Lab – 01
Experiment: Implement the NodeJS Programs.

	Course Name: Advanced Web Technology	**EXPERIMENT NO. 1**	
	Course Code: 20CP314P **Faculty: Komal Singh**	**Branch:** **CSE**	**Semester: VI**
(To be filled by Student) **Submitted by:** Patel Vraj ChetanKumar **Roll no:** 21BCP362			

Objective: Building a Simple Web Server in Node.js

Q.1 Write Node JS programs for following modules:

a. http
b. filesystem
c. url
d. event

In Practical 1, I developed a Node.js web application that illustrates the practical use of core Node.js modules: **http**, **fs**, **url**, and **events**. The application serves a web page with two main functionalities: a file upload simulation that leverages the **fs** module to write to a file, and a URL parsing feature that utilizes the **url** module to dissect and display the components of a user-submitted URL. This setup provided a hands-on experience in handling HTTP requests, file operations, and URL manipulations within a Node.js environment.

The server, created with the **http** module, listens on port 3000 and dynamically responds to user actions. The design incorporates a simple yet elegant user interface styled with CSS, aiming for clarity and ease of use. Through this project, I explored event-driven programming by emitting and handling events related to the file upload process, demonstrating the asynchronous capabilities and flexibility of Node.js.

This practical reinforced my understanding of Node.js's modular architecture and its potential for developing scalable web applications. By integrating different modules to create a functional server, I gained insights into the Node.js ecosystem and its application in real-world scenarios. This project serves as a foundational step towards more complex developments, emphasizing the importance of core modules in Node.js for backend programming.

index.html file

```
<!DOCTYPE html> <html lang="en"> <head> <meta charset="UTF-8"> <title>Node.js Combined Modules
Example</title> <style> body { background-color: #f9f9f9; display: flex; justify-content:
center; align-items: center; height: 100vh; margin: 0; font-family: Arial, sans-serif; }
.form-container { background-color: #002b36; padding: 20px; border-radius: 8px; box-shadow: 0
4px 6px rgba(0,0,0,0.1); color: #ffffff; } form { display: flex; flex-direction: column; }
input[type="text"], input[type="file"] { margin-bottom: 20px; padding: 10px; border-radius:
4px; border: 1px solid #004359; background-color: #003648; color: #ffffff; }
input[type="submit"] { background-color: #2496ED; color: white; padding: 10px 20px; border:
none; border-radius: 4px; cursor: pointer; transition: background-color 0.3s; }
input[type="submit"]:hover { background-color: #0072C6; } h2 { color: #61DAFB; } </style>
</head> <body> <div class="form-container"> <h2>File Upload</h2> <form action="/upload"
method="post" enctype="multipart/form-data"> <input type="file" name="sourcefile"> <input
type="submit" value="Upload File"> </form> <h2>URL Parsing</h2> <form action="/parse-url"
method="post"> <input type="text" name="url" placeholder="Enter URL"> <input type="submit"
value="Parse URL"> </form> </div> </body> </html>
```

server.js

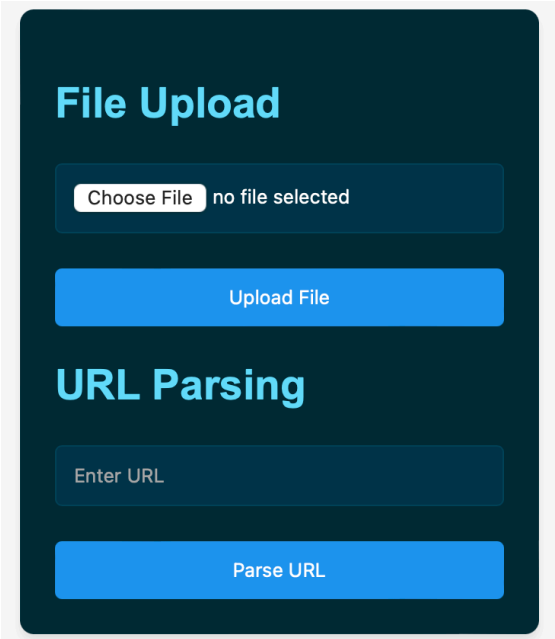
```
const http = require('http'); const fs = require('fs'); const url = require('url'); const {
parse } = require('querystring'); const EventEmitter = require('events'); class MyEmitter
extends EventEmitter {} const myEmitter = new MyEmitter(); const server =
http.createServer((req, res) => { if (req.method === 'GET') { fs.readFile('./index.html',
(err, data) => { if (err) { res.writeHead(500); return res.end('Error loading index.html'); }
res.writeHead(200, { 'Content-Type': 'text/html' }); res.end(data); }); } else if (req.method
=== 'POST') { if (req.url === '/upload') { fs.writeFile('destination.txt', 'FS worked
successfully', (err) => { if (err) { res.writeHead(500); return res.end('Error writing file');
} myEmitter.emit('fileUploaded'); res.writeHead(200, { 'Content-Type': 'text/plain' });
res.end('File uploaded and message written to destination.txt'); }); } else if (req.url ===
'/parse-url') { let body = ''; req.on('data', chunk => { body += chunk.toString(); // Convert
Buffer to string. }); req.on('end', () => { const parsedBody = parse(body); const parsedUrl =
new URL(parsedBody.url); res.writeHead(200, { 'Content-Type': 'text/plain' }); res.end(`Parsed
URL:\nHost: ${parsedUrl.host}\nPathname: ${parsedUrl.pathname}\nSearch Params:
${parsedUrl.search}`); }); } } }); const PORT = 3000; server.listen(PORT, () =>
console.log(`Server running on port ${PORT}`)); myEmitter.on('fileUploaded', () => {
console.log('A file was uploaded and processed.'); });
```

GITHUB LINK FOR RAW CODES

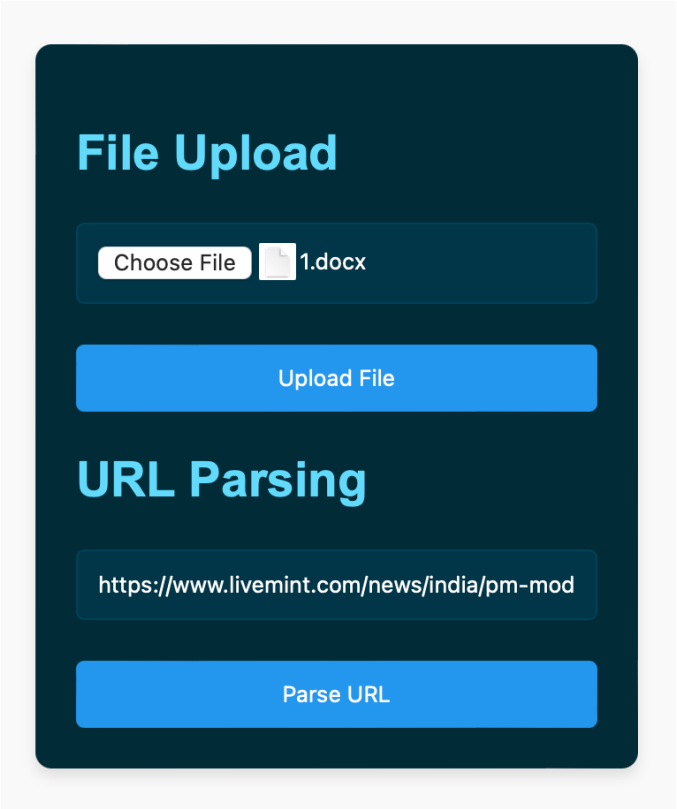
GITHUB_RAW_CODE

OUTPUT

```
vrajpatel@vrajs-MacBook-Pro p1 % nodemon server.js
[nodemon] 3.0.3
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node server.js`
Server running on port 3000
```



INPUT => FILE & URL



OUTPUT OF “FS” MODULE

```
File uploaded and message written to destination.txt
```

```
FS worked successfully
```

destination.txt
index.html
server.js

OUTPUT OF “URL” MODULE

https://www.livemint.com/news/india/pm- modi-says-i-fell-ashamed-at-arambagh-rally-on-sandeshkhali-incident-mamata-banerjee-tmc-bjp-11709289383371.html

```
Parsed URL:
Host: www.livemint.com
Pathname: /news/india/pm-modi-says-i-fell-ashamed-at-arambagh-rally-

-on-sandeshkhali-incident-mamata-banerjee-tmc-bjp-11709289383371.html
```

OUTPUT OF “EVENT” MODULE

we were able to track Events using evet module