// players collection schema

{

\_id: ObjectId,

name: String,

age: Number,

team: String,

role: String // batsman, bowler, all-rounder, etc.

}

// teams collection schema

{

\_id: ObjectId,

name: String,

country: String,

coach: String

}

// matches collection schema

{

\_id: ObjectId,

team1: String,

team2: String,

date: Date,

venue: String

}

// performance collection schema

{

\_id: ObjectId,

playerId: ObjectId,

matchId: ObjectId,

runs: Number,

wickets: Number,

catches: Number

}

const express = require('express');

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

const MongoClient = require('mongodb').MongoClient;

const app = express();

const PORT = 3000;

app.use(bodyParser.json());

const mongoUrl = 'mongodb://localhost:27017';

const dbName = 'cricketDB';

MongoClient.connect(mongoUrl, { useNewUrlParser: true, useUnifiedTopology: true }, (err, client) => {

if (err) throw err;

const db = client.db(dbName);

// GET endpoint to retrieve all players

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

const players = await db.collection('players').find({}).toArray();

res.json(players);

});

// GET endpoint to retrieve all teams

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

const teams = await db.collection('teams').find({}).toArray();

res.json(teams);

});

// POST endpoint to add a new player

app.post('/players', async (req, res) => {

const newPlayer = req.body;

const result = await db.collection('players').insertOne(newPlayer);

res.json(result);

});

// POST endpoint to add a new team

app.post('/teams', async (req, res) => {

const newTeam = req.body;

const result = await db.collection('teams').insertOne(newTeam);

res.json(result);

});

// DELETE endpoint to delete a player by ID

app.delete('/players/:id', async (req, res) => {

const playerId = req.params.id;

const result = await db.collection('players').deleteOne({ \_id: ObjectId(playerId) });

res.json(result);

});

// PUT endpoint to update player information

app.put('/players/:id', async (req, res) => {

const playerId = req.params.id;

const updatedPlayer = req.body;

const result = await db.collection('players').updateOne({ \_id: ObjectId(playerId) }, { $set: updatedPlayer });

res.json(result);

});

app.listen(PORT, () => {

console.log(`Server is running on port ${PORT}`);

});

});

<!-- app.component.html -->

<h1>Cricket Data Management</h1>

<hr>

<div>

<h2>Players</h2>

<ul>

<li \*ngFor="let player of players">

{{ player.name }} ({{ player.team }})

</li>

</ul>

</div>

<div>

<h2>Teams</h2>

<ul>

<li \*ngFor="let team of teams">

{{ team.name }} ({{ team.country }})

</li>

</ul>

</div>

// app.component.ts

import { Component, OnInit } from '@angular/core';

import { HttpClient } from '@angular/common/http';

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent implements OnInit {

players: any[] = [];

teams: any[] = [];

constructor(private http: HttpClient) {}

ngOnInit() {

this.getPlayers();

this.getTeams();

}

getPlayers() {

this.http.get<any[]>('http://localhost:3000/players').subscribe(data => {

this.players = data;

});

}

getTeams() {

this.http.get<any[]>('http://localhost:3000/teams').subscribe(data => {

this.teams = data;

});

}

}

This Angular component will display a list of players and teams fetched from the Node.js API endpoints we defined earlier. You'll need to set up Angular CLI and include HttpClientModule in your Angular app to make HTTP requests.