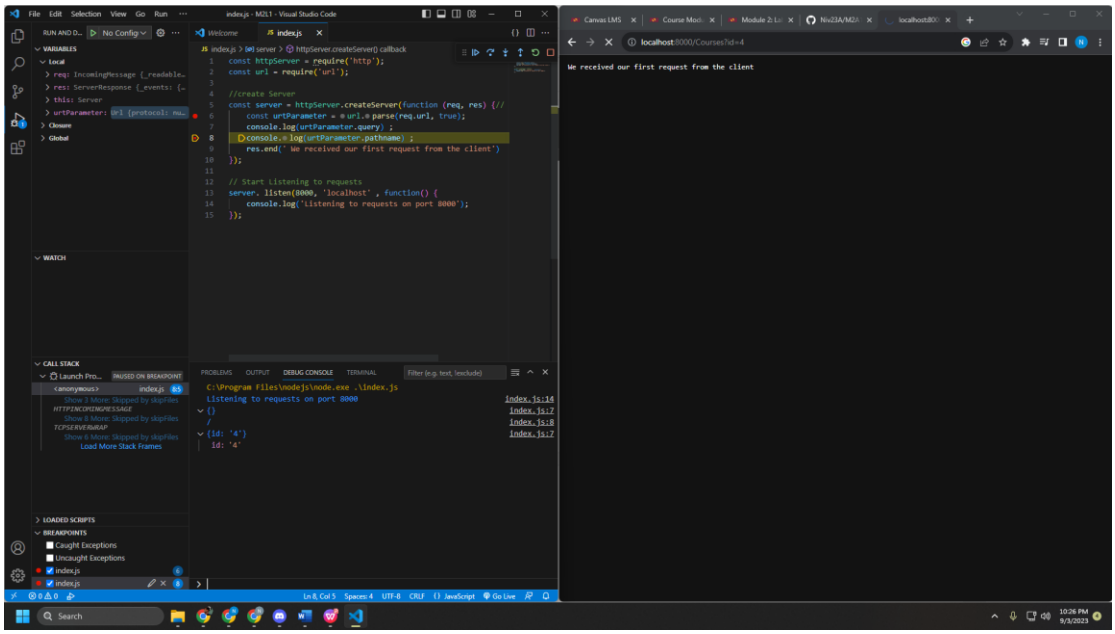
IFT544: Middleware Prog & Database Security

Module 2: Lab 1, Java Script

Author: Nived Abdulsathar (1225125746)
Instructor: **Dinesh Sthapit**
Date of Submission: September 3, 2023

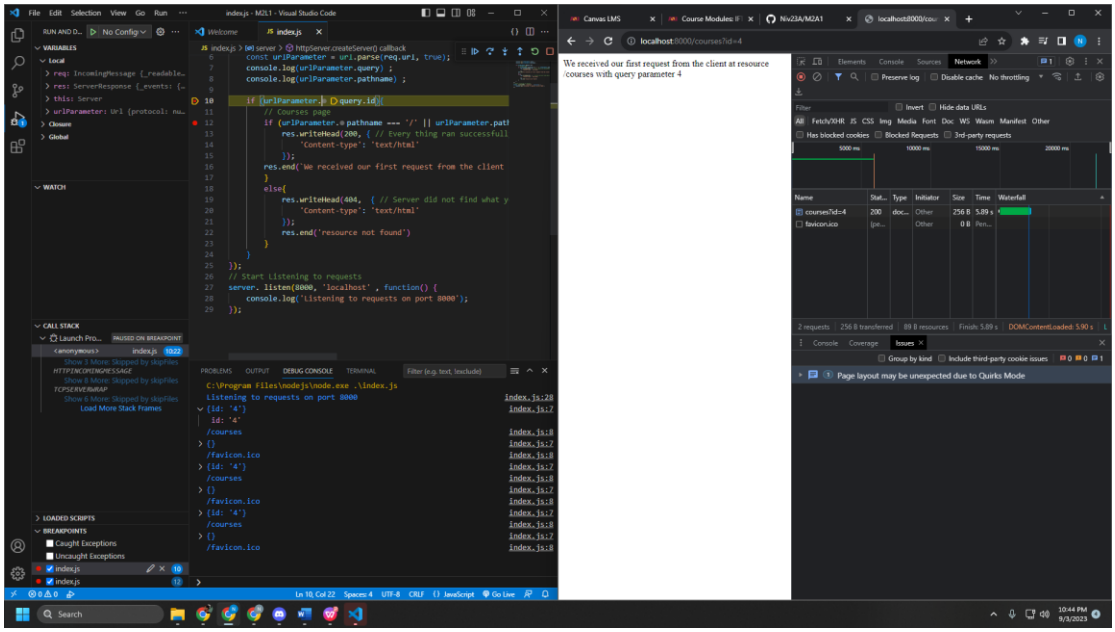
Part A

```
const httpServer = require('http');
const url = require('url');

//create Server
const server = httpServer.createServer(function (req, res) {// call
back function
    const urtParameter = url.parse(req.url, true);
    console.log(urtParameter.query) ;
    console.log(urtParameter.pathname) ;
    res.end(' We received our first request from the client')
});

// Start Listening to requests
server. listen(8000, 'localhost' , function() {
    console.log('Listening to requests on port 8000');
});
```



Part B

```
const httpServer = require('http');
const url = require('url');

//create Server
const server = httpServer.createServer(function (req, res) {// call
back function
    const urlParameter = url.parse(req.url, true);
    console.log(urlParameter.query) ;
    console.log(urlParameter.pathname) ;

    if (urlParameter.query.id){
        // Courses page
        if (urlParameter.pathname === '/' ||
urlParameter.pathname.toLowerCase() === '/courses') {
            res.writeHead(200, { // Every thing ran successfully
                'Content-type': 'text/html'
            });
        res.end(`We received our first request from the client at
resource ${urlParameter.pathname.toLowerCase()} with query parameter
${urlParameter.query.id}` )
        }
        else{
            res.writeHead(404,  { // Server did not find what you were
looking for
                'Content-type': 'text/html'
            });
            res.end('resource not found')
        }
    }
});
// Start Listening to requests
server. listen(8000, 'localhost' , function() {
    console.log('Listening to requests on port 8000');
});
```


index.js - M2L1 - Visual Studio Code
const urlParameter = url.parse(req.url, true);
console.log(urlParameter.query) ;
console.log(urlParameter.pathname) ;
if (urlParameter.p query.id){
// Courses page
if (urlParameter.pathname === '/' || urlParameter.path
res.writeHead(200, { // Every thing ran successfull
'Content-type': 'text/html'
});
res.end('We received our first request from the client
}
else{
res.writeHead(404, { // Server did not find what y
'Content-type': 'text/html'
});
res.end('resource not found')
}
}
});
// Start Listening to requests
server. listen(8000, 'localhost' , function() {
console.log('listening to requests on port 8000');
});
C:\Program Files\nodejs\node.exe .\index.js
Listening to requests on port 8000
{id: '4'}
id: '4'
/courses
{}
/favicon.ico
{id: '4'}
/courses
{}
/favicon.ico
{id: '4'}
/courses
{}
/favicon.ico
localhost:8000/courses?id=4
We received our first request from the client at resource courses with query parameter 4
Page layout may be unexpected due to Quirks Mode

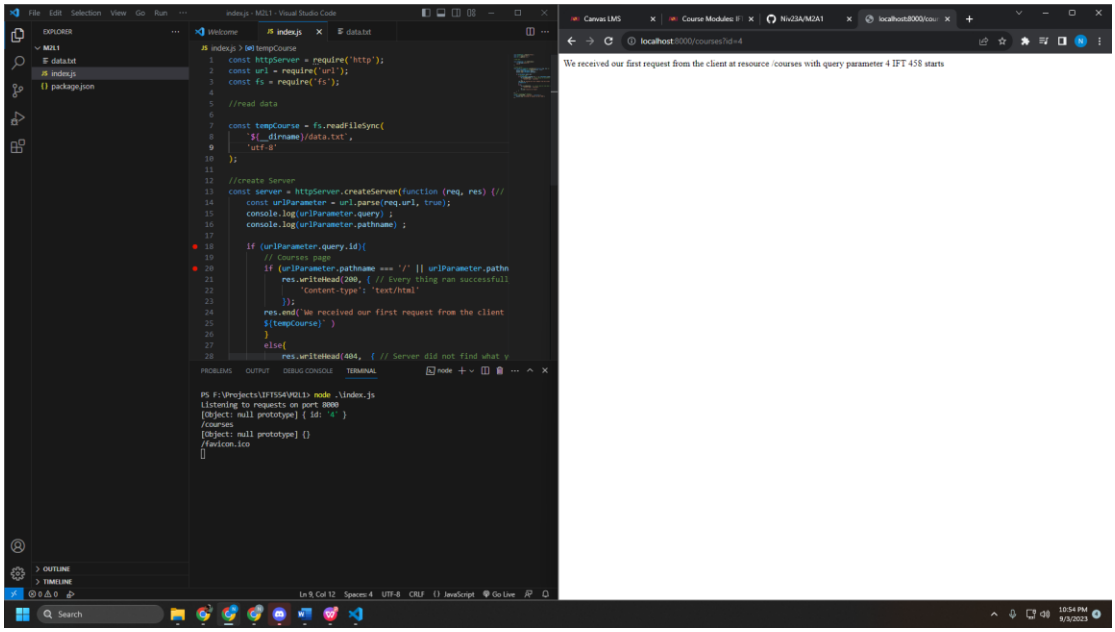
Step 6: Reading Data From File and Sending the Data to the Client

```
const httpServer = require('http');
const url = require('url');
const fs = require('fs');

//read data

const tempCourse = fs.readFileSync(
    `${__dirname}/data.txt`,
    'utf-8'
);

//create Server
const server = httpServer.createServer(function (req, res) {// call
back function
    const urlParameter = url.parse(req.url, true);
    console.log(urlParameter.query) ;
    console.log(urlParameter.pathname) ;

    if (urlParameter.query.id){
        // Courses page
        if (urlParameter.pathname === '/' ||
urlParameter.pathname.toLowerCase() === '/courses') {
            res.writeHead(200, { // Every thing ran successfully
                'Content-type': 'text/html'
            });
        res.end(`We received our first request from the client at
resource ${urlParameter.pathname.toLowerCase()} with query parameter
${urlParameter.query.id}
        ${tempCourse}` )
        }
        else{
            res.writeHead(404,  { // Server did not find what you were
looking for
                'Content-type': 'text/html'
            });
            res.end('resource not found')
        }
    }
});
// Start Listening to requests
server. listen(8000, 'localhost' , function() {
    console.log('Listening to requests on port 8000');
});
```

Data.txt :

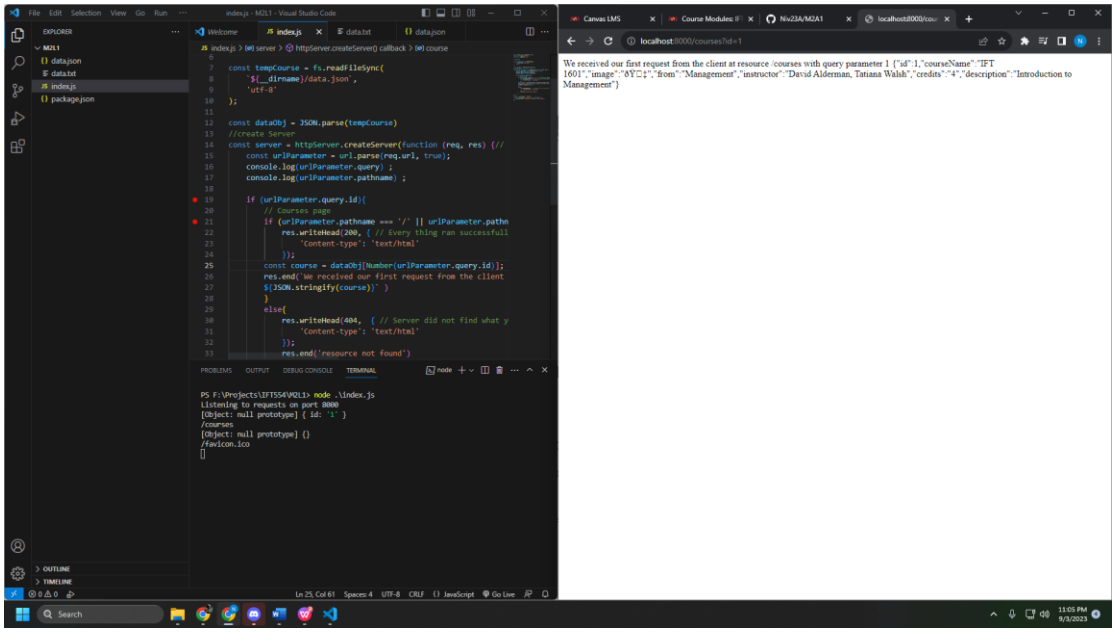
IFT 458

starts


index.js - M2L1 - Visual Studio Code
EXPLORER
M2L1
data.txt
index.js
package.json
Welcome
index.js
data.txt
const httpServer = require('http');
const url = require('url');
const fs = require('fs');
//read data
const tempCourse = fs.readFileSync(
`${__dirname}/data.txt`,
'utf-8'
);
//create Server
const server = httpServer.createServer(function (req, res) {//
const urlParameter = url.parse(req.url, true);
console.log(urlParameter.query) ;
console.log(urlParameter.pathname) ;
if (urlParameter.query.id){
// Courses page
if (urlParameter.pathname === '/' || urlParameter.pathn
res.writeHead(200, { // Every thing ran successfull
'Content-type': 'text/html'
});
res.end('We received our first request from the client
${tempCourse}`)
}
else{
res.writeHead(404, { // Server did not find what y
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
node
PS F:\Projects\IFT554\M2L1> node .\index.js
Listening to requests on port 8000
[Object: null prototype] { id: '4' }
/courses
[Object: null prototype] {}
/favicon.ico
OUTLINE
TIMELINE
Ln 9, Col 12
Spaces: 4
UTF-8
CRLF
JavaScript
Go Live
Canvas LMS
Course Modules: F
Niv23A/M2A1
localhost:8000/cou
localhost:8000/courses?id=4
We received our first request from the client at resource /courses with query parameter 4 IFT 458 starts
Search
10:54 PM
9/3/2023

Step 7: Reading JSON data from the file

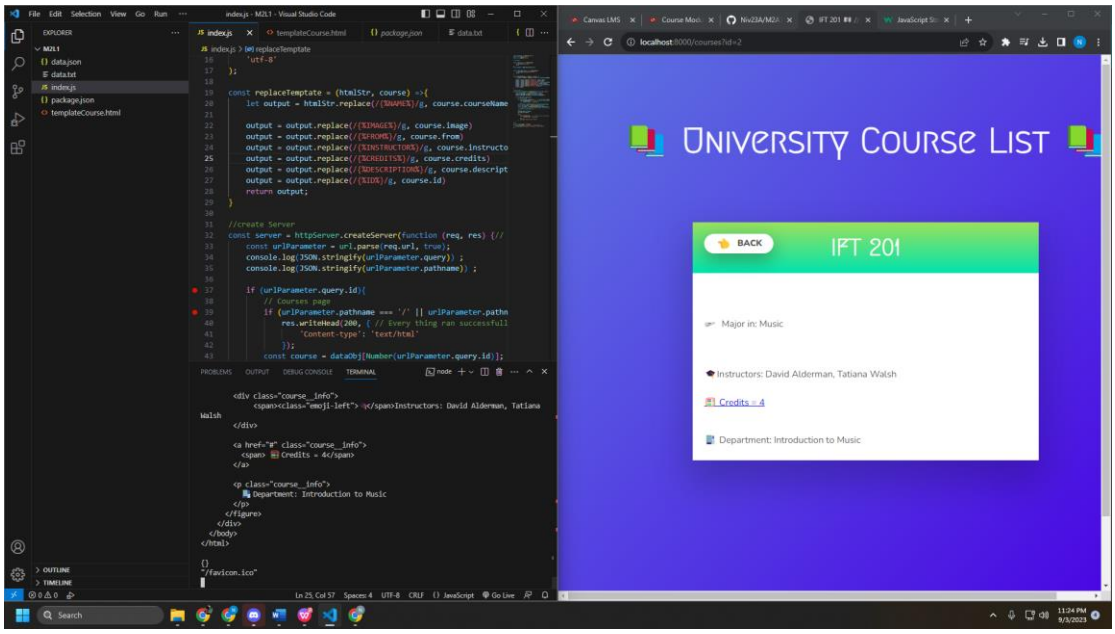
```
const httpServer = require('http');
const url = require('url');
const fs = require('fs');

//read data

const tempCourse = fs.readFileSync(
    `${__dirname}/data.json`,
    'utf-8'
);

const dataObj = JSON.parse(tempCourse)
//create Server
const server = httpServer.createServer(function (req, res) {// call
back function
    const urlParameter = url.parse(req.url, true);
    console.log(urlParameter.query) ;
    console.log(urlParameter.pathname) ;

    if (urlParameter.query.id){
        // Courses page
        if (urlParameter.pathname === '/' ||
urlParameter.pathname.toLowerCase() === '/courses') {
            res.writeHead(200, { // Every thing ran successfully
                'Content-type': 'text/html'
            });
        const course = dataObj[Number(urlParameter.query.id)];
        res.end(`We received our first request from the client at
resource ${urlParameter.pathname.toLowerCase()} with query parameter
${urlParameter.query.id}
        ${JSON.stringify(course)}` )
        }
        else{
            res.writeHead(404,  { // Server did not find what you were
looking for
                'Content-type': 'text/html'
            });
            res.end('resource not found')
        }
    }
});
// Start Listening to requests
server. listen(8000, 'localhost' , function() {
    console.log('Listening to requests on port 8000');
});
```


File Edit Selection View Go Run
index.js - M2L1 - Visual Studio Code
EXPLORER
M2L1
data.json
data.txt
index.js
package.json
Welcome
index.js
data.txt
data.json
const tempCourse = fs.readFileSync(
`${__dirname}/data.json`,
'utf-8'
);
const dataObj = JSON.parse(tempCourse)
//create Server
const server = httpServer.createServer(function (req, res) {//
const urlParameter = url.parse(req.url, true);
console.log(urlParameter.query) ;
console.log(urlParameter.pathname) ;
if (urlParameter.query.id){
// Courses page
if (urlParameter.pathname === '/' || urlParameter.pathn
res.writeHead(200, { // Every thing ran successfull
'Content-type': 'text/html'
});
const course = dataObj[Number(urlParameter.query.id)];
res.end(`We received our first request from the client
${JSON.stringify(course)}`)
}
else{
res.writeHead(404, { // Server did not find what y
'Content-type': 'text/html'
});
res.end('resource not found')
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
node
PS F:\Projects\IFTSS4\M2L1> node .\index.js
Listening to requests on port 8000
[Object: null prototype] { id: '1' }
/courses
[Object: null prototype] {}
/favicon.ico
OUTLINE
TIMELINE
Ln 25, Col 61
Spaces: 4
UTF-8
CRLF
JavaScript
Go Live
Canvas LMS
Course Modules: F
Niv23A/M2A1
localhost:8000/cou
localhost:8000/courses?id=1
We received our first request from the client at resource /courses with query parameter 1 {"id":1,"courseName":"IFT 1601","image":"ðŸ‘¨â€ðŸ«","from":"Management","instructor":"David Alderman, Tatiana Walsh","credits":"4","description":"Introduction to Management"}
Search
11:05 PM
9/3/2023

Step 8: Adding a User Interface in HTML:

```
const httpServer = require('http');
const url = require('url');
const fs = require('fs');

//read data

const tempCourse = fs.readFileSync(
    `${__dirname}/data.json`,
    'utf-8'
);
const dataObj = JSON.parse(tempCourse)


const templateHTMLCourse = fs.readFileSync(
    `${__dirname}/templateCourse.html`,
    'utf-8'
);

const replaceTemptate = (htmlStr, course) =>{
    let output = htmlStr.replace(/{%NAME%}/g, course.courseName)

    output = output.replace(/{%IMAGE%}/g, course.image)
    output = output.replace(/{%FROM%}/g, course.from)
    output = output.replace(/{%INSTRUCTOR%}/g, course.instructor)
    output = output.replace(/{%CREDITS%}/g, course.credits)
    output = output.replace(/{%DESCRIPTION%}/g, course.description)
    output = output.replace(/{%ID%}/g, course.id)
    return output;
}

//create Server
const server = httpServer.createServer(function (req, res) {// call
back function
    const urlParameter = url.parse(req.url, true);
    console.log(JSON.stringify(urlParameter.query)) ;
    console.log(JSON.stringify(urlParameter.pathname)) ;

    if (urlParameter.query.id){
        // Courses page
        if (urlParameter.pathname === '/' ||
urlParameter.pathname.toLowerCase() === '/courses') {
            res.writeHead(200, { // Every thing ran successfully
                'Content-type': 'text/html'
            });
        const course = dataObj[Number(urlParameter.query.id)];
        const strCourseName = JSON.stringify(course);
        const courseHTML = replaceTemptate(templateHTMLCourse, course);
```

```
        // res.end(`We received our first request from the client at
resource ${urlParameter.pathname.toLowerCase()} with query parameter
${urlParameter.query.id}
        // ${JSON.stringify(course)}` )
        // }
        console.log(courseHTML)
        res.end(courseHTML);
        }
        else{
            res.writeHead(404,  { // Server did not find what you were
looking for
                'Content-type': 'text/html'
            });
            res.end('resource not found')
        }
    }
});
// Start Listening to requests
server. listen(8000, 'localhost' , function() {
    console.log('Listening to requests on port 8000');
});
```



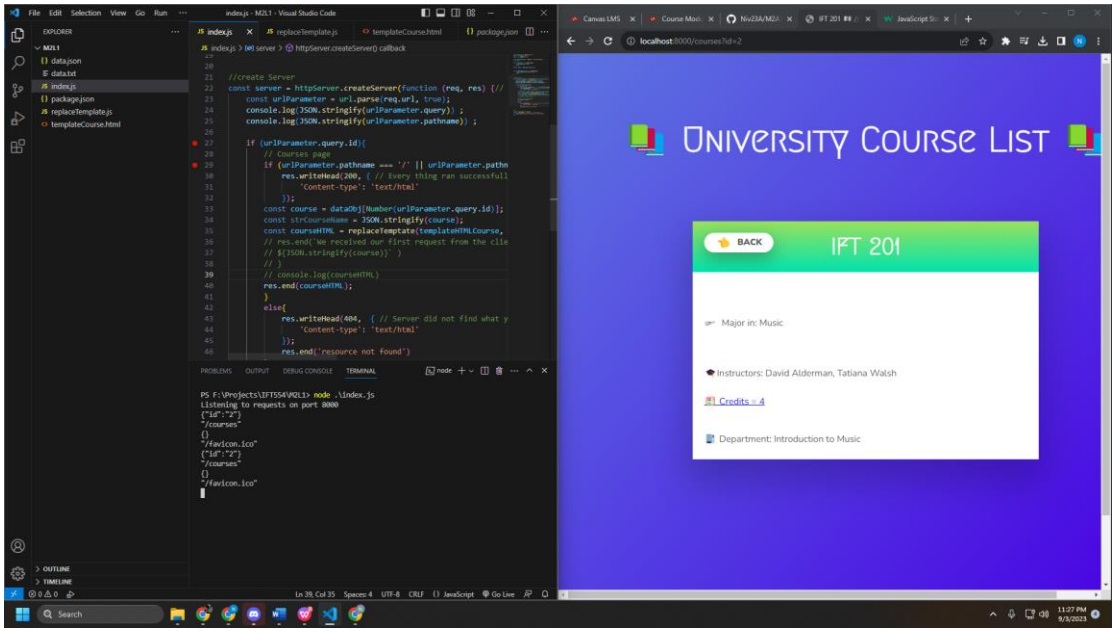
Step 9: Module Export (reusable functions)

```
const httpServer = require('http');
const url = require('url');
const fs = require('fs');

const replaceTemptate = require('./replaceTemplate')
//read data

const tempCourse = fs.readFileSync(
    `${__dirname}/data.json`,
    'utf-8'
);
const dataObj = JSON.parse(tempCourse)


const templateHTMLCourse = fs.readFileSync(
    `${__dirname}/templateCourse.html`,
    'utf-8'
);


//create Server
const server = httpServer.createServer(function (req, res) {// call
back function
    const urlParameter = url.parse(req.url, true);
    console.log(JSON.stringify(urlParameter.query)) ;
    console.log(JSON.stringify(urlParameter.pathname)) ;

    if (urlParameter.query.id){
        // Courses page
        if (urlParameter.pathname === '/' ||
urlParameter.pathname.toLowerCase() === '/courses') {
            res.writeHead(200, { // Every thing ran successfully
                'Content-type': 'text/html'
            });
        const course = dataObj[Number(urlParameter.query.id)];
        const strCourseName = JSON.stringify(course);
        const courseHTML = replaceTemptate(templateHTMLCourse, course);
        // res.end(`We received our first request from the client at
resource ${urlParameter.pathname.toLowerCase()} with query parameter
${urlParameter.query.id}
        // ${JSON.stringify(course)}` )
        // }
        // console.log(courseHTML)
        res.end(courseHTML);
        }
        else{
```

```
            res.writeHead(404,  { // Server did not find what you were
looking for
                'Content-type': 'text/html'
            });
            res.end('resource not found')
        }
    }
});
// Start Listening to requests
server. listen(8000, 'localhost' , function() {
    console.log('Listening to requests on port 8000');
});
```

replaceTemplate.js:

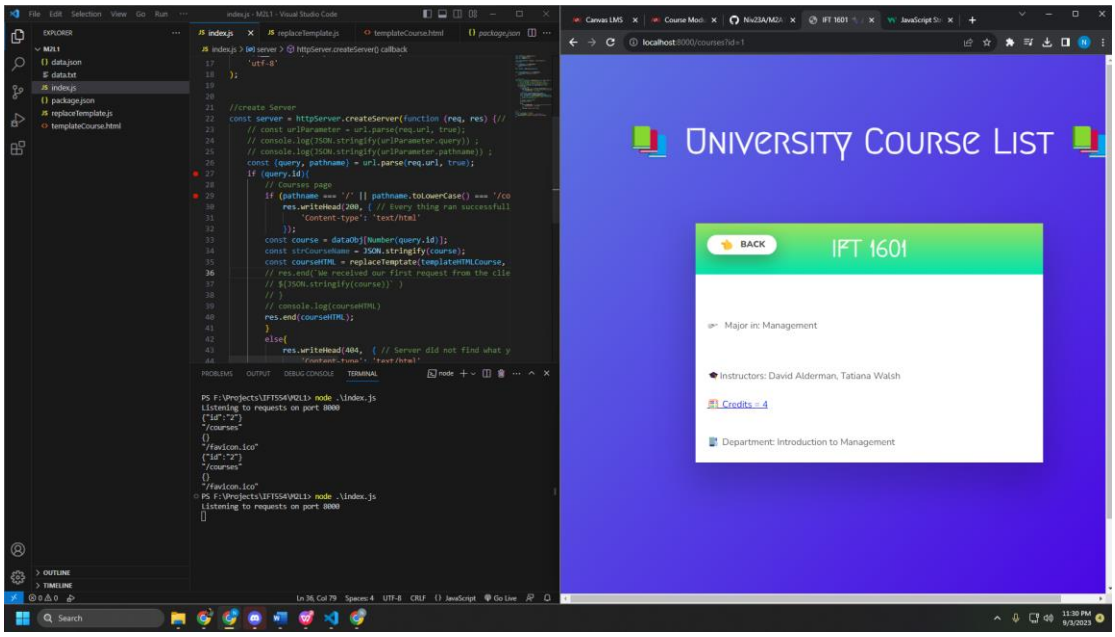
```
module.exports  = (htmlStr, course) =>{
    let output = htmlStr.replace(/{%NAME%}/g, course.courseName)

    output = output.replace(/{%IMAGE%}/g, course.image)
    output = output.replace(/{%FROM%}/g, course.from)
    output = output.replace(/{%INSTRUCTOR%}/g, course.instructor)
    output = output.replace(/{%CREDITS%}/g, course.credits)
    output = output.replace(/{%DESCRIPTION%}/g, course.description)
    output = output.replace(/{%ID%}/g, course.id)
    return output;
}
```



Step 10: Object Destruction

```
const httpServer = require('http');
const url = require('url');
const fs = require('fs');

const replaceTemptate = require('./replaceTemplate')
//read data

const tempCourse = fs.readFileSync(
    `${__dirname}/data.json`,
    'utf-8'
);
const dataObj = JSON.parse(tempCourse)


const templateHTMLCourse = fs.readFileSync(
    `${__dirname}/templateCourse.html`,
    'utf-8'
);


//create Server
const server = httpServer.createServer(function (req, res) {// call
back function
    // const urlParameter = url.parse(req.url, true);
    // console.log(JSON.stringify(urlParameter.query)) ;
    // console.log(JSON.stringify(urlParameter.pathname)) ;
    const {query, pathname} = url.parse(req.url, true);
    if (query.id){
        // Courses page
        if (pathname === '/' || pathname.toLowerCase() === '/courses')
{
            res.writeHead(200, { // Every thing ran successfully
                'Content-type': 'text/html'
            });
        const course = dataObj[Number(query.id)];
        const strCourseName = JSON.stringify(course);
        const courseHTML = replaceTemptate(templateHTMLCourse, course);
        // res.end(`We received our first request from the client at
resource ${urlParameter.pathname.toLowerCase()} with query parameter
${urlParameter.query.id}
        // ${JSON.stringify(course)}` )
        // }
        // console.log(courseHTML)
        res.end(courseHTML);
        }
        else{
```

```
            res.writeHead(404,  { // Server did not find what you were
looking for
                'Content-type': 'text/html'
            });
            res.end('resource not found')
        }
    }
});
// Start Listening to requests
server. listen(8000, 'localhost' , function() {
    console.log('Listening to requests on port 8000');
});
```



Step 11: Refactoring and re-arranging the folder structure.

```
const httpServer = require('http');
const url = require('url');
const fs = require('fs');

const replaceTemptate = require('./template/replaceTemplate')
//read data

const tempCourse = fs.readFileSync(
    `${__dirname}/data/data.json`,
    'utf-8'
);
const dataObj = JSON.parse(tempCourse)


const templateHTMLCourse = fs.readFileSync(
    `${__dirname}//template/templateCourse.html`,
    'utf-8'
);


//create Server
const server = httpServer.createServer(function (req, res) {// call
back function
    // const urlParameter = url.parse(req.url, true);
    // console.log(JSON.stringify(urlParameter.query)) ;
    // console.log(JSON.stringify(urlParameter.pathname)) ;
    const {query, pathname} = url.parse(req.url, true);
    if (query.id){
        // Courses page
        if (pathname === '/' || pathname.toLowerCase() === '/courses')
{
            res.writeHead(200, { // Every thing ran successfully
                'Content-type': 'text/html'
            });
        const course = dataObj[Number(query.id)];
        const strCourseName = JSON.stringify(course);
        const courseHTML = replaceTemptate(templateHTMLCourse, course);
        // res.end(`We received our first request from the client at
resource ${urlParameter.pathname.toLowerCase()} with query parameter
${urlParameter.query.id}
        // ${JSON.stringify(course)}` )
        // }
        // console.log(courseHTML)
        res.end(courseHTML);
        }
        else{
```

```
            res.writeHead(404,  { // Server did not find what you were
looking for
                'Content-type': 'text/html'
            });
            res.end('resource not found')
        }
    }
});
// Start Listening to requests
server. listen(8000, 'localhost' , function() {
    console.log('Listening to requests on port 8000');
});
```