### Step 1: Add reCAPTCHA to Your Form

First, sign up for reCAPTCHA and get your site key from the Google reCAPTCHA website.

Add the reCAPTCHA widget to your HTML form using the site key:

***<!DOCTYPE html>***

***<html lang="en">***

***<head>***

***<meta charset="UTF-8">***

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

***<title>reCAPTCHA Example</title>***

***<script src="https://www.google.com/recaptcha/api.js" async defer></script>***

***</head>***

***<body>***

***<form id="myForm" action="javascript:void(0);" onsubmit="submitForm()">***

***<input type="text" name="name" placeholder="Enter your name" required>***

***<input type="email" name="email" placeholder="Enter your email" required>***

***<div class="g-recaptcha" data-sitekey="YOUR\_SITE\_KEY"></div>***

***<br/>***

***<input type="submit" value="Submit">***

***</form>***

***<script>***

***function submitForm() {***

***const recaptchaResponse = document.getElementById('g-recaptcha-response').value;***

***if (recaptchaResponse) {***

***fetch('https://your-backend-service.com/verify-recaptcha', {***

***method: 'POST',***

***headers: {***

***'Content-Type': 'application/json'***

***},***

***body: JSON.stringify({***

***recaptchaResponse: recaptchaResponse***

***})***

***})***

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

***.then(data => {***

***if (data.success) {***

***alert('Captcha verification successful. Proceed with form processing.');***

***// Continue with form processing (e.g., submitting data to your backend)***

***} else {***

***alert('Captcha verification failed. Please try again.');***

***}***

***})***

***.catch(error => {***

***console.error('Error:', error);***

***alert('An error occurred. Please try again.');***

***});***

***} else {***

***alert('Please complete the captcha.');***

***}***

***}***

***</script>***

***</body>***

***</html>***

Replace YOUR\_SITE\_KEY with your actual site key.

### Explanation

1. **HTML Form**: The g-recaptcha div with the data-sitekey attribute is where the CAPTCHA widget appears. This part goes into your HTML form.
2. **JavaScript Verification**: When the form is submitted, the submitForm function is triggered. This function retrieves the value of the g-recaptcha-response field and sends it to your backend service using fetch.
3. **Backend Verification**: In this example, the form submits the CAPTCHA response to an external backend service (e.g., https://your-backend-service.com/verify-recaptcha). The backend service would be responsible for verifying the CAPTCHA response with the Google reCAPTCHA API.

### Backend Service Example (Node.js)

To complete the example, here's how you might implement the backend verification in Node.js using Express:

***const express = require('express');***

***const bodyParser = require('body-parser');***

***const fetch = require('node-fetch');***

***const app = express();***

***const PORT = process.env.PORT || 3000;***

***app.use(bodyParser.json());***

***app.post('/verify-recaptcha', async (req, res) => {***

***const secretKey = 'YOUR\_SECRET\_KEY';***

***const recaptchaResponse = req.body.recaptchaResponse;***

***const verificationUrl = `https://www.google.com/recaptcha/api/siteverify?secret=${secretKey}&response=${recaptchaResponse}`;***

***try {***

***const response = await fetch(verificationUrl, {***

***method: 'POST'***

***});***

***const data = await response.json();***

***if (data.success) {***

***res.json({ success: true });***

***} else {***

***res.json({ success: false });***

***}***

***} catch (error) {***

***console.error('Error verifying reCAPTCHA:', error);***

***res.status(500).json({ success: false, error: 'Internal Server Error' });***

***}***

***});***

***app.listen(PORT, () => {***

***console.log(`Server is running on port ${PORT}`);***

***});***

Replace YOUR\_SECRET\_KEY with your actual secret key.