Episode -7

Diving into the APIs

1) How to add a dynamic data in the database through API?

→ usually the data comes from frontend through API, we validate it and store it in DB & send res back to the client.

→ we can also send the data from postman for testing purposes.

2) How to pass data from postman for testing API?

→ Create a POST request with the API endpoint URL

→ Select the Body tab

→ Select the appropriate format for sending data. we choose raw format as we need to send json data.

→ select the format type to JSON

→ After entering the data in the body click on send

→ Postman will send the POST request with the specified body data.

3) How to receive the data passed in the body and store it in DB?

→ the data we sent is present in request body and can access it using `req.body`

→ As we are sending JSON data first we need to convert it to JS object before storing it to DB otherwise the req.body will be undefined.

→ To convert the JSON data to JS object we use a built in express middleware `express.json()`

→ This middleware has to be on top after imports as all the data

we recieve should convert just before storing it in DB.

→ store the data recieved in the place of hardcoded data.

eg:
```
app.use(express.json())

app.post("/signup", async (req, res) => {
    const user = new User(req.body)
    try{
        await user.save()
        res.send("user added successfully")
    }catch(err){
        res.status(400).send("Error saving the user")
    }
})
```

4) what is the difference between Js object and JSON?

JavaScript object	JSON
It is a data structure in JS used to store collection of data as key-value pairs	It is data format used for transferring data between servers and web applications.
Keys don't have to be in quotes	Keys should always be in quotes
It can hold any data types like strings, numbers, array, objects, functions, undefined and null	Can hold only certain data types cannot hold function or undefined.
eg: const user = { name: "Shan", age: 50, }	eg: const user = { "name": "Shan", "age": 50 }

5) How to get a data from a db?

→ 1st step: create an get api

→ 2nd step: query the model which contains that data either by using find(), findById(), findOne() depending upon the use case.

eg: finding one document from a collection.

```
app.get("/user", async (req, res) => {
    const userEmail = req.body.email

    try {
        const user = await User.find({email : userEmail})
                                  ↓
                              Model.find
                              Name
        res.send(user)
    } catch(err) {
        res.status(400).send("Something went wrong")
    }
})
```

→ 3rd step: send the email Id from postman in body section.

Note: If there are multiple user with the same email Id find method will returns all that users in an array.

→ If the user is not found it gives an empty array so we need to check the length of the result before sending response.

eg:

```
if (user.length === 0) {
    res.status(401).send("user Not found")
} else {
    res.send(user)
}
```

6) How to get all the document from a particular collection?

→ using Model.find({})

eg:

```
app.get("/allusers", async (req, res) => {
    try {
        const users = await User.find({})
        res.send(users)
    } catch (err) {
        res.status(500).send("internal server error")
    }
})
```

Note:

→ Since we are retrieving all the documents we no need to pass anything in the body.

→ It returns all the document in an array.

7) what is difference between find({query}) & findone()?

→ find method with query:

returns all the document that matches the query in array.

→ findOne():

returns the first document which matches the query.

↳ It returns "null" if no document matches the query.

8) How to delete a document from collection?

→ There are various ways of deleting a document from the collection such as findByIdAndDelete(), findOneAndDelete()

eg:

```
app.delete("/user", asyn (req, res) => {
    const userId = req.body.userId.
    try {
        await User.findByIdAndDelete(userId)
        res.send("user deleted successfully")
    }
```

```
catch (err) {
    res.status(400).send("Something went wrong")
})
```

9) How to update a document in the collection?

→ There are 2 ways of doing it either by using findByIdAND Update() (or) findOneAndUpdate()

→ findByIdAndUpdate() method accept 3 values one is Id, 2nd is the updated data, 3rd is options (optional).

eg:

```
app.patch("/user", async (req, res) => {
    const userId = req.body.userId
    const data = req.body
    try {
        const user = await User.findByIdandUpdate(
                        {_id: userId}, data)
        res.send("user updated successfully")
    }
    catch (err) {
        res.status(400).send("Something went wrong")
    }
})
```
