# Everlytic Developer Assessment

|  |  |
| --- | --- |
| Name | Thembinkosi Fumba |

|  |  |
| --- | --- |
| Date | 2023/02/06 |

Please complete the answers to the questions below. The assessment should take roughly 30 minutes.

### What is the difference between public, protected and private in a class definition?

|  |
| --- |
| Public class - can be accessed from anywhere, inside or outside the class |
| Protected class - Can be accessed within the class and its subclasses. |
| Private class - Can only be accessed within the class and is not accessible from outside the class or from subclasses. |

### Given this code: function doSomething(&$foo) { $bar = $foo; $foo += 1; return $foo; } $value = 3; $result = doSomething($value); echo "value: $value, result: $result"; What will be output to screen and why?

|  |
| --- |
| Output – ‘value: 4, result: 4’ |
| Reason – the variable value is being passed by ref, This allows you to modify the original variable within the function, and the changes will persist outside of the function. |
|  |

### What is wrong with this query: "SELECT \* FROM table WHERE id = $\_POST[ 'id' ]"?

|  |
| --- |
| The query is vulnerable to SQL injection attacks - The query should be sanitized to ensure that only expected data is used in the query, and the data type of the id value should be validated before being used in the query. |
|  |
|  |

### What is the cause of this warning: 'Warning: Cannot modify header information - headers already sent', and what is a good practice to prevent it?

|  |
| --- |
| By outputting content (e.g. HTML, whitespace, etc.) before sending HTTP headers. |
|  |
|  |

### What is wrong with this code: class Foo { protected $bar; public function \_\_construct() { $this->bar = 1; } public static function doSomething() { return $this->bar; } }

|  |
| --- |
| The code has an error in the **doSomething** method. The method is declared as static, When the method is called, **$this** refers to the class itself, not to an instance of the class, so **$this->bar** will cause an error. |
|  |
|  |

### Write a program that prints the numbers from 1 to 100. But for multiples of three print "Fizz" instead of the number and for the multiples of five print "Buzz". For numbers which are multiples of both three and five print "FizzBuzz".

for ($i = 1; $i <= 100; $i++) {

if ($i % 3 == 0 && $i % 5 == 0) {

echo "FizzBuzz\n";

} elseif ($i % 3 == 0) {

echo "Fizz\n";

} elseif ($i % 5 == 0) {

echo "Buzz\n";

} else {

echo "$i\n";

}

}

### **What does the following code do? Explain what’s going on there.**

$date = '08/26/2003';

print preg\_replace('/([0-9]+)\/([0-9]+)\/([0-9]+)/'‚ '$2/$1/$3', $date);

|  |
| --- |
| Using the regular expression-based search-and-replace function **preg\_replace** will replace the date string with a new string in the **dd/mm/yyyy** format, and the print statement will output the result. |
|  |

### **Given a line of text $string, how would you write a regular expression to strip all the HTML tags from it?**

|  |
| --- |
| $string = preg\_replace('/<[^>]\*>/', '', $string); |
|  |
|  |

### A palindrome is a word that reads the same backward or forward. Write a function that checks is a given word is a palindrome. Characters case should be ignored. EG. Deleveled is a palindrome and should return true as character case is ignored.

### <?php class Palindrome

### {

### public static function isPalindrome($word)

{  
 $word = strtolower($word);

$reversedWord = strtolower(strrev($word));

return $word == $reversedWord;

}

### } echo Palindrome::isPalindrome('Deleveled');

### Considering message\_text stores a combination of html and text. What security issue is prevalent in the code below and how would you fix it? <?php $messageStmt = $db->query('select message\_text from messages where message\_id = 1'); $messageStmt->execute(); $message = $messageStmt->fetch(PDO::FETCH\_OBJ); ?> <div><?php echo $message->message\_text; ?></div> \_ The code is vulnerable to a security issue known as **cross-site scripting (XSS).** To fix this issue, the user-supplied data should be sanitized before being displayed. This can be done using the **htmlspecialchars()** function.

### **<div><?php echo htmlspecialchars($message->message\_text); ?></div>\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

### Write an inner join for the following tables

|  |
| --- |
| SELECT \*  FROM User  INNER JOIN Address  ON User.UserKey = Address.UsrKey; |
|  |

### Complete the JS function below that validates the conditions of a password: 1. The password must be greater then 7 characters 2. The first character must be a capital letter 3. The password must contain at least one number

function isPasswordValid($password) {

var isValid = false;

if (password.length > 7) {

if (/^[A-Z]/.test(password)) {

if (/\d/.test(password)) {

isValid = true;

}

}

}

return isValid;

}