**Web Implementation Architect Project Assessment**

**Introduction to Web Implementation Architect Assessment**

Based on our customer’s requirements, SOCi develops Locators and Local Pages based on the given design. This role requires excellent HTML, JS, Web Development skills to develop not only a working webpage, which matches design, but also follows unique JS rules based on the customer’s needs. You will be required to work with the SOCi TAM to develop and execute on project requirements taking into account timelines and quality assurance to meet all given projects requirements.

The following two assessments will help us understand your level of proficiency and quality of the code for HTML / CSS and JS.

**• Design to HTML conversion:** Based on the design pasted below, please create HTML page which will match the mockup as close as possible:

• For the map, use a static image of any MAP. Use Google maps to screen grab any geographical area. Ignore the icons on the map, they do not need to be part of the completed design

• The list with addresses on the left should include 10 locations, same address can be used, the list should have scrolling option

• The search above the list does not need any functionality, however it needs to match the design

• When someone clicks on the filter icon, we would like to see filter presented on the mock up

• For any images seen on the design, pls use zipped document

• Do not use any tools that generate HTML, the HTML/CSS has to be hand-coded by you

• Time limit: 45 min

• The design does not need to be perfect, we want to understand how far it can be taken within 45 min.

MOBILE VIEW:

**2. JavaScript assessment: Please answer the following questions**

**• In the following code snippet can you please predict the output and / or explain the error?**

<!DOCTYPE html>

<html>

<body>

<h2> <> Sample: TEST MY JS</> </h2>

<script>

var studentName = "Margaret S "; // String 'Margaret SE' stored in studentName

var studentName; // variable is decalredd again

document.getElementById("studentName").innerHTML =

"Redeclaring the variable will not lose the value!.<br>"

+"Here the value in studentName is "+ studentName;

</script>

</body>

</html>

**ANSWER:**

**Error because it can not find the element with an Id = “studentName”**

**• In the following code snippet can you please predict the output and / or explain the error?**

<!DOCTYPE html>

<html>

<body>

<h2> <> Sample: TEST MY JS</> </h2>

<p style='text-decoration:underline'>Example Const Variable </p>

<p id="display"></p>

<script>

const first\_num;

first\_num =1000;

document.getElementById("display").innerHTML = "First Number:"+ first\_num;

</script>

</body>

</html>

**ANSWER:**

**It supposed to display this:** First Number:1000

**But since there is an error in initializing a variable, it did not output correctly and return a syntax error.**

**• What browsers are you using for debugging and how ?**

**Possible Answers: I am using Google Chrome/Firefox and using its inspect Elements, Console and other Developer Tools on it for debugging.**

**• What’s the difference between a variable that is: null, undefined or undeclared? How would you go about checking for any of these states?**

**ANSWER:**

**The ‘null’ variable is intentionally set to have no meaningful value while ‘undefined’ variable is declared but not assigned to any value yet and the ‘undeclared’ variable is a variable that has not been defined in the current scope.**

**Question: For this code block:**

**• Describe the basic functionality of this function?**

**• Why is the preventDefault() function necessary here?**

$("#filter\_button\_up").on('keydown', function(e) {

if (e.keyCode == 9 || e.which == 9){

e.preventDefault();

$("#newField").focus();

}

});



**Answer:**

**• When the Tab key (keycode9) is pressed on the ‘filter\_button\_up’ element, the focus is set to the element with the ID ‘newField’**

**• The preventDefault() will stops the default Tab key behavior, ensuring focus control on ‘newField’ instead of moving to the next element in the tab order.**

**Question: What is the purpose of this setTimeout function:**

<input name="addressline" id="inputaddress" placeholder="Enter street address, city, state or zip code" />

<select id="country" name="country" class="search-box">

<option value="CA">CA</option>

<option value="US">US</option>

<option value="PR">Puerto Rico</option>

</select>

<script>

setTimeout(function(){

var searchCountry = document.getElementById('country')

for (var i = searchCountry.length - 1; i >= 0; i--) {

if (countryOptions[i].value === 'US') {

countryOptions[i].textContent = 'USA'

}

}

$('#inputaddress').focus();

}, 800);

</script>



**Answer:**

**The purpose of the ‘setTimeout’ function is to delay changing the text content of the ‘US’ option to ‘USA’ and setting the focus on the ‘input address’ input field by 800 milliseconds or (0.8 seconds)**

**Question: How would you make the following form ADA compliant?**

<form id="contact">

<input type="text" name="name" placeholder="Enter name">

<select name="state">

<option>CA</option>

<option>IL</option>

<option>TX</option>

</select>

<a href="javascript:void(0);">Submit</a>

</form>



**Answers:**

**• To make the form ADA (Americans with Disabilities Act) compliant and improve its accessibility, you can implement the following changes:**

**⁃ Add labels for form elements: Use <label> elements to associate labels with form controls. This helps screen readers identify the purpose of each input field and select element.**

**⁃ Provide a visible focus indication: Ensure that users can easily identify which element has focus, especially for keyboard users. You can add CSS styles to indicate focus using the :focus pseudo-class.**

**⁃ Use a button element for submission: Instead of using an anchor (<a>) element for submission, use a <button> element with appropriate accessibility attributes. This ensures better handling by assistive technologies.**

**⁃ Here is the revised form:**

<form id="contact">

<label for="name">Name:</label>

<input type="text" id="name" name="name" placeholder="Enter name" aria-required="true" />

<label for="state">State:</label>

<select id="state" name="state" aria-required="true">

<option value="">Select a state</option>

<option value="CA">CA</option>

<option value="IL">IL</option>

<option value="TX">TX</option>

</select>

<button type="submit">Submit</button>

</form>

**Question: what html code will be generated by the following block of code?**

$(function(){

const featuresContainer = document.createElement("li"); featuresContainer.classList.add('ProductCore-featItem');

...

const leftContainer = document.createElement("span");

...

const rightContainer = document.createElement("span");

...

featuresContainer.appendChild(leftContainer);

featuresContainer.appendChild(rightContainer);

...

})



**Answer: The above code will generate an html structure similar to this below:**

<li class="ProductCore-featItem">

<span></span>

<span></span>

</li>