HTML and CSS - Text lecture
Use this link to learn/build a site using HTML/CSS ->
http://learn.shayhowe.com/html-css/

Assignment: Once complete you can either paste images of this completed site to the Q & A section, or you can create your own static site and post pictures of that. Good luck!

Project start - Text lecture
Right click on workspace folder and select the option to create a new folder
Name the folder FormCollector
Right click on the FormCollector folder and select 'New File', name it OptIn.html
We are going to put static HTML form in this file to collect information like name and email address. This could be used as part of a larger website that is asking people to join the mailing list.
HTML is fairly easy to read. Everything has "tags" which are words or phrases inside of <> and </> to show the beginning and end of that tag. For Example, at the beginning of an HTML document, we have the tag <HTML> meaning anything after this and prior to </HTML> is html code. Similarly, we have a Header section of the HTML marked by <HEAD> and </HEAD>
http://www.w3schools.com/html/ is a great resource when referencing HTML, along with http://learn.shayhowe.com/html-css/
Something handy to know: Commenting stuff out is done by adding <!-- add comment here -->
Open OptIn.html, paste the following

```
<HTML>
  <HEAD>
  </HEAD>
  <BODY>
    <h1>Sign Up</h1>
  </BODY>
</HTML>
```

Save and close.
Run it by clicking on run project and going to the screen.

Notice that “Sign Up” is in a pretty big font due to H1 tag.

Now let's add a tag for title.
Add the title tag between the HEAD tags
```
<TITLE>OptIn</TITLE>
```
Now add the form tag
Place the following below the <h1>Sign Up</h1>
```
    <form action="receiver.php" method = "POST" >
      <input type="submit" class='button'>
    </form>
```
We haven’t created it yet, but when the form is submitted, it will send the contents of the form to the form action page. In this case “reciever.php”
Test it out.
Notice that we have a method of POST. This can be thought of as we are submitting/posting information to the other page. There is another method called GET, but for our purposes, POST works well.

Notice that the submit button is generated by this:
```
<input type="submit" class='button'>
```
All of the information we send in the form needs to be part of an input.

To collect the information, we add Inputs for First Name, Last Name, and Email address prior to the submit button.
```
<p>First Name:  <input type="text" name="FNAME"
maxlength='100'  required/></p>
<p>Last Name:  <input type="text" name="LNAME"
maxlength='100'  required/></p>
<p>Email: <input type="text" name="EMAIL"
maxlength='100'  required/></p>
```
The <p> tag is for paragraph and it simply creates some space between the inputs.
“First Name: “, “Last Name: “, and “Email: “ are simply labels
“required” is an HTML5 setting that requires the fields to be filled out before the form will submit.
“maxlength” prevents users from entering too much data. Depending on what we plan to do with the input, this length will vary.
The name of “FNAME” or “LNAME” is what the input will be named when the form is submitted.

Type sets the type of input. There are a number of types, test types, submit types and lots of others.
Another type is “hidden” which simply is an input that the user won’t see, but will be sent as part of the form.
Test this out.
Let's add some hidden inputs for use later on in the project:

```
<input type="hidden" name="formID" value="OptIn" />
<input type="hidden" name="successredirecturl" value="Success.html">
<input type="hidden" name="rejectredirecturl" value="Fail.html">
```

At this point, the full file should be as follows:

```
<HTML>
  <HEAD>
    <TITLE>OptIns</TITLE>
  </HEAD>
  <BODY>
    <h1>Sign Up</h1>
      <form action="receiver.php" method = "POST">
        <input type="hidden" name="formID" value="OptIn" />
        <input type="hidden" name="successredirecturl"
value="Success.html">
        <input type="hidden" name="rejectredirecturl" value="Fail.html">
        <p>First Name:  <input type="text" name="FNAME"
maxlength='100'  required/></p>
        <p>Last Name:  <input type="text" name="LNAME"
maxlength='100'  required/></p>
        <p>Email: <input type="text" name="EMAIL"
maxlength='100'  required/></p>
        <input type="submit" class='button'>
      </form>
  </BODY>
</HTML>
```

The plan for the successredirecturl and rejectredirecturl is that our application will send the user to the predefined page depending on the outcome of the form submission.

Add styling to form - Text follow-up
Add the following to OptIn.html directly under the TITLE tags

```
 <link href="styles.css" rel="stylesheet" type="text/css">
```

Something else we need to add to our HTML are div tags and classes. This is simply a way to connect the CSS styles to the HTML code. Div is simply a section to apply the settings corresponding to the class. Update the OptIn.HTML to the following

```
<HTML>
  <HEAD>
    <TITLE>OptIns</TITLE>
    <link href="styles.css" rel="stylesheet" type="text/css">
  </HEAD>
  <BODY>
    <div class="backgrounds">
      <div class="main-content">
        <h1>Sign Up</h1>
        <form action="receiver.php" method = "POST">
          <input type="hidden" name="formID" value="OptIn" />
          <input type="hidden" name="successredirecturl"
value="Success.html">
          <input type="hidden" name="rejectredirecturl" value="Fail.html">
          <p>First Name:  <input type="text" name="FNAME"
maxlength='100'  required/></p>
          <p>Last Name:  <input type="text" name="LNAME"
maxlength='100'  required/></p>
          <p>Email: <input type="text" name="EMAIL"
maxlength='100'  required/></p>
          <input type="submit" class='button'>
        </form>
      </div><!--main-content-->
    </div><!--background-->
  </BODY>
</HTML>
```

Create a new file named styles.css to add the styling
This file is where you specify colors, sizes, fonts and most things related to styling HTML.
We are going to use this style sheet to set a background color, border, make the button look cool, etc.

Check out this page to generate custom HTML and CSS code for buttons:
http://livetools.uiparade.com/button-builder.html
Here are the full contents of the CSS file:
*** Caution. Your browser sometimes stores a version of the CSS and reuses that instead of pulling in updates. You may have to clear your "Cached images and files" to see the changes.

```
body
{
background-color:#000000;
background-repeat:repeat-x;
background-position:center top;
}
.backgrounds {
  position: relative;
  margin: auto;
  border-style: solid;
   border-color: red;
   padding: 1% 1% 1% 1%; /*top right bottom left */
  /*background-image:url(images/background.png); */
  background-repeat: no-repeat;
  background-size: contain;
  background-color: #4d4d4d;
  max-width: 1345px;
  background-position: top;
  vertical-align: text-top;
  height: auto;
}
.main-content {
   position:relative;
   border-style: solid;
  border-color:red;
  background-color:rgba(0, 0, 0, 0.8);
  height: auto;
  text-align: center;
  color: #ffffff;
  font-size: 150%;
  padding: 5px 5px 5px 5px;
}
```

```
.button {
   border: 1px solid #401818;
   background: #9c3e3e;
   background: -webkit-gradient(linear, left top, left bottom, from(#d66565),
to(#9c3e3e));
   background: -webkit-linear-gradient(top, #d66565, #9c3e3e);
   background: -moz-linear-gradient(top, #d66565, #9c3e3e);
   background: -ms-linear-gradient(top, #d66565, #9c3e3e);
   background: -o-linear-gradient(top, #d66565, #9c3e3e);
   background-image: -ms-linear-gradient(top, #d66565 0%, #9c3e3e
100%);
   padding: 10px 20px;
   -webkit-border-radius: 11px;
   -moz-border-radius: 11px;
   border-radius: 11px;
   -webkit-box-shadow: rgba(255,255,255,0.4) 0 1px 0, inset
rgba(255,255,255,0.4) 0 1px 0;
   -moz-box-shadow: rgba(255,255,255,0.4) 0 1px 0, inset
rgba(255,255,255,0.4) 0 1px 0;
   box-shadow: rgba(255,255,255,0.4) 0 1px 0, inset rgba(255,255,255,0.4) 0
1px 0;
   text-shadow: #bd7f7f 0 1px 0;
   color: #ffffff;
   font-size: 17px;
   font-family: helvetica, serif;
   text-decoration: none;
   vertical-align: middle;
   }
.button:hover {
   border: 1px solid #401818;
   text-shadow: #571e1e 0 1px 0;
   background: #9c3e3e;
   background: -webkit-gradient(linear, left top, left bottom, from(#d66565),
to(#9c3e3e));
   background: -webkit-linear-gradient(top, #d66565, #9c3e3e);
   background: -moz-linear-gradient(top, #d66565, #9c3e3e);
   background: -ms-linear-gradient(top, #d66565, #9c3e3e);
   background: -o-linear-gradient(top, #d66565, #9c3e3e);
   background-image: -ms-linear-gradient(top, #d66565 0%, #9c3e3e
100%);
   color: #fff;
   }
.button:active {
   text-shadow: #571e1e 0 1px 0;
   border: 1px solid #401818;
   background: #d66565;
   background: -webkit-gradient(linear, left top, left bottom, from(#9c3e3e),
to(#9c3e3e));
   background: -webkit-linear-gradient(top, #9c3e3e, #d66565);
   background: -moz-linear-gradient(top, #9c3e3e, #d66565);
   background: -ms-linear-gradient(top, #9c3e3e, #d66565);
   background: -o-linear-gradient(top, #9c3e3e, #d66565);
   background-image: -ms-linear-gradient(top, #9c3e3e 0%, #d66565
100%);
   color: #fff;
   }
```
Paste the code into styles.css, save it and close.

Now test it by visiting the page!

The plan now is to have the user fill out the form, then hit Submit. The HTML form will reach out to the receiver.php file with all the information from the form, with an IP, and what to do in the case of a success or failure. If it’s successful, reciever.php will redirect the user to the success page, and if there is an issue, the user will be directed to the Fail.html page.

To create the success and fail pages:
Right click on FormCollector folder and create both Success.html and Fail.html files.
Inside Fail.html, simply type the word “Fail!”, then save, test it in browser and close
Inside Success.html, type the word “Success!”, then save, test it in browser and close

Create table - Text follow-up

Create a database, use it and then create a table from mysql cli:

```
1) CREATE DATABASE formcollector;

2) use formcollector;

3) CREATE TABLE OptIn ( OptInID INT NOT NULL AUTO_INCREMENT
                  ,formID varchar(50) DEFAULT NULL
                  ,NAME varchar(100) DEFAULT NULL
                  ,EMAIL varchar(100) DEFAULT NULL
                  ,ENTRY_TimeStamp datetime DEFAULT NULL
                  ,SOURCE_IP char(15) DEFAULT NULL
                  ,successredirecturl char(200) DEFAULT NULL
                  ,rejectredirecturl char(200) DEFAULT NULL
                  ,PRIMARY KEY (OptInID));
```

Our PHP scripts will need to connect to the database, so the easiest way to maintain the connection information is using a config file. So create a config.php file for the application connection information under the formcollector folder

Config and other php - Text follow-up

Update config.php with the following code:

```
<?php
/*Configuration Settings*/
define('DB_HOST', localhost'); /*Database Server*/
define('DB_NAME', 'formcollector'); /*Database Name*/
define('DB_USER', 'root'); /*Database Username*/
define('DB_PWD',  ''); /*Database Password*/
?>
```

This information could have been stored with the rest of the application, but separating it out into it’s own file simplifies things when moving the application to a new location.

Create another file named receiver.php
Put the following contents in it:

```
<?php
```

```
/*Bring in our custom library of php and functions, we will create this next*/
include('SQLFunctions.php');
/*First we are going to handle what is being submitted in the form by
displaying it, so we display what is in the _POST*/
/*echo simply means write the following out to the page, adding this
description and the line breaks will make the output legible */
/*The echo statements aren’t actually needed, but they are really helpful in
troubleshooting*/
  echo '<br>Display full contents of the _POST: <br>';
/*var_dump is a function that write all of names and values from the post*/
  var_dump($_POST);
/*We need to esatblish a connection with the database
   f_sqlConnect() is a function we will create in SLQFunctions.php and will to
the database based on config.php file settings. $link is a variable where we
store the connection so we can reference it*/
  $link = f_sqlConnect();

/*We are going to pull out the destination table for this data*/
/*isset() is a function that checks whether or not anything is stored under
the name formID from $_POST, if so, we are storing that as the table
name*/
    if(isset($_POST['formID'])){ $table = $_POST['formID']; }
    echo '<br>Destination Table: ' . $table;
/*We are going to divide the values from the form in an array that we will
use later*/
/*implode a function that reads through _POST and divides the names into
an array, we will call it keys, and then we use it again for the _POST values,
put them into an array call values.*/
  $keys = implode(", ", (array_keys($_POST)));
  echo '<br>Parsed Key :'.$keys;
  $values = implode("', '", (array_values($_POST)));
  echo '<br>Parsed Values :'.$values;
/*We now want to record the source IP as well as date/time
PHP offers some really cool features like detecting the IP that the _POST
came from and recording the time that the transaction happened.*/
  $x_fields = 'ENTRY_TimeStamp, SOURCE_IP';
  $x_values = date("Y-m-d H:i:s") . "', '" . f_getIP();  /*f_getIP() is a custom
```

```
function from SQLFunctions.php*/
  echo '<br>x_values :'.$x_values;
/*Then we want to confirm that the destination table exists prior to actually
attempting to insert the data.  We use another custom function from
SQLFunctions.php, f_tableExists*/
/*check to see if the table exists*/
  if (!f_tableExists($link, $table, DB_NAME) ) {
    die('<br>Destination Table Does Not Exist:'.$table);
  }
/*Grab the success and reject URLs from the _POST and store them in
variables that we can use later.*/
  if(isset($_POST['rejectredirecturl'])){
    $rejectredirecturl = $_POST['rejectredirecturl'];
    echo '<br>rejectredirecturl :'.$rejectredirecturl;
  }
  if(isset($_POST['successredirecturl'])){
    $successredirecturl = $_POST['successredirecturl'];
    echo '<br>successredirecturl :'.$successredirecturl;
  }
/*We now assemble the SQL that will insert our values into the database.*/
  $sql="INSERT INTO $table ($keys, $x_fields) VALUES ('$values',
'$x_values')";
  echo '<br>sql :'.$sql;
/*We attempt the SQL Insert, mysqli_query() actually tries to execute the sql
against the database connection provided. mysqli_error() is a function that
stores any MySQL errors that occurred when the script was run*/
  if (!mysqli_query($link,$sql)) {
    echo '<br>Error: ' . mysqli_error($link);
    if (!empty ($rejectredirecturl)) {
                  /*header Location with a url will redirect the user to the url
specified.
                   We have this row commented out so we can review the
troubleshooting
                   text written above*/
                  /*header("Location: $rejectredirecturl?msg=1");*/
               }
  }else if (!empty ($successredirecturl)) {
```

```
                /*header("Location: $successredirecturl?msg=1");*/
              }
/*Lastly - It’s always good practice to close the database connection*/
  mysqli_close ($link);
?>
```

Complete receiver - Text follow-up
All the code for this file was provided in the text lecture for the prior video, please reference that for comparisons.

Start SQL Functions - Text lecture
Create another file named SQLFunctions.php

Put the following in SQLFunctions.php

```
<?php
include('config.php');
function f_sqlConnect() {
  /*mysqli is a php function that requires the database hostname, database
name, username and password in order to create a database connection*/
  $link = new mysqli(DB_HOST, DB_USER, DB_PWD, DB_NAME);
  if ($link->connect_error) {
              /*if an error occurs while establishing the connection, stop
processing and
              write out an error message*/
    die("Connection failed: " . $link->connect_error);
  }
  echo "<br>Connected successfully to the database<br><br>";
/*return defines what gets sent back when the function completes.  In this
case, a database connection variable named $link*/
  return $link;
}
/*This function is to check if the IP identified is in the correct format and
not in a non-routable range.*/
function f_validIP($ip) {
       /*!empty($ip) checks if $ip is populated*/
       /*ip2long($ip) is a form of validation on the input*/
  if (!empty($ip) && ip2long($ip)!=-1) {
              /*Create an array of arrays standard non routable IPs so we can
filter
              out IPs that aren't useful*/
    $reserved_ips = array (
      array('0.0.0.0','2.255.255.255'),
      array('10.0.0.0','10.255.255.255'),
                array('127.0.0.0','127.255.255.255'),
                array('169.254.0.0','169.254.255.255'),
                array('172.16.0.0','172.31.255.255'),
                array('192.0.2.0','192.0.2.255'),
                array('192.168.0.0','192.168.255.255'),
                array('255.255.255.0','255.255.255.255')
    );
              /*Compare the IP to each array and return a false if the IP is within
any
              of the ranges*/
    foreach ($reserved_ips as $r) {
      $min = ip2long($r[0]);
      $max = ip2long($r[1]);
      if ((ip2long($ip) >= $min) && (ip2long($ip) <= $max)) return false;
    }
              /*if the ip is populated and isn't in the non routable range, return
true*/
    return true;
  } else {
      return false;
  }
}
/*This function attempts various methods to get a valid IP, checking each
with the f_validIP function.  */
function f_getIP() {
  if (f_validIP($_SERVER["HTTP_CLIENT_IP"])) {
    return $_SERVER["HTTP_CLIENT_IP"];
  }
  foreach (explode(",",$_SERVER["HTTP_X_FORWARDED_FOR"]) as $ip) {
    if (f_validIP(trim($ip))) {
      return $ip;
      }
    }
  if (f_validIP($SERVER["HTTP_X_FORWARDED"])) {
    return $_SERVER["HTTP_X_FORWARDED"];
  } elseif (f_validIP($_SERVER["HTTP_FORWARDED_FOR"])) {
    return $_SERVER["HTTP_FORWARDED_FOR"];
  } elseif (f_validIP($_SERVER["HTTP_FORWARDED"])) {
    return $_SERVER["HTTP_FORWARDED"];
  } elseif (f_validIP($_SERVER["HTTP_X_FORWARDED"])) {
    return $_SERVER["HTTP_X_FORWARDED"];
  } else {
    return $_SERVER["REMOTE_ADDR"];
  }
}
/*Check if table exists mysqli_query is a standard php function that requires
the following inputs:connection, query, and resultmode*/
/*with $database = false, we create a new variable named $database and
set it to false*/
function f_tableExists(mysqli $link, $tablename, $database = false) {
  if(!$database) {
    $res = mysqli_query($link, "SELECT DATABASE()");
    $database = mysql_result($res, 0);
  }
  $res = mysqli_query($link," SELECT *
                                    FROM information_schema.tables
                                    WHERE table_schema = '$database'
                                    AND table_name = '$tablename'"
            );
  echo '<br>Table Exist:'.($res->num_rows);
  return $res->num_rows == 1;
}
?>
```

Preview of web app - Text follow-up

Run the project go to OptIn page and try out some entries

You may run into an error if the FNAME and LNAME fields aren't set up in your db. Your challenge is to drop the OptIn table and create a new one, with the update FNAME and LNAME in place of NAME. Then it should work, if not, check the syntax for errors in receiver.php and SQLFunctions.php and the naming.

You may also want to update SELECT * to SELECT COUNT(*) as count in the f_tableExists function of the SQLFunctions.php file

Add browser display - Text follow-up
Create a new file titled OptInDisplay.php

```
<?php
include('SQLFunctions.php');
/*Open the database connection based on config.php file settings*/
  $link = f_sqlConnect();

/*Set Source Table*/
  $table = 'OptIn';
  echo '<br>Source Table: ' . $table;

/*check to see if the table exists*/
  if (!f_tableExists($link, $table, DB_NAME) ) {
    die('<br>Destination Table Does Not Exist:'.$table);
  }
/*Query contents of source table*/
  $sql="SELECT * FROM $table ";
  echo '<br>sql :'.$sql;
    if ($result = mysqli_query($link,$sql)){
      echo "<table border=''1'' style=''width:100%''>";
        //header
        echo "<tr>";
          echo "<td>OptInID</td>";
          echo "<td>formID</td>";
          echo "<td>First Name</td>";
          echo "<td>Last Name</td>";
          echo "<td>Email Addess</td>";
          echo "<td>Date/Time</td>";
          echo "<td>Source IP</td>";
          echo "<td>SuccessUrl</td>";
          echo "<td>RejectUrl</td>";
        echo "</tr>";

      //for each row returned in the query, separate them into <td> tags
      while ($row = mysqli_fetch_array($result))  {
        echo "<tr>";
          echo "<td>{$row[0]}</td>";
          echo "<td>{$row[1]}</td>";
          echo "<td>{$row[2]}</td>";
          echo "<td>{$row[3]}</td>";
          echo "<td>{$row[4]}</td>";
          echo "<td>{$row[5]}</td>";
          echo "<td>{$row[6]}</td>";
          echo "<td>{$row[7]}</td>";
          echo "<td>{$row[8]}</td>";
        echo "</tr>";
      }
      echo "</table>";
    }
    /*mysqli_free_result() clears the $result variable*/
    mysqli_free_result($result);
/*display any sql error encountered */
  if (mysqli_error($link)) {
    echo '<br>Error: ' . mysqli_error($link);
  }else echo '<br>Success';
/*Close database connection*/
  mysqli_close ( $link );

?>
```

Project - Solution text

Copy contents of OptIn.html into a new file, I used PoliticalPoll.html

```
<!DOCTYPE html>
<HTML>
  <HEAD>
    <TITLE>Political Poll</TITLE>
    <META http-equiv=Content-Type content="text/html; charset=utf-8">
    <link href="styles.css" rel="stylesheet" type="text/css">
<script type="text/javascript">
function validateForm(){
  return true;
};
</script>
  </HEAD>
  <BODY>
      <div class="backgroundimage" >
      <div id="main-content" class="main-content" >
    <h1>Politics</h1>
      <form action="receiver.php" method = "POST" onsubmit='return
validateForm()' />
        <input type="hidden" name="formID" value="PoliticalPoll" />
        <input type="hidden" name="successredirecturl"
value="Success.html">
        <input type="hidden" name="rejectredirecturl" value="Fail.html">
        <p>Gender:
          <select name="GENDER" required>
            <option value="">Please Select</option>
            <option value="Male">Male</option>
            <option value="Female">Female</option>
            <option value="Other" >Other</option>
          </select>
        </p>
        <p>Age:  <input type="number" name="AGE" min="0"
max="110"  required/></p>
        <p>How often do you vote:
          <select name="VOTE_FREQ" required>
            <option value="">Please Select</option>
            <option value="Always">Always</option>
            <option value="Sometimes">Sometimes</option>
            <option value="Never" >Never</option>
          </select>
        </p>
        <p>Political Party:
          <select name="PARTY" required>
            <option value="">Please Select</option>
            <option value="Republican">Republican</option>
            <option value="Democrat">Democrat</option>
            <option value="Libertarian">Libertarian</option>
            <option value="Not Listed">Not Listed</option>
            <option value="Not Affiliated">Not Affiliated</option>
          </select>
        </p>
        <p>Who will you vote for:
          <select name="CANDIDATE" required>
            <option value="">Please Select</option>
            <option value="Bernie">Bernie</option>
            <option value="Hillary">Hillary</option>
            <option value="Trump">Trump</option>
            <option value="Cruz">Cruz</option>
            <option value="Rubio">Rubio</option>
          </select>
        </p>
        <br>
        <input type="submit" class='button'>
      </form>
    </div>
  </div>
  </BODY>
</HTML>
```

Create the Table:

```
CREATE TABLE PoliticalPoll ( PoliticalPollID INT NOT NULL
AUTO_INCREMENT
                            ,formID varchar(20) DEFAULT NULL
                            ,GENDER varchar(20) DEFAULT NULL
                            ,AGE varchar(20) DEFAULT NULL
                            ,VOTE_FREQ varchar(20) DEFAULT NULL
                            ,PARTY varchar(20) DEFAULT NULL
                            ,CANDIDATE varchar(20) DEFAULT NULL
```

```
                              ,ENTRY_TimeStamp datetime DEFAULT NULL
                              ,SOURCE_IP char(15) DEFAULT NULL
                              ,successredirecturl char(200) DEFAULT NULL
                              ,rejectredirecturl char(200) DEFAULT NULL
                              ,PRIMARY KEY (PoliticalPollID));
```

For browser viewing of table data copy over OptInDisplay.php and make necessary updates to column and row data

OptInDisplayPoll.php

```
<?php
include('SQLFunctions.php');
/*Open the database connection based on config.php file settings*/
  $link = f_sqlConnect();

/*Set Source Table*/
    $table = 'PoliticalPoll';
  echo '<br>Source Table: ' . $table;

/*check to see if the table exists*/
  if (!f_tableExists($link, $table, DB_NAME) ) {
    die('<br>Destination Table Does Not Exist:'.$table);
  }
/*Query contents of source table*/
  $sql="SELECT * FROM $table ";
  echo '<br>sql :'.$sql;
    if ($result = mysqli_query($link,$sql)){
      echo "<table border=''1'' style=''width:100%''>";
        //header
        echo "<tr>";
          echo "<td>PoliticalPollID</td>";
          echo "<td>formID</td>";
          echo "<td>Gender</td>";
          echo "<td>Age</td>";
          echo "<td>Voting Frequency</td>";
          echo "<td>Party</td>";
          echo "<td>Cadidate</td>";
          echo "<td>EntryTimeStamp</td>";
          echo "<td>Source IP</td>";
          echo "<td>SuccessUrl</td>";
          echo "<td>RejectUrl</td>";
        echo "</tr>";

      //data
      while ($row = mysqli_fetch_array($result))  {
        echo "<tr>";
          echo "<td>{$row[0]}</td>";
          echo "<td>{$row[1]}</td>";
          echo "<td>{$row[2]}</td>";
          echo "<td>{$row[3]}</td>";
          echo "<td>{$row[4]}</td>";
          echo "<td>{$row[5]}</td>";
          echo "<td>{$row[6]}</td>";
          echo "<td>{$row[7]}</td>";
          echo "<td>{$row[8]}</td>";
          echo "<td>{$row[9]}</td>";
          echo "<td>{$row[10]}</td>";
        echo "</tr>";
      }
      echo "</table>";
    }
    mysqli_free_result($result);
  if (mysqli_error($link)) {
    echo '<br>Error: ' . mysqli_error($link);
  }else echo '<br>Success';
/*Close database connection*/
  mysqli_close ( $link );

?>
```