const bcrypt = require("bcrypt"); //instalamos esto para mejorar el nivel de seguridad en la encriptacion de contraseñas

const saltRounds = 10; //complemento a la linea anterior

const app = express();

app.use(express.static("public"));

app.set("view engine", "ejs");

app.use(bodyParser.urlencoded({ extended: true }));

//conexion con el servidor de mongoDB

mongoose.connect("mongodb://localhost:27017/userDB", { useNewUrlParser: true, useUnifiedTopology: true});

//creacion de esquema para cada documento (conjunto de datos)

const userSchema = new mongoose.Schema ({ //esquema creado por medio de la clase "mongoose.Schema"

    email: String,

    password: String

});

//creacion de modelo de tabla con base en el esquema (la cual mongoDB pluraliza posteriormente)

const User = new mongoose.model("User", userSchema);

app.get("/", function(req, res) {

    res.render("home");

})

//VENTANA LOGIN

app.get("/login", function(req, res) {

    res.render("login");

});

app.post("/login", function(req, res){

    const username = req.body.username;

    const password = req.body.password;

    //verificamos si el email del usuario se encuentra registrado, en caso tal se verifica si la contraseña ingresada es la misma que se encuentra almacenada

    User.findOne({email: username}, function(err, foundUser) {

        if(err) {

            console.log(err);

        } else {

            if(foundUser) {

                bcrypt.compare(password, foundUser.password, function(err, result) { //esta funcion permite comparar la contraseña encriptada que se encuentra previamente almacenada, con la contraseña que ingresa el usuario

                    if(result){

                        res.render("secrets");

                    }

                });

            }

        }

    });

});

//VENTANA REGISTER

app.get("/register", function(req, res) {

    res.render("register");

});

app.post("/register", function(req, res) {

    bcrypt.hash(req.body.password, saltRounds, function(err, hash) {

        const newUser = new User({

            email: req.body.username, //se relaciona con el input que se encuentra en register.ejs cuyo "name" es "username"

            password: hash //se usa el hash generado por la funcion bcrypt.hash, la cual realiza una encriptacion de mayor complejidad

        });

        newUser.save(function(err) { //justo en este momento, al ejecutar .save, se realiza la encriptacion del dato seleccionado previamente, en ete caso se encripta "password"

            if(err) {

                console.log(err);

            } else {

                res.render("secrets");

            }

        });

    });

});

NIVEL 4 DE SEGURIDAD EN LA ENCRIPTACION DE CONTRASEÑAS