const express = require('express');

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

const jwt = require('jsonwebtoken');

const bcrypt = require('bcrypt');

const mongoose = require('mongoose');

const app = express();

const port = process.env.PORT || 3000;

mongoose.connect('mongodb://localhost/myapp', { useNewUrlParser: true, useUnifiedTopology: true });

const userSchema = new mongoose.Schema({ name: String,email: String, password: String});

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

app.use(bodyParser.json());

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

const { name, email, password } = req.body;

const existingUser = await User.findOne({ email });

if (existingUser) { return res.status(409).json({ message: 'Email already exists' });

}

const hashedPassword = await bcrypt.hash(password, 10);

const user = new User({name,email, password: hashedPassword });

await user.save();

const token = jwt.sign({ userId: user.\_id }, 'secret\_key', { expiresIn: '1h' });

res.json({ token });

});

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

const { email, password } = req.body;

const user = await User.findOne({ email });

if (!user) {

return res.status(401).json({ message: 'Invalid credentials' });

}

const isPasswordValid = await bcrypt.compare(password, user.password);

if (!isPasswordValid) {

return res.status(401).json({ message: 'Invalid credentials' });

}

const token = jwt.sign({ userId: user.\_id }, 'secret\_key', { expiresIn: '1h' });

res.json({ token });

});

app.get('/api/check-auth', (req, res) => {

const authHeader = req.headers.authorization;

if (!authHeader) {

return res.status(401).json({ message: 'No token provided' });

}

const token = authHeader.split(' ')[1];

try {

const decodedToken = jwt.verify(token, 'secret\_key');

res.sendStatus(200);

} catch {

res.status(401).json({ message: 'Invalid token' });

}

});

app.listen(port, () => console.log(`Server listening on port ${port}`));