By Swarnava Chakraborty

JBDL-60

Digital Library

4th September 2023

OVERVIEW

GOALS

1. Create an ER diagram based on the database schema
2. Create API/endpoints as per the specified controllers
3. Adding Spring Security [Hold]
4. Adding Redis as database query cache [Hold]
5. Adding database seed file [Hold]

SPECIFICATIONS

Entities:

1. Book
2. Author
3. User
4. Student
5. Card
6. Transaction

Database Schema

Author

1. Id
2. Name
3. Email
4. Age
5. country

Book

1. Id
2. Name
3. Author_ID(Mapping)
4. Number of Pages
5. Language
6. Available
7. Genre
8. ISBN Number
9. Published Date

User [Hold]

1. Id
2. Name
3. USername
4. Email
5. Password
6. Authority

Student

1. Id
2. Age
3. Name
4. Country
5. Email
6. Phone Number
7. CreatedOn
8. UpdatedOn
9. CardID

Card

1. Id
2. Status
3. Email
4. ValidUpto
5. CreatedOn

6. UpdatedOn

Transaction

1. Id
2. CardId
3. BookID
4. TransactionDAte
5. BookDueDate
6. IsIssued
7. IsReturned
8. FineAmount
9. Status
10. CreatedOn
11. UpdatedOn

Entities Relation

Source	Destination	Relation
Student	Card	1-1
Book	Author	~~1-1~~ 1-N author / book
Book	Transaction	1-N
Card	Transaction	1-N

Controllers/API

Student Controller

RequestMapping: /student/<endpoint>

1. CRUD API for Student

Book Controller

RequestMapping: /book/<endpoint>

1. CRUD API for Book

Author Controller

RequestMapping: /author/<endpoint>

1. CRUD API for Author

Transaction Controller

RequestMapping: /transact/<endpoint>

1. Issue Book(Parameter: BookId, CardId
    a. If the card is active
    b. If the book is available
    c. Each card can have utmost 3 books being issued.
    d. If all of the above is ok, then issue the book
        i. Create a new transaction
        ii. Mark the book as Unavailable
        iii. <Later>: Drop a kafka message for the email notification to be sent out that the book is issued with the details of the book and the last due date.
    e. If any of the above steps return an error, insert the transaction with status as failure
2. Return (Parameter: BookId, CardID)
    a. If the book id and the card id are a valid combination
    b. If the card is active
    c. If all the above are ok, then process the return
        i. Mark the book as available
        ii. Each card can have utmost 3 books
        iii. If the todays date s higher than the due date, calculate the fine and store it
        iv. In the transaction table, set the card_id as null
3. Get details of a transaction based on id
4. Get details of all transaction based on book_id
5. Get details of all transaction based on card_id

```
if (isCardActive &&
    !isBookAvailable &&
  totalIssuedBooks > 0 &&
  totalIssuedBooks <= 3 &&
    !isBookReturned &&
     isBookIssued &&
transactionStatus.equals(TransactionStatus.ISSUED))
```

Report Controller [Optional]

RequestMapping: /report/<endpoint>

1. All books issued in between a date range
2. Total fine collected in between a date range
3. Total Students signed up on a dat

4. Number of books returned in a date range
5. List of all active students
6. List of students with inactive card
7. List of Cards with student name who have 3 books issued
8. All book due to be returned today