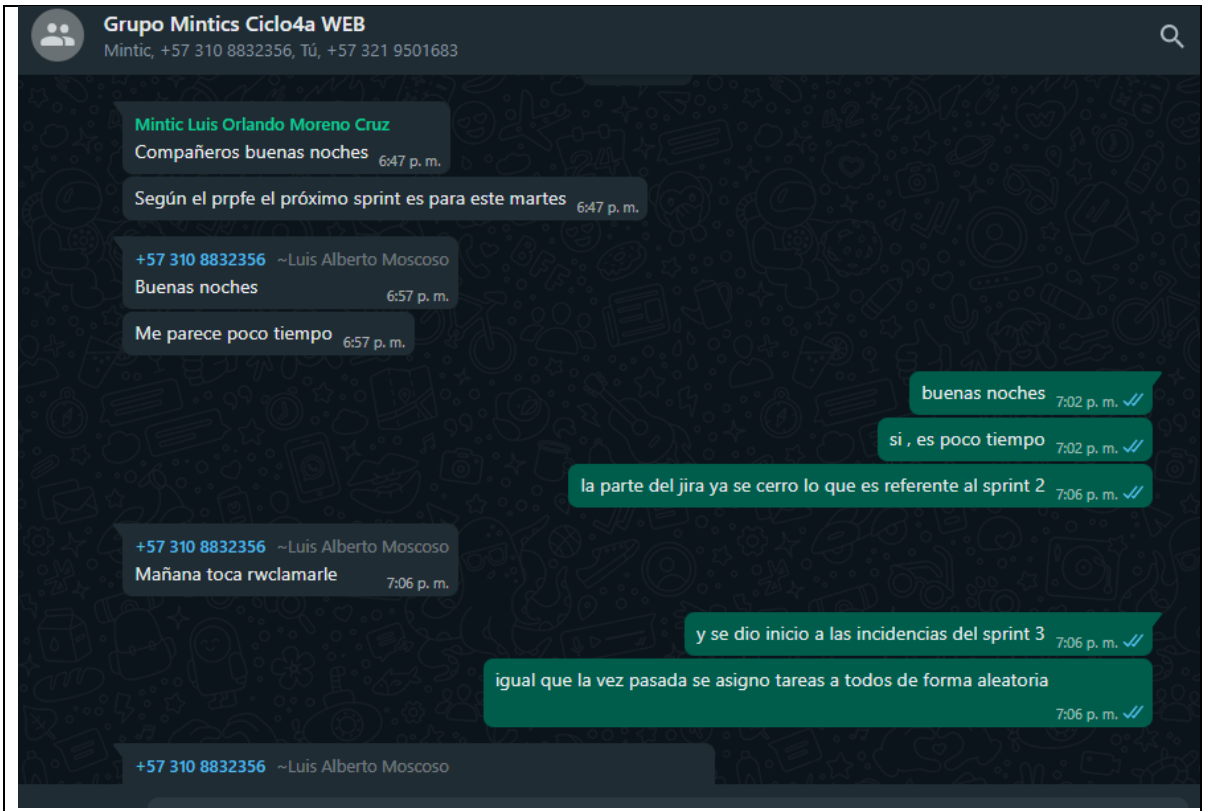
Formato de Informe de Seguimiento

Nombre del equipo:	Grupo 3 - Luis Alberto Moscoso Herrera - Luis Fernando Cortes - Luis Orlando Moreno Cruz - Luis Camilo Rosero Grijalba
Sprint No.:	3 (Tres)

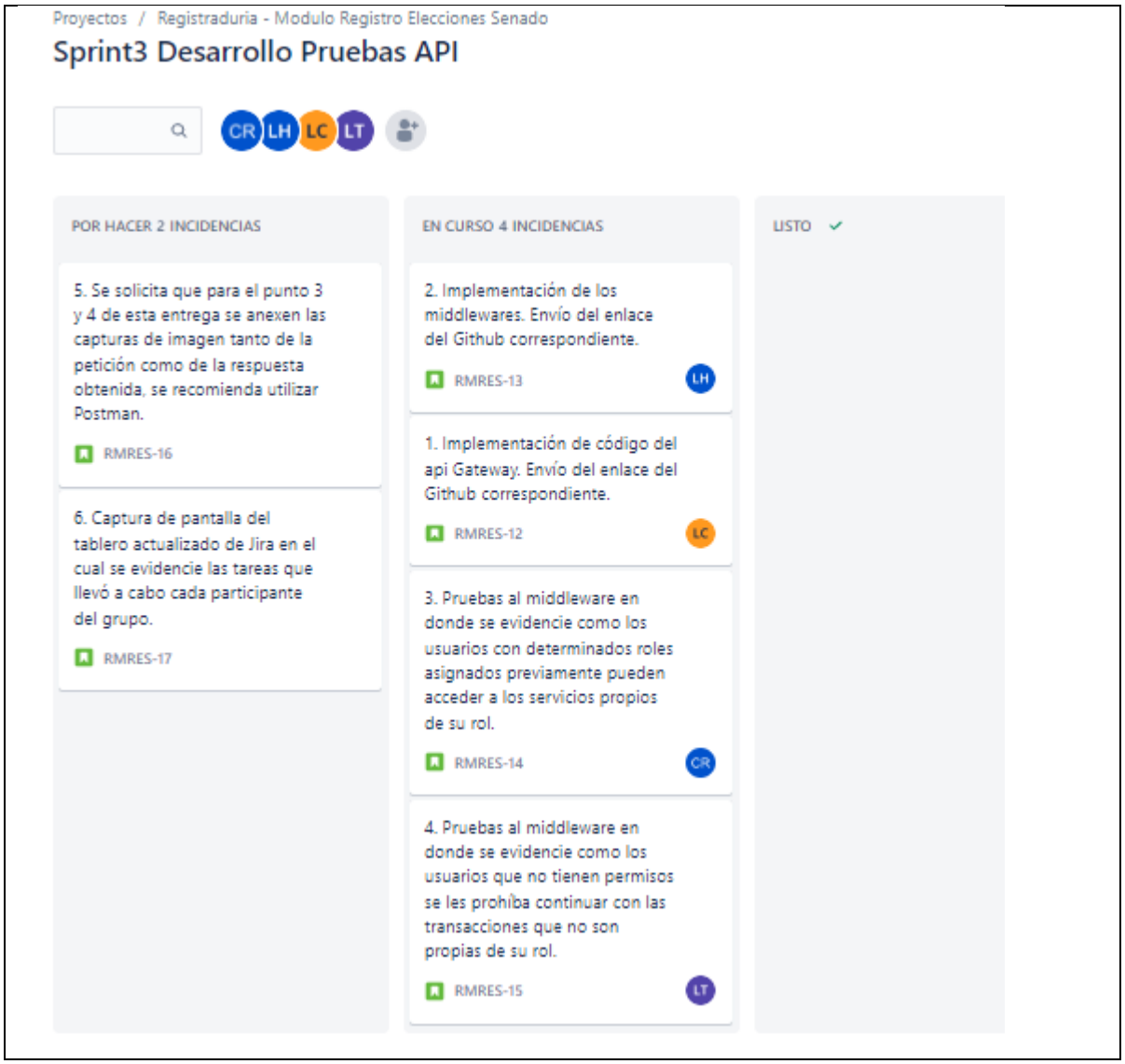
1. Primera reunión (plan inicial del sprint) - domingo.
   Pantallazo:



   Observaciones:

Reunión de inicio para el sprint 3 en el cual se plantearon las diferentes tareas para el desarrollo de éste

2. Reunión diaria de seguimiento - Lunes.
   Pantallazo:

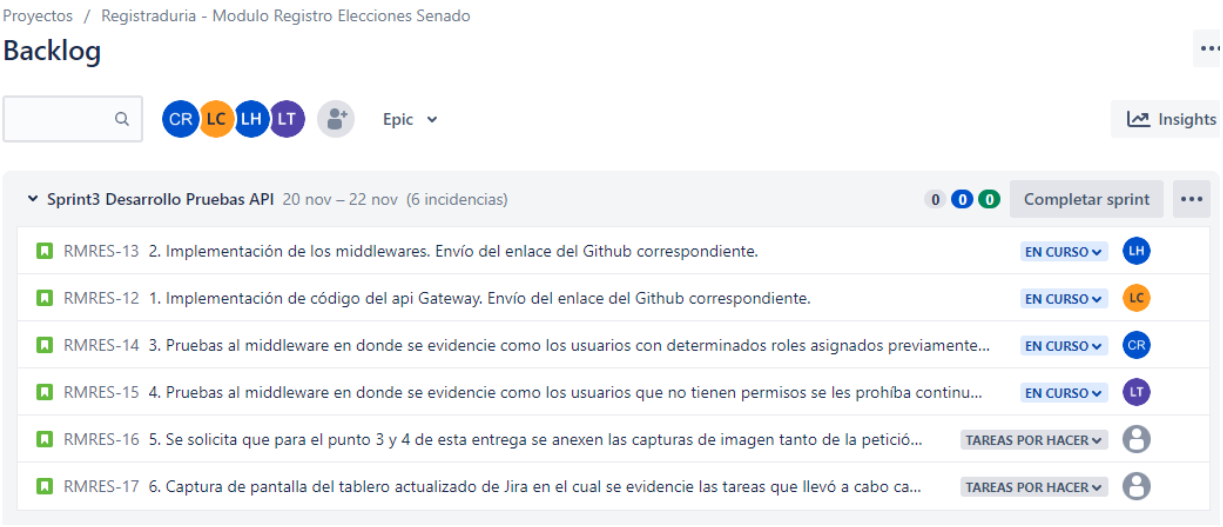


Observaciones:

A través de Jira se organizó y distribuyó las tareas a realizar para el sprint 3

Resumen de actividades

Como parte inicial de este sprint se organizó y distribuyo las diferentes actividades en la plataforma jira, la cual puede ser verificada en el siguiente enlace: https://kamirocode.atlassian.net/jira/software/projects/RMRES/boards/4

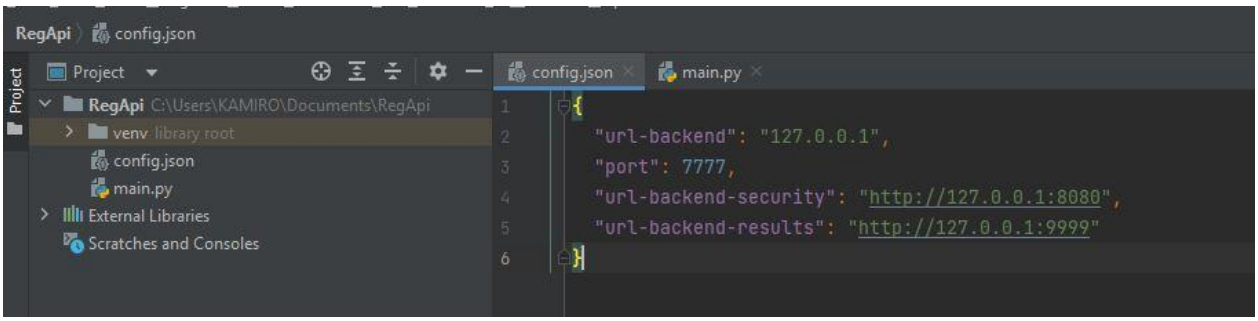


Implementación de código del api Gateway. Implementación de los middlewares. Envío del Pruebas al middleware en donde se evidencie como los usuarios con determinados roles asignados previamente pueden acceder a los servicios propios de su rol. Pruebas al middleware en donde se evidencie como los usuarios que no tienen permisos se les prohíba continuar con las transacciones que no son propias de su rol

La codificación implementada se encuentra en el siguiente enlace

https://github.com/luismoscoso49/MINTIC_CICLO4/tree/main/BACK_END/Api%20Gateway

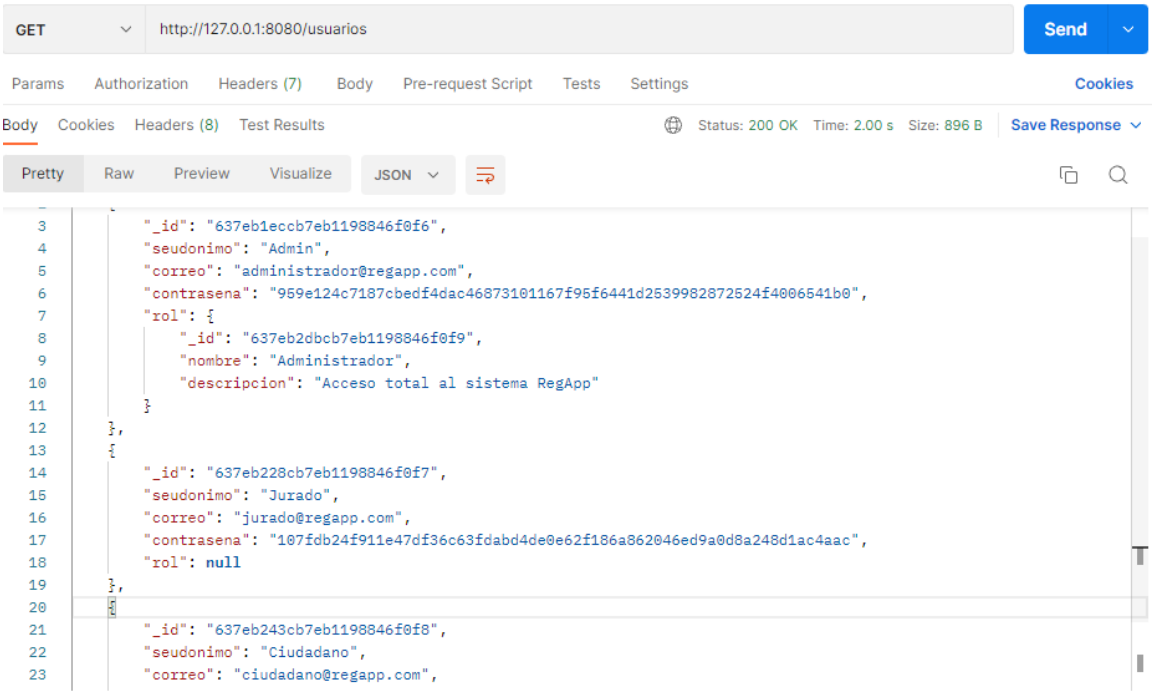
Implementación de código del api Gateway



Implementación del código middleware

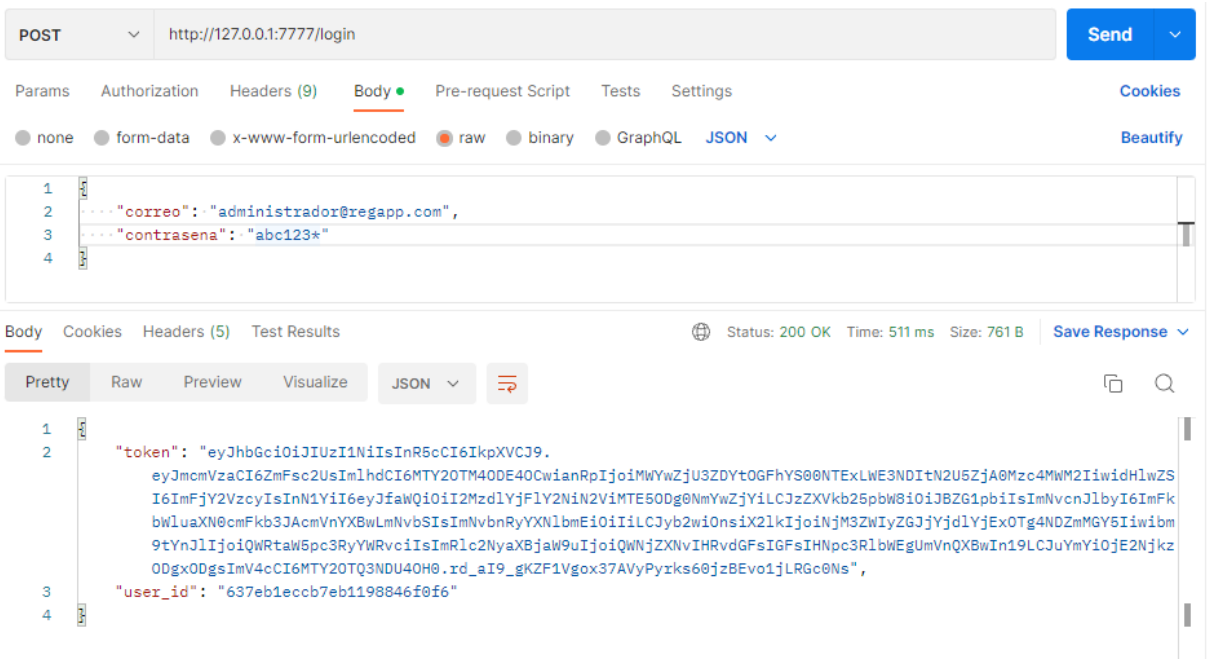
```python
# -------Middleware-------#
@app.route("/login", methods=["POST"])
def create_token():
    data = request.get_json()
    headers = {"Content-Type": "application/json; charset=utf-8"}
    url=dataConfig["url-backend-security"]+'/usuarios/validar'
    response = requests.post(url, json=data, headers=headers)
    if response.status_code == 200:
        user = response.json()
        expires = datetime.timedelta(seconds=60 * 60*24)
        access_token = create_access_token(identity=user, expires_delta=expires)
        return jsonify({"token": access_token, "user_id": user["_id"]})
    else:
        return jsonify({"msg": "Bad username or password"}), 401

@app.before_request
def before_request_callback():
    endPoint=limpiarURL(request.path)
    excludedRoutes=["/login"]
    if excludedRoutes.__contains__(request.path):
        pass
    elif verify_jwt_in_request():
        usuario = get_jwt_identity()
        if usuario["rol"]is not None:
            tienePermiso=validarPermiso(endPoint,request.method,usuario["rol"]["_id"])
```

Luego de haber realizado la respectiva estructuración del código en pycharm para el middleware, se realizaron pruebas con usuarios a los cuales previamente se les a asignado unos roles como lo son administrador, jurado y ciudadano.

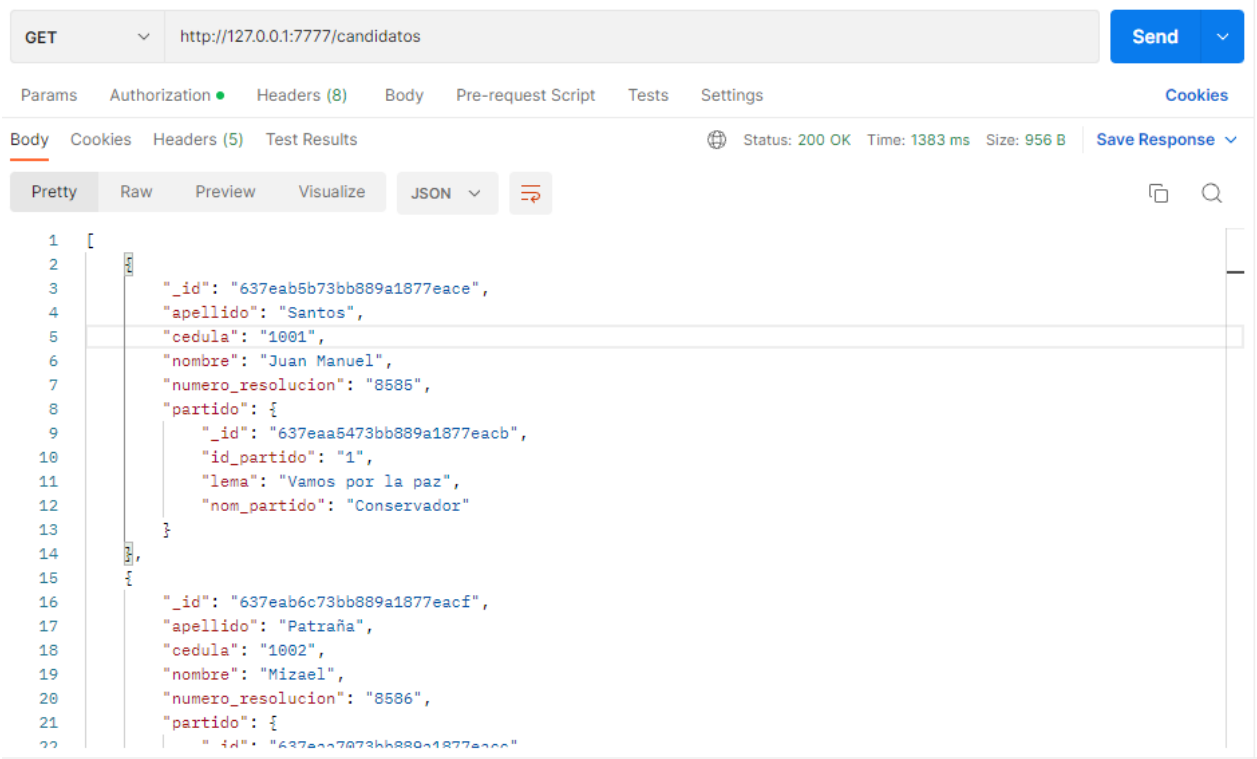


Como se puede observar en la anterior imagen solo el usuario Admin tiene un rol asignado que es el de Administrador, este usuario podrá acceder a unas rutas a través

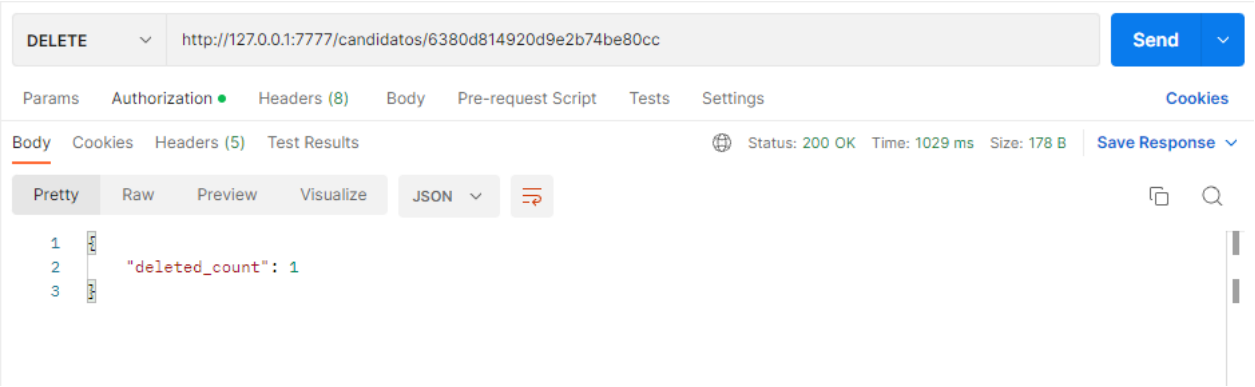
de unos permisos que se le fueron otorgados previamente a ese rol. Primero accedo al login en el cual se nos asigna un token



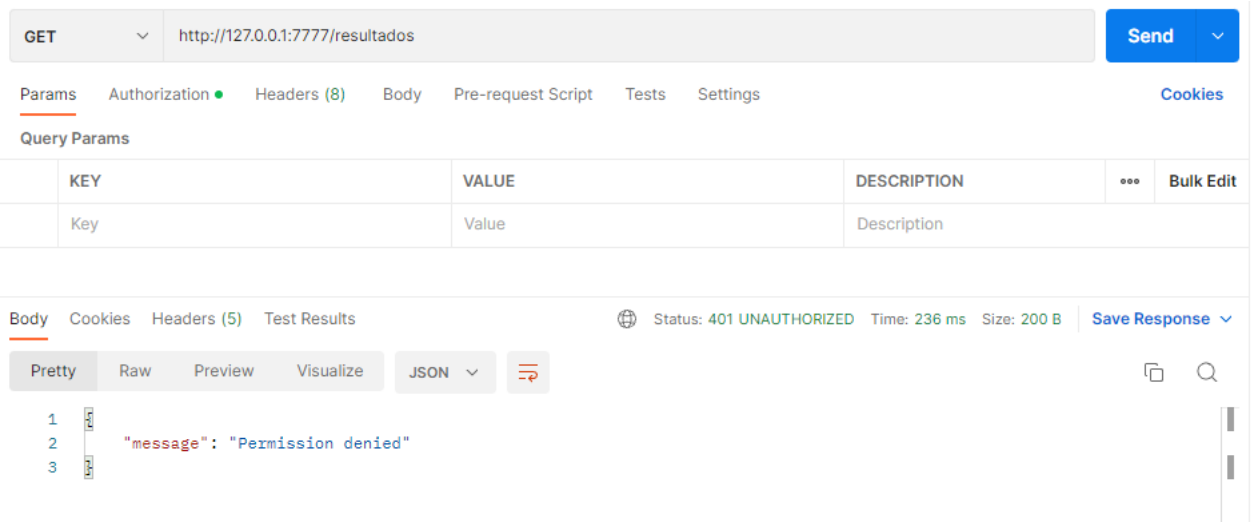
Luego con el token obtenido me dirijo a la ruta del api Gatewat que a través de la verificación que hace el middleware me dará el respectivo acceso a las rutas establecidas anteriormente.



Probamos otra ruta como con el método delete para un determinado dato con su respectivo id y se obtuvo lo siguiente

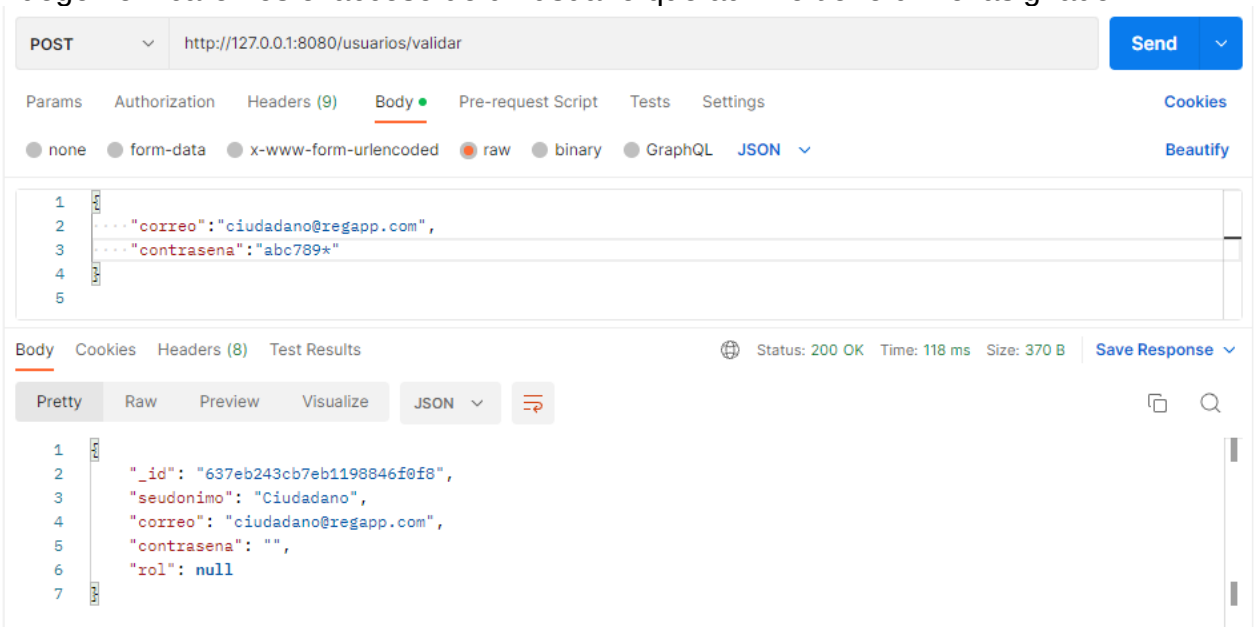


Finalmente, a este usuario no se le han asignado diferentes rutas de acceso, a continuación, se puede verificar la intensión de ingresar a una ruta diferente con su respuesta de acceso denegado

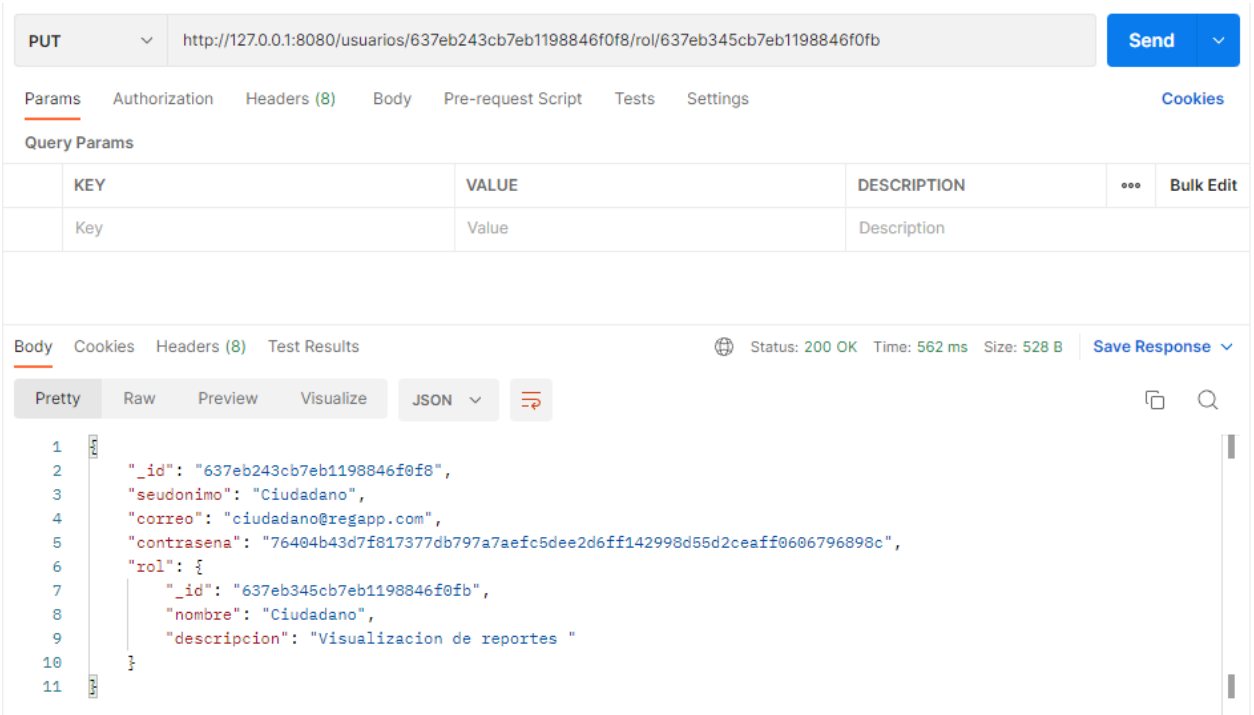


Pruebas de usuario sin acceso

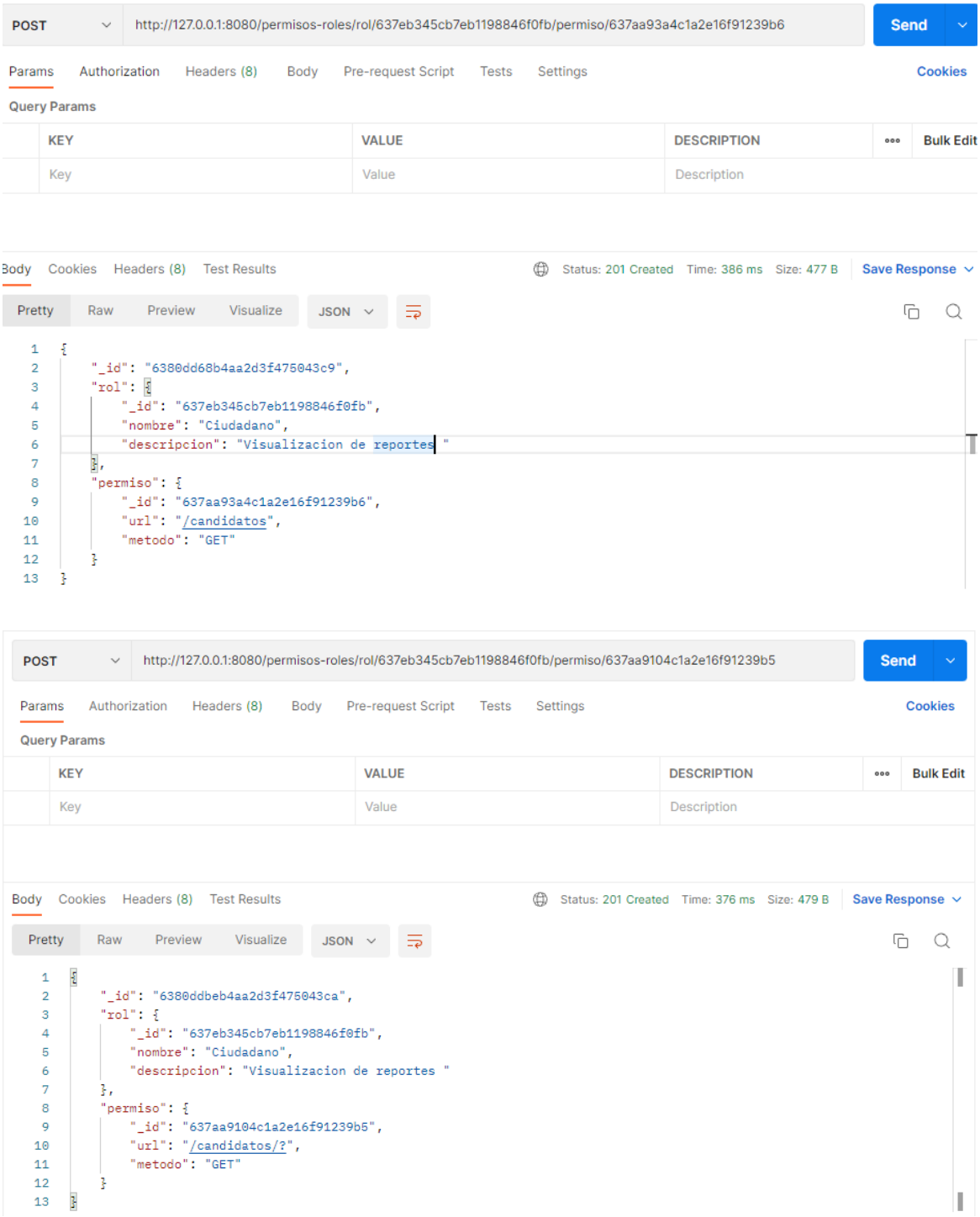
Luego verificaremos el acceso de un usuario que aun no tiene un rol asignado



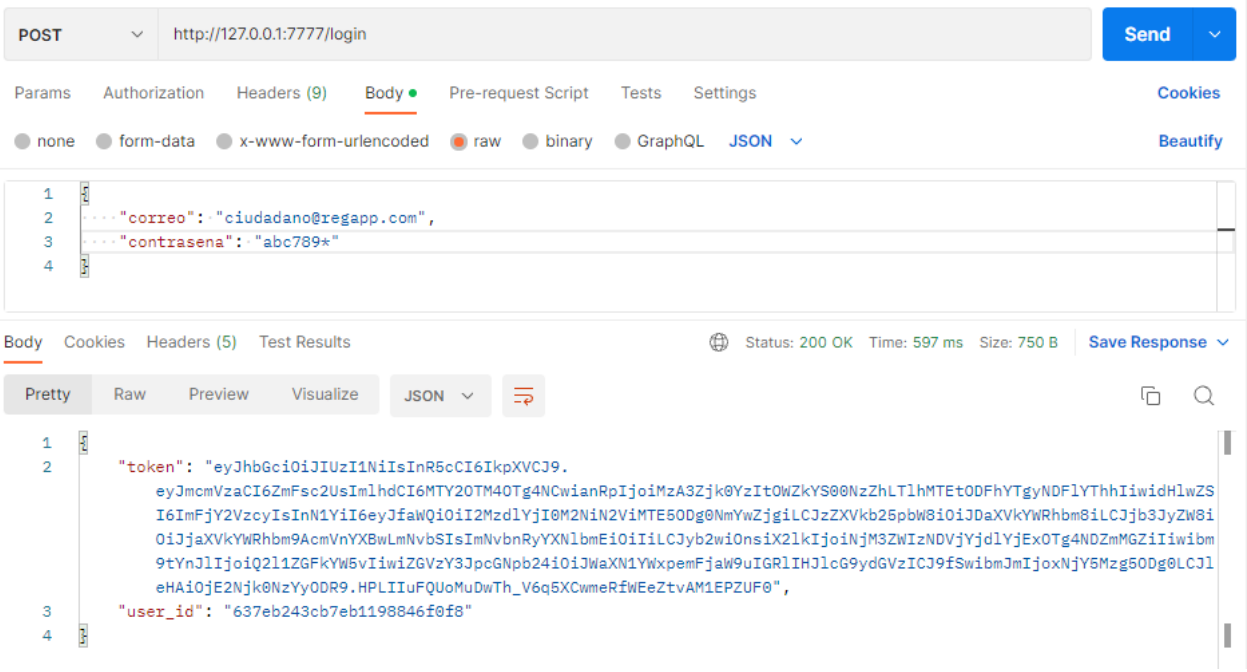
Luego le asignamos un rol a este usuario, en este caso solo tendrá permisos de lectura como se puede observar en la descripción



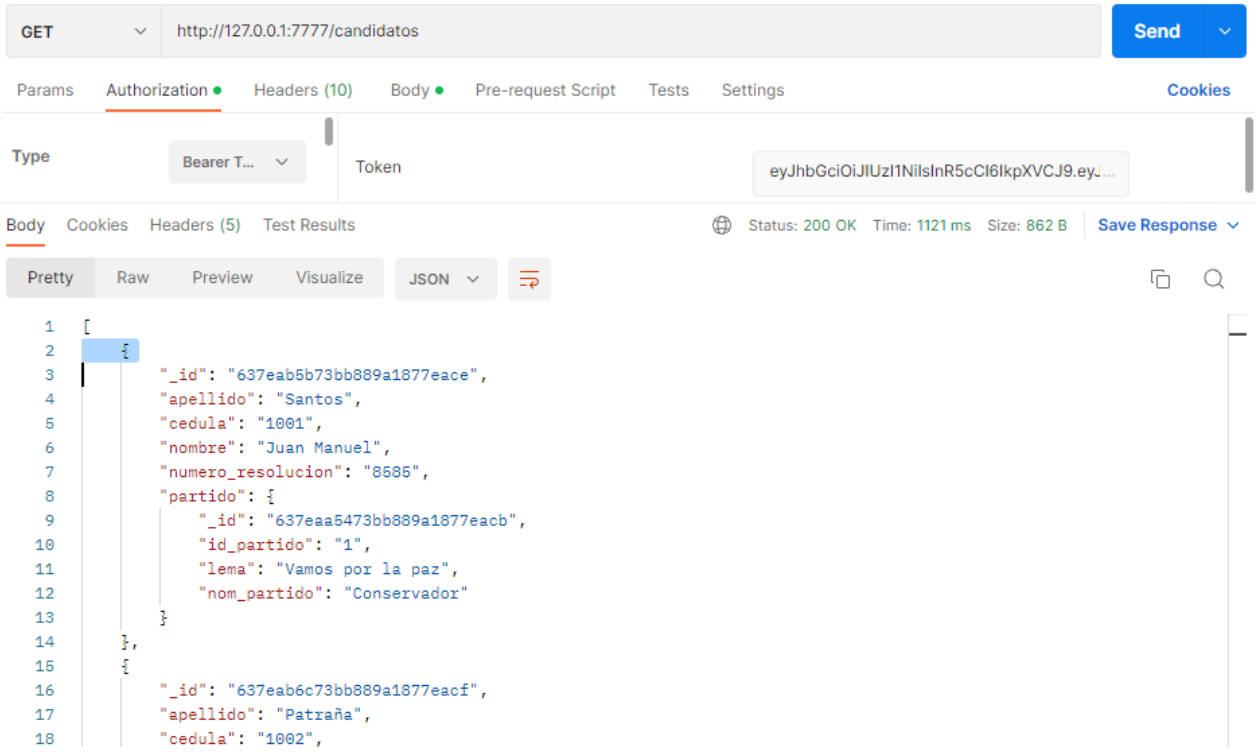
Después de tener su respectivo rol le asignamos algunos permisos para el acceso al backend de resultados, en este caso solo podrá ver la lista de candidatos o un candidato con su respectivo número de id:



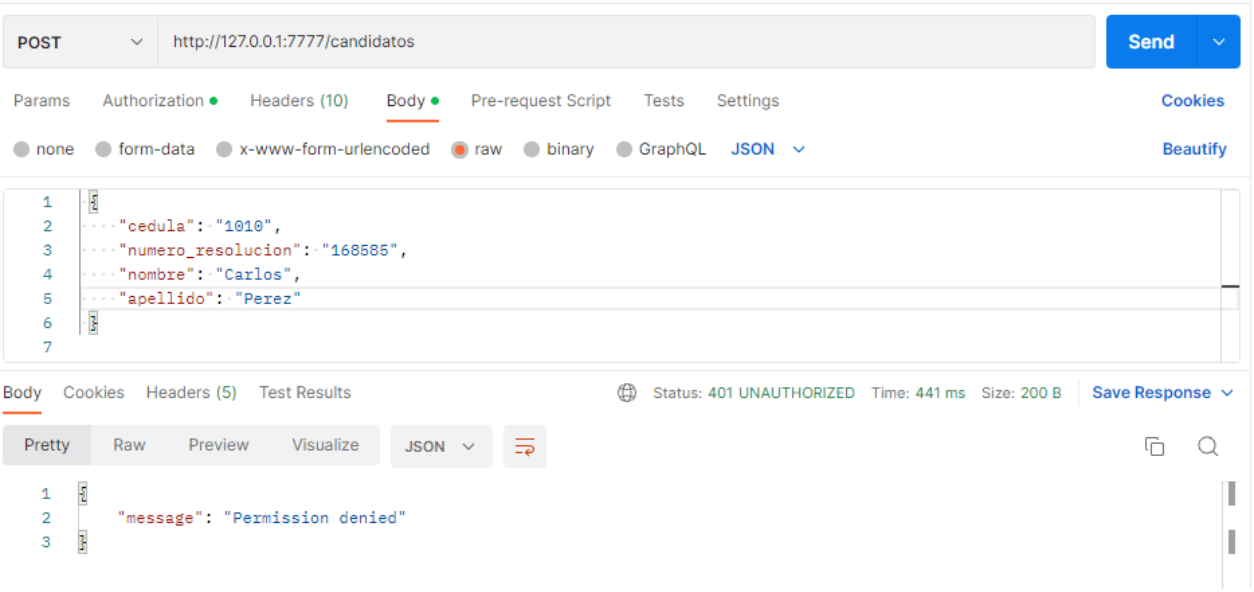
Ahora para la verificación de estos permisos, hacemos el respectivo login con el usuario ciudadano al cual se le asigna su respectivo token



Ya con el token generado para este usuario ingreso a la ruta que se le ha asignado como permisos-roles obteniendo la respuesta con el listado de candidatos almacenados en mongodb



Finalmente realizamos la prueba a través del método post que según lo establecido en el código me permite crear un nuevo candidato como ejemplo fijado en esta ruta, pero como este usuario no tiene permisos para acceder a esta ruta con su respectivo método, el permiso de acceso debe ser denegado



Formato de Informe de Retrospectiva

Nombre del equipo:	Grupo 3 - Luis Alberto Moscoso Herrera - Luis Fernando Cortes - Luis Orlando Moreno Cruz - Luis Camilo Rosero Grijalba
Sprint No.:	3 (Tres)

1. Resumen de los aspectos positivos y negativos del sprint:

Positivos • Conocer nuevas herramientas de trabajo como lo es Api Gateway que gestiona el ingreso autorizado y no autorizado a un sistema • Investigación sobre el tema a trabajar.
Negativos • Dificultad a la hora de sincronizar el grupo de trabajo

2. Propuesta (o propuestas) de cambio seleccionada(s) para el siguiente sprint:

• Afianzar el trabajo en equipo. • Profundizar en los temas con fuentes externas.