**API Endpoints Documentation**

This document provides a comprehensive guide to the API endpoints for managing books, users, and transactions in the system.

**Base URL**

All endpoints are prefixed with /react\_task.

**User Management (userRoutes.js)**

**Base URL:** /react\_task/users

1. **Get All Users**
   * **Endpoint:** GET /get\_users
   * **Access:** Admin
   * **Description:** Retrieve all users.
   * **Response:**
     + **200 OK:** List of users.
     + **500 Internal Server Error:** Server error.
2. **Register a User**
   * **Endpoint:** POST /register
   * **Access:** Public
   * **Description:** Register a new user.
   * **Request Body:**

{

"name": "User Name",

"email\_id": "user@example.com",

"mobile": "1234567890",

"role": "user",

"password": "password"

}

* + **Response:**
    - **200 OK:** User registered successfully.
    - **400 Bad Request:** Validation error.
    - **409 Conflict:** Email already exists.
    - **500 Internal Server Error:** Server error.

1. **Login a User**
   * **Endpoint:** POST /login
   * **Access:** Public
   * **Description:** Login a user.
   * **Request Body:**

{

"email\_id": "user@example.com",

"password": "password"

}

* + **Response:**
    - **200 OK:** User logged in successfully, returns token and user details.
    - **400 Bad Request:** Validation error.
    - **401 Unauthorized:** Invalid email or password.
    - **500 Internal Server Error:** Server error.

**Book Management (bookRoutes.js)**

**Base URL:** /react\_task/books

1. **Create a Book**
   * **Endpoint:** POST /
   * **Access:** Admin
   * **Description:** Create a new book.
   * **Request Body:**

{

"title": "Book Title",

"author": "Author Name",

"count": 5,

"available": true

}

* + **Response:**
    - **201 Created:** Book created successfully.
    - **400 Bad Request:** Validation error.

1. **Update a Book**
   * **Endpoint:** PUT /:id
   * **Access:** Admin
   * **Description:** Update an existing book by ID.
   * **Request Body:**

{

"title": "Updated Title",

"author": "Updated Author",

"available": false

}

* + **Response:**
    - **200 OK:** Book updated successfully.
    - **404 Not Found:** Book not found.

1. **Delete a Book**
   * **Endpoint:** DELETE /:id
   * **Access:** Admin
   * **Description:** Delete a book by ID.
   * **Response:**
     + **200 OK:** Book deleted successfully.
     + **404 Not Found:** Book not found.
2. **Get All Books**
   * **Endpoint:** GET /
   * **Access:** Logged in user
   * **Description:** Retrieve all books.
   * **Response:**
     + **200 OK:** List of books.
3. **Get Book by ID**
   * **Endpoint:** GET /:id
   * **Access:** Admin
   * **Description:** Retrieve a book by ID.
   * **Response:**
     + **200 OK:** Book details.
     + **404 Not Found:** Book not found.

**Transaction Management (transactionRoutes.js)**

**Base URL:** /react\_task/transactions

1. **Create a Transaction**
   * **Endpoint:** POST /create
   * **Access:** Admin
   * **Description:** Create a new transaction (borrow or return).
   * **Request Body:**

{

"userId": "UserId",

"bookId": "BookId",

"dueDate": "YYYY-MM-DD",

"transactionType": "borrowed"

}

* + **Response:**
    - **201 Created:** Transaction created successfully.
    - **400 Bad Request:** Invalid transaction type or validation error.
    - **404 Not Found:** User or Book not found.

1. **Get Transactions by User ID**
   * **Endpoint:** GET /user/:userId
   * **Access:** Logged user
   * **Description:** Retrieve transactions for a specific user.
   * **Response:**
     + **200 OK:** List of transactions.
     + **404 Not Found:** No transactions found for this user.
2. **Get Transactions by Book ID and Type**
   * **Endpoint:** GET /bookId/:bookId/transactionType/:transactionType
   * **Access:** Admin
   * **Description:** Retrieve transactions for a specific book and type (borrowed or returned).
   * **Response:**
     + **200 OK:** List of transactions.
     + **500 Internal Server Error:** Failed to fetch transactions.
3. **Toggle Transaction Type**
   * **Endpoint:** PUT /toggle/:transactionId
   * **Access:** Admin
   * **Description:** Toggle the transaction type (borrowed or returned) for a specific transaction.
   * **Request Body:**

{

"transactionType": "returned"

}

* + **Response:**
    - **200 OK:** Transaction updated successfully.
    - **404 Not Found:** Transaction not found.
    - **500 Internal Server Error:** Server error.