

Alumno: Aureliano Bernal Melgarejo - 4675680

https://github.com/AureBernal/study_app_proyect-.git

AUTHORIZATION-PROTECCION DE RUTAS

Middleware

```
const { sequelize } = require("../connection");
const jwt = require("jsonwebtoken");

const auth = async function (req, res, next) {
  if (!req.headers.authorization) {
    res.json({
      success: false,
      error: "No authorization header",
    });
    return;
  } else {
    let token = req.headers.authorization;
    const userDB = await sequelize.query(
      "SELECT * FROM users WHERE token = '" + token + "'"
    );
    let user = null;
    if (userDB.length > 0 && userDB[0].length > 0) {
      user = userDB[0][0];
      console.log("Token del usuario: ", user);
      res.locals.userId = user.id;
      next();
    } else {
      res.json({
        success: false,
        error: "Token inválido",
      });
    }
  }
};

module.exports = {
  auth,
};
```

User

```
Backend > src > route > users > JS users.route.js > <unknown> > exports
1  const userController = require('../../controller/users/users.controller');
2  const Middleware = require('../../middleware/auth.controller');
3
4  module.exports = function(app) {
5
6
7      app.delete("/users/delete/:id",Middleware.auth, userController.eliminar);
8      app.post("/users/login/", userController.login);
9      app.post("/users/logout/",Middleware.auth, userController.logout);
10     app.get("/users/list",Middleware.auth, userController.listar);
11     app.get("/users/:id", Middleware.auth,userController.consultarPorCodigo);
12     app.post("/users/update",Middleware.auth, userController.actualizar);
13
14 }
```

Topics

```
Backend > src > route > topics > JS topics.route.js > <unknown> > exports
1  const Middleware = require('../../middleware/auth.controller');
2  const topicsController = require('../../controller/topics/topics.controller');
3  module.exports = function(app) {
4
5
6      app.delete("/topics/delete/:id",Middleware.auth, topicsController.eliminar);
7      app.get("/topics/:id",Middleware.auth, topicsController.consultarPorCodigo);
8      app.get("/topics/list",Middleware.auth, topicsController.listar);
9      app.post("/topics/update",Middleware.auth, topicsController.actualizar);
10
11 }
```

BACKEND - topics

Model

```
const {DataTypes} = require('sequelize');
const {sequelize}= require('../connection');
const TopicsModel = sequelize.define('Topics', {
  id: {
    type: DataTypes.INTEGER,
    primaryKey: true,
    autoIncrement: true,
  },
  create_date: {
    type: DataTypes.DATE,
    allowNull: false,
  },
  name: {
    type: DataTypes.STRING,
    allowNull: false,
  },
  topic_id: {
    type: DataTypes.INTEGER,
    allowNull: true,
  },
  order: {
    type: DataTypes.INTEGER,
    allowNull: false,
  },
  priority: {
    type: DataTypes.INTEGER,
    allowNull: false,
  },
  color: {
    type: DataTypes.STRING,
    allowNull: false,
  },
  owner_user_id: {
    type: DataTypes.INTEGER,
    allowNull: false,
  },
},{
  tableName: 'topics',
  timestamps: false
});

module.exports={
  TopicsModel
}
```

CONTROLLER

Listar

```
backend > src > controller > topics > topics.controller.js > ...
1  const {sequelize} = require("../connection");
2  const {TopicsModel}= require("../model/topic.model");
3  const TopicService= require("../service/topic.service")
4  const listar = async function(req, res) {
5      console.log("listar topics");
6      try {
7          const topics = await TopicService.listar(req.query.filtro || '');
8          // EN topics[0] SE ENCUENTRA NUESTRO LISTADO DE SQL
9          if(topics && topics[0]){
10             res.json({
11                 success:true,
12                 topics: topics[0]
13             });
14         }else{
15             res.json({
16                 success:true,
17                 topics: []
18             });
19         }
20     } catch (error) {
21         res.json({
22             success:false,
23             error: error.message
24         });
25     }
26 };
27 const mostrarPorFiltro = async function(req, res) {
```

Consultar

```
const consultarPorCodigo = async function(req, res) {
  console.log("consultar Topics por codigo");
  try {
    const topics = await TopicService.consultarPorCodigo(req.params.id);

    // EN topics[0] SE ENCUENTRA NUESTRO LISTADO DE SQL
    if(topics){
      res.json({
        success:true,
        topics: topics
      });
    }else{
      res.json({
        success:true,
        topics: []
      });
    }
  } catch (error) {
    console.log(error);
    res.json({
      success:false,
      error: error.message
    });
  }
};
```

Actualizar y eliminar

```
const actualizar = async function(req, res) {
  let topicsRetorno=null; //GUARDARA EL TOPICO QUE SE VA A INCLUIR O EDITAR
  const data =req.body; // SE OBTIENE LOS DATOS DEL CUERPO DE LA PETICION
  try {
    topicsRetorno = await TopicService.actualizar(data);
    res.json({
      success: true,
      topics: topicsRetorno
    });
  } catch (error) {
    res.json({
      success:false,
      error: error.message
    });
  }
};

const eliminar = async function( req, res) {
  console. log( "eliminar topics" ) ;
  try {
    await TopicService.eliminar(req.params.id)
    res.json({
      success: true
    });
  } catch (error) {
    res.json({
      success: false,
      error: error.message
    });
  }
};

module.exports = {
  listar,
  actualizar,
  eliminar,
  consultarPorCodigo
};
```

SERVICE - topics

Listar

```
const listar = async function (textoBuscar) {  
  console.log("listar topics ACA");  
  try {  
    const topics = await sequelize.query(`SELECT topics.*, users.name as Owner  
      FROM topics  
      left join users on users.id=topics.owner_user_id  
      WHERE  
      UPPER(topics.name) LIKE UPPER('%${textoBuscar}%')  
      ORDER BY topics.id`);  
    // EN topics[0] SE ENCUENTRA NUESTRO LISTADO DE SQL  
    if (topics && topics[0]) {  
      return topics;  
    } else {  
      return [];  
    }  
  } catch (error) {  
    console.log(error);  
    throw error;  
  }  
};
```

Consultar

```
5  const consultarPorCodigo = async function (id) {
6    console.log("consultar topics por codigo");
7    try {
8      const topics = await TopicsModel.findByPk(id);
9
10     // EN topics SE ENCUENTRA NUESTRO topics BUSCADO
11     if (topics) {
12       return topics;
13     } else {
14       return [];
15     }
16   } catch (error) {
17     console.log(error);
18     throw error;
19   }
20 };
21
```

Actualizar y Eliminar

```
22  const actualizar = async function (data) {
23    console.log("actualizar topics");
24    let topicsRetorno = null; //GUARDARA EL TEMA QUE SE VA A INCLUIR O EDITAR
25    const id = data.id; // ID PASADO
26
27    let topicExiste = null;
28
29    try {
30      if (id) {
31        topicExiste = await TopicsModel.findByPk(id);
32      }
33      if (topicExiste) {
34        topicsRetorno = await TopicsModel.update(data, {where: {id: id}});
35      } else {
36        topicsRetorno = await TopicsModel.create(data);
37      }
38      return topicsRetorno;
39    } catch (error) {
40      throw error;
41    }
42  };
43
44  const eliminar = async function (id) {
45    console.log("eliminar topics ");
46    try {
47      const topics = await TopicsModel.findByPk(id);
48      if (topics) {
49        await TopicsModel.destroy({
50          where: { id: id },
51        });
52        return true;
53      } else {
54        return false;
55      }
56    } catch (error) {
57      throw error;
58    }
59  };
60
```

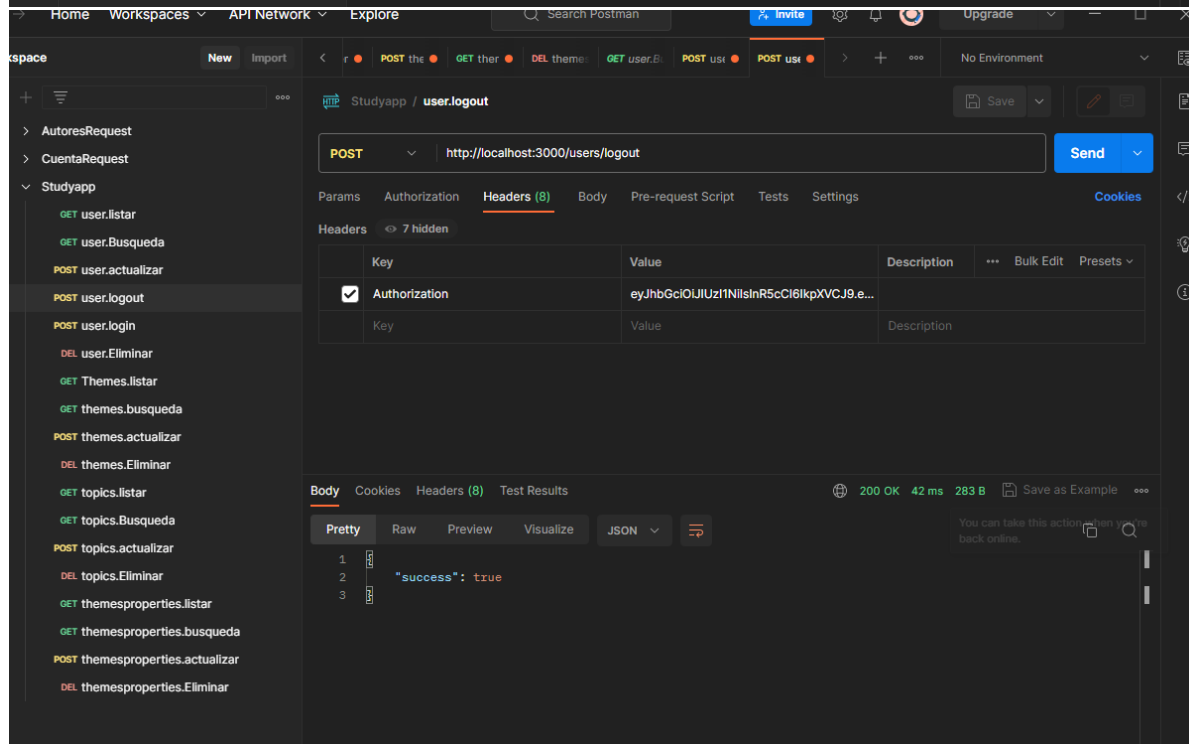
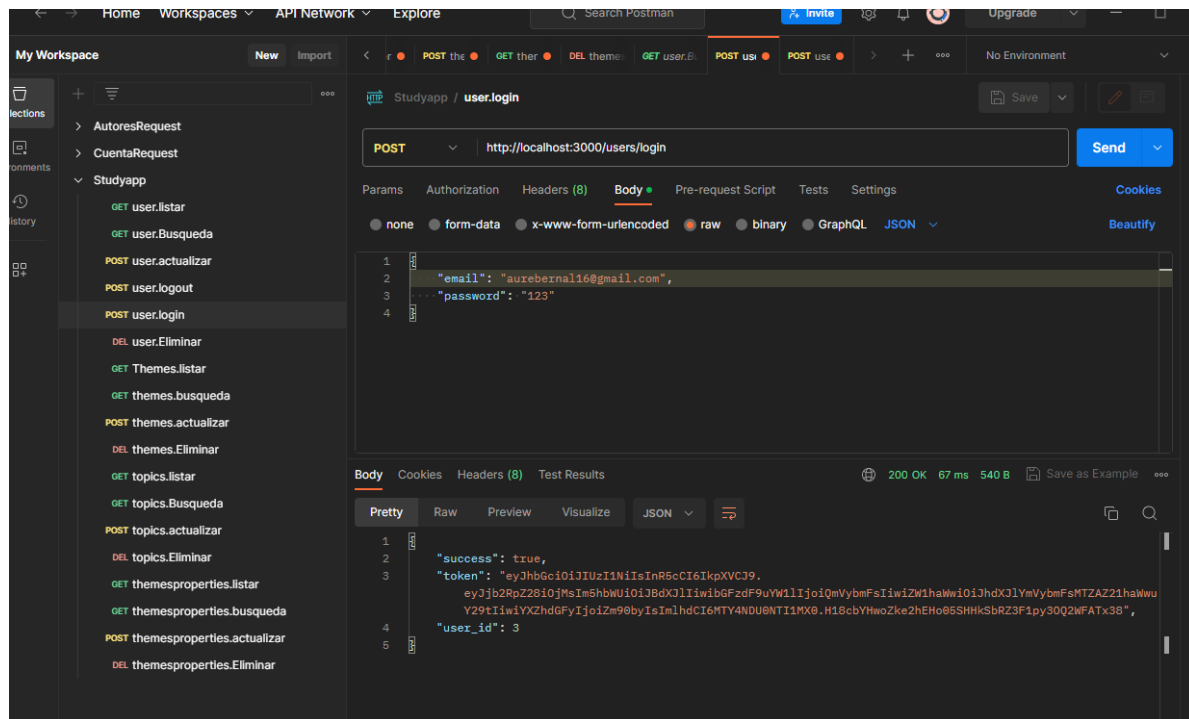

LOGIN EN BACKEND:

```
const login = async function( req, res) {
  console.log ("login usuarios");
  try {
    // Buscar en la base de datos el usuario con el correo electrónico y contraseña
    const usersDB = await sequelize.query("SELECT * FROM users WHERE email = '"+ req.body.email + "' AND password = '"+ req.body.password + "'");
    console.log ("users",usersDB);
    let user = null;
    // Verificar si se encontraron resultados en la consulta a la base de datos y asignar el primer resultado
    if (usersDB.length > 0 && usersDB[0].length > 0) {
      user = usersDB[0][0];
      // Asignar el primer registro encontrado a la variable "user"
      if(user.token){
        res.json({
          success: false,
          error: 'usuario ya autenticado'
        });
      }
      let token= jwt.sign({
        codigo: user.id,
        name: user.name,
        last_name: user.last_name,
        email:user.email,
        avatar: user.avatar
      },'passwd');
      const usersDBUpdate = await sequelize.query("UPDATE users set token= '"+ token + "' WHERE id = "+user.id)
      //si encuentra su cuarto en la base de datos se mostrara true y usuario.
      res.json({
        success: true,
        token:token,
        user_id: user.id
      });
    } else{
      res.json({
        success: false,
        error: 'usuario no encontrado'
      });
    }
  }
  // Si no se encuentra el usuario en la base de datos,
}
```

LOGOUT


```
36
37   }catch (error) {
38     // Si ocurre un error, devolver
39     console.log(error);
40     res.json ({
41       success: false,
42       error: error .message
43     });
44   }
45
46 };
47 const logout = async function( req, res) {
48
49   try {
50     const userDb= await sequelize.query("UPDATE users set token=null where id= " +res.locals.userId+ "");
51     res.json({
52       success:true
53     })
54   } catch (error) {
55     res.json({
56       success:false,
57       error: error.message
58     })
59   }
60
61 };
62 module.exports = {
63   listar, actualizar, eliminar,consultarPorCodigo,login,logout
64 };
```

LOGIN Y LOGOUT EN POSTMAN



LOGIN FRONT FUNCIONAL

Aureliano Bernal
Login



Email aurebernal16@gmail.com

Contraseña 123

