Projet Artisans – Frontend + Backend

Page de garde

- **Titre du projet** : Projet Artisans – Frontend + Backend
- **Nom de l’étudiante** : Manuela Pinera
- **Date de remise** : [14/11/2025]

Sommaire

1. Contexte du projet

Présentation de l’entreprise / du projet :

Le projet consiste à développer une application web pour gérer et présenter les artisans d’une région donnée. L’objectif est de permettre aux utilisateurs de rechercher facilement des artisans selon leur métier, leur nom ou leur spécialité, et d’accéder à une fiche détaillée pour chaque artisan. Le projet est réalisé dans le cadre d’un exercice de développement web, avec un focus sur la mise en place d’une base de données et d’une interface utilisateur interactive.

Expression des besoins :

Pour les utilisateurs :

Rechercher un artisan par nom ou par métier.
Consulter les informations détaillées de chaque artisan (nom, spécialité, ville, note, description, contact, site web).
Naviguer facilement entre les différentes catégories d’artisans.

Pour les administrateurs / développeurs :

Stocker et gérer les données des artisans dans une base MySQL sécurisée.
Assurer la communication entre le frontend React et le backend Flask via des API REST.
Garantir la sécurité des données et la robustesse de l’application.

Contraintes :

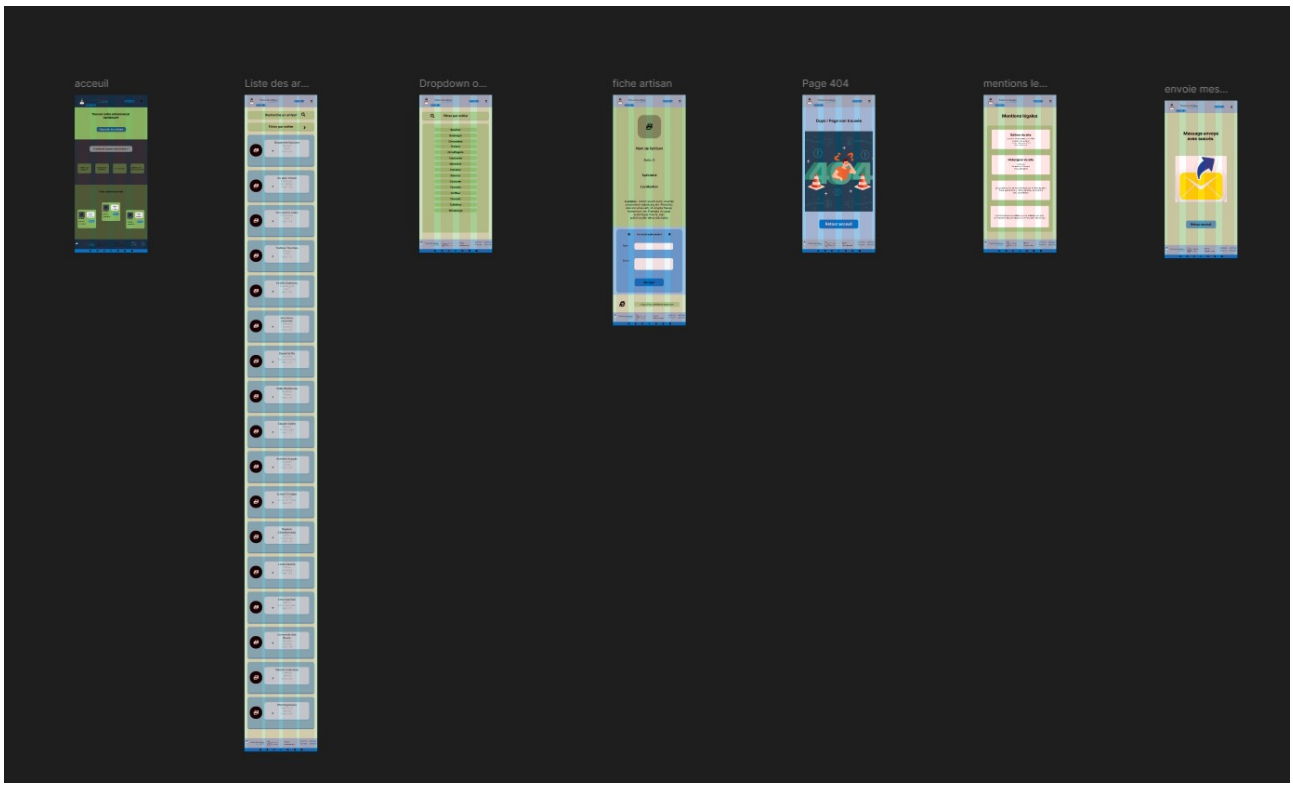
L’application doit être **responsive** et utilisable sur en mobile first et Desktop.
Les données doivent être sécurisées contre les attaques courantes (injections SQL, XSS, etc.).
Le projet doit être fonctionnel même en local, et le code doit être structuré et maintenable.
Les livrables doivent inclure le **frontend React**, le **backend Flask**, la **base MySQL**, ainsi que la **documentation du projet**.

Livrables attendus :

Application web complète avec interface React et backend Flask.
Base de données MySQL avec la table des artisans.
Documentation détaillant la structure de la base de données (MCD/MLD), les mesures de sécurité, et le fonctionnement de l’application.
Maquette du projet Figma .

2. Maquettes Figma

Capture d’écran des maquettes



LIEN maquettes figma :

https://www.figma.com/design/T9iSIw9vKwmfjJLAuvzO8h/Accueil?node-id=0-1&t=EVdakKhK2pFX98IQ-1

3. Présentation de la base de données

a) Modèle Conceptuel des Données (MCD)

Table	Champ	Type	Description
artisans	id	INT AUTO_INCREMENT PRIMARY KEY	Identifiant unique
artisans	nom	VARCHAR(100)	Nom de l'artisan
artisans	specialite	VARCHAR(100)	Métier ou spécialité
artisans	note	FLOAT	Note moyenne
artisans	ville	VARCHAR(100)	Localisation
artisans	a_propos	TEXT	Description de l'artisan
artisans	email	VARCHAR(100)	Email de contact
artisans	site_web	VARCHAR(255)	Site web éventuel
artisans	categorie	VARCHAR(50)	Catégorie
artisans	top	BOOLEAN	Mise en avant

b) Modèle Logique des Données (MLD)

- Description des relations, vues, clés primaires et étrangères.

Table : artisansReprend exactement le MCD décrit ci-dessus.

La table artisans_vue (vue SQL) simplifie certaines requêtes côté frontend pour l'affichage.

4. Sécurité mise en place

Authentification base → utilisateur dédié avec mot de passe sécurisé

- Connexion côté serveur → identifiants stockés côté backend
- Requêtes préparées → protection contre injection SQL
- Validation côté serveur → filtrage et contrôle des données
- CORS → accès contrôlé aux API
- Sanitisation des entrées → protection XSS
- Gestion mots de passe → hachage si comptes utilisateurs

Point de sécurité	Mise en œuvre	Intérêt
Authentification base	Création d'un utilisateur dédié avec mot de passe	Empêche l'accès non autorisé à la base
Connexion sécurisée coté serveur	Identifiants stockés dans le backend, jamais exposés au frontend	Évite la divulgation de mots de passe
Requêtes SQL paramétrées	Utilisation de paramètres db,query (,,,)	Protège contre les injections SQL
Contrôle des erreurs	Les erreurs SQL retournent des messages génériques au frontend	Ne révèle pas d'informations sensibles
CORS	Configuration CORS pour limiter l'accés aux origines autorisées	Limite l'accès depuis d'autres sites
Validation et sanitisation	Tous les champs récupérés depuis formulaires ou paramètres sont filtrés	Évite les injections de scripts ou code malveillant

5. Veille sur les vulnérabilités

Risques identifiés : injection SQL, XSS, CSRF, mots de passe faibles

- **Mesures prises** :
- Requêtes préparées
- Validation côté backend
- CORS limité
- Protection formulaires
- Stockage sécurisé des mots de passe
- Appels API HTTPS

Durant le développement, une veille sur les vulnérabilités courantes des applications React + Node/Express + PostgreSQL/MySQL a été réalisée.

Mesures mises en place

Prévention des injections SQL
Requêtes préparées (`db.query("SELECT * FROM artisans WHERE id=$1", [id])`)
Empêche la manipulation directe des requêtes SQL

Validation côté serveur
Toutes les données reçues sont vérifiées avant traitement
Assure cohérence et sécurité des informations
Contrôle d'accès via CORS
Limite l'accès aux seules origines autorisées
Protège les API contre les sites tiers malveillants
Protection des formulaires
Validation et filtrage côté serveur
Réduit les risques XSS et injection de code
Gestion sécurisée des mots de passe
Hachage prévu si comptes utilisateurs (ex. bcrypt)
Aucune information sensible exposée dans le frontend ou les logs
Sécurité côté frontend
Pas de stockage de données sensibles en clair
Appels API via HTTPS en production
Journalisation et suivi
Erreurs SQL et accès non autorisés sont logés
Permet de détecter rapidement toute activité suspecte

Intérêt : Ces pratiques garantissent la protection des données des artisans et utilisateurs, réduisent les risques de compromission de la base, et assurent la stabilité et l'intégrité du site.

6. GitHub et scripts SQL

- **Repository GitHub :**
- https://github.com/Manue70/Artisan

- **Contenu :**

- Code frontend et backend
- Script création base de données (.sql)
- Script alimentation base de données (.sql)

7. Lien du site en ligne

https://artisan-ik47.onrender.com/

8. Annexes : Rapports Lighthouse

Accueil

- Performance : 76
- Accessibilité : 91
- Best Practice : 100
- SEO : 92
- Capture écran rapport Lighthouse

Dropdown

- Performance : 99
- Accessibilité : 91
- Best Practice : 100
- SEO : 92
- Capture écran rapport Lighthouse

Message envoyé

- Performance : 93
- Accessibilité : 91
- Best Practice : 93
- SEO : 92
- Capture écran rapport Lighthouse

Fiche Artisan

- Performance : 87
- Accessibilité : 92
- Best Practice : 96
- SEO : 92
- Capture écran rapport Lighthouse

Liste Artisans

- Performance : 98
- Accessibilité : 91
- Best Practice : 100
- SEO : 92
- Capture écran rapport Lighthouse