**UNIDAD 4: Integración con el Front-end**

1. [UNIDAD 4: Introducción](https://learn.nextu.com/mod/page/view.php?id=3574&pid=P_WEBDEV_V2)

Bienvenido a Interactuando con Bases de Datos

En esta unidad aprenderás a enlazar las bases de datos con el Front-end, usando el lenguaje PHP para acceder a una base de datos. Dicha a integración requiere la intervención de todas las partes del sistema. Además, guardarás y obtendrás información en una base de datos SQL y NoSQL.

**Unidad 4: Integración con el Front-end**  
¡Aquí comienza la Unidad 4: Integración con el Front-end!

Los objetivos del Programa que corresponden a esta Unidad son:

* Crear ambientes de programación y a gestionar recursos.

Los objetivos del Curso 7 que corresponden a esta Unidad son:

* Entender, diseñar y construir bases de datos NoSQL.
* Administrar información con bases de datos NoSQL.

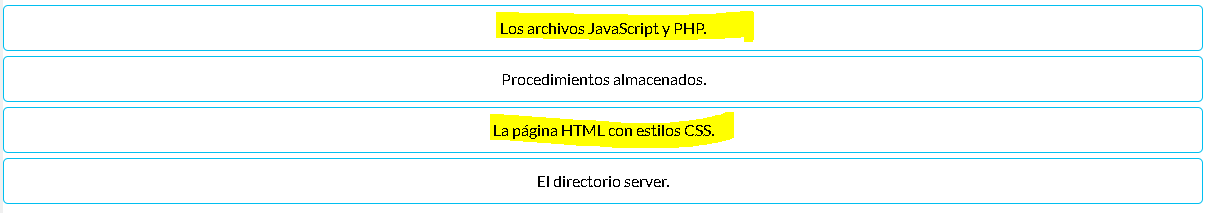
1. Lección 1: Guardar información en una base de datos SQL

[2.1. Acceso a la base de datos desde el Front-end y PHP](https://learn.nextu.com/mod/lesson/view.php?id=3575&pid=P_WEBDEV_V2)

2.2. Actividad 1

Tiempo de pensar

Hasta ahora hemos estudiado cómo enlazar las bases de datos con el Front-end y cómo usar PHP para acceder a una base de datos. Ahora te invitamos a poner en práctica tus conocimientos haciendo esta actividad.  
  
Instrucciones:  
De las opciones que ves a continuación, escoge aquellas que responden el enunciado.  
  
Enunciado:  
La integración de una base de datos con el Front-end requiere la intervención de todas las partes del sistema, ¿cuáles de las siguientes opciones necesitamos para llevar a cabo esta tarea?



2.3. Usuarios y roles

Para dar mayor seguridad a un sistema:

* Implementación de usuarios y roles
  + Acceso a algunas partes de la base de datos
  + Leer, Ingresar, Modificar registros

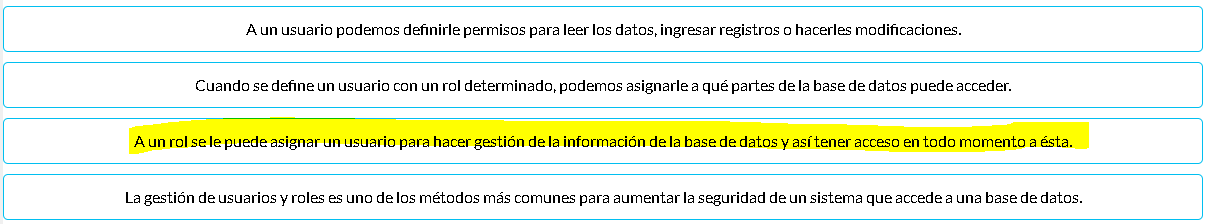
NOTA: si el json retorna errors en php se va a generar un error en el AJAX de vuelta, para evitar los mensajes de notice se debe colocar la línea:

error\_reporting(0);

2.4. Actividad 2

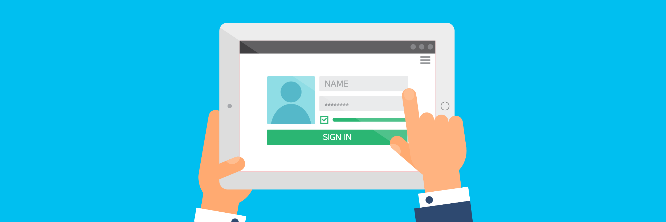
Tiempo de pensar

En esta ocasión estudiamos la integración del Front-end con la base de datos, haciendo uso de los usuarios y roles, pues es preciso realizar algunas acciones adicionales para aumentar la seguridad del sistema. ¿Qué te parece si, de acuerdo a lo que estudiamos, practicamos un poco? Realiza la siguiente actividad y afirma tu aprendizaje.  
  
Instrucciones:  
  
Lee atentamente el enunciado y escoge la respuesta más acertada.  
  
Enunciado:  
  
En el video que acabamos de ver, aprendimos algunas características de los usuarios y los roles. De las siguientes opciones, ¿cuál no es parte de ellas?

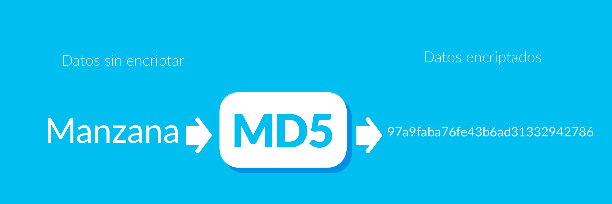


2.5. Lectura: Encriptación de contraseñas

Encriptación de contraseñas



El almacenamiento de la información en una base de datos involucra un viaje importante de cada dato, desde su digitación hasta llegar a un registro de una tabla. Es posible que, durante este trayecto, la información sea interceptada por violaciones a la seguridad del sistema y los datos caigan en manos no deseadas. Por lo tanto, una de las mejores estrategias para contrarrestar la fuga de información es la encriptación de los datos.  
  
Encriptar es una técnica para codificar información evitando que ésta sea accedida por terceros. En la informática se usan diferentes tipos de algoritmos para realizar esta codificación al enviar mensajes y decodificarlos al recibirlos.  
  
Al momento de guardar contraseñas en una base de datos, por ejemplo, es vital que éstas se codifiquen en el servidor mediante un algoritmo de encriptación, de tal manera que si alguien no deseado logra acceder a la base de datos, no podrá descifrar cuáles son realmente los datos que está viendo.  
  
PHP ofrece un algoritmo de codificación que le otorga gran seguridad al manejo de contraseñas. Los algoritmos de encriptación más comunes son MD5, SHA1 o SHA256, los cuales son métodos que, a partir de una cadena de caracteres ingresada por el usuario, generan un resultado encriptado de la información.



En la gráfica puedes ver que tenemos un dato con la palabra ‘Manzana’ y al someterlo al algoritmo de encriptación MD5 se transforma en un conjunto de números y letras de una longitud mayor. Como puedes verlo, es imposible descifrar la cadena encriptada a simple vista.  
  
Estos algoritmos son los más comunes debido a su rapidez y eficiencia de operación, sin embargo no son lo suficientemente seguros para ser aplicados a las contraseñas de nuestros sistemas de información, ya que con técnicas modernas de fuerza bruta se convierte en algo sencillo decodificarlas. Es por esto que, para algo tan importante como una contraseña, se deben usar algoritmos de máxima seguridad que reduzcan su vulnerabilidad al máximo.  
  
PHP ofrece dentro de su core un método para la encriptación de contraseñas llamado *password\_hash().* Este método recibe como parámetro la cadena a codificar y una constante de algoritmo de contraseñas que indique bajo qué algoritmo se realizará el proceso. Se recomienda usar el valor por defecto con la constante PASSWORD\_DEFAULT, en la encriptación la palabra hash hace referencia al resultado del procedimiento, es decir, la cadena encriptada.

password\_hash(‘Manzana’, PASSWORD\_DEFAULT)

El valor retornado por el método es una cadena de caracteres correspondiente al hash del valor original. Este valor es el que se debe almacenar en la base de datos, sin embargo, es importante que, al definir la tabla que lo va a almacenar, su campo correspondiente sea de tipo cadena de caracteres y con un tamaño de 255, ya que la longitud del hash puede variar.  
  
Si necesitamos verificar una contraseña ingresada con la almacenada en una base de datos, tal y como se realiza en un proceso de Log in, se usa el método password\_verify(). Este método recibe como parámetros una cadena a comparar y un hash creado por medio de la función *password\_hash.*

password\_verify(‘Árbol’, $hash)

2.6. Del formulario a la Base de Datos

1. Lección 2: Obtener información desde una base de datos SQL

[3.1. Lectura: fetch\_assoc](https://learn.nextu.com/mod/lesson/view.php?id=3576&pid=P_WEBDEV_V2)

Fetch\_assoc



Una de las principales necesidades al momento de integrar una base de datos a un sistema Web, es mostrar la información que se almacena de manera óptima en el navegador. En este punto, sabes que para obtener información de la base de datos se usan las consultas SQL. Sin embargo, es necesario hacer uso de algunas herramientas que provee PHP para enviar los resultados de dichas consultas a la vista para que sean renderizadas.  
  
El resultado de la ejecución de una consulta sobre la base de datos desde PHP, se almacena en un objeto que almacena toda la información recibida.

$resultado = $conexion->query(‘SELECT \* FROM tabla’);

Sin embargo, es preciso que el objeto con el resultado de la consulta sea procesado para poder acceder a los datos que contiene. Este procesamiento se realiza por medio del método fetch\_assoc, el cual convierte un objeto, resultado de una consulta SQL, en un arreglo asociativo de valores. De esta manera, podemos acceder a los valores recibidos de la consulta, indicando, en la llave del arreglo asociativo, el nombre del campo al que deseamos acceder.

$fila = $resultado->fetch\_assoc();

Es importante tener en cuenta que éste método sólo convierte una fila del resultado a la vez. Esto quiere decir que, si la consulta arroja más de una fila como resultado, debemos usar un ciclo para recorrer la información de todas las filas. Generalmente se usa un ciclo while para realizar este recorrido.

while($fila = $resultado->fetch\_assoc()){

echo $fila[‘nombre’];

}

Este método retorna true si los valores coinciden, de lo contrario retorna false.  
  
En el ejemplo anterior, vemos en pantalla el valor del campo nombre arrojado por la consulta de todas las filas resultantes, así que la salida en pantalla puede ser algo similar a lo siguiente:

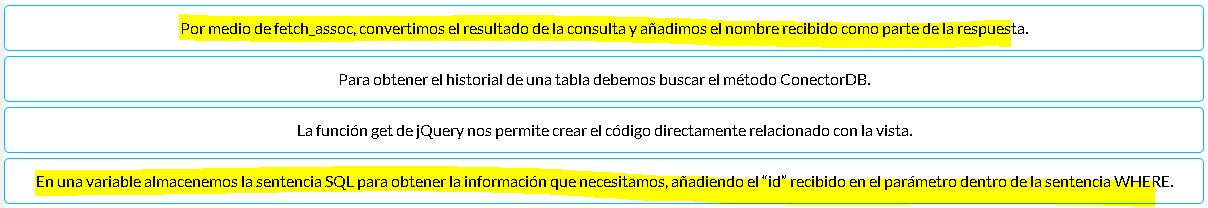
Juan Ramirez  
Pedro García  
Carlos Suarez

3.2. Mostrar información en el front-end

3.3. Actividad

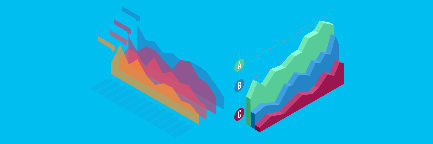
Tiempo de pensar

Hemos analizado sobre cómo obtener información desde una base de datos SQL. Te invitamos a que realices la siguiente actividad para que valides tu conocimiento.  
  
Instrucciones:  
Lee la pregunta con atención y selecciona las opciones que crees que son correctas.  
  
Enunciado:  
¿Cuál de estas opciones hacen parte del proceso para mostrar la información que se encuentra en la base de datos SQL en el Front-end?

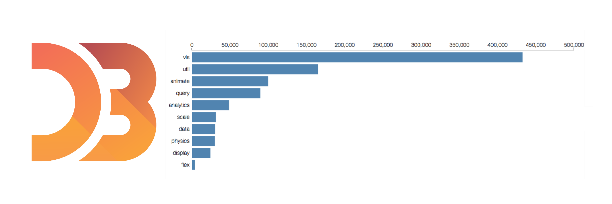


3.4. Lectura: Gráficos a partir de consultas

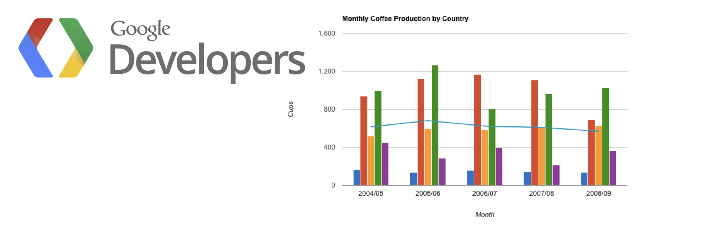
Gráficos a partir de consultas



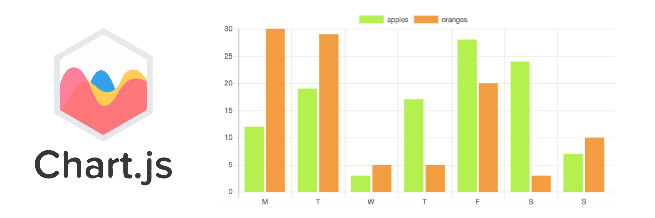
Dentro de las muchas aplicaciones que se le puede dar a una integración entre el front-end y una base de datos, la presentación de los datos en gráficas es una de las más importantes. La presentación de la información gráficamente es una práctica comúnmente usada para la fácil toma de decisiones, y por lo tanto es muy importante que estés preparado para implementar este tipo de herramientas en tus proyectos web.  
  
Existe una gran cantidad de herramientas que permiten tomar datos y crear gráficas vistozas e interactivas mediante JavaScript, HTML5 y SVG. A continuación veremos unas de las librerías más usadas para esta práctica.  
  
**D3.js**  
Esta librería es una de las más populares debido a su flexibilidad y modularidad, lo cual permite crear tantas gráficas como puedas imaginarlas. D3 renderiza gráficos a partir de cualquier tipo de datos y los renderiza a través de HTML5 y SVG, siendo compatible con jQuery y Prototype para la definición JavaScript de sus componentes.  
  
Ejemplo de gráfico con D3:



**Google Charts**  
  
La librería propuesta por Google para la realización de gráficos. Es muy sencilla de utilizar y contiene una gran cantidad de gráficos prediseñados. Usa HTML5 y SVG, siendo compatible con Internet Explorer y dispositivos móviles.  
Ejemplo de gráfico con Google Charts:



**Chart JS**  
  
Esta librería proporciona gráficos con diseños planos y altamente estéticos. Se caracteriza por su compatibilidad con frameworks JavaScripts como React.js. Muestra los gráficos a través de canvas de HTML5 y es compatible con Internet Explorer mediante Polyfill. Sus modelos son adaptativos por defecto, lo que permite una renderización adecuada en smartphones y tablets.  
  
Ejemplo de gráfico en Chart Js:



3.5. Ejercicio

HORA DE  
**codificar**

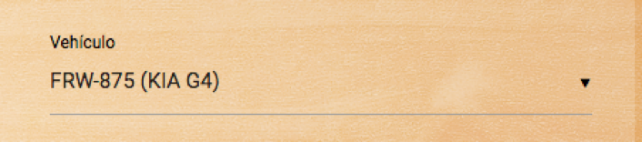
En la misma línea de importancia en que se encuentra la calidad en el almacenamiento de información en una base de datos, se encuentra su presentación visual. Una presentación adecuada de datos a partir de una cantidad determinada de datos comprende un proceso complejo de peticiones y envío de información entre cliente y servidor. Es por tal motivo que se debe tener muy en cuenta el rol que cumple cada parte del proceso en la operación de consulta a una base de datos y su presentación en el Front-end.  
  
En este ejercicio completarás el sistema web para la gestión de viajes que hemos venido realizando durante esta lección. En esta ocasión debes añadir la funcionalidad al sistema de agregar un nuevo viaje, y almacenarlo en la base de datos.

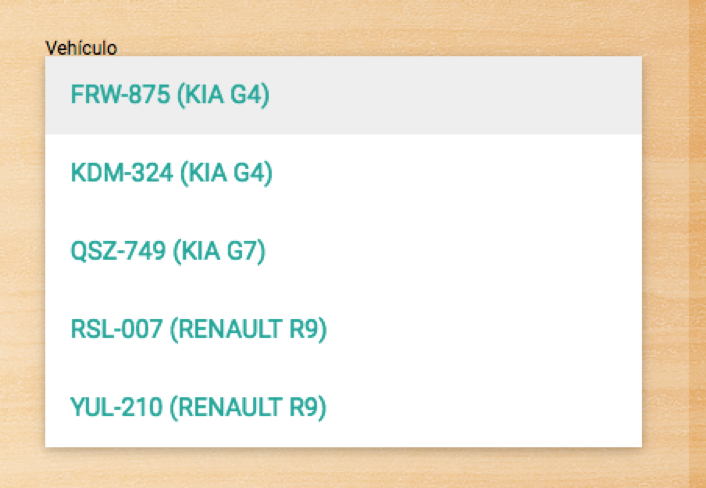
Contenido

Para el desarrollo de este ejercicio debes descargar el código base que contiene el sistema que se ha desarrollado durante esta lección y la anterior, adicionalmente, cuenta con un archivo SQL que debes ejecutar en tu base de datos MySQL para montar la base de datos compatible con el sistema. Antes de ejecutar el archivo .sql debes haber creado una base de datos con el nombre transporte\_db en tu servidor de bases de datos, posteriormente ejecutas el código en el archivo. Sigue las instrucciones para añadir la funcionalidad de añadir un nuevo viaje en la base de datos.

Instrucciones

1. Observa el archivo new\_viaje.html el cual cuenta con una serie de elementos tipo select. Cada uno debe cargar sus opciones con la misma información depositada en la base de datos. Crea en el archivo new\_viaje.js una petición al servidor que obtenga los datos correspondientes a cada select y los añada a la página antes de invocar el método material\_select(). En el ejemplo se muestra el resultado en el select para el vehículo, que obtiene las placas de los vehiculos en la base de datos y los añade al select, luego se invoca la función material\_select() para que se construya el selector de materializecss.





1. En el archivo select\_info.php consulta a la base de datos cada uno de los campos necesarios para construir los 4 selects del formulario. Para los select ciudad origen y ciudad destino debes obtener de la base de datos, los nombres de todas las ciudades allí almacenadas; para el select de vehículos, las placas de los registros almacenados y para el select conductor, el nombre de los usuarios en la base de datos. Retorna un JSON con toda la información necesaria a la vista.
2. En el archivo new\_viaje.js, crea las funciones necesarias para obtener los datos diligenciados de los formularios y enviarlos al archivo add\_viaje.php. Con la respuesta del servidor debes mostrar un mensaje de éxito y redirigir a la página welcome.html, o mostrar un mensaje de error si aplica.
3. Crea en el archivo add\_viaje.php todo el código necesario para tomar los datos recibidos de la vista y añadir un nuevo registro a la tabla viajes de la base de datos. Retorna a la vista un mensaje de éxito, o uno de error correspondientemente.

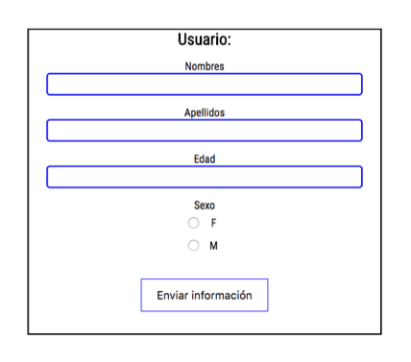
Buenas Prácticas

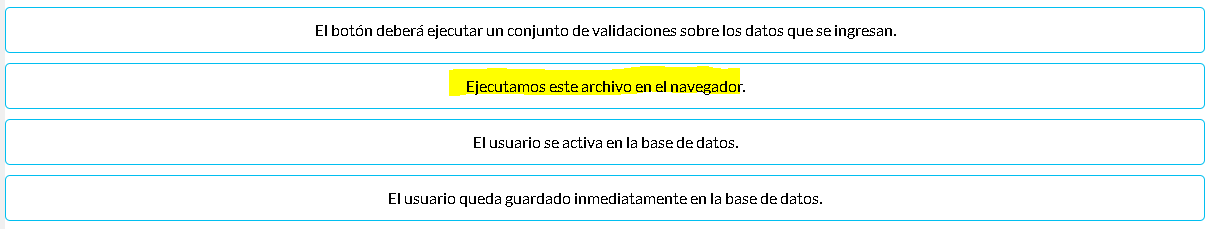
Mantener el modelo Modelo Vista Controlador, es muy importante al interactuar con bases de datos ya que esto permite diferenciar segmentos del sistema y aportar a la seguridad y mantenibilidad del código. Es por este motivo, que se hacen uso de peticiones AJAX para la interacción del Front-end con el servidor.

1. Lección 3: Guardar información en una base de datos NoSQL
   1. [Explicación del frontend](https://learn.nextu.com/mod/lesson/view.php?id=3577&pid=P_WEBDEV_V2)
   2. Actividad 1

Tiempo de pensar

Hemos realizado una aplicación en Node.js que se conecta a MongoDB y, por medio de una interfaz gráfica construida con el framework Bootstrap, creamos unos campos para registrar usuarios y guardarlos directamente en la base de datos. Ahora es tiempo de reforzar lo aprendido a través de una actividad.  
  
Instrucciones:  
  
Lee detenidamente el enunciado y responde a la pregunta.  
Enunciado:  
  
De acuerdo con los contenidos que hemos estudiado, observa la imagen y responde la siguiente pregunta:  
  
¿Cuál de las siguientes opciones no corresponde a lo que pasa cuando presionamos el botón Enviar información?





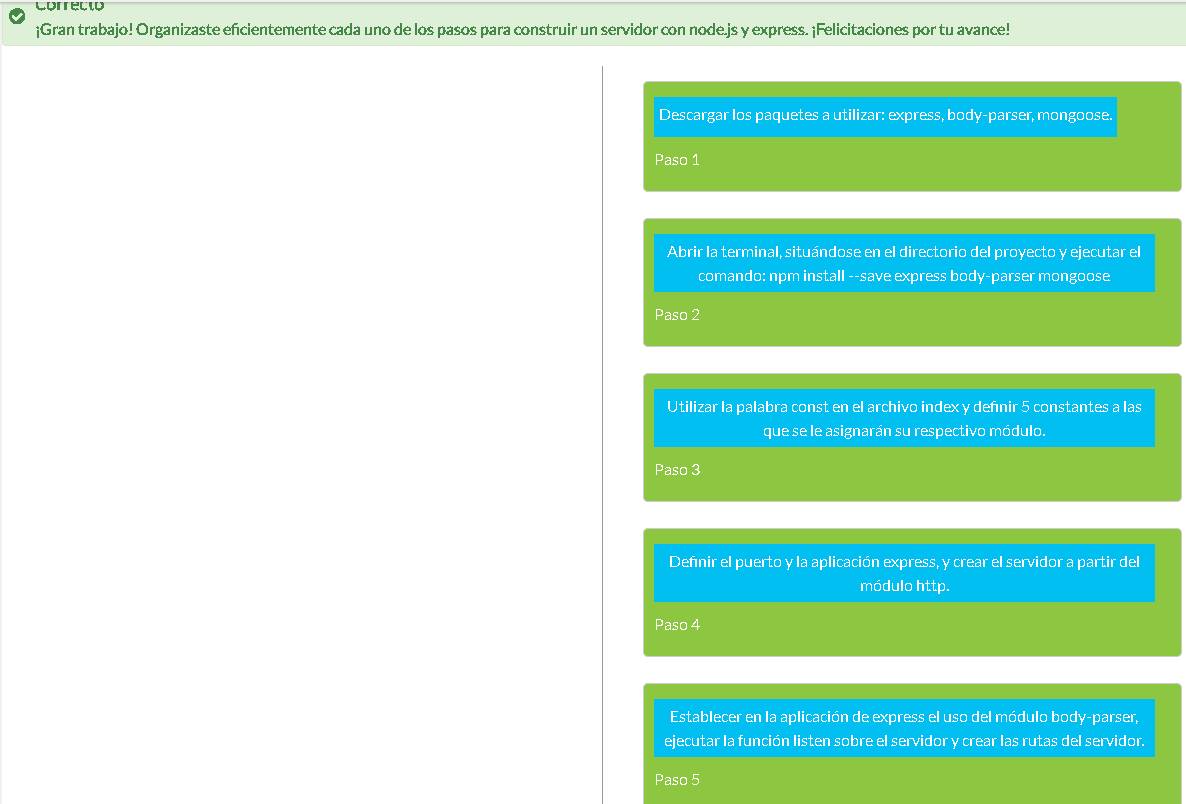
* 1. Construcción del server y rutas



* 1. Actividad 2

Tiempo de pensar

Hemos construido un servidor en node.js utilizando el módulo express. Este servidor funciona como intermedio entre el cliente y la base de datos y siempre debe haber una capa que se encarga de comunicar estas dos partes. Afianza tus conocimientos con esta actividad.  
  
Instrucción:  
  
En esta oportunidad, tu trabajo será relacionar cada servicio con su respectiva definición. ¡Adelante!



* 1. Lectura: Conexión con mongoDB, esquema y modelo usuario

Conexión con MongoDB, esquema y modelo usuario

En los videos anteriores, construimos el servidor y las rutas necesarias para la interacción entre el cliente, el servidor y la base de datos.  
  
En esta pequeña lectura vamos a agregar la conexión a la base de datos y a generar el modelo con su esquema correspondiente para la colección usuarios.  
  
1. Crea un archivo dentro del directorio server llamado model.js:

const mongoose = require('mongoose')

const Schema = mongoose.Schema

let UserSchema = new Schema({

userId:{ type: Number, require: true, unique:true },

nombres:{ type: String, require: true },

apellidos:{ type: String, require: true },

edad:{ type: Number, require: true },

sexo:{ type: String, require: true, enum: ['M', 'S']},

estado:{ type: String, require: true, enum: ['Activo', 'Inactivo'] }

})

lef UserModel = mongoose.model('Usuario', UserSchema)

module.exports = UserSchema

2. Conexión con MongoDB a través de Mongoose:

mongoose.connect('mongodb://localhost/c7')

La base de datos la llamaremos c7 (curso 7). Para probar que la conexión se realiza con éxito y se crean tanto la base de datos como la colección, abriremos una terminal: entramos al REPL de Mongo y listamos las bases de datos con el comando show dbs. Recuerda tener corriendo el servicio MongoD.  
  
3. Ve a las bases de datos y verifica que existe c7:

>show dbs

c7 0.000GB

4. Usa la base de datos y verifica que exista la colección usuarios:

> use c7

switched to db c7

> show colletion

usuarios

>

Con esto, nuestro servidor se encuentra correctamente enlazado con MongoDB y tiene el modelo de usuarios definido.

4.6. Ejercicio

1. Lección 4: Obtener información desde una base de datos NoSQL
2. UNIDAD 4: Prueba