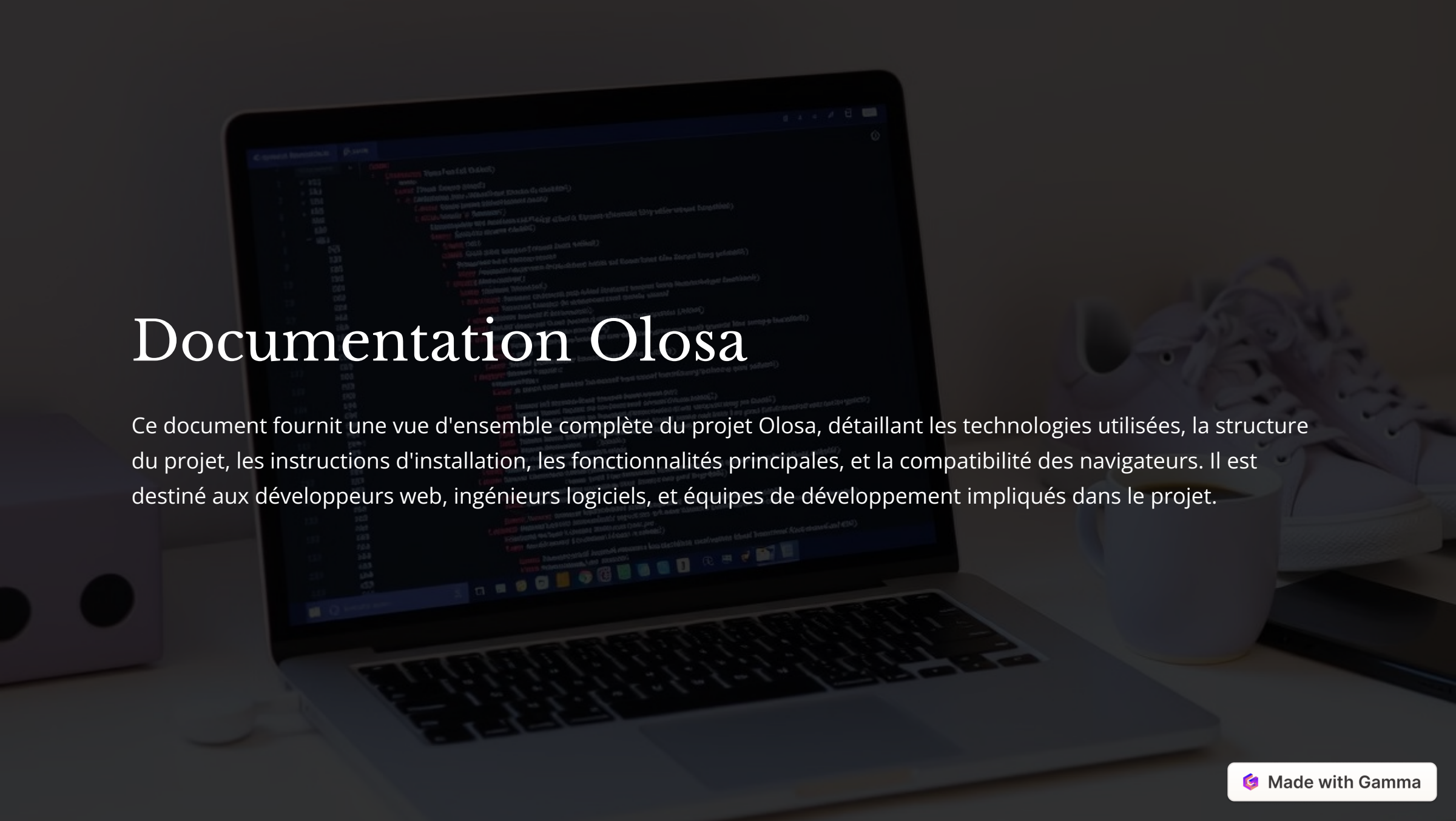

Documentation Olosa

Ce document fournit une vue d'ensemble complète du projet Olosa, détaillant les technologies utilisées, la structure du projet, les instructions d'installation, les fonctionnalités principales, et la compatibilité des navigateurs. Il est destiné aux développeurs web, ingénieurs logiciels, et équipes de développement impliqués dans le projet.

Technologies Utilisées et Justification

Ce projet e-commerce de chaussures repose sur un ensemble de technologies modernes, chacune ayant été soigneusement sélectionnée pour optimiser la performance, la maintenabilité et l'expérience utilisateur. Voici un aperçu des technologies clés et de leur rôle :

- **React 18.2.0 :** En tant que framework JavaScript principal, React offre une approche basée sur les composants pour la construction d'interfaces utilisateur interactives. Sa popularité, sa vaste communauté et son écosystème riche en bibliothèques en font un choix idéal pour le développement frontend.
- **Redux Toolkit & React-Redux :** Redux Toolkit simplifie la gestion de l'état global de l'application en fournissant des outils pour la création de slices Redux, la gestion des reducers et la configuration du store. React-Redux permet de connecter facilement les composants React au store Redux, facilitant ainsi la gestion des données à travers l'application.
- **Material-UI (MUI) :** MUI est une bibliothèque de composants UI qui implémente les principes de conception de Material Design. Elle offre un ensemble riche de composants pré-construits, personnalisables et accessibles, permettant de créer rapidement une interface utilisateur moderne et cohérente.
- **React Router DOM :** Cette bibliothèque est utilisée pour la gestion du routage dans l'application. Elle permet de définir des routes, de naviguer entre les pages et de gérer l'historique de navigation.
- **Axios :** En tant que client HTTP, Axios facilite les requêtes API vers le backend. Il offre une API simple et intuitive pour effectuer des requêtes GET, POST, PUT et DELETE, et prend en charge les intercepteurs pour la gestion des erreurs et l'authentification.
- **Swiper :** Swiper est une bibliothèque pour la création de carrousels et de sliders responsives et interactives. Elle offre une grande variété d'options de configuration et de personnalisation, permettant de créer des présentations de produits attrayantes.
- **Day.js :** Day.js est une bibliothèque légère pour la manipulation des dates. Elle offre une API similaire à Moment.js, mais avec une empreinte mémoire beaucoup plus faible. Elle est utilisée pour formater et manipuler les dates dans l'application.

Structure Détaillée du Projet

La structure du projet est conçue pour faciliter la navigation, la maintenance et l'évolutivité du code. Voici une description détaillée des principaux répertoires et fichiers :

- **public/ :** Ce répertoire contient les ressources statiques de l'application, telles que les images, les polices et le fichier **index.html**. Ces ressources sont servies directement par le serveur web.
- **src/ :** C'est le cœur du code source de l'application. Il est divisé en plusieurs sous-répertoires, chacun ayant une responsabilité spécifique :
  - **components/ :** Ce répertoire contient les composants React réutilisables, tels que les boutons, les formulaires, les listes de produits et les carrousels.
  - **pages/ :** Ce répertoire contient les pages de l'application, telles que la page d'accueil, la page de catalogue, la page de produit et la page de panier.
  - **redux/ :** Ce répertoire contient le code Redux, y compris les actions, les reducers et le store. Il est utilisé pour la gestion de l'état global de l'application.
  - **services/ :** Ce répertoire contient les services API, qui encapsulent la logique d'interaction avec le backend. Ils sont utilisés pour effectuer des requêtes HTTP et gérer les réponses.
  - **utils/ :** Ce répertoire contient les fonctions utilitaires réutilisables, telles que les fonctions de formatage de date, les fonctions de validation de formulaire et les fonctions de gestion des erreurs.
  - **App.js :** C'est le composant racine de l'application. Il contient le routage principal et le fournisseur Redux.
  - **index.js :** C'est le point d'entrée de l'application. Il rend le composant **App** dans le DOM.
- **package.json :** Ce fichier contient la configuration et les dépendances du projet. Il est utilisé par npm ou yarn pour installer les dépendances et exécuter les scripts.
- **README.md :** Ce fichier contient la documentation générale du projet, y compris une description du projet, les instructions d'installation, les fonctionnalités principales et la compatibilité des navigateurs.

Guide d'Installation et de Configuration

Pour installer et configurer le projet, suivez ces étapes :

1. **Cloner le repository :** Utilisez la commande **git clone [URL du repository]** pour cloner le repository sur votre machine locale.
2. **Installer les dépendances :** Accédez au répertoire du projet et exécutez la commande **npm install** pour installer toutes les dépendances répertoriées dans le fichier **package.json**.
3. **Configurer les variables d'environnement :** Créez un fichier **.env** à la racine du projet et définissez les variables d'environnement requises, telles que l'URL de l'API backend :