# GPT DOM Extraction API Documentation

This document provides instructions on how to use the deployed GPT DOM Extraction project as an API and as a webhook. You can fetch DOM extraction results either directly through the API or asynchronously via a webhook endpoint.

## 1. API Usage

Base URL:

https://gpt-dom-extraction-project.onrender.com

Endpoint:

POST /analyze

Content-Type: application/json

Body must contain a 'url' field with the website to analyze.

### Example Requests

cURL Example:

curl -X POST https://gpt-dom-extraction-project.onrender.com/analyze  
 -H "Content-Type: application/json" \  
 -d '{"url":"https://example.com"}'

### Node.js Example (fetch):

async function analyze () {  
 const response = await fetch ("https://gpt-dom-extraction-project.onrender.com/analyze", {  
 method: "POST",  
 headers: {“Content-Type": "application/json" },  
 body: JSON.stringify({url: "https://example.com" })  
 });  
 const data = await response.json();  
 console.log(data);  
}

### Python Example (requests):

import requests  
  
resp = requests.post(  
 "https://gpt-dom-extraction-project.onrender.com/analyze",  
 json={"url": "https://example.com"}  
)  
print(resp.json())

### Example Response

{  
 "url": "https://example.com",  
 "above\_the\_fold": [  
 {  
 "element\_name": "main page title/header",  
 "text": "Welcome to Example",  
 "dom\_path": "html > body > h1"  
 },  
 {  
 "element\_name": "strap-line",  
 "text": "The best example site",  
 "dom\_path": "html > body > p:nth-of-type(1)"  
 },  
 {  
 "element\_name": "primary CTA button",  
 "text": "Get Started",  
 "dom\_path": "html > body > a.btn"  
 }  
 ]  
}

## 2. Webhook Usage

By default, the API returns result immediately. If you prefer to receive results asynchronously, you can provide a webhook URL.

Flow:

1. Provide a webhook URL (e.g., https://client.com/webhook-endpoint).

2. Server scrapes the site.

3. Server sends POST request with results to client’s webhook.

### Webhook Request Example

POST /analyze  
{  
 "url": "https://example.com",  
 "webhook": "https://client.com/webhook-endpoint"  
}

### Webhook Delivery Example

{  
 "url": "https://example.com",  
 "results": {  
 "above\_the\_fold": [  
 {  
 "element\_name": "main page title/header",  
 "text": "Welcome to Example",  
 "dom\_path": "html > body > h1"  
 },  
 ...  
 ]  
 }  
}

### Node.js Webhook Forwarding Example

const fetch = require("node-fetch");  
  
if (req.body.webhook) {  
 await fetch(req.body.webhook, {  
 method: "POST",  
 headers: { "Content-Type": "application/json" },  
 body: JSON.stringify({ url, results: data })  
 });  
 return res.json({ status: "Results will be sent to your webhook" });  
}

## 3. Recommendations

- Consider adding an API key for authentication.

- Puppeteer may take a few seconds to return results (be mindful of timeouts).

- Use API for real-time results, webhook for async processing.

## 4. Deployment Note

Currently, this API is deployed on the free plan of Render.com. Performance may be slower compared to premium plans. For faster and more reliable response times, consider upgrading to a Render.com premium plan.