Visual Product Matcher – Approach and Aspect

This Web-based application enables visual product similarity search using AI/ML(HuggingFace). The React/TypeScript frontend provides an intuitive interface for image upload (file)or URL with real-time filtering.
The Rust/Actix backend leverages Hugging Face's CLIP model to generate image embeddings and compute cosine similarity scores.

Key technical decisions:
Rust for high-performance backend computation, React for responsive UI, and Hugging Face inference API for accessible ML capabilities without local model deployment.
The system includes graceful fallbacks - when the ML API is unavailable, it uses mock embeddings with category-based similarity scoring.

But Why Rust and not Typescript/React Both?
The core intention for the rust is to implement the fast and the low usage of data, but also for making it safe that why rust and typescript, but still why react for frontend , its react offers the best implementation of Frontend by far and also have huge ways to make it clean and responsive.

Architecture:
The architecture separates concern cleanly: frontend handles user interactions and state management, while backend focuses on image processing and similarity algorithms.
Deployment uses Railway for Rust backend hosting and Vercel for frontend, ensuring scalability and cost-effectiveness.

Features:
Features include similarity score filtering, category-based organization, and mobile-responsive design. The project demonstrates practical AI integration for e-commerce use cases while maintaining performance and user experience.