API – Trouve ton artisan

Backend REST développé avec Node.js, Express et MySQL.

Cette API permet de fournir les données nécessaires au frontend (catégories, artisans, recherche, fiche détail).

🚀 Stack technique

- Node.js
- Express
- MySQL / MariaDB
- mysql2 (promise)
- dotenv
- cors

📂 Architecture

backend/ src/ app.js # Configuration de l'application Express server.js # Point d'entrée du serveur db/ pool.js # Configuration du pool MySQL routes/ # Déclaration des routes API controllers/ # Logique métier et requêtes SQL

⚙️ Installation

Depuis le dossier backend :

```
npm install
🔐 Configuration
Créer un fichier .env basé sur .env.example.
```

Exemple :

```
PORT=3001
DB_HOST=localhost
DB_USER=app_trouve_artisan
DB_PASSWORD=CHANGE_ME
DB_NAME=trouve_ton_artisan
⚠️ Le fichier .env ne doit jamais être versionné.
```

▶️ Lancer l'API

En développement :

```
npm run dev
```

En production :

```
npm start
```

♡ Endpoint de test

GET /health Permet de vérifier que l'API est opérationnelle.

Réponse :

```
{
  "status": "API OK"
}
```

Endpoints disponibles

GET /api/categories Retourne la liste des catégories triées par ordre alphabétique.

Exemple de réponse :

```
[
  {
    "id": 1,
    "name": "Alimentation"
  },
  {
    "id": 2,
    "name": "Bâtiment"
  }
]
```

Sécurité

Utilisation d'un utilisateur MySQL dédié à l'application.

Principe du moindre privilège (SELECT, INSERT, UPDATE, DELETE uniquement).

Séparation des secrets via variables d'environnement.

Aucun mot de passe réel versionné dans le dépôt.

Bonnes pratiques appliquées

Séparation app / server

Pool de connexions MySQL

Architecture routes / controllers

Gestion des erreurs SQL

Standardisation charset utf8mb4 + collation utf8mb4_unicode_ci

🤔 Évolutions prévues

Endpoint artisans mis en avant

Recherche dynamique (nom, spécialité, ville)

Documentation Swagger (OpenAPI)

Validation des données (middleware)