VISVESVARAYA TECHNOLOGICAL UNIVERSITY

WEB TECHNOLOGY LAB

BCSL504

EXPERIMENT – 08

PHP WEB APPLICATIONS

Visitor Counter & Student Record Sorting

5th Semester

Part A: Visitor Counter using PHP

Part B: Student Records from Database

Selection Sort • PHP • HTML • CSS

Prepared by Search Creators

Learn • Practice • Execute

1. Experiment 08

Aim

Develop the following applications using PHP with HTML and CSS:

a. A web application that keeps track of the number of visitors visiting the webpage.

b. A web application to sort student records stored in a database using the **Selection Sort** algorithm.

2. Requirements

- PHP-supported web server environment
- MySQL database for Part (b)
- Modern web browser
- HTML and CSS
- PHP
- **Search Creators Compiler / Code Studio** for coding practice:

  https://searchcreators.org/compiler/

- Alternatively, a local PHP development environment and suitable code editor

3. Important Execution Note

Important

PHP is a **server-side scripting language**.
Therefore, unlike a normal HTML file, PHP code requires a PHP-supported server/runtime for execution.
For Part (b), a MySQL database server is also required.
Use the Search Creators Compiler / Code Studio while practising supported web programs and concepts:

https://searchcreators.org/compiler/

For actual PHP/MySQL execution, use an environment that supports PHP and MySQL.

4. Introduction to PHP

PHP is a server-side scripting language used for developing dynamic web applications.

PHP code begins with:

```
<?php
```

and ends with:

```
?>
```

A simple PHP statement is:

```
<?php
echo "Hello World";
?>
```

The `echo` statement is used to display output.

5. Client-Server Working

6. Part (a): Visitor Counter

6.1 Aim

Develop a PHP web application that keeps track of the number of visitors visiting a webpage.

6.2 Concept

A simple visitor counter can store the current count in a text file.

Whenever the PHP page is requested:

1. Read the current count.
2. Increase the count by one.
3. Store the updated count.
4. Display the count on the webpage.

6.3 Algorithm

1. Start.
2. Specify the counter file.
3. Check whether the counter file exists.
4. If the file exists, read the existing visitor count.
5. Otherwise, initialize the count to zero.
6. Increment the count by one.
7. Write the updated count into the file.
8. Display the visitor count.
9. Stop.

6.4 Program – visitor.php

Create:

visitor.php

```php
<?php

$file = "counter.txt";

if (file_exists($file)) {

    $count = (int) file_get_contents($file);

} else {

    $count = 0;
}

$count++;

file_put_contents(
    $file,
    $count,
    LOCK_EX
);

?>

```

```
<!DOCTYPE html>
<html lang="en">

<head>

<meta charset="UTF-8">

<title>Visitor Counter</title>

<style>

body {
    font-family: Arial, sans-serif;
    background-color: #eaf4ef;
    text-align: center;
}

.container {
    width: 450px;
    margin: 100px auto;
    padding: 30px;
    background-color: white;
    border: 2px solid darkgreen;
    border-radius: 10px;
}

h2 {
    color: darkgreen;
    font-size: 30px;
}

.counter {
    background-color: lightyellow;
    color: darkblue;
    padding: 20px;
    margin-top: 20px;
    font-size: 24px;
    font-weight: bold;
}

</style>

</head>

<body>

<div class="container">

```

```
    <h2>Visitor Counter</h2>

    <p>
        Welcome to our webpage.
    </p>

    <div class="counter">

        Total Visitors:
        <?php echo $count; ?>

    </div>

</div>

</body>

</html>
```

6.5 Program Explanation

The counter file is specified using:

```
$file = "counter.txt";
```

The program checks whether the file exists:

```
file_exists($file)
```

The existing value is read using:

```
file_get_contents($file)
```

The count is incremented:

```
$count++;
```

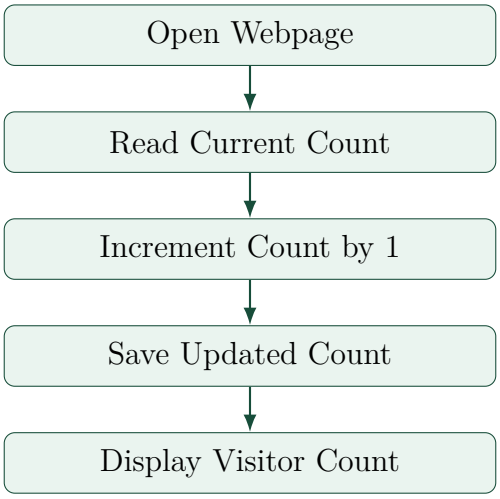
The updated value is stored using:

```
file_put_contents(
    $file,
    $count,
    LOCK_EX
);
```

Finally, PHP displays the visitor count:

```
<?php echo $count; ?>
```

6.6 Visitor Counter Flow



6.7 Expected Output – Part (a)

Expected Browser Output

Visitor Counter

Welcome to our webpage.

Total Visitors: 15

Important

This simple laboratory program increments the counter whenever the PHP page is requested.
Therefore, it demonstrates a basic page-visit counter; it does not identify unique human visitors.

7. Part (b): Sort Student Records Using Selection Sort

7.1 Aim

Develop a PHP web application to retrieve student records stored in a database and sort them using the **Selection Sort** algorithm.

7.2 What is Selection Sort?

Selection Sort is a comparison-based sorting algorithm.

For each position, it finds the smallest element from the remaining unsorted portion and places it in the correct position.

For an array of n elements, its typical time complexity is:

$$\boxed{O(n^2)}$$

7.3 Selection Sort Algorithm

For an array:

$$A[0], A[1], \ldots, A[n-1]$$

the algorithm is:

1. Set the current position as the minimum.
2. Compare it with all remaining elements.
3. Find the smallest element.
4. Swap the smallest element with the current element.
5. Repeat until the array is sorted.

7.4 Example

Suppose student marks are:

$$70, \quad 45, \quad 85, \quad 60$$

After sorting:

$$45, \quad 60, \quad 70, \quad 85$$

8. Database Creation

Create a database:

`studentdb`

SQL:

```
CREATE DATABASE studentdb;
```

Select the database:

```
USE studentdb;
```

9. Create Student Table

Create the table:

```
CREATE TABLE students (
    id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(50) NOT NULL,
    usn VARCHAR(20) NOT NULL UNIQUE,
    marks INT NOT NULL
);
```

10. Insert Sample Records

```
INSERT INTO students
(name, usn, marks)
VALUES
('Ravi', '1XX22CS001', 70),
('Asha', '1XX22CS002', 45),
('Kiran', '1XX22CS003', 85),
('Megha', '1XX22CS004', 60);
```

The database now contains:

ID	Name	USN	Marks
1	Ravi	1XX22CS001	70
2	Asha	1XX22CS002	45
3	Kiran	1XX22CS003	85
4	Megha	1XX22CS004	60

11. PHP Program – sortstudents.php

Create:

`sortstudents.php`

```
<?php

$host = "localhost";
$user = "root";
$password = "";
```

```php
$database = "studentdb";

$conn = new mysqli(
    $host,
    $user,
    $password,
    $database
);

if ($conn->connect_error) {

    die(
        "Connection failed: " .
        $conn->connect_error
    );
}

/*
   Read student records
   from the database.
*/
$sql =
    "SELECT id, name, usn, marks
     FROM students";

$result = $conn->query($sql);

$students = array();

if ($result) {

    while ($row = $result->fetch_assoc()) {

        $students[] = $row;
    }
}

/*
   Selection Sort
   Sort students by marks
   in ascending order.
*/
$n = count($students);

for ($i = 0; $i < $n - 1; $i++) {

    $minIndex = $i;

```

```
    for ($j = $i + 1; $j < $n; $j++) {

        if (
            (int)$students[$j]["marks"] <
            (int)$students[$minIndex]["marks"]
        ) {

            $minIndex = $j;
        }
    }

    if ($minIndex != $i) {

        $temp = $students[$i];

        $students[$i] =
            $students[$minIndex];

        $students[$minIndex] =
            $temp;
    }
}

?>

<!DOCTYPE html>
<html lang="en">

<head>

<meta charset="UTF-8">

<title>Student Records</title>

<style>

body {
    font-family: Arial, sans-serif;
    background-color: #f2f2f2;
}

.container {
    width: 80%;
    margin: 40px auto;
    background-color: white;
    padding: 25px;
    border: 2px solid darkgreen;
}
```

```
102
103 h2 {
104     color: darkgreen;
105     text-align: center;
106 }
107
108 table {
109     width: 100%;
110     border-collapse: collapse;
111 }
112
113 th {
114     background-color: darkgreen;
115     color: white;
116     padding: 12px;
117 }
118
119 td {
120     border: 1px solid #333;
121     padding: 10px;
122     text-align: center;
123 }
124
125 tr:nth-child(even) {
126     background-color: #eaf4ef;
127 }
128
129 </style>
130
131 </head>
132
133 <body>
134
135 <div class="container">
136
137 <h2>
138 Student Records Sorted by Marks
139 </h2>
140
141 <table>
142
143 <tr>
144
145     <th>ID</th>
146     <th>Name</th>
147     <th>USN</th>
148     <th>Marks</th>
149
```

```
</tr>

<?php

foreach ($students as $student) {

    echo "<tr>";

    echo "<td>" .
        htmlspecialchars(
            (string)$student["id"]
        ) .
        "</td>";

    echo "<td>" .
        htmlspecialchars(
            $student["name"]
        ) .
        "</td>";

    echo "<td>" .
        htmlspecialchars(
            $student["usn"]
        ) .
        "</td>";

    echo "<td>" .
        htmlspecialchars(
            (string)$student["marks"]
        ) .
        "</td>";

    echo "</tr>";
}

?>

</table>

</div>

</body>

</html>

<?php

$conn->close();
```

```
198
199 ?>
```

12. Program Explanation – Part (b)

12.1 Database Connection

The database connection is created using:

```
$conn = new mysqli(
    $host,
    $user,
    $password,
    $database
);
```

The program checks the connection using:

```
if ($conn->connect_error)
```

12.2 Retrieve Student Records

The records are retrieved using:

```
SELECT id, name, usn, marks
FROM students
```

Each database row is stored in the PHP array:

```
$students[] = $row;
```

12.3 Selection Sort

The number of student records is:

```
$n = count($students);
```

For every position:

```
$minIndex = $i;
```

The program searches for the record having the smallest marks value:

```
if (
    (int)$students[$j]["marks"] <
    (int)$students[$minIndex]["marks"]
)
```

When a smaller value is found:

```
$minIndex = $j;
```

Finally, the records are swapped.

13. Selection Sort Working

For marks:

$$70, \quad 45, \quad 85, \quad 60$$

Initial:

$$70, \quad 45, \quad 85, \quad 60$$

Smallest value is 45.

After first pass:

$$45, \quad 70, \quad 85, \quad 60$$

From the remaining values, the smallest is 60.

After second pass:

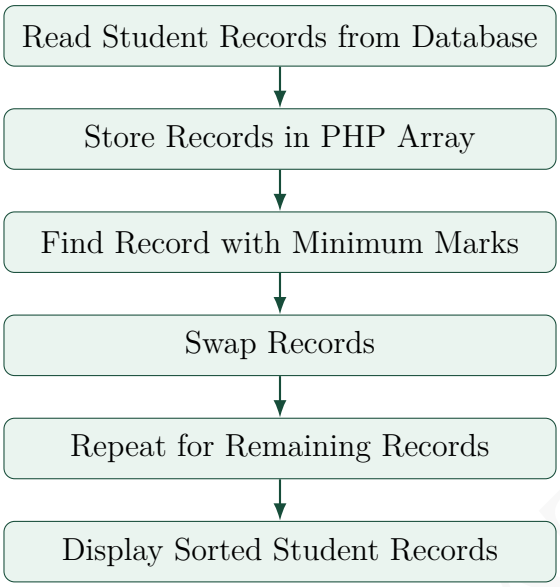
$$45, \quad 60, \quad 85, \quad 70$$

From the remaining values, the smallest is 70.

Final result:

$$\boxed{45, \quad 60, \quad 70, \quad 85}$$

14. Selection Sort Flow



15. Expected Output – Part (b)

After applying Selection Sort by marks in ascending order:

ID	Name	USN	Marks
2	Asha	1XX22CS002	45
4	Megha	1XX22CS004	60
1	Ravi	1XX22CS001	70
3	Kiran	1XX22CS003	85

16. How to Execute Experiment 08

Practice with Search Creators

For Search Creators coding practice, visit:

Search Creators Compiler / Code Studio

https://searchcreators.org/compiler/

For the actual PHP programs in this experiment, execute them in a **PHP-supported server environment**.
For Part (b), ensure that the MySQL database is running and the required database/table has been created.

17. Procedure – Part (a)

1. Create a PHP file named `visitor.php`.
2. Enter the visitor counter program.
3. Use `counter.txt` to store the count.
4. Place the PHP file in a PHP-supported web server environment.
5. Make sure the PHP process has permission to create or update `counter.txt`.
6. Start the web server.
7. Open `visitor.php` through the server URL.
8. Observe the displayed visitor count.
9. Reload the page.
10. Verify that the count increases.

18. Procedure – Part (b)

1. Start the PHP web server and MySQL database server.
2. Create the database `studentdb`.
3. Create the `students` table.
4. Insert sample student records.
5. Create the file `sortstudents.php`.
6. Configure the database connection values according to the local/server environment.
7. Connect PHP to MySQL.
8. Retrieve student records.
9. Store the records in a PHP array.
10. Apply Selection Sort on the marks field.
11. Display the sorted records in an HTML table.
12. Open the PHP page through the server URL.
13. Verify that records appear in ascending order of marks.

19. Important PHP Functions and Concepts

Function / Concept	Purpose
`file_exists()`	Checks whether a file exists.
`file_get_contents()`	Reads file contents.
`file_put_contents()`	Writes data into a file.
`new mysqli()`	Creates a MySQL database connection.
`query()`	Executes an SQL query.
`fetch_assoc()`	Fetches a result row as an associative array.
`count()`	Returns the number of elements in an array.
`foreach`	Iterates through array elements.
`htmlspecialchars()`	Escapes special HTML characters before displaying database values.

20. Visitor Counter vs Student Sorting

Feature	Part (a)	Part (b)
Application	Visitor Counter	Student Record Sorting
Backend	PHP	PHP
Storage	Text file	MySQL database
Main Operation	Increment visitor count	Selection Sort
Frontend	HTML + CSS	HTML + CSS
Output	Visitor count	Sorted student table

21. Important Viva Questions

Important Viva Questions

1. **What is PHP?**

   PHP is a server-side scripting language used for developing dynamic web applications.

2. **Where is PHP executed?**

   PHP is executed on the server.

3. **What is the purpose of the visitor counter?**

   It keeps track of page visits in this experiment.

4. **Which file stores the visitor count in this program?**

   `counter.txt`

5. **What does file_exists() do?**

It checks whether a specified file exists.

6. **What does file_get_contents() do?**

   It reads the contents of a file.

7. **What does file_put_contents() do?**

   It writes data into a file.

8. **What is MySQL?**

   MySQL is a relational database management system.

9. **What is mysqli?**

   MySQLi is a PHP extension used to interact with MySQL databases.

10. **What is Selection Sort?**

    Selection Sort repeatedly selects the smallest element from the unsorted portion and places it in its correct position.

11. **What is the time complexity of Selection Sort?**

$$O(n^2)$$

12. **Which field is used for sorting in this example?**

    Student marks.

13. **In which order are the records sorted?**

    Ascending order of marks.

14. **Why are database records first stored in a PHP array?**

    The experiment applies the Selection Sort algorithm in PHP to the retrieved records.

15. **What does fetch_assoc() return?**

    It returns a result row as an associative array.

16. **What is the purpose of htmlspecialchars() in this program?**

    It safely escapes special HTML characters before database values are displayed in the webpage.

17. **Does the simple visitor counter count unique users?**

    No. It increments whenever the PHP page is requested.

18. **Why is PHP different from normal browser JavaScript?**

    PHP executes on the server, while normal browser JavaScript executes in the browser.

19. **Which platform can students use for Search Creators coding practice?**

Search Creators Compiler / Code Studio

22. Quick Revision

Concept	Remember
Experiment 08(a)	Visitor Counter
Language	PHP
Visitor Storage	`counter.txt`
Read File	`file_get_contents()`
Write File	`file_put_contents()`
Experiment 08(b)	Student Record Sorting
Database	MySQL
PHP Database API	MySQLi
Sorting Algorithm	Selection Sort
Example Sorting Field	Marks
Sorting Order	Ascending
Selection Sort Complexity	$O(n^2)$
Database Fetch	`fetch_assoc()`
Output Safety	`htmlspecialchars()`
Frontend	HTML + CSS
Practice Platform	Search Creators Compiler
Website	`searchcreators.org/compiler/`

23. Result

Result

Part (a):
A PHP web application was developed to keep track of page visits using a visitor counter.

Part (b):
A PHP web application was developed to retrieve student records stored in a database and sort them using the **Selection Sort** algorithm.
HTML and CSS were used to present the results in the browser.

For Search Creators coding practice:

Search Creators Compiler / Code Studio

`https://searchcreators.org/compiler/`

END OF EXPERIMENT – 08

BCSL504 – Web Technology Lab

Prepared by Search Creators

Practice Online: searchcreators.org/compiler/