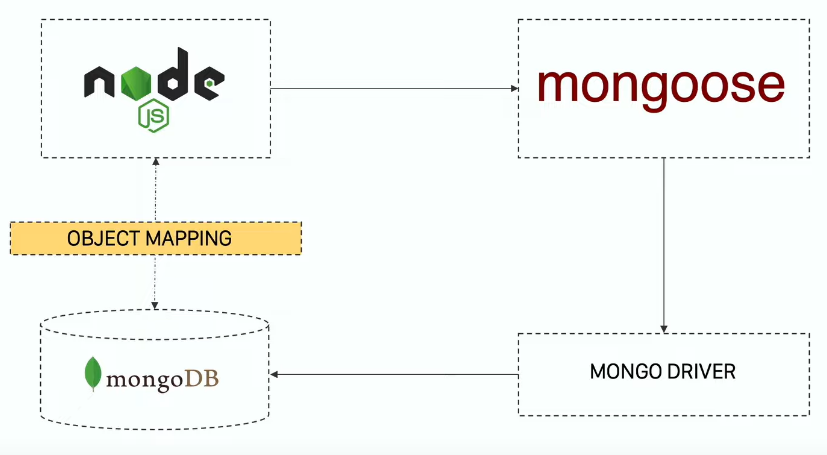
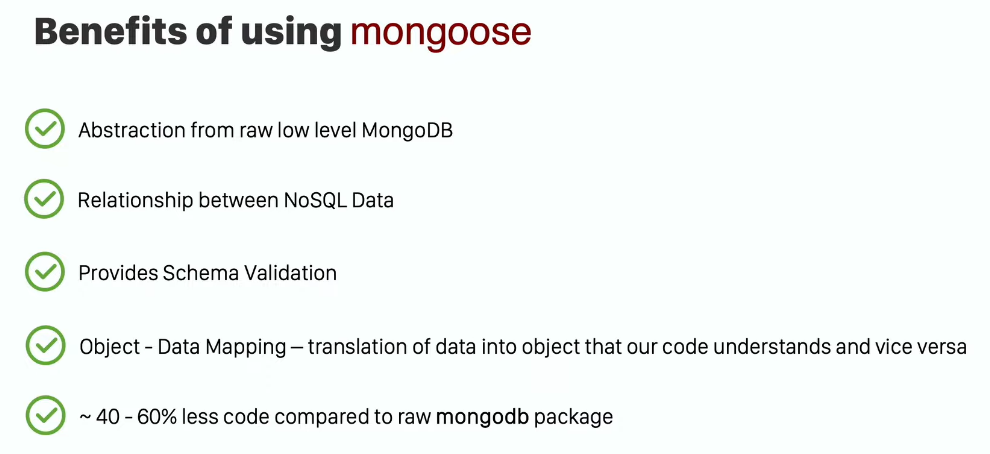
Mongoose CRUD Application with express js

Ajke amra mongodb er ekta package ‘mongoose’ (jetar maddhome ami mongodb er shathe express js ke connect korte parbo) --- sheta niye discuss korbo.

Amra keno mongoose use korbo? Node js er jonno mongodb er nijjossho ekta package ache ‘mongodb’ name. ‘npm i mongodb’ kore sheta install kora jay. But eta use na kore amra mongoose use korbo. Mongoose hocche node js er jonno ekta object data modeling (ODM). Ei object bolte js object ke bujhtesi. Js e shob kichui kei object er moto chinta kora hoy. Js e jekono jinish niye kaj korar jonno shetake object e niye jawa hocche shobche easy ekta bepar. Mongodb database e data gula bison (json er moto) akare thake. Ei jinishta niye directly kaj korte chaile raw mongodb niye kaj korte hobe. Eta ektu problematic bepar. Tai amra mongoose er maddhome kaj korbo. Ei mongoose database er datagulake js object er moto kore modeling korbe. orthat databse ke simply amra ekta js object er moto kore imagine korte parbo. etakei bole ODM. Orthat object ke datar moto kore model kora.



To summarize, amra node js ke mongodb er shathe connect korte pari ‘mongodb’ package er maddhome. But eta na kore jodi mongoose use kori tahole kaj ta ektu ghuriye hobe. Nodejs ba express js er shathe amra first e mongoose ke connect korbo. Etuku korei amader kaj shesh. Mongoose in the back end mongodb er ekta driver er maddhome mongodb database er shathe implicitly ekta connectivity create korbe. er fole nodejs ba express js er shathe mongodb er ekta object mapping hoye jabe. Er mane amra database theke je datagula pabo shegula amra simple js object er moto kore chinta korte parbo. Ebong shevabei amra sheta niye kaj korbo. Amra jokhoni kono database query korbo dekhle mone amra ekta js object er shathe kaj kortesi. Infact amra jokhon mongoose er maddhome mongodb theke data retrieve korbo tokhon amra dekhbo shei jinishta ekta js object. Amra jodi ei kajtai ‘mongodb’ diye kortam amra direct object petam na. sheta ke object banay nite hoto.



Ebar ekta simple ‘todo’ application banabo jekhane ekta CRUD system thakbe. ekta API banabo. API bananor maddhome kivabe mongoose connect korte hoy, kivabe datar upor CRUD operation chalate hoy egula janbo.

Index.js …

*const* express = *require*("express");

*// express app initialization*

*const* app = *express*();

app.*use*(express.*json*());

*// application routes*

*// default error handler*

*function* *errorHandler*(*err*, *req*, *res*, *next*) {

  if (res.headersSent) {

*return* *next*(err);

  } else {

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

  }

}

app.*listen*(3000, () *=>* {

  console.*log*("Listening on port 3000");

});

Npm install mongoose kore package ta install korlam. Index.js e mongoose ke require kore anbo ekhon. Then mongoose.connect() kore express er shathe mongodb connect korbo. Ei connect() method er moddhe database er url ta diye dite hobe. To ami “mongodb://localhost/todos”. Ekhane ‘todos’ amar database er name. jodio todos database mongodb database e ekhono nai. Eta runtime e create hobe.

Mongoose.connect(“mongodb://localhost/todos”) --- etar ortho holo ami bole dicchi tumi mongoose er maddhome connect koro mongodb er shathe. Localhost hocche amar host. Ebong database er name hocche ‘todos’. Ei method ta ekta promise return korbe. jodi connection successful hoy tahole then er moddhe console.log(‘connection successful’) likhe dilam. Ar jodi error ashe console.log(err) kore dilam catch er moddhe. Npm start kore dilam. Console e ‘connection successful’ ashlo.

Index.js …

*const* express = *require*("express");

*const* mongoose = *require*("mongoose");

*// express app initialization*

*const* app = *express*();

app.*use*(express.*json*());

*// database connection with mongoose*

mongoose

  .*connect*("mongodb://localhost/todos")

  .*then*(() *=>* console.*log*("connection successful"))

  .*catch*((*err*) *=>* console.*log*(err));

*// application routes*

*// default error handler*

*function* *errorHandler*(*err*, *req*, *res*, *next*) {

  if (res.headersSent) {

*return* *next*(err);

  } else {

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

  }

}

app.*listen*(3000, () *=>* {

  console.*log*("Listening on port 3000");

});

simple etuku code likhe amra database er shathe connection kore fellam. jodi amra local database er shathe connection na kore kono remote mongodb database er shathe connect korte chai jeta amra porobortite amader series e dekhbo tahole amra connect() method er moddhe shei host er url ta dibo. Ei url ta ke bole mongodb er connection string. Ei string er moddhe username, pswd etc deya jay. Ebar amra amader application e bivinno route declare korbo. Ebong shei route er maddhome bivinno api banabo. Amra ekta simple todo api banabo. Shetar jonno application routes section e app.use(‘/todo’, todoHandler) kore dilam. Orthat keu jokhon /todo route hit korbe tokhon amra todoHandler diye request ta handle korbo. TodoHandler ta amra onno ekta file e likhbo then index.js e import korbo. routeHandler name ekta folder create korlam. Tate todoHandler.js name ekta file nicchi. Shekhane express require kore ene express.Router() call kore router object create kore get, post, put, delete er jonno dorkari codegula likhbo.

todoHandler.js …

*const* express = *require*("express");

*const* router = express.*Router*();

*// get all the TODOS*

router.*get*("/", async (*req*, *res*) *=>* {});

*// get a TODO by id*

router.*get*("/:id", async (*req*, *res*) *=>* {});

*// post TODO*

router.*post*("/", async (*req*, *res*) *=>* {});

*// post multiple TODO*

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;

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*());

*// database connection with mongoose*

mongoose

  .*connect*("mongodb://localhost/todos")

  .*then*(() *=>* console.*log*("connection successful"))

  .*catch*((*err*) *=>* console.*log*(err));

*// application routes*

app.*use*("/todo", todoHandler);

*// default error handler*

*function* *errorHandler*(*err*, *req*, *res*, *next*) {

  if (res.headersSent) {

*return* *next*(err);

  } else {

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

  }

}

app.*listen*(3000, () *=>* {

  console.*log*("Listening on port 3000");

});

Ekhane todoHandler.js e ekekta router er callback hishebe asynchronous function use kora hoise.

First e dekhbo kivabe ekta todo amader database e insert korte pari. So first e amra router.post er moddhe kaj korbo. Mongoose use korar ekta benefit hocche amra database er ekta schema design korte pari. Orthat amader database er structure ta Kemon hobe sheta ami mongodb ke enforce korte parbo. Databese je ‘todo’ name je collection create korbo tar chehara ta Kemon hobe sheta schema diye design korbo. ‘schemas’ name ekta folder create kore ‘todoSchema.js’ name ekta file nibo. Etate mongoose require kore anlam. Schema babanor khetre mongoose amader mongoose.Schema() ekta method dey. Etar moddhe argument hishabe ekta object dibo. Ei object er moddhe amar database collection er basic structure thakbe. ei object er 1st field ba property dilam ‘title’. Ei ‘title’ er ‘type’ dilam string. ‘required’ dilam true. Mane eta html er required er moto required. Orthat database e data deyar shomoy eta faka rakha jabe na. mongoose use korar karone ami ei validation pacchi. Raw mongodb te direct erokom schema validation nai. Jehetu ami mongoose er maddhome mongodb te connected hocchi data ta jokhon mongodb te insert hote jabe tokhon sheta mongoose er maddhome jabe ebong mongoose check korbe amar deya data schema object er shathe match kore kina. Match na korle mongodb te data dibe na. schema object er propertygula shomporke jante er documentation dekhte hobe. Erpor object e 2nd field ba property dibo ‘description’. Etate required option rakthe chacchi na. just type er value string rakhte chacchi. tai just ‘description: string’ kore dilei hobe. Required rakhle ‘title’ er moto kore value dite hoto. Erpor ‘status’ field e type er pashapashi status field er value duita value er moddhe rakhte chacchi ---- ‘active’ or ‘inactive’. Er jonno ‘enum’ property er moddhe er value set kore dilam. Ekhane je type use kortesi sheta js er type na. eta mongoose er alada type. At last todoSchema.js theke todoSchema ke export kore dilam.

Now todoHandler.js e mongoose ebong todoSchema ke require kore anlam ebong ‘todoSchema’ namok variable er moddhe save korlam. Ei todoSchema er maddhome ami ekta model banabo. Model bananor jonno Todo name ekta variable nilam. Er moddhe new mongoose.model(“Todo”, todoSchema) call korlam. Eta ekta class return kore. Tai new disi. Ekhane new mongoose.model() er vitore ‘Todo’ ta hocche model er name. convention holo ei model er name capital letter diye shuru korte hoy ebong singular number e dite hoy like ‘Todo’ deya jabe but ‘Todos’ deya jabe na. ‘Todo’ dile mongodb collection create hobe ‘Todos’ name. 2nd parameter e todoSchema diye bole dilam ei schema ta follow korte. So summary holo amra first e ekta schema banabo. Then ei schema er upor base kore ekta model banabo. Jate ei model er maddhome amra object mapping korte pari.

Ekhon shobar first e router.post() er maddhome ekta todo post korbo. Ami expect korchi ami jokhon postman theke ekta todo post korbo tokhon sheta express er req.body er moddhe save hobe. Req.body er moddhe save howar beparta ensure korechi index.js er moddhe app.use(express.json()) korar maddhome. Orthat client request er shathe ja ashbe sheta ekta parsed js object hobe. router.post(“/”) er callback function e newTodo name ekta constant variable nicchi. Etar moddhe Todo class ta call kore ekta object create korchi. Call korar shomoy req.body ta argument akare dibo. Req.body er moddhe amar pura object ta ache. Erpor await newTodo.save() kore dibo. Await use korlam karon databse e data insert korata ekta asynchronous kaj. Data insertion er shomoy error hote pare. Ei error ta save() method er moddhe handle korbo. Save() method er moddhe parameter akare ekta function dibo error handling er jonno.

Ekhon postman e giye post method e localhost:3000/todo route e {

    "title": "Learn Vanilla JS from w3schools",

    "description": "Vanilla JS is awesome",

    "status": "active"

}

Ei json diye hit korlam.

Console e ashlo “Todo was inserted successfully”

Ekhon router.post(“/all”) e multiple todo insert korar way ta dekhbo. ekhetre Postman theke request er shathe ekta array of objects ashbe. Ekhon router.post() er callback function e await Todo.insertMany(req.body, errorhandlingfunction) call korbo. Ekhane array ta req.body er moddhe thakbe. ager barer moto ekta error handling function insertMany() er 2nd parameter e diye dilam. Ekhon postman e localhost:3000/todo/all route e post method e request e

[

    {"title": "Learn React", "description": "Learn React details from LWS", "status":"active"},

    {"title": "Learn Node js", "description": "Learn Node details from LWS", "status":"active"},

    {"title": "Learn CSS", "description": "Learn CSS details from LWS", "status":"inactive"}

]

Eta shoho hit korlam. Body te raw ebong json select kora chilo. Console e ashlo ‘Todos were inserted successfully’. mongodbCompass e giye dekhlam 3 ta notun todo shoho mot 4 ta todo chole ashche.

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*());

*// database connection with mongoose*

mongoose

  .*connect*("mongodb://localhost:27017/todos")

  .*then*(() *=>* console.*log*("connection successful"))

  .*catch*((*err*) *=>* console.*log*(err));

*// application routes*

app.*use*("/todo", todoHandler);

*// default error handler*

*function* *errorHandler*(*err*, *req*, *res*, *next*) {

  if (res.headersSent) {

*return* *next*(err);

  } else {

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

  }

}

app.*listen*(3000, () *=>* {

  console.*log*("Listening on port 3000");

});

todoSchema.js …

*const* mongoose = *require*("mongoose");

*const* todoSchema = mongoose.*Schema*({

  title: {

    type: String,

    required: true,

  },

  description: String,

  status: {

    type: String,

    enum: ["active", "inactive"],

  },

  date: {

    type: Date,

    default: Date.*now*(),

  },

});

module.exports = todoSchema;

todoHandler.js …

*const* express = *require*("express");

*const* mongoose = *require*("mongoose");

*const* todoSchema = *require*("../schemas/todoSchema");

*const* router = express.*Router*();

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

*// get all the TODOS*

router.*get*("/", async (*req*, *res*) *=>* {});

*// get a TODO by id*

router.*get*("/:id", async (*req*, *res*) *=>* {});

*// post a TODO*

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*

router.*post*("/all", async (*req*, *res*) *=>* {

*await* Todo.*insertMany*(req.body, (*err*) *=>* {

    if (err) {

      res.*status*(500).*json*({ error: "There was a server side error." });

    } else {

      res.*status*(200).*json*({ message: "Todos were inserted successfully" });

    }

  });

});

*// put TODO*

router.*put*("/:id", async (*req*, *res*) *=>* {});

*// delete TODO*

router.*delete*("/:id", async (*req*, *res*) *=>* {});

module.exports = router;

ekhon amra dekhbo kivabe amra ekta todo update korte pari. Router.put() er moddhe kaj korbo. Ekhane route er parameter hishebe id pacchi. mongoCompass e giye dekhlam prottekta todo er ekta unique \_id ache ja mongodb nije generate kortese. Ami chacchi postman ba client theke localhost/todo/id keu hit korbe. onno pashe ami server theke bujhbo amake oi particular id ta update korte hobe.

Final codes …

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*());

*// database connection with mongoose*

mongoose

  .*connect*("mongodb://localhost:27017/todos")

  .*then*(() *=>* console.*log*("connection successful"))

  .*catch*((*err*) *=>* console.*log*(err));

*// application routes*

app.*use*("/todo", todoHandler);

*// default error handler*

*function* *errorHandler*(*err*, *req*, *res*, *next*) {

  if (res.headersSent) {

*return* *next*(err);

  } else {

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

  }

}

app.*listen*(3000, () *=>* {

  console.*log*("Listening on port 3000");

});

todoSchema.js …

*const* mongoose = *require*("mongoose");

*const* todoSchema = mongoose.*Schema*({

  title: {

    type: String,

    required: true,

  },

  description: String,

  status: {

    type: String,

    enum: ["active", "inactive"],

  },

  date: {

    type: Date,

    default: Date.*now*(),

  },

});

module.exports = todoSchema;

todoHandler.js …

*const* express = *require*("express");

*const* mongoose = *require*("mongoose");

*const* todoSchema = *require*("../schemas/todoSchema");

*const* router = express.*Router*();

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

*// get all the TODOS*

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

  try {

*const* result = *await* Todo.*find*({ description: "Vanilla JS is awesome" })

      .*select*({ \_id: 0, date: 0 })

      .*exec*();

    res.*status*(200).*json*(result);

  } catch (error) {

    res.*status*(500).*json*(err);

  }

});

*// get a TODO by id*

router.*get*("/:id", async (*req*, *res*) *=>* {});

*// post a TODO*

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*

router.*post*("/all", async (*req*, *res*) *=>* {

*await* Todo.*insertMany*(req.body, (*err*) *=>* {

    if (err) {

      res.*status*(500).*json*({ error: "There was a server side error." });

    } else {

      res.*status*(200).*json*({ message: "Todos were inserted successfully" });

    }

  });

});

*// put TODO*

router.*put*("/", async (*req*, *res*) *=>* {

  try {

*const* updateData = { $set: { title: "haluitti" } };

*const* result = *await* Todo.*updateMany*({}, updateData);

    res.*json*(result);

  } catch (error) {

    res.*status*(500).*json*(error);

  }

});

*// delete TODO*

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

  try {

*await* Todo.*deleteOne*({ \_id: req.params.id });

    res.*json*({ msg: "Todo deleted" });

  } catch (error) {

    res.*json*(err);

  }

});

module.exports = router;