



UNIVERSIDAD  
NACIONAL  
DEL COMAHUE

JUNIO 2023

# Proyecto de Beca PPU - Poder Judicial Neuquén

Web Service Rest sobre  
plataforma XROAD

## INTEGRANTES:

Mariano Agustín Vergara Flores  
Maria Laura Pino

AÑO 2023

# ÍNDICE

<b>I. INTRODUCCIÓN</b>	
<i>Introducción</i>	<b>3</b>
<b>II. TECNOLOGIAS UTILIZADAS</b>	
<i>Tecnologias</i>	<b>4</b>
<b>III. INTRODUCCION A X-ROAD</b>	
<i>¿Que es X-Road?</i>	<b>5</b>
<i>Vision General</i>	<b>5</b>
<i>Ecosistema X-Road</i>	<b>6</b>
<i>Modelo Organizacional</i>	<b>8</b>
<i>Arquitectura</i>	<b>9</b>
<b>IV. INTRODUCCION A DOCKER</b>	
<i>¿Que es Docker?</i>	<b>13</b>
<i>¿Que beneficios tiene usar Docker?</i>	<b>14</b>
<b>V. ESTRUCTURA DE LA APLICACION</b>	
<i>Estructura</i>	<b>15</b>
<b>VI. VERIFICAR FUNCIONAMIENTO DE LOS WEB SERVICES</b>	
<i>Verificar Funcionamiento</i>	<b>16</b>
<b>VII. AGREGAR UN NUEVO WEB SERVICE</b>	
<i>Agregar nuevo web service</i>	<b>20</b>

# ÍNDICE

## VIII. DESPLIEGUE DE LA APLICACION

<i>Introducción</i>	<b>23</b>
---------------------	-----------

## IX. DOCKER COMPOSE

<i>¿Que es Docker Compose?</i>	<b>25</b>
--------------------------------	-----------

## X. ANEXO. CONSULTAS

<i>Rubricas</i>	<b>28</b>
<i>Socios</i>	<b>29</b>
<i>Antecedentes</i>	<b>30</b>
<i>Inscripcion y Sede</i>	<b>31</b>

## XI. ANEXO. DOCS API

<i>Rubricas</i>	<b>32</b>
<i>Socios</i>	<b>40</b>
<i>Antecedentes</i>	<b>44</b>
<i>Inscripcion y Sede</i>	<b>50</b>

# BECA PPU

Universidad nacional del comahue



## INTRODUCCION

El presente trabajo es un informe sobre la arquitectura, herramientas y tecnologías que se usaron para desarrollar e implementar los servicios WEB requeridos por el Poder Judicial de Neuquén

### Tecnologías Usadas

- Java 17
  - Framework Spring Boot 3
  - Maven
  - MariaDB (versión mayor a la 5)
  - Swagger 3
- IDE Utilizado (IntelliJ IDEA Community Edition)

## ACCESO AL PROYECTO

[https://github.com/marianvf10/Poder\\_Judicialv2](https://github.com/marianvf10/Poder_Judicialv2)

# BECA PPU

Universidad nacional del comahue



## TECNOLOGIAS UTILIZADAS



Se utilizo swagger para documentar cada uno de los servicios web



La base de datos utilizada para las pruebas fue brindada por el poder judicial



**POSTMAN**

Postman fue utilizado para hacer las pruebas en etapas de desarrollo y produccion



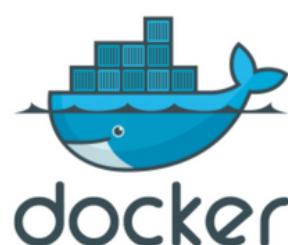
Se utilizo spring boot para desarrollar los servicios web



Se utilizo X-ROAD para el intercambio de informacion entre el Centro Pymes y el Poder Judicial



Se uso la plataforma GitHub para almacenar el repositorio de los servicios



Docker se utilizo para hacer el despliegue de la aplicacion



## 1. ¿Qué es X-ROAD?

X-Road® es un software de código abierto y una solución de ecosistema que proporciona un intercambio de datos unificado y seguro entre las organizaciones.

La idea básica de X-Road es que los miembros de un ecosistema intercambien datos a través de puntos de acceso (Servidores de Seguridad - Security Servers) que implementan las mismas especificaciones técnicas.

X-Road es un bien público digital verificado por la Alianza de Bienes Públicos Digitales, y se publica bajo la licencia de código abierto del MIT y está disponible de forma gratuita.

## 2. Vision General

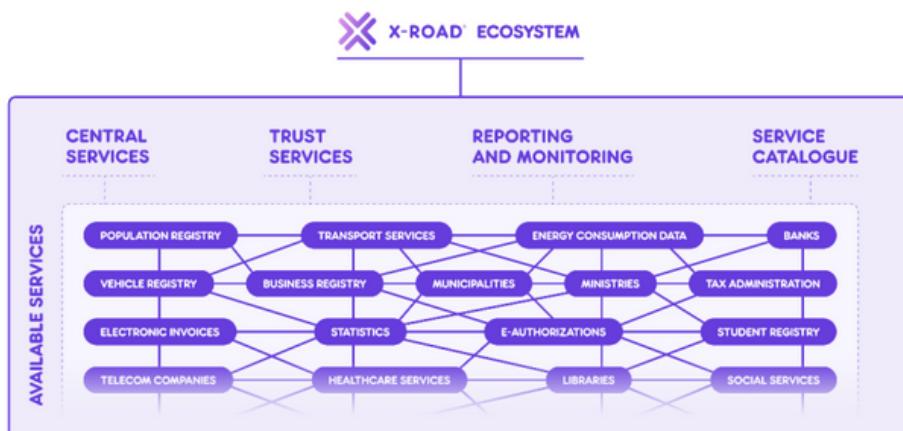
X-Road es una capa de intercambio de datos distribuida administrada centralmente entre sistemas de información que proporciona una forma estandarizada y segura de producir y consumir servicios.

X-Road implementa un conjunto de características estándar para apoyar y facilitar el intercambio de datos y garantiza la confidencialidad, integridad e interoperabilidad entre las partes del intercambio de datos:

- gestión de direcciones (address management)
- enrutamiento de mensajes (message routing)
- administración de permisos de acceso (access rights management)
- autenticación a nivel de organización (organization-level authentication)
- autenticación a nivel de máquina (machine-level authentication)
- cifrado a nivel de transporte (transport-level encryption)
- sellado de tiempo (time-stamping)
- firma digital de mensajes (digital signature of messages)
- Registro (logging)
- manejo de errores (error handling)

## 2.1 Ecosistema X-ROAD

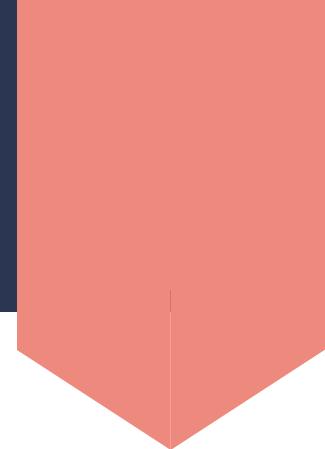
Un ecosistema X-Road es una comunidad de organizaciones que utilizan la misma instancia del software X-Road para producir y consumir servicios. El propietario del ecosistema, el Operador de X-Road, controla quién puede unirse al Ecosistema, y el propietario define las regulaciones y prácticas que el ecosistema debe seguir.



## 2.2 Red de Confianza (Trusted Network)

Incluso si el software X-Road es de código abierto, unirse a un ecosistema X-Road requiere pasar por un proceso de incorporación. Durante el proceso, la identidad de cada organización y punto de acceso técnico se verifica mediante certificados emitidos por una entidad de certificación (CA) de confianza. Las identidades se mantienen de forma centralizada, pero todos los datos se intercambian directamente entre un consumidor de servicios y un proveedor de servicios.

El enrutamiento de mensajes se basa en identificadores de nivel de servicio y organización que X Road asigna a las ubicaciones de red físicas de los servicios. Toda la evidencia relacionada con el intercambio de datos es almacenada localmente por las partes del intercambio de datos, y ningún tercero tiene acceso a los datos. El sellado de tiempo y la firma digital juntos garantizan el no repudio de los datos enviados a través de X-Road. Los registros proporcionados por X-Road se pueden utilizar en un procedimiento judicial como prueba.



## 2.3 Marco de Autorización

X-Road implementa un framework de autorización que se utiliza para administrar los permisos de acceso a los servicios. La administración de los permisos de acceso se basa en los identificadores de nivel de servicio y organización.

La idea clave de X-Road es que cada proveedor de servicios es propietario de sus datos y es responsable de administrar los permisos de acceso de sus servicios. En otras palabras, la publicación del servicio en X-Road no significa que el servicio sea accesible automáticamente para todas las organizaciones miembros de X-Road. Por lo general, los permisos de acceso se otorgan a nivel del sistema de información: un proveedor de servicios otorga a un sistema de información específico acceso a un servicio.

## 2.4 Monitoreo e informes

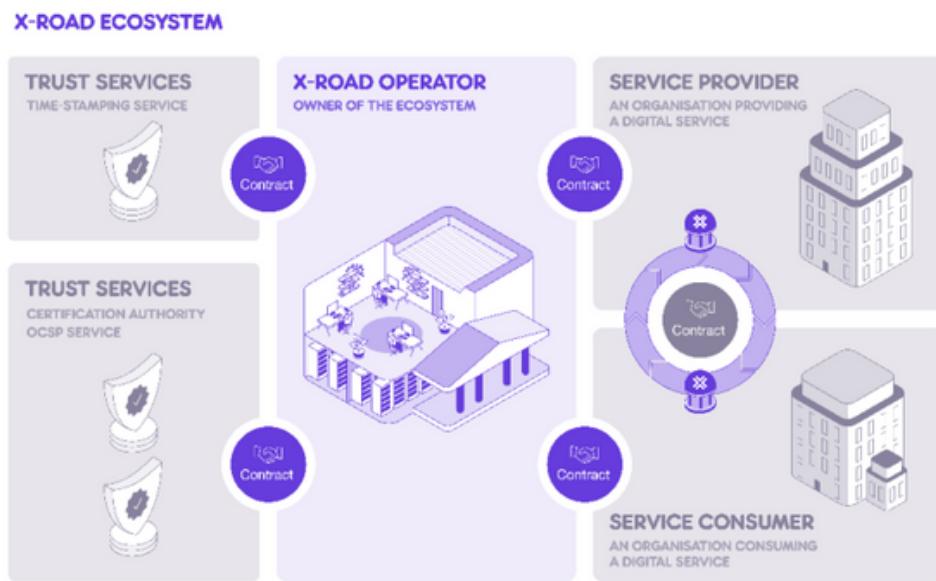
X-Road proporciona capacidades de monitoreo e informes que se pueden usar para recopilar datos de informes operativos e información de monitoreo técnico del ecosistema. La información se puede utilizar para medir el uso de servicios individuales, comprender dependencias y relaciones entre diferentes sistemas y servicios de información, monitorear el estado del servicio, monitorear las versiones de software X-Road utilizadas, etc. Cada organización miembro de X-Road puede acceder a sus propios datos, mientras que el operador de X-Road puede acceder a todos los datos de los miembros.

## 2.5 Intercambio transfronterizo de datos

X-Road proporciona soporte integrado para el intercambio de datos transfronterizos a través de la federación, lo que significa unir dos ecosistemas de X-Road. Los miembros de los ecosistemas federados pueden publicar y consumir servicios entre sí como si fueran miembros del mismo ecosistema. Es posible crear conexiones de federación con varios ecosistemas, pero no se admiten relaciones de federación transitivas. Un ecosistema no tiene una relación de federación con otro ecosistema con el que no esté directamente federado.

## 3. Modelo Organizacional

El ecosistema de X-Road consiste en un Operador de X-Road, organizaciones Miembros y Proveedor(es) de Servicios de Confianza

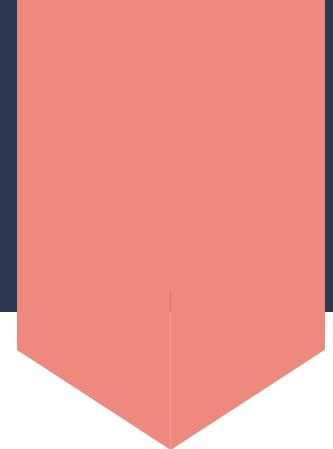


### 3.1 Operador de X-Road

Como propietario del ecosistema X-Road, el Operador es responsable de todos los aspectos de las operaciones. Las responsabilidades incluyen definir regulaciones y prácticas, aceptar nuevos miembros, brindar soporte a los Miembros y operar los componentes centrales del software X-Road.

### 3.2 Miembros de X-Road

Los Miembros de X-Road son organizaciones que se han unido al ecosistema y producen y/o consumen servicios con y de otros Miembros. Una organización miembro puede ser un proveedor de servicios, un consumidor de servicios o ambos. Las organizaciones pueden convertirse en miembros de un ecosistema completando el proceso de incorporación definido por el Operador. Además, los miembros deben tener acceso al componente técnico que se requiere para intercambiar mensajes a través de X-Road, el servidor de seguridad.



## 3.3 Proveedores de servicio de Confianza

Para que un ecosistema X-Road funcione se requieren dos tipos de servicios de confianza: 1) autoridad de sellado de tiempo (TSA) y 2) autoridad de certificación (CA). Los proveedores de servicios de confianza son organizaciones que brindan estos servicios. Los proveedores de servicios de confianza pueden ser terceros comerciales, o los servicios también pueden ser proporcionados y mantenidos por el Operador de X-Road.

## 4. Arquitectura

Técnicamente, el ecosistema de X-Road consiste en Servicios Centrales, Servidores de Seguridad, Sistemas de Información, TSA y CA

### 4.1 Servicios Centrales

Los servicios centrales consisten en el servidor central y el proxy de configuración. El Servidor Central contiene el registro de los miembros de X-Road y sus servidores de seguridad. Además, el servidor central contiene la directiva de seguridad de la instancia de X-Road que incluye una lista de entidades de certificación de confianza, una lista de autoridades de sellado de tiempo de confianza y parámetros de configuración. Tanto el registro de miembros como la directiva de seguridad se ponen a disposición de los servidores de seguridad a través del protocolo HTTP. Este conjunto distribuido de datos forma la configuración global que los servidores de seguridad utilizan para mediar los mensajes enviados a través de X-Road. El Operador de X-Road es responsable de operar el Servidor Central.

El Proxy de configuración es un componente opcional que se puede utilizar como proxy para publicar la configuración global en Servidores de seguridad para su descarga. El proxy de configuración primero descarga la configuración global del servidor central y, a continuación, la distribuye de forma segura. El proxy de configuración se puede utilizar para aumentar la disponibilidad del sistema mediante la creación de un origen de configuración adicional y reducir la carga en el servidor central. El operador de X-Road es responsable de operar el proxy de configuración

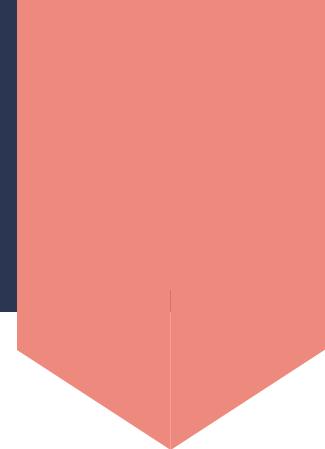
## 4.2 Servidor de Seguridad

El Servidor de Seguridad es el punto de entrada a X-Road, y es necesario tanto para producir como para consumir servicios a través de X-Road. El servidor de seguridad media las llamadas de servicio y las respuestas de servicio entre los sistemas de información. Éste encapsula los aspectos de seguridad de la infraestructura de X-Road: administración de claves para la firma y autenticación, envío de mensajes a través de un canal seguro, creación del valor de prueba para mensajes con firmas digitales, sellado de tiempo y registro. Para un consumidor de servicios y un sistema de información de proveedor de servicios, el Servidor de Seguridad ofrece un protocolo de mensajes basado en REST y soap. El protocolo es el mismo tanto para el cliente como para el proveedor de servicios, lo que hace que el Servidor de Seguridad sea transparente para las aplicaciones

El Servidor de Seguridad administra dos tipos de claves. Las claves de autenticación se asignan a un servidor de seguridad y se utilizan para establecer canales de comunicación criptográficamente seguros con otros servidores de seguridad. Las claves de firma se asignan a los clientes del Servidor de Seguridad y se utilizan para firmar los mensajes intercambiados. Una entidad de certificación de confianza emite certificados para las claves. Los certificados emitidos por otras entidades de certificación se consideran inválidos.

Para poder mediar mensajes, el Servidor de Seguridad debe tener una copia válida de la configuración global disponible todo el tiempo. El servidor de seguridad descarga la configuración global del servidor central con regularidad y utiliza una copia local mientras procesa los mensajes. El servidor de seguridad permanece operativo siempre que tenga una copia válida de la configuración global disponible localmente. Del mismo modo, la información de validez del certificado se descarga de la entidad de certificación y se almacena en caché localmente. El almacenamiento en caché permite que el Servidor de Seguridad funcione incluso cuando los orígenes de datos de configuración no están disponibles.

El Servidor Seguridad tiene un equilibrador de carga interno del lado del cliente y también admite el equilibrio de carga externo. El equilibrador de carga del lado del cliente es una característica integrada y proporciona alta disponibilidad. En su lugar, el equilibrio de carga externo proporciona alta disponibilidad y escalabilidad desde el punto de vista del rendimiento



## 4.3 Sistema de Informacion

El Sistema de Información produce y/o consume servicios a través de X-Road y es propiedad de un miembro de X-Road. X-Road admite el consumo y la producción de servicios REST y SOAP. Sin embargo, X-Road no proporciona conversiones automáticas entre diferentes tipos de mensajes y servicios.

Para un sistema de información al consumidor de servicios, el servidor de seguridad actúa como un punto de entrada a todos los servicios de X-Road. El consumidor puede descubrir a los miembros registrados de X-Road y sus servicios disponibles mediante el protocolo de metadatos de X-Road

Un sistema de información del proveedor de servicios implementa un servicio REST y/o SOAP y lo pone a disposición a través de X-Road. Los servicios REST existentes no requieren ningún cambio, ya que se pueden publicar tal cual. En su lugar, los servicios SOAP deben implementar el protocolo de mensajes X-Road para SOAP. Las descripciones de servicio de los servicios REST se definen mediante la especificación OpenAPI3 y las descripciones de servicio de los servicios SOAP se definen mediante WSDL. Los consumidores de servicios pueden descargar descripciones de servicios mediante el protocolo de metadatos X-Road

## 4.4 Autoridad de Sellado de Tiempo

Todos los mensajes enviados a través de X-Road tienen un sellado de tiempo y son registrados por el servidor de seguridad. El propósito del sellado de tiempo es certificar la existencia de elementos de datos en un momento determinado. La TSA proporciona un servicio de sellado de tiempo que el servidor de seguridad utiliza para marcar la hora de todas las solicitudes/resuestas entrantes/salientes. Solo se pueden utilizar los TSA de confianza definidos en el Servidor Central.

La autoridad de sellado de tiempo debe implementar el protocolo de sellado de tiempo admitido por X-Road. X-Road utiliza el sellado de tiempo por lotes, lo que reduce la carga del servicio de sellado de tiempo. La carga no depende del número de mensajes intercambiados a través de X-Road. En su lugar, depende del número de servidores de seguridad en el Ecosistema

## 4.5 Autoridad de Certificación

La Autoridad de Certificación (AC) emite certificados a los servidores de seguridad (certificados de autenticación) y a las organizaciones miembros de X-Road (certificados de firma). Los certificados de autenticación se utilizan para proteger la conexión entre dos servidores de seguridad. Los certificados de firma se utilizan para firmar digitalmente los mensajes enviados por los miembros de X-Road. Solo se pueden usar certificados emitidos por entidades de certificación de confianza definidas en el servidor central.

El servidor de seguridad comprueba la validez de los certificados de firma y autenticación a través del Protocolo de estado de certificados en línea (OCSP). Cada servidor de seguridad es responsable de consultar la información de validez de sus certificados y, a continuación, compartir la información con otros servidores de seguridad como parte del proceso de intercambio de mensajes. Solo los servidores de seguridad con certificados de firma y autenticación válidos pueden intercambiar mensajes con otros servidores de seguridad.

## ¿Qué es Docker?

La idea detrás de Docker es crear contenedores ligeros y portables para las aplicaciones software que puedan ejecutarse en cualquier máquina con Docker instalado, independientemente del sistema operativo que la máquina tenga por debajo, facilitando así también los despliegues.

Este concepto ya es antiguo, y viene de Linux, pero para hacer una analogía con el mundo real, vendría a ser como esos contenedores que suelen llevar los barcos de mercancías, que contienen distintos productos.

Es algo auto contenido en sí, que se puede llevar de un lado a otro de forma independiente, es portable.

Ahora, volviendo al software, para que podamos acceder como usuarios normales a una aplicación, dicha aplicación software necesita estar ejecutándose en una máquina, en un ordenador. Pero además, dependiendo del tipo de aplicación, dicho ordenador también necesita tener instaladas una serie de cosas para que la aplicación se ejecute correctamente: cierta versión de Java instalado, un servidor de aplicaciones (p.e tomcat, que es el software que realmente estará ejecutando mi aplicación y haciendo que pueda interactuar con ella).

Docker, permite meter en un contenedor (“una caja”, algo auto contenido, cerrado) todas aquellas cosas que mi aplicación necesita para ser ejecutada (Java, Maven, Tomcat...) y la propia aplicación. De esta manera, se puede llevar ese contenedor a cualquier máquina que tenga instalado Docker y ejecutar la aplicación sin tener que hacer nada más, ni preocuparse de qué versiones de software tiene instalada esa máquina, de si tiene los elementos necesarios para que funcione mi aplicación , de si son compatibles...etc.

La aplicación se ejecuta desde el contenedor de Docker, y dentro de él estarán todas las librerías y cosas que necesita dicha aplicación para funcionar correctamente.

.

## ¿Qué beneficios tiene usar Docker?

Docker es una herramienta diseñada para beneficiar tanto a desarrolladores, testers, como administradores de sistemas, en relación a las máquinas, a los entornos en sí donde se ejecutan las aplicaciones software, los procesos de despliegue, etc.

En el caso de los desarrolladores, el uso de Docker hace que puedan centrarse en desarrollar su código sin preocuparse de si dicho código funcionará en la máquina en la que se ejecutará.

Por ejemplo, sin utilizar Docker un posible escenario podría ser el siguiente (hay otras formas de solucionar este escenario, pero por poner un ejemplo claro):

- Pepe tiene en su ordenador instalado Java 8, y está programando una funcionalidad específica de la aplicación con algo que solo está disponible en esa versión de Java.
- José tiene instalado en su máquina Java 7, porque está en otro proyecto trabajando sobre otro código, pero Pepe quiere que José ejecute el código de su aplicación en su máquina. O José se instala Java 8, o la aplicación en su máquina fallará.

Este escenario desaparece con Docker. Para ejecutar la aplicación, Pepe se crea un contenedor de Docker con la aplicación, la versión 8 de Java y el resto de recursos necesarios, y se lo pasa a José.

José, teniendo Docker instalado en su ordenador, puede ejecutar la aplicación a través del contenedor, sin tener que instalar nada más.

Por eso Docker también es muy bueno para el testing, para tener entornos de pruebas. Por un lado, es muy sencillo crear y borrar un contenedor, además de que son muy ligeros, por lo que podemos ejecutar varios contenedores en una misma máquina (donde dicho contenedor tendría el entorno de nuestra aplicación: base de datos, servidor, librerías...). Por otro, un mismo contenedor funcionará en cualquier máquina Linux: un portátil, el ordenador de tu casa, máquinas alojadas en Amazon, tu propio servidor...

Esto además beneficia a la parte de sistemas, ya como los contenedores son más ligeros que las máquinas virtuales, se reduce el número de máquinas necesarias para tener un entorno.

## ESTRUCTURA DEL PROYECTO

Para el desarrollo de los servicios utilizamos SpringBoot, una tecnologia de Spring.

Con Spring Boot nos facilitamos las configuraciones como las dependencias de conexion con la base de datos, despliegue de servidor, entre otras.

El patron utilizado para organizar la estructura del proyecto se divide en distintas capas, lo cual nos permite separar la logica de negocios (entidades, servicios, repositorio) de los controladores que manejan las peticiones HTTP especificas.

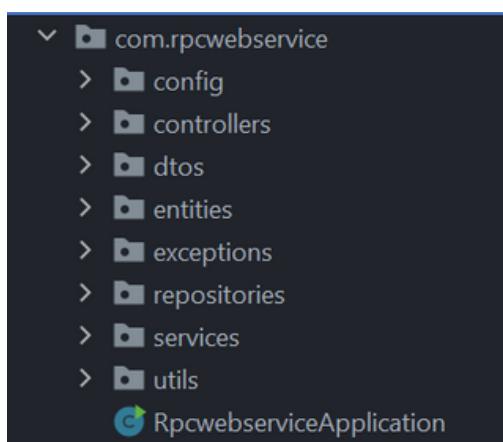
Tambien se utilizaron DTOs (Data Transfer Objects) debido a que nos facilitan mostrar la informacion requerida por cada uno de los servicios web.

El patrón DTO tiene como finalidad de crear un objeto plano (POJO) con una serie de atributos que puedan ser enviados o recuperados del servidor en una sola invocación, de tal forma que un DTO puede contener información de múltiples fuentes o tablas y concentrarlas en una única clase simple.

El paquete config se utiliza para almacenar la configuracion de OpenAPI que se utiliza para la documentacion de los servicios.

En el paquete de exceptions se almacenan distintos tipos de excepciones particulares para cada situacion

Por ultimo, el paquete utils es para utilidades, como por ejemplo, se utiliza un validador de cuit que se utiliza en el controlador a la hora de chequear que el formato y tipo de cuit sea el correcto



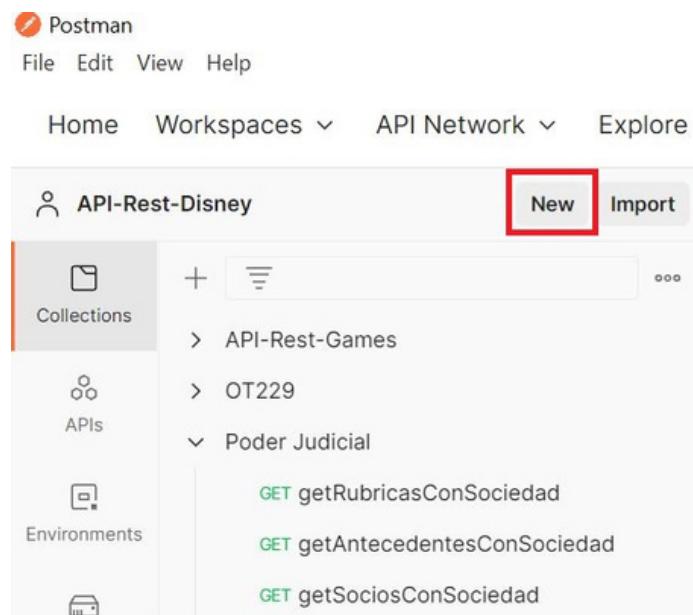
## Verificar Funcionamiento de los Web Services

Para el verificar el correcto funcionamiento de cada uno de los web services utilizamos POSTMAN.

Voy a mostrar un ejemplo de como verificar el funcionamiento del servicio web de rubricas (en el anexo de los consultas se encuentra la descripcion).

Primero que nada debemos ejecutar nuestra aplicacion y luego debemos ir a postman.

Una vez abierto POSTMAN debemos hacer click en donde dice "New" y seleccionar HTTP Request



# Beca PPU

Universidad nacional del comahue



Luego de hacer click, se nos abre la siguiente ventana

Untitled Request

Save |

GET  Enter request URL

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Response

  
Enter the URL and click Send to get a response

Donde dice GET lo que hacemos es seleccionar el tipo de petición HTTPS que estamos haciendo, como los web service que construí son todos de consulta, todos van a utilizar GET.

Del lado derecho de donde esta GET, se ubica la url de la aplicación del ambiente de desarrollo, por lo general cuando trabajamos con spring boot la url de desarrollo es localhost y el puerto por defecto es 8080. Ejemplo: <http://localhost:8080>

En la sección de Response es donde muestra el resultado de esa petición.

Luego del puerto, va el path del servicio web que estamos queriendo verificar, en el ejemplo de rubricas yo defini el siguiente path:

# Beca PPU

Universidad nacional del comahue



GET ▼ http://localhost:8080/api/sociedad\_rubricas Send ▾

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Response

  
Click Send to get a response

Por ultimo, luego del path lo que ubico es el parametro de busqueda que viene definido por las consultas en el documento de la especificacion. En este caso el parametro del busqueda es el CUIT. Por lo cual, debemos ubicar el "CUIT" sin guiones y puntos seguido del path para que el cuit se recupere a traves del mismo.

# Beca PPU

Universidad nacional del comahue



Con esto esto dicho, el resultado de esa consulta seria el siguiente:

The screenshot shows a Postman request for a GET API endpoint. The URL is `http://localhost:8080/api/sociedad_rubricas/30708012293`. The 'Body' tab is selected, showing a JSON response with the following data:

```
1 {
2     "numero_apertura": "UNO",
3     "numero_clausura": null,
4     "fecha_apertura": "2002-04-02T03:00:00.000+00:00",
5     "ru_Denominacion": "ACTAS DE ASAMBLEAS",
6     "sociedadId": 87
7 },
8 {
9     "numero_apertura": "UNO",
```

The response status is 200 OK, with a duration of 578 ms and a size of 3.77 KB.

El resultado se muestra en formato JSON en la seccion de Response. El codigo 200 OK es un codigo de HTTP que indica que la solicitud se completo exitosamente.

En el caso de que se ingrese un cuit que no exista en la base de datos se devuelve un codigo 404 que indica que la entidad que se estaba buscando no se encuentra en la base de datos

Si se llegara a ingresar un cuit invalido, el codigo de respuesta es un 400 que significa que el servidor no puede procesar la peticion debido a un error del cliente, en este caso por el cuit invalido.

Estos codigos de error se utilizan para que la API Rest nos brinde informacion apropiada para cada peticion

## ¿Cómo agregar un nuevo web service?

Primero que nada se debe definir la consulta y en base a esta, lo que hacemos es definir una clase DTO para mostrar solo los campos que la consulta solicita. Luego de esto, lo que hacemos es definir en el repositorio de la entidad correspondiente la consulta deseada.

Hay varias maneras de recuperar informacion de la base de datos con spring, una de ellas es a traves de los metodos de consulta.

### Supported query method subject keywords

The following table lists the subject keywords generally supported by the Spring Data repository query derivation mechanism to express the predicate. Consult the store-specific documentation for the exact list of supported keywords, because some keywords listed here might not be supported in a particular store.

Table 9. Query subject keywords

Keyword	Description
<code>find...By</code> , <code>read...By</code> , <code>get...By</code> , <code>query...By</code> , <code>search...By</code> , <code>stream...By</code>	General query method returning typically the repository type, a <code>Collection</code> or <code>Streamable</code> subtype or a result wrapper such as <code>Page</code> , <code>GeoResults</code> or any other store-specific result wrapper. Can be used as <code>findBy...</code> , <code>findMyDomainTypeBy...</code> or in combination with additional keywords.
<code>exists...By</code>	Exists projection, returning typically a <code>boolean</code> result.
<code>count...By</code>	Count projection returning a numeric result.
<code>delete...By</code> , <code>remove...By</code>	Delete query method returning either no result ( <code>void</code> ) or the delete count.
<code>...First&lt;number&gt;...</code> , <code>...Top&lt;number&gt;...</code>	Limit the query results to the first <code>&lt;number&gt;</code> of results. This keyword can occur in any place of the subject between <code>find</code> (and the other keywords) and <code>by</code> .

Referencia:

[https://docs.spring.io/spring-](https://docs.spring.io/spring-data/jpa/docs/current/reference/html/#repositories.query-methods.details)

[data/jpa/docs/current/reference/html/#repositories.query-methods.details](https://docs.spring.io/spring-data/jpa/docs/current/reference/html/#repositories.query-methods.details)

Una vez que tenemos la consulta lo que hacemos es elegir el tipo de entidad que va a retornar la misma, si definimos que la consulta retorna un tipo de dato DTO, tenemos que tener sumo cuidado a la hora de definir los atributos en el DTO, es decir, el DTO debe tener los mismos atributos que define la consulta, para que el mapeo entre la entidad retornada y el DTO sea exitoso

# Beca PPU

Universidad nacional del comahue



```
public interface RubricaRepository extends CrudRepository<Rubrica, Integer> {  
  
    1 usage  ↗ mariano  
    List<RubricaDTO> findBySociedadCuit(String cuit, Sort sort);  
}
```

Luego de esto, lo que hacemos es crear el servicio e injectamos el repositorio para poder llamarlo y recuperar la consulta.

```
@Service  
public class RubricaService {  
  
    1 usage  
    @Autowired  
    RubricaRepository rubricaRepository;  
  
    1 usage  ↗ mariano  
    @Transactional  
    public List<RubricaDTO> getSociedadRubricasByCuit(String cuit){  
        return rubricaRepository.findBySociedadCuit(cuit, Sort.by( ...properties: "apertura").ascending().and(Sort.by( ...properties: "denominacion").ascending()));  
    }  
}
```

Por ultimo lo que hacemos es crear el controlador para poder manejar las peticiones HTTP.

```
@GetMapping("/sociedad_rubricas/{cuit}")  
public ResponseEntity<List<RubricaDTO>> getRubricasConSociedad(@PathVariable("cuit") String cuit) {  
    List<RubricaDTO> rubricas;  
  
    String cuitFormateado = Validador.validarCuit(cuit);  
    rubricas = rubricaService.getSociedadRubricasByCuit(cuitFormateado);  
    if (rubricas.isEmpty()) {  
        throw new ResourceNotFoundException("No se encontró sociedad con el cuit ingresado");  
    }  
    return new ResponseEntity<>(rubricas, HttpStatus.OK);  
}
```

Con la notacion @GetMapping indicamos que se espera una peticion de tipo GET, luego escribimos la url que queremos definir para esa peticion, por ultimo en el encabezado indicamos que el parametro de busqueda lo recuperamos del path (notacion @PathVariable), que en este caso es el CUIT

## ¿Cómo hacer el despliegue de la aplicación?

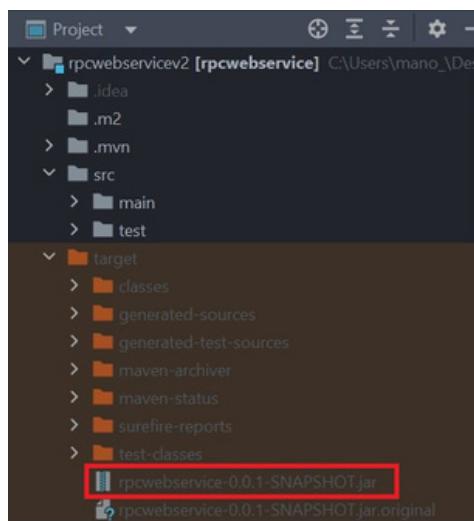
Primero que nada debemos instalar y configurar Docker Desktop en nuestra maquina.

<https://docs.docker.com/engine/install/>

Una vez que docker este instalado y ejecutandose en nuestra maquina, debemos crear el ejecutable de nuestra aplicacion. Para hacer esto abrimos la terminal en el IDE que estemos utilizando y escribimos el siguiente comando

```
PS C:\Users\mano_\Desktop\ProyectoPoderJudicial\Codigo\rpcwebservicev2> ./mvnw clean install
```

Luego de ejecutar este comando se va a crear una carpeta denominada "target" en el directorio de nuestro proyecto, dentro de esta carpeta se va a crear el ejecutable .jar de nuestra aplicacion.



Con este archivo ya podemos crear el Dockerfile que utiliza docker para leer y ejecutar nuestra aplicacion.

# Beca PPU

Universidad nacional del comahue



Lo que hacemos es crear un nuevo archivo denominado Dockerfile en el directorio del proyecto y escribimos los comandos para crear la imagen de nuestra aplicación

```
FROM openjdk:17
ARG JAR_FILE=target/*.jar
COPY ${JAR_FILE} app.jar
ENTRYPOINT ["java","-jar","/app.jar"]
```

1. **FROM openjdk:17.** Este comando indica la imagen base que se utilizará para construir la nueva imagen. En este caso, se está utilizando la imagen openjdk:17, que es una imagen oficial de OpenJDK versión 17. Esto significa que la imagen de Docker resultante tendrá instalado Java Development Kit (JDK) 17.
2. **ARG JAR\_FILE=target/\*.jar.** Este comando define una variable llamada JAR\_FILE y le asigna el valor target/\*.jar. La variable JAR\_FILE se utiliza posteriormente para copiar el archivo JAR de la aplicación en la imagen de Docker. En este caso, se espera que exista un archivo JAR en la carpeta target del contexto de construcción.
3. **COPY \${JAR\_FILE} app.jar.** Aquí se copia el archivo JAR de la aplicación en la imagen de Docker. Se utiliza la variable JAR\_FILE definida anteriormente para especificar la ruta del archivo JAR en el contexto de construcción. El archivo JAR se copiará en la ubicación /app.jar dentro de la imagen de Docker.
4. **ENTRYPOINT ["java","-jar","/app.jar"].** Este comando establece el punto de entrada (entrypoint) para la imagen de Docker. Cuando se ejecute un contenedor basado en esta imagen, se ejecutará el comando java -jar /app.jar. Esto significa que el contenedor ejecutará el archivo JAR de la aplicación utilizando el intérprete de Java



## Docker Compose

Es una herramienta para definir y ejecutar aplicaciones Docker multicontenedor que permite simplificar el uso de Docker a partir de archivos YAML, de esta forma es mas sencillo crear contenedores que se relacionen entre sí, conectarlos, habilitar puertos, volumenes, etc. Nos permite lanzar un solo comando para crear e iniciar todos los servicios desde su configuración(YAML), esto significa que puedes crear diferentes contenedores y al mismo tiempo diferentes servicios en cada contenedor, integrarlos a un volumen común e iniciarlos y/o apagarlos, etc. Este es un componente fundamental para poder construir aplicaciones y microservicios.

Docker-Compose funciona en todos los entornos: production, staging, development, testing, así como flujos de trabajo basados en Continuous Integration(CI).

La idea de utilizar Docker Compose es poder facilitar la creacion y configuracion de cada contenedor, es decir el contenedor que contiene la aplicacion y el contenedor que tiene la base de datos para que se conecten de forma automatica.

## ¿Como usar Docker Compose?

Lo que tenemos es crear un archivo de configuracion con la extension .yml En el mismo debemos escribir la configuracion de nuestros contenedores que serian la imagen que se va utilizar para cada uno de los contenedores, los puertos que se van a utilizar, los volumenes que se van a utilizar y las variables de entorno a utilizar.

Por otro lado, se necesita crear un archivo .env para almacenar las variables de entorno correspondientes, y luego llamar a este archivo desde el docker-compose.yml

Ejemplo

# BECA PPU

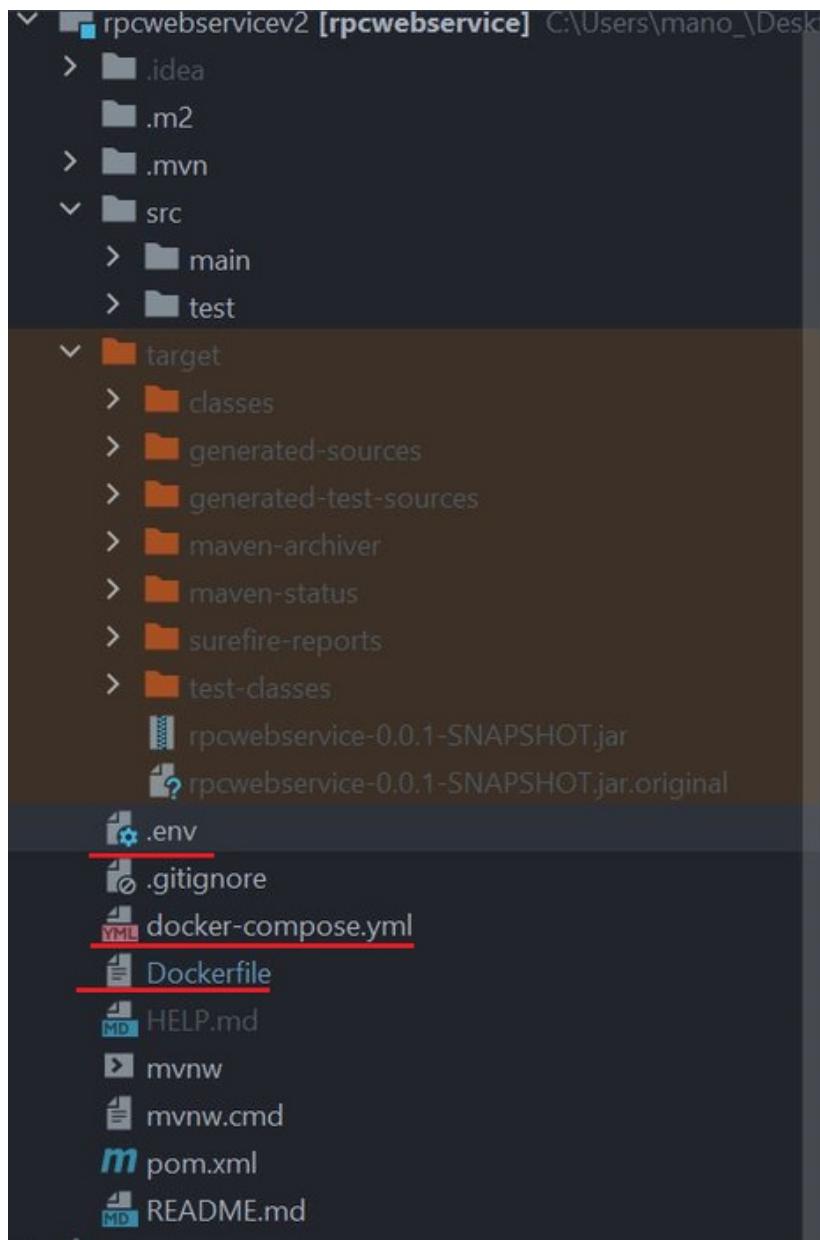
Universidad nacional del comahue



```
1  version: "3.8"
2
3  services:
4    mariadb:
5      image: mariadb
6      restart: unless-stopped
7      env_file: ./.env
8      environment:
9        - MARIADB_ROOT_PASSWORD=$MARIADB_ROOT_PASSWORD
10       - MARIADB_DATABASE=$MARIADB_DATABASE
11      ports:
12        - $MARIADB_LOCAL_PORT:$MARIADB_DOCKER_PORT
13      volumes:
14        - db:/var/lib/mariadb
15
16  app:
17    depends_on:
18      - mariadb
19    build: ./rpc_registro
20    restart: on-failure
21    env_file: ./.env
22    ports:
23      - $SPRING_LOCAL_PORT:$SPRING_DOCKER_PORT
24    environment:
25      SPRING_APPLICATION_JSON: '{
26        "spring.datasource.url" : "jdbc:mariadb://mariadb:$MARIADB_DOCKER_PORT/$MARIADB_DATABASE?useSSL=false",
27        "spring.datasource.username" : "$MARIADB_USER",
28        "spring.datasource.password" : "$MARIADB_ROOT_PASSWORD",
29        "spring.jpa.hibernate.ddl-auto" : "none"
30      }'
31
32    volumes:
33      - .m2:/root/.m2
34    stdin_open: true
35    tty: true
36
37  volumes:
38    db:
```

# Beca PPU

Universidad nacional del comahue



Una vez que tenemos estos 3 archivos creados, podemos proceder a ejecutar el comando para correr nuestra aplicación en el entorno de docker. Abrimos la terminal y ejecutamos el siguiente comando

```
PS C:\Users\mano_\Desktop\ProyectoPoderJudicial\Codigo\rpcwebservicev2> docker-compose up
```

# Beca PPU

Universidad nacional del comahue



## Consulta Rubricas

```
select distinct r.id_sociedad, r.apertura as Fecha_apertura, r.denominacion  
as Ru_Denominacion,r.numero_apertura, r.numero_clausura from  
rpc_registro.sociedades s left join rpc_registro.rubricas r on s.id =  
r.id_sociedad where s.cuit = 30708012293
```

Resultado:

	Fecha Apertura	Denominacion	# Apertura	# Clausura
02/04/2002	INVENTARIO Y BALANCES		UNO	
02/04/2002	ACTAS DE ASAMBLEAS		UNO	
02/04/2002	ACTAS DE DIRECTORIO	UNO	UNO	
02/04/2002	DEPOSITO DE ACCIONES Y REGISTRO DE ASISTENCIA			UNO
02/04/2002	REGISTRO DE ACCIONISTAS		UNO	
10/04/2002	DIARIO	UNO		
14/02/2003	IVA COMPRA	S	UNO	
14/02/2003	IVA VENTAS		UNO	
10/12/2007	INVENTARIO Y BALANCES	DOS	UNO	
30/04/2008	DIARIO	DOS	UNO	
13/11/2009	INVENTARIO Y BALANCES		CUATRO	CUATRO
13/11/2009	IVA COMPRA	DOS		
28/12/2010	IVA VENTAS	DOS		
30/12/2010	DIARIO	5	4	
09/02/2012	INVENTARIO Y BALANCES	TRES	DOS	
09/02/2012	DIARIO	SEIS	CINCO	
23/02/2012	DIARIO	SIETE	SEIS	
12/02/2015	DIARIO	OCHO	SIETE	
13/11/2009	DIARIO	4	3	
30/07/2009	DIARIO	3	2	
05/02/2016	INVENTARIO Y BALANCES	4	3	
30/10/2020	INVENTARIO Y BALANCES	CINCO	4	
30/10/2020	DIARIO	NUEVE	OCHO	
04/08/2021	ACTAS DE DIRECTORIO		DOS	

# Beca PPU

Universidad nacional del comahue



## Consulta Socios

```
select s.iddenominacion, tp.nombre as Participacion, pj.denominacion deno,
p2.nombre, p2.apellido, p2.numero_documento, p2.sexo
from
rpc_registro.sociedades s left join rpc_registro.socios s2 on s.id =
s2.id_sociedad left JOIN rpc_registro.personas p2 on s2.id_persona = p2.id
left JOIN rpc_registro.personas_juridicas pj on s2.id_persona_juridica =
pj.id or s2.id_sociedad = pj.id_sociedad left JOIN
rpc_registro.tipos_participacion tp on s2.id_tipo_participacion = tp.id
WHERE s2.activo = 1 AND where s.cuit = 30708012293
```

Resultado:

### Socios

Nombre o Denominacion	Documento	Participacion	Cuil/Cuit
DIEGO EDUARDO MANFIO	D.N.I. 23384618	4000 Acciones	20233846180
INGENIERIA SIMA S.A. REPRESENT. POR SILVEIRO ALBERTO MANFIO	D.N.I. 10.909.318	S.A.	CUIT: 16000 Acciones

# BECA PPU

Universidad nacional del comahue



## Consulta Antecedentes

```
SELECT
    an.fecha,
    descripcion,
    tan.nombre TipoAntecedente,
    concat(i.numero,'/',i.tomo_anio) Inscripcion
FROM
    rpc_registro.antecedentes an inner join
    rpc_registro.tipos_antecedentes tan on an.id_tipo_antecedente = tan.id
inner JOIN
    rpc_registro.inscripciones i on i.id = an.id_inscripcion inner JOIN
    rpc_registro.sociedades s on s.id = an.id_sociedad
WHERE s.cuit = '30708012293'
order by fecha
```

Resultado:

Fecha	Descripción	Tipo	Inscripción
04/03/2021	DC -Inscripción cambio de dirección de la sede social (Acta de Directorio del 02/10/2019)	Inscripción Actas	13/2021
12/02/2021	DC-Inscripción designación de directores (Acta de Asamblea Gral. ordinaria del 07/09/2020)	Inscripción Actas	5/2021
29/10/2019	DC--- INSC AUMENTO Y REDUCCION DEL CAPITAL, MODIFICACION DEL ART 6° Y LA APROBACION DE SU TEXTO ORDENADO (ACTAS DEL 19/10/17, 30/10/18, 28/02/19) --- el artículo 6° se refiere al capital---	Inscripción Aumento de Capital	130/2019
01/11/2018	INSCRIPCION CAMBIO DE DIRECCION DE LA SEDE SOCIAL	Inscripción Actas	103/2018
07/09/2017	INSC. Aumentos de capital y modificación de los Arts. 6°, 9°, 12° y 19° (ref al capital, acciones, administración y cierre de ejercicio respectivamente) - Instrumentos del 24/04/08, 04/07/11, 15/09/11, 03/04/12, 09/04/12, 19/11/14, 22/04/15 y 23/12/15	Inscripción Modificación de Contrato Social	86/2017
14/12/2009	INSCRIPCIÓN ACTAS, ACTA DE ASAM. GRAL. ORDINARIA DE FECHA 07/09/2.007-POR LA CUAL SE APRUEBA LA DOC. PREV. EN EL ART.234, INC. 1), LEY 19550 CORRESP. AL EJ. ECONOMICO N°7 FINALIZADO EL 30/06/07, AUMENTO DEL CAPITAL Y MOD. ART.6°(CAPITAL)	Inscripción Actas	135/2009
04/03/2009	INSCRIPCIÓN DE LA RENUNCIA DE LOS MIEMBROS DEL DIRECTORIO, LA DESIGNACIÓN DE DIRECTORES, LA MODIFICACIÓN DEL ART. 1° POR CAMBIO DE DENOMINACIÓN POR 'SAN JORGE PETROLEUM S.A.', LA MODIFICACIÓN DEL ART. 2° REFERIDO AL DOMICILIO Y LA DENUNCIA DE LA SEDE SOCIAL	Inscripción Modificación de Contrato Social	19/2009
12/09/2005	INSCRIPCIÓN ACTAS, ACTA DE ASAMBLEA GENERAL ORDINARIA (05/09/03 INCLUYE APROB BCE AL 30/06/03)	Inscripción Actas	80/2005
11/02/2004	INSCRIPCIÓN MODIFICACIÓN DE CONTRATO SOCIAL, SE MODIFICARON LOS ARTICULOS 6°(CAPITAL), 9° (ACCIONES) Y 12°(ADMINISTRACION).	Inscripción Modificación de Contrato Social	1/2004
17/11/2003	INSCRIPCIÓN ACTA DE ASAMBLEA GENERAL ORDINARIA (06/09/02)-POR LA CUAL SE APRUEBA EL EJERCICIO ECONOMICO N°2 Y SE DESIGNA EL DIRECTORIO- Y ACTA DE DIRECTORIO N° 6 (06,09,02 IDISTRIB DE CARGOS-	Inscripción Actas	85/2003
23/10/2003	INSCRIPCIÓN ACTAS, ACTA DE ASAMBLEA GENERAL ORDINARIA-POR LA CUAL SE APRUEBA EL EJERCICIO ECONOMICO N°1-	Inscripción Actas	73/2003
29/12/2000	INSCRIPCIÓN CONTRATO SOCIAL Y ESCRITO QUE DENUNCIA EL DOMICILIO DE LA SEDE SOCIAL (NO TIENE FIRMA CERTIFICADA)	Inscripción Contrato Social	337/2000

# Beca PPU

Universidad nacional del comahue



## Consulta Inscripcion y Sede

```
select distinct s.denominacion, s.cuit, ts.siglas as TipoSociedad,  
s.fecha_instrumento,  
MIN(i.fecha) as Fecha_Inscripcion, i.numero,  
i.folio_desde, i.folio_hasta, i.tomo_numero, i.tomo_anio,  
ss.calle, ss.numero, ss.otros as observaciones, l.nombre as Localidad,  
p.nombre as Provincias  
from  
rpc_registro.sociedades s left join  
rpc_registro.expedientes e on s.id = e.id_sociedad inner JOIN  
rpc_registro.inscripciones i on e.id = i.id_expediente inner JOIN  
rpc_registro.tipos_sociedad ts on s.id_tipo_sociedad = ts.id  
inner JOIN  
rpc_registro.sede_social ss on s.id = ss.id_sociedad inner JOIN  
rpc_registro.localidades l on ss.id_localidad = l.id inner JOIN  
rpc_registro.provincias p on l.id_provincia = p.id  
WHERE  
s.cuit = '30708012293'  
and ss.validado = 1  
order by s.id, s.denominacion asc  
Resultado:
```

### Inscripcion/sede

Denominacion: SAN JORGE PETROLEUM S.A. CUIT: 30708012293

Inscripcion: Nro: 337 Folio: 2004/2010 Tomo: XIII/2000 - SOCIEDADES

Fecha Inscripcion: 29/12/2000 Fecha Instrumento: 06/09/2000

Duracion: 99 AÑOS

Sede Social: HIPOLITO YRIGOYEN 135 TRANSITORIAS (CAMBIO DE DIRECCIÓN POR ACTA DE DIRECTORIO DEL 02/10/2019, INSC. 13/2021) NEUQUEN  
NEUQUEN



## Documentacion Web Service de Rubricas

Servicio Web: “/sociedad\_rubricas” en X-Road

Url de la fuente Autentica

<https://xroad.jusneuquen.gov.ar/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/>

Entorno	Metodo	Service Code	Punto Acceso	Ejemplo
Test	GET	desa	api/sociedad_rubricas	<a href="https://&lt;url xroad local&gt;/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/desa/sociedad_rubricas">https://&lt;url xroad local&gt;/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/desa/sociedad_rubricas</a>
Produccion	GET	prod	api/sociedad_rubricas	<a href="https://&lt;url xroad local&gt;/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/prod/sociedad_rubricas">https://&lt;url xroad local&gt;/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/prod/sociedad_rubricas</a>

El cuit se debe enviar a traves de la URL. Luego del punto de acceso va el cuit, sin guiones ni puntos.

Formato	Ejemplo
String	30708012293



## Response

Status	Metodo
200	[ { "numero_apertura": "UNO", "numero_clausura": null, "ru_Denominacion": "ACTAS DE ASAMBLEAS", "fecha_apertura": "2002-04-02T03:00:00.000+00:00", "sociedadId": 87 }, { "numero_apertura": "UNO", "numero_clausura": "UNO", "ru_Denominacion": "ACTAS DE DIRECTORIO", "fecha_apertura": "2002-04-02T03:00:00.000+00:00", "sociedadId": 87 }, { "numero_apertura": "UNO", "numero_clausura": null, "ru_Denominacion": "DEPOSITO DE ACCIONES Y REGISTRO DE ASISTENCIA", "fecha_apertura": "2002-04-02T03:00:00.000+00:00", "sociedadId": 87 }, { "numero_apertura": "UNO", "numero_clausura": null, "ru_Denominacion": "INVENTARIO Y BALANCES", "fecha_apertura": "2002-04-02T03:00:00.000+00:00", "sociedadId": 87 }]

# Beca PPU

Universidad nacional del comahue



```
{  
    "numero_apertura": "UNO",  
    "numero_clausura": null,  
    "ru_Denominacion": "REGISTRO DE ACCIONISTAS",  
    "fecha_apertura": "2002-04-02T03:00:00.000+00:00",  
    "sociedadId": 87  
},  
{  
    "numero_apertura": "UNO",  
    "numero_clausura": null,  
    "ru_Denominacion": "DIARIO",  
    "fecha_apertura": "2002-04-10T03:00:00.000+00:00",  
    "sociedadId": 87  
},  
{  
    "numero_apertura": "UNO",  
    "numero_clausura": null,  
    "ru_Denominacion": "IVA COMPRAS",  
    "fecha_apertura": "2003-02-14T03:00:00.000+00:00",  
    "sociedadId": 87  
},  
{  
    "numero_apertura": "UNO",  
    "numero_clausura": null,  
    "ru_Denominacion": "IVA VENTAS",  
    "fecha_apertura": "2003-02-14T03:00:00.000+00:00",  
    "sociedadId": 87  
},  
{  
    "numero_apertura": "DOS",  
    "numero_clausura": "UNO",  
    "ru_Denominacion": "INVENTARIO Y BALANCES",  
    "fecha_apertura": "2007-12-10T03:00:00.000+00:00",  
    "sociedadId": 87  
},  
}
```

# Beca PPU

Universidad nacional del comahue



```
{  
    "numero_apertura": "3",  
    "numero_clausura": "2",  
    "ru_Denominacion": "DIARIO",  
    "fecha_apertura": "2009-07-30T03:00:00.000+00:00",  
    "sociedadId": 87  
},  
{  
    "numero_apertura": "4",  
    "numero_clausura": "3",  
    "ru_Denominacion": "DIARIO",  
    "fecha_apertura": "2009-11-13T03:00:00.000+00:00",  
    "sociedadId": 87  
},  
{  
    "numero_apertura": "CUATRO",  
    "numero_clausura": "CUATRO",  
    "ru_Denominacion": "INVENTARIO Y BALANCES",  
    "fecha_apertura": "2009-11-13T03:00:00.000+00:00",  
    "sociedadId": 87  
},  
{  
    "numero_apertura": "DOS",  
    "numero_clausura": null,  
    "ru_Denominacion": "IVA COMPRAS",  
    "fecha_apertura": "2009-11-13T03:00:00.000+00:00",  
    "sociedadId": 87  
},  
{  
    "numero_apertura": "DOS",  
    "numero_clausura": null,  
    "ru_Denominacion": "IVA VENTAS",  
    "fecha_apertura": "2010-12-28T03:00:00.000+00:00",  
    "sociedadId": 87  
},
```

# Beca PPU

Universidad nacional del comahue



```
{  
    "numero_apertura": "5",  
    "numero_clausura": "4",  
    "ru_Denominacion": "DIARIO",  
    "fecha_apertura": "2010-12-30T03:00:00.000+00:00",  
    "sociedadId": 87  
},  
{  
    "numero_apertura": "SEIS",  
    "numero_clausura": "CINCO",  
    "ru_Denominacion": "DIARIO",  
    "fecha_apertura": "2012-02-09T03:00:00.000+00:00",  
    "sociedadId": 87  
},  
{  
    "numero_apertura": "TRES",  
    "numero_clausura": "DOS",  
    "ru_Denominacion": "INVENTARIO Y BALANCES",  
    "fecha_apertura": "2012-02-09T03:00:00.000+00:00",  
    "sociedadId": 87  
},  
{  
    "numero_apertura": "SIETE",  
    "numero_clausura": "SEIS",  
    "ru_Denominacion": "DIARIO",  
    "fecha_apertura": "2012-02-23T03:00:00.000+00:00",  
    "sociedadId": 87  
},  
{  
    "numero_apertura": "OCHO",  
    "numero_clausura": "SIETE",  
    "ru_Denominacion": "DIARIO",  
    "fecha_apertura": "2015-02-12T03:00:00.000+00:00",  
    "sociedadId": 87  
},
```

# Beca PPU

Universidad nacional del comahue



```
[  
  {  
    "numero_apertura": "4",  
    "numero_clausura": "3",  
    "ru_Denominacion": "INVENTARIO Y BALANCES",  
    "fecha_apertura": "2016-02-05T03:00:00.000+00:00",  
    "sociedadId": 87  
  },  
  {  
    "numero_apertura": "NUEVE",  
    "numero_clausura": "OCHO",  
    "ru_Denominacion": "DIARIO",  
    "fecha_apertura": "2020-10-30T03:00:00.000+00:00",  
    "sociedadId": 87  
  },  
  {  
    "numero_apertura": "CINCO",  
    "numero_clausura": "4",  
    "ru_Denominacion": "INVENTARIO Y BALANCES",  
    "fecha_apertura": "2020-10-30T03:00:00.000+00:00",  
    "sociedadId": 87  
  },  
  {  
    "numero_apertura": "DOS",  
    "numero_clausura": null,  
    "ru_Denominacion": "ACTAS DE DIRECTORIO",  
    "fecha_apertura": "2021-08-04T03:00:00.000+00:00",  
    "sociedadId": 87  
  }  
]
```

# Beca PPU

Universidad nacional del comahue



404	<pre>{\n    \"statusCode\": 404,\n    \"timestamp\": \"2023-05-12T17:52:26.791+00:00\",\n    \"message\": \"No se encontro sociedad con el cuit\ningresado\",\\n    \"description\": \"uri=/api/sociedad_rubricas/111111111111\"\n}</pre>
400	<pre>{\n    \"statusCode\": 400,\n    \"timestamp\": \"2023-05-12T17:53:44.213+00:00\",\n    \"message\": \"Formato de Cuit Invalido\",\\n    \"description\": \"uri=/api/sociedad_rubricas/3012fsd2\"\n}</pre>
500	<pre>{\n    \"status\": \"error\",\\n    \"result\": {\n        \"error_id\": \"500\",\\n        \"error_msg\": \"Server Internal Error\"\n    }\n}</pre>

# Beca PPU

Universidad nacional del comahue



## Ejemplo Usando Postman

A screenshot of the Postman application interface. The top bar shows a GET request to the URL [https://xroad.jusneuquen.gov.ar/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/api/sociedad\\_rubricas...](https://xroad.jusneuquen.gov.ar/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/api/sociedad_rubricas...). The "Send" button is visible on the right. Below the URL, there are tabs for Params, Authorization, Headers (6), Body, Pre-request Script, Tests, Settings, and Cookies. The Headers tab is selected, showing 8 items. The Body tab is also selected, displaying a JSON array of three objects. Each object has fields: "numero\_clausura" (null), "numero\_apertura" (UNO), "sociedadId" (87), "fecha\_apertura" ("2002-04-02T03:00:00.000+00:00"), and "ru\_Denominacion" (ACTAS DE ASAMBLEAS). The JSON view includes line numbers 1 through 17. The interface also includes a "Pretty" button, "Raw" button, "Preview" button, "Visualize" button, and a "Save Response" button.

# BECA PPU

Universidad nacional del comahue



## Documentacion Web Service de Socios

Servicio Web: “/sociedad\_socios” en X-Road

Url de la fuente Autentica

<https://xroad.jusneuquen.gov.ar/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/>

Entorno	Metodo	Service Code	Punto Acceso	Ejemplo
Test	GET	desa	api/sociedad_socios	<a href="https://&lt;url xroad local&gt;/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/desa/sociedad_socios">https://&lt;url xroad local&gt;/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/desa/sociedad_socios</a>
Producción	GET	prod	api/sociedad_socios	<a href="https://&lt;url xroad local&gt;/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/prod/sociedad_socios">https://&lt;url xroad local&gt;/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/prod/sociedad_socios</a>

El cuit se debe enviar a traves de la URL. Luego del punto de acceso va el cuit, sin guiones ni puntos.

Formato	Ejemplo
String	30708012293

# Beca PPU

Universidad nacional del comahue



## Response

Status	Metodo
200	[ { "nombre": "DIEGO EDUARDO", "apellido": "MANFIO", "sexo": null, "denominacion": "SAN JORGE PETROLEUM S.A. ", "participacion": "Acciones", "sociedadId": 87, "numeroDocumento": "23384618", "deno": "" }, { "nombre": "", "apellido": "", "sexo": "", "denominacion": "SAN JORGE PETROLEUM S.A. ", "participacion": "Acciones", "sociedadId": 87, "numeroDocumento": "", "deno": "INGENIERIA SIMA S.A. REPRESENT. POR SILVEIRO ALBERTO MANFIO D.N.I.10.909.318" } ]
404	{ "status": "error", "result": { "error_id": "404", "error_msg": "Cuit ingresado no corresponde a un cuit existente" } }

# Beca PPU

Universidad nacional del comahue



400	{ "status": "error", "result": { "error_id": "400", "error_msg": "Cuit ingresado no es valido, tiene un formato incorrecto" } }
500	{ "status": "error", "result": { "error_id": "500", "error_msg": "Server Internal Error" } }

# Beca PPU

Universidad nacional del comahue



## Ejemplo usando Postman

Poder Judicial / getSociosConSociedad

Save ... Send Cookies </> ? ! ?

GET https://xroad.jusneuquen.gov.ar/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/api/sociedad\_socios/30708012293

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Body Cookies Headers (8) Test Results 200 OK 12 ms 669 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "nombre": "DIEGO EDUARDO",
3   "apellido": "MANFIO",
4   "sexo": null,
5   "denominacion": "SAN JORGE PETROLEUM S.A. ",
6   "participacion": "Acciones",
7   "sociedadId": 87,
8   "deno": "",
9   "numeroDocumento": "23384618"
10 },
11 {
12   "nombre": "",
13   "apellido": "",
14   "sexo": "",
15   "denominacion": "SAN JORGE PETROLEUM S.A. ",
16   "participacion": "Acciones"
17 }
```

# BECA PPU

Universidad nacional del comahue



## Documentacion Web Service de Antecedentes

### Servicio Web: “/sociedad\_antecedentes” en X-Road

Url de la fuente Autentica

<https://xroad.jusneuquen.gov.ar/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/>

Entorno	Metodo	Service Code	Punto Acceso	Ejemplo
Test	GET	desa	api/sociedad_antecedentes	<a href="https://&lt;url xroad local&gt;/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/desa/sociedad_antecedentes">https://&lt;url xroad local&gt;/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/desa/sociedad_antecedentes</a>
Producción	GET	prod	api/sociedad_antecedentes	<a href="https://&lt;url xroad local&gt;/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/prod/sociedad_antecedentes">https://&lt;url xroad local&gt;/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/prod/sociedad_antecedentes</a>

El cuit se debe enviar a traves de la URL. Luego del punto de acceso va el cuit, sin guiones ni puntos.

Formato	Ejemplo
String	30708012293

# BECA PPU

Universidad nacional del comahue



## Response

Status	Metodo
200	[ { "inscripcion": "337/2000", "descripcion": "INSCRIPCIÓN CONTRATO SOCIAL Y ESCRITO QUE DENUNCIA EL DOMICILIO DE LA SEDE SOCIAL (NO TIENE FIRMA CERTIFICADA)", "fecha": "2000-12-29 00:00:00.0", "tipoAntecedente": "Inscripción Contrato Social" }, { "inscripcion": "73/2003", "descripcion": "INSCRIPCIÓN ACTAS, ACTA DE ASAMBLEA GENERAL ORDINARIA-POR LA CUAL SE APRUEBA EL EJERCICIO ECONOMICO N°1-", "fecha": "2003-10-23 00:00:00.0", "tipoAntecedente": "Inscripción Actas" }, { "inscripcion": "85/2003", "descripcion": "INSCRIPCIÓN ACTA DE ASAMBLEA GENERAL ORDINARIA (06/09/02)-POR LA CUAL SE APRUEBA EL EJERCICIO ECONOMICO N°2 Y SE DESIGNA EL DIRECTORIO- Y ACTA DE DIRECTORIO N° 6 (06,09,02 )DISTRIB DE CARGOS-", "fecha": "2003-11-17 00:00:00.0", "tipoAntecedente": "Inscripción Actas" },

# BECA PPU

Universidad nacional del comahue



## Response

```
{  
  "inscripcion": "1/2004",  
  "descripcion": "INSCRIPCIÓN MODIFICACIÓN DE CONTRATO  
SOCIAL, SE MODIFICARON LOS ARTICULOS 6°(CAPITAL), 9°  
(ACCIONES) Y 12°(ADMINISTRACION).",  
  "fecha": "2004-02-11 00:00:00.0",  
  "tipoAntecedente": "Inscripción Modificación de Contrato Social"  
},  
{  
  "inscripcion": "80/2005",  
  "descripcion": "INSCRIPCIÓN ACTAS, ACTA DE ASAMBLEA  
GENERAL ORDINARIA (05/09/03 INCLUYE APROB BCE AL  

```

# Beca PPU

Universidad nacional del comahue



## Response

```
{
  "inscripcion": "86/2017",
  "descripcion": "INSC. Aumentos de capital y modificacion de los Arts. 6°, 9°, 12° y 19° (ref al capital, acciones, administración y cierre de ejercicio respectivamente) - Instrumentos del 24/04/08, 04/07/11, 15/09/11, 03/04/12, 09/04/12, 19/11/14, 22/04/15 y 23/12/15",
  "fecha": "2017-09-07 00:00:00.0",
  "tipoAntecedente": "Inscripción Modificación de Contrato Social"
},
{
  "inscripcion": "103/2018",
  "descripcion": "INSCRIPCION CAMBIO DE DIRECCION DE LA SEDE SOCIAL",
  "fecha": "2018-11-01 00:00:00.0",
  "tipoAntecedente": "Inscripción Actas"
},
{
  "inscripcion": "130/2019",
  "descripcion": "DC--- INSC AUMENTO Y REDUCCION DEL CAPITAL, MODIFICACION DEL ART 6° Y LA APROBACION DE SU TEXTO ORDENADO (ACTAS DEL 19/10/17, 30/10/18, 28/02/19) --- el artículo 6° se refiere al capital--",
  "fecha": "2019-10-29 00:00:00.0",
  "tipoAntecedente": "Inscripción Aumento de Capital"
},
{
  "inscripcion": "5/2021",
  "descripcion": "DC-Inscripción designación de directores (Acta de Asamblea Gral. ordinaria del 07/09/2020)",
  "fecha": "2021-02-12 00:00:00.0",
  "tipoAntecedente": "Inscripción Actas"
},
{
  "inscripcion": "13/2021",
  "descripcion": "DC -Inscripción cambio de dirección de la sede social (Acta de Directorio del 02/10/2019)",
  "fecha": "2021-03-04 00:00:00.0",
  "tipoAntecedente": "Inscripción Actas"
}]
```

## Response

404	<pre>{\n    \"statusCode\": 400,\n    \"timestamp\": \"2023-05-12T18:28:15.647+00:00\", \n    \"message\": \"Formato de Cuit Invalido\", \n    \"description\": \n    \"uri=/api/sociedad_antecedentes/30%207080122%20931\"\n}</pre>
400	<pre>{\n    \"statusCode\": 404,\n    \"timestamp\": \"2023-05-12T18:28:50.478+00:00\", \n    \"message\": \"No se encontro sociedad con el cuit ingresado\", \n    \"description\": \"uri=/api/sociedad_antecedentes/11321321321\"\n}</pre>
500	<pre>{\n    \"status\": \"error\", \n    \"result\": {\n        \"error_id\": \"500\", \n        \"error_msg\": \"Server Internal Error\"\n    }\n}</pre>

# Beca PPU

Universidad nacional del comahue



## Ejemplo usando Postman

Poder Judicial / getAntecedentesConSociedad

GET https://xroad.jusneuquen.gov.ar/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/api/sociedad\_antecedentes/30708012293

Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (8) Test Results 200 OK 119 ms 3.46 KB Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "inscripcion": "337/2000",
3   "tipoAntecedente": "Inscripcion Contrato Social",
4   "descripcion": "INSCRIPCION CONTRATO SOCIAL Y ESCRITO QUE DENUNCIA EL DOMICILIO DE LA SEDE SOCIAL (Nº 337/2000)",
5   "fecha": "2000-12-29 00:00:00.0"
6 },
7 {
8   "inscripcion": "73/2003",
9   "tipoAntecedente": "Inscripcion Actas",
10  "descripcion": "INSCRIPCION ACTAS, ACTA DE ASAMBLEA GENERAL ORDINARIA-POR LA CUAL SE APRUEBA EL EJEI",
11  "fecha": "2003-10-23 00:00:00.0"
12 },
13 {
14   "inscripcion": "85/2003",
15   "tipoAntecedente": "Inscripcion Actas",
16   "descripcion": "INSCRIPCION ACTA DE ASAMBLEA GENERAL ORDINARIA (85/2003)-POR LA CUAL SE APROBAN LOS ESTATUTOS DE LA SOCIEDAD Y SE CONFIRMAN LOS NOMBRAMIENTOS DE LOS DIFERENTES FUNCIONARIOS DE LA SOCIEDAD",
17   "fecha": "2003-10-23 00:00:00.0"
}
```



## Documentacion Web Service de Inscripcion y Sede

Servicio Web: “/sociedad\_inscripcion\_sedesocial” en X-Road

Url de la fuente Autentica

<https://xroad.jusneuquen.gov.ar/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/>

Entorno	Metodo	Service Code	Punto Acceso	Ejemplo
Test	GET	desa	api/sociedad_inscripcion_sedesocial	<a href="https://&lt;url xroad local&gt;/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/desa/api/sociedad_inscripcion_sedesocial">https://&lt;url xroad local&gt;/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/desa/api/sociedad_inscripcion_sedesocial</a>
Produccion	GET	prod	api/sociedad_inscripcion_sedesocial	<a href="https://&lt;url xroad local&gt;/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/prod/api/sociedad_inscripcion_sedesocial">https://&lt;url xroad local&gt;/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/prod/api/sociedad_inscripcion_sedesocial</a>

El cuit se debe enviar a traves de la URL. Luego del punto de acceso va el cuit, sin guiones ni puntos.

Formato	Ejemplo
String	30708012293

# Beca PPU

Universidad nacional del comahue



## Response

Status	Metodo
200	<pre>{   "fechalincripcion": "2000-12-29T03:00:00.000+00:00",   "numero": 337,   "folio_desde": 2004,   "folio_hasta": 2010,   "tomo_numero": "XIII",   "tomo_anio": "2000",   "denominacion": "SAN JORGE PETROLEUM S.A. ",   "cuit": "30708012293",   "fecha_instrumento": "2000-09-06T03:00:00.000+00:00",   "siglas": "S.A.",   "duracion": 99,   "sedesSocialesInfo": [     {       "calle": "HIPOLITO YRIGOYEN",       "numeroSedeSocial": "135",       "observaciones": "TRANSITORIAS (CAMBIO DE DIRECCIÓN POR ACTA DE DIRECTORIO DEL 02/10/2019, INSC. 13/2021)",       "localidad": "NEUQUEN",       "provincia": "NEUQUEN"     }   ] }</pre>

## Response

404	<pre>{\n    "status": "error",\n    "result": {\n        "error_id": "404",\n        "error_msg": "No se encontro sociedad con el cuit ingresado"\n    }\n}</pre>
400	<pre>{\n    "status": "error",\n    "result": {\n        "error_id": "400",\n        "error_msg": "Formato de cuit invalido"\n    }\n}</pre>
500	<pre>{\n    "status": "error",\n    "result": {\n        "error_id": "500",\n        "error_msg": "Server Internal Error"\n    }\n}</pre>



## Ejemplo usando Postman

GET [https://xroad.jusneuquen.gov.ar/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/api/sociedad\\_inscripcion\\_sedesocial/30708012293](https://xroad.jusneuquen.gov.ar/r1/OPTIC/GOB/GOB00014/GP-PODERJUDICIAL/api/sociedad_inscripcion_sedesocial/30708012293) Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (8) Test Results 200 OK 37 ms 770 B Save Response

Pretty Raw Preview Visualize JSON

```
1 "fechaIncripcion": "2000-12-29T03:00:00.000+00:00",
2 "numero": 337,
3 "folio_desde": 2004,
4 "folio_hasta": 2010,
5 "tomo_numero": "XIII",
6 "tomo_anio": "2000",
7 "denominacion": "SAN JORGE PETROLEUM S.A. ",
8 "cuit": "30708012293",
9 "fecha_instrumento": "2000-09-06T03:00:00.000+00:00",
10 "siglas": "S.A.",
11 "duracion": 99,
12 "sedesSocialesInfo": [
13     {
14         "calle": "HIPOLITO YRIGOYEN",
15         "numeroSedeSocial": "135",
16         "observaciones": "TRANSPORTADAS /CAMRTO DE DIRECCION POR ACTA DE DIRECTORIO DEL 02/10/2010 TNCC"
17     }
]
```