Guide de déploiement — Application Node.js sur Windows Server + IIS

Cas d'usage : petite application de questionnaires (trafic léger)

Ce guide couvre deux méthodes de déploiement :

- **Méthode recommandée** — IIS en reverse proxy vers Node.js (process géré par NSSM, en tant que service Windows).
- **Méthode alternative** — iisnode (module historique, intégration plus directe avec IIS mais moins maintenu).

> Le sommaire ci-dessous est généré automatiquement par DokuWiki à partir des titres de section (comportement par défaut dès qu'une page contient plus de 3 titres).

---

1. Objectif et architecture retenue

Ce document décrit, pas à pas, comment déployer une application Node.js sur un Windows Server existant, en utilisant IIS comme point d'entrée public. Il est pensé pour une application à faible trafic (formulaires / réponses à des questionnaires), mais les étapes restent valables pour la plupart des applications Node.js classiques (Express, Fastify, etc.).

Architecture retenue : IIS reste le serveur web exposé sur le port 80/443. Il n'exécute pas directement le code Node — il agit comme reverse proxy et transmet les requêtes à l'application Node, qui tourne en arrière-plan comme un service Windows sur un port local (ex. 3000). Cette approche est celle recommandée aujourd'hui (plus proche de ce qui se fait avec nginx sous Linux), car iisnode n'est plus activement maintenu.

Schéma : Internet → IIS (port 443, certificat SSL) → module URL Rewrite + ARR → Node.js (127.0.0.1:3000, exécuté par NSSM en tant que service Windows).

2. Prérequis

- Accès administrateur au Windows Server (2016, 2019, 2022 ou équivalent).
- Le rôle « Serveur Web (IIS) » disponible via Gestionnaire de serveur.
- Accès Internet sortant depuis le serveur (ou paquets téléchargés à l'avance) pour installer Node.js, URL Rewrite, ARR et NSSM.
- Le code de l'application Node.js prêt (dépôt Git ou archive), avec un fichier package.json valide et un point d'entrée clair (ex. server.js).
- Un nom de domaine ou sous-domaine interne pointant vers l'IP du serveur, si un accès via nom d'hôte est souhaité.
- Un certificat SSL (interne ou public) si l'application doit être servie en HTTPS — fortement recommandé dès lors que le questionnaire collecte des données.

3. Étape 1 — Installer le rôle IIS

Si IIS n'est pas encore installé sur le serveur :

```
# PowerShell, en tant qu'administrateur
Install-WindowsFeature -Name Web-Server -IncludeManagementTools
```

Ou via l'interface graphique : Gestionnaire de serveur → Ajouter des rôles et fonctionnalités → Serveur Web (IIS) → cocher au minimum : Contenu statique, Document par défaut, Journalisation HTTP, Filtrage des requêtes, Console de gestion IIS.

4. Étape 2 — Installer Node.js

- Télécharger la version LTS depuis nodejs.org (installeur .msi 64 bits).
- Lancer l'installation avec les options par défaut (cocher « Add to PATH »).
- Vérifier l'installation :

```
node -v
npm -v
```

Les deux commandes doivent afficher un numéro de version sans erreur.

5. Étape 3 — Installer les modules IIS nécessaires

5.1 URL Rewrite Module

Indispensable pour rediriger les requêtes entrantes vers l'application Node. Télécharger et installer « URL Rewrite Module 2.1 » depuis le site officiel IIS (iis.net/downloads/microsoft/url-rewrite).

5.2 Application Request Routing (ARR)

Nécessaire pour que IIS agisse comme reverse proxy vers un port local. Télécharger et installer « Application Request Routing 3.0 » depuis le même site (iis.net/downloads/microsoft/application-request-routing).

Après installation d'ARR, ouvrir IIS Manager → sélectionner le nœud du serveur (racine) → double-cliquer sur « Application Request Routing Cache » → dans le panneau Actions à droite, cliquer sur « Server Proxy Settings » → cocher « Enable proxy » → Appliquer.

> **Redémarrage :** un redémarrage du service IIS (ou du serveur) est recommandé après l'installation des deux modules : `iisreset` en ligne de commande administrateur.

6. Étape 4 — Préparer l'application Node.js

- Copier le code de l'application dans un dossier dédié, par exemple : D:\Apps\questionnaire-app\
- Installer les dépendances de production :

```
cd D:\Apps\questionnaire-app
npm ci --omit=dev
```

- S'assurer que l'application écoute uniquement en local (127.0.0.1) sur un port dédié, par exemple 3000, et non sur 0.0.0.0 avec un port public — c'est IIS qui doit rester la seule porte d'entrée externe :

```
// server.js
const PORT = process.env.PORT || 3000;
app.listen(PORT, "127.0.0.1", () => {
  console.log(`App started on 127.0.0.1:${PORT}`);
});
```

- Tester en local avant d'aller plus loin :

```
node server.js
# puis, dans un autre terminal :
curl http://127.0.0.1:3000
```

7. Étape 5 — Exécuter Node.js comme service Windows (NSSM)

Sans cette étape, l'application s'arrête dès que la session qui l'a lancée se ferme, et ne redémarre pas après un reboot du serveur. NSSM (Non-Sucking Service Manager) résout ce problème simplement.

- Télécharger NSSM (nssm.cc) et extraire nssm.exe (version 64 bits) dans un dossier, ex. C:\Tools\nssm\.
- Installer le service, en ligne de commande administrateur :

```
cd C:\Tools\nssm
nssm install QuestionnaireApp
```

Une fenêtre NSSM s'ouvre. Renseigner :

- Path : chemin complet vers node.exe (ex. C:\Program Files\nodejs\node.exe)
- Startup directory : D:\Apps\questionnaire-app
- Arguments : server.js

Dans l'onglet « I/O », rediriger la sortie standard et les erreurs vers des fichiers de log (ex. D:\Apps\questionnaire-app\logs\out.log et err.log) pour faciliter le diagnostic.

Dans l'onglet « Exit actions », choisir « Restart application » en cas d'arrêt inattendu.

Démarrer et vérifier le service :

```
nssm start QuestionnaireApp
Get-Service QuestionnaireApp
curl http://127.0.0.1:3000
```

> **Alternative sans NSSM :** le module npm « node-windows » permet aussi de créer un service Windows directement depuis un script Node, mais NSSM reste plus simple à administrer pour une équipe non habituée à Node.

8. Étape 6 — Configurer IIS comme reverse proxy

Créer un fichier web.config à la racine du dossier qui sera utilisé comme racine du site IIS (peut être un dossier vide dédié, distinct du code Node, ex. D:\Sites\questionnaire\) :

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
  <system.webServer>
    <rewrite>
      <rules>
        <rule name="ReverseProxyToNode" stopProcessing="true">
          <match url="(.*)" />
          <action type="Rewrite"
                  url="http://127.0.0.1:3000/{R:1}" />
        </rule>
      </rules>
    </rewrite>
  </system.webServer>
</configuration>
```

Ce fichier indique à IIS (via URL Rewrite + ARR) de transmettre toutes les requêtes reçues vers l'application Node en écoute sur le port local 3000.

9. Étape 7 — Créer le site dans IIS Manager

1. Ouvrir IIS Manager → clic droit sur « Sites » → Ajouter un site web.
2. Nom du site : ex. QuestionnaireApp.
3. Chemin d'accès physique : D:\Sites\questionnaire\ (le dossier contenant le web.config créé à l'étape précédente).
4. Liaison (Binding) : type http, port 80 (et https / port 443 une fois le certificat installé, voir étape suivante), nom d'hôte = le domaine/sous-domaine prévu.
5. Pool d'applications : créer ou assigner un pool dédié, en mode « No Managed Code » (Node gère son propre runtime, IIS n'a pas besoin du pipeline .NET).
6. Démarrer le site.

10. Étape 8 — Configurer HTTPS

Fortement recommandé dès que le questionnaire transmet des données, même peu sensibles.

- Obtenir un certificat SSL : certificat interne (PKI de l'entreprise) si le site n'est accessible qu'en interne, ou certificat public (ex. Let's Encrypt, via l'outil win-acme, ou un certificat acheté) si le site est exposé sur Internet.
- Dans IIS Manager, sélectionner le site → « Certificats de serveur » → importer le certificat.
- Ajouter une liaison https sur le port 443 pour le site, en sélectionnant ce certificat.
- Ajouter une règle de redirection HTTP → HTTPS dans le web.config (ou via URL Rewrite) pour forcer le chiffrement :

```
<rule name="ForceHTTPS" stopProcessing="true">
  <match url="(.*)" />
  <conditions>
    <add input="{HTTPS}" pattern="off" />
  </conditions>
  <action type="Redirect"
          url="https://{HTTP_HOST}/{R:1}"
          redirectType="Permanent" />
</rule>
```

Cette règle doit être placée avant la règle ReverseProxyToNode dans le web.config.

11. Étape 9 — Pare-feu Windows

Autoriser uniquement les ports nécessaires en entrée : 80 et 443 pour IIS. Le port 3000 (Node) ne doit pas être ouvert à l'extérieur — il ne doit rester accessible qu'en local puisque Node écoute sur 127.0.0.1.

```
New-NetFirewallRule -DisplayName "IIS HTTP" -Direction Inbound -Protocol TCP
-LocalPort 80 -Action Allow
New-NetFirewallRule -DisplayName "IIS HTTPS" -Direction Inbound -Protocol
TCP -LocalPort 443 -Action Allow
```

12. Étape 10 — Tester le déploiement

1. Depuis le serveur : `curl http://127.0.0.1:3000` doit répondre (Node direct).
2. Depuis le serveur : `curl http://localhost/` doit répondre la même chose (via IIS + reverse proxy).
3. Depuis un poste client : accéder à `http://nom-du-site` puis `https://nom-du-site` dans un navigateur.
4. Vérifier que le formulaire/questionnaire fonctionne de bout en bout (soumission, enregistrement des réponses).
5. Redémarrer le serveur (ou au moins le service NSSM et IIS) et vérifier que tout redémarre automatiquement sans intervention manuelle.

13. Étape 11 — Sécuriser l'application

Quelques points à vérifier côté code Node, en plus de la configuration IIS/réseau :

- Utiliser le middleware helmet (`npm install helmet`) pour les en-têtes de sécurité HTTP par défaut.
- Valider et assainir les entrées du questionnaire côté serveur (pas seulement côté client).
- Mettre en place une limite de débit (rate limiting, ex. express-rate-limit) pour éviter les soumissions abusives ou automatisées.
- Ne jamais stocker de secrets (clés API, identifiants base de données) en dur dans le code — utiliser des variables d'environnement définies au niveau du service NSSM ou dans un fichier .env exclu du dépôt Git.
- Journaliser les erreurs applicatives (logs NSSM déjà configurés à l'étape 5) et surveiller régulièrement.

14. Étape 12 — Logs et supervision

- Logs applicatifs Node : fichiers définis dans NSSM (D:\Apps\questionnaire-app\logs\out.log et err.log).
- Logs IIS : activés par défaut dans %SystemDrive%\inetpub\logs\LogFiles\, utiles pour voir les requêtes entrantes et les codes de retour HTTP.
- Pour un suivi plus poussé, envisager un outil de supervision (ex. Uptime Kuma en interne, ou une simple tâche planifiée qui vérifie la disponibilité du endpoint et alerte par email en cas d'échec).

15. Étape 13 — Procédure de mise à jour de l'application

1. Récupérer la nouvelle version du code (`git pull` ou copie de fichiers) dans D:\Apps\questionnaire-app\.
2. Installer les éventuelles nouvelles dépendances : `npm ci –omit=dev`.
3. Arrêter le service : `nssm stop QuestionnaireApp`.
4. Redémarrer le service : `nssm start QuestionnaireApp`.
5. Vérifier rapidement (étape 10) que le site répond correctement.

> **Bonnes pratiques :** pour limiter les interruptions, prévoir les mises à jour en dehors des heures d'utilisation du questionnaire, et garder une copie de la version précédente pour pouvoir revenir en arrière rapidement en cas de problème.

16. Méthode alternative — iisnode

iisnode est un module IIS historique qui permet à IIS d'exécuter directement le code Node (sans passer par un reverse proxy ni un service séparé). Il fonctionne toujours mais n'est plus activement maintenu ; il reste néanmoins une option valable pour un déploiement simple et rapide si l'équipe préfère éviter la gestion d'un service Windows séparé (NSSM).

16.1 Installation

- Installer le module URL Rewrite (voir étape 3.1).
- Télécharger et installer iisnode (dépôt GitHub officiel « azure/iisnode »), version correspondant à l'architecture du serveur (x64).

16.2 Configuration

web.config à placer à la racine de l'application (le dossier contenant directement le code Node, ex. server.js) :

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <system.webServer>
    <handlers>
      <add name="iisnode" path="server.js" verb="*"
           modules="iisnode" />
    </handlers>
    <rewrite>
      <rules>
        <rule name="DynamicContent">
          <match url="/*" />
          <action type="Rewrite" url="server.js" />
        </rule>
      </rules>
    </rewrite>
  </system.webServer>
</configuration>
```

Créer ensuite le site IIS en pointant directement sur ce dossier (pool d'application en « No Managed Code »). Les logs sont générés automatiquement par iisnode dans un sous-dossier « iisnode » à côté de l'application, utile en cas d'erreur.

> **Quand préférer NSSM + reverse proxy plutôt qu'iisnode :** iisnode redémarre le process Node à chaque requête après une période d'inactivité (comportement pensé pour IIS classique), ce qui peut introduire une latence. La méthode NSSM (étapes 5 à 9) garde le process Node actif en permanence et est plus proche des pratiques actuelles — c'est l'approche à privilégier pour un déploiement durable.

17. Checklist finale

Élément	Statut
Rôle IIS installé	☐
Node.js installé et vérifié (node -v)	☐
URL Rewrite + ARR installés, proxy activé	☐
Application copiée, dépendances installées (npm ci)	☐
Application testée en local sur 127.0.0.1:3000	☐
Service NSSM créé, démarré, redémarrage auto configuré	☐

Élément	Statut
web.config reverse proxy en place	☐
Site créé dans IIS, binding http configuré	☐
Certificat SSL installé, binding https configuré	☐
Redirection HTTP → HTTPS active	☐
Pare-feu : ports 80/443 ouverts, 3000 fermé en externe	☐
Test de bout en bout du questionnaire réussi	☐
Redémarrage serveur testé (reprise automatique)	☐
Logs NSSM et IIS vérifiés	☐
Sécurité applicative revue (helmet, validation, rate limit)	☐

18. Annexe — Dépannage rapide

Symptôme	Piste de résolution
Erreur 502.3 (Bad Gateway) dans IIS	Vérifier que le service NSSM tourne (`Get-Service QuestionnaireApp`) et que `curl http://127.0.0.1:3000` répond en local.
Erreur 500.19 sur le site IIS	web.config invalide ou module URL Rewrite/ARR non installé — vérifier la syntaxe XML et la présence des modules.
Le site répond en HTTP mais pas en HTTPS	Vérifier le binding 443 et que le certificat sélectionné est valide et non expiré.
L'application ne redémarre pas après reboot du serveur	Vérifier que le service NSSM est configuré en démarrage automatique (Services.msc → QuestionnaireApp → Type de démarrage = Automatique).
Modifications du code non prises en compte	S'assurer d'avoir bien exécuté `nssm stop` / `nssm start` après la mise à jour du code (voir étape 13).

Document préparé le 18 août 2026.