QUIXOTE ASSIGNMENT

Introduction :

This the basic library management program which uses
Python, Flask, MongoDB as backend components and AWS - EC2 and Mongo Cloud to deploy the application and the database.

How to Use :

You can just access the endpoints from the program to access all of the different required methods.

Click here for all routes

You can access routes either providing input just after the link or by providing in the form of json object according to the routes.

What does the routes needs to access :

Wherever variable name is written after the endpoint you can just enter it and access the endpoint but for endpoints it's not written in url you need to enter them as a json object.

- "BOOK ISSUED STATUS": "http://52.27.152.231:8080/book-issue-list/<bookname>"
- "BOOKS BY AUTHOR NAME": "http://52.27.152.231:8080/book-by-author/<authorname>"
- "BOOKS BY VARIOUS CONSTRAINTS": "http://52.27.152.231:8080/book-by-constraints/<term>/<authorname>/<low>/<high>"
- "BOOKS ISSUED BY PARTICULAR LEARNER": "http://52.27.152.231:8080reader-issue/<lender>"
- "BOOKS ISSUED IN GIVEN DATE RANGE": "http://52.27.152.231:8080books-issued-in-daterange/<ldate>/<hdate>"
  Provide both Date inputs for the range in<YYYY-MM-DD>Format
- "Books By Name or a Term in Name": "http://52.27.152.231:8080/book-by-term/<term>"
- "TO ISSUE BOOK": "http://52.27.152.231:8080/issue-book"
  Provide Issue Date input in <YYYY-MM-DD> and string Format
- "TO RETURN BOOK": "http://52.27.152.231:8080/return-book"
  Provide Return Date input in <YYYY-MM-DD> and string Format
- "TOTAL RENT GENERATED BY A BOOK":

Program :

```
from flask import Flask, jsonify, request
from flask_pymongo import PyMongo
import os,re,datetime,pymongo

app = Flask(__name__)

app.config['MONGO_DBNAME'] = 'books'
app.config['MONGO_URI'] = 'mongodb+srv://
quixote:Arianna22@cluster0.pzwws.mongodb.net/books?
retryWrites=true&w=majority'



mongo = PyMongo(app)

# Instance for Books Collection
library = mongo.db.Books

# Instance for Transaction Collection
transaction = mongo.db.Transactions


# GET ALL ROUTES FOR REQUIRED USECASE

@app.route('/routes', methods=['GET'])
def getAllRoutes():

    result = []
    result.append({'Books By Name or a Term in Name':'http://
52.27.152.231:8080/book-by-term/<term>',
```

```
    'BOOKS BY AUTHOR NAME':'http://52.27.152.231:8080/book-by-
author/<authorname>',
    'BOOKS BY VARIOUS COMSTRAINTS':'http://52.27.152.231:8080/
book-by-constraints/<term>/<authorname>/<low>/<high>',
    'TO ISSUE BOOK':'http://52.27.152.231:8080/issue-book',
    'TO RETURN BOOK':'http://52.27.152.231:8080/return-book',
    'BOOK ISSUED STATUS':'http://52.27.152.231:8080/book-issue-list/
<bookname>',
    'BOOKS ISSUED BY PARTICULAR LEARNER':'http://
52.27.152.231:8080reader-issue/<lender>',
    'TOTAL RENT GENERATED BY A BOOK':'http://
52.27.152.231:8080total-rent/<bookname>',
    'BOOKS ISSUED IN GIVEN DATE RANGE':'http://
52.27.152.231:8080books-issued-in-daterange/<ldate>/<hdate>'
     })

    return jsonify({'result' : result})

# GET BOOKS BY BOOK NAME or CERTAIN TERM IN THEIR NAME

@app.route('/book-by-term/<term>', methods=['GET'])
def searchByTerm(term):

    result = []
    s = { 'book_name': { '$regex': '^'+term},
        'book_name': { '$regex': '.'+term},
        'book_name': { '$regex': term}}

    for q in library.find(s):
        result.append({'book_name':q['book_name'],
'category':q['category'],
'rent_per_day':q['rent_per_day'],'author_name':q['author_name']})

    return jsonify({'result' : result})
```

```
# GET BOOKS BY AUTHOR NAME

@app.route('/book-by-author/<authorname>', methods=['GET'])
def searchByAuthor(authorname):

    result = []

    s = { 'author_name':authorname}

    for q in library.find(s):
        result.append({'book_name':q['book_name'],
'category':q['category'],
'rent_per_day':q['rent_per_day'],'author_name':q['author_name']})

    return jsonify({'result' : result})


# GET BOOKS BY MULTIPLE VALUES term + authorname + rentrange
@app.route('/book-by-constraints/<term>/<category>/<low>/<high>',
methods=['GET'])
def searchByMultipleConstraints(term,category,low,high):

    result = []

    s = { 'book_name': { '$regex': '^'+term},
        'book_name': { '$regex': '.'+term},
        'book_name': { '$regex': term},
        'category':category,
        'rent_per_day':{ '$gt' : int(low), '$lt' : int(high)}
        }

    for q in library.find(s):
        result.append({'book_name':q['book_name'],
'category':q['category'],
'rent_per_day':q['rent_per_day'],'author_name':q['author_name']})
```

```
    return jsonify({'result' : result})



# Method for Book Issue - This will set status to 0 - Status Field will be used to check wether book is issued or it's in inventory

@app.route('/issue-book', methods=['POST'])
def issueBook():

    book_name = request.get_json("book_name")
    lender_name = request.get_json("lender_name")
    issued_on = request.get_json("issued_on")

    if book_name and lender_name and request.method == 'POST':
        book_id =
transaction.insert_one({'book_name':list(book_name.values())[0],
'lender_name':list(lender_name.values())[1],
                                    'issued_on': str(list(issued_on.values())
[2]),'returned_on':"","status':0, 'total_rent':0 })
        return jsonify("Book is Issued")



# Method for Book Return - This will change status to 1 indicating book has been submitted

@app.route('/return-book',methods=['PUT'])
def returnBook():

    book_name = request.get_json("book_name")
    lender_name = request.get_json("lender_name")
    returned_on = request.get_json("returned_on")



transaction.find_one_and_update({'book_name':list(book_name.values())
[0],'lender_name':list(lender_name.values())[1]},
```

```
    {'$set':{'returned_on':str(list(returned_on.values())[2]),'status':1}})

    return jsonify("Book is returned, Thank You ! Hope you got what you
needed")



# List of people that have issued this book and list of people that has
them issued currently

@app.route('/book-issue-list/<bookname>',methods=['GET'])
def bookIssuedByLearners(bookname):
    s = {'book_name': { '$regex': bookname}}

    total_issue_count = []
    current_issue_count = []
    for q in transaction.find(s):
        # if it is 0 which mean it is currently issued
        if q['status'] is 0:
            print(list(q.values())[5])
            current_issue_count.append(list(q.values())[2]) # As 2nd index in
the dictionary model contains readers name
        # This list will contains all of issue activity
        total_issue_count.append(list(q.values())[2])

    return jsonify('Total Issues', total_issue_count, 'Current Issues',
current_issue_count)

# List Of Books Issued By Particular Learner

@app.route('/reader-issue/<lender>',methods=['GET'])
def learnerIssues(lender):
    s = {'lender_name': { '$regex': lender}}

    books_issued = []
```

```
    for q in transaction.find(s):
        # This list will contains all the books that are currently issued by
this reader
        if q['status'] is 0:
            books_issued.append(list(q.values())[1])  # As 1st index in the
dictionary model contains books name

    return jsonify('Book Issued', books_issued)


# Total Rent Generated By The Book

# @app.route('/total-rent/<bookname>',methods=['GET'])
# def totalRent(lender):

#     return pass



# List of books issued in given date range

@app.route('/books-issued-in-daterange/<ldate>/
<hdate>',methods=['GET'])
def booksByIssueDate(ldate,hdate):
    output = []
    for q in transaction.find():

        if {datetime.datetime.strptime(q['issued_on'], "%Y-%m-%d").date():
{ '$gt' : datetime.datetime.strptime(ldate, "%Y-%m-%d").date(),
                                                        '$lt' :
datetime.datetime.strptime(hdate, "%Y-%m-%d").date()}}:
            output.append({'book_name':q['book_name']})

    # This line is used for filtering out duplicates which were recorded
because multiple people issued same book
    result = [dict(t) for t in {tuple(d.items()) for d in output}]
```

```
    return jsonify(result)



# Project Execution Starts From Here

if __name__ == "__main__":
    app.run(debug=True)
```

Postman API Collections :


```
{
	"info": {
		"_postman_id": "a76917c3-b7ee-4af9-875d-028c276a47e6",
		"name": "Quixote-APIs",
		"schema": "https://schema.getpostman.com/json/collection/v2.1.0/collection.json"
	},
	"item": [
		{
			"name": "Books by name or term in name",
			"request": {
				"method": "GET",
				"header": [],
				"url": {
					"raw": "http://52.27.152.231:8080/book-by-term/<term>",
					"protocol": "http",
					"host": [
						"52",
						"27",
						"152",
						"231"
					],
					"port": "8080",
					"path": [
						"book-by-term",
						"<term>"
					]
				}
			},
			"response": []
		},
```

```
		{
			"name": "Books currently issued by learners",
			"request": {
				"method": "GET",
				"header": [],
				"url": {
					"raw": "http://52.27.152.231:8080/reader-issue/<lender>",
					"protocol": "http",
					"host": [
						"52",
						"27",
						"152",
						"231"
					],
					"port": "8080",
					"path": [
						"reader-issue",
						"<lender>"
					]
				}
			},
			"response": []
		},
		{
			"name": "Books by Author's Name",
			"request": {
				"method": "GET",
				"header": [],
				"url": {
					"raw": "http://52.27.152.231:8080/book-by-author/<authorname>",
					"protocol": "http",
					"host": [
						"52",
						"27",
```

```
                                "152",
                                "231"
                            ],
                            "port": "8080",
                            "path": [
                                "book-by-author",
                                "<authorname>"
                            ]
                        }
                    },
                    "response": []
                },
                {
                    "name": "Books by term in name + category + price
range",
                    "request": {
                        "method": "GET",
                        "header": [],
                        "url": {
                            "raw": "http://52.27.152.231:8080/book-by-
constraints/<term>/<category>/<low>/<high>",
                            "protocol": "http",
                            "host": [
                                "52",
                                "27",
                                "152",
                                "231"
                            ],
                            "port": "8080",
                            "path": [
                                "book-by-constraints",
                                "<term>",
                                "<category>",
                                "<low>",
                                "<high>"
                            ]
```

```
                }
            },
            "response": []
        },
        {
            "name": "To Issue A Book",
            "request": {
                "method": "POST",
                "header": [],
                "url": {
                    "raw": "http://52.27.152.231:8080/issue-
book",
                    "protocol": "http",
                    "host": [
                        "52",
                        "27",
                        "152",
                        "231"
                    ],
                    "port": "8080",
                    "path": [
                        "issue-book"
                    ]
                }
            },
            "response": []
        },
        {
            "name": "To Return a issued book",
            "request": {
                "method": "PUT",
                "header": [],
                "url": {
                    "raw": "http://52.27.152.231:8080/return-
book",
                    "protocol": "http",
```

```
                        "host": [
                                "52",
                                "27",
                                "152",
                                "231"
                        ],
                        "port": "8080",
                        "path": [
                                "return-book"
                        ]
                }
        },
        "response": []
},
{
        "name": "List of readers that have issued the book",
        "request": {
                "method": "GET",
                "header": [],
                "url": {
                        "raw": "http://52.27.152.231:8080/book-
issue-list/<bookname>",
                        "protocol": "http",
                        "host": [
                                "52",
                                "27",
                                "152",
                                "231"
                        ],
                        "port": "8080",
                        "path": [
                                "book-issue-list",
                                "<bookname>"
                        ]
                }
        },
```

```
            "response": []
        },
        {
            "name": "Total Rent Generated By a book",
            "request": {
                "method": "GET",
                "header": [],
                "url": {
                    "raw": "http://52.27.152.231:8080/total-
rent/<bookname>",
                    "protocol": "http",
                    "host": [
                        "52",
                        "27",
                        "152",
                        "231"
                    ],
                    "port": "8080",
                    "path": [
                        "total-rent",
                        "<bookname>"
                    ]
                }
            },
            "response": []
        },
        {
            "name": "Books issued in given time range",
            "request": {
                "method": "GET",
                "header": [],
                "url": {
                    "raw": "http://52.27.152.231:8080/books-
issued-in-daterange/<ldate>/<hdate>",
                    "protocol": "http",
                    "host": [
```

```
                                "52",
                                "27",
                                "152",
                                "231"
                            ],
                            "port": "8080",
                            "path": [
                                "books-issued-in-daterange",
                                "<ldate>",
                                "<hdate>"
                            ]
                        }
                    },
                    "response": []
                }
            ]
        }
```

Submitted By :

Kartikey Sinha