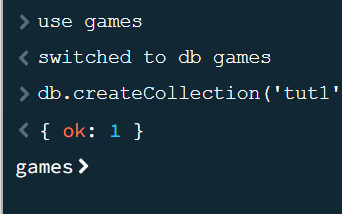
**MongoDB**

**Command:**

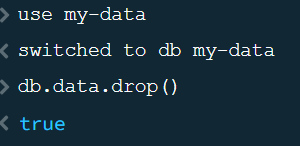
**show dbs or show databases**

**A databases atleast have 1 collection to use**

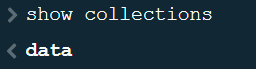
**Create Database**

****

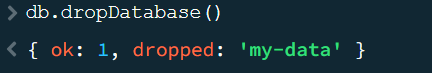
**Delete collection**

****

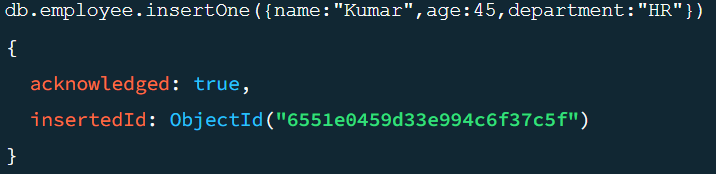
**Display collections**

****

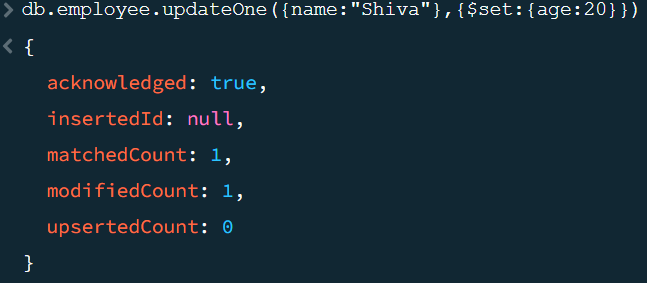
**Drop Database**

****

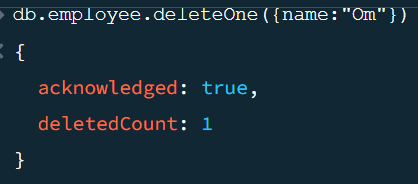
**Insert data**

****

**Update Data**

****

**Delete Data**

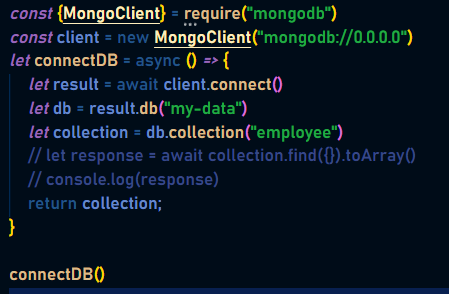
****

**Connect MongoDB with Node Js**

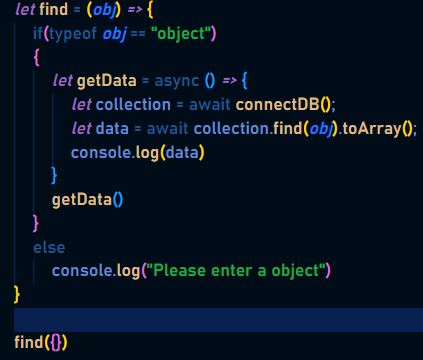
**Mongodb npm**

**npm i mongodb**

**Connect localhost,db and collection**

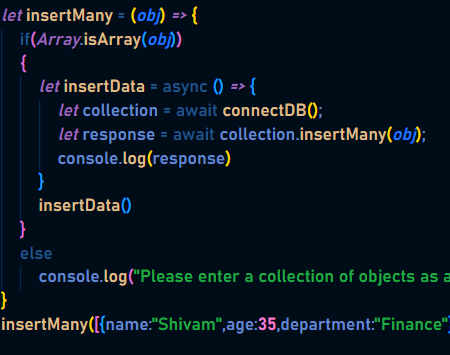
****

**Display data in console**

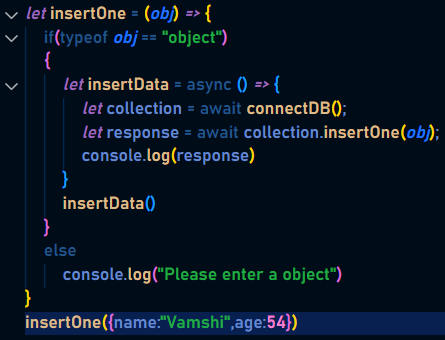
****

**Insert Data**

InsertMany



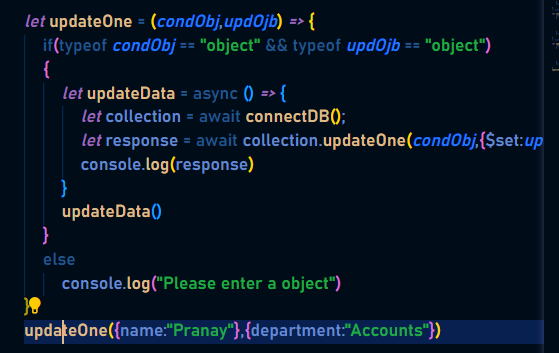
InsertOne

****

**Update Data**

UpdateMany

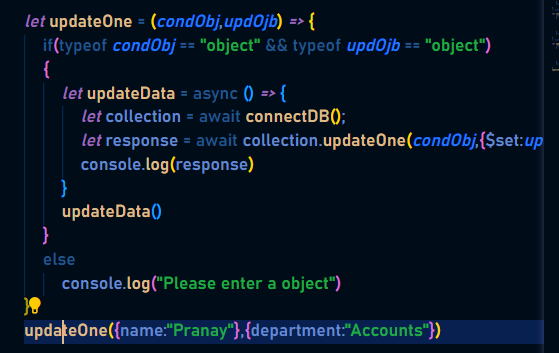
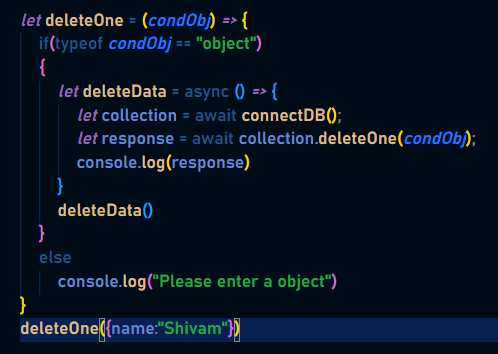
UpdateOne



**Delete Data**

deletMany

deleteOne



*module*.*exports* = {

    find: async (*obj*) *=>* {

        if (typeof *obj* == "object") {

*let* collection = await connectDB();

            return collection.find(*obj*).toArray();

        }

        else

            console.log("Please enter a object")

    },

    insertOne: async (*obj*) *=>* {

        if (typeof *obj* == "object") {

*let* collection = await connectDB();

            return collection.insertOne(*obj*);

        }

        else

            console.log("Please enter a object")

    },

    insertMany: async (*obj*) *=>* {

        if (*Array*.isArray(*obj*)) {

*let* collection = await connectDB();

            return collection.insertMany(*obj*);

        }

        else

            console.log("Please enter a collection of objects as a array")

    },

    updateOne:async (*condObj*, *updOjb*) *=>* {

        if (typeof *condObj* == "object" && typeof *updOjb* == "object") {

*let* collection = await connectDB();

                return collection.updateOne(*condObj*, { $set: *updOjb* });

        }

        else

            console.log("Please enter a object")

    },

    deleteOne:async (*condObj*) *=>* {

        if (typeof *condObj* == "object") {

*let* collection = await connectDB();

                return collection.deleteOne(*condObj*);

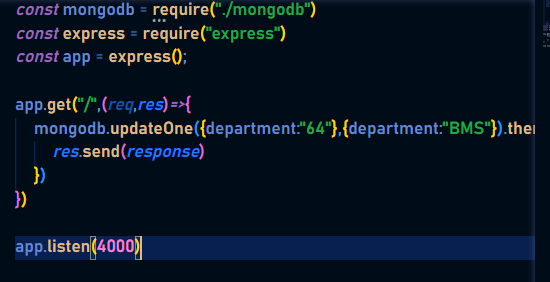
        }

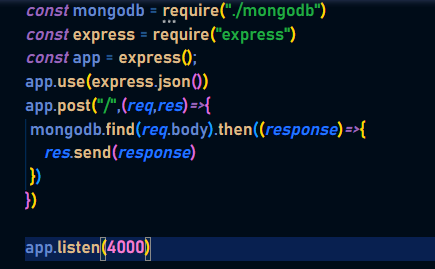
        else

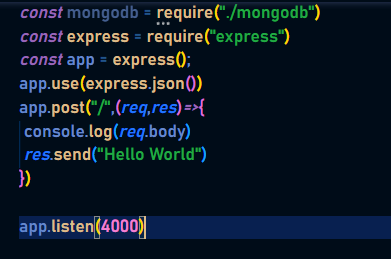
            console.log("Please enter a object")

    },

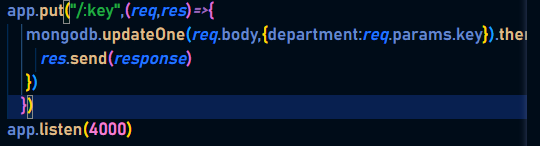
}



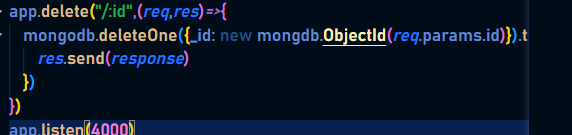
POST API



PUT API



DELETE API



Mongoose

Mongoose is the advanced means it provides some features

npm I mongoose

*const* mongoose = require("mongoose")

*const* main = async () *=>* {

    await mongoose.connect("mongodb://0.0.0.0/my-data")

*let* employeeScheme = new mongoose.Schema({

        name: *String*,

        department: String

    })

*let* employeeModel = mongoose.model('employee',employeeScheme)

*let* data = new employeeModel({name:"Bavesh",department:"HR"})

*let* result = await data.save();

    console.log(result)

}

main()

Mongoose CURD Operation

*const* exp = require("constants")

*const* mongoose = require("mongoose")

mongoose.connect("mongodb://0.0.0.0/my-data")

*let* employeeSchema =new mongoose.Schema({

    name:*String*,

    age: *Number*,

    department: *String*

})

*let* readdata = async () *=>* {

*let* employeeModel = mongoose.model("employee",employeeSchema)

*let* data = await employeeModel.find({})

    return data;

}

*let* saveData = async (*name*) *=>* {

*let* employeeModel = mongoose.model("employee",employeeSchema)

*let* response = await employeeModel.create({name:*name*})

    return response;

}

*let* deleteData = async (*id*) *=>* {

*let* employeeModel = mongoose.model("employee",employeeSchema)

*let* response = await employeeModel.deleteOne({\_id: *id*})

    return response;

}

*let* updatedata = async (*id*,*key*,*value*) *=>* {

*let* data = {}

    data[*key*]=*value*

*let* employeeModel = mongoose.model("employee",employeeSchema)

*let* response = await employeeModel.updateOne({\_id:*id*},{$set:data})

    return response;

}

*const* express = require("express")

*const* path = require("path")

*const* mainPath = path.join(\_\_dirname,"public")

*const* app = express()

app.use(express.static(mainPath))

app.set("view engine","ejs")

app.get("/data",(*\_*,*res*)*=>*{

    readdata().then((*data*)*=>*{

*res*.render("data",{data})

    })

})

app.get("/senddata",(*req*,*res*)*=>*{

    saveData(*req*.query.name).then((*response*)*=>*{

*res*.send(*response*)

    })

})

app.get("/deletedata",(*req*,*res*)*=>*{

    deleteData(*req*.query.id).then((*response*)*=>*{

*res*.send(*response*)

    })

})

app.get("/updatedata",(*req*,*res*)*=>*{

    updatedata(*req*.query.id,*req*.query.name,*req*.query.value).then((*response*)*=>*{

*res*.send(*response*)

    })

})

app.listen(4000)

All mongoose API’s

*const* exp = require("constants")

*const* mongoose = require("mongoose")

mongoose.connect("mongodb://0.0.0.0/my-data")

*let* employeeSchema =new mongoose.Schema({

    name:*String*,

    age: *Number*,

    department: *String*

})

*let* readdata = async () *=>* {

*let* employeeModel = mongoose.model("employee",employeeSchema)

*let* data = await employeeModel.find({})

    return data;

}

*let* saveData = async (*obj*) *=>* {

*let* employeeModel = mongoose.model("employee",employeeSchema)

*let* response = await employeeModel.create(*obj*)

    return response;

}

*let* deleteData = async (*id*) *=>* {

*let* employeeModel = mongoose.model("employee",employeeSchema)

*let* response = await employeeModel.deleteOne({\_id: *id*})

    return response;

}

*let* updatedata = async (*id*,*obj*) *=>* {

*let* employeeModel = mongoose.model("employee",employeeSchema)

*let* response = await employeeModel.updateOne({\_id:*id*},{$set:*obj*})

    return response;

}

*const* express = require("express")

*const* path = require("path")

*const* mainPath = path.join(\_\_dirname,"public")

*const* app = express()

app.use(express.json())

app.get("/",(*\_*,*res*)*=>*{

    readdata().then((*data*)*=>*{

*res*.send(*data*)

    })

})

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

    saveData(*req*.body).then((*response*)*=>*{

*res*.send(*response*)

    })

})

app.delete("/:id",(*req*,*res*)*=>*{

    deleteData(*req*.params.id).then((*response*)*=>*{

*res*.send(*response*)

    })

})

app.put("/:key",(*req*,*res*)*=>*{

    updatedata(*req*.params.key,*req*.body).then((*response*)*=>*{

*res*.send(*response*)

    })

})

app.listen(4000)

Search API in Mongoose

*let* searchData = async (*value*) *=>* {

*let* employeeModel = mongoose.model("employee",employeeSchema)

*let* response = await employeeModel.find({

        "$or": [

            {"name":{$regex:*value*}},

            {"department":{$regex:*value*}}

        ]

    })

    return response;

}

app.get("/search/:exp",(*req*,*res*)*=>*{

    searchData(*req*.params.exp).then((*response*)*=>* {

*res*.send(*response*)

    })

})

Upload File (multer)

*const* express = require("express")

*const* multer = require("multer")

*const* upload = multer({

    storage:multer.diskStorage({

        destination: (*req*,*file*,cb) *=>* {

            cb(null,"Files")

        },

        filename: (*req*,*file*,cb) *=>* {

            cb(null,`${*file*.fieldname}${*Date*.now()}.pdf`)

        }

    })

}).single("user\_pdf\_file")

*const* app = express()

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

*res*.send("Home Page")

})

app.post("/upload",upload,(*req*,*res*)*=>*{

*res*.send("Upload File Here")

})

app.listen(4000)