1. Template engine – ejs
2. Recursos estáticos
3. Diferencia y el porque usar EJS en ves de HTML
4. Compartir info desde los controladores hasta las vistas
5. Automatizar las vistas
6. Notas de la clase

1.

Template engine (motor de plantilla) hace html dinámicos. SERA EL ENCARGADO DE MANEJAR LAS VISTAS.

Por ejemplo pug ejs handlebars mustache, etc

**Archvo html super poderoso**, Toma info de los controladores y además muestra las misma.

**EJS**: es comodo, misma sintaxis de html. Extensión **.ejs**

**Instalación:**

Como cualquier cosa que queramos instalar en proyecto de express usamos npm: npm install ejs

App.js: app.set(), dos parámetros: (“view engine”, “(motor de plantilla que usemos ej: ejs)”).

La parte de “view engine”, sirve para que no tener que indicarle la extinción, cuando queramos renderizarlo en el controller

Todos documentos .ejs dentro de carpeta views. Si qeuremos que la carpeta s ellame distinsta hay que avisar a expres: app.set(“views”, \_\_dirname + “/ejs-html”); ejs-html es el nombre de la carpeta ejemplo.

Primera vista:

Las **vistas** se comunican con los **controladores**y toman la información que estos reciben de los **modelos**. Para poder**mostrar**una **vista**en el navegador, es importante aclararle al **controlador** cuál **vista**es la que deseamos **enviar.**. Al proceso de enviar la vista al navegador lo conocemos como **renderización de vista**.

Renderizar las vistas:

Enviar primer vista como respuesta (res) al servidor.

Const controler ={

mostrartProductos : (req, res) => {

res.**render**(“producto”);

};

1. No es necesario definir la exptencion del archivo.
2. No es necesario nobrar la carpeta de views,

Ejs es igual que html (mismas etiquedas, formato, etc)

2.

Recursos estáticos: dentro de app.js

app.use(express.static(path.join(\_\_dirname, “../public)));

Requerir path\*

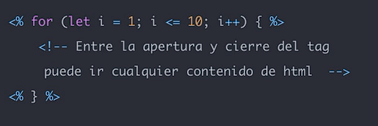
Los archivos que esten dentro de public que linkeemos en los html o ejs, sus rutas son /css/style.css por ejemplo. La primer / hace referencia a la carpeta public.

<img src=”/images/logo.png”> ejemplo.

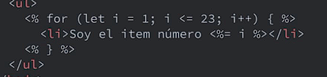
3.

Como trabajar con EJS: motro de renderizacion con etiquetas nuevas (documento debe ser .ejs (resiste la misma estructua de HTML)).

<% %>: inplemeta cualquier control de flijo: if , for, etc.



<%= %>: indica que todo lo que este dentro deberá ser impreso literalmente dentro del contenido html, por ejemeplo la variable i del for



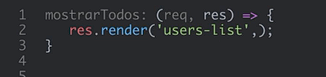
Esto imprimiría 23 veces el string “ Soy un ítem numero (y la variable i)”

4.

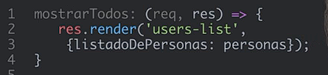
Compartir info desde los controladores hasta las vistas



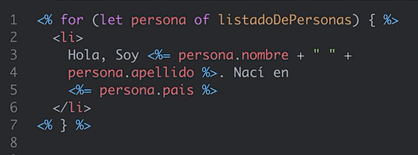
Tenemos esta array en un controlador.



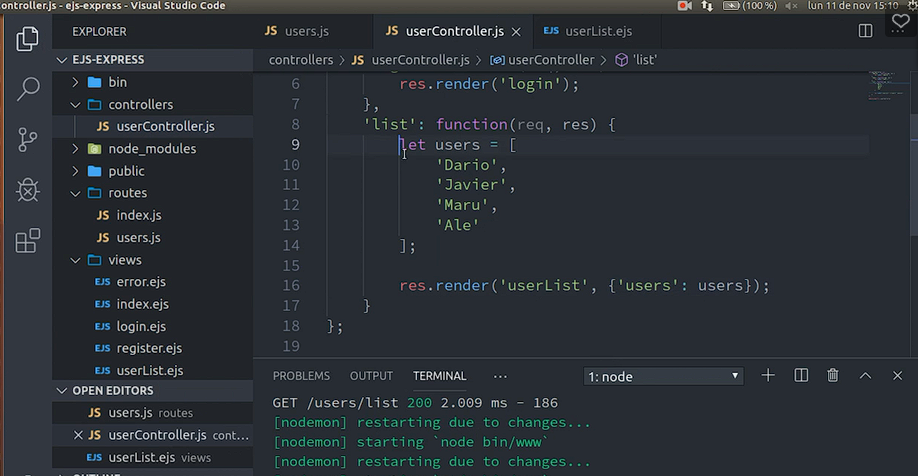
Como segundo parámetro le pasamos el objeto literal :

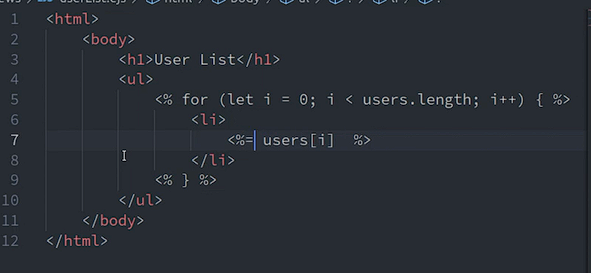


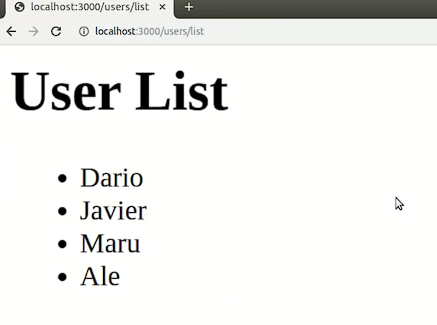
En el EJS podría usarse asi:



Ejemplo mas claro:





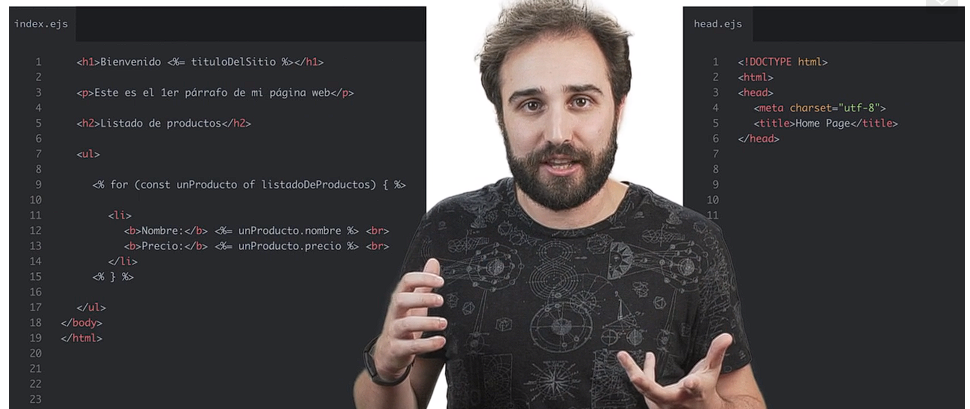


5.Automatizar las vistas

Repetir header, footer, etc. haciendo plantillas

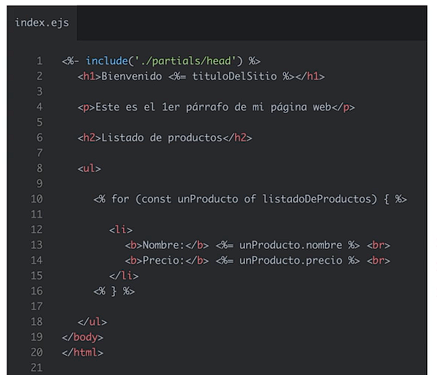
\*NO HACE FALTA SEPARAR LAS ETIQUETAS DE APERTURA Y DE CIERRRE DEL HTML EN LOS DOS ARCHIVOS COMO MUESTRA EL EJEMPLO, SOLO PONES EL HEADER EN EL ARCHIVO HEADER.EJS\*

Header en otro archivoi .EJS dentro de una carptea llamada **partials** (nombre opcional):



Como los junto?

**Nueva etiqueta de EJS:** <%- uncludes() %>, como parámetro le pasamos la ruta del archivo parcial que queremos incluir:



6.**notas de la clase:**

Diferencia entre "dependencies" y "devDependencies"

"dependencies": {

    "ejs": "^3.1.8",

    "express": "^4.18.2"

  },

  "devDependencies": {

    "nodemon": "^2.0.20"

  }

"dependencies":: dependencias que el proyecto necesita para funcionar

"devDependencies":: no necesita para funcionar. Por ejemplo en render son ignoradas porque no hacen falta y ocupan lugar al dope.

Dentro de un main-router.js:

const { Router } = require('express');

const router = Router();

// abajo y arriba hacen lo mismo en este caso

//const express = require('express');

//const router = express.Router();

\*abajo es como le gusta a playground.

app.set('view engine', 'ejs');

app.set('views', path.join(\_\_dirname, './views'));

la primer línea le dice que vamos a usar el templet engine EJS, la segunda línea le dice donde están esas vistas.

si no se pone esas líneas en el app.js salta este error:

