Notes

Interaction with system

There are two methods by which we can interact with the system/machine/computer:

- **GUI ⇒ Graphical User Interface**

  > 💡 This is the pretty common way that we use to interact with the system, that includes taking the help of graphics to demonstrate and visualise all the actions that we are doing.

- **CLI ⇒ Command Line Interface**

  > 💡 This kind of interaction mainly involves writing specific commands to perform any kind of operation in the system.

Common CLI commands

Please explore the following CLI commands, they are really fun:

1.	ls	Directory listing
2.	ls -al	Formatted listing with hidden files
3.	ls -lt	Sorting the Formatted listing by time modification
4.	cd dir	Change directory to dir
5.	cd	Change to home directory
6.	pwd	Show current working directory
7.	mkdir dir	Creating a directory dir
8.	cat >file	Places the standard input into the file
9.	more file	Output the contents of the file
10.	head file	Output the first 10 lines of the file
11.	tail file	Output the last 10 lines of the file
12.	tail -f file	Output the contents of file as it grows,starting with the last 10 lines
13.	touch file	Create or update file
14.	rm file	Deleting the file
15.	rm -r dir	Deleting the directory
16.	rm -f file	Force to remove the file
17.	rm -rf dir	Force to remove the directory dir
18.	cp file1 file2	Copy the contents of file1 to file2
19.	cp -r dir1 dir2	Copy dir1 to dir2;create dir2 if not present
20.	mv file1 file2	Rename or move file1 to file2,if file2 is an existing directory

External Node Modules

These are also called as third party libraries, as they are written by someone else and we can directly use them in our project.

- Install `is-even` package using `npm` .
- Play around with it as we did in the session to understand the concept.
- You can install any other package as well and plat around with it.

Start with the project

> 💡 Do all the following things using `CLI` only.

- Create a `node` project.
- Create a `index.js` file.
- Create a `text.txt` file.
- Write some content inside the file.
- Use node's inbuilt module `fs` to read the content of file and print them to the console.
  - `readFile` :- This will read the file asynchronously.

```
const fs=require("fs")

fs.readFile("./text.txt",{encoding:"utf-8"},(err,data)=>{
    if(err){
        console.log("Cannot read the file")
        console.log(err)
    } else {
        console.log(data)
    }
})

console.log("Bye Guys!!")
```

  - `readFileSync` :- This will read the file synchronously, until the file reading is not finished, compiler is not going to the **Bye Guys!!** statement.

```
const fs=require("fs")

const data=fs.readFileSync("./text.txt",{encoding:"utf-8"})

console.log(data)
console.log("Bye Guys!!")
```

- Write into a file using the `fs` module again.
- While writing it will automatically create the file.
  - `writeFile` :- This will write the file asynchronously.

```
const fs=require("fs")

fs.writeFile("./log.txt", "This is me first time wrinting in the file", (err)=>{
    if(err){
        console.log("Cannot write in the file")
        console.log(err)
    } else {
        console.log("Data has been written in the file")
    }
})
```

  - `writeFileSync` :- This will write the file synchronously, first the writing will be finished then the next task.

```
const fs=require("fs")

fs.writeFileSync("./log.txt", "This is me second time wrinting in the file")
```

- `writeFile` will overwrite the previous content of the file.

- So try `appendFile` to overcome this.

```
const fs=require("fs")

fs.appendFile("./log.txt", "\nThis is me third time wrinting in the file\n",(err)=>{
    if(err){
        console.log("Cannot be appended")
        console.log(err)
    } else {
        console.log("Data has been appended in the file")
    }
})
```

- There is `appendFileSync` as well, it works in the same as `readFileSync` and `writeFileSync`.

> 💡 **Important Note:**
>
> 1. Research how to take arguments from terminal.
> 2. Explore https://nodejs.dev/en/learn/ for easier understanding about the node documentation, `cannot be used in evaluations`.