Homework 1

ECE567: Software Engineering 1

Tanya Sharma
tds104

3 October 2023

Design

The program contains the following features:

- The user will start the program and will be able to see the csv file displayed with a well-maintained table structure
- The user is prompted for an input formula that they can provide
- The data.csv file is currently being used as the file to be displayed. This can be changed by using a different file name in the main function.

Commands

In order to execute the both the program as well as test cases the following instructions can be followed:

1. **npm install**

Install all necessary dependencies

2. **node main.js**

This command will execute the main function where all our logic takes place

3. **jasmine**

This command will execute the test suite

Code Snippets

Main Program: ans.js

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

//Function to check if file exists
function fileExists(filePath){
  try{
    fs.accessSync(filePath,fs.constants.F_OK);
    return true;
  }catch(err){
    return false;
  }
}

function readAndPrintCSV(fileName) {
  // Handle case when file name does not exist
  if (!fileExists(fileName)) {
    console.error(`Error: The file does not exist.`);
    process.exit(1);
  }
  const fileStream = fs.createReadStream(fileName);
  const rl = readline.createInterface({
    input: fileStream,
    crlfDelay: Infinity,
  });
  let header = null;
  let rows = [];
  rl.on("line", (line) => {
    const values = line.split(",");
    // Handle case when the CSV is invalid because it has invalid columns
    if (!header) {
      header = values;
    } else {
      if (values.length !== header.length) {
        console.error("Error: Mismatched columns in the CSV file.");
        rl.close();
        return;
      }
      rows.push(values);
    }
  });
  rl.on("close", () => {
    // Handle an empty CSV
    if (!header) {
      console.error("Error: The CSV file is empty.");
    } else {
      // Calculate column widths based on header and data rows
      const columnWidths = header.map((col, index) => {
        return Math.max(col.length, ...rows.map((row) => row[index].length));
      });
      // Print the header
      console.log(
        header.map((col, index) => pad(col, columnWidths[index])).join(" | "),
      );
      // Print separator line
      console.log(
        columnWidths.map((width) => "-".repeat(width)).join(" | "),
      );
      // Print the data rows
      for (const row of rows) {
        console.log(
          row.map((col, index) => pad(col, 
columnWidths[index])).join(" | "),
        );
      }

      // Wait for user input to enter a formula
      const input = readline.createInterface({
        input: process.stdin,
        output: process.stdout,
      });
      input.question("Enter a formula: ", (formula) => {
        console.log(`You entered: ${formula}`);
        input.close();
      });
    }
  });
}

// Function to pad a string to a specified width with spaces
function pad(str, width) {
  const diff = width - str.length;
  return str + " ".repeat(diff > 0 ? diff : 0);
}

function main() {
  const fileName = "data.csv"; /
  readAndPrintCSV(fileName);
}

main();

module.exports = {
  readAndPrintCSV,
  fileExists
};
```

Test: test.spec.js

```
const {readAndPrintCSV, fileExists} = require('./main');
const fs=require('fs');

describe('Main Function',()=>{

  it('should handle an empty CSV file',(done)=>{
    fs.writeFileSync('empty.csv','');
    const 
consoleErrorSpy=spyOn(console,'error').and.callThrough();
    readAndPrintCSV('empty.csv');
    setTimeout(()=>{
      expect(consoleErrorSpy).toHaveBeenCalledWith('Error: The CSV file is empty.');
      fs.unlinkSync('empty.csv');
      done();
    },1000);
  });

  it('should handle a non-existent CSV file',(done)=>{
    const invalidFilePath='doesnotexist.csv';
    const exists=fileExists(invalidFilePath);
    setTimeout(()=>{
      expect(exists).toBe(false);
      done();
    },1000);
});

it('should handle CSV files with mismatched columns(invalid CSV)',(done)=>{
fs.writeFileSync('invalid.csv','Header1,Header2\nValue1\n');
  const consoleErrorSpy=spyOn(console,'error').and.callThrough();
  readAndPrintCSV('invalid.csv');
  setTimeout(()=>{
    expect(consoleErrorSpy).toHaveBeenCalledWith('Error: Mismatched columns in the CSV file.');
    fs.unlinkSync('invalid.csv');
    done();
  },1000);
});

});
```

Conclusion

This program accepts a csv file and displays it in plain text in the console using node.js. It also takes a formula from the user as input. In Homework 2 we will tackle the next set of problems associated with processing the formula.