const text = $input.first().json.cleaned\_text || $input.first().json.markdown\_content || '';

const indicators = {

domains: [],

urls: [],

ips: [],

file\_hashes: [],

emails: [], // emails ignored for hasIndicators

malware\_exploits: [], // only full CVE IDs

threat\_hunting\_keywords: [] // exclude 'CVE' keyword entirely

};

const domainRegex = /\b[a-z0-9\-]+(?:\[\.\]|\.)(?:[a-z0-9\-]+\.)+[a-z]{2,}\b/gi;

const urlRegex = /\bhttps?:\/\/[^\s]+/gi;

const ipRegex = /\b(?:\d{1,3}\.){3}\d{1,3}\b/g;

const md5Regex = /\b[a-f0-9]{32}\b/gi;

const sha1Regex = /\b[a-f0-9]{40}\b/gi;

const sha256Regex = /\b[a-f0-9]{64}\b/gi;

const cveRegex = /\bCVE-\d{4}-\d{4,7}\b/gi;

// Threat hunting keywords WITHOUT 'CVE'

const threatHuntingKeywords = [

'IOC', 'indicator of compromise', 'APT', 'threat hunting',

'malware', 'phishing', 'exploit', 'ransomware', 'lateral movement',

'TTP', 'MITRE ATT&CK', 'exfiltration', 'payload', 'malicious'

];

function normalizeHash(hash) {

return hash.toLowerCase();

}

let match;

// Extract domains

while ((match = domainRegex.exec(text)) !== null) {

indicators.domains.push(match[0]);

}

// Extract URLs

while ((match = urlRegex.exec(text)) !== null) {

indicators.urls.push(match[0]);

}

// Extract IPs

while ((match = ipRegex.exec(text)) !== null) {

indicators.ips.push(match[0]);

}

// Extract hashes

while ((match = md5Regex.exec(text)) !== null) {

indicators.file\_hashes.push({ type: 'MD5', hash: normalizeHash(match[0]) });

}

while ((match = sha1Regex.exec(text)) !== null) {

indicators.file\_hashes.push({ type: 'SHA1', hash: normalizeHash(match[0]) });

}

while ((match = sha256Regex.exec(text)) !== null) {

indicators.file\_hashes.push({ type: 'SHA256', hash: normalizeHash(match[0]) });

}

// Extract full CVE IDs ONLY

while ((match = cveRegex.exec(text)) !== null) {

indicators.malware\_exploits.push(match[0]);

}

// Extract threat hunting keywords except 'CVE'

threatHuntingKeywords.forEach(keyword => {

const regex = new RegExp(`\\b${keyword}\\b`, 'i');

if (regex.test(text) && !indicators.threat\_hunting\_keywords.includes(keyword)) {

indicators.threat\_hunting\_keywords.push(keyword);

}

});

// IMPORTANT: hasIndicators is TRUE only if there's at least one full CVE ID

const hasIndicators = indicators.malware\_exploits.length > 0;

return [

{

json: {

title: $input.first().json.title || null,

url: $input.first().json.url || null,

published\_time: $input.first().json.published\_time || null,

cleaned\_text: text,

has\_indicators: hasIndicators,

indicators

}

}

];