

HexProperty Development Manifesto TOC v1.6 -- 2024-11-20

Monday, November 25, 2024

8:57 PM

1. Core Principles

1.1 Micro-Architecture Philosophy

- 1.1.1 Micro-frontends for UI components
- 1.1.2 Microservices for backend functionality
- 1.1.3 Micro-state management
- 1.1.4 Micro-styling with atomic CSS
- 1.1.5 Micro-testing with focused test suites

1.2 Domain-Driven Design

- 1.2.1 Core domain entities

1.3 Event-Driven Architecture

2. Architecture Guidelines

2.1 Microservices Architecture

- 2.1.1 Service Boundaries
- 2.1.2 Inter-service Communication
- 2.1.3 Deployment Strategies

2.2 CQRS and Event Sourcing

2.3 Offline Capabilities

3. Development Standards

3.1 Type Safety and Code Quality

3.2 Component Architecture

3.3 Internationalization

4. Cross-Cutting Concerns

4.1 Security and Compliance

- 4.1.1 Security Configuration
- 4.1.2 Security Service

4.2 Monitoring and Observability

- 4.2.1 Domain-Driven Monitoring
 - 4.2.1.1 Core Monitoring Domains
 - 4.2.1.2 Provider Configuration
- 4.2.2 Base Monitoring Provider
- 4.2.3 Provider Implementations
 - 4.2.3.1 Sentry Provider
 - 4.2.3.2 Stackdriver Provider
- 4.2.4 Performance Hooks
- 4.2.5 Error Boundary Integration

4.3 Performance Monitoring

- 4.3.1 Core Performance Metrics
- 4.3.2 React Component Performance

4.4 Error Handling Strategy

- 4.4.1 Error Classification
- 4.4.2 Error Recovery Strategies

- 4.5 Performance Monitoring
- 4.6 Accessibility
- 4.7 Domain-Driven Monitoring Infrastructure
 - 4.7.1 Base Monitoring Provider
 - 4.7.2 Domain-Specific Types
 - 4.7.3 Provider Implementations
 - 4.7.3.1 Sentry Provider
 - 4.7.3.2 Stackdriver Provider
 - 4.7.4 Performance Hooks
 - 4.7.5 Error Boundary Enhancement
 - 4.7.6 Monitoring Context Management
 - 4.7.7 Performance Profiling
 - 4.7.8 API Performance Monitoring
 - 4.7.9 Real User Monitoring (RUM)
- 5. Quality Assurance
 - 5.1 Unit Tests (Jest + React Testing Library)
 - 5.2 Integration Tests (Cypress)
 - 5.3 Performance Tests (Lighthouse)
 - 5.4 Accessibility Tests (axe-core)
 - 5.5 Security Tests (OWASP ZAP)
- 6. Operations and Deployment
 - 6.1 Environment-specific configurations
 - 6.2 Feature flags
 - 6.3 A/B testing support
 - 6.4 Automated testing
 - 6.5 Automated deployment
- 7. Continuous Improvement
 - 7.1 Technical Debt Management
 - 7.1.1 TechnicalDebtItem
 - 7.1.2 DebtManagementStrategy
 - 7.2 Architecture Evolution
 - 7.2.1 Regular architecture reviews
 - 7.2.2 Performance optimization cycles
 - 7.2.3 Security audits
 - 7.2.4 Accessibility improvements
 - 7.3 Feedback Integration
 - 7.3.1 User feedback collection
 - 7.3.2 Performance metrics analysis
 - 7.3.3 Error rate monitoring
 - 7.3.4 Feature usage tracking