10/MAYO/2022

Servicio de contratos de electricidad

Integrantes:

Bryan Alberto Duran Cuellar

Pablo Lucas Negrete

Luis Armando Villa Ceballos

Iván Antonio García Campos

Equipo no.4

Experiencia educativa: Integración de soluciones

Profesor: José Rafael Rojano Cáceres

ÍNDICE

Introducción

El diseño del software tiende a ser cada vez más modular. Los Servicios Web nos permitirán distribuir nuestra aplicación a través de Internet, pudiendo una aplicación utilizar los servicios ofrecidos por cualquier servidor conectado a Internet. Y para el desarrollo de estos servicios web se hacen uso de tecnologías como SOAP y REST que son dos enfoques distintos para la transmisión de datos, ambos definen como diseñar interfaces de programación de aplicaciones (API).

Para la realización de este proyecto se hará uso de tecnologías SOAP y REST para realizar los servicios que haremos, también se hará uso del lenguaje de programación Java, y finalmente Docker para contenerizar nuestros servicios.

Motivación

Nuestra motivación es el pensar en el resultado que este proyecto puede darnos en muchos factores, en primer lugar, por la experiencia que estaremos adquiriendo aunada a un montón de conocimiento que ganaremos además del enseñado por nuestro profesor; misma experiencia que nos será de gran ayuda en nuestra vida laboral. También nos dará una idea definida acerca de la manera en que están hechos muchos de los servicios que utilizamos y consumimos cotidianamente como lo son los servicios web a disposición de la población, los cuales facilitan acciones necesarias que antes conllevaban una inversión de tiempo mayor por lo tanto ofrecen una facilidad y ahorro de tiempo entre otros recursos.

Problemática

La globalización, el aumento del número y velocidad de las transacciones y la movilidad, provocados por la rápida evolución de la tecnología han dejado obsoleta la forma de atender los negocios. Para un gran número de empresas es casi un «obligación» ofrecer servicios web y cada vez más frecuentemente las empresas nacen con el objetivo único de que sus servicios se consulten en internet. Las ventajas son múltiples, destacando el aumento de clientes potenciales, la ausencia de horario comercial y la comodidad de poder ofrecer a nuestros clientes administrar su servicio desde cualquier ubicación. Ahora bien, todos estos beneficios se ponen en ejecución para ofrecer a los clientes los medios adecuados para que las contrataciones se realicen lo más rápidas, seguras y sencillas posibles, así como la administración de sus servicios.

Solución

Como solución pensamos en desarrollar un servicio web para la creación de contratos y para el pago de servicios de luz que se lleva a cabo bimestralmente, con la finalidad de reducir los tiempos de espera en lugares físicos y hacerlo de forma más rápida y segura en el servicio.

Donde se pretende que los personas cuenten con las siguientes operaciones:

Operaciones por SOAP

- Agregar contrato: La persona proporciona su nombre, domicilio, teléfono y curp. Entonces el contrato será creado, y se le proporcionará su número de contrato y su firma electrónica (token). La firma es generada gracias al consumo de nuestro microservicio de firmas.

- Cancelar Servicio: La persona proporciona su número de contrato, nombre, domicilio, teléfono, firma electrónica y curp, entonces el sistema validará si la información es correcta para proceder con la operación. La validación de la firma es gracias a la operación validar de nuestro microservicio de firmas.

- Consultar Servicio: La persona proporciona su número de contrato, entonces el sistema retornara sus datos ingresados al crear el contrato.

- Modificar Servicio: La persona proporciona sus datos originales, después ingresa los datos que modificará, el microservicio de firmas valida que los primeros datos sean correctos para proceder con la modificación de los datos.

- Mostrar contratos: Para esta operación no es necesario proporcionar datos, el sistema retornara la lista de contratos creados, con datos como el nombre, número de contrato y la firmae (Esta operación no es para los clientes, es para nosotros como desarrolladores para poder ver todos los contratos dados de alta).

Operaciones por REST

- firmar: Esta operación es de tipo POST, donde le es enviado el objeto Contrato. Al crear un contrato se genera una firma que servirá para autenticar otras operaciones que desee realizar con su contrato.

- Validar: Esta operación es de tipo POST, donde le es enviado el objeto Contrato. Al realizar operaciones como Cancelar o Modificar esta operación servirá para validar los datos ingresados y permitir la operación o negarla.

- Historial de pago: La persona proporciona su número de contrato en el siguiente enlace **https://microservicio-pago.herokuapp.com/pagos/contrato/nContrato** .En vez de nContrato ira el número de contrato proporcionado al crear el contrato en Contratos WSDL

- Pagar: La persona proporciona el monto a pagar, fecha y el número de contrato, esta operación es enviada por POST, por lo que se debe usar POSTMAN.

Costos

Se realizo la investigación de distintos servicios de hosting, que también contaran con compatibilidad con Docker, y encontramos los siguientes:

Heroku

Heroku es una plataforma basada en la nube como servicio (PaaS) diseñada para que los desarrolladores y equipos creen, envíen, supervisen y escalen aplicaciones modernas. Esta plataforma basada en contenedores brinda a los desarrolladores más tiempo para centrarse en el producto principal sin tener que preocuparse de mantener la infraestructura de la aplicación. Heroku ofrece herramientas, servicios y flujos de trabajo integrados para ayudar a las organizaciones de todos los tamaños a maximizar la productividad individual y del equipo para poder entregar aplicaciones en el mercado más rápidamente.

Y cabe destacar que Heroku permite usarse totalmente gratis, aunque es obvio que al usarlo gratuitamente no se contara con las mismas características si se cuenta con algún plan de pago. Un plan gratuito ofrece:

- RAM: 512 MBs.
- Permite el despliegue desde Git.
- 2 procesadores.
- Dominios personalizados.

Fuente: Pricing | Heroku (https://www.heroku.com/pricing)

Clever Cloud

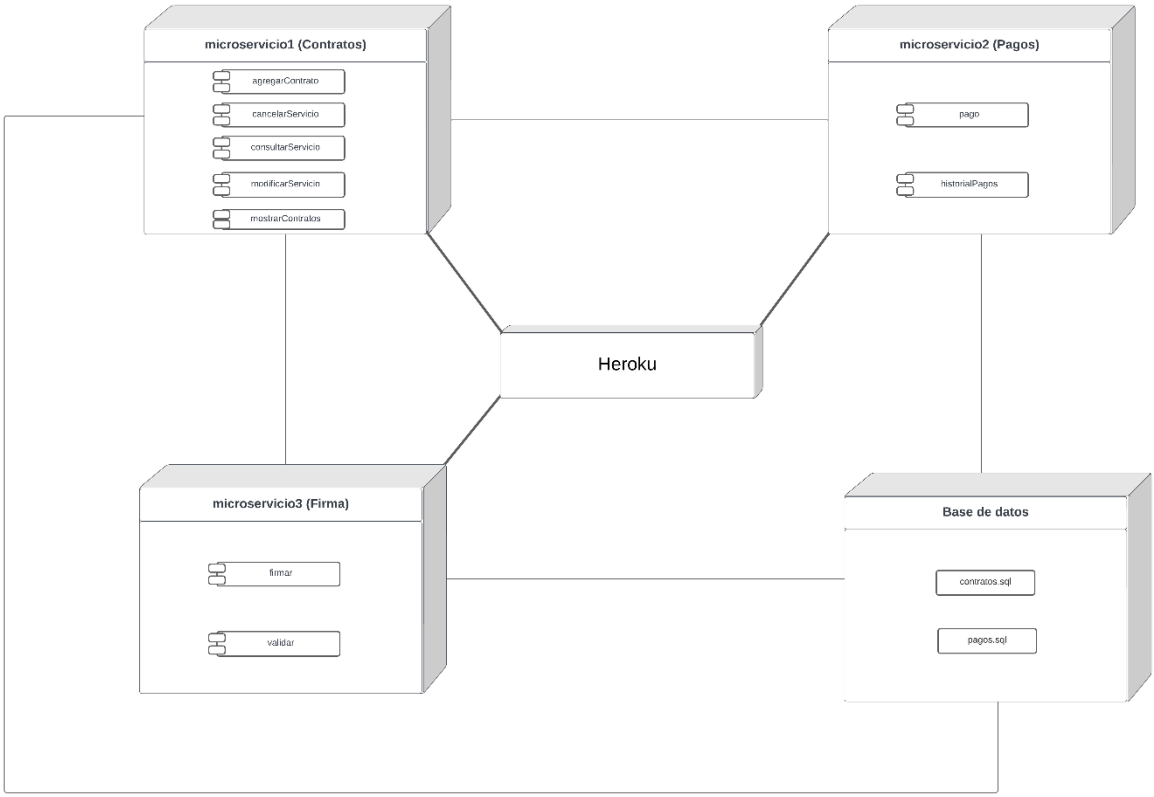
Clever Cloud es una plataforma como servicio que ha sido diseñada para ayudar a las organizaciones a operar, automatizar y ampliar sus negocios con diversos tiempos de ejecución y complementos. Permite a los equipos obtener información general del estado actual de los Scalers y la actividad presente de la memoria RAM, la CPU, el disco y la red. Entre las principales funciones de Clever Cloud, figuran la administración de aplicaciones, redimensionamiento de aplicaciones, migración de bases de datos, nombres de dominio y enlaces de implementación.

Clever Cloud será usado para que los registros que se hagan permanezcan, debido a que en Heroku tras un periodo de inactividad se reinicia, y no los datos no persisten. El plan básico de Clever Cloud de base de datos MySQL es el siguiente:

	PLAN NAME	BACKUPS	LOGS	MAX CONNECTION LIMIT	MAX DB SIZE	MEMORY	METRICS	TYPE	VCPUS	PRICE
○	DEV	Daily - 7 Retained	No	5	10 MB	Shared	No	Shared	Shared	0.00 €

Fuente Pricing – Clever Cloud(https://www.clever-cloud.com/pricing/#databases)

Modelo de despliegue

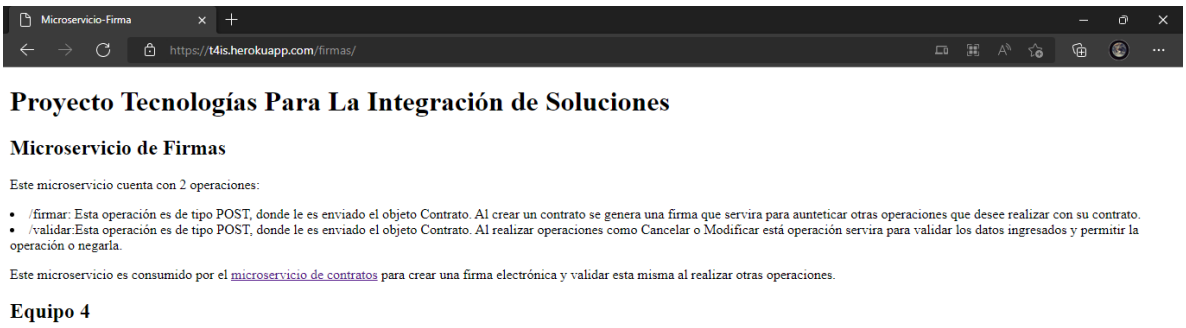


Documentación del proxy

Un servidor proxy inverso se sitúa delante de los servidores web y reenvía las solicitudes de los clientes a esos servidores web. Los recursos solicitados se devuelven entonces al cliente, y parece como si se hubieran originado en el propio servidor proxy.

Firmas

https://t4is.herokuapp.com/firmas/



Ambas operaciones de firmas son de tipo POST, pero estas son consumidas por el microservicio de contrato.

Consumo al añadir contrato de la operación firmar

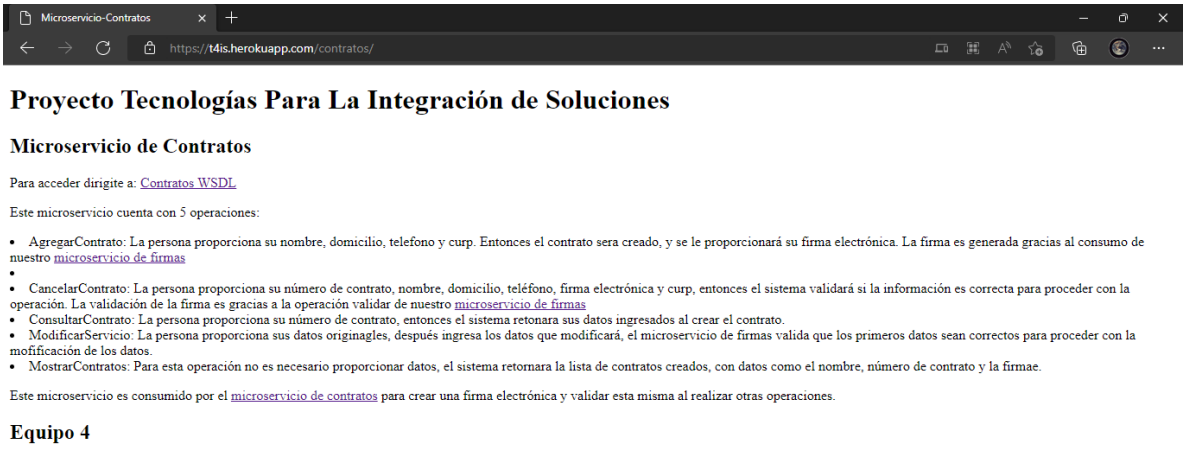
```
String url = "https://t4is.herokuapp.com/firmas/firmar";
ResponseEntity<Contrato> contratoRespuesta = restTemplate.postForEntity(url, contratoGuardado,
        Contrato.class);
String firmae = contratoRespuesta.getBody().getFirmae();
contrato.setFirmae(firmae);
AgregarContratoResponse.Contrato contratoResponse = new AgregarContratoResponse.Contrato();
```

Consumo al modificar y cancelar contrato de la operación valiar

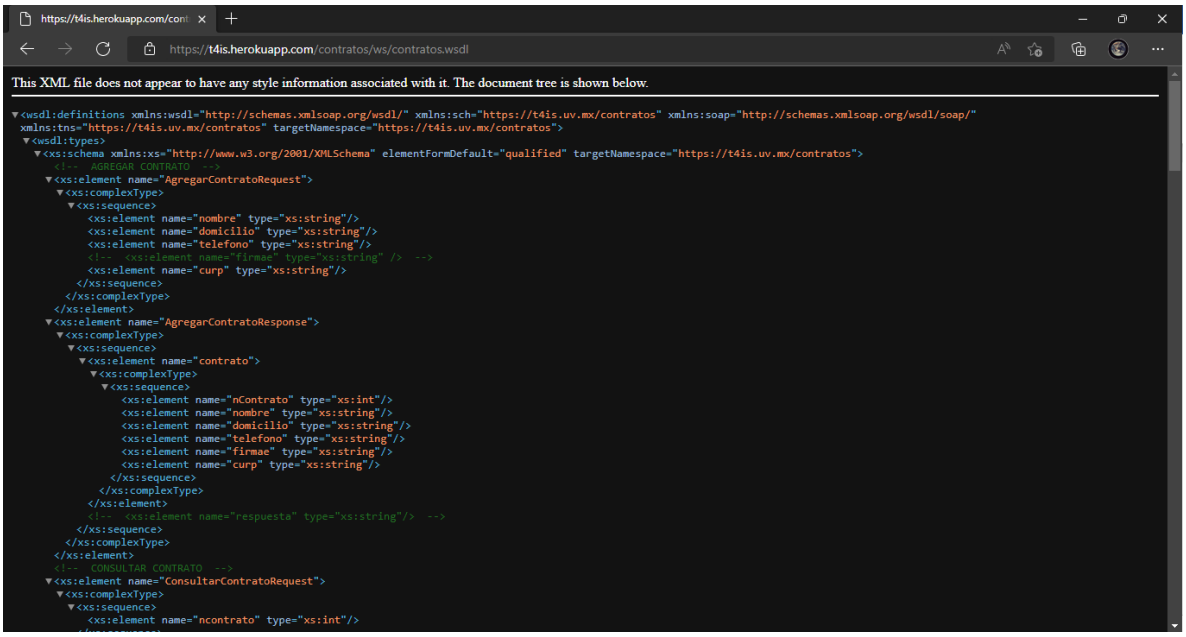
```
String url = "https://t4is.herokuapp.com/firmas/validar";
ResponseEntity<Boolean> esValidoRespuesta = restTemplate.postForEntity(url, contratoABorrar, Boolean.class);
boolean esValido = esValidoRespuesta.getBody();

//
if (!esValido) {
    respuesta.setRespuesta("No se ha logrado validar su contrato.");
    return respuesta;
}

//String url = "https://microservicio-firma.herokuapp.com/validar";
String url = "https://t4is.herokuapp.com/firmas/validar";
ResponseEntity<Boolean> esValidoRespuesta = restTemplate.postForEntity(url, contratoAModificar, Boolean.class);
boolean esValido = esValidoRespuesta.getBody();
```

Contratos

https://t4is.herokuapp.com/contratos/



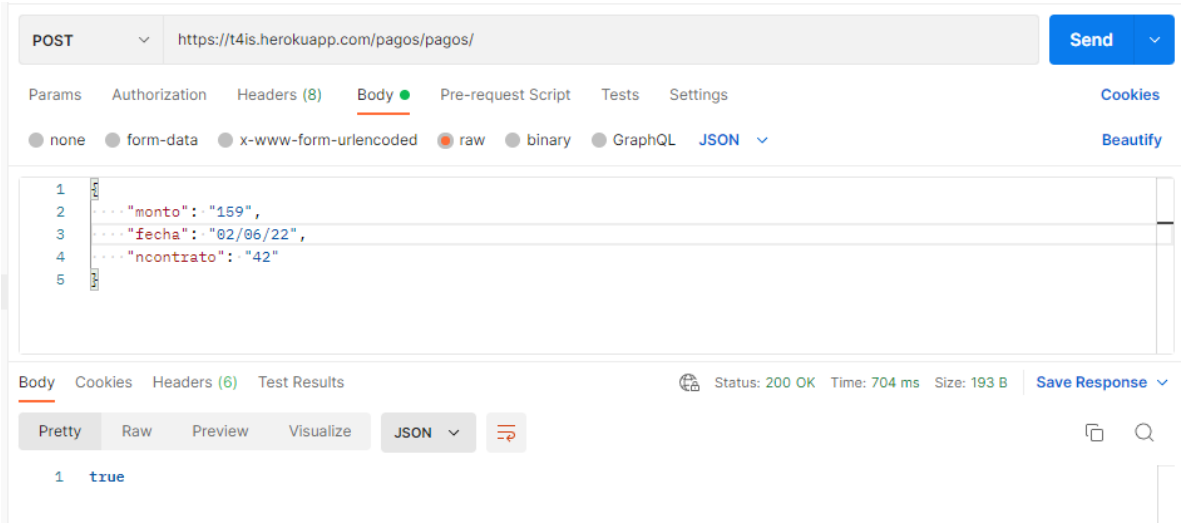
Accediendo al WSDL a través del proxy



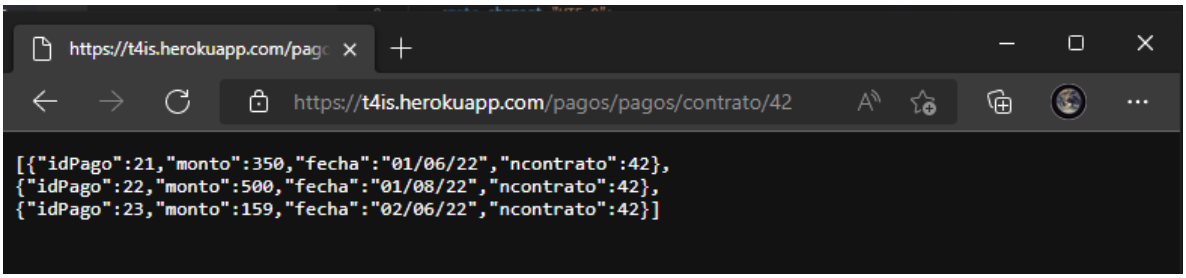
Pagos

https://t4is.herokuapp.com/pagos/

Accediendo a la consulta de pagos, esto se debe hace desde postman debido a que es una solicitud de tipo POST.



Consultándolo desde el proxy



Documentación del API SOAP y REST

Los endpoints son las URLs de un API o un backend que responden a una petición. Para la realización de nuestro proyecto proponemos los siguientes endpoints.

Endpoint SOAP

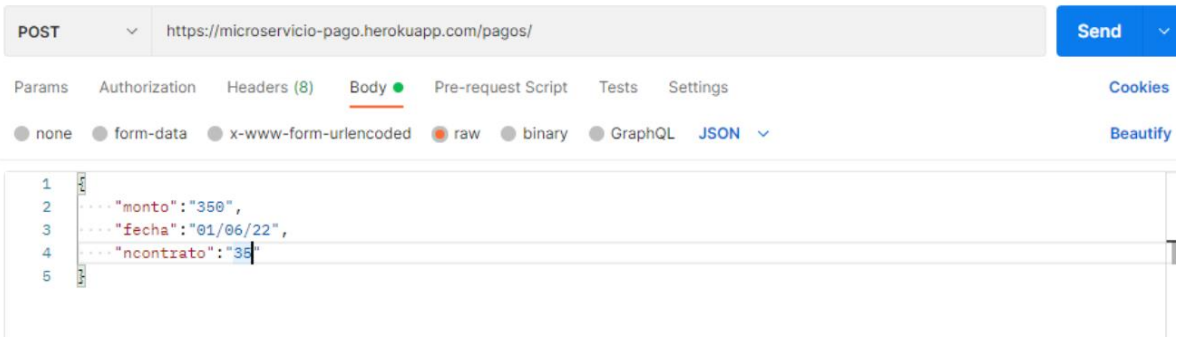
Este es el primer microservicio donde se efectuarán las operaciones del servicio.

Debido a que son realizados por SOAP estas son mostradas desde el wsdl, con ayuda de la extensión de Wizdler.

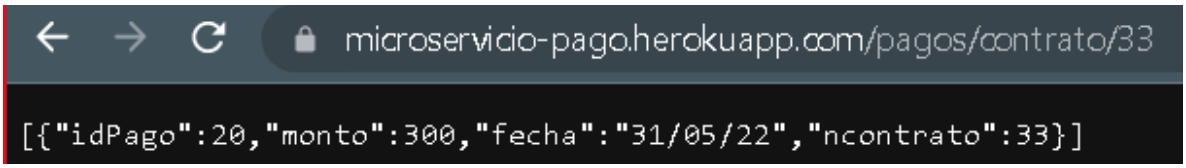
- AgregarContrato
- CancelarServicio
- ConsultarContrato
- ModificarServicio
- MostrarContratos

Endpoint REST

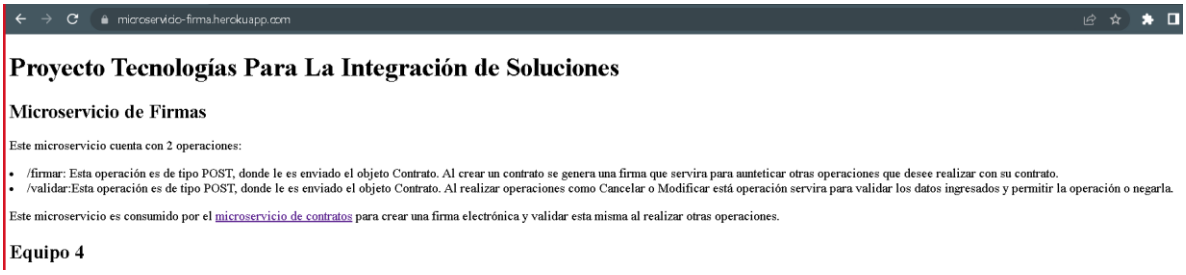
Este segundo microservicio (Pagar) es realizado por REST, mandamos el monto, la fecha y el ncontrato al que se le hará el pago correspondiente y se hará el registro a la base de datos.



Además, en este segundo microservicio podemos consultar un historial de pagos que se han realizado en ese contrato para ello solo se necesita de su número de contrato.



Este tercer microservicio crea una firma electrónica (token) y un numero de contrato para poder efectuar operaciones como agregar, modificar o cancelar el contrato.



Parámetros de recepción

Los parámetros de recepción serán datos que el cliente proporcionará. Además, habrá distintos tipos de datos que se recibirán, para los cuales dicha recepción será estricta y no procederá la acción del servicio si falta algún dato por proporcionar o no cumple con las características requeridas. Los parámetros de recepción serán los siguientes:

SOAP

- AgregarContrato: El cliente proporcionara su nombre, domicilio, número telefónico y su curp para generar una firma electrónica (token) y con ella un numero de contrato.

POST https://microservicio-contrato.herokuapp.com:443/ws

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
    <Body>
        <AgregarContratoRequest xmlns="https://t4is.uv.mx/contratos">
            <nombre>Bryan duran</nombre>
            <domicilio>ursulo galvan</domicilio>
            <telefono>22823224322</telefono>
            <curp>ABCDE12345678</curp>
        </AgregarContratoRequest>
    </Body>
</Envelope>
```

- CancelarServicio: El cliente proporcionara sus datos proporcionados como ncontrato, nombre, domicilio, teléfono, curp y la firma electrónica que se le otorgo, de esa forma podemos hacer la cancelación del contrato.

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
    <Body>
        <CancelarServicioRequest xmlns="https://t4is.uv.mx/contratos">
            <contrato>
                <ncontrato>38</ncontrato>
                <nombre>Bryan negrete</nombre>
                <domicilio>centro</domicilio>
                <telefono>2282324343</telefono>
                <firmae>eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwOi8vZmlyYW1hcy51di5teC8iLCJjb250cmF0byI6eyJuY29udHJhdG8iOjM4LCJub21icmUiOiJCcnlhbiBuZWdyZXRlIiwiZG9taWNpbGlvIjoi
                    Y2VudHJvIiwidGVsZWZvbm8iOiIyMjgyMzI0MzQzIiwiZmlybWFlIjpudWxsLCJjdXJwIjoiQUJDREUxMjM0NTY3OCJ9fQ.vZBPhcAAG2LXB7sFakvJGRmVzL0kFtKZW1-lYTRrPts</firmae>
                <curp>ABCDE12345678</curp>
            </contrato>
        </CancelarServicioRequest>
    </Body>
</Envelope>
```

- ConsultarContrato: El cliente proporcionara su número de contrato.

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
    <Body>
        <ConsultarContratoRequest xmlns="https://t4is.uv.mx/contratos">
            <ncontrato>28</ncontrato>
        </ConsultarContratoRequest>
    </Body>
</Envelope>
```

Muestra los datos del contrato, obviamente no muestra la firma electrónica por seguridad de los usuarios.

POST https://microservicio-contrato.herokuapp.com:443/ws

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns2:ConsultarContratoResponse xmlns:ns2="https://t4is.uv.mx/contratos">
            <ns2:contratos>
                <ns2:ncontrato>32</ns2:ncontrato>
                <ns2:nombre>Prueba</ns2:nombre>
                <ns2:domicilio>Prueba</ns2:domicilio>
                <ns2:telefono>123456789</ns2:telefono>
            </ns2:contratos>
        </ns2:ConsultarContratoResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

- ModificarServicio: El cliente proporcionara su número de contrato, firma electrónica y sus datos, seguido de los datos a actualizar; nombre, domicilio, y teléfono.

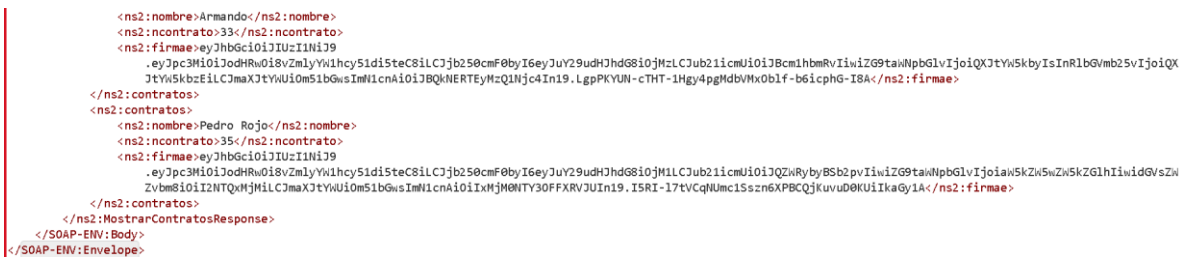
```
POST  https://microservicio-contrato.herokuapp.com:443/ws  Go

<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
    <Body>
        <ModificarServicioRequest xmlns="https://t4is.uv.mx/contratos">
            <contratoAModificar>
                <ncontrato>37</ncontrato>
                <nombre>Bryan duran</nombre>
                <domicilio>ursulo galvan</domicilio>
                <telefono>2282324356</telefono>
                <firmae>eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwOi8vZmlyYW1hcy51di5teC8iLCJjb250cmF0byI6eyJuY29udHJhdG8iOjM3LCJub21icmUiOiJCcnlhbiBkdXJhbiIsImRvbWljaWxpbyI6InVy
                    c3VsbyBnYWx2YW4iLCJ0ZWxlZm9ubyI6IjIyODIzMjQzNTYiLCJmaXJtYWUiOm51bGwsImN1cnAiOiJBQkNERTEyMzQ1Njc4In19.rjrnvMg5YFH9RA-Xv0nYFawFRR9eGjjG1ZNqCwOq3So</firmae>
                <curp>ABCDE12345678</curp>
            </contratoAModificar>
            <contratoNuevo>
                <nombre>Bryan duran cuellar</nombre>
                <domicilio>colonia centro</domicilio>
                <telefono>2282556678</telefono>
            </contratoNuevo>
        </ModificarServicioRequest>
    </Body>
</Envelope>
```

- Aquí ya está modificado.

```
POST  https://microservicio-contrato.herokuapp.com:443/ws  Go

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns2:ModificarServicioResponse xmlns:ns2="https://t4is.uv.mx/contratos">
            <ns2:contrato>
                <ns2:nContrato>37</ns2:nContrato>
                <ns2:nombre>Bryan duran cuellar</ns2:nombre>
                <ns2:domicilio>colonia centro</ns2:domicilio>
                <ns2:telefono>2282556678</ns2:telefono>
                <ns2:firmae>eyJhbGciOiJIUzI1NiJ9
                    .eyJpc3MiOiJodHRwOi8vZmlyYW1hcy51di5teC8iLCJjb250cmF0byI6eyJuY29udHJhdG8iOjM3LCJub21icmUiOiJCcnlhbiBkdXJhbiBjdWVsbGFyIiwiZG9taWNpbGlvIjoiY29sb25pYSBjZW50
                    cm8iLCJ0ZWxlZm9ubyI6IjIyODI1NTY2NzgiLCJmaXJtYWUiOm51bGwsImN1cnAiOiJBQkNERTEyMzQ1Njc4In19.ChvU6jeqGh7V0Viwxydvwa NDYZ9sX0L8HdDm4cIOyOc</ns2:firmae>
                <ns2:curp>ABCDE12345678</ns2:curp>
            </ns2:contrato>
        </ns2:ModificarServicioResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

- MostrarContratos: Para esta operación no es necesario proporcionar datos y es una operación para nosotros los desarrolladores para ver que se efectúan las demás operaciones.

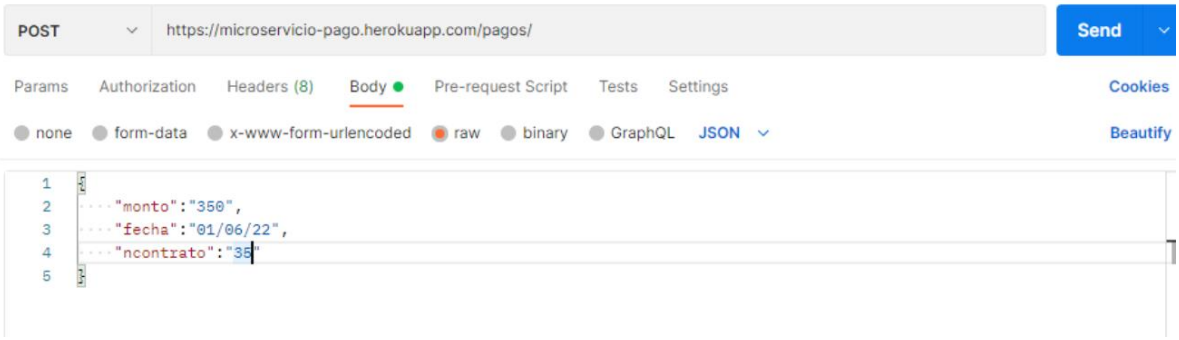
```
POST  https://microservicio-contrato.herokuapp.com:443/ws

<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
    <Body>
        <MostrarContratosRequest xmlns="https://t4is.uv.mx/contratos">[any]</MostrarContratosRequest>
    </Body>
</Envelope>
```



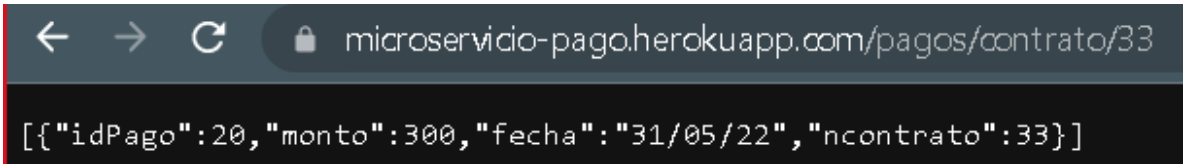
```
            <ns2:nombre>Armando</ns2:nombre>
            <ns2:ncontrato>33</ns2:ncontrato>
            <ns2:firmae>eyJhbGciOiJIUzI1NiJ9
                .eyJpc3MiOiJodHRwOi8vZmlyYW1hcy51di5teC8iLCJjb250cmF0byI6eyJuY29udHJhdG8iOjMzLCJub21icmUiOiJBcm1hbmRvIiwiZG9taWNpbGlvIjoiQXJtYW5kbyIsInRlbGVmb25vIjoiQX
                JtYW5kbzEiLCJmaXJtYWUiOm51bGwsImN1cnAiOiJBQkNERTEyMzQ1Njc4In19.LgpPKYUN-cTHT-1Hgy4pgMdbVMxOblf-b6icphG-I8A</ns2:firmae>
        </ns2:contratos>
        <ns2:contratos>
            <ns2:nombre>Pedro Rojo</ns2:nombre>
            <ns2:ncontrato>35</ns2:ncontrato>
            <ns2:firmae>eyJhbGciOiJIUzI1NiJ9
                .eyJpc3MiOiJodHRwOi8vZmlyYW1hcy51di5teC8iLCJjb250cmF0byI6eyJuY29udHJhdG8iOjM1LCJub21icmUiOiJQZWRybyBSb2pvIiwiZG9taWNpbGlvIjoiaW5kZW5wZW5kZGlhIiwidGVsZW
                Zvbm8iOiI2NTQxMjMiLCJmaXJtYWUiOm51bGwsImN1cnAiOiIxMjM0NTY3OFFXRVJUIn19.I5RI-l7tVCqNUmc1Sszn6XPBCQjKuvuD0KUiIkaGy1A</ns2:firmae>
        </ns2:contratos>
    </ns2:MostrarContratosResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

REST

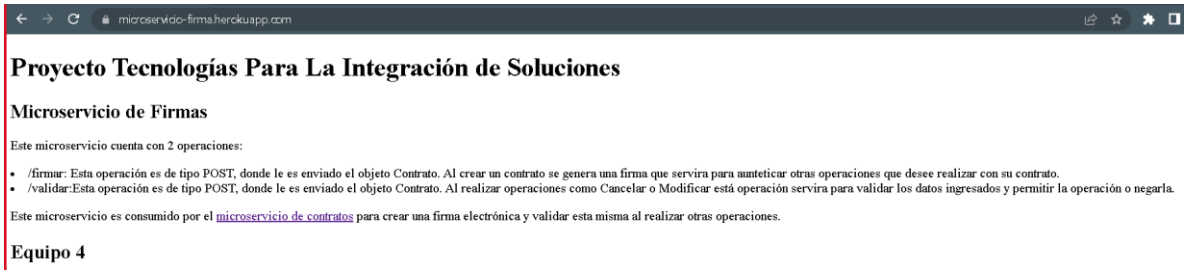
- Pagar: Aquí si es necesario mandar los datos como el monto, la fecha y el número de contrato al cual se le va a asignar



- ConsultarHistoriaPagos: En esta operación solo mandamos el número de contrato para que nos traiga el historial de todos los pagos que se han realizado con anterioridad.



- Este tercer microservicio crea una firma electrónica (token) y un numero de contrato para poder efectuar operaciones como agregar, modificar o cancelar el contrato.



Parámetros devueltos

Los parámetros que devolverá el servicio serán dependiendo de qué operación esté siendo usada, los parámetros que se retornarán serán los datos en la base de datos en la mayoría de los casos.

SOAP

- AgregarContrato: Retornará un mensaje de éxito, con el nombre del que realizó el contrato. Si algún dato se encuentra vacío retornara un mensaje de error.

- CancelarServicio: Retornara un mensaje de éxito, si los datos del contrato así como su firma electrónica es valida.

POST https://microservicio-contrato.herokuapp.com:443/ws

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns2:CancelarServicioResponse xmlns:ns2="https://t4is.uv.mx/contratos">
            <ns2:respuesta>Has cancelado con éxito tu servicio. Adios.</ns2:respuesta>
        </ns2:CancelarServicioResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

- ConsultarContrato: Retornara los datos del cliente; nContrato, nombre, domicilio, teléfono.

POST https://microservicio-contrato.herokuapp.com:443/ws

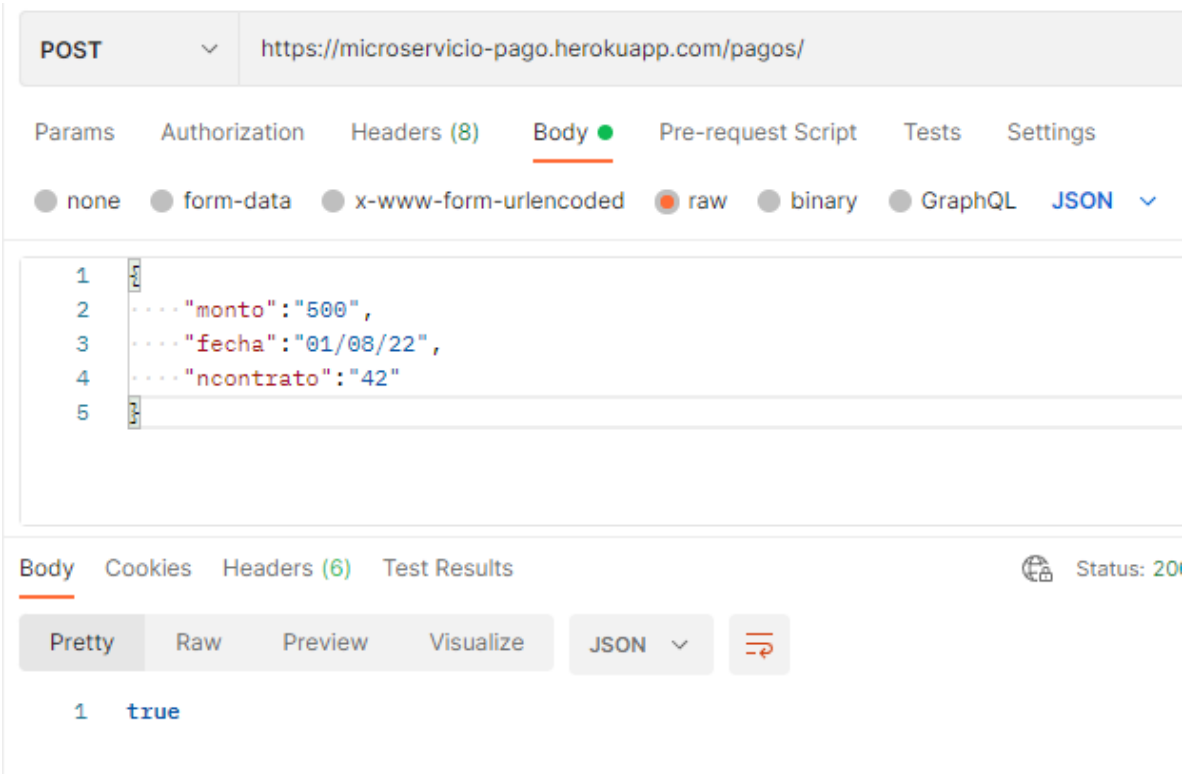
```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns2:ConsultarContratoResponse xmlns:ns2="https://t4is.uv.mx/contratos">
            <ns2:contratos>
                <ns2:ncontrato>28</ns2:ncontrato>
                <ns2:nombre>Pedro Juan Jun</ns2:nombre>
                <ns2:domicilio>av 15</ns2:domicilio>
                <ns2:telefono>123456</ns2:telefono>
            </ns2:contratos>
        </ns2:ConsultarContratoResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

- ModificarServicio: Se introducen los datos validos del contrato incluyendo firma, ncontrato. Se introducen los nuevos valores y nos devuelve el contrato modificado con una nueva firma para efectuar las demás operaciones.

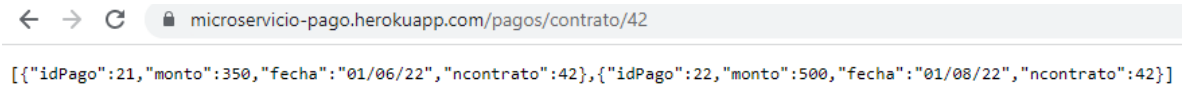
- MostrarContratos: Esta operación retornara el nombre de las personas, su número de contrato y firma electrónica de todas las personas que hayan realizado su contrato.

REST

GenerarPago: Esta operación retornara verdadero si se cumple la operación y los datos los almacena en la tabla pagos de la base de datos.



ConsultarHistorialPago: Esta operación retornara un arreglo con los pagos realizados con ese contrato.



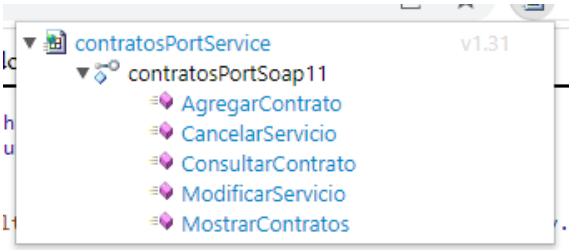
Proxy

Plan de pruebas

El plan de prueba describe el ámbito del esfuerzo de prueba general y proporciona un registro del proceso de planificación de prueba.

SOAP

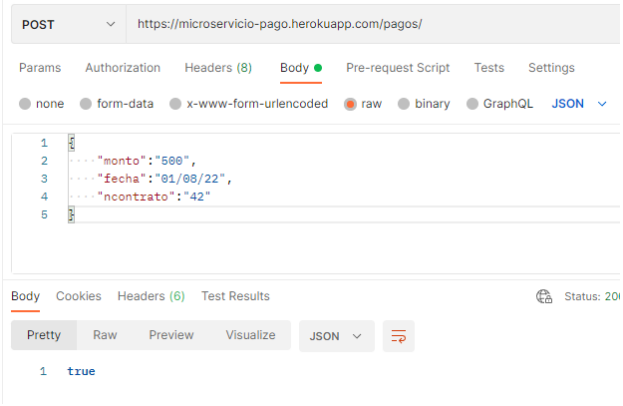
Para probar las 5 operaciones hechas en SOAP se usará la extensión para los navegadores Wizdler .Esta es una herramienta, desarrollada con el fin de realizar pruebas a aplicaciones con arquitectura orientada a servicio y transferencia de estado representacional (SOAP y REST).



Y ya usando la extensión, se da click sobre las operaciones disponibles.

REST

Para probar estas operaciones hechas en REST se necesitan poner los datos en la URL para poder mandarlos a la base de datos, así como también poder recibir.



Repositorio en GitHub

El repositorio donde se alojará el proyecto será el siguiente:

https://github.com/BryanDuran/CFE

Documentación en los archivos Docker file y Docker-compose.yml

Para nuestro proyecto se optó por usar Docker files.

Microservicio 1

El microservicio 1 se encuentra disponible: https://microservicio-contrato.herokuapp.com/ws/contratos.wsdl

Explicación del Docker File,

1. Se inicializa la imagen base para las instrucciones subsecuentes en ente caso se usará la imagen de JDK8.
2. Se establece el directorio de trabajo; /app.
3. CMD se encarga de pasar el comando al contenedor, en este caso ejecuta el archivo *ejecutar.sh.*
4. ADD se encarga de copiar el archivo JAR de la carpeta app local a la carpeta app de la imagen.
5. Lo mismo pasa con ejecutar, es copiada a la carpeta app de la imagen.
6. Finalmente ejecuta el archivo ejecutar.sh, dentro del archivo ejecutar.sh, se establece el puerto donde se ejecutará, debido a que expose no funciona al hacer el deploy con heroku.

El microservicio 1 contiene todas las operaciones del SOAP, previamente mencionadas.

```
1   From rrojano/jdk8
2   workdir /app
3   cmd ["/app/ejecutar.sh"]
4   add app/ContratosCFE-0.0.1-SNAPSHOT.jar /app/ContratosCFE-0.0.1-SNAPSHOT.jar
5   add ejecutar.sh /app/ejecutar.sh
6   run chmod 755 /app/ejecutar.sh
```

Ejecutar.sh

```
/usr/bin/java -jar -Dserver.port=$PORT ContratosCFE-0.0.1-SNAPSHOT.jar
```

Microservicio 2 y 3

El microservicio 2 se encuentra disponible: https://microservicio-pago.herokuapp.com/

El microservicio 3 se encuentra disponible: https://microservicio-firma.herokuapp.com/

Ambos microservicios funcionan igual que el 1, sus dockerfiles son iguales, su único cambio son los archivos JAR generados.

.

Proxy

El Proxy se encuentra disponible: http://t4is.herokuapp.com/

Explicación del Docker File:

1. El comando *FROM nginx* refiere al uso de una imagen como base.
2. Se añade el archivo *proxy.conf* en la carpeta proxy,
3. Se añade el archivo *index.html* en la misma carpeta,
4. El comando *CMD* realiza la búsqueda del puerto en Heroku y lo pasa al archivo *.conf*.

```
FROM nginx
add proxy/proxy.conf /etc/nginx/conf.d/default.conf
add proxy/index.html /etc/nginx/html/index.html
CMD sed -i -e 's/$PORT/'"$PORT"'/g' /etc/nginx/conf.d/default.conf && nginx -g 'daemon off;'
```