### **1.HTML & CSS**

### 

### 

### **AIM:**

Create a Web Page to Embed a Map with Hotspots, Frames & Links.

### **ALGORITHM:**

**Step 1:** Create an HTML file (index.html)

* Define the document structure using <html>, <head>, and <body>.
* Set the page title and include internal CSS for basic styling.

**Step 2:** Embed an Image Map

* Use the <img> tag to insert an image (map).
* Define a <map> element with a name attribute.
* Add <area> elements inside the <map> with different shapes (rectangle, circle, polygon).
* Assign href attributes to the <area> elements to make them clickable.

**Step 3:** Create Hyperlinks

* Add <a> tags that allow navigation to different pages.
* Use the target attribute to open the linked pages in a frame.

**Step 4:** Add an Inline Frame (iframe)

* Use the <iframe> tag to display linked pages within the same webpage.
* Set the name attribute for the iframe to target it from links.

**Step 5:** Create Additional Pages (page1.html, page2.html)

* Define a simple HTML structure.
* Apply CSS styles for a visually appealing design.
* Test the project to ensure the map hotspots and frames work correctly.

Index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Interactive Map with User-Selected Directions</title>

<!-- Leaflet CSS -->

<link rel="stylesheet" href="https://unpkg.com/leaflet@1.7.1/dist/leaflet.css" />

<!-- Leaflet Routing Machine CSS -->

<link rel="stylesheet" href="https://unpkg.com/leaflet-routing-machine@3.2.5/dist/leaflet-routing-machine.css" />

<!-- Inline CSS -->

<style>

#map {

width: 100%;

height: 80vh;

}

.controls {

margin: 20px;

}

.controls input, .controls button {

padding: 10px;

font-size: 14px;

margin: 5px;

}

.error {

color: red;

font-weight: bold;

}

</style>

</head>

<body>

<div class="controls">

<label for="start-location">Start Location (Area Name): </label>

<input type="text" id="start-location" placeholder="Enter Start Location">

<br>

<label for="end-location">End Location (Area Name): </label>

<input type="text" id="end-location" placeholder="Enter End Location">

<br>

<button onclick="setRoute()">Generate Route</button>

<p id="error-message" class="error"></p> <!-- Error message will be shown here -->

</div>

<!-- Map container -->

<div id="map"></div>

<!-- Leaflet and Leaflet Routing Machine JavaScript -->

<script src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js"></script>

<script src="https://unpkg.com/leaflet-routing-machine@3.2.5/dist/leaflet-routing-machine.js"></script>

<script>

// Initialize the map

var map = L.map('map').setView([13.085, 80.235], 13); // Initial center of the map (Latitude, Longitude) and zoom level

// Add OpenStreetMap tiles (using the tile layer from OSM)

L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {

attribution: '&copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a> contributors'

}).addTo(map);

// Variable to store the routing control object

var routeControl;

// Function to set the route based on user input

function setRoute() {

// Get user input for start and end locations

var startLocation = document.getElementById('start-location').value;

var endLocation = document.getElementById('end-location').value;

var errorMessage = document.getElementById('error-message');

// Clear previous error messages

errorMessage.innerHTML = '';

// Validate input

if (!startLocation || !endLocation) {

errorMessage.innerHTML = 'Please enter both start and end locations!';

return;

}

// Use Nominatim API to get lat/lng for the start and end locations

getCoordinates(startLocation, function(startLatLng) {

if (!startLatLng) {

errorMessage.innerHTML = 'Start location could not be found!';

return;

}

getCoordinates(endLocation, function(endLatLng) {

if (!endLatLng) {

errorMessage.innerHTML = 'End location could not be found!';

return;

}

// If a route already exists, remove it

if (routeControl) {

map.removeControl(routeControl);

}

// Create markers at the start and end locations

L.marker(startLatLng).addTo(map).bindPopup("Start Location: " + startLatLng.toString()).openPopup();

L.marker(endLatLng).addTo(map).bindPopup("End Location: " + endLatLng.toString()).openPopup();

// Create a route between the start and end points

routeControl = L.Routing.control({

waypoints: [

L.latLng(startLatLng), // Start point

L.latLng(endLatLng) // End point

],

routeWhileDragging: true // Allow dragging the route while dragging the markers

}).addTo(map);

// Adjust the map view to fit the route

map.fitBounds(routeControl.getBounds());

});

});

}

// Helper function to get coordinates from Nominatim API

function getCoordinates(location, callback) {

var url = 'https://nominatim.openstreetmap.org/search?format=json&q=' + encodeURIComponent(location);

// Make an HTTP request to Nominatim API

fetch(url)

.then(response => response.json())

.then(data => {

if (data.length > 0) {

// Use the first result's lat and lon

var lat = parseFloat(data[0].lat);

var lon = parseFloat(data[0].lon);

callback([lat, lon]);

} else {

callback(null);

}

})

.catch(error => {

console.error('Error fetching coordinates:', error);

callback(null);

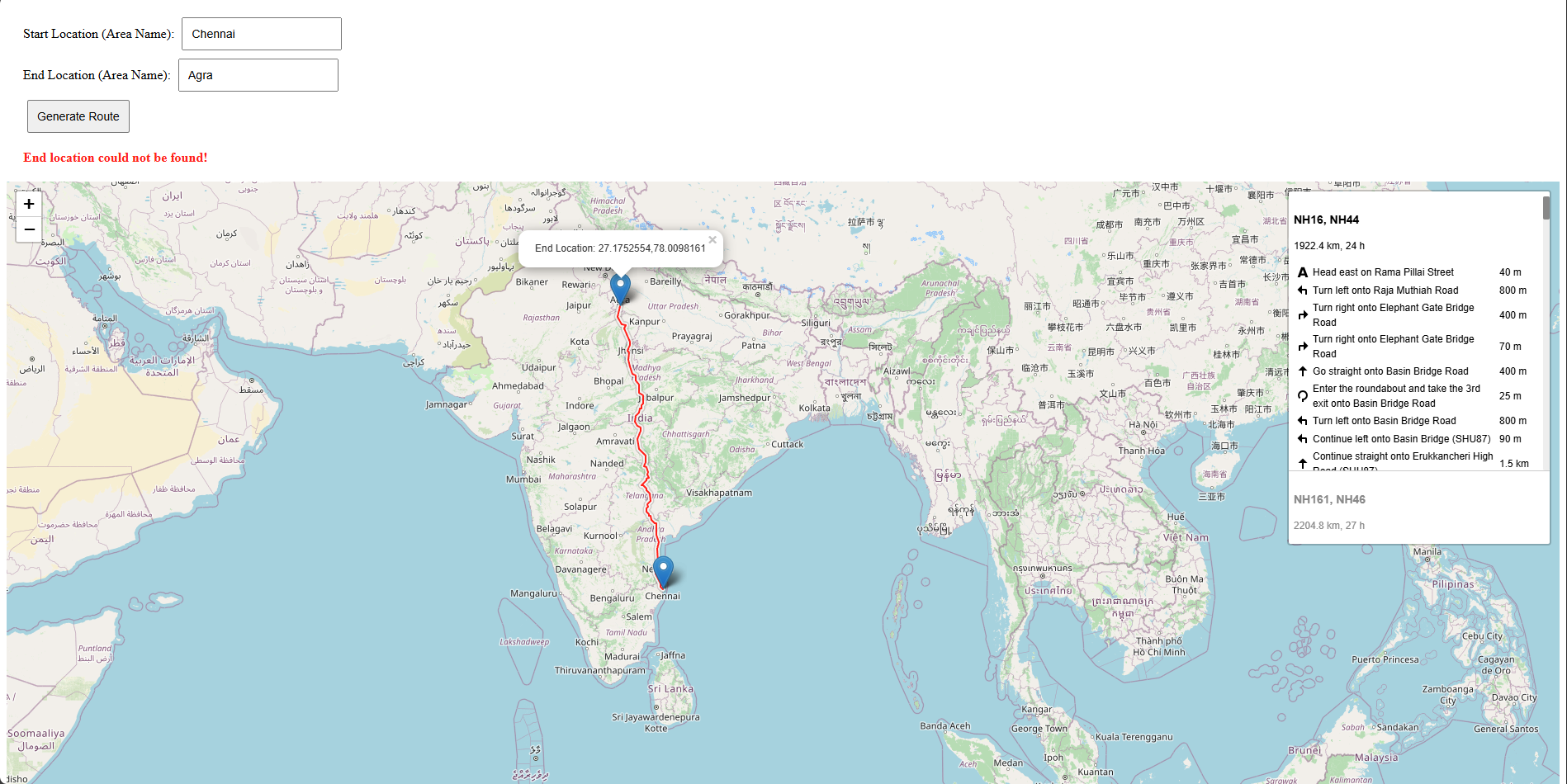
});

}

</script>

</body>

</html>



**RESULT:** The image map with hotspots, frames, and links is successfully created.  
  
**AIM:**

To create a web page that demonstrates the use of **Embedded**, **External**, and **Inline CSS**.

## **ALGORITHM:**

### **Step 1: Create an External CSS File (style.css).**

* Define styles for the body, headings, and paragraphs.
* Save the CSS file in the same directory as index.html.

### **Step 2: Create an HTML File (index.html).**

* Use <html>, <head>, and <body> tags to structure the page.
* Add a <title> tag for the page title.

### **Step 3: Link External CSS.**

* Use <link rel="stylesheet" href="style.css"> inside the <head> tag.

### **Step 4: Apply Embedded CSS.**

* Use a <style> tag inside <head>.
* Define styles for a <div> box with width, height, background color, and border-radius.

### **Step 5: Apply Inline CSS.**

* Use the style attribute in a <p> tag to apply color and font-weight directly.

### **Step 6: Display Content.**

* Add <h2> and <p> tags to demonstrate different CSS types.
* Insert a styled <div> box using embedded CSS.

### **Step 7: Test the Page.**

* Open the index.html file in a browser to check if all styles are applied correctly.

CODE:

Index.html  
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Elegant CSS Styling Example</title>

<link rel="stylesheet" href="styles.css"> <!-- External CSS -->

<style>

/\* Embedded CSS \*/

.embedded-style {

color: #007bff;

font-size: 20px;

font-weight: bold;

text-align: center;

padding: 10px;

border: 2px solid #007bff;

border-radius: 5px;

width: 50%;

margin: 20px auto;

box-shadow: 0px 4px 10px rgba(0, 0, 0, 0.1);

background: #f0f8ff;

}

</style>

</head>

<body style="background-color: #f8f9fa; font-family: 'Poppins', sans-serif; text-align:

center;">

<h1 style="color: #ff5733; font-size: 28px;">🌟 Styled with Inline CSS 🌟</h1>

<p class="embedded-style">✨ This text is styled using Embedded CSS ✨</p>

<p class="external-style">🎨 This text is styled using External CSS 🎨</p>

</body>

</html>

Styles.css

.external-style {

color: #28a745;

font-size: 22px;

font-weight: bold;

text-align: center;

padding: 10px;

border: 2px solid #28a745;

border-radius: 5px;

width: 60%;

margin: 20px auto;

box-shadow: 0px 4px 10px rgba(0, 0, 0, 0.1);

background: #e9ffe9;

transition: 0.3s;

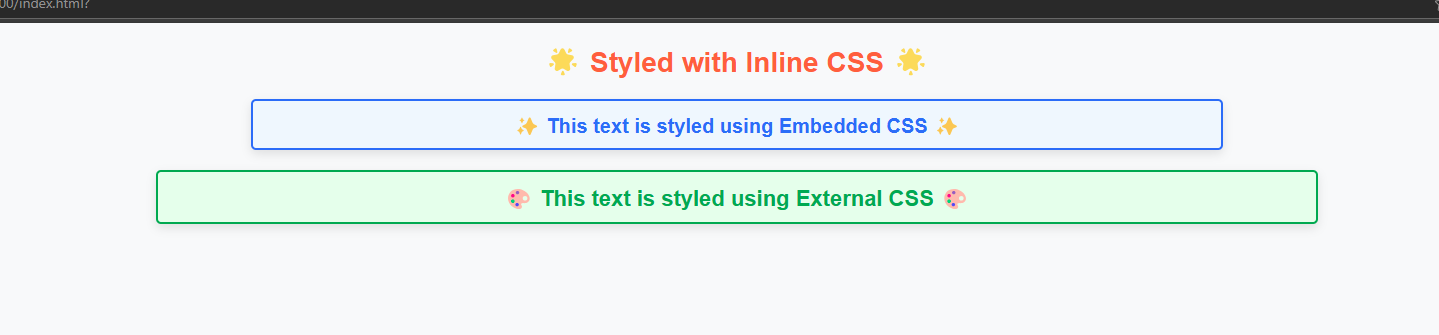
}

external-style:hover {

background: #d4f8d4;

transform: scale(1.05);

}

OUTPUT:  


**Result:**  
The webpage demonstrates external, embedded, and inline CSS styles effectively.