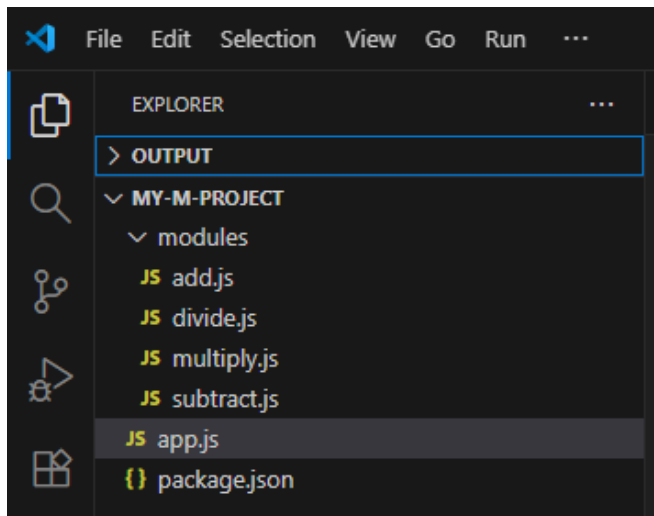



1. Project Overview

This project demonstrates modularization in Node.js — the process of dividing an application into smaller, independent modules (files). Each module performs one specific task, which makes the program easier to understand, maintain, and reuse.

2. Project Structure




3. Code Implementation

-  modules/add.js


```
modules > JS add.js > ...  
1  function add(a,b){  
2      return a+b;  
3  }  
4  module.exports=add;  
5  
6
```

-  modules/subtract.js

```
modules > JS subtract.js > ...  
1  function subtract(a, b) {  
2      return a - b;  
3  }  
4  
5  module.exports = subtract;  
6
```

-  modules/multiply.js

```
modules > JS multiply.js > ...
1 function multiply(a, b) {
2   return a * b;
3 }
4
5 module.exports = multiply;
6
```

-  modules/divide.js

```
modules > JS divide.js > ...
1 function divide(a, b) {
3   return "Cannot divide by zero";
4 }
5   return a / b;
6 }
7
8 module.exports = divide;
9
```

-  app.js

```
JS app.js > ...
1 const add = require('./modules/add.js');
2 const subtract = require('./modules/subtract.js');
3 const multiply = require('./modules/multiply.js');
4 const divide = require('./modules/divide.js');
5
6 const a = 20;
7 const b = 5;
8
9 console.log("Addition:", add(a, b));
10 console.log("Subtraction:", subtract(a, b));
11 console.log("Multiplication:", multiply(a, b));
12 console.log("Division:", divide(a, b));
13
```

4. Running the Application

To run the application:

1. Open the terminal in VS Code.
2. Run: `node app.js`
3. Expected Output:

Addition: 25

Subtraction: 15

Multiplication: 100

Division: 4

5. Concept of Modularization

Modularization is the technique of dividing a large program into smaller, manageable, and independent files called modules. Each module handles a specific part of the functionality. Node.js supports modularization using 'module.exports' to export and 'require()' to import modules.

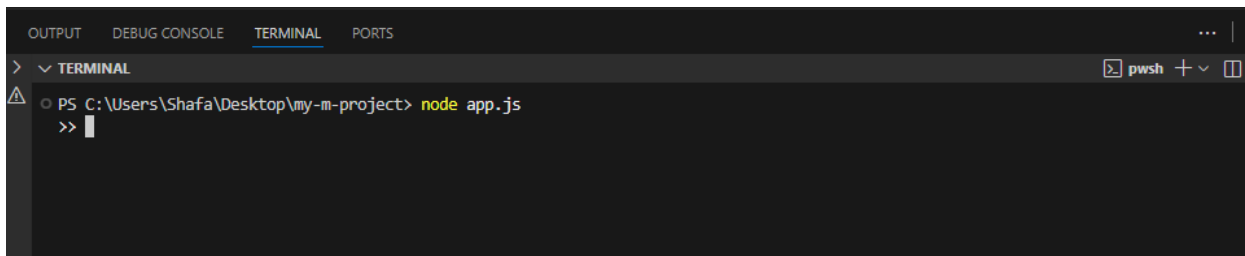
Benefits include easier maintenance, reusability, and collaboration.

6. How the Modules Work Together

Each module (add, subtract, multiply, divide) contains a mathematical function. The `app.js` file imports these modules using `require()`, and executes them to display results in the console.

7. Project Verification

The command:



```
OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
> ▼ TERMINAL
⚠ PS C:\Users\Shafa\Desktop\my-m-project> node app.js
>> |
```

The result:



```
DEBUG CONSOLE  TERMINAL  PORTS
> ▼ TERMINAL
PS C:\Users\Shafa\Desktop\my-m-project> node app.js
Addition: 25
Subtraction: 15
Multiplication: 100
Division: 4
PS C:\Users\Shafa\Desktop\my-m-project> |
```

8. Conclusion

This project successfully demonstrates Node.js modularization by dividing functionalities into separate files and connecting them using the `require` and `module.exports` syntax. It highlights how modular programming improves clarity, reusability, and maintainability.