Name:Mohana priyaa A

Class:CSE –A III- year

Register Number: 73772214160

Course Code:60CSE12

Course Name:Node.js and React.js

ASSIGNMENT-I

1.Blocking and Non-Blocking Code in Node.js

Node.js is naturally non-blocking because it uses an event-driven, asynchronous model. However, blocking code can still be written, for example, by using synchronous functions.

Blocking Code Example in Node.js (Synchronous)

```
const fs = require('fs');
console.log('Task 1 starting...');
const data = fs.readFileSync('file.txt', 'utf8');
console.log(data);
console.log('Task 1 finished.');
console.log('Task 2 starting...');
for (let i = 0; i < 1e9; i++) {}
console.log('Task 2 finished.');
```

Non-Blocking Code Example in Node.js (Asynchronous)

```
const fs = require('fs');
console.log('Task 1 starting...');
fs.readFile('file.txt', 'utf8', (err, data) => {
    if (err) throw err;
    console.log(data);
    console.log('Task 1 finished.');
});
console.log('Task 2 starting...');
for (let i = 0; i < 1e9; i++) {}
console.log('Task 2 finished.');
```

---

2. File System Module in Node.js with Example

Node.js provides the `fs` module for interacting with the file system :

```
const fs = require('fs');
fs.writeFile('example.txt', 'Hello, this is a sample file.', (err) => {
    if (err) throw err;
    console.log('File written successfully!');
    fs.readFile('example.txt', 'utf8', (err, data) => {
        if (err) throw err;
        console.log('File content:', data);
    });
});
```

3. Develop a REPL Program to Find Odd or Even Numbers (Node.js)

```
const readline = require('readline');
const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout
});

rl.setPrompt('Enter a number: ');
rl.prompt();
rl.on('line', (input) => {
  const num = parseInt(input, 10);
  if (isNaN(num)) {
    console.log('Please enter a valid number.');
  } else if (num % 2 === 0) {
    console.log(`${num} is an even number.`);
  } else {
    console.log(`${num} is an odd number.`);
  }
  rl.prompt();
}).on('close', () => {
  console.log('REPL closed.');
  process.exit(0);
});
```

---

4. Develop DNS Module in Node.js with Example

```
const dns = require('dns');
dns.lookup('example.com', (err, address, family) => {
    if (err) throw err;
    console.log('Address:', address);
```

```
  console.log('Family:', family);
});
dns.reverse('93.184.216.34', (err, hostnames) => {
  if (err) throw err;
  console.log('Hostnames:', hostnames);
});
```

---

5. Develop TCP Server and Client Program in Node.js

TCP Server Example

```
const net = require('net');
const server = net.createServer((socket) => {
  console.log('Client connected');
        socket.on('data', (data) => {
    console.log('Received:', data.toString());
    socket.write('Hello from server');
  });
        socket.on('end', () => {
    console.log('Client disconnected');
  });
});
server.listen(8080, () => {
  console.log('Server listening on port 8080');
});
```

TCP Client Example:

```
const net = require('net');
const client = net.createConnection({ port: 8080 }, () => {
  console.log('Connected to server');
  client.write('Hello from client');
});

client.on('data', (data) => {
  console.log('Received:', data.toString());
  client.end();
});

client.on('end', () => {
  console.log('Disconnected from server');
});
```