Introduction

Dans le cadre d'un projet personnel, j'ai développé une application web permettant de transformer n'importe quel PC Windows en box TV complète pour un environnement Free Box, contrôlable depuis un smartphone. Le système offre un accès à 28 chaînes de télévision gratuites et propose une expérience utilisateur fluide comparable aux solutions commerciales.

Contexte du projet

L'objectif initial était simple : pouvoir regarder la télévision sur mon PC de travail pendant les pauses, avec un contrôle à distance depuis mon smartphone pour éviter les allers-retours constants entre le bureau et l'écran. Plutôt que d'investir dans une box TV ou un abonnement supplémentaire, j'ai choisi de développer une solution sur mesure exploitant les ressources déjà disponibles.

Architecture technique

Stack technologique

L'application repose sur une architecture Node.js classique :

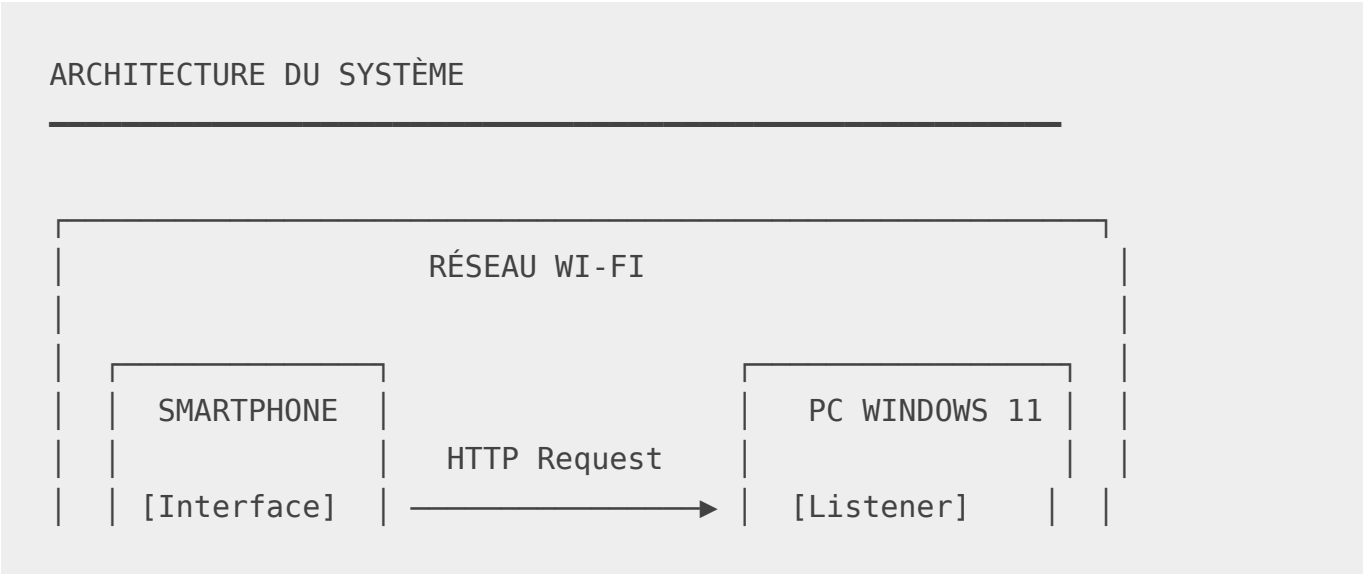
- Backend : Express.js pour l'API REST
- Frontend : HTML/CSS/JavaScript vanilla (pas de framework)
- Automation : Puppeteer-core pour le contrôle du navigateur
- System Control : NirCmd pour les interactions système Windows

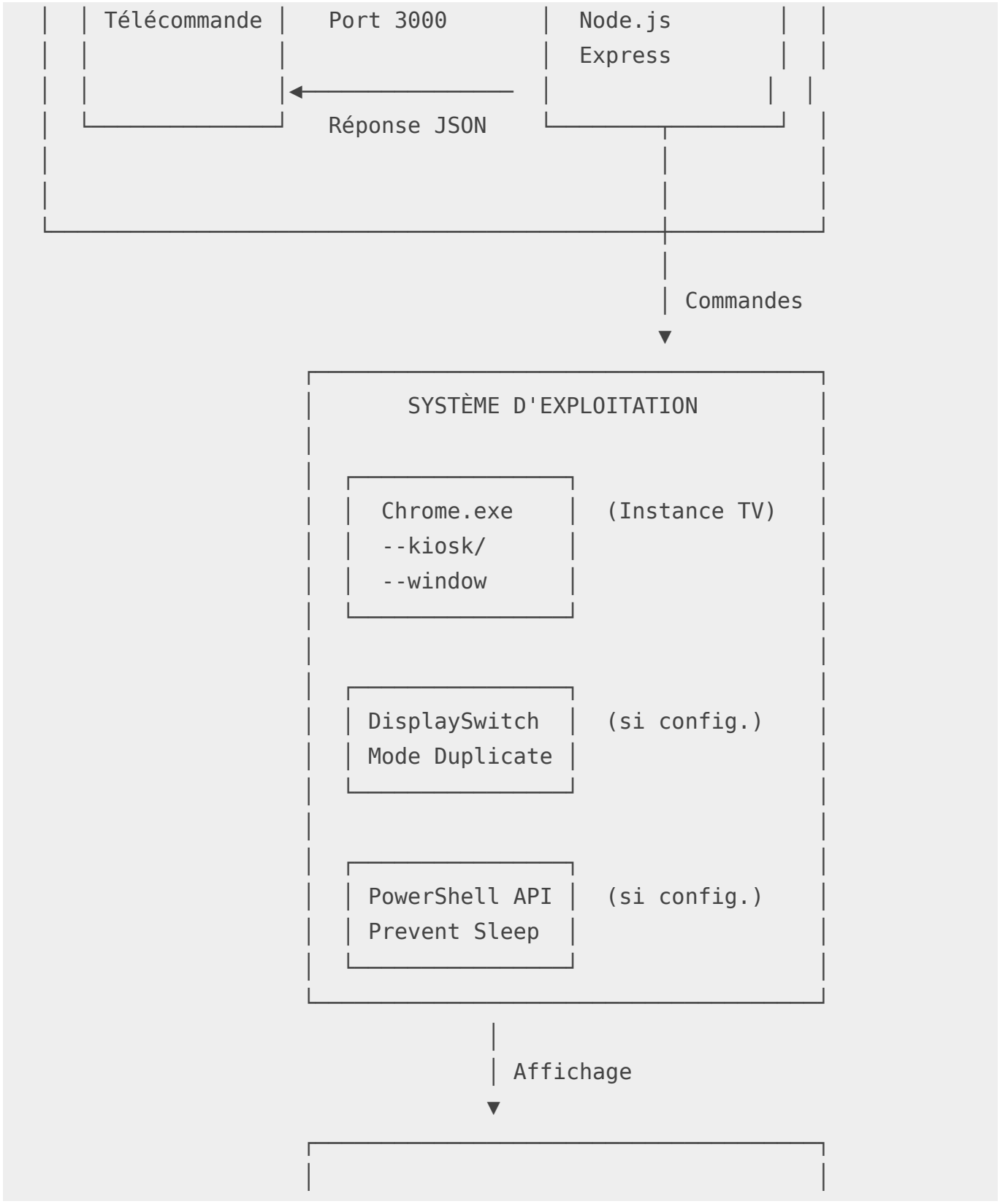
Fonctionnement

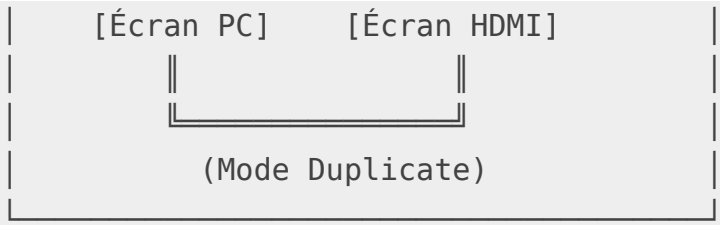
Le serveur Node.js expose une API REST accessible depuis le réseau local. L'interface web, optimisée pour mobile, communique avec le serveur pour contrôler une instance Chrome en mode kiosk qui diffuse les flux vidéo.

Le choix de Puppeteer-core plutôt qu'un simple spawn de processus permet d'obtenir des transitions instantanées entre chaînes (environ 500ms) grâce à la navigation programmée dans la même instance de navigateur.

```
// Exemple simplifié de changement de chaîne
async function navigateToChannel(url) {
    await page.goto(url, {
        waitUntil: 'networkidle2',
        timeout: 30000
    });
}
```




Télécommande
Port 3000
Node.js
Express
Réponse JSON
Commandes
SYSTÈME D'EXPLOITATION
Chrome.exe
--kiosk/
--window
(Instance TV)
DisplaySwitch
Mode Duplicate
(si config.)
PowerShell API
Prevent Sleep
(si config.)
Affichage



Fonctionnalités principales

Contrôle à distance

L'interface mobile propose :

- Sélection parmi 28 chaînes TNT gratuites
- Contrôle du volume système (4 niveaux : 0%, 30%, 50%, 75%)
- Démarrage et arrêt du mode TV
- État en temps réel de l'application

Optimisations système

Plusieurs mécanismes ont été mis en place pour une expérience optimale :

Gestion d'affichage : Basculement automatique en mode « duplicate » via DisplaySwitch.exe pour projeter sur un second écran si nécessaire.

Audio : Sélection automatique du périphérique audio configuré et activation du son en cas de désactivation accidentelle.

Fenêtrage : La fenêtre Chrome est définie en « always on top » (HWND_TOPMOST) via l'API Windows pour rester visible en permanence.

Énergie : Désactivation de la mise en veille pendant la diffusion via SetThreadExecutionState.

Performance

- Changement de chaîne : environ 500ms (navigation simple)
- Pas d'interruption visuelle (pas de flash noir)

Défis techniques rencontrés

Gestion du focus fenêtre

Le principal défi a été de maintenir la fenêtre Chrome au premier plan de manière fiable. Plusieurs approches ont été testées :

1. SetForegroundWindow : Insuffisant en mode kiosk, le focus était parfois perdu
2. Minimisation des autres fenêtres : Fonctionnel mais perturbant pour le workflow
3. HWND_TOPMOST : Solution finale retenue, combinée au mode kiosk

Élimination des popups indésirables

Chrome affiche par défaut plusieurs éléments gênants en automatisation :

- Barre « Chrome est contrôlé par un logiciel de test automatisé »
- Popups de traduction automatique
- Détection du flag navigator.webdriver

Ces éléments ont été désactivés via une combinaison d'arguments Chrome (- -

`exclude-switches=enable-automation`), de préférences de profil, et d'injection JavaScript (`navigator.webdriver = undefined`).

Gestion de l'état

Le système maintient la cohérence entre l'état du serveur et celui de l'interface :

- Vérification périodique du statut (polling toutes les 5 secondes)
- Synchronisation automatique en cas de déconnexion/reconnexion
- Nettoyage proper des ressources à l'arrêt

Progressive Web App

L'application est configurée comme PWA (Progressive Web App) avec :

- Manifest.json pour l'installation sur l'écran d'accueil
- Icônes adaptatives (192×192 et 512×512)
- Mode standalone (plein écran sans barre d'adresse)
- Méta-tags pour iOS et Android

Cela permet une expérience proche d'une application native lorsqu'elle est ajoutée à l'écran d'accueil du smartphone.

Installation et déploiement

Le déploiement est volontairement simple :

```
# Installation des dépendances
```

```
npm install puppeteer-core

# Configuration (fichier JSON)
# - Liste des chaînes
# - Paramètres système (audio, display, etc.)

# Lancement
node tv-remote-server.js
```

L'application écoute sur le port 3000 et est accessible depuis n'importe quel appareil du réseau local. Pour une utilisation régulière, un service Windows ou un script de démarrage automatique peut être configuré.

Limitations et améliorations futures

Limitations actuelles

- Fonctionne uniquement sur Windows (dépendances système spécifiques)
- Nécessite Chrome installé sur le système
- Limité aux chaînes gratuites disponibles sur tv.free.fr
- Pas de gestion multi-utilisateurs

Pistes d'amélioration

Plusieurs évolutions pourraient être envisagées :

- Support multi-plateforme (Linux, macOS)
- Enregistrement de programmes
- Programmation d'enregistrements

- Picture-in-Picture
- Contrôle vocal
- Historique de visionnage

Conclusion

Ce projet illustre comment une combinaison de technologies web standards et d'APIs système peut produire une solution fonctionnelle et performante pour un besoin spécifique. L'approche modulaire et l'utilisation de Puppeteer offrent une flexibilité importante pour des évolutions futures.

Le code source complet et la documentation technique sont disponibles sur demande pour les personnes intéressées par les aspects d'implémentation détaillés.

Développé en février 2026
Stack : Node.js, Express, Puppeteer-core, Vanilla JavaScript
Plateforme : Windows 10/11

Author

Patrick

GPM Factory