```
// server.js
import express from "express";
import bodyParser from "body-parser";
import cors from "cors";
import OpenAI from "openai";

const app = express();
app.use(cors());
app.use(bodyParser.json());

const openai = new OpenAI({
  apiKey: process.env.OPENAI_API_KEY,
});

// In-memory stores
const agents = new Map();     // id -> agent definition
const tasks = new Map();      // id -> task
const sessions = new Map();   // id -> session context

// Simple ID generator
const makeId = (prefix = "id") =>
  `${prefix}_${Math.random().toString(36).slice(2, 10)}`;

// ----------------------
// SESSION ROUTES
// ----------------------

app.post("/sessions", (req, res) => {
  const { name, brand, context = {}, owner_id } = req.body || {};
  if (!name) {
    return res.status(400).json({ error: "name is required" });
  }
  const id = makeId("session");
  const session = {
    id,
    name,
    brand: brand || "",
    context,
    owner_id: owner_id || null,
    created_at: new Date().toISOString(),
  };
  sessions.set(id, session);
  res.json(session);
});
```

```
app.get("/sessions/:id", (req, res) => {
  const session = sessions.get(req.params.id);
  if (!session) {
    return res.status(404).json({ error: "Session not found" });
  }
  res.json(session);
});

app.patch("/sessions/:id/context", (req, res) => {
  const session = sessions.get(req.params.id);
  if (!session) {
    return res.status(404).json({ error: "Session not found" });
  }

  const { context = {} } = req.body || {};
  session.context = {
    ...(session.context || {}),
    ...context,
  };

  sessions.set(session.id, session);
  res.json(session);
});

// ----------------------
// AGENT ROUTES
// ----------------------

app.get("/team/agents", (req, res) => {
  res.json({ agents: Array.from(agents.values()) });
});

app.post("/team/agents", (req, res) => {
  const {
    id,
    name,
    role,
    specialization,
    system_prompt,
    metadata = {},
  } = req.body || {};

  if (!name || !role || !specialization) {
    return res.status(400).json({
      error: "name, role, and specialization are required",
    });
  }

  const agentId = id || makeId("agent");

  const agent = {
    id: agentId,
    name,
    role,
    specialization,
    system_prompt: system_prompt || "",
    metadata,
  };

  agents.set(agentId, agent);
  res.json(agent);
});

app.get("/team/agents/:agent_id", (req, res) => {
  const agent = agents.get(req.params.agent_id);
  if (!agent) {
    return res.status(404).json({ error: "Agent not found" });
  }
  res.json(agent);
});

// ----------------------
// TASKS (including sync agent execution)
// ----------------------

app.get("/team/tasks", (req, res) => {
  const { status } = req.query;
  let allTasks = Array.from(tasks.values());
  if (status) {
    allTasks = allTasks.filter((t) => t.status === status);
  }
  res.json({ tasks: allTasks });
});

app.get("/team/tasks/:task_id", (req, res) => {
  const task = tasks.get(req.params.task_id);
  if
```