```
require("dotenv").config();

const express = require("express");
const cors = require("cors");
const multer = require("multer");
const { spawn } = require("child_process");
const path = require("path");
const fs = require("fs");

const app = express();
const PORT = process.env.PORT || 5000;
const uploadDir = path.join(__dirname, process.env.UPLOAD_DIR || "uploads");

// Ensure uploads folder exists
if (!fs.existsSync(uploadDir)) {
  fs.mkdirSync(uploadDir, { recursive: true });
}

// CORS Configuration
app.use(cors({
  origin: "*",
  methods: "GET, POST",
  allowedHeaders: "Content-Type, Authorization"
}));

// Multer storage configuration
const storage = multer.diskStorage({
  destination: (req, file, cb) => {
    cb(null, uploadDir);
  },
  filename: (req, file, cb) => {
    const fileExtension = path.extname(file.originalname);
    const timestamp = Date.now();
    cb(null, `${timestamp}${fileExtension}`);
  },
});

const upload = multer({ storage });

// File upload and scan route
app.post("/uploads", upload.single("file"), (req, res) => {
  if (!req.file) {
    return res.status(400).json({ error: "No file uploaded" });
  }

  const filePath = path.join(uploadDir, req.file.filename);

  if (!fs.existsSync(filePath)) {
    return res.status(500).json({ error: "Uploaded file not found" });
  }

  const pythonPath = process.env.PYTHON_PATH || "python";
  const pythonScriptPath = path.join(__dirname, "malware_scan.py");

  console.log(`Running: ${pythonPath} ${pythonScriptPath} ${filePath}`);

  const pythonProcess = spawn(pythonPath, [pythonScriptPath, filePath]);

  let output = "";
  let errorOutput = "";

  pythonProcess.stdout.on("data", (data) => {
```

```
  output += data.toString();
});

pythonProcess.stderr.on("data", (data) => {
  errorOutput += data.toString();
});

pythonProcess.on("close", (code) => {
  console.log("Python stdout:", output);
  console.log("Python stderr:", errorOutput);

  if (code !== 0 || errorOutput.trim().length > 0) {
    return res.status(500).json({ error: "Error running malware scan", details: errorOutput });
  }

  // Ensure output is valid
  const lines = output.trim().split("\n");
  if (lines.length < 3) {
    return res.status(500).json({ error: "Unexpected Python script output", details: output });
  }

  // ✅ Extract the actual risk score using regex
  let riskScoreLine = lines.find(line => line.toLowerCase().includes("final risk score"));
  let riskScore = "Unknown";
  if (riskScoreLine) {
  const match = riskScoreLine.match(/(\d+(\.\d+)?)/); // Capture decimal numbers like 4.5
    if (match) {
      riskScore = `${match[1]}/10`; // Converts "4.5" to "4.5/10"
      // Extracted "8" or similar format
    }
  }
```

```
    // ✅ Get the last line as the result (Clean, Infected, etc.)
    let result = lines[lines.length - 1]?.trim().toLowerCase();

    // Standardize result output
    if (result.includes("infected")) {
      result = "Infected";
    } else if (result.includes("clean")) {
      result = "Clean";
    } else if (result.includes("suspicious")) {
      result = "Suspicious";
    } else {
      result = "Unknown";
    }

    res.json({
      message: result,
      riskScore: riskScore, // Correctly extracted "2/10"
    });
  });
});

// Start the server
app.listen(PORT, () => {
  console.log(`Server running on http://localhost:${PORT}`);
});
```