

**INSTITUTE OF PRINTING TECHNOLOGY  
&  
GOVERNMENT POLYTECHNIC COLLEGE  
SHORANUR**



**Department Of Computer Engineering**

**2021-2022**

**PROJECT REPORT ON**

**DAYLIGHT-TORRENT DOWNLOADER**

Submitted by:

**NITHIN P T**(Reg No. : 19138122)

**PRINITTA C JAISON** (Reg No. : 19138123)

**VISHNU T M**(Reg No. : 19138126)

**ADITHYAN P A**(Reg No. : 19138117)

**RANJITH K**(Reg No. : 19138124)

**GUIDED BY: ABDUL KHADER**

**INSTITUTE OF PRINTING TECHNOLOGY &  
GOVERNMENT POLYTECHNIC COLLEGE  
SHORANUR, KERALA**

**CERTIFICATE**

This is to certify that the project titled **DAYLIGHT-TORRENT DOWNLOADER** that has been presented by **NITHIN P T(Reg. No. : 19138122), RANJITH K (Reg. No. : 19138124), VISHNU T M(Reg. No. : 19138126), ADITHYAN P A(Reg. No. : 1918117), PRINITTA C JAISON(Reg. No. :19138123)**, final year students of COMPUTER ENGINEERING, in partial fulfillment of the requirements for the award of the DIPLOMA IN COMPUTER ENGINEERING under the Directorate of the Technical Education, Kerala, during the year 2021-2022.

**Guided By**

**Head of Department**

**Internal Examiner**

**External Examiner**

**Place: SHORANUR**

**Date:**

## **ACKNOWLEDGEMENT**

First of all I thank the Almighty for showering his infinite blessing throughout my life. I extend my sincere thanks especially to Mr. **Abdul Khader**, Head of Department for his co- operation and full support .

I would like to express my sincere and profound gratitude to the staff in charge **Mrs. Radhika** and all other supporting staff for remaining a source of inspiration and encouragement throughout to enable me to succeed in the endeavour .

**NITHIN P T(Reg. No. : 19138122)**

**PRINITTA C JAISON (Reg. No. : 19138123)**

**VISHNU T M(Reg. No. : 19138126)**

**ADITHYAN P A(Reg. No. : 19138117)**

**RANJITH K (Reg. No. :19138124)**

**COMPUTER ENGINEERING [2021- 2022]**

# **CONTENTS**

1. INTRODUCTION
2. SYSTEM DESIGNING
3. DATA FLOW DIAGRAM
4. CODING
5. TECHNOLOGY FRAME WORKS
6. IMPLEMENTATION
7. SCOPE OF PROJECT
8. CONCLUSION
9. REFERENCE

## **INTRODUCTION**

The Torrent downloader can be used to reduce the server and network impact of distributing large files. Rather than downloading a file from a single source server, the DayLight allows users to join a "swarm" of hosts to upload and download from each other simultaneously. The DayLight client enables a user to search for and download torrent files using a built-in search box in the main window, which instantly downloads the file to the user's computer.

DayLight client can be used to download extremely large torrent files by entering torrent file link. Large torrent files include movies, games, additional files for games etc requires a torrent client to download. Daylight torrent downloader is built using HTML and CSS as front-end and Node JS as backend.





## **SYSTEM DESIGNING**

### **HARDWARE REQUIRMENTS :**

Process - Intel Core i5

Hard Disk -256GB(SSD Recommanded)

RAM - 4GB

### **SOFTWARE REQUIRMENTS :**

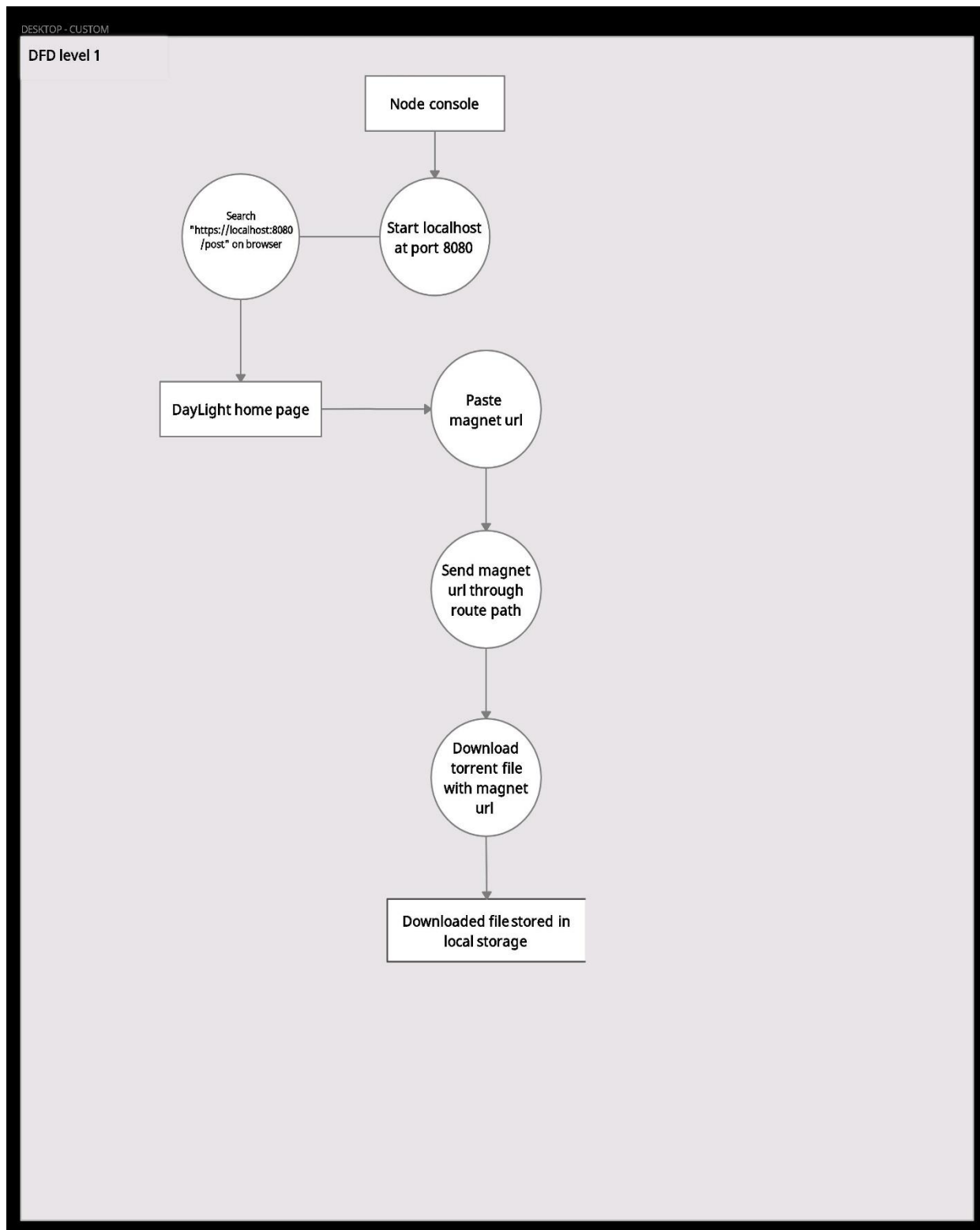
Operating system - windows

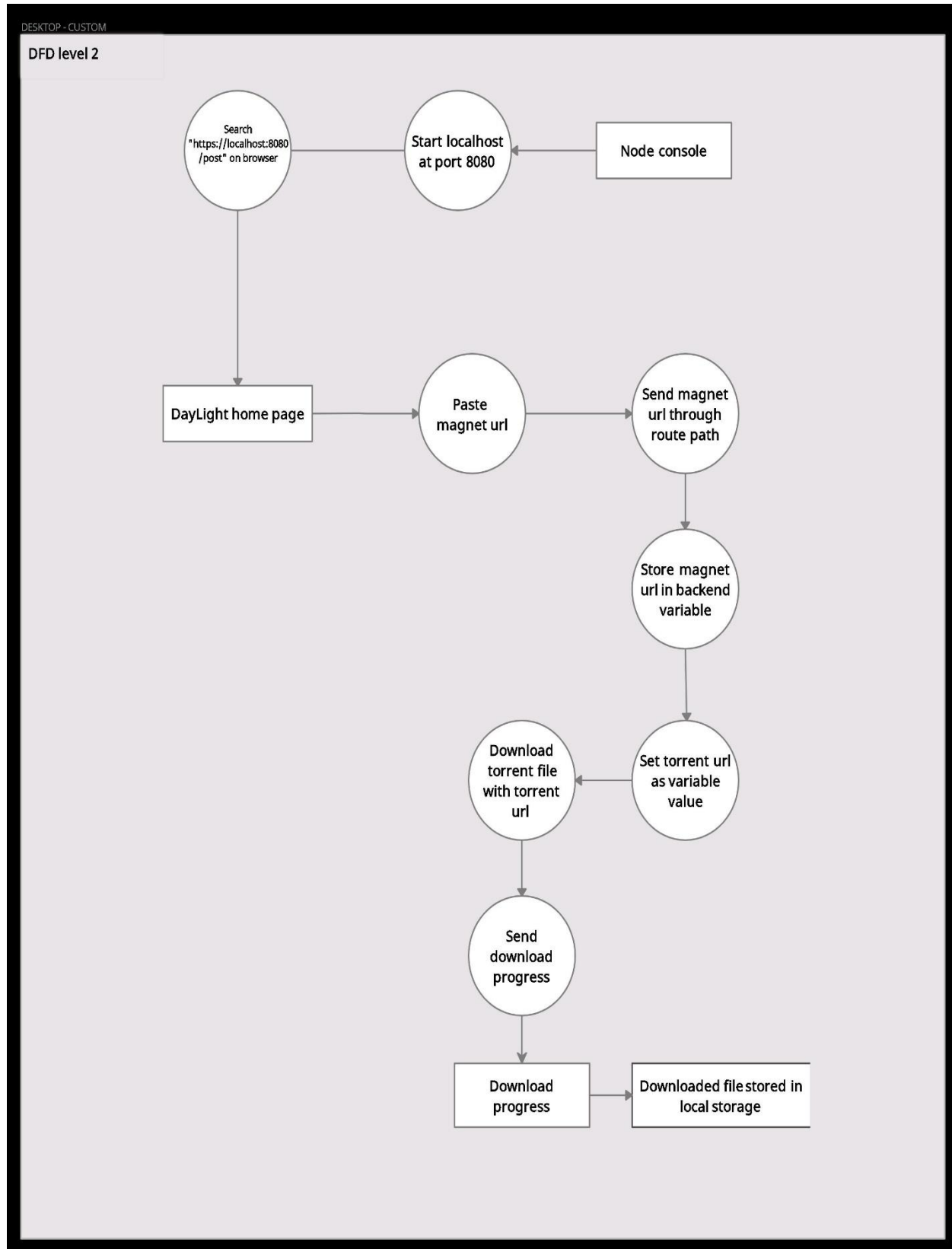
Software Tools - Node js,command  
Prompt, GitHub,  
Html

Programming language - Java script



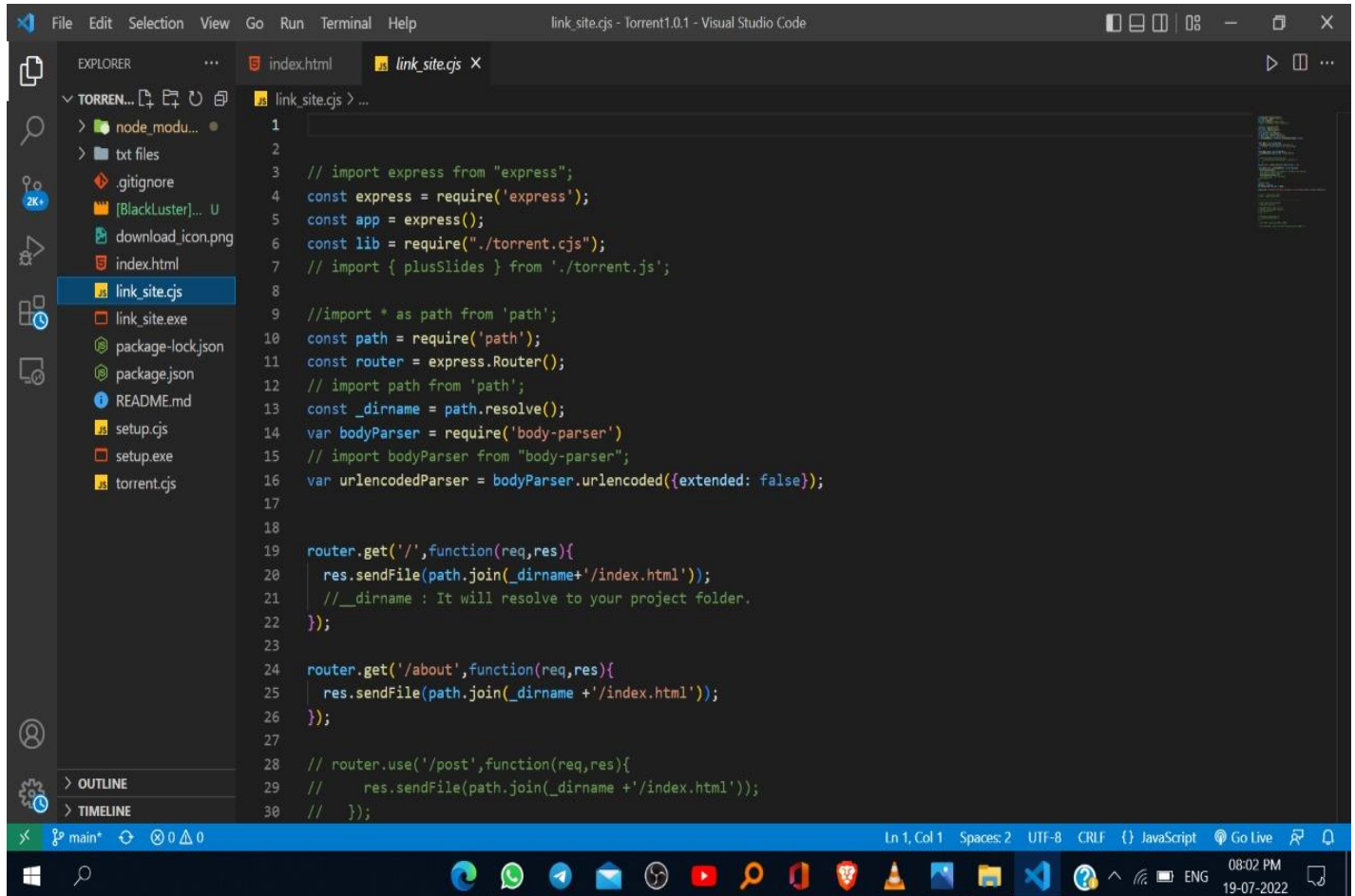
# DATA FLOW DIAGRAM



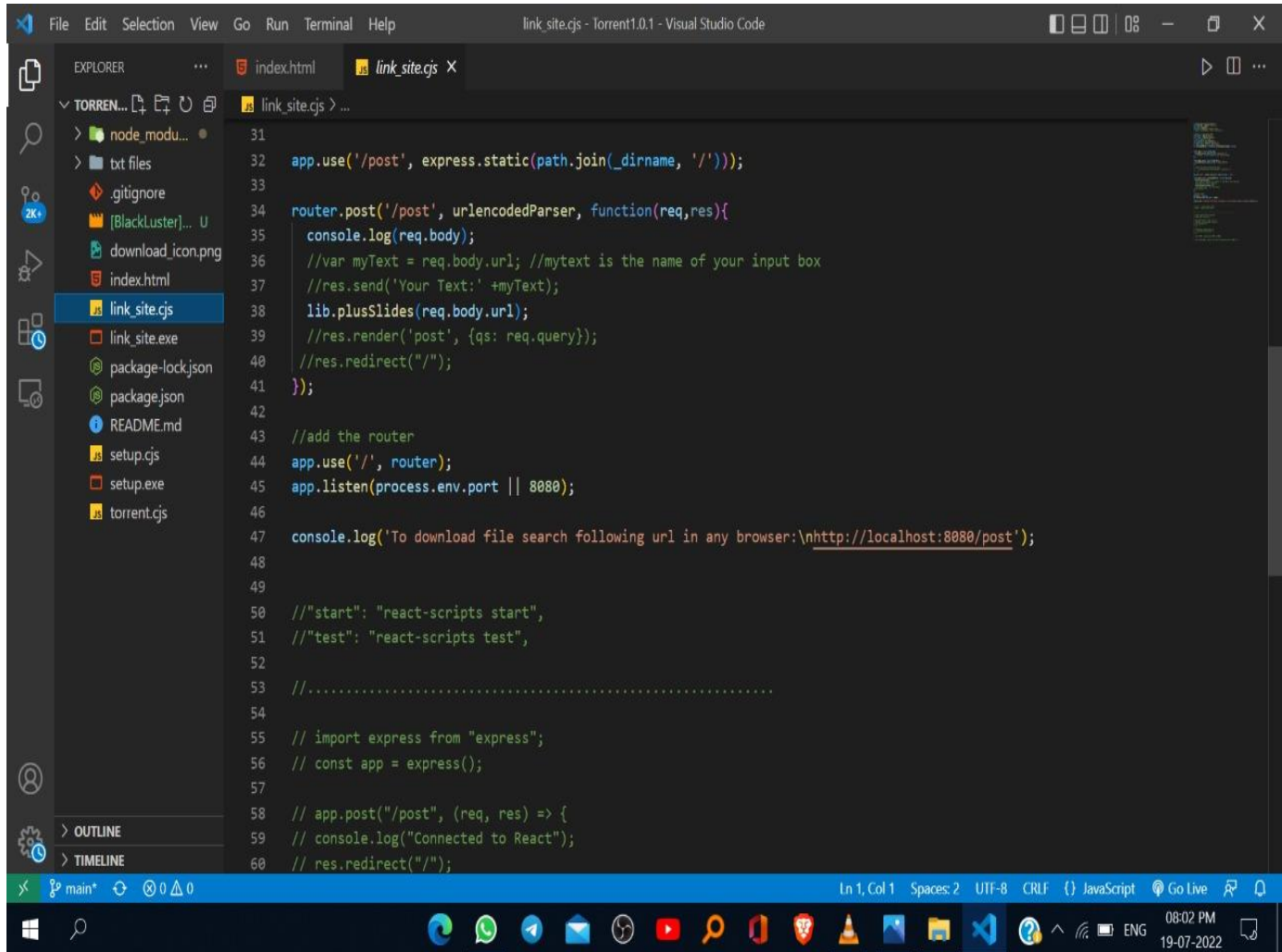


# CODING

## Link\_site.cjs

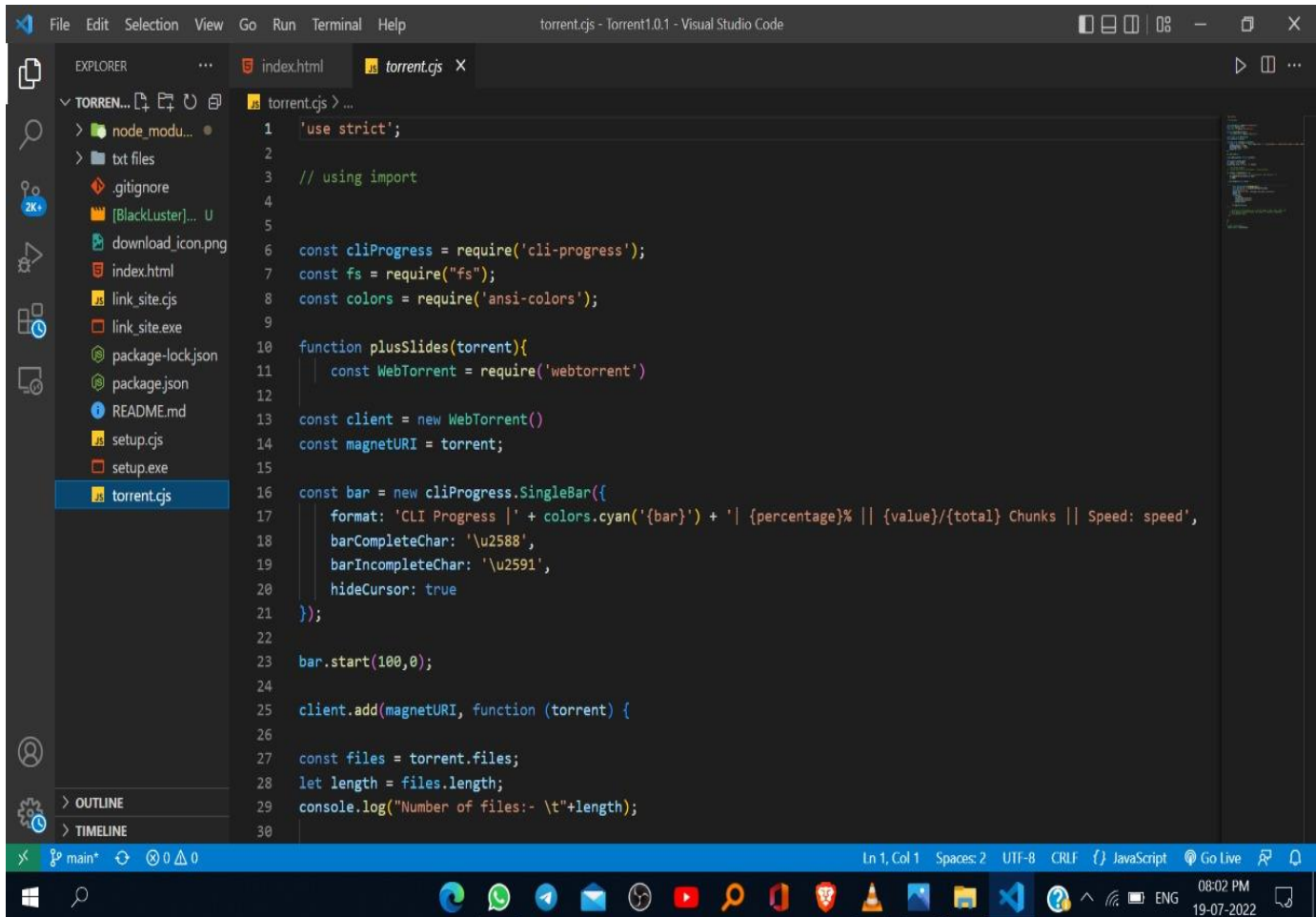


```
1
2
3 // import express from "express";
4 const express = require('express');
5 const app = express();
6 const lib = require("./torrent.cjs");
7 // import { plusSlides } from './torrent.js';
8
9 //import * as path from 'path';
10 const path = require('path');
11 const router = express.Router();
12 // import path from 'path';
13 const _dirname = path.resolve();
14 var bodyParser = require('body-parser')
15 // import bodyParser from "body-parser";
16 var urlencodedParser = bodyParser.urlencoded({extended: false});
17
18
19 router.get('/',function(req,res){
20   res.sendFile(path.join(_dirname+'/index.html'));
21   //__dirname : It will resolve to your project folder.
22 });
23
24 router.get('/about',function(req,res){
25   res.sendFile(path.join(_dirname + '/index.html'));
26 });
27
28 // router.use('/post',function(req,res){
29 //   res.sendFile(path.join(_dirname + '/index.html'));
30 // });
```

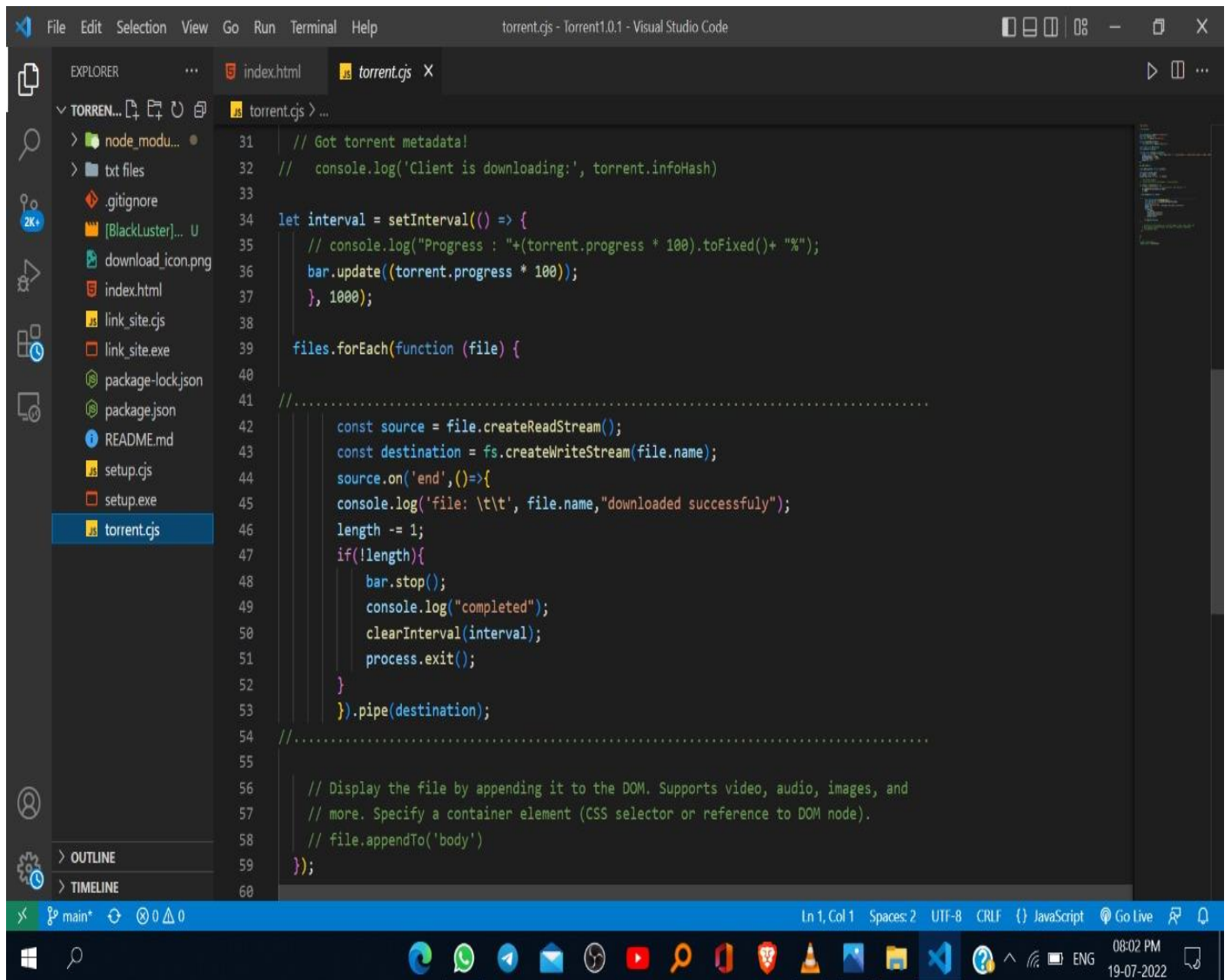


```
31
32 app.use('/post', express.static(path.join(__dirname, '/')));
33
34 router.post('/post', urlencodedParser, function(req,res){
35   console.log(req.body);
36   //var myText = req.body.url; //mytext is the name of your input box
37   //res.send('Your Text:' +myText);
38   lib.plusSlides(req.body.url);
39   //res.render('post', {qs: req.query});
40   //res.redirect("/");
41 });
42
43 //add the router
44 app.use('/', router);
45 app.listen(process.env.port || 8080);
46
47 console.log('To download file search following url in any browser:\nhttp://localhost:8080/post');
48
49
50 // "start": "react-scripts start",
51 // "test": "react-scripts test",
52
53 // .....
54
55 // import express from "express";
56 // const app = express();
57
58 // app.post("/post", (req, res) => {
59 //   console.log("Connected to React");
60 //   res.redirect("/");
```

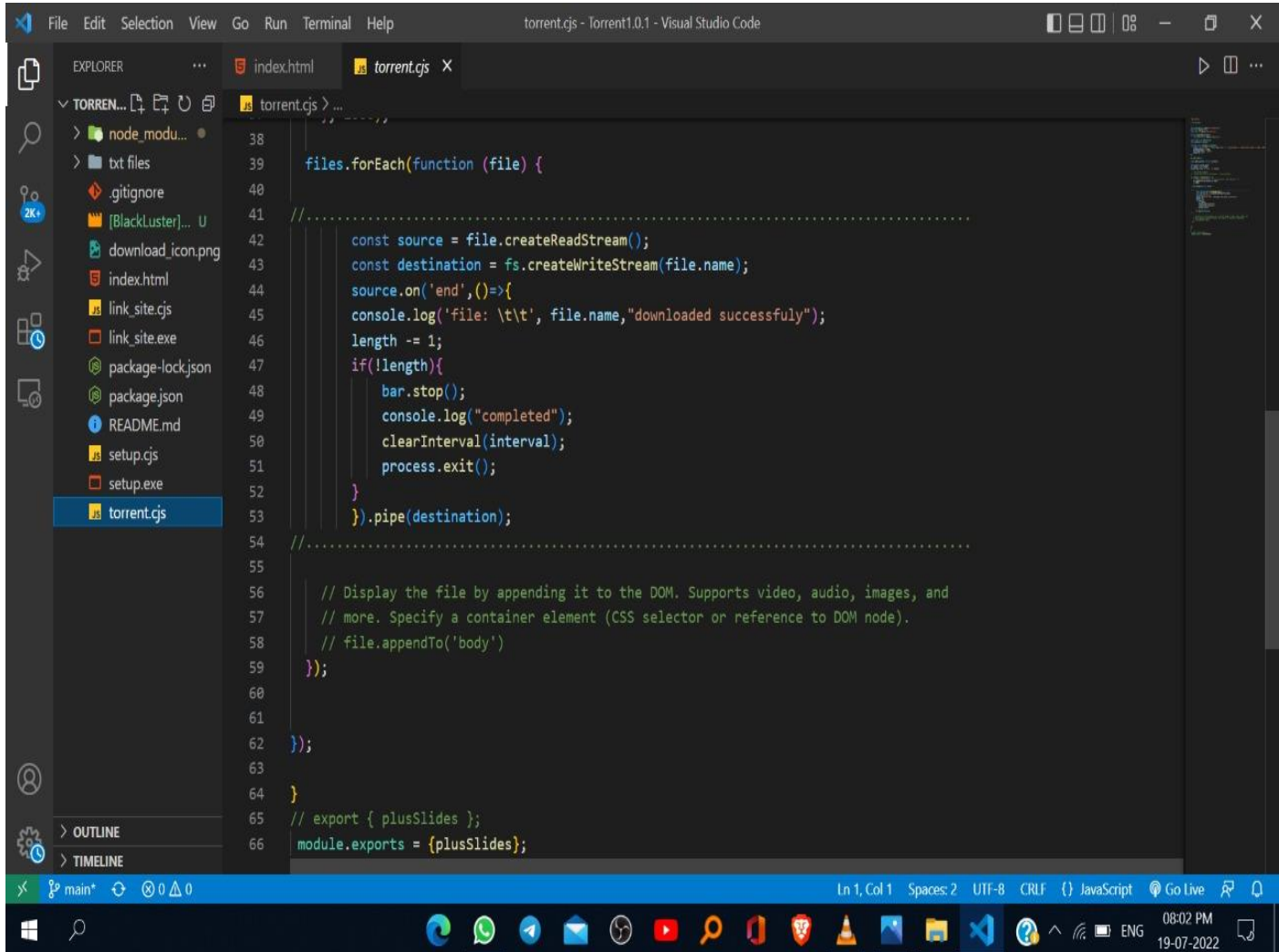
## Torrent .cjs



```
1 'use strict';
2
3 // using import
4
5
6 const cliProgress = require('cli-progress');
7 const fs = require("fs");
8 const colors = require('ansi-colors');
9
10 function plusSlides(torrent){
11   const WebTorrent = require('webtorrent')
12
13   const client = new WebTorrent()
14   const magnetURI = torrent;
15
16   const bar = new cliProgress.SingleBar({
17     format: 'CLI Progress |' + colors.cyan('{bar}') + ' | {percentage}% || {value}/{total} Chunks || Speed: speed',
18     barCompleteChar: '\u2588',
19     barIncompleteChar: '\u2591',
20     hideCursor: true
21   });
22
23   bar.start(100,0);
24
25   client.add(magnetURI, function (torrent) {
26
27     const files = torrent.files;
28     let length = files.length;
29     console.log("Number of files:- \t"+length);
30   });
```



```
31 // Got torrent metadata!
32 // console.log('Client is downloading:', torrent.infoHash)
33
34 let interval = setInterval(() => {
35 // console.log("Progress : "+(torrent.progress * 100).toFixed()+ "%");
36 bar.update((torrent.progress * 100));
37 }, 1000);
38
39 files.forEach(function (file) {
40
41 //.....
42 const source = file.createReadStream();
43 const destination = fs.createWriteStream(file.name);
44 source.on('end', ()=>{
45 console.log('file: \t\t', file.name, "downloaded successfully");
46 length -= 1;
47 if(!length){
48 bar.stop();
49 console.log("completed");
50 clearInterval(interval);
51 process.exit();
52 }
53 }).pipe(destination);
54 //.....
55
56 // Display the file by appending it to the DOM. Supports video, audio, images, and
57 // more. Specify a container element (CSS selector or reference to DOM node).
58 // file.appendTo('body')
59 });
60
```



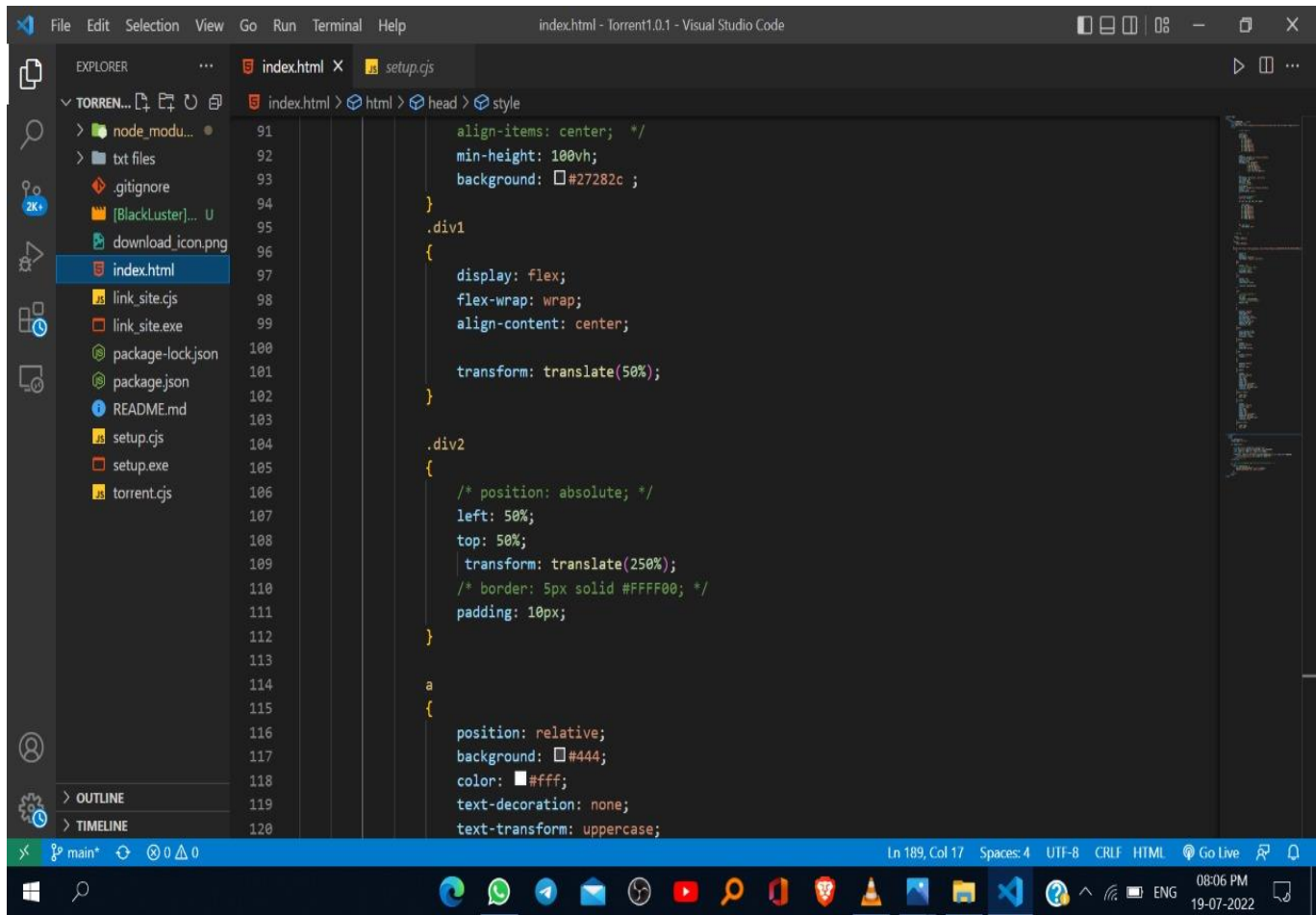
```
38
39   files.forEach(function (file) {
40
41     //.....
42     const source = file.createReadStream();
43     const destination = fs.createWriteStream(file.name);
44     source.on('end', ()=>{
45       console.log('file: \t\t', file.name, "downloaded successfully");
46       length -= 1;
47       if(!length){
48         bar.stop();
49         console.log("completed");
50         clearInterval(interval);
51         process.exit();
52       }
53     }).pipe(destination);
54
55     //.....
56     // Display the file by appending it to the DOM. Supports video, audio, images, and
57     // more. Specify a container element (CSS selector or reference to DOM node).
58     // file.appendTo('body')
59   });
60
61
62 });
63
64 }
65 // export { plusSlides };
66 module.exports = {plusSlides};
```



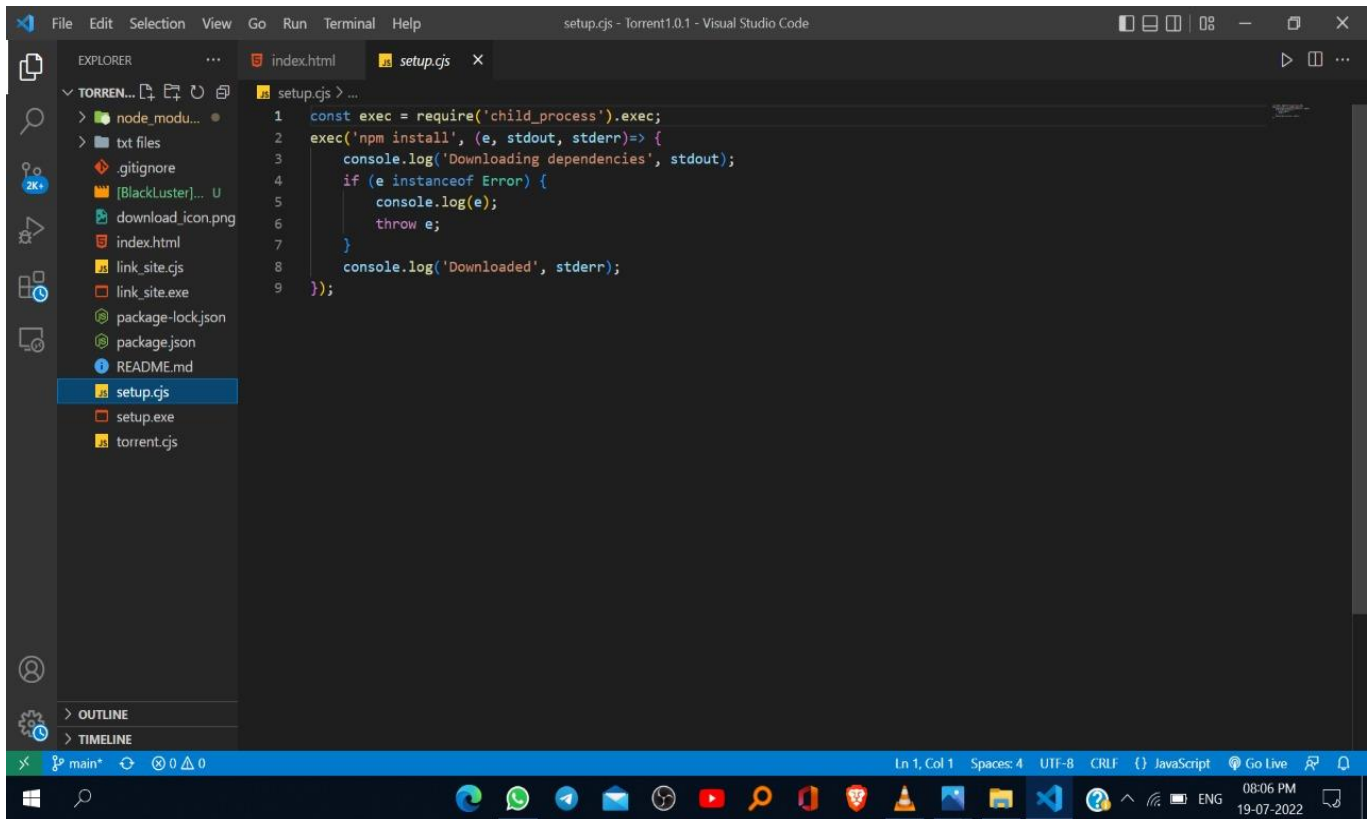




```
121 font-size: 1.5em;
122 letter-spacing: 0.1em;
123 padding: 10px 30px;
124 transition: 0.5s;
125 }
126 a:hover
127 {
128     letter-spacing: 0.25em;
129     background: var(--clr);
130     color: var(--clr);
131     box-shadow: 0 0 35px;
132 }
133 a::before
134 {
135     content: '';
136     position: absolute;
137     inset: 2px;
138     background: #27282c;
139 }
140 a span
141 {
142     position: relative;
143     z-index: 1;
144 }
145 a i
146 {
147     position: absolute;
148     inset: 0;
149     display: block;
150 }
```



## Setup.cjs



```
1  const exec = require('child_process').exec;
2  exec('npm install', (e, stdout, stderr) => {
3    console.log('Downloading dependencies', stdout);
4    if (e instanceof Error) {
5      console.log(e);
6      throw e;
7    }
8    console.log('Downloaded', stderr);
9  });
```

## Technology Frame works :

### GitHub:

GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on to the projects from anywhere. GitHub is an highly popularly program resource used for code sharing. It is the most Of the prominent source code host with over 60 million new repositories created in 2020.



## Node js :

Node. Js (Node) is an open source of the development platform for executing JavaScript code to server-side. Node is useful for developing applications that require a persistent connection from the browser to the server and is often used for real-time applications as chat, news feeds and web push to the notifications. Node.



## Html :

HyperText Mark up Language (HTML) is the set of mark up symbols or codes inserted into a file intended for display on the Internet. The mark up tells web browsers how to display a web page's words and images.

Each individual piece mark up code (which would fall between "<" and ">" characters) is referred to as an element element, though many people also refer to it as a tag. Some elements come in pairs that indicate when some display effect is to begin and when it is to end.

## **CODE REVIEW AND TESTING**

- Code Review, also known as Peer Code Review, is the act between the consciously and systematically convening with one's fellow of programmers to check each other's code for mistakes and have to been repeatedly shown to accelerate and streamline the process of software development like few other practices can. There are peer code review tools and software, but the concept itself is important to understand. Software is written by human beings. Software is therefore often riddled with mistakes.
  
- Software testing is the process of evaluating *and* verifying the software product or application does what it is supposed to do. The benefits of testing include preventing bugs, reducing development costs and improving performance.

## **IMPLEMENTATION**

*The Torrent downloader specification is free to use and many clients are open source, so the BitTorrent clients have been created for all the common operating systems using a variety of the programming languages. The official BitTorrent client, µTorrent, qBittorrent. Transmission, Vase, and BitComet are some of the most popular clients.*

*For example, this can be used to be centralized file sharing on a single dedicated server which users share access to on the network. Server-oriented Torrent downloader implementations can also be hosted by hosting the providers at co-located facilities with the high bandwidth Internet connectivity (eg, datacentre) which can provide dramatic speed benefits over using BitTorrent from a regular home broadband connection.*



## **SCOPE OF THE PROJECT**

An innovative torrent program to search for the torrents and downloading into your default computer system.

Functionality:

- Search by multiple trackers and result grouping.
- Creating a server for instant playback media files with VLC

## **CONCLUSION**

*In this paper we have presented detailed study and analysis of torrent downloader. We Believe that this study is the contribution of ongoing effort to gain insight into the behaviour of widely used peer to peer system. In order to share our findings we have all published our raw data files. One of the big advantage of this project is high level integrity of both content a and the meta data due to the working of global components.*

## **REFERENCE**

- <https://www.w3schools.com/css>
- <https://expressjs.com/en/starter/static-files.html>
- <https://www.npmjs.com/package/webtorrent-fix>