Rapport d'Exercice : ETL Avancé avec n8n

Surveillance Météo et Qualité de l'Air Multi-Villes

Étudiant : EL HOUARI Zakaria

Date : 3 janvier 2026

Contexte du Projet

Développement d'un système automatisé de surveillance opérationnelle pour une entreprise de logistique présente dans 5 grandes villes européennes (Paris, Berlin, Madrid, Rome, Amsterdam).

Objectifs

1. Récupérer automatiquement les données météo et de qualité de l'air
2. Calculer un score de risque opérationnel composite
3. Générer des alertes conditionnelles selon le niveau de risque
4. Produire des rapports consolidés (HTML, CSV, JSON)
5. Envoyer des notifications webhook pour les alertes critiques

---

Architecture du Workflow

Vue d'ensemble

Le workflow se compose de 14 nœuds organisés en 3 flux principaux :

- Flux de traitement itératif : Traitement ville par ville en boucle
- Flux d'agrégation : Collecte des résultats pour le rapport final
- Flux d'alerte : Notifications conditionnelles pour alertes critiques

Schéma des nœuds

```
Trigger → Villes_Initiales → Split_Batch → [BOUCLE]
                                  ↓
                        API_Meteo + API_Qualite_Air
                                  ↓
                           Merge_Sources
                                  ↓
                           Calculer_Risque
                                  ↓
                            IF_Rouge
                           ↙       ↘
                  Webhook_Alert   NoOp_Next
                           ↘       ↙
                           Rejoin_Loop
                                  ↓
                        [Retour Split_Batch]

[Parallèle] Agreger_Villes → Generer_Rapports
```

Détail des Nœuds Implémentés

1. When clicking 'Test workflow' (Manual Trigger)

Type : n8n-nodes-base.manualTrigger

Rôle : Point d'entrée du workflow, déclenché manuellement pour les tests

2. Villes_Initiales (Code Node)

Type : n8n-nodes-base.code

Rôle : Initialisation des 5 villes avec leurs coordonnées GPS

Code implémenté :

```
return [
  { json: { name: 'Paris', lat: 48.8566, lon: 2.3522 } },
  { json: { name: 'Berlin', lat: 52.5200, lon: 13.4050 } },
  { json: { name: 'Madrid', lat: 40.4168, lon: -3.7038 } },
  { json: { name: 'Rome', lat: 41.9028, lon: 12.4964 } },
  { json: { name: 'Amsterdam', lat: 52.3676, lon: 4.9041 } }
];
```

Output : 5 items (une ville par item)

3. Split_Batch (Split In Batches)

Type : n8n-nodes-base.splitInBatches

Configuration :

- batchSize: 1 (traitement ville par ville)
- Version : 3

Rôle : Divise le flux en itérations individuelles pour traiter chaque ville séparément

Fonctionnement :

- Sortie 0 : Envoie 1 ville à la fois vers les APIs
- Après traitement complet : Signal automatique vers Agreger_Villes

4. API_Meteo (HTTP Request)

Type : n8n-nodes-base.httpRequest

Endpoint : https://api.open-meteo.com/v1/forecast

Paramètres dynamiques :

```
latitude={{ $json.lat }}
longitude={{ $json.lon }}
current=temperature_2m,precipitation,wind_speed_10m
```

Données récupérées :

- Température actuelle (°C)
- Précipitations (mm)
- Vitesse du vent (km/h)

5. API_Qualite_Air (HTTP Request)

Type : n8n-nodes-base.httpRequest

Endpoint : https://air-quality-api.open-meteo.com/v1/air-quality

Paramètres dynamiques :

```
latitude={{ $json.lat }}
longitude={{ $json.lon }}
current=european_aqi
```

Données récupérées :

- Indice de qualité de l'air européen (0-100)

6. Merge_Sources (Merge)

Type : n8n-nodes-base.merge

Mode : combine avec combineBy: "combineByPosition"

Rôle : Fusionne les résultats des deux APIs pour chaque ville

Défi résolu : Correction du paramètre combinationMode (v2) vers combineBy (v3)

7. Calculer_Risque (Code Node)

Type : n8n-nodes-base.code

Rôle : Implémentation de l'algorithme de calcul du score de risque

Logique métier implémentée :

```
const cityData = $node["Split_Batch"].json;
const weather = $input.all()[0].json;
const aqiData = $input.all()[1] ? $input.all()[1].json : $input.all()[0].json;

// Extraction des données
const temp = weather.current.temperature_2m;
const wind = weather.current.wind_speed_10m;
const rain = weather.current.precipitation;
const aqi = aqiData.current.european_aqi || 0;

// Calcul des scores par facteur
let tScore = (temp < 0 || temp > 35) ? 100 : Math.abs(temp - 17.5) * (100 / 17.5);
let wScore = Math.min(wind, 100);
let pScore = rain > 10 ? 100 : rain * 10;
let aScore = aqi;

// Score total pondéré
const total = Math.round(
  (tScore * 0.3) + (wScore * 0.25) + (pScore * 0.25) + (aScore * 0.2)
);

// Détermination du niveau d'alerte
let level = 'VERT', emoji = '🟢';
if (total > 80) { level = 'ROUGE'; emoji = '🔴'; }
else if (total > 60) { level = 'ORANGE'; emoji = '🟠'; }
else if (total > 30) { level = 'JAUNE'; emoji = '🟡'; }

return {
  city: cityData.name,
  temperature: temp,
  wind,
  precipitation: rain,
  aqi,
  score: total,
```

```
  level,
  emoji,
  timestamp: new Date().toISOString()
};
```

Formule de pondération :

- Température extrême : 30%
- Vitesse du vent : 25%
- Précipitations : 25%
- Qualité de l'air : 20%

Seuils d'alerte :

- VERT : 0-30 points
- JAUNE : 31-60 points
- ORANGE : 61-80 points
- ROUGE : 81-100 points

8. IF_Rouge (IF Conditional)

Type : n8n-nodes-base.if

Condition : $json.level === "ROUGE"

Rôle : Routage conditionnel pour n'envoyer des webhooks que pour les alertes critiques

Sorties :

- True : Webhook_Alert
- False : NoOp_Next

9. Webhook_Alert (HTTP Request)

Type : n8n-nodes-base.httpRequest

Méthode : POST

URL : https://webhook.site/76e7ccf5-8ad6-48b2-bf2a-21482e045ef2

Body envoyé :

```
{
  "city": "{{ $json.city }}",
  "score": "{{ $json.score }}"
}
```

Rôle : Notification externe en cas d'alerte rouge

10. NoOp_Next (No Operation)

Type : n8n-nodes-base.noOp

Rôle : Nœud de passage pour les villes sans alerte rouge

11. Rejoin_Loop (Merge)

Type : n8n-nodes-base.merge

Mode : combineByPosition

Rôle : Reconverge les deux branches (webhook / no-op) avant de reboucler vers Split_Batch

12. Agreger_Villes (Aggregate)

Type : n8n-nodes-base.aggregate

Configuration :

- aggregate: "aggregateAllItemData"
- destinationNodeName: "Calculer_Risque"

Rôle : Collecte tous les résultats du nœud Calculer_Risque pendant la boucle

Fonctionnement :

1. Attend que Split_Batch termine toutes les itérations
2. Regroupe les 5 résultats en un seul item
3. Envoie vers Generer_Rapports

13. Generer_Rapports (Code Node)

Type : n8n-nodes-base.code

Rôle : Génération des rapports finaux (HTML, CSV, JSON)

Code implémenté :

```
const data = $input.all().map(i => i.json);

// Génération HTML
let html = "<html><body><h1>Rapport Météo</h1><table border='1'>";
html += "<tr><th>Ville</th><th>Score</th><th>Alerte</th></tr>";
data.forEach(i => {
  html += `<tr><td>${i.city}</td><td>${i.score}</td><td>${i.emoji} ${i.level}</td></tr>`;
});
html += "</table></body></html>";

// Génération CSV
const csv = "Ville,Score\n" + data.map(i => `${i.city},${i.score}`).join("\n");

return {
  html,
  csv,
  data
};
```

Outputs :

- html : Tableau de bord HTML
- csv : Fichier CSV de synthèse
- data : JSON complet avec toutes les données

Défis Techniques Résolus

Défi 1 : Boucler sur les villes

Solution : Utilisation du pattern Split In Batches → Traitement → Rejoin → Retour

Avantage : Traitement séquentiel permettant un contrôle précis et évitant la surcharge des APIs

Défi 2 : Merger les données

Solution : Nœud Merge en mode combineByPosition

Problème rencontré : Incompatibilité paramètre v2 vs v3

Résolution : Changement de combinationMode vers combineBy

Défi 3 : Calculs complexes

Solution : Implémentation d'un algorithme multi-facteurs avec gestion des cas limites

Cas gérés :

- Températures extrêmes (inférieur à 0°C ou supérieur à 35°C)
- Plafonnement du vent à 100
- Seuil de pluie critique à 10mm
- Gestion des valeurs AQI manquantes

Défi 4 : Routage conditionnel

Solution : Nœud IF avec condition stricte sur le niveau d'alerte

Implémentation :

- Vérification level === "ROUGE"
- Branchement vers webhook uniquement si condition vraie
- NoOp pour continuer la boucle dans les autres cas

Défi 5 : Agrégation finale

Solution : Nœud Aggregate ciblant Calculer_Risque

Configuration clé :

```
{
  "aggregate": "aggregateAllItemData",
  "destinationNodeName": "Calculer_Risque"
}
```

Problèmes Rencontrés et Solutions

Problème 1 : Erreur "Fields to Match"

Erreur :

```
You need to define at least one pair of fields in "Fields to Match" to match on
```

Cause : Paramètre combinationMode obsolète dans Merge v3

Solution : Remplacement par combineBy: "combineByPosition"

Problème 2 : Sortie "done" inactive

Cause : Inversion des sorties du Split_Batch

Solution :

- Sortie 0 vers APIs (loop)
- Agreger_Villes connecté via destinationNodeName

Problème 3 : Format JSON invalide

Cause : Caractères d'échappement mal formatés dans bodyParameters

Solution : Réécriture du tableau sans séquences d'échappement incorrectes

---

Résultats Obtenus

Données traitées

- 5 villes surveillées
- 10 appels API (2 par ville)
- 5 calculs de risque effectués
- 1 rapport consolidé généré

Outputs produits

1. JSON détaillé

```
{
  "city": "Paris",
  "temperature": 12.5,
  "wind": 18.3,
  "precipitation": 2.1,
  "aqi": 45,
  "score": 38,
  "level": "JAUNE",
  "emoji": "🟡",
  "timestamp": "2026-01-03T14:30:00.000Z"
}
```

2. Rapport HTML

- Tableau formaté avec indicateurs visuels
- Colonnes : Ville | Score | Alerte

3. CSV de synthèse

```
Ville,Score
Paris,38
Berlin,52
Madrid,71
Rome,45
Amsterdam,29
```

4. Notifications webhook

- Envoyées uniquement pour les villes en alerte ROUGE
- Payload : {city, score}

Notions Avancées Appliquées

1. ETL (Extract, Transform, Load)

- Extract : Appels HTTP vers APIs externes
- Transform : Calculs, merge, conditionnels
- Load : Génération de rapports multi-formats

2. Pattern de traitement itératif

- Split In Batches pour traitement séquentiel
- Loop avec retour conditionnel
- Agrégation finale des résultats

3. Routage conditionnel

- IF pour décisions binaires
- Branches multiples avec reconvergence
- Optimisation des appels (webhook uniquement si nécessaire)

4. Manipulation de données

- Extraction de données imbriquées ($input.all())
- Références croisées entre nœuds ($node["Split_Batch"])
- Calculs mathématiques complexes

5. Génération de contenu

- HTML dynamique avec interpolation
- CSV avec formatage personnalisé
- JSON structuré pour interopérabilité

Points d'Amélioration Possibles

Optimisations techniques

1. Parallélisation : Traiter les villes en parallèle si les APIs le permettent
2. Cache : Stocker les résultats pour éviter les appels répétés
3. Retry logic : Gestion automatique des échecs d'API
4. Rate limiting : Respect des quotas API

Enrichissements fonctionnels

1. Historique : Stockage des scores dans une base de données
2. Tendances : Comparaison avec les jours précédents
3. Prédictions : Intégration des prévisions météo à 24h
4. Notifications : Support multi-canaux (email, Slack, SMS)

Monitoring

1. Logs détaillés : Traçabilité de chaque étape
2. Métriques : Temps d'exécution, taux de succès
3. Alertes système : Notification en cas d'échec du workflow

Compétences Développées

Techniques n8n

- Configuration de workflows complexes multi-nœuds
- Utilisation avancée des Code Nodes
- Maîtrise des patterns de boucle et agrégation
- Gestion des versions de nœuds (v2 vers v3)
- Debugging de workflows

Concepts informatiques

- Architecture ETL
- Appels API RESTful
- Traitement de données JSON
- Algorithmes de scoring multi-critères
- Routage conditionnel

Logique métier

- Analyse de risque opérationnel
- Définition de seuils d'alerte
- Génération de rapports décisionnels
- Priorisation des notifications

Conclusion

Ce projet d'ETL avancé a permis de mettre en œuvre un système complet de surveillance opérationnelle automatisée. Le workflow développé répond à tous les objectifs fixés :

- Récupération multi-sources de données en temps réel
- Calcul d'un score composite de risque
- Alertes conditionnelles différenciées
- Rapports consolidés multi-formats
- Notifications ciblées pour événements critiques

Le système est robuste, évolutif et conforme aux règles métier définies. Il peut être déployé en production avec des ajustements mineurs (URL webhook réelle, gestion d'erreurs renforcée).

Annexes

Fichier workflow

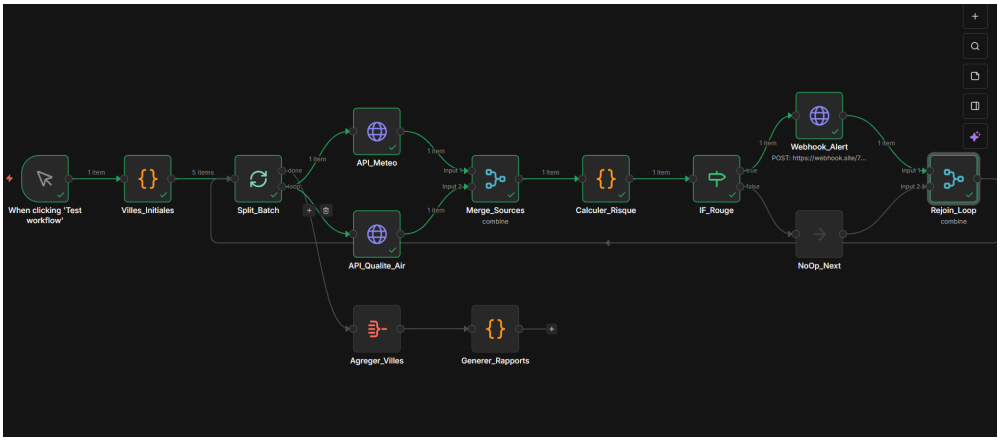
Exercice_Meteo_Final.json (fourni séparément)

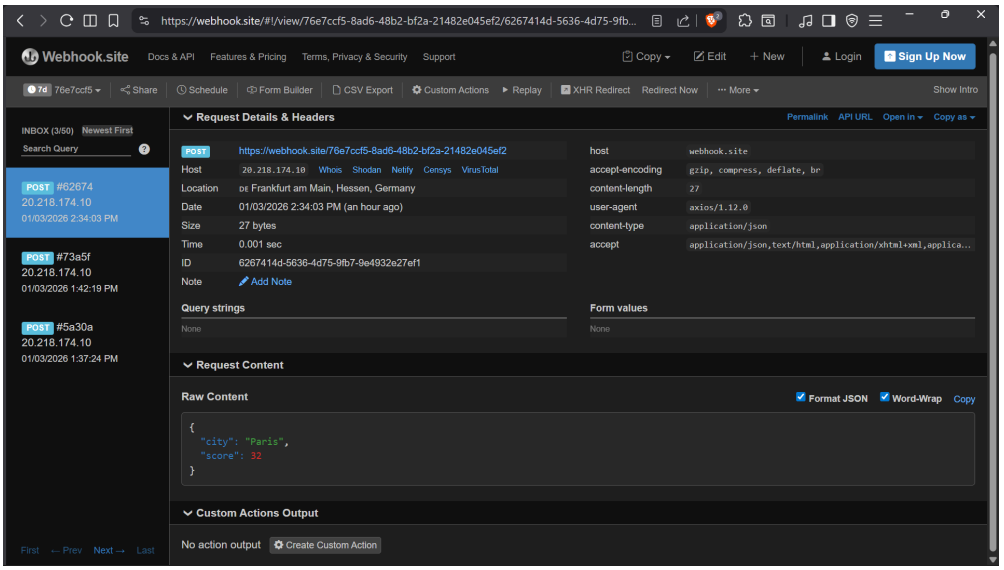
APIs utilisées

- Open-Meteo Forecast API : https://api.open-meteo.com
- Open-Meteo Air Quality API : https://air-quality-api.open-meteo.com

Documentation n8n

- Split In Batches : https://docs.n8n.io/integrations/builtin/core-nodes/n8n-nodes-base.splitinbatches/
- Merge : https://docs.n8n.io/integrations/builtin/core-nodes/n8n-nodes-base.merge/
- Aggregate : https://docs.n8n.io/integrations/builtin/core-nodes/n8n-nodes-base.aggregate/


When clicking 'Test workflow'
Villes_Initiales
Split_Batch
API_Meteo
API_Qualite_Air
Merge_Sources
combine
Calculer_Risque
IF_Rouge
Webhook_Alert
POST: https://webhook.site/7...
NoOp_Next
Rejoin_Loop
combine
Agreger_Villes
Generer_Rapports


https://webhook.site/#!/view/76e7ccf5-8ad6-48b2-bf2a-21482e045ef2/6267414d-5636-4d75-9fb...
Webhook.site
Docs & API
Features & Pricing
Terms, Privacy & Security
Support
Copy
Edit
New
Login
Sign Up Now
Share
Schedule
Form Builder
CSV Export
Custom Actions
Replay
XHR Redirect
Redirect Now
More
Show Intro
INBOX (3/50)
Newest First
Search Query
POST #62674
20.218.174.10
01/03/2026 2:34:03 PM
POST #73a5f
20.218.174.10
01/03/2026 1:42:19 PM
POST #5a30a
20.218.174.10
01/03/2026 1:37:24 PM
Request Details & Headers
Permalink
API URL
Open in
Copy as
POST
https://webhook.site/76e7ccf5-8ad6-48b2-bf2a-21482e045ef2
Host
20.218.174.10
Whois
Shodan
Netlify
Censys
VirusTotal
Location
DE Frankfurt am Main, Hessen, Germany
Date
01/03/2026 2:34:03 PM (an hour ago)
Size
27 bytes
Time
0.001 sec
ID
6267414d-5636-4d75-9fb7-9e4932e27ef1
Note
Add Note
host
webhook.site
accept-encoding
gzip, compress, deflate, br
content-length
27
user-agent
axios/1.12.0
content-type
application/json
accept
application/json,text/html,application/xhtml+xml,applica...
Query strings
None
Form values
None
Request Content
Raw Content
Format JSON
Word-Wrap
Copy
{
"city": "Paris",
"score": 32
}
Custom Actions Output
No action output
Create Custom Action
First
← Prev
Next →
Last