1. File Handling and Modules

In Node.js, modules are a way to organize code into reusable files. They help in breaking down the code into smaller, manageable pieces. For your task, you can use the built-in `fs` (File System) module to read and write files.

Here's a simple Node.js program that reads a text file, modifies its content, and writes the updated content to a new file:

```javascript
// Importing the ‘fs’ module
Const fs = require(‘fs’);

// Function to read and modify file content
Function modifyFileContent(inputFilePath, outputFilePath) {
  // Reading the content of the input file
  Fs.readFile(inputFilePath, ‘utf8’, (err, data) => {
    If (err) {
      Console.error(‘Error reading file:’, err);
      Return;
    }

    // Modifying the content (for example, converting to uppercase)
    Const modifiedContent = data.toUpperCase();

    // Writing the modified content to the output file
    Fs.writeFile(outputFilePath, modifiedContent, ‘utf8’, (err) => {
      If (err) {
        Console.error(‘Error writing file:’, err);
        Return;
      }

      Console.log(‘File has been modified and saved successfully!’);
    });
  });
}

// Example usage
Const inputFilePath = ‘input.txt’;
Const outputFilePath = ‘output.txt’;

modifyFileContent(inputFilePath, outputFilePath);
```

In this example, the `fs` module is used to read the content of the input file (`input.txt`), modify it (converting to uppercase), and then write the modified content to the output file (`output.txt`).

Modules are essential here for keeping the code organized and modular. The `fs` module provides a set of functions to interact with the file system, encapsulating file-related operations and making the code more readable. Additionally, modularizing code facilitates code reuse and maintenance, making it easier to understand and update in the future.