Simple Logination

Basic Logination Approach With PHP

1.0 Database Configuration

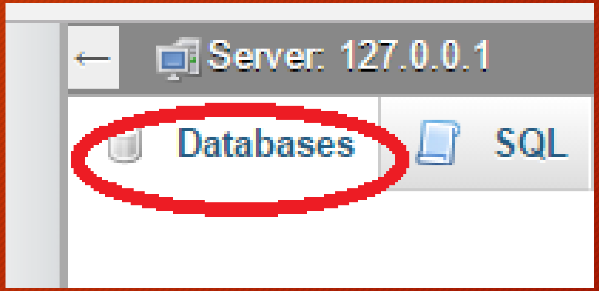
Using PHPMyAdmin in MySQL/MariaDB

1.1 Introduction

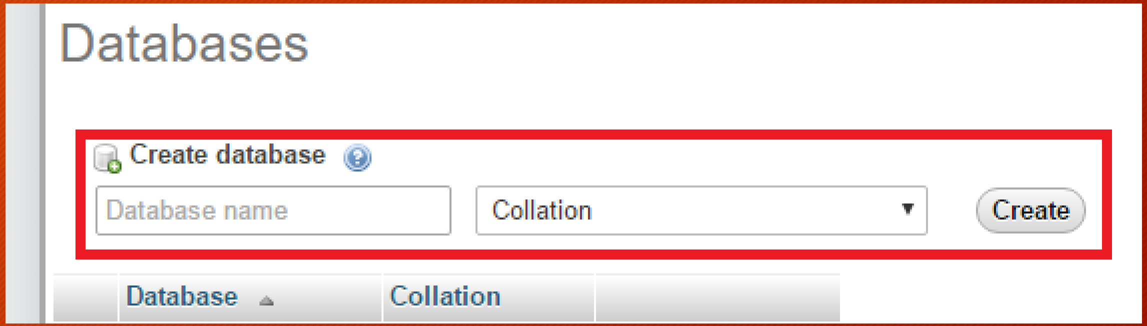
- We will be needing a databased created with a table to store credentials.
- Let ‘MYDB’ is the database name for this example and ‘ACCOUNTS’ will be the table name to store credentials.
- In the table ‘ACCOUNTS’ there should have at least TWO (2) columns which is identification and password.
- Let identification column be ‘ID’ and password column be ‘PASS’.
- Let ‘ID’ be an INT with max length of 10 and ‘PASS’ be a VARCHAR with max length of 50.

1.2 Database creation

- Launch ‘PHPMyAdmin’ by typing ‘http://localhost/phpmyadmin/’ in your web browser.
- On the navigation tab, click ‘Database’.

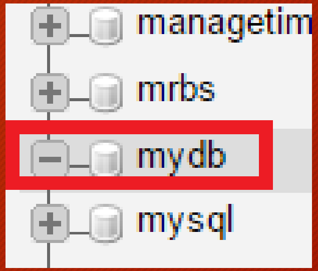


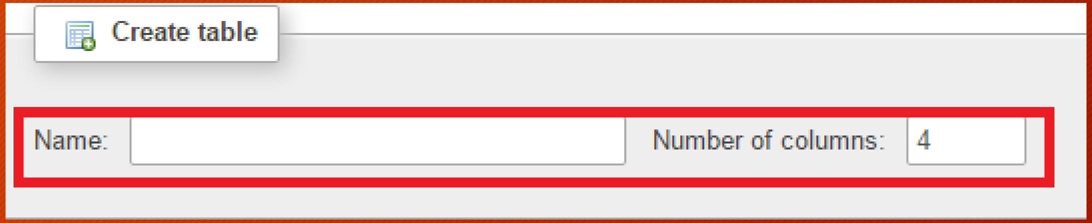
- Type in ‘MYDB’ in the ‘Create database’ section and click ‘Create’.



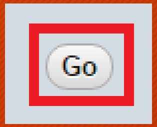
1.3.1 Table creation

- Click your database in the database tree.
- Type in ‘ACCOUNTS’ in the ‘Create table’ section and type ‘2’ is the ‘Number of columns’.



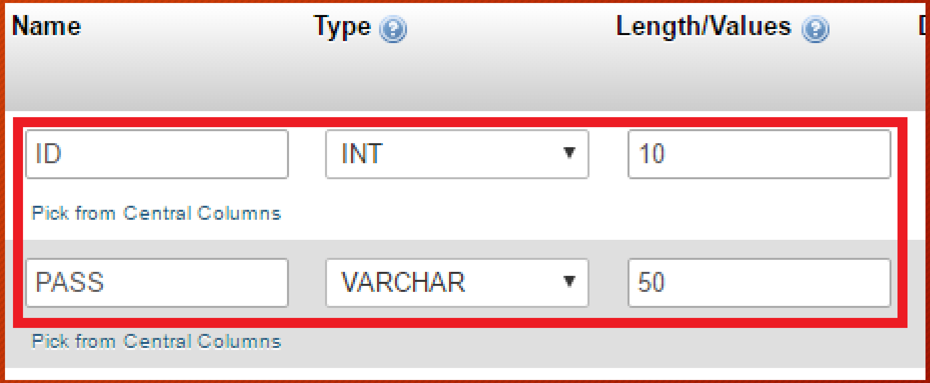


- Click the button ‘Go’.

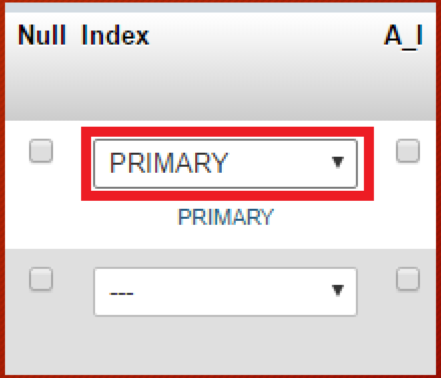


1.3.2 Column assignments

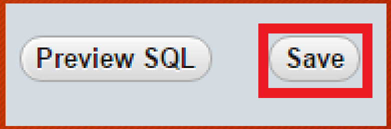
- Type in ‘ID’ and ‘PASS’ in the column name.
- For the column ‘ID’ set:
  - Type - INT
  - Length - 10
  - Index – PRIMARY
- For the column ‘PASS’ set:
  - Type – VARCHAR
  - Length – 50
- Click ‘Save’.



Name	Type	Length/Values
ID	INT	10
Pick from Central Columns		
PASS	VARCHAR	50
Pick from Central Columns		

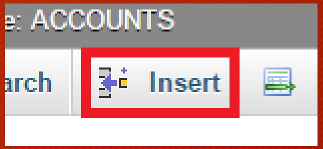


Null	Index	A_I
☐	PRIMARY	☐
	PRIMARY	
☐	---	☐

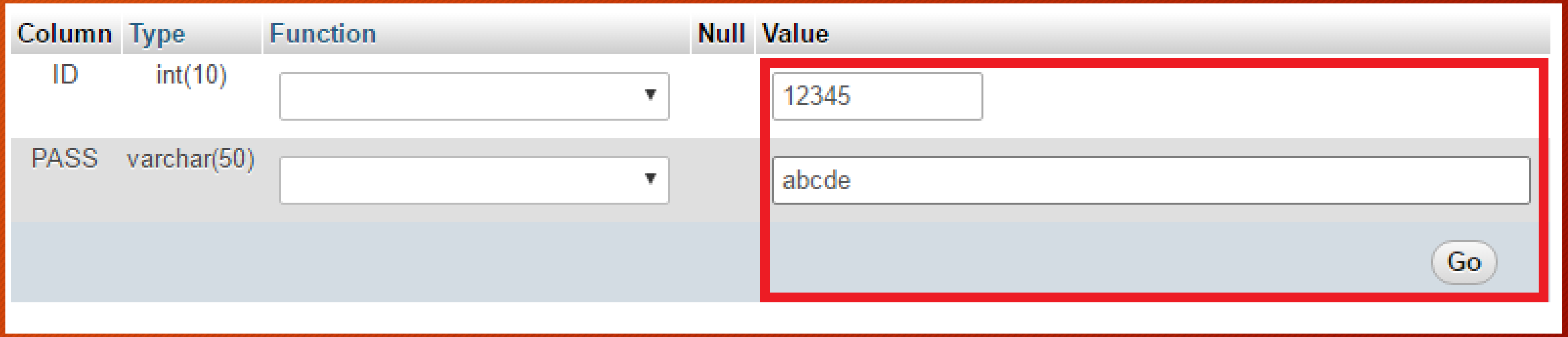


1.4 Account creation

- While still connected with the table ‘ACCOUNTS’ click the ‘Insert’ tab at the navigation tab.



- Lets create our first account with the following credentials:
  - ID – 12345
  - PASS – abcde
- Type in the credential value in appropriate fields and click the button ‘Go’.



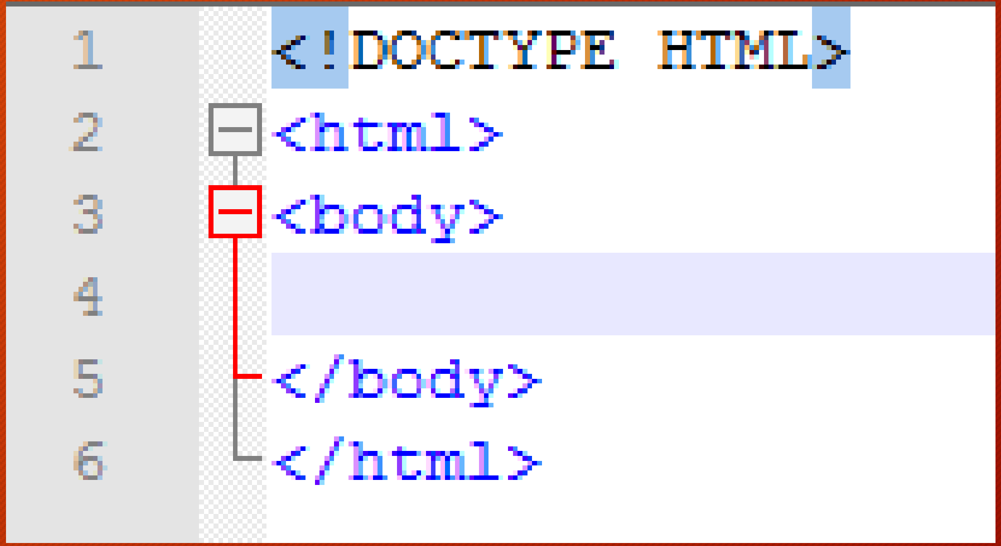
Column	Type	Function	Null	Value
ID	int(10)			12345
PASS	varchar(50)			abcde

2.0 HTML Login Form

Creation of a HTML form to input credentials

2.1 Create a HTML page

- Launch ‘Notepad++’ and create new page with the name ‘login.html’.
- In the page, write the basic html tags as in the right of this slide.
- Proceed to the next slide for form creation.

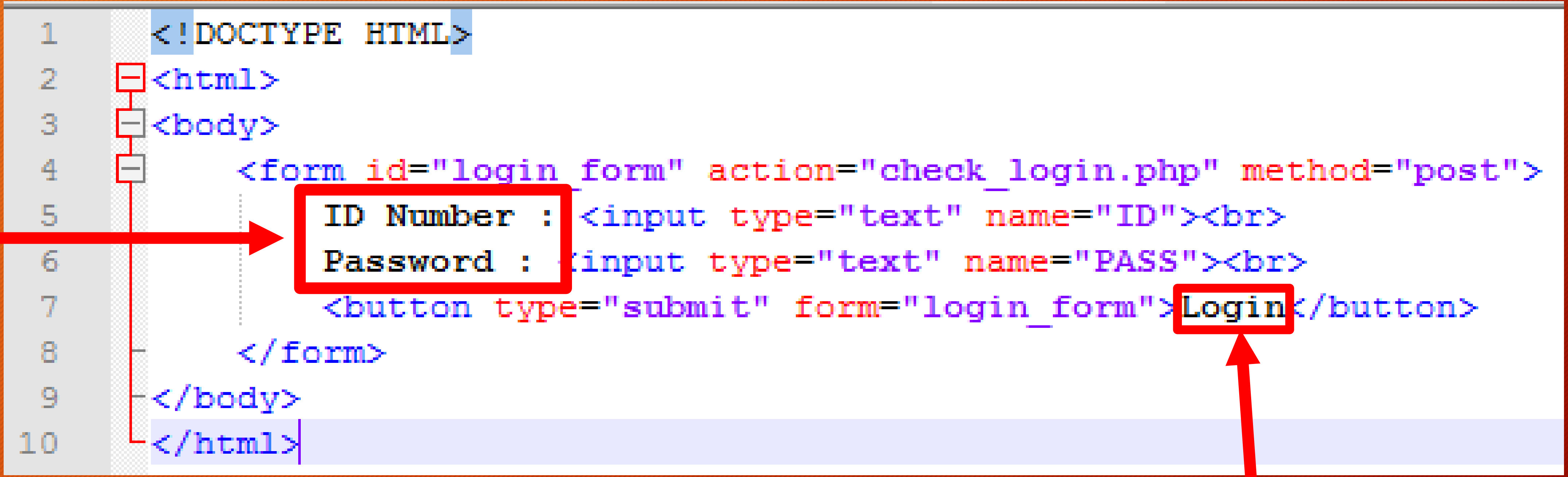


```
<!DOCTYPE HTML>
<html>
<body>

</body>
</html>
```

2.2.1 HTML form creation

- To create a HTML form, simply open and close the tag ‘<form>’.
- In the open tag 3 things should be declared:
  - Id – login_form (the name of the form)
  - Action – check_login.php (the page to process the login credentials)
  - Method – post (type of form values handling)
- In between the open an close tag, it should contain 2 input fields for the ‘ID’ & ‘PASS’ and also a button to submit the form.
- To create an input field, simply type in a single tag ‘<input> and should be declared 2 things:
  - Type – text (the type of input)
  - Name – ID (the attribute/variable name of the input in the page)
- To create a button, simple open and close the tag ‘<button’> with declaration in the open tag as such:
  - Type – submit (the button action type)
  - Form – login_form (the form name as declared in the form tag)

2.2.2 Full HTML form example



```
<!DOCTYPE HTML>
<html>
<body>
    <form id="login_form" action="check_login.php" method="post">
        ID Number : <input type="text" name="ID"><br>
        Password : <input type="text" name="PASS"><br>
        <button type="submit" form="login_form">Login</button>
    </form>
</body>
</html>
```

Label for the inputs

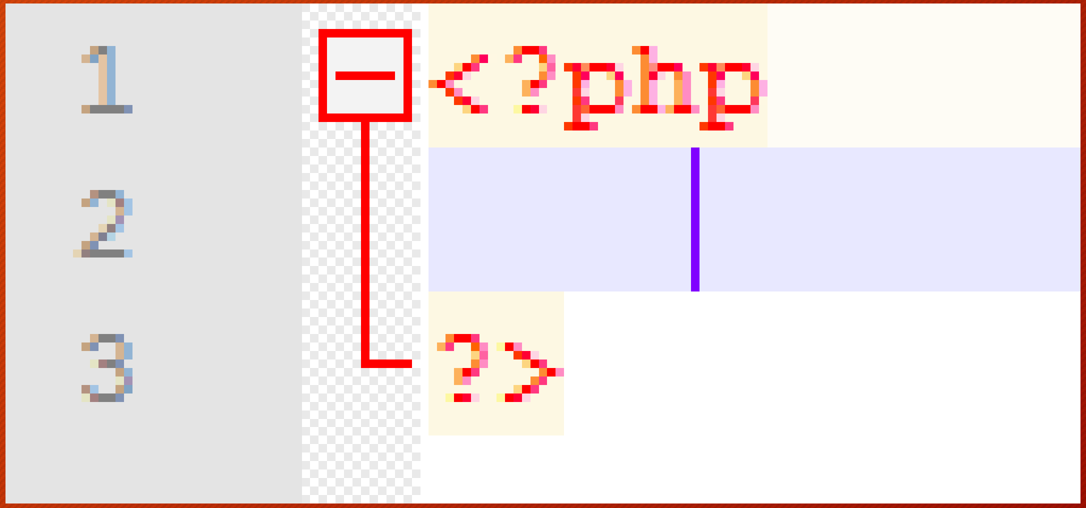
Label for the button

3.0 PHP Process Page

The page to process the input credentials

3.1 Create a PHP page

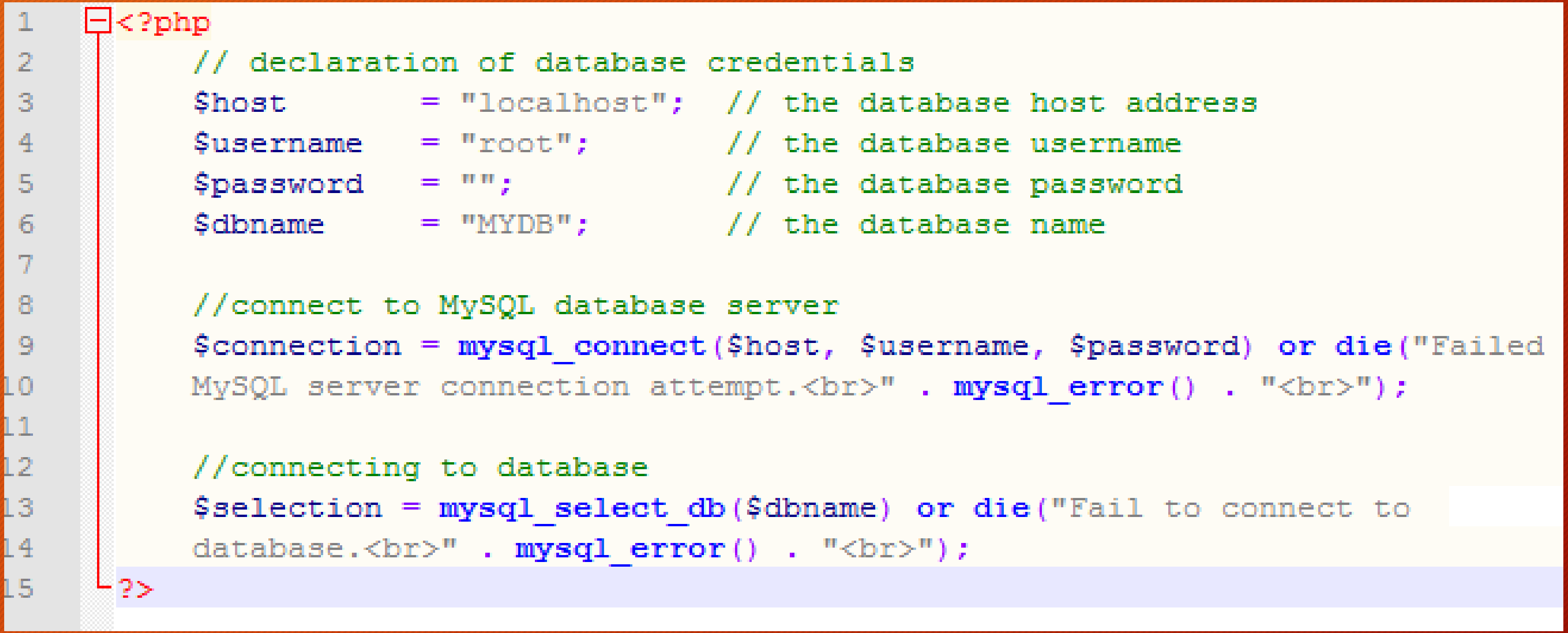
- In the ‘Notepad++’ create a new page named ‘check_login.php’.
- Open and close the page with a PHP tag as shown in the right.
- Proceed to the next slide for the process programming.



```
<?php

?>
```

3.2 Create a database connection script

- Create a new page named ‘connectDB.php’ so that you will not need to rewrite the connection codes everytime but rather call this page with the command as follow: `include(“connectDB.php”);`

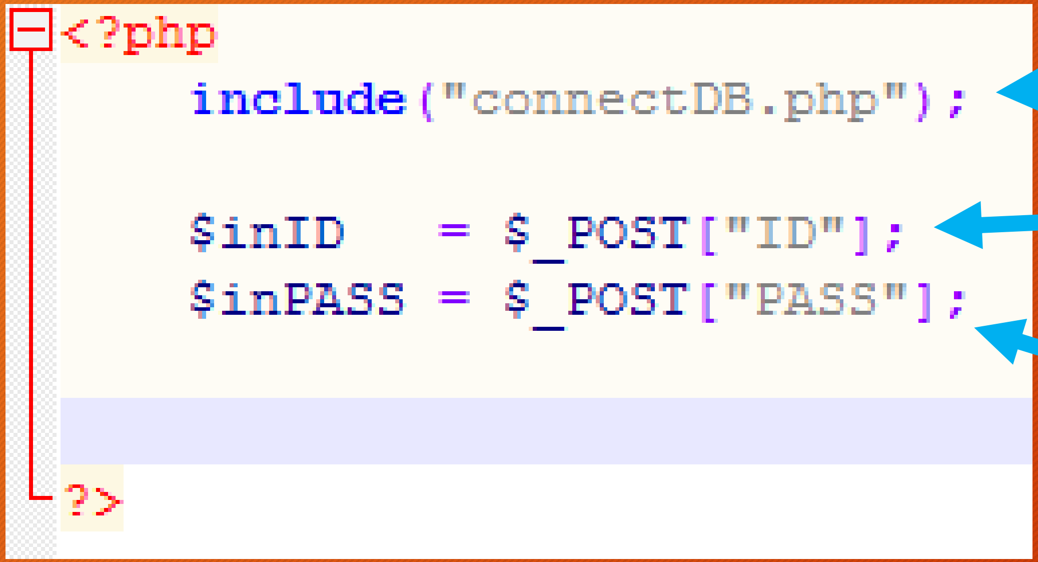


```php
<?php
    // declaration of database credentials
    $host        = "localhost";  // the database host address
    $username    = "root";       // the database username
    $password    = "";           // the database password
    $dbname      = "MYDB";       // the database name

    //connect to MySQL database server
    $connection = mysql_connect($host, $username, $password) or die("Failed
    MySQL server connection attempt.<br>" . mysql_error() . "<br>");

    //connecting to database
    $selection = mysql_select_db($dbname) or die("Fail to connect to
    database.<br>" . mysql_error() . "<br>");
?>
```

3.3 Get the input value from the form

- Back to ‘check_login.php’, include the ‘connectDB.php’ script.
- Fetch the values from the page ‘login.html’ using the ‘$_POST[“”]’ array variable.
- The index name of that variable should 100% same with the variable name that we want to fetch from the page ‘login.php’.
- Insert the fetch value into a new variable in PHP.
- Proceed to the next slide to view the full example.

Example



```
<?php
    include("connectDB.php");

    $inID   = $_POST["ID"];
    $inPASS = $_POST["PASS"];


?>
```

- Include the connection script
- The value from variable ‘ID’ in ‘login.html’ is fetched and inserted into new PHP variable name ‘inID’
- The value from variable ‘PASS’ in ‘login.html’ is fetched and inserted into new PHP variable name ‘inPASS’

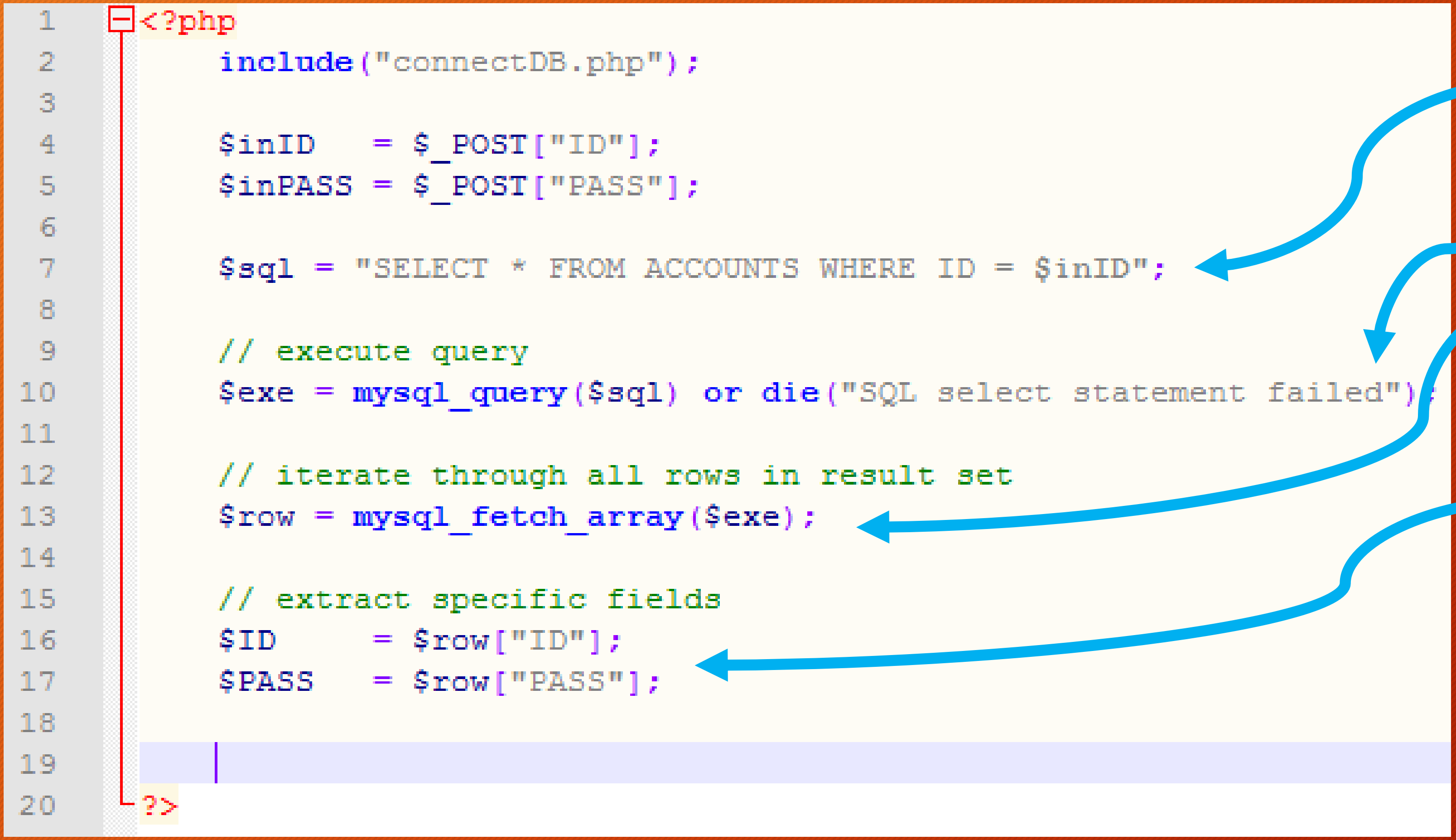
3.4 SQL query and execution

- The next step is to create a SQL query variable name ‘$sql’ and input a SQL query into its value.
- We will be using SELECT query and the query should be as follow:
  - “SELECT * FROM <table name>;”
  - Example for this tutorial : “SELECT * FROM ACCOUNTS WHERE ID = $inID;”
- Next we will execute with a PHP function ‘mysql_query();’ and the return value shall be inserted into a new variable named ‘$exe’.
- After that we should use PHP function ‘mysql_fetch_array();’ to convert the returned value in ‘$exe’ into a legitimate array.

Continuation of 3.4 SQL query and execution

- The returned value from the function ‘mysql_fetch_array();’ should be inserted into new variable named ‘$row’. This variable will be an array variable in accordance to the column number that was selected from the database.
- Next we will extract those array value into a single variable so that it is easy for us to do the matching process of the credentials.
- Proceed to the next slide to seek full example.

Example



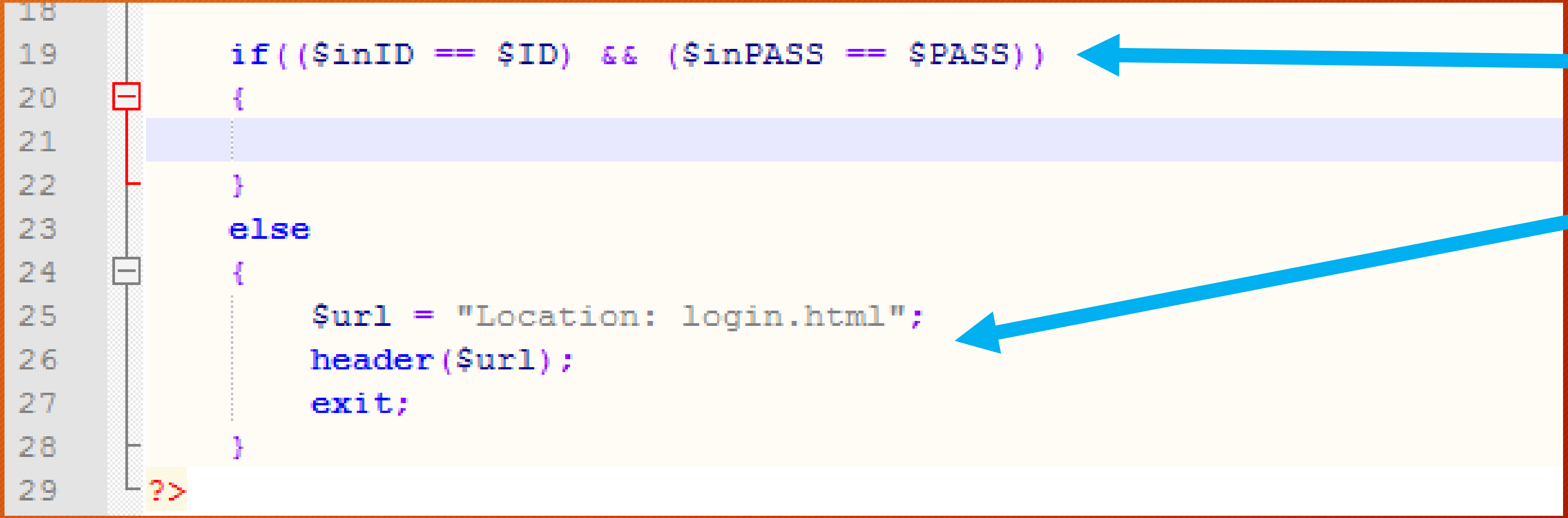
```
<?php
    include("connectDB.php");

    $inID   = $_POST["ID"];
    $inPASS = $_POST["PASS"];

    $sql = "SELECT * FROM ACCOUNTS WHERE ID = $inID";

    // execute query
    $exe = mysql_query($sql) or die("SQL select statement failed");

    // iterate through all rows in result set
    $row = mysql_fetch_array($exe);

    // extract specific fields
    $ID     = $row["ID"];
    $PASS   = $row["PASS"];


?>
```

- SQL statement
- SQL query execution
- Returned result from database inserted into new variable
- Extracting column values into new created PHP variables

3.5 Processing the credentials

- Now we should compare both the input values and the fetched values from the database.
- Both the ‘ID’ and ‘PASS’ values MUST BE EXACTLY THE SAME in order to grant a successful login to a user.
- We will be using if..else condition to compare the input from user and the values stored into the database.
- If true then a SESSION will be created. (SESSION will be covered in the next sub-topic)
- If false we will bring back the user to the login page.

Example



```
    if(($inID == $ID) && ($inPASS == $PASS))
    {

    }
    else
    {
        $url = "Location: login.html";
        header($url);
        exit;
    }
?>
```

- Validating the credentials
- When credential didn’t matched then the user will be ‘kicked’ back to the login page.

4.0 Introduction to SESSION

Basic introduction and implementation of $_SESSION[“”]

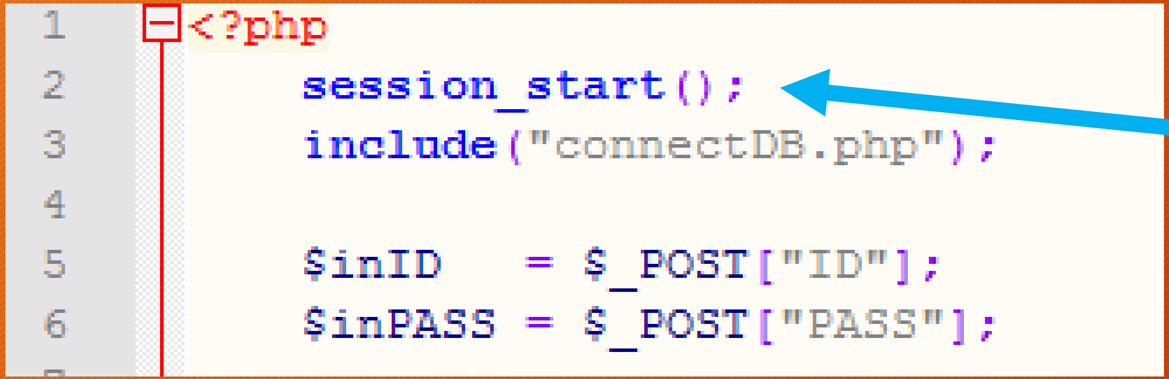
4.1 Introduction to SESSION

- SESSION is a global variable used to set a value for every session of a visiting web page and a PHP function ‘session_start();’ should be declared in the first line after the PHP open tag.
- SESSION it self is an array and looks like below:

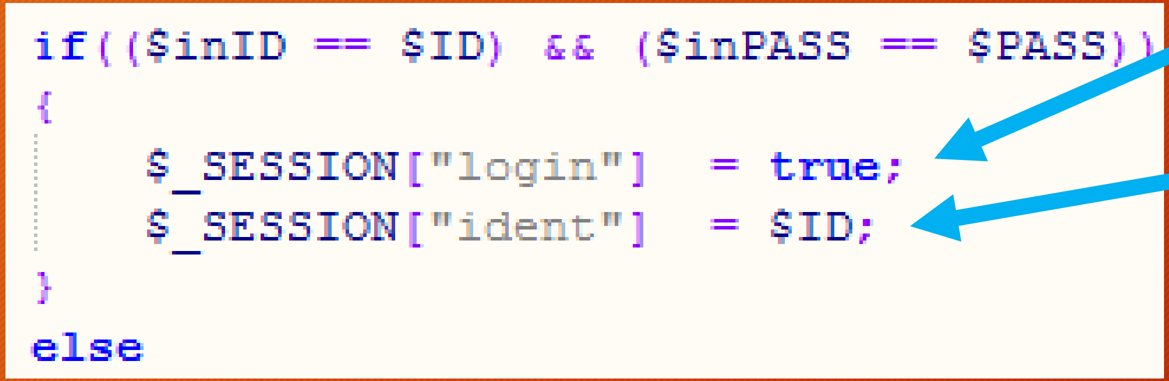
```
$_SESSION[“”];
```

- SESSION is used to determine whether a user is logged in or not and if logged in what are the identity information of that logged in user.
- Once a true match during the process of a login, a SESSION ‘$_SESSION[“login”]’ shall be set to ‘TRUE’ Boolean value.

Example



```php
<?php
    session_start();
    include("connectDB.php");

    $inID   = $_POST["ID"];
    $inPASS = $_POST["PASS"];
```



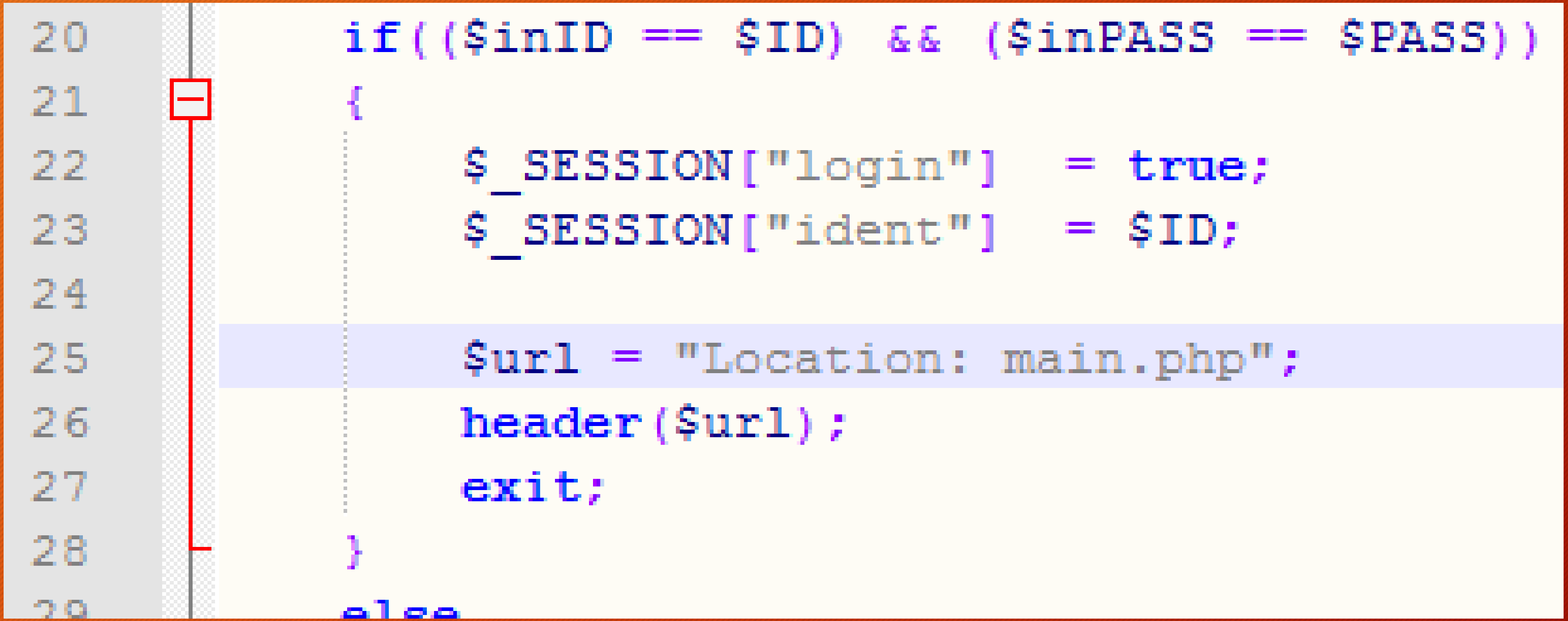
```php
if(($inID == $ID) && ($inPASS == $PASS))
{
    $_SESSION["login"]  = true;
    $_SESSION["ident"]  = $ID;
}
else
```

- SESSION initialization declaration
- Set ‘$_SESSION[“login”]’ as ‘TRUE’
- Set ‘$_SESSION[“ident”]’ with the value of a user’s identification number.

4.2 Send the user to the main page

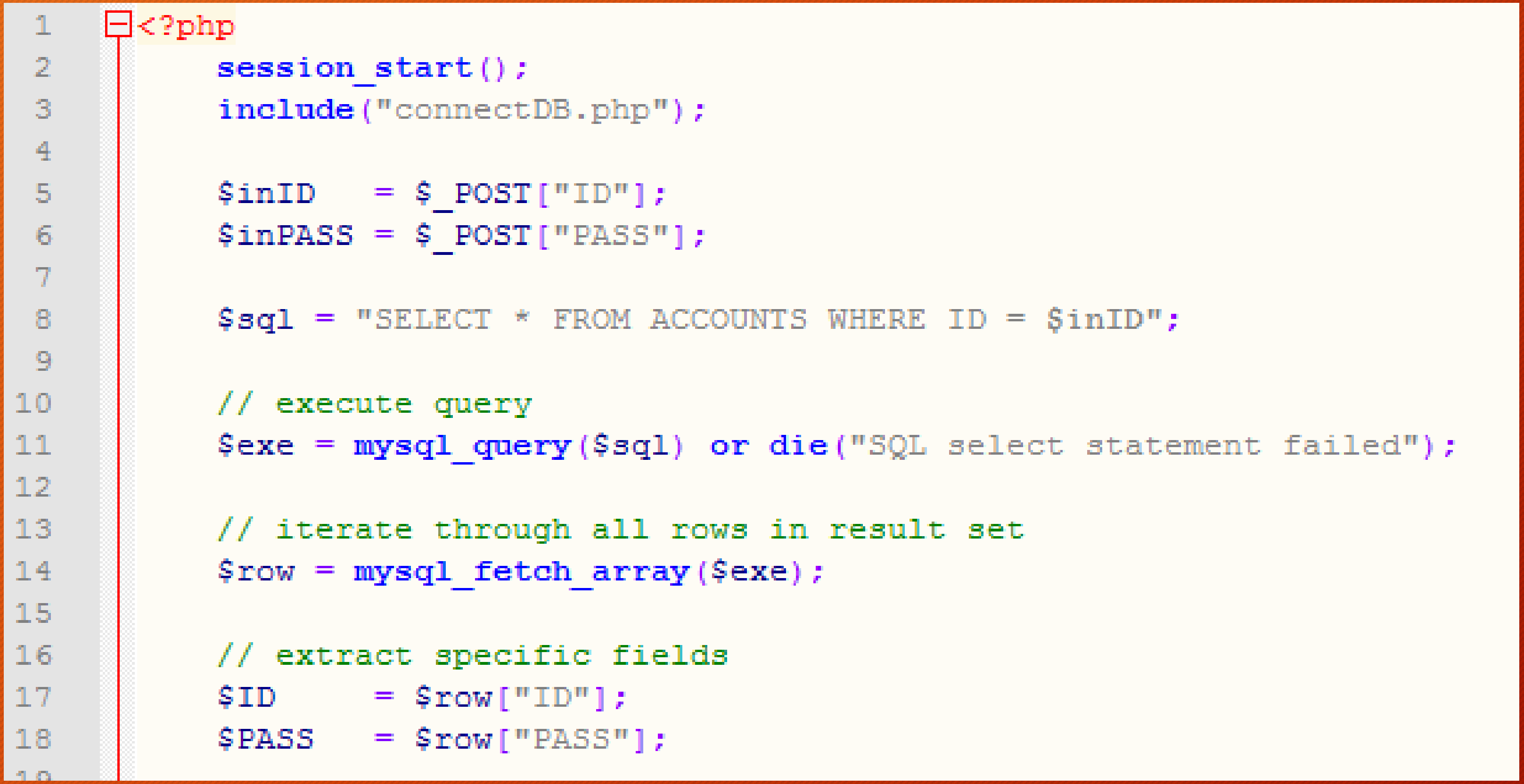
- After all login process was executed and it passed the compare test, SESSIONs was set.
- After the SESSIONs was set, we should redirect the user to the main logged in page using the same method to ‘kick’ the user back to the login page whenever the credentials are not match.
- In this tutorial we assumed that the main page is named ‘main.php’
- Seek example at the following slide and the full example at the slides ahead.

Example



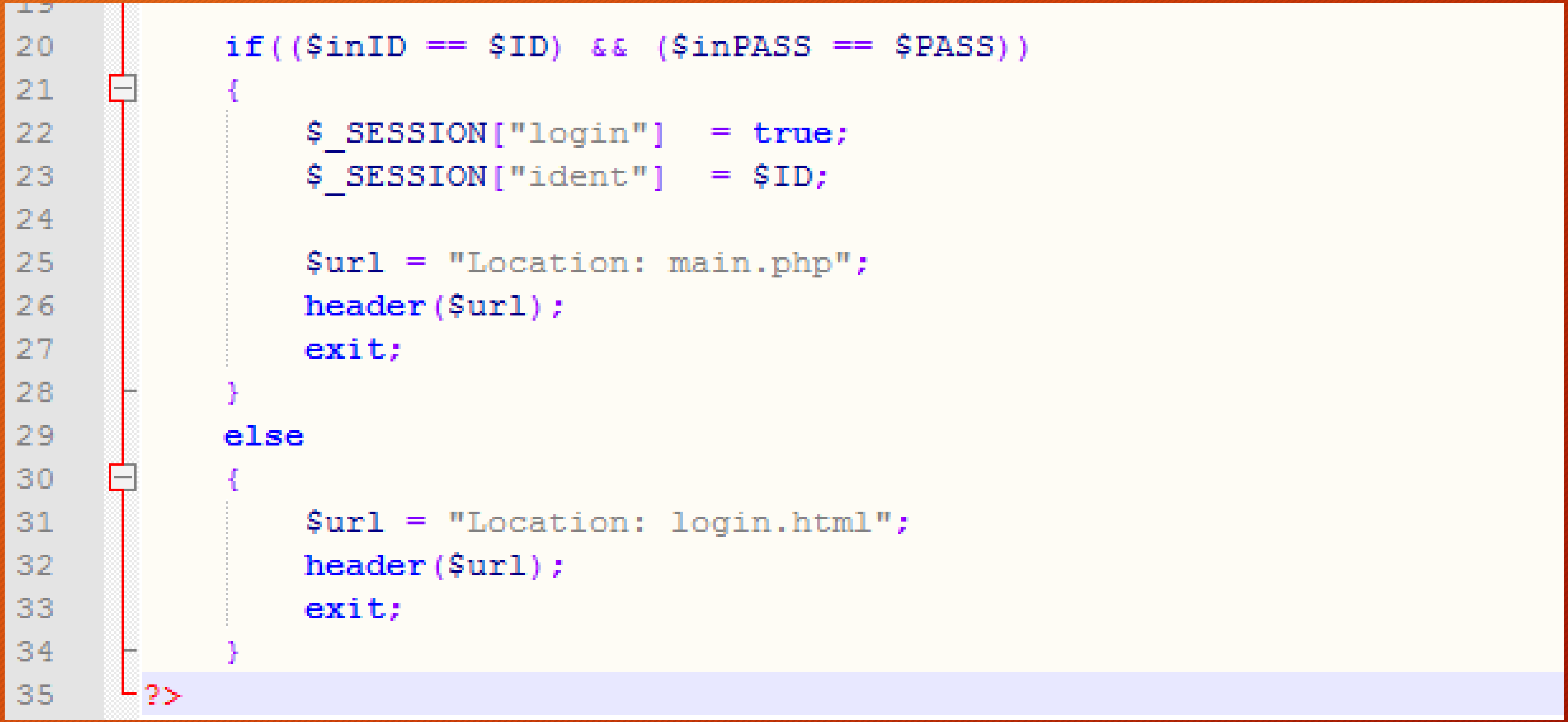
```
    if(($inID == $ID) && ($inPASS == $PASS))
    {
        $_SESSION["login"]  = true;
        $_SESSION["ident"]  = $ID;

        $url = "Location: main.php";
        header($url);
        exit;
    }
```

Full example for check_login.php



```php
<?php
	session_start();
	include("connectDB.php");

	$inID   = $_POST["ID"];
	$inPASS = $_POST["PASS"];

	$sql = "SELECT * FROM ACCOUNTS WHERE ID = $inID";

	// execute query
	$exe = mysql_query($sql) or die("SQL select statement failed");

	// iterate through all rows in result set
	$row = mysql_fetch_array($exe);

	// extract specific fields
	$ID     = $row["ID"];
	$PASS   = $row["PASS"];
```

Continuation of full example



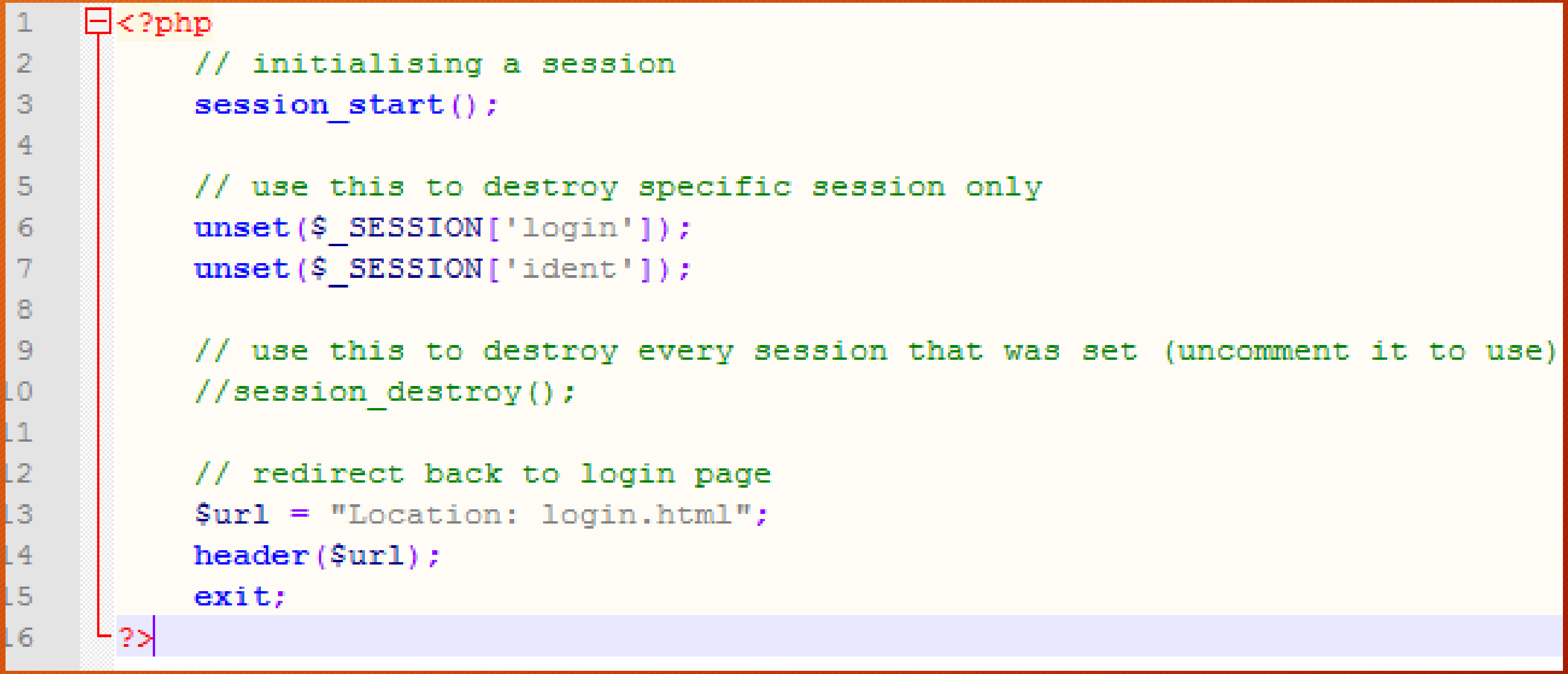
```
    if(($inID == $ID) && ($inPASS == $PASS))
    {
        $_SESSION["login"]  = true;
        $_SESSION["ident"]  = $ID;

        $url = "Location: main.php";
        header($url);
        exit;
    }
    else
    {
        $url = "Location: login.html";
        header($url);
        exit;
    }
?>
```

5.0 Logout SESSION Destroy

Logout process by destroying current SESSION

5.1 Creating a logout process page

- Create a new PHP page named ‘logout.php’.
- Write the code as shown below into the ‘logout.php’.



```php
<?php
    // initialising a session
    session_start();

    // use this to destroy specific session only
    unset($_SESSION['login']);
    unset($_SESSION['ident']);

    // use this to destroy every session that was set (uncomment it to use)
    //session_destroy();

    // redirect back to login page
    $url = "Location: login.html";
    header($url);
    exit;
?>
```

5.2 Types of SESSION destroy

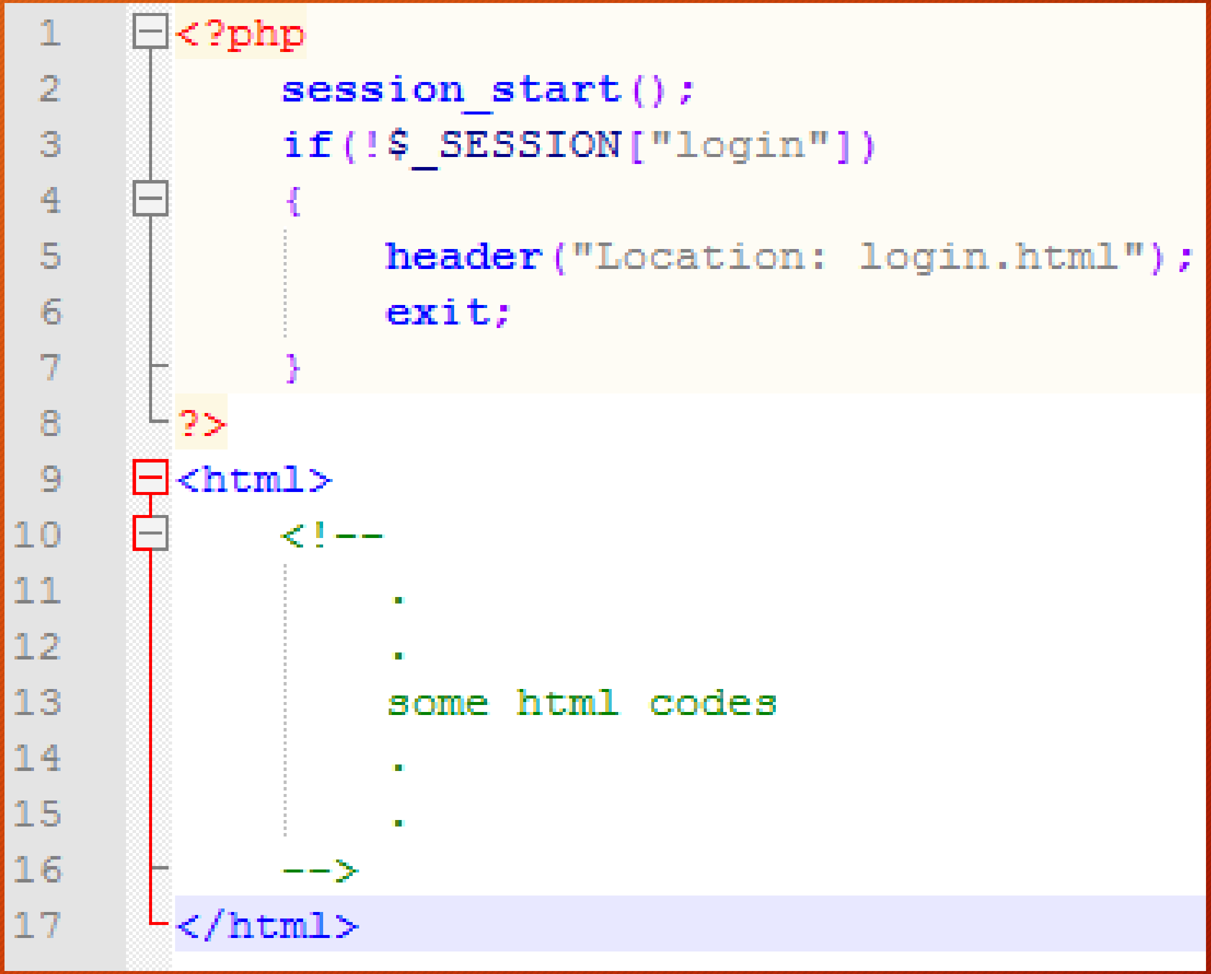
- There are TWO (2) ways to destroy a SESSION which is by using a PHP function as follow
  - session_destroy() – to destroy ALL session that was set
  - unset($_SESSION[“login”]) – destroy only specific session
- In order to log out, a link or a button must be pointing to the page ‘logout.php’.
- Example:
  - Link - <a href=“logout.php> Logout </a>
  - Button - <a href=“logout.php><button> Logout </button></a>
- The only difference is the button tag between the link tag.

IMPORTANT NOTES

IMPORTANT NOTES!

- Every page that need to be ‘logged in’ in order to access the page MUST declare ‘session_start();’ at the very beginning of those pages and should also accompany with a process that check current user login status.
- Next slide is the example of the process to kick user to the login page if the user try to access a logged in only page without actually successfully logged in into the system or the user has logged out from the system or even the SESSION has expired due to certain duration of inactivity.

Example



```
<?php
    session_start();
    if(!$_SESSION["login"])
    {
        header("Location: login.html");
        exit;
    }
?>
<html>
    <!--
        .
        .
        some html codes
        .
        .
    -->
</html>
```

END OF TUTORIAL

Prepared by MieSaF