**Module 66-Genius Car Node Mongo CRUD Recap**

66-1 (conceptual) Node, Mongo, React, client, server, Rest, CRUD

**What is REST API?**

1) RESTful

2) API

An API is an interface through which one program or web site talks to another.

They are used to share data and services, and they come in many different

formats and types.

A RESTful API is one of the many possible ways that programs, servers,

and web sites can share data and services. REST (Representational State Transfer

) describes the general rules for how the data and services are represented

through the API so that other programs will be able to correctly request and

receive the data and services that an API makes available.

**What is express js?**

Express.js, or simply Express, is a back end web application framework for Node.js, released as free and open-source software under the MIT License. It is designed for building web applications and APIs. It has been called the de facto standard server framework for Node.js.

#### 66-2 Setup simple node express server with dotenv

const express = require("express");

const cors=require("cors");

require('dotenv').config();

const port=process.env.PORT||5000;

const app = express();

// middleware

app.use(cors());

app.use(express.json());

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

    res.send("Running Genius Server");

});

app.listen(port,()=>{

    console.log("Listening Port",port);

});

#### 66-3 Connect to database with secure password on environment variable

#### 66-4 Load all services and create single service API

async function run() {

  try {

    await client.connect();

    const serviceCollection = client.db("geniusCar").collection("service");

    //load data multiple

    app.get("/service", async (req, res) => {

      const query = {};

      const cursor = serviceCollection.find(query);

      const service = await cursor.toArray();

      res.send(service);

    });

    //load data single

    app.get("/service/:id", async (req, res) => {

      const id = req.params.id;

      const query = { \_id: ObjectId(id) };

      const service = await serviceCollection.findOne(query);

      res.send(service);

    });

  } finally {

  }

}

#### 66-5 Load single Service, Use React hook form to add service

<https://react-hook-form.com/get-started#Quickstart>

#### 66-6 Insert Service data to the mongodb cloud database

    // POST

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

        const newService=req.body;

        const result=await serviceCollection.insertOne(newService);

        res.send(result);

    });

    const onSubmit = (data) => {

      console.log(data);

      const url = `http://localhost:5000/service`;

      fetch(url, {

        method: "POST",

        headers: {

          "content-type": "application/json",

        },

        body: JSON.stringify(data),

      })

        .then((res) => res.json())

        .then((result) => {

          console.log(result);

        });

    };

#### 66-7 Manage Services and explore delete api

  const handleDelete = (id) => {

    const proceed = window.confirm("Are you sure?");

    if (proceed) {

      const url = `http://localhost:5000/service/${id}`;

      console.log(url);

      fetch(url, {

        method: "DELETE",

      })

        .then((res) => res.json())

        .then((data) => {

        //   console.log(data);

          const remaining = services.filter((service) => service.\_id !== id);

          setServices(remaining);

        });

    }

  };

    // DELETE

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

        const id=req.params.id;

        const query = { \_id:ObjectId(id)};

        const result = await serviceCollection.deleteOne(query);

        res.send(result);

    });

#### 66-8 Module Summary and git push