const express = require('express');

const cors = require('cors');

const { MongoClient, ObjectId } = require('mongodb');

const app = express(); // ✅ Only define app ONCE

const port = 3000;

app.use(cors());

app.use(express.json());

let db;

async function connectToMongoDB() {

console.log("Script is running...");

const uri = "mongodb://localhost:27017";

const client = new MongoClient(uri);

try {

console.log("Attempting to connect to MongoDB...");

await client.connect();

console.log("Connected to MongoDB!");

db = client.db("testDB");

} catch (err) {

console.error("Error:", err);

}

}

connectToMongoDB();

app.listen(port, () => {

console.log(`Server running on port ${port}`);

});

//---------------------------RIDES (RMU) COLLECTION---------------------------------------

// GET /rides - Fetch All Rides

app.get('/rides', async (req, res) => {

try {

const rides =await db.collection('rides').find().toArray();

res.status(200).json(rides);

} catch (err) {

res.status(500).json({ error: "Failed to fetch rides" });

}

});

// DELETE /rides/:id - Cancel A Ride

app.delete('/rides/:id', async (req, res) => {

try {

const result = await db.collection('rides').deleteOne(

{ \_id: new ObjectId(req.params.id) }

);

if (result.deletedCount === 0) {

return res.status(404).json({ error: "Ride Not Found" });

}

res.status(200).json({ deleted: result.deletedCount });

} catch (err) {

res.status(400).json({ error: "Invalid Ride Id Or Data" });

}

});

//------------------------ADMIN COLLECTION---------------------------------

// DELETE /admins/:id - Block users

app.delete('/admin/:id', async (req, res) => {

try {

const result = await db.collection('users').deleteOne(

{ \_id: new ObjectId(req.params.id) }

);

if (result.deletedCount === 0) {

return res.status(404).json({ error: "RideNot Found" });

}

res.status(200).json({ deleted: result.deletedCount });

} catch (err) {

res.status(400).json({ error: "Invalid Ride Or Data" });

}

});

// GET /admin - Show All Users Info

app.get('/admin/users', async (req, res) => {

try {

const rides = await db.collection('users').find().toArray();

res.status(200).json(rides);

} catch (err) {

res.status(500).json({ error: "Failed to fetch rides" });

}

});

// GET /admin - Show All Drivers Info

app.get('/admin/drivers', async (req, res) => {

try {

const rides = await db.collection('drivers').find().toArray();

res.status(200).json(rides);

} catch (err) {

res.status(500).json({ error: "Failed to fetch rides" });

}

});

//--------------------------DRIVER COLLECTION---------------------------------

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

try {

const result = await db.collection('drivers').insertOne(req.body);

res.status(201).json({ id: result.insertedId });

} catch (err) {

res.status(400).json({ error: "Invalid Ride Data" });

}

});

// PATCH /drivers/:id - Update driver's availability status

app.patch('/drivers/:id', async (req, res) => {

try {

const result = await db.collection('rides').updateOne(

{ \_id: new ObjectId(req.params.id) },

{ $set: { status: req.body.status } }

);

if (result.matchedCount === 0) {

return res.status(404).json({ error: "Ride Not Found" });

}

res.status(200).json({ updated: result.modifiedCount });

} catch (err) {

res.status(500).json({ error: "Invalid Ride Id Or Data" });

}

});

app.post('/drivers/acceptBook', async (req, res) => {

try {

const result = await db.collection('rides').insertOne(req.body);

res.status(201).json({ id: result.insertedId });

} catch (err) {

res.status(400).json({ error: "Invalid Ride Data" });

}

});

//--------------------------USER COLLECTION-------------------------

//POST /users - Book A Ride

app.post('/users/book', async (req, res) => {

try {

const result = await db.collection('rides').insertOne(req.body);

res.status(201).json({ id: result.insertedId });

} catch (err) {

res.status(400).json({ error: "Invalid Ride Data" });

}

});

//POST /users - Register A Passenger

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

try {

const result = await db.collection('users').insertOne(req.body);

res.status(201).json({ id: result.insertedId });

} catch (err) {

res.status(400).json({ error: "Invalid Ride Data" });

}

});

//POST /users - Login a passenger

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

const { email, password } = req.body;

try {

if (!email || !password) {

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

}

const user = await db.collection('users').findOne({ email, password });

if (!user) {

return res.status(401).json({ error: "Invalid email or password" });

}

// Login successful, return user profile (excluding password)

const { \_id, Username } = user;

res.status(200).json({ \_id, Username, email });

} catch (err) {

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

    }

});

// PATCH /users/ :id - Give driver's rating

app.patch('/users/:id', async (req, res) => {

try {

const result = await db.collection('drivers').updateOne(

{ \_id: new ObjectId(req.params.id) },

{ $set: { rating: req.body.rating } }

);

if (result.matchedCount === 0) {

return res.status(404).json({ error: "Ride Not Found" });

}

res.status(200).json({ updated: result.modifiedCount });

} catch (err) {

res.status(500).json({ error: "Invalid Ride Id Or Data" });

}

});