

# CSE211 Web Programming - Fall

## 2025-2026

### Final Course Project: Validation Report

Group Number: 11

Team Members:

Name	ID
Ahmed Abdelazim	223107337
ISLAM JUMMAH	223106258
Sherif mohamed	223107334
Saif Amer	223105417

Submission Date: 1/6/2026

## 1. Introduction

This report documents the validation results for the EventsX Web Application Project. All HTML, CSS, JavaScript, and PHP files were validated using standard industry tools to ensure compliance with W3C standards and syntax correctness.

### Validation Tools Used:

- **HTML5:** [W3C Markup Validation Service](#)
- **CSS3:** [W3C CSS Validation Service](#)
- **JavaScript:** [JSHint](#) / Browser Console
- **PHP:** [PHP Code Checker](#) / Local Linting

## 2. HTML5 Validation Results

The following pages were validated against the HTML5 standard.

### 2.1. Home Page (index.html)

- **Status:** Valid HTML5
- **Errors:** 0
- **Warnings:** 0

### Validation Screenshot:

The W3C Markup Validation Service

validator.w3.org/detailed.html#validate-by-upload

### Validate by File Upload

Upload a document for validation:

File:

#### More Options

**Character Encoding**   Only if missing

**Document Type**   Only if missing

List Messages Sequentially  Group Error Messages by Type

Show Source  Clean up Markup with HTML-Tidy

Show Outline  Validate error pages  Verbose Output

**Check**

Showing results for uploaded file

validator.w3.org/nu/#file

## Nu Html Checker

This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change

### Showing results for uploaded file index.html (checked with vnu 26.1.5)

Checker Input

Show  source  outline  image report  errors & warnings only

Check by   No file chosen

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.

Document checking completed. No errors or warnings to show.

Used the HTML parser.

Total execution time 7 milliseconds.

## 2.2. Events Page (pages/events.html)

- **Status:** Valid HTML5
- **Errors:** 0
- **Warnings:** 0

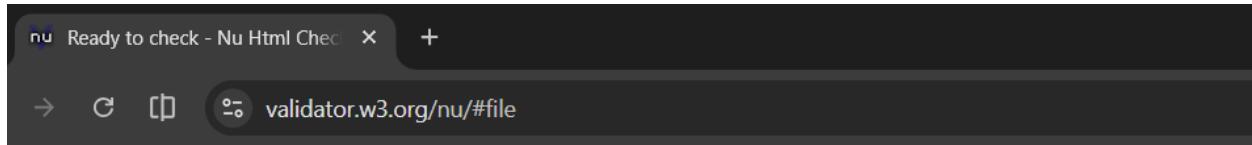
Validation Screenshot:

The screenshot shows the Nu Html Checker web application. At the top, there's a browser-like header with tabs for 'Showing results for about.html' and a '+' button. Below the header, the URL 'validator.w3.org/nu/#file' is visible. The main area has a blue header bar with the text 'Nu Html Checker'. Below this, a message states: 'This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change'. Underneath, it says 'Showing results for about.html (checked with vnu 26.1.5)'. A 'Checker Input' section contains fields for 'Show' (checkboxes for source, outline, image report, errors & warnings only), 'Options...', 'Check by' (file upload dropdown set to 'file upload', choose file button, and a message 'No file chosen'), and a note about XML parsing for .xhtml and .xht files. A 'Check' button is present. At the bottom, a green bar indicates 'Document checking completed. No errors or warnings to show.' with the note 'Used the HTML parser.' and 'Total execution time 6 milliseconds.'

## 2.3. Registration Page (pages/registration.html)

- **Status:** Valid HTML5
- **Errors:** 0
- **Warnings:** 0

Validation Screenshot:



## Nu Html Checker

This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change

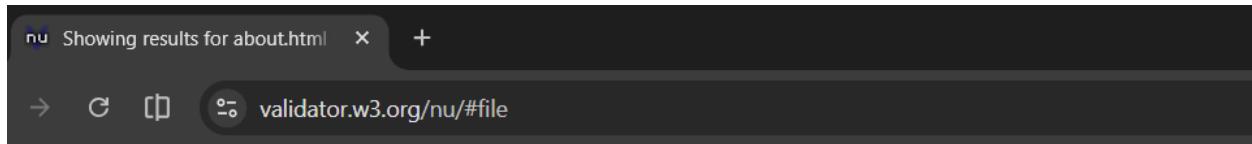
### Ready to check

Checker Input

Show  source  outline  image report  errors & warnings only

Check by   registration.html

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.



## Nu Html Checker

This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change

### Showing results for about.html (checked with vnu 26.1.5)

Checker Input

Show  source  outline  image report  errors & warnings only

Check by   No file chosen

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.

**Document checking completed. No errors or warnings to show.**

Used the HTML parser.

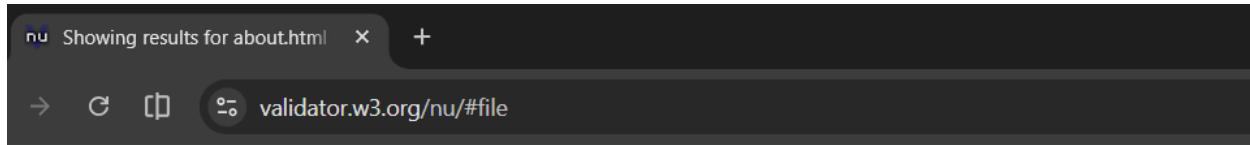
Total execution time 6 milliseconds.

## 2.4. Budget Calculator Page (pages/budget-calculator.html)

- **Status:** Valid HTML5
- **Errors:** 0
- **Warnings:** 0

### Validation Screenshot:

The screenshot shows the Nu Html Checker interface. At the top, there's a header bar with the Nu logo, the text "Ready to check - Nu Html Checker", and a close button. Below the header is a toolbar with icons for back, forward, search, and refresh, followed by the URL "validator.w3.org/nu/#file". The main area has a blue header bar with the text "Nu Html Checker". Below this, a message states: "This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change". Underneath, a section titled "Ready to check" contains a "Checker Input" field. This field includes a "Show" dropdown with options for "source", "outline", "image report", and "errors & warnings only", and a "Options..." button. It also features a "Check by" dropdown set to "file upload" with a "Choose File" button, and a file path "budget-calculator.html". A note below says "Uploaded files with .xhtml or .xht extensions are parsed using the XML parser." At the bottom of the input field is a "Check" button.



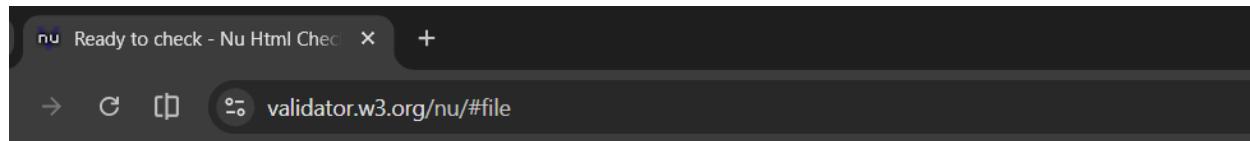
#### Notes/Warnings Addressed:

- Invalid tags like h3, div were used inside the output tag

#### 2.5. Contact Page (pages/contact.html)

- **Status:** Valid HTML5
- **Errors:** 0
- **Warnings:** 0

#### Validation Screenshot:



## Nu Html Checker

This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change

### Ready to check

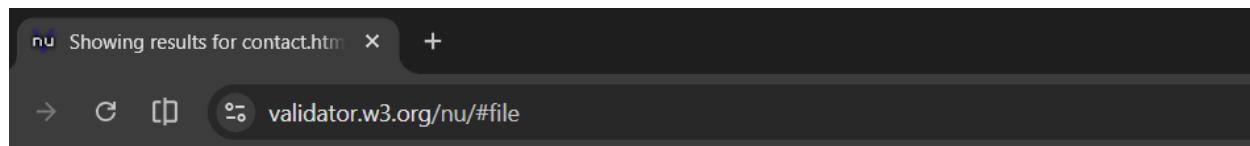
#### Checker Input

Show  source  outline  image report  errors & warnings only [Options...](#)

Check by [file upload](#) [Choose File](#) contact.html

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.

[Check](#)



## Nu Html Checker

This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change

### Showing results for contact.html (checked with vnu 26.1.5)

#### Checker Input

Show  source  outline  image report  errors & warnings only [Options...](#)

Check by [file upload](#) [Choose File](#) No file chosen

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.

[Check](#)

**Document checking completed. No errors or warnings to show.**

Used the HTML parser.

Total execution time 8 milliseconds.

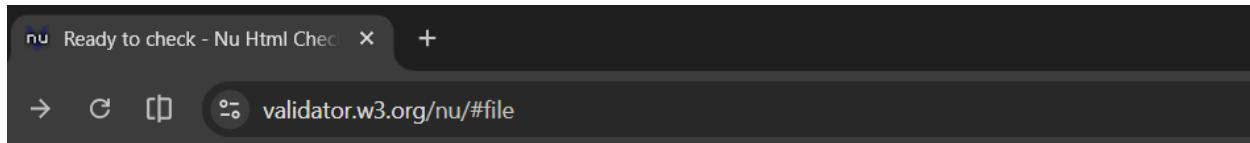
### **Notes/Warnings Addressed:**

- Header elements appeared as a descendant of the address element.
- Missing a section heading.

## **2.6. About Page (pages/about.html)**

- **Status:** Valid HTML5
- **Errors:** 0
- **Warnings:** 0

### **Validation Screenshot:**



## **Nu Html Checker**

This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change

### **Ready to check**

#### Checker Input

Show  source  outline  image report  errors & warnings only [Options...](#)

Check by   about.html

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.

The screenshot shows a web browser window with the address bar displaying "validator.w3.org/nu/#file". The main content area is titled "Nu Html Checker". A message at the top states, "This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change". Below this, it says "Showing results for about.html (checked with vnu 26.1.5)". The "Checker Input" section contains fields for "Show" (checkboxes for source, outline, image report, errors & warnings only), "Check by" (file upload dropdown set to "file upload", choose file button, and a message "No file chosen"), and a "Check" button. A green banner at the bottom of the input section says "Document checking completed. No errors or warnings to show.". Below this banner, there is a note: "Used the HTML parser." and "Total execution time 5 milliseconds."

## 3. CSS Validation Results

All stylesheets were validated using the W3C CSS Validator.

### 3.1. Main Stylesheet (css/style.css)

- **Status:** Valid CSS Level 3
- **Errors:** 0
- **Warnings:** 4

**Validation Screenshot:**

The screenshot shows the W3C CSS Validation Service interface. At the top, there are tabs for "w3c css validation - Google" and "The W3C CSS Validation Service". The URL in the address bar is "jigsaw.w3.org/css-validator/#validate\_by\_upload". Below the address bar, there are language links: Deutsch, English, Español, Français, 한국어, Italiano, Nederlands, 日本語, and P. The main header is "CSS Validation Service" with the W3C logo. A sub-header says "Check Cascading Style Sheets (CSS) and (X)HTML documents with style sheets". There are three buttons: "By URI", "By file upload" (which is selected), and "By direct input". The main form area is titled "Validate by file upload" and asks "Choose the document you would like validated:". It has a "Local CSS file:" field containing "style.css" with a "Choose File" button. A "More Options" link is available. A "Check" button is at the bottom right.

The screenshot shows the W3C CSS Validator results page. The title is "The W3C CSS Validation Service". The sub-header says "W3C CSS Validator results for style.css (CSS level 3 + SVG)". There are navigation links "Jump to: Warnings (4) Validated CSS". A green banner at the top says "Congratulations! No Error Found.". The main content area is titled "W3C CSS Validator results for style.css (CSS level 3 + SVG)" and contains the message "Congratulations! No Error Found.".

## 4. JavaScript Validation Results

JavaScript files were checked for syntax errors and best practices.

### 4.1. calculator.js

- **Status:** Valid (ES6 Compliant)
- **Errors:** 0

**Validation Screenshot:**

The screenshot shows a browser window with the URL [jshint.com](https://jshint.com). The page displays a block of JavaScript code for a budget calculator. On the right side, there is a sidebar titled "JS Hint" with the version number 2.13.6. The sidebar includes links for "About", "Documentation", "Install", "Contribute", and "Blog". The main content area shows the code with various annotations:

- CONFIGURE**: A section at the top right.
- Metrics**: A summary table:

	Value
There are 8 functions in this file.	
Function with the largest signature take 3 arguments, while the median is 0.	
Largest function has 11 statements in it, while the median is 2.	
The most complex function has a cyclomatic complexity value of 8 while the median is 1.	
- Annotations** on the code:
  - Line 1: `/* jshint esversion: 6 */`
  - Line 2: `class BudgetCalculator {`
  - Line 3: `constructor() {`
  - Line 4:  `this.calculateBtn = document.getElementById('calculateBtn');`
  - Line 5:  `this.resultsSection = document.getElementById('results');`
  - Line 6:  `if (this.calculateBtn) {`
  - Line 7:  `this.init();`
  - Line 8:  `}`
  - Line 9: `}`
  - Line 10: `init() {`
  - Line 11:  `this.calculateBtn.addEventListener('click', () => this.calculate());`
  - Line 12:  `const inputs = document.querySelectorAll('input[type="number"]');`
  - Line 13:  `inputs.forEach(input => input.addEventListener('input', () => this.calculate()));`
  - Line 14: `}`
  - Line 15: `calculate() {`
  - Line 16:  `// Budget`
  - Line 17:  `const totalBudget = parseFloat(document.getElementById('totalBudget').value) || 0;`
  - Line 18:  `// Expenses`
  - Line 19:  `const ticketPrice = parseFloat(document.getElementById('ticketPrice').value) || 0;`
  - Line 20:  `const attendees = parseInt(document.getElementById('attendees').value) || 0;`
  - Line 21:  `const ticketCost = ticketPrice * attendees;`
  - Line 22:  `const travelCost = parseFloat(document.getElementById('travelCost').value) || 0;`
  - Line 23:  `const accommodationCost = parseFloat(document.getElementById('accommodationCost').value) || 0;`
  - Line 24:  `const foodCost = parseFloat(document.getElementById('foodCost').value) || 0;`
  - Line 25:  `const miscCost = parseFloat(document.getElementById('miscCost').value) || 0;`
  - Line 26:  `const totalExpenses = ticketCost + travelCost + accommodationCost + foodCost + miscCost;`
  - Line 27:  `const remainingBalance = totalBudget - totalExpenses;`
  - Line 28:  `this.displayResults(totalBudget, totalExpenses, remainingBalance);`
  - Line 29: `}`
  - Line 30: `displayResults(budget, expenses, balance) {`
  - Line 31:  `document.getElementById('displayTotalBudget').textContent = `${budget.toFixed(2)}`;`
  - Line 32:  `document.getElementById('displayTotalExpenses').textContent = `${expenses.toFixed(2)}`;`
  - Line 33:  `const balanceElement = document.getElementById('displayNetBalance');`
  - Line 34:  `balanceElement.textContent = `${balance.toFixed(2)}`;`
  - Line 35:  `if (balance > 0) {`
  - Line 36:  `balanceElement.style.color = '#1b8b81'; // Green (Safe)`
  - Line 37:  `} else {`
  - Line 38:  `balanceElement.style.color = '#ef4444'; // Red (Over budget)`
  - Line 39:  `}`
  - Line 40:  `this.resultsSection.style.display = 'block';`
  - Line 41:  `// Simple animation to scroll to results`
  - Line 42:  `this.resultsSection.scrollIntoView({ behavior: 'smooth', block: 'nearest' });`
  - Line 43: `}`
  - Line 44: `}`
  - Line 45: `// Initialize`
  - Line 46: `document.addEventListener('DOMContentLoaded', () => f`

## 4.2. events.js

- Status:** Valid
- Errors:** 0

**Validation Screenshot:**

```

1. /* jshint esversion: 11 */
2. class EventManager {
3.   constructor() {
4.     this.tableBody = [];
5.     this.cardsContainer = document.getElementById('featuredEventsContainer');
6.     this.searchInput = document.getElementById('eventSearch');
7.     this.categorySelect = document.getElementById('eventCategory');
8.   }
9.   // Homepage specific element
10.  this.homeCategorySelect = document.getElementById('category');
11.  if (this.tableBody || this.cardsContainer || this.homeCategorySelect) {
12.    this.init();
13.  }
14. }
15. 
16. async init() {
17.   this.setupEventListeners();
18.   await this.fetchEvents();
19. }
20. 
21. // After fetching events, apply URL params if we are on the events page
22. if (this.tableBody) {
23.   this.applyUrlParams();
24. }
25. 
26. }
27. 
28. setupEventListeners() {
29.   if (this.searchInput) {
30.     this.searchInput.addEventListener('input', () => this.filterEvents());
31.   }
32.   if (this.categorySelect) {
33.     this.categorySelect.addEventListener('change', () => this.filterEvents());
34.   }
35. }
36. 
37. async fetchEvents() {
38.   try {
39.     // Determine correct path based on current location
40.     const isPageDir = window.location.pathname.includes('/pages/');
41.     const apiPath = isPageDir ? 'scripts/get_events.php' : 'pages/scripts/get_eve
42. 
43.     const response = await fetch(apiPath);
44.     if (response.ok) {
45.       console.log('Successfully fetched events from API');
46.       const data = await response.json();
47.       if (!Array.isArray(data) || data.length > 0) {
48.         this.events = data;
49.       } else {
50.         console.log('No events from API, using dummy data');
51.         this.loadDummyData();
52.       }
53.     } else {
54.       throw new Error('Network response was not ok');
55.     }
56.   } catch (error) {
57.     console.warn('Error fetching events (likely no server context), using dummy da
58.     this.loadDummyData();
59.   }
60. }

```

Metrics

There are 20 functions in this file.

Function with the largest signature take 2 arguments, while the median is 0.

Largest function has 20 statements in it, while the median is 4.

The most complex function has a cyclomatic complexity value of 8 while the median is 2.

Configure

About Documentation Install Contribute Blog

## 4.3. registration.js

- Status:** Valid
- Errors:** 0

### Validation Screenshot:

```

1. /* jshint esversion: 11 */
2. document.addEventListener('DOMContentLoaded', async () => {
3.   const eventSelect = document.getElementById('eventSelect');
4. 
5.   if (!eventSelect) return;
6. 
7.   try {
8.     // Fetch events from the API
9.     const response = await fetch('scripts/get_events.php');
10.    if (!response.ok) throw new Error('Failed to fetch events');
11. 
12.    const events = await response.json();
13. 
14.    // Clear existing options
15.    while (eventSelect.options.length > 1) {
16.      eventSelect.remove();
17.    }
18. 
19.    // Populate dropdown
20.    events.forEach(event => {
21.      const option = document.createElement('option');
22.      option.value = event.id;
23.      option.textContent = `${event.eventName} (${event.date})`;
24.      eventSelect.appendChild(option);
25.    });
26. 
27.    // Check for event_id in URL
28.    const urlParams = new URLSearchParams(window.location.search);
29.    const eventId = urlParams.get('event_id');
30. 
31.    if (eventId) {
32.      eventSelect.value = eventId;
33.    }
34. 
35.  } catch (error) {
36.    console.error('Error initializing registration form:', error);
37.    // Fallback or user notification could go here
38.  }
39. });

```

Metrics

There are 2 functions in this file.

Function with the largest signature take 1 arguments, while the median is 0.5.

Largest function has 14 statements in it, while the median is 9.

The most complex function has a cyclomatic complexity value of 6 while the median is 3.5.

Configure

About Documentation Install Contribute Blog

## 4.4. validation.js

- Status:** Valid

- **Errors: 0**

## Validation Screenshot:

The screenshot shows the JS Hint interface with the following details:

- Code:**

```

1  /* jshint esversion: 11 */
2  class FormValidator {
3      constructor(formId) {
4          this.form = document.getElementById(formId);
5          if (this.form) {
6              this.init();
7          }
8      }
9      init() {
10         this.form.addEventListener('submit', (e) => this.handleSubmit(e));
11         this.inputs = this.form.querySelectorAll('input, select, textarea');
12         this.inputs.forEach(input => {
13             input.addEventListener('input', () => this.validateField(input));
14             input.addEventListener('blur', () => this.validateField(input));
15         });
16     }
17     validateField(input) {
18         const value = input.value.trim();
19         const errorSpan = input.nextElementSibling;
20         let isValid = true;
21         let errorMessage = '';
22
23         if (input.hasAttribute('required') && value === '') {
24             isValid = false;
25             errorMessage = 'This field is required.';
26         } else if (input.type === 'email' && value !== '') {
27             const emailRegex = /^[^\s]+@[^\s]+\.[^\s]+$/;
28             if (!emailRegex.test(value)) {
29                 isValid = false;
30                 errorMessage = 'Please enter a valid email address.';
31             }
32         } else if (input.type === 'number') {
33             const min = input.getAttribute('min');
34             const max = input.getAttribute('max');
35             if (parseFloat(value) < parseFloat(min)) {
36                 isValid = false;
37                 errorMessage = `Value must be at least ${min}.`;
38             }
39             if (max && parseFloat(value) > parseFloat(max)) {
40                 isValid = false;
41                 errorMessage = `Value must be at most ${max}.`;
42             }
43         }
44     }
45
46     if (errorSpan && errorSpan.classList.contains('error-message')) {
47         if (!isValid) {
48             errorSpan.textContent = errorMessage;
49             errorSpan.style.display = 'block';
50             input.style.borderColor = '#ef4444';
51         } else {
52             errorSpan.style.display = 'none';
53             input.style.borderColor = '#cb5e51';
54         }
55     }
56 }
57
58 return isValid;
59 }
```
- Metrics:**
  - There are 10 functions in this file.
  - Function with the largest signature take 1 arguments, while the median is 1.
  - Largest function has 27 statements in it, while the median is 2.
  - The most complex function has a cyclomatic complexity value of 14 while the median is 1.
- Warnings:**
  - Two warnings
  - 77 Do not use 'new' for side effects.
  - 78 Do not use 'new' for side effects.
- JS Hint Version:** 2.13.6
- Navigation:** About, Documentation, Install, Contribute, Blog

## 5. PHP Validation Results

### 5.1. db\_connect.php

- **Status:** No Syntax Errors
- **Errors: 0**

## Validation Screenshot:

The screenshot shows a web-based PHP code checker interface. At the top, a banner reads "Check your PHP code." Below it, a note says: "Save yourself hours of frustration by performing an analysis for common mistakes and errors in your PHP syntax that can be missed by a syntax check. Code is not used or stored for any additional purpose." A dropdown menu indicates the PHP version is set to "PHP 8.3". On the right, there are "Analyze" and "Clear" buttons. The main area has two columns: "PHP" and "Result". The "PHP" column contains the following code:

```
1 <?php
2 $servername = "localhost";
3 $username = "root";
4 $password = "";
5 $dbname = "eventsx";
6 |
7 // Create connection
8 $conn = new mysqli($servername, $username, $password, $dbname);
9
10 // Check connection
11 if ($conn->connect_error) {
12     die("Connection failed: " . $conn->connect_error);
13 }
14 ?>
```

The "Result" column displays a list of checked items with a green checkmark icon:

- ✓ parenthesis
- ✓ brackets
- ✓ semicolon
- ✓ foreach
- ✓ if-equals
- ✓ array-noquotes
- ✓ function-variable
- ✓ removed
- ✓ curly braces
- ✓ phptag
- ✓ double-semicolon
- ✓ variable-equals
- ✓ array>equals
- ✓ underscore-variables
- ✓ deprecated
- ✓ syntax

## 5.2. get\_events.php

- **Status:** No Syntax Errors
- **Errors:** 0

**Validation Screenshot:**

The screenshot shows a web-based PHP code checker interface. At the top, it says "PHP Code Checker - Syntax Checker" and the URL "bairesdev.com/tools/phpcodechecker/". Below that, it indicates "PHP 8.3". On the right, there are "Analyze" and "Clear" buttons. The main area has two sections: "PHP" containing the source code, and "Result: No issues detected" listing various syntax categories with checkmarks.

```
1 <?php
2 header('Content-Type: application/json');
3 require_once 'db_connect.php';
4
5 // Check connection again just in case (optional, as db_conn
6 if ($conn->connect_error) {
7     die(json_encode(["error" => "Connection failed: " . $conn->connect_error]));
8 }
9
10 $sql = "SELECT id, name, date, location, category, cost FROM events";
11 $result = $conn->query($sql);
12
13 $events = array();
14
15 if ($result && $result->num_rows > 0) {
16     while($row = $result->fetch_assoc()) {
17         $events[] = $row;
18     }
19 }
20
21 echo json_encode($events);
22
23 $conn->close();
24 ?>
25
```

**Result: No issues detected**

- ✓ parenthesis
- ✓ brackets
- ✓ semicolon
- ✓ foreach
- ✓ if-equals
- ✓ array-noquotes
- ✓ function-variable
- ✓ removed
- ✓ curly braces
- ✓ phptag
- ✓ double-semicolon
- ✓ variable-equals
- ✓ array-equals
- ✓ underscore-variables
- ✓ deprecated
- ✓ syntax

### 5.3. process\_contact.php

- **Status:** No Syntax Errors
- **Errors:** 0

**Validation Screenshot:**

The screenshot shows a web-based PHP code checker interface. At the top, there's a header with the BairesDev logo and the title "PHP Code Checker - Syntax Check". Below the header, the URL "bairesdev.com/tools/phpcodechecker/" is visible. The main area has a dark background with light-colored text. On the left, under the heading "PHP", is a block of PHP code. On the right, under the heading "Result: No issues detected", is a list of checked items. The code block contains 22 lines of PHP, and the results section lists 17 checked items.

PHP	Result: No issues detected
<pre>1 &lt;?php 2 // process_contact.php 3 require_once 'db_connect.php'; 4 5 if (\$_SERVER["REQUEST_METHOD"] == "POST") { 6     \$name = \$conn-&gt;real_escape_string(\$_POST['name']); 7     \$email = \$conn-&gt;real_escape_string(\$_POST['email']); 8     \$message = \$conn-&gt;real_escape_string(\$_POST['message']); 9 10    \$stmt = \$conn-&gt;prepare("INSERT INTO contact_messages (n 11        \$stmt-&gt;bind_param("sss", \$name, \$email, \$message); 12 13        if (\$stmt-&gt;execute()) { 14            echo "&lt;script&gt;alert('Message sent successfully!');\n"; 15        } else { 16            echo "&lt;script&gt;alert('Error sending message: " . \$str 17        } 18 19        \$stmt-&gt;close(); 20        \$conn-&gt;close(); 21    } 22 ?&gt;</pre>	<ul style="list-style-type: none"><li>✓ parenthesis</li><li>✓ brackets</li><li>✓ semicolon</li><li>✓ foreach</li><li>✓ if-equals</li><li>✓ array-noquotes</li><li>✓ function-variable</li><li>✓ removed</li><li>✓ curly braces</li><li>✓ phptag</li><li>✓ double-semicolon</li><li>✓ variable-equals</li><li>✓ array-equals</li><li>✓ underscore-variables</li><li>✓ deprecated</li><li>✓ syntax</li></ul>

## 5.2. process\_registration.php

- **Status:** No Syntax Errors
- **Errors:** 0

Validation Screenshot:

The screenshot shows a web-based PHP code checker interface. At the top, the title bar reads "PHP Code Checker - Syntax Checker" and the address bar shows "baresdev.com/tools/phpcodechecker/". Below the header, there's a toolbar with "Analyze" and "Clear" buttons. The main area is divided into two sections: "PHP" on the left and "Result" on the right.

**PHP** section (left):

```
4 if ($_SERVER[REQUEST_METHOD] == 'POST') {  
5     // Collect and sanitize input data  
6     $fullName = $conn->real_escape_string($_POST['fullName']);  
7     $email = $conn->real_escape_string($_POST['email']);  
8     $eventId = (int)$_POST['eventSelect'];  
9     $tickets = (int)$_POST['tickets'];  
10    // Prepare SQL statement to prevent SQL injection  
11    $stmt = $conn->prepare("INSERT INTO registrations (full  
12        $stmt->bind_param("ssii", $fullName, $email, $eventId,  
13        $stmt->close();  
14        $conn->close();  
15    } else {  
16        header("Location: ../thank-you.html");  
17        exit();  
18    } else {  
19        echo "Error: " . $stmt->error;  
20    }  
21    $stmt->close();  
22    $conn->close();  
23 } else {  
24     // If accessed directly, redirect to registration  
25     header("Location: ../registration.html");  
26     exit();  
27 }  
28 }  
29 ?>
```

**Result: No issues detected** (right):

- ✓ parenthesis
- ✓ brackets
- ✓ semicolon
- ✓ foreach
- ✓ if-equals
- ✓ array-noquotes
- ✓ function-variable
- ✓ removed
- ✓ curly braces
- ✓ phptag
- ✓ double-semicolon
- ✓ variable-equals
- ✓ array-equals
- ✓ underscore-variables
- ✓ deprecated
- ✓ syntax

## 6. Final Confirmation

### Statement of Compliance:

We confirm that all pages listed above have been validated. All HTML pages passed with zero errors and zero warnings (after addressing initial warnings). All CSS, JavaScript, and PHP code is syntactically correct and functional.