**1. Introduction**

Ce projet est conçu pour **déployer un logiciel** à distance sur plusieurs ordinateurs Windows et pour **collecter des informations** sur la version d’un service ou d’une application déjà installée. L’application fournit :

1. Une **interface web** (tableau de bord) où l’on peut spécifier une liste d’ordinateurs et un chemin ou une URL d’installeur.
2. Un **serveur Node.js/Express** qui gère les requêtes, exécute des scripts PowerShell en local (WinRM) pour interagir à distance avec les machines cibles.
3. Des **scripts PowerShell** qui font les actions suivantes :
   * Vérifier la connectivité WinRM des machines.
   * Lire une clé de registre pour déterminer la version d’un logiciel.
   * Déployer un exécutable d’installation en mode silencieux et suivre la progression.

Grâce à ce projet, l’utilisateur peut superviser sur un tableau de bord la progression de l’installation et l’état (en ligne/hors ligne, version installée, etc.) de chaque ordinateur.

**2. Aperçu de l’Architecture**

[ Interface Web ] --(HTTP/REST)--> [ Serveur Node.js/Express ]

|

[ exécute PowerShell ]

|

[ Machines Windows cibles ]

* **Interface Web** : Une page HTML/JavaScript (Chart.js pour les graphiques, Tailwind pour le style) qui communique avec le serveur via des endpoints REST.
* **Serveur Node.js/Express** :
  + Fournit des routes (/api/get-computer-info, /api/deploy, etc.).
  + Lance des commandes PowerShell via child\_process.exec pour exécuter localement (ou via WinRM) les opérations de vérification et d’installation.
* **Machines Windows cibles** :
  + Doivent être configurées pour accepter WinRM ou posséder un accès distant autorisé.
  + Le script PowerShell exécute localement des actions (vérification de registre, installation, etc.).

**3. Prérequis**

1. **Node.js** : version 14 ou supérieure recommandée.
2. **NPM** : pour gérer les dépendances du projet.
3. **Accès WinRM** : Les ordinateurs cibles doivent autoriser les connexions WinRM (port TCP 5985/5986 par défaut) et le compte utilisé doit avoir des droits d’administrateur à distance.
4. **Développement local** (optionnel) :
   * Un éditeur de code (VS Code, par exemple).
   * Un environnement Windows ou un autre OS (Windows/Mac/Linux) configuré pour que WinRM puisse atteindre les cibles Windows (attention à la configuration du pare-feu).

**4. Installation & Configuration**

**4.1 Récupération du projet**

* Cloner le dépôt ou copier les fichiers du projet dans un dossier local.
* Installer les dépendances :

npm install dotenv

* Créez un fichier .env à la racine du projet (ne surtout pas le committer dans Git), par exemple :

WINRM\_USERNAME=MonUtilisateur

WINRM\_PASSWORD=MonMotDePasse

**4.3 Fichiers Clés**

* **index.js (ou server.js)** : Fichier principal du serveur Node.js/Express.
* **public/** : Dossier contenant les fichiers statiques (HTML, CSS, JS) pour l’interface web.
* **.env** (non committé) : Stocke les identifiants et autres variables sensibles.

**5. Lancement en Mode Développement**

1. Assurez-vous que .env est configuré (nom d’utilisateur et mot de passe WinRM).
2. Lancez le serveur :

npm run start

1. Ouvrez votre navigateur à l’adresse :

<http://localhost:3000>

**6. Utilisation de l’Interface Web**

1. **Champ “Software URL”** : Indiquez l’URL publique de l’exécutable d’installation (par ex. https://exemple.com/installer.exe).
2. **Champ “Computer List”** : Entrez la liste de noms d’hôte (ou d’adresses IP) des machines cibles, une par ligne.
3. **Vérification de Versions** :
   * Dès que vous collez ou modifiez la liste des machines, le projet exécute l’endpoint /api/get-computer-info.
   * L’application affiche pour chacune la disponibilité (en ligne / hors ligne) et la version installée. Un petit **spinner** (loading) s’affiche le temps de la requête.
4. **Lancement du Déploiement** :
   * Cliquez sur **“Start Deployment”**.
   * Le serveur lance un script PowerShell sur chaque machine (si atteignable) pour télécharger et installer l’exécutable de façon silencieuse.
   * Une barre de progression et un statut (“deploying”, “success”, “failed”) seront affichés pour chaque ordinateur. Les graphiques (Chart.js) se mettent à jour en temps réel.

**7. Script PowerShell de Déploiement**

Le script PowerShell généré (et exécuté à distance) fait notamment :

* **Test de connexion WinRM** : Test-WSMan ou New-PSSession
* **Création d’un dossier temporaire** : C:\Windows\Temp\install\_...
* **Téléchargement** de l’exécutable (via WebClient.DownloadFile).
* **Installation silencieuse** (divers paramètres : /S, /qn, etc.) pour supporter différents types d’installateurs.
* **Redémarrage** planifié de la machine à la fin (optionnel, selon vos besoins).

Chaque progression est renvoyée via des lignes “PROGRESS: …” que l’interface analyse pour mettre à jour la barre de progression.

**9. Bonnes Pratiques de Sécurité**

1. **.env dans le .gitignore** : Ne jamais committer le fichier .env avec vos identifiants.
2. **HTTPS** : Protégez les échanges entre l’interface web et le serveur (certificat SSL).
3. **Limitation d’accès WinRM** : Restreindre le trafic WinRM au réseau interne ou via un VPN.
4. **Droits d’administrateur** : Le compte WinRM doit avoir les droits nécessaires, mais idéalement pas plus de privilèges que nécessaire.
5. **Rotation des mots de passe** : Changer régulièrement votre mot de passe ou utiliser un gestionnaire de secrets (Azure Key Vault, AWS Parameter Store, etc.).

**10. Conclusion**

Ce projet offre un **système de déploiement distant** convivial via un **tableau de bord Web** :

* **Suivi en temps réel** : Progrès d’installation, état des machines, réussite/échec.
* **Interface intuitive** : Saisie de la liste de machines, affichage d’infos de version, graphiques.
* **Extensible** : Vous pouvez adapter le script PowerShell pour d’autres cas d’utilisation (désinstallation, mises à jour, configuration, etc.).

Vous pouvez désormais exécuter et maintenir ce service :

1. **Installer les dépendances**
2. **Configurer .env** pour les identifiants
3. **Lancer le serveur** (et éventuellement le “build” si votre configuration l’exige)
4. **Accéder** au tableau de bord via http://localhost:3000 (ou le port configuré)

N’hésitez pas à adapter ce **projet** selon vos besoins (ajout d’authentification, de logs centralisés, etc.) pour un déploiement à plus grande échelle.

**Ressources Supplémentaires**

* **Node.js** : https://nodejs.org/en/docs/
* **Express** : https://expressjs.com/fr/
* **Documentation de WinRM** : <https://docs.microsoft.com/fr-fr/windows/win32/winrm/>
* **PM2** : https://pm2.keymetrics.io/
* **Chart.js** : <https://www.chartjs.org/>
* **Tailwind CSS** : <https://tailwindcss.com/>