MongoDB

MongoDB Json structure….jodio atake bson bola hoi. Dekte json er moto so json bolle problem nai. Json eta txt formet. Akhon txt format a rakle she take read kora & query kore data bar kore ana shomoi shapekho or performance kharap hote pare. Jar karone mongodb binary kore rakhe mane bson kore rakhe.

#After install Command Prompt run korbo with administration

```
"C:\Program Files\MongoDB\Server\4.4\bin\mongo.exe" –version
```

#EVN   //env search korbo environment variable create korbo

```
#C:\Windows\system32>mongo            //Comand promt a mongod, mongo run korbo,
MongoDB shell version v4.4.5
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Error: couldn't connect to server 127.0.0.1:27017, connection attempt failed: SocketException: Error connecting to 127.0.0.1:27017 :: caused by :: No connection could be made because the target machine actively refused it. :
connect@src/mongo/shell/mongo.js:374:17
@(connect):2:6
exception: connect failed
exiting with code 1
```

- ⇨ Ai rokonm hole amr run korbo.   #Connection fail karon amra mongoDB server start kori nai. Amr server start hoinai service theke aita chara e kaj kore

#Service search korbo, mongodn start kore dibo service a. start na hoile o thik ase.

```
#mongo
MongoDB shell version v4.4.5
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("b0ef29d0-bc61-4175-8cbd-88823c7efb89") }
MongoDB server version: 4.4.5
---
The server generated these startup warnings when booting:
        2021-10-31T21:11:47.278+06:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
        2021-10-31T21:11:47.278+06:00: This server is bound to localhost. Remote systems will be unable to connect to this server. Start the server with --bind_ip <address> to specify which IP addresses it should serve responses from, or with --bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1 to disable this warning
---
---
        Enable MongoDB's free cloud-based monitoring service, which will then receive and display
        metrics about your deployment (disk utilization, CPU, operation statistics, etc).

        The monitoring data will be available on a MongoDB website with a unique URL accessible to you
        and anyone you share the URL with. MongoDB may use this information to make product
        improvements and to suggest MongoDB products and deployment options to you.

        To enable free monitoring, run the following command: db.enableFreeMonitoring()
        To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
```

```
==Aita asha mane thik ase
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
```

```
#Akta record add korer jonno insertOne
Database create command: >use ecommerce> db > db.products.insertOne({name: 'iphone', price: 10000});
Ans: {
        "acknowledged" : true,
        "insertedId" : ObjectId("617f5a5777fb623e91faf304")
}
> db.products.find()
Ans: { "_id" : ObjectId("617f5a5777fb623e91faf304"), "name" : "iphone", "price" : 10000 }


>cls    //command shell clear korer jonno cls

> db.products.find().pretty()
{
        "_id" : ObjectId("617f5a5777fb623e91faf304"),
        "name" : "iphone",
        "price" : 10000
}
//Sundor kore dekhanor jonno pretty

>db.products.insertMany([{name: 'HP', price: '1234'}, {name: 'Dell', price: 12365}])
> db.products.find({name: 'HP'}).pretty()  // product dekte chile

//kono kisu dekte/dekhate na chile 0,1
> db.products.find({name: 'HP'}, {price: 0}).pretty()

//koita data dekte chi bole dite pari
> db.products.find({name: 'HP'}, {price: 0}).pretty() .limit(1)
//porer data ta dekte chassi
> db.products.find({name: 'HP'}, {price: 0}).pretty() .limit(1).skip(1)
```

//FindOnek er shate pretty use korte parbo na

//Data update korbo er jonno updateOne, updateMany

```
>db.products.updateOne({name: 'HP'}, {$set: {price: 0987}})
```

//onek gula update korbo

```
>db.products.updateMany({}, {$set: {active: true}})
```

//shob product dekte

```
>db.products.find()
```

//delete korbo

```
>db.products.deleteOne({name: 'HP'})
>db.products.deleteMany({})
```

Akta database er kaj muloto 4 ta CRUD.

MYSql a myAmin ase tamon MongoDB te mongo Compass ase shob dekhar jonno ai vabe type kore kore command kore likte hobe na.

Mongo compass a new Connection a er jaita ta te ekta utl, host dite hoi ami kothai run korte chassi. Na dile o default connection hobe.
Password, username na thakle = mongodb://localhost:27017   dita dibo.

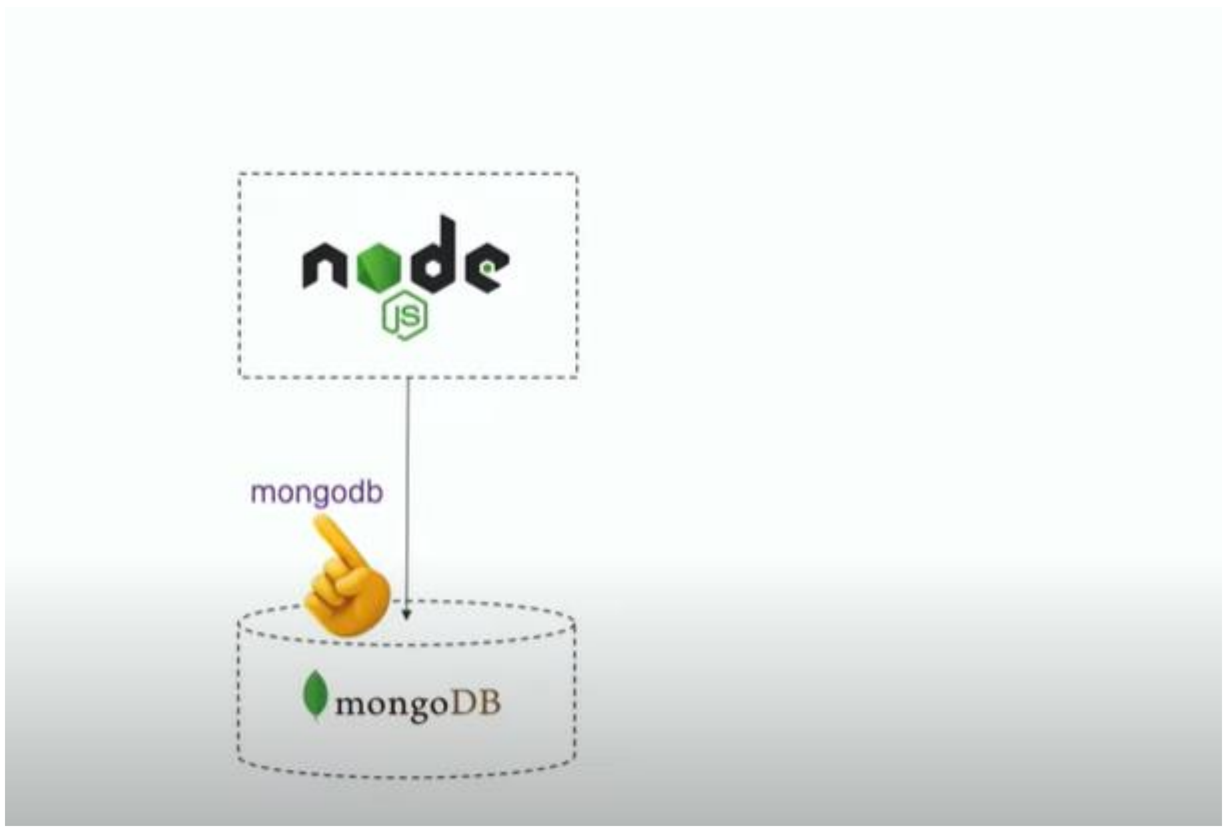
Tutorial Dekhar por kokhono tutorial dekher dorkar nai. Ai note gula e dekbo & documentation porbo.

Mongoose

Mongoose kano use korbo?

Aita Object Data Modeling= ODM.

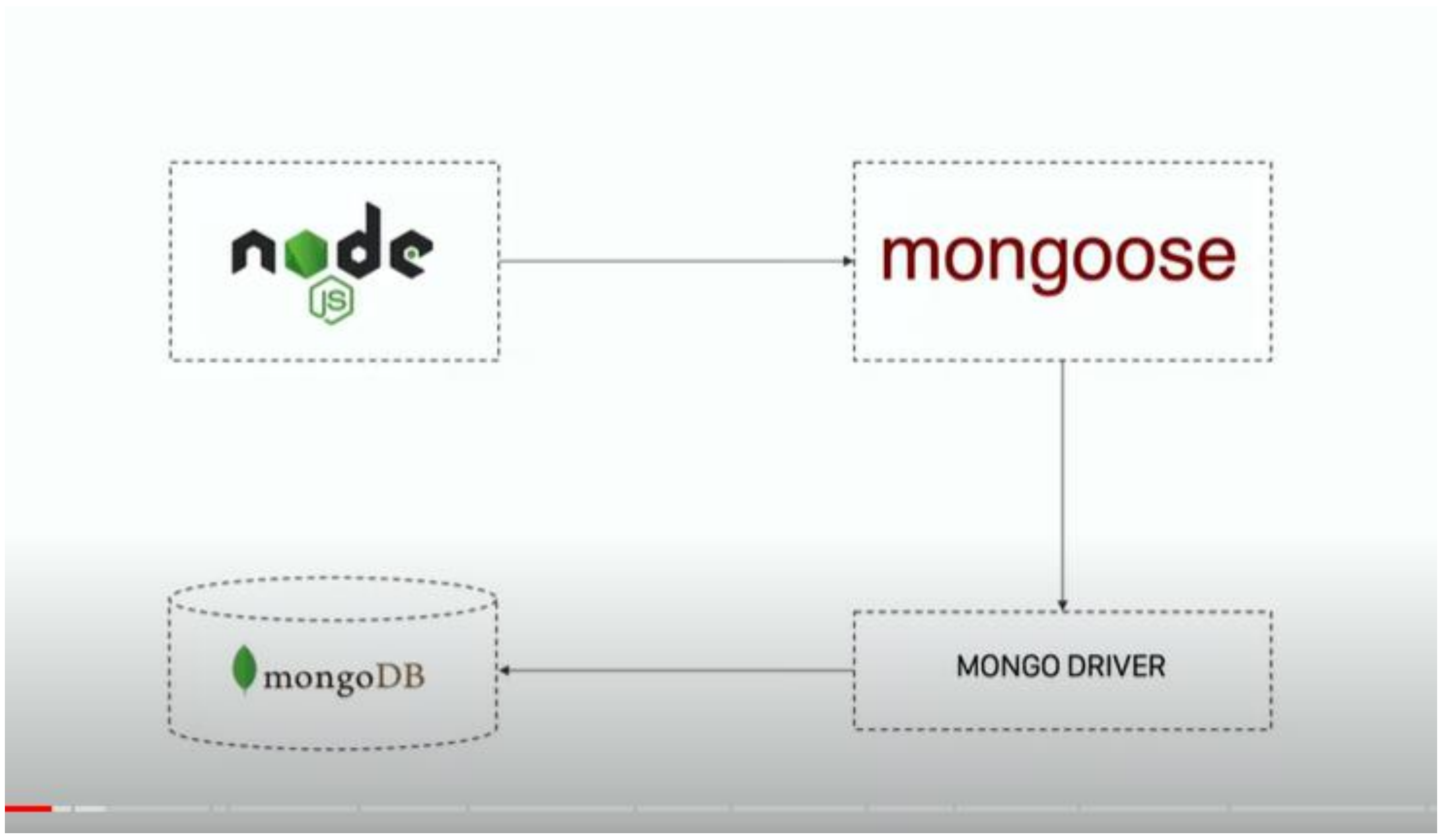
MongoDB te data bson akare thake. So amra ai data niya kaj korte chile amder raw mongodb niya kaj korte hobe. Aita ektu problematic tai amra mongoose diya kaj korbo. Karon mongoose data gulo k object er moto kore modeling korbe.



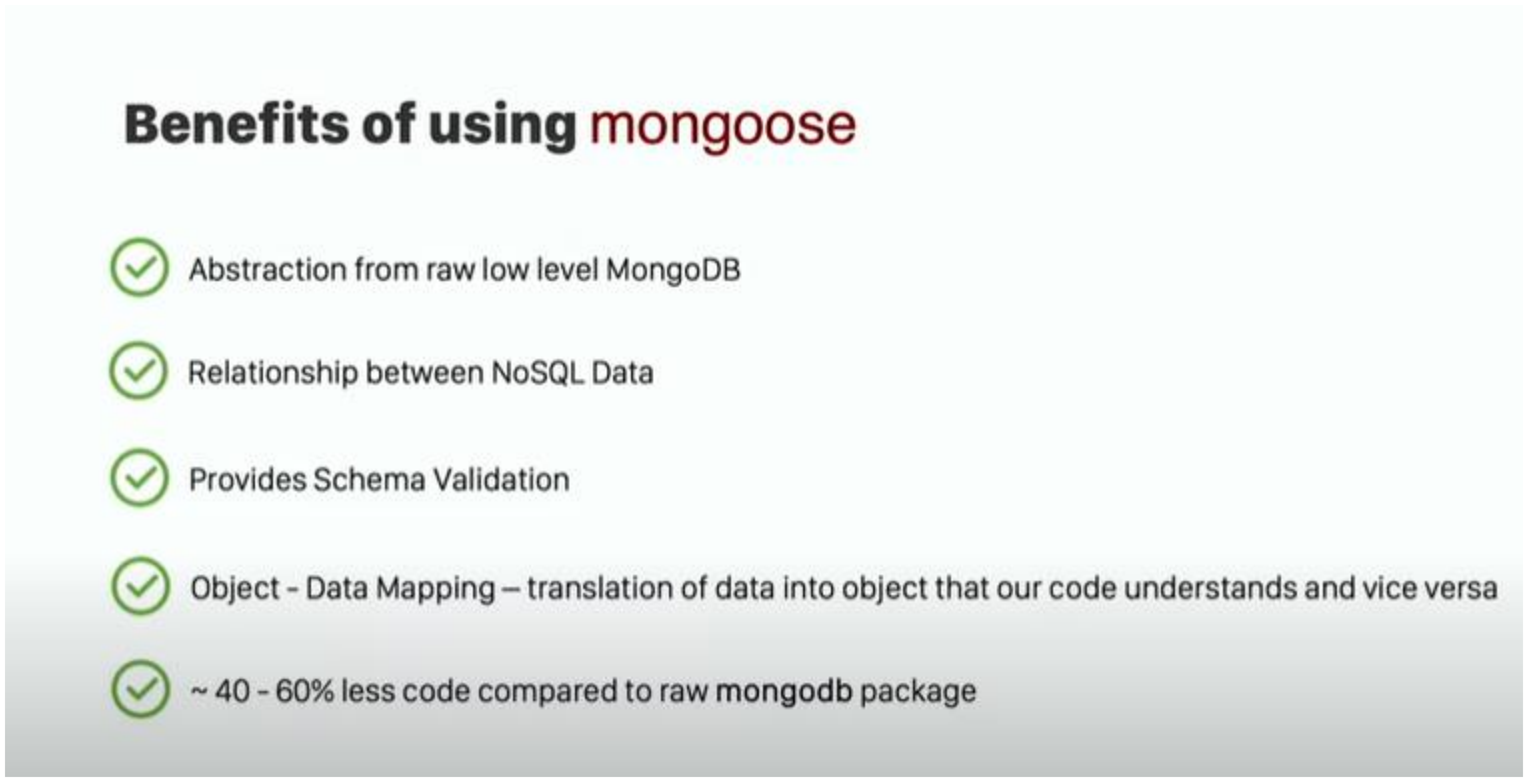
Amra chile shora shori node & mongodb k connect korte pari mongodb package er maddome.

npm install mongodb kore node er shate use kora jai. But aita hosse raw mongodb So aita ai vabe use korbo na amra mongoose er maddome use korbo.Mane kaj ta ektu guriya krobo.

//Amra nodejs er shate mongoose k connect kore dibo er por mongoose in the backend mongo driver er madhome mongodb k connect kore nibe. (Driver hosse ekta interface toiri kore jetar maddome amra connectivity korte pari So mongo driver hosse mongoose er shate mongodb er ekta driver).



Eta korer karone implicitly nodeJs er shate ekta object mapping hoiya jabe mongoDB database er. Mane amra data base theke je data gulo pabo she gulo k amra simple vanilla js er moto kore kolpona korte parbo. Abong she vabe e she gulo k niya kaj korbo. So er por amra jokhon e kono database query korbo dekle mone hobe amra kono js object er shate kaj korci. So amra Jodi ai kaj ta direct mongodb diya kortam amra ek bare object petam na amder object baniya nite hoto. Mut kotha amra ekta lear toiri koraci



Benefits of using mongoose

- Abstraction from raw low level MongoDB
- Relationship between NoSQL Data
- Provides Schema Validation
- Object - Data Mapping – translation of data into object that our code understands and vice versa
- ~ 40 - 60% less code compared to raw **mongodb** package

Akhon shudu mongodb use na kore shate mongoose use koraer karone hoito ektu performance kharap hote pare but aita khub e neglageble/kogono. Mongoose developer der jai shubida ta disse onek bashi ai performance er theke. Big application a mongoose oboshoi lagbe. Er maddome amder onek kom code likha lage.

npm install mongoose

```
const express = require('express');
const mongoose = require('mongoose');

const app = express();
app.use(express.json());


//database connection with mongoos locally & remote amra onno ek shomoi dekbo
//Connection string
mongoose.connect('mongodb://localhost/todos', {useNewUrlParse: true,
useUnifiedTopolo})   //aita akta async way
  .then(() => console.log('connection successful'))
  .catch((err) => console.log(err));
```

Index.js

```
const express = require("express");
const mongoose = require("mongoose");
const todoHandler = require('./routeHandler/todoHandler')

//express app initialization
const app = express();
app.use(express.json());//amra jeta pabo ekta js object e pabo

//database connection with mongoos locally & remote amra onno ek shomoi dekbo
//todos name amder kono database nai but kaj korbe
mongoose
  .connect("mongodb://localhost/todos", {
    useUnifiedTopology: true,
  }) //aita akta async way
  .then(() => console.log("connection successful"))
  .catch((err) => console.log(err));


app.use('/todo', todoHandler);


//default error handler
function errorHandler(err, req, res, next) {
    if (req.headler) {
        return next(err);
    }
    res.status(500).json({ error: err });
}

app.listen(3000, () => {
    console.log('app listening at port 3000');
});
```

routeHandler/todoHandler.js

```
const express = require('express');
const mongoose = require('mongoose');
const router = express.Router();
const todoSchema = require('../schemas/todoSchema');

const Todo = new mongoose.model('Todo', todoSchema);//mongoose.model ekta class
return kore tai new dite hosse //fast paramtter model er name ja singular dibo, polr
na. 2nd parameter dibo kon schema use korbe.
//akhon amra ekta Todo object pelam

//todo name database create hobe collection/table toiri hobe todos name

//Routing setup.....

//get all todos
router.get('/', async (req, res) => {

})

//get a todo by ID
router.get('/:id', async (req, res) => {

})

//post todos
router.post('/', async (req, res) => {
    const newTodo = new Todo(req.body);
    await newTodo.save((err) => {
        if (err) {
            res.status(500).json({
                error: "There was a server side error!",
            });
        }
        else {
            res.status(200).json({
                message: "Todo was inserted successfully!",
            })
        }
    })
})

//post multiple todo todos
router.post('/all', async (req, res) => {

})

//put todo
router.put('/:id', async (req, res) => {

})

//Delete todo
router.delete('/:id', async (req, res) => {

})
module.exports = router;
```

todoSchema.js

```
//schema design

const mongoose = require('mongoose');
const todoSchema = mongoose.Schema({
    //fild... validation korbo
    title: {//mongo te aita korte parbo na
        type: String,
        requird: true,
    },
    description: String,   //aita r require korno na
    status: {
        type: String,
        enum: ['active', 'inactive']
    },
    date: {
        type: Date,
        default: Date.now,
    }
})
//mongoose documentation ai gula ase

//ai Schema k use kore model banabo, shei model er maddome amra bibbino method, 
property use kore table a data insert update shob kisu korte parbo.
```

```
module.exports = todoSchema;
```

post: http://localhost:3000/todo

Body-raw-json:

```
{
    "title": "From learn with Sumit",
    "description": "Very Informative",
    "status": "active"
}
{
    "message": "Todo was inserted successfully!"
}
//database:
```

{"_id":{"$oid":"617fd89f84424fe0631dde61"},"title":"From learn with Sumit","description":"Very Informative","status":"active","date":{"$date":"2021-11-01T12:07:59.493Z"},"__v":0}

todoHandler.js

```
//post multiple todos
router.post("/all", async (req, res) => {
  await Todo.insertMany(req.body, (err) => {
    //2nd paramert ekta callback

    if (err) {
      res.status(500).json({
        error: "There was a server side error!",
      });
    } else {
      res.status(200).json({
        message: "Todos were inserted successfully!",
      });
    }
  });
});
```

Postman:

Post: http://localhost:3000/todo/all

```
[
    {"title": "From learner","description": "Informative","status": "active"},
    {"title": "Sumit","description": "Good Person","status": "inactive"},
    {"title": "Mongoose","description": "Database","status": "active"}

]

{
    "message": "Todos were inserted successfully!"
}
```

//Run hoi ni ai gula

```
//put todo
router.put("/:id", async (req, res) => {
  await Todo.updateOne(
    { _id: req.params.id },
    {
      $set: {
        status: "active",
      },
    },
    (err) => {
      if (err) {
        res.status(500).json({
          error: "There was a server side error!",
        });
      } else {
        res.status(200).json({
          message: "Todos was updated successfully!",
        });
      }
    }
  );
});
```

Postman:

Put: http://localhost:3000/todo/617fe3269baa9f45f5018205

Aro are update many ai gula MongoDB Documentation theke shikhe nibo. Documentation porte hobe.

```
//get all todos
router.get("/", async (req, res) => {
  await Todo.find({ status: 'active' }, (err, data) => {
    if (err) {
      res.status(500).json({
        error: "There was a server side error!",
      });
    } else {
      res.status(200).json({
        result: data,
        message: "Todo was inserted successfully!",
      });
    }
  })
});
```

Postman: Get: http://localhost:3000/todo/

.Ai code a ekta vule ase ta holo async await & callback use kora hoise ja kokhono e 1 shate use kora hoi na. ja kono ekta korte hoi .

```
//get all todos chain kore
router.get("/", async (req, res) => {
  await Todo.find({ status: 'active' }).select({
    //ai gula user k dekhabo na tai 0 kore disi
    _id: 0,
    _v: 0,//v dakhabai
    date: 0,
  })
.limit(2)
.exec((err, data) => {
    if (err) {
      res.status(500).json({
        error: "There was a server side error!",
      });
    } else {
      res.status(200).json({
        message: "success!",
      });
    }
  })
  //ai execution a callback pabo
});
```

Postman: Get: http://localhost:3000/todo/

Vule er update:

```
//get all todos chain kore
router.get("/", async (req, res) => {
  try {
    const data = await Todo.find({ _id: req.params.id });
    res.status(200).json({
      result: data,
      message: "Success"
    })
  }
  catch (err) {
    res.status(500).json({
      error: "There was a server side error!"
    })
  }
});
```

```
//Delete todo
router.delete("/:id", async (req, res) => {
   await Todo.deleteOne({_id: req.params.id}, (err, data) => {
     if (err) {
       res.status(500).json({
         error: "There was a server side error!",
       });
     } else {
       res.status(200).json({
         message: "Todo was delete successfully!",
       });
     }
   });
});
```

Postman: delete: http://localhost:3000/todo/617fe3269baa9f45f5018205

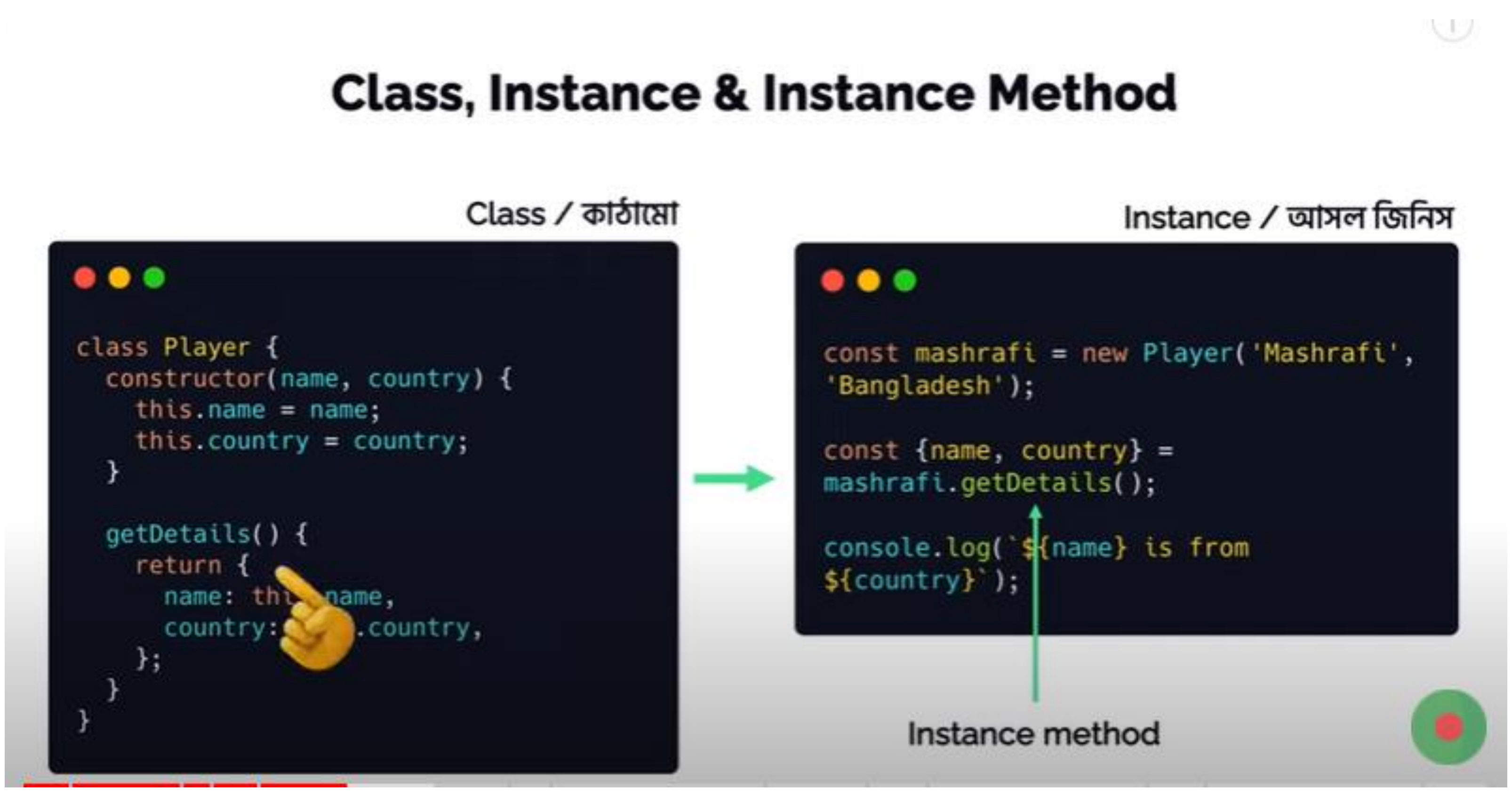
Mongoose Instance Method, Static & Query Helpers:

Instance:**

New keyword use hoi class er age.

Mashrafi Player class er ekta instance. Class hosse kathamo & Instance hosse ashol jinish.

getDetails() hosse ai khane instance method.



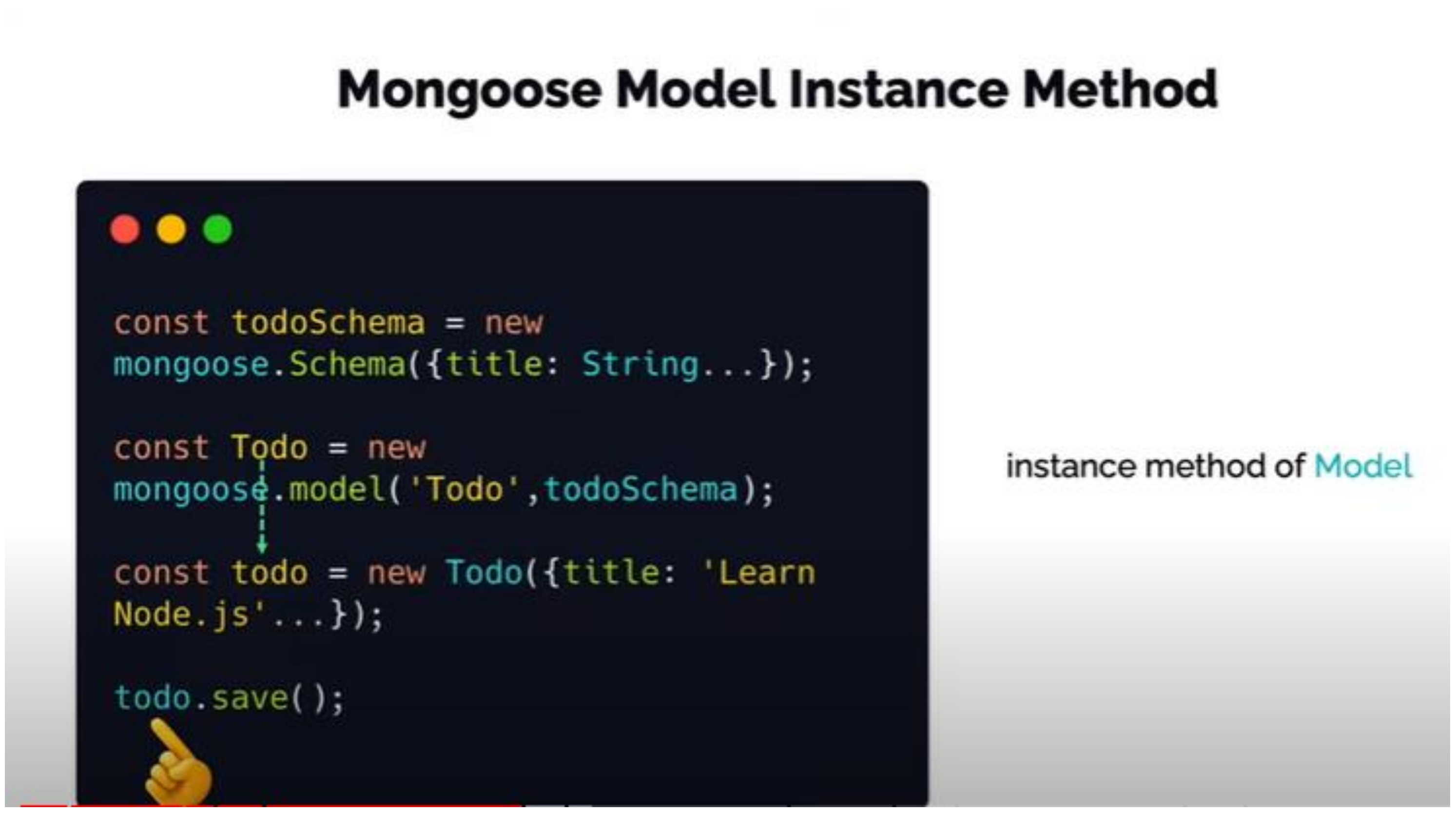
Class, Instance & Instance Method

```
class Player {
  constructor(name, country) {
    this.name = name;
    this.country = country;
  }

  getDetails() {
    return {
      name: this.name,
      country: this.country,
    };
  }
}
```

```
const mashrafi = new Player('Mashrafi',
'Bangladesh');

const {name, country} =
mashrafi.getDetails();

console.log(`${name} is from
${country}`);
```

Mongoose Schema: Amra database a je data ta rakbo shai data tar ekta proper structure dawa ta hosse Schema. Schema k ekta javascript object er moto chinta korte pari. Je object er modda bole dewa ase amr data ta kamon hobe.

Mongoose.Schema mongoose er ekta function. Je take call kore amra ekta class passi. Mongoose,Schema amk ekta class disse.



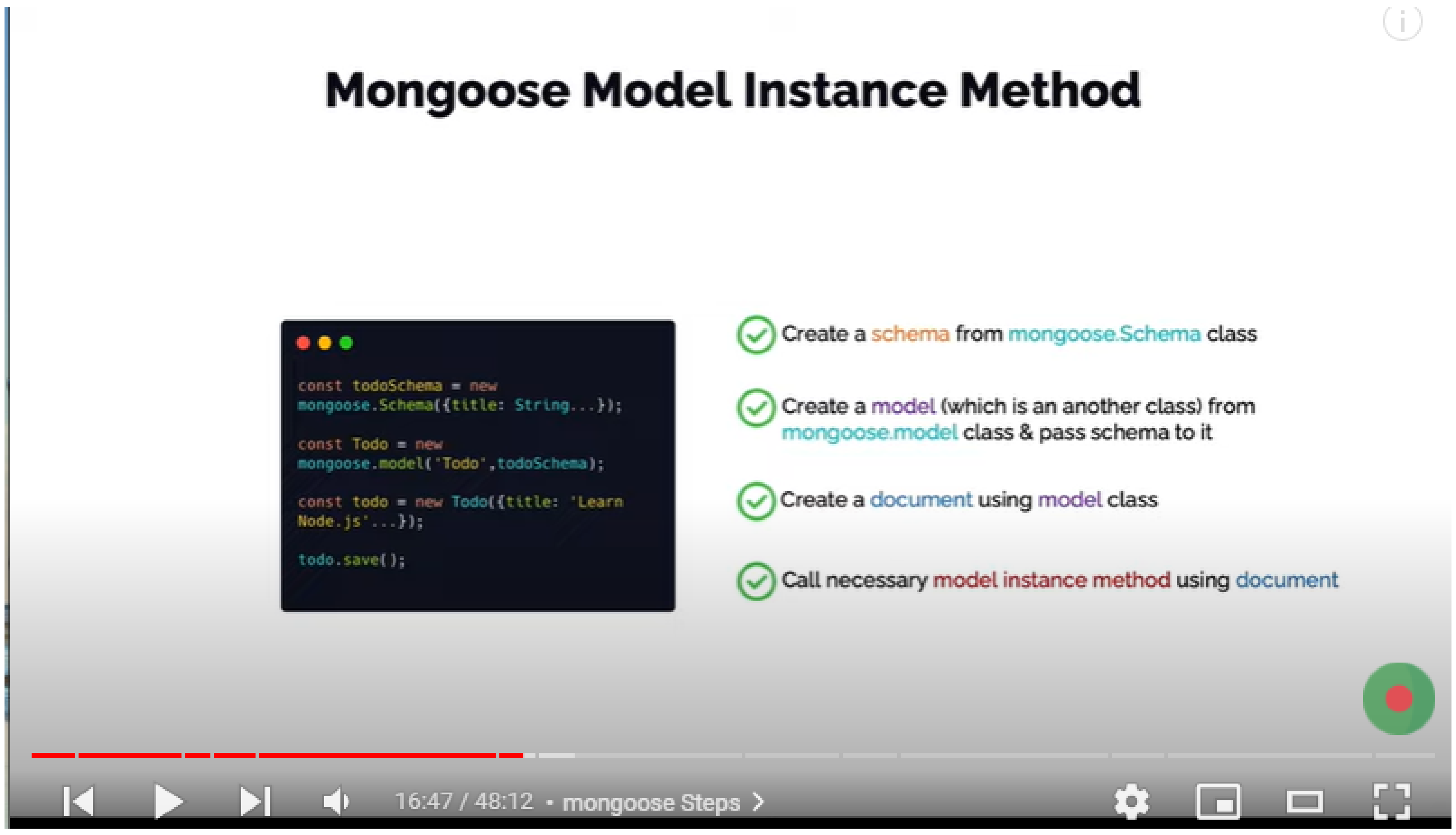
Mongoose Model Instance Method

```
const todoSchema = new
mongoose.Schema({title: String...});

const Todo = new
mongoose.model('Todo',todoSchema);

const todo = new Todo({title: 'Learn
Node.js'...});

todo.save();
```

todoSchema ta hosse amr actual schema.

Model ta hosse amr database je table ta ase tar model.

Todo ta hosse document class.



Mongoose Model Instance Method

```
const todoSchema = new
mongoose.Schema({title: String...});

const Todo = new
mongoose.model('Todo',todoSchema);

const todo = new Todo({title: 'Learn
Node.js'...});

todo.save();
```

- ✓ Create a schema from mongoose.Schema class
- ✓ Create a model (which is an another class) from mongoose.model class & pass schema to it
- ✓ Create a document using model class
- ✓ Call necessary model instance method using document

Instance Method:

Method = schema er ekta property

Customs Instance Method…

```
//get Active todos
router.get("/active", async (req, res) => {
  const todo = new Todo();
  const data = await todo.findActive();
  res.status(200).json({
    data,
  })
});
```

```
//instance method
todoSchema.methods = {
    findActive: function () {
        //mongoose query
        return mongoose.model('Todo').find({status: 'active'});
    }
}
```

Postman: get: http://localhost:3000/todo/active

//Get Active todo using callback

```
//get Active todos using callback
router.get("/active-callback", async (req, res) => {
  const todo = new Todo();
  todo.findActiveCallback((err, data) => {
   res.status(200).json({
     data,
   });
 });

});
```

```
//instance method
todoSchema.methods = {
  findActive: function () {
    //mongoose query
    return mongoose.model("Todo").find({ status: "active" });
  },
  findActive: function (cb) {
    //mongoose query
    return mongoose.model("Todo").find({ status: "active" }, cb);
  },
};
```

Static Method:

```
//JavaScript Static
class Person{
    constructor(name, age) {
        this.name = name;
    }
    eat() {
        console.log("Eat")
    }
    static isEqualAge() {
        console.log('I am Static');
    }
}
let sakib = new Person("sakib", 34);

Person.isEqualAge(); //static method tai er sakib.isEqualAge korte hoi nai.
sakib.eat();
```

custom static method:

```
//custom static method
router.get('/js', async (req, res) => {
  const data = await Todo.findByJS();
  res.status(200).json({
    data,
  })
});
```

```
//static method
todoSchema.statics = {
    findByJS: function () {
        return this.find({title: /js/i});
    }
}
```

Query Helper:

```
//get todos by Language Query helper
router.get('/language', async (req, res) => {
  const data = await Todo.find().byLanguage("js");
  res.status(200).json({
    data,
  })
});
```

```
//Query helpers
todoSchema.query = {
    byLanguage: function (language) {
      return this.find({ title: new RegExp(language, "i") }); //new RegExp(language, "i") or {title: /js/i} = aita te variable dewa jai na.
    }
}
```

Query helper amder quey korte help korbe.

Instance, static, query method function she gulo te arrow function use kora jabe na. Karon egulo titly cupple hoiya ase class er shate.