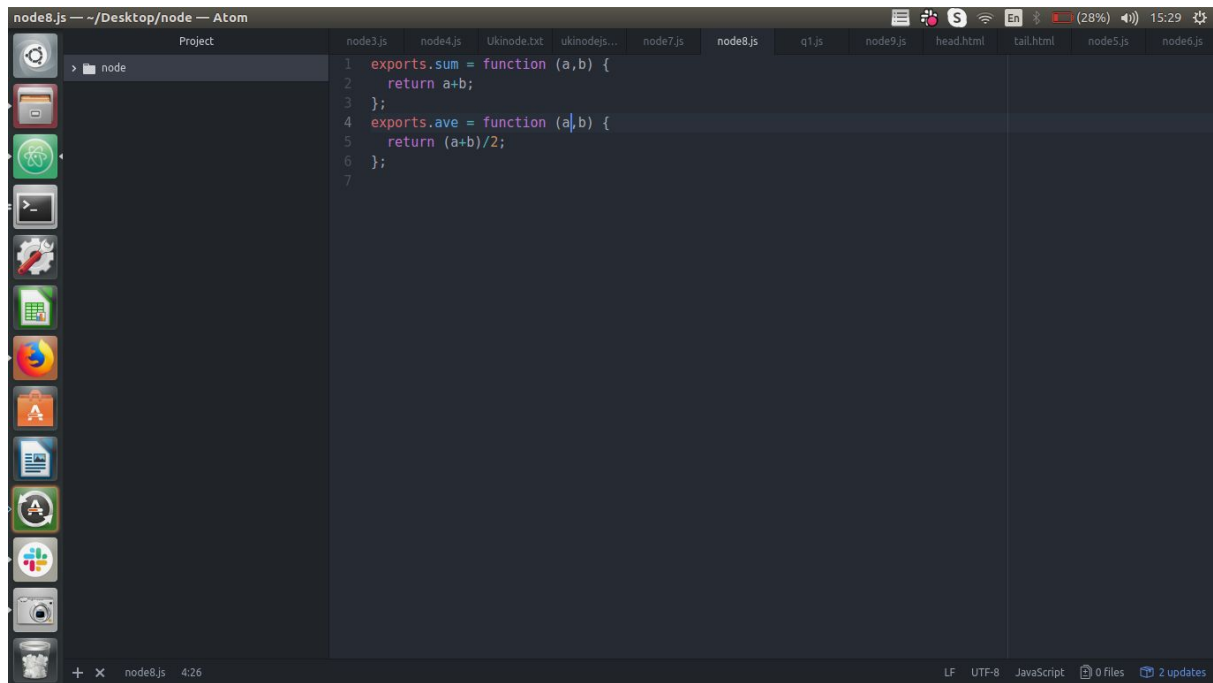


Node.js Exercise 1

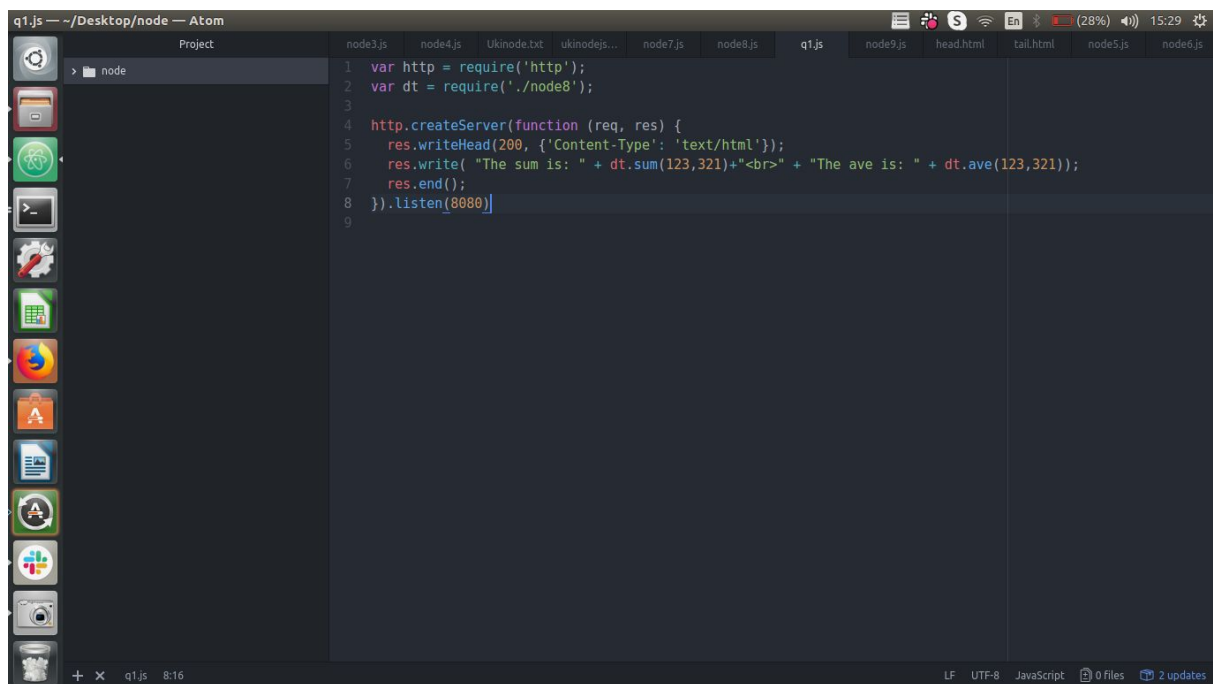
1. Create a custom module which returns the sum and average of any two numbers passed into it. Require the module and run the server by passing 123 and 321 so that the server prints out the sum and average.



The screenshot shows the Atom editor interface with a project named 'node'. The file 'node8.js' is open and contains the following JavaScript code:

```
1 exports.sum = function (a,b) {  
2   return a+b;  
3 };  
4 exports.ave = function (a,b) {  
5   return (a+b)/2;  
6 };  
7
```

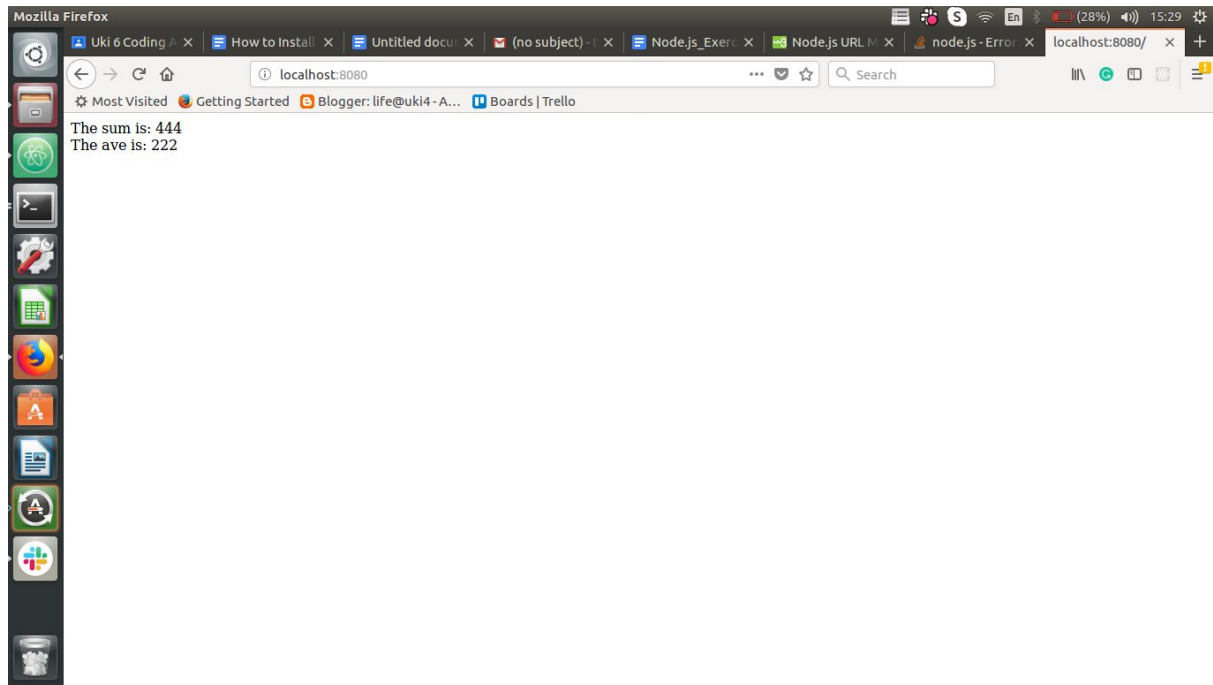
The status bar at the bottom indicates the file is 'node8.js', 4,26 lines long, using 'LF' line endings, 'UTF-8' encoding, 'JavaScript' language, with '0 files' and '2 updates'.



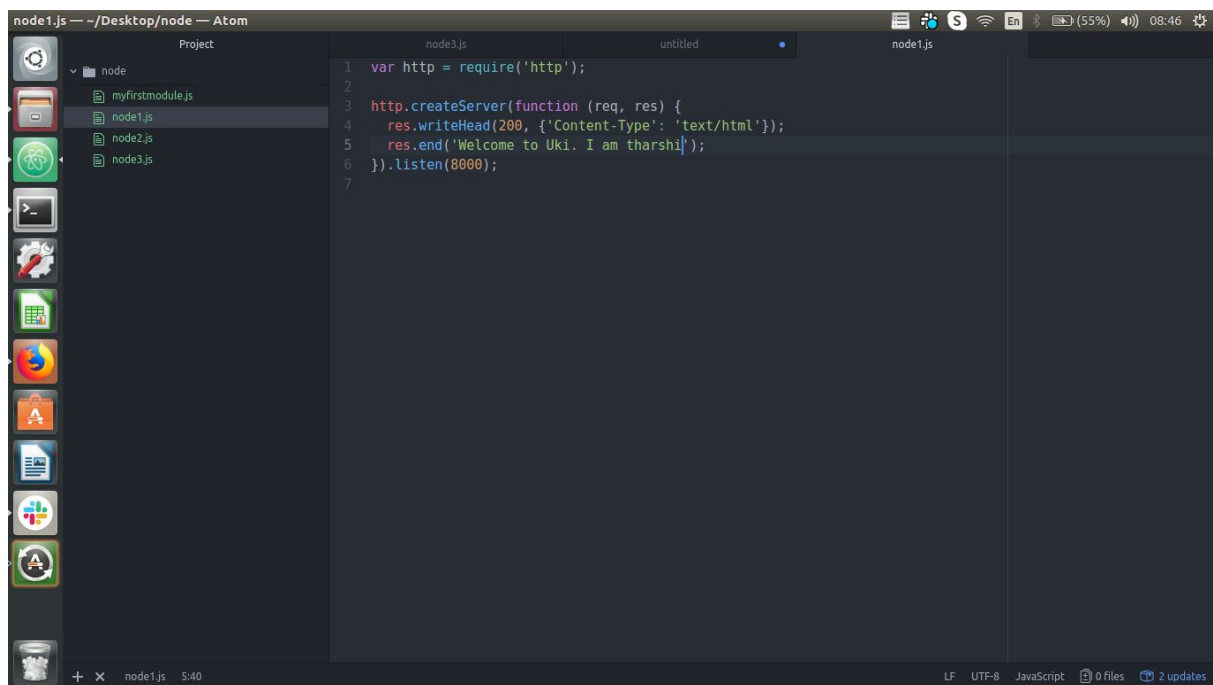
The screenshot shows the Atom editor interface with the same project 'node'. The file 'q1.js' is open and contains the following JavaScript code:

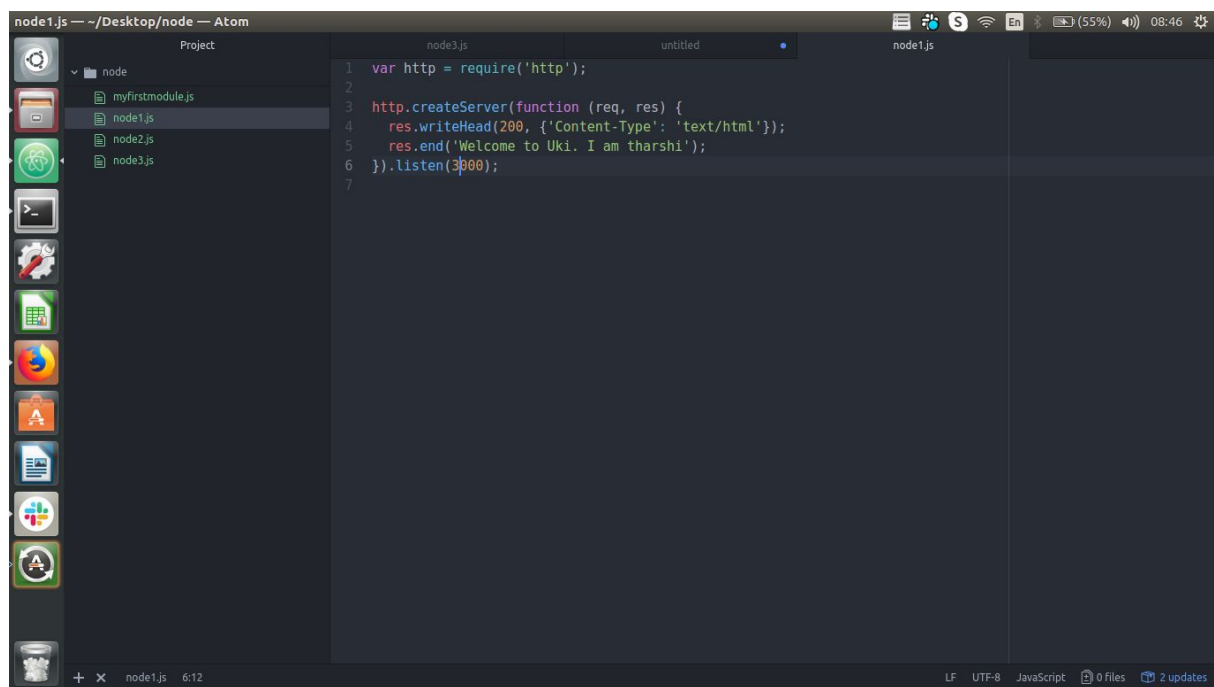
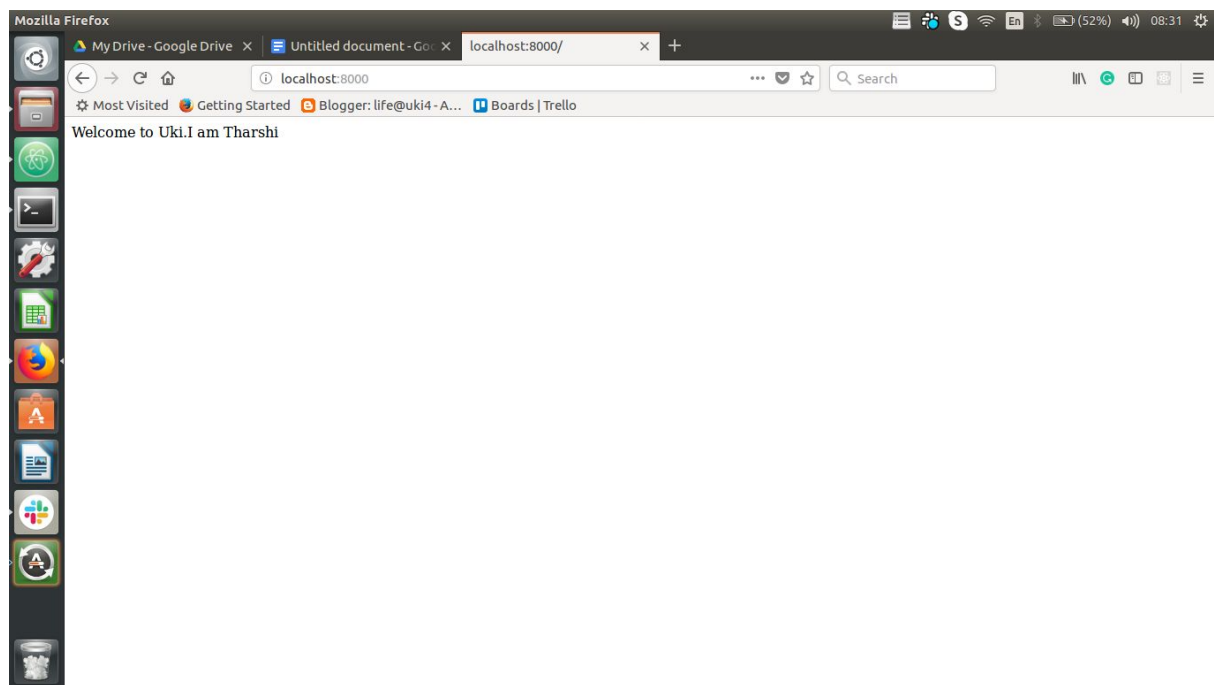
```
1 var http = require('http');  
2 var dt = require('./node8');  
3  
4 http.createServer(function (req, res) {  
5   res.writeHead(200, {'Content-Type': 'text/html'});  
6   res.write( "The sum is: " + dt.sum(123,321)+"<br>" + "The ave is: " + dt.ave(123,321));  
7   res.end();  
8 }).listen(8080)  
9
```

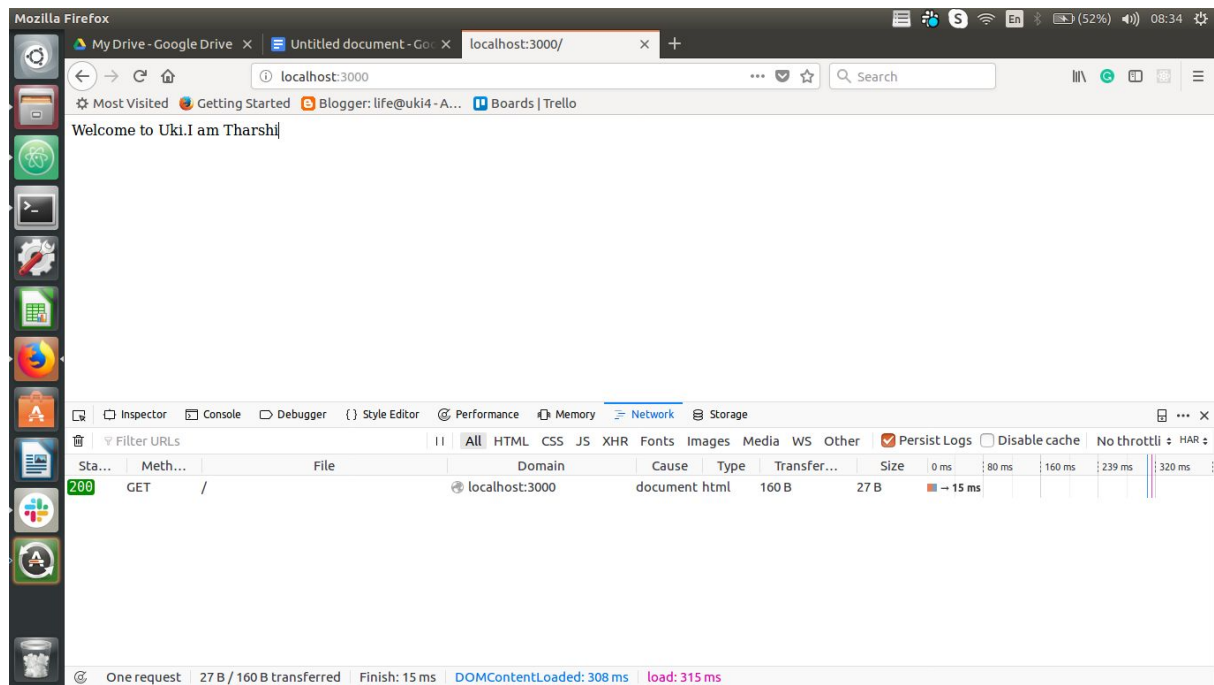
The status bar at the bottom indicates the file is 'q1.js', 8,16 lines long, using 'LF' line endings, 'UTF-8' encoding, 'JavaScript' language, with '0 files' and '2 updates'.



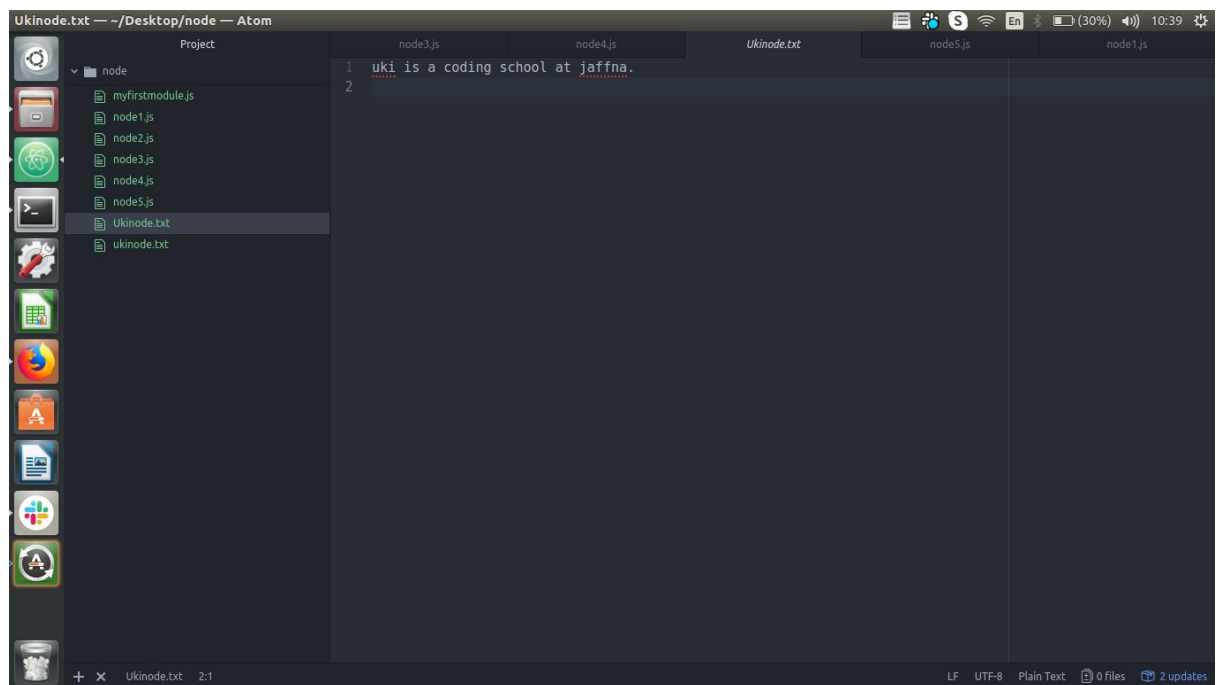
2. Create a simple http server and print “Welcome to Uki. I am **yourname**” when a request is sent to your server via the port 8000. (Note - Change different port numbers and check)



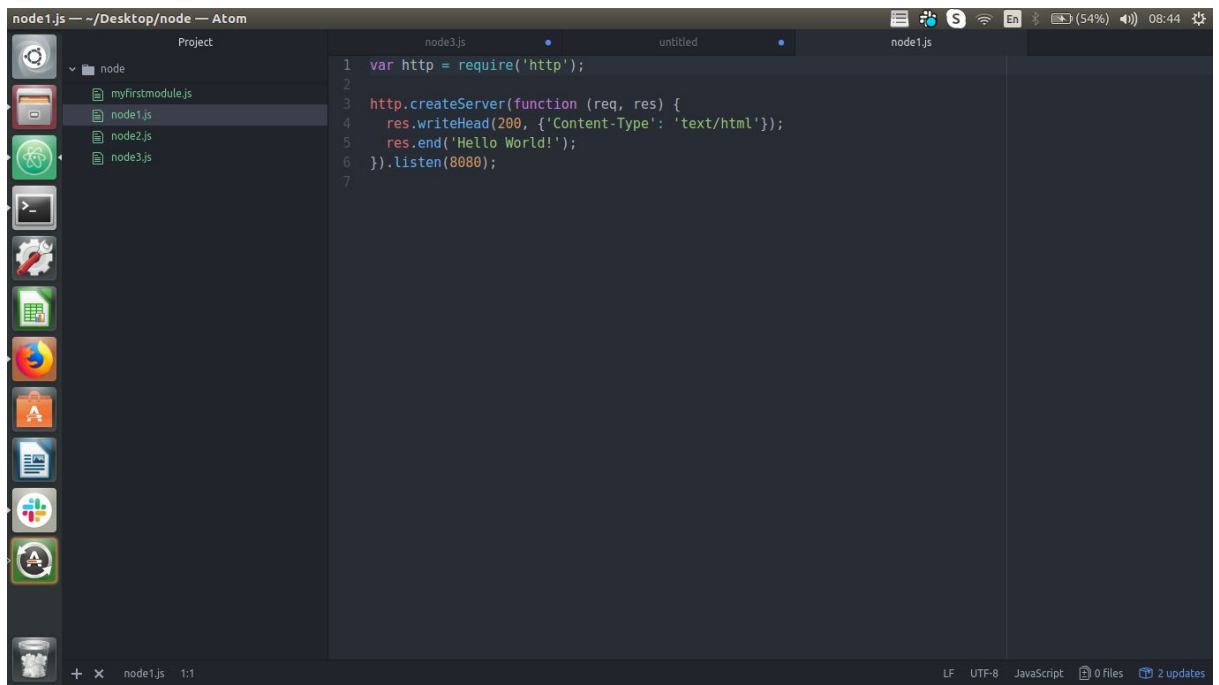




3. Using the file system module create a new file called ukinode.txt
 - 3.1 Write a paragraph about Uki into that file

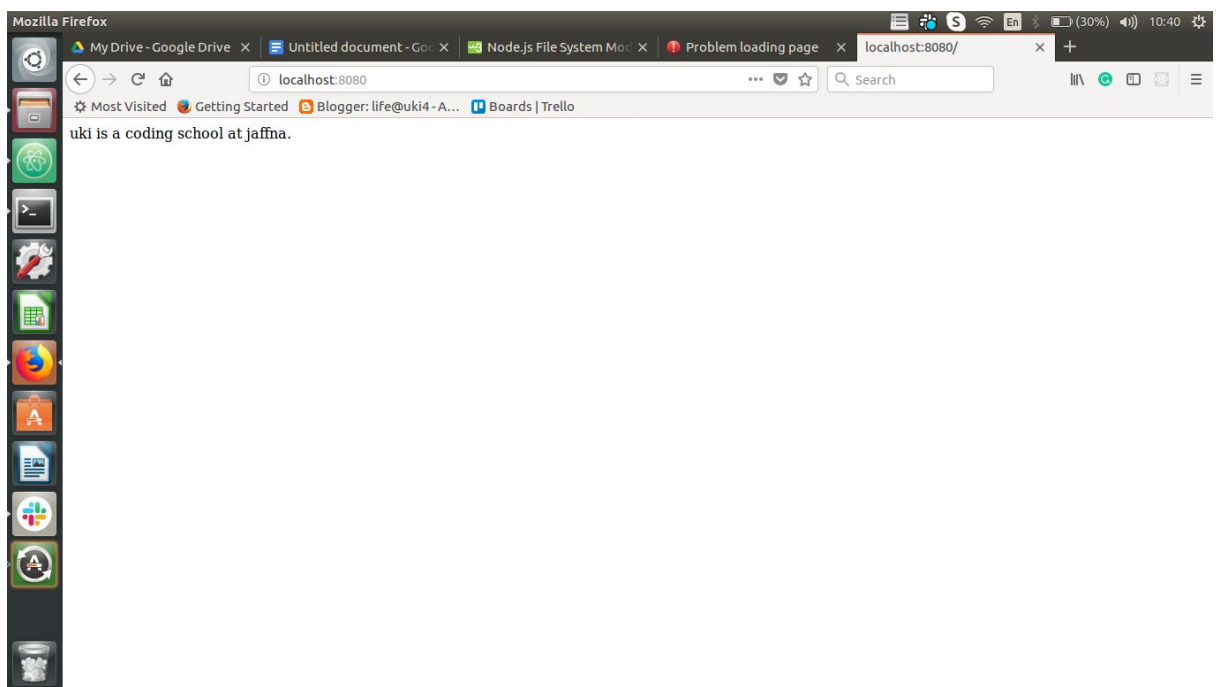


- 3.2 Serve that file to the client (Read File) over your server

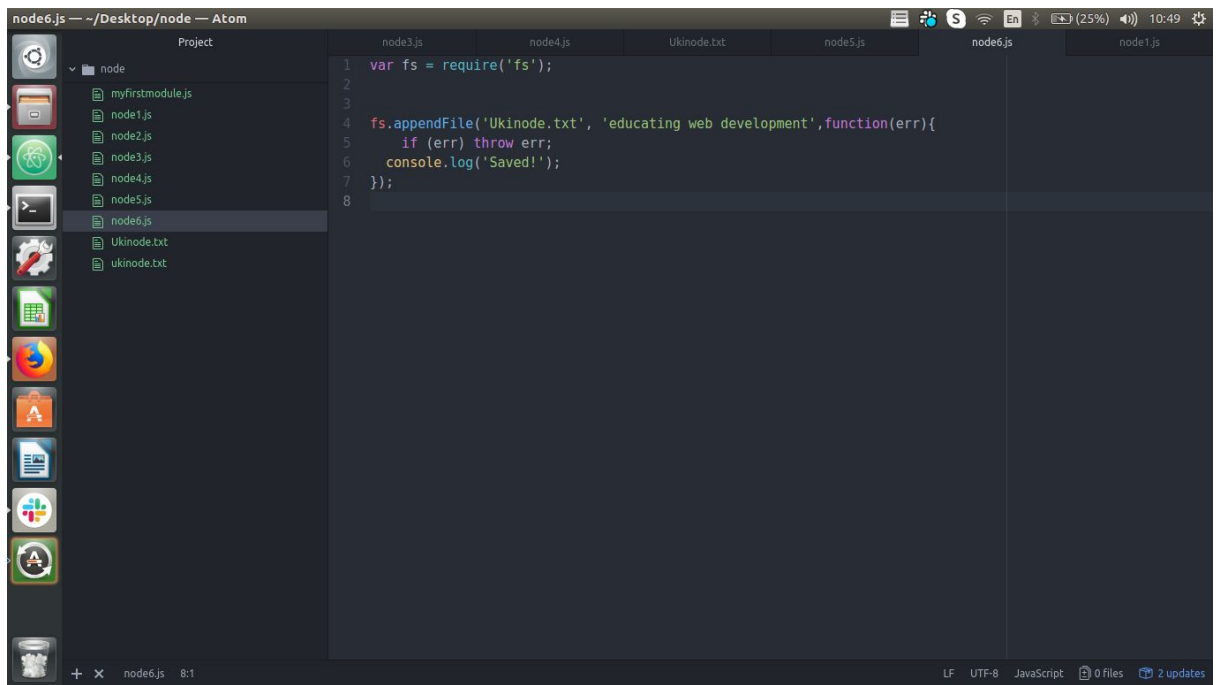


The screenshot shows the Atom code editor with a project named 'node' open. The file explorer on the left shows files: 'myfirstmodule.js', 'node1.js', 'node2.js', and 'node3.js'. The main editor area shows the content of 'node1.js', which is a simple HTTP server script. The status bar at the bottom indicates the file is 'node1.js', 1:1, and the encoding is 'UTF-8'.

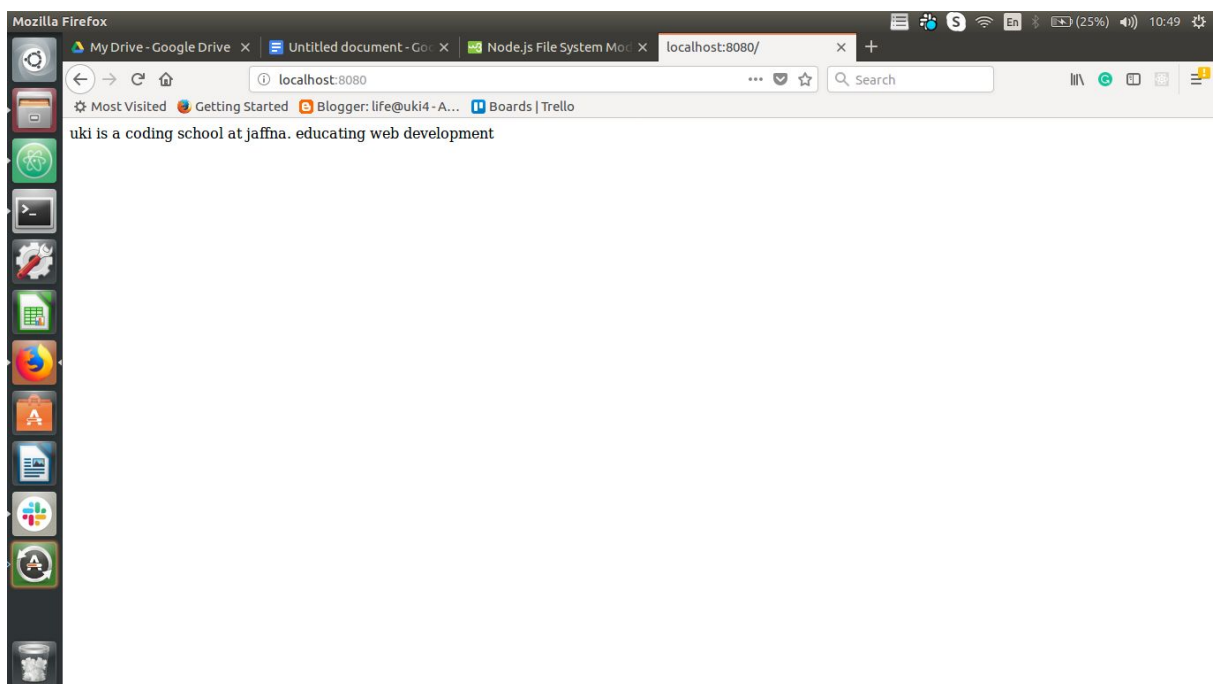
```
1 var http = require('http');
2
3 http.createServer(function (req, res) {
4   res.writeHead(200, {'Content-Type': 'text/html'});
5   res.end('Hello World!');
6 }).listen(8080);
7
```



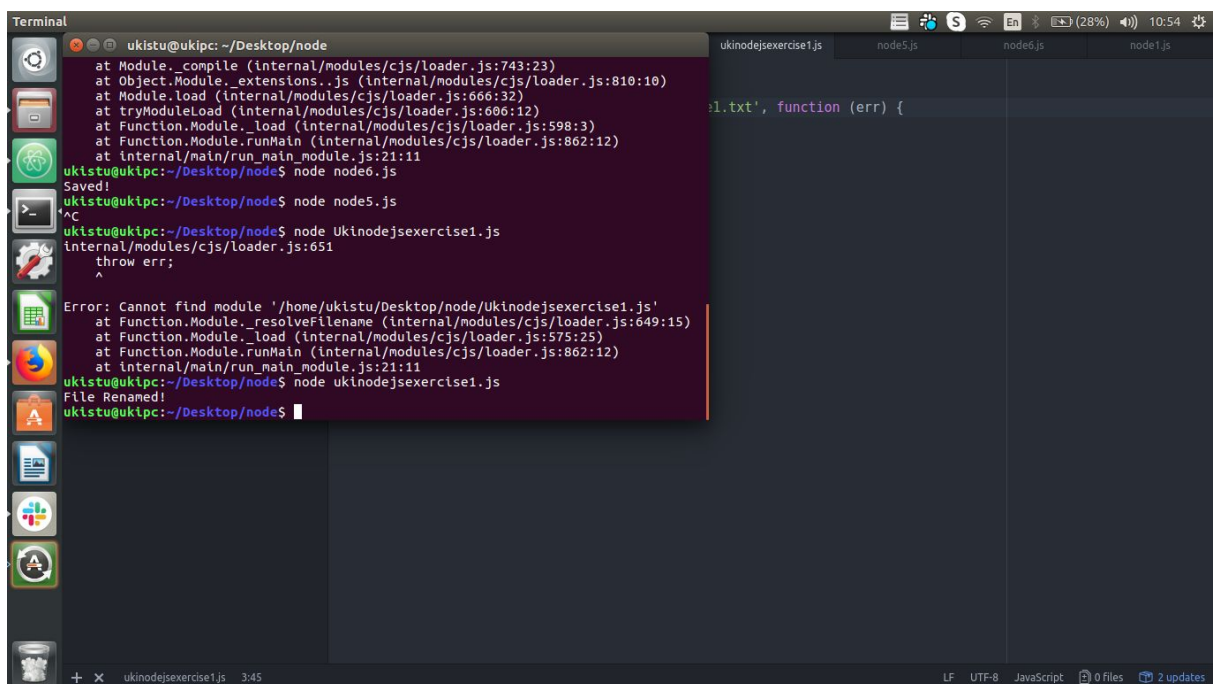
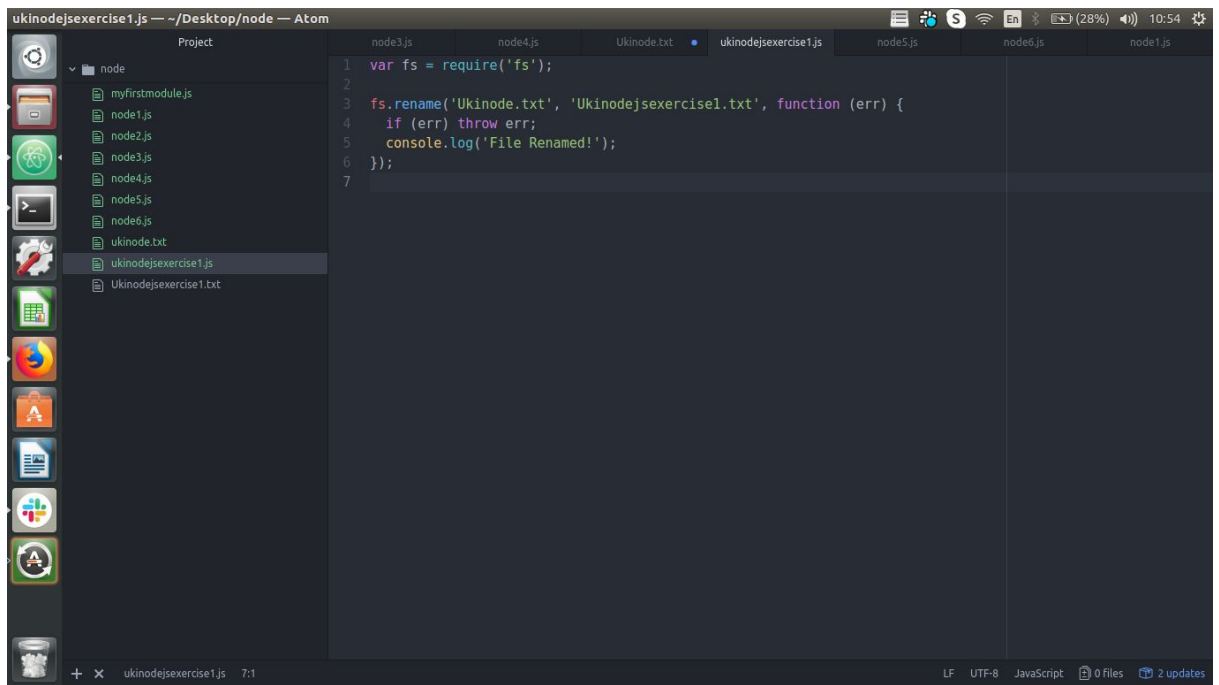
3.3 Append another paragraph about Uki and now serve the new file



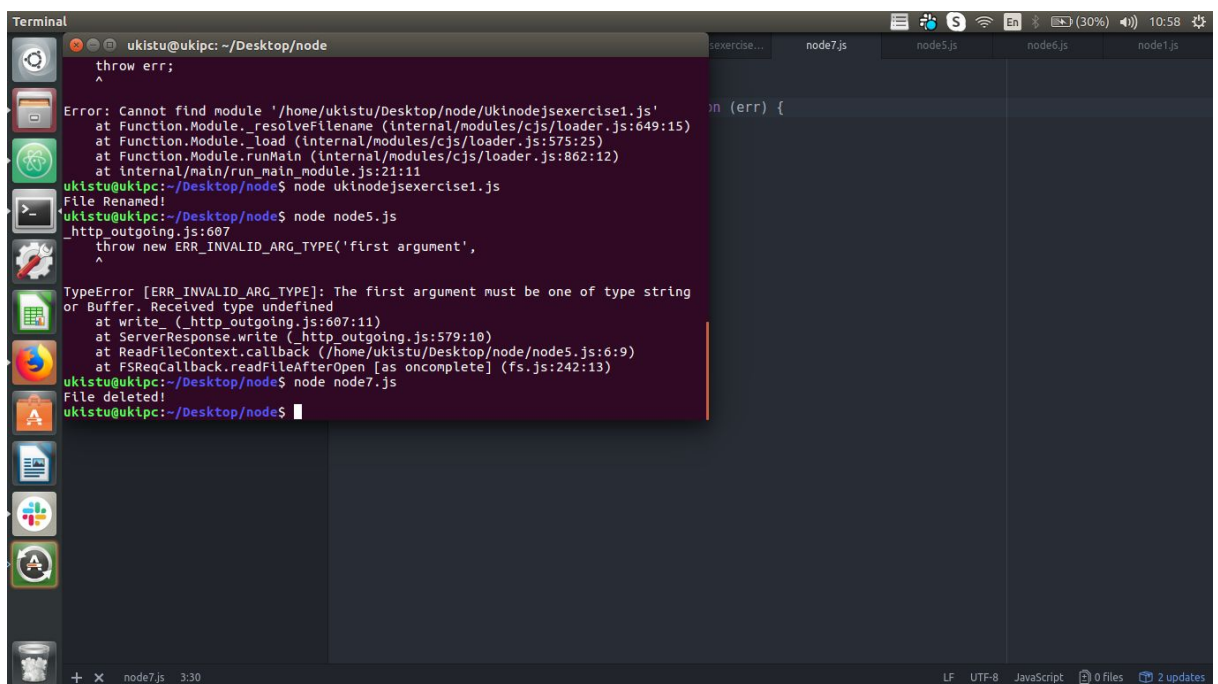
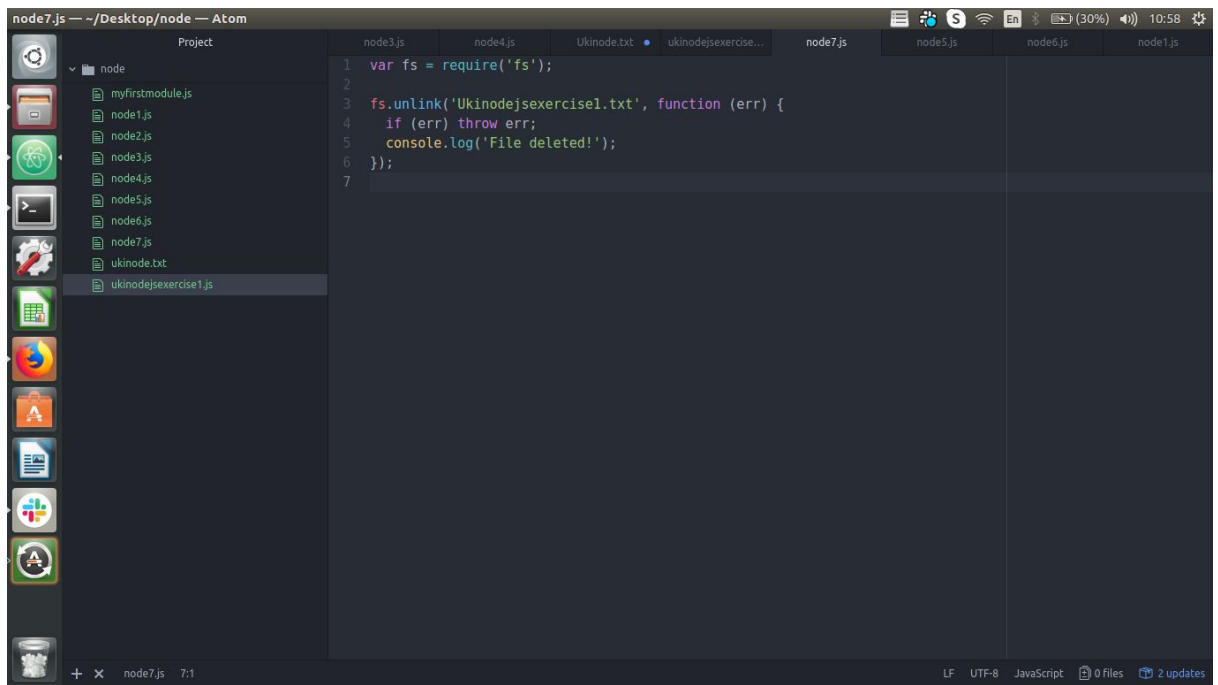
```
1 var fs = require('fs');
2
3
4 fs.appendFile('Ukinode.txt', 'educating web development',function(err){
5   if (err) throw err;
6   console.log('Saved!');
7 });
8
```



3.4 Rename the file as ukinodejsexercise1.txt



3.5 Delete the file you created



4. Create two html files called head.html which is a web page which says 'you have got head' and tail.html which is a web page which says 'you have got tail' and save them in the same folder as your node.js files. Create a Node.js file that opens the requested file and returns the content to the client. If anything goes wrong, throw a 404 error.

If you have followed the correct steps you should see two different results when opening these two addresses:

<http://localhost:8080/head.html> -> You have got head

<http://localhost:8080/tail.html> -> You have got tail

head.html — ~/Desktop/node — Atom

Project

node3.js node4.js Ukinode.txt ukinodejs... node7.js node8.js q1.js node9.js head.html tail.html node5.js node6.js

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <h1>head</h1>
5 <p>you have got head</p>
6 </body>
7 </html>
8
```

head.html 4:9

LF UTF-8 HTML 0 files 2 updates

tail.html — ~/Desktop/node — Atom

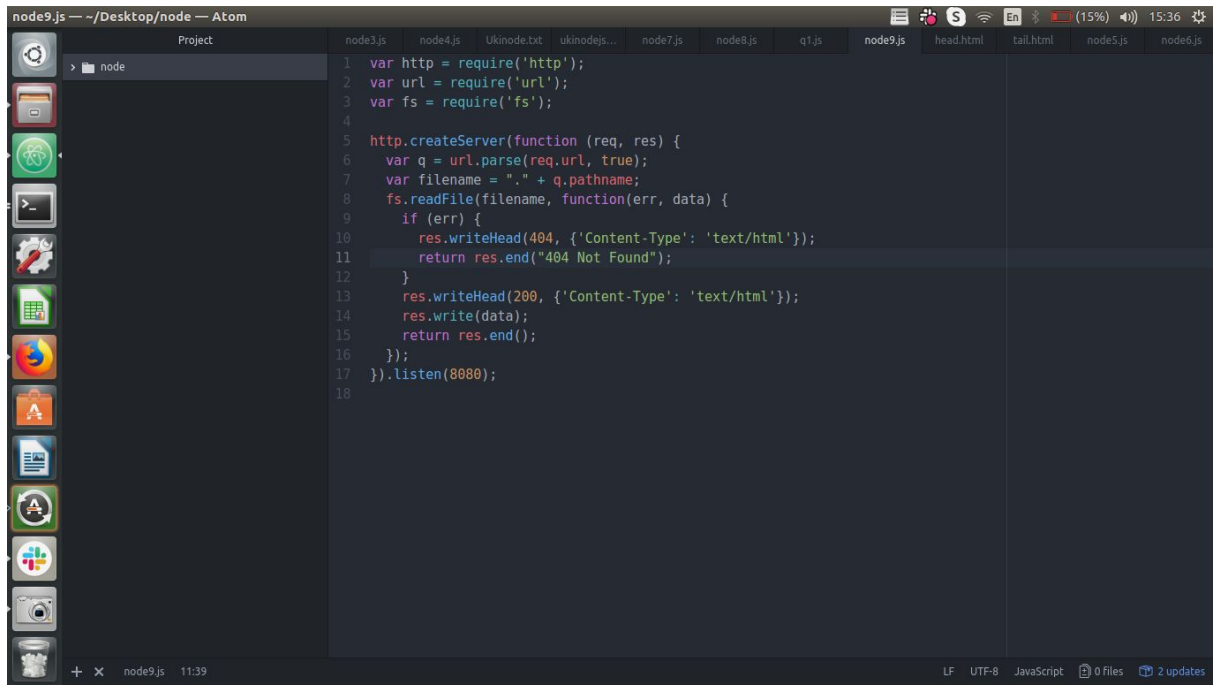
Project

node3.js node4.js Ukinode.txt ukinodejs... node7.js node8.js q1.js node9.js head.html tail.html node5.js node6.js

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <h1>tail</h1>
5 <p>you have got tail</p>
6 </body>
7 </html>
8
```

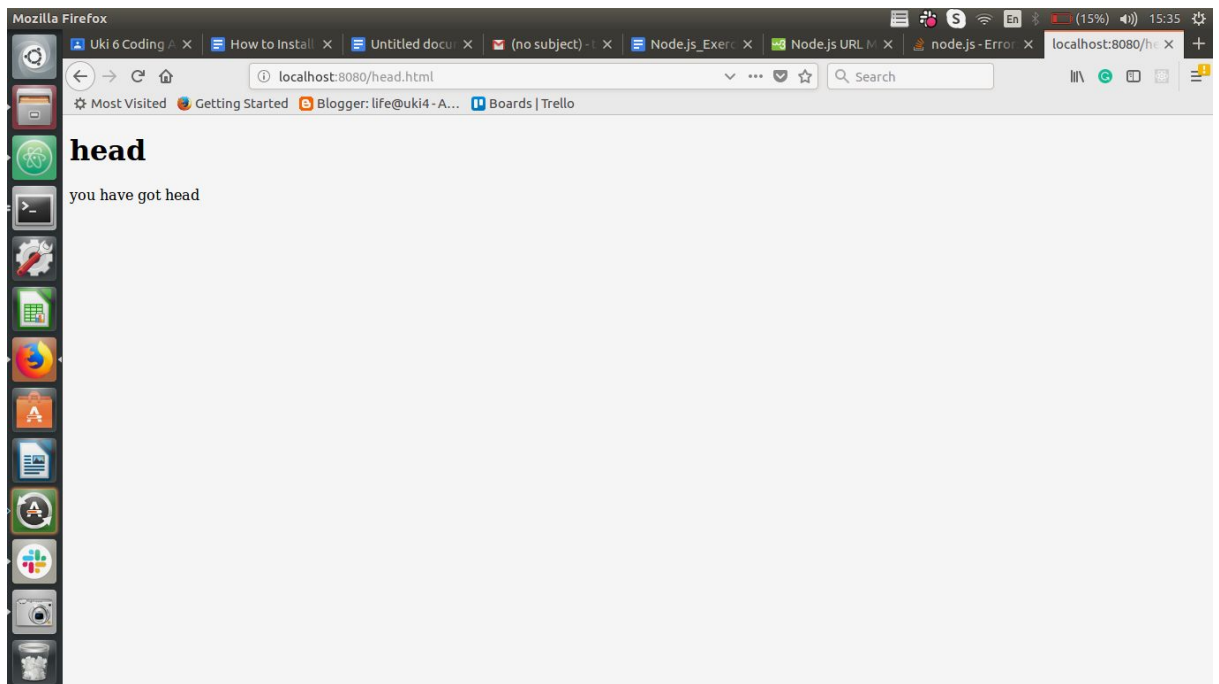
tail.html 4:9

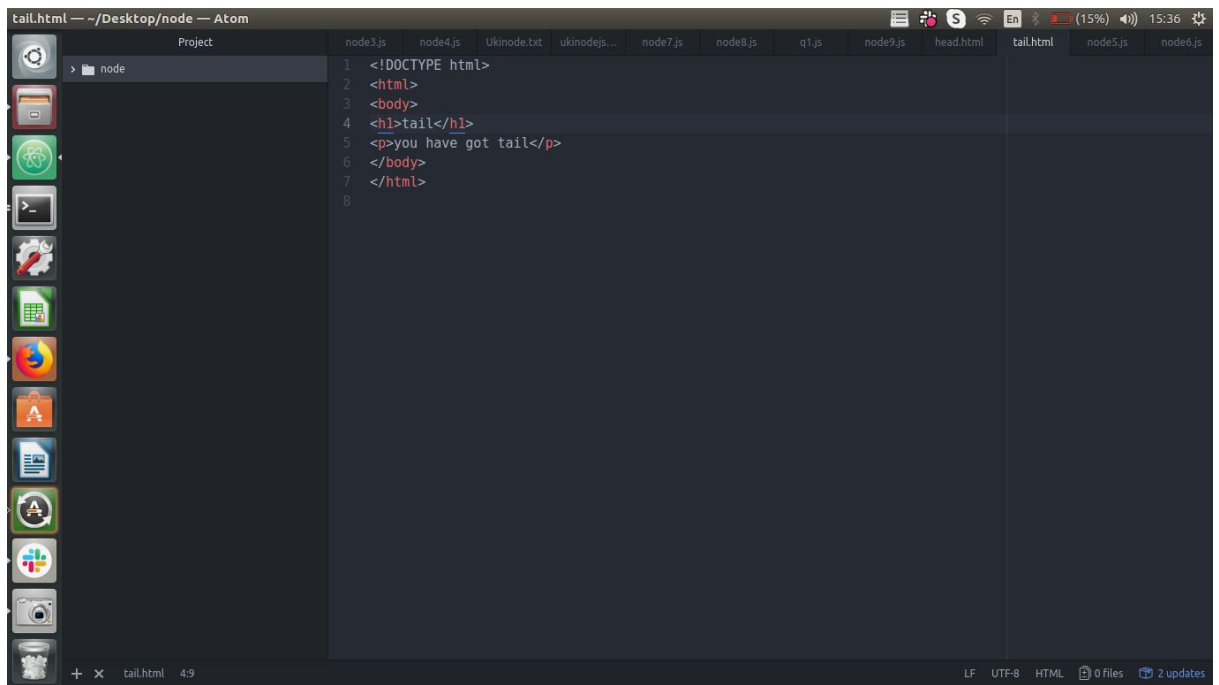
LF UTF-8 HTML 0 files 2 updates



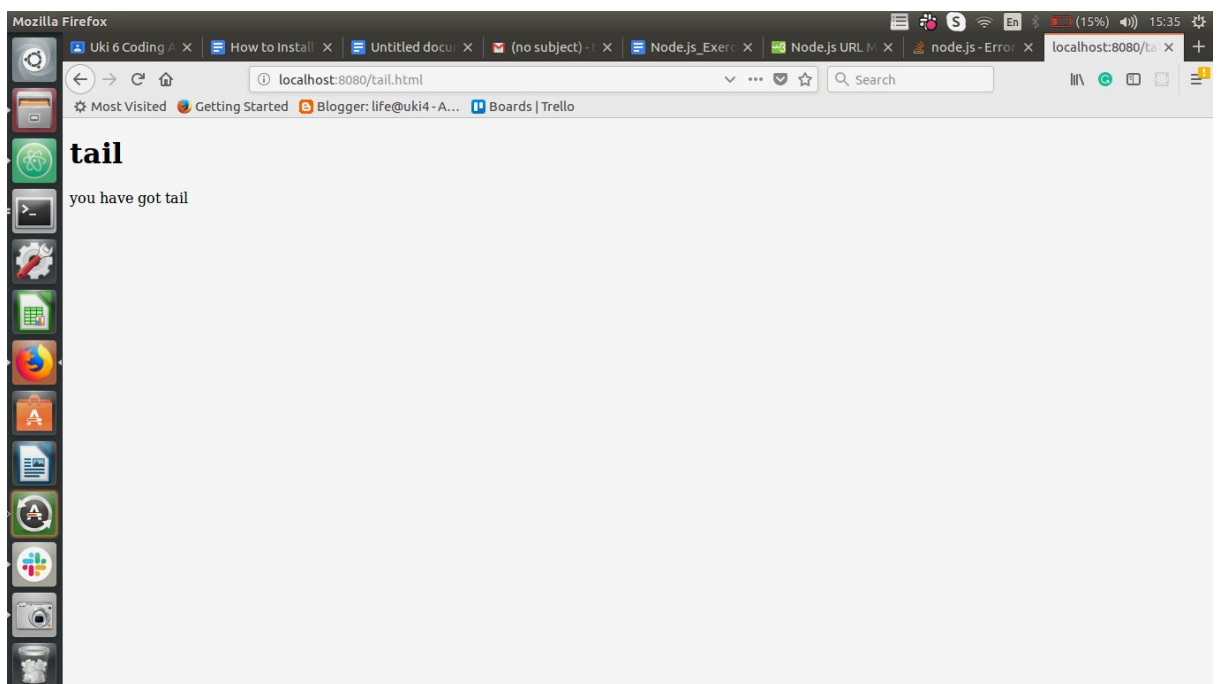
The screenshot shows the Atom code editor with a project named 'node' open. The file 'node9.js' is selected in the sidebar and is open in the main editor. The code is a Node.js script that uses the 'http' and 'fs' modules to create a simple web server. The server listens on port 8080. It checks for the presence of a file at the requested URL. If the file is not found, it returns a 404 status with the message '404 Not Found'. If the file is found, it returns a 200 status and the content of the file. The file 'node9.js' is the only file in the project.

```
1 var http = require('http');
2 var url = require('url');
3 var fs = require('fs');
4
5 http.createServer(function (req, res) {
6   var q = url.parse(req.url, true);
7   var filename = "." + q.pathname;
8   fs.readFile(filename, function(err, data) {
9     if (err) {
10      res.writeHead(404, {'Content-Type': 'text/html'});
11      return res.end("404 Not Found");
12    }
13    res.writeHead(200, {'Content-Type': 'text/html'});
14    res.write(data);
15    return res.end();
16  });
17 }).listen(8080);
18
```

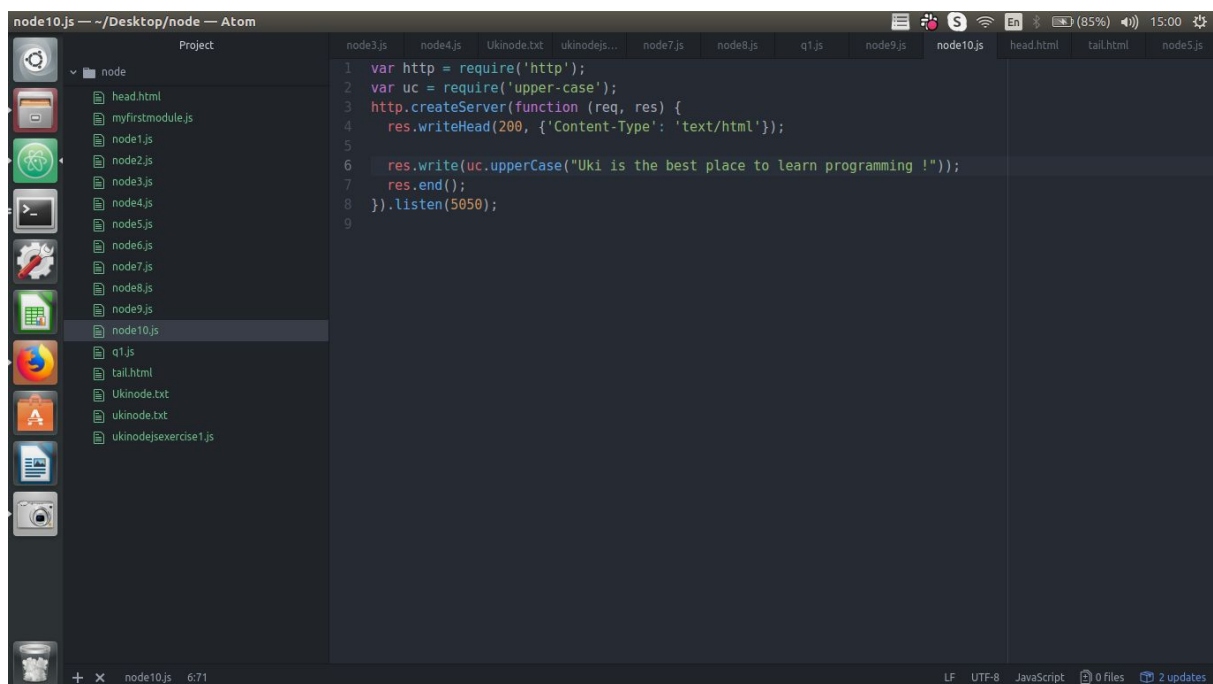
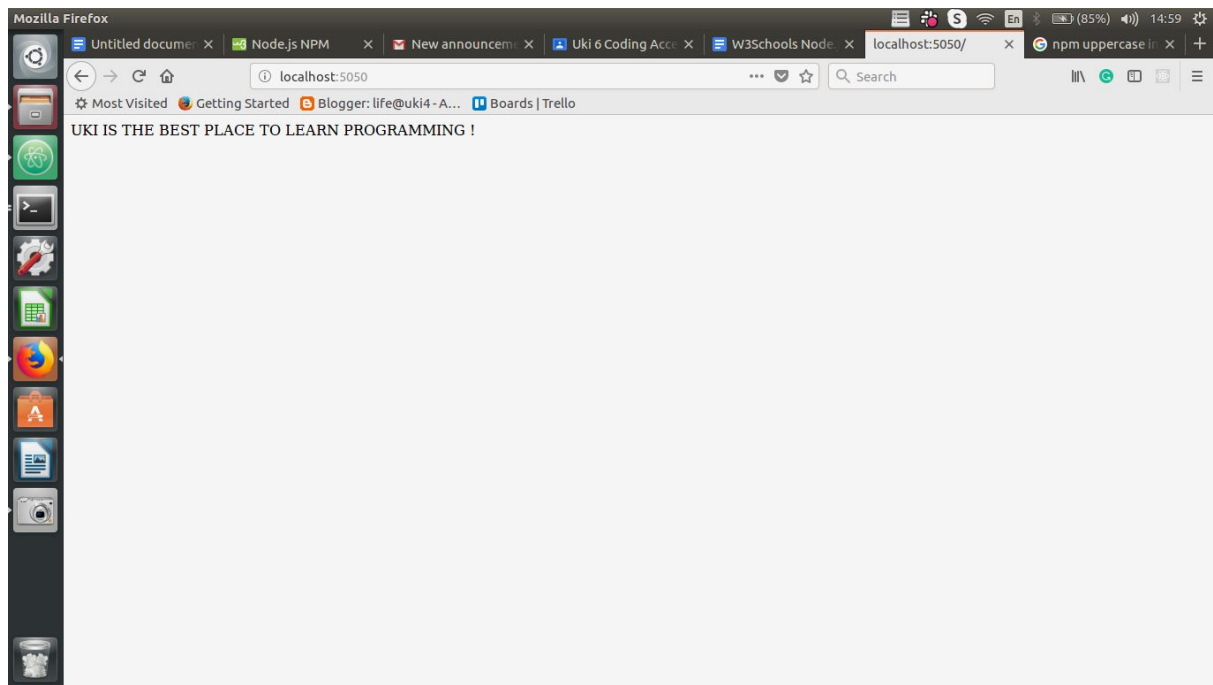




```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <h1>tail</h1>
5 <p>you have got tail</p>
6 </body>
7 </html>
8
```



5. Install the package “upper-case” using NPM and create a Node.js file that will convert the output "Uki is the best place to learn programming !" into upper-case letters.

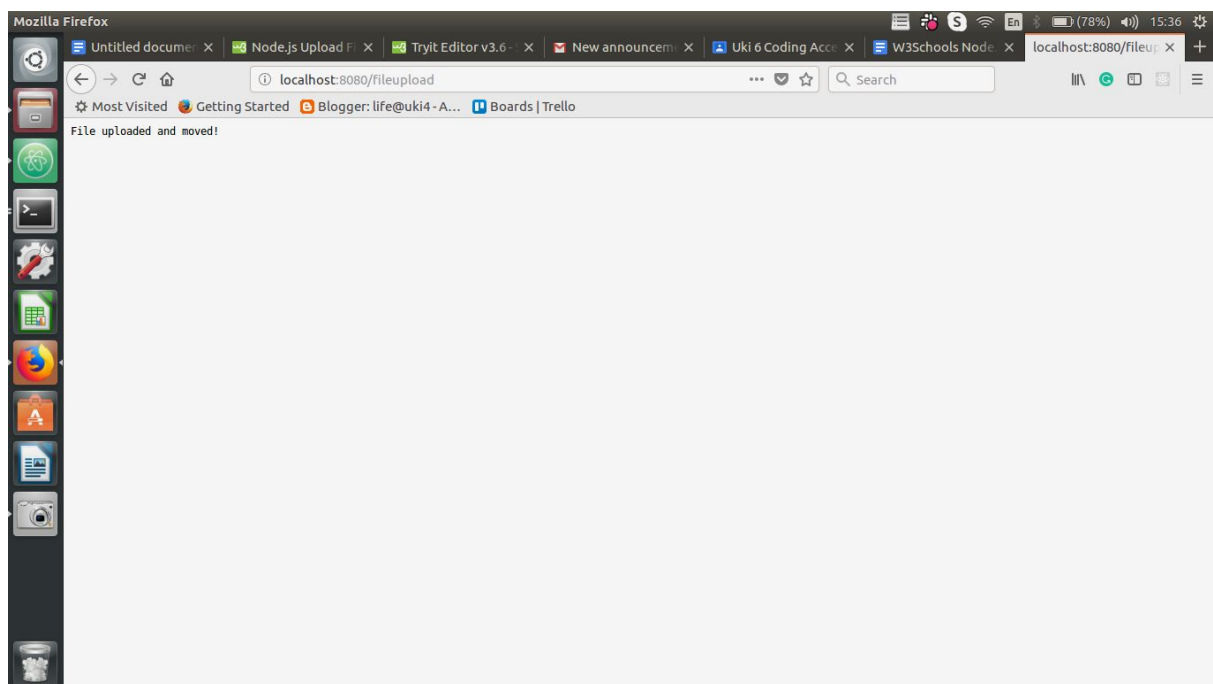


6. Create an event handler function that will say "I bark when I see strangers !" when a "bark" event is fired.

node12.js -- ~/Desktop/node -- Atom

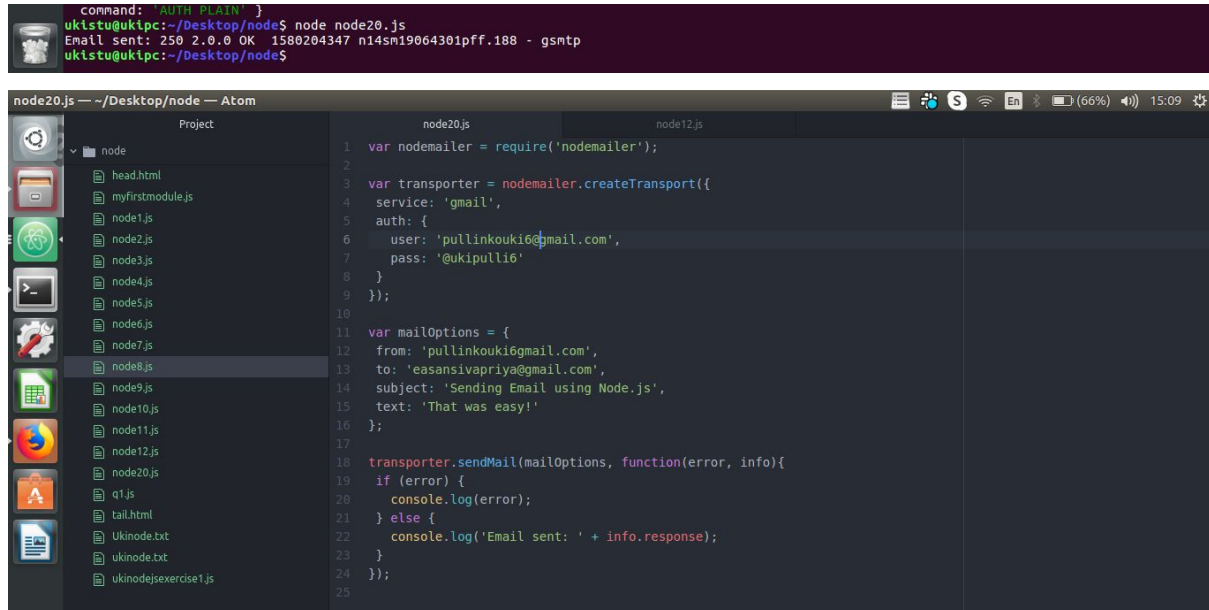
```
1 var http = require('http');
2 var formidable = require('formidable');
3 var fs = require('fs');
4
5 http.createServer(function (req, res) {
6   if (req.url == '/fileupload') {
7     var form = new formidable.IncomingForm();
8     form.parse(req, function (err, fields, files) {
9       var oldpath = files.fileupload.path;
10      var newpath = '/home/ukistu/' + files.fileupload.name;
11      fs.rename(oldpath, newpath, function (err) {
12        if (err) throw err;
13        res.write('File uploaded and moved!');
14        res.end();
15      });
16    });
17   } else {
18     res.writeHead(200, {'Content-Type': 'text/html'});
19     res.write('<form action="fileupload" method="post" enctype="multipart/form-data">');
20     res.write('<input type="file" name="fileupload"><br>');
21     res.write('<input type="submit">');
22     res.write('</form>');
23     return res.end();
24   }
25 }).listen(8080);
26
```

node12.js 8:23 LF UTF-8 JavaScript 0 files 2 updates



8. Using the Nodemailer module create a server and send a mail to info@uki.life with the subject : “Testing my nodemailer module” , text: “This is easy !”

8.1 Now instead of text send a basic html formatted mail.



The image shows a terminal window at the top and an Atom editor window below it. The terminal window displays the command `command: 'AUTH PLAIN' }` and the output `uklistu@ukipc:~/Desktop/node$ node node20.js`, followed by a successful email send confirmation: `Email sent: 250 2.0.0 OK 1580204347 n14sm19064301pff.188 - gsmt`. The Atom editor window shows the file `node20.js` open, with a file explorer on the left listing various files including `head.html`, `myfirstmodule.js`, and several `nodeX.js` files. The code in `node20.js` is as follows:

```
1 var nodemailer = require('nodemailer');
2
3 var transporter = nodemailer.createTransport({
4   service: 'gmail',
5   auth: {
6     user: 'pullinkouki6@gmail.com',
7     pass: '@ukipulli6'
8   }
9 });
10
11 var mailOptions = {
12   from: 'pullinkouki6@gmail.com',
13   to: 'easansivapriya@gmail.com',
14   subject: 'Sending Email using Node.js',
15   text: 'That was easy!'
16 };
17
18 transporter.sendMail(mailOptions, function(error, info){
19   if (error) {
20     console.log(error);
21   } else {
22     console.log('Email sent: ' + info.response);
23   }
24 });
25
```