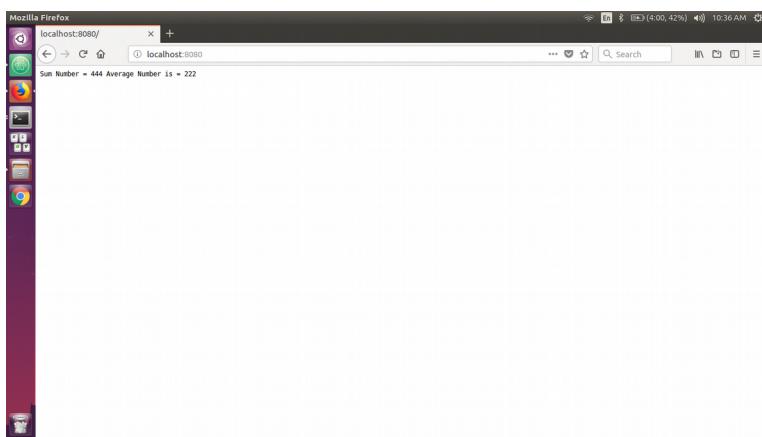


1) Create a custom module which returns the sum and average of any two numbers passed into it. Require the module and run the server by passing 123 and 321 so that the server prints out the sum and average.

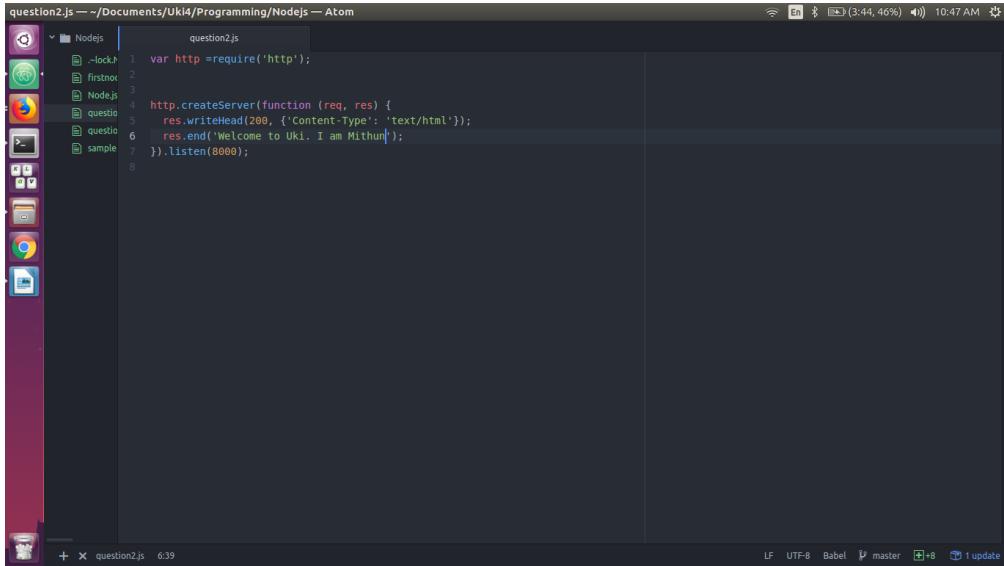
The screenshot shows the Atom code editor interface. On the left, there's a sidebar titled 'Nodes' containing project files: 'question1.js', 'sample.js', 'firstno', 'backap', 'second', 'sample', and 'sumfun'. The main editor area has two tabs: 'question1.js' (active) and 'sample.js'. The 'question1.js' tab contains the following code:question1.js
```js
var http = require('http');
var sample = require('./sample.js');

http.createServer(function (req, res) {
 res.writeHead(200, {'Content-Type': 'text/plain'});
 res.write("Sum Number = " + sample.sumnum(123, 321));
 res.write("Average Number is = " + sample.average(123,321));
 res.end();
}).listen(8080);
```
The 'sample.js' tab contains the following code:sample.js
```js
exports.sumnum = function (a,b){
 return a+b ;
}
exports.average = function (a,b){
 return (a+b)/2;
}
```
Status bar at the bottom indicates: LF, UTF-8, Babel, master, 4+0, 1 update.

The screenshot shows the Atom code editor interface. The sidebar on the left shows 'Nodes' with files: 'question1.js', 'sample.js', 'firstno', 'backap', 'second', 'sample', and 'sumfun'. The main editor area shows the 'sample.js' file with the same code as in the previous screenshot. Status bar at the bottom indicates: LF, UTF-8, Babel, master, 4+0, 1 update.



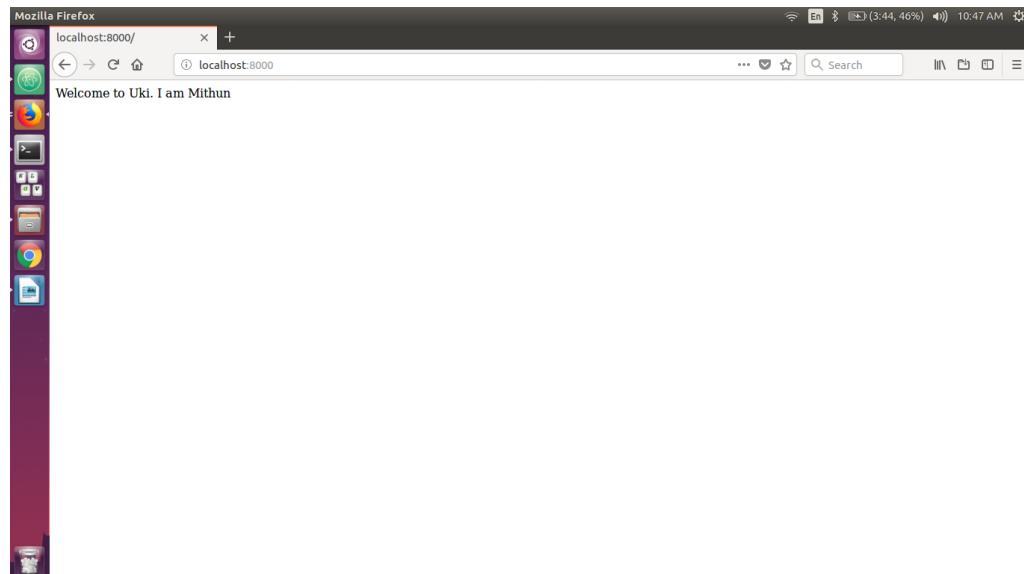
2) Create a simple http server and print “Welcome to Uki. I am **yourname**” when a request is sent to your server via the port 8000. (Note - Change different port numbers and check)



The screenshot shows the Atom code editor interface. The left sidebar displays a file tree with a folder named 'Nodejs' containing several files: '-lock.h', 'firstno', 'Node.js', 'questio', 'questio', and 'sample'. The main editor area contains the following code:

```
question2.js — ~/Documents/Uki4/Programming/Nodejs — Atom
question2.js
var http = require('http');

http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.end('Welcome to Uki. I am Mithun');
}).listen(8000);
```



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

3.1 Write a paragraph about Uki into that file

A screenshot of a Linux desktop environment. On the left, there's a vertical dock with icons for various applications like a file manager, terminal, and browser. The main window shows a terminal session with the following content:

```
question31.js -- ~/Documents/Uk14/Programming/Nodejs -- Atom
okistu0@uki:~/Documents/Uk14/Programming/Nodejs$ node question31.js
[nodemon] 1.14.11
[nodemon] to restart at any time, enter `r`
[nodemon] watching: *.*
[nodemon] starting mode question31.js
Saved!
[nodemon] clean exit - waiting for changes before restart
```

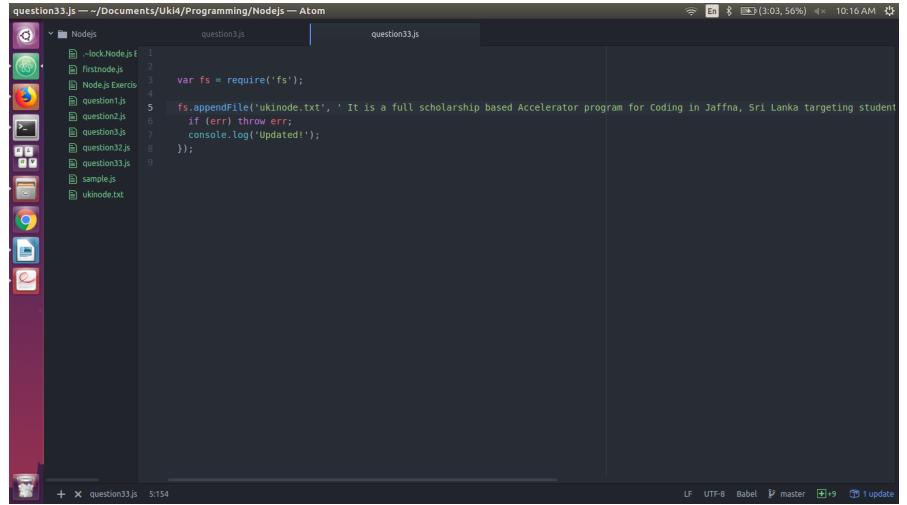
The terminal window has tabs for 'question31.js' and 'question31.js'. At the bottom of the screen, there's a dock with icons for the terminal, file manager, and browser.

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

The screenshot shows a Linux desktop environment with a dark theme. In the top left, there's a terminal window titled 'question3.js' containing Node.js code. In the top right, a Mozilla Firefox browser window is open, showing the URL 'localhost:8000/' and the page content 'Hello World Welcome to uki.'. Below the browser is a dock with various application icons. In the bottom left, there's another terminal window titled 'question3.js' with the same code as the first one. The desktop background is a purple gradient.

This screenshot shows a terminal window on a Linux system with a light-colored background. The window title is 'ukistu09@ukipc09: ~/Documents/Uki4/Programming/Nodejs'. The terminal output shows a Node.js application running under nodemon. It starts by printing 'Hello World Welcome to uki. It is a full'. Then it throws a SyntaxError: Invalid or unexpected token at line 7, which is 'res.write('Hello World Welcome to uki. It is a full'. The stack trace for this error is provided. After the crash, nodemon restarts the application multiple times, each time printing 'starting node question3.js'. The desktop environment includes a dock with icons for various applications like a file manager, terminal, and browser.

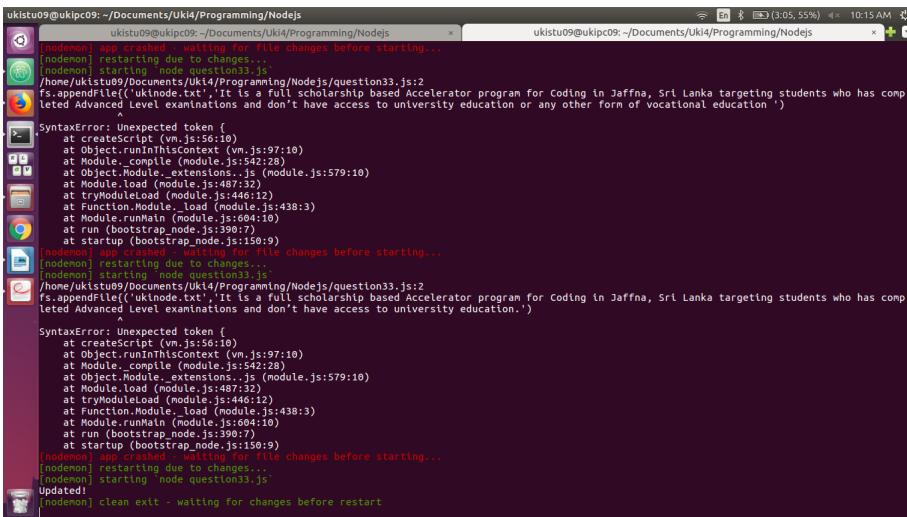
3.3 Append another paragraph about Uki and now serve the new file



A screenshot of the Atom code editor. The left sidebar shows a file tree with several Node.js files like lockNode.js, firstnode.js, Node.js Exercis, question1.js, question2.js, question3.js, question32.js, question33.js, sample.js, and ukinode.txt. The main editor area contains the following code:

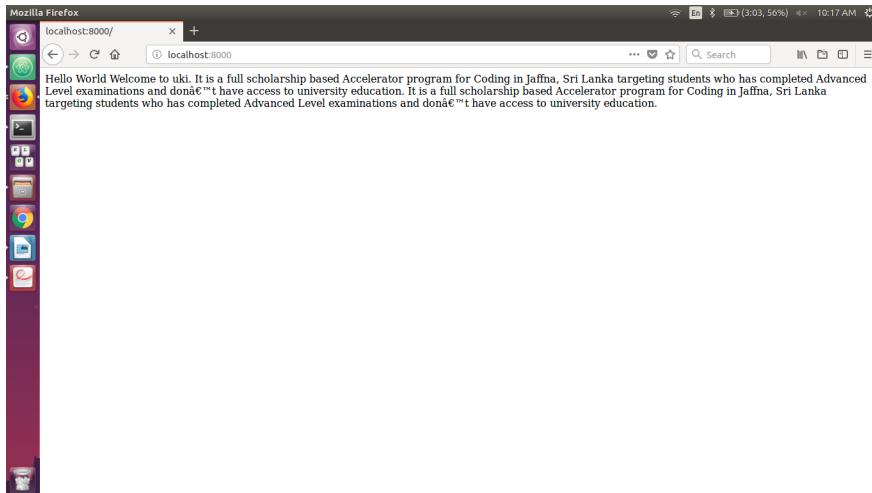
```
question33.js — ~/Documents/Uki4/Programming/Nodejs — Atom
question33.js
1
2
3
4
5
6
7
8
9
```

```
var fs = require('fs');
fs.appendFile('ukinode.txt', 'It is a full scholarship based Accelerator program for Coding in Jaffna, Sri Lanka targeting student
if (err) throw err;
console.log('Updated!');");
});
```

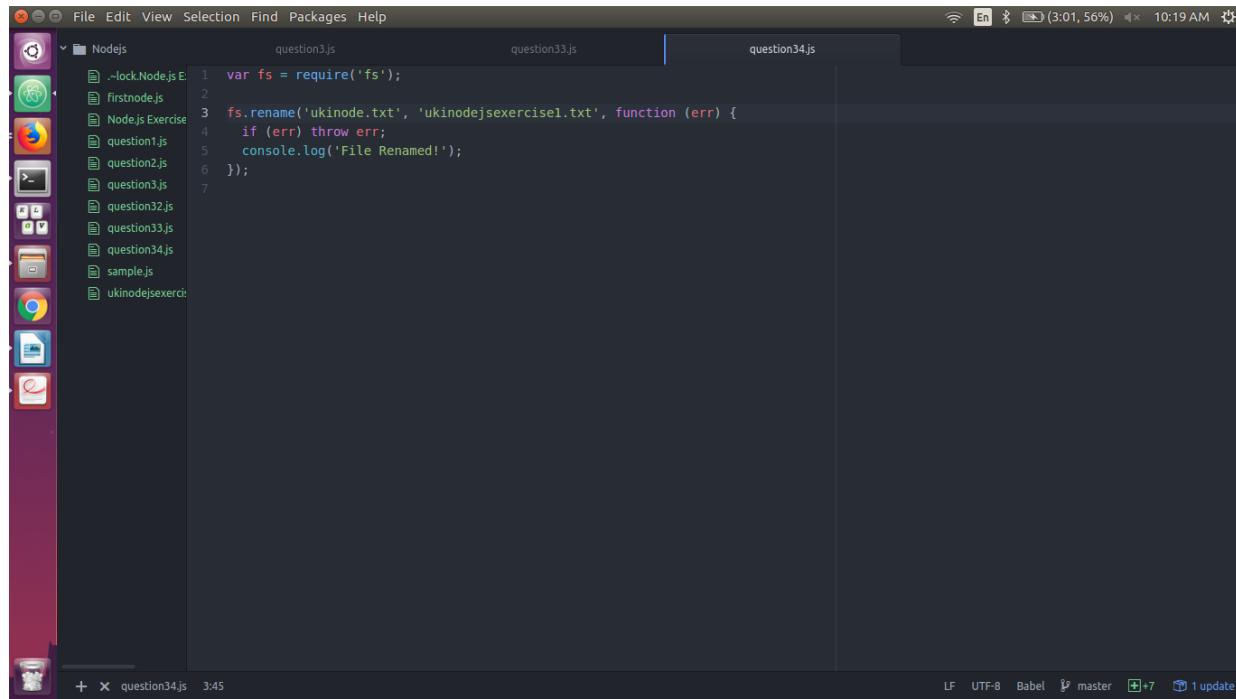


A screenshot of a terminal window titled 'node question33.js'. The output shows the file being appended to 'ukinode.txt' and then the file being served via node.js.

```
[node] app crashed - waiting for file changes before starting...
[nodemon] restarting due to changes...
[nodemon] starting [node] question33.js
/home/ukistu09/Documents/Uki4/Programming/Nodejs/question33.js:2
fs.appendfile('ukinode.txt','It is a full scholarship based Accelerator program for Coding in Jaffna, Sri Lanka targeting students who has comp
leted Advanced Level examinations and don't have access to university education or any other form of vocational education ')
SyntaxError: Unexpected token {
  at createScript (vm.js:56:10)
  at Object.runInThisContext (vm.js:97:10)
  at Module._compile (module.js:542:28)
  at Object.Module._extensions..js (module.js:579:10)
  at Module.load (module.js:487:32)
  at tryModuleLoad (module.js:446:12)
  at Function.Module.load (module.js:438:3)
  at Module.runMain (module.js:604:10)
  at run (bootstrap_node.js:390:7)
  at startup (bootstrap_node.js:150:9)
[nodemon] app crashed - waiting for file changes before starting...
[nodemon] restarting due to changes...
[nodemon] starting [node] question33.js
/home/ukistu09/Documents/Uki4/Programming/Nodejs/question33.js:2
fs.appendfile('ukinode.txt','It is a full scholarship based Accelerator program for Coding in Jaffna, Sri Lanka targeting students who has comp
leted Advanced Level examinations and don't have access to university education.')
SyntaxError: Unexpected token {
  at createScript (vm.js:56:10)
  at Object.runInThisContext (vm.js:97:10)
  at Module._compile (module.js:542:28)
  at Object.Module._extensions..js (module.js:579:10)
  at Module.load (module.js:487:32)
  at tryModuleLoad (module.js:446:12)
  at Function.Module.load (module.js:438:3)
  at Module._runMain (module.js:604:10)
  at run (bootstrap_node.js:390:7)
  at startup (bootstrap_node.js:150:9)
[nodemon] app crashed - waiting for file changes before starting...
[nodemon] restarting due to changes...
[nodemon] starting [node] question33.js
Updated!
[nodemon] clean exit - waiting for changes before restart
```



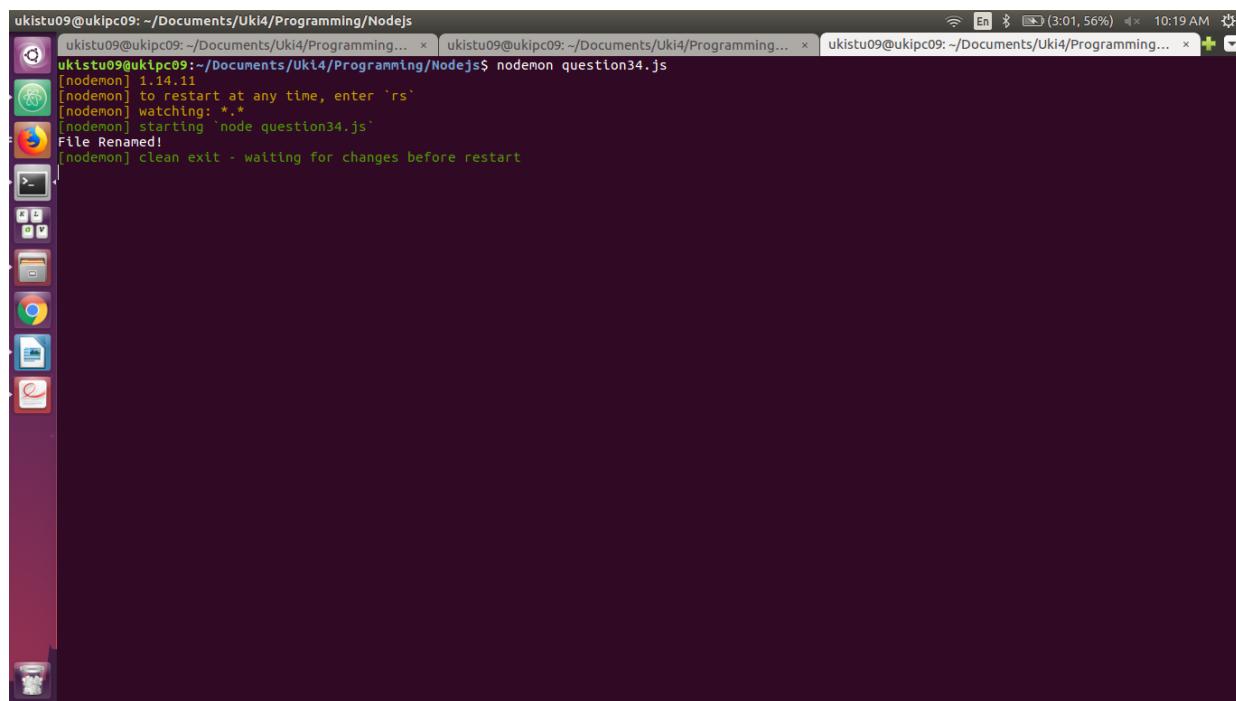
3.4 Rename the file as ukinodejsexercise1.txt



A screenshot of a code editor window titled "Nodejs". The sidebar shows a folder structure with files like ".lock.Nodejs", "firstnode.js", "Node.js Exercise", "question1.js", "question2.js", "question3.js", "question32.js", "question33.js", "question34.js", "sample.js", and "ukinodejsexercis". The main editor area has tabs for "question3.js", "question33.js", and "question34.js". The "question34.js" tab is active, displaying the following code:

```
1 var fs = require('fs');
2
3 fs.rename('ukinode.txt', 'ukinodejsexercise1.txt', function (err) {
4     if (err) throw err;
5     console.log('File Renamed!');
6 });
7
```

The status bar at the bottom shows "LF" and "master" along with a "1 update" icon.



A screenshot of a terminal window with three tabs. The current tab shows the command "nodemon question34.js" being run. The output is:

```
ukistu09@ukipc09: ~/Documents/Uki4/Programming/Nodejs
ukistu09@ukipc09: ~/Documents/Uki4/Programming... <-- Tab 1
ukistu09@ukipc09: ~/Documents/Uki4/Programming/Nodejs$ nodemon question34.js
[nodemon] 1.14.11
[nodemon] to restart at any time, enter 'rs'
[nodemon] watching: '*.*'
[nodemon] starting 'node question34.js'
File Renamed!
[nodemon] clean exit - waiting for changes before restart
```

The terminal window has a dark theme and a vertical toolbar on the left side.

3.5 Delete the file you created

```
var fs = require('fs');
fs.unlink('ukinodejsexercise1.txt', function (err) {
  if (err) throw err;
  console.log('File deleted!');
});
```

```
ukistu09@ukipc09: ~/Documents/Uki4/Programming/Nodejs$ nodemon question35.js
[nodemon] 1.14.11
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: `.*`
[nodemon] starting 'node question35.js'
File deleted!
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting 'node question35.js'
/home/ukistu09/Documents/Uki4/Programming/Nodejs/question35.js:4
  if (err) throw err;
  ^
Error: ENOENT: no such file or directory, unlink 'ukinodejsexercise1.txt'
  at Error (native)
[nodemon] app crashed - waiting for file changes before starting...
[nodemon] restarting due to changes...
[nodemon] starting 'node question35.js'
/home/ukistu09/Documents/Uki4/Programming/Nodejs/question35.js:4
  if (err) throw err;
  ^
Error: ENOENT: no such file or directory, unlink 'ukinodejsexercise1.txt'
  at Error (native)
[nodemon] app crashed - waiting for file changes before starting...
[nodemon] restarting due to changes...
[nodemon] starting 'node question35.js'
File deleted!
[nodemon] clean exit - waiting for changes before restart
```

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.

The screenshot shows a Linux desktop environment with a dark theme. In the top panel, there are icons for a terminal, file manager, and system settings. The system tray shows battery level at 62%, signal strength, and the date and time (10:42 AM). Below the top panel, there are two windows:

- Atom Editor:** The title bar says "question4.js — ~/Documents/Uki4/Programming/Nodejs — Atom". The left sidebar shows a "Nodejs" folder icon. The main editor area contains the following Node.js code:

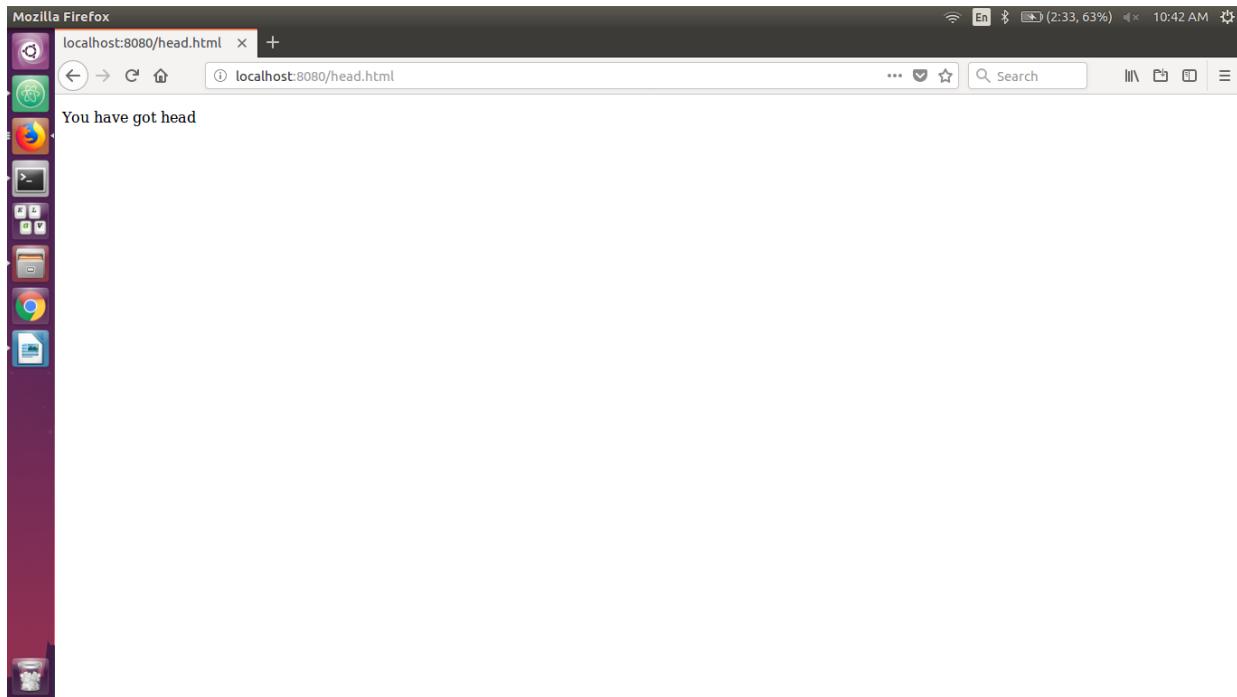
```
question4.js
1 var http = require('http');
2 var url = require('url');
3 var fs = require('fs');
4
5 http.createServer(function (req, res) {
6   var q = url.parse(req.url, true);
7   var filename = "." + q.pathname;
8   fs.readFile(filename, function(err, data) {
9     if (err) {
10       res.writeHead(404, {'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
```
- Mozilla Firefox:** The title bar says "Mozilla Firefox". A single tab is open with the URL "localhost:8080". The page content is "404 Not Found".

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

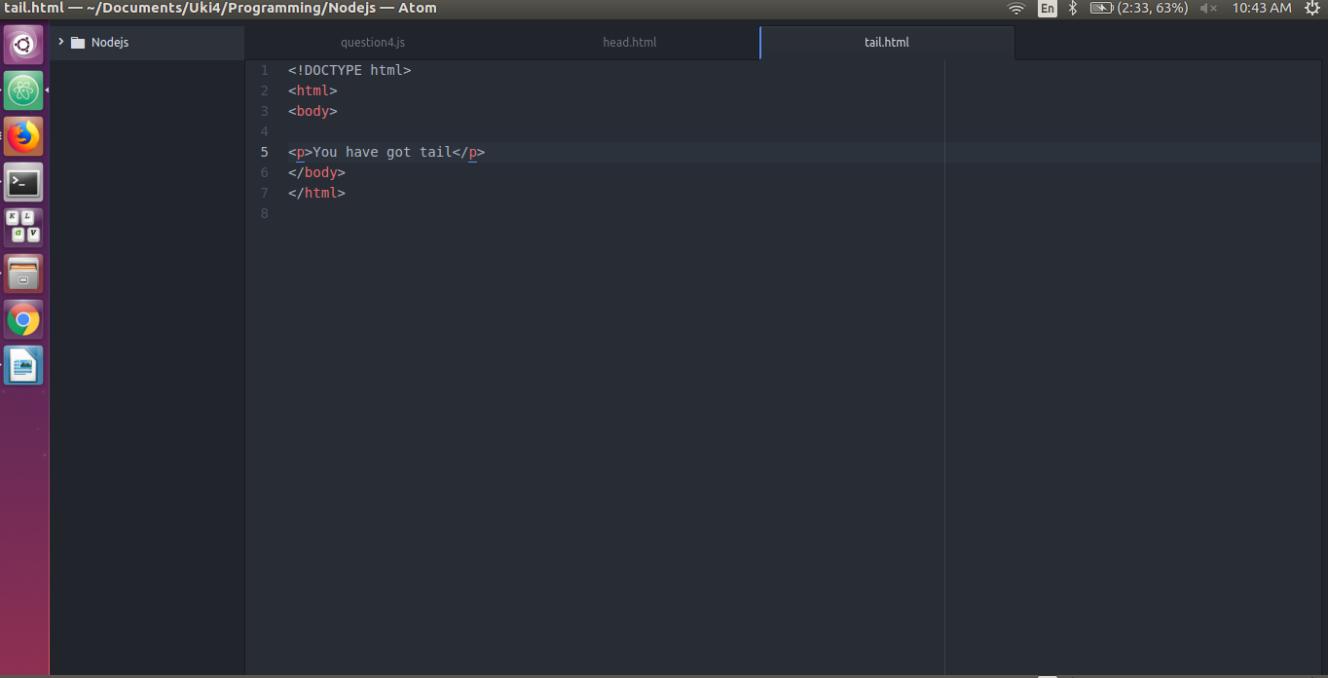
<http://localhost:8080/head.html> -> You have got head

The screenshot shows the Atom code editor interface. On the left is a vertical sidebar with icons for Node.js, GitHub, and other development tools. The main area has three tabs: 'question4.js', 'head.html', and 'tail.html'. The 'head.html' tab is active and contains the following code:

```
<!DOCTYPE html>
<html>
<body>
<p>You have got head</p>
</body>
</html>
```

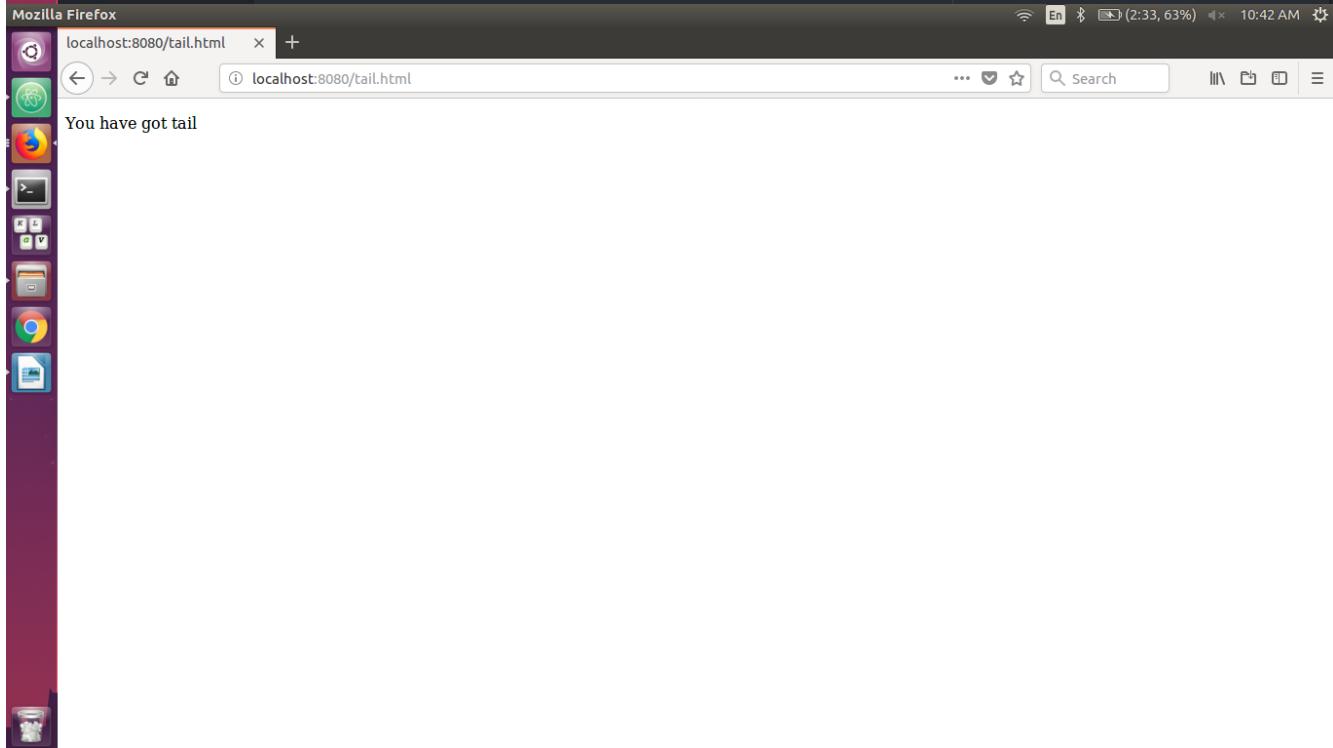


<http://localhost:8080/tail.html> -> You have got tail



The screenshot shows the Atom code editor interface. On the left is a sidebar with icons for various applications like terminal, file manager, and browser. The main area has tabs for 'tail.html', 'question4.js', and 'head.html'. The 'tail.html' tab is active, displaying the following HTML code:

```
<!DOCTYPE html>
<html>
<body>
<p>You have got tail</p>
</body>
</html>
```



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.

The screenshot shows a Linux desktop environment with several windows open:

- Code Editor:** An Atom window titled "question5.js" containing the following Node.js code:

```
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);
```
- Terminal:** A terminal window titled "Terminal" showing the command "npm install upper-case" being run. The output shows several "WARN" messages about missing files like package.json, README, and license.

```
ukistu09@ukipc09:~/Documents/Uki4/Programming/Nodejs$ npm install upper-case
/home/ukistu09/Documents/Uki4/Programming/Nodejs
└── upper-case@1.1.3
npm WARN enoent ENOENT: no such file or directory, open '/home/ukistu09/Documents/Uki4/Programming/Nodejs/package.json'
npm WARN Nodejs No description
npm WARN Nodejs No repository field.
npm WARN Nodejs No README data
npm WARN Nodejs No license field.
ukistu09@ukipc09:~/Documents/Uki4/Programming/Nodejs$ |
```
- Browser:** A Google Chrome window titled "localhost:8080" showing the text "UKI IS THE BEST PLACE TO LEARN PROGRAMMING !" in uppercase.

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

A screenshot of a terminal window titled "Nodejs". It shows two tabs: "question5.js" and "question6.js". The "question6.js" tab contains the following code:

```
1 var events = require('events');
2 var eventEmitter = new events.EventEmitter();
3
4
5 var myEventHandler = function () {
6   console.log('I bark when I see strangers !');
7 }
8
9 //Assign the eventhandler to an event:
10 eventEmitter.on('bark', myEventHandler);
11
12 //Fire the 'scream' event:
13 eventEmitter.emit('bark');
```

The terminal window also displays the command "node question6.js" being run and the output "I bark when I see strangers !".

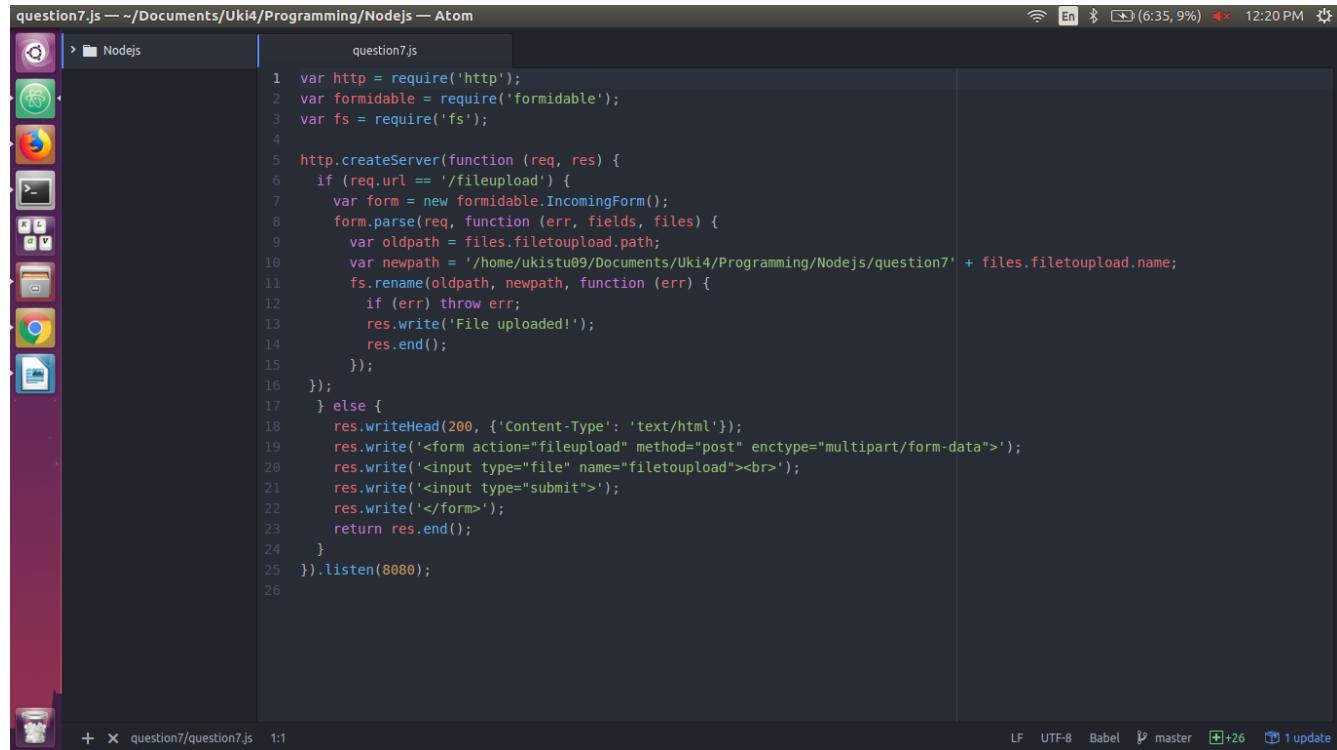
A screenshot of a terminal window titled "Nodejs". It shows two tabs: "question5.js" and "question6.js". The "question6.js" tab contains the following code:

```
1 var events = require('events');
2 var eventEmitter = new events.EventEmitter();
3
4
5 var myEventHandler = function () {
6   console.log('I bark when I see strangers !');
7 }
8
9 //Assign the eventhandler to an event:
10 eventEmitter.on('bark', myEventHandler);
11
12 //Fire the 'scream' event:
13 eventEmitter.emit('bark');
```

The terminal window also displays the command "node question6.js" being run and the output "I bark when I see strangers !".

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

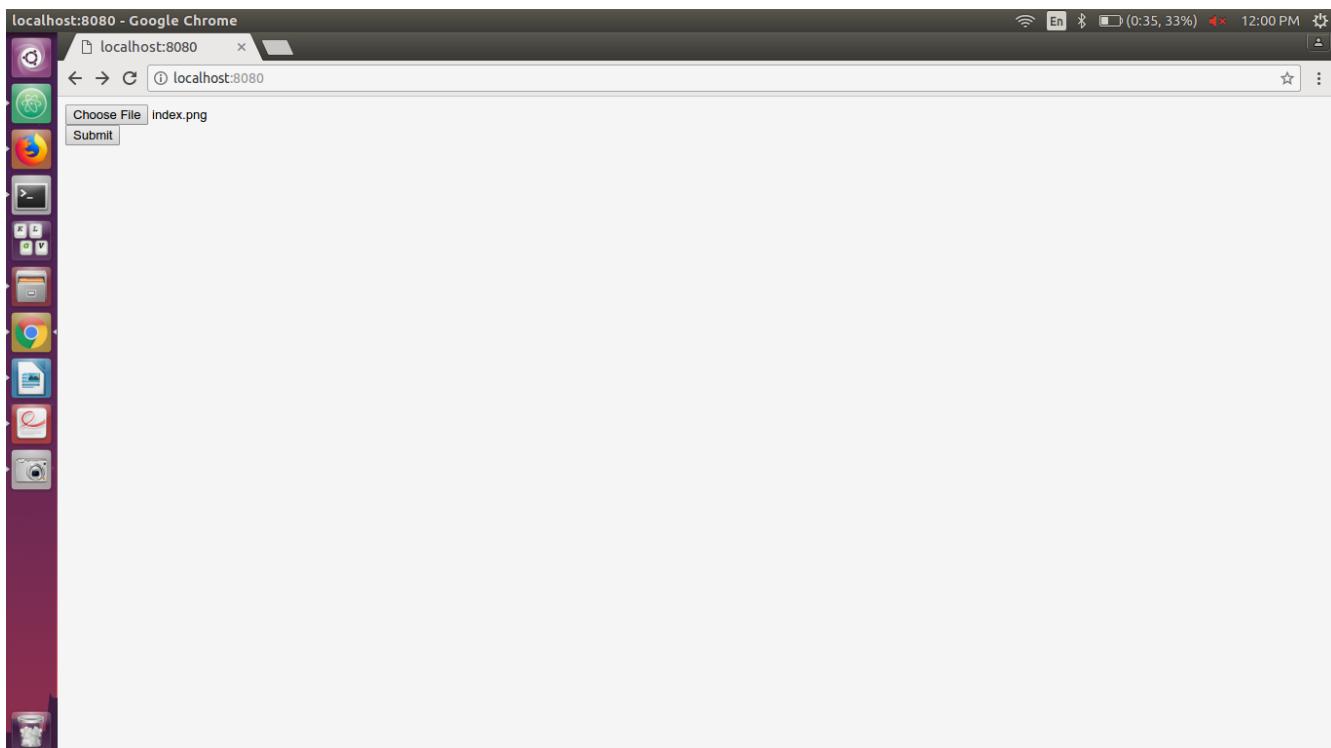
7.1 Save that uploaded file into your Documents directory.

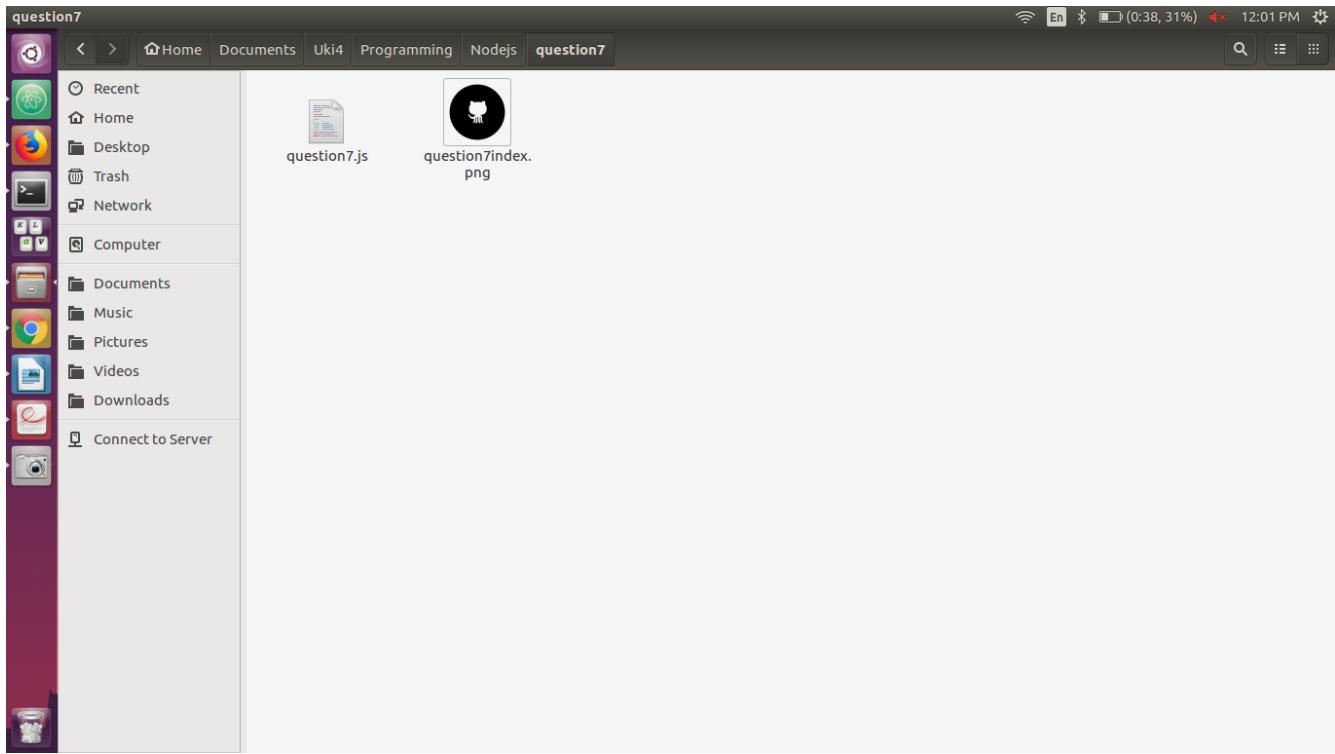


The screenshot shows the Atom code editor interface. The left sidebar has a 'Nodejs' folder icon. The main editor area contains the following Node.js code:

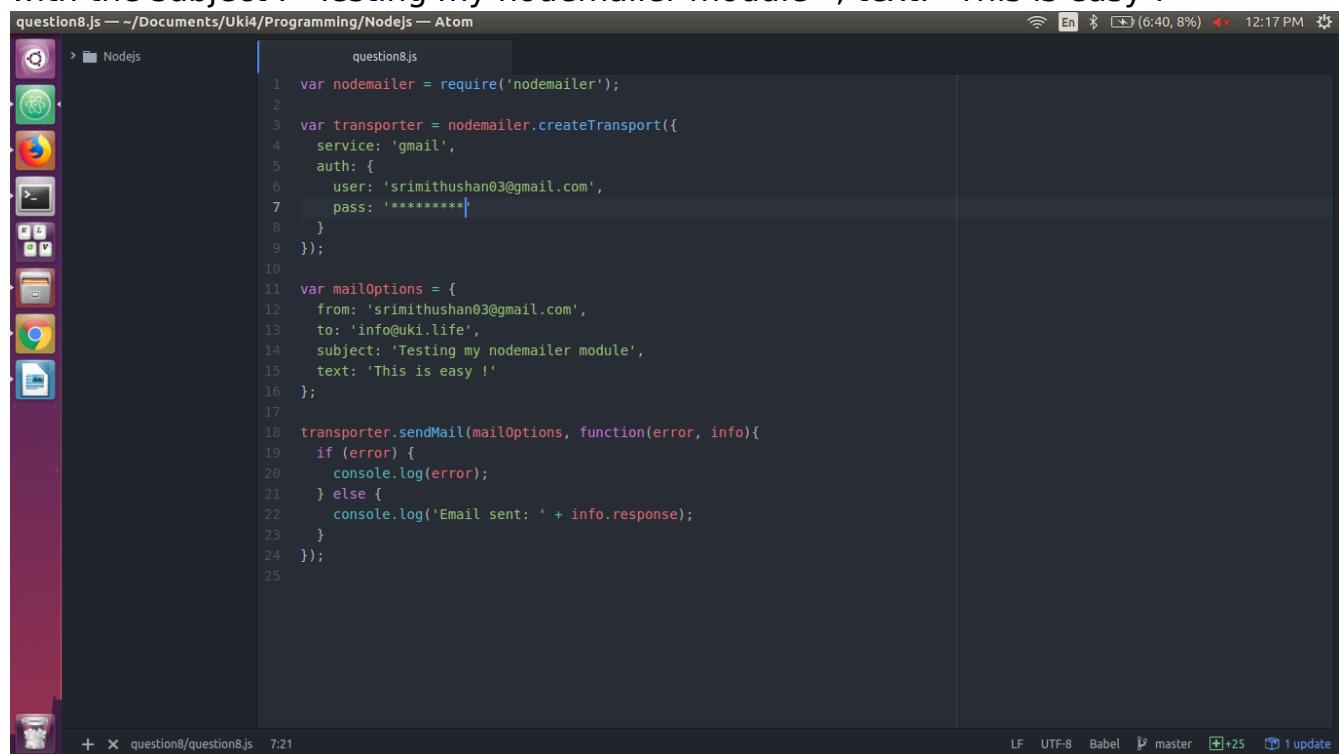
```
question7.js — ~/Documents/Uki4/Programming/Nodejs — Atom
question7js
1 var http = require('http');
2 var formidable = require('formidable');
3 var fs = require('fs');
4
5 http.createServer(function (req, res) {
6   if (req.url == '/fileupload') {
7     var form = new formidable.IncomingForm();
8     form.parse(req, function (err, fields, files) {
9       var oldpath = files.fileupload.path;
10      var newpath = '/home/ukistu09/Documents/Uki4/Programming/Nodejs/question7' + files.fileupload.name;
11      fs.rename(oldpath, newpath, function (err) {
12        if (err) throw err;
13        res.write('File uploaded!');
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
```

The status bar at the bottom shows: LF UTF-8 Babel master +26 1 update.

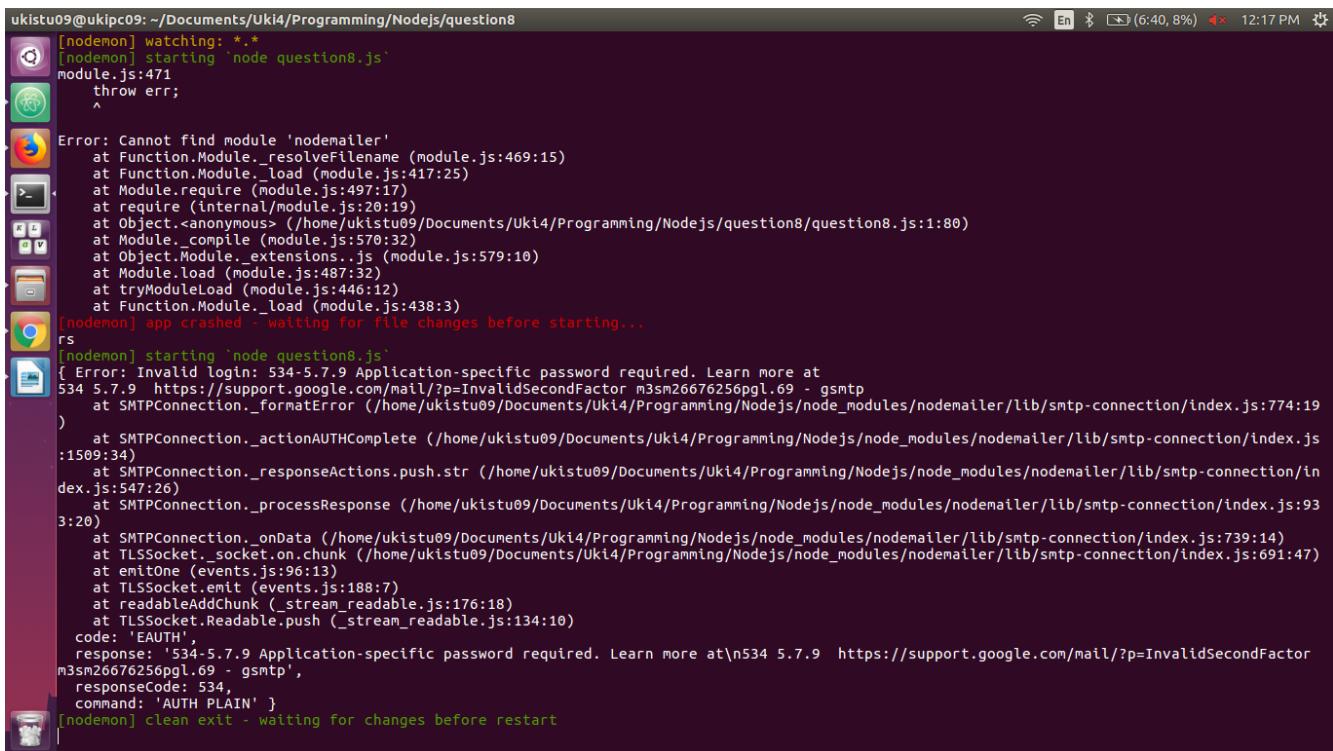




8. Using the Nodemailer module create a server and send a mail to info@uki.life with the subject : “Testing my nodemailer module” , text: “This is easy !”



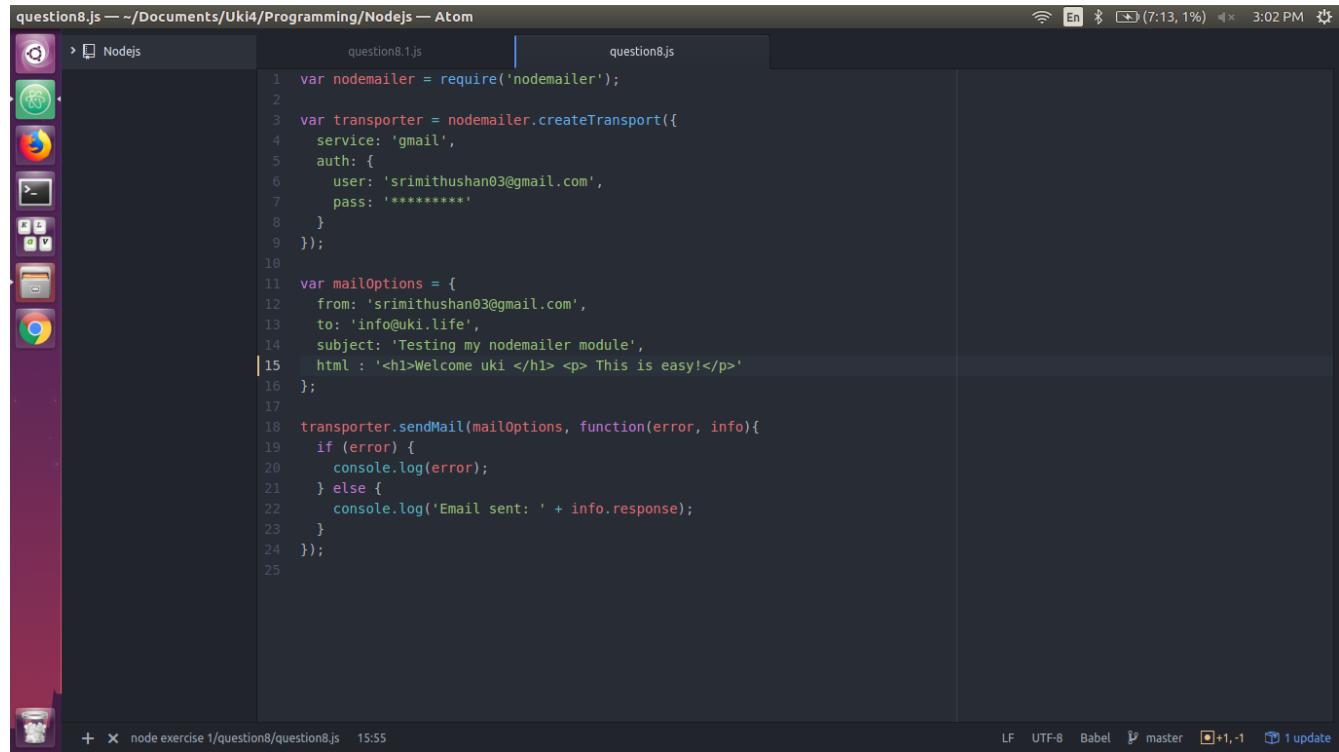
```
question8.js -- ~/Documents/Uki4/Programming/Nodejs — Atom
question8.js
1 var nodemailer = require('nodemailer');
2
3 var transporter = nodemailer.createTransport({
4   service: 'gmail',
5   auth: {
6     user: 'srimitushan03@gmail.com',
7     pass: '*****'
8   }
9 });
10
11 var mailOptions = {
12   from: 'srimitushan03@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
```



```
ukistu09@ukipc09:~/Documents/Uki4/Programming/Nodejs/question8
[nodemon] watching: *.*
[nodemon] starting `node question8.js`
module.js:471
    throw err;
^

Error: Cannot find module 'nodemailer'
  at Function.Module._resolveFilename (module.js:469:15)
  at Function.Module._load (module.js:417:25)
  at Module.require (module.js:497:17)
  at require (internal/module.js:20:19)
  at Object.<anonymous> (/home/ukistu09/Documents/Uki4/Programming/Nodejs/question8/question8.js:1:80)
  at Module._compile (module.js:570:32)
  at Object.Module._extensions..js (module.js:579:10)
  at Module.load (module.js:487:32)
  at tryModuleLoad (module.js:446:12)
  at Function.Module._load (module.js:438:3)
[nodemon] app crashed - waiting for file changes before starting...
rs
[nodemon] starting `node question8.js`
{ Error: Invalid login: 534-5.7.9 Application-specific password required. Learn more at
  534 5.7.9 https://support.google.com/mail/?p=InvalidSecondFactor m3sm26676256pgl.69 - gsmtp
  at SMTPConnection._formatError (/home/ukistu09/Documents/Uki4/Programming/Nodejs/node_modules/nodemailer/lib/smtp-connection/index.js:774:19)
  )   at SMTPConnection._actionAUTHComplete (/home/ukistu09/Documents/Uki4/Programming/Nodejs/node_modules/nodemailer/lib/smtp-connection/index.js:1509:34)
  )   at SMTPConnection._responseActions.push.str (/home/ukistu09/Documents/Uki4/Programming/Nodejs/node_modules/nodemailer/lib/smtp-connection/index.js:547:26)
  )   at SMTPConnection._processResponse (/home/ukistu09/Documents/Uki4/Programming/Nodejs/node_modules/nodemailer/lib/smtp-connection/index.js:3:20)
  )   at SMTPConnection._onData (/home/ukistu09/Documents/Uki4/Programming/Nodejs/node_modules/nodemailer/lib/smtp-connection/index.js:739:14)
  )   at TLSSocket._socket.on.chunk (/home/ukistu09/Documents/Uki4/Programming/Nodejs/node_modules/nodemailer/lib/smtp-connection/index.js:691:47)
  )   at emitOne (events.js:96:13)
  )   at TLSSocket.emit (events.js:188:7)
  )   at readableAddChunk (_stream_readable.js:176:18)
  )   at TLSSocket.Readable.push (_stream_readable.js:134:10)
  code: 'EAUTH',
  response: '534-5.7.9 Application-specific password required. Learn more at\n534 5.7.9 https://support.google.com/mail/?p=InvalidSecondFactor
m3sm26676256pgl.69 - gsmtp',
  responseCode: 534,
  command: 'AUTH PLAIN' }
[nodemon] clean exit - waiting for changes before restart
```

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



The screenshot shows the Atom code editor interface. The title bar indicates the file is 'question8.js' located at '~/Documents/Uki4/Programming/Nodejs'. The status bar shows the file path as 'node exercise 1/question8/question8.js' and the current time as '15:55'. The code editor has two tabs: 'question8.js' (active) and 'question8.1.js'. The code in 'question8.js' uses the nodemailer module to send an HTML email. It includes code to require 'nodemailer', create a transporter for Gmail, define mail options with a subject and an HTML body containing an h1 tag and a p tag, and finally send the email using the transporter's sendMail method. The code editor's sidebar shows icons for various applications like Node.js, GitHub, and Google Chrome.

```
question8.js — ~/Documents/Uki4/Programming/Nodejs — Atom
question8.1.js question8.js

1 var nodemailer = require('nodemailer');
2
3 var transporter = nodemailer.createTransport({
4   service: 'gmail',
5   auth: {
6     user: 'srimithushan03@gmail.com',
7     pass: '*****'
8   }
9 });
10
11 var mailOptions = {
12   from: 'srimithushan03@gmail.com',
13   to: 'info@uki.life',
14   subject: 'Testing my nodemailer module',
15   html : '<h1>Welcome uki </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
```

+ × node exercise 1/question8/question8.js 15:55 LF UTF-8 Babel master +1,-1 1 update