Date: 01-04-2025

Lab Manual Introduction to PHP and MongoDB for Experiment

Objective:

To implement the following using PHP MongoDB:

- Understand PHP syntax and server-side scripting fundamentals
- Learn MongoDB document-based database structure
- Connect PHP applications to MongoDB databases
- Perform CRUD operations using PHP and MongoDB
- Build basic web applications with PHP backend and MongoDB storage

Pre-requisite tools:

- Basic understanding of programming concepts
- Familiarity with HTML and web technologies
- Local development environment setup (XAMPP/WAMP/MAMP or equivalent)
- MongoDB Community Server installed
- PHP MongoDB driver installed
- Text editor (VS Code, Sublime Text)
- Postman or similar API testing tool

PHP Theory

PHP (Hypertext Preprocessor) is a widely-used open source server-side scripting language designed for web development. It executes on the server and generates HTML which is then sent to the client. PHP can connect to various databases, handle forms, manage user sessions, and build complete web applications.

MongoDB Theory:

MongoDB is a document-oriented NoSQL database that stores data in flexible, JSON-like documents. Unlike relational databases, MongoDB has a dynamic schema which means documents in the same collection can have different fields. It provides high performance, high availability, and easy scalability.

Key Features of PHP:

- Server-side execution - Runs on web server before page is sent to browser
- Cross-platform compatibility - Works on Windows, Linux, macOS
- Extensive library support - Built-in functions for common web tasks
- Database connectivity - Supports MySQL, PostgreSQL, MongoDB, and more
- Session management - Built-in support for user sessions
- Object-oriented capabilities - Full OOP support since PHP 5
- Error handling - Robust error reporting and handling mechanisms

Key Features of MongoDB:

- Document-oriented storage - Data stored as BSON (Binary JSON) documents
- Schema-less design - Flexible document structure
- Horizontal scalability - Sharding for distributed data
- High performance - Indexing and memory-mapped files
- Rich query language - Supports ad-hoc queries, indexing, and real-time aggregation
- Replication - High availability with replica sets
- GridFS - For storing and retrieving large files

PHP Example:

```
<?php
```

```
// Variables and data types

$name = "Nikhil Kumar";  // String

$Branch ="M.tech Data science"; // String
$Rollno = 2354004;      // Integer

$CGPA = 8.5;              // Float

$isActive = true;         // Boolean


// Output

echo "Hello, $name! Your  Branch $Branch Rollno $Rollno CGPA $CGPA.<br>";

?>
```

Array Example

```
<?php

// Indexed array

$colors = ["Red", "Green", "Blue"];

echo $colors[1]; // Output: Green


// Associative array

$user = [

    "name" => "NITP",

    "email" => "Nitp@example.com"

];

echo $user["email"]; // Output: Nitp@example.com

?>
```

LOOP

```
<?php

// For loop
```

```
for ($i = 1; $i <= 3; $i++) {
    echo "Count: $i<br>";
}

// Foreach loop
$fruits = ["Apple", "Banana", "Cherry"];
foreach ($fruits as $fruit) {
    echo "$fruit<br>";
}
?>
```

Form Handling

HTML Form (form.html):

```
<form action="submit.php" method="POST">
    <input type="text" name="username" placeholder="Username">
    <button type="submit">Submit</button>
</form>
```

Key Notes:

1. Always start PHP files with <?php.
2. Use echo to output data.
3. End statements with ;.
4. For security:
    - Sanitize inputs with htmlspecialchars().
    - Use prepared statements for databases.

Setup & Connection MongoDB:

```
<?php
require 'vendor/autoload.php'; // Include Composer's autoloader
```

```
// Connect to MongoDB (default: localhost)
$client = new MongoDB\Client("mongodb://localhost:27017");

// Select database and collection
$db = $client->test_database;
$collection = $db->users;

echo "Connected to MongoDB!";
?>
```

Assignment

1. Write a PHP script to connect to a MongoDB database and insert a document into a collection named "users" with fields for name, email, and registration date.
2. Create a PHP form that accepts product details (name, price, category) and stores them in a MongoDB "products" collection. Include basic validation for the input fields.
3. Write a PHP function that queries a MongoDB collection for all documents where a "status" field equals "active" and returns the results as a JSON array.
4. Implement a PHP script that updates a document in MongoDB based on a provided _id, changing only the fields that are provided in the update request.
5. Create a simple PHP application that demonstrates pagination of results from a MongoDB collection, showing 5 documents per page with navigation links.

Learning Outcomes

After completing this lab session, students should be able to:

- Understand PHP syntax and server-side execution
- Comprehend NoSQL concepts and MongoDB document structure

- Perform CRUD operations using PHP and MongoDB
- Handle data validation and error management in PHP
- Build basic web applications with PHP backend and MongoDB storage

References

1. PHP Manual - https://www.php.net/manual/
2. MongoDB PHP Driver Documentation -
3. "PHP and MongoDB Web Development" by Rubayeet Islam
4. "MongoDB: The Definitive Guide" by Shannon Bradshaw
5. MongoDB - https://docs.mongodb.com/drivers/php/