2025-I

UNAM

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

MATERIA

Programación Web 2

TEMA

Tokens

PROFESOR

Mtro. ISC. MIGUEL ANGEL SÁNCHEZ HERNÁNDEZ

Contenido

1. Ocultamiento de datos en variables de entorno

Para guardar nuestras variables de entorno ocuparemos instalamos dornev

1. npm i dotenv
2. Crear en raiz el archivo “.env”
3. Dentro de el agregar lo siguiente:

```
BD_NOMBRE=hoteles
BD_USUARIO=root
BD_CLAVE=1234
BD_DIALEC=mariadb
BD_HOST=127.0.0.1
BD_PORT=3306
```

4. Cambiar el archivo db.js por lo siquiente:

```
import { Sequelize } from "sequelize";
import dotenv from 'dotenv';
dotenv.config({path:'.env'});
const db=new Sequelize( process.env.BD_NOMBRE,process.env.BD_usuario,process.env.BD_clave,{
    dialect:process.env.BD_DIALEC,
    dialectOptions:{
        host:process.env.BD_HOST,
        port:process.env.BD_PORT,
        timestamps:false,
        underscore:false,
        pool:{
            max:5,
            min:0,
            acquire:30000,
            idle:10000
        },
        operatorAlies:false

    }
});
export default db;
```

5. Arracar el sevidor: **npm run server**

2. Modelo de Usuario y Rol

En la carpeta models, crear los archivos Usuario.js y Rol.js, su contenido es el siguiente:

Usuario.js

```
import Sequelize from "sequelize";
import {DataTypes} from 'sequelize';
import db from '../config/db.js';
import Rol from "./Rol.js";

const Usuario= db.define('usuarios',{
    id_usr: {
        type: Sequelize.INTEGER,
        autoIncrement: true,
        primaryKey: true,
      },
    nombre:{
        type:DataTypes.STRING,
        allowNull:false
    },
    correo:{
        type:DataTypes.STRING,
        allowNull:false
    },
    password:{
        type:DataTypes.STRING,
        allowNull:false
    },
    confirmar:DataTypes.BOOLEAN,
    token:DataTypes.STRING

},{
    timestamps: false,
});
Rol.hasOne(Usuario, {
    foreignKey: {
      name: "id_rls",
    },
  });
Usuario.belongsTo(Rol, {
    foreignKey: {
      name: "id_rls",
    },
  });

export default Usuario;
```

Rol.js

```
import Sequelize from "sequelize";
import {DataTypes} from 'sequelize';
import db from '../config/db.js';

const Rol= db.define('roles',{
    id_rls: {
        type: Sequelize.INTEGER,
        autoIncrement: true,
        primaryKey: true,
      },
    nombre:{
        type:DataTypes.STRING,
        allowNull:false
    }

},{
    timestamps: false,
});

export default Rol;
```

3. Controlador para autentificación

1.- Sustituir en index.js lo siguienes:

```
import credenciales_router from "./routes/credenciales_router.js";
.
.
app.use("/", credenciales_router);
app.use("/hotel",router_Hotel);
app.use("/gerente",router_Gerente);
```

2.- Crear la carpeta **routes** y dentro de ella el archivo **credenciales_router.js**

```
import express from "express";
import {inicioSesion} from "../controllers/credenciales/loginController.js"
const routerCredenciales=express.Router();
//Routing
//para la vista alta credenciales
routerCredenciales.get('/',inicioSesion);

export default routerCredenciales
```

3.- Crear la carpeta **controller/credenciales** y dentro de ella el archivo **loginController.js**

```
import Usuario from "../../models/Usuario.js";

const inicioSesion = (req, res) => {
    res.render("credenciales/login", {
      pagina: "Autentificación",
    });
  };
  export {inicioSesion};
```

4.- Crear la carpeta **view/credenciales** y dentro de ella el archivo **login.pug**

```
doctype html
html(lang="es")
   head
      meta(charset="UTF-8")
      meta(name="viewport", content="width=device-width, initial-scale=1.0")
      title Credenciales
      link(rel="stylesheet", href="/css/bootstrap.css")
      script(src="/js/bootstrap.bundle.js")
      script(src="/js/pooper.min.js")

   body
      h1.text-center=pagina
         div(class="row justify-content-center")
            if errores
               div(class="row justify-content-center text-bg-danger" )
                  div(class="col-md-5")
                     each er in errores
                        h1(class="fw-bold fs-6")=er.msg
            div(class="col-md-5")
               form(class="space-y-5" action="/cred/registrar" method="POST" noValidate)
                  br
                  br
                  div(class="mb-1")
                     label(for="usuario" class="form-label") Usuario
                     input#usuar(class="form-control" type="text" placeHolder="Usuario" name="usuario")
                  div(class="mb-1")
                     label(for="clave" class="form-label") Clave
                     input#clave(class="form-control" type="password" placeHolder="Clave" name="clave")
                  div(class="mb-1")
                     button(type="submit" class="btn btn-primary col-auto") Enviar
                  div(class="mb-4")
                     h6
                        a(class="icon-link" href=" /registrar") Crear Registro
                        |     
                        a(class="icon-link" href="/registrar") Olvide Password
```

4. Controlador para registrar usuario

1.- Modificar el archivo **credenciales_router.js**

```
import express from "express";
import {inicioSesion,registrandoEnlace} from "../controllers/credenciales/loginController.js"
const routerCredenciales=express.Router();
//Routing
//para la vista alta credenciales
routerCredenciales.get('/',inicioSesion);
routerCredenciales.get('/registrar',registrandoEnlace);

export default routerCredenciales
```

3.- Modificar el archivo **loginController.js**

```
import Usuario from "../../models/Usuario.js";

const inicioSesion = (req, res) => {
   res.render("credenciales/login", {
     pagina: "Autentificación",
   });
 };
 const registrandoEnlace = (req, res) => {
   res.render("credenciales/registrar", {
     pagina: "Alta Usuario",
   });
 };

 export {inicioSesion,registrandoEnlace};
```

4.- Crear en **view/credenciales** el archivo **registrar.pug**

```
doctype html
html(lang="es")
   head
      meta(charset="UTF-8")
      meta(name="viewport", content="width=device-width, initial-scale=1.0")
      title Credenciales
      link(rel="stylesheet", href="/css/bootstrap.css")
      script(src="/js/bootstrap.bundle.js")
      script(src="/js/pooper.min.js")

   body
      h1.text-center=pagina
         div(class="row justify-content-center")
            if errores
               div(class="row justify-content-center text-bg-danger" )
                  div(class="col-md-5")
                     each er in errores
                        h1(class="fw-bold fs-6")=er.msg
            div(class="col-md-5")
               form(class="space-y-5" action="/registrar" method="POST" noValidate)
                  br
                  br
                  div(class="mb-1")
                     label(for="usuario" class="form-label") Usuario
                     input#usuar(class="form-control" type="text" placeHolder="Usuario" name="usuario")
                  div(class="mb-1")
                     label(for="correo" class="form-label") Correo
                     input#correo(class="form-control" type="text" placeHolder="demo@demo.com" name="correo")

                  div(class="mb-1")
                     label(for="clave" class="form-label") Clave
                     input#clave(class="form-control" type="password" placeHolder="Clave" name="clave")
                  div(class="mb-1")
                     button(type="submit" class="btn btn-primary col-auto") Enviar
```

5. Almacenar usuario

1.- Modificar el archivo **credenciales_router.js**

```
import express from "express";
import {inicioSesion,,registrandoEnlace,registrando} from "../controllers/credenciales/loginController.js"
const routerCredenciales=express.Router();
//Routing
//para la vista alta credenciales
routerCredenciales.get('/',inicioSesion);
routerCredenciales.get('/registrar',registrandoEnlace);
routerCredenciales.post('/registrar',registrando);

export default routerCredenciales
```

2.- Instalar express-validator

```
npm i express-validator
```

3.- Crear la carpeta **helpers** en raiz y crear dentro de ella el archivo **tokens.js**

```
const idGenera=()=>Math.random().toString(32).substring(2)+Date.now().toString(32);
export {
   idGenera
}
```

4.- Modificar el archivo **loginController.js**

```
import Usuario from "../../models/Usuario.js";
import { check, validationResult } from "express-validator";
import { idGenera } from "../../helpers/tokens.js";

const inicioSesion = (req, res) => {
   res.render("credenciales/login", {
     pagina: "Autentificación",
   });
 };
 const registrandoEnlace = (req, res) => {
   res.render("credenciales/registrar", {
     pagina: "Alta Usuario",
   });
 };

const registrando = async (req, res) => {
   let valido = await validacionFormulario(req);

   if (!valido.isEmpty()) {
     return res.render("credenciales/registrar", {
       pagina: "Alta Usuario",
       errores: valido.array(),
     });
   }
   const usuario = await Usuario.create({
     nombre: req.body.usuario,
     password: req.body.clave,
```

```
    correo:req.body.correo,
    id_rls:2,
    token:idGenera()
  });
  await usuario.save();

  //mostrar mensaje de confirmacions
  res.render("credenciales/confirmacion", {
    pagina: "Usuario se registro exitosamente",

  });
};

async function validacionFormulario(req) {
  await check("usuario")
    .notEmpty()
    .withMessage("Usuario no debe ser vacio")
    .run(req);
  await check("clave")
    .notEmpty()
    .withMessage("Clave no debe ser vacio")
    .run(req);
  await check("correo")
    .notEmpty()
    .withMessage("Correo no debe ser vacio")
    .run(req);

  let salida = validationResult(req);
  return salida;
}

export {inicioSesion,registrandoEnlace,registrando};
```

4.- Crear en **view/credenciales** el archivo **confirmacion.pug**

```
doctype html
html(lang="es")
  head
    meta(charset="UTF-8")
    meta(name="viewport", content="width=device-width, initial-scale=1.0")
    title Credenciales
    link(rel="stylesheet", href="/css/bootstrap.css")
    script(src="/js/bootstrap.bundle.js")
    script(src="/js/pooper.min.js")

  body
    h1.text-center=pagina
      div(class="row justify-content-center")
        h3(class="fw-bold fs-6")=mensaje
        <a class="btn btn-primary" href="/" role="button">Inicio</a>
```

Ejercicio

Crear la lógica necesario para validar las credenciales del usuario y enviarlo a la pagina de inicio.pug