

# Compte rendu — Workflow n8n : Recherche de stages à l'étranger

Leo Torres — DO3

19 janvier 2026

## Table des matières

<b>1</b>	<b>Contexte et objectifs</b>	<b>2</b>
1.1	Objectif du projet . . . . .	2
1.2	Périmètre et hypothèses . . . . .	2
<b>2</b>	<b>Architecture du workflow n8n</b>	<b>2</b>
2.1	Vue d'ensemble du workflow . . . . .	2
2.2	Description des étapes . . . . .	2
2.2.1	Étape 1 — Génération des combinaisons ville × technologie . . . . .	2
2.2.2	Étape 2 — Recherche web (Tavily) . . . . .	2
2.2.3	Étape 3 — Enrichissement scoring via MCP . . . . .	2
2.2.4	Étape 4 — Extraction d'informations via LLM . . . . .	3
2.2.5	Étape 5 — Génération des résumés via LLM . . . . .	3
2.2.6	Étape 6 — Filtrage personnalisé . . . . .	3
2.2.7	Étape 7 — Export des résultats . . . . .	3
2.2.8	Étape 8 — Synthèse finale . . . . .	3
2.2.9	Étape 9 — Notification . . . . .	3
<b>3</b>	<b>Exemples de résultats</b>	<b>3</b>
3.1	Offres extraites (5 à 10 exemples) . . . . .	3
3.2	Fichier CSV exporté (ou capture) . . . . .	4
3.3	Workflow n8n (JSON) dans le repo . . . . .	4
<b>4</b>	<b>Bonus — Méthode de scoring</b>	<b>5</b>
4.1	Méthode utilisée . . . . .	5
4.2	Choix de scoring et justification . . . . .	5
4.3	Évaluation : est-ce que ça fonctionne comme attendu ? . . . . .	5
<b>5</b>	<b>Analyse critique</b>	<b>5</b>
5.1	Difficultés rencontrées . . . . .	5
5.2	Critères de filtrage choisis et justification . . . . .	5
5.3	Apports du workflow et axes d'amélioration . . . . .	6
<b>6</b>	<b>Conclusion</b>	<b>6</b>
<b>A</b>	<b>Annexes</b>	<b>6</b>
A.1	Prompts LLM (extraits) . . . . .	6
A.2	Exemple de sortie JSON complète . . . . .	7

# 1 Contexte et objectifs

## 1.1 Objectif du projet

Le workflow automatise la recherche d'offres de stage a l'étranger a partir d'une matrice *ville × technologie*. Les resultats sont enrichis par un score de ville (MCP), normalises par LLM, dedoublonnes, puis exportes vers Google Sheets et resumes dans une synthese envoyee sur Discord.

## 1.2 Périmètre et hypothèses

- Villes : Berlin, Stockholm.
- Technologies : crypto, devops, cybersecurity, iot (8 combinaisons).
- Source principale : recherche web via Tavily (1 resultat par requete).
- LLM : Google Gemini (gemma-3-4b-it pour extraction, gemma-3-27b-it pour resume, gemini-robotics-er-1.5-preview pour synthese) (à changé en fonction de la dispo des quotas gratuit).
- Limites : dependance aux API, champs souvent absents (salaire, remote), redondances d'URLs.

# 2 Architecture du workflow n8n

## 2.1 Vue d'ensemble du workflow

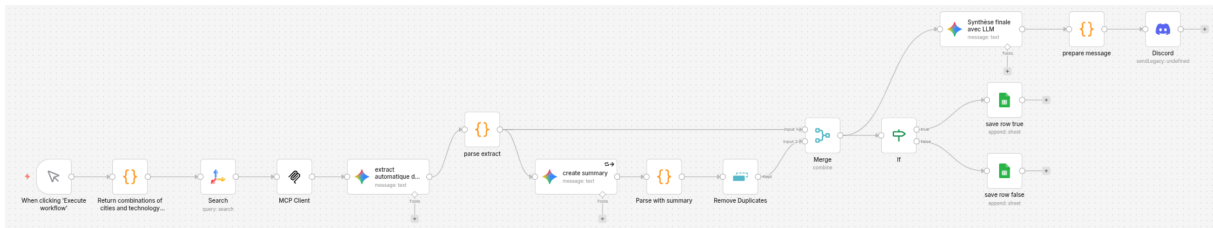


FIGURE 1 – Workflow n8n complet.

## 2.2 Description des étapes

### 2.2.1 Étape 1 — Génération des combinaisons ville × technologie

Un node Code genere les couples ville x technologie a partir de deux tableaux statiques.

### 2.2.2 Étape 2 — Recherche web (Tavily)

Chaque couple declenche une recherche Tavily avec la requete type `company <tech> <city> internship`. La recherche est en `basic`, avec `max_results=2`.

### 2.2.3 Étape 3 — Enrichissement scoring via MCP

Le MCP Client appelle l'outil `find_city_score` avec le nom de ville. Si la ville est connue, un score est renvoye et ajoute aux donnees d'entree; sinon le score est absent et le MCP indique `found=false`.

#### **2.2.4 Étape 4 — Extraction d'informations via LLM**

Le LLM retourne un JSON strict, sans texte additionnel, conforme a un schema fixe (company, job\_title, city, country, remote\_policy, contract\_type, salary, currency, duration, application\_deadline, skills, languages, source\_url, source\_title, city\_score). Les valeurs manquantes sont forcees a null.

#### **2.2.5 Étape 5 — Génération des résumés via LLM**

Un second LLM produit un resume FR de 2 a 3 phrases par offre. Les sorties non-JSON sont nettoyees et parsees dans un node Code.

#### **2.2.6 Étape 6 — Filtrage personnalisé**

Le filtrage est minimal : dedoublonnage par `source_url`, puis routage selon la presence d'un salaire pour separer deux onglets de stockage.

#### **2.2.7 Étape 7 — Export des résultats**

Les offres sont ecrites dans Google Sheets (equivalent CSV), dans deux onglets en fonction du filtre (pour les offres ok selon le filtre et pour les autres). Colonnes : company, job\_title, contract\_type, city, country, salary, currency, duration, application\_deadline, skills, languages, source\_url, source\_title, city\_score, summary, remote\_policy.

#### **2.2.8 Étape 8 — Synthèse finale**

Un LLM genere une synthese lisible avec un titre, le nombre d'offres, un top 3 argumente et des statistiques globales (villes, contrats, remote, salaires).

#### **2.2.9 Étape 9 — Notification**

La synthese est envoyee sur Discord via webhook. Un node Code tronque le message a 1900 caracteres pour respecter la limite.

### **3 Exemples de résultats**

#### **3.1 Offres extraites (5 à 10 exemples)**

Exemple de 5 a 10 offres extraites (toutes les infos dans le JSON) :

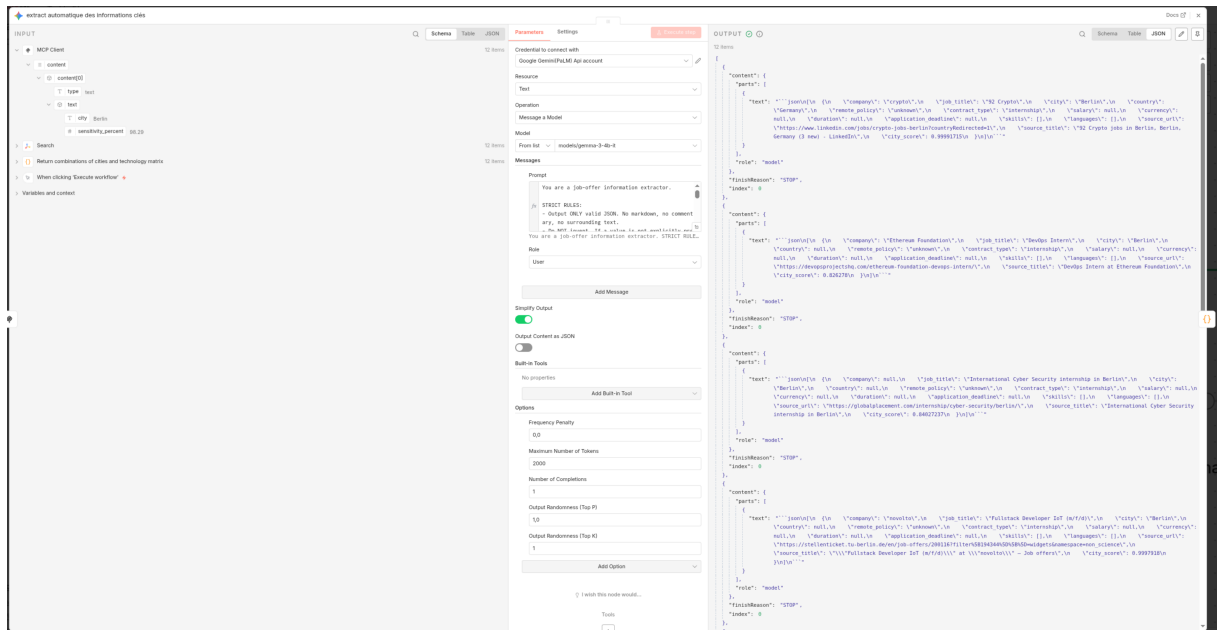


FIGURE 2 – Exemple de 5-10 offres extraites (JSON).

## 3.2 Fichier CSV exporté (ou capture)

company	job_title	city	country	remote_policy	contract_type	salary	currency	duration	application_deadline	skills	languages	source_url	source_title	city_score	summary
Centiglobe	Summer intern	Stockholm		unknown	internship	Competitive				[Blockchain, crypto]		<a href="https://helix.io">https://helix.io</a>	Summer intern	0.78944516	Centiglobe propose un stage d'été en Blockchain et cryptomonnaie à Stockholm. Il s'agit d'un c
crypto	Web3 internship	Barcelona	Spain	unknown	internship	\$30k/year to \$120k/year						<a href="https://helix.io">https://helix.io</a>	Web3 Internship	0.869683	Crypto propose des stages Web3 à Barcelone, en Espagne. Il s'agit d'un contrat de stage avec

FIGURE 3 – Aperçu de l'export CSV / Google Sheets.

## 3.3 Workflow n8n (JSON) dans le repo

```

1 repo/
2   workflow-final.json

```

## 4 Bonus — Méthode de scoring

### 4.1 Méthode utilisée

- Source : je me suis appuyée sur le document de reference present dans `docs/` (Kaklauskas et al., 2018, colonne `Sj`).
- Analyse : j’ai derive un score comparable entre villes a partir de `Sj` pour obtenir une echelle 0–100 (plus lisible).
- Acces via MCP : outil `find_city_score` expose par le serveur MCP.
- Normalisation : suppression d’accents, casse, ponctuation et alias courants (ex. *Krakow*).
- Valeurs manquantes : si la ville n’est pas trouvee, le MCP renvoie `found=false` et aucun score.

Calcul utilise :

$$\text{city\_score} = \left(1 - \frac{Sj - \min(Sj)}{\max(Sj) - \min(Sj)}\right) \times 100$$

Ainsi, un `Sj` faible (forte concordance) donne un score plus eleve, tandis qu’un `Sj` eleve est penalise.

Je me suis aidé de l’IA afin de trouver le calcul pour rendre le calcul plus lisible, précis et cohérent.

### 4.2 Choix de scoring et justification

- Pertinence : indicateur externe pour comparer rapidement l’attractivite des villes.
- Simplicite : table pre-calculée, sans besoin de recalcul complexe.
- Robustesse : normalisation et aliases limitent les erreurs de correspondance.

### 4.3 Évaluation : est-ce que ça fonctionne comme attendu ?

Pour les villes presentes dans la table, le score est coherent et stable. Les limites viennent des villes absentes ou des libelles atypiques (ex. quartiers, regions). Les aliases ameliorent le taux de correspondance, mais un enrichissement géographique serait utile.

## 5 Analyse critique

### 5.1 Difficultés rencontrées

- Choix d’un LLM gratuit : trouver un modele avec un plan gratuit, assez rapide et suffisamment performant pour des extractions fiables.
- Parsing et fusion : difficultes a parser les sorties LLM puis a merger les flux pour n’obtenir qu’un seul objet exploitable.
- Webhook Discord : configuration et tests pour respecter les contraintes de taille des messages.
- Google Sheets : configuration Google Cloud API (OAuth) plus longue que prevu avant de pouvoir ecrire dans un tableur.

### 5.2 Critères de filtrage choisis et justification

Le filtrage est volontairement leger pour maximiser la couverture : dedoublonnage par URL et separation selon `salary` present ou non. Cela permet d’identifier rapidement les offres les plus informatives sans exclure trop de candidats.

### 5.3 Apports du workflow et axes d'amélioration

- Gain de temps important sur la collecte et la mise en forme des offres.
- Améliorations : augmenter `max_results`, ajouter des sources, filtrer par `city_score`, enrichir la deduplication, meilleure gestion d'erreurs et retries.

## 6 Conclusion

Le workflow automatise efficacement la recherche et la synthèse d'offres de stage. L'extraction structurée et l'export permettent une analyse rapide, tandis que le scoring MCP apporte un signal supplémentaire. Les principaux axes d'amélioration concernent la couverture des sources et la robustesse face aux données incomplètes.

## A Annexes

### A.1 Prompts LLM (extraits)

```
1 Prompt d'extraction (Gemini) :
2 You are a job-offer information extractor.
3
4 STRICT RULES:
5 - Output ONLY valid JSON. No markdown, no commentary, no surrounding
   text.
6 - Do NOT invent. If a value is not explicitly present, use null.
7 - Empty lists must be [].
8 - Extract ONE output object per offer in the input array, preserving the
   same order and length.
9
10 OUTPUT FORMAT:
11 Return a JSON array of objects. Each object must follow EXACTLY this
   schema:
12
13 {
14   "company": string|null,
15   "job_title": string|null,
16   "city": string|null,
17   "country": string|null,
18   "remote_policy": "onsite"|"hybrid"|"remote"|"unknown"|null,
19   "contract_type": "internship"|"apprenticeship"|"full_time"|"
   part_time"|"unknown"|null,
20   "salary": string|null,
21   "currency": string|null,
22   "duration": string|null,
23   "application_deadline": string|null,
24   "skills": string[],
25   "languages": string[],
26   "source_url": string|null,
27   "source_title": string|null,
28   "city_score": number|null
29 }
30
31 MAPPING RULES:
32 - source_url = offer.url if present, else null
33 - source_title = offer.title if present, else null
34 - city_score = offer.score if present, else use input.search_score if
   provided, else null
```

```

35 - remote_policy:
36   - "remote" if clearly remote
37   - "hybrid" if clearly hybrid
38   - "onsite" if clearly onsite/on-site
39   - otherwise "unknown"
40 - contract_type:
41   - "internship" if stage/internship/intern
42   - "apprenticeship" if alternance/apprenticeship
43   - otherwise "unknown"
44 - salary: keep original text
45 - currency: "EUR", "USD", "GBP" if clearly identifiable, else null
46
47 Prompt de resume (Gemini) :
48 Generate a short readable summary for each extracted offer.
49
50 STRICT RULES:
51 - Output ONLY valid JSON. No markdown, no commentary.
52 - Return a JSON array with the same length/order as the input array.
53 - Do not include any other keys.
54
55 OUTPUT SCHEMA (per item):
56 {
57   "source_url": string|null,
58   "summary": string
59 }
60
61 SUMMARY RULES:
62 - 2 to 3 sentences maximum.
63 - Include: company (if known), role, location, remote policy, contract
64   type, and salary if present.
65 - If a field is missing, omit it (do not guess).
66 - Write the summary in French.
67
68 Prompt de synthese finale (Gemini) :
69 Tu es un assistant charge de produire un rapport clair et synthetique
70 a partir d'une liste d'offres de stage.
71
72 Ta tache :
73 1) Un titre : "Synthese des offres de stage"
74 2) Le nombre total d'offres pertinentes trouvees
75 3) Le Top 3 des meilleures offres avec pour chacune :
76   - entreprise
77   - poste
78   - ville
79   - pourquoi elle est interessante (1 a 2 phrases max)
80 4) Des statistiques globales :
81   - Villes les plus frequentes
82   - Types de contrats dominants
83   - Remote vs onsite vs hybride
84   - Salaires observes (si presents)
85 5) Une conclusion courte (2-3 phrases) sur la qualite globale des offres

```

## A.2 Exemple de sortie JSON complète

Exemple synthetique (format complet) :

```

1 {
2   "company": "Example Corp",

```

```
3  "job_title": "DevOps Intern",
4  "city": "Berlin",
5  "country": "Germany",
6  "remote_policy": "hybrid",
7  "contract_type": "internship",
8  "salary": "1200 EUR/month",
9  "currency": "EUR",
10 "duration": "6 months",
11 "application_deadline": null,
12 "skills": ["linux", "docker", "kubernetes"],
13 "languages": ["english"],
14 "source_url": "https://example.com/job",
15 "source_title": "DevOps Internship - Example Corp",
16 "city_score": 92.68,
17 "summary": "Example Corp propose un stage DevOps a Berlin en mode
             hybride..."
18 }
```