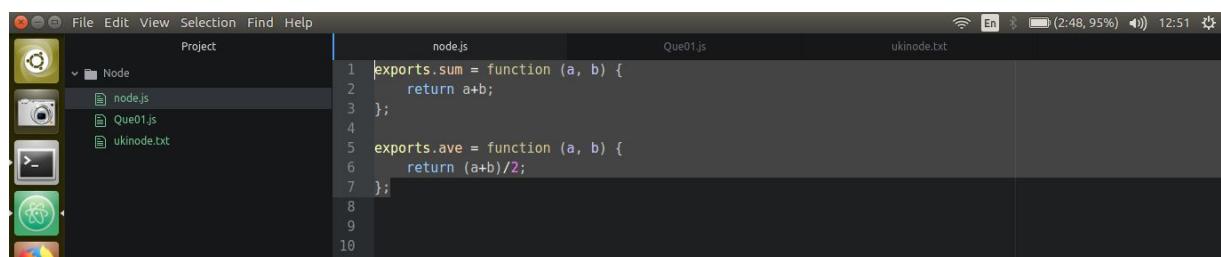


1. Create a custom module which returns the sum and average of any two numbers passed into it. Require the module and run the server by passing 123 and 321 so that the server prints out the sum and average.

```
exports.sum = function (a, b) {
    return a+b;
};
```

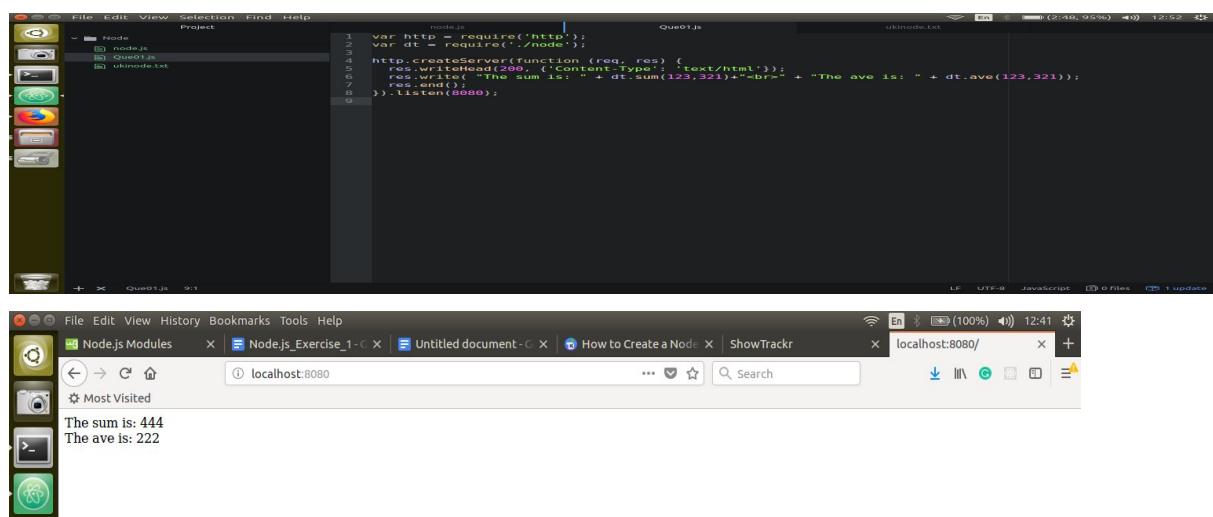
```
exports.ave = function (a, b) {
    return (a+b)/2;
};
```



A screenshot of a Mac OS X desktop environment. At the top, there's a menu bar with File, Edit, View, Selection, Find, Help. Below the menu bar is a dock with various icons. The main area shows a terminal window titled 'node.js' containing the code for the custom module. To the left of the terminal is a 'Project' sidebar showing files: node.js, Que01.js, and ukinode.txt. To the right of the terminal is another terminal window titled 'Que01.js' showing the command-line interface. Below these windows is a browser window with the address bar set to 'localhost:8080/'. The browser displays the output of the application, which includes the sum and average of the numbers 123 and 321.

```
var http = require('http');
var dt = require('./node');
```

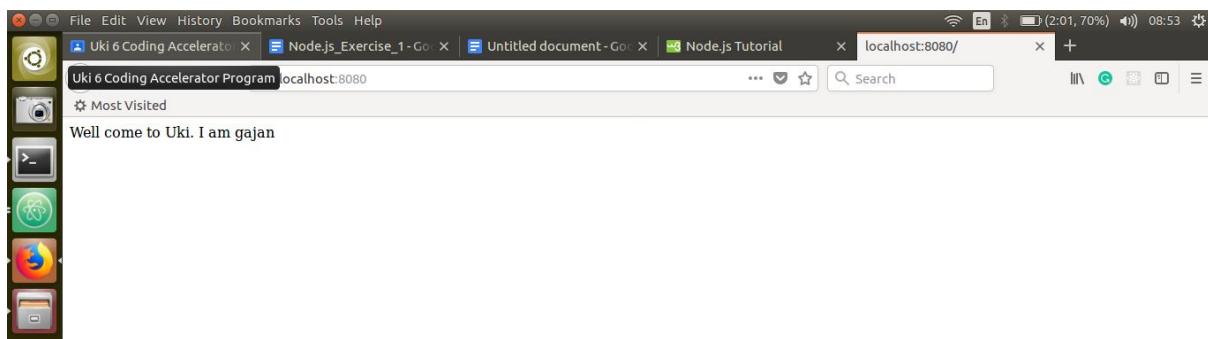
```
http.createServer(function (req, res) {
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.write("The sum is: " + dt.sum(123,321)+"<br>" + "The ave is: " + dt.ave(123,321));
    res.end();
}).listen(8080);
```



A screenshot of a Mac OS X desktop environment. At the top, there's a menu bar with File, Edit, View, Selection, Find, Help. Below the menu bar is a dock with various icons. The main area shows a terminal window titled 'node.js' containing the code for the server. To the left of the terminal is a 'Project' sidebar showing files: node.js, Que01.js, and ukinode.txt. To the right of the terminal is another terminal window titled 'Que01.js' showing the command-line interface. Below these windows is a browser window with the address bar set to 'localhost:8080/'. The browser displays the output of the application, which includes the sum and average of the numbers 123 and 321.

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)

```
var http = require('http');
http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.end('Well come to Uki. I am gajan');
}).listen(8080);
```

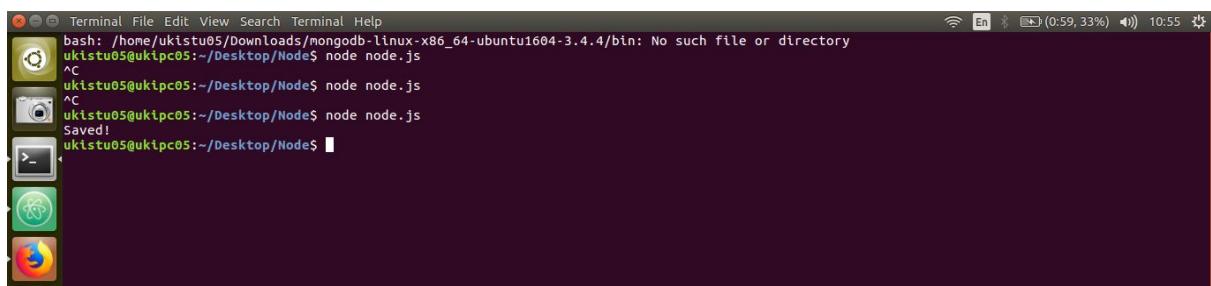
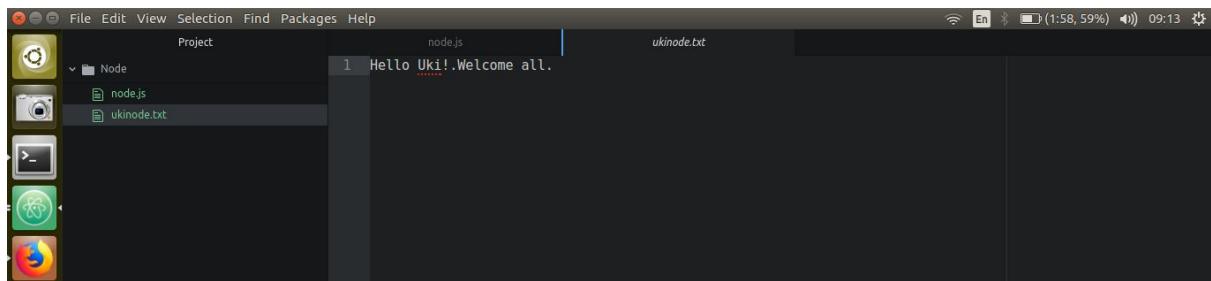


```
var http = require('http');
http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.end('Well come to Uki. I am gajan');
}).listen(8000);
```

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

3.1 Write a paragraph about Uki into that file

```
var fs = require('fs');
fs.appendFile('ukinode.txt', 'Hello Uki!.Welcome all.', function (err) {
if (err) throw err
console.log('Saved!');
});
```

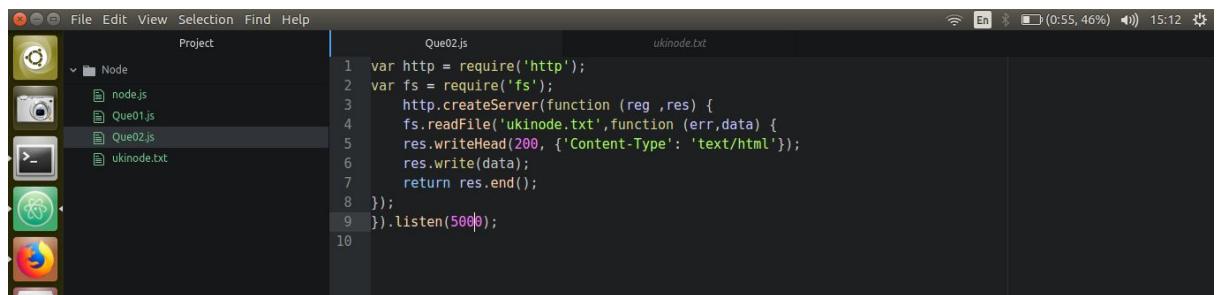


- 3.2 Serve that file to the client (Read File) over your server

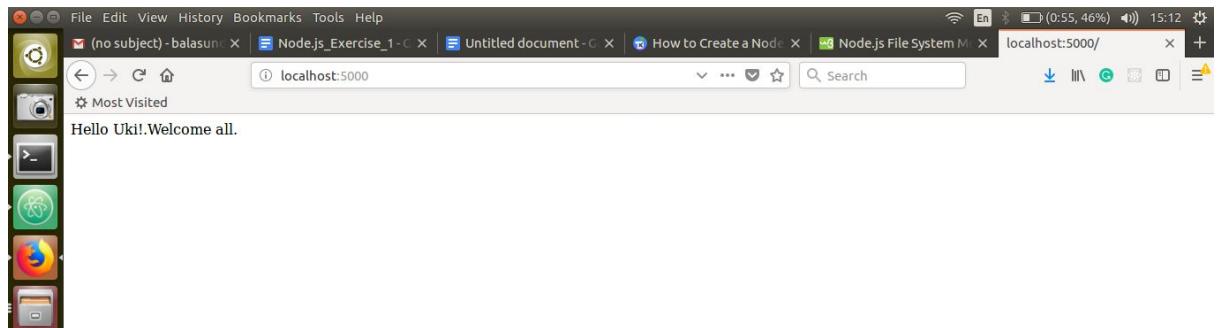
```
var http = require('http');
var fs = require('fs');

http.createServer(function (req ,res) {
  fs.readFile('ukinode.txt',function (err,data) {
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.write(data);
    return res.end();
});
```

```
}).listen(5000);
```



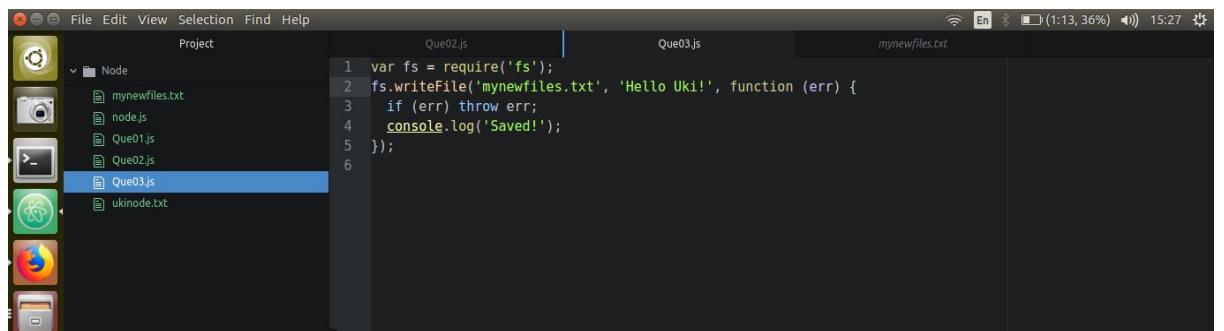
```
File Edit View Selection Find Help
Project Que02.js ukinode.txt
1 var http = require('http');
2 var fs = require('fs');
3 http.createServer(function (req ,res) {
4   fs.readFile('ukinode.txt',function (err,data) {
5     res.writeHead(200, {'Content-Type': 'text/html'});
6     res.write(data);
7     return res.end();
8   });
9 }).listen(5000);
10
```



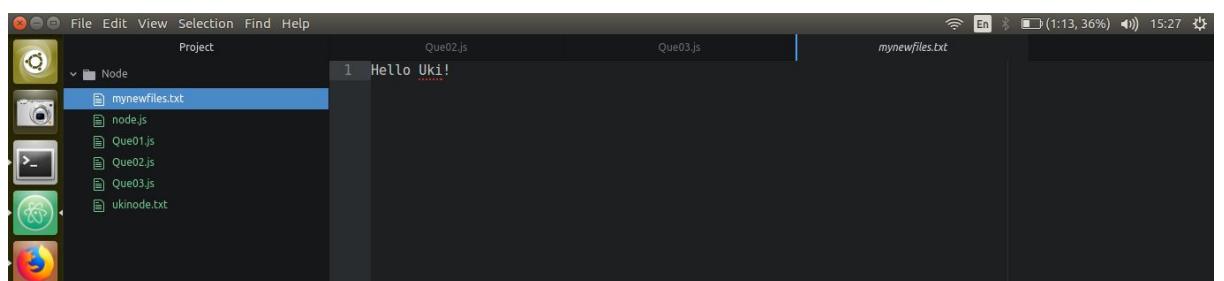
Hello Uki!.Welcome all.

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

```
var fs = require('fs');
fs.writeFile('mynewfiles.txt', 'Hello Uki!', function (err) {
  if (err) throw err;
  console.log('Saved!');
});
```



```
File Edit View Selection Find Help
Project Que02.js Que03.js mynewfiles.txt
1 var fs = require('fs');
2 fs.writeFile('mynewfiles.txt', 'Hello Uki!', function (err) {
3   if (err) throw err;
4   console.log('Saved!');
5 });
6
```



```
File Edit View Selection Find Help
Project Que02.js Que03.js mynewfiles.txt
1 Hello Uki!
```

3.4 Rename the file as ukinodejsexercise1.txt

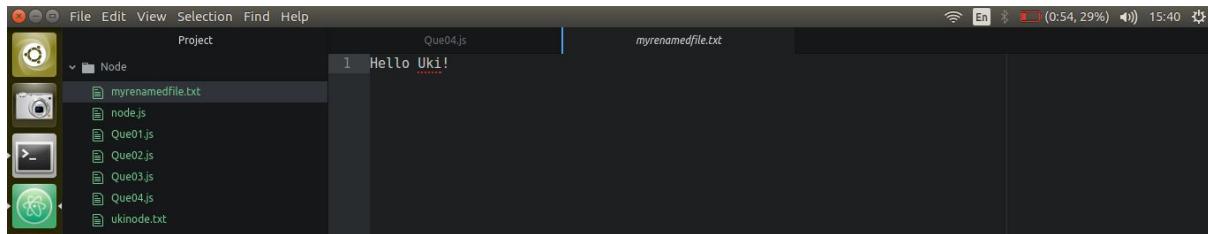
```
var fs = require('fs');
fs.rename('mynewfiles.txt', 'myrenamedfile.txt', function (err) {
  if (err) throw err;
  console.log('File Renamed!');
```

});



```
File Edit View Selection Find Help
Project
Que02.js Que03.js Que04.js
1 var fs = require('fs');
2 fs.rename('myrenamedfile.txt', 'myrenamedfile.txt', function (err) {
3   if (err) throw err;
4   console.log('File Renamed!');
5 });
6
```

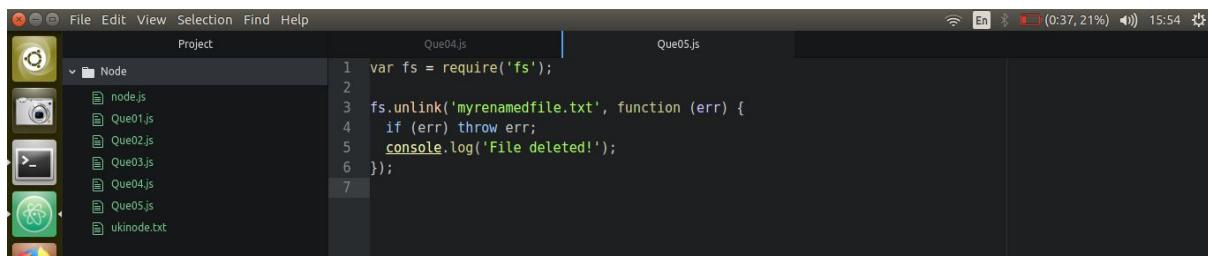
myrenamedfile.txt



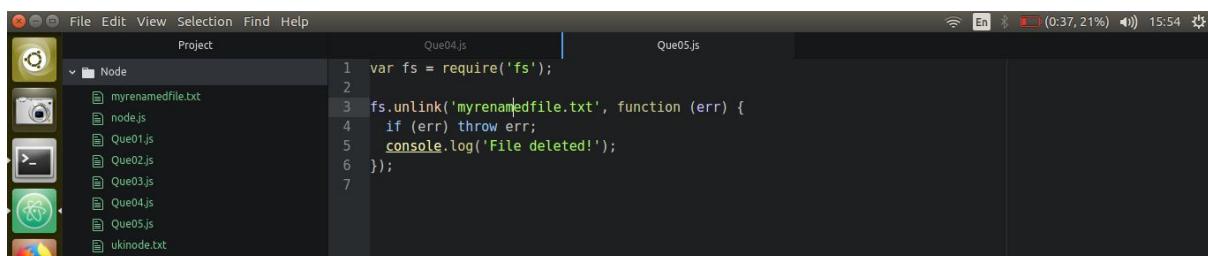
```
File Edit View Selection Find Help
Project
Que04.js myrenamedfile.txt
1 Hello Uki!
```

3.5 Delete the file you created

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



```
File Edit View Selection Find Help
Project
Que04.js Que05.js
1 var fs = require('fs');
2
3 fs.unlink('myrenamedfile.txt', function (err) {
4   if (err) throw err;
5   console.log('File deleted!');
6 });
7
```



```
File Edit View Selection Find Help
Project
Que04.js Que05.js
1 var fs = require('fs');
2
3 fs.unlink('myrenamedfile.txt', function (err) {
4   if (err) throw err;
5   console.log('File deleted!');
6 });
7
```

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

```
<!DOCTYPE html>
<html>
<body>
<h1>Head</h1>
<p>you have got head</p>
</body>
</html>
```

```
<!DOCTYPE html>
<html>
<body>
<h1>Tail</h1>
<p>you have got tail</p>
</body>
</html>
```

```
var http = require('http');
var url = require('url');
var fs = require('fs');

http.createServer(function (req, res) {
  var q = url.parse(req.url, true);
  var filename = "." + q.pathname;
  fs.readFile(filename, function(err, data) {
    if (err) {
      res.writeHead(404, {'Content-Type': 'text/html'});
      return res.end("404 Not Found");
    }
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.write(data);
    return res.end();
  });
}).listen(8080);
```

A screenshot of a terminal window showing the file structure of a Node.js project. The project directory is 'Node' and contains files: head.html, node.js, Que01.js, Que02.js, Que03.js, Que04.js, Que05.js, Que06.js, tail.html, and ukinode.txt. The current file being edited is 'head.html'. The code in 'head.html' is:

```
<!DOCTYPE html>
<body>
<h1>Head</h1>
<p>you have got head</p>
</body>
</html>
```

A screenshot of a terminal window showing the file structure of a Node.js project. The project directory is 'Node' and contains files: head.html, node.js, Que01.js, Que02.js, Que03.js, Que04.js, Que05.js, Que06.js, tail.html, and ukinode.txt. The current file being edited is 'tail.html'. The code in 'tail.html' is:

```
<!DOCTYPE html>
<html>
<body>
<h1>Tail</h1>
<p>you have got tail</p>
</body>
</html>
```

A screenshot of a terminal window showing the file structure of a Node.js project. The project directory is 'Node' and contains files: head.html, node.js, Que01.js, Que02.js, Que03.js, Que04.js, Que05.js, Que06.js, tail.html, and ukinode.txt. The current file being edited is 'Que06.js'. The code in 'Que06.js' is:

```
var http = require('http');
var url = require('url');
var fs = require('fs');

http.createServer(function (req, res) {
  var q = url.parse(req.url, true);
  var filename = "." + q.pathname;
  fs.readFile(filename, function(err, data) {
    if (err) {
      res.writeHead(404, {'Content-Type': 'text/html'});
      return res.end("404 Not Found");
    }
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.write(data);
    return res.end();
  });
}).listen(8080);
```

A screenshot of a web browser showing the result of running 'head.html' on port 8080. The URL is 'localhost:8080/head.html'. The page displays the content 'Head' and 'you have got head'.

A screenshot of a web browser showing the result of running 'tail.html' on port 8080. The URL is 'localhost:8080/tail.html'. The page displays the content 'Tail' and 'you have got tail'.

A screenshot of a web browser showing the result of running the Node.js URL Module on port 8080. The URL is 'localhost:8080/'. The page displays the content '404 Not Found'.