ADR-4

Shift from React to Vanilla HTML/CSS/JS

Date: 14-04-2025

Status

Accepted

Author

Frontend Team

Context

The initial frontend of the platform was planned to be implemented using React. However, later on, it became clear that the primary focus of the project was on backend service integration, booking coordination, and payment workflows. The added complexity of managing a React-based stack was diverting time and resources away from core architectural goals like microservice orchestration and API reliability.

Decision

We decided to shift from using React to a simpler stack comprising vanilla HTML, CSS, and JavaScript for the frontend.

Alternatives

Alternative	Pros	Cons
Continue with React	Component-based architecture, reusable UI, strong community	High setup/maintenance overhead, unnecessary complexity for our current needs
Vanilla HTML/CSS/JS (Chosen)	Minimal dependencies, faster startup, easier for backend-first team	Less modularity, some repetition in DOM handling and event logic

Rationale

Since the frontend is not the primary focus of the project and is relatively static, introducing a framework like React or Vue added unnecessary overhead. Simplifying the stack helped reduce build complexities, made onboarding easier, and kept attention on the real value drivers: microservice robustness and backend reliability.

Consequences

- **Positive**: Simplified deployment, reduced dev overhead, quicker debugging, no dependency management issues.
- **Negative**: Reduced reusability and modularity, some duplication in UI code, harder scalability if UI grows later.
- **Trade-off**: Prioritized backend focus and system performance over modern frontend tooling.