

Controllers\Previa.controller.js

```
1 // Importamos el objeto 'db' que contiene los modelos de la base de datos
2 const db = require('../Models');
3
4 // Definimos el método 'lista' que se encargará de obtener todas las instancias de '
Previa'
5 exports.lista = (req, res, next) =>{
6   // Utilizamos el método 'findAll' de Sequelize para obtener todas las instancias de '
Previa'
7   db.Previa.findAll()
8   .then(previas => {
9     // Si la operación es exitosa, enviamos las instancias obtenidas como respuesta en
formato JSON
10     res.json(previas);
11   }).catch(err => {
12     next(err); // Pasamos el error al middleware de manejo de errores
13   });
14 }
15
16 // Definimos el método 'filtrar' que se encargará de obtener las instancias de 'Previa'
que coincidan con un valor específico en un campo específico
17 exports.filtrar = (req, res, next) =>{
18   // Obtenemos el campo y el valor de los parámetros de la ruta de la solicitud HTTP
19   const campo = req.params.campo;
20   const valor = req.params.valor;
21   // Utilizamos el método 'findAll' de Sequelize para obtener las instancias de 'Previa'
que tienen ese valor en ese campo
22   db.Previa.findAll({
23     where: {
24       [campo]: valor
25     }
26   })
27   .then(previas => {
28     // Si la operación es exitosa, enviamos las instancias obtenidas como respuesta en
formato JSON
29     res.json(previas);
30   }).catch(err => {
31     next(err); // Pasamos el error al middleware de manejo de errores
32   });
33 }
34
35 // Definimos el método 'nuevo' que se encargará de crear una nueva instancia de 'Previa'
36 exports.nuevo = (req, res, next) =>{
37   // Verificamos que los datos necesarios estén presentes en el cuerpo de la solicitud
HTTP
38   if(!req.body.dni_alumno || !req.body.id_condicion || !req.body.id_materia){
39     // Si faltan datos, enviamos un mensaje de error con un código de estado HTTP 400
40     res.status(400).send({
41       message: "Faltan datos"
42     });
43     return;
44   }
45   // Creamos un objeto con los datos de la nueva instancia
46   const previa = {
47     dni_alumno: req.body.dni_alumno,
48     id_condicion: req.body.id_condicion,
49     id_materia: req.body.id_materia,
50     aprobado: req.body.aprobado ? req.body.aprobado : false
51   }
```

```

52 // Utilizamos el método 'create' de Sequelize para crear la nueva instancia en la base
de datos
53 db.Previa.create(previa)
54 .then(data => {
55 // Si la operación es exitosa, enviamos la nueva instancia como respuesta en
formato JSON
56 res.json(data);
57 }).catch(err => {
58 next(err); // Pasamos el error al middleware de manejo de errores
59 });
60 }
61
62 // Definimos el método 'actualizar' que se encargará de actualizar una instancia existente
de 'Previa'
63 exports.actualizar = (req, res, next) =>{
64 // Obtenemos el ID de los parámetros de la ruta de la solicitud HTTP
65 const id = req.params.id;
66 // Utilizamos el método 'update' de Sequelize para actualizar la instancia en la base
de datos
67 db.Previa.update(req.body,{
68 where: {id_previa: id}
69 })
70 .then(num => {
71 // Si la operación es exitosa y se actualizó una instancia, enviamos un mensaje de
éxito
72 if(num == 1){
73 res.send({
74 message: "previa actualizado"
75 });
76 }else{
77 // Si no se pudo actualizar la instancia, enviamos un mensaje de error
78 res.send({
79 message: "No se pudo actualizar la previa"
80 });
81 }
82 }).catch(err => {
83 next(err); // Pasamos el error al middleware de manejo de errores
84 });
85 }
86
87 // Definimos el método 'eliminar' que se encargará de eliminar una instancia existente de
'Previa'
88 exports.eliminar = (req, res, next) =>{
89 // Obtenemos el ID de los parámetros de la ruta de la solicitud HTTP
90 const id = req.params.id;
91 // Imprimimos un mensaje en la consola indicando que se intentará eliminar la
instancia con ese ID
92 console.log(`Eliminar previa con id: ${id}`);
93 // Utilizamos el método 'destroy' de Sequelize para eliminar la instancia en la base
de datos
94 db.Previa.destroy({
95 where: {id_previa: id}
96 })
97 .then(num => {
98 // Si la operación es exitosa y se eliminó una instancia, enviamos un mensaje de
éxito
99 if(num == 1){
100 res.send({
101 message: "previa eliminado"
102 });
103 }else{
104 // Si no se pudo eliminar la instancia, enviamos un mensaje de error

```

```
105         res.send({
106             message: "No se pudo eliminar la previa"
107         });
108     }
109 }).catch(err => {
110     next(err); // Pasamos el error al middleware de manejo de errores
111 });
112 }
```