Admin-app/app.js:

const express = require('express');

const bodyParser = require('body-parser');

const sqlite3 = require('sqlite3').verbose();

const multer = require('multer');

const path = require('path');

const cors = require('cors');

const fs = require('fs');

const app = express();

const PORT = 3001;

app.use(cors());

app.use(bodyParser.json());

app.use(express.static('public'));

app.use('/pdfs', express.static(path.join(\_\_dirname, 'pdfs')));

app.use('/covers', express.static(path.join(\_\_dirname, 'covers')));

// Ensure pdfs and covers directories exist

const pdfDir = path.join(\_\_dirname, 'pdfs');

const coverDir = path.join(\_\_dirname, 'covers');

if (!fs.existsSync(pdfDir)) {

fs.mkdirSync(pdfDir);

}

if (!fs.existsSync(coverDir)) {

fs.mkdirSync(coverDir);

}

// Storage setup for multer

const storage = multer.diskStorage({

destination: (req, file, cb) => {

if (file.fieldname === 'pdf') {

cb(null, 'pdfs');

} else if (file.fieldname === 'cover') {

cb(null, 'covers');

}

},

filename: (req, file, cb) => {

cb(null, `${Date.now()}-${file.originalname}`);

}

});

const upload = multer({ storage });

// Database setup

const db = new sqlite3.Database(':memory:');

db.serialize(() => {

db.run("CREATE TABLE books (id INTEGER PRIMARY KEY, title TEXT, author TEXT, pdf TEXT, cover TEXT)");

});

// Routes

app.get('/books', (req, res) => {

db.all("SELECT \* FROM books", (err, rows) => {

if (err) {

console.error('Error fetching books:', err.message);

res.status(500).send(err.message);

return;

}

res.json(rows);

});

});

app.post('/books', upload.fields([{ name: 'pdf', maxCount: 1 }, { name: 'cover', maxCount: 1 }]), (req, res) => {

const { title, author } = req.body;

const pdf = req.files['pdf'][0].filename;

const cover = req.files['cover'][0].filename;

if (!title || !author || !pdf || !cover) {

res.status(400).send('Title, Author, PDF, and Cover are required');

return;

}

const stmt = db.prepare("INSERT INTO books (title, author, pdf, cover) VALUES (?, ?, ?, ?)");

stmt.run(title, author, pdf, cover, function(err) {

if (err) {

console.error('Error inserting book:', err.message);

res.status(500).send(err.message);

return;

}

res.status(201).send({ id: this.lastID });

});

stmt.finalize();

});

app.listen(PORT, () => {

console.log(`Admin server is running on http://localhost:${PORT}`);

});

Admin-app/public/index.html:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Library Admin</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<div class="container">

<h1>Library Admin</h1>

<div class="form-container">

<input type="text" id="title" placeholder="Book Title">

<input type="text" id="author" placeholder="Author">

<input type="file" id="pdf">

<input type="file" id="cover" accept="image/\*">

<button onclick="addBook()">Add Book</button>

</div>

<div class="book-list">

<h2>Book List</h2>

<ul id="books"></ul>

</div>

</div>

<script src="script.js"></script>

</body>

</html>

Admin-app/public/Script.js:

async function addBook() {

const title = document.getElementById('title').value;

const author = document.getElementById('author').value;

const pdf = document.getElementById('pdf').files[0];

const cover = document.getElementById('cover').files[0];

if (!title || !author || !pdf || !cover) {

alert('Please enter title, author, select a PDF, and upload a cover image');

return;

}

const formData = new FormData();

formData.append('title', title);

formData.append('author', author);

formData.append('pdf', pdf);

formData.append('cover', cover);

try {

const response = await fetch('http://localhost:3001/books', {

method: 'POST',

body: formData

});

if (response.ok) {

alert('Book added successfully');

loadBooks();

} else {

const errorText = await response.text();

alert('Error adding book: ' + errorText);

}

} catch (error) {

alert('Error adding book: ' + error.message);

}

}

async function loadBooks() {

const response = await fetch('http://localhost:3001/books');

const books = await response.json();

const booksList = document.getElementById('books');

booksList.innerHTML = '';

books.forEach(book => {

const li = document.createElement('li');

li.innerHTML = `

${book.title} by ${book.author}

<a href="http://localhost:3001/pdfs/${book.pdf}" target="\_blank">Read</a>

<img src="http://localhost:3001/covers/${book.cover}" alt="Book Cover" style="max-width: 100px;">

`;

booksList.appendChild(li);

});

}

document.addEventListener('DOMContentLoaded', loadBooks);

admin-app/public/styles.css:

body {

font-family: Arial, sans-serif;

background-color: #f0f0f0;

margin: 0;

padding: 0;

}

.container {

max-width: 800px;

margin: 50px auto;

background-color: #fff;

padding: 20px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

border-radius: 8px;

}

h1 {

text-align: center;

color: #333;

}

.form-container {

display: flex;

flex-direction: column;

gap: 10px;

}

input[type="text"], input[type="file"] {

padding: 10px;

border: 1px solid #ccc;

border-radius: 4px;

font-size: 16px;

}

button {

padding: 10px 15px;

border: none;

background-color: #28a745;

color: #fff;

font-size: 16px;

cursor: pointer;

border-radius: 4px;

transition: background-color 0.3s ease;

}

button:hover {

background-color: #218838;

}

.book-list {

margin-top: 20px;

}

.book-list h2 {

color: #333;

}

.book-list ul {

list-style: none;

padding: 0;

}

.book-list li {

padding: 10px;

border-bottom: 1px solid #ccc;

display: flex;

justify-content: space-between;

align-items: center;

}

.book-list a {

text-decoration: none;

color: #007bff;

transition: color 0.3s ease;

}

.book-list a:hover {

color: #0056b3;

}

Admin-app/Dockerfile:

FROM node

RUN mkdir /app

COPY . /app

WORKDIR /app

RUN rm -rf node\_modules

RUN npm install

CMD ["node", "app.js"]

User-app/app.js:

const express = require('express');

const cors = require('cors');

const app = express();

const PORT = 3000;

app.use(cors());

app.use(express.static('public'));

app.listen(PORT, () => {

console.log(`User server is running on http://localhost:${PORT}`);

});

User-app/public/index.html:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Library</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

<link rel="stylesheet" href="styles.css">

</head>

<body>

<div class="container">

<h1 class="my-4">Library</h1>

<div class="row" id="books">

<!-- Books will be loaded here -->

</div>

</div>

<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>

<script src="script.js"></script>

</body>

</html>

User-app/public/styles.css:

body {

font-family: 'Arial', sans-serif;

background-color: #f8f9fa;

}

h1 {

color: #007bff;

}

.list-group-item {

cursor: pointer;

}

.list-group-item:hover {

background-color: #007bff;

color: white;

}

.book-card {

border: 1px solid #dee2e6;

border-radius: 5px;

margin-bottom: 20px;

padding: 10px;

background-color: white;

}

.book-card img {

max-width: 100%;

height: auto;

}

.book-title {

font-size: 1.2em;

color: #343a40;

}

.book-author {

color: #6c757d;

}

.book-link {

margin-top: 10px;

}

User-app/public/script.js:

async function loadBooks() {

try {

const response = await fetch('http://localhost:3001/books');

const books = await response.json();

const booksList = document.getElementById('books');

booksList.innerHTML = '';

books.forEach(book => {

const col = document.createElement('div');

col.className = 'col-md-4';

const card = document.createElement('div');

card.className = 'book-card';

// Display the cover image

const img = document.createElement('img');

img.src = `http://localhost:3001/covers/${book.cover}`;

img.alt = 'Book Cover';

img.style.maxWidth = '100%';

card.appendChild(img);

const title = document.createElement('div');

title.className = 'book-title';

title.innerText = book.title;

card.appendChild(title);

const author = document.createElement('div');

author.className = 'book-author';

author.innerText = book.author;

card.appendChild(author);

const link = document.createElement('a');

link.className = 'btn btn-primary book-link';

link.href = `http://localhost:3001/pdfs/${book.pdf}`;

link.target = '\_blank';

link.innerText = 'Read';

card.appendChild(link);

col.appendChild(card);

booksList.appendChild(col);

});

} catch (error) {

console.error('Error loading books:', error.message);

// Handle error display or logging

}

}

document.addEventListener('DOMContentLoaded', loadBooks);

userfile/Dockerfile:

FROM node

RUN mkdir /app

COPY . /app

WORKDIR /app

RUN npm install

CMD ["node", "app.js"]