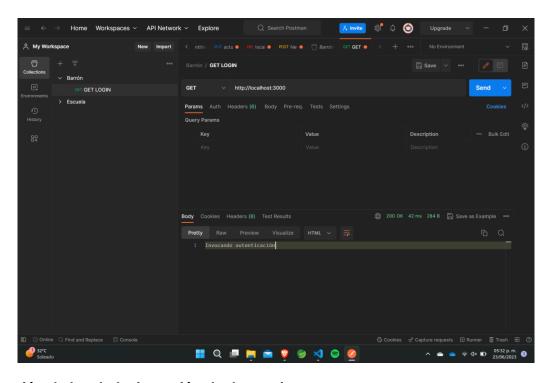
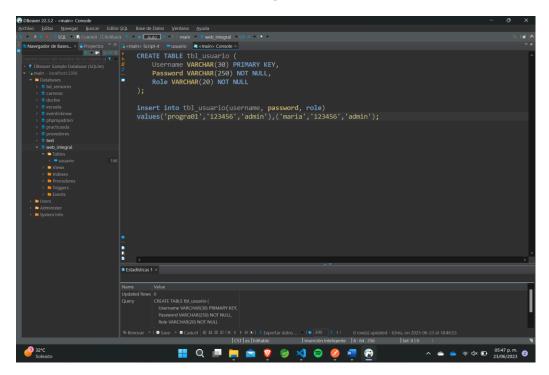
# María Guadalupe Mendoza Ramírez GDGS3091

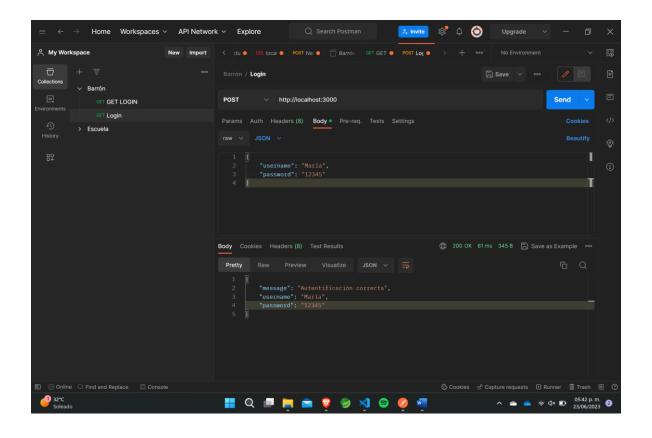
### Práctica Backend con NodeJS

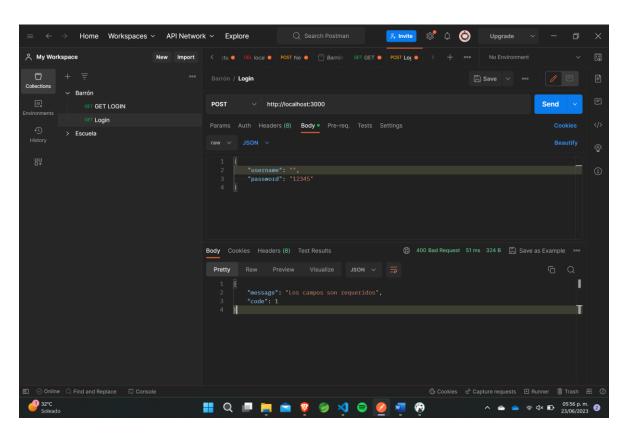
### **Autenticación**

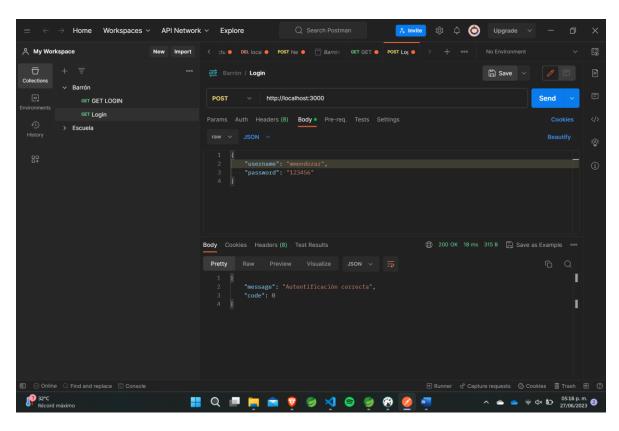


# Creación de la tabal e inserción de dos registros









### **CRUD**

# Configuración de rutas

```
★ File Edit Selection View Go …

                                                                                                                      TS authRoutes.ts TS usuarioRoutes.ts X TS usuarioController.ts
Ф
     ∨ BACKEND
      > build
                                            private config() {
       > node modules
                                                this.router.get('/listar', usuarioController.listar);
                                                this.router.post('/insertar', usuarioController.insertar)
        > middleware
        TS usuarioModelo.ts

∨ routes

        TS authRoutes.ts
        TS usuarioRoutes.ts
        TS connection.ts
                                                                                                                    ☑ node + ∨ Ⅲ 🛍 ··· ^ ×
                                 [1] role: 'creadorEventos'
[1] }
[1] POST /insertar 200 138.078 ms - 59
[1] pmartinez
[1] Eliminando DAO
[1] DELETE /eliminar/pmartinez 200 5.165 ms - 60
      {} package-lock.json
     > OUTLINE
     > TIMELINE
                                                                 Screen Reader Optimized Ln 25, Col 30 Spaces: 4 UTF-8 CRLF {} TypeScript © Go Live 💆 🚨
⊗ 0 △ 0
```

# Configuración de usuarioModelo para realizar las consultas

```
TS usuarioModelo.ts X TS connection.ts
Ф
                                src > models > TS usuarioModelo.ts > 😫 UsuarioModelo
     ∨ BACKEND
      > node_modules
                                           public async insertar(usuario: any) {
   const result = await pool.then( async (connection) => {
                                                 return await connection.query(
        TS connection.ts
       TS usuarioController.ts
                                                        "UPDATE tbl usuario SET ? WHERE username = ?".
                                                       [usuario, username]
        TS authRoutes.ts
        TS usuarioRoutes.ts
        TS connection.ts
                                          public async eliminar(username: string) {
        TS utils.ts
                                               console.log('Eliminando DAO');
                                                   return await connection.query(
      .env
      {} package-lock.json
tsconfig.json
                                 PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

    □ node + ∨ □ 
    □ ··· ^ ×

> OUTLINE
                                 17:18:30 - File change detected. Starting incremental compilation...
     > TIMELINE
                                                                 Screen Reader Optimized Ln 23, Col 1 Spaces: 4 UTF-8 CRLF () TypeScript © Go Live 💆 🗘
⊗ 0 △ 0
                                       🔡 Q 🔎 🛅 💼 🦁 🧶 💆 🖨 💠 👼 🖂 🤣
```

#### Partes modificadas del Controller

```
// inserción de los datos
//Agregar enunciado

const result = await model.insertar(usuario);

if (result.affectedRows > 0) {
    return res.json({message: "Los datos se guardaron correctamente", code: 0});
} else {
    return res.status(404).json({ message: result.message, code: 1});
}
```

```
// actualización de los datos
//Agregar enunciado

const result = await model.actualizar(newUser, usuario.username);

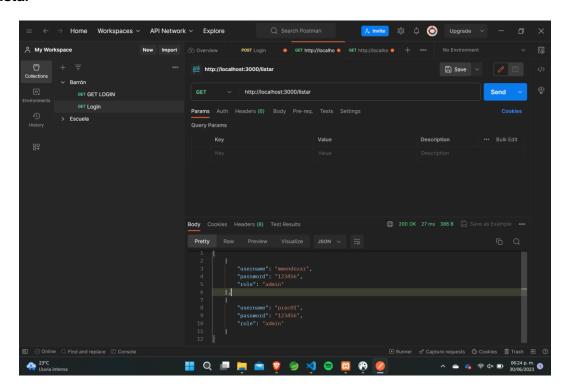
if (result.affectedRows > 0) {
    return res.json({message: "Los datos se actualizaron correctamente", code: 0});
} else {
    return res.status(404).json({ message: result.message, code: 1});
}
```

```
//Agregar enunciado

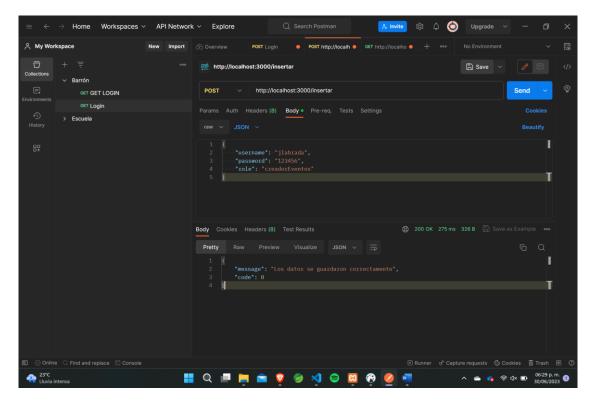
const result = await model.eliminar(username);

if (result.affectedRows > 0) {
    return res.json({message: "Los datos se eliminaron correctamente", code: 0});
} else {
    return res.status(404).json({ message: result.message, code: 1});
}
```

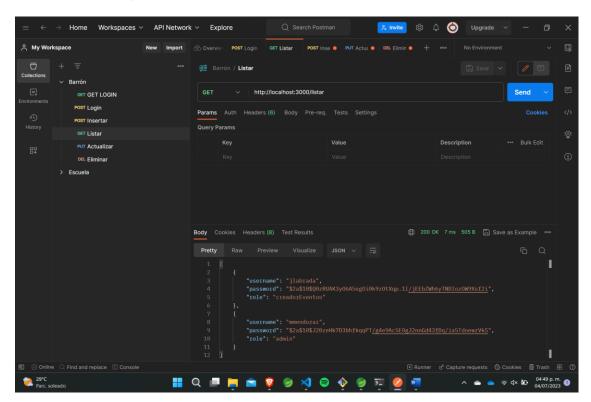
### Listar



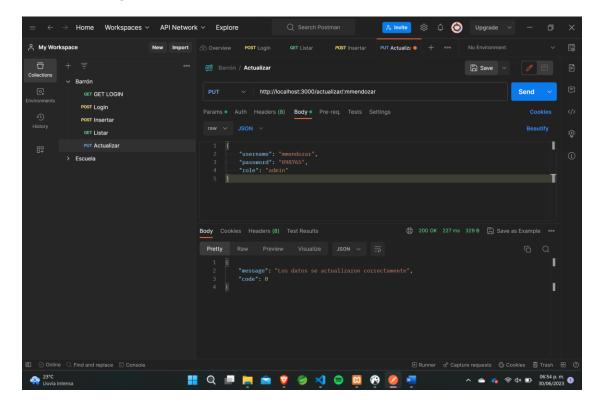
### Insertar un nuevo registro



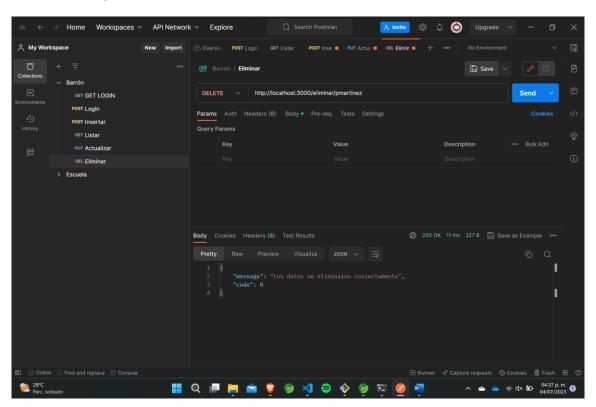
### Contraseñas encriptadas



# Actualizar un registro



# Eliminar un registro



### **JWT**

