Notes

Moongoose

- It is basically used to establish a connection between the MongoDB and your server.
- You can do this with `Mongo Driver` as well, but `Mongoose` is a pretty good and easy to understand.
- What actually Mongoose do and why we are learning it?
    - Connect our server to the mongo DB database.
    - It helps us in defining the structure of the document.
    - Also helps in validation as well.
    - Now you would consider that, how it is flexible if we are fixing the structure ⇒ We can just basically change it anytime we want.
- Now lets go ahead and create a backend project.
- Install mongoose as well. ⇒ `npm i mongoose`
- Lets connect our Mongo db.
- As the connection is asynchronous and we will get the `connected to database`, even if there is some error as it may take some time to connect.

```
//connect.js
const mongoose=require("mongoose")

const main=async()=>{
  try{
    const connection=await mongoose.connect("mongodb://127.0.0.1:27017")
    console.log("Connected to Database")
    connection.disconnect() //can be used to disconnect the server
    console.log("Disconnected")
  }catch(err){
    console.log("Error connecting to DB")
    console.log(err)
  }
}

main()
```

```
//I can also use the following to disconnect the databas
connection.disconnect() //just to show we can disconnect as well
```

- Let us now define the `structure`
- We are going to learn `Schema` for it.
    - It is basically a `structure` or `blueprint` of the documents.
- What is the meaning of Model?
    - It is basically a `Mould`

```
//connect.js
const mongoose=require("mongoose")

//1. connecting the MongoDB
const main= async ()=>{
  try{
    const connection=await mongoose.connect("mongodb://127.0.0.1:27017/newdb")
    console.log("connected to database")
    await Studentmodel.insertMany([{name:"Rahul",age:22,city:"Pune"}])
    console.log("Added the data")
    connection.disconnect()
    console.log("Disconnected")
  }catch(err){
    console.log("Error connecting to DB")
    console.log(err)
  }
}

main()

//2. Defining the structure of the document
const studentSchema=mongoose.Schema({
  name:String,
  age:Number,
  city:String,
})
const Studentmodel=mongoose.model("student",studentSchema)
```

- Once we add the `document` we might get following:
    - `_id` ⇒ For unique identification and indexing, added by `mongoDB`
    - Similarly mongoose adds `__v`, to keep track of version, this process of adding somethings to the document is called `hydration`.
    - We can turn is off as well, by adding following to the schema:
        - `{versionKey:fasle}`

- Everyone must have learnt `constructor`, it is a blueprint using which we can create objects, right?
- Similarly using `Model` we are creating documents.
- Model is a `constructor function`
- Mongoose does not have `insertOne` , It just have `insertMany` .
- You can use `insertMany` to insert one document as well but **what if don’t want to use it** then?

```
//save the one document in a separate variable, create using new keyword
const student=new Studentmodel({
  name:"Pulkit",
  age:27,
  city:"Delhi"
})

await student.save()
```

- We have seen how to `add`, now we will learn how to `read` ?

```
const students=await Studentmodel.find()
console.log(students)

//all the students will be printed in the terminal
```

- Now you can use any kind of things that we saw in the `Mongo CRUD` session.

💡 **Validation**

Go through the last 10 minutes of the session for understanding in a better way, Also please research about it as well.