# Product Requirements Document (PRD)

## Book Management System - Bulk Upload Feature

### Document Information

* **Product**: Book Management System
* **Feature**: Bulk Book Upload
* **Version**: 1.0
* **Date**: 2025-01-26
* **Status**: Approved for Development

────────────────────────────────────────────────────────────────────────────────

## Executive Summary

The Book Management System currently supports individual CRUD operations for book records. This PRD outlines the requirements for adding bulk upload capabilities, allowing users to efficiently import large numbers of books via CSV and Excel files.

### Business Impact

* **Efficiency Gain**: Reduce book entry time from 2-3 minutes per book to bulk processing
* **User Satisfaction**: Address #1 user-requested feature (bulk import)
* **Scale Support**: Enable migration of existing book inventories from other systems
* **Error Reduction**: Automated validation reduces manual entry errors

────────────────────────────────────────────────────────────────────────────────

## Problem Statement

### Current Pain Points

1. **Manual Entry Inefficiency**: Users must create books one at a time through individual API calls
2. **Data Migration Barriers**: No easy way to import existing book inventories
3. **Time-Intensive Operations**: Large inventory setup requires hundreds of individual requests
4. **Error-Prone Process**: Manual entry increases likelihood of data entry mistakes

### User Feedback

"We have 5,000 books in our existing Excel spreadsheet. Creating them one by one would take weeks." - Library Administrator  
"Our bookstore needs to quickly upload seasonal inventory changes. The current API isn't practical for bulk operations." - Bookstore Manager

────────────────────────────────────────────────────────────────────────────────

## Product Objectives

### Primary Goals

1. **Enable Bulk Import**: Support CSV and Excel file uploads for book creation
2. **Maintain Data Quality**: Provide comprehensive validation with detailed error reporting
3. **Ensure Reliability**: Handle partial failures gracefully without data corruption
4. **Preserve Performance**: Process large files efficiently without blocking other operations

### Success Metrics

* **Adoption Rate**: 80% of new large customers use bulk upload within first month
* **Efficiency**: Average time to upload 1000 books < 30 seconds
* **Error Rate**: <5% validation failures on properly formatted files
* **User Satisfaction**: 90% satisfaction rating for bulk upload feature

────────────────────────────────────────────────────────────────────────────────

## Target Users

### Primary Users

1. **Library Staff Members**
2. Need to migrate existing book catalogs
3. Manage seasonal book acquisitions
4. Handle donations and new collection additions
5. **Bookstore Managers**
6. Import supplier catalogs
7. Update inventory from distribution systems
8. Manage promotional book collections

### Secondary Users

1. **System Administrators**
2. Need bulk data loading for system setup
3. Perform data recovery operations
4. Handle system migrations

────────────────────────────────────────────────────────────────────────────────

## User Stories & Acceptance Criteria

### Epic: Bulk Book Upload

#### Story 1: CSV File Upload

**As a** library staff member  
**I want to** upload books via CSV file  
**So that** I can quickly import our existing catalog

**Acceptance Criteria:**  
- ✅ Upload CSV files via web API  
- ✅ Support standard CSV format with headers  
- ✅ Validate all book data before import  
- ✅ Return detailed success/failure report  
- ✅ Handle files up to 10MB and 10,000 records

#### Story 2: Excel File Upload

**As a** bookstore manager  
**I want to** upload books via Excel file  
**So that** I can import supplier catalogs directly

**Acceptance Criteria:**  
- ✅ Support Excel formats (.xlsx, .xls)  
- ✅ Read from first worksheet  
- ✅ Handle Excel formatting (dates, currency)  
- ✅ Skip empty rows automatically  
- ✅ Preserve numeric precision for prices

#### Story 3: Validation & Error Reporting

**As a** system user  
**I want to** receive detailed validation feedback  
**So that** I can correct data issues efficiently

**Acceptance Criteria:**  
- ✅ Validate each record individually  
- ✅ Continue processing valid records despite invalid ones  
- ✅ Provide row-level error details  
- ✅ Include field-specific validation messages  
- ✅ Return summary with success/failure counts

────────────────────────────────────────────────────────────────────────────────

## Functional Requirements

### File Processing

|  |  |
| --- | --- |
| **Requirement** | **Details** |
| Supported Formats | CSV (.csv), Excel (.xlsx, .xls) |
| Maximum File Size | 10MB |
| Maximum Records | 10,000 books per file |
| Encoding | UTF-8 for CSV, native for Excel |
| Headers | Required for CSV, optional for Excel |

### Data Validation

|  |  |
| --- | --- |
| **Field** | **Validation Rules** |
| Name | Required, non-empty, max 255 characters |
| Author | Required, non-empty, max 255 characters |
| Price | Required, positive BigDecimal, max 2 decimal places |
| ISBN | Optional, valid ISBN-10 or ISBN-13 format |
| Category | Optional, max 100 characters |

### Processing Behavior

* **Transaction Handling**: Partial success allowed (valid books saved)
* **Duplicate Handling**: Allow duplicates (no unique constraint on name/author)
* **Memory Management**: Stream processing for large files
* **Performance**: Process 1000 records in <30 seconds

────────────────────────────────────────────────────────────────────────────────

## Technical Requirements

### API Specification

POST /books/bulk  
Content-Type: multipart/form-data  
  
Parameters:  
- file: The CSV or Excel file (required)  
- validationLevel: strict|lenient (optional, default: strict)

### Response Format

{  
 "operationId": "bulk-upload-123456",  
 "status": "completed",  
 "summary": {  
 "totalRecords": 1000,  
 "successfulRecords": 950,  
 "failedRecords": 50  
 },  
 "errors": [  
 {  
 "row": 15,  
 "field": "price",  
 "value": "invalid",  
 "message": "Price must be a valid decimal number"  
 }  
 ],  
 "processedAt": "2025-01-26T10:30:00Z"  
}

### Security Requirements

* **File Type Validation**: Reject non-CSV/Excel files
* **Content Validation**: Scan for malicious content
* **Size Limits**: Enforce file size restrictions
* **Input Sanitization**: Clean all text inputs

────────────────────────────────────────────────────────────────────────────────

## Non-Functional Requirements

### Performance

* **Upload Response Time**: <5 seconds for file validation
* **Processing Time**: <30 seconds for 1000 records
* **Memory Usage**: <100MB for largest supported file
* **Concurrent Uploads**: Support 5 simultaneous operations

### Reliability

* **Data Integrity**: No partial record commits
* **Error Recovery**: Graceful handling of processing failures
* **Validation Accuracy**: 100% consistent with individual record validation
* **File Corruption Handling**: Detect and report corrupted files

### Scalability

* **Record Limit**: 10,000 books per file (v1.0)
* **File Size Limit**: 10MB (v1.0)
* **Future Growth**: Architecture supports increasing limits

────────────────────────────────────────────────────────────────────────────────

## User Experience Requirements

### Error Handling

1. **Clear Error Messages**: Human-readable validation feedback
2. **Row-Level Details**: Specific location of each error
3. **Bulk Error Summary**: Overview of common issues
4. **Actionable Feedback**: Suggestions for fixing common problems

### Progress Indication

1. **Upload Feedback**: Immediate response on file receipt
2. **Processing Status**: Progress updates for large files
3. **Completion Notice**: Clear success/failure notification

────────────────────────────────────────────────────────────────────────────────

## Implementation Phases

### Phase 1: Core Functionality (Week 1-2)

* [ ] CSV file upload and parsing
* [ ] Basic validation and error reporting
* [ ] Simple response format
* [ ] Unit and integration tests

### Phase 2: Excel Support (Week 3)

* [ ] Excel file parsing (.xlsx, .xls)
* [ ] Enhanced validation logic
* [ ] Improved error reporting

### Phase 3: Advanced Features (Week 4)

* [ ] Progress tracking for large files
* [ ] Enhanced security validation
* [ ] Performance optimization
* [ ] Comprehensive error handling

────────────────────────────────────────────────────────────────────────────────

## Risk Analysis

### Technical Risks

|  |  |  |
| --- | --- | --- |
| **Risk** | **Impact** | **Mitigation** |
| Memory Usage | High | Stream processing, chunked uploads |
| File Corruption | Medium | Robust parsing with error recovery |
| Performance Degradation | Medium | Background processing, rate limiting |

### Business Risks

|  |  |  |
| --- | --- | --- |
| **Risk** | **Impact** | **Mitigation** |
| User Adoption | Medium | Clear documentation, example files |
| Data Quality Issues | High | Comprehensive validation, error reporting |
| Support Overhead | Low | Self-service error resolution |

────────────────────────────────────────────────────────────────────────────────

## Success Criteria

### Launch Criteria

* [ ] All user stories implemented and tested
* [ ] Performance benchmarks met
* [ ] Security review completed
* [ ] Documentation and examples available

### Post-Launch Metrics (30 days)

* **Usage**: >50% of enterprise users try bulk upload
* **Success Rate**: >95% of properly formatted files process successfully
* **Performance**: <30 second processing time for 1000 records
* **Support**: <10% of uploads require user support

────────────────────────────────────────────────────────────────────────────────

## Future Enhancements

### Version 2.0 Considerations

1. **Bulk Export**: Export books to CSV/Excel formats
2. **Scheduled Imports**: Automated processing of uploaded files
3. **Advanced Validation**: ISBN database lookup, duplicate detection
4. **Real-time Progress**: WebSocket-based progress updates
5. **Template Generator**: Create sample files for users

### Integration Opportunities

1. **External Catalogs**: Direct integration with book databases
2. **Cloud Storage**: Import from Google Drive, Dropbox
3. **API Partnerships**: Integration with library management systems

────────────────────────────────────────────────────────────────────────────────

## Appendix

### Sample File Formats

#### CSV Format

name,author,price,isbn,category  
"The Great Gatsby","F. Scott Fitzgerald",12.99,"978-0743273565","Fiction"  
"To Kill a Mockingbird","Harper Lee",13.99,"978-0446310789","Fiction"  
"1984","George Orwell",14.99,"978-0452284234","Dystopian"

#### Excel Format

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Author** | **Price** | **ISBN** | **Category** |
| The Great Gatsby | F. Scott Fitzgerald | $12.99 | 978-0743273565 | Fiction |
| To Kill a Mockingbird | Harper Lee | $13.99 | 978-0446310789 | Fiction |
| 1984 | George Orwell | $14.99 | 978-0452284234 | Dystopian |

### Validation Error Examples

{  
 "errors": [  
 {  
 "row": 5,  
 "field": "name",  
 "value": "",  
 "message": "Book name cannot be empty"  
 },  
 {  
 "row": 12,  
 "field": "price",  
 "value": "abc",  
 "message": "Price must be a valid decimal number"  
 }  
 ]  
}