|  |  |
| --- | --- |
| Códigos de error o respuesta | |
| No se proporcionó el token de acceso | 101 |
| Error con los datos enviados (incorrectos/incompletos) | 102 |
| Los datos no coinciden con los de la base de datos | 103 |
| Se realizó correctamente la solicitud | 200 |
| Error al procesar la solicitud en la base de datos | 104 |
| Token invalido | 105 |
| No tienes permisos para acceder a esa ruta | 401 |
| Token expirado | 106 |
| datos ya se encuentran registrados en la base de datos | 107 |
| Se realizó correctamente la solicitud pero no hay información para devolver | 204 |
| Se envio correctamente código de verificación al correo electronico | 108 |

|  |  |  |  |
| --- | --- | --- | --- |
| METODO | DESCRIPCION | URL | EJEMPLO |
|  |  |  |  |
|
| POST | Registro de usuarios | http://127.0.0.1:3000/api/register | datos={  Nom\_User: "user",  Ape\_User: "apeUser",  Ema\_User: "user01@gmail.com",  Pass\_User: "1234567890",  Dir\_Ip: "192.168.0.X"  }  fetch('http://127.0.0.1:3000/api/register’, {  method: "POST",  body: JSON.stringify(datos),  headers: { "Content-type": "application/json; charset=UTF-8" }  }).then(data=>data.json())  .then(data => console.log(data)) |
| Verificar Email mediante el código de verificación que se genera al registrar al usuario. | http://127.0.0.1:3000/api/emailvalidate | datos={  Id\_User: 30,  codigo: "22d380b1-34b7-439f-bd22-3b49b80d92df"  }  fetch('http://127.0.0.1:3000/api/emailvalidate, {  method: "POST",  body: JSON.stringify(datos),  headers: { "Content-type": "application/json; charset=UTF-8" }  }).then(data=>data.json())  .then(data => console.log(data)) |
| Iniciar sesion | http://127.0.0.1:3000/api/login/ | datos={  Nom\_User: “user01” o Ema\_User: user01@example.com ,  PassUser: "1234567890",  Dir\_Ip: “192.168.x.x”  }  fetch('http://127.0.0.1:3000/api/login, {  method: "POST",  body: JSON.stringify(datos),  headers: { "Content-type": "application/json; charset=UTF-8" }  }).then(data=>data.json())  .then(data => console.log(data)) |
| Validar Codigo si la Ip Es nueva al iniciar sesion | http://127.0.0.1:3000/api/codevalidate | datos={  Id\_User: 1,  Codigo:”xxx-xxx-xxx-xxx”,  Dir\_Ip: “192.168.x.x”  }  fetch('http://127.0.0.1:3000/api/login, {  method: "POST",  body: JSON.stringify(datos),  headers: { "Content-type": "application/json; charset=UTF-8" }  }).then(data=>data.json())  .then(data => console.log(data)) |
| GET | Obtener lista de todos los usuarios (Rol Admin) | http://127.0.0.1:3000/api/users | Const token = 'token de sesion'  fetch('http://127.0.0.1:3000/api/users, {  method: "GET",  headers:  {  "Content-type": "application/json;  charset=UTF-8";  'Authorization': “Bearer” + token  }  }).then(data=>data.json())  .then(data => console.log(data)) |
|  |  |  |

**CURSOS**

|  |  |  |  |
| --- | --- | --- | --- |
| METODO | DESCRIPCION | URL | EJEMPLO |
|  |  |  |  |
|
| GET | Obtener lista de cursos publicados | http://127.0.0.1:3000/api/cursos | fetch('http://127.0.0.1:3000/api/cursos)  .then(data=>data.json())  .then(data => console.log(data)) |
| Obtener lista de cursos publicados por categoría | http://127.0.0.1:3000/api/cursos/categoria/:id | fetch('http://127.0.0.1:3000/api/cursos/5)  .then(data=>data.json())  .then(data => console.log(data)) |
| POST | Crear nuevo curso  (Rol Admin) | http://127.0.0.1:3000/api/cursos/new | datos={  Id\_User: 1,  Nom\_Cur: "Curso de Programación Avanzada",  Des\_Cur: "Este curso cubre temas avanzados de programación en varios lenguajes.",  Hor\_Cont\_Total: 40,  Fech\_Crea\_Cur: "2024-02-05",  Id\_Cat\_FK: 1,  "Fot\_Cur": "https://ejemplo.com/imagen\_curso.jpg"  }  Const token = “token de sesion”  fetch('http://127.0.0.1:3000/api/cursos/new, {  method: "POST",  body: JSON.stringify(datos),  headers:  {  "Content-type": "application/json;  charset=UTF-8";  'Authorization': “Bearer” + token  }  }).then(data=>data.json())  .then(data => console.log(data)) |
| PUT | Actualizar datos del Curso.  (Rol Admin/Instructor)  Nota: No es necesario enviar todos los datos del curso para actualizar, el campo enviado será actualizado, sin afectar a los otros. | http://127.0.0.1:3000/api/cursos/update/:id | datos={  Nom\_Cur: "Curso de Programación Avanzada",  Des\_Cur: "Este curso cubre temas avanzados de programación en varios lenguajes.",  Hor\_Cont\_Total: 40,  Fech\_Crea\_Cur: "2024-02-05",  Id\_Cat\_FK: 1,  Fot\_Cur: <https://ejemplo.com/imagen_curso.jpg>  }  Const token = “token de sesion”  fetch('http://127.0.0.1:3000/api/cursos/update/1, {  method: "PUT",  body: JSON.stringify(datos),  headers:  {  "Content-type": "application/json;  charset=UTF-8";  'Authorization': “Bearer” + token  }  }).then(data=>data.json())  .then(data => console.log(data)) |
| **Eliminar curso**  **(Rol Admin/Instructor)**  Nota: El curso no se elimina, lo que se hace es actualizar el estado del curso (Est\_Cur) a el valor 3, el cual hace referencia a que esta eliminado | http://127.0.0.1:3000/api/cursos/update/:id | Let datos={  Est\_Cur: 3  }  Const token = “token de sesion”  fetch('http://127.0.0.1:3000/api/cursos/update/1, {  method: "PUT",  body: JSON.stringify(datos),  headers:  {  "Content-type": "application/json;  charset=UTF-8";  'Authorization': “Bearer” + token  }  }).then(data=>data.json())  .then(data => console.log(data)) |

**CATEGORIAS**

|  |  |  |  |
| --- | --- | --- | --- |
| METODO | DESCRIPCION | URL | EJEMPLO |
|  |  |  |  |
|
| GET | Obtener lista de categorias | http://127.0.0.1:3000/api/categories | fetch('http://127.0.0.1:3000/api/cursos)  .then(data=>data.json())  .then(data => console.log(data)) |
| Obtener categoria por Id | http://127.0.0.1:3000/api/categories/:id | fetch('http://127.0.0.1:3000/api/cursos/5)  .then(data=>data.json())  .then(data => console.log(data)) |
| POST | Crear nueva categoria  (Rol Admin) | http://127.0.0.1:3000/api/categories/new | datos={  Nom\_Cat: "Programacion",  }  Const token = “token de sesion”  fetch('http://127.0.0.1:3000/api/cursos/new, {  method: "POST",  body: JSON.stringify(datos),  headers:  {  "Content-type": "application/json;  charset=UTF-8";  'Authorization': “Bearer” + token  }  }).then(data=>data.json())  .then(data => console.log(data)) |
| PUT | Actualizar categoría  (Rol Admin) | http://127.0.0.1:3000/api/categories/update/:id | datos={  Nom\_Cat: "Programacion",  }  Const token = “token de sesion”  fetch('http://127.0.0.1:3000/api/cursos/update/1, {  method: "PUT",  body: JSON.stringify(datos),  headers:  {  "Content-type": "application/json;  charset=UTF-8";  'Authorization': “Bearer” + token  }  }).then(data=>data.json())  .then(data => console.log(data)) |
| **Eliminar categoría**  **(Rol Admin)** | http://127.0.0.1:3000/api/categories/delete/:id | Let datos={  Nom\_Cat: 3  }  Const token = “token de sesion”  fetch('http://127.0.0.1:3000/api/cursos/update/1, {  method: "PUT",  body: JSON.stringify(datos),  headers:  {  "Content-type": "application/json;  charset=UTF-8";  'Authorization': “Bearer” + token  }  }).then(data=>data.json())  .then(data => console.log(data)) |