

17. fs, http, url Routing

Tuesday, June 6, 2023

3:08 PM

Sync = fs.readFileSync

Sync => Blocking (1 after another)

Async => Non-blocking => Heavy tasks as Callbacks

Heavy tasks:

-Networking

-File Access

```
const fs = require('fs');
// Sync
const textInput = fs.readFileSync('./txt/input.txt', (err, data) => {
  if (err) {
    console.log('error: ', err)
  }
  return data;
})

// const textOutput = textInput.toString();
const textOutput = `This is what we know about the avocado:\n
${textInput}.\nCreated on ${new Date()}`;
console.log('textOutput: ', textOutput);
// Save output to a file
// fs.writeFileSync('./txt/output.txt', textOutput);

// Async - Non-blocking I/O & Network requests
// PHP - 1 new thread for each user

// Callbacks != Async
// Callbacks are NOT Asynchronous by nature
fs.readFile('./txt/input.txt', 'utf-8', (err, data) => {
  if (err) {
    console.log('error: ', err);
  }
  console.log('data: ', data);
  return data;
});
```


ES6 Promises | | ES8 Async Await to tackle Callback hell

====Reading & Writing files asynchronously

```
// Async - Non-blocking I/O & Network requests
// PHP - 1 new thread for each user
// Callbacks != Async

// As soon as node reads this line => move on
fs.readFile('./txt/start.txt', 'utf-8', (err, data) => {
  if (err) console.log('error: ', err);
  console.log('data: ', data);
  return data;
});

// Finish reading this line first
console.log('Will read file');
```

```
$ node index.js
Will read file
data: read-this
```

// Callback hell

```
fs.readFile('./txt/start.txt', 'utf-8', (err, data1) => {
  if (err) return console.log('error: ', err);
  console.log('data1: ', data1);
  // return data1;
  fs.readFile(`./txt/${data1}.txt`, 'utf-8', (err, data2) => {
    if (err) return console.log('error2: ', err);
    console.log('data2: ', data2);
    fs.readFile('./txt/append.txt', 'utf-8', (err, data3) => {
      if (err) return console.log('error3: ', err);
      console.log('data3: ', data3);
      fs.writeFile('./txt/final.txt', `${data2}\n${data3}`,
        'utf-8', err => {
          if (err) return console.log('error: ', err);
          console.log('File has been written');
        })
    })
  })
});
```



```
console.log('Wil read file');
```

Wil read file

data1: read-this

data2: The avocado is also used as the base for the Mexican dip known as guacamole, as well as a spread on corn tortillas or toast, served with spices.

data3: APPENDIX: Generally, avocados are served raw, but some cultivars can be cooked for a short time without becoming bitter.

File has been written

```
JS App.js M JS server.js JS index.js ≡ final.txt X
complete-node-bootcamp > 1-node-farm > starter > txt > ≡ final.txt
1 The avocado is also used as the base for the Mexican dip known as guacamole
2 APPENDIX: Generally, avocados are served raw, but some cultivars can be cooked
```

====Creating a simple server

```
const fs = require('fs');
// Networking capabilities => Building a HTTP server
const http = require('http');

// Server
const server = http.createServer((req, res) => {
  console.log('req.body ', req.body);
  res.end('Hello from server')
});
// server.listen(port, localhost||loopback address)
server.listen(port=8880, localhost='127.0.0.1', () => {
  console.log(`Server is now listening on ${localhost}:${port}`);
})
// Listens to incoming requests from clients
```

```
$ node index.js
Server is now listening on 127.0.0.1:8880
req.body undefined
req.body undefined
```



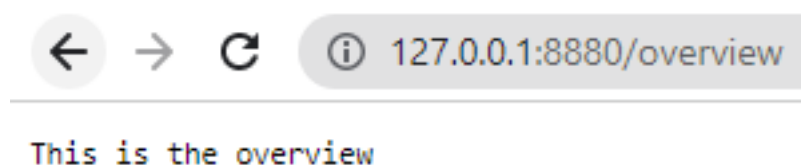
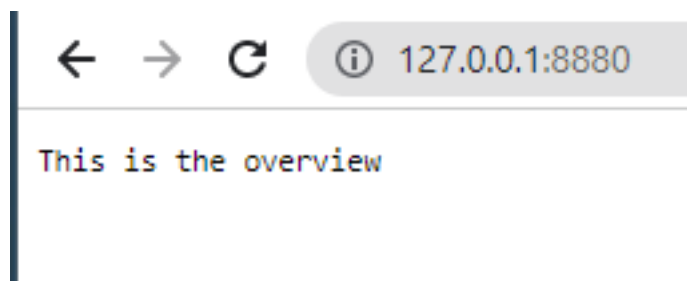
// After editing the script => ctrl + C => node script.js again

// After editing the script => ctrl + C => node script.js again

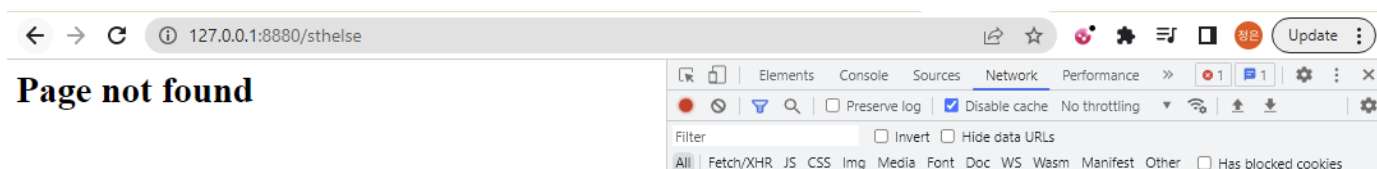
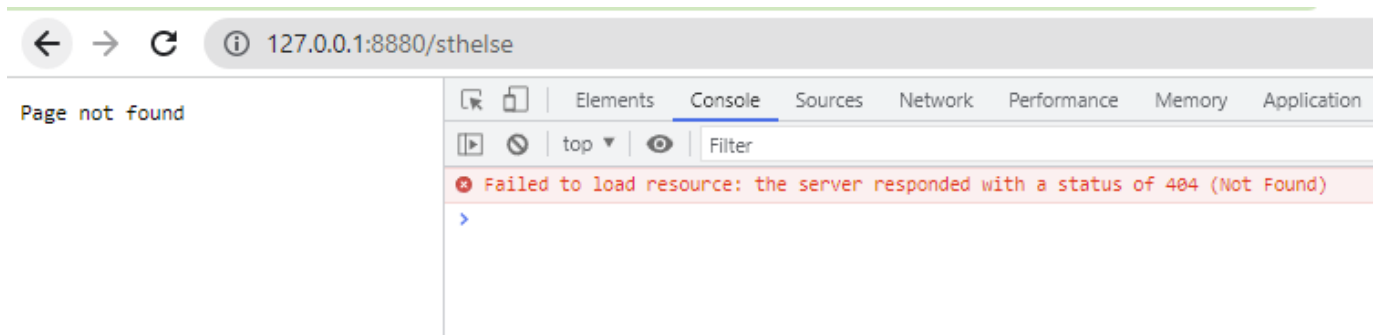
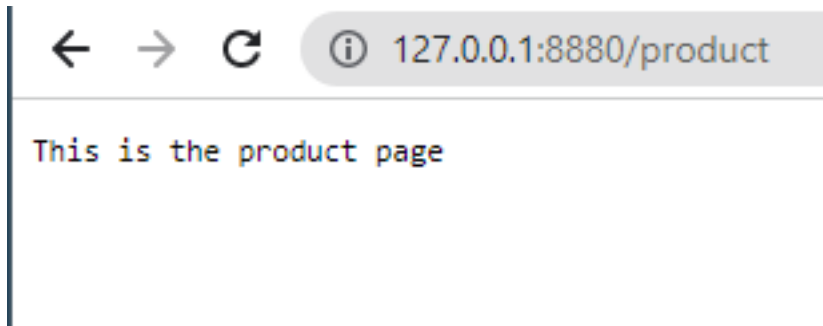
====Routing

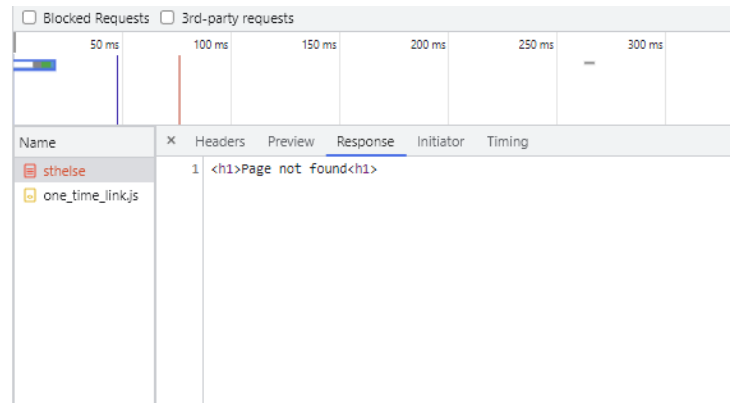
```
const fs = require('fs');
// Networking capabilities => Building a HTTP server
const http = require('http');
// URL module
const url = require('url');




// Server
const server = http.createServer((req, res) => {
  const pathName = req.url;
  if (pathName === '/' || pathName === '/overview') {
    res.end('This is the overview');
  } else if (pathName === '/product') {
    res.end('This is the product page');
  } else {
    res.writeHead(404);
    res.end('Page not found');
  }
});
// server.listen(port, localhost||loopback address)
server.listen(port=8880, localhost='127.0.0.1', () => {
  console.log(`Server is now listening on ${localhost}:
${port}`);
})
// Listens to incoming requests from clients
// 127.0.0.1:8880/product
```



Name	×	Headers	Preview	Response	Initiator	Timing
overview	▼	General				
one_time_link.js		Request URL:	http://127.0.0.1:8880/overview			
		Request Method:	GET			
		Status Code:	● 200 OK			
		Remote Address:	127.0.0.1:8880			
		Referrer Policy:	strict-origin-when-cross-origin			
	▼	Response Headers	<input type="checkbox"/> Raw			
		Connection:	keep-alive			
		Content-Length:	20			
		Date:	Tue, 06 Jun 2023 08:25:47 GMT			
		Keep-Alive:	timeout=5			
	▼	Request Headers	<input type="checkbox"/> Raw			
		Accept:	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application			
2 requests		320 B				





Name	×	Headers	Preview	Response	Initiator	Timing
 sthelse	▼ General					
 one_time_link.js	Request URL:		http://127.0.0.1:8880/sthelse			
	Request Method:		GET			
	Status Code:		 404 Not Found			
	Remote Address:		127.0.0.1:8880			
	Referrer Policy:		strict-origin-when-cross-origin			
	▼ Response Headers		<input type="checkbox"/> Raw			
	Connection:		keep-alive			
	Content-Type:		text/html			
	Date:		Tue, 06 Jun 2023 08:23:18 GMT			
	Keep-Alive:		timeout=5			
	My-Own-Header:		NOT FOUND :(
	Transfer-Encoding:		chunked			
	▶ Request Headers (16)					
2 requests 402 B 1						

====Building a simple API

```
const fs = require('fs');
// Networking capabilities => Building a HTTP server
const http = require('http');
// URL module
const url = require('url');

// Server
const server = http.createServer((req, res) => {
  const pathName = req.url;
  if (pathName === '/' || pathName === '/overview') {
```



```

    res.end('This is the overview');
  } else if (pathName === '/product') {
    res.end('This is the product page');
  } else if (pathName === '/api') {
    fs.readFile(`${__dirname}/dev-data/data.json`, 'utf-8',
(err, data) => {
      const productData = JSON.parse(data);
      res.writeHead(200, {
        'Content-type': 'application/json',
        'custom-header': 'Nice :D'
      })
      // console.log(productData);
      res.end(data); // need to send back string
    });

  } else {
    res.writeHead(404, {
      'Content-type': 'text/html',
      'my-own-header': 'NOT FOUND :( '
    });
    res.end('<h1>Page not found<h1>');
  }
});
// server.listen(port, localhost||loopback address)
server.listen(port=8880, localhost='127.0.0.1', () => {
  console.log(`Server is now listening on ${localhost}:
${port}`);
});

```

The screenshot shows a web browser at the address `127.0.0.1:8880/api`. The main content area displays a JSON array with two objects representing products:

```

[
  {
    id: 0,
    productName: "Fresh Avocados",
    image: "🥑",
    from: "Spain",
    nutrients: "Vitamin B, Vitamin K",
    quantity: "4 🥑",
    price: "6.50",
    organic: true,
    description: "A ripe avocado yields to gentle pressure when held in the palm of the hand and squeezed. The fruit is not sweet, but distinctly and subtly flavored, with smooth texture. The avocado is popular in vegetarian cuisine as a substitute for meats in sandwiches and salads because of its high fat content. Generally, avocado is served raw, though some cultivars, including the common 'Hass', can be cooked for a short time without becoming bitter. It is used as the base for the Mexican dip known as guacamole, as well as a spread on corn tortillas or toast, served with spices."
  },
  {
    id: 1,
    productName: "Goat and Sheep Cheese",
    image: "🧀",
    from: "Portugal",
    nutrients: "Vitamin A, Calcium",
    quantity: "250g",
    price: "5.00",
    organic: false,
    description: "Creamy and distinct in flavor, goat cheese is a dairy product enjoyed around the world. Goat cheese comes in a wide variety of flavors and textures, from soft and
  
```

The right-hand side of the browser window shows the Network tab with a list of requests. The selected request is for `api`. The Headers sub-tab is active, showing the following details:

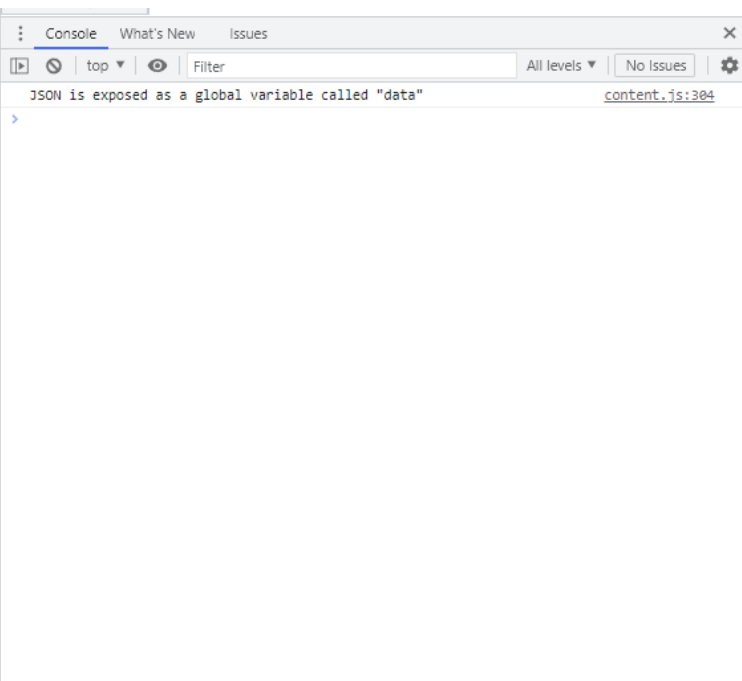
General	
Request URL:	http://127.0.0.1:8880/api
Request Method:	GET
Status Code:	200 OK
Remote Address:	127.0.0.1:8880
Referrer Policy:	strict-origin-when-cross-origin

Response Headers	
Connection:	keep-alive
Content-Type:	application/json
Custom-Header:	Nice :D
Date:	Tue, 06 Jun 2023 08:38:30 GMT
Keep-Alive:	timeout=5
Transfer-Encoding:	chunked

At the bottom of the Network tab, it indicates "3 requests" and "6.3 kB" of data transferred.


```
spreadable fresh cheese to salty, crumbly aged cheese.
Although it's made using the same coagulation and separation
process as cheese made from cow's milk, goat cheese differs
in nutrient content."
```

```
  },
  {
    id: 2,
    productName: "Apollo Broccoli",
    image: "🥦",
    from: "Portugal",
    nutrients: "Vitamin C, Vitamin K",
    quantity: "3 🥦",
    price: "5.50",
    organic: true,
    description: "Broccoli is known to be a hearty and tasty
    vegetable which is rich in dozens of nutrients. It is said to
    pack the most nutritional punch of any vegetable. When we
    think about green vegetables to include in our diet, broccoli
    is one of the foremost veggies to come to our mind. Broccoli
    is a cruciferous vegetable and part of the cabbage family,
    which includes vegetables such as Brussel sprouts and kale.
    Although the tastes are different, broccoli and these other
    vegetables are from the same family."
  },
  {
    id: 3,
    productName: "Baby Carrots",
    image: "🥕",
    from: "France",
    nutrients: "Vitamin A, Vitamin K",
    quantity: "20 🥕",
    price: "3.00",
    organic: true,
    description: "The carrot is a root vegetable that is often
```



```
// Only read data.json file once upon server starting using
fs.readFileSync()
// Only execute once
const data = fs.readFileSync(`${__dirname}/dev-data/data.json`,
'utf-8');
const dataObj = JSON.parse(data);
const server = http.createServer((req, res) => {
  const pathName = req.url;
  if (pathName === '/' || pathName === '/overview') {
    res.end('This is the overview');
  } else if (pathName === '/product') {
    res.end('This is the product page');
  } else if (pathName === '/api') {

    res.writeHead(200, {
      'Content-type': 'application/json',
      'custom-header': 'Nice :D'
    })
    // console.log(productData);
    res.end(data); // need to send back string

  } else {
    res.writeHead(404, {
      'Content-type': 'text/html',
      'my-own-header': 'NOT FOUND :('
    });
    res.end('<h1>Page not found<h1>');
  }
},
```



```

    }
  });
  // server.listen(port, localhost||loopback address)
  server.listen(port=8880, localhost='127.0.0.1', () => {
    console.log(`Server is now listening on ${localhost}:
    ${port}`);
  })
}

```

====Building HTML templates

./templates/template-product.html

./templates/template-overview.html

./templates/template-card.html

index.js:

```

// Server
// Top-level code only executes once
// Can only use Sync for top-level code
const tempOverview = fs.readFileSync(`
${__dirname}/templates/template-overview.html`, 'utf-8');
const tempCard = fs.readFileSync(`
${__dirname}/templates/template-card.html`, 'utf-8');
const tempProduct = fs.readFileSync(`
${__dirname}/templates/template-product.html`, 'utf-8');
const data = fs.readFileSync(`${__dirname}/dev-data/data.json`,
'utf-8');
const dataObj = JSON.parse(data);

const server = http.createServer((req, res) => {
  const pathName = req.url;
  // /overview page
  if (pathName === '/' || pathName === '/overview') {
    // Whenever receiving 127.0.0.1/overview request
    // return from memory
    res.writeHead(200, {
      'Content-type': 'text/html'
    })
    res.end(tempOverview);
  }
  // /product page
  } else if (pathName === '/product') {
    res.end('This is the product page');
  }
  // /api page
  } else if (pathName === '/api') {

```

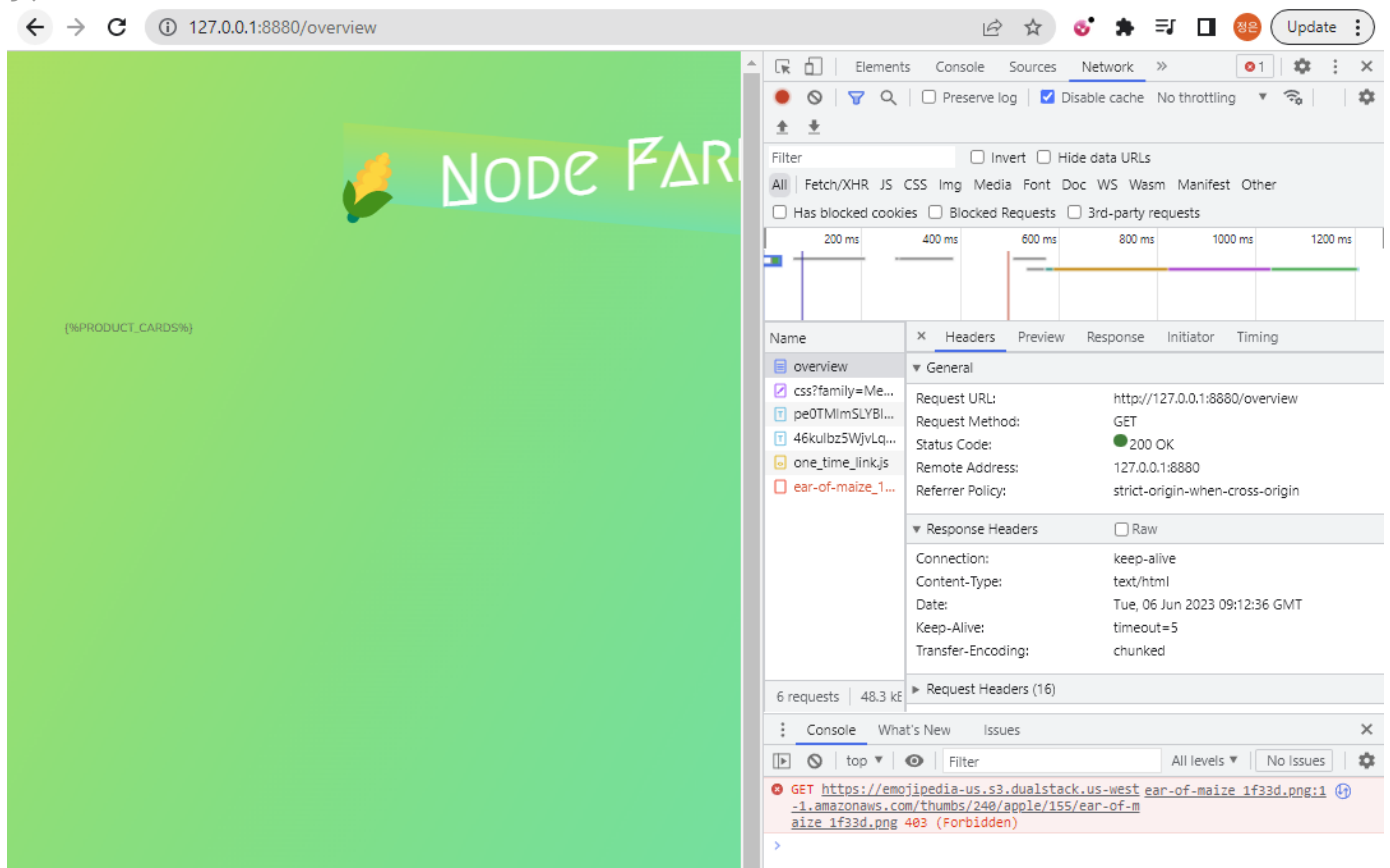


```

res.writeHead(200, {
  'Content-type': 'application/json',
  'custom-header': 'Nice :D'
})
// console.log(productData);
res.end(data); // need to send data back as string

// 404 NOT FOUND
} else {
  res.writeHead(404, {
    'Content-type': 'text/html',
    'my-own-header': 'NOT FOUND :('
  });
  res.end('<h1>Page not found<h1>');
}
});
// server.listen(port, localhost||loopback address)
server.listen(port=8880, localhost='127.0.0.1', () => {
  console.log(`Server is now listening on ${localhost}:
${port}`);
});

```



```
// Server
```



```

// server
const replaceTemplate = (template-xx.html, object) => {...};

const replaceTemplate = (temp, product) => {
  // Avoid direct mutation of original template by storing
  temp in a new var
  let output = temp.replace(/{%PRODUCTNAME%}/g,
product.productName);

  // this function mutates all {%}% in template-xx.html with
  each obj in json

  // const cardsHtml = dataObj.map(element =>
  replaceTemplate(tempCard, element)).join('');

  // Start mutating now
  // Replacing all {%}% with
  output = output.replace(/{%IMAGE%}/g, product.image);
  output = output.replace(/{%PRICE%}/g, product.price);
  output = output.replace(/{%FROM%}/g, product.from);
  output = output.replace(/{%NUTRIENTS%}/g,
product.nutrients);
  output = output.replace(/{%QUANTITY%}/g, product.quantity);
  output = output.replace(/{%DESCRIPTION%}/g,
product.description);
  output = output.replace(/{%ID%}/g, product.id);
  if (!product.organic) output = output.replace(/{%
NOT_ORGANIC%}/g, 'not-organic')
  return output; // output final HTML template after replacing
}

// Top-level code only executes once
// Can only use Sync for top-level code
const tempOverview = fs.readFileSync(`
${__dirname}/templates/template-overview.html`, 'utf-8');
const tempCard = fs.readFileSync(`
${__dirname}/templates/template-card.html`, 'utf-8');
const tempProduct = fs.readFileSync(`
${__dirname}/templates/template-product.html`, 'utf-8');
const data = fs.readFileSync(`${__dirname}/dev-data/data.json`,
'utf-8');
const dataObj = JSON.parse(data);

const server = http.createServer((req, res) => {
  const pathName = req.url;

```



```

// /overview page
if (pathName === '/' || pathName === '/overview') {
  // 1. Whenever receiving 127.0.0.1/overview request
  // return from memory
  console.log(req.url);

  // 2. Write HTTP Header 200 OK
  res.writeHead(200, {
    'Content-type': 'text/html'
  });

  // 3. Loop through dataObj to return each element
  // join all elements 1 after another to display all of
  them
  const cardsHtml = dataObj.map(element =>
replaceTemplate(tempCard, element)).join('');
  // console.log('cardsHtml: ', cardsHtml);

  const output = tempOverview.replace('{%PRODUCT_CARDS%}',
cardsHtml);
  // 4. Return template-overview.html with {%}% all
replaced
  // by json obj & props
  res.end(output);

  // /product page
} else if (pathName === '/product') {
  res.end('This is the product page');
  // /api page
} else if (pathName === '/api') {

  res.writeHead(200, {
    'Content-type': 'application/json',
    'custom-header': 'Nice :D'
  })
  // console.log(productData);
  res.end(data); // need to send data back as string

  // 404 NOT FOUND
} else {
  res.writeHead(404, {
    'Content-type': 'text/html',
    'my-own-header': 'NOT FOUND :('
  });
  res.end('<h1>Page not found<h1>');
}

```



```

    }
  });
  // server.listen(port, localhost||loopback address)
  server.listen(port=8880, localhost='127.0.0.1', () => {
    console.log(`Server is now listening on ${localhost}:
    ${port}`);
  })
})

```

====Parsing variables from URLs

```

const fs = require('fs');
// Networking capabilities => Building a HTTP server
const http = require('http');
// URL module
const url = require('url');

// Server
// Arrow function to replace {%} with product props in
data.json
const replaceTemplate = (temp, product) => {
  // Avoid direct mutation of original template by storing
temp in a new var
  let output = temp.replace(/{%PRODUCTNAME%}/g,
product.productName);
  // this function mutates all {%} in template-xx.html with
each obj in json
  // const cardsHtml = dataObj.map(element =>
replaceTemplate(tempCard, element)).join('');
  // Start mutating now
  // Replacing all {%} with
  output = output.replace(/{%IMAGE%}/g, product.image);
  output = output.replace(/{%PRICE%}/g, product.price);
  output = output.replace(/{%FROM%}/g, product.from);
  output = output.replace(/{%NUTRIENTS%}/g,
product.nutrients);
  output = output.replace(/{%QUANTITY%}/g, product.quantity);
  output = output.replace(/{%DESCRIPTION%}/g,
product.description);
  output = output.replace(/{%ID%}/g, product.id);
  if (!product.organic) output = output.replace(/{%
NOT_ORGANIC%}/g, 'not-organic')
  return output; // output final Html after replacing
}
// Top level code only executes once

```



```
// top-level code only executes once
// Can only use Sync for top-level code
const tempOverview = fs.readFileSync(`
${__dirname}/templates/template-overview.html`, 'utf-8');
const tempCard = fs.readFileSync(`
${__dirname}/templates/template-card.html`, 'utf-8');
const tempProduct = fs.readFileSync(`
${__dirname}/templates/template-product.html`, 'utf-8');
const data = fs.readFileSync(`${__dirname}/dev-data/data.json`,
'utf-8');
const dataObj = JSON.parse(data);
```

```
const server = http.createServer((req, res) => {
  console.log('req.url ', req.url);
  // To parse 127.0.0.1:8880/product?id=0
  // parsing request Query Strings to an object
  console.log('url.parse(req.url, true) ', url.parse(req.url,
true));
```

```
/
req.url /product?id=0
url.parse(req.url, true) Url {
  protocol: null,
  slashes: null,
  auth: null,
  host: null,
  port: null,
  hostname: null,
  hash: null,
  search: '?id=0',
  query: [Object: null prototype] { id: '0' },
  pathname: '/product',
  path: '/product?id=0',
  href: '/product?id=0'
}
```

```
const pathName = req.url;
// /overview page
if (pathName === '/' || pathName === '/overview') {
  // Whenever receiving 127.0.0.1/overview request
  // return from memory
  console.log(req.url);
  res.writeHead(200, {
    'Content-type': 'text/html'
  });
  // loop through dataObj to return each object
  const cardHtml = dataObj.map(element =>
```



```

        const cardsHtml = dataArray.map(element =>
replaceTemplate(tempCard, element)).join('');
        // console.log('cardsHtml: ', cardsHtml);
        const output = tempOverview.replace('{%PRODUCT_CARDS%}',
cardsHtml);
        res.end(output);
    // /product page
    } else if (pathname === '/product') {
        res.end('This is the product page');
    // /api page
    } else if (pathname === '/api') {

        res.writeHead(200, {
            'Content-type': 'application/json',
            'custom-header': 'Nice :D'
        });
        // console.log(productData);
        res.end(data); // need to send data back as string

    // 404 NOT FOUND
    } else {
        res.writeHead(404, {
            'Content-type': 'text/html',
            'my-own-header': 'NOT FOUND :('
        });
        res.end('<h1>Page not found<h1>');
    }
});
// server.listen(port, localhost||loopback address)
server.listen(port=8880, localhost='127.0.0.1', () => {
    console.log(`Server is now listening on ${localhost}:
${port}`);
})

```

----returning response to clients for Query
127.0.0.1:8880/product?id=

```

else if (pathname === '/product') {
    // console.log('query: \n', query);

    // write a HTTP Header for 127.0.0.1:8880/product?id=
    res.writeHead(200, {'Content-type': 'text/html'});

    // if request Query Strings is: 127.0.0.1:8880/product?

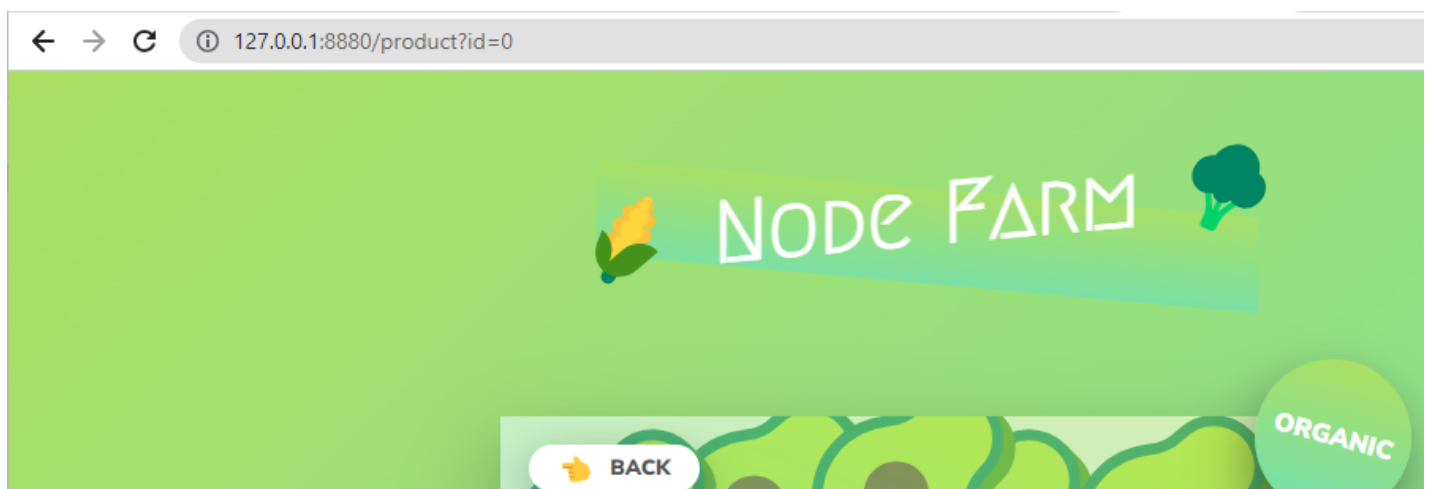
```


id=0

```
// dataObj[0] will return
// {
//   "id": 0,
//   "productName": "Fresh Avocados",
//   "image": "🥑",
//   "from": "Spain",
//   "nutrients": "Vitamin B, Vitamin K",
//   "quantity": "4 🥑",
//   "price": "6.50",
//   "organic": true,
//   "description": "A ripe avocado yields to gentle
pressure when held in the palm of the hand and squeezed. The
fruit is not sweet, but distinctly and subtly flavored, with
smooth texture. The avocado is popular in vegetarian cuisine as
a substitute for meats in sandwiches and salads because of its
high fat content. Generally, avocado is served raw, though some
cultivars, including the common 'Hass', can be cooked for a
short time without becoming bitter. It is used as the base for
the Mexican dip known as guacamole, as well as a spread on corn
tortillas or toast, served with spices."
// }
// and so on for id: 1, 2, 3, ...
const product = dataObj[query.id];


// Replace {%} in template-product.html with data obj
in data.json
const output = replaceTemplate(tempProduct, product);


// Send response of replaced template-product.html to
clients
res.end(output);
```







FRESH AVOCADOS

 Spain

 Vitamin B, Vitamin K

 4 

 6.50€



ADD TO SHOPPING CARD (6.50€)

A ripe avocado yields to gentle pressure when held in the palm of the hand and squeezed. The fruit is not sweet, but distinctly and subtly flavored, with smooth texture. The avocado is popular in vegetarian cuisine as a substitute for meats in sandwiches and salads because of its high fat content. Generally, avocado is served raw, though some cultivars, including the common 'Hass', can be cooked for a short time without becoming bitter. It is used as the base for the Mexican dip known as guacamole, as well as a spread on corn tortillas or toast, served with spices.

