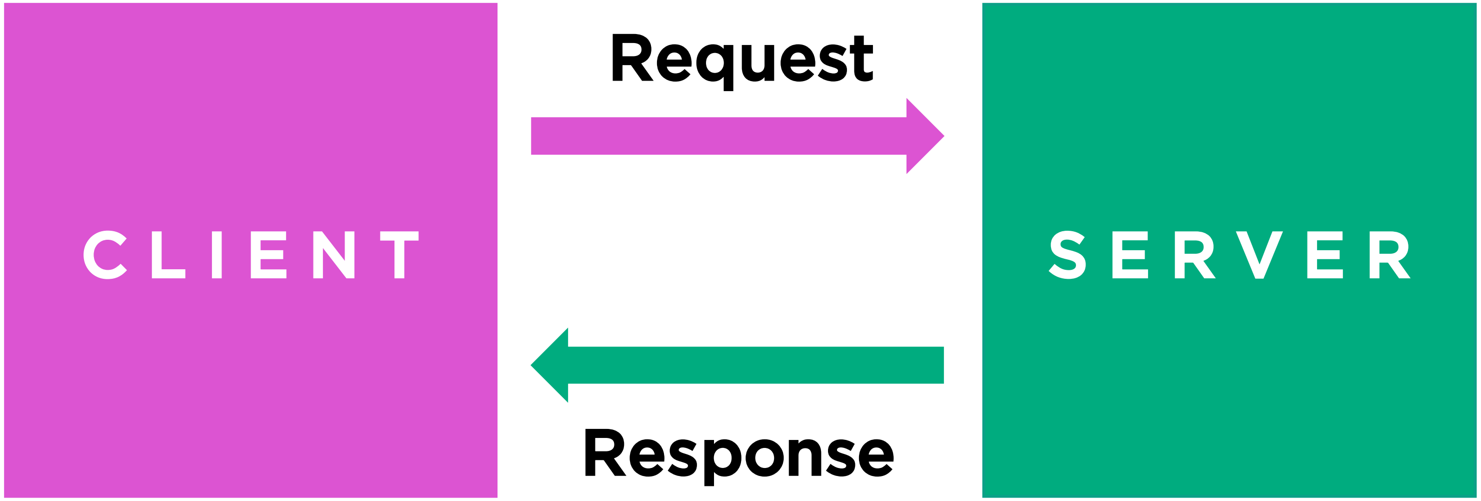
Back-End Development

Exploring

#3 HTTP Module

Purwadhika
Startup and Coding School

Clients & Servers



When we surf in a website sometimes we do something, for instance click a button, it will request data from the server & we'll see its response, like opening a new webpage.

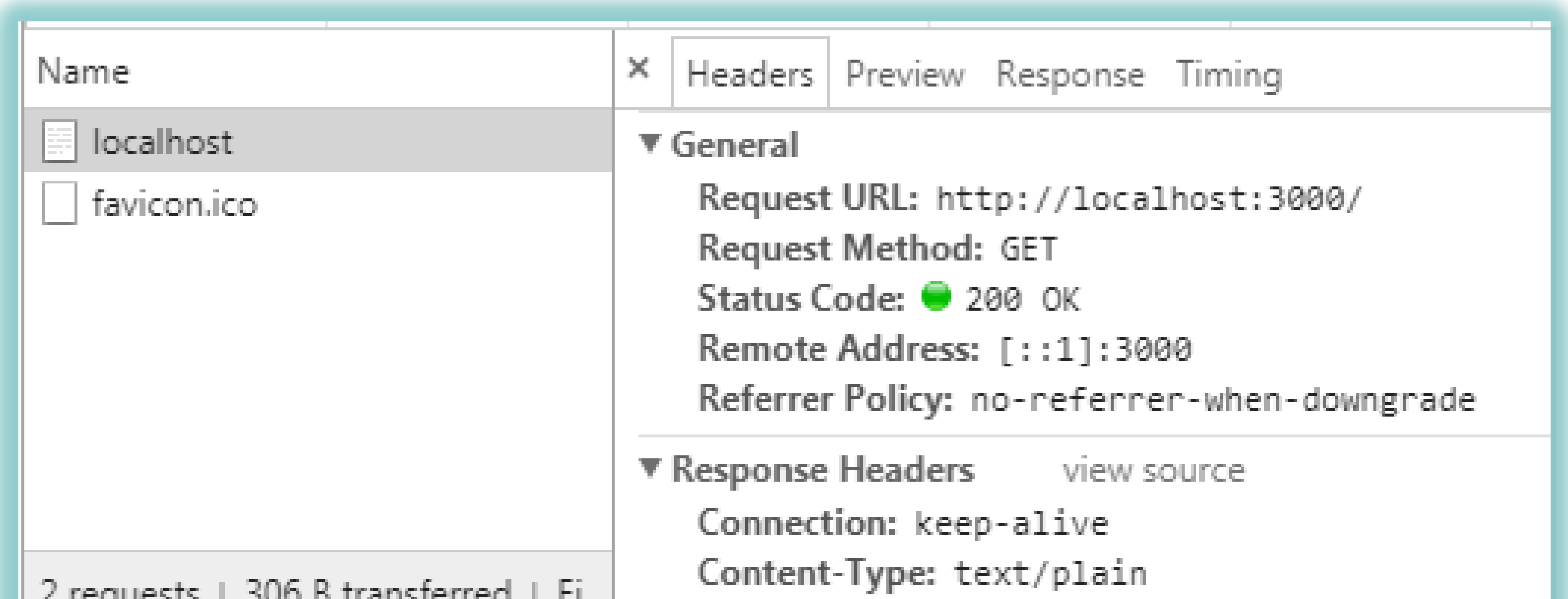
Client & server communicate each other, using *protocols*: a set of communication rules, that both client & server agree to use when communicating.

HTTP Methods

- *HTTP (The Hypertext Transfer Protocol)* is designed to enable communications between clients & servers. It works as a request & response protocol between a client & server.
- A web browser may be the client, and an application on a computer that hosts a web site may be the server.
- The most commonly used HTTP Methods are POST, GET, PUT, PATCH & DELETE.

Create a Server

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

var server = http.createServer(function(req,res){
      res.writeHead(200, {'Content-Type' :
'text/plain'});
      res.end('Halo semuanya!');
});

server.listen(3000);
// buka di browser = localhost : 3000
console.log('Server aktif. Listening port 3000.');
```



▼ General

Request URL: http://localhost:3000/
Request Method: GET
Status Code: ● 200 OK
Remote Address: [::1]:3000
Referrer Policy: no-referrer-when-downgrade

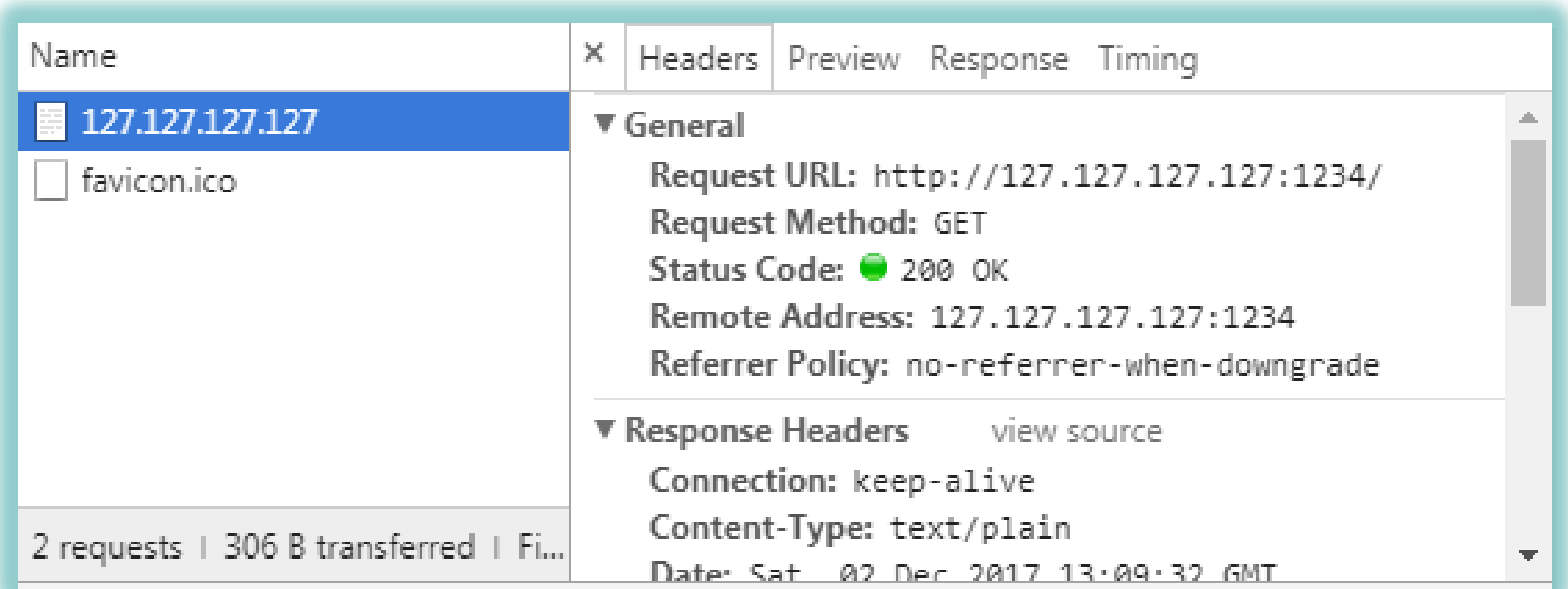
▼ Response Headers

Connection: keep-alive
Content-Type: text/plain

2 requests | 306 B transferred | Fi

Create a Server
# set url

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

var server = http.createServer(function(req,res){
     res.writeHead(200, {'Content-Type' : 'text/plain'});
     res.end('Halo semuanya!');
});

server.listen(1234, '127.127.127.127');
// buka di browser = 127.127.127.127 : 1234
console.log('Server aktif. Listening port 1234.');
```



▼ General

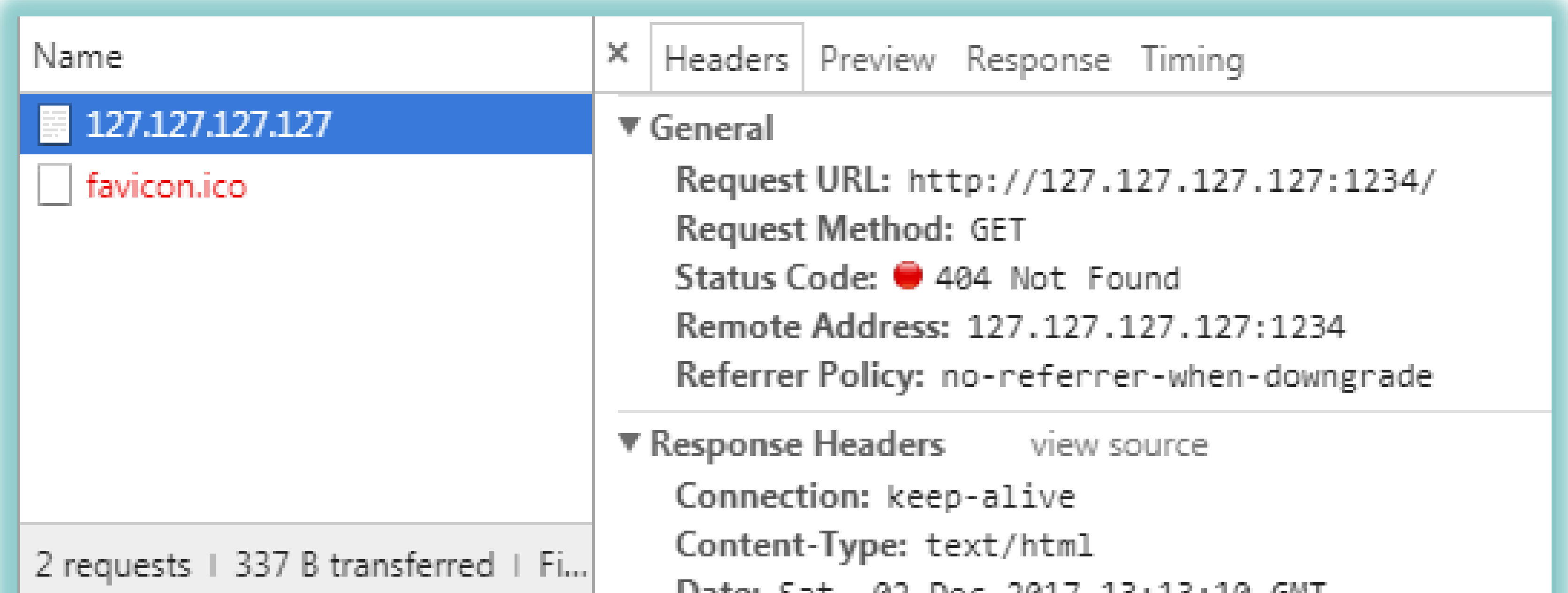
Request URL: http://127.127.127.127:1234/
Request Method: GET
Status Code: ● 200 OK
Remote Address: 127.127.127.127:1234
Referrer Policy: no-referrer-when-downgrade

▼ Response Headers view source

Connection: keep-alive
Content-Type: text/plain
Date: Sat, 02 Dec 2017 13:09:32 GMT

Create a Server
# set response

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

var server = http.createServer(function(req,res){
     res.writeHead(404, {'Content-Type' : 'text/html'});
     res.end('<h1>Halo semuanya!</h1>');
});

server.listen(1234, '127.127.127.127');
// buka di browser = 127.127.127.127 : 1234
console.log('Server aktif. Listening port 1234.');
```



127.127.127.127
favicon.ico

2 requests | 337 B transferred | Fi...

▼ General

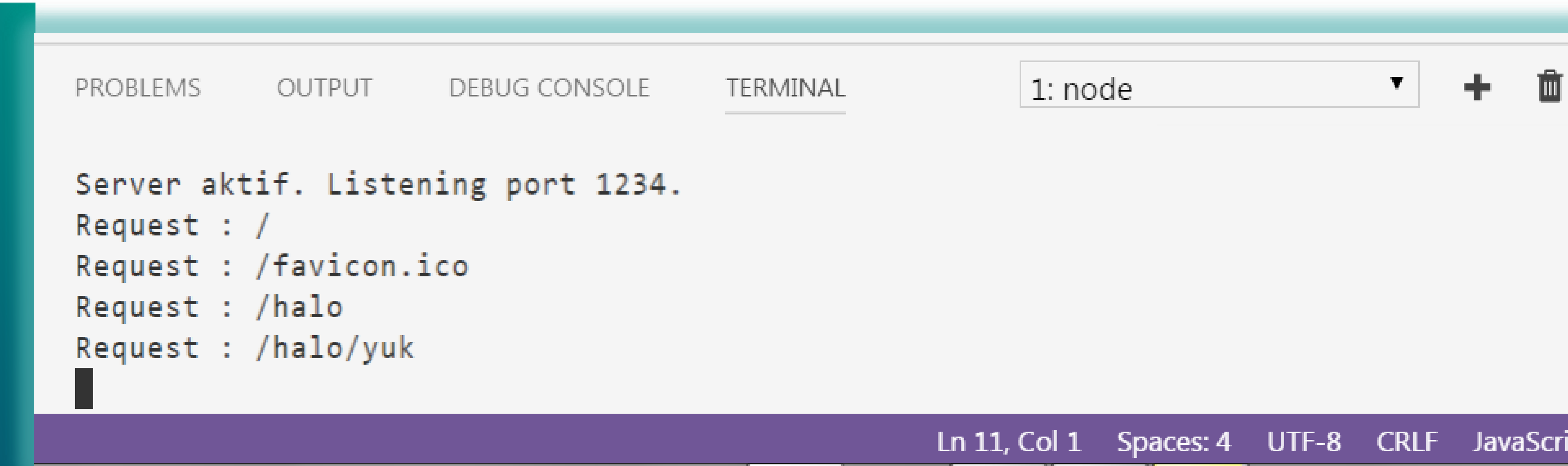
Request URL: http://127.127.127.127:1234/
Request Method: GET
Status Code: ● 404 Not Found
Remote Address: 127.127.127.127:1234
Referrer Policy: no-referrer-when-downgrade

▼ Response Headers

Connection: keep-alive
Content-Type: text/html
Date: Sat, 02 Dec 2017 13:13:10 GMT

Create a Server
# print request url

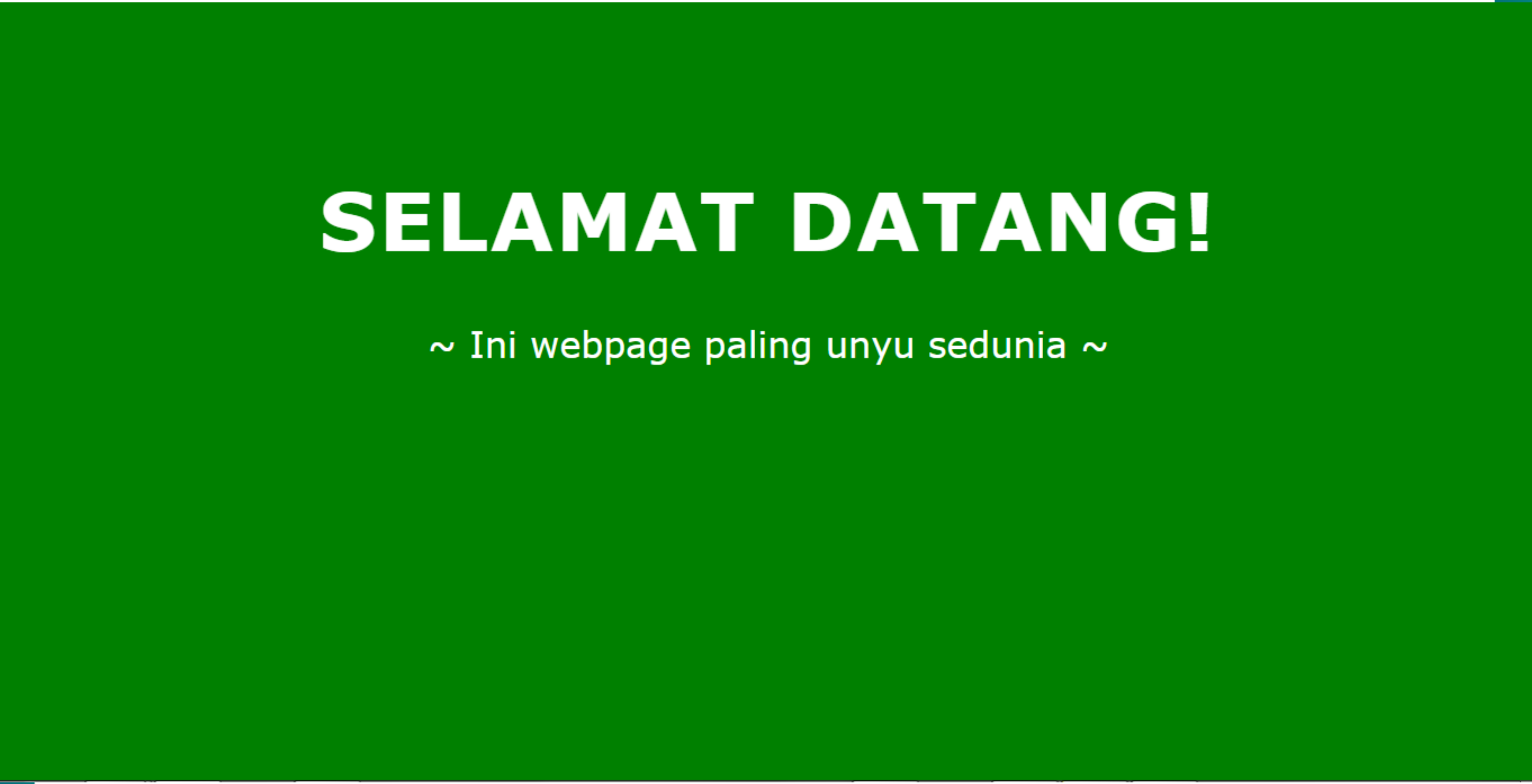
```
var http = require('http');

var server = http.createServer(function(req,res){
     console.log('Request : ' + req.url);
     res.writeHead(404, {'Content-Type' : 'text/html'});
     res.end('<h1>Halo semuanya!</h1>');
});

server.listen(1234, '127.127.127.127');
console.log('Server aktif. Listening port 1234.');
// buka di browser, tambahkan url target /sesuatu
```



```
Server aktif. Listening port 1234.
Request : /
Request : /favicon.ico
Request : /halo
Request : /halo/yuk
```

Serving HTML File

satu.html



SELAMAT DATANG!

~ Ini webpage paling unyu sedunia ~

Serving HTML File

satu.html

```
<!DOCTYPE html>
<html>
   <head><title>Lintang</title>
      <style>
      body{background:green; font-family:verdana;
color:white; padding:150px;}
      h1{font-size:72px; text-transform:uppercase;
letter-spacing:2px; text-align:center;}
      p{font-size:32px; text-align:center;}
      </style>
   </head>
   <body>
      <h1>Selamat Datang!</h1>
      <p>~ Ini webpage paling unyu sedunia ~</p>
   </body>
</html>
```

Serving HTML File
#1 using FS readFileSync

satu.js

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

var x = fs.readFileSync('satu.html', 'utf8');

var server = http.createServer(function(req,res){
     console.log('Request : ' + req.url);
     res.writeHead(200, {'Content-Type':'text/html'});
     res.end(x);
});

server.listen(3000);
console.log('Server aktif. Listening port 3000.');
```

Purwadhika
Startup and Coding School

Serving HTML File
#2 using FS readStream

satu.js

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

var server = http.createServer(function(req,res){
   console.log('Request : ' + req.url);
   res.writeHead(200, {'Content-Type':'text/html'});

   var bacaStream
=fs.createReadStream(__dirname+'/satu.html','utf8');
   bacaStream.pipe(res);
});

server.listen(3000);
console.log('Server aktif. Listening port 3000.');
```

Purwadhika
Startup and Coding School

JSON

- **JSON (*Javascript Object Notation*)** is a lightweight data-interchange format that based on a subset of JS. It's easy to read, write and can be used with any modern language.

- **JSON is not a JavaScript Object!**

JSON rules:

- Uses property/value pairs = {"nama" : "Andi"}
- Uses doublequotes on its prop & val (except number)
- Must use specified data type
- File type is ".json"
- Mime type is "application/json"

Purwadhika
Startup and Coding School

Serving JSON Data

satu.js

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

var server = http.createServer(function(req,res){
   console.log('Request : ' + req.url);
   res.writeHead(200, {'Content-Type':'application/json'});
   var dataku = {
      nama: 'Lintang',
      usia: 24,
      jomblo: true
   };
   res.end(JSON.stringify(dataku));
});

server.listen(3000);
console.log('Server aktif. Listening port 3000.');
```

Serving JSON File #1

satu.json

```
{
    "nama":"Lintang",
    "usia": 24,
    "status": "jomblo"
}
```

Serving JSON File #2

satu.js

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

var x = fs.readFileSync('satu.json', 'utf8');

var server = http.createServer(function(req,res){
     console.log('Request : ' + req.url);
     res.writeHead(200, {'Content-Type':'text/plain'});
     res.end(x);
});

server.listen(3000);
console.log('Server aktif. Listening port 3000.');
```

Routing

/beranda

/profil

/berita

/galeri

/tentang

/kontak

Basic Routing
#1 Home Page

home.js

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

var server = http.createServer(function(req,res){
   console.log('Request : ' + req.url);
   if(req.url === '/home' || req.url === '/'){
      res.writeHead(200, {'Content-Type':'text/html'});
      fs.createReadStream(__dirname+'/home.html').pipe(res);
   }
});

server.listen(3000);
console.log('Server aktif, port 3000.');
```

**** Dengan cara serupa, dapat dibuat route lainnya!***

Purwadhika
Startup and Coding School

Basic Routing

#2 Error 404 Not Found

home.js

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

var server = http.createServer(function(req,res){
   console.log('Request : ' + req.url);
   if(req.url === '/home' || req.url === '/'){
      res.writeHead(200, {'Content-Type':'text/html'});
      fs.createReadStream(__dirname+'/home.html').pipe(res);
   } else {
      res.writeHead(404, {'Content-Type':'text/html'});
      fs.createReadStream(__dirname+'/404.html').pipe(res);
   }
});

server.listen(3000);
console.log('Server aktif, port 3000.');
```

Back-End Development

Exploring node JS

#3 HTTP Module

Purwadhika
Startup and Coding School