is passed to the database where the MySQL statement is executed and a new record is created in the table “WalkInQueue”.

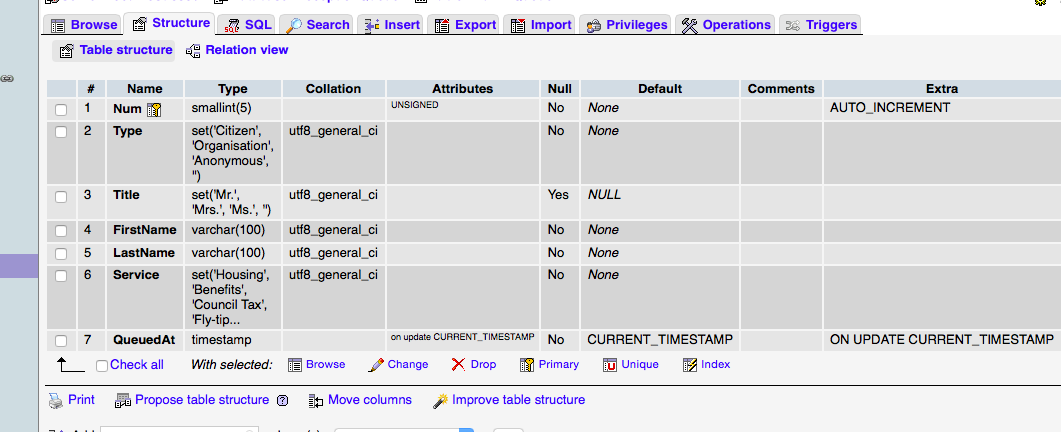
PHP was also used to read records from the database and display them in a HTML table in the second column on the main page.

A MySQL database containing one table “WalkInQueue”, was created using myPHPAdmin. This table contains 7 columns.

The First Column is an auto increment column which is used to number the records in the table.

The next five columns correspond to user entered data such as Customer type, Service requested and Personal customer details.

The last column is a timestamp column which is used to track the time a customer joined the queue. On insertion of a new record this column is automatically filled in with the current time.



There are several issues with this code which are unresolved, namely the PHP code inserting empty records into the database on each page reload. I am assuming that this is due to PHP being executed on the server. By the time the page has loaded on the client side the PHP code has already been processed and stripped away and an empty record has been created in the database . Additionally PHP’s POST method does not capture values associated with simple html buttons and as such the customer type field remains empty . I have done some research online and it appears that Ajax may be the solution to these issues, however I have not at this stage had any experience with this and as such I haven’t solved these issues at this time.

Finally the timestamp values are stored as Year:Month:date:time.

|  |
| --- |
| 2017-03-22 23:20:19 |

I have found some code snippets which appear to trim away the Year:Month:date:

Part of this field:

date("H:i:s",strtotime($time));

I have also managed to get access to individual column entries in the table by binding columns in the result set to PHP variables :

$result = $stmt->setFetchMode(PDO::FETCH\_BOUND);

$stmt->bindColumn('Num', $Num);

$stmt->bindColumn('Type', $CustType);

$stmt->bindColumn('Title', $CustTitle);

$stmt->bindColumn('FirstName', $FName);

$stmt->bindColumn('LastName', $LName);

$stmt->bindColumn('Service', $ServiceType);

$stmt->bindColumn('QueuedAt', $Time);

Unfortunately I have not at this stage managed to get this working satisfactorily and as such this code is not included in my solution.