

Date: 12 June 2024

Day: Wednesday

Day 6 of the internship focused on mastering the creation and utilization of custom modules in Node.js. Custom modules are essential for modularizing code, promoting reusability, and enhancing the organization of complex applications.

Learning Objectives:

- **Custom Module Creation:**
 - Explored the concept of custom modules in Node.js, which allows encapsulating functionality into reusable components.
 - Learned how to create custom modules using `module.exports` to export functions, objects, or classes from one module to another.
 - Implemented modular design principles to break down application logic into manageable and reusable components.

```
CustomNodejs > JS LoggerUsed.js > ...
1  const logger = require('./Logger');
2  const {config} = require('./Logger') // called as import of function
3  logger('this is log message from Loggerused file')
4
5  console.log('log file name:', logger.config.logFileName)
6  console.log('log file Dir:',config.logDirectory)
```

- **Application of Custom Modules:**

- Applied custom modules to refactor existing code, improving code organization and maintainability.
- Utilized custom modules to encapsulate related functionalities, promoting separation of concerns and enhancing code readability.
- Integrated custom modules into larger applications to demonstrate their role in building scalable and modular software architectures.

```
CustomNodejs > JS Logger.js > ...
1  const fs = require('fs')
2
3  const logMessage = (message) => {
4    fs.appendFile('app.log', message + "\n", (error)=>{
5      if (error){
6        console.error('error writing to log file', error, "", "\n")
7      }
8      else{
9        console.log("Message logged:", message)
10     }
11   })
12 }
13 logMessage('this is log message from logger file')
14
15 module.exports = logMessage
16 // when you have to call from this module other module functions but you cannot rewrite module.
17 // exports after writing same it loses the previous one so we write
18 module.exports.config={
19   logFileName: 'app.js',
20   logDirectory: './'
21 };
```

Activities and Insights:

- **Building Modular Components:**

- Created custom modules for specific tasks such as data manipulation, utility functions, and API interactions.
- Encapsulated related functionalities into modules, ensuring each module focused on a single responsibility for better code maintenance and reusability.

- **Exporting and Importing Modules:**

- Experimented with different export styles to suit varying project requirements, including default exports and named exports.
- Explored importing strategies to efficiently include external modules in Node.js applications, ensuring proper dependency management and module resolution.