Table of Contents

# 1 Software Requirements Specification (SRS)

## 1.1 The Commons Academic Publishing Platform

**Document Version:** 1.1  
**Date:** January 10, 2025  
**Prepared by:** Software Engineering Team  
**Project:** The Commons Platform Modernization

## 1.2 Table of Contents

1. [Introduction](#Xe3d0fc0bea9a42ce7605565d0964033d7f6ee47)
2. [Overall Description](#X9193add25e2ce1e86c38972eca5adfa00384b97)
3. [System Features](#X5ff9624f5f78ecbb0a32b6ee906da277e1e165e)
4. [External Interface Requirements](#X88c92eec6db15cf96f948f7a6a1ac64485171fa)
5. [Non-Functional Requirements](#X1d0793fab0a3ad4cc8b1b715ac245e2ccf65199)
6. [Data Models and Architecture](#X2994f637afa4dc0cae4f8622ec69f5814b32502)
7. [Appendices](#X8cf020d2cfbf480e417545296e44ec21fed2773)

## 1.3 1. Introduction

### 1.3.1 1.1 Purpose

This Software Requirements Specification (SRS) document describes the comprehensive requirements for The Commons Academic Publishing Platform. The document serves as the authoritative specification for rebuilding and modernizing the existing scholarly publishing system, ensuring all critical functionality, business logic, and performance requirements are preserved while enabling future enhancements.

This SRS will guide: - Architecture and design decisions - Development implementation - Quality assurance testing - System integration and deployment

### 1.3.2 1.2 Scope of the Project

The Commons is a revolutionary academic publishing platform offering fair, low-cost, 100% open-access scholarly publishing. The system encompasses:

**In Scope:** - Complete manuscript submission and review workflow - Multi-role dashboard system (authors, editors, reviewers, administrators) - Intelligent peer review management with AI-powered review assistance - AI-driven search and citation discovery system - Payment processing for article processing charges (APCs) - Real-time analytics and business intelligence - Public article browsing and advanced search functionality - Advanced editorial decision workflows - Automated communication and notification systems

**Out of Scope:** - Legacy data migration from previous systems - Third-party journal integrations - Print publication workflows

### 1.3.3 1.3 Definitions, Acronyms, and Abbreviations

| Term | Definition |
| --- | --- |
| APC | Article Processing Charge |
| AI | Artificial Intelligence |
| DOI | Digital Object Identifier |
| ORCID | Open Researcher and Contributor ID |
| RAG | Retrieval-Augmented Generation |
| RLS | Row Level Security |
| SLA | Service Level Agreement |
| API | Application Programming Interface |
| COI | Conflict of Interest |
| GDPR | General Data Protection Regulation |
| WCAG | Web Content Accessibility Guidelines |

### 1.3.4 1.4 References

* PLAN.md - Project implementation roadmap
* DESIGN\_SYSTEM.md - UI/UX design specifications
* Database Migration Files - Schema definitions
* Component Library - Existing UI patterns

### 1.3.5 1.5 Overview of the Remainder of the Document

The remainder of this SRS provides: - System context and user characteristics (Section 2) - Detailed functional requirements organized by feature (Section 3) - Interface specifications for external systems (Section 4) - Performance, security, and quality requirements (Section 5) - Technical architecture and data models (Section 6)

## 1.4 2. Overall Description

### 1.4.1 2.1 Product Perspective

The Commons operates as a standalone web-based academic publishing platform that integrates with several external systems:

┌─────────────────────────────────────────┐  
│ The Commons Platform │  
├─────────────────────────────────────────┤  
│ • Author Dashboard • Editor Tools │  
│ • Review System • Public Portal │  
│ • Analytics Engine • Admin Panel │  
└─────────────────────────────────────────┘  
 ▲ ▲ ▲  
 │ │ │  
 ┌─────────────┐ ┌──────────┐ ┌────────────┐  
 │ Stripe │ │ Supabase │ │ Redis │  
 │ Payments │ │ Database │ │ Cache │  
 └─────────────┘ └──────────┘ └────────────┘  
 ▲ ▲ ▲  
 │ │ │  
 ┌─────────────┐ ┌──────────┐ ┌────────────┐  
 │ Resend │ │ Auth0 │ │ ORCID │  
 │ Email │ │ Auth │ │ API │  
 └─────────────┘ └──────────┘ └────────────┘

### 1.4.2 2.2 Product Functions

The Commons provides the following major functions:

**Core Publishing Workflow:** - Manuscript submission with multi-step wizard - Peer review assignment and management - Editorial decision processing - Publication and distribution

**User Management:** - Multi-role authentication (Author, Editor, Reviewer, Admin) - Profile management with academic credentials - Role-based access control

**Quality Assurance:** - AI-powered review quality analysis - AI reviewer integration for instant feedback - Bias detection and mitigation - Performance analytics and reporting

**Business Operations:** - Payment processing for APCs - Comprehensive analytics and reporting - Automated workflow management

### 1.4.3 2.3 User Characteristics

**Primary Users:**

| User Type | Technical Expertise | Domain Knowledge | Usage Frequency |
| --- | --- | --- | --- |
| **Authors** | Low to Medium | High (Academic) | Periodic |
| **Editors** | Medium | Very High | Daily |
| **Reviewers** | Medium | High | Weekly |
| **Administrators** | High | Medium | Daily |

**User Goals:** - **Authors:** Submit high-quality manuscripts, track submission status, receive timely feedback - **Editors:** Efficiently manage submissions, ensure review quality, make informed decisions - **Reviewers:** Provide constructive feedback, manage workload, develop professionally - **Administrators:** Monitor system performance, generate reports, manage users

### 1.4.4 2.4 Constraints

**Technical Constraints:** - Must use Next.js 14+ with App Router architecture - Database operations limited to Supabase PostgreSQL - Frontend must support all modern browsers (Chrome 90+, Firefox 88+, Safari 14+) - Must maintain WCAG 2.1 AA accessibility compliance

**Business Constraints:** - Payment processing must comply with PCI DSS standards - Must support GDPR compliance for European users - Response times must not exceed 2 seconds for core functions - 99.9% uptime requirement for production systems

**Regulatory Constraints:** - Academic integrity and ethical publishing standards - Data protection and privacy regulations - Financial transaction compliance requirements

### 1.4.5 2.5 Assumptions and Dependencies

**Assumptions:** - Users have reliable internet connections - Users possess basic web browser proficiency - Academic institutions support ORCID integration - Stripe payment processing remains available and compliant

**Dependencies:** - Supabase database service availability - Railway Redis infrastructure - Vercel deployment platform - Third-party API service levels (Stripe, Resend, Auth0)

## 1.5 3. System Features

### 1.5.1 3.1 User Authentication and Authorization

#### 1.5.1.1 3.1.1 Description and Priority

**Priority:** Critical  
**Description:** Secure user authentication system supporting multiple authentication methods with role-based access control.

#### 1.5.1.2 3.1.2 Stimulus/Response Sequences

**User Registration Flow:** 1. User accesses registration page 2. System presents registration form 3. User provides email, password, and basic information 4. System validates input and checks for duplicate accounts 5. System sends verification email 6. User confirms email and completes profile setup 7. System grants appropriate role-based access

**Authentication Flow:** 1. User provides credentials 2. System validates against Auth0/Supabase Auth 3. System creates session and redirects to appropriate dashboard 4. System maintains session state across requests

#### 1.5.1.3 3.1.3 Functional Requirements

**FR-AUTH-001:** The system SHALL support email/password authentication **FR-AUTH-002:** The system SHALL require email verification for account activation **FR-AUTH-003:** The system SHALL support password reset via secure email links **FR-AUTH-004:** The system SHALL implement role-based access control (Author, Editor, Reviewer, Admin) **FR-AUTH-005:** The system SHALL integrate with ORCID for academic identity verification **FR-AUTH-006:** The system SHALL maintain secure session management with automatic timeout **FR-AUTH-007:** The system SHALL log all authentication attempts for security monitoring

### 1.5.2 3.2 Manuscript Submission System

#### 1.5.2.1 3.2.1 Description and Priority

**Priority:** Critical  
**Description:** Comprehensive manuscript submission workflow enabling authors to submit scholarly articles with complete metadata, files, and payment processing.

#### 1.5.2.2 3.2.2 Stimulus/Response Sequences

**Submission Workflow:** 1. Author initiates new submission 2. System presents 6-step submission wizard 3. Author completes manuscript type and field selection 4. Author provides title, abstract, and keywords 5. Author adds co-authors and affiliations 6. Author uploads manuscript files and supplementary materials 7. Author provides additional information (funding, COI, etc.) 8. System processes payment via Stripe 9. System generates submission number and confirms receipt 10. System begins editorial processing

#### 1.5.2.3 3.2.3 Functional Requirements

**FR-SUB-001:** The system SHALL provide a multi-step submission wizard with progress tracking **FR-SUB-002:** The system SHALL validate required fields at each step with clear error messaging **FR-SUB-003:** The system SHALL support file uploads with drag-and-drop interface **FR-SUB-004:** The system SHALL generate unique submission tracking numbers **FR-SUB-005:** The system SHALL process Article Processing Charges (APCs) via Stripe integration **FR-SUB-006:** The system SHALL auto-save submission drafts to prevent data loss **FR-SUB-007:** The system SHALL validate file types and sizes according to submission guidelines **FR-SUB-008:** The system SHALL support multiple co-authors with contribution statements **FR-SUB-009:** The system SHALL allow authors to suggest and exclude potential reviewers

### 1.5.3 3.3 Editorial Management System

#### 1.5.3.1 3.3.1 Description and Priority

**Priority:** Critical  
**Description:** Comprehensive editorial workflow management system enabling editors to process submissions, assign reviewers, and make publication decisions.

#### 1.5.3.2 3.3.2 Stimulus/Response Sequences

**Editorial Processing:** 1. Editor accesses manuscript queue 2. System displays submissions with filtering options 3. Editor selects manuscript for review 4. System presents manuscript details and files 5. Editor assigns manuscript to self or other editors 6. System enables reviewer invitation process 7. Editor monitors review progress 8. Editor makes editorial decision based on reviews 9. System processes decision and notifies stakeholders

#### 1.5.3.3 3.3.3 Functional Requirements

**FR-EDIT-001:** The system SHALL provide a sortable, filterable manuscript queue interface **FR-EDIT-002:** The system SHALL support bulk operations for manuscript management **FR-EDIT-003:** The system SHALL enable manuscript assignment to editors **FR-EDIT-004:** The system SHALL provide decision templates with customization options **FR-EDIT-005:** The system SHALL track editorial processing times and SLA compliance **FR-EDIT-006:** The system SHALL generate automated status update notifications **FR-EDIT-007:** The system SHALL support internal editorial notes separate from author communications

### 1.5.4 3.4 Peer Review System

#### 1.5.4.1 3.4.1 Description and Priority

**Priority:** Critical  
**Description:** Intelligent peer review system with reviewer matching, invitation management, and quality assessment capabilities.

#### 1.5.4.2 3.4.2 Stimulus/Response Sequences

**Review Assignment Process:** 1. Editor initiates reviewer search 2. System suggests reviewers based on expertise matching 3. Editor selects reviewers and customizes invitations 4. System sends invitation emails with manuscript details 5. Reviewer responds to invitation 6. System tracks invitation status and sends reminders 7. Reviewer accesses review interface 8. Reviewer completes structured review form 9. System analyzes review quality and flags issues 10. System notifies editor of review completion

#### 1.5.4.3 3.4.3 Functional Requirements

**FR-REV-001:** The system SHALL provide AI-powered reviewer recommendation based on manuscript content **FR-REV-002:** The system SHALL detect conflicts of interest automatically **FR-REV-003:** The system SHALL support structured review forms with required sections **FR-REV-004:** The system SHALL enable draft review saving with progress tracking **FR-REV-005:** The system SHALL implement review quality analysis with scoring **FR-REV-006:** The system SHALL provide reviewer performance analytics **FR-REV-007:** The system SHALL support reviewer workload management and availability tracking **FR-REV-008:** The system SHALL generate automated reminder notifications for overdue reviews

### 1.5.5 3.5 Review Quality Analysis System

#### 1.5.5.1 3.5.1 Description and Priority

**Priority:** High  
**Description:** AI-powered system for analyzing review quality, detecting bias, and providing feedback to improve review standards.

#### 1.5.5.2 3.5.2 Stimulus/Response Sequences

**Quality Analysis Process:** 1. Reviewer submits completed review 2. System triggers automated quality analysis 3. System analyzes review completeness, depth, and constructiveness 4. System detects potential bias indicators 5. System generates quality report with scores and recommendations 6. System flags reviews requiring editorial attention 7. System updates reviewer performance metrics 8. System provides feedback to reviewers for improvement

#### 1.5.5.3 3.5.3 Functional Requirements

**FR-QUAL-001:** The system SHALL analyze review completeness and depth automatically **FR-QUAL-002:** The system SHALL detect bias indicators in review text **FR-QUAL-003:** The system SHALL generate quality scores with detailed breakdowns **FR-QUAL-004:** The system SHALL flag reviews falling below quality thresholds **FR-QUAL-005:** The system SHALL provide improvement recommendations to reviewers **FR-QUAL-006:** The system SHALL track quality trends over time **FR-QUAL-007:** The system SHALL correlate quality scores with reviewer characteristics

### 1.5.6 3.6 Analytics and Reporting System

#### 1.5.6.1 3.6.1 Description and Priority

**Priority:** High  
**Description:** Comprehensive analytics platform providing real-time insights into system performance, user behavior, and business metrics.

#### 1.5.6.2 3.6.2 Stimulus/Response Sequences

**Analytics Dashboard Access:** 1. Authorized user accesses analytics dashboard 2. System presents role-appropriate metrics and visualizations 3. User selects date ranges and filter criteria 4. System updates dashboard with filtered data 5. User exports reports or shares insights 6. System logs analytics access for audit purposes

#### 1.5.6.3 3.6.3 Functional Requirements

**FR-ANAL-001:** The system SHALL provide real-time performance dashboards **FR-ANAL-002:** The system SHALL track submission-to-publication conversion rates **FR-ANAL-003:** The system SHALL monitor editorial processing times and bottlenecks **FR-ANAL-004:** The system SHALL analyze reviewer performance and satisfaction **FR-ANAL-005:** The system SHALL generate financial reports and APC tracking **FR-ANAL-006:** The system SHALL provide geographic distribution analysis **FR-ANAL-007:** The system SHALL support custom report generation and scheduling

### 1.5.7 3.7 Public Article Access System

#### 1.5.7.1 3.7.1 Description and Priority

**Priority:** Medium  
**Description:** Public-facing portal for browsing, searching, and accessing published articles with full-text search capabilities.

#### 1.5.7.2 3.7.2 Stimulus/Response Sequences

**Article Discovery Process:** 1. Public user accesses article portal 2. System presents search interface and featured content 3. User enters search criteria or browses by field 4. System returns relevant results with filtering options 5. User selects article for viewing 6. System displays full article with citation tools 7. System tracks article views and downloads 8. System provides related article recommendations

#### 1.5.7.3 3.7.3 Functional Requirements

**FR-PUB-001:** The system SHALL provide advanced search functionality with multiple criteria **FR-PUB-002:** The system SHALL support browsing by academic field and category **FR-PUB-003:** The system SHALL display article metrics (views, downloads, citations) **FR-PUB-004:** The system SHALL provide citation export in multiple formats **FR-PUB-005:** The system SHALL implement SEO optimization for article discovery **FR-PUB-006:** The system SHALL generate RSS feeds for content syndication **FR-PUB-007:** The system SHALL support social media sharing integration

### 1.5.8 3.8 AI-Powered Search and Citation System

#### 1.5.8.1 3.8.1 Description and Priority

**Priority:** High  
**Description:** Intelligent search system using Vertex AI Search and RAG (Retrieval-Augmented Generation) technology to provide contextual search results with direct citations and references from the platform’s publication repository, encouraging user retention and research discovery.

#### 1.5.8.2 3.8.2 Stimulus/Response Sequences

**AI Search Process:** 1. User enters natural language research query 2. System processes query using AI agent to understand research intent 3. System searches publication repository using RAG technology 4. System retrieves relevant articles, sections, and citations 5. AI agent synthesizes contextual response with direct references 6. System presents results with in-line citations and source links 7. User clicks citations to access full articles on platform 8. System tracks search patterns and refines AI recommendations 9. System suggests related queries and research paths

**Citation Discovery Flow:** 1. User browses article or search results 2. System identifies key concepts and research areas 3. AI agent suggests related publications within platform 4. System presents citation network visualization 5. User explores citation relationships and references 6. System maintains user research session and history

#### 1.5.8.3 3.8.3 Functional Requirements

**FR-AISEARCH-001:** The system SHALL implement Vertex AI Search for natural language query processing **FR-AISEARCH-002:** The system SHALL use RAG technology to provide contextual responses with citations **FR-AISEARCH-003:** The system SHALL maintain a searchable vector database of all published content **FR-AISEARCH-004:** The system SHALL provide real-time citation suggestions based on content similarity **FR-AISEARCH-005:** The system SHALL generate bibliographies and reference lists automatically **FR-AISEARCH-006:** The system SHALL support complex academic queries with Boolean and field-specific operators **FR-AISEARCH-007:** The system SHALL track search analytics and user research patterns **FR-AISEARCH-008:** The system SHALL provide search result explanations showing relevance reasoning **FR-AISEARCH-009:** The system SHALL integrate with existing article metadata for enhanced search accuracy **FR-AISEARCH-010:** The system SHALL support export of search results and citations in academic formats

### 1.5.9 3.9 AI Reviewer Integration System

#### 1.5.9.1 3.9.1 Description and Priority

**Priority:** High  
**Description:** Automated AI reviewer system that provides instant, preliminary manuscript reviews to supplement human reviewer feedback, offering rapid turnaround times while maintaining human editorial oversight and final decision authority.

#### 1.5.9.2 3.9.2 Stimulus/Response Sequences

**AI Review Process:** 1. Editor assigns manuscript for review (including AI reviewer) 2. System automatically triggers AI review upon assignment 3. AI reviewer analyzes manuscript content, structure, and methodology 4. System generates comprehensive AI review within minutes 5. AI review includes scoring, comments, and recommendations 6. System makes AI review available to human reviewers for reference 7. Human reviewers complete their reviews with AI insights visible 8. Editor receives both AI and human reviews for decision making 9. System tracks AI review accuracy against human reviewer consensus

**AI Review Quality Assurance:** 1. System compares AI reviews with human reviewer outcomes 2. System identifies patterns where AI and human reviews diverge 3. System updates AI models based on editorial decisions and feedback 4. System flags manuscripts where AI review significantly differs from humans 5. Editorial team reviews AI performance and adjusts parameters

#### 1.5.9.3 3.9.3 Functional Requirements

**FR-AIREV-001:** The system SHALL automatically assign an AI reviewer to every manuscript submission **FR-AIREV-002:** The system SHALL generate AI reviews within 15 minutes of assignment **FR-AIREV-003:** The system SHALL provide structured AI reviews with scores and detailed comments **FR-AIREV-004:** The system SHALL analyze manuscript methodology, literature review, and conclusions **FR-AIREV-005:** The system SHALL identify potential plagiarism and citation issues **FR-AIREV-006:** The system SHALL flag manuscripts requiring special editorial attention **FR-AIREV-007:** The system SHALL provide AI review confidence scores and uncertainty indicators **FR-AIREV-008:** The system SHALL allow human reviewers to view AI reviews as reference material **FR-AIREV-009:** The system SHALL track AI review accuracy and performance metrics **FR-AIREV-010:** The system SHALL support AI model updates and training based on editorial feedback **FR-AIREV-011:** The system SHALL clearly distinguish AI reviews from human reviews in all interfaces **FR-AIREV-012:** The system SHALL ensure AI reviews do not replace human editorial decision-making authority

## 1.6 4. External Interface Requirements

### 1.6.1 4.1 User Interfaces

#### 1.6.1.1 4.1.1 General UI Requirements

* **Responsive Design:** All interfaces must be fully responsive (320px to 2560px)
* **Accessibility:** WCAG 2.1 AA compliance mandatory
* **Browser Support:** Chrome 90+, Firefox 88+, Safari 14+, Edge 90+
* **Loading Performance:** Initial page load < 2 seconds, subsequent navigation < 1 second

#### 1.6.1.2 4.1.2 Design System Specifications

* **Color Palette:** Deep Academic Blue (#1e3a8a), Scholarly Gold (#d97706), Sage Green (#16a34a)
* **Typography:** Playfair Display (headings), Inter (UI), Crimson Text (articles)
* **Component Library:** Shadcn/UI with custom academic extensions
* **Spacing System:** 4px base unit with systematic scaling

### 1.6.2 4.2 Hardware Interfaces

#### 1.6.2.1 4.2.1 Server Infrastructure

* **Application Hosting:** Vercel Edge Functions
* **Database:** Supabase PostgreSQL with connection pooling
* **Cache Layer:** Railway Redis with persistence
* **Storage:** Supabase Storage with CDN distribution

#### 1.6.2.2 4.2.2 Client Requirements

* **Minimum RAM:** 4GB for optimal performance
* **Network:** Broadband internet connection (>10 Mbps recommended)
* **Storage:** 500MB browser cache allocation

### 1.6.3 4.3 Software Interfaces

#### 1.6.3.1 4.3.1 Authentication Services

* **Auth0 Integration**
  + **Purpose:** Primary authentication provider
  + **Protocol:** OAuth 2.0 / OpenID Connect
  + **Data Exchange:** User profiles, session tokens
  + **Endpoints:** Login, logout, user management APIs

#### 1.6.3.2 4.3.2 Payment Processing

* **Stripe Integration**
  + **Purpose:** Article Processing Charge (APC) collection
  + **Protocol:** REST API with webhooks
  + **Data Exchange:** Payment intents, transaction confirmations
  + **Security:** PCI DSS Level 1 compliance required

#### 1.6.3.3 4.3.3 Database Services

* **Supabase PostgreSQL**
  + **Purpose:** Primary data storage and real-time subscriptions
  + **Protocol:** PostgREST API with Row Level Security
  + **Data Exchange:** All application data with JSON/JSONB support
  + **Backup:** Automated daily backups with point-in-time recovery

#### 1.6.3.4 4.3.4 Caching and Session Management

* **Redis (Railway)**
  + **Purpose:** Session storage, rate limiting, performance caching
  + **Protocol:** Redis Protocol (RESP)
  + **Data Exchange:** Serialized session data, cached queries
  + **Persistence:** AOF and RDB snapshots enabled

#### 1.6.3.5 4.3.5 Email Communications

* **Resend Email Service**
  + **Purpose:** Transactional email delivery
  + **Protocol:** REST API
  + **Data Exchange:** Email templates, delivery confirmations
  + **Deliverability:** DKIM/SPF configured domains

#### 1.6.3.6 4.3.6 External Academic Services

* **ORCID API**
  + **Purpose:** Author and reviewer identity verification
  + **Protocol:** REST API with OAuth 2.0
  + **Data Exchange:** Academic profiles, publication history
  + **Rate Limits:** 24 requests per second per IP

#### 1.6.3.7 4.3.7 AI and Machine Learning Services

* **Google Vertex AI Search**
  + **Purpose:** AI-powered search and RAG capabilities
  + **Protocol:** REST API with OAuth 2.0
  + **Data Exchange:** Search queries, vector embeddings, search results
  + **Rate Limits:** Enterprise tier with scalable quotas
  + **Features:** Natural language processing, semantic search, citation analysis
* **OpenAI GPT Models (via API)**
  + **Purpose:** AI reviewer analysis and content generation
  + **Protocol:** REST API with API key authentication
  + **Data Exchange:** Manuscript content, review analysis, structured feedback
  + **Rate Limits:** Based on token usage and tier limits
  + **Features:** Content analysis, bias detection, academic writing assessment
* **Vector Database (Pinecone/Weaviate)**
  + **Purpose:** Semantic search and similarity matching for publications
  + **Protocol:** REST API with API key authentication
  + **Data Exchange:** Document embeddings, similarity queries, metadata
  + **Rate Limits:** Based on service tier and request volume
  + **Features:** Vector similarity search, metadata filtering, hybrid search

### 1.6.4 4.4 Communications Interfaces

#### 1.6.4.1 4.4.1 API Specifications

* **Protocol:** HTTPS with TLS 1.3
* **Data Format:** JSON with UTF-8 encoding
* **Authentication:** Bearer tokens (JWT)
* **Rate Limiting:** Tiered based on endpoint criticality
* **Documentation:** OpenAPI 3.0 specification

#### 1.6.4.2 4.4.2 Real-time Communications

* **WebSocket Support:** For real-time notifications and live updates
* **Server-Sent Events:** For dashboard metric updates
* **Push Notifications:** Browser-based notifications for critical events

## 1.7 5. Non-Functional Requirements

### 1.7.1 5.1 Performance Requirements

#### 1.7.1.1 5.1.1 Response Time Requirements

* **Page Load Time:** < 2 seconds for initial load, < 1 second for navigation
* **API Response Time:** < 500ms for 95th percentile requests
* **Database Query Time:** < 200ms for single table queries
* **File Upload Speed:** Support up to 100MB files with progress indication
* **AI Search Response:** < 3 seconds for complex RAG queries
* **AI Review Generation:** < 15 minutes for complete manuscript analysis

#### 1.7.1.2 5.1.2 Throughput Requirements

* **Concurrent Users:** Support 1,000 simultaneous active users
* **Submission Volume:** Handle 500 manuscript submissions per day
* **Review Processing:** Support 2,000 reviews per month (human) + automated AI reviews
* **Search Queries:** Process 10,000 search requests per hour
* **AI Search Queries:** Support 1,000 complex AI search requests per hour
* **Vector Database Operations:** Handle 50,000 similarity queries per day

#### 1.7.1.3 5.1.3 Scalability Requirements

* **Horizontal Scaling:** Auto-scale based on CPU/memory utilization
* **Database Scaling:** Read replicas for query distribution
* **CDN Integration:** Global content distribution for static assets
* **Load Balancing:** Automatic traffic distribution across instances

### 1.7.2 5.2 Safety/Security Requirements

#### 1.7.2.1 5.2.1 Authentication Security

* **Password Policy:** Minimum 12 characters with complexity requirements
* **Session Management:** Secure session tokens with 2-hour timeout
* **Multi-Factor Authentication:** Optional 2FA for elevated roles
* **Account Lockout:** Progressive delays for failed login attempts

#### 1.7.2.2 5.2.2 Data Protection

* **Encryption at Rest:** AES-256 encryption for sensitive data
* **Encryption in Transit:** TLS 1.3 for all communications
* **Data Anonymization:** Reviewer identity protection in manuscript sharing
* **Backup Security:** Encrypted backups with access controls

#### 1.7.2.3 5.2.3 Application Security

* **Input Validation:** Server-side validation for all user inputs
* **SQL Injection Prevention:** Parameterized queries and ORMs only
* **XSS Protection:** Content Security Policy and input sanitization
* **CSRF Protection:** CSRF tokens for state-changing operations

#### 1.7.2.4 5.2.4 Rate Limiting and Abuse Prevention

* **API Rate Limits:** Tiered limits based on endpoint sensitivity
* **Upload Restrictions:** File type validation and malware scanning
* **Spam Prevention:** CAPTCHA integration for public forms
* **Suspicious Activity Detection:** Automated monitoring and alerting

### 1.7.3 5.3 Software Quality Attributes

#### 1.7.3.1 5.3.1 Reliability

* **Uptime Requirement:** 99.9% availability (8.76 hours downtime/year)
* **Mean Time to Recovery:** < 4 hours for critical issues
* **Data Integrity:** ACID transactions with referential integrity
* **Fault Tolerance:** Graceful degradation during partial outages

#### 1.7.3.2 5.3.2 Availability

* **Service Monitoring:** Real-time health checks and alerting
* **Backup Systems:** Automated failover for critical components
* **Maintenance Windows:** Scheduled maintenance during low-usage periods
* **Geographic Redundancy:** Multi-region deployment for disaster recovery

#### 1.7.3.3 5.3.3 Maintainability

* **Code Quality:** TypeScript strict mode with comprehensive type safety
* **Testing Coverage:** 85% minimum test coverage for critical paths
* **Documentation:** API documentation with examples and changelog
* **Deployment Automation:** CI/CD pipeline with automated testing

#### 1.7.3.4 5.3.4 Usability

* **Accessibility:** WCAG 2.1 AA compliance verified through automated testing
* **User Experience:** Task completion rate > 90% for core workflows
* **Error Handling:** Clear, actionable error messages with recovery options
* **Help System:** Contextual help and comprehensive user guides

#### 1.7.3.5 5.3.5 Portability

* **Browser Compatibility:** Cross-browser testing for major browsers
* **Mobile Responsiveness:** Full functionality on mobile devices
* **API Standardization:** RESTful APIs with standard HTTP methods
* **Data Export:** Standard formats (JSON, CSV) for data portability

### 1.7.4 5.4 Business Rules

#### 1.7.4.1 5.4.1 Editorial Workflow Rules

* **Review Assignment:** Minimum 2 reviewers required per manuscript
* **Conflict of Interest:** Automatic detection and prevention of COI assignments
* **Decision Timeline:** Maximum 90 days from submission to decision
* **Quality Thresholds:** Reviews below quality threshold flagged for editor review

#### 1.7.4.2 5.4.2 Financial Rules

* **Payment Processing:** APC payment required before manuscript processing
* **Refund Policy:** Full refund available within 30 days of rejection
* **Currency Support:** USD primary currency with international payment support
* **Transaction Logging:** Complete audit trail for all financial transactions

#### 1.7.4.3 5.4.3 User Role Rules

* **Role Assignment:** Default role assignment based on registration data
* **Permission Inheritance:** Hierarchical permission model (Admin > Editor > Author/Reviewer)
* **Role Changes:** Administrative approval required for role elevation
* **Account Deactivation:** Graceful handling of deactivated user content

#### 1.7.4.4 5.4.4 Data Retention Rules

* **Manuscript Retention:** Permanent retention for published articles
* **Review Data:** 7-year retention for completed reviews
* **User Data:** GDPR-compliant data retention and deletion policies
* **Audit Logs:** 5-year retention for security and compliance auditing

## 1.8 6. Data Models and Architecture

### 1.8.1 6.1 System Architecture

#### 1.8.1.1 6.1.1 High-Level Architecture

┌─────────────────────────────────────────────────────────┐  
│ Frontend Layer │  
│ ┌─────────────┐ ┌─────────────┐ ┌─────────────────┐ │  
│ │ Next.js 14 │ │ React 19 │ │ Tailwind CSS │ │  
│ │ App Router │ │ Components │ │ Shadcn/UI │ │  
│ └─────────────┘ └─────────────┘ └─────────────────┘ │  
└─────────────────────────────────────────────────────────┘  
 │  
┌─────────────────────────────────────────────────────────┐  
│ Application Layer │  
│ ┌─────────────┐ ┌─────────────┐ ┌─────────────────┐ │  
│ │ API Routes │ │ Middleware │ │ Service Layer │ │  
│ │ (REST/tRPC) │ │ (Auth/CORS) │ │ (Business Logic)│ │  
│ └─────────────┘ └─────────────┘ └─────────────────┘ │  
└─────────────────────────────────────────────────────────┘  
 │  
┌─────────────────────────────────────────────────────────┐  
│ Data Layer │  
│ ┌─────────────┐ ┌─────────────┐ ┌─────────────────┐ │  
│ │ Supabase │ │ Redis │ │ File Storage │ │  
│ │ PostgreSQL │ │ Cache/Queue │ │ (Supabase) │ │  
│ └─────────────┘ └─────────────┘ └─────────────────┘ │  
└─────────────────────────────────────────────────────────┘  
 │  
┌─────────────────────────────────────────────────────────┐  
│ External Services │  
│ ┌─────────────┐ ┌─────────────┐ ┌─────────────────┐ │  
│ │ Auth0 │ │ Stripe │ │ Resend Email │ │  
│ │ ORCID │ │ Vercel │ │ Railway Redis │ │  
│ └─────────────┘ └─────────────┘ └─────────────────┘ │  
└─────────────────────────────────────────────────────────┘

#### 1.8.1.2 6.1.2 Component Architecture

**Frontend Components:** - **Layout Components:** Header, footer, navigation, mobile menu - **Dashboard Components:** Role-specific dashboards with widgets - **Form Components:** Multi-step wizards, validation, file upload - **Display Components:** Article viewers, data tables, charts - **UI Components:** Buttons, modals, alerts, loading states

**Backend Services:** - **Authentication Service:** User management, session handling - **Manuscript Service:** Submission processing, file management - **Review Service:** Assignment, quality analysis, notifications - **Analytics Service:** Metrics collection, report generation - **Email Service:** Template processing, delivery tracking

### 1.8.2 6.2 Database Schema

#### 1.8.2.1 6.2.1 Core Entity Relationships

┌─────────────┐ ┌─────────────────┐ ┌─────────────┐  
│ profiles │────▶│ manuscripts │◀────│ reviews │  
│ │ │ │ │ │  
│ - id (PK) │ │ - id (PK) │ │ - id (PK) │  
│ - full\_name │ │ - title │ │ - summary │  
│ - email │ │ - abstract │ │ - recommendation │  
│ - role │ │ - author\_id (FK)│ │ - reviewer\_id (FK)│  
│ - expertise │ │ - editor\_id (FK)│ │ - manuscript\_id (FK)│  
└─────────────┘ │ - status │ └─────────────┘  
 └─────────────────┘ │  
 │ │  
 ┌─────────────────┐ │  
 │review\_assignments│ │  
 │ │ │  
 │ - id (PK) │ │  
 │ - manuscript\_id (FK) │  
 │ - reviewer\_id (FK) │  
 │ - status │◀─────────┘  
 │ - due\_date │  
 └─────────────────┘

#### 1.8.2.2 6.2.2 Key Tables and Relationships

**Primary Tables:** - profiles - User profiles with role and expertise data - manuscripts - Core manuscript data and metadata - reviews - Review submissions with ratings and comments - review\_assignments - Reviewer invitations and status tracking - editorial\_decisions - Editorial decisions with templates and workflow - payments - APC transaction records and billing information

**Supporting Tables:** - manuscript\_files - File attachments and version control - manuscript\_coauthors - Co-author information and contributions - activity\_logs - System activity audit trail - notifications - User notification queue and status - fields\_of\_study - Academic field taxonomy - editorial\_templates - Decision letter templates and components

#### 1.8.2.3 6.2.3 Data Types and Constraints

**Enumerated Types:** - user\_role: ‘author’, ‘editor’, ‘reviewer’, ‘admin’ - manuscript\_status: ‘draft’, ‘submitted’, ‘with\_editor’, ‘under\_review’, ‘revisions\_requested’, ‘accepted’, ‘rejected’, ‘published’ - review\_recommendation: ‘accept’, ‘minor\_revisions’, ‘major\_revisions’, ‘reject’ - assignment\_status: ‘invited’, ‘accepted’, ‘declined’, ‘completed’, ‘expired’

**JSON Data Structures:** - suggested\_reviewers: Reviewer recommendations with expertise matching - excluded\_reviewers: COI declarations and exclusion reasons - billing\_details: Stripe payment information and addresses - quality\_metrics: AI analysis results and scoring details

### 1.8.3 6.3 Security Architecture

#### 1.8.3.1 6.3.1 Row Level Security (RLS) Policies

**Profile Access:**

-- Users can view public profiles  
CREATE POLICY "Public profiles viewable" ON profiles FOR SELECT USING (true);  
  
-- Users can update their own profile  
CREATE POLICY "Users update own profile" ON profiles FOR UPDATE   
USING (auth.uid() = id);

**Manuscript Access:**

-- Authors can view their own manuscripts  
CREATE POLICY "Authors view own manuscripts" ON manuscripts FOR SELECT   
USING (auth.uid() = author\_id);  
  
-- Editors can view assigned manuscripts  
CREATE POLICY "Editors view assigned manuscripts" ON manuscripts FOR SELECT   
USING (auth.uid() = editor\_id OR EXISTS (  
 SELECT 1 FROM profiles WHERE id = auth.uid() AND role IN ('editor', 'admin')  
));  
  
-- Published manuscripts are public  
CREATE POLICY "Published manuscripts public" ON manuscripts FOR SELECT   
USING (status = 'published');

#### 1.8.3.2 6.3.2 API Security Layers

**Authentication Middleware:** - JWT token validation - Session management - Rate limiting by user/IP

**Authorization Middleware:** - Role-based access control - Resource-level permissions - Operation-specific checks

**Input Validation:** - Schema-based validation (Zod) - File type and size restrictions - SQL injection prevention

### 1.8.4 6.4 Performance Architecture

#### 1.8.4.1 6.4.1 Caching Strategy

**Redis Caching Layers:**

// Application-level caching  
const CACHE\_KEYS = {  
 USER\_PROFILE: (id: string) => `profile:${id}`,  
 MANUSCRIPT\_DETAILS: (id: string) => `manuscript:${id}`,  
 SEARCH\_RESULTS: (query: string) => `search:${hash(query)}`,  
 ANALYTICS\_DASHBOARD: (role: string) => `analytics:${role}`,  
}  
  
// Cache TTL (Time To Live) settings  
const CACHE\_TTL = {  
 USER\_PROFILES: 3600, // 1 hour  
 SEARCH\_RESULTS: 1800, // 30 minutes  
 ANALYTICS\_DATA: 300, // 5 minutes  
 STATIC\_CONTENT: 86400, // 24 hours  
}

**Database Query Optimization:** - Strategic indexing on frequently queried columns - Connection pooling for efficient resource utilization - Read replicas for analytical queries - Materialized views for complex aggregations

#### 1.8.4.2 6.4.2 Frontend Performance

**Code Splitting Strategy:**

// Route-based code splitting  
const AuthorDashboard = lazy(() => import('@/components/dashboard/author'))  
const EditorDashboard = lazy(() => import('@/components/dashboard/editor'))  
const ReviewerDashboard = lazy(() => import('@/components/dashboard/reviewer'))  
  
// Component-based lazy loading  
const HeavyAnalyticsChart = lazy(() => import('@/components/charts/heavy-chart'))

**Asset Optimization:** - Image optimization with Next.js Image component - Font subset loading for faster text rendering - CSS-in-JS with build-time optimization - Bundle analysis and dead code elimination

## 1.9 7. Appendices

### 1.9.1 7.1 Technology Stack Summary

| Layer | Technology | Version | Purpose |
| --- | --- | --- | --- |
| **Frontend** | Next.js | 15.3.4 | React framework with App Router |
| **Frontend** | React | 19.0.0 | Component library |
| **Frontend** | TypeScript | 5.x | Type-safe JavaScript |
| **Frontend** | Tailwind CSS | 3.4.0 | Utility-first CSS framework |
| **UI Components** | Shadcn/UI | Latest | Component library |
| **UI Components** | Radix UI | Latest | Headless component primitives |
| **Database** | Supabase | Latest | PostgreSQL with real-time features |
| **Cache/Queue** | Redis | Latest | In-memory data structure store |
| **Authentication** | Auth0 | 4.9.0 | Identity and access management |
| **Payments** | Stripe | 18.2.1 | Payment processing |
| **Email** | Resend | 4.6.0 | Transactional email service |
| **AI Search** | Google Vertex AI | Latest | AI-powered search and RAG capabilities |
| **AI Models** | OpenAI API | Latest | AI reviewer and content analysis |
| **Vector Database** | Pinecone/Weaviate | Latest | Semantic search and similarity matching |
| **Deployment** | Vercel | Latest | Frontend hosting and serverless functions |
| **Infrastructure** | Railway | Latest | Redis hosting and management |

### 1.9.2 7.2 API Endpoint Summary

#### 1.9.2.1 7.2.1 Authentication Endpoints

* POST /api/auth/login - User authentication
* POST /api/auth/logout - Session termination
* POST /api/auth/register - New user registration
* POST /api/auth/verify - Email verification
* POST /api/auth/reset-password - Password reset initiation

#### 1.9.2.2 7.2.2 Manuscript Management

* POST /api/manuscripts/submit - New manuscript submission
* GET /api/manuscripts/[id] - Retrieve manuscript details
* PUT /api/manuscripts/[id] - Update manuscript information
* POST /api/manuscripts/[id]/files - File upload handling
* POST /api/manuscripts/[id]/checkout - Payment processing

#### 1.9.2.3 7.2.3 Editorial Workflow

* GET /api/editorial/manuscripts - Editorial queue management
* POST /api/editorial/decision - Editorial decision processing
* POST /api/editorial/reviewers/invite - Reviewer invitation
* GET /api/editorial/analytics - Editorial performance metrics

#### 1.9.2.4 7.2.4 Review System

* GET /api/reviews/[id] - Review form access
* POST /api/reviews/submit - Review submission
* POST /api/reviews/assign - Reviewer assignment
* GET /api/reviews/quality-report - Quality analysis results

#### 1.9.2.5 7.2.5 AI-Powered Features

* POST /api/ai/search - AI-powered search with RAG
* GET /api/ai/citations/suggest - Citation recommendations
* POST /api/ai/review/generate - Generate AI review
* GET /api/ai/review/[id] - Retrieve AI review results
* POST /api/ai/embeddings/generate - Generate document embeddings
* GET /api/ai/similar-articles - Find similar publications

### 1.9.3 7.3 Quality Metrics and Success Criteria

#### 1.9.3.1 7.3.1 Performance Benchmarks

* **Lighthouse Performance Score:** > 90
* **First Contentful Paint (FCP):** < 1.5 seconds
* **Largest Contentful Paint (LCP):** < 2.5 seconds
* **Cumulative Layout Shift (CLS):** < 0.1
* **Time to Interactive (TTI):** < 3.5 seconds

#### 1.9.3.2 7.3.2 Business Success Metrics

* **Submission Processing Time:** Average < 60 days from submission to decision
* **Author Satisfaction:** > 90% positive feedback on submission experience
* **Review Quality Score:** Average review quality > 4.0/5.0 (human and AI combined)
* **System Availability:** > 99.9% uptime
* **Payment Success Rate:** > 95% successful APC transactions
* **AI Search Engagement:** > 70% of users utilize AI search features
* **AI Review Accuracy:** > 85% correlation with human reviewer recommendations
* **Citation Discovery Rate:** > 40% increase in cross-platform article references

#### 1.9.3.3 7.3.3 Security and Compliance Metrics

* **Security Vulnerabilities:** Zero critical vulnerabilities in production
* **Data Breach Incidents:** Zero incidents with data exposure
* **GDPR Compliance:** 100% compliance with data protection requests
* **Authentication Security:** < 0.1% fraudulent access attempts

### 1.9.4 7.4 Migration and Deployment Considerations

#### 1.9.4.1 7.4.1 Data Migration Strategy

* **Legacy System Analysis:** Complete assessment of existing data structures
* **Schema Mapping:** Field-by-field mapping from legacy to new schema
* **Data Validation:** Comprehensive validation of migrated data integrity
* **Rollback Procedures:** Complete rollback capability during migration phase

#### 1.9.4.2 7.4.2 Deployment Pipeline

# CI/CD Pipeline Stages  
stages:  
 - code\_quality: # ESLint, Prettier, TypeScript checking  
 - unit\_tests: # Jest unit tests with coverage  
 - integration\_tests: # API and component integration tests  
 - security\_scan: # SAST/DAST security scanning  
 - build: # Production build generation  
 - e2e\_tests: # Playwright end-to-end tests  
 - staging\_deploy: # Deploy to staging environment  
 - manual\_approval: # Manual testing and approval  
 - production\_deploy: # Production deployment  
 - post\_deploy\_tests: # Smoke tests and monitoring

#### 1.9.4.3 7.4.3 Rollback and Recovery Procedures

* **Blue-Green Deployment:** Zero-downtime deployment with instant rollback
* **Database Migrations:** Reversible migrations with backup points
* **Feature Flags:** Gradual rollout with instant disable capability
* **Monitoring and Alerting:** Real-time issue detection and automated responses

### 1.9.5 7.5 Compliance and Regulatory Requirements

#### 1.9.5.1 7.5.1 Academic Publishing Standards

* **COPE Guidelines:** Committee on Publication Ethics compliance
* **DOAJ Standards:** Directory of Open Access Journals requirements
* **Crossref Integration:** DOI registration and metadata standards
* **ORCID Integration:** Author identification and attribution

#### 1.9.5.2 7.5.2 Data Protection and Privacy

* **GDPR Compliance:** European data protection regulation adherence
* **CCPA Compliance:** California Consumer Privacy Act requirements
* **Data Minimization:** Collection of only necessary personal data
* **Right to Erasure:** Complete data deletion upon user request

#### 1.9.5.3 7.5.3 Financial and Security Compliance

* **PCI DSS Level 1:** Payment card industry data security standards
* **SOC 2 Type II:** Service organization control requirements
* **ISO 27001:** Information security management standards
* **AICPA Standards:** Financial reporting and audit requirements

**Document Control:** - **Version:** 1.1 - **Last Updated:** January 10, 2025 - **Next Review:** March 10, 2025 - **Approved By:** Software Engineering Team - **Status:** Updated with AI Features

This Software Requirements Specification serves as the comprehensive blueprint for The Commons academic publishing platform modernization project. All development activities should reference and comply with the requirements outlined in this document.