Configurateur de Voiture par Contraintes (CSP)

Ce projet interactif utilise la programmation par contraintes (CSP) pour configurer des voitures, garantissant la compatibilité des choix en temps réel.

NAJMITDINOV Alekseï, TIENDJEU NGALEU Yannick, SELVA Vicram

Vue d'Ensemble du Projet

Objectif Principal

Implémenter un configurateur de voiture interactif avec Google OR-Tools CP-SAT.

Fonctionnalité Clé

Permettre la sélection d'options (modèle, moteur, couleur) tout en respectant les contraintes de compatibilité.

Bénéfice Pédagogique

Illustrer la résolution de problèmes de configuration complexes en temps réel par CSP.

Architecture Technique

Stack Technologique

- Backend: Python + Google OR-Tools CP-SAT
- Frontend: HTML5 + JavaScript
- Communication: API REST JSON via CORS

JS

Google OR-Tools

Variables de Configuration

Le configurateur gère 7 dimensions clés pour la personnalisation du véhicule.

1

Modèle de Voiture

5 options (Civic, Golf, 330i, X3, Mustang)

2

Moteur

5 options (Essence 1.5L, 2.0L, 3.0L, Diesel 2.0L, Hybride 2.0L)

3

Transmission

2 options (Manuelle 6 vitesses, Automatique 8 vitesses)

4

Drivetrain

3 options (FWD, RWD, AWD)

5

Couleur

6 options (Blanc, Noir, Argent, Bleu, Rouge, Gris)

6

Intérieur

3 options (Tissu, Cuir classique, Cuir premium Nappa)

7

Pack d'Équipement

4 options (Base, Sport, Luxe, AMG Performance)

Espace de recherche total: 10 800 combinaisons candidates, Nombre de configurations valides : **1 416**

Contraintes Métier Implémentées

Le système intègre 12 contraintes fonctionnelles complexes pour assurer la compatibilité.

1

Contraintes par Modèle

Ex: Honda Civic → Moteurs 1.5L, 2.0L, Hybride uniquement; FWD obligatoire.

2

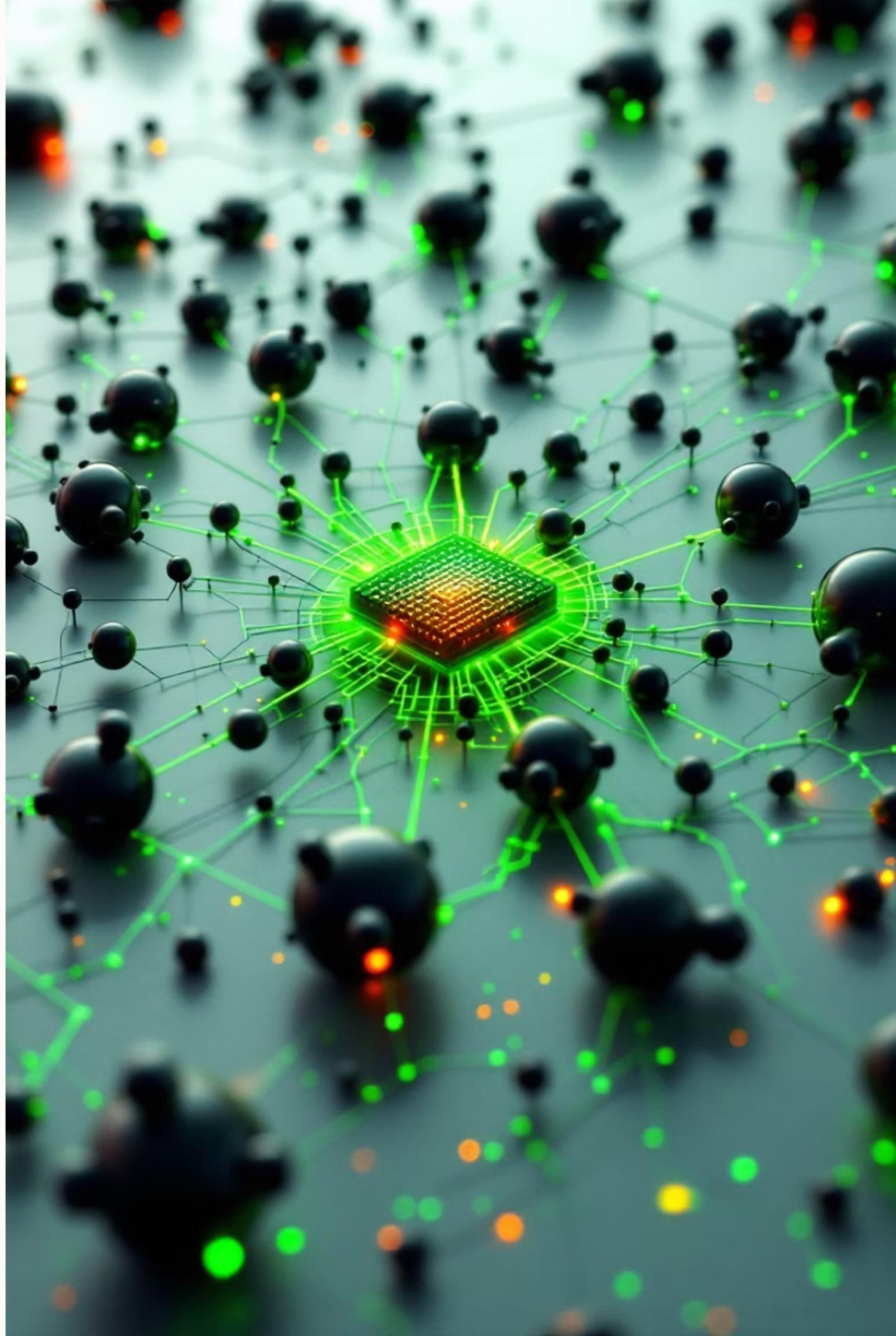
Contraintes Globales

Ex: Moteur 3.0L exclusif aux BMW X3 et Mustang; Pack AMG uniquement pour BMW 330i et X3.

3

Incompatibilités

Ex: Cuir premium incompatible avec Pack Base; AWD toujours en Automatique.



Algorithme de Résolution: CP-SAT d'OR-Tools

1

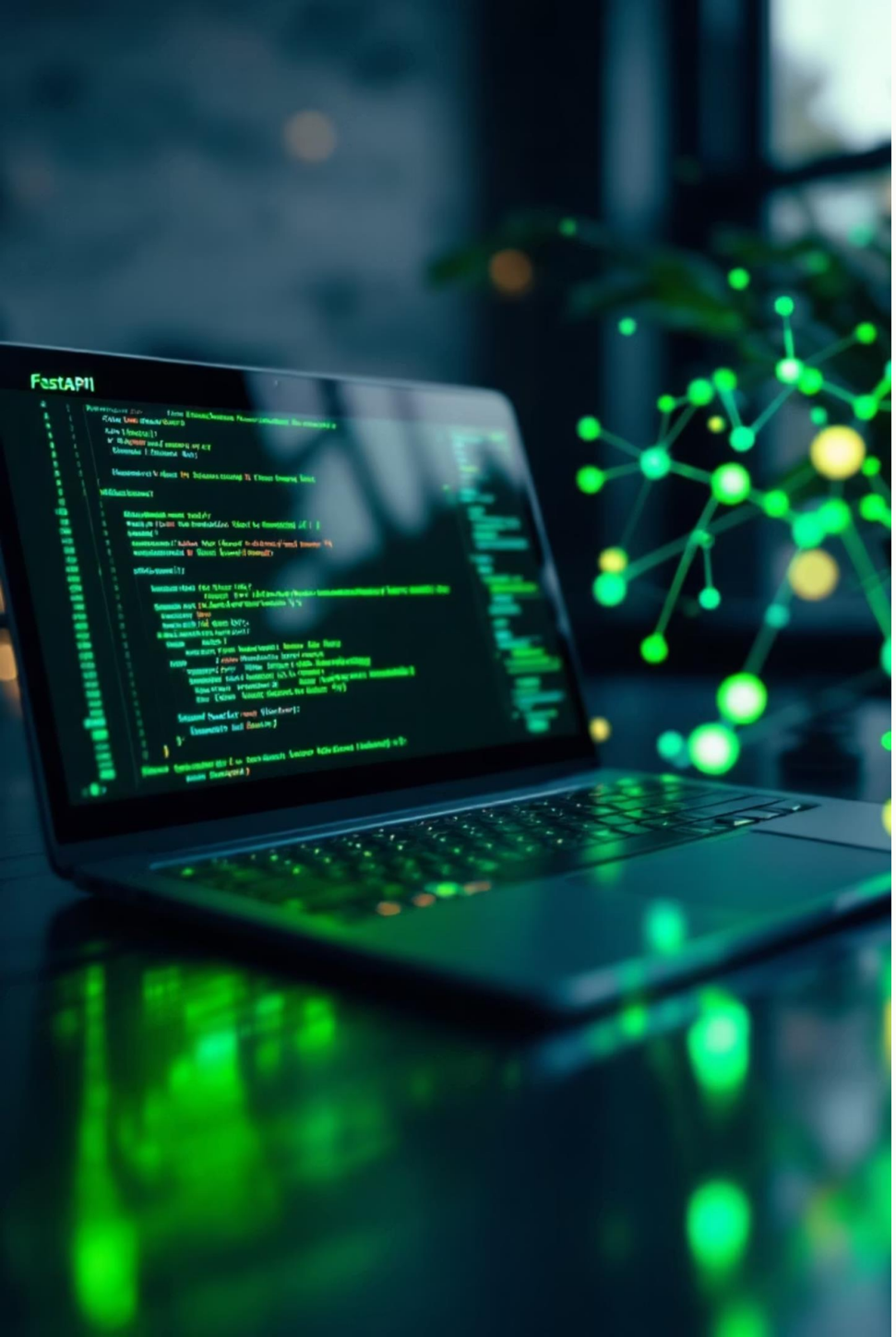
Propagation des Domaines

Détermine les valeurs possibles pour chaque variable après des choix partiels.

2

Résolution Complète

Trouve une configuration valide et complète en utilisant les optimisations d'OR-Tools



Implémentation Backend (FastAPI)

Structure du Code (solver.py)

- Définition des domaines des variables.
- Fonction de construction du modèle CSP.
- Fonctions pour la propagation des domaines et la résolution complète.

Endpoints API REST

- **/propagate (POST):** Retourne les domaines réduits et la validité.
- **/solve (POST):** Trouve et retourne une configuration complète valide.
- **/ping (GET):** Healthcheck de l'API.

Implémentation Frontend (JavaScript)

L'interface utilisateur offre une expérience intuitive et réactive pour la configuration.

Fonctionnalités UI

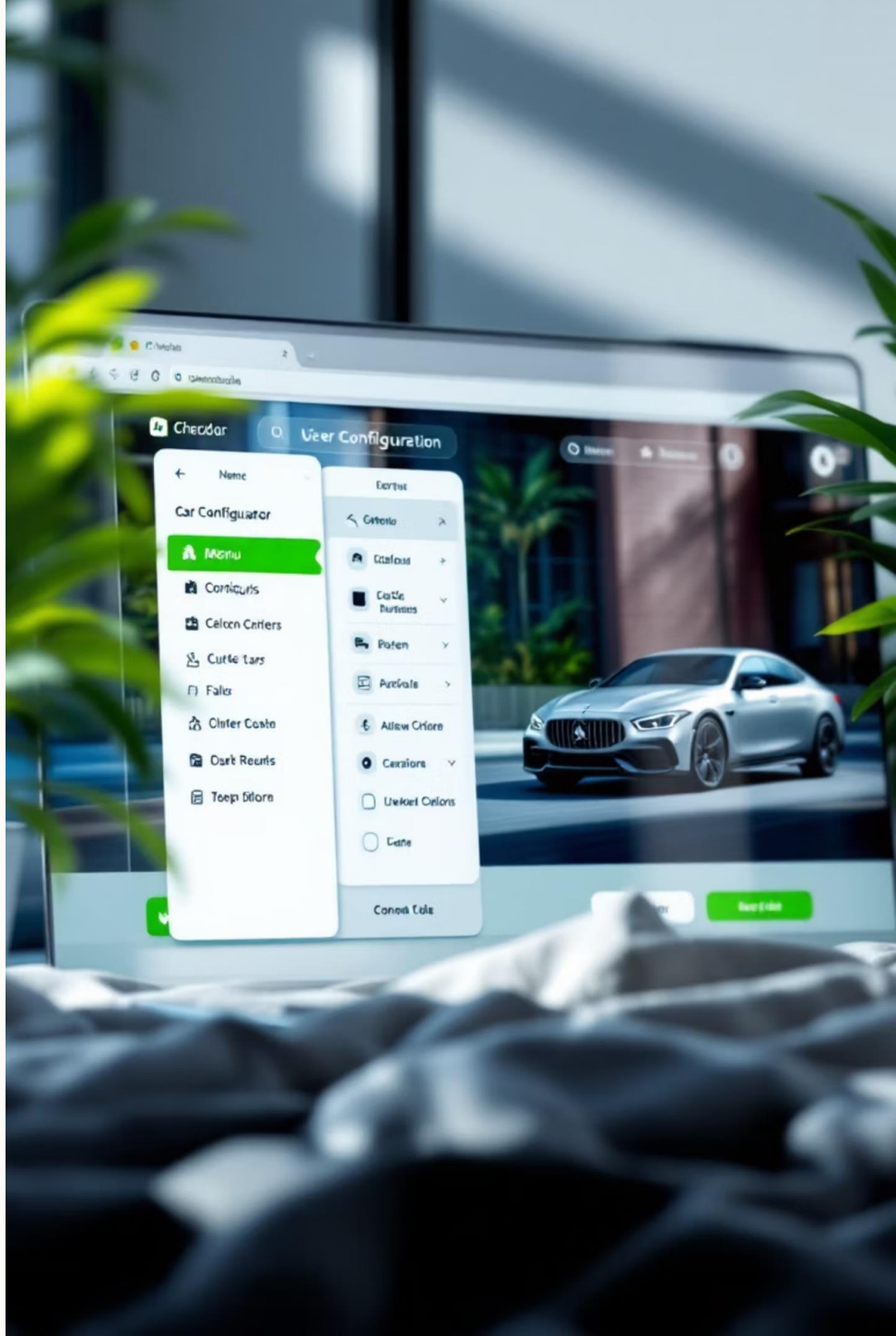
7 sélecteurs HTML, désactivation dynamique, 5 presets, boutons.

Mise à Jour Dynamique

Appels API pour propager les contraintes et mettre à jour les options disponibles.

Indicateur de Validité

Affiche si la configuration est compatible ou incompatible.



Conclusion: La Puissance du CSP

Ce projet démontre l'efficacité de la programmation par contraintes pour la configuration de produits complexes.

Correction

Toute configuration proposée est garantie valide.

Complétude

Exploration systématique de l'espace de solution.

Performance

Réponse en temps réel malgré la complexité.

Scalabilité

Facilité d'extension à de nouvelles variables/contraintes.

Utilisabilité

Guidage interactif pour l'utilisateur.