MIT Art, Design and Technology University,
Loni Kalbhor, Pune

MIT College of Management (MITCOM)

A
Laboratory Manual For

PHP Framework
(23MCAC304)

M.C.A.-II Semester –III

Prof. Deepak V. Ulape

Asst. Professor.

(M.C.A., Ph.D. Pursuing)

Sr. No	Title of the Practicals
1.	Write a PHP Program in CodeIgniter to determine given number is Even or ODD.
2.	Write a PHP Program in CodeIgniter to check if a given number is divisible by 3, and display an appropriate message.
3.	Write a PHP Program in CodeIgniter to displays the name of the day based on a given number.
4.	Write a PHP Program in CodeIgniter to evaluate a score and display the corresponding grade using CodeIgniter.
5.	Write a PHP Program in CodeIgniter to calculates the sum of natural numbers up to a specified limit.
6	Write a PHP Program in CodeIgniter to generates and displays a multiplication table for a specified number using do while loop.
7	Write a PHP Program in CodeIgniter to calculates the factorial of a given number using a for loop.
8	Write a PHP Program in CodeIgniter to that generates the Fibonacci series up to a specified number of terms.
9	Write a PHP Program in CodeIgniter to that iterates through an array of student names and displays them using simple array.
10	Write a PHP Program in CodeIgniter to Write a PHP program to create an indexed array of fruits and display them.
11	Write a PHP Program in CodeIgniter to calculate the length of String.
12	Write a PHP Program in CodeIgniter to count the number of words in string without using string functions
13	Write a PHP Program in CodeIgniter to to demonstrate use of various built-in string functions.
14	Create a CodeIgniter PHP program that demonstrates inheritance with an Animal superclass (with properties name and age and a speak() method) and a Dog subclass that overrides speak() to include the dog's name and age.
15	Write a PHP Program in CodeIgniter to Create a Car_model class with a constructor to initialize properties like make, model, and year etc.
16	Write a PHP program in CodeIgniter to design a web page featuring a text box for name input, radio buttons for selecting a contact method (Email or Phone), check boxes for choosing interests (Sports, Music, Reading), and buttons for submitting or resetting the form.

17	Write a simple PHP program in CodeIgniter that demonstrates introspection and serialization. Use a class to create an object, and then showcase how to inspect its properties and methods using PHP's reflection
18	Write a PHP program in CodeIgniter to implement session management and cookie handling for a user login system
19	Write a PHP program in CodeIgniter to perform the following tasks: a) Create a form to enter user information (name and email) and save this data into a database. b) Retrieve and display the saved user information in a table format on a separate page.
20	Write a PHP program in CodeIgniter to develop a simple application that allows users to Update existing records by modifying user information (e.g., name and email).

Practical NO 1:

Write a PHP Program in CodeIgniter to determine given number is Even or ODD.

Steps:

1. **Setup CodeIgniter:**
   - Download the CodeIgniter framework from CodeIgniter's official site.
   - Extract it to your server's document root (e.g., htdocs for XAMPP).
2. **Create a Controller:**
   - Navigate to application/controllers.
   - Create a new file named NumberCheck.php and add the provided controller code.
3. **Create Views:**
   - Navigate to application/views.
   - Create two new files: number_check_form.php and number_check_result.php.
   - Add the respective HTML code provided above.
4. **Configure Routes:**
   - Open application/config/routes.php.
   - Set the default controller to NumberCheck as shown.
5. **Testing the Application:**
   - Start your local server (e.g., XAMPP).
   - Navigate to http://localhost/your_project_folder in your web browser.
   - Enter a number in the input field and submit the form.
   - Observe the result indicating whether the number is even or odd.

CodeIgniter Program

1. **Setup CodeIgniter**: Make sure you have CodeIgniter set up on your local server.
2. **Create a Controller**: Create a new controller named NumberCheck.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class NumberCheck extends CI_Controller {
    public function index() {
        $this->load->view('number_check_form');
```

```
    }
    public function check() {
        $number = $this->input->post('number');
        if ($number % 2 == 0) {
            $result = "$number is Even.";
        } else {
            $result = "$number is Odd.";
        }
        $data['result'] = $result;
        $this->load->view('number_check_result', $data);
    }
}
```

3. **Create Views**:
   - **View for Input Form (number_check_form.php)**:

```
<!DOCTYPE html>
<html>
<head>
    <title>Even or Odd Checker</title>
</head>
<body>
    <h1>Even or Odd Checker</h1>
    <form method="post" action="<?php echo site_url('NumberCheck/check'); ?>">
        <label for="number">Enter a Number:</label>
        <input type="number" name="number" required>
        <input type="submit" value="Check">
    </form>
</body>
</html>
```

   - **View for Result (number_check_result.php)**:

```
<!DOCTYPE html>
<html>
<head>
```

```
    <title>Result</title>
</head>
<body>
    <h1>Result</h1>
    <p><?php echo $result; ?></p>
    <a href="<?php echo site_url('NumberCheck'); ?>">Check another number</a>
</body>
</html>
```

4. **Configure Routes**: Add routes to application/config/routes.php.

```
$route['default_controller'] = 'NumberCheck';
```

Practical NO 2:

Write a PHP Program in CodeIgniter to check if a given number is divisible by 3, and display an appropriate message.

Steps:

1. **Setup CodeIgniter**:
    - Download the CodeIgniter framework from CodeIgniter's official site.
    - Extract it to your server's document root (e.g., htdocs for XAMPP).
2. **Create a Controller**:
    - Navigate to application/controllers.
    - Create a new file named DivisibilityCheck.php and add the provided controller code.
3. **Create Views**:
    - Navigate to application/views.
    - Create two new files: divisibility_check_form.php and divisibility_check_result.php.
    - Add the respective HTML code provided above.
4. **Configure Routes**:
    - Open application/config/routes.php.
    - Set the default controller to DivisibilityCheck as shown.
5. **Testing the Application**:
    - Start your local server (e.g., XAMPP).
    - Navigate to http://localhost/your_project_folder in your web browser.
    - Enter a number in the input field and submit the form.
    - Observe the result indicating whether the number is divisible by 3 or not.

CodeIgniter Program

1. **Create a Controller**: Create a new controller named DivisibilityCheck.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class DivisibilityCheck extends CI_Controller {
    public function index() {
        $this->load->view('divisibility_check_form');
    }

    public function check() {
```

```
        $number = $this->input->post('number');
        if ($number % 3 == 0) {
            $result = "$number is divisible by 3.";
        } else {
            $result = "$number is not divisible by 3.";
        }
        $data['result'] = $result;
        $this->load->view('divisibility_check_result', $data);
    }
}
```

2. **Create Views**:

- **View for Input Form (divisibility_check_form.php)**:

```
<!DOCTYPE html>
<html>
<head>
    <title>Divisibility Checker</title>
</head>
<body>
    <h1>Divisibility Checker</h1>
    <form method="post" action="<?php echo site_url('DivisibilityCheck/check'); ?>">
        <label for="number">Enter a Number:</label>
        <input type="number" name="number" required>
        <input type="submit" value="Check">
    </form>
</body>
</html>
```

- **View for Result (divisibility_check_result.php)**:

```
<!DOCTYPE html>
<html>
<head>
    <title>Result</title>
</head>
```

```
<body>
    <h1>Result</h1>
    <p><?php echo $result; ?></p>
    <a href="<?php echo site_url('DivisibilityCheck'); ?>">Check another number</a>
</body>
</html>
```

3. **Configure Routes**: Add the following route to application/config/routes.php.

```
$route['default_controller'] = 'DivisibilityCheck';
```

Practical NO 3:

Write a PHP Program in CodeIgniter to displays the name of the day based on a given number.

Steps:

1. **Create a Controller**:
    - Navigate to application/controllers.
    - Create a new file named DayName.php and add the provided controller code.
2. **Create Views**:
    - Navigate to application/views.
    - Create two new files: day_name_form.php and day_name_result.php.
    - Add the respective HTML code provided above.
3. **Configure Routes**:
    - Open application/config/routes.php.
    - Set the default controller to DayName as shown.
4. **Testing the Application**:
    - Start your local server (e.g., XAMPP).
    - Navigate to http://localhost/your_project_folder in your web browser.
    - Enter a number between 1 and 7 in the input field and submit the form.

CodeIgniter Program

1. **Create a Controller**: Create a new controller named DayName.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class DayName extends CI_Controller {
    public function index() {
        $this->load->view('day_name_form');
    }

    public function check() {
        $dayNumber = $this->input->post('day_number');
        $days = [
            1 => "Sunday",
            2 => "Monday",
```

```
            3 => "Tuesday",
            4 => "Wednesday",
            5 => "Thursday",
            6 => "Friday",
            7 => "Saturday"
        ];
        $result = isset($days[$dayNumber]) ?
       $days[$dayNumber] : "Invalid number! Please enter a number between 1 and 7.";
        $data['result'] = $result;
        $this->load->view('day_name_result', $data);
    }
}
```

2. **Create Views**:

- **View for Input Form (day_name_form.php)**:

```
<!DOCTYPE html>
<html>
<head>
    <title>Day Name Finder</title>
</head>
<body>
    <h1>Find the Name of the Day</h1>
    <form method="post" action="<?php echo site_url('DayName/check'); ?>">
        <label for="day_number">Enter a number (1-7):</label>
        <input type="number" name="day_number" min="1" max="7" required>
        <input type="submit" value="Get Day Name">
    </form>
</body>
</html>
```

- **View for Result (day_name_result.php)**:

```
<!DOCTYPE html>
<html>
```

```
<head>
    <title>Result</title>
</head>
<body>
    <h1>Result</h1>
    <p><?php echo $result; ?></p>
    <a href="<?php echo site_url('DayName'); ?>">Check another number</a>
</body>
</html>
```

3. **Configure Routes**: Add the following route to application/config/routes.php.

```
$route['default_controller'] = 'DayName';
```

Practical NO 4:

Write a PHP Program in CodeIgniter to evaluate a score and display the corresponding grade using CodeIgniter.

Steps:

1. **Create a Controller**:
    - Navigate to application/controllers.
    - Create a new file named GradeEvaluator.php and add the provided controller code.
2. **Create Views**:
    - Navigate to application/views.
    - Create two new files: grade_evaluator_form.php and grade_evaluator_result.php.
    - Add the respective HTML code provided above.
3. **Configure Routes**:
    - Open application/config/routes.php.
    - Set the default controller to GradeEvaluator as shown.
4. **Testing the Application**:
    - Start your local server (e.g., XAMPP).
    - Navigate to http://localhost/your_project_folder in your web browser.
    - Enter a score between 0 and 100 in the input field and submit the form.
    - Observe the result displaying the corresponding grade.

CodeIgniter Program

1. **Setup CodeIgniter**: Make sure you have CodeIgniter installed and configured on your local server.
2. **Create a Controller**: Create a new controller named GradeEvaluator.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class GradeEvaluator extends CI_Controller {
  public function index() {
     $this->load->view('grade_evaluator_form');
   }

   public function evaluate() {
     $score = $this->input->post('score');
```

```
        // Determine the grade based on the score
        if ($score >= 90 && $score <= 100) {
            $grade = 'A';
        } elseif ($score >= 80) {
            $grade = 'B';
        } elseif ($score >= 70) {
            $grade = 'C';
        } elseif ($score >= 60) {
            $grade = 'D';
        } elseif ($score >= 0) {
            $grade = 'F';
        } else {
            $grade = 'Invalid score! Please enter a score between 0 and 100.';
        }
        $data['result'] = "Score: $score, Grade: $grade";
        $this->load->view('grade_evaluator_result', $data);
    }
}
```

3. **Create Views**:

- **View for Input Form (grade_evaluator_form.php)**:

```
<!DOCTYPE html>
<html>
<head>
    <title>Grade Evaluator</title>
</head>
<body>
    <h1>Grade Evaluator</h1>
    <form method="post" action="<?php echo site_url('GradeEvaluator/evaluate'); ?>">
        <label for="score">Enter the Score (0-100):</label>
        <input type="number" name="score" min="0" max="100" required>
        <input type="submit" value="Evaluate Grade">
    </form>
```

```
</body>
</html>
```

- **View for Result (grade_evaluator_result.php)**:

```
<!DOCTYPE html>
<html>
<head>
    <title>Result</title>
</head>
<body>
    <h1>Result</h1>
    <p><?php echo $result; ?></p>
    <a href="<?php echo site_url('GradeEvaluator'); ?>">Evaluate another score</a>
</body>
</html>
```

4. **Configure Routes**: Add the following route to application/config/routes.php.

```
$route['default_controller'] = 'GradeEvaluator';
```

Practical NO 5:

Write a PHP Program in CodeIgniter to calculates the sum of natural numbers up to a specified limit.

Steps:

1. **Create a Controller**:
    - Navigate to application/controllers.
    - Create a new file named SumNaturalNumbers.php and add the provided controller code.
2. **Create Views**:
    - Navigate to application/views.
    - Create two new files: sum_natural_numbers_form.php and sum_natural_numbers_result.php.
    - Add the respective HTML code provided above.
3. **Configure Routes**:
    - Open application/config/routes.php.
    - Set the default controller to SumNaturalNumbers as shown.
4. **Testing the Application**:
    - Start your local server (e.g., XAMPP).
    - Navigate to http://localhost/your_project_folder in your web browser.
    - Enter a non-negative number in the input field and submit the form.
    - Observe the result displaying the sum of natural numbers up to the specified limit.

CodeIgniter Program

1. **Create a Controller**: Create a new controller named SumNaturalNumbers.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class SumNaturalNumbers extends CI_Controller {
    public function index() {
        $this->load->view('sum_natural_numbers_form');
    }
    public function calculate() {
        $limit = $this->input->post('limit');
        // Validate input
```

```
        if ($limit < 0) {
            $result = "Please enter a non-negative number.";
        } else {
            // Calculate the sum of natural numbers
            $sum = ($limit * ($limit + 1)) / 2;
            $result = "The sum of natural numbers up to $limit is $sum.";
        }
        $data['result'] = $result;
        $this->load->view('sum_natural_numbers_result', $data);
    }
}
```

2. **Create Views**:

- **View for Input Form (sum_natural_numbers_form.php)**:

```
<!DOCTYPE html>
<html>
<head>
    <title>Sum of Natural Numbers</title>
</head>
<body>
    <h1>Calculate Sum of Natural Numbers</h1>
    <form method="post" action="<?php echo site_url('SumNaturalNumbers/calculate'); ?>">
        <label for="limit">Enter the Limit:</label>
        <input type="number" name="limit" min="0" required>
        <input type="submit" value="Calculate Sum">
    </form>
</body>
</html>
```

- **View for Result (sum_natural_numbers_result.php)**:

```
<!DOCTYPE html>
<html>
<head>
    <title>Result</title>
```

```
</head>
<body>
    <h1>Result</h1>
    <p><?php echo $result; ?></p>
    <a href="<?php echo site_url('SumNaturalNumbers'); ?>">Calculate another sum</a>
</body>
</html>
```

3. **Configure Routes**: Add the following route to application/config/routes.php.

```
$route['default_controller'] = 'SumNaturalNumbers';
```

Practical NO 6:

Write a PHP Program in CodeIgniter to generates and displays a multiplication table for a specified number using do while loop.

Steps:

1. **Create a Controller**:
    - Navigate to application/controllers.
    - Create a new file named MultiplicationTable.php and add the provided controller code.
2. **Create Views**:
    - Navigate to application/views.
    - Create two new files: multiplication_table_form.php and multiplication_table_result.php.
    - Add the respective HTML code provided above.
3. **Configure Routes**:
    - Open application/config/routes.php.
    - Set the default controller to MultiplicationTable as shown.
4. **Testing the Application**:
    - Start your local server (e.g., XAMPP).
    - Navigate to http://localhost/your_project_folder in your web browser.
    - Enter a number in the input field and submit the form.
    - Observe the multiplication table generated and displayed as a list.

CodeIgniter Program

1. **Setup CodeIgniter**: Ensure you have CodeIgniter installed and configured on your local server.
2. **Create a Controller**: Create a new controller named MultiplicationTable.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class MultiplicationTable extends CI_Controller {
    public function index() {
        $this->load->view('multiplication_table_form');
    }

    public function generate() {
```

```
        $number = $this->input->post('number');
        $table = [];
        // Generate multiplication table using do while loop
        $i = 1;
        do {
            $table[] = "$number x $i = " . ($number * $i);
            $i++;
        } while ($i <= 10);
        $data['table'] = $table;
        $this->load->view('multiplication_table_result', $data);
    }
}
```

3. **Create Views**:

- **View for Input Form (multiplication_table_form.php)**:

```
<!DOCTYPE html>
<html>
<head>
    <title>Multiplication Table</title>
</head>
<body>
    <h1>Generate Multiplication Table</h1>
    <form method="post" action="<?php echo site_url('MultiplicationTable/generate'); ?>">
        <label for="number">Enter a Number:</label>
        <input type="number" name="number" required>
        <input type="submit" value="Generate Table">
    </form>
</body>
</html>
```

- **View for Result (multiplication_table_result.php)**:

```
<!DOCTYPE html>
<html>
<head>
```

```
    <title>Multiplication Table Result</title>
</head>
<body>
    <h1>Multiplication Table</h1>
    <ul>
        <?php foreach ($table as $line): ?>
            <li><?php echo $line; ?></li>
        <?php endforeach; ?>
    </ul>
    <a href="<?php echo site_url('MultiplicationTable'); ?>">Generate another table</a>
</body>
</html>
```

4. **Configure Routes**: Add the following route to application/config/routes.php.

```
$route['default_controller'] = 'MultiplicationTable';
```

Practical NO 7:

Write a PHP Program in CodeIgniter to calculates the factorial of a given number using a for loop.

Steps:

1. **Create a Controller**:
    - Navigate to application/controllers.
    - Create a new file named FactorialCalculator.php and add the provided controller code.
2. **Create Views**:
    - Navigate to application/views.
    - Create two new files: factorial_form.php and factorial_result.php.
    - Add the respective HTML code provided above.
3. **Configure Routes**:
    - Open application/config/routes.php.
    - Set the default controller to FactorialCalculator as shown.
4. **Testing the Application**:
    - Start your local server (e.g., XAMPP).
    - Navigate to http://localhost/your_project_folder in your web browser.
    - Enter a non-negative integer in the input field and submit the form.
    - Observe the result displaying the factorial of the entered number.

CodeIgniter Program

1. **Setup CodeIgniter**: Ensure you have CodeIgniter installed and configured on your local server.
2. **Create a Controller**: Create a new controller named FactorialCalculator.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class FactorialCalculator extends CI_Controller {

    public function index() {
        $this->load->view('factorial_form');
    }
    public function calculate() {
        $number = $this->input->post('number');
```

```
        $factorial = 1;
        // Calculate factorial using a for loop
        if ($number < 0) {
            $result = "Factorial is not defined for negative numbers.";
        } else {
            for ($i = 1; $i <= $number; $i++) {
                $factorial *= $i;
            }
            $result = "The factorial of $number is $factorial.";
        }
        $data['result'] = $result;
        $this->load->view('factorial_result', $data);
    }
}
```

3. **Create Views**:

- **View for Input Form (factorial_form.php)**:

```
<!DOCTYPE html>
<html>
<head>
    <title>Factorial Calculator</title>
</head>
<body>
    <h1>Calculate Factorial</h1>
    <form method="post" action="<?php echo site_url('FactorialCalculator/calculate'); ?>">
        <label for="number">Enter a Non-Negative Integer:</label>
        <input type="number" name="number" min="0" required>
        <input type="submit" value="Calculate Factorial">
    </form>
</body>
</html>
```

- **View for Result (factorial_result.php)**:

```
<!DOCTYPE html>
```

```
<html>
<head>
    <title>Factorial Result</title>
</head>
<body>
    <h1>Result</h1>
    <p><?php echo $result; ?></p>
    <a href="<?php echo site_url('FactorialCalculator'); ?>">Calculate another factorial</a>
</body>
</html>
```

4. **Configure Routes**: Add the following route to application/config/routes.php.

```
$route['default_controller'] = 'FactorialCalculator';
```

Practical NO 8:

Write a PHP Program in CodeIgniter to that generates the Fibonacci series up to a specified number of terms.

Steps:

1. **Create a Controller**:
    - Navigate to application/controllers.
    - Create a new file named FibonacciSeries.php and add the provided controller code.
2. **Create Views**:
    - Navigate to application/views.
    - Create two new files: fibonacci_form.php and fibonacci_result.php.
    - Add the respective HTML code provided above.
3. **Configure Routes**:
    - Open application/config/routes.php.
    - Set the default controller to FibonacciSeries as shown.
4. **Testing the Application**:
    - Start your local server (e.g., XAMPP).
    - Navigate to http://localhost/your_project_folder in your web browser.
    - Enter a positive integer in the input field and submit the form.
    - Observe the Fibonacci series generated and displayed.

CodeIgniter Program

1. **Setup CodeIgniter**: Ensure you have CodeIgniter installed and configured on your local server.
2. **Create a Controller**: Create a new controller named FibonacciSeries.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class FibonacciSeries extends CI_Controller {
    public function index() {
        $this->load->view('fibonacci_form');
    }
    public function generate() {
        $terms = $this->input->post('terms');
        $fibonacci = [];
```

```
        // Generate Fibonacci series
        if ($terms <= 0) {
            $result = "Please enter a positive integer.";
        } else {
            $fibonacci[0] = 0;
            if ($terms > 1) {
                $fibonacci[1] = 1;
                for ($i = 2; $i < $terms; $i++) {
                    $fibonacci[$i] = $fibonacci[$i - 1] + $fibonacci[$i - 2];
                }
            }
            $result = "Fibonacci series up to $terms terms: " . implode(", ", $fibonacci);
        }
        $data['result'] = $result;
        $this->load->view('fibonacci_result', $data);
    }
}
```

3. **Create Views**:

- **View for Input Form (fibonacci_form.php)**:

```
<!DOCTYPE html>
<html>
<head>
    <title>Fibonacci Series Generator</title>
</head>
<body>
    <h1>Generate Fibonacci Series</h1>
    <form method="post" action="<?php echo site_url('FibonacciSeries/generate'); ?>">
        <label for="terms">Enter the number of terms:</label>
        <input type="number" name="terms" min="1" required>
        <input type="submit" value="Generate Series">
    </form>
</body>
```

```
</html>
```

- **View for Result (fibonacci_result.php)**:

```
<!DOCTYPE html>
<html>
<head>
    <title>Fibonacci Series Result</title>
</head>
<body>
    <h1>Result</h1>
    <p><?php echo $result; ?></p>
    <a href="<?php echo site_url('FibonacciSeries'); ?>">Generate another series</a>
</body>
</html>
```

4. **Configure Routes**: Add the following route to application/config/routes.php.

```
$route['default_controller'] = 'FibonacciSeries';
```

Practical NO 9:

Write a PHP Program in CodeIgniter to that iterates through an array of student names and displays them using simple array.

Steps:

1. **Create a Controller:**
    - Navigate to application/controllers.
    - Create a new file named StudentList.php and add the provided controller code.
2. **Create Views:**
    - Navigate to application/views.
    - Create a new file named student_list.php.
    - Add the HTML code provided above to display the student names.
3. **Configure Routes:**
    - Open application/config/routes.php.
    - Set the default controller to StudentList as shown.
4. **Testing the Application:**
    - Start your local server (e.g., XAMPP).
    - Navigate to http://localhost/your_project_folder in your web browser.
    - You should see a list of student names displayed on the page.

CodeIgniter Program

1. Setup CodeIgniter: Ensure you have CodeIgniter installed and configured on your local server.
2. Create a Controller: Create a new controller named StudentList.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class StudentList extends CI_Controller {
  public function index() {
    $students = ["Alice", "Bob", "Charlie", "David", "Eva"];
    $data['students'] = $students;
    $this->load->view('student_list', $data);
  }
}
```

3. **Create Views:**

- View for Displaying Student Names (student_list.php):

```
<!DOCTYPE html>
<html>
<head>
  <title>Student List</title>
</head>
<body>
  <h1>List of Students</h1>
  <ul>
    <?php foreach ($students as $student): ?>
      <li><?php echo $student; ?></li>
    <?php endforeach; ?>
  </ul>
</body>
</html>
```

4. Configure Routes: Add the following route to application/config/routes.php.

```
$route['default_controller'] = 'StudentList';
```

Practical NO 10:

Write a PHP Program in CodeIgniter to Write a PHP program to create an indexed array of fruits and display them.

Steps

1. **Setup CodeIgniter**
    - Download CodeIgniter from the official website.
    - Extract the files and set up the project in your local server's root directory.
2. **Create a Controller**
    - Navigate to application/controllers.
    - Create a new file named Fruits.php.
    - Copy and paste the provided code to define the controller and the index method that initializes an indexed array of fruits.
3. **Create a View**
    - Navigate to application/views.
    - Create a new file named fruits_view.php.
    - Copy and paste the provided HTML and PHP code to create a view that displays the fruits in a list format.
4. **Configure Routes**
    - Open application/config/routes.php.
    - Set the default controller to fruits by modifying the route.
5. **Run the Application**
    - Start your local server.
    - Navigate to http://localhost/your_project_name in your web browser.

CodeIgniter Program

Step1: Create a Controller

Create a new controller named Fruits.php in the application/controllers directory.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class Fruits extends CI_Controller {
    public function index() {
        // Create an indexed array of fruits
        $fruits = array("Apple", "Banana", "Cherry", "Date", "Elderberry");
```

```
        // Load the view and pass the fruits array
        $this->load->view('fruits_view', ['fruits' => $fruits]);
    }
}
```

Step 2: Create a View

Create a new view file named fruits_view.php in the application/views directory.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Fruits List</title>
</head>
<body>
    <h1>List of Fruits</h1>
    <ul>
        <?php foreach ($fruits as $fruit): ?>
            <li><?php echo $fruit; ?></li>
        <?php endforeach; ?>
    </ul>
</body>
</html>
```

Step 3: Configure Routes

Open application/config/routes.php and set the default controller to Fruits.

```
$route['default_controller'] = 'fruits';
```

Practical NO 11:

Write a PHP Program in CodeIgniter to calculate the length of String.

Steps to Create the Program:

1. **Setup CodeIgniter**:
   - Download and install CodeIgniter from the official website.
   - Extract the files and place them in the web server's root directory.
2. **Create the Controller**:
   - Navigate to application/controllers/.
   - Create a file named StringLength.php.
   - Implement the methods to display the input form and calculate the string length.
3. **Create the Views**:
   - Navigate to application/views/.
   - Create string_length_form.php for the input form.
   - Create string_length_result.php to display the result after calculation.
4. **Configure Routes**:
   - Open application/config/routes.php.
   - Add the route for the controller:

     $route['stringlength'] = 'stringlength/index';
5. **Run the Application**:
   - Start your web server.
   - Open a web browser and navigate to http://localhost/your_project_directory/index.php/stringlength.

CodeIgniter Program to Calculate String Length

1. **Controller**: Create a new controller called StringLength.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class StringLength extends CI_Controller {
  public function index() {
      $this->load->view('string_length_form');
   }

   public function calculate() {
```

```
        $input_string = $this->input->post('input_string');
        $length = strlen($input_string);
        $data['length'] = $length;
        $data['input_string'] = $input_string;
        $this->load->view('string_length_result', $data);
    }
}
?>
```

2. **View for Input Form**: Create a view file named string_length_form.php.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>String Length Calculator</title>
</head>
<body>
    <h1>Calculate String Length</h1>
    <form action="<?php echo site_url('stringlength/calculate'); ?>" method="post">
        <label for="input_string">Enter a string:</label>
        <input type="text" name="input_string" id="input_string" required>
        <input type="submit" value="Calculate Length">
    </form>
</body>
</html>
```

3. **View for Result**: Create another view file named string_length_result.php.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>String Length Result</title>
</head>
<body>
```

```
    <h1>String Length Result</h1>

    <p>The length of the string "<?php echo $input_string; ?>" is: <?php echo $length; ?> characters.</p>

    <a href="<?php echo site_url('stringlength'); ?>">Calculate another string</a>

</body>

</html>
```

Practical NO 12:

Write a PHP Program in CodeIgniter to count the number of words in string without using string functions

Steps to Create the Program:

1. **Create the Controller**:
    - Navigate to application/controllers/.
    - Create a file named WordCount.php.
    - Implement methods to display the input form and to count the words.
2. **Create the Views**:
    - Navigate to application/views/.
    - Create word_count_form.php for the input form.
    - Create word_count_result.php to display the result.
3. **Configure Routes**:
    - Open application/config/routes.php.
    - Add the following route:

        $route['wordcount'] = 'wordcount/index';
4. **Run the Application**:
    - Start your web server.
    - Open a web browser and navigate to http://localhost/your_project_directory/index.php/wordcount.

CodeIgniter Program to Count Words in a String

1. **Controller**: Create a new controller called WordCount.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class WordCount extends CI_Controller {
  public function index() {
    $this->load->view('word_count_form');
  }
  public function count_words() {
    $input_string = $this->input->post('input_string');
    $word_count = $this->calculate_word_count($input_string);
    $data['word_count'] = $word_count;
```

```
        $data['input_string'] = $input_string;
        $this->load->view('word_count_result', $data);
    }
    private function calculate_word_count($string) {
        $count = 0;
        $in_word = false;
        for ($i = 0; $i < strlen($string); $i++) {
            if ($string[$i] != ' ') {
                if (!$in_word) {
                    $in_word = true;
                    $count++;
                }
            } else {
                $in_word = false;
            }
        }
        return $count;
    }
}
?>
```

2. **View for Input Form**: Create a view file named word_count_form.php.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Word Count Calculator</title>
</head>
<body>
    <h1>Calculate Word Count</h1>
    <form action="<?php echo site_url('wordcount/count_words'); ?>" method="post">
        <label for="input_string">Enter a string:</label>
        <input type="text" name="input_string" id="input_string" required>
```

```
        <input type="submit" value="Count Words">
    </form>
</body>
</html>
```

3. **View for Result**: Create another view file named word_count_result.php.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Word Count Result</title>
</head>
<body>
    <h1>Word Count Result</h1>
    <p>The number of words in the string "<?php echo htmlspecialchars($input_string); ?>" is: <?php echo $word_count; ?>.</p>
    <a href="<?php echo site_url('wordcount'); ?>">Count another string</a>
</body>
</html>
```

Practical NO 13:

Write a PHP Program in CodeIgniter to demonstrate use of various built-in string functions.

Steps to Create the Program:

1. **Create the Controller**:
    - Navigate to application/controllers/.
    - Create a file named StringFunctions.php.
    - Implement methods to display the input form and demonstrate various string functions.
2. **Create the Views**:
    - Navigate to application/views/.
    - Create string_functions_form.php for the input form.
    - Create string_functions_result.php to display results for various string functions.
3. **Configure Routes**:
    - Open application/config/routes.php.
    - Add the following route: $route['stringfunctions'] = 'stringfunctions/index';
4. **Run the Application**:
    - Start your web server.
    - Open a web browser and navigate to http://localhost/your_project_directory/index.php/stringfunctions.

CodeIgniter Program

1. **Controller**: Create a new controller called StringFunctions.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class StringFunctions extends CI_Controller {
   public function index() {
      $this->load->view('string_functions_form');
   }
  public function demonstrate() {
      $input_string = $this->input->post('input_string');

      // Demonstrating various string functions
```

```
        $data['original'] = $input_string;
        $data['length'] = strlen($input_string);
        $data['uppercase'] = strtoupper($input_string);
        $data['lowercase'] = strtolower($input_string);
        $data['reversed'] = strrev($input_string);
        $data['word_count'] = str_word_count($input_string);
        $data['substring'] = substr($input_string, 0, 5); // First 5 characters
        $this->load->view('string_functions_result', $data);
    }
}
?>
```

2. **View for Input Form**: Create a view file named string_functions_form.php.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>String Functions Demonstration</title>
</head>
<body>
    <h1>Demonstrate Built-in String Functions</h1>
    <form action="<?php echo site_url('stringfunctions/demonstrate'); ?>" method="post">
        <label for="input_string">Enter a string:</label>
        <input type="text" name="input_string" id="input_string" required>
        <input type="submit" value="Demonstrate">
    </form>
</body>
</html>
```

3. **View for Results**: Create another view file named string_functions_result.php.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
```

```
    <title>String Functions Result</title>
</head>
<body>
    <h1>String Functions Result</h1>
    <p><strong>Original String:</strong> "<?php echo htmlspecialchars($original); ?>"</p>
    <p><strong>Length:</strong> <?php echo $length; ?> characters</p>
    <p><strong>Uppercase:</strong> <?php echo htmlspecialchars($uppercase); ?></p>
    <p><strong>Lowercase:</strong> <?php echo htmlspecialchars($lowercase); ?></p>
    <p><strong>Reversed:</strong> <?php echo htmlspecialchars($reversed); ?></p>
    <p><strong>Word Count:</strong> <?php echo $word_count; ?> words</p>
    <p><strong>Substring (First 5 Characters):</strong> "<?php echo htmlspecialchars($substring);
     ?>"</p>
    <a href="<?php echo site_url('stringfunctions'); ?>">Try another string</a>
</body>
</html>
```

Practical NO 14:

Create a CodeIgniter PHP program that demonstrates inheritance with an Animal superclass (with properties name and age and a speak() method) and a Dog subclass that overrides speak() to include the dog's name and age.

Steps to Create the Program:

1. **Create the Models**:
    - Navigate to application/models/.
    - Create a file named Animal.php for the Animal superclass.
    - Create a file named Dog.php for the Dog subclass.
2. **Create the Controller**:
    - Navigate to application/controllers/.
    - Create a file named AnimalController.php.
    - Implement methods to demonstrate inheritance.
3. **Create the Views**:
    - Navigate to application/views/.
    - Create a view file named animal_view.php to display the information.
4. **Configure Routes**:
    - Open application/config/routes.php.
    - Add the following route: $route['animal'] = 'animalcontroller/index';
5. **Run the Application**:
    - Start your web server.
    - Open a web browser and navigate to http://localhost/your_project_directory/index.php/animal.
    - View the output demonstrating inheritance.

CodeIgniter Program

1. **Animal Superclass**: Create a model file named Animal.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class Animal {
    protected $name;
    protected $age;
    public function __construct($name, $age) {
        $this->name = $name;
```

```
        $this->age = $age;
    }
    public function speak() {
        return "I am an animal.";
    }
}
?>
```

2. **Dog Subclass**: Create a model file named Dog.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class Dog extends Animal {
    public function speak() {
        return "Woof! My name is {$this->name} and I am {$this->age} years old.";
    }
}
?>
```

3. **Controller**: Create a controller file named AnimalController.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class AnimalController extends CI_Controller {
    public function index() {
        // Create an instance of the Dog subclass
        $dog = new Dog("Buddy", 3);
        $data['message'] = $dog->speak();
        // Load the view
        $this->load->view('animal_view', $data);
    }
}
?>
```

4. **View**: Create a view file named animal_view.php.

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

```
<head>
    <meta charset="UTF-8">
    <title>Animal Inheritance</title>
</head>
<body>
    <h1>Animal Inheritance Demonstration</h1>
    <p><?php echo $message; ?></p>
</body>
</html>
```

Practical NO 15:

Write a PHP Program in CodeIgniter to Create a Car_model class with a constructor to initialize properties like make, model, and year etc

Steps to Create the Program:

1. **Create the Car Model Class**:
    - Navigate to application/models/.
    - Create a file named Car_model.php.
    - Implement the Car_model class with properties and a constructor.
2. **Create the Controller**:
    - Navigate to application/controllers/.
    - Create a file named CarController.php.
    - Implement methods to instantiate the Car_model class and display its properties.
3. **Create the Views**:
    - Navigate to application/views/.
    - Create a view file named car_view.php to display the car details.
4. **Configure Routes**:
    - Open application/config/routes.php.
    - Add the following route:

        $route['car'] = 'carcontroller/index';
5. **Run the Application**:
    - Start your web server.
    - Open a web browser and navigate to http://localhost/your_project_directory/index.php/car.
    - View the output displaying the car's details.

CodeIgniter Program

1. **Car Model Class**: Create a model file named Car_model.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class Car_model {
    public $make;
    public $model;
    public $year;
```

```
    // Constructor to initialize properties
    public function __construct($make, $model, $year) {
        $this->make = $make;
        $this->model = $model;
        $this->year = $year;
    }
}
?>
```

2. **Controller**: Create a controller file named CarController.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class CarController extends CI_Controller {
    public function index() {
        // Create an instance of Car_model
        $car = new Car_model("Toyota", "Camry", 2022);
       // Prepare data for the view
        $data['make'] = $car->make;
        $data['model'] = $car->model;
        $data['year'] = $car->year;
       // Load the view
        $this->load->view('car_view', $data);
    }
}
?>
```

3. **View**: Create a view file named car_view.php.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Car Model Details</title>
</head>
<body>
```

```
    <h1>Car Model Details</h1>
    <p><strong>Make:</strong> <?php echo htmlspecialchars($make); ?></p>
    <p><strong>Model:</strong> <?php echo htmlspecialchars($model); ?></p>
    <p><strong>Year:</strong> <?php echo htmlspecialchars($year); ?></p>
</body>
</html>
```

Practical NO 16:

Write a PHP program in CodeIgniter to design a web page featuring a text box for name input, radio buttons for selecting a contact method (Email or Phone), check boxes for choosing interests (Sports, Music, Reading), and buttons for submitting or resetting the form

Steps to Create the Program:

1. **Create the Controller**:
    - Navigate to application/controllers/.
    - Create a file named UserFormController.php.
    - Implement methods to display the form and handle the form submission.
2. **Create the Views**:
    - Navigate to application/views/.
    - Create a view file named user_form.php for the input form.
    - Create a view file named form_result.php to display the submitted data.
3. **Configure Routes**:
    - Open application/config/routes.php.
    - Add the following route:

      $route['userform'] = 'userformcontroller/index';
4. **Run the Application**:
    - Start your web server.
    - Open a web browser and navigate to http://localhost/your_project_directory/index.php/userform.
    - Interact with the form to see its functionality.

CodeIgniter Program

1. **Controller**: Create a controller file named UserFormController.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class UserFormController extends CI_Controller {
   public function index() {
      $this->load->view('user_form');
   }
   public function submit() {
      // Retrieve input data
      $name = $this->input->post('name');
```

```
        $contact_method = $this->input->post('contact_method');
        $interests = $this->input->post('interests');
        // Prepare data for the view
        $data['name'] = $name;
        $data['contact_method'] = $contact_method;
        $data['interests'] = $interests;
        // Load the result view
        $this->load->view('form_result', $data);
    }
}
?>
```

2. **View for Input Form**: Create a view file named user_form.php.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>User Input Form</title>
</head>
<body>
    <h1>User Input Form</h1>
    <form action="<?php echo site_url('userformcontroller/submit'); ?>" method="post">
        <label for="name">Name:</label>
        <input type="text" name="name" id="name" required><br><br>
        <label>Contact Method:</label><br>
        <input type="radio" name="contact_method" value="Email" required>Email<br>
        <input type="radio" name="contact_method" value="Phone">Phone<br><br>
        <label>Interests:</label><br>
        <input type="checkbox" name="interests[]" value="Sports">Sports<br>
        <input type="checkbox" name="interests[]" value="Music">Music<br>
        <input type="checkbox" name="interests[]" value="Reading">Reading<br><br>
        <input type="submit" value="Submit">
        <input type="reset" value="Reset">
```

```
    </form>
</body>
</html>
```

3. **View for Result**: Create a view file named form_result.php.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Form Submission Result</title>
</head>
<body>
    <h1>Submitted Information</h1>
    <p><strong>Name:</strong> <?php echo htmlspecialchars($name); ?></p>
    <p><strong>Contact Method:</strong> <?php echo htmlspecialchars($contact_method); ?></p>
    <p><strong>Interests:</strong>
        <?php
        if (!empty($interests)) {
            echo implode(", ", $interests);
        } else {
            echo "None";
        }
        ?>
    </p>
    <a href="<?php echo site_url('userformcontroller'); ?>">Go back to form</a>
</body>
</html>
```

Practical NO 17:

Write a simple PHP program in CodeIgniter that demonstrates introspection and serialization. Use a class to create an object, and then showcase how to inspect its properties and methods using PHP's reflection.

Steps to Create the Program:

1. **Create the Class**:
    - Navigate to application/models/.
    - Create a file named Person.php for the class that will be used for introspection.
2. **Create the Controller**:
    - Navigate to application/controllers/.
    - Create a file named ReflectionController.php.
    - Implement methods to create an object, inspect it, and serialize it.
3. **Create the Views**:
    - Navigate to application/views/.
    - Create a view file named reflection_view.php to display introspection results and serialized data.
4. **Configure Routes**:
    - Open application/config/routes.php.
    - Add the following route: $route['reflection'] = 'reflectioncontroller/index';
5. **Run the Application**:
    - Start your web server.
    - Open a web browser and navigate to http://localhost/your_project_directory/index.php/reflection.
    - View the output showcasing introspection and serialization.

CodeIgniter Program

1. **Person Class**: Create a model file named Person.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class Person {
    public $name;
    public $age;
    public function __construct($name, $age) {
        $this->name = $name;
```

```
        $this->age = $age;
    }
    public function greet() {
        return "Hello, my name is " . $this->name;
    }
}
?>
```

2. **Controller**: Create a controller file named ReflectionController.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class ReflectionController extends CI_Controller {
    public function index() {
        // Create an instance of the Person class
        $person = new Person("Alice", 30);

        // Use Reflection to inspect the Person class
        $reflection = new ReflectionClass($person);
        // Get properties and methods
        $properties = $reflection->getProperties();
        $methods = $reflection->getMethods();

        // Serialize the object
        $serialized_data = serialize($person);

        // Prepare data for the view
        $data['properties'] = $properties;
        $data['methods'] = $methods;
        $data['serialized_data'] = $serialized_data;
        // Load the view
        $this->load->view('reflection_view', $data);
    }
}
```

```
?>
```

3. **View**: Create a view file named reflection_view.php.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Introspection and Serialization</title>
</head>
<body>
    <h1>Introspection and Serialization Demo</h1>
    <h2>Properties:</h2>
    <ul>
        <?php foreach ($properties as $property): ?>
            <li><?php echo htmlspecialchars($property->getName()); ?></li>
        <?php endforeach; ?>
    </ul>
    <h2>Methods:</h2>
    <ul>
        <?php foreach ($methods as $method): ?>
            <li><?php echo htmlspecialchars($method->getName()); ?></li>
        <?php endforeach; ?>
    </ul>

    <h2>Serialized Object:</h2>
    <pre><?php echo htmlspecialchars($serialized_data); ?></pre>
</body>
</html>
```

Practical NO 18:

Step 1: Setting Up CodeIgniter

1. **Setup Environment: Extract the files to your web server's root directory (e.g., htdocs for XAMPP).**
2. **Database Configuration: Create a database (e.g., ci_login_system) and set up a user table.**

```
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50) NOT NULL,
    password VARCHAR(255) NOT NULL
);
```

Step 2: Configure CodeIgniter

1. **Database Connection: Open application/config/database.php and set up your database credentials.**

```
$db['default'] = array(
    'dsn'   => '',
    'hostname' => 'localhost',
    'username' => 'your_username',
    'password' => 'your_password',
    'database' => 'ci_login_system',
    'dbdriver' => 'mysqli',
    ...
);
```

2. **Session Configuration: In application/config/config.php, ensure session settings are configured.**

```
$config['sess_driver'] = 'files'; // Session storage
$config['sess_cookie_name'] = 'ci_session';
$config['sess_expiration'] = 7200; // 2 hours
```

Step 3: Create the User Model

Create a model named User_model.php in application/models/.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class User_model extends CI_Model {
```

```
    public function register($data) {
        return $this->db->insert('users', $data);
    }

    public function login($username, $password) {
        $this->db->where('username', $username);
        $query = $this->db->get('users');

        if ($query->num_rows() == 1) {
            $user = $query->row();
            if (password_verify($password, $user->password)) {
                return $user;
            }
        }
        return false;
    }
}
```

Step 4: Create the User Controller

Create a controller named User.php in application/controllers/.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class User extends CI_Controller {
    public function __construct() {
        parent::__construct();
        $this->load->model('User_model');
        $this->load->library('session');
    }

    public function register() {
        // Load the registration view
        $this->load->view('register');
    }
```

```
public function register_user() {
    $data = [
        'username' => $this->input->post('username'),
        'password' => password_hash($this->input->post('password'), PASSWORD_BCRYPT)
    ];
    $this->User_model->register($data);
    redirect('user/login');
}

public function login() {
    // Load the login view
    $this->load->view('login');
}

public function login_user() {
    $username = $this->input->post('username');
    $password = $this->input->post('password');

    $user = $this->User_model->login($username, $password);

    if ($user) {
        $this->session->set_userdata('user_id', $user->id);
        $this->session->set_userdata('username', $user->username);
        redirect('user/dashboard');
    } else {
        $this->session->set_flashdata('error', 'Invalid login credentials');
        redirect('user/login');
    }
}

public function dashboard() {
    if (!$this->session->userdata('user_id')) {
        redirect('user/login');
```

```
        }
        $this->load->view('dashboard');
    }
    public function logout() {
        $this->session->sess_destroy();
        redirect('user/login');
    }
}
```

Step 5: Create Views

1. Login View (application/views/login.php):

```
<h2>Login</h2>
<?php echo $this->session->flashdata('error'); ?>
<form method="post" action="<?php echo site_url('user/login_user'); ?>">
    <input type="text" name="username" placeholder="Username" required>
    <input type="password" name="password" placeholder="Password" required>
    <button type="submit">Login</button>
</form>
<a href="<?php echo site_url('user/register'); ?>">Register</a>
```

2. Registration View (application/views/register.php):

```
<h2>Register</h2>
<form method="post" action="<?php echo site_url('user/register_user'); ?>">
    <input type="text" name="username" placeholder="Username" required>
    <input type="password" name="password" placeholder="Password" required>
    <button type="submit">Register</button>
</form>
<a href="<?php echo site_url('user/login'); ?>">Login</a>
```

3. Dashboard View (application/views/dashboard.php):

```
<h2>Welcome, <?php echo $this->session->userdata('username'); ?>!</h2>
<a href="<?php echo site_url('user/logout'); ?>">Logout</a>
```

Step 6: Enable Cookies (Optional)

To set a cookie after login, you can modify the login_user function:

```
if ($user) {
```

```
    $this->session->set_userdata('user_id', $user->id);
    $this->session->set_userdata('username', $user->username);
    // Set a cookie
    $this->input->set_cookie('username', $user->username, '86400'); // 1 day
    redirect('user/dashboard');
}
```

Step 7: Test the Application

1. Run your application: Open your browser and navigate to http://localhost/your_project/index.php/user/login.
2. Register a user: Fill out the registration form and submit.
3. Login: Use the registered credentials to log in.
4. Access the dashboard: Verify the session management and cookie handling.

Practical NO 19:

Write a PHP program in CodeIgniter to perform the following tasks: a) Create a form to enter user information (name and email) and save this data into a database. b) Retrieve and display the saved user information in a table format on a separate page.

Steps to Create the Program:

1. **Setup CodeIgniter**:
    - Download CodeIgniter from the official website and extract it into your web server's root directory.
2. **Create the Database**:
    - Open your database management tool (e.g., phpMyAdmin).
    - Create a new database named user_info_db.
    - Run the following SQL command to create a table for storing user information:

```
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    email VARCHAR(100) NOT NULL
);
```

3. **Configure Database Connection**:

- Open application/config/database.php.
- Set the database connection settings to match your environment:

```
$db['default'] = array(
    'dsn'   => '',
    'hostname' => 'localhost',
    'username' => 'your_username',
    'password' => 'your_password',
    'database' => 'user_info_db',
    'dbdriver' => 'mysqli',
    // Other settings...
);
```

4. **Create the Model**:

- Navigate to application/models/.
- Create a file named User_model.php.

```
<?php
```

```
defined('BASEPATH') OR exit('No direct script access allowed');
class User_model extends CI_Model {
    public function save_user($data) {
        return $this->db->insert('users', $data);
    }
    public function get_users() {
        return $this->db->get('users')->result();
    }
}
?>
```

5. **Create the Controller**:

- Navigate to application/controllers/.
- Create a file named UserController.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class UserController extends CI_Controller {
    public function __construct() {
        parent::__construct();
        $this->load->model('User_model');
    }
    public function index() {
        $this->load->view('user_form');
    }

    public function save() {
        $data = array(
            'name' => $this->input->post('name'),
            'email' => $this->input->post('email')
        );
        $this->User_model->save_user($data);
        redirect('usercontroller/display');
    }
```

```
    public function display() {
        $data['users'] = $this->User_model->get_users();
        $this->load->view('user_list', $data);
    }
}
?>
```

6. **Create the Views**:

- Navigate to application/views/.
- Create a view file named user_form.php.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>User Information Form</title>
</head>
<body>
    <h1>User Information Form</h1>
    <form action="<?php echo site_url('usercontroller/save'); ?>" method="post">
        <label for="name">Name:</label>
        <input type="text" name="name" id="name" required><br><br>
        <label for="email">Email:</label>
        <input type="email" name="email" id="email" required><br><br>
        <input type="submit" value="Submit">
    </form>
</body>
</html>
```

- **Create another view file named user_list.php.**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>User List</title>
</head>
```

```
<body>
    <h1>Saved User Information</h1>
    <table border="1">
        <tr>
            <th>ID</th>
            <th>Name</th>
            <th>Email</th>
        </tr>
        <?php foreach ($users as $user): ?>
        <tr>
            <td><?php echo htmlspecialchars($user->id); ?></td>
            <td><?php echo htmlspecialchars($user->name); ?></td>
            <td><?php echo htmlspecialchars($user->email); ?></td>
        </tr>
        <?php endforeach; ?>
    </table>
    <a href="<?php echo site_url('usercontroller'); ?>">Add another user</a>
</body>
</html>
```

7. **Configure Routes**:
   - Open application/config/routes.php.
   - Add the following routes:

```
$route['user'] = 'usercontroller/index';
$route['usercontroller/save'] = 'usercontroller/save';
$route['usercontroller/display'] = 'usercontroller/display';
```

8. **Run the Application**:
   - Start your web server.
   - Open a web browser and navigate to http://localhost/your_project_directory/index.php/user.
   - Fill out the form to submit user information, and then view the saved data on the display page.

Practical NO 20:

Write a PHP program in CodeIgniter to develop a simple application that allows users to Update existing records by modifying user information (e.g., name and email).

Steps to Create the Program:

1. **Setup CodeIgniter**:
   - Download CodeIgniter from the official website and extract it into your web server's root directory.
2. **Create the Database**:
   - Open your database management tool (e.g., phpMyAdmin).
   - Use the existing database user_info_db created in the previous lab manual.
   - Ensure the users table is available. If not, create it with the following SQL command:

```
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    email VARCHAR(100) NOT NULL
);
```

3. **Configure Database Connection**:
   - Open application/config/database.php.
   - Set the database connection settings to match your environment.
4. **Create the Model**:
   - Navigate to application/models/.
   - Open or create a file named User_model.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class User_model extends CI_Model {
    public function save_user($data) {
        return $this->db->insert('users', $data);
    }
    public function get_users() {
        return $this->db->get('users')->result();
    }

    public function get_user($id) {
```

```
        return $this->db->get_where('users', ['id' => $id])->row();
    }

    public function update_user($id, $data) {
        $this->db->where('id', $id);
        return $this->db->update('users', $data);
    }
}
?>
```

5. **Create the Controller**:
    - Navigate to application/controllers/.
    - Create or open a file named UserController.php.

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class UserController extends CI_Controller {
    public function __construct() {
        parent::__construct();
        $this->load->model('User_model');
    }

    public function index() {
        $data['users'] = $this->User_model->get_users();
        $this->load->view('user_list', $data);
    }
    public function edit($id) {
        $data['user'] = $this->User_model->get_user($id);
        $this->load->view('user_edit', $data);
    }
    public function update($id) {
        $data = array(
            'name' => $this->input->post('name'),
            'email' => $this->input->post('email')
```

```
        );
        $this->User_model->update_user($id, $data);
        redirect('usercontroller');
    }
}
?>
```

6. **Create the Views**:
    - Navigate to application/views/.
    - Create a view file named user_list.php to display the list of users.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>User List</title>
</head>
<body>
    <h1>User List</h1>
    <table border="1">
        <tr>
            <th>ID</th>
            <th>Name</th>
            <th>Email</th>
            <th>Actions</th>
        </tr>
        <?php foreach ($users as $user): ?>
        <tr>
            <td><?php echo htmlspecialchars($user->id); ?></td>
            <td><?php echo htmlspecialchars($user->name); ?></td>
            <td><?php echo htmlspecialchars($user->email); ?></td>
            <td>
                <a href="<?php echo site_url('usercontroller/edit/' . $user->id); ?>">Edit</a>
            </td>
```

```
        </tr>
        <?php endforeach; ?>
    </table>
    <a href="<?php echo site_url('usercontroller/add'); ?>">Add New User</a>
</body>
</html>
```

- **Create another view file named user_edit.php for the edit form.**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Edit User</title>
</head>
<body>
    <h1>Edit User Information</h1>
    <form action="<?php echo site_url('usercontroller/update/' . $user->id); ?>" method="post">
        <label for="name">Name:</label>
        <input type="text" name="name" id="name" value="<?php echo htmlspecialchars($user->name); ?>" required><br><br>
        <label for="email">Email:</label>
        <input type="email" name="email" id="email" value="<?php echo htmlspecialchars($user->email); ?>" required><br><br>
        <input type="submit" value="Update">
        <a href="<?php echo site_url('usercontroller'); ?>">Cancel</a>
    </form>
</body>
</html>
```

7. **Configure Routes**:
    - Open application/config/routes.php.
    - Add the following routes:

```
$route['usercontroller'] = 'usercontroller/index';
$route['usercontroller/edit/(:num)'] = 'usercontroller/edit/$1';
$route['usercontroller/update/(:num)'] = 'usercontroller/update/$1';
```

8. **Run the Application**:
    - Start your web server.
    - Open a web browser and navigate to http://localhost/your_project_directory/index.php/usercontroller.
    - View the list of users, click on "Edit" to modify user information.

******