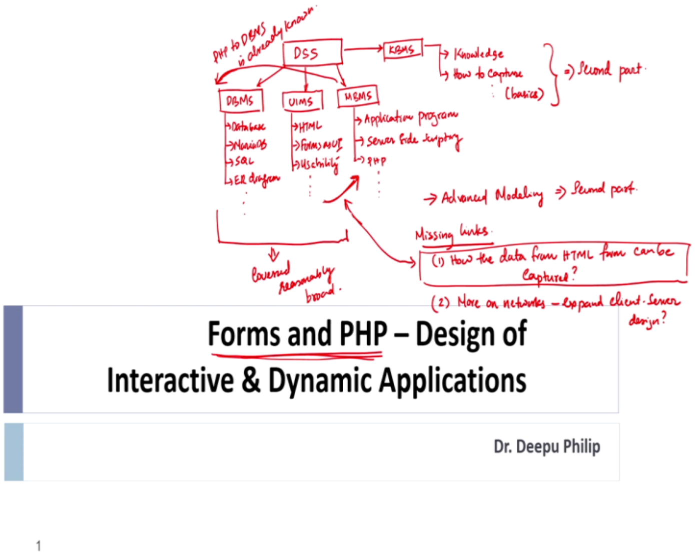
Computer Aided Decision Systems Industrial Practices using Big Analytics
Professor Deepu Philip
Department of Industrial & Management Engineering
Indian Institute of Technology Kanpur
Professor Amandeep Singh
Imagineering Laboratory
Indian Institute of Technology Kanpur

Lecture 49

Forms and PHP - Design of Interactive & Dynamic Applications

Good afternoon, everyone. Welcome to the final lecture of the course, invokes the title Web-based Decision Support System for engineers and practitioners. And, I am Sorry, I apologize for my whole voice because I was not very well for a few weeks and I just got my voice back. So, kindly bear with me in this lecture for this villain-like voice. But I will try to do my level best in this process.

(Refer Slide Time: 00:48)



Forms and PHP – Design of Interactive & Dynamic Applications

Dr. Deepu Philip

1

So, today, we are going to look into the forms and PHP if you look at the title. So, just as a quick recap, because there were a couple of major questions that people had. We were looking at the course of the Decision Support System.

And, I have been doing mostly the technicalities of DSS and we talked about four aspects of the Decision Support System. Number one we had was the DBMS (Database Management System) then, we had what we call the UIMS (User Interface Management System) then, we have what we call the MBMS (Model-based Management System) and then, we also have what we call the

KBMS (Knowledge-based Management System). And, as part of DBMS, we are seeing what a database is and we are seeing what MariaDB is and we are seeing what SQL is and we also see an ER diagram etcetera, the design of the database. So, we have seen so many components of this.

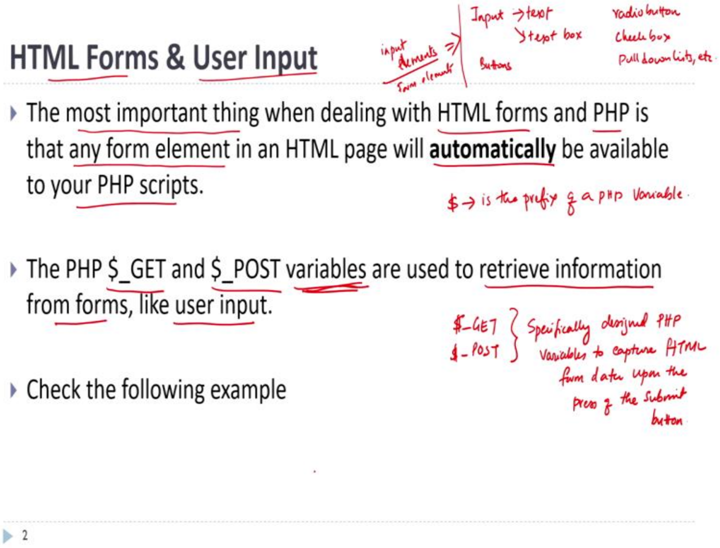
In UIMS (User Interface Management System), we have studied HTML then, forms as UI, we studied about Usability, etcetera as part of this. And, in Model-based Management Systems we kind of study towards the Application Program, Server-side Scripting, we studied PHP, etcetera. And then, the Knowledge-based Management System we started studying, we started looking at something simple called what are Models or what we call as instead of Models? We can call it knowledge and how to capture the very basics of it, that is all we study. Advanced of this portion will be in the second part of this course, the next part of this course, the Advanced Modeling. It will also be the second part. Because we did not have too much time.

These two we have covered reasonably broad. So, two major points that we missed as part of this whole thing are the missing links for us today. The number one question is, how can the data from HTML form be captured? This is the one question. So, that is what first we will see.

The second question we will have been more on how networks expand client server design? So, these are some questions based, asked by a lot of people. Why did we choose the Server-side Model and stuff like that? So, in the first lecture, today's lecture we are going to focus on how the data from the HTML form can be captured.

So, that is why it is called Forms and PHP. So, this HTML form gets captured to PHP and then, from PHP it goes into the DBMS. So, we already saw how this transition from a model base to PHP. So, how the PHP to DBMS is already known. We know this. We are going to study this missing link which is this portion of how this data from HTML form is captured into PHP.

(Refer Slide Time: 05:16)



HTML Forms & User Input

- The most important thing when dealing with HTML forms and PHP is that any form element in an HTML page will **automatically** be available to your PHP scripts.
- The PHP $_GET and $_POST variables are used to retrieve information from forms, like user input.
- Check the following example

2

input elements ⇒ Form elements

Input → text
→ text box
Buttons

radio button
Check box
Pull down lists, etc.

$ → is the prefix of a PHP variable.

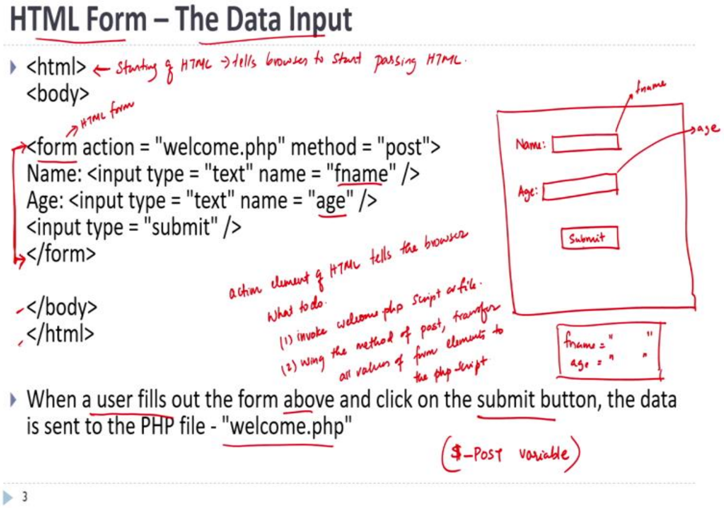
$_GET
$_POST } Specifically designed PHP Variables to capture HTML form data upon the press of the Submit button

So, let us look into just a quick refresher HTML forms a user input.

- The most important thing or the thing that we need to consider very clearly when you deal with HTML forms and PHP is that any form element on an HTML page will automatically be available in your PHP script. So, whatever be the form element, the major form element we had was input or we had a text. We have Textbox, Buttons, Radio Button Checkbox. Then, what we call Pull down lists, etcetera. So, we have seen so many of these form elements or we can call all of them as input elements also. You can call them input elements or also known as form elements. So, any of these form elements or input elements can automatically be available in PHP script.
- So, the easiest way to do this, you need to use two functions as part of this. The one is the PHP $_GET and the $_POST. So, remember that $ is the prefix of a PHP variable. I hope you guys remember from that. So, anything that you put a $ sign on it, $ and underscore. So, this is kind of two arcs as two variables but these are reserved variables, these variables, you cannot use this for something else. It is designed and specifically, this $_GET and $_POST specifically, designed PHP variables to capture HTML form data upon the press of the Submit Button.

So, when you press the Submit Button, these two variables are custom built. Specifically designed in PHP and what they do is, they can capture the information and you can use these variables to retrieve the information from HTML forms, whatever the user input be.

(Refer Slide Time: 08:15)



HTML Form – The Data Input

```
<html> ← Starting of HTML → tells browser to start parsing HTML.
<body>

<form action = "welcome.php" method = "post">
Name: <input type = "text" name = "fname" />
Age: <input type = "text" name = "age" />
<input type = "submit" />
</form>

</body>
</html>
```

- When a user fills out the form above and click on the submit button, the data is sent to the PHP file - "welcome.php"

action element of HTML tells the browser
What to do.
(1) invoke welcome php script or file.
(2) using the method of post, transfer all values of form elements to the php script

fname = "   "
age = "   "

($_POST variable)

- So, let us look into a very simple example as part of this. So, let us look at this sample code: HTML forms - the Data input. So, this is just for a refresher. Starting of HTML tells the browser to start parsing HTML. Then, the body tag so, that means the rest of it is the body. We do not have any header tag as part of it.

So then, we have the form, that says HTML form, creates an HTML form. And then, action = welcome.php. So, action implies there is a script in some place which is titled welcome . PHP and method is POST. So, when would this be triggered, when you trigger it, when you press the submit input type, it is a Submit Button? So, if you look into this HTML form, the form will look like this, this is a web page. The first thing in input type is text name is fname.

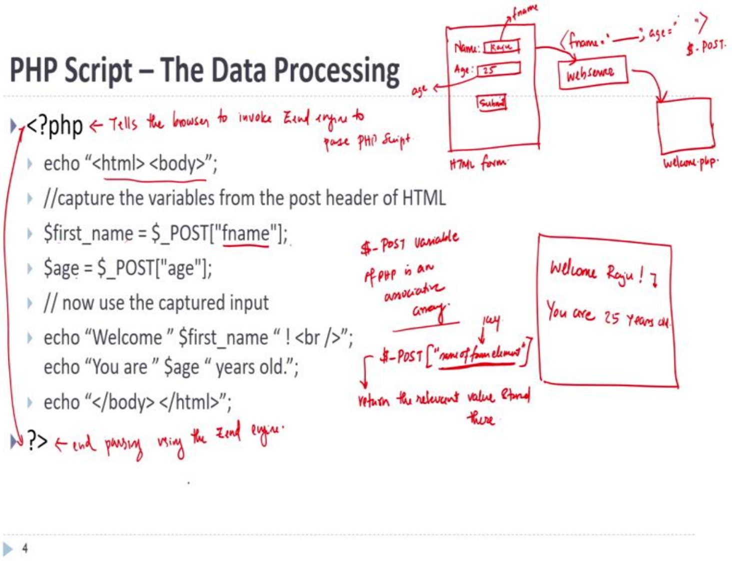
So, what does it do? It will create a form like this name to give your textbox. There is no size, nothing of the sorts, so, default size will go to 50 and you will get that bit of Textbox like this. And, this Textbox is named as fname, that is the unique name of this textbook. The next one is called age and there is also nothing so it will be a similar size of a textbox and this is named as age. So, the fname and age, these are the two names of this Textbox. And, the last input type is Submit. So, you have a button, it creates a button called Submit.

So, this is what the user will see when they see this form. What does it also do? This form within these two bodies of the comment, it tells what to do when the button is pressed. The action element of HTML tells the browser what to do. So, it tells when you press the button, the action is when it is on pressing the button.

1) It says, invoke welcome.php, script or file, whichever is in that current directory, wherever this HTML file is.
2) And then, using the method of POST, transfer all values of form elements to the PHP script. So, what it says is, when the user presses the Submit Button, you will have two things, the fname will have some value typed by the user and, the age will have some value typed by the user. Both of these values, whatever it captures, with appropriate identifier, transfer it into the welcome.php script file and that is what this is being mentioned. And then, you close the body tag and then the HTML tag, so, HTML is over that way.

- So, when the user fills out the form, when any user fills out the form above and clicks the Submit Button, the data is sent to the PHP file, welcome.php. And, it is done with the POST, the method is POST. So, it will go in the $_POST variable. So, if you want to find where this data is in that PHP file, we just need to go there and call the $_POST variable and you can extract all those variables and values right there at that point. So, that is what happened in this.

(Refer Slide Time: 12:46)



PHP Script – The Data Processing

```
<?php
    echo "<html> <body>";
    //capture the variables from the post header of HTML
    $first_name = $_POST["fname"];
    $age = $_POST["age"];
    // now use the captured input
    echo "Welcome " $first_name " ! <br />";
    echo "You are " $age " years old.";
    echo "</body> </html>";
?>
```

← Tells the browser to invoke Zend engine to parse PHP Script

← end parsing using the Zend engine.

$_POST variable of PHP is an associative array.

$_POST ["name of form element"]

return the relevant value stored there

Now in PHP, how will you handle this? How will this form be? Now, let us look at the PHP side. So now, let us redo the HTML form again. Form is, you have the user see something like name, age. And, the name, for the internal name for this is fname. The internal name for this is age, the second box. And then, there is a Submit Button and we know that upon Submit, this is the HTML page, HTML form.

And then, there is an intermediary web server. And then, there is something called a PHP file, somewhere there is a welcome.php, a PHP file is there. So, when the user presses a Submit Button, the variables, the fname equal to some value and age equal to some value is captured in the $_POST variable by the web server and then, it gets transferred to the welcome.php. So, the welcome.php, what happens? So, PHP tells the browser to invoke ZEND engine to parse PHP script. So, we are now going to parse the PHP script.

So, first thing is you are echoing HTML and body. So, what it does is it creates another HTML page and because it is HTML and it says there is no head its body. So then, you are capturing two variables. $first_name = $_POST [“fname”]. So, the $_POST, ideally, variable of PHP, is an associative array. So, if I say $_POST fname, this fname is the name of this form element. So, if you use $_POST, name of form element. If I do this, then, I can get whatever value this will return the relevant value stored there. So, if I want to access fname, whatever the value stored in fname is, I just need to say $ underscore POST fname within double quotes.

So then, it will take that value and it will assign it to the $ first name variable. Same way, $age variable, same $_POST age which is the name of the second form element. So now, $first_name and $age, both of these values associated with those variables.

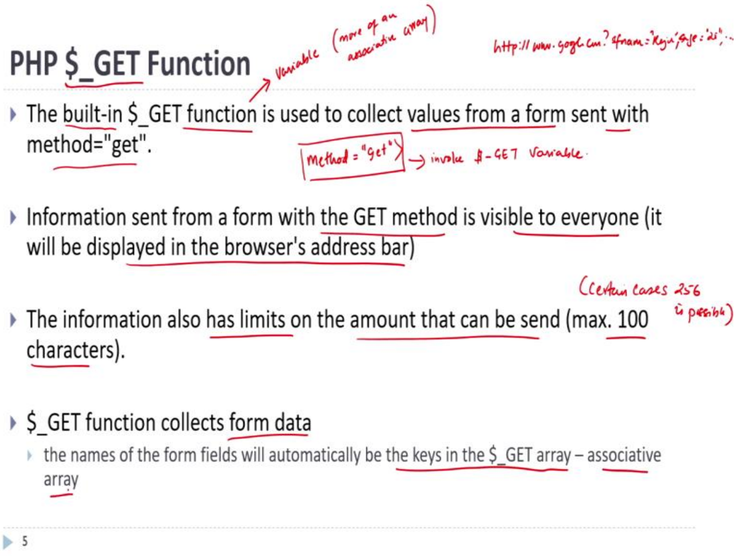
Now, what you do is you are going to print echo welcome. So, first it will print, welcome. Then, $first_name, and then, there is an exclamation mark and a line break. So, let us say I will put the first name as in this case is Raju. That is the value type. So then, this $ underscores the first name. It will show us Raju and then, we will put an exclamation mark and then, there is a line break. So, it will go to the next line. The next line will say you are, and then, it will print the age let us say put the age as 25. So, we will put 25 then, it will print years old. And then, that is it.

So, this captured the value of the form elements, Raju and the age, whatever is typed with the help of web server using the dedicated variable of $ underscore POST, which is an associative array, as I said, and then, it transferred it into the PHP script.

And then, the PHP script using the variable the $_POST to serve a variable associative array with the name of the form element, given as the key value. With that key, you can retrieve whatever the value is stored, and can be assigned to the variable. Once you get the value in the PHP script, you can do whatever you want with that. And then, you close the HTML and this completes the PHP script.

So, at this point, it will end parsing using the ZEND engine. So, that will happen. That is how this translation or this movement between the HTML form to the PHP script happens.

(Refer Slide Time: 18:20)



PHP $_GET Function

→ Variable (more of an associative array)

http://www.google.com?fname="Kejri";age="25";..

- The built-in $_GET function is used to collect values from a form sent with method="get".

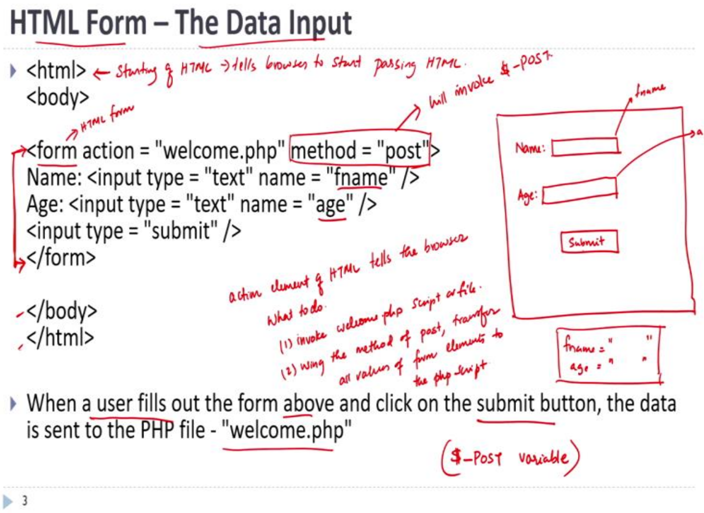
method = "get" → invoke $-GET Variable.

- Information sent from a form with the GET method is visible to everyone (it will be displayed in the browser's address bar)
- The information also has limits on the amount that can be send (max. 100 characters).

(Certain cases 256 is possible)

- $_GET function collects form data
  - the names of the form fields will automatically be the keys in the $_GET array – associative array

5



HTML Form – The Data Input

```
<html> ← Starting of HTML → tells browsers to start passing HTML.
<body>

<form action = "welcome.php" method = "post">
Name: <input type = "text" name = "fname" />
Age: <input type = "text" name = "age" />
<input type = "submit" />
</form>

</body>
</html>
```

→ HTML form

will invoke $-POST

action element of HTML tells the browser what to do.
(1) invoke welcome.php script or file.
(2) using the method of post, transfer all values of form elements to the php script

fname
age
Name:
Age:
Submit
fname = " "
age = " "

- When a user fills out the form above and click on the submit button, the data is sent to the PHP file - "welcome.php"

($-POST variable)

3

Now, another variable now lets us look at this. I said there is another one called $_GET. We just saw what $_POST. So, $_GET is also built-in. It is not that we call it the built-in function, let us call it as a built-in variable for the time being, it is actually more of an associative array.

- And, it is used to collect values from a form the values from a form sent with the method GET. So, in the previous example, we saw the method is POST. So, it will invoke $_POST, if you change this to method = “get”> if I do this, So, this will invoke $_GET variable or method.
- The main difference with the get in the POST is the information sent from a form using the GET method is visible to everyone. It is visible to everyone means where it will be displayed

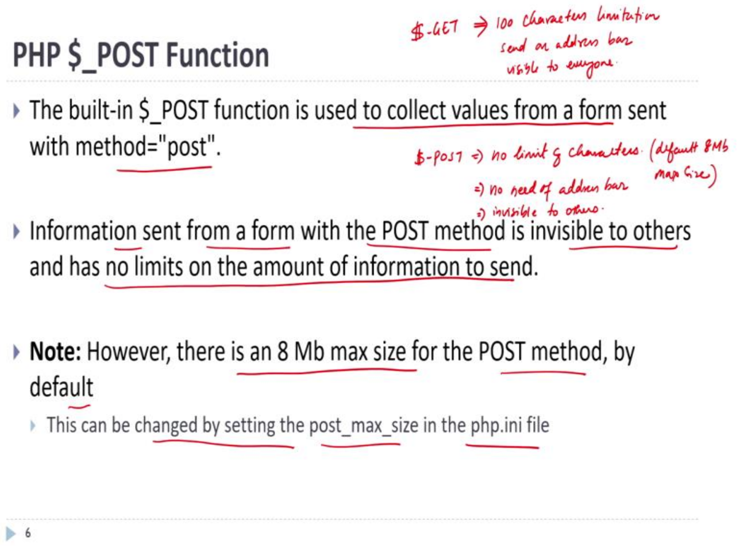
on the browser's address bar. So, if you write something it will actually show somewhere http://www.google.com?fname=”Raju”,age=”25”

So, whatever the value that you type in the HTML form, it will be visible to everyone at the browser's address bar. It has a disadvantage because if you use password and other things that will also be made visible to this.

- Plus, also there is another disadvantage, there are also limits on the amount of data that can be sent maximum 100 characters. In certain cases, 256 is possible, but that is it, you cannot send any more than, usually, you can take it as 100 characters is what you get.

So, as I said earlier, the function collects the form data, and the names in the form fields will automatically be the keys in the $ GET array, it is an as I said, it is an associative array. So, it will automatically get that.

(Refer Slide Time: 21:09)



PHP $_POST Function

$_GET ⇒ 100 characters limitation
send on address bar
visible to everyone.

- The built-in $_POST function is used to collect values from a form sent with method="post".

$_POST ⇒ no limit of characters. (default 8Mb max size)
⇒ no need of address bar
⇒ invisible to others.

- Information sent from a form with the POST method is invisible to others and has no limits on the amount of information to send.
- **Note:** However, there is an 8 Mb max size for the POST method, by default
  - This can be changed by setting the post_max_size in the php.ini file

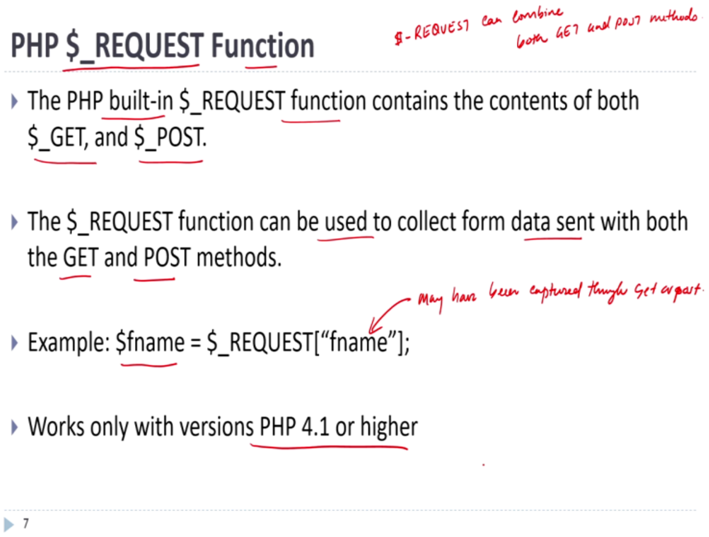
6

- So, where is the other advantage is the POST function, the built-in $_POST function can collect the values from a form sent with the method POST, as I showed you, it will collect all the values from a HTML form and it sends using the POST method.
- The main advantage is the information sent from a form with the POST method is invisible to others and has no limits on the amount of information to send. So, in $_GET there is a 100 characters limitation sent on the address bar visible to everyone; these three things will happen with $_GET. $_POST, no limit of characters. There is a max size that is a default

8 Mb max size. This can be changed but can be changed in with no need for an address bar and invisible to others. So, you will preserve this antiquity or safety of the information that is sent out as part of this form.

- So, as I said earlier, there is an 8 Mb max size for the POST method by the default, but it can be changed by the setting POST max size in the php.ini file. So, it is not a big deal, you can change it into whatever you want, just need to go and update the php.ini file and it can change to 10 Mb, 20 Mb, whatever the stuff you want to do.

(Refer Slide Time: 23:12)



PHP $_REQUEST Function

$-REQUEST can combine both GET and POST methods.

- The PHP built-in $_REQUEST function contains the contents of both $_GET, and $_POST.
- The $_REQUEST function can be used to collect form data sent with both the GET and POST methods.
- Example: $fname = $_REQUEST["fname"];

May have been captured through Get or post.

- Works only with versions PHP 4.1 or higher

7

So then, there is another function where we need to look into this $_REQUEST function. This is another built-in PHP function.

- So, the built-in $_REQUEST function. PHP contains the content of both $_GET and $_POST. So, $_REQUEST can combine both GET and POST methods. So, the REQUEST function can be used.
- It can be used to collect form data sent with both GET and POST methods. So, instead of remembering whether it is GET or POST, just use REQUEST, that is another way to do.
- Example is $fname = $_REQUEST ["fname"];

So, fname may have been captured through GET or POST, it does not matter. Whatever way it was captured, the request will be able to handle this.

- However, there is one important thing to this, this is only worse with PHP 4.1 or higher. So, if it is an old version of PHP, you may not be able to use the request function. So, if it is PHP 4.1 or higher then, you can definitely use this function.

So, that brings us to the end of this topic. And, as I said earlier, we just covered how to capture the form element and how to get the data to PHP. We saw the usage of GET, we saw the usage of POST. And, we also saw how we can combine GET and POST with the help of REQUEST. The only problem with the REQUEST is, it is only available from PHP 4.1 or no.

But REQUEST can handle either GET or POST, you do not have to worry which method it is captured in, whether GET or POST REQUEST can handle that. But, in older versions of PHP, when you are dealing with legacy codes and stuff like that, remember that REQUEST will not work.

So, ideally, the right thing to do is, if you use POST to capture the, if you are going to use the method of POST to send the data from the HTML form, it is better to use the POST in the PHP script. If you are sending the GET from the HTML form then, use the GET.

My advice is as much as possible stay away from GET, unless it is really, really, really simple data. And, it does not matter whether somebody sees it or not, it does not matter to you. That is the case. And, there is a limitation of 100 characters, if it is well within that then, you can use GET, it is the quickest and fastest way to do it. But, if there is anything sensitive, like a bank account information, some personal stuff, date of birth, etcetera, that you do not want people to see.

So then, it is always better to go with the POST method, nobody gets to see that. But remember, there is a PHP by default There is an 8 Mb limitation. But, if you want to change that limitation, 20 MB, you can go to the PHP . ini file, and you can change the variable with the POST max size. And, you can change it into whatever you want and then, you can go from there.

So, with that we come to the conclusion of this topic. And, now you already know how to transfer data from PHP to a database. We saw how to write a query, how to create the connection, and how to store all that information. So, that is what it is and the next lecture or the final lecture, we will just look into the different types of computer networks because that has been a major question by people and what are the type of networks and etcetera. We will cover that in the next class, which will complete the lecture for this particular last week. Thank you very much.