# DB, mongoose.

Date:
Page 67

## Database, Schema & Models | Mongoose

first lets create config folder
- Here we will put all the config files.

install mongoose via npm
- npm i mongoose

lets use this in our database

```
1. const mongoose = require ("mongoose");
   (mongoose.connect ("Key Database key string"));
```

↑ this is not a good way use this with async.

```
2  const connectDB = async() => {
3      await mongoose.connect ("key string");
4  };                    ↑ it returns a promise
5  export this function.
6  connectDB().then(() => {
7    console.log ("data base connection established..");
8    }) ↑ add here app.listen here.   VI
9    .catch((err) => {
10     console.error("database connection getting")
11   });
```

↑ This is config file.  do it on app.js page.

(VI) Always first connect to the database and then listen to the server

(1) (2) once connect DB is successful then do app.listen.

(VI) Always connect DB then app.listen

Read mongoose documentation.

Date
Page 68

Let's create our first schema

=>

what do you mean by schema of the database ?

→ ~~whenever we create a database~~

→ Schema is basically a identity for that collection of that document.

We will create a schema using mongoose.

For that we will create another folder.

- models.

```
Const userSchema = mongoose.Schema({
    firstName : {
        type: String
    },
    lastName : {
        type: String
    },
    emailId : {
        type: String
    },
    Password : {
        type : String
    },
    age : {
        type: number
    },
    gender : {
        type : String
    }
});
```

This Schema basically tells us what all information about the user are we storing into a database.
what this userSchema can store or hold.

once we created a schema now we will create a mongoose model.

```
mongoose.model("user", userSchema)

Const UserModel = mongoose.model("user", userSchema);

Module.exports = UserModel.
```

or

```
Module.exports = mongoose.model("user", userSchema);
```

All done
Lets create Sign up API (Post API)

```
app.post("/signup", async (req, res) => {
  Const user = new User({
      firstName: "Praveen",
      LastName: "yadav",
      emailId: "ypraveen760@gmail.com",
      Password: "Praveen123@
  });
  await user.save();
      res.send("User added Successfully!");
});
```

Diffrent between javascript and json ob

Date
Page 70

# API

Till now we have hard coded the data, but this is now how it's works, we have to take this from user for this we will use and create API

Instead of passing data directly or manually ~~for~~ we want to pass data through API while making a post call and api should read the data and push it to the database

In postman : Top portion is - request
bottom portion is - response.

To top portion go to Body

there are diffrent ways to send data inside a api, eg: form-data, raw-data, binary etc

We want to keep our api very simple, we will send json data to our server and server should read that data and then push it into the database

Send data → Server → Push database.

Body
↓raw
↓Json

xml was used widely before json come into picture.

Pass json object.

```
{
  "firstname": "Praveen",
  "lastname": "Yadav",
  "Password": "1234"
}
```

Diffrent between JS object and JSON.

key or major.

| JSON ob | Javascript object |
|---|---|
| we pass data in string<br>eg "firstName": "Praveen"<br>Both are String ✓ | key value pair<br>eg firstName: "Praveen",<br>only this is string. |
| but at end it will throw error. | at end it can have comma |

Now we want to send this data and we have to revive the data, req.body

How will we do that : req.body

(data is send in chunks).

[but it will show undefined] because it is in the json format, & for this we use middleware which express has given us

Req. is send in json format and our server is not able to read json data to read that data we need a middleware.

We need middleware for all our API that convert jsonObject into readable format.

~~middle~~ middleware that is given by express.

express.json([options]) ← we can use directly.

app.use(express.json()); ← this middleware will apply for all & the routes.

if we give it a rout or method then it will work for only that.

app.use(express.json());

↑ everything started.

Get API:

Lets make the get API and it this's get API we want to get all the user from the database.

Feed API - Get /feed - get all the data from database.

~~app.get~~ How to get data from database.

Hard

whenever you want to fetch the data from database you should know which model you have to use what you are getting from database.

User.find( ) {emailId: 'user email'}

↑ model name. ↑ filter here

if we pass {} filter then it will send all the user.

app.get("/user", async (req, res) => {
const userEmail = req.body.emailId;
try {
~~User.~~ const userData = user.find({emailId: userEmail});
~~res.~~ if (!user) {
res.status(404).send("user not found");
} else {
res.send(userData)

To delete use deleteone() or findindoc everything is same or refer Github.

* Lets update our user database.

```
Model.findByIdAndUpdate()
```

↳ It have 3 parameters

```
(id, [update], [options])
```

```
if (!mongoose.Types.ObjectId.isValid(userId)){
    return res.status(500).send("Error No valid id");
}
```