

Módulos

Contenido

Accesos a Zoom y Grabaciones

Proyecto integrador: sprints

Info general

Ocultar <<

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC

Si bien esta pregunta no tiene una respuesta 100% definitiva, si podemos pensar en una manera estándar que permita organizarnos de la mejor manera posible. Es por este motivo que, a continuación, compartimos una estructura de archivos sugerida para optimizar mucho más el proceso de programar.

Estructura de archivos recomendada:

```
node_modules ← Carpeta de módulos locales de proyecto
public ← Carpeta de archivos públicos
src ← Carpeta de archivos funcionales de la aplicación
    ├── controllers
    │   └── (acá van los controladores)
    ├── views
    │   └── (acá van las vistas)
    ├── routes
    │   └── (acá van las rutas)
    └── app.js ← Archivo de entrada del proyecto
    └── package.json
    └── package-lock.json
```

Con dicha estructura, vamos a poder separar por responsabilidades cada uno de los componentes de nuestra aplicación y, de esta manera, gestionar mejor nuestro trabajo y el del equipo. En caso de tener que repartir las tareas, estas ya se podrán dividir por



Módulos Contenido Accesos a Zoom y Grabaciones Proyecto integrador: sprints Info general

Ocultar <

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC

MVC: componentes y flujo

Repasemos los componentes pasando el mouse sobre los mismos.

Vista: Conforma la **interfaz gráfica** de la aplicación y contiene todos los elementos que son visibles al usuario. A través de ella, el usuario interactúa enviando y solicitando información al servidor. Su responsabilidad es **definir la apariencia** de los datos y **mostrarlos** en pantalla.

Ver flujo >

Modelo: Representa los datos y las lógicas del sistema. Los datos se almacenan en bases de datos y se procesan por servicios de negocio.

Módulos **Contenido** Accesos a Zoom y Grabaciones Proyecto integrador: sprints Info general

Ocultar <

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC**
- Controladores
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC

Introducción a MVC

Intentos: ilimitados

¡Excelente!
Respuestas correctas: 100%

Mira las respuestas de cada pregunta:

2 **MVC significa...**

- Model-Value-Control
- Model-View-Controller
- Most-Value-Coder
- Massive-View-Controller

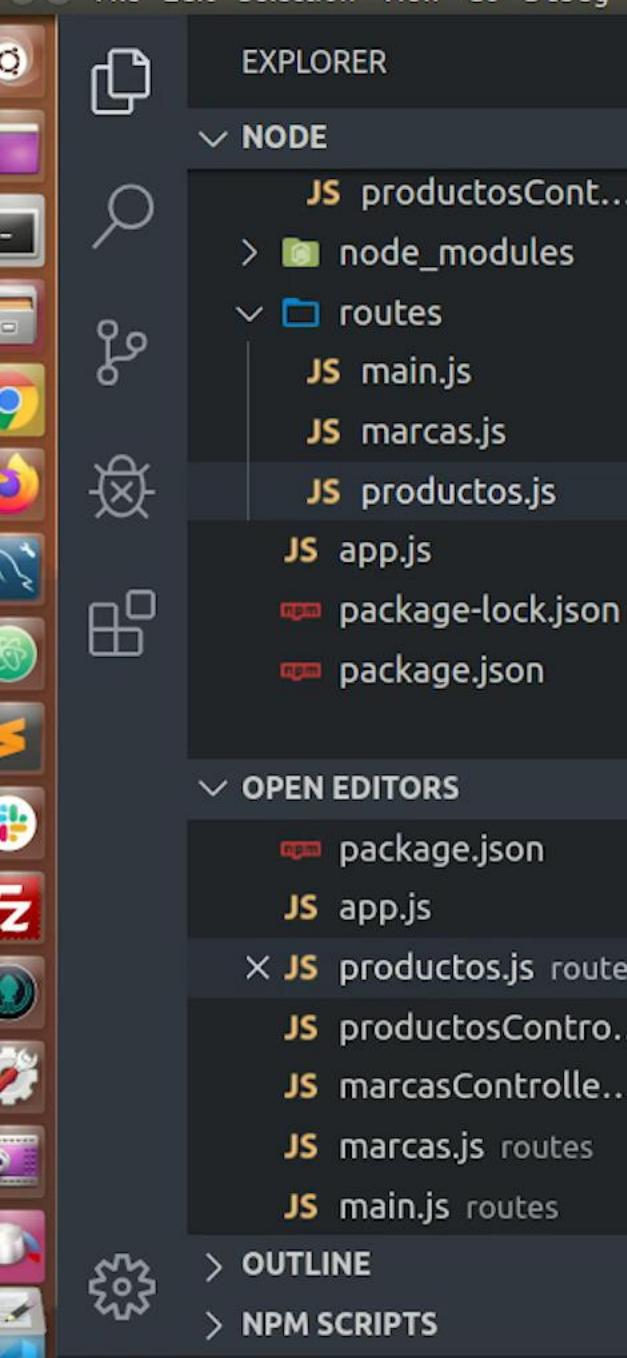
¡Bien hecho! El MVC representa el Modelo-Vista-Controlador.

The screenshot shows the Visual Studio Code interface with the following details:

- Explorer View:** Shows the project structure under the **NODE** category:
 - controllers:** Contains **marcasController.js** and **productosController.js**.
 - node_modules:** A folder.
 - routes:** Contains **main.js**, **marcas.js**, **productos.js**, **app.js**, and **package-lock.json**.
- Open Editors:** Shows multiple files:
 - package.json**
 - app.js**
 - productos.js routes**
 - productosController.js** (highlighted in red)
 - marcasController.js**
 - marcas.js routes**
 - main.js routes**
- Editor View:** The **productosController.js** file is open, displaying the following code:

```
let productosController = {
    listado: function() {},
    crear: function() {},
    detalle: function(req,res) {
        res.send("Bienvenidos al detalle del producto " + req.p
    },
    detalleComentarios: function(req,res) {
        if (req.params.idComentario == undefined) {
            res.send("Bienvenidos a los comentarios del producto")
        } else {
            res.send("Bienvenidos a los comentarios del producto")
        }
    }
};

module.exports = productosController;
```
- Status Bar:** Shows the current file is **productosController.js**, line 13, column 6, with 4 spaces, using **UTF-8** encoding, and the language is **JavaScript**. It also shows a **Blockframe** status.



routes > **JS productos.js** > ...

```
1 let express = require('express');
2 let productosController = require('../controllers/
3   productosController.js');
4 let router = express.Router();
5
6 router.get('/:idProducto', productosController.detalle);
7
8 router.get('/:idProducto/comentarios/:idComentario?',
9   productosController.detalleComentarios);
10 module.exports = router;
```



EXPLORER

NODE

JS productosCont...

> node_modules

routes

JS main.js

JS marcas.js

JS productos.js

JS app.js

npm package-lock.json

npm package.json

OPEN EDITORS

npm package.json

X JS app.js

JS productos.js routes

JS productosContro...

JS marcasControlle...

JS marcas.js routes

JS main.js routes

> OUTLINE

> NPM SCRIPTS

JS app.js X

JS productos.js

JS productosController.js

JS marcasController.js



JS app.js > [o] rutasProductos

```
1 let express = require('express');
2 let rutasProductos = require('./routes/productos.js');
3 let rutasMain = require('./routes/main.js');
4
5 let app = express();
6
7 app.listen(3000, () => console.log('Esto fue exitoso'));
8
9 app.use('/productos', rutasProductos);
10 app.use('/', rutasMain);
```

App.js van recursos y prefijos

En archivos de rutas: solamente el estado y diccionario de la ruta

Controller: la funcionalidad y las consecuencias que se tendrán

playground.digitalhouse.com/course/2815/unit/01842f3b-f8a7-11eb-876f-12eb947d8fb1/lesson/27d57df7-f8a9-11eb-876f-12eb947d8fb1/topic/d21f9c96-f8a7-11eb-876f-12eb94...

Módulos Contenido Accesos a Zoom y Grabaciones Proyecto integrador: sprints Info general

Ocultar <>

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC ^

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repasso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC

El código final

En el archivo del controlador, escribimos los métodos de cada pedido y exportamos el objeto literal resultante.

```
const controlador = {  
  index: (req, res) => {  
    res.send('Index de productos');  
  },  
};  
module.exports = controlador;
```

En el archivo de rutas, llamamos a nuestro controlador y en cada ruta ejecutamos el método que corresponda.

```
{} const productosController = require('../controllers/productosController');  
router.get('/', productosController.index);
```

Los controladores DigitalHouse> Google Slides

12 :: Descargar

playground.digitalhouse.com/course/2815/unit/01842f3b-f8a7-11eb-876f-12eb947d8fb1/lesson/27d57df7-f8a9-11eb-876f-12eb947d8fb1/topic/d21f9c96-f8a7-11eb-876f-12eb94...

Módulos Contenido Accesos a Zoom y Grabaciones Proyecto integrador: sprints Info general

Ocultar <<

Módulo: 5 Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC

Mira las respuestas de cada pregunta:

1 ¿Qué es un controlador?

- El encargado de la visualización de datos.
- El encargado del modelado de datos.
- El que mantiene la lógica de negocio.

✓ El controlador es justamente el que mantiene la lógica de negocio, no sabe cómo se muestran las cosas en pantalla y tampoco tiene la definición del modelado de datos.

Siguiente →

1 2 ✓ ✓

playground.digitalhouse.com/course/2815/unit/01842f3b-f8a7-11eb-876f-12eb947d8fb1/lesson/27d57df7-f8a9-11eb-876f-12eb947d8fb1/topic/d21f9c96-f8a7-11eb-876f-12eb94...

Módulos Contenido Accesos a Zoom y Grabaciones Proyecto integrador: sprints Info general

Ocultar <>

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC ^

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores**
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC

2 ¿Cuál de las siguientes frases es correcta? El controlador...

mantiene diálogo con la vista.

no mantiene diálogo con la vista.

mantiene diálogo con el modelo.

no mantiene diálogo con el modelo.

✓ Muy bien. Podemos interpretar al controlador como un intermediario, un interlocutor entre la vista y el modelo.

Anterior

1 2 ✓ ✓

Módulos Contenido Accesos a Zoom y Grabaciones Proyecto integrador: sprints Info general

Ocultar <>

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC ^

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC  ▾

1 Nos llega un pedido para diseñar un sistema que permita manejar el stock de un kiosco. Decidimos aplicar nuestros conocimientos en Node.js y encarar el proyecto utilizando esta tecnología. Yendo más al detalle, nos informan que tenemos que llevar un registro de los clientes como si fueran usuarios. Debemos registrar datos como el nombre, DNI, productos que compró. También debemos llevar los detalles de los productos: precios, stock y registrar las ventas del mismo.

A la hora de armar el sistema, comenzamos a preguntarnos qué controladores creemos que podríamos llegar a necesitar.

¡Comencemos!

¿Cuál de los siguientes controladores podrían estar en el sistema?

preciosController

productosController

galletitasController

✓ Excelente. El controlador productosController es el que necesitamos para poder manejar todos los distintos productos. También puede manejar un montón de detalles de los mismos, como el precio, descripción, etcétera.

playground.digitalhouse.com/course/2815/unit/01842f3b-f8a7-11eb-876f-12eb947d8fb1/lesson/27d57df7-f8a9-11eb-876f-12eb947d8fb1/topic/d21f9c96-f8a7-11eb-876f-12eb94...

Módulos Contenido Accesos a Zoom y Grabaciones Proyecto integrador: sprints Info general

Ocultar <>

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores**
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC

2 Mira las respuestas de cada pregunta.

¿Cuál de los siguientes controladores podrían estar en el sistema?

- pantallaController
- rutasController
- stockController

Muy bien. Está bien pensar que necesitaremos un controller para lo que es el stock de los productos. Si bien podríamos tener esta responsabilidad en el productosController, al ser algo muy importante para el negocio, puede ser una buena decisión manejar toda esa lógica en forma separada. Por otro lado, pantallaController no es en sí necesario, ya que a priori parece que tiene mayor relación con la visualización de cosas y eso es parte de la vista. Por último, llevar el control de las rutas no es en sí una tarea de un controlador específico.

Anterior Siguiente

1 2 3

playground.digitalhouse.com/course/2815/unit/01842f3b-f8a7-11eb-876f-12eb947d8fb1/lesson/27d57df7-f8a9-11eb-876f-12eb947d8fb1/topic/d21f9c96-f8a7-11eb-876f-12eb94...

Módulos Contenido Accesos a Zoom y Grabaciones Proyecto integrador: sprints Info general

Ocultar <>

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC

3 ¿Cuál de los siguientes controladores podrían estar en el sistema?

- afipController
- clientesController
- morososController

Perfecto. En este caso podemos inferir que moroso es un caso particular de cliente, por lo cual no sería necesario crear una lógica completamente diferente. Luego, el enunciado no habla de tener que dialogar con la AFIP, así que a priori no haría falta.

Anterior

1 2 3

✓ ✓ ✓

playground.digitalhouse.com/course/2815/unit/01842f3b-f8a7-11eb-876f-12eb947d8fb1/lesson/27d57df7-f8a9-11eb-876f-12eb947d8fb1/topic/d21f9c96-f8a7-11eb-876f-12eb94...

¡Bien hecho! Felicitaciones.

Ocultar <>

PRIMER controller

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Comenzamos un nuevo día y llega un nuevo requerimiento. Estamos trabajando en el controlador de un carrito de e-commerce y nos piden que, al agregar un ítem, el controlador debe devolver el mensaje "Ítem Agregado".

El líder técnico del área nos indica que para cumplir con este requerimiento debemos modificar la función de agregarItem para que reciba los parámetros `req` y `res`. Dicho esto, el líder técnico se aleja silenciosamente confiando plenamente en nuestras habilidades.

Dame una pista ^

Recordar que para enviar una respuesta debemos usar el parámetro `res`.

PlaygroundCode <> script.js x

SRC PATH: SRC > SCRIPT.JS

script.js

```
let carritoController = {
  sacarItem: function(){}
  consultarItem: function(){}
  agregarItem: function(req,res){res.send('Ítem Agregado')}
};

module.exports = carritoController
```

playground.digitalhouse.com/course/2815/unit/01842f3b-f8a7-11eb-876f-12eb947d8fb1/lesson/27d57df7-f8a9-11eb-876f-12eb947d8fb1/topic/d21f9c96-f8a7-11eb-876f-12eb94...

¡Excelente! Gran trabajo.

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores**
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC

Habiendo finalizado un día exitoso y estando apunto de levantarnos de la silla, viene nuestro Project Manager y nos implora que antes de irnos agreguemos una funcionalidad más: "Por favor, necesito que se pueda agregar un producto al carrito, porfis, porfis". Resulta que si bien se trabajó en el `carritoController`, este no fue redireccionado por una ruta. Por lo cual, desde el navegador, no se puede acceder a la lógica de negocio del carrito. Al analizar el `carritoController`, el Project Manager nos comenta que la ruta del carrito tiene un parámetro dinámico que debemos llamar `item`, el cual representa el item a agregar (antes de ser Project Manager era programador también).

Debemos analizar el siguiente código y agregar tanto la funcionalidad `agregarItem` del `carritoController` como crear la ruta dinámica `item`.

Dame una pista ^

Recordar que, para poder utilizar rutas dinámicas, estas deben utilizar el ":" y luego el nombre con el que se va a referir el parámetro. Ejemplo: `/:productId` Nos dice que la ruta va a ir variando y ese valor será almacenado como `productId`

PlaygroundCode

script.js

SRC

PATH: SRC > SCRIPT.JS

```
1 let express = require('express')
2 let router = express.Router();
3 let carritoController = require
4 ('./controllers/carritoController')
5 router.get('/:item', carritoController.agregarItem)
6 module.exports = router
```

determinado?

Ocultar <

Módulo: 5

Express

Clase 19 - Patrones de diseño MV

- Introducción al módulo
 - ¿Qué vamos a ver en esta clase
 - Introducción a MVC
 - Controladores
 - Repaso de rutas
 - Rutas parametrizadas
 - Sistema de ruteo
 - Estructura de archivos
 - #BonusTrack express-generator
 - Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrón de diseño MVC

The screenshot shows the Visual Studio Code (VS Code) interface. The top menu bar includes File, Edit, Selection, View, Go, Debug, Terminal, and Help. The status bar at the bottom displays the date and time (vie 25 de oct 16:40), battery level (89%), signal strength, and file information (Line 16, Col 26, Spaces: 4, UTF-8, JavaScript).

The Explorer sidebar on the left lists the project structure:

- EXPLORER
- NODE
 - node_modules
 - JS app.js
 - package-lock.json
 - package.json
- OPEN EDITORS
 - package.json
 - JS app.js
- OUTLINE

The main editor area displays the `app.js` file content:

```
package.json JS app.js ×

JS app.js > app.get('/productos/:idProducto') callback
1 let express = require('express');
2
3 let app = express();
4
5 app.listen(3000, () => console.log('Esto fue exitoso'));
6
7 app.get('/', function(req, res) {
8     res.send('Bienvenidos al sitio!!!!!!');
9 });
10
11 app.get('/contacto', function(req, res) {
12     res.send('Dejanos tu contacto!');
13 });
14
15 app.get('/productos/:idProducto', function(req,res) {
16     res.send("Bienvenidos al detalle del producto " +
17         req.params.idProducto);
18 });

03:31 05:42
```

playground.digitalhouse.com/course/2815/unit/01842f3b-f8a7-11eb-876f-12eb947d8fb1/lesson/27d57df7-f8a9-11eb-876f-12eb947d8fb1/topic/d21f9fa1-f8a7-11eb-876f-12...

Módulos Contenido Accesos a Zoom y Grabaciones Proyecto integrador: sprints Info general

Ocultar <>

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repasso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC

determinado?

localhost:3000/productos/12 - Google Chrome

localhost:3000/productos/12

Bienvenidos al detalle del producto 12

06:55

DigitalHouse >

determinado?

Ocultar <

Módulo: 5

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
 - ¿Qué vamos a ver en esta clase
 - Introducción a MVC
 - Controladores
 - Repaso de rutas
 - Rutas parametrizadas
 - Sistema de ruteo
 - Estructura de archivos
 - #BonusTrack express-generator
 - Hacia la clase en vivo

 Clase 19 - Clase en vivo: Patrón de diseño MVC

The screenshot shows a Visual Studio Code interface with the following details:

- Title Bar:** app.js - node - Visual Studio Code
- Left Sidebar (Icons):** Includes icons for file operations, search, file browser, terminal, and other development tools.
- Left Sidebar (Tree View):**
 - EXPLORER:** Shows a tree structure with a **NODE** folder containing `node_modules`, `app.js`, `package-lock.json`, and `package.json`.
 - OPEN EDITORS:** Shows `package.json` and `app.js` as open files.
 - OUTLINE** and **NPM SCRIPTS** are also listed.
- Central Area:** The main editor window displays the `app.js` file content. The code is a simple Express.js application with routes for the root, contact, product detail, and product comments.

```
JS app.js > ⓘ app.get('/productos/:idProducto/comentarios') callback
3 let app = express();
4
5 app.listen(3000, () => console.log('Esto fue exitoso'));
6
7 app.get('/', function(req, res) {
8   res.send('Bienvenidos al sitio!!!!!!');
9 });
10
11 app.get('/contacto', function(req, res) {
12   res.send('Dejanos tu contacto!');
13 });
14
15 app.get('/productos/:idProducto', function(req,res) {
16   res.send("Bienvenidos al detalle del producto " +
17     req.params.idProducto);
18 });
19 app.get('/productos/:idProducto/comentarios', function(req,
20 res) {
21   res.send("Bienvenidos a los comentarios del producto " +
22     req.params.idProducto);
23 });
```

- Bottom Status Bar:** Shows the current file is `app.js`, line 20, column 44, with 4 spaces, encoding is UTF-8, and the language is JavaScript.

playground.digitalhouse.com/course/2815/unit/01842f3b-f8a7-11eb-876f-12eb947d8fb1/lesson/27d57df7-f8a9-11eb-876f-12eb947d8fb1/topic/d21f9fa1-f8a7-11eb-876f-12...

Módulos Contenido Accesos a Zoom y Grabaciones Proyecto integrador: sprints Info general

Ocultar <>

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repasso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC

determinado?

localhost:3000/productos/12/comentarios - Google Chrome

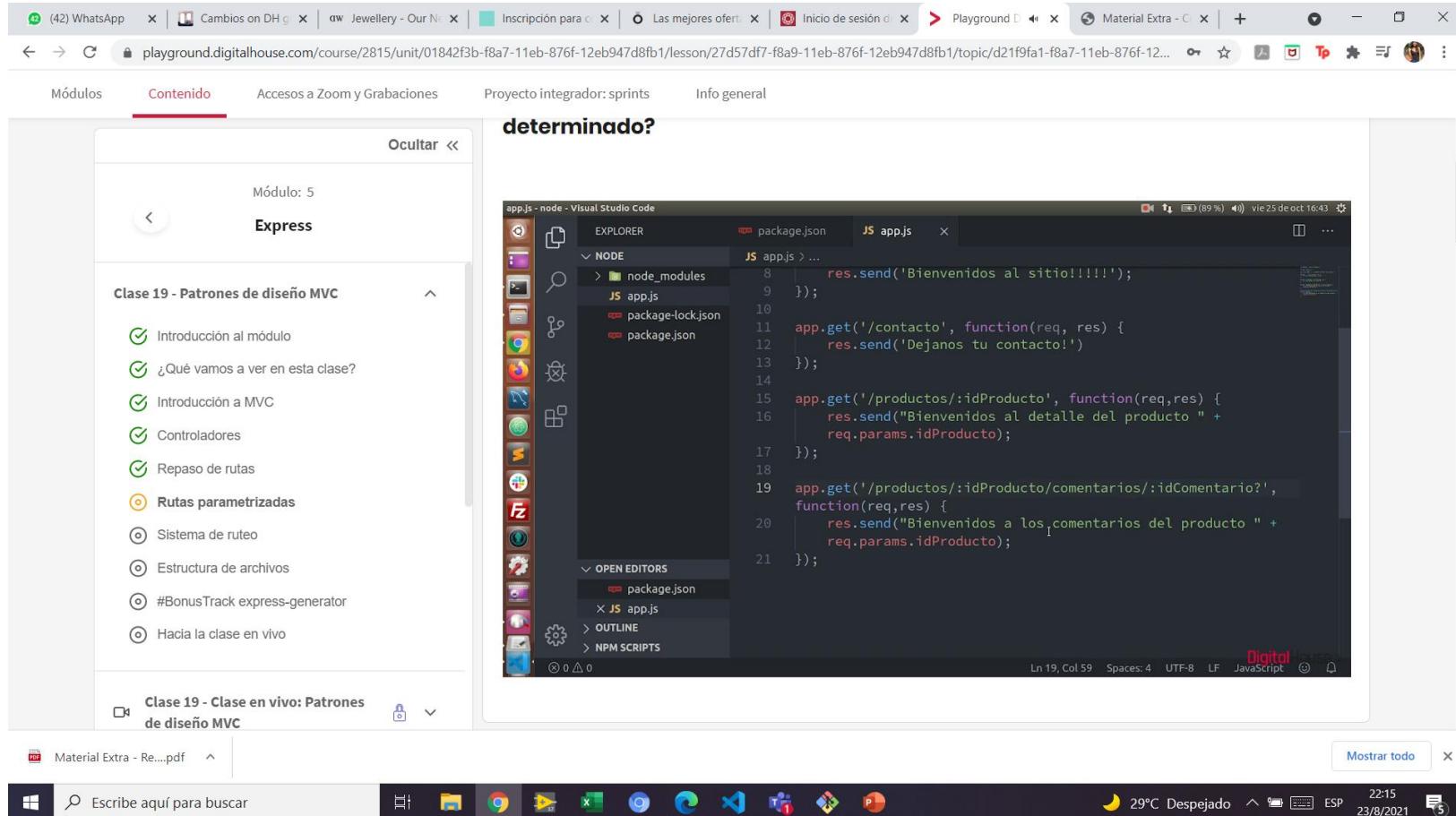
localhost:3000/productos/12/comentarios

Bienvenidos a los comentarios del producto 12

06:59

DigitalHouse>

? PARA QUE SEA OPTATIVO, FUNCIONASIN EL



¡Excelente! Gran trabajo.

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Dame una pista ▾

PlaygroundCode <> script.js ×

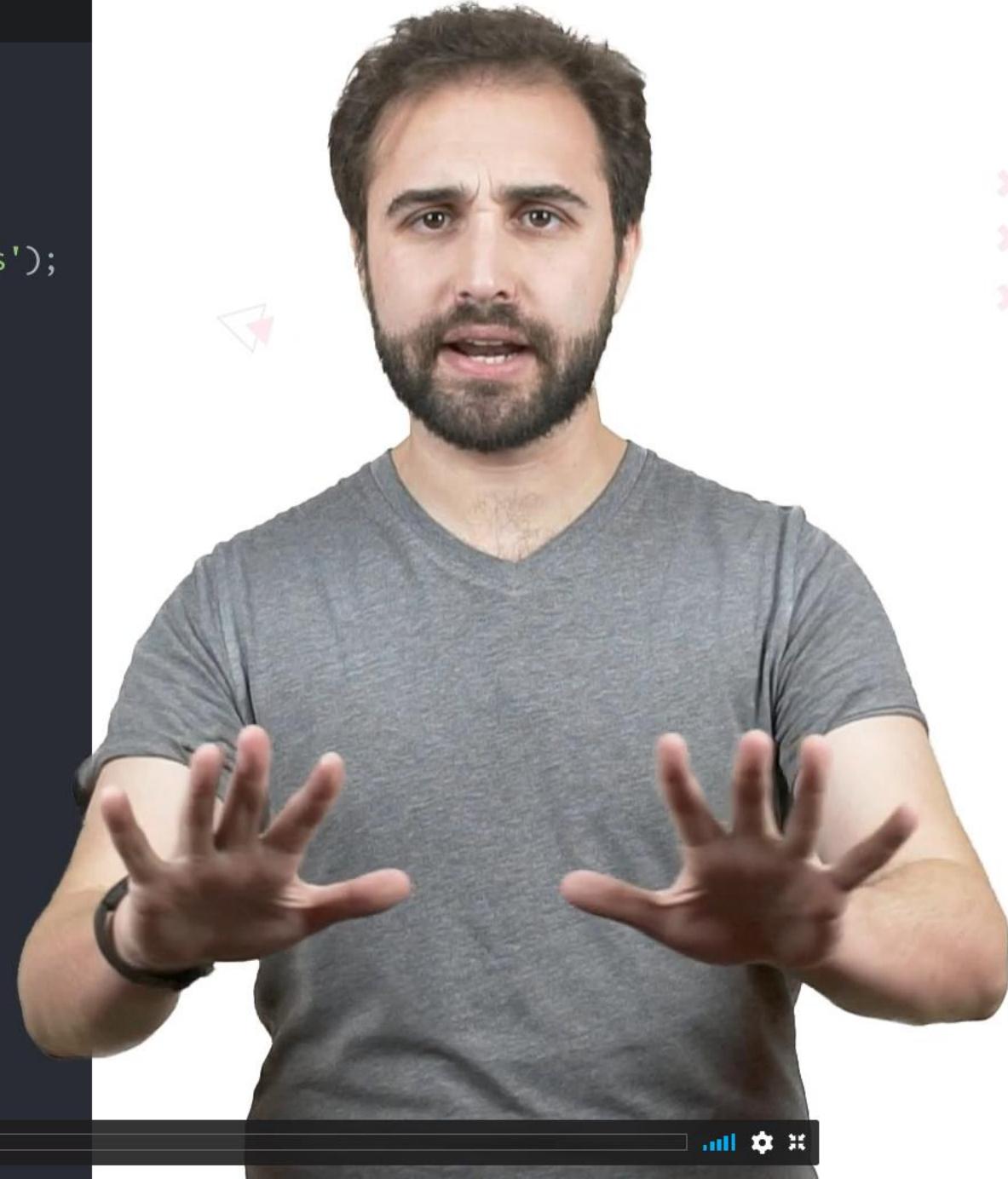
SRC PATH: SRC > SCRIPT.JS

script.js

```
const express = require('express');
const app = express();
const rutasSeries = require('./routes/series');
app.use("/series", rutasSeries)
```

Clase 19 - Clase en vivo: Patrones de diseño MVC

```
1 const express = require('express')
2 const app = express()
3 const rutasProductos = require('./routes/productos');
4
5 app.use('/productos', rutasProductos);
6 ↑
7
8
9
10
11 /productos/crear
12
13
14
15
16
17
```



Antes en app.js

The screenshot shows the Visual Studio Code interface with the following details:

- Left Sidebar:** A list titled "Clase 19 - Patrones de diseño MVC" containing the following items:
 - Introducción al módulo
 - ¿Qué vamos a ver en esta clase?
 - Introducción a MVC
 - Controladores
 - Repaso de rutas
 - Rutas parametrizadas
 - Sistema de ruteo
 - Estructura de archivos
 - #BonusTrack express-generator
 - Hacia la clase en vivo
- Explorer View:** Shows the file structure of a Node.js project:
 - node_modules
 - routes
 - main.js
 - marcas.js
 - productos.js
 - app.js
 - package-lock.json
 - package.json
- Editor View:** The "app.js" file is open, displaying the following code:

```
19 app.get('/productos/:idProducto', function(req,res) {  
20   res.send("Bienvenidos al detalle del producto " +  
21   req.params.idProducto);  
});  
  
23 app.get('/productos/:idProducto/comentarios/:idComentario?',  
function(req,res) {  
24   if (req.params.idComentario == undefined) {  
25     res.send("Bienvenidos a los comentarios del producto  
" + req.params.idProducto);  
26   } else {  
27     res.send("Bienvenidos a los comentarios del producto  
" + req.params.idProducto + " y estas enfocado en el  
comentario " + req.params.idComentario);  
28   }  
});  
});
```
- Open Editors:** Shows other open files:
 - package.json
 - app.js
 - productos.js routes
 - marcas.js routes
 - main.js routes

Después

Clase 19 - Patrones de diseño MVC

- ✓ Introducción al módulo
- ✓ ¿Qué vamos a ver en esta clase?
- ✓ Introducción a MVC
- ✓ Controladores
- ✓ Repaso de rutas
- ✓ Rutas parametrizadas
- Sistema de ruteo
- ✓ Estructura de archivos
- ✓ #BonusTrack express-generator
- ✓ Hacia la clase en vivo

```
File Edit Selection View Go Debug Terminal Help
EXPLORER package.json JS app.js JS productos.js × JS marcas.js JS main.js ...
NODE
  > node_modules
  > routes
    JS main.js
    JS marcas.js
    JS productos.js
    JS app.js
    package-lock.json
    package.json
OPEN EDITORS
  package.json
  JS app.js
  × JS productos.js routes
  JS marcas.js routes
  JS main.js routes
OUTLINE
NPM SCRIPTS
vie 25 de oct 16:51
1 let express = require('express');
2
3 let router = express.Router();
4
5 router.get('/:idProducto', function(req,res) {
6   res.send("Bienvenidos al detalle del producto " +
7   req.params.idProducto);
8
9 router.get('/:idProducto/comentarios/:idComentario?', function
10 (req,res) {
11   if (req.params.idComentario == undefined) {
12     res.send("Bienvenidos a los comentarios del producto
13     " + req.params.idProducto);
14   } else {
15     res.send("Bienvenidos a los comentarios del producto
16     " + req.params.idProducto + " y estas enfocado en el
17     comentario " + req.params.idComentario);
18   }
19
20 module.exports = router;
```

¡Excelente! Gran trabajo.

Ocultar <>

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC

La carpeta routes

app.js - node - Visual Studio Code

EXPLORER NODE JS app.js JS productos.js JS marcas.js JS main.js

routes
main.js
marcas.js
productos.js
app.js
package-lock.json
package.json

OPEN EDITORS package.json JS app.js JS productos.js routes JS marcas.js routes JS main.js routes

```
let express = require('express');
let rutasProductos = require('./routes/productos.js');
let rutasMain = require('./routes/main.js');

let app = express();

app.listen(3000, () => console.log('Esto fue exitoso'));

app.use('/productos', rutasProductos);
app.use('/', rutasMain);
```

Digital

¡Excelente! Gran trabajo.

Ocultar <>

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

La carpeta routes

main.js - node - Visual Studio Code

EXPLORER package.json JS app.js JS productos.js JS marcas.js JS main.js

routes > JS main.js > ...

```
1 let express = require('express');
2
3 let router = express.Router();
4
5 router.get('/', function(req, res) {
6   res.send('Bienvenidos al sitio!!!!!!');
7 });
8
9 router.get('/contacto', function(req, res) {
10   res.send('Dejanos tu contacto!');
11 });
12
13
14 module.exports = router;
```

OPEN EDITORS package.json JS app.js JS productos.js routes JS marcas.js routes X JS main.js routes OUTLINE NPM SCRIPTS

Digital

¡Excelente! Gran trabajo.

Ocultar <>

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repasso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Modularicemos las rutas para las series.

Queremos que el archivo `series.js` administre TODOS los request del recurso. Debemos, como primer paso, crear la variable `express` y almacenar en ella el módulo express.

Luego, deberemos crear la variable `router` y almacenar en ella la ejecución del método que nos permite manejar un sistema de rutas.

Dame una pista ^

Deberemos usar el método Router()

PlaygroundCode <> script.js x

SRC PATH: SRC > SCRIPT.JS

```
script.js
1 const express =require('express');
2
3 const router =express.Router();
```

Módulos

Contenido

Accesos a Zoom y Grabaciones

Proyecto integrador: sprints

Info general

Ocultar <>

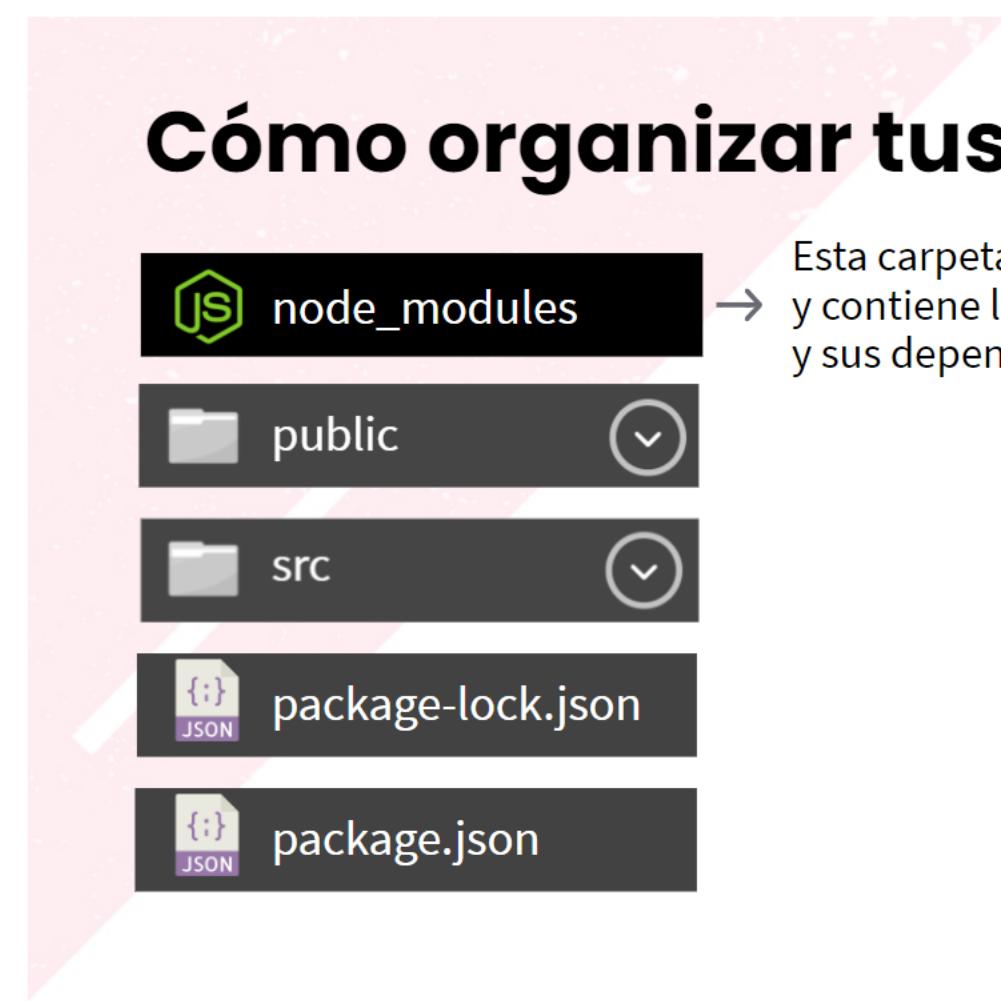
Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC



→ Esta carpeta la genera Node.js y contiene los módulos instalados y sus dependencias.

Módulos

Contenido

Accesos a Zoom y Grabaciones

Proyecto integrador: sprints

Info general

Ocultar <>

Módulo: 5

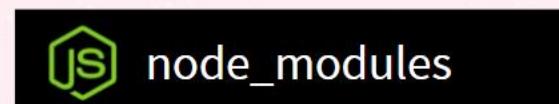
Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC

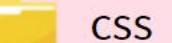
Cómo organizar tus archivos



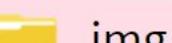
node_modules



public



css



img



js



src



package-lock.json

Aquí pondremos todos los recursos estáticos. Aquellos que no pasan por el sistema de ruteo de Node.js: imágenes, CSS y JavaScript de front-end.



Módulos **Contenido** Accesos a Zoom y Grabaciones Proyecto integrador: sprints Info general

Ocultar <>

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repasso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos**
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC

node_modules

- public
- src
 - controllers
 - database
 - routes
 - views
- app.js

package-lock.json

Aquí pondremos todo nuestro proyecto de NODE.

Módulos Contenido Accesos a Zoom y Grabaciones Proyecto integrador: sprints Info general

Ocultar <>

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC

views

partials

products

users

app.js

package-lock.json

package.json

DigitalHouse>
Coding School

Contiene el registro de las versiones específicas de los paquetes que se instalaron en nuestro proyecto.

Módulos

Contenido

Accesos a Zoom y Grabaciones

Proyecto integrador: sprints

Info general

Ocultar <>

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos**
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC

views

partials

products

users

app.js

package-lock.json

package.json

DigitalHouse>
Coding School

Contiene el registro de los paquetes que el proyecto requiere para funcionar y otras configuraciones del proyecto.



Módulos Contenido Accesos a Zoom y Grabaciones Proyecto integrador: sprints Info general

Ocultar <>

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos**
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC

hemos visto que es recomendable separar la estructura del proyecto según responsabilidades y aplicar el patrón de diseño Modelo Vista Controlador. Repasemos cómo podemos organizar dicha estructura. Para realizar este ejercicio debemos mover los archivos a la carpeta correspondiente.

node_modules

public

- css
- img
- js

src

- controllers
 - productsController.js
 - usersController.js
- database
- products.json
- routes
- main.js

Módulos

Contenido

Accesos a Zoom y Grabaciones

Proyecto integrador: sprints

Info general

Ocultar <>

Módulo: 5

Express

Clase 19 - Patrones de diseño MVC

- Introducción al módulo
- ¿Qué vamos a ver en esta clase?
- Introducción a MVC
- Controladores
- Repaso de rutas
- Rutas parametrizadas
- Sistema de ruteo
- Estructura de archivos**
- #BonusTrack express-generator
- Hacia la clase en vivo

Clase 19 - Clase en vivo: Patrones de diseño MVC



Verificar



Excelente, ya podemos armar la estructura de nuestros proyectos MVC

src

- controllers
 - productsController.js
 - usersController.js
- database
 - products.json
- routes
 - main.js
 - products.js
- views
 - partials
 - products
- app.js
- package-lock.json



Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda app.js - mi-portafolio-mvc - Visual Studio Code

EXPLORADOR ... JS app.js X JS mainController.js JS main.js 5 home.html style.css package.json

EDITORES ABIERTOS JS app.js > ...

```
1 const express = require('express');
2 const app = express();
3 const path = require("path");
4 const mainRouter = require("./routers/main");
5 const publicPath = path.resolve(__dirname, "./public");
6
7 app.use(express.static(publicPath));
8
9 app.use('/', mainRouter);
10
11 let port = process.env.PORT || 3000;
12
13 app.listen(port, () =>{
14   console.log('Servidor funcionando ' + port);
15});
```

MI-PORTAFOLIO-MVC controllers JS mainController.js node_modules public css style.css images australian.jpg bridge.jpg cathedral.jpg european.jpg hedgehog.jpg lights.jpg prague.jpg road.jpg routers main.js views about.html home.html app.js package-lock.json package.json

ESQUEMA METADATA

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL

[nodemon] starting `node app.js`
Servidor funcionando 3000
[nodemon] restarting due to changes...
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
Servidor funcionando 3000
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
[nodemon] restarting due to changes...
Servidor funcionando 3000
[nodemon] starting `node app.js`
Servidor funcionando 3000

0 △ 0 Escribe aquí para buscar 22:11 29°C Despejado ESP 24/8/2021 5 Lín. 9, col. 26 Espacios: 4 UTF-8 CRLF JavaScript

Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda mainController.js - mi-portafolio-mvc - Visual Studio Code

EXPLORADOR ... JS app.js JS mainController.js X JS main.js home.html style.css package.json

EDTORES ABIERTOS controllers > JS mainController.js > controller > about

```
1 const path = require ("path");
2
3
4 const controller = {
5   index: (req, res)=>{
6     res.sendFile(path.resolve("./views/home.html"));
7   },
8   about: (req, res)=>[
9     res.sendFile(path.resolve("./views/about.html"));
10 ]
11 };
12
13 module.exports = controller;
```

MI-PORTAFOLIO-MVC controllers JS mainController.js > node_modules public > css style.css > images australian.jpg bridge.jpg cathedral.jpg european.jpg hedgehog.jpg lights.jpg prague.jpg road.jpg > routers main.js > views about.html home.html app.js > package-lock.json > package.json

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL

```
[nodemon] starting `node app.js`
Servidor funcionando 3000
[nodemon] restarting due to changes...
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
Servidor funcionando 3000
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
[nodemon] restarting due to changes...
Servidor funcionando 3000
[nodemon] starting `node app.js`
Servidor funcionando 3000
```

ESQUEMA METADATA

0 △ 0 Escribe aquí para buscar

Lín. 10, col. 6 Espacios: 4 UTF-8 CRLF JavaScript 22:11 24/8/2021 29°C Despejado ESP 5

Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda main.js - mi-portfolio-mvc - Visual Studio Code

EXPLORADOR ... JS app.js JS mainController.js JS main.js X 5 home.html style.css package.json

EDTORES ABIERTOS routers > JS main.js > ...

```
1 const express = require("express");
2 const router = express.Router();
3 const mainController = require("../controllers/mainController");
4
5
6 router.get("/", mainController.index);
7 router.get("/about", mainController.about);
8
9 module.exports =router;
```

MI-PORTAFOLIO-MVC controllers JS mainController.js

- > node_modules
- public
 - css style.css
- images
 - australian.jpg
 - bridge.jpg
 - cathedral.jpg
 - european.jpg
 - hedgehog.jpg
 - lights.jpg
 - prague.jpg
 - road.jpg
- routers JS main.js
- views
 - about.html
 - home.html
- JS app.js
- package-lock.json
- package.json

ESQUEMA METADATA

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL

[nodemon] starting `node app.js`
Servidor funcionando 3000
[nodemon] restarting due to changes...
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
Servidor funcionando 3000
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
[nodemon] restarting due to changes...
Servidor funcionando 3000
[nodemon] starting `node app.js`
Servidor funcionando 3000

0 △ 0 Escribe aquí para buscar 29°C Despejado 22:11 24/8/2021

Lín. 7, col. 42 Espacios: 4 UTF-8 CRLF JavaScript