api rest dinámica

autor: estiven

dirección carrera 111d n12 113

Contenido

[*1 introducción* 3](#_Toc110440364)

[*2 patrón de diseño* 3](#_Toc110440365)

[*3 archivos y sus funcionamientos.* 4](#_Toc110440366)

[*4 como utilizarla* 5](#_Toc110440367)

Documentación

# 1 introducción

Este api fue diseñada para que se ajuste a cualquier base de datos ya que es dinámica, dando la posibilidad de ser reutilizada en diversos tipos de software.

# 2 patrón de diseño

Este api está enfocado en llevar las buenas prácticas de programación

Se llevo a cabo en el patrón: mvc que en sus siglas en ingles es modelo, vista , controlador.

# 3 archivos y sus funcionamientos.

3.1 Tiene una carpeta llamada config es donde se encuentra la conexión a la base de datos.

3.2 tiene una carpeta models es donde se encuentra todo lo referente a los archivos que hacen las consultas, hay 4 archivos llamados: get\_model, post\_model, put\_model y delete\_model.

3.3 tiene una carpeta controller es donde se controla las rutas, hay 5 archivos, uno controla el servicio get, el otro el servicio post, el otro el servicio put, el otro el servicio delete y el otro controlador llamado route\_controller es el que inicia todo el funcionamiento del api, llevándome a la carpeta route.

3.4 tiene una carpeta llamada route, esta carpeta tiene un archivo route que es el encargado de validar mediante cual método me solicito el cliente y que a su vez también tiene una carpeta llamada services

3.5 tiene una carpeta llamada services, esta carpeta se encuentra dentro de la carpeta routes, la carpeta services tiene 4 archivos, llamados get, post, put delete, el objetivo de estos archivos son resivir la información que envía el cliente por medio de la url.

3.6 tiene un archivo llamado htaccess, este archivo es para proteger los archivos del servidor, htaccess es un archivo que nos permite darle instrucciones a todos los servidores que funcionen con la tecnología apache.

3.7 tiene archivo índex que es el encargado de iniciar y recibir todo.

# 4 como utilizarla

Este api funciona por medio de la url, entonces como dije anteriormente es dinámica, entonces me permite hacer consultas directamente por la url.

En la carpeta config en el archivo conexión pone la base de datos donde se va conectar, luego pasa la api a un servidor o la pone en xammp en el servidor local.

Para utilizar la api vamos hacer un dominio.

Para crear el dominio local vamos a poner la siguiente dirección en el gestor de archivos de Windows:

|  |
| --- |
| C:\xampp\apache\conf\extra |

Después de poner va a ver un archivo llamado httpd-vhosts , vamos abrirlo con visual studio code o block de notas

Vamos activar lo siguiente

|  |
| --- |
| ##<VirtualHost \*:80>      ##ServerAdmin webmaster@dummy-host2.example.com      ##DocumentRoot "C:/xampp/htdocs/dummy-host2.example.com"      ##ServerName dummy-host2.example.com      ##ErrorLog "logs/dummy-host2.example.com-error.log"      ##CustomLog "logs/dummy-host2.example.com-access.log" common  ##</VirtualHost> |

Le quitamos los # para ativarlo, luego vamos a borrar todo menos las etiquetas virtualHost, también dejamos documenRoot y serverName. Debe quedar como lo siguiente

|  |
| --- |
| <VirtualHost \*:80>      DocumentRoot "C:/xampp/htdocs/apis"      ServerName api.com  </VirtualHost> |

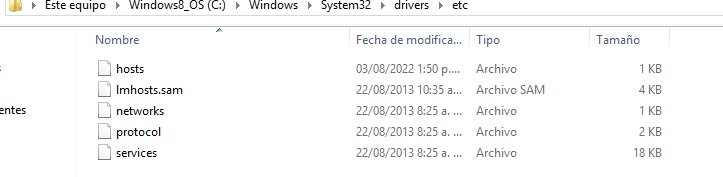
En documentRoot ponemos la ruta de la image anterior, despues ponemos el nombre de la carpeta donde esta guardado el proyecto, cambiamos el nombre donde dice api y ponemos el nombre de la carpeta.

En serveName pones como queremos que se llame nuestro hosting y ya lo podemos cerrar.

El siguiente paso es poner lo siguiente en la url de la ventana del gestor de archivo windows:

|  |
| --- |
| C:\Windows\System32\drivers\etc |

Una vez este abierta esta ruta va aparecer la siguiente archivos:



Abre la carpeta hosts, la puede abrir con un editor de código o block de notas.

Vamos a configurar lo siguiente:

# Copyright (c) 1993-2009 Microsoft Corp.

#

# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.

#

# This file contains the mappings of IP addresses to host names. Each

# entry should be kept on an individual line. The IP address should

# be placed in the first column followed by the corresponding host name.

# The IP address and the host name should be separated by at least one

# space.

#

# Additionally, comments (such as these) may be inserted on individual

# lines or following the machine name denoted by a '#' symbol.

#

# For example:

#

#      102.54.94.97     rhino.acme.com          # source server

#       38.25.63.10     x.acme.com              # x client host

# localhost name resolution is handled within DNS itself.

#   127.0.0.1       localhost

#   ::1             localhost

Cogemos el archivo donde dice 127.0.0.1 y le quitamos el símbolo # que esta al lado izquierdo para des comentarlo, y donde dice localhost le ponemos el nombre del dominio que le pusimos anteriormente, debe quedar asi:

# Copyright (c) 1993-2009 Microsoft Corp.

#

# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.

#

# This file contains the mappings of IP addresses to host names. Each

# entry should be kept on an individual line. The IP address should

# be placed in the first column followed by the corresponding host name.

# The IP address and the host name should be separated by at least one

# space.

#

# Additionally, comments (such as these) may be inserted on individual

# lines or following the machine name denoted by a '#' symbol.

#

# For example:

#

#      102.54.94.97     rhino.acme.com          # source server

#       38.25.63.10     x.acme.com              # x client host

# localhost name resolution is handled within DNS itself.

#   ::1             localhost

127.0.0.1       api.com

Apagamos el servidor y lo encendemos Y listo ya tenemos nuestro dominio local.

Para empezar a utilizar el dominio local ponemos en el navegador api.com

Nota: la configuración anterior es para decirle al navegador que no busque dominio en la red, sino que busque ese dominio en nuestro pc.

Método get

Por la url, pone lo siguiente:

|  |
| --- |
| http://prueba/productos |

En la imagen anterior muestra los parámetros, el primero es el protocolo http, después el aparece prueba que es el nombre de la carpeta que se llama prueba

Y después sigue la consulta que se llama productos, ese tercer parámetro es el que esta consultando en la base de datos.

Nota: cuando pone el nombre de la tabla es suficiente para que nos muestre todo los registros no es necesario poner \*

|  |
| --- |
| [**http://localhost:4200/productos**](http://localhost:4200/productos)**?select=id\_producto** |

En la imagen anterior después del signo de interrogación ese parámetro llamado select es una variable que almacena el id\_producto, que a su ves en la consulta es como si hubiera puesto “select id\_producto from productos”

Nota: se puede poner mas campos para mostra ejemplo: <http://localhost:4200/productos>?select=id\_producto,nombre\_producto o también se

puede poner \*

|  |
| --- |
| [**http://localhost:4200/productos**](http://localhost:4200/productos)**?select=id\_producto&linkto=nombre\_producto&equalto=piña** |

Clasula where

En la imagen anterior después del signo de aperlan & ese parámetro llamado linkto es una variable que almacena el nombre de la tabla, nombre\_producto y la variable equalto almacena el nombre del producto que quiere que consida.

Es como si en la consulta hubiera puesto “select \* from productos where nombre\_producto=piña”

|  |
| --- |
| [**http://localhost:4200/productos**](http://localhost:4200/productos)**?select=id\_producto&linkto=nombre\_producto&equalto=piña**  **&column=id\_producto&order=asc** |

Clausula order by con where

En la imagen anterior se puede ver que se aplicaron dos parámetros mas comparando la consulta anterior, los parámetros son column y order, estos dos parámetros son para ordenar la consulta, el parámetro column es para saber que columna voy a escoger para ordenarla y el parámetro order es para saber el orden si descendente o ascendente, el equivalente a hacer una consulta en sql “SELECT id\_producto FROM productos where nombre\_producto=’piña’ order by asc”;

|  |
| --- |
| [**http://localhost:4200/productos**](http://localhost:4200/productos)**?select=id\_producto&linkto=nombre\_producto&equalto=piña**  **&columna=id\_producto&order=asc&init=0&finit=5** |

Clausula order by, where y limit

En la imagen anterior ya se puede aprecias que hay dos parámetros mas comparando la consulta anterior, eso dos parámetros extras son para poder controlar la cantidad de registros que queremos ver donde una solución para poder hacer paginación, las dos variables extras son: init y finit, el init nos permite definir desde que sesión registro queremos empezar y el finit es para saber cuantos registros queremos ver.

|  |
| --- |
| http://localhost:4200/relation?rel=tabla\_productos,categoria&type=categoria\_categoria |

Relacionar tablas inner join

En la imagen anterior ya se puede aprecias que esta url es algo diferente a las anteriores esta es la forma de relacionar las tablas. Hay tres variables nuevas el relation, rel, type. El relation está encargada de hacer una validación. El rel es el encargado de relacionar las tablas. El type es el encargado de relacioname los ides de las tablas productos id\_categoria con la tabla categoría id\_categoria. Como sabemos las tablas se relacionan es por medio de la Forey key de la tabla producto con el id de la tabla categoría. entonces normalmente la Forey key se escribe igual a la tabla primary key de la tabla con la que se va relacionar. Ejp: id\_categoria, esta es la tabla categoría. id\_categoria, este id es el de la tabla productos. como puede ver en la url no tiene id\_. Esto no es necesario ponerlo ya que en una línea de código esta encargada de poner id\_ automáticamente.

Nota: también se puede usar filtros ejemplo:

|  |
| --- |
| **http://localhost:4200/relation?rel=tabla\_productos,categoria&type=categoria\_categoria& linkto=nombre\_producto&equalto=piña**  **&columna=id\_producto&order=asc&init=0&finit=5** |

Clausula like

En la imagen anterior se puede apreciar que es muy similares a las anteriores pero esta consulta tiene una nueva variable llamada Search. Esta variable es el equivalente en el sql como poner like.

|  |
| --- |
| [**http://localhost:4200/productos**](http://localhost:4200/productos)**?select=id\_producto&linkto=nombre\_producto&search=piña** |

Clausula like y where para hacer búsquedas

En la imagen anterior se puede apreciar que es muy similares a las anteriores. Como puede ver hay una variable. Search, pero la diferencia de esta es que está utilizando la cláusula where

Nota: también podemos buscar el id o cualquier atributo con la cláusula where. Para hacer eso debemos poner en linkto= nombre\_product,id\_producto en este ejemplo estoy poniendo dos tablas una para buscar con el like y el otro para buscar con el where y en el Search ponemos Search=piña\_2, así dividimos este segunco parámetro para buscar el nombre del producto y el id

|  |
| --- |
| [**http://localhost:4200/productos**](http://localhost:4200/productos)**?select=\*&linkto=nombre\_producto,id\_producto&search=piña\_2**  **&columna=id\_producto&order=asc&init=0&finit=5** |

Equivalente en la base de datos:

|  |
| --- |
| SELECT $select FROM $table WHERE nombre\_producto LIKE ‘%piña%’ $AND id\_producto = 2  order by id\_producto desc limit 0,5 |

Clausula like ,where inner join para hacer búsquedas con relaciones

en esta consulta estamos haciendo relaciones entre tablas y además una vez este relacionadas podemos hacer búsquedas con la cláusula like

|  |
| --- |
| **https://ysoft.com/relation?rel=tabla\_productos,categoria&type=categoria,categoria&linkto=nombre\_producto&search=p** |

Equivalente en la base de datos:

|  |
| --- |
| SELECT $select FROM $rel\_array[0]  $inner\_join  WHERE  $linkto\_array[0] LIKE  '%$search\_array[0]%' $AND" |

Clausule between

En esta nueva consulta hay dos variables nuevas que nos permite hacer consultas con rangos estas dos variables between1, betweem2, además de eso debemos indicarle el nombre de la tabla para poder definir el rango esa variable es linkto.

|  |
| --- |
| [**http://localhost:4200/productos**](http://localhost:4200/productos)**?linkto=id\_productos&between1=1&between2=3** |

Equivalente en la base de datos:

|  |
| --- |
| SELECT $select FROM $table WHERE $linkto BETWEEN $between1 and $between2  $filterto |

Clausule between y inner join

En esta nueva consulta vamos hacer relaciones y poner rangos:

|  |
| --- |
| [**http://localhost:4200/**](http://localhost:4200/s)**relation?rel=tabla\_productos,categoria&type=categoria,categoria linkto=id\_productos&between1=1&between2=3** |

Equivalente en la base de datos:

|  |
| --- |
| SELECT $select FROM $rel\_array[0] $inner\_join  WHERE $linkto BETWEEN $between1 and $between2 $filterto  ORDER BY $column $orde\_by limit $init , $finit" |

Método post

en el método post enviamos datos por debajo, ósea que los datos no se envían por la url, en la carpeta services el post siempre va estar validando que llegue una imagen por si la tabla que el usuario tiene una columna imagen para ser guardada en la una carpeta img, además de eso también el api está programada para resivir parámetros que son de tipo contraseñas. Los nombres que identifica el api para hacer una encriptación son los siguientes: password, clave y contrasena.

Método put

En el método put necesitamos 4 parámetros la tabla donde vamos a guardar los datos , los atributos que vamos a editar, el id de el campo que vamos a cambiar y el nombre del id que vamos a escoger.

Para hacer la consulta debemos poner los siguientes parametros:

|  |
| --- |
| [**http://localhost:4200/loguin?id=10&nameid=id\_loguin**](http://localhost:4200/loguin?id=10&nameid=id_loguin) |

Equivalente en la base de datos:

|  |
| --- |
| $query="UPDATE $table SET $set WHERE $nameid = :$nameid"; |

Nota: los parámetros no se envían por la url.

Método delete

en el metodo delete la es lo mismo que la anterior, la única diferencia es que se envia por metodo delete para eliminar

|  |
| --- |
| [**http://localhost:4200/loguin?id=10&nameid=id\_loguin**](http://localhost:4200/loguin?id=10&nameid=id_loguin) |

Equivalente en la base de datos:

|  |
| --- |
| $query="DELETE FROM $table  WHERE $nameid = :$nameid |