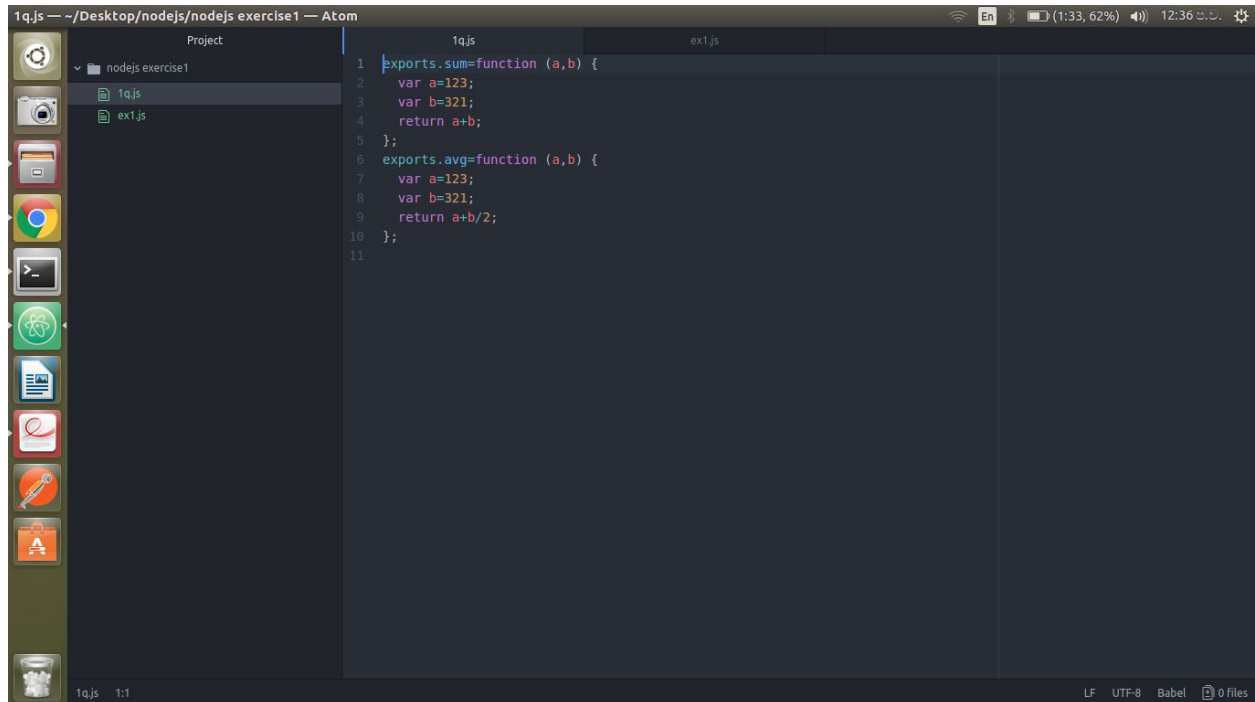


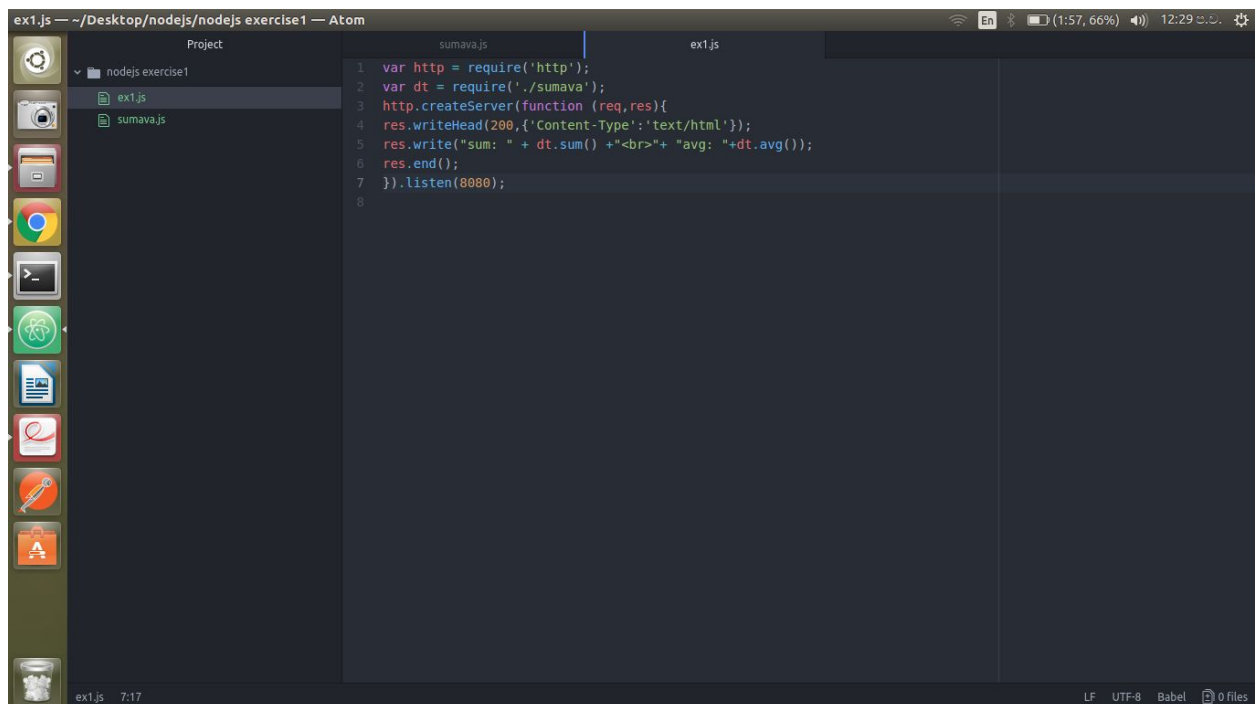
1. Create a custom module which returns the sum and average of any two numbers passed into it. Require the module and run the server by passing 123 and 321 so that the server prints out the sum and average.



The screenshot shows the Atom editor interface with a project named 'nodejs exercise1'. The file explorer on the left shows two files: '1q.js' and 'ex1.js'. The main editor window is displaying '1q.js' with the following code:

```
1 exports.sum=function (a,b) {
2   var a=123;
3   var b=321;
4   return a+b;
5 };
6 exports.avg=function (a,b) {
7   var a=123;
8   var b=321;
9   return a+b/2;
10 };
11
```

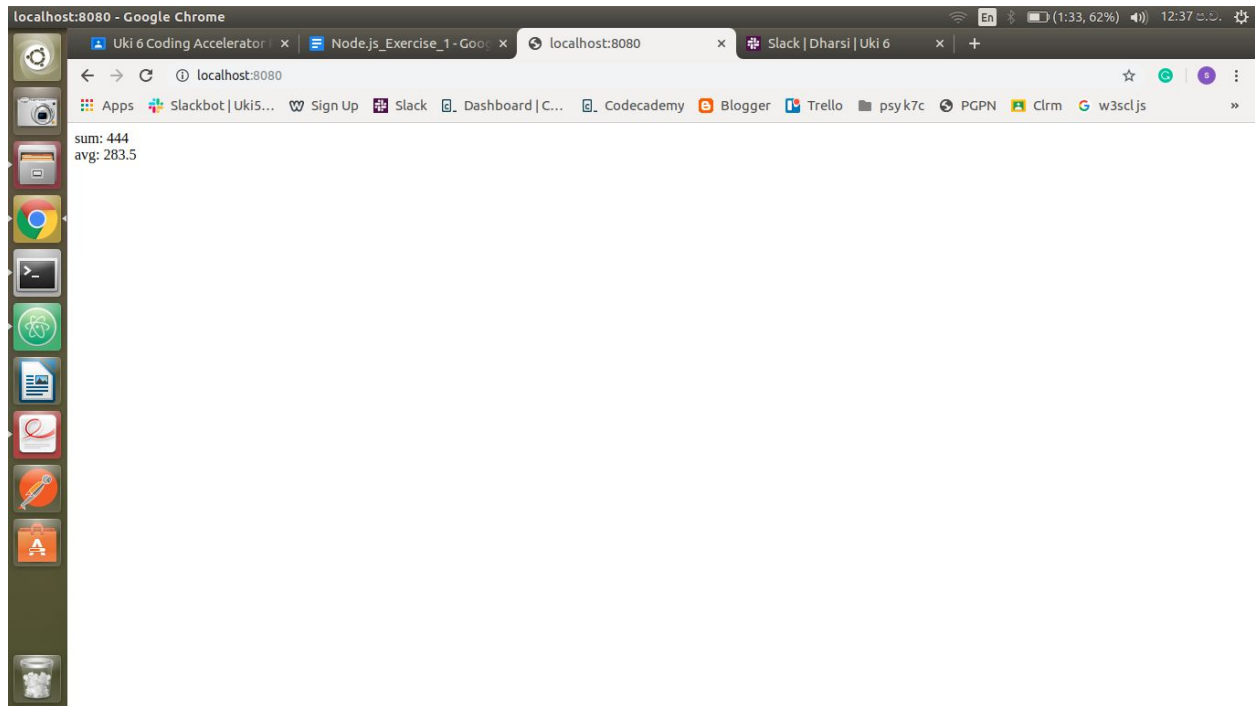
The status bar at the bottom indicates '1q.js 1:1', 'LF', 'UTF-8', 'Babel', and '0 files'.



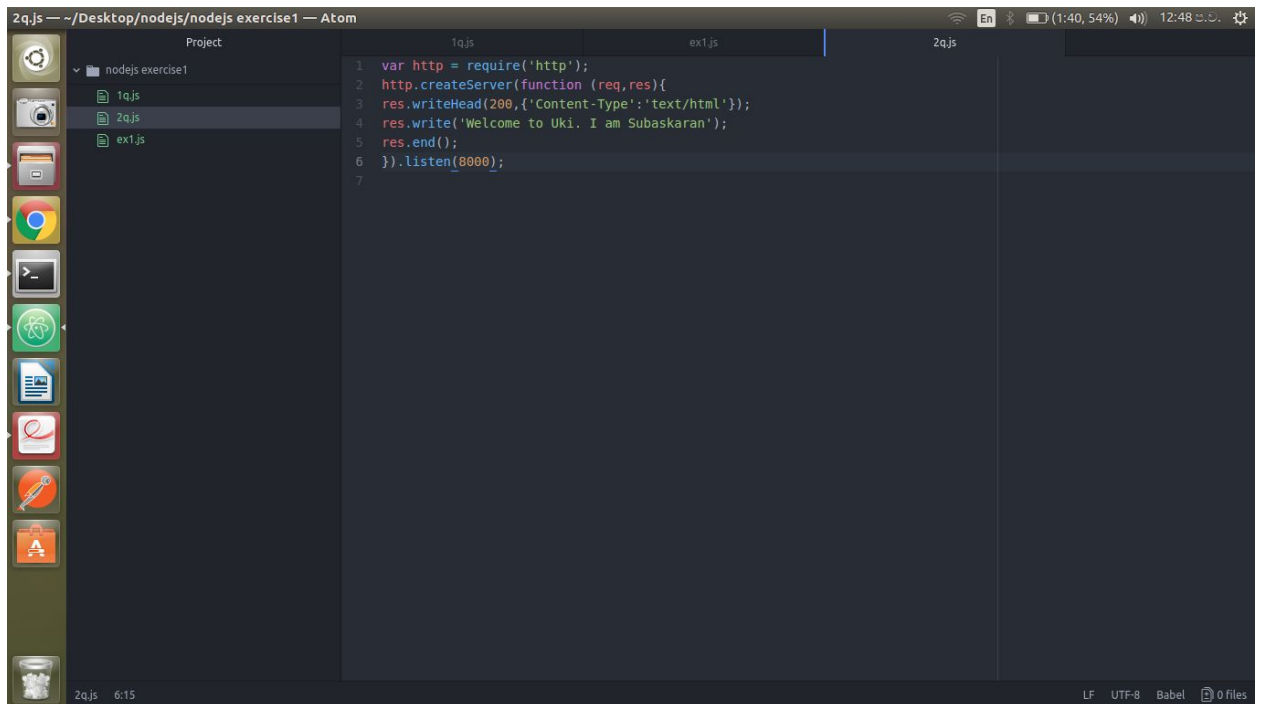
The screenshot shows the Atom editor interface with the same project. The file explorer on the left shows three files: 'ex1.js', 'sumava.js', and 'sumava.js'. The main editor window is displaying 'ex1.js' with the following code:

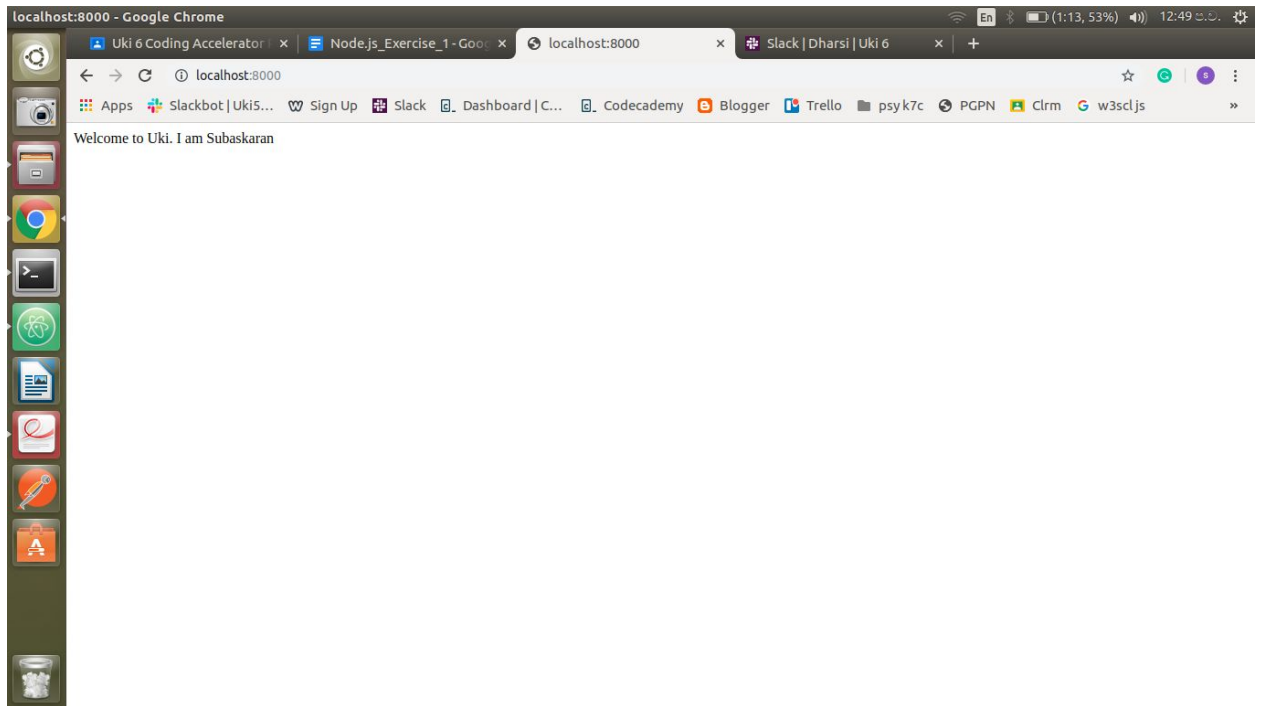
```
1 var http = require('http');
2 var dt = require('./sumava');
3 http.createServer(function (req,res){
4   res.writeHead(200,{'Content-Type':'text/html'});
5   res.write("sum: " + dt.sum() + "<br>" + "avg: " + dt.avg());
6   res.end();
7 }).listen(8080);
8
```

The status bar at the bottom indicates 'ex1.js 7:17', 'LF', 'UTF-8', 'Babel', and '0 files'.



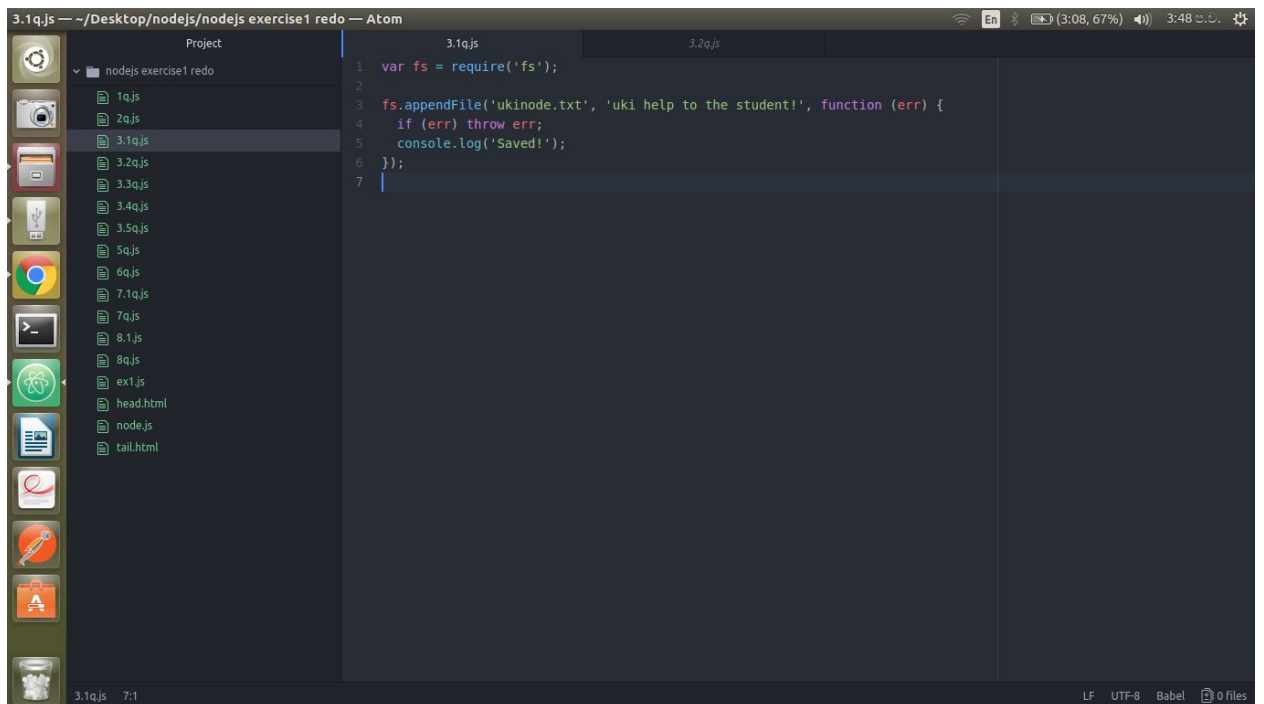
2. Create a simple http server and print “Welcome to Uki. I am **yourname**” when a request is sent to your server via the port 8000. (Note - Change different port numbers and check)

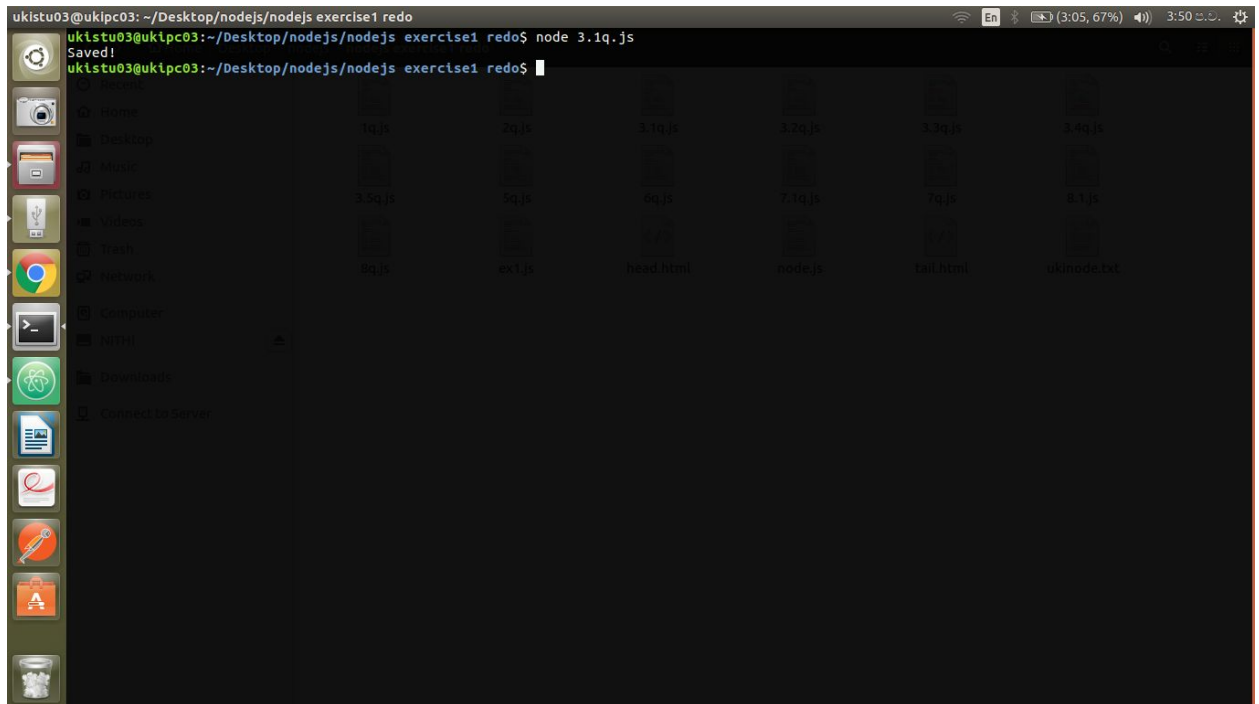




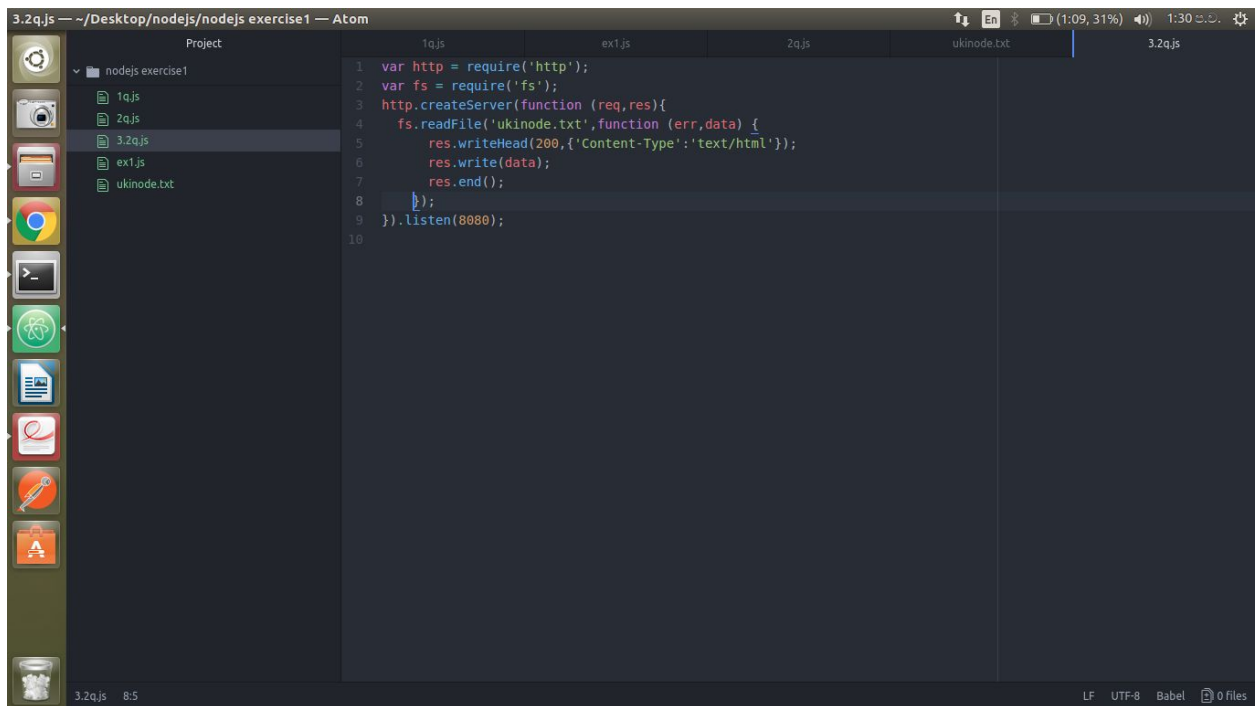
3. Using the file system module create a new file called ukinode.txt

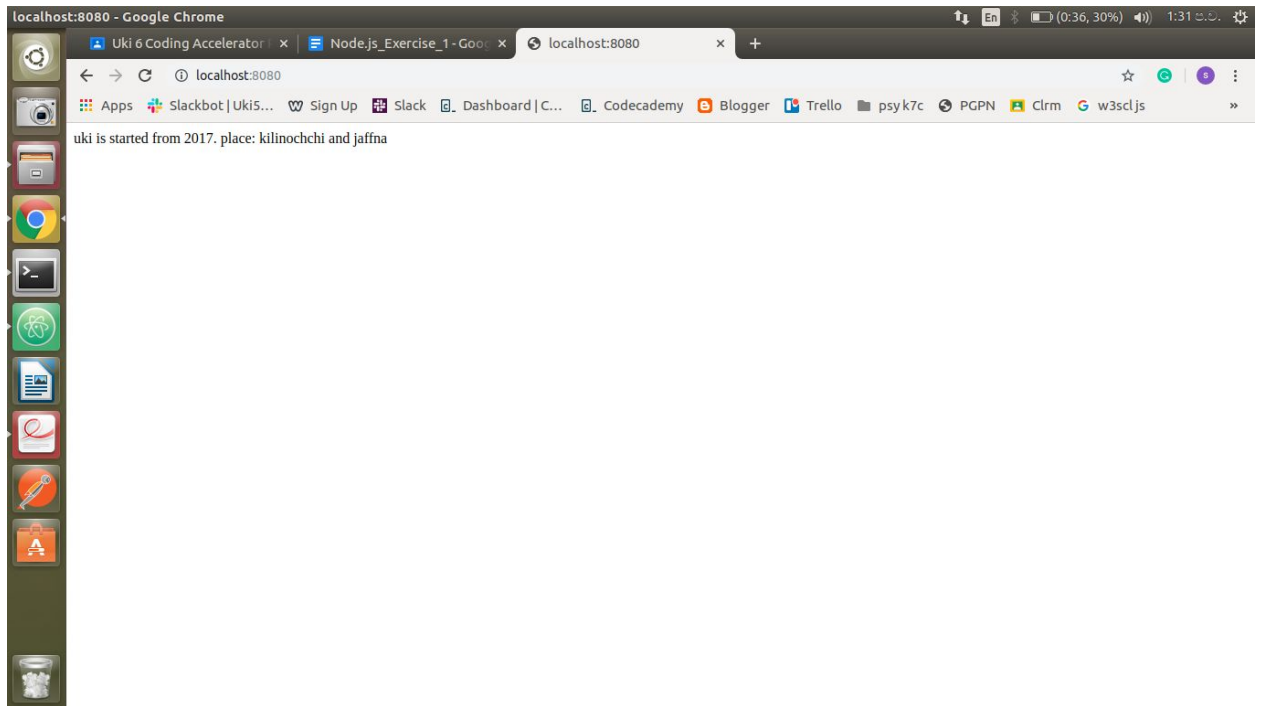
3.1 Write a paragraph about Uki into that file



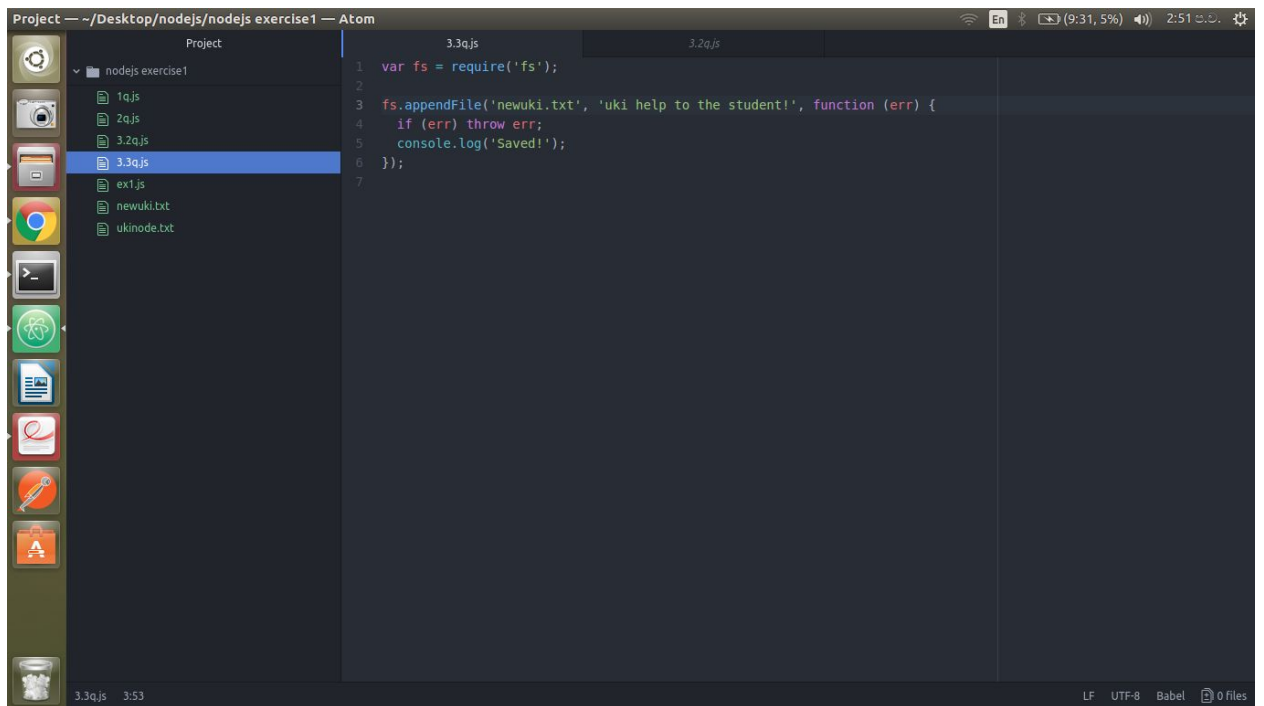


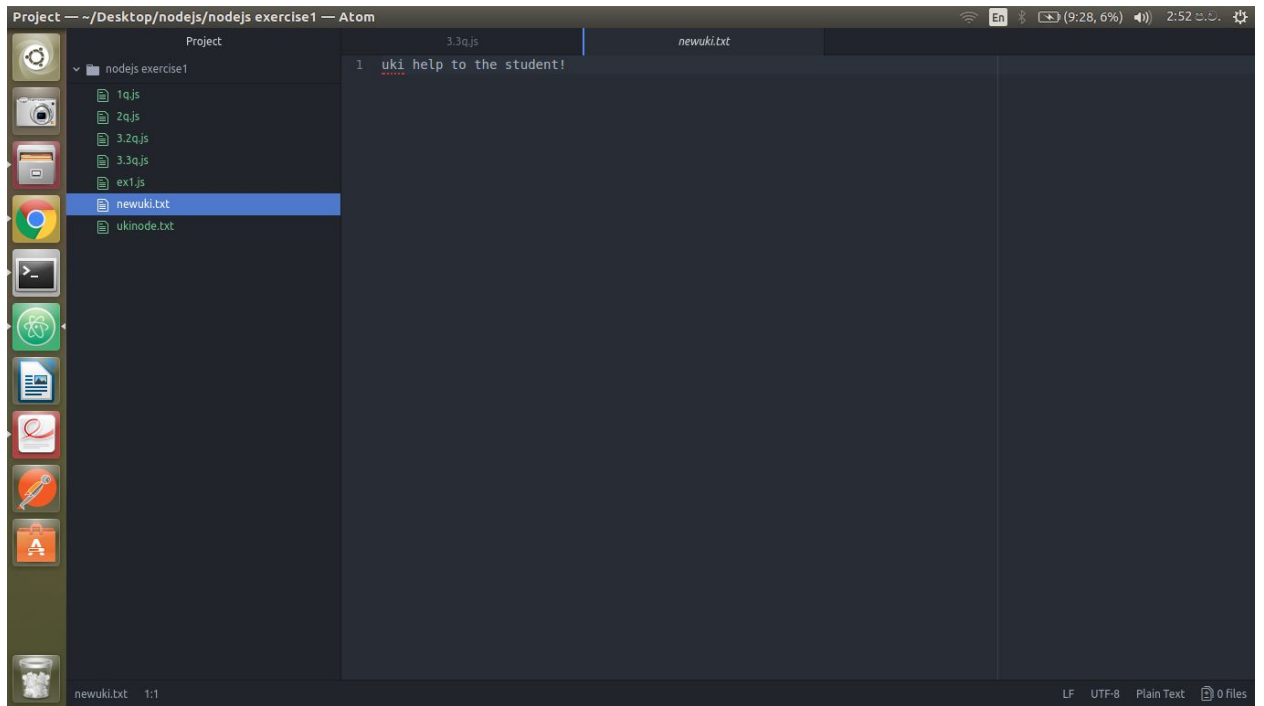
### 3.2 Serve that file to the client (Read File) over your server



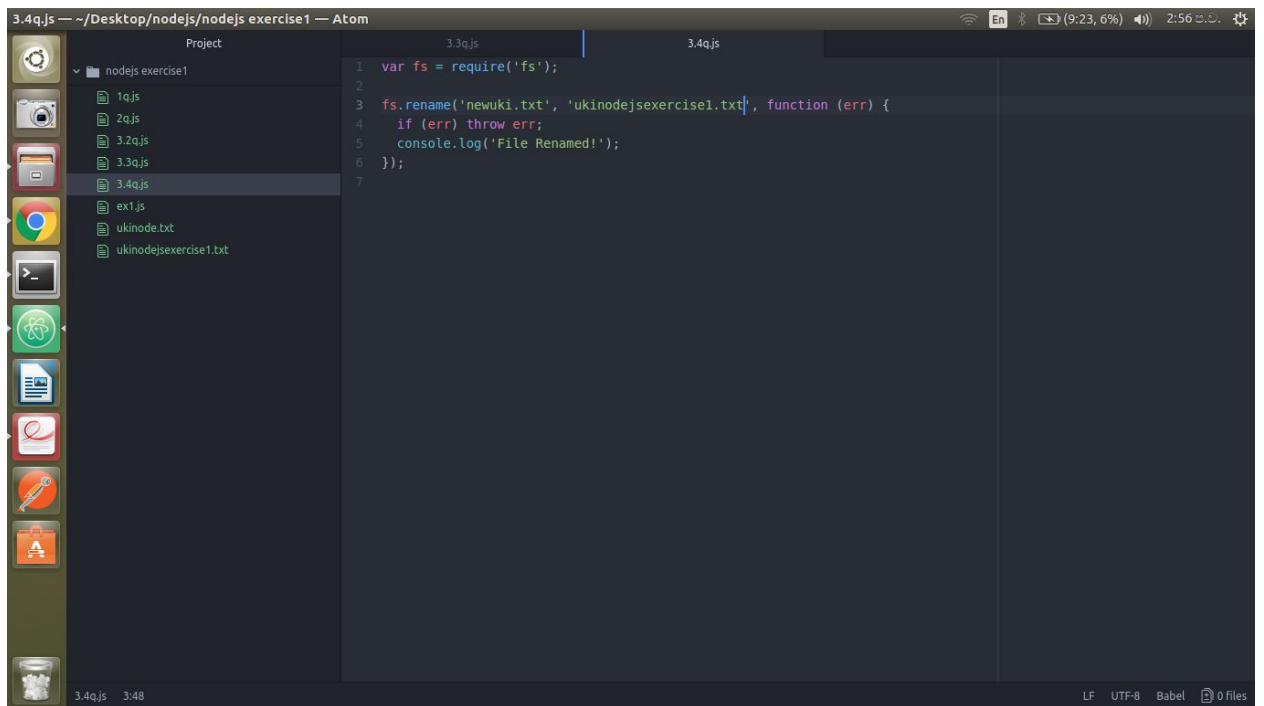


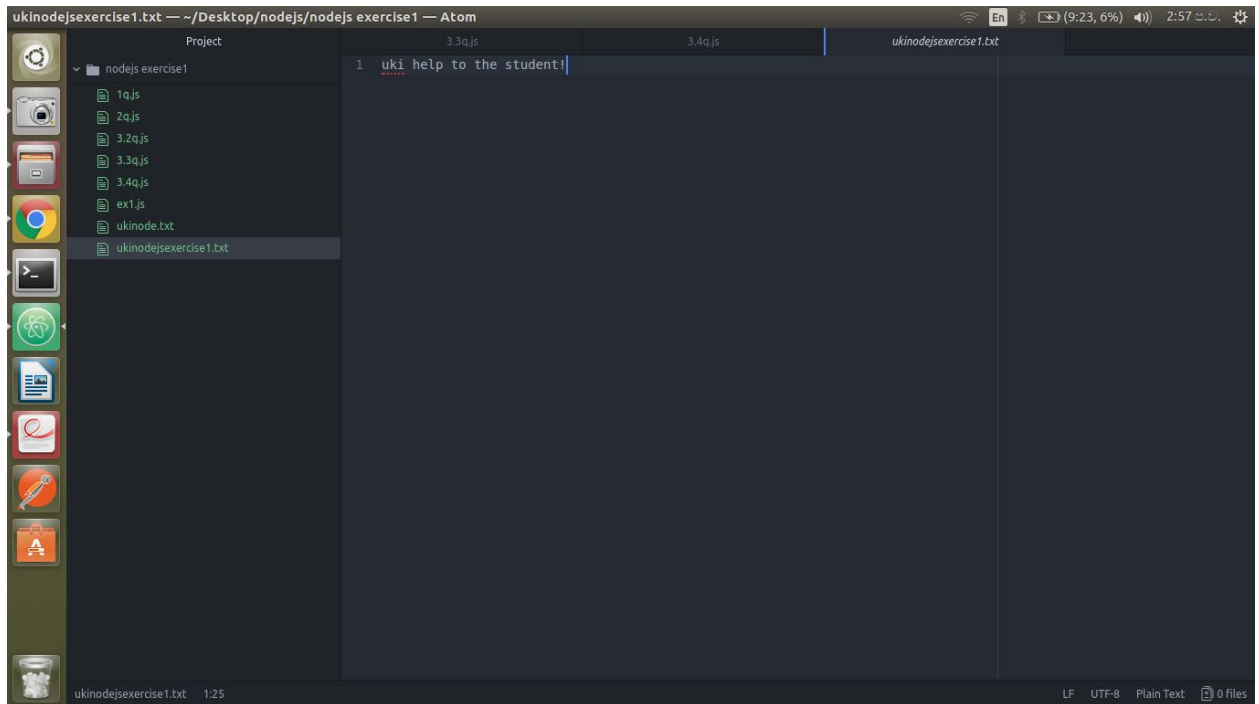
### 3.3 Append another paragraph about Uki and now serve the new file



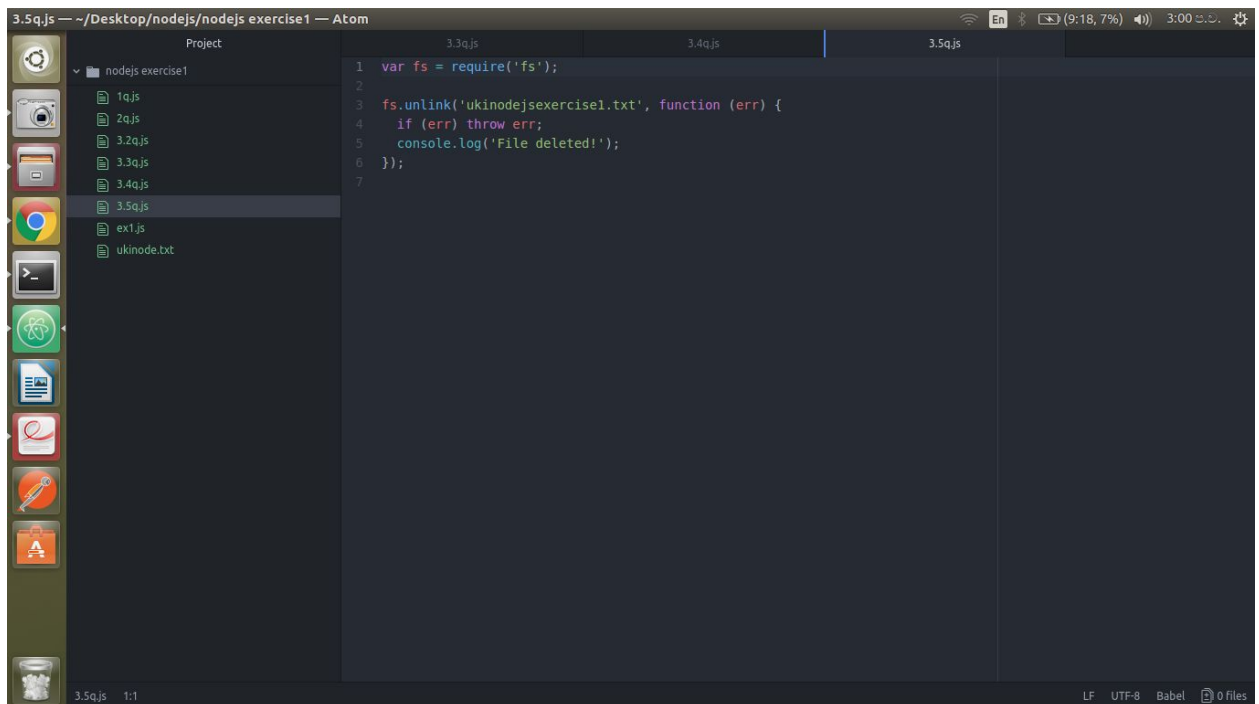


### 3.4 Rename the file as ukinodejsexercise1.txt





### 3.5 Delete the file you created

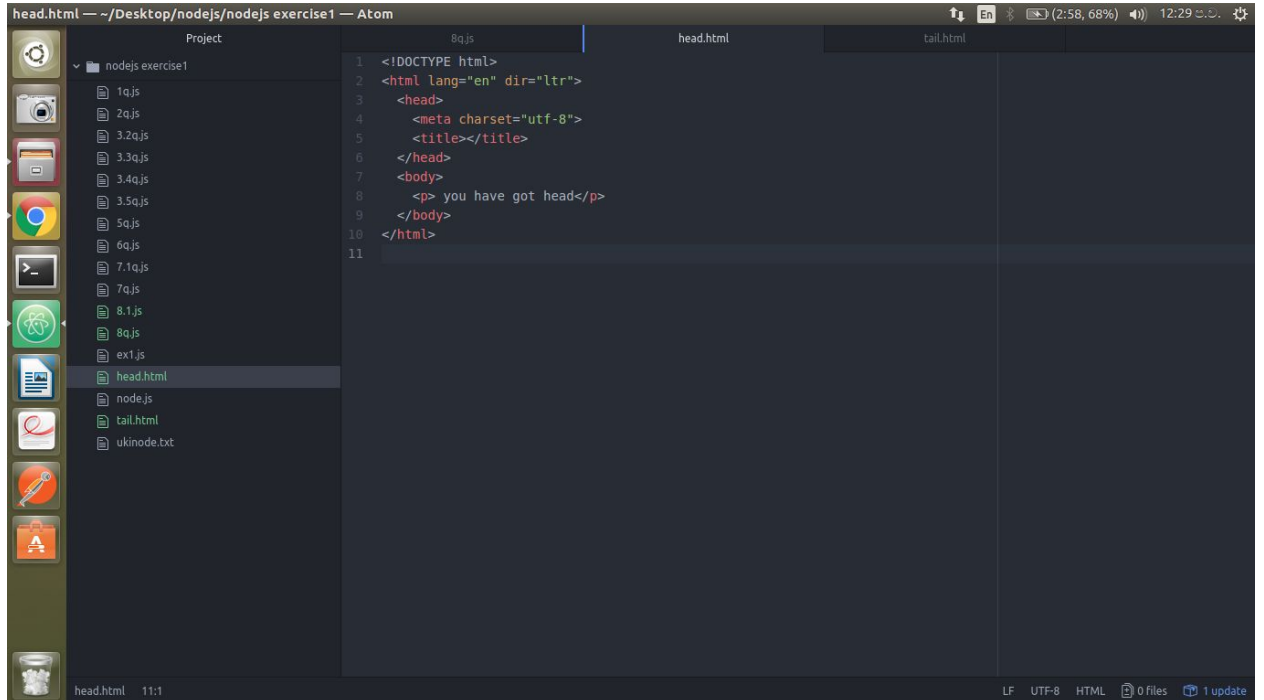


4. Create two html files called head.html which is a web page which says 'you have got head' and tail.html which is a web page which says 'you have got tail' and save them in the same folder as your node.js files. Create a Node.js file that opens the requested file and returns the content to the client.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

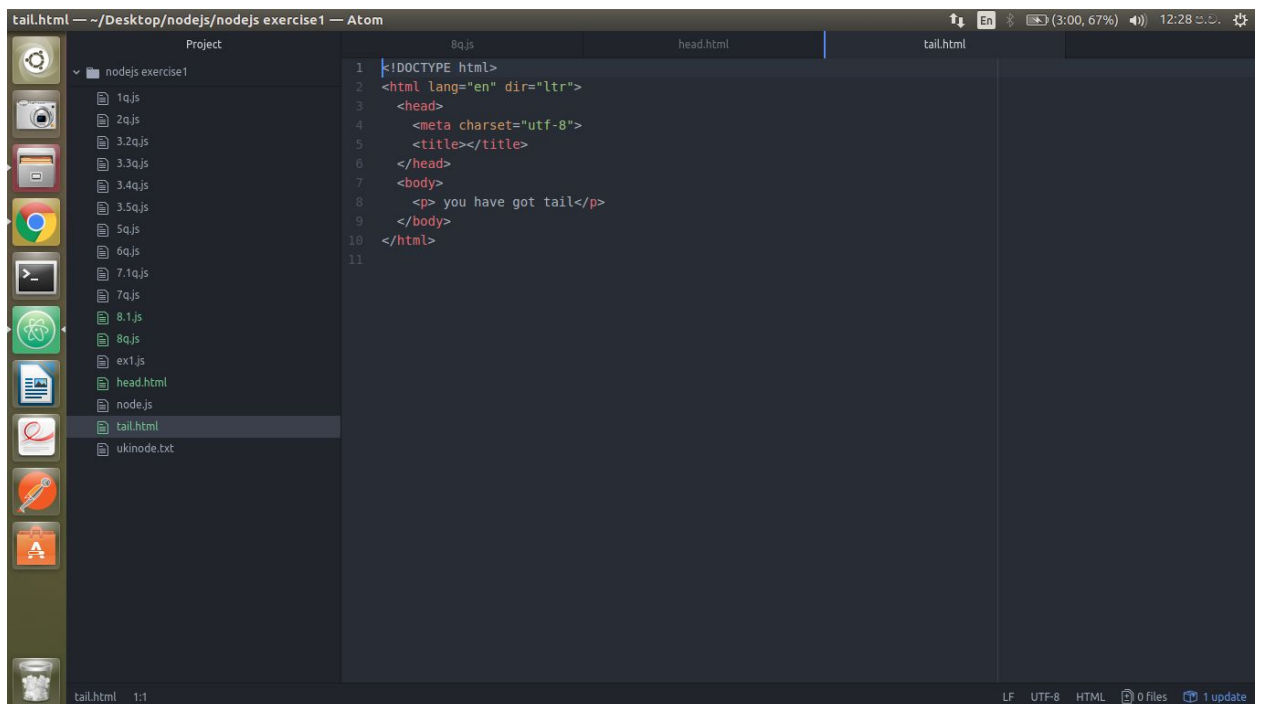
<http://localhost:8080/head.html> -> You have got head

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



The screenshot shows the Atom editor interface with the project 'nodejs exercise1' open. The file explorer on the left lists files from 1.q.js to ukinode.txt, with 'head.html' selected. The main editor pane displays the content of 'head.html', which is an HTML document with a head section containing a meta charset and a title, and a body section with the text 'you have got head'. The status bar at the bottom indicates the current file is 'head.html' at line 11:1, with LF line endings, UTF-8 encoding, and HTML syntax.

```
1 <!DOCTYPE html>
2 <html lang="en" dir="ltr">
3   <head>
4     <meta charset="utf-8">
5     <title></title>
6   </head>
7   <body>
8     <p> you have got head</p>
9   </body>
10 </html>
11
```



The screenshot shows the Atom editor interface with the project 'nodejs exercise1' open. The file explorer on the left lists files from 1.q.js to ukinode.txt, with 'tail.html' selected. The main editor pane displays the content of 'tail.html', which is an HTML document with a head section containing a meta charset and a title, and a body section with the text 'you have got tail'. The status bar at the bottom indicates the current file is 'tail.html' at line 1:1, with LF line endings, UTF-8 encoding, and HTML syntax.

```
1 <!DOCTYPE html>
2 <html lang="en" dir="ltr">
3   <head>
4     <meta charset="utf-8">
5     <title></title>
6   </head>
7   <body>
8     <p> you have got tail</p>
9   </body>
10 </html>
11
```



node.js -- ~/Desktop/nodejs/nodejs exercise1 — Atom

Project

nodejs exercise1

- 1q.js
- 2q.js
- 3.2q.js
- 3.3q.js
- 3.4q.js
- 3.5q.js
- ex1.js
- head.html
- node.js
- tail.html
- ukinode.txt

node.js

head.html

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

node.js 7:23 LF UTF-8 Babel 0 files

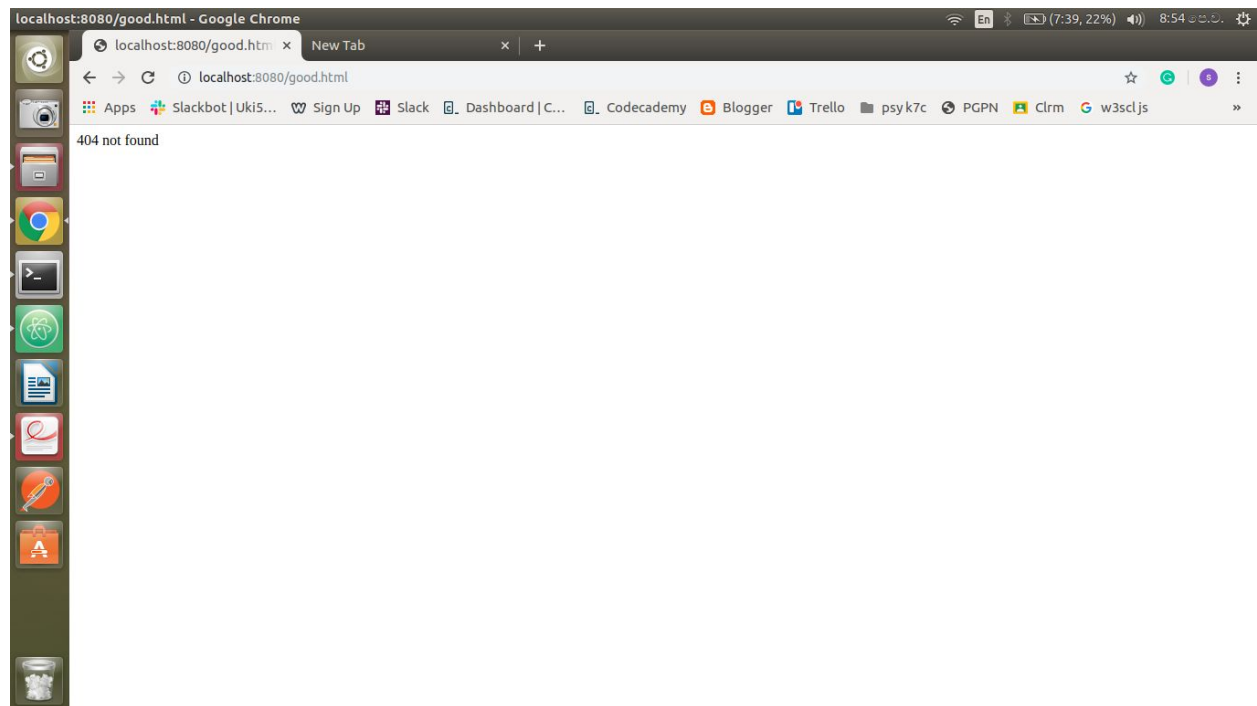
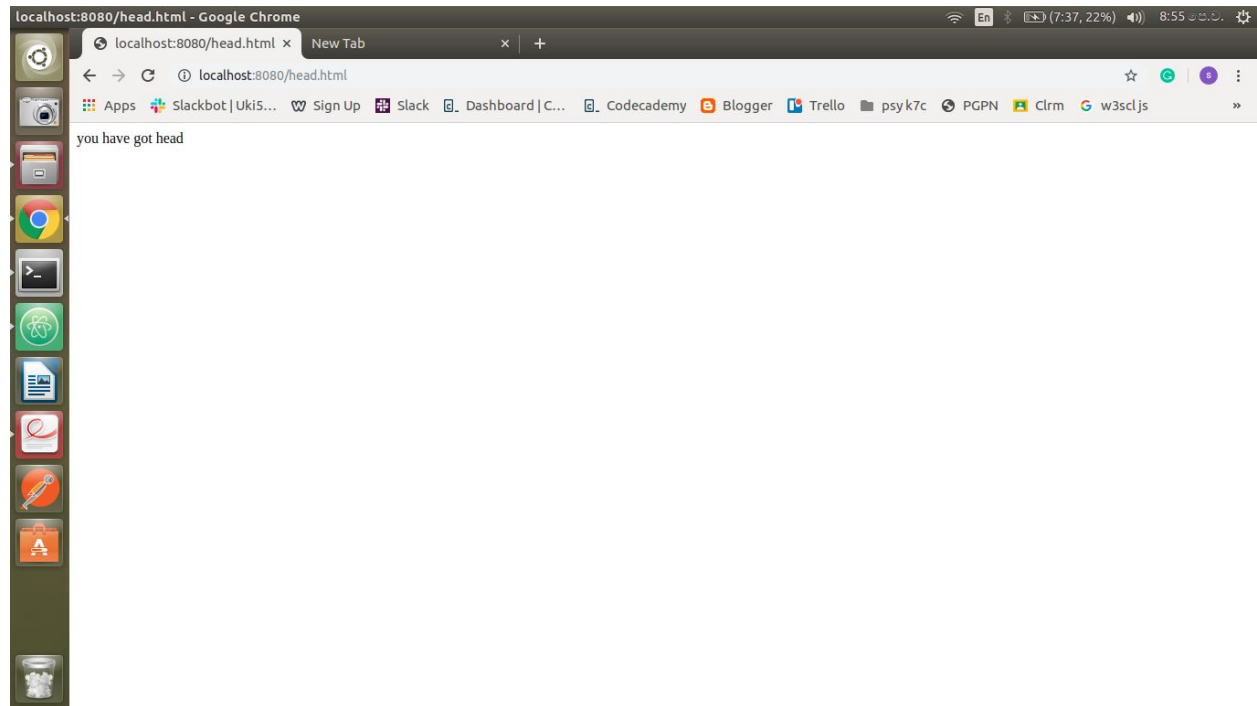
localhost:8080/tail.html - Google Chrome

localhost:8080/tail.html x New Tab x +

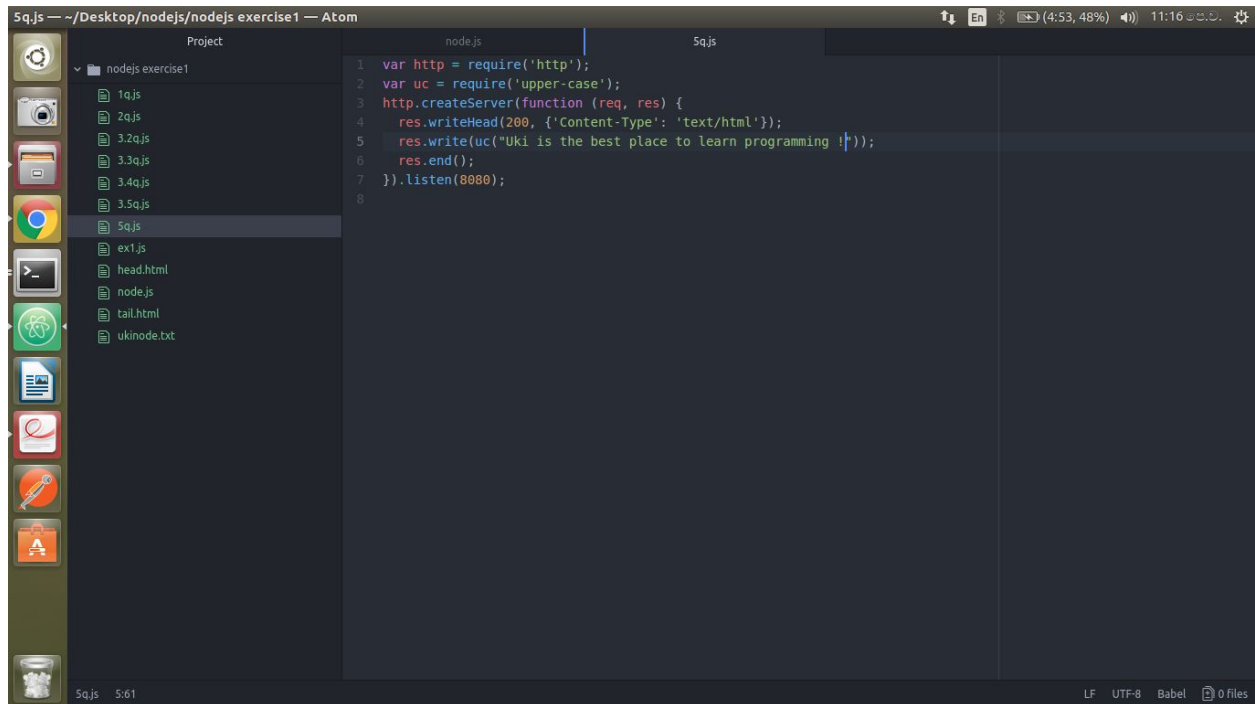
localhost:8080/tail.html

Apps Slackbot | UkiS... Sign Up Slack Dashboard | C... Codecademy Blogger Trello psy k7c PGP Clrm w3scljs

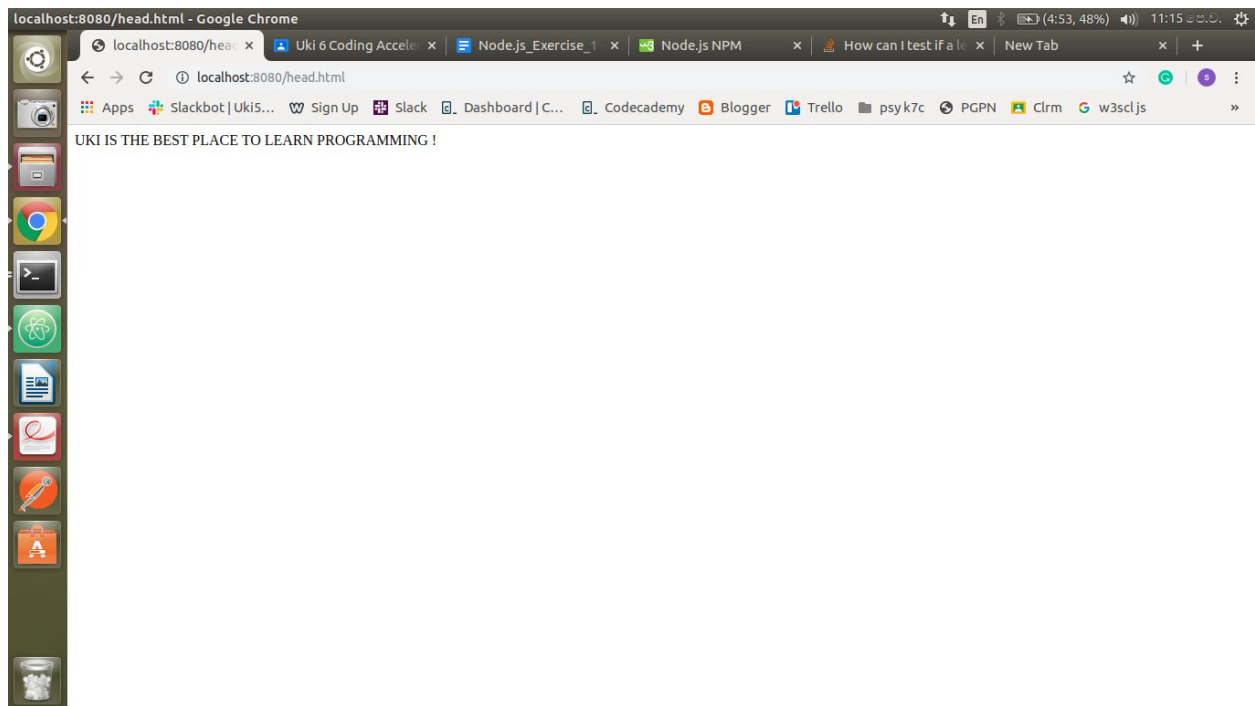
you have got tail



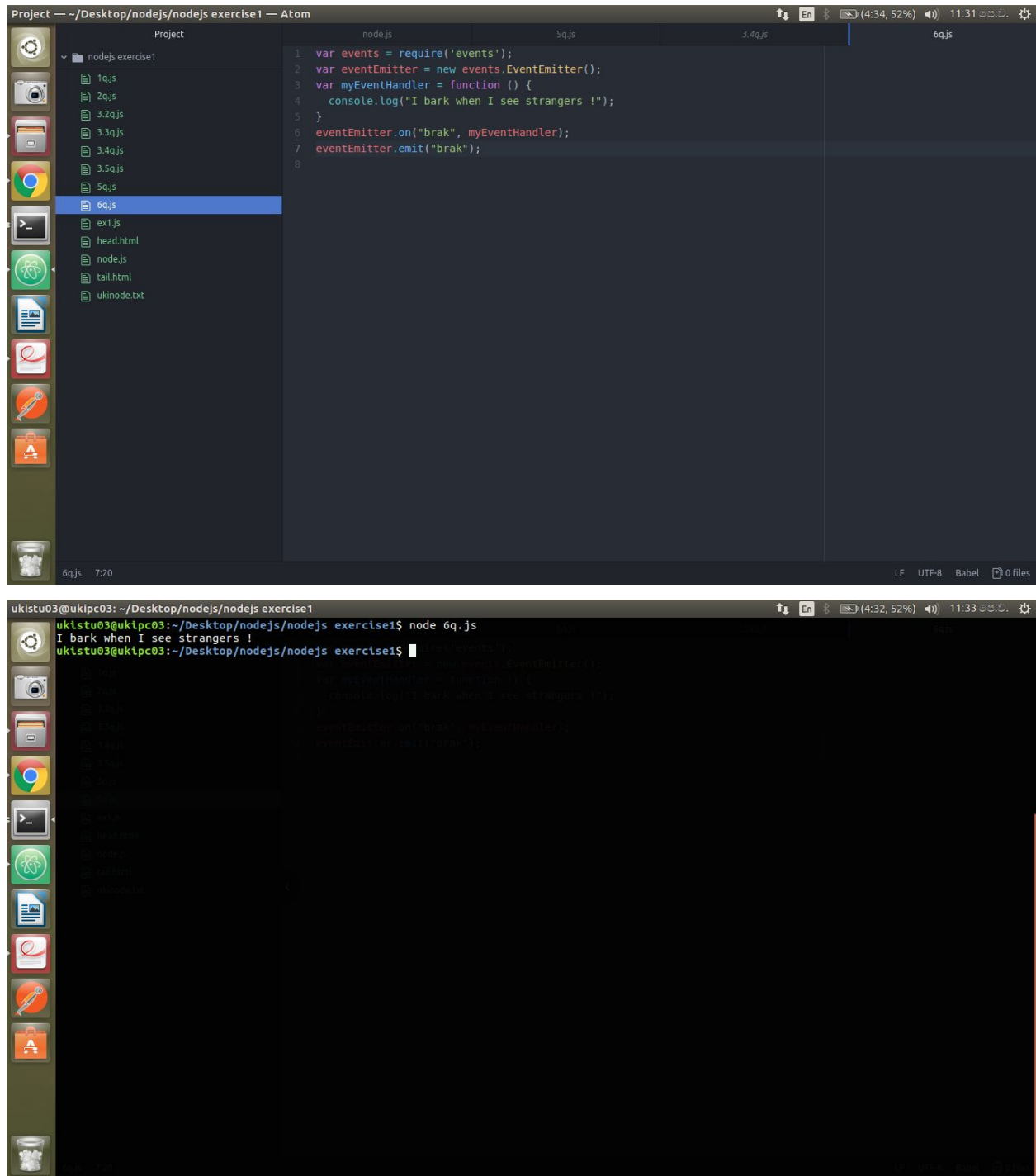
5. Install the package “upper-case” using NPM and create a Node.js file that will convert the output "Uki is the best place to learn programming !" into upper-case letters.



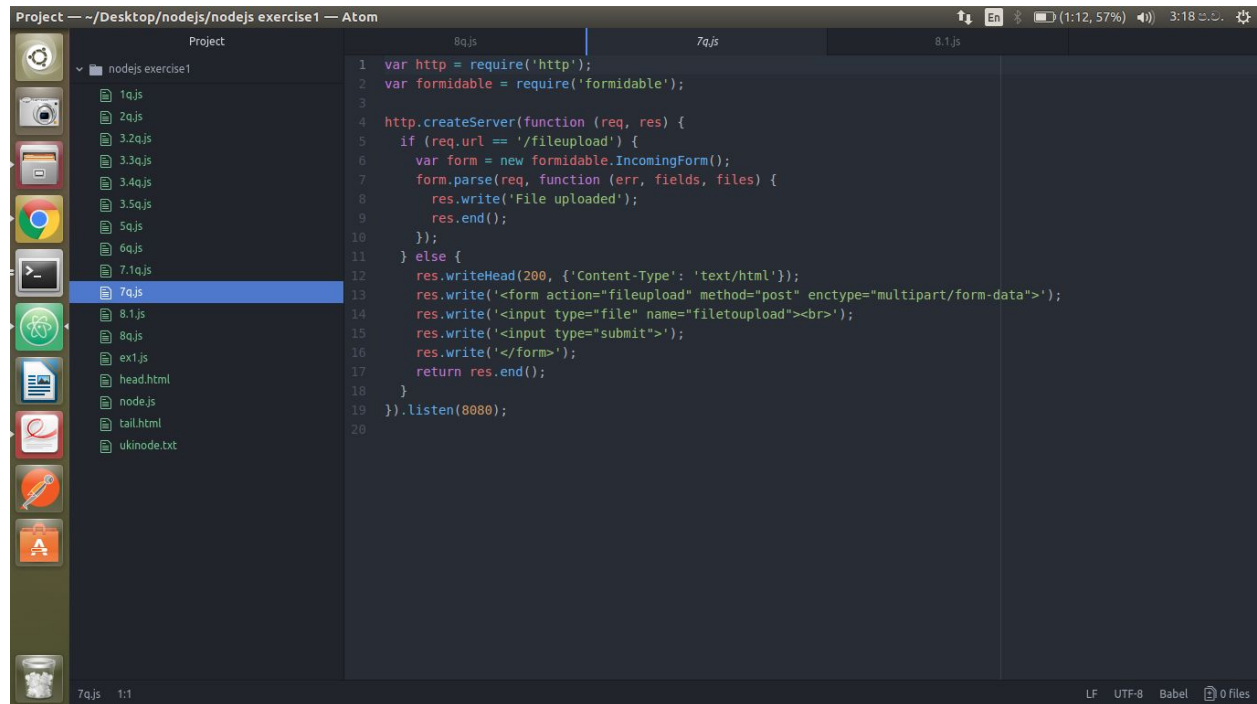
```
1 var http = require('http');
2 var uc = require('upper-case');
3 http.createServer(function (req, res) {
4   res.writeHead(200, {'Content-Type': 'text/html'});
5   res.write(uc("Uki is the best place to learn programming !"));
6   res.end();
7 }).listen(8080);
8
```



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

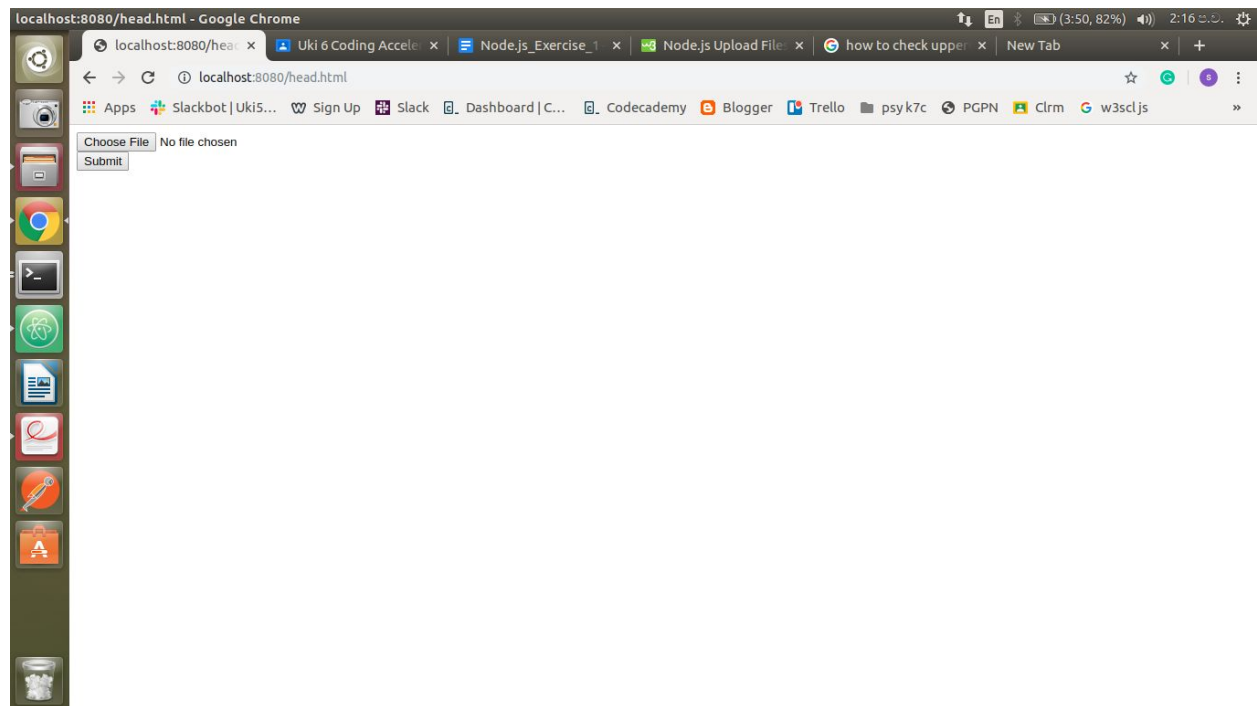


7. Install “formidable” module using npm and make a web page in Node.js that lets the user upload files to your computer.

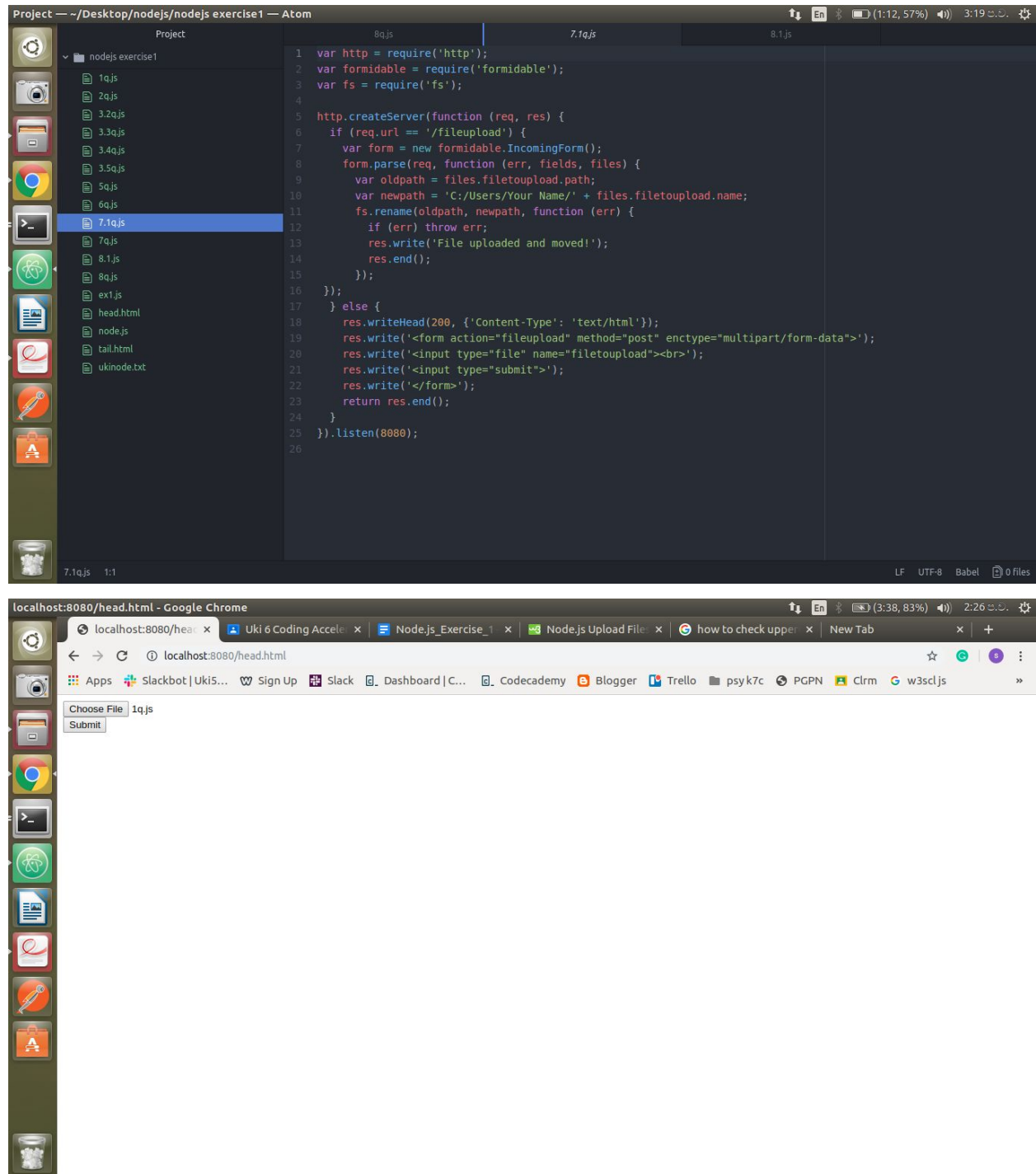


The screenshot shows the Atom code editor with a project named "nodejs exercise1". The file explorer on the left lists several files: 1q.js, 2q.js, 3.2q.js, 3.3q.js, 3.4q.js, 3.5q.js, 5q.js, 6q.js, 7.1q.js, 7q.js (selected), 8.1.js, 8q.js, ex1.js, head.html, node.js, tail.html, and ukinode.txt. The main editor area displays the code in 7q.js:

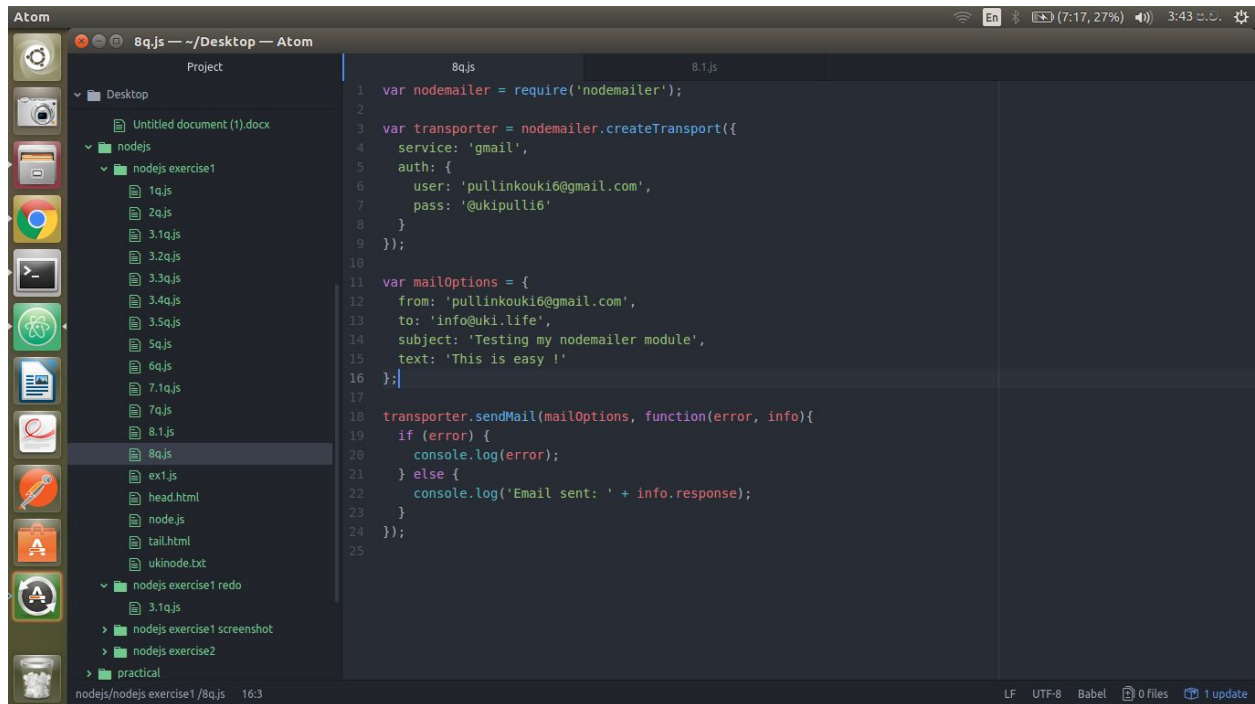
```
1 var http = require('http');
2 var formidable = require('formidable');
3
4 http.createServer(function (req, res) {
5   if (req.url == '/fileupload') {
6     var form = new formidable.IncomingForm();
7     form.parse(req, function (err, fields, files) {
8       res.write('File uploaded');
9     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



7.1 Save that uploaded file into your Documents directory.

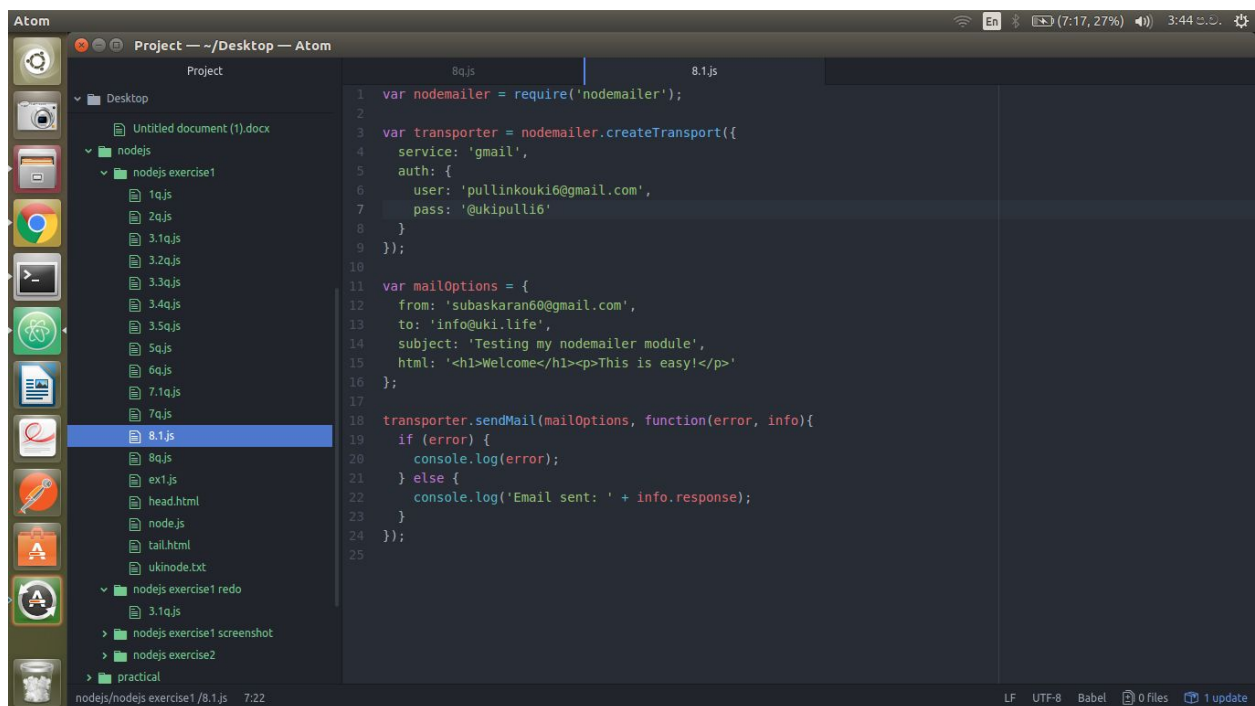


8. Using the Nodemailer module create a server and send a mail to `info@uki.life` with the subject : "Testing my nodemailer module" , text: "This is easy !"



```
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: 'info@uki.life',
14   subject: 'Testing my nodemailer module',
15   text: 'This is 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
```

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



```
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: 'subaskaran60@gmail.com',
13   to: 'info@uki.life',
14   subject: 'Testing my nodemailer module',
15   html: '<h1>Welcome</h1><p>This is easy!</p>'
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
```