Devoir bilan trouve ton artisan

Documentation Technique

Table des matières

1. Présentation du projet

Trouve ton artisan est une plateforme web développée pour la région Auvergne-Rhône-Alpes permettant aux particuliers de trouver et contacter des artisans qualifiés dans leur région.

> **Mission :** Créer une interface intuitive et accessible permettant de mettre en relation les particuliers avec les artisans de la région, classés par catégories et spécialités.

Contexte

La région Auvergne-Rhône-Alpes souhaite valoriser le savoir-faire de ses artisans locaux en offrant une plateforme moderne et accessible à tous. Ce projet s'inscrit dans une démarche de digitalisation des services régionaux et de soutien à l'économie locale.

Public cible

- **Particuliers** : recherche d'artisans pour des travaux, services ou achats
- **Artisans** : visibilité accrue auprès des habitants de la région
- **Région** : mise en valeur du tissu économique local

2. Maquettes Figma

Les maquettes du projet ont été réalisées sur **Figma**, outil de design collaboratif permettant de concevoir les interfaces utilisateur et de partager les prototypes avec l'équipe de développement.

> 🎨 **Accès aux maquettes :**
> **https://www.figma.com/design/cIWMzWMWhwhUYBmtgqdak6/DEV-Trouve-ton-artisan**

Écrans disponibles

Version Desktop

- **Page d'accueil** : Hero, section "Comment trouver mon artisan", Top 3 artisans, catégories
- **Liste des artisans** : Affichage par catégorie avec filtres
- **Fiche artisan** : Détails complets avec formulaire de contact
- **Pages légales** : Mentions légales, Données personnelles, Accessibilité, Gestion des cookies

Version Mobile (Responsive)

- Adaptation complète de toutes les pages pour écrans mobiles
- Menu hamburger pour la navigation
- Cartes artisans empilées verticalement
- Formulaires optimisés pour le tactile

Aperçu des maquettes

Version Desktop

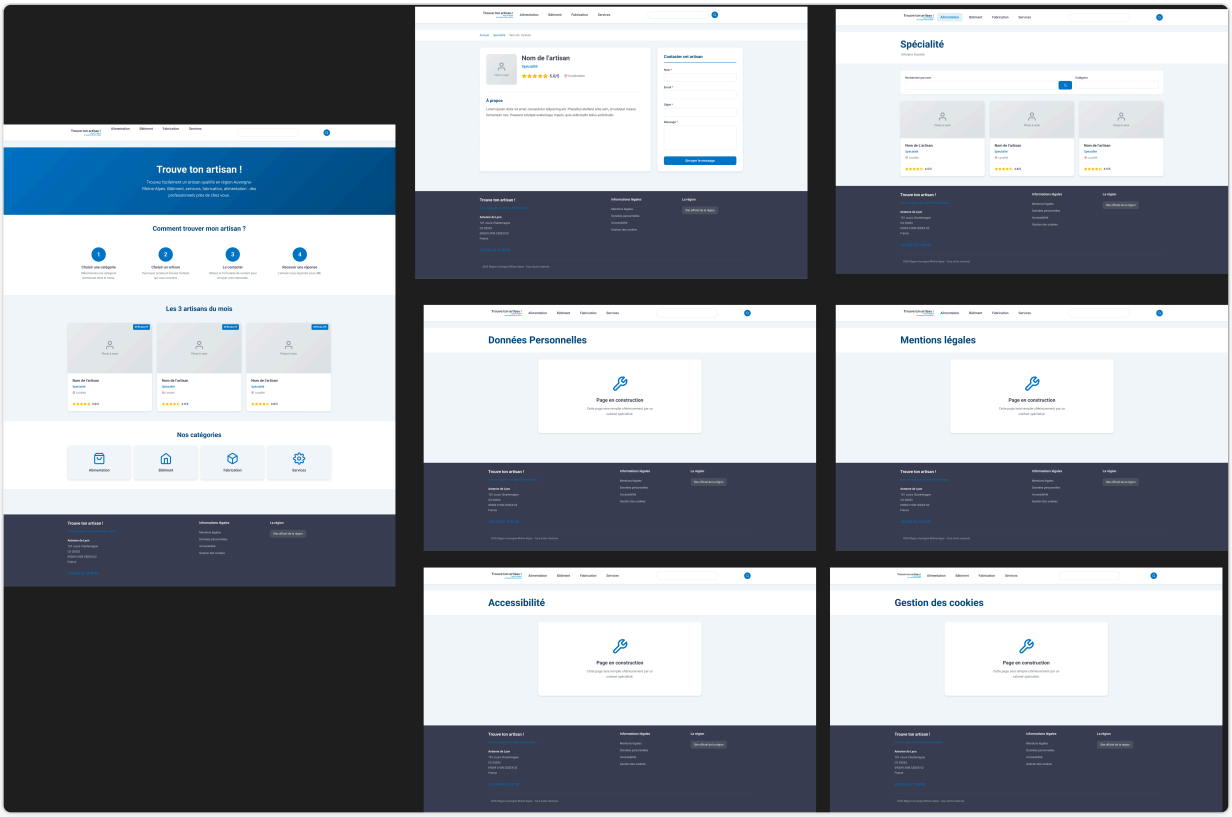


Figure 1 : Aperçu des maquettes desktop (Page d'accueil, Fiche artisan, Liste par catégorie, Pages légales)

Version Mobile

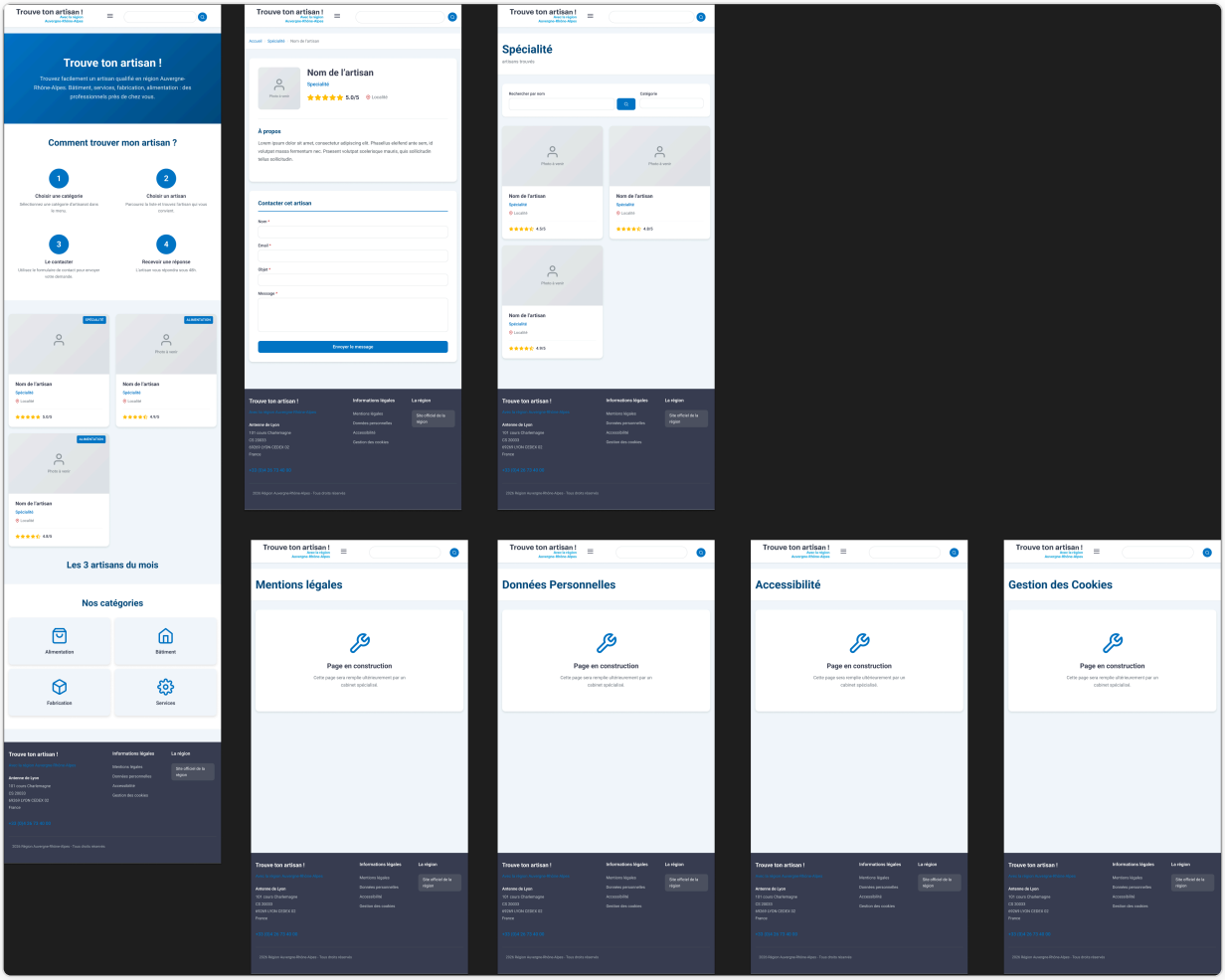


Figure 2 : Aperçu des maquettes mobile responsive

3. Objectifs et fonctionnalités

Objectifs principaux

1. Permettre aux particuliers de rechercher un artisan par catégorie ou par nom
2. Consulter une fiche artisan détaillée avec note, spécialité et localisation
3. Contacter l'artisan via un formulaire sécurisé
4. Garantir une accessibilité conforme aux normes WCAG 2.1
5. Assurer un design responsive (Mobile First)

Fonctionnalités implémentées

Frontend

Fonctionnalité	Description
Page d'accueil	Présentation du service avec section "Comment trouver mon artisan"
Artisans du mois	Affichage des 3 artisans mis en avant (Top artisans)
Navigation par catégories	4 catégories : Bâtiment, Services, Fabrication, Alimentation
Recherche	Recherche d'artisans par nom
Fiches artisans	Page détaillée avec notation par étoiles, contact, site web
Formulaire de contact	Envoi d'email sécurisé à l'artisan
Page 404	Page d'erreur personnalisée
Mentions légales	Page des mentions légales et CGU

Backend

Fonctionnalité	Description
API REST	Endpoints pour catégories, artisans et contact
Authentification API	Protection par clé API
Validation	Validation des entrées avec express-validator
Envoi d'emails	Nodemailer avec support SMTP
Sécurité	Helmet, CORS, Rate limiting, XSS protection

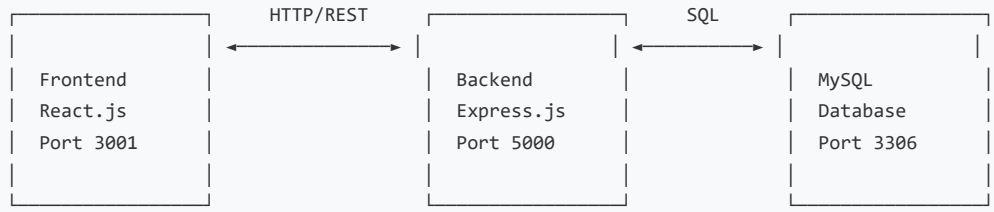
4. Technologies utilisées

Couche	Technologies principales
Frontend	React.js 18.2, React Router 6.20, Bootstrap 5.3, Sass, Axios
Backend	Node.js 18+, Express.js 4.18, Sequelize 6.35 (ORM)
Base de données	MySQL 8.0+
Sécurité	Helmet, CORS, express-rate-limit, express-validator, XSS
Autres	Nodemailer (emails), React Helmet Async (SEO), React Icons

5. Architecture du projet

Architecture globale

> Le projet suit une architecture **Client-Serveur** classique avec séparation claire entre le frontend (React) et le backend (Node.js/Express).



Structure des dossiers

Backend

```
backend/
├── config/
│   └── database.js          # Configuration Sequelize/MySQL
├── controllers/
│   ├── artisanController.js # Logique métier artisans
│   ├── categorieController.js # Logique métier catégories
│   └── contactController.js # Logique envoi emails
├── middleware/
│   ├── apiKeyAuth.js        # Authentification par clé API
│   └── validateRequest.js   # Validation des requêtes
├── models/
│   ├── index.js             # Export des modèles
│   ├── Artisan.js           # Modèle Artisan
│   ├── Categorie.js         # Modèle Catégorie
│   └── Specialite.js        # Modèle Spécialité
├── routes/
│   ├── artisanRoutes.js     # Routes /api/artisans
│   ├── categorieRoutes.js   # Routes /api/categories
│   └── contactRoutes.js     # Routes /api/contact
├── scripts/
│   └── sql/
│       ├── create_database.sql  # Création BDD
│       └── seed_database.sql    # Données initiales
├── .env                     # Variables d'environnement
├── package.json
└── server.js                # Point d'entrée
```

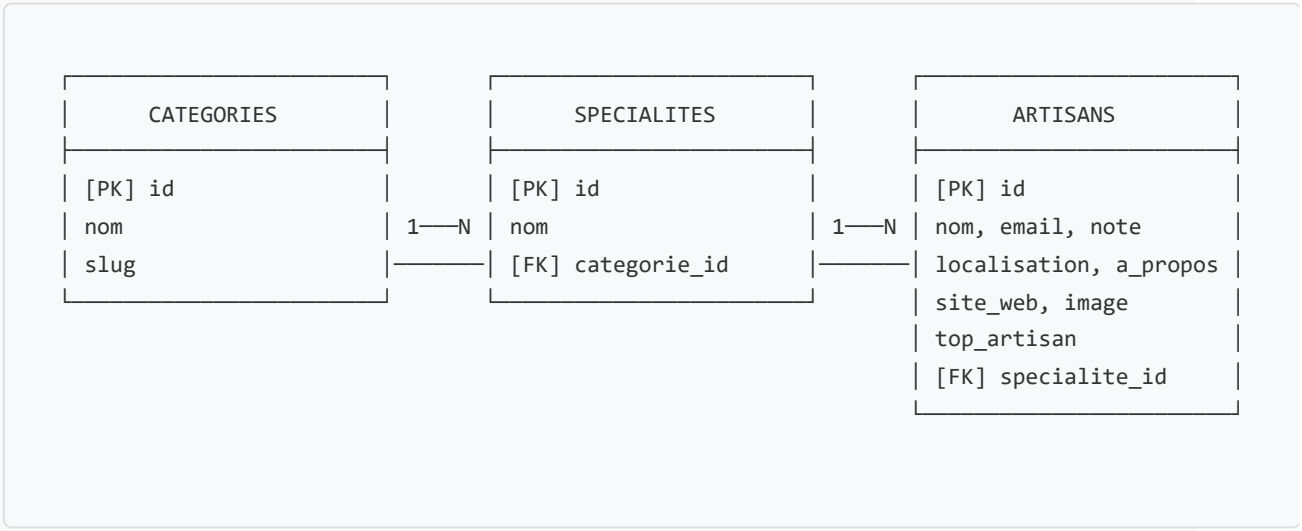
Frontend

```
frontend/
├── public/
│   ├── index.html           # Template HTML
│   ├── manifest.json        # PWA manifest
│   └── favicon.ico          # Favicon
├── src/
│   ├── components/
│   │   ├── common/          # Composants réutilisables
│   │   │   ├── ArtisanCard.jsx
│   │   │   ├── Loader.jsx
│   │   │   └── StarRating.jsx
│   │   └── layout/          # Composants de mise en page
│   │       ├── Header.jsx
│   │       └── Footer.jsx
│   ├── pages/
│   │   ├── HomePage.jsx
│   │   ├── ArtisansListPage.jsx
│   │   ├── ArtisanDetailPage.jsx
│   │   ├── LegalPage.jsx
│   │   └── NotFoundPage.jsx
│   ├── services/
│   │   └── api.js           # Client Axios
│   ├── styles/
│   │   ├── _variables.scss  # Variables SCSS
│   │   ├── _mixins.scss     # Mixins SCSS
│   │   └── global.scss      # Styles globaux
│   ├── App.js               # Composant principal
│   └── index.js             # Point d'entrée
├── .env
└── package.json
```

6. Base de données

Modèle de données

La base de données MySQL comprend 3 tables liées par des relations 1:N :



Notation textuelle

```
CATEGORIES (id, nom, slug)

SPECIALITES (id, nom, #categorie_id)

ARTISANS (id, nom, email, note, localisation, a_propos, site_web, image, top_artisan, #specialite_id)

souligné = clé primaire | # = clé étrangère
```

Règles de gestion

- Une catégorie contient plusieurs spécialités (1:N)
- Un artisan exerce une seule spécialité (N:1)
- La note d'un artisan est comprise entre 0 et 5
- Un artisan peut être désigné "Top artisan" du mois

Données initiales

4 catégories : Bâtiment, Services, Fabrication, Alimentation

17 artisans dont 3 Top artisans mis en avant sur la page d'accueil.

7. Documentation API

> **Base URL :** `http://localhost:5000/api`
> **Authentification :** Header `x-api-key` requis

Endpoints disponibles

Méthode	Endpoint	Description
GET	/api/categories	Liste des catégories avec spécialités
GET	/api/categories/:slug/artisans	Artisans d'une catégorie
GET	/api/artisans	Liste tous les artisans
GET	/api/artisans/top	Top artisans du mois
GET	/api/artisans/search?q=query	Recherche par nom
GET	/api/artisans/:id	Détails d'un artisan
POST	/api/contact	Envoi d'email à un artisan

8. Guide d'installation

Prérequis

- **Node.js** version 18 ou supérieure
- **npm** (inclus avec Node.js)
- **MySQL** version 8.0 ou supérieure
- **Git**

Étapes d'installation

1. Cloner le repository

```
git clone https://github.com/votre-username/trouve-ton-artisan.git
cd trouve-ton-artisan
```

2. Installer les dépendances

```
# Backend
cd backend
npm install

# Frontend
cd ../frontend
npm install
```

3. Configurer la base de données

```
# Se connecter à MySQL et exécuter les scripts
mysql -u root -p

SOURCE backend/scripts/sql/create_database.sql;
SOURCE backend/scripts/sql/seed_database.sql;
```

4. Configurer les variables d'environnement

Créer un fichier `.env` dans le dossier backend :

```
PORT=5000
NODE_ENV=development
DB_HOST=localhost
DB_PORT=3306
DB_NAME=trouve_ton_artisan
DB_USER=root
DB_PASSWORD=votre_mot_de_passe
FRONTEND_URL=http://localhost:3001
```

```
API_KEY=votre_cle_api_secrete
```

Créer un fichier `.env` dans le dossier frontend :

```
REACT_APP_API_URL=http://localhost:5000/api
REACT_APP_API_KEY=votre_cle_api_secrete
REACT_APP_SITE_NAME=Trouve ton artisan
```

5. Lancer l'application

```
# Terminal 1 - Backend
cd backend
npm run dev

# Terminal 2 - Frontend
cd frontend
npm start
```

> **✓ Application accessible sur :**
> Frontend : `http://localhost:3001`
> Backend API : `http://localhost:5000/api`

9. Sécurité

- **Headers HTTP sécurisés** : Helmet.js (protection XSS, clickjacking, MIME sniffing)
- **CORS** : Restriction des origines autorisées
- **Rate Limiting** : 100 requêtes max par IP / 15 min
- **Validation** : express-validator pour toutes les entrées
- **Protection SQL Injection** : Requêtes préparées via Sequelize ORM
- **Authentification API** : Clé API requise dans les headers
- **Variables d'environnement** : Données sensibles dans fichiers .env (exclus de Git)

10. Accessibilité (WCAG 2.1)

- **Navigation clavier** : Focus visible, tabindex appropriés
- **Formulaires** : Labels associés (for/id, aria-label)
- **Contrastes** : Conformité WCAG AA
- **Images** : Attributs alt descriptifs
- **Structure** : Hiérarchie des titres h1 > h2 > h3
- **ARIA** : Landmarks (banner, navigation, main)
- **Skip links** : Accès rapide au contenu principal
