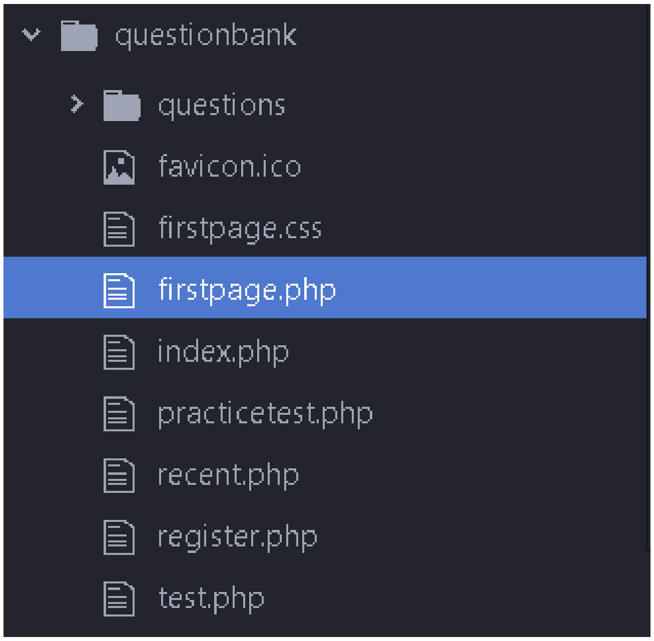
**Word count: 950**

**Questionbank**
**Criterion C: Development**
**Introduction:**

The questionbank will focus on allowing search and upload of questions, along with creation of precisely timed practice tests based on user input. The primary languages being implemented will be html for the framework, PHP for scripting and database connections, and Structured Query Language (SQL) for database management.

This solution allows client X to navigate and upload filtered questions based on topic (unit) and section (A or B), along with allowing creation of practice tests for their students.

**Structure of the program:**



```
questionbank
    questions
    favicon.ico
    firstpage.css
    firstpage.php
    index.php
    practicetest.php
    recent.php
    register.php
    test.php
```

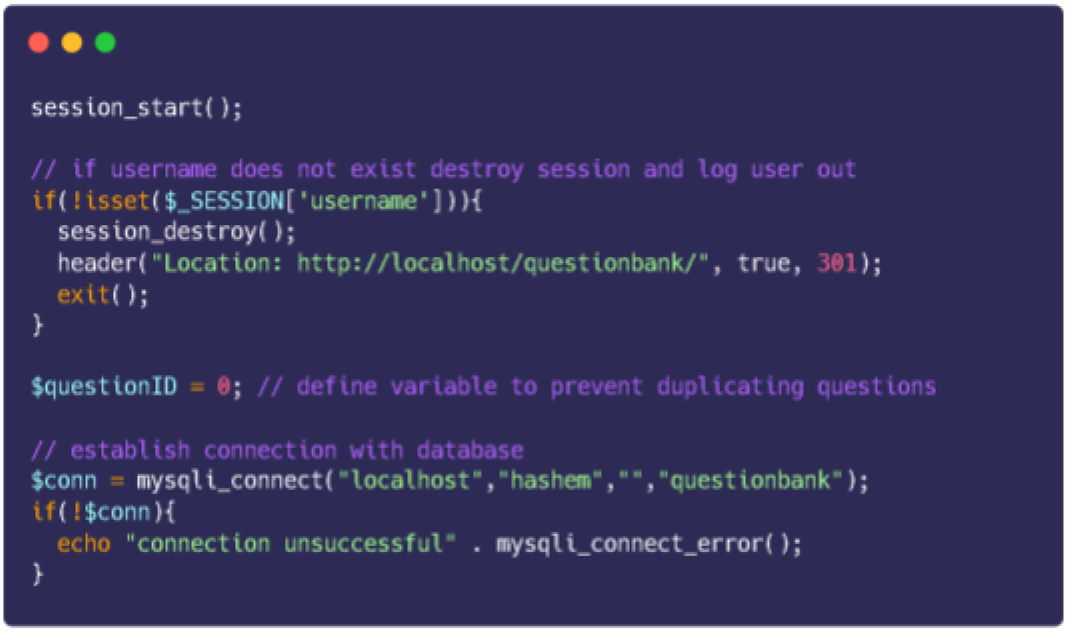
*Figure 1: Folder Structure of questionbank*

As seen in Figure 1, the *questionbank* folder contains all files of the project. The *questions* folder contains the question and markscheme images saved in the database, allowing them to be displayed in the web application. Other *.php* files are the web pages the user has access to.

**Summary list of all techniques:**

- **Parameter Passing**
- **Random index selection from array**
- **Foreach / While loop**
- **Conditionals and nested conditionals**
- **Methods**
- **String arrays**
- **2D arrays**
- **Stacks**
- **Searching**
- **Error Handling**
- **Structured Query Language**
- **Database connection**
- **Reading from/to database and database updating**
- **Selection Sort**
- **Array to string and string to array conversion**
- **Session management**

1. **Database Connection, Session Management, Error Handling**

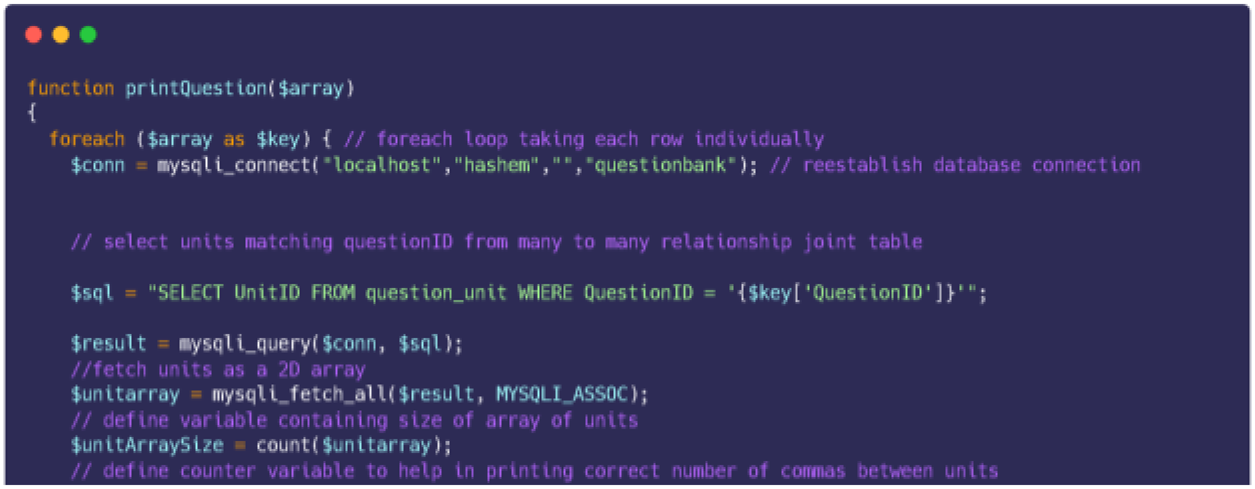


```
session_start();

// if username does not exist destroy session and log user out
if(!isset($_SESSION['username'])){
  session_destroy();
  header("Location: http://localhost/questionbank/", true, 301);
  exit();
}

$questionID = 0; // define variable to prevent duplicating questions

// establish connection with database
$conn = mysqli_connect("localhost","hashem","","questionbank");
if(!$conn){
  echo "connection unsuccessful" . mysqli_connect_error();
}
```

*Figure 1: Example of Database Connection, Session management, conditionals and error handling*

The code in *Figure 1* establishes a session, then has a conditional to check if the user’s username is defined from the login page. If not, the user’s session will destroy and force the user to the login page. A connection with the database is then established. And a conditional ensures that if the connection fails, an error message is displayed which is an example of error handling.

2. **Method, Parameter passing, Structured Query Language, 2D array**

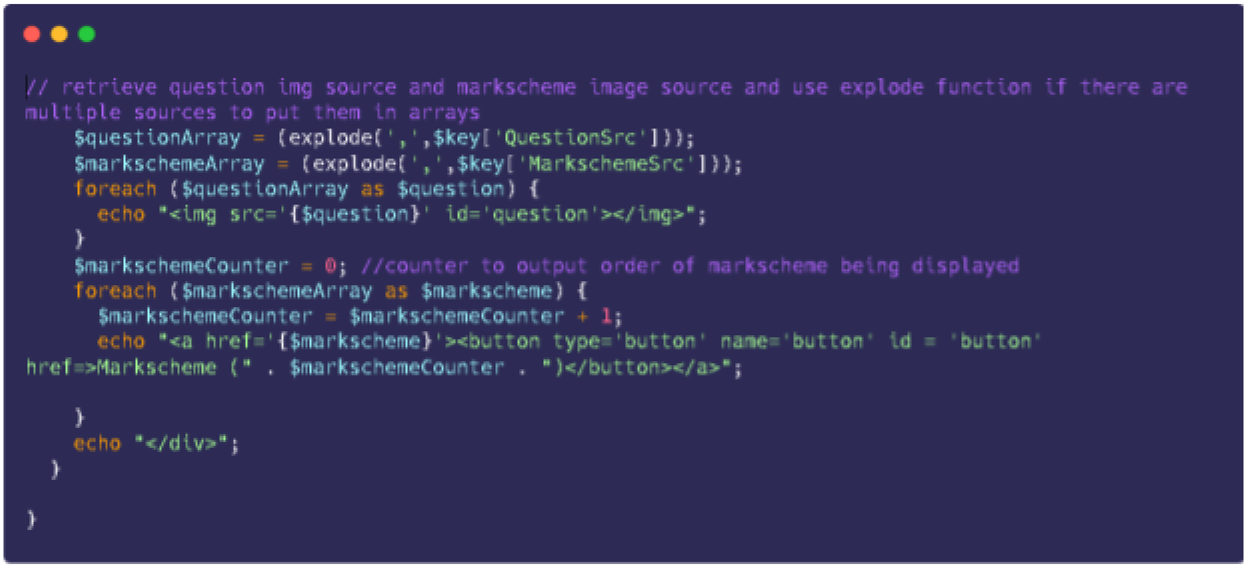


```
function printQuestion($array)
{
  foreach ($array as $key) { // foreach loop taking each row individually
    $conn = mysqli_connect("localhost","hashem","","questionbank"); // reestablish database connection


    // select units matching questionID from many to many relationship joint table

    $sql = "SELECT UnitID FROM question_unit WHERE QuestionID = '{$key['QuestionID']}'";

    $result = mysqli_query($conn, $sql);
    //fetch units as a 2D array
    $unitarray = mysqli_fetch_all($result, MYSQLI_ASSOC);
    // define variable containing size of array of units
    $unitArraySize = count($unitarray);
    // define counter variable to help in printing correct number of commas between units
```

*Figure 2: Code Segment for part one of printQuestion($array) method, containing foreach loop and SQL*

*Figure 2* shows the beginning of method *printQuestion($array)* which accepts a 2-dimensional array as a parameter. The function's purpose is to take every row of the 2D array input as a single question and output that question with all its details, in relation to Success Criterion number 9 where all details of the question must be displayed. A 2D array is used as the data structure for printing questions; because each question has attributes such as its unit, image source, etc. which each require an individual column and this why it was the best data structure to be used for this case.

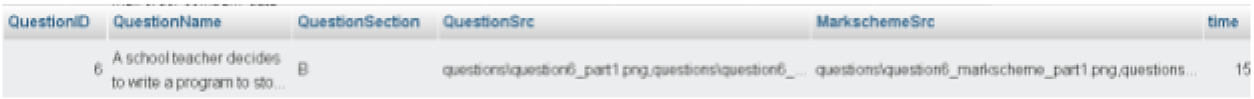
### 3. String to Array Conversion



```
// retrieve question img source and markscheme image source and use explode function if there are
multiple sources to put them in arrays
    $questionArray = (explode(',',$key['QuestionSrc']));
    $markschemeArray = (explode(',',$key['MarkschemeSrc']));
    foreach ($questionArray as $question) {
      echo "<img src='{$question}' id='question'></img>";
    }
    $markschemeCounter = 0; //counter to output order of markscheme being displayed
    foreach ($markschemeArray as $markscheme) {
      $markschemeCounter = $markschemeCounter + 1;
      echo "<a href='{$markscheme}'><button type='button' name='button' id = 'button'
href=>Markscheme (" . $markschemeCounter . ")</button></a>";

    }
    echo "</div>";
  }

}
```

*Figure 3: String to Array conversion*

In cases where a question or mark scheme might require multiple images to be displayed, multiple image sources are saved in the database in a single cell, separated by commas. *Figure 4* exemplifies this:



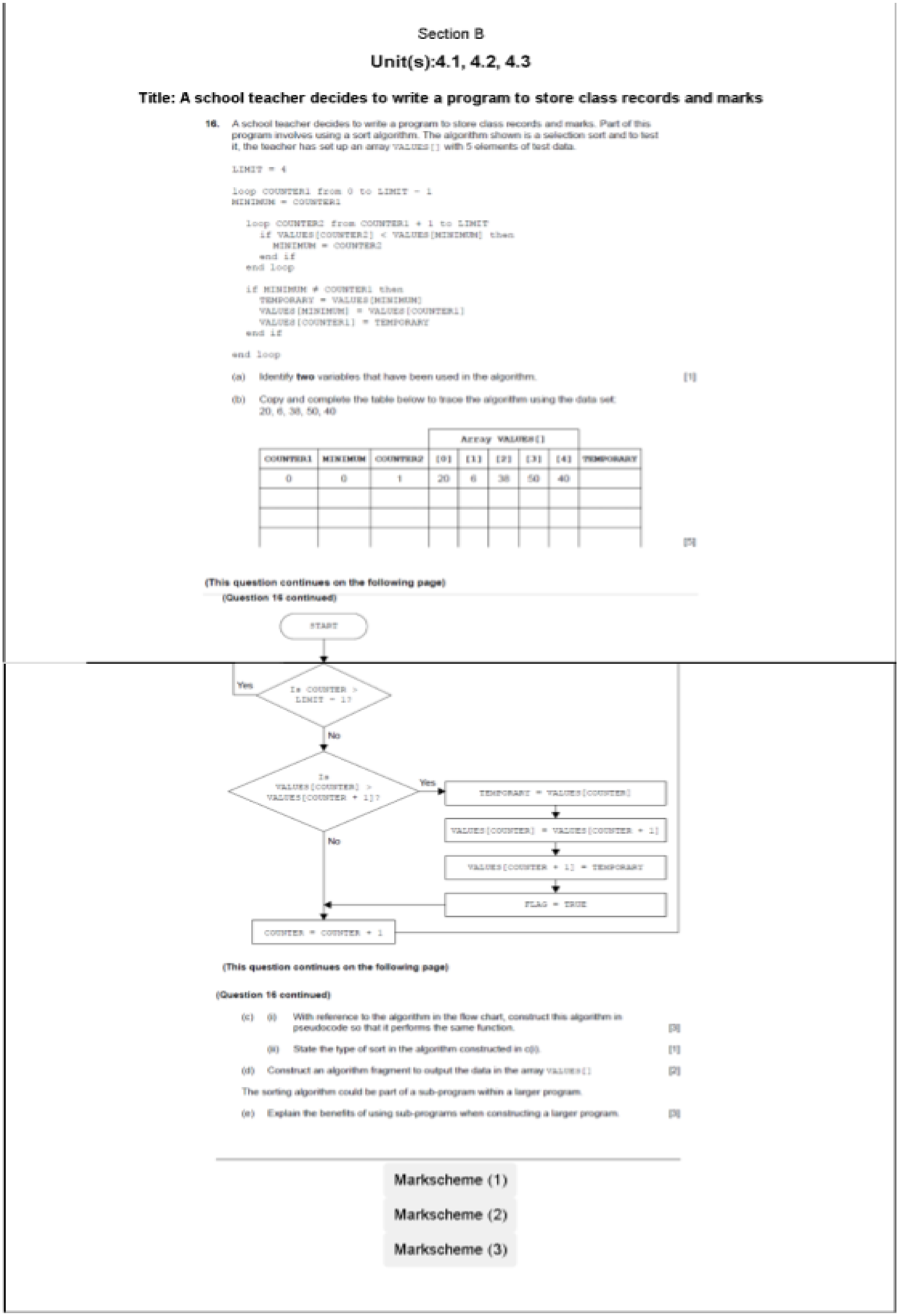
| QuestionID | QuestionName | QuestionSection | QuestionSrc | MarkschemeSrc | time |
|---|---|---|---|---|---|
| 6 | A school teacher decides to write a program to sto... | B | questions\question6_part1.png,questions\question6_... | questions\question6_markscheme_part1.png,questions... | 15 |

questions\question6_part1.png,questions\question6_part2.png,questions\question6_part3.png

questions\question6_markscheme_part1.png,questions\question6_markscheme_part2.pngquestions\question6_markscheme_part3.png

*Figure 4: example record from question table in database*

The explode function in *Figure 3* converts the string in the record into an array, using a foreach loop to output every image in numerical order, using a counter to clarify the order of each markscheme.



Section B

**Unit(s):4.1, 4.2, 4.3**

**Title: A school teacher decides to write a program to store class records and marks**

**16.** A school teacher decides to write a program to store class records and marks. Part of this program involves using a sort algorithm. The algorithm shown is a selection sort and to test it, the teacher has set up an array `VALUES[]` with 5 elements of test data.

```
LIMIT = 4

loop COUNTER1 from 0 to LIMIT - 1
MINIMUM = COUNTER1

  loop COUNTER2 from COUNTER1 + 1 to LIMIT
    if VALUES[COUNTER2] < VALUES[MINIMUM] then
      MINIMUM = COUNTER2
    end if
  end loop

  if MINIMUM ≠ COUNTER1 then
    TEMPORARY = VALUES[MINIMUM]
    VALUES[MINIMUM] = VALUES[COUNTER1]
    VALUES[COUNTER1] = TEMPORARY
  end if

end loop
```

(a) Identify **two** variables that have been used in the algorithm. [1]

(b) Copy and complete the table below to trace the algorithm using the data set: 20, 6, 38, 50, 40

|  |  |  | Array VALUES[] |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|
| COUNTER1 | MINIMUM | COUNTER2 | [0] | [1] | [2] | [3] | [4] | TEMPORARY |
| 0 | 0 | 1 | 20 | 6 | 38 | 50 | 40 |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

[5]

**(This question continues on the following page)**

**(Question 16 continued)**

START
Yes
Is COUNTER > LIMIT - 1?
No
Is VALUES[COUNTER] > VALUES[COUNTER + 1]?
Yes
TEMPORARY = VALUES[COUNTER]
VALUES[COUNTER] = VALUES[COUNTER + 1]
VALUES[COUNTER + 1] = TEMPORARY
No
FLAG = TRUE
COUNTER = COUNTER + 1

**(This question continues on the following page)**

**(Question 16 continued)**

(c) (i) With reference to the algorithm in the flow chart, construct this algorithm in pseudocode so that it performs the same function. [3]

(ii) State the type of sort in the algorithm constructed in c(i). [1]

(d) Construct an algorithm fragment to output the data in the array `VALUES[]` [2]

The sorting algorithm could be part of a sub-program within a larger program.

(e) Explain the benefits of using sub-programs when constructing a larger program. [3]

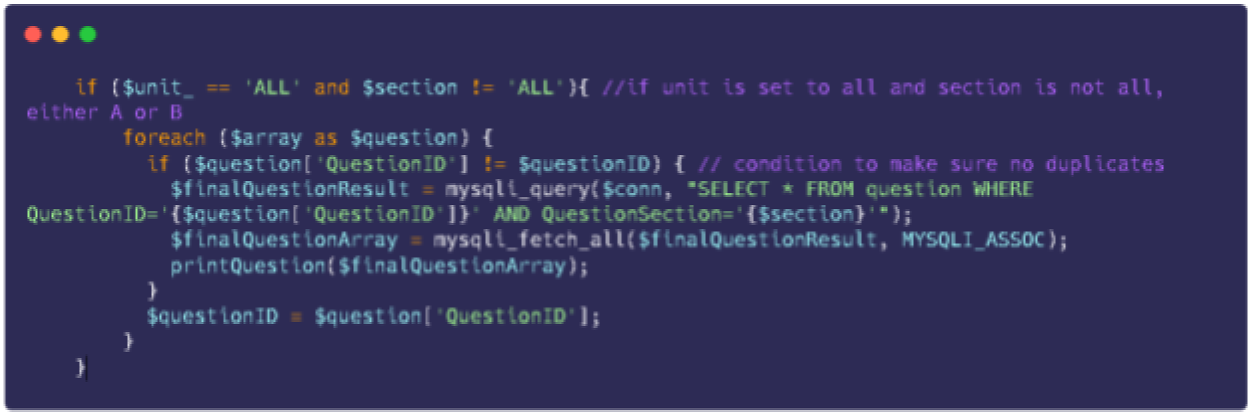
Markscheme (1)

Markscheme (2)

Markscheme (3)

*Figure 5: example question with multiple images from application*

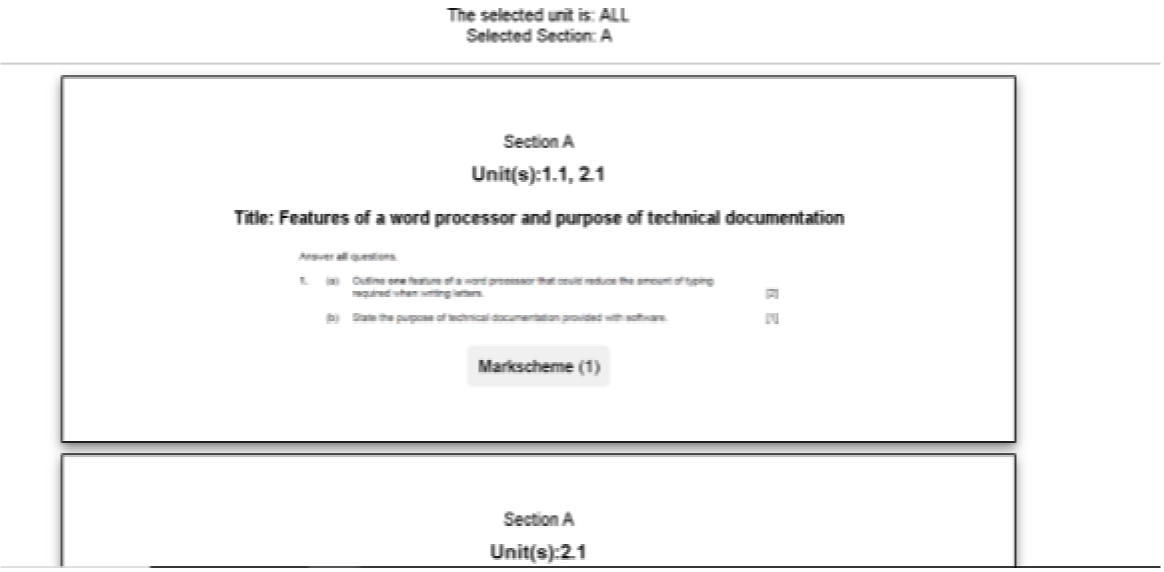
## 4. Methods, Database retrieval, Filtering using nested conditionals, Duplication prevention, and SQL



```
    if ($unit_ == 'ALL' and $section != 'ALL'){ //if unit is set to all and section is not all,
either A or B
        foreach ($array as $question) {
          if ($question['QuestionID'] != $questionID) { // condition to make sure no duplicates
            $finalQuestionResult = mysqli_query($conn, "SELECT * FROM question WHERE
QuestionID='{$question['QuestionID']}' AND QuestionSection='{$section}'");
            $finalQuestionArray = mysqli_fetch_all($finalQuestionResult, MYSQLI_ASSOC);
            printQuestion($finalQuestionArray);
          }
          $questionID = $question['QuestionID'];
        }
    }
```

*Figure 6: Filtering and search code segment*

*Figure 6* contains a code segment responsible for providing questions after the user filtered for a specific section, but didn't choose a topic. A nested conditional is necessary in this case to prevent duplicated questions, where the code checks if the printed question has the same *questionID* as the question printed before it. In order to filter questions, a *WHERE* clause is implemented in the SQL statement, only selecting questions from the database that match the *$section* variable that saved the user input. This information retrieved from the database is saved in a 2D array that is printed using the *printQuestion($array)* method. These techniques help fulfill success criteria number 8.



The selected unit is: ALL
Selected Section: A

Section A

Unit(s):1.1, 2.1

Title: Features of a word processor and purpose of technical documentation

Answer all questions.

1. (a) Outline one feature of a word processor that could reduce the amount of typing required when writing letters. [2]

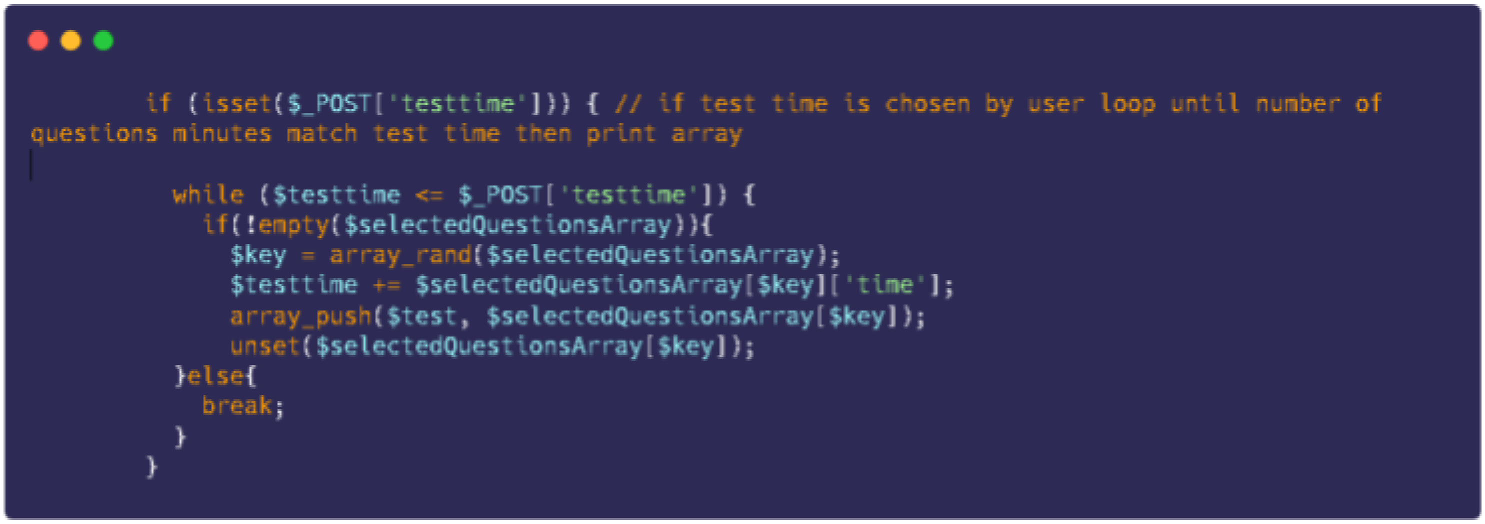
(b) State the purpose of technical documentation provided with software. [1]

Section A

Unit(s):2.1

*Figure 7: example of selected section A filter for questions*

### 5. While loop and random selection in Stack data structure

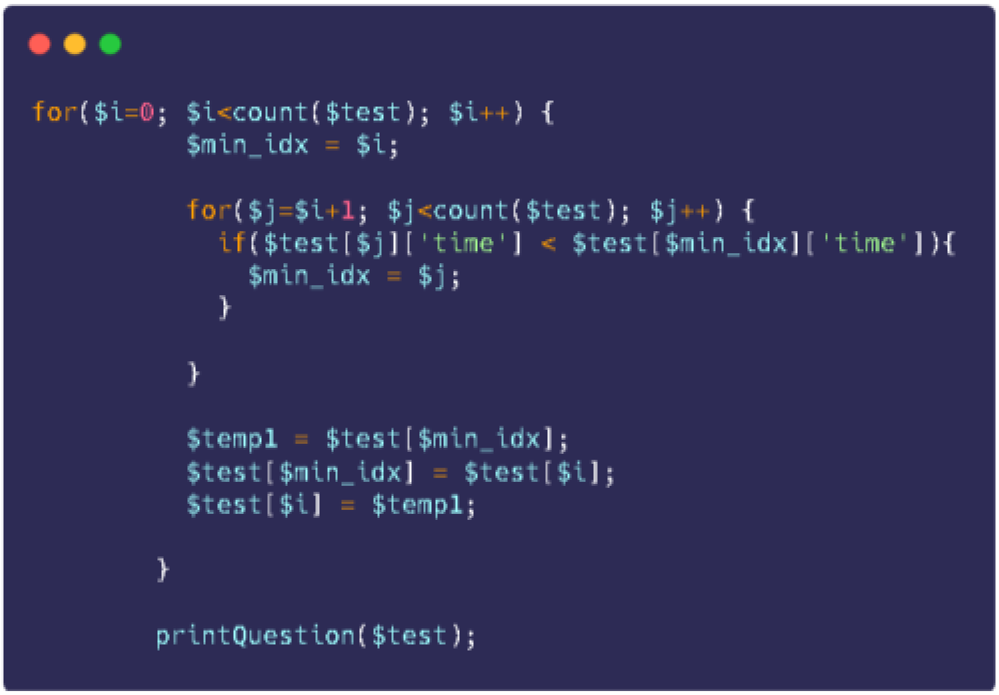


```
        if (isset($_POST['testtime'])) { // if test time is chosen by user loop until number of
questions minutes match test time then print array

          while ($testtime <= $_POST['testtime']) {
            if(!empty($selectedQuestionsArray)){
              $key = array_rand($selectedQuestionsArray);
              $testtime += $selectedQuestionsArray[$key]['time'];
              array_push($test, $selectedQuestionsArray[$key]);
              unset($selectedQuestionsArray[$key]);
          }else{
            break;
          }
        }
```

*Figure 8: While loop, random number generation in Stack*

When creating a practice test, the number of questions printed varies based on the time chosen, this makes the Stack abstract data structure suitable for practice tests. The stack is a dynamic data structure with no fixed size, and the code in *Figure 8* takes random selections from an array of questions based on user selection. Afterwards, it uses the stack *_push* function to add these questions to the stack. This is contained in a *while* loop that ensures that the total time of the questions does not exceed the time specified by the user.

### 6. Selection sort on 2D array



```
for($i=0; $i<count($test); $i++) {
          $min_idx = $i;

          for($j=$i+1; $j<count($test); $j++) {
            if($test[$j]['time'] < $test[$min_idx]['time']){
              $min_idx = $j;
            }

          }

          $temp1 = $test[$min_idx];
          $test[$min_idx] = $test[$i];
          $test[$i] = $temp1;

        }

        printQuestion($test);
```

*Figure 9: Selection Sort performed on practice test*

In order to make sure that the test questions are in order of lowest to greatest time needed, according to success criterion 11, a selection sort method will be applied on the 2D array containing practice test questions before printing them using the *printQuestion($array)* method. The code in *Figure 9* applies this algorithm by rearranging indexes of the 2D array rows based on the value of their *time* attribute.

**Software tools used:**

- **Atom IDE:** I used the Atom IDE as it has a friendly layout that is not over complicated. It also is free of cost and allows developers to change the theme easily. In addition to this, navigation of project folders is clear and not time consuming, allowing me to focus on writing code while knowing exactly which folder I am in.
- **XAMPP:** I used XAMPP as a web server stack solution package as it is not only free of cost, but also allows local apache hosting of a web server, along with hosting of my database management system which allows me to access my database through phpMyAdmin.
- **phpMyAdmin:** This easy-to-use interface allows me to manage databases, tables and records with ease by typing SQL or by using the GUI, it also has a built in designer tool that allows me to properly understand and explain all the components of the database management system, and demonstrate the relationships between tables in the relational model.

## User interface:

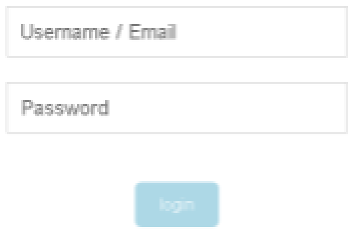


*Figure 10: Login Page*

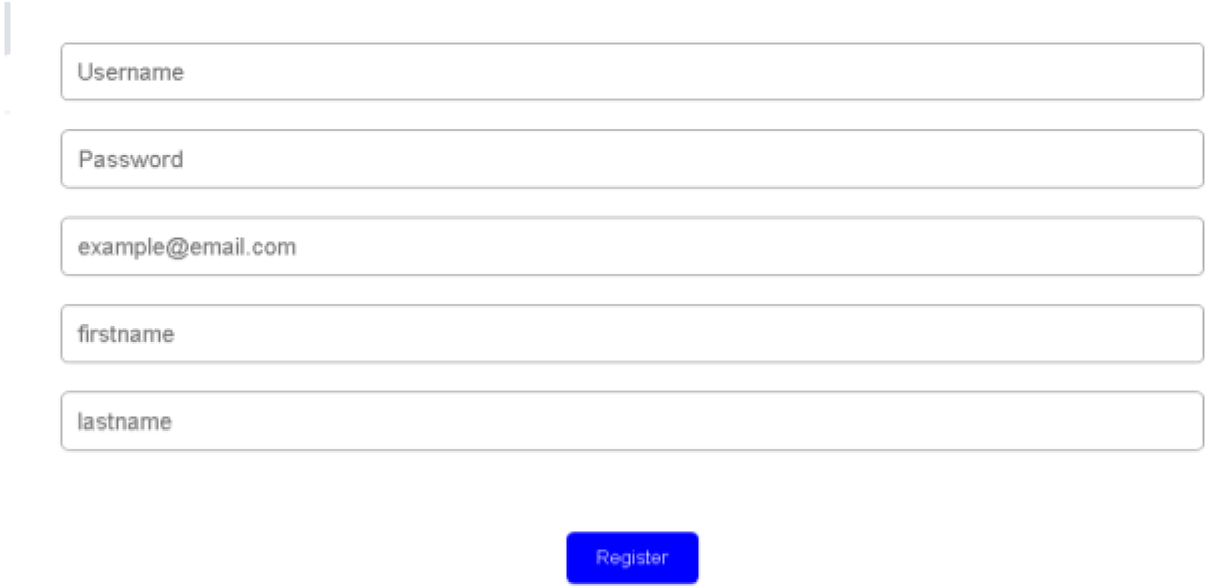
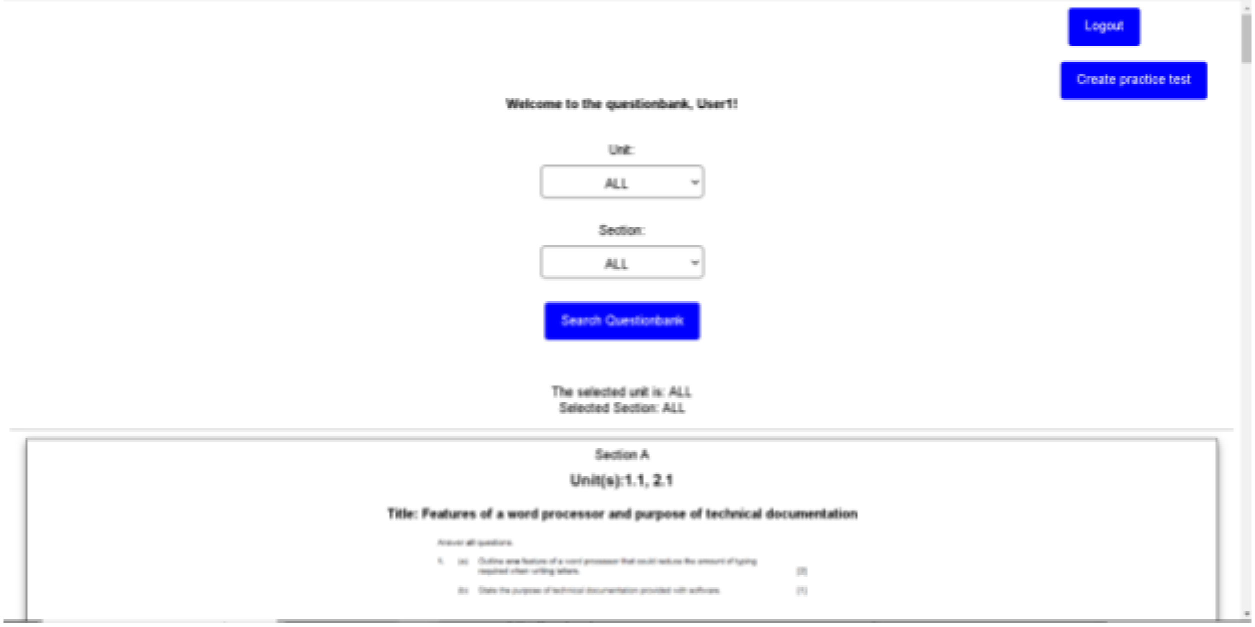


*Figure 11: Registration Page*



**Welcome to the questionbank, User1!**

The selected unit is: ALL
Selected Section: ALL

Section A

**Unit(s):1.1, 2.1**

**Title: Features of a word processor and purpose of technical documentation**

Answer all questions.

1. (a) Outline one feature of a word processor that could reduce the amount of typing required when writing letters. [2]

(b) State the purpose of technical documentation provided with software. [1]

*Figure 12: Questionbank main page*

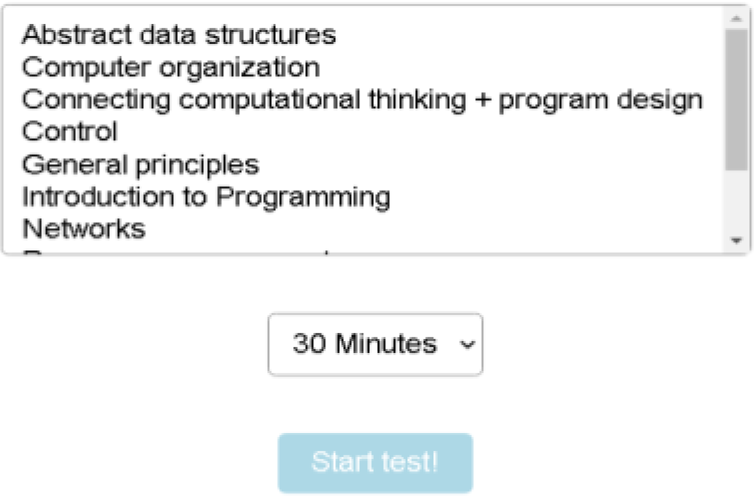
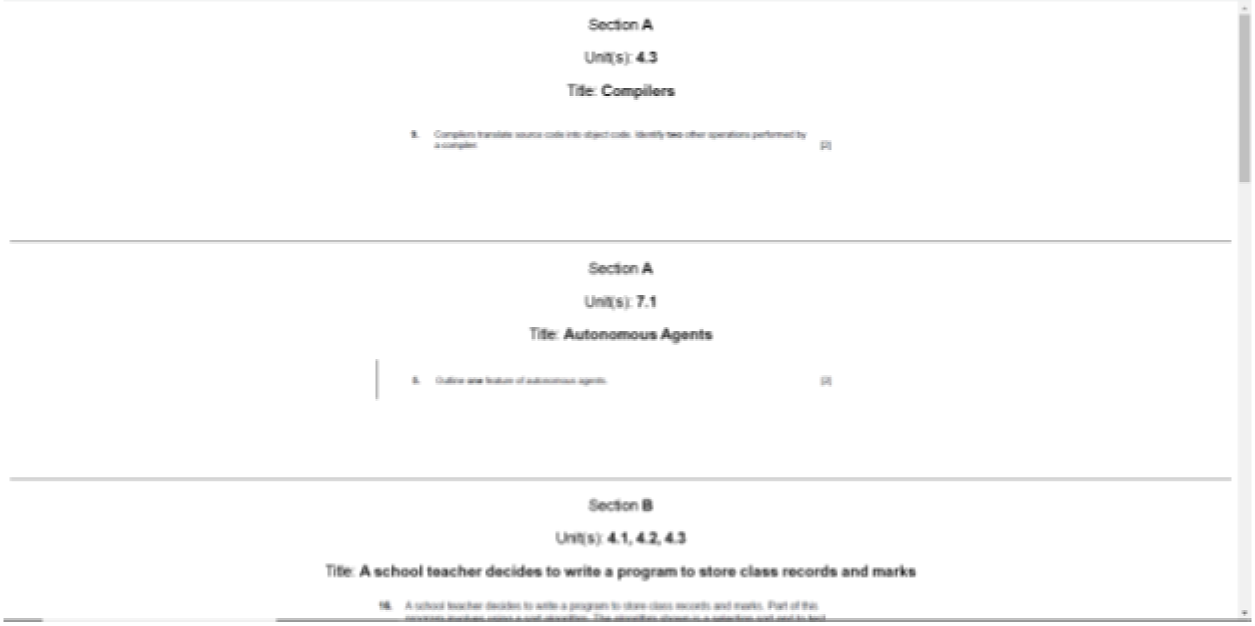


*Figure 13: Questionbank creation tab*



Section **A**

Unit(s): **4.3**

Title: **Compilers**

9. Compilers translate source code into object code. Identify **two** other operations performed by a compiler. [2]

Section **A**

Unit(s): **7.1**

Title: **Autonomous Agents**

5. Outline **one** feature of autonomous agents. [2]

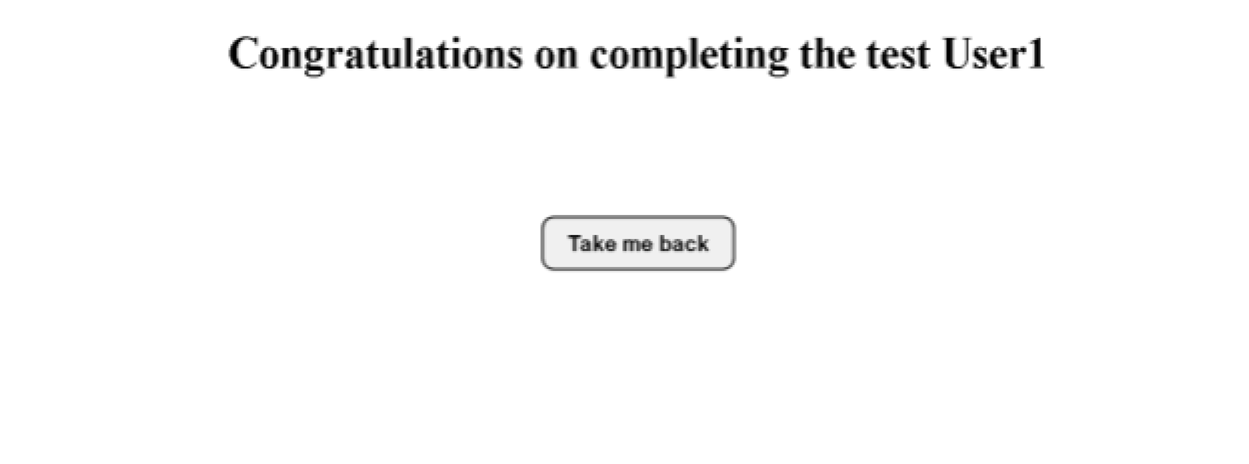
Section **B**

Unit(s): **4.1, 4.2, 4.3**

Title: **A school teacher decides to write a program to store class records and marks**

16. A school teacher decides to write a program to store class records and marks. Part of the program involves using a sort algorithm. The algorithm shown is a selection sort and to test

*Figure 14: Sample practice test*



Congratulations on completing the test User1

*Figure 15: Message after completing practice test and link to go back to home page*

# Sources:

- *PHP/25 Essential Functions - Wikiversity* (2023). Available at: https://en.wikiversity.org/wiki/PHP/25_Essential_Functions (Accessed: 5 February 2023).
- *PHP Program for Selection Sort - AlphaCodingSkills* (2023). Available at: https://www.alphacodingskills.com/php/pages/php-program-for-selection-sort.php (Accessed: 5 February 2023).

- *PHP Multidimensional Arrays* (2023). Available at: https://www.w3schools.com/php/php_arrays_multidimensional.asp (Accessed: 5 February 2023).
- *PHP: Sort a list of elements using Selection sort - w3resource* (2022). Available at: https://www.w3resource.com/php-exercises/searching-and-sorting-algorithm/searching-and-sorting-algorithm-exercise-4.php (Accessed: 5 February 2023).
- *PHP: Function and Method listing - Manual* (2023). Available at: https://www.php.net/manual/en/indexes.functions.php (Accessed: 5 February 2023).
- *PHP - GET & POST Methods* (2023). Available at: https://www.tutorialspoint.com/php/php_get_post.htm (Accessed: 5 February 2023).
- *What are magic methods and how to use them in PHP ? - GeeksforGeeks* (2021). Available at: https://www.geeksforgeeks.org/what-are-magic-methods-and-how-to-use-them-in-php/ (Accessed: 5 February 2023).