## Context: You are an accessibility expert reviewing HTML for accessibility issues. Analyze the following code in the context of WCAG (Web Content Accessibility Guidelines) 2.2, listing any issues that may violate these guidelines. You will only be analyzing the HTML, ignoring any included or missing styles and scripts.

Code:

```<!DOCTYPE html>

<html lang="eg">

<head>

<meta charset="UTF-8">

<title>Restaurant Menu</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<header>

<nav>

<!-- Navigation bar -->

<ul>

<li><a href="#home">Home</a></li>

<li><a href="#menu">Menu</a></li>

<li><a href="#about">About Us</a></li>

<!-- More navigation links -->

</ul>

</nav>

</header>

<main>

<section id="home">

<!-- Homepage content with background image, text, buttons, and maybe a video -->

<div class="hero">

<h1>Welcome to Our Restaurant</h1>

<font size="5"> Delicious food just for you!</font>

<a href="#menu" class="btn">View Menu</a>

</div>

</section>

<section id="menu">

<!-- Menu items with images, text descriptions, search filters, and sorting filters -->

<h2>Our Menu</h2>

<div class="search-filter">

<!-- Search field and filters -->

<label for="search">Search:</label>

<label for="search">Search:</label>

<input type="text" id="search" placeholder="Search...">

<!-- Sorting options -->

<select id="sort">

<option value="default">Default</option>

<option value="price">Price</option>

<!-- More sorting options -->

</select>

</div>

<div class="menu-items">

<!-- Individual menu item -->

<div class="menu-item" onmouseover="ExternalFunction()">

<img src="food1.jpg" alt="Food 1">

<h3>Food Item Name</h3>

<p>Description of the food item.</p>

<span class="price">$10.99</span>

<!-- Add to cart button -->

<button>Add to Cart</button>

</div>

<!-- More menu items -->

</div>

</section>

<!-- Other sections like About Us, Contact, etc. -->

</main>

<footer>

<!-- Footer content -->

<p>&copy; 2023 Restaurant Name. All rights reserved.</p>

</footer>

<script src="scripts.js"></script>

</body>

</html>

```

Response format:

Snippet with error: …

Location of error: …

WCAG Guideline violated: …

Detection Algorithm: …

Explanation of Error: …

### Snippet with error:

```html

<div class="menu-item" onmouseover="ExternalFunction()">

```

### Location of error:

In the `<section id="menu">` section, within the `<div class="menu-item">` element.

### WCAG Guideline violated:

\*\*2.4.7 Focus Visible (AA)\*\*

### Detection Algorithm:

The algorithm checks for elements that rely solely on visual cues (like color changes) to convey state or function, without providing alternative means for users who might not perceive those changes.

### Explanation of Error:

The `onmouseover` attribute is used to change the appearance of the `.menu-item` div when the mouse hovers over it. This approach relies solely on visual feedback (color change, hover effect) to indicate interactivity, which does not meet the WCAG guideline that requires functional information to be available through senses other than sight. Users relying on screen readers or having low vision would not receive this interactive cue, violating the guideline for focus visibility.

To fix this issue, consider adding keyboard-accessible alternatives or using ARIA roles and properties to enhance accessibility. For example, if the hover effect is purely cosmetic and does not affect functionality, it should be removed or replaced with a keyboard-triggered event that provides equivalent functionality.

Citations:

## Are there any more issues? List all that you find.

Based on the provided code snippet and the WCAG 2.2 AA checklist from the sources, here are additional potential accessibility issues:

### Issue 1: Lack of Semantic HTML

- \*\*WCAG Guideline Violated:\*\* 2.4.6 Headings and Sections (A)

- \*\*Detection Algorithm:\*\* Checks for proper use of heading tags (`<h1>` to `<h6>`) and semantic elements like `<nav>`, `<main>`, `<section>`, and `<article>` to structure content logically.

- \*\*Explanation of Error:\*\* The code uses `<section>` elements for different parts of the webpage but lacks explicit `<article>` elements for individual menu items. Each menu item could be wrapped in an `<article>` tag to better define its role as a self-contained composition in a document, page, application, or site, which is crucial for screen reader users to navigate and understand the content structure.

### Issue 2: Missing Alt Text for Background Images

- \*\*WCAG Guideline Violated:\*\* 1.1.1 Non-text Content (A)

- \*\*Detection Algorithm:\*\* Looks for images without appropriate `alt` attributes.

- \*\*Explanation of Error:\*\* The code snippet mentions a "background image" in the homepage section but does not show the actual HTML for it. If there's a background image in the `.hero` class or elsewhere, it should have an `alt` attribute describing the content of the image for users who cannot see it. Since the `alt` attribute was not shown in the provided code, this is a potential issue.

### Issue 3: Use of Font Tag

- \*\*WCAG Guideline Violated:\*\* 2.4.10 Section Headings (AAA)

- \*\*Detection Algorithm:\*\* Identifies misuse of presentational elements like `<font>` for styling text.

- \*\*Explanation of Error:\*\* The `<font>` tag is deprecated and should not be used for styling purposes. Instead, CSS should be used to style text. Using `<font>` tags can lead to accessibility issues because they do not support modern styling capabilities and can interfere with screen readers interpreting the content correctly.

### Issue 4: Inline JavaScript Event Handler

- \*\*WCAG Guideline Violated:\*\* 2.1.1 Keyboard (A)

- \*\*Detection Algorithm:\*\* Searches for inline JavaScript event handlers that prevent full keyboard accessibility.

- \*\*Explanation of Error:\*\* The `onmouseover` attribute in the `.menu-item` div is an inline JavaScript event handler. While not inherently bad, it's generally recommended to avoid inline JavaScript for accessibility reasons, especially for events that can significantly impact usability, like focusing and interacting with interactive elements. Moving such logic to external JavaScript files can improve maintainability and ensure consistent handling across browsers.

### Issue 5: Lack of ARIA Roles for Interactive Elements

- \*\*WCAG Guideline Violated:\*\* 1.3.1 Info and Relationships (A)

- \*\*Detection Algorithm:\*\* Checks for the presence of ARIA roles on interactive elements to ensure proper semantics for assistive technologies.

- \*\*Explanation of Error:\*\* The code snippet does not explicitly include ARIA roles for interactive elements like buttons or links. Adding appropriate ARIA roles (e.g., `role="button"` for buttons) can enhance the experience for users of assistive technologies by providing clear semantics about the purpose and behavior of interactive elements.

### Issue 6: No Keyboard-Accessible Toggle for Hidden Content

- \*\*WCAG Guideline Violated:\*\* 2.4.7 Focus Visible (AA)

- \*\*Detection Algorithm:\*\* Looks for content that is meant to be interactively revealed or hidden (e.g., accordions, tabs) but lacks keyboard-accessible mechanisms to toggle visibility.

- \*\*Explanation of Error:\*\* While not explicitly shown in the provided code, if there are elements intended to hide or reveal content upon interaction (common in menus, FAQs, or settings panels), they should be made accessible via keyboard controls. This ensures that users who cannot use a mouse can still interact with these elements effectively.

These issues highlight areas where the provided HTML code could be improved to better align with WCAG 2.2 AA guidelines, enhancing accessibility for all users.

Citations:

[1] https://adabook.medium.com/wcag-2-2-aa-guide-checklist-for-2021-web-accessibility-66c6fdaea034

[2] https://www.levelaccess.com/blog/wcag-2-2-aa-summary-and-checklist-for-website-owners/

[3] https://www.wuhcag.com/wcag-checklist/

[4] https://www.w3.org/TR/WCAG22/

[5] https://www.digitala11y.com/wcag-checklist/

[6] https://webaim.org/standards/wcag/checklist

[7] https://wcag.com/blog/wcag-2-2-aa-summary-and-checklist-for-website-owners/

[8] https://adasitecompliance.com/wcag-2-2-checklist/

[9] https://www.boia.org/wcag-2.2aa-checklist

[10] https://media.dequeuniversity.com/en/docs/web-accessibility-checklist-wcag-2.2.pdf

## Are there any more issues? List all that you find.

Reviewing the provided HTML code against the WCAG 2.2 AA checklist from the sources, here are additional potential accessibility issues:

### Issue 7: Focus Visibility for Sticky Elements

- \*\*WCAG Guideline Violated:\*\* 2.4.11 Focus Visible (AA)

- \*\*Detection Algorithm:\*\* Checks for elements that become focused but are obscured by other content, such as sticky headers or footers.

- \*\*Explanation of Error:\*\* The code does not explicitly mention sticky elements, but if any part of the website uses sticky positioning (not shown in the provided code), it's crucial to ensure that focused elements remain visible and accessible. Sticky elements can obscure focusable elements, making them difficult to interact with for keyboard users.

### Issue 8: Target Size for Interactive Elements

- \*\*WCAG Guideline Violated:\*\* 2.5.8 Target Size (AA)

- \*\*Detection Algorithm:\*\* Ensures that interactive elements are large enough to interact with, typically at least 24x24 CSS pixels.

- \*\*Explanation of Error:\*\* The code snippet does not specify sizes for interactive elements like buttons or links. Ensuring that these elements are large enough to interact with, especially on touch devices, is crucial for accessibility. Smaller elements can be difficult to select accurately, particularly for users with motor control issues.

### Issue 9: Consistency of Help Options

- \*\*WCAG Guideline Violated:\*\* 3.2.6 Consistent Identification (AA)

- \*\*Detection Algorithm:\*\* Looks for consistency in the placement and identification of help options across different pages.

- \*\*Explanation of Error:\*\* The provided code does not include help options like support links or contact information. However, if such elements exist elsewhere in the website, it's important to ensure they are consistently placed and identified across all pages to aid users in finding assistance.

### Issue 10: Authentication Methods

- \*\*WCAG Guideline Violated:\*\* 3.3.7 Alternative Authentication (AA)

- \*\*Detection Algorithm:\*\* Checks for reliance on memory-based authentication methods alone, without offering alternative, less cognitively demanding methods.

- \*\*Explanation of Error:\*\* The code does not detail the authentication process. It's crucial to offer alternative authentication methods besides those that rely heavily on memory, such as email links or phone verification codes, to accommodate users who may struggle with traditional username/password combinations.

### Issue 11: Auto-Fill for Form Fields

- \*\*WCAG Guideline Violated:\*\* 3.3.8 Input Assistance (AA)

- \*\*Detection Algorithm:\*\* Verifies that previously entered information is available through auto-fill or selection features in forms.

- \*\*Explanation of Error:\*\* The code snippet does not include form fields. However, if forms are present on the website, they should allow users to auto-fill or select previously entered information to streamline the input process and reduce errors.

### Issue 12: Keyboard Shortcut Customization

- \*\*WCAG Guideline Violated:\*\* 2.1.4 Keyboard Operation (AA)

- \*\*Detection Algorithm:\*\* Ensures that keyboard shortcuts can be customized, disabled, or limited to specific contexts.

- \*\*Explanation of Error:\*\* The code does not mention keyboard shortcuts. Implementing customizable or context-specific keyboard shortcuts can enhance accessibility by allowing users to tailor the interface to their preferences or needs.

These issues highlight further areas for improvement to enhance the accessibility of the web content beyond the initial analysis.

Citations:

[1] https://adabook.medium.com/wcag-2-2-aa-guide-checklist-for-2021-web-accessibility-66c6fdaea034

[2] https://www.digitala11y.com/wcag-checklist/

[3] https://webaim.org/standards/wcag/checklist

[4] https://accessible.org/wcag/

[5] https://www.w3.org/TR/WCAG22/

[6] https://www.wuhcag.com/wcag-checklist/

[7] https://media.dequeuniversity.com/en/docs/web-accessibility-checklist-wcag-2.2.pdf

[8] https://www.levelaccess.com/blog/wcag-2-2-aa-summary-and-checklist-for-website-owners/

[9] https://help.blackboard.com/Ally/Ally\_for\_WCM/Teacher\_Editor/Accessibility\_Report/Checklist

[10] https://usability.yale.edu/web-accessibility/articles/wcag2-checklist