1) Node.js and 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	Asynchronous
Execute line by line	Non-blocking
Slow down Servers	Executes in background
Block next code until current task finishes	uses callback, promise or async await

3) Purpose 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 root 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	Express
Basic & manual routing	Easy routing
More code	less code
No built in middleware	Middleware support
Complex handling	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 methods

/home => Route path

Callback => function to handle request

7)

req.params	req.query	req.body
Url parameters	Query string	Data sent in post request
/user/10	/search?name=John	Form data / 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 successfull or failed.

Common statusCode:

200 -> Ok
404 -> Not found
500 -> Server Error

10) MVC Architecture

Model View Controller

Component	Role
Model	Handle database
View	UI Frontend
Controller	logic between Model & View

Use MVC:

=> Clean code Structure
=> Easy maintenance.

11) MongoDB:

=> No Sql database that store data in json like documents.

Difference:

SQL	MongoDB
Tables	Collections
Rows	Documents
Structured schema	Flexible schema
Joins	Embedded documents.

12) Mongoose:

=> Object Data Modelling library for mongoDB in Node.js.

Use it:

Schema validation
Cleaner queries
Middleware support
Model based structure

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("Users", userSchema);
```

15) Mongoose methods for CRUD:

Create: -> Create()
-> Save()

Read: find()
findById()
findOne()

Update: findByIdAndUpdate()
UpdateOne()

Delete: findByIdAndDelete()
deleteOne()