

Trouve ton artisan


Projet fullstack réalisé dans le cadre de ma formation développeur web.

L'objectif de l'application est de permettre aux utilisateurs de trouver facilement un artisan local à partir de sa catégorie ou d'une recherche.

Fonctionnalités principales :

- consulter les artisans par catégorie
- afficher les artisans mis en avant
- rechercher un artisan (nom, ville, spécialité)
- consulter la fiche détaillée d'un artisan
- contacter un artisan via un formulaire

État d'avancement du projet

- ☒ Maquettage UX/UI (Figma : desktop / tablette / mobile)
- ☒ Modélisation base de données (MCD / MLD / EER)
- ☒ Scripts SQL (création, seed, tests)
- ☒ API REST Node.js / Express
- ☒ Documentation API Swagger (OpenAPI)
-  Frontend React (en cours)

Stack technique

Backend

- Node.js
- Express
- MySQL / MariaDB
- mysql2
- Swagger UI (OpenAPI)
- dotenv
- cors
- morgan (logs HTTP)

Base de données

- MySQL / MariaDB
- charset : `utf8mb4`
- collation : `utf8mb4_unicode_ci`

Frontend

- React (en cours de développement)

Structure du projet

```
|— backend/ # API Express + documentation Swagger
|— docs/ # rapport de projet et documentation
|— frontend/ # application React
|— sql/ # scripts SQL (database, schema, seed)
|— tests/ # requêtes de test SQL et exports commentés
|— README.md
```

Installation du projet

1 Base de données

Les scripts SQL se trouvent dans le dossier :

- sql/

Ordre d'exécution recommandé :

- 1 00_create_database.sql
- 2 01_create_user_and_grants.sql
- 3 02_schema.sql
- 4 03_seed.sql
- 5 04_tests.sql

2 Lancer l'API backend

```
cd backend
npm install
npm run dev
```

API disponible sur :

<http://localhost:3001>

Endpoint de test :

<http://localhost:3001/health>

Documentation Swagger :

<http://localhost:3001/api-docs>

Endpoints principaux

- **Catégories:**
 - GET /api/categories
 - GET /api/categories/:id/artisans
 - **Artisans:**
 - GET /api/artisans/featured
 - GET /api/artisans/:id
 - GET /api/artisans?search=...
-

Sécurité

- utilisateur MySQL dédié à l'application
 - principe du moindre privilège
 - secrets stockés dans .env (non versionné)
 - gestion centralisée des erreurs API
-

Fonctionnalités backend

- API REST Express
- pagination des résultats
- logs HTTP avec Morgan
- validation des paramètres (middleware)
- gestion centralisée des erreurs

Auteur

Loïc

Projet réalisé dans le cadre d'une formation développeur web.