BOOKS – DIRECTORY

INHOUSE INTERNSHIP CUM TECHNICAL TRAINING ON DEVOPS REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

Master of Computer Application

SUBMITTED TO

Dayananda Sagar College of Engineering,

Department of MCA

SUBMITTED BY

VARUN M (1DS21MC117)

SUPERVISED BY

PROF. MAHENDRA KUMAR

May – 2023

Dayananda Sagar College of Engineering, Department of MCA, Bangalore

ACKNOWLEDGEMENT

I would like to place on record my deep sense of gratitude to **Dr. Samitha Khaiyum,** HOD-Dept. of MCA, Dayananda Sagar College of Engineering, India for his generous guidance, help and useful suggestions.

Placement Cell - Mr. Guru Venkatesh – VP – Placement & Training – Quality control.

Mr. Rajeev Enhanced Knowledge Works, Bengaluru – Mrs./Ms. Shilpa – Corporate Trainer & Freelance Trainer.

I express my sincere gratitude to Prof. Mahendra Kumar, Dept. of MCA, Dayananda Sagar College of Engineering India, for his stimulating guidance, continuous encouragement and supervision throughout the course of present work.

Signature of Student

Varun M

CERTIFICATE

I hereby certify that the work which is being presented in the MCA, Mini Project Report entitled **"Books Directory",** in partial fulfillment of the requirements for the award of the In-House Internship cum Technical Training and submitted to the **Department of MCA, Dayananda Sagar College of Engineering, Bangalore** is an authentic record of my own work carried out during a period from **24th April 2023 to 9th June 2023** (**IV semester**) under the supervision of **Prof. Mahendra Kumar, MCA Department**.

The matter presented in this Project Report has not been submitted by me for the award of any other degree elsewhere.

Signature of Student

This is to certify that the above statement made by the student(s) is correct to the best of my knowledge.

Signature of Supervisor

Date: **Name & Designation**

Signature of HOD

Dr. Samitha Khaiyum

Table of Contents

INTRODUCTION

It gives users room to share their stories or books with other users, though any user can manage these books, I didn't have security, authorization and authentication in mind when i built this project. So it basically gives its users room to upload/share/manage their books with others (most importantly book summary)

Functionalities: A librarian can do the following things with this Library Management System:

- Display Available Books
- Add New Books
- Delete Books
- Update Books

Approach: We are going to use Body Parser by which we can capture user input values from the form such as Book Name, Author Name, Pages and Price & store them in a collection. Then we are going to send the data of the book to the web page using EJS. EJS is a middleware that makes it easy to send data from your server file (app.js or server.js) to a web page. We are also going to create the Delete Route for deleting books, an Issue Route for issuing the books, and a Return Route for returning the books.

Directories provide names, addresses, affiliations, etc. of people, organizations, or institutions. They can be used to verify addresses, name spellings, and provide contact information. As in other reference sources, directories may be general or focused on a particular subject.

Books play a significant role in our life, especially for children. Reading books increases the knowledge of students, improves their intellect, makes students aware of the various societies, and civilizations across the globe. Moreover, reading books enhances imagination and creativity in the student's mind.

ABSTRACT

Book Management System is a system which maintains the information about the books present in the library, their authors, the members of library to whom books are issued, library staff and all. This is very difficult to organize manually. Maintenance of all this information manually is a very complex task. Owing to the advancement of technology, organization of an Online Library becomes much simple. The Online Library Management has been designed to computerize and automate the operations performed over the information about the members, book issues and returns and all other operations. This computerization of library helps in many instances of its maintenances. It reduces the workload of management as most of the manual work done is reduced.

E-Library Management System is an application which refers to library systems which are generally small or medium in size. It is used by librarian to manage the library using a computerized system where he/she can add new books, videos and Page sources. Books and student maintenance modules are also included in this system which would keep track of the students using the library and also a detailed description about the books a library contains. With this computerized system there will be no loss of book record or member record which generally happens when a non computerized system is used. All these modules are able to help librarian to manage the library with more convenience and in a more efficient way as compared to library systems which are not computerized.

IMPLEMENTATION

INDEX.JS:-

```
const express = require('express');
const bodyParser = require('body-parser');
const api = require('./routes/api');

const app = express();
// create a specific port for execution
const PORT = 5000;

app.use(bodyParser.json());
app.use('/', api);

// Successful msg
app.listen(PORT, () => console.log(`App listening on port ${PORT}`));
console.log("Mongoose connection is established ")
```

API.JS:-

```
const router = require('express').Router();
const bookModel = require('../model/book_model');

// Displays all the book data
router.get('/books', async function (req, res) {
   const bookList = await bookModel.find();
   console.log(bookList);
   res.send(bookList);
});

// Displays the data of a specified book
router.get('/books/:id', async function (req, res) {
    const { id } = req.params;
    const book = await bookModel.findOne({isbn : id});
    if(!book) return res.send("Book Not Found");
    res.send(book);
});

// Insert new book data
router.post('/books', async function (req, res) {
    const title= req.body.title;
    const isbn = req.body.isbn;
    const author = req.body.author;
    const bookExist = await bookModel.findOne({isbn : isbn});

    if (bookExist) return res.send('Book already exist');
```

```
    var data = await bookModel.create({title,isbn,author});
    data.save();

    res.send("Book Uploaded");
});

// Update an existing book data
router.put('/books/:id', async function (req, res) {
    const { id } = req.params;
    const {
        title,
        author,
    } = req.body;

    const bookExist = await bookModel.findOne({isbn : id});
    if (!bookExist) return res.send('Book Do Not exist');


    const updateField = (val, prev) => !val ? prev : val;

    const updatedBook = {
        ...bookExist ,
        title: updateField(title, bookExist.title),
        author: updateField(author, bookExist.author),

    };

    await bookModel.updateOne({isbn: id},{$set :{title : updatedBook.title,
author: updatedBook.author}})

    res.status(200).send("Book Updated");
});

// Delete a specific book from the collection
router.delete('/books/:id', async function (req, res) {
    const { id } = req.params;

    const bookExist = await bookModel.findOne({isbn : id});
    if (!bookExist) return res.send('Book Do Not exist');

   await bookModel.deleteOne({ isbn: id }).then(function(){
        console.log("Data deleted"); // Success
        res.send("Book Record Deleted Successfully")
    }).catch(function(error){
        console.log(error); // Failure
    });
});
```

```
// Delete a all book from the collection
router.delete('/books', async function (req, res) {

    const bookExist = await bookModel.find();
    if (!bookExist) return res.send('Book Do Not exist');

   await bookModel.deleteMany().then(function(){
        console.log("All Data deleted"); // Success
        res.send("All Book Record Deleted Successfully")
    }).catch(function(error){
        console.log(error); // Failure
    });
});

module.exports = router;
```

BOOKMODEL.JS:-

```
const mongoose = require('mongoose');
const db = require('../config/db');

// Book info to be stored with columns
const bookSchema = new mongoose.Schema({
    title:{
        type:String,
        default:"----"
    },
    isbn:{
      type:Number,
    },
    author:{
        type:String,
        default:"----"
    }
});
//creating models
const bookmodel = db.model('books',bookSchema);

module.exports = bookmodel;
```

DB.JS:-

```
//required mongoose
const mongoose = require('mongoose');

// creates a specified url to store book details
var url ='mongodb://localhost:27017/booksDB';

//establishing connections
const connection = mongoose.createConnection(url);

module.exports = connection;
```

PACKAGE.JSON :

```
{
  "name": "package.json",
  "version": "1.0.0",
  "description": "Json file",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "dev": "nodemon index.js"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "body-parser": "^1.20.2",
    "express": "^4.18.2",
    "mongoose": "^7.2.2",
    "nodemon": "^2.0.22"
  }
}
```

TOOLS AND TECHNOLOGY

Requirements:

1. **Hardware Technology:**
   Monitor/Screen Display: TFT Screen(1366*768)
   Processor: I3 or Higher Processors
   HDD: Minimum of 5GB
   RAM: 4GB

2. **Software Technology:**
   Operating System: Windows
   IDE: Visual Studio Code,
   Postman Application,
   Google Chrome

3. **Languages:** HTML5, CSS, JavaScript, NodeJS

4. **Database:** MongoDB

CONCLUSION

This website provides a computerized version of library management system which will benefit the students as well as the staff of the library. It makes entire process online where student can search books, staff can generate reports and do book transactions. It also has a facility for student login where student can login and can see status of books issued as well request for book or give some suggestions. It has a facility of teacher's login where teachers can add lectures notes and also give necessary suggestion to library and also add info about workshops or events happening in our college or nearby college in the online notice board.

FUTURE SCOPE

There is a future scope of this facility that many more features such as online lectures video tutorials can be added by teachers as well as online assignments submission facility , a feature Of group chat where students can discuss various issues of engineering can be added to this project thus making it more interactive more user friendly and project which fulfills each users need in the best way possible.

BIBLIOGRAPHY

1. https://www.w3schools.com/
2. https://fontawesome.com/
3. https://www.mongodb.com/try/download/community
4. https://nodejs.org/en/download