// server.js

const express = require("express");

const mongoose = require("mongoose");

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

const cors = require("cors");

// --- Setup ---

const app = express();

app.use(bodyParser.json());

app.use(cors());

// --- MongoDB Connection ---

mongoose.connect("mongodb://127.0.0.1:27017/seema888", {

  useNewUrlParser: true,

  useUnifiedTopology: true,

}).then(() => console.log("✅ MongoDB connected"))

  .catch(err => console.error("❌ MongoDB error:", err));

// --- Schema ---

const userSchema = new mongoose.Schema({

  role: { type: String, required: true, enum: ["student", "clubhead", "faculty"] },

  studentUsername: String,

  studentPassword: String,

  clubUsername: String,

  clubPassword: String,

  facultyName: String,

  facultyEmail: String,

  facultyPassword: String,

  createdAt: { type: Date, default: Date.now }

});

const User = mongoose.model("User", userSchema);

// --- Validation Regex ---

const regex = {

  student: /^(22|23|24|25)BD1A05[A-G][0-9]$/,

  clubUser: /^CLUB-[A-Z]{1,10}$/,

  clubPass: /^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*\d)(?=.\*[!@#$%^&\*]).{8,}$/,

  facultyName: /^[A-Za-z]{1,20}$/,

  facultyEmail: /^[A-Za-z]{10}[0-9]{0,3}@gmail\.com$/,

  facultyPass: /^(?=^[A-Z])(?=(?:.\*[A-Za-z]){8,})(?=.\*\d)(?=.\*[!@#$%^&\*])[A-Za-z\d!@#$%^&\*]{10,}$/

};

// --- API Endpoint ---

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

  try {

    const { role } = req.body;

    if (role === "student") {

      const { studentUsername, studentPassword } = req.body;

      if (!regex.student.test(studentUsername)) return res.status(400).json({ error: "Invalid Student Roll No." });

      if (studentUsername !== studentPassword) return res.status(400).json({ error: "Student password must equal Roll No." });

    }

    if (role === "clubhead") {

      const { clubUsername, clubPassword } = req.body;

      if (!regex.clubUser.test(clubUsername)) return res.status(400).json({ error: "Invalid Club Username." });

      if (!regex.clubPass.test(clubPassword)) return res.status(400).json({ error: "Weak Club Password." });

    }

    if (role === "faculty") {

      const { facultyName, facultyEmail, facultyPassword } = req.body;

      if (!regex.facultyName.test(facultyName)) return res.status(400).json({ error: "Invalid Faculty Name." });

      if (!regex.facultyEmail.test(facultyEmail)) return res.status(400).json({ error: "Invalid Faculty Email." });

      if (!regex.facultyPass.test(facultyPassword)) return res.status(400).json({ error: "Weak Faculty Password." });

    }

    const newUser = new User(req.body);

    await newUser.save();

    res.json({ message: "✅ Registration Successful!" });

  } catch (err) {

    console.error(err);

    res.status(500).json({ error: "Server error" });

  }

});

// --- Start Server ---

const PORT = 5000;

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