

**DESARROLLO DE APLICACIONES WEB AVANZADO**

**LABORATORIO N° 05**

**Drives para Node.js de MongoDB**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Alumno(s):*** | ***Sosa Cruz Joshua Jesús*** | | | | | | ***Nota*** |  | |
| ***Grupo:*** |  | | ***Ciclo:V*** | | | | | | |
| ***Criterio de Evaluación*** | | ***Excelente (4pts)*** | | ***Bueno (3pts)*** | ***Requiere mejora (2pts)*** | ***No acept. (0pts)*** | | | ***Puntaje Logrado*** |
| Entiende cómo funciona mongoose | |  | |  |  |  | | |  |
| Utiliza mixins para reutilización de código | |  | |  |  |  | | |  |
| Desarrolla aplicaciones web | |  | |  |  |  | | |  |
| Realiza con éxito lo propuesto en la tarea. | |  | |  |  |  | | |  |
| Es puntual y redacta el informe adecuadamente | |  | |  |  |  | | |  |

**Laboratorio 05:**

**Drives para Node.js de MongoDB**

**Objetivos:**

Al finalizar el laboratorio el estudiante será capaz de:

* Entender el funcionamiento de Jade
* Desarrollar aplicaciones web con el framework Express
* Implementación correcta de mongoose

**Seguridad:**

* Ubicar maletines y/o mochilas en el gabinete del aula de Laboratorio.
* No ingresar con líquidos, ni comida al aula de Laboratorio.
* Al culminar la sesión de laboratorio apagar correctamente la computadora y la pantalla, y ordenar las sillas utilizadas.

**Equipos y Materiales:**

* Una computadora con:
* Windows 7 o superior
* VMware Workstation 10+ o VMware Player 7+
* Conexión a la red del laboratorio
* Máquinas virtuales:
* Windows 7 Pro 64bits Español - Plantilla
* Instalador de node.js

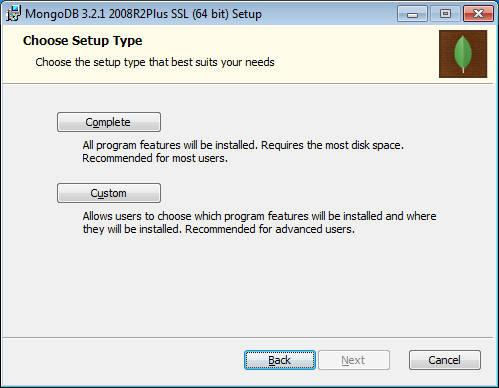
**Procedimiento:**

**Lab Setup**

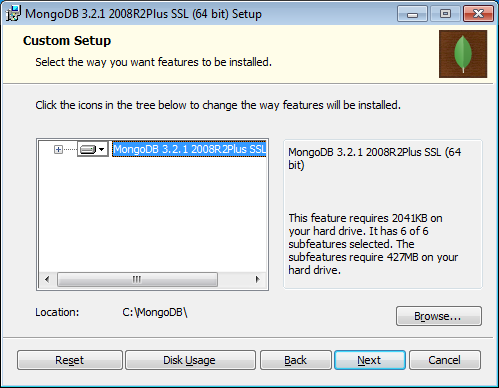
1. **Proceso de instalación de MongoDB**
   1. Solicite al instructor, el archivo de instalación de MongoDB

**mongodb-win32-x86\_64-2008plus-ssl-3.2.1-signed.msi**

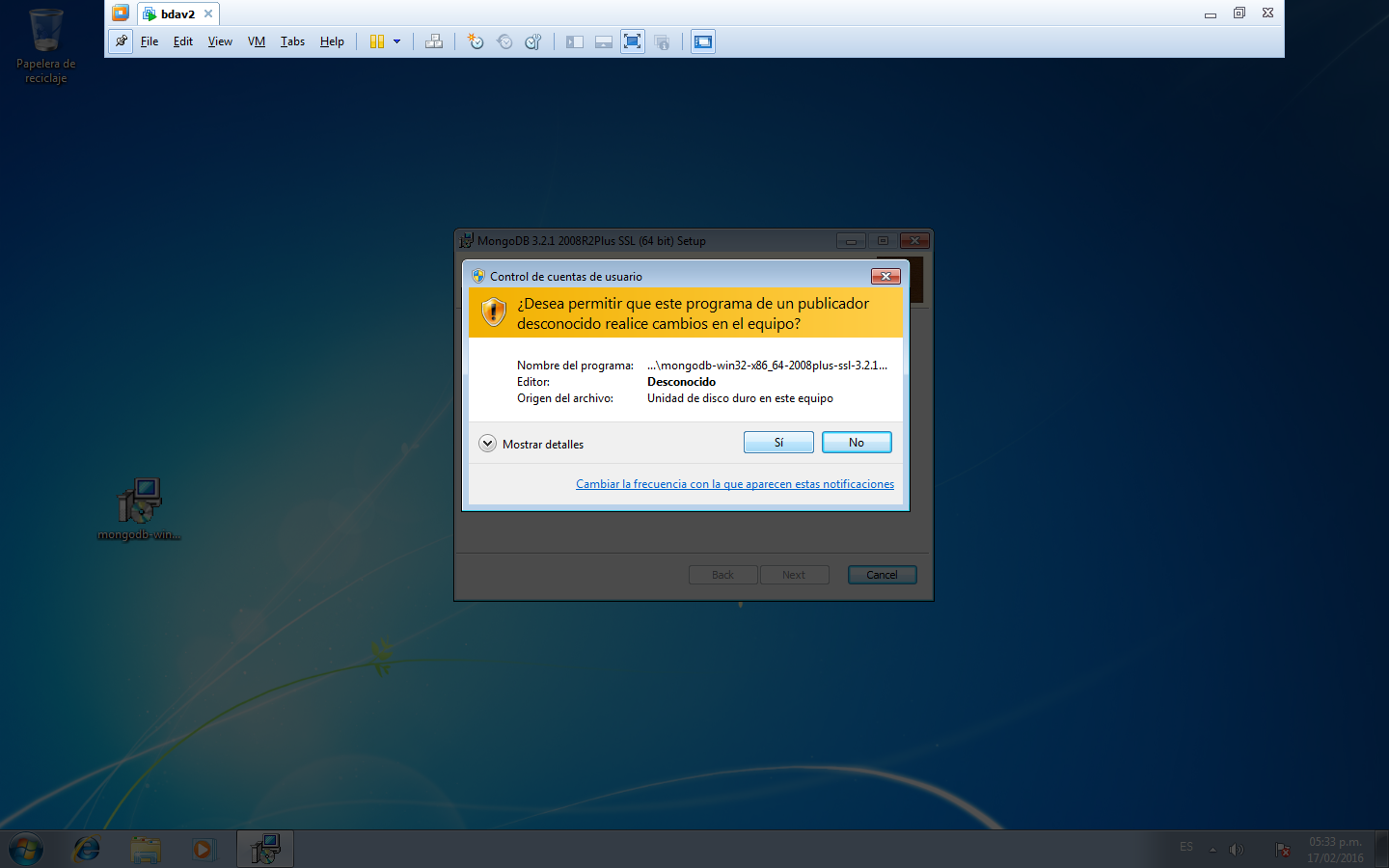
* 1. Copie el archivo instalador al escritorio del equipo virtual
  2. Doble clic en el instalador para iniciar el proceso de instalación.
  3. Cuando llegue a la ventana “**Choose Setup Type**”, elija la opción “**Custom**”



* 1. En la ventana “**Custom Setup**”, empleando el botón **Browse**, cambie la ruta de instalación a la carpeta **C:\MongoDB**



* 1. Proceda con la instalación del programa
  2. En caso se solicite autorizar cambios que realice el programa. Clic en el botón **Si**



* 1. Espere que finalice el proceso de **instalación**. Clic en el botón **Finish**

**Configuración de MongoDB**

1. **Creación de la carpeta para almacenamiento de datos**
   1. Crear una carpeta de nombre **Data** dentro de la carpeta **c:\MongoDB**
   2. Crear una carpeta de nombre **db** dentro de la carpeta **c:\MongoDB\Data**
   3. Ejecutar el siguiente comando dentro de **C:\MongoDB\bin**



* 1. Esto iniciará la instancia de MongoDB a la que accederemos durante el laboratorio. Para conectarse a dicha instancia, en otra consola ubicada en **C:\MongoDB\bin** ejecutamos:



* 1. Estamos conectados a la base de datos **test**, para validar que todo está ok ingresaremos un documento de prueba y luego haremos la consulta para ver su existencia.





1. **Continuación de laboratorio anterior**
   1. En el laboratorio anterior usted creó un sitio web con el Framework Express, para poder representar el trabajo de su grupo de proyecto. Usará el mismo código para nuestra sesión actual.
   2. Ya que tenemos node.js instalado y express configurado, procederemos a instalar mongoose para continuar con nuestra implementación.



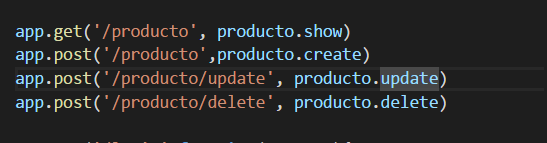
* 1. Usaremos también body-parser, el cual nos permite hacer un tratamiento de los argumentos recibidos durante las peticiones.



* 1. En la raíz de nuestro proyecto, crearemos el archivo lab10.js con el siguiente contenido



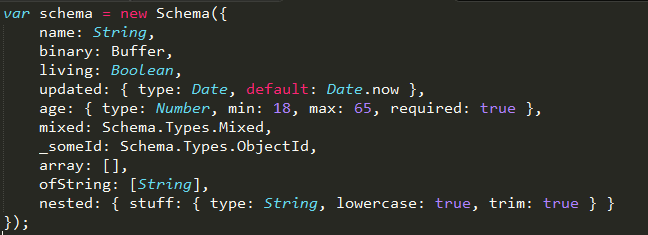
* 1. Estamos creando un tratamiento de datos para nuestra colección producto a través de las cuatro operaciones principales del CRUD, indique que líneas corresponden a cada una de estas.



* 1. Necesitamos crear la carpeta models donde ubicaremos nuestros archivos de conexión a MongoDB. Dentro de esta carpeta crearemos el archivo prod.js con el siguiente contenido.

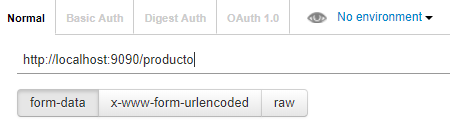


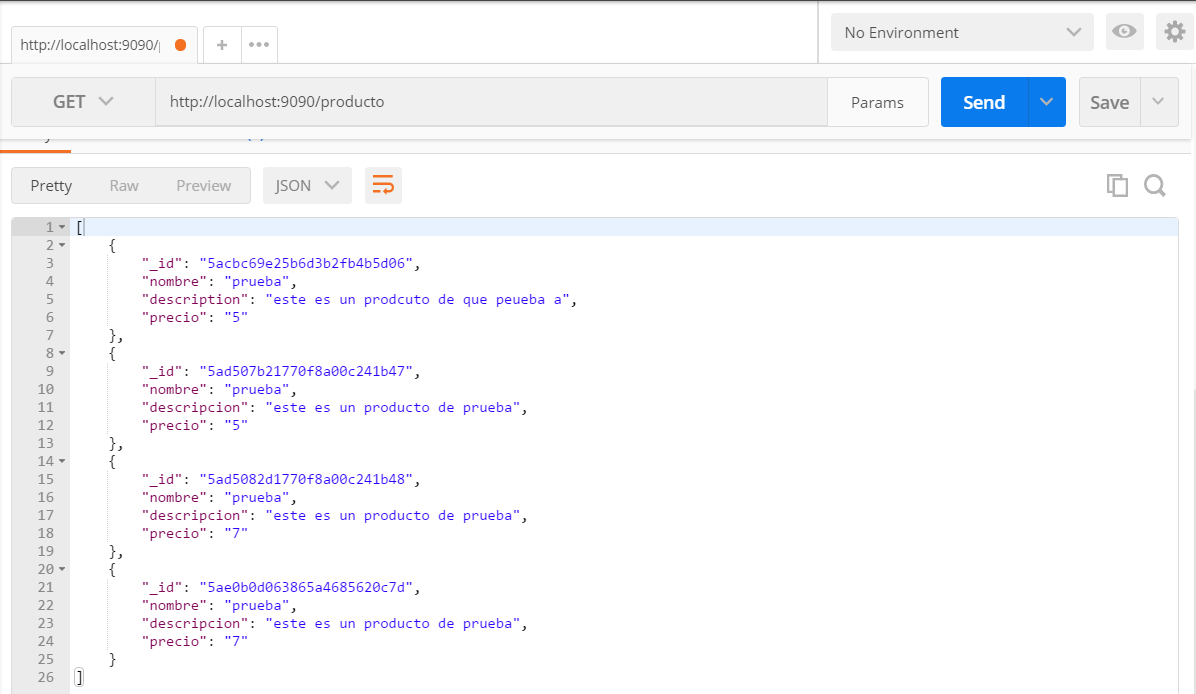
* 1. Para poder hacer operaciones sobre las colecciones de MongoDB, debemos declarar un Schema. Nosotros hemos declarado uno con los campos de producto. Así como MongoDB proporciona distintos tipos de variables, mongoose (el encargado de la conexión con node.js) también soporta varios tipos de variables en la declaración del Schema.



* 1. En el módulo exportado del archivo prod.js, hemos creado una función show. Esta hace referencia a prod\_model, que es un modelo del Schema antes creado. Nosotros hacemos operaciones sobre los modelos, quienes a su vez utilizan los esquemas para poder parsear la data recibida; es decir, para comunicarnos con MongoDB desde node.js necesitamos de un modelo, y el modelo necesita un Schema para poder conocer la estructura de datos a recibir.

En el laboratorio de la semana 4 instalamos el navegador Chrome con la extensión Postman REST. Haga uso de él para hacer las pruebas pertinentes.

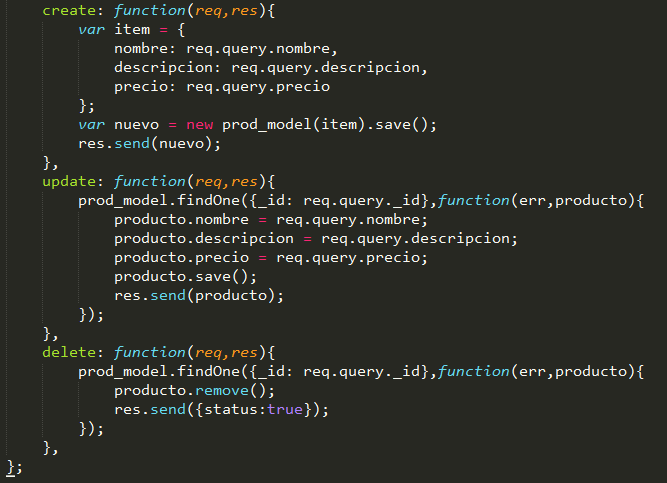




* 1. Adjunte una captura del resultado de ejecutar un GET en la url localhost:9090/producto

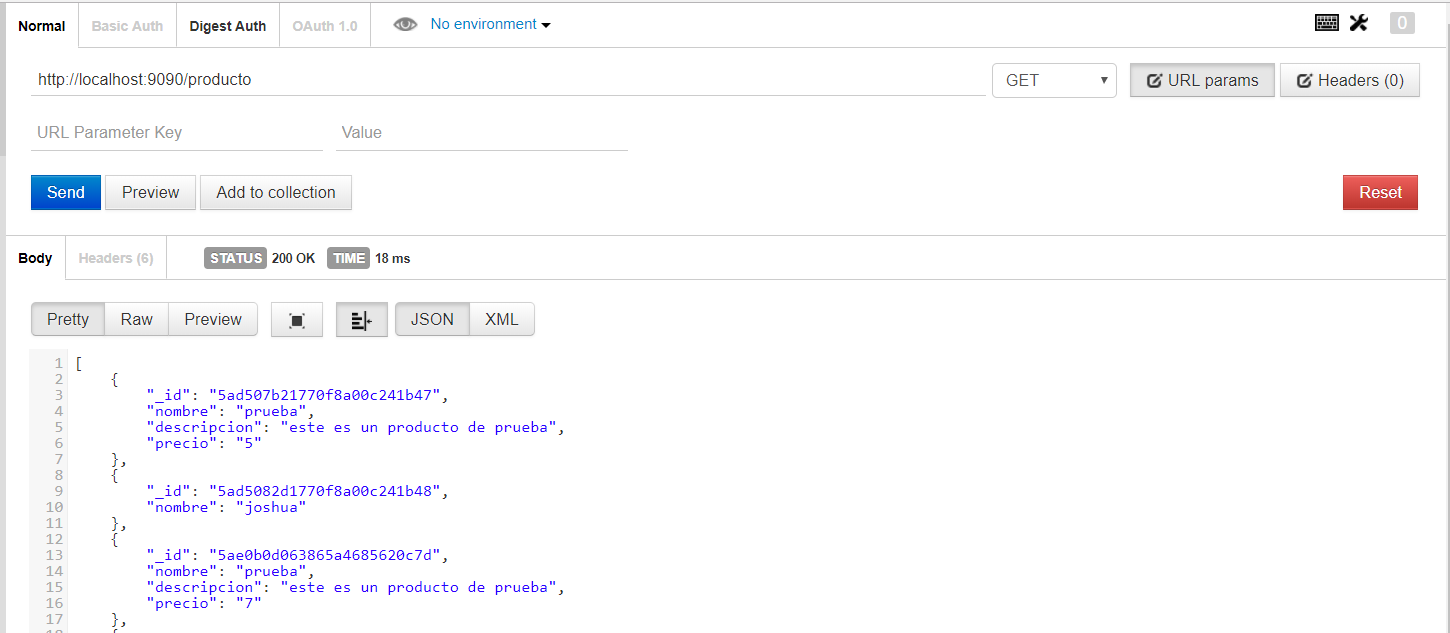
Ya hemos probado que tenemos conexión con nuestra base de datos, procederemos a crear las funciones restantes para tratar con toda la data. El archivo prod.js terminará luciendo de la siguiente manera.



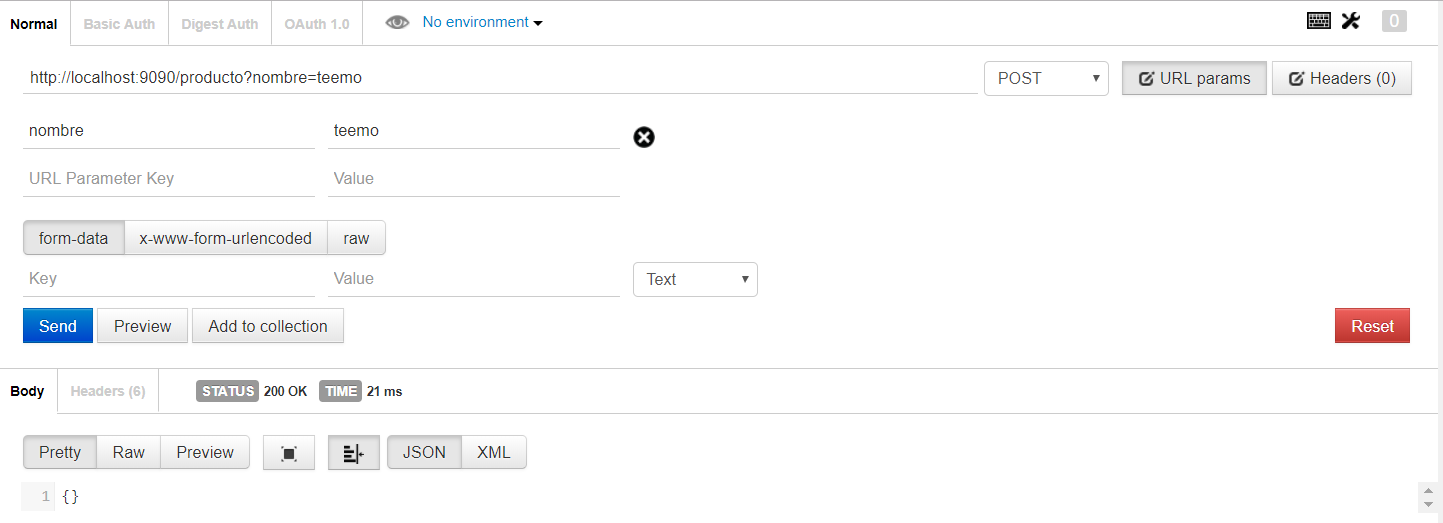


* 1. Adjunte capturas haciendo pruebas de las cinco operaciones, tome en cuenta que la url localhost:9090/producto ahora acepta un argumento \_id opcional, en caso de ser recibido, devolverá el documento que coincida con ese \_id.

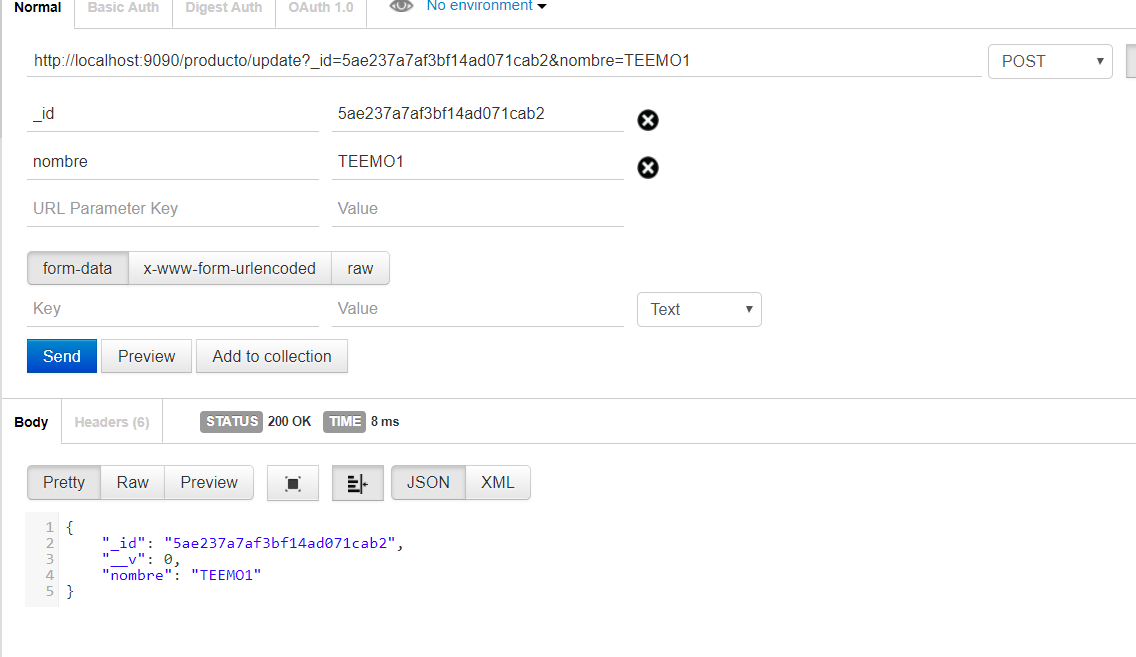
**Show:**



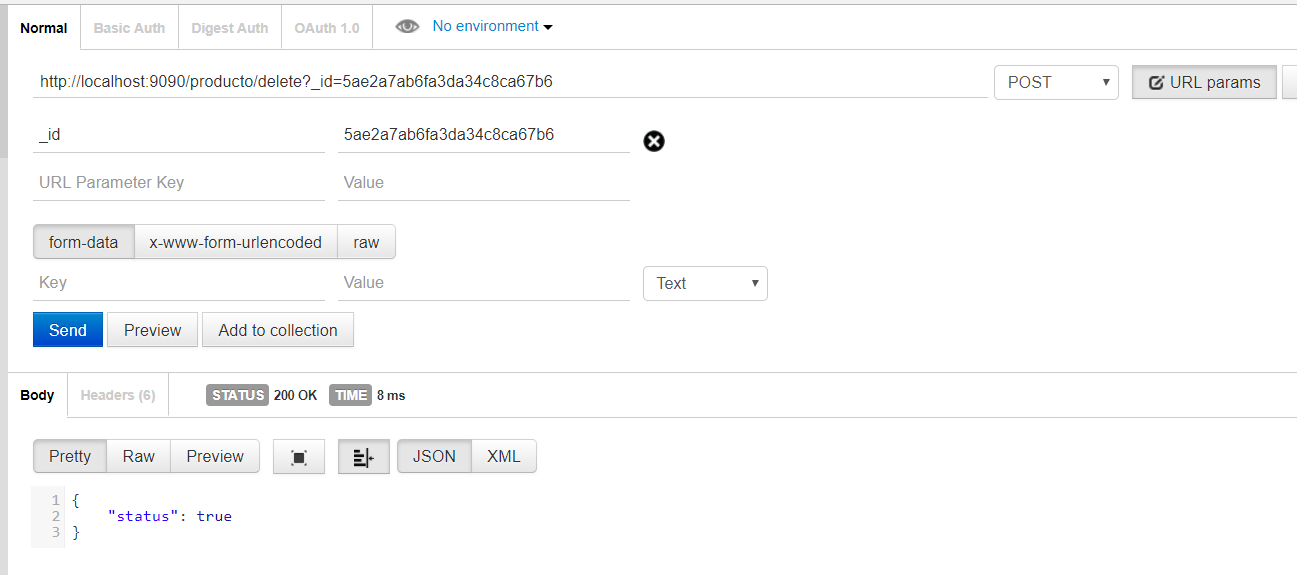
**Create:**



**Update**



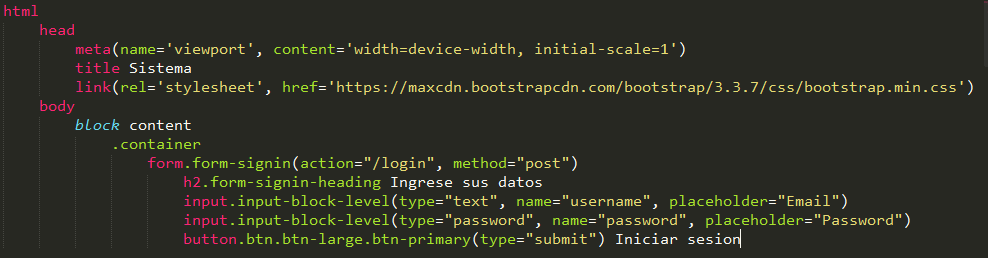
**Delete**



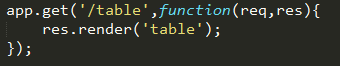
* 1. Implementaremos una vista de login, por lo que agregaremos el siguiente middleware en nuestro archivo principal.



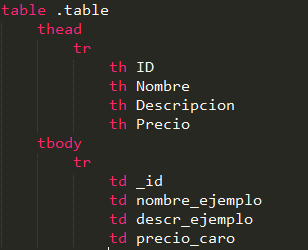
* 1. Seguidamente, en la carpeta views, crearemos el archivo login.jade con el siguiente contenido.



* 1. Antes de continuar con nuestra implementación de una validación de usuarios, haremos pruebas con jade y mongoose. Implementemos el middleware table para renderizar una tabla en bootstrap.



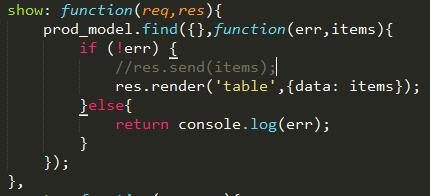
* 1. Cree la vista table.jade en la carpeta views.



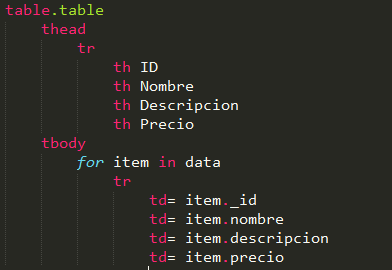
* 1. Adjunte una captura de la vista generada en el navegador.



* 1. Ahora mezclaremos lo aprendido. Modifique la función show, ubicada en prod.js, de tal forma que luzca así.

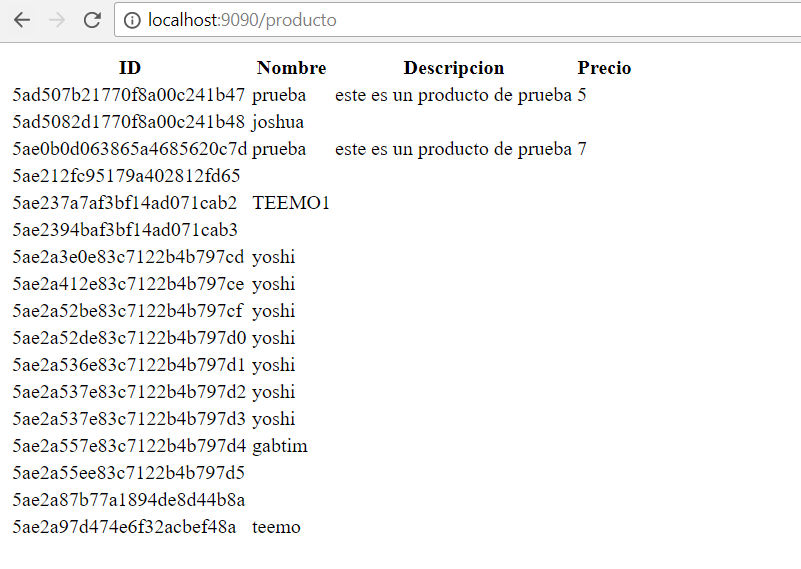


* 1. Ahora modificaremos la vista table.jade para que luzca de la siguiente manera

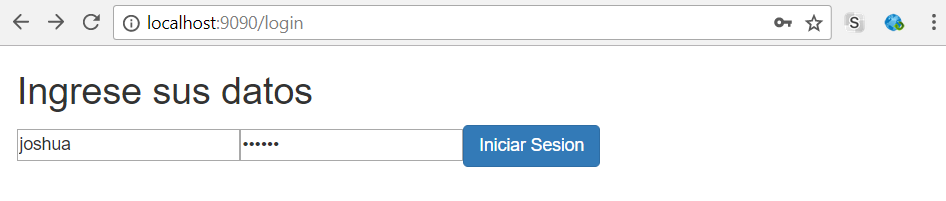


* 1. Adjunte capturas con el resultado de ingresar a la siguiente URL.

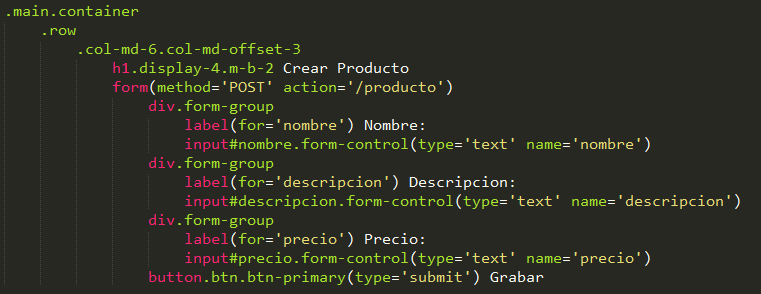




* 1. Implemente un archivo user.js en la carpeta models con toda la lógica necesaria para acceder al sistema (hacer que los usuarios validen ante esa vista login).



* 1. Implemente un CRUD para la tabla de productos. Para su facilidad, se adjunta un ejemplo de cómo luce un formulario en jade.



1. **Finalizar la sesión**
   1. Apagar el equipo virtual
   2. Apagar el equipo

**Tarea**

Implementar el CRUD de al menos tres maestros generales de su proyecto en MongoDB con vistas basadas en Jade y Bootstrap. Adjuntar capturas del código utilizado y de la estructura del proyecto.

**Conclusiones:**

Indicar las conclusiones que llegó después de los temas tratados de manera práctica en este laboratorio.

|  |
| --- |
| * Se pudo utilizar conocimientos aprendidos anteriormente para implementar el laboraotrio. * Se pudo utilizar nuevas herramientas brindadas por express. * Se pudo implementar el archivo de validación login. * Se aprendió a crear archivos en jade. * Se instalo mooongose para implementar el laboratorio. |