NAME: NDIKUBWIMANA Donat

REGN: 21RP06350

CLASS: L6Y2 IT A

ITLBP601: Develop backend using PHP

1. **PHP** (recursive acronym for PHP: Hypertext Preprocessor) is a widely-used open source general-purpose scripting language that is especially suited for web

PHP is a server side scripting language. That is used to develop Static websites or Dynamic websites or Web applications. PHP stands for Hypertext Pre-processor, that earlier stood for Personal Home Pages.

PHP scripts can only be interpreted on a server that has PHP installed.

A script is a set of programming instructions that is interpreted at runtime.

2. PHP allows web developers to create dynamic content and interact with databases. PHP is known for its simplicity, speed, and flexibility — features that have made it a cornerstone in the web development world.

   Though these days it is not considered the sexiest of languages, here are some key PHP benefits that help explain why it is still so important in web development.

   .It's easy to learn and use: One of the main reasons PHP became so commonplace is that it is relatively simple to get started with. Even without extensive knowledge or experience in web development, most people could create a web page with a single PHP file in a relatively short period of time. The syntax is simple and command functions are easy to learn, meaning the barriers to entry with PHP are lower than with many other languages.

   .It's open source (and therefore free!): This also helps developers get started with PHP - it can be installed quickly and at zero cost. There is also open access to a wide range of PHP frameworks, such as

Laravel and Symfony. This feature is also appealing to companies as it helps control the costs of web development.
It’s versatile: One of the major benefits of PHP is that it is platform independent, meaning it can be used on Mac OS, Windows, Linux and supports most web browsers. It also supports all the major web servers, making it easy to deploy on different systems and platforms at minimal additional cost.

.It enjoys strong community support: As a veteran scripting language that is widely used, PHP now has a large and loyal community base to support it. There are tons of tutorials, FAQs, and tips to help new PHP developers and to continue pushing the boundaries of what the language can achieve through regular updates.

.It’s fast and secure: Two things that every organization wants their website or application to be are fast and secure. PHP uses its own memory and competes well on speed, especially when using the newer versions. There have been questions in the past about PHP security, though it is important to note that it is not inherently more or less secure than other programming languages. One important benefit is that because of its widespread use and community support there are now many tools, frameworks and best practices to help fix vulnerabilities and protect against cyberattacks.

.It is well connected with databases: PHP makes it easy to connect securely with almost any kind of database. This gives developers more freedom when choosing which database is best suited for the application being built.

.It is tried and tested: One major benefit of being around for a quarter century is that PHP code has been put to the test in all kinds of real-life environments. The main bugs have been found and fixed, making the language more stable and trusted by developers. Moreover, many frameworks and tools have been built over time, helping to make PHP web development more secure, efficient and effective.

.There’s a lot of legacy code: OK, this isn’t really a benefit of using PHP in itself, but when so many existing websites have been written with PHP it becomes an important consideration. Put simply, it is usually easier to make updates in the same language rather than try to rewrite everything in another. This helps PHP endure even when some younger developers may have a personal preference for another language. This legacy effect also means it is usually easier to find a PHP developer for your organization, though as we have written before on this blog you want to make sure they have the right skills.

why we use PHP for web development and choose it widely in our best projects.

. Cost-effective choice. Using PHP for web application development reduces the costs of creating apps and websites as it’s open-source, so most frameworks and libraries are available for free. There’s also a large international community behind it, so the language is constantly upgraded.
. Security. PHP is considered one of the safest programming languages, as it comes with built-in security features which better protect websites from viruses, malware, and common security threats.

. Performance. Compared to other programming languages, PHP scripts execute much faster, so PHP-based websites or web apps come with a faster loading speed and enhanced response time. Plus, it can be easily integrated with other languages and a wide range of databases and works on any platform and web browser.

. PHP for web development. The language can be embedded into HTML, so it’s a perfect choice if you plan to introduce a web application to the market. But, as a mature and flexible language, it can be used for other purposes too.

. Ease of use. It’s no surprise that PHP is that popular among developers — it’s easy to get started with, as the language is clear and well organized. Developers who code in other languages easily understand the command PHP functions. So, the language is simple for a newcomer, but a variety of advanced features makes professional programmers more confident — they can join the team in the middle of a project without getting lost.

3. We have collected all the official information and code available for past PHP releases. You can find more details on the current release on our downloads

   PHP 5.2.13 Release... · PHP 5.2.12 Release... · PHP 5.2.9 Release... · PHP 8.1.0
   PHP 8.1, released in 2021, brings major new features such as Enums, Fibers, never return type, Intersection Types, readonly properties, and more, while ironing out some of its undesired legacy features by deprecating them.

Complete Guide to PHP Versions

PHP Version	**Release Date**	**Key Features**
PHP 8.1 (PHP current version)	November 25, 2021	Enumerations, read-only properties, noreturn type, new in initializers, final class constants, fibers (asynchronous PHP)
PHP 8.0	November 26, 2020	Major performance improvements, just-in-time compilation, nullsafe operator, match expression, constructor property promotion, union types, mixed type, static return type
PHP 7.4	November 28, 2019	Custom object serialization, null coalescing assignment operator, reflection for references, foreign function interface
PHP 7.3	December 6, 2018	Reference assignment for destructured arrays, flexible heredoc and nowdoc syntax
PHP 7.2	November 30, 2017	Object type hint, abstract function overriding
PHP 7.1	December 1, 2016	Void return type, class constant visibility
PHP 7.0	December 3, 2015	Major performance improvements, null coalescing operator, return types
PHP 5.6	August 28, 2014	Variadic functions, argument unpacking

For an exhaustive list of changes, see the PHP 5.x changelog, the PHP 7.x changelog, and the PHP 8.x changelog.

4.?

5. What Can PHP Do?

- PHP can generate dynamic page content.
- PHP can create, open, read, write, delete, and close files on the server.
- PHP can collect form data.
- PHP can send and receive cookies.
- PHP can add, delete, modify data in your database.
- PHP can be used to control user-access.
- PHP can encrypt data.

6. Features of PHP Language

Let us now dive into the unique and top features of the PHP language.

It's Open-source: Meaning free to use. It doesn't matter if you're a part-time developer or a large company with locations around the world. Because PHP is open-source, you can save a lot of money and use it where it is most needed. Another one of those advantages of PHP is this.

- **It is adaptable**: It works flawlessly when downloaded and used on Windows, Linux, or Mac OS. It also supports almost all web servers in use, and it's quite easy to develop and deploy a project that works with most operating systems right out of the box.
- **Excellent database connectivity**: PHP most likely has the ability to connect to any database that exists. For instance- Access, Firebird, MySQL, Oracle, and iBase are among the most popular ones. One of

the most crucial features and the biggest advantages of PHP language is its interoperability and cross-platform capabilities.

- **Support for third-party applications and security**: Many of PHP's built-in functions include data encryption options, making the language safer. Applications from third parties can be used to secure data by users as well.

Monitoring of real-time access: PHP additionally offers a list of users' most recent logging accesses.

- **Availability of Information on memory and CPU utilization**: is available through PHP functions like memory get usage() and memory get peak usage(), which can assist programmers to optimize their code. Similar to this, it is possible to retrieve the CPU usage statistics of any script in order to further optimise it.

- **Features of object-oriented programming**: PHP supports these features, making it faster and adding new ones like data encapsulation and inheritance at various levels.

Despite several other languages and tools coming and getting extinct, PHP has continued to remain and doing exceptionally great. The fundamentals involved in PHP remain the same along with a few enhancements in the language.

6. **With a help of examples explain why php is case sensitive?**

PHP classes are a mix between variables and functions, so they are partially case-sensitive. As you can see in the example above, the variables $num and $NUM can have different values. But when you declare two functions with the same name, PHP produces a fatal error: cannot redeclare the function

I thought I might add, for the benefit of those who think I am asking which, the following list:

Case Sensitive

Strings

Variables

Object Properties

Constants, by default

Case Insensitive

Key Words etc

Functions

Object Methods

Constants, if defined accordingly

Class Names

7. A comment in PHP code is **a line that is not executed as a part of the program**. Its only purpose is to be read by someone who is looking at the code

   Purpose of comment in between the lines between code is to make your code more understandable for later use

comments can be used to:

- Let others understand your code
- Remind yourself of what you did - Most programmers have experienced coming back to their own work a year or two later and having to re-figure out what they did. Comments can remind you of what you were thinking when you wrote the code

Example

Syntax for single-line comments:

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

<?php
// This is a single-line comment

# This is also a single-line comment
?>
```

```
</body>
</html>
```

Example

Syntax for multiple-line comments:

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

<?php
/*
This is a multiple-lines comment block
that spans over multiple
lines
*/
?>

</body>
</html>
```

Example

Using comments to leave out parts of the code:

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

<?php
// You can also use comments to leave out parts of a code line
$x = 5 /* + 15 */ + 5;
echo $x;
?>

</body>
</html>
```

Output is 10

a.echo() vs print() : echo and print are more or less the same. They are both used to output data to the screen.

The differences are small: echo has no return value while print has a return value of 1 so it can be used in expressions. echo can take multiple parameters (although such usage is rare) while print can take one argument. echo is marginally faster than print.

The PHP echo Statement

The echo statement can be used with or without parentheses: echo or echo().

Display Text

The following example shows how to output text with the echo command (notice that the text can contain HTML markup):

Example

```php
<?php
echo "<h2>PHP is Fun!</h2>";
echo "Hello world!<br>";
echo "I'm about to learn PHP!<br>";
echo "This ", "string ", "was ", "made ", "with multiple parameters.";
?>
```

Display Variables

The following example shows how to output text and variables with the echo statement:

Example

```php
<?php
$txt1 = "Learn PHP";
$txt2 = "W3Schools.com";
$x = 5;
$y = 4;

echo "<h2>" . $txt1 . "</h2>";
echo "Study PHP at " . $txt2 . "<br>";
echo $x + $y;
?>
```

The PHP print Statement

The print statement can be used with or without parentheses: print or print().

Display Text

The following example shows how to output text with the print command (notice that the text can contain HTML markup):

Example

```
<?php
print "<h2>PHP is Fun!</h2>";
print "Hello world!<br>";
print "I'm about to learn PHP!";
?>
```

Display Variables

The following example shows how to output text and variables with the print statement:

Example

```
<?php
$txt1 = "Learn PHP";
$txt2 = "W3Schools.com";
$x = 5;
$y = 4;

print "<h2>" . $txt1 . "</h2>";
print "Study PHP at " . $txt2 . "<br>";
print $x + $y;
?>
```

8. a) Echo() vs print():echo and print are more or less the same. They are both used to output data to the screen.

   The differences are small: echo has no return value while print has a return value of 1 so it can be used in expressions. echo can take multiple parameters (although such usage is rare) while print can take one argument. echo is marginally faster than print.

The PHP echo Statement

The echo statement can be used with or without parentheses: echo or echo().

Display Text

The following example shows how to output text with the echo command (notice that the text can contain HTML markup):

Example

```php
<?php
echo "<h2>PHP is Fun!</h2>";
echo "Hello world!<br>";
echo "I'm about to learn PHP!<br>";
echo "This ", "string ", "was ", "made ", "with multiple parameters.";
?>
```

The PHP print Statement

The print statement can be used with or without parentheses: print or print().

Display Text

The following example shows how to output text with the print command (notice that the text can contain HTML markup):

Example

```php
<?php
print "<h2>PHP is Fun!</h2>";
print "Hello world!<br>";
print "I'm about to learn PHP!";
?>
```

b) Print() vs printf(): The print() function **prints the specified message to the screen, or other standard output device**. The message can be a string, or any other object, the object will be converted into a string before written to the screen.

The printf() function is used to format and print a series of characters and values to the standard output.

Syntax:

```
int printf(const char *format-string, argument-list);
```

c) Printf() vs print_r(): The printf() function is used to format and print a series of characters and values to the standard output.

Syntax:

```
int printf(const char *format-string, argument-list);
```

The print_r() function prints the information about a variable in a more human-readable way.

Syntax

print_r(*variable*, *return*);

d) Print_r vs var_dump(): The print_r() function prints the information about a variable in a more human-readable way.

Syntax

print_r(*variable*, *return*);

The var_dump() function dumps information about one or more variables. The information holds type and value of the variable(s).

Syntax

var_dump(*var1*, *var2*, ...);

9. **PHP Data Types**

PHP data types are used to hold different types of data or values. PHP supports 8 primitive data types that can be categorized further in 3 types:

1. Scalar Types (predefined)
2. Compound Types (user-defined)
3. Special Types

Scalar Types

It holds only single value. There are 4 scalar data types in PHP.

1. boolean
2. integer
3. float
4. string

Boolean: Booleans are the easiest type. It can be either TRUE or FALSE. It is used in control structure like the testing portion of an if statement.

PHP Integer

Integer means numeric data with a negative or positive sign. It holds only whole numbers, i.e., numbers without fractional part or decimal points.

Rules for integer:

- An integer can be either positive or negative.
- An integer must not contain decimal point.
- Integer can be decimal (base 10), octal (base 8), or hexadecimal (base 16).
- The range of an integer must be lie between 2,147,483,648 and 2,147,483,647 i.e., -2^31 to 2^31.

Example:

```
<?php
```

```php
    $dec1 = 34;
    $oct1 = 0243;
    $hexa1 = 0x45;
    echo "Decimal number: " .$dec1. "</br>";
    echo "Octal number: " .$oct1. "</br>";
    echo "HexaDecimal number: " .$hexa1. "</br>";
?>
```

Output:

```
Decimal number: 34
Octal number: 163
HexaDecimal number: 69
```

PHP Float

A floating-point number is a number with a decimal point. Unlike integer, it can hold numbers with a fractional or decimal point, including a negative or positive sign.

PHP String

A string is a non-numeric data type. It holds letters or any alphabets, numbers, and even special characters.

String values must be enclosed either within **single quotes** or in **double quotes**. But both are treated differently.

PHP Data Types: Compound Types

It can hold multiple values. There are 2 compound data types in PHP.

1. array
2. object

array: **In PHP, there are three types of arrays:**

- Indexed arrays - Arrays with a numeric index.
- Associative arrays - Arrays with named keys.
- Multidimensional arrays - Arrays containing one or more arrays.

Object:

In PHP, Object is a **compound data type (along with arrays)**. Values of more than one types can be stored together in a single variable. Object is an instance of either a built-in or user defined class. In addition to properties, class defines functionality associated with data.

Special Types

There are 2 special data types in PHP.

1. resource
2. NULL

Resource

The special resource type is not an actual data type. It is **the storing of a reference to functions and resources external to PHP**. A common example of using the resource data type is a database call.

Null

Null is a special data type which can have only one value: NULL. A variable of data type NULL is a variable that has no value assigned to it. Tip: If a variable is created without a value, it is automatically assigned a value of NULL.

9. What Is php variables

PHP variables are **characters that stores value or information such as text or integers in your code**. It is important to know that variables in PHP are usually represented by a dollar sign($) followed by the name of the variable.

Rules for PHP variables:

- A variable starts with the $ sign, followed by the name of the variable
- A variable name must start with a letter or the underscore character
- A variable name cannot start with a number
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)

- Variable names are case-sensitive ($age and $AGE are two different variables)

10. The PHP superglobal variables are:

    $GLOBALS

    $_SERVER

    $_REQUEST

    $_POST

    $_GET

    $_FILES

    $_ENV

    $_COOKIE

    $_SESSION

HP $GLOBALS

$GLOBALS is a PHP super global variable which is used to access global variables from anywhere in the PHP script (also from within functions or methods).

PHP stores all global variables in an array called $GLOBALS[*index*]. The *index* holds the name of the variable.

The example below shows how to use the super global variable $GLOBALS:

Example

```
<?php
$x = 75;
$y = 25;

function addition() {
  $GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y'];
}
```

```
addition();
echo $z;
?>
```

PHP $_SERVER

$_SERVER is a PHP super global variable which holds information about headers, paths, and script locations.

The example below shows how to use some of the elements in $_SERVER:

Example

```
<?php
echo $_SERVER['PHP_SELF'];
echo "<br>";
echo $_SERVER['SERVER_NAME'];
echo "<br>";
echo $_SERVER['HTTP_HOST'];
echo "<br>";
echo $_SERVER['HTTP_REFERER'];
echo "<br>";
echo $_SERVER['HTTP_USER_AGENT'];
echo "<br>";
echo $_SERVER['SCRIPT_NAME'];
?>
```

PHP $_REQUEST

PHP $_REQUEST is a PHP super global variable which is used to collect data after submitting an HTML form.

The example below shows a form with an input field and a submit button. When a user submits the data by clicking on "Submit", the form data is sent to the file specified in the action attribute of the <form> tag. In this example, we point to this file itself for processing form data. If you wish to use another PHP file to process form data, replace that with the filename of your choice. Then, we can use the super global variable $_REQUEST to collect the value of the input field:

Example

```
<html>
<body>
```

```
<form method="post" action="<?php echo $_SERVER['PHP_SELF'];?>">
  Name: <input type="text" name="fname">
  <input type="submit">
</form>

<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
  // collect value of input field
  $name = $_REQUEST['fname'];
  if (empty($name)) {
    echo "Name is empty";
  } else {
    echo $name;
  }
}
?>

</body>
</html>
```

PHP $_POST

PHP $_POST is a PHP super global variable which is used to collect form data after submitting an HTML form with method="post". $_POST is also widely used to pass variables.

The example below shows a form with an input field and a submit button. When a user submits the data by clicking on "Submit", the form data is sent to the file specified in the action attribute of the <form> tag. In this example, we point to the file itself for processing form data. If you wish to use another PHP file to process form data, replace that with the filename of your choice. Then, we can use the super global variable $_POST to collect the value of the input field:

Example

```
<html>
<body>

<form method="post" action="<?php echo $_SERVER['PHP_SELF'];?>">
  Name: <input type="text" name="fname">
  <input type="submit">
```

```
</form>

<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
  // collect value of input field
  $name = $_POST['fname'];
  if (empty($name)) {
    echo "Name is empty";
  } else {
    echo $name;
  }
}
?>

</body>
</html>
```

PHP $_GET

PHP $_GET is a PHP super global variable which is used to collect form data after submitting an HTML form with method="get".

$_GET can also collect data sent in the URL.

Assume we have an HTML page that contains a hyperlink with parameters:

```
<html>
<body>

<a href="test_get.php?subject=PHP&web=W3schools.com">Test $GET</a>

</body>
</html>
```

When a user clicks on the link "Test $GET", the parameters "subject" and "web" are sent to "test_get.php", and you can then access their values in "test_get.php" with $_GET.

The example below shows the code in "test_get.php":

Example

```
<html>
<body>

<?php
echo "Study " . $_GET['subject'] . " at " . $_GET['web'];
?>

</body>
</html>
```