Week - 3

⇒ 3.$\underset{2}{1}$:

- Middleware, global catches and zod.

→ <u>Middleware</u> is used for dry.
It is one kind of function which will do checkups and then give flow to main function.

→ next(); → it will handle thread and execute next function

→ app.use (middlewarefn): every function after this line will call this middlewarefunction.

⇒ Why we need Input Validation?
→ user can request anything so if user send something crazy server will throw error

⇒ Global catches: it will not show errors to user. It is one type of middleware

Syn: app.use (function (err, req, res, next) {
 res.json({ msg : "error" });
})

⇒ authentication :-

ZOD :- it is input validation library.

- Single variable input validation :-

```
input = z.array(z.number());
```

*Don't forget () after any type

- Object Validation :-

```
schema = z.object({
        email : z.string().email(),
        pass : z.String(),
        country : z.literal("IN").or(z.li(ca))
});
```

→ How to validate?

```
input.safeParse(variable);
```

→ 3.2 - Fetch, Authentication and DB

→ fetch req by default is Get.
- it is Async function.
- response.json() is also Async function.

→ Authentication :-

- Type of Authentication :-

1) Hashing
2) Encryption
3) Json web tokens
4) Local storage.

1) Hashing is 1 way encryption

2) Encryption is 2 way which use key

3) Json web token : it is Json encryption
It is one type of digital signature.

4) Local Storage - store JWT token at browser.

-> Tokens will be always in headers

-> always add Bearer before token
(Got error and took 13min LoL)

⇒ Databases :-

1) VectorDB
2) GraphDB
3) SQLDB
4) NoSQLDB

⇒ JWT - JSON Web token

1) Generate
2) Decode
3) Verify

-> Dom :- Document object model
↳ platform independent

window.onload () => { }

↑ run when document is loaded.

```
.getElementByID ("ID");
.innerHTML = "I am" ;
.elements[0] / .elements
.createElement ("p");
.innerText = "I am";
.getElementByTagName("LOL");
.getElement ByClassName ("BG");
.querySelectorAll ("p");
.setAttribute ("class", "IDK");
                 type , btn
```

⇒ document.

```
.createElement (     );
.removeChild (     );
.appendChild (     );
.replaceChild (     );
.write (text);
```

⇒ Node.

```
.childNodes              .nodeValue
.firstChild              ELEMENT_NODE
.lastChild               ATTRIBUTE_NODE
.parentNode
.nextSibling
.previousSibling
```