



El futuro digital  
es de todos

MinTIC

# «Misión TIC 2022»



Universidad de Caldas

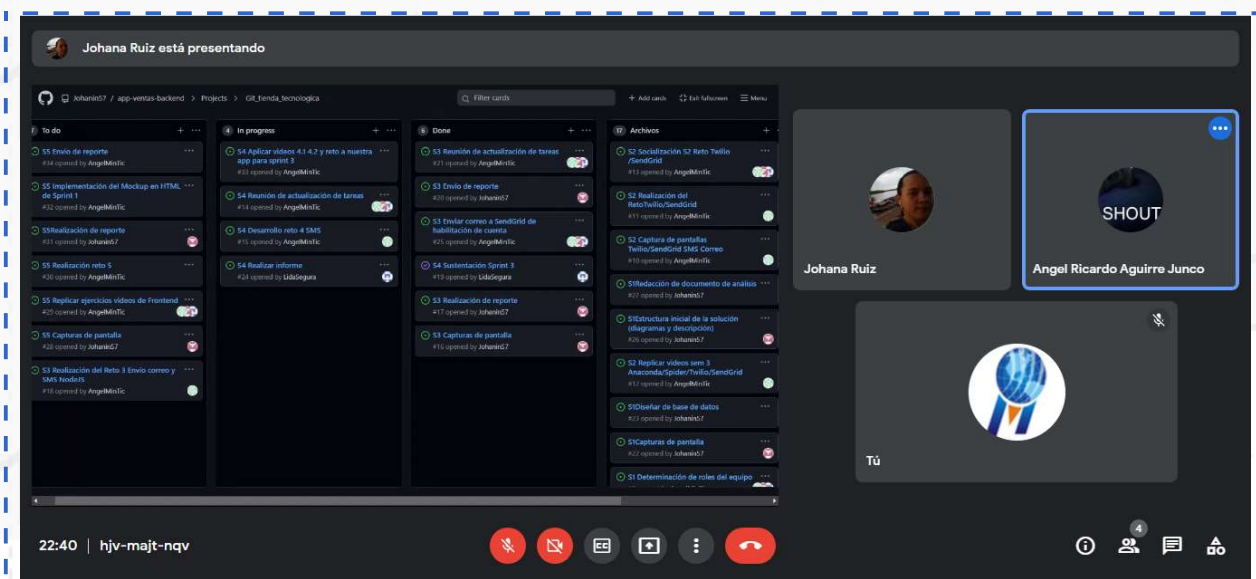


## Formato de Informe de Seguimiento

Integrantes (Nombre completo)	Cédula	Rol	Nivel de participación (Alto, Medio, Bajo, Retirado)
Segura Arbeláez Lida María	38886255	Administrador configuración	Alto
Aguirre Junco Ángel Ricardo	7175405	Diseñador de Software - Tester	Alto
Ruiz Johana Paola	41951132	Líder	Alto
Romero Peña Harol Andrés	1000156154	Diseñador UI	Alto
Sandoval Bermúdez Nicolás David	1000156154	No responde	Retirado

### 1. Primera reunión (plan inicial del sprint) - lunes.

- Pantallazo:



#### Observaciones:

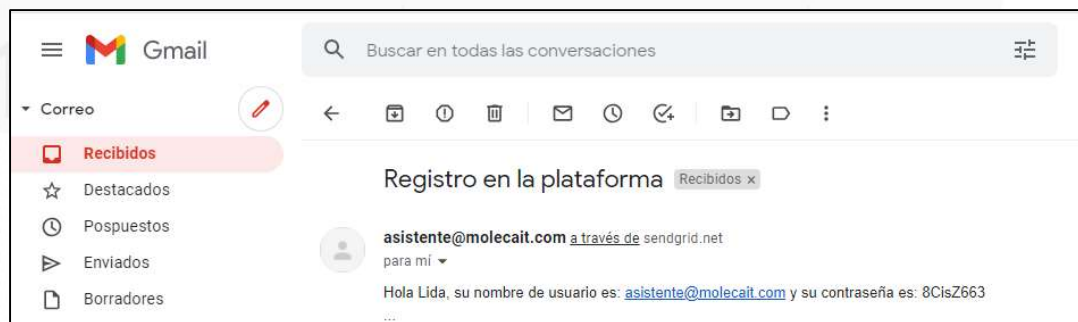
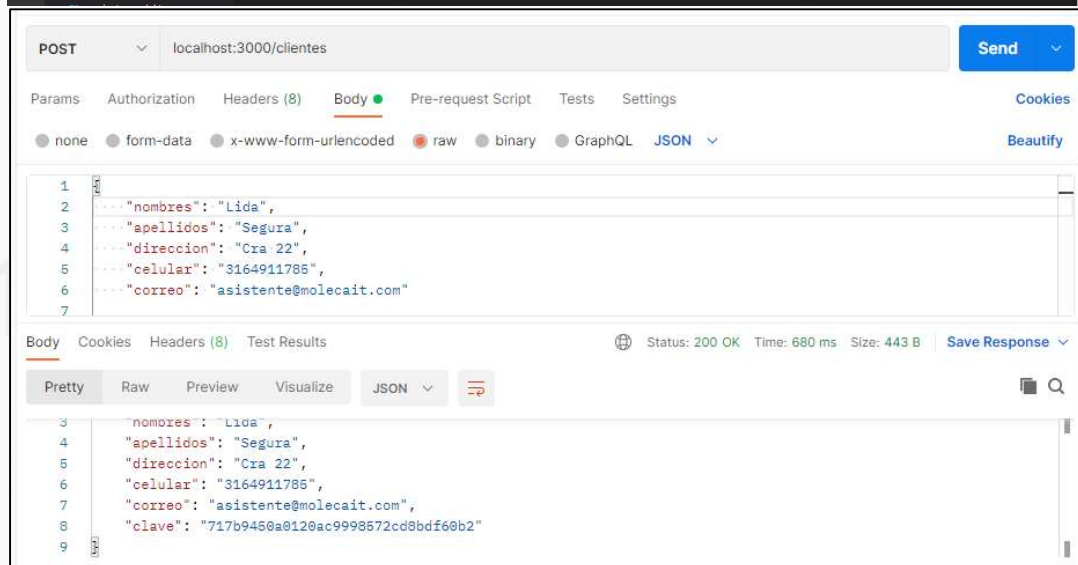
Se crean las tareas en GitHub, sobre las labores a realizar con referencia al desarrollo del código, se realiza el informe para la entrega del sprint 3.



## 2. Reunión diaria de seguimiento - martes.

- Pantallazo:

```
src > controllers > TS cliente.controller.ts M TS cliente.model.ts M TS autenticacion.service.ts M
src > controllers > TS cliente.controller.ts M TS cliente.controller.ts M create
52 let clave = this.servicioAutenticacion.GenerarClave();
53 let claveCifrada = this.servicioAutenticacion.CifrarClave(clave);
54 cliente.clave = claveCifrada;
55 let p = await this.clienteRepository.create(cliente);
56
57 //Notificar al usuario
58 let destino = cliente.correo;
59 let asunto = 'Registro en la plataforma';
60 let contenido = `Hola ${cliente.nombres}, su nombre de usuario es: ${cliente.correo} y su contraseña es: ${claveCifrada}`;
61 fetch('http://127.0.0.1:5000/envio-correo?correo_destino=${destino}&asunto=${asunto}&contenido=${contenido}')
62 .then((data:any) => {
63   console.log(data);
64 })
65 return p;
66
67 }
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
```







Observaciones:

Para realizar el sprint # 3, fue necesario la instalación de los siguientes comandos:  
npm i crypto-js,  
npm i password-generator,  
npm i node-fetch@2.6.5,  
npm i @types/node-fetch.

Se comparte el pantallazo de la integración del código en Visual Studio Code para envío de email, con la clave cifrada para el cliente. Se evidencia la prueba en Postman y envío del correo de manera efectiva.

### 3. Reunión diaria de seguimiento - miércoles.

- Pantallazo:

The screenshot shows the Visual Studio Code interface with the Explorer, Search, and Run and Debug views. The Explorer view shows the project structure with folders like .vscode, node\_modules, public, src, \_\_tests\_\_, config, llaves.ts, and controllers. The Search view shows the results of a search for 'llaves.ts'. The Run and Debug view shows the execution of a command in the terminal.

```
C:\Users\USER\MisionTIC2022\Loopback\BackendVentas>lb4 model
? Nombre de clase Model: Credenciales
? Seleccione la clase base de modelo Model (A business domain object)
? ¿Desea permitir propiedades (de formato libre) adicionales? No
Model Credenciales se creará en src/models/credenciales.model.ts

Vamos a añadir una propiedad a Credenciales
Especifique un nombre de propiedad vacío cuando haya terminado

? Especifique el nombre de propiedad: usuario
? Tipo de propiedad: string
? ¿Es usuario la propiedad de ID? No
? ¿Es necesario?: Yes

Vamos a añadir otra propiedad a Credenciales
Especifique un nombre de propiedad vacío cuando haya terminado

? Especifique el nombre de propiedad: clave
? Tipo de propiedad: string
? ¿Es clave la propiedad de ID? No
? ¿Es necesario?: Yes

Vamos a añadir otra propiedad a Credenciales
Especifique un nombre de propiedad vacío cuando haya terminado

? Especifique el nombre de propiedad:
create src\models\credenciales.model.ts
update src\models\index.ts

Model Credenciales se ha/han creado en src\models

C:\Users\USER\MisionTIC2022\Loopback\BackendVentas>
```




to la primera parte, donde fue neces

- Instalar los siguientes paquetes:  
npm i jsonwebtoken,
- Importar cliente:  
import {Cliente} from '../models';
- Crear nueva carpeta:  
src/config/llaves.ts.
- Adicionalmente se crea un nuevo modelo que se llama Credenciales.

- Pantallazo:





 **JWT**

Depurador Bibliotecas Introducción Pedir El

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJkYXRhIjp7ImlkIjoibjE5OTc1MzI1NjU0ODgxeZDc0ZTY2YjNkIiwiaWV29ybmVvIjoieXNpc3RlbnRlQG1vbGVjYW10LmNvbSI6Im5vbWJyZSI6IkpXVCJ9GEGU2VndXJhIn0sIm1hdCI6MTYzNzQ1NzYxNH0u7ZHxUCtr42foADPMCx44WuCJC89kwM9BN4ZASKSpKU
```

ENCABEZAMIENTO: ALGORITMO Y TIPO DE TOKEN

```
{  "alg": "HS256",  "typ": "JWT"}
```

CARGA ÚTIL: DATOS

```
{  "datos": {    "id": "619975325654881d74e66b3d",    "correo": "asistente@molecait.com",    "nombre": "Lida Segura"  },  "iat": 1637457614}
```

VERIFICAR FIRMA

HMACSHA256 (

base64UrlEncode (encabezado) + "." +

base64UrlEncode (carga útil),

) ☐ secreto codificado en base64

Observaciones:

Se realiza el proceso de generar el Token para identificar de cliente.

Se hace la prueba en Postman, dando como respuesta el Token, y la respectiva confirmación del en la plataforma de [www.jwt.io](http://www.jwt.io) donde se puede validar que es válido.

## 5. Reunión diaria de seguimiento – viernes.

- Pantallazo:



The screenshot shows the Visual Studio Code interface with the following details:

- Explorer:** Shows the project structure with folders like `node_modules`, `public`, `src`, and `config`. The `src` folder is expanded, showing `_tests_` and `config`.
- Open Editors:** Lists several TypeScript files, including `cliente.controller.ts`, `admin.strategy.ts`, `cliente.model.ts`, `autenticacion.service.ts`, and `llaves.ts`.
- Editor:** Displays the implementation of the `authenticate` method in `admin.strategy.ts`. The code is as follows:

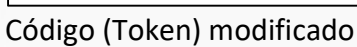
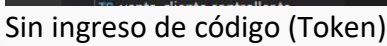
```
1 import {AuthenticationStrategy} from '@loopback/authentication';
2 import {UserProfile} from '@loopback/security';
3
4 export class EstrategiaAdministrador implements AuthenticationStrategy{
5   name: string = 'admin';
6
7   async authenticate(request: Request): Promise<UserProfile | undefined>{
8     |
9   }
10 }
11
```
- Terminal:** Shows the command `producto.controller.ts - BackendVentas - Visual Studio Code`.

#### Observaciones:

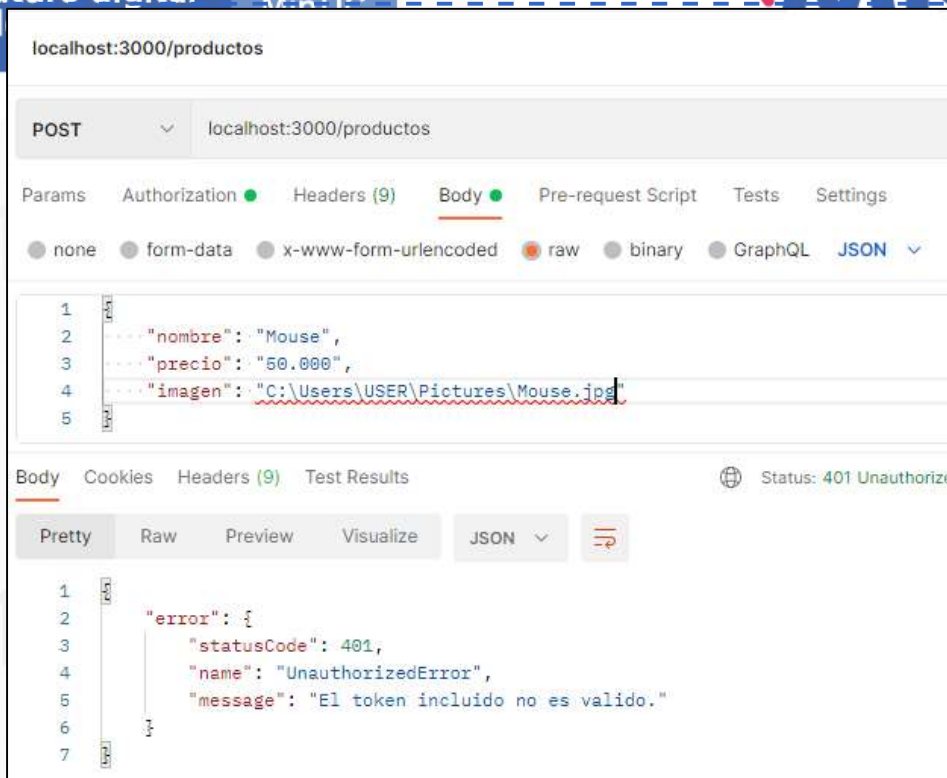
Se implementa lo visto en el video 4.2 segunda parte en nuestro proyecto, para lo cual fue necesario realizar los siguientes pasos:  
Se instalan los paquetes  
`npm i --force @loopback/authentication,`  
`npm i --force @loopback/security`  
`npm i parse-bearer-token,`  
Luego se crea una nueva carpeta que se llama *Estrategia Administrador*, esto con el fin de ingresar una solicitud de autenticación en Productos.

## 6. Reunión diaria de seguimiento – viernes.

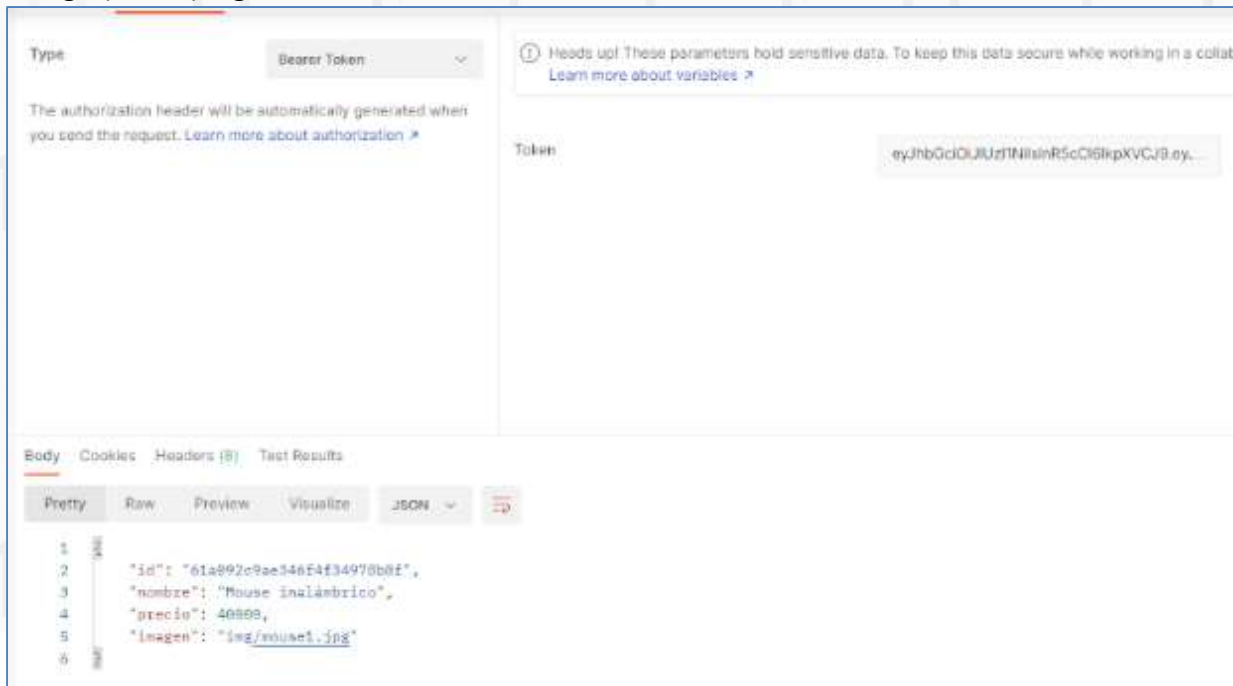
- Pantallazo:







Código (Token) ingresado correctamente.





El futuro digital  
es de todos

MinTIC

«Misión  
TIC2022»

Observaciones:

Se ingresa el código de Token de autenticación en la archivo de `producto.controllers.ts`, y se hacen las respectivas pruebas en Postman.

La primera prueba se hace sin ingresar el token, la segunda ingresando el token modificado y la tercera con el token bien escrito, lo que permite que el producto se cree correctamente.



Universidad de Caldas