**Sending WhatsApp Messages Using Endless Data's WhatsApp API with HTML & JavaScript**

**Introduction**

In this article, we’ll explore how to use Endless Data's WhatsApp API to send WhatsApp messages from a simple web application built with HTML and JavaScript. This API allows developers to integrate WhatsApp messaging into web applications for purposes like customer notifications, marketing, and support.

The steps in this tutorial include:

1. Setting up a basic HTML form to input the message details.
2. Using JavaScript to send a message request to the API.
3. Handling the response to display success or error messages.

**Step 1: Link Your WhatsApp Number with the API**

To get started with the Endless Data WhatsApp API, you need to link your WhatsApp account.

1. **Log into the Endless Data portal** using your WhatsApp-registered phone number.
2. **Scan the QR code** displayed on the portal by opening WhatsApp on your mobile, navigating to **Linked Devices**, and scanning the code.
3. Once linked, access your **API Key** from the portal. This key is required for authenticating requests from your web application.

**Step 2: Create an HTML Interface for Sending Messages**

Create a simple HTML form to capture message details like the recipient’s phone number, message text, and the API key.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Send WhatsApp Message</title>

<style>

.input-group { margin: 10px 0; }

.btn { padding: 10px 20px; cursor: pointer; }

</style>

</head>

<body>

<h2>Send WhatsApp Message</h2>

<div class="input-group">

<label>API Key:</label>

<input type="text" id="apiKey" required>

</div>

<div class="input-group">

<label>To (Phone Number):</label>

<input type="text" id="phoneNumber" required>

</div>

<div class="input-group">

<label>Message:</label>

<textarea id="messageText" rows="4" required></textarea>

</div>

<button class="btn" onclick="sendMessage()">Send Message</button>

<div id="responseMessage"></div>

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

</body>

</html>

This HTML structure provides an input form for the API key, recipient phone number, and message text. The button triggers the sendMessage() function to send the message.

**Step 3: JavaScript to Send Message Using the API**

Create a JavaScript file (whatsapp.js) to handle the form submission, construct the request, and display the response.

**JavaScript Code**

async function sendMessage() {

const apiKey = document.getElementById('apiKey').value;

const phoneNumber = document.getElementById('phoneNumber').value;

const messageText = document.getElementById('messageText').value;

const apiUrl = 'https://w1.endlessmessages.com/send\_message';

// Construct request payload

const payload = {

apikey: apiKey,

number: phoneNumber,

text: messageText,

};

// Send POST request

try {

const response = await fetch(apiUrl, {

method: 'POST',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(payload)

});

const result = await response.json();

// Display response

if (response.ok) {

document.getElementById('responseMessage').innerText = 'Message sent successfully: ' + result.message\_id;

} else {

document.getElementById('responseMessage').innerText = 'Error: ' + result.message;

}

} catch (error) {

document.getElementById('responseMessage').innerText = 'Error: ' + error.message;

}

}

**Explanation of the JavaScript Code**

1. **Form Input Retrieval**: We get the values for the API key, recipient’s phone number, and message text from the HTML form.
2. **Define the API URL**: The apiUrl variable stores the endpoint to send messages.
3. **Create Payload**: The payload object is structured based on the API requirements, with fields for apikey, number, and text.
4. **Send POST Request**: The fetch() function sends an HTTP POST request to the API with the payload as JSON.
5. **Display Response**: If the message is sent successfully, we display a success message with the message ID. If there’s an error, the error message is shown.

**Step 4: Handling Errors and Responses**

The API may return various response codes indicating the status of the request. Below are common response codes you might encounter:

* **0**: Message sent successfully.
* **-1**: Invalid API key.
* **-2**: Missing parameter.
* **-5**: Unpaired device.
* **-6**: Missing message text.

You can customize the JavaScript to interpret and display different messages based on these codes:

if (response.ok) {

document.getElementById('responseMessage').innerText = 'Message sent successfully: ' + result.message\_id;

} else {

switch (result.code) {

case -1:

document.getElementById('responseMessage').innerText = 'Error: Invalid API key.';

break;

case -2:

document.getElementById('responseMessage').innerText = 'Error: Missing parameter.';

break;

case -5:

document.getElementById('responseMessage').innerText = 'Error: Device not paired.';

break;

case -6:

document.getElementById('responseMessage').innerText = 'Error: Message text is missing.';

break;

default:

document.getElementById('responseMessage').innerText = 'Error: Unknown error occurred.';

break;

}

}

This additional code block categorizes errors and provides feedback based on specific error codes, making it easier for users to understand what went wrong.

**Conclusion**

Using the Endless Data WhatsApp API with HTML and JavaScript, you can quickly set up a web interface to send WhatsApp messages. This approach is lightweight, easy to integrate, and provides real-time feedback on message status.

This example covers the basics, but you can expand the functionality by adding file attachments, scheduling messages, or integrating it into a larger system for customer engagement. For more details on the Endless Data WhatsApp API, refer to the official documentation [Endless Data Documentation](https://www.edatame.com/) .

To access a free API key visit [whatsapp-api APIKEY](https://www.edatame.com/whatsapp-api.html).