Experiment 11

**Title**

REST API for Playing Card Collection Using Express.js

**Objective**

Build a RESTful API using Express.js to manage a collection of playing cards. This task helps you understand routing, handling HTTP methods, and basic data manipulation in a Node.js backend environment.

**Task Description**

Create an Express.js server that provides API endpoints to manage a playing card collection. The API should allow you to perform operations such as listing all cards, adding a new card (with properties like suit and value), retrieving a specific card by ID, and deleting a card by ID. Store card data in an in-memory array for simplicity. The API should follow RESTful principles and handle different HTTP methods (GET, POST, DELETE) clearly, responding with appropriate JSON data.

Code

const express = require("express");

const app = express();

const PORT = 3000;

// Middleware to parse JSON requests

app.use(express.json());

// In-memory array to store playing cards

let cards = [];

let idCounter = 1;

/\*\*

 \* GET /cards

 \* List all cards

 \*/

app.get("/cards", (req, res) => {

  res.json(cards);

});

/\*\*

 \* POST /cards

 \* Add a new card (expects {suit, value})

 \*/

app.post("/cards", (req, res) => {

  const { suit, value } = req.body;

  if (!suit || !value) {

    return res.status(400).json({ error: "Suit and value are required" });

  }

  const newCard = { id: idCounter++, suit, value };

  cards.push(newCard);

  res.status(201).json(newCard);

});

/\*\*

 \* GET /cards/:id

 \* Retrieve a specific card by ID

 \*/

app.get("/cards/:id", (req, res) => {

  const id = parseInt(req.params.id);

  const card = cards.find((c) => c.id === id);

  if (!card) {

    return res.status(404).json({ error: "Card not found" });

  }

  res.json(card);

});

/\*\*

 \* DELETE /cards/:id

 \* Delete a card by ID

 \*/

app.delete("/cards/:id", (req, res) => {

  const id = parseInt(req.params.id);

  const index = cards.findIndex((c) => c.id === id);

  if (index === -1) {

    return res.status(404).json({ error: "Card not found" });

  }

  const deletedCard = cards.splice(index, 1);

  res.json({ message: "Card deleted", card: deletedCard[0] });

});

// Start the server

app.listen(PORT, () => {

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

});

Output

