

# Cahier des Charges : Application de Transport

## 1. Contexte et Objectif

Développement d'une application de gestion de transport (réservation, suivi, gestion des trajets) basée sur une architecture client-serveur. Le serveur Node.js sera hébergé sur Windows, tandis que la base de données sera hébergée sur Linux.

## 2. Architecture Technique

### 2.1 Serveur

Technologie : Node.js avec Express.js

Système d'exploitation : Windows Server

Fonctionnalités :

- API REST pour la gestion des trajets, utilisateurs, véhicules
- Authentification (JWT ou OAuth2)
- Journalisation des requêtes (logging)
- Sécurité (CORS, HTTPS, validation des entrées)

### 2.2 Base de Données

Système : Linux (Ubuntu Server)

SGBD : PostgreSQL ou MySQL

Accès : via connexion sécurisée (SSH ou VPN)

Structure :

- Table utilisateurs
- Table trajets
- Table véhicules
- Table réservations

### 2.3 Client

Technologie : React.js / Vue.js ou application Electron

Fonctionnalités :

- Interface utilisateur intuitive
- Visualisation des trajets
- Réservation en ligne
- Suivi en temps réel (si applicable)

## 3. Fonctionnalités Principales

### 3.1 Gestion des Utilisateurs

- Inscription / Connexion
- Rôles : Admin, Chauffeur, Client

### 3.2 Gestion des Trajets

- Création, modification, suppression
- Affectation des véhicules et chauffeurs

### **3.3 Réservation**

- Réservation par les clients
- Confirmation et suivi

### **3.4 Reporting**

- Statistiques sur les trajets
- Historique des réservations

## **4. Sécurité**

- Chiffrement des données sensibles
- Authentification sécurisée
- Protection contre les injections SQL et XSS

## **5. Déploiement**

- Serveur Node.js : déployé sur Windows avec PM2 ou IIS Node
- Base de données : hébergée sur Ubuntu avec accès distant sécurisé
- Client : déployé sur navigateur ou packagé en application desktop

## **6. Maintenance et Évolutivité**

- Documentation technique
- Tests unitaires et d'intégration
- Possibilité d'ajouter des modules (paiement, géolocalisation, etc.)