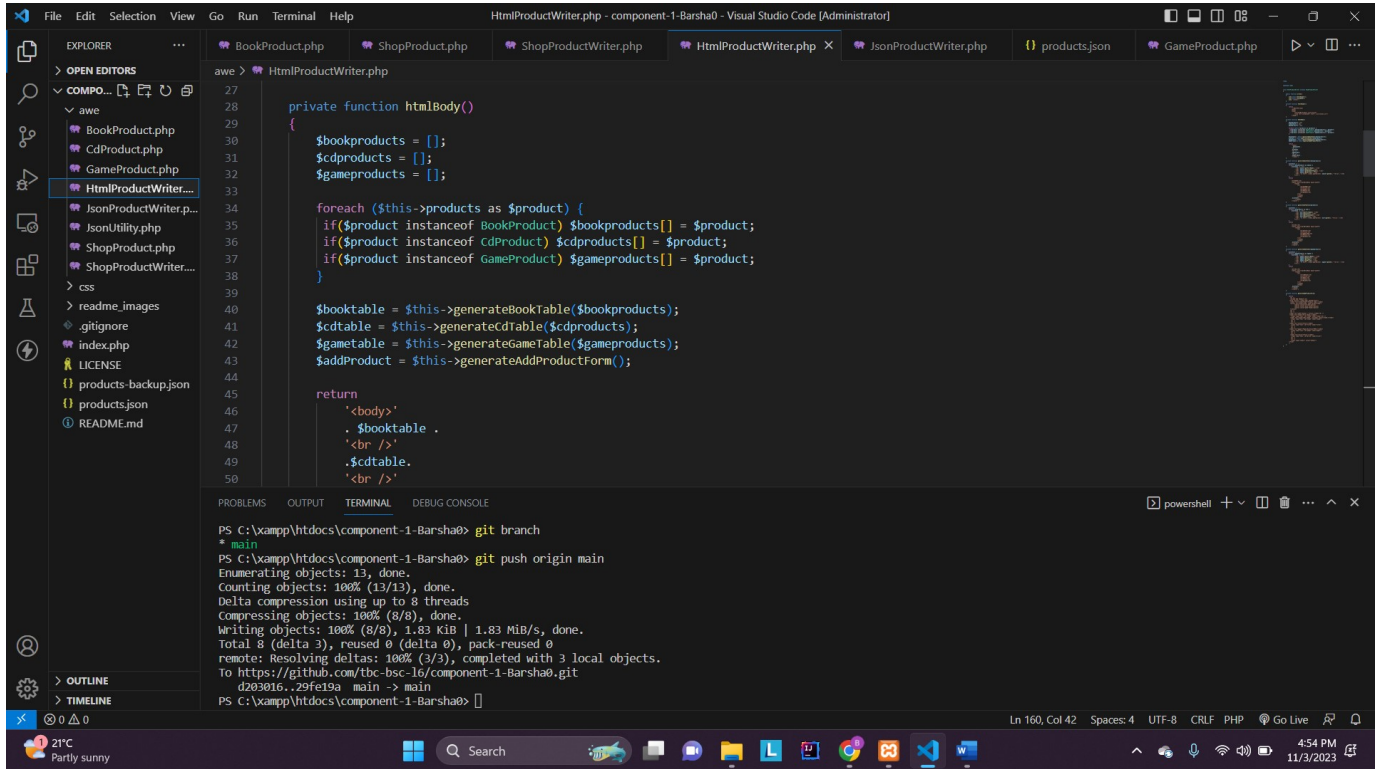


COMPONENT-1 DOCUMENTATION (BARSHA DHAKAL)

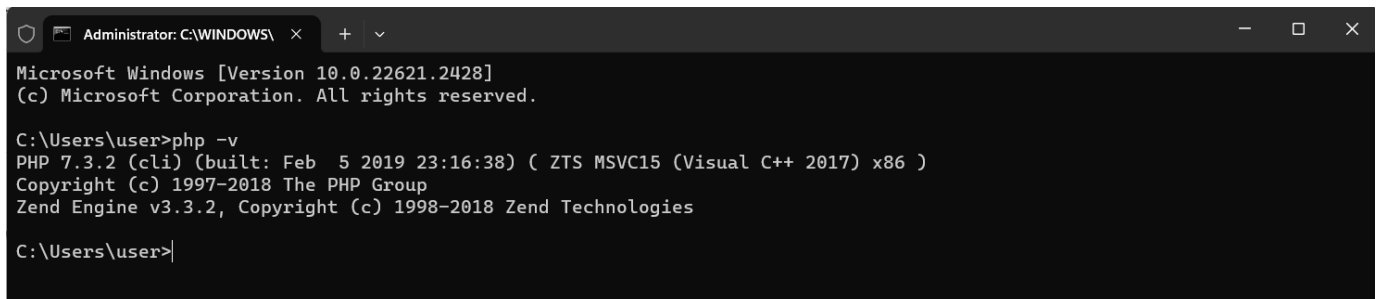
1. DEVELOPMENT CHOICE : **VS Code** (sublime/php storm/vs code/others ?)

screenshot of your IDE with component 1 project code open in the IDE.



2. PHPVERSION being used : **7.3.2** (fill with the php version in your system/xampp)

screenshot of output of php -v in command line here.



3. Github account component 1 repository link :

<https://github.com/tbc-bsc-l6/component-1-Barsha0.git>

(your component 1 repo link here)

No screenshot needed for this

4. PHP Exercise in component 1 completed ? **YES** (yes, no, partial)

add multiple(2-4) screenshots of the frontend and source codes showing the changes

you made in component 1

The screenshot shows a web browser window with the URL `localhost/component-1-Barsha0/index.php`. The page displays three tables of product data:

AUTHOR	TITLE	PAGES	PRICE	DELETE
Matt Zandstra	PHP Objects, Patterns and Practice	608	44.99	X

CDs

ARTIST	TITLE	DURATION	PRICE	DELETE
Georgia	Seeking Thrills	45	8.5	X

CDs

CONSOLE	TITLE	PEGI	PRICE	DELETE
Nintendo Switch	Animal crossing : New Horizons	3	44.99	X

ADD NEW PRODUCT

Product Type:

Author / Artist:

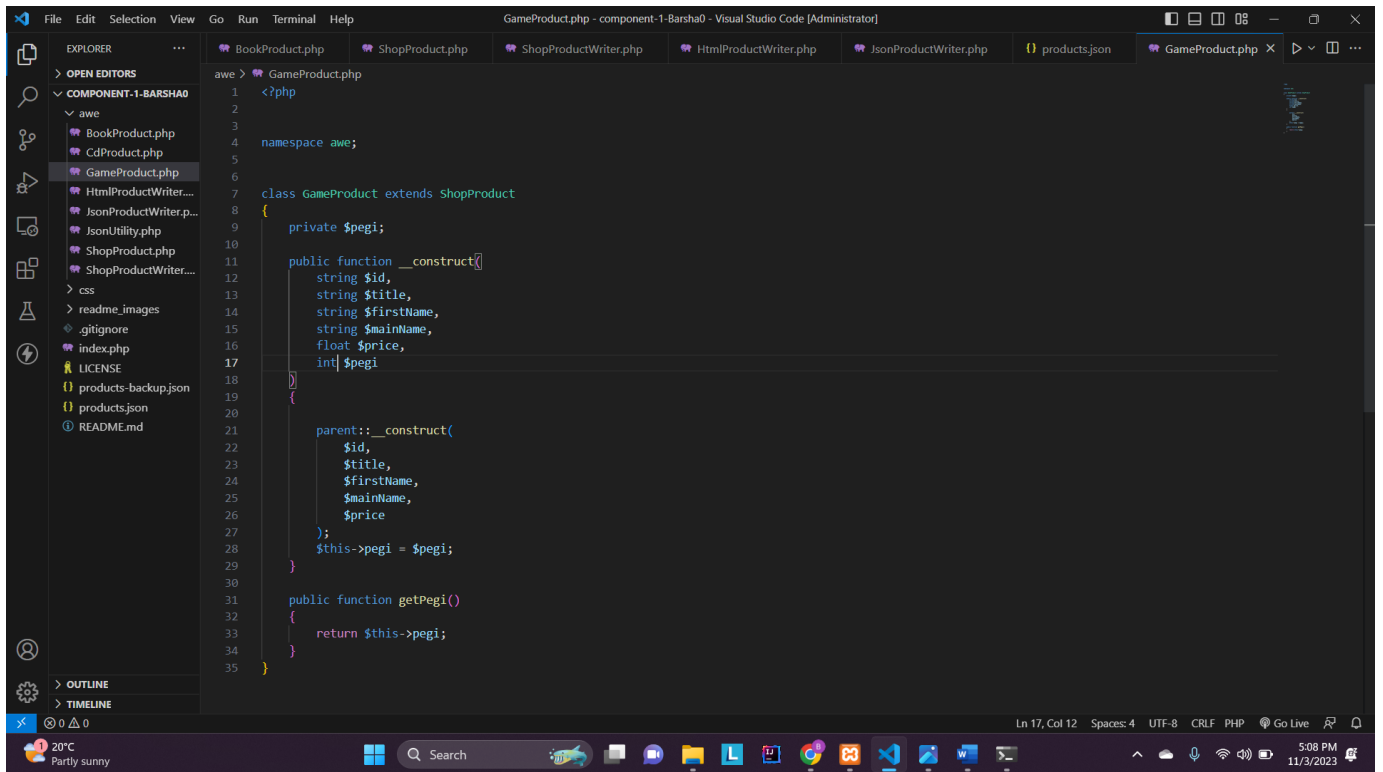
First Name:

Main Name / Surname/ Console Name:

Title:

Pages/Duration/PEGI:

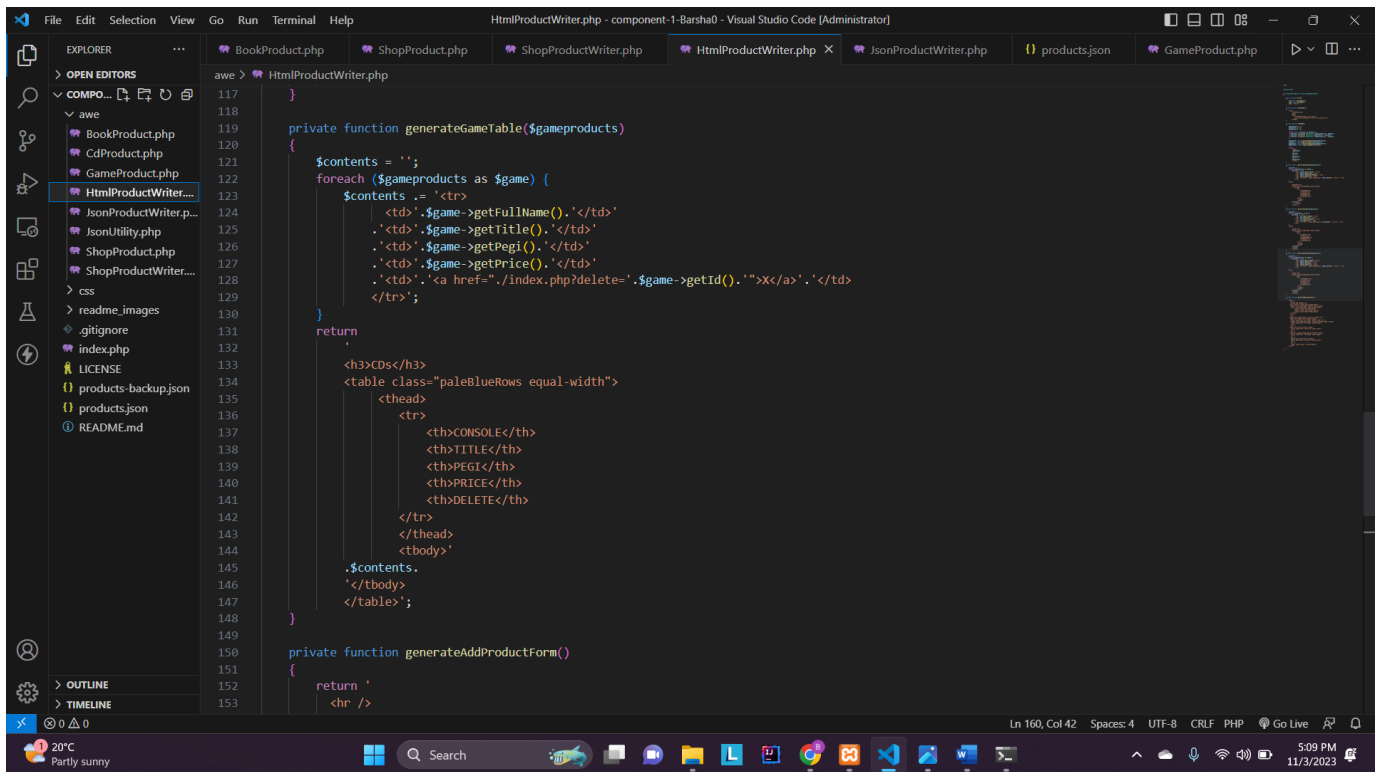
Price:



This screenshot shows the Visual Studio Code editor with the file `GameProduct.php` open. The Explorer sidebar on the left shows a project structure for `COMPONENT-1-BARSHAO` with various PHP files and JSON files. The main editor area displays the code for `GameProduct.php`, which includes a namespace declaration, a class definition extending `ShopProduct`, and methods for constructor, getter, and setter.

```
1 <?php
2
3
4 namespace awe;
5
6
7 class GameProduct extends ShopProduct
8 {
9     private $pegi;
10
11     public function __construct(
12         string $id,
13         string $title,
14         string $firstName,
15         string $mainName,
16         float $price,
17         int $pegi
18     ) {
19
20         parent::__construct(
21             $id,
22             $title,
23             $firstName,
24             $mainName,
25             $price
26         );
27         $this->pegi = $pegi;
28     }
29
30     public function getPegi()
31     {
32         return $this->pegi;
33     }
34
35 }
```

The status bar at the bottom indicates the cursor is at line 17, column 12, with 4 spaces, UTF-8 encoding, CRLF line endings, and PHP language mode.



This screenshot shows the Visual Studio Code editor with the file `HtmlProductWriter.php` open. The Explorer sidebar on the left shows the same project structure. The main editor area displays the code for `HtmlProductWriter.php`, which includes a private function `generateGameTable` that iterates over a list of game products and generates an HTML table. It also includes a private function `generateAddProductForm`.

```
117 }
118
119 private function generateGameTable($gameproducts)
120 {
121     $contents = '';
122     foreach ($gameproducts as $game) {
123         $contents .= '<tr>
124             <td>'. $game->getFullName(). '</td>'
125             . '<td>'. $game->getTitle(). '</td>'
126             . '<td>'. $game->getPegi(). '</td>'
127             . '<td>'. $game->getPrice(). '</td>'
128             . '<td>'. '<a href= "/index.php?delete=" . $game->getId(). ">X</a>' . '</td>'
129             . '</tr>';
130     }
131     return
132     <h3>CDs</h3>
133     <table class="paleBlueRows equal-width">
134         <thead>
135             <tr>
136                 <th>CONSOLE</th>
137                 <th>TITLE</th>
138                 <th>PEGI</th>
139                 <th>PRICE</th>
140                 <th>DELETE</th>
141             </tr>
142         </thead>
143         <tbody>
144             <tbody>
145             </tbody>
146         </tbody>
147     </table>';
148 }
149
150 private function generateAddProductForm()
151 {
152     return '
153     <hr />
154 }
```

The status bar at the bottom indicates the cursor is at line 160, column 42, with 4 spaces, UTF-8 encoding, CRLF line endings, and PHP language mode.

```
8
9
10 public function write()
11 {
12     $json_str = '[';
13     foreach ($this->products as $product) {
14         $json_str .= $this->addEachProductAsJSON($product).',';
15     }
16     $json_str = rtrim($json_str, ","); //remove final ',' from outputted json string
17
18     $json_str .= "];";
19     echo $json_str;
20 }
21
22 private function addEachProductAsJSON($product){
23     $json_product = [];
24     $json_product['id'] = $product->getId();
25     $json_product['title'] = $product->getTitle();
26     $json_product['firstname'] = $product->getFirstName();
27     $json_product['mainname'] = $product->getMainName();
28     $json_product['price'] = $product->getPrice();
29
30     if($product instanceof BookProduct) {
31         $json_product['numpages'] = $product->getNumberOfPages();
32         $json_product['type'] = "book";
33     }
34     if($product instanceof CdProduct) {
35         $json_product['playlength'] = $product->getPlayLength();
36         $json_product['type'] = "cd";
37     }
38     if($product instanceof GameProduct) {
39         $json_product['pegi'] = $product->getPegi();
40         $json_product['type'] = "game";
41     }
42
43     return json_encode($json_product);
44 }
```

```
1 <?php
2
3 namespace awe;
4
5
6
7 class JsonUtility
8 {
9     public static function makeProductArray(string $file) {
10         $string = file_get_contents($file);
11
12         $productsJson = json_decode($string, true);
13
14         $products = [];
15         foreach ($productsJson as $product) {
16             switch($product['type']) {
17                 case "cd":
18                     $cdproduct = new \awe\CdProduct($product['id'], $product['title'], $product['firstname'],
19                         $product['mainname'], $product['price'], $product['playlength']);
20                     $products[] = $cdproduct;
21                     break;
22                 case "book":
23                     $bookproduct = new \awe\BookProduct($product['id'], $product['title'], $product['firstname'],
24                         $product['mainname'], $product['price'], $product['numpages']);
25                     $products[] = $bookproduct;
26                     break;
27                 case "game":
28                     $gameproduct = new \awe\GameProduct($product['id'], $product['title'], $product['firstname'],
29                         $product['mainname'], $product['price'], $product['pegi']);
30                     $products[] = $gameproduct;
31                     break;
32             }
33         }
34         return $products;
35     }
36
37     public static function deleteProductWithId(string $file, int $id) {
```

Export the completed version of this document as .doc , .docx or .pdf file and push to the remote repository of your component 1.

All the best.