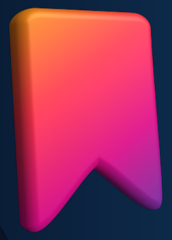
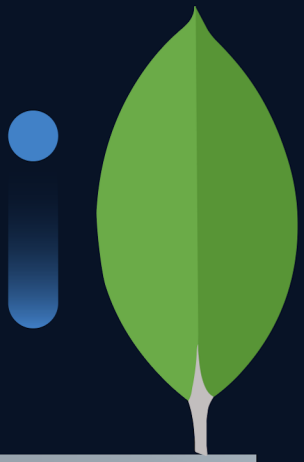
CryptoMinds



Mongoose

By Darshan Vasani
& Krupali Desai

Installation



```
1  npm init -y
2  npm install express mongoose
```

Connect To MongoDB With Localhost

Connect MongoDB

```
// Require Module
const express = require('express');
const app = express();
const mongoose = require('mongoose');

mongoose.connect("mongodb://0.0.0.0:27017/dp",{ useNewUrlParser: true, useUnifiedTopology:
true })
  .then(() => {
    console.log("MongoDB Is Connected");
    app.listen(5000, () => {
      console.log("Server Is Running On Port 5000");
    });
  })
  .catch((error) => {
    console.log("MongoDB Is Not Connected");
    console.error(error);
  });
```

Schema With Validation


```
  //Schema
// Validation Must Be In Schema Only
// Schema Define Structure of Document
const playlistSchema=new mongoose.Schema({
  name:{
    type:String,
    unique:true,
    required:true,
    uppercase:true,
    minlength:[2, "Length Should be more than 2"]  //Custom Error

//     properties: [Object],
//     kind: 'minlength',
//     path: 'name',
//     value: 'A',
//     reason: undefined,
//     [Symbol(mongoose:validatorError)]: true
//   }
// },
// _message: 'Playlist validation failed'
// }


  },
  ctype: {type:String
          // enum:["frontend", "backend", "database"]
          // ctype only frontend , backend , database
  },
  videos:{type:Number,
          validate(value){
            if(value<0){
              throw new Error("Number Must Be Positive");  //User Define Validator -> Custom Validation
              // videos: ValidatorError: Number Must Be Positive
              // properties: [Object],
              // kind: 'user defined',
              // path: 'videos',
              // value: -5,
              // reason: Error: Number Must Be Positive
            }
          }

  },
  author:String,
  email:{
    type: String,
    required:true,
    unique:true,
    // NPM Validator -> User Define Validation -> When validate is used their is always User Define Validation
    validate(value){
      if(!validator.isEmail(value)){
        throw new Error("Email Is Invalid");
      }
    }
  },
  active:Boolean,
  date:{
    type:Date,
    default:Date.now
  }
})
```

Create

```
// Create Document
const Playlist = new mongoose.model("Playlist",playlistSchema);

const createDocument = async () =>{
    //Create Document Or Insert
    try{
      const reactPlaylist = new Playlist({
        name:"darshan",
        ctype: "Front End",
        videos:78,
        author:"CryptoMinds",
        active:true,
      })

      const nodePlaylist = new Playlist({
        name:"rutvika",
        ctype: "node",
        videos:18,
        author:"CryptoMinds",
        active:true,
      })

      const mongPlaylist = new Playlist({
        name:"ankita",
        ctype: "mongo",
        videos:50,
        author:"CryptoMinds",
        active:true,
      })

      const reduxPlaylist = new Playlist({
        name:"krupali",
        ctype: "redux",
        videos:28,
        author:"CryptoMinds",
        active:true,
      })

      const nextPlaylist = new Playlist({
        name:"harshil",
        ctype: "next",
        videos:38,
        author:"CryptoMinds",
        active:true,
      })
      // Always use try and catch with async function

      // const result = await reactPlaylist.save(); For One Document
      const result = await Playlist.insertMany([nextPlaylist,reduxPlaylist,mongPlaylist,nodePlaylist,reactPlaylist]);
      console.log(result);
    }catch(error){
      console.log(error);
    }

}
//Promise <<< Async  [Better]
//Here I also done with Promise But it's too Difficult
createDocument();
```

Update

```
// Update
//Directly From MongoDB Compass
// Through Mongoose
const updateDocument = async (_id) => {
  try {
    // Update the document
    // const result = await Playlist.updateOne({ _id }, { => Show Updated Document Count
    const result = await Playlist.findByIdAndUpdate({ _id }, { // Show Updated Document
      $set: {
        name: "Dpvasani"
      }
    });

    console.log(result);
  } catch (err) {
    console.log(err);
  }
};

// Call the function with a valid _id
updateDocument("647d826e4e60d7f149677a19");
```

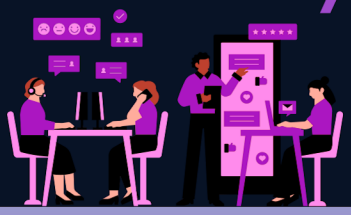
Delete

```
// Delete Document
const deleteDocument = async (_id) => {
  try {
    //const result = await Playlist.findByIdAndUpdate({ _id }); => Show Deleted Document
      const result = await Playlist.deleteOne({ _id });  //=>{ acknowledged: true, deletedCount: 0 } =>Count

    console.log(result);
  } catch (err) {
    console.log(err);
  }
};

// Call the function with a valid _id
deleteDocument("647d826e4e60d7f149677a19");
```

Read

```
        //    Read
const getDocument = async () =>{
    const result = await Playlist.find();
    console.log(result);
}


// Specific type
const getDocument = async () =>{
  const result = await Playlist
  .find({name:"darshan"})
  .select({name:1}).
  limit(1);
getDocument();
```

Comparison Query Operator

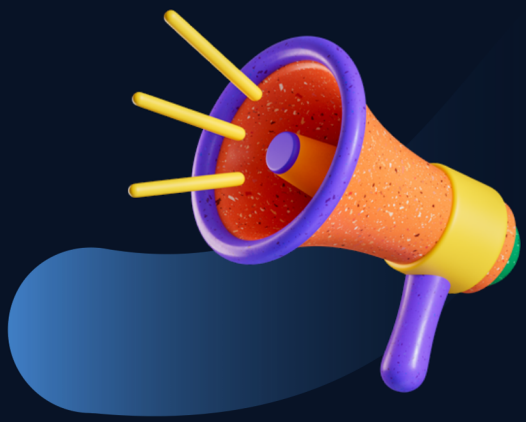


```
//Comparison Query Operator
 //Greater Than 50;

  const getDocument = async () =>{
    const result = await Playlist
    .find({videos:{$gt:50}});


//ctype:"Front End", "mongo"
  const getDocument = async () =>{
       const result = await Playlist
       .find({ctype:{$in:["Front End", "mongo"]}});



  //Logical Operator With Counting
  const getDocument = async () =>{
        const result = await Playlist
          .find({ $or: [ {ctype:"mongo"}, { author:"CryptoMinds"}] })
          .count();


    //Sorting
  const getDocument = async () => {
    const result = await Playlist
      .find()
      .sort({name:1});
    console.log(result);
```

Official Documentation

mongoDB https://www.mongodb.com/docs/manual/

Mongoose https://mongoosejs.com/docs/guides.html



Meet The Team

Darshan Vasani

Cofounder & CEO

darshan-vasani-3299ba245/

dp_vasani56

https://dpvasani-57.netlify.app/

Krupali Desai

Cofounder & CEO

krupali-desai-17269a235

k_rdesai03

https://dkrupali-56.netlify.app

Code Availaible Here
dpvasani/MongoDB Tutorial