ECS417U Project Phase 2 Documentation

|  |  |
| --- | --- |
| **Name:** | Rayaan Mubashir Uddin |
| **Student Number:** | 230335705 |

|  |  |
| --- | --- |
| **Assessment Criteria 7**  Each time a new entry is added, the most recent post appears on top, followed by the next most recent post using a PHP-driven technique (writing a sorting algorithm) instead of a SQL Query. | |
| Instruction: use this column to paste the code from the files that have been used to develop this functionality.  **Paste the code directly from visual studio code, do not include a screenshot of the code as this document has a Turnitin requirement and this will lead to an automatic mark of zero. See example below.** | Instruction: use this column to address the following in your own words:   1. Describe the steps you took to develop this feature (including the prompts). 2. Describe how the sorting algorithm works.   Give reference to the code you have written when answering the questions above.  You can use diagrams (e.g. flow charts) or screenshots of your webpage output to illustrate your answer. |
| Filename: database.php  <?php  session\_start();  $db = new mysqli('localhost', 'RayaanUddin', 'Rayaan10', 'portfolio');  $posts = array();  $permission\_levels = array(  0 => "User",  1 => "Admin"  );  class Account {  public $fname;  public $lname;  public $email;  public $id;  private $permission;  public function \_\_construct() {  global $db;  if ($db->connect\_error) {  die("Connection failed: " . $db->connect\_error);  }  }  public function getFullname() {  return $this->fname." ".$this->lname;  }  private function accountExists($email) {  global $db;  $sql = "SELECT \* FROM accounts WHERE email = '".$email."'";  $result = $db->query($sql);  if ($result->num\_rows === 1) {  return true;  } else {  return false;  }  }  public function register($fname, $lname, $email, $password, $perm) {  global $db;  if ($this->accountExists($email)) {  return false;  } else {  $sql = "INSERT INTO accounts (fname, lname, email, password, permission) VALUES ('".$fname."', '".$lname."', '".$email."', '".$password."', '".$perm."')";  if ($db->query($sql)) {  return true;  } else {  return false;  }  }  }  public function login($email, $password) {  global $db;  $sql = "SELECT \* FROM accounts WHERE email = '".$email."' AND password = '".$password."'";  $result = $db->query($sql);  if ($result->num\_rows === 1) {  $row = $result->fetch\_assoc();  $this->fname = $row["fname"];  $this->lname = $row["lname"];  $this->email = $row['email'];  $this->id = $row['accountId'];  $this->permission = $row['permission'];  return true;  } else {  return false;  }  }  public function isAdmin() {  global $permission\_levels;  if ($permission\_levels[$this->permission] == "Admin") {  return true;  } else {  return false;  }  }  }  class Comment {  public $comment;  public $accountId;  public $date;  public $authorFullname;  public $id;  public function \_\_construct($comment, $author, $date, $id) {  $this->comment = $comment;  $this->accountId = $author;  $this->date = strtotime($date);  $this->id = $id;  global $db;  // Get author of comment name  $sql = "SELECT fname, lname FROM accounts WHERE accountId = '".$author."'";  $result = $db->query($sql);  if ($result->num\_rows === 1) {  $row = $result->fetch\_assoc();  $this->authorFullname = $row["fname"]." ".$row["lname"];  } else {  $this->authorFullname = "Unknown author";  }  }  public function get\_date\_str() {  $datestr = "";  $yeardiff = date('Y') - date('Y', $this->date);  $monthdiff = date("m") - date("m", $this->date);  if ($monthdiff < 0) {  $yeardiff--;  $monthdiff+=12;  }  $datediff = date("d") - date("d", $this->date);  if ($datediff < 0) {  $monthdiff--;  $datediff+=30;  }  $hourdiff = date("H") - date("H", $this->date);  if ($hourdiff < 0) {  $datediff--;  $hourdiff+=24;  }  $minutediff = date("i") - date("i", $this->date);  if ($minutediff < 0) {  $hourdiff--;  $minutediff+=60;  }  $seconddiff = date("s") - date("s", $this->date);  if ($seconddiff < 0) {  $minutediff--;  $seconddiff+=60;  }  if ($yeardiff > 0) {  $datestr .= $yeardiff." year".($yeardiff > 1 ? "s " : " ");  } elseif ($monthdiff > 0) {  $datestr .= $monthdiff." month".($monthdiff > 1 ? "s " : " ");  } elseif ($datediff > 0) {  $datestr .= $datediff." day".($datediff > 1 ? "s " : " ");  } elseif ($hourdiff > 0) {  $datestr .= $hourdiff." hour".($hourdiff > 1 ? "s " : " ");  } elseif ($minutediff > 0) {  $datestr .= $minutediff." minute".($minutediff > 1 ? "s " : " ");  } elseif ($seconddiff > 0) {  $datestr .= $seconddiff." second".($seconddiff > 1 ? "s " : " ");  }  if ($datestr == "") {  return "Just now";  }  return $datestr."ago";  }  }  class Post {  public $title;  public $content;  public $accountId;  public $date;  public $id;  public $authorFullname;  public $comments;  private function getComments() {  global $db;  $sql = "SELECT \* FROM comments WHERE blogId = '".$this->id."'";  $result = $db->query($sql);  $comments = array();  if ($result->num\_rows > 0) {  while ($row = $result->fetch\_assoc()) {  $comment = new Comment($row['comment'], $row['accountId'], $row['date'], $row['commentId']);  array\_push($comments, $comment);  }  }  return $comments;  }  public function get\_date\_str() {  return date('dS F Y | H:i', $this->date);  }  public function \_\_construct($title, $content, $author, $date, $id) {  $this->title = $title;  $this->content = $content;  $this->accountId = $author;  $this->date = $date;  $this->id = $id;  global $db;  // Get author of post name  $sql = "SELECT fname, lname FROM accounts WHERE accountId = '".$author."'";  $result = $db->query($sql);  if ($result->num\_rows === 1) {  $row = $result->fetch\_assoc();  $this->authorFullname = $row["fname"]." ".$row["lname"];  } else {  $this->authorFullname = "Unknown author";  }  // Get posts comments  $this->comments = orderByDateDesc($this->getComments());  }  }  function updatePosts() {  global $db;  global $posts;  $sql = "SELECT \* FROM blog";  $result = $db->query($sql);    if ($result->num\_rows > 0) {  while ($row = $result->fetch\_assoc()) {  $date = strtotime($row['date']);  $post = new Post($row['title'], $row['content'], $row['accountId'], $date, $row['blogId']);  array\_push($posts, $post);  }  }  return $posts;  }  function orderByDateDesc($objectArray) {  // Custom sorting function to sort objects by date in descending order  usort($objectArray, function($a, $b) {  return strcmp($b->date, $a->date);  });  // Return the sorted array  return $objectArray;  }  function addPost($title, $content, $author) {  global $db;  $date = date('Y-m-d H:i:s');  $sql = "INSERT INTO blog (title, content, accountId, date) VALUES ('$title', '$content', '$author', '$date')";  if ($db->query($sql) == true) {  return true;  } else {  return false;  }  }  function getPost($id) {  global $db;  $sql = "SELECT \* FROM blog WHERE blogId = '".$id."'";  $result = $db->query($sql);  if ($result->num\_rows === 1) {  $row = $result->fetch\_assoc();  $dateFromSQL = strtotime($row['date']);  $date = date('dS F Y | H:i', $dateFromSQL);  $post = new Post($row['title'], $row['content'], $row['accountId'], $date, $row['blogId']);  return $post;  } else {  return null;  }  }  function addComment($id, $content, $author) {  global $db;  $date = date('Y-m-d H:i:s');  $sql = "INSERT INTO comments (comment, blogId, accountId, date) VALUES ('$content', '$id', '$author', '$date')";  if ($db->query($sql) == true) {  echo "Comment added";  } else {  echo "Error: " . $sql . "<br>" . $db->error;  }  }  function deleteComment($id) {  global $db;  if ($\_SESSION["loggedIn"] == true) {  if ($\_SESSION["account"]->isAdmin()) {  $sql = "DELETE FROM comments WHERE commentId = '".$id."'";  if ($db->query($sql) == true) {  return true;  } else {  return false;  }  } else {  $accountId = $\_SESSION["account"]->id;  $sql = "SELECT commentId FROM comments WHERE commentId = '".$id."' AND accountId = '".$accountId."'";  $result = $db->query($sql);  if ($result->num\_rows === 1) {  $sql = "DELETE FROM comments WHERE commentId = '".$id."' AND accountId = '".$accountId."'";  if ($db->query($sql) == true) {  return true;  } else {  return false;  }  } else {  return false;  }  }  } else {  return false;  }  }  function deletePost($id) {  global $db;  if ($\_SESSION["loggedIn"] == true) {  if ($\_SESSION["account"]->isAdmin()) {  $sql = "DELETE FROM blog WHERE blogId = '".$id."'";  if ($db->query($sql) == true) {  $sql = "DELETE FROM comments WHERE blogId = '".$id."'";  if ($db->query($sql) == true) {  return true;  } else {  return false;  }  } else {  return false;  }  } else {  return false;  }  } else {  return false;  }  }  ?>  Filename: viewBlog.php  <?php  include 'scripts/php/database.php';  ?>  <!DOCTYPE html>  <html lang="en">  <?php include 'scripts/php/head.php'; ?>  <body>  <div class="container">  <?php  include 'scripts/php/header.php';  ?>  <section id="blog-error">  <?php  if (isset($\_GET['error'])) {  echo "<p>Error: ".$\_GET['error']."</p>";  }  ?>  </section>  <?php  if (isset($\_GET['id'])) {  $post = getPost($\_GET['id']);  if ($post != null) {  echo "<article id='blog\_full'>";  echo "<nav id='blog-nav'>";  echo "<a href='viewBlog.php#".$\_GET['id']."'><i class='fa fa-arrow-left'></i><p>Back</p></a>";  echo "</nav>";  echo "<time datetime='".$post->date."'>".$post->date."</time>";  echo "<h3>".$post->title."</h3>";  echo "<p class='content'>".$post->content."</p>";  echo "<p class='author'>Author: ".$post->authorFullname."</p>";  echo "<div id='comments'>";  echo "<form action='scripts/php/addComment.php' method='get'>";  echo "<textarea name='comment' placeholder='Enter comment here' required></textarea>";  echo "<input type='hidden' name='id' value='".$post->id."' required>";  echo "<input type='submit' value='Send'>";  echo "</form>";  echo "<ul id='comments\_list'>";  for ($i = 0; $i < count($post->comments); $i++) {  echo "<li class='comment'>";  echo "<p class='comment\_author'>".$post->comments[$i]->authorFullname."</p>";  echo "<p class='comment\_content'>".$post->comments[$i]->comment."</p>";  echo "<time datetime='".$post->comments[$i]->get\_date\_str()."'>".$post->comments[$i]->get\_date\_str()."</time>";  if (isset($\_SESSION['account']) && ($post->comments[$i]->accountId == $\_SESSION['account']->id || $\_SESSION['account']->isAdmin())) {  echo '<a class="post\_comment\_delete" href="scripts/php/deleteBlog.php?commentId='.$post->comments[$i]->id.'&postId='.$\_GET['id'].'"><i class="fa fa-trash"></i></a>';  }  echo "</li>";  }  echo "</ul>";  echo "</div>";  echo "</article>";  }  else {  echo "<p>Post not found</p>";  }  }  else {  $posts = orderByDateDesc(updatePosts());  if (isset($\_GET['title']) && isset($\_GET['content']) && isset($\_SESSION['account']) && $\_SESSION['account']->isAdmin()) {  ?>  <nav id="blog-nav">  <a href="addEntry.php?title=<?php echo $\_GET['title']; ?>&content=<?php echo $\_GET['content']; ?>"><i class='fa fa-arrow-left'></i><p>Back</p></a>  <p>Preview</p>  <a href="scripts/php/addPost.php?title=<?php echo $\_GET['title']; ?>&content=<?php echo $\_GET['content']; ?>"><p>Post</p><i class='fa fa-check'></i></a>  </nav>  <a id='addpost\_button' href='addEntry.php'>Add Post</a>  <section id='blog'>  <article id="preview">  <time>Preview</time>  <h3><?php echo $\_GET['title']; ?></h3>  <p class="content"><?php echo $\_GET['content']; ?></p>  <p class="author">Author: <?php echo $\_SESSION['account']->getFullname(); ?></p>  </article>  <?php } else { ?>  <a id='addpost\_button' href='addEntry.php'>Add Post</a>  <section id='blog'>  <?php }  for ($i = 0; $i < count($posts); $i++) {  echo "<article id=".$posts[$i]->id." >";  echo "<time datetime='".$posts[$i]->date."'>".$posts[$i]->get\_date\_str()."</time>";  echo "<h3>".$posts[$i]->title."</h3>";  echo "<p class='content'>".$posts[$i]->content."</p>";  if (isset($\_SESSION['account']) && ($posts[$i]->accountId == $\_SESSION['account']->id || $\_SESSION['account']->isAdmin())) {  echo '<a class="post\_comment\_delete" href="scripts/php/deleteBlog.php?postId='.$posts[$i]->id.'"><i class="fa fa-trash"></i></a>';  }  echo "<p class='author'>Author: ".$posts[$i]->authorFullname."</p>";  echo "<a class='comments\_button' href='viewBlog.php?id=".$posts[$i]->id."'><i class='fa fa-comments-o'></i><p>".count($posts[$i]->comments)."</p></a>";  echo "</article>";  }  if (count($posts) === 0) { ?>  <article id="no\_posts">  <p class="content">No posts found</p>  </article>  <?php }  echo "</section>";  }  include 'scripts/php/footer.php';  ?>  </div>  </body>  </html> | The function orderByDateDesc was generated using AI Prompt. The prompt was the following…  “In the programming language php, write a sorting algorithm as a function, called orderByDateDesc, with an object array as parameter. The object has the attribute $date. Order the array in descending order by $date. Then return that object. $date is a variable of type time, read from the database and already converted to time from str, in the form Y-m-d H:i:s”.  It takes an array of objects, in this case, instances of Post or Comment. It uses the php usort function to sort the array in comparison to the function given. Usort function is a function that compares pairs of elements of the array and sorts them in order using the function given by programmer (AI), which should return a negative integer if first argument should be placed before second, zero if both are equal and positive integer if first argument should be placed after second. The function given by the AI to usort returns strcmp(second argument date , first argument date) which returns negative if first date argument is less than second and so on. The reason AI has generated it the argument passed in this order as it was requested to order it in descending order in the prompt.  The function is then used to sort Post read from the database in viewBlog.php, when the blog page outputs the posts, when “($\_GET['id']” is undefined.    Figure 1: Before using sorting algorithm.    Figure 2: After using the sorting algorithm. |

|  |  |
| --- | --- |
| **Assessment Criteria 8 - Extra Feature 1**  (Note: delete any of the two features below)   * Allow blog viewers to log in and add comments to the entries. You, as the administrator of the blog should be able to delete entries or comments. | |
| Instruction: use this column to paste the code from the files that have been used to develop this functionality.  **Paste the code directly from visual studio code, do not include a screenshot of the code as this document has a Turnitin requirement and this will lead to an automatic mark of zero. See example below.** | Instruction: use this column to address the following in your own words:   1. How did you develop this feature? Describe the steps involved. 2. Describe the inner workings of the approach you developed.   Give reference to the code you have written when answering the questions above.  You can use diagrams (e.g. flow charts) or screenshots of your webpage output to illustrate your answer. |
| Filename: database.php  <?php  session\_start();  $db = new mysqli('localhost', 'RayaanUddin', 'Rayaan10', 'portfolio');  $posts = array();  $permission\_levels = array(  0 => "User",  1 => "Admin"  );  class Account {  public $fname;  public $lname;  public $email;  public $id;  private $permission;  public function \_\_construct() {  global $db;  if ($db->connect\_error) {  die("Connection failed: " . $db->connect\_error);  }  }  public function getFullname() {  return $this->fname." ".$this->lname;  }  private function accountExists($email) {  global $db;  $sql = "SELECT \* FROM accounts WHERE email = '".$email."'";  $result = $db->query($sql);  if ($result->num\_rows === 1) {  return true;  } else {  return false;  }  }  public function register($fname, $lname, $email, $password, $perm) {  global $db;  if ($this->accountExists($email)) {  return false;  } else {  $sql = "INSERT INTO accounts (fname, lname, email, password, permission) VALUES ('".$fname."', '".$lname."', '".$email."', '".$password."', '".$perm."')";  if ($db->query($sql)) {  return true;  } else {  return false;  }  }  }  public function login($email, $password) {  global $db;  $sql = "SELECT \* FROM accounts WHERE email = '".$email."' AND password = '".$password."'";  $result = $db->query($sql);  if ($result->num\_rows === 1) {  $row = $result->fetch\_assoc();  $this->fname = $row["fname"];  $this->lname = $row["lname"];  $this->email = $row['email'];  $this->id = $row['accountId'];  $this->permission = $row['permission'];  return true;  } else {  return false;  }  }  public function isAdmin() {  global $permission\_levels;  if ($permission\_levels[$this->permission] == "Admin") {  return true;  } else {  return false;  }  }  }  class Comment {  public $comment;  public $accountId;  public $date;  public $authorFullname;  public $id;  public function \_\_construct($comment, $author, $date, $id) {  $this->comment = $comment;  $this->accountId = $author;  $this->date = strtotime($date);  $this->id = $id;  global $db;  // Get author of comment name  $sql = "SELECT fname, lname FROM accounts WHERE accountId = '".$author."'";  $result = $db->query($sql);  if ($result->num\_rows === 1) {  $row = $result->fetch\_assoc();  $this->authorFullname = $row["fname"]." ".$row["lname"];  } else {  $this->authorFullname = "Unknown author";  }  }  public function get\_date\_str() {  $datestr = "";  $yeardiff = date('Y') - date('Y', $this->date);  $monthdiff = date("m") - date("m", $this->date);  if ($monthdiff < 0) {  $yeardiff--;  $monthdiff+=12;  }  $datediff = date("d") - date("d", $this->date);  if ($datediff < 0) {  $monthdiff--;  $datediff+=30;  }  $hourdiff = date("H") - date("H", $this->date);  if ($hourdiff < 0) {  $datediff--;  $hourdiff+=24;  }  $minutediff = date("i") - date("i", $this->date);  if ($minutediff < 0) {  $hourdiff--;  $minutediff+=60;  }  $seconddiff = date("s") - date("s", $this->date);  if ($seconddiff < 0) {  $minutediff--;  $seconddiff+=60;  }  if ($yeardiff > 0) {  $datestr .= $yeardiff." year".($yeardiff > 1 ? "s " : " ");  } elseif ($monthdiff > 0) {  $datestr .= $monthdiff." month".($monthdiff > 1 ? "s " : " ");  } elseif ($datediff > 0) {  $datestr .= $datediff." day".($datediff > 1 ? "s " : " ");  } elseif ($hourdiff > 0) {  $datestr .= $hourdiff." hour".($hourdiff > 1 ? "s " : " ");  } elseif ($minutediff > 0) {  $datestr .= $minutediff." minute".($minutediff > 1 ? "s " : " ");  } elseif ($seconddiff > 0) {  $datestr .= $seconddiff." second".($seconddiff > 1 ? "s " : " ");  }  if ($datestr == "") {  return "Just now";  }  return $datestr."ago";  }  }  class Post {  public $title;  public $content;  public $accountId;  public $date;  public $id;  public $authorFullname;  public $comments;  private function getComments() {  global $db;  $sql = "SELECT \* FROM comments WHERE blogId = '".$this->id."'";  $result = $db->query($sql);  $comments = array();  if ($result->num\_rows > 0) {  while ($row = $result->fetch\_assoc()) {  $comment = new Comment($row['comment'], $row['accountId'], $row['date'], $row['commentId']);  array\_push($comments, $comment);  }  }  return $comments;  }  public function get\_date\_str() {  return date('dS F Y | H:i', $this->date);  }  public function \_\_construct($title, $content, $author, $date, $id) {  $this->title = $title;  $this->content = $content;  $this->accountId = $author;  $this->date = $date;  $this->id = $id;  global $db;  // Get author of post name  $sql = "SELECT fname, lname FROM accounts WHERE accountId = '".$author."'";  $result = $db->query($sql);  if ($result->num\_rows === 1) {  $row = $result->fetch\_assoc();  $this->authorFullname = $row["fname"]." ".$row["lname"];  } else {  $this->authorFullname = "Unknown author";  }  // Get posts comments  $this->comments = orderByDateDesc($this->getComments());  }  }  function updatePosts() {  global $db;  global $posts;  $sql = "SELECT \* FROM blog";  $result = $db->query($sql);    if ($result->num\_rows > 0) {  while ($row = $result->fetch\_assoc()) {  $date = strtotime($row['date']);  $post = new Post($row['title'], $row['content'], $row['accountId'], $date, $row['blogId']);  array\_push($posts, $post);  }  }  return $posts;  }  function orderByDateDesc($objectArray) {  // Custom sorting function to sort objects by date in descending order  usort($objectArray, function($a, $b) {  return strcmp($b->date, $a->date);  });  // Return the sorted array  return $objectArray;  }  function addPost($title, $content, $author) {  global $db;  $date = date('Y-m-d H:i:s');  $sql = "INSERT INTO blog (title, content, accountId, date) VALUES ('$title', '$content', '$author', '$date')";  if ($db->query($sql) == true) {  return true;  } else {  return false;  }  }  function getPost($id) {  global $db;  $sql = "SELECT \* FROM blog WHERE blogId = '".$id."'";  $result = $db->query($sql);  if ($result->num\_rows === 1) {  $row = $result->fetch\_assoc();  $dateFromSQL = strtotime($row['date']);  $date = date('dS F Y | H:i', $dateFromSQL);  $post = new Post($row['title'], $row['content'], $row['accountId'], $date, $row['blogId']);  return $post;  } else {  return null;  }  }  function addComment($id, $content, $author) {  global $db;  $date = date('Y-m-d H:i:s');  $sql = "INSERT INTO comments (comment, blogId, accountId, date) VALUES ('$content', '$id', '$author', '$date')";  if ($db->query($sql) == true) {  echo "Comment added";  } else {  echo "Error: " . $sql . "<br>" . $db->error;  }  }  function deleteComment($id) {  global $db;  if ($\_SESSION["loggedIn"] == true) {  if ($\_SESSION["account"]->isAdmin()) {  $sql = "DELETE FROM comments WHERE commentId = '".$id."'";  if ($db->query($sql) == true) {  return true;  } else {  return false;  }  } else {  $accountId = $\_SESSION["account"]->id;  $sql = "SELECT commentId FROM comments WHERE commentId = '".$id."' AND accountId = '".$accountId."'";  $result = $db->query($sql);  if ($result->num\_rows === 1) {  $sql = "DELETE FROM comments WHERE commentId = '".$id."' AND accountId = '".$accountId."'";  if ($db->query($sql) == true) {  return true;  } else {  return false;  }  } else {  return false;  }  }  } else {  return false;  }  }  function deletePost($id) {  global $db;  if ($\_SESSION["loggedIn"] == true) {  if ($\_SESSION["account"]->isAdmin()) {  $sql = "DELETE FROM blog WHERE blogId = '".$id."'";  if ($db->query($sql) == true) {  $sql = "DELETE FROM comments WHERE blogId = '".$id."'";  if ($db->query($sql) == true) {  return true;  } else {  return false;  }  } else {  return false;  }  } else {  return false;  }  } else {  return false;  }  }  ?>  Filename: deleteBlog.php  <?php include("database.php");  if ($\_SESSION["loggedIn"] == true) {  $postId = $\_GET['postId'];  $accountId = $\_SESSION["account"]->id;  if (isset($\_GET['commentId'])) {  if (deleteComment($\_GET['commentId'])) {  $commentId = $\_GET['commentId'];  header("Location: ../../viewBlog.php?id=$postId");  } else {  header("Location: ../../viewBlog.php?id=$postId&error=You are not the author of this comment");  }  } elseif (isset($postId)) {  if (deletePost($postId)) {  header("Location: ../../viewBlog.php");  } else {  header("Location: ../../viewBlog.php?error=You are not the author of this post");  }  }  exit();  } else {  header("Location: ../../login.php?error=Required to login first");  exit();  }  ?>  Filename: viewBlog.php  <?php  include 'scripts/php/database.php';  ?>  <!DOCTYPE html>  <html lang="en">  <?php include 'scripts/php/head.php'; ?>  <body>  <div class="container">  <?php  include 'scripts/php/header.php';  ?>  <section id="blog-error">  <?php  if (isset($\_GET['error'])) {  echo "<p>Error: ".$\_GET['error']."</p>";  }  ?>  </section>  <?php  if (isset($\_GET['id'])) {  $post = getPost($\_GET['id']);  if ($post != null) {  echo "<article id='blog\_full'>";  echo "<nav id='blog-nav'>";  echo "<a href='viewBlog.php#".$\_GET['id']."'><i class='fa fa-arrow-left'></i><p>Back</p></a>";  echo "</nav>";  echo "<time datetime='".$post->date."'>".$post->date."</time>";  echo "<h3>".$post->title."</h3>";  echo "<p class='content'>".$post->content."</p>";  echo "<p class='author'>Author: ".$post->authorFullname."</p>";  echo "<div id='comments'>";  echo "<form action='scripts/php/addComment.php' method='get'>";  echo "<textarea name='comment' placeholder='Enter comment here' required></textarea>";  echo "<input type='hidden' name='id' value='".$post->id."' required>";  echo "<input type='submit' value='Send'>";  echo "</form>";  echo "<ul id='comments\_list'>";  for ($i = 0; $i < count($post->comments); $i++) {  echo "<li class='comment'>";  echo "<p class='comment\_author'>".$post->comments[$i]->authorFullname."</p>";  echo "<p class='comment\_content'>".$post->comments[$i]->comment."</p>";  echo "<time datetime='".$post->comments[$i]->get\_date\_str()."'>".$post->comments[$i]->get\_date\_str()."</time>";  if (isset($\_SESSION['account']) && ($post->comments[$i]->accountId == $\_SESSION['account']->id || $\_SESSION['account']->isAdmin())) {  echo '<a class="post\_comment\_delete" href="scripts/php/deleteBlog.php?commentId='.$post->comments[$i]->id.'&postId='.$\_GET['id'].'"><i class="fa fa-trash"></i></a>';  }  echo "</li>";  }  echo "</ul>";  echo "</div>";  echo "</article>";  }  else {  echo "<p>Post not found</p>";  }  }  else {  $posts = orderByDateDesc(updatePosts());  if (isset($\_GET['title']) && isset($\_GET['content']) && isset($\_SESSION['account']) && $\_SESSION['account']->isAdmin()) {  ?>  <nav id="blog-nav">  <a href="addEntry.php?title=<?php echo $\_GET['title']; ?>&content=<?php echo $\_GET['content']; ?>"><i class='fa fa-arrow-left'></i><p>Back</p></a>  <p>Preview</p>  <a href="scripts/php/addPost.php?title=<?php echo $\_GET['title']; ?>&content=<?php echo $\_GET['content']; ?>"><p>Post</p><i class='fa fa-check'></i></a>  </nav>  <a id='addpost\_button' href='addEntry.php'>Add Post</a>  <section id='blog'>  <article id="preview">  <time>Preview</time>  <h3><?php echo $\_GET['title']; ?></h3>  <p class="content"><?php echo $\_GET['content']; ?></p>  <p class="author">Author: <?php echo $\_SESSION['account']->getFullname(); ?></p>  </article>  <?php } else { ?>  <a id='addpost\_button' href='addEntry.php'>Add Post</a>  <section id='blog'>  <?php }  for ($i = 0; $i < count($posts); $i++) {  echo "<article id=".$posts[$i]->id." >";  echo "<time datetime='".$posts[$i]->date."'>".$posts[$i]->get\_date\_str()."</time>";  echo "<h3>".$posts[$i]->title."</h3>";  echo "<p class='content'>".$posts[$i]->content."</p>";  if (isset($\_SESSION['account']) && ($posts[$i]->accountId == $\_SESSION['account']->id || $\_SESSION['account']->isAdmin())) {  echo '<a class="post\_comment\_delete" href="scripts/php/deleteBlog.php?postId='.$posts[$i]->id.'"><i class="fa fa-trash"></i></a>';  }  echo "<p class='author'>Author: ".$posts[$i]->authorFullname."</p>";  echo "<a class='comments\_button' href='viewBlog.php?id=".$posts[$i]->id."'><i class='fa fa-comments-o'></i><p>".count($posts[$i]->comments)."</p></a>";  echo "</article>";  }  if (count($posts) === 0) { ?>  <article id="no\_posts">  <p class="content">No posts found</p>  </article>  <?php }  echo "</section>";  }  include 'scripts/php/footer.php';  ?>  </div>  </body>  </html> | Upon loading the posts in viewBlog.php, it is checked if the user is an admin, if so the user is able to delete the posts, hence a button with the link to delete the post is displayed to the user. The way this work is that it redirects to deleteBlog with post id using a query string. The deleteBlog.php uses the deletePost function within the database.php to delete the post from the database by sending the post id as an argument. The function first checks that the user is an admin since only admins can delete posts. If the user is an admin it deletes the posts, else it returns an error on the viewBlog page which it redirects back to using a query string.    Figure 3: Add and delete comments by users and admins. |

|  |  |
| --- | --- |
| **Assessment Criteria 8 - Extra Feature 2**  (Note: delete any of the two features below)   * Add a “preview” button in addpost. When this button is clicked, the new entry is previewed, and you can then decide (via a set of navigational links) whether to upload the entry or go back to edit it. | |
| Instruction: use this column to paste the code from the files that have been used to develop this functionality.  **Paste the code directly from visual studio code, do not include a screenshot of the code as this document has a Turnitin requirement and this will lead to an automatic mark of zero. See example below.** | Instruction: use this column to address the following in your own words:   1. How did you develop this feature? Describe the steps involved. 2. Describe the inner workings of the approach you developed.   Did you use any Generative AI tools to achieve this? If yes, can you provide the prompts you wrote to achieve this?  Give reference to the code you have written when answering the questions above.  You can use diagrams (e.g. flow charts) or screenshots of your webpage output to illustrate your answer. |
| Filename: test.php  <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta http-equiv="X-UA-Compatible" content="IE=edge">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <title>Test</title>  </head>  <body>  <h1>My first PHP page</h1>  <?php  echo "Hello World!";  ?>  </body>  </html>  Filename: test1.php  <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta http-equiv="X-UA-Compatible" content="IE=edge">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <title>Test</title>  </head>  <body>  <h1>My first PHP page</h1>  <?php  echo "Hello World!";  ?>  </body>  </html> | I did not use any Generative AI tools to achieve this.  I created a preview button and styled it using html and css. Using JavaScript, upon a click event of the button preview (when the button is pressed) the script checks that the title and content is entered and not an empty string. If not, it alerts the site user to fill in all fields. If it is then it proceeds in redirecting the current page to viewBlog.php with the title and content variables using a query string. The viewBlog.php always checks upon loading if there is such case where title and content variables are given. If so it displays a preview of the comment along with a navigation section on top with options to go back to editing or posting. To go back to editing, it adds the same variables it GETs from the query string, within the redirect url to addEntry.php.    Figure 4: A image of the preview in use. |

|  |
| --- |
| **Assessment Criteria 14 and 15 - Identification of semantic HTML5 elements and importance of semantic markup**  What semantic HTML5 elements have been used within your portfolio site?  Why is semantic markup important?  You will need to provide examples related to your project to illustrate your answer. (max. 250 words) |
| Semantic markup is important for the content of your html rather than the visual structure/ style. It defines the meaning of the content placed within the tags. It also makes the markup language much more readable by humans, as it should be.  One semantic element I used was “header”. It is used to demonstrate the top of the page, like a banner, such as the title and navigation. Navigation, included within header, is also has a semantic element “nav”. It holds all the links to other pages to navigate through the portfolio.  Another semantic element used is the “footer”. Like the header, footer element which represents the bottom of the page content. This generally is to do with odd information such as, contacting, the copyright license and more.  Another semantic element used is “section”, which is used to group together related content. These would typically include a heading. An example is when I was creating my index.php, my home page, I have a section with the heading education and a section with the heading license and certifications, both group the content related to the headings.  Another semantic element used is “article”, which is used to contain standalone pieces of content. An example is within the viewBlog.php, article is used for each blog post.  Another semantic element used is “aside”, which contains content somewhat related to the block it is contained within, however it is either not as important or does not constitute to the main content. |

|  |
| --- |
| **Assessment Criteria 16 - Content organisation**  Explain your portfolio site's structure and navigation.  Justify the arrangement of the content and how it enhances user experience.  You should also include a site navigation diagram. |
| This is a site map (Top-Down structure), which consists of all the pages within my portfolio. The website starts from Home Page. All level 1 pages are accessible by all using navigation bar. You can sign up from the login page and add posts from the blog page.  Home Page: Contains main information, such as about me, experiences, licenses and certifications, and my current post (currently student). This is a good introduction to a user to my site to tell them all about me and my career goals.  Education: Contains all institutions I visited and gained qualifications from, stating which qualifications, what I have personally learnt and why, grades achieved, skills that I have gained and start/ end year of course. This is a page intended for users more interested about me, by clicking on the page in navigation.  Projects: Contains any projects and dates of when I have been working on it since and till. Presents my skills and abilities further along with interests within my work of field.  Blog: A blog page with posts to my users about my recent activities. This is at the end of the navigation on left most to make it stand out more. The add post page is placed on the page since it correlates for its function; it has no purpose anywhere else.  Login: Allows user to login to add comments. It also has a link to access signup page since it correlates for its function; it has no purpose anywhere else. Login button is placed in the corner right most of the navigation bar as traditionally seen on other websites online to make it visible by the user. |

|  |
| --- |
| **Assessment Criteria 17 and 18 – Design concept and rationale**  Describe the website design, including the theme and colour scheme.  What CSS style rules did you write to achieve this?  Did you use any Generative AI tools to achieve this? If yes, can you provide the prompts you wrote to achieve this?  Finally, provide a rationale behind the chosen design and how it reflects your personal brand. |
| No AI tools were used to produce the design for the portfolio.  The website has a clean design with the background primarily being back along with text being beige or white. The website uses a coverall flexbox structure for all pages with a gap of 1rem between header, content and footer.  **FORM**  The form pages (login.php, addEntry.php) uses a white background to give a back shadow of the form. It also has interactive labels and input boxes to show where the user is selected and when there is an input within a box.  The CSS styles to display a back shadow of the form was box-shadow declared for the form elements.  For imperativeness, a transition declaration was used for the input boxes to for smooth transitions like the “border-bottom” declaration when the focus is upon the input box. Each button/ input type submit/ reset also use transition declaration to have a slow transition between the background changing upon hover. |

|  |
| --- |
| **Assessment Criteria 19 - Web technologies/framework**  For this project, you have made use of HTML, CSS, JavaScript and PHP. If you were to conduct this project again, which other web technologies/frameworks would you use to develop your portfolio site? Justify your choice. (max. 250 words) |
|  |