API in node js - that will perform the crud operation for the below table. Database used is mysql should include config, model , repositories, services, controller, middleware, validation , routes folder structures. query should not be used in repositories.

**ChatGPT said:**

Here's a complete **Node.js RESTful API** structure using **MySQL** and **Sequelize ORM** (since raw SQL queries are not to be used in repositories). It includes:

✅ Config  
✅ Model  
✅ Repository  
✅ Service  
✅ Controller  
✅ Middleware  
✅ Validation  
✅ Routes  
✅ Proper Folder Structure

**✅ Sample Table**

sql

CopyEdit

CREATE TABLE employees (

employee\_id INT PRIMARY KEY AUTO\_INCREMENT,

first\_name VARCHAR(20),

last\_name VARCHAR(25) NOT NULL,

email VARCHAR(25) NOT NULL,

phone\_number VARCHAR(20),

hire\_date DATE NOT NULL,

job\_id VARCHAR(10),

salary DECIMAL(8,2)

);

**📁 Folder Structure**

lua

CopyEdit

project-root/

│

├── config/

│ └── db.config.js

│

├── models/

│ └── index.js

│ └── employee.model.js

│

├── repositories/

│ └── employee.repository.js

│

├── services/

│ └── employee.service.js

│

├── controllers/

│ └── employee.controller.js

│

├── middlewares/

│ └── error.middleware.js

│

├── validations/

│ └── employee.validation.js

│

├── routes/

│ └── employee.routes.js

│

├── app.js

└── server.js

**🔧 1. config/db.config.js**

js

CopyEdit

const { Sequelize } = require("sequelize");

const sequelize = new Sequelize("your\_database", "your\_user", "your\_password", {

host: "localhost",

dialect: "mysql",

logging: false

});

module.exports = sequelize;

**📦 2. models/index.js**

js

CopyEdit

const Sequelize = require("sequelize");

const sequelize = require("../config/db.config");

const db = {};

db.Sequelize = Sequelize;

db.sequelize = sequelize;

db.Employee = require("./employee.model")(sequelize, Sequelize);

module.exports = db;

**📦 3. models/employee.model.js**

js

CopyEdit

module.exports = (sequelize, DataTypes) => {

return sequelize.define("Employee", {

employee\_id: {

type: DataTypes.INTEGER,

primaryKey: true,

autoIncrement: true

},

first\_name: DataTypes.STRING,

last\_name: {

type: DataTypes.STRING,

allowNull: false

},

email: {

type: DataTypes.STRING,

allowNull: false

},

phone\_number: DataTypes.STRING,

hire\_date: {

type: DataTypes.DATE,

allowNull: false

},

job\_id: DataTypes.STRING,

salary: DataTypes.DECIMAL(8, 2)

}, {

tableName: "employees",

timestamps: false

});

};

**📚 4. repositories/employee.repository.js**

js

CopyEdit

const { Employee } = require("../models");

class EmployeeRepository {

async create(data) {

return await Employee.create(data);

}

async findAll() {

return await Employee.findAll();

}

async findById(id) {

return await Employee.findByPk(id);

}

async update(id, data) {

const emp = await Employee.findByPk(id);

if (!emp) return null;

return await emp.update(data);

}

async delete(id) {

const emp = await Employee.findByPk(id);

if (!emp) return null;

await emp.destroy();

return emp;

}

}

module.exports = new EmployeeRepository();

**🧠 5. services/employee.service.js**

js

CopyEdit

const employeeRepo = require("../repositories/employee.repository");

class EmployeeService {

async createEmployee(data) {

return await employeeRepo.create(data);

}

async getAllEmployees() {

return await employeeRepo.findAll();

}

async getEmployeeById(id) {

return await employeeRepo.findById(id);

}

async updateEmployee(id, data) {

return await employeeRepo.update(id, data);

}

async deleteEmployee(id) {

return await employeeRepo.delete(id);

}

}

module.exports = new EmployeeService();

**🎮 6. controllers/employee.controller.js**

js

CopyEdit

const employeeService = require("../services/employee.service");

exports.create = async (req, res, next) => {

try {

const emp = await employeeService.createEmployee(req.body);

res.status(201).json(emp);

} catch (err) {

next(err);

}

};

exports.getAll = async (req, res, next) => {

try {

const emps = await employeeService.getAllEmployees();

res.json(emps);

} catch (err) {

next(err);

}

};

exports.getById = async (req, res, next) => {

try {

const emp = await employeeService.getEmployeeById(req.params.id);

if (!emp) return res.status(404).json({ message: "Employee not found" });

res.json(emp);

} catch (err) {

next(err);

}

};

exports.update = async (req, res, next) => {

try {

const updated = await employeeService.updateEmployee(req.params.id, req.body);

if (!updated) return res.status(404).json({ message: "Employee not found" });

res.json(updated);

} catch (err) {

next(err);

}

};

exports.remove = async (req, res, next) => {

try {

const removed = await employeeService.deleteEmployee(req.params.id);

if (!removed) return res.status(404).json({ message: "Employee not found" });

res.json({ message: "Deleted successfully" });

} catch (err) {

next(err);

}

};

**🛡️ 7. middlewares/error.middleware.js**

js

CopyEdit

module.exports = (err, req, res, next) => {

console.error(err.stack);

res.status(500).json({ message: err.message });

};

**✅ 8. validations/employee.validation.js (optional with Joi)**

js

CopyEdit

const Joi = require("joi");

const employeeSchema = Joi.object({

first\_name: Joi.string().optional(),

last\_name: Joi.string().required(),

email: Joi.string().email().required(),

phone\_number: Joi.string().optional(),

hire\_date: Joi.date().required(),

job\_id: Joi.string().optional(),

salary: Joi.number().precision(2).optional()

});

module.exports = { employeeSchema };

**🚏 9. routes/employee.routes.js**

js

CopyEdit

const express = require("express");

const router = express.Router();

const employeeController = require("../controllers/employee.controller");

router.post("/", employeeController.create);

router.get("/", employeeController.getAll);

router.get("/:id", employeeController.getById);

router.put("/:id", employeeController.update);

router.delete("/:id", employeeController.remove);

module.exports = router;

**🚀 10. app.js**

js

CopyEdit

const express = require("express");

const app = express();

const sequelize = require("./config/db.config");

const employeeRoutes = require("./routes/employee.routes");

const errorHandler = require("./middlewares/error.middleware");

app.use(express.json());

app.use("/api/employees", employeeRoutes);

app.use(errorHandler);

sequelize.sync().then(() => {

console.log("Database synced.");

});

module.exports = app;

**🌐 11. server.js**

js

CopyEdit

const app = require("./app");

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

app.listen(PORT, () => {

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

});

**✅ To Run:**

bash

CopyEdit

npm install express sequelize mysql2 joi

node server.js