

1) Node.js is used in backend development

⇒ It is a javascript runtime built on chrome V8 engine that allows you to run javascript outside the browser

Used in backend development:

⇒ Handles many users using non-blocking I/O  
⇒ Fast and light weight  
⇒ Same language for front and backend

2)

Synchronous

Execute line by line

Slow down process

Block next code  
until current task  
finishes

Non-blocking

Executes in background

uses callbacks, promise  
on async await

3)

Properties of FS Module

⇒ Read files

⇒ Write files

⇒ Delete files

⇒ Create folder

Two methods

fs.readfile()

fs.writefile()

4)

Relative Path vs Absolute Path

Based on current  
location

Ex: ./data/file.txt

Full path from directory

Example: c/users/  
project/data/file.txt

### 5) Express :

=> Minimal web framework for Node.js  
=> Used to build APIs and web applications

easily

Node HTTP

Basic & manual routing

None code

No built-in middleware

Complex handling

Express

Easy routing

less code

Middleware Support

Simple & Structured

### 6) Routing in Express :

=> How the server responds to client

requests:

Example :-

```
app.get('/home', (req, res) => {  
    res.send("Welcome Home");  
});
```

app.get() =>

HTTP method

Home =>

Route path

Callback =>

function to handle request

### 7)

req.params

URL parameters

/user/10

req.query

Query string

/search?name=John

req.body

Data sent in  
post request

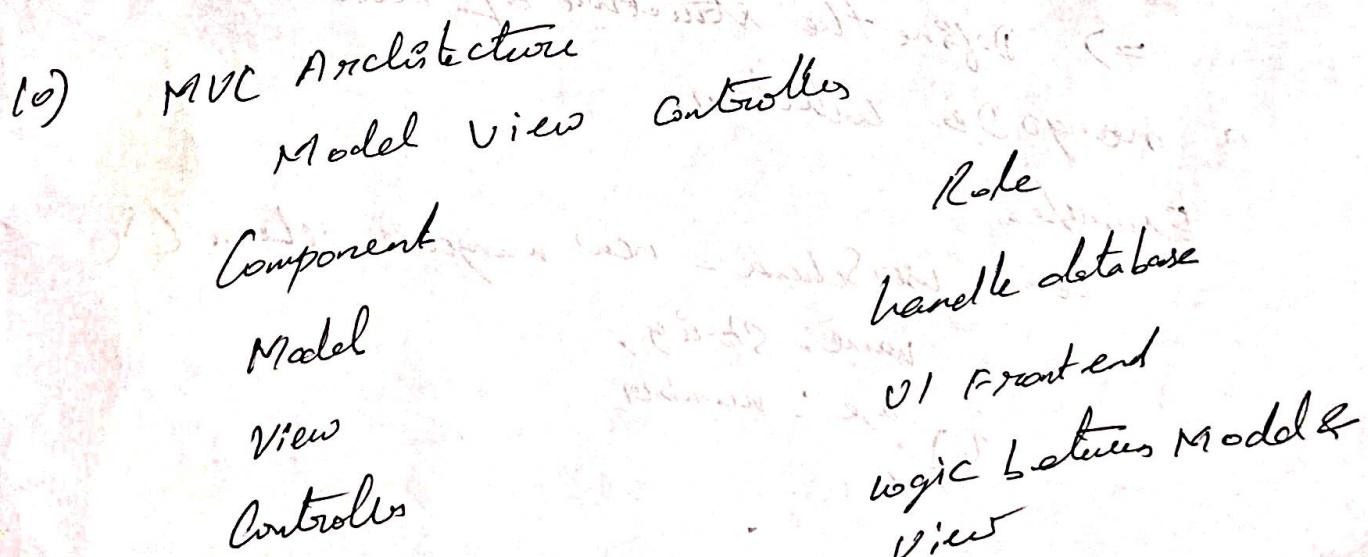
Formdata/JSON

- 8) Response object:
- ⇒ res is the object used to send data back to the client
- Methods:
- res.send()
  - res.json()
  - res.status()
- Without res, the client will not receive any response.

- 9) Use of http codes:
- ⇒ Tell the client whether a request was successful or failed.

Common statusCode:

- 200 → ok
- 404 → Not found
- 500 → Server Error



- Use MVC:
- ⇒ Clean code structure
  - ⇒ Easy maintenance.

## 11) MongoDB:

⇒ No SQL database that stores data in JSON like documents.

Difference:

SQL

Tables

Rows

Structured schema

Joins

MongoDB

Collections

Documents

Flexible schema

Embedded documents

## 12) Mongoose:

⇒ Object Data Modelling library for mongoDB in Node.js.

Use it:

Schema validation

Cleaner queries

Middleware support

Model based structure

ORM

## 13)

Schema in Mongoose:

⇒ Define the structure of documents inside a mongoDB collection.

Example:

```
const userSchema = new mongoose.Schema({
```

name: String,

age: Number

```
});
```

14) Model in Mongoose:

=> created from a schema used to interact with the database.

Example:

```
const user = mongoose.model("User", userSchema);
```

15) Mongoose methods for CRUD:

Create:      → Create()  
                → Save()

Read:         find()  
                findById()  
                findOne()

Update:         findAndReplace()  
                updateOne()

Delete:         findAndDelete()  
                deleteOne()