Assignment 9

Complete the same questions twice:Once using **Express.js with MongoDB**(submit your project files)and once using **MongoDB Shell** (submit solutions in a **mongosh-solutions.txt** file):

1. Create an **explicit collection** named **"books"** with a validation rule to ensure that each document **has a non-empty "title" field**. **(0.5 Grade)**
   - URL: POST /collection/books

```
{ "ok": 1 }
```

2. Create an **implicit collection** by inserting data directly into a new collection named **"authors"**. **(0.5 Grade)**
   - URL: POST /collection/authors

```
{
  "name": "Author1",
  "nationality": "British"
}
```

```
{ "acknowledged": true, "insertedId": "64b5c0b2f82c4a765ef0" }
```

3. Create a **capped collection** named "logs" with a size limit of 1MB. **(0.5 Grade)**
   - URL: POST /collection/logs/capped

```
{ "ok": 1 }
```

4. Create an index on the books collection for the title field. **(0.5 Grade)**
   - URL: POST /collection/books/index

```
title_1
```

5. Insert one document into the books collection. **(0.5 Grade)**
   - URL: POST /books

```
{
  "title": "Book1",
  "author": "Ali",
  "year": 1937,
  "genres": ["Fantasy", "Adventure"]
}
```

```
{ "acknowledged": true, "insertedId": "64b5c0b2f82c4a765ef0" }
```

6. Insert multiple documents into the books collection with at **least three records**. **(0.5 Grade)**
   - URL: POST /books/batch

```
[
  {
    "title": "Future",
    "author": "George Orwell",
    "year": 2020,
    "genres": ["Science Fiction"]
  },
  {
    "title": "To Kill a Mockingbird",
    "author": "Harper Lee",
    "year": 1960,
    "genres": ["Classic", "Fiction"]
  },
  {
    "title": "Brave New World",
    "author": "Aldous Huxley",
    "year": 2006,
    "genres": ["Dystopian", "Science Fiction"]
  }
]
```

```
{
  "acknowledged": true,
  "insertedIds": {
    "0": "64b5c2d8a123456ef8901",
    "1": "64b5c2d8a123456ef8902",
    "2": "64b5c2d8a123456ef8903"
  }
}
```

7. Insert a new log into the logs collection. **(0.5 Grade)**
   - URL: POST /logs

```
{
  "book_id": "64b5c2d8a123456ef8914",
  "action": "borrowed"
}
```

```
{ "acknowledged": true, "insertedId": "64b5c0b2f82c4a765ef0" }
```

8. Update the book with title "Future" change the year to be 2022. **(0.5 Grade)**
   - URL: PATCH/books/Future

```
{ "acknowledged": true, "matchedCount": 1, "modifiedCount": 1 }
```

Assignment 9

9. Find a Book with title “Brave New World”. **(0.5 Grade)**
   - URL: GET /books/title => /books/title?title=Brave New World


```
{
  _id: ObjectId('67532515fa2dd78f5b8264ea'),
  title: 'Brave New World',
  author: 'Aldous Huxley',
  year: 1932,
  genres: [
    'Dystopian',
    'Science Fiction'
  ]
}
```


10. Find all books published between 1990 and 2010. **(0.5 Grade)**
    - URL: GET /books/year => /books/year?from=1990&to=2010


```
[
  {
    "title": "Future",
    "author": "George Orwell",
    "year": 2020,
    "genres": ["Science Fiction"]
  },
  {
    "title": "Brave New World",
    "author": "Aldous Huxley",
    "year": 2006,
    "genres": ["Dystopian", "Science Fiction"]
  }
]
```


11. Find books where the genre includes **"Science Fiction".(0.5 Grade)**
    - URL: /books/genre?genre=Science Fiction


```
[
  {
    "title": "Future",
    "author": "George Orwell",
    "year": 2020,
    "genres": ["Science Fiction"]
  },
  {
    "title": "Brave New World",
    "author": "Aldous Huxley",
    "year": 2006,
    "genres": ["Dystopian", "Science Fiction"]
  }
]
```


12. Skip the first two books, limit the results to the next three, sorted by year in descending order. **(0.5 Grade)**
    - URL: GET /books/skip-limit


```
{
  "_id": "64b5c2d8a123456ef8904",
  "title": "The Great Gatsby",
  "author": "F. Scott Fitzgerald",
  "year": 1925,
  "genres": ["Classic", "Fiction"]
},
{
  "_id": "64b5c2d8a123456ef8905",
  "title": "Moby Dick",
  "author": "Herman Melville",
  "year": 1851,
  "genres": ["Adventure", "Classic"]
},
{
  "_id": "64b5c2d8a123456ef8906",
  "title": "War and Peace",
  "author": "Leo Tolstoy",
  "year": 1869,
  "genres": ["Historical Fiction", "Philosophy"]
}
```

Assignment 9

13. Find books where the year field stored as an integer. **(0.5 Grade)**
    - URL: GET /books/year-integer


```
[
  {
    "_id": "64b5c2d8a123456ef8908",
    "title": "Pride and Prejudice",
    "author": "Jane Austen",
    "year": 1813,
    "genres": ["Romance", "Classic"]
  },
  {
    "_id": "64b5c2d8a123456ef8909",
    "title": "The Alchemist",
    "author": "Paulo Coelho",
    "year": 1988,
    "genres": ["Adventure", "Philosophy"]
  }
]
```


14. Find all books where the genres field **does not include** any of the genres "Horror" or "Science Fiction". **(0.5 Grade)**
    - URL: GET /books/exclude-genres


```
[
  {
    "_id": "64b5c2d8a123456ef8912",
    "title": "To Kill a Mockingbird",
    "author": "Harper Lee",
    "year": 1960,
    "genres": ["Classic", "Drama"]
  },
  {
    "_id": "64b5c2d8a123456ef8913",
    "title": "Pride and Prejudice",
    "author": "Jane Austen",
    "year": 1813,
    "genres": ["Romance", "Classic"]
  }
]
```


15. Delete all books published before 2000. **(0.5 Grade)**
    - DELETE: GET /books/before-year?year=2000


```
{ "acknowledged": true, "deletedCount": 2 }
```


16. Using aggregation Functions, Filter books published after 2000 and sort them by year descending. **(0.5 Grade)**
    - URL: GET /books/aggregate1


```
[
  {
    "title": "Future",
    "author": "George Orwell",
    "year": 2020,
    "genres": ["Science Fiction"]
  },
  {
    "title": "Brave New World",
    "author": "Aldous Huxley",
    "year": 2006,
    "genres": ["Dystopian", "Science Fiction"]
  }
]
```

Assignment 9

17. Using aggregation functions, Find all books published after the year **2000**. For each matching book, show only the title, author, and year fields. **(0.5 Grade)**
    - URL: GET /books/aggregate2


```
[
  {
    "title": "The Road",
    "author": "Cormac McCarthy",
    "year": 2006
  },
  {
    "title": "The Kite Runner",
    "author": "Khaled Hosseini",
    "year": 2003
  }
]
```


18. Using aggregation functions,break an array of genres into separate documents. **(0.5 Grade)**
    - URL: GET /books/aggregate3


```
[
  { "title": "The Hobbit", "genres": "Fantasy" },
  { "title": "The Hobbit", "genres": "Adventure" }
]
```


19. Using aggregation functions, Join the books collection with the logs collection. **(1 Grade)**
    - URL: GET /books/aggregate4


```
{
  "action": "borrowed",
  "book_details": [
    {
      "title": "The Road",
      "author": "Cormac McCarthy",
      "year": 2006
    }
  ]
},
{
  "action": "returned",
  "book_details": [
    {
      "title": "The Kite Runner",
      "author": "Khaled Hosseini",
      "year": 2003
    }
  ]
}
```

⚠️ Important Notes about postman

1. **Name the endpoint with a meaningful name like 'Add User', not dummy names.**
2. **Save your changes on each request (ctrl+s).**
3. **Include the Postman collection link (export your Postman collection) in the email with your assignment link**

Bonus (2 Grades)

How to deliver the bonus?

1- Solve the problem Roman to Integer on **LeetCode**

2- Inside your assignment folder, create a **SEPARATE FILE** and name it “bonus.js”

3- Copy the code that you have submitted on the website inside ”bonus.js” file