**Non-Functional Requirements**

# 目录

Non-Functional Requirements ........................ 2

1. Product Requirements ........................ 3

1.1 Usability Requirements ........................ 4

1.2 Efficiency Requirements ........................ 5

1.3 Reliability Requirements ........................ 6

1.4 Portability Requirements ........................ 7

2. Organizational Requirements ........................ 8

2.1 Delivery Requirements ........................ 9

2.2 Implementation Requirements ........................ 10

2.3 Standards Requirements ........................ 11

3. External Requirements ........................ 12

3.1 Interoperability Requirements ........................ 13

3.2 Ethical Requirements ........................ 14

3.3 Security Requirements ........................ 15

4. Maintainability Requirements ........................ 16

5. Performance Requirements ........................ 17

6. Quality Attribute Scenarios ........................ 18

6.1 Performance Scenario ........................ 19

6.2 Security Scenario ........................ 20

6.3 Reliability Scenario ........................ 21

6.4 Usability Scenario ........................ 22

6.5 Maintainability Scenario ........................ 23

7. Trade-offs and Priorities ........................ 24

*[在Microsoft Word中，您可以右键点击上方目录区域并选择'更新域'来自动生成准确的页码]*

# **Non-Functional Requirements**

This document details the non-functional requirements for the ArtisanConnect platform, following Ian Sommerville’s categorization approach. Non-functional requirements specify criteria that judge the operation of the system, rather than specific behaviors or functions. These requirements are critical for ensuring the quality, reliability, and usability of the platform.

## **1. Product Requirements**

These requirements specify product behavior characteristics and attributes.

### **1.1 Usability Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement | Description | Verification Method |
| NFR-U01 | **Responsive Design** | The user interface shall adapt to different screen sizes (mobile, tablet, desktop) with no loss of functionality. | Visual inspection across multiple devices and viewport sizes. |
| NFR-U02 | **Accessibility Compliance** | The system shall comply with WCAG 2.1 Level AA accessibility standards. | Automated accessibility testing tools and manual review. |
| NFR-U03 | **Interface Consistency** | The user interface shall maintain consistent design patterns, terminology, and interaction models throughout the application. | Design review and user testing. |
| NFR-U04 | **Learnability** | New users shall be able to complete basic tasks (search, view provider, book service) without assistance within 5 minutes of first use. | User testing with task completion metrics. |
| NFR-U05 | **User Feedback** | The system shall provide clear feedback for all user actions within 500ms of the action being performed. | User testing and performance monitoring. |
| NFR-U06 | **Error Messages** | All error messages shall be clear, specific, and suggest corrective actions where applicable. | Manual review of error messages and user testing. |

### **1.2 Efficiency Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement | Description | Verification Method |
| NFR-E01 | **Response Time** | The system shall respond to user interactions within 2 seconds under normal load conditions. | Performance testing with simulated user load. |
| NFR-E02 | **Page Load Time** | All pages shall load within 3 seconds on a standard broadband connection (10 Mbps). | Performance testing with network throttling. |
| NFR-E03 | **Resource Utilization** | The system shall operate efficiently on client devices with minimum specifications of: 2GB RAM, 1GHz processor, and modern web browser. | Testing on reference hardware configurations. |
| NFR-E04 | **Database Performance** | Database queries shall execute within 500ms for standard operations. | Query performance monitoring and optimization. |
| NFR-E05 | **Image Optimization** | All images shall be optimized for web delivery, with provider portfolio images loading within 3 seconds. | Asset size analysis and load time testing. |

### **1.3 Reliability Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement | Description | Verification Method |
| NFR-R01 | **Availability** | The system shall maintain 99.9% uptime during business hours (8:00 AM to 10:00 PM local time). | Uptime monitoring and reporting. |
| NFR-R02 | **Mean Time Between Failures** | The system shall have a mean time between failures (MTBF) of at least 720 hours. | Failure tracking and analysis. |
| NFR-R03 | **Fault Tolerance** | The system shall handle component failures gracefully without data loss. | Chaos engineering and failure injection testing. |
| NFR-R04 | **Recovery Time** | The system shall be able to recover from failures within 1 hour. | Recovery testing and measurement. |
| NFR-R05 | **Data Integrity** | The system shall maintain data integrity across all transactions, with no data loss during normal operations. | Transaction testing and database integrity checks. |

### **1.4 Portability Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement | Description | Verification Method |
| NFR-P01 | **Browser Compatibility** | The system shall function correctly on the latest versions of Chrome, Firefox, Safari, and Edge browsers. | Cross-browser testing. |
| NFR-P02 | **Mobile Compatibility** | The system shall function correctly on iOS 14+ and Android 10+ mobile operating systems. | Mobile device testing. |
| NFR-P03 | **Deployment Flexibility** | The system shall be deployable to multiple cloud environments (AWS, Azure, Vercel) without code modifications. | Deployment testing on different platforms. |

## **2. Organizational Requirements**

These requirements derive from policies and procedures in the customer and developer organizations.

### **2.1 Delivery Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement | Description | Verification Method |
| NFR-D01 | **Development Methodology** | The system shall be developed using an Agile methodology with two-week sprints. | Process review and sprint documentation. |
| NFR-D02 | **Documentation** | Complete system documentation shall be delivered, including user guides, API documentation, and deployment instructions. | Documentation review against checklist. |
| NFR-D03 | **Version Control** | All code shall be maintained in a Git repository with a clear branching strategy. | Repository inspection and process review. |
| NFR-D04 | **Continuous Integration** | The system shall use continuous integration with automated testing for all code changes. | CI pipeline verification. |

### **2.2 Implementation Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement | Description | Verification Method |
| NFR-I01 | **Coding Standards** | All code shall adhere to established coding standards for the respective languages (JavaScript/TypeScript, CSS). | Automated linting and code reviews. |
| NFR-I02 | **Component Structure** | The codebase shall follow a clear component structure with separation of concerns. | Architecture review. |
| NFR-I03 | **Technology Stack** | The system shall be implemented using Next.js 14, React, Tailwind CSS, MongoDB, and NextAuth. | Technology stack verification. |
| NFR-I04 | **Code Documentation** | All code shall be documented with appropriate comments and JSDoc annotations. | Documentation coverage analysis. |

### **2.3 Standards Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement | Description | Verification Method |
| NFR-S01 | **Data Protection Compliance** | The system shall comply with relevant data protection regulations (GDPR, CCPA). | Compliance audit. |
| NFR-S02 | **Accessibility Standards** | The system shall comply with WCAG 2.1 Level AA accessibility standards. | Accessibility audit. |
| NFR-S03 | **Security Standards** | The system shall follow OWASP security best practices. | Security audit and penetration testing. |
| NFR-S04 | **API Standards** | All APIs shall follow REST principles and use standard HTTP status codes. | API design review. |

## **3. External Requirements**

These requirements derive from factors external to the system and its development process.

### **3.1 Interoperability Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement | Description | Verification Method |
| NFR-IO01 | **Payment Gateway Integration** | The system shall integrate with Stripe for payment processing. | Integration testing with test transactions. |
| NFR-IO02 | **Email Service Integration** | The system shall integrate with email service providers for notifications. | Email delivery testing. |
| NFR-IO03 | **Cloud Storage Integration** | The system shall integrate with Cloudinary for document and image storage. | Upload and retrieval testing. |
| NFR-IO04 | **API Compatibility** | The system’s APIs shall be versioned and maintain backward compatibility for at least one previous version. | API version testing. |

### **3.2 Ethical Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement | Description | Verification Method |
| NFR-ET01 | **Data Minimization** | The system shall collect only the minimum data necessary for its operation. | Data collection audit. |
| NFR-ET02 | **Transparency** | The system shall clearly communicate how user data is used. | Privacy policy review and user interface inspection. |
| NFR-ET03 | **Inclusivity** | The system shall be designed to be inclusive and avoid discriminatory practices. | Design review and accessibility testing. |

### **3.3 Security Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement | Description | Verification Method |
| NFR-SE01 | **Authentication** | The system shall implement secure authentication using industry-standard protocols. | Security audit and penetration testing. |
| NFR-SE02 | **Data Encryption** | All sensitive data shall be encrypted both in transit (HTTPS) and at rest. | Encryption verification and security audit. |
| NFR-SE03 | **Authorization** | The system shall enforce role-based access control for all protected resources. | Access control testing. |
| NFR-SE04 | **Input Validation** | All user inputs shall be validated and sanitized to prevent injection attacks. | Security testing with attack vectors. |
| NFR-SE05 | **Session Management** | User sessions shall be securely managed with appropriate timeout mechanisms. | Session security testing. |
| NFR-SE06 | **Secure API** | All API endpoints shall be protected against unauthorized access and abuse. | API security testing. |
| NFR-SE07 | **Vulnerability Management** | The system shall have a process for regular security updates and vulnerability patching. | Process review and update verification. |
| NFR-SE08 | **Audit Logging** | The system shall maintain audit logs for security-relevant events. | Log review and completeness verification. |

## **4. Maintainability Requirements**

These requirements relate to the ease with which the system can be maintained and evolved.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement | Description | Verification Method |
| NFR-M01 | **Code Modularity** | The system shall be built with modular components that can be updated independently. | Architecture review and dependency analysis. |
| NFR-M02 | **Test Coverage** | The system shall have automated tests covering at least 80% of critical functionality. | Test coverage analysis. |
| NFR-M03 | **Configuration Management** | System parameters shall be configurable without code changes. | Configuration testing. |
| NFR-M04 | **Documentation** | The system shall be thoroughly documented, including architecture, APIs, and deployment procedures. | Documentation review. |
| NFR-M05 | **Code Quality** | The system shall maintain a code quality score of at least B as measured by static analysis tools. | Static code analysis. |

## **5. Performance Requirements**

These requirements specify constraints on the system’s performance characteristics.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement | Description | Verification Method |
| NFR-PE01 | **Load Capacity** | The system shall support at least 1000 concurrent users without degradation in performance. | Load testing. |
| NFR-PE02 | **Scalability** | The system architecture shall allow horizontal scaling to accommodate growing user base. | Scalability testing. |
| NFR-PE03 | **Transaction Processing** | The system shall process booking transactions within 5 seconds. | Performance testing of transaction flows. |
| NFR-PE04 | **Search Performance** | Search queries shall return results within 3 seconds, even with complex filters. | Search performance testing. |
| NFR-PE05 | **API Response Time** | All API endpoints shall respond within 1 second under normal load. | API performance testing. |

## **6. Quality Attribute Scenarios**

Following Ian Sommerville’s approach, this section presents scenarios that illustrate how the non-functional requirements apply in specific situations.

### **6.1 Performance Scenario**

**Scenario**: During peak usage hours (evenings), 500 users simultaneously search for cleaning services in the same geographic area.

**Response Measure**: The system shall return search results within 3 seconds for all users, with no more than 10% increase in response time compared to off-peak hours.

### **6.2 Security Scenario**

**Scenario**: An unauthorized user attempts to access the admin dashboard by manipulating URL parameters.

**Response Measure**: The system shall detect the unauthorized access attempt, redirect the user to an error page, log the attempt with relevant details (IP address, timestamp, attempted URL), and maintain normal operation for legitimate users.

### **6.3 Reliability Scenario**

**Scenario**: The MongoDB database connection fails during a booking transaction.

**Response Measure**: The system shall detect the failure, gracefully inform the user that their request cannot be processed at the moment, store the transaction details for retry, automatically attempt reconnection, and restore normal operation once the database is available again.

### **6.4 Usability Scenario**

**Scenario**: A first-time user attempts to book a plumbing service.

**Response Measure**: The user shall be able to complete the booking process within 5 minutes without external assistance, with no more than 2 navigation errors.

### **6.5 Maintainability Scenario**

**Scenario**: A new payment provider needs to be integrated into the system.

**Response Measure**: A developer shall be able to implement the new payment provider integration within 5 person-days, requiring changes to no more than 3 existing components.

## **7. Trade-offs and Priorities**

This section acknowledges that some non-functional requirements may conflict, and establishes priorities.

|  |  |  |
| --- | --- | --- |
| Priority | Requirement Category | Rationale |
| 1 | Security | Given the sensitive nature of user data and payment processing, security is the highest priority. |
| 2 | Reliability | The system must be reliable to build trust with both customers and service providers. |
| 3 | Performance | Fast response times are critical for user satisfaction and conversion rates. |
| 4 | Usability | An intuitive interface is essential for user adoption and retention. |
| 5 | Maintainability | Long-term evolution of the platform requires maintainable code. |
| 6 | Portability | While important, this can be addressed incrementally. |

In cases where requirements conflict, those with higher priority shall take precedence. For example, if a security measure would impact performance, the security requirement shall be prioritized, and alternative approaches to mitigate performance impact shall be explored.