Lab 2: BMI Calculator
Course: Web Development
Level: 2nd Year of LMD in Computer Science

1 Introduction

In this lab you will create a simple BMI (Body Mass Index) Calculator web application. The BMI is calculated as:

$$BMI = \frac{Weight(kg)}{Height(m)^2}$$

The interpretation of BMI is as follows:

- **Underweight**: BMI < 18.5
- **Normal weight**: $18.5 \leq$ BMI < 25
- **Overweight**: $25 \leq$ BMI < 30
- **Obesity**: BMI ≥ 30

This lab is divided into several steps:

1. **Step 1: Separate Files (Without jQuery)**
   Create separate files for the front-end and back-end. **Front-end:** An HTML form (fields: name, weight in kg, height in m), CSS, and a separate JavaScript file for validation (all fields required and numeric check). **Back-end:** A PHP script that calculates the result and echoes the name, BMI value, and an interpretation.
2. **Step 2: Merged File**
   Regroup the HTML and PHP codes into one page so that the form always appears.
3. **Step 3: Introduce jQuery and Bootstrap**
   Enhance the application using jQuery for validation and AJAX submission, and Bootstrap to change the class of the message result (using alert classes such as warning, info, danger, etc.).
4. **Step 4: Homework**
   Extend the project with additional features (see details below).

2 Step 1: Separate Files (Without jQuery)

In this step, create the following files.

Front-end: `index.html`

This file contains an HTML form. Note that the JavaScript code is now placed in a separate file (`script.js`).

Listing 1: index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>BMI Calculator</title>
  <link rel="stylesheet" href="style.css">
  <script src="script.js"></script>
</head>
<body>
  <h1>BMI Calculator</h1>
  <form action="calculate.php" method="post" onsubmit="return validateForm();">
    <label for="name">Name:</label>
    <input type="text" id="name" name="name" required><br>
    <label for="weight">Weight (kg):</label>
    <input type="number" id="weight" name="weight" required><br>
    <label for="height">Height (m):</label>
    <input type="number" id="height" name="height" required><br>
    <input type="submit" value="Calculate">
  </form>
</body>
</html>
```

JavaScript: `script.js` (Vanilla JS for Step 1)

This file contains the simple validation logic.

Listing 2: script.js (Step 1)

```
function validateForm() {
  var name = document.getElementById(’name’).value;
  var weight = document.getElementById(’weight’).value;
  var height = document.getElementById(’height’).value;
  if(name === "" || weight === "" || height === "") {
    alert("All fields are required.");
    return false;
  }
  if(isNaN(weight) || isNaN(height)) {
    alert("Weight and Height must be numbers.");
    return false;
  }
  return true;
}
```

Style: `style.css`

A simple CSS file for basic styling.

Listing 3: style.css

```
body {
  font-family: Arial, sans-serif;
  margin: 20px;
  background-color: #e9ecef;
}
h1 {
  color: #007BFF;
}
```

Back-end: `calculate.php`

This PHP script calculates the BMI and echoes the name, BMI value, and interpretation.

Listing 4: calculate.php

```php
<?php
if(isset($_POST['name'], $_POST['weight'], $_POST['height'])) {
    $name = htmlspecialchars($_POST['name']);
    $weight = floatval($_POST['weight']);
    $height = floatval($_POST['height']);
    if($weight <= 0 || $height <= 0) {
        echo "Invalid input values.";
        exit;
    }
    $bmi = $weight / ($height * $height);
    if($bmi < 18.5) {
        $interpretation = "Underweight";
    } elseif($bmi < 25) {
        $interpretation = "Normal weight";
    } elseif($bmi < 30) {
        $interpretation = "Overweight";
    } else {
        $interpretation = "Obesity";
    }
    echo "Hello, $name. Your BMI is " . number_format($bmi,2) . " ($interpretation).";
} else {
    echo "Data not received.";
}
?>
```

3 Step 2: Merged HTML and PHP in One File

Now, combine the HTML and PHP into a single file (e.g., `index.php`). The form always appears—even after submission—with the result displayed.

Listing 5: index.php (Combined File)

```php
<?php
$result = "";
if($_SERVER['REQUEST_METHOD'] == 'POST') {
    $name = htmlspecialchars($_POST['name']);
    $weight = floatval($_POST['weight']);
    $height = floatval($_POST['height']);
    if($weight > 0 && $height > 0) {
        $bmi = $weight / ($height * $height);
        if($bmi < 18.5) {
            $interpretation = "Underweight";
        } elseif($bmi < 25) {
            $interpretation = "Normal weight";
        } elseif($bmi < 30) {
            $interpretation = "Overweight";
        } else {
            $interpretation = "Obesity";
        }
        $result = "Hello, $name. Your BMI is " . number_format($bmi,2) . " (
    $interpretation).";
    } else {
        $result = "Invalid input values.";
    }
}
?>
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>BMI Calculator</title>
  <link rel="stylesheet" href="style.css">
  <script src="script.js"></script>
</head>
<body>
  <h1>BMI Calculator</h1>
  <?php if($result != "") { echo "<p>$result</p>"; } ?>
  <form action="" method="post" onsubmit="return validateForm();">
    <label for="name">Name:</label>
    <input type="text" id="name" name="name" required><br>
    <label for="weight">Weight (kg):</label>
    <input type="number" id="weight" name="weight" required><br>
    <label for="height">Height (m):</label>
    <input type="number" id="height" name="height" required><br>
    <input type="submit" value="Calculate">
  </form>
</body>
</html>
```

4 Step 3: Introducing jQuery and Bootstrap

Enhance the application by using jQuery for validation and AJAX submission, and Bootstrap for styling and alert messages.

Front-end: `index.html` with jQuery and Bootstrap

This HTML file now includes Bootstrap and references a separate JavaScript file (`script.js`) that contains jQuery code.

Listing 6: index.html with jQuery and Bootstrap

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>BMI Calculator with jQuery and Bootstrap</title>
  <!-- Bootstrap CSS -->
  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
  <link rel="stylesheet" href="style.css">
  <!-- jQuery -->
  <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
  <!-- Bootstrap JS -->
  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
  <script src="script.js" defer></script>
</head>
<body>
  <div class="container">
    <h1 class="mt-5">BMI Calculator</h1>
    <div id="result" class="mt-3"></div>
    <form id="bmiForm" class="mt-3">
      <div class="form-group">
        <label for="name">Name:</label>
        <input type="text" id="name" name="name" class="form-control" required>
      </div>
      <div class="form-group">
        <label for="weight">Weight (kg):</label>
        <input type="number" id="weight" name="weight" class="form-control" required>
      </div>
      <div class="form-group">
        <label for="height">Height (m):</label>
        <input type="number" id="height" name="height" class="form-control" required>
      </div>
      <button type="submit" class="btn btn-primary">Calculate</button>
    </form>
  </div>
</body>
</html>
```

JavaScript: script.js (jQuery Version for Step 3)

This file now contains the jQuery code for form validation, AJAX submission, and dynamic alert message styling.

Listing 7: script.js (Step 3)

```
$(document).ready(function(){
  $(’#bmiForm’).submit(function(e){
    e.preventDefault();
    // Validate inputs using jQuery
    var name = $(’#name’).val().trim();
    var weight = parseFloat($(’#weight’).val());
    var height = parseFloat($(’#height’).val());
    if(name === "" || isNaN(weight) || isNaN(height) || weight <= 0 || height <= 0) {
      $(’#result’).html(’<div class="alert alert-warning">Please enter valid values in all fields.</div>’);
      return;
    }
    // Send data via AJAX
    $.ajax({
      url: ’calculate.php’,
      type: ’POST’,
      data: { name: name, weight: weight, height: height },
      dataType: ’json’,
      success: function(response) {
        if(response.success) {
          var alertClass = ’alert-info’;
          if(response.bmi < 18.5) {
            alertClass = ’alert-warning’;
          } else if(response.bmi < 25) {
            alertClass = ’alert-success’;
          } else if(response.bmi < 30) {
            alertClass = ’alert-info’;
          } else {
            alertClass = ’alert-danger’;
          }
          $(’#result’).html(’<div class="alert ’ + alertClass + ’">’ + response.message + ’</div>’);
        } else {
          $(’#result’).html(’<div class="alert alert-danger">’ + response.message + ’</div>’);
        }
      },
      error: function() {
        $(’#result’).html(’<div class="alert alert-danger">Server error occurred.</div>’);
      }
    });
  });
});
```

Back-end: `calculate.php` (AJAX Version)

This PHP script now returns a JSON response.

Listing 8: calculate.php (AJAX version)

```php
<?php
header(’Content-Type: application/json’);
if(isset($_POST[’name’], $_POST[’weight’], $_POST[’height’])) {
    $name = htmlspecialchars($_POST[’name’]);
    $weight = floatval($_POST[’weight’]);
    $height = floatval($_POST[’height’]);
    if($weight <= 0 || $height <= 0) {
        echo json_encode([
            ’success’ => false,
            ’message’ => ’Invalid input values. Weight and height must be greater than zero.’
        ]);
        exit;
    }
    $bmi = $weight / ($height * $height);
    if($bmi < 18.5) {
        $interpretation = "Underweight";
    } elseif($bmi < 25) {
        $interpretation = "Normal weight";
    } elseif($bmi < 30) {
        $interpretation = "Overweight";
    } else {
        $interpretation = "Obesity";
    }
    $message = "Hello, $name. Your BMI is " . number_format($bmi,2) . " ($interpretation).";
    echo json_encode([
        'success' => true,
        'bmi' => $bmi,
        'message' => $message
    ]);
    exit;
}
echo json_encode([
    'success' => false,
    'message' => 'Data not received.'
]);
exit;
?>
```

5 Step 4: Homework

For your homework, extend this lab project by:

- Adding a feature to store and display previous BMI calculations.
- Integrating a database to save user data.
- Enhancing the UI with additional Bootstrap components.
- Adding further validations and error handling on both the client and server sides.

Be creative and feel free to add any extra features you find interesting!