

Name: Lakshmana Kumar Mettu

Class ID:11

Introduction:

- Performing create, update and delete operations for library management system using meanstack.

Programming elements:

- Mongo-DB
- Java script
- HTML5
- CSS

Tools Required:

- JetBrains WebStorm

Objective:

- Understanding how to create mean stack application through local-host.
- Created database name Library and connected to database using mongodb url.
- Collection name as books created in library database.
- Corresponding code snippets for update and delete functionalities have shown below along with the output snippets.

1.Connection to Mongo DB.

```

*/
var MongoClient = require('mongodb').MongoClient;
var assert = require('assert');
var bodyParser = require("body-parser");
var express = require('express');
var cors = require('cors');
var app = express();
var path = require('path');
var url = 'mongodb+srv://lakshmanakumarreddy:ABcabc012%40*@cluster0-xlaje.mongodb.net/test?retryWrites=true'; //1.Modify
var ObjectID = require('mongodb').ObjectID;

app.use(cors());
app.use(express.static(path.join(__dirname, 'views')));
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: true }));
app.post('/create', function (req, res) {
  MongoClient.connect(url, function(err, db) {
    if(err)
    {
      res.write("Failed. Error while connecting to Database.");
    }
  });
});

```

2.Update functionality

```

app.get('/update/:toBeUpdated_id', function (req, res) {
  MongoClient.connect(url, function(err, db) {
    if (err) throw err;
    //mongo = require('mongodb');
    var dbase = db.db('library');

    var myquery = { _id:ObjectID(req.params.toBeUpdated_id) };
    var newvalues = {$set: req.body };

    dbase.collection('books').updateMany(myquery, newvalues, function(err, res) {
      if (err) throw err;
      console.log(res.result.nModified + " record(s) updated");
      db.close();
    });
  });

  //3.connect to MongoDB. Handle the error and write the logic for updating the selected field
});

```

3.Delete Functionality

```
app.get('/delete/:toBeDeleted_id', function (req, res) {
  MongoClient.connect(url, function(err, db) {
    if (err) {
      throw err;
    }
    mongo = require('mongodb');
    var dbase = db.db('library');
    var id = req.params.id;

    dbase.collection('books').deleteOne({ _id: new mongo.ObjectId(id) }, function(err, obj) {
      if (err) throw err;
      console.log( " document(s) deleted");
    });
  });
});
// 2.Connect to MongoDB . Handle the error and write the logic for deleting the desired book
});
```

4.Console


```
C:\Users\laksh\WebstormProjects\web Technologies\ICP-7\Web_Lesson7_SourceCode\UseCase\MEAN_Stack_App>node .\mongo.js
Example app listening at http://:::8081
Got All Documents
Inserted a document into the books collection.
Got All Documents
document(s) deleted
document(s) deleted
```

5.User Interface

localhost:63342/ICP-7/Web_Lesson7_SourceCode/UseCase/MEAN_Stack_App/home.html

ISBN#	Book Title	Author Name	Modify/Delete
87654321	hhh	rr	Edit Del
12345678	hhhh	eee	Edit Del
87654321	333	ql	Edit Del
87654321	333	ql	Edit Del
87654321	333	ql	Edit Del
87654321	333	ql	Edit Del
12345678	www	www	Edit Del
87654321	ase	lee	Edit Del

[Add a new Book](#)



6.library database with collection name books

The screenshot shows the MongoDB Compass interface for the 'library.books' collection. The top status bar displays 'DOCUMENTS 0' and 'INDEXES 1'. The 'Documents' tab is selected, showing a list of documents. The first document has the following fields: `_id: ObjectId("5d18f272596e582988aa301d")`, `bookName: "hhh"`, `authorName: "rrr"`, and `ISBN: "11234567"`. The second document has the following fields: `_id: ObjectId("5d18f52584e4d2390cc622b6")`, `bookName: "hhhh"`, `authorName: "eee"`, and `ISBN: "12345678"`. The interface also includes a 'FILTER' input, 'OPTIONS', 'FIND', 'RESET', and '...' buttons, and a status bar at the bottom indicating 'Displaying documents 1 - 8 of 8'.

Conclusion:

- Hence learnt how to create meanstack application using mongodb.