|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **NOMBRE Y APELLIDOS:** | | | **FECHA: 28/09/2023** | | |
| **DOCENTE: MANUEL MACÍAS PÉREZ** | | | **NOTA:** | | |
| **(IFCD0210) DESARROLLO DE APLICACIONES CON TECNOLOGÍAS WEB** | | | **Nº CURSO: 22-35/008902** | | |
| MF: | 0492 | UNIDADES DE APRENDIZAJE A LAS QUE RESPONDE: | UA1 | Duración: | 3 hrs. |
| UF: | 1846 |
| PRÁCTICA Nº: | Actividad 1 |
| DENOMINACIÓN: **API-REST** | | | | | |
| **DESCRIPCIÓN**  1.- Se propone la creación de un servidor para trabajar con bases de datos SQL. Adjuntar Códigos  Crear Modelos, Controller y Routes de:   * Cursos * Usuarios * Mensajes * Tareas   La práctica se realizará de manera individual.  **MEDIOS PARA SU REALIZACIÓN**   * Equipo informático. * Aplicación Visual Code Studio instalada en el equipo. * Navegadores actualizados   **Modelo mensajes**  import {DataTypes} from 'sequelize'  import db from '../config/connectdb'  const Mensaje = db.define('Mensaje',{      nombre: {          type: DataTypes.STRING,      },      email: {          type: DataTypes.STRING,      },      telefono: {          type: DataTypes.STRING,      },      asunto: {          type: DataTypes.STRING,      },      textomensaje: {          type: DataTypes.STRING,      },  },{      createdAt: false,      updatedAt: false  });  export default Mensaje  **Controller Mensajes**  import {Request, Response} from 'express';  import Mensaje from '../models/mensajeModels';  export const getMensajes = async (req: Request, res: Response) =>{      const listMensaje = await Mensaje.findAll();      res.json(listMensaje);  }//Método GETMensajes  export const getMensaje = async (req: Request, res: Response) =>{      const {id} = req.params;      const mensaje = await Mensaje.findByPk(id);      if(mensaje){          res.json(mensaje)      }      else{          res.status(404).json()          msg: `No existe un mensaje con el id: &{id}`      }  }//Método GETMensaje  export const deleteMensaje = async (req: Request, res: Response) =>{      const {id} = req.params;      const mensaje = await Mensaje.findByPk(id);      if(!mensaje){          res.status(404).json({          msg: `No existe un mensaje con el id: &{id}`          })      }      else{          await mensaje.destroy();          res.json({              msg: 'El mensaje fue eliminado con éxito'          })      }  }//Método DELETE  //Crear nuevo registro  export const postMensaje = async (req: Request, res: Response) =>{      const {body} = req;      try {          await Mensaje.create(body);          res.json({              msg: 'El mensaje fue agregado con éxito!'          })      } catch (error) {          console.log(error);          res.json({              msg:'Ha ocurrido un error'          })      }  }  export const updateMensaje = async (req: Request, res: Response) =>{      const {body} = req;      const {id} = req.params;      try {          const mensaje = await Mensaje.findByPk(id);          if(mensaje){              await mensaje.update(body);              res.json({                  msg: 'El mensaje se actualizó con éxito'              })          }          else{              res.status(404).json()              msg: `No existe un mensaje con el id: &{id}`          }      } catch (error) {          console.log(error);          res.json({              msg:'Ha ocurrido un error'          })      }  }  **Router Mensajes**  import { Router } from 'express'  import {deleteMensaje, getMensaje, getMensajes, postMensaje, updateMensaje  } from '../controllers/MensajeController'  const router = Router()  router.get('/',getMensajes);  router.get('/:id',getMensaje);  router.delete('/:id',deleteMensaje);  router.post('/',postMensaje);  router.put('/:id',updateMensaje);  export default router; | | | | | |