IH31・IW31　Node.js2-1 課題確認

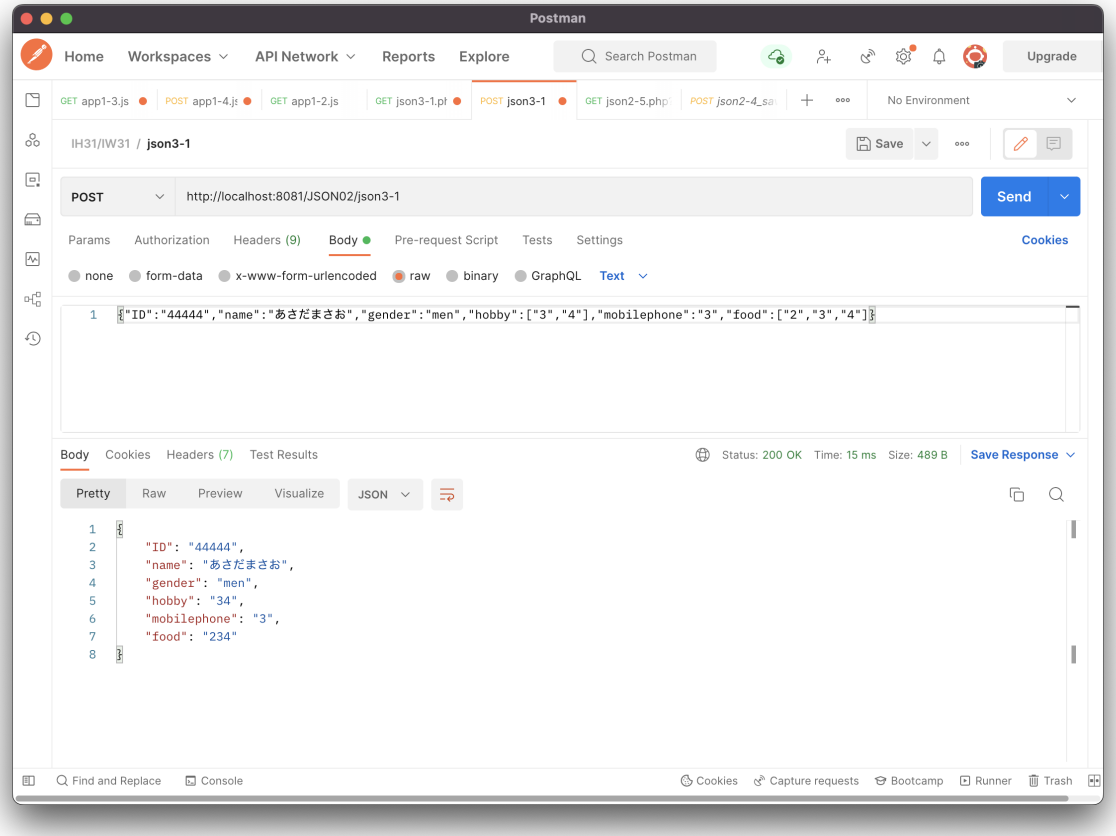
■課題提示

IH31/WA31-05 json3-1+Node.js
提出日：７月**5**日（火）　※グループ単位で直接チェック。７月１日までに完成するように。
提出形態：HAL課題表紙
内容：
＜JSON(json3-1.php）＞
・POSTMANを使用し、json3-1の実行確認。
・POST…アンケートの登録、GET…アンケートCSVの受信。
・CSVファイルは３件にしておく。
＜Node(app2-3)＞
・簡易Webサーバ５（app2-3）を実現する。

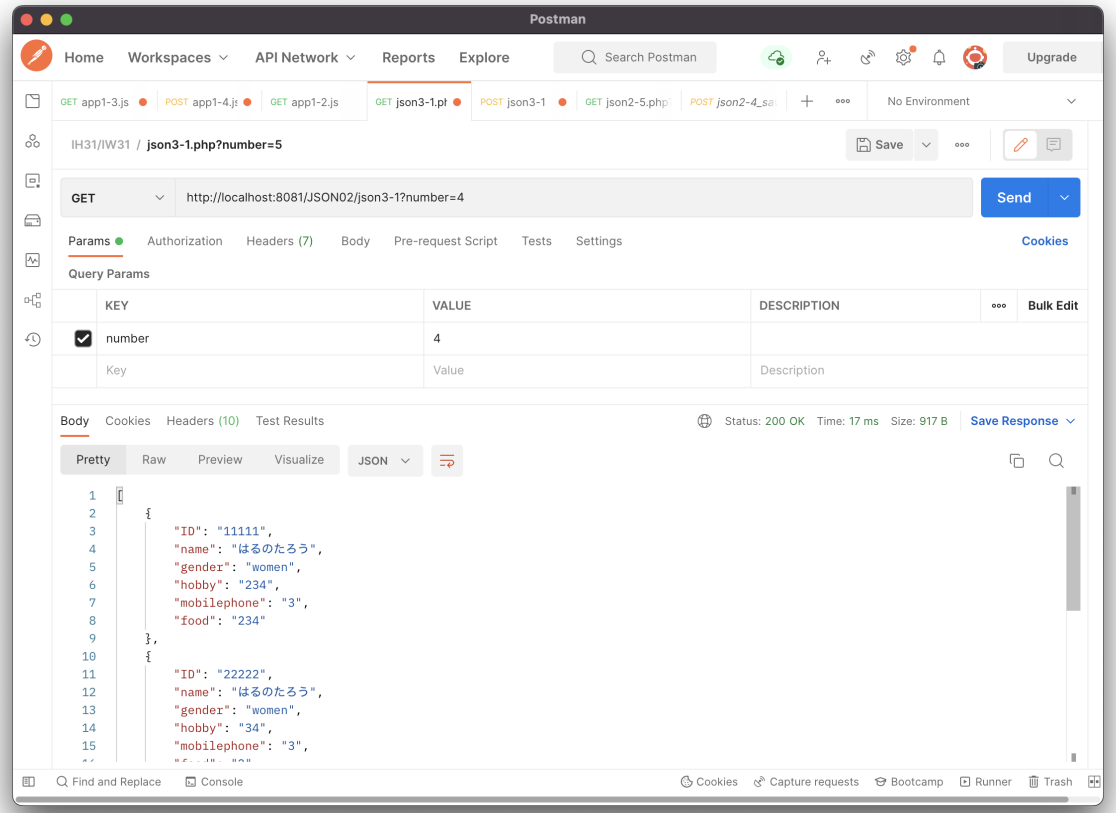
■JSON(json3-1.php)　アンケート

- スクリーンショット

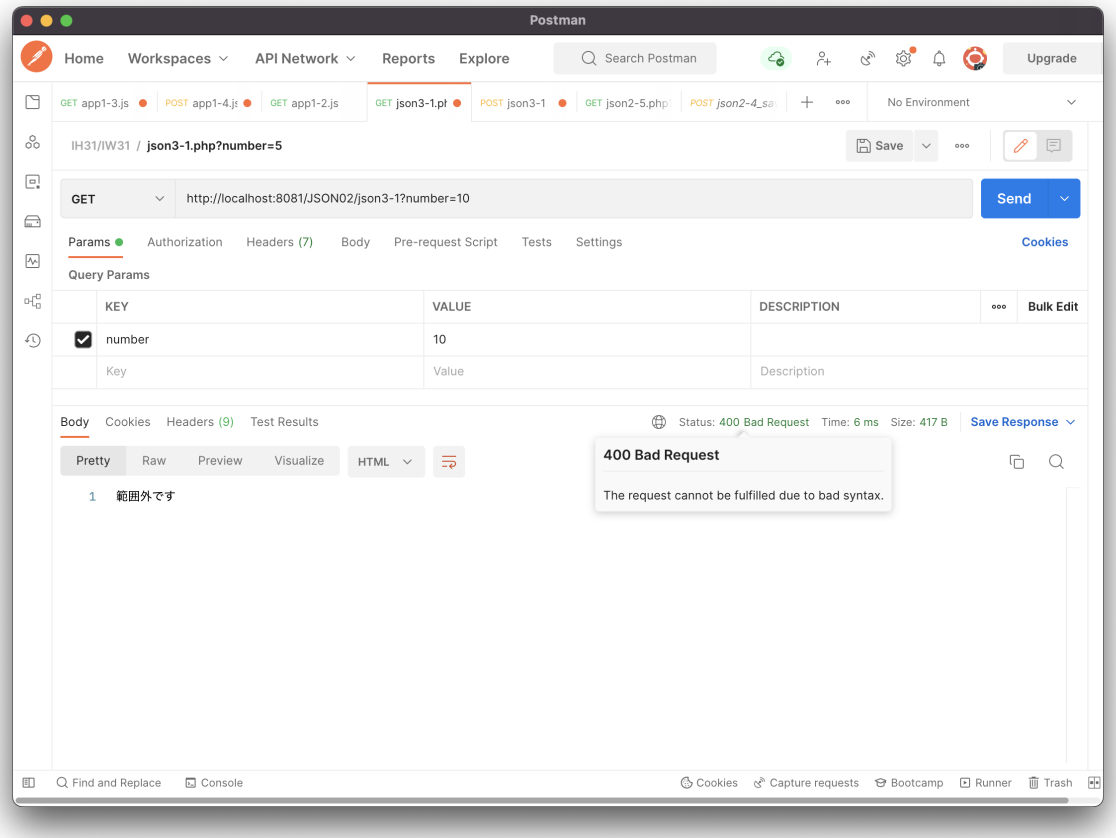
1. POST（登録）



2. GET（取得）



3. GET（取得：範囲外...0～9以外）



- 拡張子（.php）を隠す htaccessを記述する。

- apache2.４以降
  AddType application/x-httpd-php .php
  MultiviewsMatch Handlers
  Options +MultiViews

- 考え方

1. GETかPOSTの判断
    1. POSTだったら
        1. 送信されたデータをCSVで保存。（json2-4）
    2. GETだったら
        1. リクエスト(number)に値がなければ1をセット
        2. リクエスト(number)が範囲外（1〜9以外）だったら"範囲外です"を返す
        3. 指定された件数の連想配列を作成
        4. JSON文字列に変換し、クライアントに渡す

- PHP:サーバ側でMethodを判別する(GET or POST)

```
if($_SERVER["REQUEST_METHOD"] != "POST"){
    //POST以外
}else{
    //POST
}
```

- PHP:リクエストに値がない場合

```
$num = $_GET['number'];
if(is_null($_GET['number'])){
    $num=1;
}
```

- PHP:リクエストが範囲外の場合

```
if($num<1 or $num>=10){
    http_response_code(400);
    echo "範囲外です";
    exit;
}
```

- PHP:CSVを一行ずつ取得し配列化する

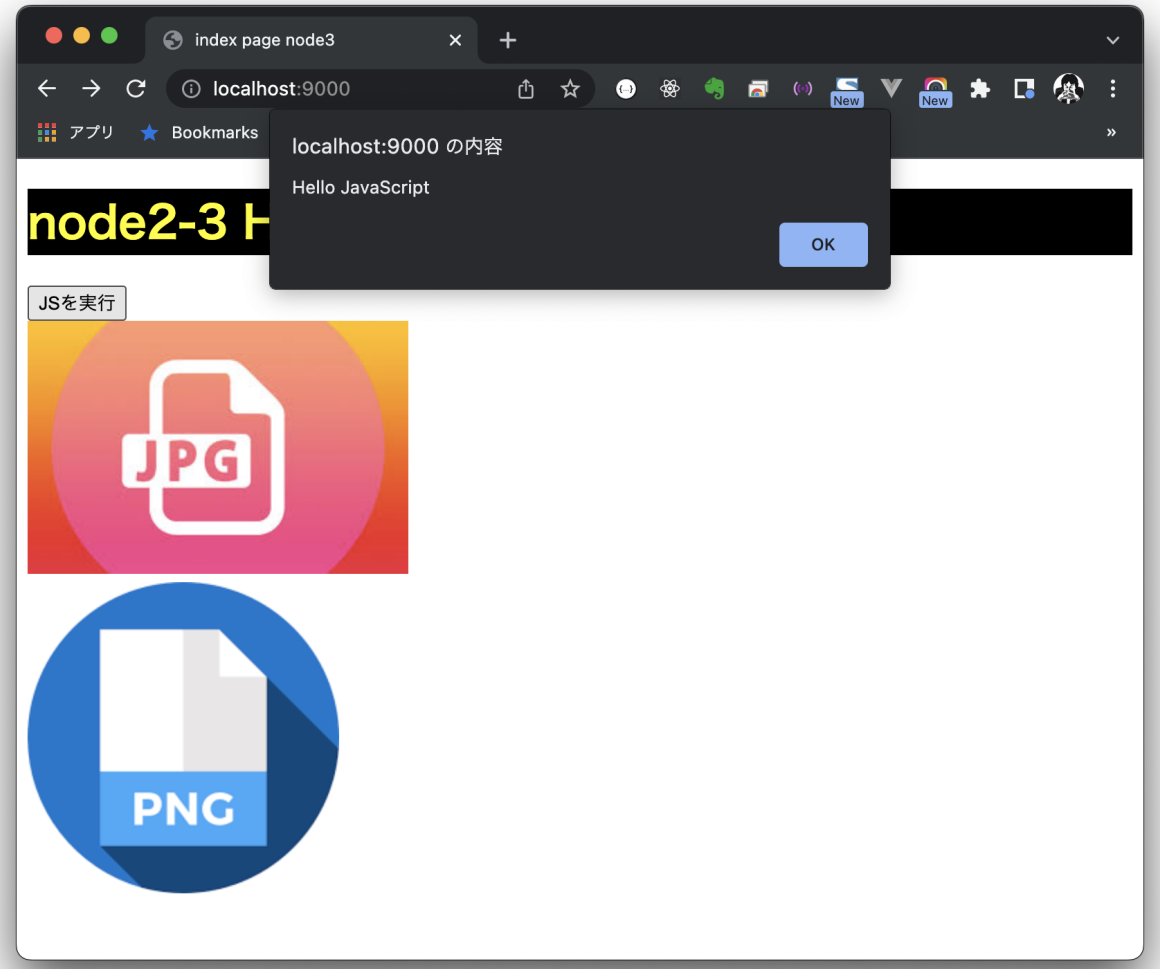
```
$filepath = 'enquete.txt';
$i = 0;
if (($handle = fopen($filepath, "r")) !== false) {
    while (($line = fgetcsv($handle, 0, ",")) !== false) {
```

```
        if($i < $num){
            foreach ($line as $key => $data){
                $data_substance = explode("=", $data);
                $array[$data_substance[0]] = $data_substance[1];
            }
            $records[] = $array;
            $i++;
        }else{
            break;
        }
    }
    fclose($handle);
}
http_response_code(200) ;
echo json_encode($records);
```

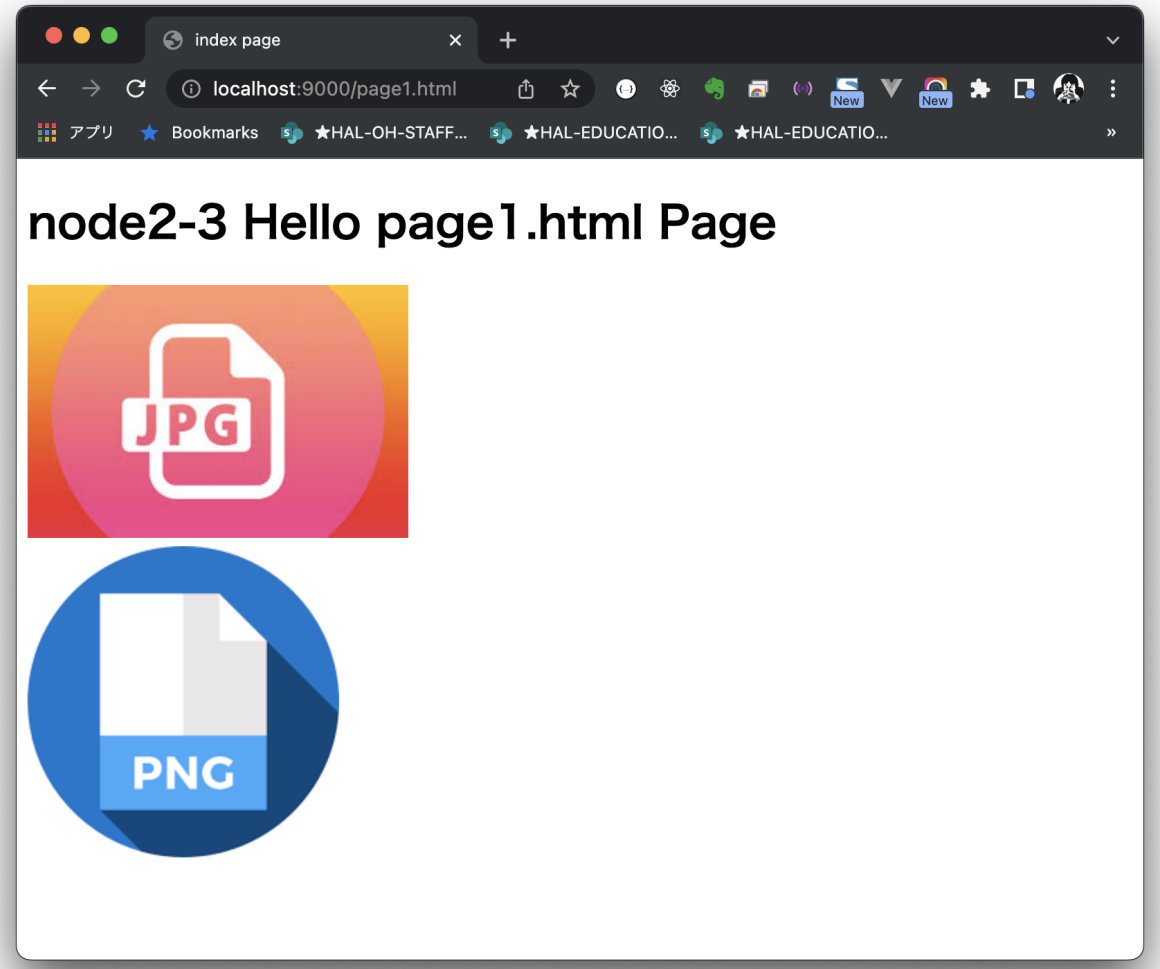
■Node(app2-3) 簡易Webサーバ５

- スクリーンショット

1. /(ルート表示)


index page node3
localhost:9000
アプリ
Bookmarks
localhost:9000 の内容
Hello JavaScript
OK
node2-3 H
JSを実行
JPG
PNG

2. page1.html



- 外部ファイルの参照
  - 参照先（呼び出される方）

```
exports.routeResponseMap={
    "/":"./views/index.html"
}
```

  - 参照元（呼び出す方）

```
var config=require('./config');
config.routeResponseMap[url]
```

- フォルダ構成

```
・・・app2-3
   ├ app2-3.js
```

```
    ├ config.js
    ├ contenttype.js
    ├ views
    │   ├ index.html
    │   └ page1.html
    └ public
        ├ css – style.css
        ├ js  – javascript.js
        └ img
            ├ image1.jpg
            └ image2.png
```

- config.js

```
exports.routeResponseMap={
    "/":"./views/index.html"
}
exports.port=9000;
```

- contenttype.js

```
exports.fileContentTypePathMap={
    ".html": {"Content-type": "text/html", "path": "./views"},
    ".css": {"Content-type": "text/css", "path": "./public"},
    ".js": {"Content-type": "text/javascript", "path": "./public"},
    ".jpg": {"Content-type": "image/jpeg", "path": "./public"},
    ".jpeg": {"Content-type": "image/jpeg", "path": "./public"},
    ".png": {"Content-type": "image/png", "path": "./public"}
}
```

- URLを取得して/かそれ以外かを判別

```
var url=request.url;
console.log(url)
if(url == "/"){
    response.writeHead(200,{"Content-type":"text/html"});
    fs.readFile(config.routeResponseMap[url],function(error,data){
        response.end(data);
    })
}else{
```

- else以降の流れ
  ①拡張子を調べる
  ②拡張子がコンテントタイプになければNotFoundを表示
  ③ファイルをブラウザへ渡す

```
var filetype = "";
for(var key in contenttype.fileContentTypePathMap){
    if(url.indexOf(key) != -1){
        filetype=key
    }
}
if(filetype == ""){
    sendErrorResponse(response);
}else{
    response.writeHead(200,{"Content-
type":contenttype.fileContentTypePathMap[filetype]["Content-type"]});
    customReadFile(contenttype.fileContentTypePathMap[filetype]
["path"]+url,response);
}
```

■Node(app2-2) 簡易Webサーバ4

```
'use strict'
const routeResponseMap={
    "/":"views/index.html"
}

const port=9000;
const http=require('http');
var fs=require('fs');
var app=http.createServer();
app.on("request",function(request,response){
    var url=request.url;
    console.log(url)
    if(url == "/"){
        response.writeHead(200,{"Content-type":"text/html"});
        fs.readFile(routeResponseMap[request.url],function(error,data){
            response.end(data);
        })
    }else if(url.indexOf(".html") != -1){
        response.writeHead(200,{"Content-type":"text/html"});
        customReadFile("views"+url,response);
    }else if(url.indexOf(".css") != -1){
        response.writeHead(200,{"Content-type":"text/css"});
        customReadFile("public"+url,response);
    }else if(url.indexOf(".js") != -1){
        response.writeHead(200,{"Content-type":"text/javascript"});
        customReadFile("public"+url,response);
    }else if(url.indexOf(".jpg") != -1){
        response.writeHead(200,{"Content-type":"image/jpeg"});
        customReadFile("public"+url,response);
    }else if(url.indexOf(".png") != -1){
        response.writeHead(200,{"Content-type":"image/png"});
        customReadFile("public"+url,response);
```

```
    }else{
        sendErrorResponse(response);
    }
});
app.listen(port);
console.log("サーバ起動:"+port+"ポート監視中");
var customReadFile = function(file_path,response){
    if(fs.existsSync(file_path)){ //ファイルが存在するか
        fs.readFile(file_path,function(error,data){
            if(error){
                sendErrorResponse(response);
                return;
            }else{
                response.write(data);
                response.end();
            }
        });
    }else{
        sendErrorResponse(response);
    }
}
var sendErrorResponse = function(response){
    response.writeHead(400,{"Content-type":"text/html"});
    response.write("<h1>Not Found</h1>");
    response.end();
}
```