import mongoose from "mongoose";

const UserSchema = new Schema({

    name : {

      type:String,

      required : true

  },

  email:

  {

    type:String,

    required:true,

    unique: true

  },

  password:

  {

    type:String,

    required:true

  },

  date:

  {

    type:Date,

    default:Date.now

  }

}

)

module.exports = mongoose.model('user', UserSchema)

{

        "age": 30,

        "name": "Vijay",

        "fees": {

        "$numberDecimal": "12000"

        }}

userController.js

import userModel from "../model/user.js";

import {body, validationResult} from "express-validator"

import bcrypt from "bcrypt";

import jwt from "jsonwebtoken";

class userController

{

//    static getAllStudents = async (req, res)=>

// {

//   try{

//       const result = await  studentModel.find();

//       res.send(result);

//   }

//   catch(error)

//   {

//     res.send(error);

//   }

//   // res.send("Get All Students");

// }

// static getStudentById = async (req, res)=>

// {

// try{

//   const result =  await studentModel.findById(req.params.id);

//   res.send(result);

// }

// catch(error)

// {

// res.send(error)

// }

// }

static userRegistration = async(req, res)=>

{

  const key ="thisissecertkey";

  //  res.render("index");

  console.log("Inside insert")

  // try{

  const {name , password ,email, date } =  req.body;

 console.log(email)

  const user = await userModel.findOne({email: email})

  console.log(user)

  if(user)

  {

    res.send({"status":"failed","message":"Email already exists"})

  }

    else if(name && email && password && date)

      {

        const salt = await bcrypt.genSalt(10);

        const secPassword = await bcrypt.hash(req.body.password, salt)

        const document =new userModel(

          {

            name : name,

            email: email,

            password: secPassword,

            date : date

          }

        )

        const data = {

          user :{

            email:document.email

          }

        }

        const token = jwt.sign(data , key,

        process.env.key, {expiresIn :'5d'})

        console.log(token);

        console.log(document)

        const result = await document.save();

        res.status(201).send({"status":"success","message":"Reg success", "token":token})

       }

    else

    {

      res.send({"status":"failed","message":"All fiels already exists"})

  }

  // }

  // catch(error)

  // {

  //   console.log("THER II S S S S WRROE")

  //   res.send(error)

  // }

}

// static editStudent = async (req, res)=>

// {

//   try{

//  const result = await studentModel.findByIdAndUpdate(req.params.id, req.body);

//  res.send(result);

//   }

// catch(error)

// {

//   res.send(error);

// }

// }

// static deleteStudent = async (req, res)=>

// {

//   try{

//          const result  = await studentModel.findByIdAndDelete(req.params.id);

//          res.status(204).send(result);

//   }

//   catch(error)

//   {

//     res.send(error)

//   }

// }

static userLogin = async (req,res)=>

{

  try{

 const {email,password} = req.body;

 if(email && password)

 {

   const user = userModel.findOne({email:email})

    if(user!=null)

{

  const isMatch = await bcrypt.compare(password, user.password)

  if((user.email ===email) && isMatch)

  {

res.send("Success");

  }

  else

  {

    res.send("do not match")

  }

}

else

{

  res.send("Not reg")

}

  }

 else

 {

   res.send("all fiels req")

 }

  }

  catch(error)

  {

    res.send(error)

  }

}

}

export default userController;

app.js

import dotenv from 'dotenv'

dotenv.config()

Middleware

import jwt from "jsonwebtoken"

import userModel from '../model/user.js';

var chekUser = async(req,res,next)=>

{

  let token

  const {authorization} = req.headers

  if(authorization && authorization.startsWith('Bearer'))

  {

    try{

token =  authorization.split(' ')[1]

console.log("Token " + token)

console.log("Auth " + authorization)

const data = {

  user :{

    email:req.body.email

  }

}

console.log("d " + data)

console.log("first")

console.log(process.env.key)

const {email} = jwt.verify(token,process.env.key)

// console.log("email is " + email)

req.user = await userModel.findOne({email:email})

console.log(req.user)

next()

}

    catch(error)

    {

      res.status(401).send({"status":"fail","message":"auautho"})

    }

  }

  if(!token)

  {

    res.status(401).send({"status":"fail","message":"No token"})

  }

}

export default chekUser;