SURANA COLLEGE AUTONOMOUS

WEB PROGRAMMING LAB MANUAL

2024-25

VIDYA A, CS DEPARTMENT

WEB PROGRAMMING PRACTICAL EXERCISES

1) Design a form to input fields for a book table like ISBN number, Book title, Book price and number of copies. Write a PHP script to validate the form.
2) Write a PHP script to set a session timeout after 30 minutes of inactivity.
3) Design a form to input student name and Phone number. Write a PHP script to create a cookie with name as username with student name as value that automatically expires after 60 minutes.
4) Design a PHP program to validate username and password using SQL database.
5) Write a program to demonstrate XMLHttpRequest() properties and methods.
6) Write JavaScript programs to demonstrate array destructuring and arrow functions.
7) Write a program using React to create a form with a textbox and a Submit button.
8) Write a program using React to create a counter.
9) Write a program using React to create TODO list.
10) Write a program to dynamically change colours using React.

STEPS TO RUN PHP PROGRAMS ON XAMPP SERVER

1. Download and Install XAMPP

1. **Download XAMPP**:
    - Go to the official XAMPP website.
    - Download the version compatible with your operating system.
2. **Install XAMPP**:
    - Run the installer and follow the installation wizard.
    - Choose the required components (ensure "Apache" and "PHP" are selected).
    - Complete the installation.

---

2. Start XAMPP and Apache Server

1. Open the XAMPP Control Panel (available in the installation directory).
2. Start the **Apache** server by clicking the "Start" button next to it.
3. If Apache starts successfully, the status will turn green.

---

3. Place Your PHP Files

1. Navigate to the htdocs directory:
    - The htdocs folder is located inside the XAMPP installation directory (e.g., C:\xampp\htdocs on Windows).
2. Create a project folder:
    - For example, create a folder named myphpapp inside htdocs.
3. Add your PHP files to this folder:
    - For example, C:\xampp\htdocs\myphpapp\index.php.

---

4. Write a PHP Program

1. Open a text editor (e.g., Notepad++ or VS Code).
2. Write a simple PHP script:
```
<?php
echo "Hello, World!";
?>
```
3. Save the file as index.php in your project folder (myphpapp).

5. Access Your PHP Program in a Browser

1. Open a web browser.
2. Type the following URL:
   http://localhost/myphpapp/index.php

1. DESIGN A FORM TO INPUT FIELDS FOR A BOOK TABLE LIKE ISBN NUMBER, BOOK TITLE, BOOK PRICE AND NUMBER OF COPIES. WRITE A PHP SCRIPT TO VALIDATE THE FORM.

HTML PROGRAM

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Book Form</title>
</head>
<body>
  <form method="POST" action="lab1.php">
    <table>
      <tr>
        <td>Book ID</td>
        <td><input type="text" name="bookID" /></td></tr>
      <tr>
        <td>Book Title</td>
        <td><input type="text" name="bookTitle" /></td></tr>
      <tr>
        <td>Book Price</td>
        <td><input type="text" name="bookPrice" /></td></tr>
      <tr>
        <td>No. of Copies </td>
        <td><input type="text" name="bookCopies" /></td></tr>
      <tr>
          <td align="center"><input type="submit" name="submit" value="Submit" />
</td>
          <td align="center"><input type="reset" name="reset" value="Reset" /></td>
        </tr>
    </table>
  </form>
</body>
</html>
```

PHP PROGRAM

```
<?php
//Lab Exercise - 01 - PHP Script
//PHP Program to validate form data
   $pattern="/[0-9]/";
   $bname=$_POST["bookID"];
   $btitle=$_POST["bookTitle"];
   $bprice=$_POST["bookPrice"];
   $bcopies=$_POST["bookCopies"];
   if(empty($bname)||empty($btitle)||empty($bprice)||empty($bcopies))
      echo "Fields cannot be empty";
   elseif(!preg_match($pattern,$bprice))
   echo "Price cannot be anything else than digits";
   elseif($bprice<=0 ||$bcopies<=0)
   echo "Price and Copies cannot be negative";
else
echo "Form data is validated";
?>
```

2. WRITE A PHP SCRIPT TO SET A SESSION TIMEOUT AFTER 30 MINUTES OF INACTIVITY.

```php
<?php

//LAB EXERCISE 02
/* Write a PHP script to set a session timeout
after 30 minutes of inactivity.*/

session_save_path('c:/xampp/htdocs/session');
session_start();

// Set the session timeout duration in seconds
$sessionTimeout = 30; // 30 minutes

// Check if the session has already been started and calculate the time since the last activity
if (isset($_SESSION['LAST_ACTIVITY'])) {
   $lastActivity = $_SESSION['LAST_ACTIVITY'];
   $currentTime = time();
   $timeSinceLastActivity = $currentTime - $lastActivity;

   // Check if the session has exceeded the timeout duration
   if ($timeSinceLastActivity > $sessionTimeout) {
      // Session expired, destroy the session
      session_unset();
      session_destroy();
      echo "Session expired. Please log in again.";
   } else {
      // Update the last activity time
      $_SESSION['LAST_ACTIVITY'] = $currentTime;
      echo "Session active.";
   }
} else {
   // Set the last activity time for the session
   $_SESSION['LAST_ACTIVITY'] = time();
   echo "Session started.";
}
?>
```

3. DESIGN A FORM TO INPUT STUDENT NAME AND PHONE NUMBER. WRITE A PHP SCRIPT TO CREATE A COOKIE WITH NAME AS USERNAME WITH STUDENT NAME AS VALUE THAT AUTOMATICALLY EXPIRES AFTER 60 MINUTES.

```
<!DOCTYPE html>
<?php
//LAB EXERCISE 03
/* Design a form to input student name and Phone number.
Write a PHP script to create a cookie with name as username with
student name as value that automatically expires after 60 minutes.*/

$cookie_name = "bca_5";
$cookie_value = "bca_student";
setcookie($cookie_name, $cookie_value, time() + 1800, "/"); // 3600 is 60 minutes

if(!isset($_COOKIE[$cookie_name])) {
   echo "Cookie named '" . $cookie_name . "' is not set!";
} else {
   echo "Cookie '" . $cookie_name . "' is set!<br>";
   echo "Value is: " . $_COOKIE[$cookie_name];
}
?>
```

STEPS TO CREATE TABLE IN MySQL

I. Start XAMPP and Open phpMyAdmin

1. Open the **XAMPP Control Panel**.
2. Start the **Apache** and **MySQL** services by clicking the "Start" buttons next to them.
3. Open a web browser and go to: http://localhost/phpmyadmin

II. Create a New Database

1. In phpMyAdmin, click on the **Databases** tab in the top menu.
2. Under the **Create database** section:
   - Enter a name for your database as STUDENT_DATABASE
   - Click the **Create** button.

III. Create a Table in the Database

1. After creating the database, you'll be redirected to the database's structure page.
2. In the **Create table** section:
   - Enter a name for the table STUDENT_LOGIN
   - Specify the number of columns - 2
   - Click the **Go** button.
3. Define the table structure:
   - For each column, specify:
     - **Name**: username and password
     - **Type**: VARCHAR
   - Click the **Save** button.

---

4. Example Table Structure

For STUDENT_LOGIN table with two columns:

Field Name	Type	Length/Values	Attributes	Index	A_I
name	VARCHAR	255			No
password	VARCHAR	255			No

IV. Insert Data into the Table

1. Navigate to the **Insert** tab in your table (e.g., users).
2. Fill in the fields with sample data:
3. Click the **Go** button to insert the data.

4. DESIGN A PHP PROGRAM TO VALIDATE USERNAME AND PASSWORD USING SQL DATABASE.

HTML PROGRAM

```
<!doctype html>
<html>

<head>
  <title>Login Page</title>
  <script>
  </script>
</head>

<body>
  <form method="post" action="Lab4_LoginPage.php">
    <label>Enter Username</label><br />
    <input type="text" name="username" required /><br />
    <label>Enter Password</label><br />
    <input type="password" name="password" required /><br />
    <input type="submit" name="submit" value="Submit"/>
    <input type="reset" name="reset" value="Reset" /> </form>
  </form>
</body>
```

PHP PROGRAM

```
<?php
if (isset($_POST['submit'])) {
  $username = $_POST['username'];
  $password = $_POST['password'];
  $conn = new mysqli("localhost", "root", "", "student_database");
  $sql = "select * from student_login where username='" . $username . "' and password = '" .
$password . "'";
  $result = mysqli_query($conn, $sql);
  if (mysqli_num_rows($result) > 0) {
    echo "Login Successful";
  } else
    echo "Username and password does not exist";
}
?>
```

5. WRITE A PROGRAM TO DEMONSTRATE XMLHTTPREQUEST() PROPERTIES AND METHODS.

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

<div id="demo">
<h1>The XMLHttpRequest Object</h1>
<button type="button" onclick="loadDoc()">Change Content</button>
</div>

<script>
function loadDoc() {
  var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
if (this.readyState == 4 && this.status == 200) {
     document.getElementById("demo").innerHTML =
this.responseText;
   }
  };
  xhttp.open("GET", "ajaxinfo.txt", true);
  xhttp.send();
}
</script>

</body>
</html>
```

STEPS TO EXECUTE THE PROGRAM

1. Type the above program in VS-CODE and save the file with HTML extension.
2. Create a text file in the same folder as the program and name it as **ajaxinfo.txt**
3. **Execute the program on XAMPP server.**

6. WRITE JAVASCRIPT PROGRAMS TO DEMONSTRATE ARRAY DESTRUCTURING AND ARROW FUNCTIONS.

```
//program to demonstrate array destructuring and arrow functions

//array destructuring
let colors = ['red', 'green', 'blue', 'violet', 'black'];
let [value1, value2,,, value3] = colors;  //array destructuring using []
console.log('Colors after destructuring');
console.log(value1, value2, value3);

//object destructuring
let obj = {
   id: 123,
   sname: 'ABC',
   percent: 94
};
let { id, sname, percent } = obj;   //object destructuring using {}
console.log('Object after destructuring');
console.log(id, sname, percent);

//arrow function
//function to calculate factorial

/* let fact=(n)=>{
   let f=1;
   for (let i = 2; i <= n; i++) {
      f *= i;
   }
   return f;
};
 */

let fact = (n) => (n == 0 ? 1 : n * fact(n - 1));
console.log(`Factorial of 5 is ${fact(4)}`);
```

STEPS TO CREATE A REACT APP USING CREATE-REACT-APP TOOL

1. Set Up Your Development Environment

Ensure you have the following installed on your system:

- **Node.js**: Download and install it from Node.js official site. It comes with npm (Node Package Manager).
- **npm** or **yarn**: These are package managers. npm is installed automatically with Node.js, but you can install yarn as an alternative.

➢ To verify installation:

```
node -v
npm -v
```

2. Install create-react-app (Optional)

From npm version 5.2+, you can directly use npx, so you don't need to globally install create-react-app.

3. Create a New React App

Using npx

```
npx create-react-app my-app
```

Replace my-app with the desired name for your project. This command will create a new directory called my-app with all the boilerplate code and dependencies needed for a React app.

4. Start Coding

Open the project in Visual Studio Code:

The main files to begin editing are in the src folder:

- **src/App.js: Main component.**
- **src/index.js: Entry point of the application**.

5. To run the application:

a) Navigate to the App Directory

After the project is created, navigate to the directory:

```
cd my-app
```

b) Start the Development Server

Start the app in development mode:

```
npm start
```

7. WRITE A PROGRAM USING REACT TO CREATE A FORM WITH A TEXTBOX AND A SUBMIT BUTTON.

Type the following code in App.js of Src folder

```
import { useState } from "react";
import ReactDOM from 'react-dom/client';

function MyForm() {
        const [name, setName] = useState("");
        const handleSubmit = (event) => {
          event.preventDefault();
          alert(`The name you entered was: ${name}`);
  }

  return (
        <form onSubmit={handleSubmit}>
          <label>Enter your name:
                <input
                      type="text"
                       value={name}
                       onChange={(e) => setName(e.target.value)}
                   />
             </label>
          <input type="submit" />
        </form>
  )
}
```

Type the following code in index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import MyForm from './App';


const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(<MyForm/>);
```

8. WRITE A PROGRAM USING REACT TO CREATE A COUNTER.

Type the following code in App.js of Src folder

```
import React, { useState } from 'react';

const Counter = () => {
        const [count, setCount] = useState(0);
        const increment = () => setCount(prevCount => prevCount + 1);
        const decrement = () => setCount(prevCount => prevCount - 1);
        const reset = () => setCount(0);
  return (
         <div>
          <h2>Count: {count}</h2>
          <button onClick={increment}>Increment</button>
          <button onClick={decrement}>Decrement</button>
          <button onClick={reset}>Reset</button>
         </div>
  );
};

const App = () => {
           return (
            <div><h1>Simple Counter App</h1>
             <Counter />
            </div>
           );
};

export default App;
```

Type the following code in index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import App from './App';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(<App/>);
```

9. WRITE A PROGRAM TO DYNAMICALLY CHANGE COLOURS USING REACT

Type the following code in App.js of Src folder

```
import React, { useState } from 'react';

function App() {

  const [color, setColor] = useState('black');
  const [index, setIndex] = useState(0);

  // Function to change the color randomly
  const changeColor = () => {
    const colors = ['red', 'blue', 'green', 'purple', 'orange', 'pink', 'brown'];
    const newColor = colors[index];
    setColor(newColor);
    setIndex((prevIndex) => (prevIndex + 1) % colors.length);
  };
  return (
    <div>
      <h1>Change Paragraph Color</h1>
      <p style={{ color: color, fontSize: '20px' }}>
        Click the button to change my color! </p>
      <button onClick={changeColor} > Change Color</button>
    </div>
  );
}
export default App;
```

Type the following code in index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(<App />);
```

10. WRITE A PROGRAM USING REACT TO CREATE TODO LIST.

Type the following code in App.js of Src folder

```
import React from 'react'
import { useState } from 'react'

function TodoList () {
            const [todos, setTodos] = useState([])
            const [inputValue, setInputValue] = useState('')

function handleChange(e){
            setInputValue(e.target.value)
}
function handleSubmit(e){
            e.preventDefault()
            setTodos([...todos, inputValue])
            setInputValue('')
}
  return (
           <div> <h1>Todo List</h1>
             <form>
               <input type='text' value={inputValue} onChange={handleChange}/>
               <button onClick={handleSubmit}>Add Todo</button>
             </form>
             <ul>
               {todos.map((todo) => (
                 <li key={todo}>{todo}
                  <button>Delete</button>
                 </li>
      ))}
             </ul> </div>
  )
}
export default TodoList
```

Type the following code in index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import TodoList from './App';
const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(<TodoList />);
```