API Specification

Library Book Loan System

Document Information

- **Document Version**: 1.0
- **Date**: July 13, 2025
- **Project**: Library Book Loan Automation System
- **API Version**: v1
- **Base URL**: `https://api.library.example.com/v1`
- **Status**: Draft

---

1. API Overview

1.1 Introduction

The Library Book Loan System API provides RESTful endpoints for managing book loans, member accounts, inventory, and fine processing. The API follows REST principles and uses JSON for data exchange.

1.2 API Design Principles

- **RESTful Architecture**: Standard HTTP methods and status codes
- **Resource-Based URLs**: Clear, hierarchical resource naming
- **Stateless**: Each request contains all necessary information
- **Consistent Response Format**: Standardized error and success responses
- **Versioning**: URL-based versioning for backward compatibility

1.3 Base URL Structure

```
https://api.library.example.com/v1/{resource}
```

---

2. Authentication & Authorization

2.1 Authentication Method

- **Type**: JWT (JSON Web Token) Bearer Authentication

- **Header**: `Authorization: Bearer <token>`
- **Token Expiry**: 24 hours (configurable)
- **Refresh Token**: Available for token renewal

2.2 Authentication Endpoints

POST /auth/login

Authenticate user and obtain access token.

Request Body:

```
json

{
  "username": "string",
  "password": "string",
  "userType": "staff" | "member"
}
```

Response (200 OK):

```
json

{
  "success": true,
  "data": {
    "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
    "refreshToken": "dGhpcyBpcyBhIHJlZnJlc2ggdG9rZW4...",
    "expiresIn": 86400,
    "user": {
      "id": 123,
      "username": "john.doe",
      "role": "LIBRARIAN",
      "firstName": "John",
      "lastName": "Doe"
    }
  }
}
```

POST /auth/refresh

Refresh access token using refresh token.

Request Body:

```
json

{
  "refreshToken": "string"
}
```

POST /auth/logout

Invalidate current session.

Request Headers:

```
Authorization: Bearer <token>
```

2.3 Authorization Levels

- **Public**: No authentication required
- **Member**: Member authentication required
- **Staff**: Staff authentication required
- **Admin**: Administrative privileges required

---

3. Common Response Formats

3.1 Success Response Format

```
json
```

```
{
  "success": true,
  "data": {},
  "meta": {
    "timestamp": "2025-07-13T10:30:00Z",
    "requestId": "req_123456789"
  }
}
```

3.2 Error Response Format

```
json

{
  "success": false,
  "error": {
    "code": "ERROR_CODE",
    "message": "Human readable error message",
    "details": "Additional error details",
    "field": "fieldName"
  },
  "meta": {
    "timestamp": "2025-07-13T10:30:00Z",
    "requestId": "req_123456789"
  }
}
```

3.3 Pagination Format

```
json
```

```
{
  "success": true,
  "data": [],
  "pagination": {
    "page": 1,
    "limit": 20,
    "total": 150,
    "totalPages": 8,
    "hasNext": true,
    "hasPrev": false
  }
}
```

---

4. Loan Management API

4.1 Process Loan Request

POST /loans

Create a new loan request and process through validation workflow.

Authentication: Staff Required

Request Body:

```
json

{
  "memberId": "MB123456",
  "bookIdentifier": "978-0123456789",
  "bookIdentifierType": "isbn" | "barcode" | "title",
  "notes": "Optional notes"
}
```

Response (201 Created):

```
json
```

```
{
  "success": true,
  "data": {
    "loanId": 12345,
    "loanNumber": "LN2025071300001",
    "member": {
      "id": 123,
      "memberId": "MB123456",
      "name": "John Smith",
      "email": "john.smith@email.com"
    },
    "book": {
      "id": 456,
      "title": "The Great Gatsby",
      "author": "F. Scott Fitzgerald",
      "isbn": "978-0123456789"
    },
    "bookCopy": {
      "id": 789,
      "barcode": "BC001234567",
      "copyNumber": "001"
    },
    "loanDate": "2025-07-13",
    "dueDate": "2025-07-27",
    "status": "ACTIVE",
    "validationResults": {
      "bookAvailable": true,
      "memberValid": true,
      "finesAcceptable": true,
      "canBorrow": true
    }
  }
}
```


Error Response (400 Bad Request):

```
json
```

```
{
  "success": false,
  "error": {
    "code": "LOAN_VALIDATION_FAILED",
    "message": "Loan request cannot be processed",
    "details": "Member has outstanding fines exceeding $10.00",
    "validationErrors": [
      {
        "stage": "fine_check",
        "passed": false,
        "reason": "Outstanding fines: $15.50 exceeds limit of $10.00",
        "resolutionSuggestion": "Process fine payment before borrowing"
      }
    ]
  }
}
```


4.2 Get Loan Details

GET /loans/{loanId}

Retrieve detailed information about a specific loan.

Authentication: Staff or Member (own loans only)

Response (200 OK):

```
json
```

```
{
  "success": true,
  "data": {
    "id": 12345,
    "loanNumber": "LN2025071300001",
    "member": {
      "id": 123,
      "memberId": "MB123456",
      "name": "John Smith"
    },
    "book": {
      "id": 456,
      "title": "The Great Gatsby",
      "author": "F. Scott Fitzgerald",
      "isbn": "978-0123456789"
    },
    "bookCopy": {
      "id": 789,
      "barcode": "BC001234567",
      "condition": "GOOD"
    },
    "loanDate": "2025-07-13",
    "dueDate": "2025-07-27",
    "returnDate": null,
    "status": "ACTIVE",
    "renewalCount": 0,
    "maxRenewals": 2,
    "checkedOutBy": {
      "id": 10,
      "name": "Library Staff"
    },
    "notes": "Member requested specific copy"
  }
}
```


4.3 Return Book

PUT /loans/{loanId}/return

Process book return and update loan status.

Authentication: Staff Required

Request Body:

```
json

{
  "returnDate": "2025-07-25",
  "condition": "GOOD" | "FAIR" | "POOR" | "DAMAGED",
  "notes": "Book returned in good condition"
}
```

Response (200 OK):

```
json

{
  "success": true,
  "data": {
    "loanId": 12345,
    "status": "RETURNED",
    "returnDate": "2025-07-25",
    "daysOnLoan": 12,
    "wasOverdue": false,
    "finesGenerated": [],
    "bookCopy": {
      "id": 789,
      "status": "AVAILABLE",
      "condition": "GOOD"
    }
  }
}
```

4.4 Renew Loan

PUT /loans/{loanId}/renew

Renew an existing loan if eligible.

Authentication: Staff or Member (own loans only)

Response (200 OK):

```
json

{
  "success": true,
  "data": {
    "loanId": 12345,
    "newDueDate": "2025-08-10",
    "renewalCount": 1,
    "maxRenewals": 2,
    "canRenewAgain": true
  }
}
```

4.5 List Loans

GET /loans

Retrieve list of loans with filtering and pagination.

Authentication: Staff Required

Query Parameters:

- `memberId` (string): Filter by member ID
- `status` (string): Filter by loan status
- `overdue` (boolean): Filter overdue loans
- `page` (integer): Page number (default: 1)
- `limit` (integer): Items per page (default: 20)
- `sortBy` (string): Sort field (default: loanDate)
- `sortOrder` (string): asc|desc (default: desc)

Response (200 OK):

```
json
```

```
{
  "success": true,
  "data": [
    {
      "id": 12345,
      "loanNumber": "LN2025071300001",
      "member": {
        "id": 123,
        "name": "John Smith"
      },
      "book": {
        "title": "The Great Gatsby",
        "author": "F. Scott Fitzgerald"
      },
      "loanDate": "2025-07-13",
      "dueDate": "2025-07-27",
      "status": "ACTIVE",
      "isOverdue": false,
      "daysRemaining": 14
    }
  ],
  "pagination": {
    "page": 1,
    "limit": 20,
    "total": 150,
    "totalPages": 8
  }
}
```


---

5. Book Management API

5.1 Check Book Availability

GET /books/{identifier}/availability

Check if a book is available for loan.

Authentication: Public

Path Parameters:

- `identifier`: Book ISBN, barcode, or title

Query Parameters:

- `type`: isbn|barcode|title (default: isbn)

Response (200 OK):

```json
{
  "success": true,
  "data": {
    "book": {
      "id": 456,
      "isbn": "978-0123456789",
      "title": "The Great Gatsby",
      "author": "F. Scott Fitzgerald",
      "publisher": "Scribner"
    },
    "availability": {
      "isAvailable": true,
      "totalCopies": 3,
      "availableCopies": 2,
      "checkedOutCopies": 1,
      "reservedCopies": 0,
      "nextAvailableDate": null
    },
    "copies": [
      {
        "id": 789,
        "barcode": "BC001234567",
        "status": "AVAILABLE",
        "condition": "GOOD",
        "location": "Fiction-A-15"
      }
    ]
  }
}
```

5.2 Search Books

GET /books/search

Search for books using various criteria.

Authentication: Public
Query Parameters:

- `q` (string): General search query
- `title` (string): Title search
- `author` (string): Author search
- `isbn` (string): ISBN search
- `category` (string): Category filter
- `available` (boolean): Only available books
- `page` (integer): Page number
- `limit` (integer): Items per page

Response (200 OK):

```
json
```

```
{
  "success": true,
  "data": [
    {
      "id": 456,
      "isbn": "978-0123456789",
      "title": "The Great Gatsby",
      "author": "F. Scott Fitzgerald",
      "publisher": "Scribner",
      "publicationDate": "1925-04-10",
      "category": "Fiction",
      "availability": {
        "isAvailable": true,
        "availableCopies": 2,
        "totalCopies": 3
      },
      "location": "Fiction-A-15"
    }
  ],
  "pagination": {
    "page": 1,
    "limit": 20,
    "total": 50
  }
}
```


5.3 Get Book Details

GET /books/{bookId}

Retrieve detailed information about a specific book.

Authentication: Public

Response (200 OK):

```
json
```

```
{
  "success": true,
  "data": {
    "id": 456,
    "isbn": "978-0123456789",
    "isbn13": "978-0123456789",
    "title": "The Great Gatsby",
    "subtitle": null,
    "author": "F. Scott Fitzgerald",
    "coAuthors": null,
    "publisher": "Scribner",
    "publicationDate": "1925-04-10",
    "edition": "First Edition",
    "language": "English",
    "pages": 180,
    "description": "A classic American novel about the Jazz Age...",
    "category": {
      "id": 1,
      "name": "Fiction",
      "deweyRange": "800-899"
    },
    "keywords": "american literature, jazz age, 1920s",
    "deweyDecimal": "813.52",
    "copies": [
      {
        "id": 789,
        "barcode": "BC001234567",
        "copyNumber": "001",
        "status": "AVAILABLE",
        "condition": "GOOD",
        "location": "Fiction-A-15",
        "acquisitionDate": "2023-01-15"
      }
    ],
    "availability": {
      "isAvailable": true,
      "totalCopies": 3,
      "availableCopies": 2,
```

```
    "checkedOutCopies": 1
  }
 }
}
```

---

6. Member Management API

6.1 Validate Member

GET /members/{memberId}/validation

Validate member eligibility for borrowing.

Authentication: Staff Required

Response (200 OK):

```
json
```

```
{
  "success": true,
  "data": {
    "member": {
      "id": 123,
      "memberId": "MB123456",
      "name": "John Smith",
      "email": "john.smith@email.com",
      "membershipType": "Standard Adult"
    },
    "validation": {
      "isValid": true,
      "membershipStatus": "ACTIVE",
      "membershipExpiry": "2025-12-31",
      "isExpired": false,
      "outstandingFines": 5.50,
      "fineThreshold": 10.00,
      "finesExceedLimit": false,
      "canBorrow": true,
      "activeLoans": 2,
      "maxLoans": 5,
      "canBorrowMore": true
    },
    "issues": []
  }
}
```


Response when validation fails (200 OK):

```
json
```

```
{
  "success": true,
  "data": {
    "member": {
      "id": 123,
      "memberId": "MB123456",
      "name": "John Smith"
    },
    "validation": {
      "isValid": false,
      "canBorrow": false,
      "outstandingFines": 15.50,
      "fineThreshold": 10.00,
      "finesExceedLimit": true
    },
    "issues": [
      {
        "type": "EXCESSIVE_FINES",
        "message": "Outstanding fines ($15.50) exceed limit ($10.00)",
        "resolutionRequired": "Process fine payment",
        "canResolve": true
      }
    ]
  }
}
```


6.2 Get Member Account

GET /members/{memberId}

Retrieve complete member account information.

Authentication: Staff or Member (own account only)

Response (200 OK):

```
json
```

```
{
  "success": true,
  "data": {
    "id": 123,
    "memberId": "MB123456",
    "firstName": "John",
    "lastName": "Smith",
    "email": "john.smith@email.com",
    "phone": "+1-555-0123",
    "address": {
      "line1": "123 Main Street",
      "line2": "Apt 4B",
      "city": "Anytown",
      "state": "NY",
      "postalCode": "12345",
      "country": "USA"
    },
    "dateOfBirth": "1985-03-15",
    "membership": {
      "type": "Standard Adult",
      "startDate": "2025-01-01",
      "endDate": "2025-12-31",
      "status": "ACTIVE"
    },
    "borrowingLimits": {
      "maxBooks": 5,
      "loanPeriodDays": 14,
      "maxRenewals": 2
    },
    "currentLoans": [
      {
        "id": 12345,
        "book": {
          "title": "The Great Gatsby",
          "author": "F. Scott Fitzgerald"
        },
        "loanDate": "2025-07-13",
        "dueDate": "2025-07-27",
```

```
      "status": "ACTIVE"
    }
   ],
   "outstandingFines": 5.50,
   "borrowingHistory": {
    "totalLoans": 45,
    "booksCurrentlyBorrowed": 2,
    "overdueCount": 1,
    "averageLoanDuration": 12
   }
  }
}
```


6.3 Update Member Information

PUT /members/{memberId}

Update member account information.

Authentication: Staff or Member (own account only)

Request Body:


```
json

{
  "firstName": "John",
  "lastName": "Smith",
  "email": "john.smith@newemail.com",
  "phone": "+1-555-0124",
  "address": {
   "line1": "456 Oak Avenue",
   "line2": null,
   "city": "Newtown",
   "state": "NY",
   "postalCode": "12346"
  }
}
```


---

7. Fine Management API

7.1 Get Member Fines

GET /members/{memberId}/fines

Retrieve all fines for a specific member.

Authentication: Staff or Member (own fines only)

Query Parameters:

- `status`: unpaid|paid|waived|all (default: all)
- `page`: Page number
- `limit`: Items per page

Response (200 OK):

```
json
```

```
{
  "success": true,
  "data": {
    "totalOutstanding": 15.50,
    "totalPaid": 25.00,
    "fines": [
      {
        "id": 1001,
        "type": "Overdue",
        "amount": 10.50,
        "description": "Overdue fine for 'The Great Gatsby' - 21 days",
        "fineDate": "2025-07-10",
        "dueDate": "2025-07-24",
        "status": "UNPAID",
        "loan": {
          "id": 12345,
          "book": {
            "title": "The Great Gatsby",
            "author": "F. Scott Fitzgerald"
          }
        }
      },
      {
        "id": 1002,
        "type": "Processing Fee",
        "amount": 5.00,
        "description": "Administrative processing fee",
        "fineDate": "2025-07-12",
        "status": "UNPAID"
      }
    ]
  },
  "pagination": {
    "page": 1,
    "limit": 20,
    "total": 2
```

```
  }
}
```

7.2 Process Fine Payment

POST /fines/payment

Process payment for outstanding fines.

Authentication: Staff Required

Request Body:

```
json
{
  "memberId": "MB123456",
  "paymentMethod": "CASH" | "CREDIT_CARD" | "DEBIT_CARD" | "ONLINE",
  "amount": 15.50,
  "fines": [
    {
      "fineId": 1001,
      "amountToPay": 10.50
    },
    {
      "fineId": 1002,
      "amountToPay": 5.00
    }
  ],
  "transactionId": "txn_123456789",
  "notes": "Full payment for outstanding fines"
}
```

Response (201 Created):

```
json
```

```
{
  "success": true,
  "data": {
    "paymentId": 5001,
    "paymentNumber": "PAY2025071300001",
    "amount": 15.50,
    "paymentMethod": "CASH",
    "paymentDate": "2025-07-13T10:30:00Z",
    "status": "COMPLETED",
    "member": {
      "id": 123,
      "memberId": "MB123456",
      "name": "John Smith"
    },
    "finesCleared": [
      {
        "fineId": 1001,
        "amountPaid": 10.50,
        "newStatus": "PAID"
      },
      {
        "fineId": 1002,
        "amountPaid": 5.00,
        "newStatus": "PAID"
      }
    ],
    "remainingBalance": 0.00,
    "receipt": {
      "receiptNumber": "REC2025071300001",
      "downloadUrl": "/receipts/REC2025071300001.pdf"
    }
  }
}
```


7.3 Calculate Overdue Fines

POST /fines/calculate

Calculate potential fines for overdue items.

Authentication: Staff Required

Request Body:

```json
{
  "loanId": 12345,
  "calculationDate": "2025-07-13"
}
```

Response (200 OK):

```json
```

```
{
  "success": true,
  "data": {
    "loanId": 12345,
    "dueDate": "2025-06-27",
    "calculationDate": "2025-07-13",
    "daysOverdue": 16,
    "fineRate": 0.50,
    "calculatedFine": 8.00,
    "existingFines": 2.50,
    "additionalFineOwed": 5.50,
    "breakdown": [
      {
        "period": "2025-06-28 to 2025-07-05",
        "days": 8,
        "rate": 0.50,
        "amount": 4.00
      },
      {
        "period": "2025-07-06 to 2025-07-13",
        "days": 8,
        "rate": 0.50,
        "amount": 4.00
      }
    ]
  }
}
```


---

8. System Configuration API

8.1 Get System Configuration

GET /config

Retrieve system configuration settings.

Authentication: Admin Required

Response (200 OK):

```json
{
  "success": true,
  "data": {
    "fineThreshold": 10.00,
    "defaultLoanPeriod": 14,
    "maxRenewals": 2,
    "overdueFineRate": 0.50,
    "maxBooksPerMember": 5,
    "emailNotifications": true,
    "smsNotifications": false,
    "autoCalculateFines": true,
    "gracePeriodDays": 1
  }
}
```

8.2 Update Configuration

PUT /config/{configKey}

Update specific configuration setting.

Authentication: Admin Required
Request Body:

```json
{
  "value": "10.00"
}
```

---

9. Reporting API

9.1 Loan Statistics

GET /reports/loans

Generate loan statistics report.

Authentication: Staff Required

Query Parameters:

- `startDate`: Start date (YYYY-MM-DD)
- `endDate`: End date (YYYY-MM-DD)
- `groupBy`: day|week|month
- `format`: json|csv|pdf

Response (200 OK):

```
json
```

```
{
  "success": true,
  "data": {
    "period": {
      "startDate": "2025-07-01",
      "endDate": "2025-07-13",
      "days": 13
    },
    "summary": {
      "totalLoans": 245,
      "totalReturns": 198,
      "currentActiveLoans": 47,
      "overdueLoans": 12,
      "averageLoansPerDay": 18.8
    },
    "dailyStats": [
      {
        "date": "2025-07-13",
        "loansIssued": 22,
        "booksReturned": 18,
        "newOverdue": 2
      }
    ],
    "popularBooks": [
      {
        "bookId": 456,
        "title": "The Great Gatsby",
        "author": "F. Scott Fitzgerald",
        "loanCount": 8
      }
    ]
  }
}
```


---

10. Error Codes

10.1 Authentication Errors

- `AUTH_TOKEN_REQUIRED`: Authentication token is required
- `AUTH_TOKEN_INVALID`: Invalid or expired token
- `AUTH_INSUFFICIENT_PERMISSIONS`: User lacks required permissions
- `AUTH_LOGIN_FAILED`: Invalid credentials provided

10.2 Validation Errors

- `VALIDATION_FAILED`: Request validation failed
- `BOOK_NOT_AVAILABLE`: Requested book is not available
- `MEMBER_INVALID`: Member account is invalid or expired
- `FINES_EXCEED_LIMIT`: Outstanding fines exceed borrowing limit
- `LOAN_LIMIT_EXCEEDED`: Member has reached maximum loan limit

10.3 Business Logic Errors

- `LOAN_ALREADY_EXISTS`: Active loan already exists for this book copy
- `BOOK_NOT_FOUND`: Specified book could not be found
- `MEMBER_NOT_FOUND`: Specified member could not be found
- `PAYMENT_FAILED`: Payment processing failed
- `INSUFFICIENT_PAYMENT`: Payment amount is insufficient

10.4 System Errors

- `INTERNAL_SERVER_ERROR`: Unexpected server error occurred
- `DATABASE_ERROR`: Database operation failed
- `EXTERNAL_SERVICE_ERROR`: External service integration failed
- `RATE_LIMIT_EXCEEDED`: API rate limit exceeded

---

11. Rate Limiting

11.1 Rate Limits

- **Authentication**: 10 requests per minute per IP
- **General API**: 100 requests per minute per authenticated user
- **Search**: 50 requests per minute per IP
- **Reports**: 10 requests per minute per user

11.2 Rate Limit Headers

```
X-RateLimit-Limit: 100
X-RateLimit-Remaining: 85
X-RateLimit-Reset: 1625140800
X-RateLimit-Window: 60
```

---

12. API Versioning

12.1 Versioning Strategy

- **URL-based versioning**: `/v1/`, `/v2/`
- **Backward compatibility**: Minimum 12 months support
- **Deprecation notice**: 6 months advance notice
- **Version header**: `API-Version: 1.0`

12.2 Version History

- **v1.0**: Initial API release
- **v1.1**: Added fine calculation endpoints (planned)
- **v2.0**: Enhanced search capabilities (planned)

---

13. SDKs and Integration

13.1 Available SDKs

- **JavaScript/Node.js**: npm package `@library/api-client`
- **Python**: pip package `library-api-client`
- **Java**: Maven dependency `com.library:api-client`
- **PHP**: Composer package `library/api-client`

13.2 Webhook Events

- `loan.created`: New loan issued
- `loan.returned`: Book returned
- `fine.created`: New fine generated
- `payment.completed`: Fine payment processed
- `member.suspended`: Member account suspended

This API specification provides comprehensive coverage of all system functionality with detailed request/response examples and proper error handling.