**Consumir servicio:**

Dentro de la base ya existe un usuario el cual es:

{

"email":"correotest@gmail.com",

"password":"Ab##hdhdssf"

}

Sin embargo, también se puede crear usuarios mediante el siguiente end-point de la api rest

Post <http://localhost:8080/api/users>

{

"email":"correoacrear@correo.com",

"password":"Ab##hdhdssf"

}

Para poder iniciar sesión y generar un token de autenticación se puede consumir el siguiente end-point y envciar el json que se muestra:

POST <http://localhost:8080/api/login>  
{

"email":"correotest@gmail.com",

"password":"Ab##hdhdssf"

}

Una vez que se consuma el end-point regresara en caso de ser correcto los usuarios lo siguiente con código http 200:   
{

"success": true,

"message": "Inicio de sesion con exito",

"jwt": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJlbWFpbCI6ImNvcnJlb3Rlc3RAZ21haWwuY29tIiwiaWRfdXNlciI6IjEiLCJpYXQiOjE3MTM5NTM4NzYsImV4cCI6MTcxMzk1NzQ3Nn0.csjRL98obpKtVafpsZ8q8cXzHyQ1Btp8EGpQwztvnRc"

}

Caso contrario nos regresara una respuesta como la siguiente, con un código http 401.

{

"success": false,

"message": "Contraseña invalida"

}

Si se desea obtener blog guardados o creados, se puede consumir el siguiente end-point, cabe mencionar que los blog viene paginados y también se puede filtrar por medio del título, autor y contenido, a continuación se muestra en el end-point:

Como parámetro se envía la página, si embargo si no se envía nada, por defecto te traerá los primero 10, también podemos enviar el content, autor y title como se muestra a continuación.

GET http://localhost:8080/api/blog?page=1&autor=valente

GET <http://localhost:8080/api/blog?page=1&content=codeginiter>

GET <http://localhost:8080/api/blog?page=1&title=codeginiter>

Para poder consumer el end-pint anterior, se necesita esta logeado, lo cual se requiere de un token, este token valida que seamos la persona que decimos ser, el token se enviar por cabeceras mediante bearer: eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJlbWFpbCI6ImNvcnJlb3Rlc3RAZ21haWwuY29tIiwiaWRfdXNlciI6IjEiLCJpYXQiOjE3MTM5NTQzMzcsImV4cCI6MTcxMzk1NzkzN30.ZZLcGQ-2XF28ULB2YJgHPEuvCefO\_kgExK4E4UH73Ac

Si el token es correcto o aun esta vigente, retorna los datos consultados como se muestran a continuación:

{

"success": true,

"message": "Blogs existentes",

"data": [

{

"id\_blog": "3",

"title": "title",

"autor": "autor",

"content": "content",

"publication\_date": "2024-04-24 01:34:36"

},

{

"id\_blog": "4",

"title": "title",

"autor": "autor",

"content": "content",

"publication\_date": "2024-04-24 01:34:36"

},

{

"id\_blog": "5",

"title": "title",

"autor": "autor",

"content": "content",

"publication\_date": "2024-04-24 01:34:36"

},

{

"id\_blog": "6",

"title": "title",

"autor": "autor",

"content": "content",

"publication\_date": "2024-04-24 01:34:36"

},

{

"id\_blog": "7",

"title": "title",

"autor": "autor",

"content": "content",

"publication\_date": "2024-04-24 01:34:36"

},

{

"id\_blog": "8",

"title": "title",

"autor": "valente",

"content": "content",

"publication\_date": "2024-04-24 01:34:36"

},

{

"id\_blog": "9",

"title": "title",

"autor": "autor",

"content": "codeginiter",

"publication\_date": "2024-04-24 01:34:36"

},

{

"id\_blog": "12",

"title": "title",

"autor": "autor",

"content": "content",

"publication\_date": "2024-04-24 01:34:36"

},

{

"id\_blog": "13",

"title": "title",

"autor": "autor",

"content": "content",

"publication\_date": "2024-04-24 01:34:36"

},

{

"id\_blog": "14",

"title": "title",

"autor": "autor",

"content": "content",

"publication\_date": "2024-04-24 01:34:36"

}

],

"pagination": {

"total\_rows": 10,

"per\_page": 10,

"current\_page": 1,

"total\_pages": 1,

"previous\_page": 1,

"next\_page": 1

}

}

Caso contrario: de que el token este corrupto, invalido o caduco no retorna lo siguiente con un código htto 401:

{

"success": false,

"message": "Usuario invalido, sin autorización"

}

Para obtener un solo blog(post) podemos consumir el siguiente end-point: donde el numero 4 es el id del blog(post)

GET <http://localhost:8080/api/blog/4>

Como respuesta nos regresa el blog(post) como se muestra a continuación

{

"success": true,

"message": "Blog encontrado",

"data": {

"id\_blog": "4",

"title": "title",

"autor": "autor",

"content": "content",

"deleted": "0",

"publication\_date": "2024-04-24 01:34:36",

"updated\_at": "2024-04-24 02:20:11",

"id\_user": "1"

}

}

Si no existe ese blog, retorna lo siguiente con un código http 404:

{

"success": false,

"message": "El blog no existe"

}

Para eliminar un blog(post) se puede consumir el siguiente end-point y enviar el token por cabeceras como se ha mencionado ,   
DEELETE <http://localhost:8080/api/blog/10>

Nos regresa la siguiente respuesta con un código http 200:

{

"success": false,

"message": "Se elimino con exito el blog"

}

Para crear blog, enviamos los datos, tales como, title, content y autor, estos son obligatorios, el cual podemos guardar mediante el end-point siguiente:

POST <http://localhost:8080/api/blog>  
como cuerpo le enviamos lo siguiente:  
{

"title":"title",

"autor":"autor",

"content":"content"

}

Si todo sale bien, nos regresara lo siguiente con un código http 201:

{

"success": true,

"message": "Blog creado con exito"

}

Para actualizar un blog(post) podemos consumir el siguiente end-point, como se ha mencionado también se debe enviar el token por cabeceras.

PUT <http://localhost:8080/api/blog/10>  
el numero 10 es el id del blog(post) y como cuerpo se envía lo siguiente

{

"title":"title",

"autor":"autor",

"content":"contenwwwwwwwwwwt"

}