

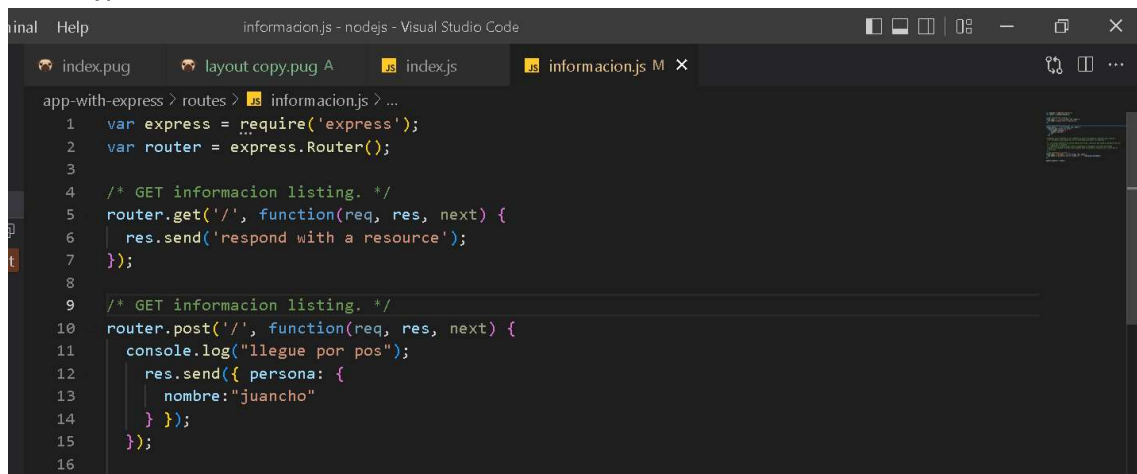
Agregando un módulo como servidor

para agregar un modulo en la aplicación que responda a una petición, debemos crear el archivo js correspondiente y en el importar el modulo de express como ya ase ha venido trabajando y crea una variable que represente el enroutamiento, es decir para que capture las url bajo las cuales va a escuchar la petición:

- **var express = require('express');**
- **var route = express.route();**

después de eso ya podemos empezar a crear los métodos de los servicios para las peticiones que vayamos a escuchar, donde simplemente llamaremos a nuestra variable route creada y mediante el método correspondiente al verbo a la petición del protocolo http, podremos escuchar la petición, pasándole como primer parámetro la url que escuchara y el segundo parametro seria una función anónima que nos proporcionará la funcionalidad o logica que nosotros realizaremos correspondiente a esa petición, esta funcion llevara como paramtetro el reques y el response que nosotros utilizaremos para responder a esa petición.

- **route.get('url bajo la que se escuchara', funtion(request, response){
response.send("hello word");
})**



```
1 var express = require('express');
2 var router = express.Router();
3
4 /* GET information listing. */
5 router.get('/', function(req, res, next) {
6   res.send('respond with a resource');
7 });
8
9 /* GET information listing. */
10 router.post('/', function(req, res, next) {
11   console.log("llegue por pos");
12   res.send({ persona: {
13     nombre: "juancho"
14   } });
15 });
16
```

Representando en las vistas pug, los parámetros e información que llegan a servidor o controller como o queramos mmar, dependiendo del verbo que utilizemos.

para recuperar los parametro que venga por el vervo get en la url, podemos recuperarlos desde el vergo correspondiente con la captura de la url de la siguiente manera:

- **router.get('/informacion/primerParametro/:segundoParametro', function(){})**

Ahora para enviarlo esos parámetros o información que queramos enviar a la visgta pug vinculada, lo podemos hacer mediante el método render que esta en el response para esa petición, en este pasamos como parámetro en primer lugar el nombre de nuestra vista que queremos que se renderice y en segundo parametro es un json, con el objeto que queremos pasar o simplemente con el atributo único de ser de esta manera, si queremos recuperar los parametros que vienen de la url de ese reques simplemnte utilizamos el objeto **request.params."nombreParamtro"**:

- `resp.render("nombreDeLaVistaAsociada", {
 persona: { nombre: req.params."nombreParametroURL",
 apellido: req.params."nombreParametroURL" })`

```
/* GET informacion listing. */
router.get('/:nombre/:edad', function(req, res, next) {
  res.render('informacion', {
    usuario: {
      nombre: req.params.nombre,
      edad: req.params.edad
    }
  }); /*con el metodo render, mencionamos que vamos
a responder renderizando una vista, en este caso la informacion pug
```

Representamos los datos pasados desde el servidor o controller a la vista:

Desde la semántica que manejamos en la vista pug, simplemente recuperamos los datos con el nombre del objeto que viaja dentro del json, sea objeto completo o un simple atributo y ya de esta manera lo podemos renderizar en el código html que maneja pug, recordando que este no utiliza etiquetas de cierre ni tampoco los <> simplemente el nombre de la etiqueta.

Ahora para la utilización de condicionales en esta vista basta simplemente con escribir if-else sin necesidad de los paréntesis para evaluar la condición, simplemente la colocamos al frente del if, como también no es necesario las {} para el código a ejecutarse según la condicional, solamente con manejar el indentado ya está.

Ahora si queremos mezclar el texto que le damos a una etiqueta con el contenido que viene desde el servidor utilizamos una especie de interpolación parecida a la ts, que nos provee estas plantillas de pug, donde simplemente utilizamos **`#{variable, atributo, etc}`**:

```
app-with-express > views > 🐼 informacion.pug
1  if usuario
2    h1 nombre: #{usuario.nombre}
3    h2 apellido: #{usuario.apelli
4    h3 edad: #{usuario.edad}
5  else
6    h2 sin datos
```