Laboratorio 19

**Capacidad del modelo RBAC Controlador centralizado de permisos:**  estos son los roles y los privilegios.

**Facilidad de administración:** asignación de roles y gestión de roles.

**Seguridad mejorada:** Acceso Basado en Necesidad Auditoría y Cumplimiento:

**Soporte del modelo en la base de datos (sql):**

-- Crear tablas para el modelo RBAC

CREATE TABLE Roles (

Id\_Rol INT PRIMARY KEY,

Descripcion VARCHAR(255)

);

CREATE TABLE Privilegios (

Id\_Privilegio INT PRIMARY KEY,

Descripcion VARCHAR(255),

Accion VARCHAR(255)

);

CREATE TABLE Usuarios (

Id\_Usuario INT PRIMARY KEY,

Nombre VARCHAR(255),

Apellidos VARCHAR(255),

Fecha\_Nacim DATE,

Fecha\_Crea DATE,

Balance DECIMAL(10, 2),

Contraseña VARCHAR(255),

Habilitado BOOLEAN

);

CREATE TABLE Trabajadores (

Id\_Usuario INT PRIMARY KEY,

RFC VARCHAR(13),

Sueldo DECIMAL(10, 2),

Seguro VARCHAR(255),

Telefono VARCHAR(10),

FOREIGN KEY (Id\_Usuario) REFERENCES Usuarios(Id\_Usuario)

);

CREATE TABLE Areas\_de\_Trabajo (

Id\_Area INT PRIMARY KEY,

Descripcion VARCHAR(255)

);

CREATE TABLE Secciones (

Id\_Seccion INT PRIMARY KEY,

Ubicacion VARCHAR(255)

);

CREATE TABLE Transacciones (

Id\_Trans INT PRIMARY KEY,

Tipo VARCHAR(255),

Descripcion VARCHAR(255)

);

CREATE TABLE Generar (

Id\_Usuario INT,

Id\_Trans INT,

Fecha DATE,

Tipo VARCHAR(255),

Monto DECIMAL(10, 2),

PRIMARY KEY (Id\_Usuario, Id\_Trans),

FOREIGN KEY (Id\_Usuario) REFERENCES Usuarios(Id\_Usuario),

FOREIGN KEY (Id\_Trans) REFERENCES Transacciones(Id\_Trans)

);

CREATE TABLE Tiene (

Id\_Rol INT,

Id\_Privilegio INT,

PRIMARY KEY (Id\_Rol, Id\_Privilegio),

FOREIGN KEY (Id\_Rol) REFERENCES Roles(Id\_Rol),

FOREIGN KEY (Id\_Privilegio) REFERENCES Privilegios(Id\_Privilegio)

);

CREATE TABLE Posee (

Id\_Usuario INT,

Id\_Rol INT,

PRIMARY KEY (Id\_Usuario, Id\_Rol),

FOREIGN KEY (Id\_Usuario) REFERENCES Usuarios(Id\_Usuario),

FOREIGN KEY (Id\_Rol) REFERENCES Roles(Id\_Rol)

);

**Obtener los permisos del usuario al autenticarse (**javascript)**:** const obtenerPermisos = async (usuarioId) => { const roles = await db.query(`

SELECT R.Descripcion AS rol, P.Descripcion AS privilegio, P.Accion

FROM Usuarios U

JOIN Posee PO ON U.Id\_Usuario = PO.Id\_Usuario

JOIN Roles R ON PO.Id\_Rol = R.Id\_Rol

JOIN Tiene T ON R.Id\_Rol = T.Id\_Rol

JOIN Privilegios P ON T.Id\_Privilegio = P.Id\_Privilegio

WHERE U.Id\_Usuario = ?`, [usuarioId]);

return roles;

};

**Generar la interfaz gráfica de manera dinámica (archivo** javascript)**:**

// Suponiendo que tienes los permisos en una variable `permisos` if (permisos.includes('ver\_dashboard')) { document.getElementById('dashboard').style.display = 'block';

} else {

document.getElementById('dashboard').style.display = 'none';

}

**Validar permisos en cada ruta (archivo** javascript**):**

const verificarPermiso = (permiso) => { return (req, res, next) => { if (req.usuario.permisos.includes(permiso)) { next(); } else {

res.status(403).send('Acceso denegado');

}

};

};

// Usar el middleware en una ruta app.get('/dashboard', verificarPermiso('ver\_dashboard'), (req, res) => { res.render('dashboard');

});

**Crear interfaz de gestión de roles y permisos (archivo HTML):**

<form action="/asignar-rol" method="post">

<label for="usuario">Usuario:</label>

<select name="usuario" id="usuario">

<!-- opciones de usuarios -->

</select>

<label for="rol">Rol:</label> <select name="rol" id="rol">

<!-- opciones de roles -->

</select>

<button type="submit">Asignar Rol</button>

</form>